101LABS®
Wireshark WCNA

101LABS
.net

LEARN • BY • DOING

Paul Browning

101LABS®
Wireshark WCNA

LEARN • BY • DOING

Paul Browning

Table of Contents

https://www.101labs.net

ISBN-13: 978-0-9928239-4-8

Published by:

Reality Press Ltd.

About the Author

Paul Browning

Paul Browning worked as a police officer in the UK for 12 years before changing careers and becoming a helpdesk technician. He passed several IT certifications and began working for Cisco Systems doing WAN support for large enterprise customers.

He started an IT consulting company in 2002 and helped to design, install, configure, and troubleshoot global networks for small to large companies. He started teaching IT courses soon after that, and through his classroom courses, online training, and study guides has helped tens of thousands of people pass their IT exams and enjoy successful careers in the IT industry.

In 2006, Paul started the online IT training portal **www.howtonetwork.com**, which has grown to become one of the leading IT certification websites.

In 2013, Paul moved to Brisbane with his family. In his spare time, he plays the guitar, reads, drinks coffee, and practices Brazilian jiu-jitsu.

Introduction–101 Labs

Welcome to your 101 Labs book.

When I started teaching IT courses back in 2002, I was shocked to discover that most training manuals were almost exclusively dedicated to theoretical knowledge. Apart from a few examples of commands to use or configuration guidelines, you were left to plow through without ever knowing how to apply what you learned to live equipment or to the real world.

Fast forward 16 years and little has changed. I still wonder how, when around 50% of your exam marks are based on hands-on skills and knowledge, most books give little or no regard to equipping you with the skills you need to both pass the exam and then make money in your chosen career as a network, security, or cloud engineer (or whichever career path you choose).

101 Labs is NOT a theory book: it’s here to transform what you have learned in your study guides into valuable skills you will be using from day one on your job as a network engineer. I don’t teach DHCP, for example; instead, I show you how to configure a DHCP server, which addresses you shouldn’t use, and which parameters you can allocate to hosts. If the protocol isn’t working, I show you what the probable cause is. Sound useful? I certainly hope so.

I choose the most relevant parts of the exam syllabus and use free software or free trials to walk you through configuration and troubleshooting commands step by step. As your confidence grows, I increase the difficulty level. If you want to be an exceptional IT engineer, you can make your own labs up, add other technologies, try to break them, fix them, and do it all over again.

101 Labs—Wireshark WCNA

There are so many IT career paths to choose from, network administration, security, wireless engineer, DevOps, service provider, data center, helpdesk, design, to name just a few. One common thread which joins them all is the requirement to understand how IP traffic crosses your network.

Capturing traffic is often referred to as sniffing. It's a vital skill for any network engineer because it allows you to:

- Baseline your network
- Troubleshoot faults
- Quickly find the root cause of issues
- Detect network attacks
- Find protocol issues with DHCP, DNS, HTTP, Email, and more
- Troubleshoot voice, video, and wireless issues

Being able to capture traffic will help you quickly find the root cause of issues so you can either resolve them yourself or present your findings to customers or managers. You will often be asked to sniff network traffic as part of your day-to-day job.

Wireshark is the premier network sniffing tool available on the market. Passing the Wireshark Certified Network Analyst exam will make you stand out from your colleagues because it proves you have advanced troubleshooting skills and a deep understanding of TCP/IP.

101 Labs—Wireshark WCNA will give you a solid foundation and help you prepare for and pass the exam. Please use these labs as a starting point. We add suggestions to the end of most of the labs, so please make your own labs up and explore every menu item, graph, and packet type you can.

We presume you already have a good understanding of TCP/IP. If you need to improve your knowledge, then please read a good quality CompTIA Network+ book, check out our video course for the Network+ on howtonetwork.com and also please read our 101 Labs—CompTIA Network+ book which will be a great complement to this one. We also teach Wireshark from scratch on **www.howtonetwork.com** .

Instructions

1. Please follow the labs from start to finish. If you get stuck, do the next lab and come back to the problem lab later. There is a good chance you will work out the solution as you gain confidence and experience in using the method.
2. Wireshark regularly updates its software, so don't worry too much if your output differs or the menu options don't match exactly. Some features are only available for specific platforms such as Mac OS.
3. A lot of additional information is available for free on our resources page **www.101labs.net/resources .**
4. All lab images are available in color for free at **https://www.101labs.net/resources/wireshark/** . These colored images will help you with many features such as colorization, which we can't represent well because the book is printed in black and white.
5. Please DO NOT follow these labs on a live network or on equipment belonging to private companies or individuals.
6. You MUST be reading or have read a suitable Wireshark study guide. I don't explain any theory in this book; it's all hands-on labs. Our video course is at **www.howtonetwork.com** .
7. It's impossible for me to give individual support to the thousands of readers of this book (sorry!), so please don't contact me for tech support. Each lab has been tested by several tech editors from beginner to expert.
8. We STRONGLY recommend you read a quality Network+ book or take a video course because this book presumes you

understand core TCP/IP and wireless networking.
9. If you get stuck with a live capture on your home network then try a capture file from the Wireshark website instead (we show you how inside).

Video Training

Each 101 Labs book has an associated video training course. You can watch the instructor configure each lab and talk you through the entire process step by step as well as share helpful tips for the real world of IT. Each course also has 200 exam-style questions to prepare you for the real thing. It’s certainly not necessary to take that course, but if you do, please use the coupon code ‘101wcna’ at the checkout page to get a big discount as a thank you for buying this book.

https://www.101labs.net

Also from Reality Press Ltd.

Cisco CCNA Simplified
Cisco CCNA in 60 Days
IP Subnetting—Zero to Guru
101 Labs—CompTIA Network+
101 Labs—IP Subnetting
101 Labs—Cisco CCNA
101 Labs—Cisco CCNP
101 Labs—Linux LPIC1
101 Labs—CompTIA Linux+

Wireshark Functionality

Lab 1. Wireshark Introduction

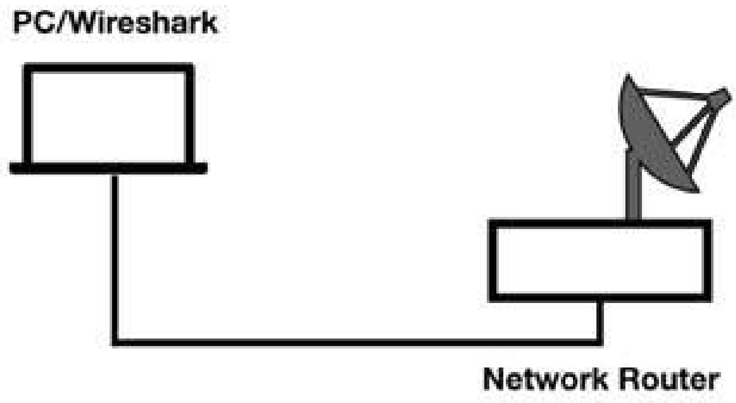
Lab Objective:
Learn about Wireshark and its uses.

Lab Purpose:
Wireshark is a network packet analyzer to capture network packets and display their details. It is similar to a measuring device for examining what's happening inside a network cable. Wireshark is open source, free, and available for almost all operating systems. Wireshark is useful for:

- Troubleshooting network problems
- Examining security problems
- Verifying network applications
- Debugging protocol implementations
- Learning about the internal functioning of network protocols

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



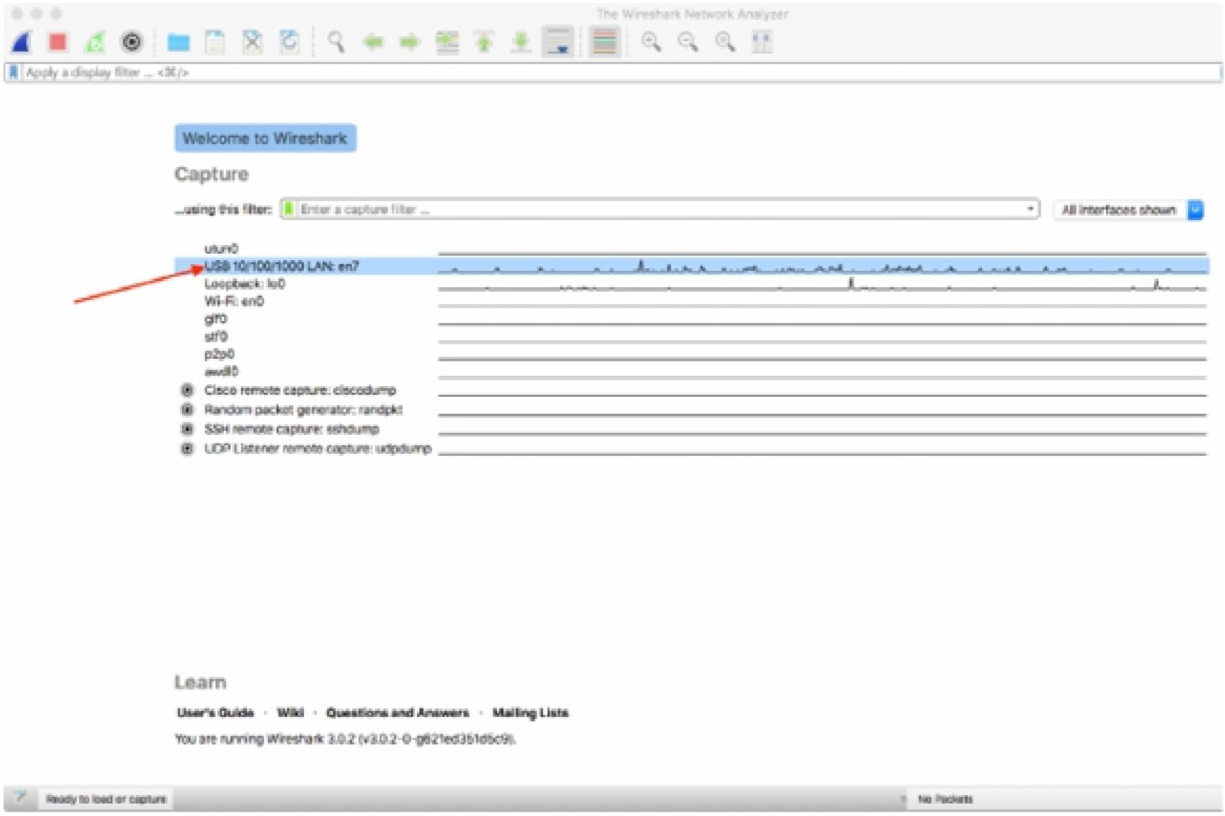
Lab Walkthrough:

Task 1:
Obtain the latest version of Wireshark. Download the latest stable version of Wireshark for your operating system from **https://www.wireshark.org/download.html** .

Install the downloaded version on your PC by following the steps of the default configuration wizard.

Task 2:
Verify the physical connection between the PC (equipped with Wireshark) and the network router connected to a network.

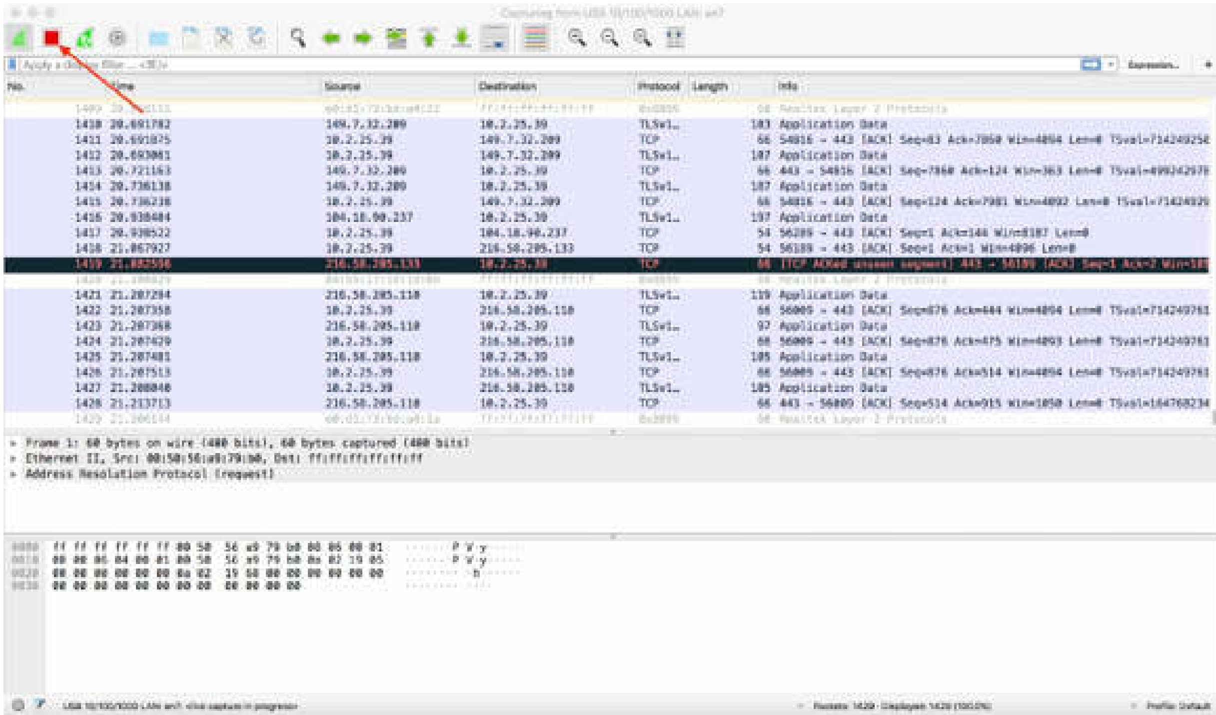
Task 3:
Open Wireshark, and in the main window, double-click the capture interface that you are using as a connection with the router. In the figure below, the interface is "LAN:en7", i.e., the cable interface.



The capture starts, and the packets are displayed in real time.

Task 4:

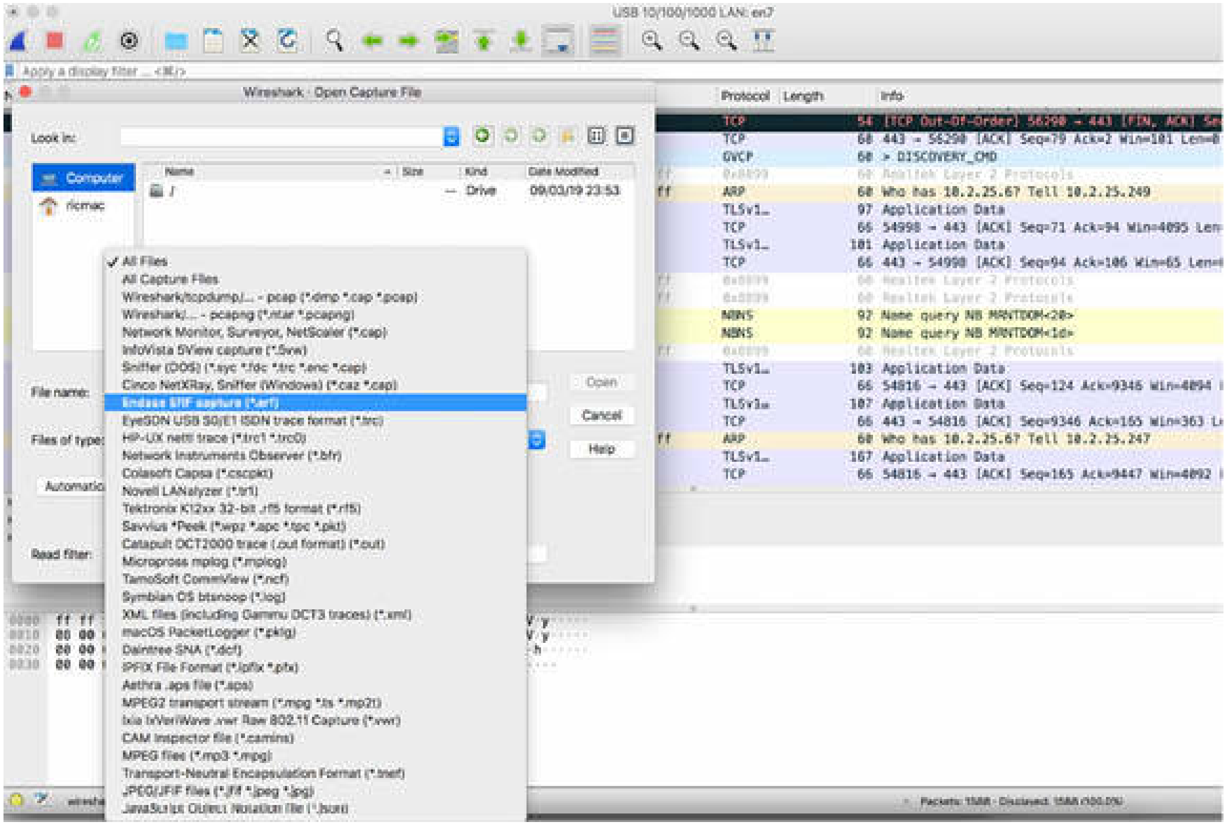
Stop the live capture by clicking the Stop button (RED square shown in the figure below) and save the capture in a file (such as TraceFile.pcap) by using the File menu. Remember that you can download all the lab images from the resources page on www.101labs.net if you need to see them in color.



Task 5:

Download the free sample capture file http.cap to your PC from https://wiki.wireshark.org/SampleCaptures#TCP , and open this file. To open the file in Wireshark, on the main menu, select File > Open.

The Open Capture File dialog box displays a list of supported file types that can be opened.



Notes:

The website https://wiki.wireshark.org/SampleCaptures#TCP contains a lot of captures. We recommend downloading captures in different file formats and opening them to view the differences between various formats and their properties.

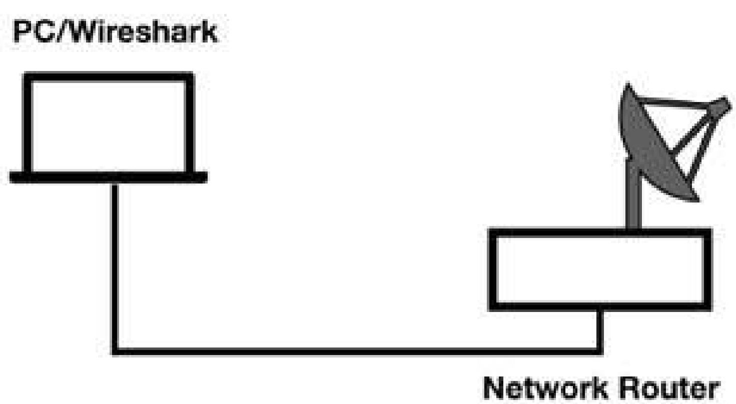
Lab 2. Wireshark Main Window

Lab Objective:
Learn about the components of the main window of Wireshark.

Lab Purpose:
Learn about the main window, the main menu, and the main toolbar of Wireshark to gain confidence in using the basic functionalities.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

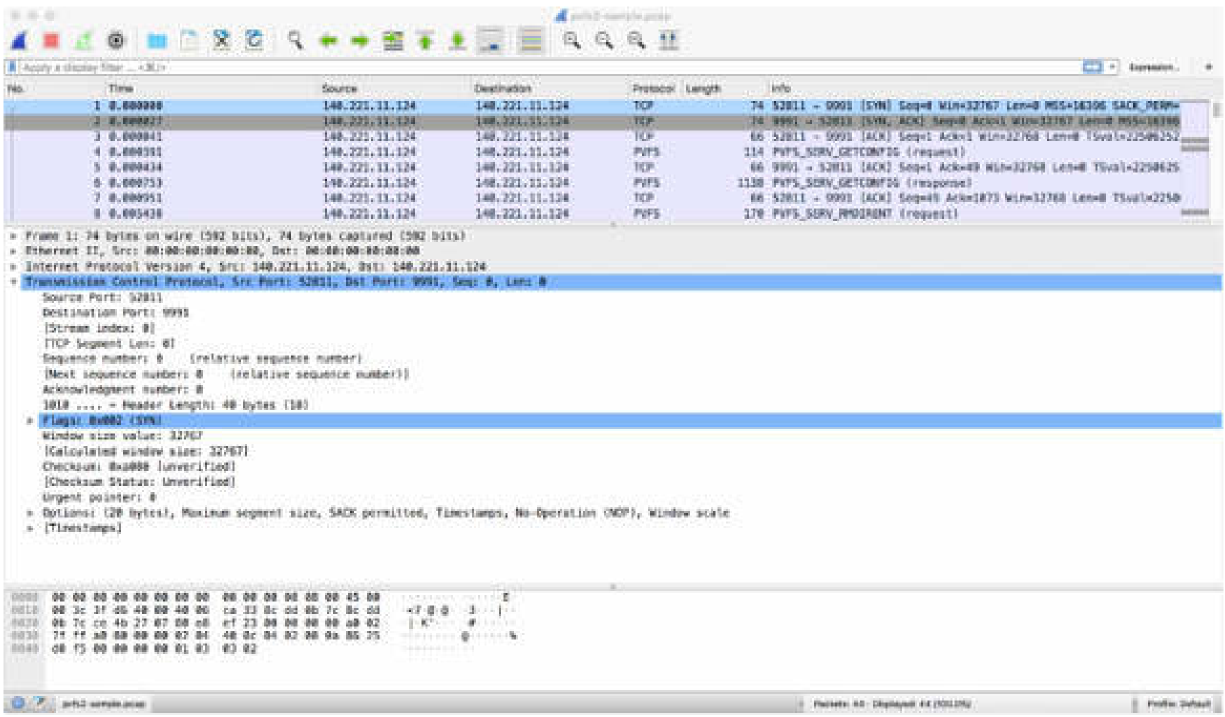


Lab Walkthrough:

Task 1:

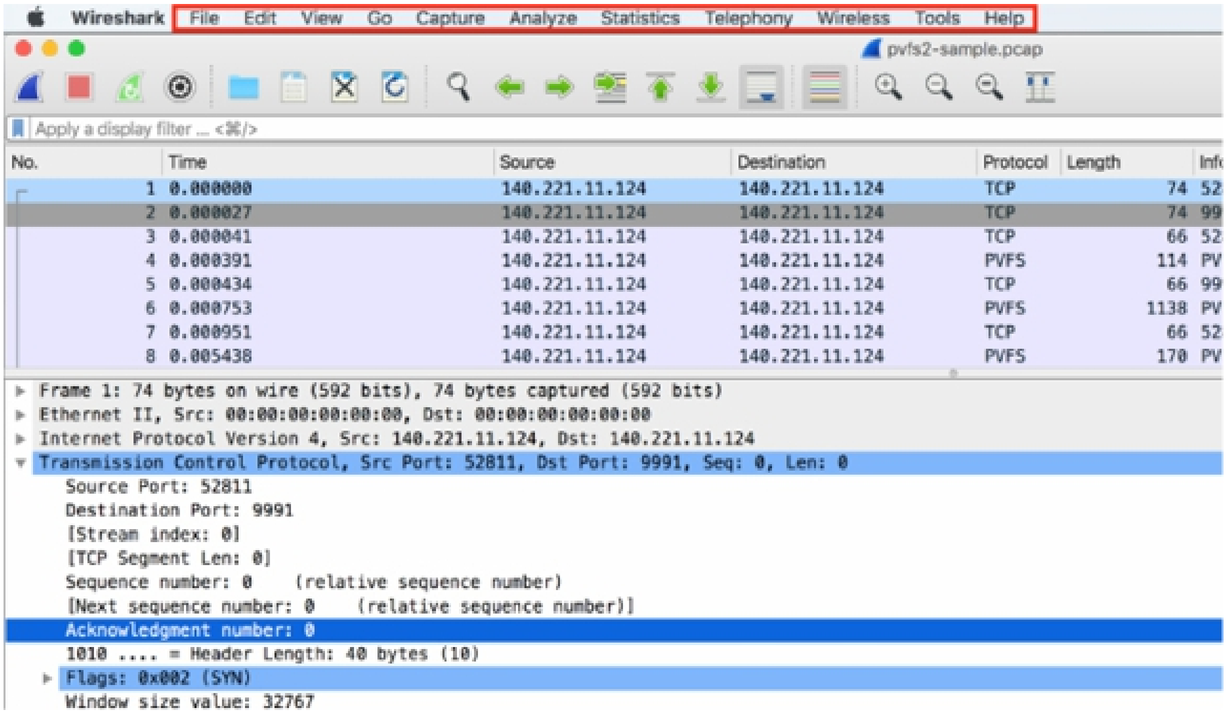
Download the free sample capture file pvfs2-sample.pcap to your PC from https://wiki.wireshark.org/SampleCaptures#TCP , and then open the downloaded file in Wireshark.

Task 2:
Identify the components of Wireshark's main window shown in the figure below.

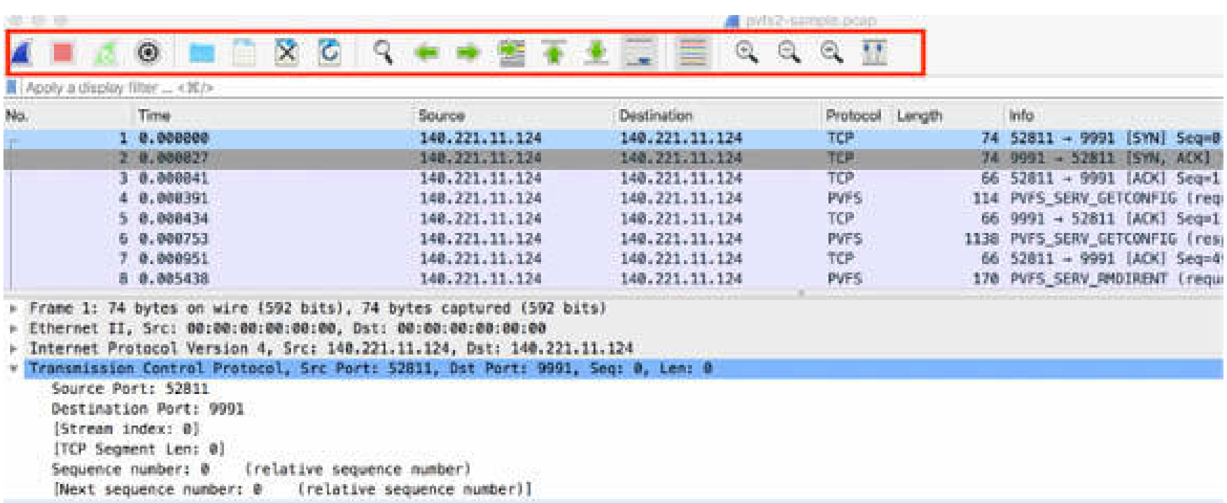


The Wireshark main window consists of the following components:

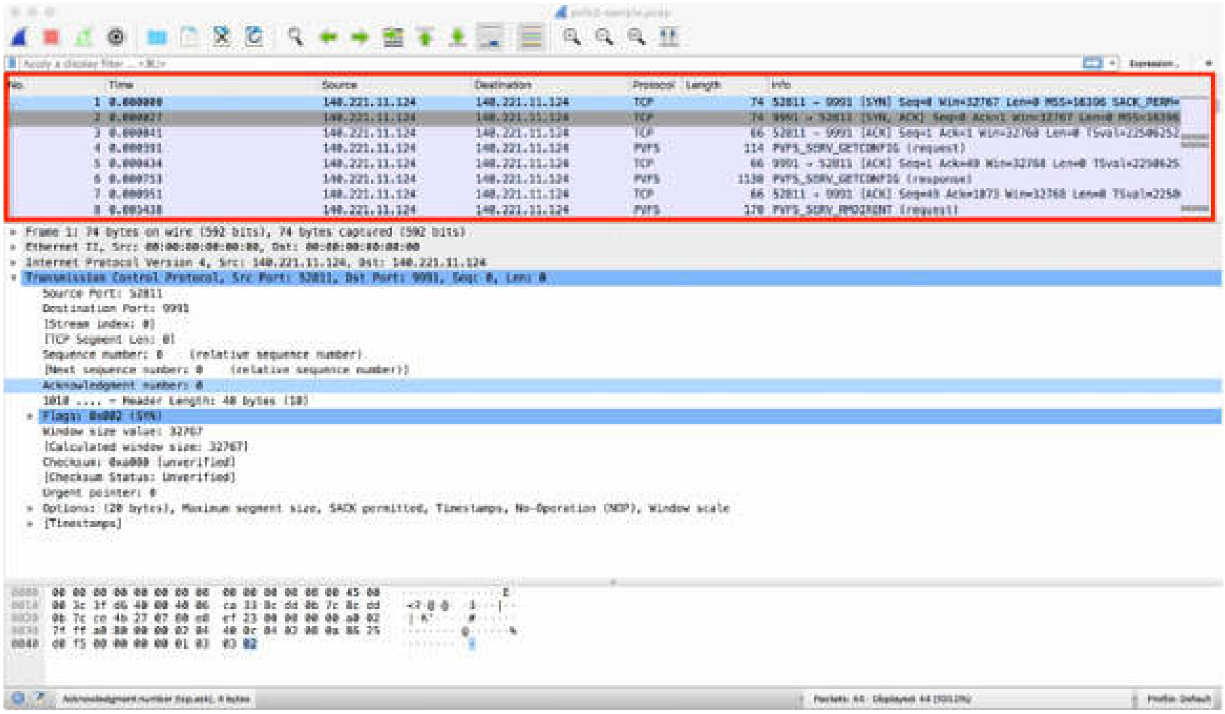
1. The main menu is used to start actions.



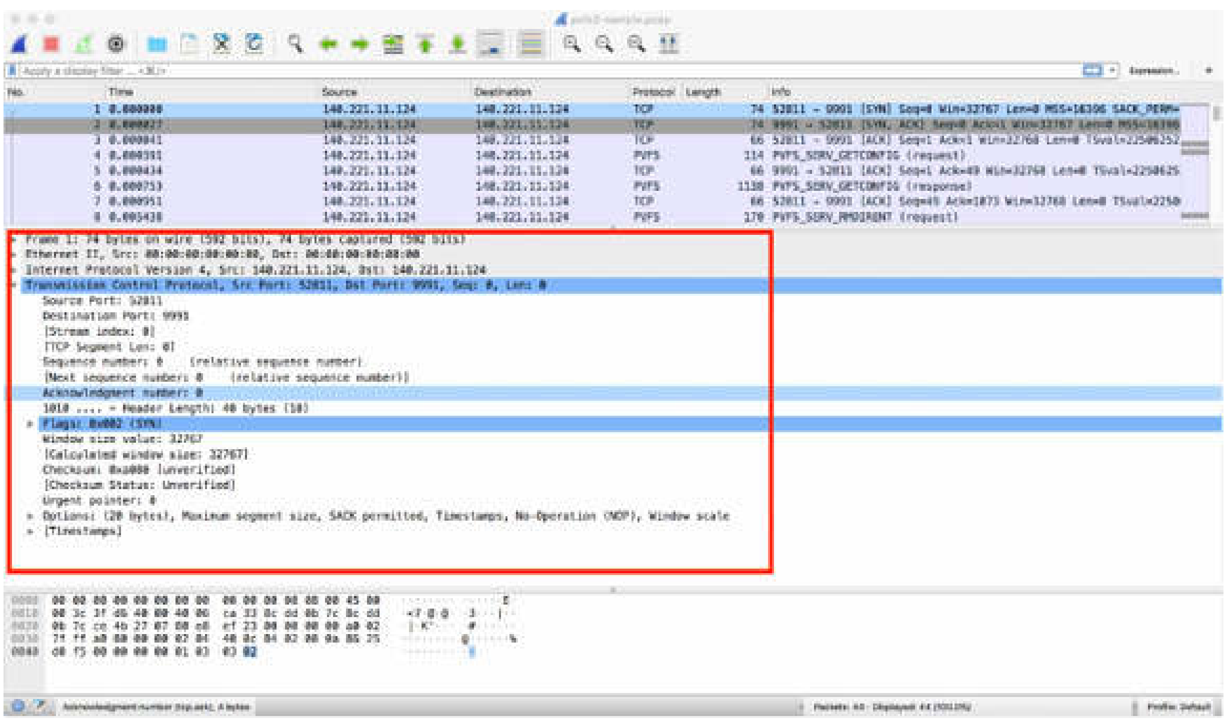
2. The main toolbar provides quick access to frequently used items from the menu.



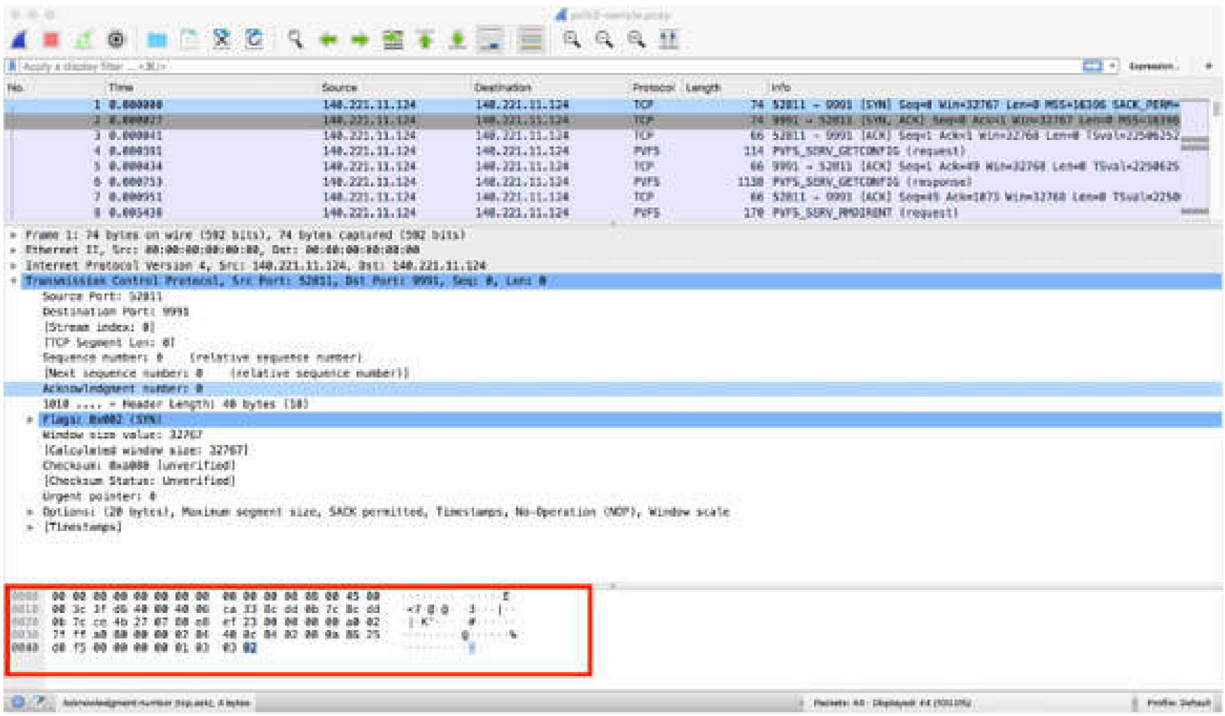
3. The Packet List pane displays a summary of each captured packet. By clicking on the packets in this pane, you control what is displayed in the other two panes.



4. The Packet Details pane displays more detailed information about the packet selected in the Packet List pane.



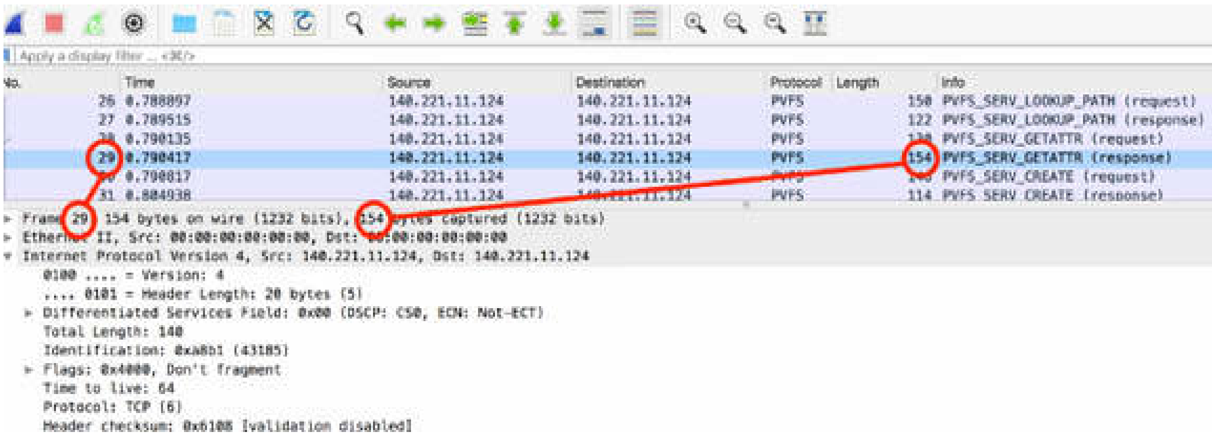
5. The Packet Bytes pane displays the data of the packet selected in the Packet List pane and highlights the field selected in the Packet Details pane.



6. The statusbar shows detailed information about the current program state and the captured data.

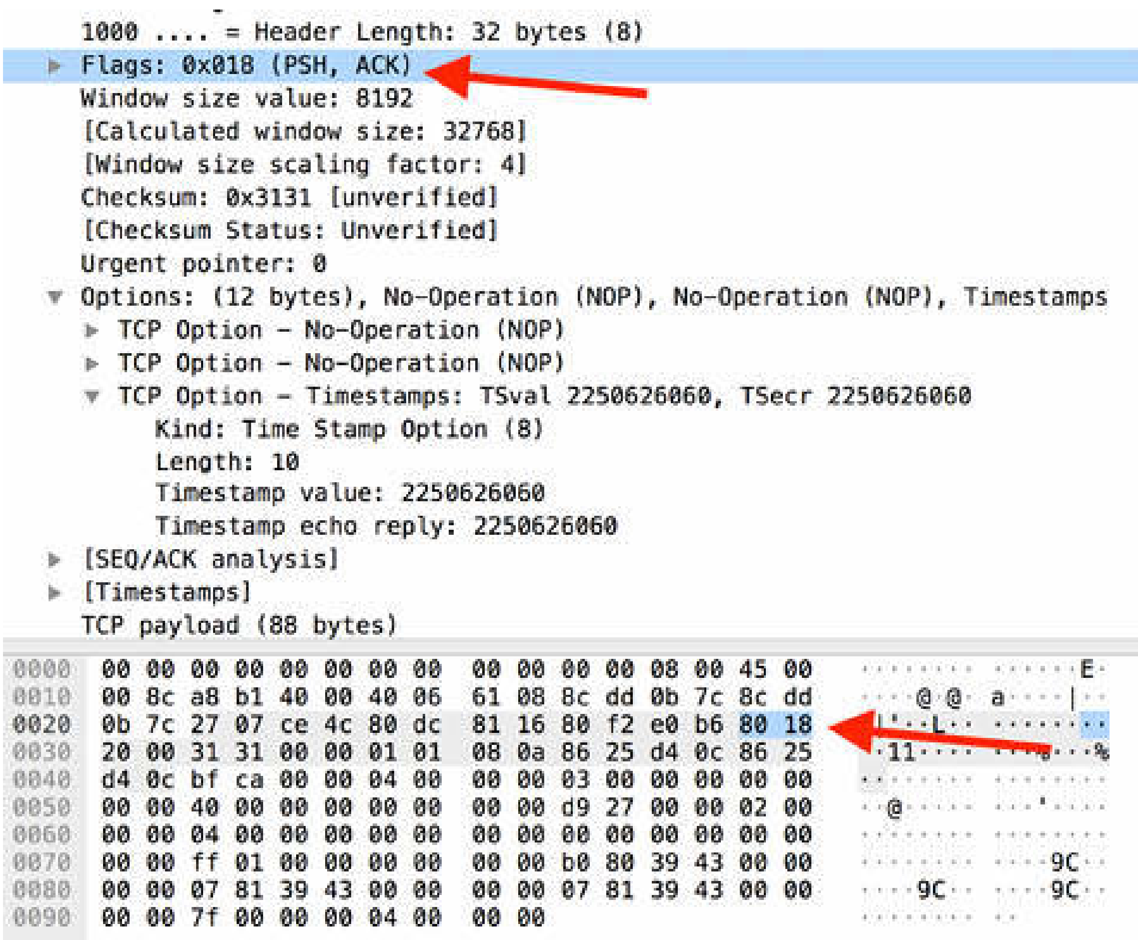
Task 3:

Click a different packet in the Packet List pane (i.e., packet #29), and see the changed details in the Packet Details pane; for example, the packet number and the packet length.

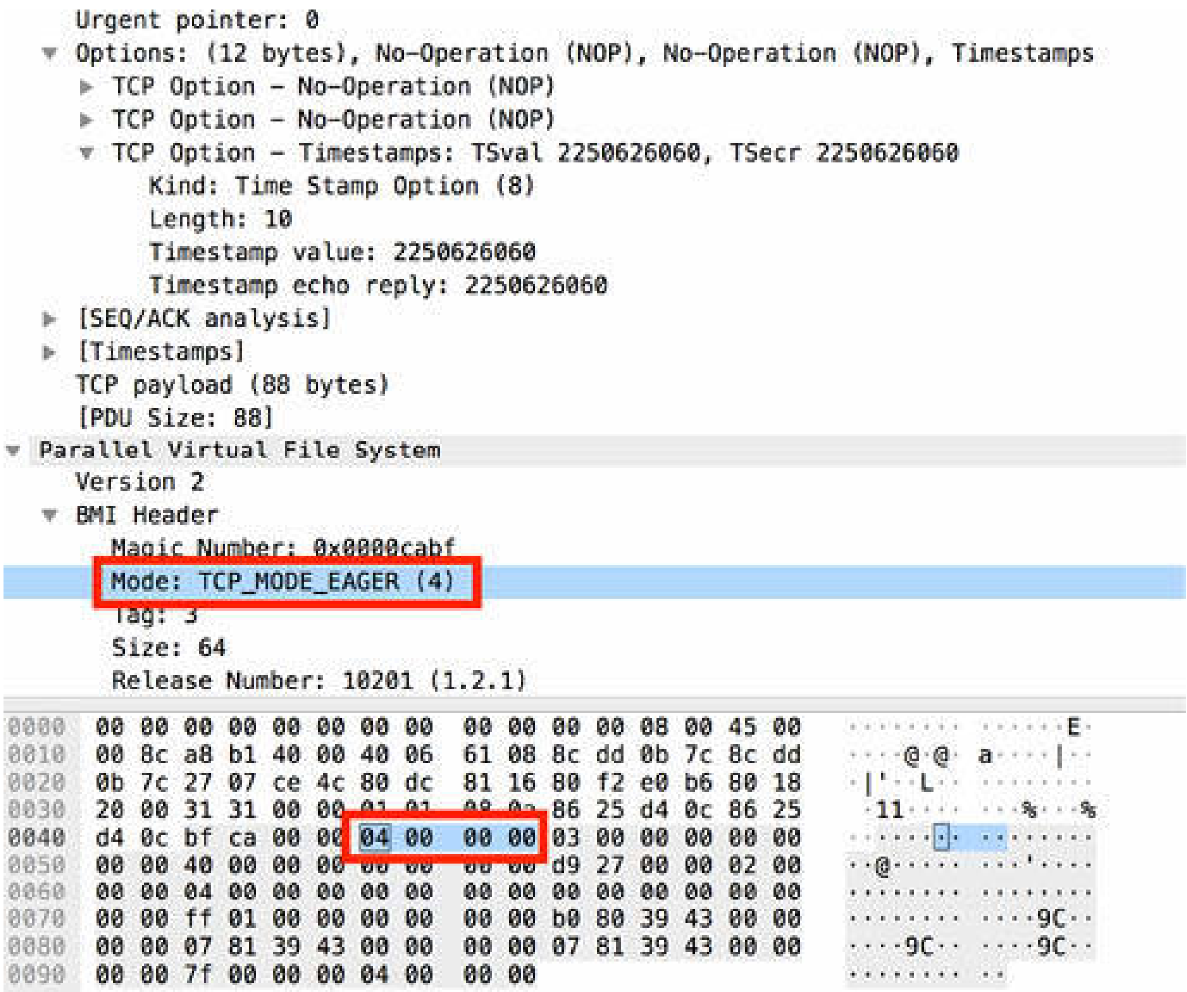


Task 4:

Click a detail (such as the Flags field shown in the figure below) in the Packet Details pane, and the related byte field is automatically selected in the Packet Bytes pane.



Task 5:
Select a byte field in the Packet Bytes pane, and the related description is automatically selected in the Packet Details pane.



Notes:
You can customize menu items and toolbar buttons to create an interface that fits your needs.

All lab images can be downloaded from the resources page on www.101labs.net .

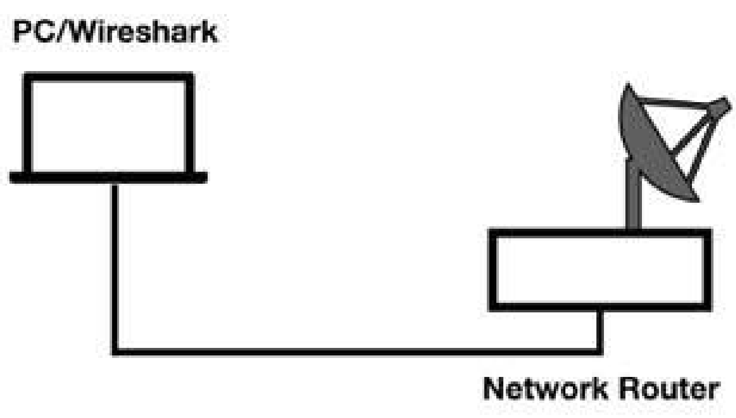
Lab 3. Wireshark Main Menu

Lab Objective:
Learn the composition of Wireshark's main menu.

Lab Purpose:
Learn about Wireshark's main menu items to gain confidence with the basic functionalities.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

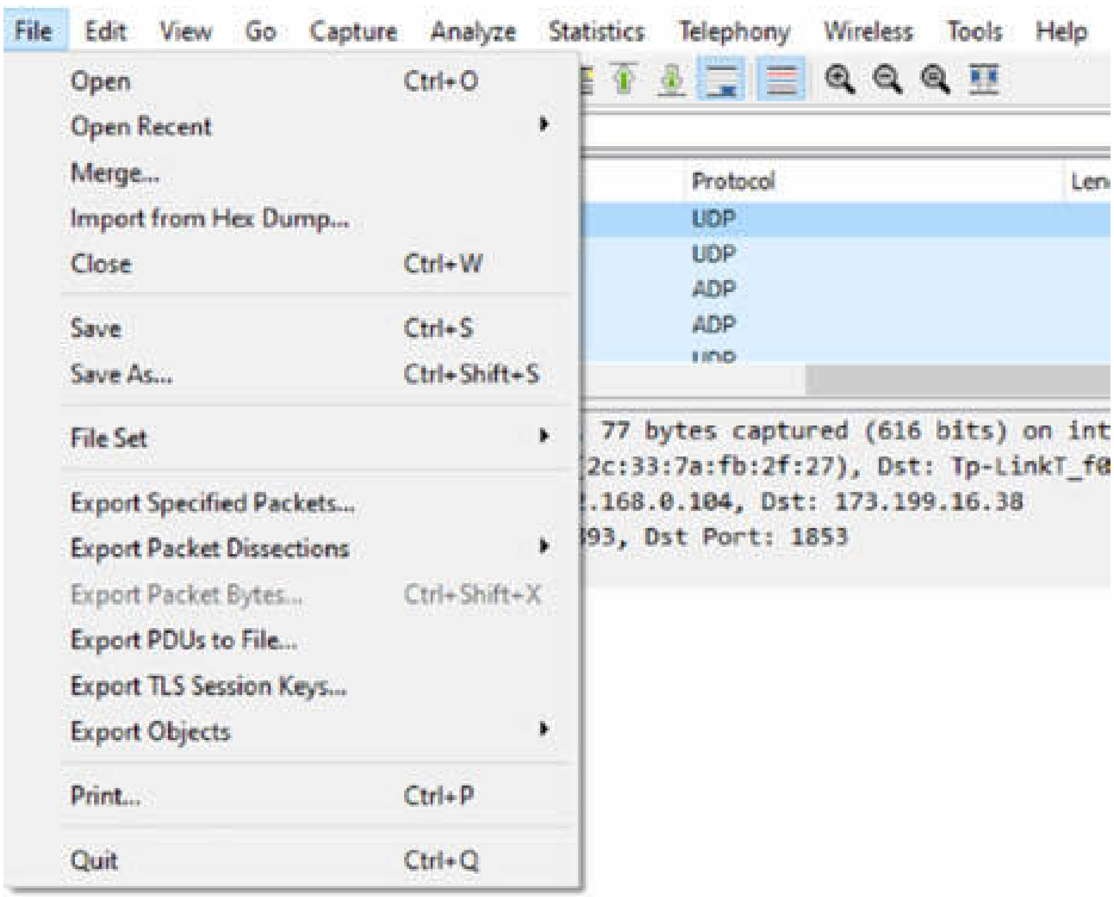


Lab Walkthrough:

Task 1:

Download the free sample capture file snmp_usm.pcap to your PC from https://wiki.wireshark.org/SampleCaptures#TCP , and then open the downloaded file in Wireshark.

Task 2:
On the main menu, click the File menu. The drop-down menu displays a complete list of available items, as shown in the figure below. Please note that your options may change depending on your version of Wireshark and operating system.

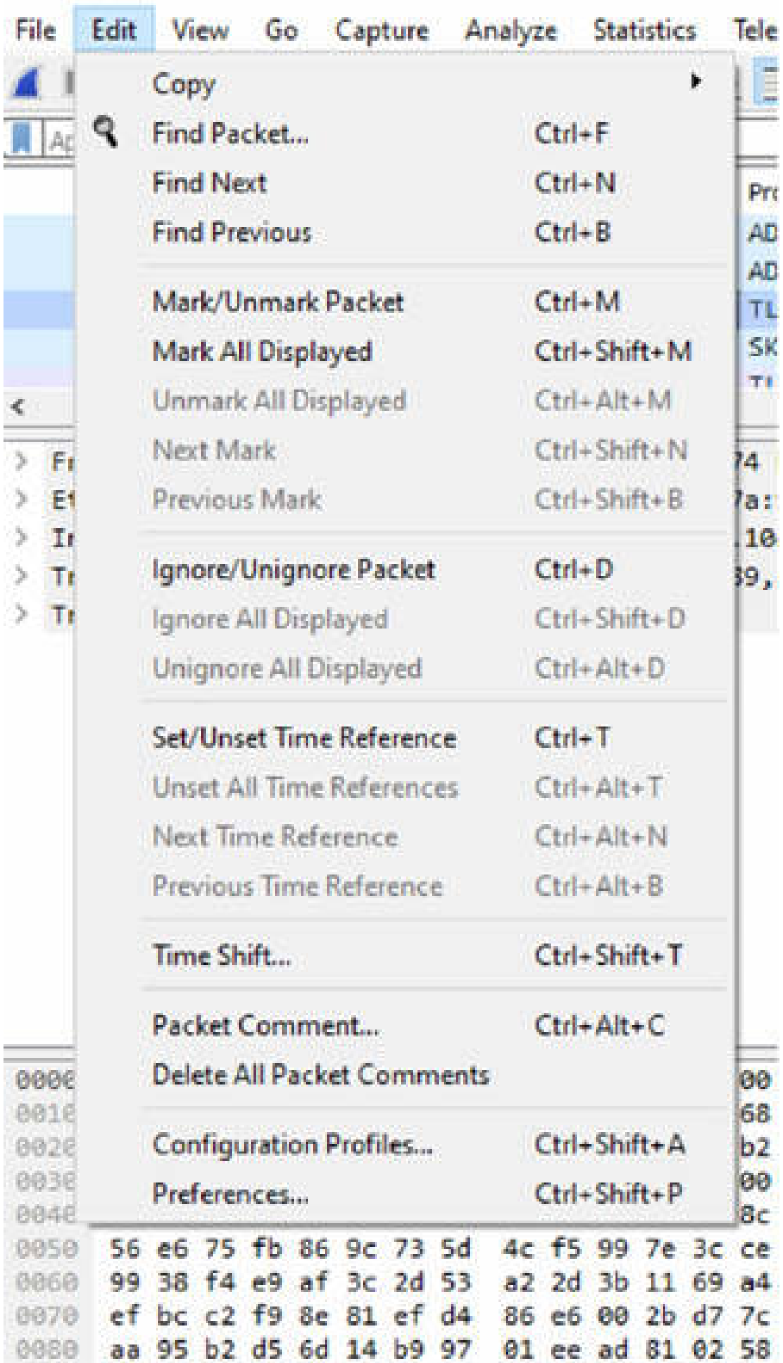


The File menu contains items to open, merge, and close capture files; import from Hex dump; save, print, or export capture files in whole or in part by selecting specific packets; and quit Wireshark.

Task 3:

On the main menu, click the Edit menu.

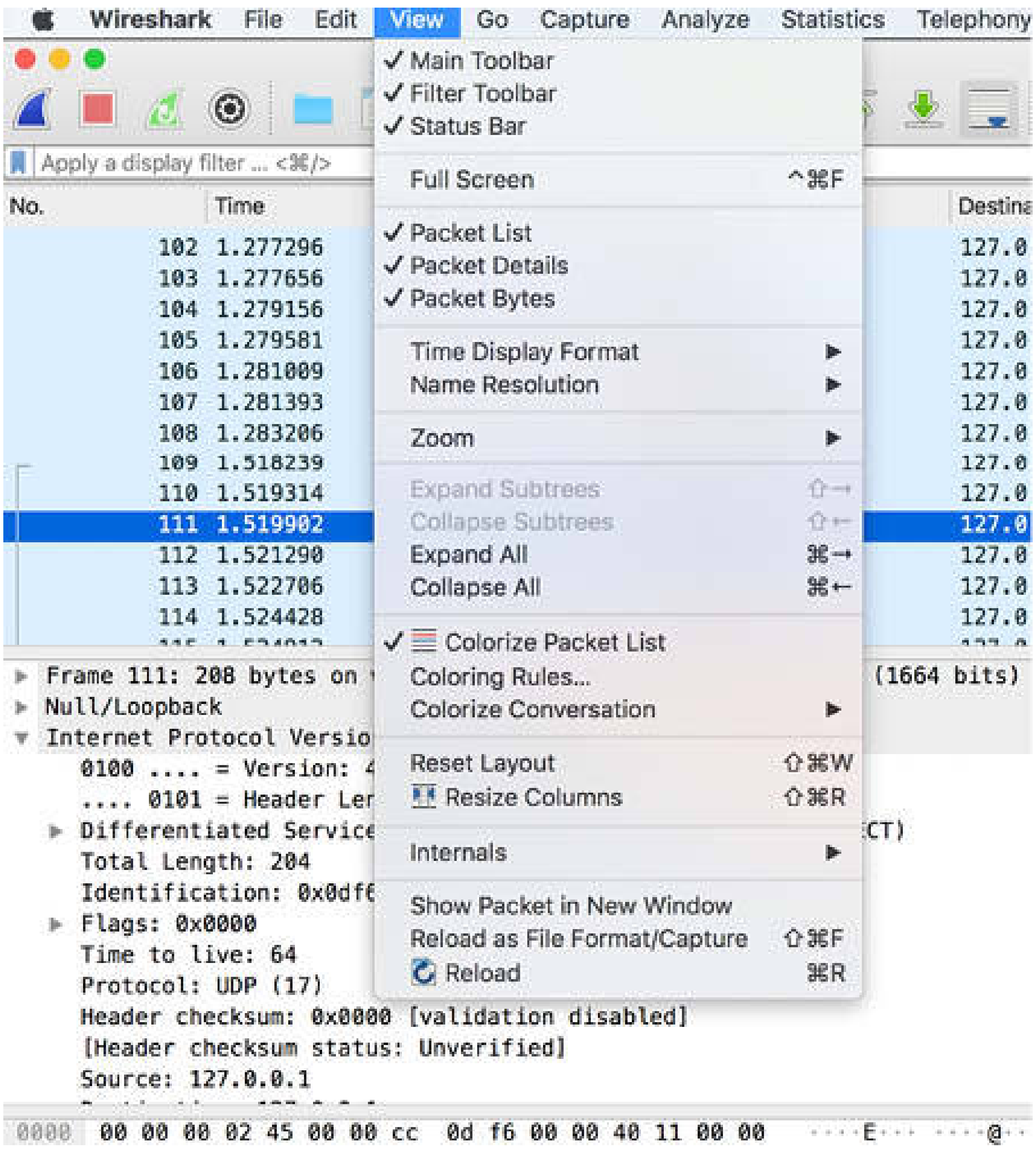
The drop-down menu displays a complete list of available items, as shown in the figure below.



The Edit menu contains items to copy packets (as Plain Text, CSV, or YAML), find a packet, set or unset time reference, set time shift of specific packets, mark and comment one or more packets, handle configuration profiles, and set your preferences.

Task 4:
On the main menu, click the View menu.

The drop-down menu displays a complete list of available items, as shown in the figure below.

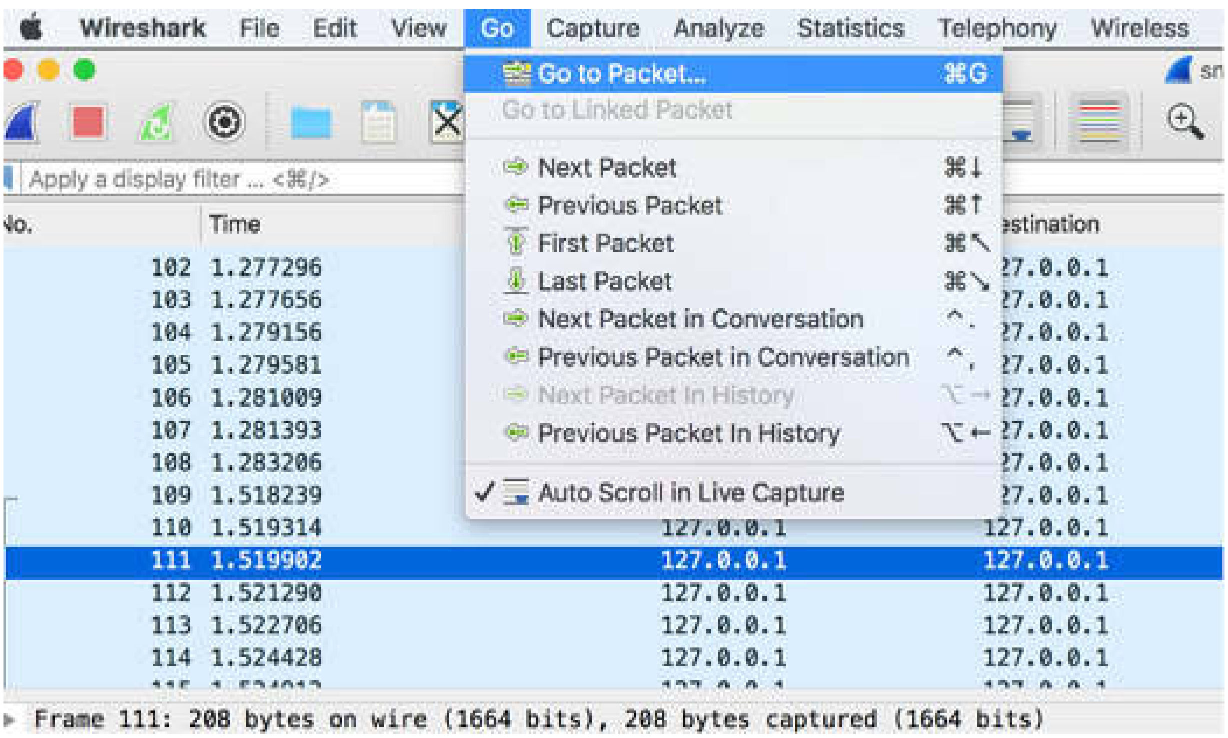


The View menu contains items to do the following:

- Control the display captured data
- Hide or show the Packet List, Packet Details, and Packet Bytes panes
- Define coloring rules for packets display
- Specify whether to colorize the packets
- Specify the format for time display
- Zoom into the font of packets
- Show a packet in a separate window
- Reload packets
- Expand and collapse trees in packet details

Task 5:
On the main menu, click the Go menu.

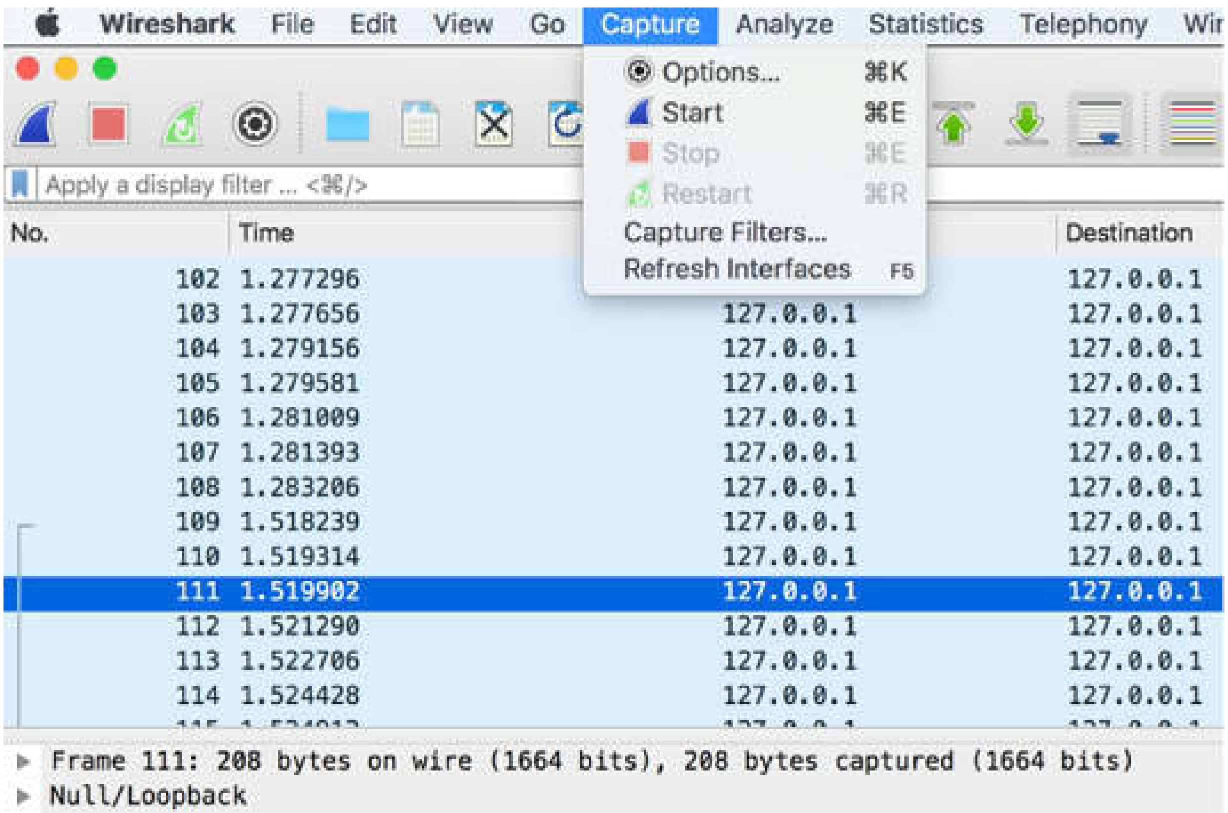
The drop-down menu displays a complete list of available items, as shown in the figure below.



The Go menu contains items to go to a specific packet by entering the packet number or go to the first, last, next, or previous packet. It also allows you to enable auto-scroll in live capture.

Task 6:
On the main menu, click the Capture menu.

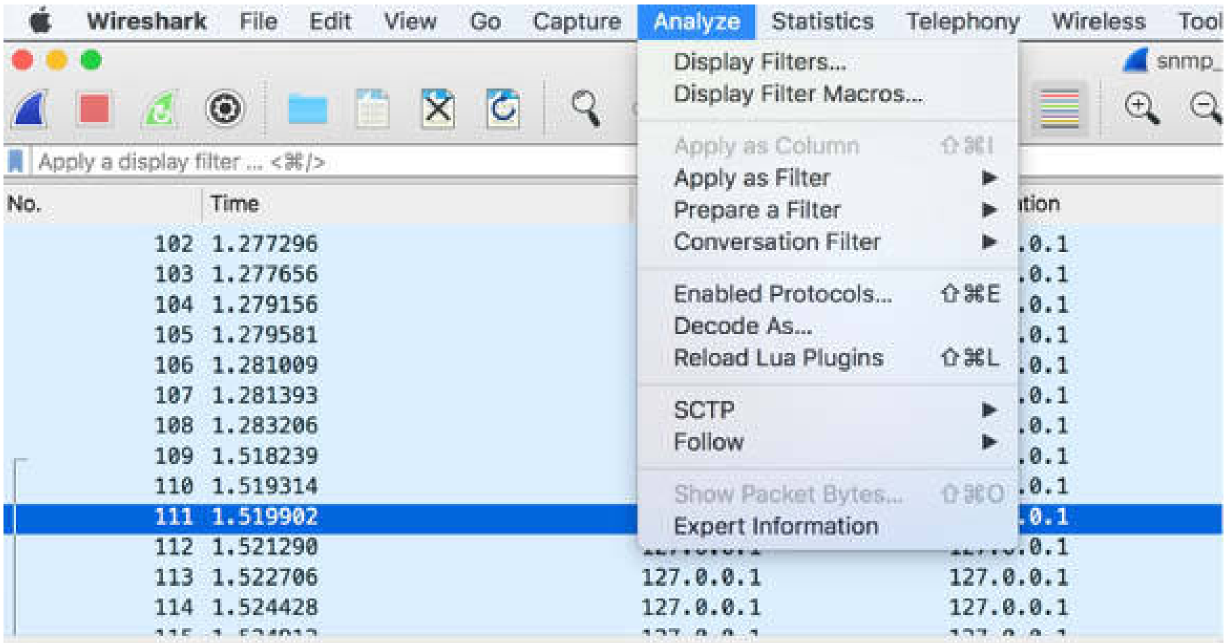
The drop-down menu displays a complete list of available items, as shown in the figure below.



The Capture menu allows you to switch to a different capture interface; start, stop, and restart captures; edit or create capture filters; and refresh capture interfaces.

Task 7:
On the main menu, click the Analyze menu.

The drop-down menu displays a complete list of available items, as shown in the figure below.

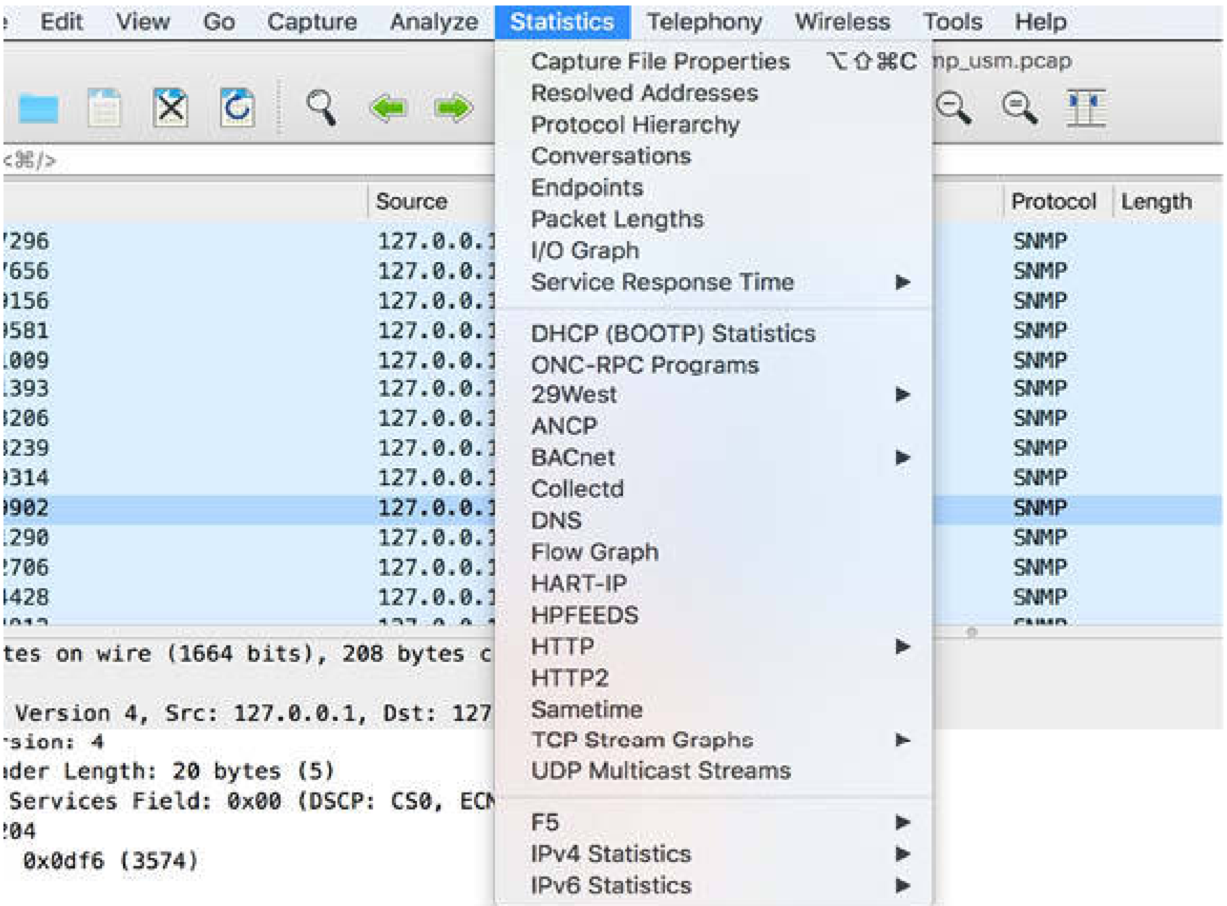


The Analyze menu contains items to:

- Edit or create display filters
- Create display filter macros for complex filtering scenarios
- Apply a filter for selected packets and specify the not/or/and criteria for selected packets
- Prepare a filter for selected packets and specify the not/or/and criteria for selected packets
- Apply conversation filter, based on the protocol, IP, or Ethernet, to enable or disable the dissection of protocols
- Configure user-specified decodes
- Follow a TCP/UDP stream

Task 8:
On the main menu, click the Statistics menu.

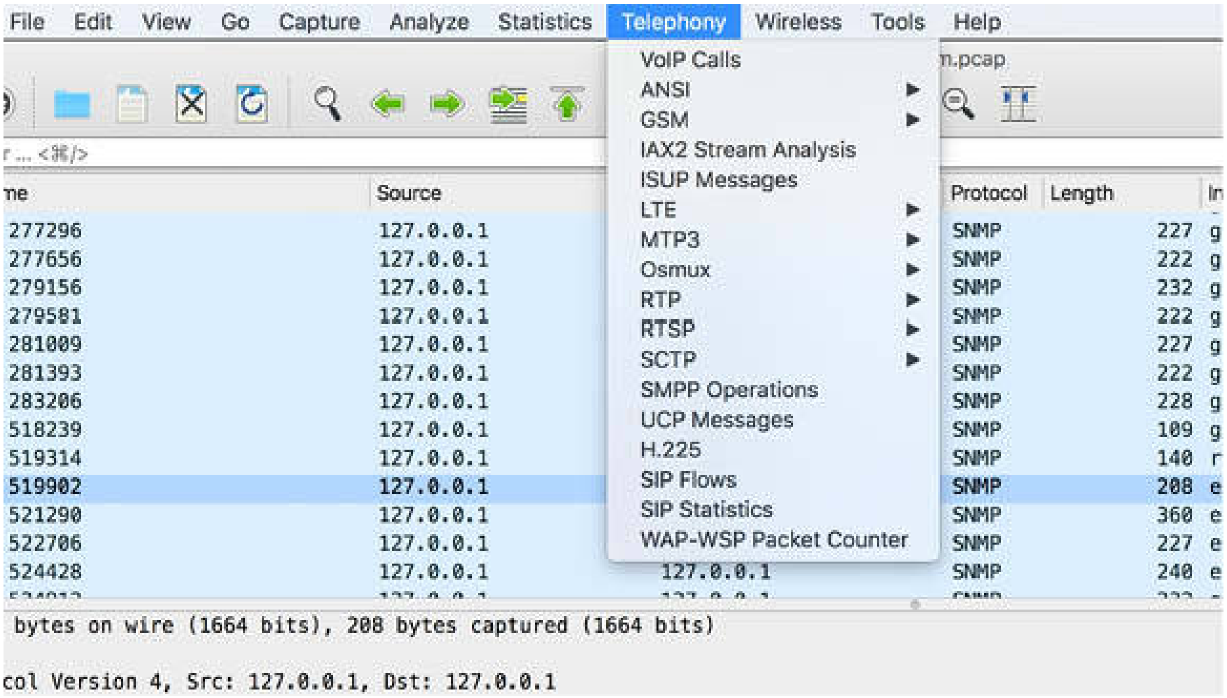
The drop-down menu displays a complete list of available items, as shown in the figure below.



The Statistics menu contains items to display capture file properties, various statistics windows based on the criteria—such as IP, protocol—including a summary of the packets that have been captured, protocol hierarchy statistics, statistics graphs, and much more.

Task 9:
On the main menu, click the Telephony menu.

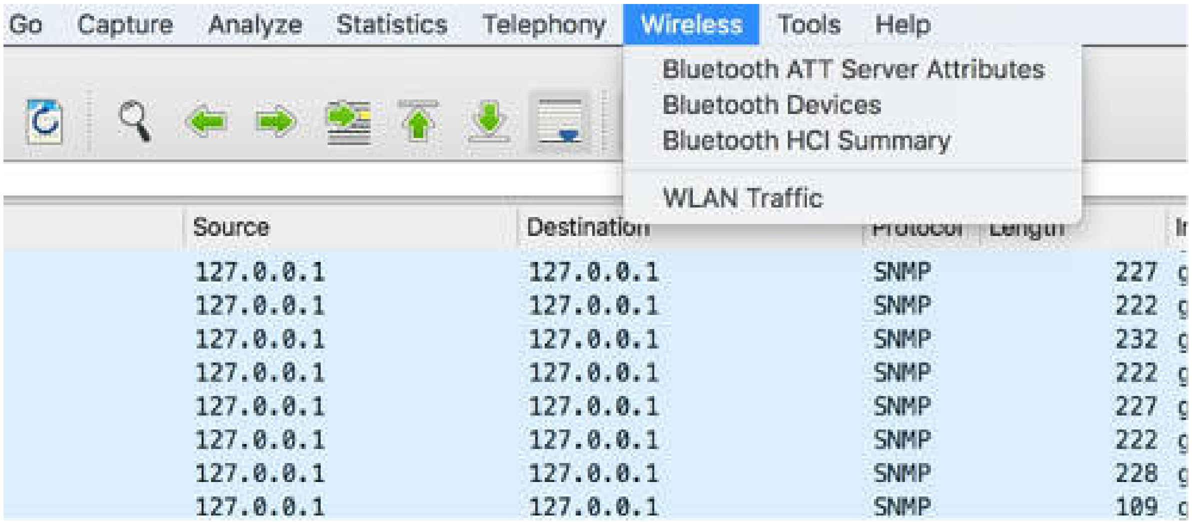
The drop-down menu displays a complete list of available items, as shown in the figure below.



The Telephony menu contains items to display various telephony-related statistics windows for VoIP calls, GSM, ANSI, media analysis, flow diagrams, protocol hierarchy statistics, and much more.

Task 10:
On the main menu, click the Wireless menu.

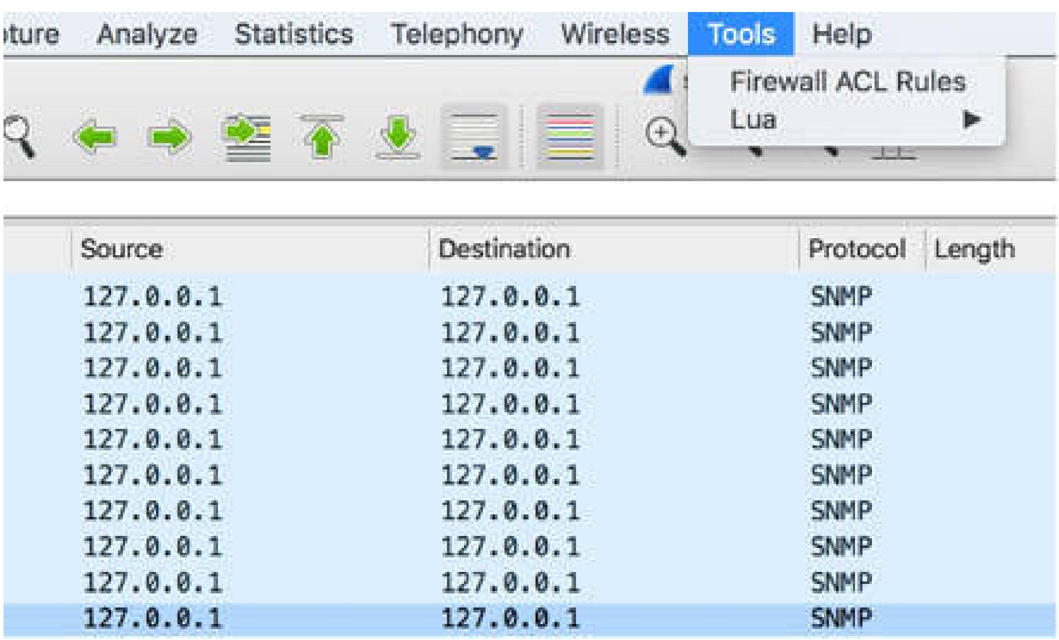
The drop-down menu displays a complete list of available items, as shown in the figure below.



The Wireless menu contains items to display Bluetooth and IEEE 802.11 wireless statistics.

Task 11:
On the main menu, click the Tools menu.

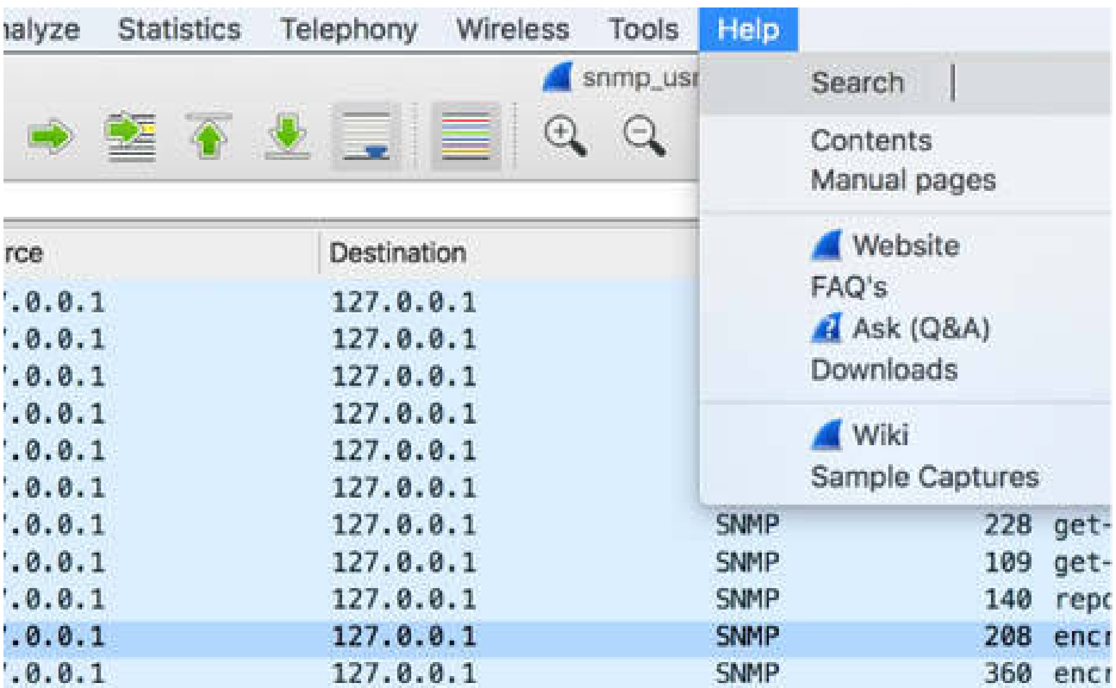
The drop-down menu displays a complete list of available items, as shown in the figure below.



The Tools menu contains various tools available in Wireshark, such as for creating Firewall ACL Rules and some scripts in Lua language.

Task 12:
On the main menu, click the Help menu.

The drop-down menu displays a complete list of available items, as shown in the figure below.



The Help menu contains items to search a topic, access the basic help, access manual pages of the various command-line tools, access some of the webpages online. It also contains the About Wireshark option that provides detailed information about Wireshark.

Notes:
You can also use keyboard shortcuts to access main menus and submenus; for example, to open the Go To Packet dialog box, press Ctrl+G.

Your menu options will differ depending on your release version of Wireshark and your operating system.

Lab 4. Wireshark Basic Capture Setup

Lab Objective:
Learn how to configure the basic capture options.

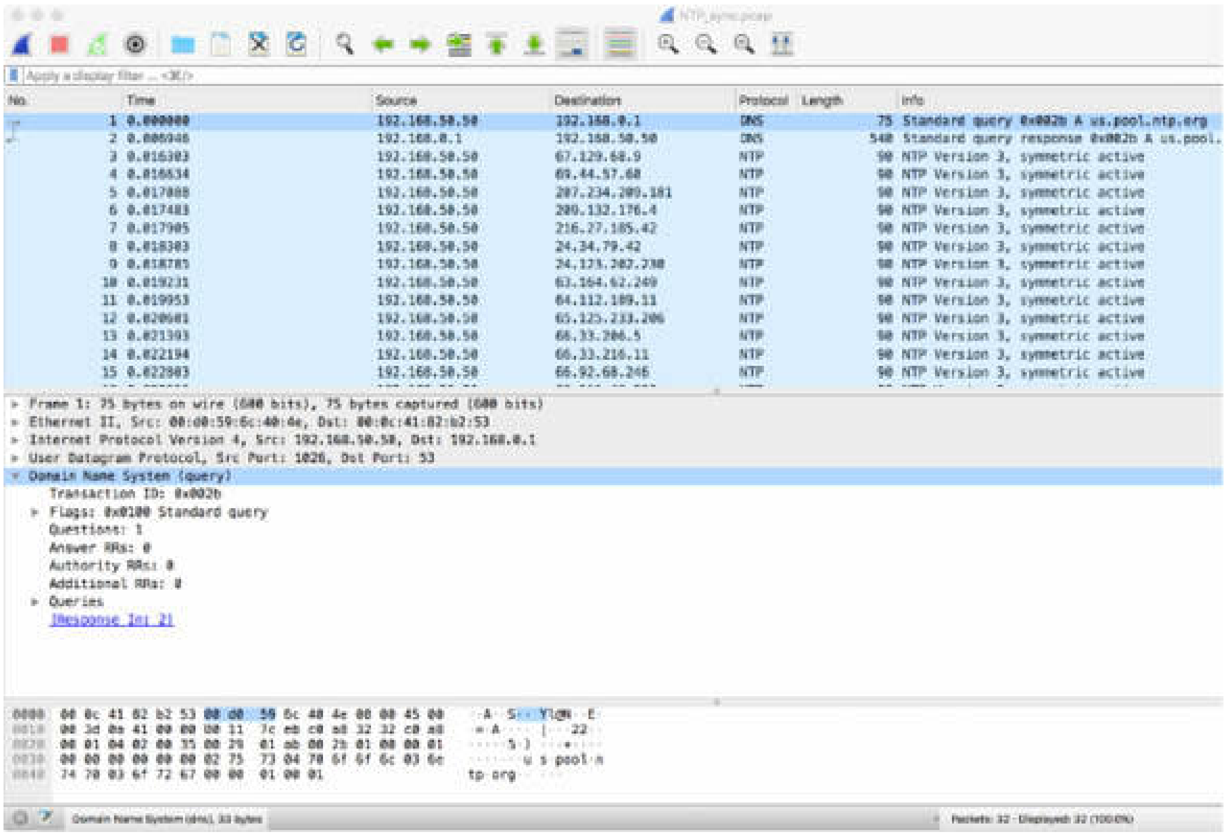
Lab Purpose:
Learn when to run Wireshark locally and when and where to tap into the network. Capture traffic on switched networks or set up a mirroring port on a network switch.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
This lab uses a different lab topology, which is described in the following section.

Lab Walkthrough:

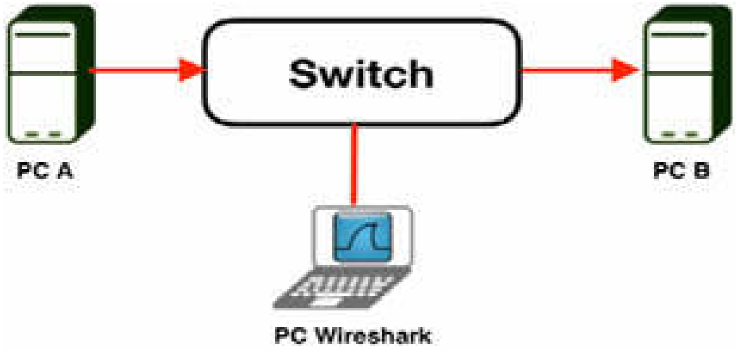
Task 1:
When Wireshark is used to analyze saved packet captures, it's not necessary to connect the PC to a network interface. You can run Wireshark locally. For example, download the packet capture file NTP_sync.pcap from **https://wiki.wireshark.org/SampleCaptures#Network_Time_Protocol** , and open it in Wireshark. The packet list and the detailed view are displayed in Wireshark, as shown in the figure below.



Task 2:

To analyze a network, you can run Wireshark in the live mode, which means that you have to choose where to tap into the network. There are several types of networks. For each network, there are different ways to analyze the traffic.

Let’s start with PC A and PC B connected through a network switch and PC Wireshark connected to any port on the network switch. Assign the IP address 192.168.1.22 to PC A, the IP address 192.168.1.23 to PC B, and the IP address 192.168.1.24 to PC Wireshark.

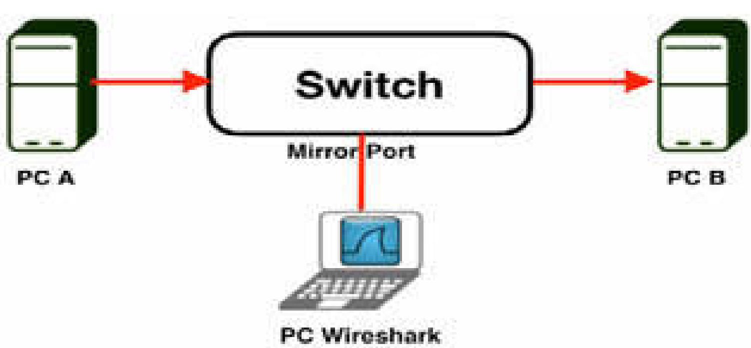


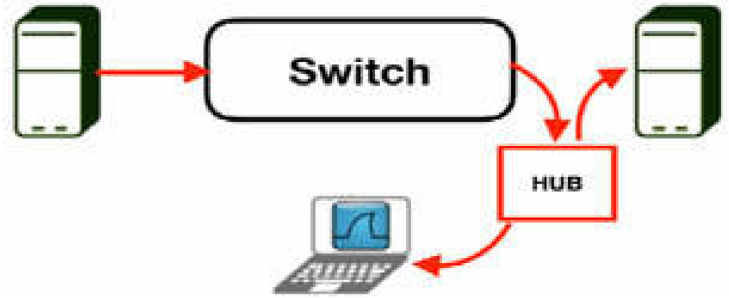
From the terminal window of PC A, run the ping –t 192.168.1.24 command to send a ping request (point-to-point unicast message) to PC Wireshark. If you start a network capture on PC Wireshark, you can see the ping requests (from PC A) and the ping answers (from PC Wireshark) because they are directed to and from the PC where the Wireshark analyzer is running.

Task 3:
In the PC A terminal window, run the ping –t 192.168.1.23 command to send a ping request to PC B. If you start a network capture on PC Wireshark, you won't see any packet coming from PC A or PC B. In such a case, you can do either of the following to capture the ping traffic:

1. Connect the PC Wireshark to a mirror port of the switch (if it has one) to replicate the whole traffic on that port.
2. Add a Tap or an "Old" Ethernet Hub to the network configuration to capture the packets directed towards and from PC B.

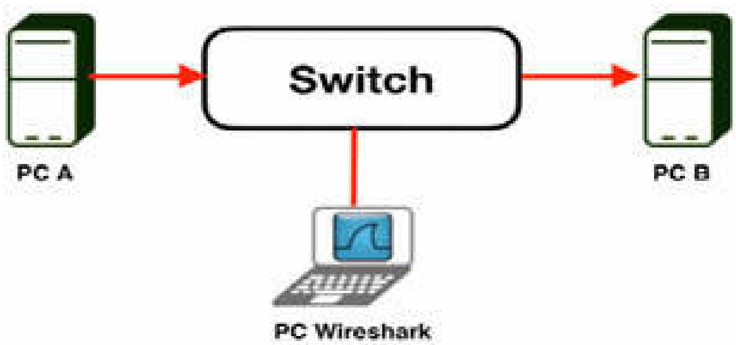
The figures below show both these cases.





Task 4:
For connections without a hub and without mirroring ports, as shown in the figure below, you can only capture broadcast traffic.

To capture the broadcast traffic, in the terminal window of PC A, run the arp –d command to clear the ARP cache and then run the ping –t 192.168.1.23 command and immediately start a capture on PC Wireshark. In this case, you can capture the ARP requests messages broadcasted from PC A to all nodes of the network.



Notes:
There are various types of network configurations. For each network configuration, you must modify the capture configuration and the connection to the monitoring port. Techniques such as tap port, hub, machine-in-the-middle, and MAC flooding are useful for capturing traffic on a network. Choose a technique that is the most suitable for your needs.

We issued these commands on a Linux machine so if you are using Windows or another OS you will need to check for the appropriate commands.

Lab 5. Wireshark Capture Interface Options

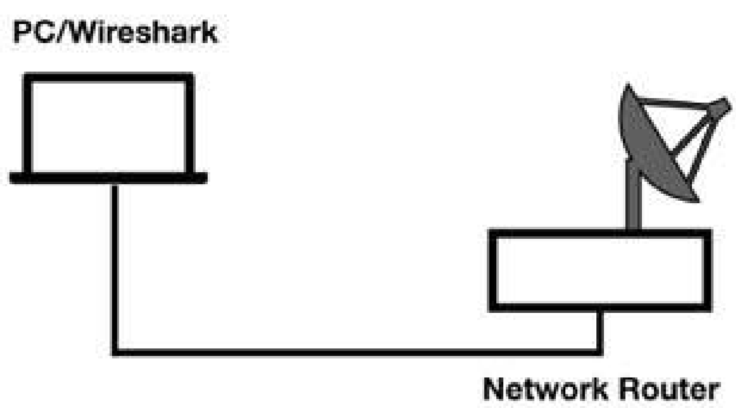
Lab Objective:
Learn how to manage different capture interface options.

Lab Purpose:
Set appropriate settings on the Wireshark capture interface options to:

- Identify the most appropriate capture interface for capturing simultaneously on multiple interfaces.
- Capture traffic remotely.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

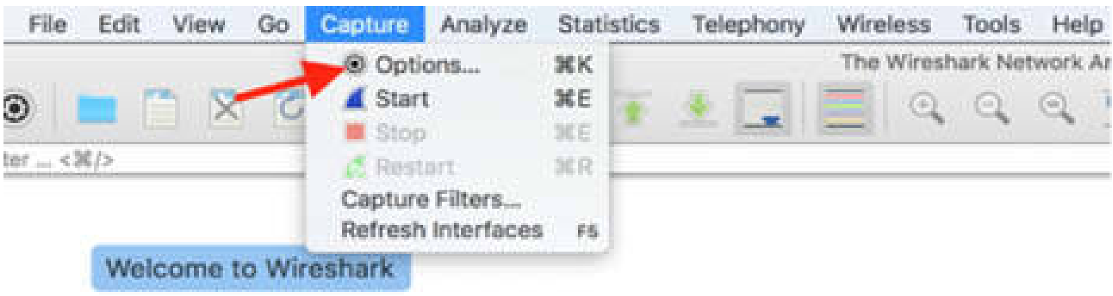
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

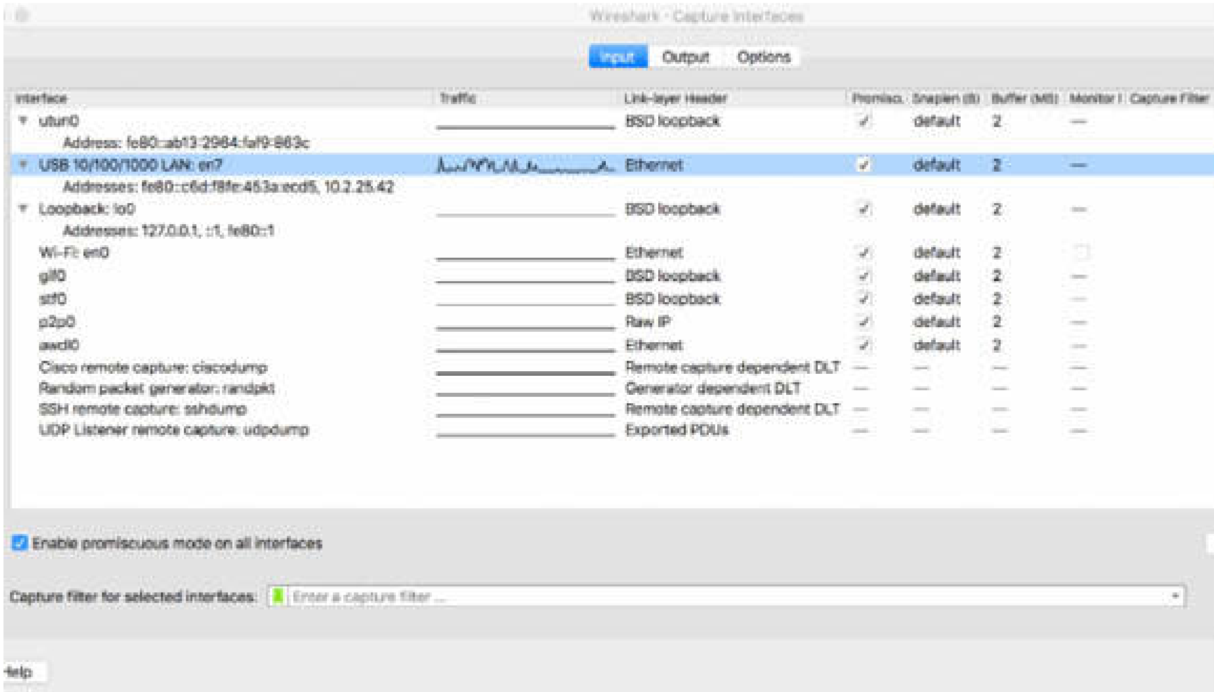
Task 1:
Verify the physical connection between the PC (equipped with Wireshark) and the network router connected to the internet.

Task 2:
Open Wireshark, and on the main menu, select Capture > Options, as displayed in the figure below.



The Capture Options dialog box is displayed.

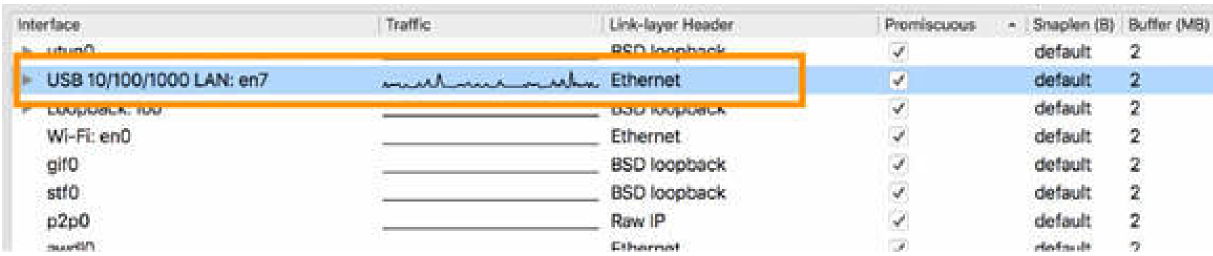
Click the Input tab. The configuration parameters related to the capture input interfaces are displayed, as shown in the figure below.



As shown in the figure above, you can view the complete list of the capture interfaces available on the PC. The interface has the following attributes:

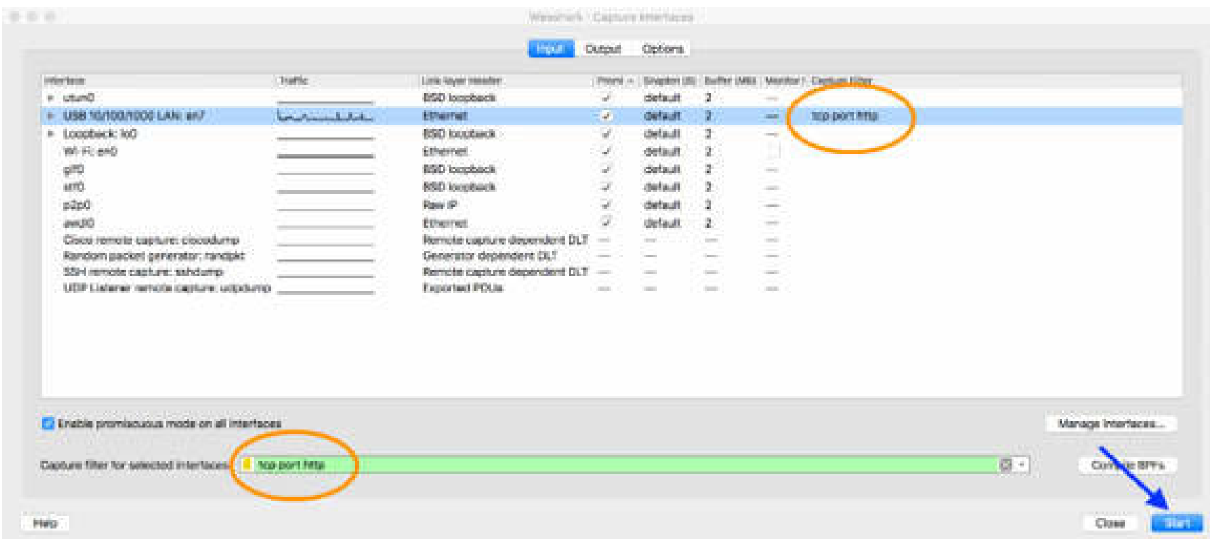
- Interface name
- Traffic (if available)
- Link-layer header
- Promiscuous setting
- Snaplen
- Buffer
- Monitor mode
- Capture filter

Task 3:
Select an interface for which the line graph displays some activity in the Traffic column, such as LAN:en7 interface shown in the figure below.



For the selected interface, in the “Capture filter for selected interfaces” box, type tcp port http . If the syntax is correct, the syntax box will have a green background. This setting is also displayed in the Capture Filter column.

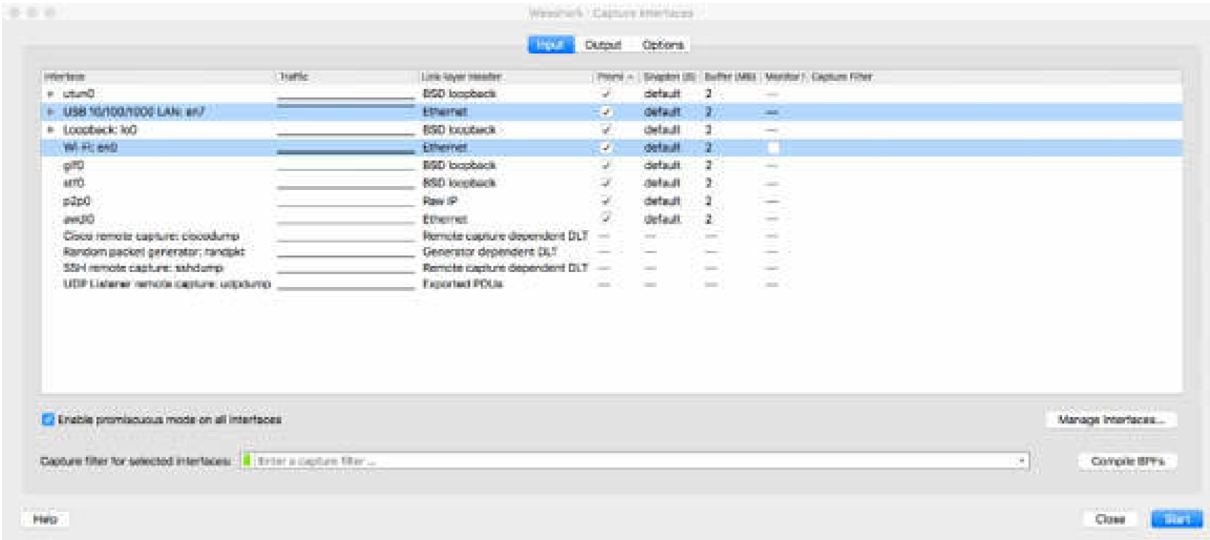
Click Start to begin the network capture.



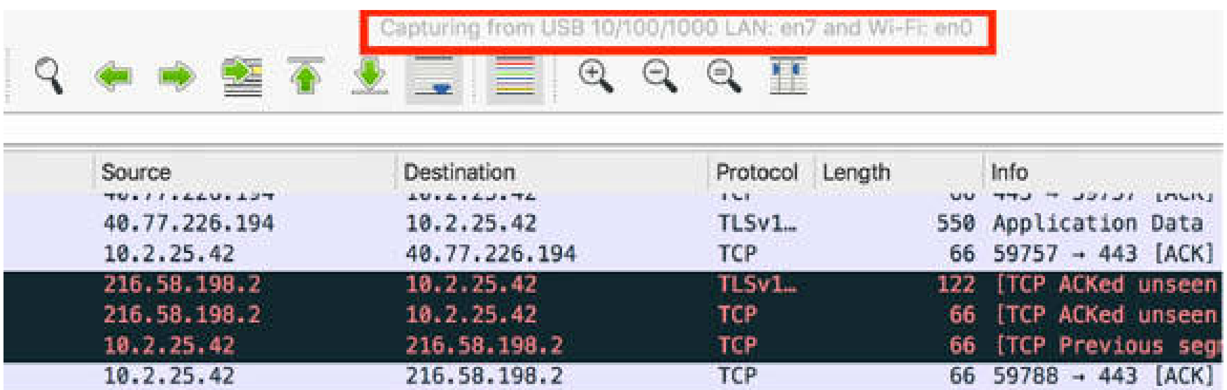
Only those packets that match the capture filter and belong to the LAN:en7 interface are displayed in the Packet List pane.

Task 4:

To capture packets on multiple network interface adapters, select Capture > Options. In the Capture Options dialog box, click the Input tab and then Ctrl+click on the network interfaces that you want to capture. Networks are captured in the same order in which they are selected. In the figure below, USB 10/100/1000 LAN:en7 and Wi-Fi:en0 networks are selected.



Click Start, and the network capture starts simultaneously on two interfaces.



Notes:

In certain situations (in cases you'll need to analyze a session from a remote server), instead of capturing packets locally, you can reach out to a capture daemon across the network. In such situations, Wireshark can connect to a Remote Packet Capture Protocol service running on a remote target platform, even if the target has no direct access to the network to be analyzed. This feature is available only in the Windows operating system. In a Linux or Unix operating system, you can get the same functionality through an SSH tunnel.

Lab 6. Wireshark Performance Optimization

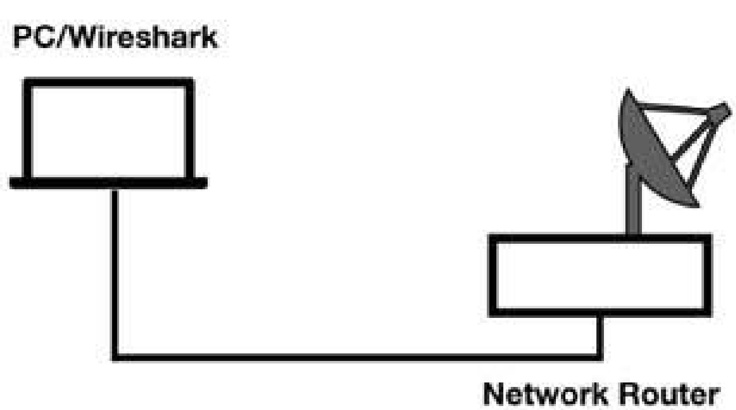
Lab Objective:
Learn how to increase Wireshark's performance.

Lab Purpose:
Set appropriate settings on Wireshark to:

- Enable memory optimization of the PC and improve Wireshark performance even in case of high packet rate in the network
- Avoid packet dropping

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

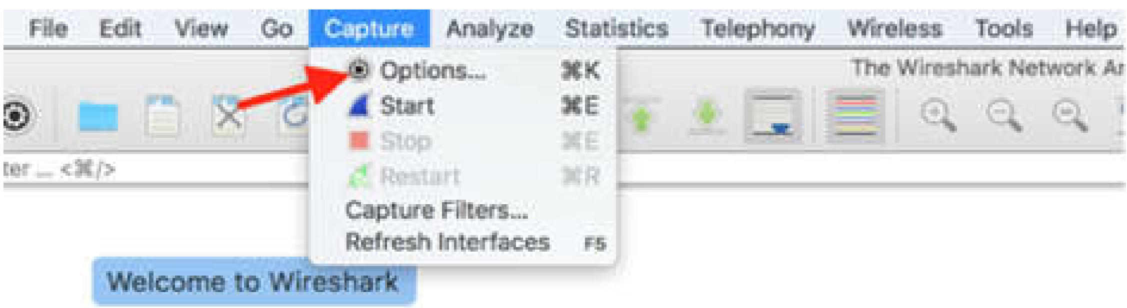


Lab Walkthrough:

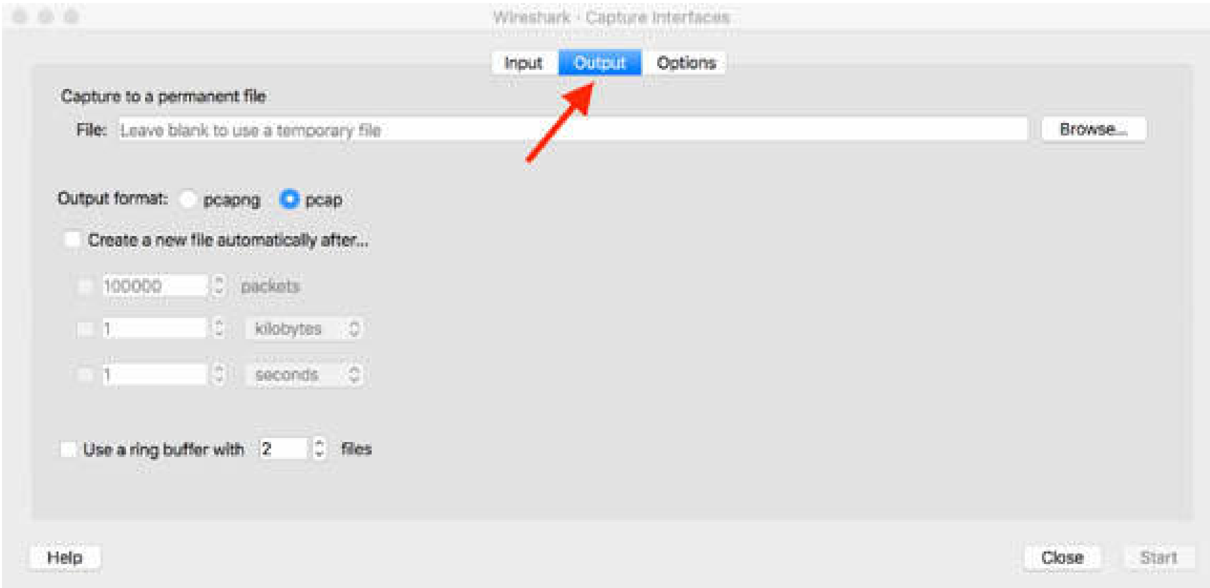
Task 1:
Verify the physical connection between the PC (equipped with Wireshark) and the network router connected to the internet.

Task 2:
For connections to a network with a high packet rate, Wireshark provides an efficient way of saving and analyzing the packets in which you can automatically save the capture log file to one or more files.

Open Wireshark, and on the main menu, select Capture > Options, as shown in the figure below.

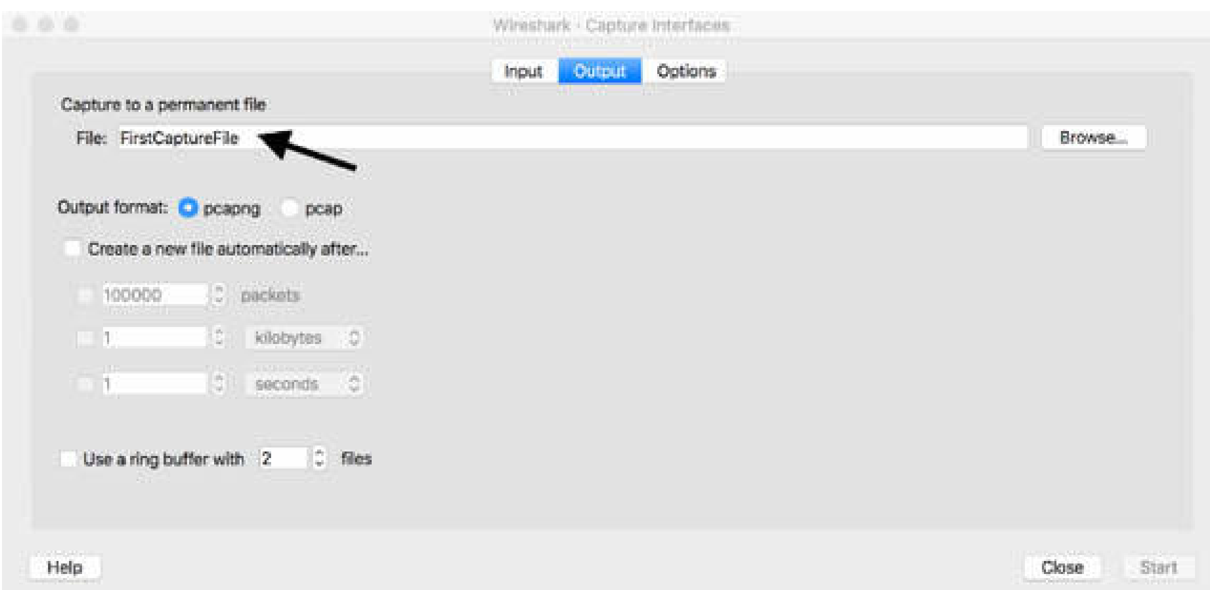


The Capture Interfaces dialog box is displayed. Click the Output tab, as shown in the figure below. The configuration parameters related to the capture output format are displayed. Note that the options and features may differ in your Wireshark version.



Task 3:

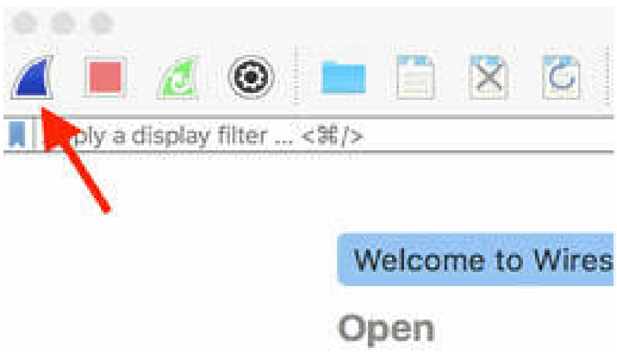
To permanently write packets to a log file, in the File box (indicated by an arrow in the figure below), enter a filename and select its location by clicking the Browse button and then click Close.



Task 4:

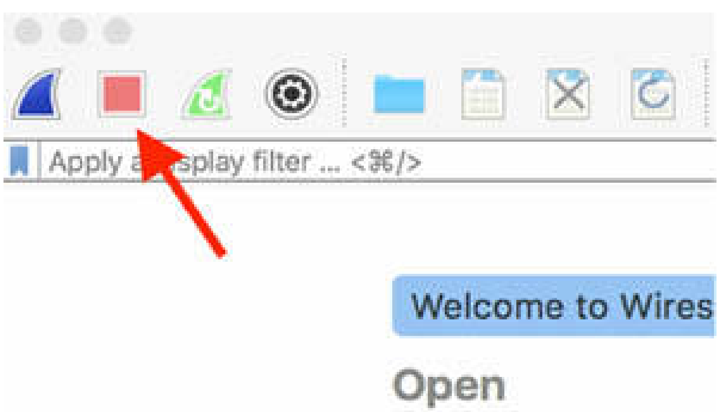
Start capturing the packets. Select the appropriate capture interface (indicated by the arrow in the figure below).

In the main toolbar, click the Start icon shown in the figure below (the shark fin).



Task 5:

When sufficient packets have been captured, stop the capture. In the main toolbar, click the Stop icon shown in the figure below.

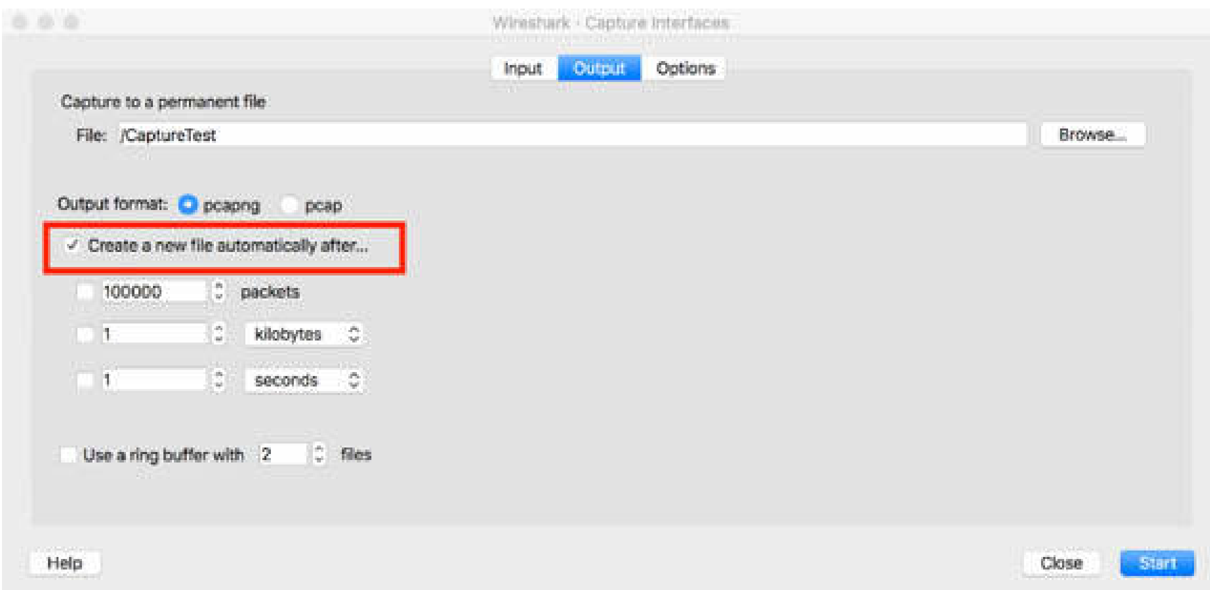


A single file named FirstCaptureFile is created and available at the location specified in Task 3. This is an efficient way to save directly to a permanent file instead of a temporary file.

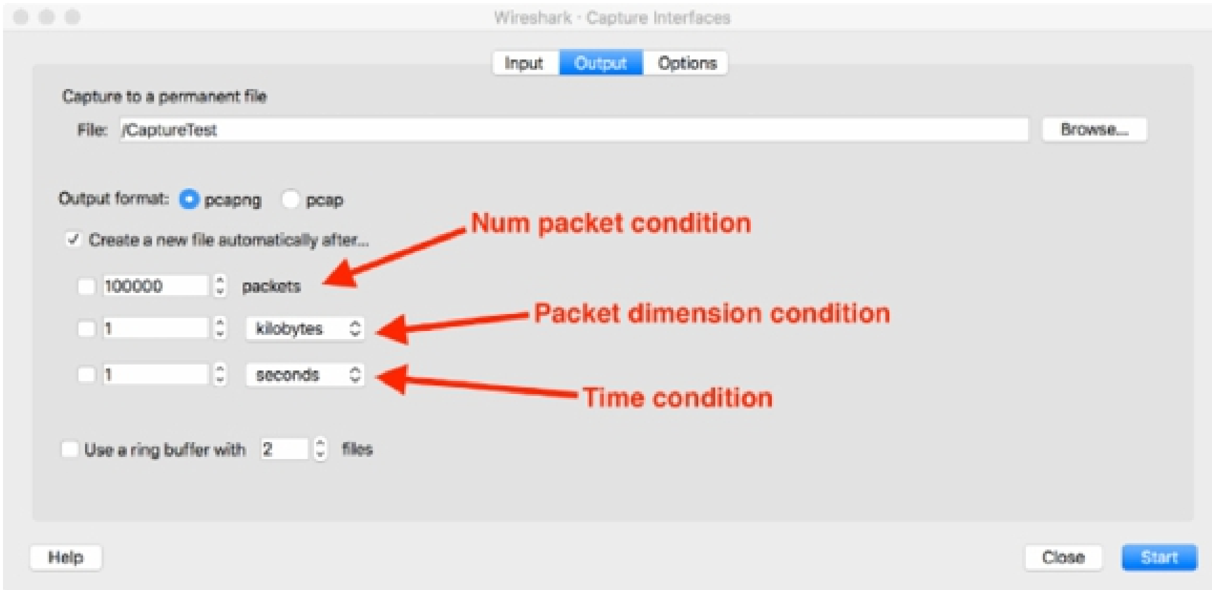
Task 6:

When the packet rate on the network is very high, the previous method creates huge files that may be difficult to manage later. To avoid this, you can split the capture file into multiple files.

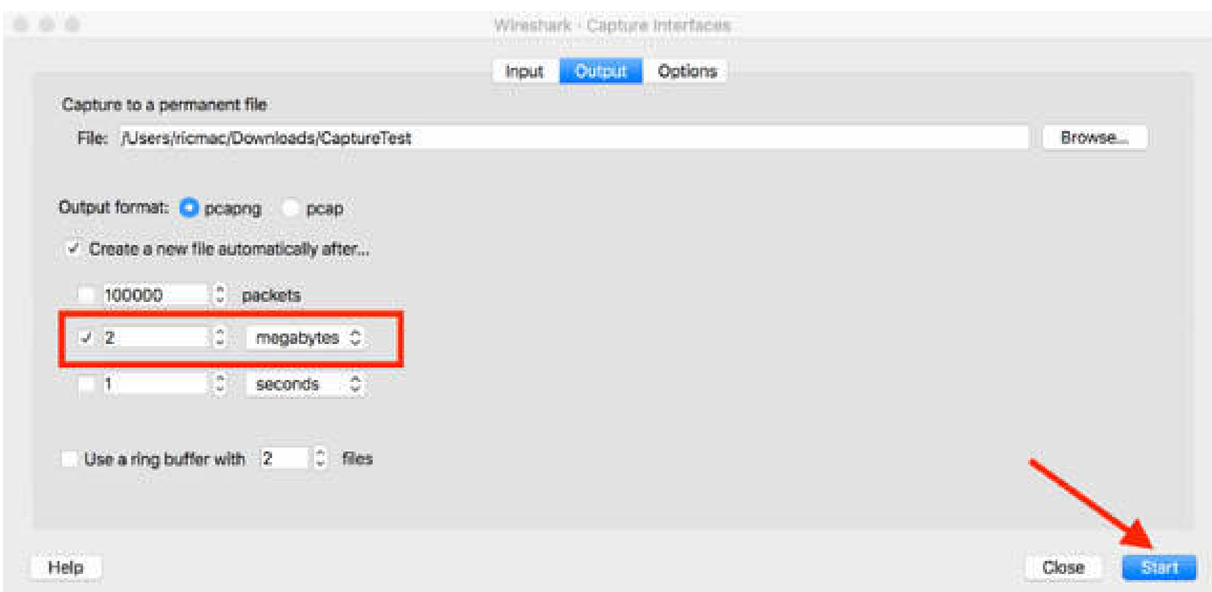
On the main menu, select Capture > Options and then click the Output tab. Select the "Create a new file automatically after" check box shown in the figure below.



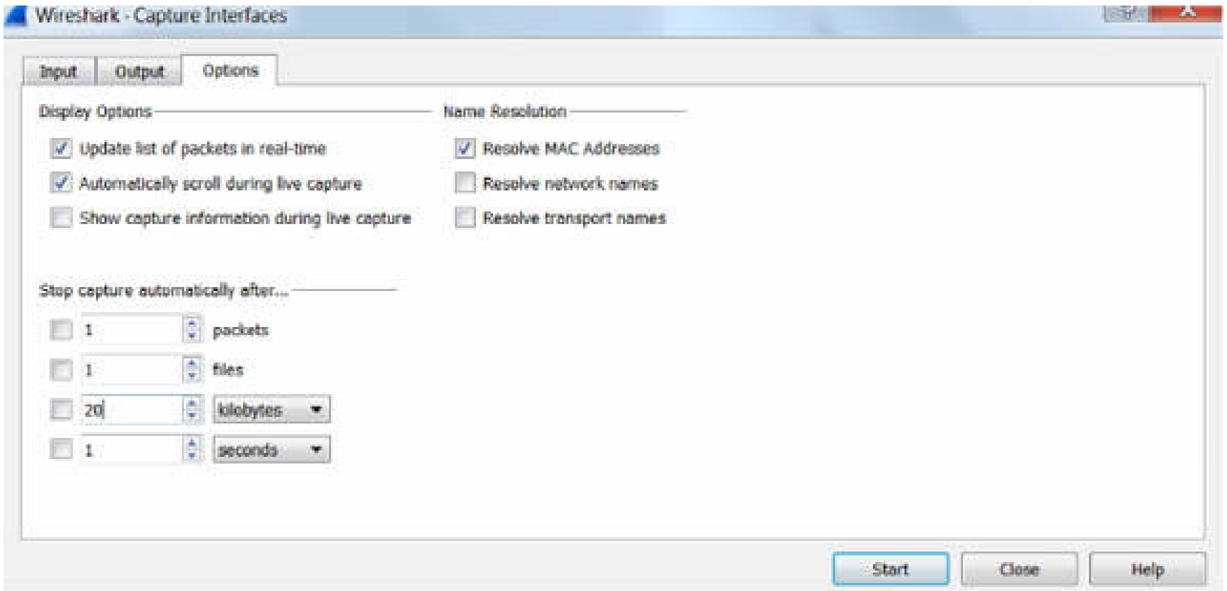
Selecting this check box enables the capture file to be split into multiple files when any of the enabled conditions, shown in the figure below, are met.



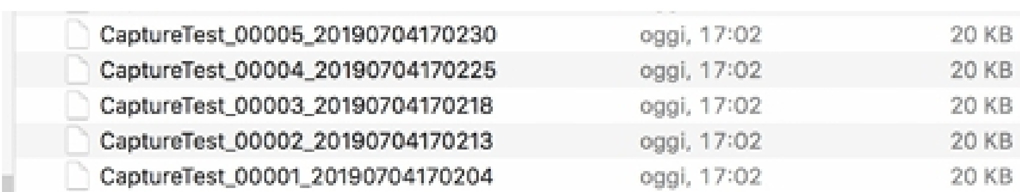
Task 7:
Enable the packet dimension condition and specify a value such as 20 Kb (2 MB is shown below so change that), and then start the packet capture.



Here is the same menu option on the Windows version of Wireshark.



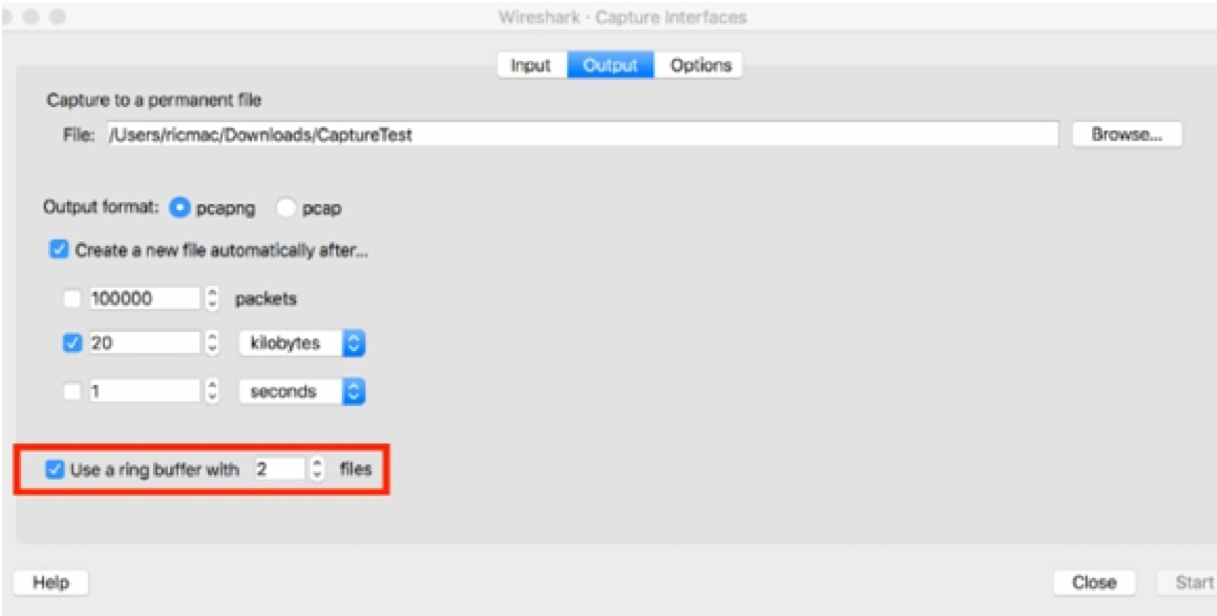
Let the capture go on for a couple of minutes and then stop it. In the folder selected in Task 3 for saving the capture files, you will find files of exactly 20Kb size and named in the <NameOfFile>_<xxxxfileNumber>_<YYYYMMDDHHmm> format, as shown in the figure below.



Task 8:

When you have limited disk space, you can specify the maximum number of files to be saved. When the capture reaches the maximum number of files (Num of files), the oldest file is overwritten until the capture is stopped.

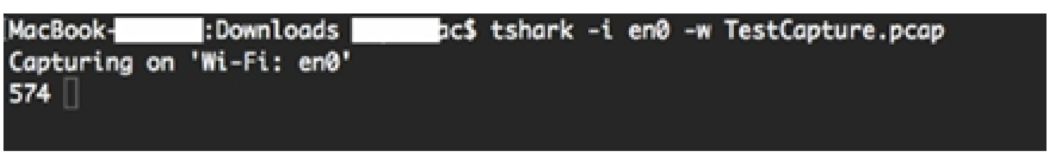
On the main menu, select Capture > Options and click the Output tab. Select the “Use a ring buffer with files” check box and specify the number of files.

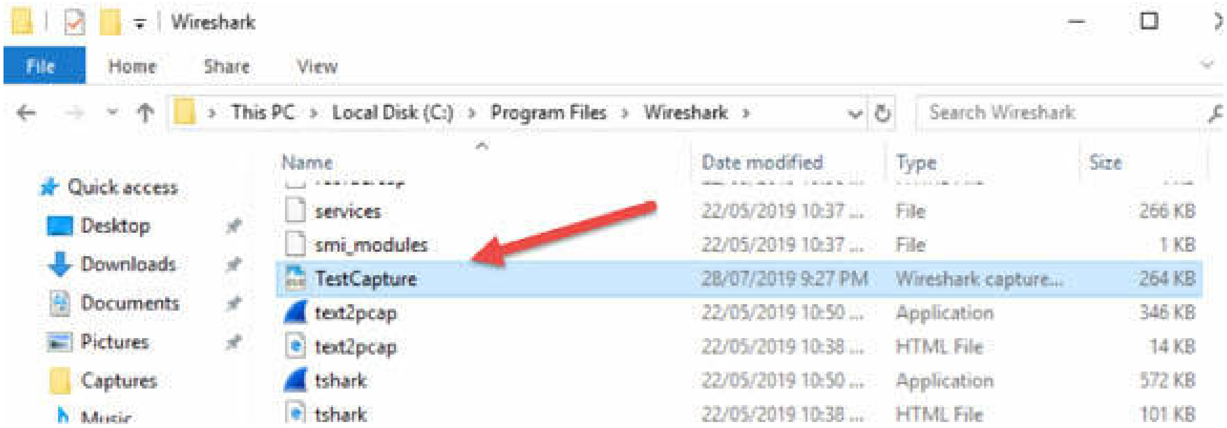


Task 9:
For higher performance, you can run Wireshark in the command line mode which supports the basic functionalities.

Open a command-line window (as an administrator) at the desired location and navigate to the directory in which you installed Wireshark; for example, C:\Program Files\Wireshark. Type the command tshark –I en0 –w TestCapture.pcap
.

A file named TestCapture is created capturing packets from the en0 network interface. After a couple of minutes, press Ctrl+C, and the capture is stopped.





Notes:
Wireshark provides a lot of command-line options and GUI customizations for better performance. For complicated network architecture, you can create a set of batch files to manage different command line windows and provide high-performance results.

If you are using Wireshark for Linux, you may need to download Wireshark using the GUI in order to access Tshark.

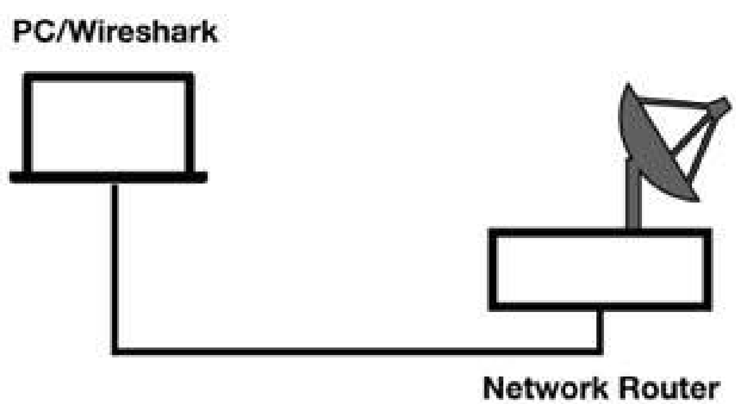
Lab 7. Wireshark Capture Filter

Lab Objective:
Learn how to build and apply a capture filter to a network interface of Wireshark.

Lab Purpose:
Wireshark is a network packet analyzer tool that enables you to view the real-time traffic on a network or to analyze the previously-saved traces. For a better packet analysis, you can create some filters (display or capture) depending upon the packet's content or type. Capture filters allow you to save only those packets that are dedicated to a specific network interface and meet the rules of predefined filters.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

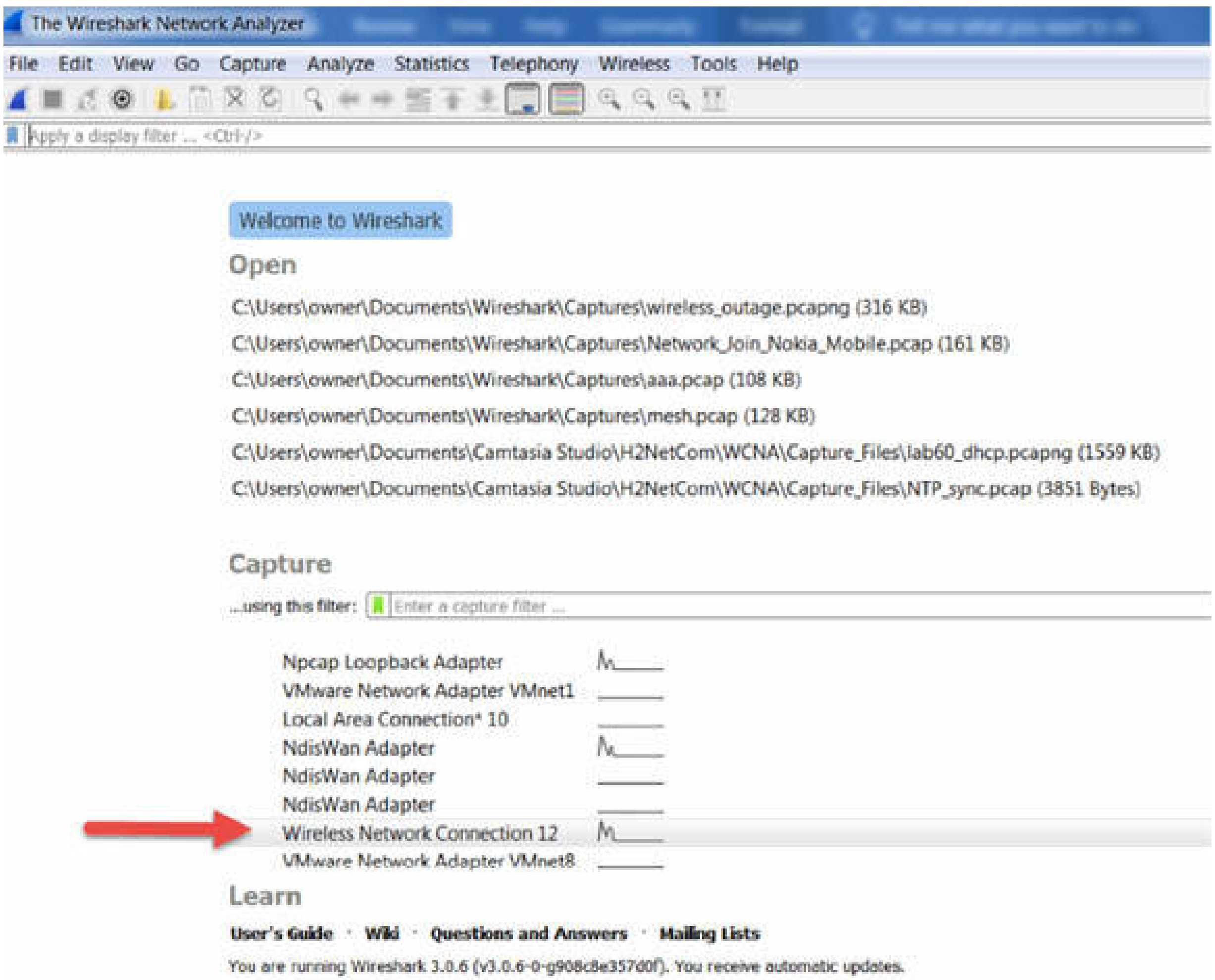
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



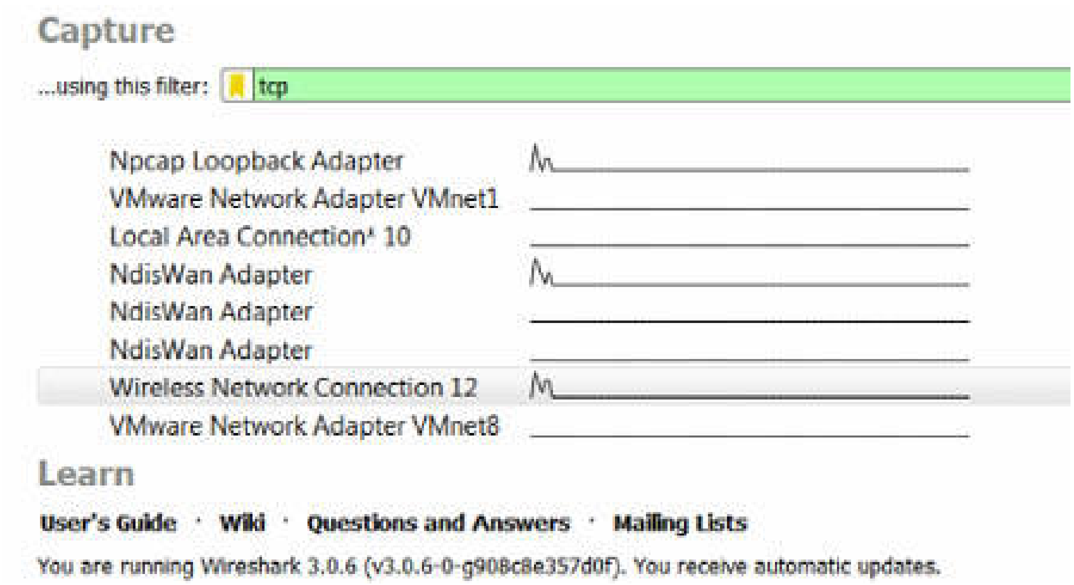
Lab Walkthrough:

Task 1:
Verify the physical connection between the PC (equipped with Wireshark) and the network router connected to the internet.

Task 2:
Open Wireshark, and in the main window, click the interface that you are using as a connection with the router (such as Wireless Network Connection 12), as shown in the figure below.

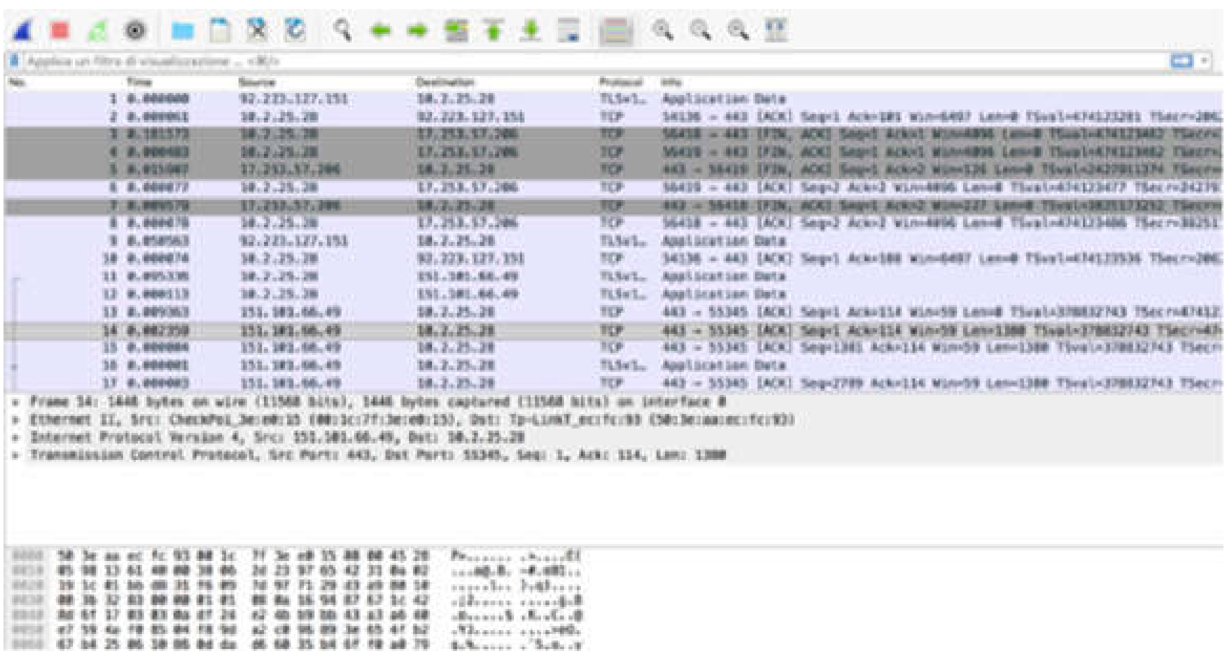


Task 3:
In the Capture Filter box shown in the figure above and below, enter a network filter name such as tcp , and press the Return key.



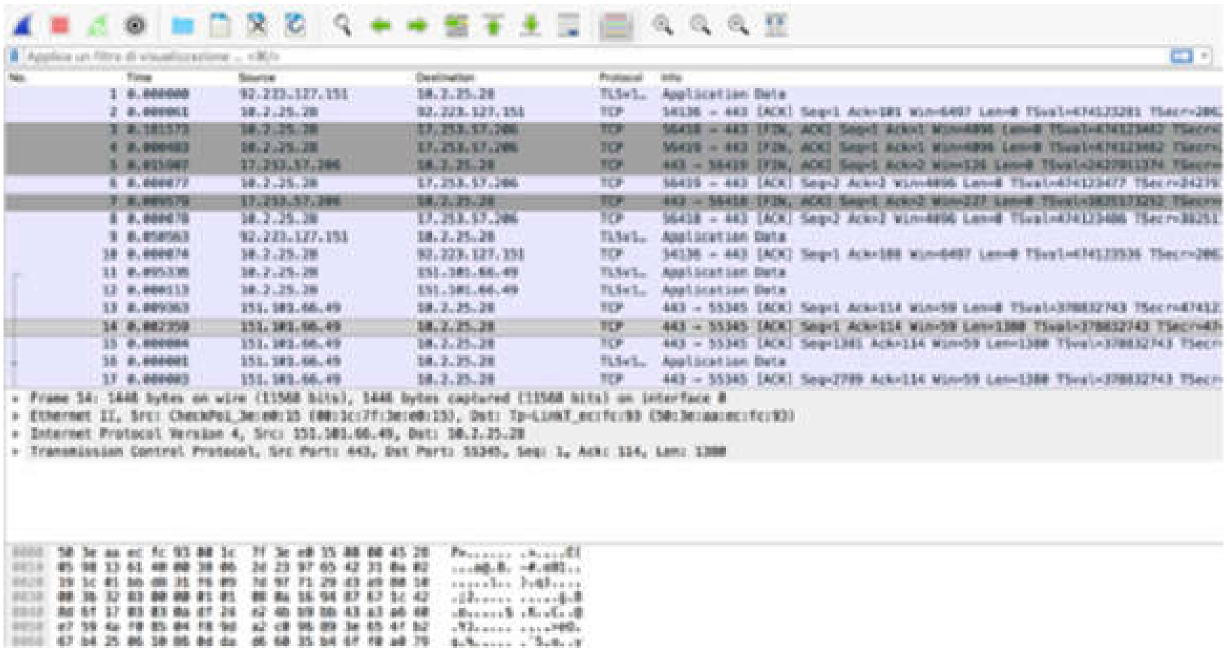
Task 4:

The Packet List pane, shown in the figure below, is displayed. It contains the live packets of type TCP captured from the network.



Task 5:

Click the Stop icon shown in the figure below to stop the live capture and then use the File menu to save the capture in a file (such as TraceFile.pcap).



Notes:

You can use a wide range of packet capture filters to save the desired network packet file. Repeat Task 2 to Task 5 to gain more confidence in using the filters. A few examples of filters are:

- host 172.18.5.4—Captures only the traffic to or from IP address 172.18.5.4
- net 192.168.0.0 mask 255.255.255.0—Captures traffic to or from a range of IP addresses
- port 53—Captures only the DNS (port 53) traffic

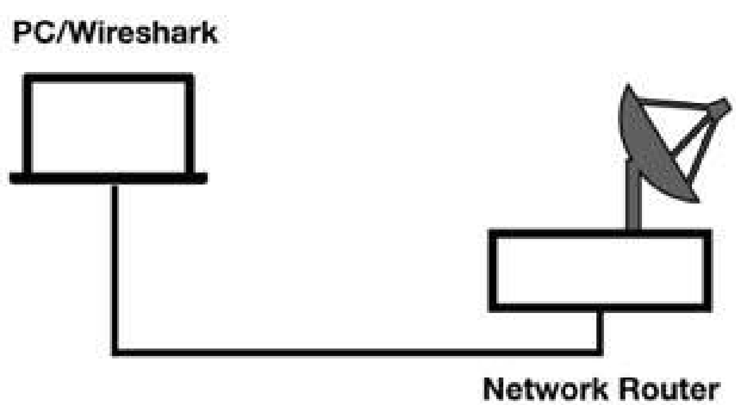
Lab 8. Wireshark Protocol-Addresses Capture Filters

Lab Objective:
Learn how to build and apply capture filters by protocol type or addresses.

Lab Purpose:
Wireshark allows the creation of capture filters based on the packet type and IP/MAC addresses. Each capture filter should be created based on the most relevant feature of the packet to be captured and the easiest and the most intuitive way to create it.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

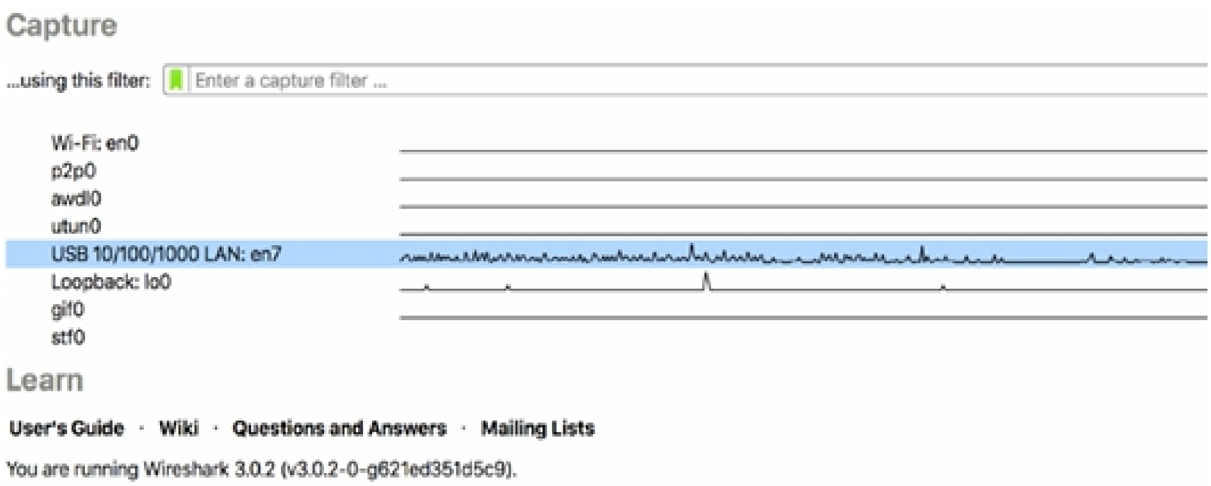
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

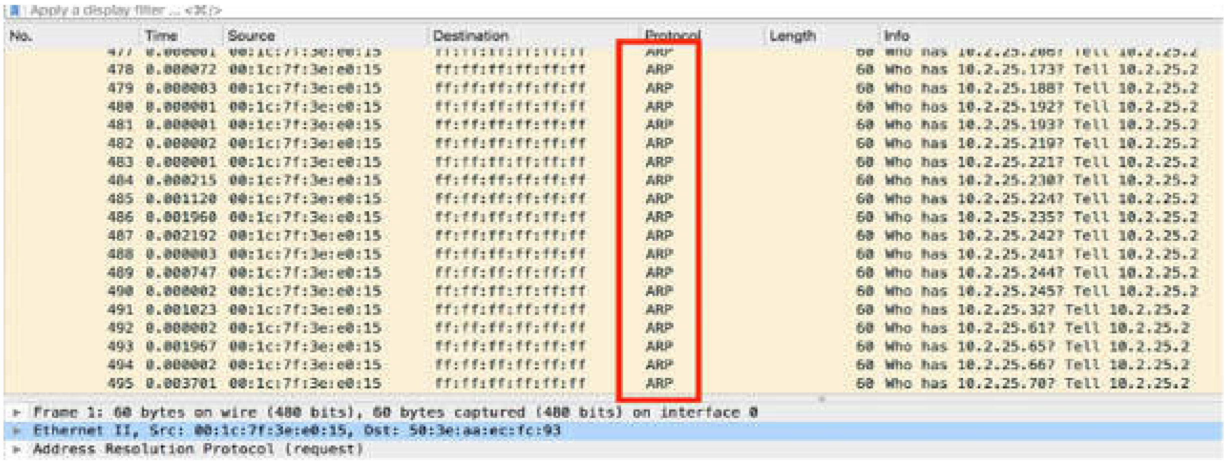
Task 1:
Verify the physical connection between the PC (equipped with Wireshark) and the network router connected to the internet.

Task 2:
Open Wireshark, and in the main window, click the interface that you are using as a connection with the router (such as en7 LAN interface), as shown in the figure below.



Task 3:
In the Capture Filter box, enter a protocol filter name such as arp , and press the Return key.

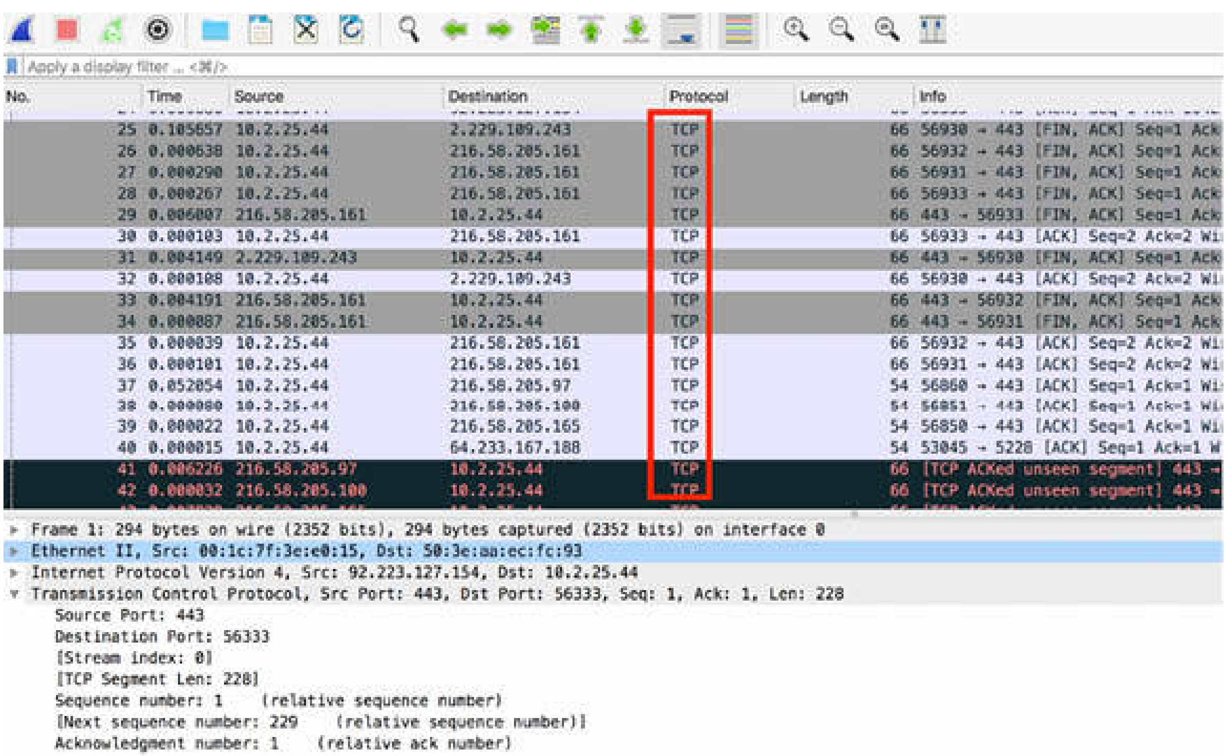
The Packet List pane, shown in the figure below, is displayed. It contains the live ARP packets captured from the network. Before pinging an IP address on your network, you may need to clear your ARP cache by running the arp -a -d command as an administrator.



Task 4:

Close the main window and in the Capture Filter box, enter a protocol filter name such as tcp and then press the Return key.

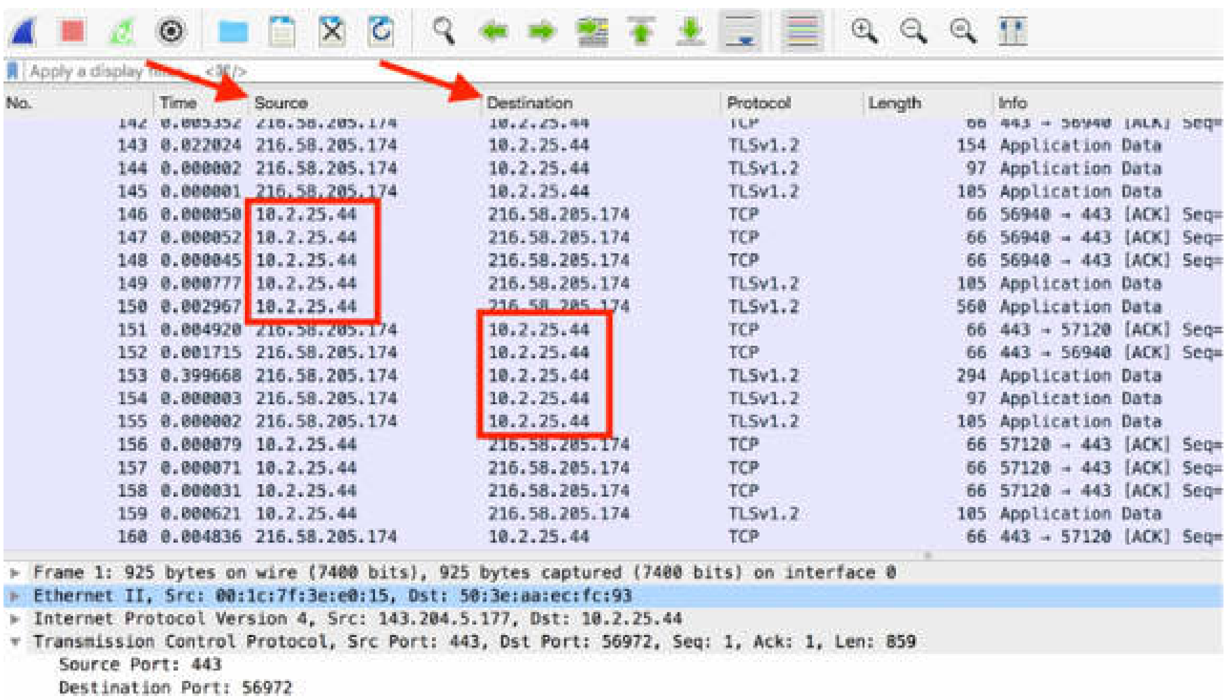
The Packet List pane, shown in the figure below, is displayed. It contains the live TCP packets captured from the network.



Task 5:
You can also apply filters based on the IP addresses. Close the main window, and in the Capture Filter box, enter net 10.2.25.44 , and press the Return key.

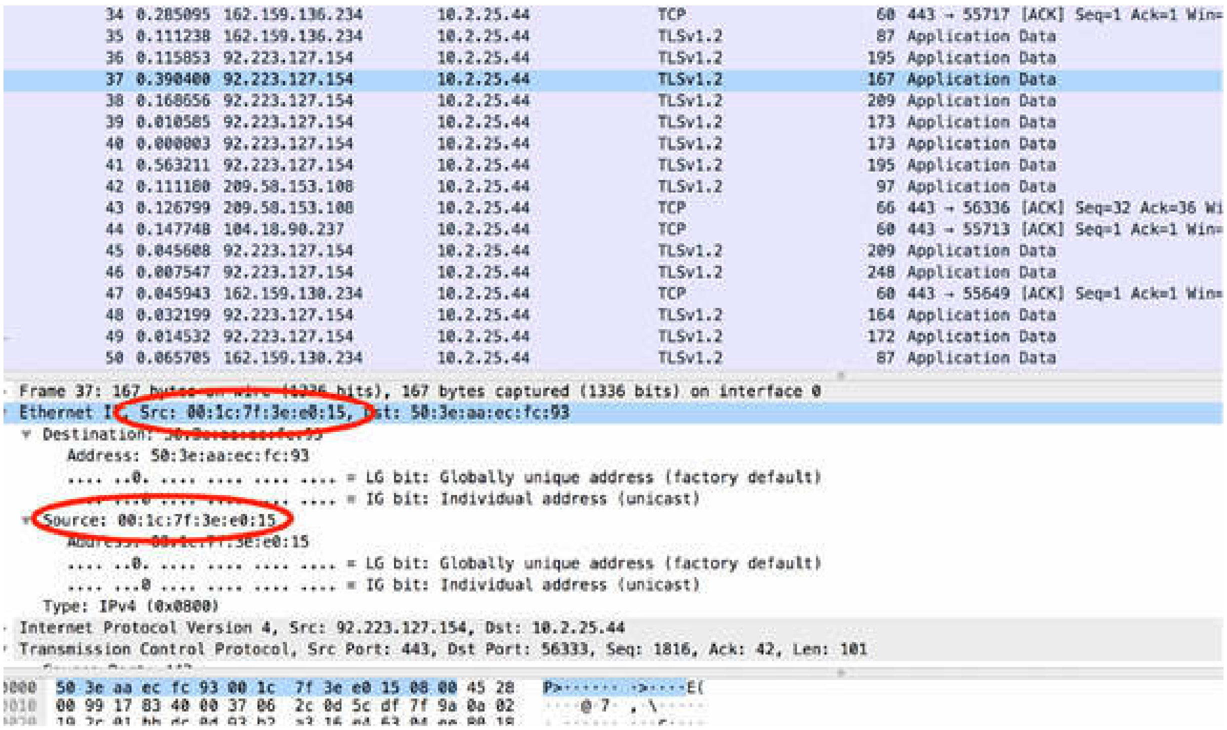
In this example, 10.2.25.44 is used as the default gateway IP address. Make sure you use your default gateway IP address, which can be found by using the ipconfig /all command in the Windows operating system.

The Packet List pane, shown in the figure below, is displayed. It contains the live packets captured on the network coming from or directed to IP address 10.2.25.44.



Task 6:
You can also apply filters based on the MAC addresses. Close the main window, and in the Capture Filter box, enter ether src 00:1c:7f:3e:e0:15 , and press the Return key.

In this example, 00:1c:7f:3e:e0:15 is used as the default gateway Ethernet MAC address. Make sure you use your default gateway Ethernet MAC address.



Notes:
You can use a wide range of packet capture filter to capture the desired network packets to a file. To gain more confidence in using the capture filters, repeat the previous tasks by using both IP addresses (combination/ranges) and MAC addresses.

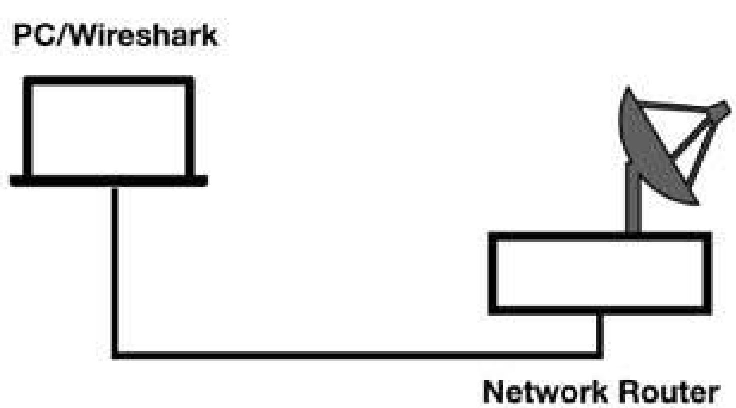
Lab 9. Wireshark Advanced Capture Filters

Lab Objective:
Learn how to build and apply complex capture filters by field content and by using operators.

Lab Purpose:
Wireshark allows you to create complex capture filters combinations by using operators and a packet's detailed content type. Each capture filter should be created based on the most relevant feature of the packet to be captured and the easiest and the most intuitive way to create it.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

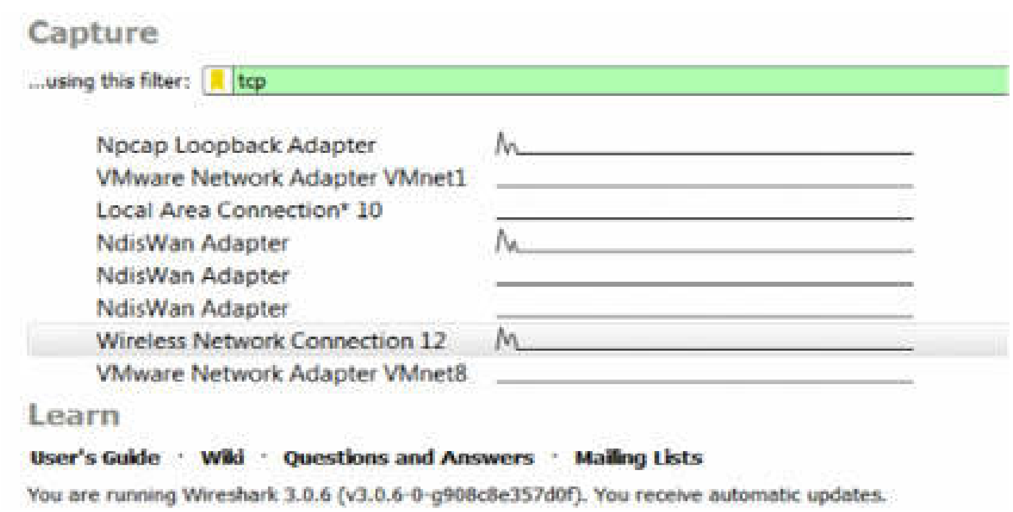


Lab Walkthrough:

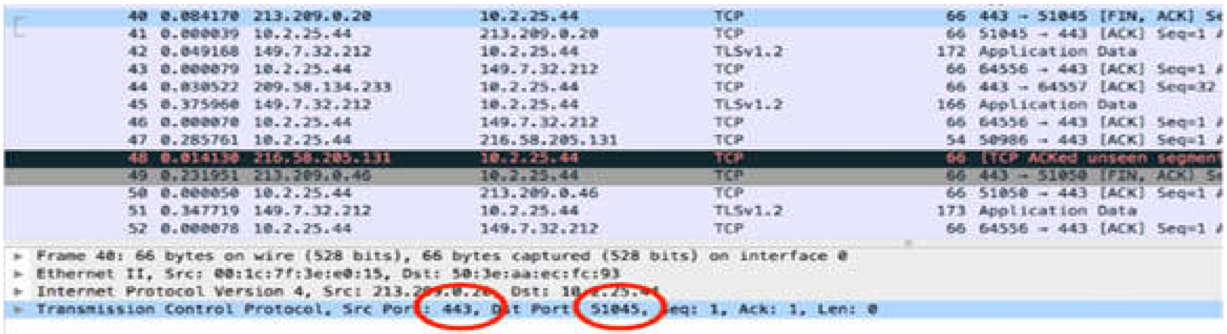
Task 1:
Verify the physical connection between the PC (equipped with Wireshark) and the network router connected to the internet.

Task 2:
Open Wireshark, and in the main window, click the interface that you are using as a connection with the router (such as wlan0 wireless interface), as shown in the figure below.

Task 3:
In the Capture Filter box shown in the figure below, enter a protocol name such as tcp , and press the Return key.



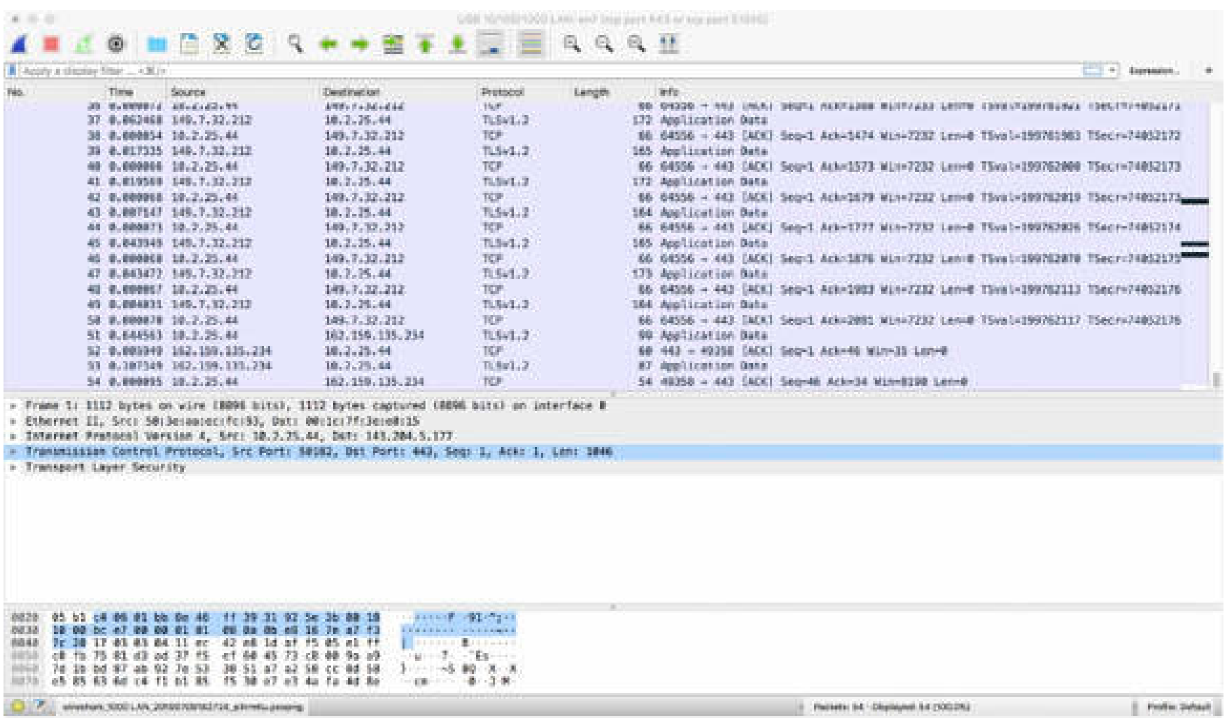
Task 4:
Capture the traffic on the specified interface for a few minutes and then stop the capture and inspect the Packet List pane. The details of the packet, such as source and destination ports, are displayed as shown in the figure below.



Task 5:

Close the main window, and in the Capture Filter box, enter a combination of protocol type filter name and details such as tcp port 443 or tcp port 51045 , and press the Return key.

The Packet List pane shown in the figure below is displayed. Only those packets that meet the first or the second condition are saved (because the logical OR condition is there), as shown in the figure below.

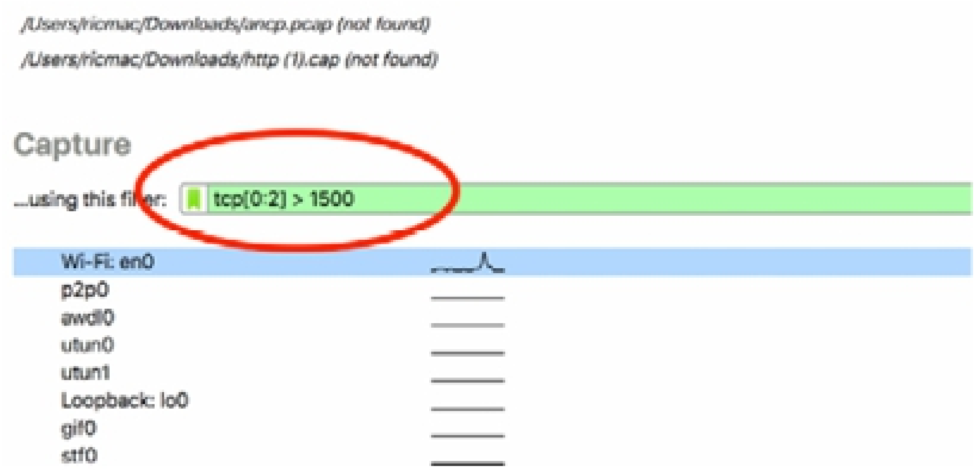


Task 6:

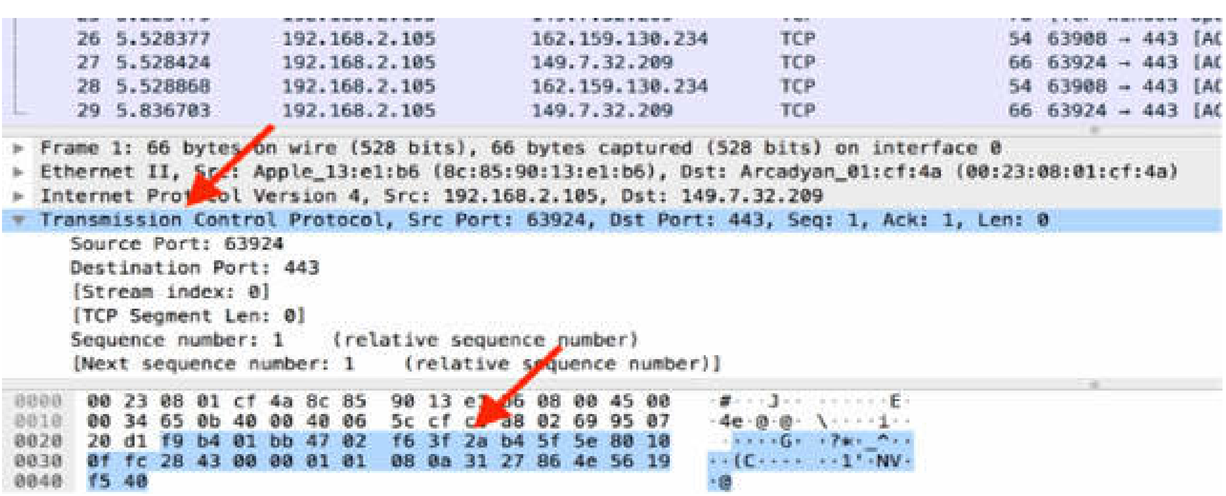
You can repeat the actions executed in Task 5 by specifying a series for a capture filter such as tcp portrange 1501–1549 . In this example, a port range of the protocol type TCP is specified.

Task 7:

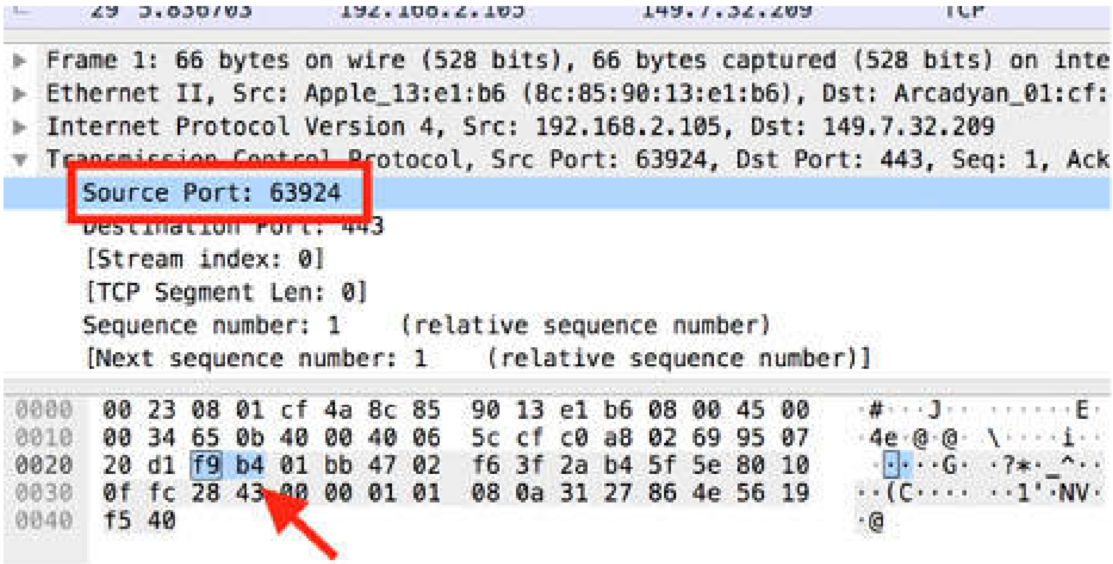
You can also specify packet bytes content for a capture filter such as tcp[0:2] > 1500 . In this example, the capture filter specifies a lower limit for the TCP port.



In the Packet List pane, only the TCP packets that have the TCP port greater than 1500 are captured. When you click a specific field in the TCP tree, the TCP byte stream is highlighted in the Packet Bytes pane.



When you click the first two bytes ([0:2]) of the TCP byte stream (with value "f9 b4"), indicated by the arrow in the figure below, the TCP source port is highlighted. This confirms that only port numbers greater than 1500 are selected, and the source port value is 63924, which is f9 b4 in hex.



You can create a wide range of capture filters for inspecting the bytes stream. The capture filters can be related to the Ethernet level (ether[x:y]), IP level (ip[x:y]), TCP level (tcp[x:y]), or other protocol types.

Notes:

Capture filters (such as tcp port 80) should not be confused with display filters (such as tcp.port == 80). Capture filters are much more limited and are used to reduce the size of a raw packet capture. Display filters are used to hide some packets from the packet list.

Capture filters are set before starting a packet capture and cannot be modified during the capture. Display filters, on the other hand, do not have this limitation, and you can modify them during the capture.

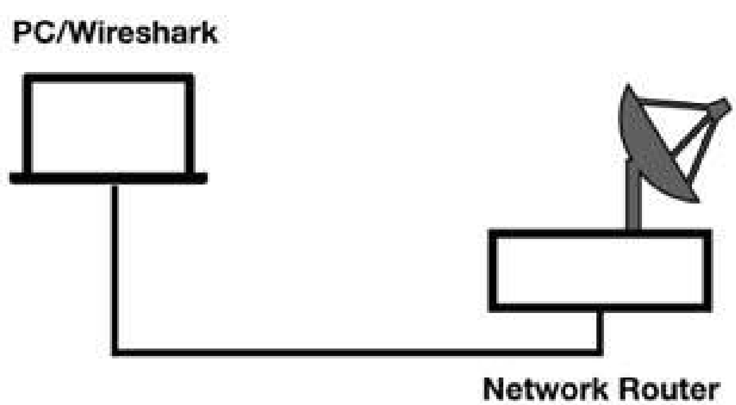
Lab 10. Customize User Interface Settings

Lab Objective:
Learn how to create and apply user interface settings to the Wireshark GUI and how to restore it to the default settings.

Lab Purpose:
Wireshark is a network packet analyzer tool with a GUI that can be customized to add useful and desired appearance to columns, colors, and layout.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

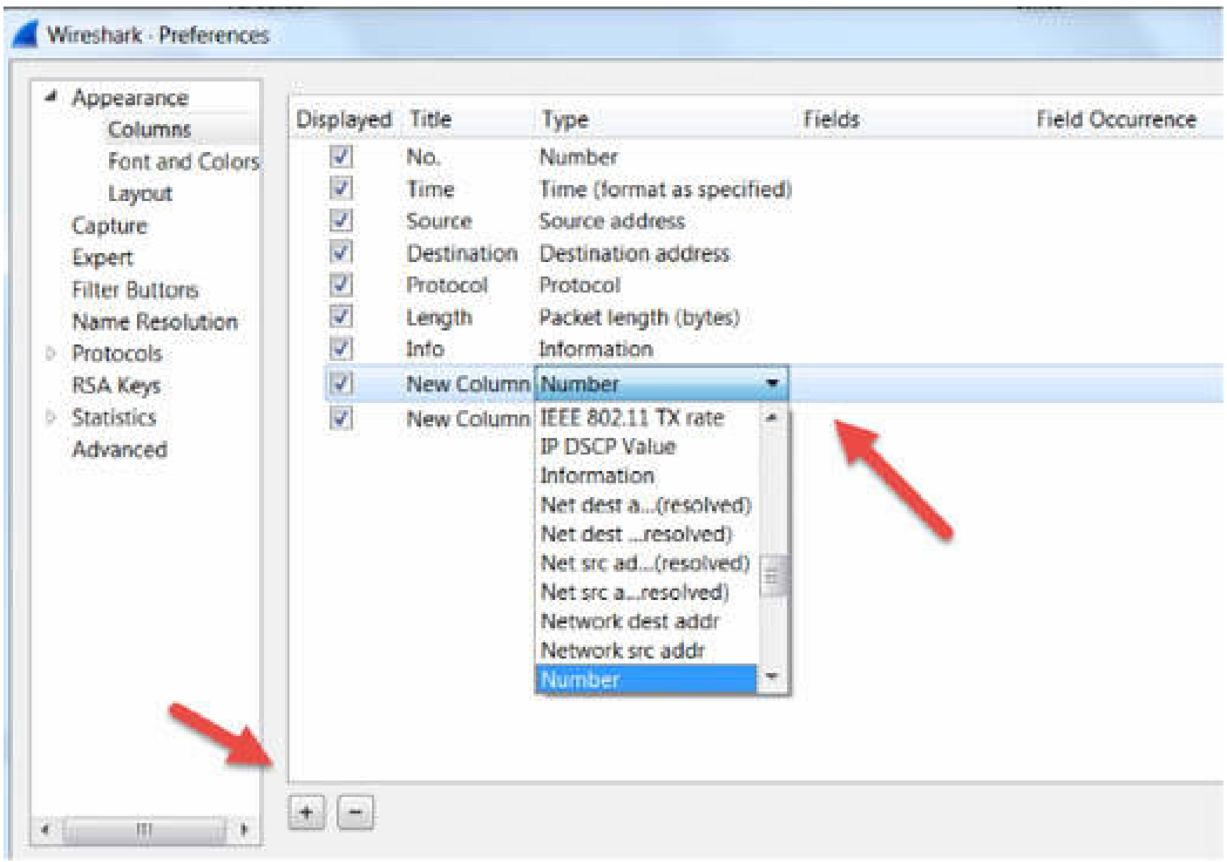
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
Open Wireshark, and on the main menu, select Edit > Preferences. In the tree view, click Appearance > Columns.

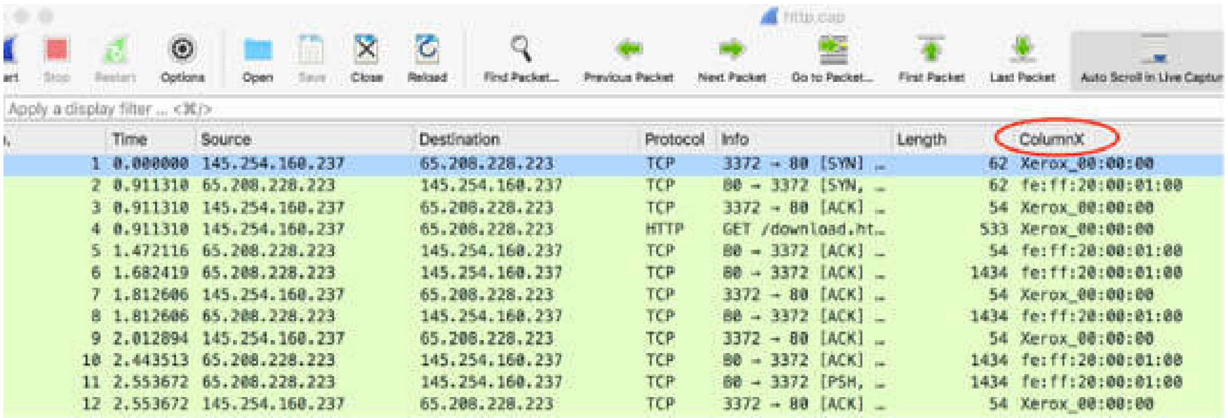
Click the add (+) icon to add a custom column. Double-click the column title and enter a title name, such as ColumnX. Double-click the column type and from the drop-down list, select an item such as ‘hardware src addr physical source mac address’.



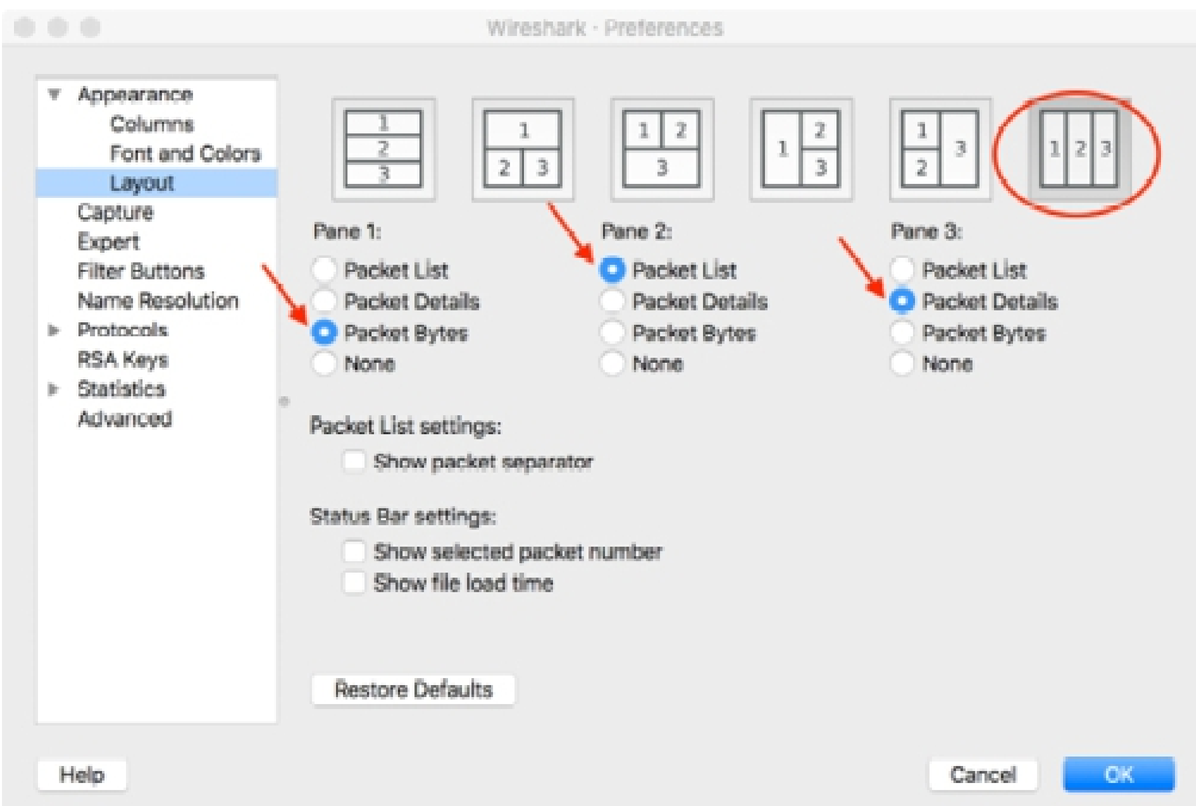
Click OK to apply the new settings.

Task 2:
Download the free sample capture http.pcap to your PC from **https://wiki.wireshark.org/SampleCaptures#TCP** , and then open the downloaded file in Wireshark.

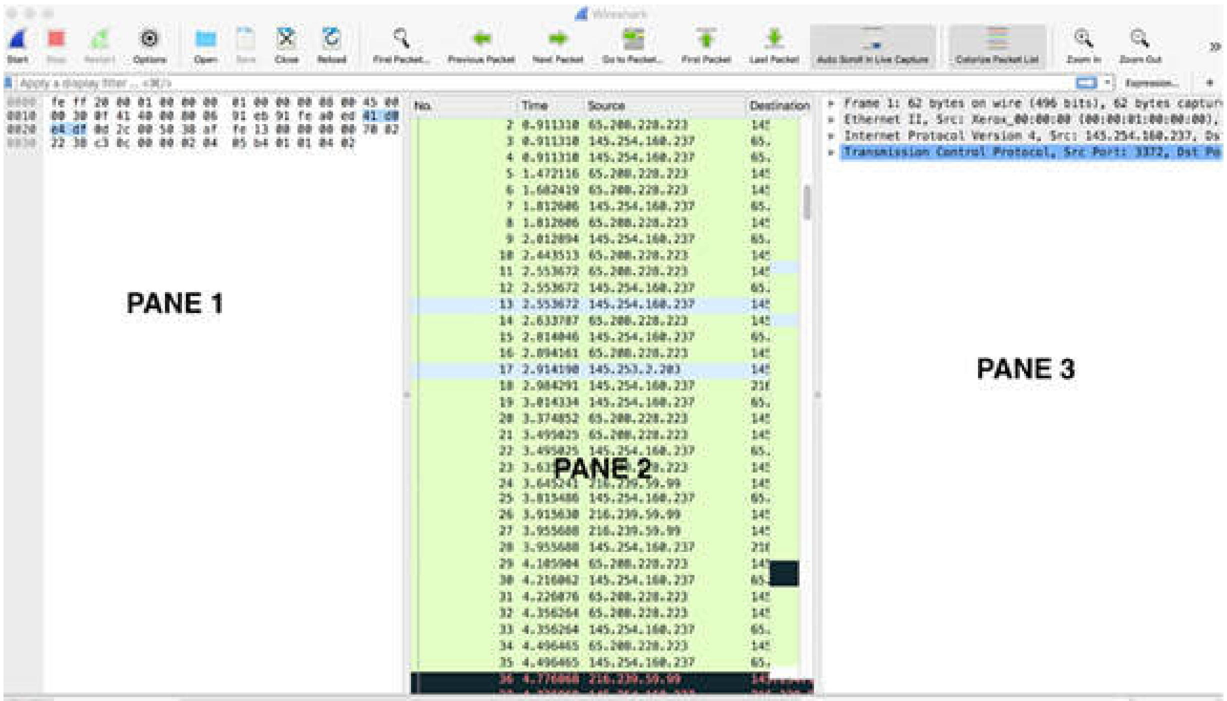
Task 3:
The newly added column is displayed in the Packet List pane, as shown in the figure below.



Task 4:
On the main menu, select Edit > Preferences and click Layout. Change the selection of Pane 1, Pane 2, and Pane 3, as shown in the figure below.

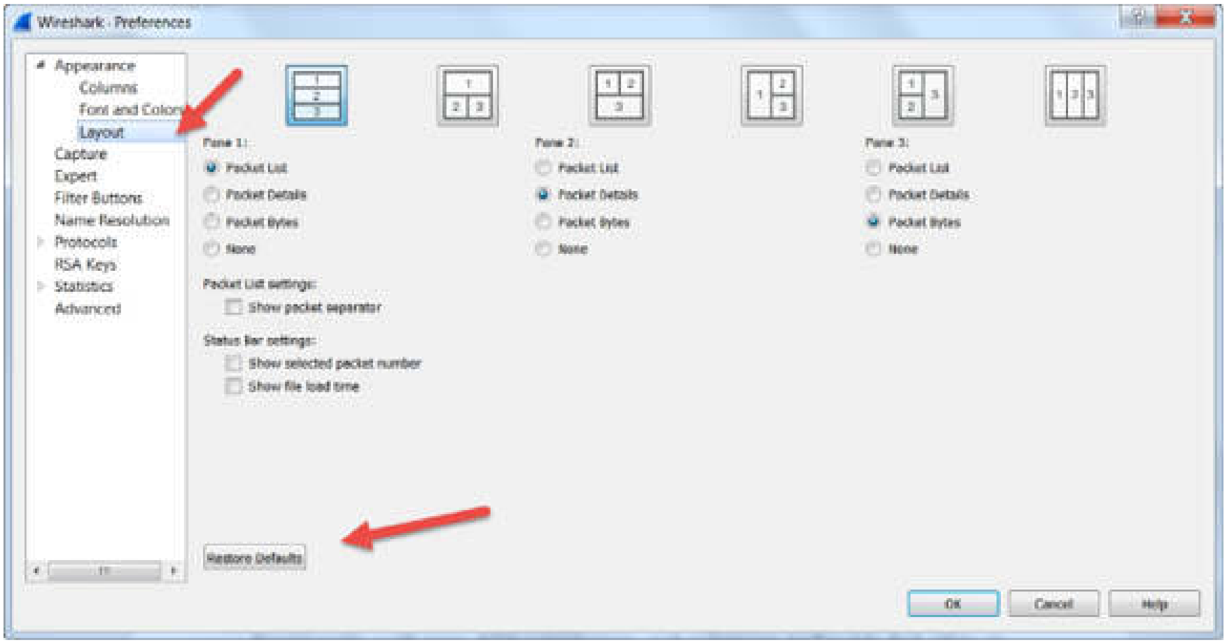


Click OK to apply the modifications, The Wireshark Packet section will look as shown in the figure below.



Task 5:

To apply the default layout, on the main menu, select Edit > Preferences and then click Layout > Restore Defaults.



To restore the default settings of the column added in Task 3:

1. On the main menu, select Help > About Wireshark > Folders.
2. Find the personal configuration folder and double-click the related location.
3. The folder is opened.
4. Locate the preference file and back it up.
5. Close Wireshark.
6. Remove the preference file.
7. Restart Wireshark.

The custom column added in Task 1 should have disappeared.

Notes:
You can enable a wide range of GUI customizations—such as language, toolbar style, font, colors—in the Appearance pane. You can also revert each customization to the default settings when you don't need it.

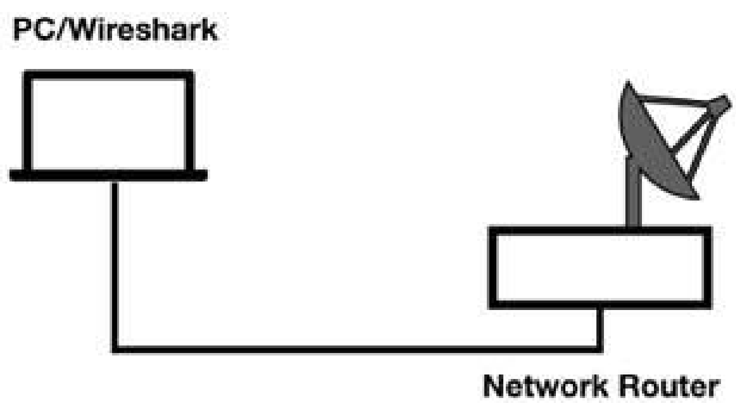
Lab 11. Custom Capture Preferences

Lab Objective:
Learn how to create and apply custom capture settings to the Wireshark and how to restore it to the default settings.

Lab Purpose:
Wireshark is a network packet analyzer tool that allows you to customize the capture settings and save them.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

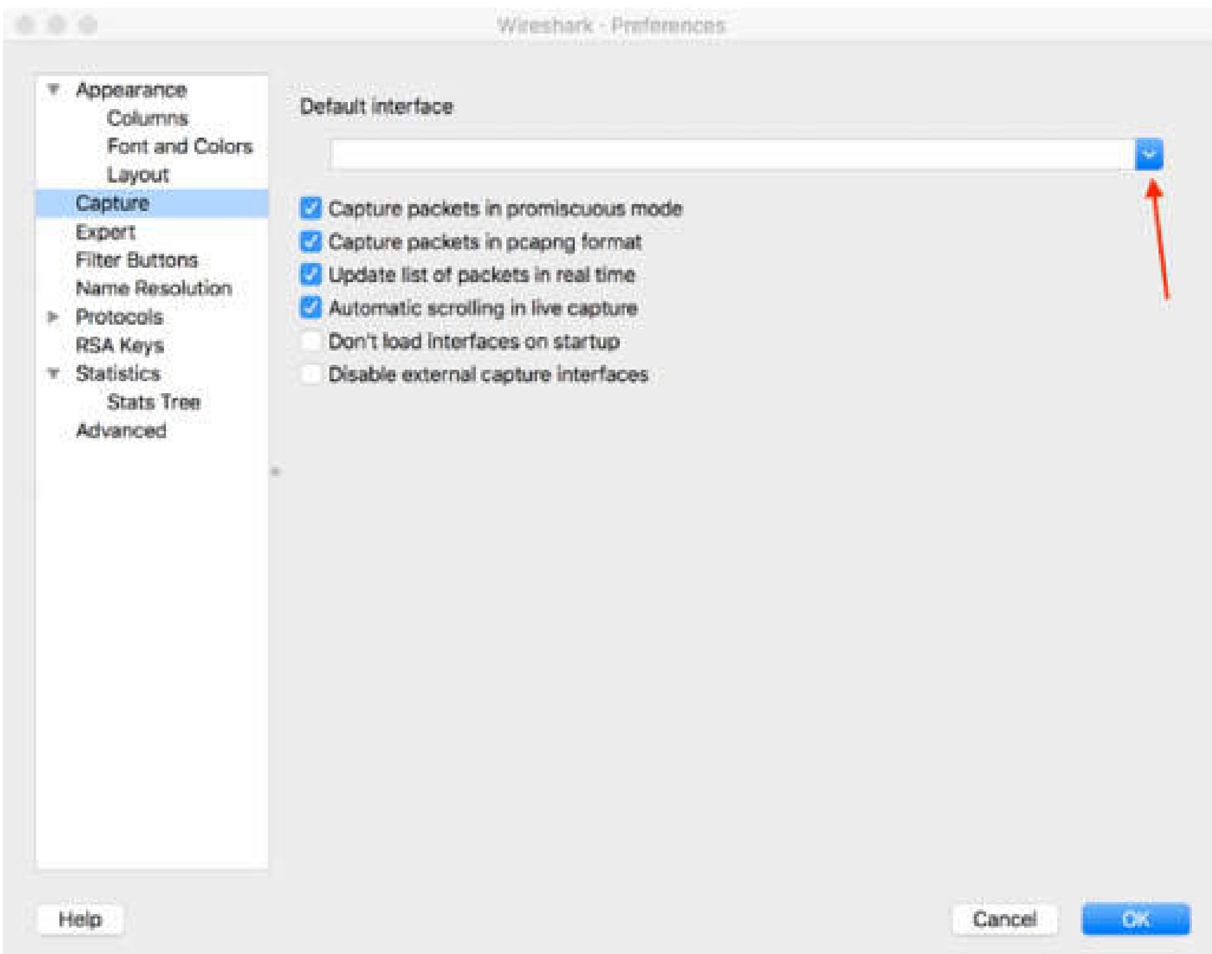


Lab Walkthrough:

Task 1:

Open Wireshark, and on the main menu, select Edit > Preferences. In the tree view, click Capture.

The Preferences dialog box, shown in the figure below, is displayed. We completed this lab on Wireshark for MAC so your layout may differ.

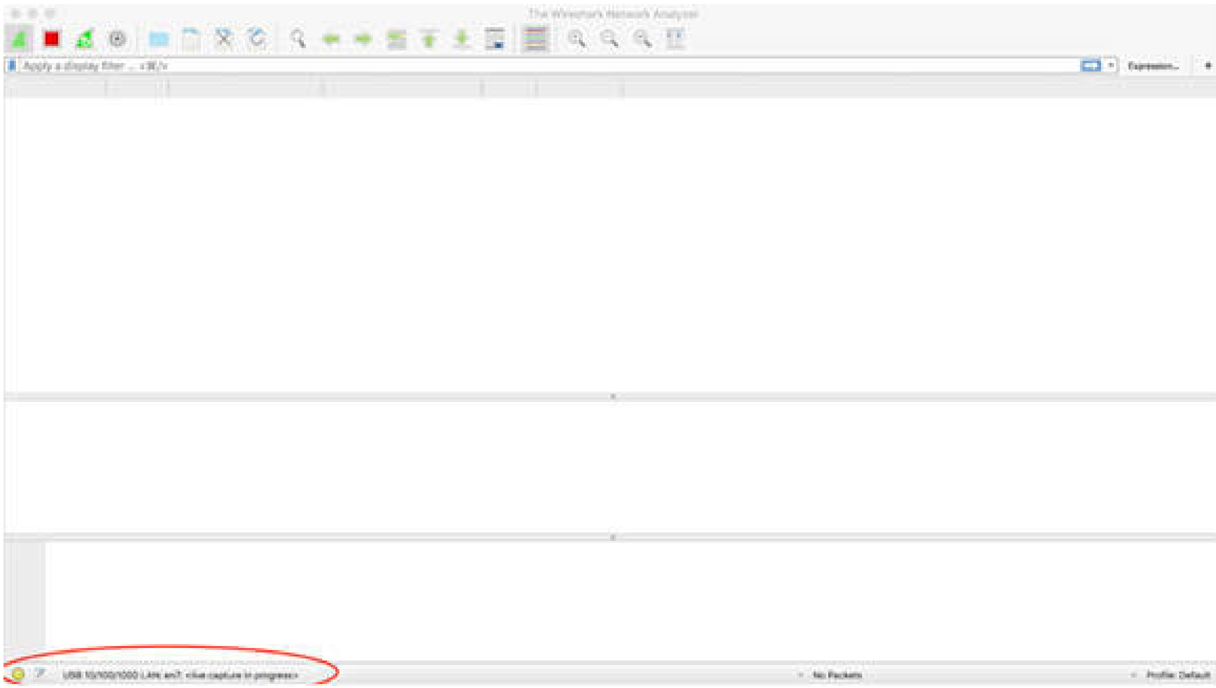


Set the default interface and restart Wireshark. Click Capture and then click Start. Setting the default interface allows you to directly start capturing traffic on the specified interface.

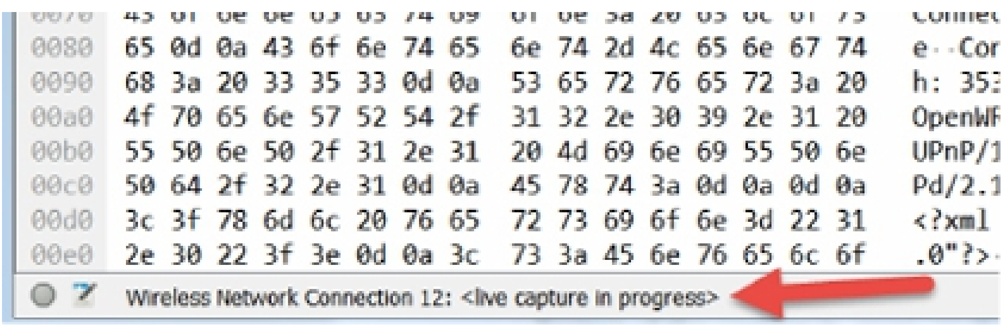
Task 2:

On the main menu, select Edit > Preferences, and in the tree menu, click Capture.

Clear the “Update list of packets in real time” check box, and start capturing. No packets are displayed in the main interface until capturing is stopped, as shown in the figure below.



Here is the Statusbar magnified (my interface differs as you can see).

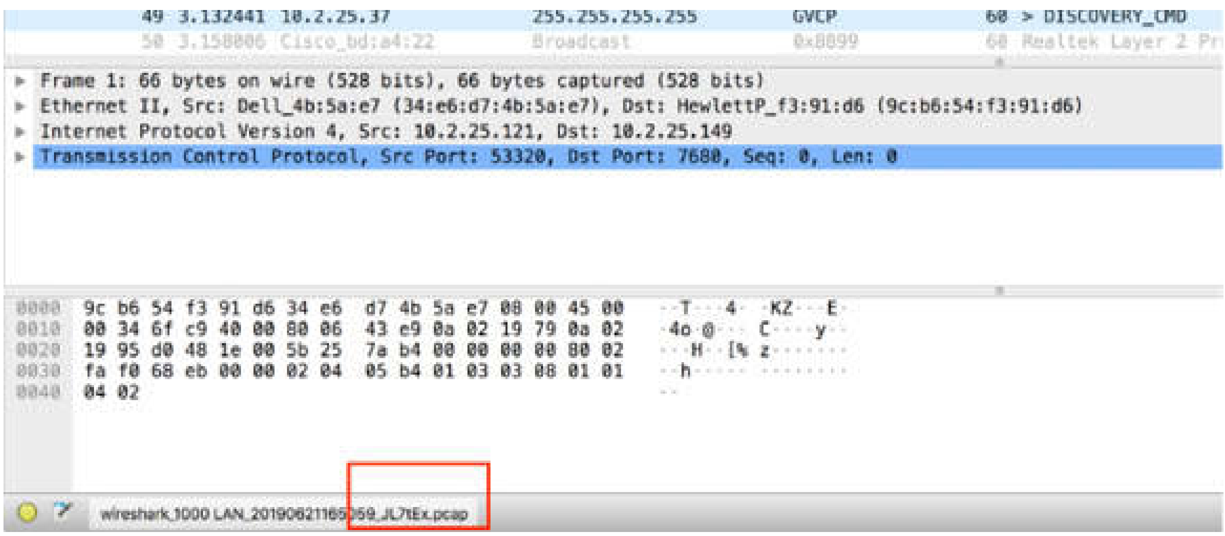


On the main menu, select Edit > Preferences, and in the tree menu, click Capture.

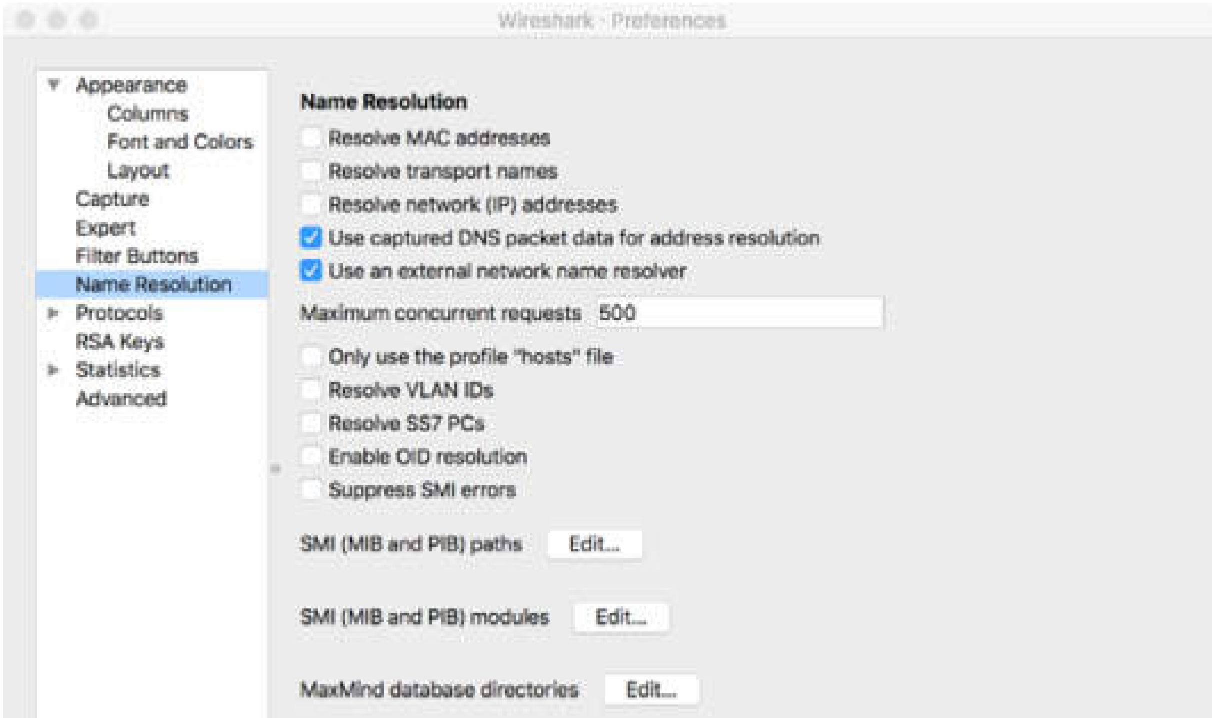
Select the “Update list of packets in real time” check box and clear the “Automatic scrolling in live capture” check box and start capturing. During the capture in progress, the packet selected is the first packet of the session and the auto-scroll is not enabled.

Task 3:
On the main menu, select Edit > Preferences and in the left tree menu, click Capture.

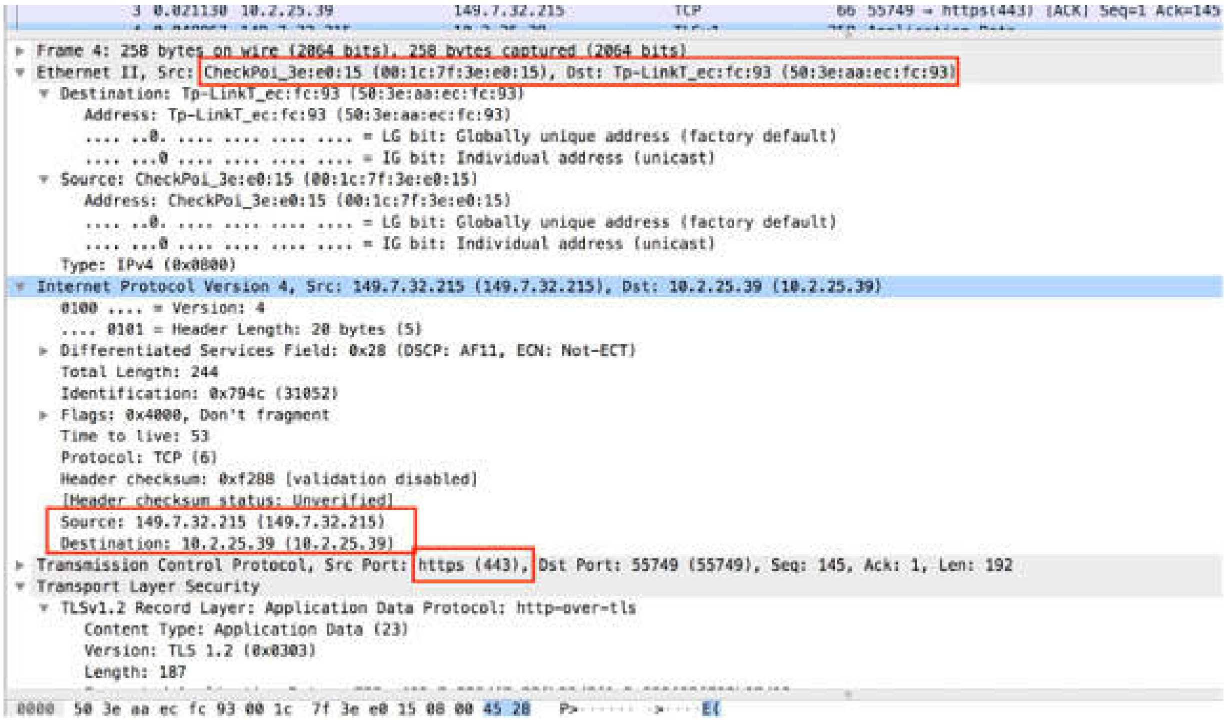
Clear the “Capture packets in pcapng format” check box and start a new capture. Now the file format of the capture is .pcap, as shown in the figure below.



Task 4:
To resolve names (IP and MAC), on the main menu, select Edit > Preferences. In the tree menu, click Name Resolution. A set of check boxes is displayed, as shown in the figure below.

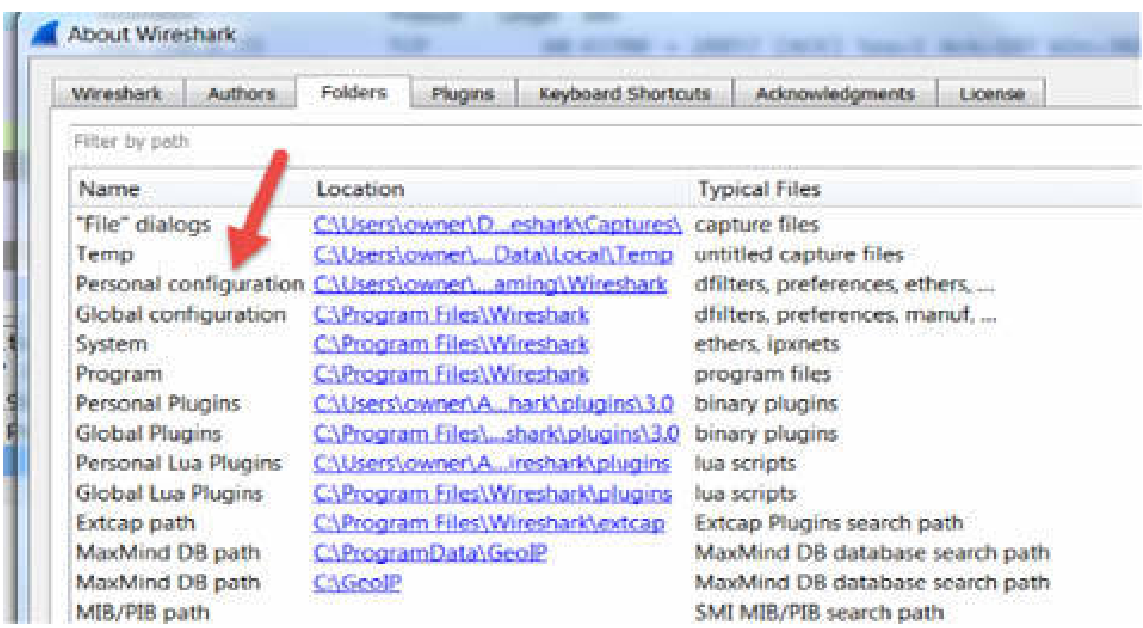


Select the first three check boxes—Resolve MAC addresses, Resolve transport names, and Resolve network IP addresses—to enable the complete name resolution. Click OK, and in the Packet Details pane, verify that the changes are displayed, as shown in the figure below.



Task 5:

On the main menu, select Help > About Wireshark > Folders. Find the personal configuration folder and double-click the related location. The folder is opened.



Locate the preference file and back it up. Close Wireshark and remove the preference file.

Restart Wireshark. The default capture preferences are restored.

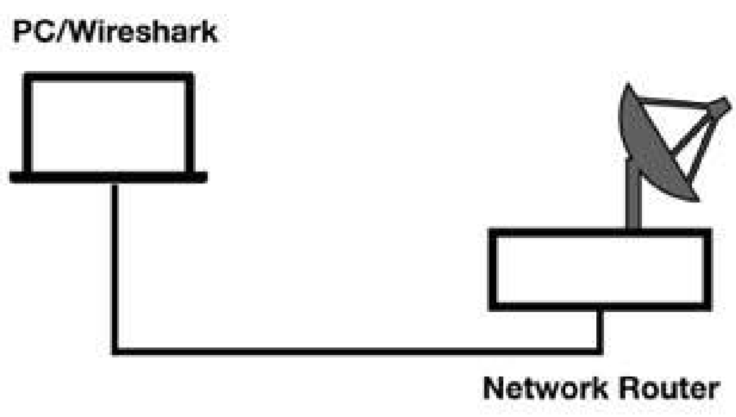
Lab 12. Name Resolution Preference

Lab Objective:
Learn how to create and apply user interface settings to the Wireshark GUI and how to restore it to the default settings.

Lab Purpose:
Wireshark is a network packet analyzer tool that allows you to customize the names resolution settings and save them.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

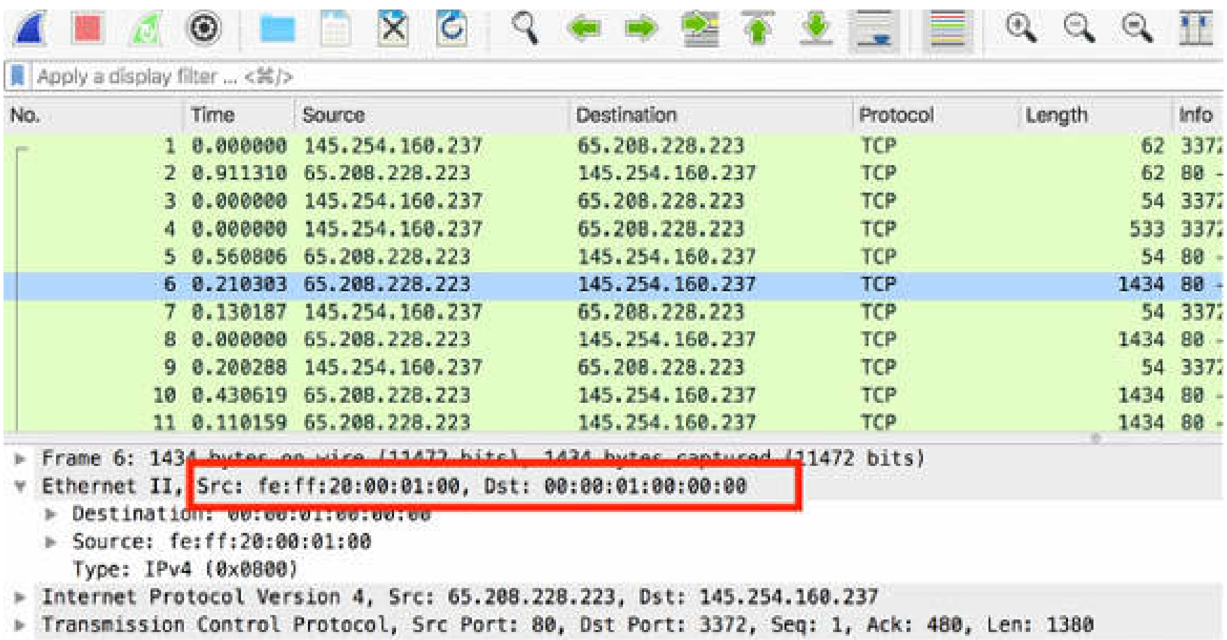
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



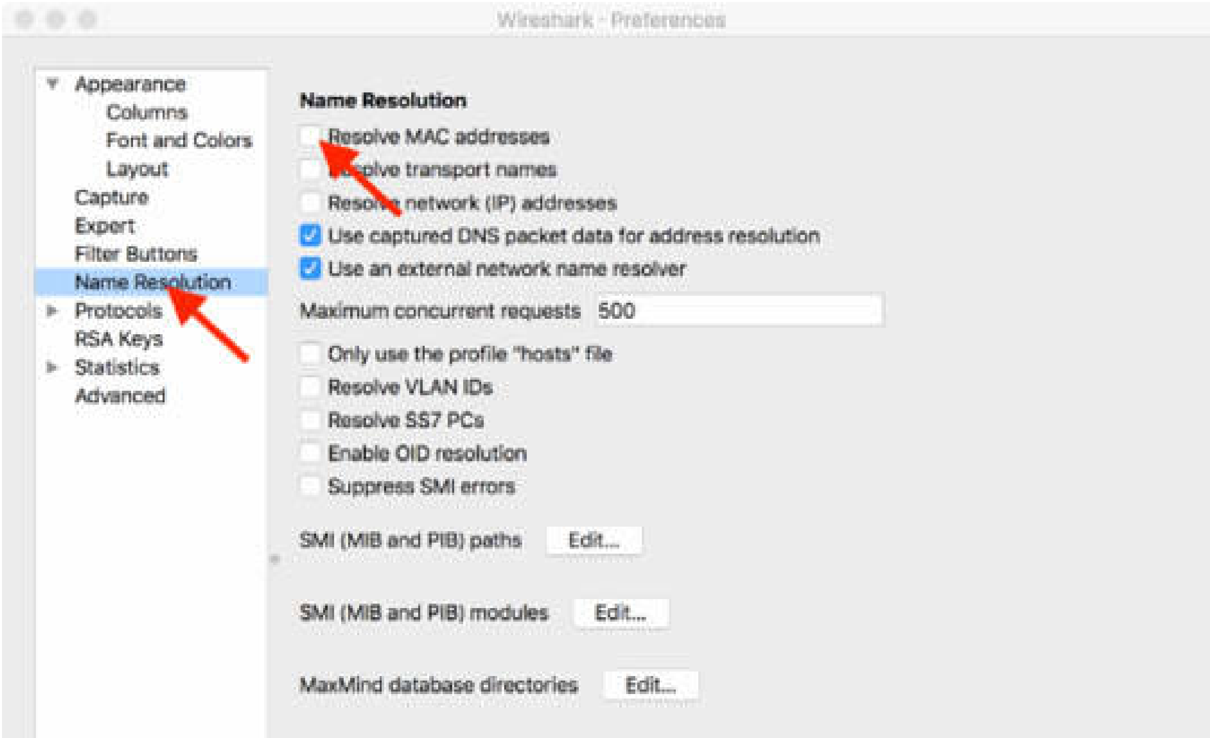
Lab Walkthrough:

Task 1:
Download the free sample capture file http.cap to your PC from **https://wiki.wireshark.org/SampleCaptures#TCP** , and then open the downloaded file in Wireshark.

In the Packet Details pane, you can observe the physical MAC address for each packet (source and destination), as shown in the figure below.

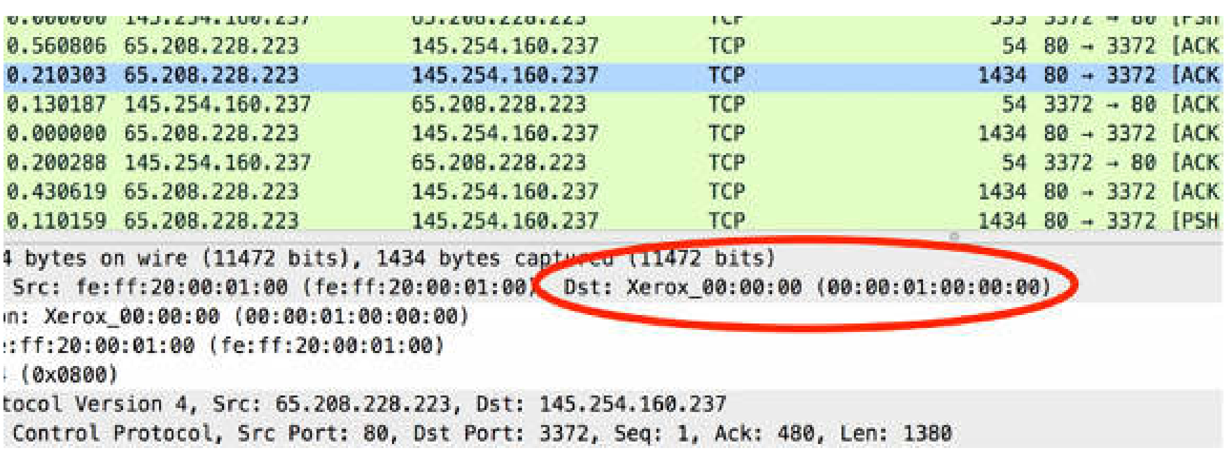


Task 2:
On the main menu, select Edit > Preferences. In the left tree view, click Name Resolution and select the “Resolve MAC Address” check box, as shown in the figure below. Finally, click OK.



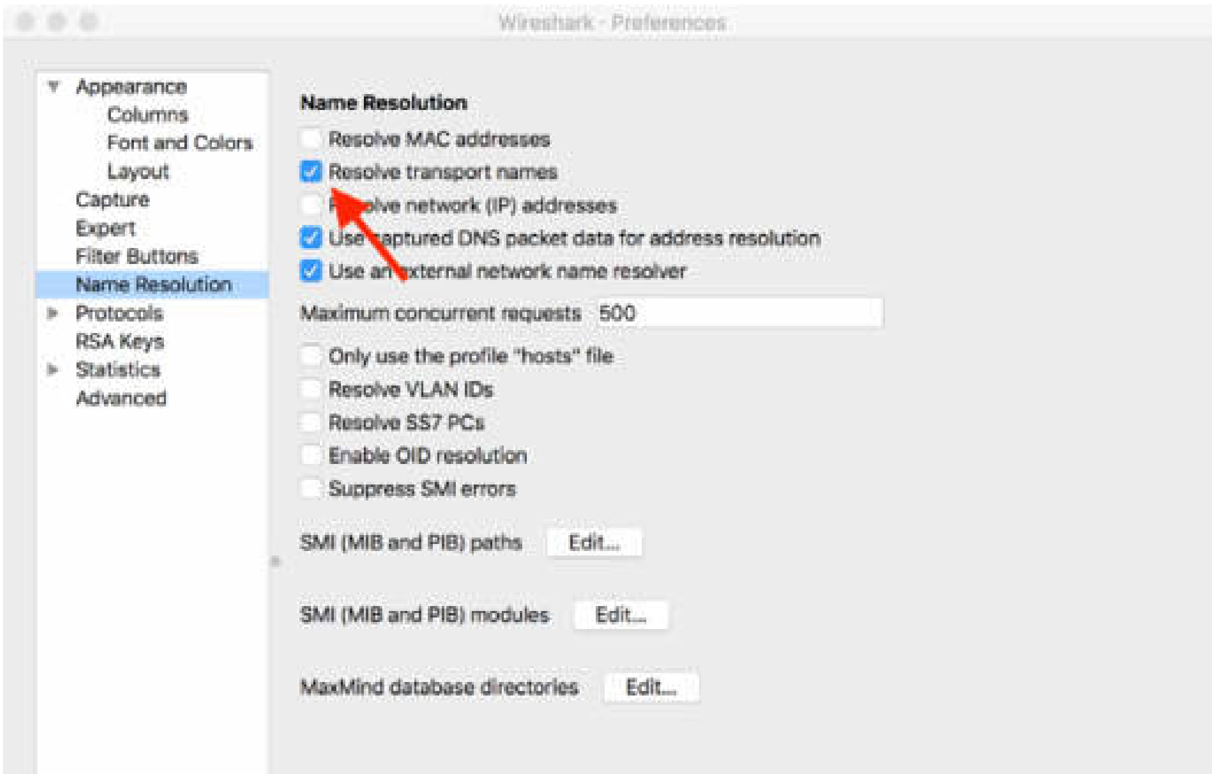
Task 3:

In the Packet Details pane, the physical MAC address is translated into names when possible. In the figure below, the destination MAC address is translated.

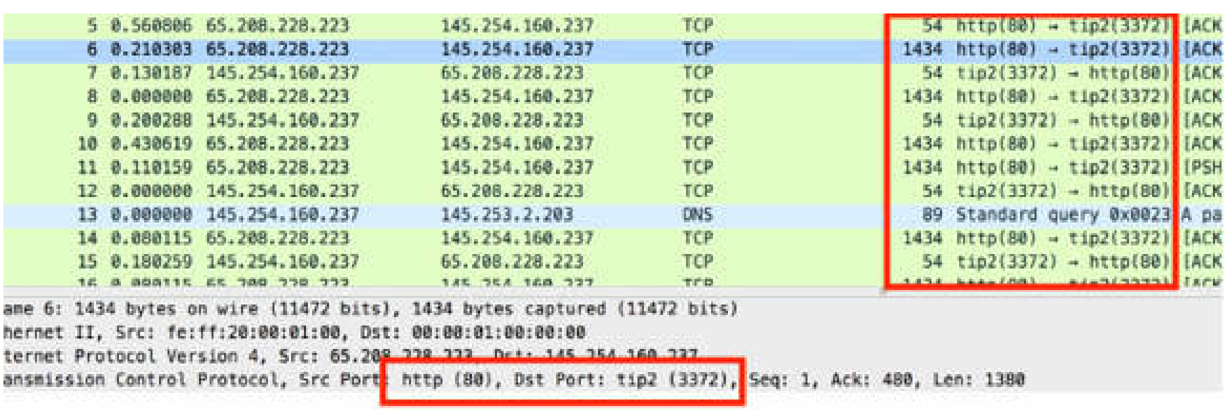


Task 4:

On the main menu, select Edit > Preferences. In the left tree view, click Name Resolution and select the "Resolve transport names" check box, as shown in the figure below. Finally, click OK.

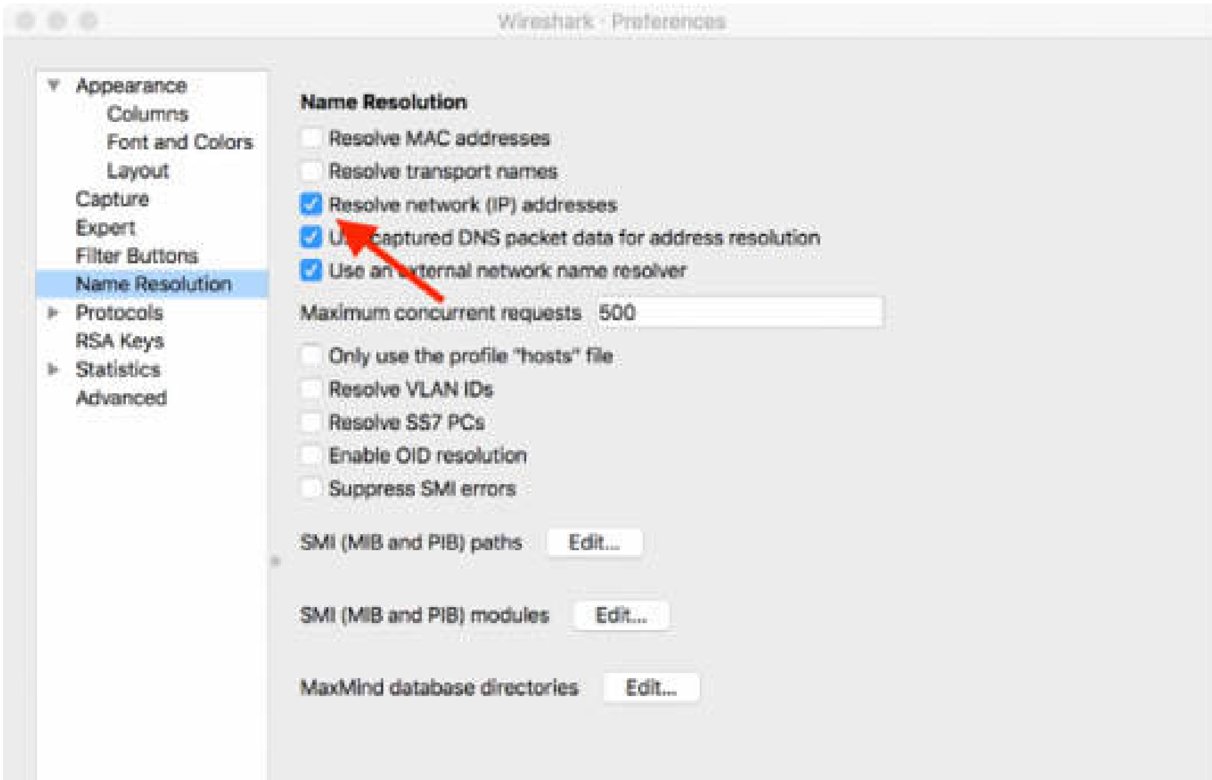


Task 5:
In the Packet List and the Packet Details panes, the port name resolution is activated, as shown in the figure below.



Task 6:

On the main menu, select Edit > Preferences. In the left tree view, click Name Resolution and select the "Resolve network (IP) addresses" check box, as shown in the figure below. Finally, click OK. This enables IP resolution.



Task 7:

In the Packet List pane, the IP name resolution is activated, as shown in the figure below.

	Time	Source	Destination
1	0.000000	dialin-145-254-160-237.pools.arcor-ip.net	65.208.228.223
2	0.911310	65.208.228.223	dialin-145-254-160-237.pools.arcor-ip.net
3	0.000000	dialin-145-254-160-237.pools.arcor-ip.net	65.208.228.223
4	0.000000	dialin-145-254-160-237.pools.arcor-ip.net	65.208.228.223
5	0.560806	65.208.228.223	dialin-145-254-160-237.pools.arcor-ip.net
6	0.210303	65.208.228.223	dialin-145-254-160-237.pools.arcor-ip.net
7	0.130187	dialin-145-254-160-237.pools.arcor-ip.net	65.208.228.223
8	0.000000	65.208.228.223	dialin-145-254-160-237.pools.arcor-ip.net
9	0.200288	dialin-145-254-160-237.pools.arcor-ip.net	65.208.228.223
10	0.430619	65.208.228.223	dialin-145-254-160-237.pools.arcor-ip.net
11	0.110159	65.208.228.223	dialin-145-254-160-237.pools.arcor-ip.net
12	0.000000	dialin-145-254-160-237.pools.arcor-ip.net	65.208.228.223
13	0.000000	dialin-145-254-160-237.pools.arcor-ip.net	145.253.2.203
14	0.080115	65.208.228.223	dialin-145-254-160-237.pools.arcor-ip.net
15	0.180259	dialin-145-254-160-237.pools.arcor-ip.net	65.208.228.223
16	0.080115	65.208.228.223	dialin-145-254-160-237.pools.arcor-ip.net

```
6: 1434 bytes on wire (11472 bits), 1434 bytes captured (11472 bits)
et II, Src: fe:ff:20:00:01:00, Dst: 00:00:01:00:00:00
et Protocol Version 4, Src: 65.208.228.223 (65.208.228.223), Dst: 145.254.160.237 (145.254.160.237)
ission Control Protocol, Src Port: 80, Dst Port: 3372, Seq: 1, Ack: 480, Len: 1380
```

Notes:

Make sure to clear all check boxes after finishing this lab.

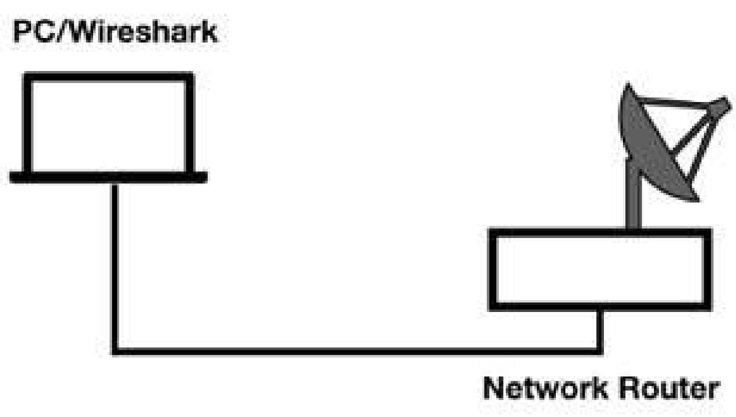
Lab 13. Colorize Traffic Preferences

Lab Objective:
Learn how to enable, disable, and modify colorizing rules on the Wireshark traffic.

Lab Purpose:
Wireshark allows packet colorization which is a useful mechanism to colorize packets according to a display filter.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

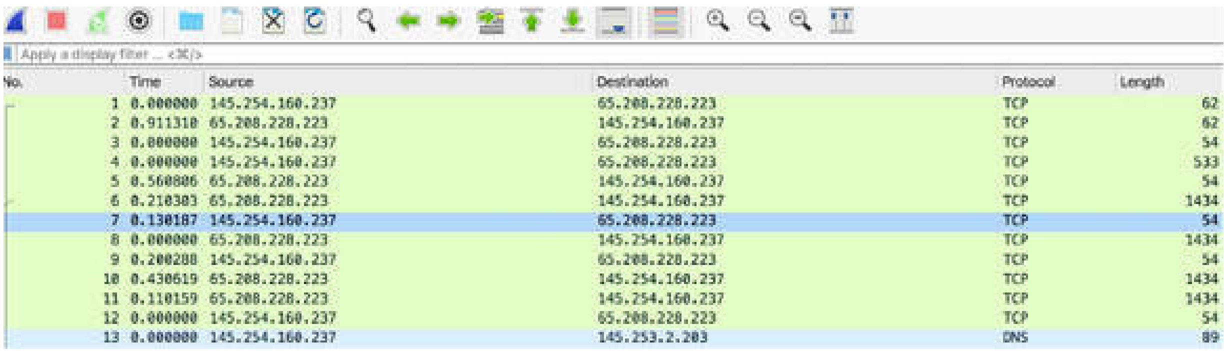
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



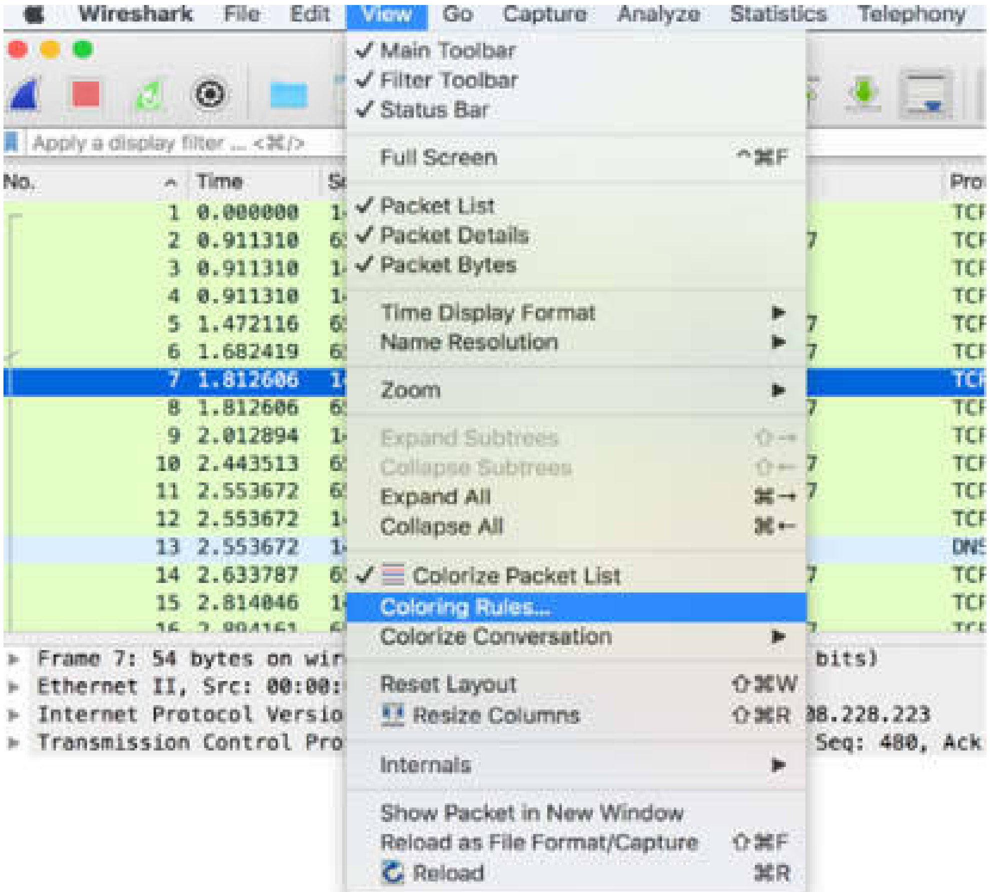
Lab Walkthrough:

Task 1:
Download the free sample capture file http.cap to your PC from **https://wiki.wireshark.org/SampleCaptures#TCP** , and then open the downloaded file in Wireshark.

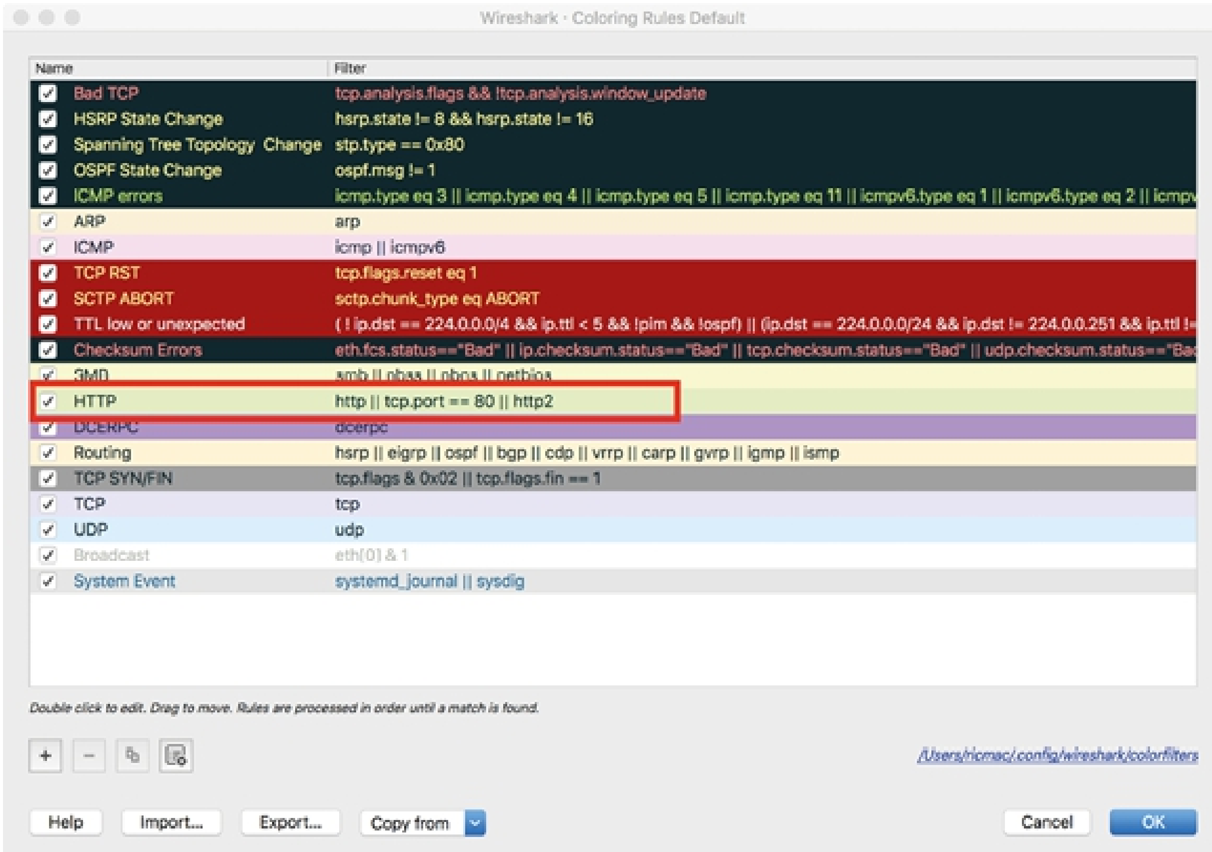
In the Packet List pane, each packet has a background color (light green in the figure below). You can download all lab images from the resources page on **www.101labs.net** .



Task 2:
On the main menu, select View > Coloring Rules, as shown in the figure below.

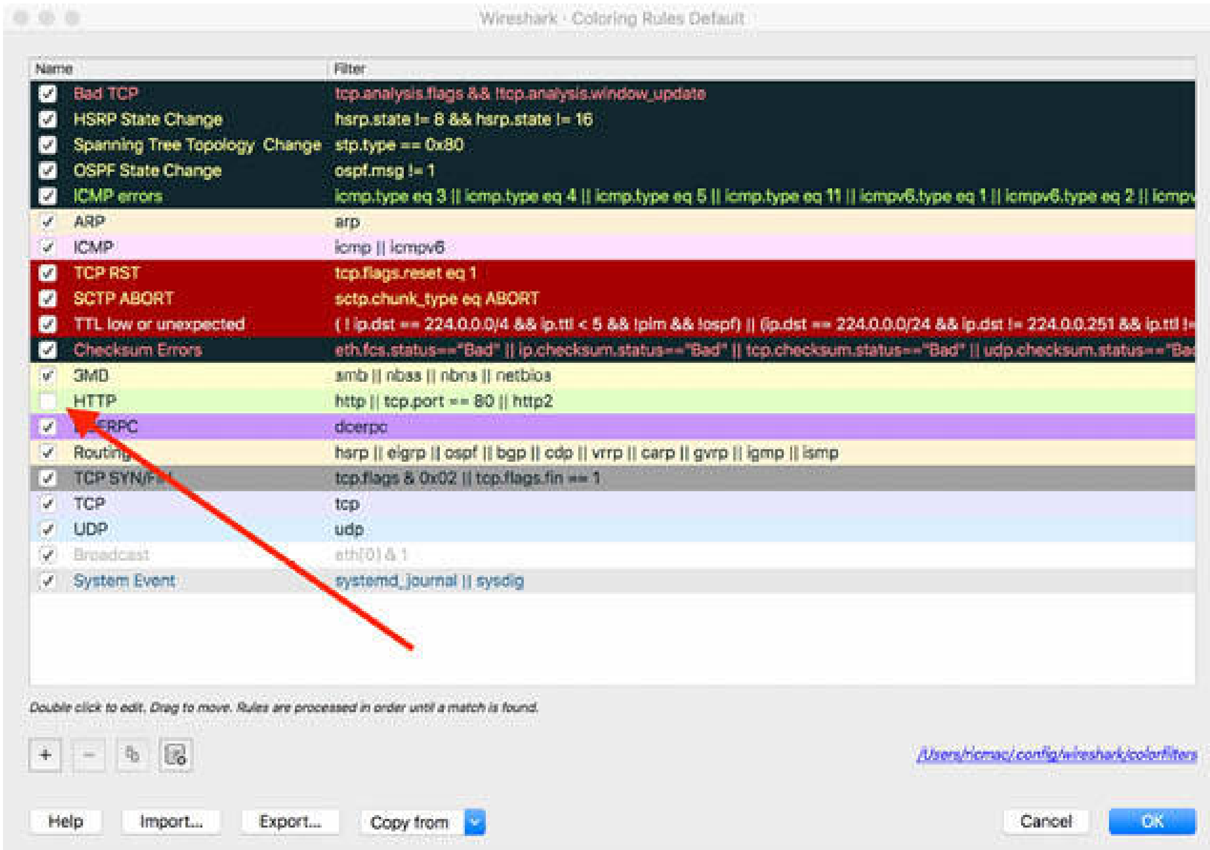


In the Coloring Rules Preference dialog box, you can check the color setting for the HTTP traffic, which is yellow/light green (as expected).

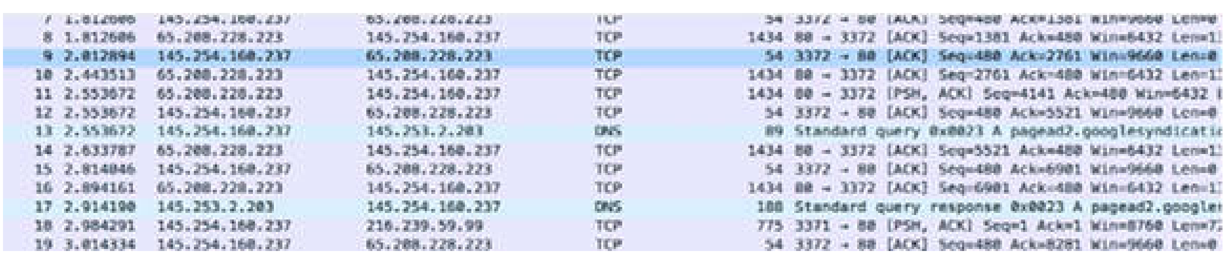


Task 3:

Clear the HTTP check box, and click OK.

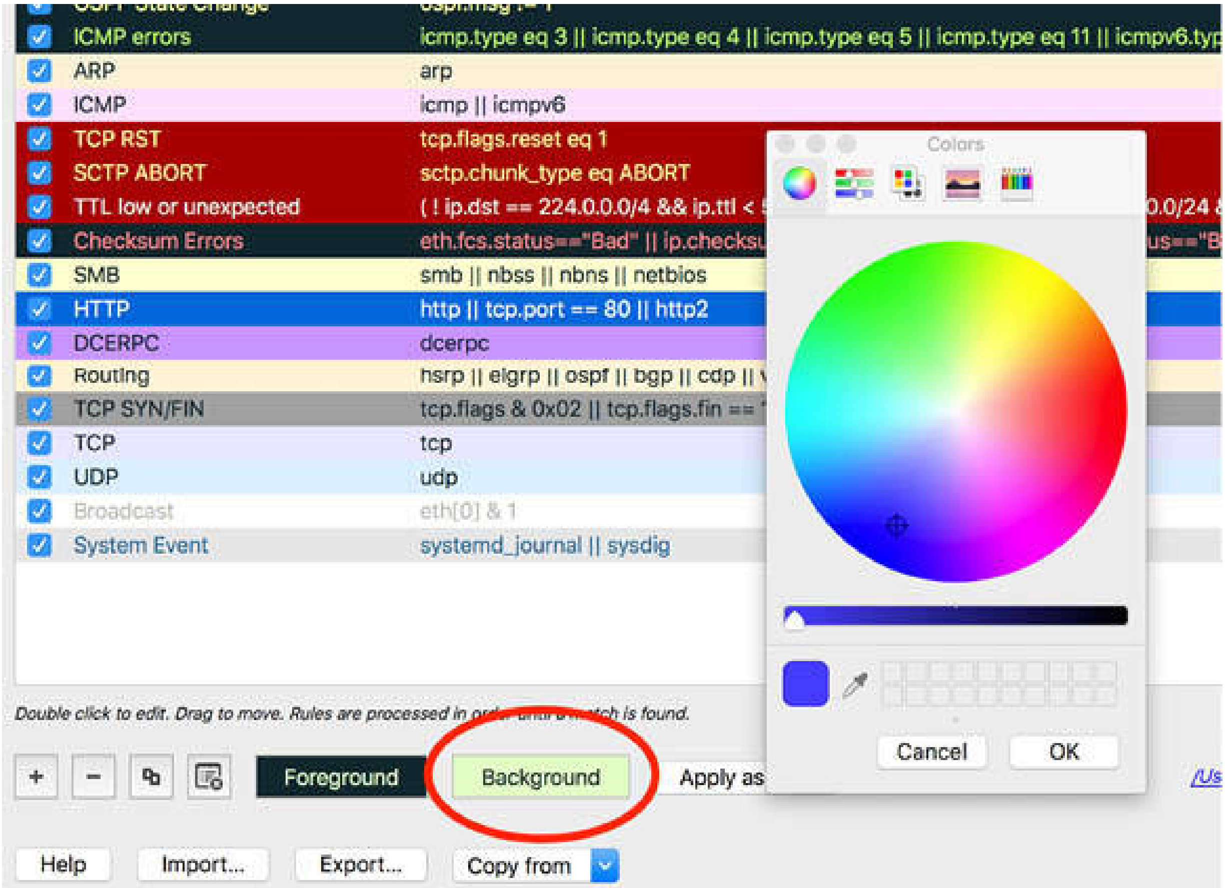


The HTTP coloring rule is disabled, and in the Packet List pane, there are no packets colored with light green color.

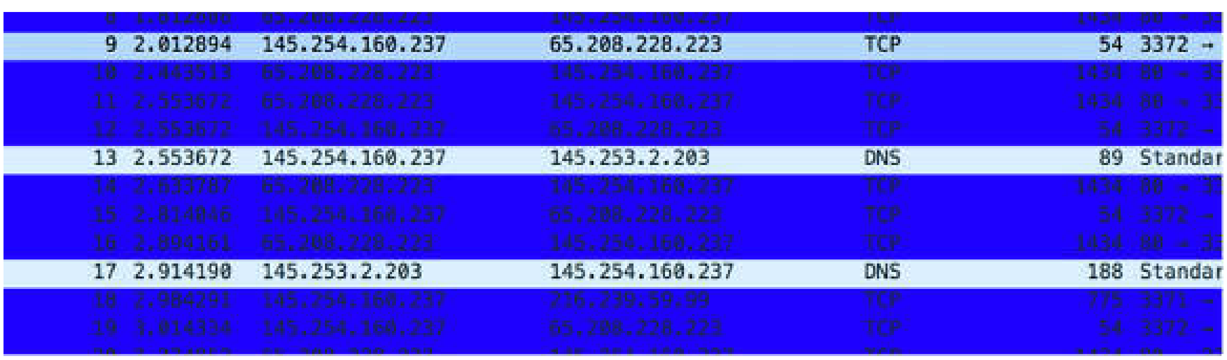


Task 4:

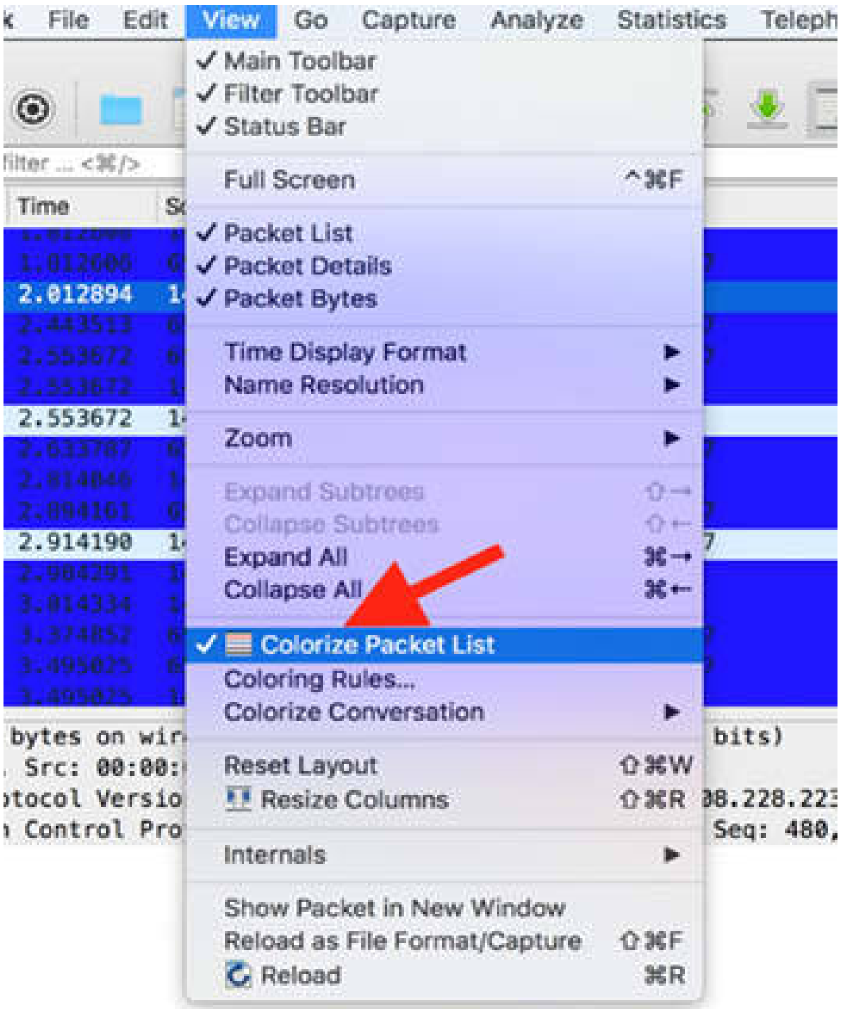
On the main menu, select View > Coloring Rules, and select the HTTP check box. At the bottom of the Preference dialog box, click the Background button. A Color Picker is displayed to change the color.



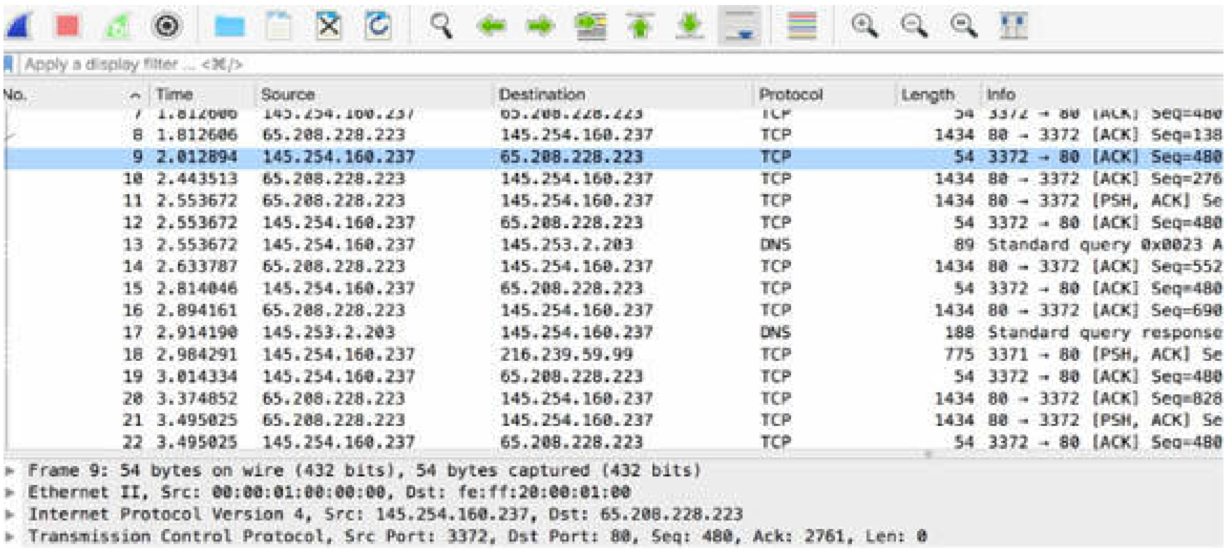
Select a different color, such as blue, and click OK. The background color of the HTTP packets is changed.



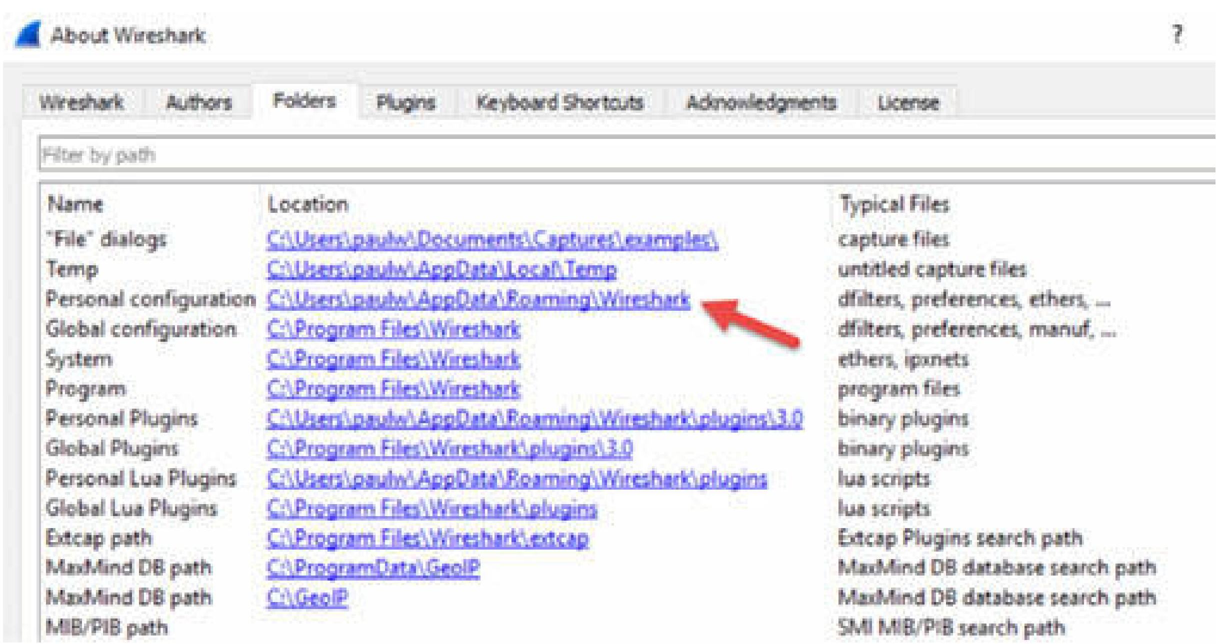
Task 5:
You can also disable all coloring rules with a single command. To do this, on the main menu, select View and clear the Colorize Packet List option.



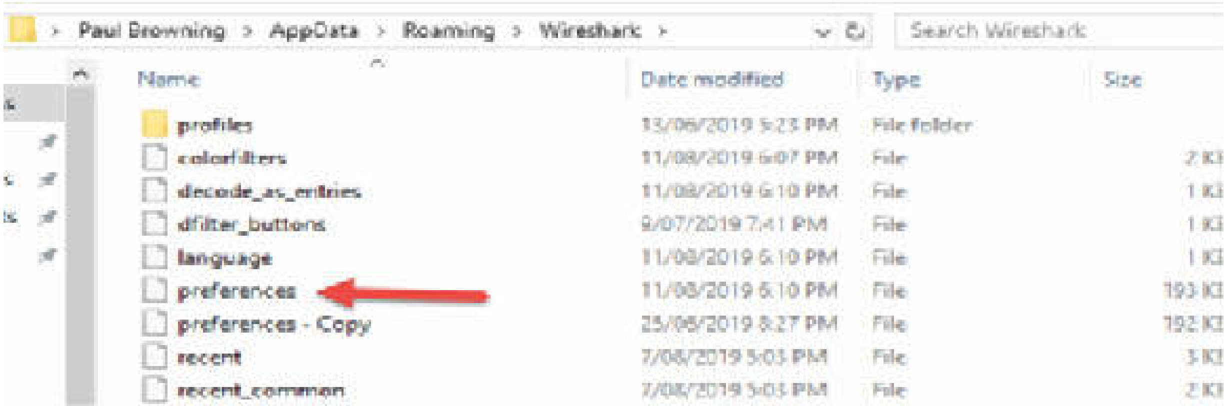
The background color is removed for all packets in the Packet List pane.



To reset Wireshark to its default settings, on the main menu, select Help > About Wireshark > Folders. Click your Personal configuration URL, as shown in the figure below. A file named preferences is available in your configuration folder. You can rename or delete the file.



Next time when you make any changes to the Wireshark settings, a new preferences file is created.



Notes:

To gain more confidence with traffic colorization, try different configurations and combinations of colors associated with protocols and filters. This feature allows you to emphasize the packets you might be interested in.

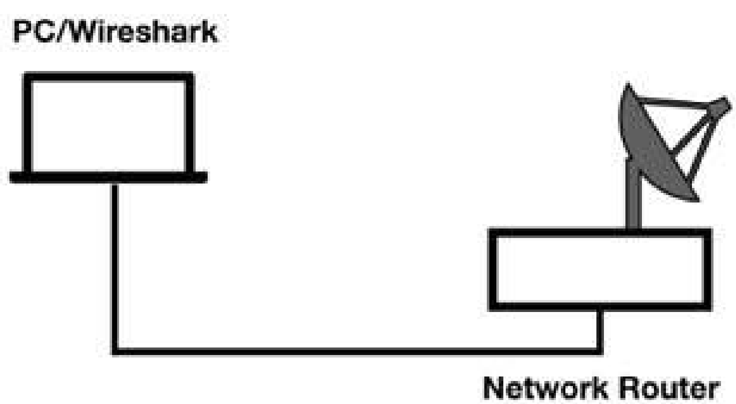
Lab 14. Temporary Colors

Lab Objective:
Learn how to apply temporary colors to a capture log.

Lab Purpose:
Wireshark allows temporary packet colorization, in particular to a conversation you are interested in.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

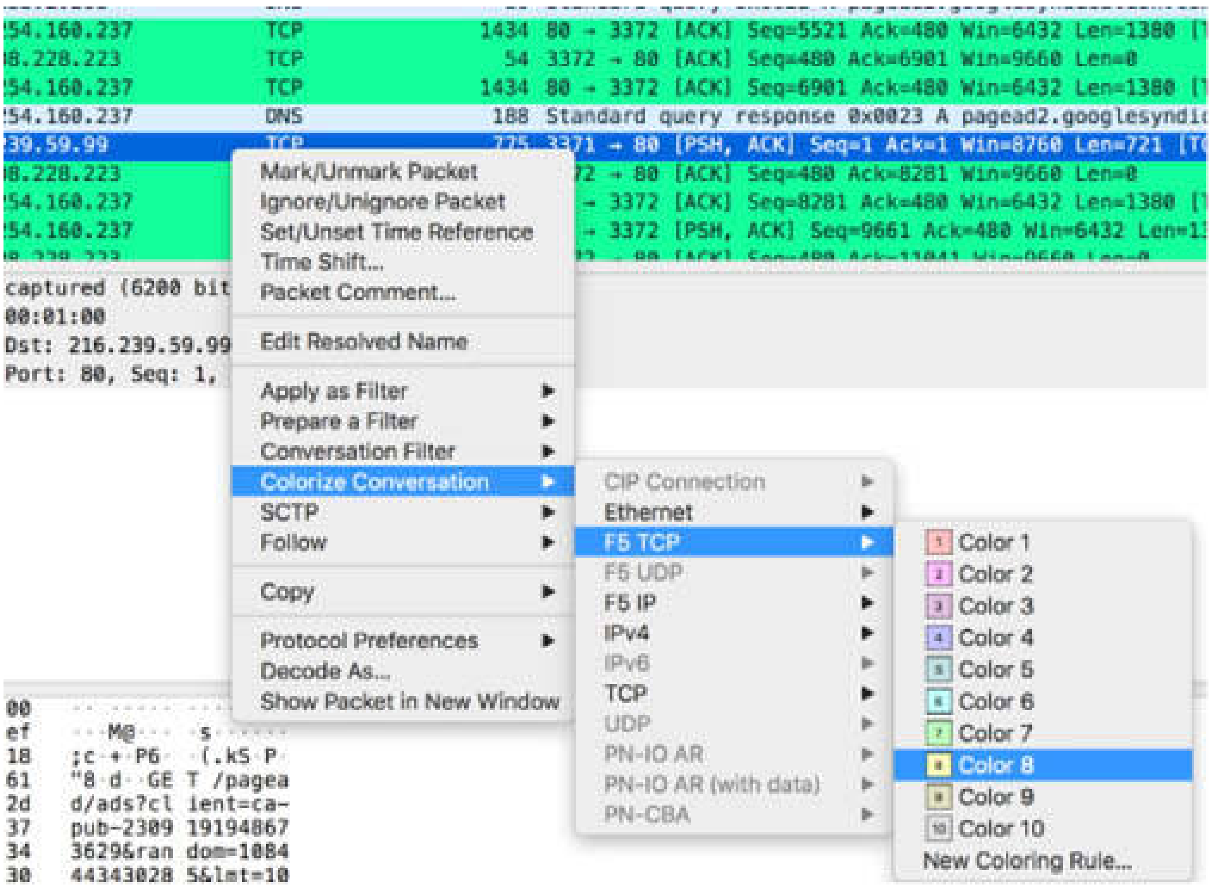


Lab Walkthrough:

Task 1:

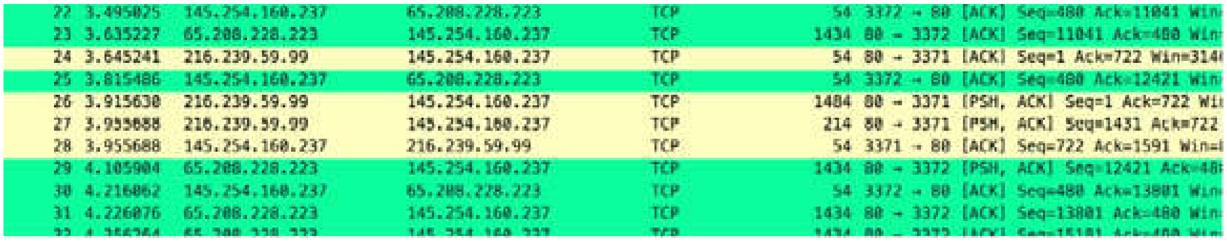
Download the free sample capture http.cap to your PC from https://wiki.wireshark.org/SampleCaptures#TCP , and then open the downloaded file in Wireshark.

In the Packet Details pane, right-click a TCP packet. A pop-up list is displayed.



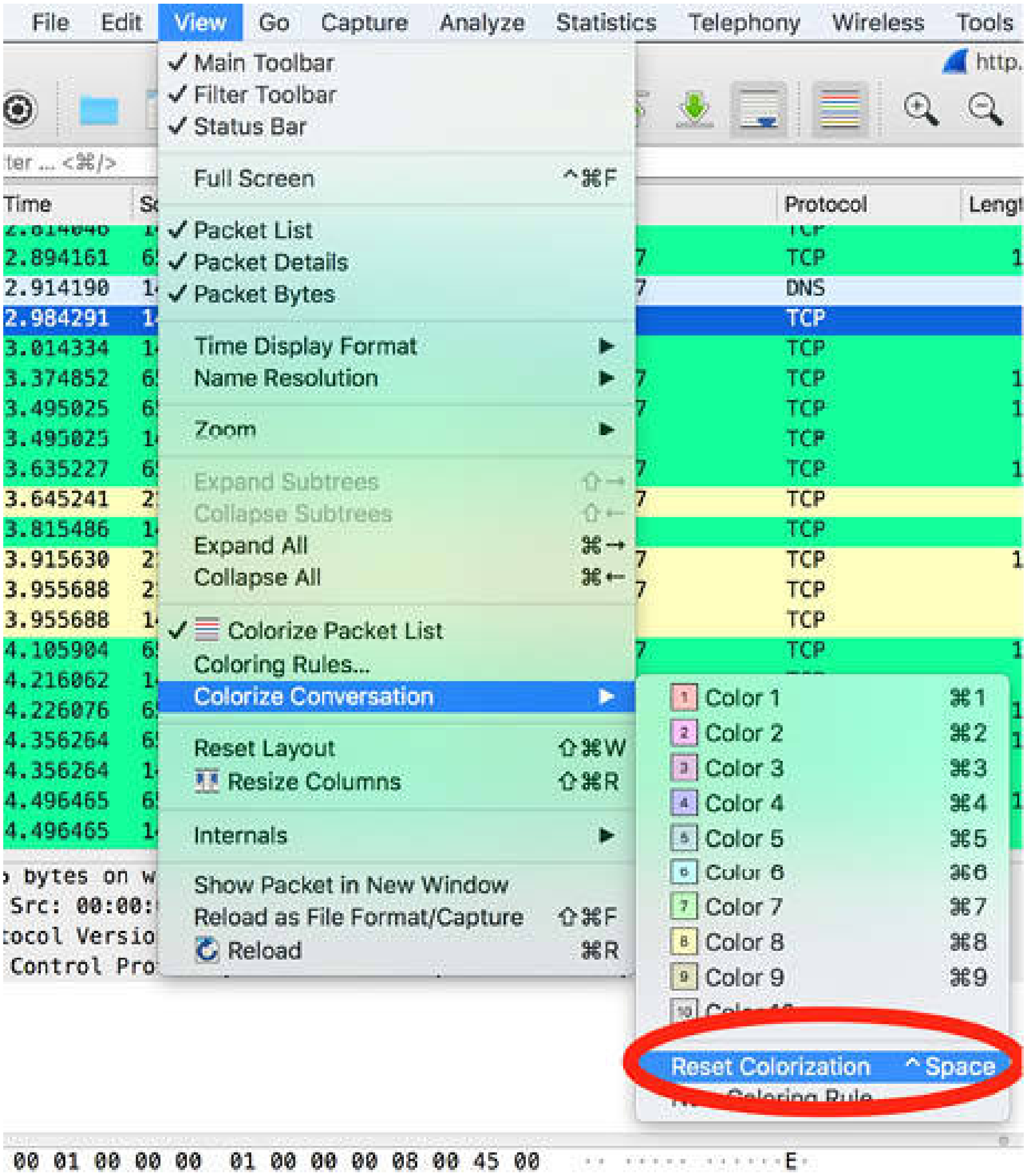
Task 2:
Select Colorize Conversation > TCP, and then choose a color.

The selected color is applied to the TCP conversation, as shown in the figure below. Packets #24, #26, #27, and #28 below acquired the newly selected color indicating that they are a part of the selected TCP conversation.



22	3.495025	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=11041 Win
23	3.635227	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=11041 Ack=480 Win
24	3.645241	216.239.59.99	145.254.160.237	TCP	54	80 → 3371 [ACK] Seq=1 Ack=722 Win=314
25	3.815486	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=12421 Win
26	3.915630	216.239.59.99	145.254.160.237	TCP	1484	80 → 3371 [PSH, ACK] Seq=1 Ack=722 Wi
27	3.955688	216.239.59.99	145.254.160.237	TCP	214	80 → 3371 [PSH, ACK] Seq=1431 Ack=722
28	3.955688	145.254.160.237	216.239.59.99	TCP	54	3371 → 80 [ACK] Seq=722 Ack=1591 Win=
29	4.105904	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [PSH, ACK] Seq=12421 Ack=48
30	4.216062	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=13801 Win
31	4.226076	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=13801 Ack=480 Win

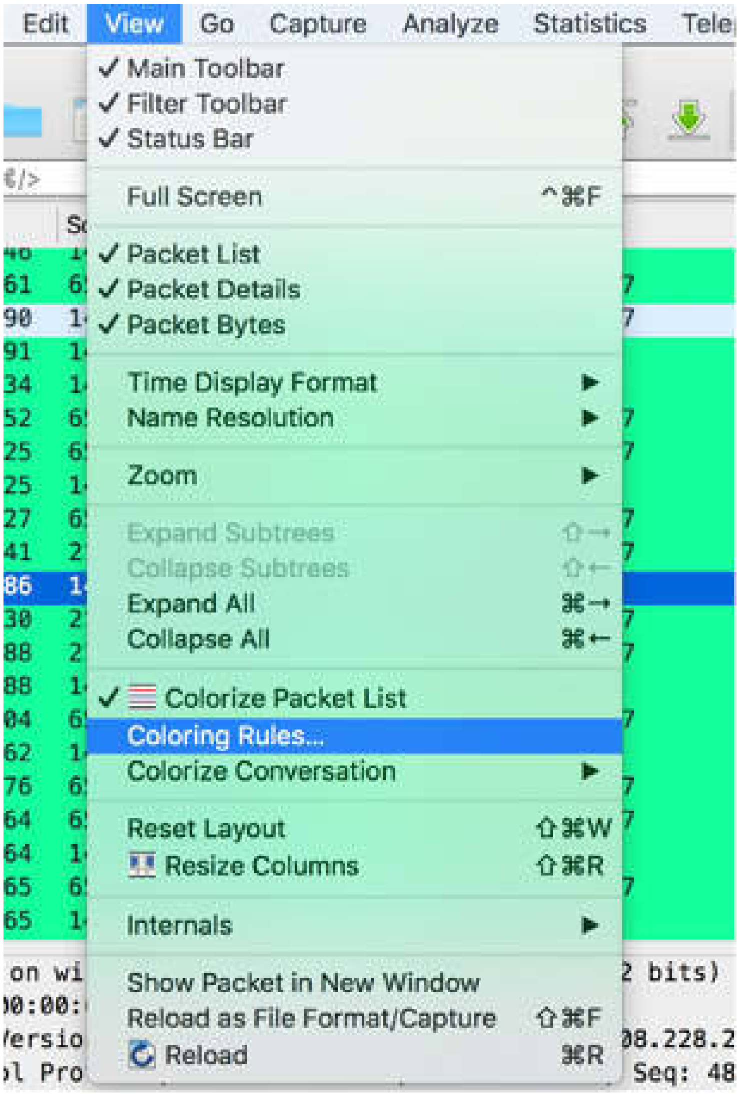
Task 3:
To restore the original packet colorization, on the main menu, select View > Colorize Conversation > Reset Colorization, as shown in the figure below.



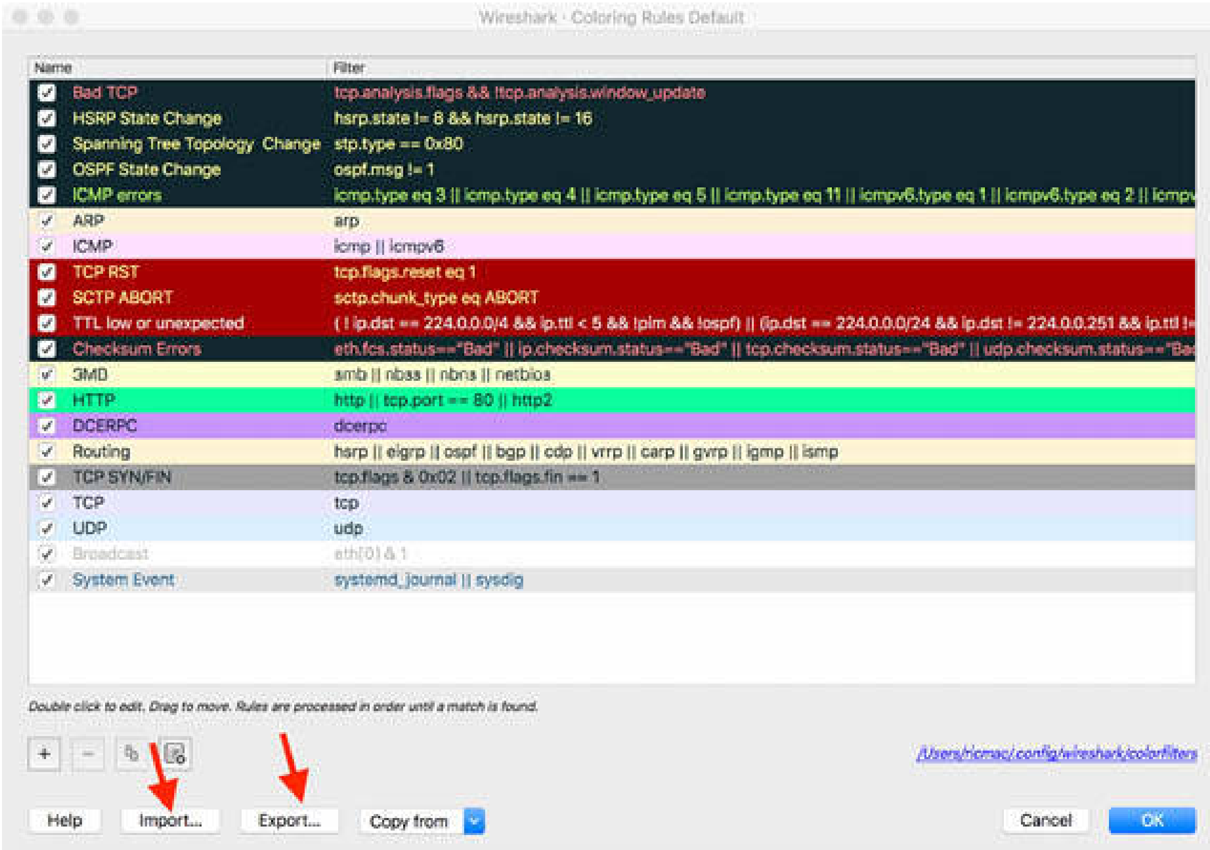
The original colorization is restored in the Packet List pane.

Task 4:

To share the saved packets colorization rules, on the main menu, select View > Coloring rules, as shown in the figure below.



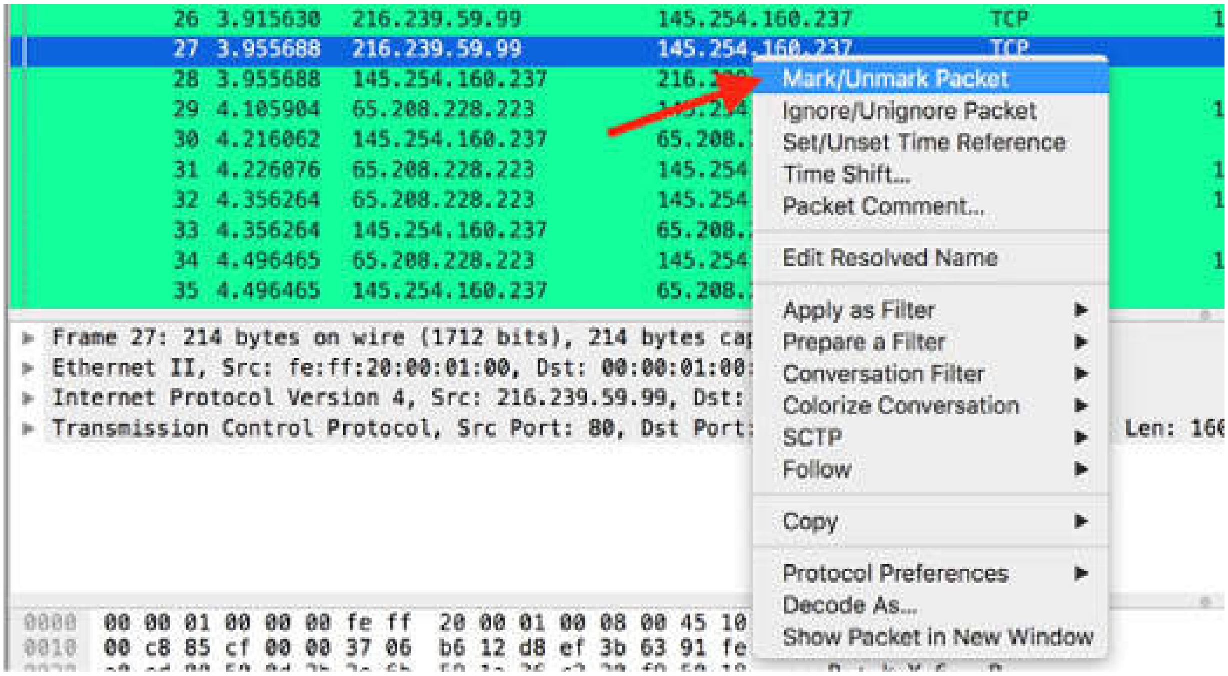
The Coloring Rules dialog box is displayed. To save your packets colorizations preferences, click Export. To apply a colorization preference shared by someone else, click Import.



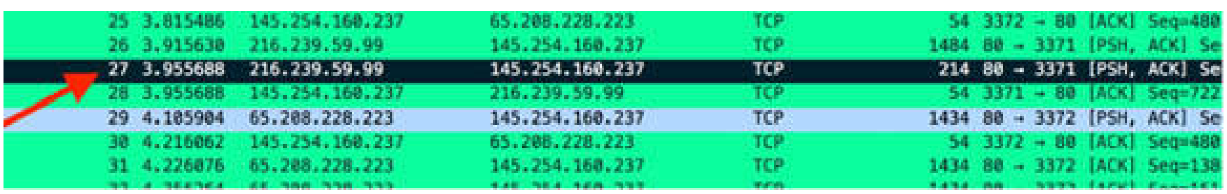
Task 5:

You can also temporarily mark one or more packets of interest. In the Packet Details pane, right-click a packet. A pop-up list is displayed.

Select "Mark/Unmark Packet", as shown in the figure below.



The background of the selected packet has black color, as shown in the figure below.



You can repeat this task to unmark the same packet or to mark more packets.

Notes:
To gain more confidence in using temporary packets colorization, select, unselect, and colorize different protocol conversations. The colorization feature allows you to highlight the packets you might be interested in.

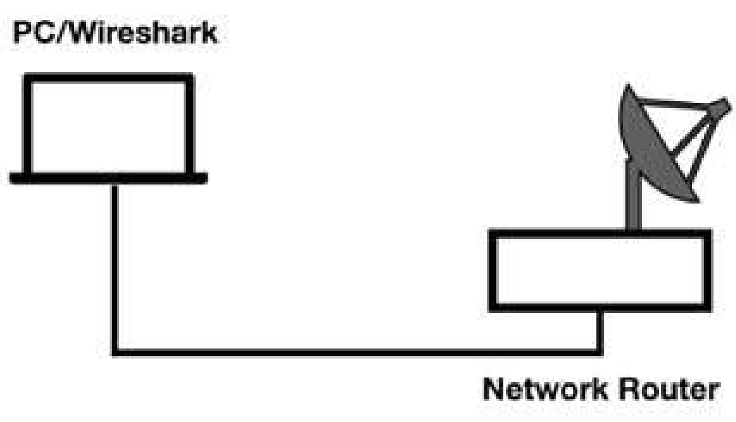
Lab 15. Packet Time Reference

Lab Objective:
Learn how to measure packet arrival time with a time Reference.

Lab Purpose:
Wireshark is a network packet analyzer tool that enables you to view the traffic in real time on a network or to analyze previously-saved traces. To measure the packet's arrival time, you can set a time reference (t0) on a selected packet and display all the time deltas for each following packet from the reference (t0).

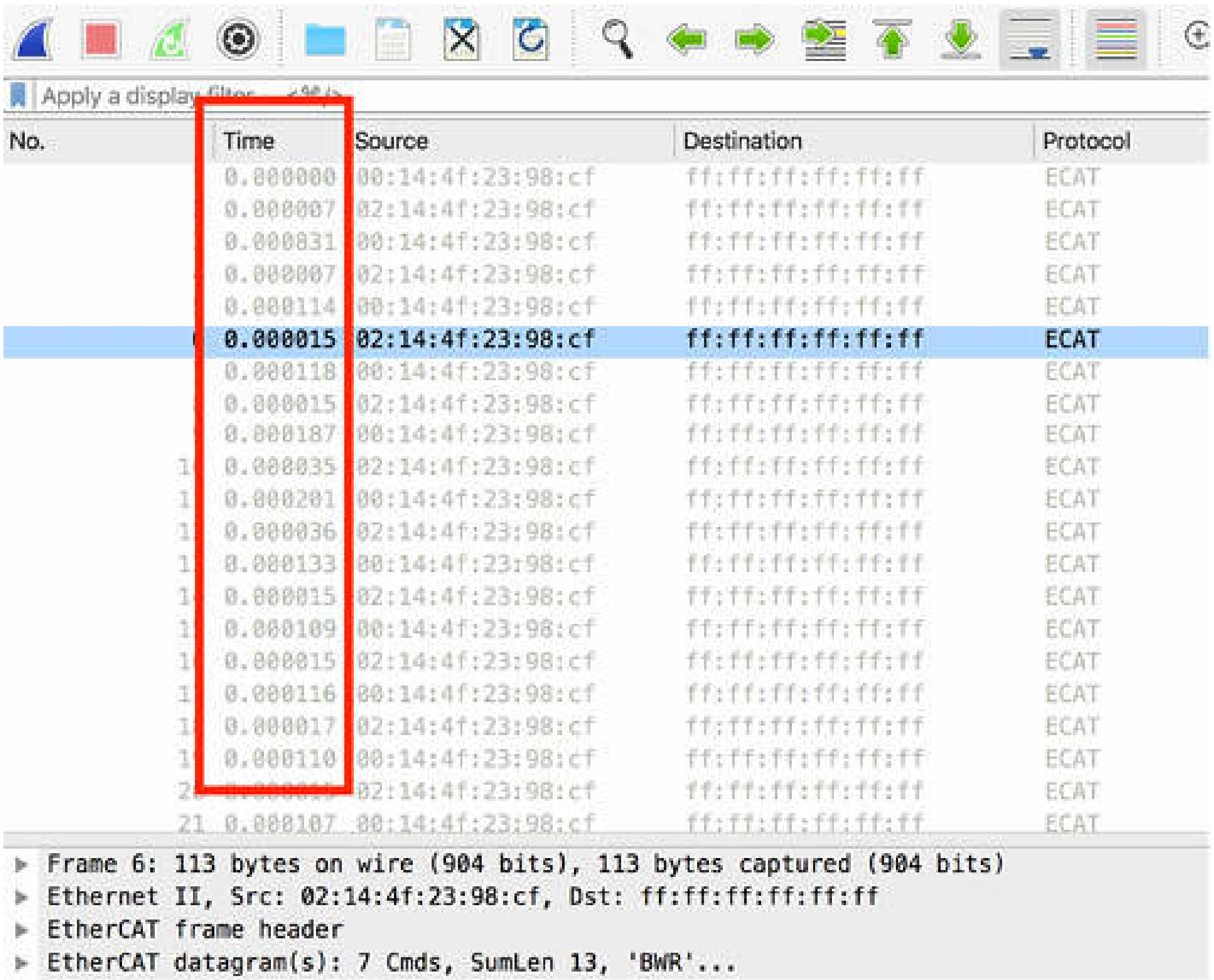
Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

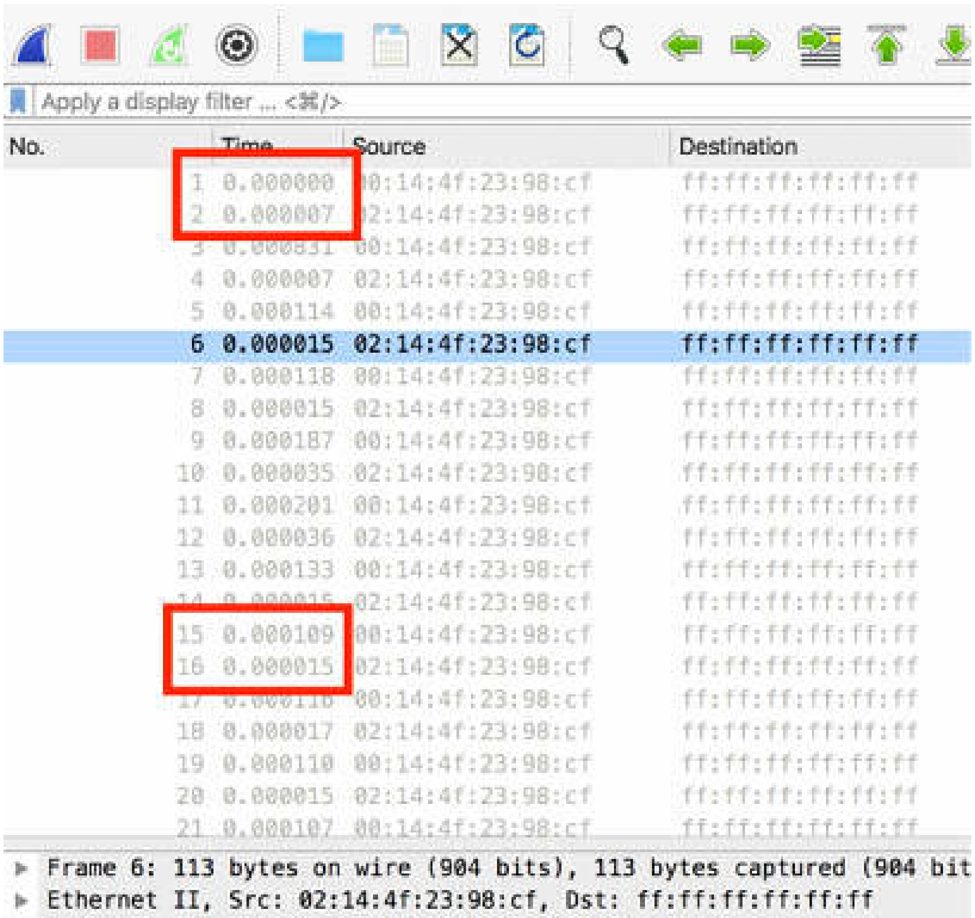


Lab Walkthrough:

Task 1:
Download the free sample capture file ethercat.cap to your PC from **https://wiki.wireshark.org/SampleCaptures** , and then open the file in Wireshark. In the main window, the default timestamp column is displayed, as shown in the figure below.

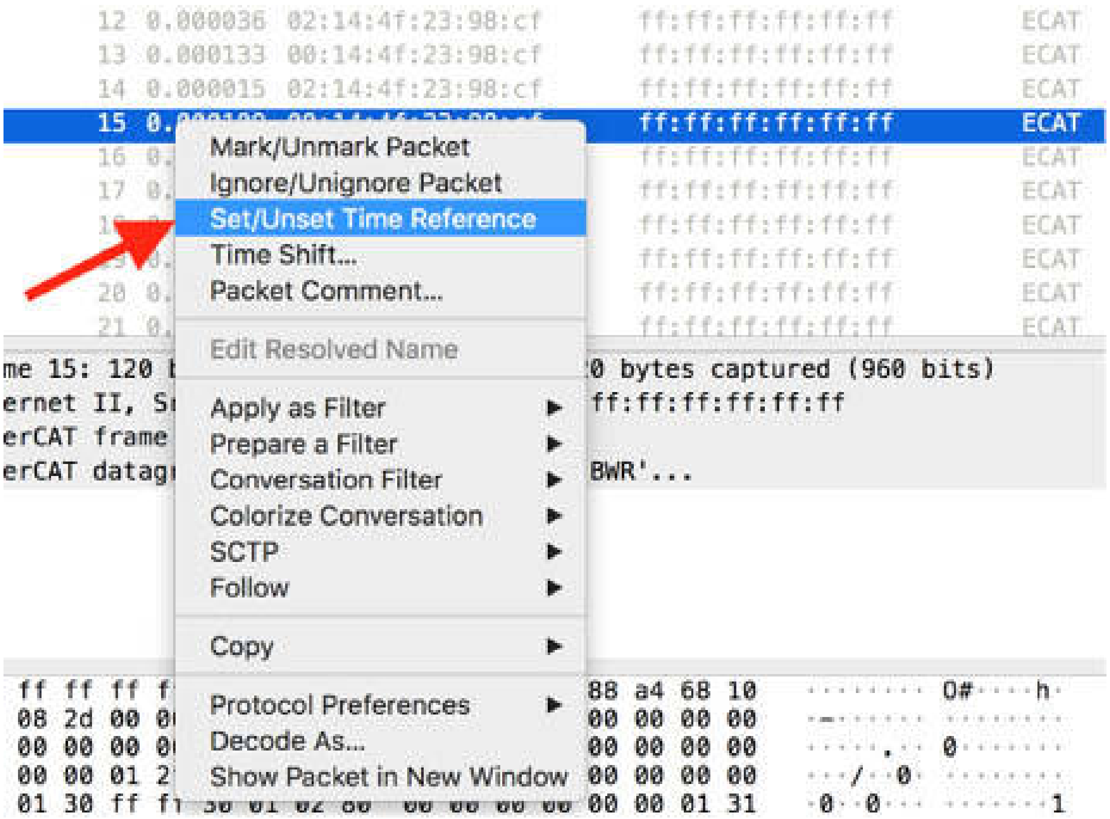


The default time display format shows the time drift in seconds (s) since the previously displayed packet. In the figure below, packet #2 arrived 0.000007s after packet #1 and packet #16 arrived 0.000015s after packet #15. Note down the time elapsed between packets #1–#2 (T_1) and #15–#16 (T_2).

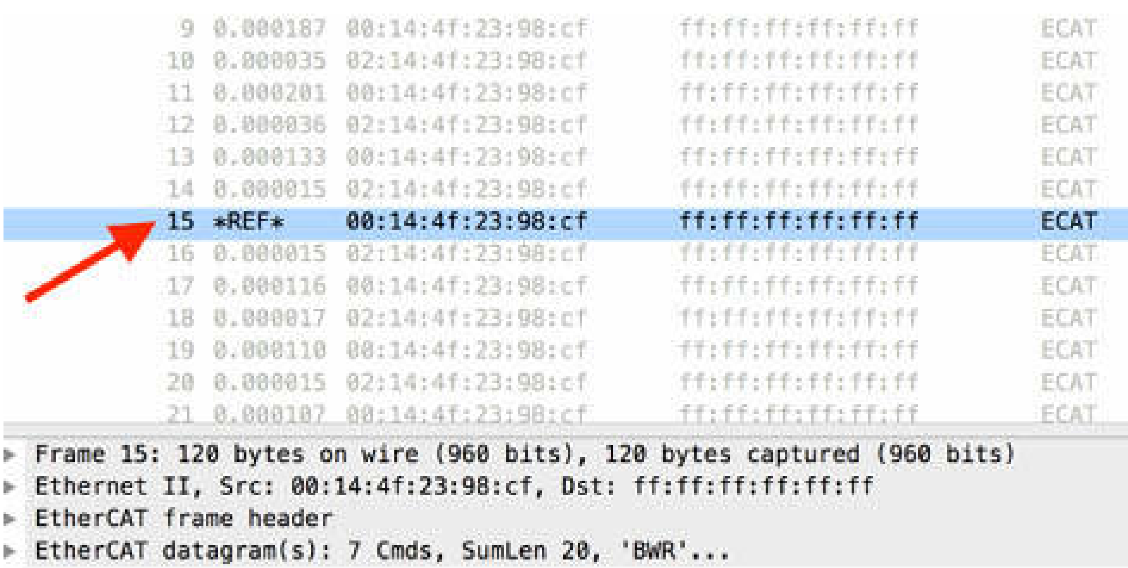


Task 2:

Right-click packet #15, and click "Set/Unset Time Reference", as shown in the figure below.

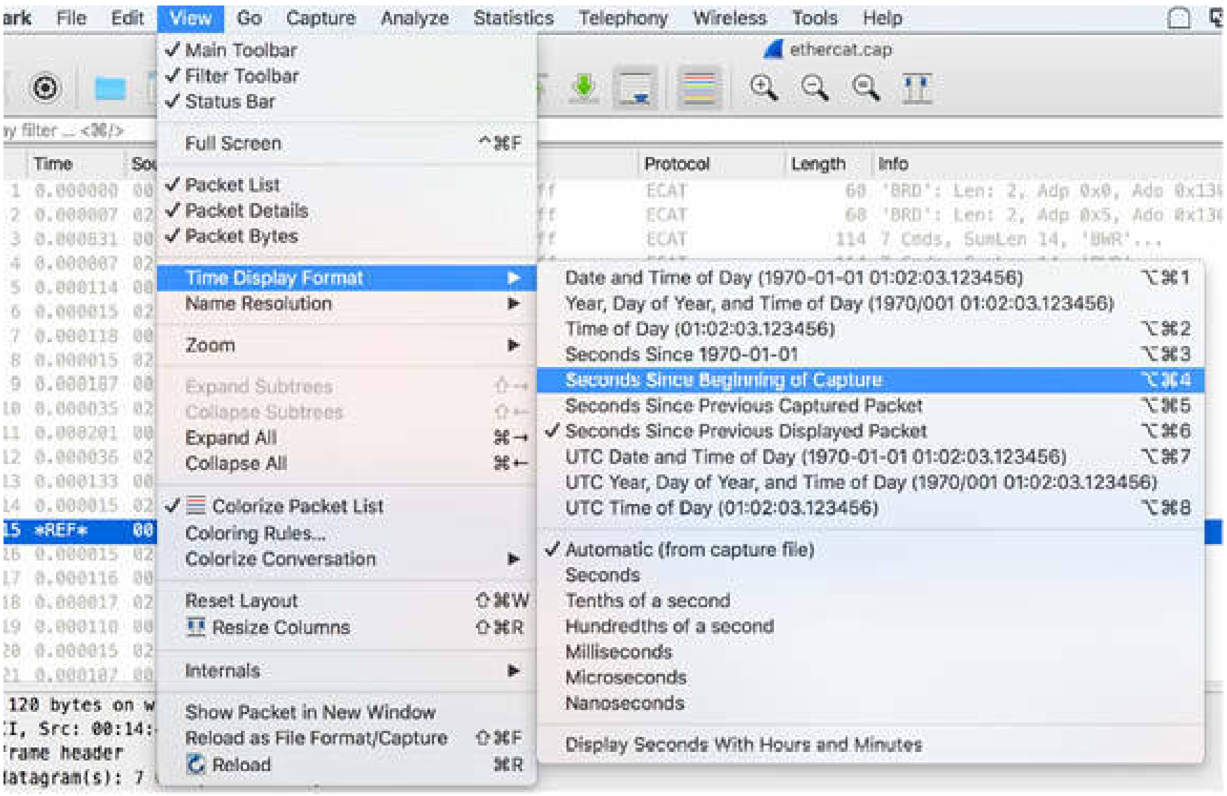


A Time Reference (t_0) is placed on packet #15.



Task 3:

On the main menu, select View > Time Display Format > Seconds Since Beginning of Capture.



For each packet following the reference (#15), the total elapsed time is displayed in seconds. Compare the figure shown in Task 1 and the figure below, and note the difference in the time displayed for each packet.

No.	Time	Source	Destination
1	0.000000	00:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
2	0.000007	02:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
3	0.000838	00:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
4	0.000845	02:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
5	0.000959	00:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
6	0.000974	02:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
7	0.001092	00:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
8	0.001107	02:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
9	0.001294	00:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
10	0.001329	02:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
11	0.001530	00:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
12	0.001566	02:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
13	0.001699	00:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
14	0.001714	02:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
15	***REF***	**00:14:4f:23:98:cf**	**ff:ff:ff:ff:ff:ff**
16	0.000015	02:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
17	0.000131	00:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
18	0.000148	02:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
19	0.000258	00:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff
20	0.000273	02:14:4f:23:98:cf	ff:ff:ff:ff:ff:ff

Task 4:
In the figure above, the following delta times are displayed:

- T_a = 0.000015s (time elapsed from the REF packet to the arrival of packet 16)
- T_b = 0.000131s (time elapsed from the REF packet to the arrival of packet 17)

You can verify that $T_a = T_1$ and $Tb = T_1 + T_2$.

Notes:
You can choose from a wide range of formats for time display. On the main menu, select View > Time Display Format and choose a time format that meets your requirements and is the most convenient, such as absolute time and day, absolute year, time, and date, relative time, UTC time and date, etc.

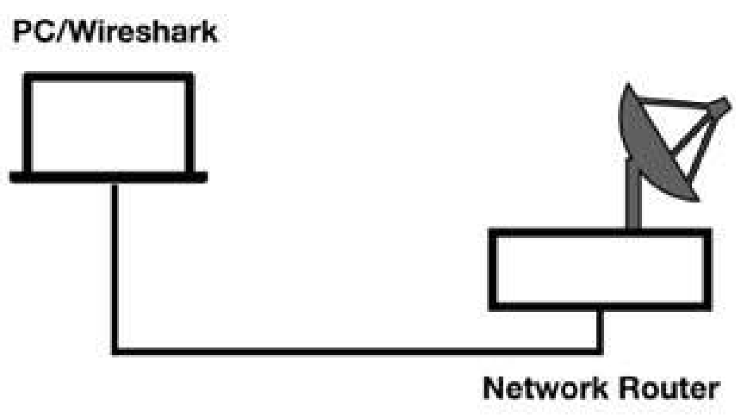
Lab 16. Additional Time Columns and Summary

Lab Objective:
Learn how to add time columns and view a summary of the file capture log.

Lab Purpose:
Wireshark allows you to add time columns to its default configuration and to view a summary related to traffic rates, packet sizes, and overall bytes transferred.

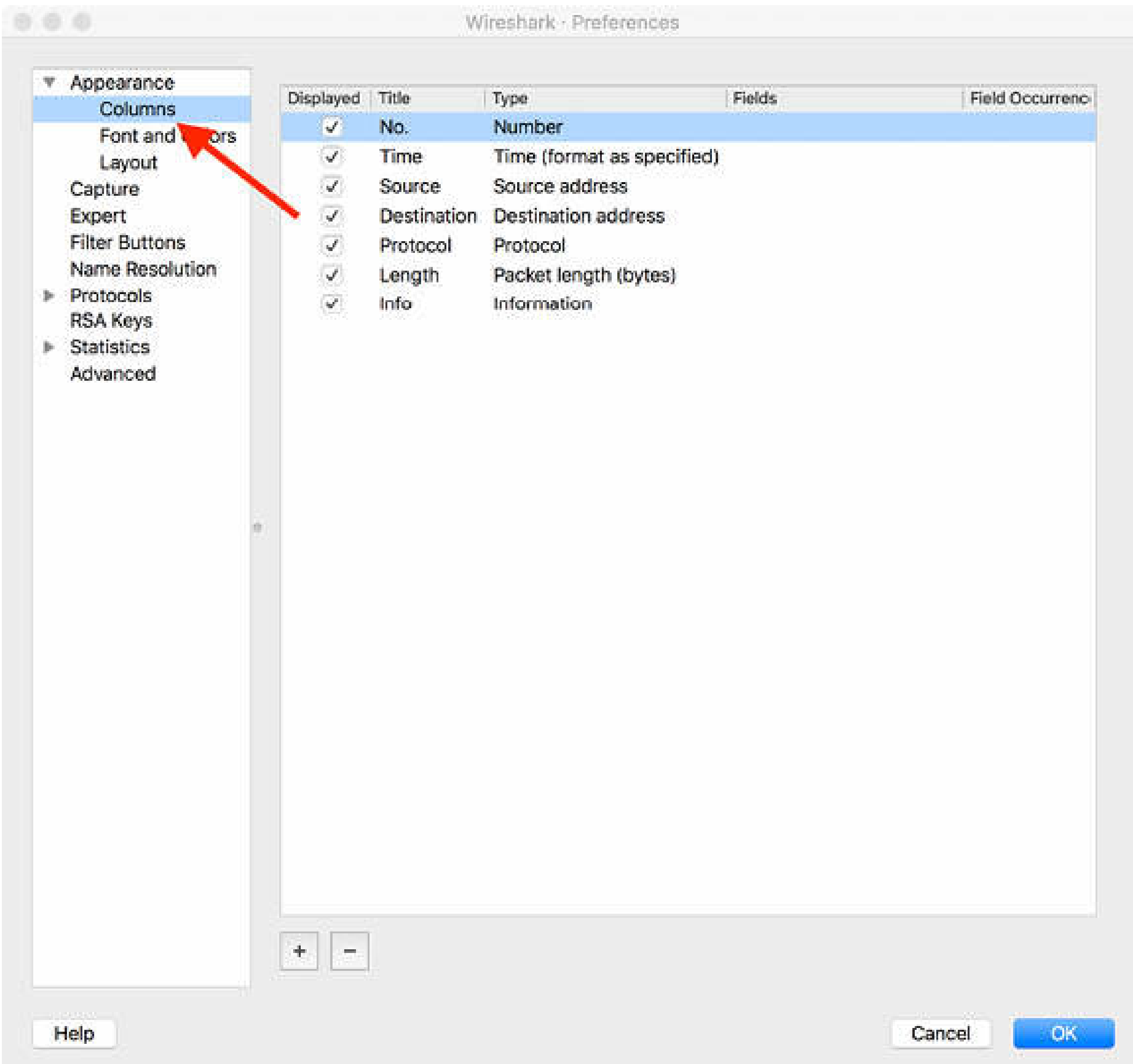
Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

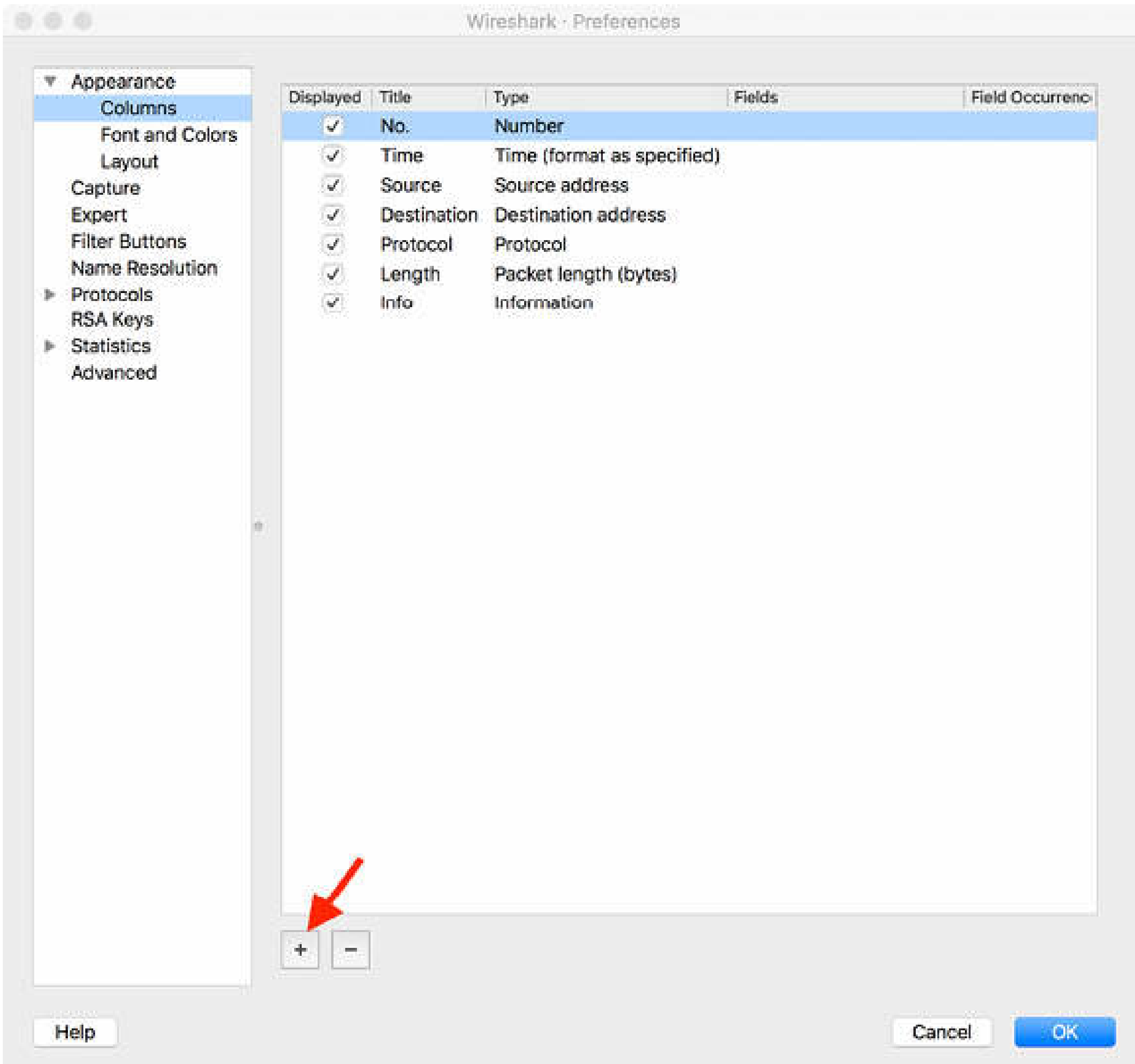


Lab Walkthrough:

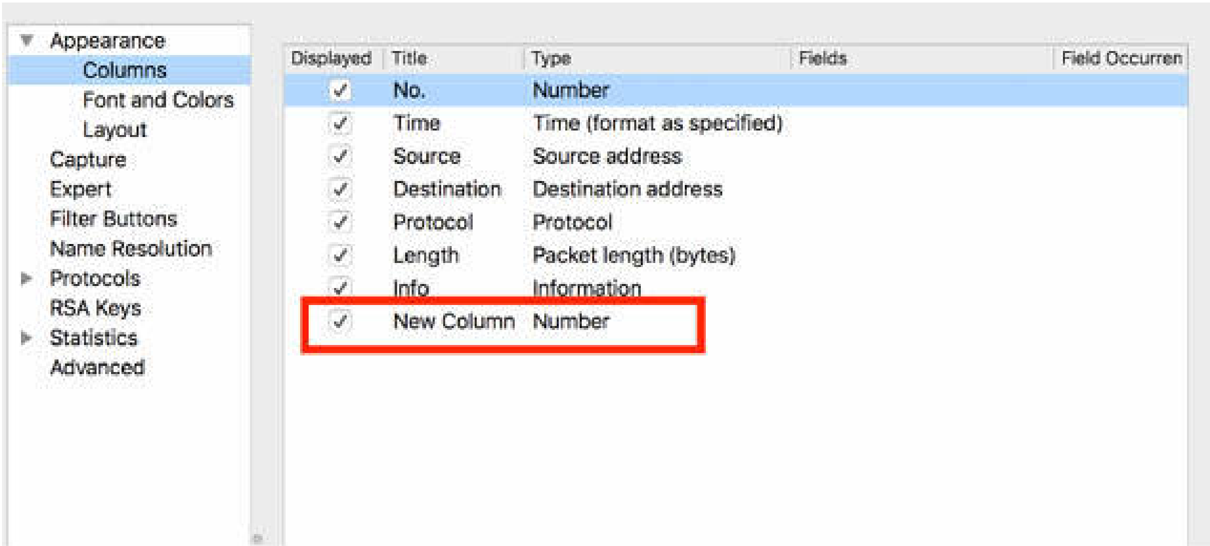
Task 1:
Download the free sample capture file http.cap to your PC from https://wiki.wireshark.org/SampleCaptures#TCP , and then open the file in Wireshark. On the main menu, select Edit > Preferences, and in the left tree view of the Preferences dialog box, select Appearance > Columns, as shown in the figure below.



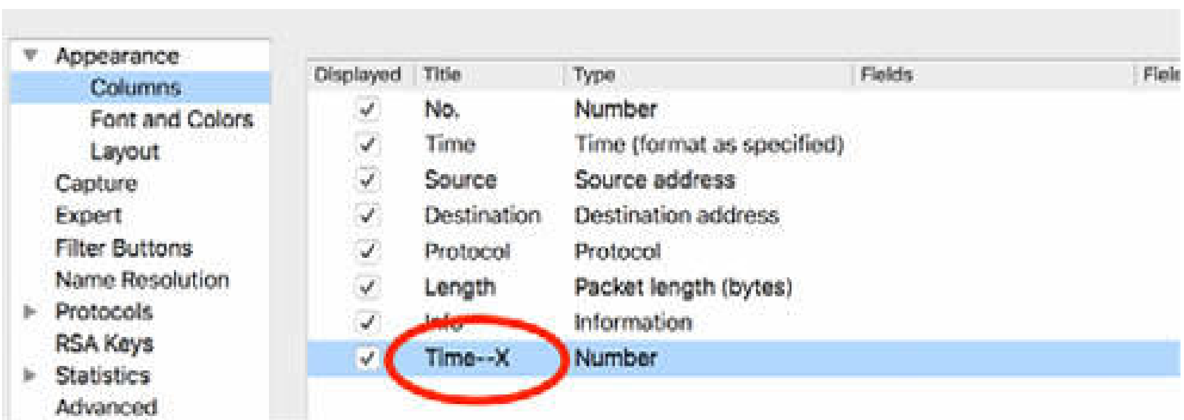
Task 2:
Click the add (+) icon to add more columns to the default ones.



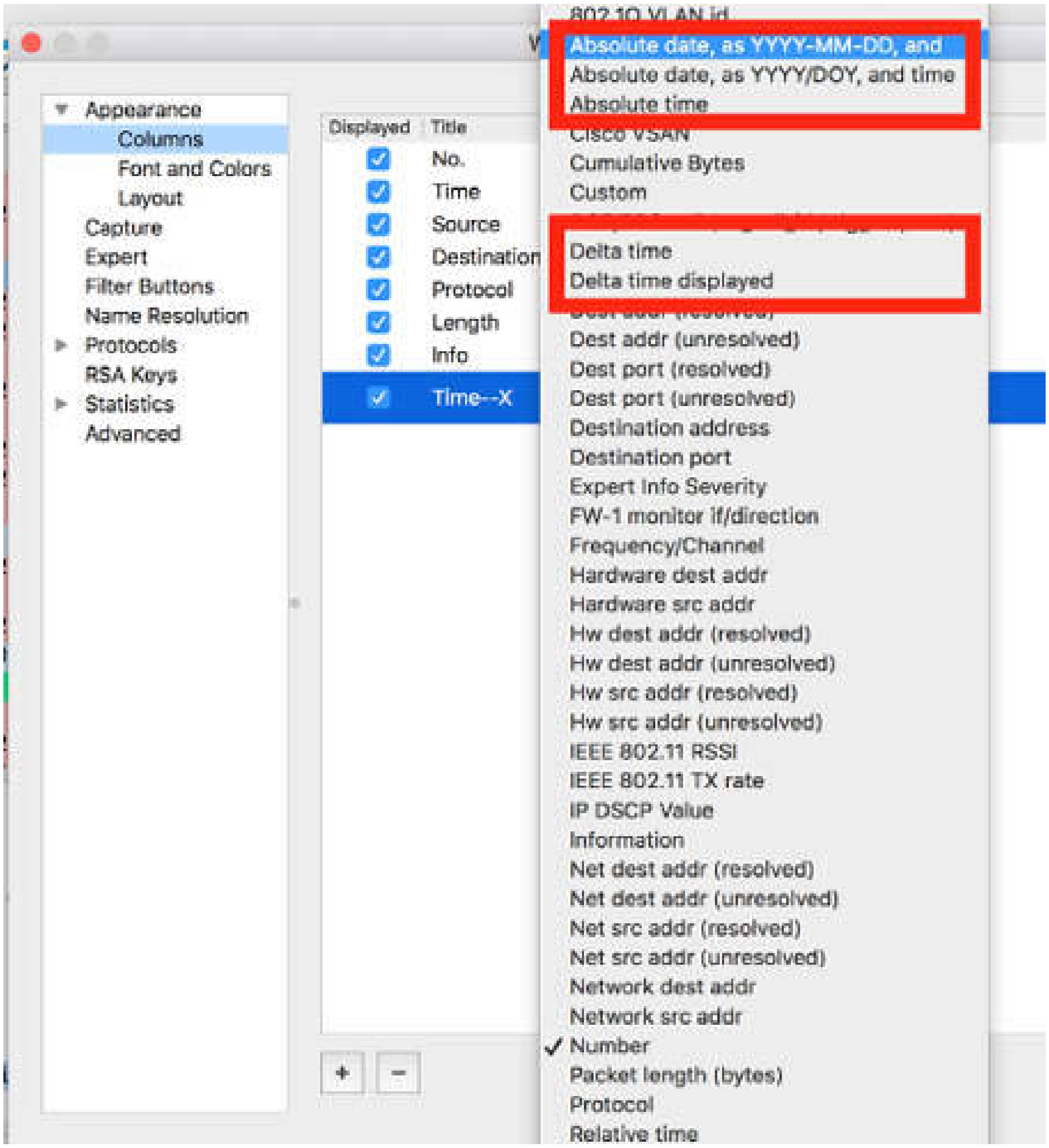
Task 3:
The new column is displayed, as shown in the figure below.

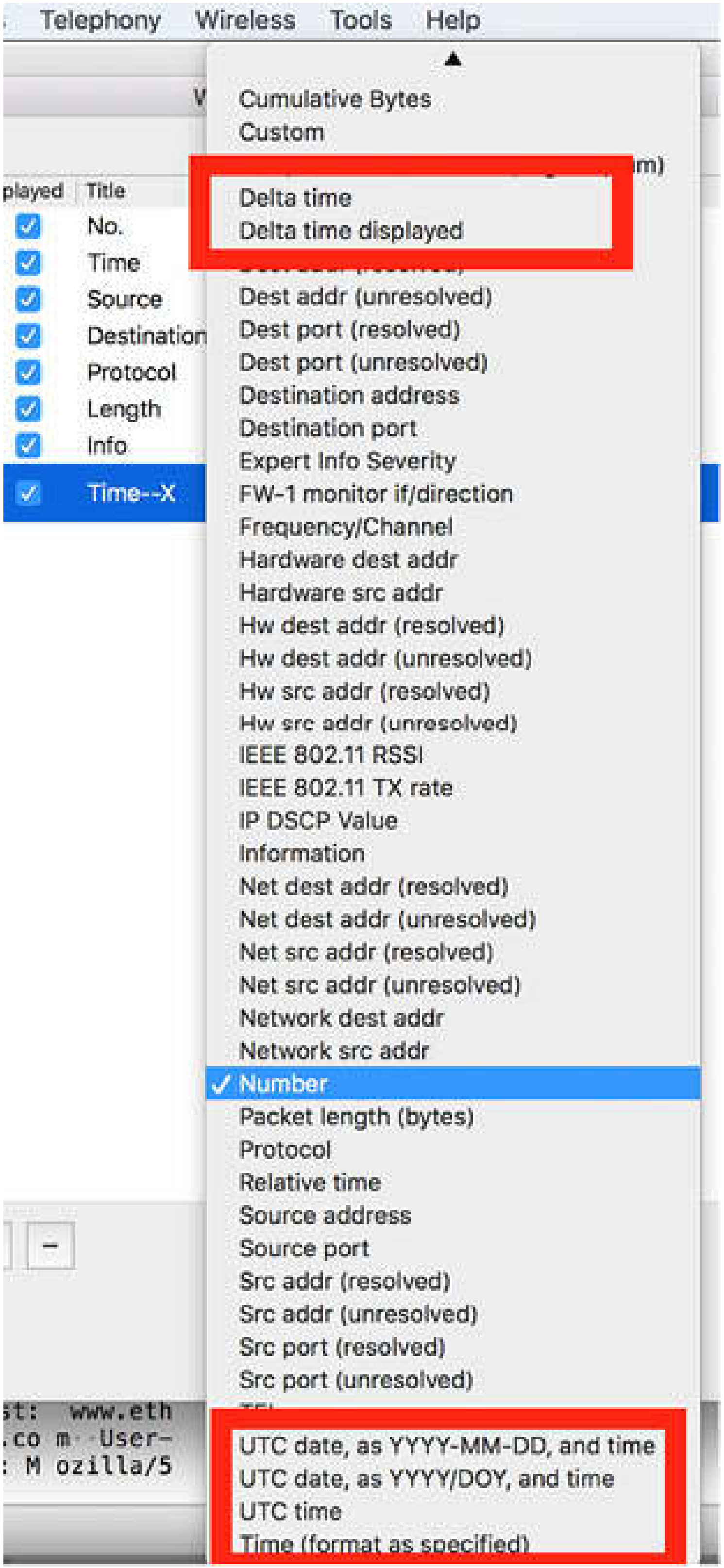


Double-click the column title in the Title field, and enter a name, such as Time--X.

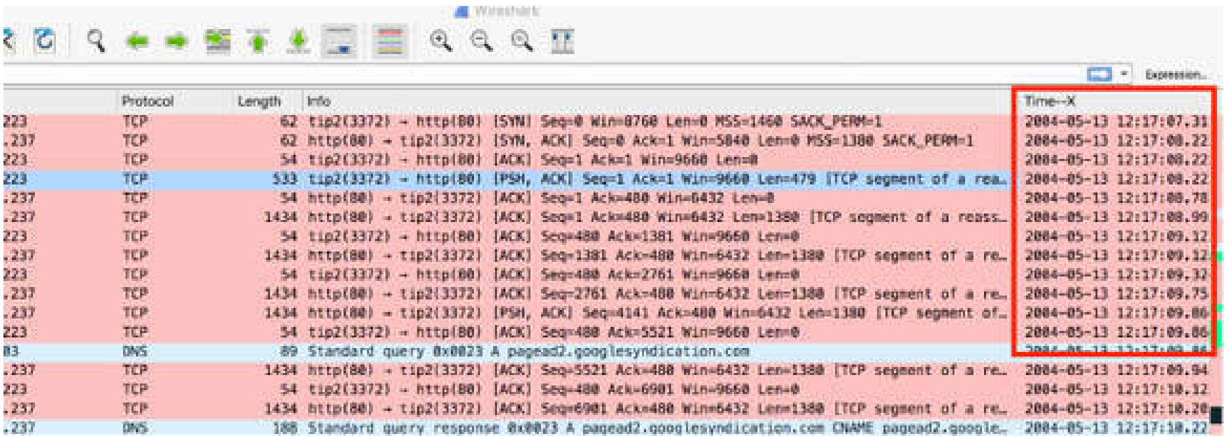


Task 4:
Double-click the column type in the Type field. A pop-up list is displayed. Select one absolute date from the list of all available time formats, displayed in the figures below.

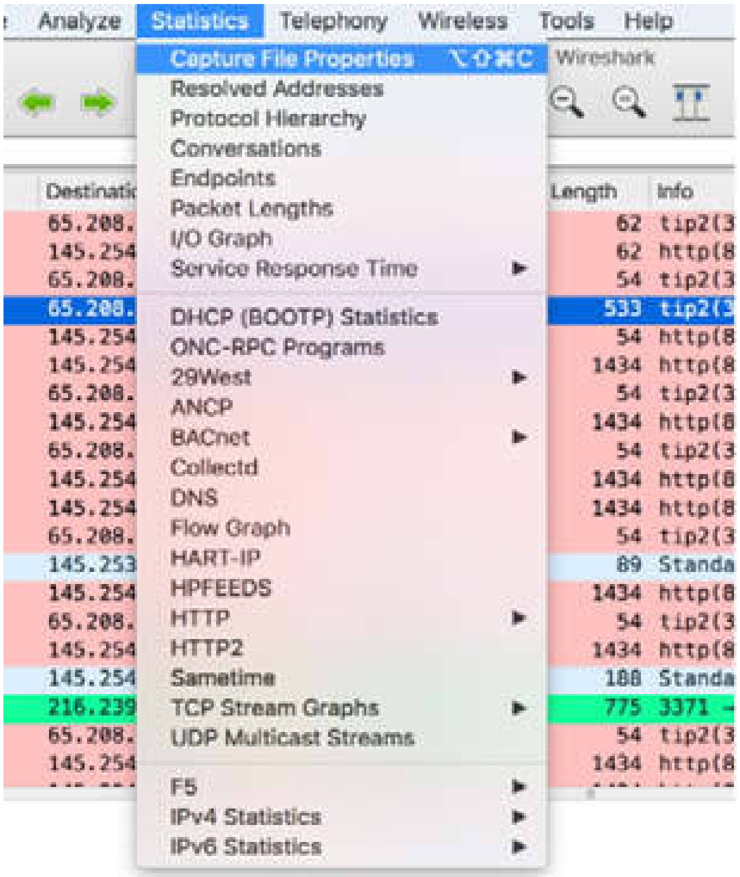

Appearance
Columns
Font and Colors
Layout
Capture
Expert
Filter Buttons
Name Resolution
Protocols
RSA Keys
Statistics
Advanced
Displayed
Title
No.
Time
Source
Destination
Protocol
Length
Info
Time--X
802.1Q VLAN id
Absolute date, as YYYY-MM-DD, and
Absolute date, as YYYY/DOY, and time
Absolute time
Cisco VSAN
Cumulative Bytes
Custom
Delta time
Delta time displayed
Dest addr (unresolved)
Dest port (resolved)
Dest port (unresolved)
Destination address
Destination port
Expert Info Severity
FW-1 monitor if/direction
Frequency/Channel
Hardware dest addr
Hardware src addr
Hw dest addr (resolved)
Hw dest addr (unresolved)
Hw src addr (resolved)
Hw src addr (unresolved)
IEEE 802.11 RSSI
IEEE 802.11 TX rate
IP DSCP Value
Information
Net dest addr (resolved)
Net dest addr (unresolved)
Net src addr (resolved)
Net src addr (unresolved)
Network dest addr
Network src addr
Number
Packet length (bytes)
Protocol
Relative time


Telephony Wireless Tools Help
Cumulative Bytes
Custom
Delta time
Delta time displayed
Dest addr (unresolved)
Dest port (resolved)
Dest port (unresolved)
Destination address
Destination port
Expert Info Severity
FW-1 monitor if/direction
Frequency/Channel
Hardware dest addr
Hardware src addr
Hw dest addr (resolved)
Hw dest addr (unresolved)
Hw src addr (resolved)
Hw src addr (unresolved)
IEEE 802.11 RSSI
IEEE 802.11 TX rate
IP DSCP Value
Information
Net dest addr (resolved)
Net dest addr (unresolved)
Net src addr (resolved)
Net src addr (unresolved)
Network dest addr
Network src addr
Number
Packet length (bytes)
Protocol
Relative time
Source address
Source port
Src addr (resolved)
Src addr (unresolved)
Src port (resolved)
Src port (unresolved)
UTC date, as YYYY-MM-DD, and time
UTC date, as YYYY/DOY, and time
UTC time
Time (format as specified)
Title
No.
Time
Source
Destination
Protocol
Length
Info
Time--X

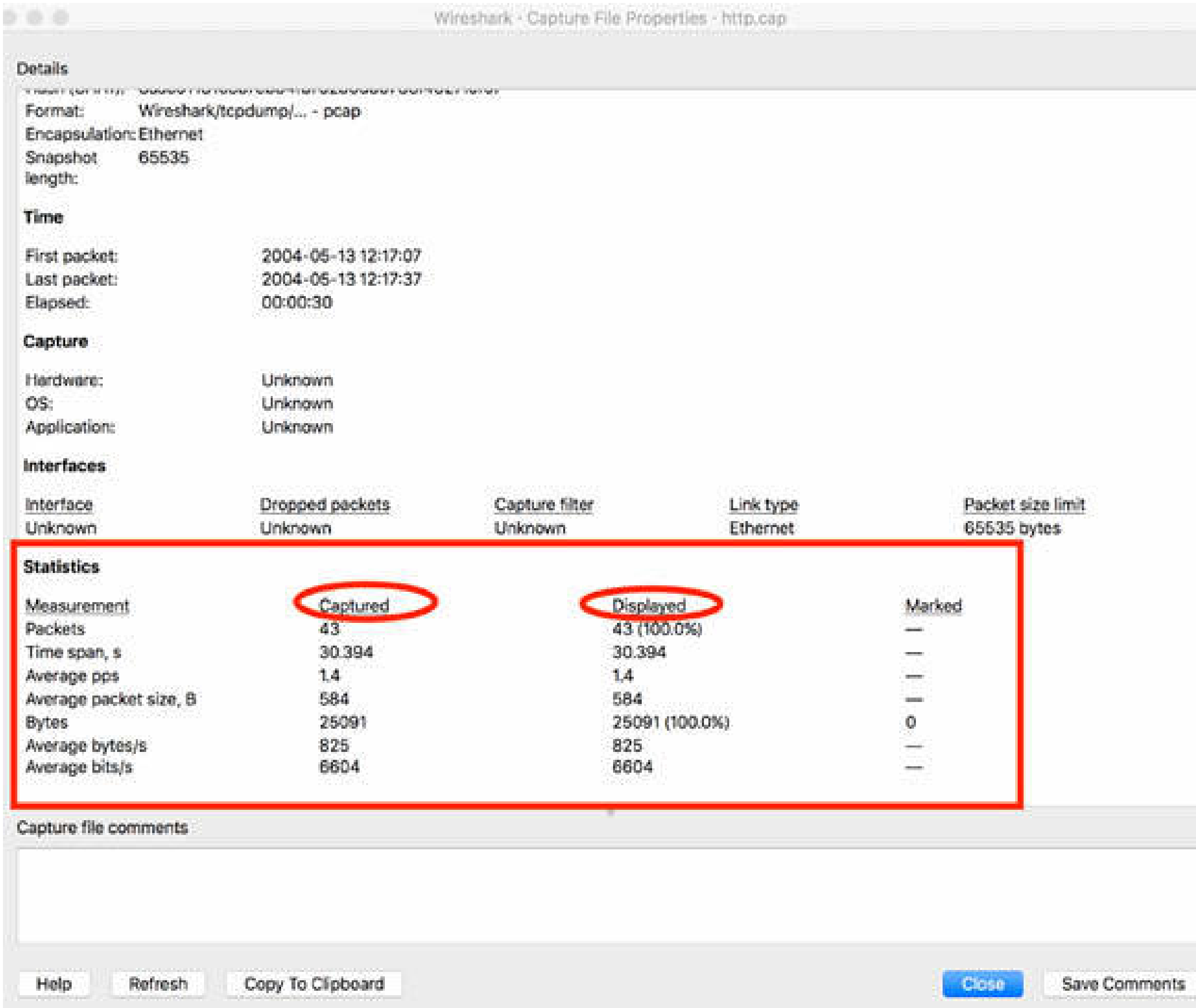
Task 5:
A new column named Column—X, with the time format chosen, is displayed in the Packet List pane.



Task 6:
To view the capture statistics, on the main menu, select Statistics > Capture File Properties.



Task 7:
The Capture File Properties dialog box is displayed containing the capture file statistics related to the traffic rates and packet bytes. The Captured column in the Statistics section shows statistics for all capture files. The Displayed column in the Statistics section shows statistics related to the displayed packets when a display filter has been applied.



Notes:

To gain more confidence in using the time columns, add more columns by using different time formats and then compare the results between different columns.

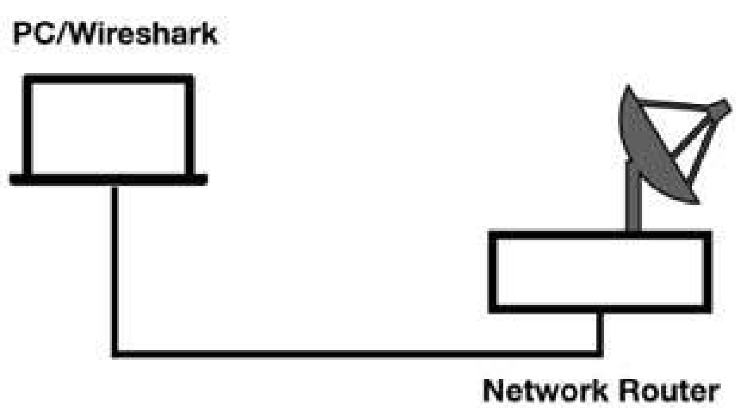
Lab 17. Statistics (Protocols—Conversation)

Lab Objective:
Learn how to launch the Statistics page and manage basic information.

Lab Purpose:
Understand the basic statistics information contained in the Statistic page to identify protocols, applications, and conversations.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

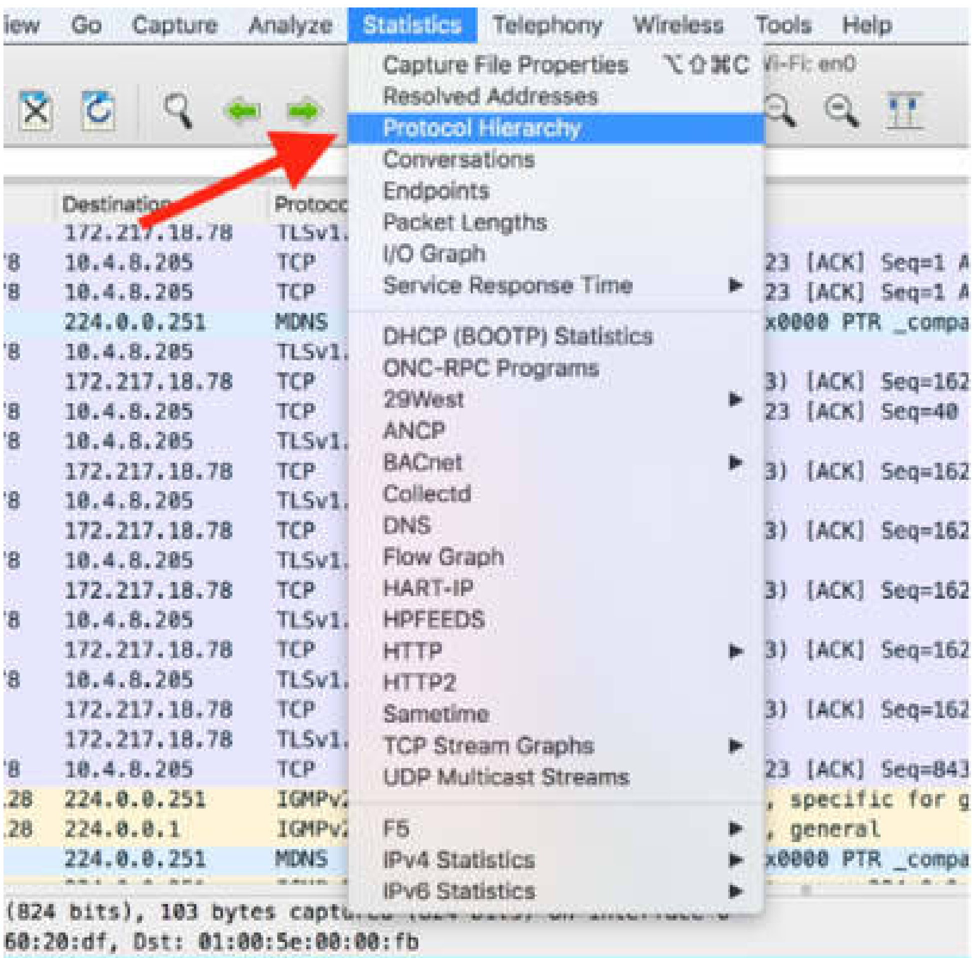


Lab Walkthrough:

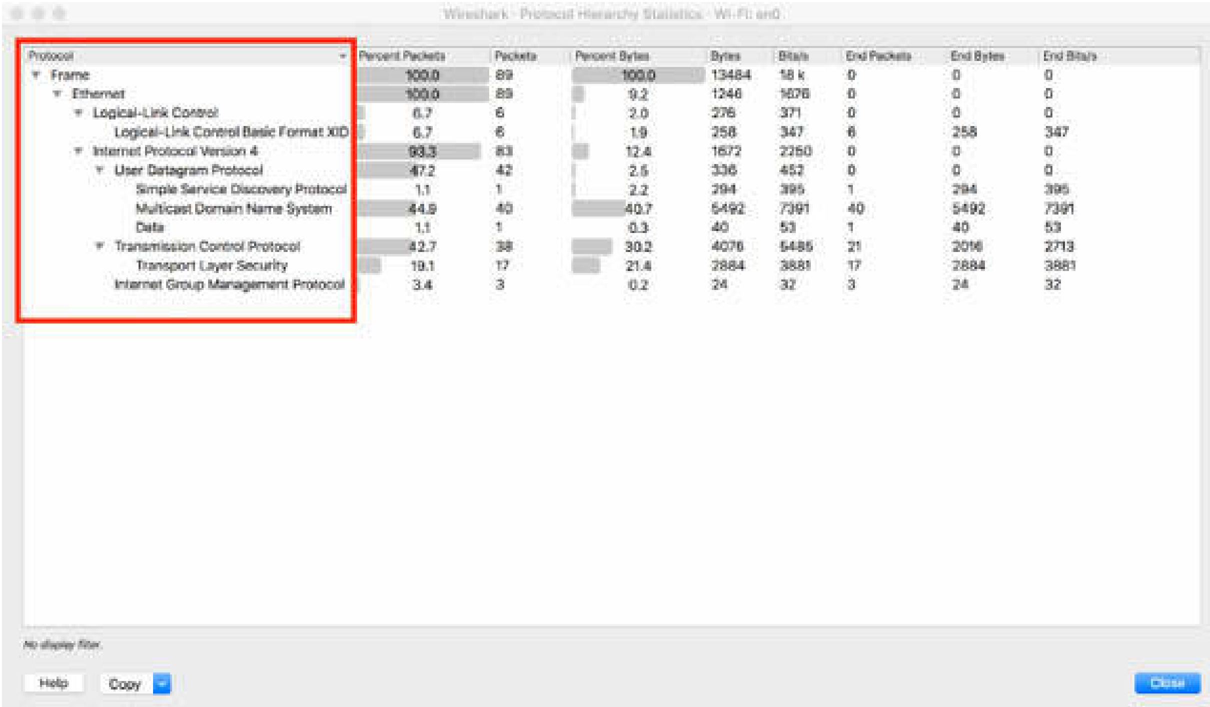
Task 1:
Verify the physical connection between the PC (equipped with Wireshark) and the network router connected to the internet.

Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column, and then capture the traffic for a few minutes.

Task 2:
On the main menu, select Statistics > Protocol Hierarchy.

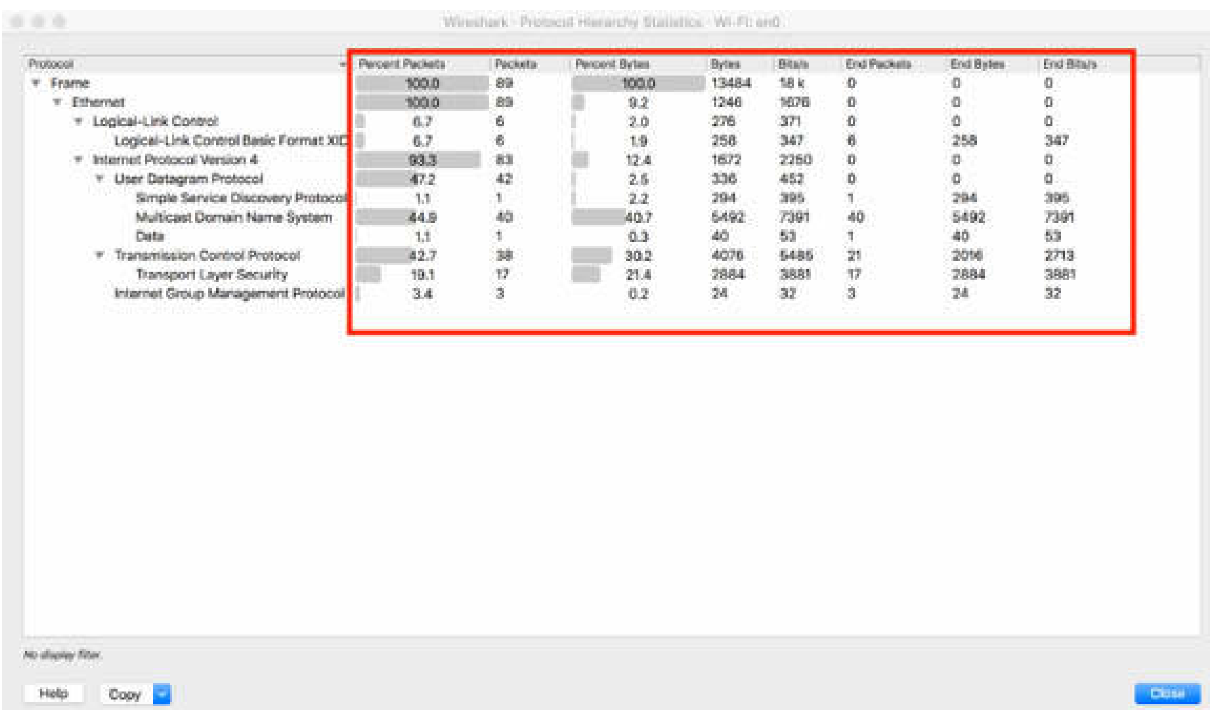


Task 3:
The Protocol Hierarchy Statistics window is displayed showing the protocol tree.



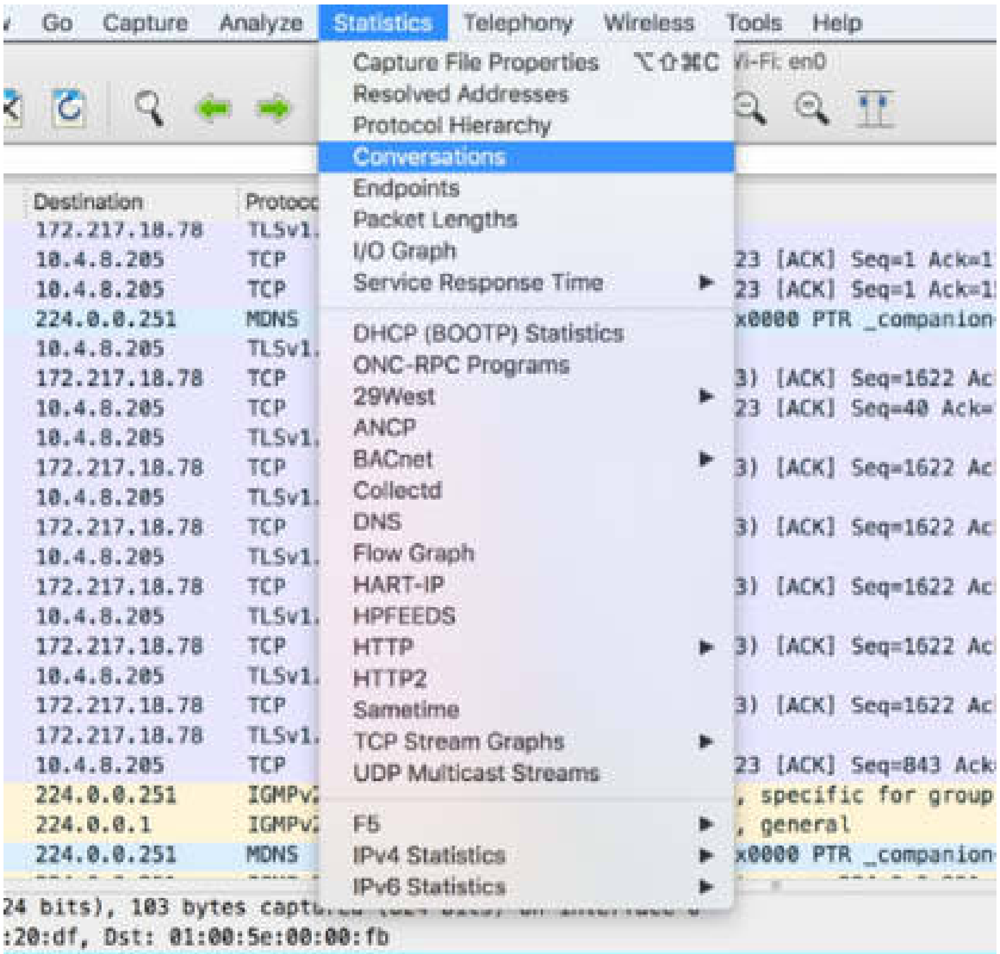
Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	89	100.0	13484	18 k	0	0	0
Ethernet	100.0	89	9.2	1246	1676	0	0	0
Logical-Link Control	6.7	6	2.0	276	371	0	0	0
Logical-Link Control Basic Format XID	6.7	6	1.9	258	347	6	258	347
Internet Protocol Version 4	93.3	83	12.4	1672	2250	0	0	0
User Datagram Protocol	47.2	42	2.5	336	452	0	0	0
Simple Service Discovery Protocol	1.1	1	2.2	294	395	1	294	395
Multicast Domain Name System	44.9	40	40.7	5492	7391	40	5492	7391
Data	1.1	1	0.3	40	53	1	40	53
Transmission Control Protocol	42.7	38	30.2	4076	5485	21	2016	2713
Transport Layer Security	19.1	17	21.4	2884	3881	17	2884	3881
Internet Group Management Protocol	3.4	3	0.2	24	32	3	24	32

For each protocol, the numbers of packets and bytes in the current log file are displayed.

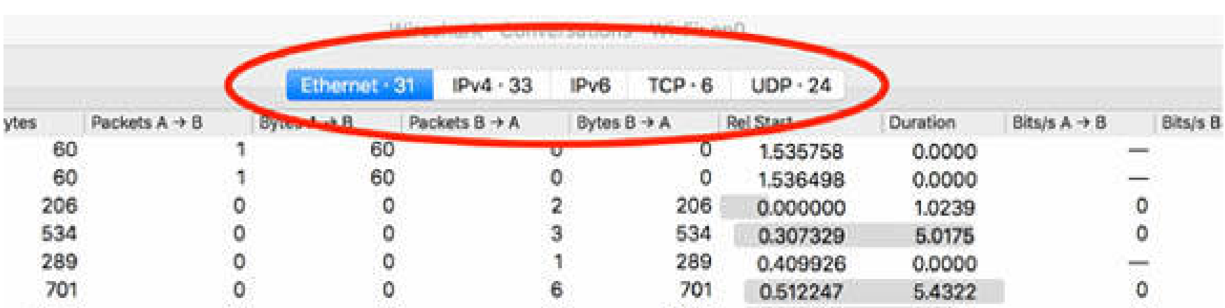


Task 4:

On the main menu, select Statistics > Conversations.



The Conversations dialog box is displayed containing the conversation details for each protocol type in the current log file.



Task 5:

Click TCP. The related statistics are displayed, containing the following:

- IP addresses of the conversation

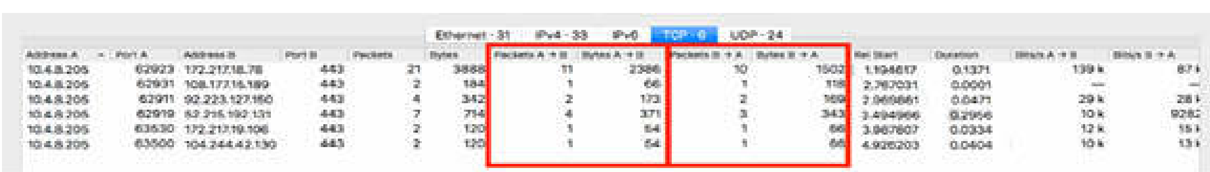
Ethernet · 31 | IPv4 · 33 | IPv6 | TCP · 6 | UDP · 24

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.4.8.205	62923	172.217.18.78	443	21	3888	11	2386	10	1502	1.194617	0.1371	139 k	87 k
10.4.8.205	62931	108.177.16.189	443	2	184	1	66	1	118	2.767031	0.0001	—	—
10.4.8.205	62911	92.223.127.160	443	4	342	2	173	2	169	2.965661	0.0471	29 k	28 k
10.4.8.205	62919	62.216.192.131	443	7	714	4	371	3	343	3.494966	0.2956	10 k	9282
10.4.8.205	63530	172.217.19.106	443	2	120	1	54	1	66	3.967607	0.0334	12 k	15 k
10.4.8.205	63500	104.244.42.130	443	2	120	1	54	1	66	4.926203	0.0404	10 k	13 k

- Associated ports

- Number of packets and bytes for each conversation

- Number of packets and bytes for each direction of the conversation (from A to B and from B to A)



- The relative start time and the duration (in seconds) for each conversation

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.4.8.205	62923	172.217.18.78	443	21	3888	11	2386	10	1502	1.194617	0.1371	139 k	87 k
10.4.8.205	62931	108.177.15.189	443	2	184	1	66	1	118	2.767031	0.0001	—	—
10.4.8.205	62911	92.223.127.160	443	4	342	2	173	2	169	2.969861	0.0471	29 k	28 k
10.4.8.205	62919	52.215.192.131	443	7	714	4	371	3	343	3.494966	0.2956	10 k	9282
10.4.8.205	63530	172.217.19.106	443	2	120	1	54	1	66	3.987807	0.0334	12 k	15 k
10.4.8.205	63500	104.244.42.130	443	2	120	1	54	1	66	4.926203	0.0404	10 k	13 k

- The bit rate for each direction of the conversation

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.4.8.205	62923	172.217.18.78	443	21	3888	11	2386	10	1502	1.194617	0.1371	139 k	87 k
10.4.8.205	62931	108.177.15.189	443	2	184	1	66	1	118	2.767031	0.0001	—	—
10.4.8.205	62911	92.223.127.160	443	4	342	2	173	2	169	2.969861	0.0471	29 k	28 k
10.4.8.205	62919	52.215.192.131	443	7	714	4	371	3	343	3.494966	0.2956	10 k	9282
10.4.8.205	63530	172.217.19.106	443	2	120	1	54	1	66	3.987807	0.0334	12 k	15 k
10.4.8.205	63500	104.244.42.130	443	2	120	1	54	1	66	4.926203	0.0404	10 k	13 k

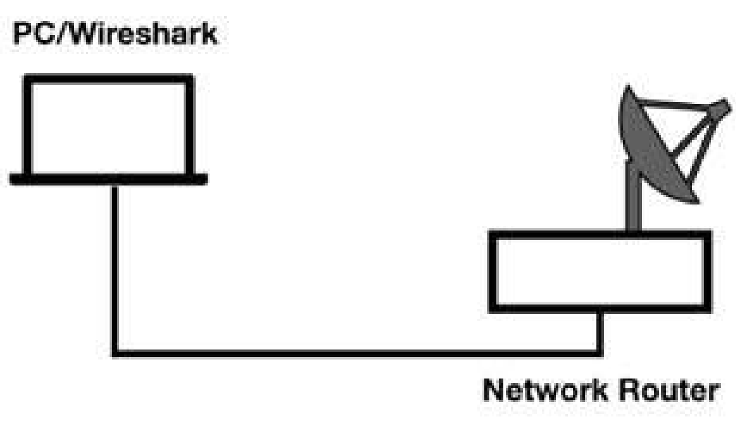
Lab 18. Statistics (Endpoint—Packet Lengths)

Lab Objective:
Learn how to view endpoint maps and evaluate packet lengths.

Lab Purpose:
Understand how to view the endpoint list and generate a map on the Earth and how to evaluate packet lengths.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:

Open a terminal window, and ping the address 101labs.net by running the ping 101labs.net command. In the Windows operating system, you can specify a count with the /n switch.

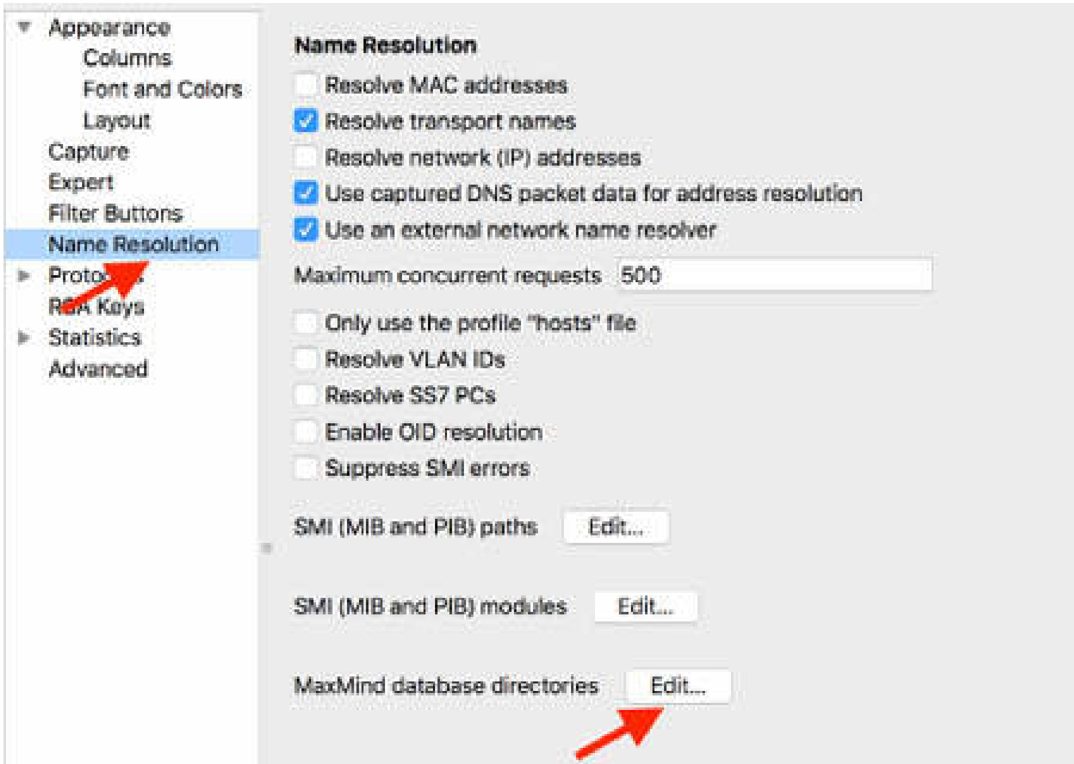
```
[redacted]$ ping 101labs.net
PING 101labs.net (146.66.102.134): 56 data bytes
64 bytes from 146.66.102.134: icmp_seq=0 ttl=51 time=53.529 ms
64 bytes from 146.66.102.134: icmp_seq=1 ttl=51 time=53.750 ms
Request timeout for icmp_seq 2
Request timeout for icmp_seq 3
64 bytes from 146.66.102.134: icmp_seq=3 ttl=51 time=1738.649 ms
64 bytes from 146.66.102.134: icmp_seq=5 ttl=51 time=76.513 ms
64 bytes from 146.66.102.134: icmp_seq=6 ttl=51 time=51.141 ms
64 bytes from 146.66.102.134: icmp_seq=7 ttl=51 time=61.170 ms
64 bytes from 146.66.102.134: icmp_seq=8 ttl=51 time=54.960 ms
64 bytes from 146.66.102.134: icmp_seq=9 ttl=51 time=67.990 ms
64 bytes from 146.66.102.134: icmp_seq=10 ttl=51 time=56.570 ms
64 bytes from 146.66.102.134: icmp_seq=11 ttl=51 time=49.258 ms
64 bytes from 146.66.102.134: icmp_seq=12 ttl=51 time=50.176 ms
64 bytes from 146.66.102.134: icmp_seq=13 ttl=51 time=51.031 ms
64 bytes from 146.66.102.134: icmp_seq=14 ttl=51 time=58.328 ms
64 bytes from 146.66.102.134: icmp_seq=15 ttl=51 time=50.975 ms
```

Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes and then stop the capture and save the file.

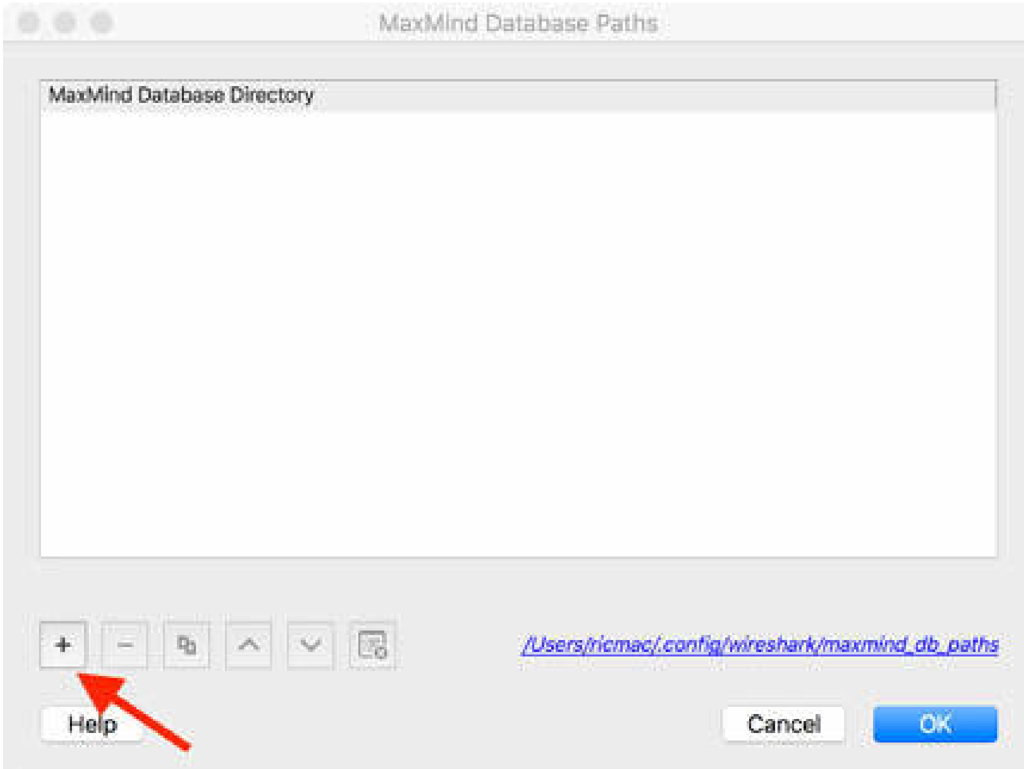
Task 2:

From **https://dev.maxmind.com/geoip/geoip2/geolite2/** , download the GeoLite2 City database file. Extract the contents to a folder on your PC and note down the location.

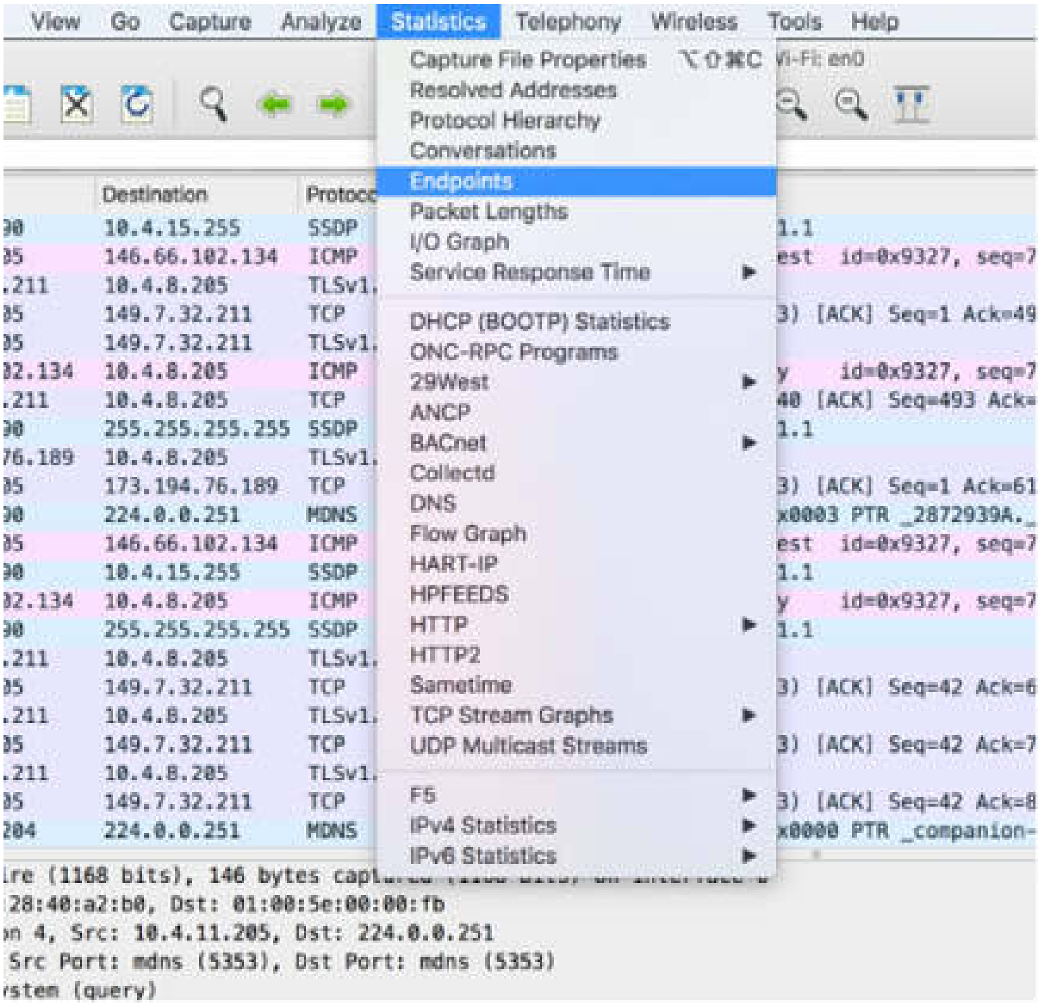
On the main menu, select Edit > Preferences. In the left tree view, click the Edit button next to the MaxMind Database Directories option, as shown in the figure below.



Task 3:
In the MaxMind Database Paths dialog box, click the add button (+) to specify the database location. Click OK, and restart Wireshark.

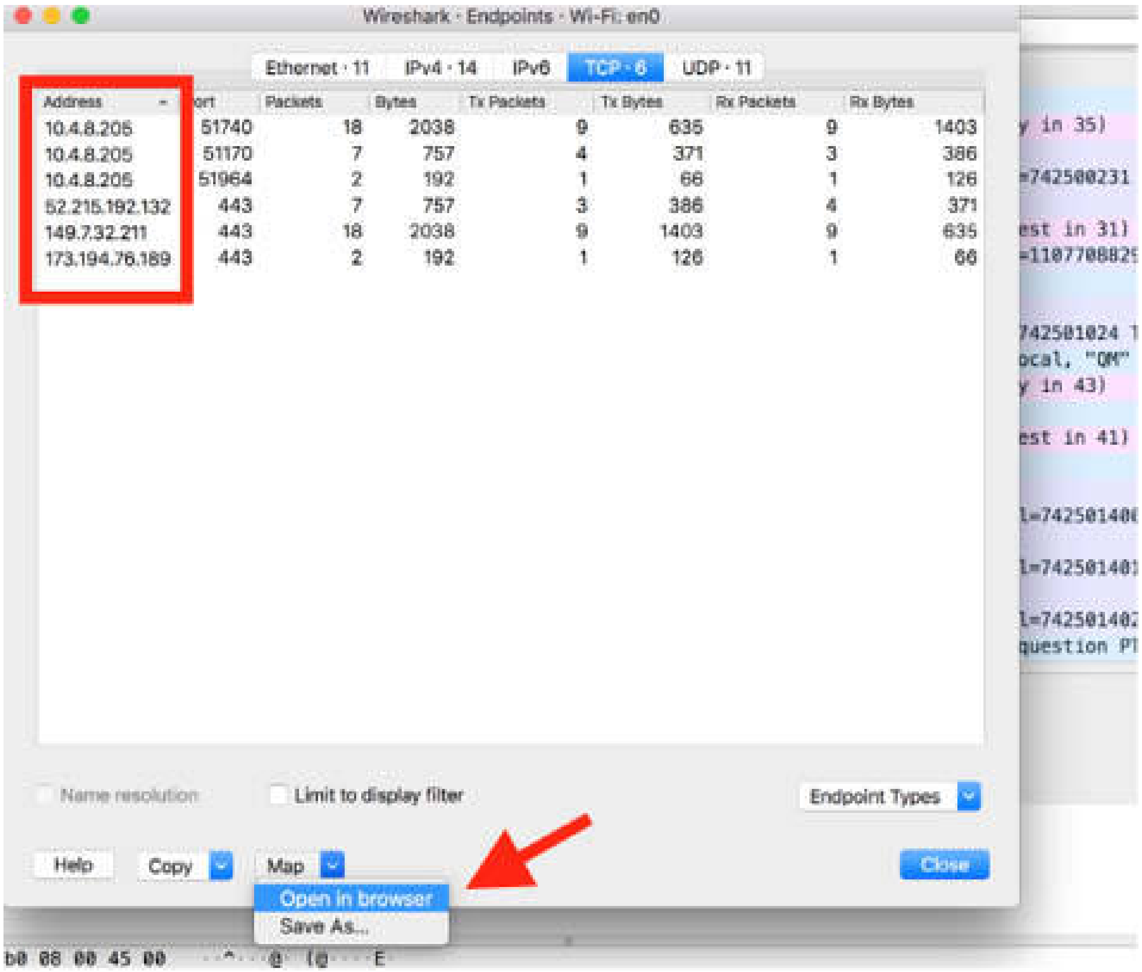


Task 4:
Open the capture file saved in Task 1 and then on the main menu, select Statistics > Endpoints.



The Endpoints dialog box is displayed containing all IP endpoints present in the saved capture file.

Click Map > Open in browser, as shown in the figure below.

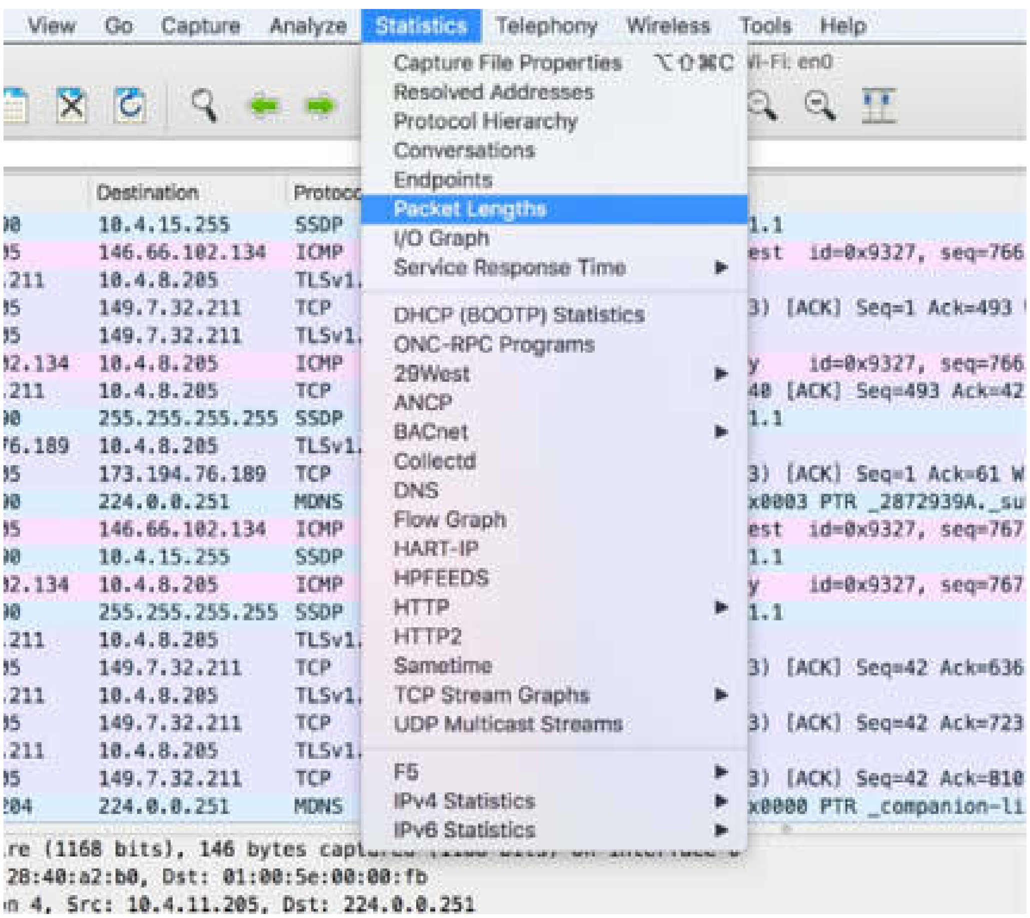


Task 5:

A browser window indicating the location of the endpoints on the Earth is displayed. Hovering over with a mouse displays the related IP addresses, as shown in the figure below.

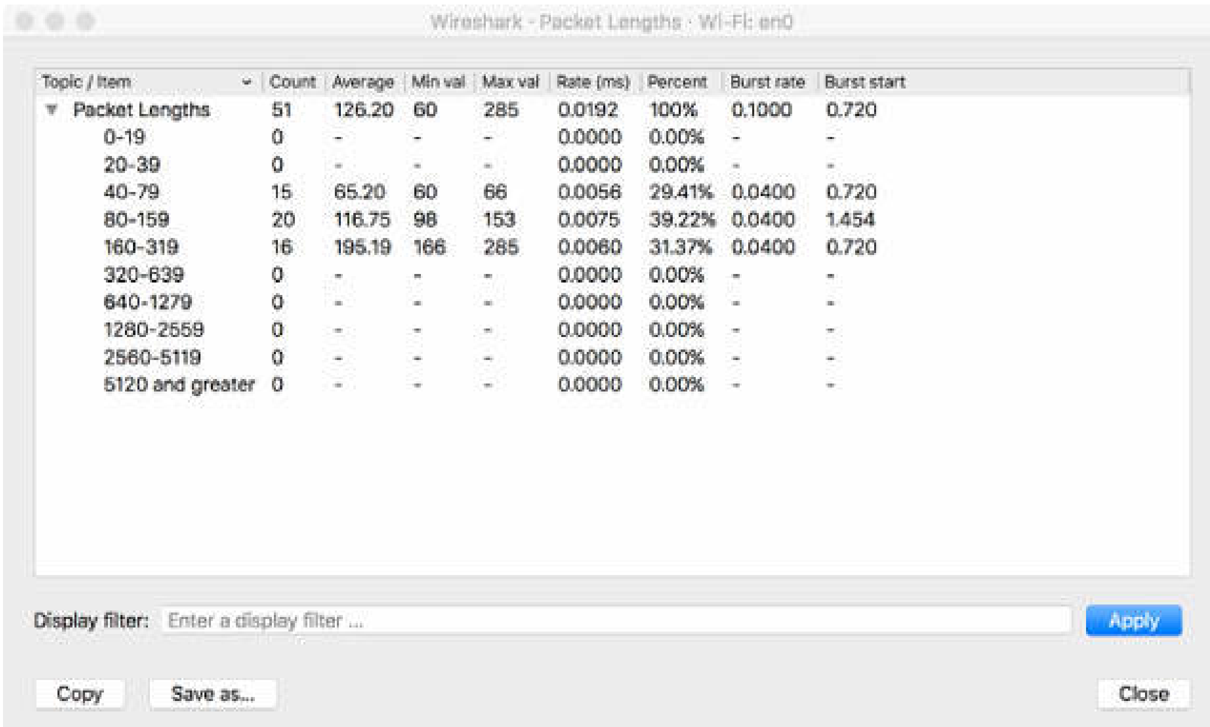
Task 6:

On the main menu, select Statistics > Packet Lengths, as shown in the figure below.



The Packet Length dialog box is displayed containing the statistics related to the packet length. For each predefined packet length range, the count of the number of packets present in the capture file is displayed along with the average, minimum, and maximum count for each range.

For each length range, the rate in ms and the percentage of the selected range over the total number of packets are displayed.



Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst rate	Burst start
Packet Lengths	51	126.20	60	285	0.0192	100%	0.1000	0.720
0-19	0	-	-	-	0.0000	0.00%	-	-
20-39	0	-	-	-	0.0000	0.00%	-	-
40-79	15	65.20	60	66	0.0056	29.41%	0.0400	0.720
80-159	20	116.75	98	153	0.0075	39.22%	0.0400	1.454
160-319	16	195.19	166	285	0.0060	31.37%	0.0400	0.720
320-639	0	-	-	-	0.0000	0.00%	-	-
640-1279	0	-	-	-	0.0000	0.00%	-	-
1280-2559	0	-	-	-	0.0000	0.00%	-	-
2560-5119	0	-	-	-	0.0000	0.00%	-	-
5120 and greater	0	-	-	-	0.0000	0.00%	-	-

Notes:
To gain more confidence in using the statistics information, ping a different IP location. Observe the change in the location of the endpoints, and if and how the packet length values change.

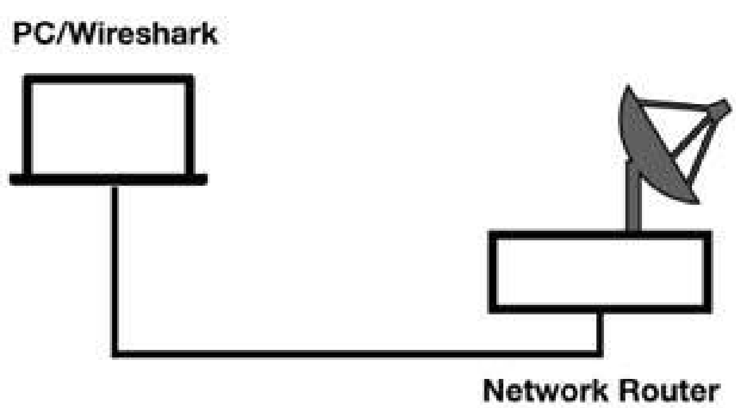
Lab 19. Statistics (I/O Graph — UDP Streams—IPv4)

Lab Objective:
Learn how to view I/O graphs, UDP multicast streams, and IPv4 statistics.

Lab Purpose:
Understand how to view I/O graphs and to display UDP multicast streams statistics and IPv4 statistics.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

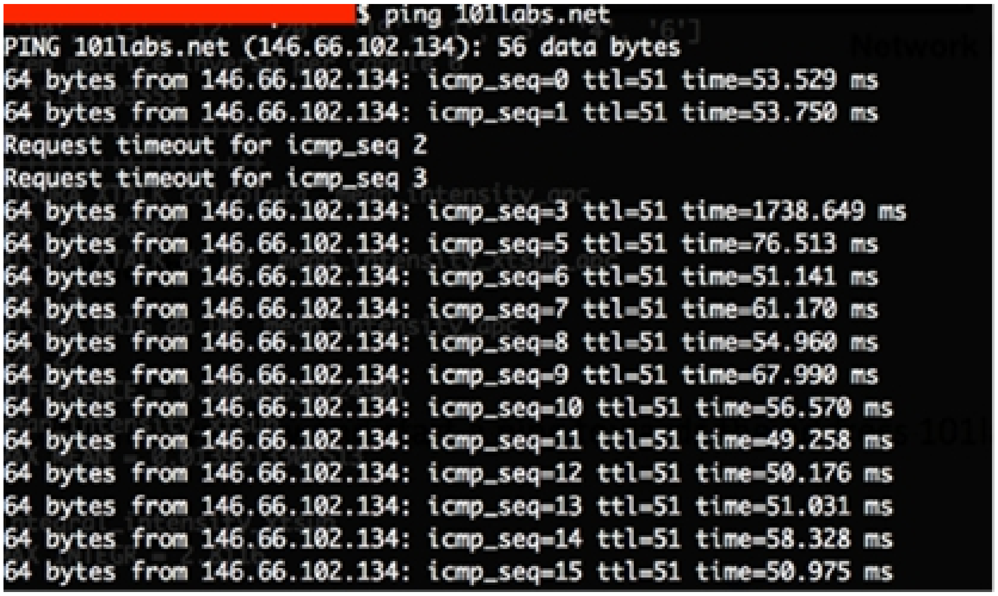
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:

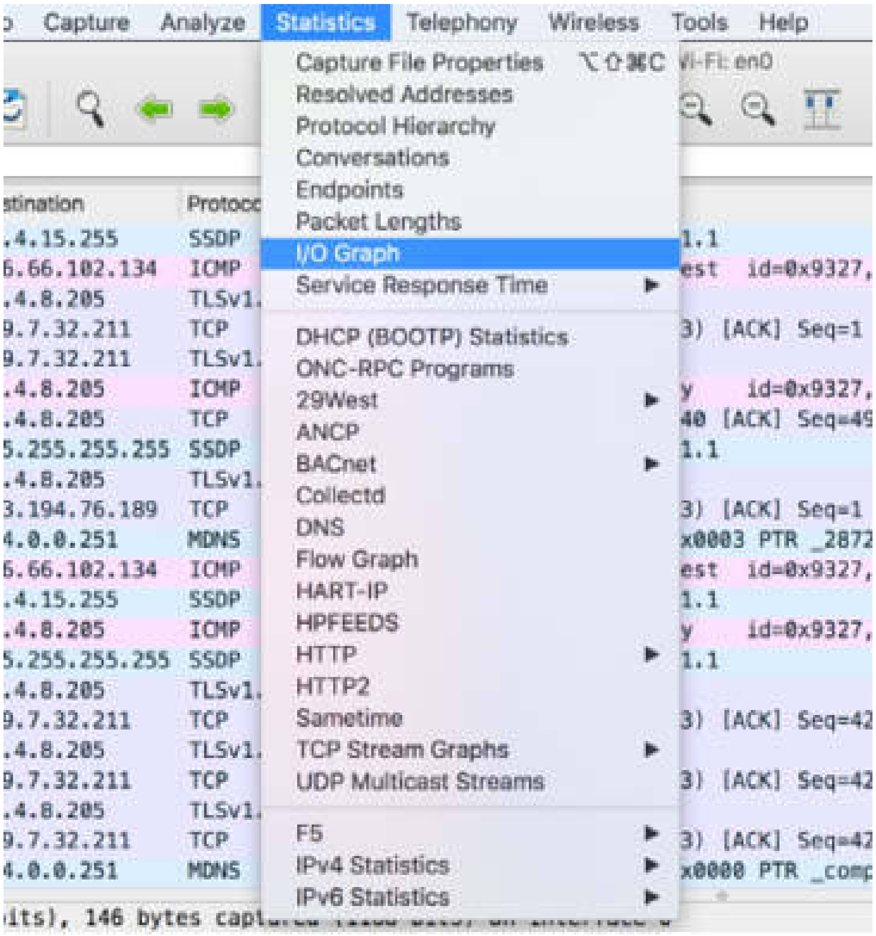
Open a terminal window, and ping the address 101labs.net by running the ping 101labs.net command.



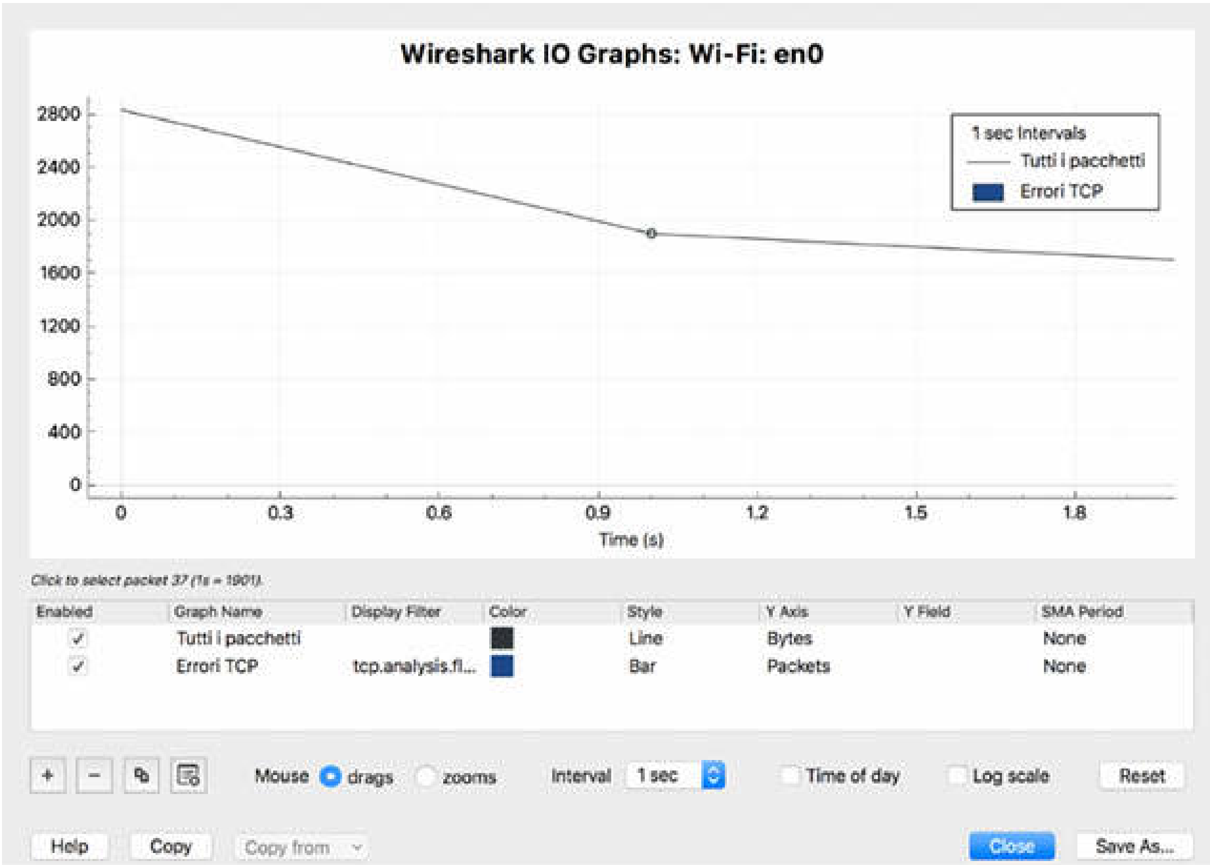
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes. Stop the capture and save the file.

Task 2:

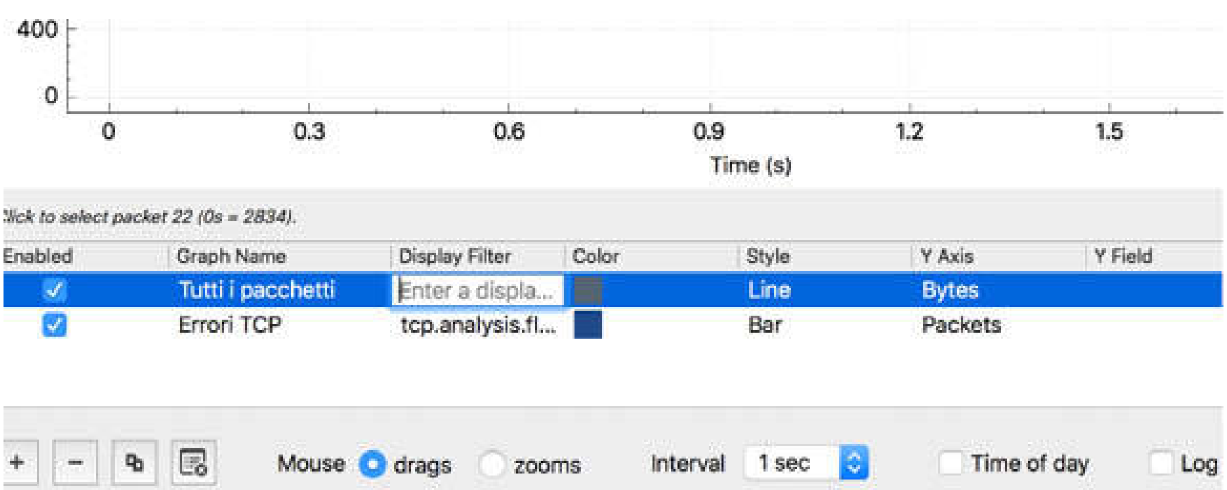
On the main menu, select Statistics > I/O Graph.



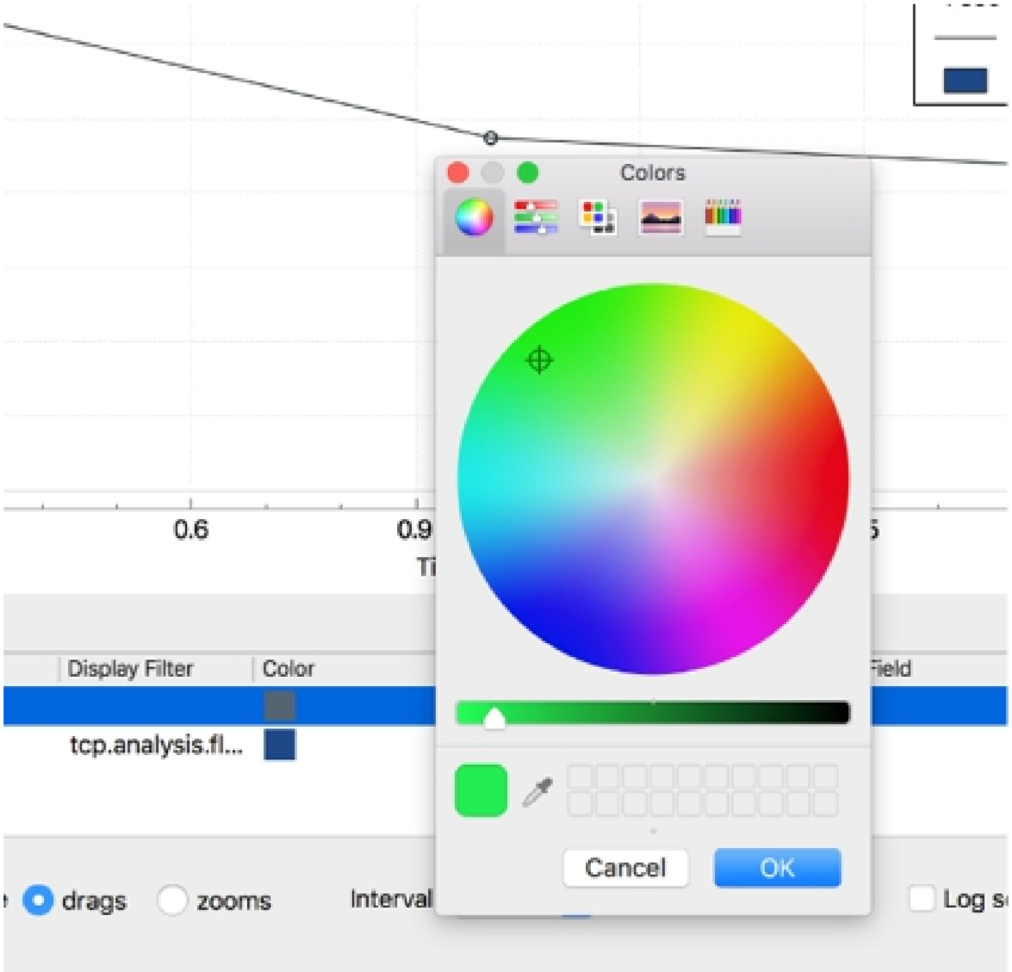
The I/O Graphs dialog box is displayed containing (in the default view) the bytes/sec for the capture log file.



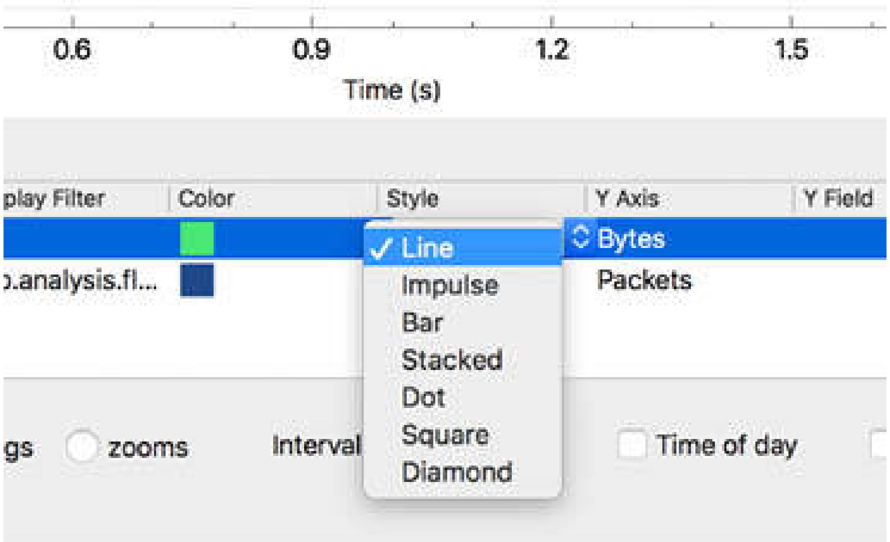
Task 3:
To refer the graph only to a subset of the packets, you can customize the graph by double-clicking the Display Filter field.



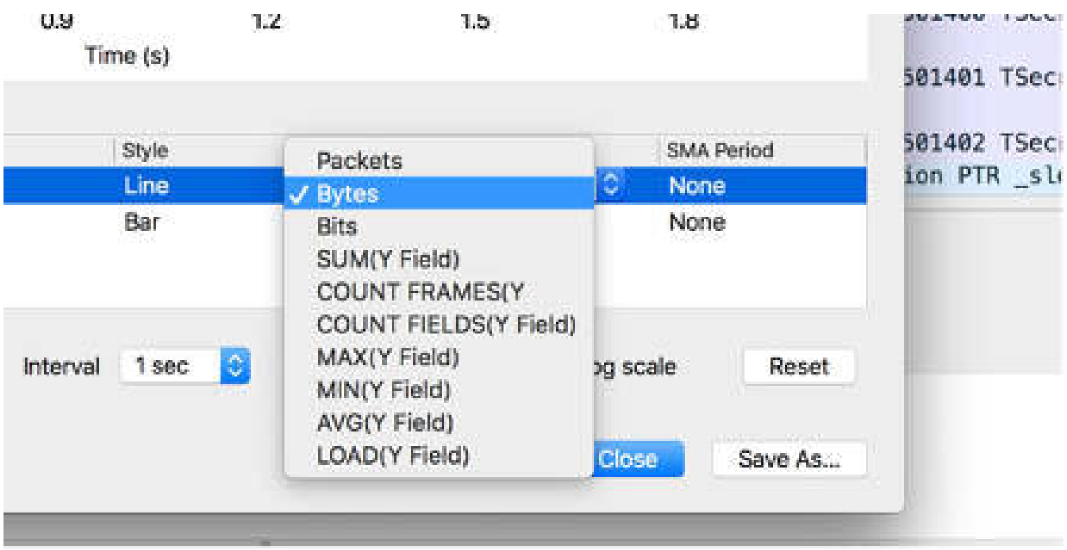
To change the default graph color, click the Color field, and choose a color.



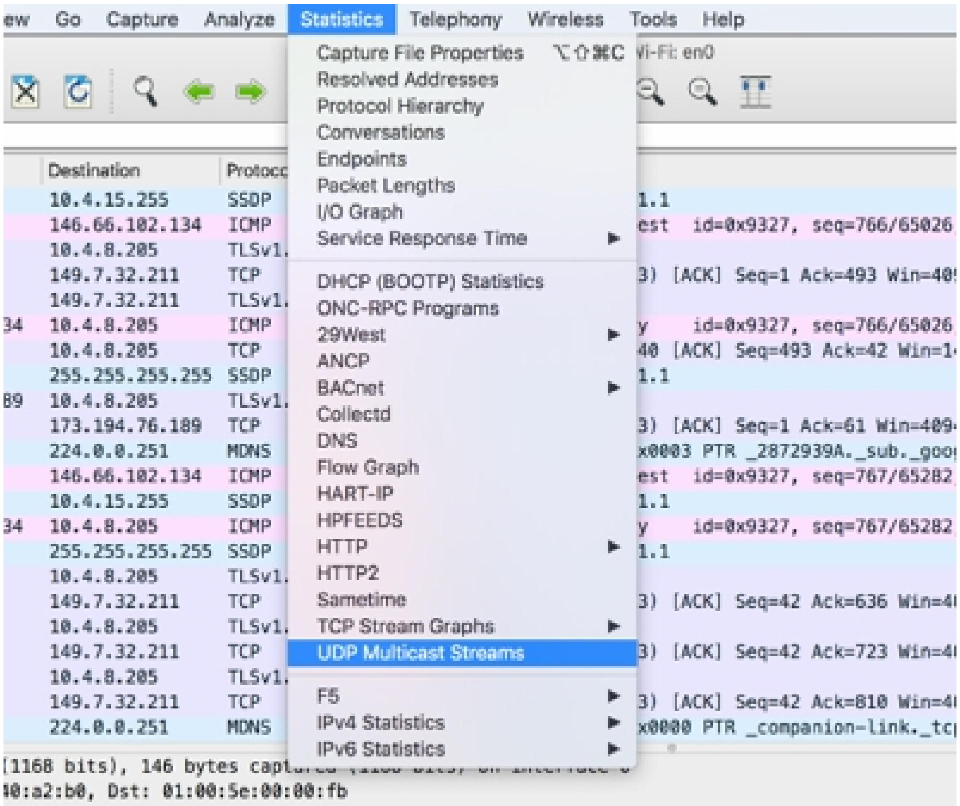
To change the graph line style, click Style, and choose a style.



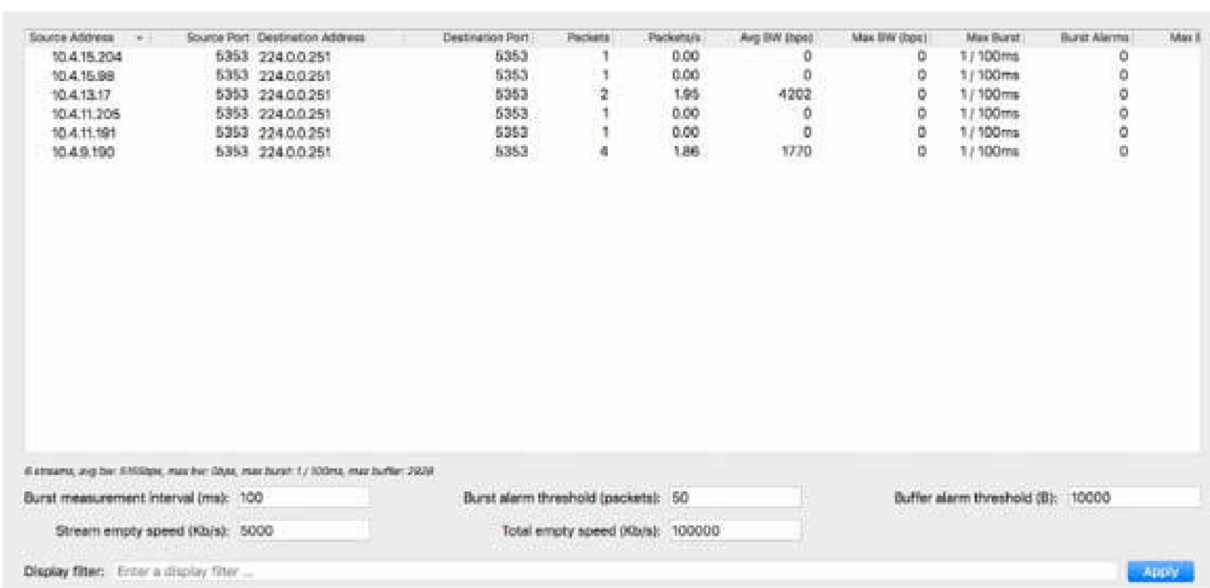
To change the default Y axis scale, click the “Y Axis” field, and choose a scale.



Task 4:
On the main menu, select Statistics > UDP Multicast Streams, as shown in the figure below.

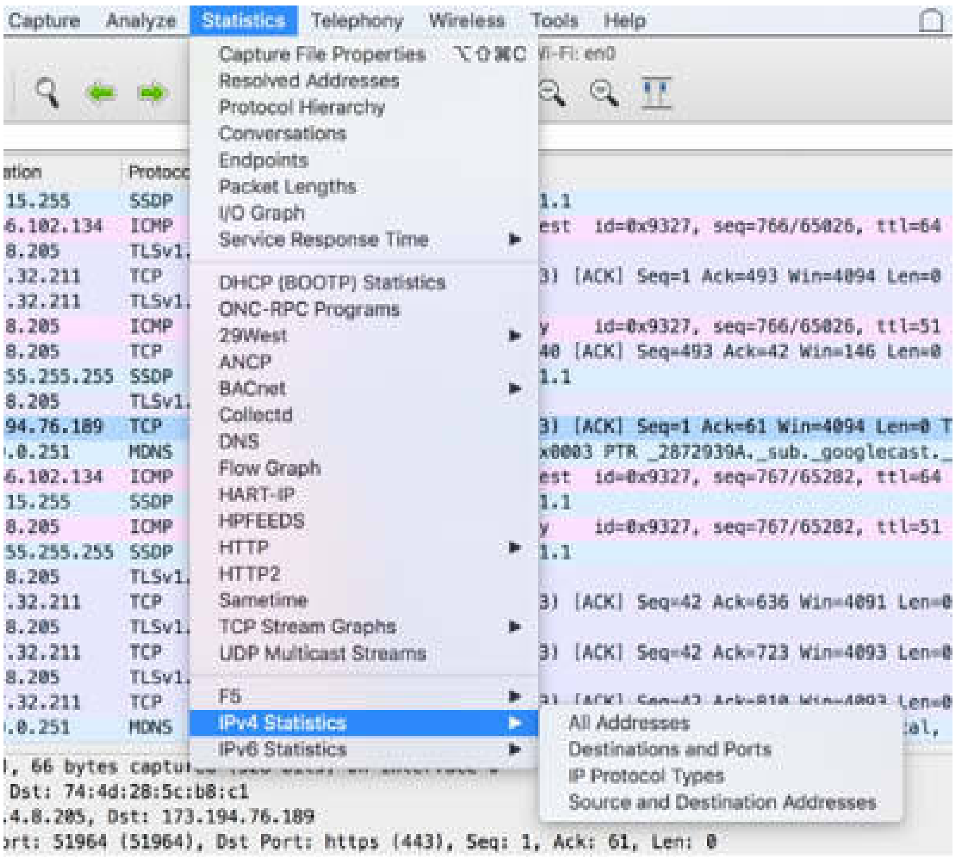


The UDP Multicast Streams dialog box is displayed, showing the statistics for each UDP multicast stream present in the capture file.

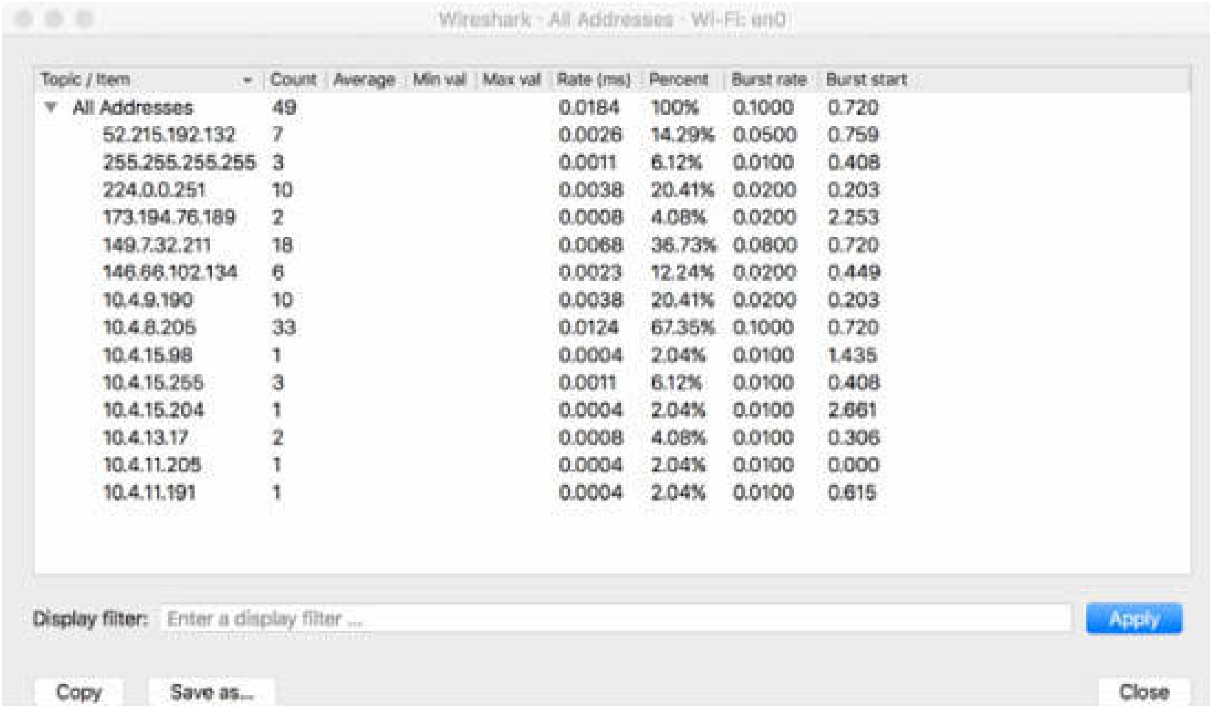


Task 5:

On the main menu, select Statistics > IPv4 Statistics. The following choices are displayed: All Addresses, Destinations and Ports, IP Protocol Types, and Source and Destination Addresses.



When you select All Addresses, the All Addresses dialog box is displayed containing all IPv4 addresses count and percentage packets.

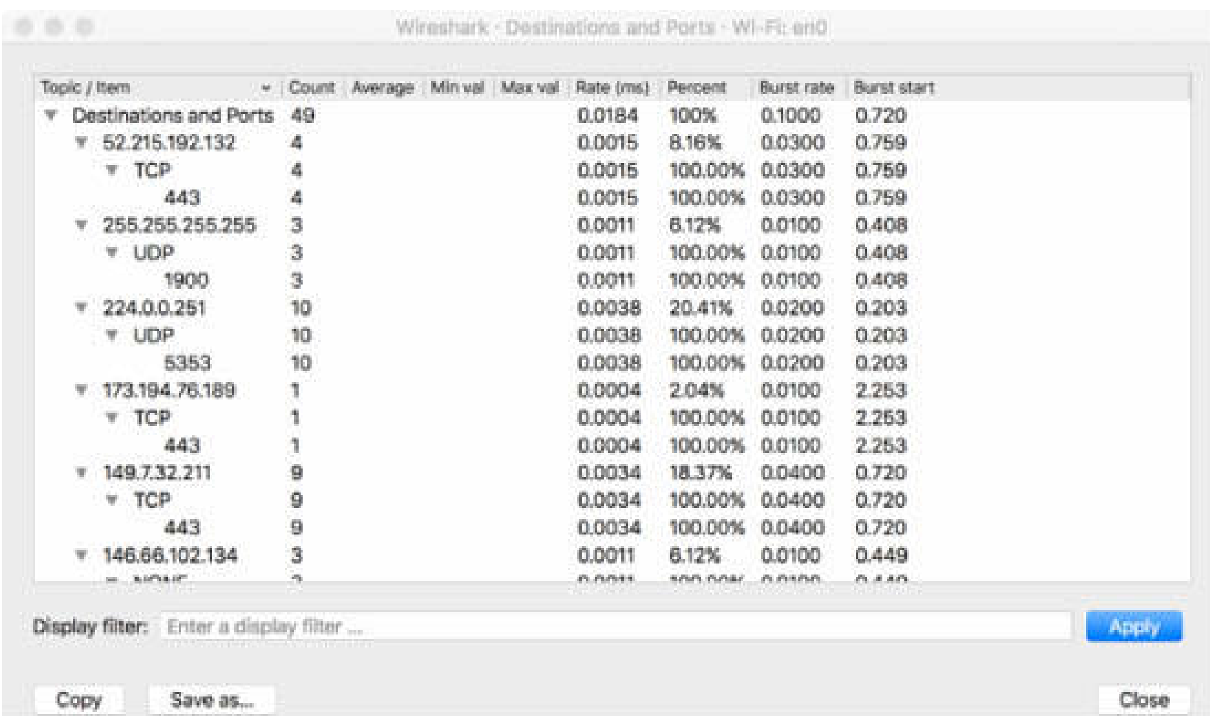

Wireshark · All Addresses · Wi-Fi: en0

Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst rate	Burst start
All Addresses	49				0.0184	100%	0.1000	0.720
52.215.192.132	7				0.0026	14.29%	0.0500	0.759
255.255.255.255	3				0.0011	6.12%	0.0100	0.408
224.0.0.251	10				0.0038	20.41%	0.0200	0.203
173.194.76.189	2				0.0008	4.08%	0.0200	2.253
149.7.32.211	18				0.0068	36.73%	0.0800	0.720
146.66.102.134	6				0.0023	12.24%	0.0200	0.449
10.4.9.190	10				0.0038	20.41%	0.0200	0.203
10.4.8.205	33				0.0124	67.35%	0.1000	0.720
10.4.15.98	1				0.0004	2.04%	0.0100	1.435
10.4.15.255	3				0.0011	6.12%	0.0100	0.408
10.4.15.204	1				0.0004	2.04%	0.0100	2.661
10.4.13.17	2				0.0008	4.08%	0.0100	0.306
10.4.11.205	1				0.0004	2.04%	0.0100	0.000
10.4.11.191	1				0.0004	2.04%	0.0100	0.615

Display filter: Enter a display filter ... Apply

Copy Save as... Close

When you select Destinations and Ports, the Destinations and Ports dialog box is displayed, showing the port used for each destination IPv4 address with associated count and percentage.

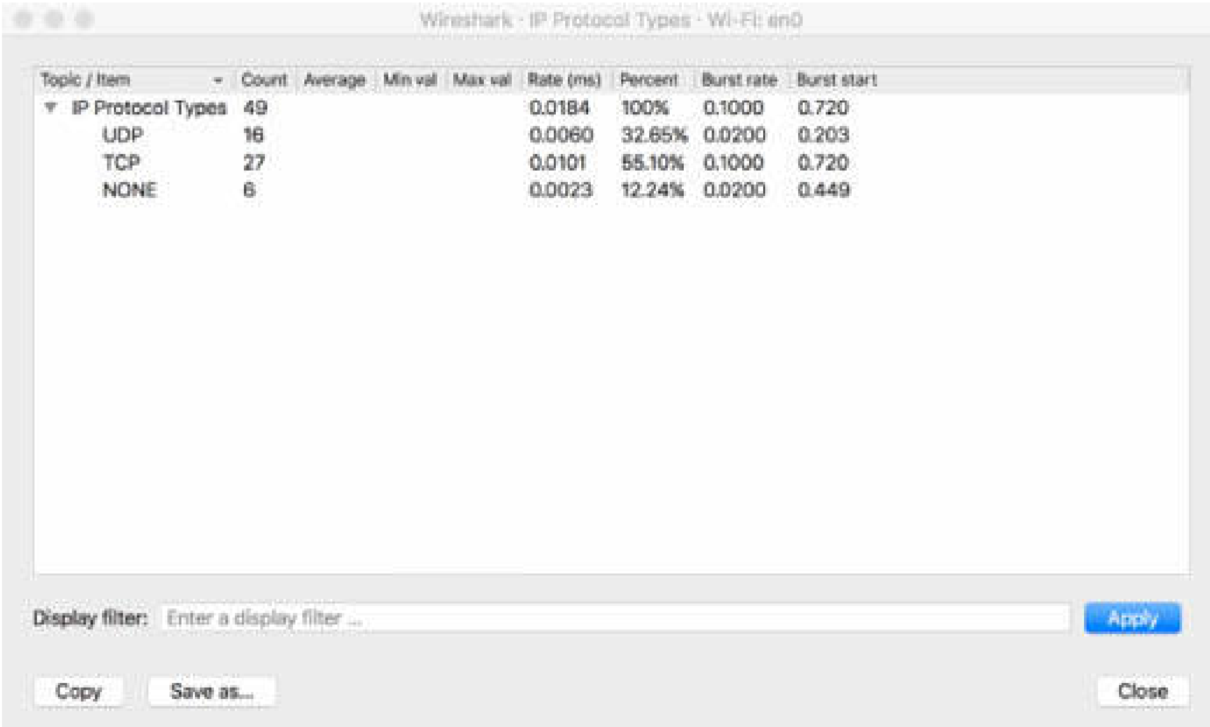

Wireshark · Destinations and Ports · Wi-Fi: en0

Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst rate	Burst start
Destinations and Ports	49				0.0184	100%	0.1000	0.720
52.215.192.132	4				0.0015	8.16%	0.0300	0.759
TCP	4				0.0015	100.00%	0.0300	0.759
443	4				0.0015	100.00%	0.0300	0.759
255.255.255.255	3				0.0011	6.12%	0.0100	0.408
UDP	3				0.0011	100.00%	0.0100	0.408
1900	3				0.0011	100.00%	0.0100	0.408
224.0.0.251	10				0.0038	20.41%	0.0200	0.203
UDP	10				0.0038	100.00%	0.0200	0.203
5353	10				0.0038	100.00%	0.0200	0.203
173.194.76.189	1				0.0004	2.04%	0.0100	2.253
TCP	1				0.0004	100.00%	0.0100	2.253
443	1				0.0004	100.00%	0.0100	2.253
149.7.32.211	9				0.0034	18.37%	0.0400	0.720
TCP	9				0.0034	100.00%	0.0400	0.720
443	9				0.0034	100.00%	0.0400	0.720
146.66.102.134	3				0.0011	6.12%	0.0100	0.449

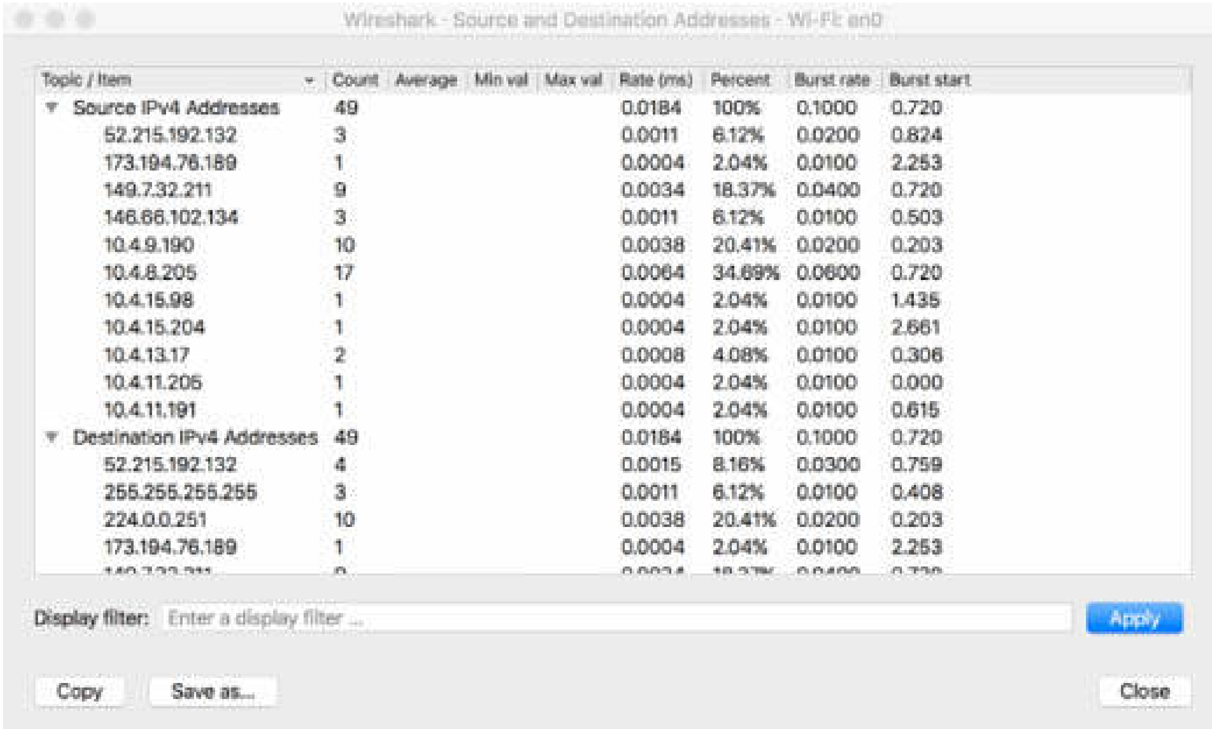
Display filter: Enter a display filter ... Apply

Copy Save as... Close

When you select IP Protocol Types, the IP Protocol Types dialog box is displayed showing the protocols types of the packets in the capture file.



When you select Source and Destination Addresses, the Source and Destination Addresses dialog box is displayed, showing the source and destination IPv4 addresses list for the packets in the capture file.



Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst rate	Burst start
Source IPv4 Addresses	49				0.0184	100%	0.1000	0.720
52.215.192.132	3				0.0011	6.12%	0.0200	0.824
173.194.76.189	1				0.0004	2.04%	0.0100	2.253
149.7.32.211	9				0.0034	18.37%	0.0400	0.720
146.66.102.134	3				0.0011	6.12%	0.0100	0.503
10.4.9.190	10				0.0038	20.41%	0.0200	0.203
10.4.8.205	17				0.0064	34.69%	0.0600	0.720
10.4.15.98	1				0.0004	2.04%	0.0100	1.435
10.4.15.204	1				0.0004	2.04%	0.0100	2.661
10.4.13.17	2				0.0008	4.08%	0.0100	0.306
10.4.11.205	1				0.0004	2.04%	0.0100	0.000
10.4.11.191	1				0.0004	2.04%	0.0100	0.615
Destination IPv4 Addresses	49				0.0184	100%	0.1000	0.720
52.215.192.132	4				0.0015	8.16%	0.0300	0.759
255.255.255.255	3				0.0011	6.12%	0.0100	0.408
224.0.0.251	10				0.0038	20.41%	0.0200	0.203
173.194.76.189	1				0.0004	2.04%	0.0100	2.253

Notes:
To gain more confidence in using the statistics information, ping a different IP location. Observe the change in the information related to different protocol types and addresses or ports used.

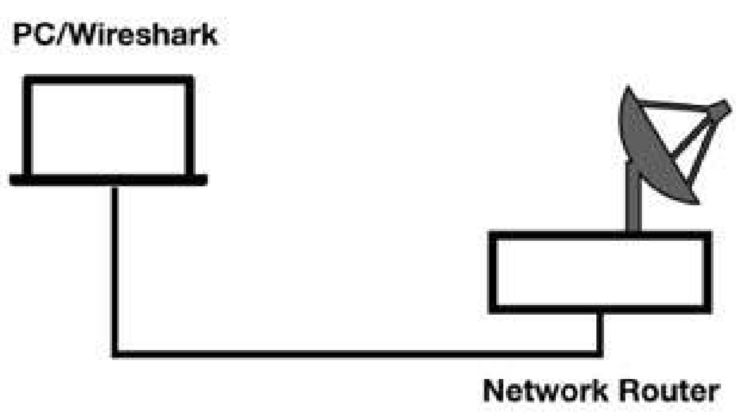
Lab 20. Display Filters—Basics

Lab Objective:
Learn how to build and apply basic display filters by protocol type or addresses.

Lab Purpose:
Wireshark allows you to create display filters based on the packet type and IP/MAC addresses. Each display filter should be created based on the most relevant feature of the captured packet and the easiest way to create it.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

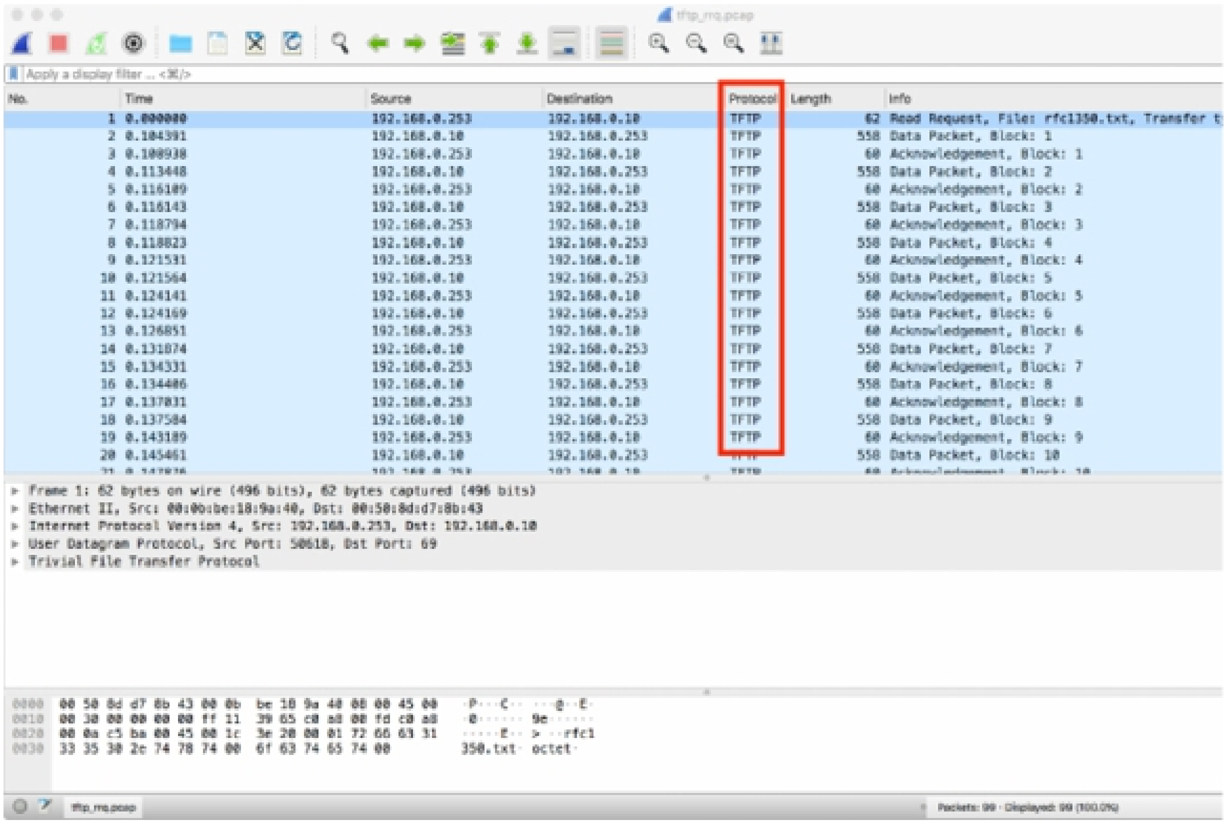
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

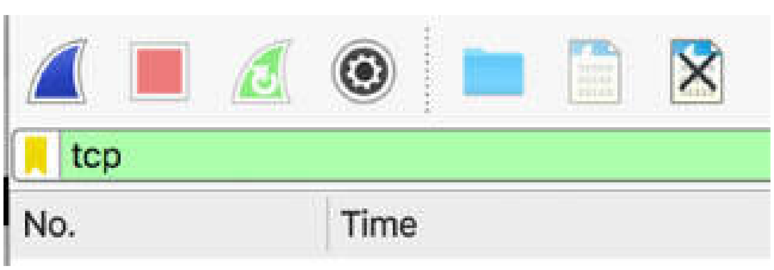
Task 1:

Download the free sample capture file tftp_rrq.pcap to your PC from **https://wiki.wireshark.org/SampleCaptures** , and then open the downloaded file in Wireshark. In the Packet List pane, the protocol type is displayed in the Protocol column containing the value TFTP.

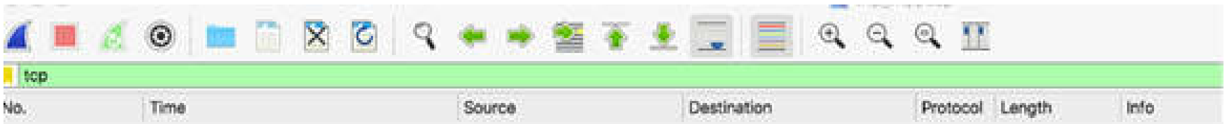


Task 2:

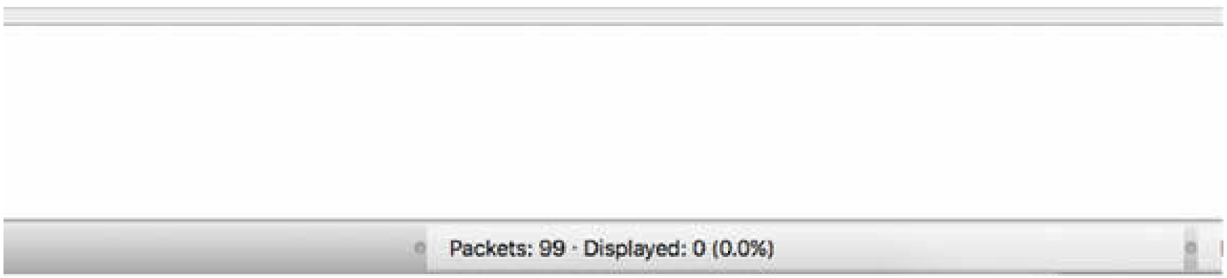
In the filter toolbar, enter a different protocol such as tcp and press the Return key. If the syntax is correct, the background becomes green, as shown in the following figure.



If there are no TCP packets in the opened capture log file, the Packet List pane will be empty, as shown in the figure below.

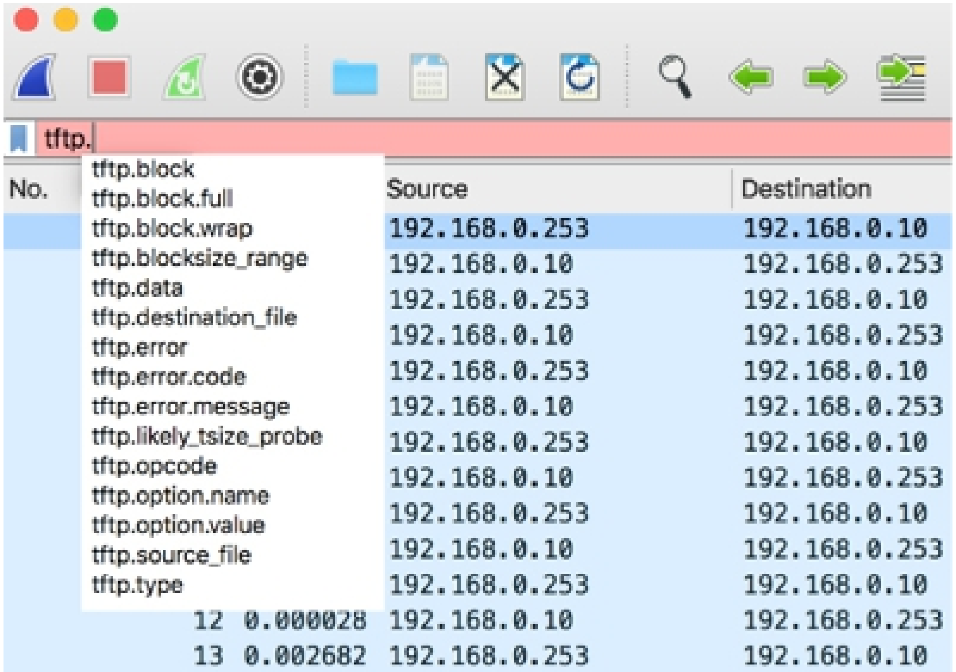


At the right bottom of the main window, the number of displayed packets that satisfy the filter condition is indicated. In this example, it is shown as 0 out of 99 packets.



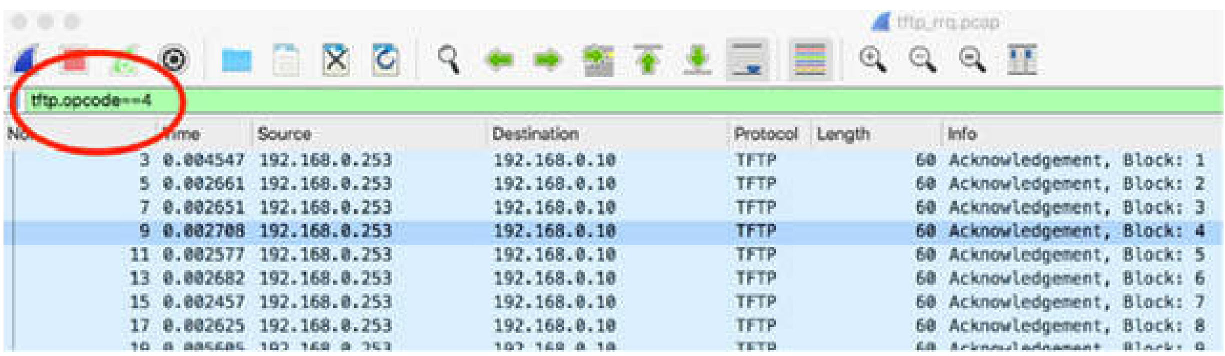
Task 3:
In the filter toolbar, enter a different protocol such as tftp and press the Return key. All 99 packets are displayed.

Type a period after tftp (such as tftp.), and the complete list of available filters starting with tftp is displayed in the popup menu, as shown in the figure below.

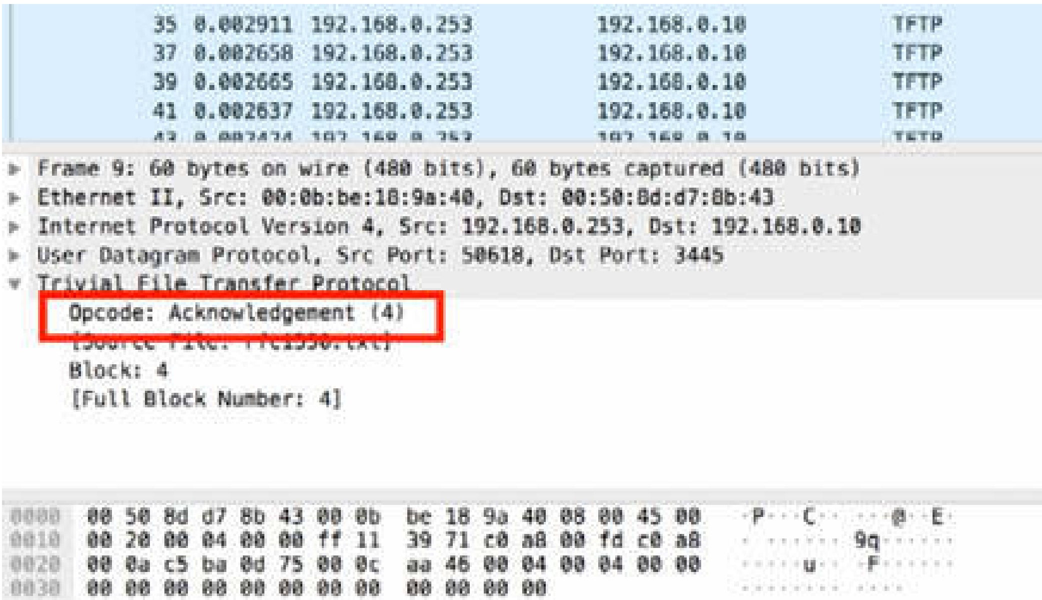


Task 4:

In the filter toolbar, enter tftp.opcode == 4 and press the Return key.



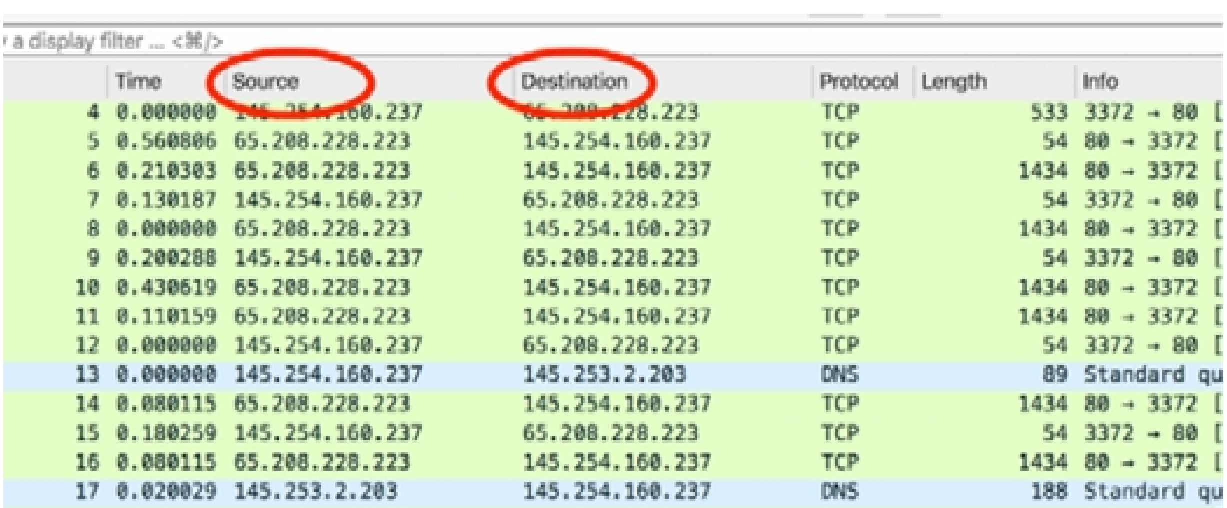
All packets of type Acknowledgement are filtered, as displayed in the Info column. The Acknowledgment value is also visible in the Packet Details pane, as shown in the figure below.



Task 5:

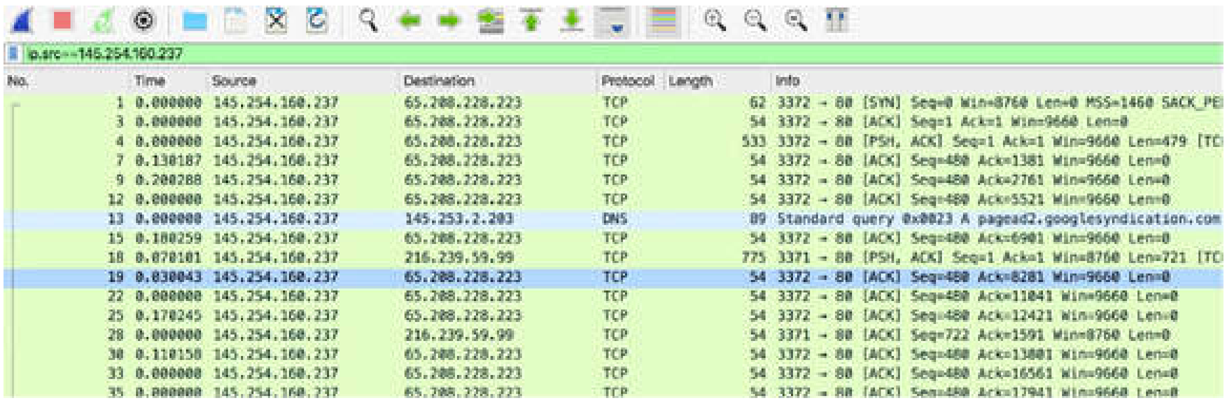
You can also filter based on the IP addresses. Download the free sample capture file http.cap to your PC from https://wiki.wireshark.org/SampleCaptures , and then open the downloaded file in Wireshark.

In the Packet List pane, the Source and the Destination columns, shown in the figure below, correspond to the source IP address and the destination IP address for each packet.



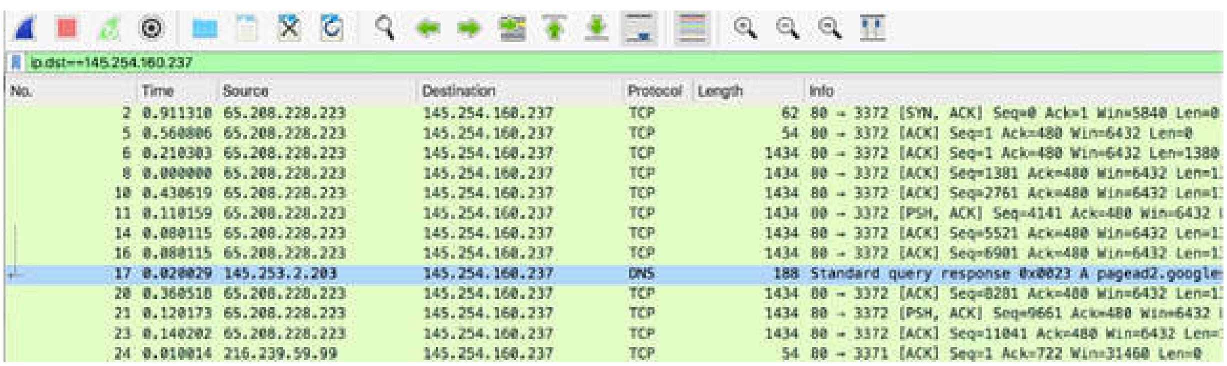
Task 6:
Note down the source address, such as 145.254.160.237. In the filter toolbar, enter ip.src==145.254.160.237 and press the Return key.

In the Packet List pane, only packets with source IP address 145.254.160.237 are displayed.



You can also create a filter using the destination IP address. In the filter toolbar, enter ip.dst ==145.254.160.237 and press the Return key.

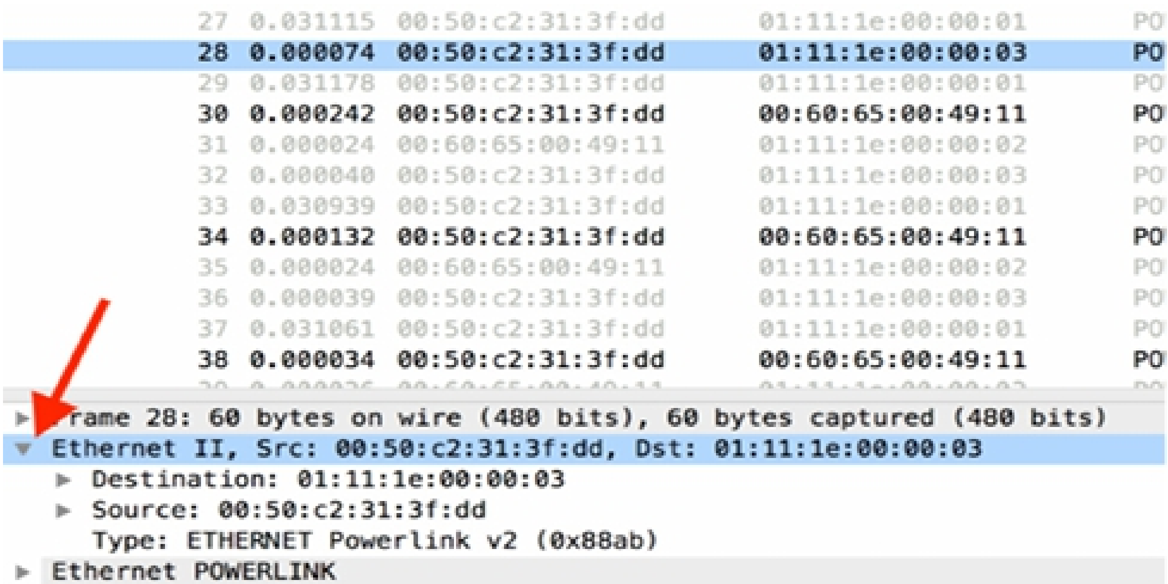
In the Packet List pane, only packets with destination IP address 145.254.160.237 are displayed.



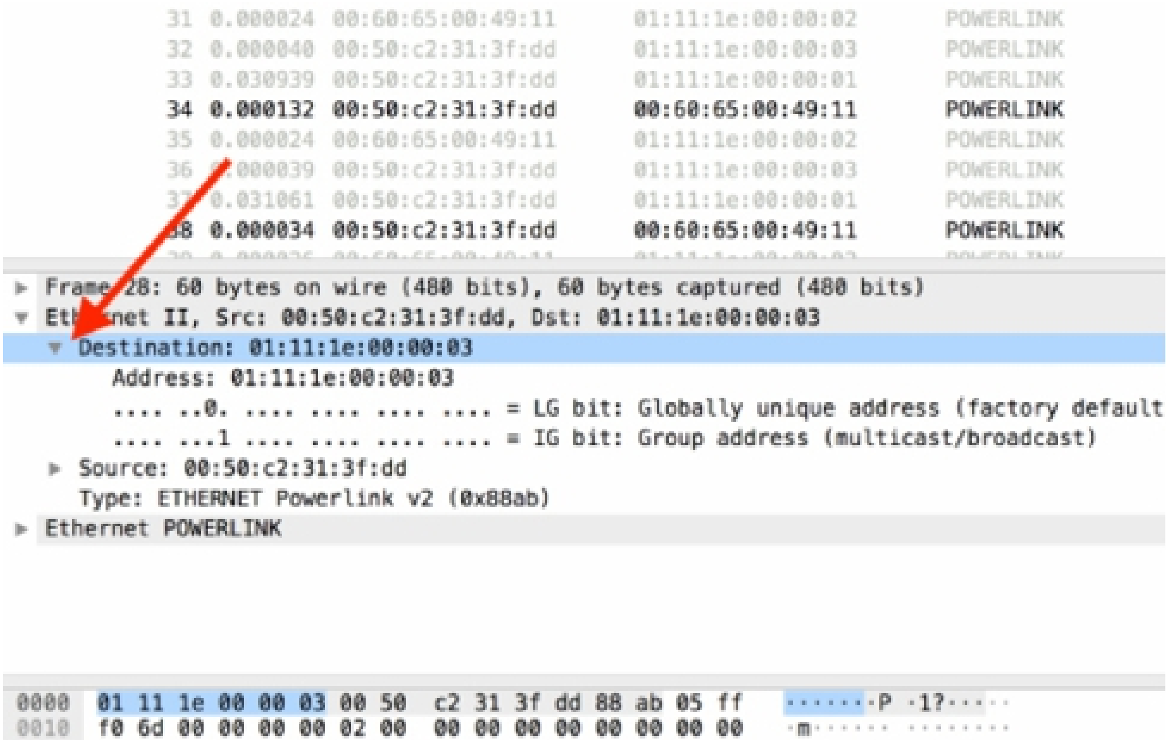
Task 7:
You can also create a display filter by using the MAC address. Download the free sample capture epl.cap.gz to your PC from

https://wiki.wireshark.org/SampleCaptures . Unpack it and then open the file in Wireshark.

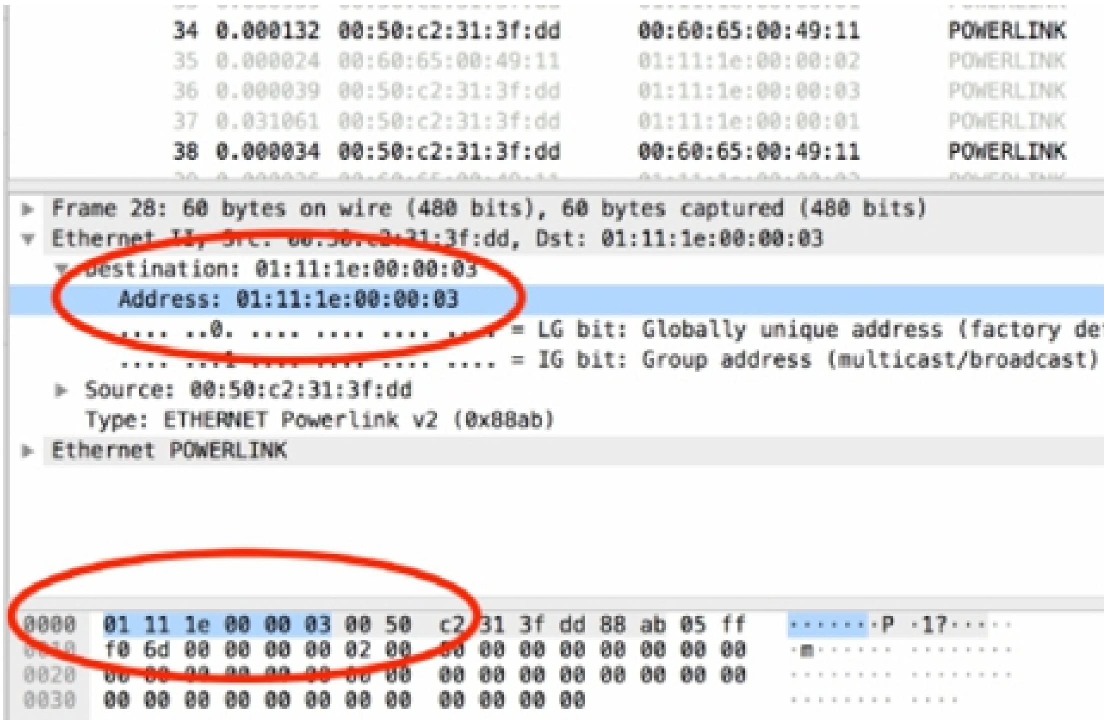
In the Packet List pane, select a packet such as packet #28. In the Packet Details pane, right-click Ethernet II to open the tree view, as shown in the figure below.



Click the arrow before Destination.

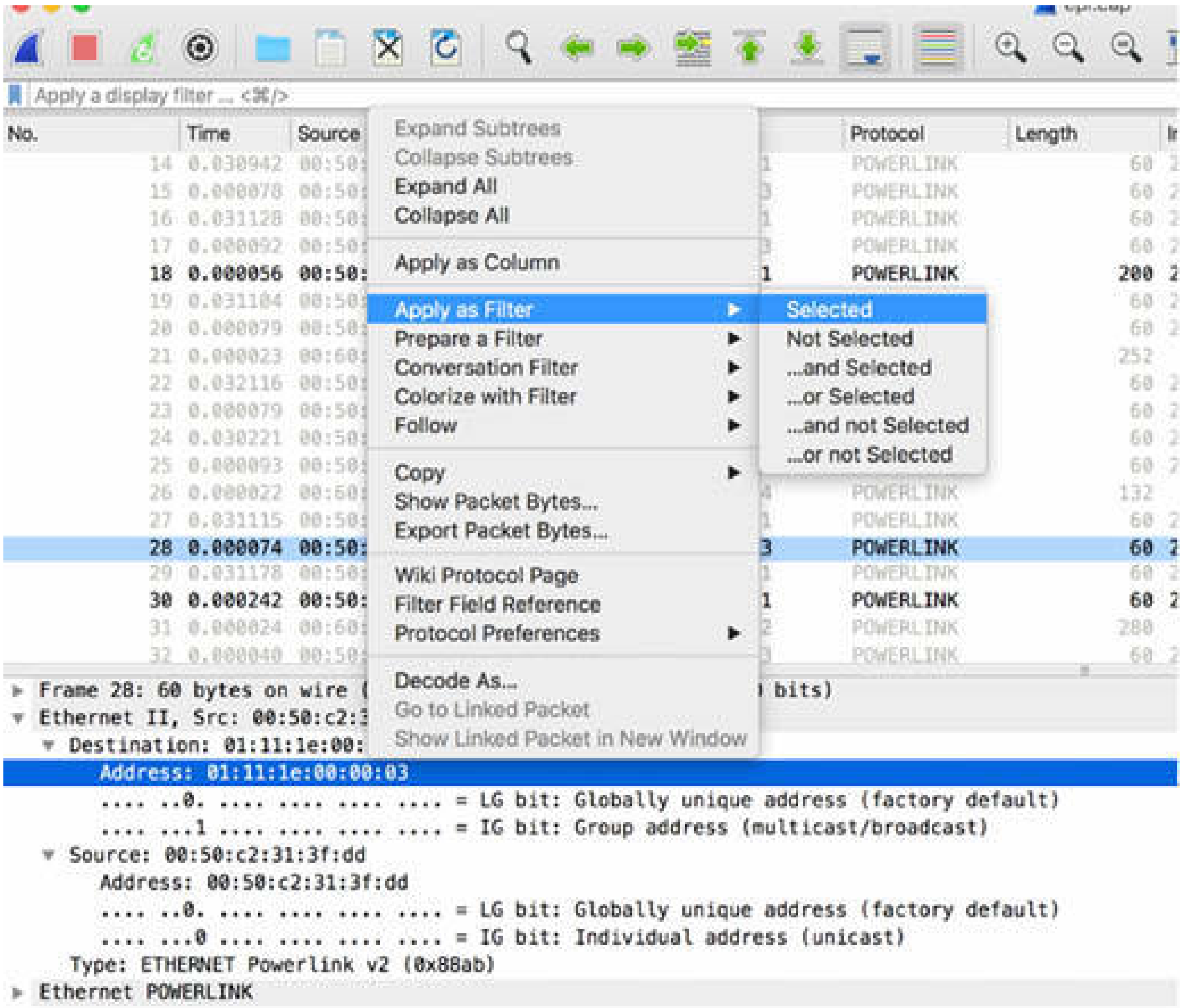


Select Address as shown in the figure below. This is the Destination MAC Address. This selection is replicated in the Packet Bytes pane. Therefore, you can observe the destination MAC address location inside a byte stream.

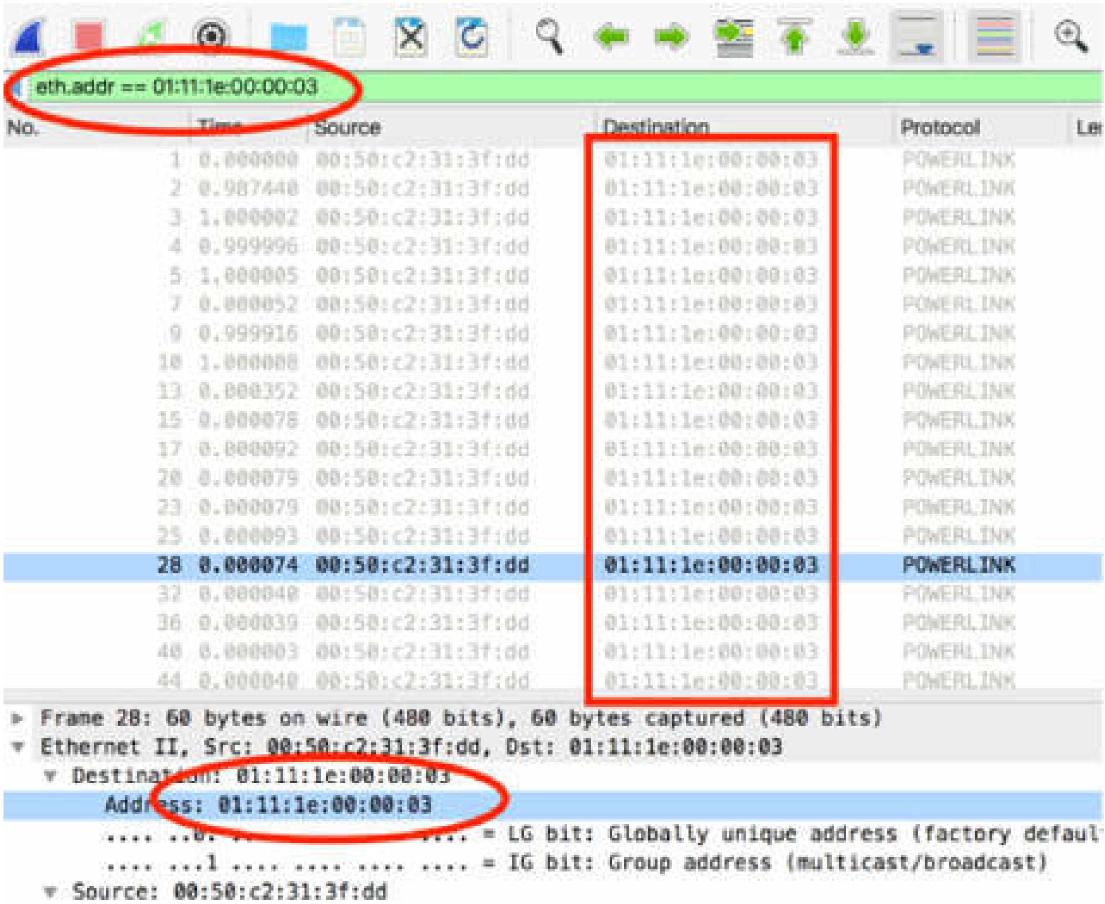


Task 8:

Right-click Address, and in the pop-up menu, select Apply as Filter > Selected, as shown in the figure below.



The filter based on the field selected is applied, and only those packets that satisfy the filter condition are displayed. The applied filter is also visible in the filter toolbar. You can verify that the value in the toolbar matches the value in the Destination field.



Notes:

You can also use the information provided in this step to filter the source IP or destination IP addresses after selecting the source IP or destination IP addresses in the Packet List or Packet Details pane.

To gain more confidence in using the filters, create more packet display filters to display the desired network packets from the sample capture files. Follow the instructions given in this lab for IP addresses and MAC addresses.

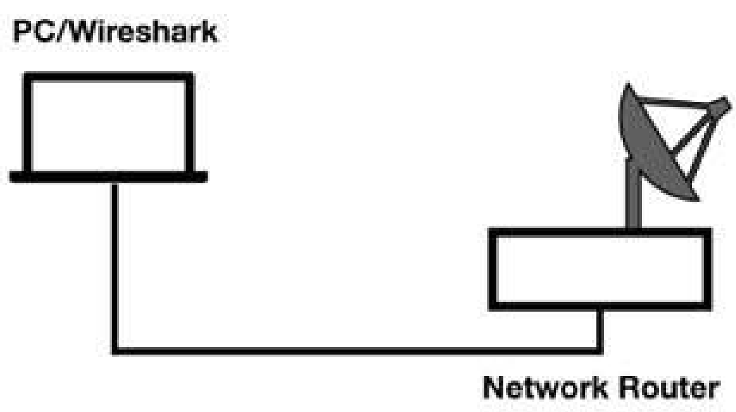
Lab 21. Display Filters—Saved Filters and Right-Click Creation

Lab Objective:
Learn how to save and manage filters by using different Wireshark options.

Lab Purpose:
Wireshark provides you the option to create or save ready-to-use filters and manage them. You can create and manage filters buttons through the main window (right-click creation) or the Preferences settings.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

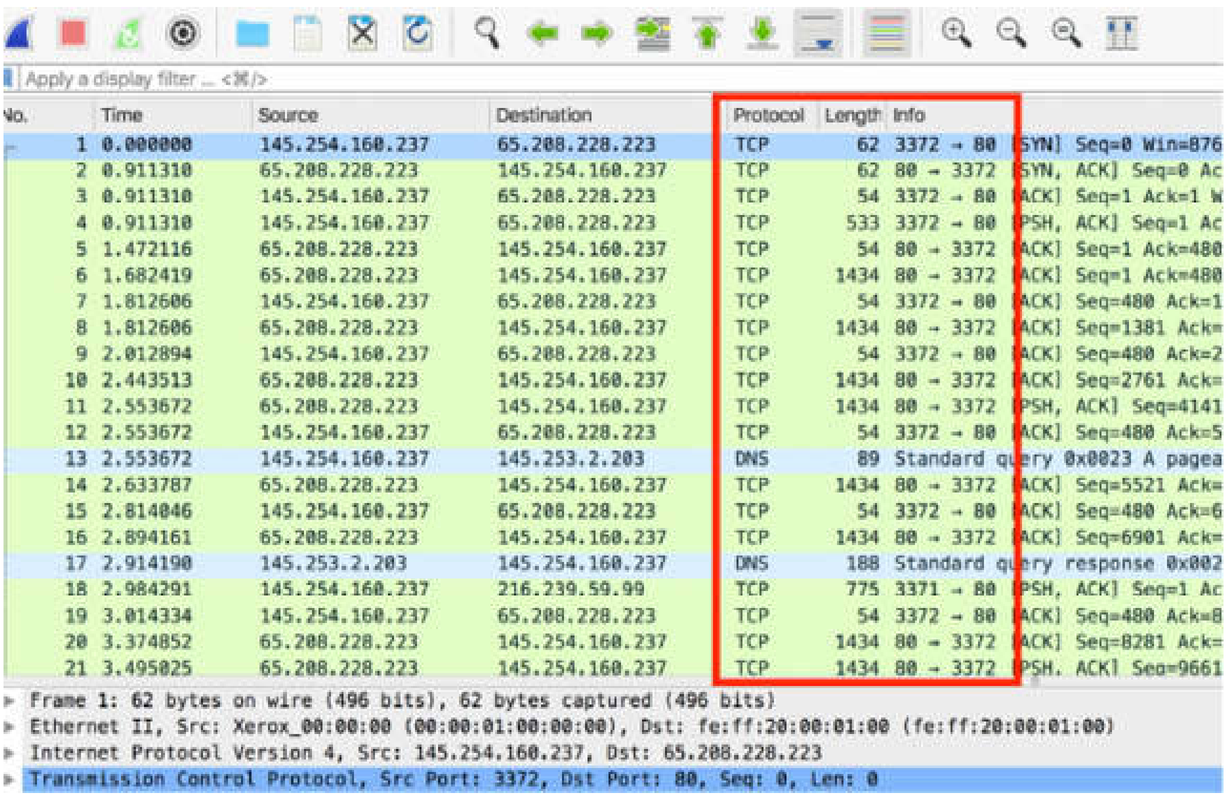
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



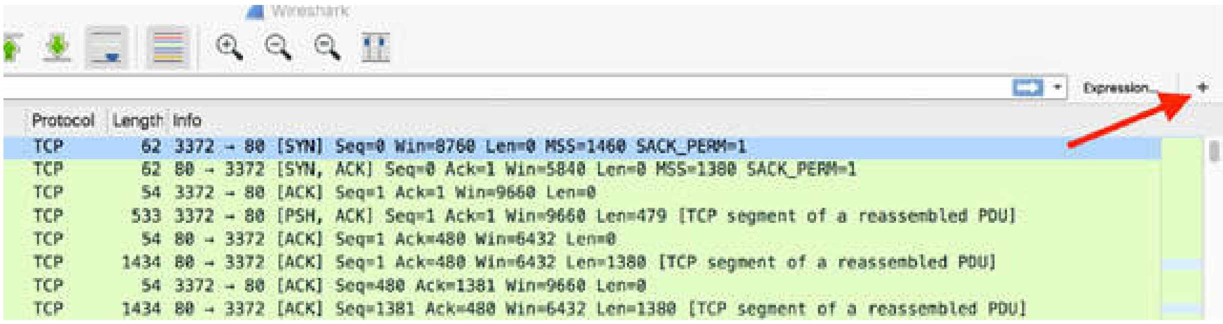
Lab Walkthrough:

Task 1:
Download the free sample capture file http.cap to your PC from **https://wiki.wireshark.org/SampleCaptures** , and then open the downloaded file in Wireshark.

As shown in the figure below, for protocols with different range of source and destination ports, all packets are all marked as TCP.

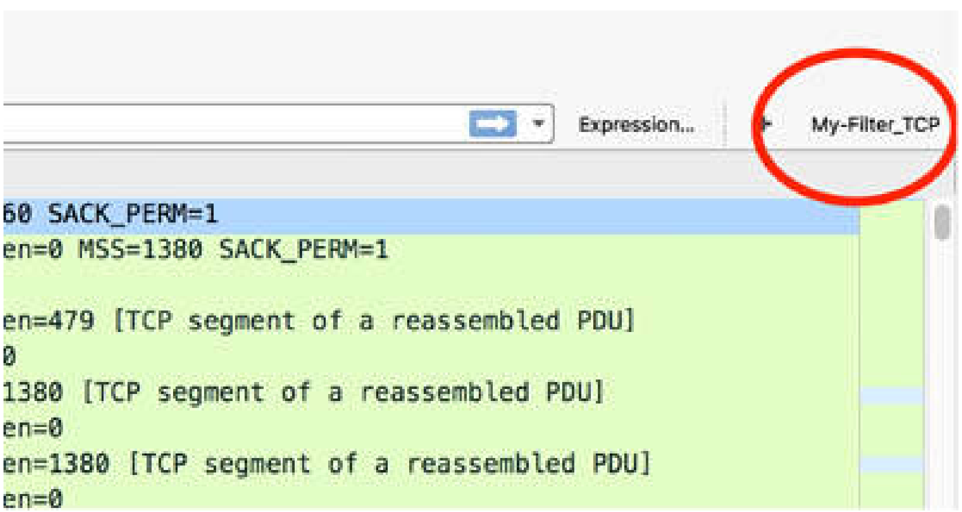


Task 2:
To have a ready-to-use filter for frequent use, you can save a filter. In the main Wireshark window, click the add button (+) on the right side of the filter toolbar, as shown in the figure below.



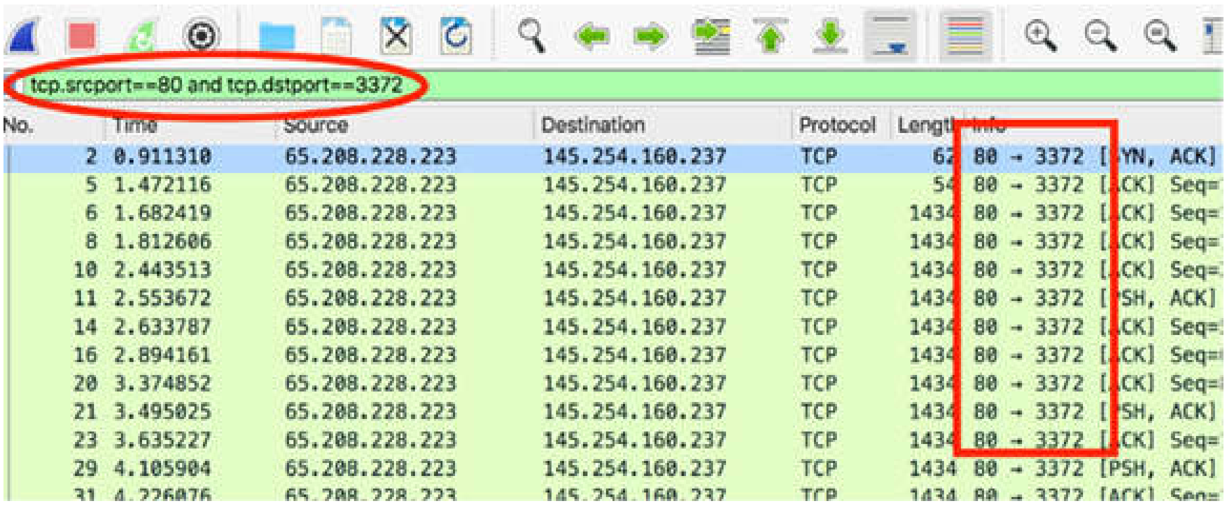
As shown in the figure below, multiple boxes are displayed under the filter toolbar, providing options for specifying the filter syntax, label, and comments. For example, to frequently filter TCP frames with source port 80 and destination port 3372, you can provide the information shown in the figure below.

When you click OK, the filter button is displayed on the right side of the filter toolbar.

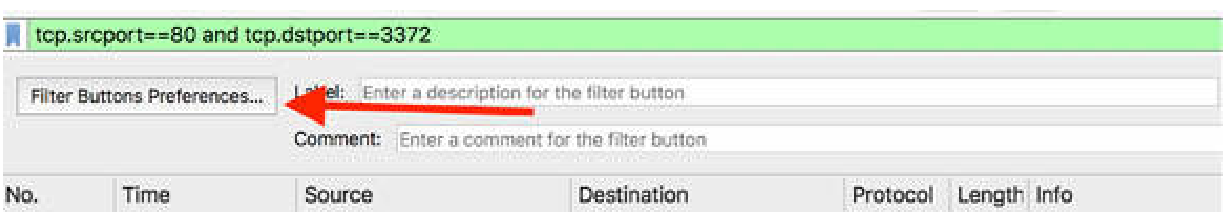


Task 3:

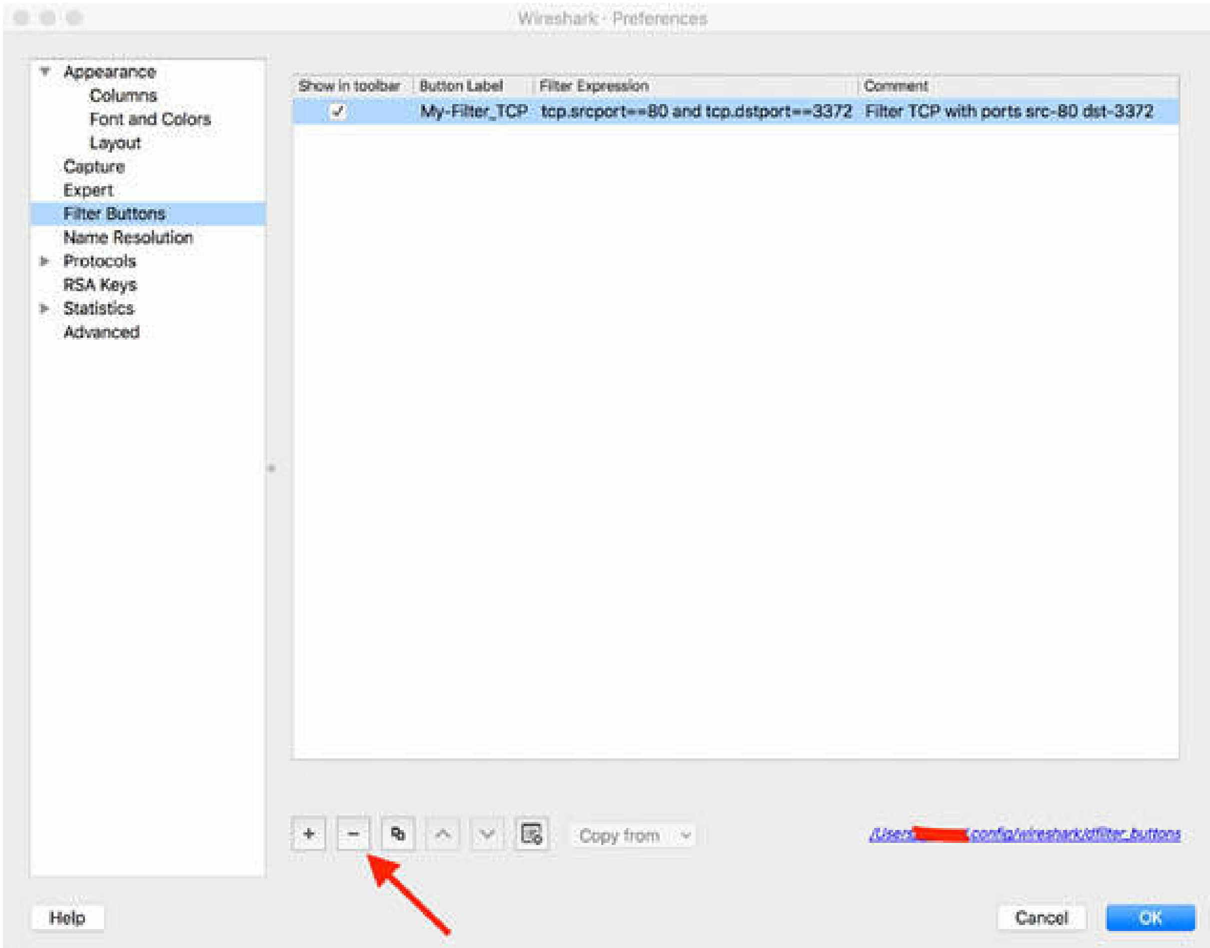
Click the just-created filter button. The filter is applied on the packets list, and the filter syntax is displayed in the filter toolbar.



Task 4:
You can add more filter buttons by following the procedure described in the previous step. To remove a button, click again the add (+) button and then click the “Filter Buttons Preferences” as shown in the figure below.

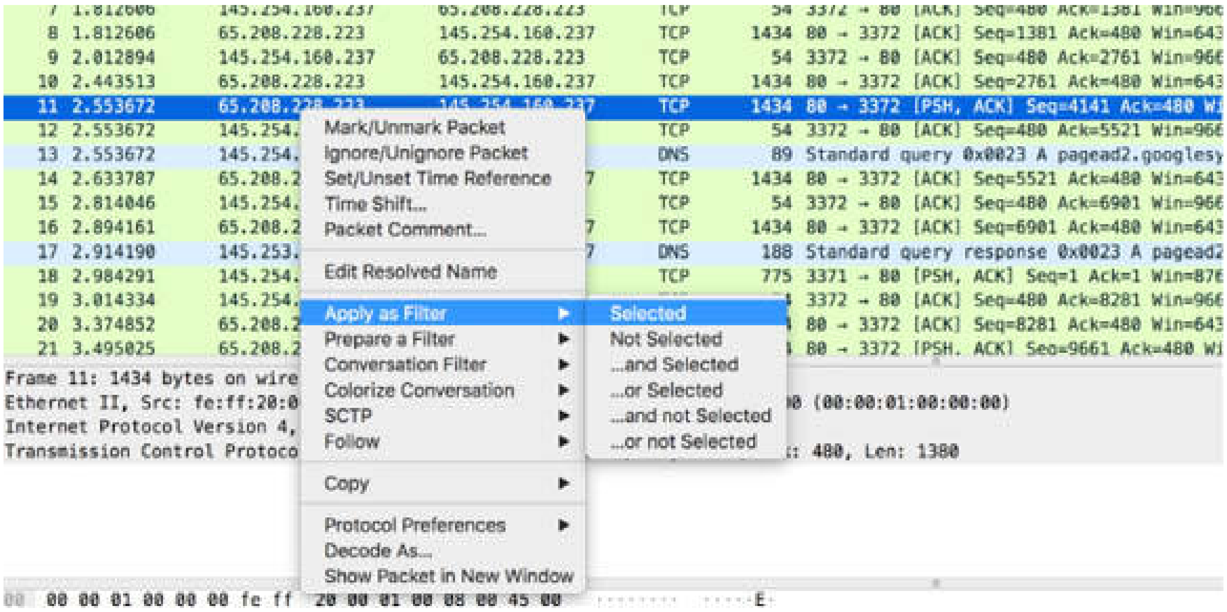


The Preferences dialog box is displayed showing all saved filters. You can add a filter button by clicking the add (+) button or delete a filter button by clicking the remove (–) button, shown in the figure below.

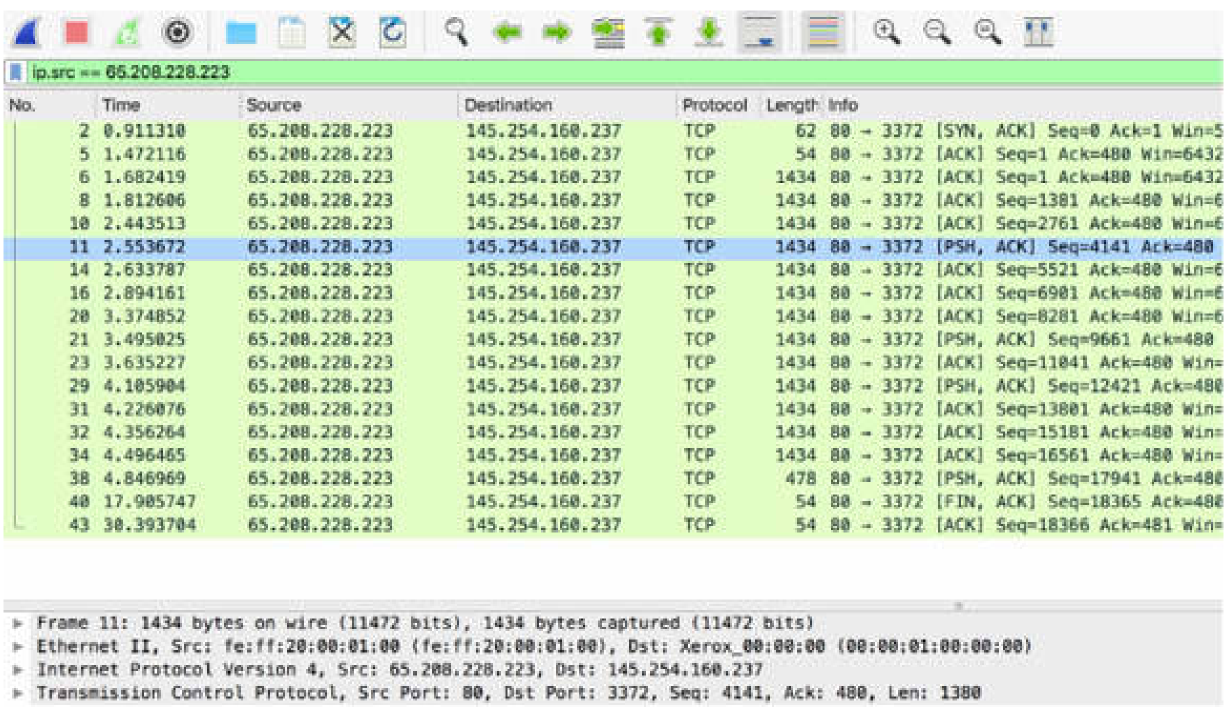


Task 5:

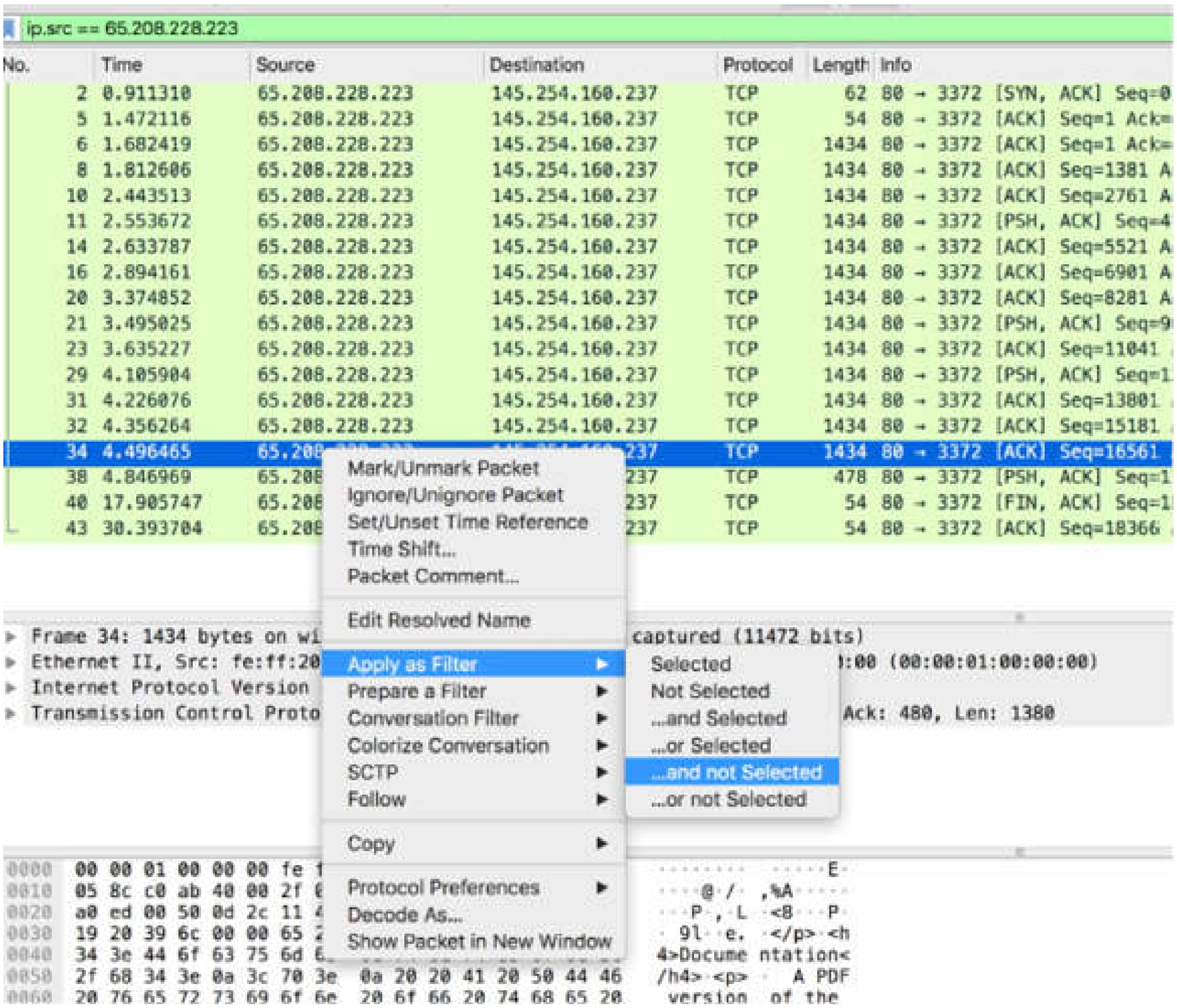
You can also create a filter by acquiring information from a specified packet in the Packet List pane. For example, if you are interested in packets similar to packet #11, right-click on it, and then select Apply as filter > Selected as shown in the figure below.



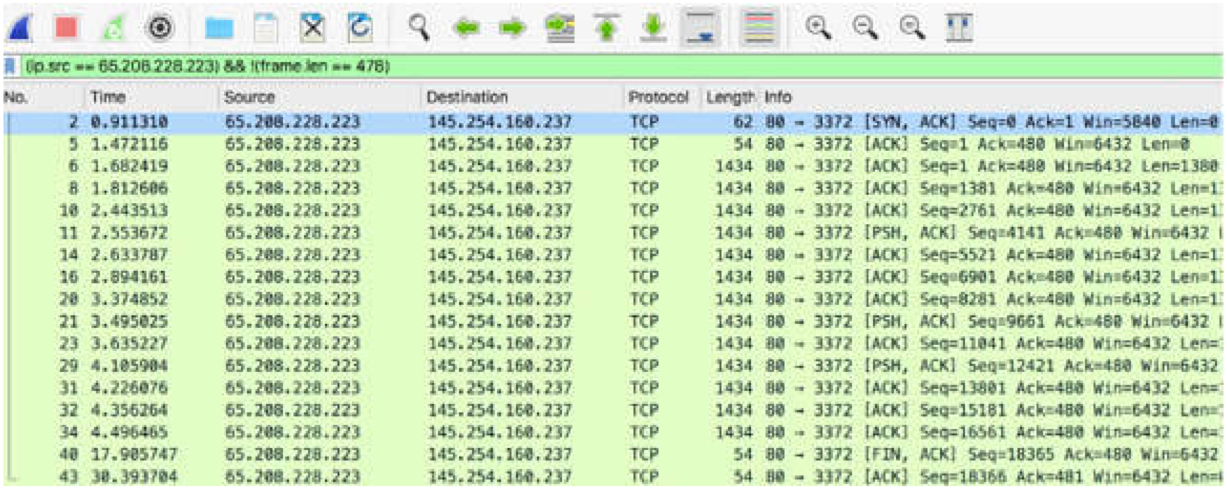
As a result, the filter (related to the source IP address) based on the features of packet #11 is created, displayed (in the filter toolbar), and applied, as shown in the figure below.



In the same way, you can create a combination of filters or apply a filter containing a feature NOT present in the selected packet, as shown in the figure below.



As a result, the selected packet is excluded by the filter (a filter based on the packet length has been applied), as shown in the figure below.



No.	Time	Source	Destination	Protocol	Length	Info
2	0.911310	65.208.228.223	145.254.160.237	TCP	62	80 → 3372 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0
5	1.472116	65.208.228.223	145.254.160.237	TCP	54	80 → 3372 [ACK] Seq=1 Ack=480 Win=6432 Len=0
6	1.682419	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=1 Ack=480 Win=6432 Len=1380
8	1.812606	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=1381 Ack=480 Win=6432 Len=1
10	2.443513	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=2761 Ack=480 Win=6432 Len=1
11	2.553672	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [PSH, ACK] Seq=4141 Ack=480 Win=6432
14	2.633787	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=5521 Ack=480 Win=6432 Len=1
16	2.894161	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=6901 Ack=480 Win=6432 Len=1
20	3.374852	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=8281 Ack=480 Win=6432 Len=1
21	3.495025	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [PSH, ACK] Seq=9661 Ack=480 Win=6432
23	3.635227	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=11041 Ack=480 Win=6432 Len=
29	4.105904	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [PSH, ACK] Seq=12421 Ack=480 Win=6432
31	4.226076	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=13801 Ack=480 Win=6432 Len=
32	4.356264	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=15181 Ack=480 Win=6432 Len=
34	4.496465	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=16561 Ack=480 Win=6432 Len=
40	17.905747	65.208.228.223	145.254.160.237	TCP	54	80 → 3372 [FIN, ACK] Seq=18365 Ack=480 Win=6432
43	30.393704	65.208.228.223	145.254.160.237	TCP	54	80 → 3372 [ACK] Seq=18366 Ack=481 Win=6432 Len=

Notes:

A wide range of packet display filter can be used to display the desired network packets from the file. Repeat the above instructions to create several custom filter buttons. You can add either from the Preferences dialog box or directly from the main window by using the add (+) button. Try also to create a combination of filters by right-clicking the selected packets.

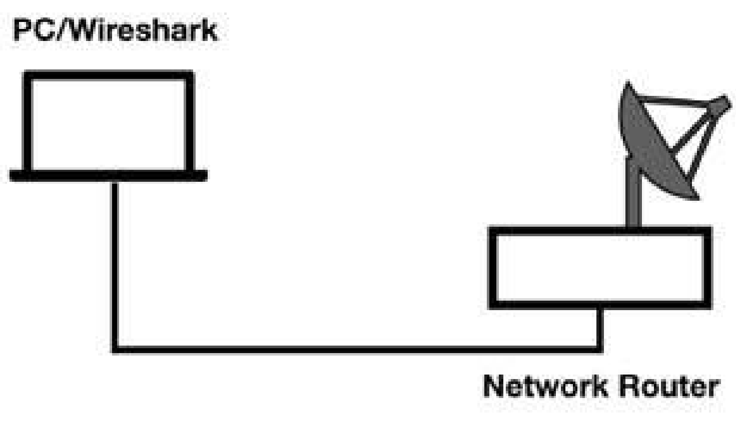
Lab 22. Display Filters—Conversations Endpoint—Comparison Operators

Lab Objective:
Learn how to filter conversations or endpoints and how to use comparison operators.

Lab Purpose:
Understand how to apply display filters in the conversations or endpoints window and how to create complex filters by combining multiple comparison operators.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

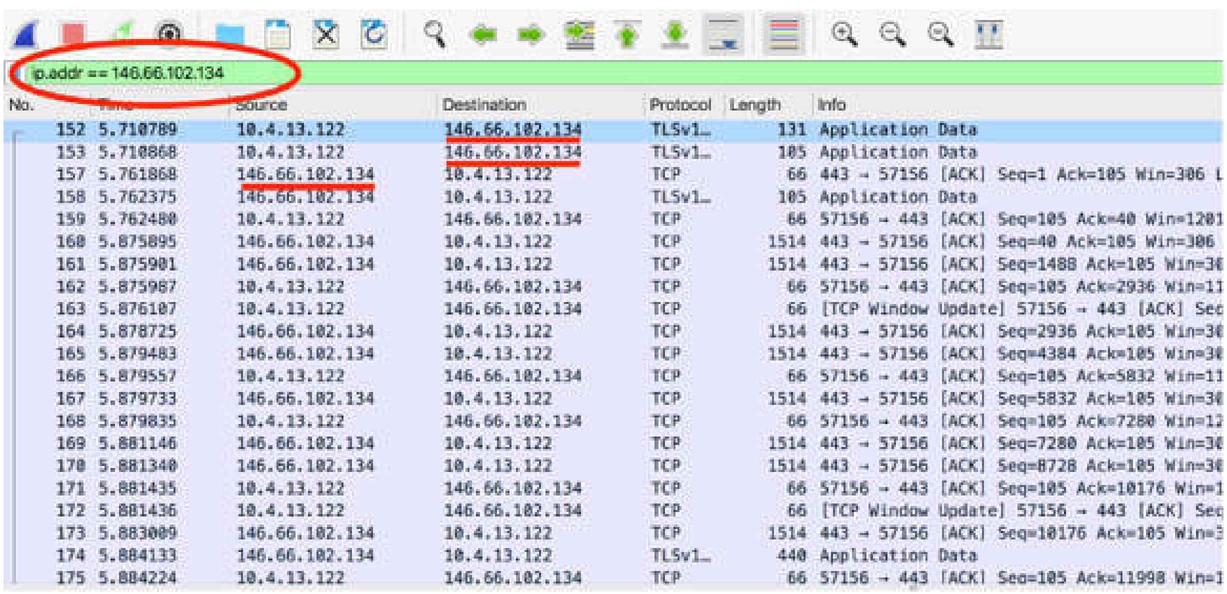


Lab Walkthrough:

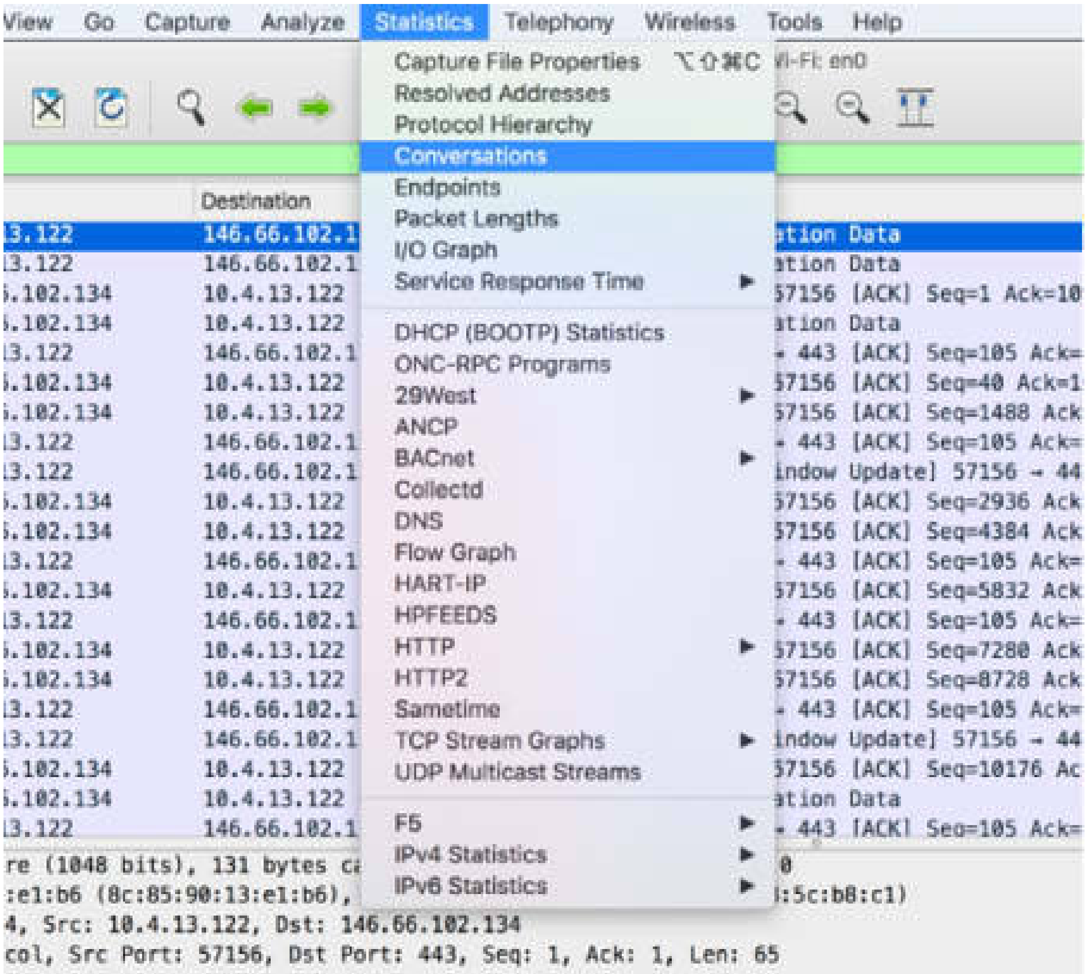
Task 1:
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column, and capture the traffic for a few minutes.

In a web browser, go to **https://www.101labs.net** and navigate through the website. Stop the capture and save the file.

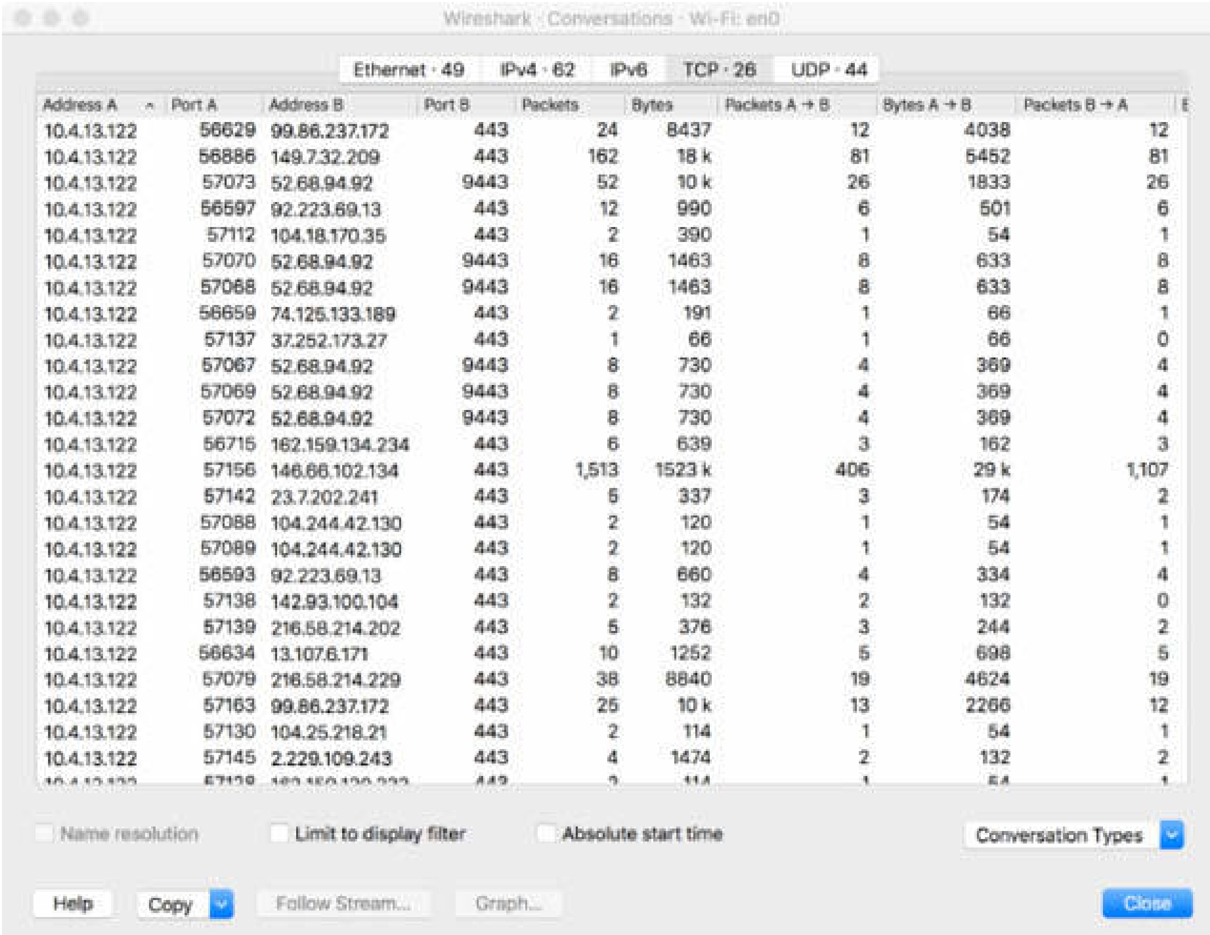
Task 2:
In the filter toolbar, enter ip.addr == 146.66.102.134 . In the Packet List pane, only traffic directed to or from **https://www.101labs.net** is displayed (146.66.102.134 is the public IP address of the website).



Task 3:
On the main menu, select Statistics > Conversations, as shown in the figure below.

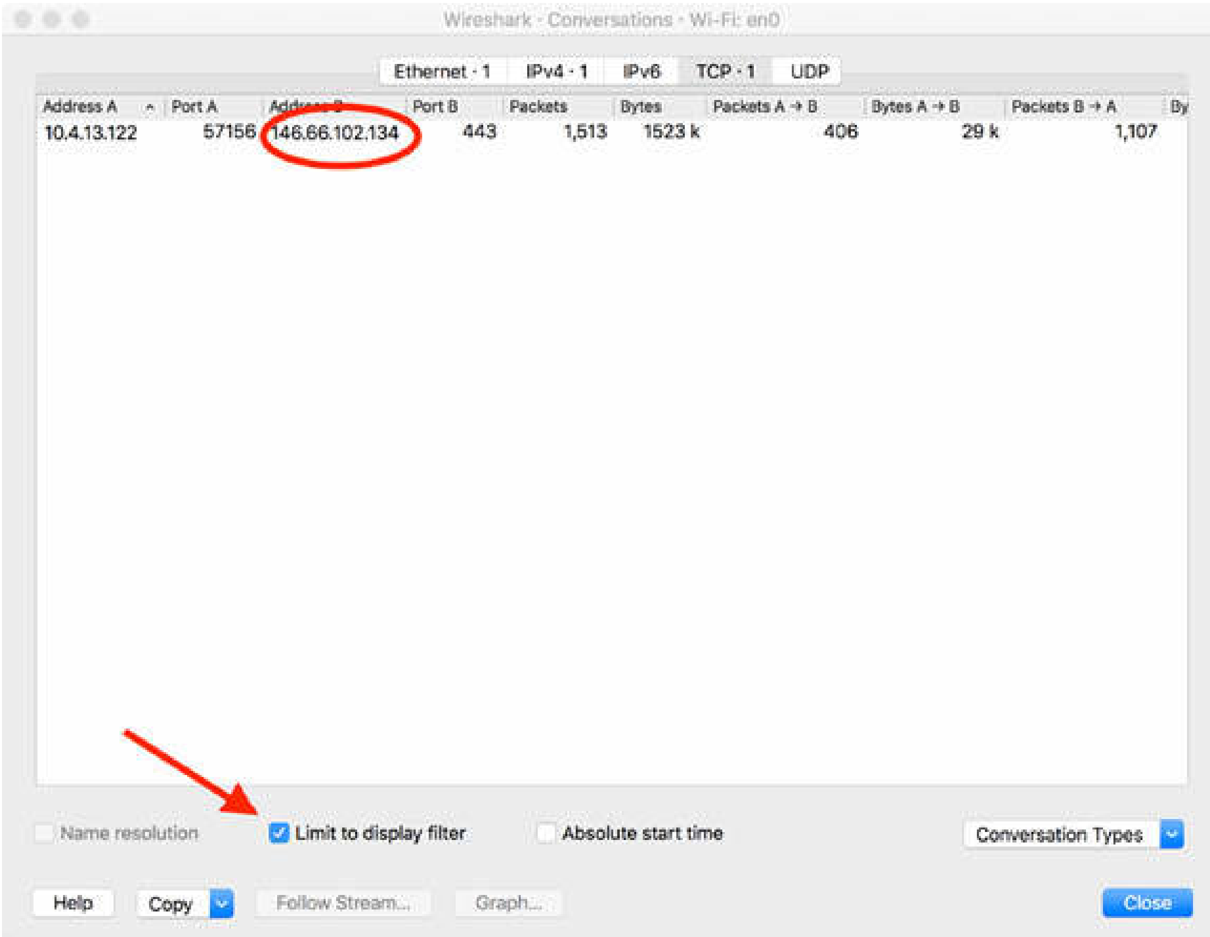


The Conversations dialog box is displayed, showing all conversations contained in the capture.

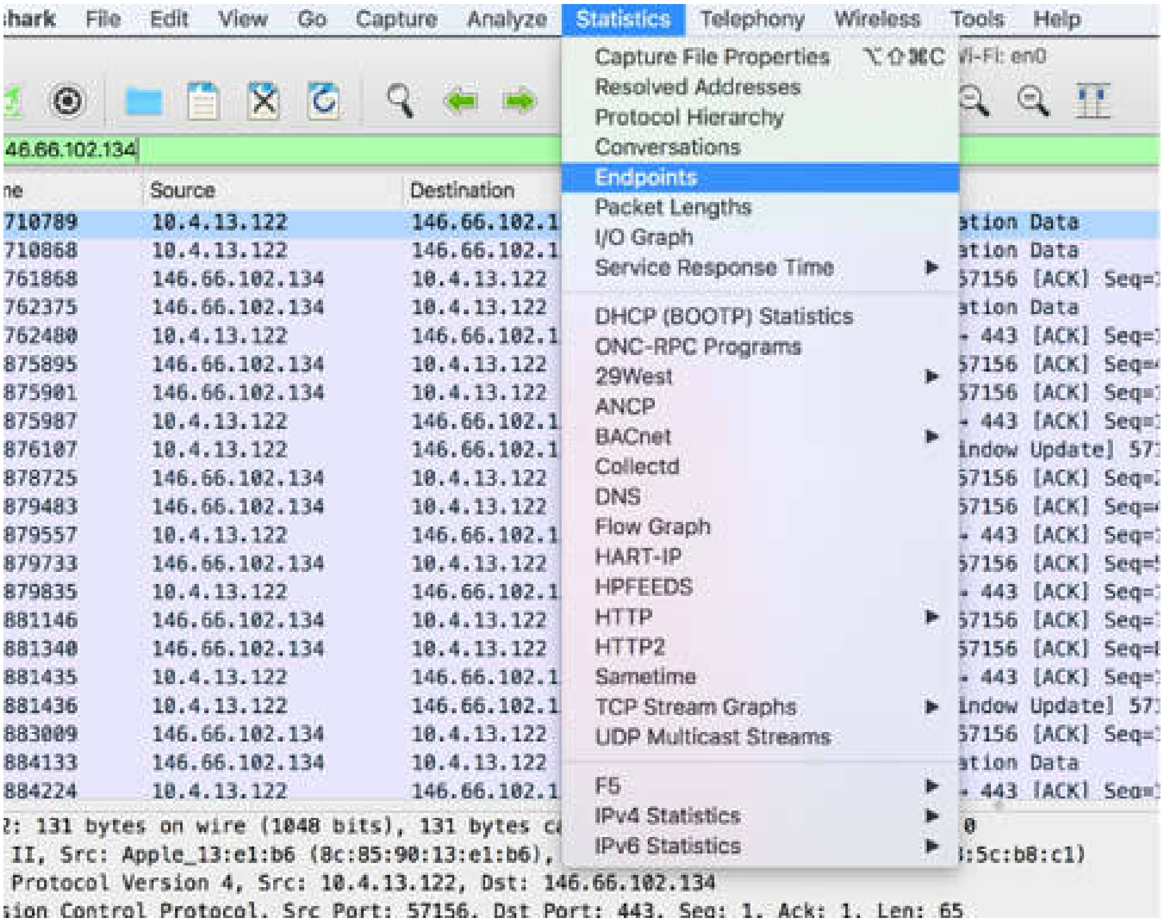


Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A
10.4.13.122	56629	99.86.237.172	443	24	8437	12	4038	12
10.4.13.122	56886	149.7.32.209	443	162	18 k	81	5452	81
10.4.13.122	57073	52.68.94.92	9443	52	10 k	26	1833	26
10.4.13.122	56597	92.223.69.13	443	12	990	6	501	6
10.4.13.122	57112	104.18.170.35	443	2	390	1	54	1
10.4.13.122	57070	52.68.94.92	9443	16	1463	8	633	8
10.4.13.122	57068	52.68.94.92	9443	16	1463	8	633	8
10.4.13.122	56659	74.125.133.189	443	2	191	1	66	1
10.4.13.122	57137	37.252.173.27	443	1	66	1	66	0
10.4.13.122	57067	52.68.94.92	9443	8	730	4	369	4
10.4.13.122	57069	52.68.94.92	9443	8	730	4	369	4
10.4.13.122	57072	52.68.94.92	9443	8	730	4	369	4
10.4.13.122	56715	162.159.134.234	443	6	639	3	162	3
10.4.13.122	57156	146.66.102.134	443	1,513	1523 k	406	29 k	1,107
10.4.13.122	57142	23.7.202.241	443	5	337	3	174	2
10.4.13.122	57088	104.244.42.130	443	2	120	1	54	1
10.4.13.122	57089	104.244.42.130	443	2	120	1	54	1
10.4.13.122	56593	92.223.69.13	443	8	660	4	334	4
10.4.13.122	57138	142.93.100.104	443	2	132	2	132	0
10.4.13.122	57139	216.58.214.202	443	5	376	3	244	2
10.4.13.122	56634	13.107.6.171	443	10	1252	5	698	5
10.4.13.122	57079	216.58.214.229	443	38	8840	19	4624	19
10.4.13.122	57163	99.86.237.172	443	25	10 k	13	2266	12
10.4.13.122	57130	104.25.218.21	443	2	114	1	54	1
10.4.13.122	57145	2.229.109.243	443	4	1474	2	132	2

Select the "Limit to display" filter check box to show only those conversations that are related to the filtered packets. The conversations shown in the figure below are displayed.

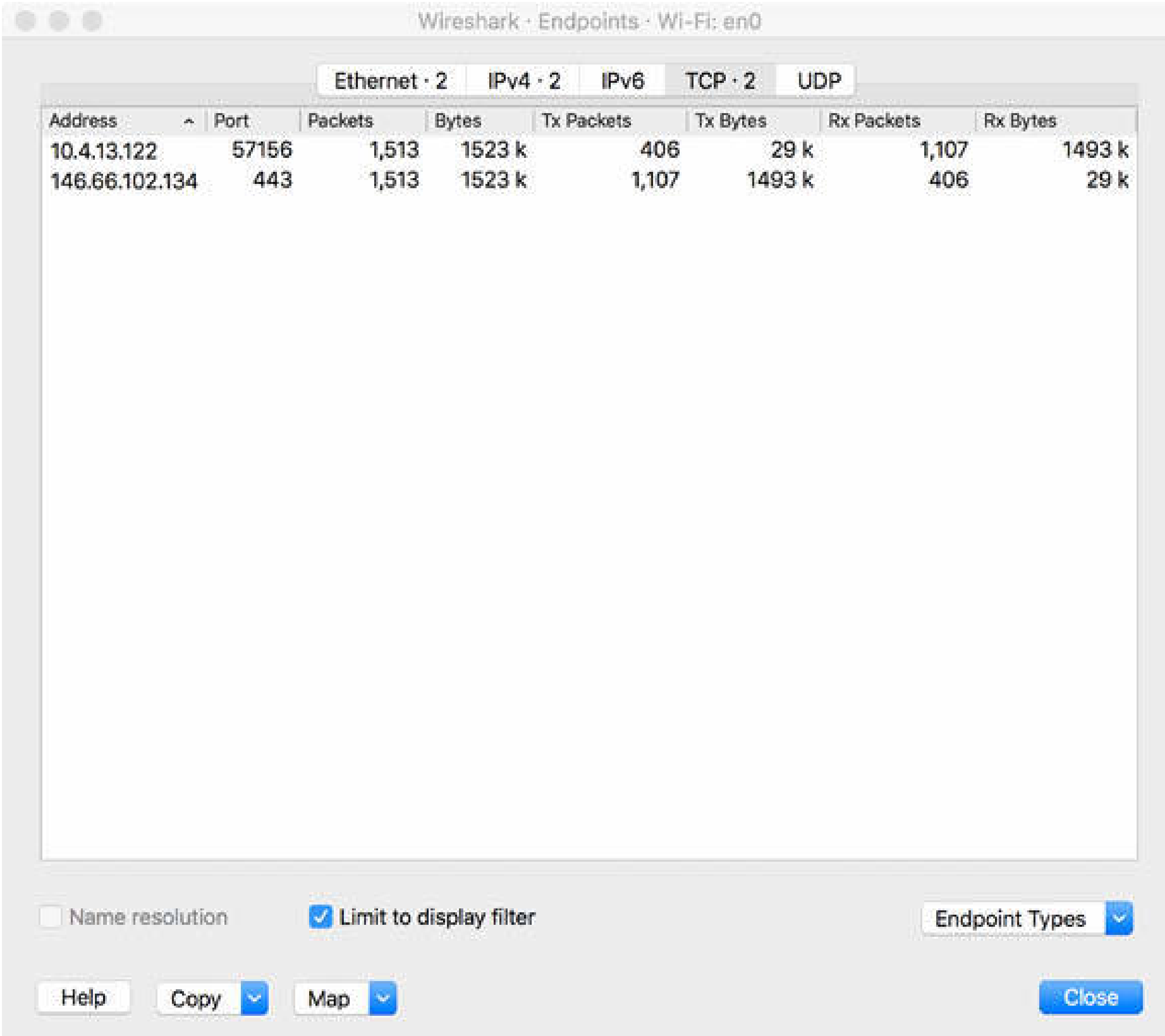


Task 4:
On the main menu, select Statistics > Endpoints, as shown in the figure below.



The Endpoints dialog box is displayed showing all endpoints contained in the capture.

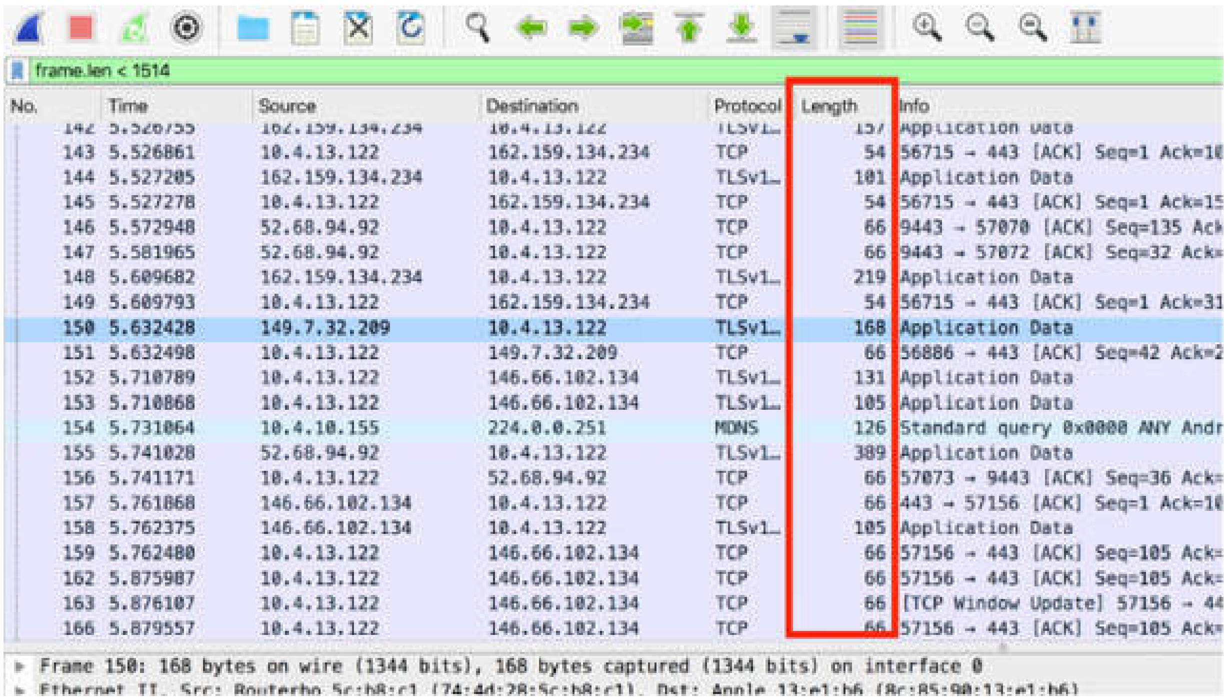
Select the “Limit to display filter” check box to show only those endpoints that are related to the filtered packets. The endpoints shown in the figure below are displayed.



Task 5:

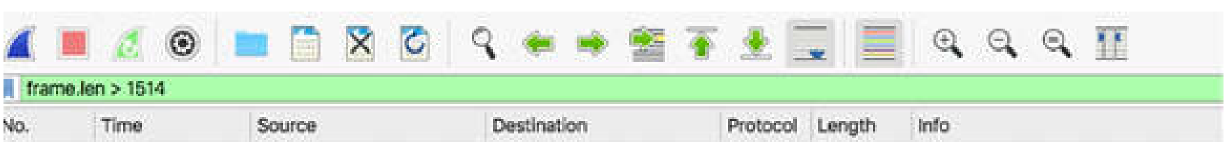
The display filter used in Task 2 is an example of using the comparison operator (==). You can also use comparison operators such as “>”, “<”, “>=”, “<=”, “!=”.

In the filter toolbar, enter frame.len < 1514 . In the Packet List pane, all packets with length less than 1514 bytes are displayed.

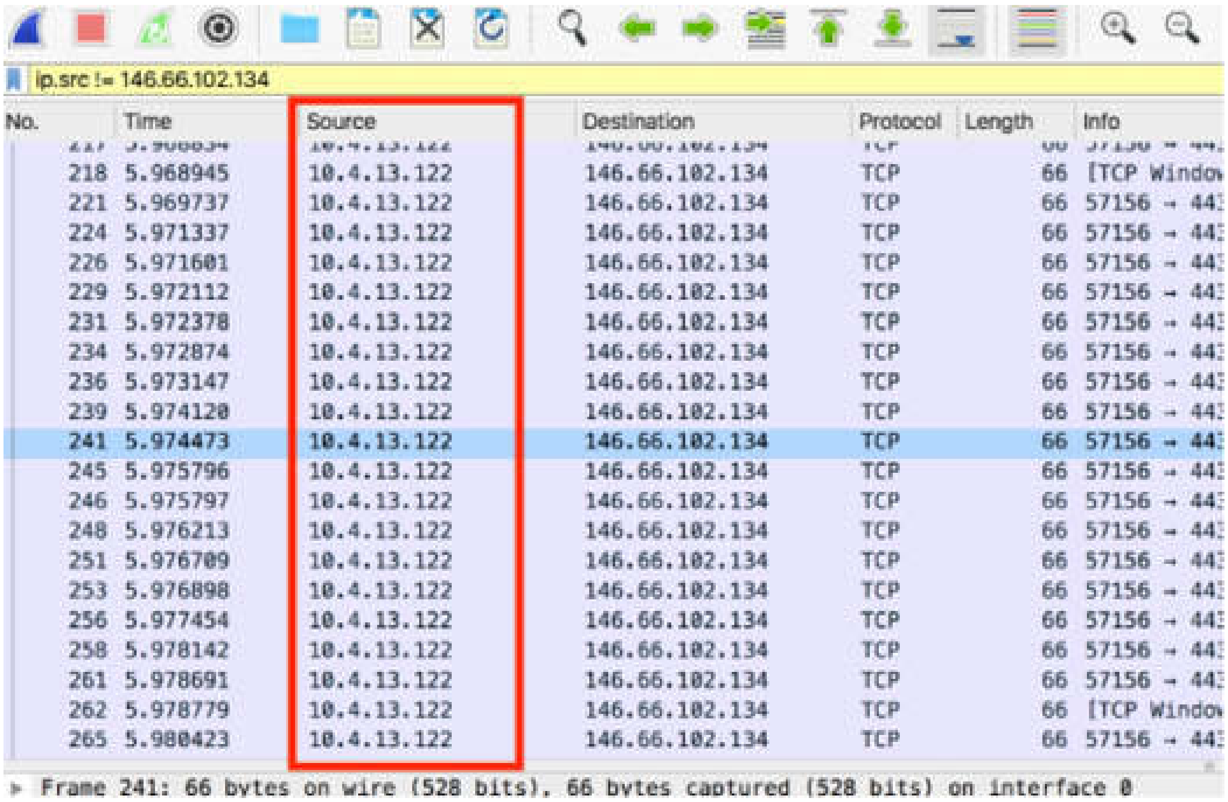


Task 6:
In the filter toolbar, enter frame.len > 1514 . In the Packet List pane, all packets with a length greater than 1514 bytes are displayed.

In this example, no packets are displayed because the maximum length of packets is 1514.

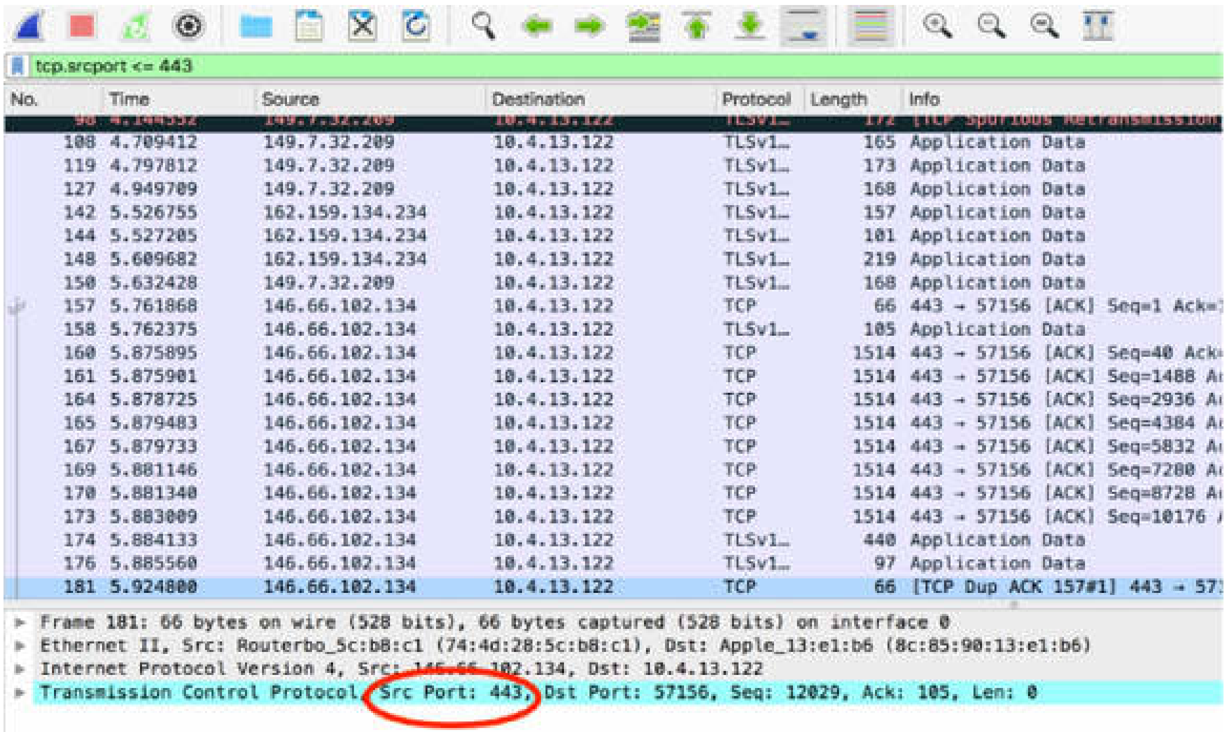


Task 7:
In the filter toolbar, enter ip.src != 146.66.102.134 . In the Packet List pane, all packets with the source IP address not equal to 146.66.102.134 are displayed.



Task 8:

In the filter toolbar, enter tcp.srcport <= 443 . In the Packet List pane, all packets with TCP source port less than or equal to the value 443 are displayed.



Notes:

To gain more confidence in using the filter syntax, create more display filters by using the comparison operators focusing on protocol types, IP addresses, or port used.

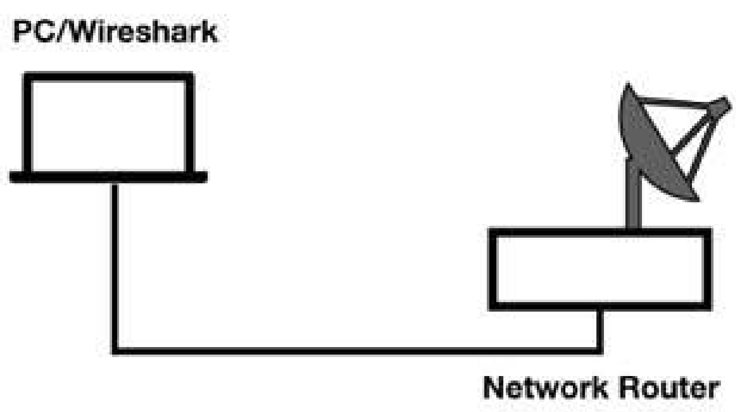
Lab 23. Display Filters—Field Existence and Byte Content

Lab Objective:
Learn how to filter based on the existence of a field and a specific byte in the packet.

Lab Purpose:
Understand how to check with a filter the existence of a field and how to search for a specific byte content inside the packet.

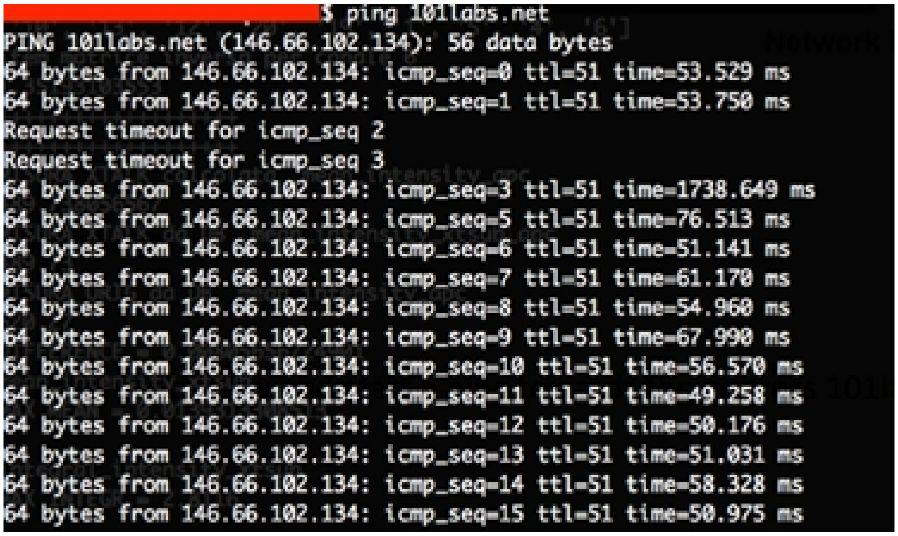
Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



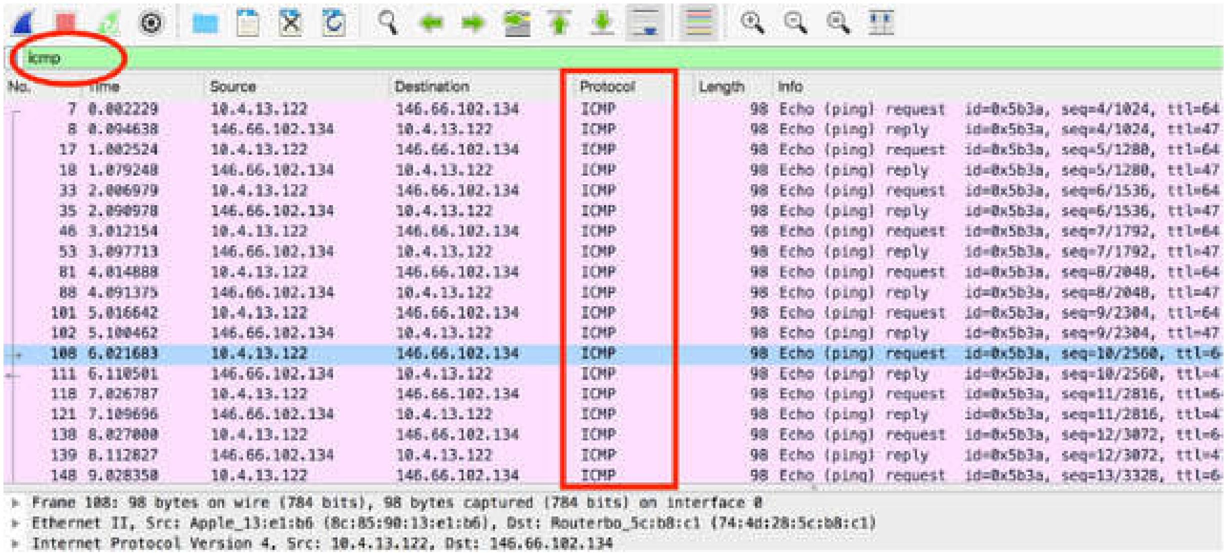
Lab Walkthrough:

Task 1:
Open a terminal window, and ping the address 101labs.net by running the ping 101labs.net command.



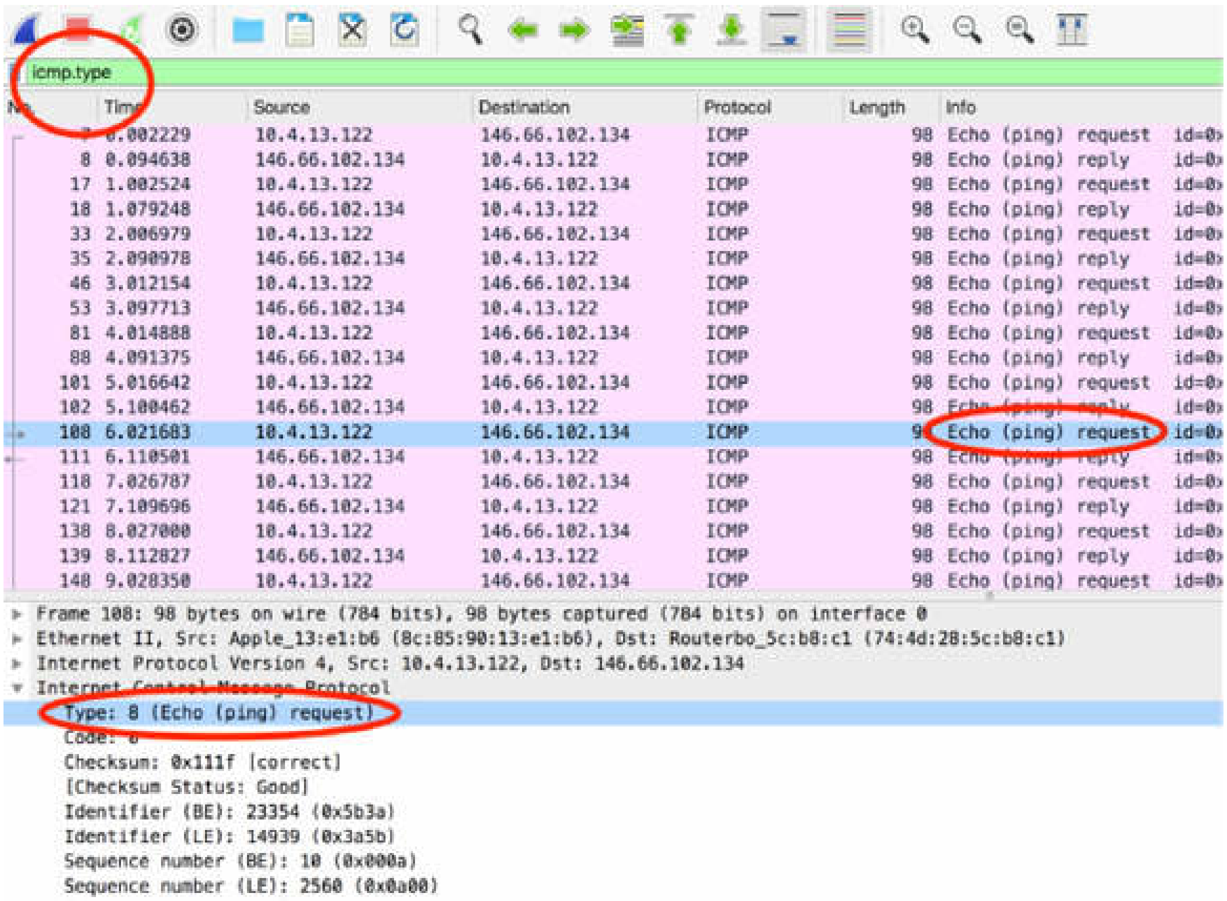
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes. Stop the capture and save the file.

Task 2:
To select only those packets from the saved capture file that were exchanged with the ping request in Task 1, in the filter toolbar, enter icmp . In the Packet List pane, only those packets that belong to the ICMP protocol are displayed.



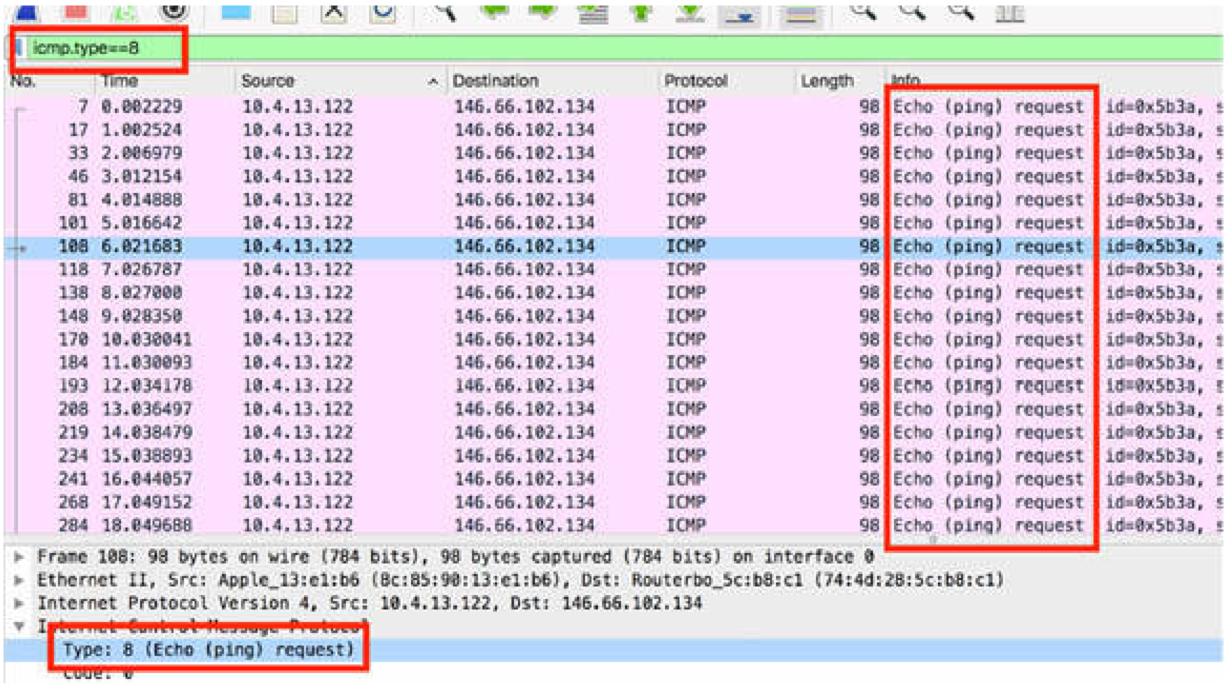
Task 3:

To identify whether an ICMP packet is a request or a response and to select ICMP packets with the Type field, in the filter toolbar, enter icmp.type . The Packet List pane displays all packets that contain the Type field of the ICMP protocol.

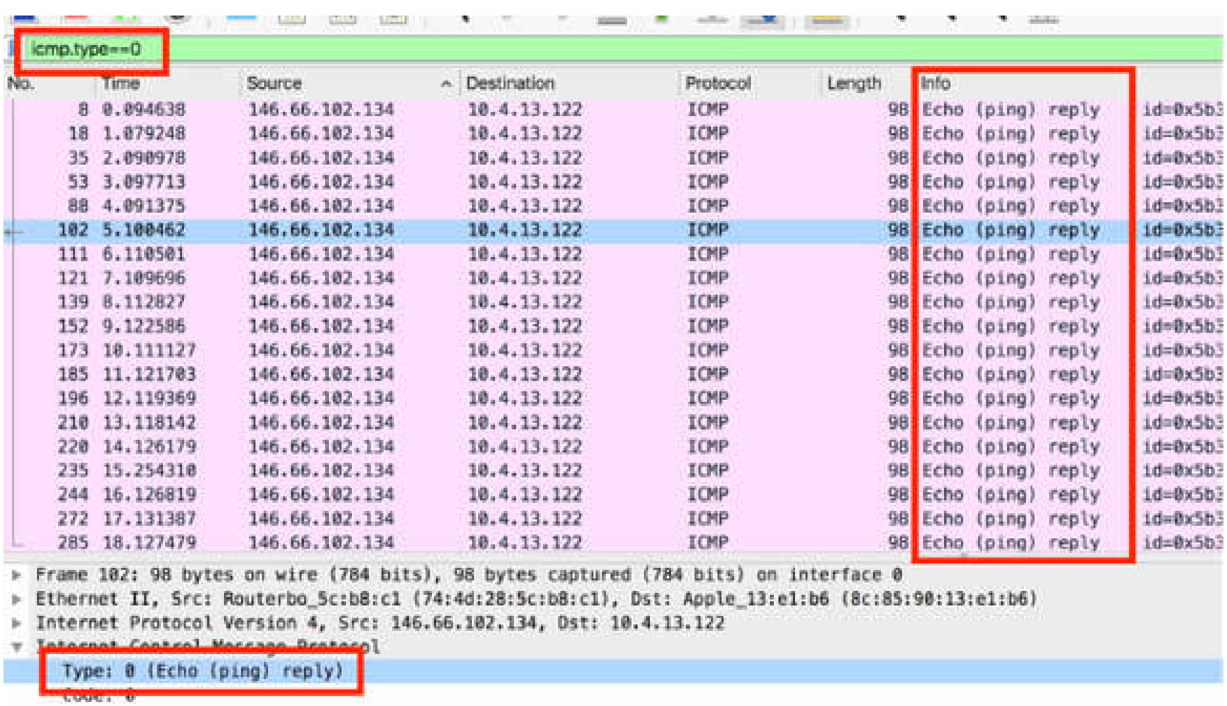


Task 4:

To select only ping requests, in the filter toolbar, enter icmp.type == 8 . In the Packet List pane, only ping requests are displayed, as shown in the figure below.



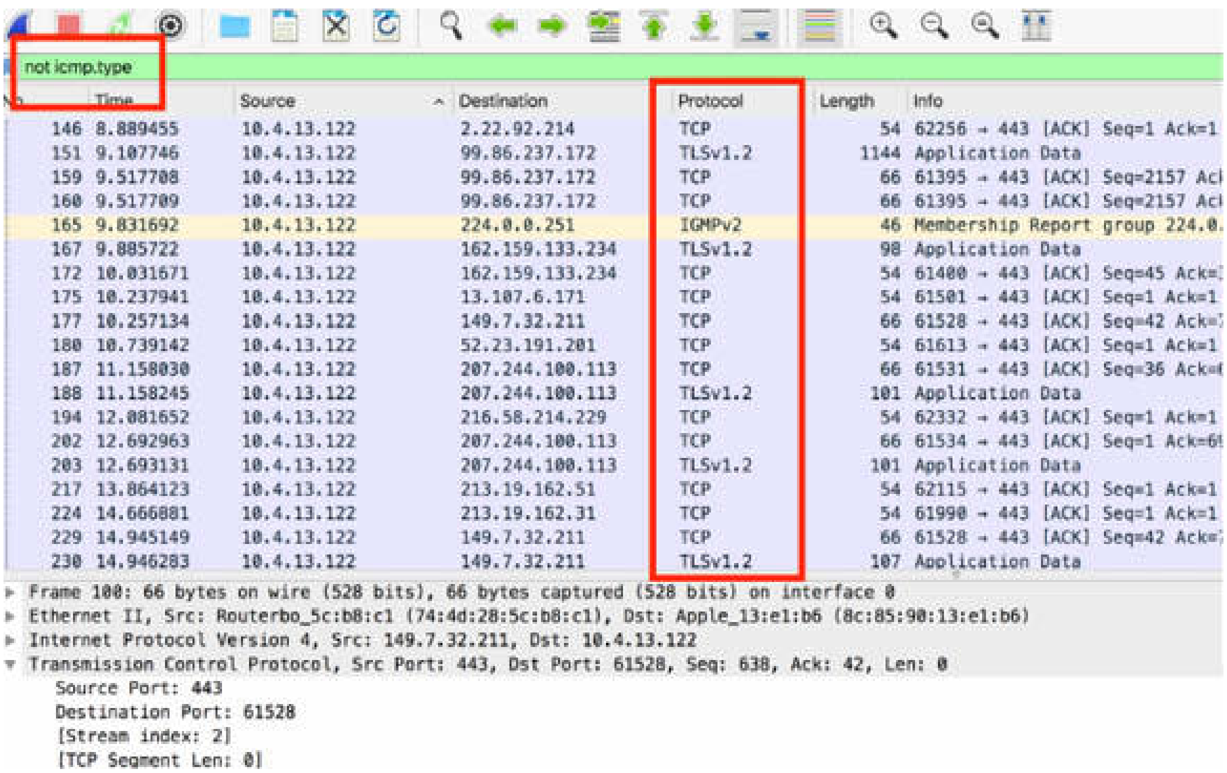
To select only ping replies, in the filter toolbar, enter icmp.type == 0 . In the Packet List pane, only ping replies are displayed, as shown in the figure below.



Task 5:

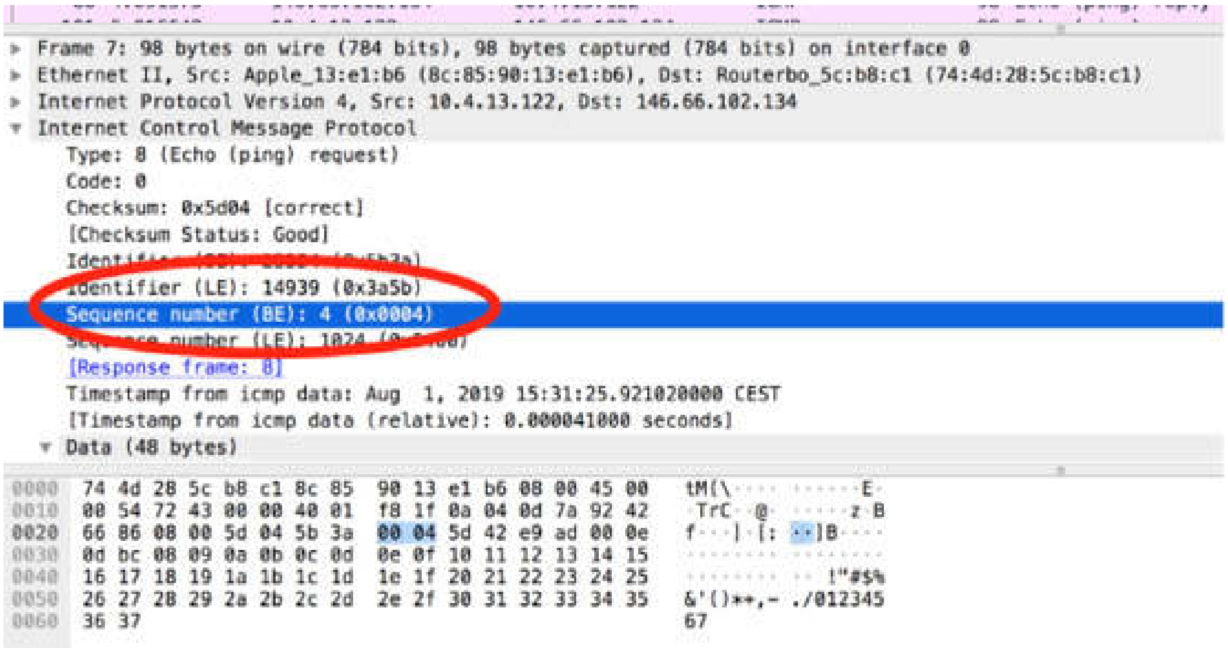
In Task 3 and 4, you checked the existence of a field with a display filter (icmp.type) and a particular value of that field (icmp.type == 0 or icmp.type == 8).

To filter all packets that do not have a particular field (such as Type), in the filter toolbar, enter not icmp.type . In the Packet List pane, all packets belonging to a protocol other than ICMP are displayed because all ICMP packets contain the Type field.

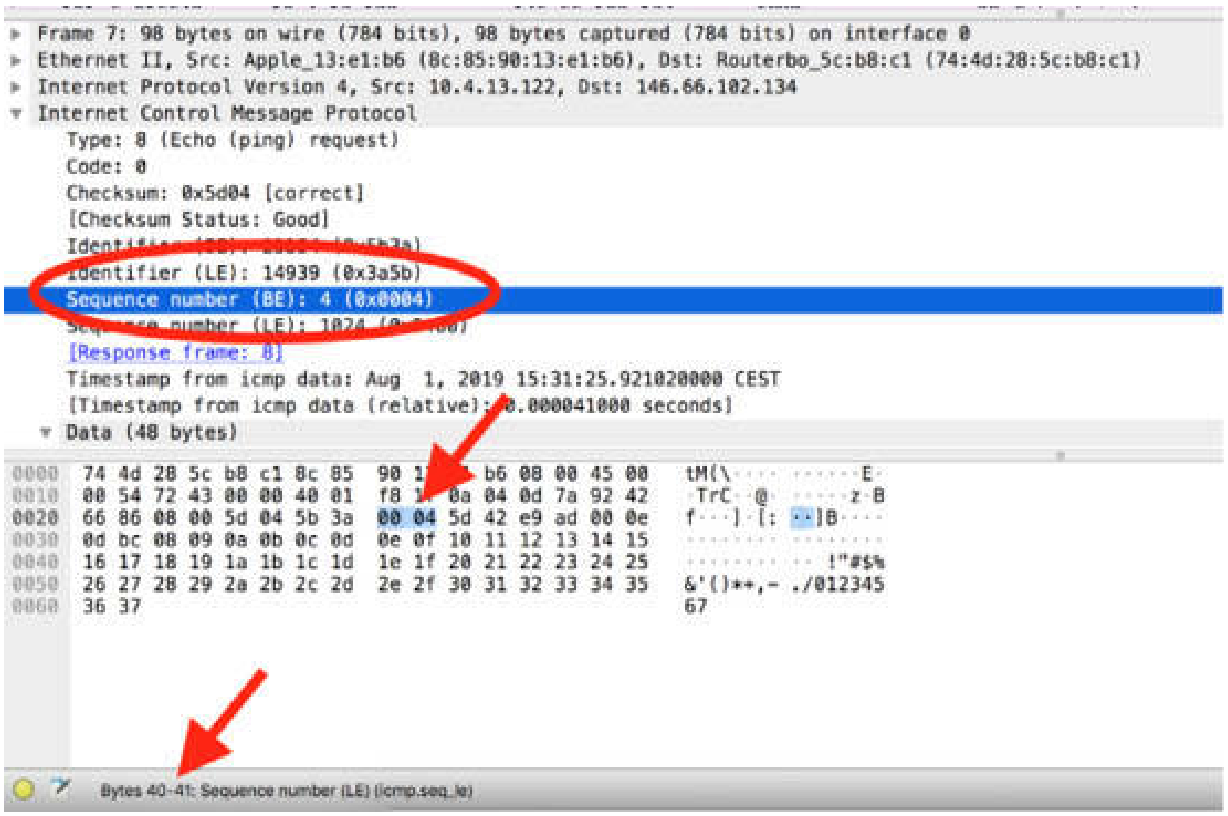


Task 6:

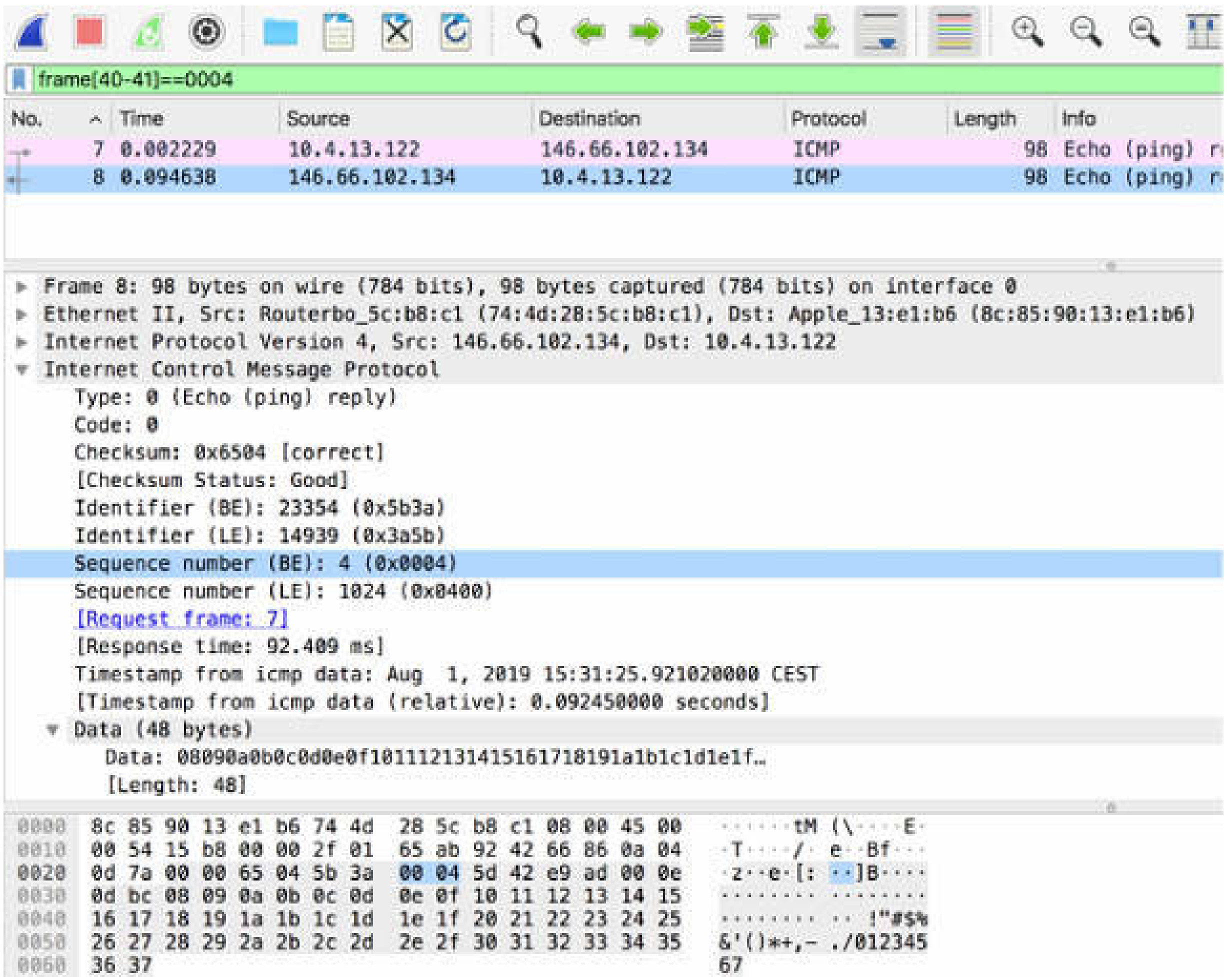
Again filter only ICMP packets, and in the Packet List pane, select a packet. In the Packet Details pane, select a field such as “Sequence number” by using the mouse.



In the Packet Bytes pane, the field correspondent to your selection and the related byte offset (such as 40–41) are highlighted, as shown in the figure below.

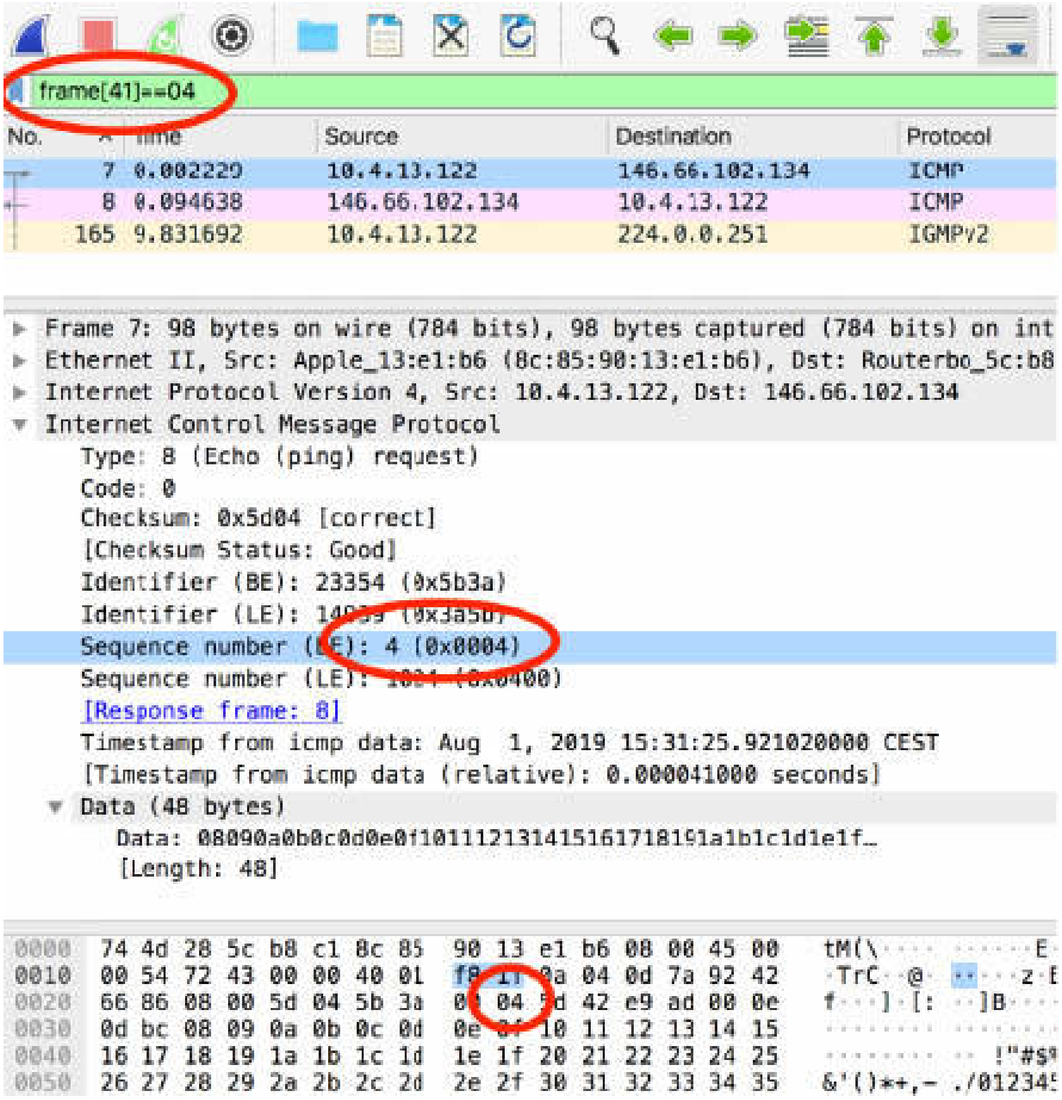


To select only those packets that have bytes 40–41 with the hexadecimal value 0x0004 , in the filter toolbar, enter frame[40-41] == 0004 , as displayed in the figure below.

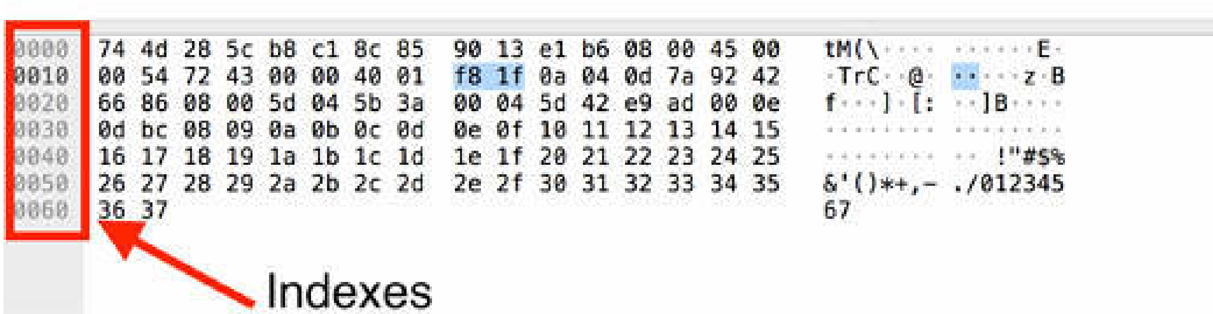


Task 7:

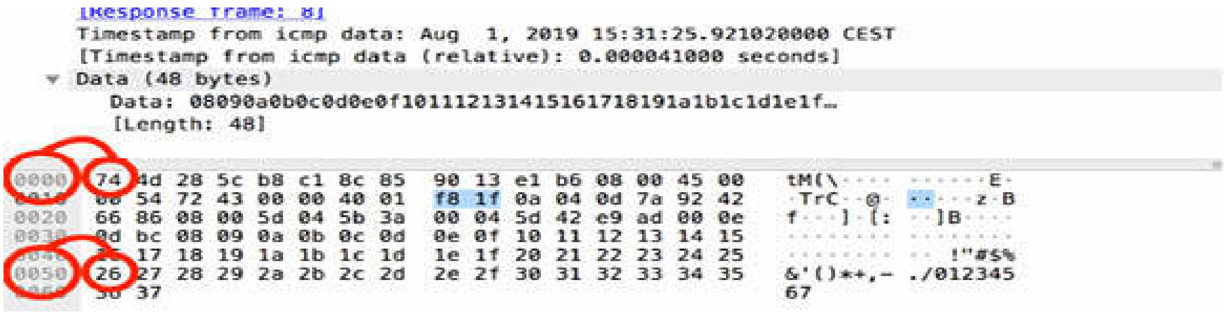
To filter for a single byte value inside the packet, in the filter toolbar, enter frame [ByteNumber] == Value , such as frame[41] == 04 . In this example, you are searching the 41st byte for a value, and as a result, you get all packets having that byte at the specified value.



As shown in the first field of the byte stream, the indexes in the byte stream start with the value 0 (zero)



The indexes column represents the index list of the first byte of each byte stream row and is expressed in HEX format.



Notes:

To gain more confidence in using the filter expression on fields and byte values, repeat the previous tasks by using the HEX notation or decimal notation.

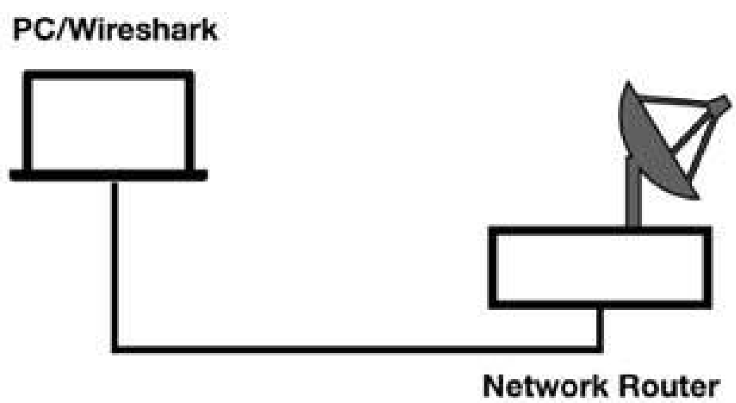
Lab 24. Display Filters—Keywords

Lab Objective:
Learn how to filter the existence of a keyword in uppercase or lowercase and avoid common mistakes.

Lab Purpose:
Understand how to use a filter to check the existence of a keyword in both uppercase or lowercase and how to avoid the most common mistakes in filter syntax.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

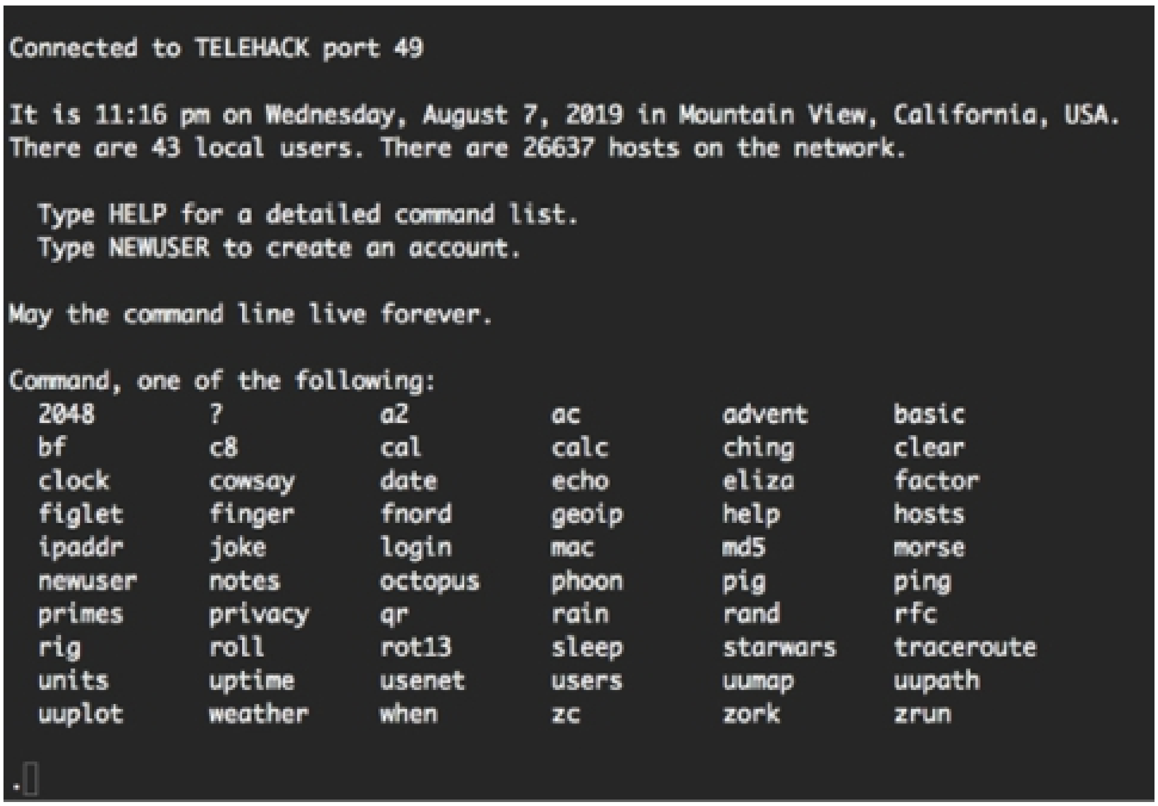
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



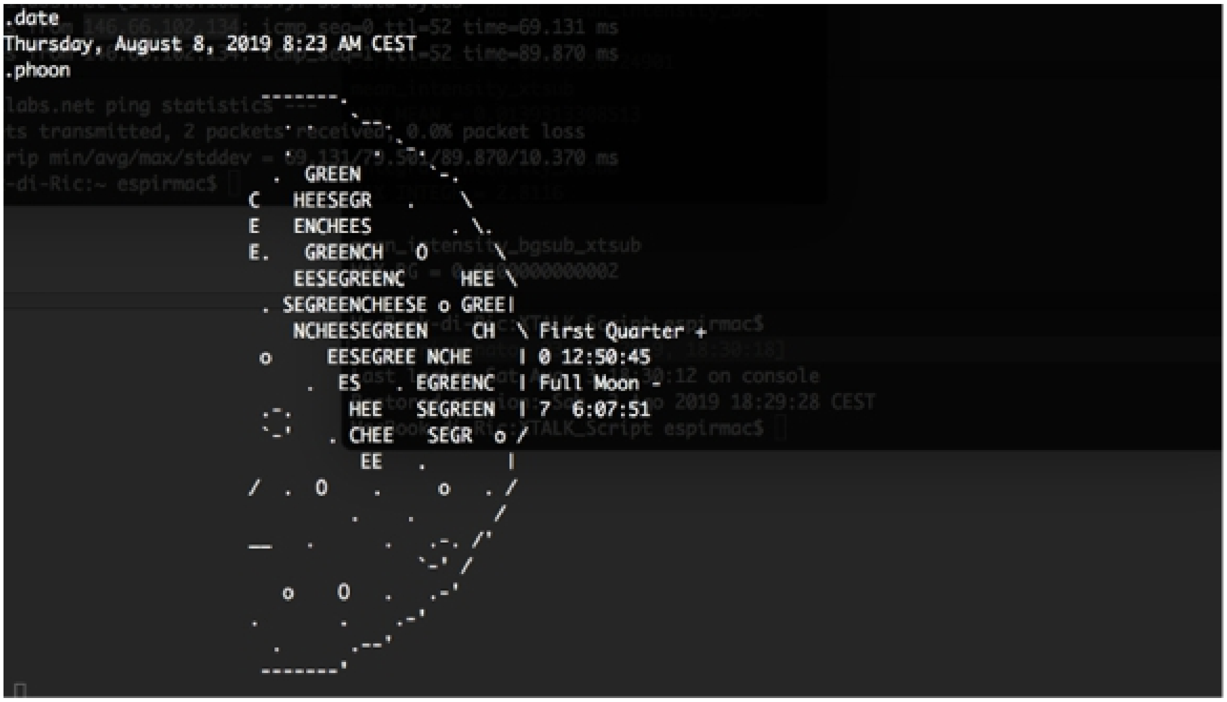
Lab Walkthrough:

Task 1:
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

Open a terminal window, and type the telnet telehack.com command. A connection towards the FTP server telehack.com is opened, as shown in the figure below. If you are using the Windows operating system, you may first need to enable Telnet (See the note at the end of this lab).

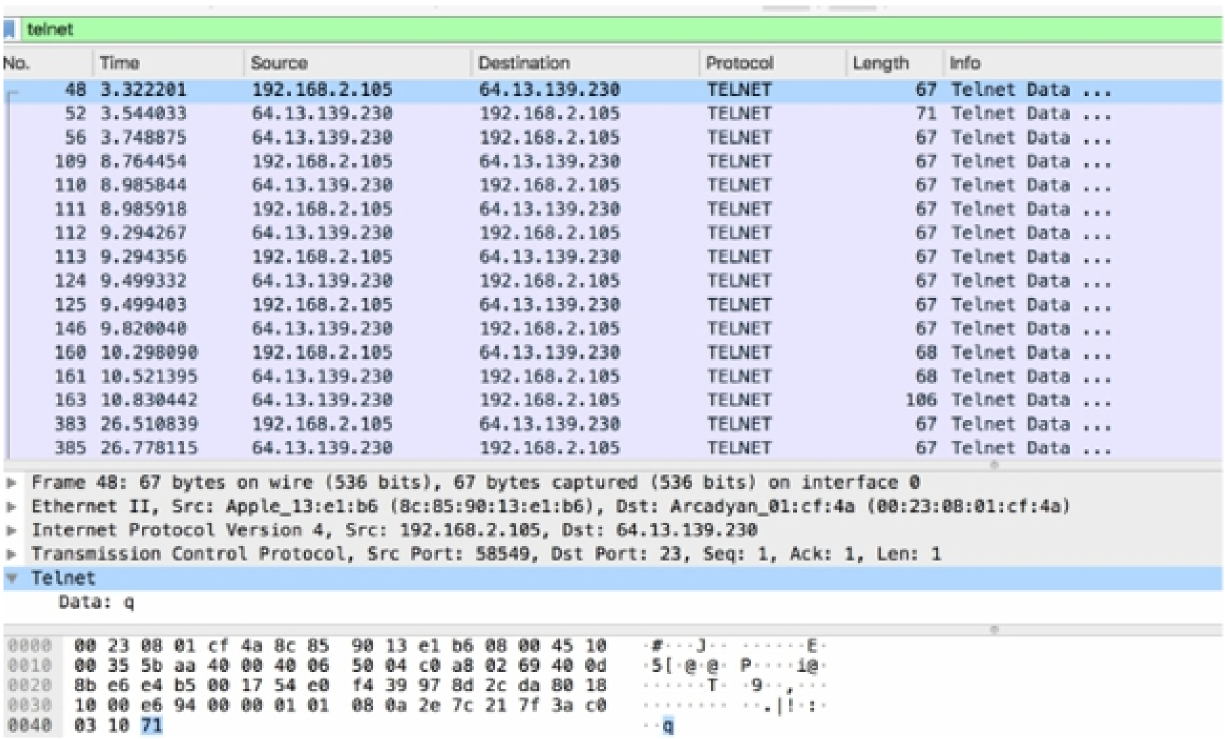


Task 2:
Run the command date to get the actual date printed, and then run the command phoon to get the moon phase printed. The terminal window displays something similar to as shown in the figure below.

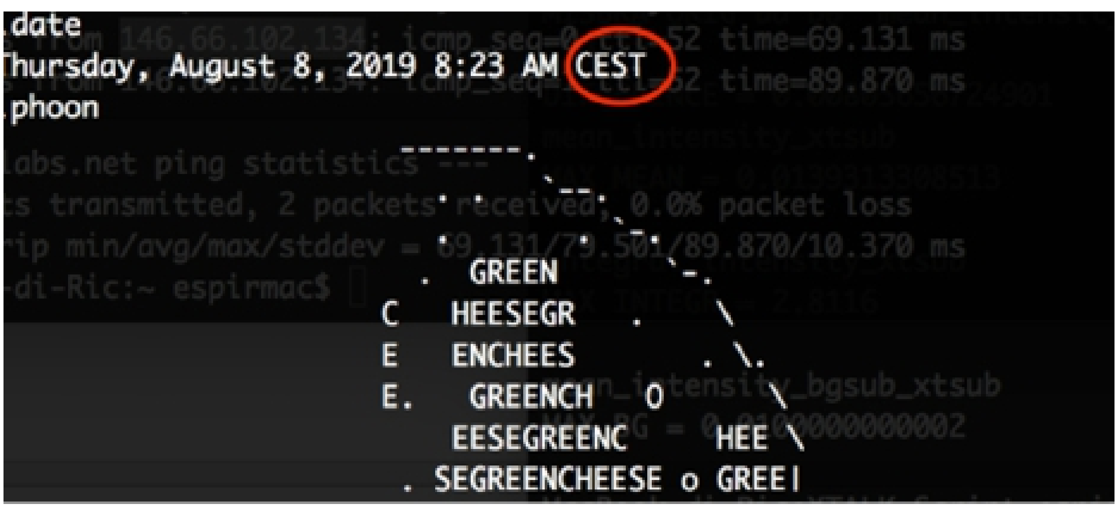


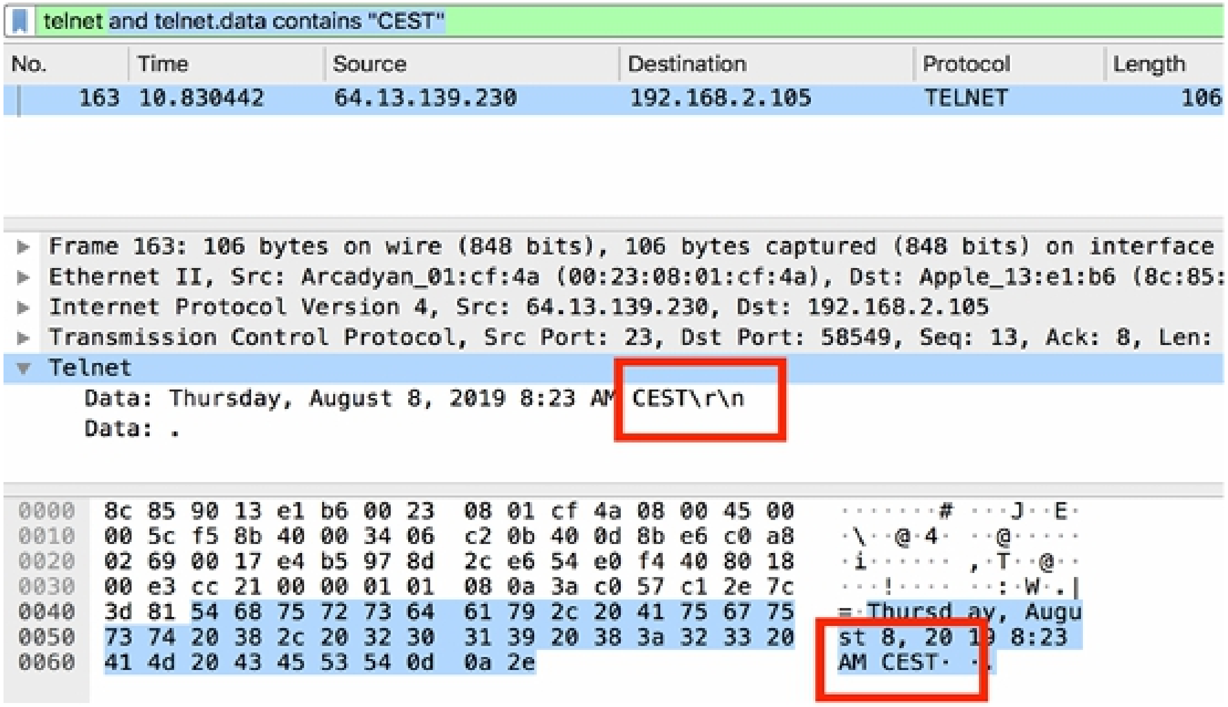
Stop the capture and save the file.

Task 3:
In the filter toolbar, enter telnet . In the Packet List pane, only Telnet packets from and to the server telehack.com are displayed.



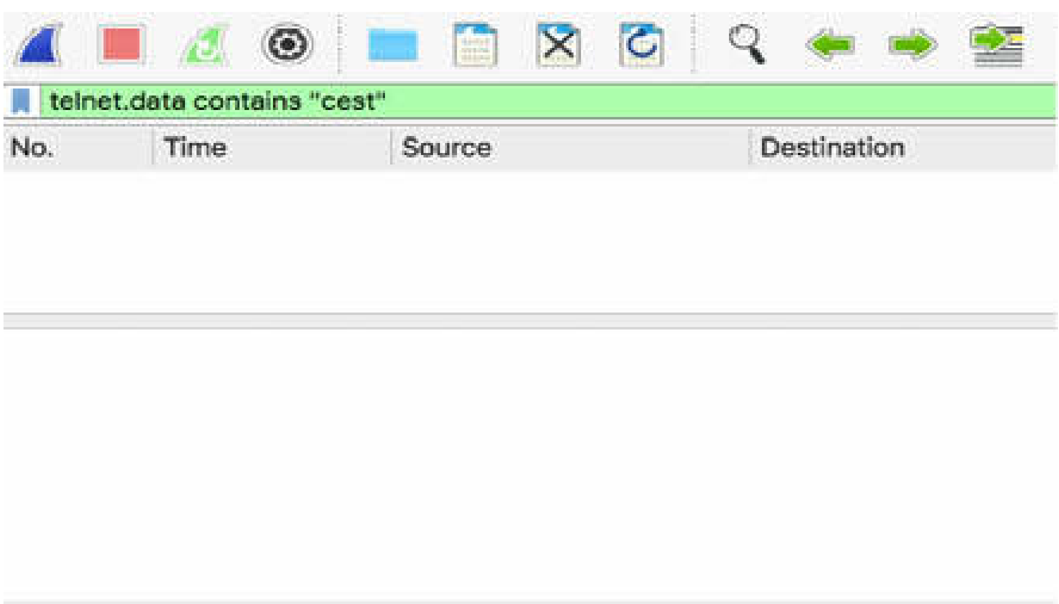
Task 4:
In the filter toolbar, enter and telnet.data contains “CEST” to capture the packet in which the display data is sent from the server. Note that the connection used for this example produced AEST. Therefore, in the syntax, “CEST” is swapped for “AEST”.



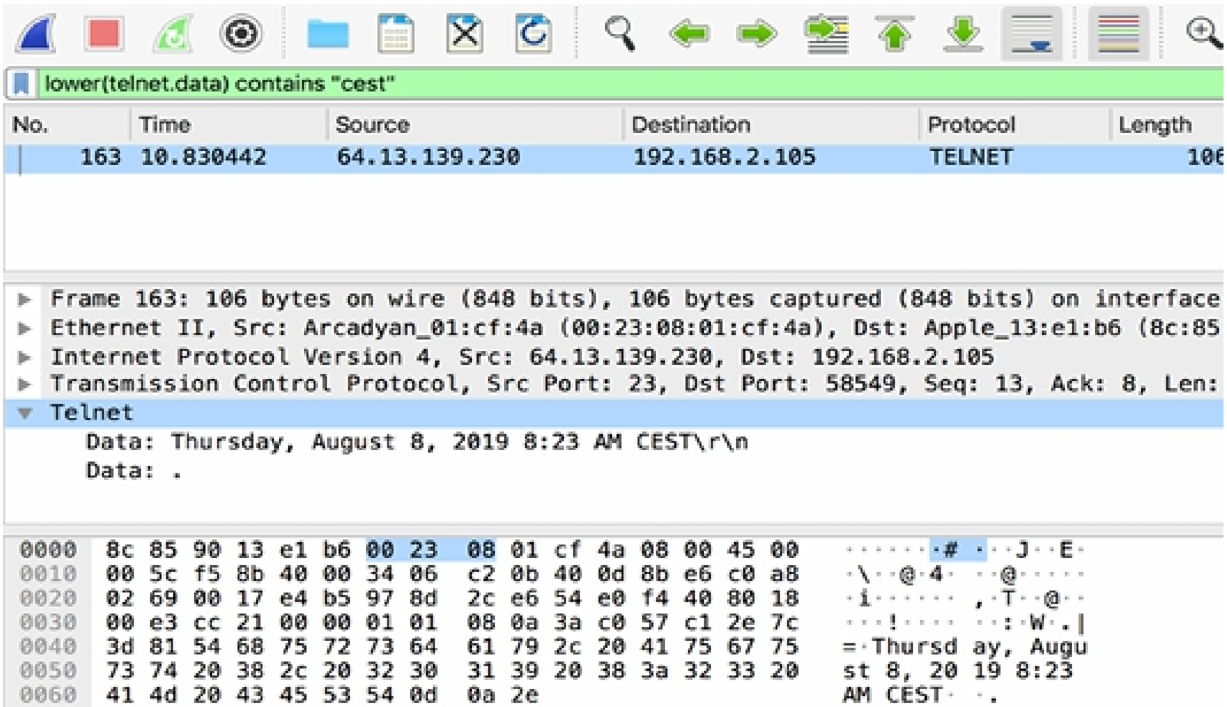


Task 5:

If you don’t know whether the string you are searching for is in uppercase or lowercase and you use the filter telnet.data contains “cest” , no packets get filtered.

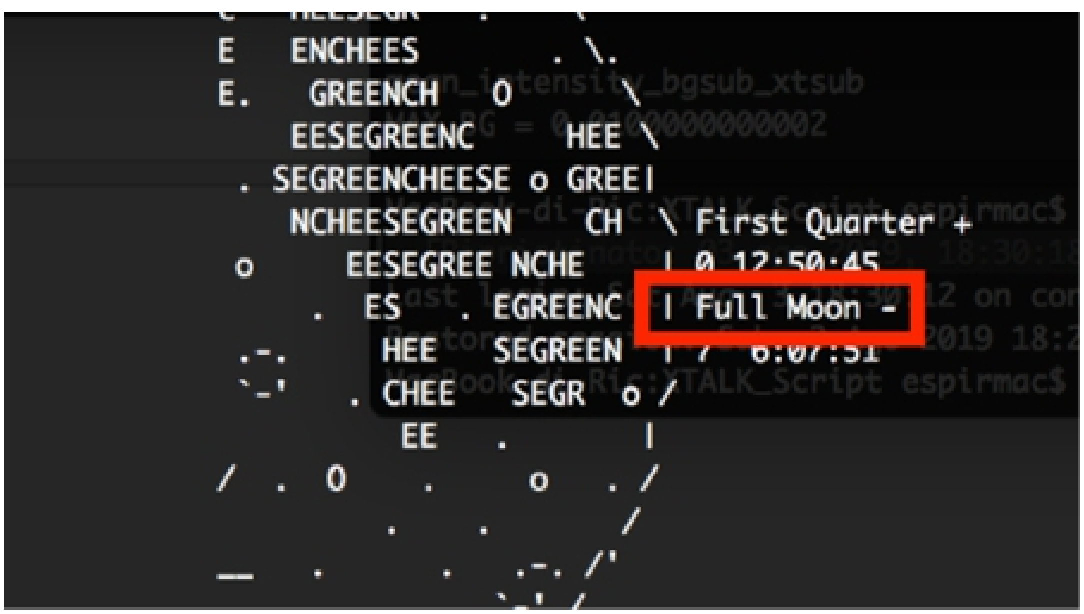


You can solve this problem by converting the string field to lowercase (or uppercase) before creating the filter, such as lower(telnet.data) contains “cest” .

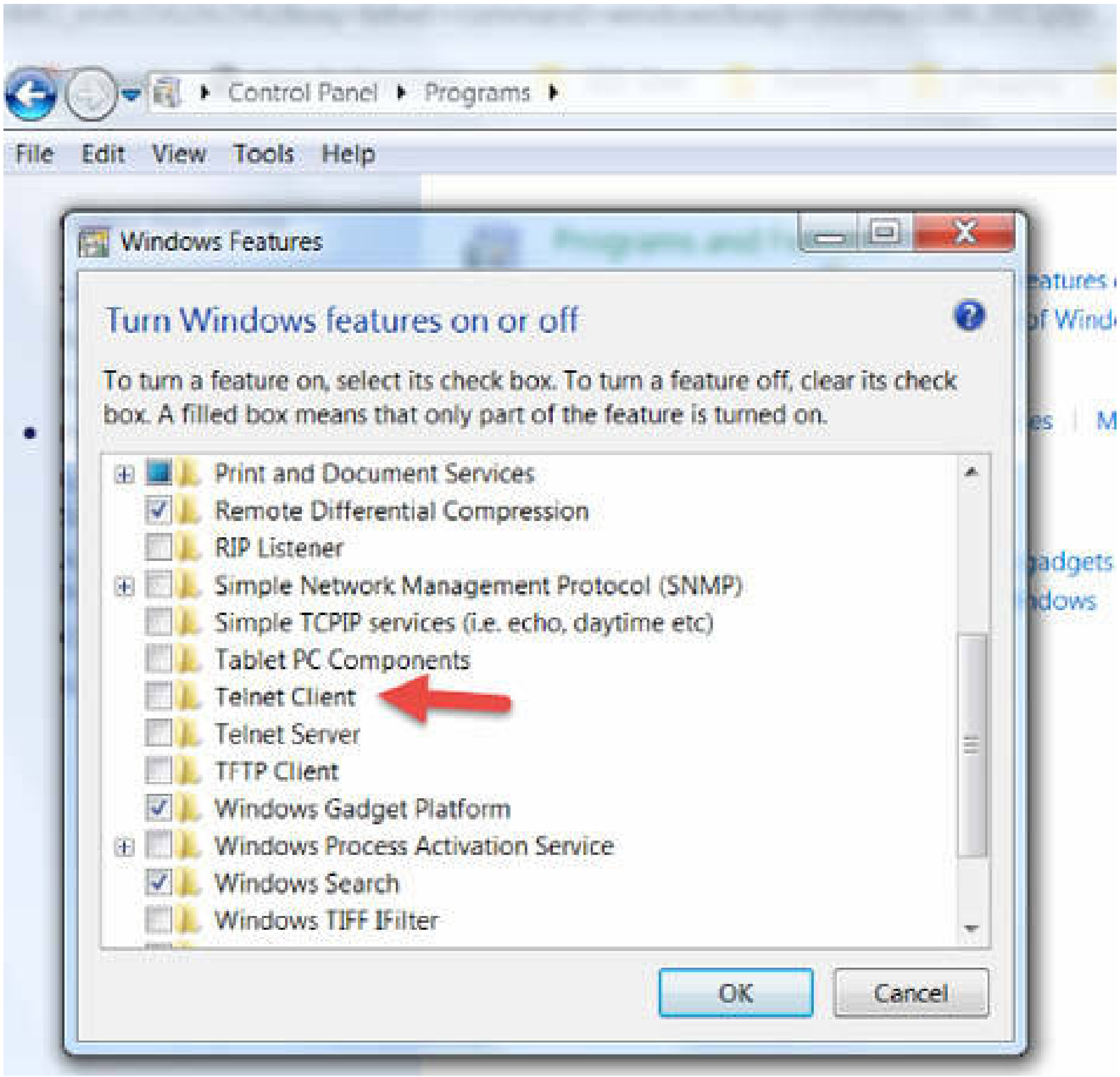


Task 6:

To get the packets containing the moon phase, you can apply the filter upper(telnet.data) contains “FULL MOON” [or lower(telnet.data) contains “full moon”].



Notes:
To gain more confidence in using keywords, repeat the previous tasks, and use both the lowercase and uppercase expressions. You can also try combinations or more complex filters to search for more than one keyword inside a packet.



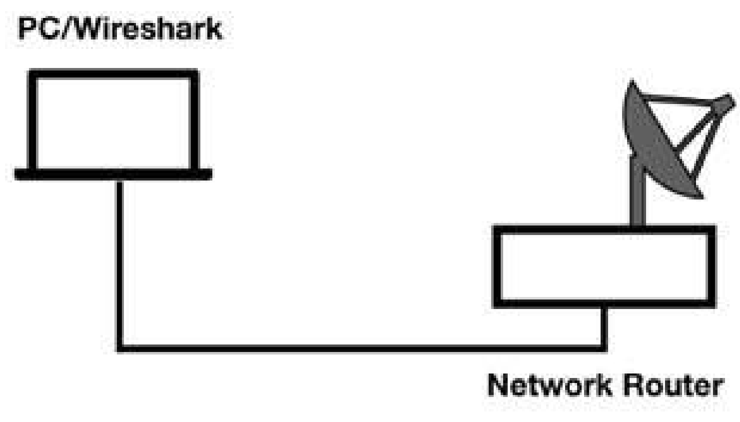
Lab 25. TCP Streams

Lab Objective:
Learn how to follow a TCP stream.

Lab Purpose:
Understand how to show only the packets in a TCP stream to understand and decipher a data stream.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

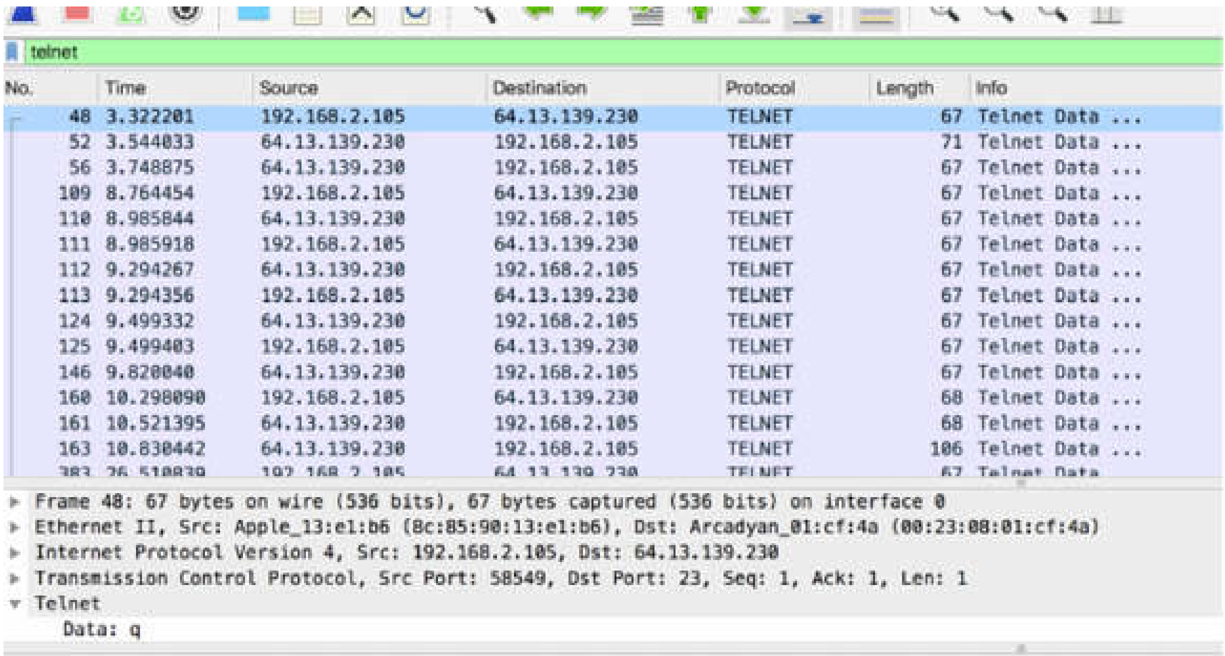
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



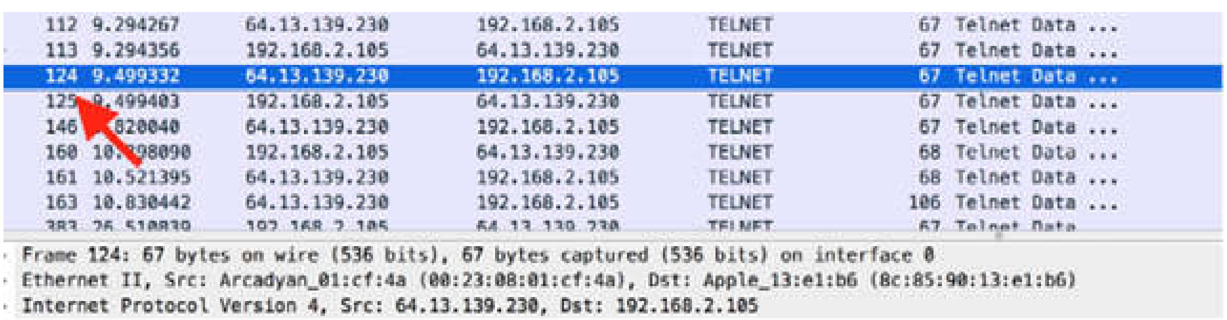
Lab Walkthrough:

Task 1:
In Wireshark, open the capture file saved in lab 24.

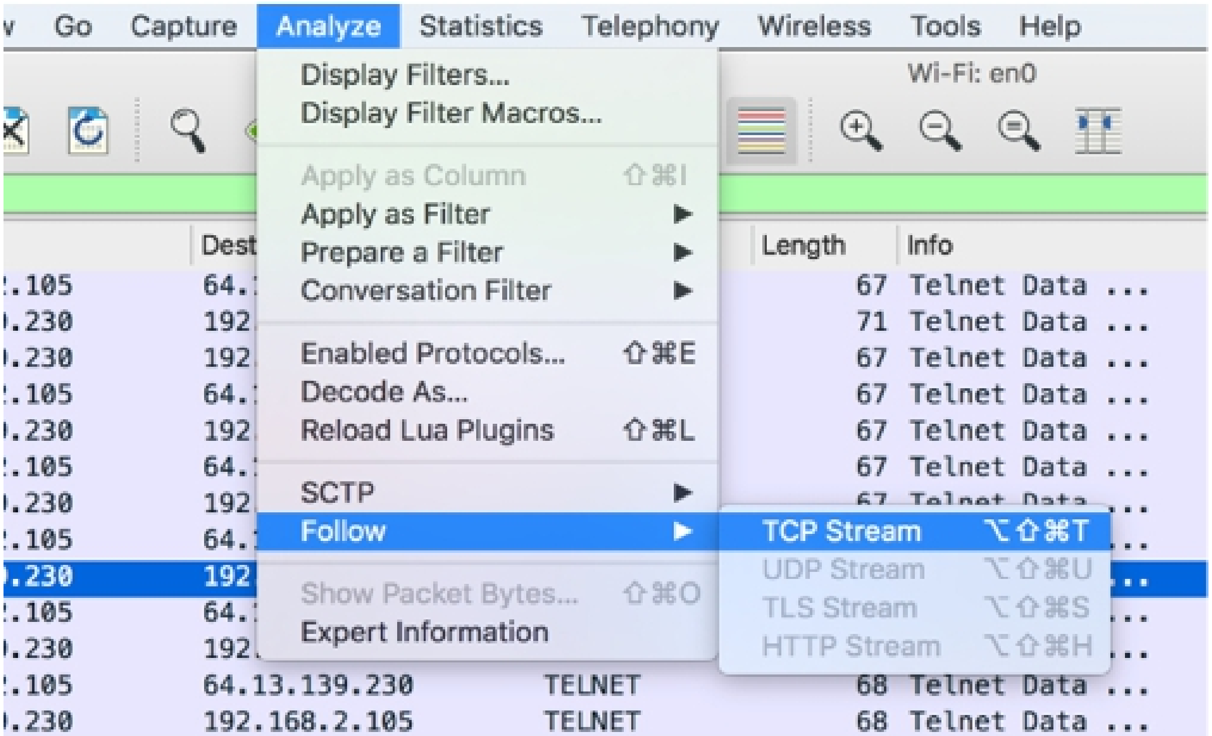
Let’s say you are interested in the Telnet packets, and you need to follow this stream because you are trying to understand and make sense of the data contained in it. First, you need to display only Telnet packets. To do so, in the filter toolbar, enter telnet .



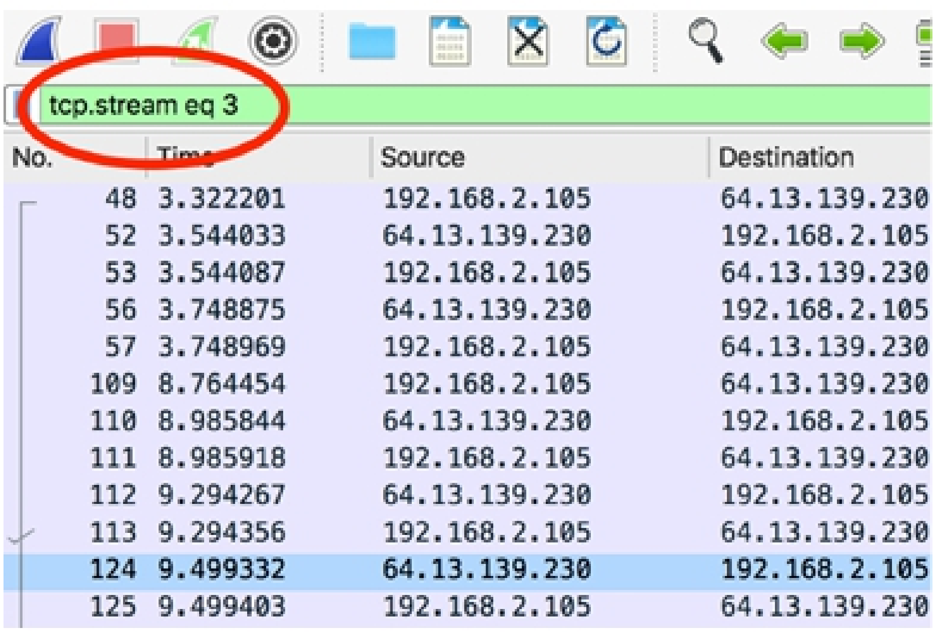
Task 2:
Select a packet (such as #124), as shown in the figure below.



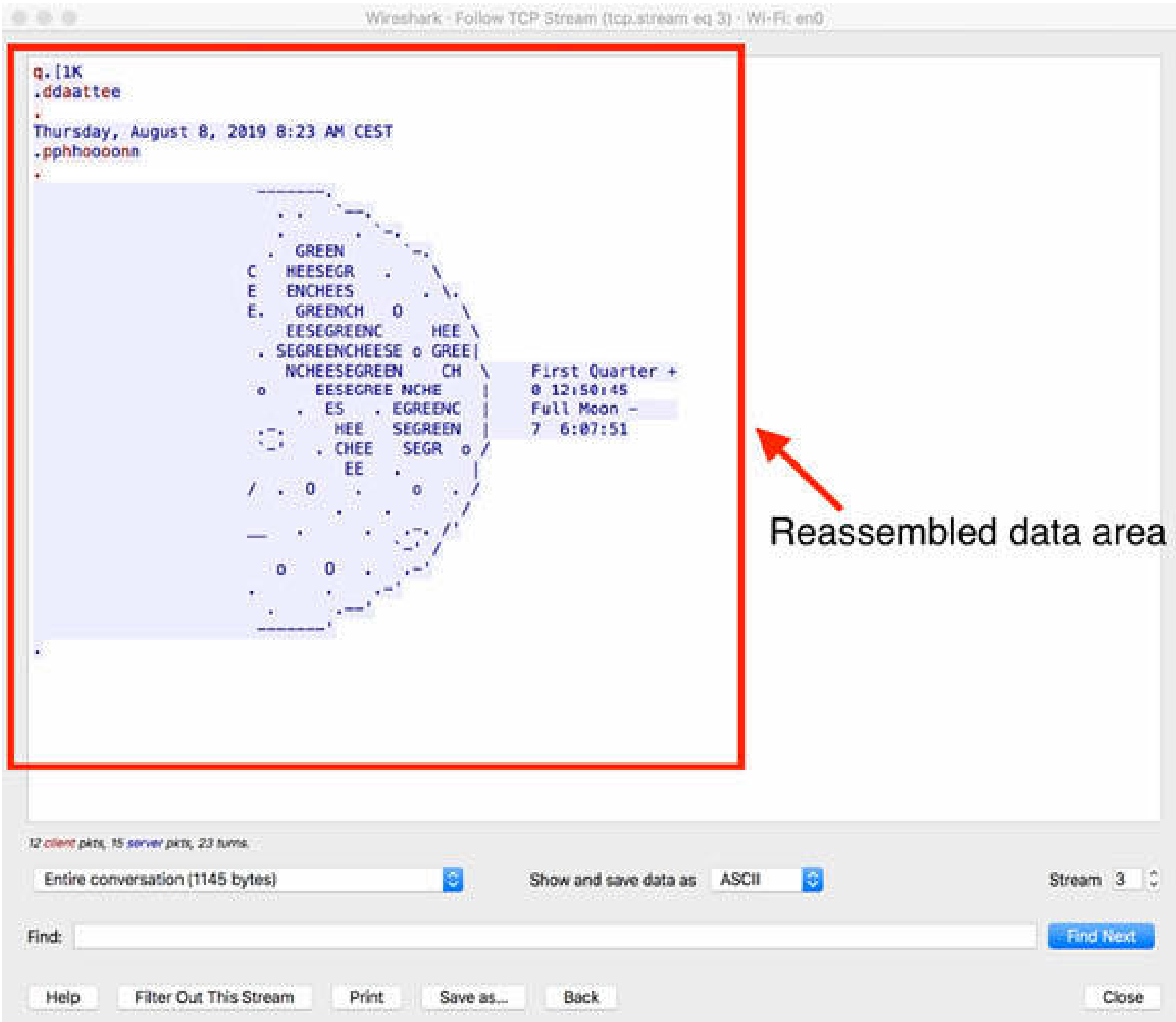
Task 3:
On the main menu, select Analyze > Follow > TCP Stream, as shown in the figure below.



Wireshark sets the appropriate filter, and a popup window is displayed containing the relevant information for the TCP stream and all the data from the TCP stream (laid out in order).



In the figure below, the arrow indicates the reassembled data area.

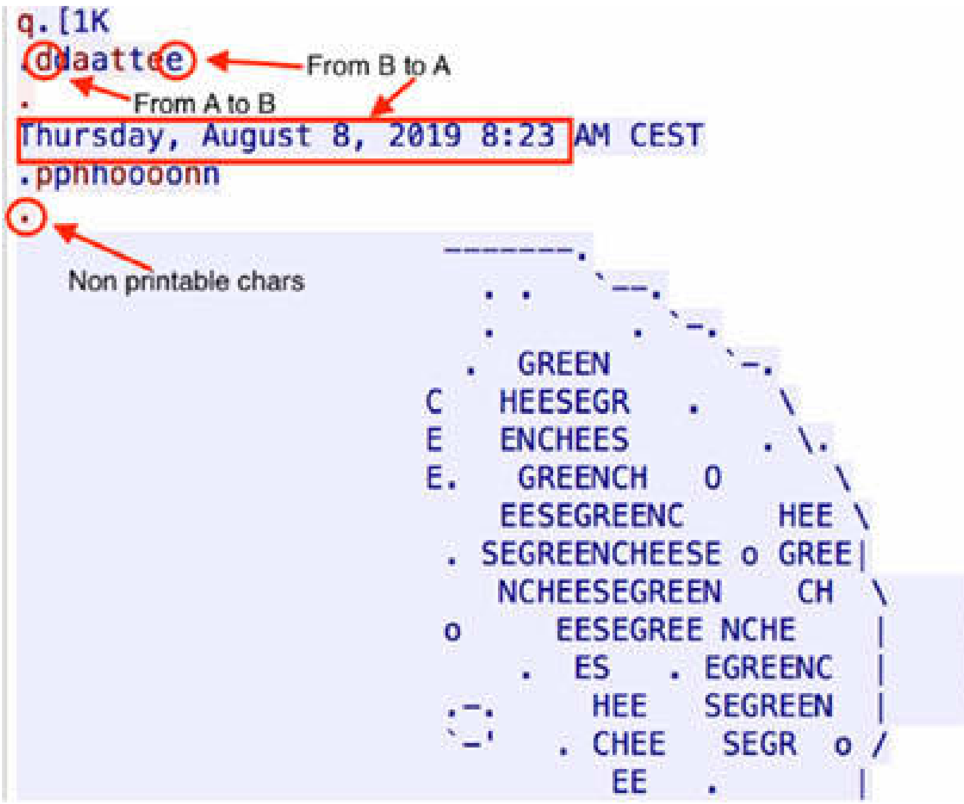


Task 4:

Observe the reassembled data area closely and note that:

- The stream content is displayed in the same sequence as it appeared on the network.
- Traffic from A to B is marked in red, whereas traffic from B to A is marked in blue. You can change these colors in the "Font and Colors" option in the Preferences dialog box. In this case, more packets are sent from the Telnet server, so almost all traffic is marked in blue.

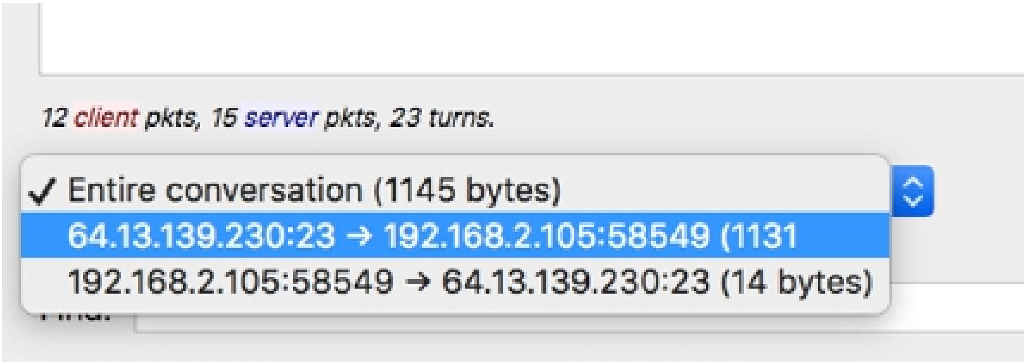
- Non-printable characters are replaced by dots.



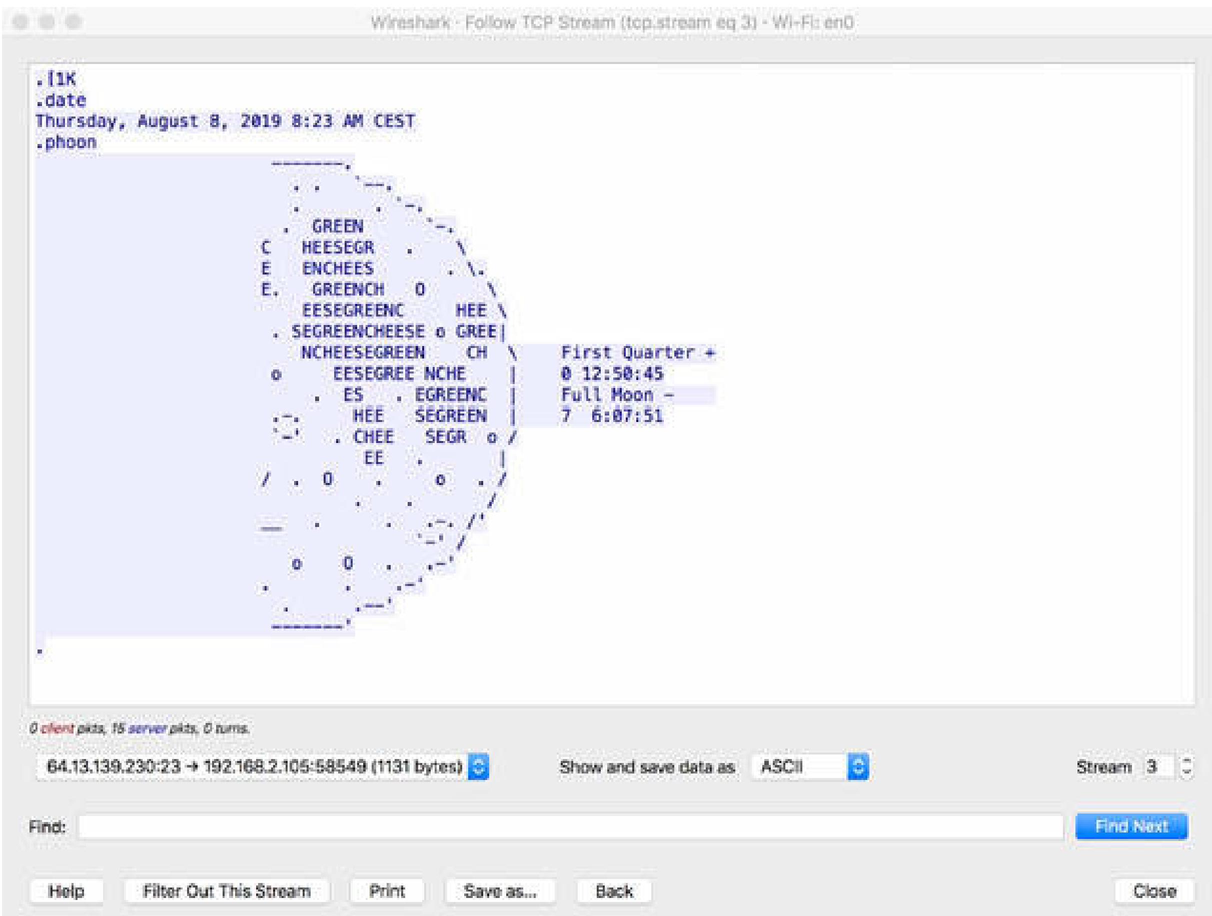
The original images for all labs can be downloaded from the resources page on 101labs.net. The stream content isn’t updated while doing a live capture. That’s why we used a saved capture file. In case you are using a live capture to get the latest content, you need to reopen the dialog box.

Task 5:

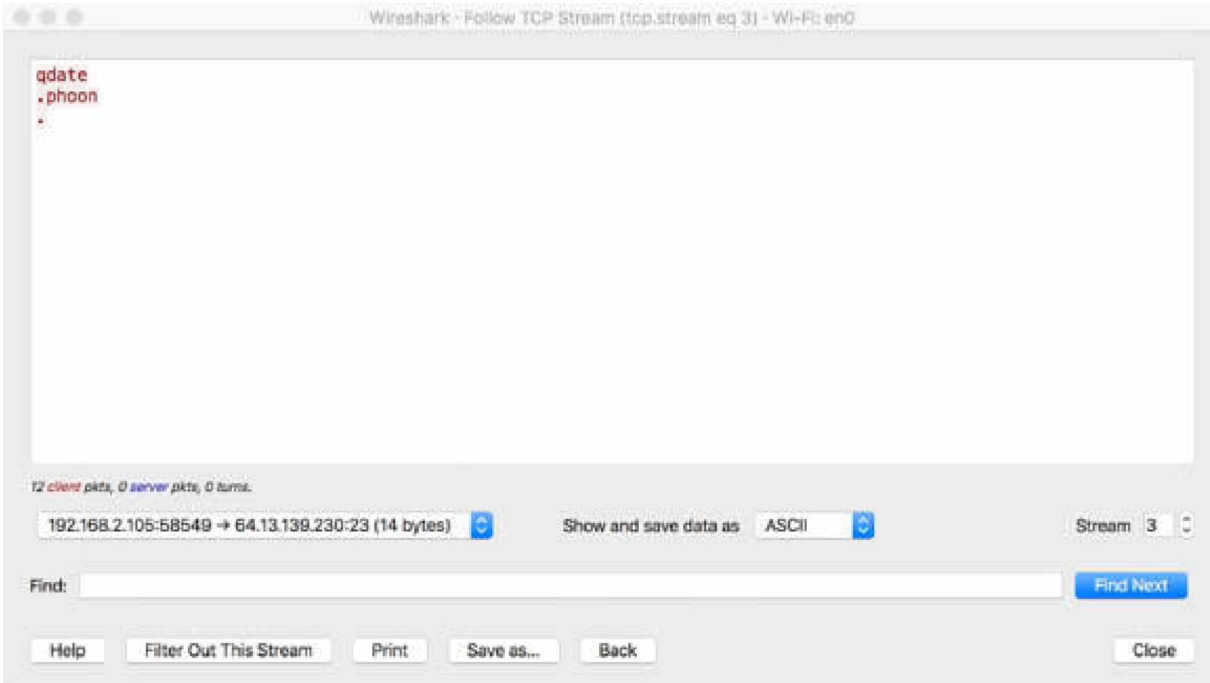
To view only the reassembled data of a single direction of the conversation, click the drop-down list, shown in the figure below.



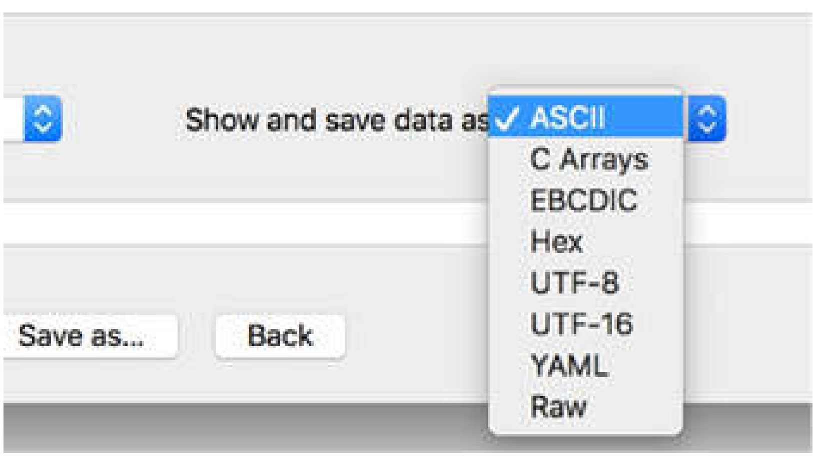
Click the first item. Only the conversation direction A → B is displayed.



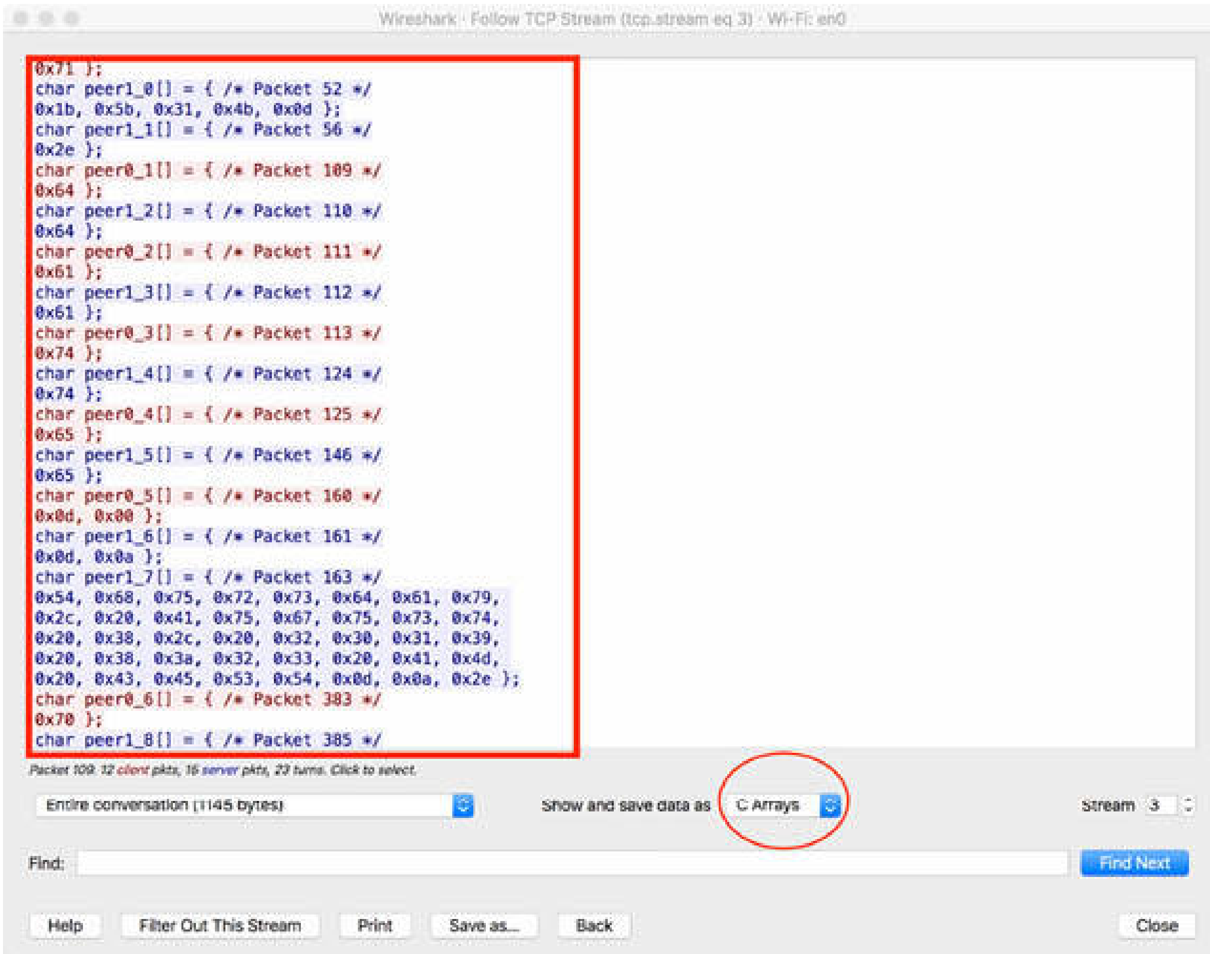
Click the second item. Only the conversation direction B → A is displayed.



Task 6:
To choose a different display and save format for the stream data (the default display format is ASCII, which is the best for ASCII-based protocols, such as HTTP), click the drop-down list shown in the figure below.

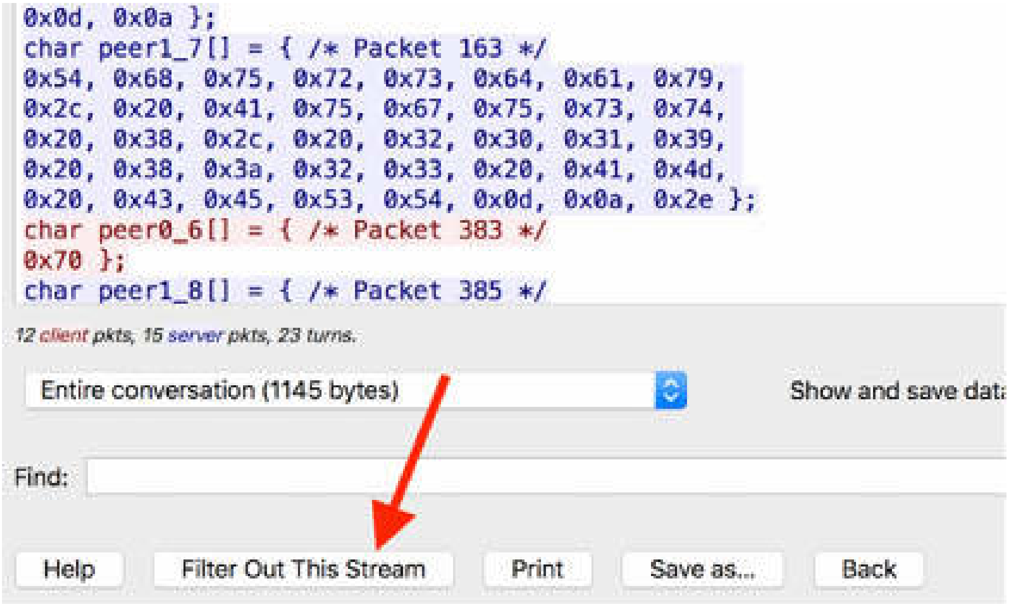


Select the format C Arrays, which is useful for importing data when you are building a C program. You'll see the packet formatted in C style, as shown in the figure below.

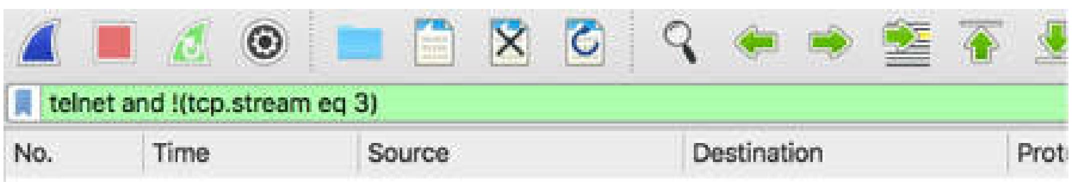


Task 7:

Click “Filter out this stream” to select all packets in the capture files except the previously analyzed stream, as shown in the figure below.



Wireshark creates and applies a new display filter, excluding the previous TCP stream. In this case, there are no other packets in the capture file.



Notes:

To gain more confidence in using the streams navigation window, repeat the previous tasks for other types of protocol streams like UDP and SSL. Try also to save and export the streams in a format different from ASCII or C arrays to understand different formatting possibilities.

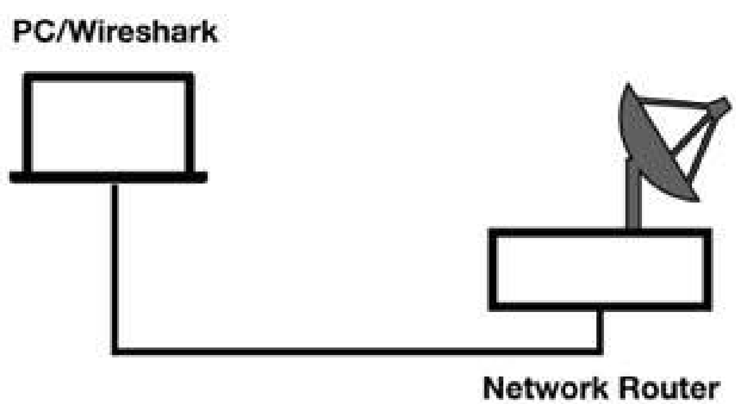
Lab 26. Profiles

Lab Objective:
Learn how to manage Wireshark profiles.

Lab Purpose:
Understand how to create and share profiles to customize Wireshark. Each profile can include configuration files related to preferences, capture filters, display filters, coloring rules, disabled protocols, and user accessible tables.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

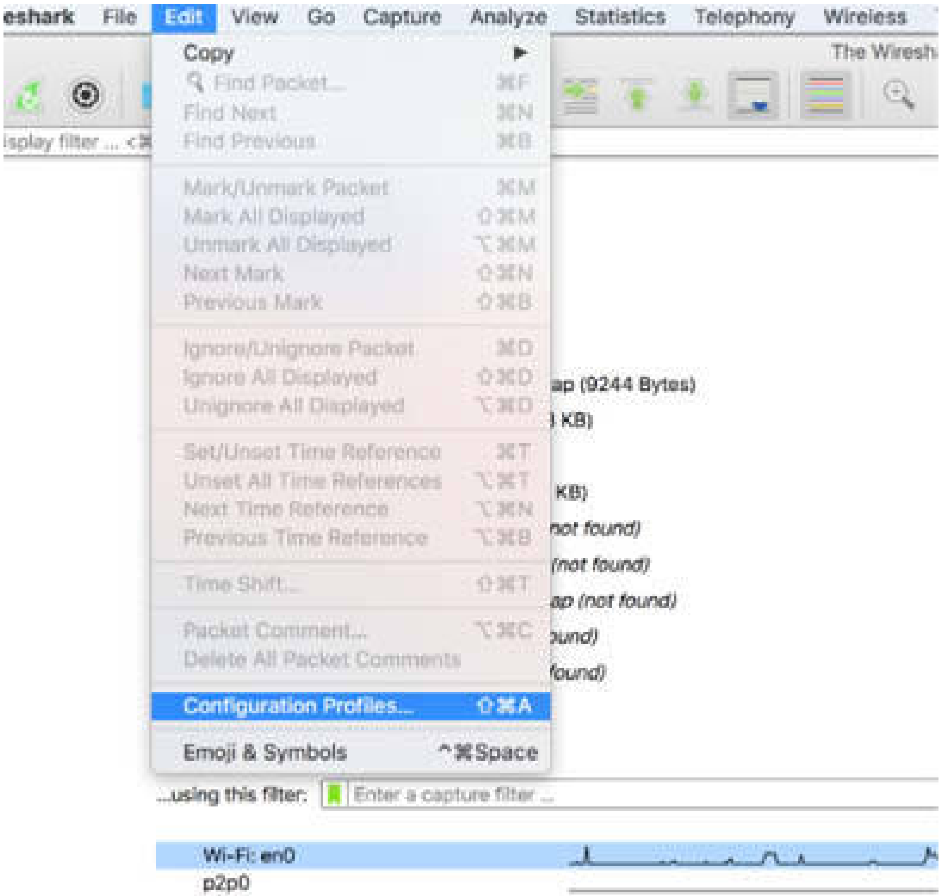
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



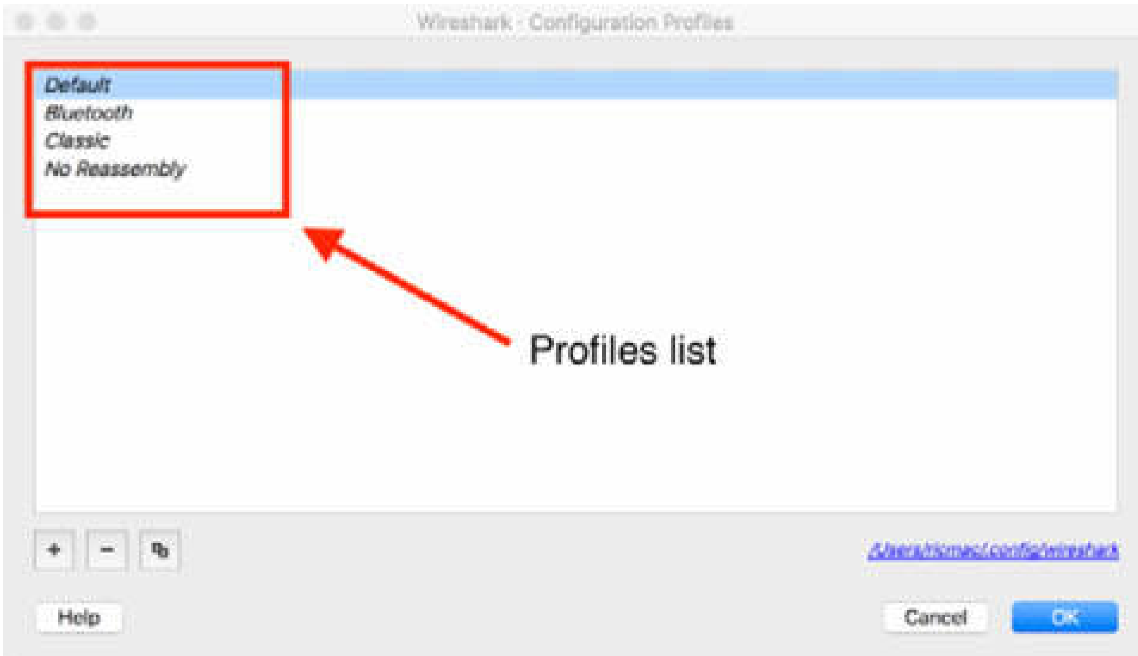
Lab Walkthrough:

Task 1:

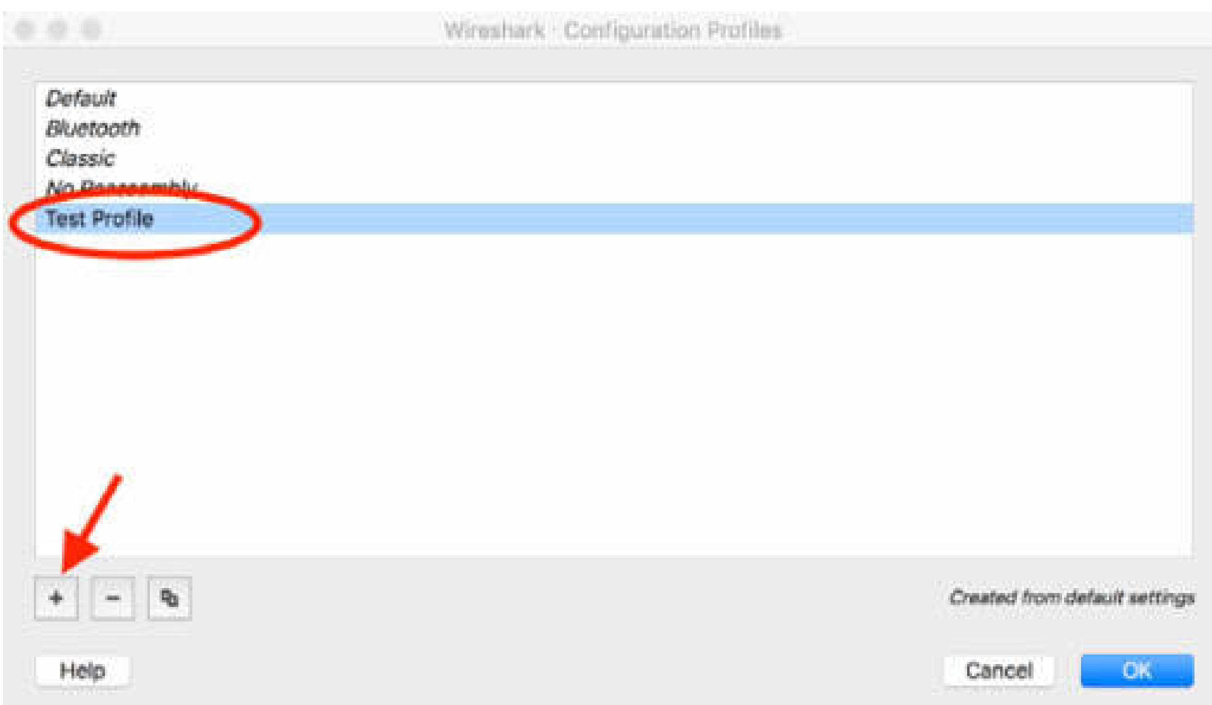
Open Wireshark, and on the main menu, select Edit > Configuration Profiles.



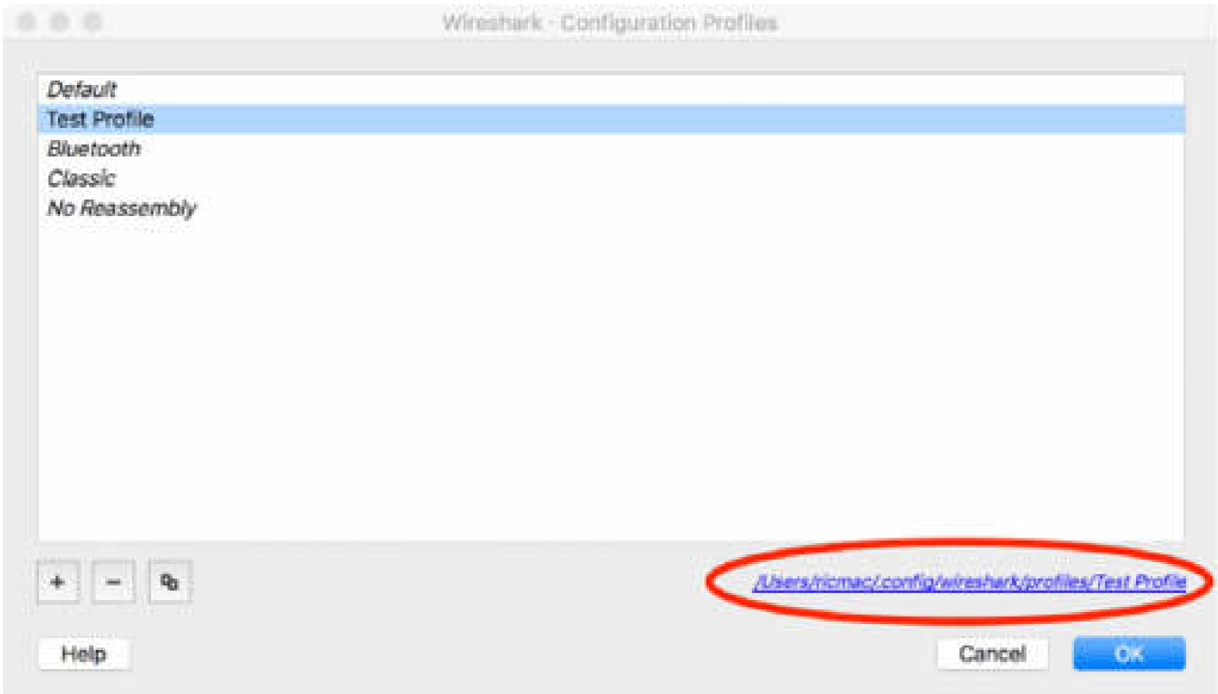
The Configuration Profile dialog box is displayed, listing all the current profiles.



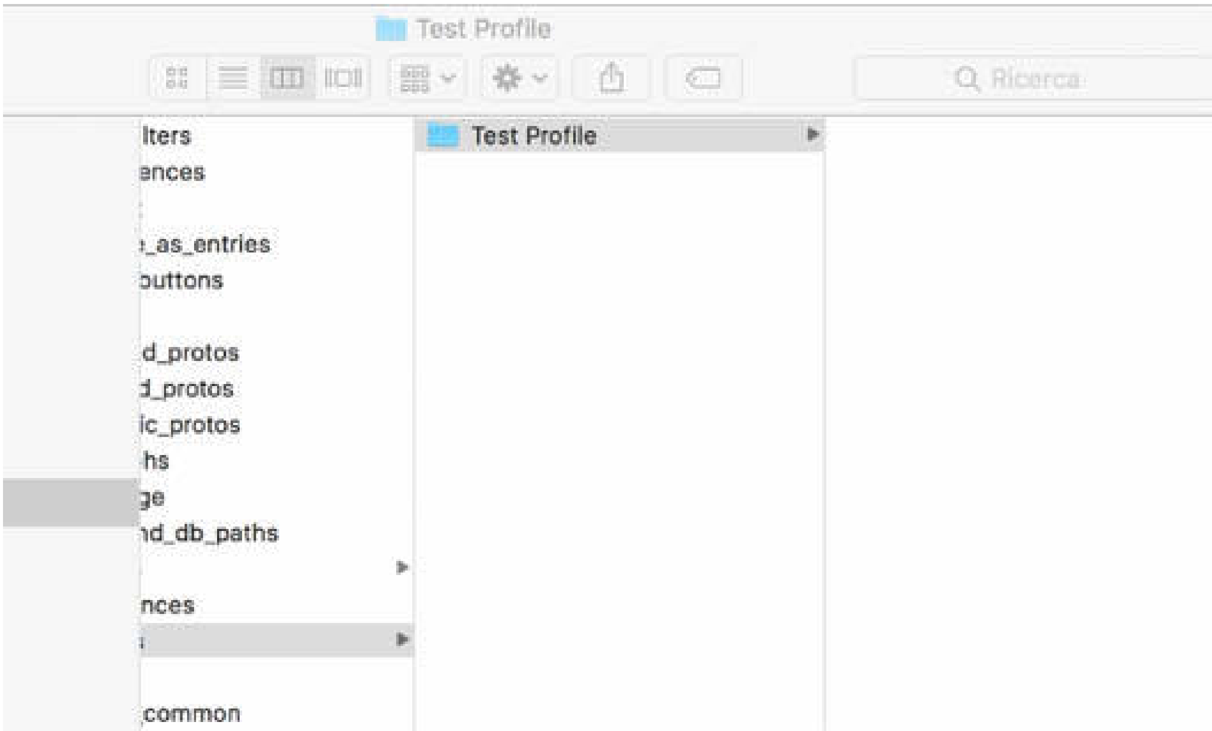
Task 2:
Click the add (+) button to create a new profile. Name it as “Test Profile”, and click OK. The new profile is inserted in the list.



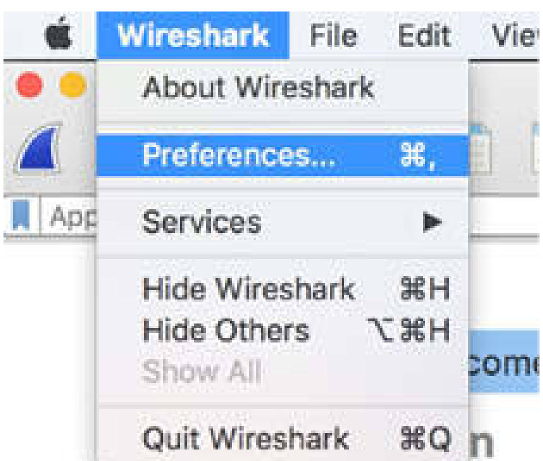
Task 3:
On the main menu, Select Edit > Configuration Profiles to open the Configuration Profiles dialog box. Select "Test Profile". A link to the profile location (the folder containing the profile file) is displayed, as shown in the figure below.



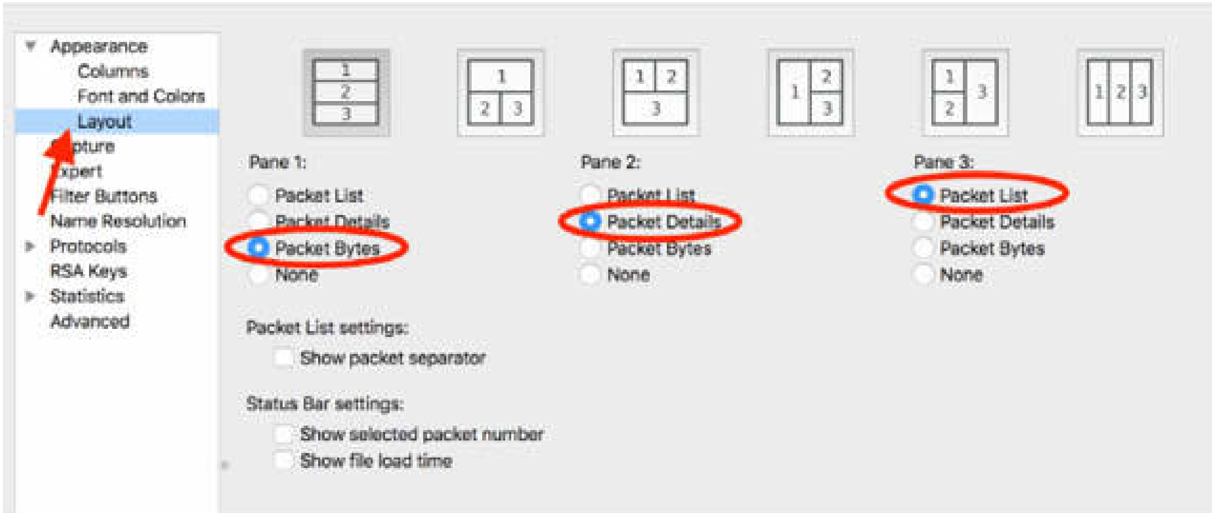
Click the profile location link. The profile folder is opened in file explorer. Because you have not done any customization to Wireshark, the profile folder is empty.



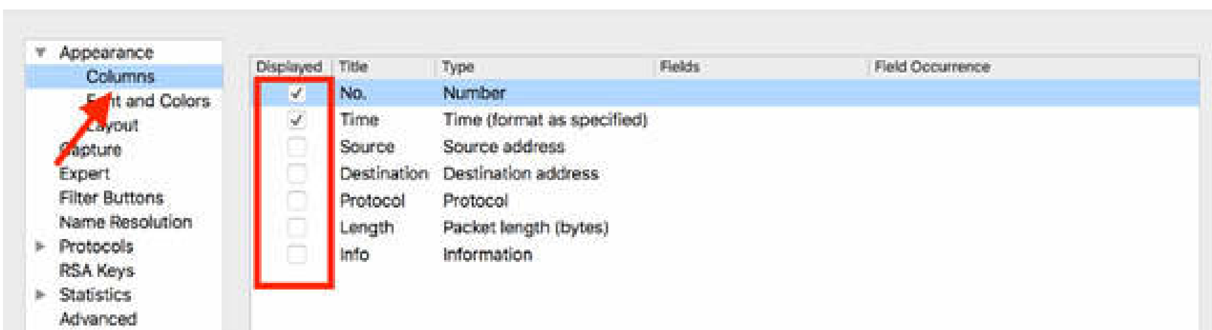
Task 4:
On the main menu, select Edit > Preferences to change the preferences to be applied to the “Test Profile” profile.



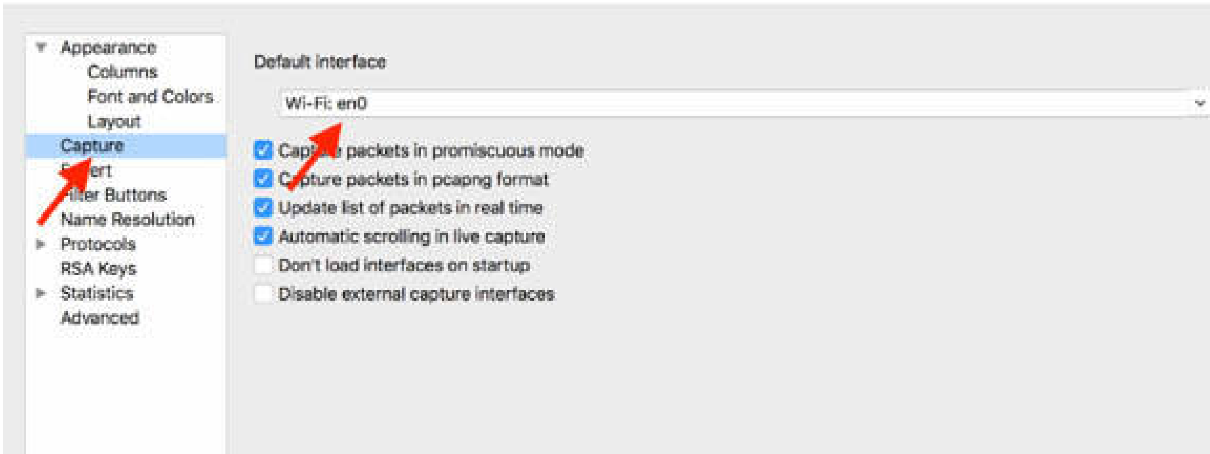
The Preferences dialog box is displayed. To change the main layout of Wireshark, click Layout and then change the selection of Pane 1, Pane 2, and Pane 3, as shown in the figure below.



To change the column view in the Packet List pane, click Columns and then match the check boxes to as shown in the figure below.

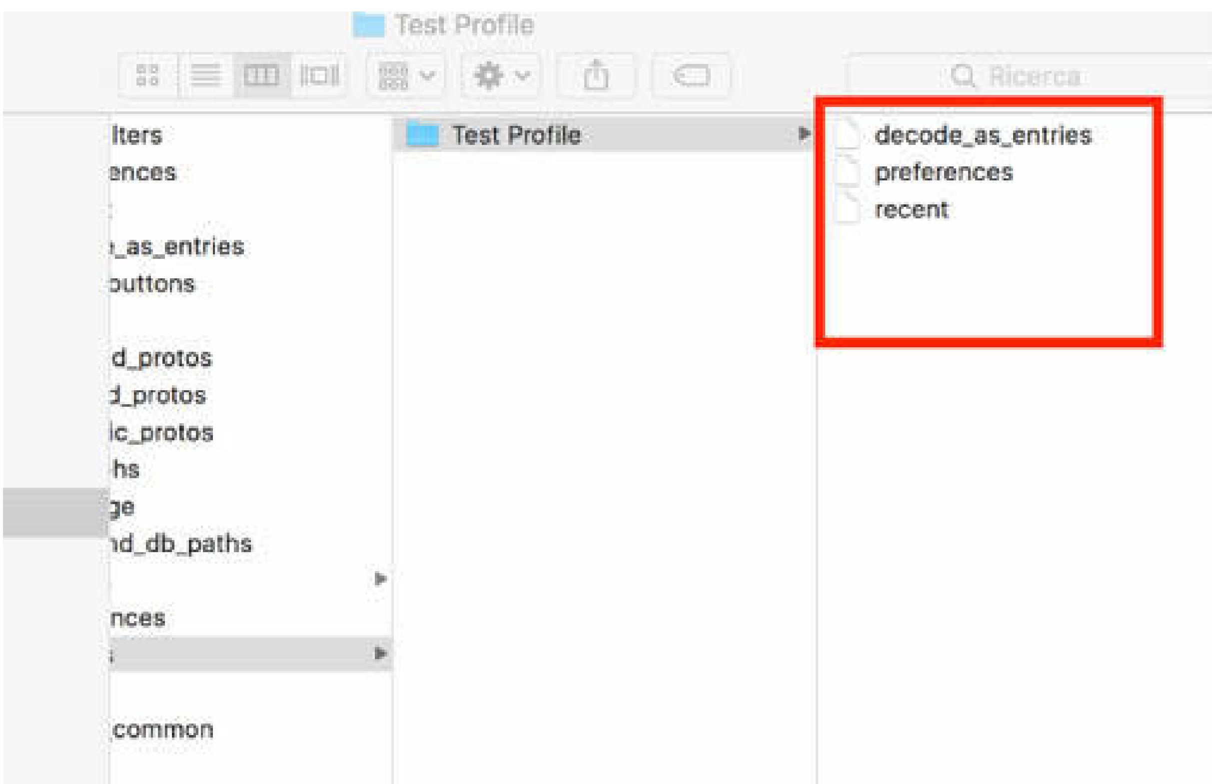


Select Capture, and then in the “Default interface” field, select the Wi-fi interface, as shown in the figure below (yours will differ from ours).

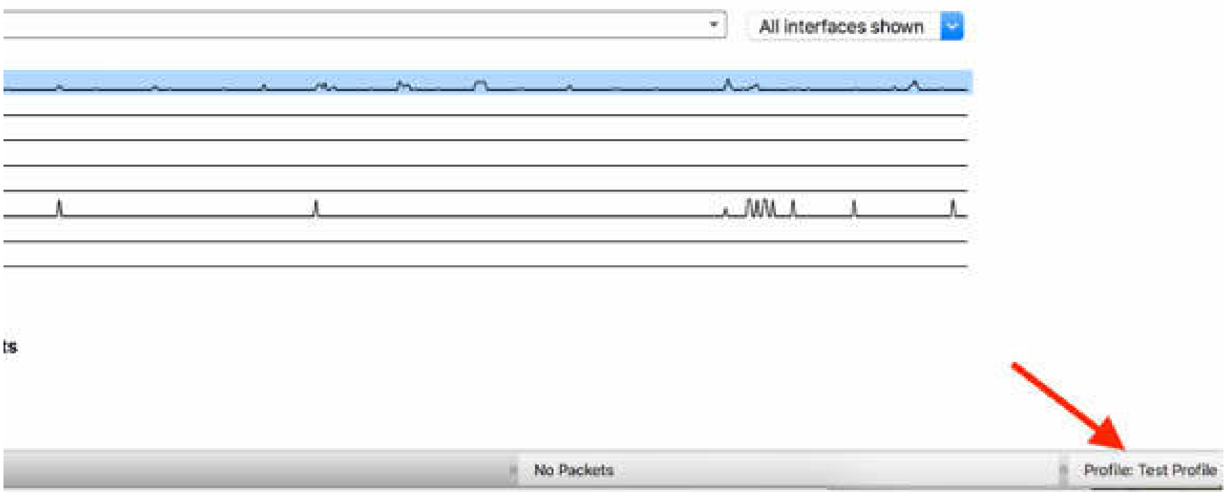


Click OK.

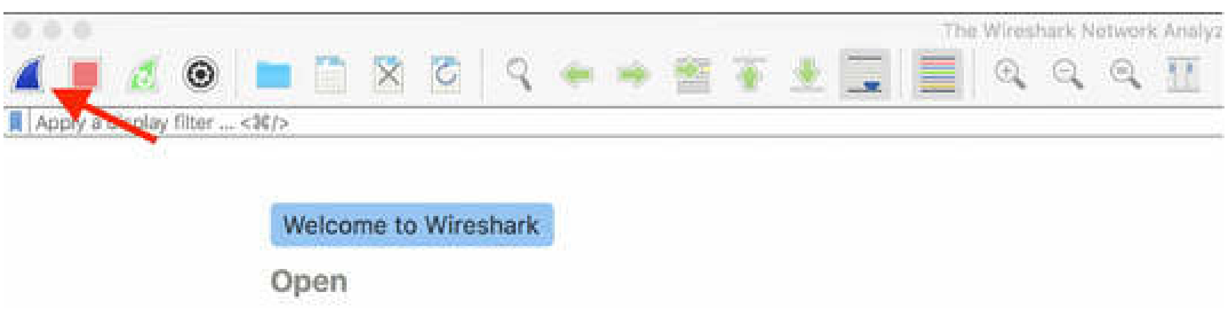
Task 5:
On the main menu, Select Edit > Configuration Profiles to open the Configuration Profile dialog box. Click the profile location link to open the profile folder in file explorer, showing the configuration files for "Test Profile".



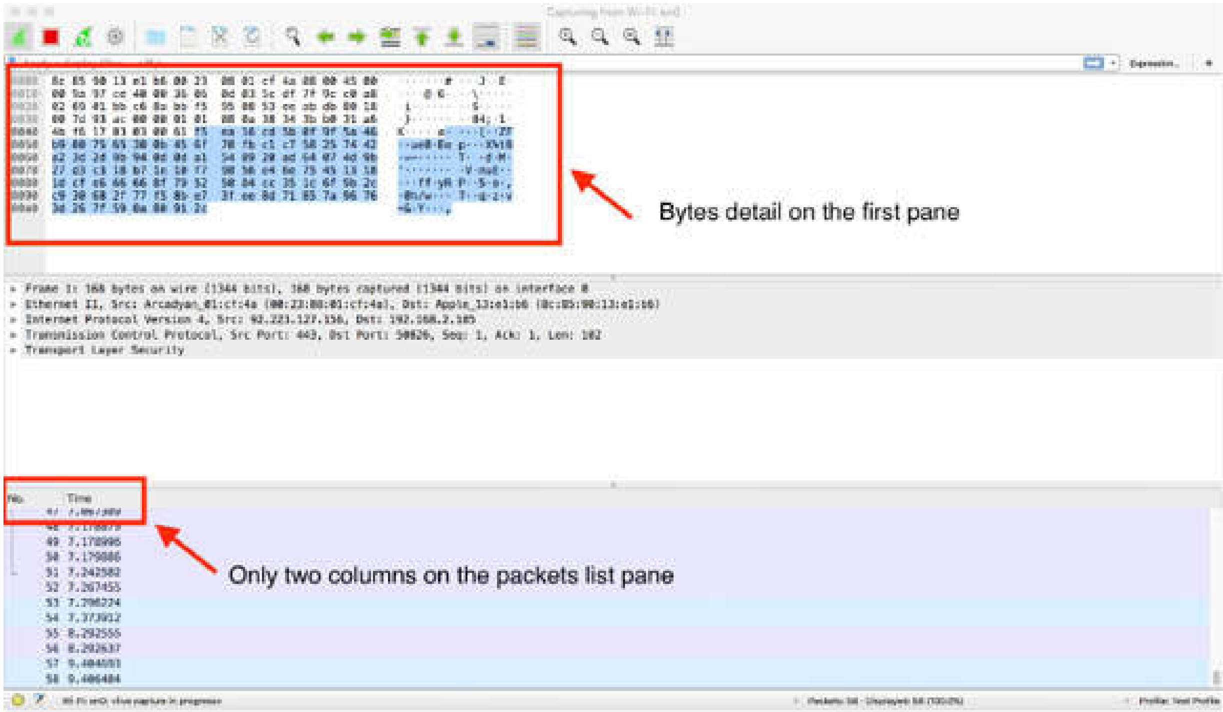
To ensure that the actual profile is “Test Profile”, inspect the profile name displayed at the bottom right corner of the Wireshark main window.



Task 6:
Start a live capture on the default interface by clicking the Start button shown in the figure below.

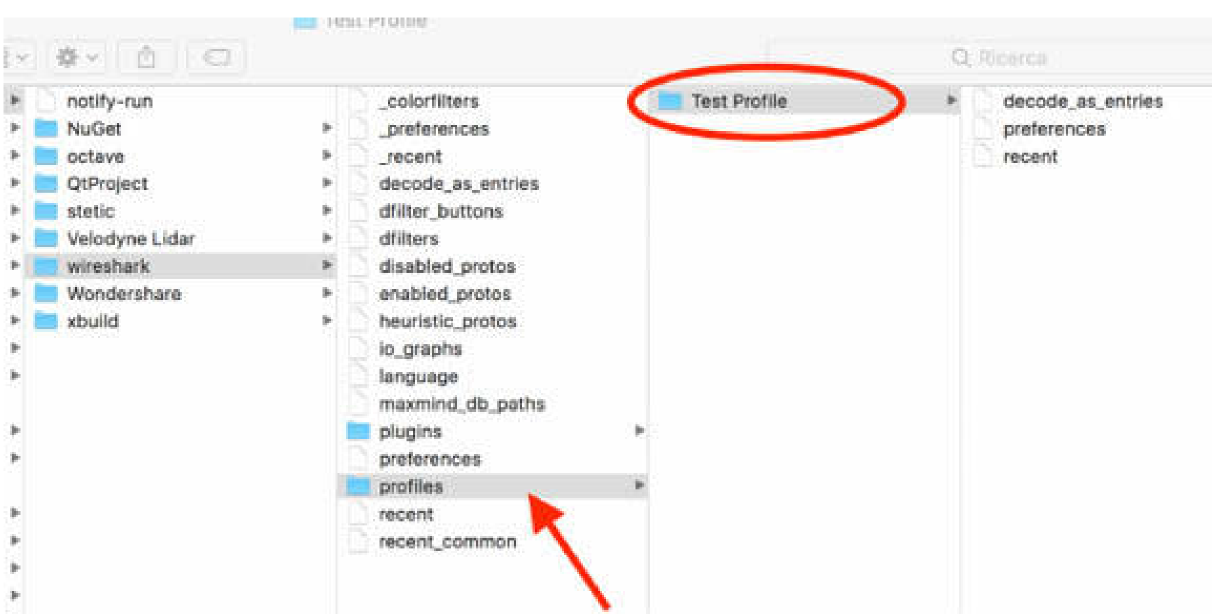


You can see the newly-created customization on the Wireshark profile.

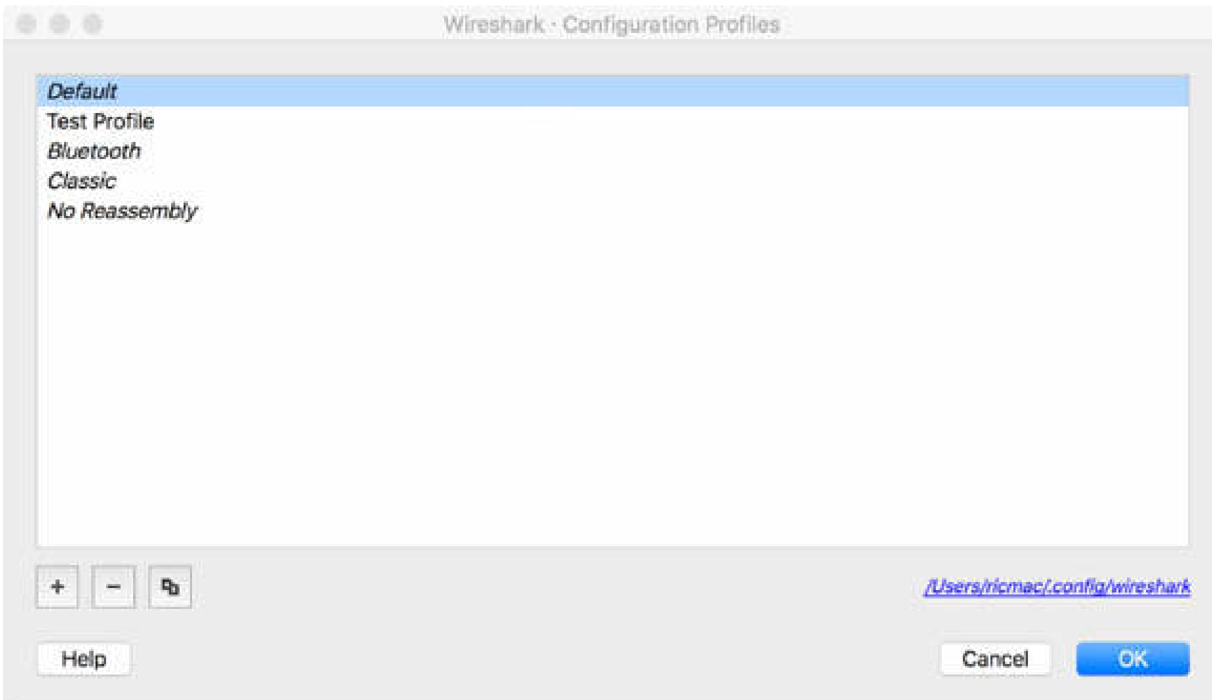


Task 7:

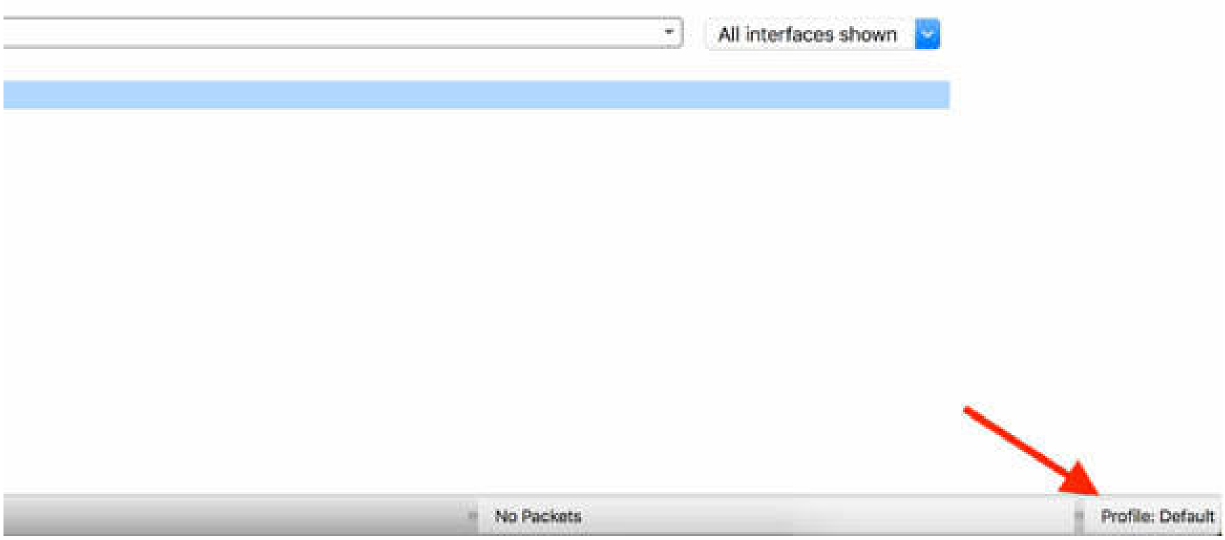
To export (or import) a profile, in the Wireshark profiles folder, you can compress the folder named as the profile name and then copy it.



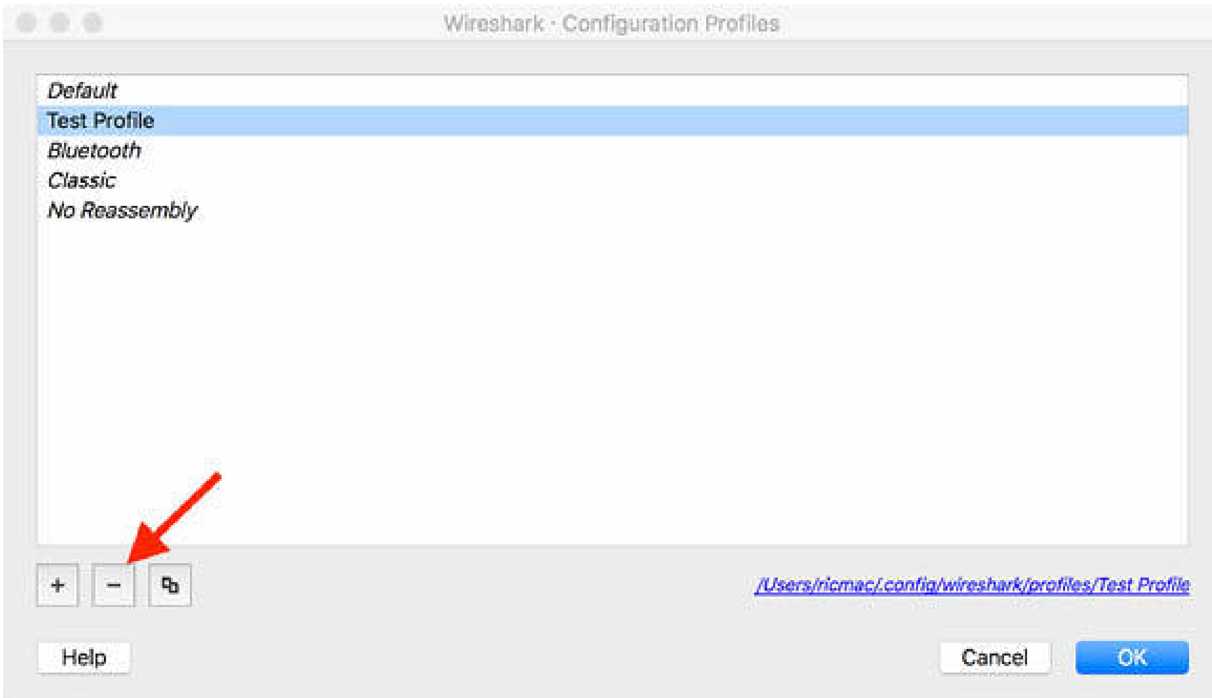
Task 8:
To switch back to the default profile, open the Configuration Profiles dialog box and select the default profile, and click OK.



The default profile is enabled.



To remove the "Test Profile" profile, click the remove (–) button in the Configuration Profiles dialog box, and the initial profile list is restored.



Notes:
To gain more confidence in using profiles sharing, repeat the previous tasks for creating new profiles, and customizing them. Try to save and export profiles and then import them on a different PC.

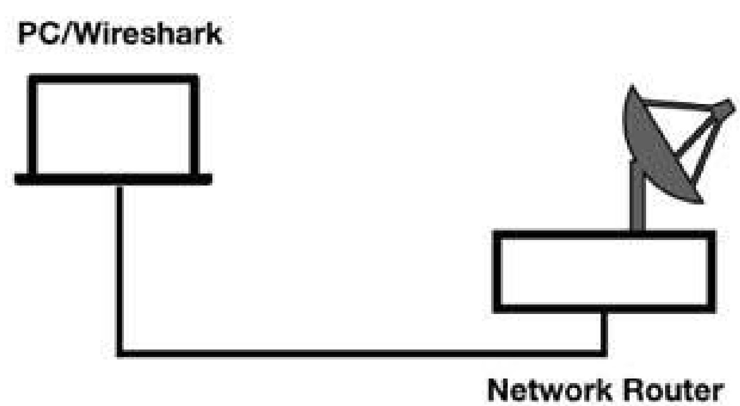
Lab 27. Annotation and Save Functionality

Lab Objective:
Learn how to use the annotation feature and save functionality.

Lab Purpose:
Wireshark allows you to annotate a single packet or an entire trace file. Moreover, you can export some packets to verify specified features.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

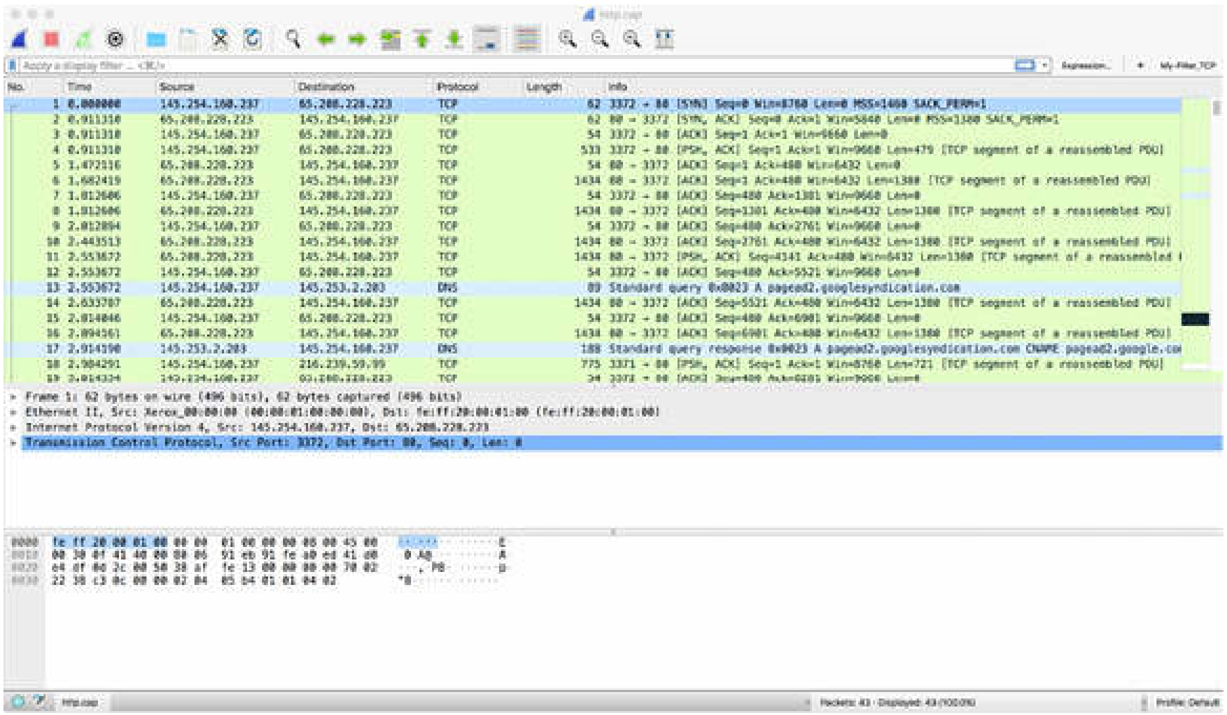
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

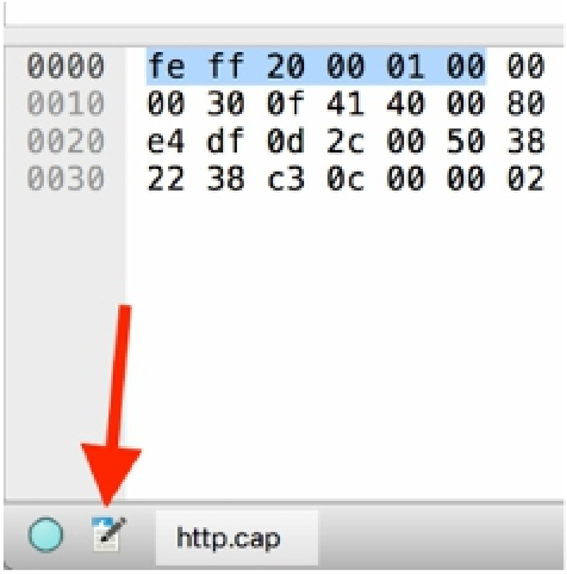
Task 1:

Download the free sample capture http.cap to your PC from https://wiki.wireshark.org/SampleCaptures#TCP , and then open the downloaded file in Wireshark.

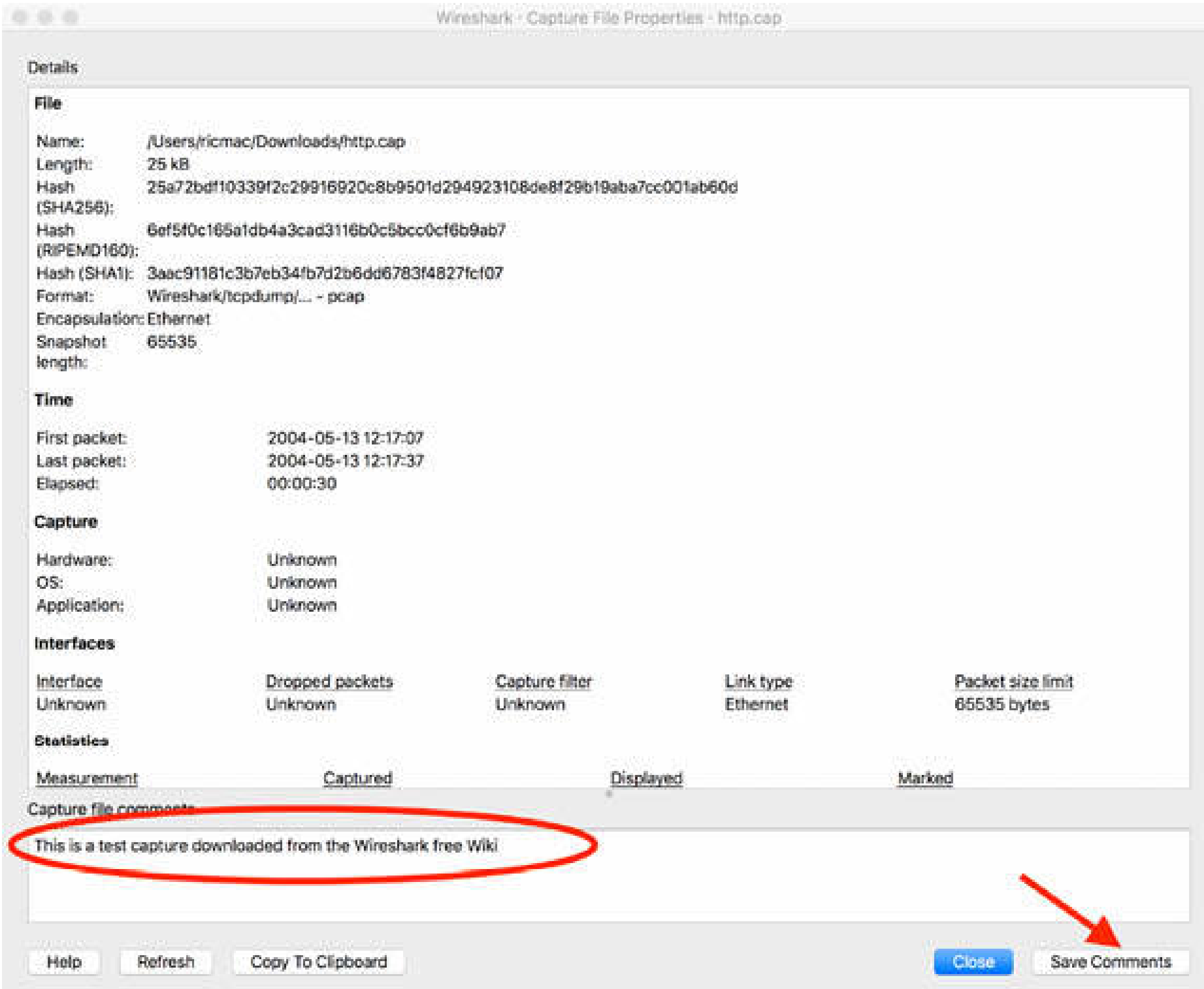


Task 2:

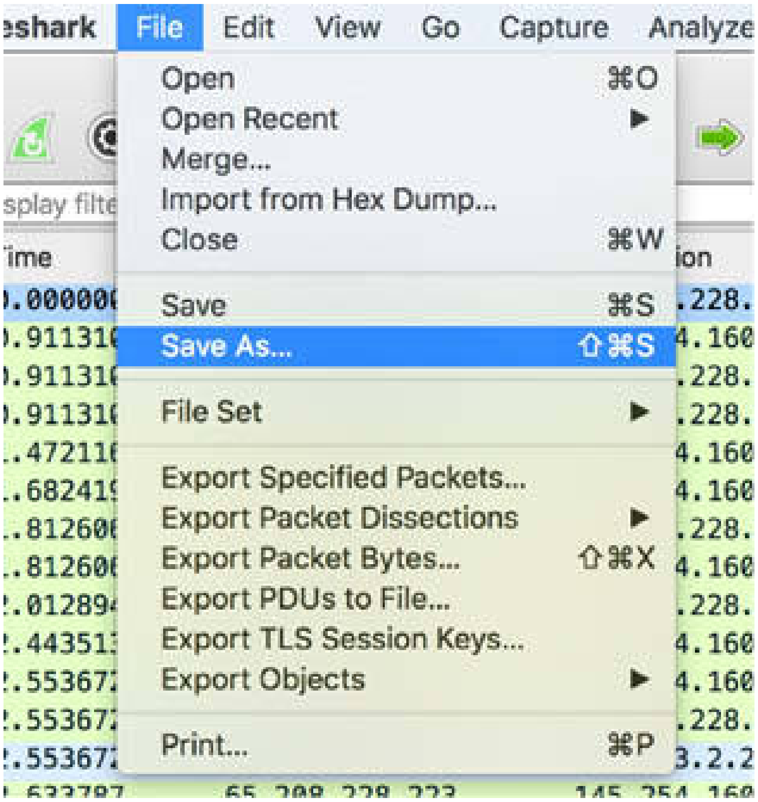
In the bottom left corner of the main window, click the pencil icon to apply an annotation to the entire file, as shown in the figure below.



The Capture File Properties dialog box is displayed. Add your comment in the appropriate pane and then click Save Comments, and close the dialog box.

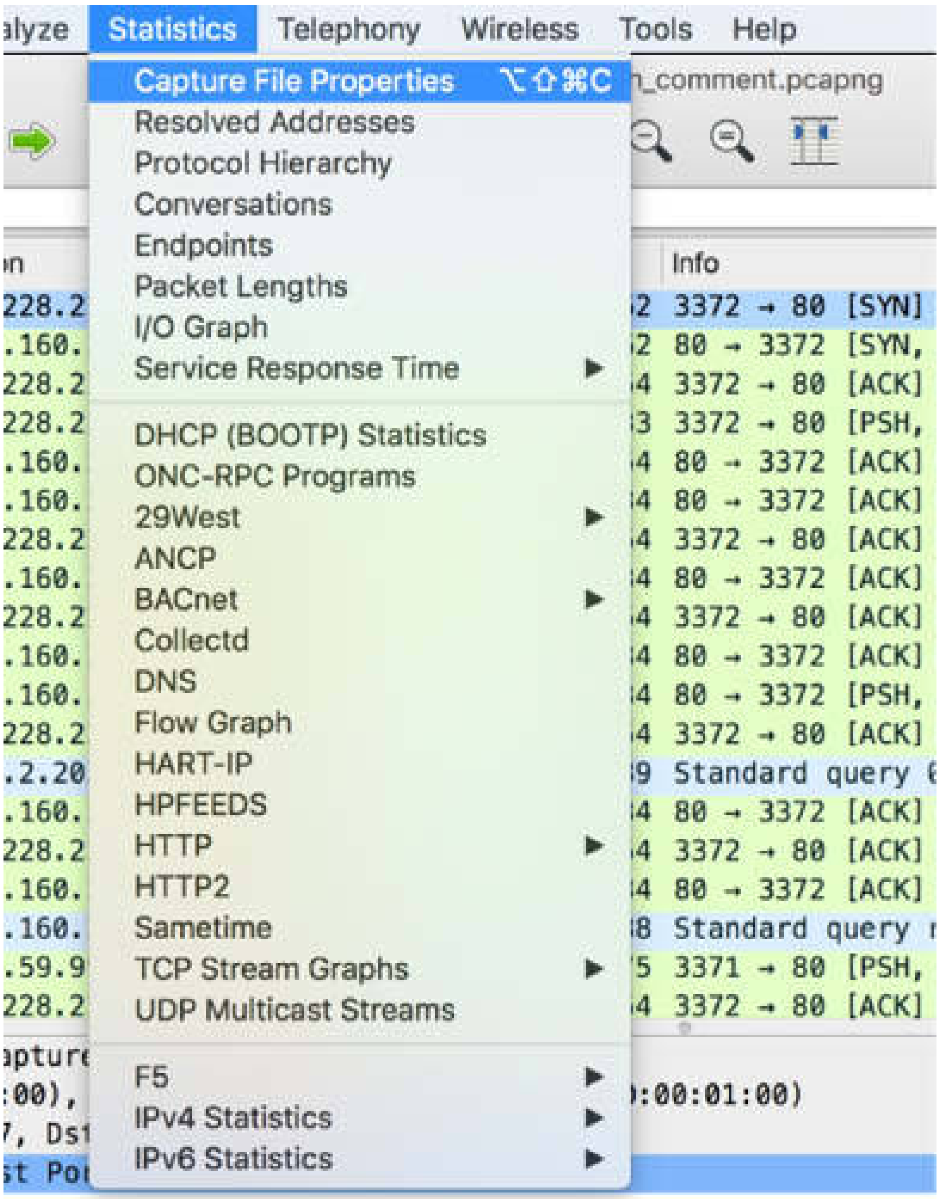


On the main menu, select File > Save as to save the trace file with the name “http_with_comment” and format “pcapng”.

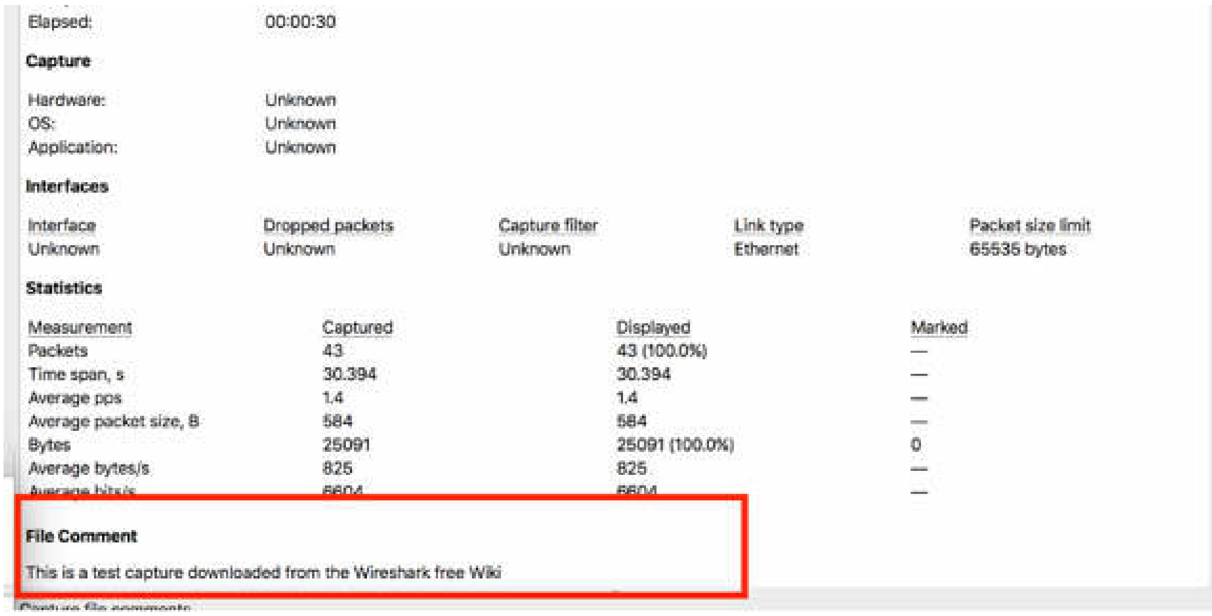


Close the file.

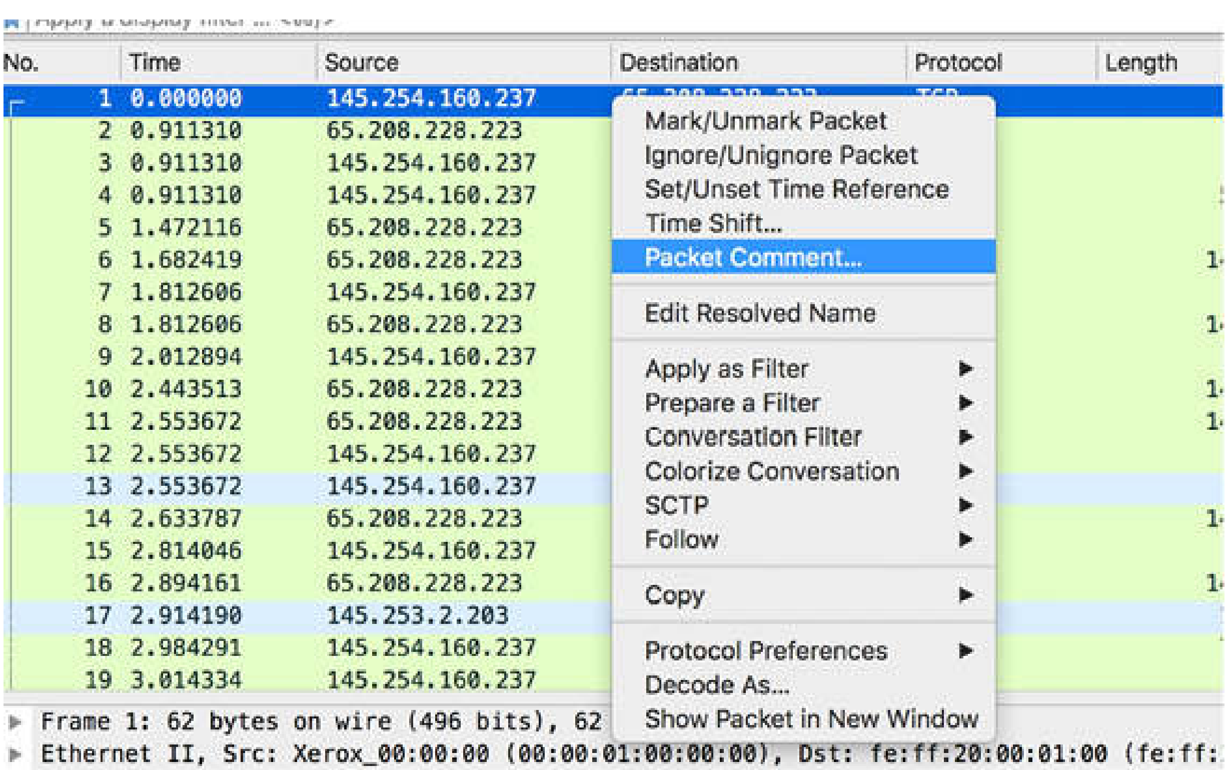
Task 2:
Open the just saved file http_with_comment.pcapng, and on the main menu, select Statistics > Capture File Properties.



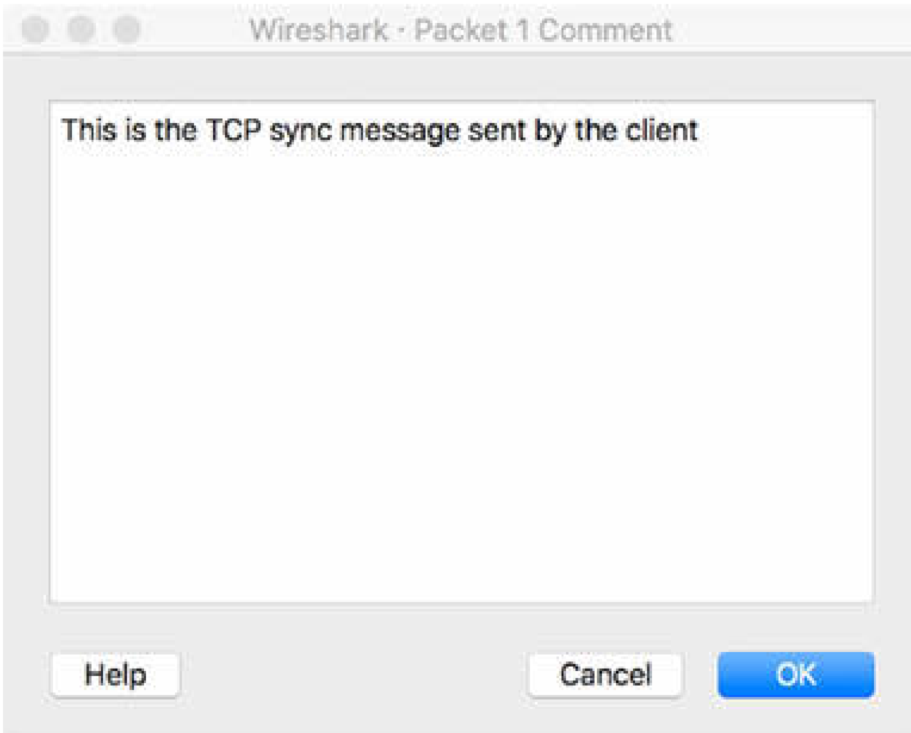
The Statistics dialog box is displayed displaying the File Comment, as shown in the figure below.



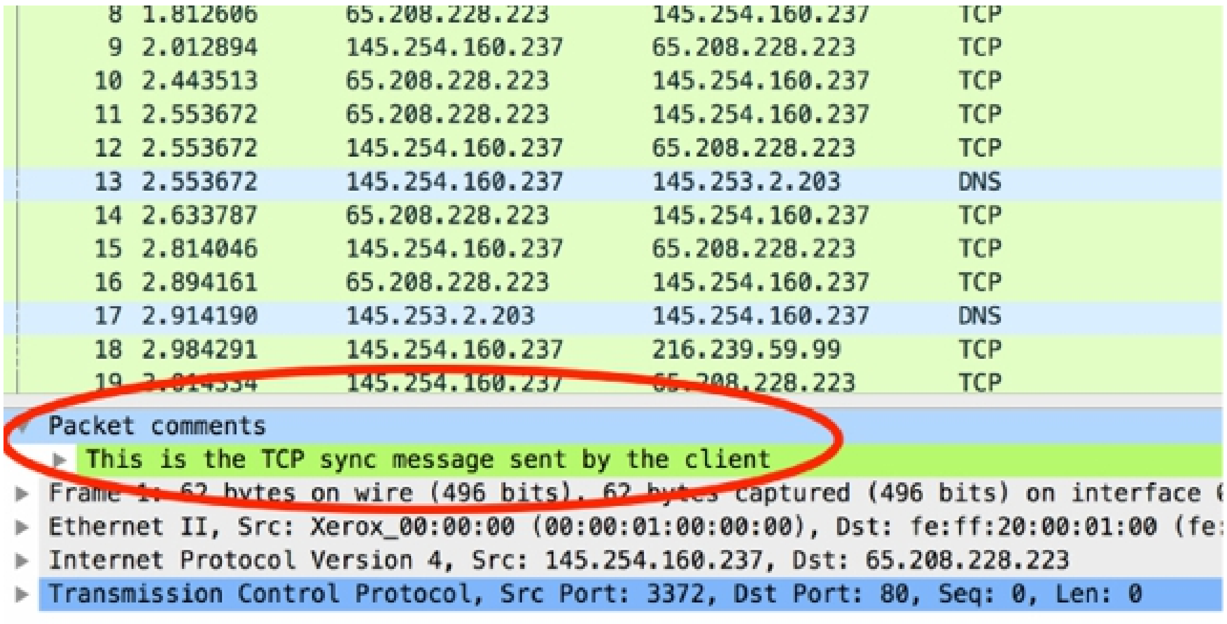
Task 3:
To annotate a single packet, in the Packet List pane, right-click the first packet, and select Packet Comment.



The Packet Comment dialog box is displayed. Add your comment in the appropriate field, and click OK.

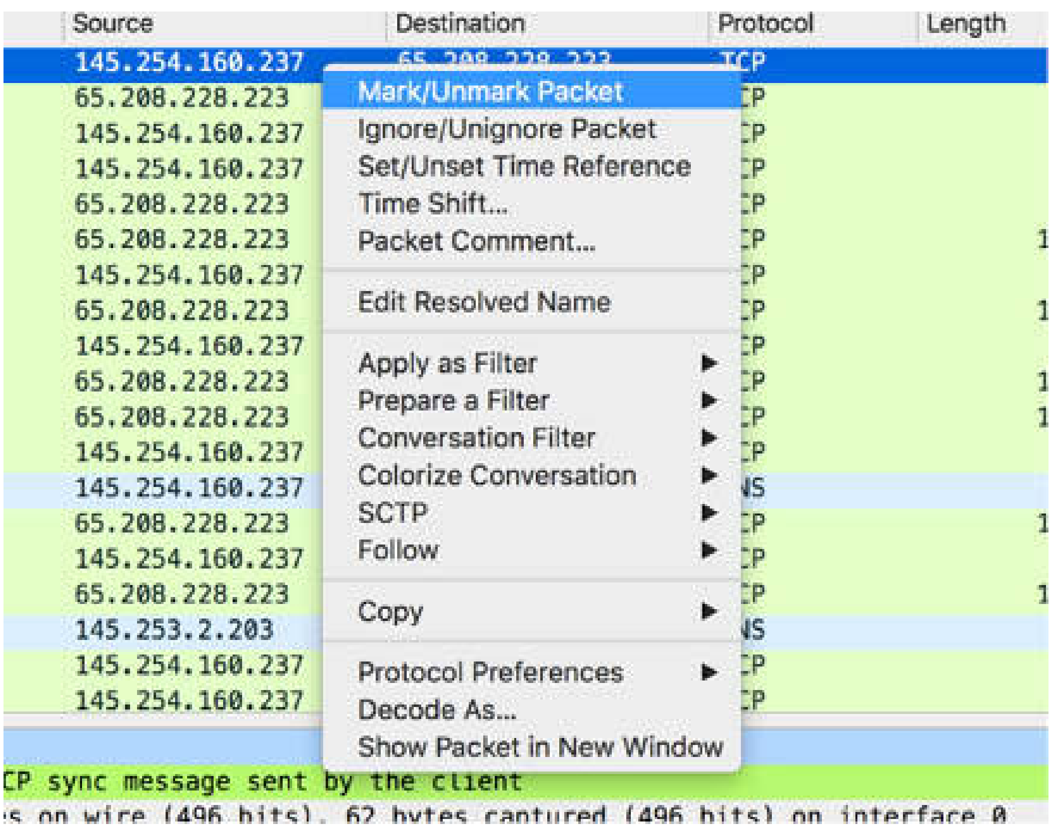


The just-added comment is displayed and highlighted in the Packet Details pane, as shown in the figure below.

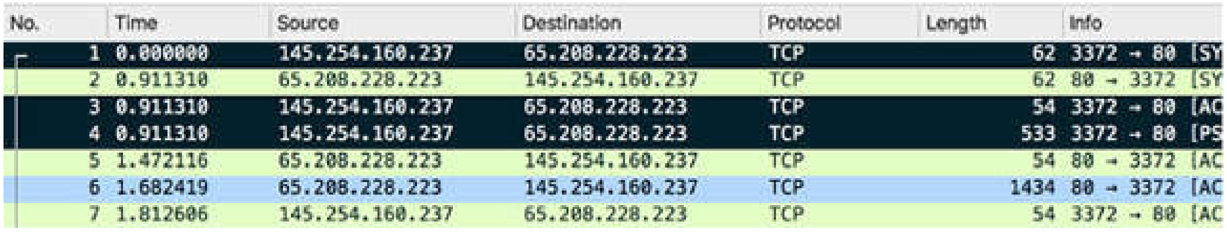


Task 4:

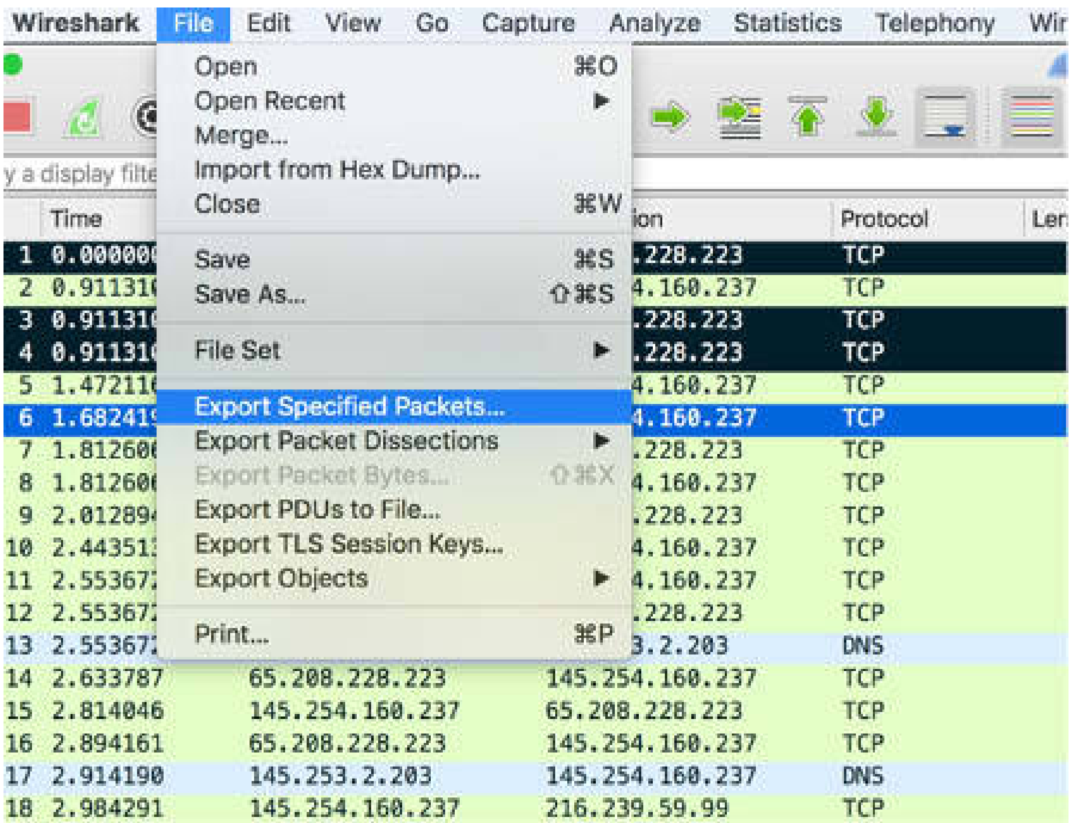
Select packets #1, #3, and #4 and then right-click and select Mark/Unmark Packet to mark the selected packets.



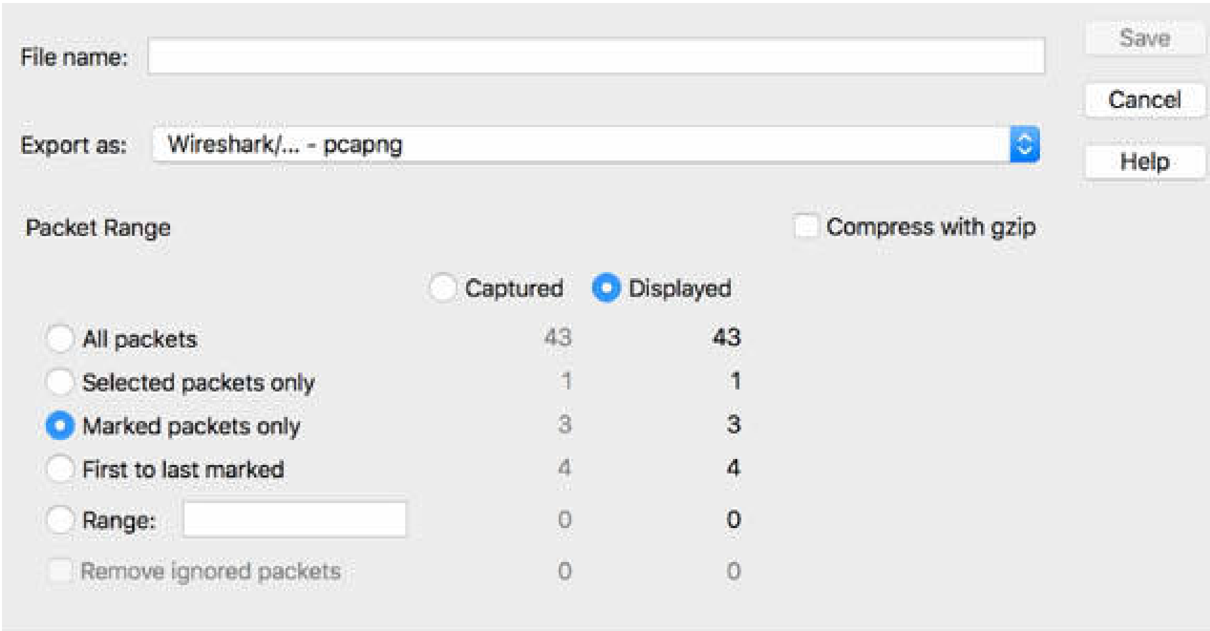
In the Packet List pane, packets #1, #3, and #4 are displayed as marked, as shown in the figure below.



To save only the desired packets, on the main menu, select File > Export Specified Packets.

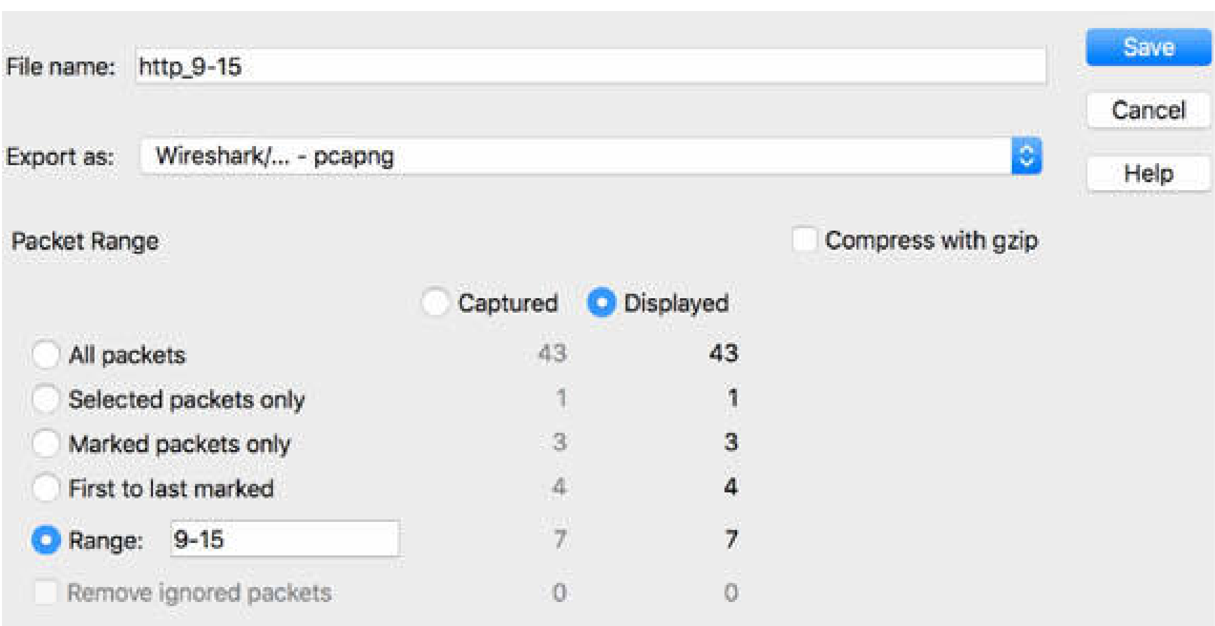


The Export Specified Packets dialog box is displayed. Enter a name for the exported file and select the “Marked packets only” option. The exported file will contain only the marked packets.

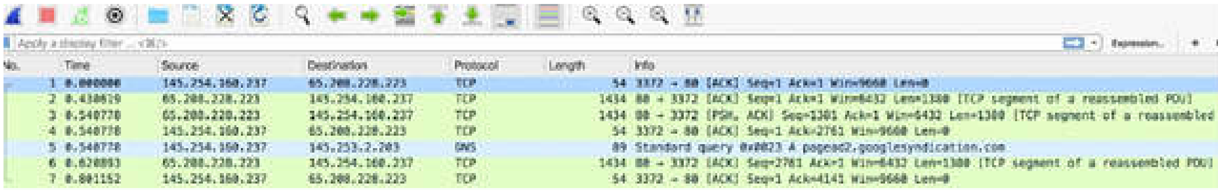


Task 5:

To export a predefined range of packets, on the main menu, select File > Export Specified Packets. In the Export Specified Packets dialog box, select the Range option and enter 9–15 as the range for the packets to be exported.



Open the exported file. In the Packet List pane, only the desired range is displayed as a new capture.



Notes:

To gain more confidence in working with export packets, create annotations on different packets and export marked packets or packets range. Explore all export options and specifically pay attention to the difference between the Captured and Displayed options. The Captured option applies to the entire capture file; the Displayed option is applicable when a display filter is in place and the displayed packets list is a subset of the entire trace.

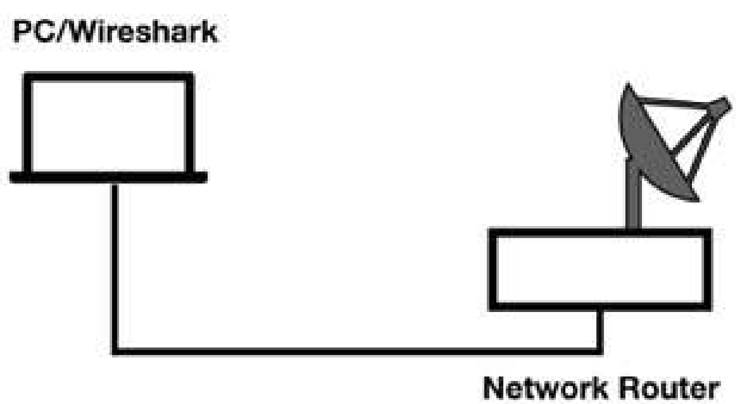
Lab 28. Export Menu

Lab Objective:
Learn how to use the export menu feature.

Lab Purpose:
Wireshark allows you to export the packets' content for use in other programs or to directly export the packet bytes data.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



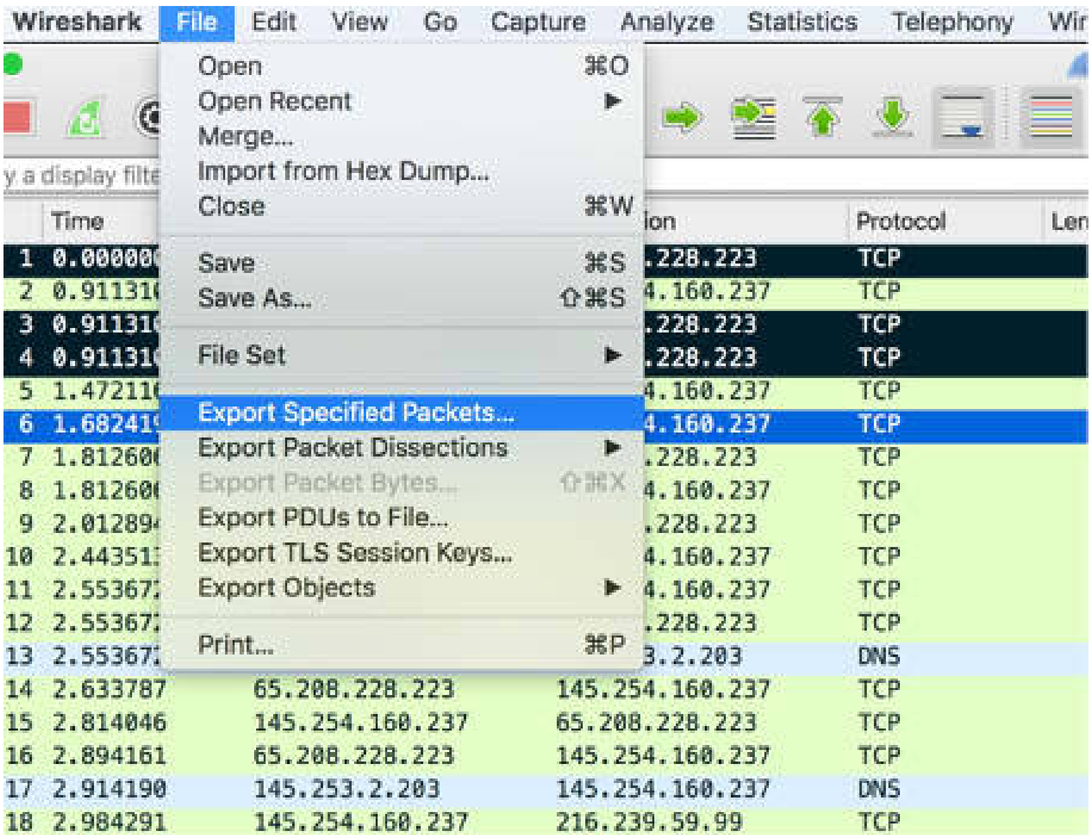
Lab Walkthrough:

Task 1:

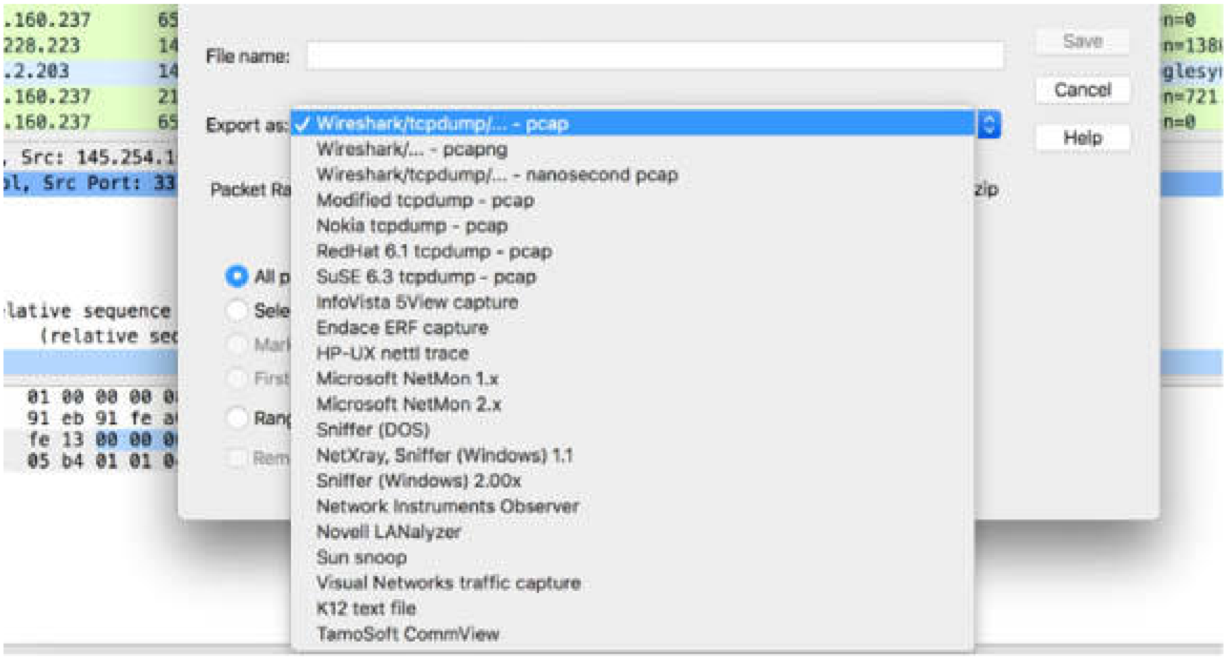
Download the free sample capture http.cap to your PC from https://wiki.wireshark.org/SampleCaptures#TCP , and then open the downloaded file in Wireshark.

Task 2:

On the main menu, select File > Export Specified Packets.

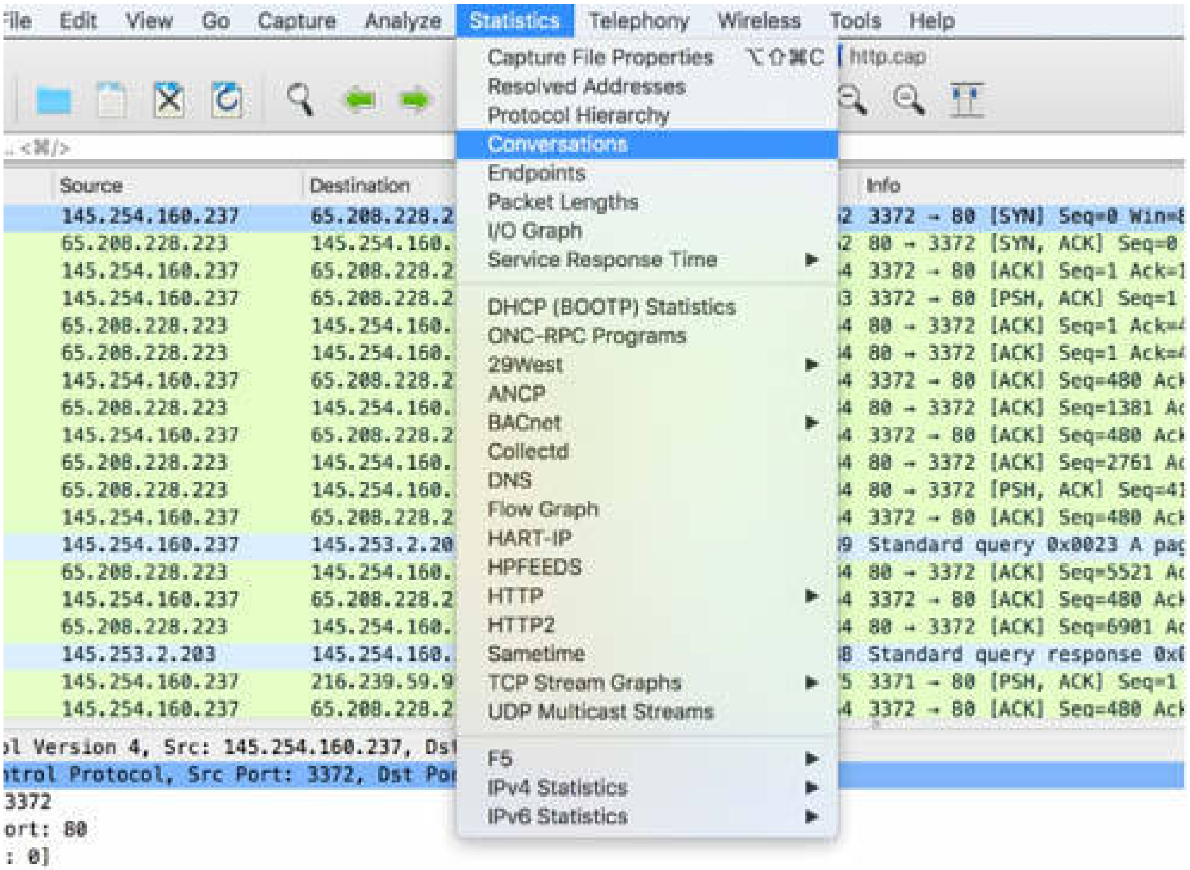


In the Export Specified Packets dialog box, select a file format specific to different programs. The list is shown in the figure below.

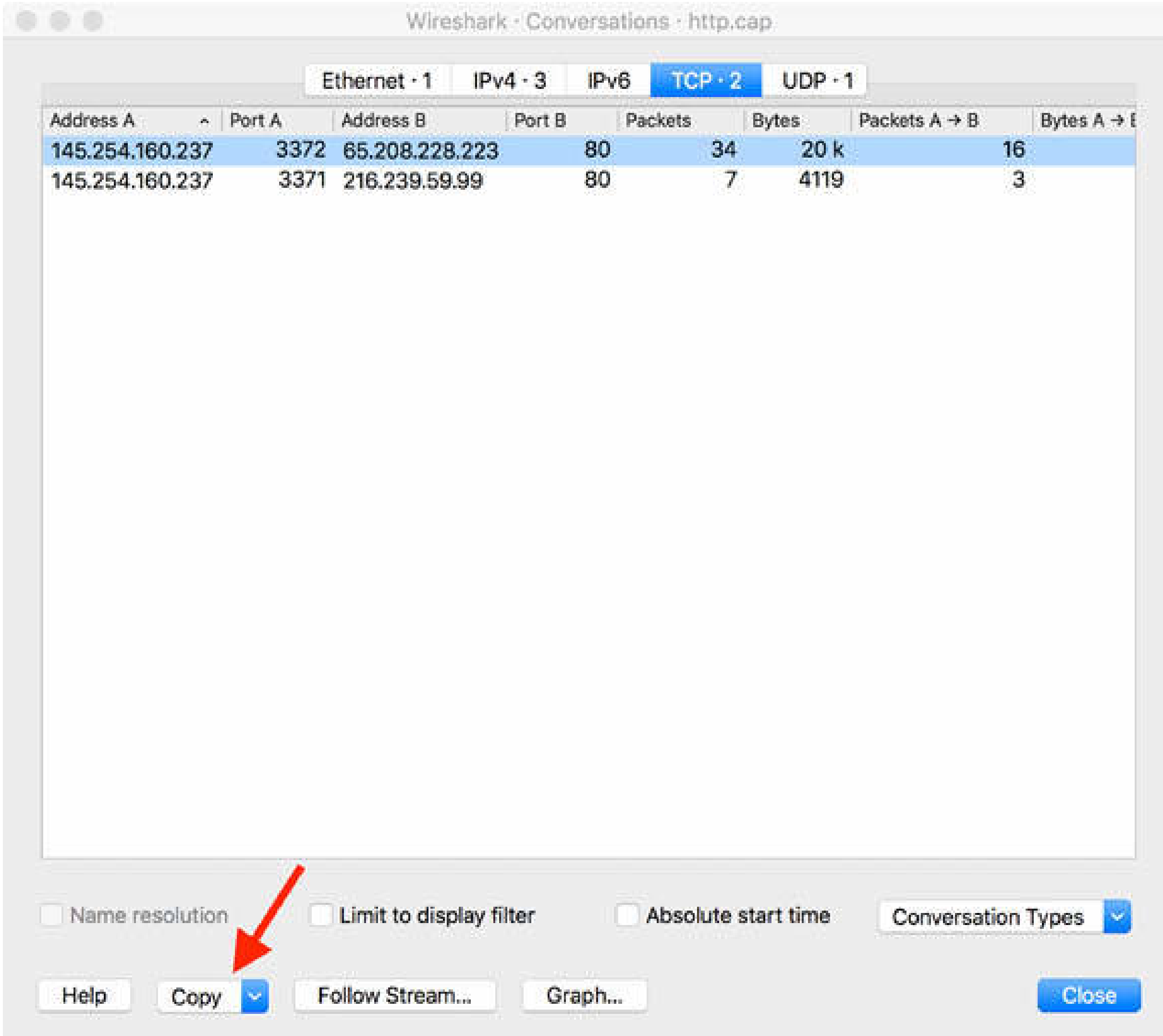


Task 3:

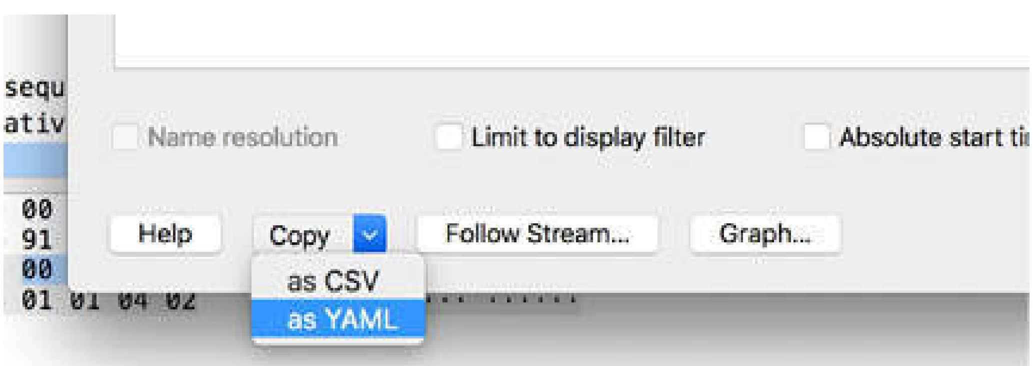
On the main menu, select Statistics > Conversations.



The Conversations dialog box is displayed, showing the actual conversations in the TCP tab. Select the first conversation, and click the down arrow next to the Copy button, shown in the figure below.



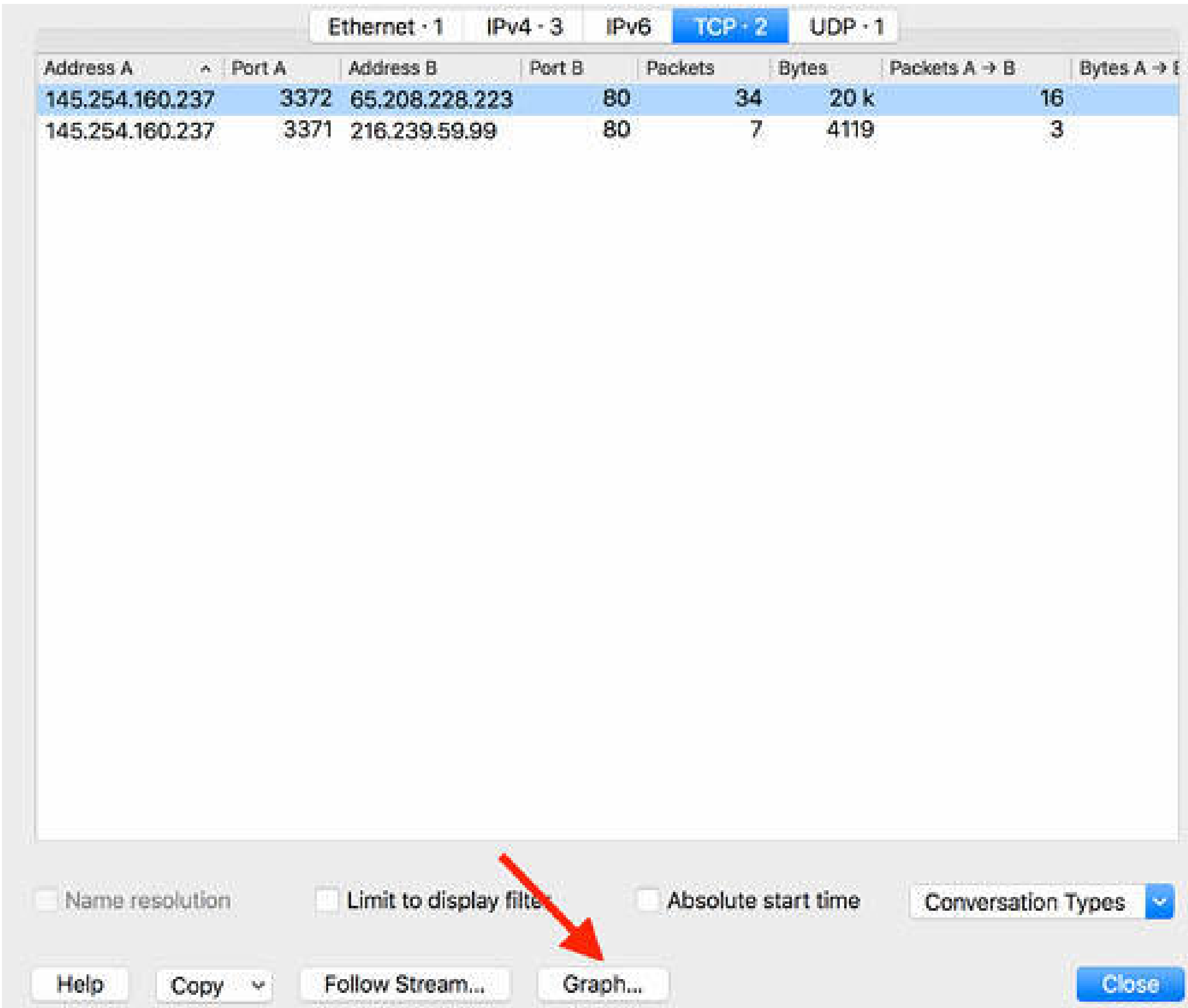
Select "As YAML" to copy the conversation in YAML format.



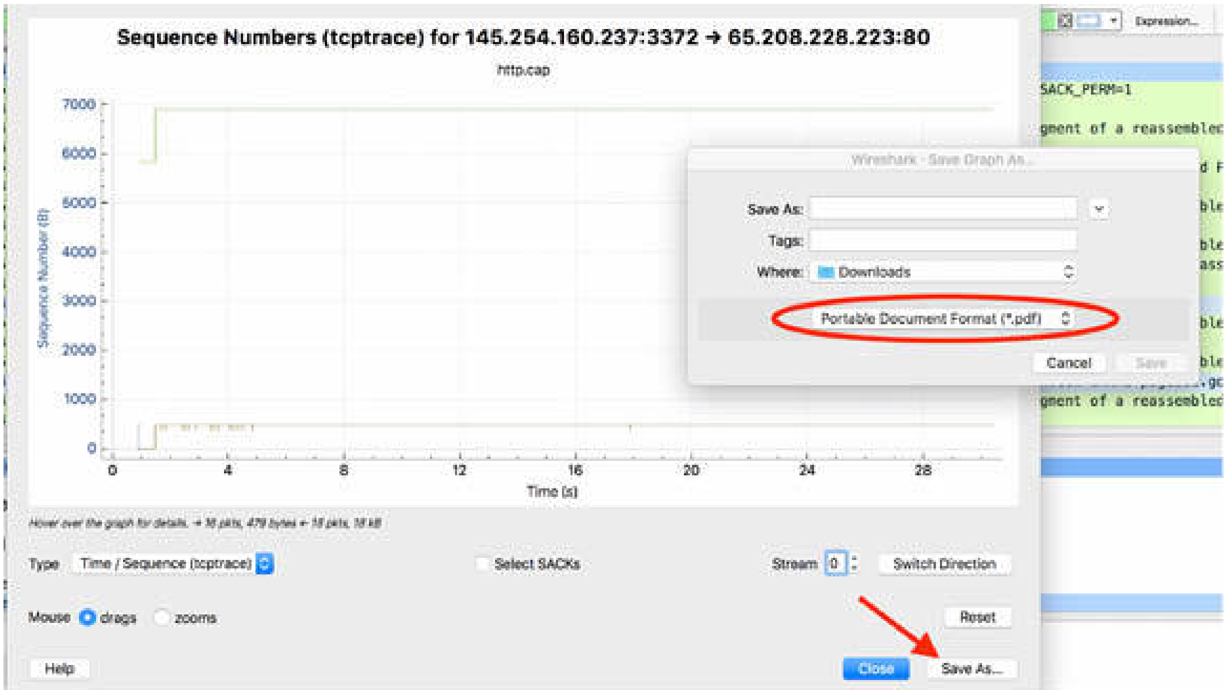
Open a text editor, and paste the contents of the clipboard, as shown in the figure below.

 - Address A
 - Port A
 - Address B
 - Port B
 - Packets
 - Bytes
 - Packets A → B
 - Bytes A → B
 - Packets B → A
 - Bytes B → A
 - Rel Start
 - Duration
 - Bits/s A → B
 - Bits/s B → A
-
 - 145.254.160.237
 - 3372
 - 65.208.228.223
 - 80
 - 34
 - 20695
 - 16
 - 1351
 - 18
 - 19344
 - 0
 - 30.393704
 - 355.5999624132682
 - 5091.580808972806
-
 - 145.254.160.237
 - 3371
 - 216.239.59.99
 - 80
 - 7
 - 4119
 - 3
 - 883
 - 4
 - 3236
 - 2.984291
 - 1.7925770000000005
 - 3940.695434561527
 - 14441.778512164326

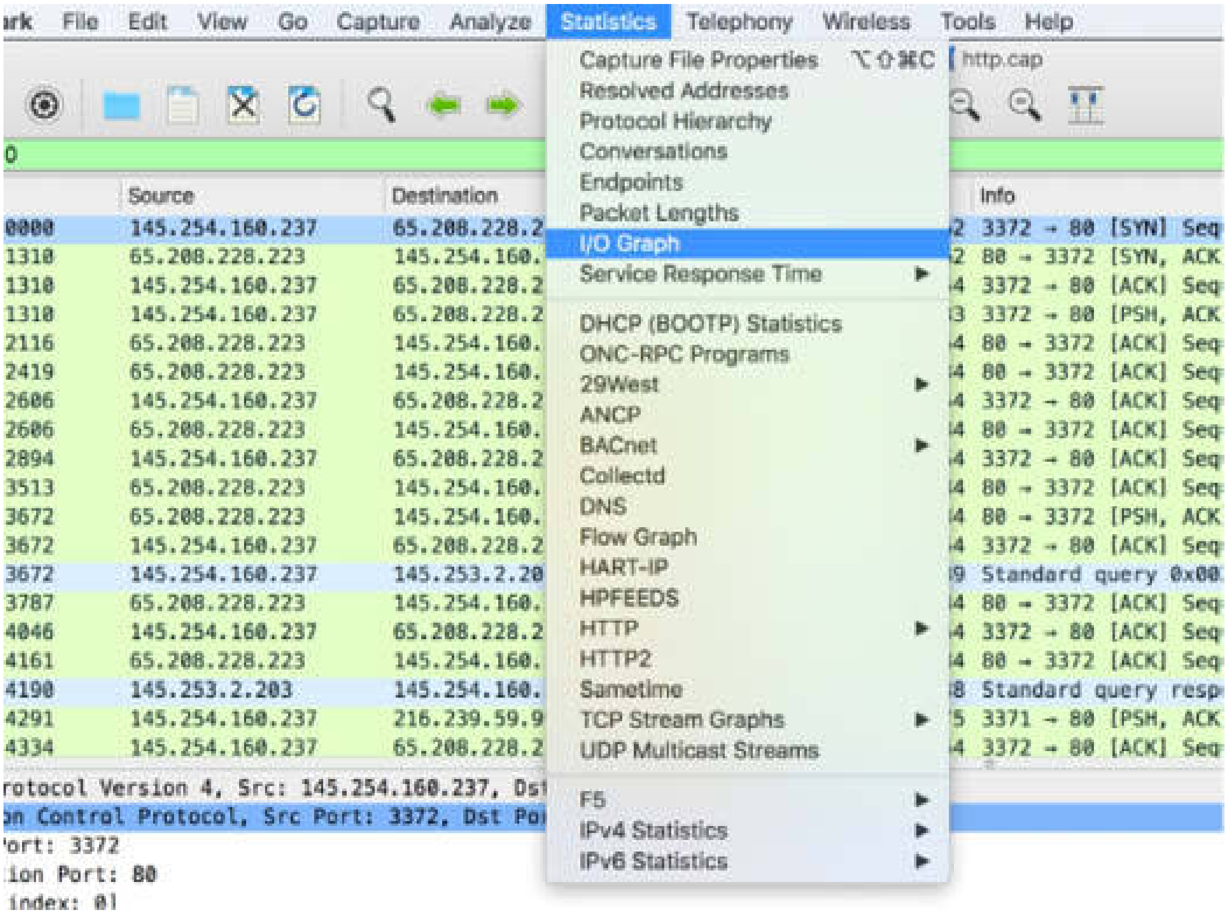
Task 4:
On the main menu, select Statistics > Conversations. In the Conversations dialog box, click the Graph button shown in the figure below.



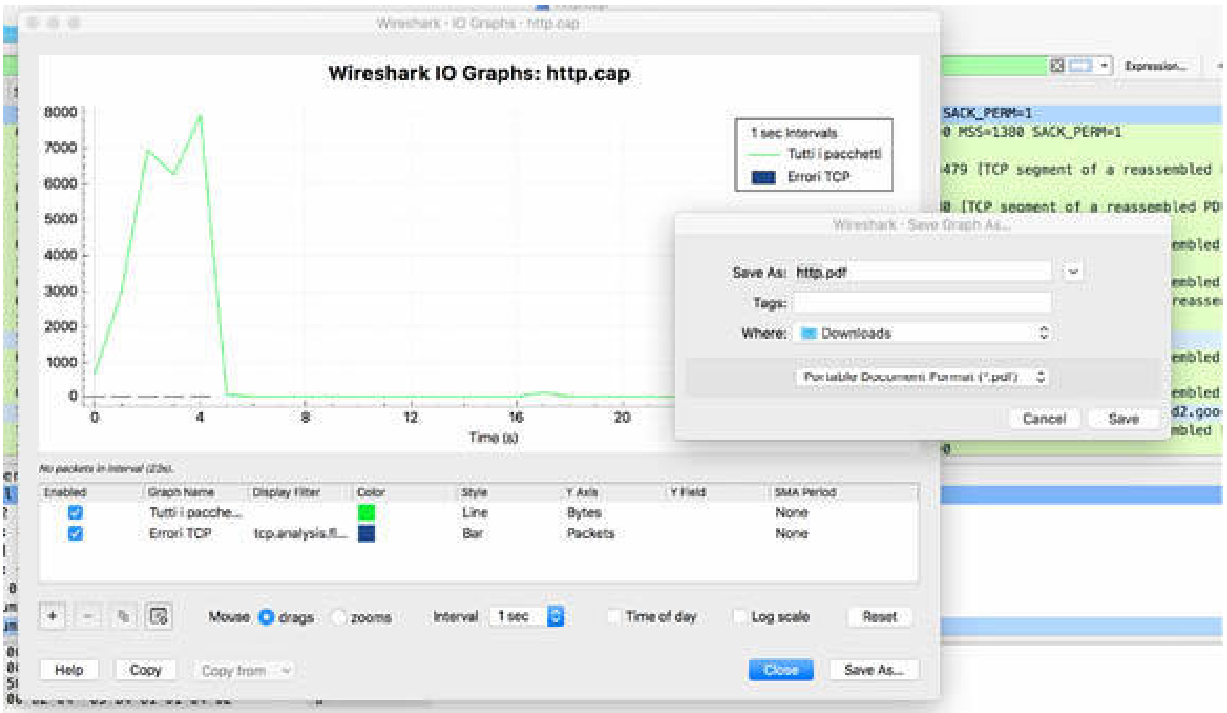
The Conversation Graph dialog box is displayed. Save the graph in PDF format.



Task 5:
On the main menu, select Statistics > I/O Graph.



In the I/O Graph dialog box, save the graph in the PDF format.



Notes:

To gain more confidence in using the export menu, repeat the previous steps to create and export information related to endpoints and flow graphs in the text or PDF format. Understand how the exporting representation is presented.

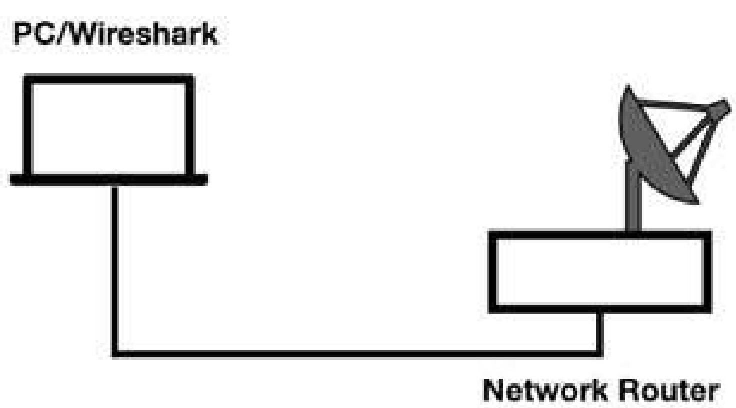
Lab 29. Save Packet Bytes and Packet Dissection

Lab Objective:
Learn how to save packet bytes and use the packet dissection feature.

Lab Purpose:
Wireshark allows you to save the packets bytes contents and packets dissection in different formats for analyzing or reporting in documents.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

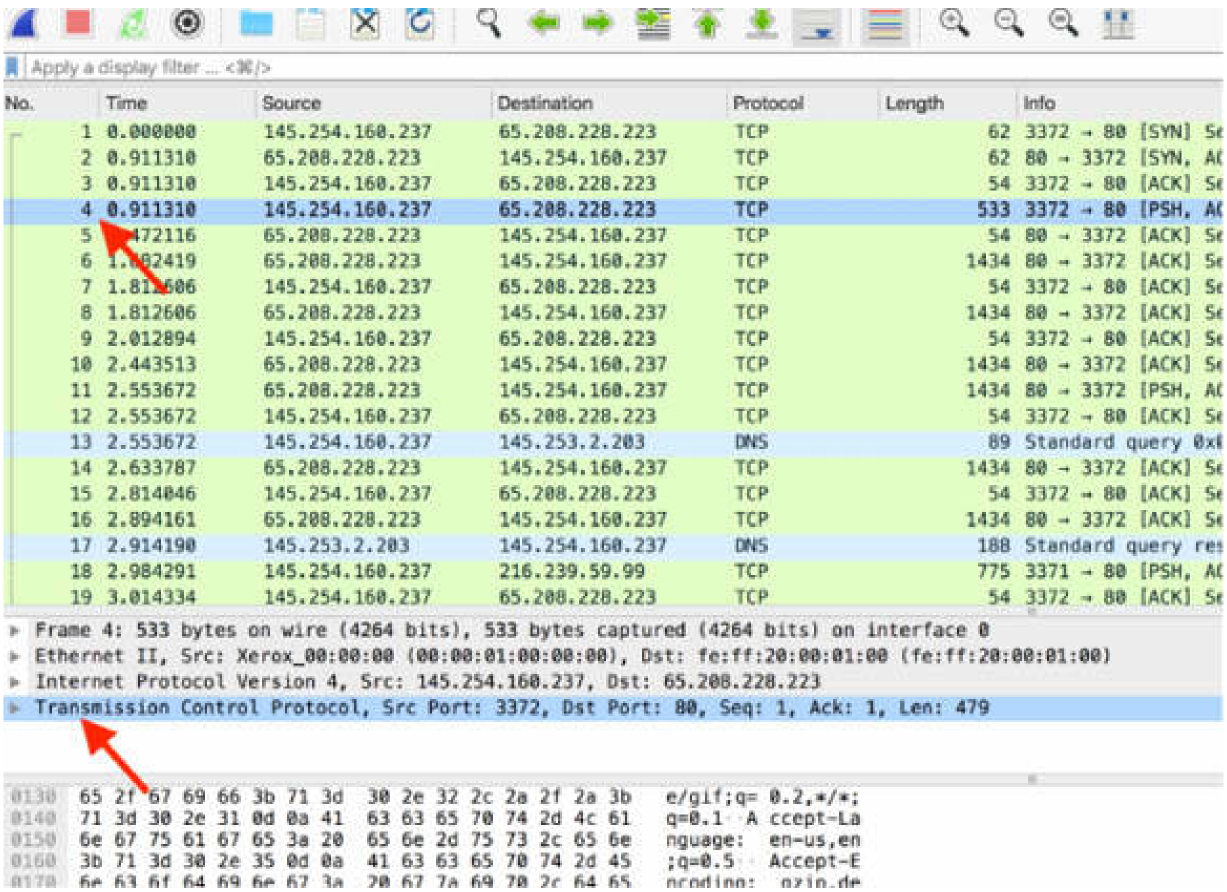
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



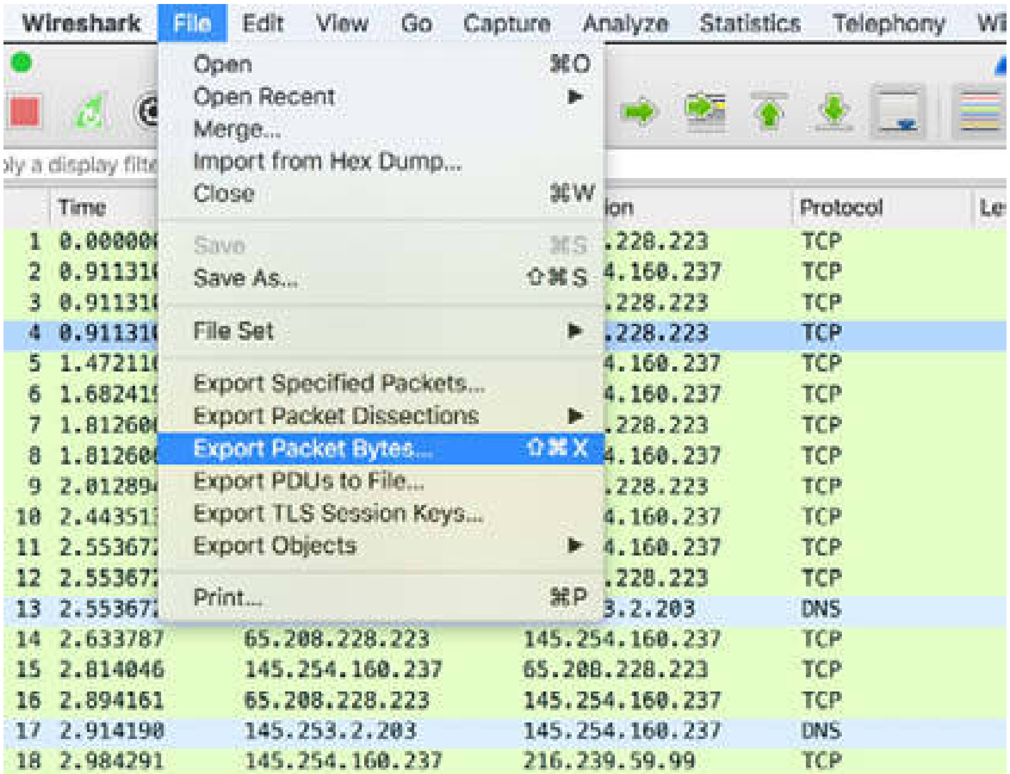
Lab Walkthrough:

Task 1:
Download the free sample capture file http.cap to your PC from https://wiki.wireshark.org/SampleCaptures#TCP , and then open the downloaded file in Wireshark.

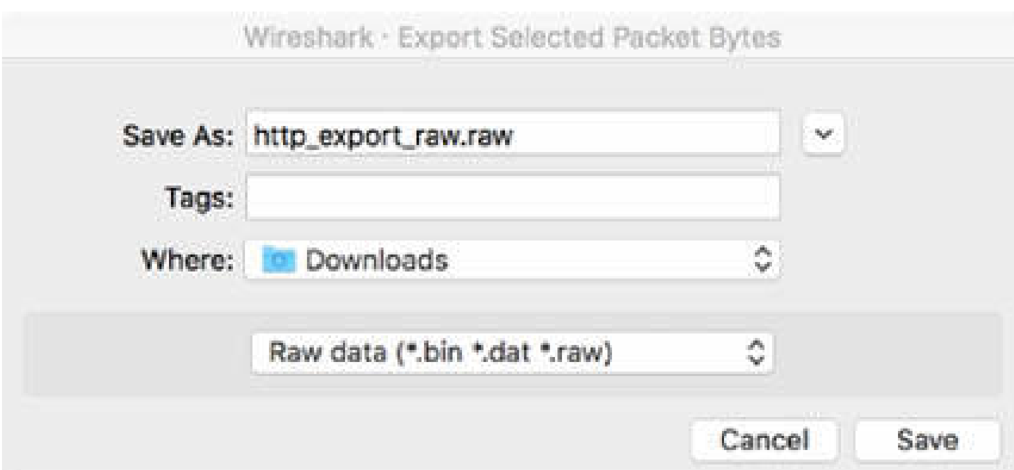
Task 2:
In the Packet List pane, select a packet, such as packet #4. In the Packet Details pane, select a layer from the ones available. In the figure below, the TCP layer is selected.



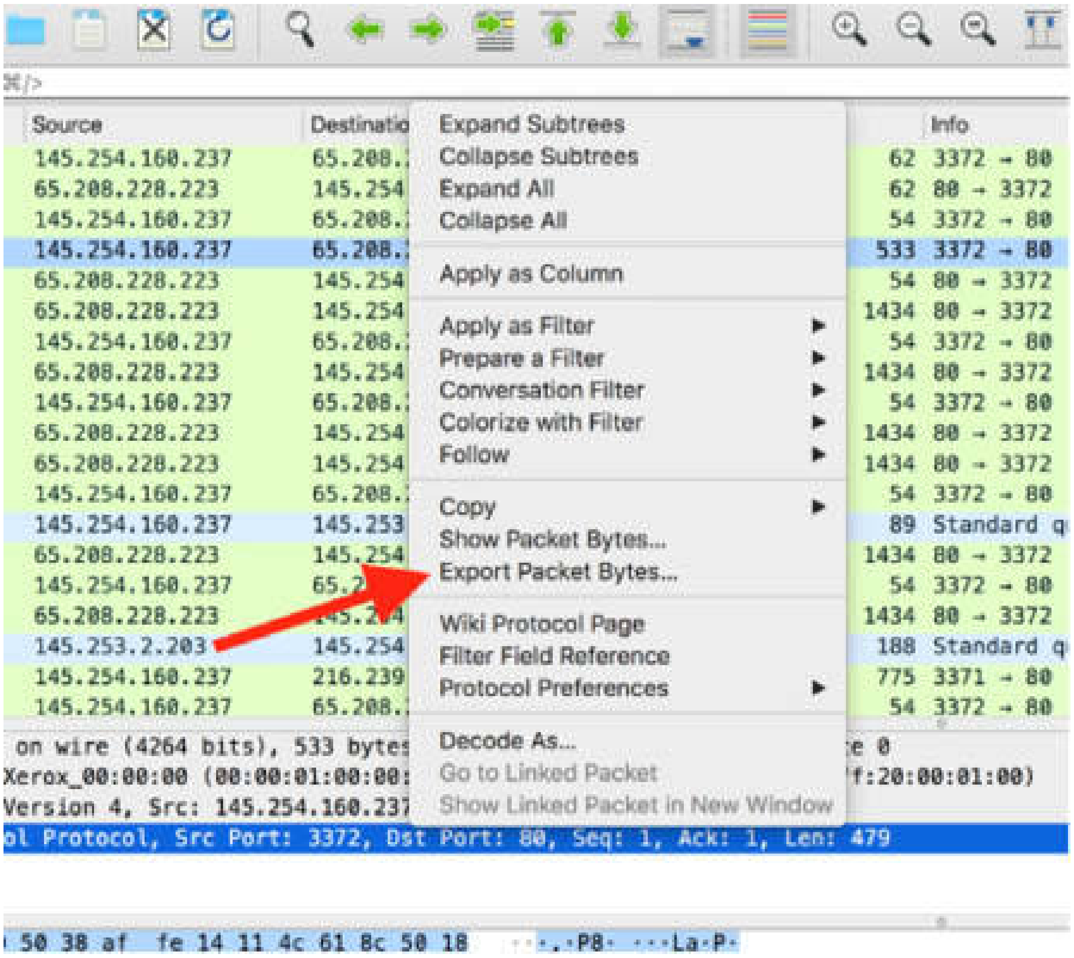
On the main menu, select File > Export Packet Bytes to export the bytes related to TCP.



The Export Selected Packet Bytes dialog box is displayed where you can specify the name, format, and destination folder; for example, use raw format to export data in a raw binary file.

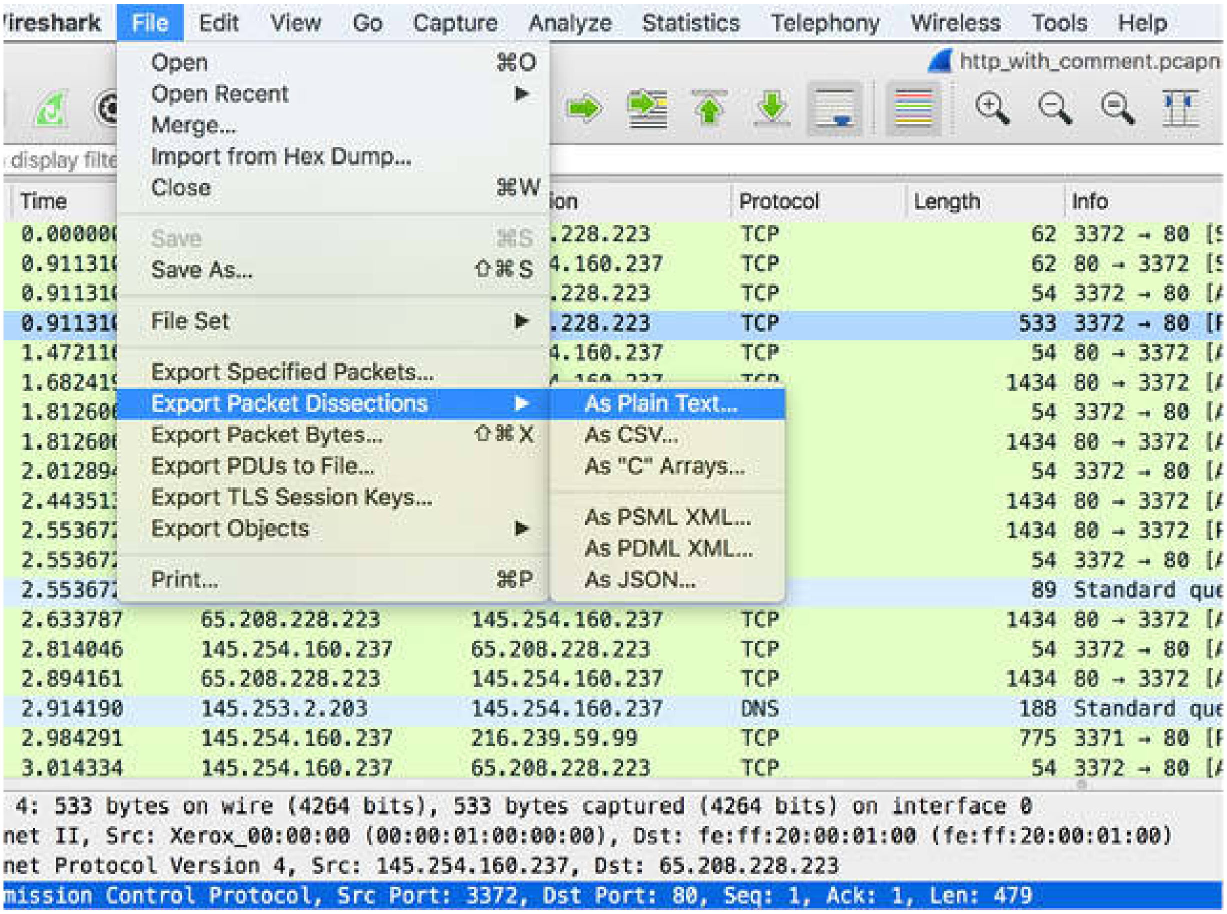


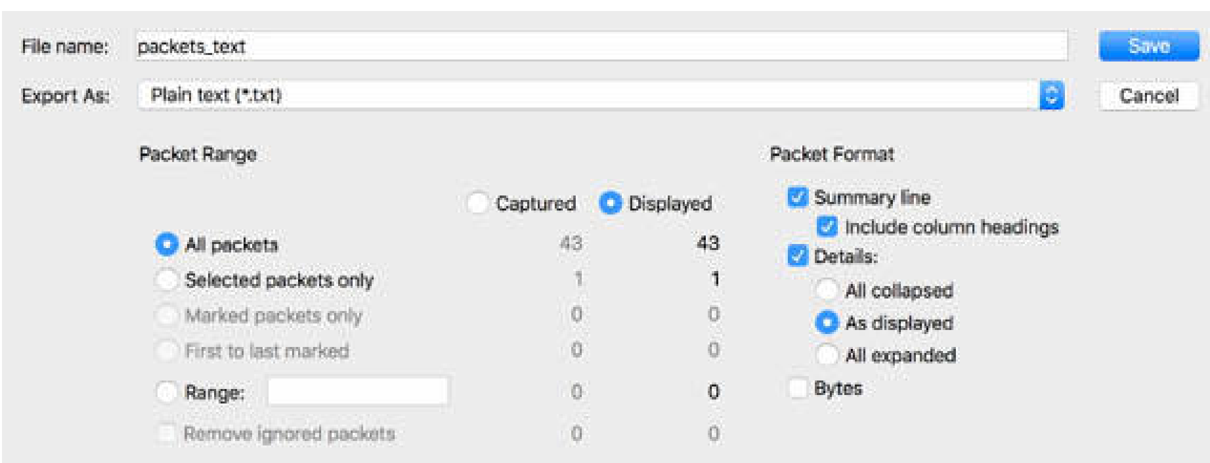
You can also access the Export Packet Bytes option by right-clicking a layer in the Packet Details pane.



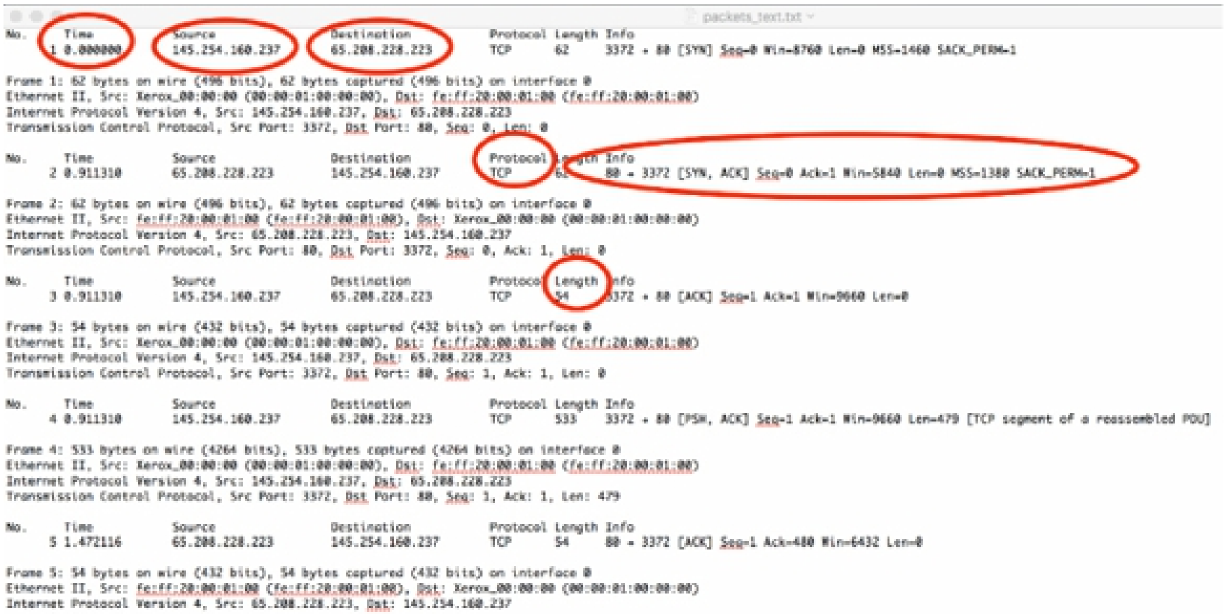
Task 3:

On the main menu, select File > Export Packet Dissection > As Plain Text, and save the file to a destination folder.





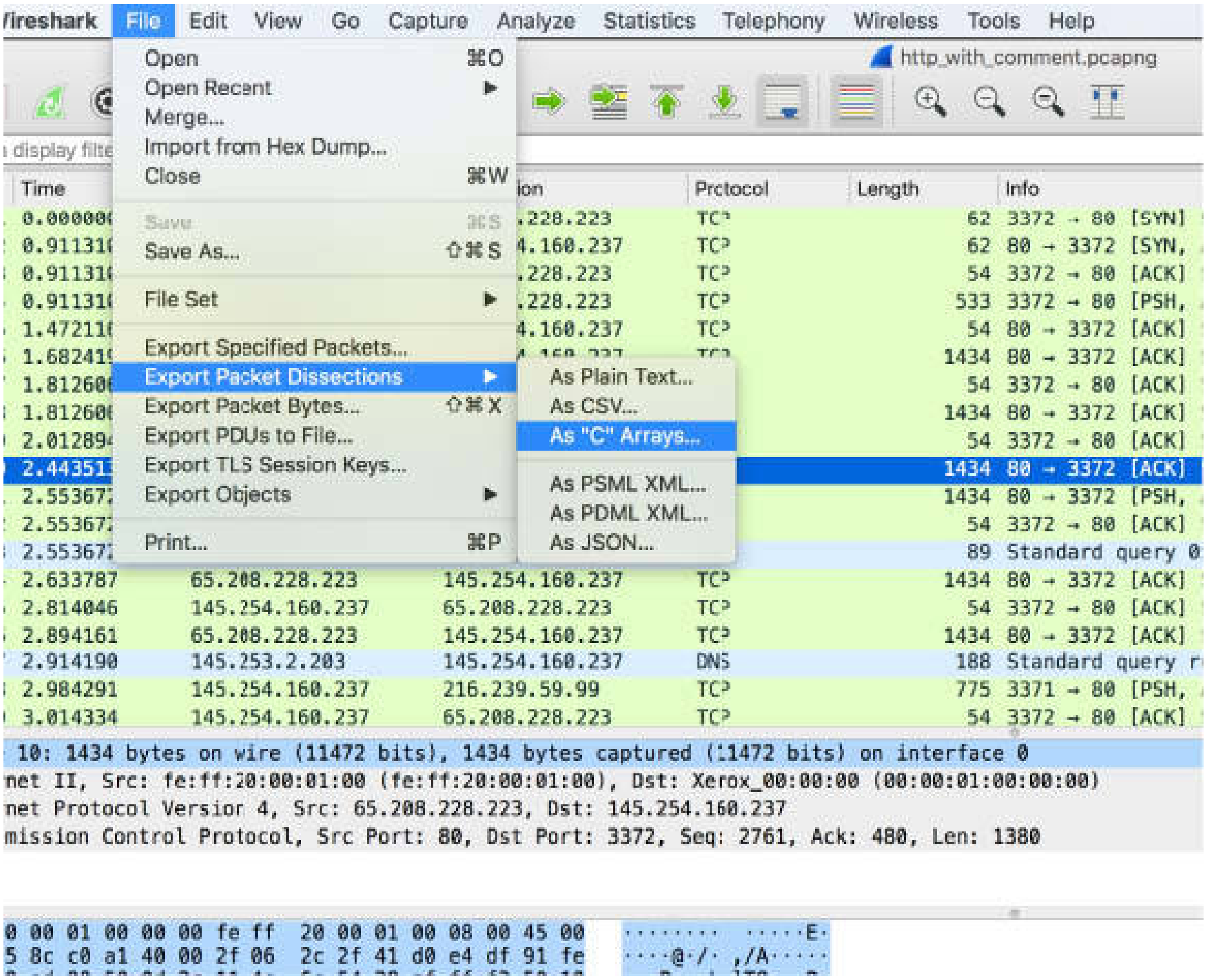
The resultant plain text file contains the relevant information about the packets from the Packet List pane, as displayed in the figure below. Please feel free to download the original images from the resources page on www.101labs.net .


packets_text.txt

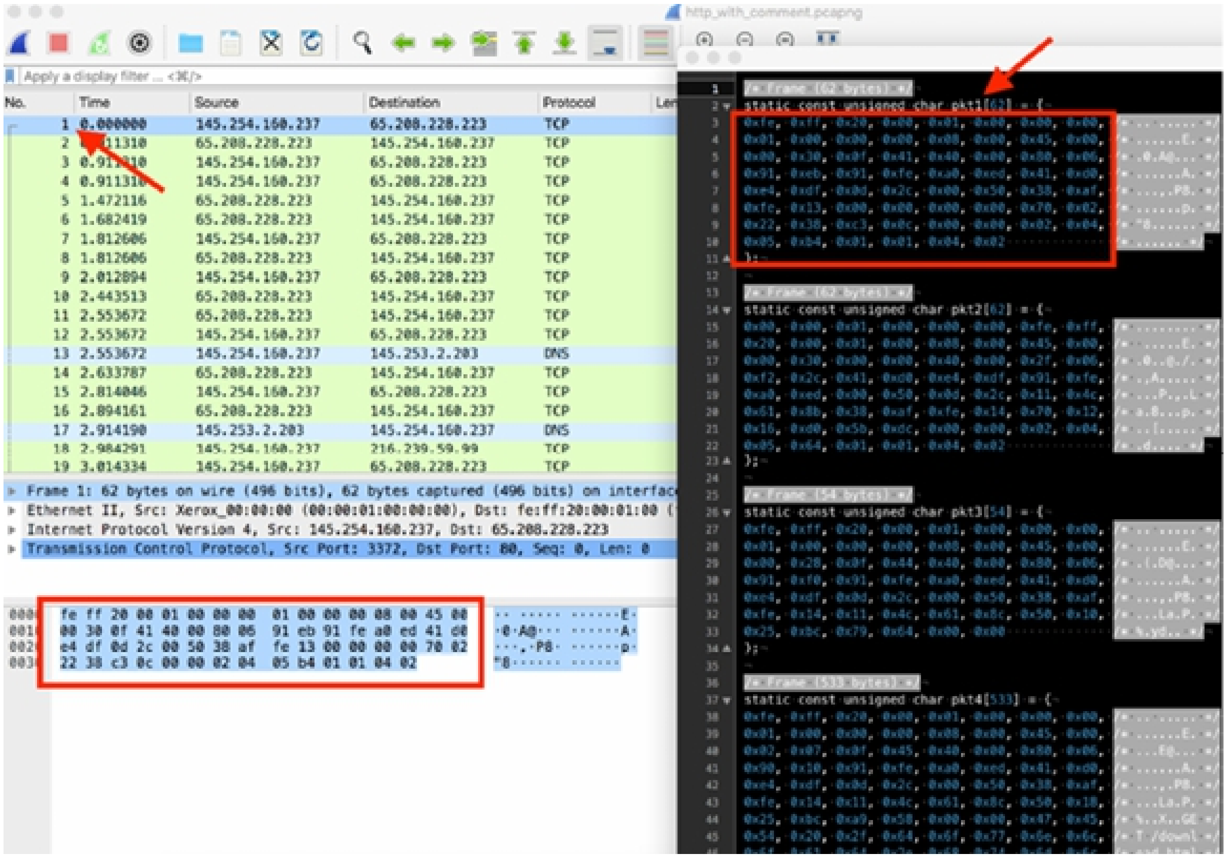
```
No.   Time        Source            Destination       Protocol Length Info
    1 0.000000    145.254.160.237   65.208.228.223    TCP      62     3372 → 80 [SYN] Seq=0 Win=8760 Len=0 MSS=1460 SACK_PERM=1

Frame 1: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface 0
Ethernet II, Src: Xerox_00:00:00 (00:00:01:00:00:00), Dst: fe:ff:20:00:01:00 (fe:ff:20:00:01:00)
Internet Protocol Version 4, Src: 145.254.160.237, Dst: 65.208.228.223
Transmission Control Protocol, Src Port: 3372, Dst Port: 80, Seq: 0, Len: 0

No.   Time        Source            Destination       Protocol Length Info
    2 0.911310    65.208.228.223    145.254.160.237   TCP      62     80 → 3372 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1380 SACK_PERM=1

Frame 2: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface 0
Ethernet II, Src: fe:ff:20:00:01:00 (fe:ff:20:00:01:00), Dst: Xerox_00:00:00 (00:00:01:00:00:00)
Internet Protocol Version 4, Src: 65.208.228.223, Dst: 145.254.160.237
Transmission Control Protocol, Src Port: 80, Dst Port: 3372, Seq: 0, Ack: 1, Len: 0

No.   Time        Source            Destination       Protocol Length Info
    3 0.911310    145.254.160.237   65.208.228.223    TCP      54     3372 → 80 [ACK] Seq=1 Ack=1 Win=9660 Len=0

Frame 3: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
Ethernet II, Src: Xerox_00:00:00 (00:00:01:00:00:00), Dst: fe:ff:20:00:01:00 (fe:ff:20:00:01:00)
Internet Protocol Version 4, Src: 145.254.160.237, Dst: 65.208.228.223
Transmission Control Protocol, Src Port: 3372, Dst Port: 80, Seq: 1, Ack: 1, Len: 0

No.   Time        Source            Destination       Protocol Length Info
    4 0.911310    145.254.160.237   65.208.228.223    TCP      533    3372 → 80 [PSH, ACK] Seq=1 Ack=1 Win=9660 Len=479 [TCP segment of a reassembled PDU]

Frame 4: 533 bytes on wire (4264 bits), 533 bytes captured (4264 bits) on interface 0
Ethernet II, Src: Xerox_00:00:00 (00:00:01:00:00:00), Dst: fe:ff:20:00:01:00 (fe:ff:20:00:01:00)
Internet Protocol Version 4, Src: 145.254.160.237, Dst: 65.208.228.223
Transmission Control Protocol, Src Port: 3372, Dst Port: 80, Seq: 1, Ack: 1, Len: 479

No.   Time        Source            Destination       Protocol Length Info
    5 1.472116    65.208.228.223    145.254.160.237   TCP      54     80 → 3372 [ACK] Seq=1 Ack=480 Win=6432 Len=0

Frame 5: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
Ethernet II, Src: fe:ff:20:00:01:00 (fe:ff:20:00:01:00), Dst: Xerox_00:00:00 (00:00:01:00:00:00)
Internet Protocol Version 4, Src: 65.208.228.223, Dst: 145.254.160.237
```

Task 4:

On the main menu, select File > Export Packet Dissection > As C Arrays, and save the file to a destination folder.



Open the resultant file in a text editor and compare it with the Packets Bytes pane, as shown in the figure below. The exported file contains each packet of the capture file byte by byte.



Notes:
Repeat the previous steps and export the packets dissections for each file format. Analyze the resultant file to understand the differences.

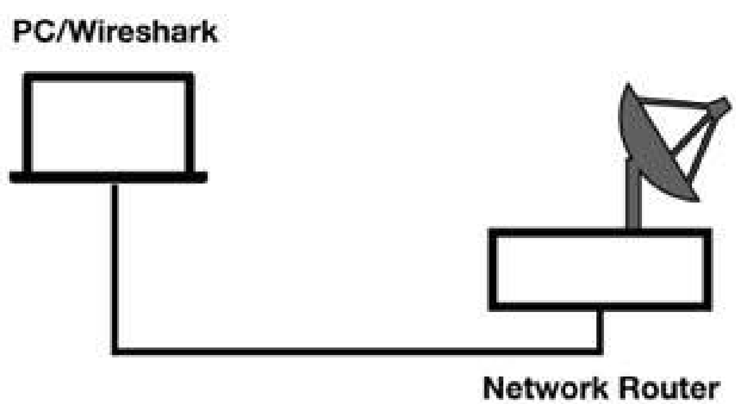
Lab 30. Expert Info

Lab Objective:
Learn how to use the Expert Info feature.

Lab Purpose:
Wireshark allows you to view and manage the expert information related to the protocol under examination to provide a quick way for finding the most relevant information (rather than using the Details tab).

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



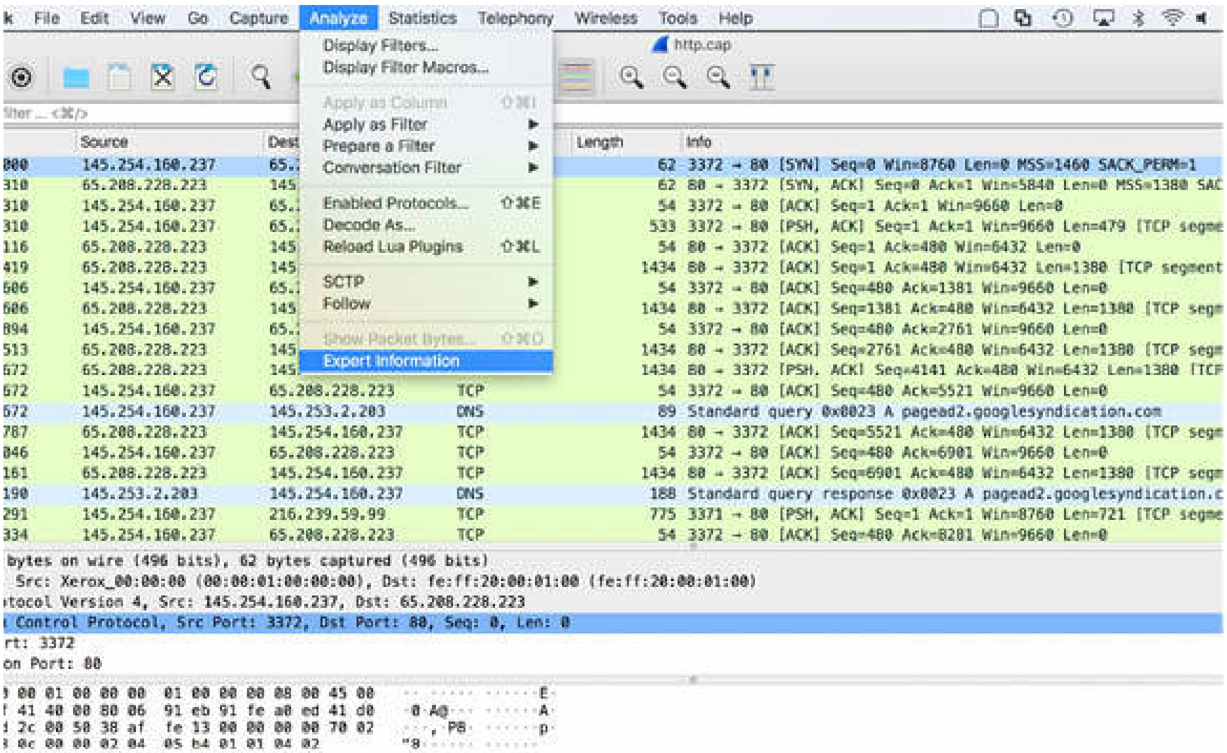
Lab Walkthrough:

Task 1:

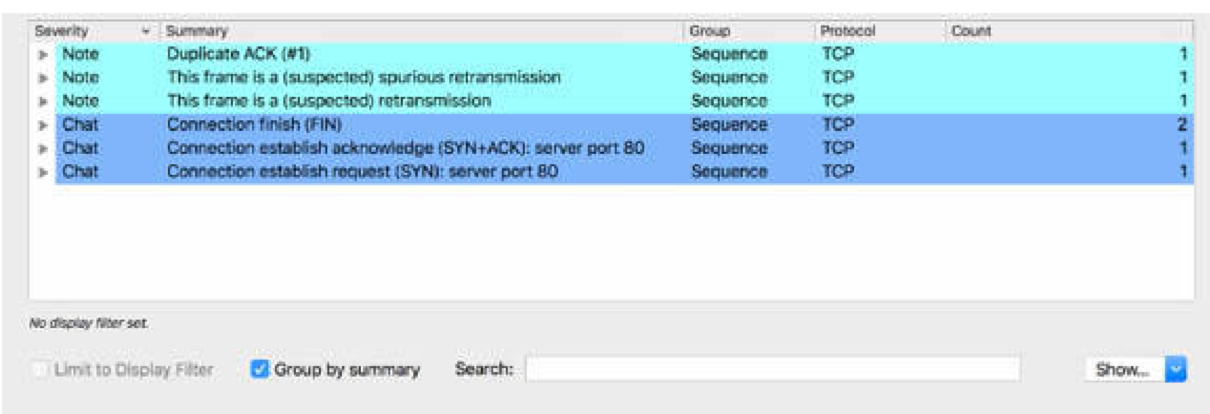
Download the free sample capture file http.cap to your PC from https://wiki.wireshark.org/SampleCaptures#TCP , and then open the downloaded file in Wireshark.

Task 2:

On the main menu, select Analyze > Expert Information.

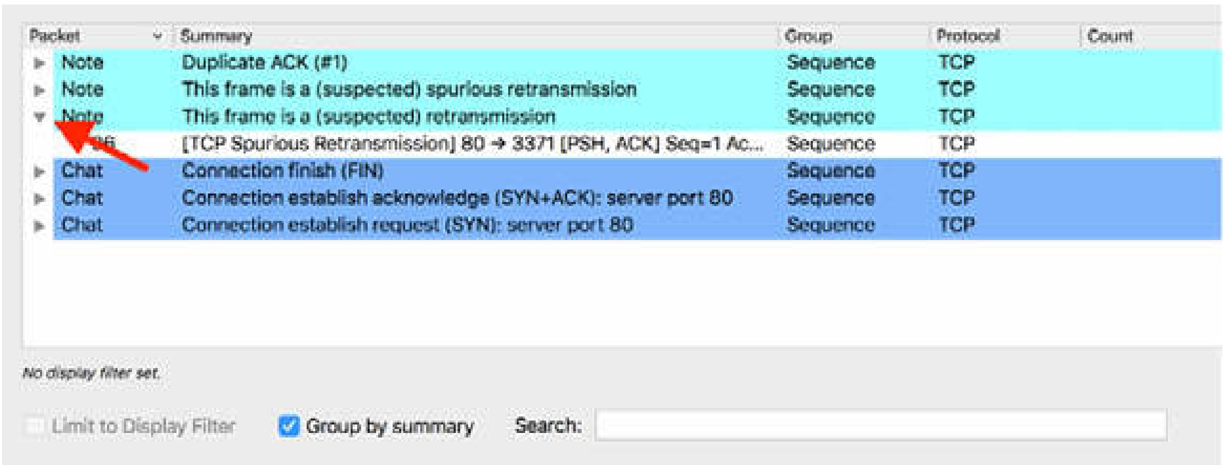


The Expert Information dialog box is displayed, showing the expert information related to the capture trace file.

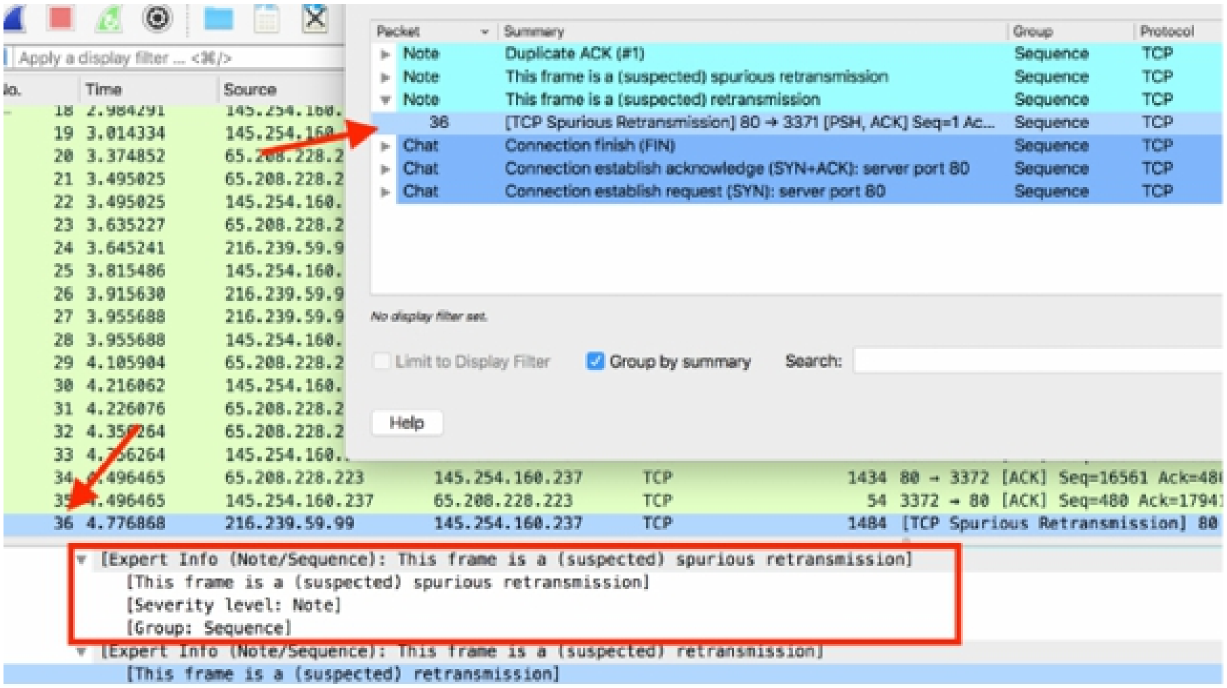


The Severity column displays the severity level, which is one of the following: Chat, Note, Warn, and Error, in ascending order. The Summary column shows a summary of the information related to the severity level.

Task 3:
Click the arrow before a grouping item to open the tree view of the expert information items.

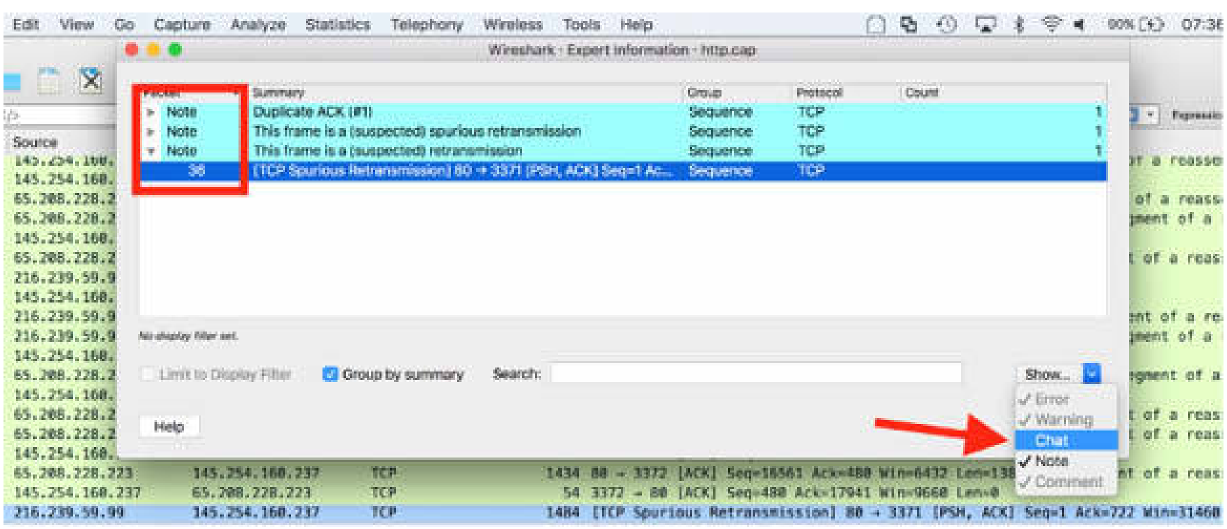


Click one of the items, and the related packet is highlighted in the Packet List pane. The expert information is displayed in the Packet Details pane, as shown in the figure below.



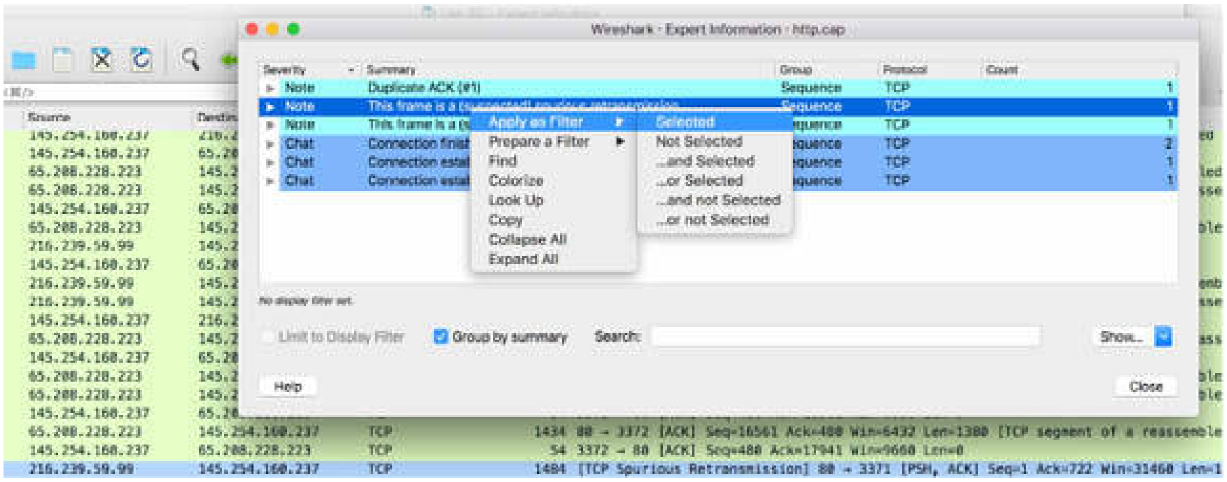
Task 4:

Click the down arrow in the Show field and disable the check on the expert info at level Chat. As a result, the related information is not displayed anymore in the Expert Information dialog box.

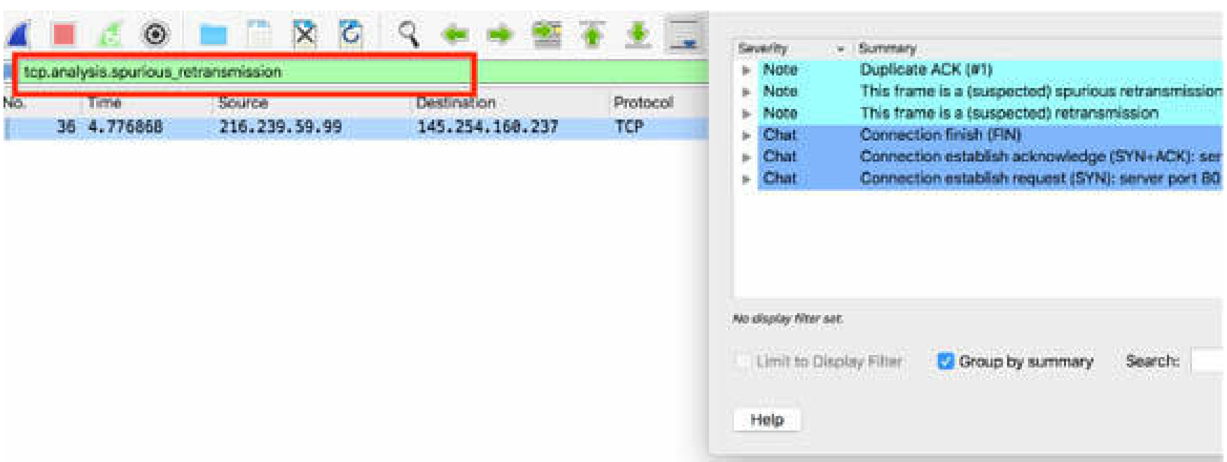


Task 5:

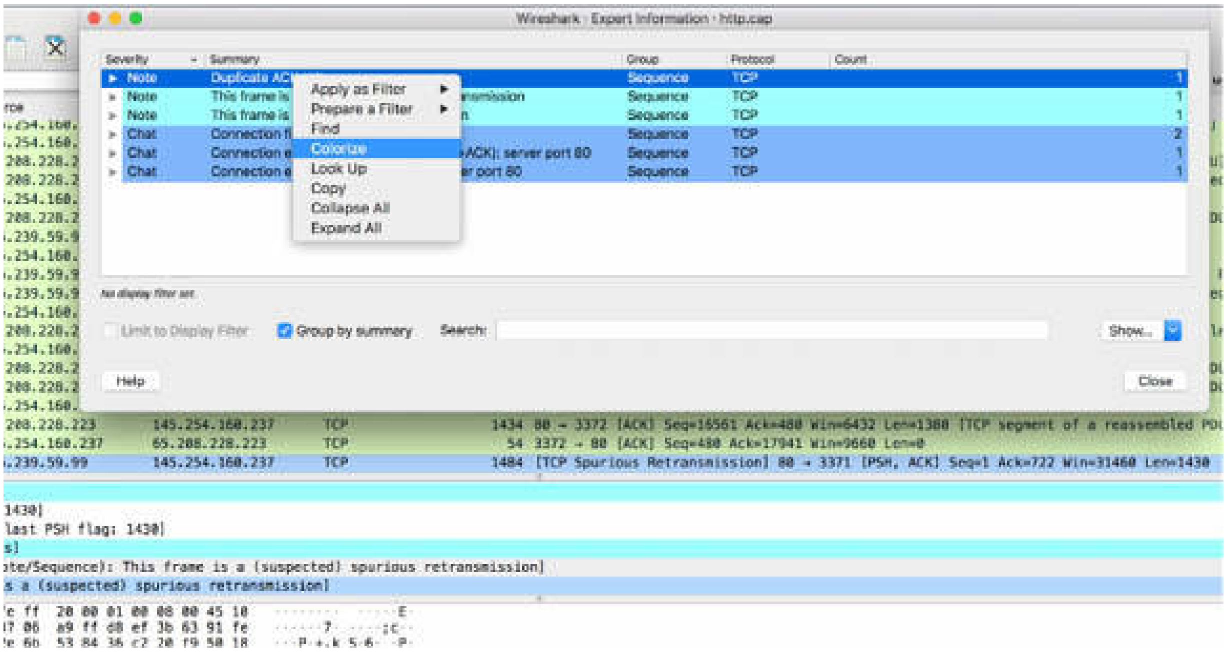
Right-click one of the grouped expert information, and select Apply as filter > Selected to filter the TCP Expert Information Elements.



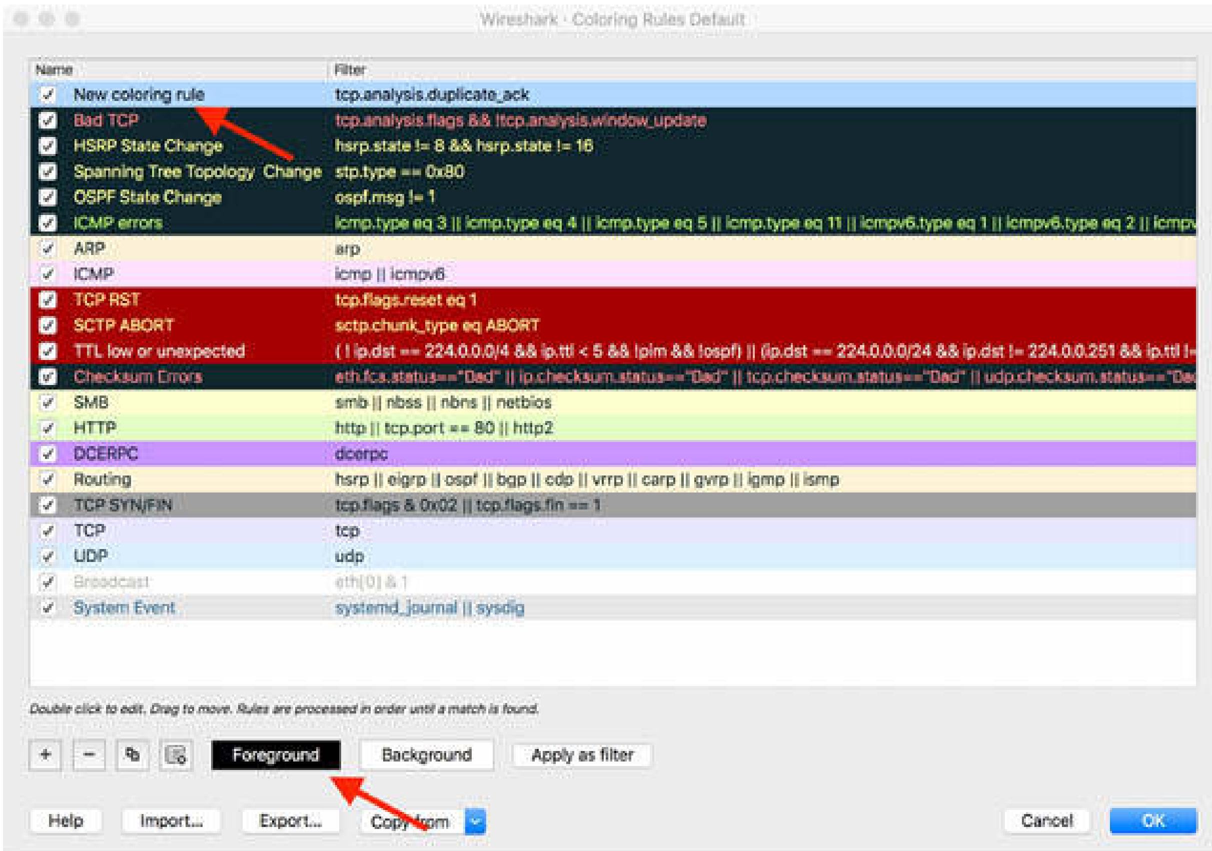
The related display filter is automatically created and displayed in the filter toolbar, as shown in the figure below. In this case, there is only one packet meeting the condition.



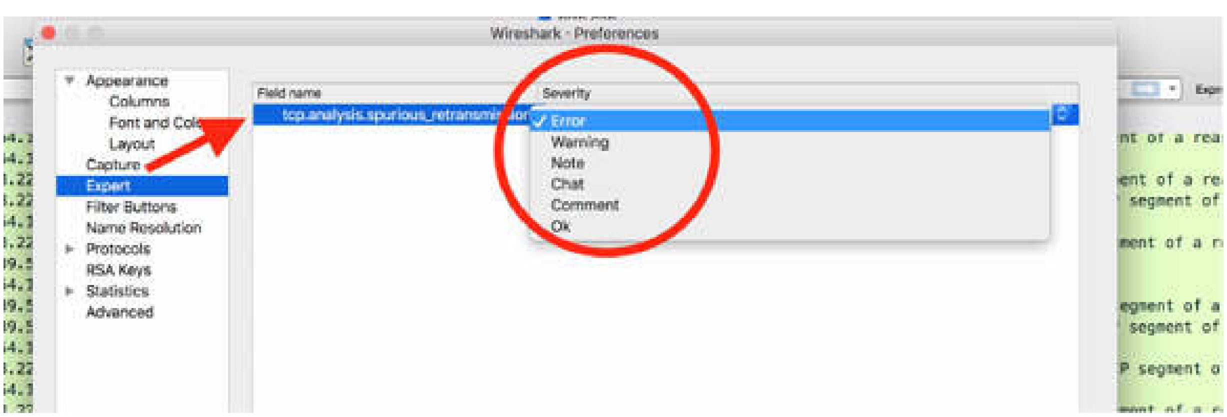
Task 6:
Right-click on an expert information item, and select Colorize to associate a default color to the related expert information.



The Coloring Rules dialog box is displayed where you can define a dedicated color and a name for the rule.



Task 7:
On the main menu, select Edit > Preferences. In the left tree view, select Expert. Add a new custom expert item by choosing a name and the related severity level, as shown in the figure below.



Notes:

To gain more confidence in using the expert information feature, repeat the previous steps by using different capture trace files and different protocols types. Take also into consideration that TCP protocol usually displays detailed expert information, but most of the other protocols currently don’t show any expert information at all. Moreover, remember that the absence of expert information doesn’t necessarily mean everything is OK.

TCP/IP Network Communications

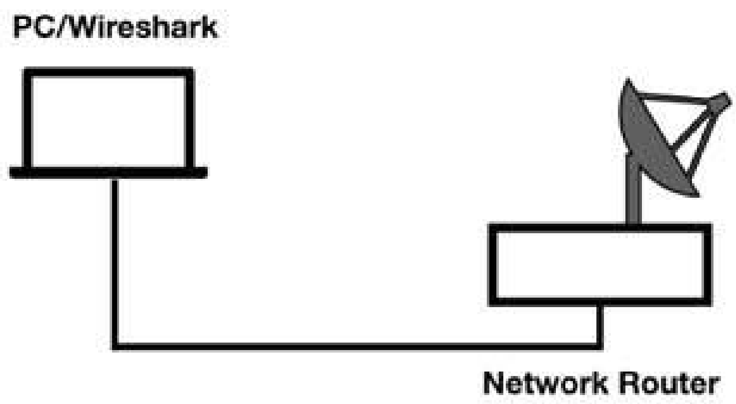
Lab 31. TCP-IP Port Number—Network Name Resolution

Lab Objective:
Learn TCP/IP basic functionalities and how the port number and network name resolution works.

Lab Purpose:
To understand how a network is behaving using Wireshark (or any network analyzer), it is important to possess a solid understanding of TCP/IP communication. TCP/IP uses a multi-step resolution process when a client communicates with a server. In this lab, we will consider both client and server on the same network.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:

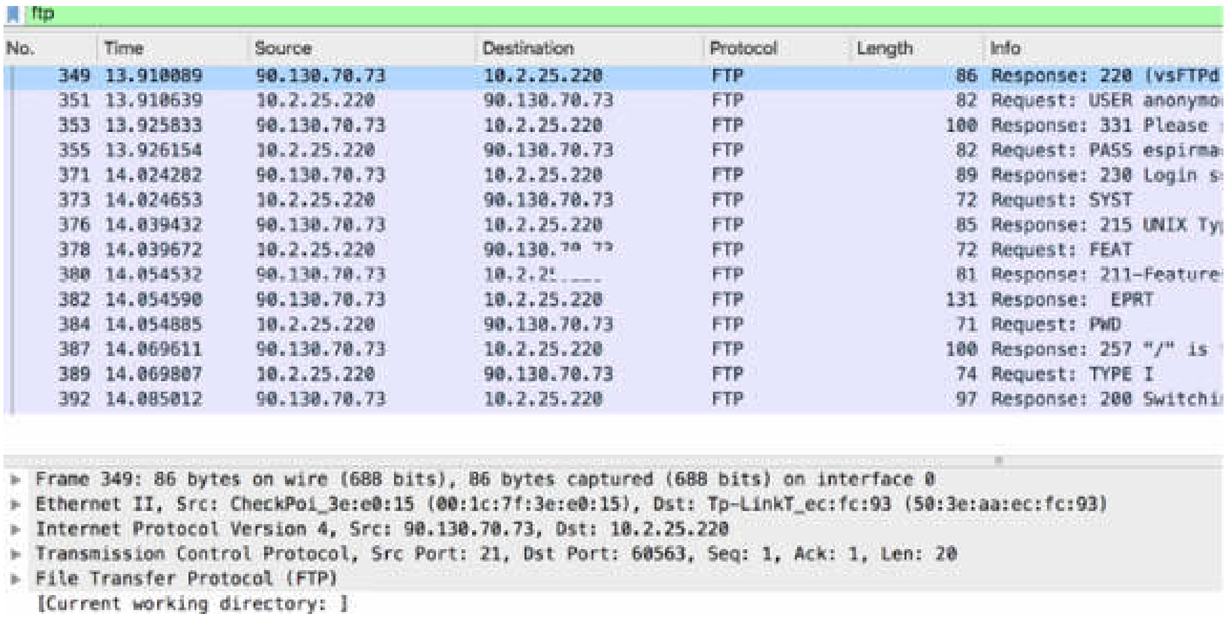
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column, and then capture the traffic for a few minutes.

Open a terminal window, and run the command `ftp ftp://speedtest.tele2.net` . A connection with the FTP server speedtest.tele2.net is opened and completed, as shown in the figure below.

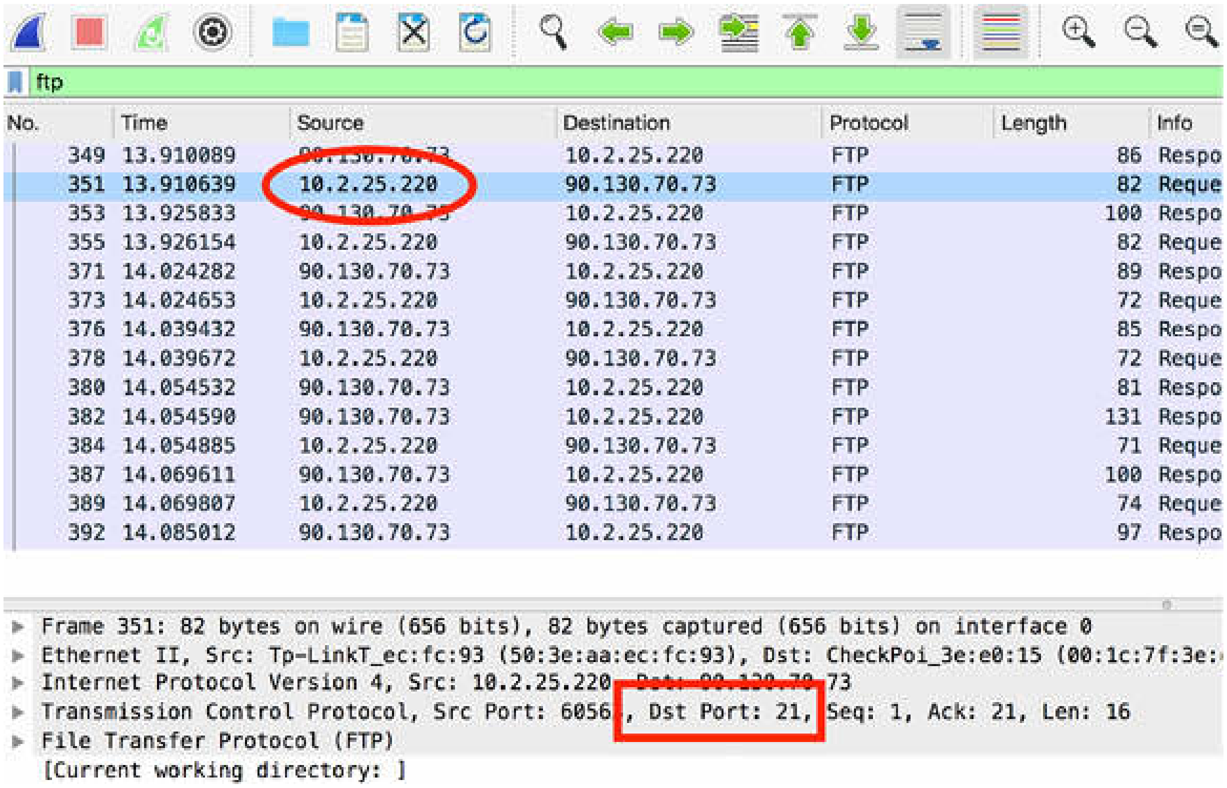
In the Windows operating system, you need to first enter `ftp` at the command prompt and then enter `open speedtest.tele2.net` . The username and password are ‘anonymous’.

Task 2:

In Wireshark, in the filter toolbar, enter ftp . All packets exchanged with the FTP server are displayed, as shown in the figure below.

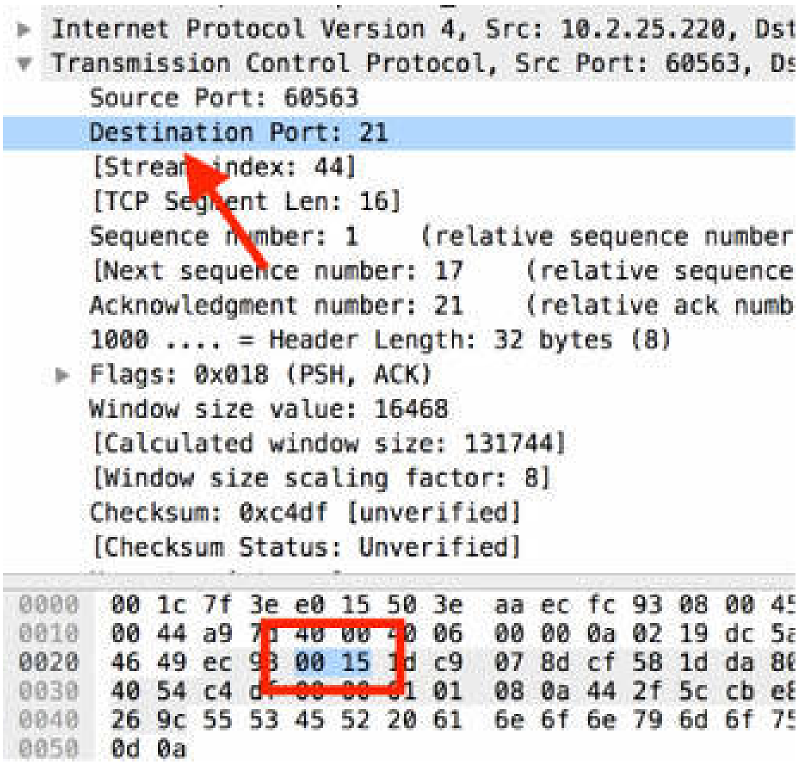


FTP typically uses port 21 for commands related to the login and password submission functions (USER and PASS). You can observe the port 21 in the Packet Details pane, as shown in the figure below. In this example, the client has the IP address 10.2.25.220, and packet #351 is selected.



Task 3:

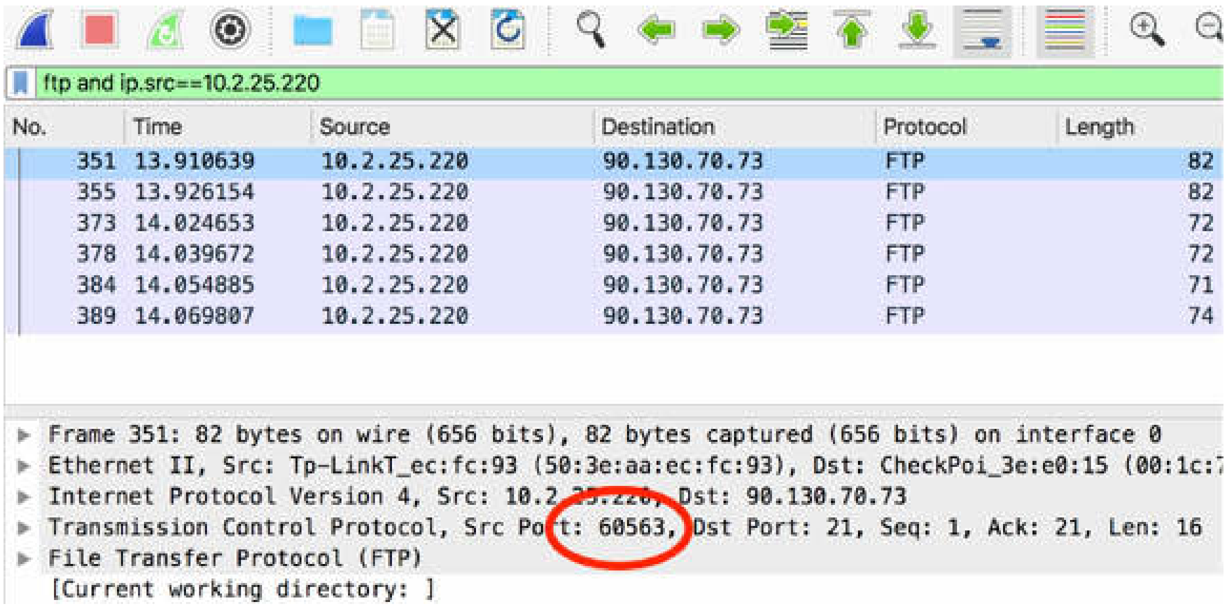
In the Packet Details pane, select the destination port. The related bytes are automatically highlighted in the Packet Bytes pane, as shown in the figure below.



This port number is contained in the etc/services file on the client. The figure above shows how this number is placed in the TCP header's destination port field of the outbound packet.

Task 4:

The client uses a dynamic port for the source port field value as displayed by selecting the packets outgoing from the IP address 10.2.25.220 (fill in the display filter with "ftp and ip.src==10.2.25.220 ").



In general, the port number resolution process does not generate traffic on the network considering that the client destination port is written in the configuration file.

Task 5:

In the previous tasks, you connected to FTP by typing in the terminal the name of the destination (speedtest.tele2.net); not the numeric IP address (90.130.70.73).

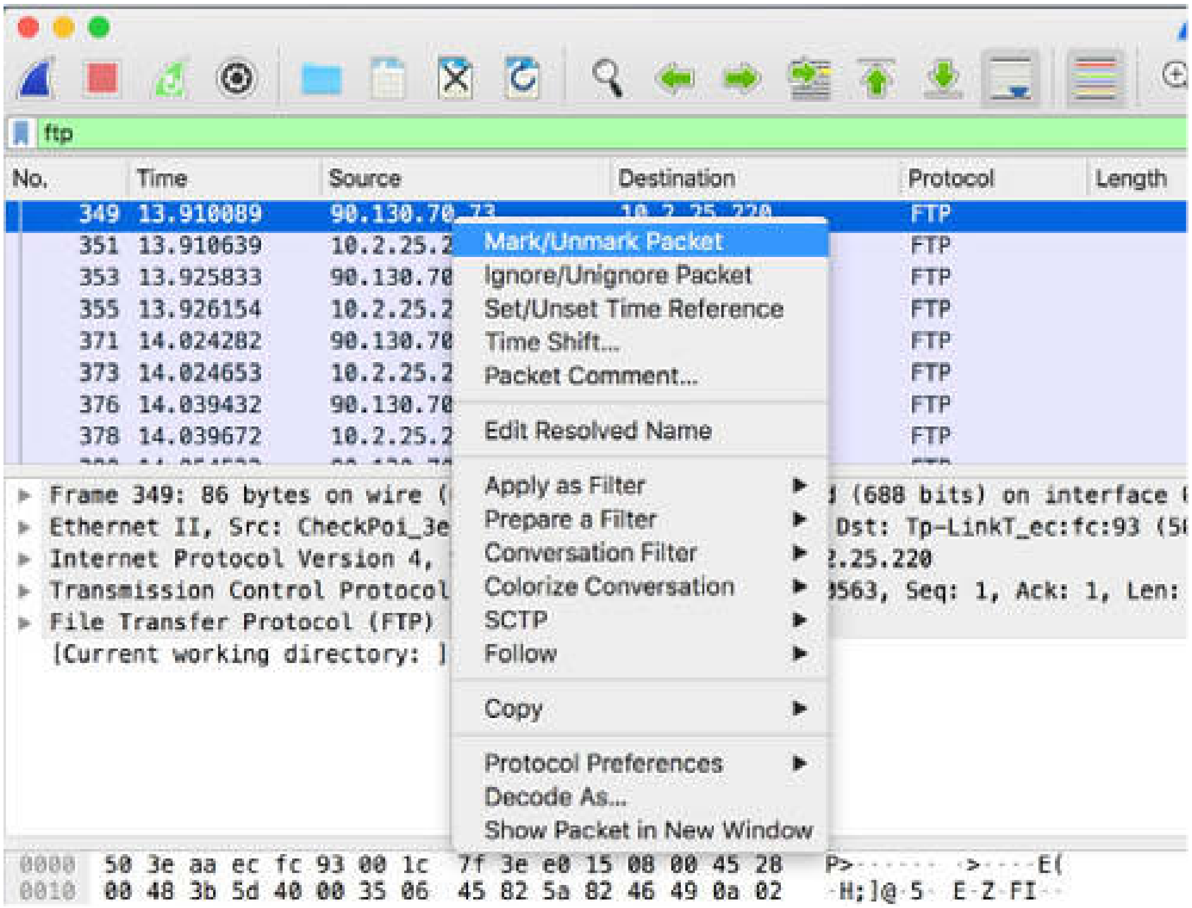
In such a case, the network name resolution process is required to obtain the IP address of the target host.

The resolution process is performed in the following order:

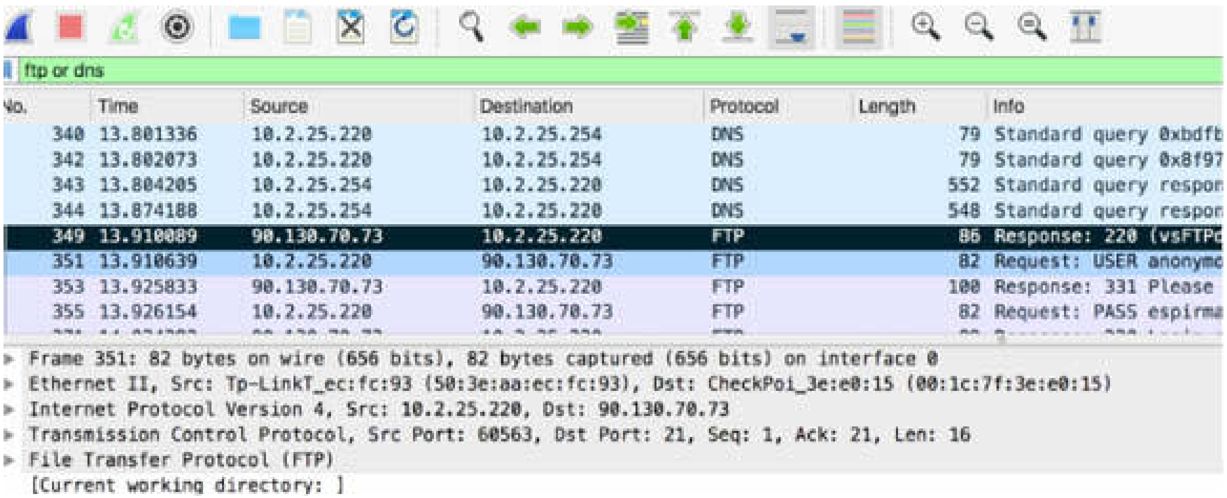
1. Look in the DNS resolver cache for the name.
2. If the entry is not in DNS resolver cache, examine the local hosts file (if one exists).
3. If the local hosts file does not exist or the desired name/address is not in the hosts file, send requests to the DNS server (if one has been configured for the local system).

Steps 1 and 2 are not applicable in this case (unless you previously connected to the same FTP server). Step 3 is the only applicable step.

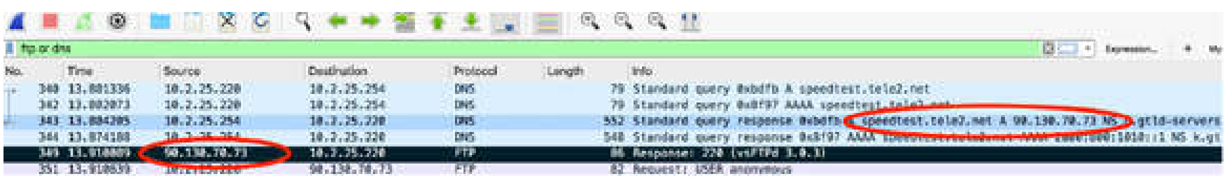
In the filter toolbar, enter ftp . In the Packet List pane, right-click the first packet and mark the packet by selecting the Mark/Unmark Packet option.



To also view the DNS packets from the capture, in the filter toolbar, enter ftp or dns .



In packets #340 and #342, you can see the DNS requests that your client machine sent to the first DNS server of your DNS server list (IP address 10.2.25.254). Similarly, in packets #343 and #344, you can see the DNS responses containing the IP of the FTP server that you were looking for.



If you didn't receive an answer from the DNS server and all DNS servers in your list did not answer, the client cannot build the packet without resolving the value to be placed in the destination IP address field.

Save this capture for the next lab.

Notes:

To gain more confidence with the port number/IP name resolution process, repeat previous steps with a different target server and/or using a non-existent FTP server name to see how the process goes ahead when the DNS resolution process fails.

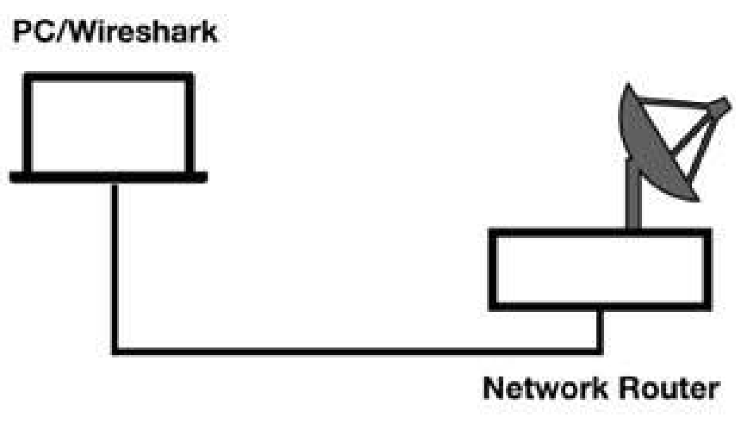
Lab 32. Route—MAC Resolution

Lab Objective:
Learn how TCP/IP route and MAC resolution work.

Lab Purpose:
TCP/IP uses a multi-step resolution process when a client communicates with a server. It is important to understand how the route resolution and the MAC address resolution processes are built.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



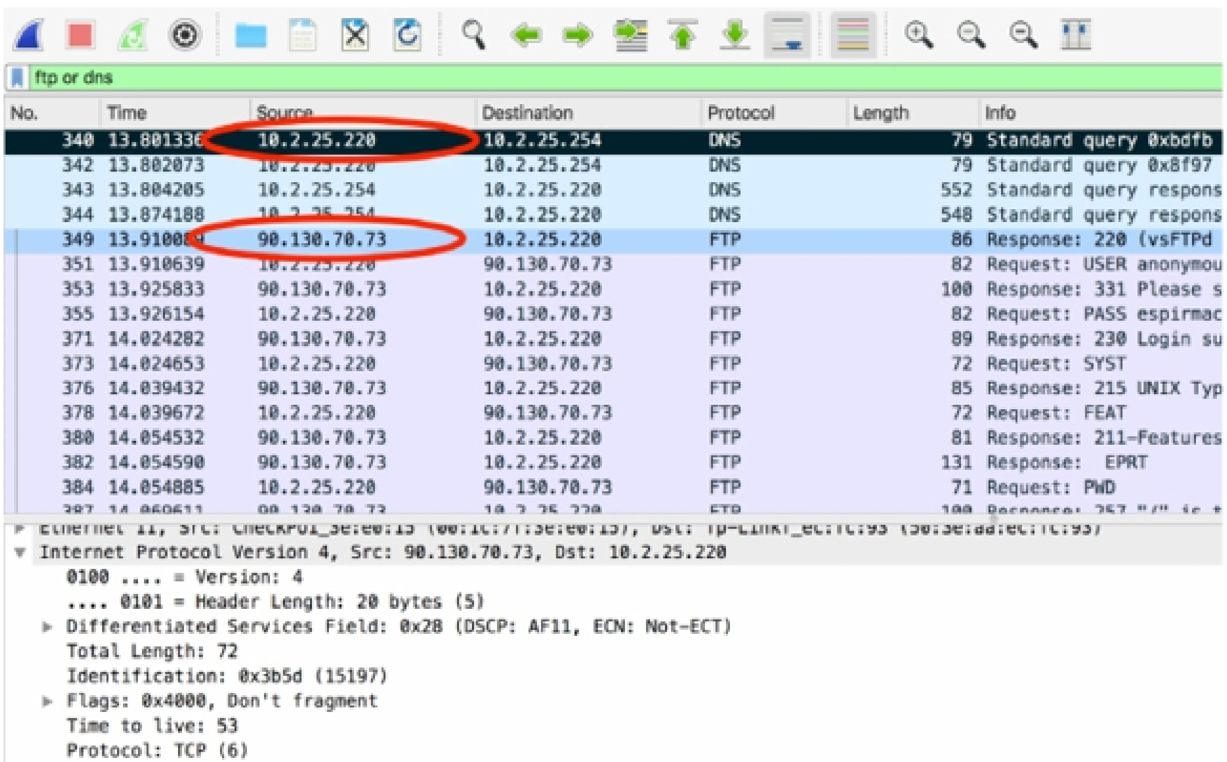
Lab Walkthrough:

Task 1:

During the Route Resolution process, the client determines whether the destination device is local (on the same network) or remote (on the other side of a router).

In Wireshark, open the capture file saved in Lab 31.

Considering you applied the display filter ftp or dns and marked the first DNS query (packet #340) sent by the client, as shown in the figure below, you can identify the IP addresses of the FTP client and server.



The client sends the first DNS request (IP address 10.2.25.220) and the server sends the first FTP response (IP address: 90.130.70.73).

To perform the route resolution, the client compares its own network address to the target network address to determine if the target is on the same network. This process, however, does not generate traffic on the network. In our example, the client is on network 10.x whereas the server is on network 90.x. This confirms that the target is remote.

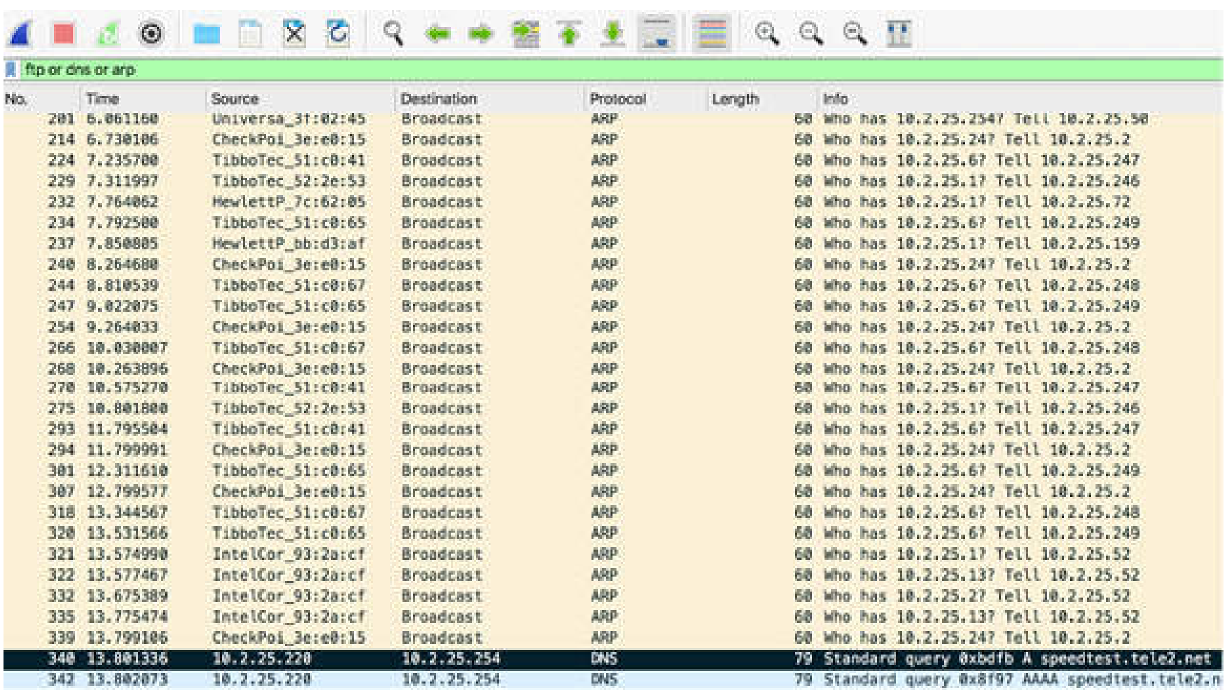
The client looks at its local routing tables to determine if it has a host or network route entry for the target. If no entry is available, the client checks for a default gateway entry. This process is not visible on the capture because it does not generate traffic on the network.

Task 2:
In the end, the client must resolve the MAC address of the next-hop router or default gateway.

The client first checks its ARP cache. If the information does not exist in the cache, the client sends an ARP broadcast to get the MAC address of the next-hop router and updates its ARP cache.

If the MAC address of the desired router is in the cache, no packets are sent. If an ARP query must be sent for the desired router, it is seen in the trace file.

In the filter toolbar, enter ftp or dns or arp to verify whether the ARP message is sent, as displayed in the figure below.

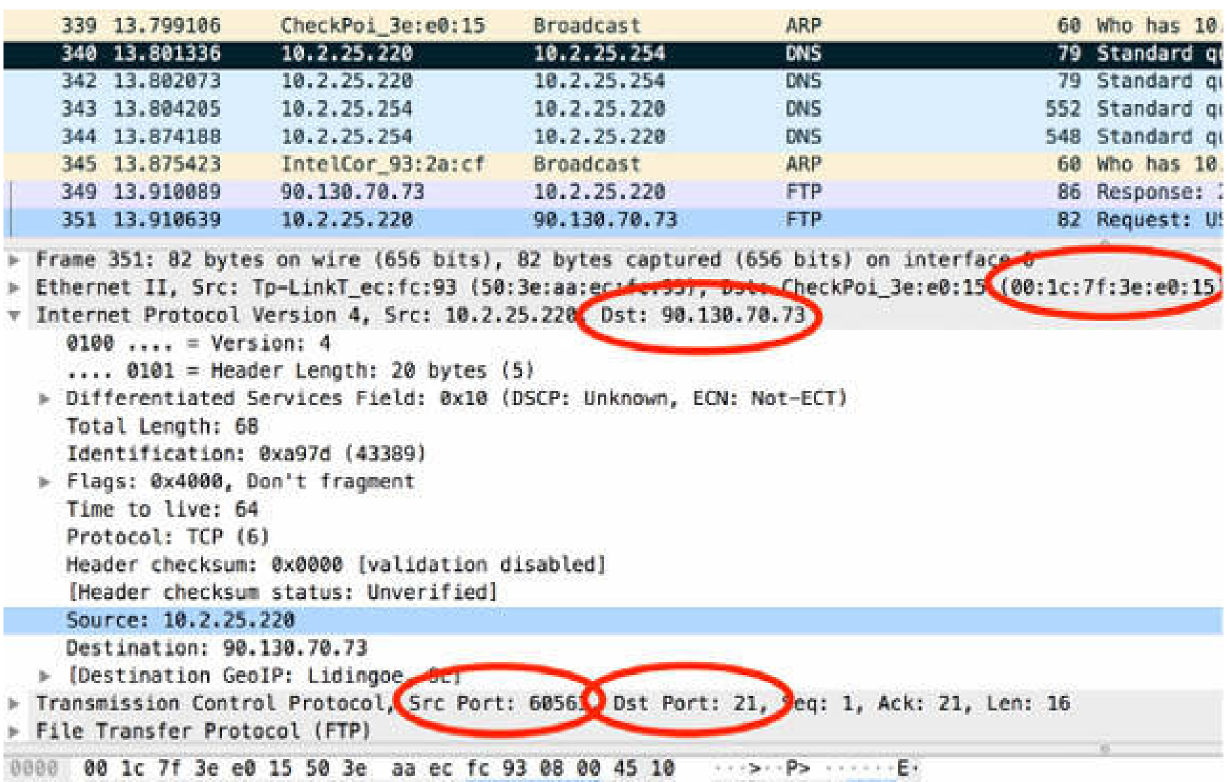


The capture file doesn't contain any ARP frame coming from the client which confirms that the client already had the default gateway MAC address in the ARP cache.

Task 3:
At the end of the multi-step resolution process, the client builds the packet by using the information retrieved. In particular:

- Destination MAC address
- Destination IP address
- Source and destination port numbers

In the Packet List pane, select the first FTP packet sent by the client (packet #351) and inspect the related details in the Packet Details pane. You can locate the destination MAC address, destination IP address, and source and destination ports, as shown in the figure below.



Notes:

To gain more confidence with the route/MAC resolution process, repeat the previous steps with a different target server, and try connecting to a local route.

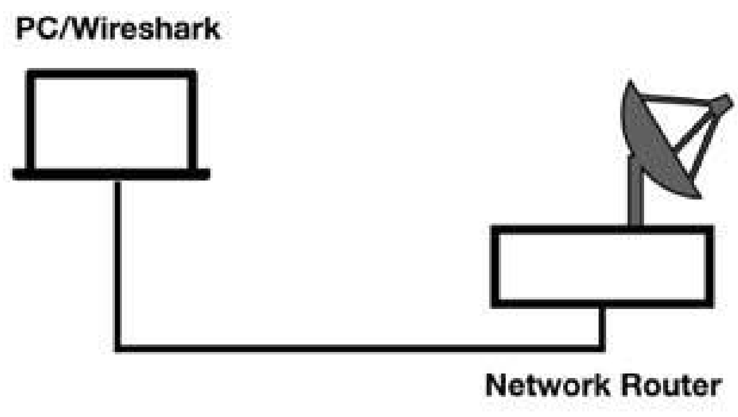
Lab 33. Domain Name System

Lab Objective:
Learn how the Domain Name System (DNS) works.

Lab Purpose:
DNS is used to convert symbolic host names, such as www.101labs.net , to IP addresses. In this lab we learn how the DNS resolution process works.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

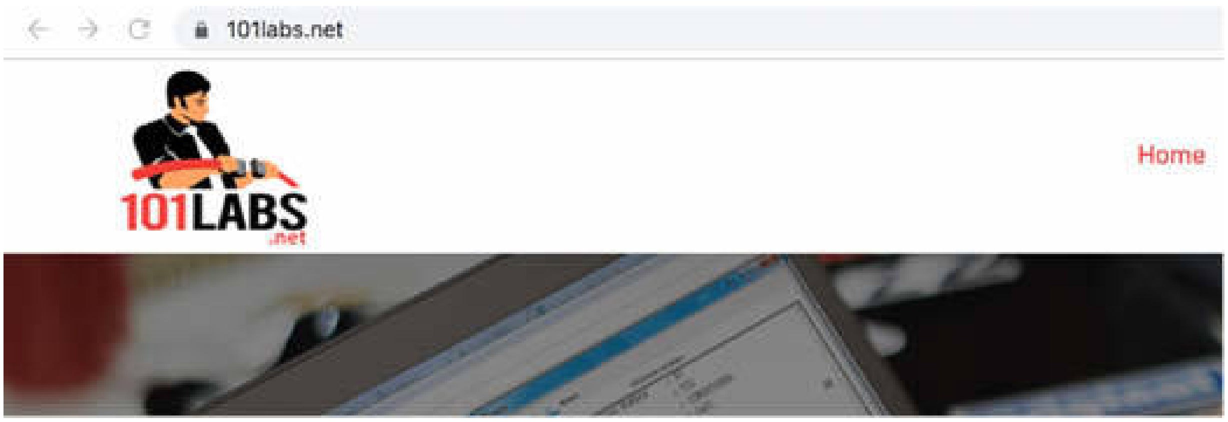


Lab Walkthrough:

Task 1:

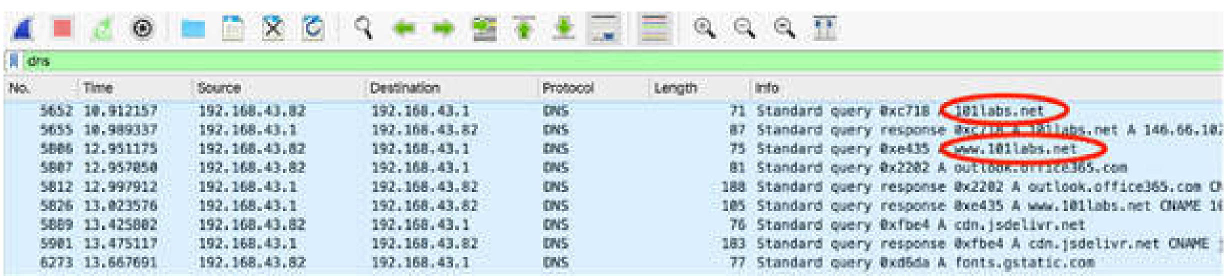
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

In a web browser, go to www.101labs.net .



Stop the capture and save the capture file.

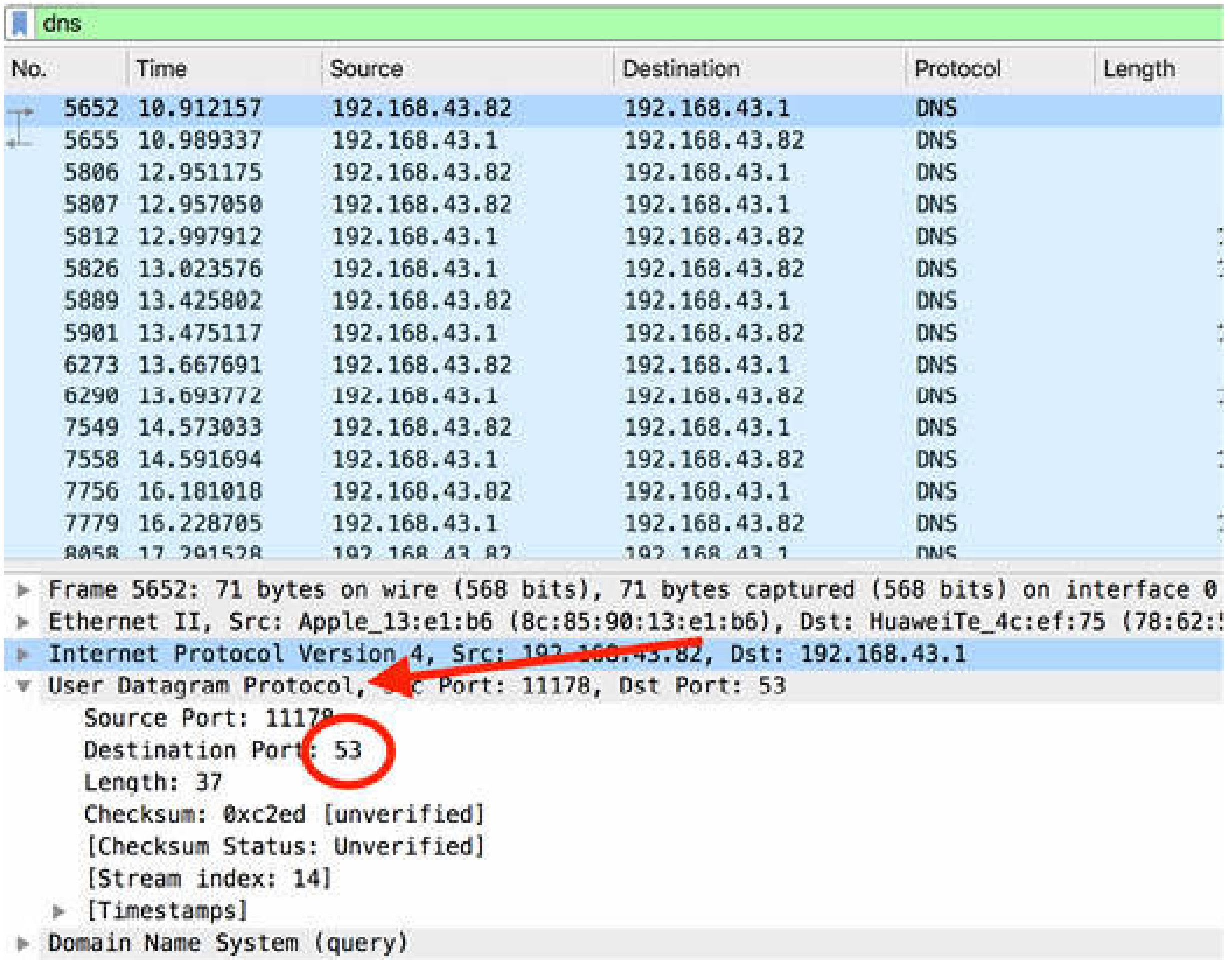
Task 2:
In the filter toolbar, enter dns , as shown in the figure below.



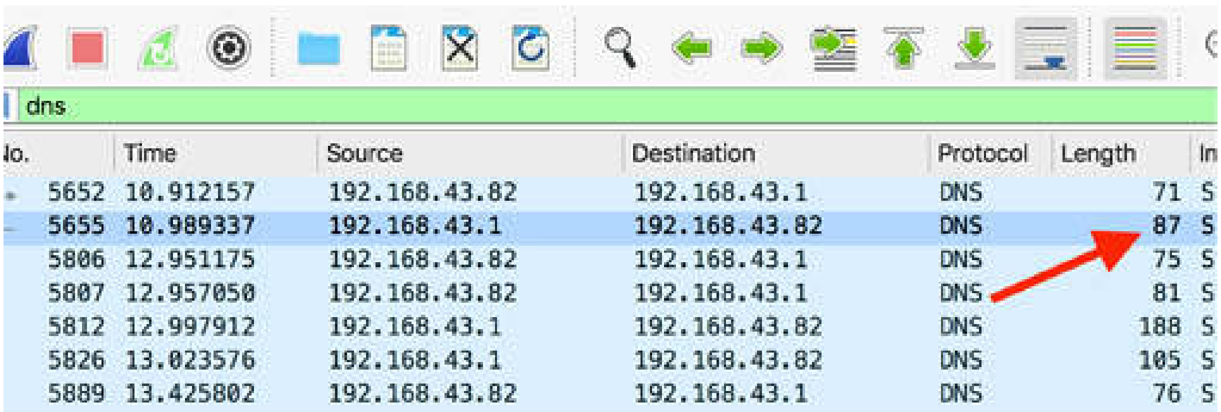
In the Packet List pane, all DNS queries and responses made by your PC during the capture are displayed. In the figure above, packets #5652, #5655, #5806, and #5807 are the DNS messages related to www.101labs.net .

Task 3:
In the Packet List pane, select the first DNS query and view the related details in the Packet Details pane. In this example, the DNS runs over UDP

(but it can also run on TCP) and uses the default DNS port 53.

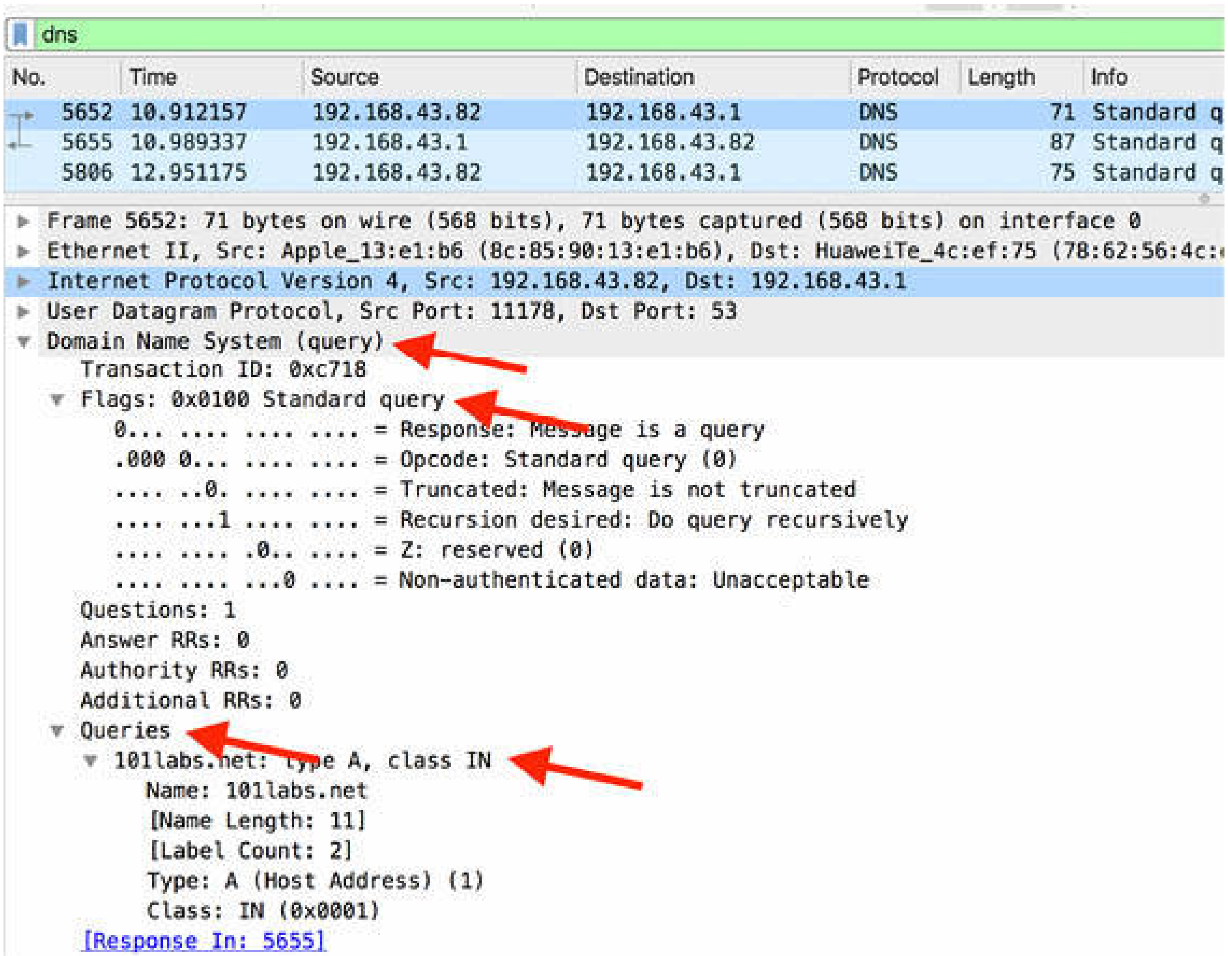


Typically, DNS is limited to 512 bytes over UDP (RFC 1035), and this length is often sufficient. If a response requires more than 512 bytes, TCP is used because it can support a larger packet size.



Task 4:
The network name resolution DNS query and response processes are very simple. A client sends a DNS query to a DNS server, typically asking for an IP address in exchange for a host name. The DNS server either responds directly with information it possesses or asks other DNS servers on behalf of the clients (recursive queries).

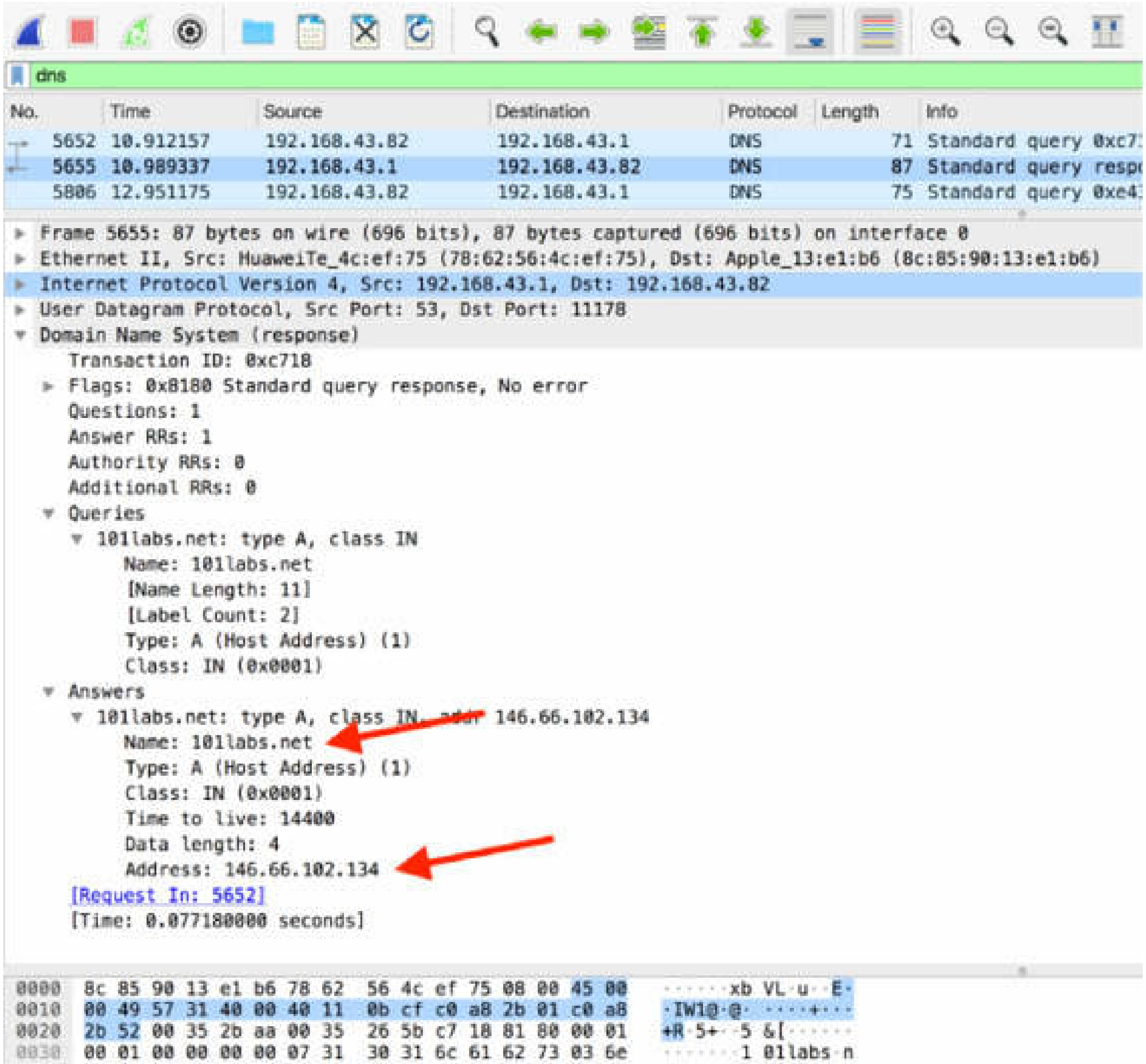
In the Packet List pane, click on the first DNS query packet and view the related information in the Packet Details pane. Enable the tree view for the following items: DNS, Flags, Queries, and record Type A, as indicated by the arrows in the figure below.



The record A contains the host address for **www.101labs.net** . This DNS query was generated automatically when you entered this host name in the browser address bar and pressed Enter.

Task 5:

In the Packet List pane, select the first DNS response and view the related information in the Packet Details pane.



In this example, observe that the name a client requests is an ‘A’ name. An IP address has directly been returned for www.101labs.net and the address for that host is 146.66.102.134.

Notes:

To gain more confidence in using the schema, repeat the previous steps for different websites by using the dns filter while you browse to see more DNS resolving processes.

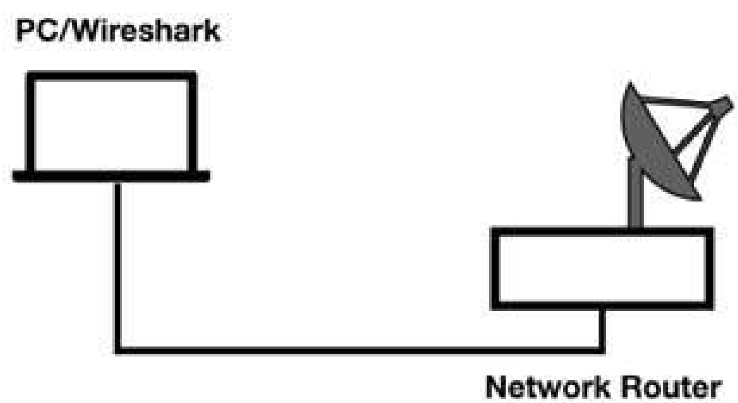
Lab 34. DNS Problems

Lab Objective:
Learn how to identify DNS problems.

Lab Purpose:
Identify the most common DNS problems.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
The most common DNS problem is the error generated because a name does not exist in the name server database. This could be caused by entering

an incorrect name or entering a new name that has not yet propagated through the internet name servers.

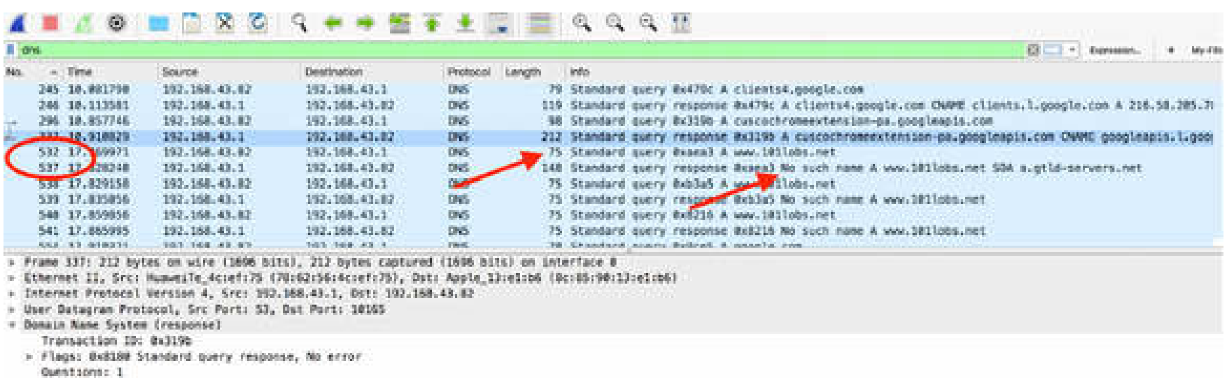
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes and then stop the capture and save the file.

In a web browser, go to **www.101lobs.net** (an intentionally incorrect name).

Stop the capture and save the capture file.

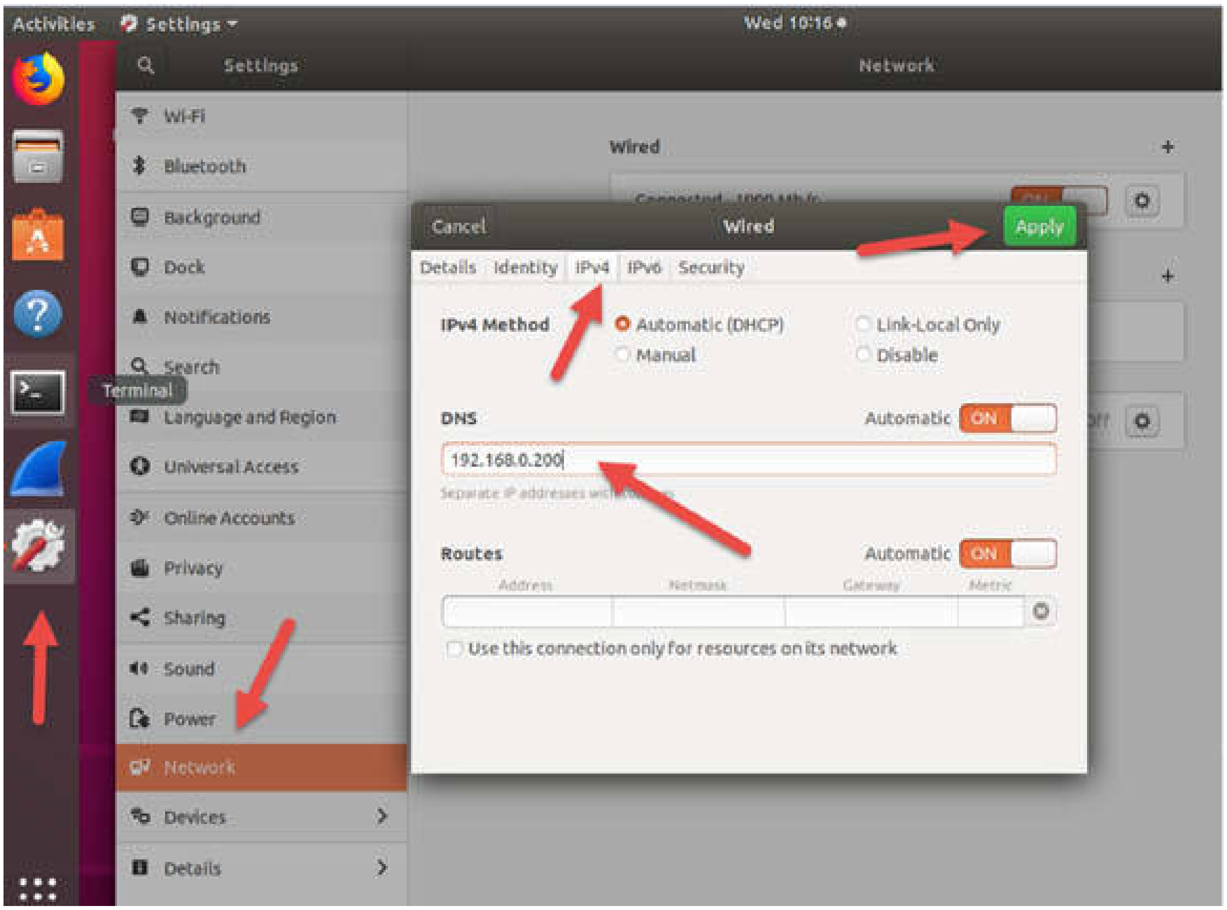
Task 2:
In the filter toolbar, enter dns to filter out all other protocols.

As shown in the figure below, when the client asks for the name "101lobs.net" (packet #532), the server responds indicating that there is no such name (packet #537).



Task 3:
Another cause of the DNS error is when the DNS server is unreachable or does not exist.

Open System Settings and click Network. This figure is taken from the Linux operating system.

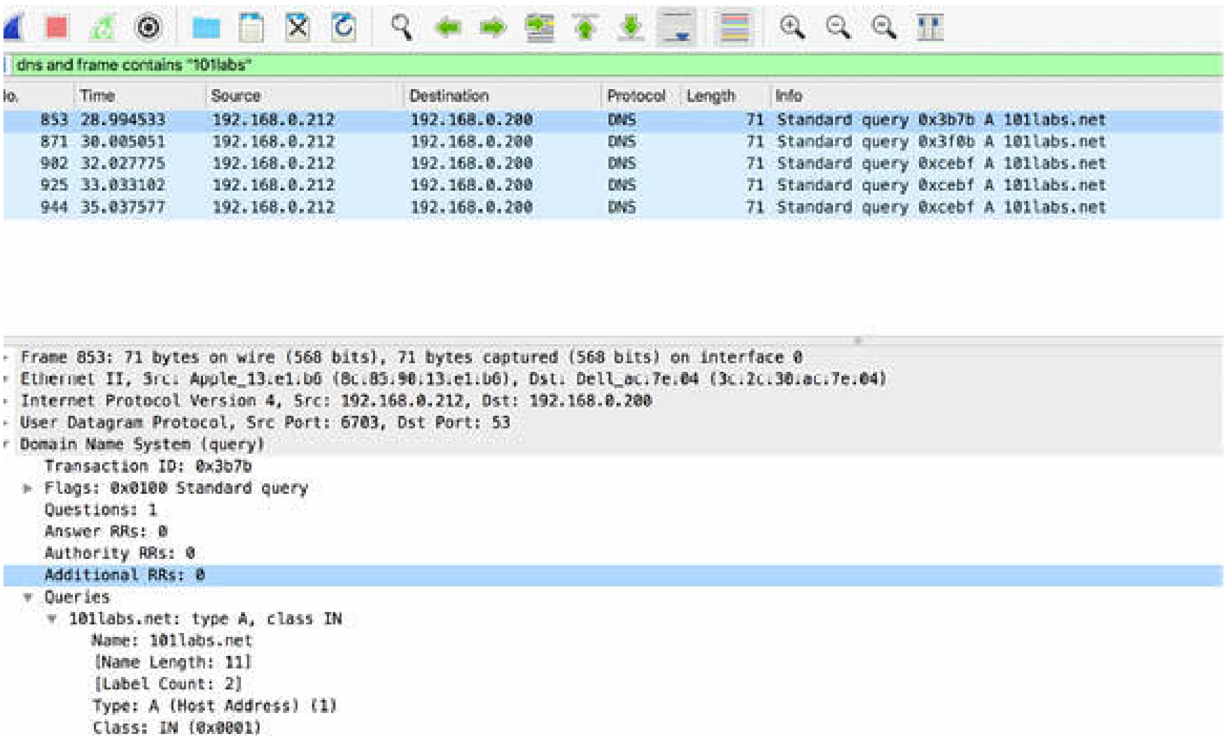


Select the IPV4 tab and add the DNS server IP address. If you are using Windows or another operating system then the steps will differ of course.

Task 4:

Start a capture again on the active interface. Go to **www.101labs.net** in the web browser. No page is displayed in the web browser because the DNS resolution is not possible.

Stop the capture and in the filter toolbar, enter dns and frame contains "101labs" to view only the DNS packets of interest.



Only five DNS requests are present and there are no answers from the server, as expected. Observe that the first two requests are sent at a time delay of 1 second. Before sending the third request, the client doubles its waiting time (i.e., 2 seconds).

Notes:

To simulate DNS problems and identify related packets in the capture from the client and the server, repeat the previous steps to send requests to existing DNS servers and non-existing DNS servers.

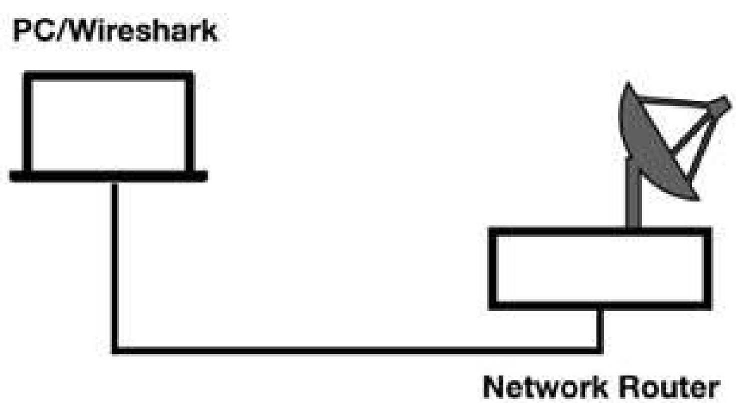
Lab 35. DNS Dissection

Lab Objective:
Learn how to analyze and recognize the structure of a DNS packet.

Lab Purpose:
Dissect the DNS packet and identify every field in the packet structure.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic

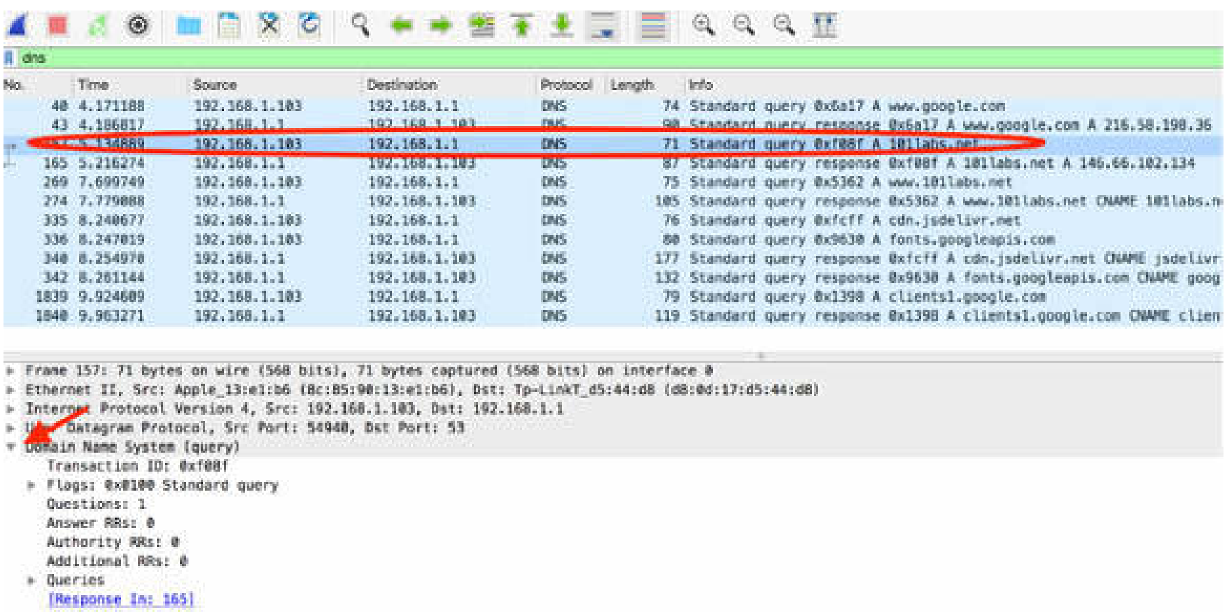
column. Capture the traffic for a few minutes.

In a web browser, go to www.101labs.net . Stop the capture and save the capture file.

Task 2:
In the filter toolbar, enter dns to filter out all other protocols.

DNS utilizes both UDP and TCP transport layers which makes it unique as compared to the other applications, which typically use only one transport layer. DNS typically uses UDP port 53 for name requests and responses and TCP port 53 for zone transfers and larger name requests and responses.

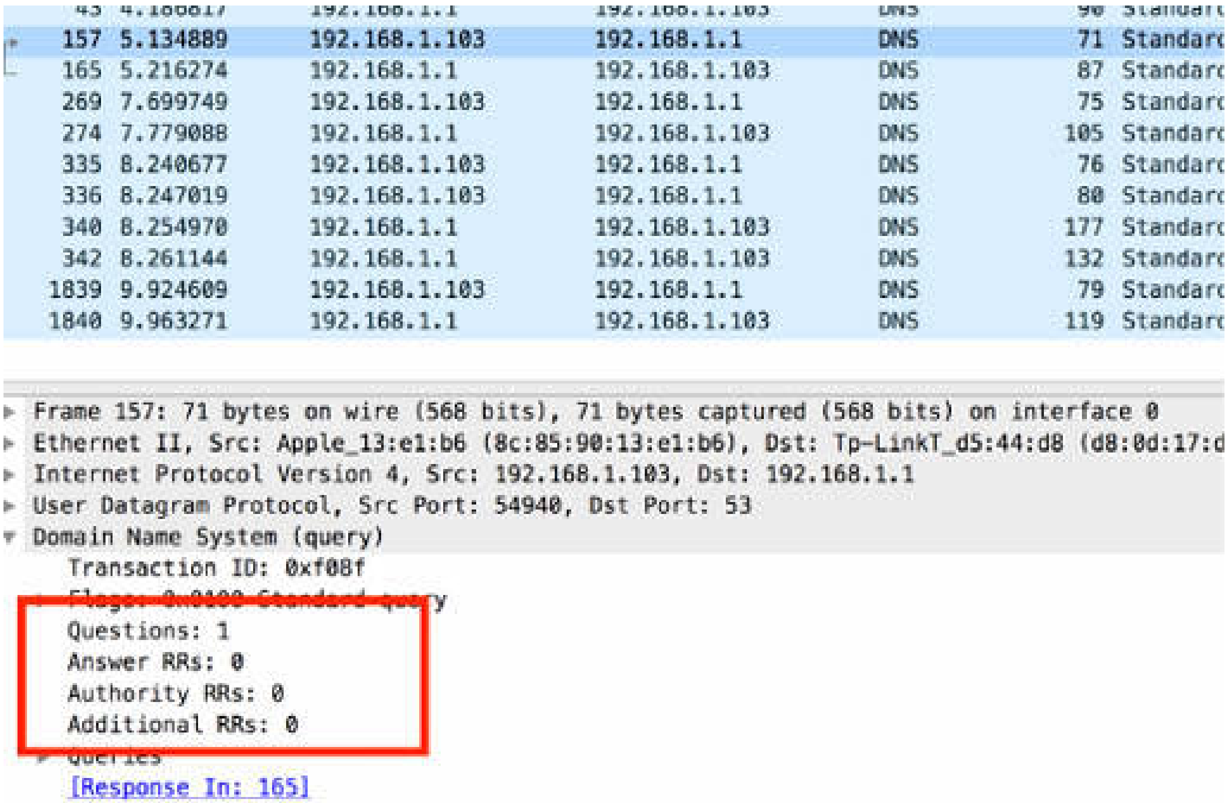
In the Packet List pane, select a DNS request. In the Packet Details pane, open the tree view, as indicated by the arrow in the figure below.



In the highlighted area shown in the figure below, you can see that all DNS packets use a single basic structure consisting of four primary parts:

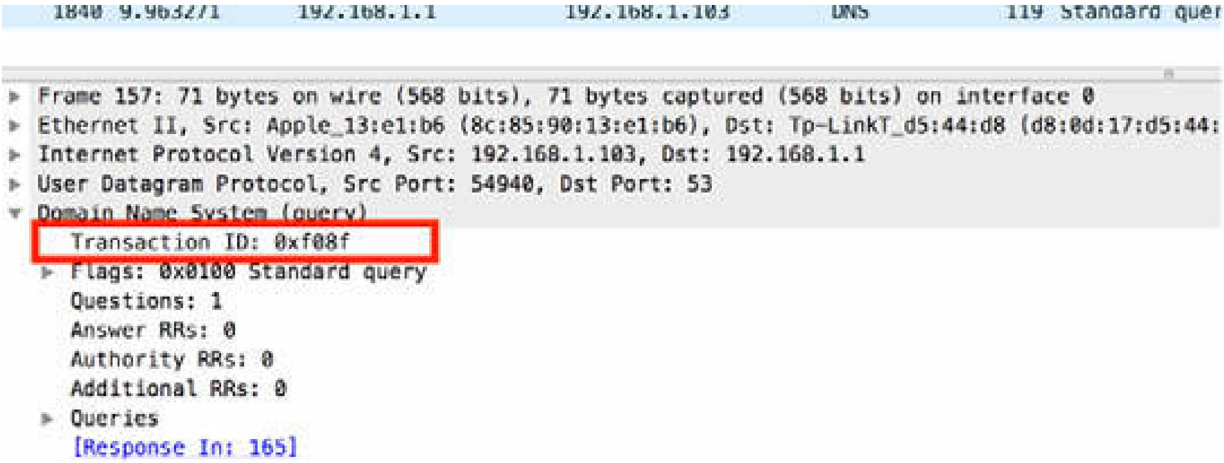
- Questions
- Answer Resource Records

- Authority Resource Records
- Additional Resource Records

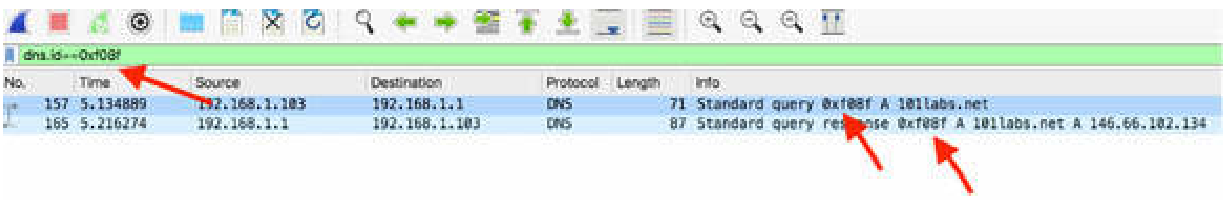


Task 3:

In the Packet Details pane, identify the first DNS field. It is the first field displayed in the DNS tree view and is named as Transaction ID, as shown in the figure below.



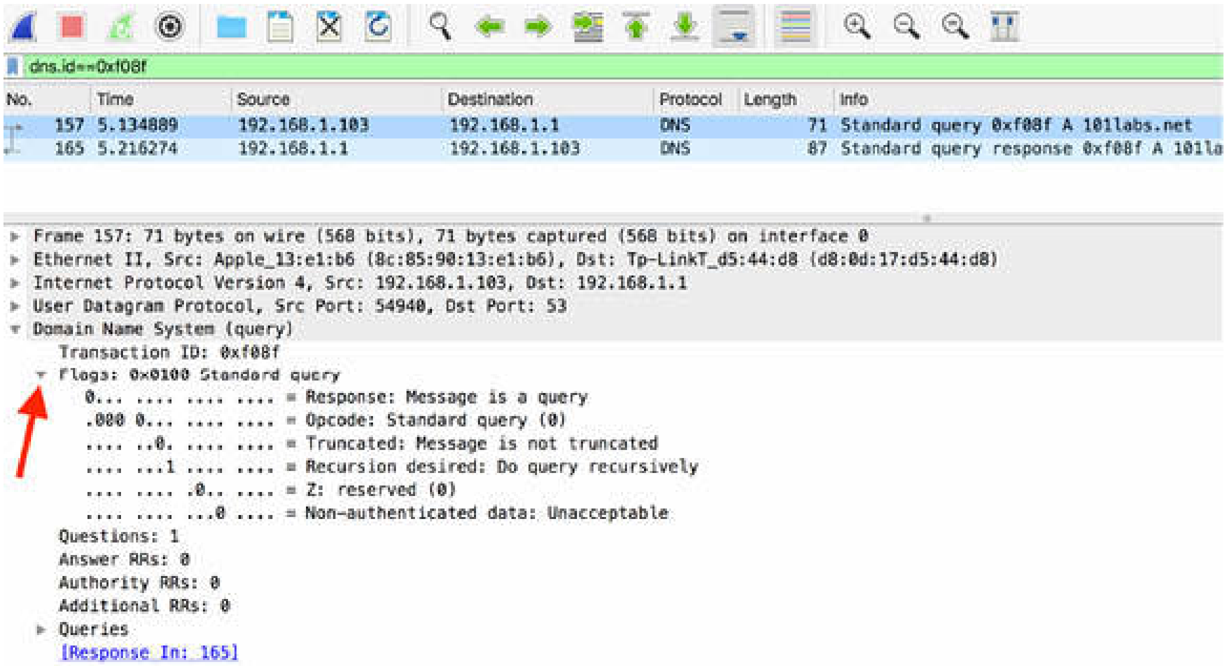
The Transaction ID field associates DNS queries with responses. You can filter out this field and its value (for example, dns.id==0xf08f) to view all associated DNS queries and responses. In the filter toolbar, enter dns.id==0xf08f to display only the messages associated with DNS.



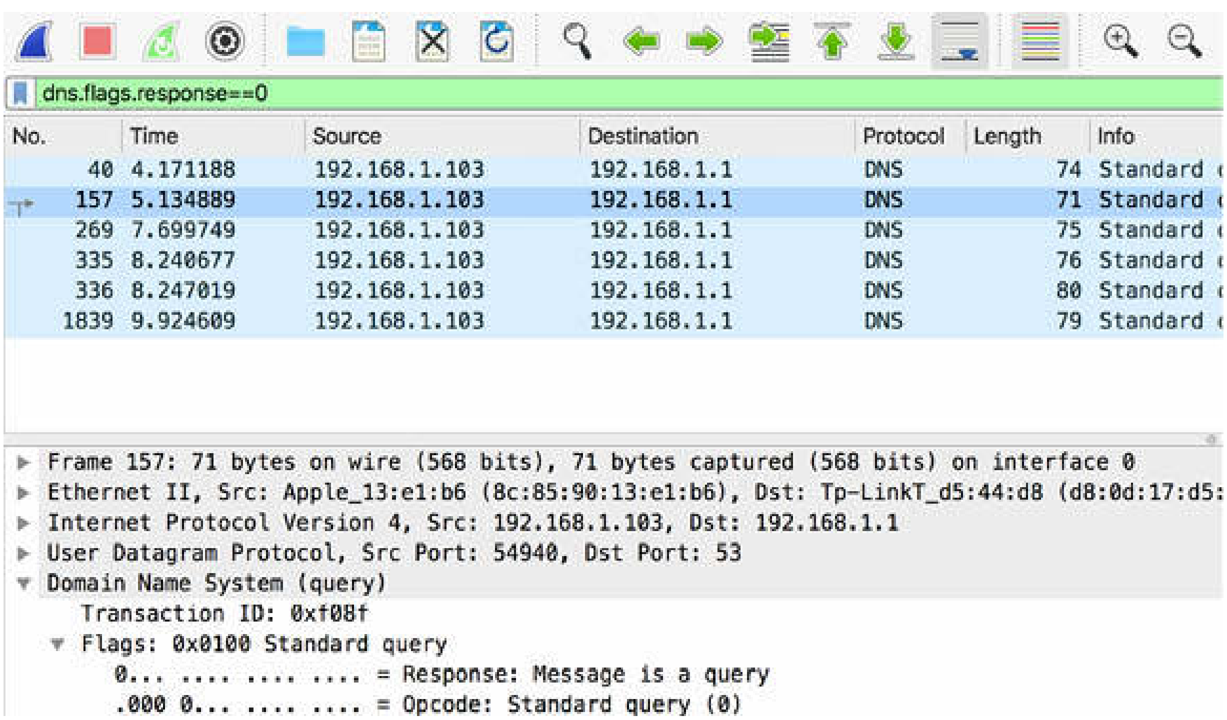
In the figure above, the Transaction ID field is displayed in the Info column to help in matching the DNS queries with their corresponding responses.

Task 4:

In the Packet Details pane, click the Flags field to open the tree view, as shown in the figure below.

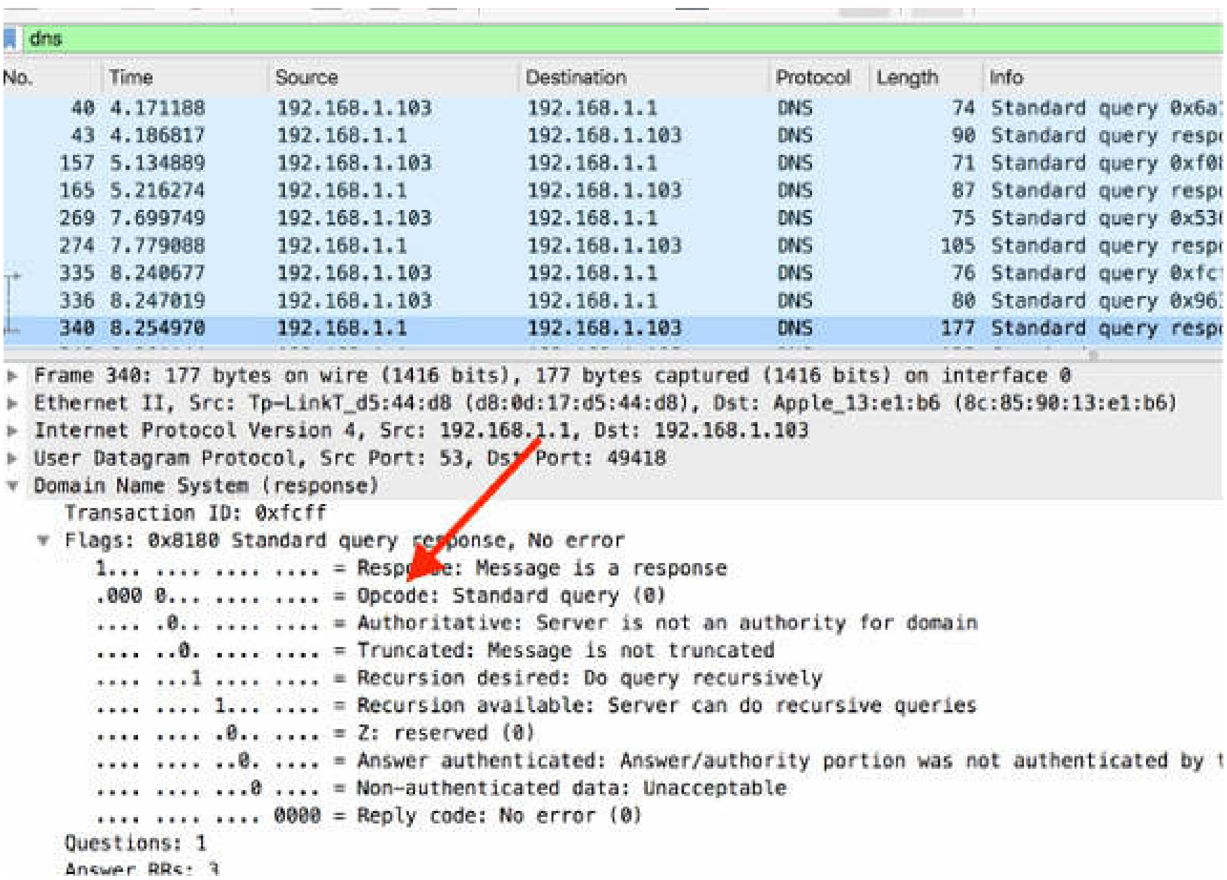


The Flags field consists of numerous fields that define the characteristics of the query. The Query/Response bit indicates whether the packet is a query (0) or a response (1). In the filter toolbar, enter dns.flags.response==0 to see only the DNS queries.

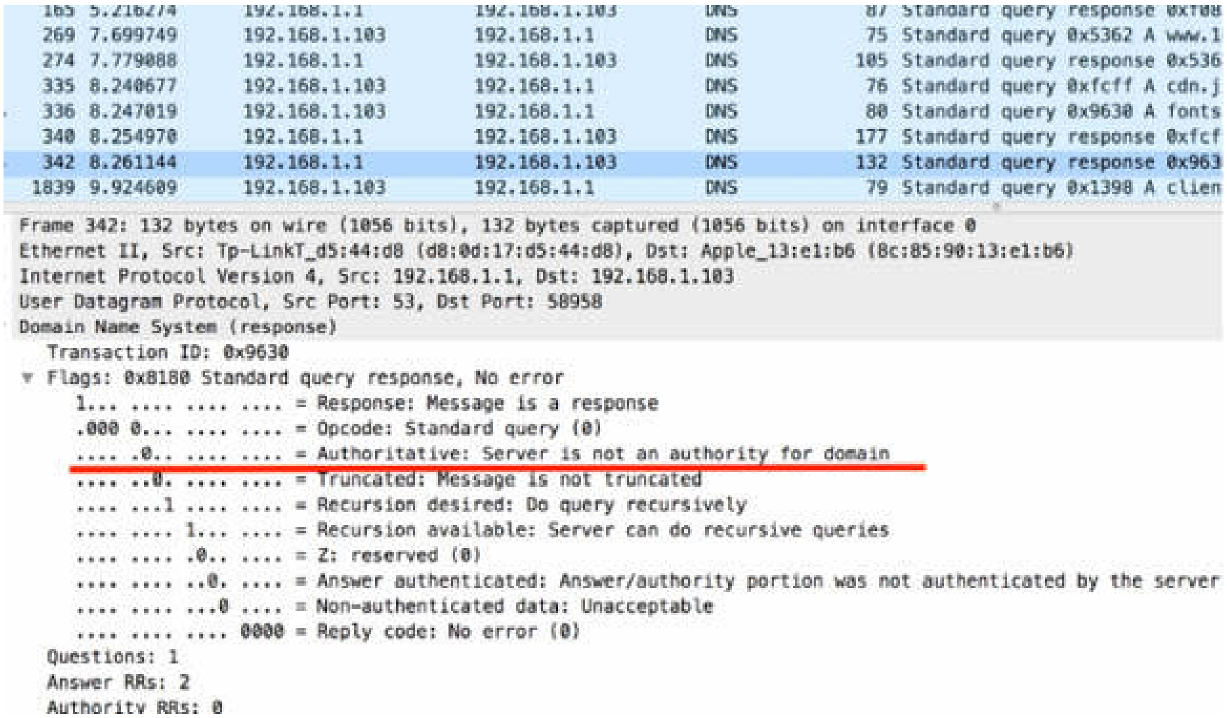


If you apply the display filter dns.flags.response==1 , only the DNS responses are displayed.

Task 5:
The Opcode field specifies the type of query, as shown in the figure below. For a standard query, it contains the value 0000, and the same value is left in the DNS response.

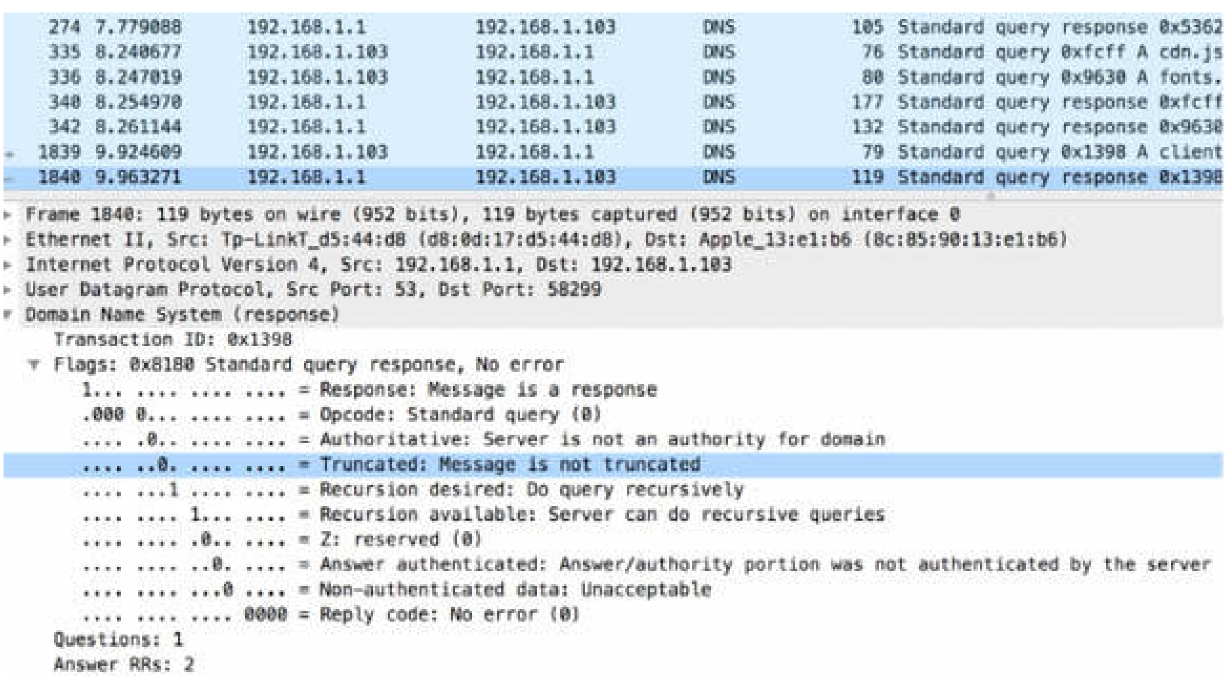


Task 6:
The Authoritative field indicates that the response is not from an authoritative server for the domain name.



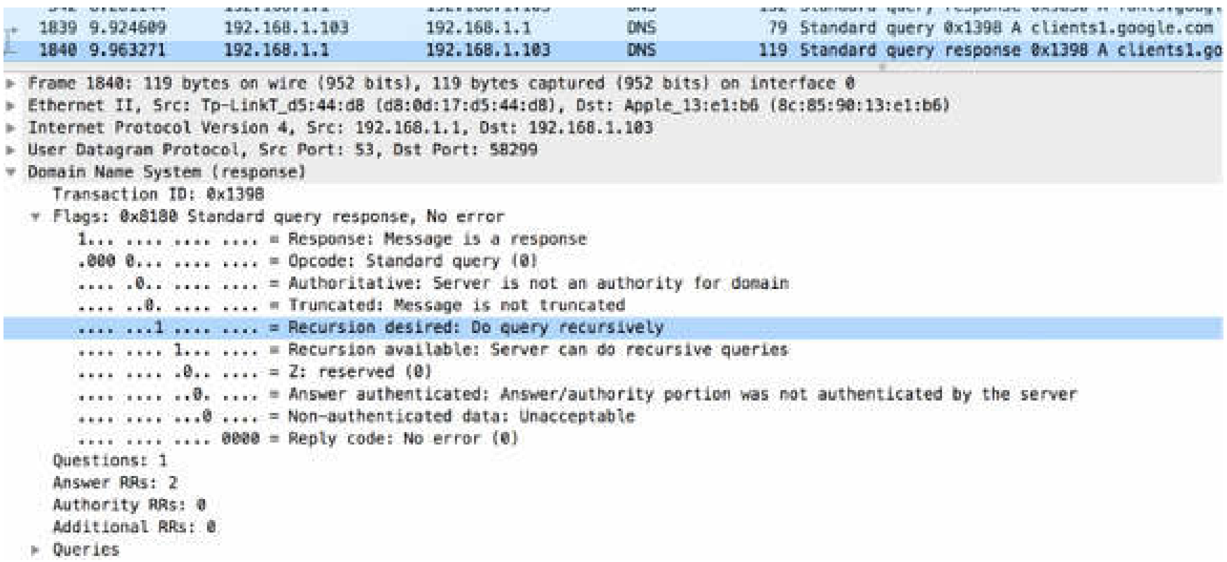
Task 7:

The Truncated field indicates whether the DNS response was truncated because of the length. This is not a common situation because if a client sees a truncated DNS response, it should retry the query over TCP.



Task 8:

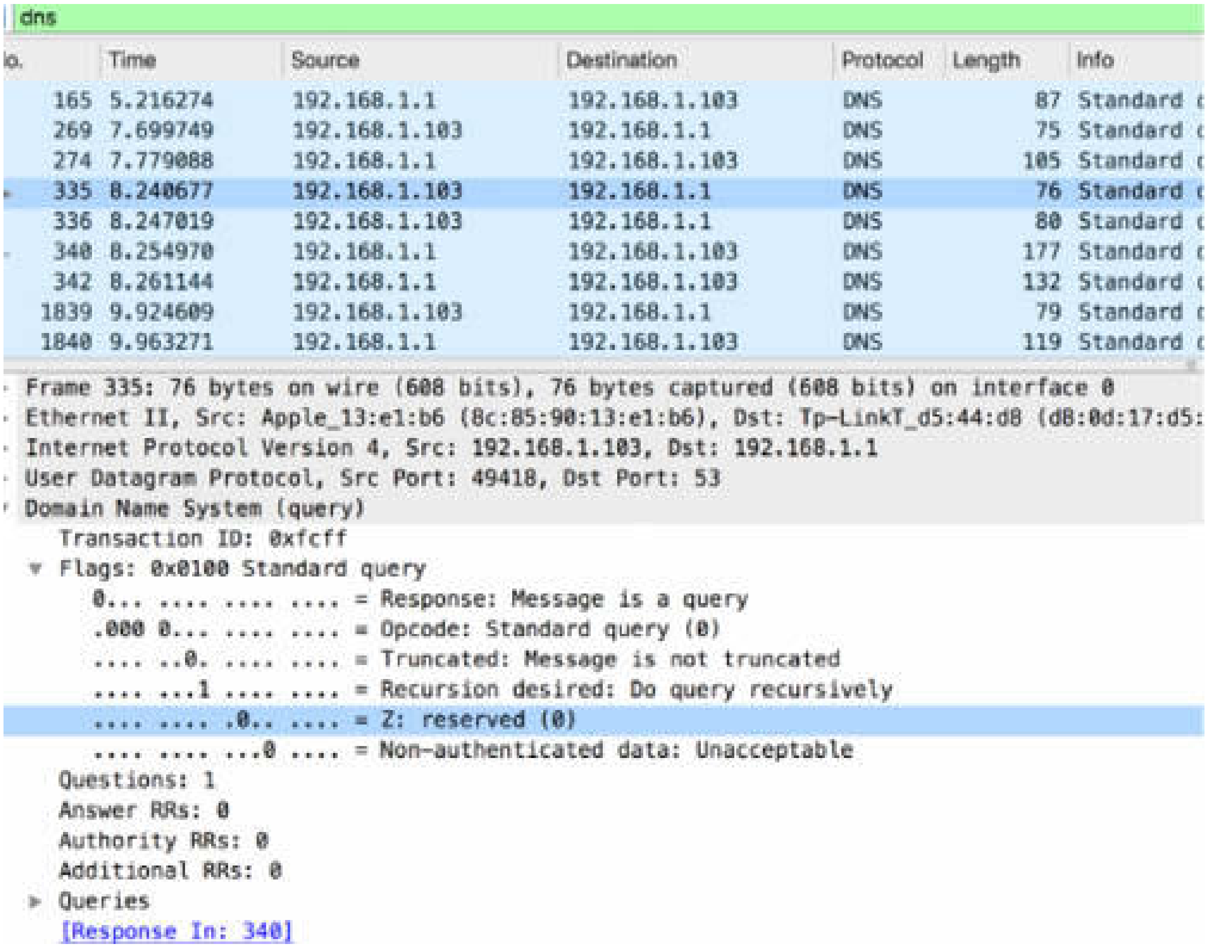
The Recursion desired field (present in DNS queries) indicates whether the server may use recursive query processes. Recursion allows a DNS server to query another server on the client's behalf. If the local name server has the answer, it replies directly. If the local name server does not have the answer, it begins the lookup process on behalf of the client.



The Recursion available field (present in the DNS response) indicates whether recursion is available at the DNS server.

Task 9:

The Reserved field is set at 0.

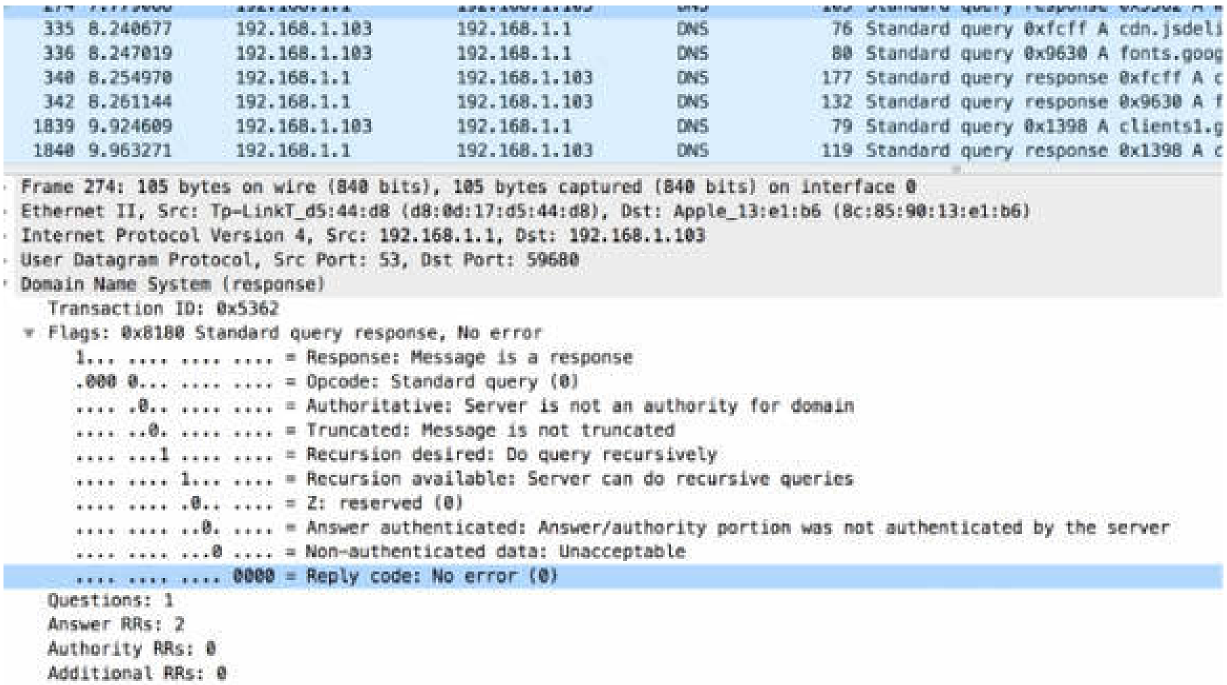


Task 10:

The Reply Code is the last field. It indicates whether an error condition exists in the response.

In the figure below, the code has the value No Error (0). Other possible values are:

- 1: Format error
- 2: Server failure
- 3: Name error
- 4: Not implemented
- 5: Refused


335 8.240677 192.168.1.103 192.168.1.1 DNS 76 Standard query 0xfcff A cdn.jsdeli
336 8.247019 192.168.1.103 192.168.1.1 DNS 80 Standard query 0x9630 A fonts.goog
340 8.254970 192.168.1.1 192.168.1.103 DNS 177 Standard query response 0xfcff A c
342 8.261144 192.168.1.1 192.168.1.103 DNS 132 Standard query response 0x9630 A f
1839 9.924609 192.168.1.103 192.168.1.1 DNS 79 Standard query 0x1398 A clients1.g
1840 9.963271 192.168.1.1 192.168.1.103 DNS 119 Standard query response 0x1398 A c
Frame 274: 105 bytes on wire (840 bits), 105 bytes captured (840 bits) on interface 0
Ethernet II, Src: Tp-LinkT_d5:44:d8 (d8:0d:17:d5:44:d8), Dst: Apple_13:e1:b6 (8c:85:90:13:e1:b6)
Internet Protocol Version 4, Src: 192.168.1.1, Dst: 192.168.1.103
User Datagram Protocol, Src Port: 53, Dst Port: 59680
Domain Name System (response)
Transaction ID: 0x5362
Flags: 0x8180 Standard query response, No error
1... .... = Response: Message is a response
.000 0... .... = Opcode: Standard query (0)
.... .0.. .... = Authoritative: Server is not an authority for domain
.... ..0. .... = Truncated: Message is not truncated
....1 .... = Recursion desired: Do query recursively
.... 1... .... = Recursion available: Server can do recursive queries
.... .0.. .... = Z: reserved (0)
....0. .... = Answer authenticated: Answer/authority portion was not authenticated by the server
....0 .... = Non-authenticated data: Unacceptable
.... 0000 = Reply code: No error (0)
Questions: 1
Answer RRs: 2
Authority RRs: 0
Additional RRs: 0

Notes:

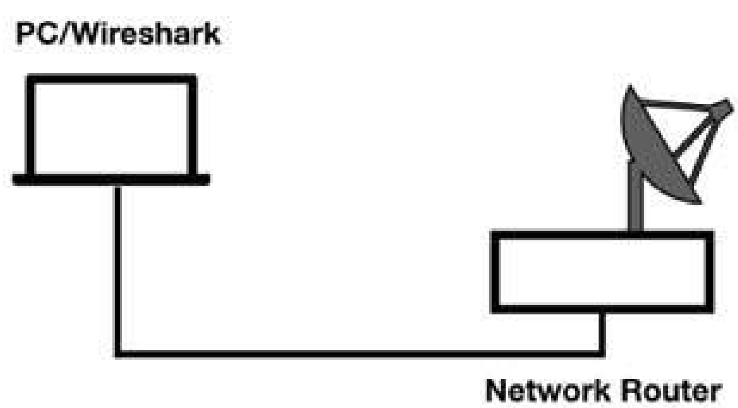
Lab 36. Address Resolution Protocol

Lab Objective:
Learn how the Address Resolution Protocol (ARP) works.

Lab Purpose:
Learn that ARP is used to associate a hardware address with an IP address on a local network.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

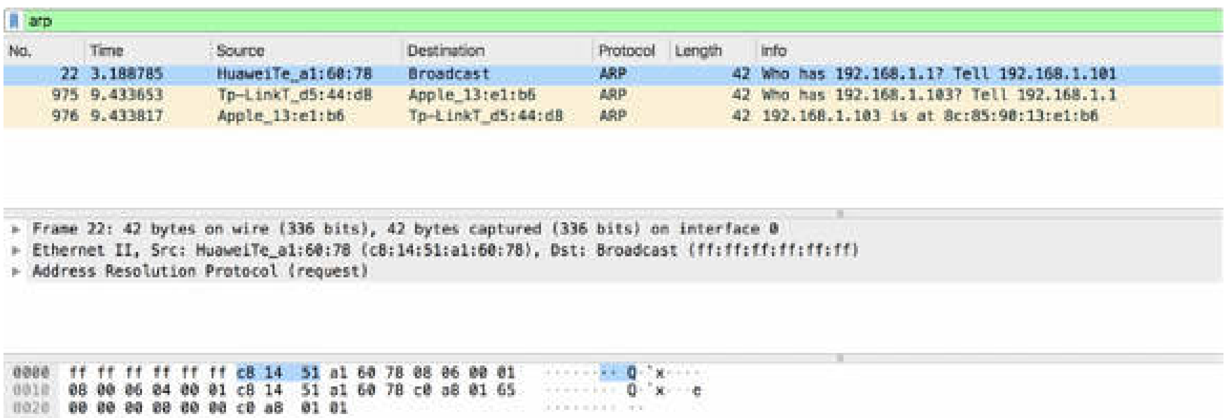
Task 1:
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

Power off and then power on your network router.

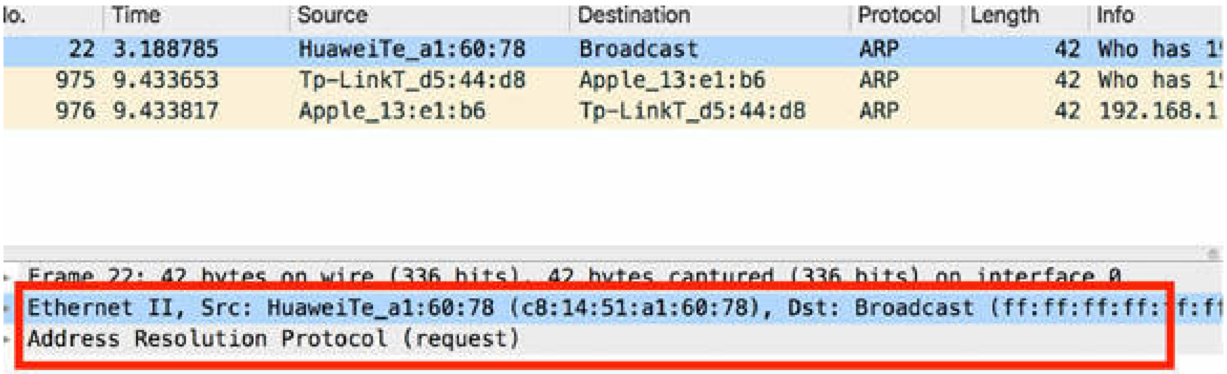
Stop the capture and save the capture file.

Task 2:
In the filter toolbar, enter arp to filter out all the other protocols.

In the Packet List pane, only ARP packets are displayed, as shown in the figure below.

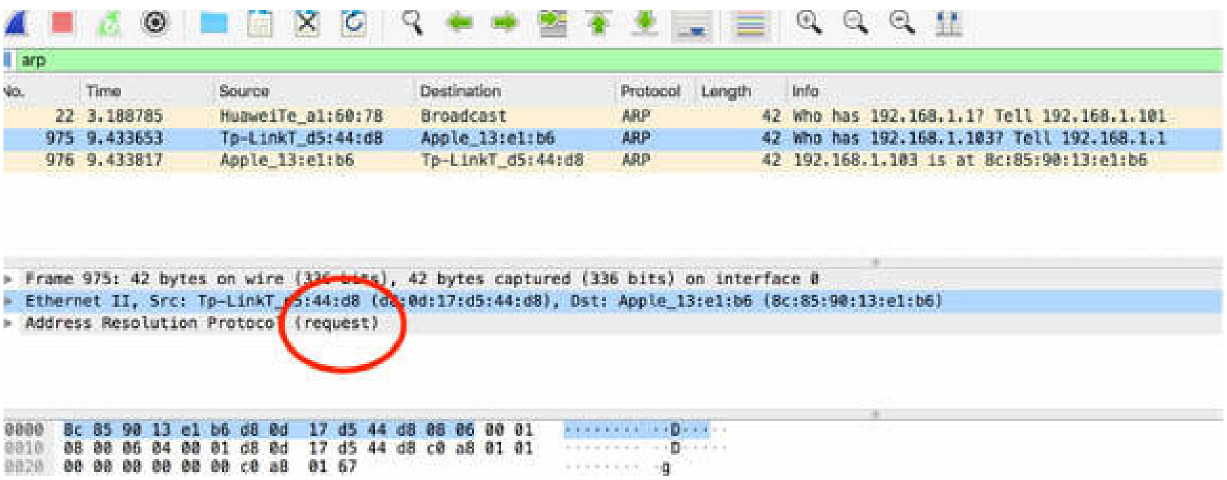


The peculiarity of ARP packets as compared to other applications in a TCP network is that they do not contain an IP header. This means that ARP packets are non-routable packets. As shown in the figure below, only Ethernet header and ARP details are displayed in the Packet Details pane.

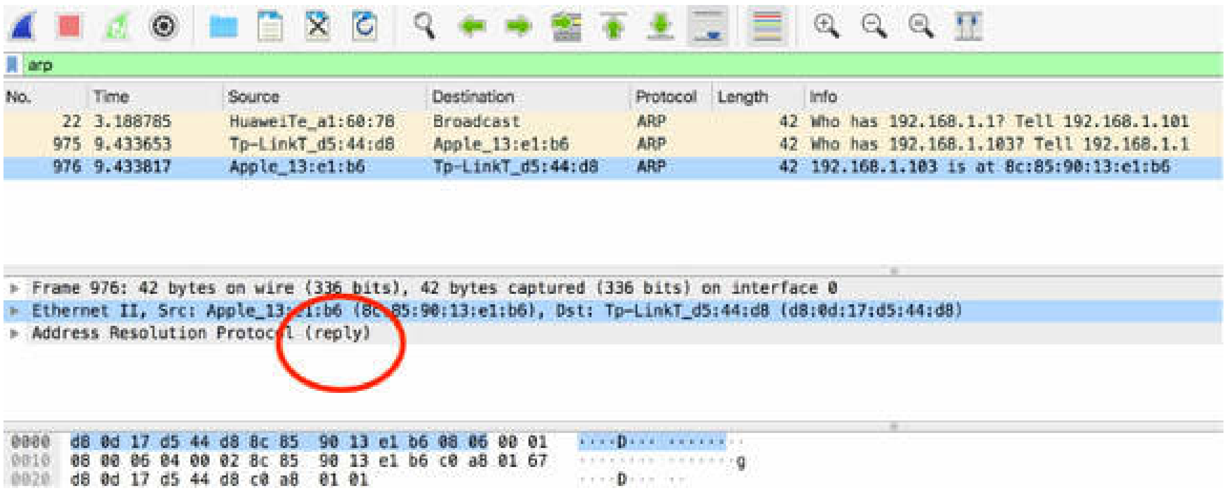


Task 3:
Normal ARP communication consists of a simple request and a simple response; for example, as shown for packets #975 and #976 in the figure below.

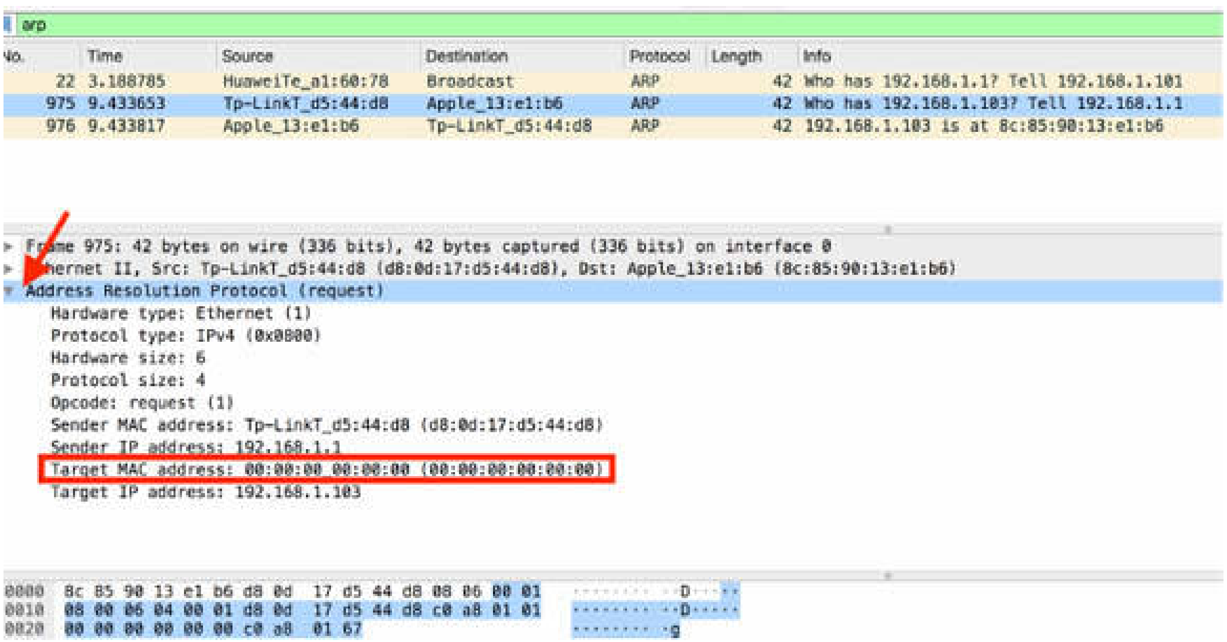
The request is the packet #975. You can see it in the Packet Details pane.



The response is the packet #976.

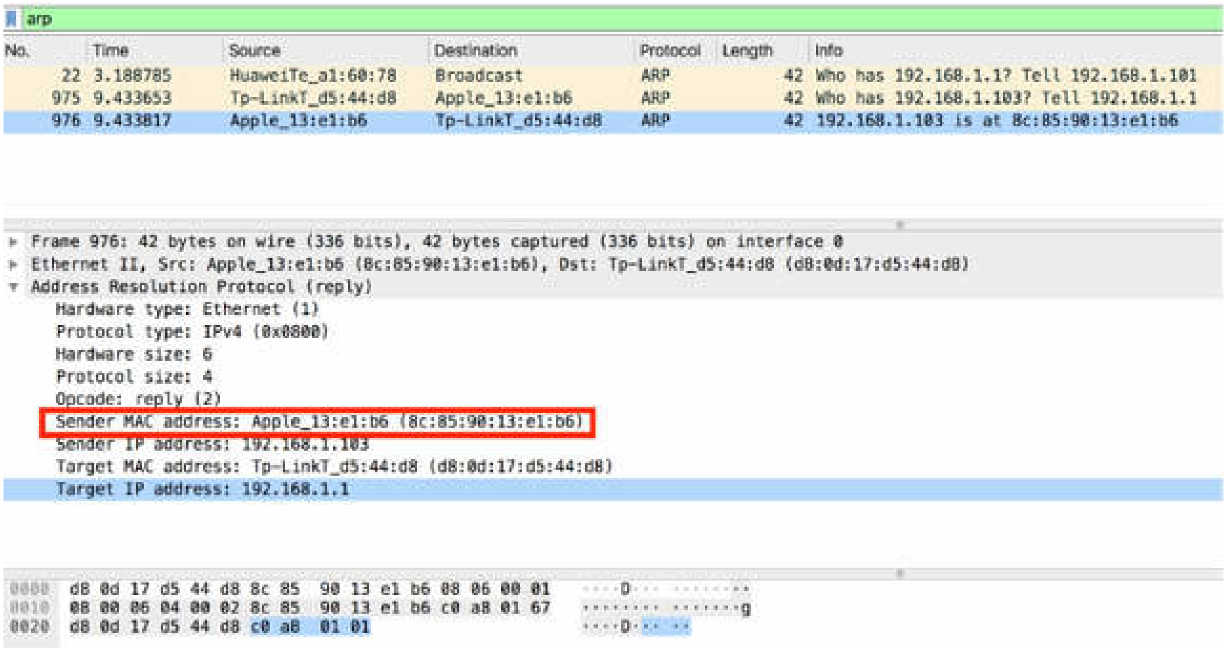


Task 4:
In the Packet List pane, select the request. In the Packet Details pane, open the tree view for ARP, as indicated by the arrow in the figure below.



As shown in the figure above, the host sends the request as a broadcast (containing the target IP address but not the target hardware address which has all zeros). The sender with HW MAC address d8:0d:17:d5:44:d8 and IP address 192.168.1.1 is looking for the HW MAC address associated with the IP address 192.168.1.103.

In the Packet List pane, click the response packet (#976).



As shown in the figure above, the response packet contains the HW MAC address (8c:85:90:13:e1:b6) associated with the IP address 192.168.1.103.

Response is now sent directly to the HW MAC address that sent the ARP request, that is, Target MAC address: Tp-LinkT_d5:44:d8 (d8:0d:17:d5:44:d8).

Notes:
To understand how the ARP message exchange works and to identify the addresses (IP and MAC) associated with requests and responses, repeat the previous steps with different hosts connected to your local network.

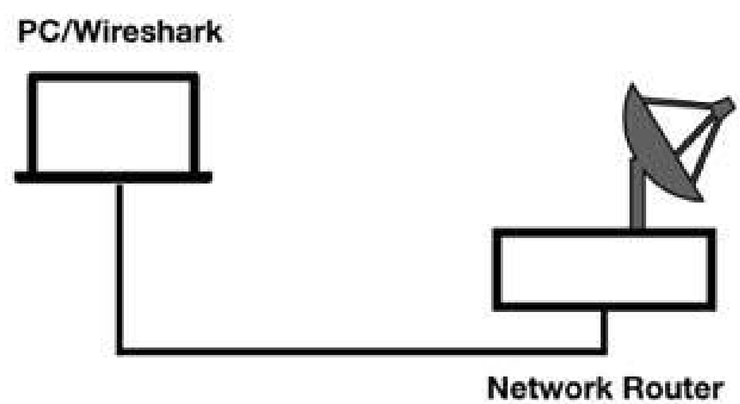
Lab 37. Gratuitous ARPs and Possible Problems

Lab Objective:
Learn about gratuitous ARPs and their purpose and how ARP issues can cause network problems.

Lab Purpose:
Learn how to identify gratuitous ARPs in a network capture and what network problems can occur when the ARP process is affected by issues.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

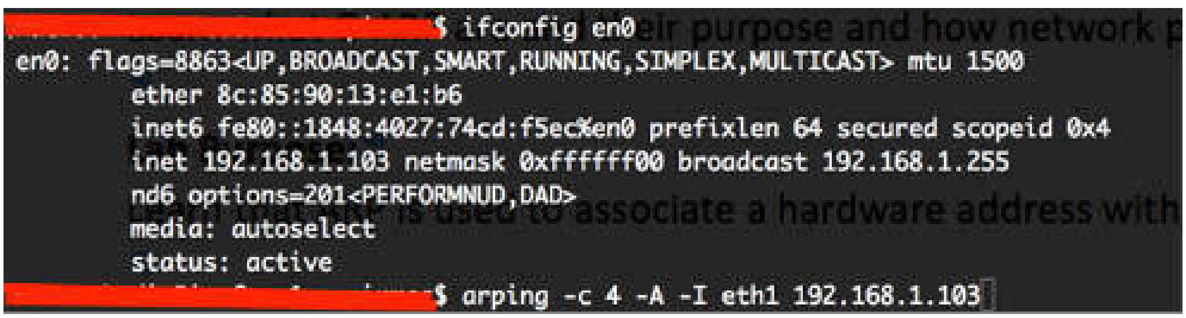


Lab Walkthrough:

Task 1:
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

Open a terminal window, and run the command ifconfig en0 to get the associated IP address, where en0 is the active network interface. This is for Linux of course.

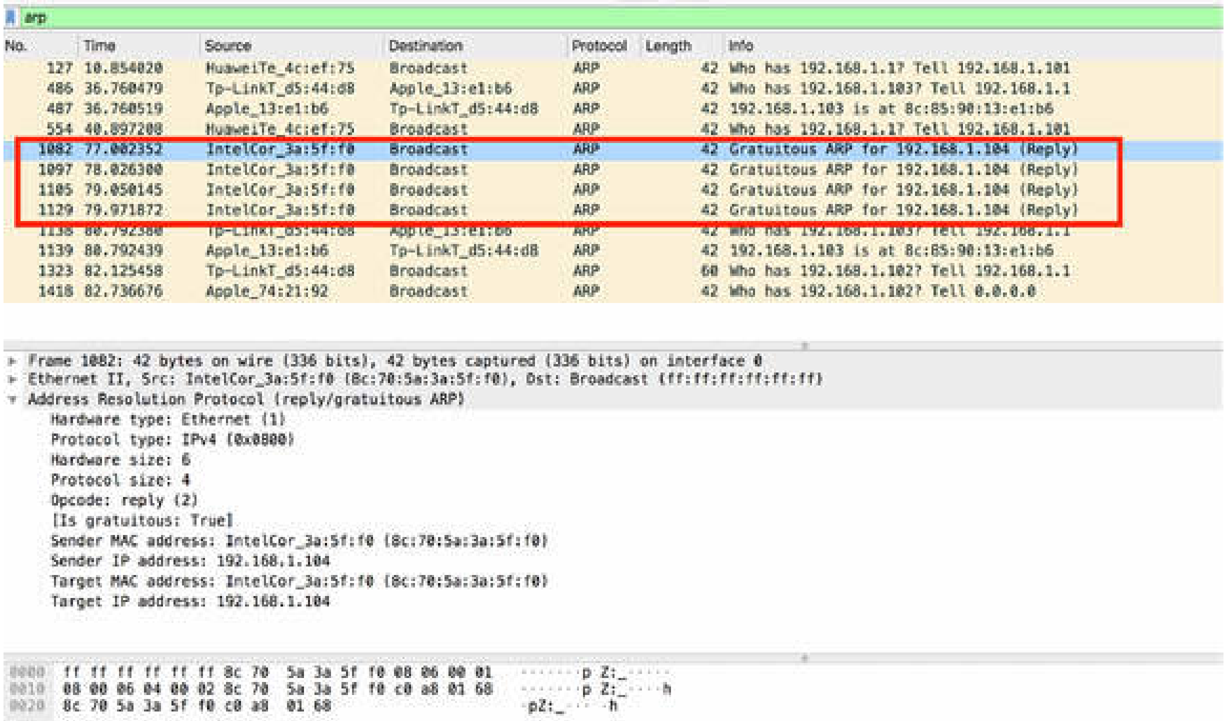
Run the command arping –c 4 -A –I en0 192.168.1.103 to send some gratuitous ARPs through the network, as shown in the figure below. This command sends the gratuitous ARP reply, four times, on the en0 interface for IP address 192.168.1.103.



Stop the capture in Wireshark and save the file.

Task 2:
In the filter toolbar, enter arp to filter out all other protocols.

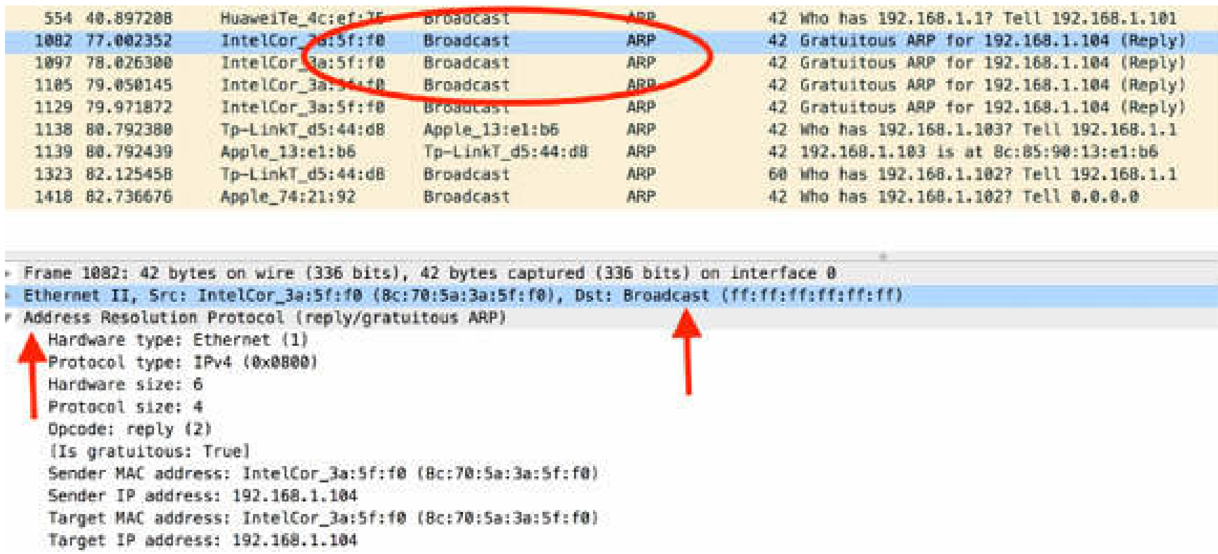
In the Packet List pane, only ARP packets are displayed, as shown in the figure below.



As shown in the figure above, the capture contains four consecutive gratuitous ARP replies—one reply every one second, as expected. They are used to determine if another host on the network has the same IP address as the sender (not applicable in this case because no answer is detected on the capture). Wireshark can identify gratuitous ARP packets and mark the ARP packet as gratuitous in the Info column, as shown in the figure above.

Task 3:

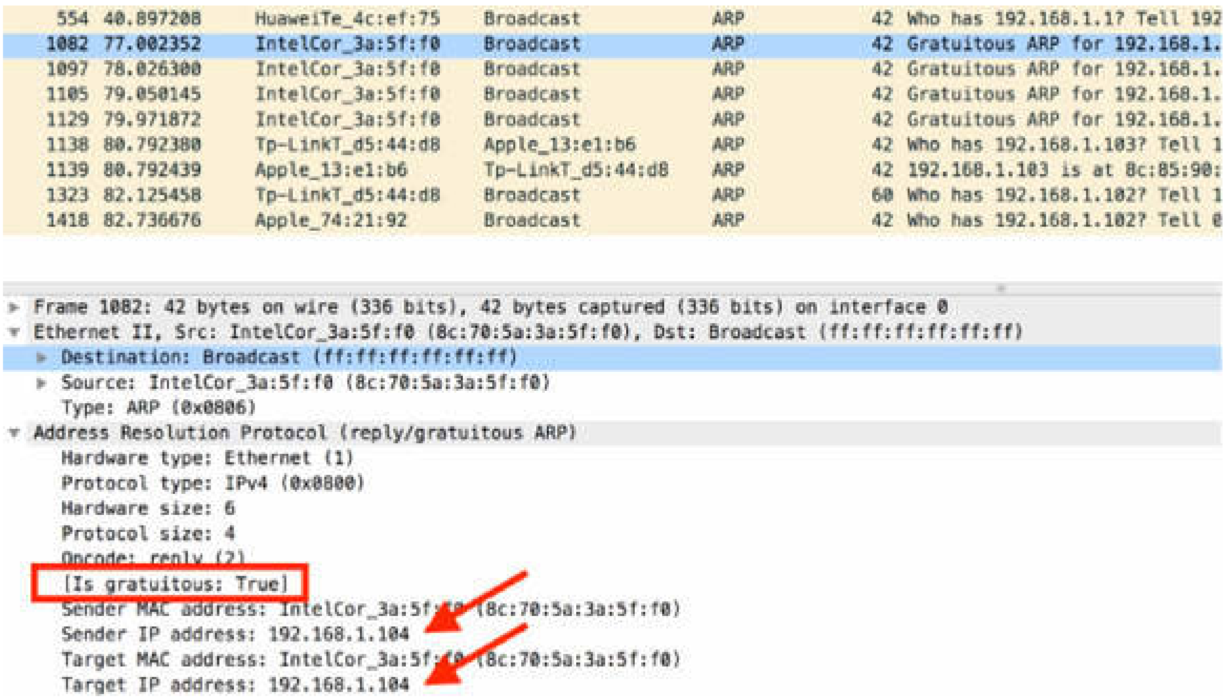
Select the first ARP in the capture file (packet #1082) and enable the tree view in the Packet Details pane, as shown in the figure below.



At Ethernet level, note that the destination MAC address is broadcast ff:ff:ff:ff:ff:ff—the message sent to all network devices to reach all of them, and the target IP address is 192.168.1.104—the IP address that is searched on the network.

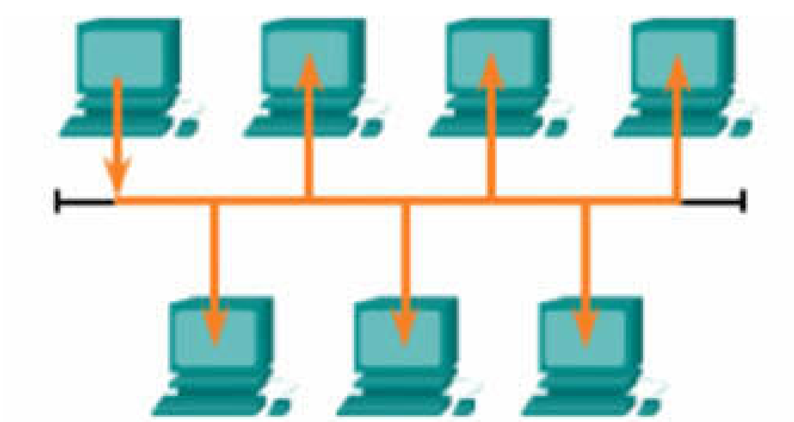
If a gratuitous ARP receives a response, another host is using the desired IP address. This typically generates a duplicate IP address alert on that host which stops the IP address initiation process.

Wireshark can detect that an ARP is gratuitous because the sender’s IP address is the same as the target IP address, as shown in the figure below:



Task 4:

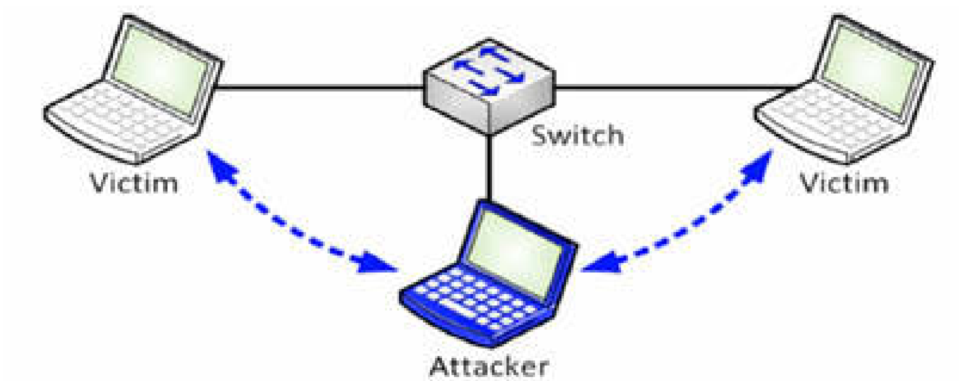
The figure below shows a scenario that can cause a potential issue related to ARP. Considering that the ARP request is a broadcast frame, it is received and processed by every device on the local network.



For example, when a large number of devices are powered up and all start accessing the network services at the same time, there could be some reduction in performance for a short duration.

Only after the devices send out the initial ARP broadcasts and have learned the necessary MAC addresses, the impact on the network is minimized.

Task 5:
Another issue related to ARP is the ARP spoofing technique. ARP spoofing is a technique that is used by hackers to inject a wrong MAC address association into a network. This is done by issuing fake ARP requests. An attacker creates the MAC address of a device so that the packets can be sent to the wrong destination, as shown in the figure below.



A way to prevent ARP spoofing is by manually configuring the static ARP associations. Another way is to grant network access to only authorized MAC addresses to specific devices.

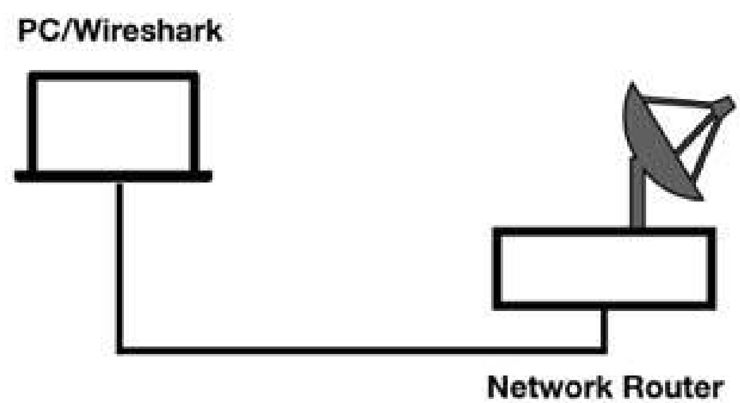
Lab 38. ARP Packet Structure

Lab Objective:
Learn how to recognize and analyze an ARP packet's structure.

Lab Purpose:
Dissect an ARP packet and identify its fields and structure.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

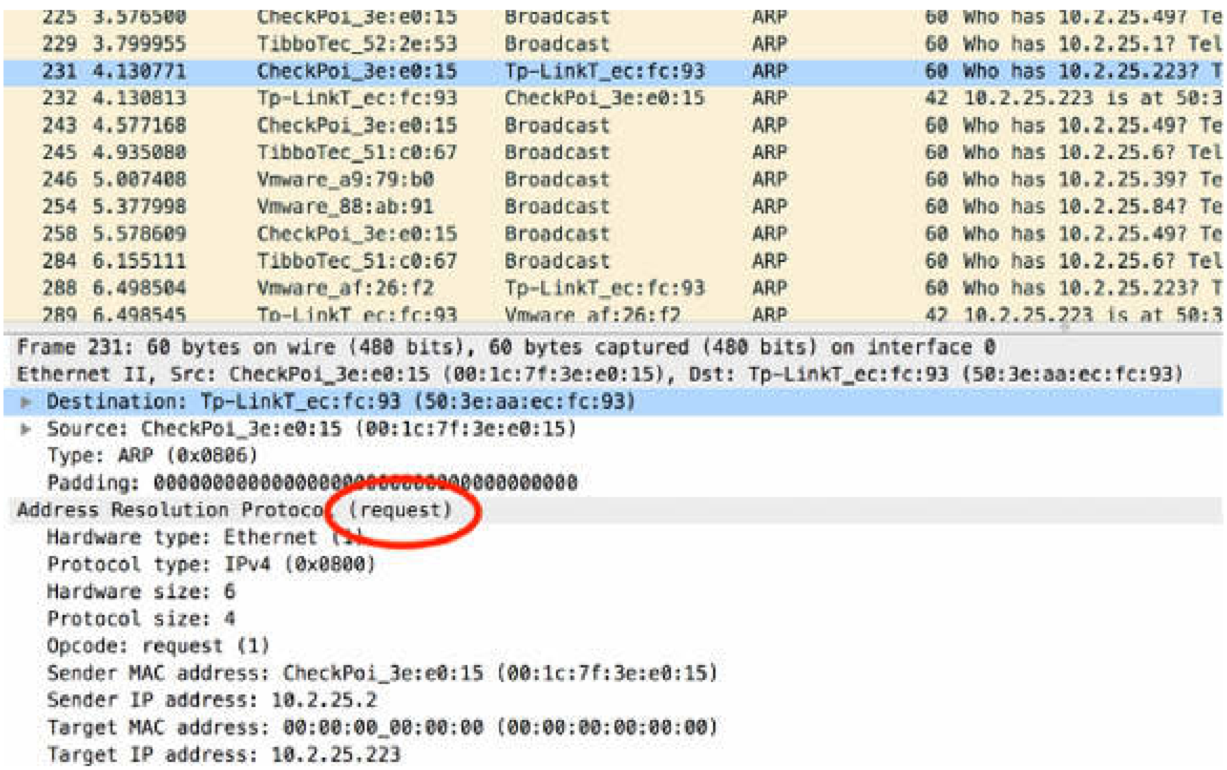
Task 1:
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic

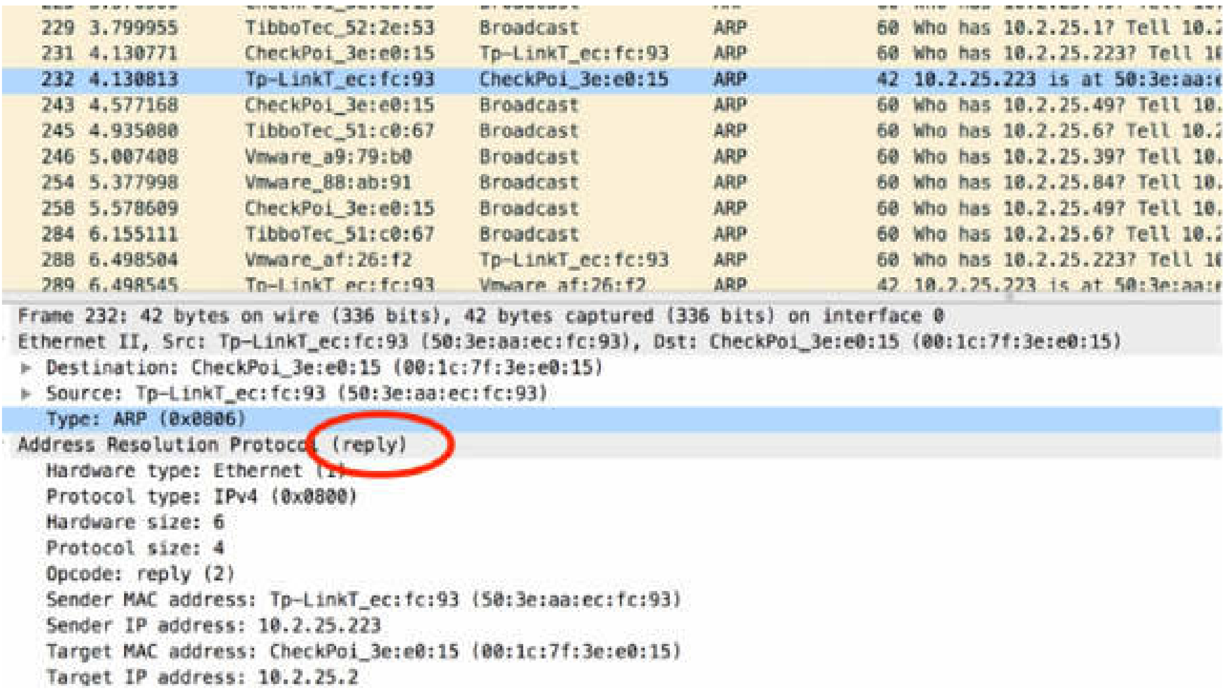
column. Capture the traffic for a few minutes.

Stop the capture and save the file.

In the filter toolbar, enter arp to filter out all other protocols. In the Packet List pane, only ARP packets are displayed.

As shown in the following two figures, the basic ARP packets are of only two types: the ARP request packet and the ARP reply packet. Both types use the same typology format.

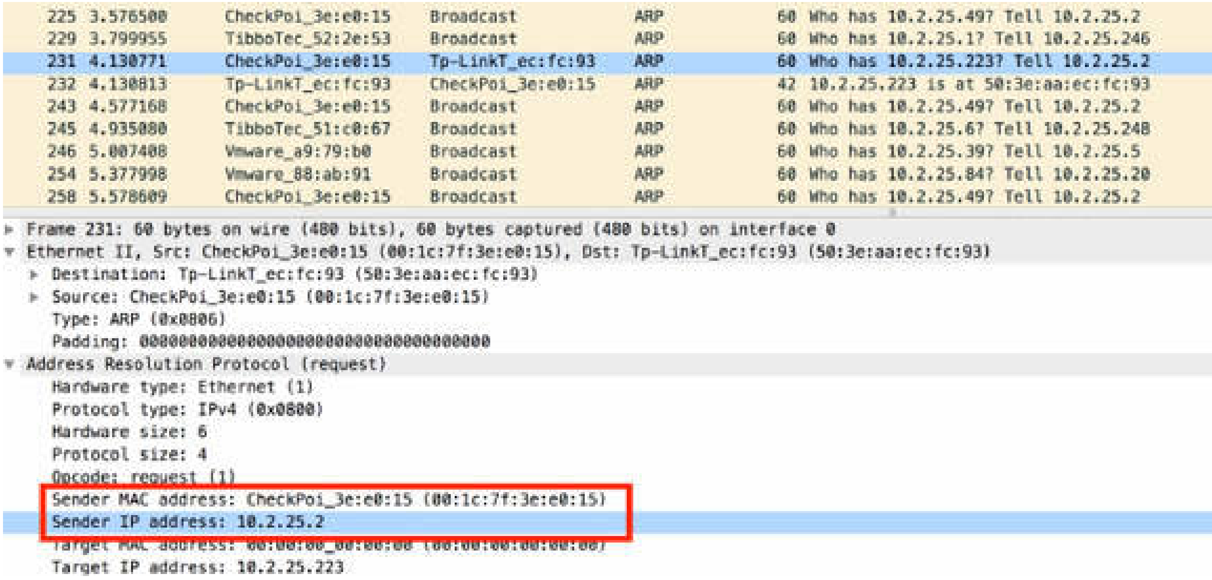




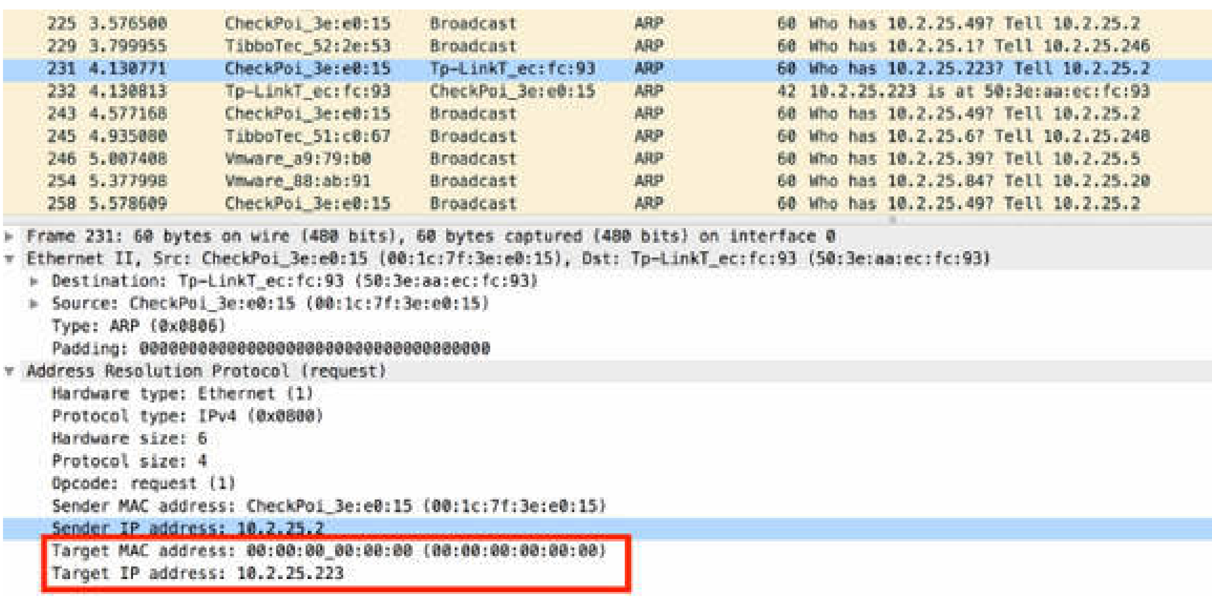
Task 2:

The part on which you need to pay attention is the sender and target address information of a broadcast ARP. In the Packet List pane, select an ARP request packet (#231). In the Packet Details pane, enable the tree view for Ethernet II and ARP, as shown in the figure below.

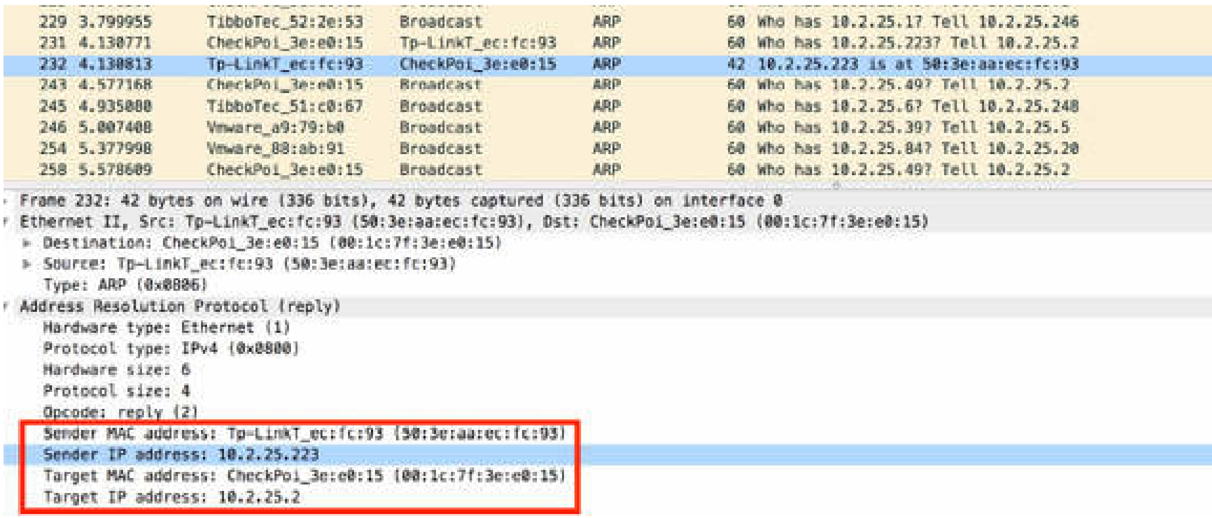
When an ARP broadcast is being sent from a host, the sending host puts its hardware and IP address in the sender address fields.



The target protocol address field is populated with the IP address of the searched device. The target hardware address field is set to all 0’s (00:00:00:00:00:00) to indicate that the information is not known.

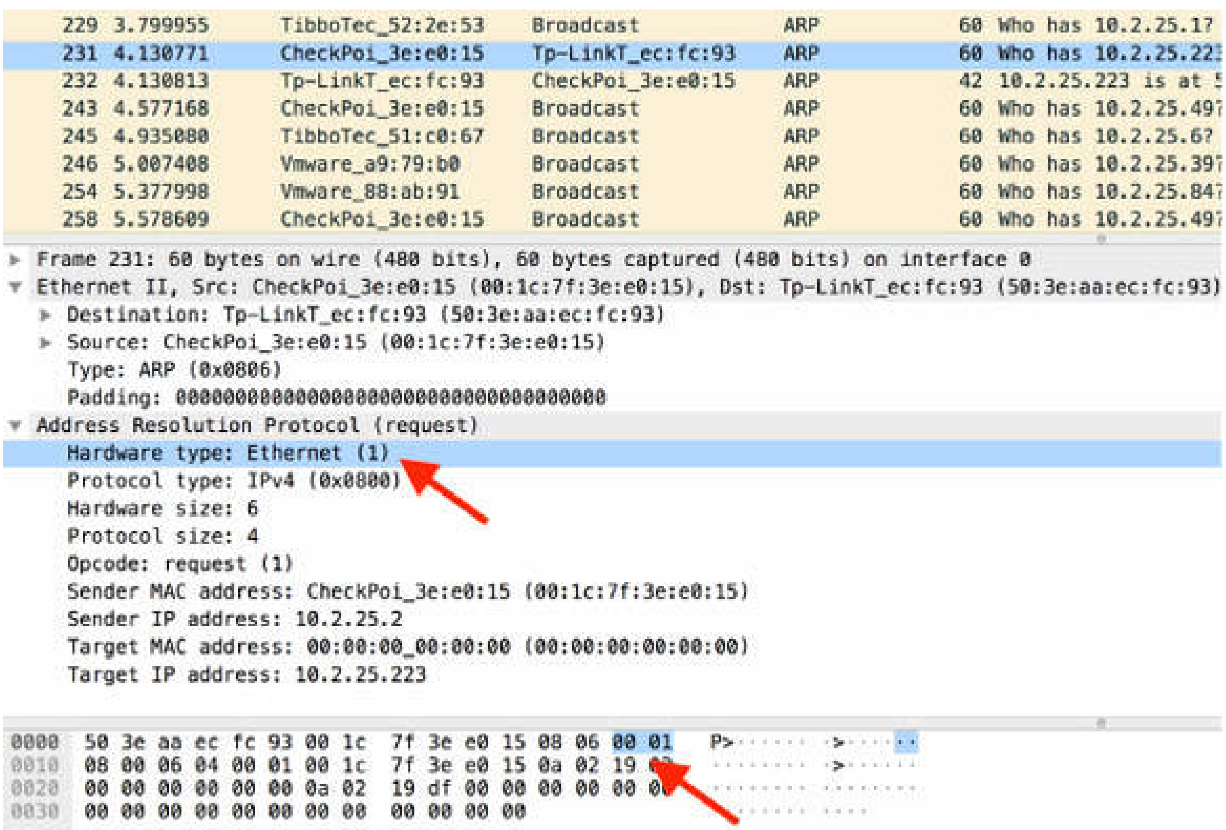


In an ARP reply, the target and the sender information is reversed to show that the ARP responder is now the sender.



Task 3:

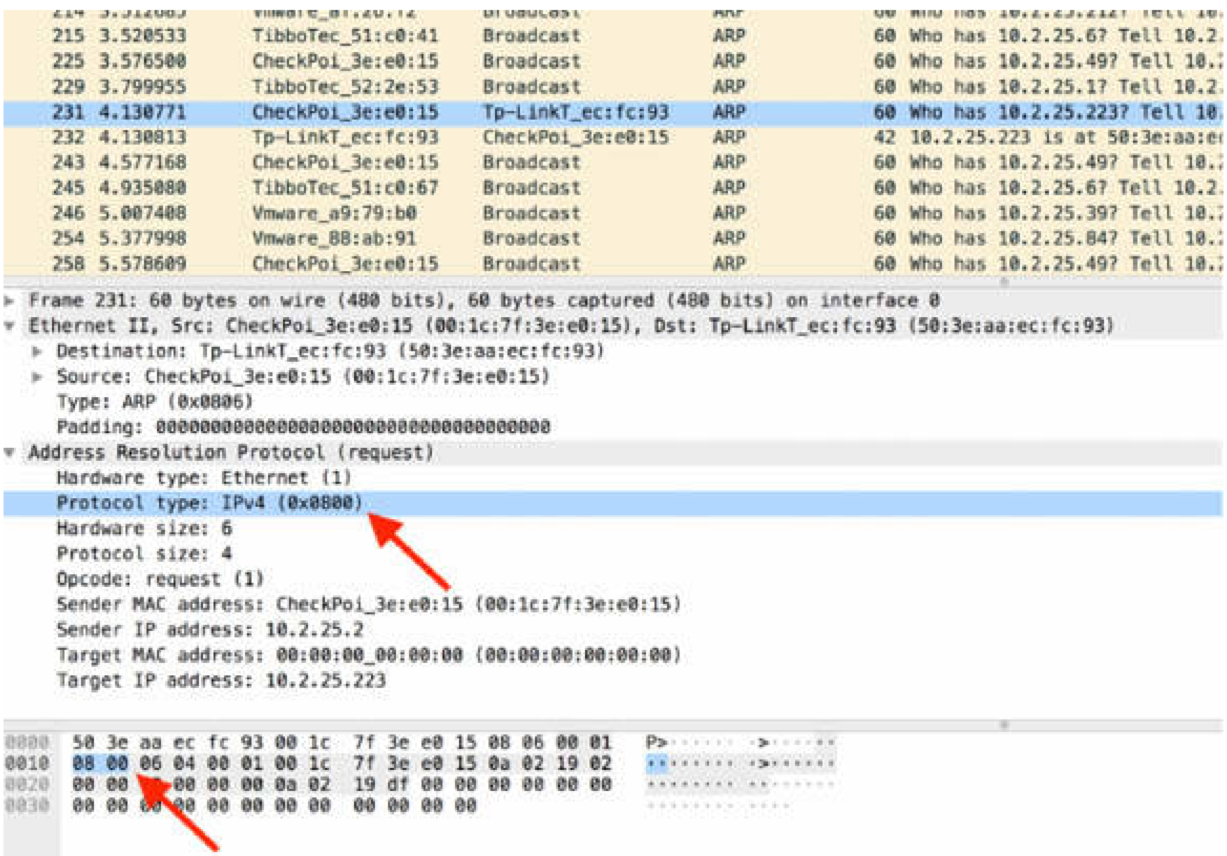
The “Hardware type” field is the first field of an ARP message. In the Packet Details pane, select the “Hardware type” field by using the mouse. The corresponding bytes are highlighted in the Packet Bytes pane, as shown in the figure below.



The “Hardware type” field defines the hardware or the data link type in use. The value 1 indicates that the hardware type is Ethernet, and it has a 6-byte hardware address length.

Task 4:

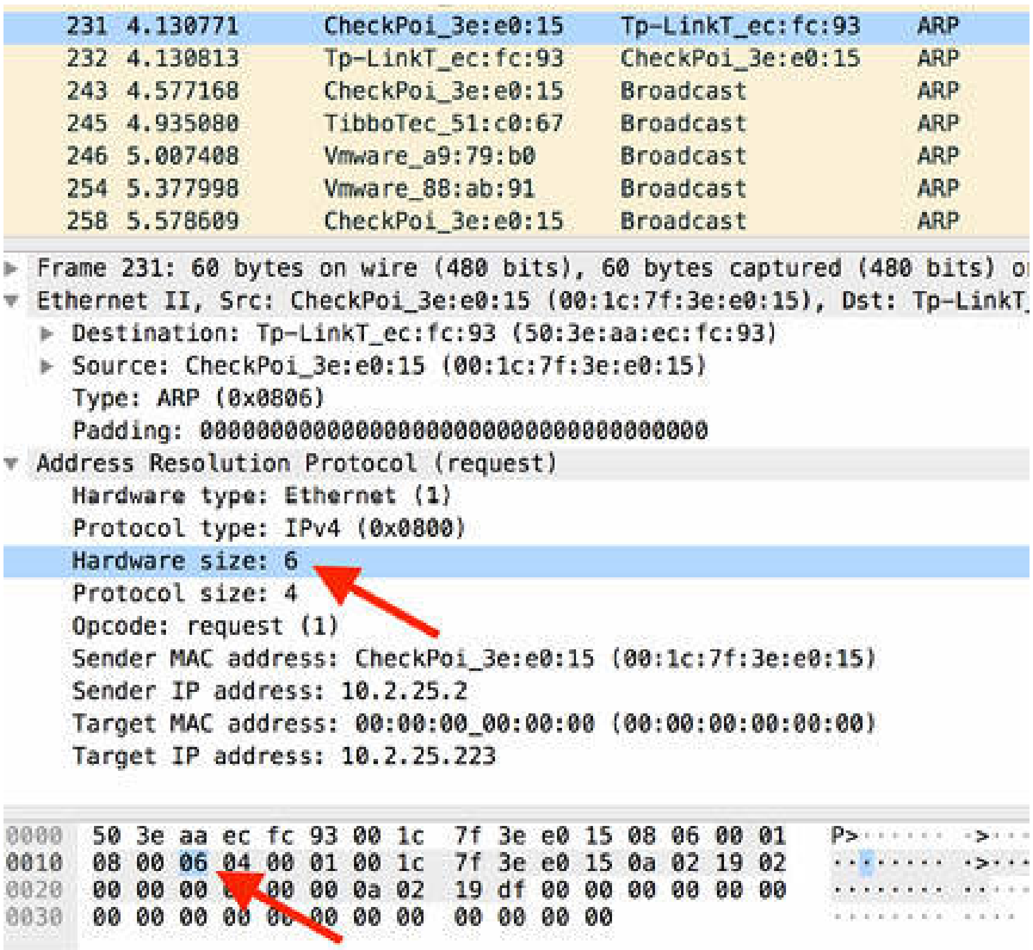
The “Protocol type” is the second field of an ARP message. In the Packet Details pane, select the “Protocol type” field by using the mouse. The corresponding bytes are highlighted in the Packet Bytes pane, as shown in the figure below.

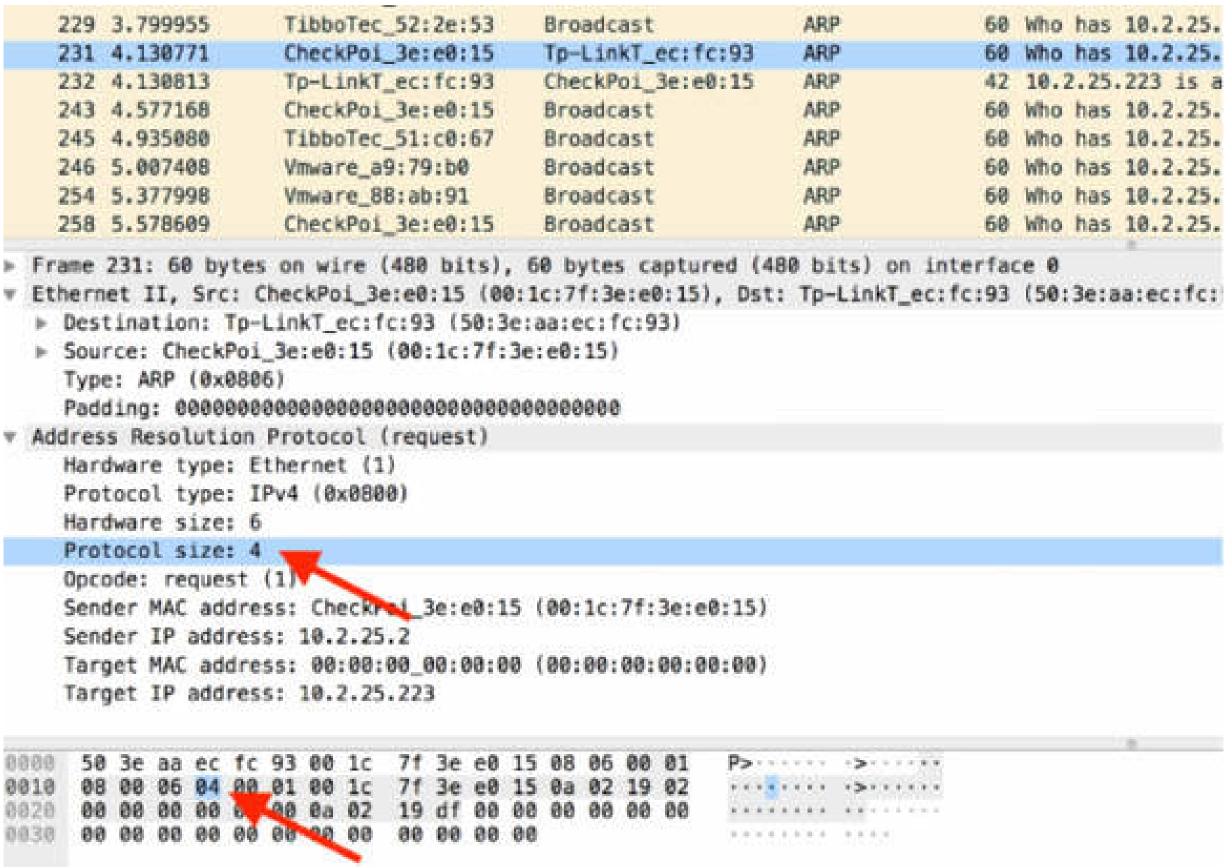


The “Protocol type” field defines the protocol address type in use. It uses the standard protocol ID values that are also used in the Ethernet II frame structure. In this case, the value is equal to IPv4.

Task 5:

The “Hardware size” and “Protocol size” fields are the next fields. In the Packet Details pane, select the “Hardware size” and “Protocol size” fields by using the mouse. The corresponding bytes are highlighted in the Packet Bytes pane, as shown in the figures below.





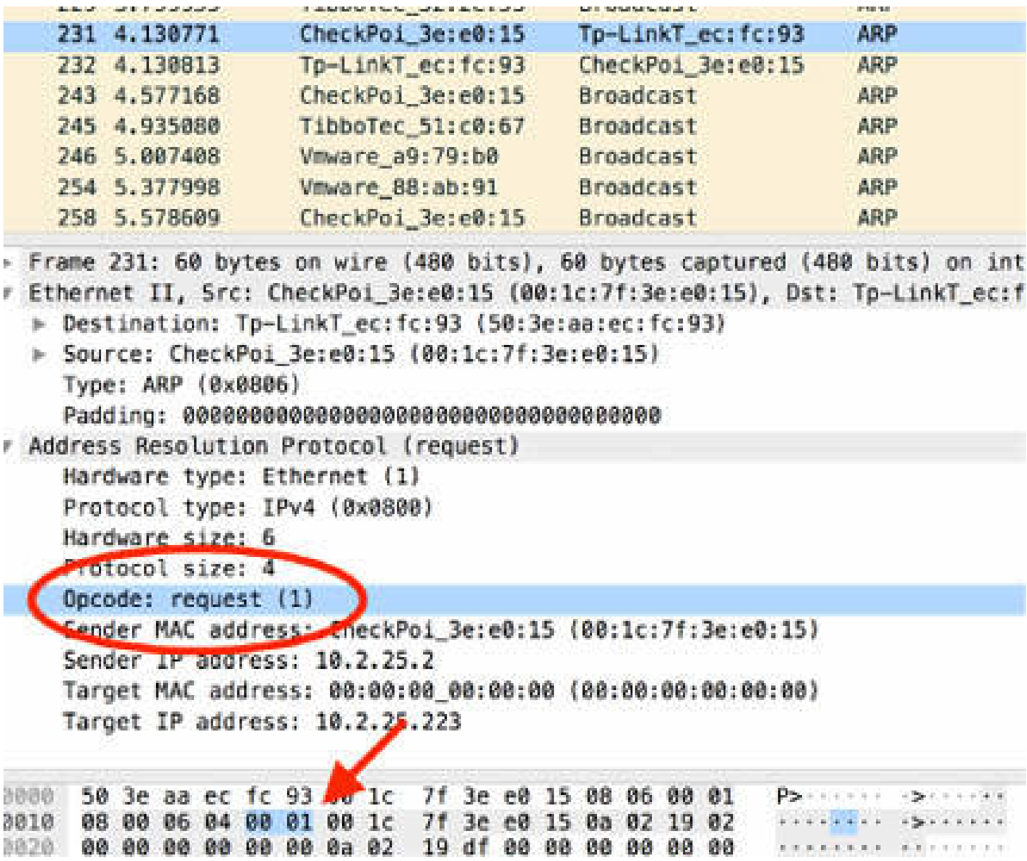
The “Hardware size” field defines the length (in bytes) of the hardware addresses used in the ARP packet under observation. The “Protocol size” field defines the length (in bytes) of the protocol (network) addresses used in the packet under observation.

Task 6:

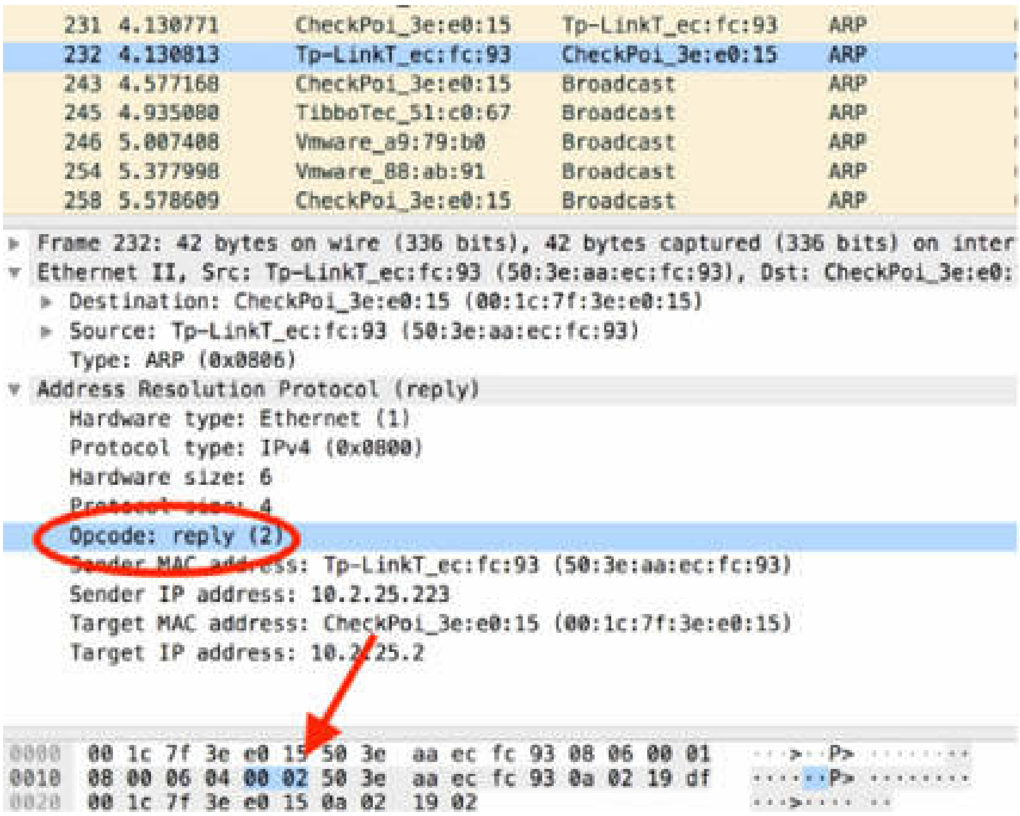
The Opcode field is the next field. It defines whether the ARP message is a request packet or a reply packet. It also defines the type of address resolution taking place. The following are possible values for the ARP and Reverse ARP (RARP) operation codes:

- Opcode 1: ARP request
- Opcode 2: ARP reply
- Opcode 3: RARP request
- Opcode 4: RARP reply

In the Packet List pane, select the packet #231, and in the Packet Details pane, verify that the Opcode is 1 (ARP request), as shown in the figure below.

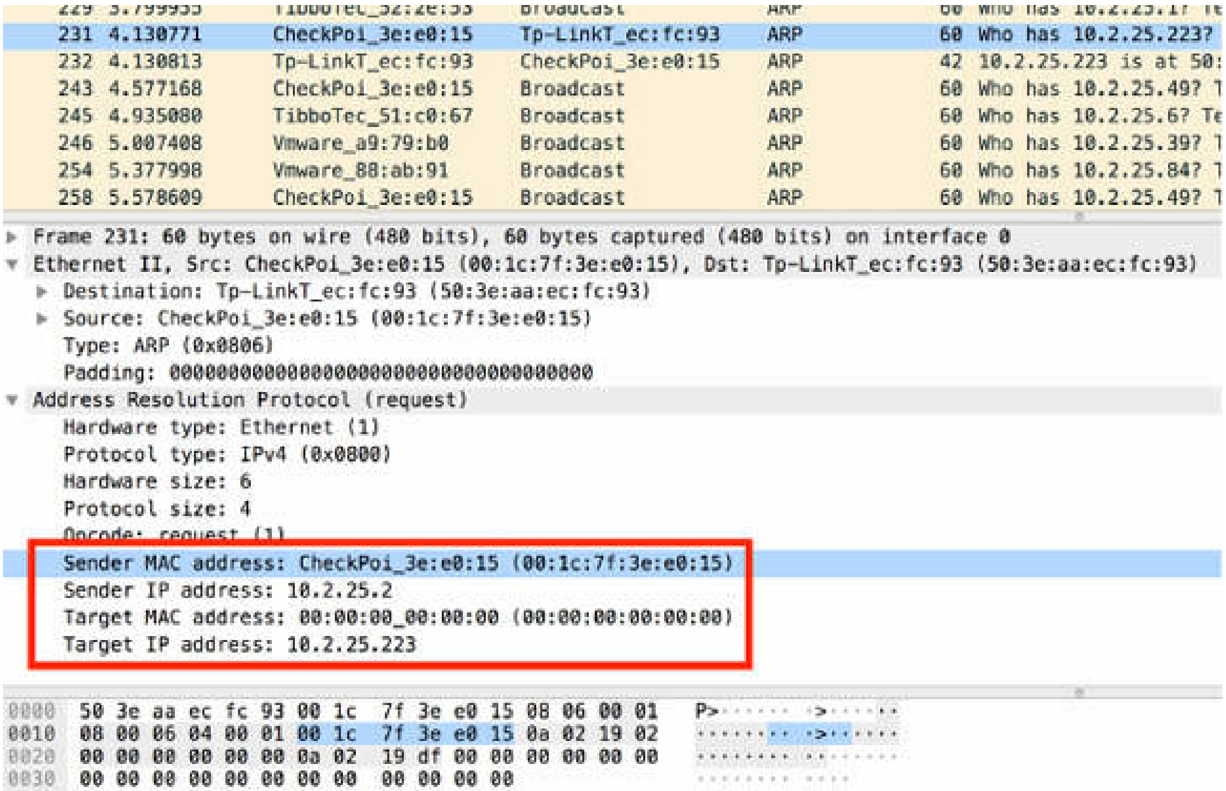


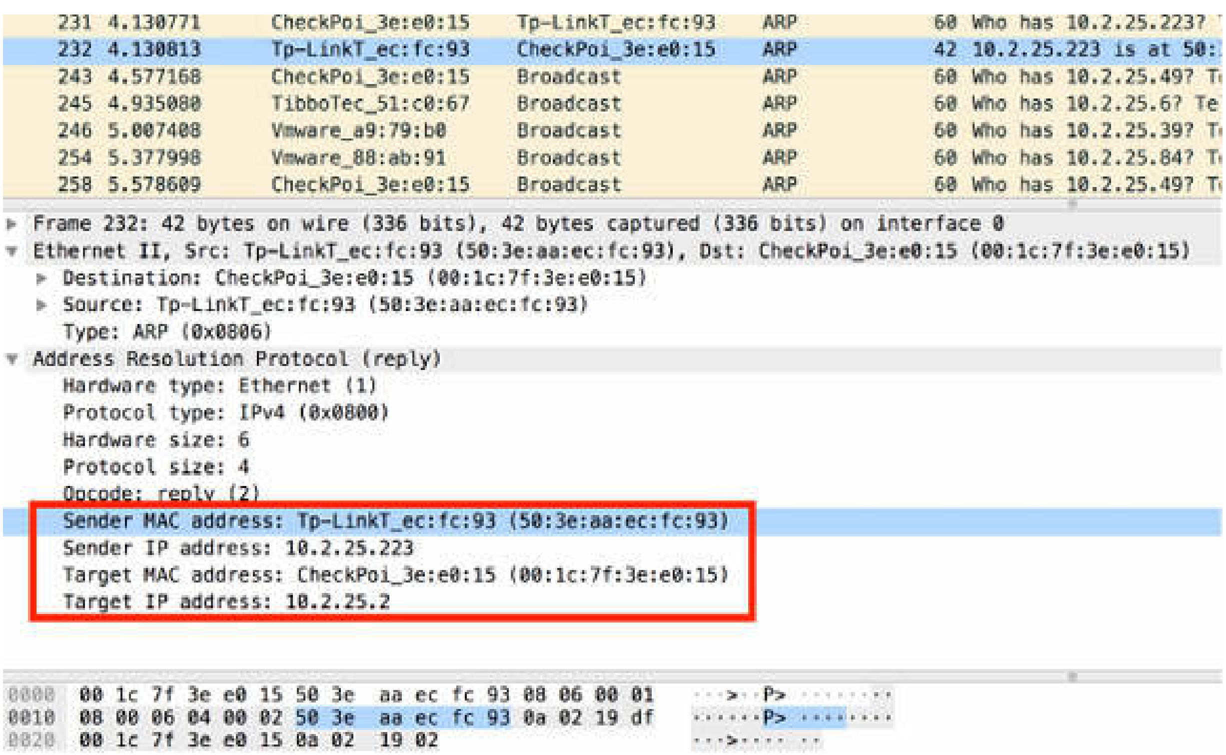
Select the packet #232, and verify that the Opcode is 2 (ARP reply), as shown in the figure below.



Task 7:

The Hardware and Protocol Addresses related to the sender and the target are the last four fields. The figures below show the last four fields of the ARP message for the request and the reply, respectively.


229 3.799935 TibboTec_52:2e:33 Broadcast ARP 60 Who has 10.2.25.1? Te
231 4.130771 CheckPoi_3e:e0:15 Tp-LinkT_ec:fc:93 ARP 60 Who has 10.2.25.223?
232 4.130813 Tp-LinkT_ec:fc:93 CheckPoi_3e:e0:15 ARP 42 10.2.25.223 is at 50:
243 4.577168 CheckPoi_3e:e0:15 Broadcast ARP 60 Who has 10.2.25.49? T
245 4.935080 TibboTec_51:c0:67 Broadcast ARP 60 Who has 10.2.25.6? Te
246 5.007408 Vmware_a9:79:b0 Broadcast ARP 60 Who has 10.2.25.39? T
254 5.377998 Vmware_88:ab:91 Broadcast ARP 60 Who has 10.2.25.84? T
258 5.578609 CheckPoi_3e:e0:15 Broadcast ARP 60 Who has 10.2.25.49? T
Frame 231: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
Ethernet II, Src: CheckPoi_3e:e0:15 (00:1c:7f:3e:e0:15), Dst: Tp-LinkT_ec:fc:93 (50:3e:aa:ec:fc:93)
Destination: Tp-LinkT_ec:fc:93 (50:3e:aa:ec:fc:93)
Source: CheckPoi_3e:e0:15 (00:1c:7f:3e:e0:15)
Type: ARP (0x0806)
Padding: 000000000000000000000000000000000000
Address Resolution Protocol (request)
Hardware type: Ethernet (1)
Protocol type: IPv4 (0x0800)
Hardware size: 6
Protocol size: 4
Opcode: request (1)
Sender MAC address: CheckPoi_3e:e0:15 (00:1c:7f:3e:e0:15)
Sender IP address: 10.2.25.2
Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
Target IP address: 10.2.25.223
0000 50 3e aa ec fc 93 00 1c 7f 3e e0 15 08 06 00 01 P>.......>......
0010 08 00 06 04 00 01 00 1c 7f 3e e0 15 0a 02 19 02 ....>......
0020 00 00 00 00 00 00 0a 02 19 df 00 00 00 00 00 00 ....
0030 00 00 00 00 00 00 00 00 00 00 00 00 ....


231 4.130771 CheckPoi_3e:e0:15 Tp-LinkT_ec:fc:93 ARP 60 Who has 10.2.25.223? T
232 4.130813 Tp-LinkT_ec:fc:93 CheckPoi_3e:e0:15 ARP 42 10.2.25.223 is at 50:
243 4.577168 CheckPoi_3e:e0:15 Broadcast ARP 60 Who has 10.2.25.49? T
245 4.935080 TibboTec_51:c0:67 Broadcast ARP 60 Who has 10.2.25.6? Te
246 5.007408 Vmware_a9:79:b0 Broadcast ARP 60 Who has 10.2.25.39? T
254 5.377998 Vmware_88:ab:91 Broadcast ARP 60 Who has 10.2.25.84? T
258 5.578609 CheckPoi_3e:e0:15 Broadcast ARP 60 Who has 10.2.25.49? T
Frame 232: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
Ethernet II, Src: Tp-LinkT_ec:fc:93 (50:3e:aa:ec:fc:93), Dst: CheckPoi_3e:e0:15 (00:1c:7f:3e:e0:15)
Destination: CheckPoi_3e:e0:15 (00:1c:7f:3e:e0:15)
Source: Tp-LinkT_ec:fc:93 (50:3e:aa:ec:fc:93)
Type: ARP (0x0806)
Address Resolution Protocol (reply)
Hardware type: Ethernet (1)
Protocol type: IPv4 (0x0800)
Hardware size: 6
Protocol size: 4
Opcode: reply (2)
Sender MAC address: Tp-LinkT_ec:fc:93 (50:3e:aa:ec:fc:93)
Sender IP address: 10.2.25.223
Target MAC address: CheckPoi_3e:e0:15 (00:1c:7f:3e:e0:15)
Target IP address: 10.2.25.2
0000 00 1c 7f 3e e0 15 50 3e aa ec fc 93 08 06 00 01 ...>..P>........
0010 08 00 06 04 00 02 50 3e aa ec fc 93 0a 02 19 df ......P>........
0020 00 1c 7f 3e e0 15 0a 02 19 02 ...>......

In the figures above:

- The Sender MAC address indicates the hardware address of the device that is sending this request or reply.
- The Sender IP address indicates the protocol (network) address of the device that is sending this request or reply.
- The Target MAC address indicates the desired target hardware address. If known, in ARP requests, this field is typically filled with all 0s. In ARP replies, this field contains the hardware address of the device that sent the ARP request.
- The Target Protocol address indicates the desired target protocol (network) address in a request. In an ARP reply, it contains the address of the device that issued the request.

Notes:
To gain more confidence in performing the ARP dissection for every field of an ARP message, repeat the previous steps for different types of ARP messages.

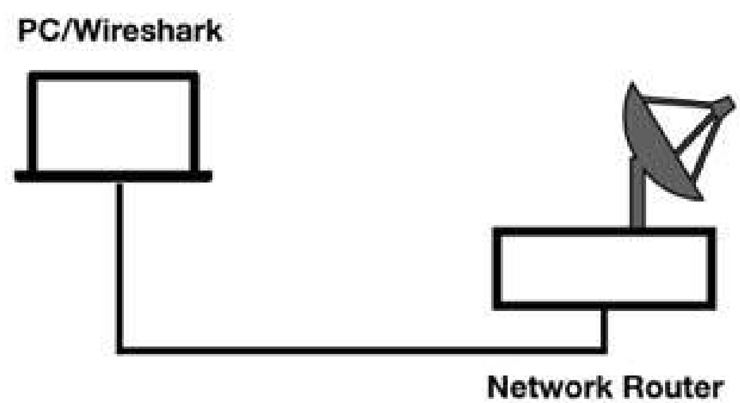
Lab 39. Internet Protocol

Lab Objective:
Learn how the Internet Protocol (IP) works.

Lab Purpose:
Learn the main purpose of the Internet Protocol and its main features.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
The main purpose of IP is to provide the datagram delivery services for networked systems and fragmentation and reassembly for low MTU

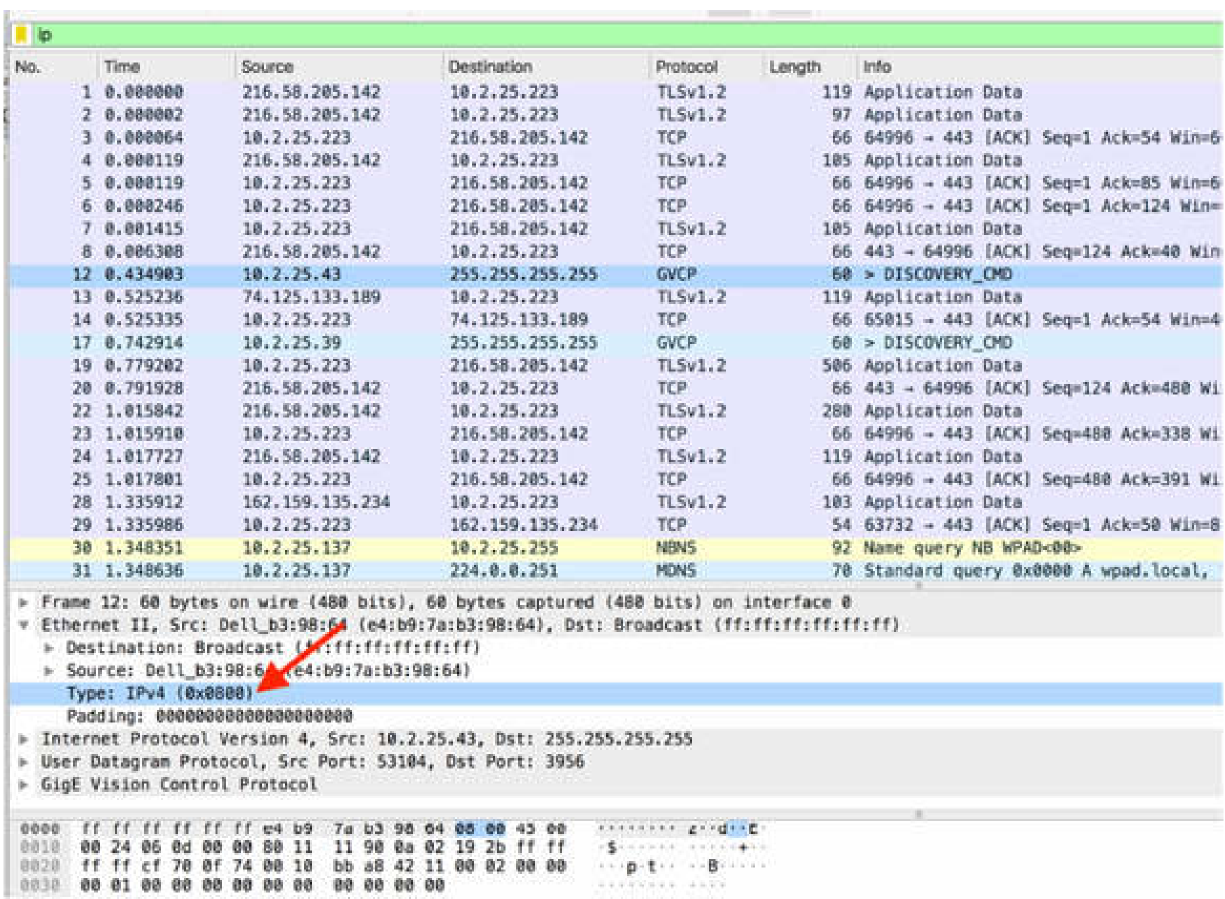
networks.

IP is connectionless and unreliable, providing the best-effort delivery of datagrams between the IP hosts, but there is no way to determine if a packet arrived at a target location. An application that needs guaranteed delivery should use TCP over IP.

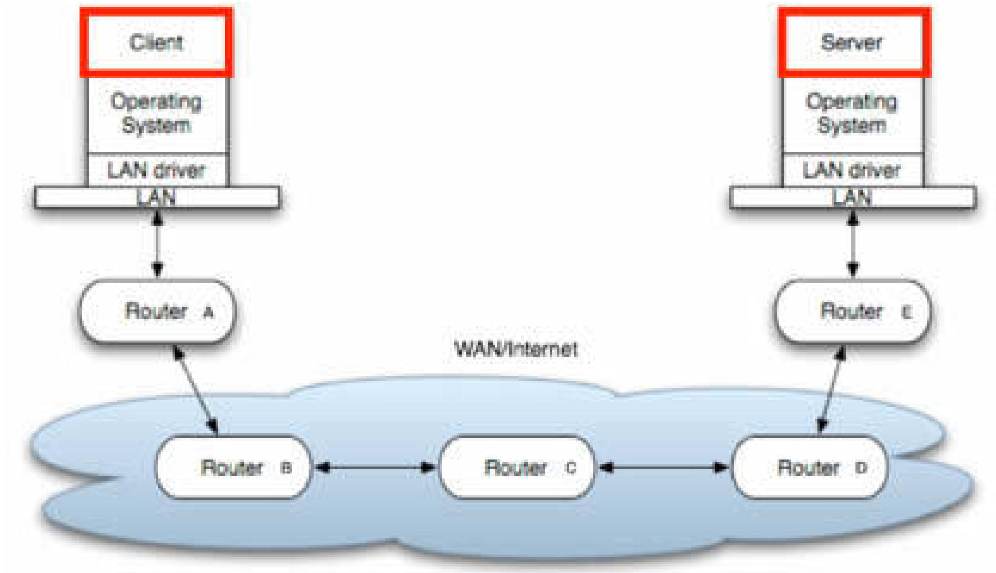
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

Stop the capture and save the file.

In the filter toolbar, enter ip to display only the IP (version 4) packets. The packet type is displayed in the Type field in the Packets Details pane, as shown in the figure below.

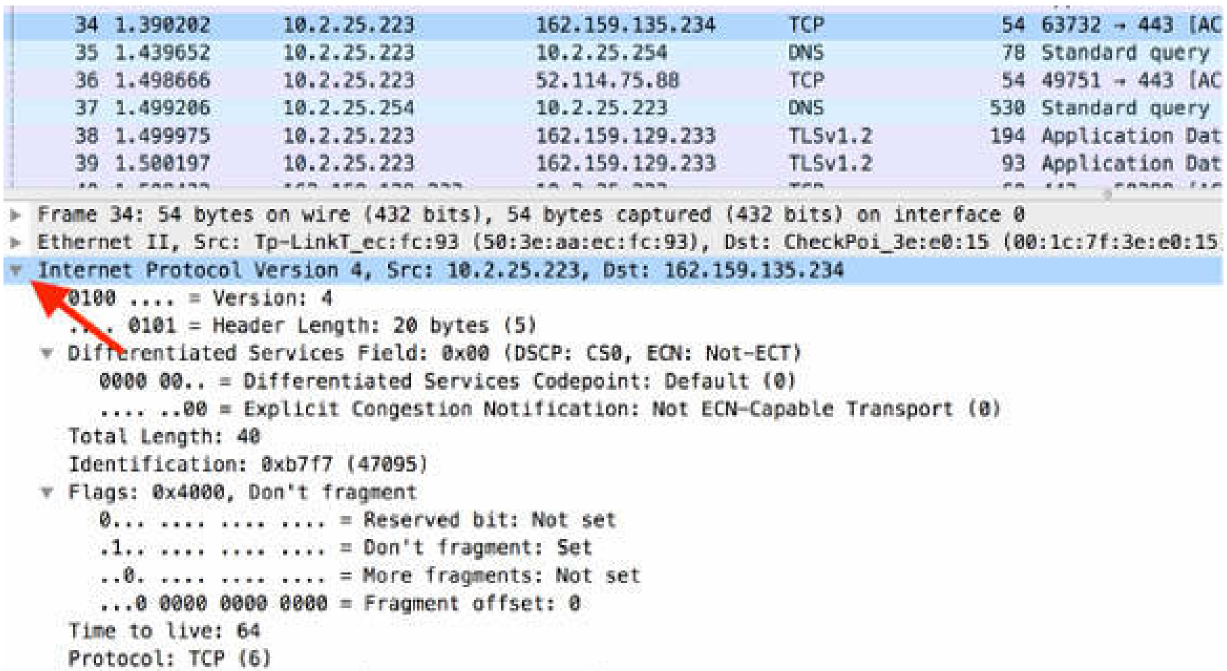


Task 2:
The figure below shows a scenario in which the client needs to communicate with the remote server and the communication path is composed of five routers.

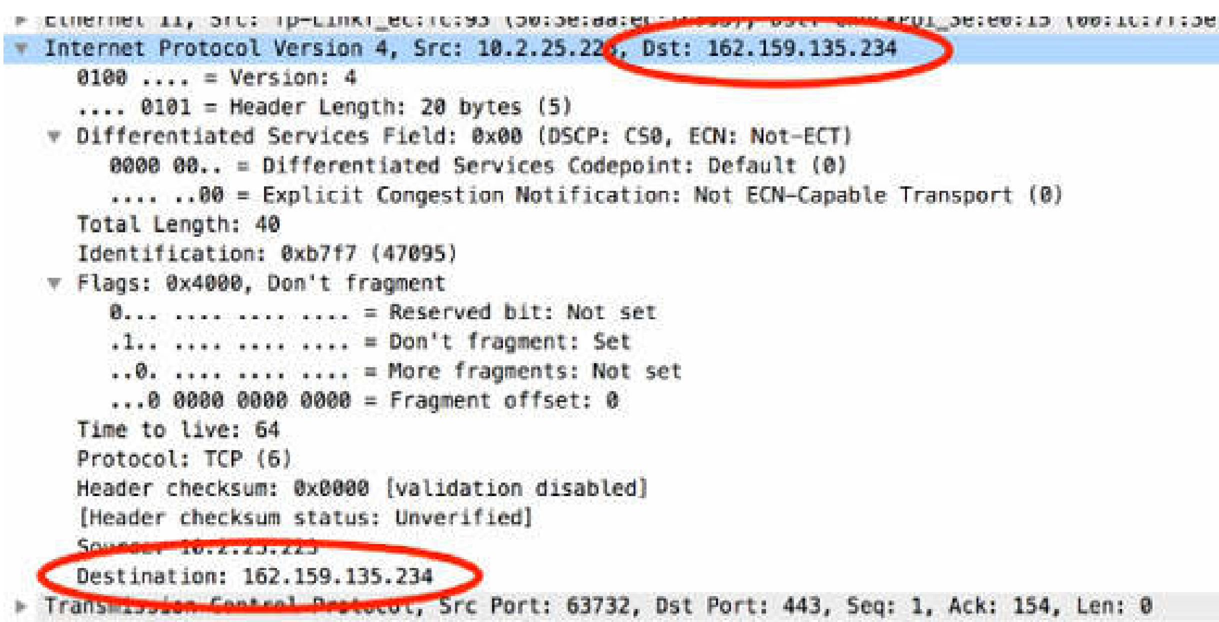


IPv4 gets packets from the client location to the server location by using the most efficient packet size.

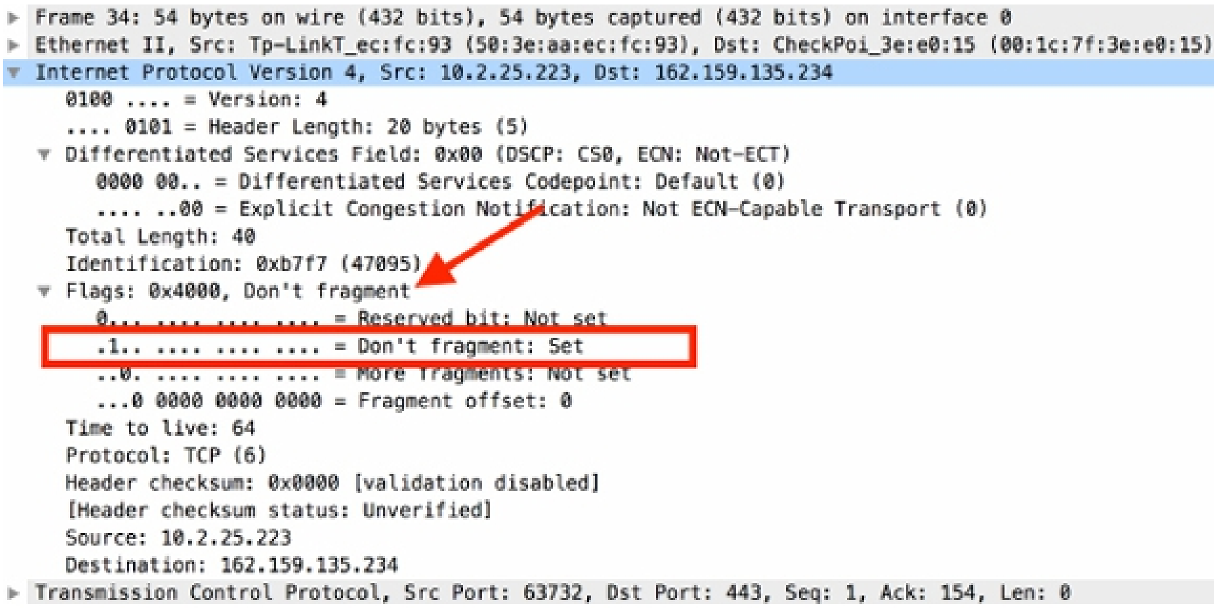
In the Packet List pane, select a TCP frame from the capture saved in Task 1. In the Packet Details pane, open the tree view for IPv4, as indicated by the arrow in the figure below.



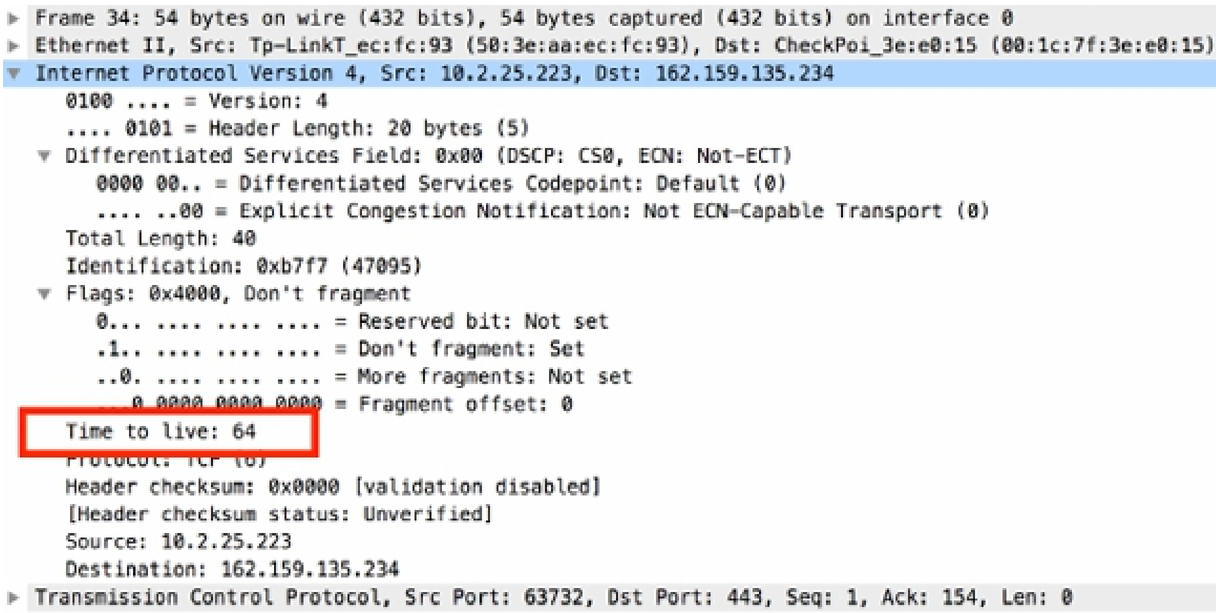
Considering that IPv4 packets are forwarded by routers, the target IP address is examined to make routing decisions. In the figure below, the target IP of the selected packet is identified.



To determine whether fragmentation is needed and allowed, the MTU size is checked against the MTU size of the next link (in this case, the next router). In the selected packet, the "Don't fragment" flag is set indicating that the fragmentation is not allowed.



The MAC header is stripped down to apply the new header related to the next network, and the "Time to Live" (TTL) value is decremented in the IP header. In this example, the TTL value is 64, and it is decremented to 63 when the packet is forwarded through the first router. At each forwarding, the TTL value is decremented by one.



If everything is working normally in IPv4 communication, the traffic should flow to and from IP address to IP address.

If a packet is too large to be forwarded to the next link in a path, the router examines the IP header's fragmentation setting. If the "Don't fragment" bit is set, the packet cannot be forwarded. If fragmentation is allowed, the router splits the single large packet into two (or more) smaller packets, defines the fragment offset, and indicates that the packets are fragments and forwards them.

Notes:
To gain more confidence with IPv4 communication packets on different data paths, repeat the previous steps for different types of IP messages.

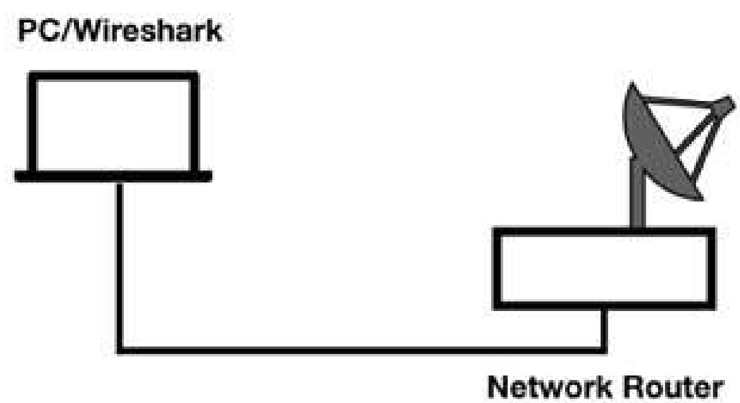
Lab 40. IP Packet Structure

Lab Objective:
Learn how the Internet Protocol (IP) works.

Lab Purpose:
Learn the main purpose of the Internet Protocol and its main features.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



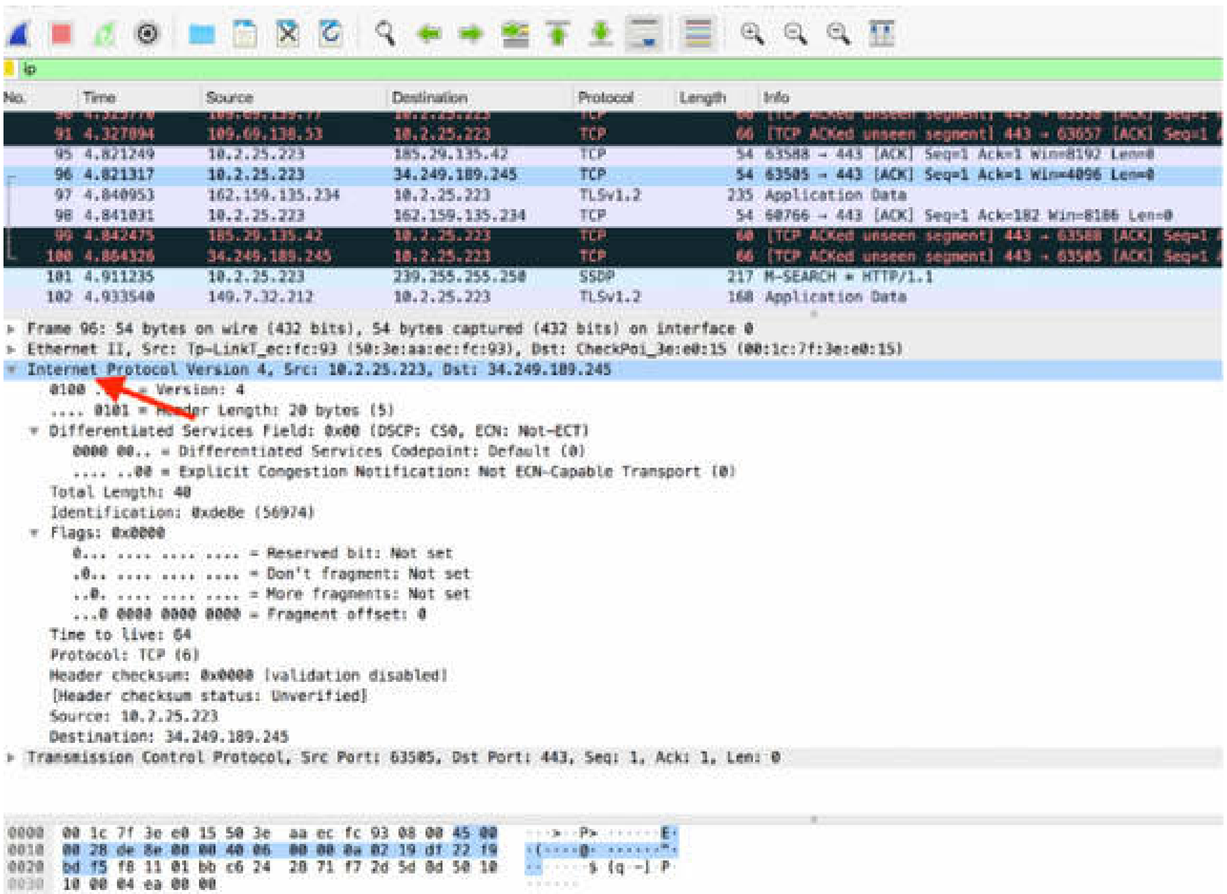
Lab Walkthrough:

Task 1:
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic

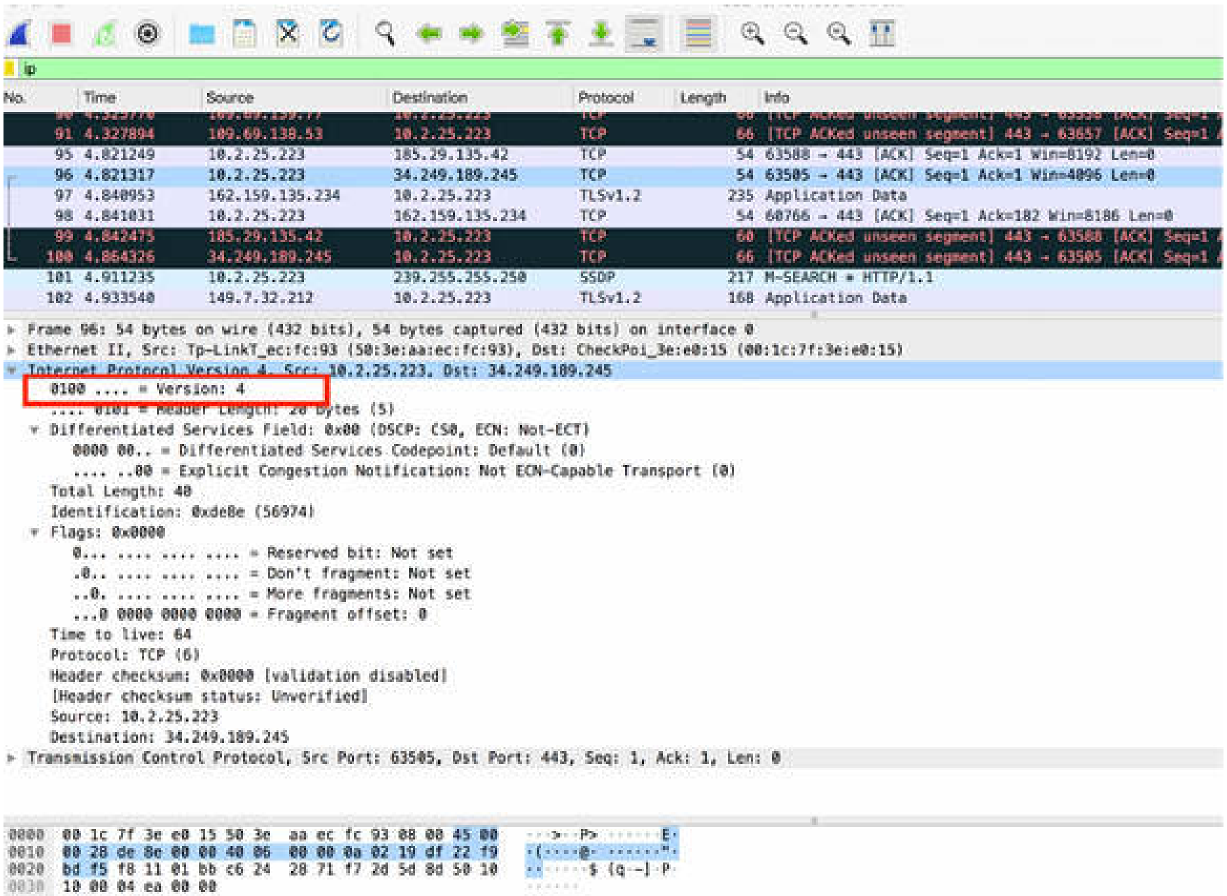
column. Capture the traffic for a few minutes.

Stop the capture and save the file.

In the filter toolbar, enter ip to display only the IP (version 4) packets. In the Packet Details pane, open the tree view for IPv4, as indicated by the arrow in the figure below.



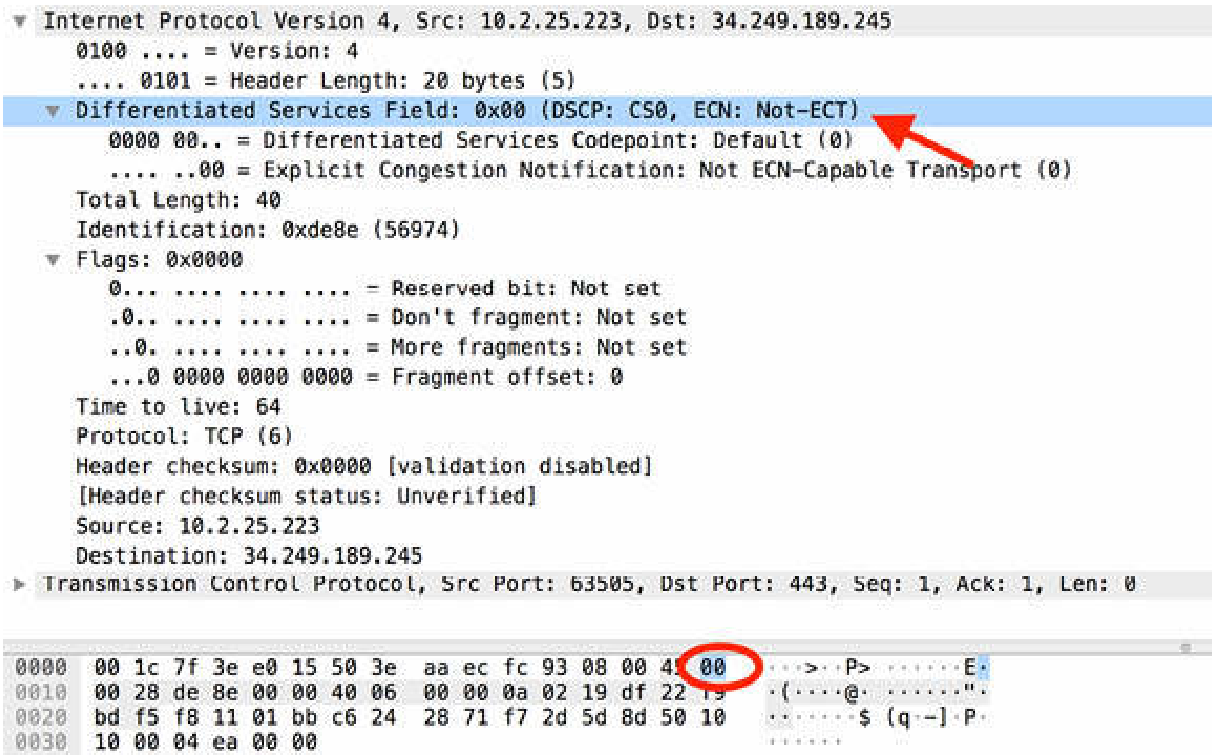
The Version field is the first field in the IP header. The value 4, shown in the figure below, indicates that the selected packet is an IPv4 packet.



The “Header Length” field denotes the length of only the IP header—just the IP header. This field is necessary because the IP header can support various options, and therefore, it may be of varying lengths. The value of this field is provided in the multiples of 4 bytes. For example, the actual decimal decode of this field is 5. Wireshark multiplies this value by 4 bytes to come up with the true IP header length value of 20 bytes, as shown in the figure below.

Task 2:

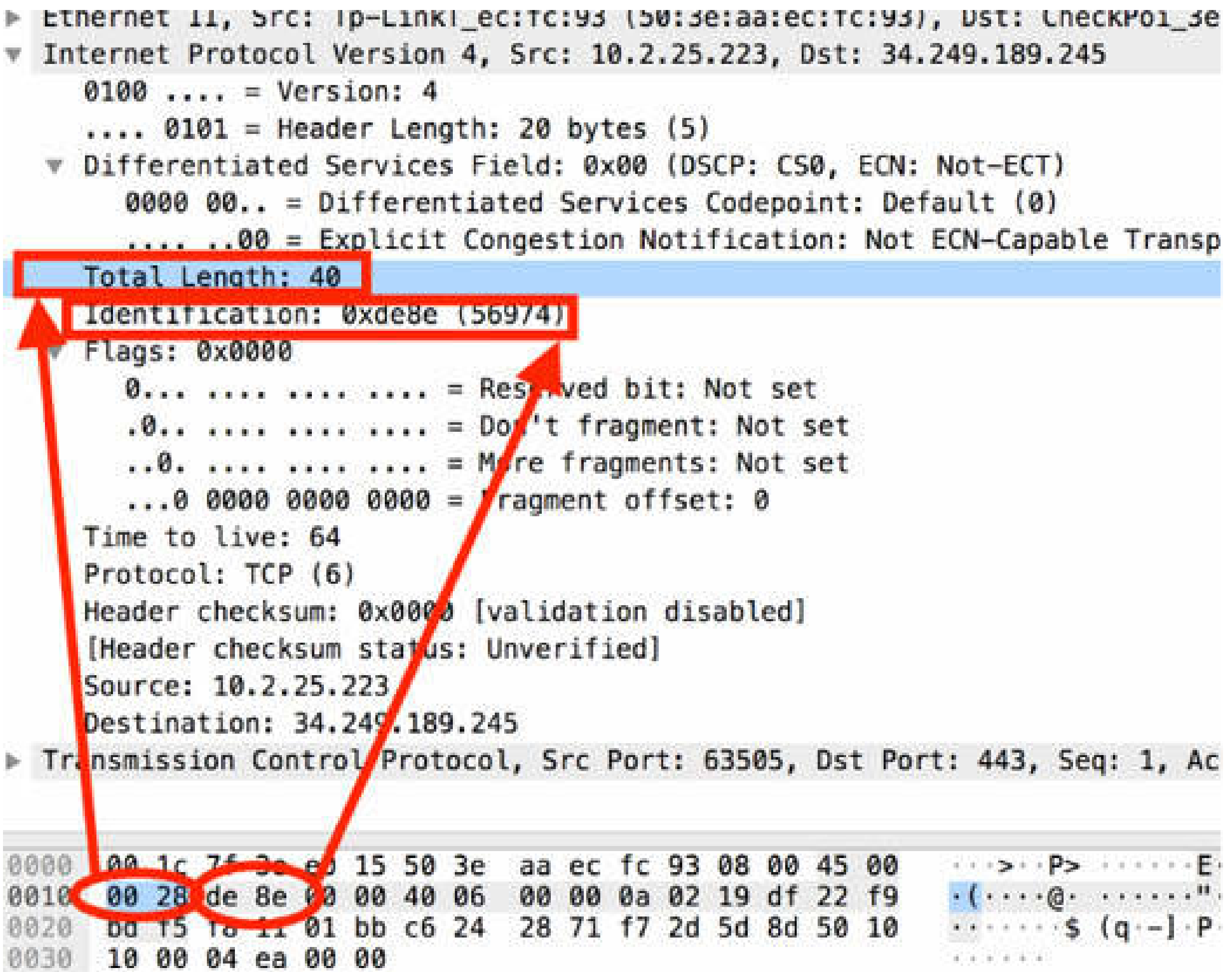
In the Packet Details pane, select “Differentiated Services Field” as indicated by the arrow in the figure below. The corresponding bytes are highlighted in the Packet Bytes pane.



"Differentiated Services Field" is composed of the following fields:

- Differentiated Services Code Point (DSCP) field: Six bits are used to prioritize traffic and provide a certain level of Quality of Service (QoS).
- Explicit Congestion Notification (ECN) field: The last two bits are used by the sender and/or routers along the path for identifying the network congestion along the route.

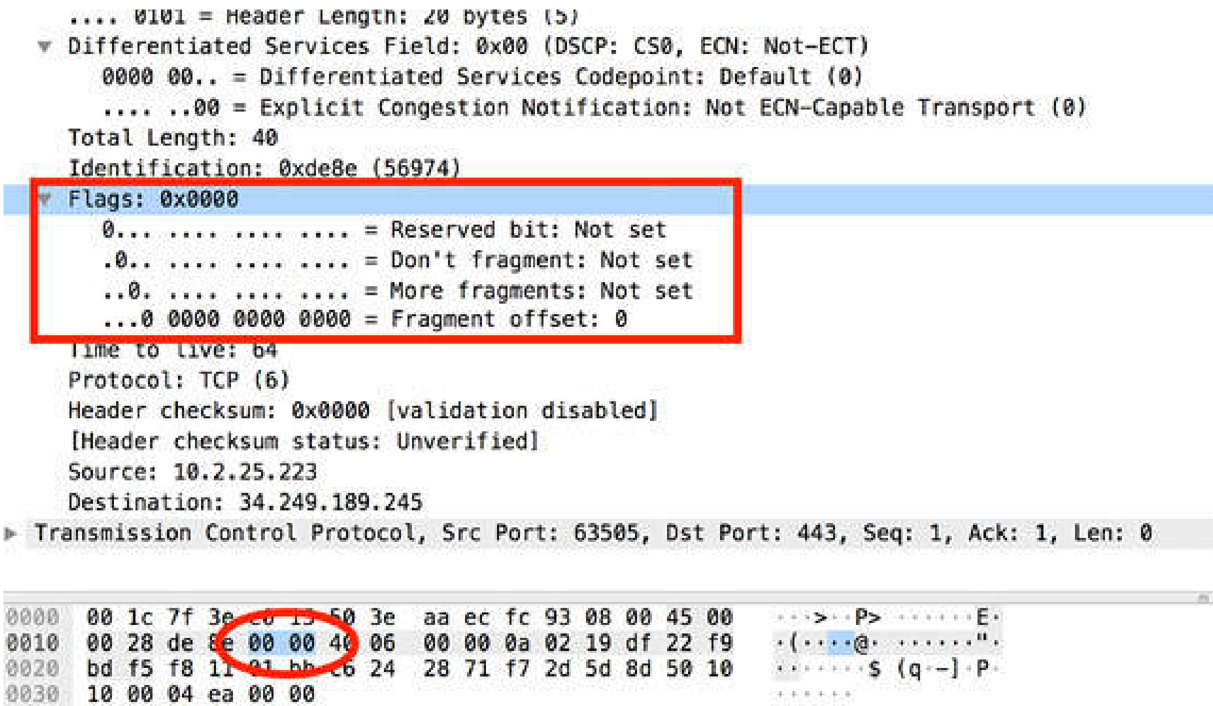
The "Total Length" field is the next field of four bytes. It defines the length of the IP header and any valid data. In the selected packet, the length is 40 bytes, as shown in the figure below.



The Identification field, shown in the figure above containing value 0xde8e , represents the unique ID value given to an IP packet when it is sent. If a packet must be fragmented (considering network size limitations), the same ID number is placed in each fragment to indicate that these fragments are a part of the same original packet.

Task 3:

In the Packet Details pane, click the Flags field to open the tree view, as shown in the figure below.

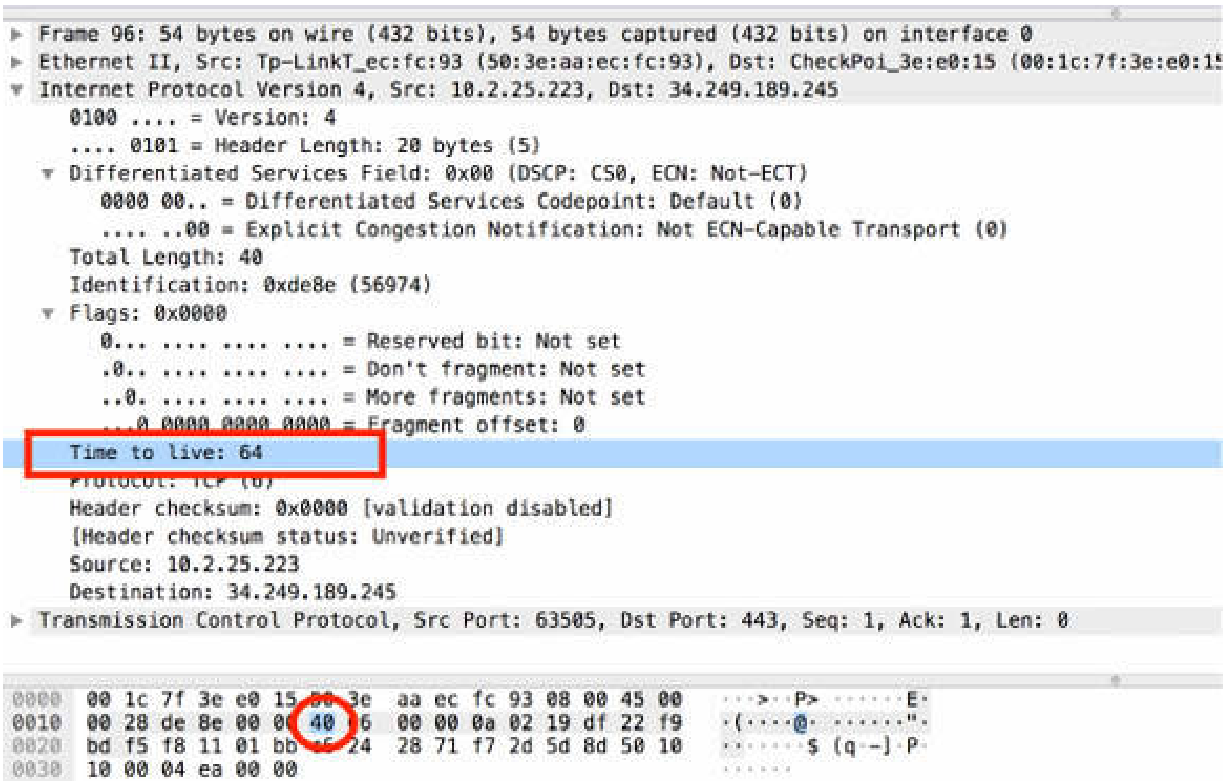


The first three bits of the Flags field are used to manage fragmentation. These bits have the following meaning:

- Bit 0: Reserved—set to 0
- Bit 1: The Don't Fragment Bit (0 = may fragment; 1 = don't fragment)
- Bit 2: The More Fragments Bit (0 = last fragment; 1 = more to come)

The "Fragment Offset" field consists of the last 13 bits. When a packet is a fragmented packet, these bits are used to indicate where to place this packet's data while reassembling the fragments into a single packet at the destination host.

Task 4:
In the Packet Details pane, select the "Time to live"(TTL) field, as shown in the figure below.



The TTL field indicates the remaining lifetime, in seconds and hops through routers, of the packet under analysis. Typical TTL values are 32, 60, 64, and 128.

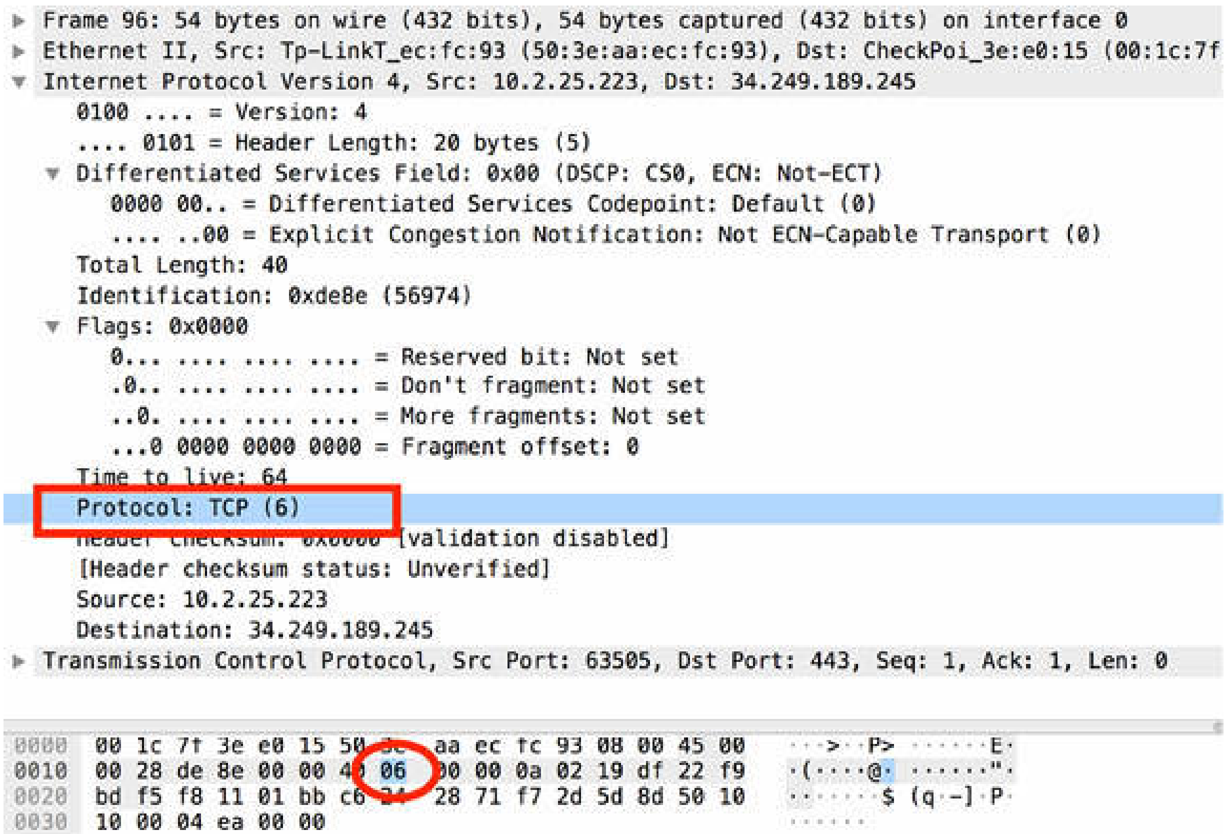
Each time a router forwards a packet, the router must decrement the TTL value by 1. If the router must hold the packet in its queue for an extended period (longer than one second), it must decrement the following:

- TTL value by the number of seconds the packet was held in the queue
- TTL for the hop

If a packet to be routed arrives at a router with TTL = 1, the router must discard the packet because it cannot decrement the TTL to 0 and forward the packet. If a packet with TTL = 1 arrives at a host, the packet is processed because the hosts do not need to decrement the TTL value.

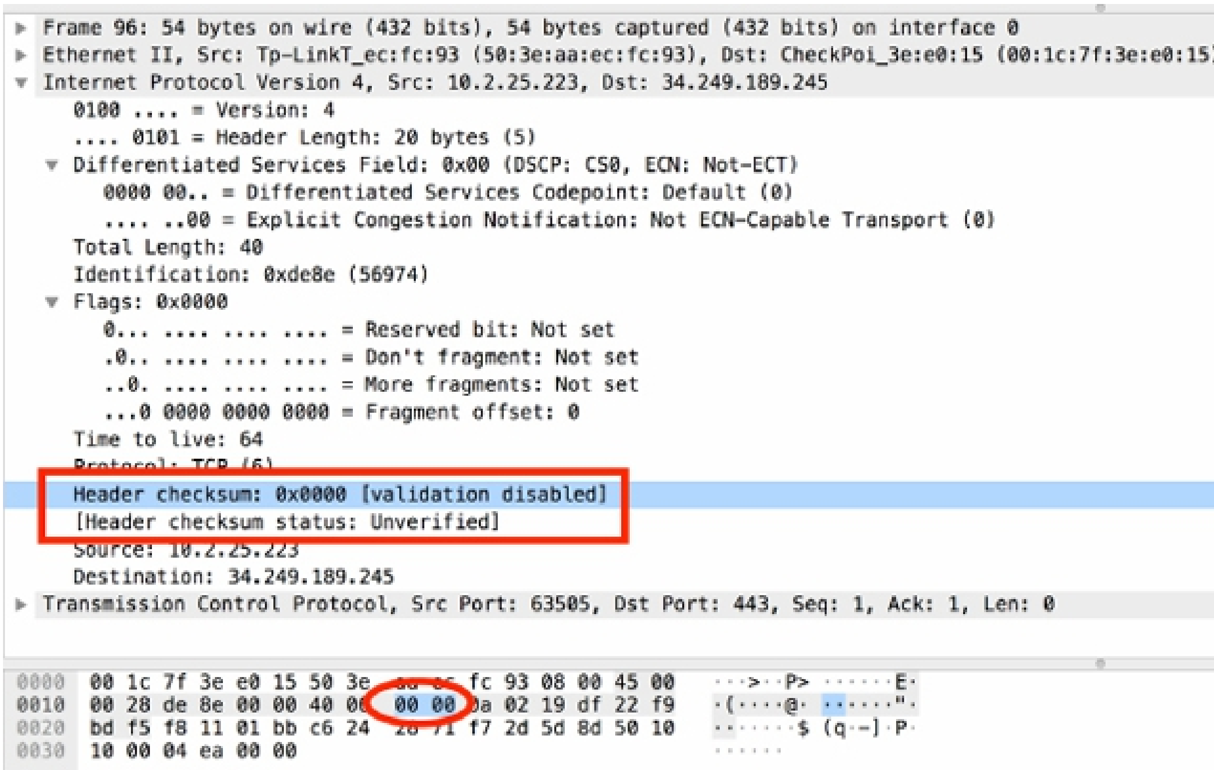
When a packet gets fragmented, all fragments are given the same TTL value.

Task 5:
In the Packet Details pane, select the Protocol field, as shown in the figure below.



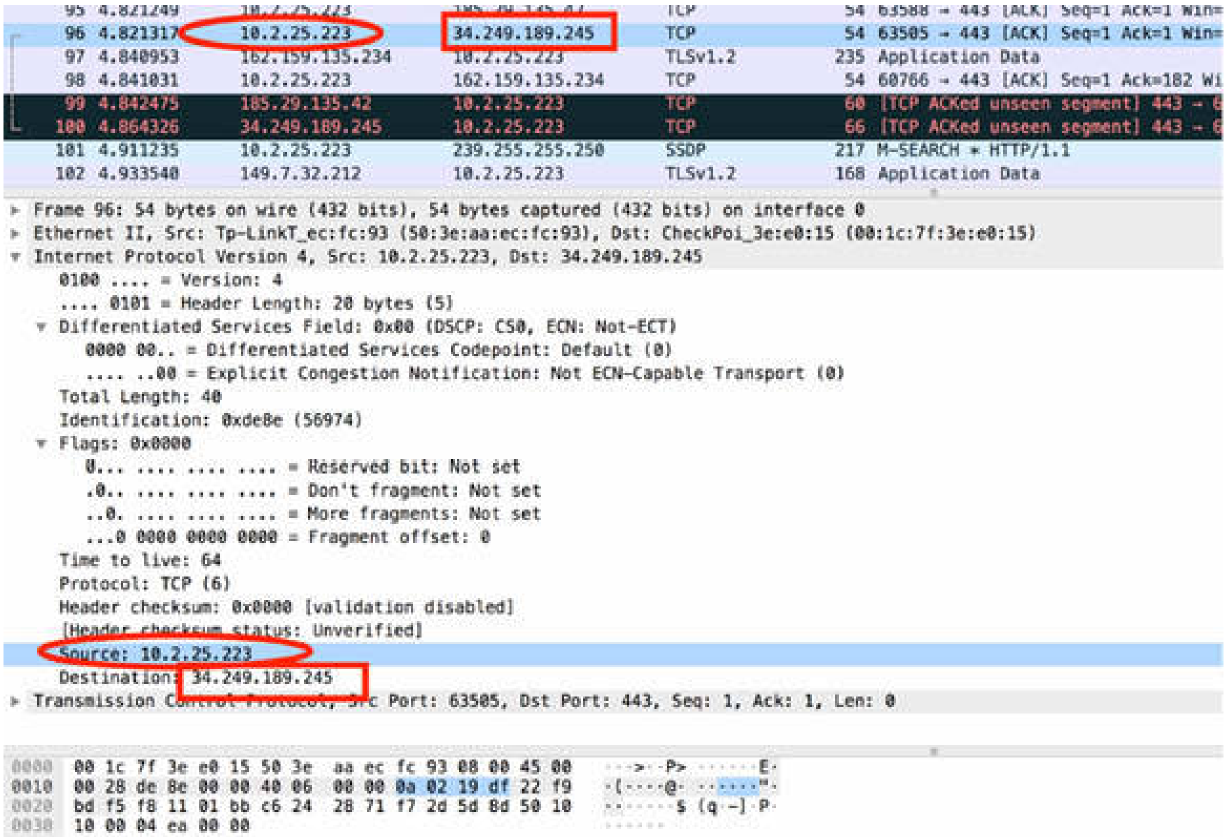
The Protocol field is present in all TCP headers and defines the type of packet that is coming up next. In this example, the value is equal to 6 (TCP protocol). Some of the possible values are: 1 (ICMP), 2 (IGMP), 4 (IPv4), and 8 (EGP).

The “Header checksum” field is the next field.



The “Header checksum” field provides error detection only on the contents of the IP header. It does not include the checksum field in the calculation. In this example, the validation is disabled, and the checksum is not verified for the packet.

The IPv4 Source and Destination fields are the next fields. These fields correspond to the IP address of the device that sent the packet and the final destination of the packet. These values are also displayed in the Packet List pane, as shown in the figure below.



The destination can be a unicast, multicast, or broadcast address.

Notes:
The Identification (ID) field is an important field for network analysis. When analyzing a network that is suspected to be flooded, if the ID field is the same and the packet is not a fragment (all fragments of a set contain the same IP ID value), you can assume that the same packet is looping the network.

IP header can be extended by using several options (although these options are not often used).

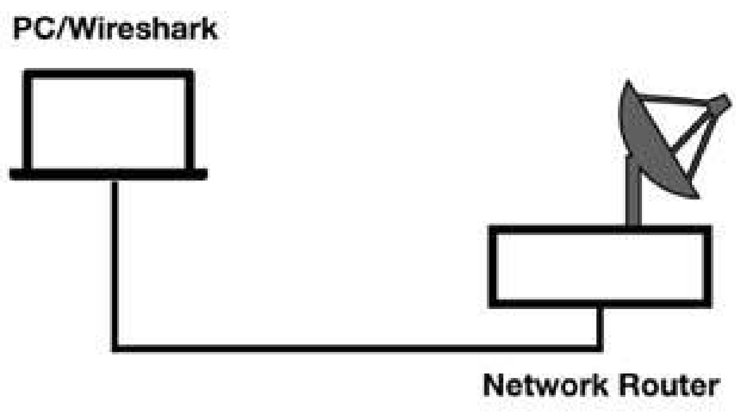
Lab 41. IP Filtering

Lab Objective:
Learn how to filter (capture/display) on the Internet Protocol (IP).

Lab Purpose:
Learn the features of capture filtering and display filtering for the Internet Protocol.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

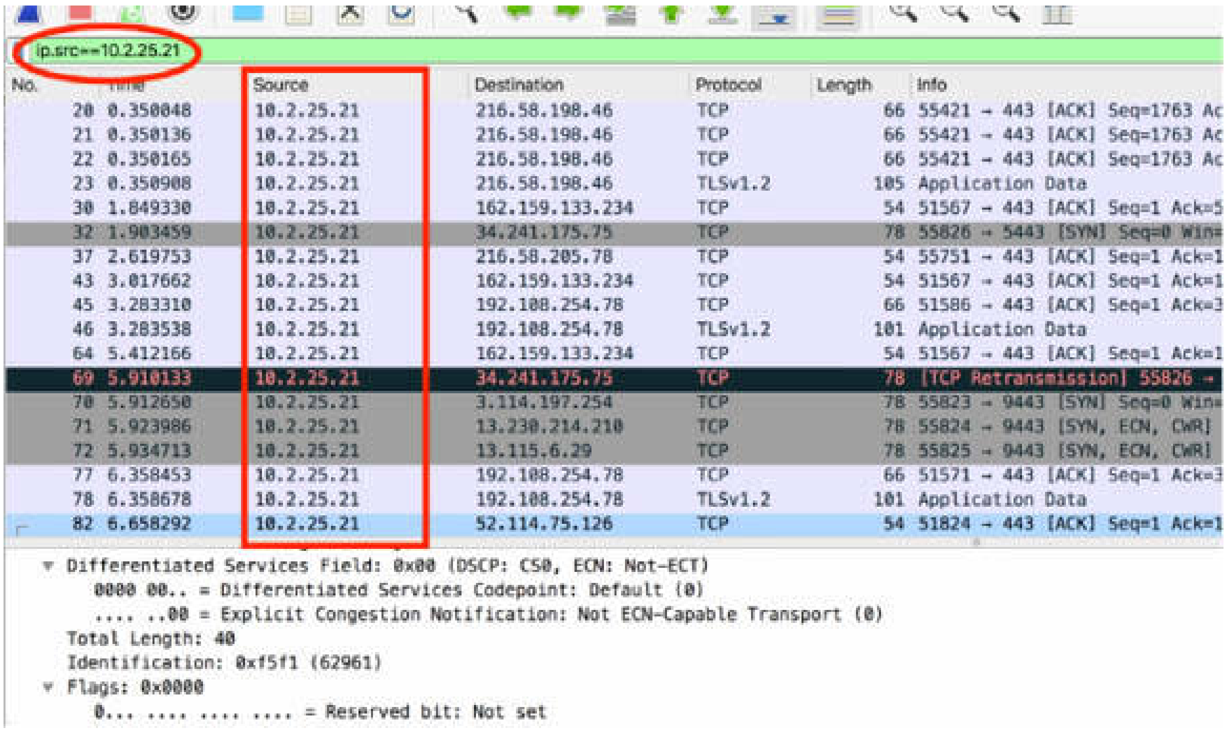
Task 1:

Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column.

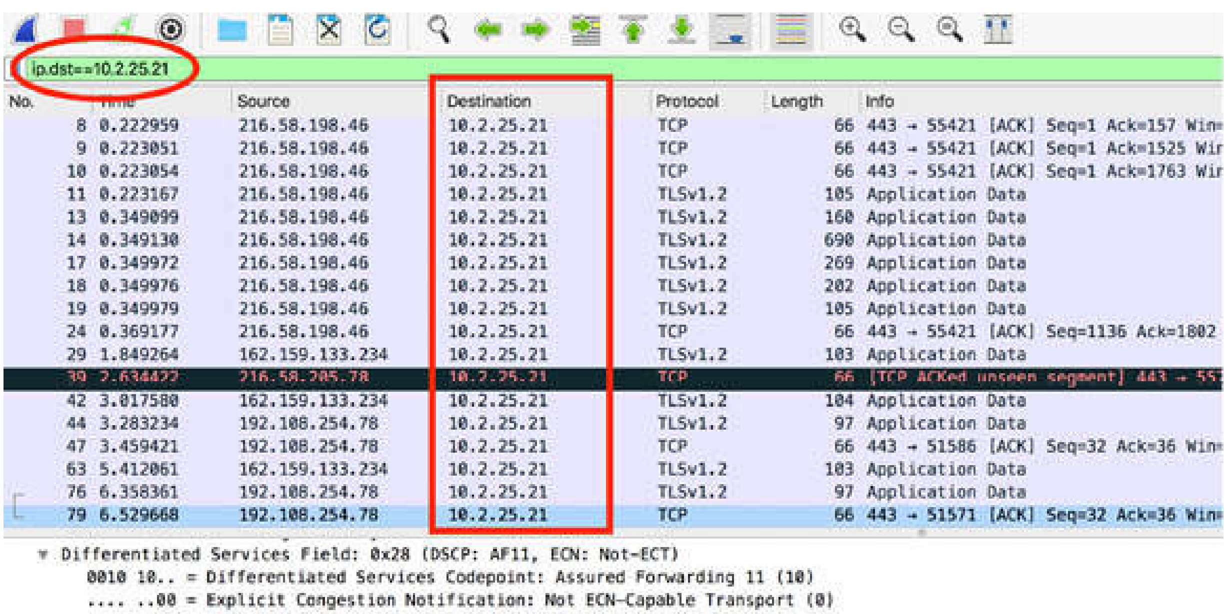
In the Capture Filter, enter ip to capture only IPv4 protocol packets.

Capture the traffic for a few minutes. Stop the capture and save the file, which will be composed of only IP packets.

To display only the IP packets transmitted from a specific IPv4 address, in the filter toolbar, enter ip.src == x.x.x.x , choosing a source IPv4 address from the ones present in the Packet List pane. In the figure below, the filter ip.src==10.2.25.21 is applied.

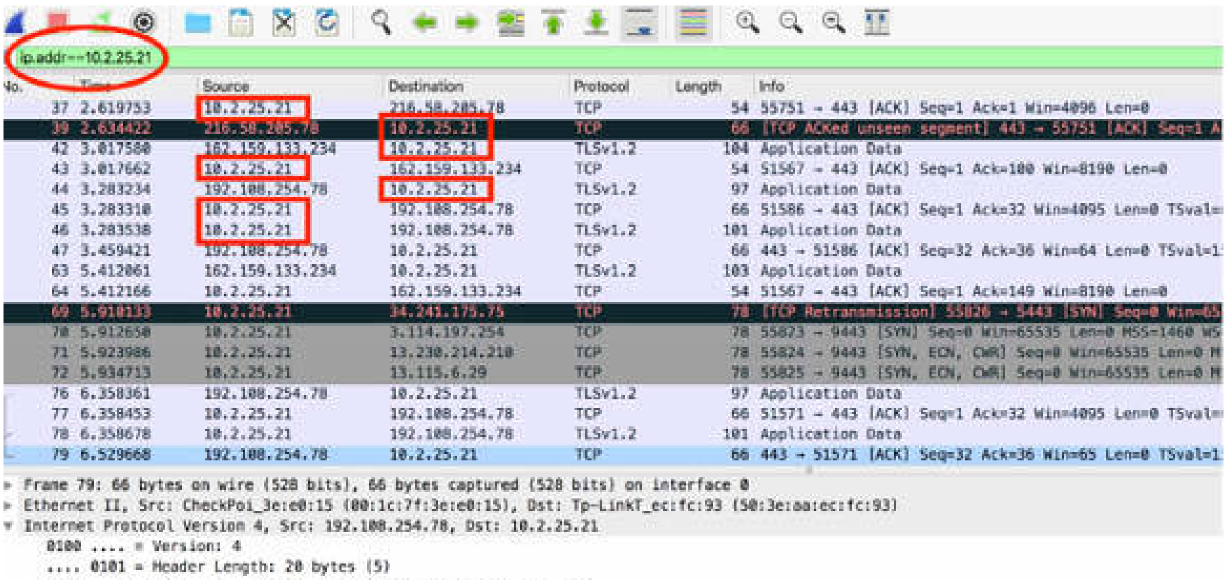


To display only the packets directed to a predefined destination, in the filter toolbar, enter ip.dst == y.y.y.y . Use a destination IPv4 address of interest. In the figure below, the ip.dst==10.2.25.21 filter is applied.

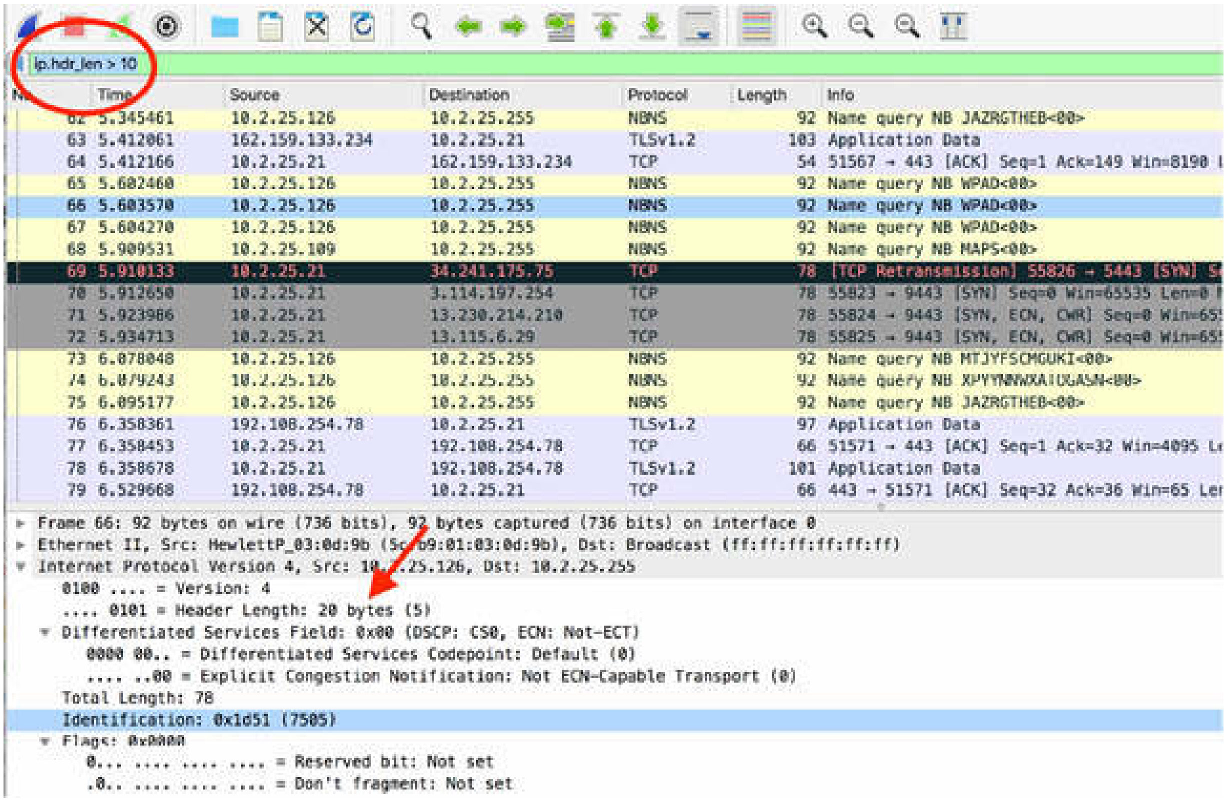


To display packets coming from and directed to a predefined IPv4 address, in the filter toolbar, enter ip.addr == k.k.k.k . Use an IPv4 address of interest. In the figure below, the ip.addr == 10.2.25.21 filter is applied.

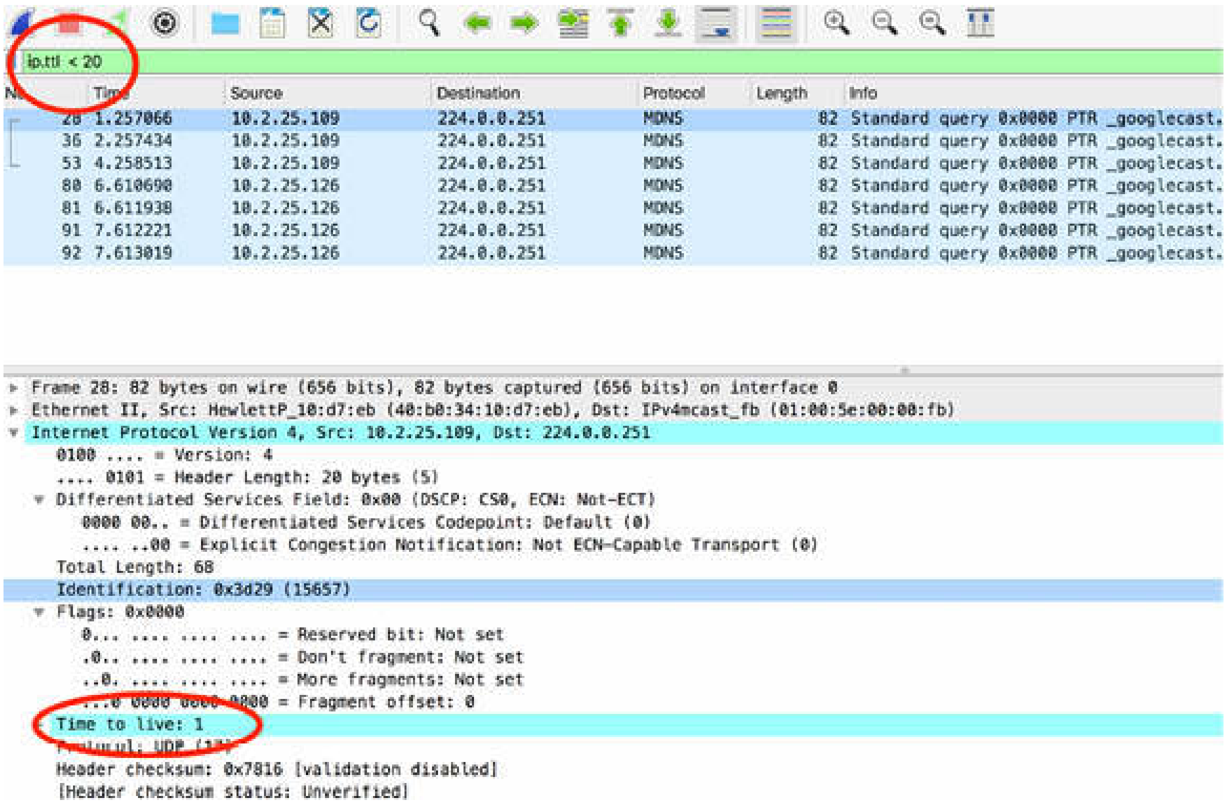
As shown in the figure below, for each packet in the Packet List pane, either the Source field or the Destination field is equal to the address specified.



Task 2:
You can also create a filter on the IP header length. In the filter toolbar, enter ip.hdr_len > 10 to select those packets that have the IP header dimension greater than ten bytes, as shown in the figure below.



To filter packets based on the value of the “Time to Live” field, in the filter toolbar, enter ip.ttl < 20 .

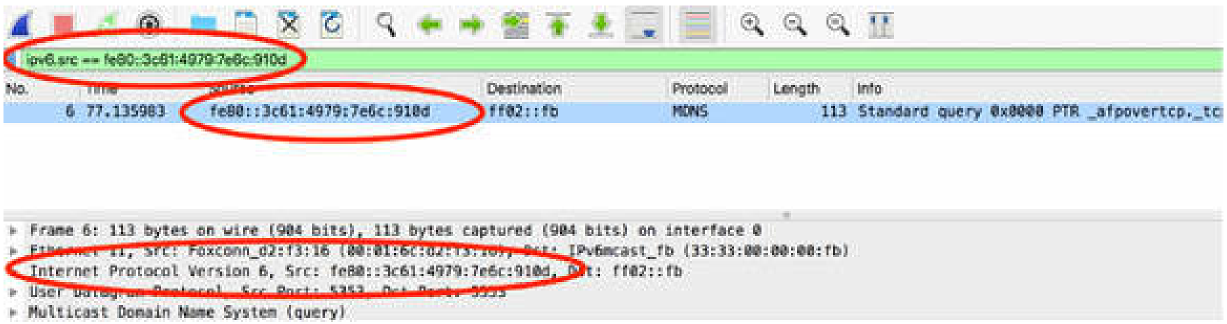


Task 3:

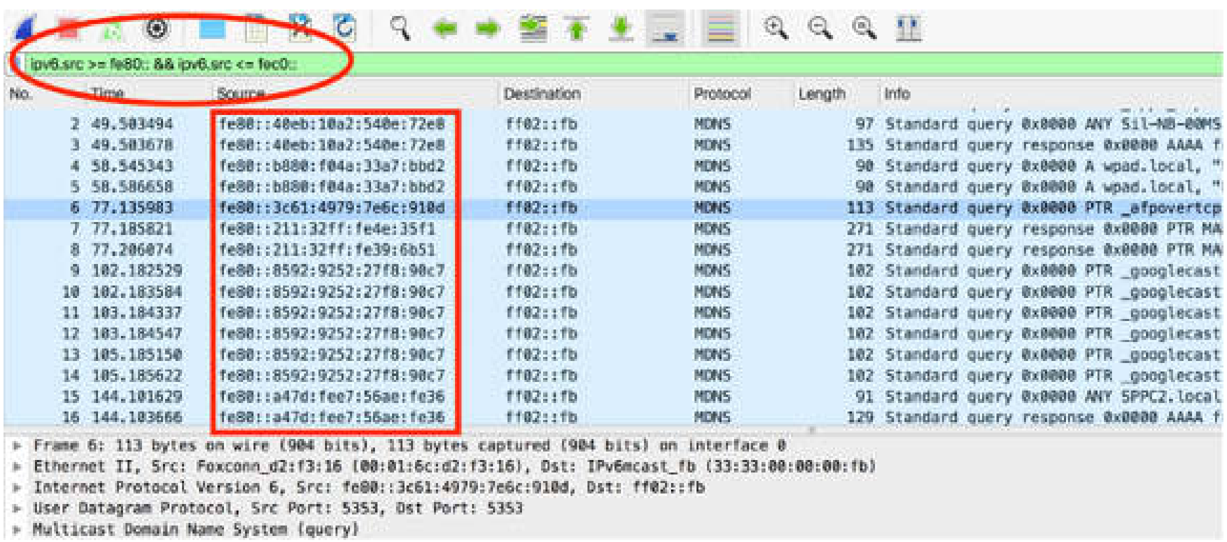
On the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column.

In the Capture Filter, enter ip6 to capture only IPv6 protocol packets. Note that the display filter requires you to enter ipv6 if you want to filter packets.

To display only the IP packets transmitted from a predefined IPv6 source address, in the filter toolbar, enter ipv6.src == fe80::3c61:4979:7e6c:910d . The Packet List pane displays packets from the specified source, as shown in the figure below.



To display only IPv6 packets from a range of source networks, in the filter toolbar, enter ipv6.src >= fe80:: && ipv6.src <= fec0:: and the filtered packets are displayed, as shown in the figure below.



Notes:

The most efficient way to find a useful display filter is by start filling in the filter toolbar. The auto-completion function helps you in viewing all available display filters.

Note that the capture filter requires ip6 but the display filter requires ipv6 — **https://wiki.wireshark.org/IPv6** .

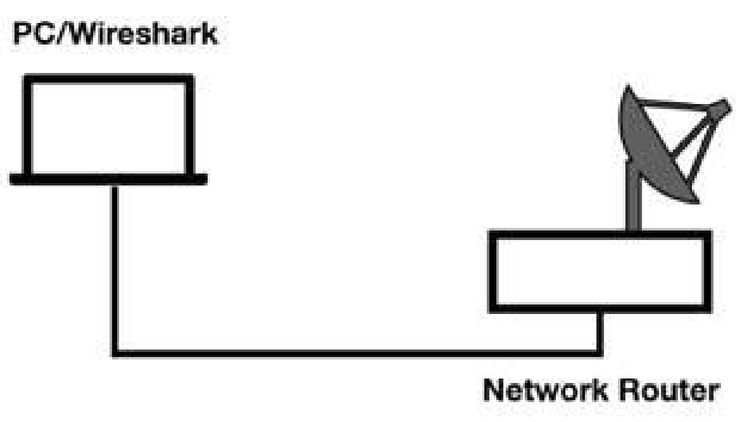
Lab 42. Internet Control Message Protocol (ICMP)

Lab Objective:
Learn how the Internet Control Message Protocol (ICMP) works and why is it used.

Lab Purpose:
Learn the main purpose of ICMP and the features of the protocol.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

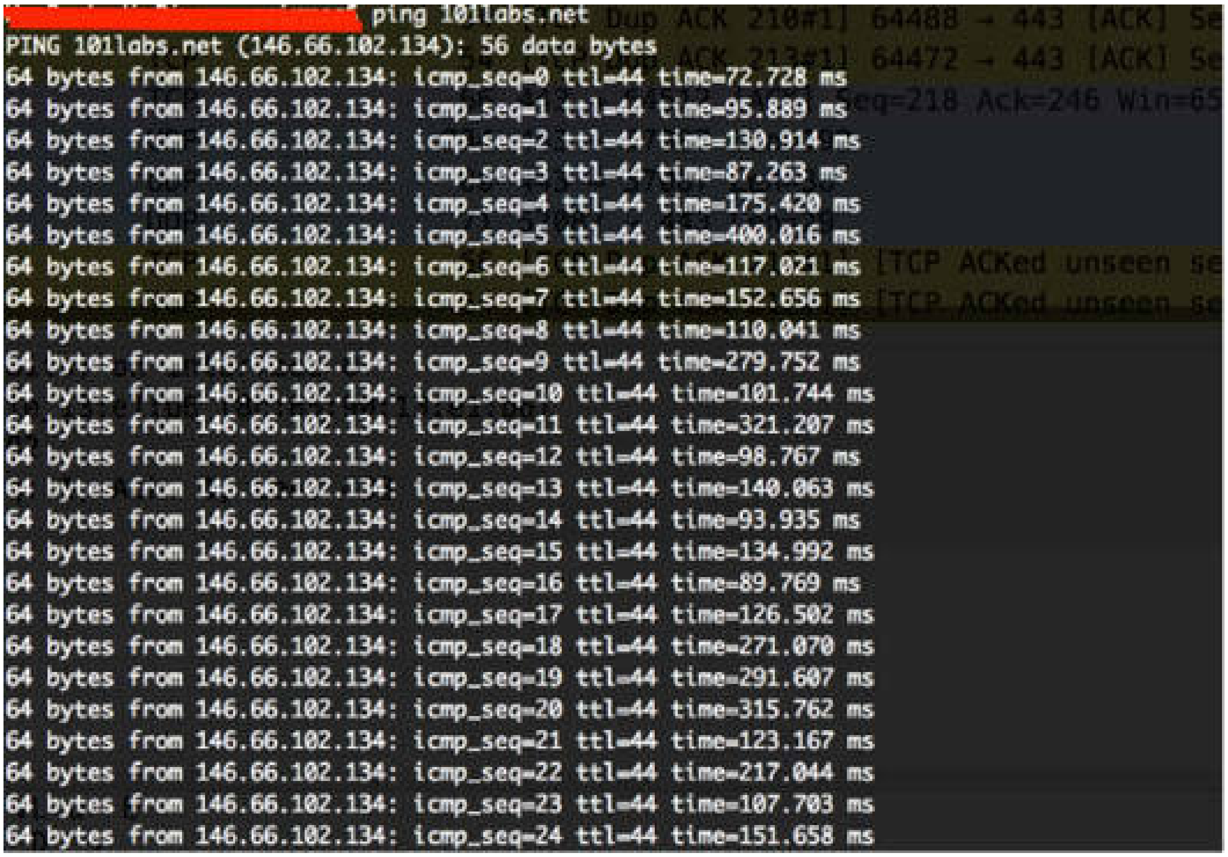
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

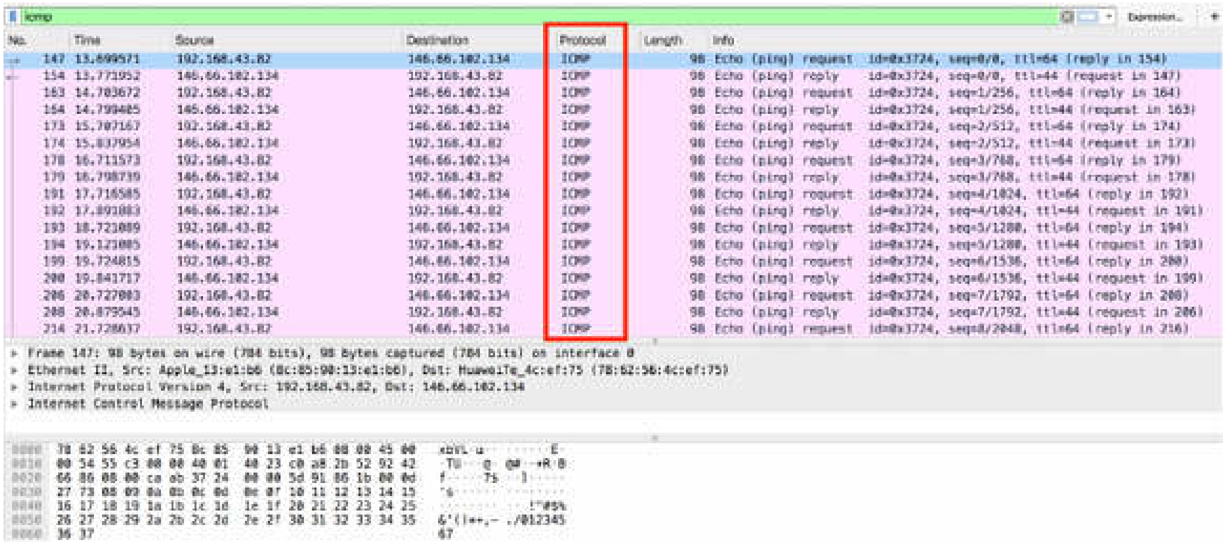
Task 1:
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column.

Open a terminal window, and run the command `ping 101labs.net` .



Capture the traffic for a few minutes. Stop the capture in Wireshark and save the file.

In the filter toolbar, enter `icmp` to display only ICMP packets.

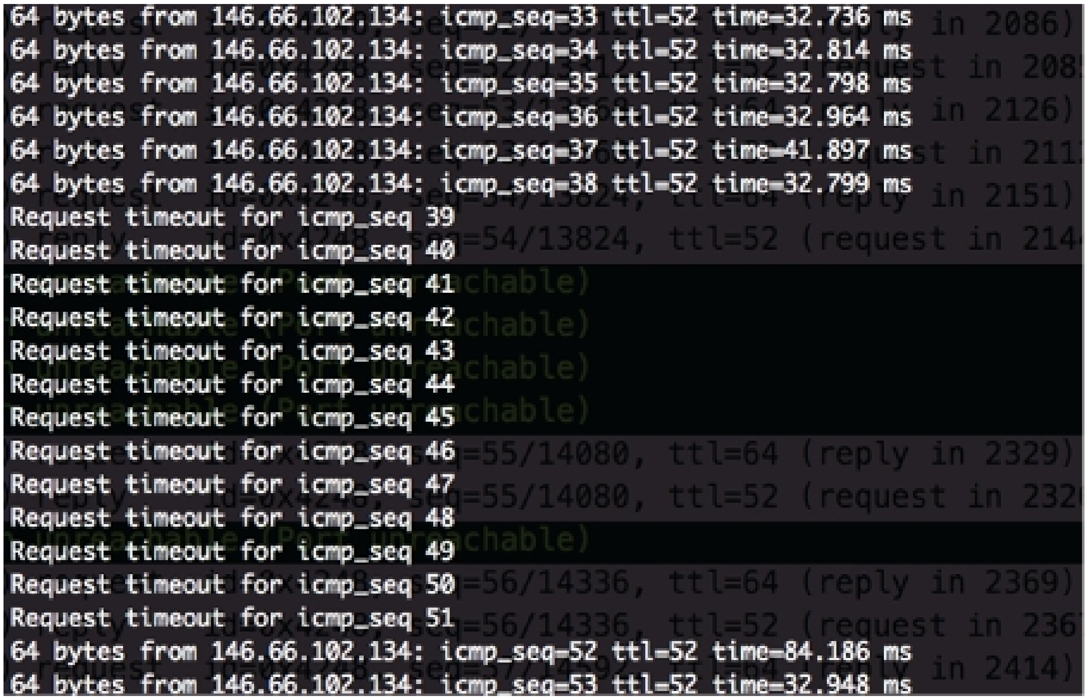


The ICMP protocol is used as a messaging system for errors, alerts, and general notifications on an IP network.

In this case, you used the ping request issued in the command shell using the Echo messages, as shown in the Info column in the figure above. In case of good connectivity, for each Echo (ping) request, an Echo (ping) reply follows, that is, the related request/reply is marked for each ICMP message.

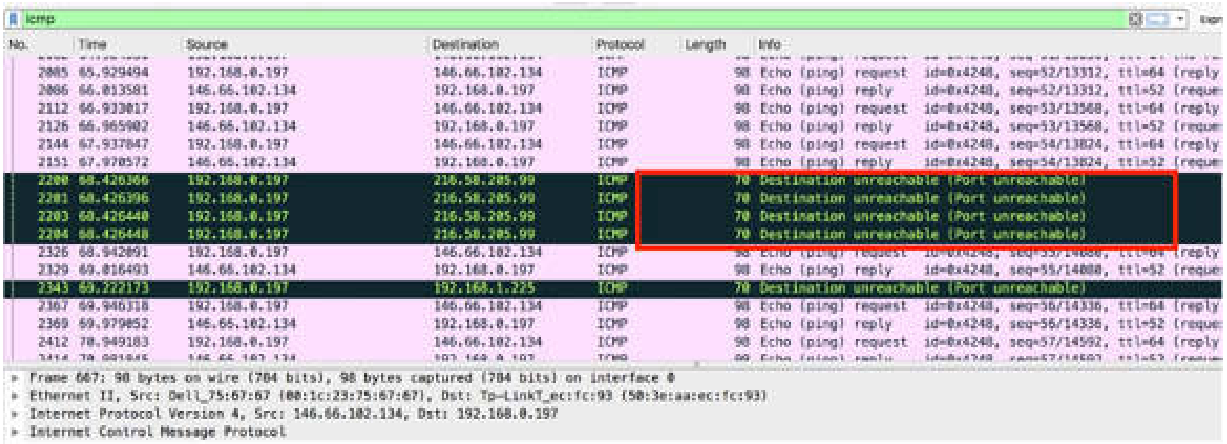
Redirect messages are another common ICMP messages. They are used by routers to notify the hosts on the data link that a better route is available for a particular destination.

Task 2:
Again start capturing packets on the network with the display filter icmp enabled. Run the command ping 101labs.net .



During the ping process, interrupt the physical connection between your modem (router) and the cabled network. The ping request is not able to reach the desired destination (as shown in the figure above where the requests get timed out). The Destination Unreachable messages are generated on the network to tell the source host that its packet could not be delivered.

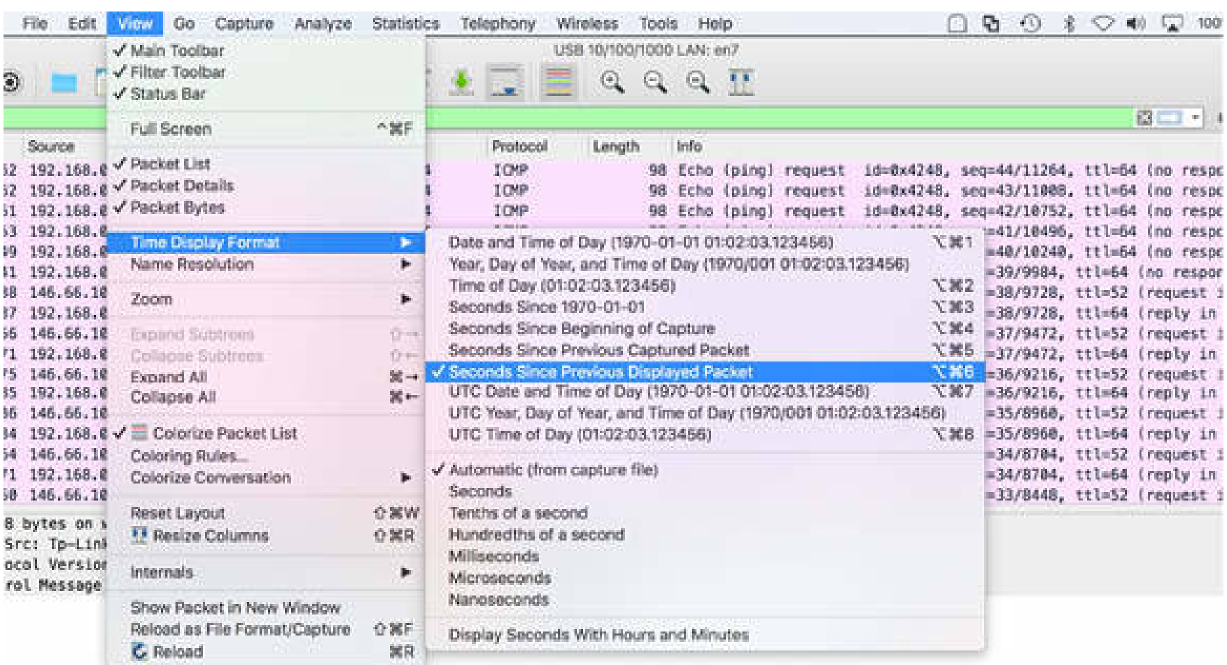
In the figure below, the packets with a black background are Destination Unreachable. The reason for the unreachability is displayed in the Info column.



In case you are monitoring a network and a large number of such packets are detected, it can be an indication of a service not running properly.

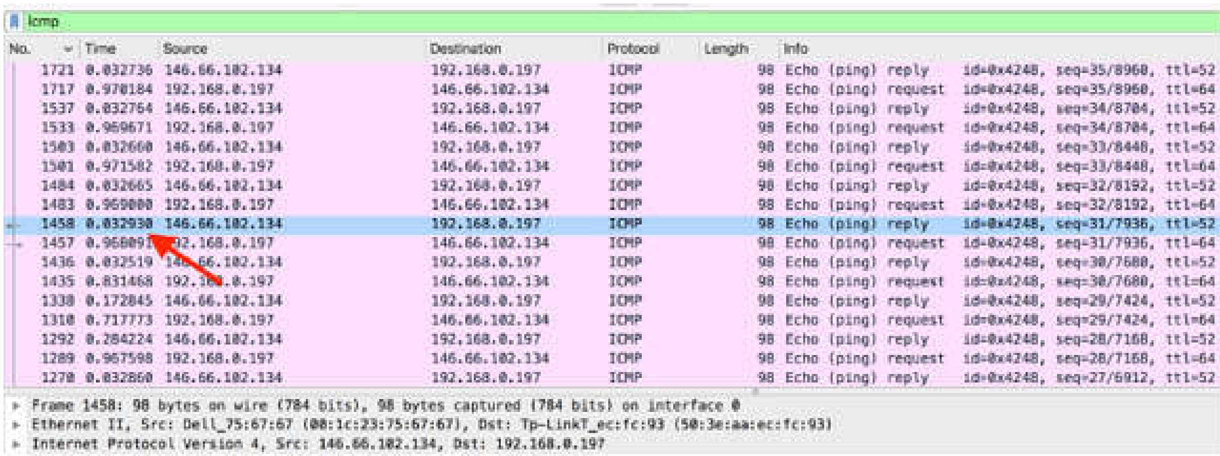
Task 3:

Open the capture saved in Task 2. On the main menu, select View > Time Display Format > Seconds Since Previous Displayed Packet to enable time visualization.



This selection allows you to view the round trip time of the Echo ping request, which is a more accurate value and has higher granularity than the value displayed in the command-line shell.

In the figure below, observe that the round trip time of message #1458 is about 32.93 ms (that is, the time when the reply is detected by Wireshark).



Notes:
Abnormal presence of ICMP messages in a network, except for indicating that something is going wrong with some service, can be a sign of a network attack. The presence of echo messages can be a sign of denial-of-service attack. Similarly, the presence of Destination Unreachable messages can be a sign of a suspicious port scan in progress. Record a few hours of traffic in a stable network and then try to identify if any suspicious ICMP messages are present.

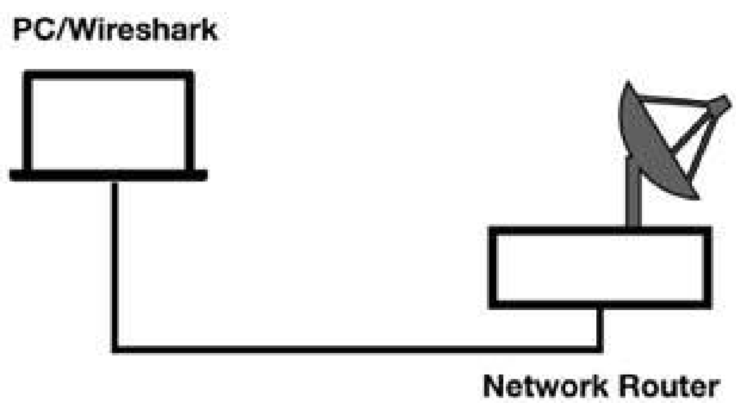
Lab 43. ICMP Problems

Lab Objective:
Learn about the more common ICMP problems.

Lab Purpose:
Learn how to detect and analyze the more common ICMP problems.

Lab Tool:
Wireshark installed on a PC, Ethernet switch/router (cable/Wi-Fi)., and a second PC

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet. Another PC (shown as PC Target in the figure below) is connected to the same network router.

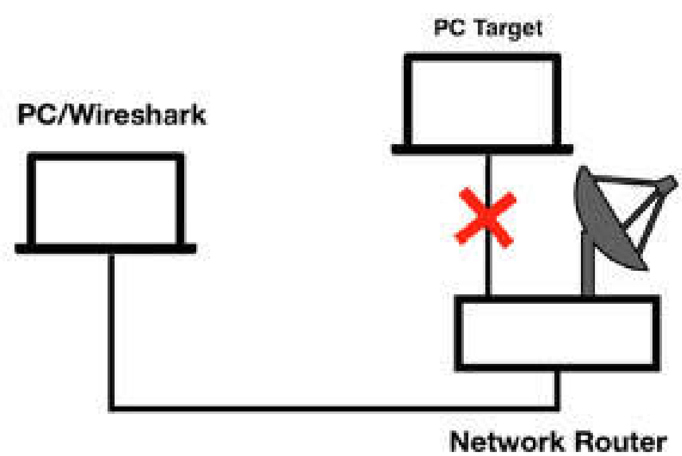


Lab Walkthrough:

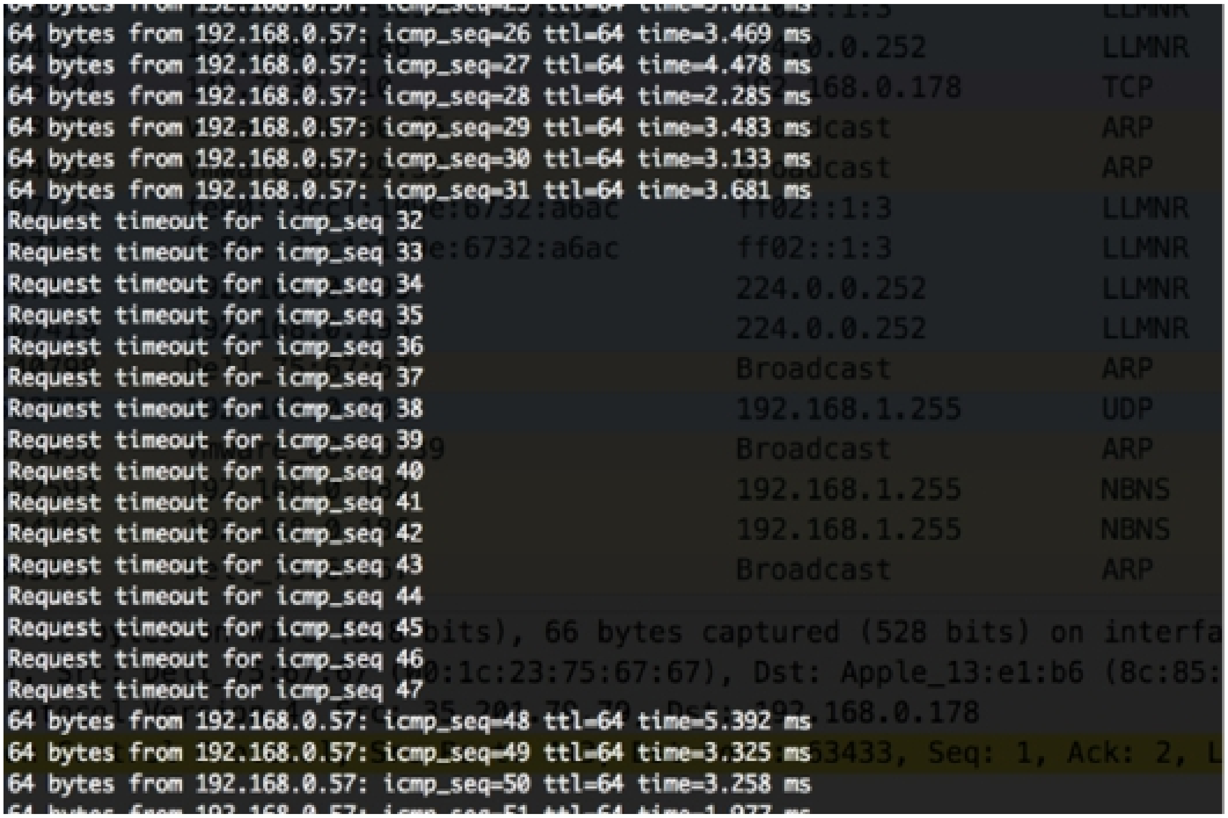
Task 1:
On PC Target, open a terminal window and run the command ip addr show (Linux command) to identify the IP address. In this example, the IP address of PC Target is 192.168.0.57.

On PC Wireshark, open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Open a terminal window and run the command ping 192.168.0.57 .

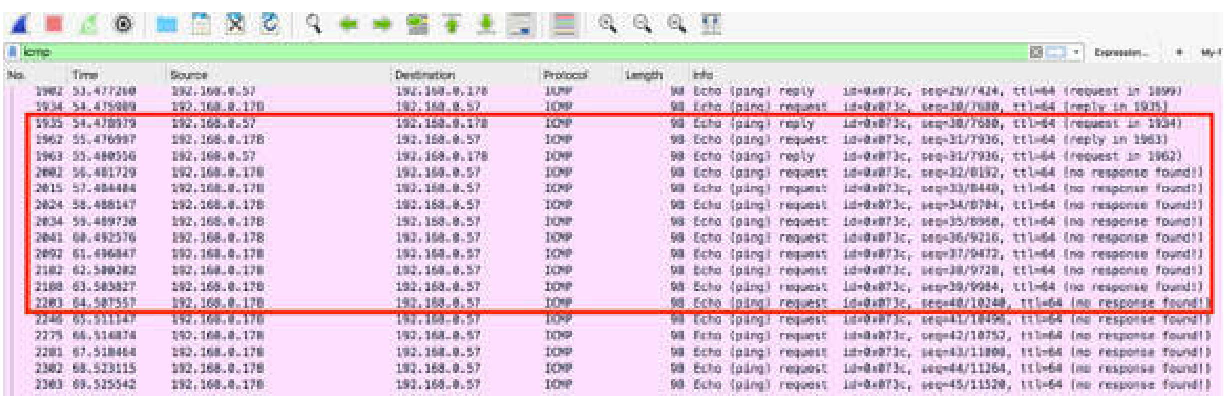
Let the ping run, and after some seconds, interrupt the physical connection between the PC Target and the router.



As a result, no answers to the ping can be identified from the command shell, and the timeout is reached. Reconnect the cable to establish the connection.



In the filter toolbar, enter icmp to display only ICMP packets. In the figure below, the highlighted area of the Packet List pane shows that some of the ICMP requests did not get replies.



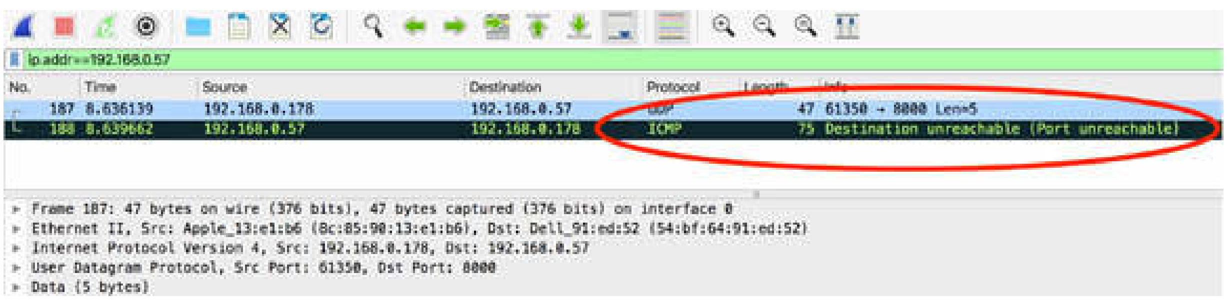
This means that there is no connectivity with the target (because you physically disconnected the target from the network).

Task 2:
To test if a particular port on the target is reachable, send a packet to the PC Target by using the echo command. Open a terminal window, and run the command echo -n “hello” >/dev/udp/192.168.0.57/8000 , as shown in the figure below.

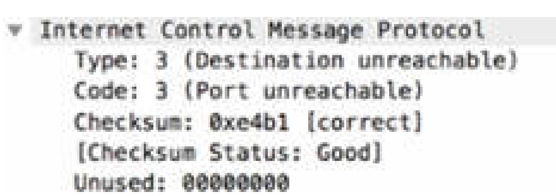
This command checks whether port 8000 is reachable on PC Target (192.168.0.57).

In the Packet List pane, you can identify that the ECHO message (packet #187) did not reach the destination (ip=192.168.0.57 and port=8000).

Moreover, the ICMP response gives Port Unreachable as the reason for failure because PC Target is reachable, but the port isn’t reachable.



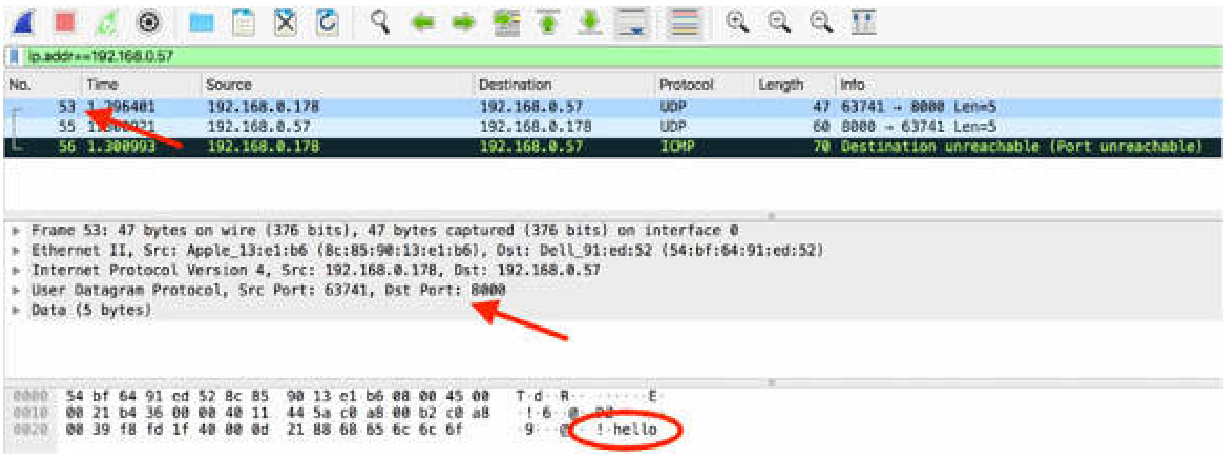
In the Packet List pane, click the ICMP packet. In the Packet Details pane, identify the ICMP response Type and Code, as shown in the figure below. Code value 3 indicates that the port is unreachable.



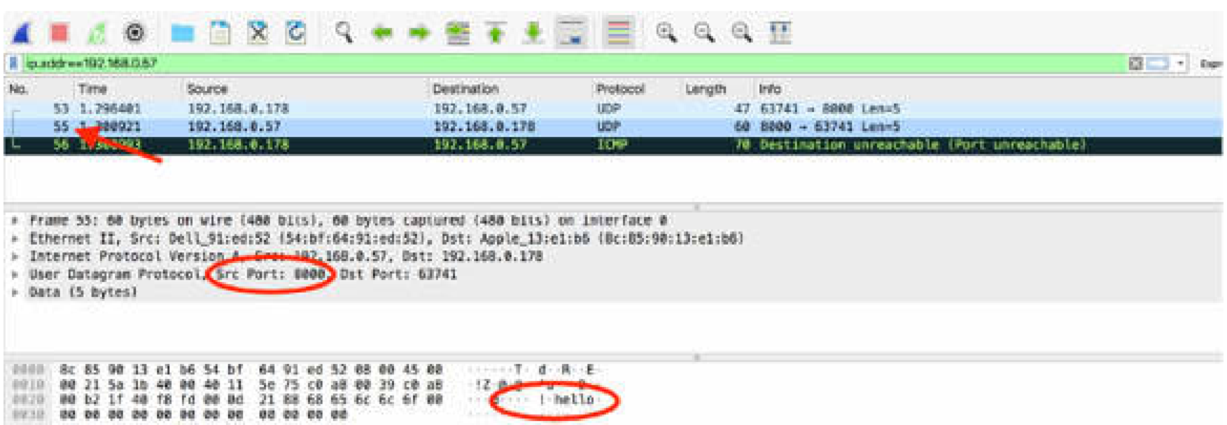
Task 3:

To enable a simple Echo Server for port 8000, on PC Target, open a terminal window and run the command ncat -l 8000 --keep-open --udp --exec "/bin/cat"
.

Repeat the Echo command from PC Wireshark and again start the capture. In the Packet List pane, you can observe that the Echo command receives an answer. The following figure shows the echo message.



The following figure shows the answer to the echo message (as expected). Note that the echo message contains 8000 as the destination port whereas the echo answer contains 8000 as the source port.



Notes:

Repeat the previous steps to test the abnormal presence of ICMP messages in the network caused by a missing physical connection (cable disconnected or broken) and a logical disconnection (a service that is not running).

For information about installing Wireshark on Ubuntu, go to:
https://linuxhint.com/install_wireshark_ubuntu/

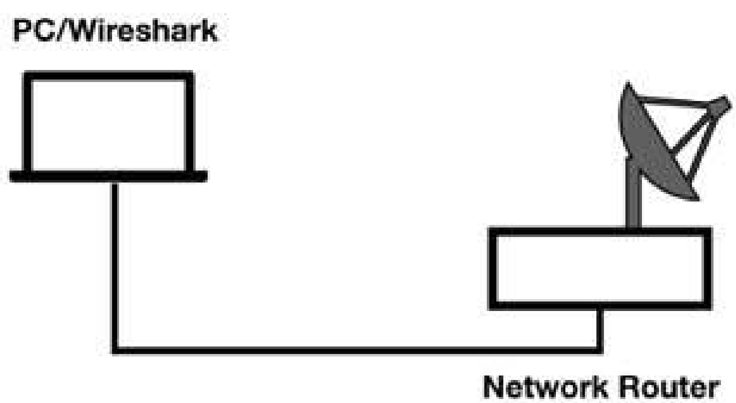
Lab 44. ICMP Packet Structure

Lab Objective:
Learn ICMP packet structure dissection.

Lab Purpose:
Learn how the packet structure of ICMP is composed.

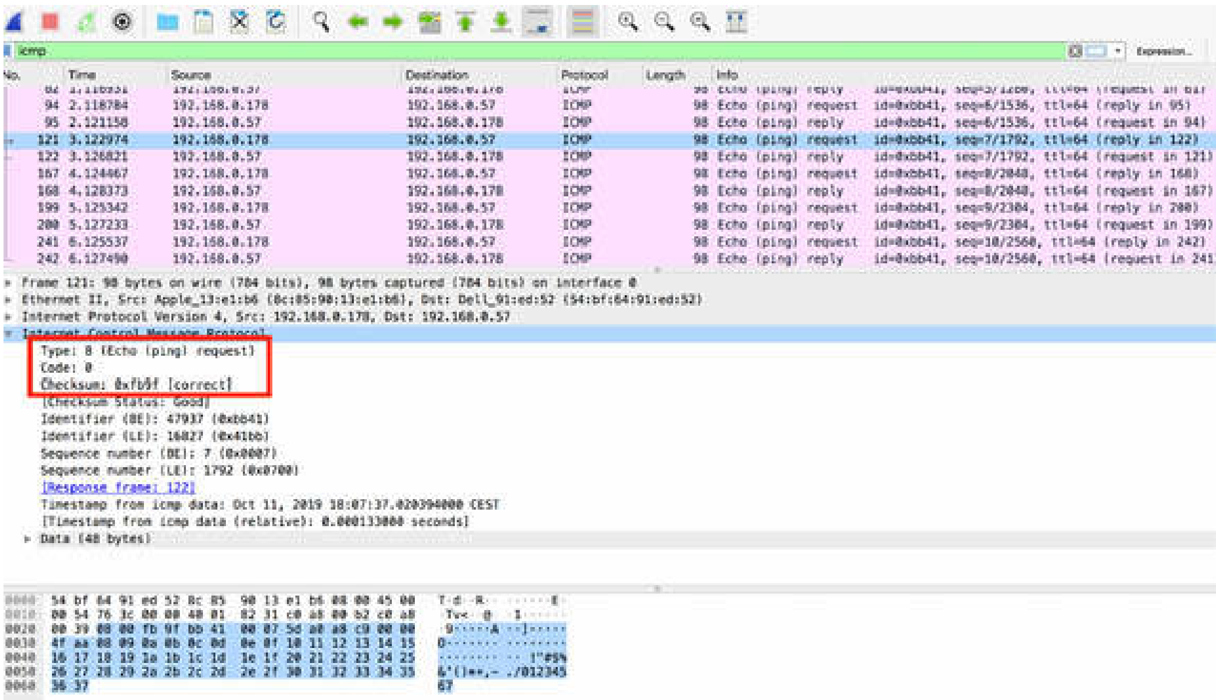
Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
Open the Wireshark capture saved in the previous lab and use the icmp display filter. The result will look like as shown in the figure below.

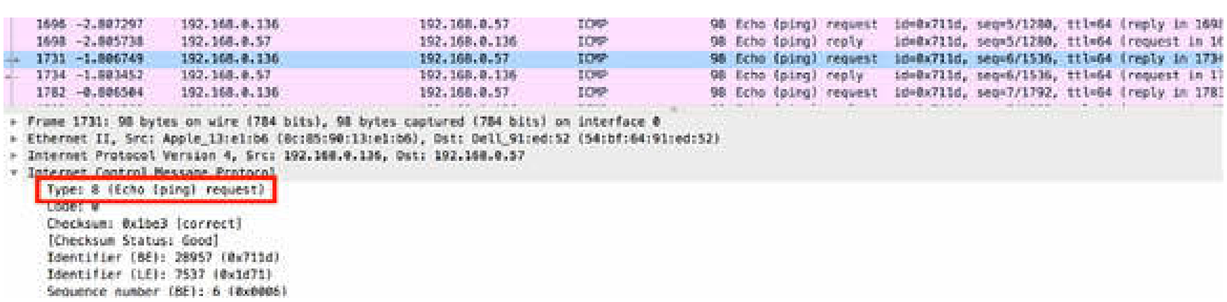


The ICMP packets do not contain the UDP/TCP header. In fact, it is not possible to filter ICMP based on a port number. The only three fields required after the IP header are: Type, Code, and Checksum.

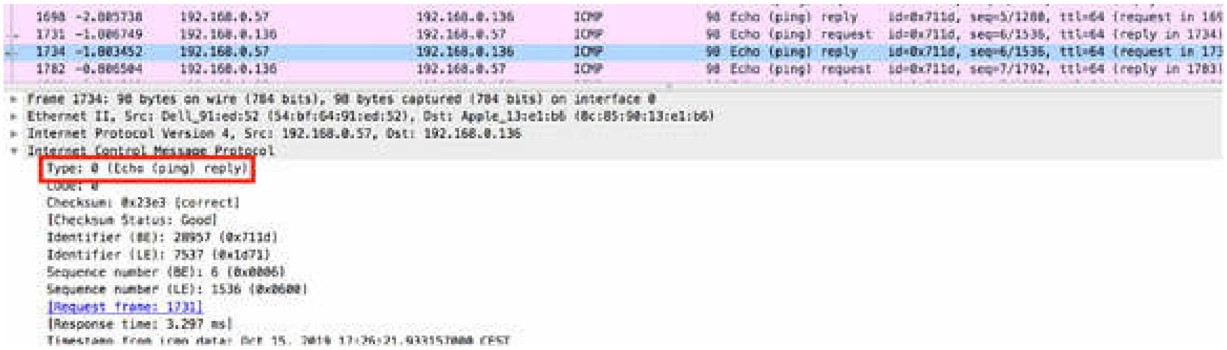
There can be a few specific ICMP packets that contain additional fields to provide additional information or more details about the message, such as, to include a gateway address or a dynamic route.

Task 2:

In the Packet List pane, select an ICMP packet and take a look at the Type field in the Packet Details pane. In the figure below, Type = 8 indicates a ping request. The Type field can also have other possible values.



Type = 0 indicates a ping reply, as shown in the figure below.



The above code is the most common along with Type = 3 (Destination Unreachable), Type = 12 (Parameter Problem).

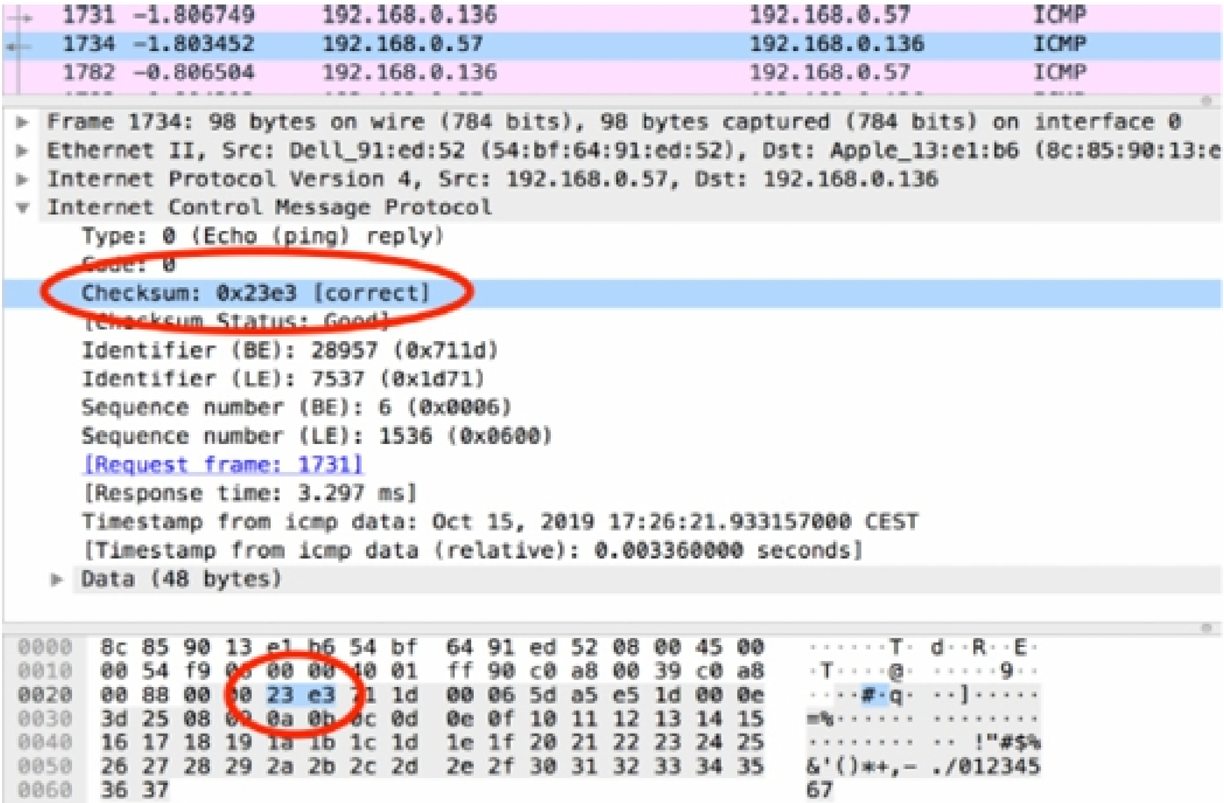
Task 3:
ICMP packet types have several possible Code field values. For Type = 3 (Destination Unreachable), some of the possible values are:

- Code 1: Host Unreachable
- Code 3: Port Unreachable
- Code 6: Destination Network Unknown

For Type = 11 (Time Exceeded Codes), values are:

- Code 0: Time to Live Exceeded in Transit
- Code 1: Fragment Reassembly Time Exceeded

The Checksum field, as shown in the figure below, is calculated from the ICMP header and verifies the offset consistency.



Notes:
Repeat the previous steps to check different types of ICMP messages with different Type and Codes fields.

In addition to the classic ICMP messages, there are the ICMPv6 messages that extend the same functionality to IPv6. The ICMPv6 packet structure is the same as the ICMP packet structure–containing the Type, Code, and Checksum fields. The icmpv6 display filter is used.

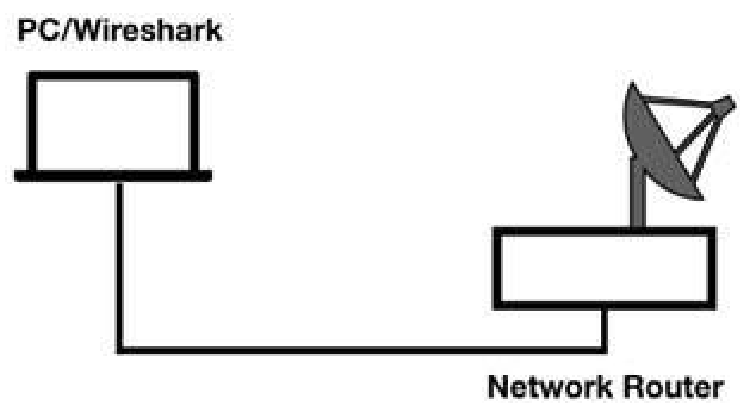
Lab 45. User Datagram Protocol

Lab Objective:
Learn how the User Datagram Protocol (UDP) works and why is it used.

Lab Purpose:
Understand the main purpose of UDP and its features.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

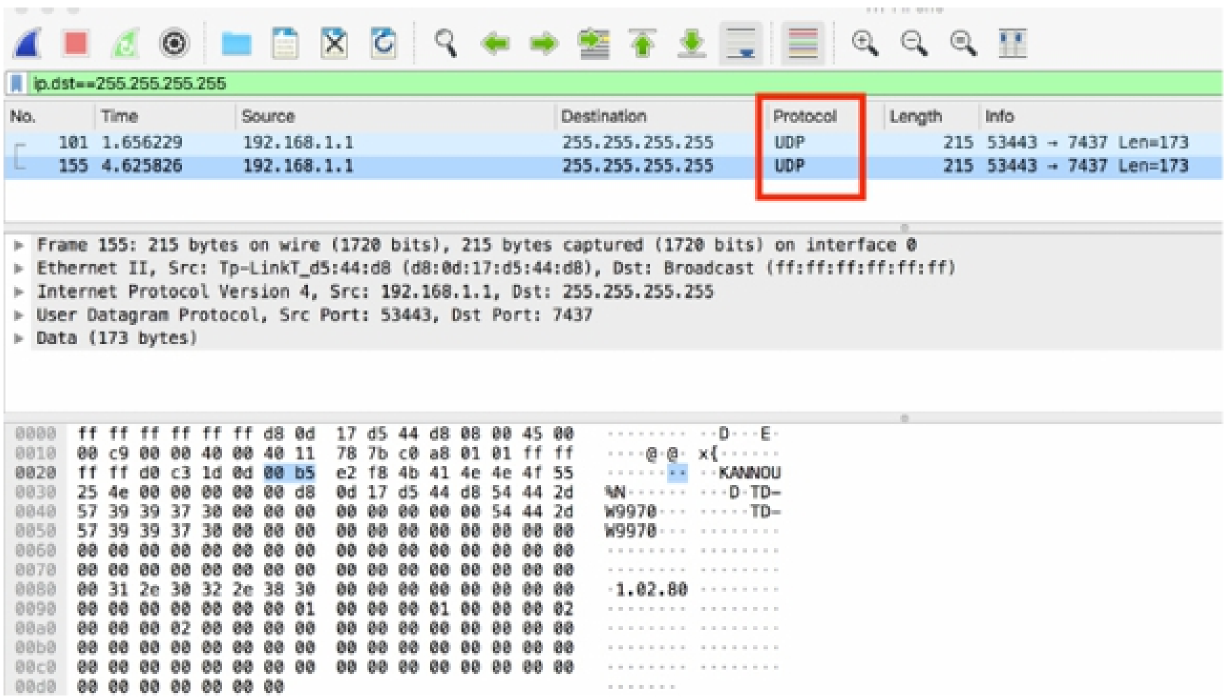


Lab Walkthrough:

Task 1:
UDP is one of the most common protocols used in networking. Open Wireshark, and on the main menu, select Capture > Options. Select an

interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes. Stop the capture and save the file.

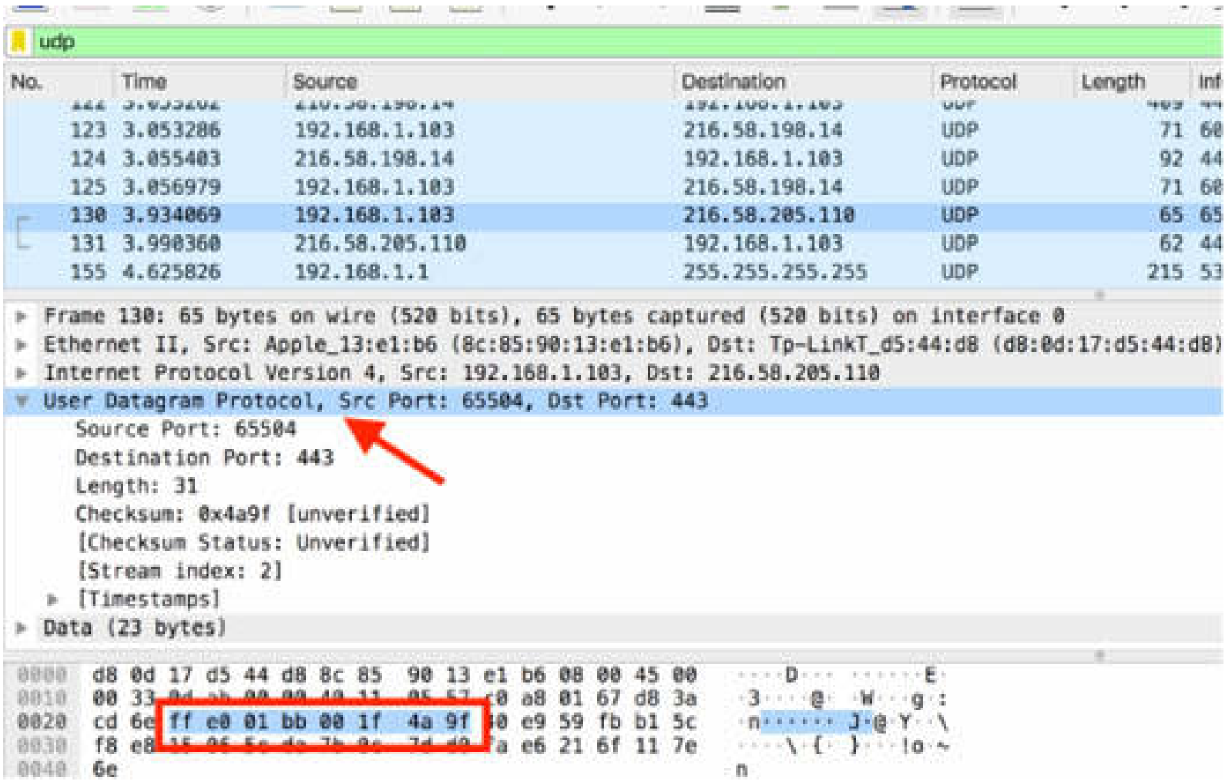
In the filter toolbar, enter ip.dst == 255.255.255.255 to display only UDP frames. In the Packet List pane, only UDP frames are displayed. In fact, if you capture broadcast or multicast traffic, you already have a lot of UDP-based communication, as shown in the figure below. UDP is used for connectionless transport services.



Task 2:

The UDP header port fields identify the application using the transport layer. Considering the fact that UDP uses a simple 8-byte header that consists of four fields, UDP rarely experiences problems during the communication process.

In the filter toolbar, enter udp and select a packet in the Packet List pane. In the Packet Details pane, select “User Datagram Protocol” to open the tree view, as shown in the figure below.



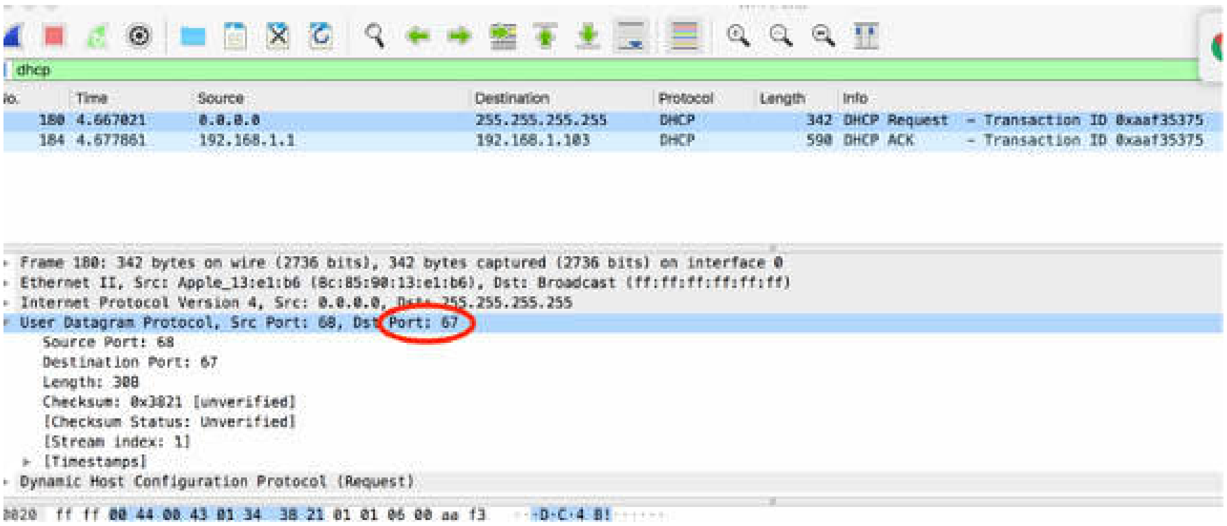
In the Packet Bytes pane, the eight bytes composing the UDP header are automatically highlighted, consisting of source/destination port, length, and checksum.

The most common applications that use UDP are DHCP/BOOTP, SIP, RTP, DNS, TFTP, and various streaming video applications.

Task 3:
Again start a capture in Wireshark.

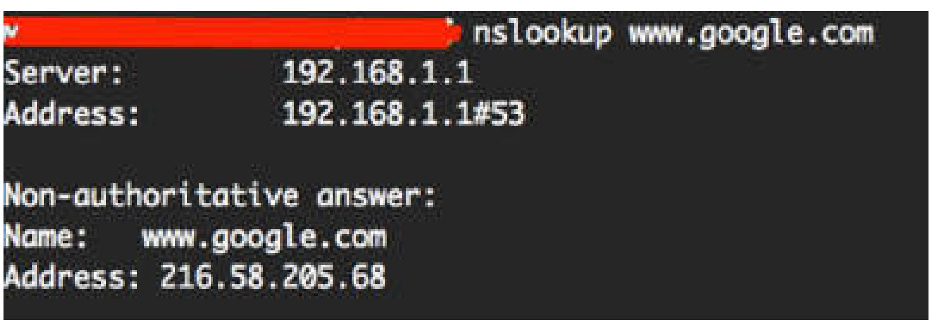
Open a terminal window and run the command `sudo dhclient en0`, where `en0` is your active network interface. This forces your network card to again send a DHCP request to the server.

Stop the capture, and in the filter toolbar, enter `dhcp` , as shown in the figure below.

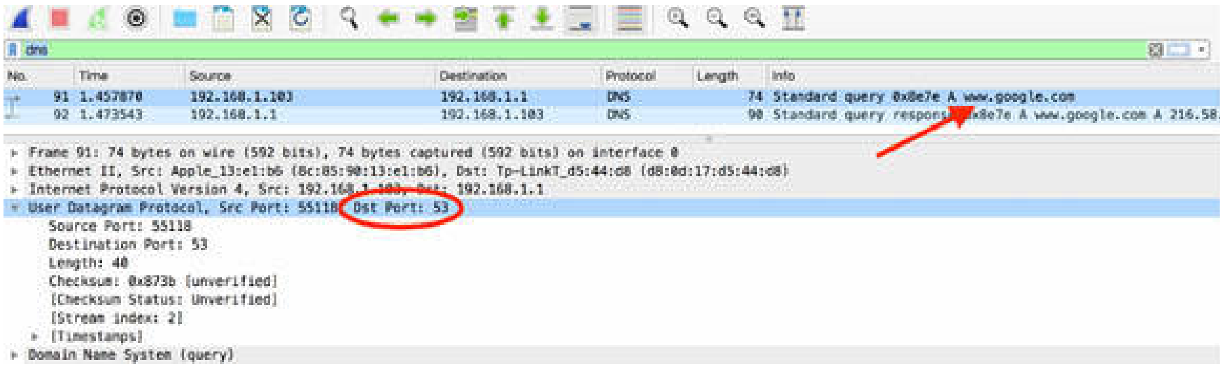


In the Packet Details pane, the highlighted area shown in the figure above, confirms that DHCP requests use the destination port 67 for the DHCP server. On the client side, the source port is a temporary port—it can be different each time.

Task 4:
Again start a capture in Wireshark. Open a terminal window and run the command nslookup [**www.google.com**](http://www.google.com) to send a DNS request to a server.



In Wireshark, in the filter toolbar, enter dns to display the DNS packets, as shown in the figure below.



In the Packet Details pane, the highlighted area shown in the figure above, confirms that DNS requests use the destination port 53. On the client side, the source port is a temporary port—it can be different each time.

Notes:

Repeat the previous steps to check different types of UDP messages (DHCP/DNS) or find a way to send a TFTP or SIP request to a server and inspect the UDP features on the captured frames.

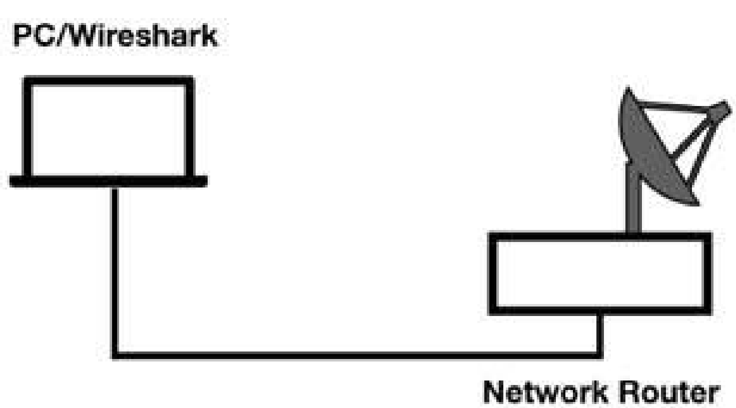
Lab 46. UDP Problems and Packet Structure

Lab Objective:
Learn about the more common UDP problems and the UDP packet structure.

Lab Purpose:
Learn how to detect and analyze the more common UDP problems. Learn about the structure and various fields of a UDP packet.

Lab Tool:
Wireshark installed on a PC, Ethernet switch/router (cable/Wi-Fi)., and a second PC

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet. Another PC (shown as PC Target in the figure below) is connected to the same network router.



Lab Walkthrough:

Task 1:

There are a few situations that can cause issues in UDP communication. In this task, we will identify one such case.

Open a terminal window and run the command ping www.101labs.net to get the IP address of the 101labs.net server, as shown in the figure below.

```
[redacted] ping www.101labs.net
PING 101labs.net (146.66.102.134): 56 data bytes
64 bytes from 146.66.102.134: icmp_seq=0 ttl=44 time=155.998 ms
64 bytes from 146.66.102.134: icmp_seq=1 ttl=44 time=383.805 ms
64 bytes from 146.66.102.134: icmp_seq=2 ttl=44 time=138.610 ms
```

The IP address of the 101labs.net server is 146.66.102.134.

Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column.

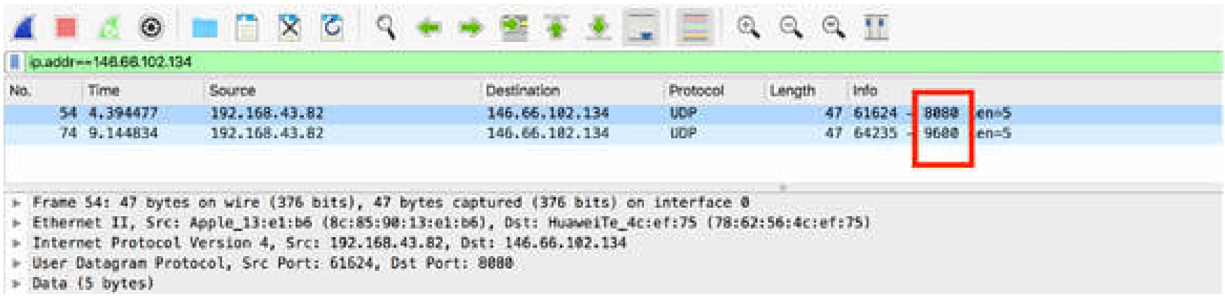
In a terminal window, run the Linux command echo -n “hello” >/dev/udp/146.66.102.134/8080 . Wait for a few seconds, and run the command echo -n “hello” >/dev/udp/146.66.102.134/9600 , as shown in the figure below.

```
echo -n "hello" >/dev/udp/146.66.102.134/8080
echo -n "hello" >/dev/udp/146.66.102.134/9600
```

Stop the capture and save the file.

The commands above send a UDP “hello” packet to the specified address and UDP port (indicated by the last parameter). The first command sends the packet to port 8080; the second command sends the packet to port 9600.

Open the saved capture file. In the filter toolbar, enter ip.addr == 146.66.102.134 . In the Packet List pane, only the packets directed to or coming from the IP address 146.66.102.134 are displayed, as shown in the figure below.



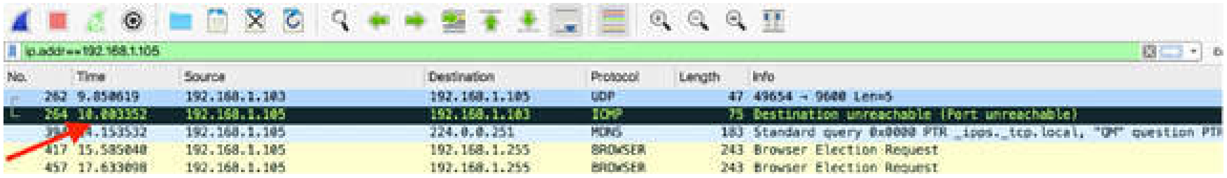
Only the packets sent to 146.66.102.134 are displayed. No packets coming from 146.66.102.134 are displayed because the server firewall silently discarded them, and so the client didn't receive any answer. This is one of the most common UDP issues on the network where a firewall filters the requests and no answer is detected.

Task 2:

On PC Target, open a terminal window and run the command ifconfig to identify the IP address. In this example, the IP address of PC Target is 192.168.1.105.

On PC Wireshark, open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Open a terminal window and run the command echo -n "hello" >/dev/udp/192.168.1.105/9600 to send a UDP "hello" packet to PC Target on port 9600.

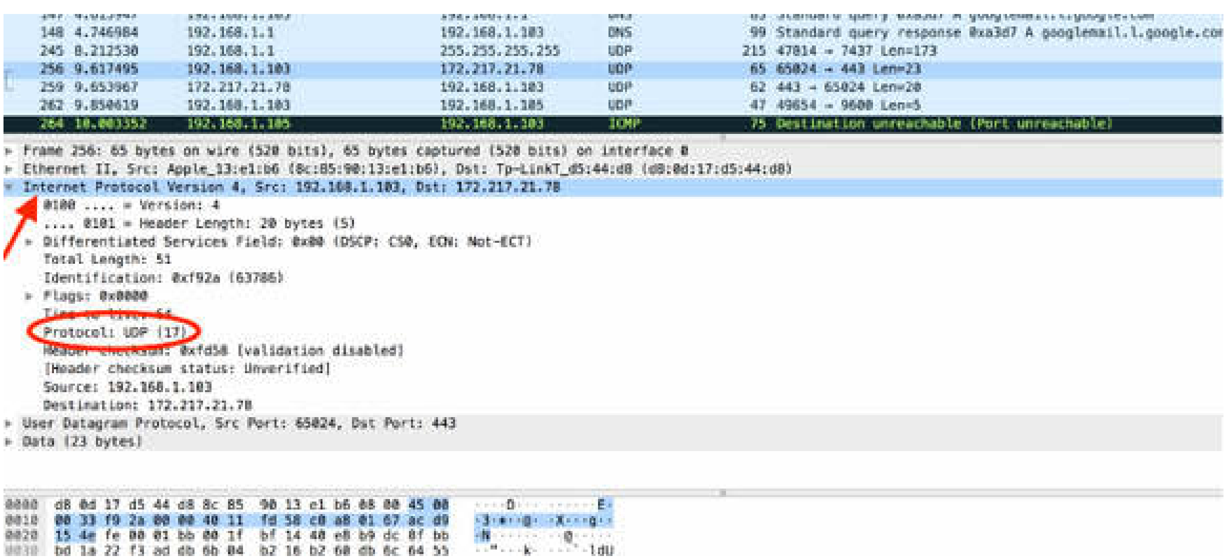
Stop the capture and save the file. In the filter toolbar, enter ip.addr == 192.168.1.105 to display only the packets of interest in the Packet List pane.



As shown in the figure above, considering that there is no firewall enabled on PC Target, when you send a UDP “hello” packet to a random port, the ICMP Destination Unreachable/Port Unreachable response is triggered because the port is not open.

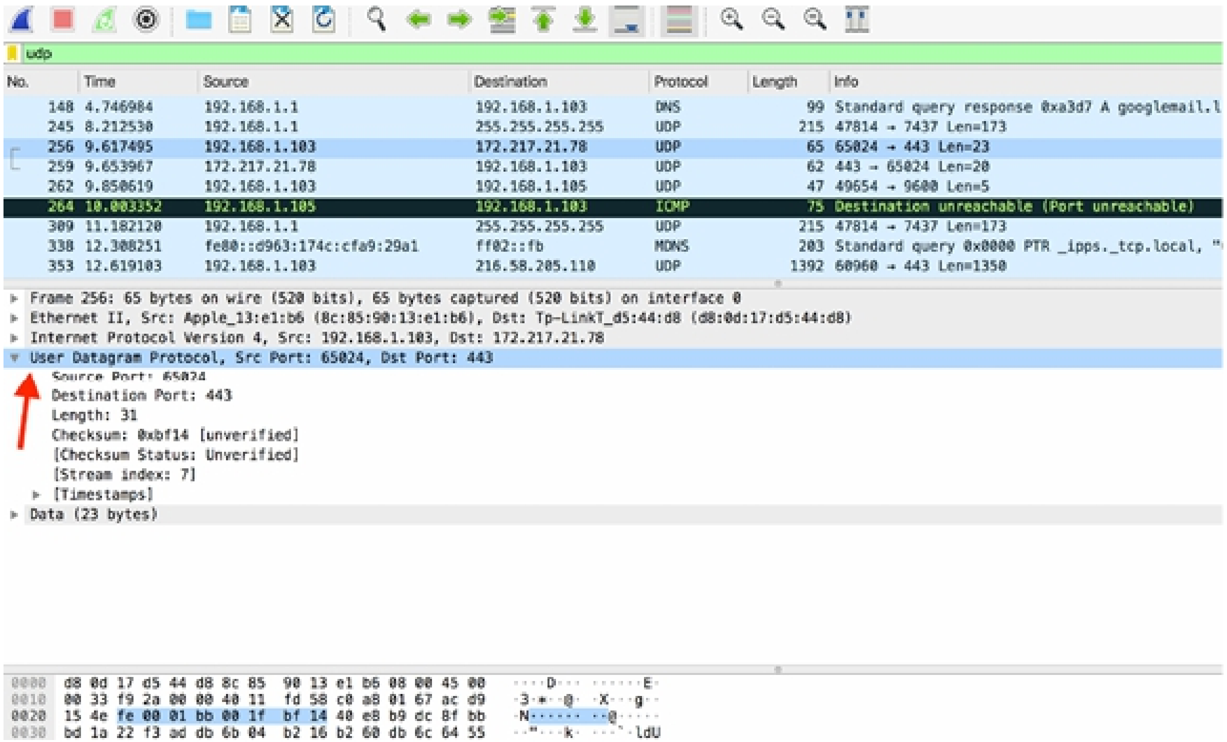
This example shows what you can see during a network scan. If the firewall is not enabled on the target of the scan, you see the Destination Unreachable messages. If the firewall is enabled, it usually blocks the scan and no ICMP packets are visible. If a service or attacker is trying to find an open port in the server, inspecting a network capture can reveal the attacker.

Task 3:
Open the network capture saved in the previous tasks. In the filter toolbar, enter udp . In the Packet List pane, select a packet. In the Packet Details pane, click the “Internet Protocol Version” field to open the tree view, as shown in the figure below.



As shown in the figure above, the UDP header is defined with value 17 (0x11) in the IP header Protocol field.

In the Packet Details pane, click the “User Datagram Protocol” field to open the tree view, as shown in the figure below.



As shown in the figure above, the UDP header is always 8-bytes long, and it contains the following four fields:

- Source Port: This field has the same purpose in TCP and UDP, that is, to open a listening port for response packets. In some cases, it also defines the application or protocol that sends the packet.
- Destination Port: This field defines the destination application or process for the packet. In some cases, the source and the destination port numbers are the same for the client and the server. In other cases, the client uses a temporary port number, and the server uses a well-known port number for communication.

- Length: This field defines the length of the packet—from the UDP header to the end of valid data (not including any data link padding).
- Checksum: This field's value is calculated using the contents of the UDP header (except the checksum field itself).

Notes:
To test the presence or absence of UDP issues, repeat the previous steps, and test different scenarios for yourself.

Remember that UDP is useful for simple communication. UDP's biggest limitation is that it doesn't provide feedback. If no answer is received from the client, you can't be sure whether the UDP message reached the destination.

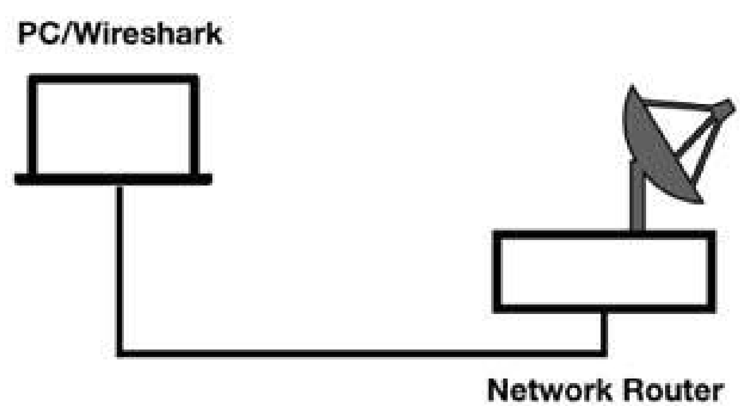
Lab 47. Transmission Control Protocol

Lab Objective:
Learn how the Transmission Control Protocol (TCP) works and why is it used.

Lab Purpose:
Understand the main purpose of TCP and its various features.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

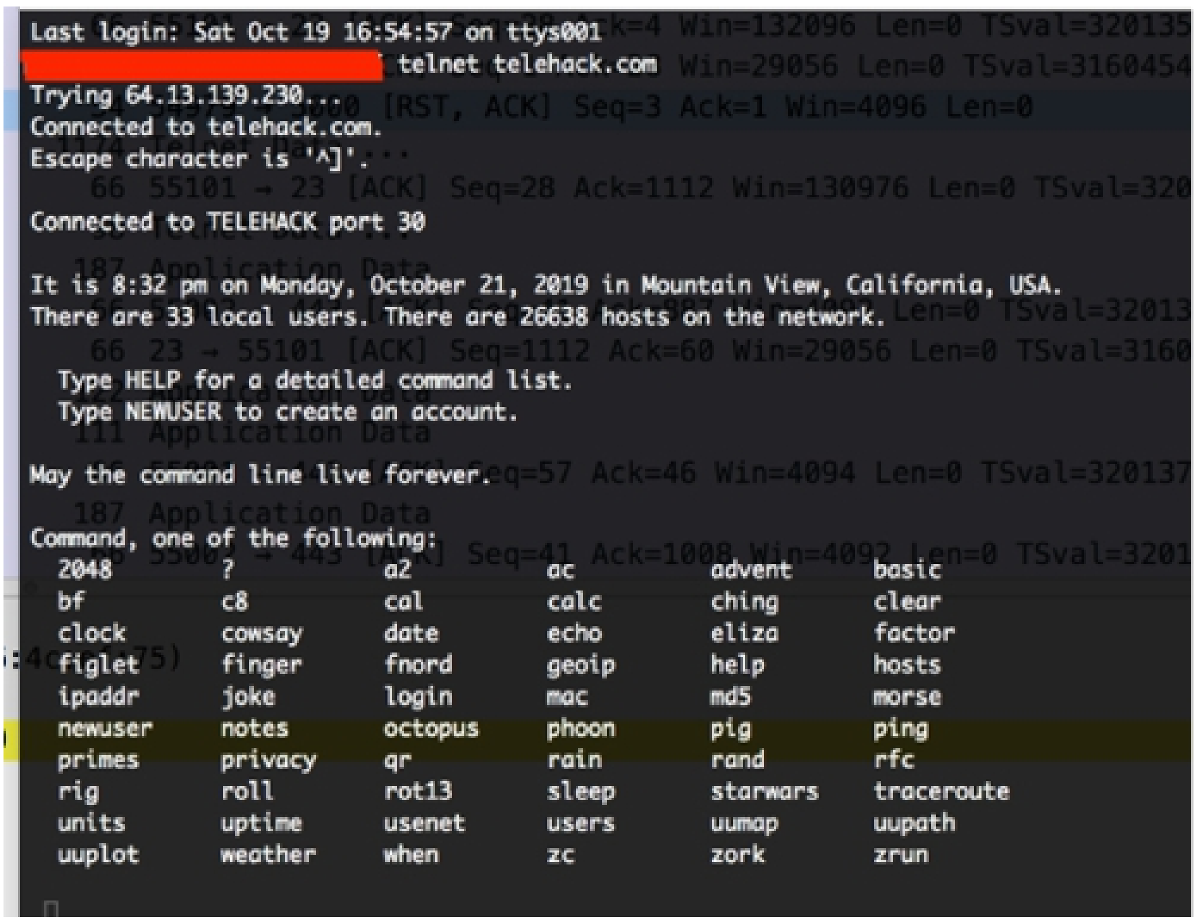


Lab Walkthrough:

Task 1:
TCP is used for connection-oriented communication. The connection begins with a handshake between two devices. Every data is in sequential order, and each packet is acknowledged to ensure delivery and automatic recovery (in case of lost packets).

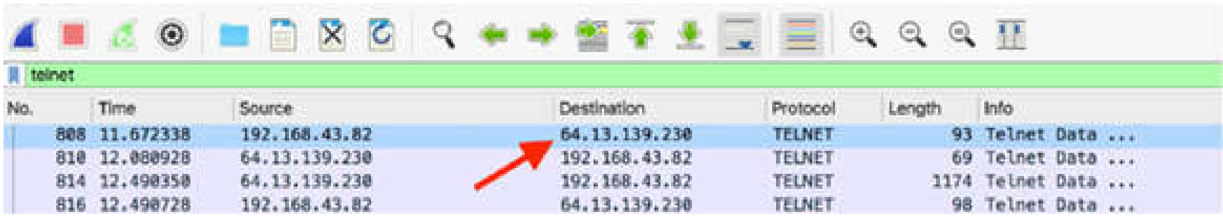
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

Open a terminal window and run the telnet telehack.com command to connect to a remote server, as shown in the figure below.

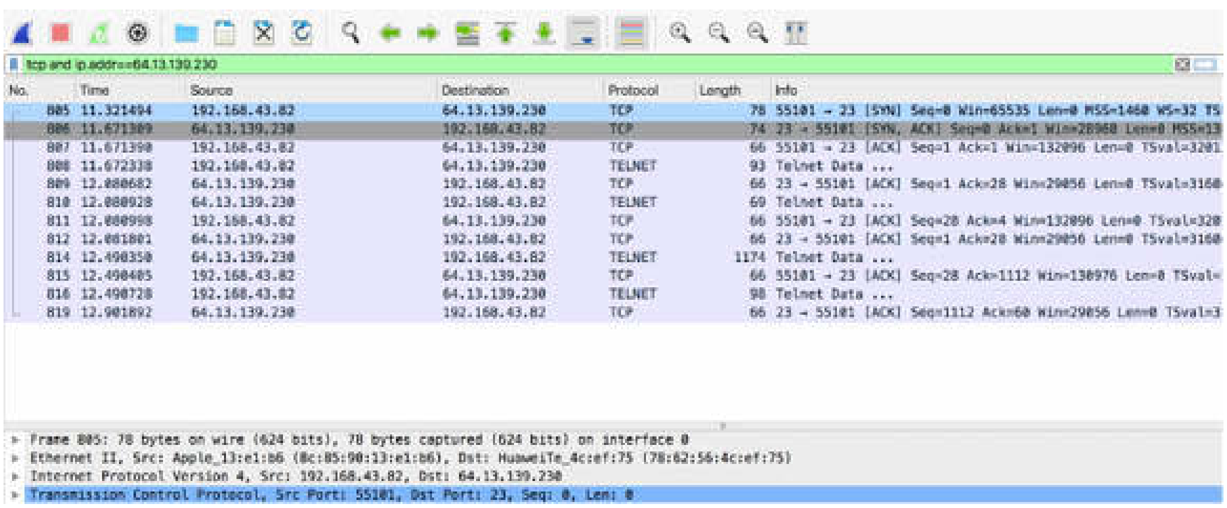


Stop the capture and save the file.

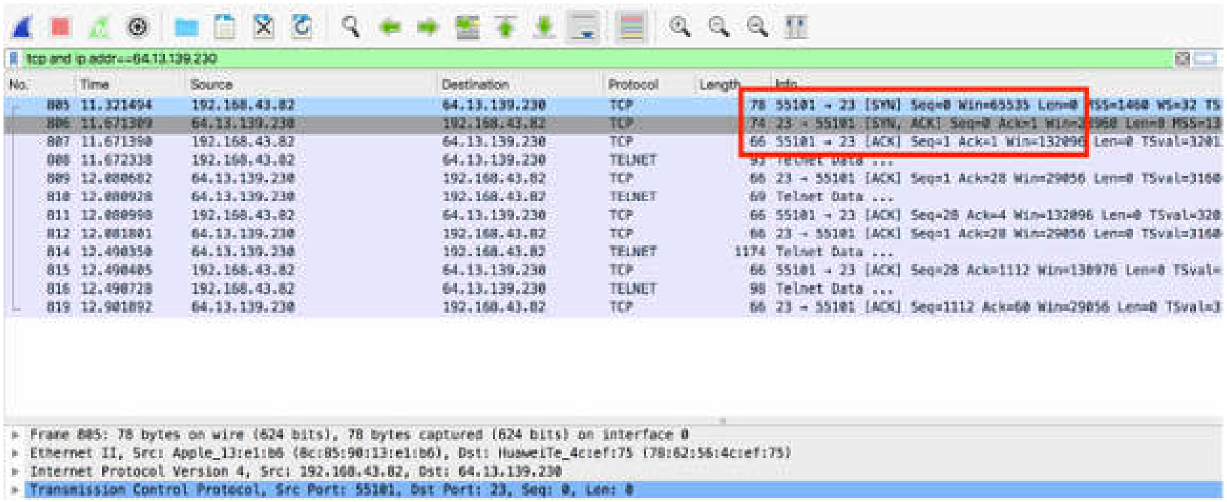
In Wireshark, in the filter toolbar, enter telnet to display only Telnet packets in the Packet List pane, as shown in the figure below.



By using the telnet telehack.com command earlier, you retrieved the IP address of the Telnet server (64.13.139.230). Use this IP address to create another filter to select the TCP connection. In the filter toolbar, enter tcp and ip.addr == 64.13.139.230 . The Packet List pane displays the results, as shown in the figure below.



The three packets (#805, #806, and #807) indicate that a TCP connection was established. Every TCP connection starts with these packets that are in the following order: SYN, SYN/ACK, ACK. These packets form the classic TCP handshake.



The SYN packets synchronize the sequence numbers to ensure that both sides know each other's starting sequence numbers (the Initial Sequence Number or ISN). This is how they keep track of the sequence of the data exchanged between them.

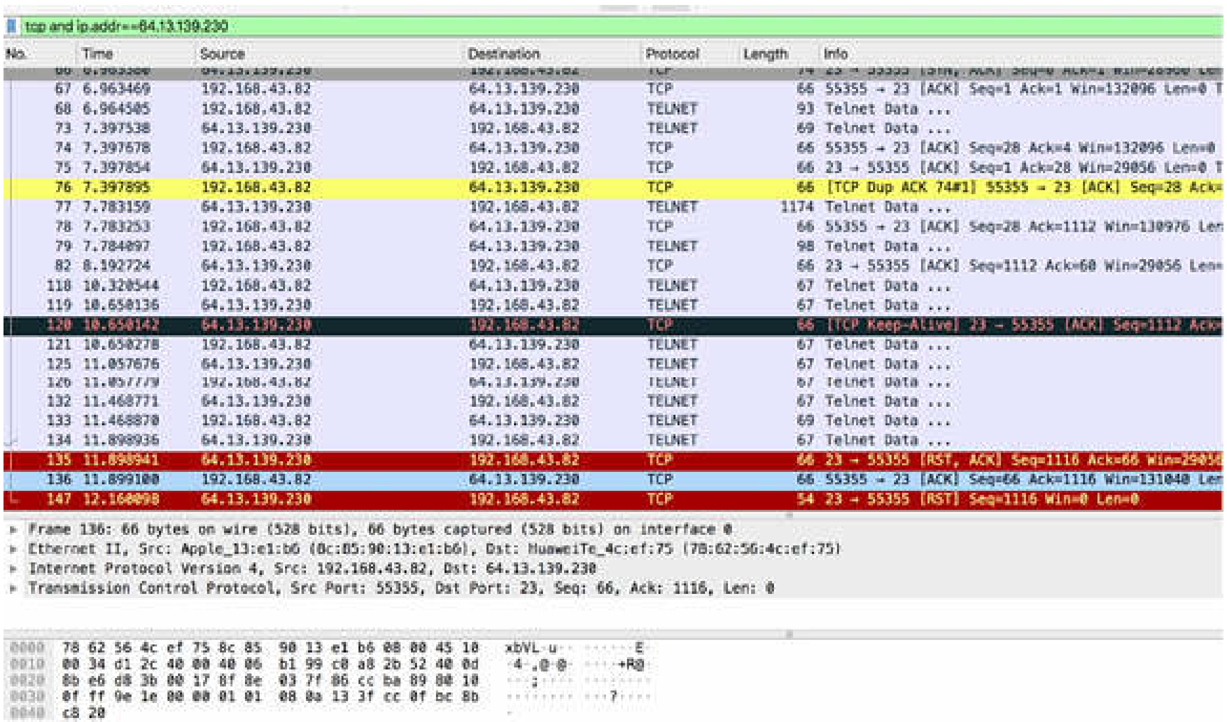
In the above figure, host 192.168.43.82 establishes a TCP connection to 64.13.139.230. In the Info column, packet #805 contains [SYN], packet #806 contains [SYN, ACK], and packet #807 contains [ACK].

Task 2:

In Wireshark, again capture some packets, and use the tcp and ip.addr == 64.13.139.230 display filter. In the terminal window, type exit to terminate the Telnet session, which also implies the termination of the TCP connection.

TCP connections can be terminated in several ways. An explicit termination uses TCP Resets (RST); an implicit termination uses TCP FIN packets. When FIN is used, a host sends a FIN packet and enters the FIN-WAIT state until its FIN is acknowledged, and the peer sends its own FIN back.

In this case, the termination is done by using an explicit termination, as shown in the figure below.



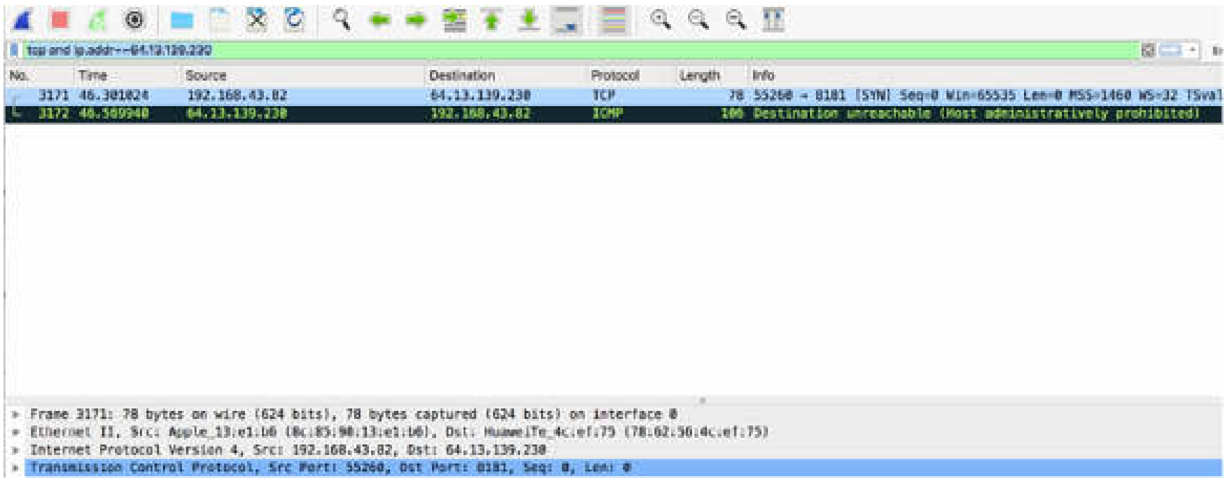
As shown in the figure above, the server sends an RST message (packet #135), and the client sends an ACK (packet #136) message accepting to terminate the connection. The reset may be preceded by FINs in few cases.

Task 3:

In Wireshark, capture some packets on the active network interface. Open a terminal window and run the command telnet telehack.com 69 to change the default Telnet port used, as shown in the figure below.

In this case, the connection is not established.

In Wireshark, use the tcp and ip.addr == 64.13.139.230 display filter. The Packet List pane displays the results, as shown in the figure below.



This means that, most likely, the target port 69 is firewalled through software. The ICMP Destination Unreachable response is being generated by the firewall. If the target host cannot be reached, the router may also respond with an ICMP message.

In the Packet List pane, select an ICMP packet. In the Packet Details pane, the ICMP Destination Unreachable packet has Type = 3 and Code = 9.

The Code value can be one of the following:

- Code 1: Host Unreachable
- Code 2: Protocol Unreachable
- Code 3: Port Unreachable
- Code 9: Communication with the Network is Administratively Prohibited
- Code 10: Communication with the Host is Administratively Prohibited
- Code 11: Destination Unreachable for the Type of Service

If the target server does not have a process listening on port 69, it responds to the SYN packet with a TCP Reset. If a TCP SYN does not receive any response, it is assumed that:

1. The SYN packet did not arrive at the target.

2. The SYN/ACK did not make it back to the host for some reason.
3. A host-based firewall silently discarded the SYN packet.

In such a case, the TCP stack automatically retransmits the SYN to attempt to establish the connection. TCP stacks vary in the number of times they reattempt a connection.

Notes:
Repeat the previous steps to connect to a Telnet server with an allowed port or to connect to another server that doesn't have the Telnet service available.

Inspect the different answers in Wireshark. Your results may differ depending on your local or ISP firewall settings (or antivirus) and your operating system.

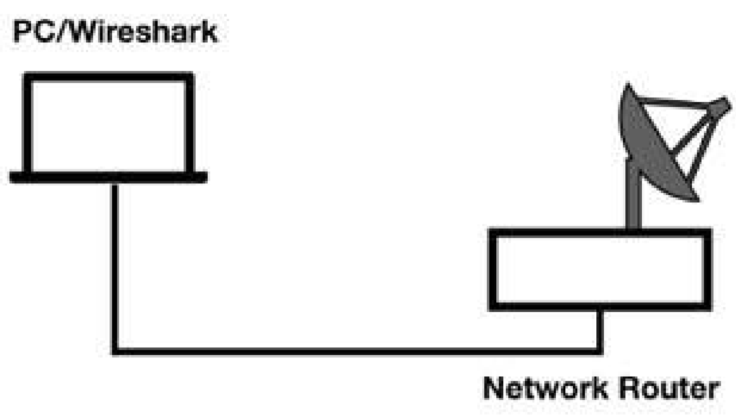
Lab 48. TCP—Sequential Management

Lab Objective:
Learn how TCP manages a sequence of packets and how to handle packet loss.

Lab Purpose:
Understand TCP features related to processing packets in sequential order and managing the packets lost during communication.

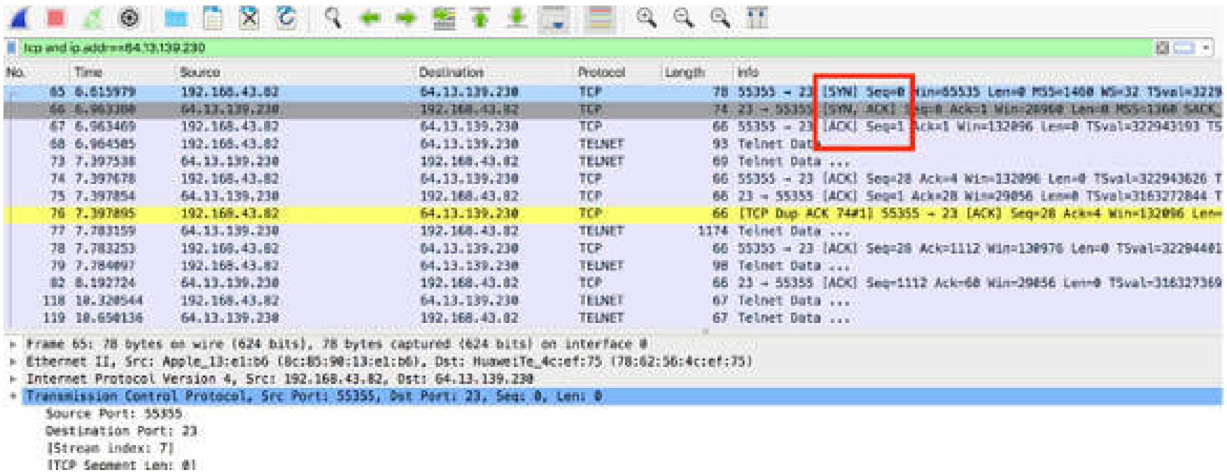
Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



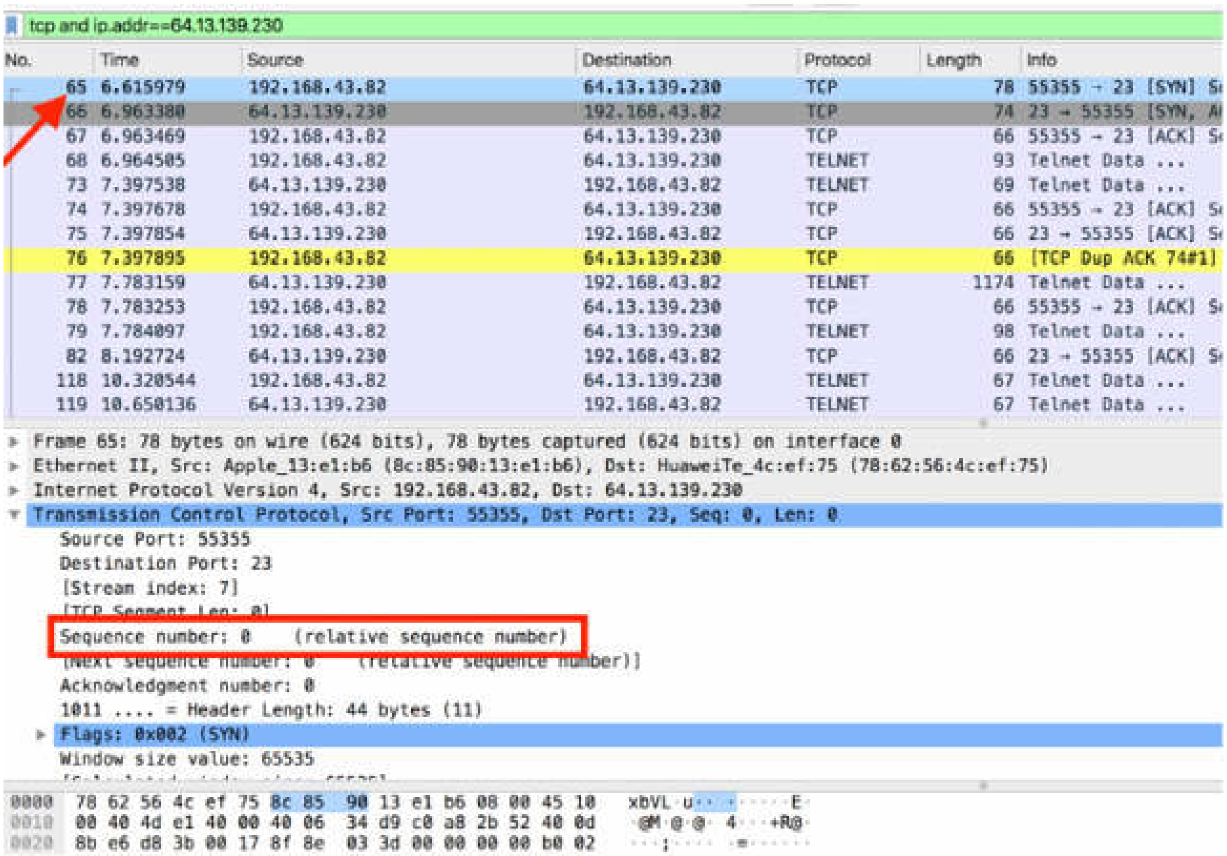
Lab Walkthrough:

Task 1:
In Wireshark, open the capture file saved in the previous lab. In the filter toolbar, enter tcp and ip.addr == 64.13.139.230 to only select a TCP communication between two hosts. The Packet List pane displays the results, as shown in the figure below.

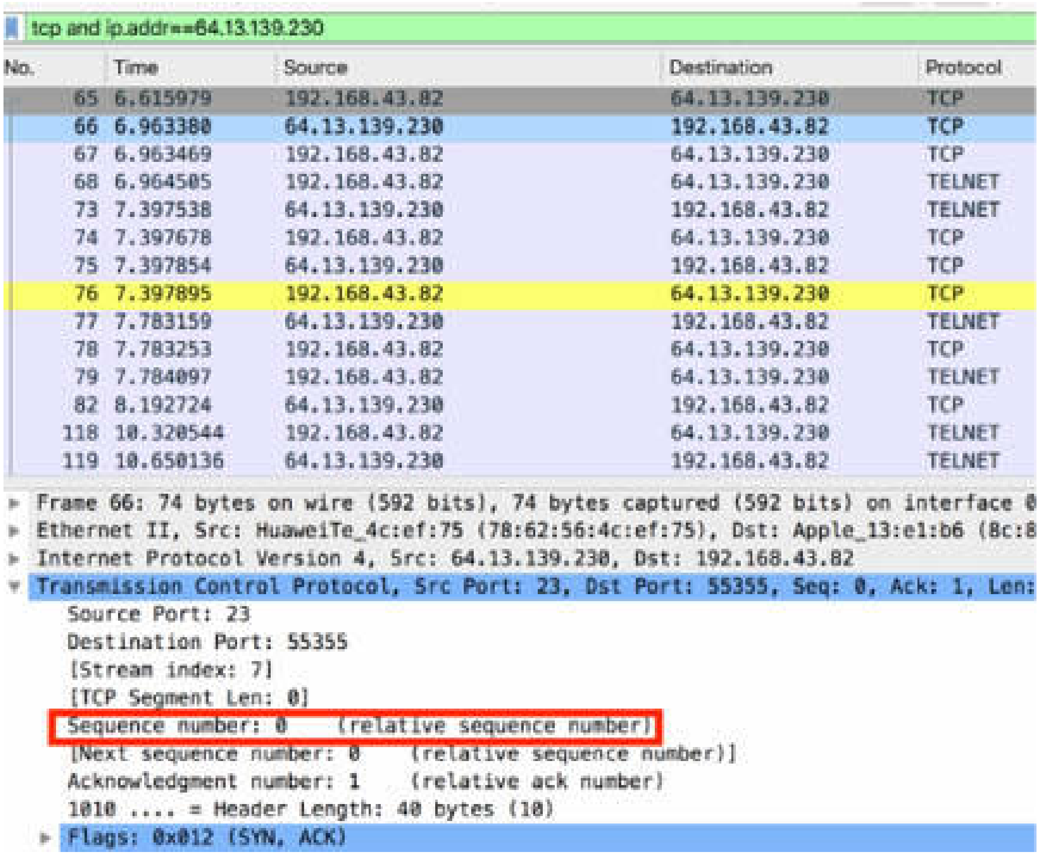


The acknowledgment process is performed in the first three packets. During the sequencing/acknowledgment process, the order of packets is tracked, and the missing segments are detected and recovered. During the handshake process, each side of the connection selects its own starting sequence number (Initial Sequence Number or INS).

In the Packet List pane, select the first TCP packet (SYN, packet #65). In the Packet Details pane, click the "Transmission Control Protocol" field to open the tree view. The sequence number (0) of the selected packet is displayed, as shown in the figure below.



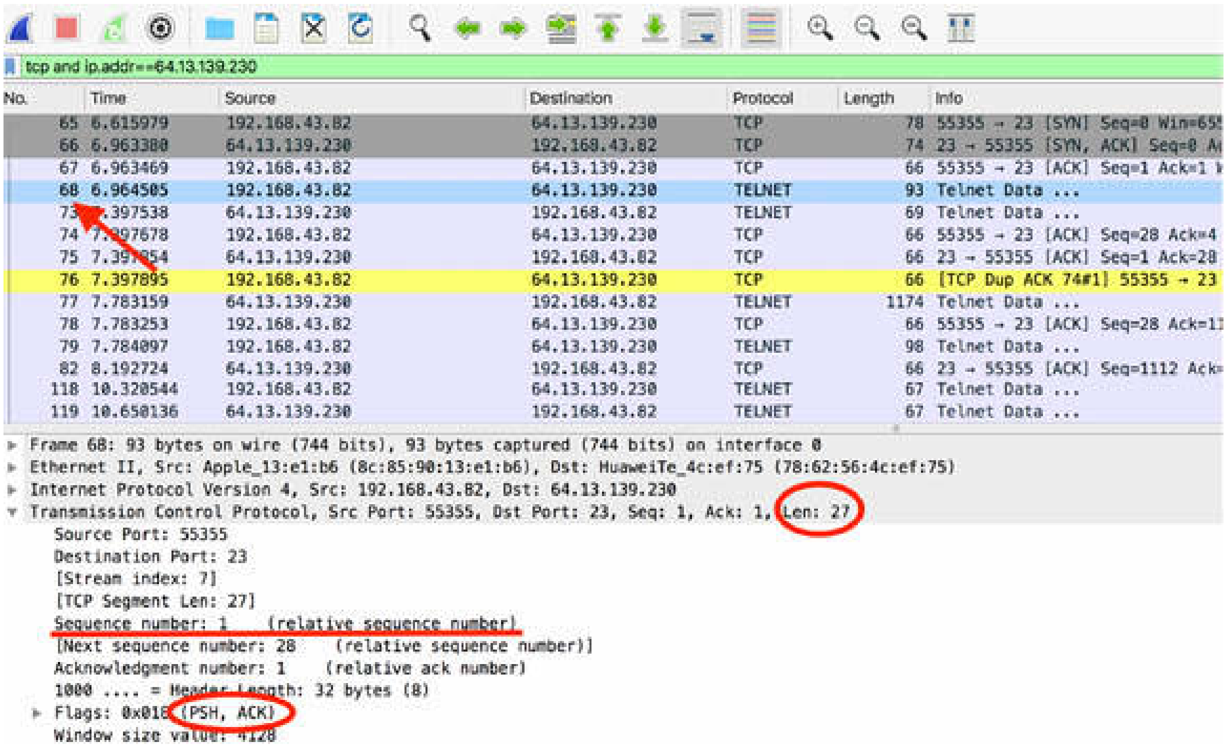
In the Packet List pane, select the second TCP packet (SYN/ACK, packet #66). In the Packet Details pane, click the “Transmission Control Protocol” field to open the tree view. The sequence number (0) of the selected packet is displayed, as shown in the figure below.



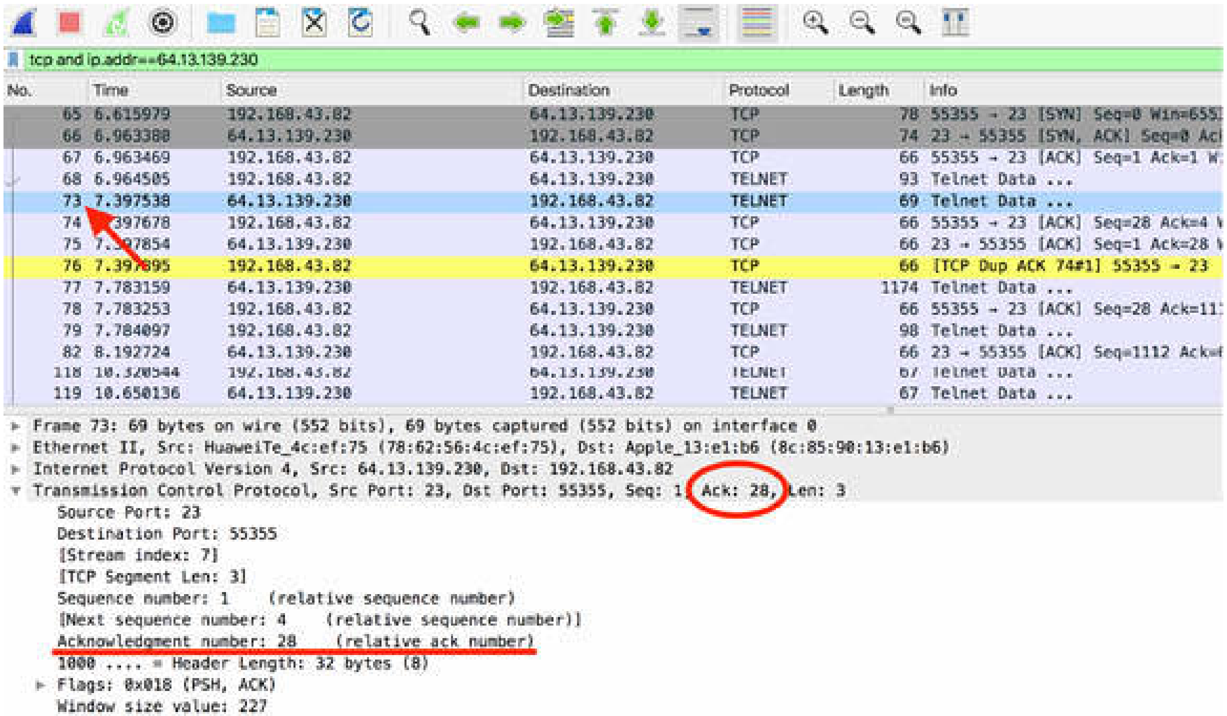
The sequence number is equal to 0. This is the expected behavior considering that this is the first packet sent from the server belonging to this TCP communication. During the handshake process, each side of the connection selects its own starting sequence number. It is important to know that the Initial Sequence Number should be randomized to prevent Sequence Number Prediction Attacks, as defined in RFC 1948.

Each side increments this sequence number by the amount of data included in each packet. To obtain the sequence number to be placed on the ACK message, calculate the sum of the Number of Bytes received and the Last Sequence Number received.

Let’s take a look at a few TCP packets to verify this sequence numbering equation. Select the first TCP message with some data content (i.e., PUSH message—PSH) and inspect the sequence number and the length of data. This information is displayed in the figure below.



In the figure above, in the Packet List pane, packet #68 is selected. It has a length of 27 bytes and sequence number 1. Applying the sequence numbering equation, the sequence number should be 27 + 1 = 28. In the figure below, details of packet #73 are displayed with the highlighted data of interest.

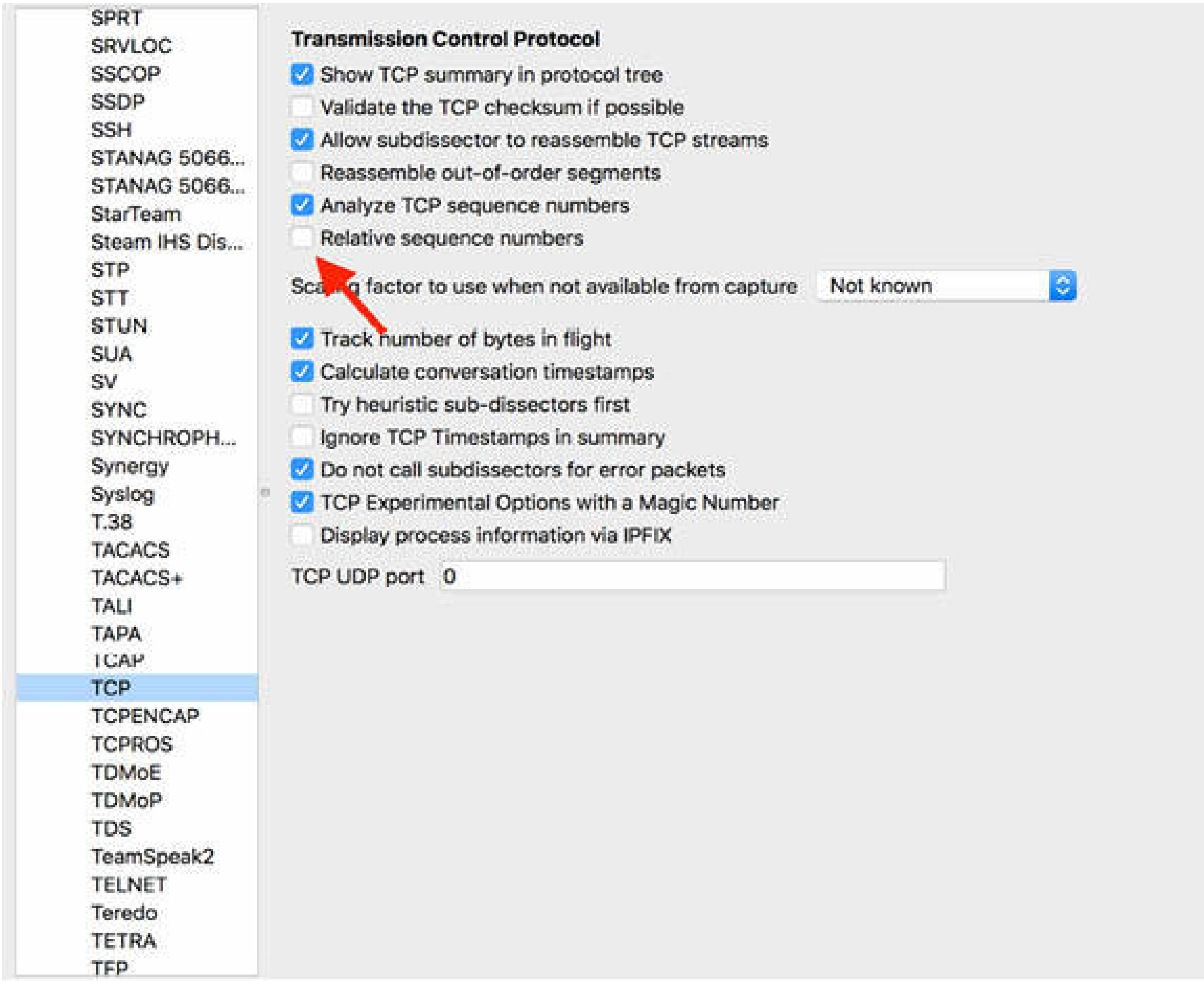


Task 2:

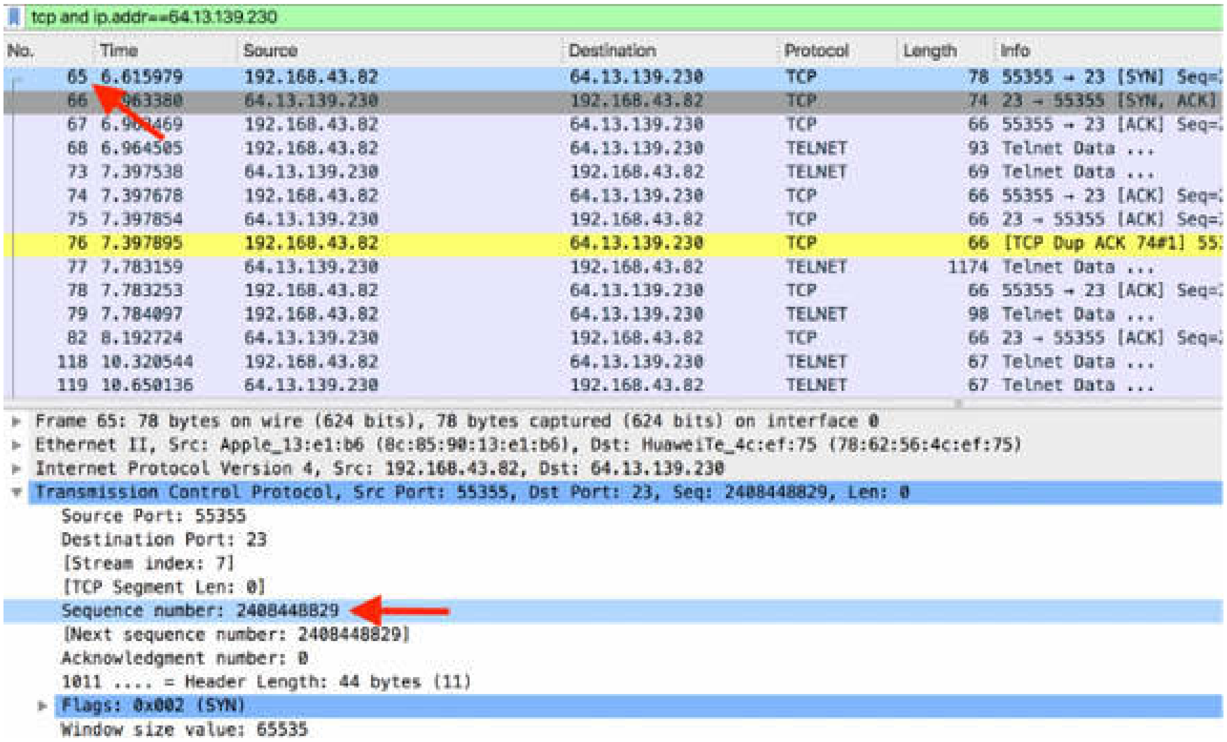
The exception to the sequence numbering equation, explained in the previous task, is that during the handshake and the teardown process, the sequence number increments by 1, even though a byte of data was not sent.

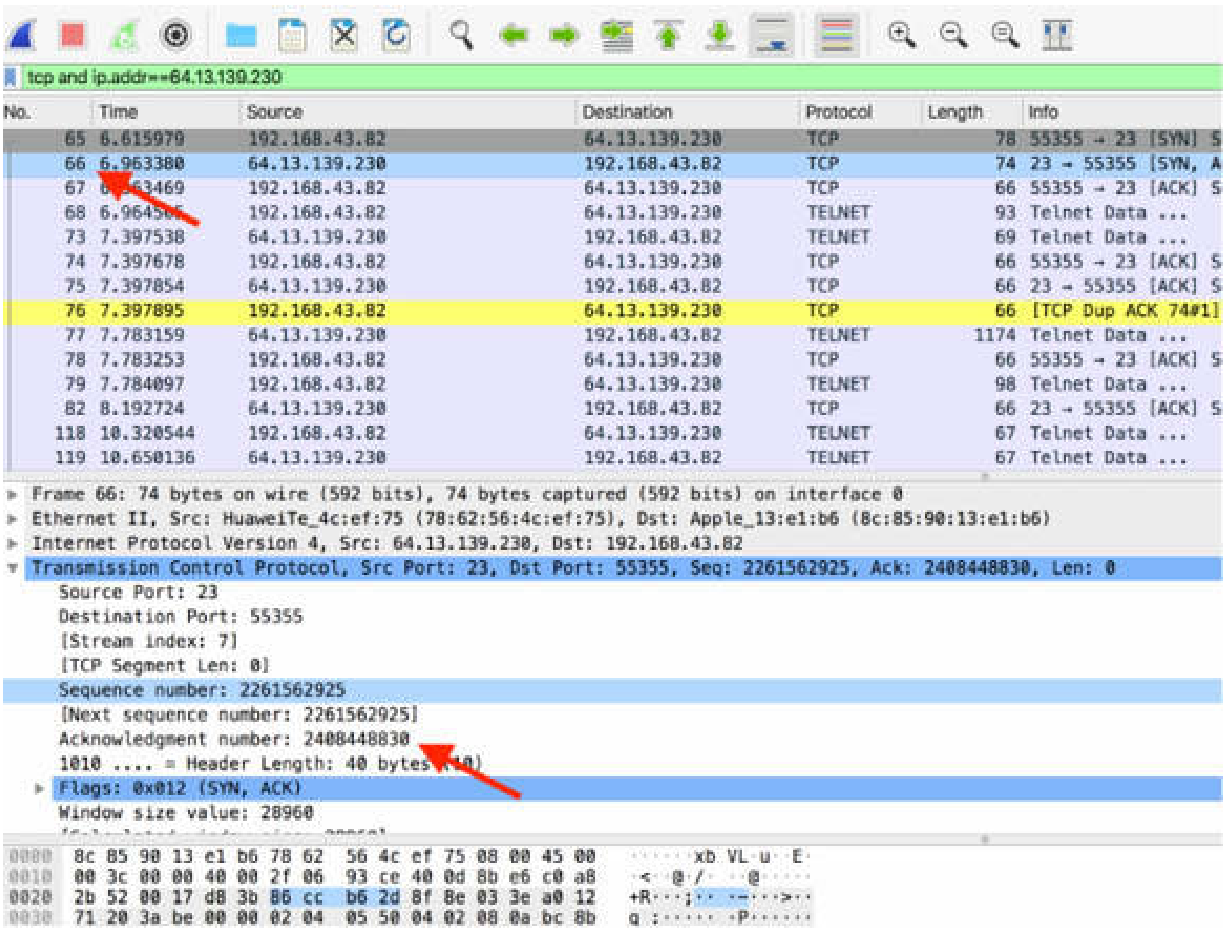
Moreover, by default, Wireshark uses Relative Sequence Numbering and for easier readability, it starts the sequence number value from 0, as you saw at the beginning of the communication.

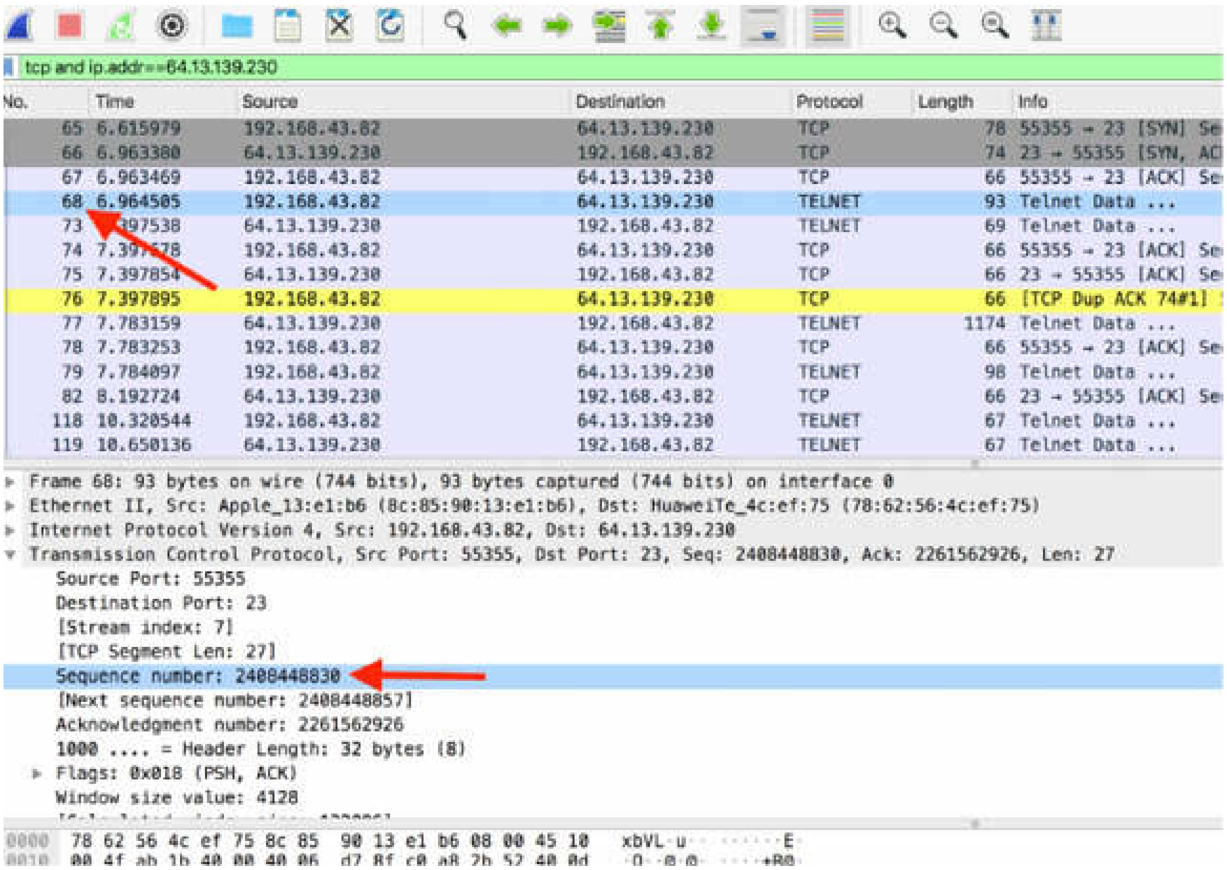
Open the Wireshark Preferences dialog box, shown in the figure below. In the left tree view, select TCP and clear the “Relative sequence number” check box.



Wireshark displays the absolute sequence numbers that are, in general, big numbers. However, when analyzing a capture, it is easier to work with smaller numbers. The figure below shows the analysis of the same sequence numbers that you examined in the previous step but with the absolute value enabled.


tcp and ip.addr==64.13.139.230
No.
Time
Source
Destination
Protocol
Length
Info
65 6.615979 192.168.43.82 64.13.139.230 TCP 78 55355 → 23 [SYN] Seq=
67 192.168.43.82 64.13.139.230 TCP 66 55355 → 23 [ACK] Seq=
68 6.964505 192.168.43.82 64.13.139.230 TELNET 93 Telnet Data ...
73 7.397538 64.13.139.230 192.168.43.82 TELNET 69 Telnet Data ...
74 7.397678 192.168.43.82 64.13.139.230 TCP 66 55355 → 23 [ACK] Seq=
75 7.397854 64.13.139.230 192.168.43.82 TCP 66 23 → 55355 [ACK] Seq=
76 7.397895 192.168.43.82 64.13.139.230 TCP 66 [TCP Dup ACK 74#1] 55
77 7.783159 64.13.139.230 192.168.43.82 TELNET 1174 Telnet Data ...
78 7.783253 192.168.43.82 64.13.139.230 TCP 66 55355 → 23 [ACK] Seq=
79 7.784097 192.168.43.82 64.13.139.230 TELNET 98 Telnet Data ...
82 8.192724 64.13.139.230 192.168.43.82 TCP 66 23 → 55355 [ACK] Seq=
118 10.320544 192.168.43.82 64.13.139.230 TELNET 67 Telnet Data ...
119 10.650136 64.13.139.230 192.168.43.82 TELNET 67 Telnet Data ...
Frame 65: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface 0
Ethernet II, Src: Apple_13:e1:b6 (8c:85:90:13:e1:b6), Dst: HuaweiTe_4c:ef:75 (78:62:56:4c:ef:75)
Internet Protocol Version 4, Src: 192.168.43.82, Dst: 64.13.139.230
Transmission Control Protocol, Src Port: 55355, Dst Port: 23, Seq: 2408448829, Len: 0
Source Port: 55355
Destination Port: 23
[Stream index: 7]
[TCP Segment Len: 0]
Sequence number: 2408448829
[Next sequence number: 2408448829]
Acknowledgment number: 0
1011 .... = Header Length: 44 bytes (11)
Flags: 0x002 (SYN)
Window size value: 65535


tcp and ip.addr==64.13.139.230
No. Time Source Destination Protocol Length Info
65 6.615979 192.168.43.82 64.13.139.230 TCP 78 55355 → 23 [SYN] S
66 6.963380 64.13.139.230 192.168.43.82 TCP 74 23 → 55355 [SYN, A
67 192.168.43.82 64.13.139.230 TCP 66 55355 → 23 [ACK] S
68 192.168.43.82 64.13.139.230 TELNET 93 Telnet Data ...
73 7.397538 64.13.139.230 192.168.43.82 TELNET 69 Telnet Data ...
74 7.397678 192.168.43.82 64.13.139.230 TCP 66 55355 → 23 [ACK] S
75 7.397854 64.13.139.230 192.168.43.82 TCP 66 23 → 55355 [ACK] S
76 7.397895 192.168.43.82 64.13.139.230 TCP 66 [TCP Dup ACK 74#1]
77 7.783159 64.13.139.230 192.168.43.82 TELNET 1174 Telnet Data ...
78 7.783253 192.168.43.82 64.13.139.230 TCP 66 55355 → 23 [ACK] S
79 7.784097 192.168.43.82 64.13.139.230 TELNET 98 Telnet Data ...
82 8.192724 64.13.139.230 192.168.43.82 TCP 66 23 → 55355 [ACK] S
118 10.320544 192.168.43.82 64.13.139.230 TELNET 67 Telnet Data ...
119 10.650136 64.13.139.230 192.168.43.82 TELNET 67 Telnet Data ...
Frame 66: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
Ethernet II, Src: HuaweiTe_4c:ef:75 (78:62:56:4c:ef:75), Dst: Apple_13:e1:b6 (8c:85:90:13:e1:b6)
Internet Protocol Version 4, Src: 64.13.139.230, Dst: 192.168.43.82
Transmission Control Protocol, Src Port: 23, Dst Port: 55355, Seq: 2261562925, Ack: 2408448830, Len: 0
Source Port: 23
Destination Port: 55355
[Stream index: 7]
[TCP Segment Len: 0]
Sequence number: 2261562925
[Next sequence number: 2261562925]
Acknowledgment number: 2408448830
1010 .... = Header Length: 40 bytes
Flags: 0x012 (SYN, ACK)
Window size value: 28960
0000 8c 85 90 13 e1 b6 78 62 56 4c ef 75 08 00 45 00
0010 00 3c 00 00 40 00 2f 06 93 ce 40 0d 8b e6 c0 a8
0020 2b 52 00 17 d8 3b 86 cc b6 2d 8f 8e 03 3e a0 12
0030 71 20 3a be 00 00 02 04 05 50 04 02 08 0a bc 8b



Notes:

Repeat the previous steps, creating a new TCP communication between a client and a server. Verify the sequence numbers for a couple of messages exchanged to gain more confidence in understanding the sequence rules.

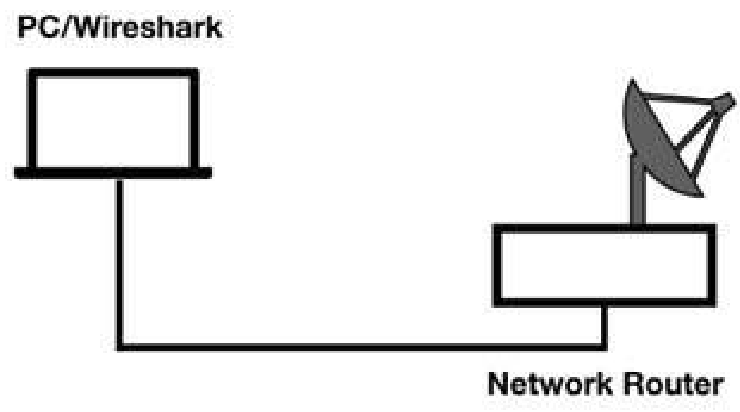
Lab 49. TCP Packet Loss

Lab Objective:
Learn how TCP recovers from packet loss.

Lab Purpose:
Understand the features of TCP regarding the management of the packets lost during communication.

Lab Tool:
Wireshark installed on a PC, Ethernet switch/router (cable/Wi-Fi)., and a second PC.

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet. Another PC (shown as PC Target in the figure below) is connected to the same network router.



Lab Walkthrough:

Task 1:

On PC Wireshark, open Wireshark, and start capturing the traffic on the interface connected to the network router. In the filter toolbar, enter tcp to show only TCP packets in the Packet List pane.

On PC Target (the server), open a terminal window and run the command nc –l 2390 to enable a TCP server in the listen mode on port 2390.

On PC Wireshark (the client), open a terminal window and run the Linux command nc IP-of-PC-Target 2390 to connect to the target server. Replace IP-of-PC-Target with the actual IP address of PC Target. In this example, the IP of PC target is 192.168.1.105. When the connection has been established, send some string commands from the client and see the echo back on the server.

The figures below show two command windows displaying the command used.

In the client terminal window, do the following:

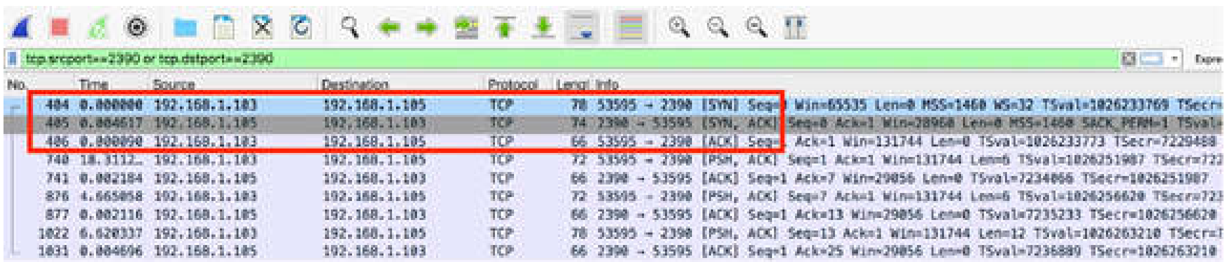
1. Type "Hello" and press Return.
2. Type "World" and press Return.
3. Type "how are you" and press Return.

The Echo server displays the same words on the server.

Stop the capture and save the file.

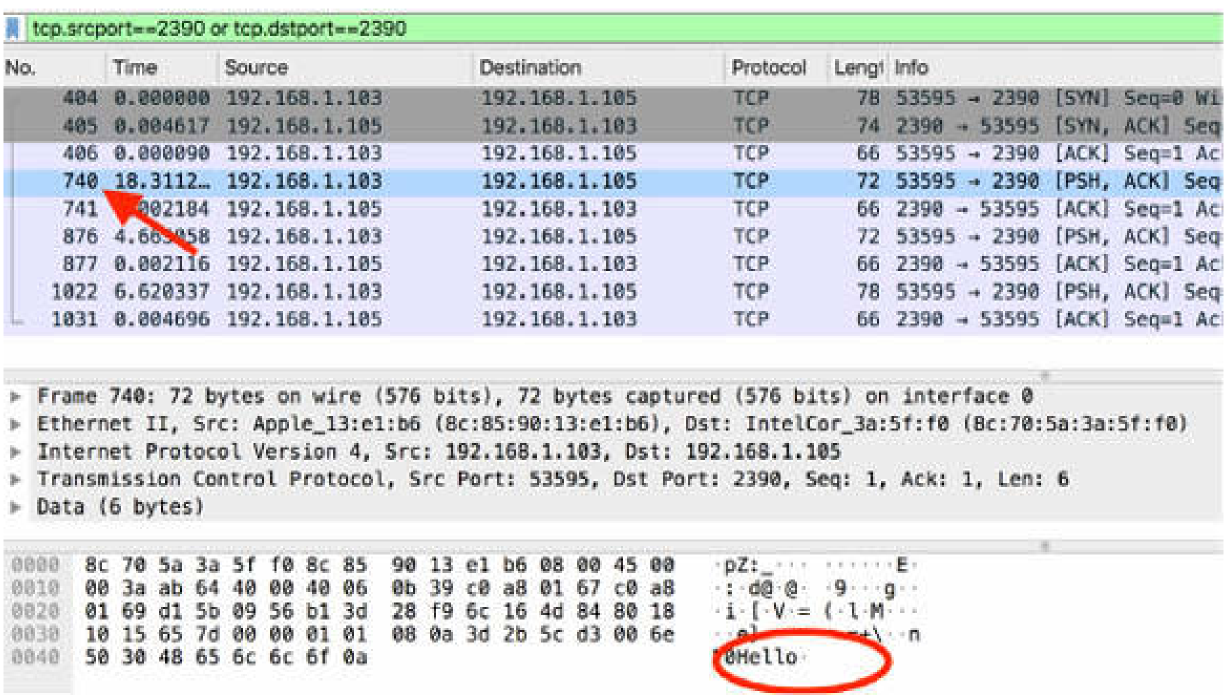
Task 2:
Open the file saved in the previous step. In the filter toolbar, enter tcp.srcport==2390 or tcp.dstport==2390 to display only the TCP packets of interest in the Packet List pane.

As shown in the figure below, the communication starts after the correct initial handshaking (packets #404, #405, and #406).

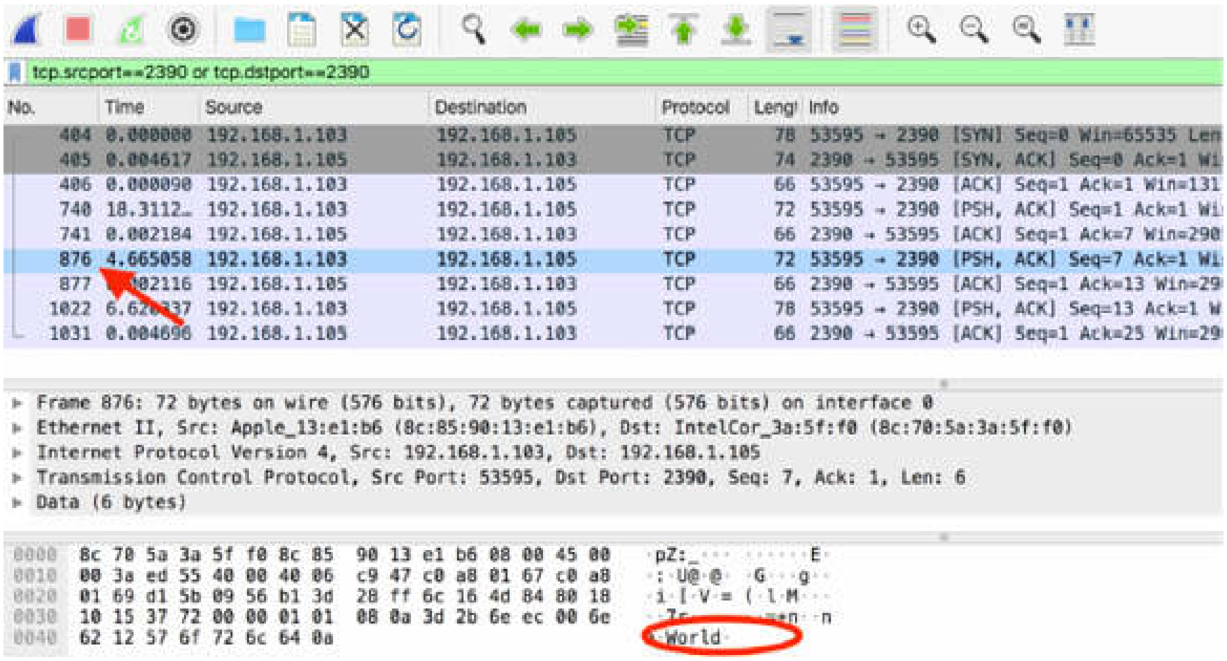


The following packets contain the data buffer sent from the client to the server:

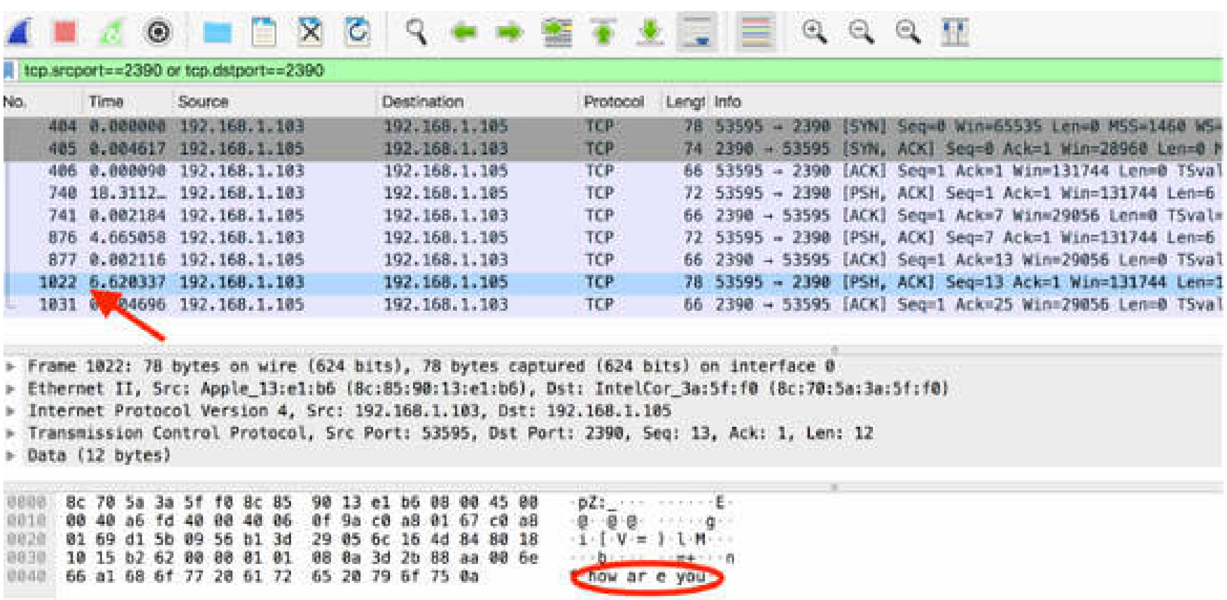
The first data packet:



The second data packet:



The third data packet:



Task 3:

Start a packet capture on the active network interface; for example, wlan0. On PC Wireshark, open a terminal window and run the command tc qdisc add dev wlan0 root netem loss 30%.

This command enables a rule to drop 30% of the packets related to the wlan0 physical interface.

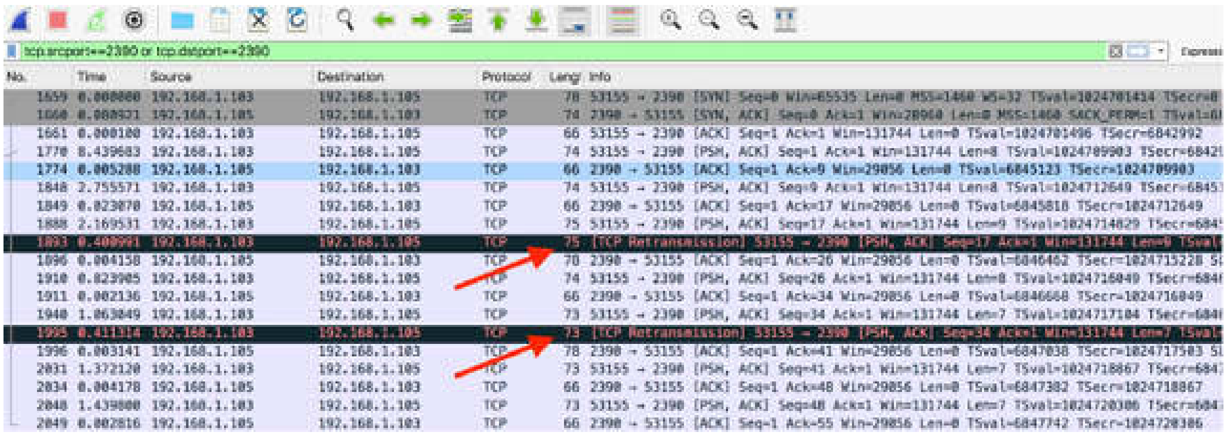
Verify the application of the previous command with the command tc -p qdisc ls dev wlan0. The following output is displayed, confirming that the rule is applied:

qdisc netem 8003: root refcnt 5 limit 1000 loss 30%

Recreate the TCP communication channel between PC Wireshark and PC Target, as done previously in Task 1. On PC Wireshark, in the terminal window, run the command nc 192.168.1.105 2390 . On PC Target, in the terminal window, run the command nc –l 2390 .

Finally, send an echo command from PC Wireshark. Stop the capture and save the file.

When you inspect the result of the echo commands, you will not see any difference between the steps above. For each issued command, you will see the related echo. However, when you take a look at the Wireshark capture and use the tcp.srcport==2390 or tcp.dstport==2390 display filter, you will notice something interesting.

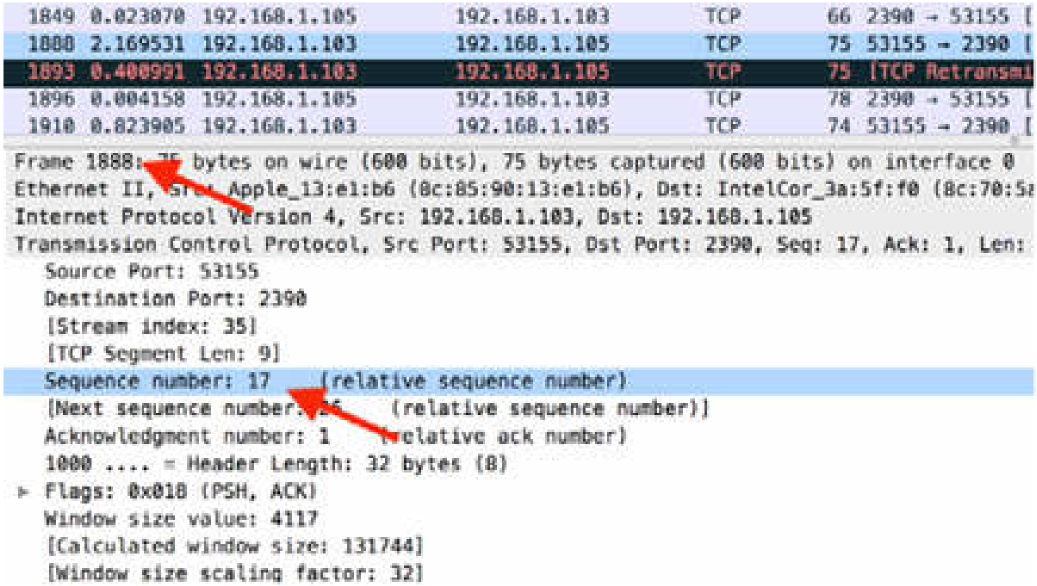


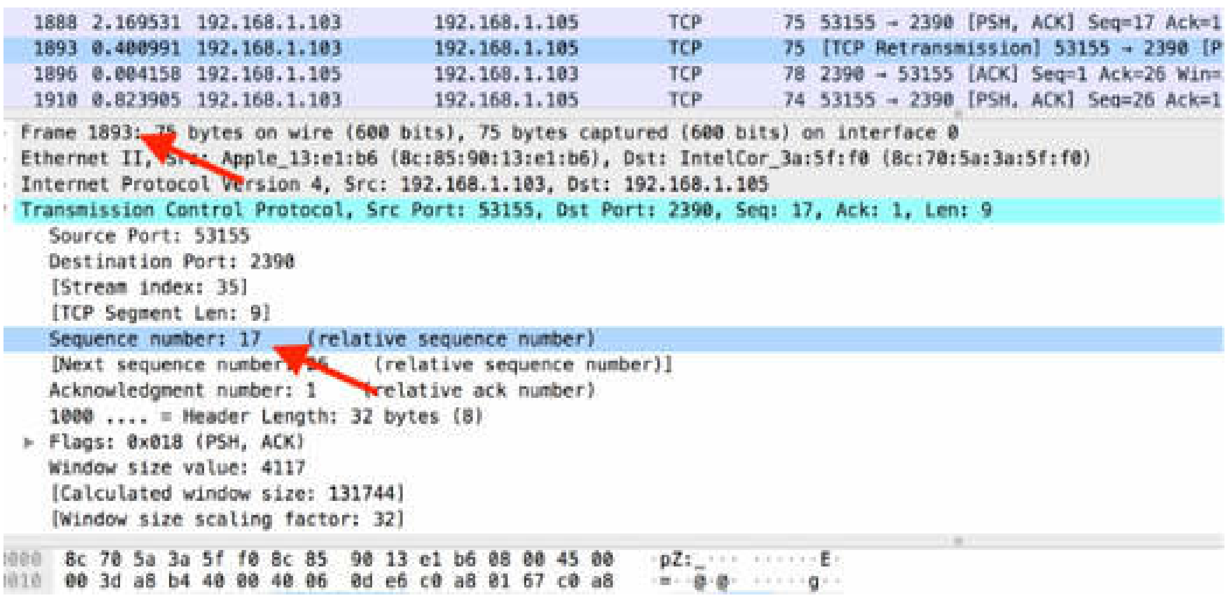
In the figure above, the following two packets are clearly identified and marked as TCP Retransmission: packet #1893 and packet #1995.

Because regular TCP packets #1888 and #1940 have not received the TCP ACK, because of the drop-by rule that you set in the previous task, the TCP layer retransmits the missing ACK packets after a time delay of about 0.4s (which is equal to the TCP Retransmission Timeout—RTO).

The general rule is that when a data packet is sent and not acknowledged before the RTO timer expires, the TCP sender can retransmit the packet using the sequence number of the original packet.

If you look at the TCP sequence number, you can see that the sequence number of the retransmitted packet is the same as the original packet. The figures below show the two consecutive TCP packets: the original one and the retransmitted one.





Notes:

Repeat the previous steps to create a new TCP communication between a client and a server and verify what happens in case of more packets dropping.

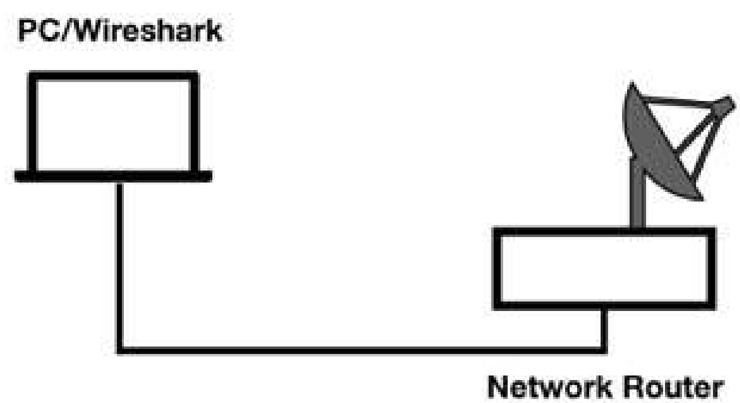
Lab 50. TCP Problems

Lab Objective:
Learn about the more common TCP problems.

Lab Purpose:
Learn how to detect and analyze the more common TCP problems.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

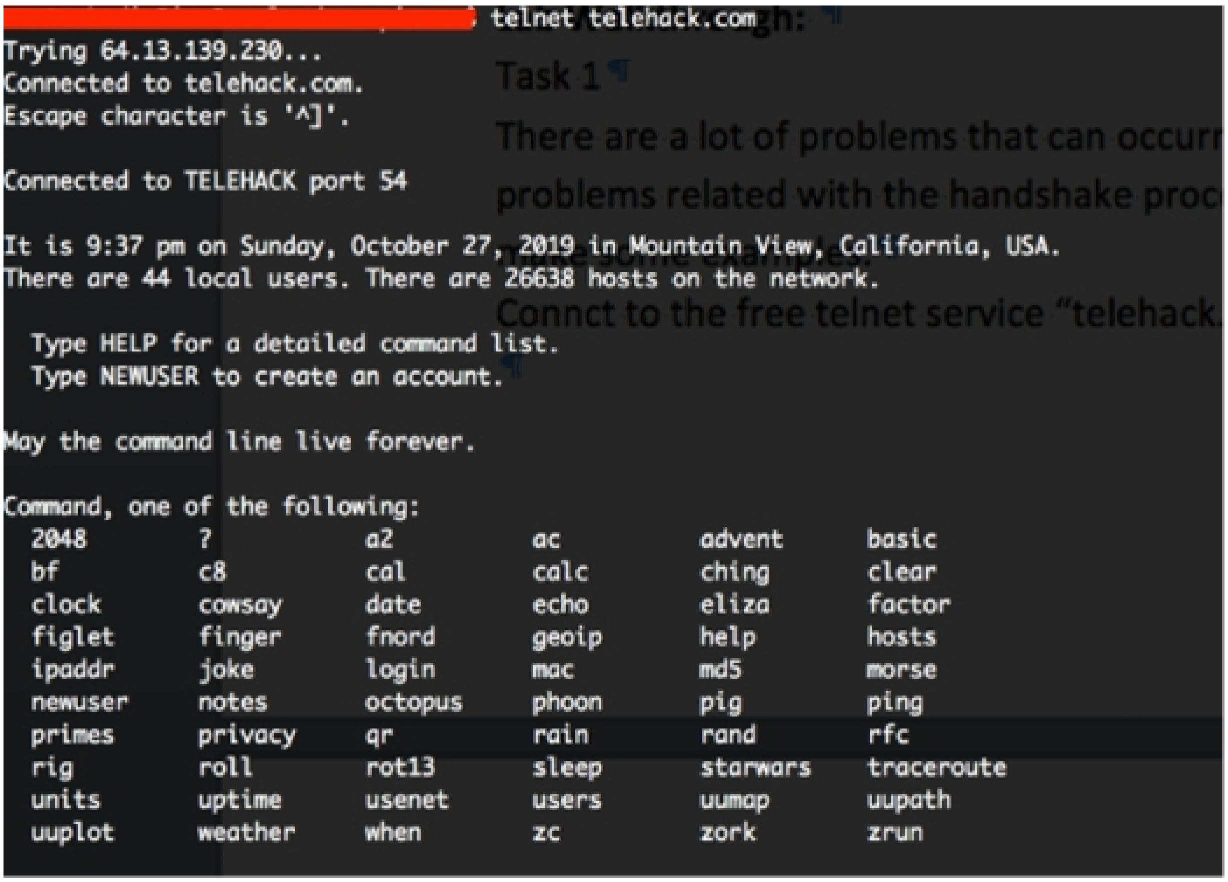


Lab Walkthrough:

Task 1:
In a communication based on the TCP layer, a lot of problems can occur related to the handshake process, packet loss, TCP disconnect, frozen

windows, and so on.

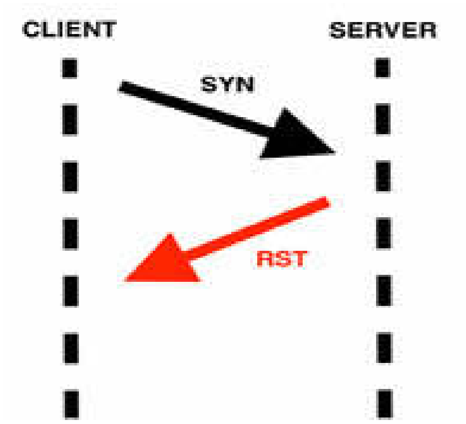
Open Wireshark, and capture the traffic for a few minutes. Open a terminal window, and run the command telnet telehack.com . A connection with the FTP server telehack.com is opened, as displayed in the figure below. Stop the capture.



In Wireshark, in the filter toolbar, enter tcp and ip.addr == 64.13.139.230 to display only the packets exchanged between the Telnet client and server

The regular TCP communication is initiated by the classic handshake exchange.

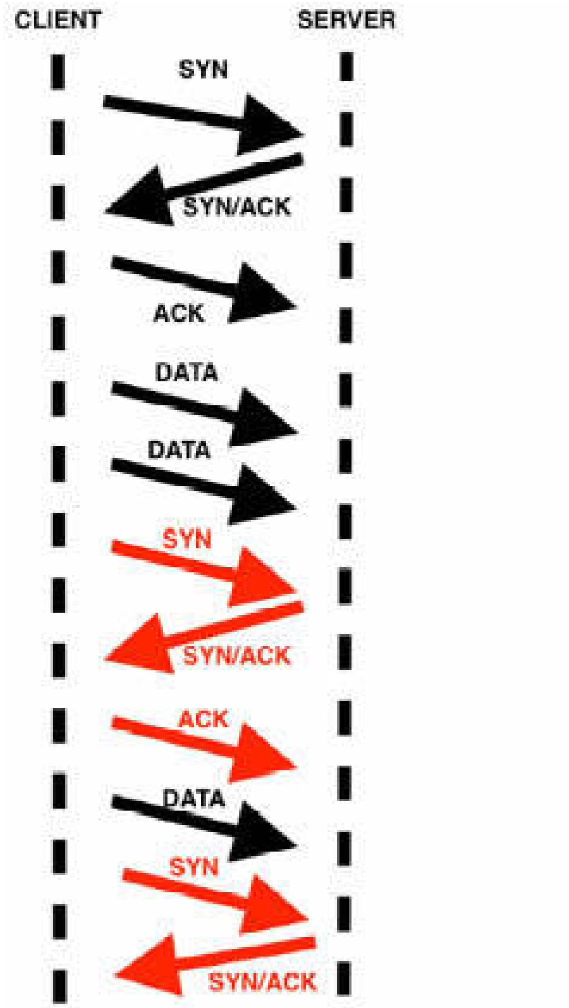
One possible TCP issue is of TCP connection refusal. The initial packet of the handshake (SYN) receives a Reset (RST/ACK) response. As a result, the connection cannot be established. If the handshake process does not complete successfully, no data is exchanged between the hosts.



If a lot of failed TCP connections are present in a network capture, it could be an indication of a TCP scan in progress.

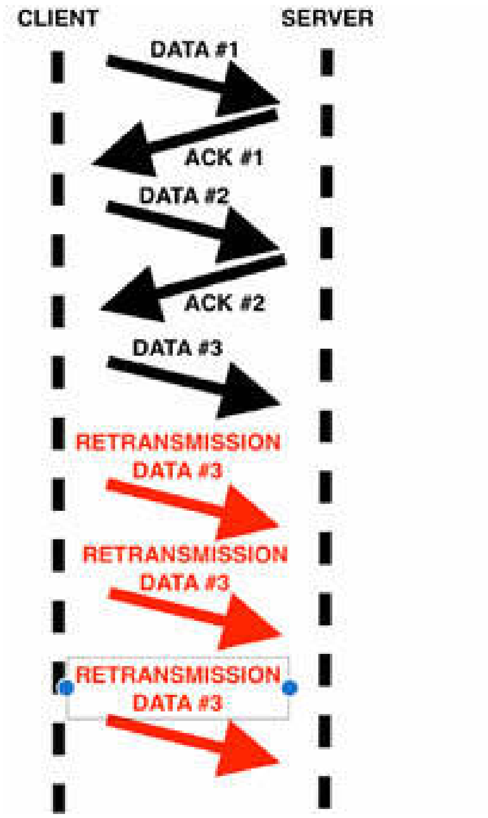
Task 2:
Another possible TCP issue is when during normal communication, some frequent handshake processes are repeated unexpectedly. After seeing the TCP handshake, if you see a rogue SYN/ACK, there is a problem with the connection. The connection is considered invalid, and it must be torn down and recreated.

Feel free to download the color images for all labs from **www.101labs.net** resources page.



In the figure above, after a successful handshake process (SYN—SYN/ACK—ACK) and some DATA exchange between the client and the server, the handshake appears again. This could be because the connection is not stable in terms of physical stability.

Task 3:
Another possible TCP issue is when there is a lot of TCP retransmissions on the inspected captured frames.



In the figure above, a situation during the normal data exchange is shown. For each exchanged data packet, the side receiving the DATA sends an ACK message to notify the reception. At a certain point, no ACK message is sent, and the sender retransmits the DATA (packet retransmission).

If this situation persists, the problem does not get resolved because of the non-synchronization of the data exchange. The application must be restarted to create a new connection.

Notes:
Repeat network acquisition on a TCP network to repeat the analysis presented in the previous tasks and gain the necessary confidence in being able to identify typical TCP issues.

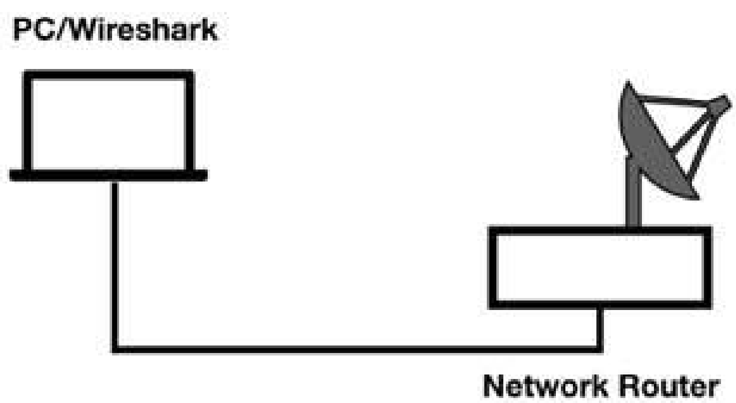
Lab 51. TCP Packet Structure

Lab Objective:
Learn about the TCP packet structure.

Lab Purpose:
Learn about the structure and various fields of a TCP packet.

Lab Tool:
Wireshark installed on a PC, Ethernet switch/router (cable/Wi-Fi)., and a second PC.

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet. Another PC (shown as PC Target in the figure below) is connected to the same network router.

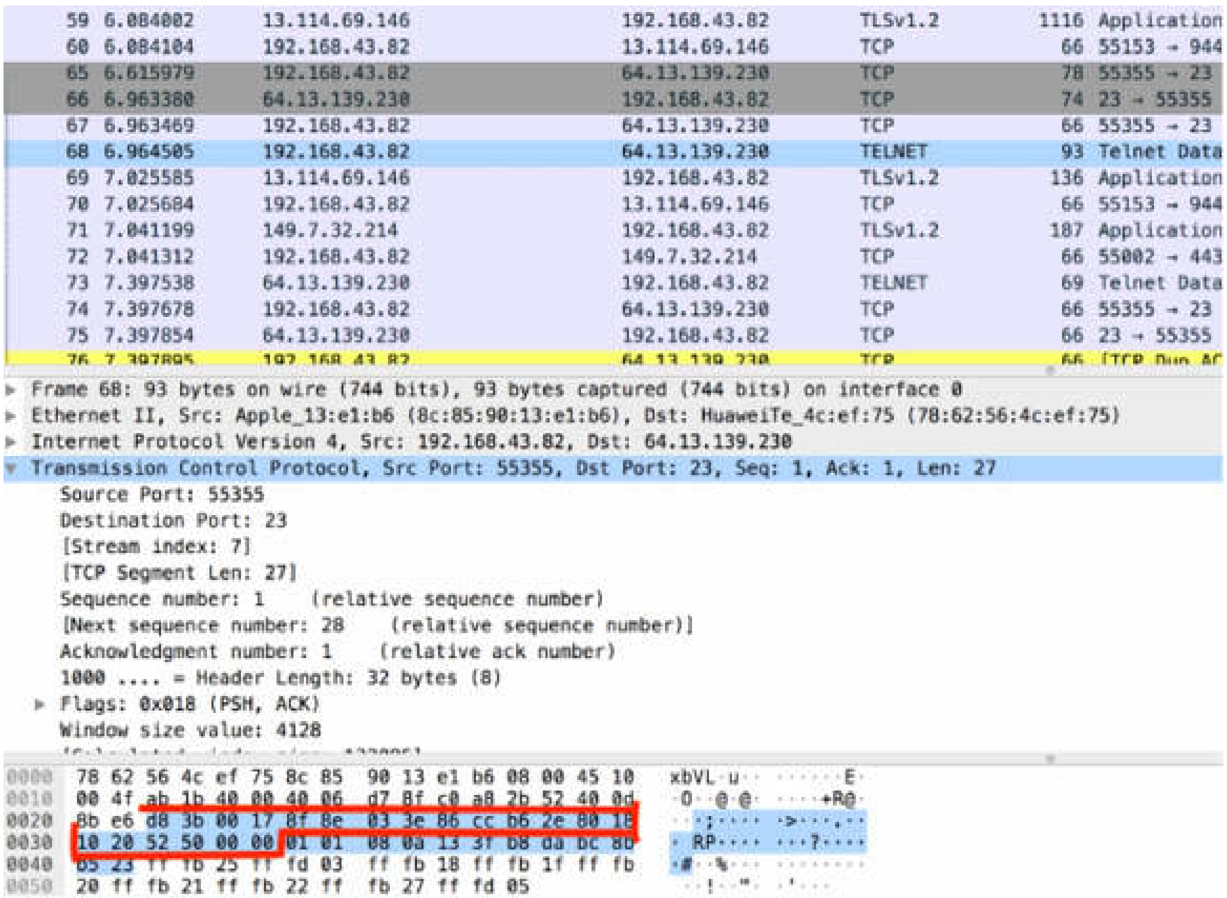


Lab Walkthrough:

Task 1:

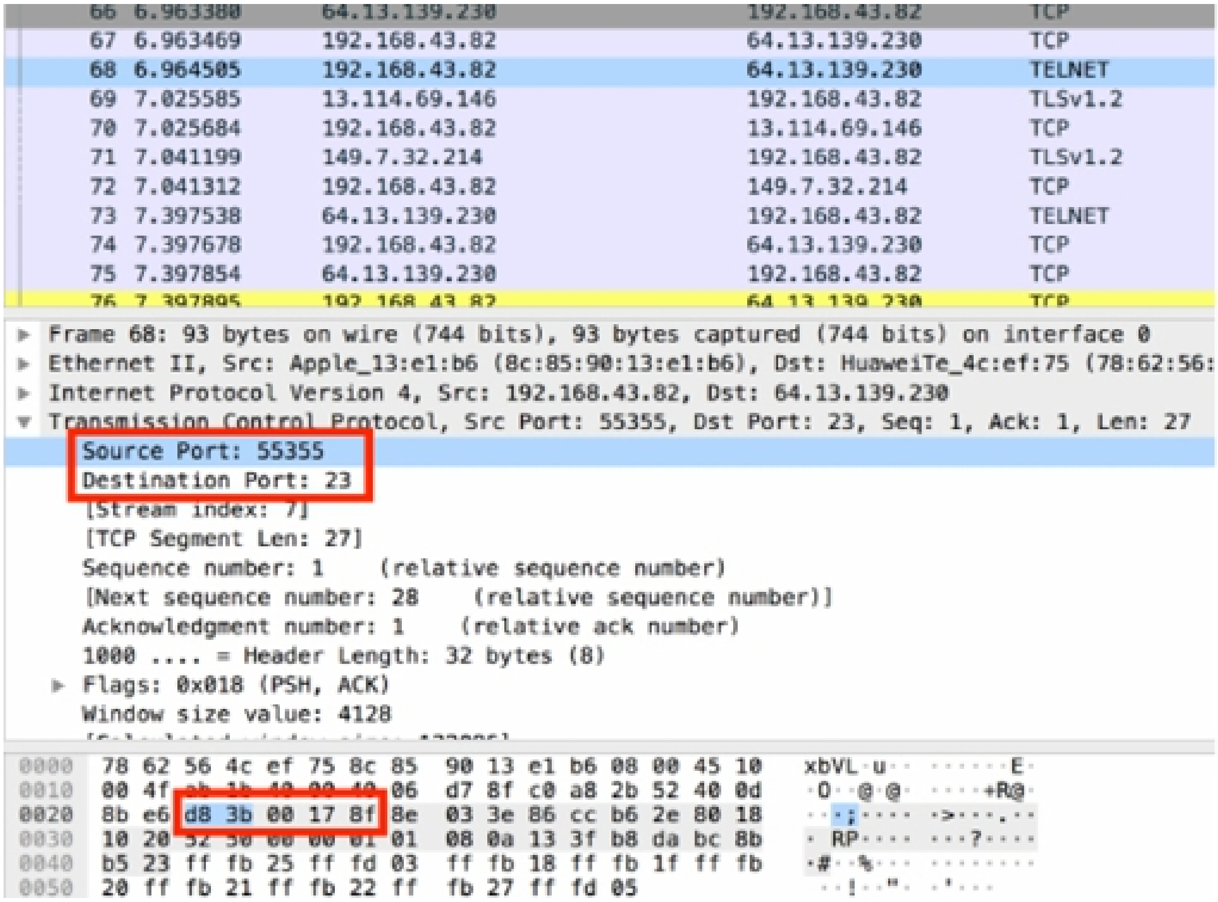
Open a packet capture saved in one of the previous labs. In the filter toolbar, enter tcp and select a TCP packet in the Packet List pane. Telnet captures are most suitable for this lab.

In the Packet Details pane, click the “Transmission Control Protocol” field to open the tree view.



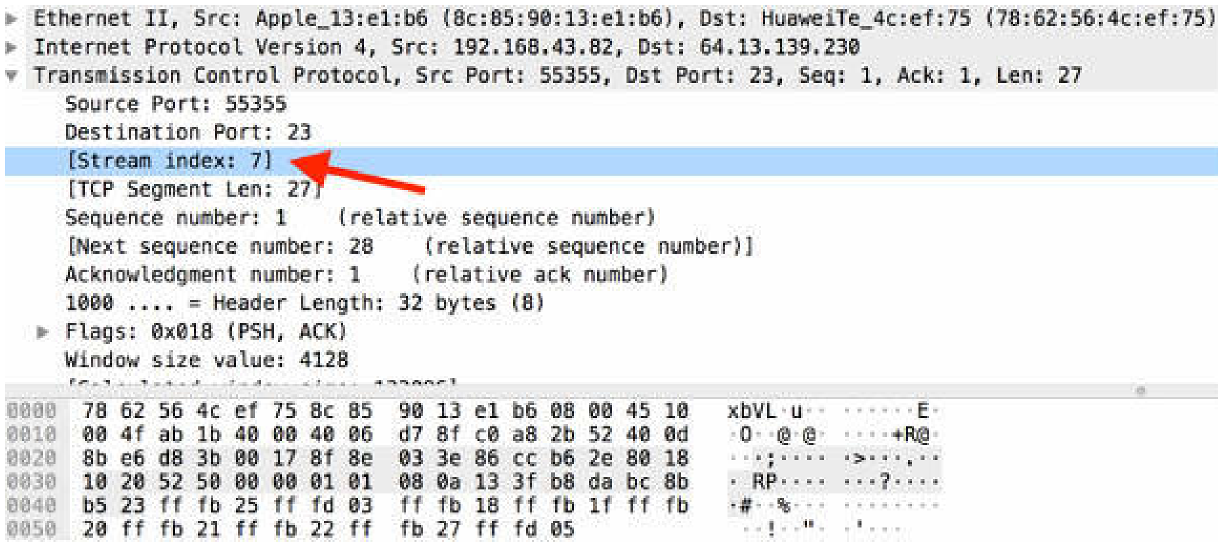
If you click on the IPv4 field, you will see that the IP header is 20 bytes long. However, the TCP header supports an Options field that can extend the header length. If you don’t see a TCP option in the handshake packet(s), the option wasn’t there when Wireshark saw the packet.

The Source Port and the Destination Port are the first two fields. The source port is the listening port open at the sender; the destination port is the target port open at the receiver.



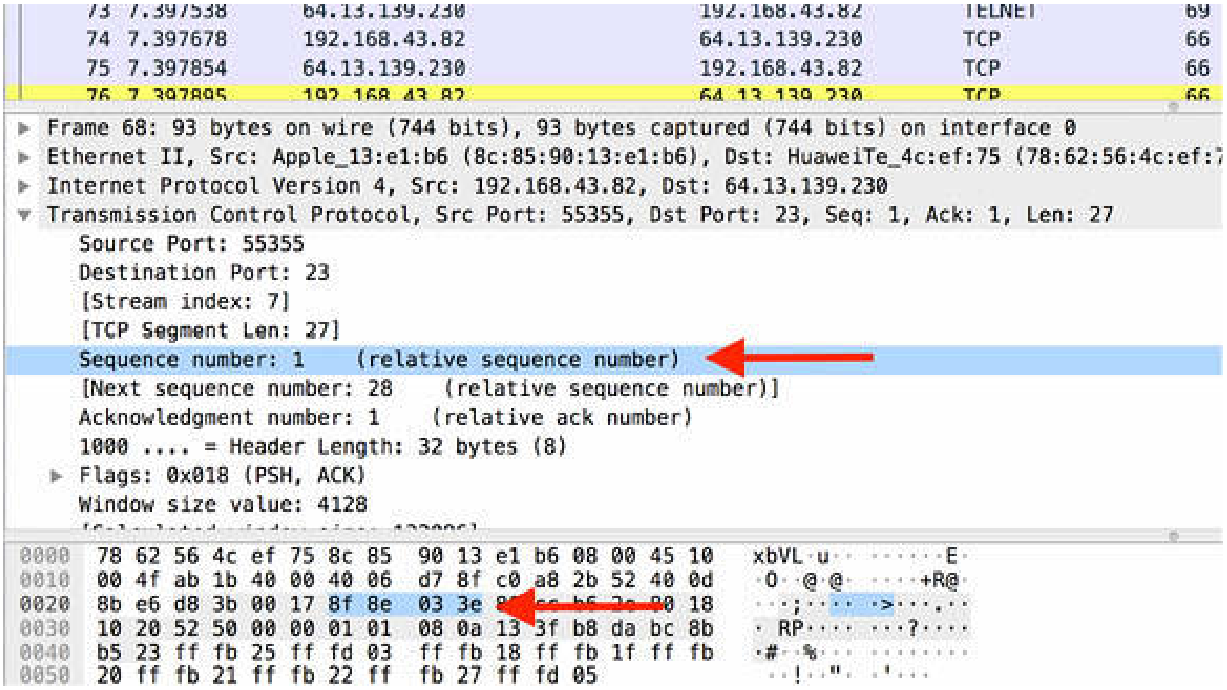
Task 2:

The “Stream index” field is the next field. It is not an actual field in the TCP header, as shown in the figure below. The “Stream index” value is defined by Wireshark, and it can be used to quickly filter a TCP conversation. The “Stream index” value in TCP conversations begins at 0 and increases by 1 for each TCP conversation seen in the trace file.



Task 3:

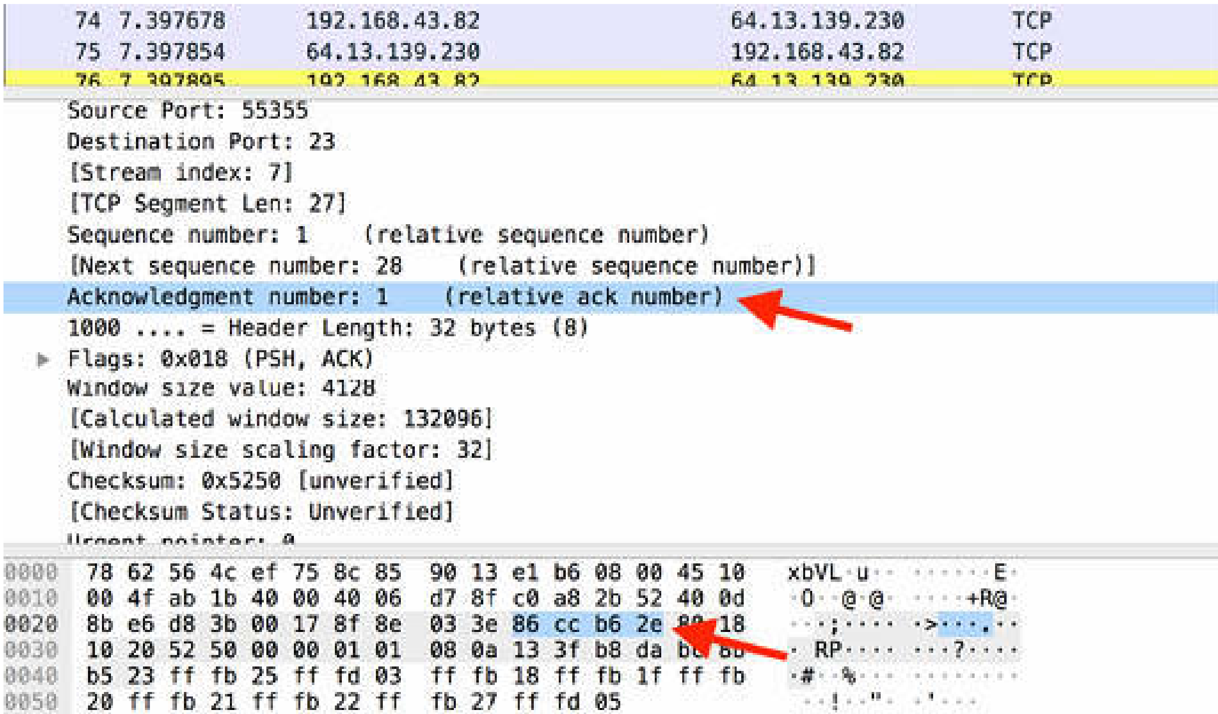
The “Sequence number” field is the next field. It contains a number that uniquely identifies the TCP segment (the data that follows the TCP header is referred to as a TCP ‘segment’). This sequence number provides an identifier for the TCP segment and enables receivers to determine when parts of a communication stream are missing. The sequence number increments by the number of data bytes contained in each packet. It is important to remember that each TCP device assigns its own Initial Sequence Number (ISN).



Task 4:

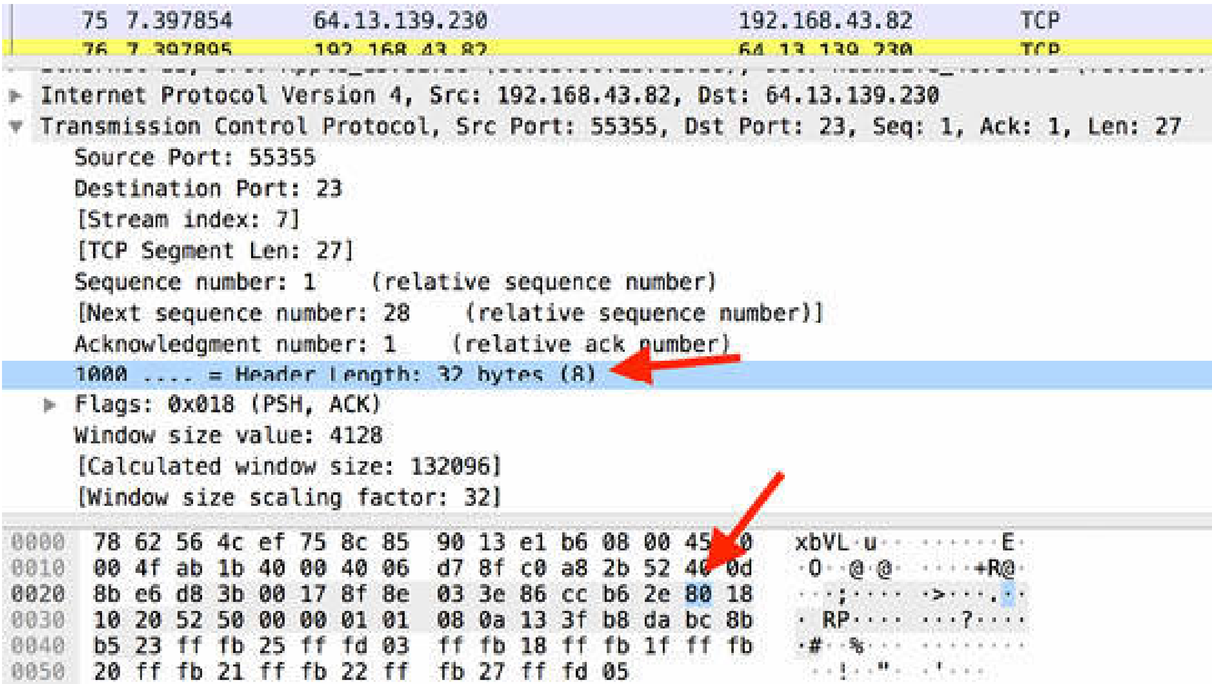
The “Sequence number” field is followed by another Wireshark field called “Next sequence number”. It is calculated as the sum of data length and current packet SN. This field appears only in the packets containing DATA; it does not appear in simple SYN or ACK messages.

The “Acknowledgment number” field is the next field, as displayed in the figure below. The “Acknowledgment number” field indicates the next expected sequence number from the other side of the communication.



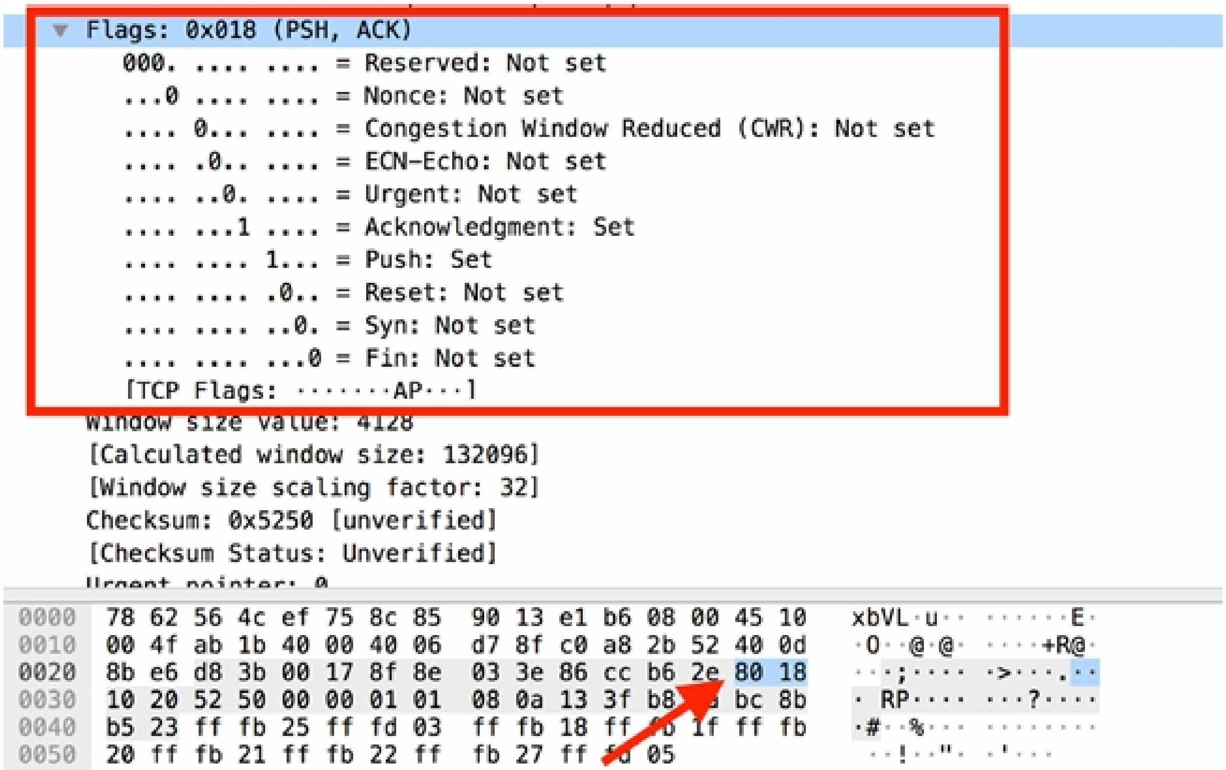
Task 5:

The “Data Offset” field is the next field. It defines the length of the TCP header in 4-byte increments, so the value (8), displayed in the figure below, indicates that the TCP header is 32 bytes long. This field is required because the TCP header length can vary depending on the TCP header options used. The TCP Options field is often used during the TCP connection setup to establish the Maximum Segment Size (MSS).



Task 6:

The Flags field is the next field. You can click this field to open the tree view, as shown in the figure below.

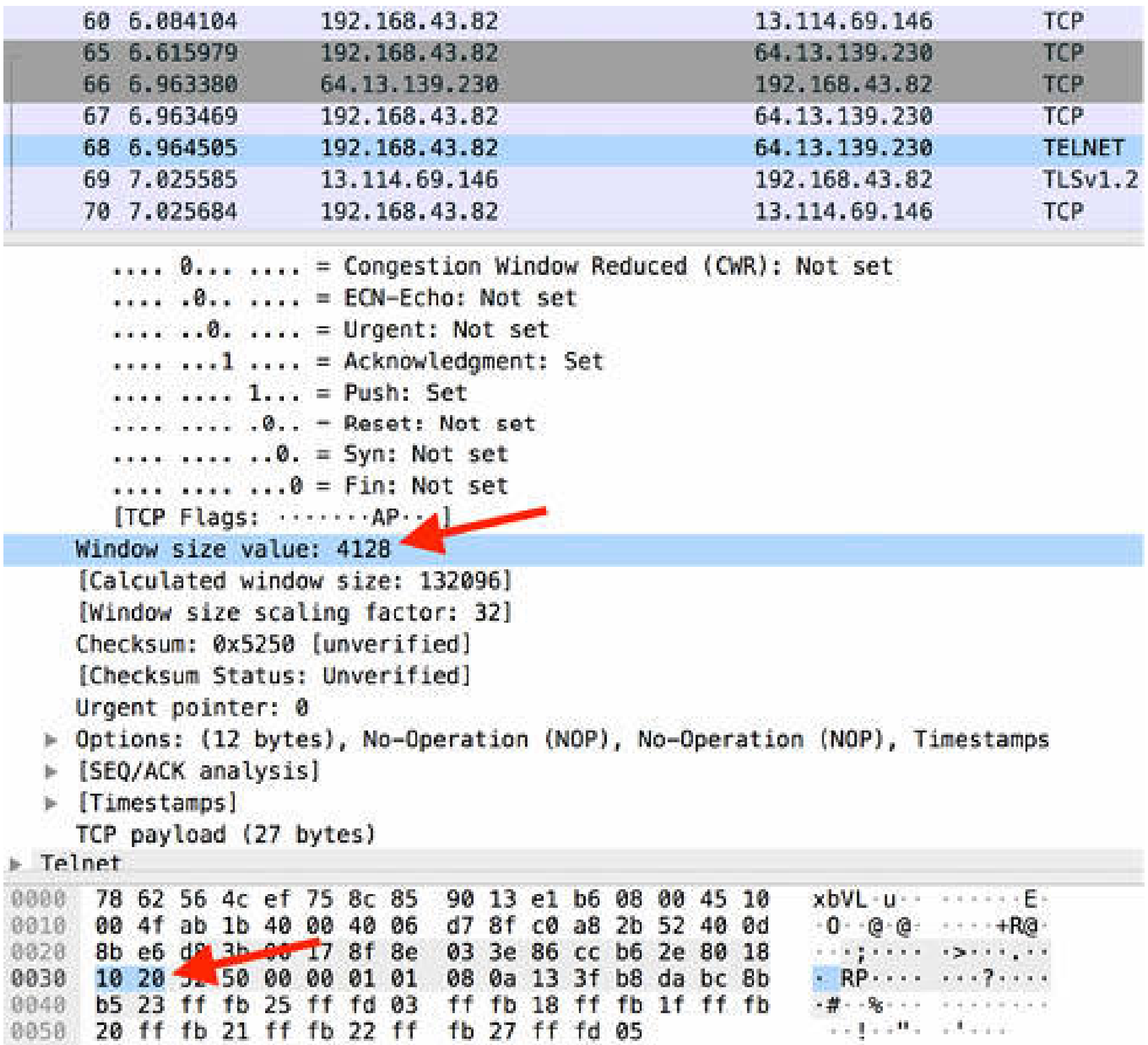


The following list describes available settings for the Flags field:

- Reserved: These three bits are set to zero.
- Nonce: The Nonce field works in conjunction with the ECN fields in the IP header.
- Congestion Window Reduced (CWR): The CWR flag is set by a data sender to inform the data receiver that the congestion window has been reduced.
- URG (Urgent): Indicates that the Urgent Pointer field should be examined. The Urgent Pointer field resides after the TCP Checksum field and is set to 0x0000 when not used. The Urgent Pointer field is processed only if this bit is set.
- ACK (Acknowledgment): Acknowledgment packet
- PSH (Push): Bypass buffering and pass data directly onto the network —do not buffer incoming data and pass it directly to the application
- RST (Reset): Close the connection explicitly

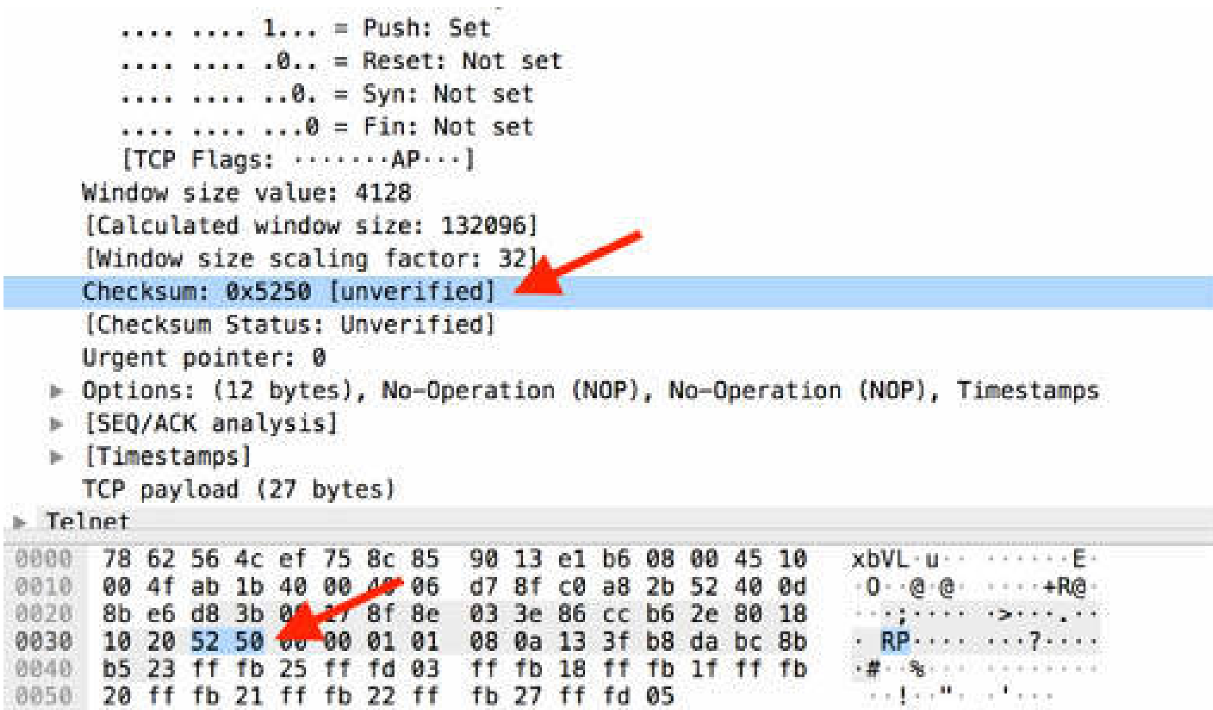
- SYN (Synchronize): Synchronize sequence numbers—used in the handshake process
- FIN (Finish): Transaction finished, but don't close the connection explicitly

The "Window size value" field indicates the size of the TCP receiver buffer in bytes. A window size of 0 indicates that the receiver has no buffer space available. The maximum value that can be denoted in this two-byte field is 65,535.



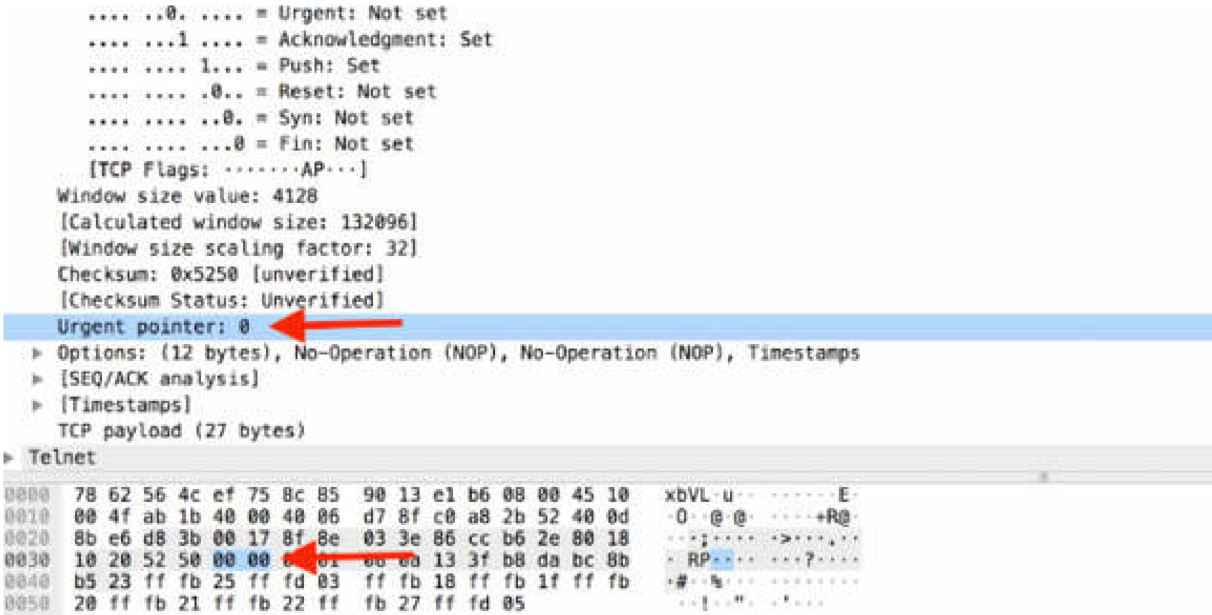
Task 7:

The Checksum field is the next field. The TCP checksum is performed on the contents of the TCP header and data (not including the data link padding).



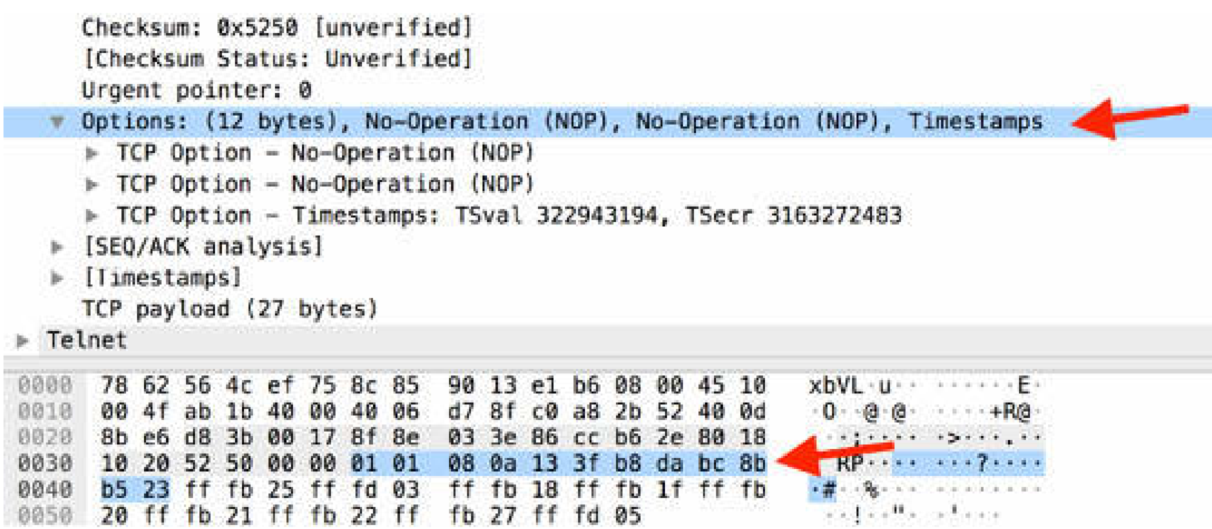
Task 8:

The "Urgent pointer" field is the next field. This field is relevant only when the URG bit is set. If the URG bit is set, the receiver must examine this field to see where to look/read first in the packet.



Task 9:

The Options field is the last field in the TCP packet. It is an optional field.



Notes:

Analyze different TCP messages to identify each field belonging to the packet to gain the necessary confidence in the dissecting a TCP message.

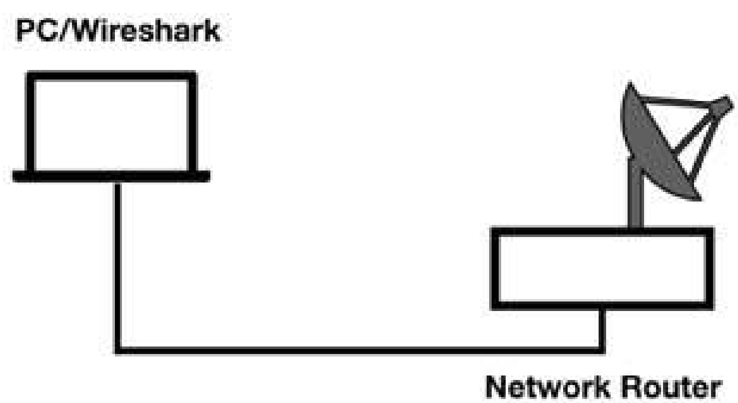
Lab 52. TCP Filtering

Lab Objective:
Learn how to use filters for TCP packets.

Lab Purpose:
Learn the filter syntax for TCP protocol.

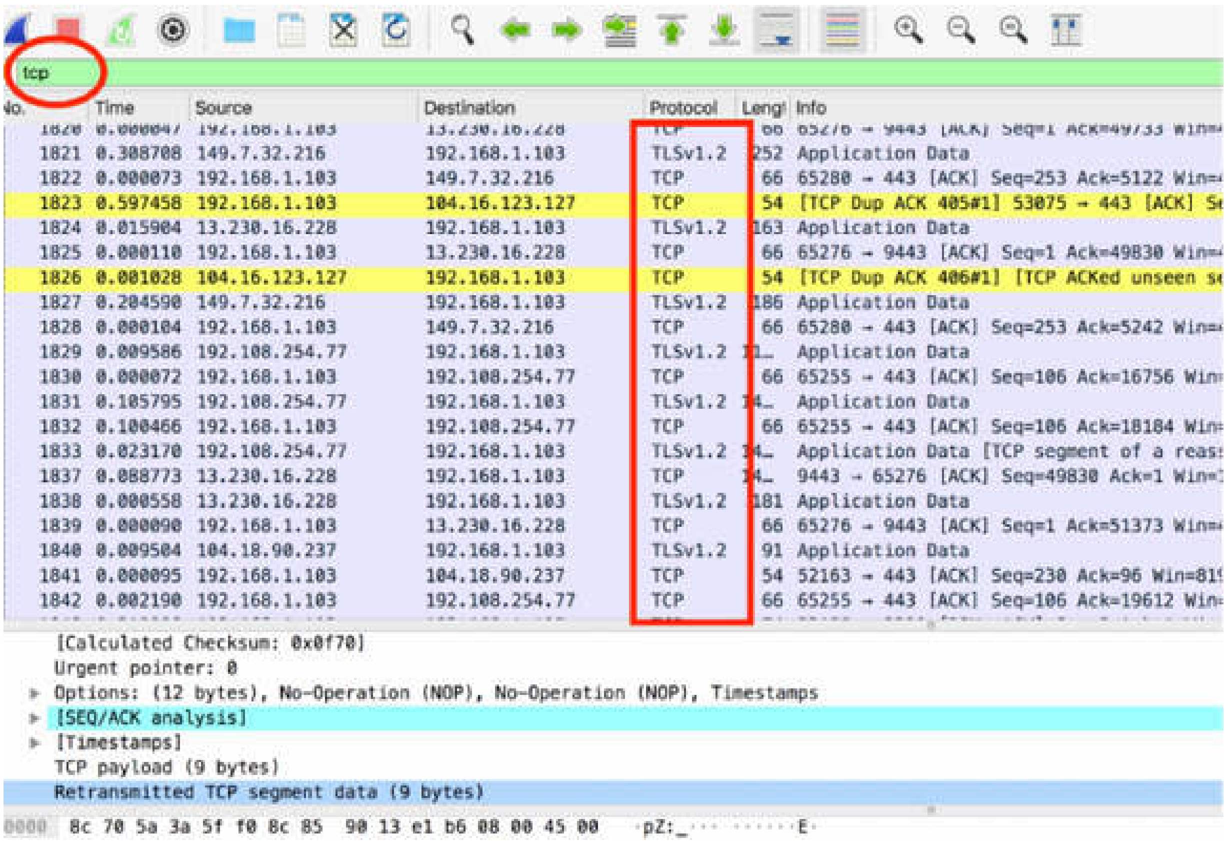
Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

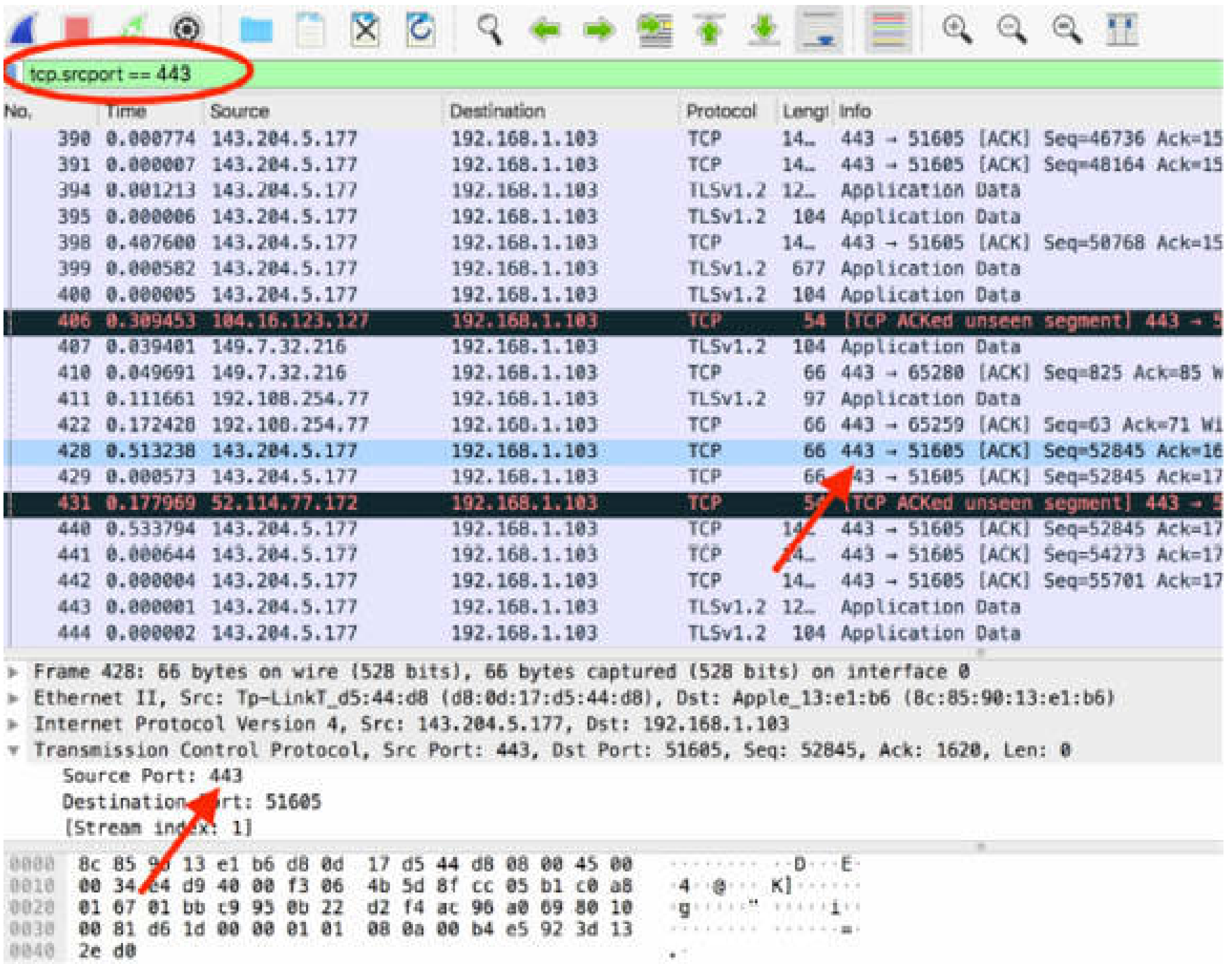
Task 1:
Open a packet capture saved in one of the previous labs. In the filter toolbar, enter tcp to display only TCP packets.



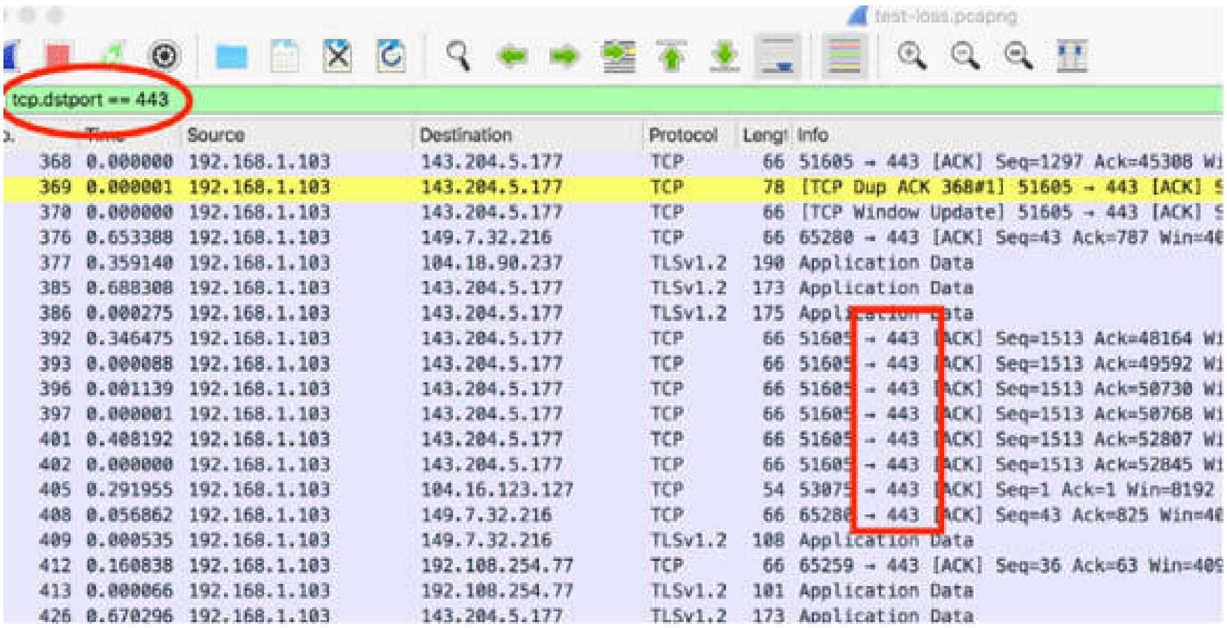
Task 2:

In the filter toolbar, enter tcp.srcport == 443 or tcp.dstport == 443 to filter packets based on the TCP source port or the TCP destination port.

The results of the filter with TCP source port are displayed in the figure below.

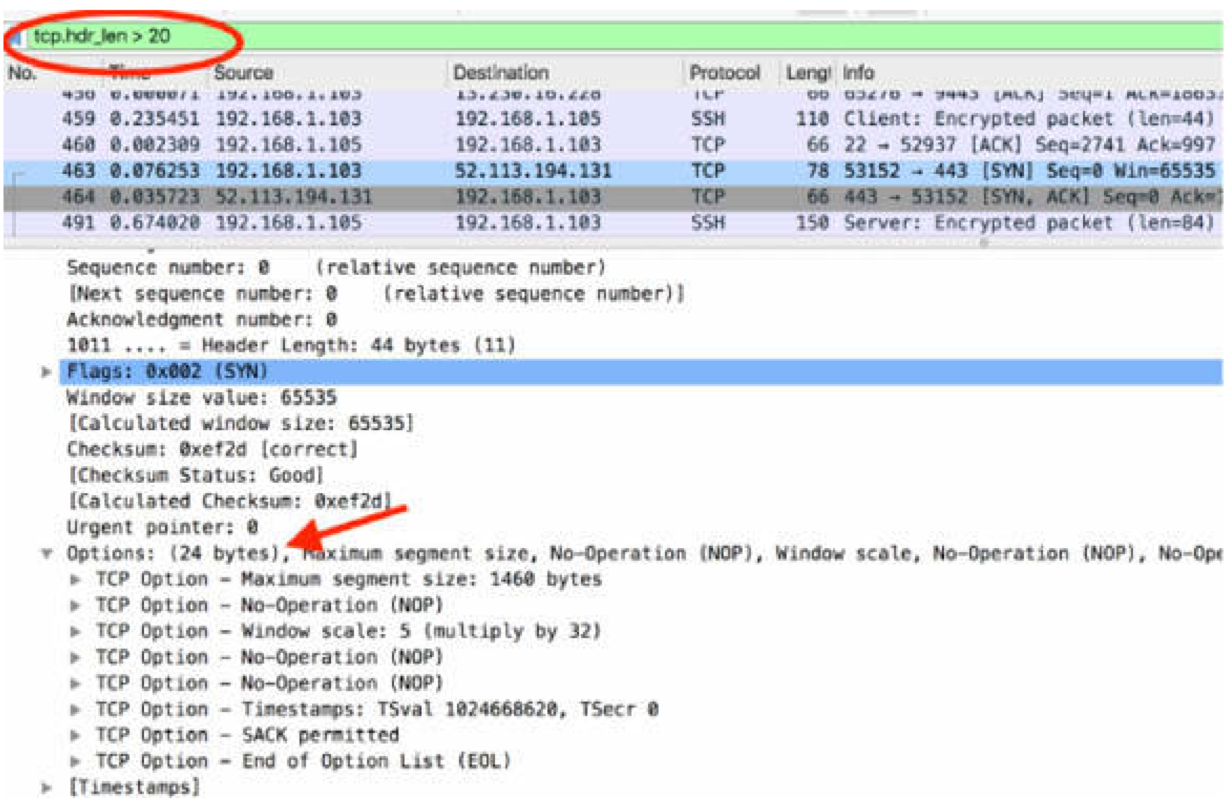


The results of the filter with the TCP destination port are displayed in the figure below.



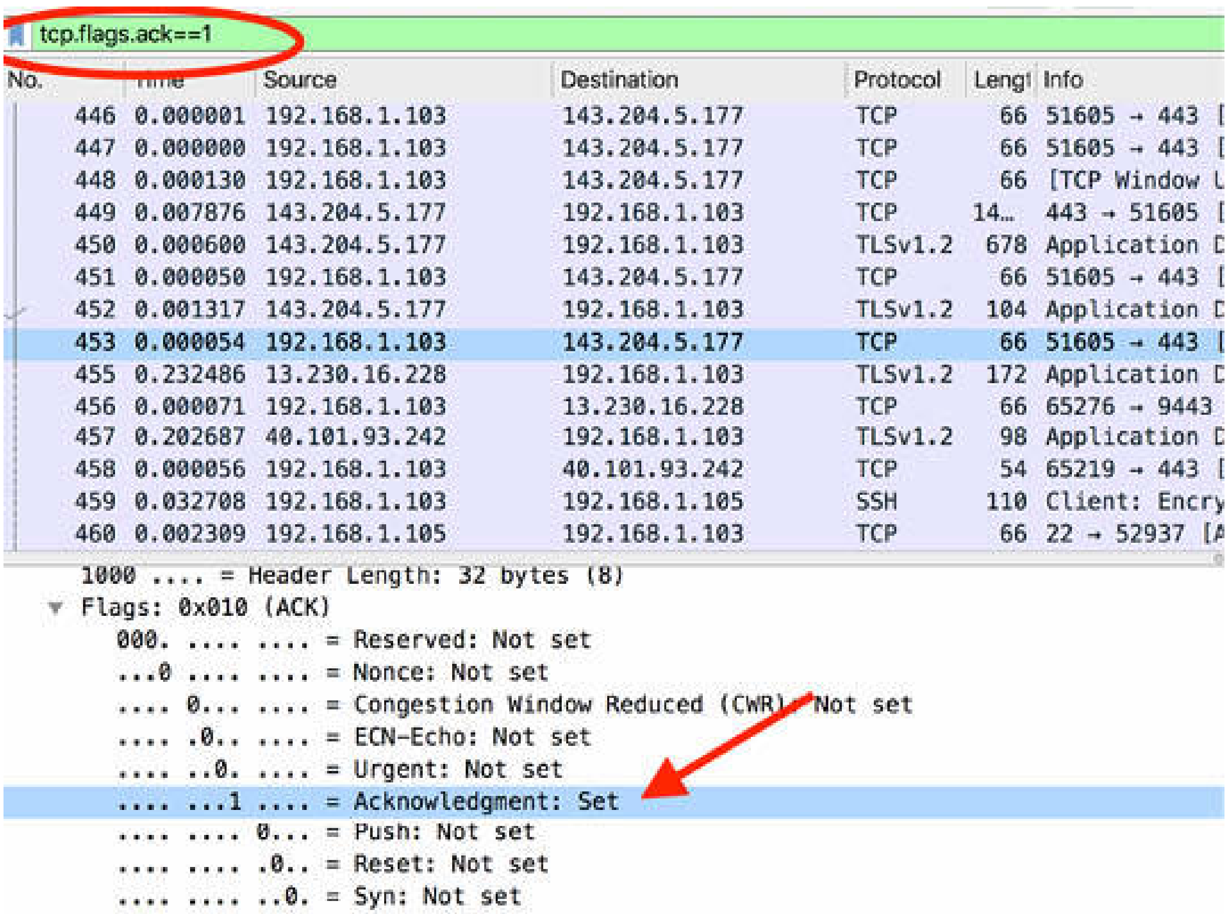
Task 3:

To select packets with headers that contain one or more options, use the tcp.hdr_len > 20 filter. The results are displayed in the figure below, indicating that the filters are being applied.



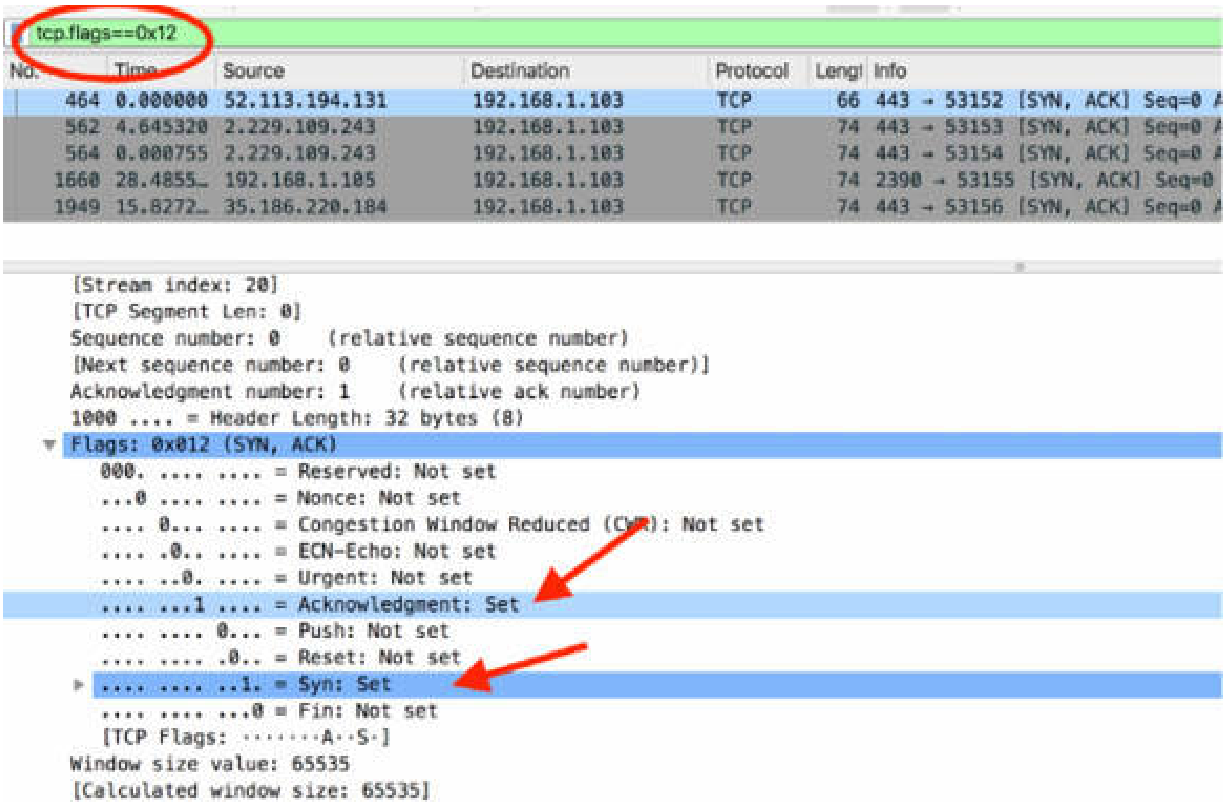
Task 4:
To catch only those packets that have the Congestion Window Reduced flag or the ECN-Echo flag set, use the !(tcp.flags.cwr==0) || !(tcp.flags.ecn==0) filter.

When working with TCP flags filters, sometimes, rather than creating a filter by using the individual bit settings—such as tcp.flags.ack==1 or tcp.flags.push==0 or tcp.flags.fin==0 —you can consider creating a filter based on the TCP flags summary line. The figure below shows an example of a single flag filtering, selecting the TCP ACK messages.



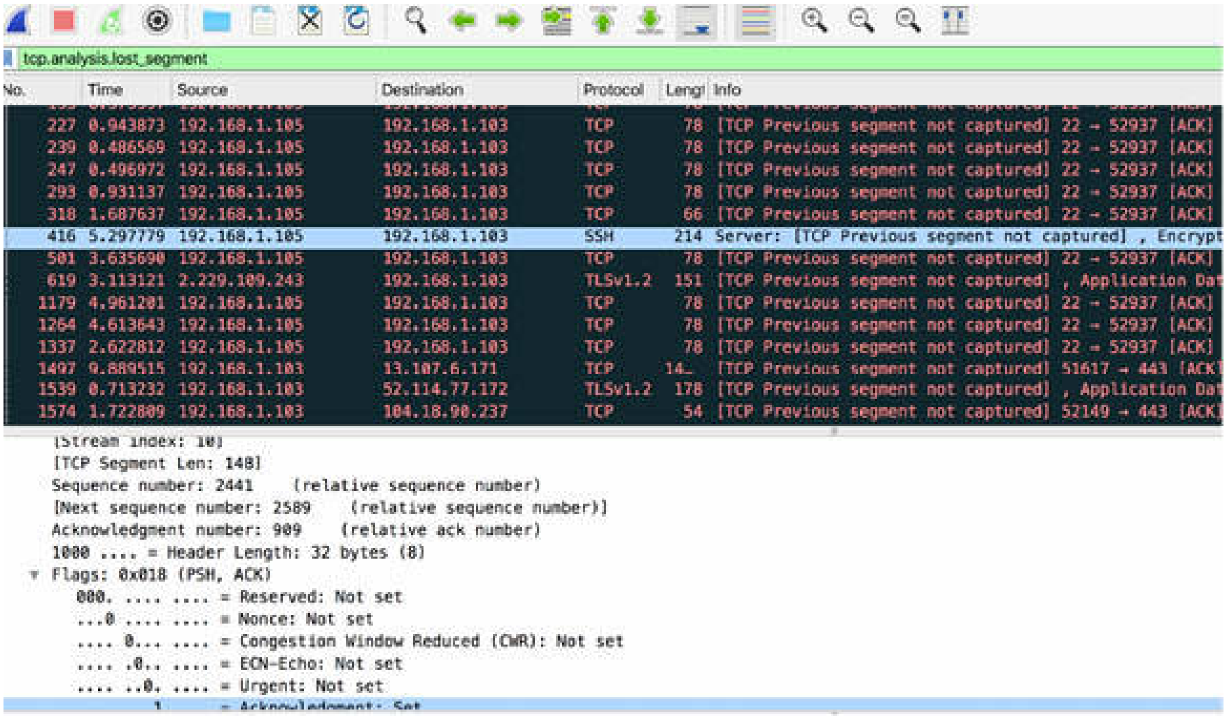
To display all SYN/ACK packets, use the tcp.flags==0x12 filter. In explicit and extended notation, this filter is equivalent to:

tcp.flags.urg==0 && tcp.flags.ack==1 && tcp.flags.push==0 && tcp.flags.reset==0 && tcp.flags.syn==1 && tcp.flags.fin==0.



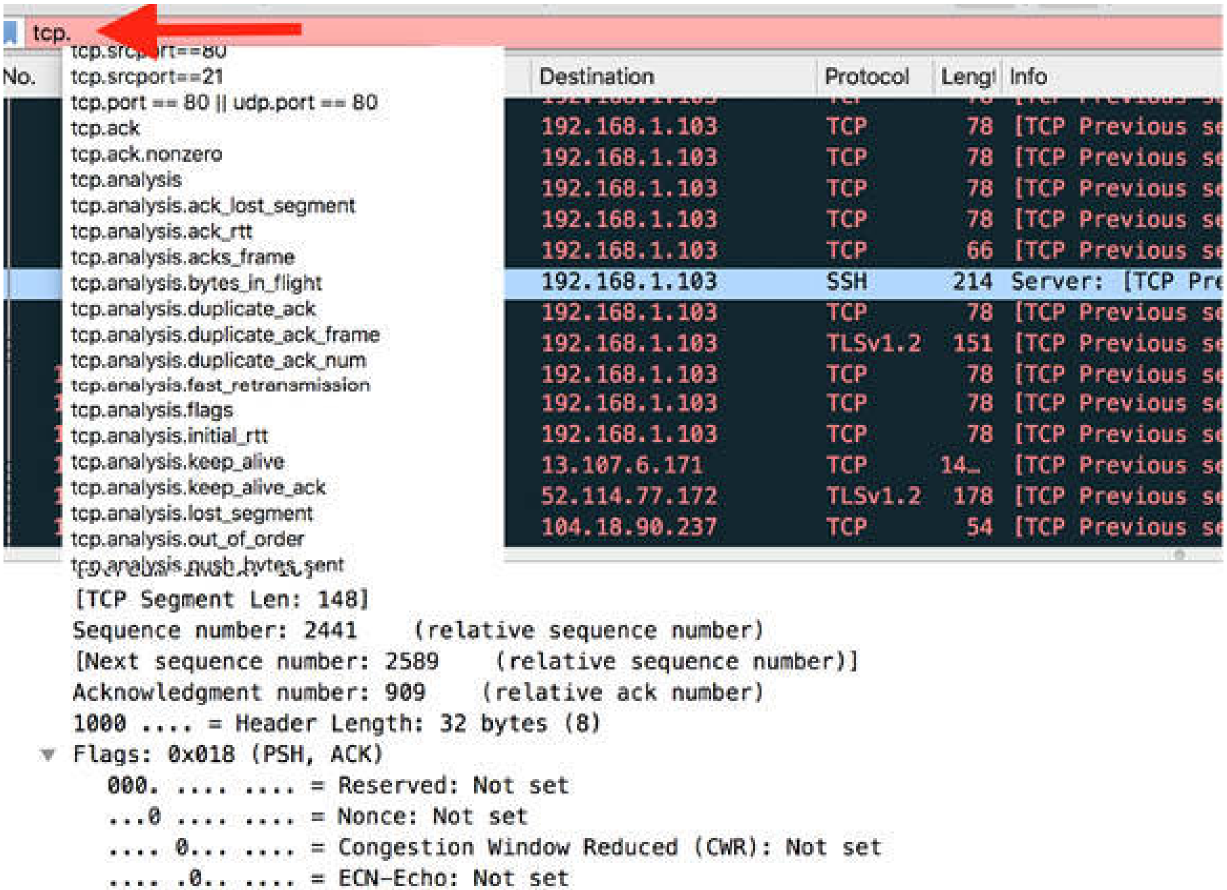
Task 5:

To filter out when a lost segment was detected before this packet, use the tcp.analysis.lost_segment filter, as shown in the figure below.



Task 6:

If you need a particular filter, you can use the auto-complete feature available in the filter toolbar. For example, if you just type tcp , the auto-complete feature displays a drop-down menu with all available TCP filters, as shown in the figure below.



Notes:

Repeat the previous tasks for different TCP filters to gain more confidence in filtering TCP fields. Use the auto-complete feature to find the most appropriate filter for selecting packets of interest and find more than one way to apply the same filter action.

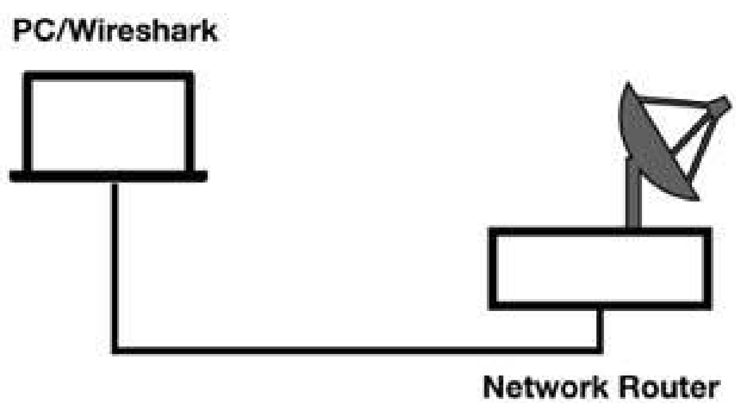
Lab 53. TCP Preferences

Lab Objective:
Learn how to use the available TCP preferences.

Lab Purpose:
Learn the various options offered in the TCP preference settings.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

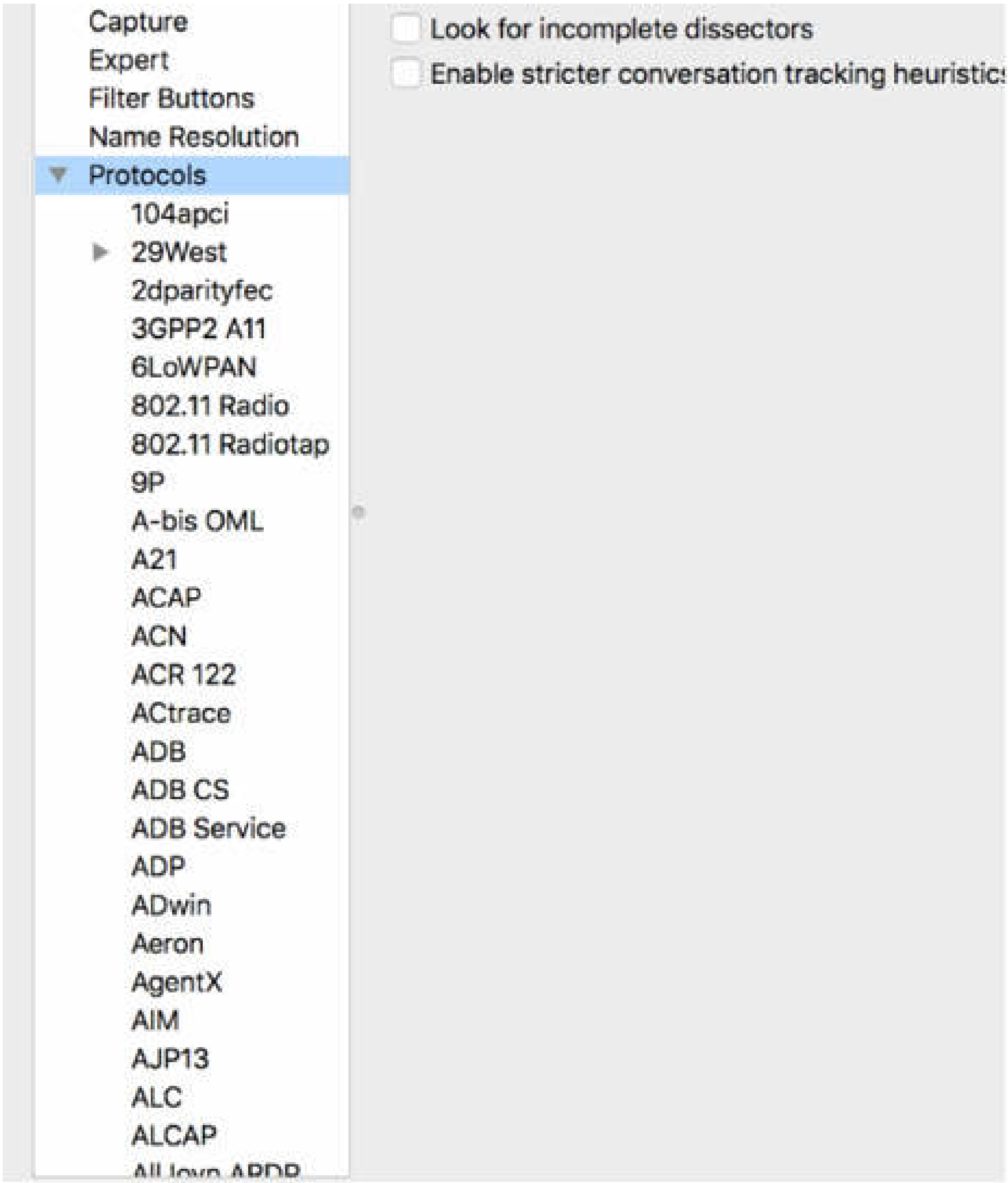
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



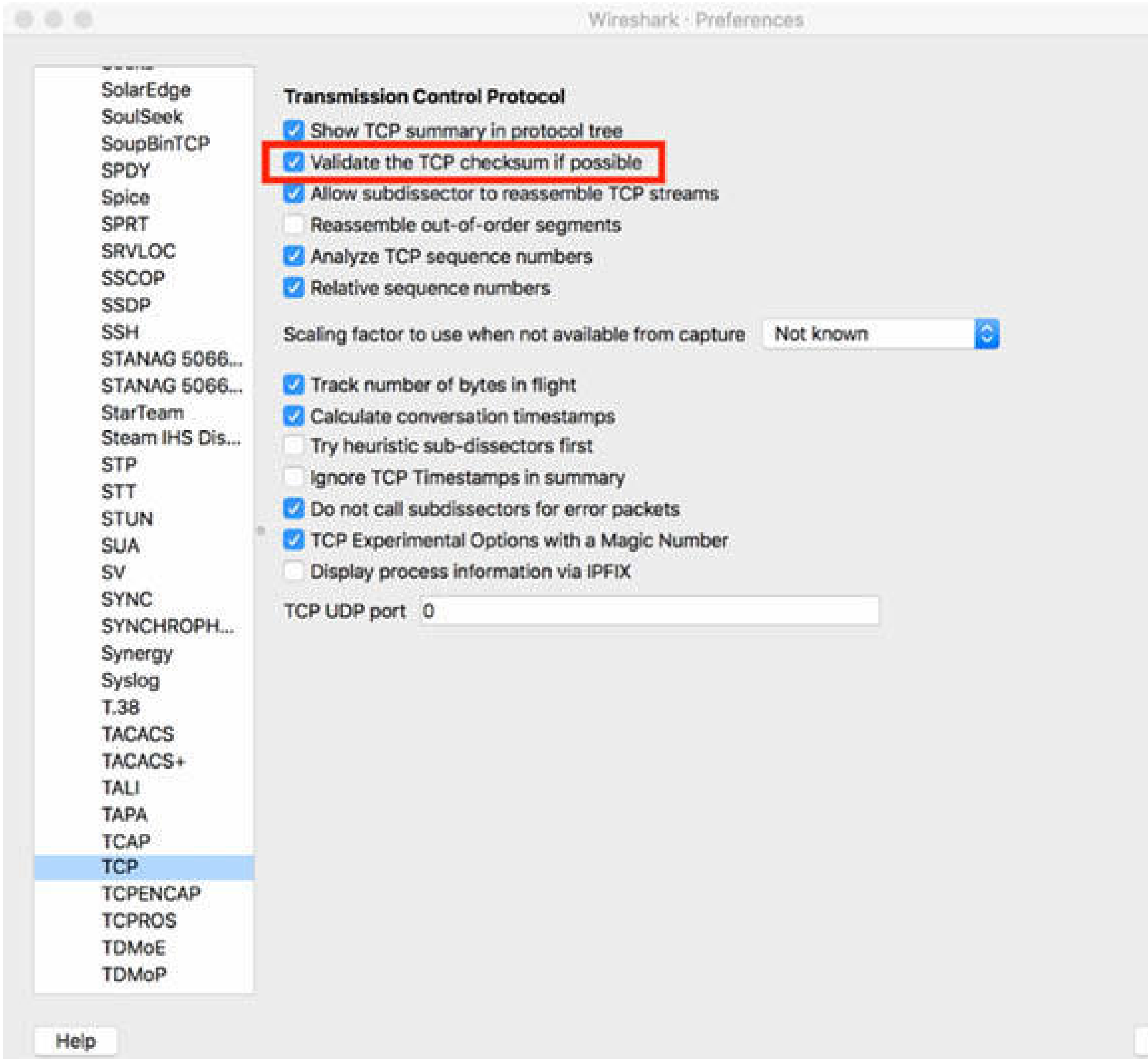
Lab Walkthrough:

Task 1:
Open a packet capture saved in one of the previous labs. In the filter toolbar, enter tcp to display only TCP packets.

On the main menu, select Edit > Preferences. In the left tree view in the Preferences dialog box, select Protocols > TCP.

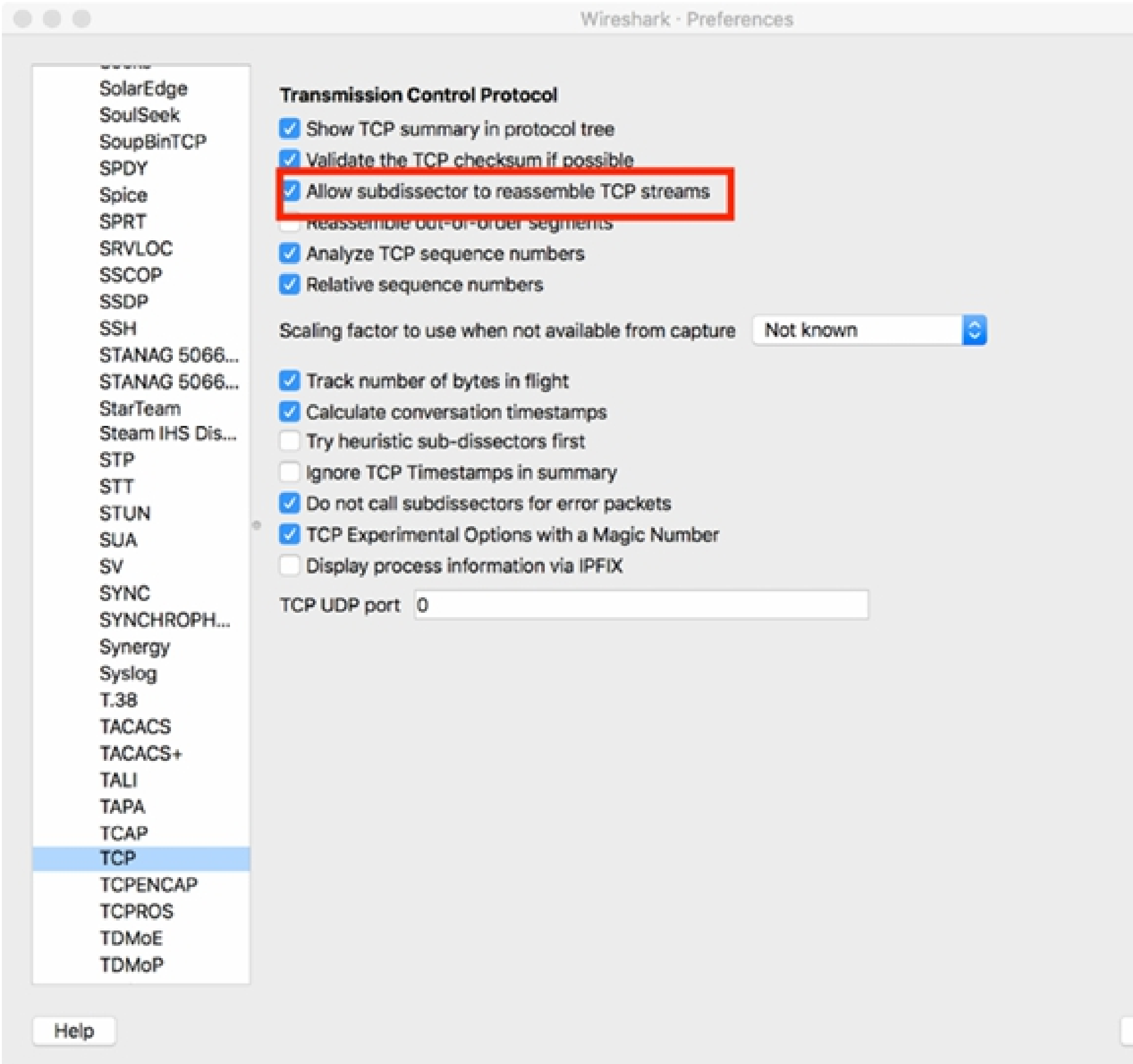


The "Validate the TCP checksum if possible" feature examines the TCP header checksum.

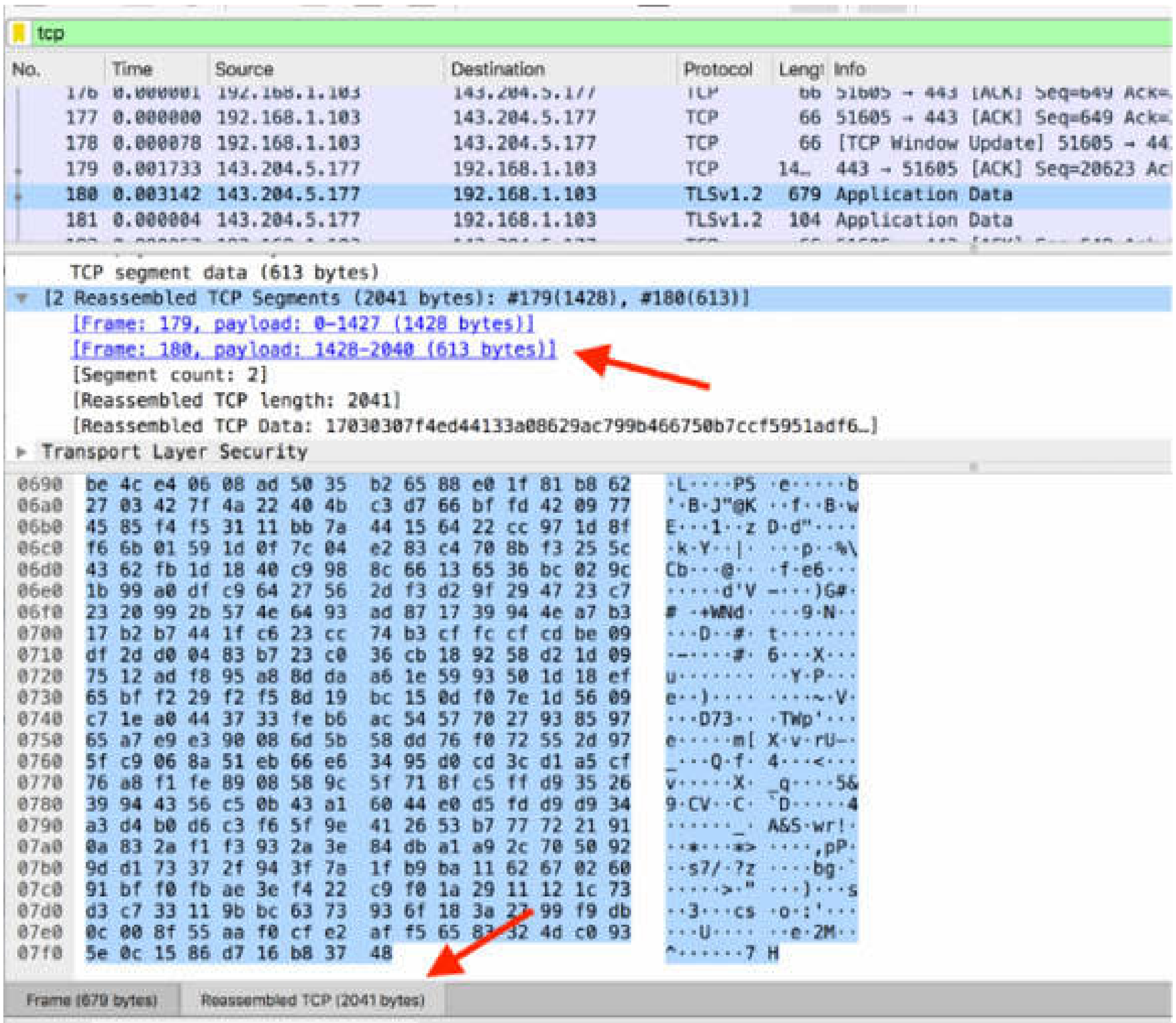


If you are capturing trace files on your own system and each TCP packet sent from your host indicates that the TCP checksum is invalid, your network interface card and the driver may support checksum offloading. You can also disable the Checksum Errors coloring rule or specific checksum validation processes.

Task 2:
The “Allow subdissector to reassemble TCP streams” feature is another useful feature.



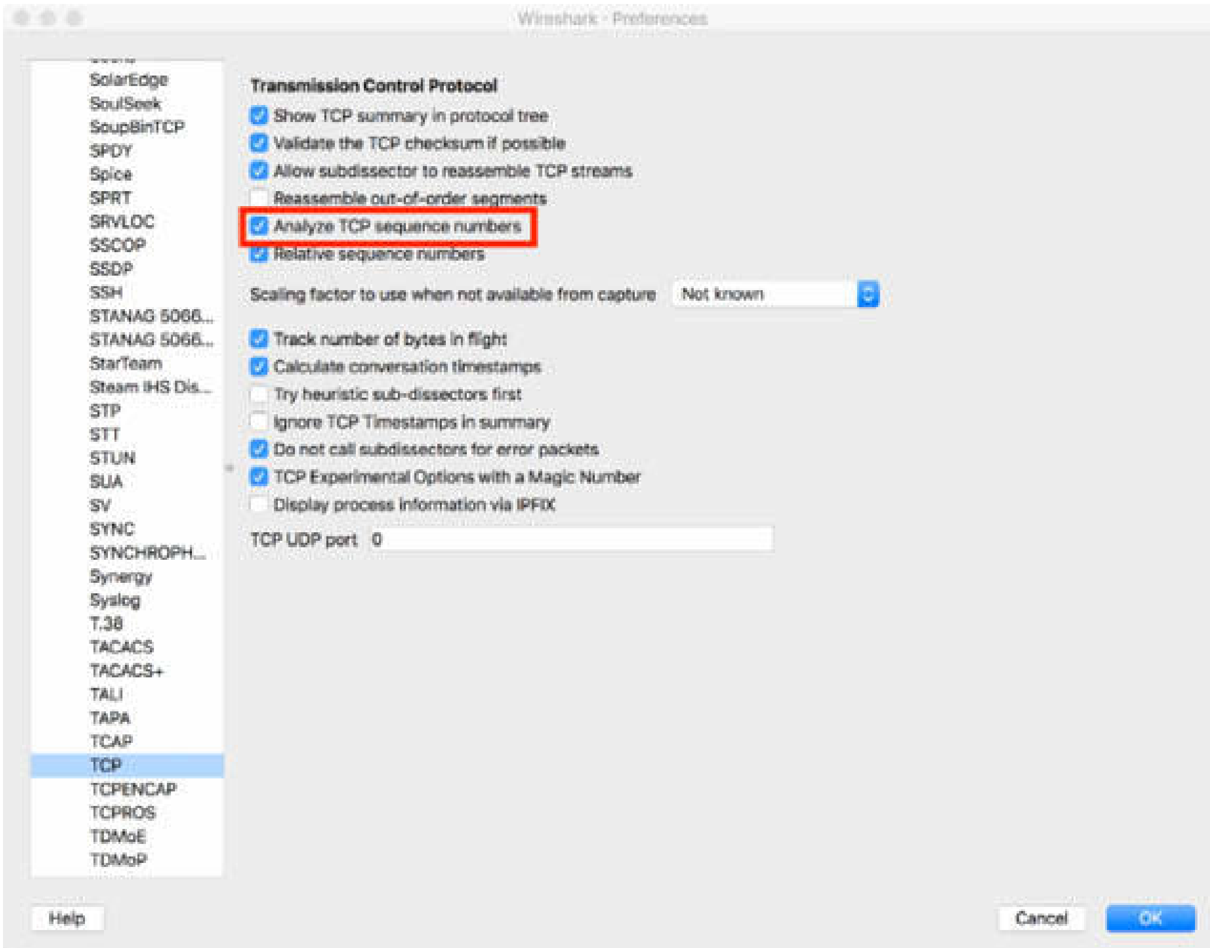
When working with a TCP data stream, you can choose to have Wireshark reassemble the stream and provide links to each packet containing data from the stream. When enabled, this setting also alters the display in the Packet List pane.



To get the Packet Details data reassembled or view a frame split in non-reassembled format, click the “Reassembled TCP” tab, shown in the figure above.

Task 3:

The “Analyze TCP sequence numbers” setting is enabled by default. It provides more efficient analysis by tracking the sequence number and the acknowledgment number values.

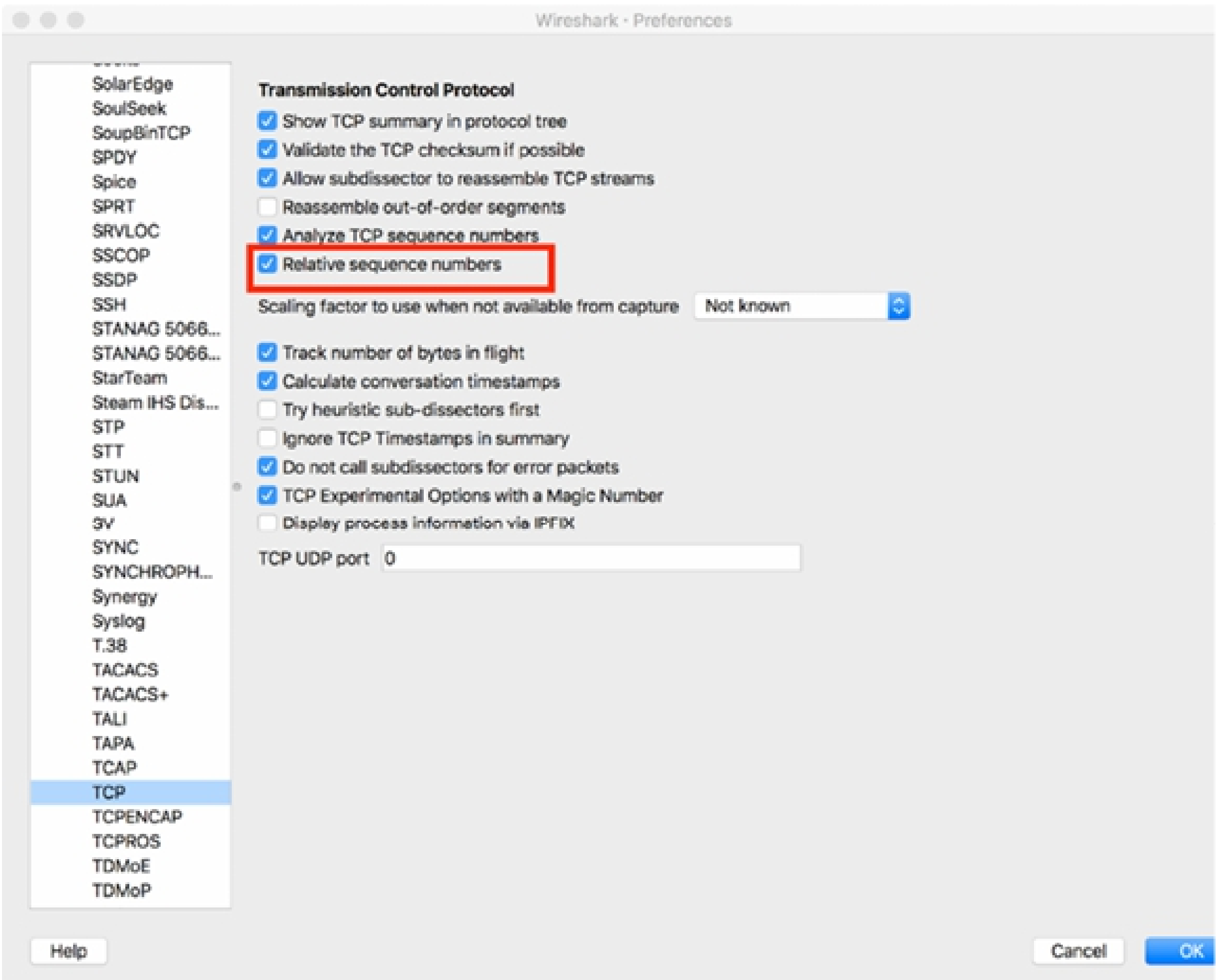


If you disable the above feature, the Expert Information related to these TCP conditions is automatically disabled. This feature is used to identify particular TCP conditions such as:

- Lost segments
- Window is Full
- Out-of-order segments
- Frozen Window
- Duplicate ACKs
- Window Updates
- Retransmissions and Fast Retransmissions

Task 4:

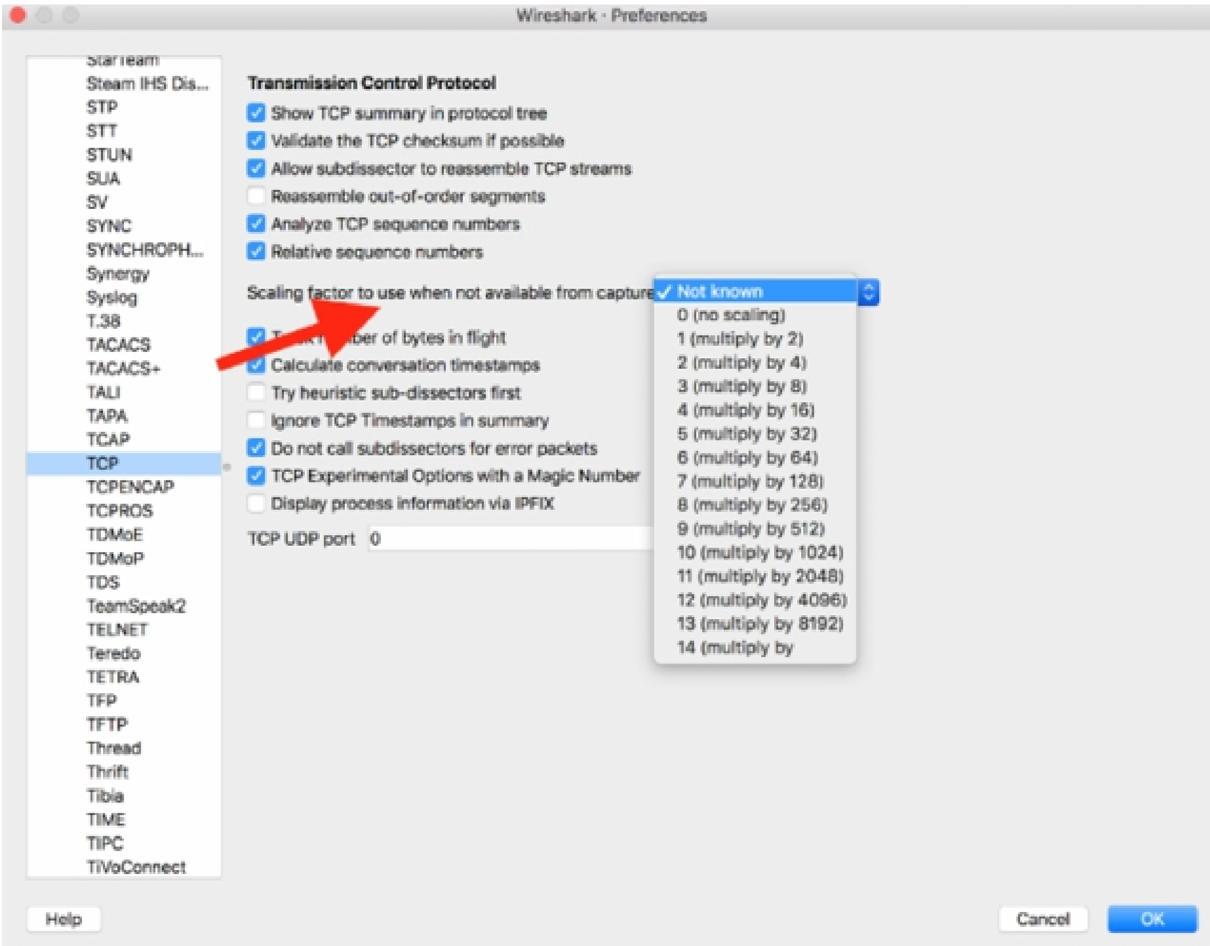
The “Relative Sequence Numbers” setting is enabled by default. It provides more efficient analysis by starting the TCP sequence number value from 0 for both sides of a TCP connection.



Task 5:

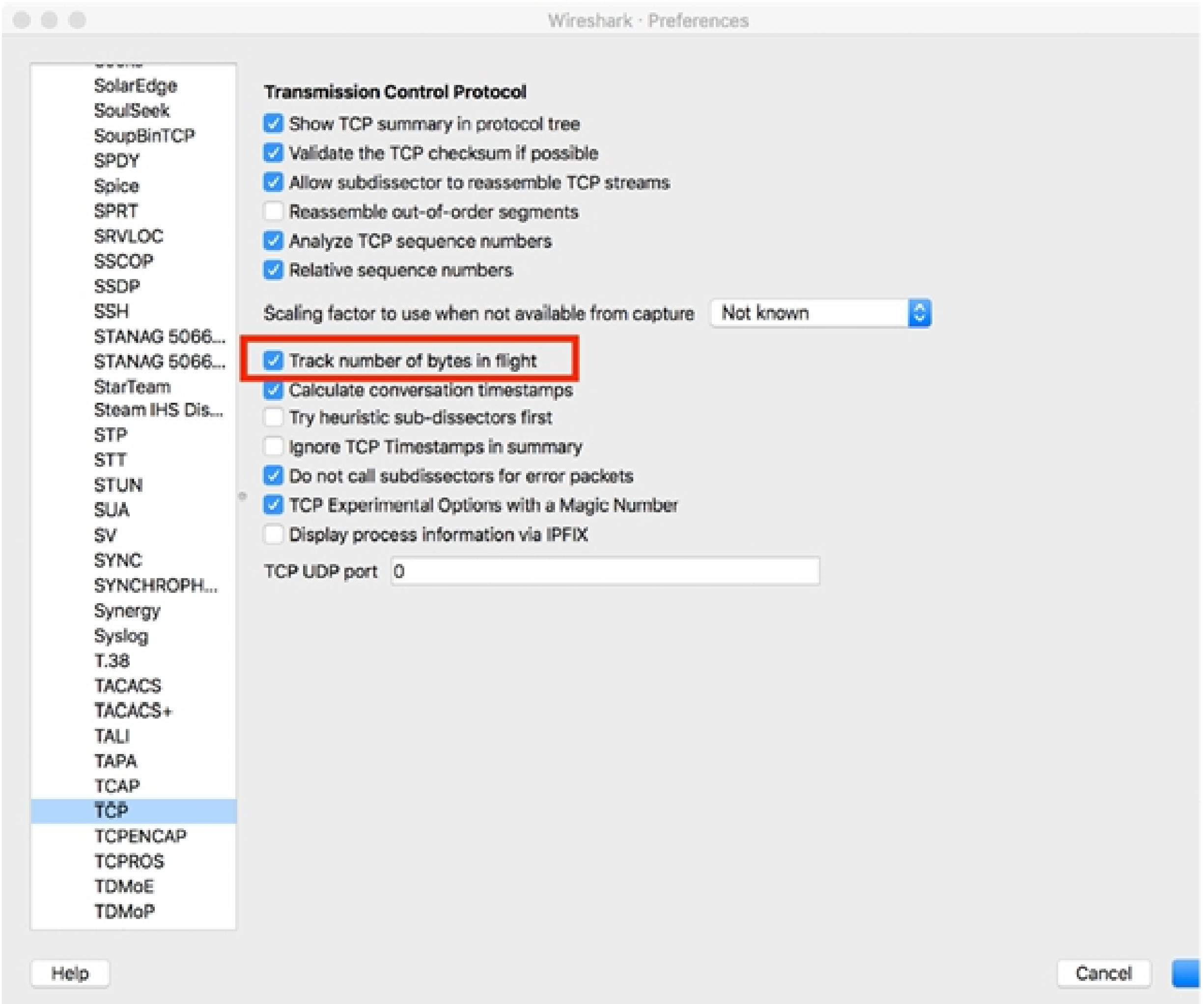
The Window Size Scaling Factor feature lets you choose a scaling preference. Wireshark displays the actual Window Size field and a separate Calculated Window Size field. If Wireshark sees the TCP handshake process, it evaluates the value of the TCP window scale option and performs the calculation to display the actual window size being advertised. If Wireshark does not see the TCP handshake, the Window Size Scaling Factor is marked as –1. If Wireshark sees the TCP handshake, but window scaling is not being used, Wireshark marks the Window Size Scaling Factor as –2.

In the figure below, the available options for the scaling factor are displayed.



Task 6:

The "Track number of bytes in flight" setting enables Wireshark to track the number of unacknowledged bytes flowing in the network.



When this setting is enabled, you can create an I/O graph depicting the total number of bytes in the trace file and the number of bytes in flight by using the tcp.analysis.bytes_in_flight display filter value. This is particularly useful when you suspect that congestion window issues are slowing file transfer processes.

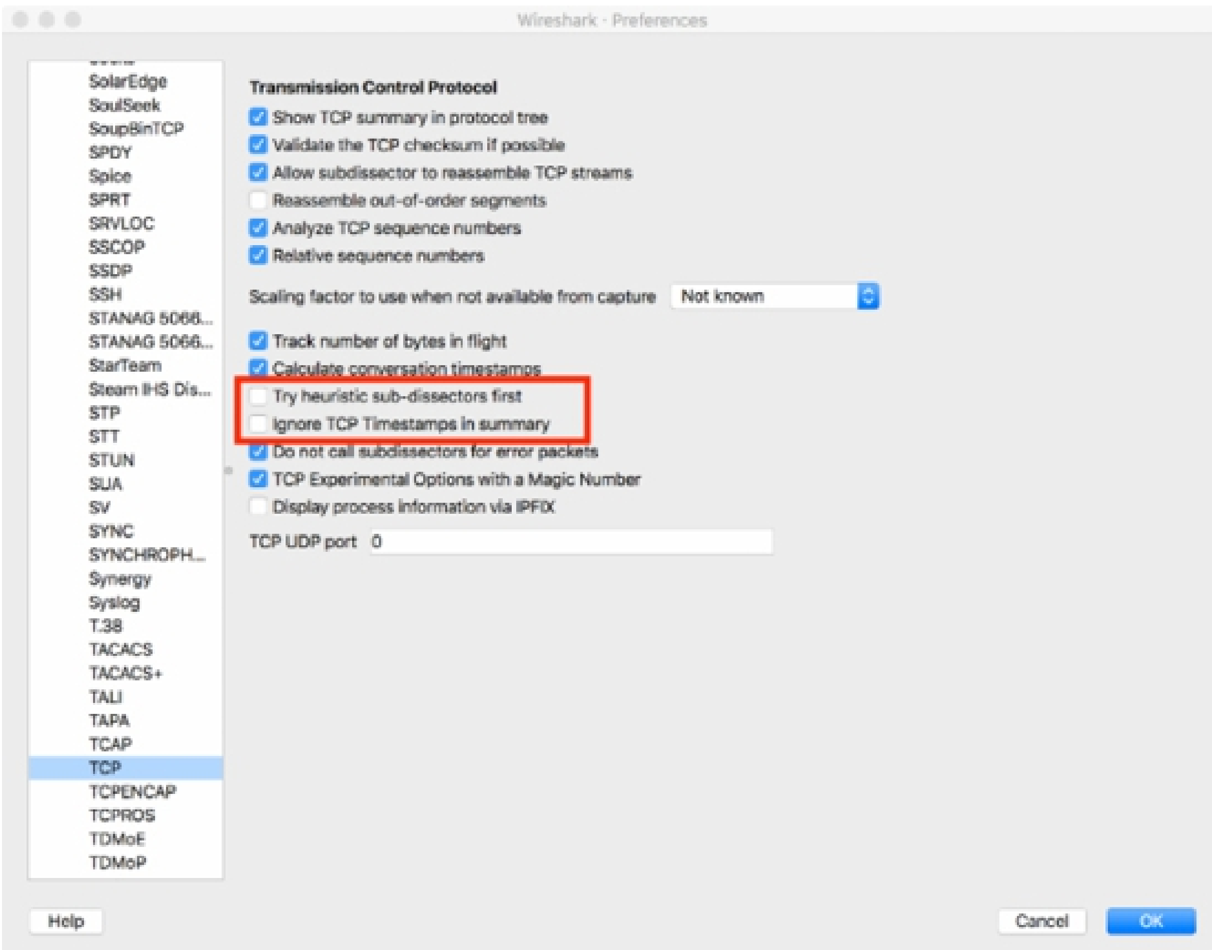
Task 7:

The "Try heuristic sub-dissectors first" and "Ignore TCP Timestamps in summary" features, shown in the figure below, are not enabled by default.

The "Try heuristic sub-dissectors first" feature is useful when you have applications running over non-standard port numbers and you want

Wireshark to automatically detect which application is in use, and accordingly, apply a proper dissector.

The "Ignore TCP Timestamps in summary" feature is useful because sometimes (or for some users in particular) having all that extra information in the Info column is considered as noise.



Notes:

Repeat the previous tasks and change the value for each field available in the preference settings of TCP. Observe the effect in the capture file.

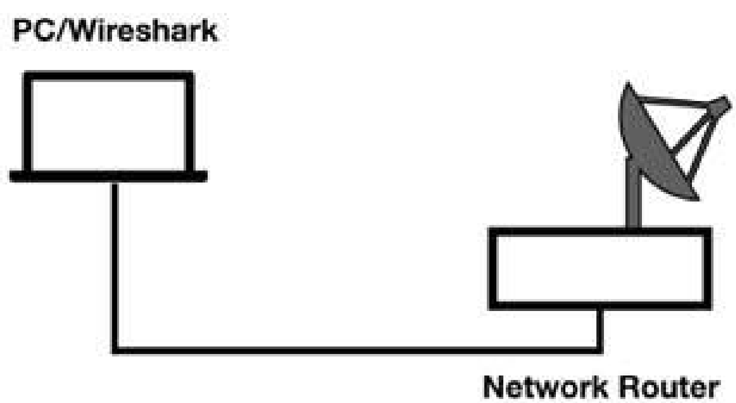
Lab 54. Basic Graphs

Lab Objective:
Learn how to create basic graphs.

Lab Purpose:
Learn how to generate basic graphs to view trends.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
Wireshark offers numerous graphs to depict traffic flow trends. Some graphs are directional, focusing on traffic flowing in a specific direction.

Other graphs, such as the Input/Output (I/O) graph, depict traffic flowing in both directions.

For I/O graphs, you can manipulate the X and Y axis values. Most other graphs automatically define the X and Y axis values based on the traffic being graphed.

I/O graphs support the display filter functionality and expressions. The advanced I/O graphs also support calculations. You can also export some graphs and save them for post-processing.

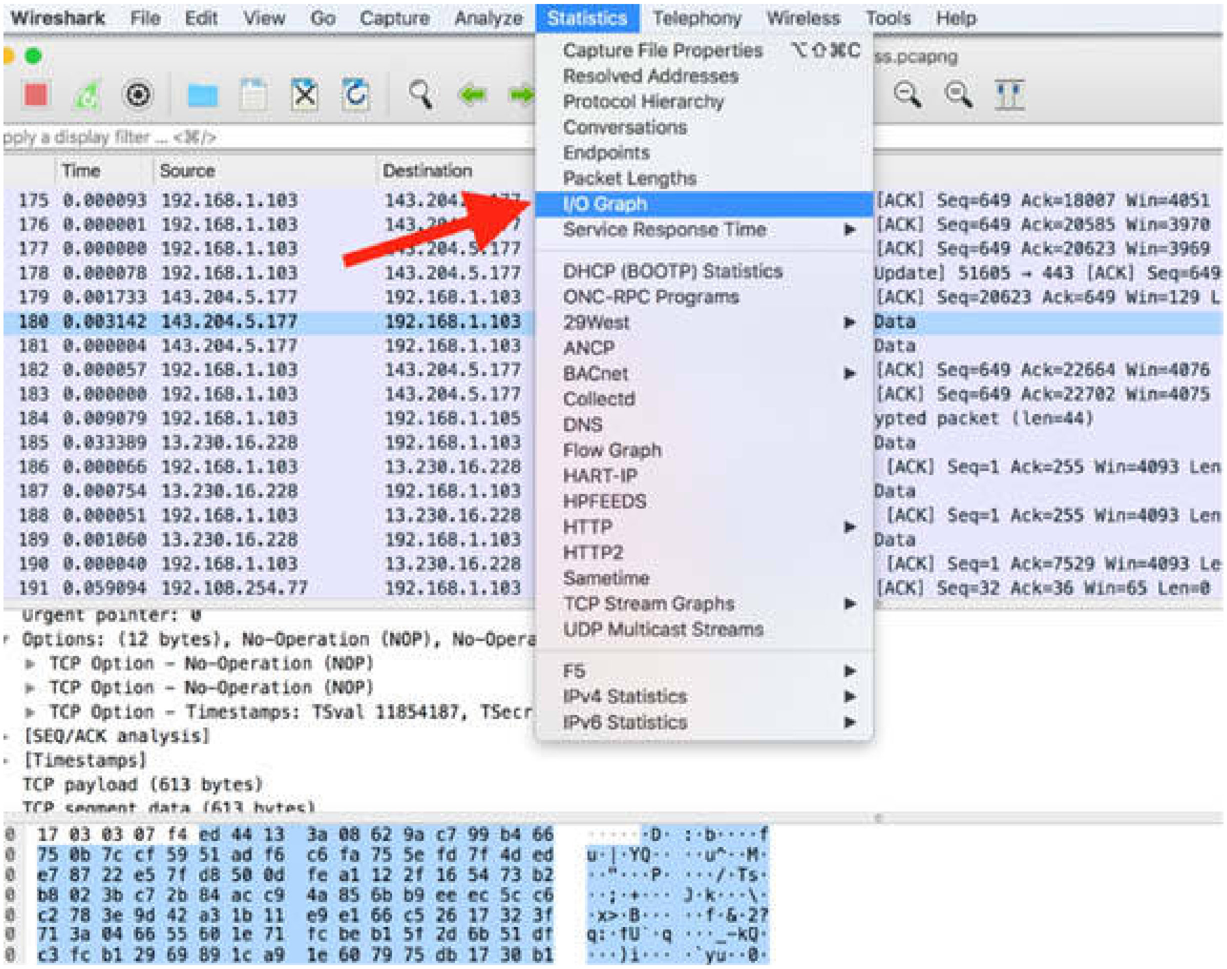
If you generate a graph that appears empty or shows too few plot points, it might be a unidirectional graph. Examine the title bar to see what traffic is being graphed. If it is a unidirectional graph and you have selected a packet flowing in the wrong direction, close the graph and rebuild it. Before rebuilding the graph, select a packet flowing in the opposite direction.

I/O graphs are very useful in showing the overall traffic seen in unsaved or saved trace files. I/O graphs depict the total amount of bytes seen including data and headers.

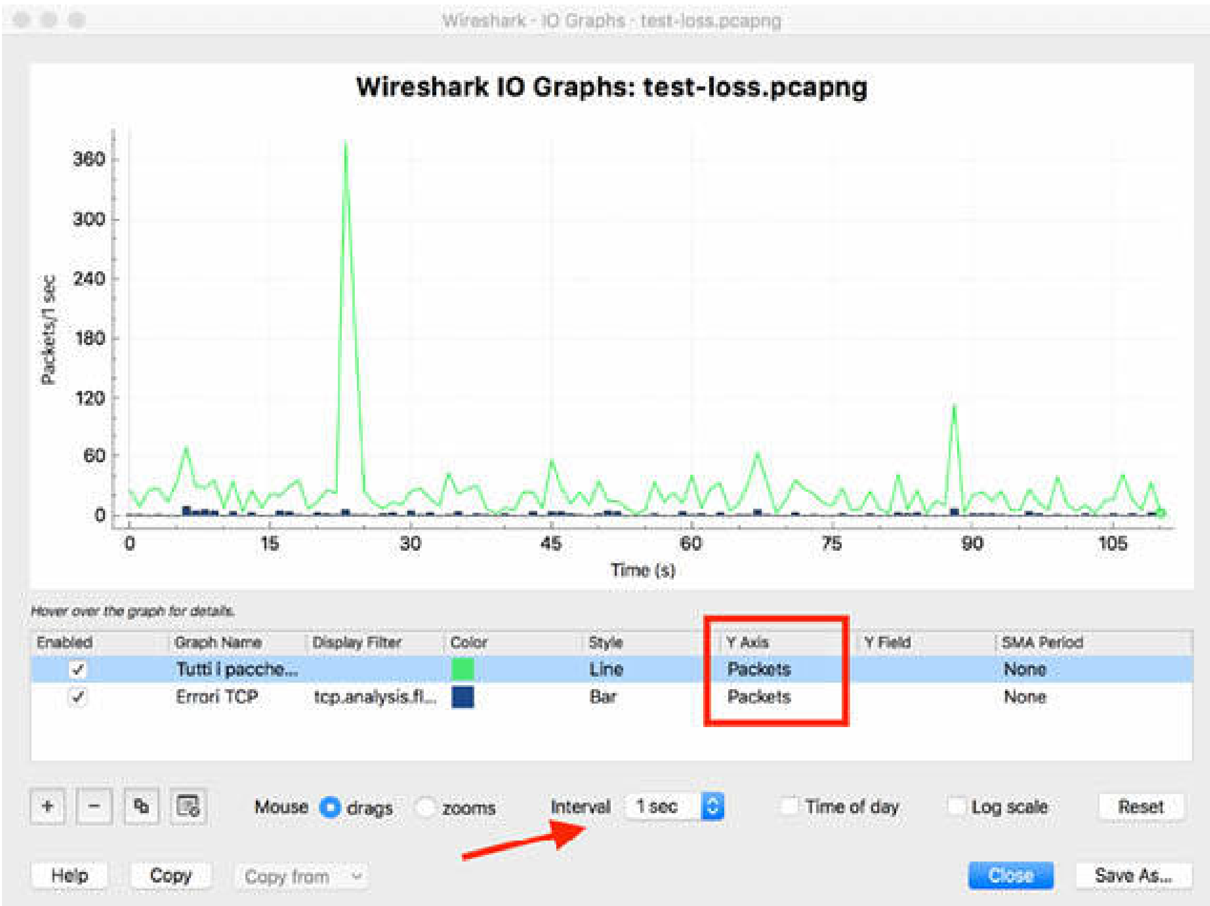
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

Stop the capture and save the file.

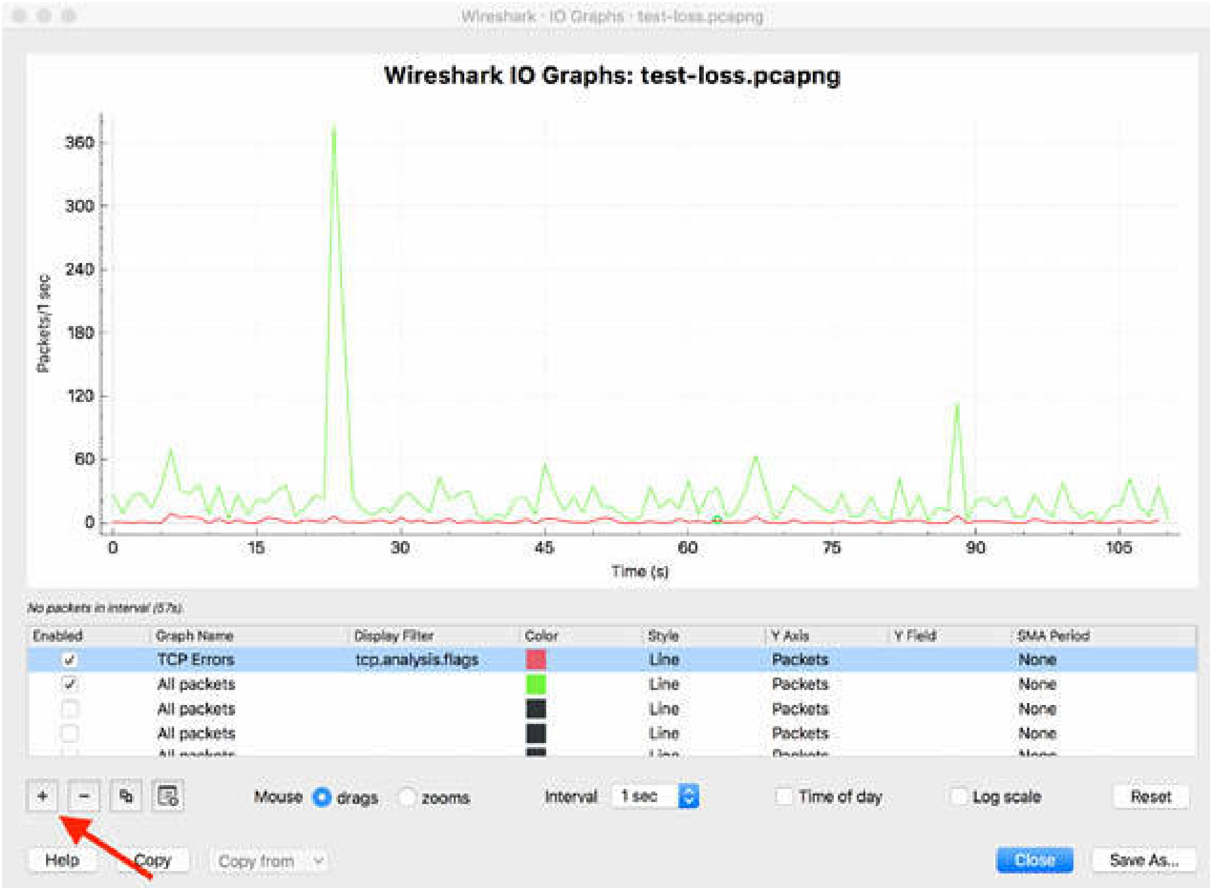
To plot the packets per second rate of all the traffic in the saved trace file, on the main menu, select Statistic > I/O Graph, as shown in the figure below.


Wireshark File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Capture File Properties
Resolved Addresses
Protocol Hierarchy
Conversations
Endpoints
Packet Lengths
I/O Graph
Service Response Time
DHCP (BOOTP) Statistics
ONC-RPC Programs
29West
ANCP
BACnet
Collectd
DNS
Flow Graph
HART-IP
HPFEEDS
HTTP
HTTP2
Sametime
TCP Stream Graphs
UDP Multicast Streams
F5
IPv4 Statistics
IPv6 Statistics

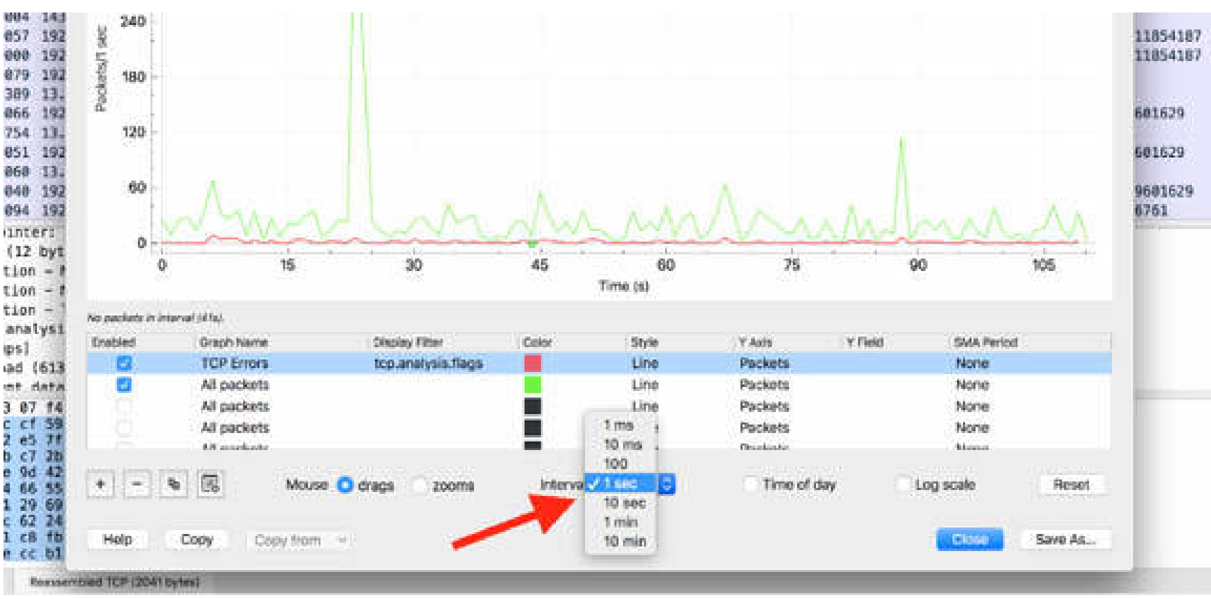
By default, the X axis is set to a tick interval of one second, and the Y axis is set to packets/tick, as shown in the figure below.



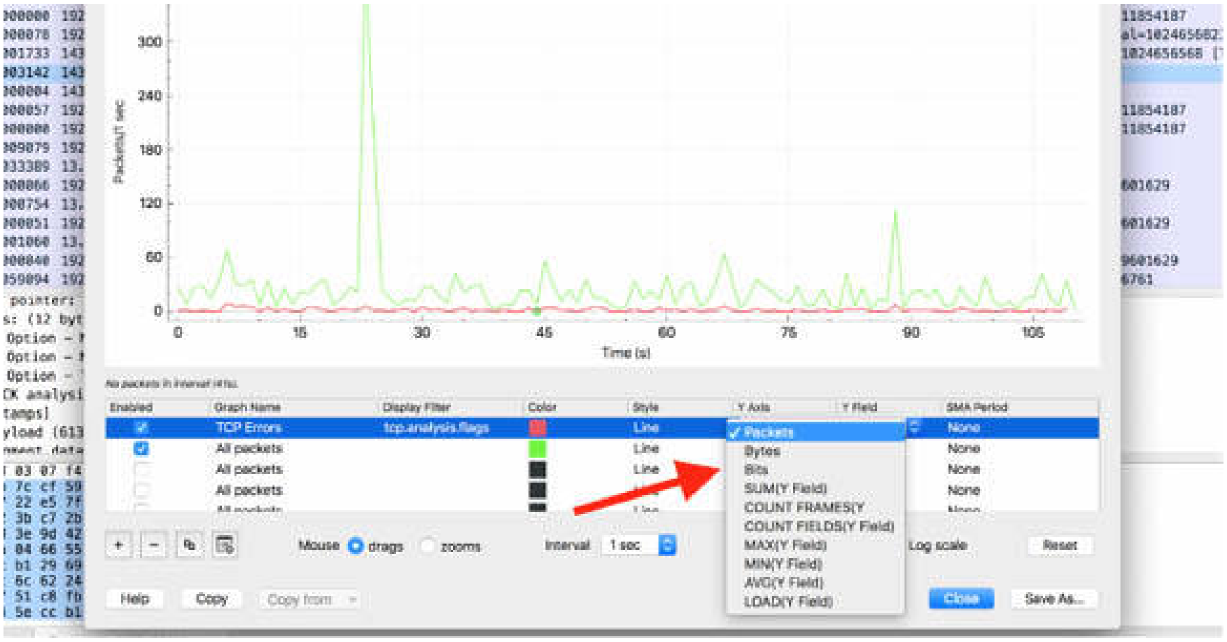
In the standard graph mode, you can add more than one traffic channel by clicking the add (+) button, as shown in the figure below.



You can also alter the X axis to change the tick interval and the pixels per tick. In the Interval list, change the tick interval.

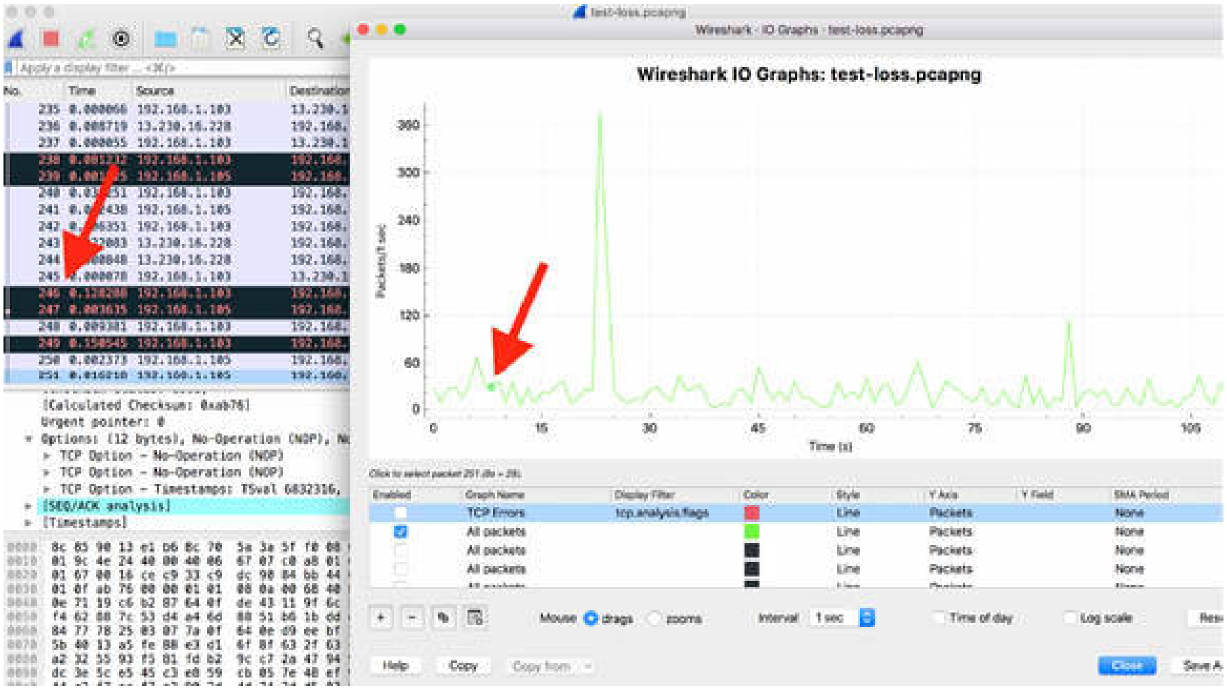


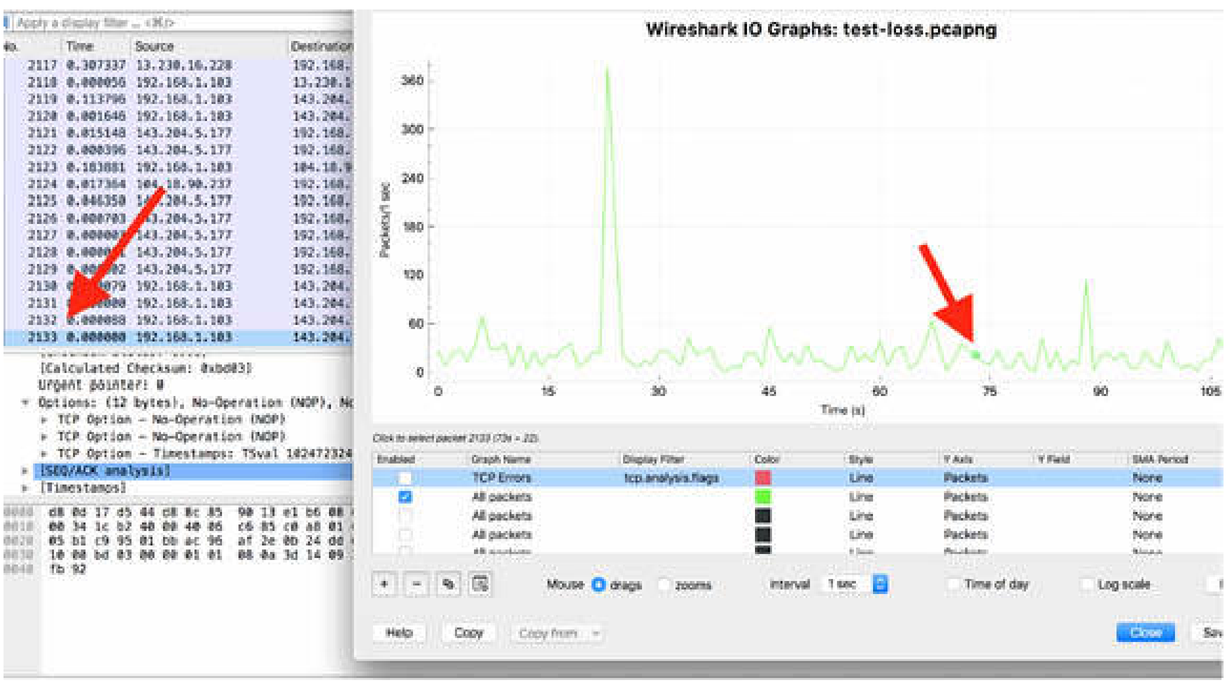
You can also adjust the Y axis to correctly set the units and scale. Double-click the “Y Axis” field to select a unit.



In the figure above, the green graph shows all packets because no display filters have been applied. Therefore, all traffic seen in the trace file has been graphed, that is, the entire size of all packets (including headers and data).

Click on a point in the I/O graph to jump to the first packet, used for calculating that graph point, in the Packet List pane. The figures below show two examples of the jump.

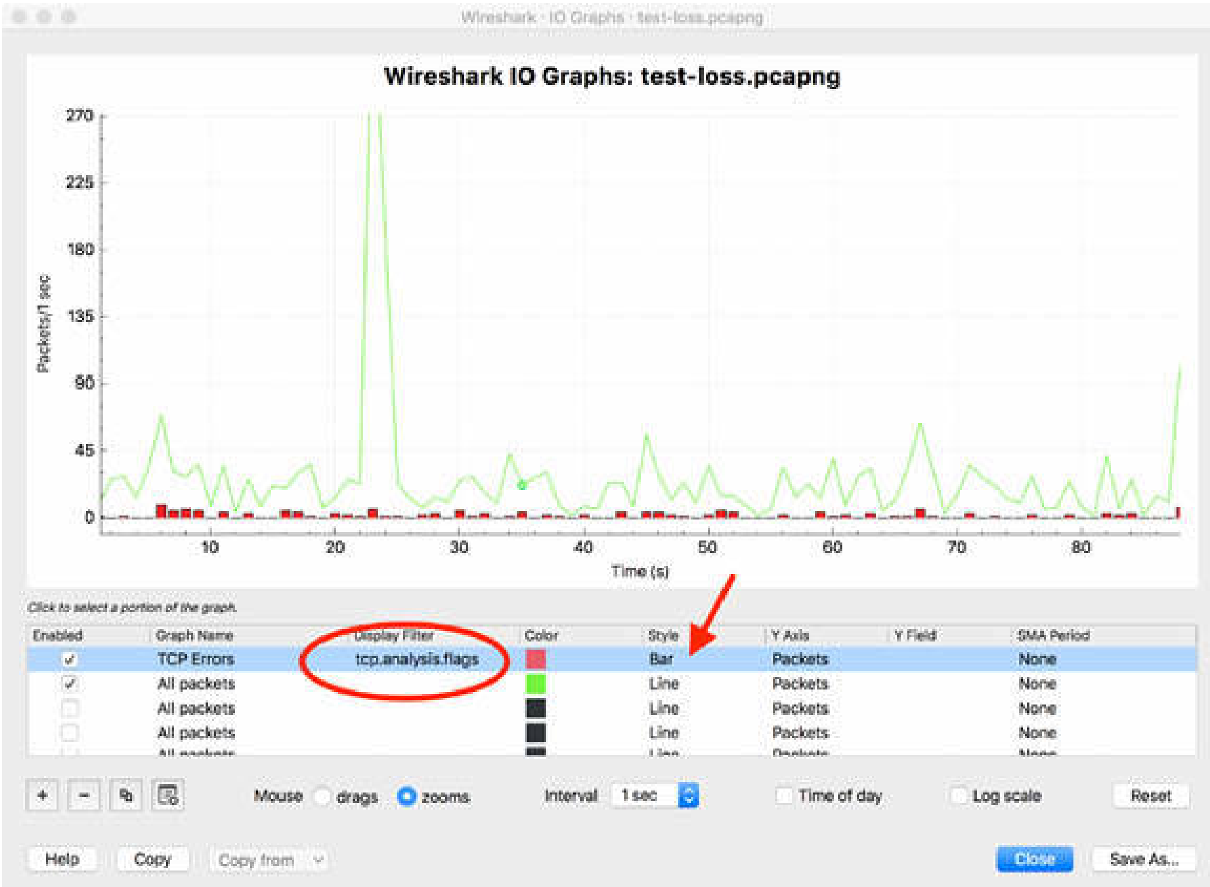




Task 2:

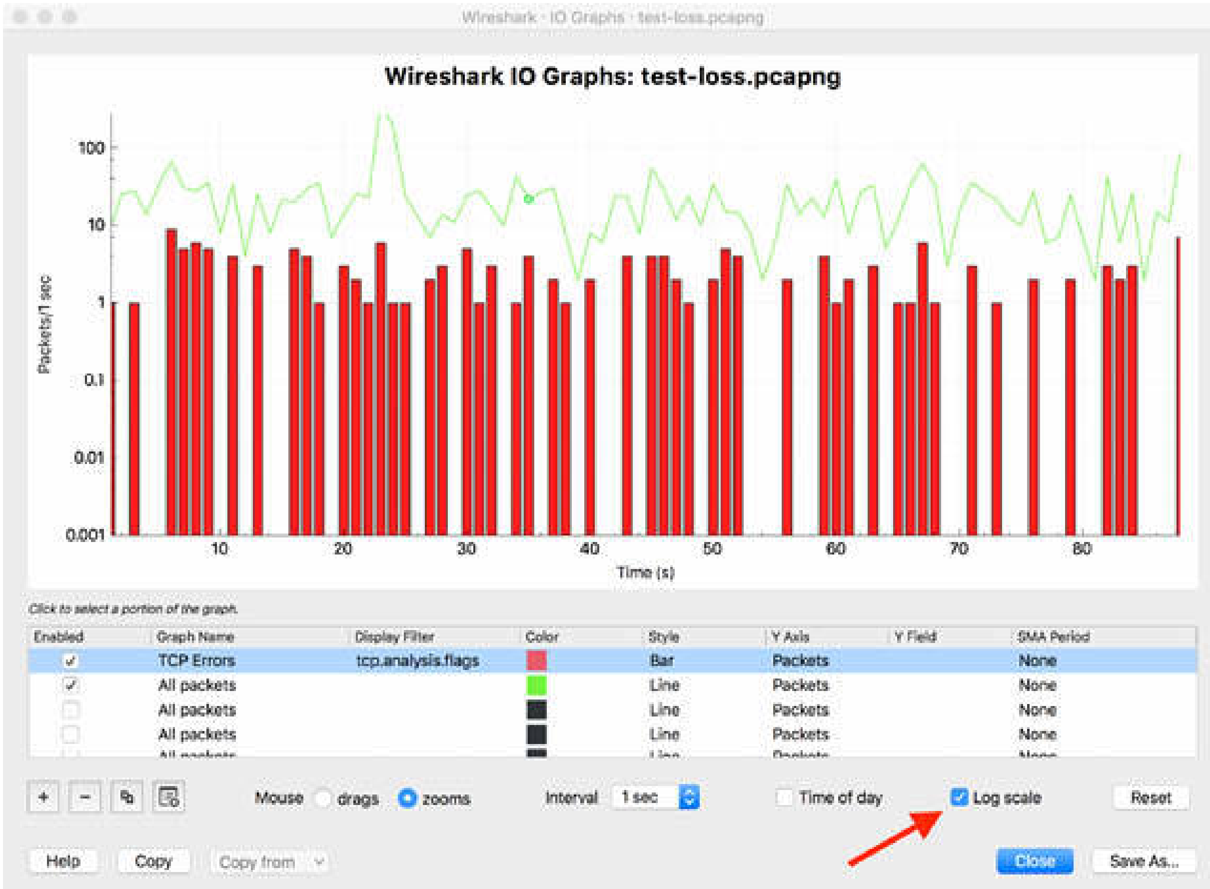
To graph specific traffic and compare it to the overall traffic, apply a filter to any of the additional graph channels.

In the figure below, the tcp.analysis.flags display filter is applied, and the style of the graph is changed to Bar.



You can set different colors for the channels used for graphs. In general, it is useful to stick to the standard color codes: green for good; red for bad.

To graph some points that have a large difference between their numerical values, you can use the logarithmic scale by selecting the “Log scale” check box, shown in the figure below.



Click the Save As button to save the generated graph. The default format is PNG but other formats—such as BMP, GIF, JPEG, and TIFF—are also available.

Notes:

To gain the necessary confidence in using the options available for graphs, repeat the previous tasks, and change the number of channels used, the coloring rules, the filters, and the resolution of the graph.

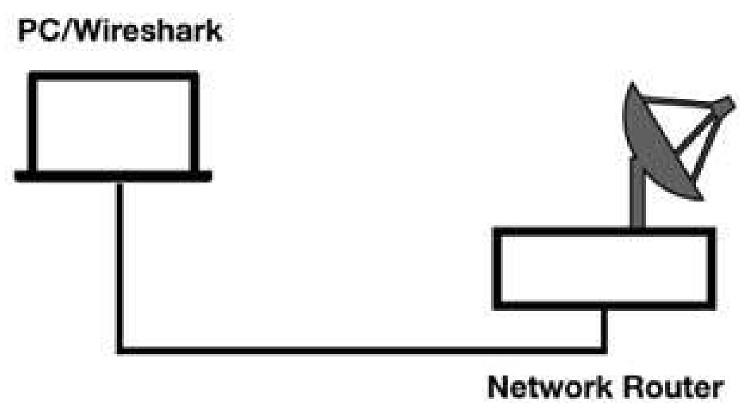
Lab 55. Advanced I/O Graphs

Lab Objective:
Learn how to create advanced I/O graphs starting with the basic graphs.

Lab Purpose:
Learn the functionality of advanced I/O graphs offered by Wireshark.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



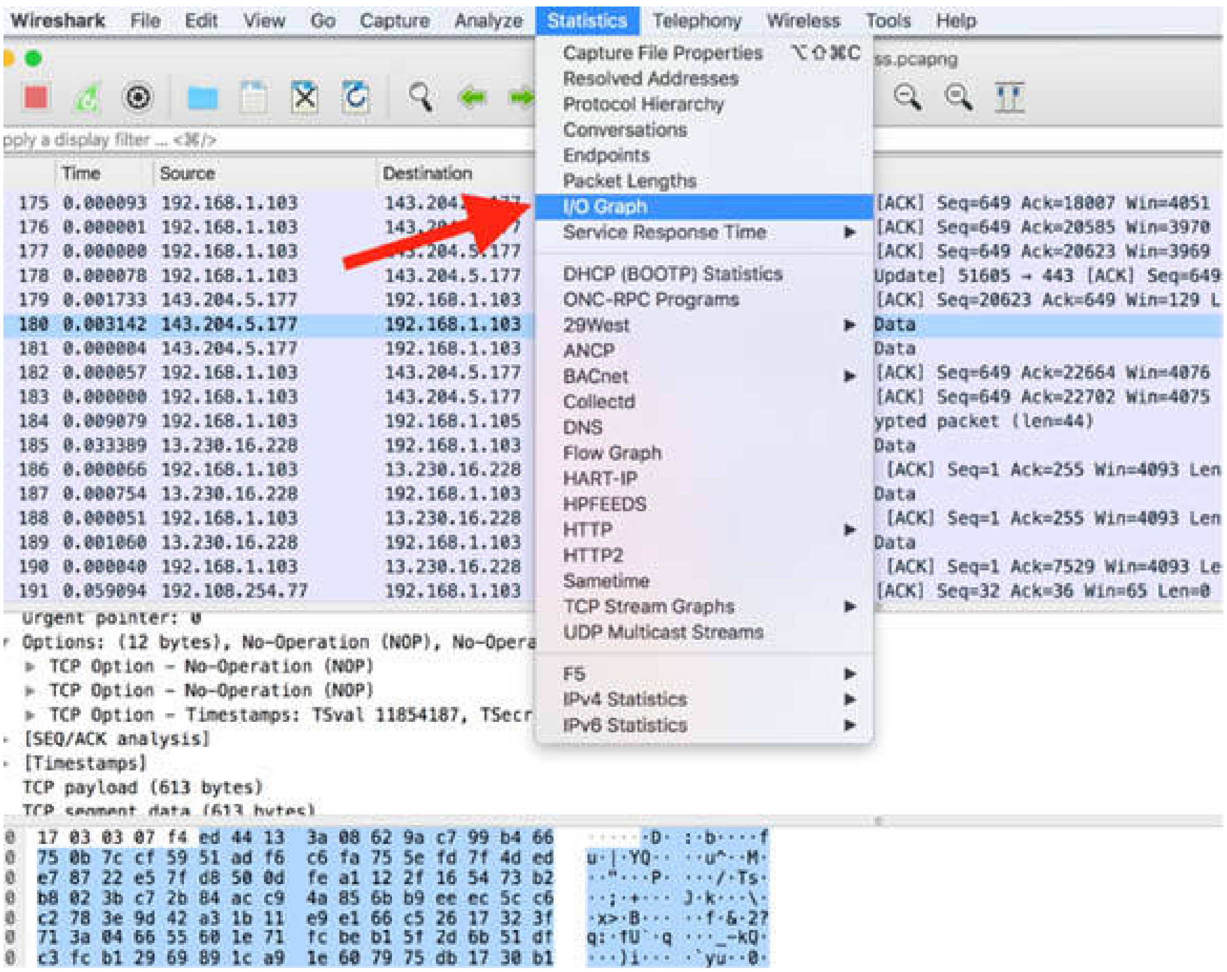
Lab Walkthrough:

Task 1:
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic

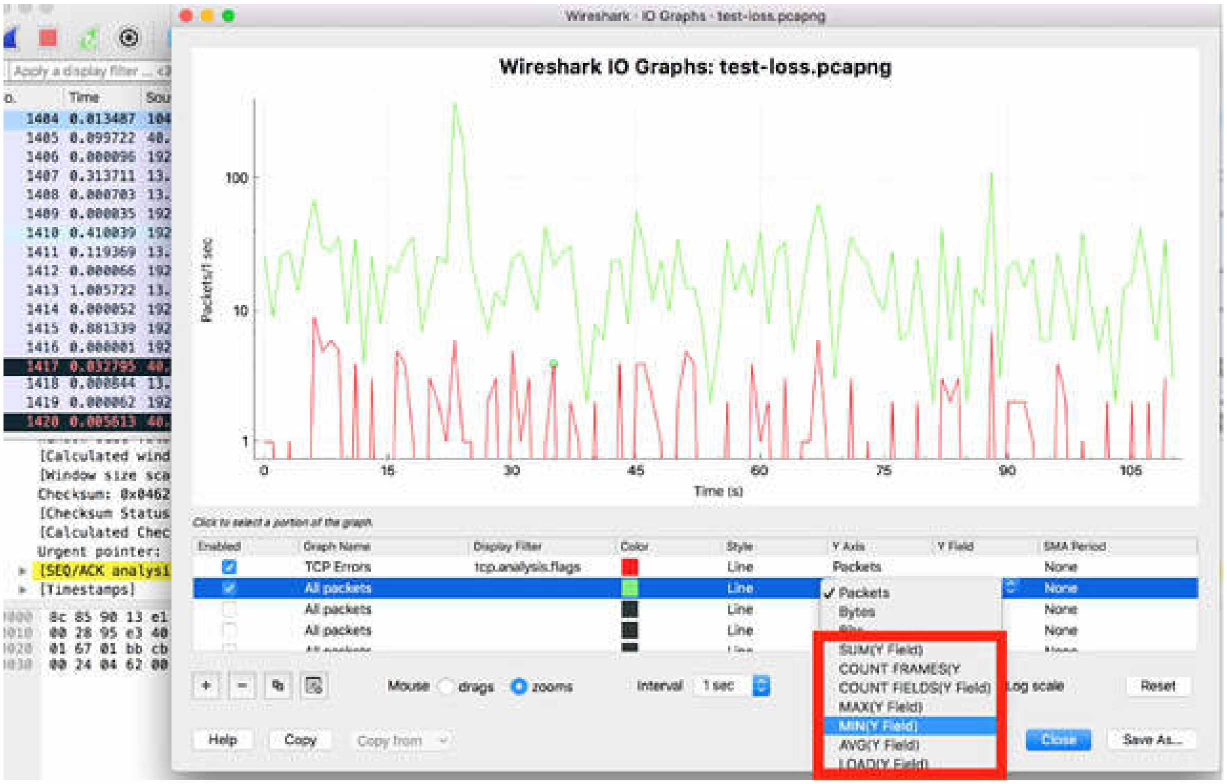
column. Capture the traffic for a few minutes.

Stop the capture and save the file.

To plot the packets per second rate of all the traffic in the saved trace file, on the main menu, select Statistics > I/O Graph, as shown in the figure below.



In the I/O Graphs dialog box, double-click the Y axis field to access the advanced I/O graphs.

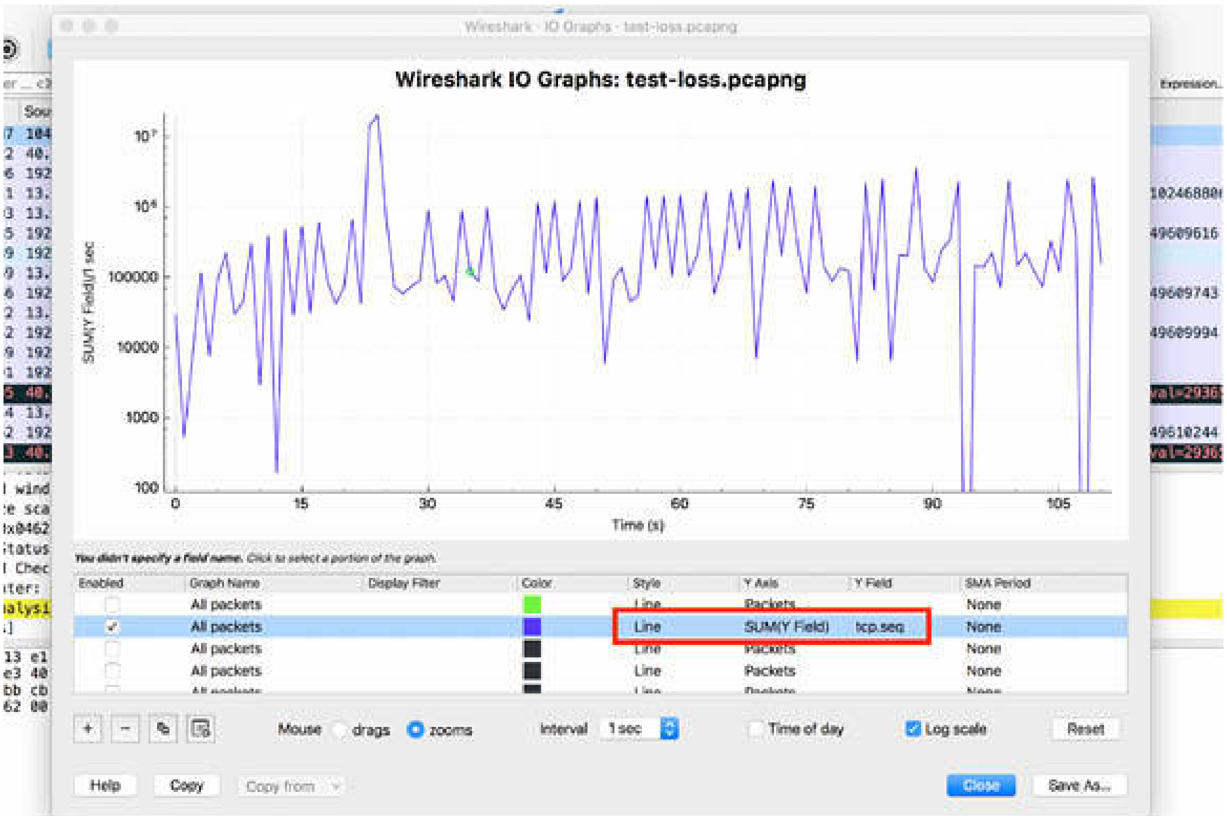


The options available in the drop-down list have the following calculation features:

- SUM(): Adds up and plots the value of a field for all instances seen during the tick interval.
- MIN(): Plots the minimum value seen in the field during the tick interval.
- AVG(): Plots the average value seen in the field during the tick interval.
- MAX(): Plots the maximum value seen in the field during the tick interval.
- COUNT(): Counts the number of occurrences of a field or characteristic seen during the tick interval.
- LOAD(): Measures only the response time fields.

Task 2:

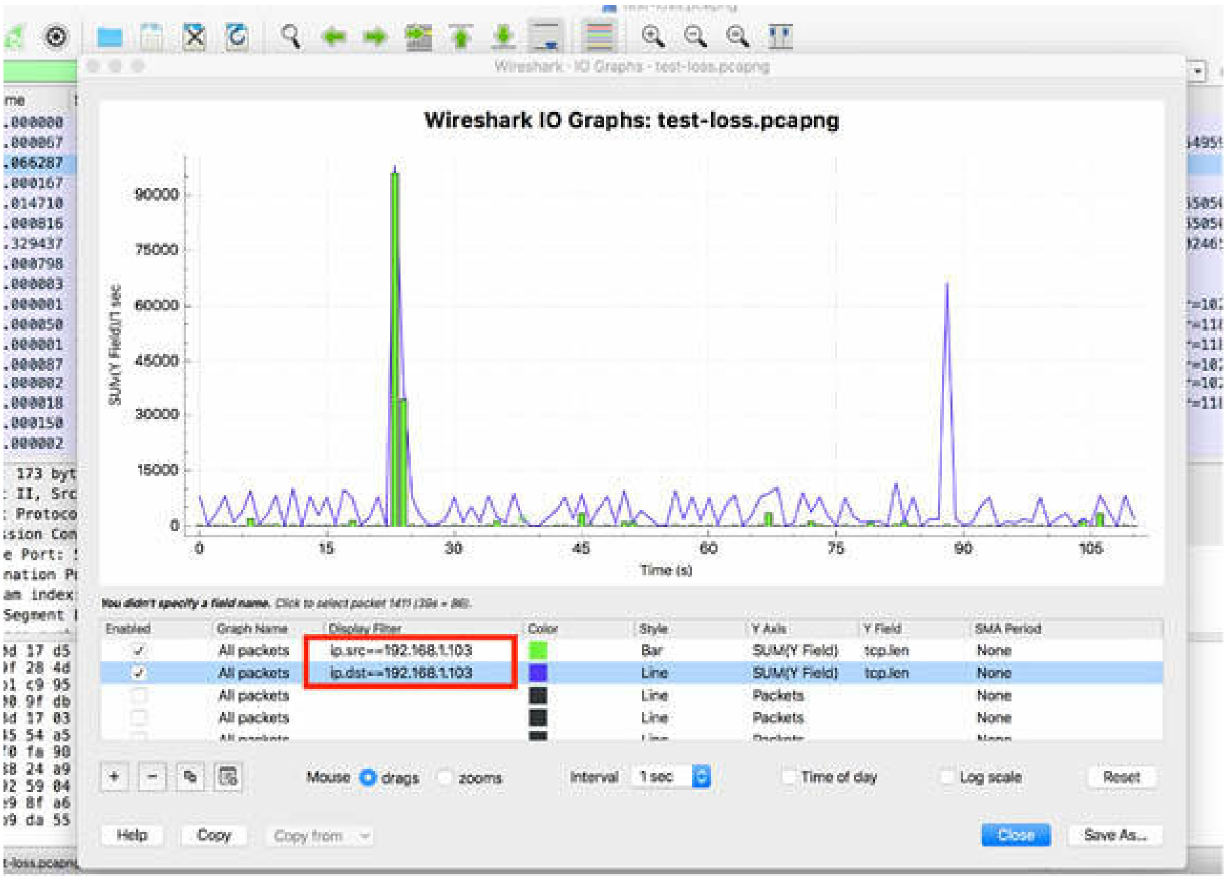
Open the I/O graph. On the plot channel, as shown in the figure below, select the SUM function for “Y Axis”. In “Y Field”, type tcp.seq to graph out the TCP sequence number as it increments.



In this example, there are two unusual drops around 100s and 110s. If you click on the graph at that specific point, you can jump to the packet in the Packet List pane and inspect whether a problem occurred in the data transfer.

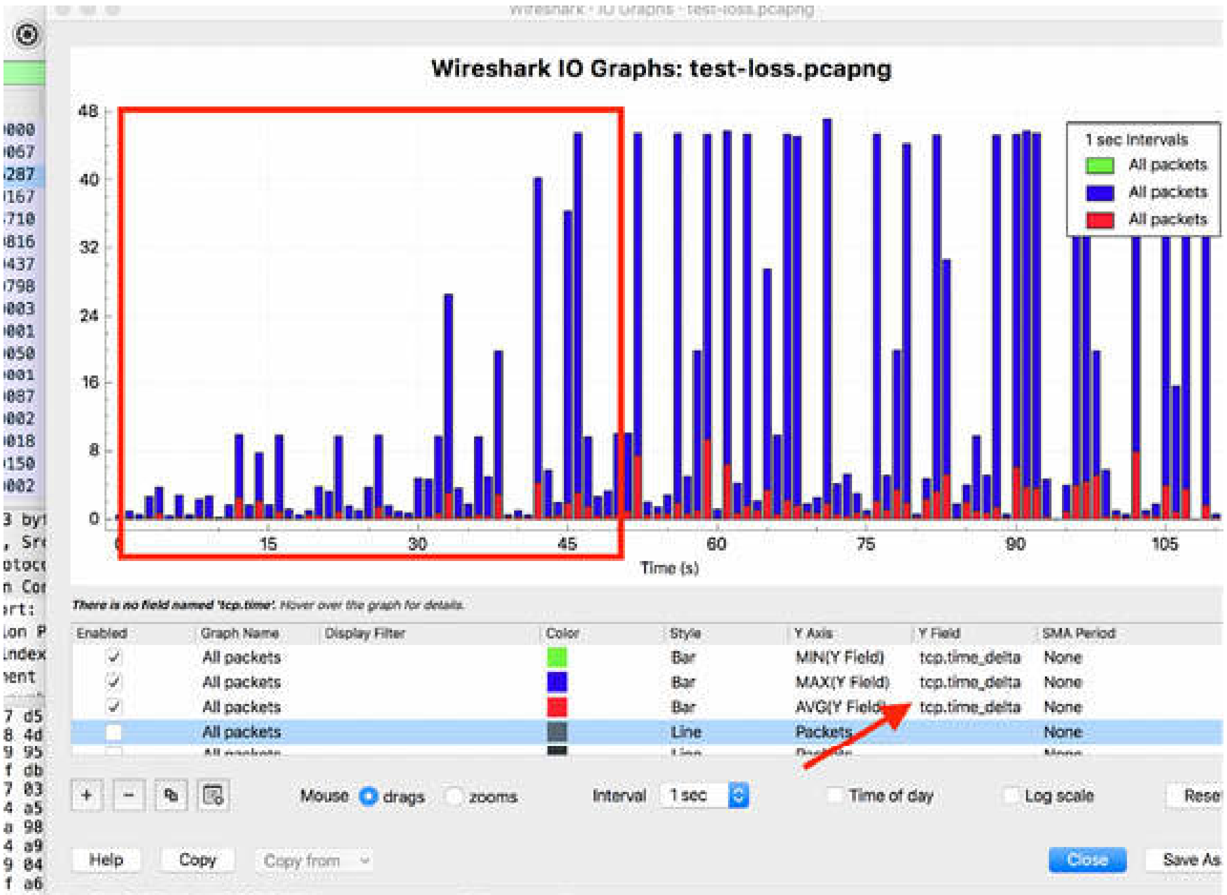
Task 3:

To plot the amount of TCP data (not including any data link, IP, or TCP headers) in your trace file, for the “Y axis”, select the SUM function for two channels. In the “Y field”, type tcp.len , as shown in the figure below. If you are interested in the amount of data crossing in a single direction of bidirectional flow of traffic, add a filter for IP source and destination addresses.



Task 4:
The MIN, AVG, and MAX options calculate and plot the minimum, average, and maximum of a field value. These calculations are very useful if you need to graph latency time between packets.

If you use the field tcp.time_delta , you can graph the minimum, average, and maximum time from the end of one packet to the end of the next packet in your trace file.



As shown in the figure above, there is a time region where an increase in the round-trip latency time is visible.

Task 5:

The COUNT calculation counts the occurrence of a characteristic, and it can be useful when graphing analysis flags such as tcp.analysis.retransmission or tcp.analysis.duplicate_ack .

In the “Y field”, enter tcp.analysis.retransmission , tcp.analysis.duplicate_ack , and tcp.analysis.lost_segment , as shown in the figure below. Apply the logarithmic scale because, in this case, you are comparing decimal number values on the graph.

The result is displayed in the figure below.

This advanced I/O graph illustrates the relationship between lost packets, duplicate ACKs, and retransmissions.

Notes:

Repeat the previous tasks by using all the functions provided by the advanced I/O graph functionality. Gain more confidence in using the options to make the best choice when you need to illustrate a problem in a trace file.

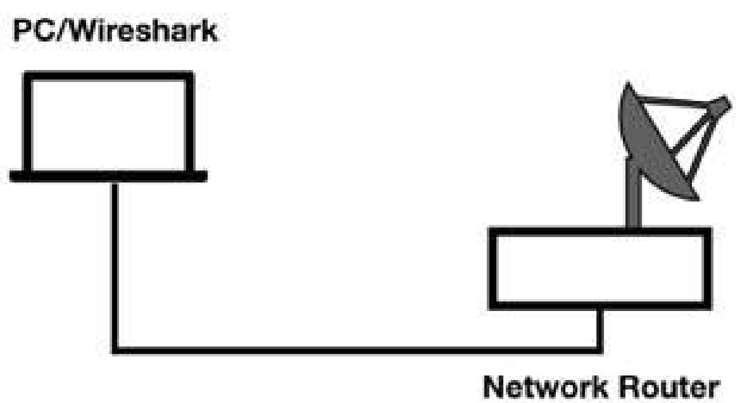
Lab 56. Traffic Trend Comparison in I/O Graphs

Lab Objective:
Learn how to compare traffic in I/O graphs.

Lab Purpose:
Learn how to simultaneously compare the I/O graphs of two trace files.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

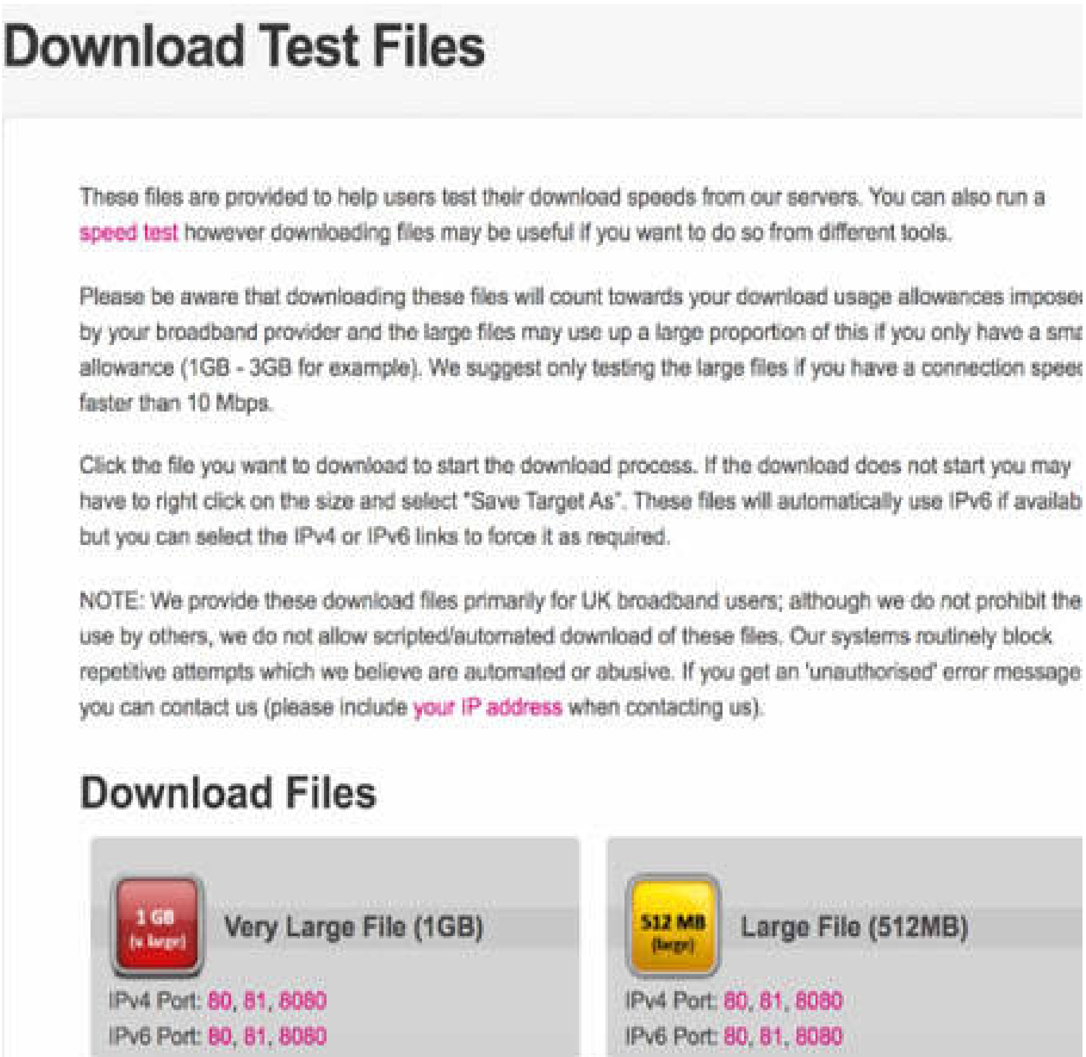


Lab Walkthrough:

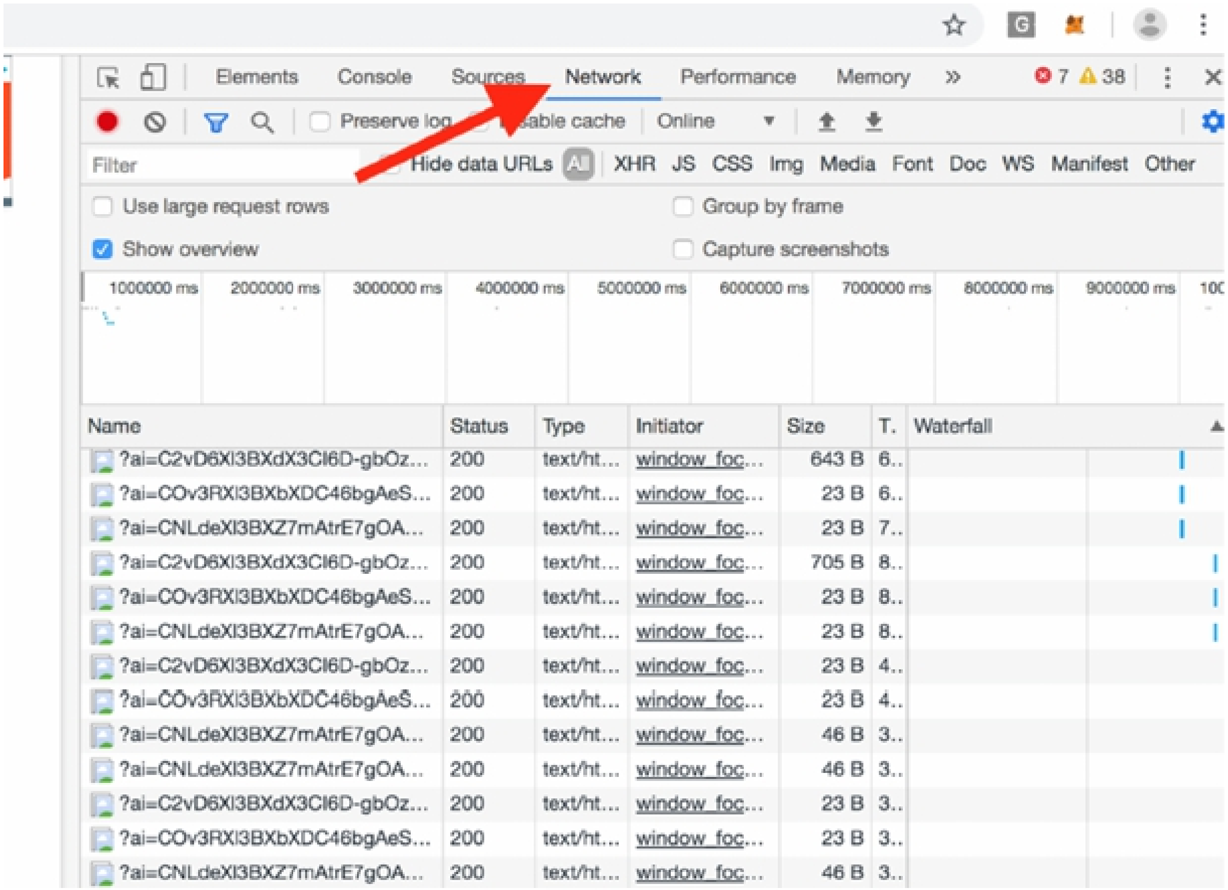
Task 1:

We will compare two trace files related to a file download in different network connections.

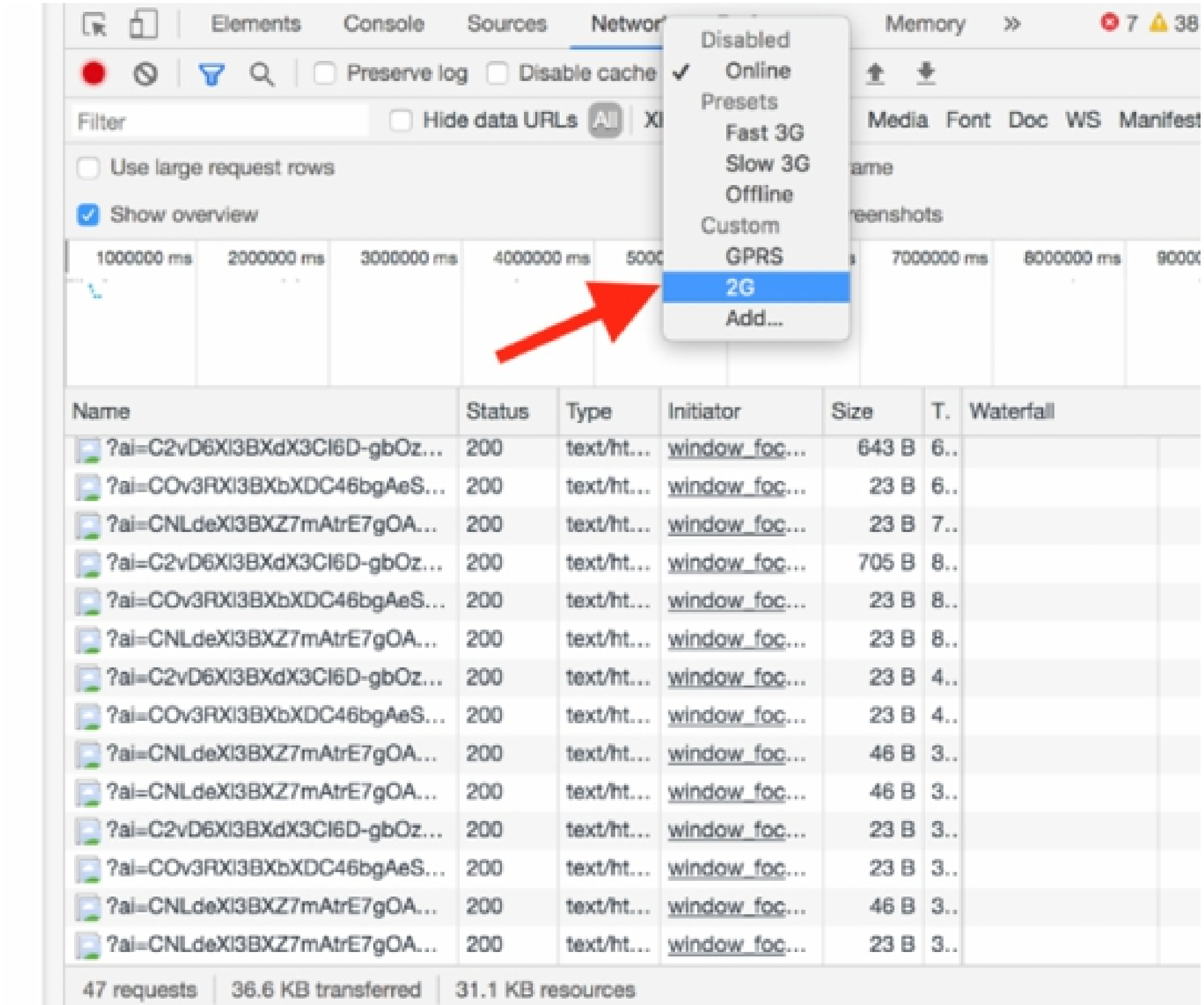
Open the Chrome web browser, and go to https://www.thinkbroadband.com/download . For this lab, we will download a test file of 5 Mb size.



To customize network performances, open “Chrome Developer Tools” and select the Network tab.



In the Online drop-down menu, select 2G to compare the network performance to a 2G connection.



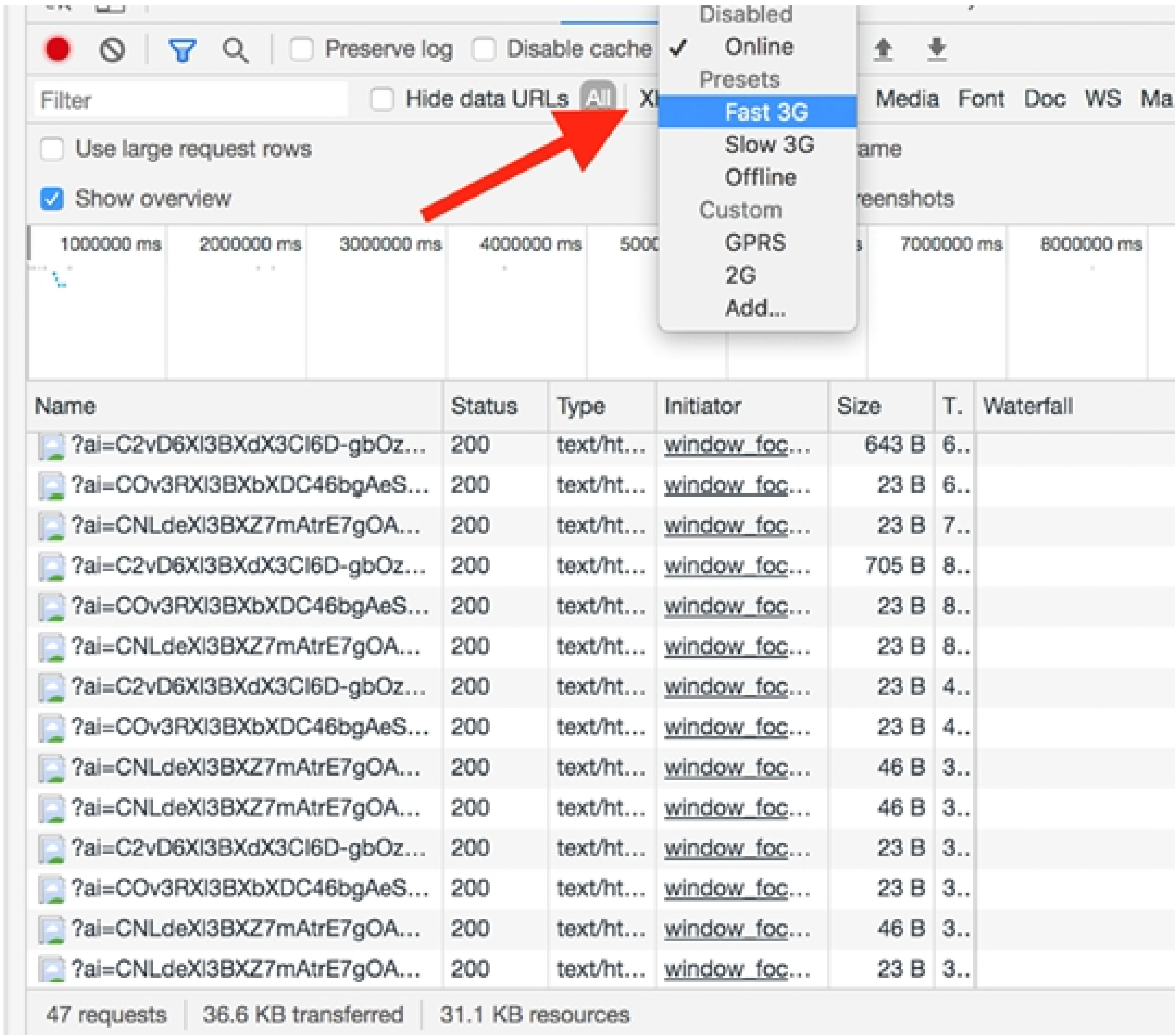
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

In Chrome, download a sample file of 5 Mb size from the opened website **https://www.thinkbroadband.com/download** .

Stop the capture and save the file as Test2G.pcapng.

Task 2:
To download the file again, in Chrome preferences, empty the stored cache.

Open "Chrome Developer Tools" and select the Network tab. From the Online drop-down menu, select Fast 3G to compare the network performance to a 3G connection.



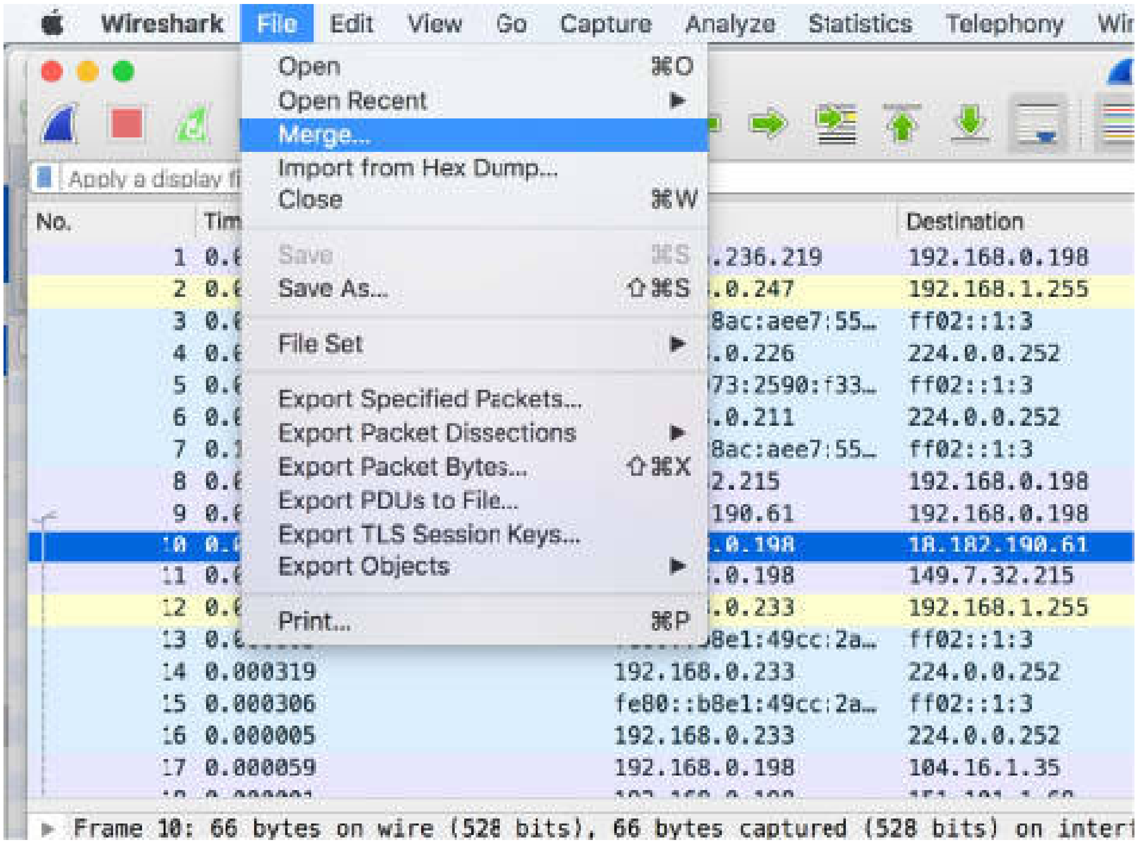
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

In Chrome, download a sample file of 5 Mb size from the opened website **https://www.thinkbroadband.com/download** .

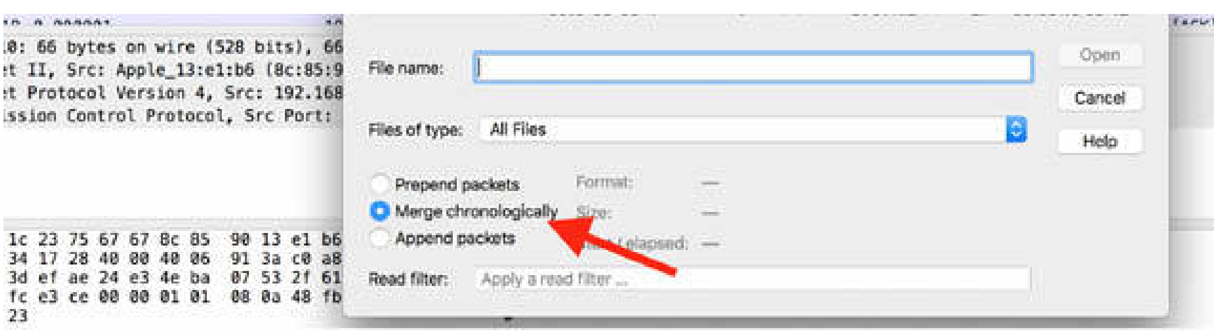
Stop the capture and save the file as Test3G.pcapng.

Task 3:

On the main menu, select File > Merge to merge the two trace files Test3G.pcapng and Test2G.pcapng.

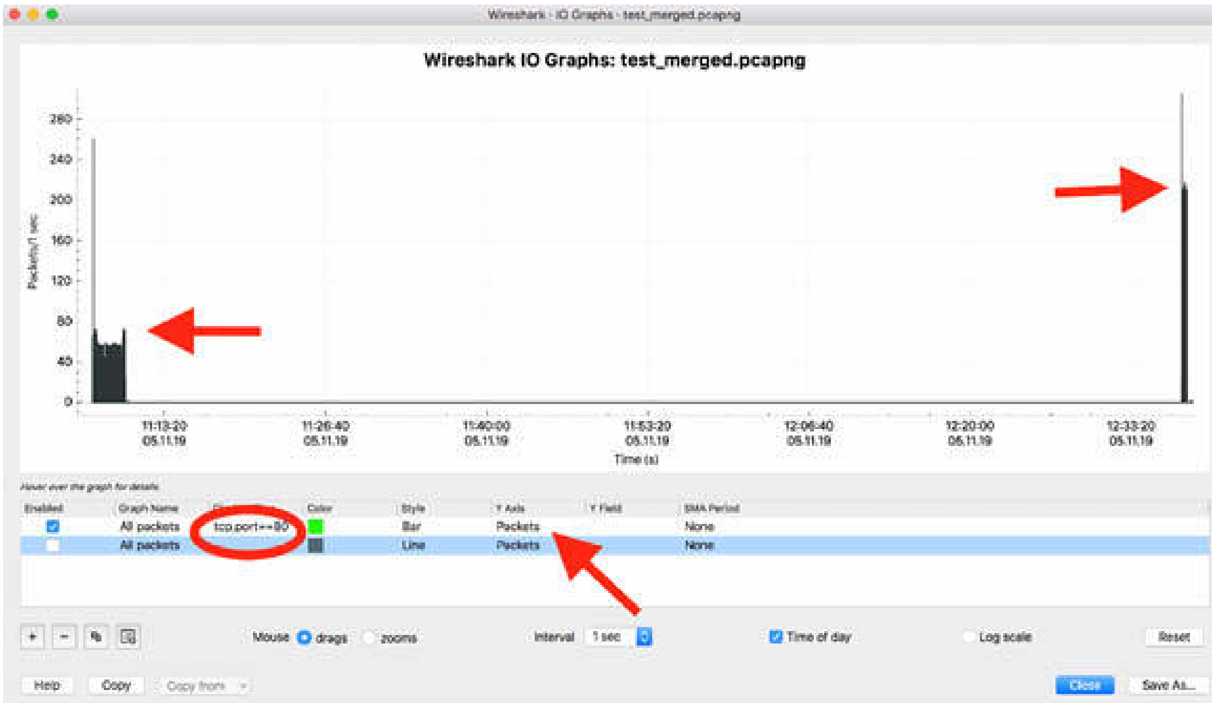


In the dialog box that appears, select the "Merge chronologically" option to preserve the order in the timestamp scale.



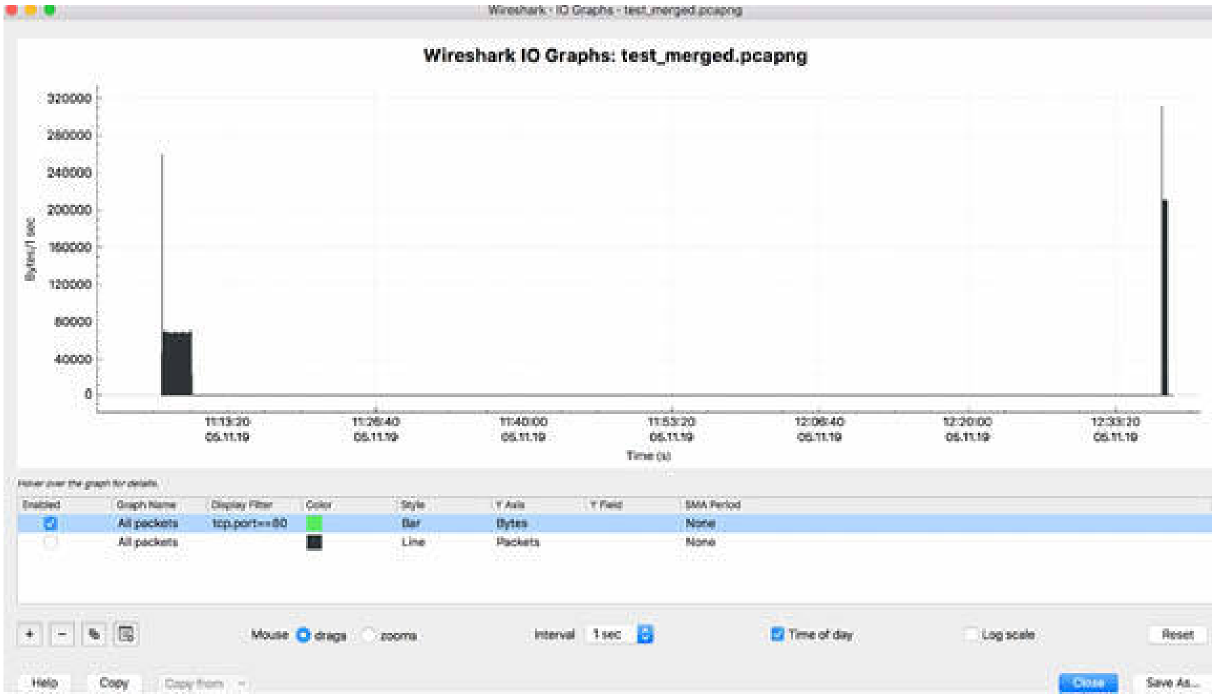
Save the generated file as test_merged.pcapng.

Task 4:
To plot the packets per second rate of all the traffic in the saved trace file, on the main menu, select Statistics > I/O Graph. In the “Y axis” field, select Packet, in the Style field, select Bar, and in the Display Filter field, enter tcp.port==80 . The result is shown in the figure below.



The trace related to the second download presents about four times the number of packets per second than the first download. This is as expected considering the different use cases for the network status.

If you change the “Y axis” to Bytes in the plot, you will better understand that the duration of the data transfer is greater in the first download because the Bytes/sec are three times in the second download.



Notes:

To compare two traffic flows side-by-side, repeat the following four-step process:

1. Examine the time difference between the trace files.
2. If necessary, change the timestamp of one of the trace file so that it will plot directly in front of or behind the other trace file.
3. Merge the two trace files.
4. Open the merged trace file and generate an I/O graph.

To gain the necessary confidence in manipulating network files and generating comparison graphs, repeat the steps above to compare different trace files taken for the same process but in different use cases of the network.

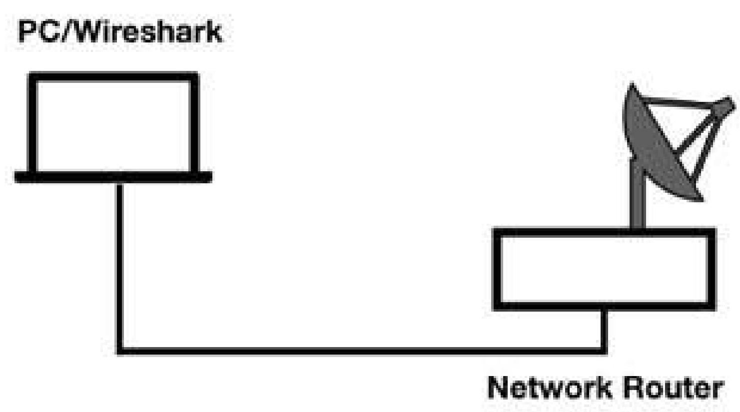
Lab 57. Round Trip Time and Throughput Rates

Lab Objective:
Learn how to create a Round Trip Time graph and view trends related to the traffic flow.

Lab Purpose:
Learn how to create a graph for Round Trip Time and generate a graph related to the TCP Throughput rates.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



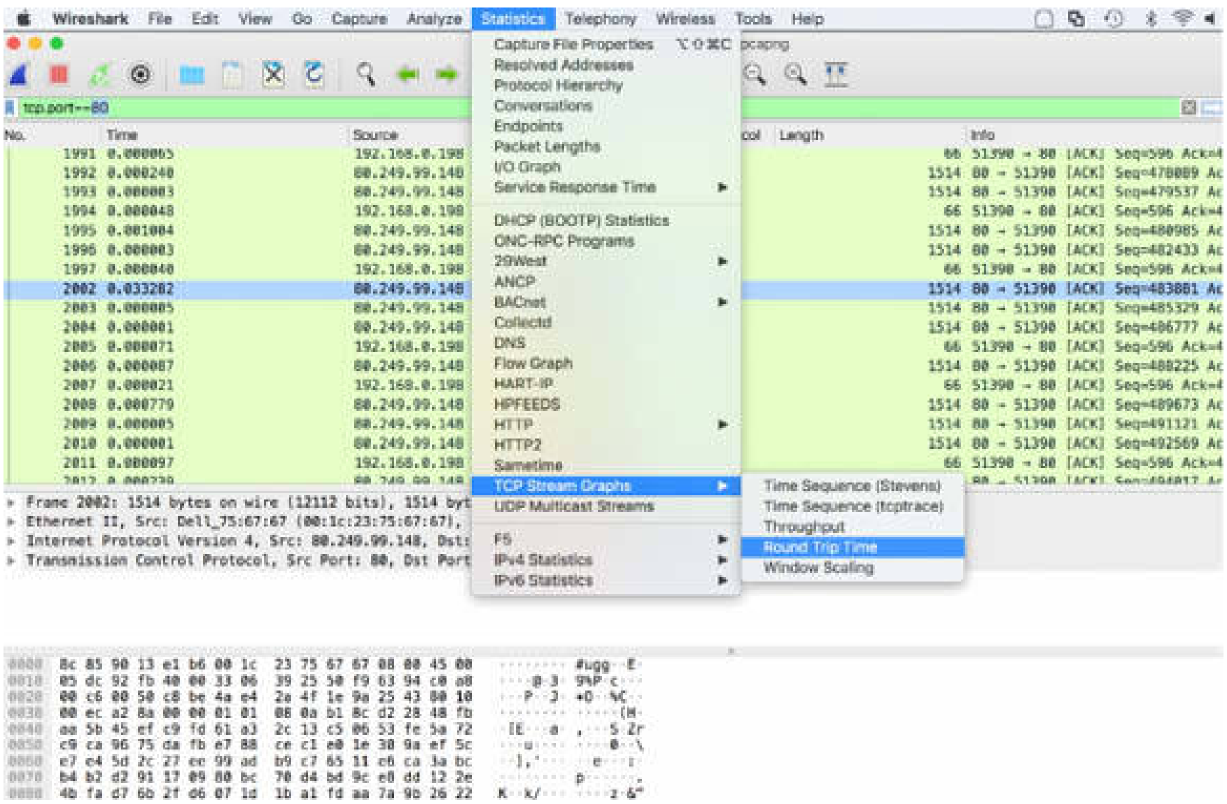
Lab Walkthrough:

Task 1:

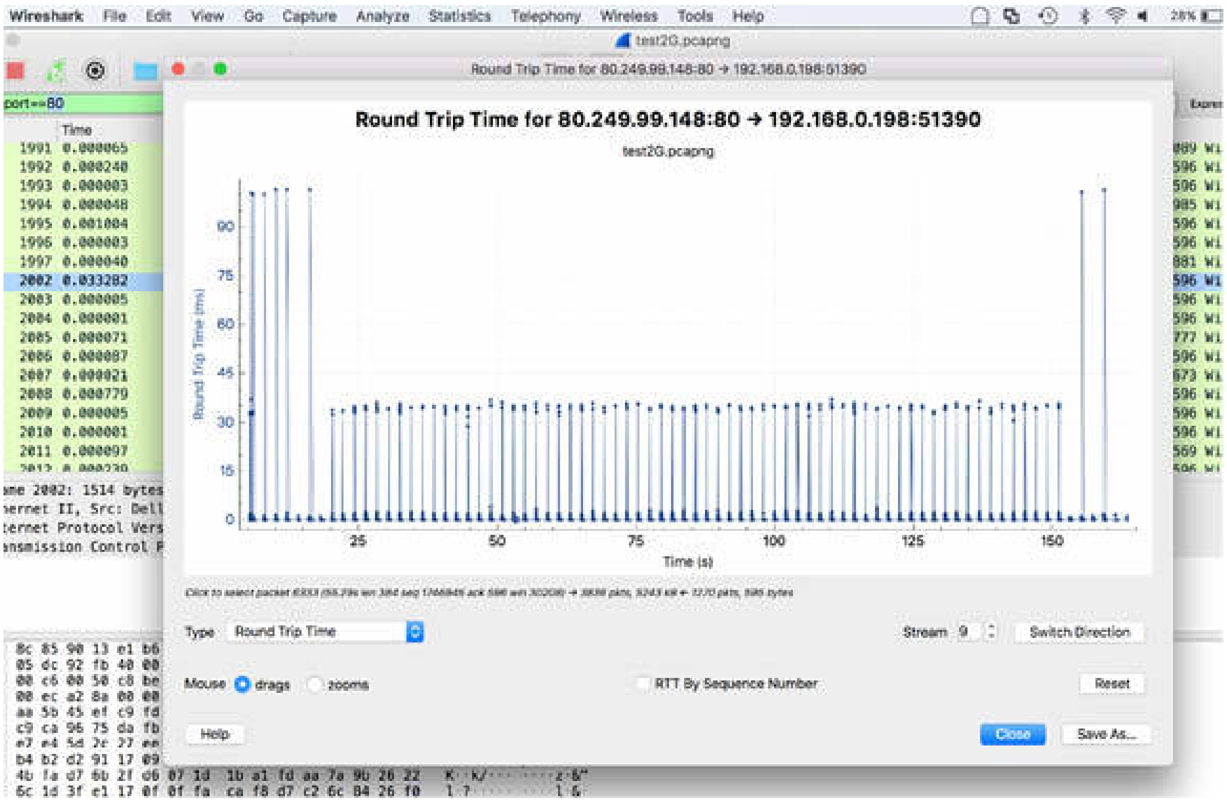
Use the capture trace file saved in the previous lab (test2G.pcapng and test3G.pcapng).

In Wireshark, open the trace file test2G.pcapng. In the filter toolbar, enter tcp.port==80 .

To depict the round trip time from a data packet to the corresponding ACK packet, on the main menu, select Statistics > TCP Stream Graphs > Round Trip Time, as shown in the figure below.

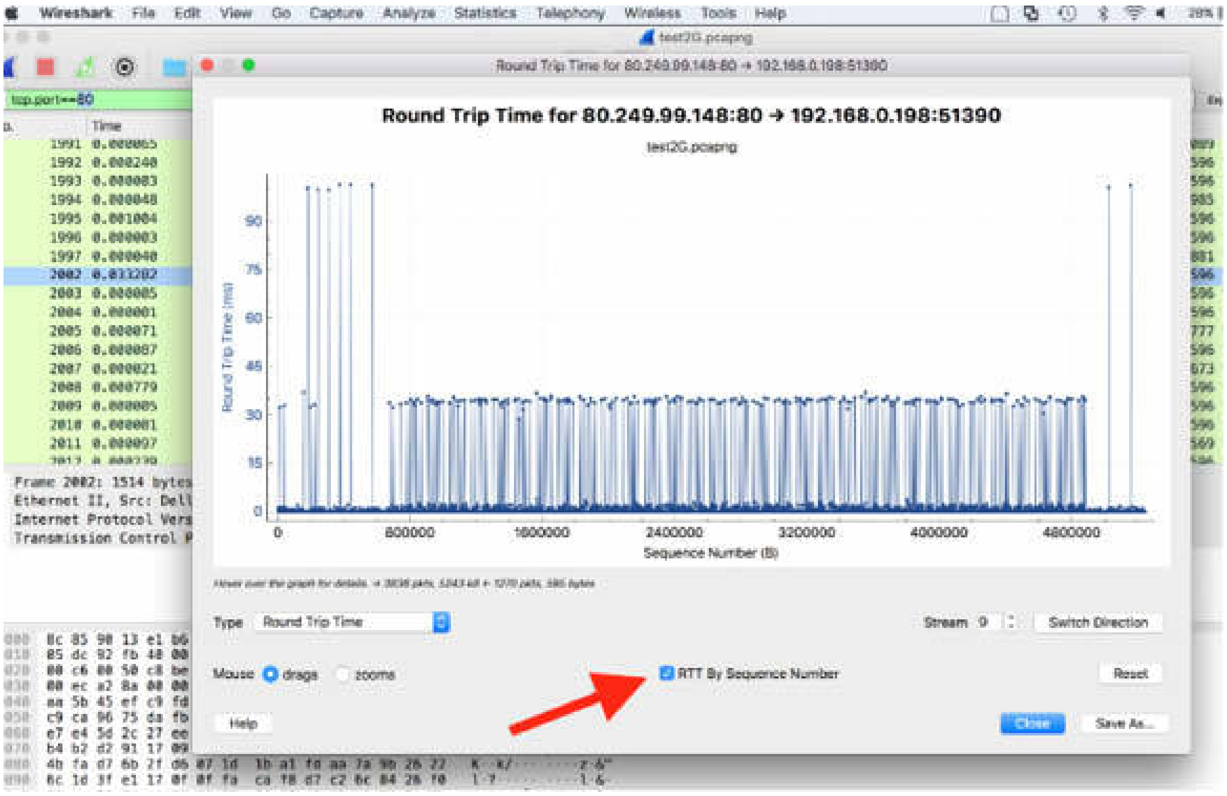


The graph, shown in the figure below, is generated. Y axis is created based on the highest round trip latency time. Latency times are calculated as the time between a TCP data packet and the related acknowledgment.



Considering that the trace file is related to a data transfer and the Y axis defines the round trip time in milliseconds and the X axis defines the time in seconds, you can observe that the latency time is quite constant and does not present high values.

To display TCP sequence number in the X axis, enable the "RTT By Sequence Number" check box, as shown in the figure below.

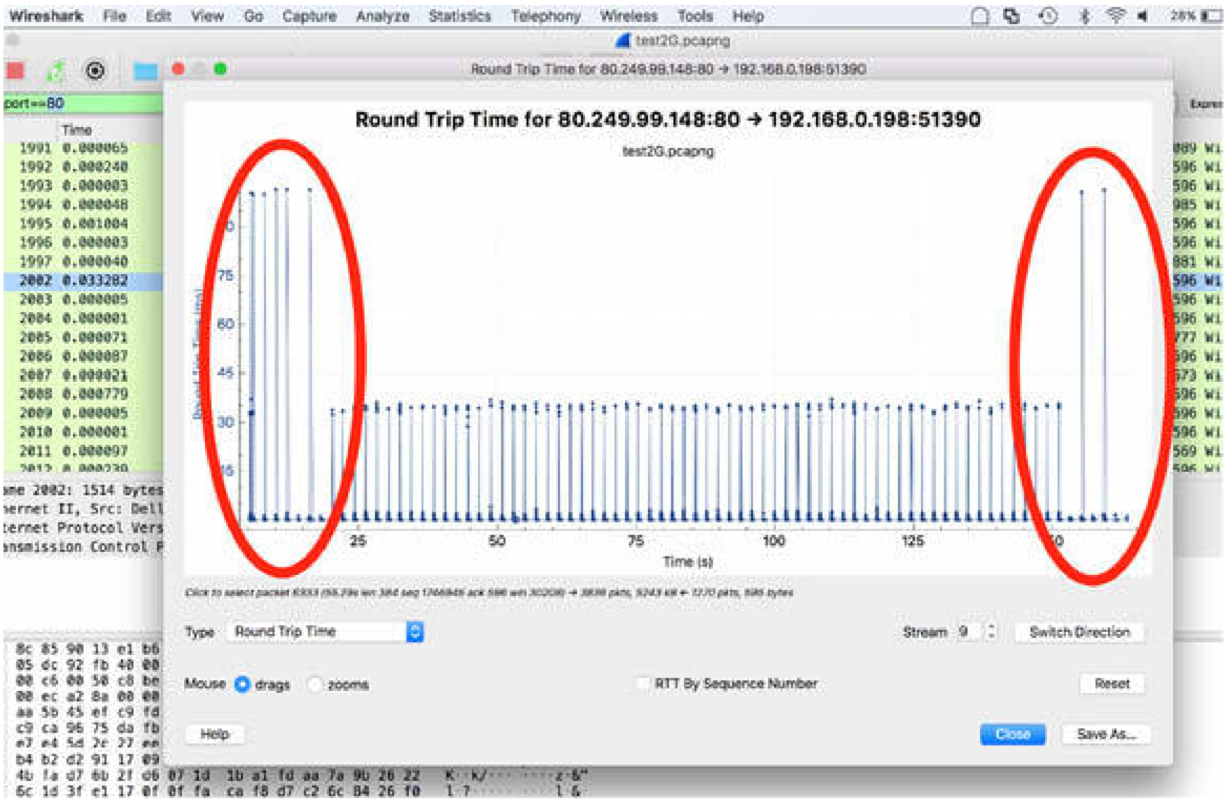


You can again see that the round trip time is quite constant.

If you open a round trip time graph and don't see anything plotted, it could be because you are looking at the wrong direction—the direction opposite to the data flow. To resolve such an issue, select a data packet traveling in the direction of data flow, and again load the graph.

Task 2:

To determine what happens during the points when you notice vertical stripes, click on one of the plot points. Wireshark jumps to that location in the trace file to enable you to investigate further.

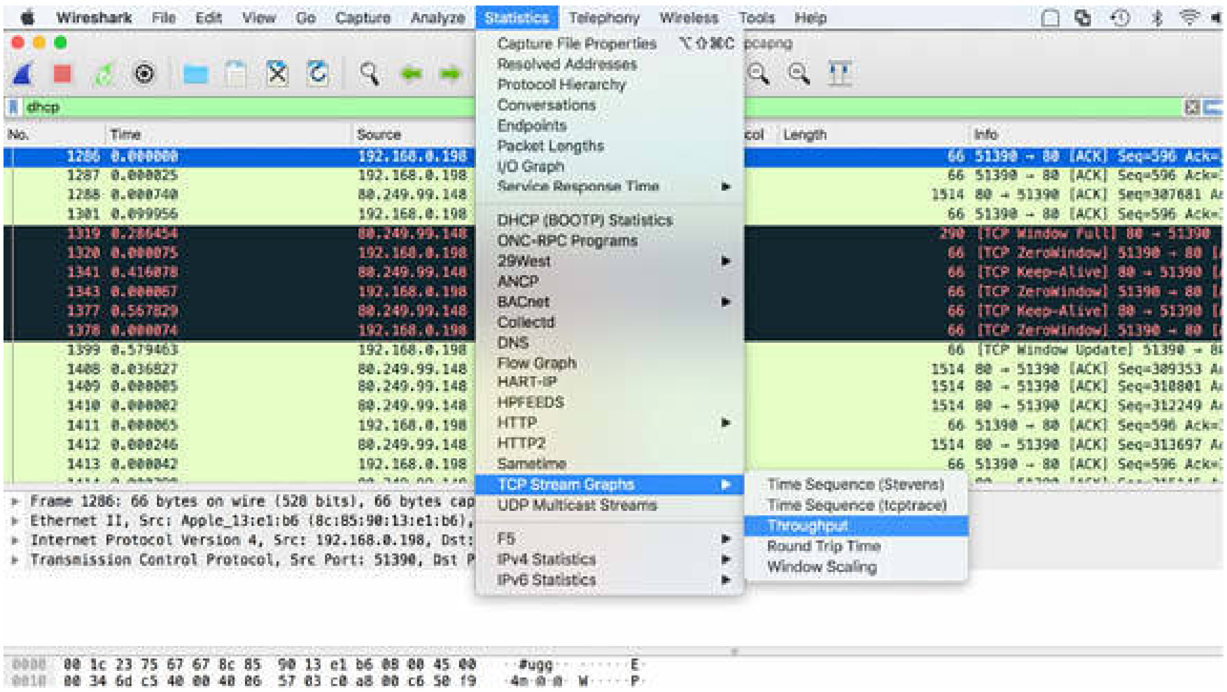


Vertical stripes can happen when the packet loss occurs and a high number of duplicate ACKs are sent. Another scenario for vertical stripes is when data is queued along a path, and then it is suddenly forwarded through the queuing device.

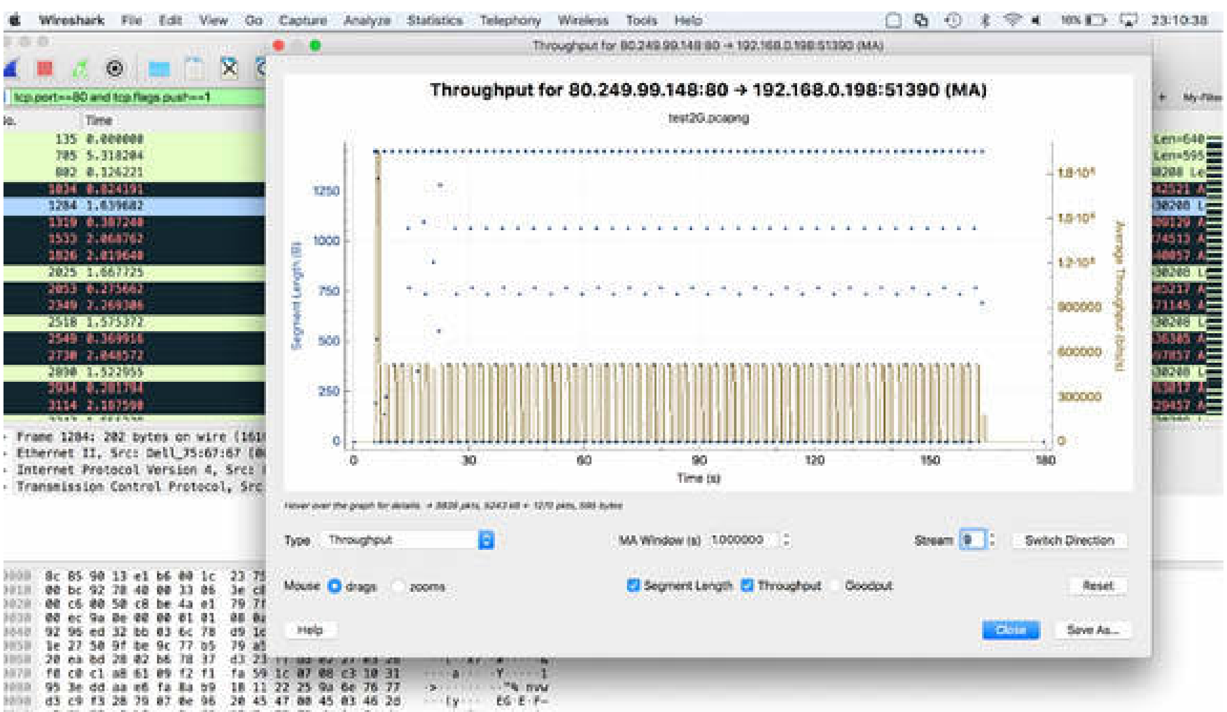
Task 3:

In the filter toolbar, enter tcp.port==80 and tcp.flags.push==1 to display only the TCP push messages.

On the main menu, select Statistics > TCP Stream Graphs > Throughput, as shown in the figure below, to view trends related to the traffic flow.



The TCP Throughput graph is closely related to the I/O Graph, but plots are done only with dots.



Again, if you do not see anything plotted when you open a Throughput graph, you might be looking at the wrong side of the communication.

You can see how this graph is similar to the round trip time graph because both are related to flow information.

Because the TCP Throughput graphs are created based on the packet selection in the Packet List pane, you can easily create these graphs for any conversation in the trace file.

Notes:
Repeat the previous steps to analyze the trace file test3G.pcapng. To gain confidence in creating graphs, recreate all the graphs.

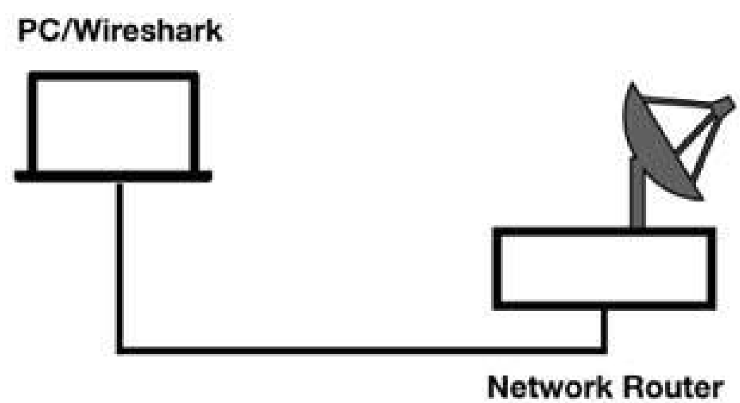
Lab 58. TCP Sequence Numbers

Lab Objective:
Learn how to create Time Sequence graphs.

Lab Purpose:
Learn how to create a graph representing TCP sequence numbers over time.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

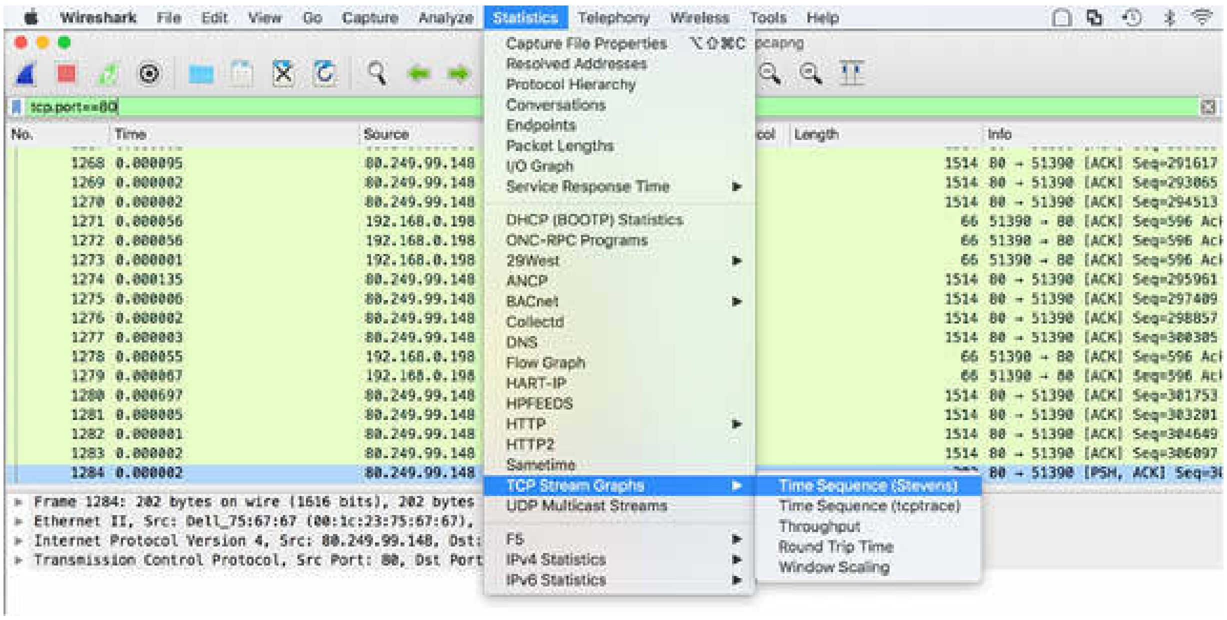


Lab Walkthrough:

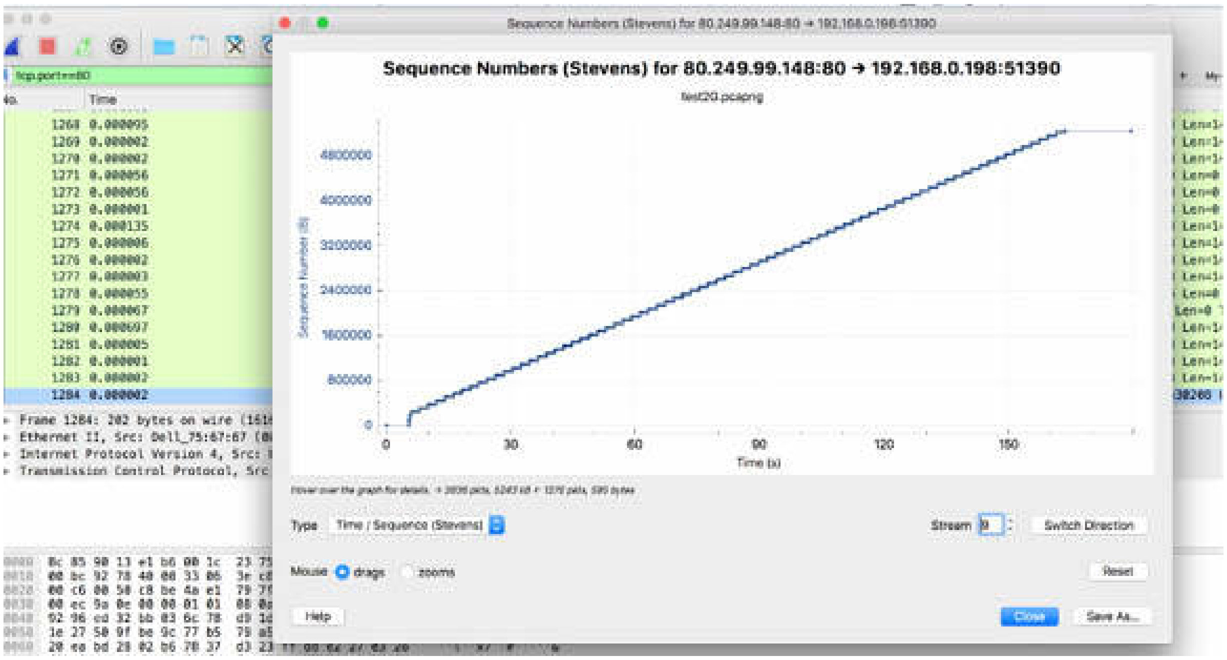
Task 1:
Use the capture trace file used in the previous lab (test2G.pcapng and test3G.pcapng).

In Wireshark, open the trace file test2G.pcapng. In the filter toolbar, enter tcp.port==80 .

To view the Time Sequence graph, on the main menu, select Statistics > TCP Stream Graphs > Time Sequence (Stevens), as shown in the figure below.

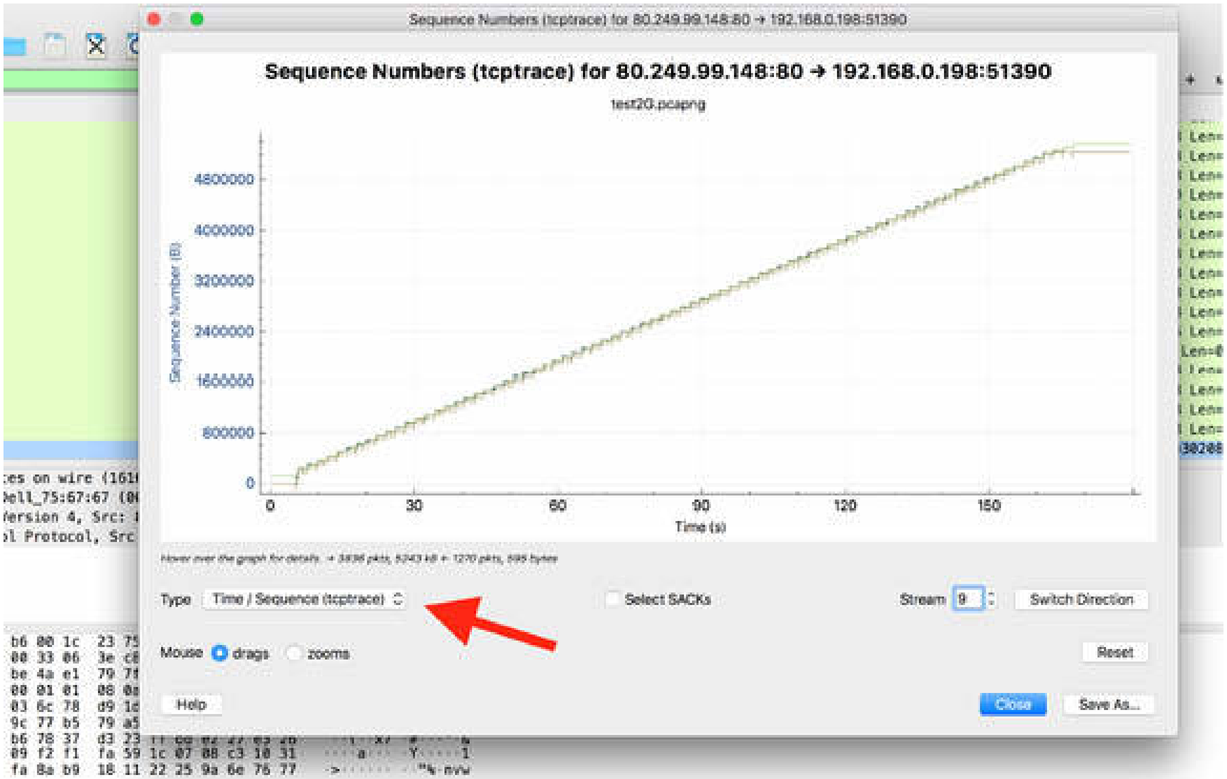


The graph, shown in the figure below, is generated. The Time Sequence graph represents TCP-based traffic. In an ideal situation, like the one shown in the figure below, the graph plots should run from the lower-left corner to the upper-right corner in a smooth diagonal line, indicating that the sequence number is incrementing linearly.



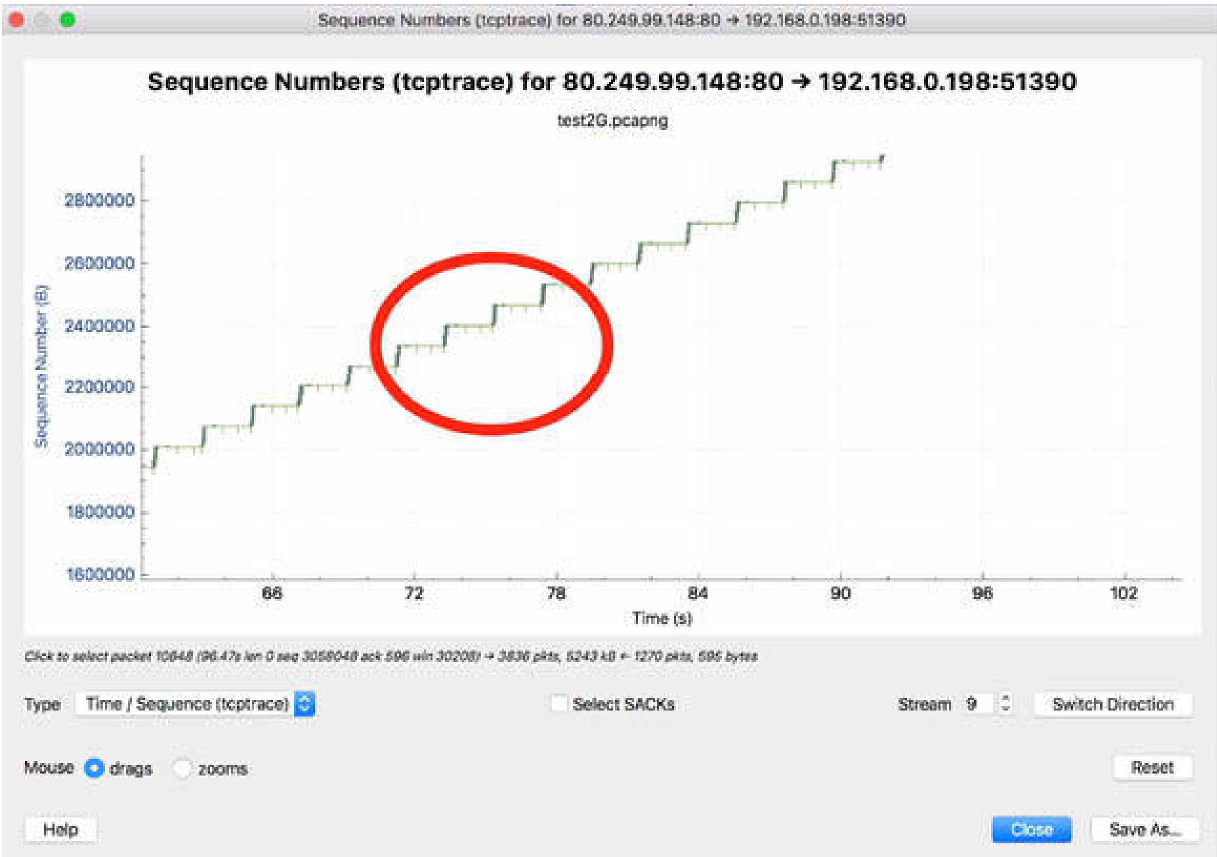
Task 2:

You can also use the Time/Sequence graph (tcptrace) and verify if some more information is available in the graph. In the Type list, select Time/ Sequence (tcptrace), as shown in the figure below.



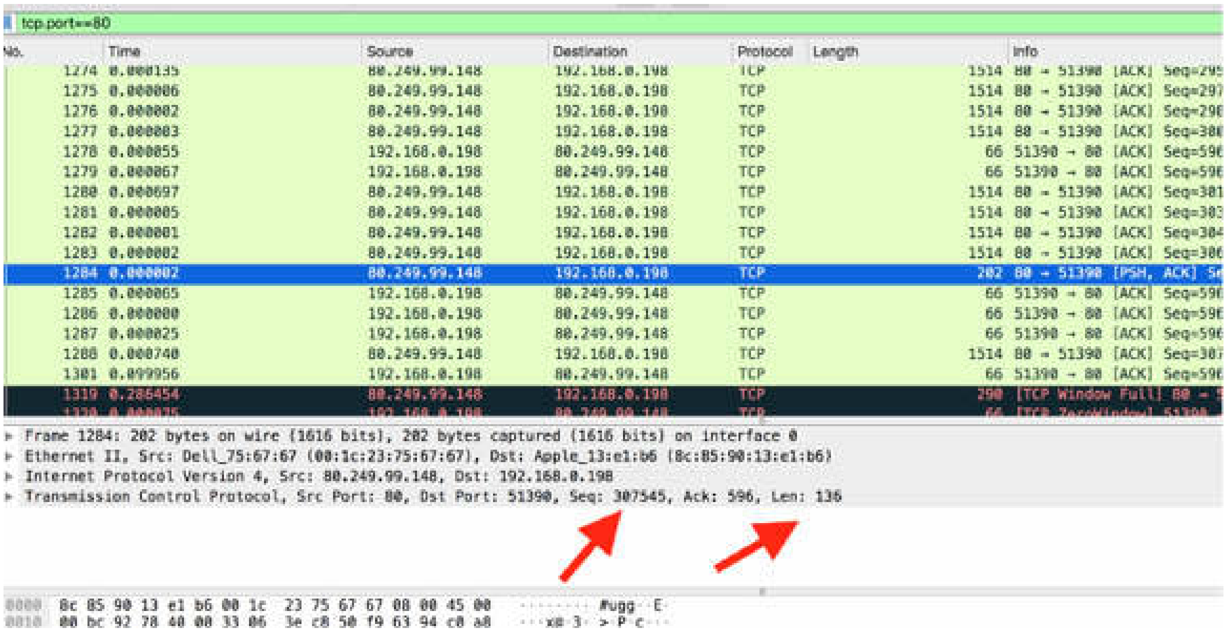
TCP headers contain a "Sequence number" field (as you experimented in previous labs) that increments by the number of bytes sent during data transfer. If a TCP header sequence number is 1000 and there are 200 bytes of data in the packet, the TCP header from this source should contain the sequence number 1200. If the next packet again contains the sequence number 1000, this is a retransmission packet. If the next TCP packet contains the sequence number 1400, a segment must have been lost.

TCP segments are plotted in an "I bar" format. Longer "I bars" contain more data. If you zoom in the previous graph, you can see that each bar is similar in length to others, meaning that each segment contains more or less the same amount of data.



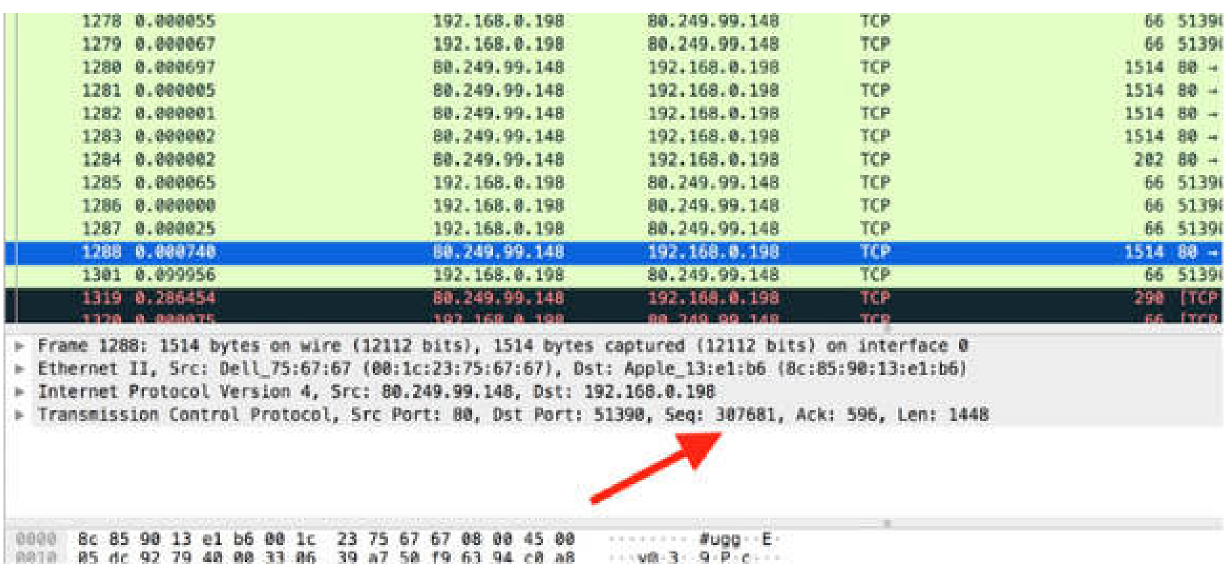
Task 3:

To view an example of the ideal TCP sequence number incrementation, in the Packet List pane, select a packet from a predefined source IP (in this case, IP address 80.249.99.148). Take note of the sequence number and the length of the packet, as shown in the figure below.



As shown in the figure above, the sequence number is 307545 and the length is 136. In an ideal situation, if no packet loss occurs and there is no retransmission, the expected sequence number will be 307545 + 136 = 307681.

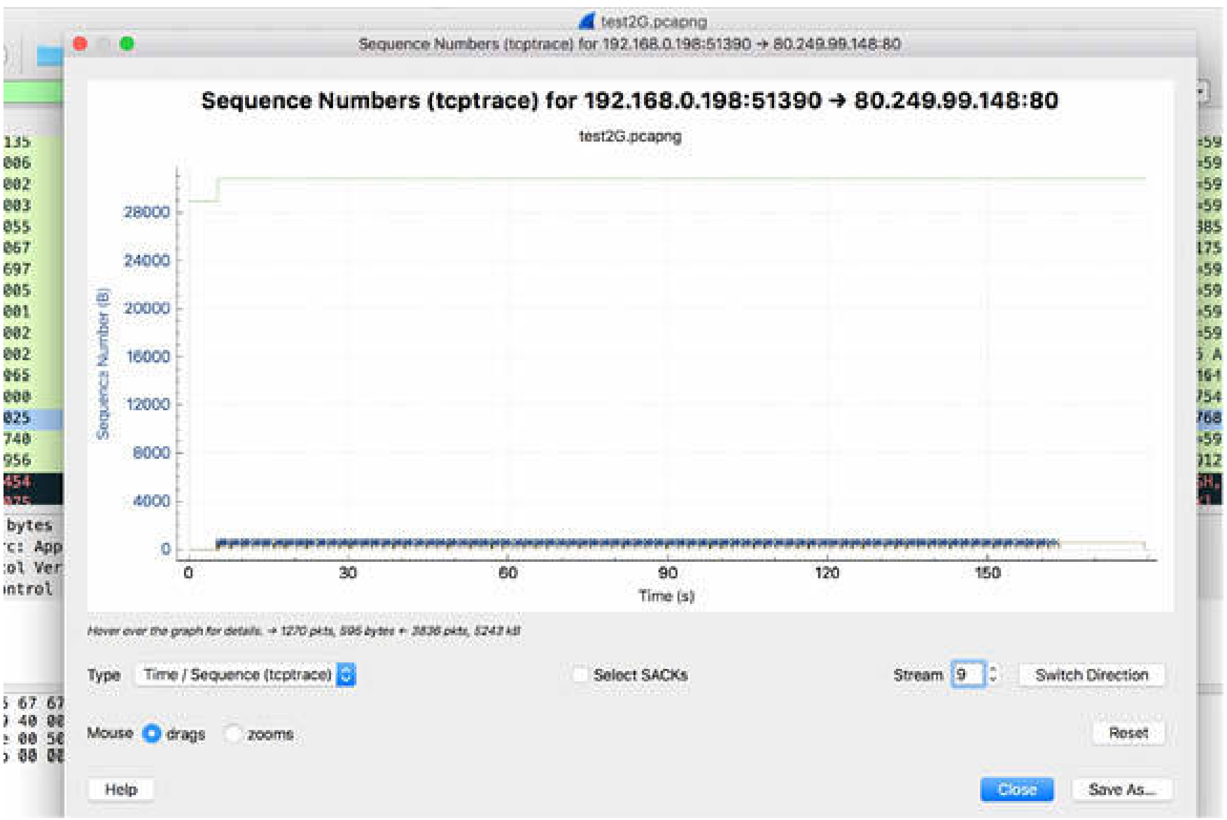
In the Packet List pane, if you click the subsequent TCP packet from the same IP source, you can verify that the sequence number has the expected value (packet #1288 in the figure below).



The TCP Time Sequence graph's data moves in one direction. Therefore, in the Packet List pane, make sure that you select a packet that contains data or that is traveling in the direction of data flow.

If the graph appears empty, look at the title bar to ensure that you are examining the right direction. To resolve such an issue, select a data packet traveling in the direction of data flow and again load the graph.

The figure below shows an empty graph when you select a packet in the opposite direction. The graph appears empty because there is no data flow in that direction.



Notes:
Repeat the previous steps to analyze the trace file test3G.pcapng. To gain confidence in creating and analyzing graphs, recreate all the graphs.

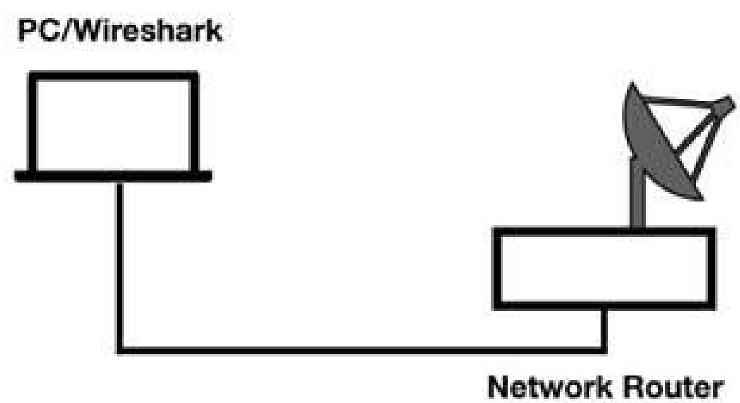
Lab 59. TCP Window Size Issues

Lab Objective:
Learn how to interpret the TCP window size issues.

Lab Purpose:
Learn how to create a graph representing the TCP window.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

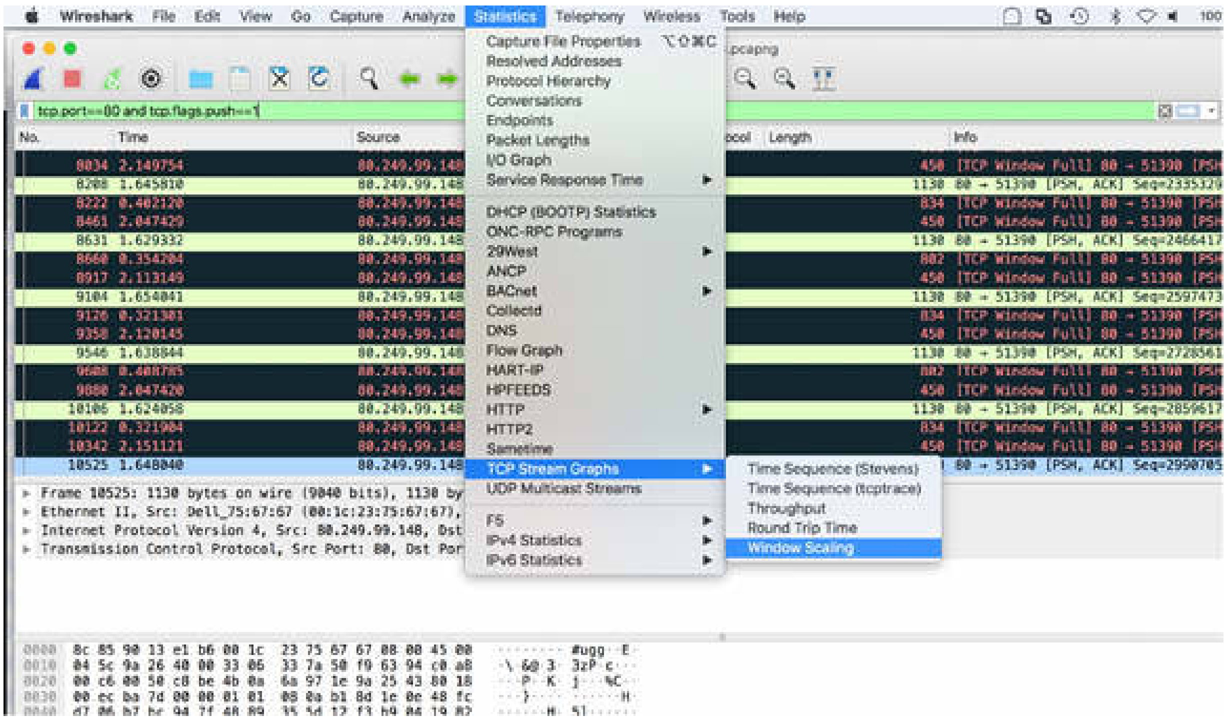


Lab Walkthrough:

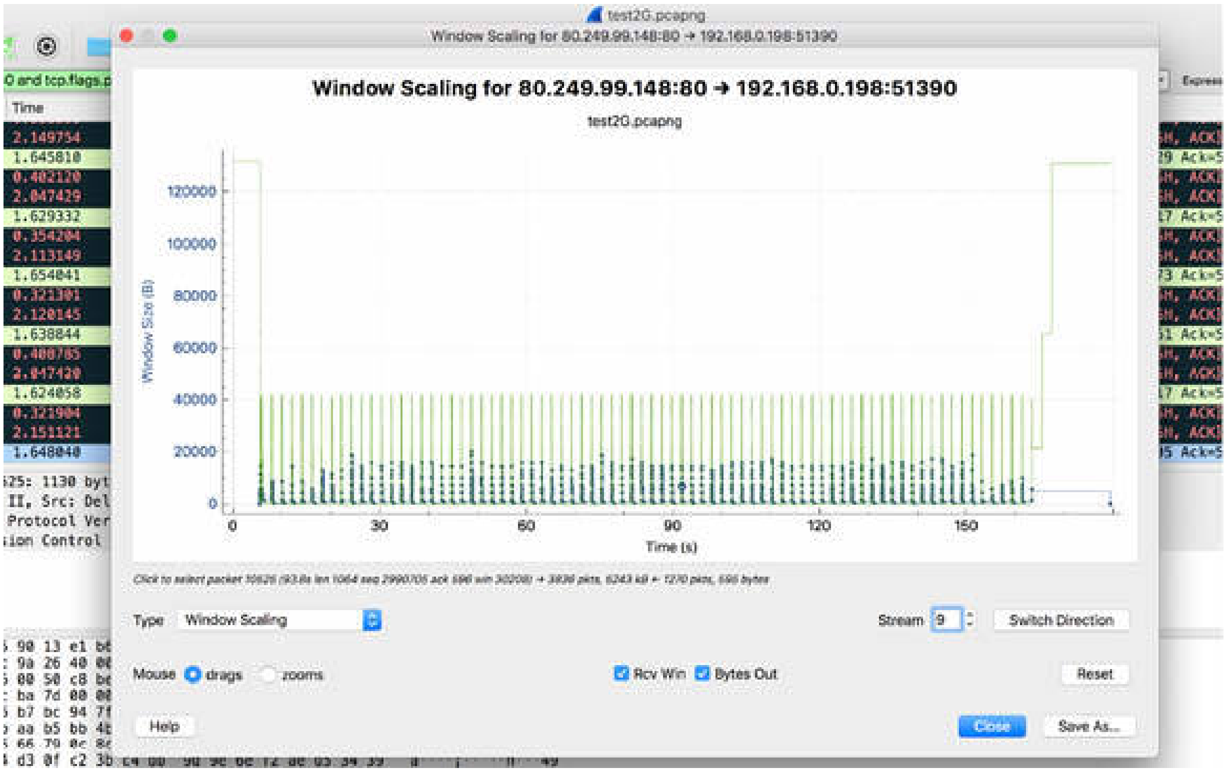
Task 1:
Use the capture trace files used in the previous lab (test2G.pcapng and test3G.pcapng).

In Wireshark, open the trace file test2G.pcapng. In the filter toolbar, enter tcp.port==80 and tcp.flags.push==1 to select only the packets containing data.

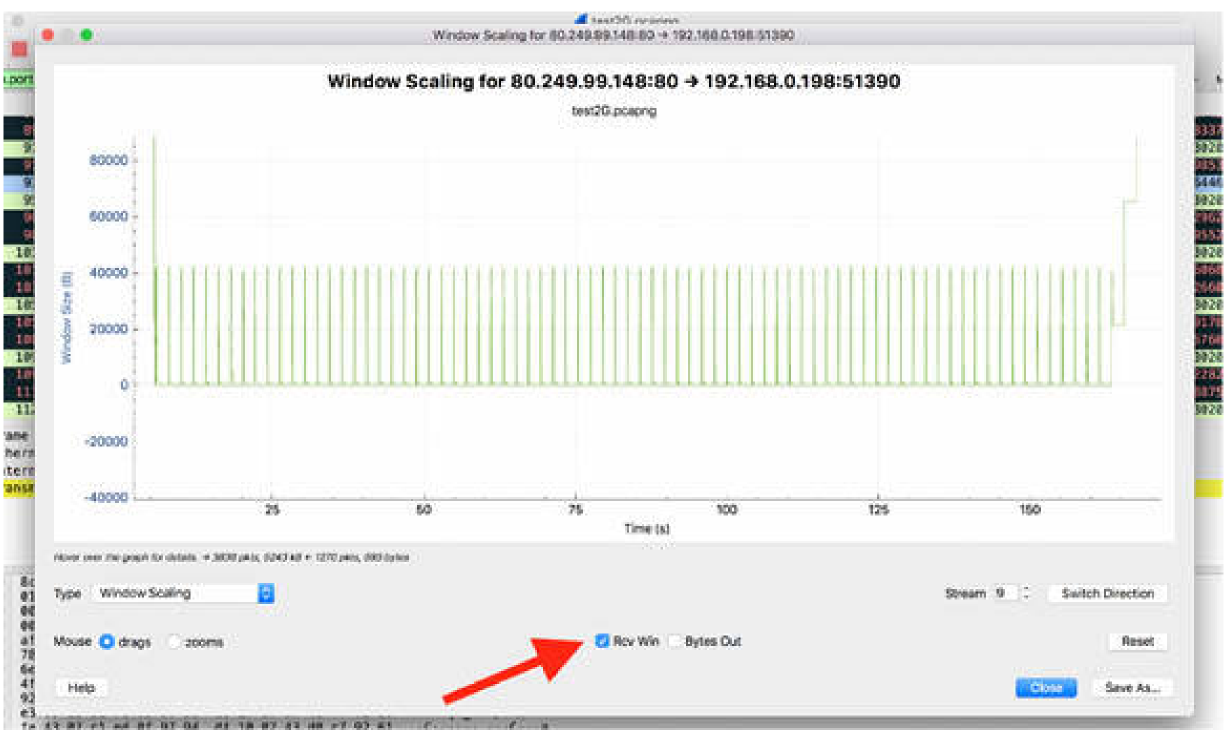
To view the TCP window graph, on the main menu, select Statistics > TCP Stream Graphs > Window Scaling, as shown in the figure below.



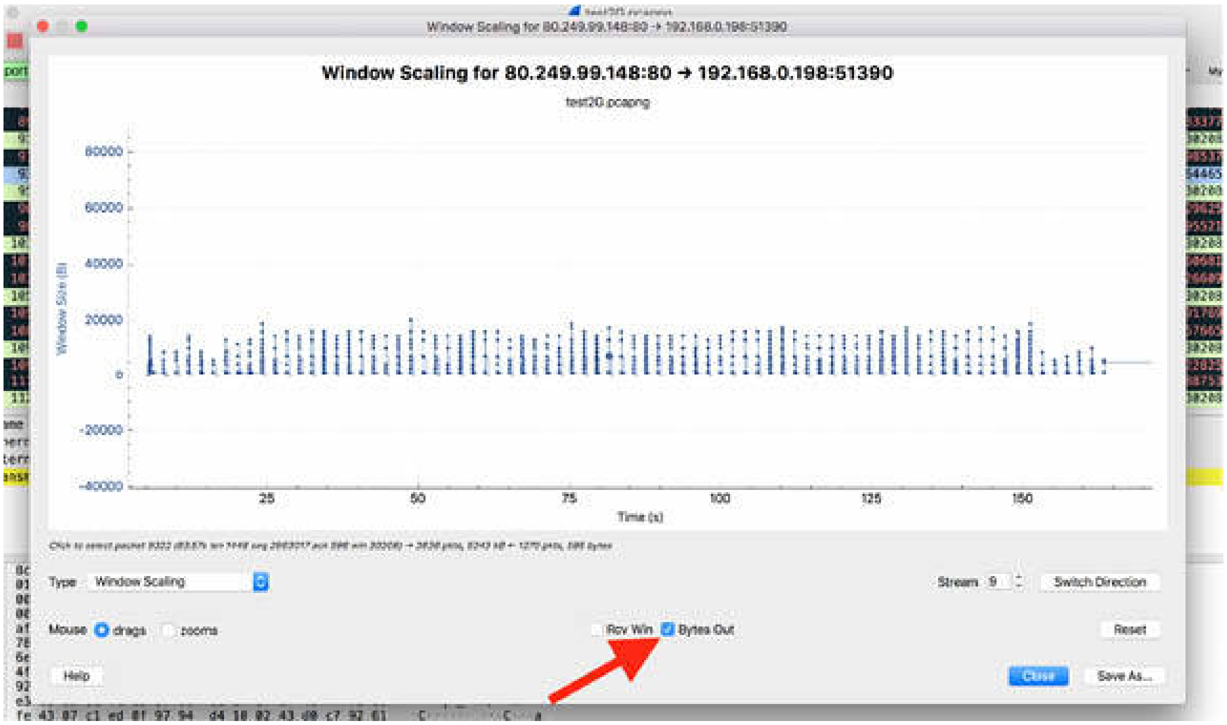
The graph, shown in the figure below, is generated. The TCP window size advertises the amount of buffer space available. When the TCP window size green line moves closer to the plotted “I bars”, the receive window size decreases. When they touch, the receiver has indicated that its TCP window size is zero, and no more data can be received. The figure below shows the resultant graph.



To clearly see the periodic window size, zoom in, and select the “Rcv Win” check box, as shown in the figure below.



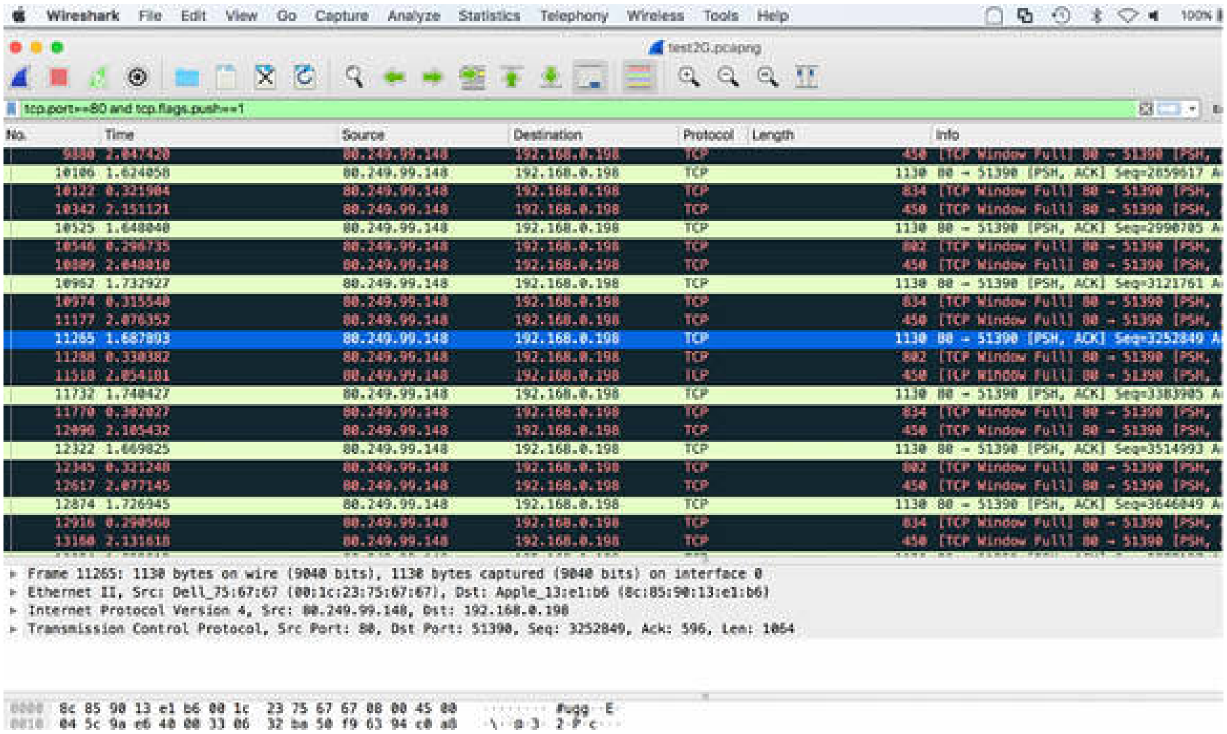
To identify the amount of data flowing with the TCP stream, select the “Bytes Out” check box, as shown in the figure below.



In general, as data is taken out of the receive buffer, the receive window should increase. From the previous figures, you can see that the receive window does not increase. Eventually, the transferred data fills up the receive window, and the data transfer stops until the receive window opens up again.

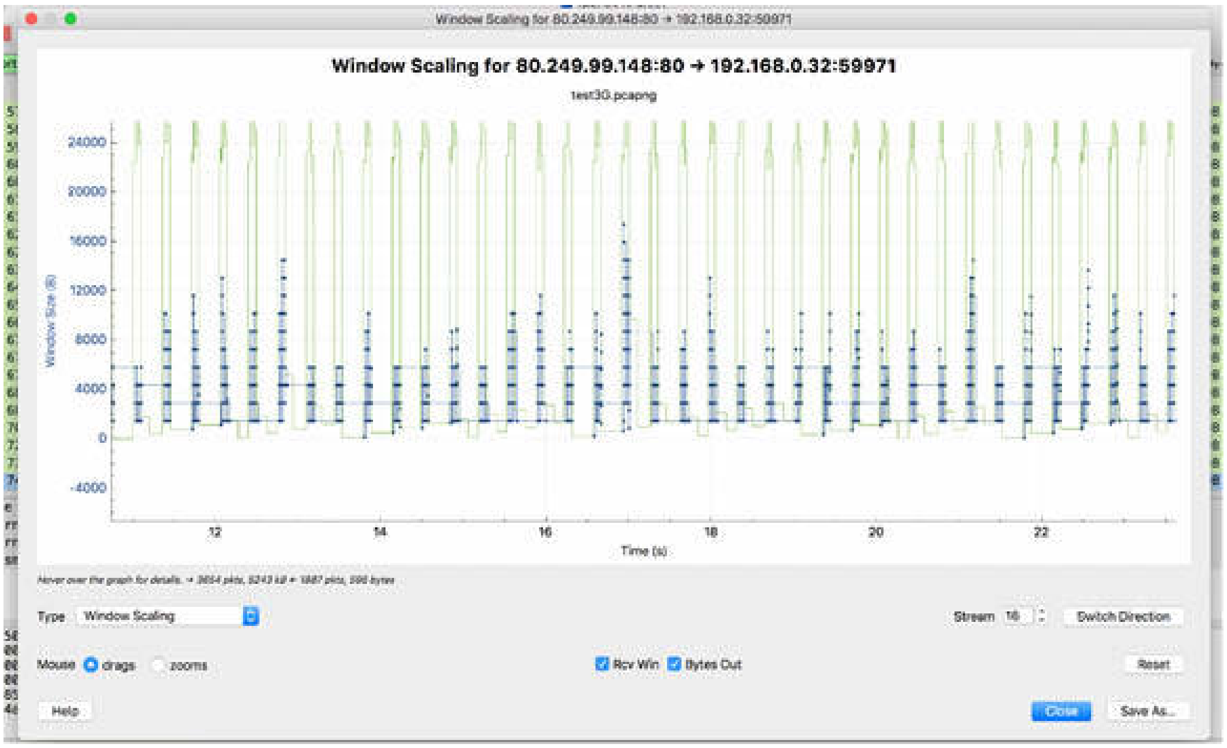
Task 2:

As shown in the figure below, in the Packet List pane, each TCP packet (TCP PSH packet) coming from IP address 80.249.99.148 is followed by two packets where the TCP window full is detected by Wireshark. The TCP window full information is clearly indicated in the figure below by using the black background.

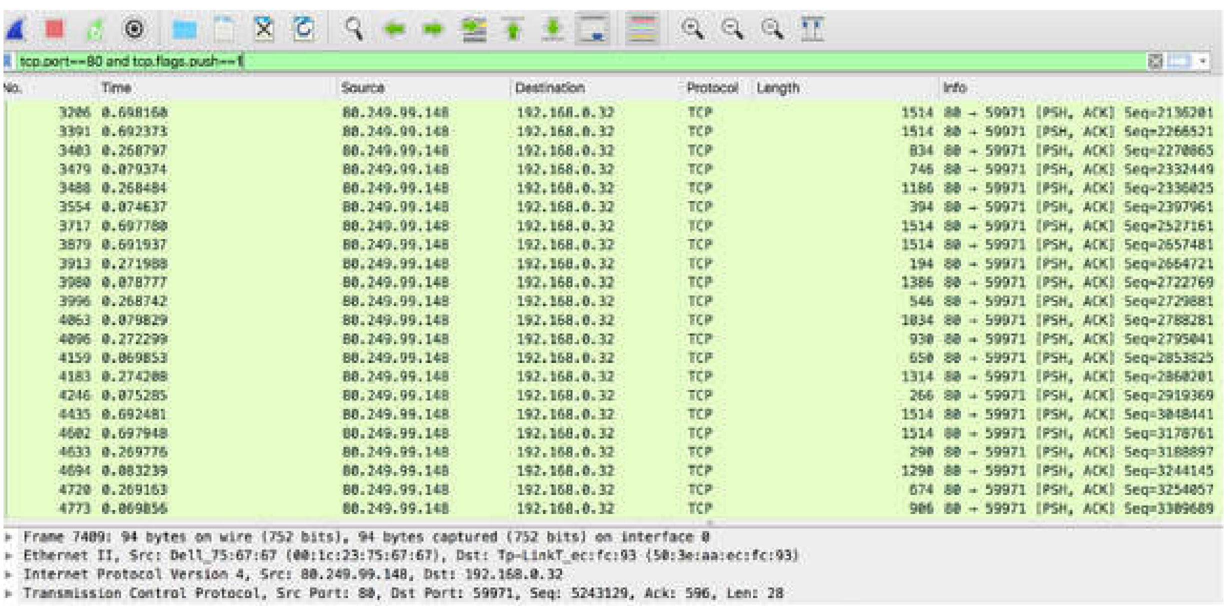


Task 3:
Open the saved trace file test3G.pcapng. In the filter toolbar, enter tcp.port==80 and tcp.flags.push==1 .

To view the TCP window graph, on the main menu, select Statistics > TCP Stream Graphs > Window Scaling. The figure below shows the resultant graph.

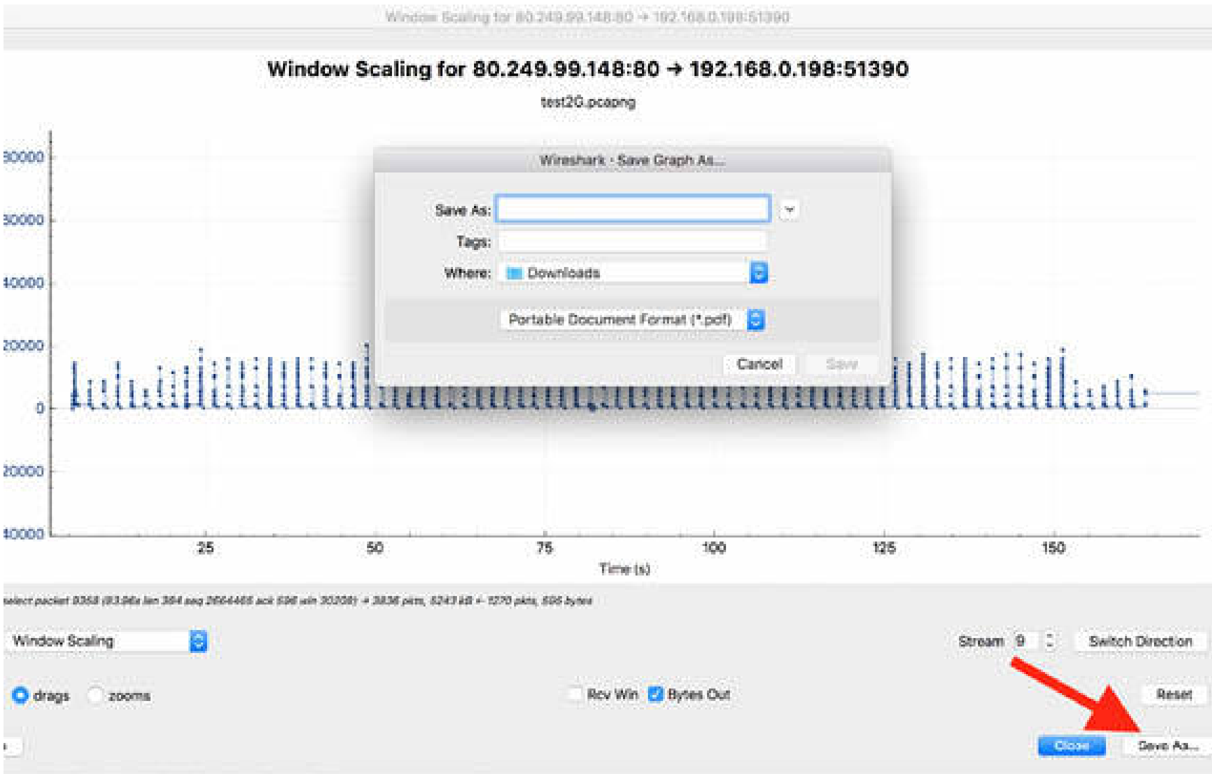


Zoom into the graph shown in the figure above to analyze it and compare it with the graphs from the earlier tasks. You will observe that the TCP window has not been saturated by the data flow, and there are no TCP Window Full messages in the Packet List pane.



Task 4:

To save each graph in PDF format, click the “Save As” button shown in the figure below. You can also use a screen capture utility.



Notes:

Repeat the previous steps to analyze the trace file “test3G.pcapng”. To gain confidence in creating and analyzing graphs, recreate all the graphs.

Network Services

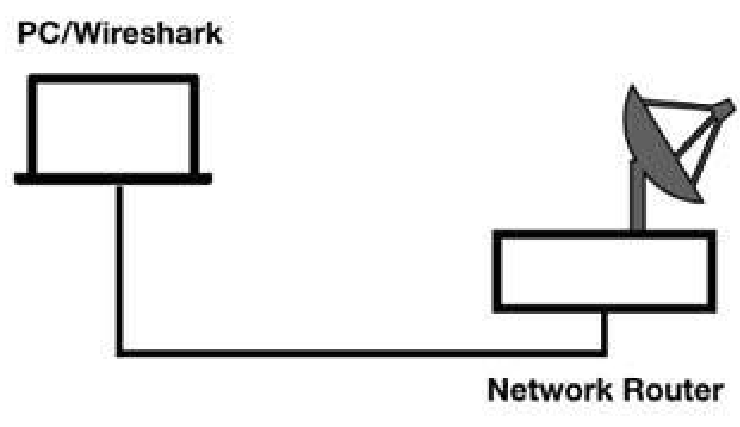
Lab 60. Dynamic Host Configuration Protocol

Lab Objective:
Learn how the Dynamic Host Configuration Protocol (DHCP) works and why is it used.

Lab Purpose:
Understand the main purpose of DHCP and its features.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

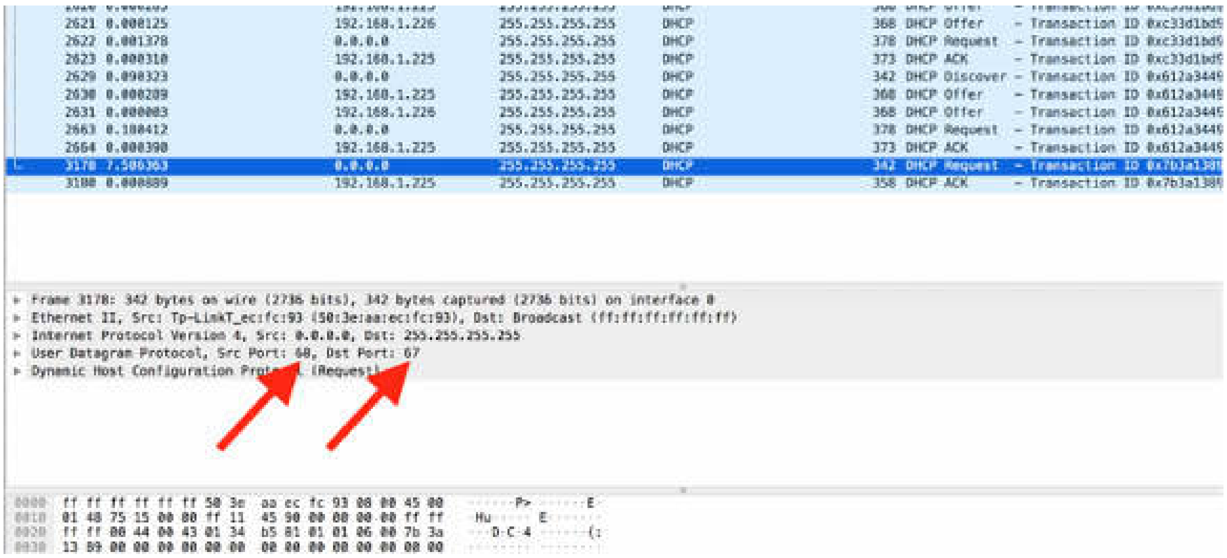
Task 1:
DHCP enables clients to obtain their IP addresses and configuration information dynamically. Based on the Bootstrap Protocol (BOOTP), DHCP is the standard for address or configuration assignments.

The DHCP uses the UDP transport layer, and it offers connectionless services for numerous configuration options.

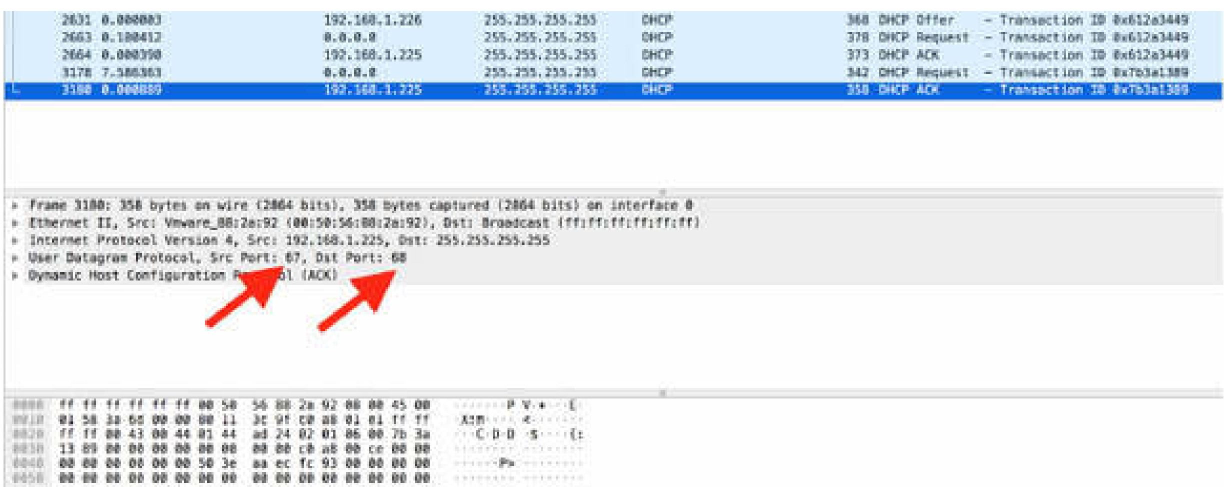
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

To force the PC to send DHCP requests, go to Network Preferences, and in the TCP/IP tab, click “Renew DHCP Lease”, as shown in the figure below. Please note that we are using MAC OS for this lab so your OS will differ.

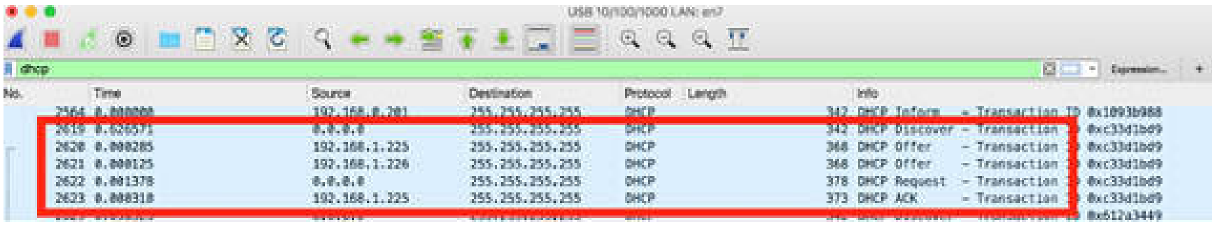
Stop the capture in Wireshark and save the file. As shown in the figure below, DHCP uses port 68 as the default port to process the client’s request.



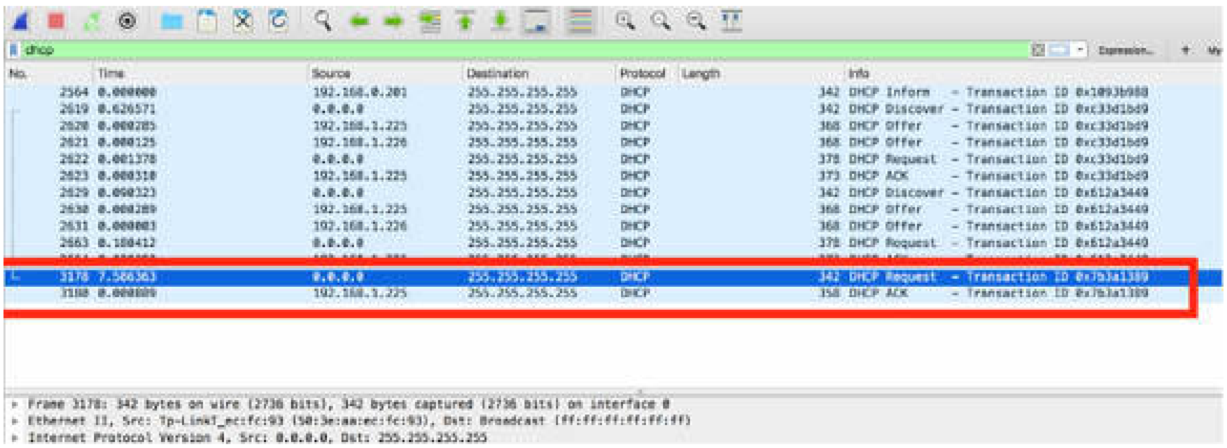
DHCP uses port 67 of the server daemon to answer the acknowledgment packet.



Depending upon the client's current configuration state and what the client wants to know from the server, the DHCP traffic can be different in terms of packets content. In the default startup of a DHCP client that is outside its address lease time, the Discover-Offer-Request-Acknowledgment sequence is used, as shown in the first sequence in the figure below.



If a client is inside its address lease time, the Request-Acknowledgment sequence is used, as shown in the last sequence in the figure below.



Task 2:
The following are the eight DHCP message types:

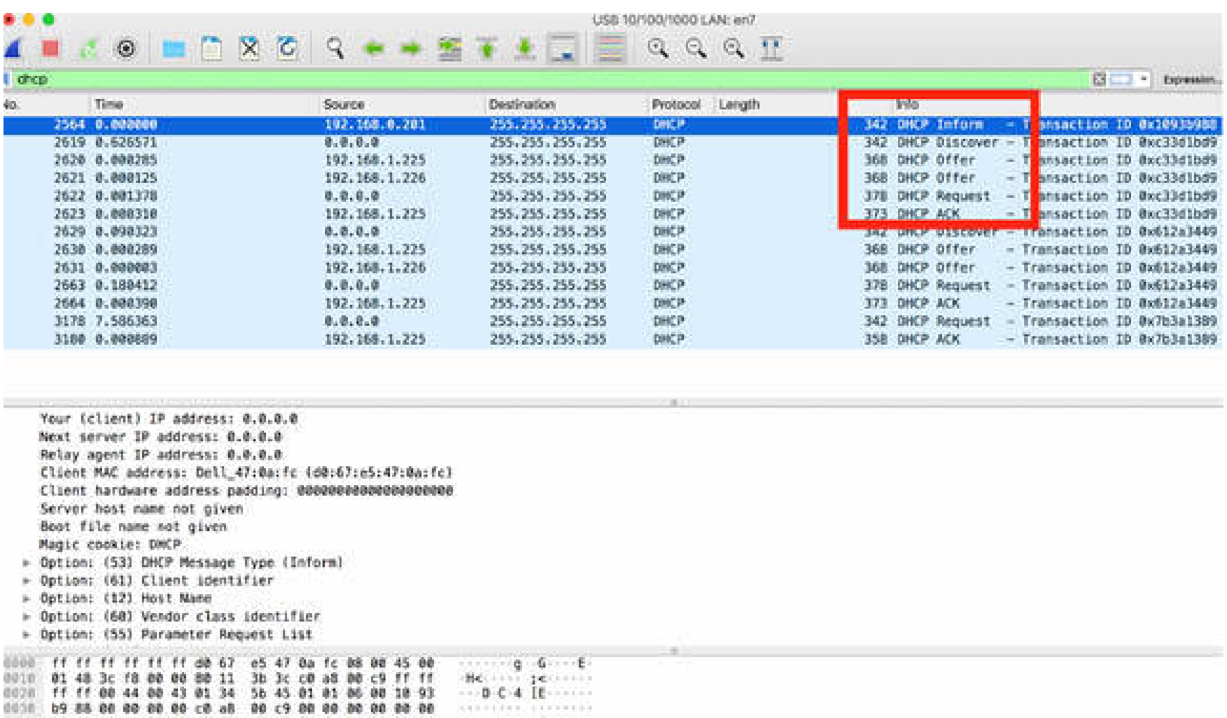
- DHCP Discover (Message Type 1): Client broadcast to locate available DHCP servers
- DHCP Offer (Message Type 2): Server to the client in response to DHCP Discover with an offer of configuration parameters
- DHCP Request (Message Type 3): Client message to servers either (a) requesting offered parameters from one server and implicitly declining offers from all others, (b) confirming the correctness of previously allocated address after a system reboot, for example, or (c) extending the lease on a particular network address
- DHCP Decline (Message Type 4): Client to the server indicating that the offered network address is not acceptable (perhaps the client

discovered the address already in use through a gratuitous ARP test process)

- DHCP Acknowledgment (Message Type 5): Server to the client with configuration parameters, including committed network address
- DHCP Negative Acknowledgment (Message Type 6): Server to the client indicating client's network address is incorrect (e.g., the client has moved to a new subnet) or the client's lease has expired
- DHCP Release (Message Type 7): Client to the server relinquishing network address and canceling the remaining lease
- DHCP Informational (Message Type 8): Client to the server, asking only for local configuration parameters; the client already has externally configured network address

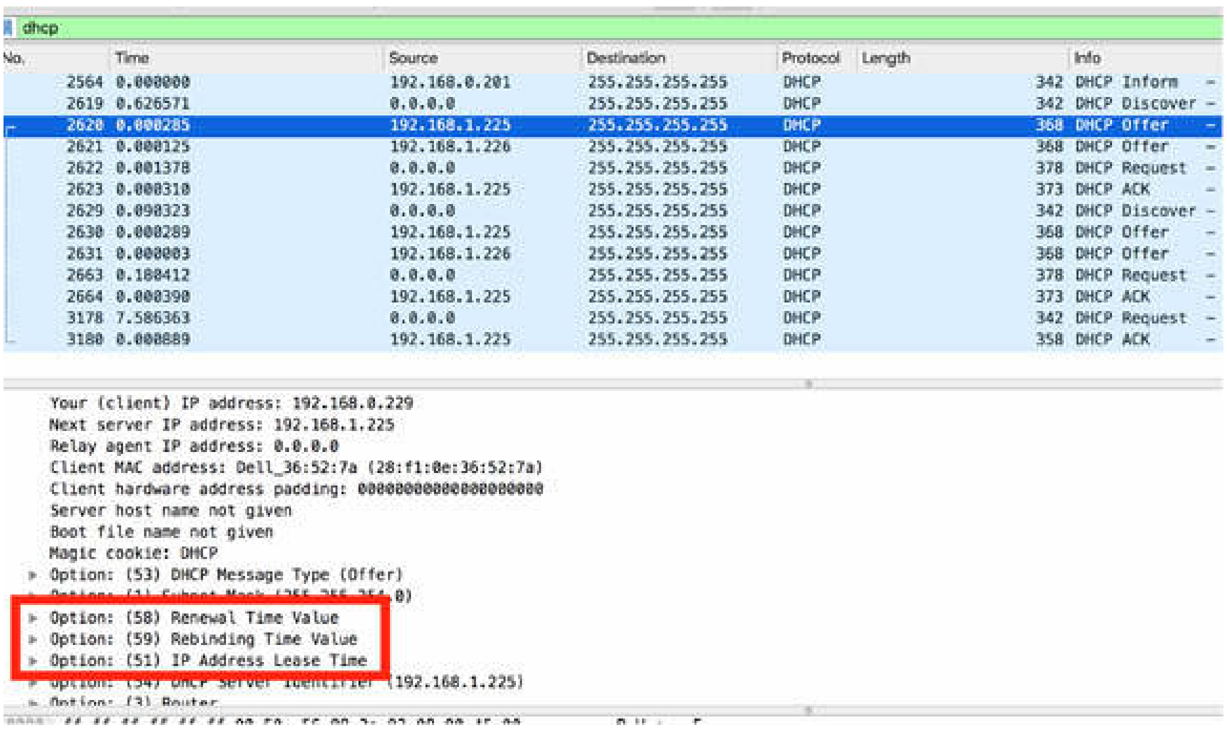
Task 3:

One of the most common uses of DHCP is the dynamic address assignment. The figure below shows the four-packet process of acquiring an address lease and parameters when a host is starting up. After the DHCP client successfully receives and acknowledges an IP address from a DHCP server, the client enters the bound state.

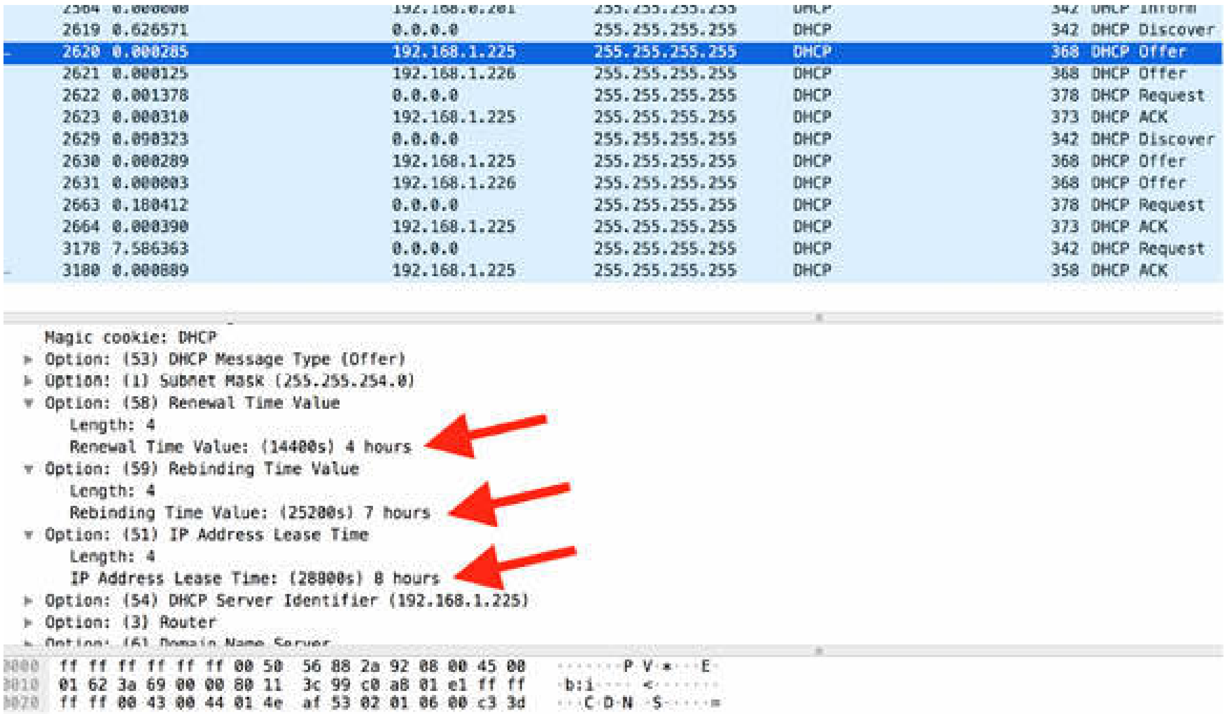


In the Packet List pane, select a DHCP Offer packet and inspect the Packet Details pane. As shown in the figure below, during the address request and assignment process, the client obtains the following three time values:

- IP Address Lease Time (LT)
- Renewal Time Value (T1)
- Rebinding Time Value (T2)



In the Packet Details pane, click each of these fields to open the tree view. The IP Address Lease Time (LT) defines for how long the client is allowed to use the assigned IP address. The Renewal Time Value (T1) is .50 LT, and the Rebinding Time Value (T2) is .875 LT.



At T1, the client moves to the renewal state and sends a unicast DHCP request to extend the lease time to the DHCP server. If the DHCP server responds with an acknowledgment, the client may return to the bound state.

If the client does not receive an acknowledgment, the client retries the DHCP request at intervals equal to one-half of the remaining time until T2 is down to a minimum of 60 seconds. If the client does not receive an acknowledgment before T2 arrives, the client enters the rebinding state. In the rebinding state, the client sends a broadcast DHCP Request to extend its lease. If the client receives an acknowledgment, it returns to the bound state.

The client retries the DHCP request at intervals equal to one-half of the remaining time until the expiration of the LT.

If the client does not receive an acknowledgment before the expiration of LT, the client must return to an uninitialized state, release its IP address, and send a DHCP broadcast to locate a DHCP server, if possible. Most DHCP client software uses a “sticky IP address”, that is, the client system remembers the last assigned IP address and requests to explicitly use that address again. In fact, Dynamic IP addressing is not as dynamic as the name implies.

Based on the description and the figure above, it is clear that DHCP relies on broadcasts for the initial DHCP Discover process. Therefore, either the DHCP server or a DHCP Relay Agent must be on the same network segment as the DHCP client.

Notes:
To gain the necessary confidence in using DHCP, repeat the previous steps to connect to a different local area network, and then try the DHCP renewal process to analyze the DHCP process.

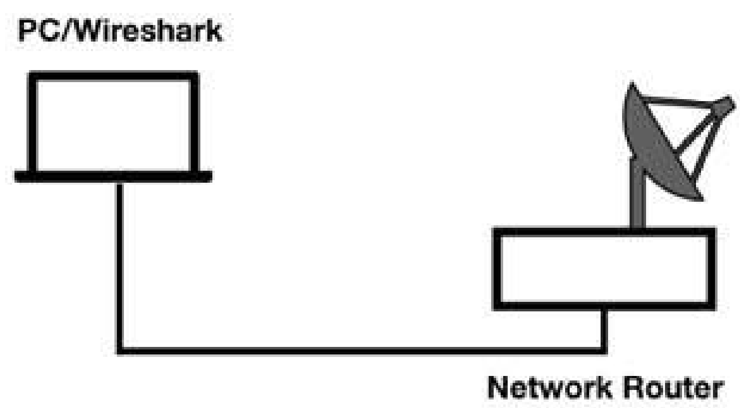
Lab 61. DHCP Problems

Lab Objective:
Learn about the more common DHCP problems.

Lab Purpose:
Learn how to detect and analyze the more common DHCP problems.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

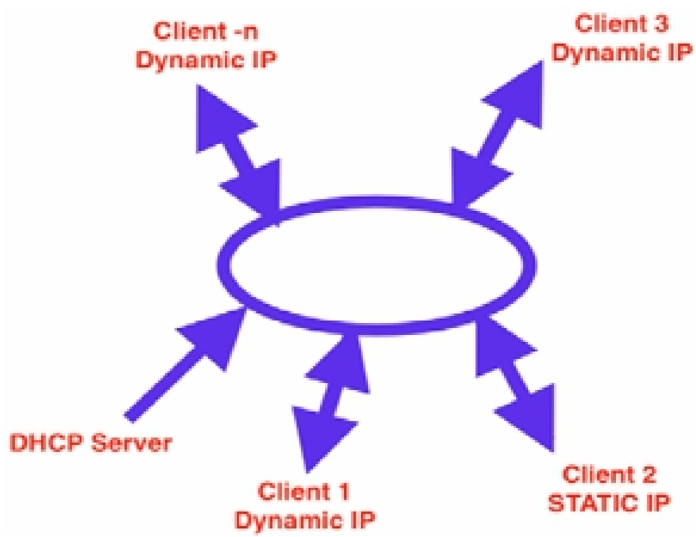


Lab Walkthrough:

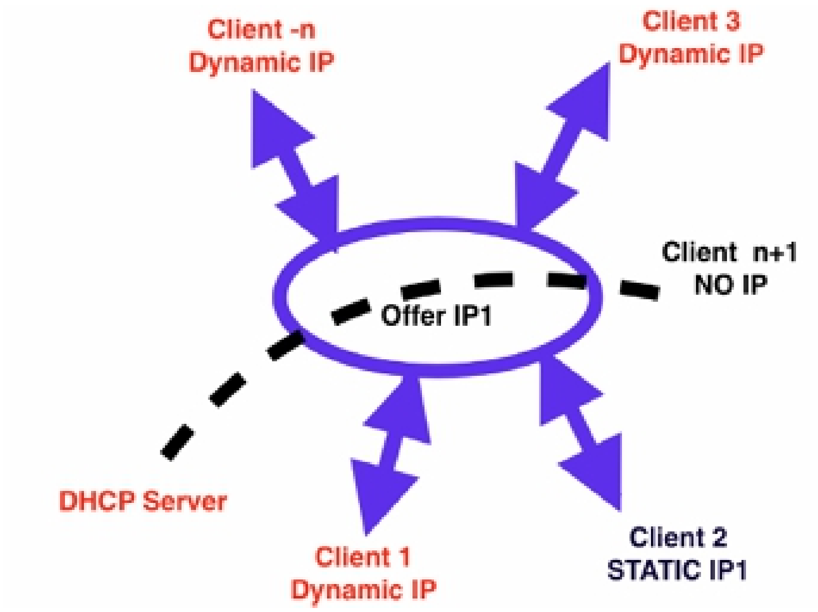
Task 1:
If DHCP doesn't work properly, the DHCP clients may not be able to obtain or maintain IP addresses or other client configurations, and as a result, a

DHCP client cannot access the internet.

The figure below shows one of the most common DHCP problem scenarios.



In this scenario, one or more hosts on the network have statically-assigned addresses and the DHCP server is unaware of this. In such a situation, the DHCP server may offer an address that is already in use in the network. This can cause a problem because two hosts with the same IP cannot exist on the same network. The figure below shows this problematic situation.



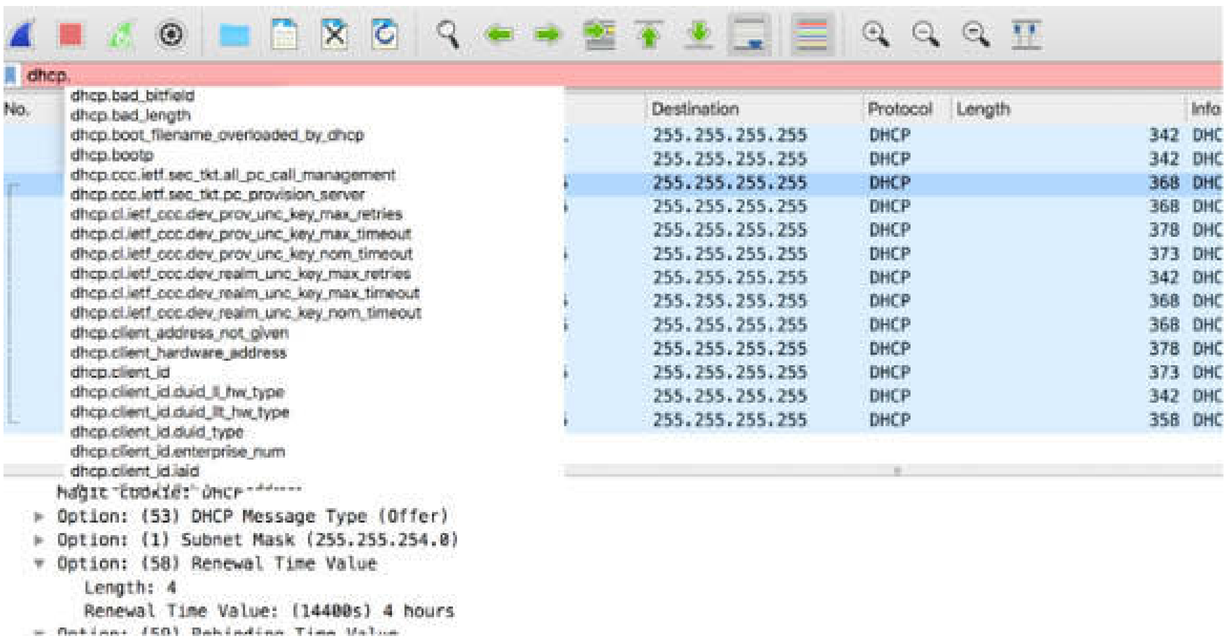
In the network, "Client 2" exists with a static IP1, while all other clients have dynamic IP. When "Client n+1" enters the network, the DHCP server offers the exact IP1 to the new client. The DHCP client can perform the duplicate address test. If the DHCP client locates another host with the same address, it must decline the IP address provided in the DHCP offer. It, however, remains with no IP address assigned.

Task 2:
A better alternative is that the DHCP server directly performs a duplicate address test (typically using ICMP Echo Requests) so that it offers only those IP addresses that are not currently not being used in the network.

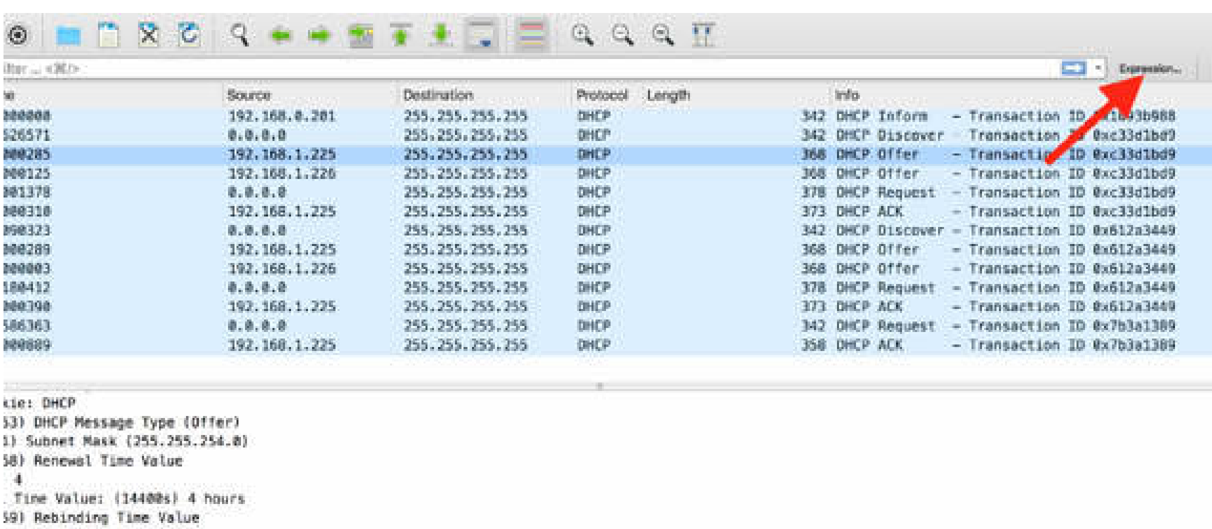
The situation described above is not easy to understand, we should apply a DHCP approach based on "Capture Filter" instead of analyzing all the traffic.

Let's suppose you capture traffic by using the capture filter port 67 or port 68 to capture only DHCP messages. In this situation, you cannot observe the duplicate address test (ICMP) made by the client or the duplicate ICMP test made by the DHCP server, thus, totally missing the problem.

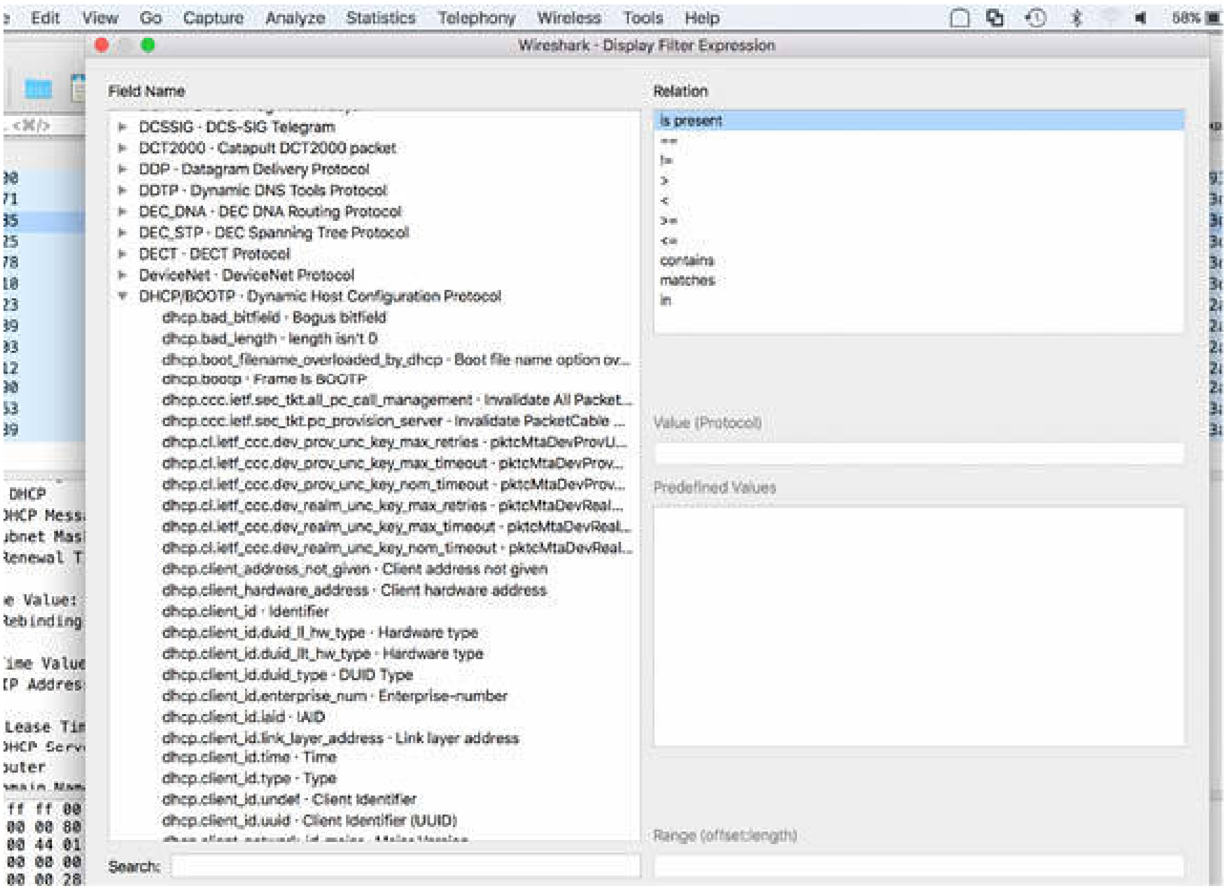
The only thing you see is that the client declined the DHCP offer but you do not know why. The solution is to capture all the traffic and analyze it with display filters. The figure below shows some of the display filters available for DHCP. In the filter toolbar, type dhcp. The auto-complete feature displays a drop-down menu with all available DHCP filters, as shown in the figure below.



Alternatively, click the Expression button on the right of the filter toolbar, as shown in the figure below.



The Display Filter Expression dialog box is displayed. To view a list of all DHCP filters, scroll down and click the DHCP/BOOTP—Dynamic Host Configuration Protocol field to open the tree view. Manually create a more appropriate display filter.



Notes:

Repeat the previous steps capturing the DHCP process with a capture filter (using port 67/68) or with a display filter and try to find the differences in the two approaches.

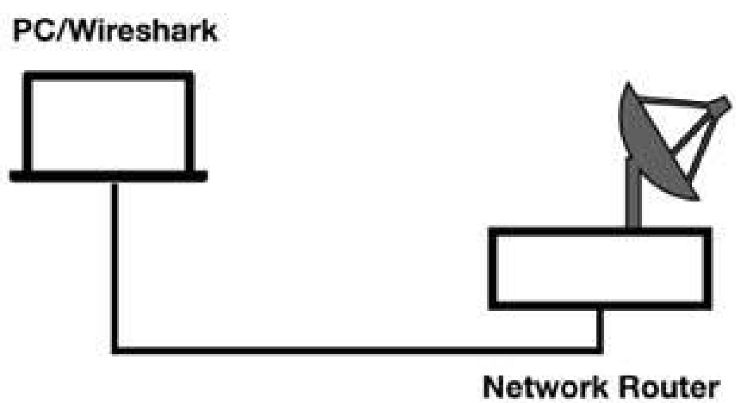
Lab 62. DHCP Packet Structure

Lab Objective:
Learn the DHCP packet structure.

Lab Purpose:
Learn about the structure and various fields of a DHCP packet.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

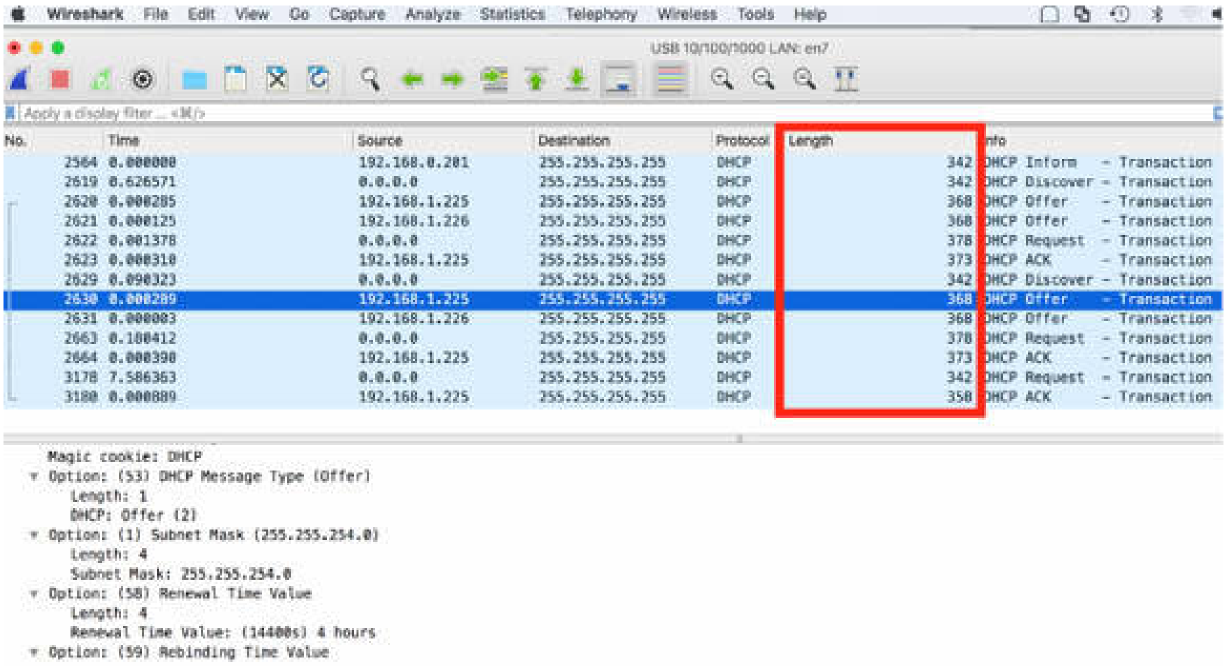
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
Open a packet capture saved in one of the previous labs—choose a capture in which DHCP packets are present. In the filter toolbar, enter `dhcp` to

display only DHCP packets, as shown in the figure below.

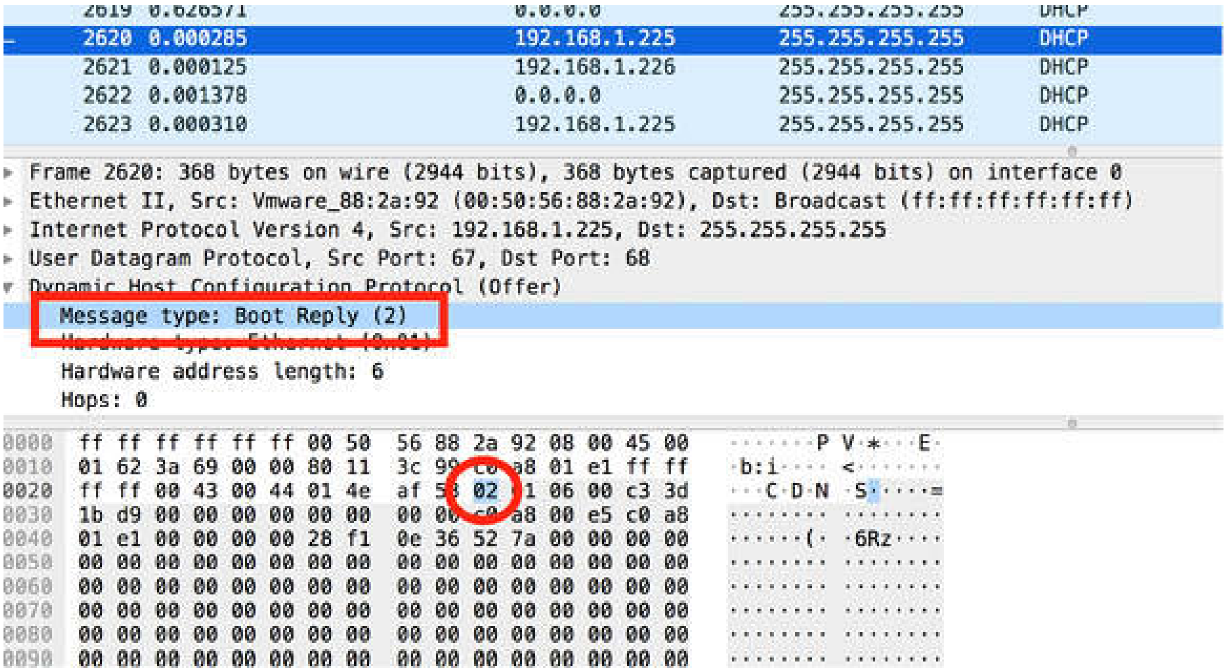


In the Packet List pane, observe that the DHCP packets have variable lengths, as shown in the figure above.

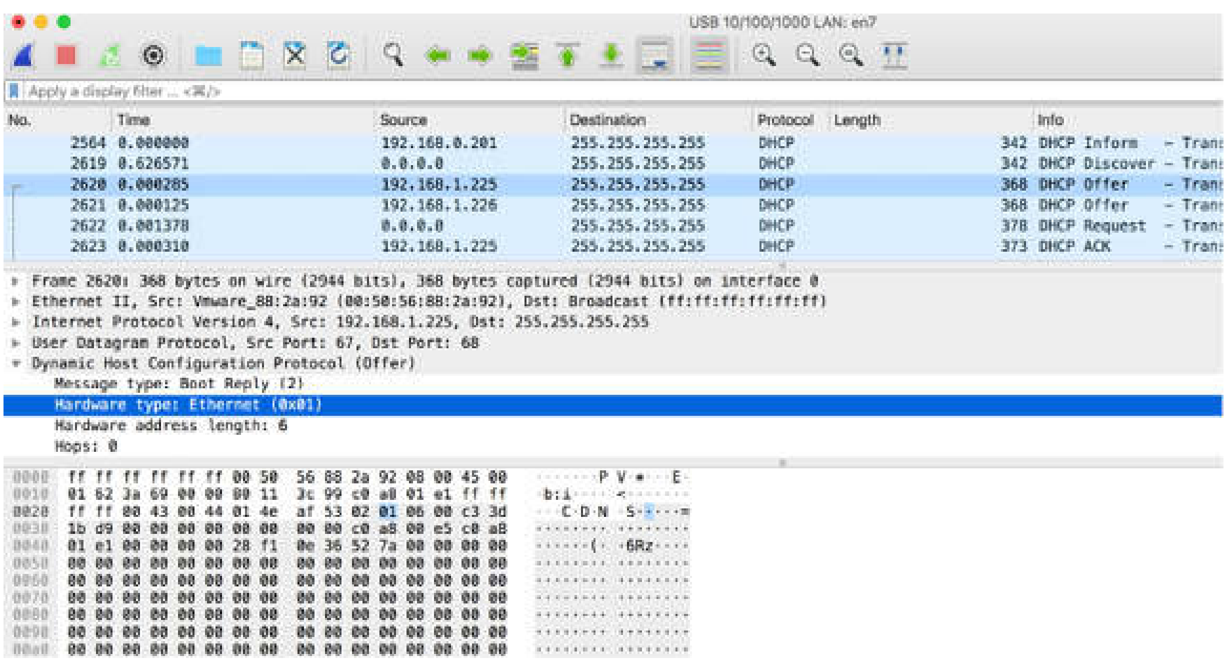
Task 2:

In the Packet List pane, select a packet. In the Packet Details pane, examine each field of the DHCP packet by selecting a field and locating it in the Packet Bytes pane.

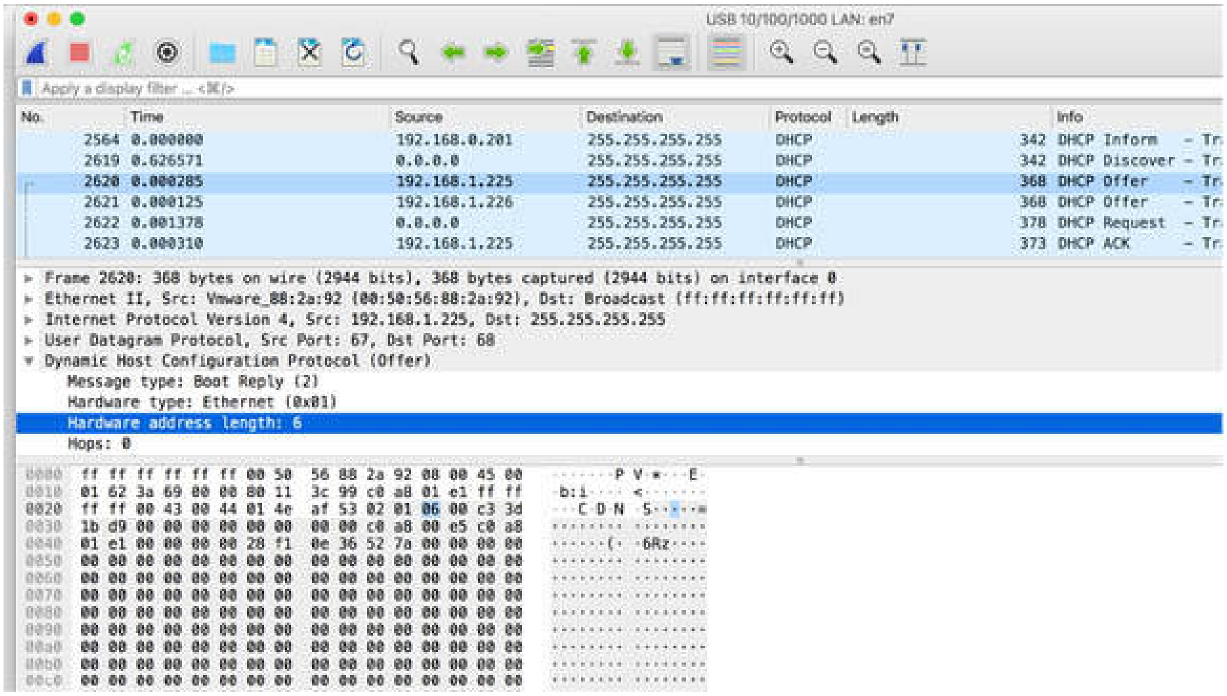
The Message Type field is the first field. This field is also referred to as the Opcode field. A value of 1 indicates a DHCP request and a value of 2 indicates a DHCP reply.



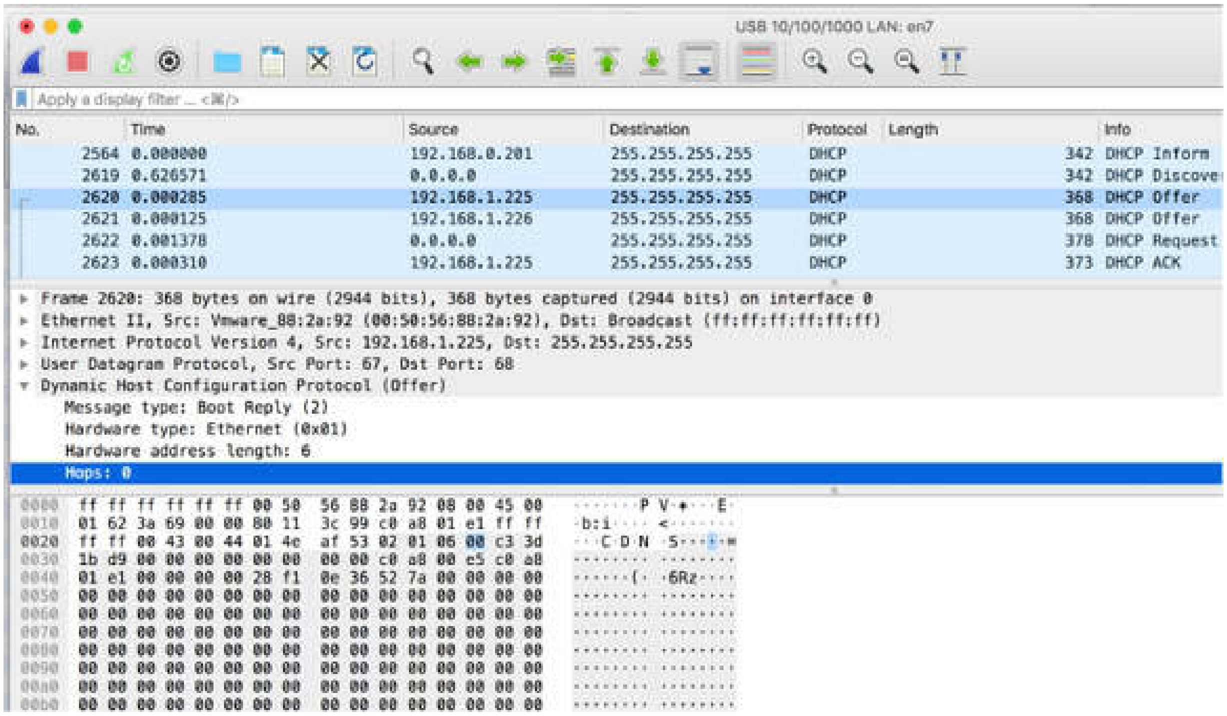
The “Hardware type” field is the next field. It defines the type of hardware address in use and matches the ARP hardware address type definitions. The value 0x0001 indicates that the hardware address is an Ethernet address.



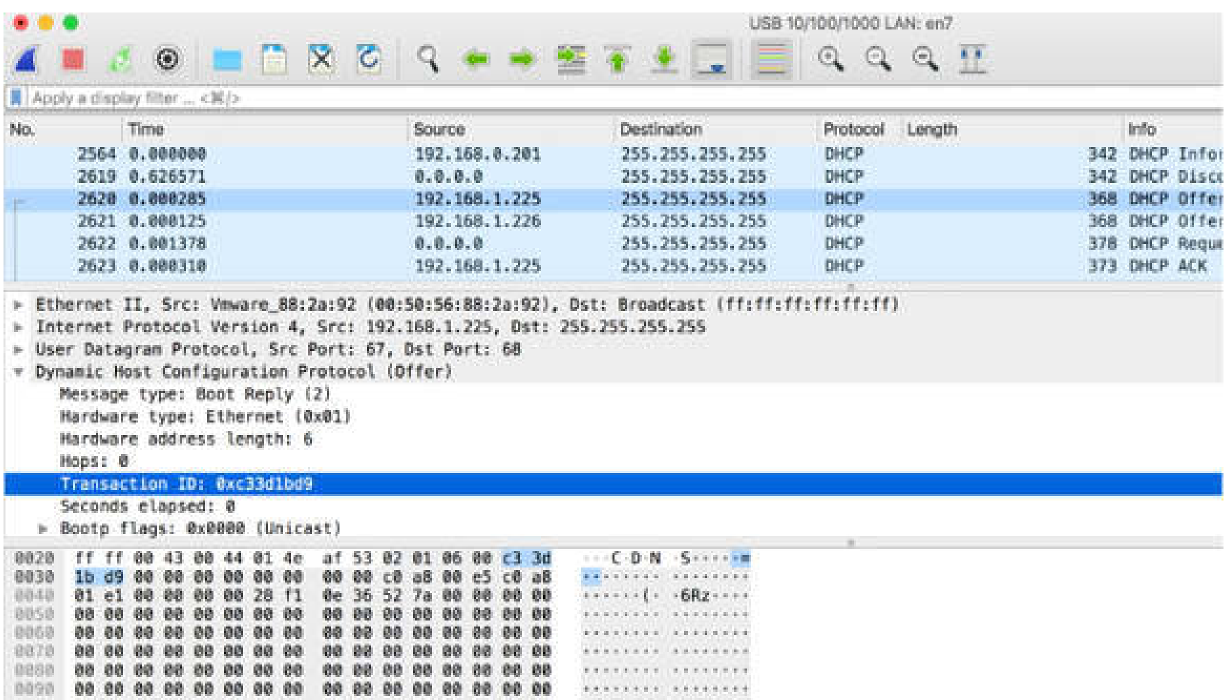
The “Hardware address length” field is the next field. It indicates the length of the hardware address, which is 6 for an Ethernet address.



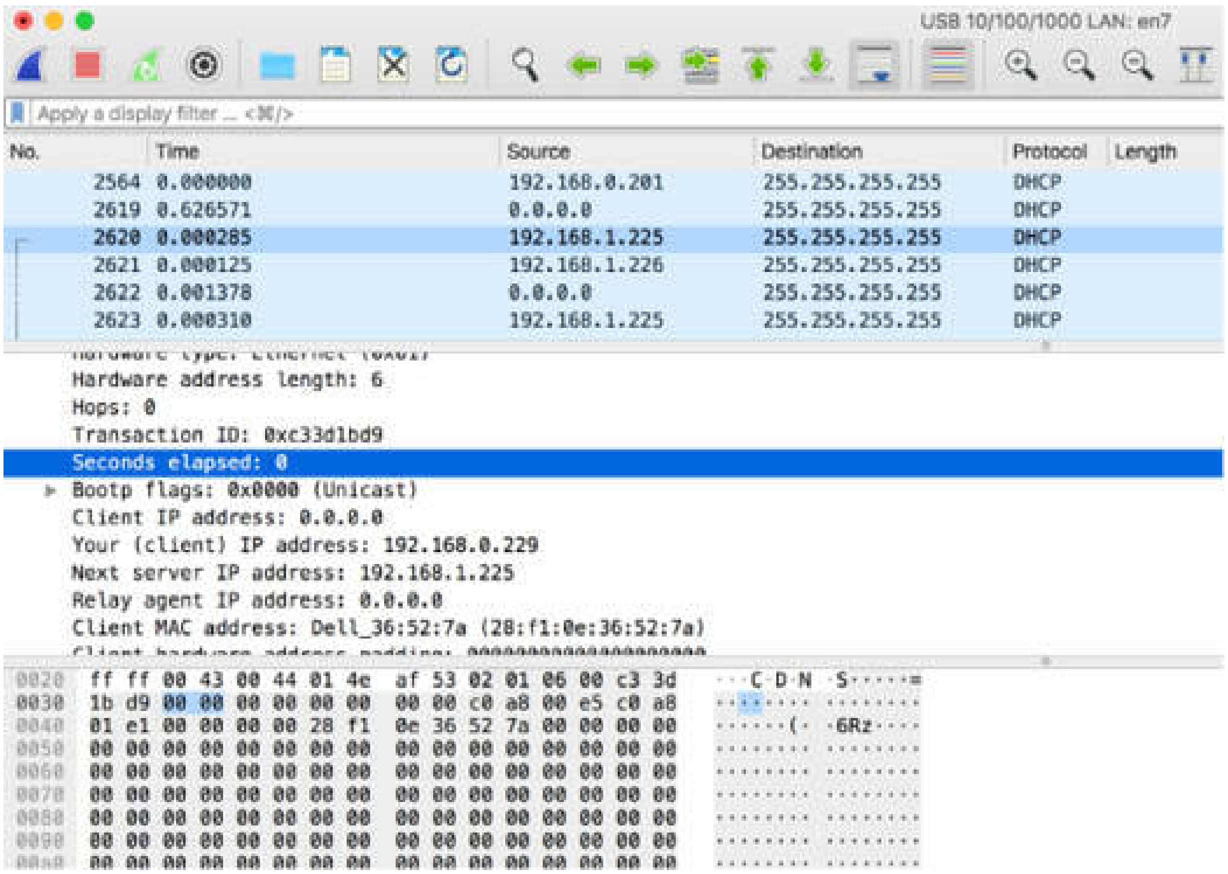
The Hops field is the next field. It is used by DHCP relay agents to define the number of networks that must be crossed to get to the DHCP server.



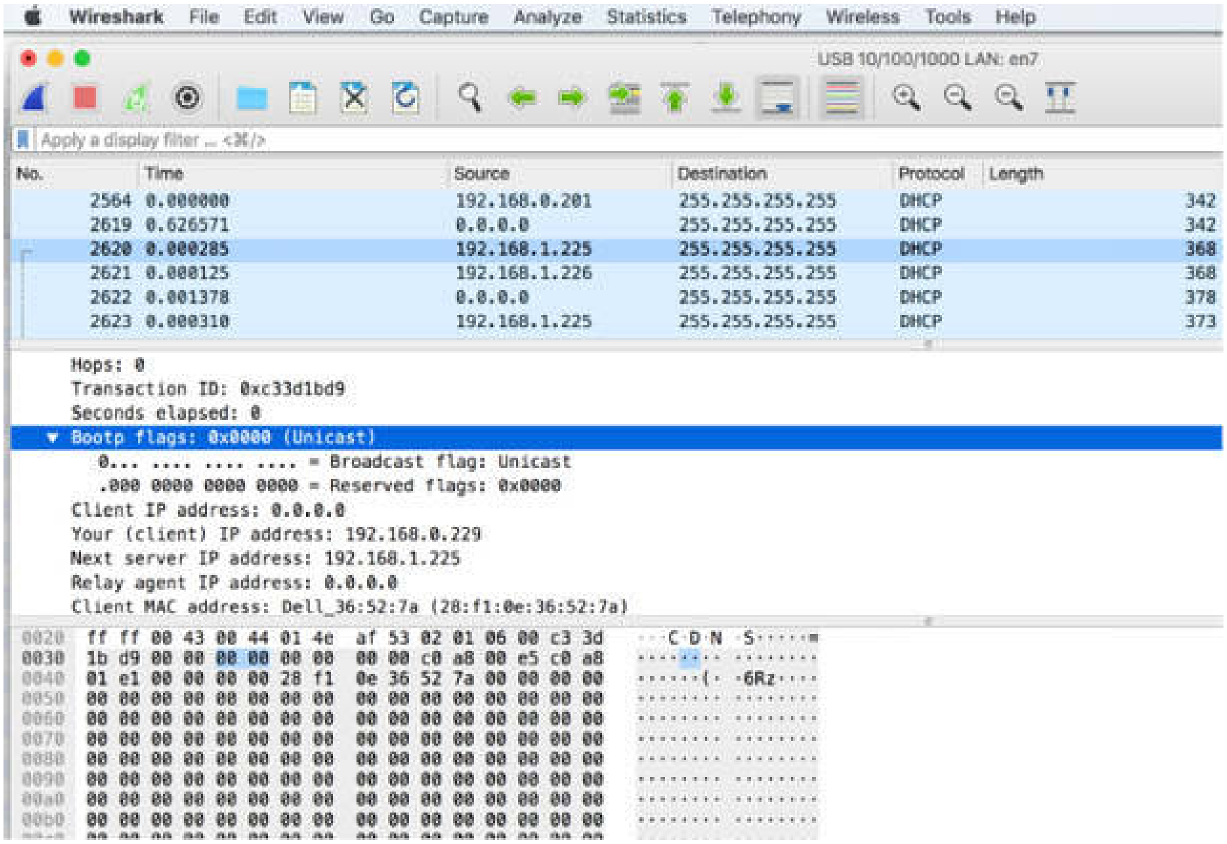
The “Transaction ID” field is the next field. It is used to match the DHCP request and response packets.



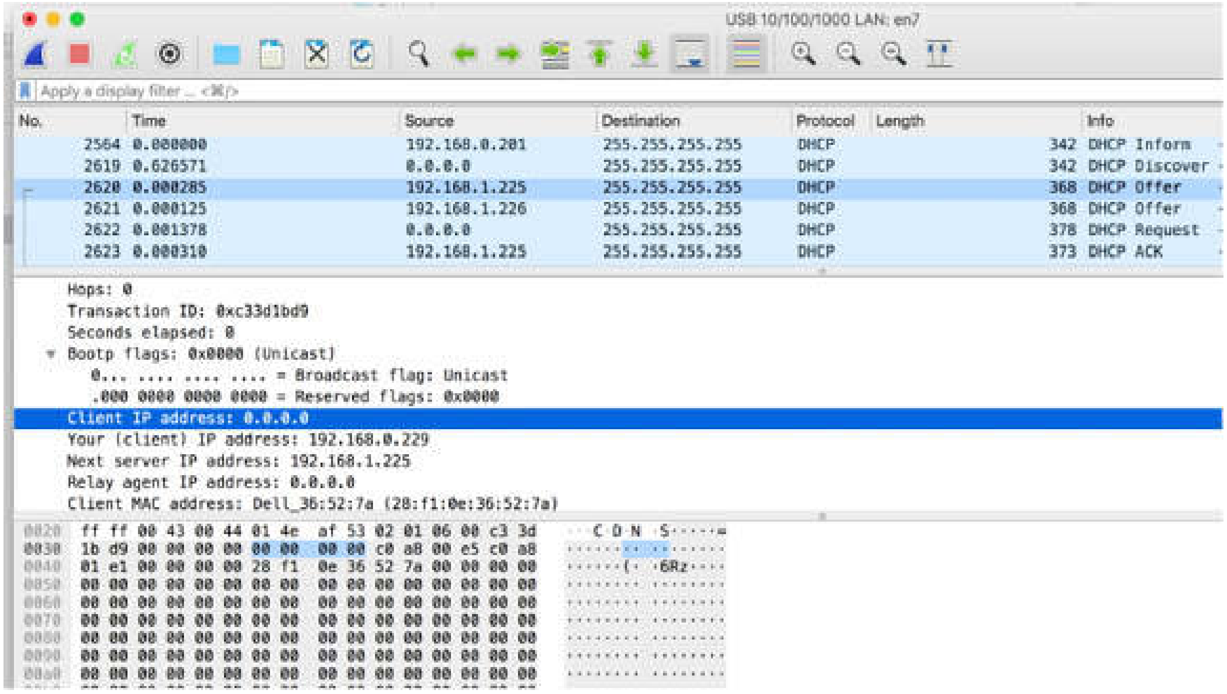
The “Seconds elapsed” field is the next field. It indicates the number of seconds that have elapsed since the client began requesting a new address or renewal of an address. You may notice an error warning from Wireshark when a vendor sets this field using the little-endian format. Wireshark interprets the field based on the big-endian format but provides a note indicating the difference.



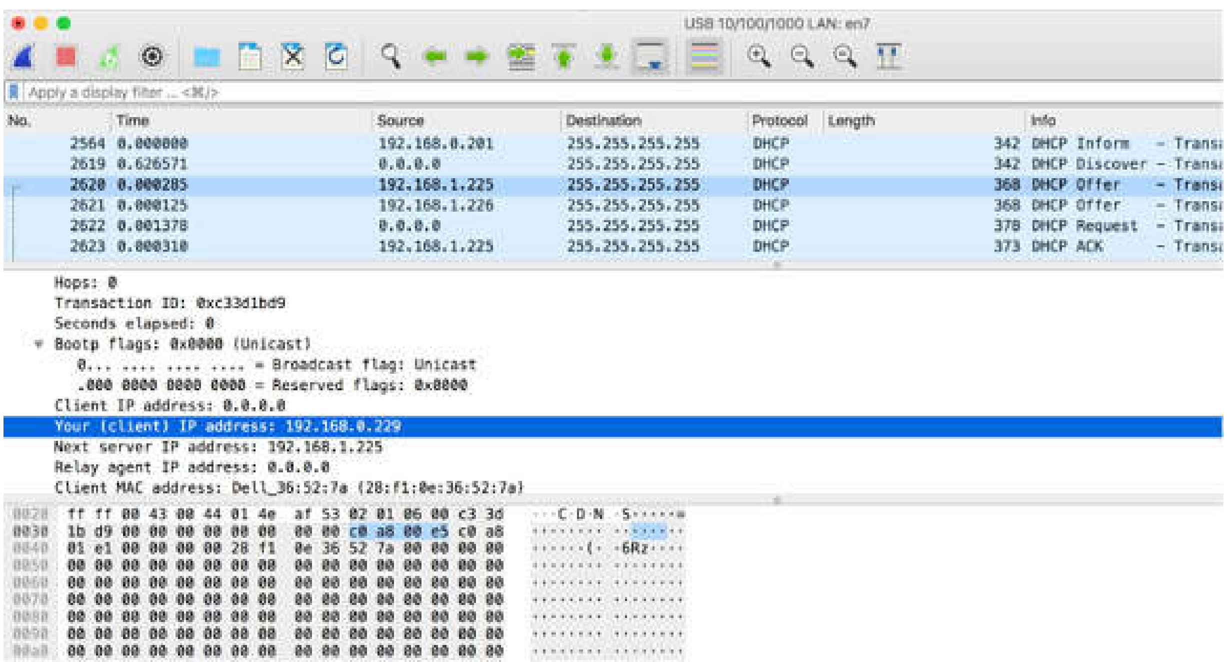
The “Bootp flags” field is the next field. These flags indicate whether clients accept unicast or broadcast MAC packets before the IP stack is completely configured. This field is not present in DHCPv6. The figure below shows the flags in the tree view.



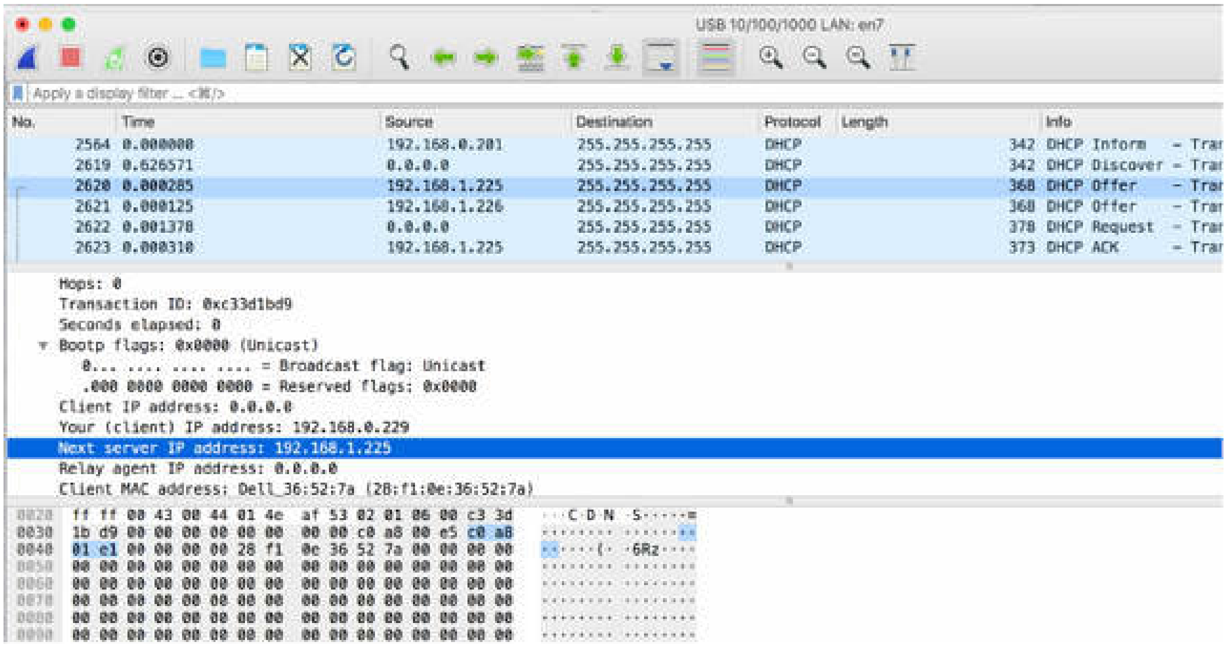
The “Client IP address” field is the next field. It is filled by the client when the DHCP server assigns an IP address to the client.



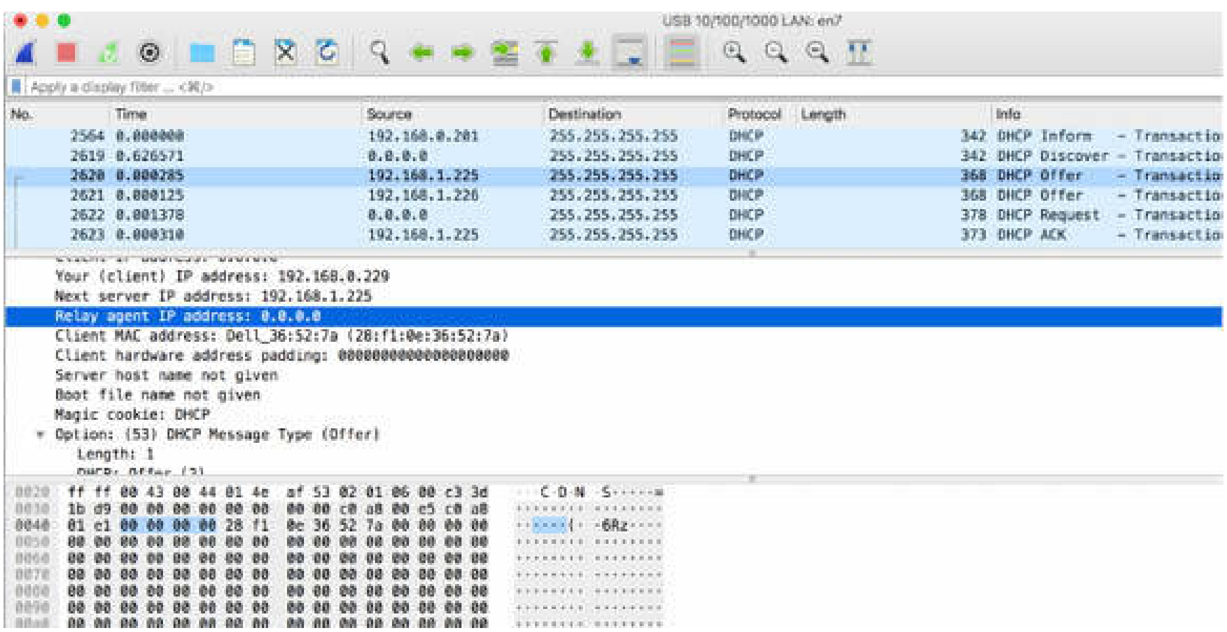
The “Your (client) IP address” field is the next field. It indicates the address offered by the DHCP server. Only the DHCP server fills in this field.



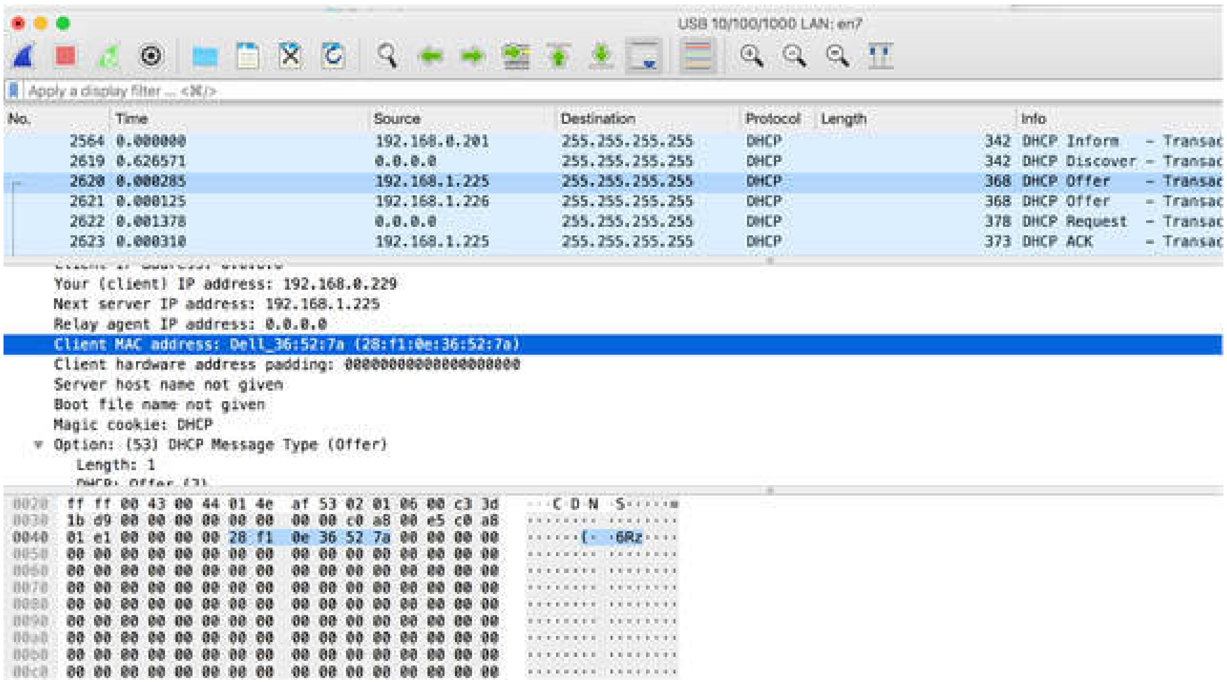
The “Next server IP address” field is the next field. It contains the address of the DHCP server when a relay agent is used.



The “Relay Agent IP address” field is the next field. It shows the address of the DHCP relay agent (if one is in use).



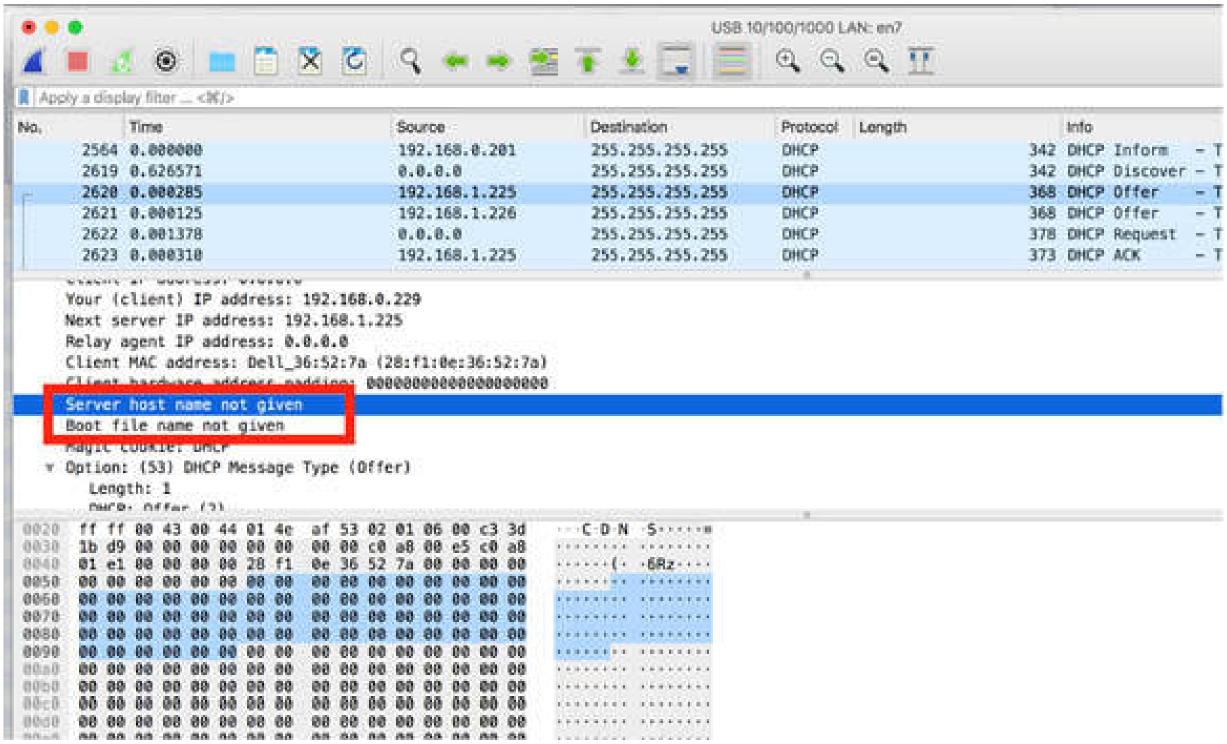
The “Client MAC address” field is the next field. It contains the MAC address of the client. This is a useful field to filter on if a user complains about the boot-up process, and you suspect it might be a DHCP problem.



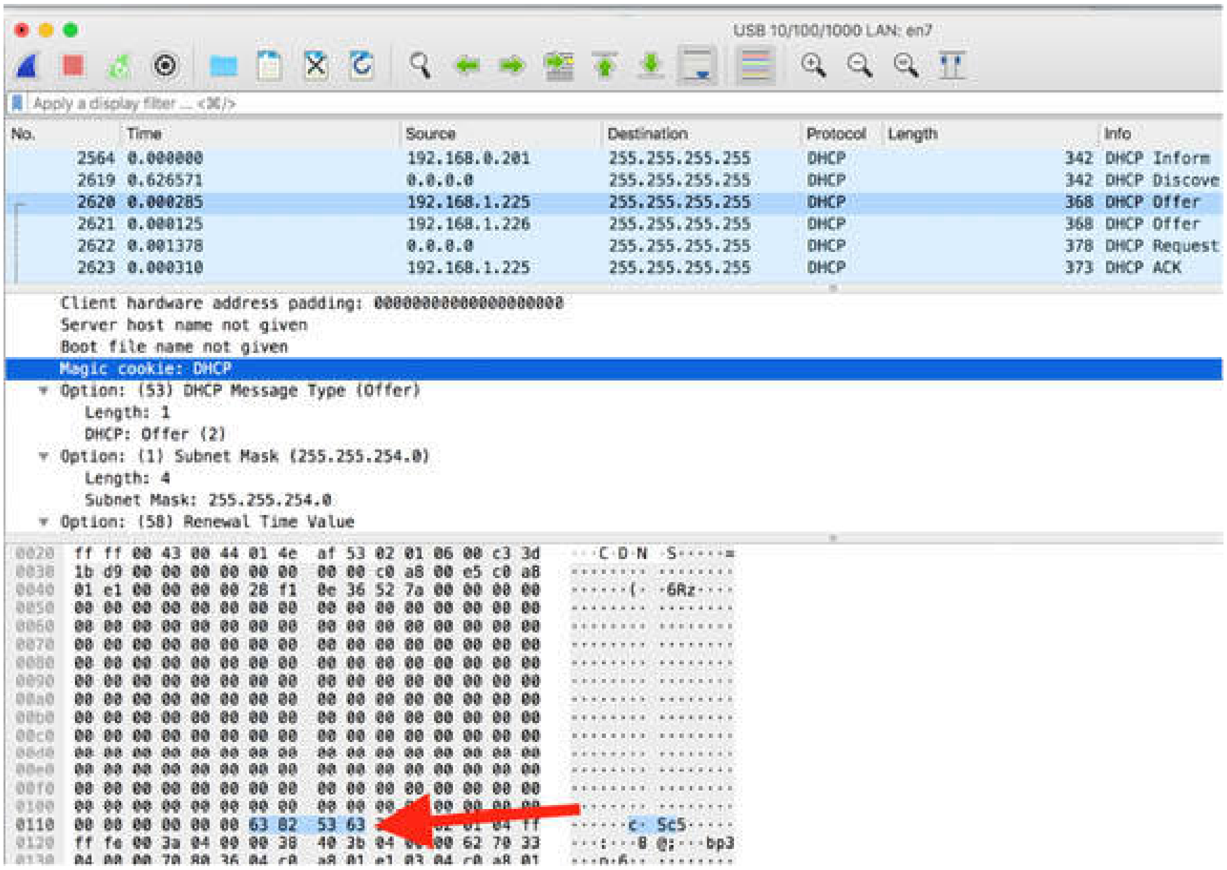
The “Server host name” field is the next field. It can contain the name of the DHCP server.

The “Boot file name” field is the next field. It indicates a boot file name.

Both these fields are optional.

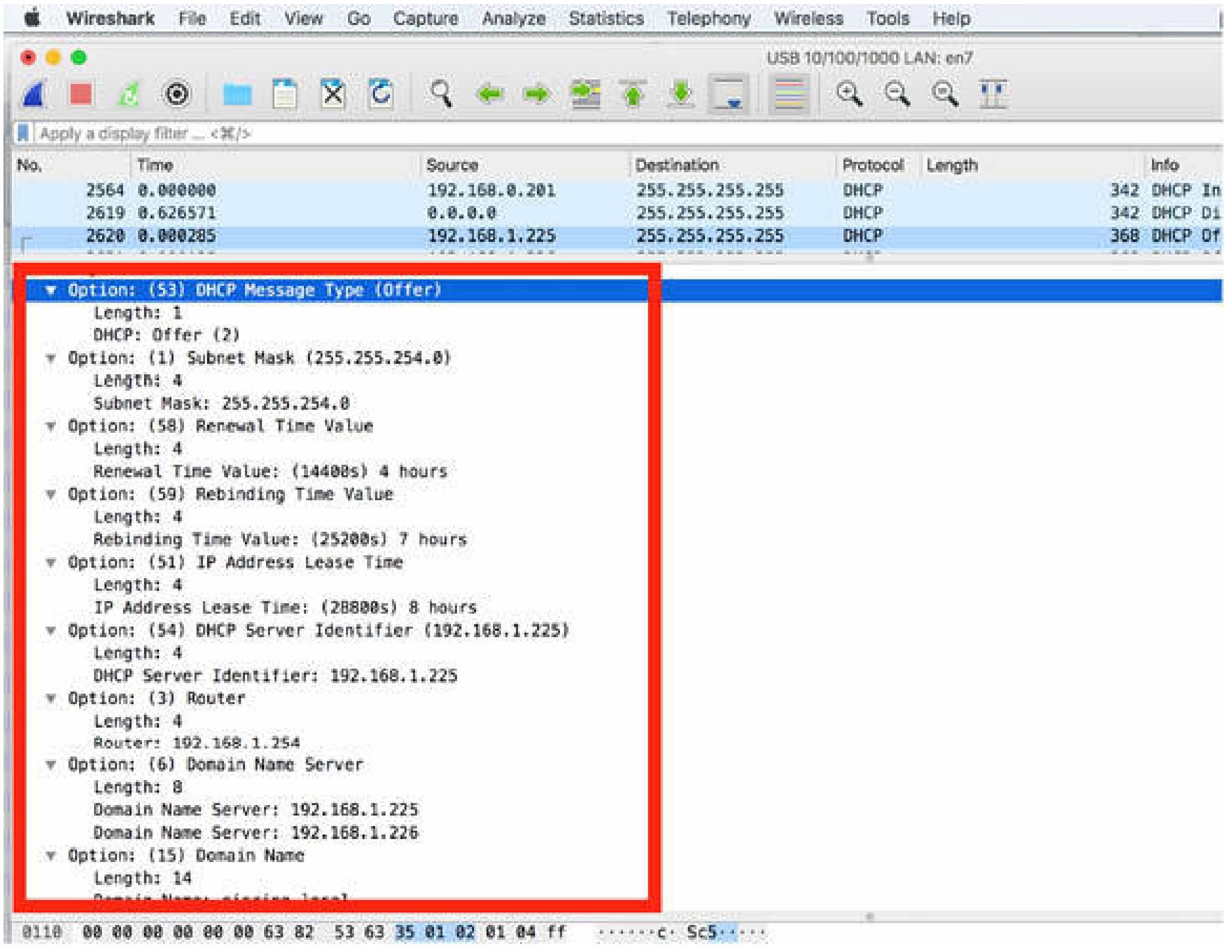


The “Magic cookie” field is the next field. It indicates the type of the data that follows. The value 0x63825363 indicates that the data is DHCP.



The Option fields are the last fields. The options are used to provide the IP address and configuration requests to the DHCP server and replies to the client. The following list contains some of the common option types:

- Option 1: Subnet Mask
- Option 3: Router
- Option 4: Time Server
- Option 5: Name Server
- Option 6: Domain Server
- Option 12: Host Name
- Option 15: Domain Name
- Option 31: Router Discovery


Wireshark File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
USB 10/100/1000 LAN: en7
Apply a display filter ... <⌘/>
No. Time Source Destination Protocol Length Info
2564 0.000000 192.168.0.201 255.255.255.255 DHCP 342 DHCP In
2619 0.626571 0.0.0.0 255.255.255.255 DHCP 342 DHCP Di
2620 0.000285 192.168.1.225 255.255.255.255 DHCP 368 DHCP Of
Option: (53) DHCP Message Type (Offer)
Length: 1
DHCP: Offer (2)
Option: (1) Subnet Mask (255.255.254.0)
Length: 4
Subnet Mask: 255.255.254.0
Option: (58) Renewal Time Value
Length: 4
Renewal Time Value: (14400s) 4 hours
Option: (59) Rebinding Time Value
Length: 4
Rebinding Time Value: (25200s) 7 hours
Option: (51) IP Address Lease Time
Length: 4
IP Address Lease Time: (28800s) 8 hours
Option: (54) DHCP Server Identifier (192.168.1.225)
Length: 4
DHCP Server Identifier: 192.168.1.225
Option: (3) Router
Length: 4
Router: 192.168.1.254
Option: (6) Domain Name Server
Length: 8
Domain Name Server: 192.168.1.225
Domain Name Server: 192.168.1.226
Option: (15) Domain Name
Length: 14
0110 00 00 00 00 00 00 63 82 53 63 35 01 02 01 04 ff

Notes:

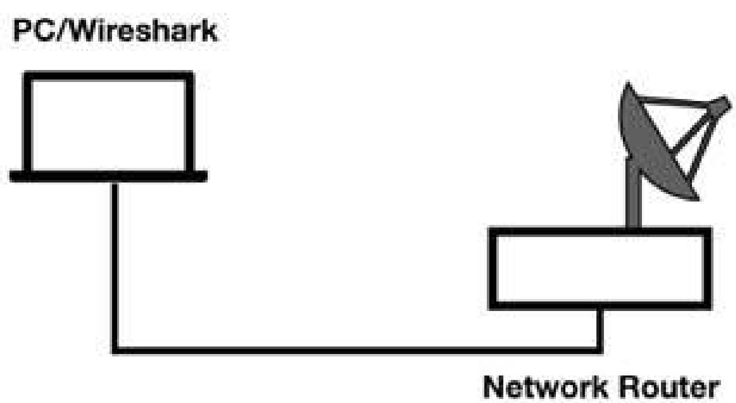
Lab 63. DHCP Statistics and Filters

Lab Objective:
Learn how to use DHCP statistics and filter features.

Lab Purpose:
Learn how to filter DHCP messages and how to display DHCP statistics.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

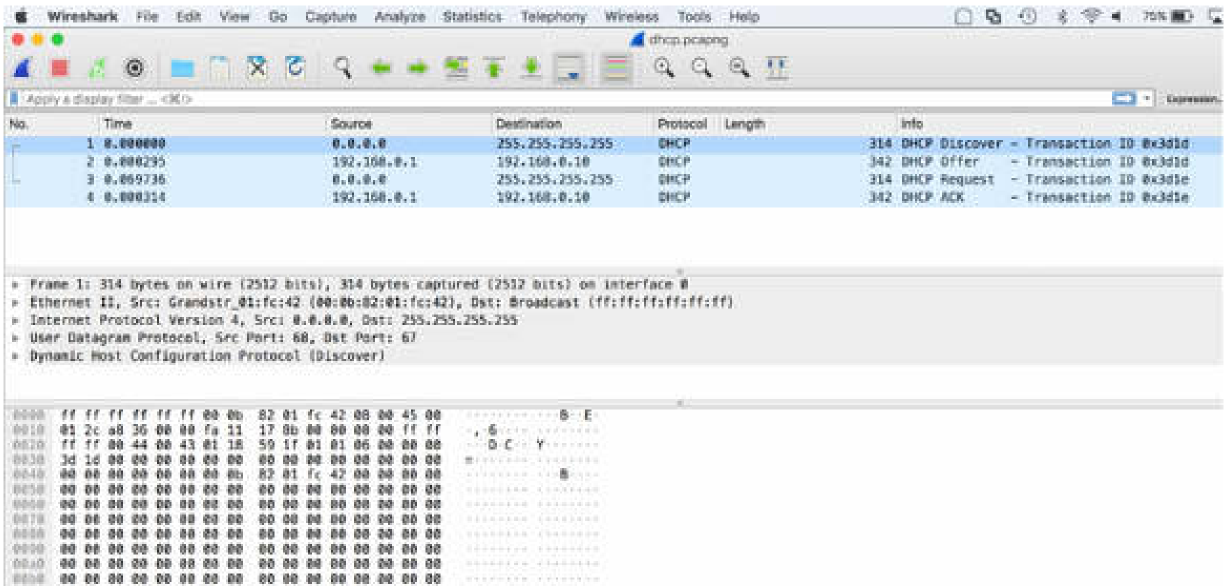


Lab Walkthrough:

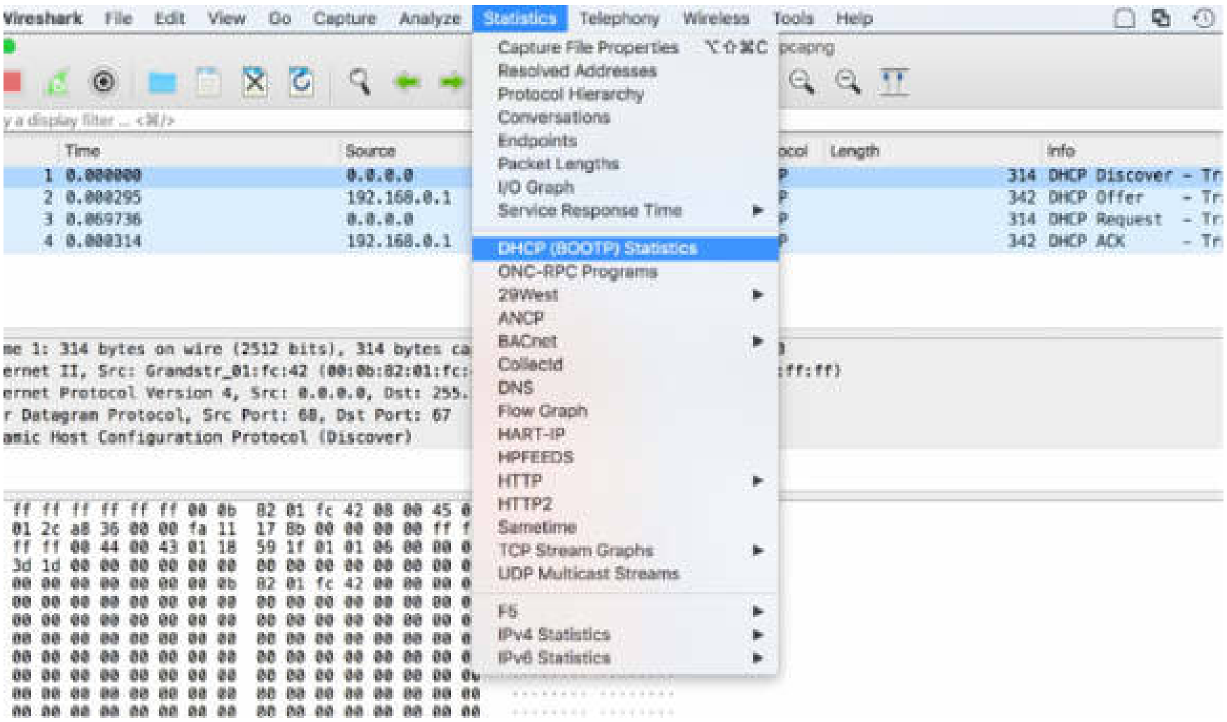
Task 1:

Download the free sample capture file dhcp.pcap to your PC from https://wiki.wireshark.org/SampleCaptures , and then open the downloaded file in Wireshark.

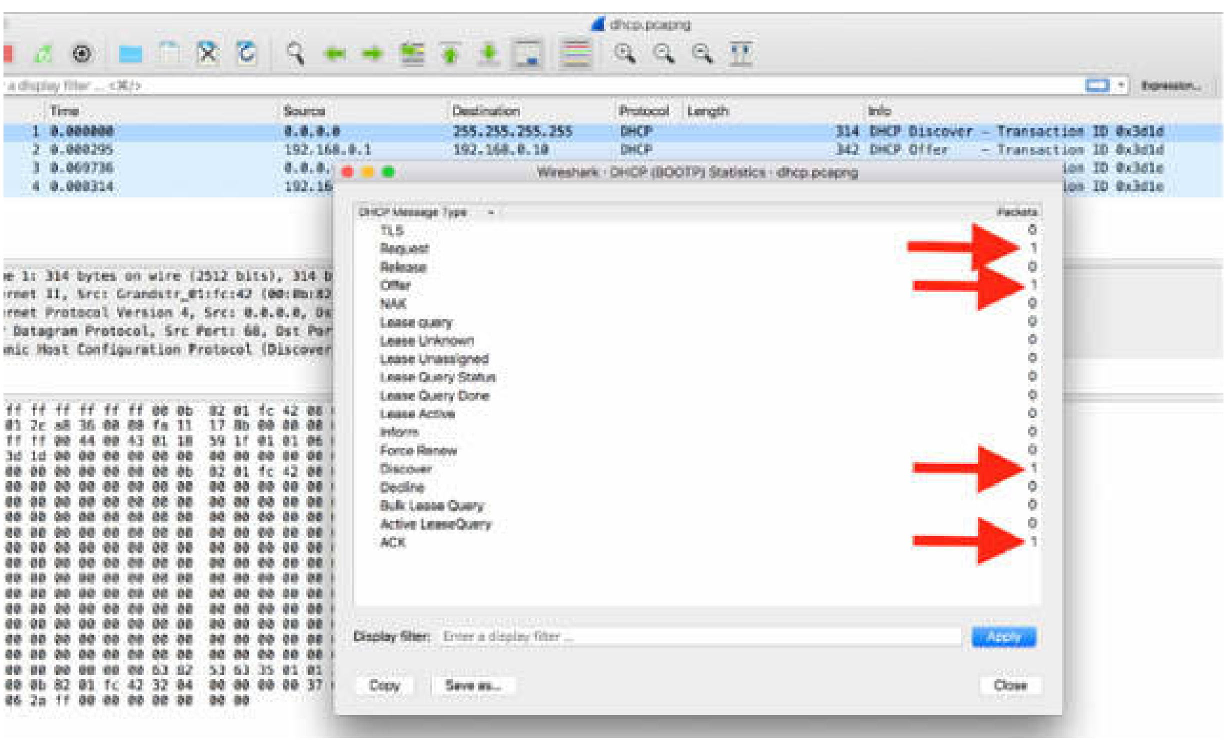
The Packet List pane appears as shown in the figure below.



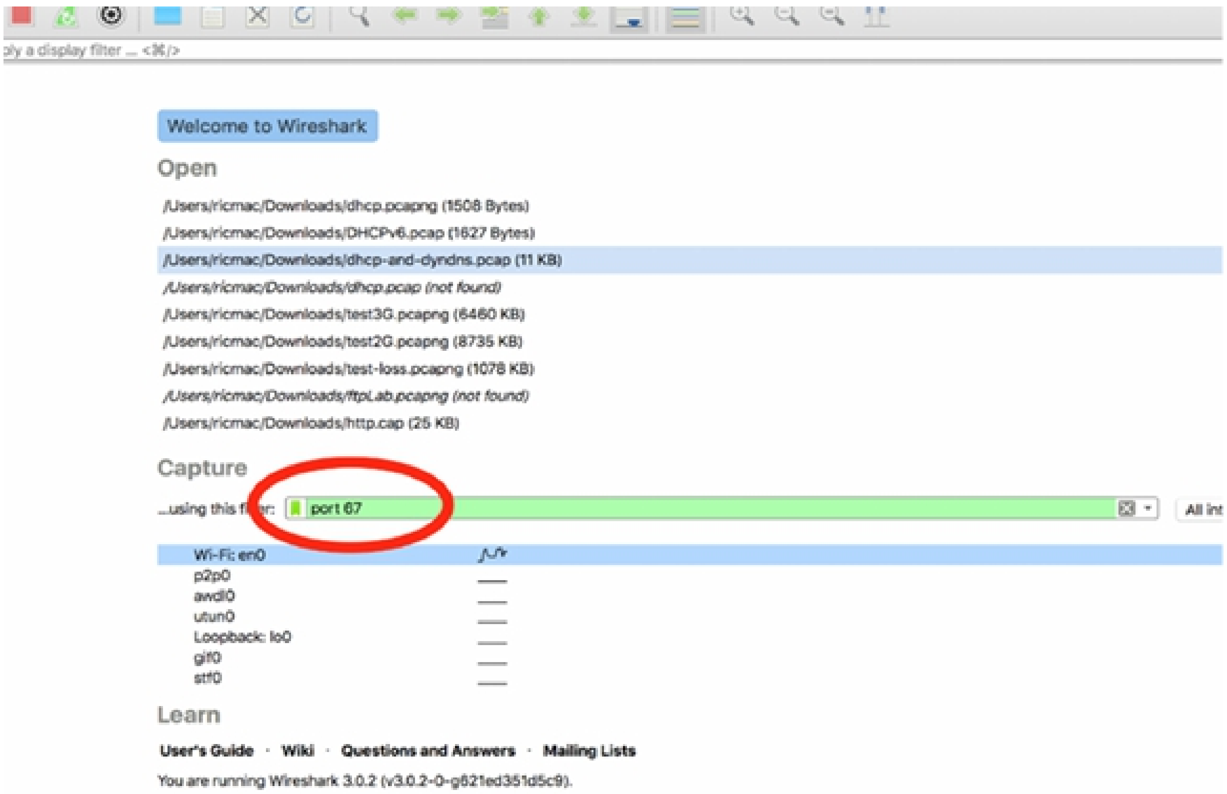
On the main menu, select Statistics > DHCP (BOOTP) Statistics to display the Statistics dialog box that summarizes the DHCPv4 message types in the trace file (currently, this feature does not support DHCPv6).



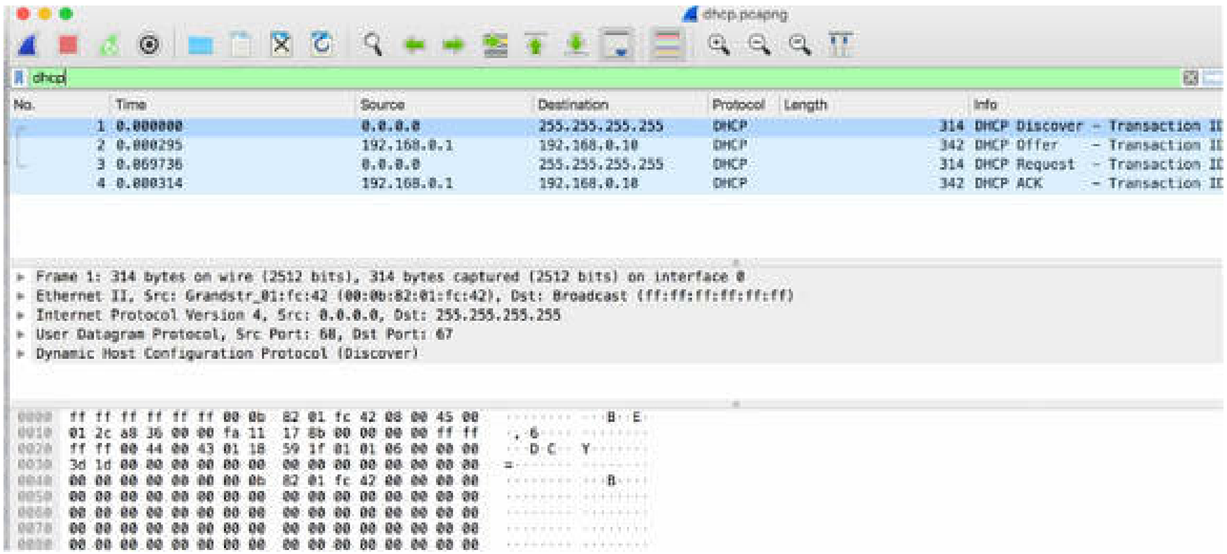
If you compare the information displayed in the Statistics dialog box with the messages in the trace file, you can simply match the number occurrences to the captured packets.



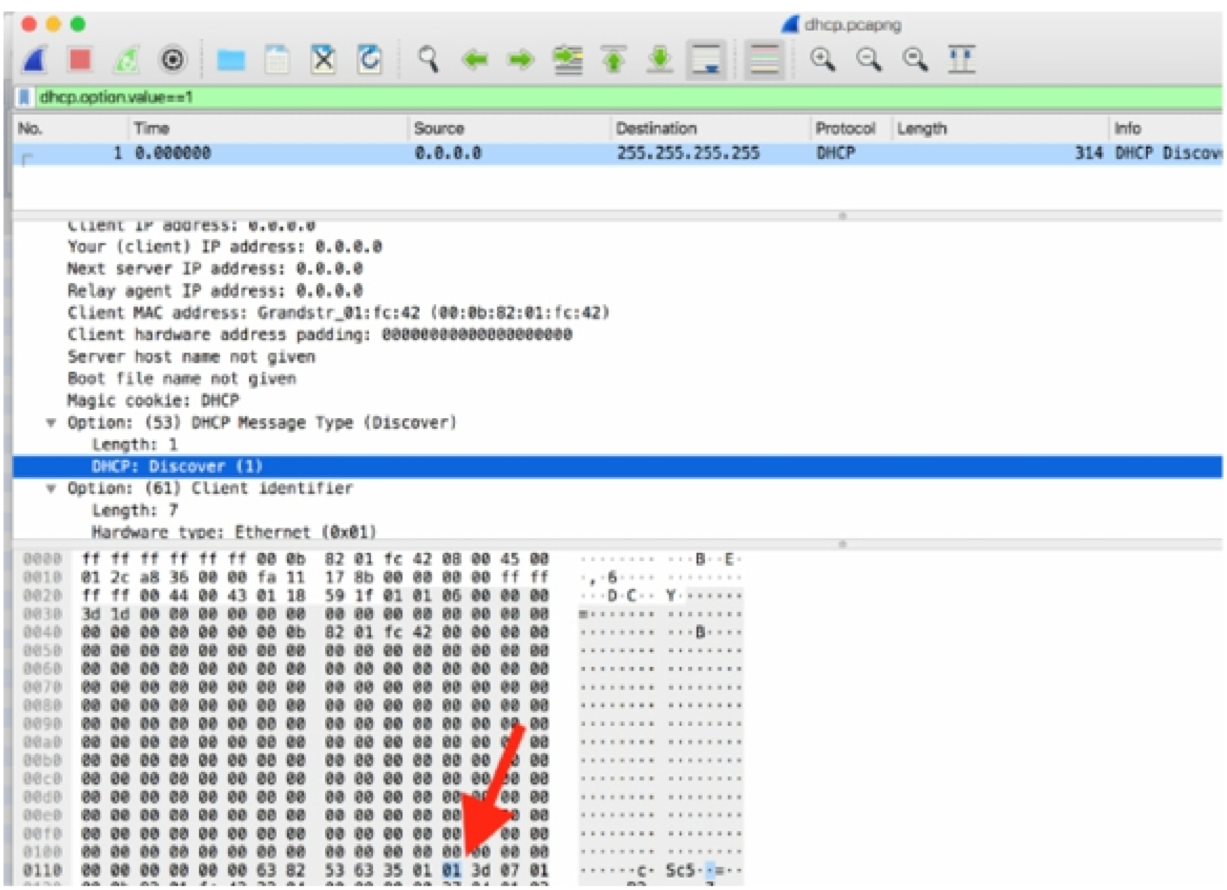
Task 2:
To capture DHCP packets, in the Capture Filter, enter port 67 , as shown in the figure below. Even though DHCP uses port 68 as the client port, the traffic always flows to or from port 67—the server port.



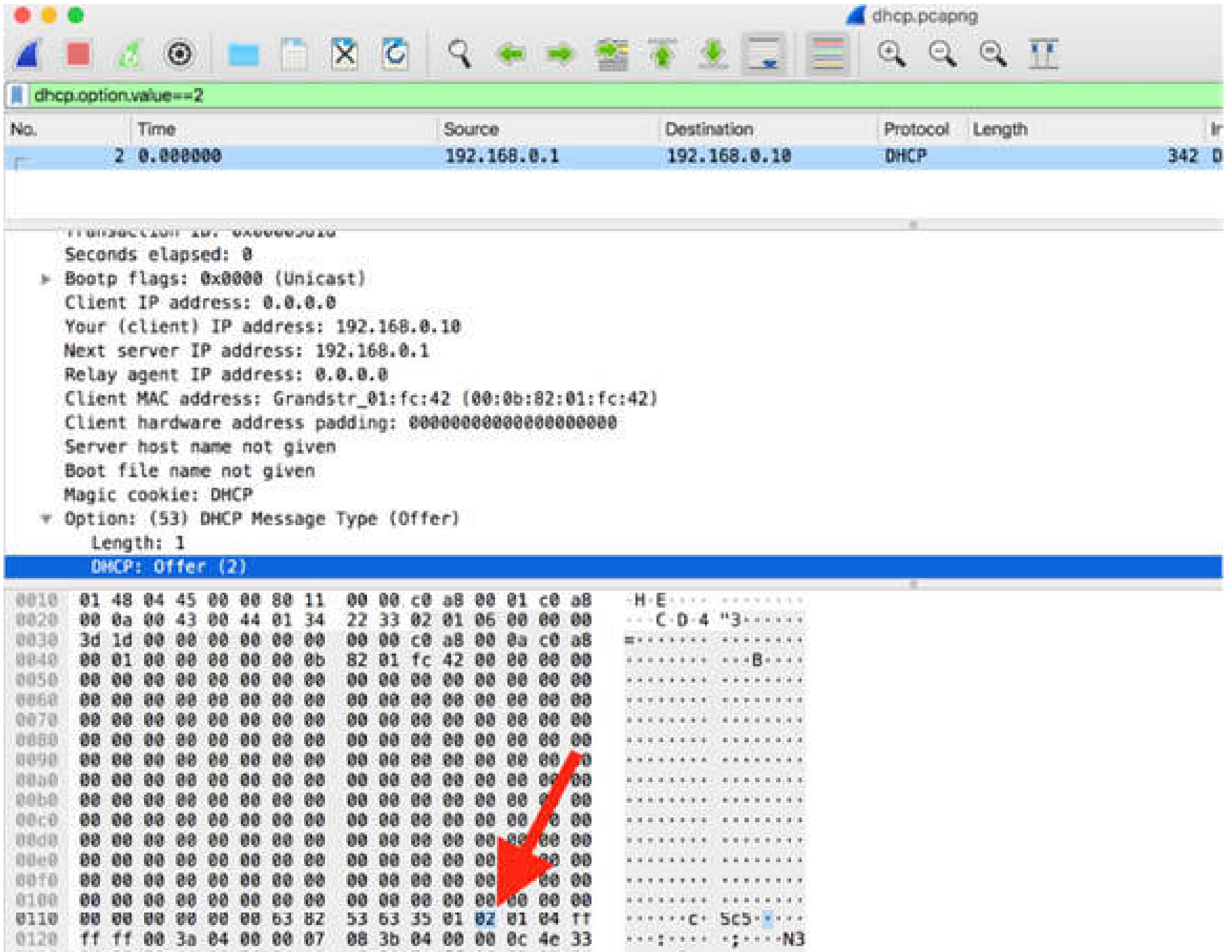
To display only DHCP packets in the Packet List pane, in the filter toolbar, enter dhcp .



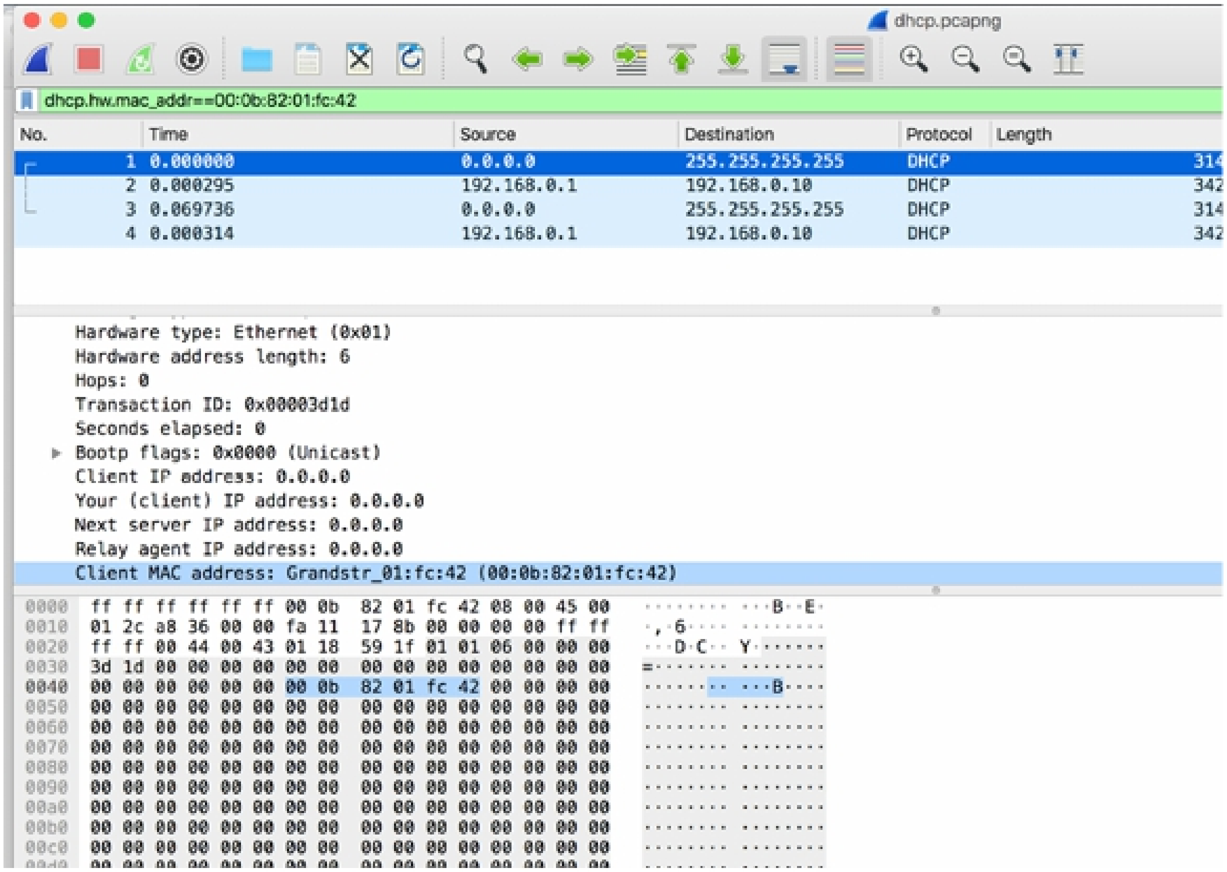
To display all DHCP discover packets in the Packet List pane, in the filter toolbar, enter dhcp.option.value==1 .



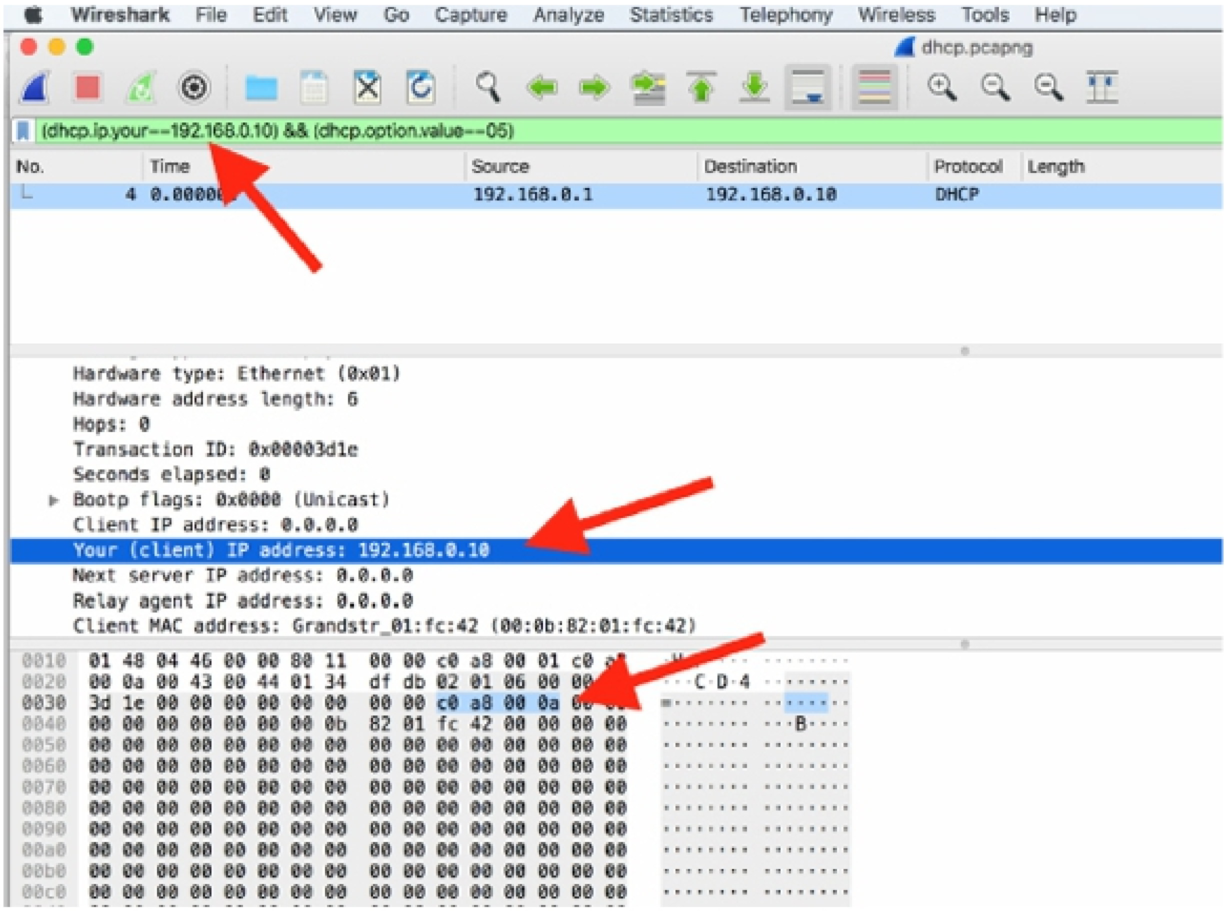
To display all DHCP offer packets in the Packet List pane, in the filter toolbar, enter dhcp.option.value==2 .



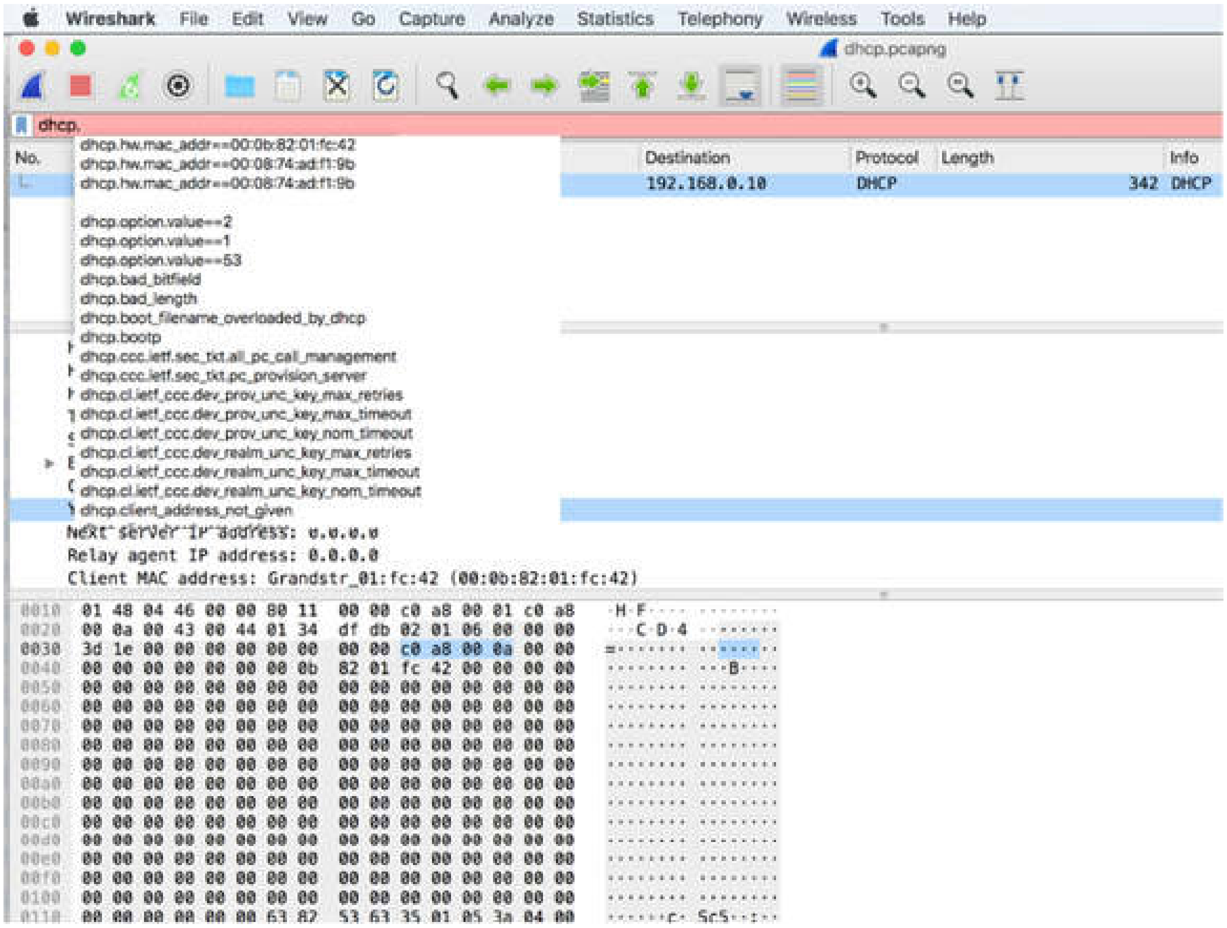
To display the DHCP messages that contain the MAC address 00:0b:82:01:fc:42, in the filter toolbar, enter dhcp.hw.mac_addr==00:0b:82:01:fc:42 .



To identify a DHCP ACK message to the client using IP address 192.168.0.10, in the filter toolbar, enter (dhcp.ip.your==192.168.0.10) && (dhcp.option.value==05) .

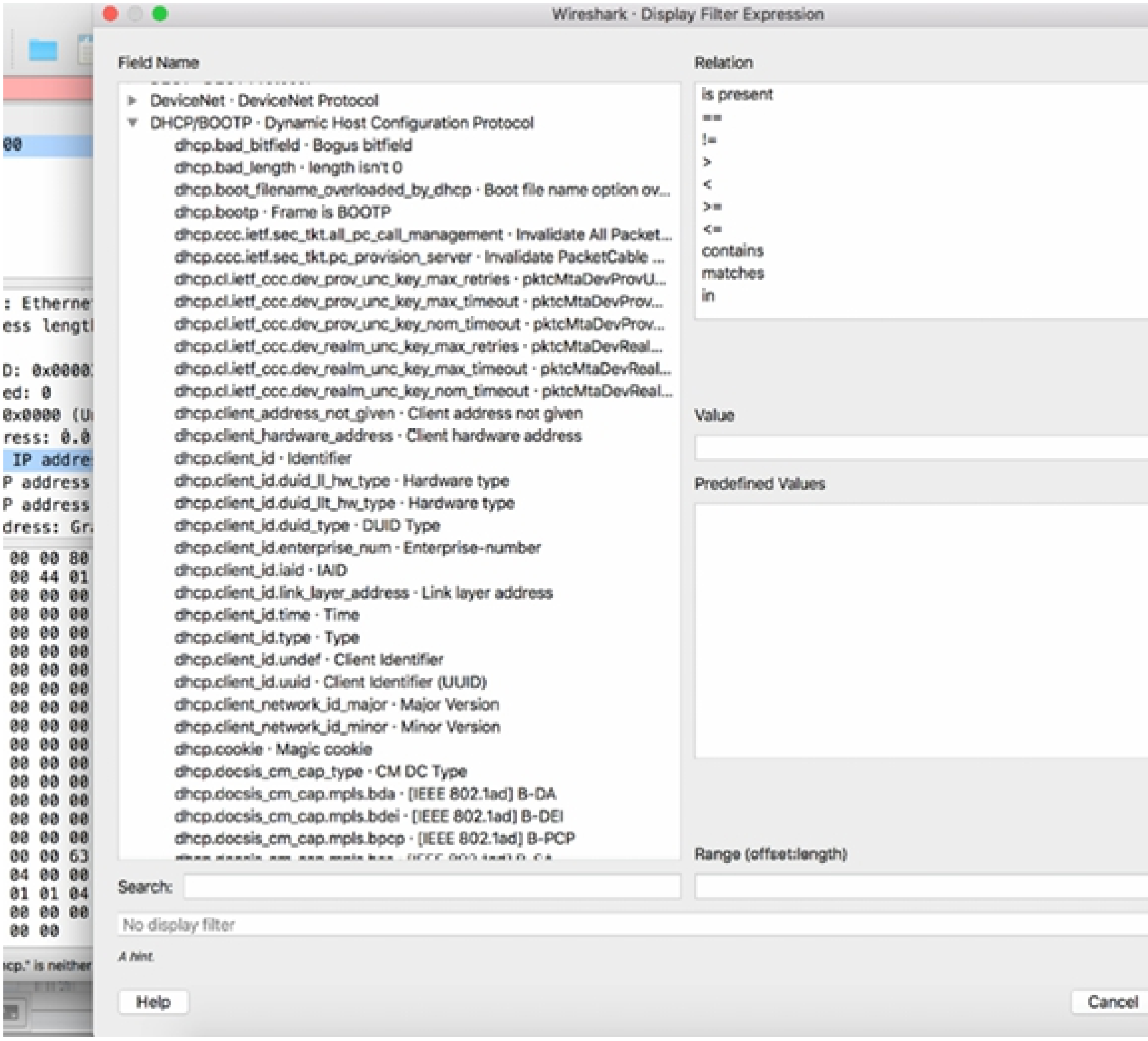


You can create a filter in two ways. The first way is to use the auto-complete feature available in the filter toolbar. For example, if you just type dhcp , the auto-complete feature displays a drop-down menu with all available DHCP filters, as shown in the figure below.



The second way to create a filter is by using the Display Filter Expression dialog box. Click the Expression button on the right of the filter toolbar. The Display Filter Expression dialog box is displayed.

To view a list of all DHCP filters, scroll down and click the DHCP/BOOTP —Dynamic Host Configuration Protocol field to open the tree view. Select the appropriate fields to create a display filter.



Notes:

To gain the necessary confidence in using various filters available, select the most appropriate display filter for different trace files. Use the Statistics dialog box to summarize the DHCP messages for different trace files and verify whether the counters reported are correct.

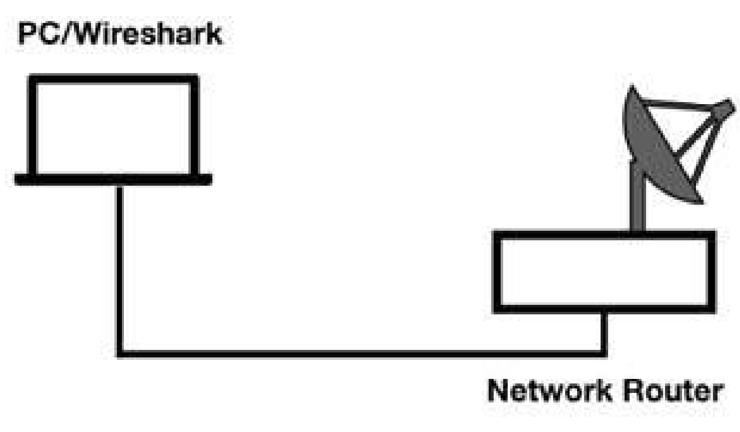
Lab 64. Hypertext Transfer Protocol

Lab Objective:
Learn how the Hypertext Transfer Protocol (HTTP) works and why is it used.

Lab Purpose:
Understand the main purpose of HTTP and the features of the protocol.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

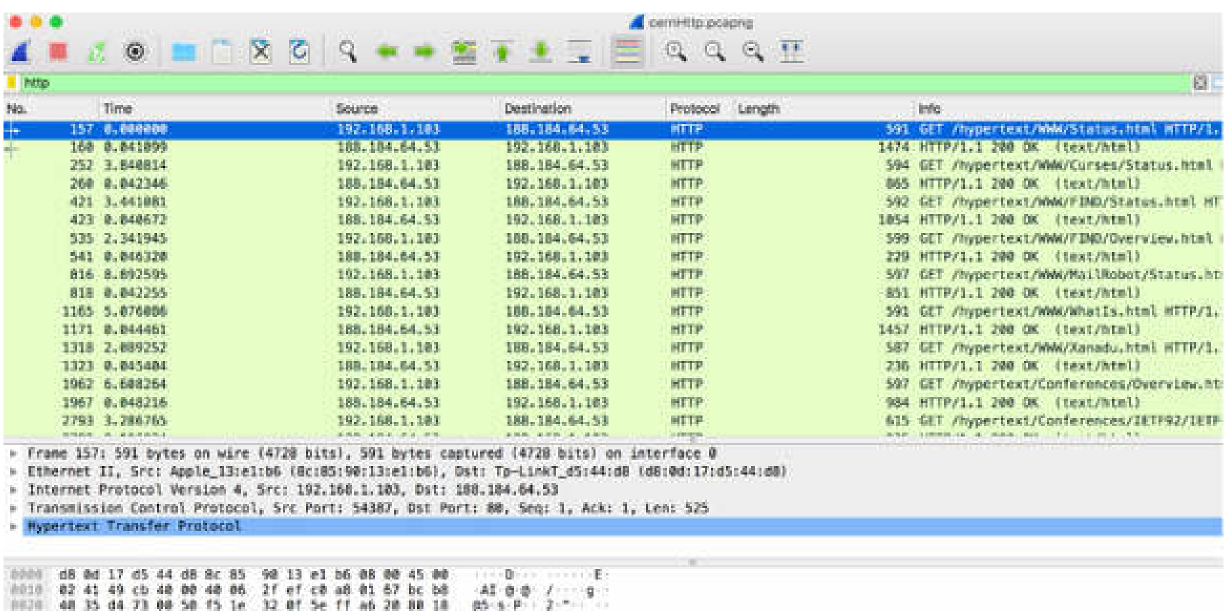
Task 1:
HTTP is referred to as a "distributed hypermedia information distribution application." HTTP application is used when you browse the internet (in an unsecured way). There are different versions of HTTP—the current version is v2.0; the most used version is v1.1 but sometimes v1.0 is also used.

Normal HTTP communication uses a request/response communication model in which a client sends a request to an HTTP server and the server responds with the status code.

Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

In a web browser, go to http://info.cern.ch/ and inspect some links on the main page. Stop the capture in Wireshark and save the file, naming it cernHttp.pcapng.

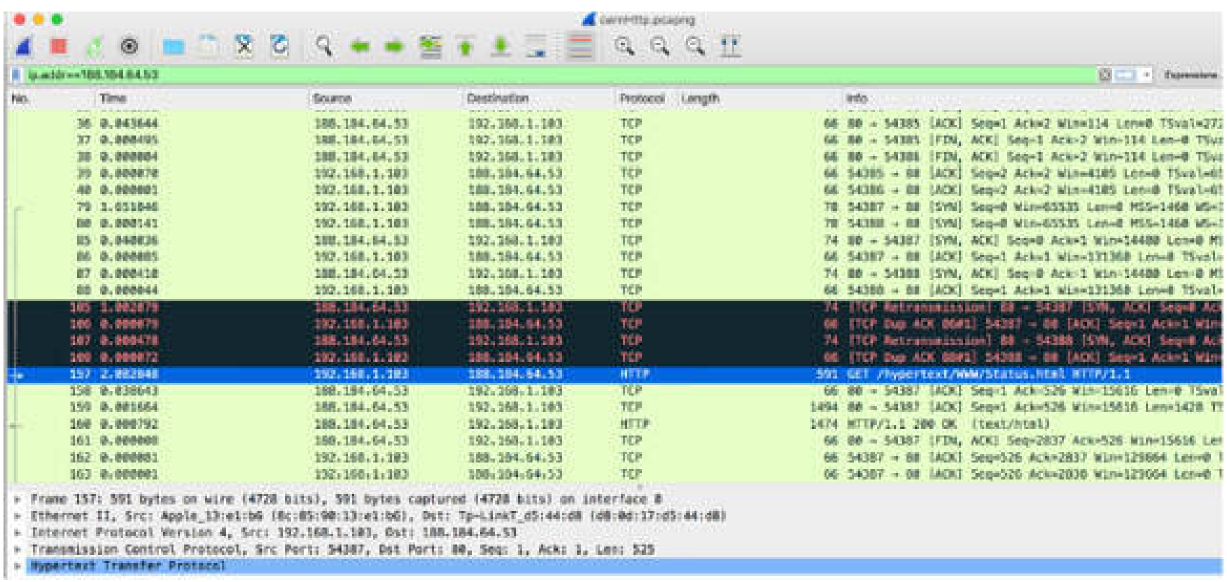
In the filter toolbar, enter http . The results will be similar to the figure below.



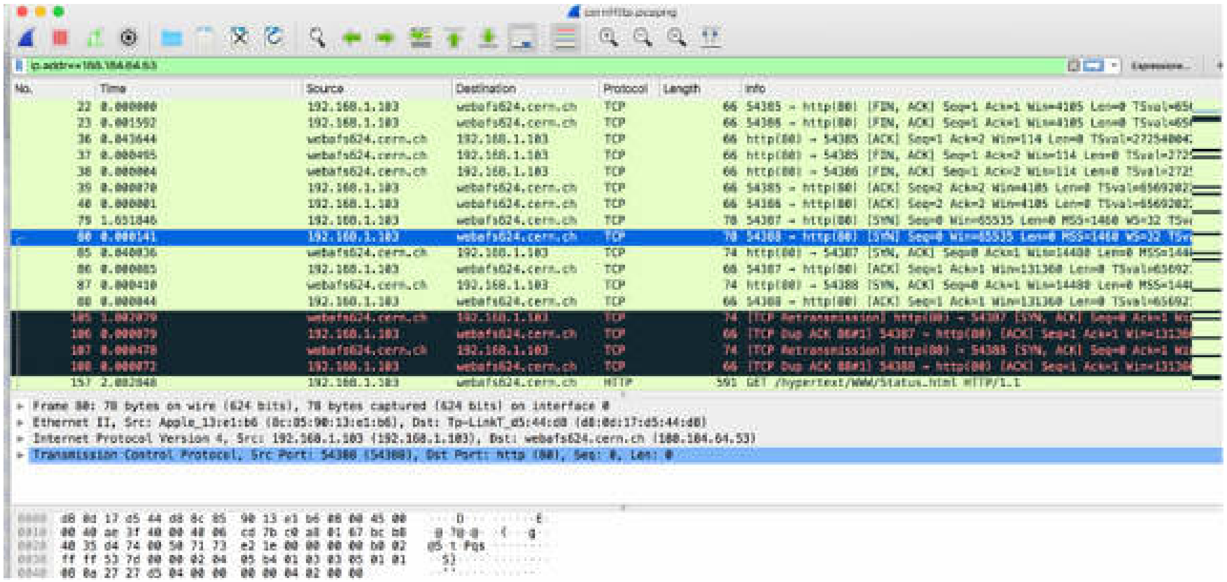
In the figure above, only the HTTP packets are displayed in the Packet List pane.

If you apply the display filter ip.addr == 188.184.64.53 , all packets exchanged with the remote server are displayed in the Packet List pane. In this case, the remote server IP address is 188.184.64.53. When you select the first HTTP packet in the Packet List pane, this IP address is also displayed as the destination in the Packet Details pane. Note that this IP address may change if the server is moved. You can check the HTTP output and verify the IP address.

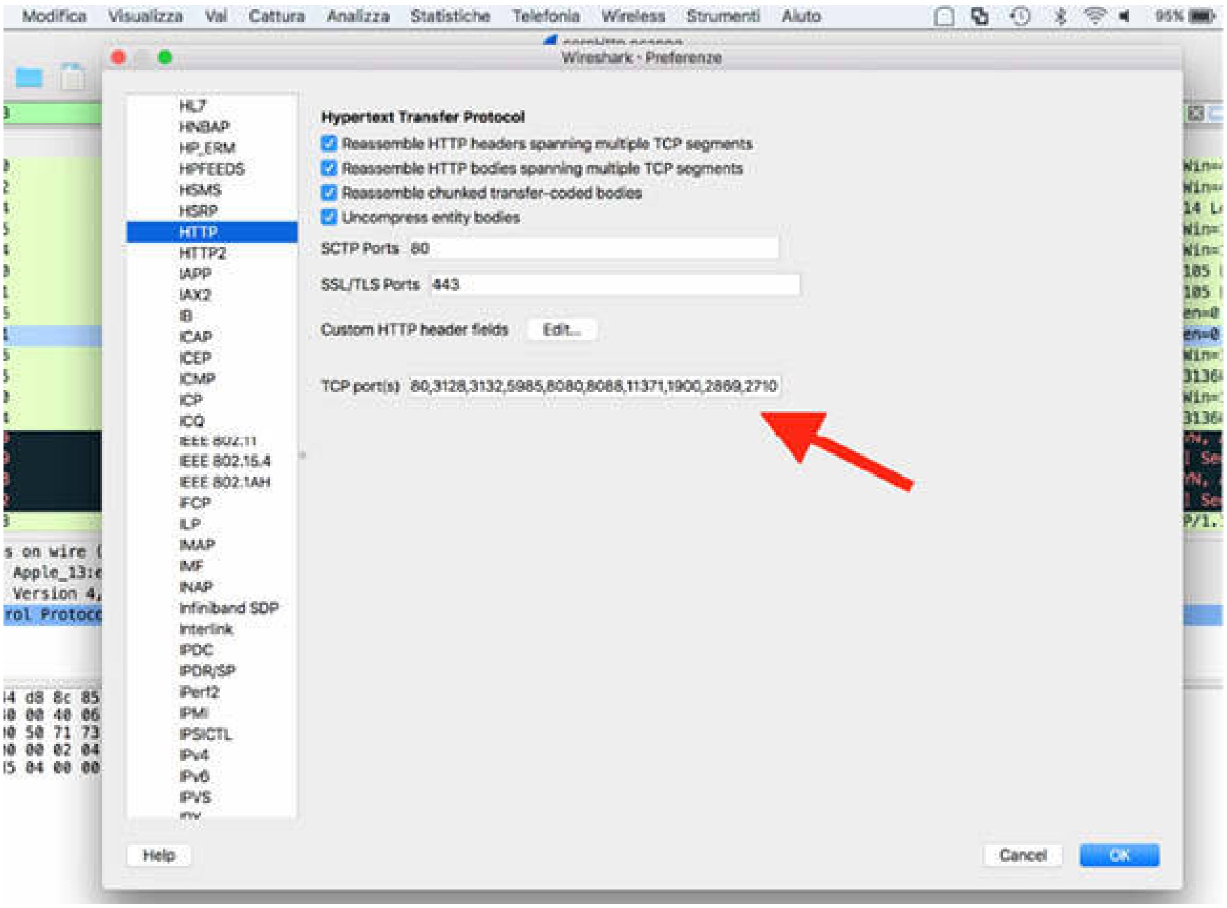
The figure below indicates packet loss and very poor response time. Moreover, it shows that some TCP retransmission occurred before the HTTP communication started.



As shown in the figure below, the client made a three-way TCP handshake from port 54388 to port 80. In the Info column, packets #80, #85, and #86 are listed as HTTP because transport name resolution is enabled.

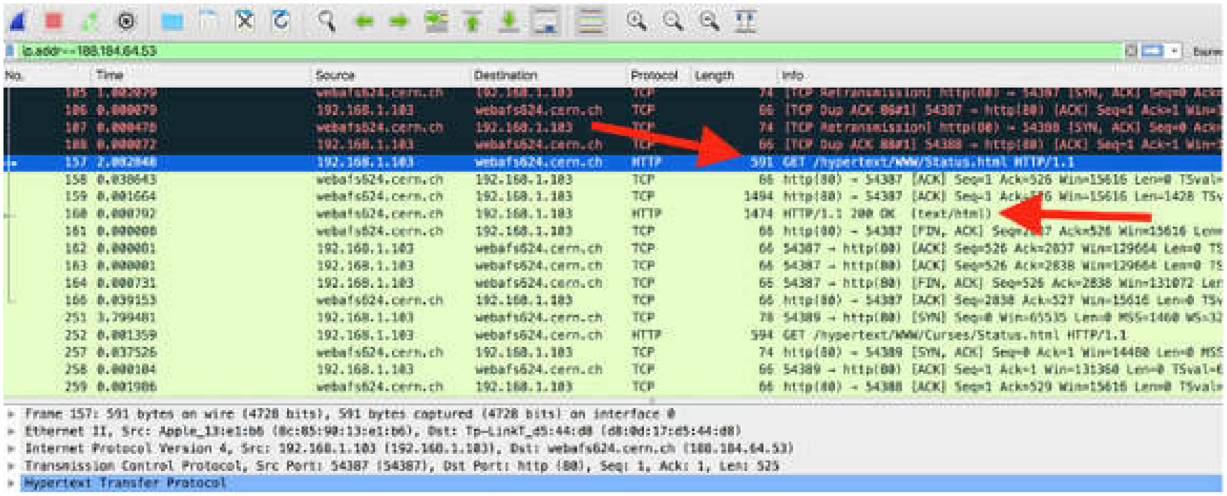


By default, Wireshark is configured to dissect HTTP on the following ten ports: 80, 3128, 3132, 5985, 8080, 8088, 11371, 1900, 2869, and 2710. However, other ports can also be used for HTTP communication. You can specify the ports in the Preferences dialog box. On the main menu, select Edit > Preferences. In the left tree view, select HTTP and then enter the ports in the TCP port(s) box.



To capture HTTP traffic that is running on another port, simply add the port number to the HTTP preferences.

After the TCP connection is established successfully, the client makes an HTTP GET request for “/” (packet #157). The server responds with the status code 200 OK and begins sending the contents of the main page to the client.



The following are all available status codes from the HTTP Status Code Registry, grouped by type such as info, success, error.

1xx Informational

100 Continue
101 Switching Protocols
102 Processing

2xx Success

200 OK
201 Created
202 Accepted
203 Non-Authoritative Information
204 No Content
205 Reset Content
206 Partial Content
207 Multi-Status
208 Already Reported
226 IM Used

3xx Redirection

300 Multiple Choices
301 Moved Permanently
302 Found
303 See Other

304 Not Modified
305 Use Proxy
306 Reserved
307 Temporary Redirect
308 Permanent Redirect

4xx Client Error

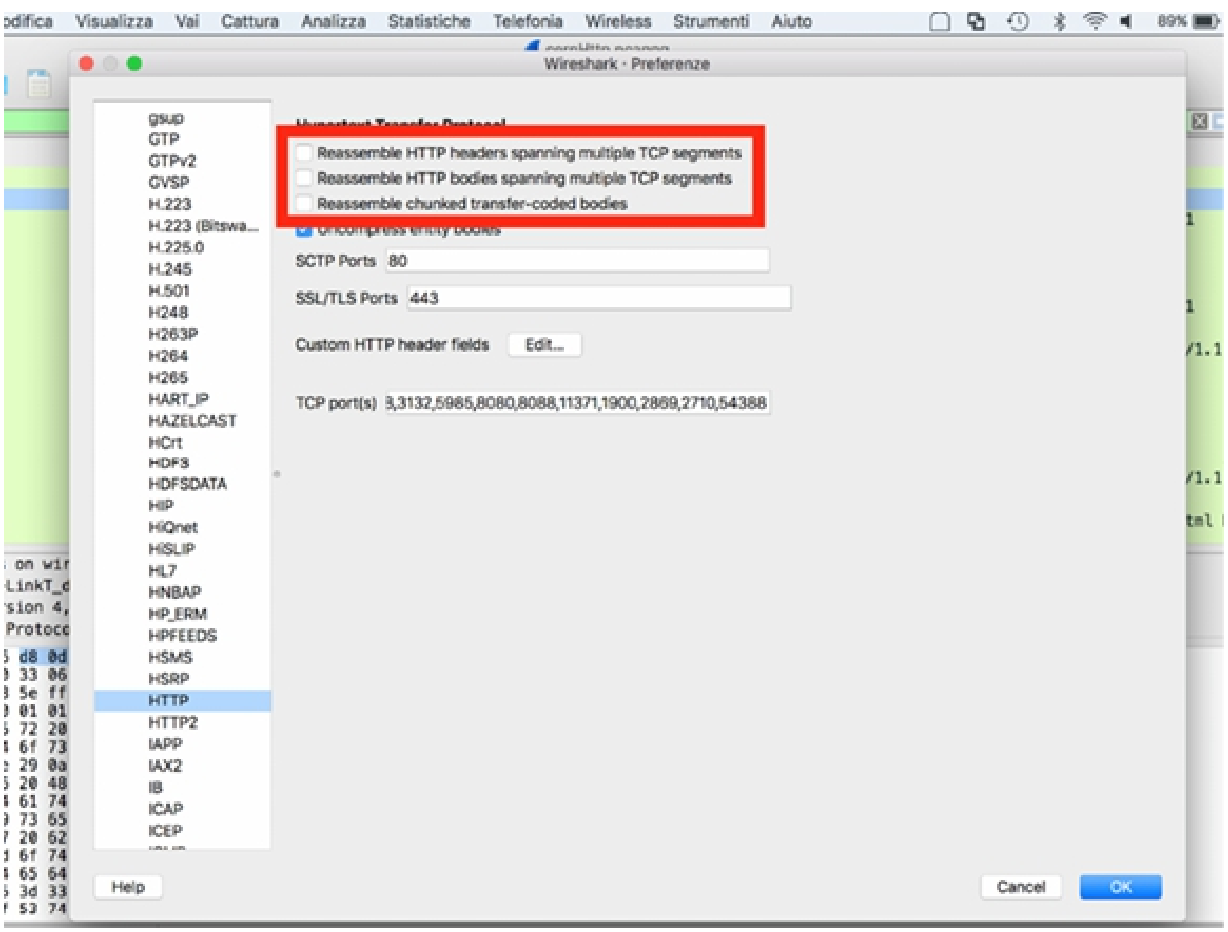
400 Bad Request
401 Unauthorized
402 Payment Required
403 Forbidden
404 Not Found
405 Method Not Allowed
406 Not Acceptable
407 Proxy Authentication Required
408 Request Timeout
409 Conflict
410 Gone
411 Length Required
412 Precondition Failed
413 Request Entity Too Large
414 Request-URI Too Long
415 Unsupported Media Type
416 Requested Range Cannot Be Satisfied
417 Expectation Failed
422 Unprocessable Entity
423 Locked
424 Failed Dependency
425 Reserved for WebDAV
426 Upgrade Required
428 Precondition Required
429 Too Many Requests
431 Request Header Fields Too Large

5xx Server Error

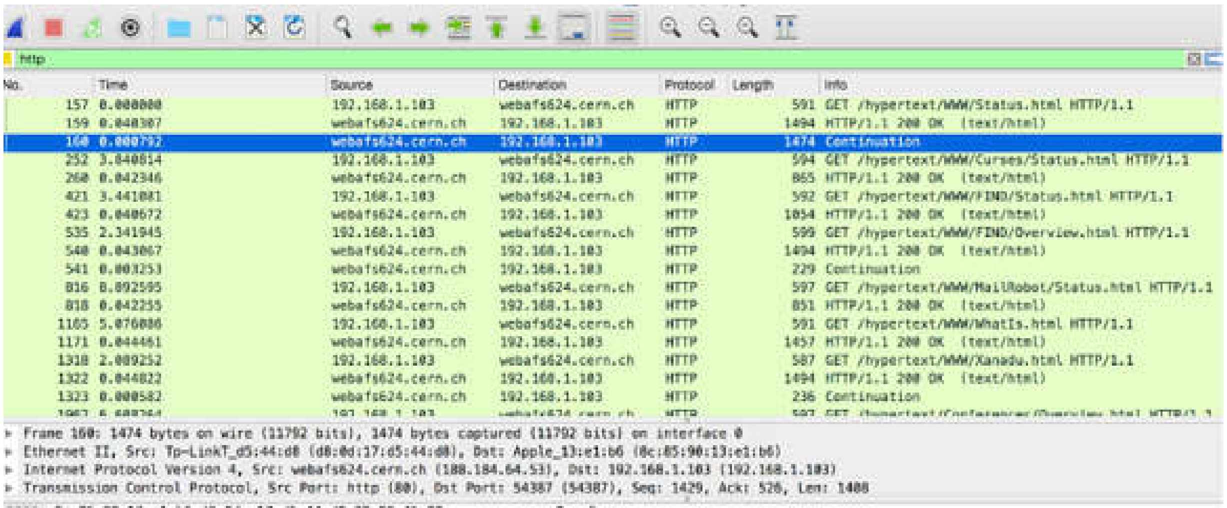
500 Internal Server Error
501 Not Implemented

502 Bad Gateway
503 Service Unavailable
504 Gateway Timeout
505 HTTP Version Not Supported
506 Variant Also Negotiates (Experimental)
507 Insufficient Storage
508 Loop Detected
510 Not Extended
511 Network Authentication Required

Task 2:
To make the HTTP view more clear, in the Preferences dialog box, select HTTP and clear the options related to "Allow subdissector to reassemble TCP streams", as shown in the figure below.



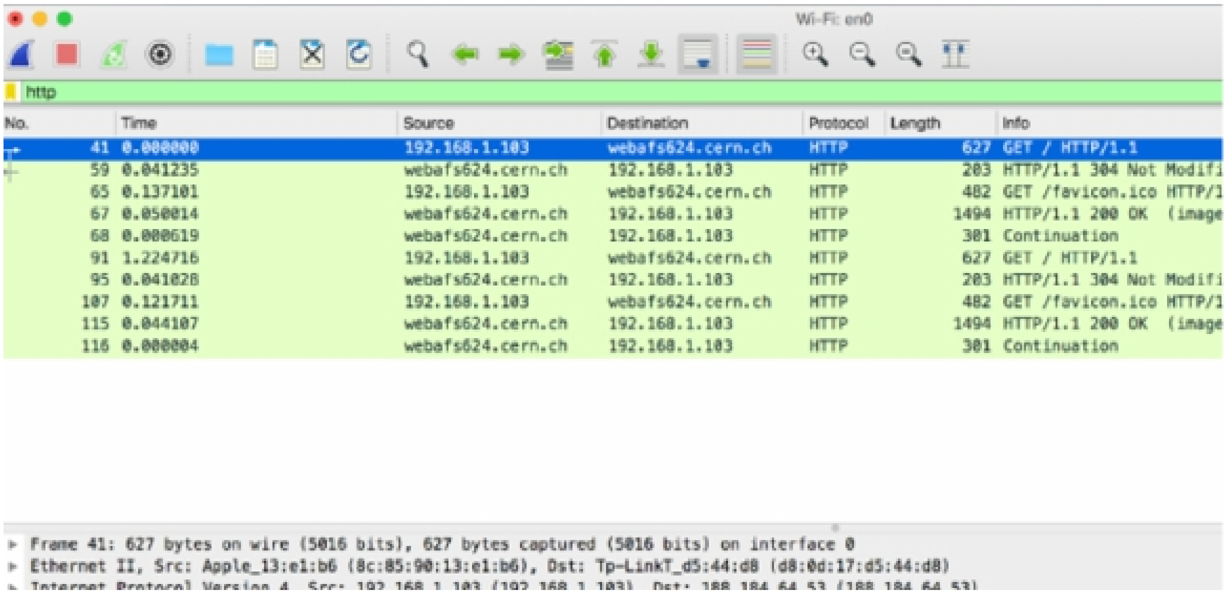
The result of this preference change is displayed in the Packet List pane, as shown in the figure below. Each individual HTTP message is now displayed as a single packet.



Task 3:
Start a capture again on the active interface. In a web browser, reload http://info.cern.ch/ . Stop the capture.

If an HTTP client has visited a page recently and that page is cached locally, the client may send the IfModified-Since parameter and provide a date and time of the previous page download. If the page is not modified, the server responds with the 304—Not Modified code. The server does not resend the page that is already cached. This is applicable in this case because you recently visited the home page. When analyzing HTTP performance, this is an important aspect of HTTP to understand. In fact, when analyzing the capture packets, you must ensure that the pages are not reloaded from the cache. Otherwise, you won't be able to see the full page download.

The following figure shows this scenario.



Notes:
Repeat the previous steps on different websites using the HTTP protocol and observe the HTTP messages. Identify the connection establishment and try to understand whether the performance of the server is good.

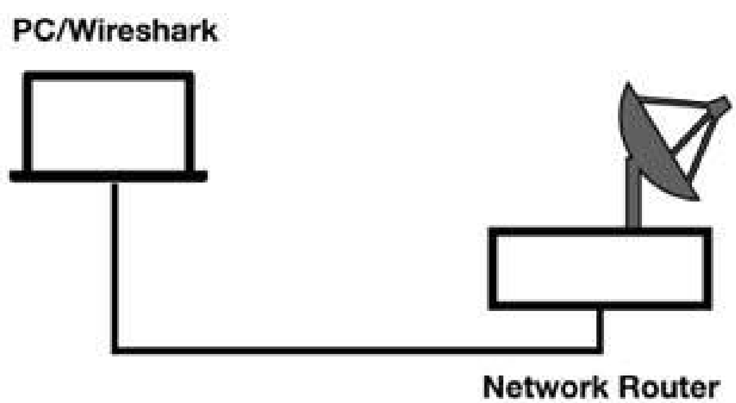
Lab 65. HTTP Problems

Lab Objective:
Learn about the more common HTTP problems

Lab Purpose:
Learn how to detect and analyze the more common HTTP problems.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

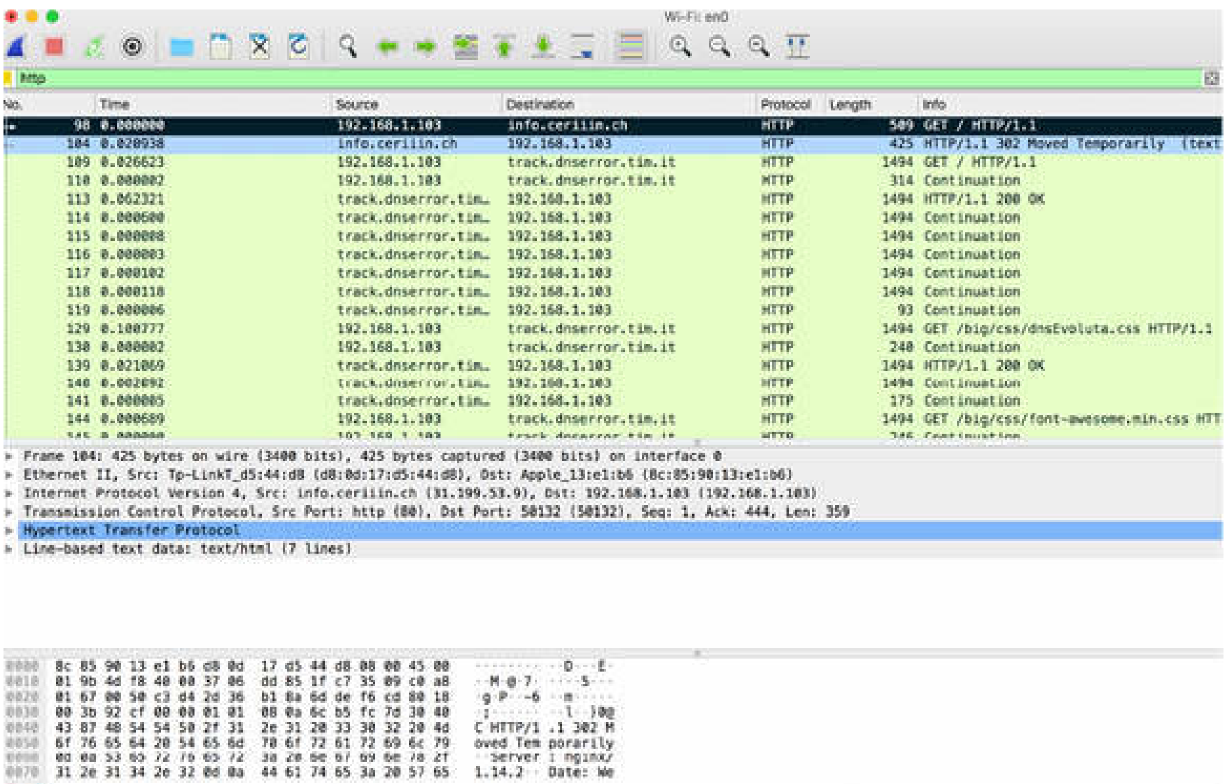
Task 1:
There are a lot of problems that can occur in HTTP communication, starting from the site name resolution to the issues with the TCP connection

process.

Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column, and then capture the traffic for a few minutes.

In a web browser, go to **http://info.ceriiin.ch/** which is an incorrect address. As a result, the page cannot be loaded and the site cannot be accessed.

Stop the capture and save the file. In the filter toolbar, enter http . The results are displayed in the Packet List pane, as shown in the figure below.

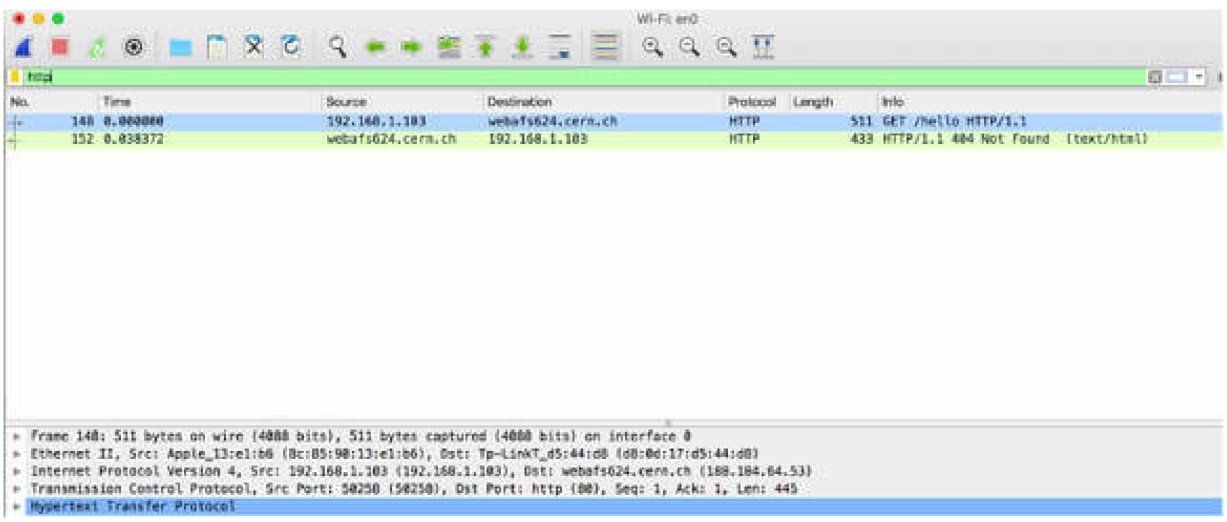


The results in the Packet List pane indicate that there is a DNS error. The (incorrect) name cannot be resolved which generates the DNS Name error. This scenario also highlights the importance of DNS traffic when analyzing web browsing problems.

Another problem scenario is when the HTTP daemon is not running on the web server. When this happens, the server responds with a TCP RST/ACK message to the client's SYN message. The connection cannot be established. This scenario shows that TCP communication should also be taken into account when analyzing web browsing problems.

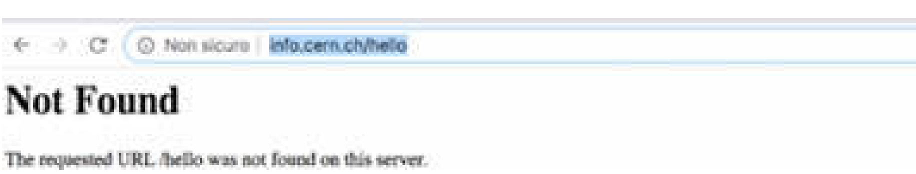
Task 2:
Again start capturing traffic in Wireshark. In a web browser, go to http://info.cern.ch/hello . In this link, the last component of the path—the hello page—does not exist.

Stop the capture and save the file. In the filter toolbar, enter http . The results are displayed in the Packet List pane, as shown in the figure below.



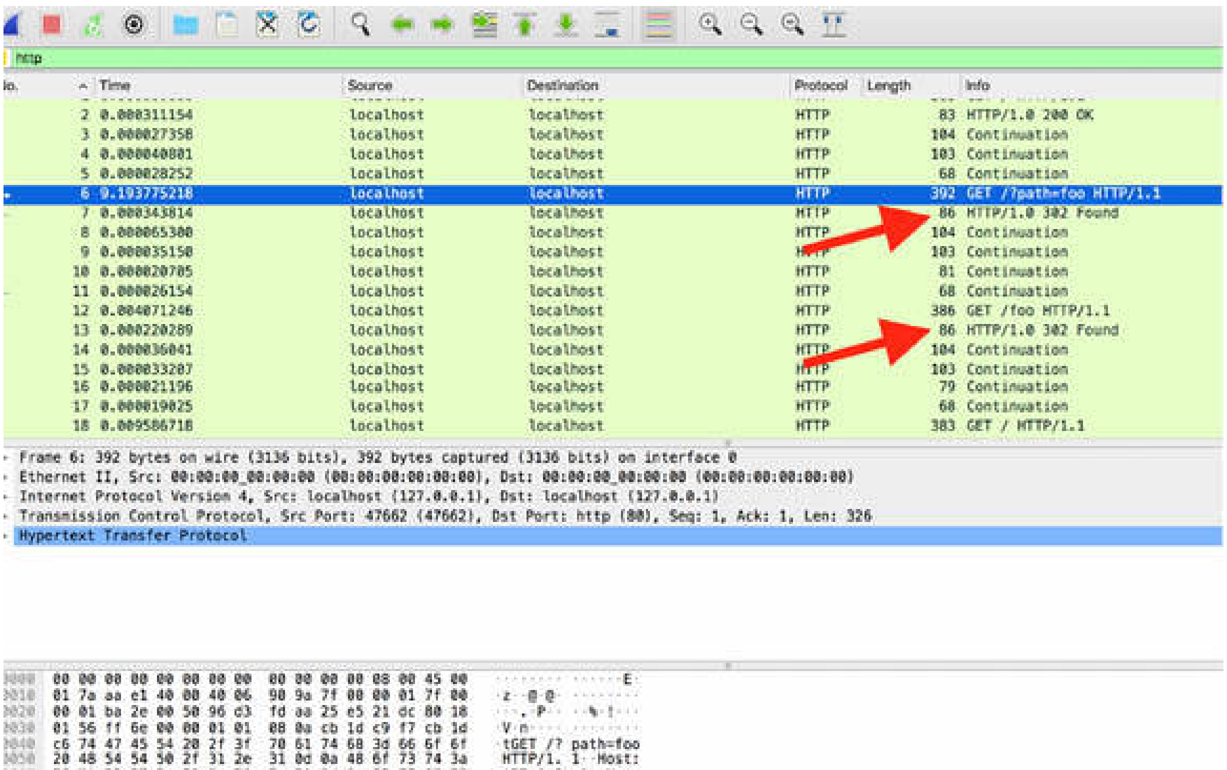
The HTTP client (web browser) connects successfully to the HTTP server but then requests a non-existent page. The web server generates the HTTP 404—Not Found error.

The figure below shows the page displayed in the web browser.



In some cases, redirection services replace the standard 404—Not Found message with suggested links or redirect the HTTP client to a different site.

To try this scenario, download the http_redirects.pcapng file from https://wiki.wireshark.org/SampleCaptures#HyperText_Transport_Protocol_.28HTTP.29 . Open the file in Wireshark. In the filter toolbar, enter `http` . The results are displayed, as shown in the figure below. There are a lot of HTTP 302 redirects messages that suggest another site for navigation.



Notes:
If your DNS server can’t find a website, you may not get an HTTP error.

Repeat the previous steps on a different website. Modify the original web address and analyze the HTTP communication results. Access a non-existent page on an HTTP server and verify the response of the server to gain confidence in interpreting HTTP error messages.

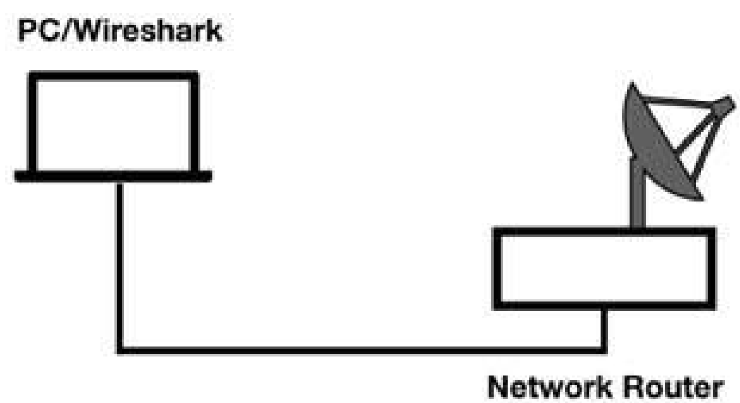
Lab 66. HTTP Problems

Lab Objective:
Learn about the more common HTTP problems.

Lab Purpose:
Learn how to detect and analyze the more common HTTP problems.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

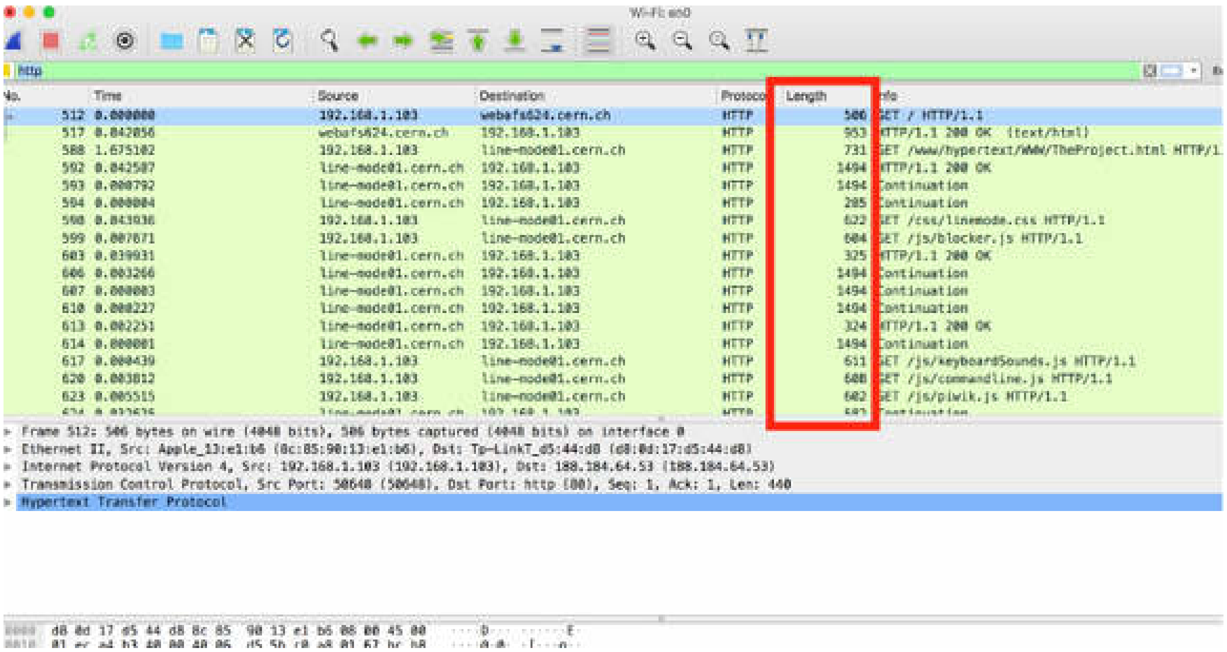


Lab Walkthrough:

Task 1:
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic

column, and then capture the traffic for a few minutes. In a web browser, go to http://info.cern.ch/ , and inspect some of the links on the main page. Stop the capture in Wireshark and save the file.

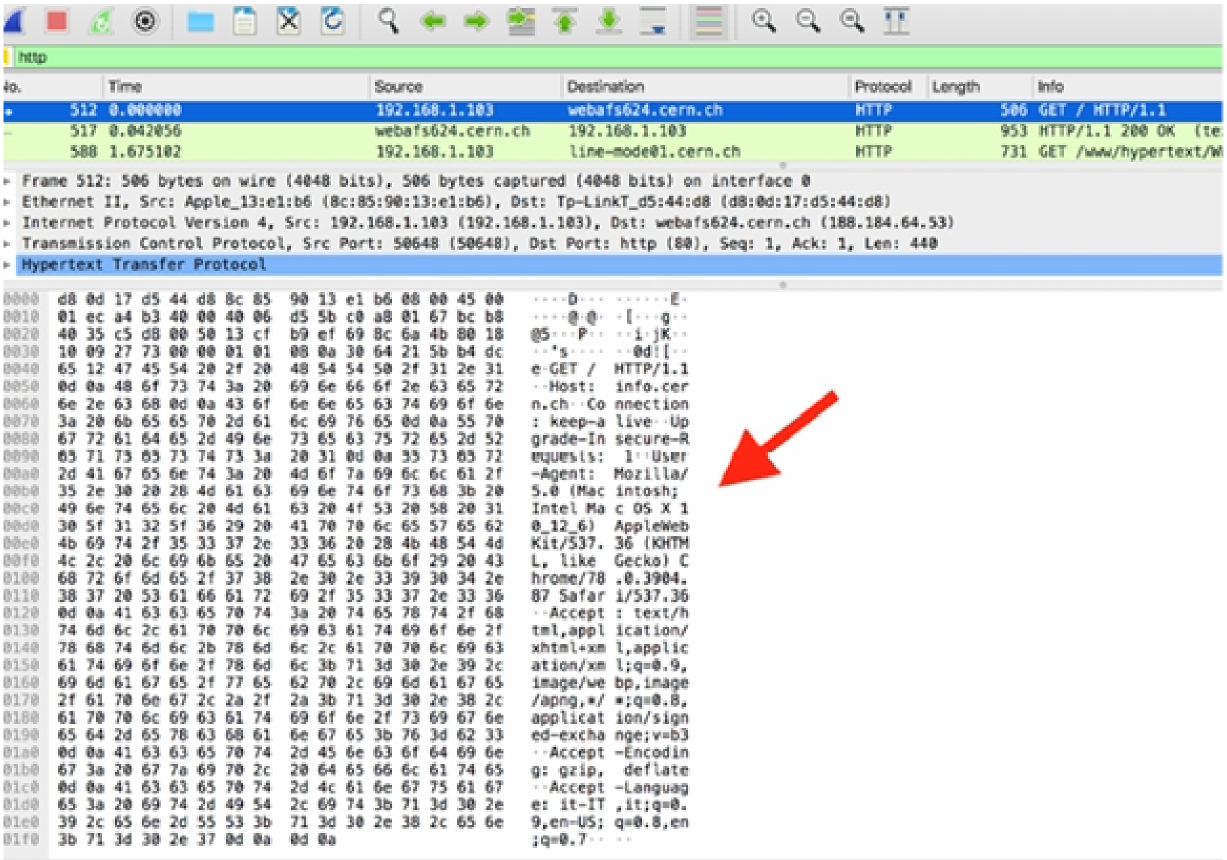
In the filter toolbar, enter http . The results are displayed in the Packet List pane, as shown in the figure below.



The results in the Packet List pane indicate that HTTP packets are of variable length. The HTTP request consists of a method that defines the purpose of the HTTP request. The HTTP response contains a numerical response code, referred to as a status code.

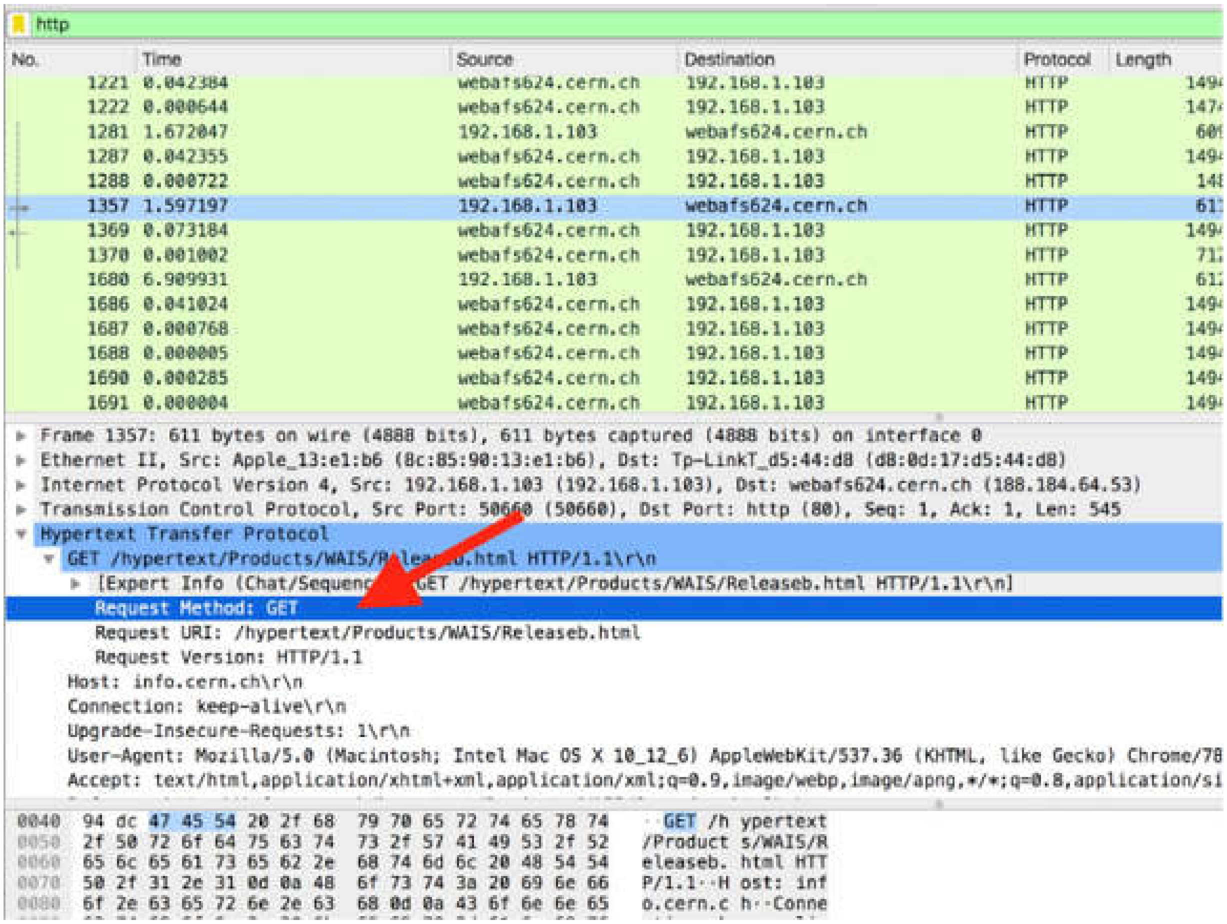
As shown in the figure below, when you select the first GET request, the related information is displayed in the Packet Bytes pane.

The GET request contains relevant information, such as the name of the target host, details about the browser issuing this GET request, and information about what data types and formats the browser accepts.



Task 2:

The first field in the Packet Details pane is the method field. A method (also known as command) defines the purpose of the HTTP packet. In the figure below, the GET method is displayed.

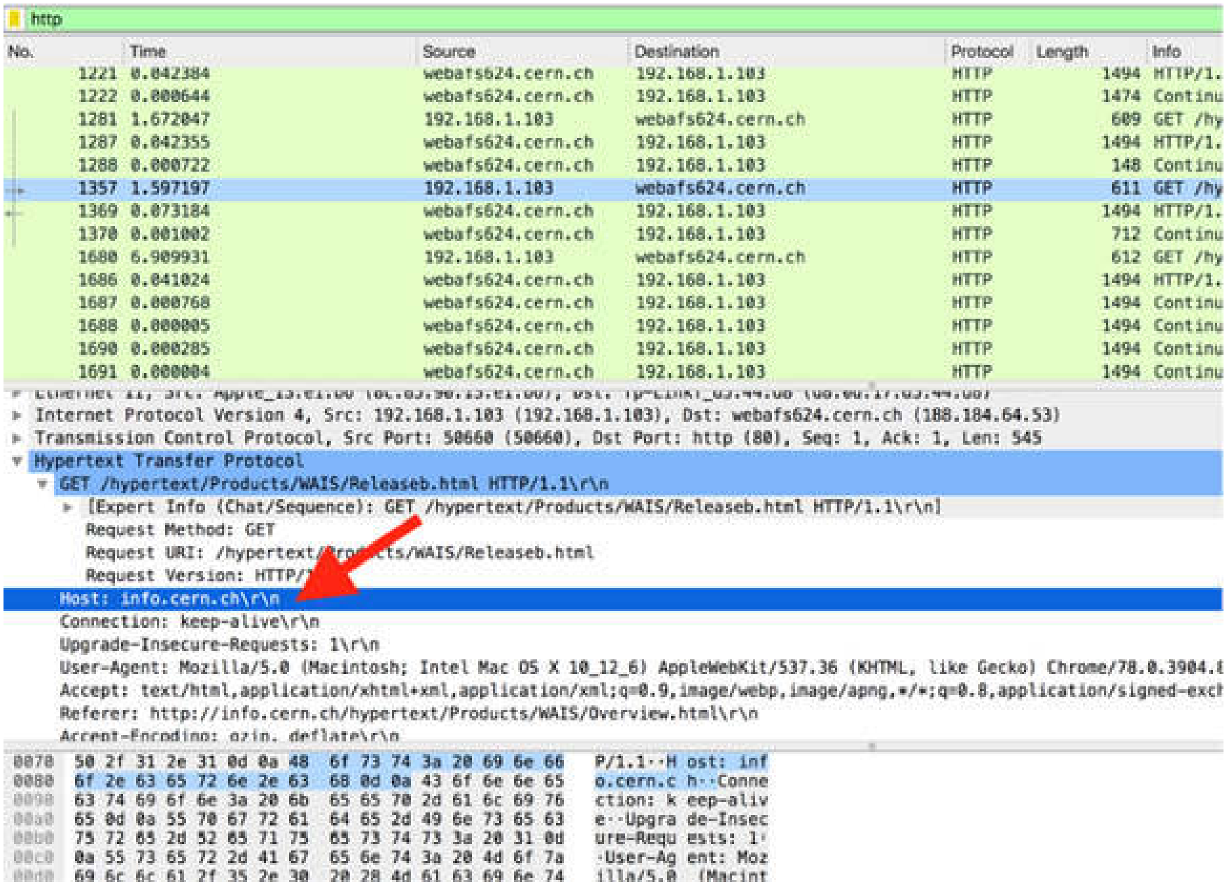


The GET method is used to retrieve information defined by the Uniform Resource Indicator(URI) field. Other HTTP methods are:

- HEAD: It is used to retrieve the metadata related to the desired URI.
- POST: It is used to send data to the HTTP server.
- OPTIONS: It is used to determine the options associated with a resource.
- PUT: It is used to send data to the HTTP server.
- DELETE: It is used to delete the resource defined by the URI.
- TRACE: It is used to invoke a remote loopback so the client can see what did the server receive from the client.
- CONNECT: It is used to connect to a proxy device.

Task 3:

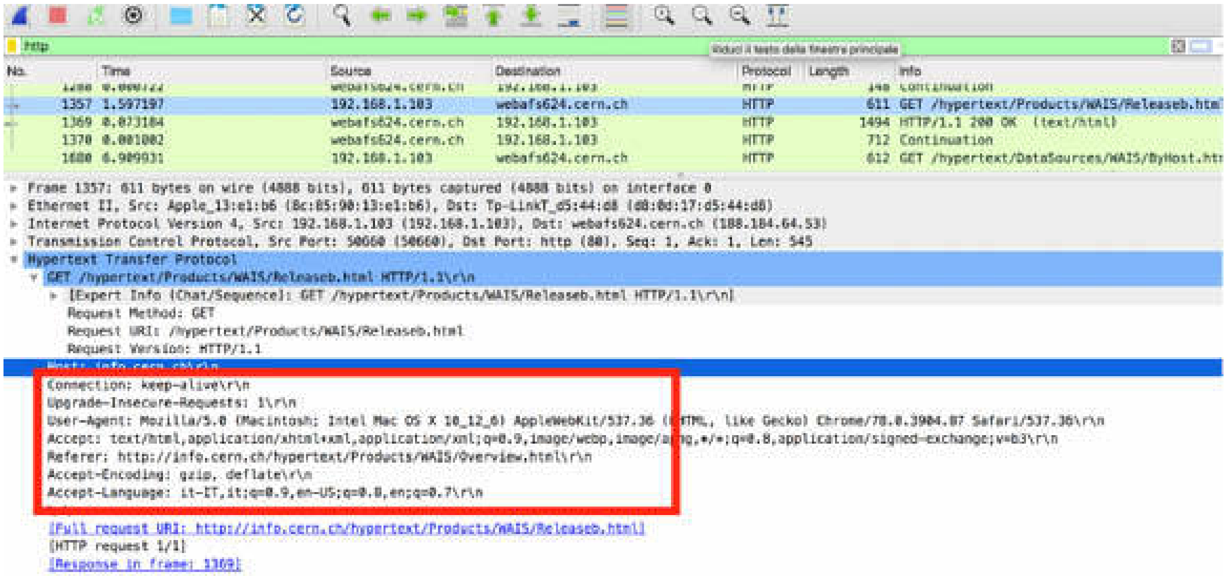
The Host field is the next field in the Packet Details pane. It is required in all HTTP/1.1 request messages. The Host field identifies the target internet host and the port number of the resource being requested. In the example of the trace file captured previously and displayed in the figure below, the host is info.cern.ch.



If no port number is specified, the default port for the service (for example, port 80 for HTTP) is used.

Task 4:

The next fields in the Packet Details pane are known as request modifiers. The response modifiers are used in HTTP requests and responses to provide details for the request. In the figure below, the request modifier fields are shown for an HTTP GET request.



Some of the common request modifiers are :

- Accept: Acceptable content types
- Accept-Charset: Acceptable character sets
- Accept-Encoding: Acceptable encodings
- Accept-Language: Acceptable languages
- Accept-Ranges: Server can accept range requests
- Authorization: Authentication credentials for HTTP authentication
- Cache-Control: Caching directives
- Connection: Type of connection preferred by the user agent
- Cookie: HTTP cookie
- Content-Length: Length of the request body (bytes)
- Content-Type: Mime type of body (used with POST and PUT requests)
- Date: Date and time message sent
- Expect: Defines server behavior expected by the client
- If-Match: Perform action if client-supplied information matches
- IfModified-Since: Provide date/time of cached data; 304—Not Modified if current
- If-Range: Request for a range of missing information

- If-Unmodified-Since: Only send if unmodified since certain date/time
- Max-Forwards: Limit number of forwards through proxies or gateways
- Proxy-Authorization: Authorization credentials for proxy connection
- Range: Request only part of an entity
- Referer: Address of previous website linking to the current one
- TE: Transfer encodings accepted
- UserAgent: User agent—typically browser and operating system
- Via: Proxies traversed

Notes:

Repeat the previous steps on a different website and try to dissect different HTTP packets to gain confidence in analyzing the packet structure details.

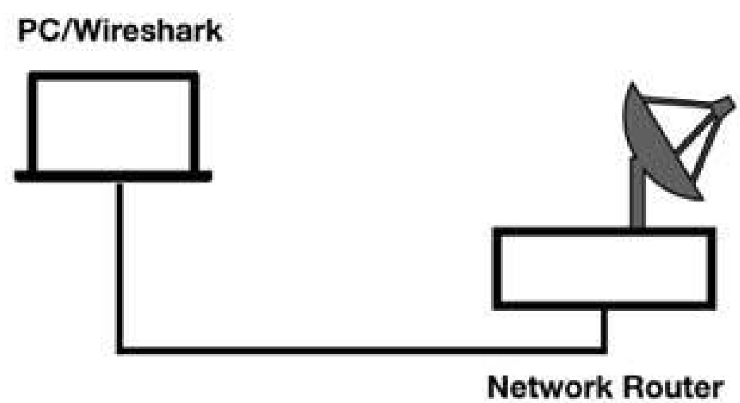
Lab 67. HTTP Filtering

Lab Objective:
Learn the HTTP filter syntax.

Lab Purpose:
Learn how to create and use HTTP display and capture filters.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

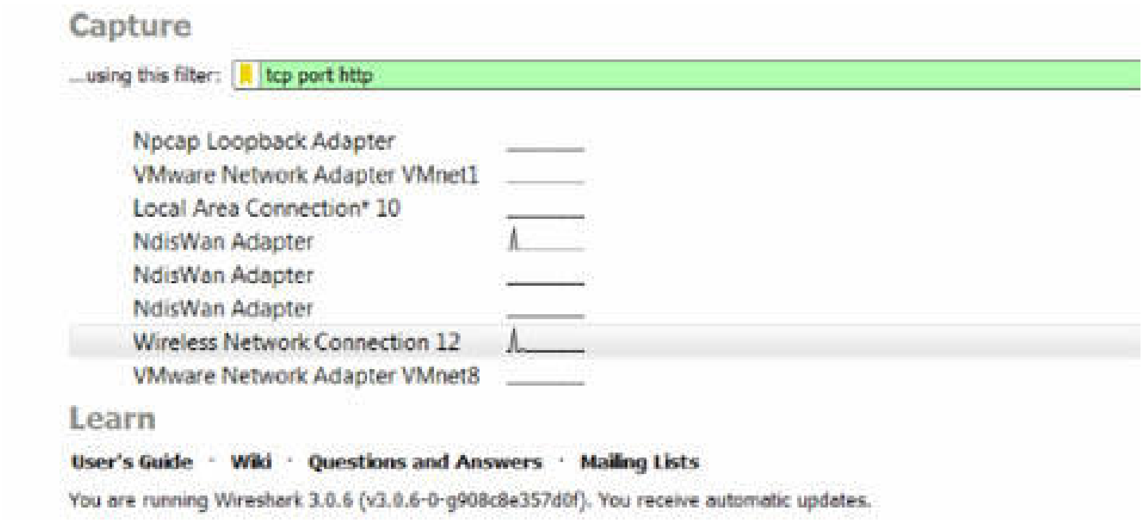


Lab Walkthrough:

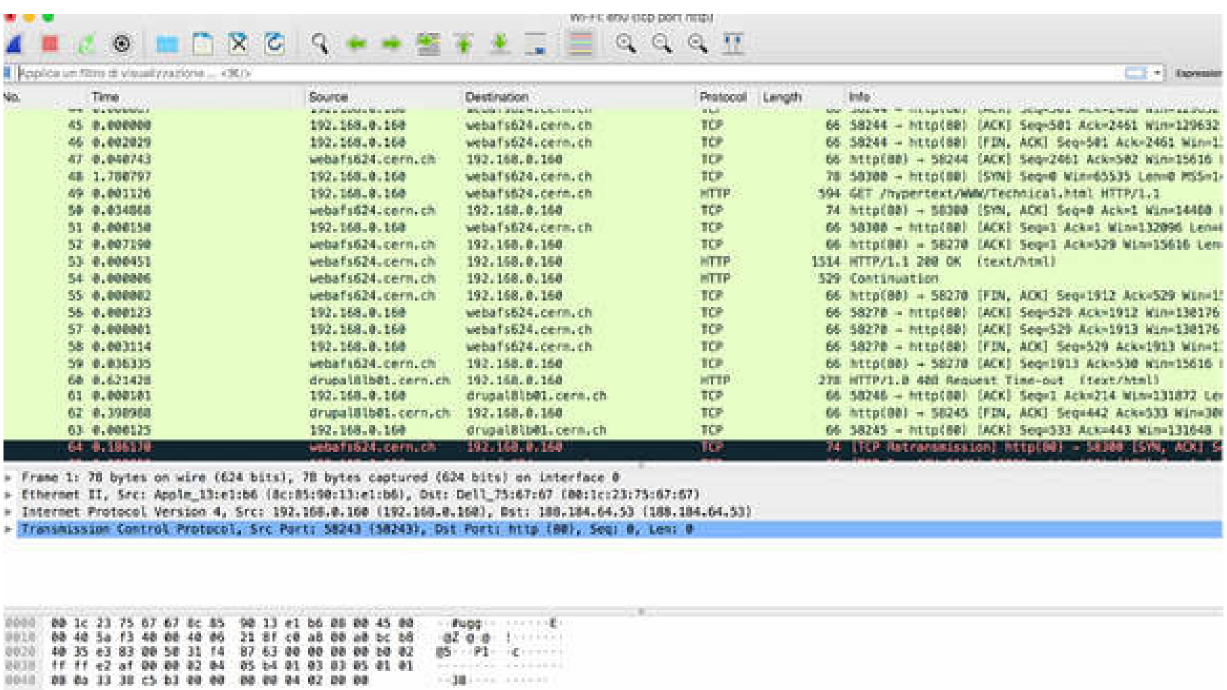
Task 1:
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic

column.

To capture the HTTP traffic, in the Capture Filter, enter tcp port http , as shown in the figure below.



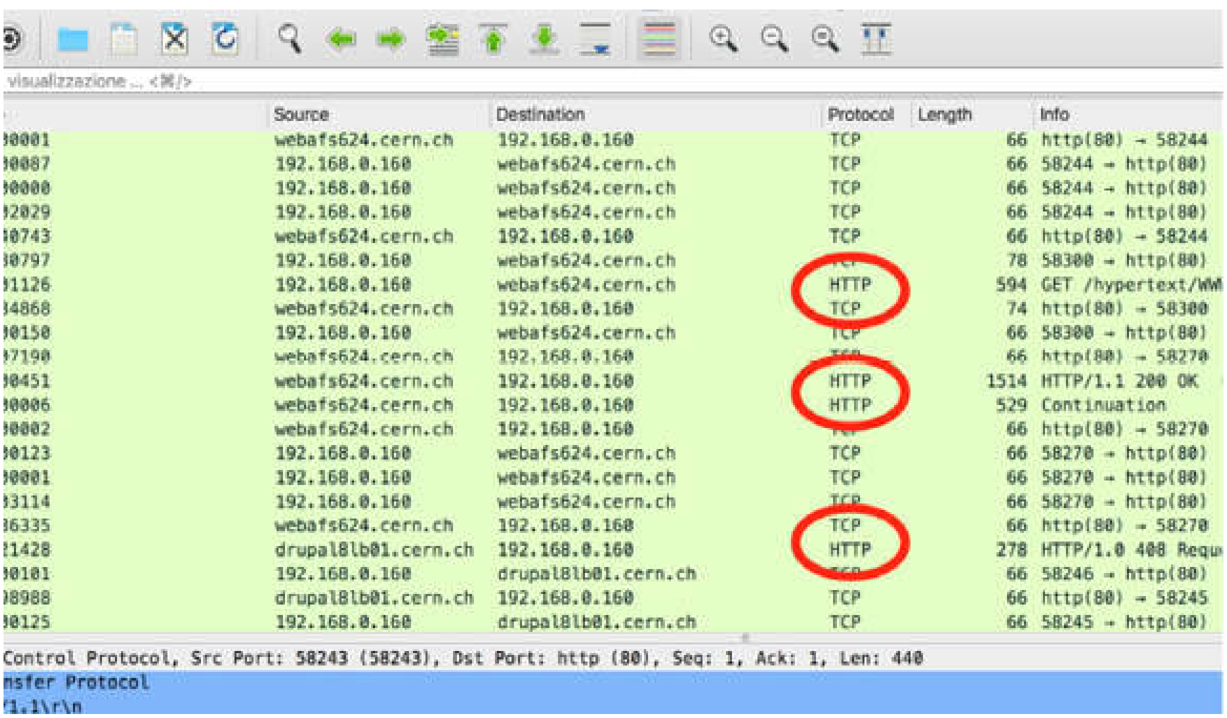
In a web browser, go to http://info.cern.ch/ , and inspect some of the links on the main page. Stop the capture in Wireshark and save the file, naming it cernFilterHttp.pcapng. The results are similar to the figure shown below.



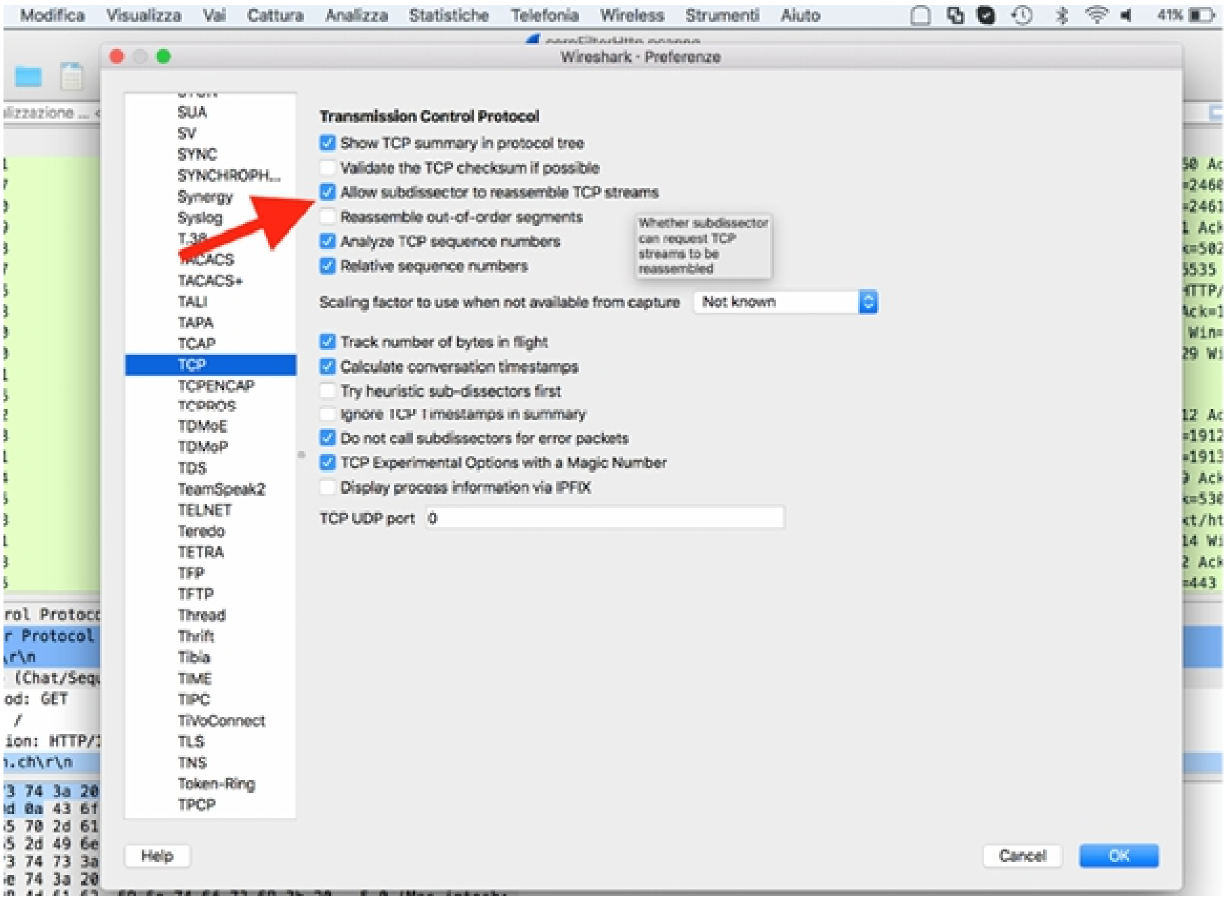
To capture the HTTPS traffic, in the Capture Filter, enter tcp port https .

The steps described in this task work only when the standard port is used for the HTTP or HTTPS stream. If the standard port is not used, in the Capture Filter, enter tcp.port == X where X is the port used.

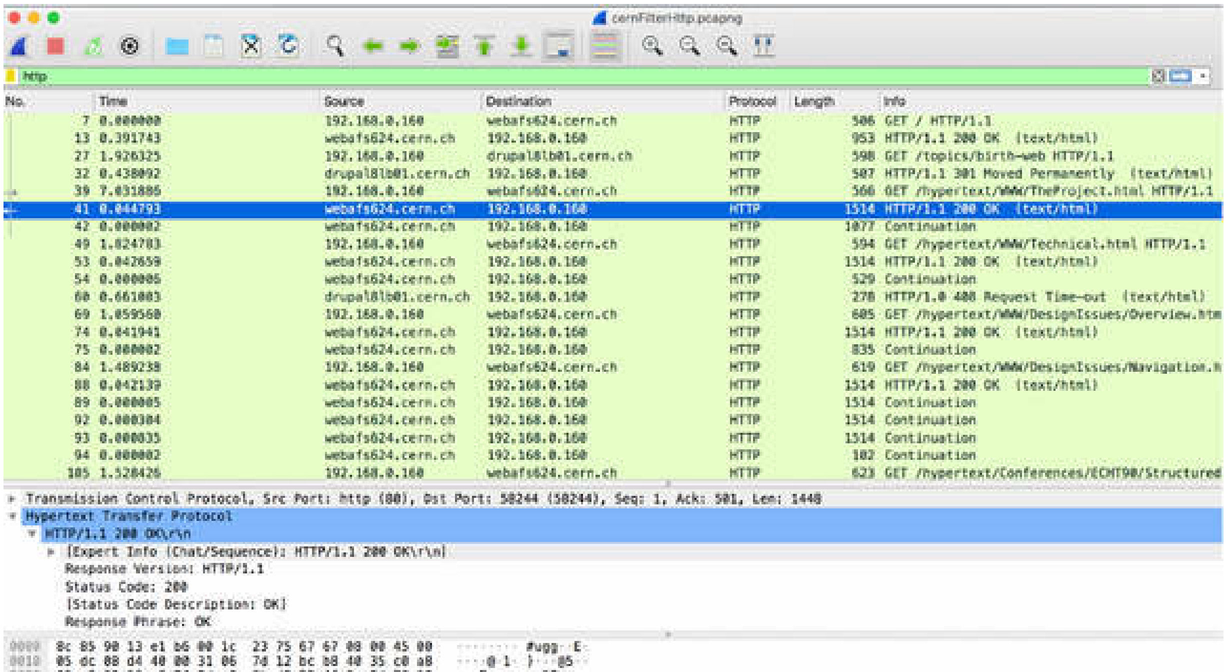
Task 2:
You should not use the http display filter to analyze the traffic for a web browsing session. This is because, from the entire list of packets reported, only a small subset satisfies the http filter, as shown in the figure below. Therefore, to perform correct analysis of a web browsing session, view all packets involved in the communication because all of them can impact the behavior.



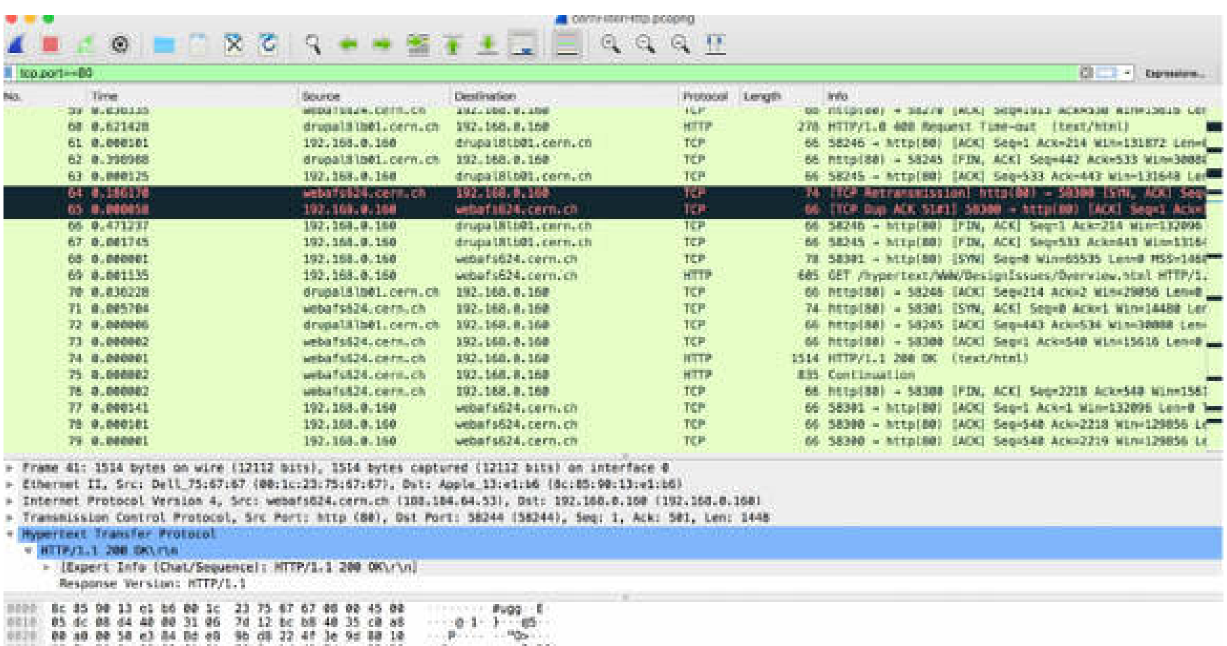
Moreover, the effectiveness of the http display filter depends upon whether TCP reassembly is enabled or disabled in TCP preferences by using the “Allow subdissector to reassemble TCP streams” check box, shown in the figure below.



For example, if you apply the http display filter to the capture file saved earlier, you will observe that you have a partial view of the web browsing session (either with TCP Reassembly ON or TCP Reassembly OFF). In addition, you cannot see the TCP handshake and the FIN, RST, or ACK packets, as shown in the figure below.



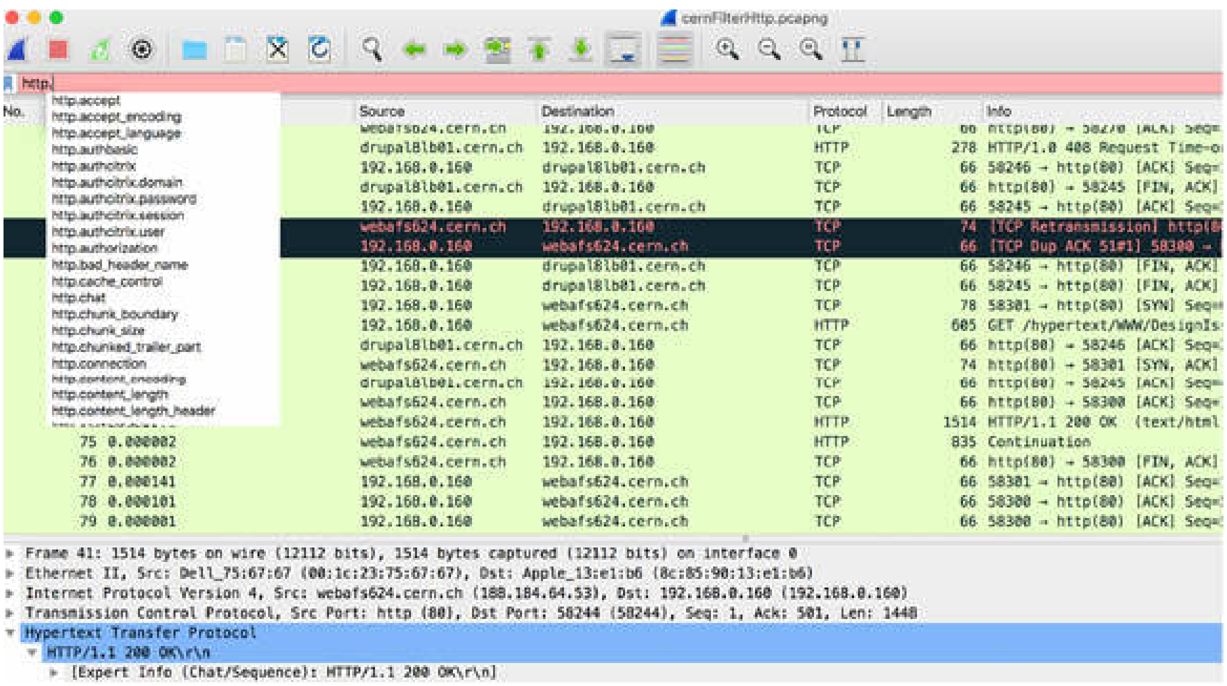
To view all packets in a web browsing session, in the filter toolbar, enter tcp.port==80 .



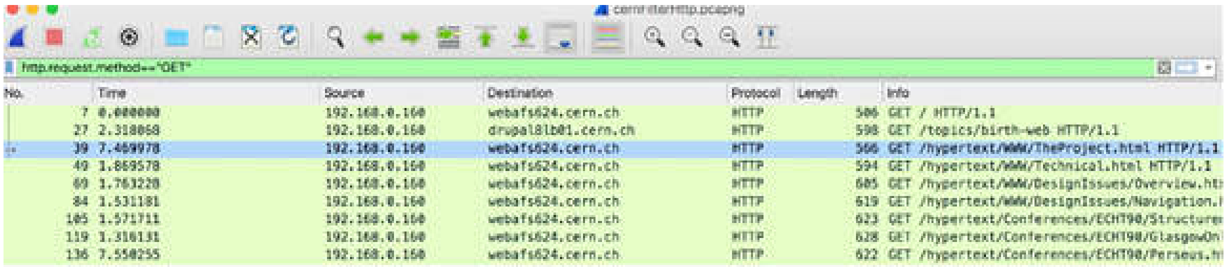
The filter for HTTP or HTTPS must be based on the port in use, such as port 443 for HTTPS (tcp.port==443). Alternatively, you can use the ssl display filter. HTTPS uses Transport Layer Security (TLS) which is based on SSL. If you use the ssl display filter, you will not see the TCP handshake process or ACK packets. Therefore, it is best to use a port number based display filter to see all packets in an SSL conversation.

Task 3:

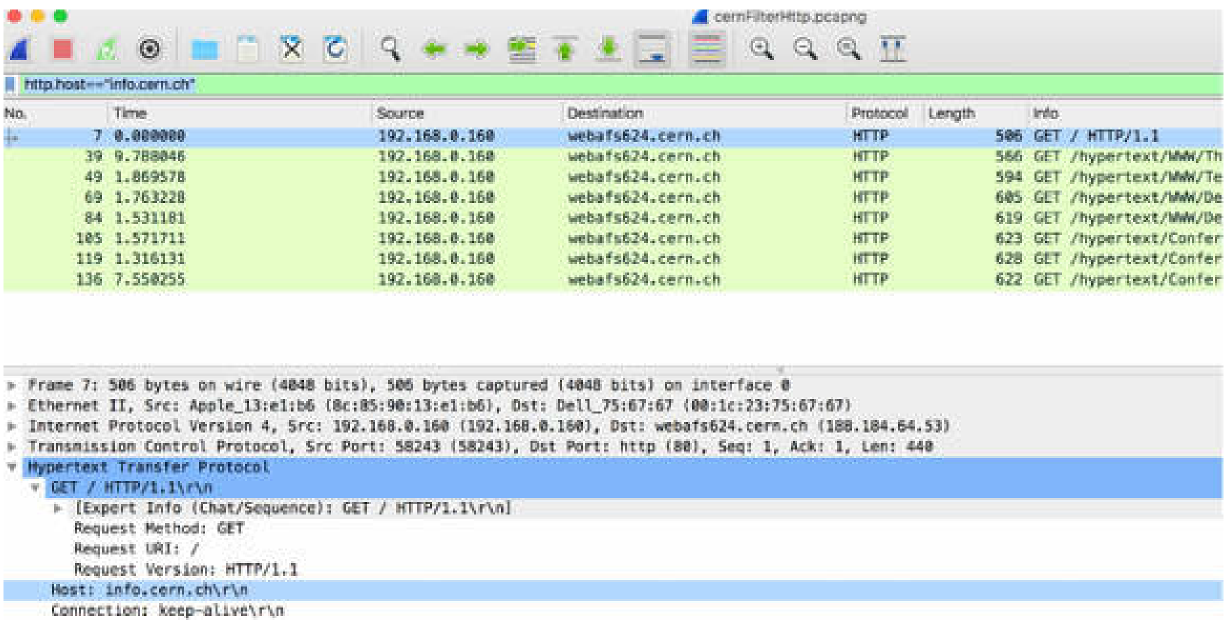
To view a list of all available HTTP filters, you can use the auto-complete feature. For example, if you just type http. in the filter toolbar, the auto-complete feature displays a drop-down menu with all available HTTP filters, as shown in the figure below.



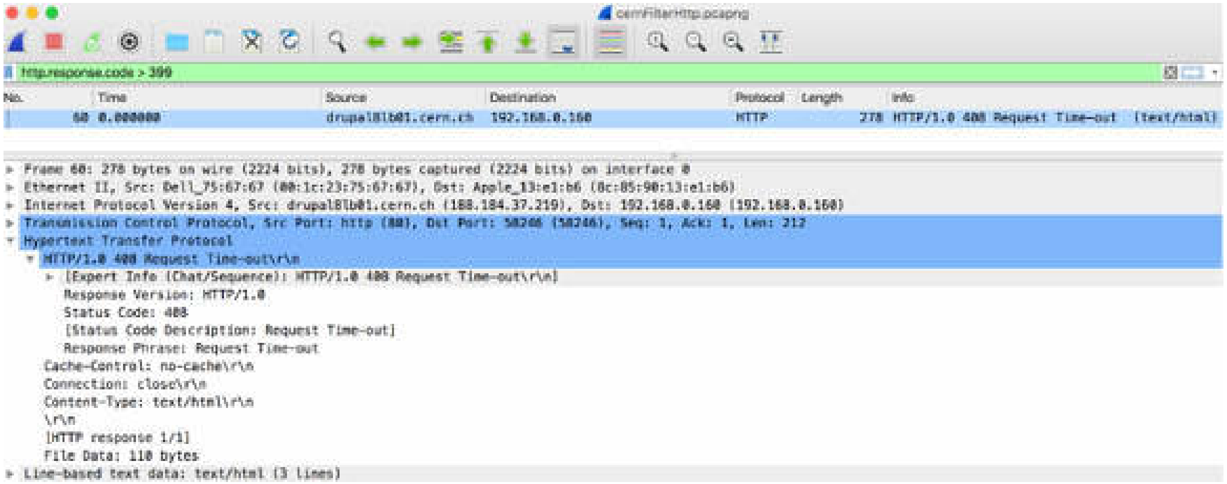
To display only HTTP GET requests, in the filter toolbar, enter http.request.method==”GET” .



To display only a specific target host, in the filter toolbar, enter http.host==”info.cern.ch” .



To display only client/server errors, in the filter toolbar, enter http.response.code > 399 .



The display filters described in this task are for HTTP. You can use the similar display filters for HTTPS.

Notes:
To gain the necessary confidence in searching the right fields in a capture trace, repeat the previous tasks by using the different display and capture filters.

Analyze the HTTPS communication to understand the differences that a secure browsing session presents.

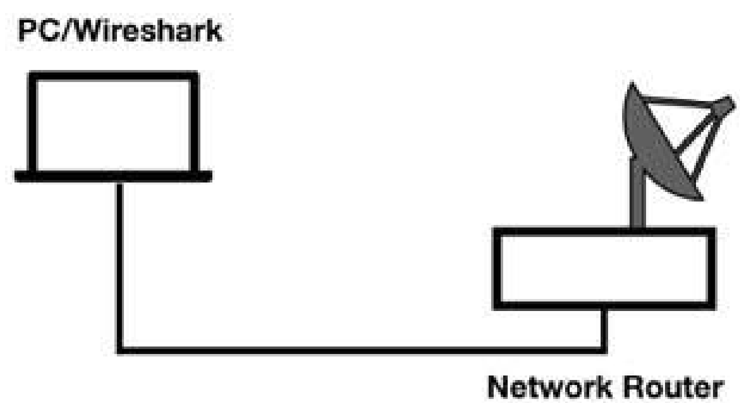
Lab 68. HTTP Statistics

Lab Objective:
Learn the HTTP statistics graph feature.

Lab Purpose:
Learn how to create and use HTTP statistics graphs.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

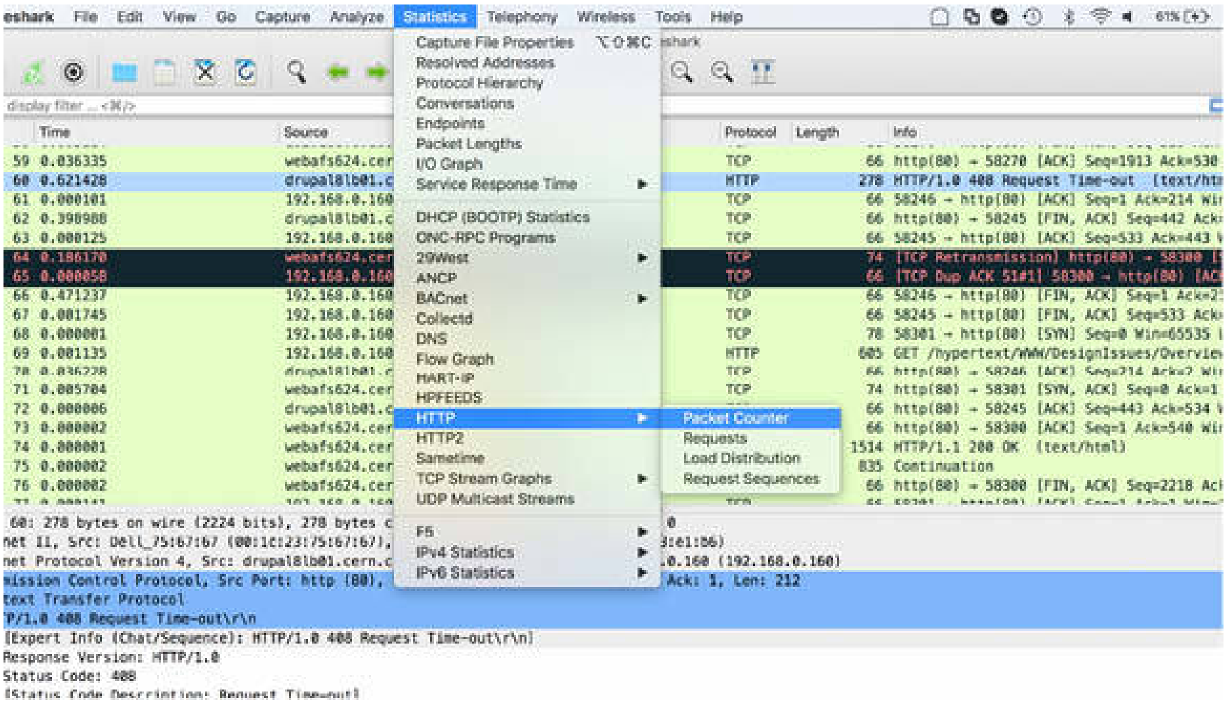
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



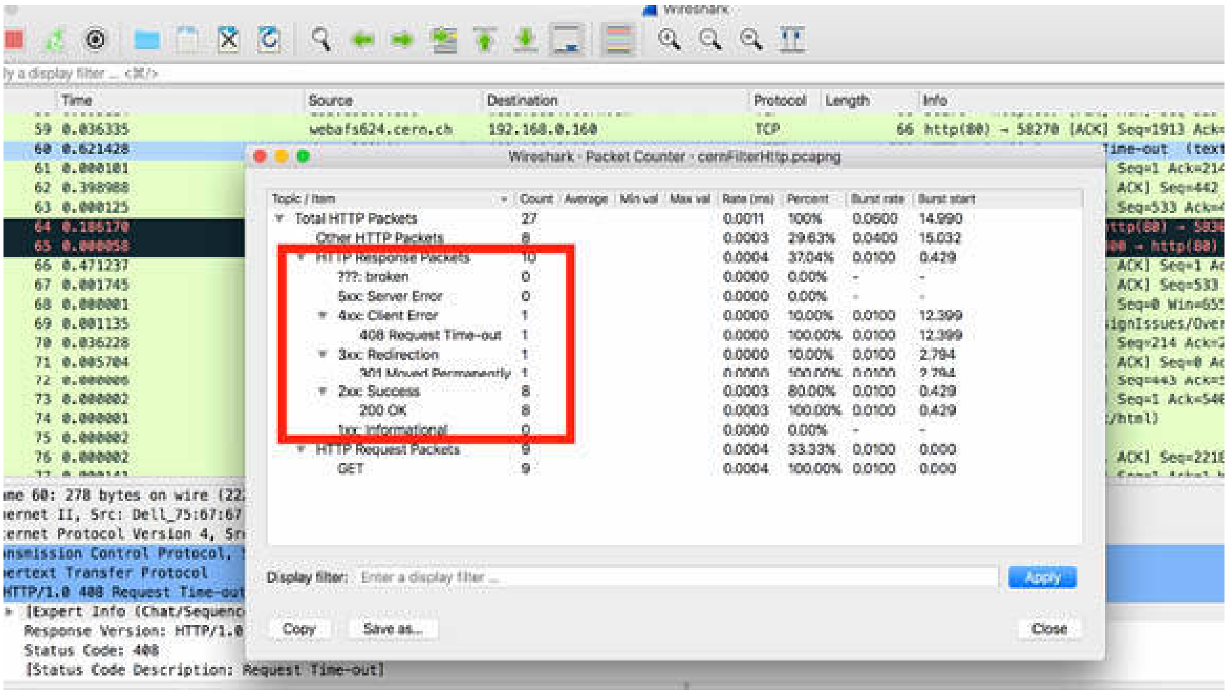
Lab Walkthrough:

Task 1:
In Wireshark, open a trace file saved in the previous labs.

On the main menu, select Statistics > HTTP > Packet Counter.

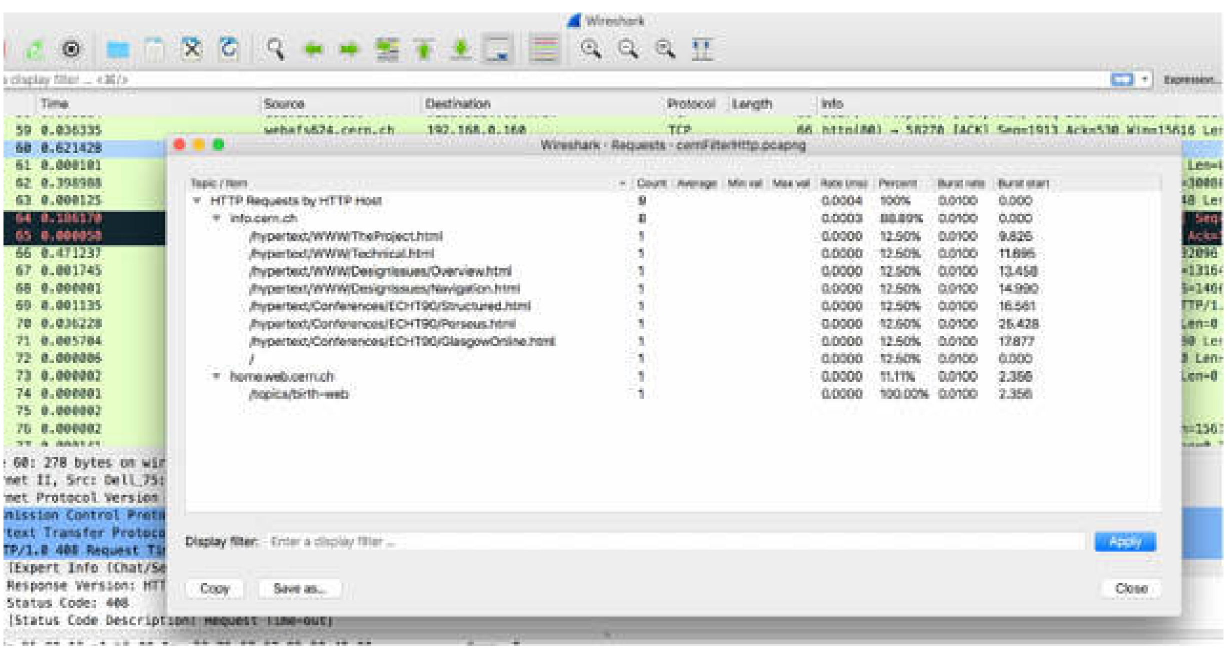


The Packet Counter dialog box is displayed listing the Status Code responses. When analyzing HTTP communication, this information is useful because it makes it easy to spot 4xx Client Error or 5xx Server Error responses, as shown in the figure below.

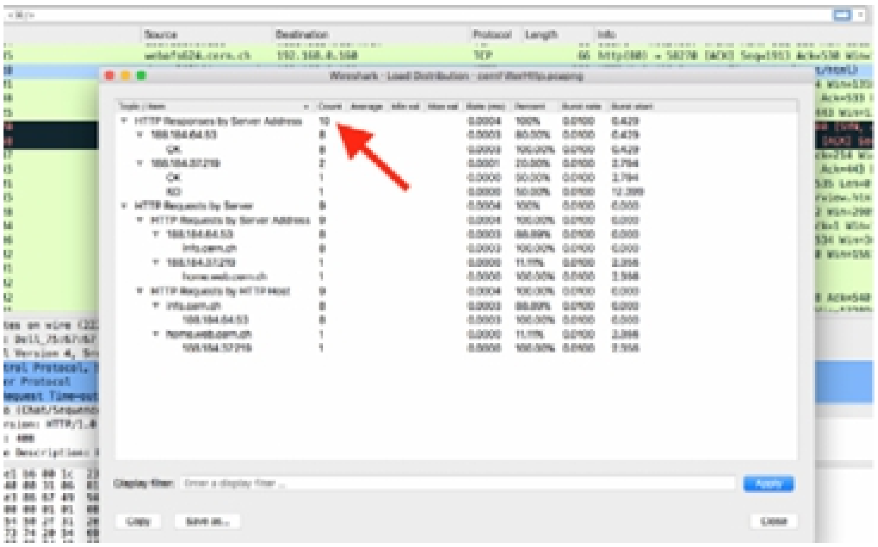


Task 2:

On the main menu, select Statistics > HTTP > Requests. The Requests dialog box is displayed listing each item requested during the web browser navigation.

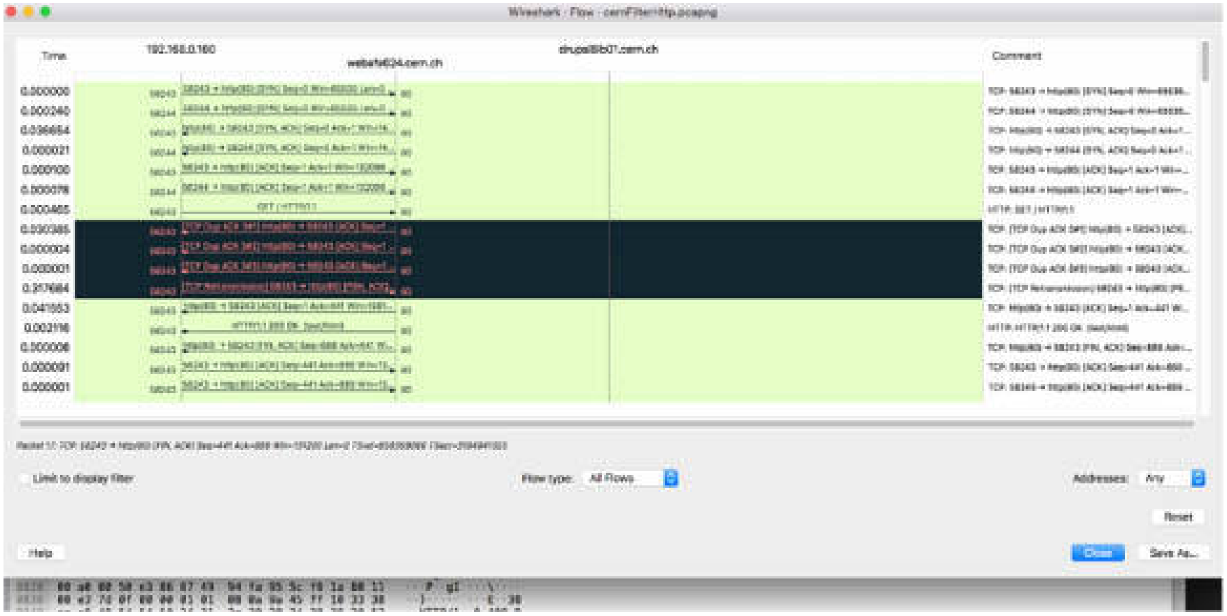


On the main menu, select Statistics > HTTP > Load Distribution. The Load Distribution dialog box is displayed listing HTTP requests and responses by the server. As shown in the figure above, the HTTP Requests by HTTP Host field lists the hosts contacted and the number of request packets sent to each one. These statistics are useful in determining the website redirections and dependencies. Based on the statistics shown in the figure below, you can conclude that a simple browsing session to http://info.cern.ch/ creates HTTP sessions with 10 different servers.

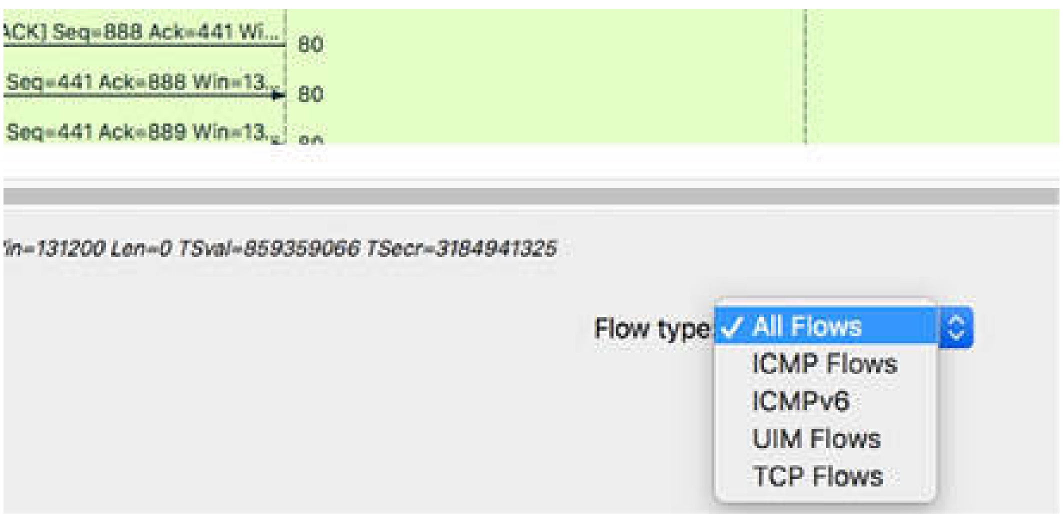


Task 3:

On the main menu, select Statistics > Flow Graph. The Flow dialog box is displayed providing a visual representation of the communication that occurs during an HTTP session. This information is very important when troubleshooting slow web browsing sessions. Each target host is listed in a column and every packet is listed in a row, as shown in the figure below.

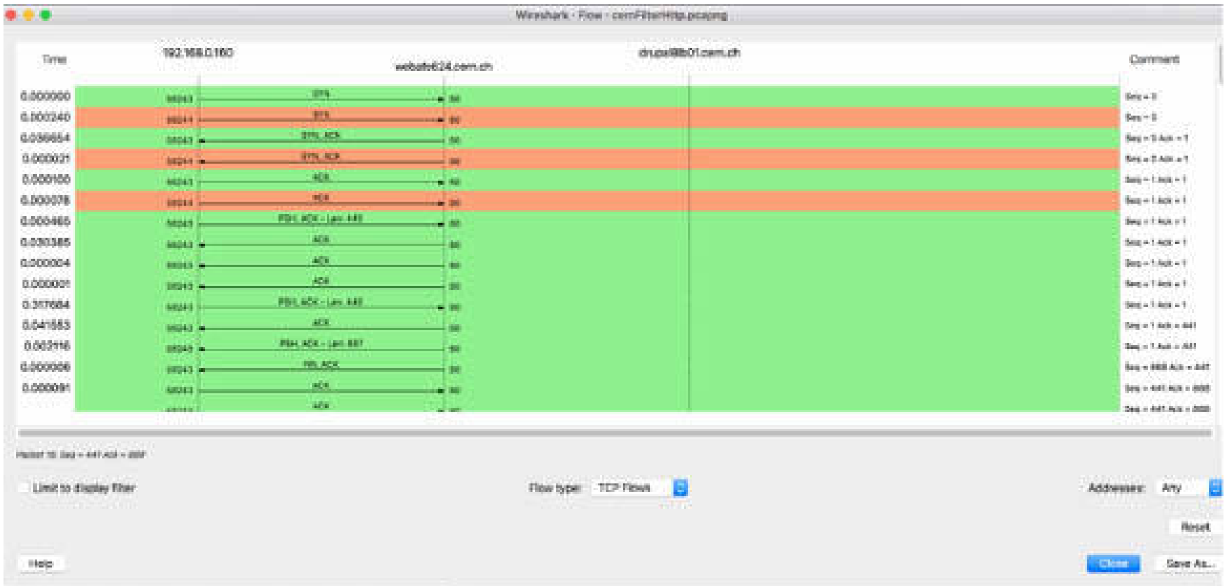


To customize the Flow graph, click “Flow Type” and select a type from the drop-down list, as shown in the figure below.



The general flow view includes application-layer information, such as requests and replies. The TCP Flow graph shows the TCP header values, such as the sequence number and acknowledgment number values and the TCP flag settings.

When you select “TCP Flows”, the result will be similar to the ones shown in the figure below.



You can also choose a node address type. Choose standard source/destination addresses when many hosts are communicating with each other, and you need to narrow the output. This option shows the IP addresses of devices listed in the graph. Choose network source/destination addresses when you are using network name resolution with Wireshark.

Notes:
To observe different results for different web servers browsed, repeat the previous steps by using the statistics and graph features for different trace files.

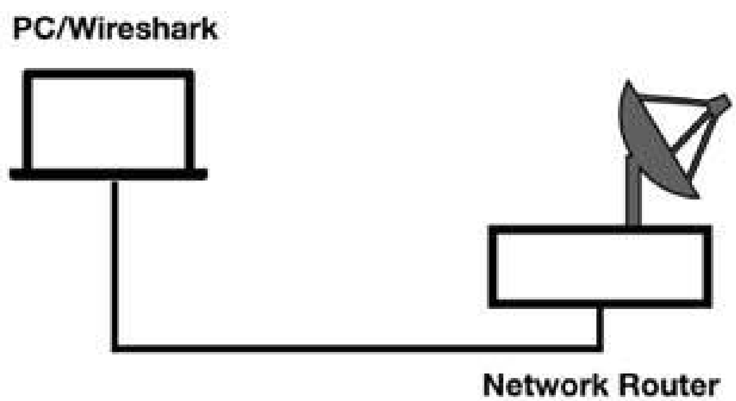
Lab 69. HTTPS Communications

Lab Objective:
Learn to analyze HTTPS communications.

Lab Purpose:
Learn how to analyze HTTPS communications by understanding the main features.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

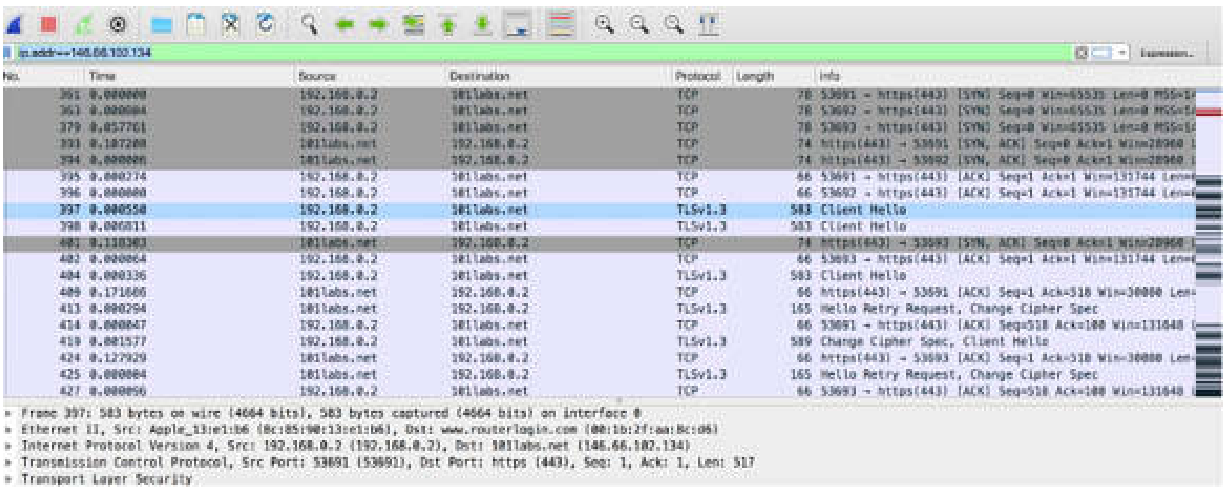


Lab Walkthrough:

Task 1:

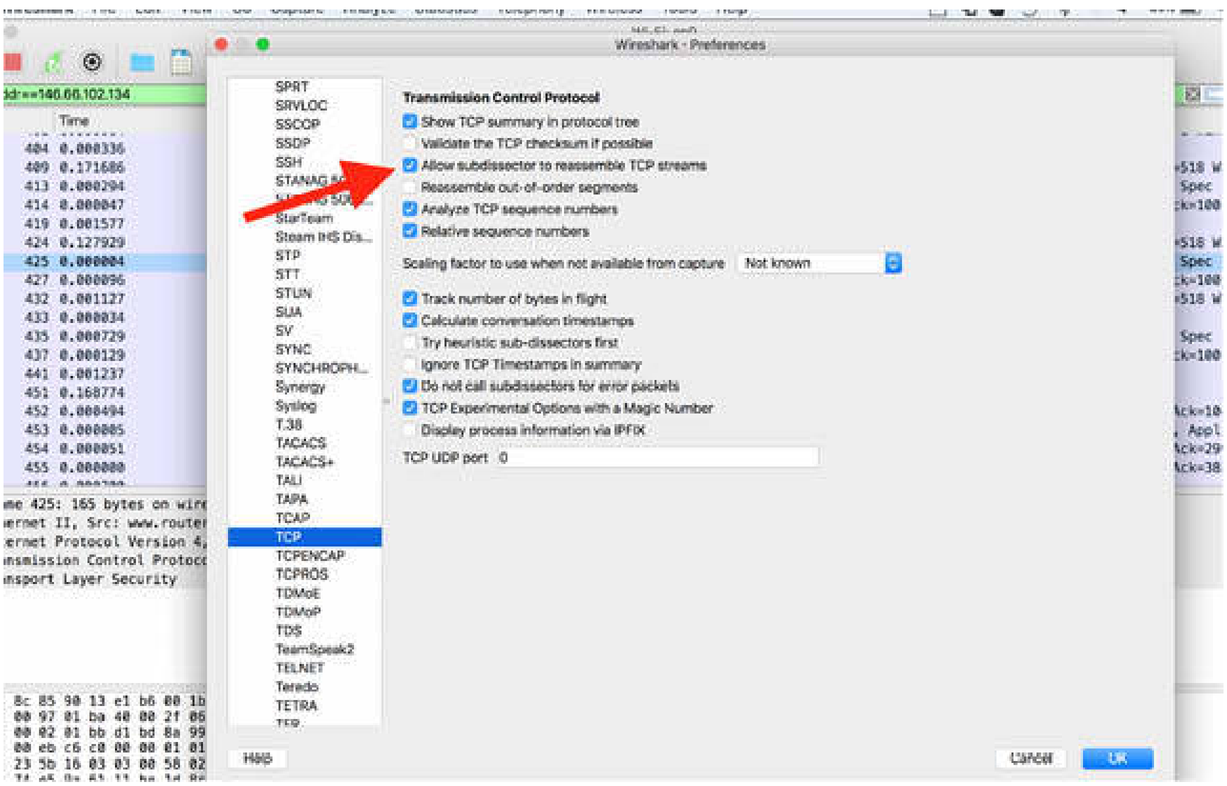
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes. In a web browser, go to www.101labs.net and inspect some of the links on the main page. Stop the capture in Wireshark and save the file, naming it https101labs.pcapng.

In the filter toolbar, enter ip.addr==146.66.102.134 . The results similar to the ones shown in the figure below are displayed.



As shown in the figure above, at the start of a secure HTTP conversation, a standard TCP handshake is executed (packets #361 to #396) which is followed by a secure handshake process (packets #397 to #451). The HTTP over Transport Layer Security (TLS) is based on SSL version 3.

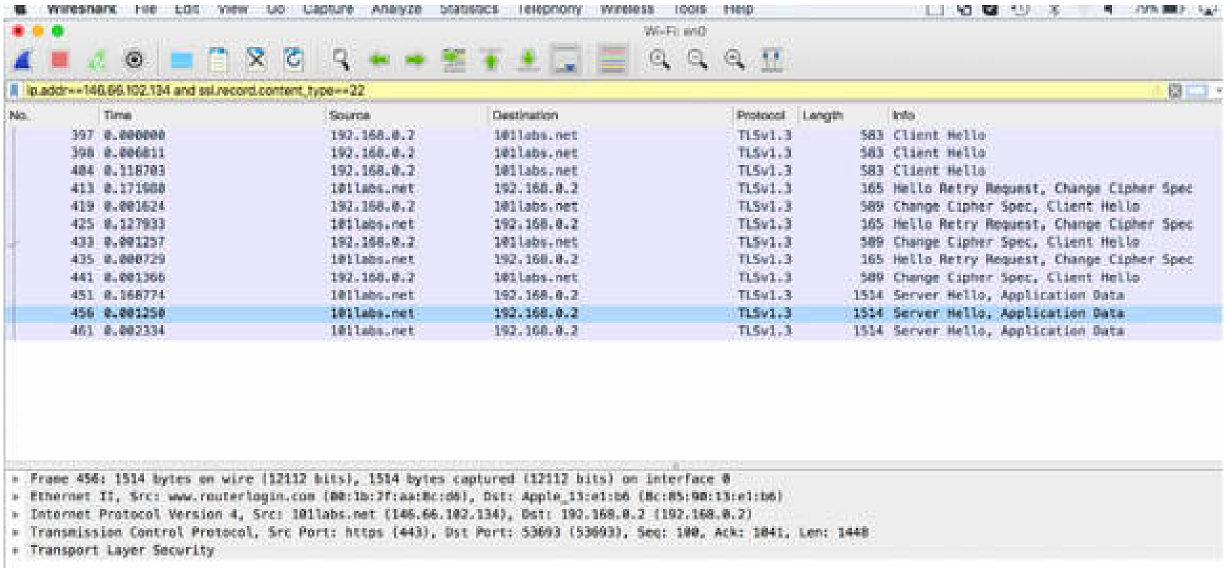
For better identification of the TLS packets, when analyzing HTTPS traffic, enable the “Allow subdissector to reassemble TCP streams” check box for TCP in the Preferences dialog box.



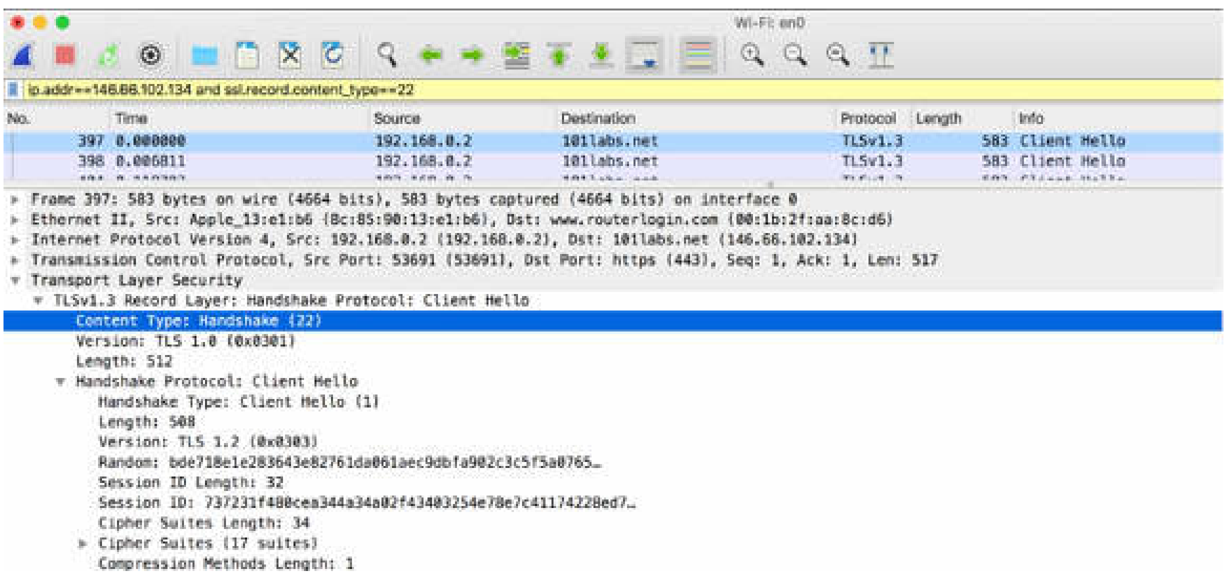
Based on the results shown in the Packet List pane, it is clear that the communication that you are analyzing uses the standard HTTPS port number 443. There are also other port options. To use other ports, you can add them in the Preferences dialog box.

Task 2:

In the filter toolbar, enter ssl.record.content_type==22 to display only the TLS packets related to the initial handshake (the TLS handshake consists of a series of packets with a content type value of 22), as shown in the figure below.



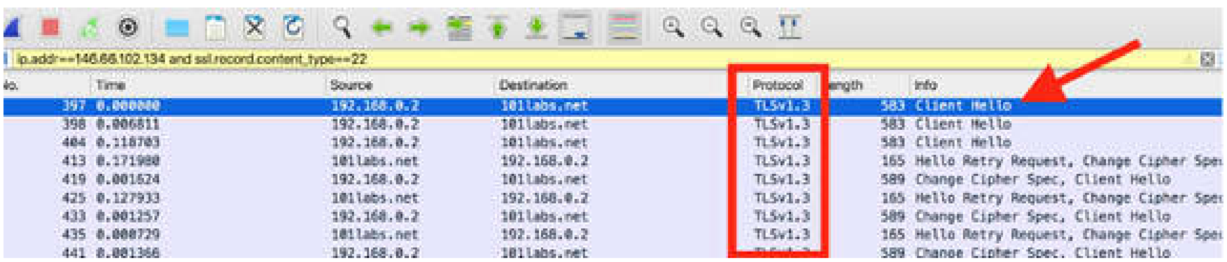
You can use the Packet Details pane, shown in the figure below, to verify that only the TLS packets related to the initial handshake are displayed.



The TLS handshake enables peers to agree on security parameters for the exchange of data and to authenticate themselves. In addition, errors during the handshake process are relayed in the TLS handshake packets. The handshake process usually includes the following:

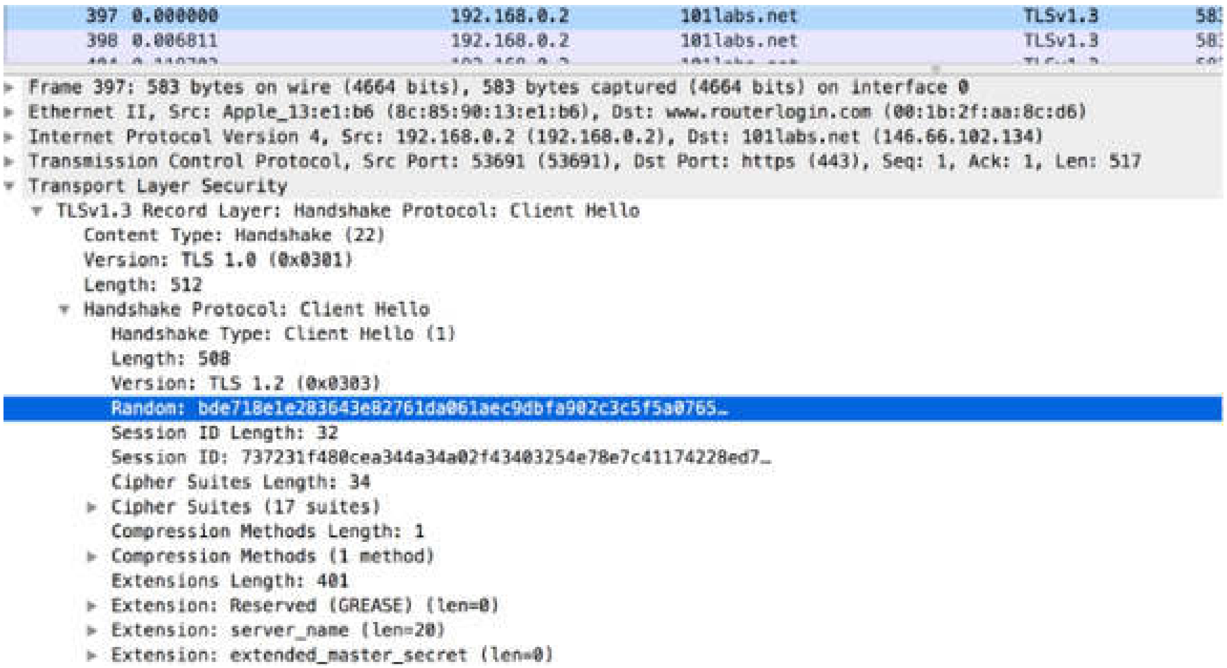
- Session identifier: Identifies a new or resumed session
- Peer certificate: X509 certificate of the peer
- Compression method: Compression method for data before encryption
- Cipher spec: Defines the data encryption algorithm
- Master secret: 48-byte secret shared between the client and the server

The first packet in our handshake process is "Client Hello", as indicated in the Info field in the figure below. The client denotes that it is using TLS version 1.3.

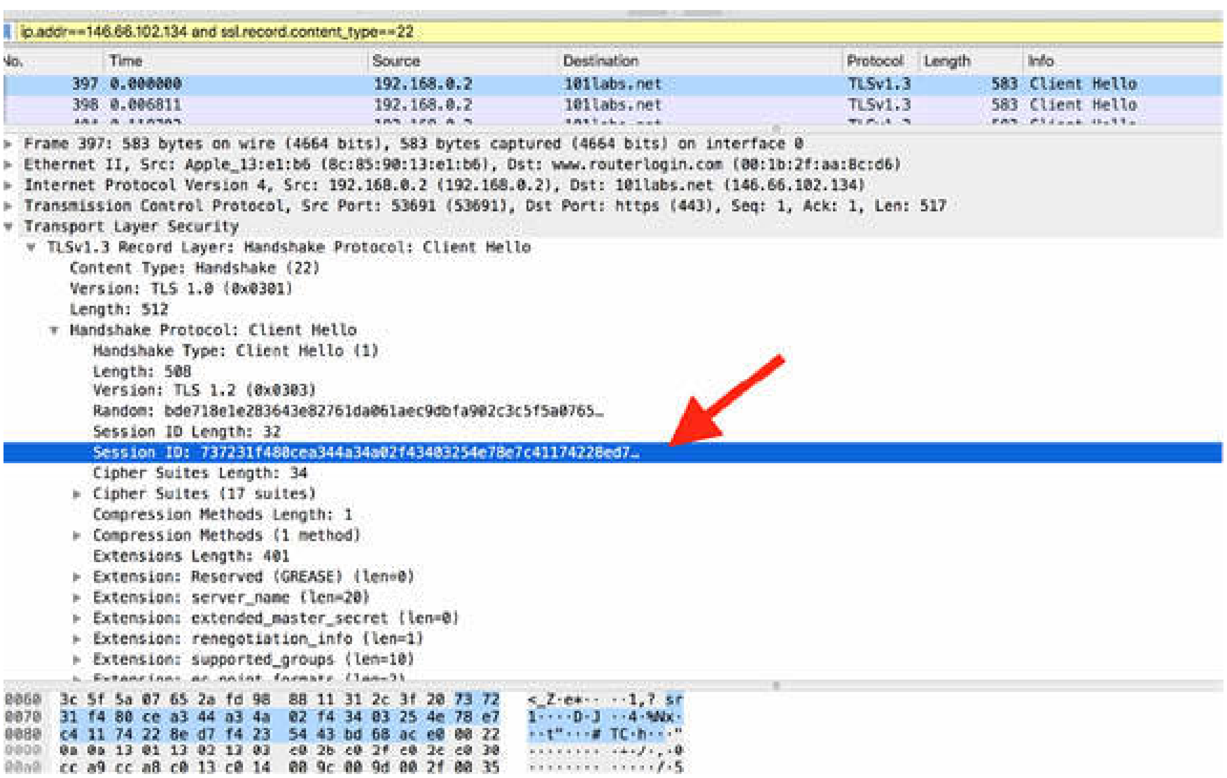


Task 3:
In the Packet List pane, select the first handshake packet. In the Packet Details pane, open the TLS tree view to identify the TLS fields.

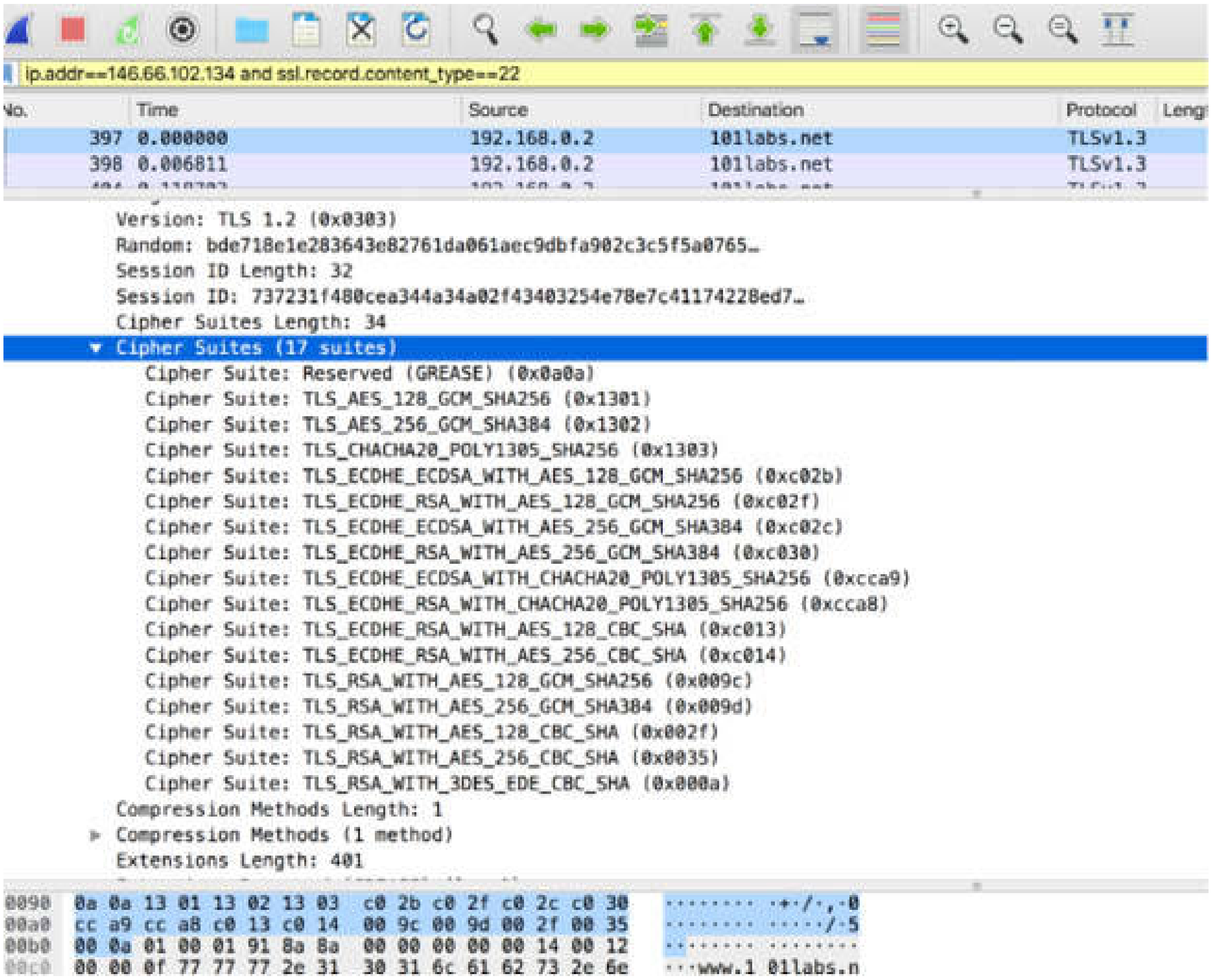
As shown in the figure below, the Random field contains the Universal Coordinated Time (UTC) of the client in the Unix format.



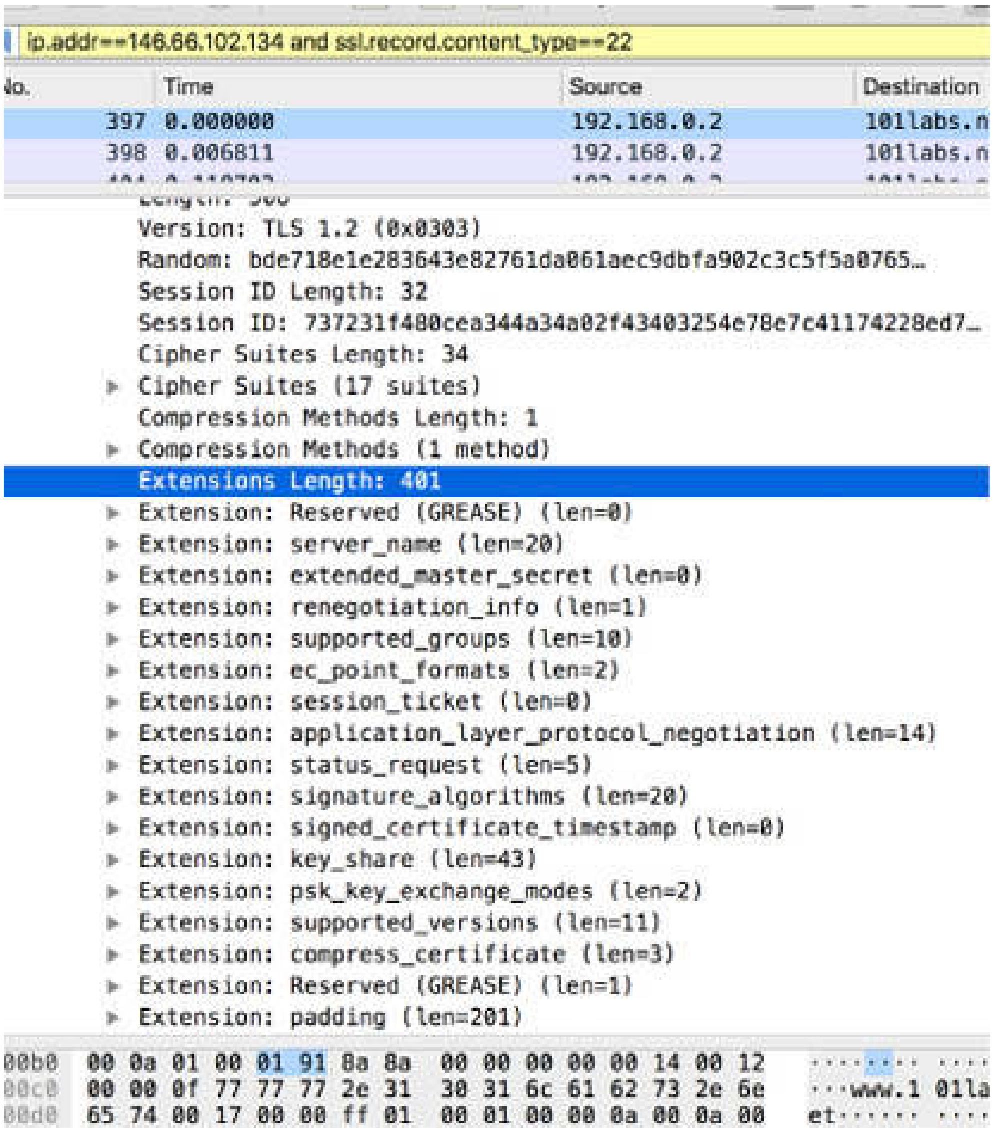
The Session ID field contains a non-zero value, indicating that this is a resumed session. The Session ID field is set to 0 for a new session.



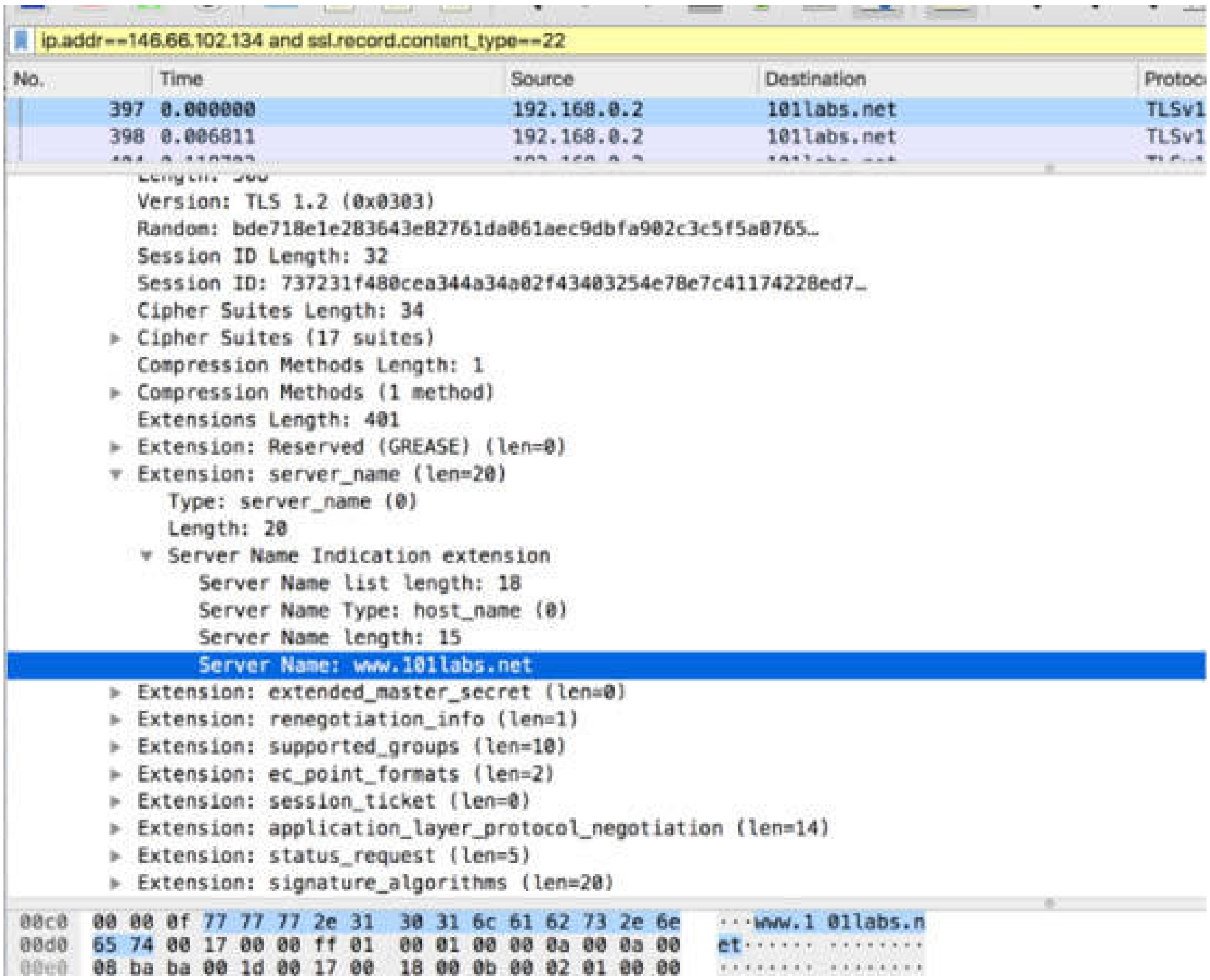
The client provides the list of cipher suites (inside the “Cipher Suites” field) supported by the browser. In this case, the client supports 17 cipher suites and lists them all in the “Client Hello” packet. In the end, the server decides which cipher suite to use, but the cipher listed at the top is the client’s preference.



At the end of the TLS packet, after the “Compression Methods” field, there are a series of extensions. These extensions add functionality to TLS.

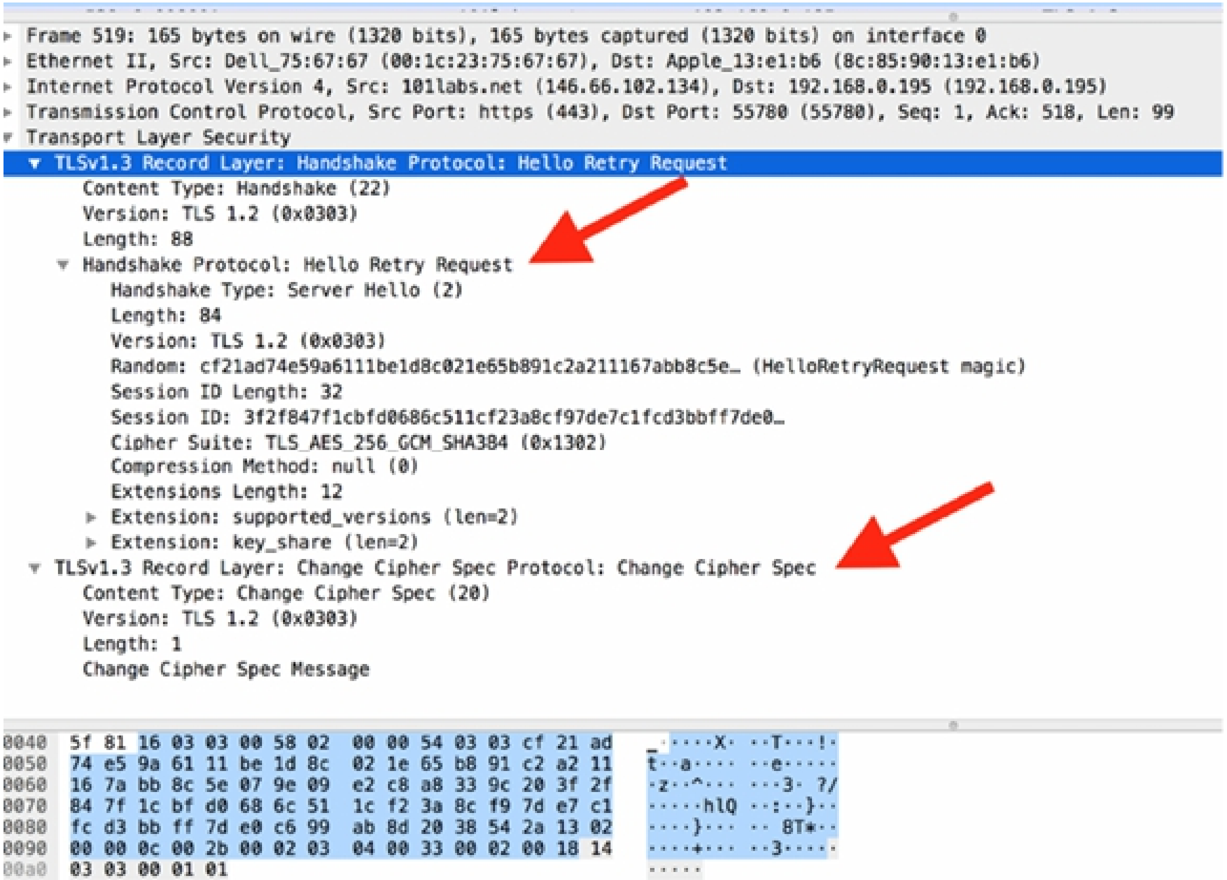


The server name extension provides the server name, which, in this case, is www.101labs.net . The server name extension enables the client to create a secure connection to a virtual server that may be hosted on a machine that supports numerous servers at a single IP address.



Task 4:

In the first TLS packet from the server (packet #413), the server responds with a packet containing two functions: “Hello Retry request” and “Change Cipher Spec”.



In the Random section, the server provides 28 random bytes and a 32-byte Session ID value to allow the client to reconnect later. This set of random bytes is sent again later in the handshake, but at that time, it is encrypted with the client’s public key. These random bytes are used for key generation.

Out of the 17 cipher suites offered, the server selected the cipher suite TLS_AES_256_GCM_SHA384 (0x1302).

Notes:
To gain more confidence in doing the HTTPS communications analysis, repeat the previous steps, and connect to a different server. Start a browsing session and analyze the saved packets in Wireshark.

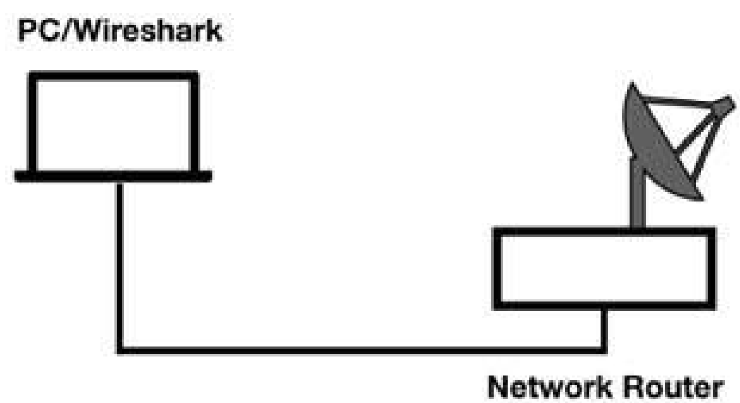
Lab 70. File Transfer Protocol

Lab Objective:
Learn how the File Transfer Protocol (FTP) works and why is it used.

Lab Purpose:
Understand the main purpose of FTP and its features.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
FTP is used to transfer files over TCP. In a typical FTP communication, a command channel is established to port 21 on the FTP server. To transfer

data (such as directory contents or files), a secondary data channel is established by using dynamic port numbers. The specification defines that port 20 should be used for the data channel, but in reality, you will notice dynamic port numbers being used for this channel.

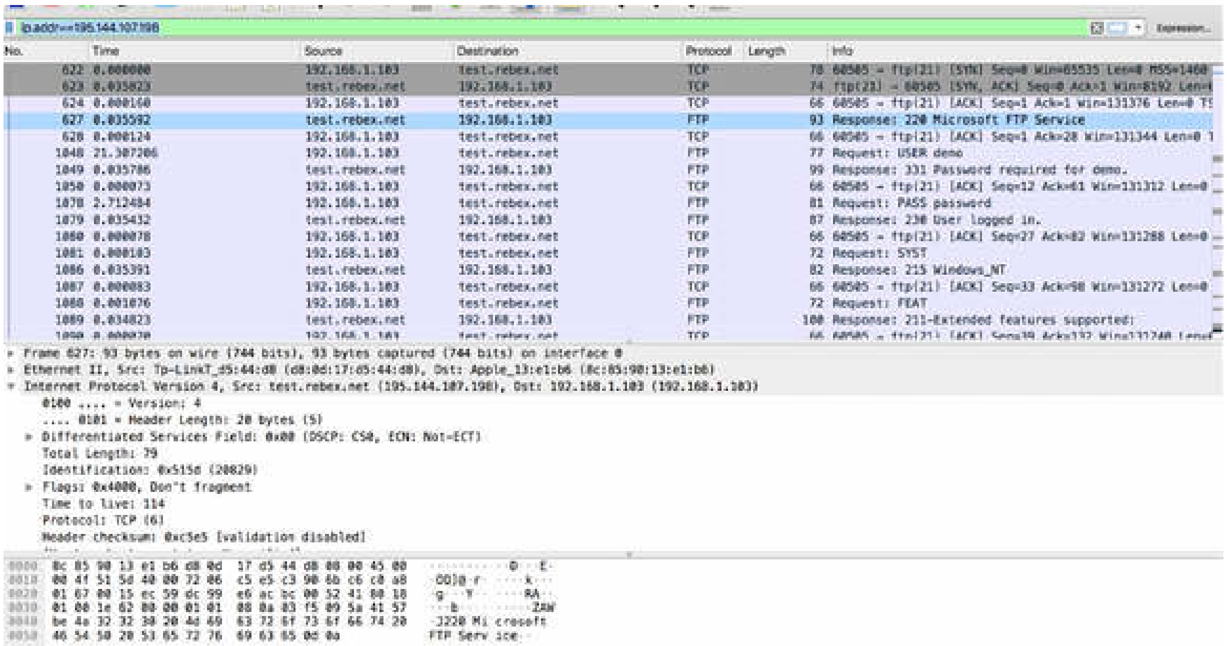
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column.

Open a terminal window, and run the command ftp test.rebex.net . Type demo for the user name and password for the password. At the end of the process, you are logged into the demo FTP service.

Type the command ls to view all the files and folders present on the remote FTP server and navigate through some of the folders. On a Windows PC, you may see error messages from the server. This is because of the way the Windows operating system uses FTP. In such a case, use an FTP client such as Filezilla to browse the FTP server.

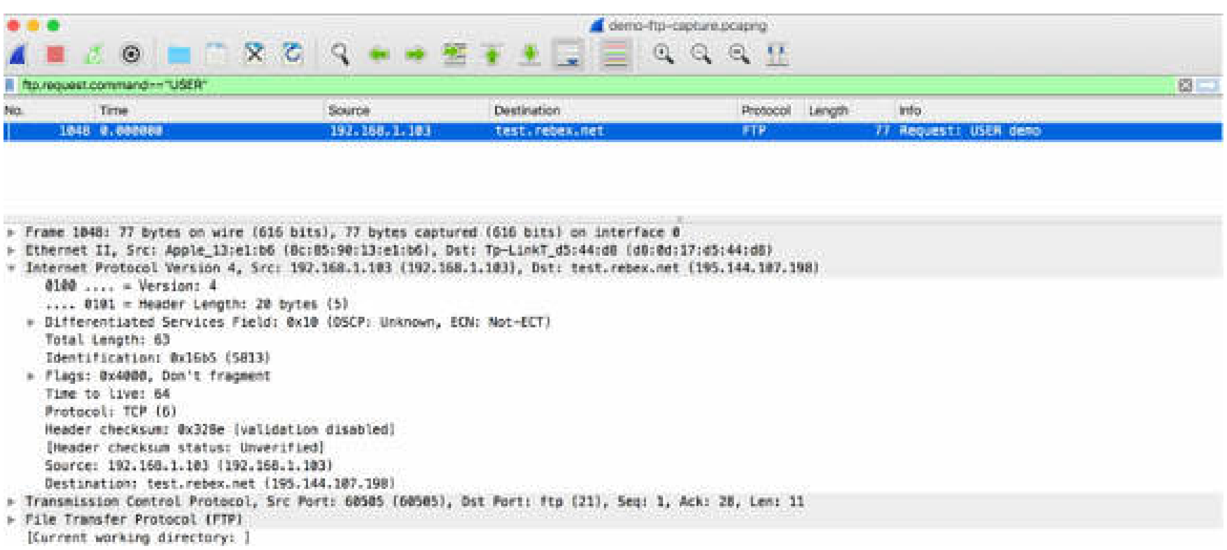
Stop the capture in Wireshark and save the file, naming it as demo-ftp-capture.pcapng.

To display only the communication with the FTP server (IP address 195.144.107.198), in the filter toolbar, enter ip.addr==195.144.107.198. The results are displayed in the Packet List pane, as shown in the figure below.



In the figure above, as shown in packets #622, #623, and #624, an FTP connection begins with a TCP handshake followed by the client waiting for the banner.

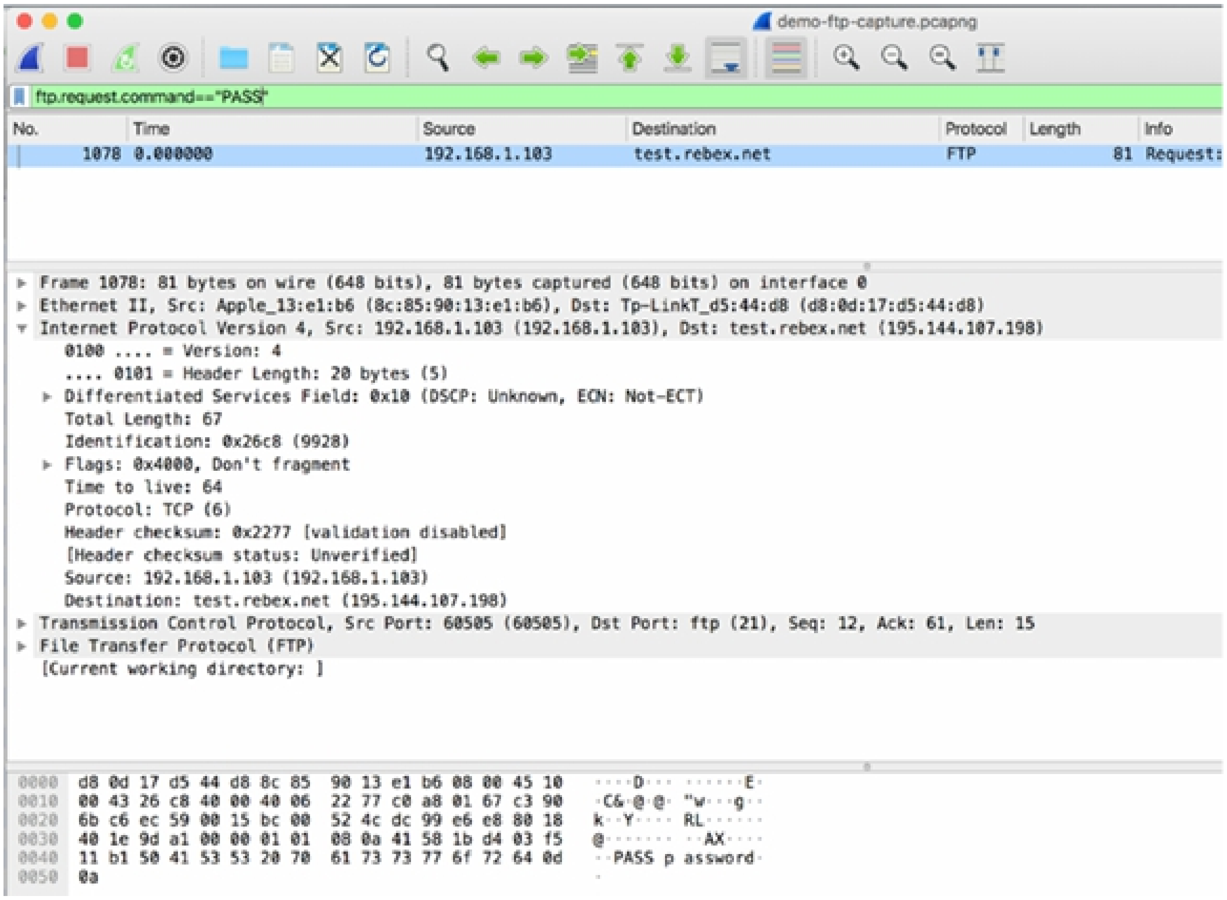
To display all hosts attempting to login to an FTP server, in the filter toolbar, enter ftp.request.command=="USER" . The results are displayed in the Packet List pane, as shown in the figure below. You can verify that only one connection is attempting to log in (a single login packet).



A list of possible FTP commands is shown below. Just be aware that FTP translates the PUT command to STOR and the GET command to RETR.

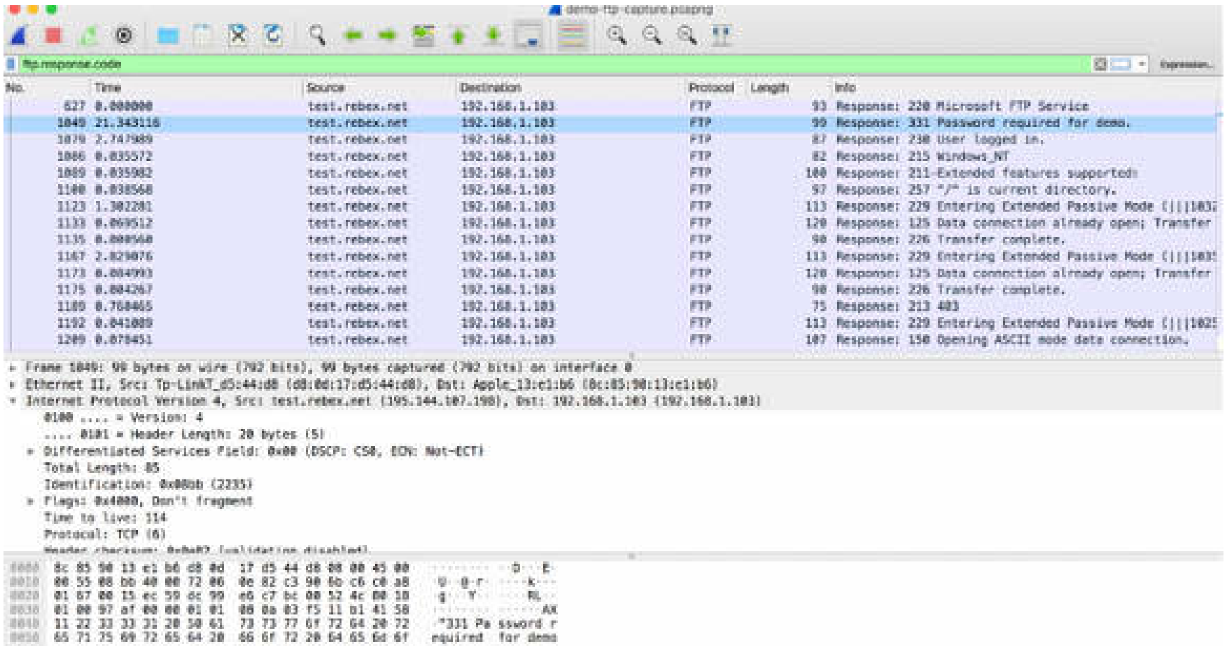
- USER: Identifies the user accessing the FTP server
- PASS: Indicates the user's password
- CWD: Changes working directory
- QUIT: Terminates the connection
- PORT: Sets up a data connection IP address and port number at the client (active mode FTP)
- PASV: Requests the server to listen on a non-default data port for the client to establish a data connection (passive mode FTP)
- TYPE: Indicates the type of data to be transferred
- RETR: Retrieve a file from the FTP server
- STOR: Sends a file to the FTP server
- DELE: Deletes a file
- RMD: Removes a directory
- MKD: Makes a directory
- PWD: Prints (display) working directory contents
- NSLT: Name list—displays directory on the server
- HELP: Shows commands supported by the server

In the filter toolbar, enter ftp.request.command==”PASS” . The results are displayed as shown in the figure below.



Task 2:

To display only the packets containing a response code, in the filter toolbar, enter ftp.response.code . The results are displayed, as shown in the figure below.



The following list provides some of the response codes:

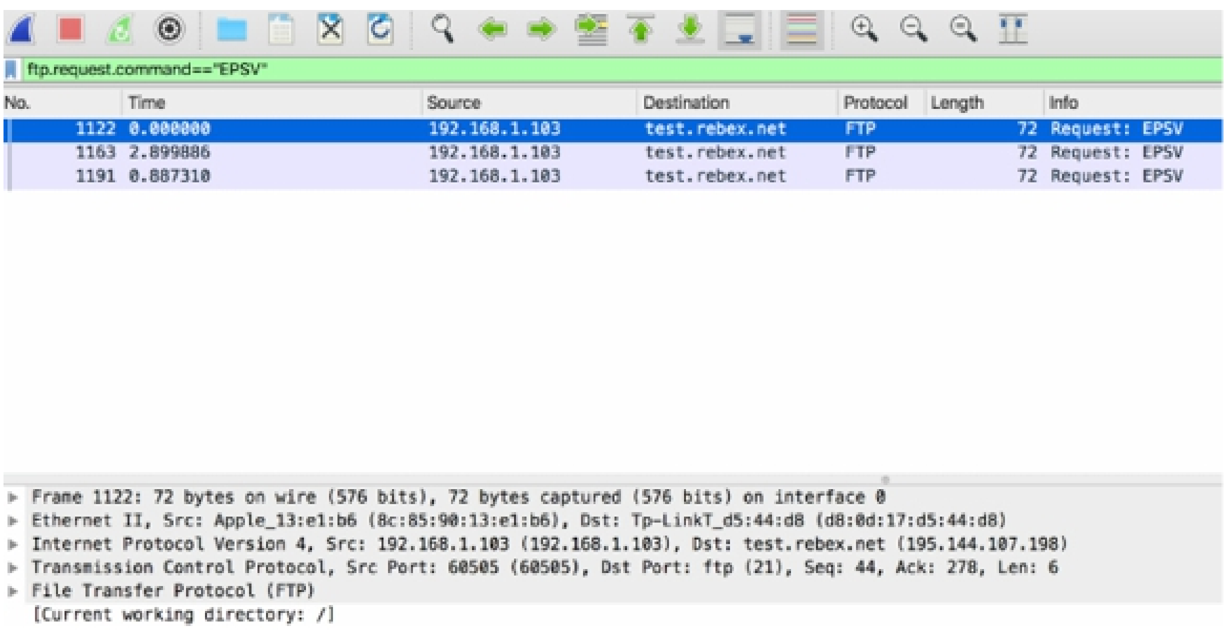
- 220: Service ready for a new user (packet #627).
- 331: User name okay, need a password.
- 230: User logged in, proceed.
- 215: NAME system type, where NAME is an official system name from the list in the Assigned Numbers document.
- 211: System status or system help reply.
- 257: “PATHNAME” created.

The FTP client sends the USER command (all FTP commands are in upper case) followed by a username and then the PASS command followed by the password. If the FTP username is incorrect, the server still responds with the response code 331 Password Required for username. If the password is incorrect, the server responds with the response code 530 Password Not Accepted. After logging in, the user can use commands to examine the directory contents, change directories, and launch a second channel for data transfer.

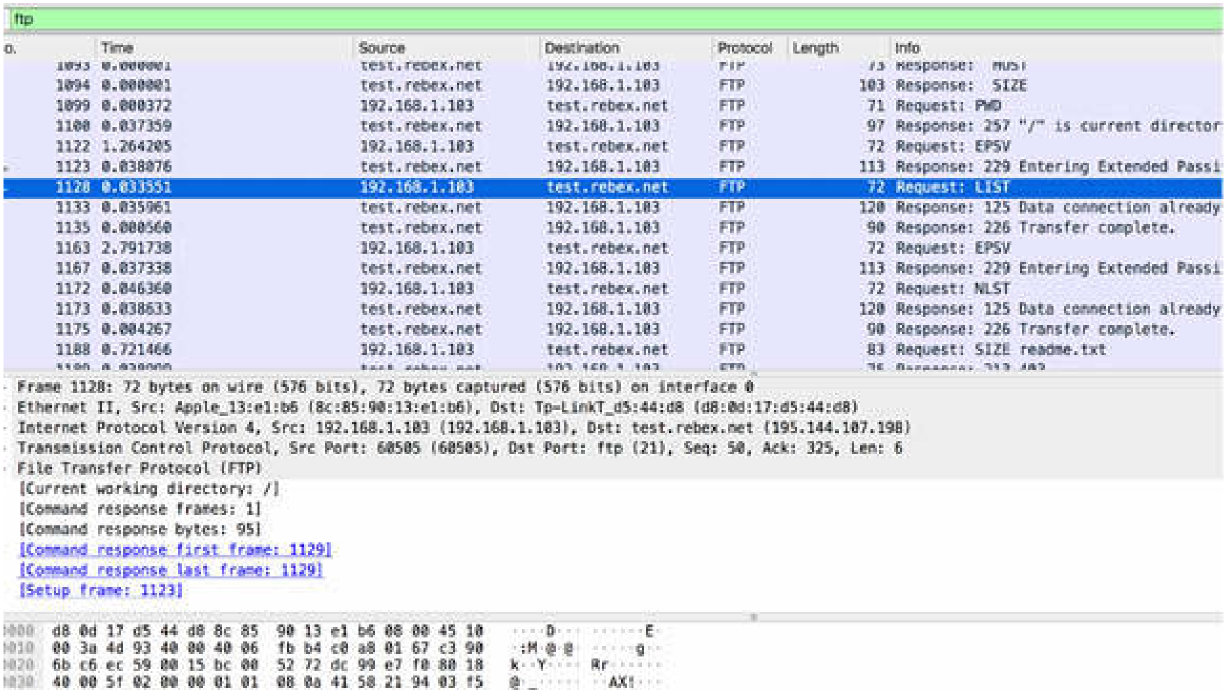
The data transfer takes place over a separate connection from the command connection. Data transferred may be a file or the contents of a directory.

Task 3:
To display only the requests for the passive mode connection, in the filter toolbar, enter ftp.request.command==”EPSV” . The results are displayed as shown in the figure below.

The PASV command is issued by the client to request that the server listens on a separate data connection to be established by the client. When the server responds to the PASV command, it includes its IP address and the port number that it will be listening on for the PASV connection. The result is displayed, as shown in the figure below.



After entering the passive mode connection, the client can request for a list of files (packet #1128).



To view the FTP command channel and the FTP data channel, in the filter toolbar, enter ftp || ftp-data .

The FTP data channel traffic can be run over a dynamically-defined port number. To identify traffic matching the ftp-data filter, Wireshark parses the address and port information in packets containing the PORT command or in response packets to the PASV command.

In an active mode data transfer, the client issues the PORT command and indicates the IP address and port number that it will listen on for a data channel connection that will be established by the server. The PORT command is sent by an FTP client to establish a secondary connection (address and port) for data to travel over. In some FTP implementations port 20 is used for data, but that is the exception rather than the rule.

All servers aren't able to apply and set the data server.

Notes:

Repeat the previous steps on a different FTP server navigating from the root to some of the available paths (by using the command cd dir-name). Try to download a file (by using the command `get file-name`) and gain confidence with the possible FTP messages that are involved in the transfer. Identify the connection establishment and the mode used for communication.

If you are using Linux, you may want to connect via FTP in passive mode to list files and directories on the server, i.e., `ftp -p test.rebex.net` .

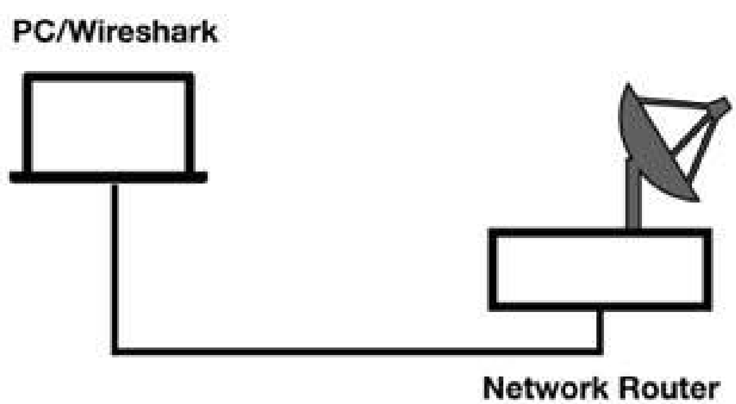
Lab 71. FTP Problems and Packet Structure

Lab Objective:
Learn about the more common FTP problems and how to dissect the packet structure.

Lab Purpose:
Learn how to detect and analyze the more common FTP problems and recognize each field of the FTP packet.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
A lot of problems can occur during FTP connections. In this task, we will identify one such problem.

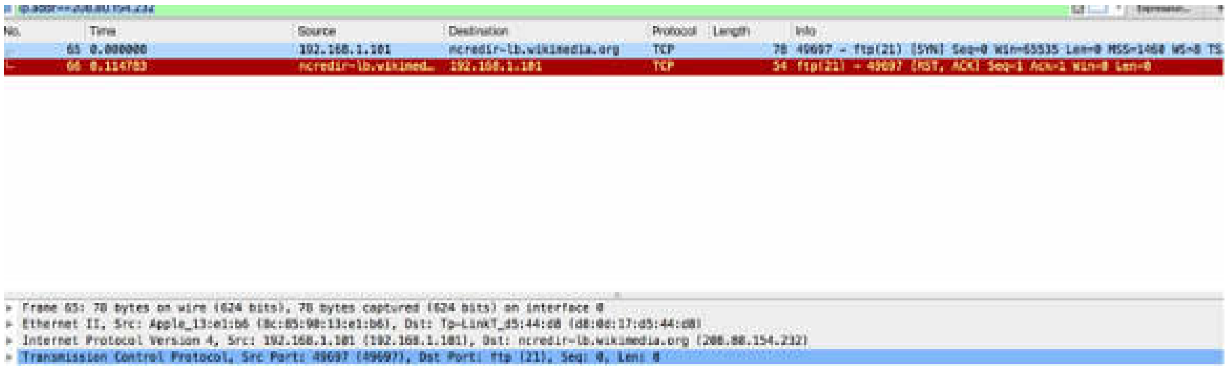
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

Open a terminal window and run the command ftp **www.wikipedia.com** . You will see that the connection couldn't be established, as shown in the figure below.

Stop the capture and save the file, naming it "ftp-problems-1.pcapng".

In Wireshark, in the filter toolbar, enter ip.addr==208.80.154.232 , where 208.80.154.232 is the public IP address of **www.wikipedia.com** . You can verify it in the command terminal by running the command ping www.wikipedia.com . Note that this IP address may change over time.

In this case, you aren't able to connect to the FTP server because an FTP daemon is not running on wikipedia.com. Observing the capture file in detail, you can see that the TCP handshake started with packet #65 (TCP SYN) is stopped with a TCP RST response (packet #66), as shown in the figure below.



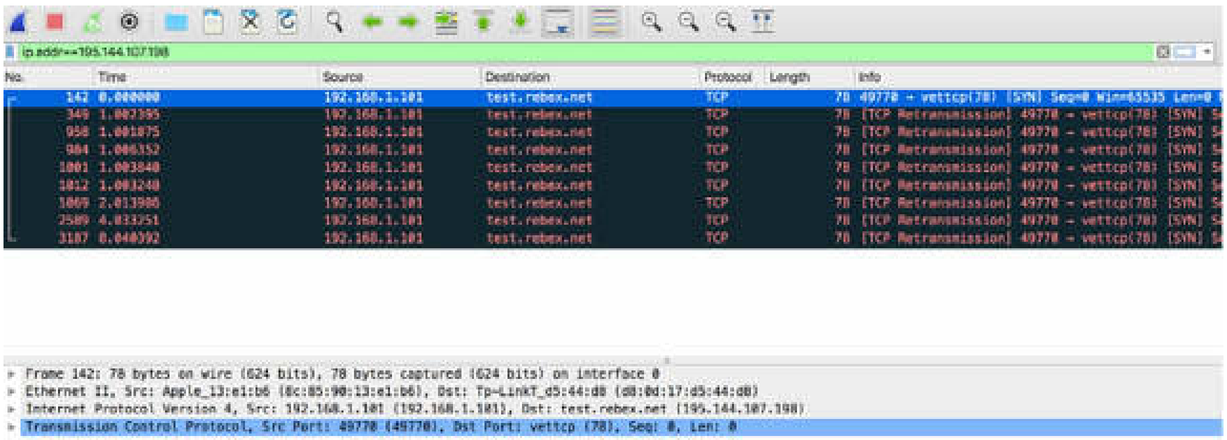
Task 2:

Another case of unsuccessful FTP access happens when the FTP server is configured to use a different port than the client uses. This results in an FTP connection that cannot be established properly.

Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column. Capture the traffic for a few minutes.

Open a terminal window and run the command `ftp test.rebex.net:78`. In this command, you are trying to access the FTP server through a port different than the default one. As a result, you are not able to access the FTP server. Stop the capture and save the file. On Ubuntu, the command is `ftp test.rebex.net 78` .

Inspecting the trace file indicates that the TCP handshake cannot be performed. In fact, no response to the TCP SYN can be identified because the server never sent a response. The figure below shows the first TCP SYN packet (#142) and eight TCP retransmission packets because none of them have received a response.

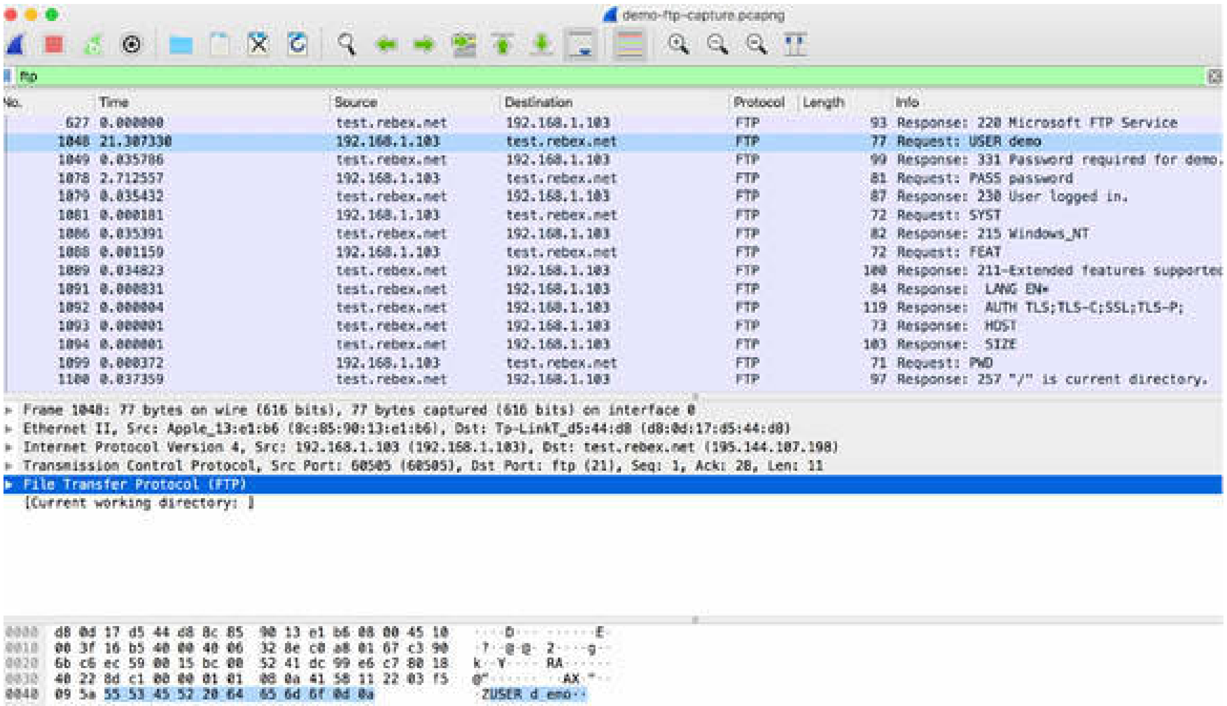


There are other cases where firewalls block the passive mode support so that the passive mode connection attempts fail. In such cases, the client sends the PASV command, and the server responds with its IP address and a port number for the passive mode connection. This indicates that the server itself supports passive mode connections. The client attempts to make a connection on the port provided, but the server does not respond to the connection attempts.

If the port is open, the server should respond with the SYN/ACK response. If the port is closed, the server should respond with a TCP RST response, as shown in the previous task. If no response is received, a firewall along the path or on the server may be blocking connection attempts to this port. After some attempts to make a connection, the client gives up. In addition, the client shuts down the command channel.

Task 3:

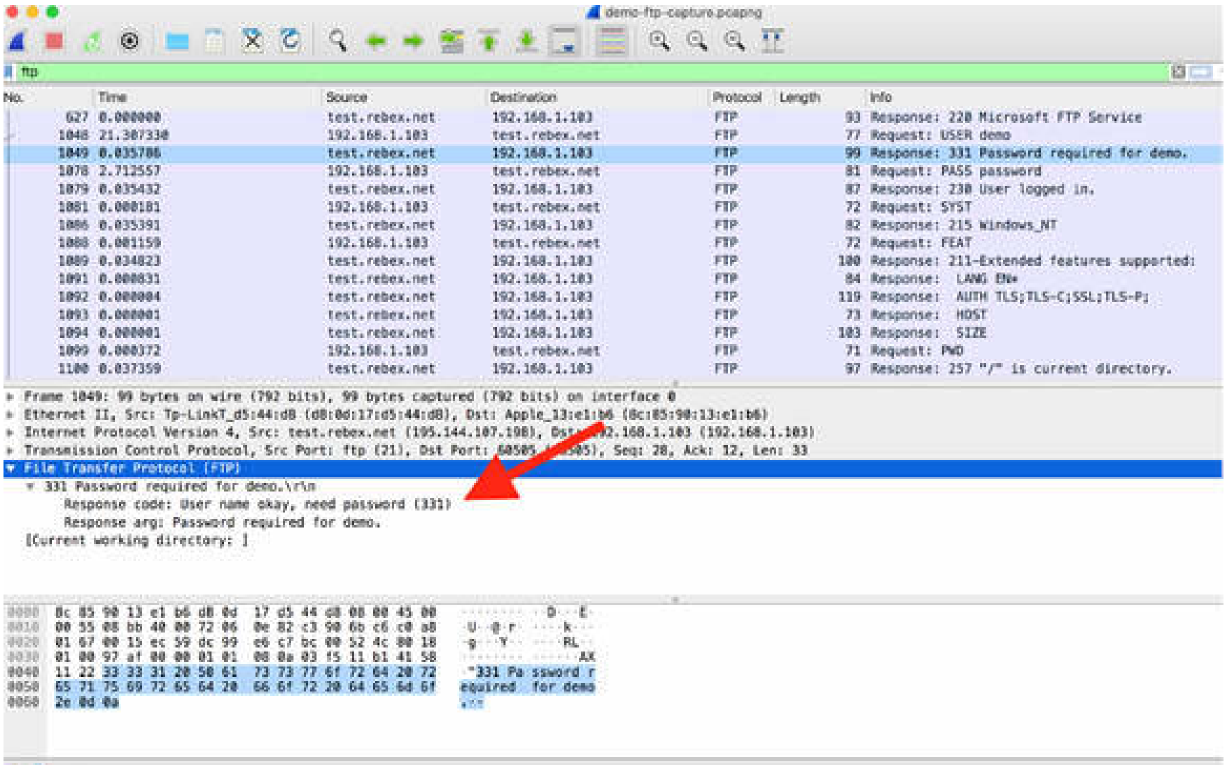
Open the capture file demo-ftp-capture.pcapng saved in the previous lab. In the filter toolbar, enter ftp . The results are similar to the ones displayed in the figure below, where the FTP packet structure is very clear.



The FTP packet structure is very simple. The FTP commands follow immediately after the TCP header, as shown in the figure above. Some commands include an argument. The following list provides a list of commands that use arguments:

- USER: username
- PASS: password
- RETR: directory/file name
- TYPE: representation type
- PORT: IP address, port number

Responses contain a numerical code and text, as shown in the figure below (from the same capture file). In the response packet, the response code and the response arguments are in plain text.



Data packets have an even simpler format because the data follows the TCP header, and no extra commands are required or allowed on this channel.

Notes:

To identify the FTP problems and practice packet dissection, repeat the previous steps on different FTP servers supporting and not supporting FTP services.

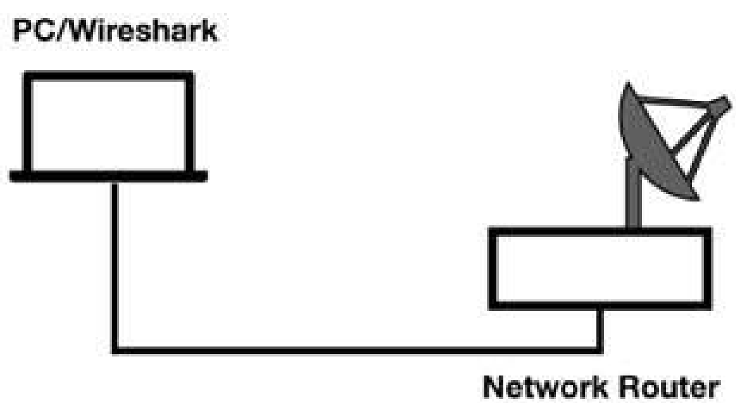
Lab 72. Email Traffic—Post Office Protocol

Lab Objective:
Learn how the Post Office Protocol (POP) works and why is it used.

Lab Purpose:
Understand the main purpose of POP and its features.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

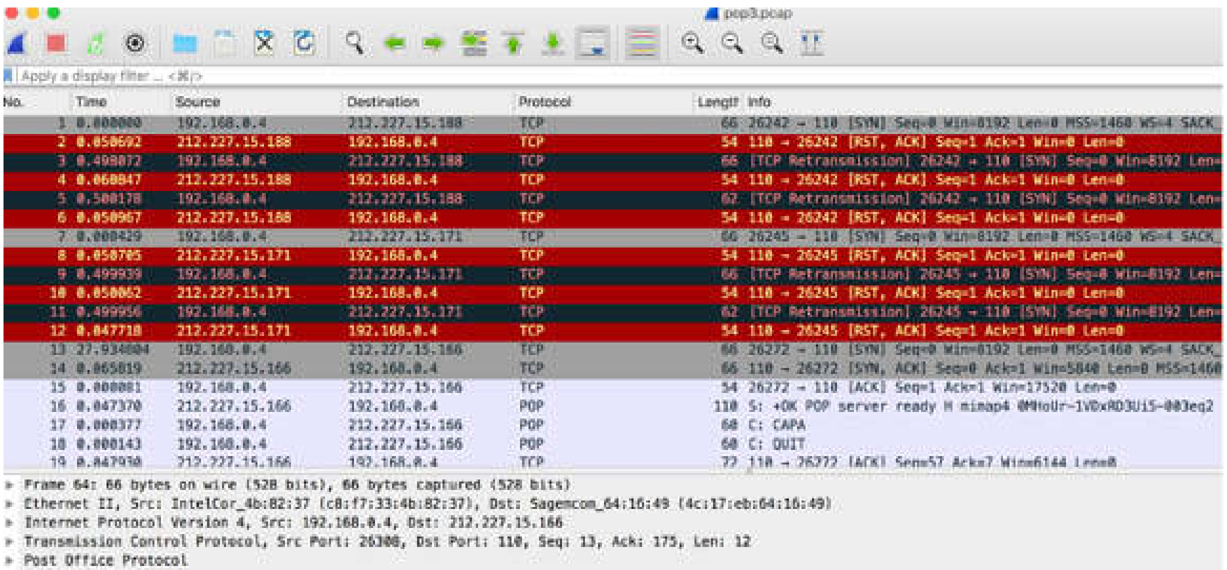


Lab Walkthrough:

Task 1:

POP, defined in RFC 1939, is still a very popular method for retrieving emails. POP itself does not provide security in email data transfer. The third-party applications and tools provide this added functionality.

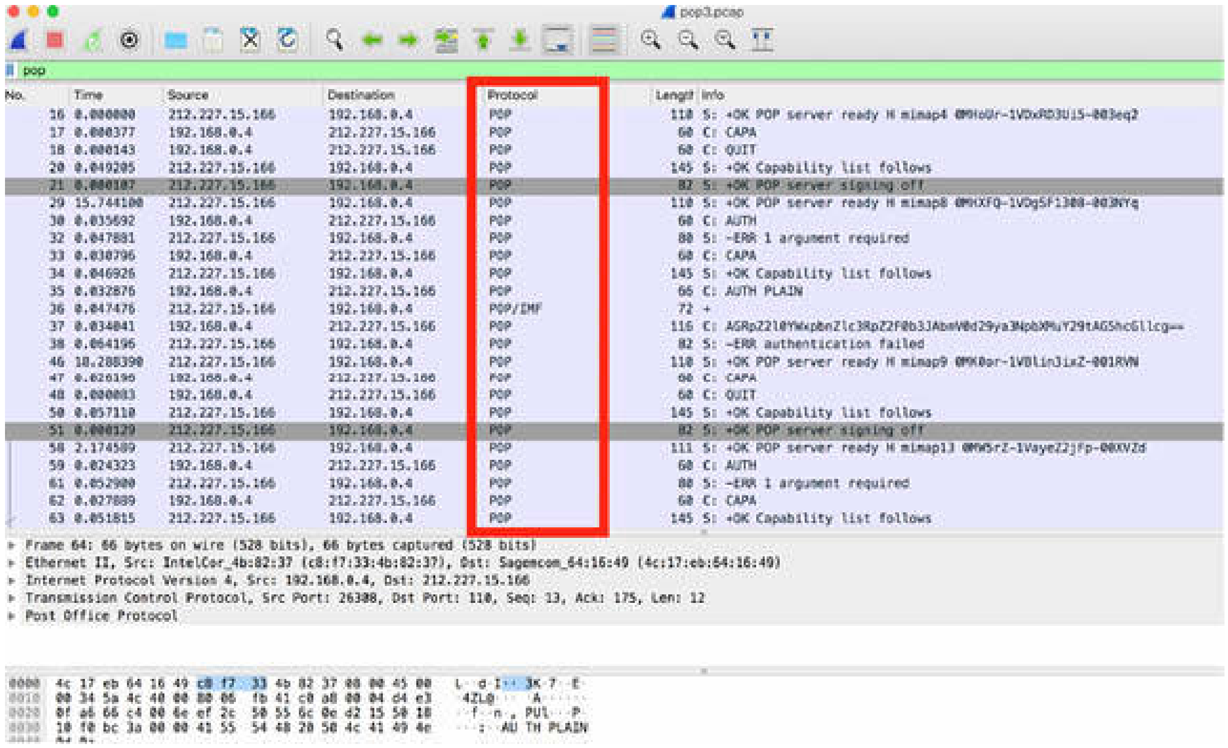
Download the pop3.zip file from https://asecuritysite.com/forensics/pcap?infile=pop3.pcap . Unzip this file and open it in Wireshark. The result will be similar to the ones shown in the figure below.



Notice that at the beginning of the communication, there are some problems in the TCP connection because of latency and packet loss. This is evident from the rate of packet retransmission. The connection is lost, and the client issues two TCP handshake requests (SYN) to again connect with the server.

Task 2:

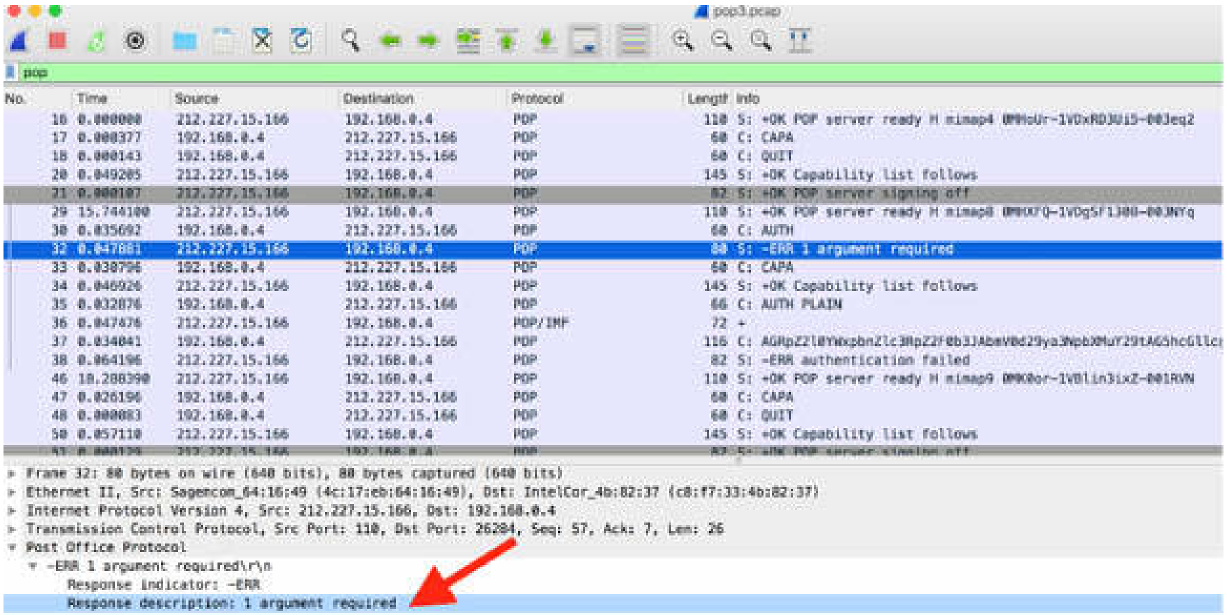
In the filter toolbar, enter pop . All packets belonging to POP are displayed in the Packet List pane, as shown in the figure below.



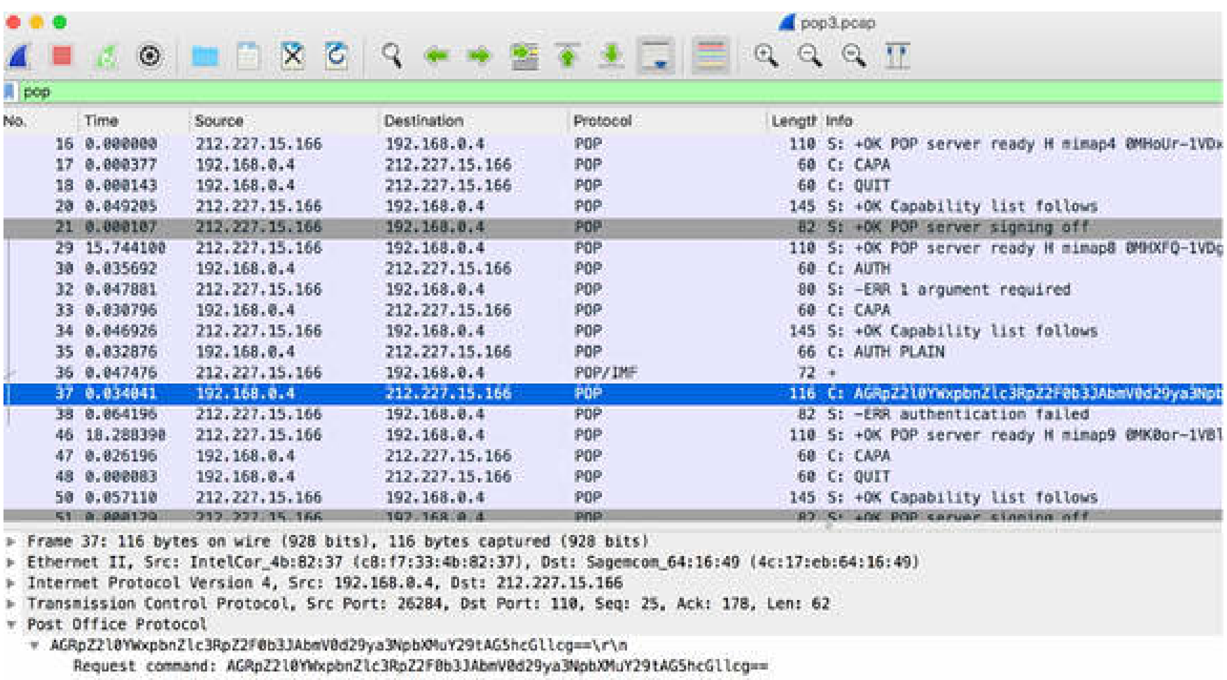
As shown in the figure above, to enable you to interpret the entire process quite easily, the Info column provides enough details about POP communications.

Packet #29 signals that the server is ready to accept communication from the client. In the next packet (#30), the user tries to log in with the AUTH command.

Packet #32 is the response of the server. In the Packet List pane, select packet #32, and in the Packet Details pane, open the tree view. The results are similar to the figure below, where you can see that the password is required.

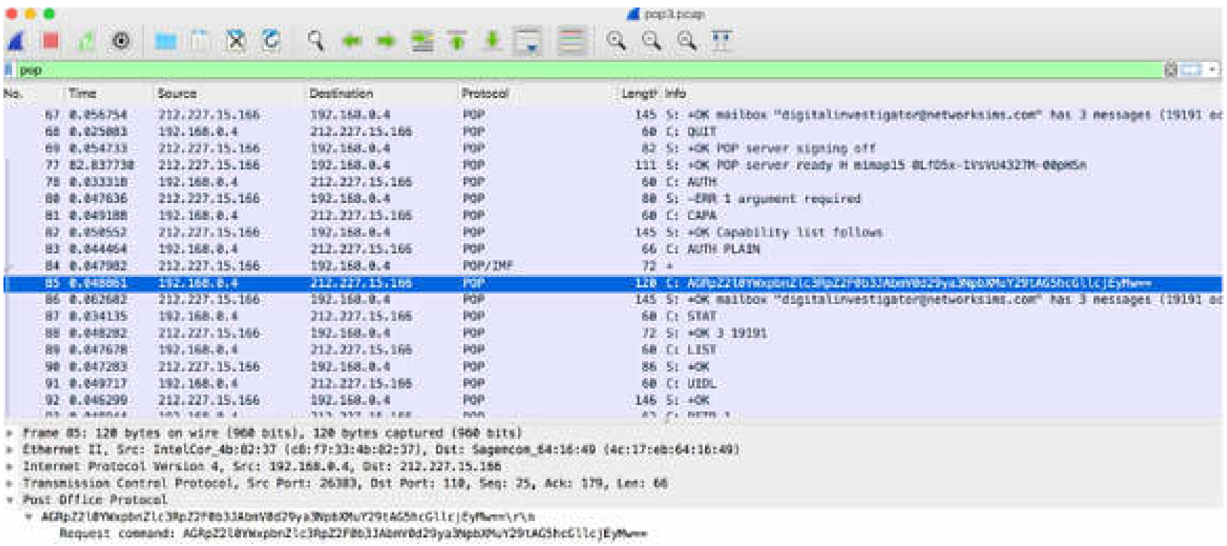


Task 3:
In the Packet List pane, select packet #37, and in the Packet details pane, open the tree view. As shown in the figure below, the client provides the required password with the command AUTH.



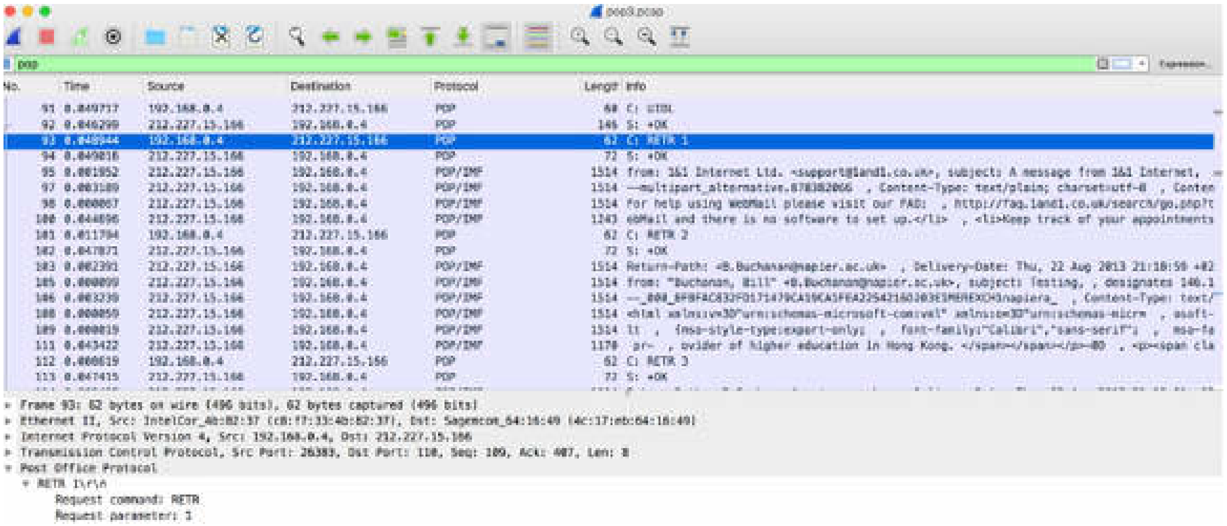
The password, however, is incorrect. As a result, the server issues the "Authentication failed" message (packet #38).

In such a case, a new authentication procedure is required to correctly access the required grants. This is done by the client with packet #85. Successful login is displayed in the figure below.



The POP server opens the mailbox and tells the user that three messages are waiting (packet #86).

The client asks for the Unique Identification Listing (UIDL) with packet #91 before issuing the RETR command (packet #93). The POP server begins sending the data to the client over multiple TCP packets (if necessary). As shown in the figure below, the data transfer starts (from packet #95).



In general (but this is not evident in this capture example), upon successful download of the email message, the client sends the delete command (DELE), and the server responds indicating it has deleted the message. The POP communications are then terminated by the client.

POP does not maintain a persistent connection—the connection is established to retrieve the email and then terminated upon successful completion (packets #120 and #121).

Notes:
To identify the process that a user goes through to log in and authenticate to a POP server, and download the messages, repeat the previous steps by using different POP capture examples. Using different use cases will give you the necessary confidence in understanding the general process related to POP.

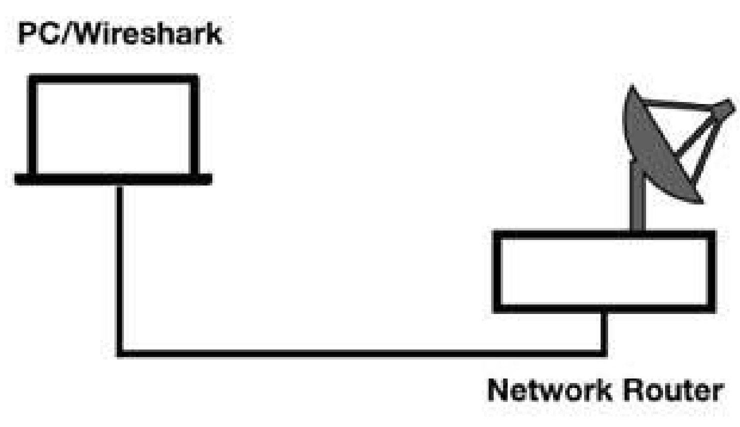
Lab 73. POP Packet Structure and Filtering

Lab Objective:
Learn how to dissect POP packets and how to use filters.

Lab Purpose:
Understand the structure and various fields of a POP packet and learn to build and use appropriate filters in Wireshark.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

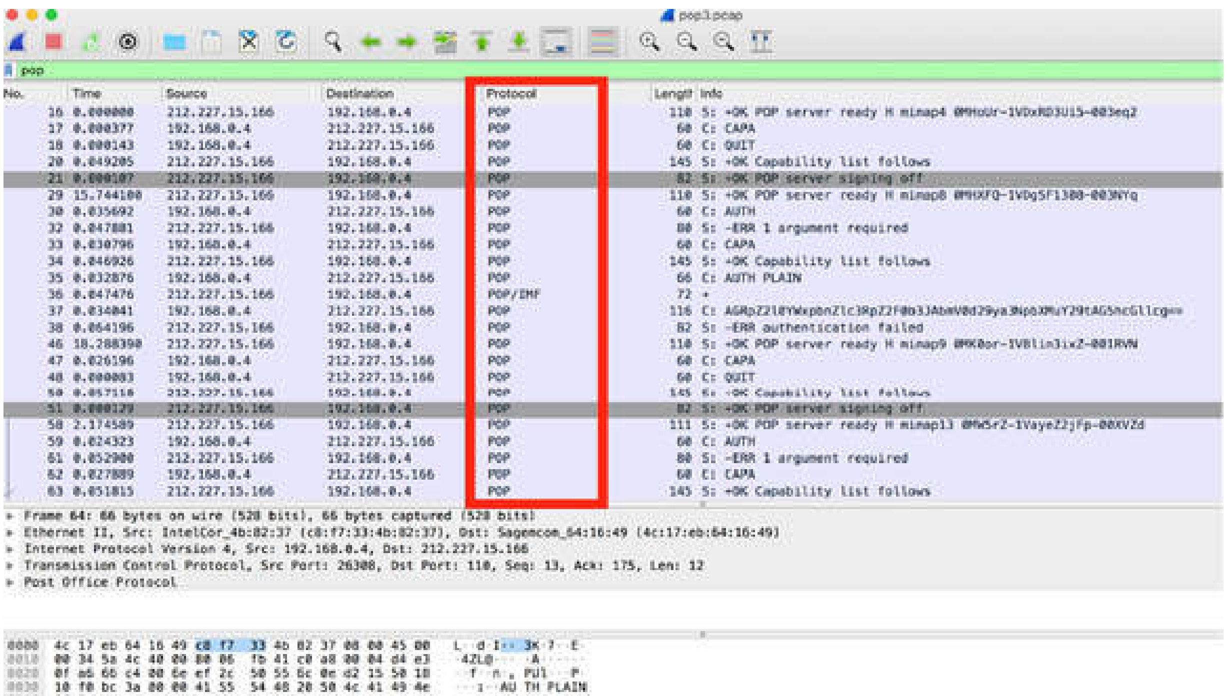


Lab Walkthrough:

Task 1:

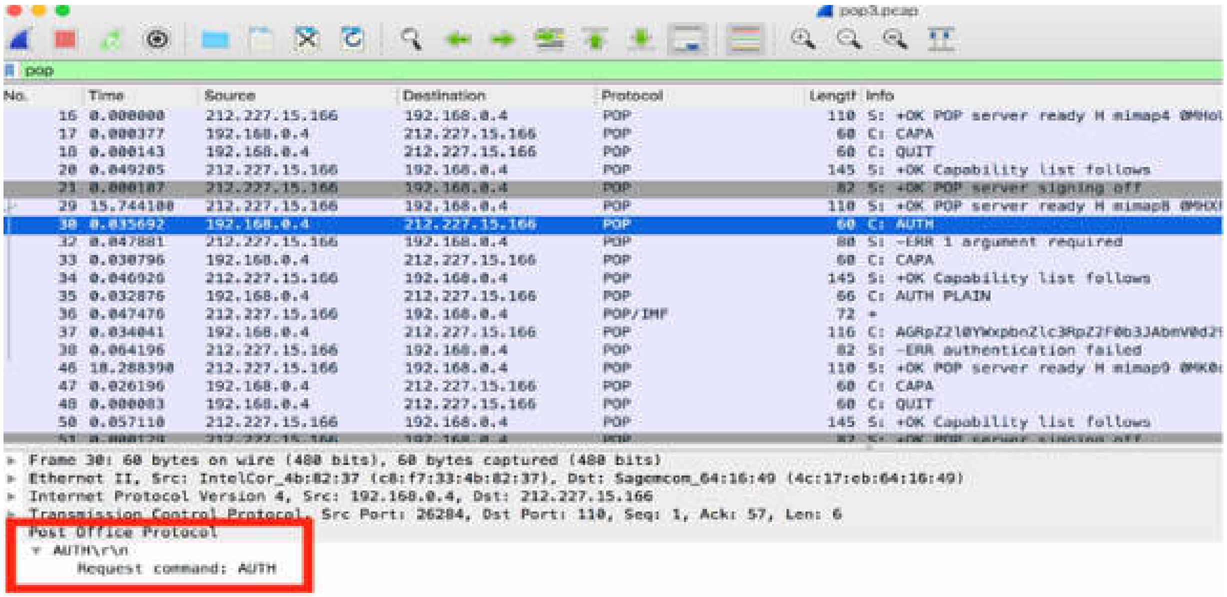
Download the pop3.zip file from **https://asecuritysite.com/forensics/pcap?infile=pop3.pcap** . Unzip this file and open it in Wireshark.

In the filter toolbar, enter pop . All packets belonging to POP are displayed in the Packet List pane, as shown in the figure below.

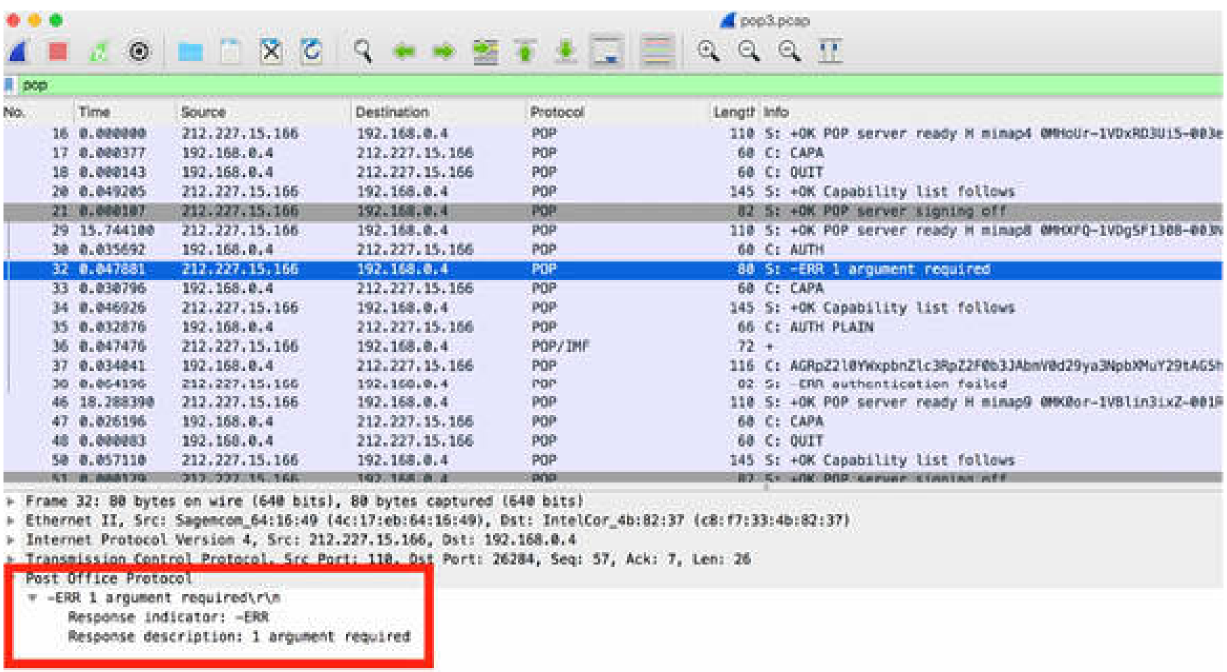


The structure of a POP packet is very simple. POP requests consist of a request command and a request parameter. POP responses consist of a response indicator and a response description.

In the Packet List pane, select packet #30 (a request from the client), and inspect the related content in the Packet Details pane.



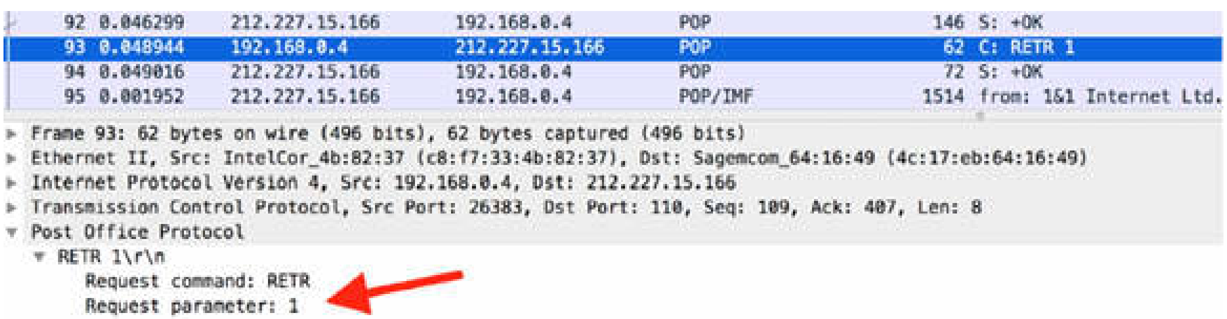
The client issues the AUTH Request command. As a response, the server sends packet #32, with response indicator "–ERR" and response description "1 argument required", as shown in the figure below.



The following list describes some of the request commands:

- USER: Indicates the user name
- PASS: Indicates the password
- QUIT: Terminates the connection
- AUTH: Indicates an authentication mechanism to the server
- STAT: Obtains the server status
- LIST: Lists message and message size
- RETR: Retrieves a message
- DELE: Deletes a message
- PIPELINING: Indicates that the server can accept multiple commands at a time
- Unique ID List (UIDL): Lists all emails

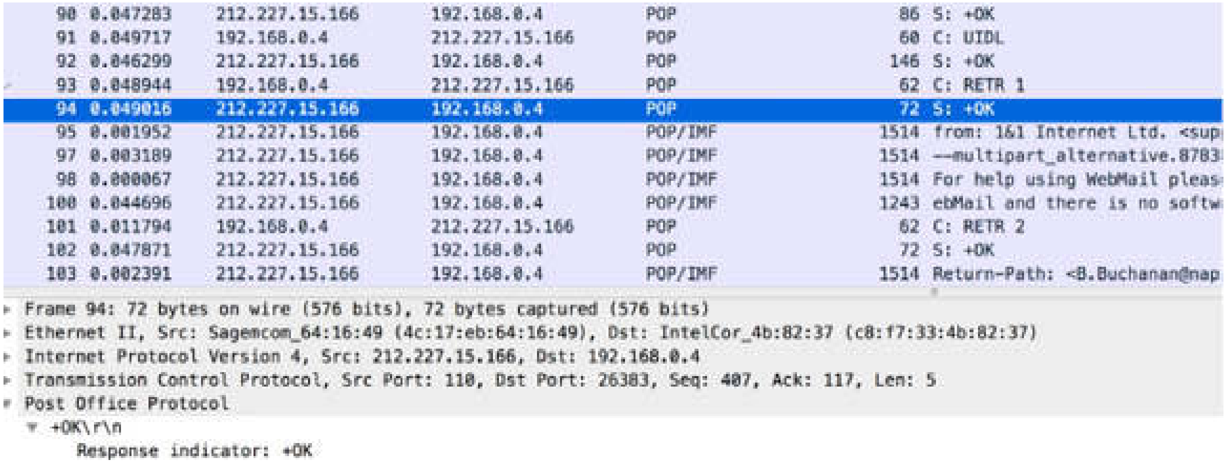
For example, as shown in the figure below, for packet #93 (RETR command), the Request parameter is 1. This means that the client wants to retrieve the message number 1.



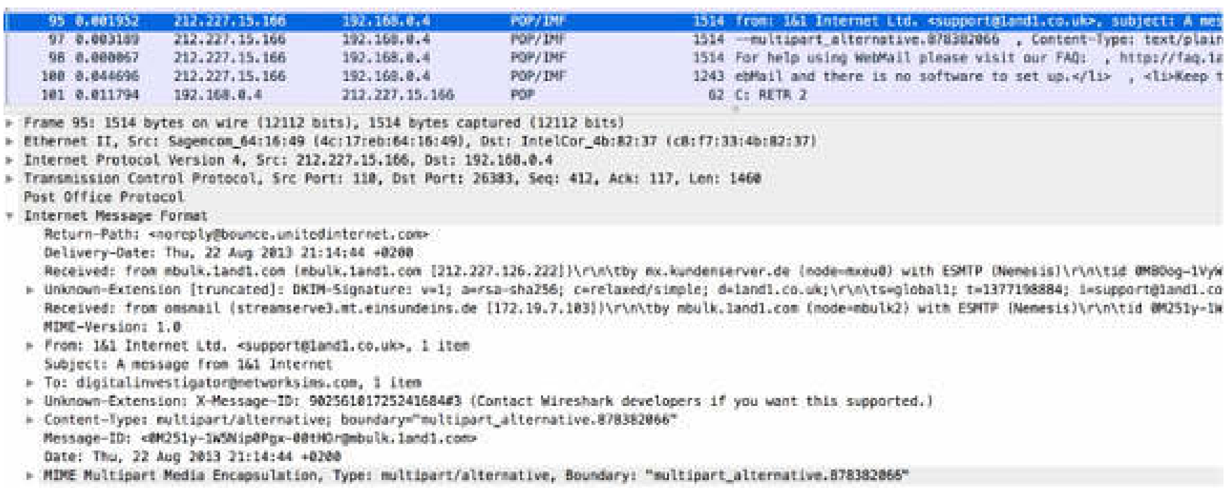
The response begins with the response indicator and the response description. Only two response indicators are used in POP communications: +OK and –ERR.

The first one is a positive response; the second one indicates an error. In case of an error, the Response description provides the error details.

In response to the RETR command, the server sends the +OK response (packet #94), as shown in the figure below:



To identify the path a packet took through mail exchange servers, in the Packet List pane, select packet #95 containing the mail message, and in the Packet Details pane, inspect the content.

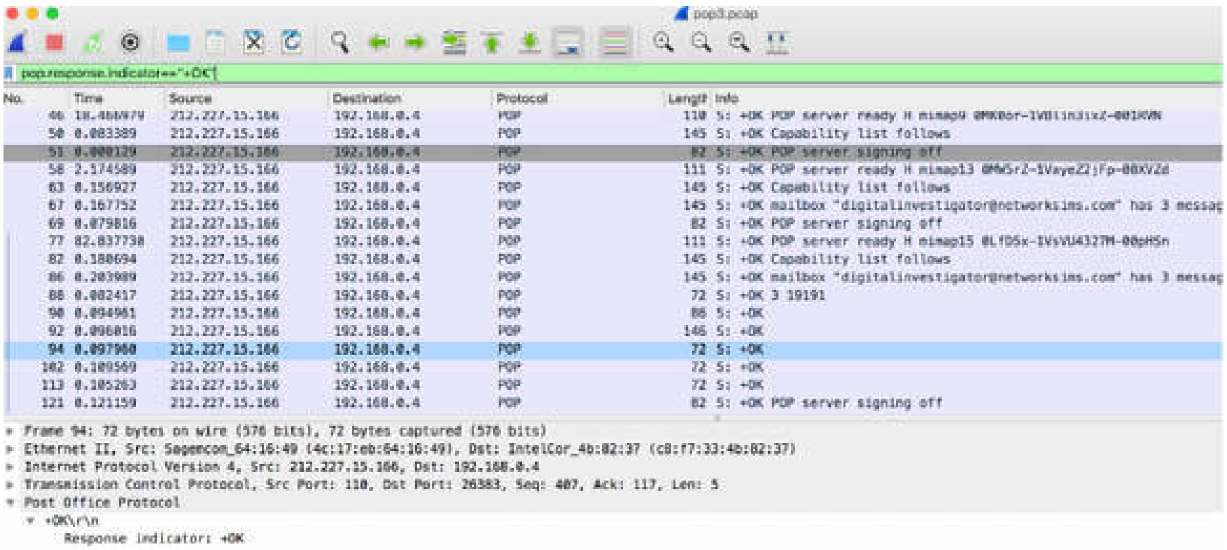


Task 2:

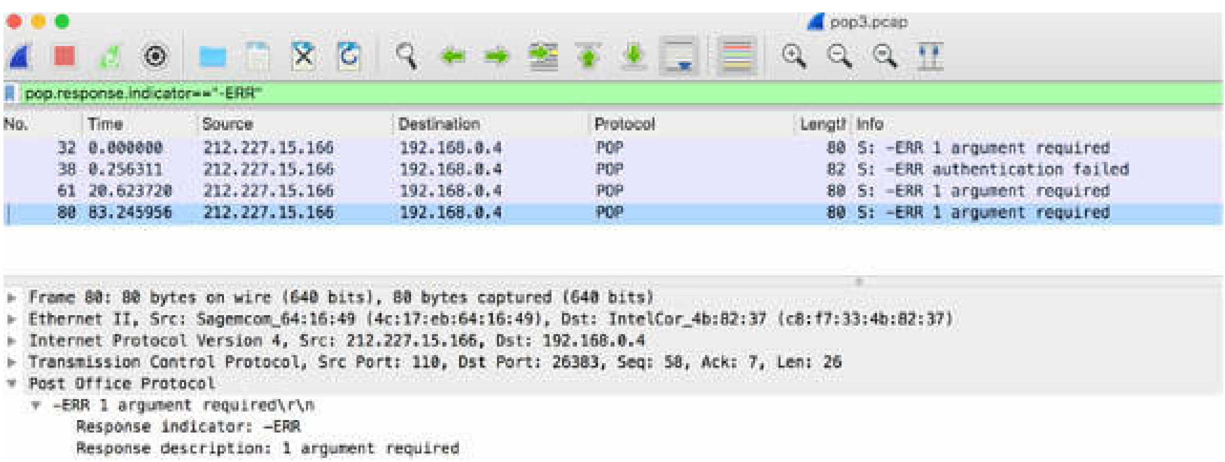
As shown in the previous task, the pop display filter is used for POP. For a capture filter, use tcp.port == 110 because, by default, POP communication uses TCP port 110. If the POP traffic runs on a different port, change the capture filter accordingly.

To display only specific messages belonging to POP communication in the Packet List pane, use one of the following filters.

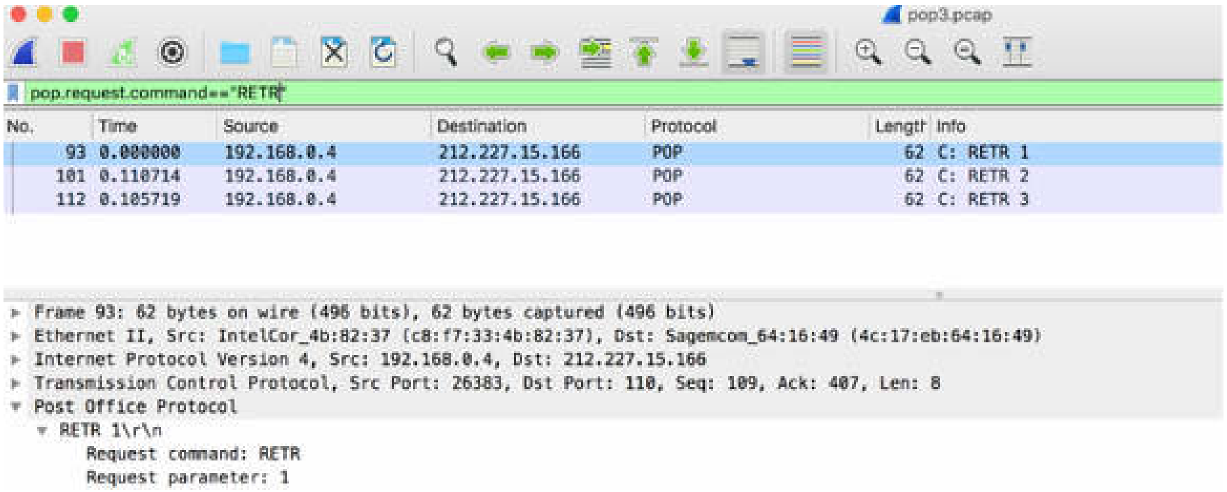
To display only the POP +OK responses, in the filter toolbar, enter pop.response.indicator==”+OK , as shown in the figure below.



To display only the POP –ERR responses, in the filter toolbar, enter pop.response.indicator==”-ERR” , as shown in the figure below.



To display only a specific request command (for example, RETR), in the filter toolbar, enter pop.request.command==”RETR” , as shown in the figure below.



To display only a specific email message requested with the RETR command, in the filter toolbar, enter (pop.request.command==”RETR”) && (pop.request.parameter==”1”) . This is a composite filter.

Notes:

To gain confidence with the protocol dissection and to be able to apply an appropriate filter (both capture and display), repeat the previous steps by using a different pop capture example. Create more composite filters to select only those packets that you are interested in.

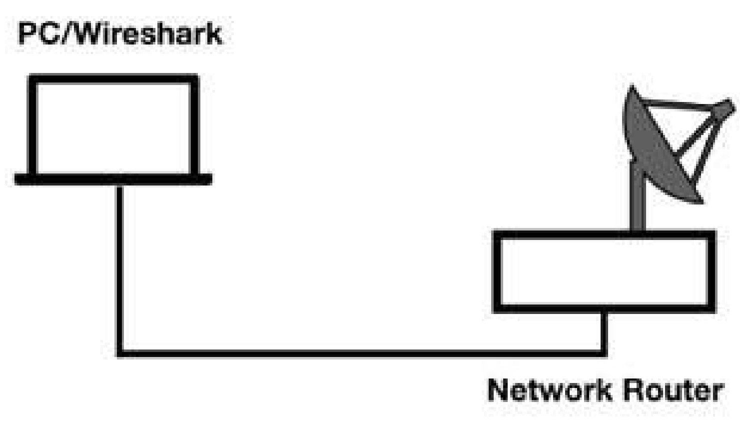
Lab 74. Email Traffic—Simple Mail Traffic Protocol

Lab Objective:
Learn how the Simple Mail Traffic Protocol (SMTP) works and why is it used.

Lab Purpose:
Understand the main purpose of SMTP and its features.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

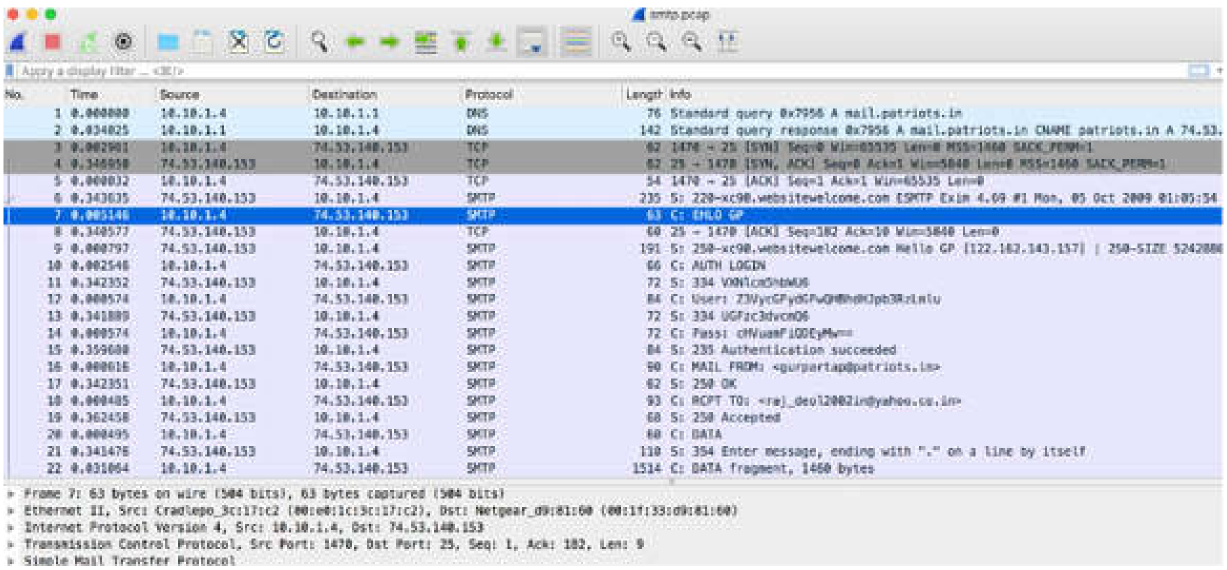
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



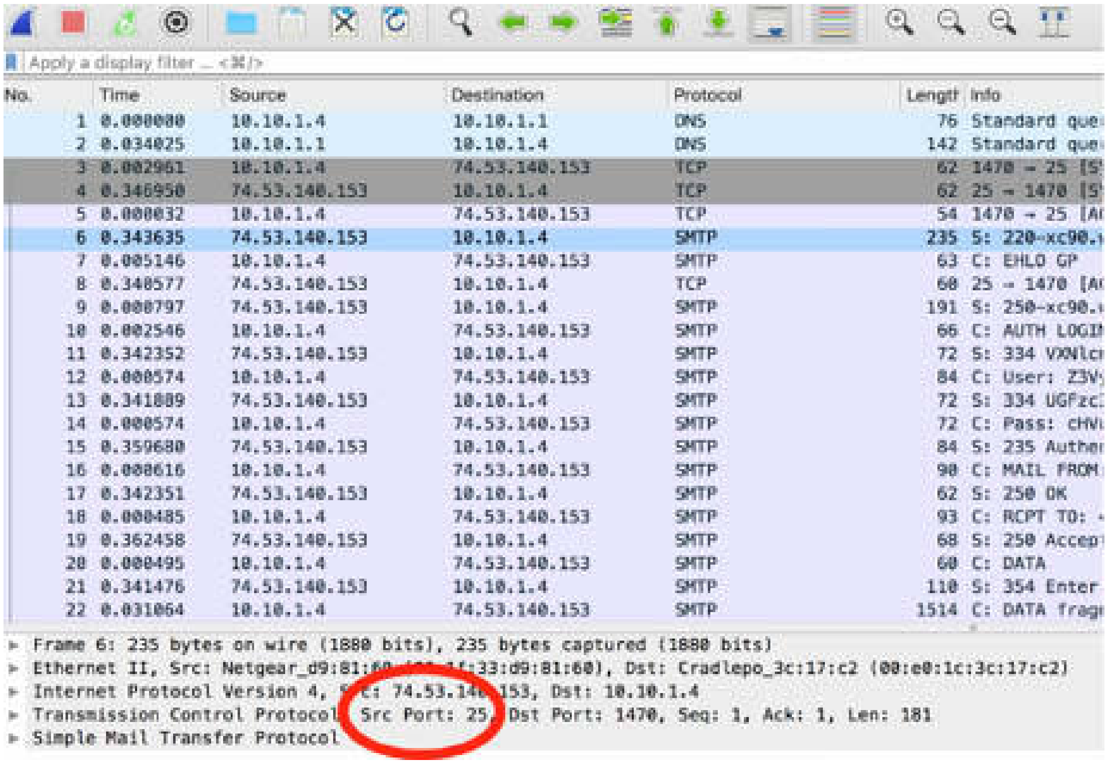
Lab Walkthrough:

Task 1:
SMTP, defined in RFC 5321, is the standard application used for sending emails. SMTP uses Sender-SMTP and Receiver-SMTP processes. By default, SMTP communications are not secure.

Download the sample capture file smtp.pcap to your PC from **https://wiki.wireshark.org/SampleCaptures** and then open the file in Wireshark. The Packet List pane will look as shown in the figure below.

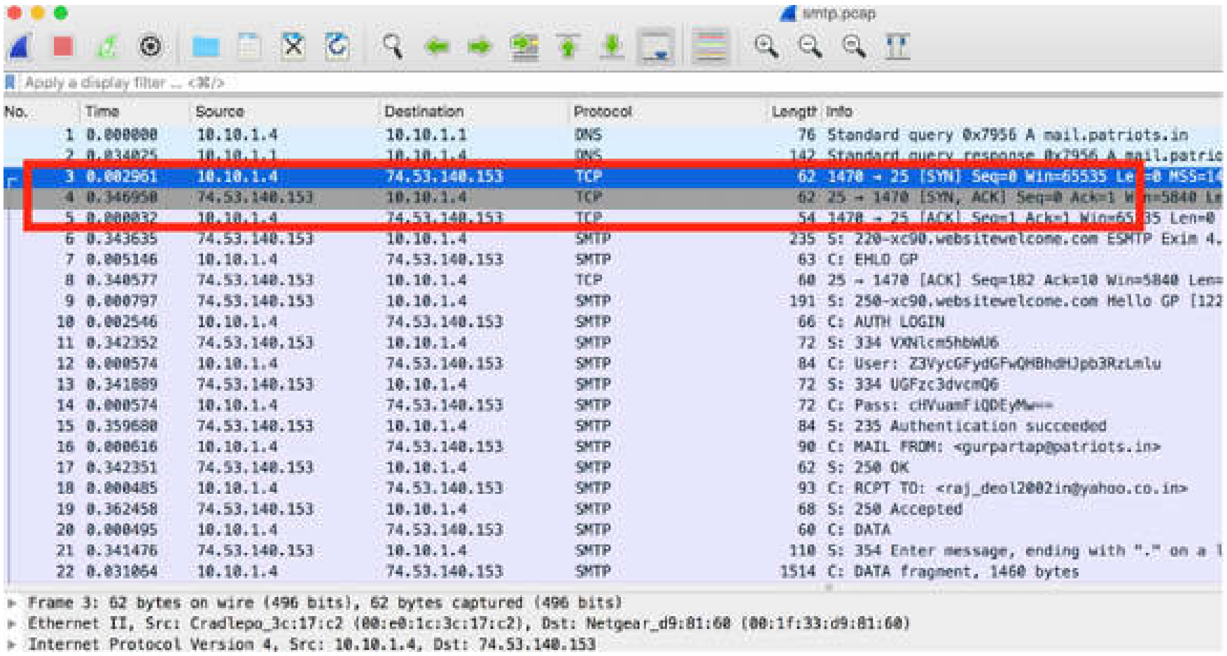


The default port used for SMTP communications is port 25. In the Packet List pane, select an SMTP packet (#6). In the Packet Details pane, port 25 is displayed, as shown in the figure below.

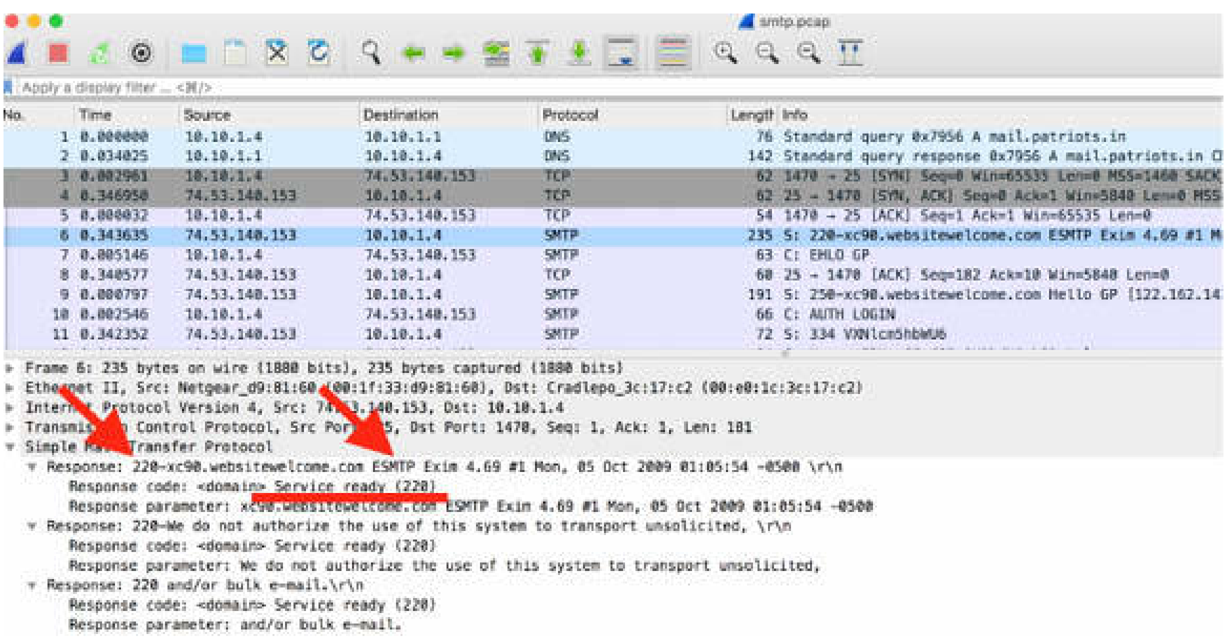


You can also configure SMTP to run on another port number. In fact, a lot of Internet Service Providers and Firewall configurations block the default port 25 to stop the spam from going out through the network on port 25.

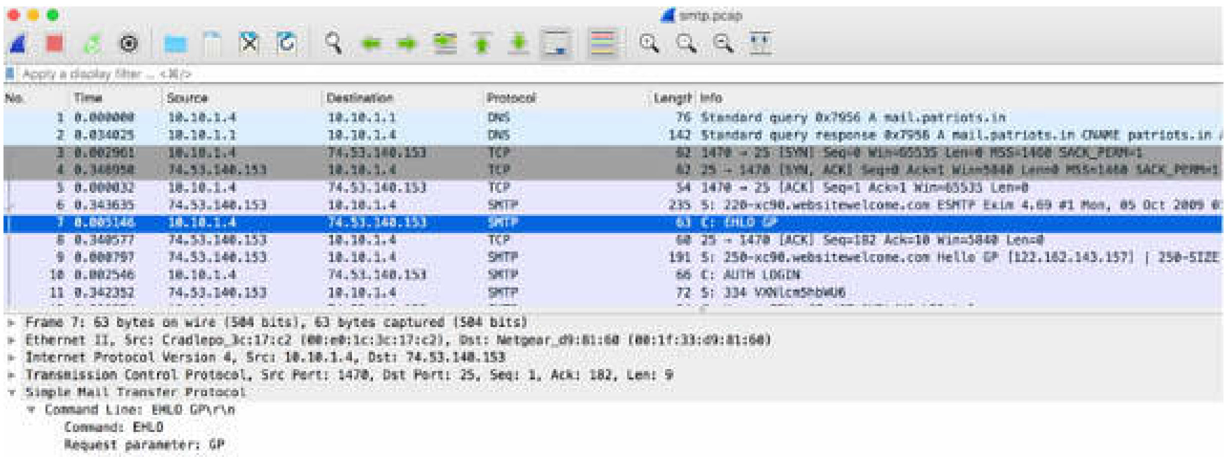
Task 2:
SMTP communication starts with a TCP handshake. In the figure below, the TCP handshake is shown in packets #3, #4, and #5.



In the Packet List pane, select the first answer from the SMTP server (packet #6). In the Packet Details pane, open the tree view. As shown in the figure below, the SMTP response code field contains value 220 indicating that the service is ready. This response also identifies the SMTP server and indicates that the server supports mail extensions by including ESMTP in the greeting.

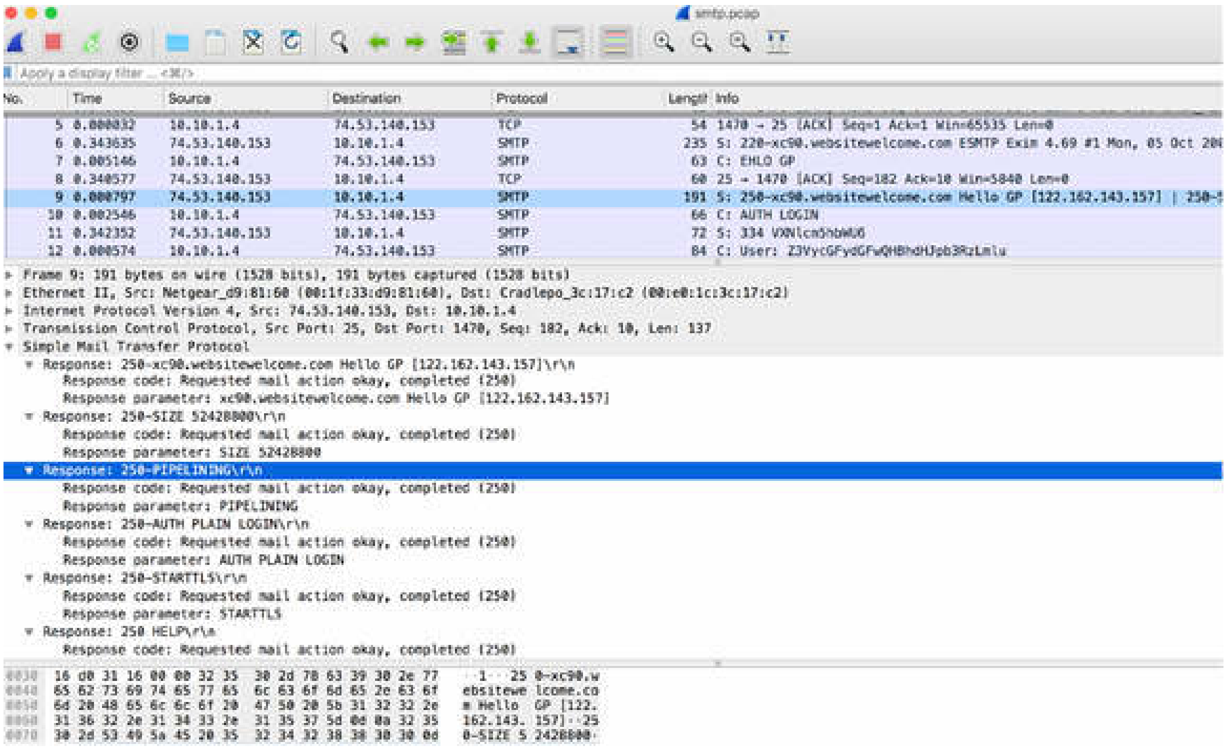


In the Packet List pane, select a packet sent from the client (packet #7). In the Packet Details pane, open the tree view to see the details shown in the figure below.

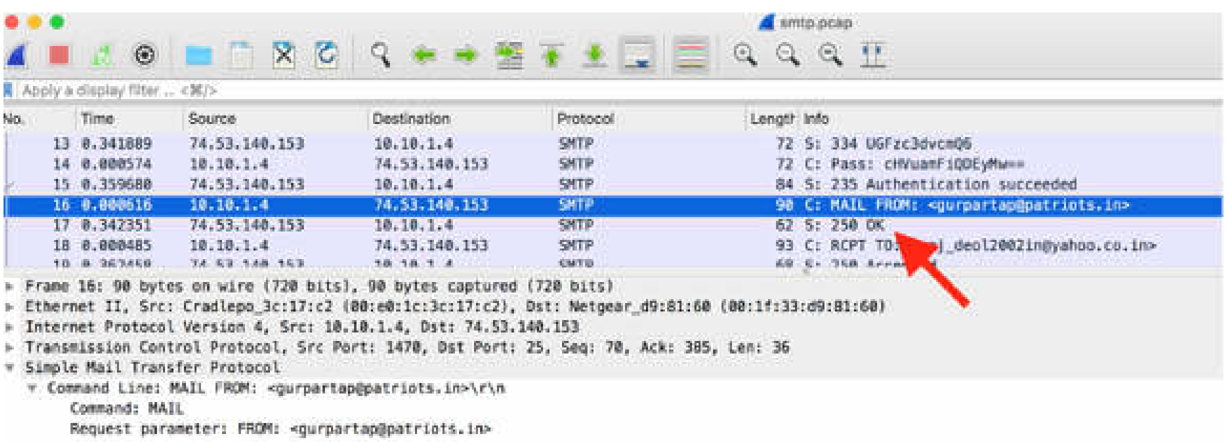


As shown in the figure above, there is an EHLO message with the hostname of the client. This client uses EHLO because the server indicated that it supports mail service extensions in its greeting. An alternative would be if the client had sent a HELO message. HELO initiates a standard SMTP session, whereas EHLO initiates an SMTP session that supports mail service extensions.

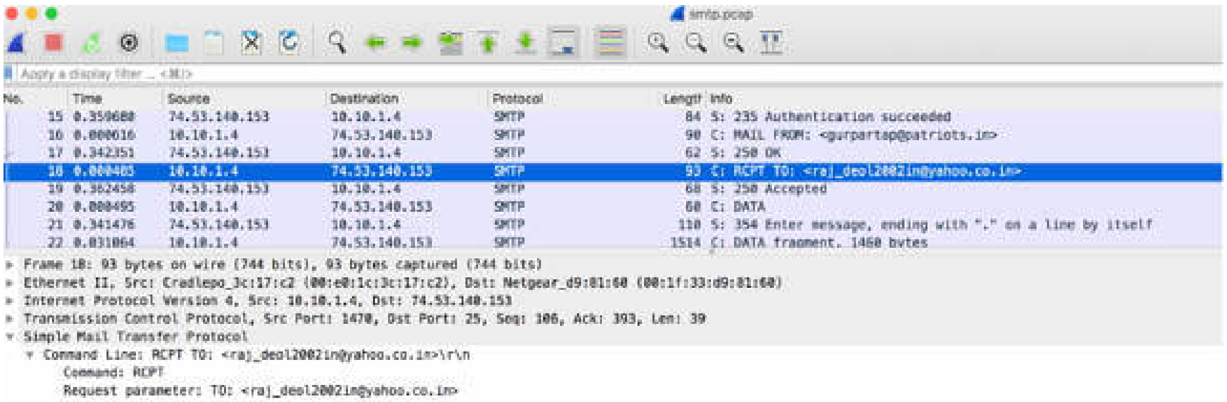
At this point, the server can send capability information to the client. In the figure below, the server sends a packet indicating that it supports pipelining (packet #9). Pipelining indicates that the client can send another request to the server without waiting for responses to the previous one(s).



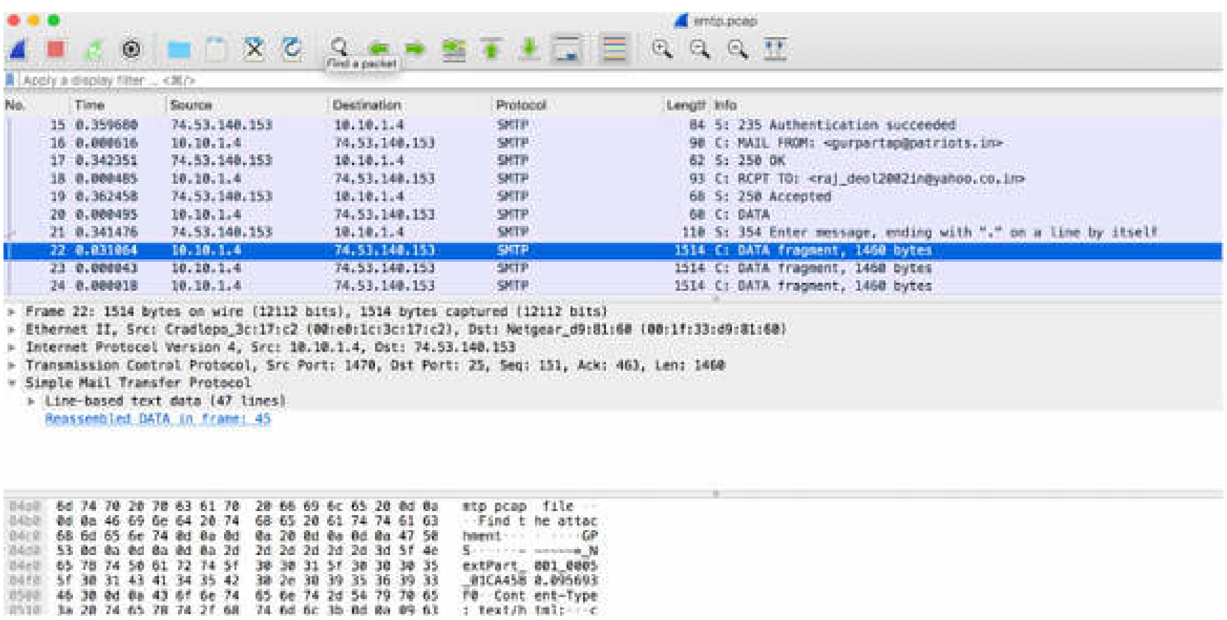
After a successful authentication process (packets #10 to #15), the SMTP client sends the MAIL FROM message and provides its source email address to the SMTP server (packet #16). This address must be approved by the SMTP server (packet #17).



The client sends the RCPT TO message with the next packet, indicating to whom the email will be sent (packet #18), as shown in the figure below.



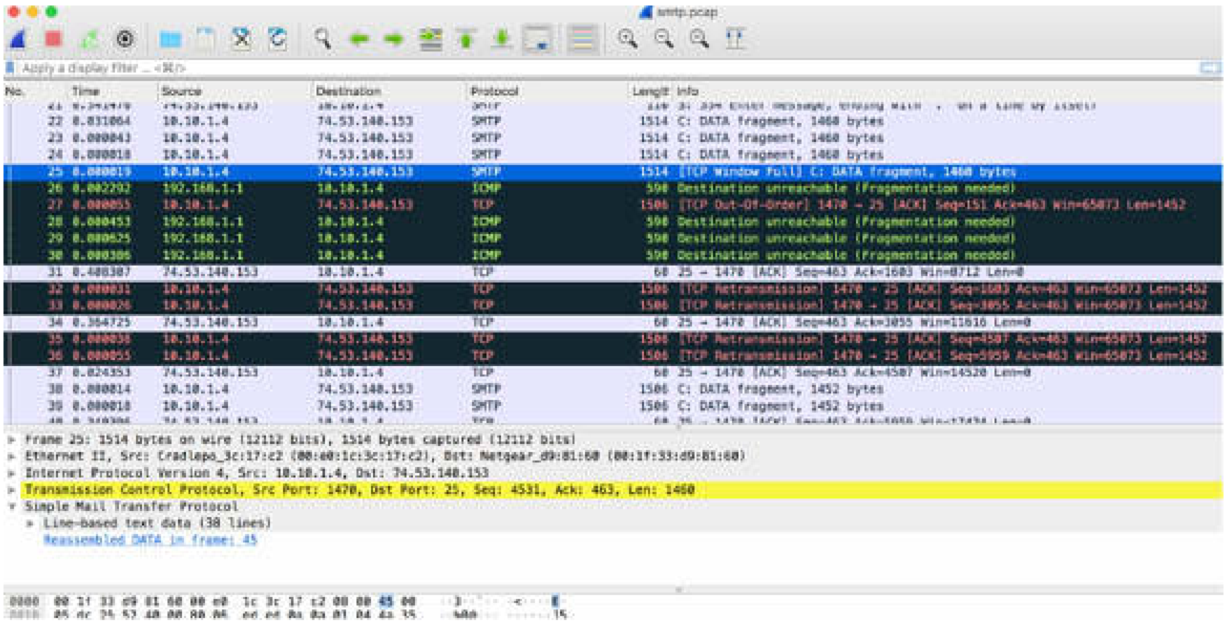
The DATA command (packet #20) indicates that the client is ready to send the email, and if the server is ready (as in this case), it responds with 354 Start Mail Input. Now the client can send the email to the SMTP server. In this case, the message is split into three segments, as shown in the figure below.



After the email is sent, the client issues the QUIT command to begin the connection termination process.

SMTP communication problems can begin with the TCP connection process, and they can also be affected by high latency and packet loss.

As shown in the figure below (from packet #25 to packet #36), when the client transmits the DATA, it causes the TCP window overflow. In such a case, fragmentation is needed, and the TCP retransmission starts.



Notes:

To gain more confidence in analyzing SMTP communication and correctly recognizing the SMTP messages, repeat the previous steps. Analyze SMTP communication between a client and a server and identify the transmission pattern, as described earlier.

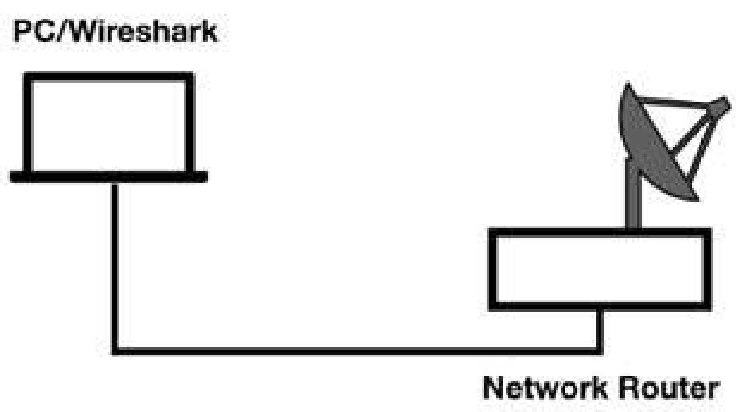
Lab 75. SMTP Packet Structure and Filtering

Lab Objective:
Learn how to dissect SMTP packets and how to use filters.

Lab Purpose:
Understand the structure and fields of an SMTP packet and learn to build and use appropriate filters in Wireshark.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

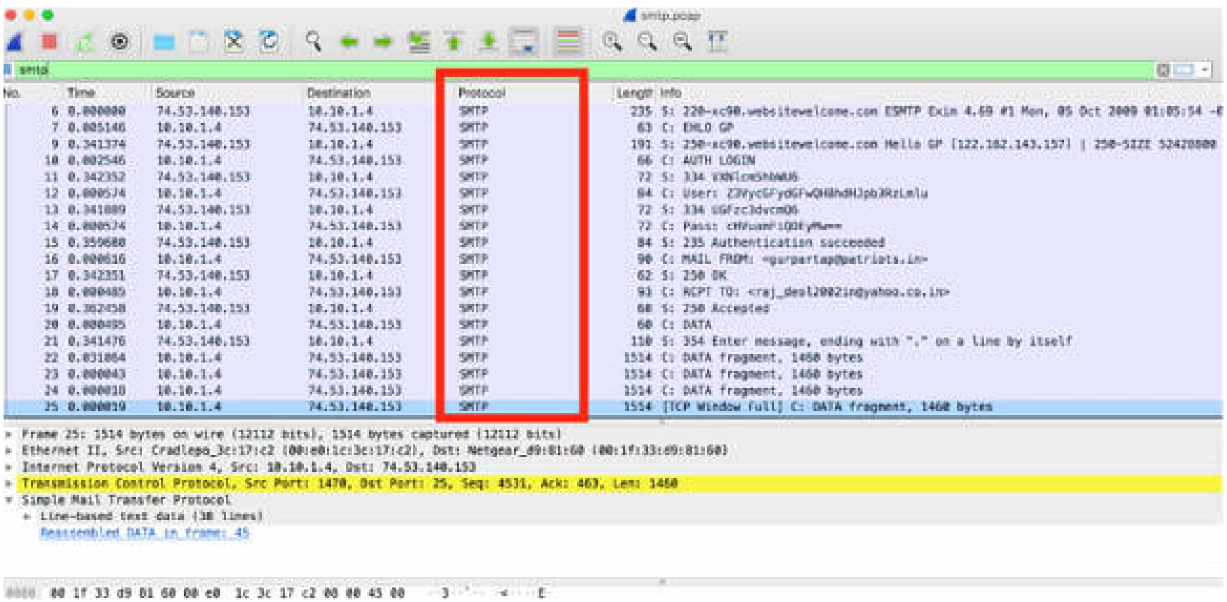
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
Download the sample capture file smtp.pcap to your PC from **https://wiki.wireshark.org/SampleCaptures** and then open the file in Wireshark.

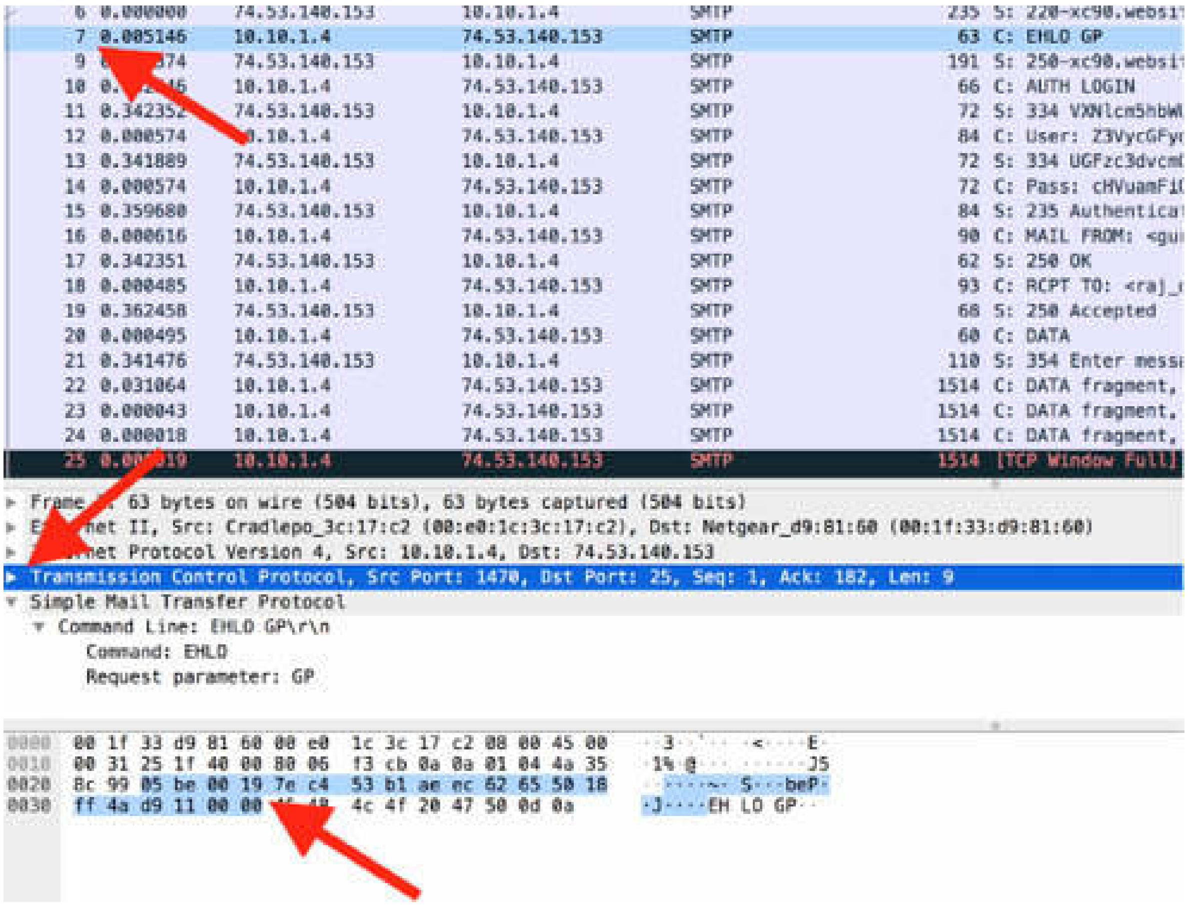
The Packet List pane will look as shown in the figure below. In the filter toolbar, enter smtp to display only SMTP packets.



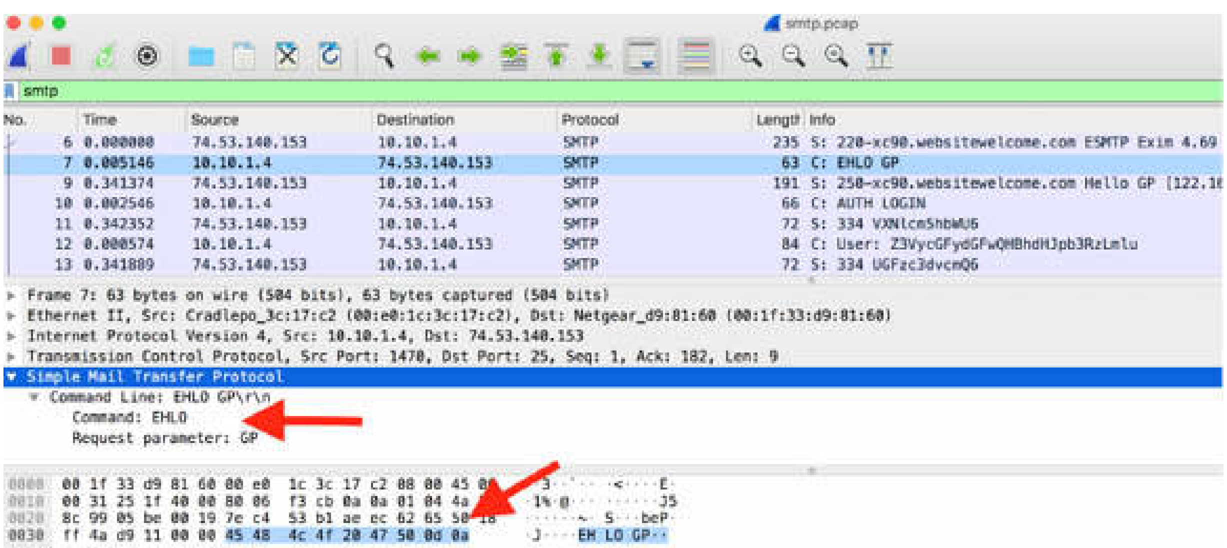
SMTP communications consist of commands and response codes.

In the Packet List pane, select a packet (#7). In the Packet Details pane, open the tree view. As shown in the figure below, SMTP commands and response codes come immediately after the TCP header.

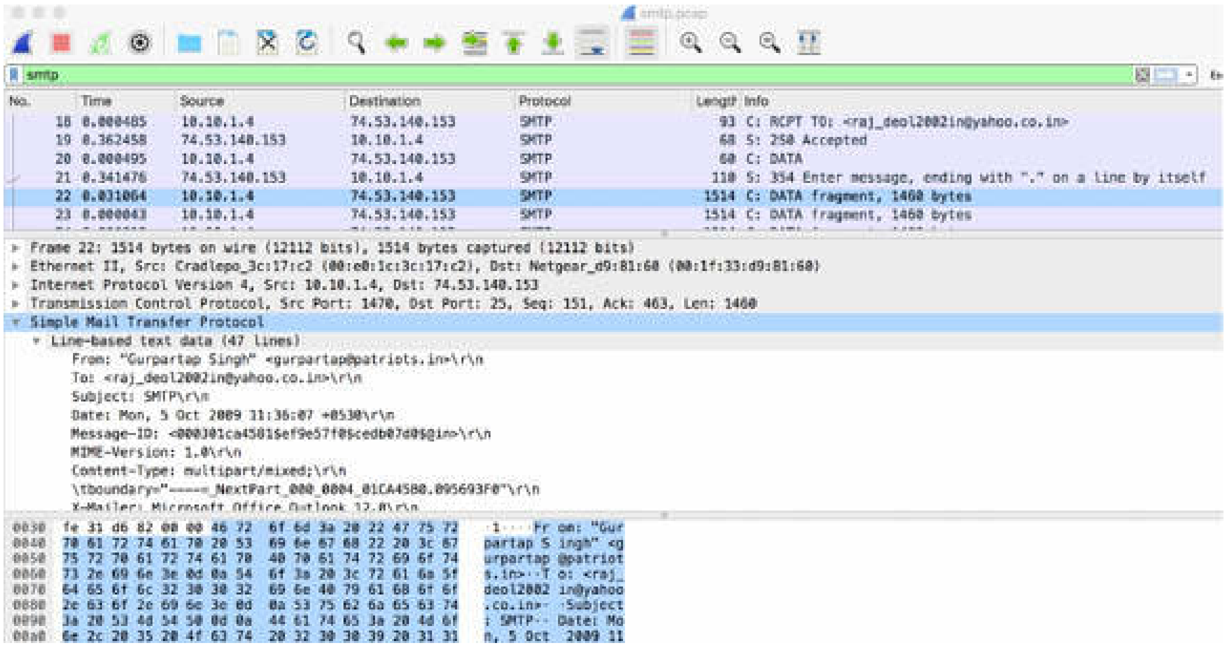
In the Packet Details pane, select the Transmission Control Protocol field. In the Packet Bytes pane, the related bytes are highlighted, as shown in the figure below.



In the figure below, the SMTP part is highlighted inside the EHLO command packet. In this packet, the EHLO command is followed by a request parameter containing the name of the host sending the email.



In the Packet List pane, select packet #22. In the Packet Details pane, open the tree view to see the details. The DATA message, highlighted in the figure below, initiates the email data transfer.



The following list describes the most common SMTP client commands:

- HELO: Initiates an SMTP session
- EHLO: Initiates an SMTP session from a sender that supports SMTP mail service extensions
- MAIL: Initiates mail transfer
- RCPT: Identifies mail recipient
- DATA: Initiates mail data transfer
- VRFY: Verifies recipient exists
- RSET: Aborts mail transaction
- NOOP: Tests connection to the server
- QUIT: Closes SMTP connection
- EXPN: Expands mailing list
- HELP: Lists help information

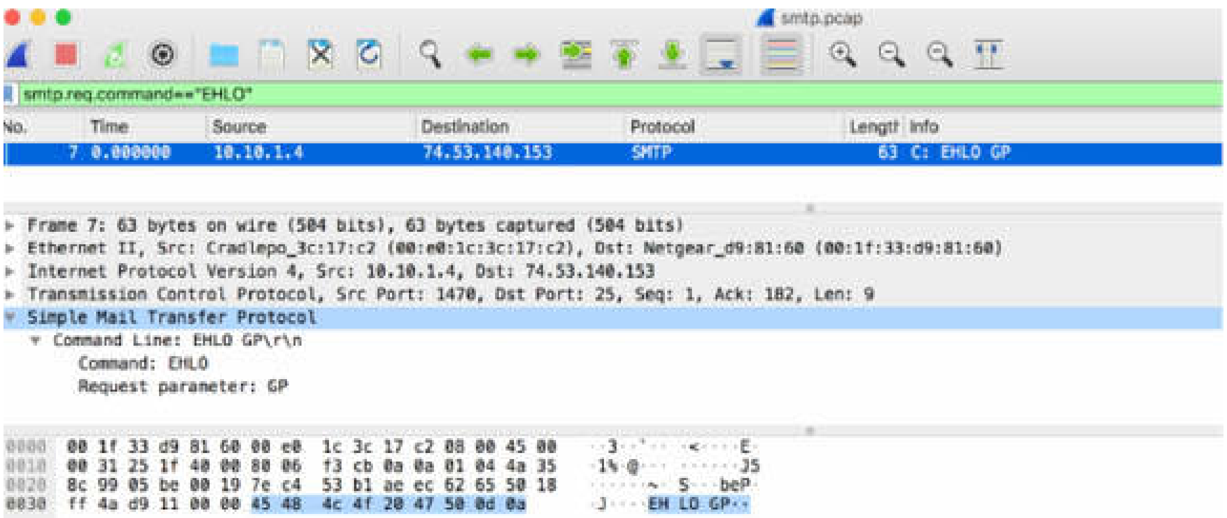
The following list describes the most common SMTP reply codes sent from the SMTP server:

- Code 211: System status
- Code 214: Help message
- Code 220: <domain> service ready
- Code 221: <domain> service closing channel
- Code 250: Requested action okay and completed
- Code 251: User not local; will forward to <path>
- Code 354: Start mail input
- Code 421: <domain> service not available
- Code 450: Mailbox unavailable
- Code 451: Local error
- Code 452: Insufficient storage
- Code 500: Syntax error, command unrecognized
- Code 501: Syntax error in parameters or arguments
- Code 502: Command not implemented
- Code 503: Bad sequence of commands
- Code 504: Command parameter not implemented
- Code 521: <domain> does not accept mail (see rfc1846)
- Code 550: Mailbox unavailable
- Code 551: User not local, please try <path>
- Code 552: Exceeded storage allocation
- Code 553: Mailbox name not allowed
- Code 554: Transaction failed

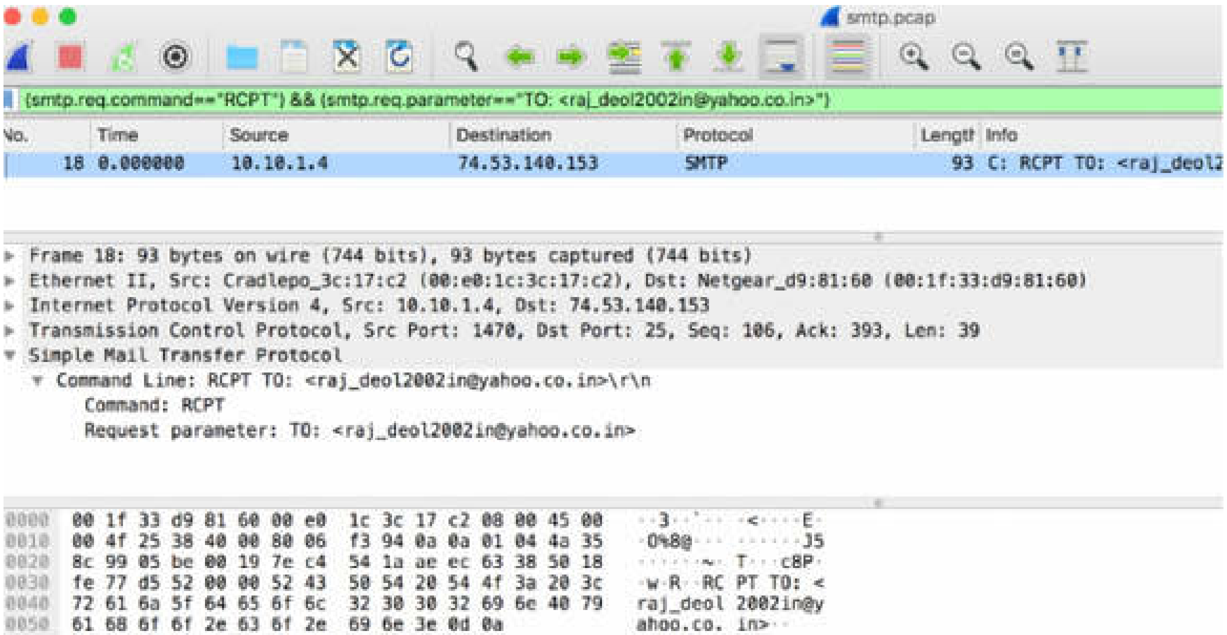
Task 2:
As shown in the previous task, the smtp display filter is used for SMTP. For a capture filter, use tcp.port == 25 because, by default, SMTP communication uses TCP port 25. If the SMTP traffic runs on a different port, change the capture filter accordingly.

To display only specific messages belonging to SMTP communication in the Packet List pane, use one of the following filters.

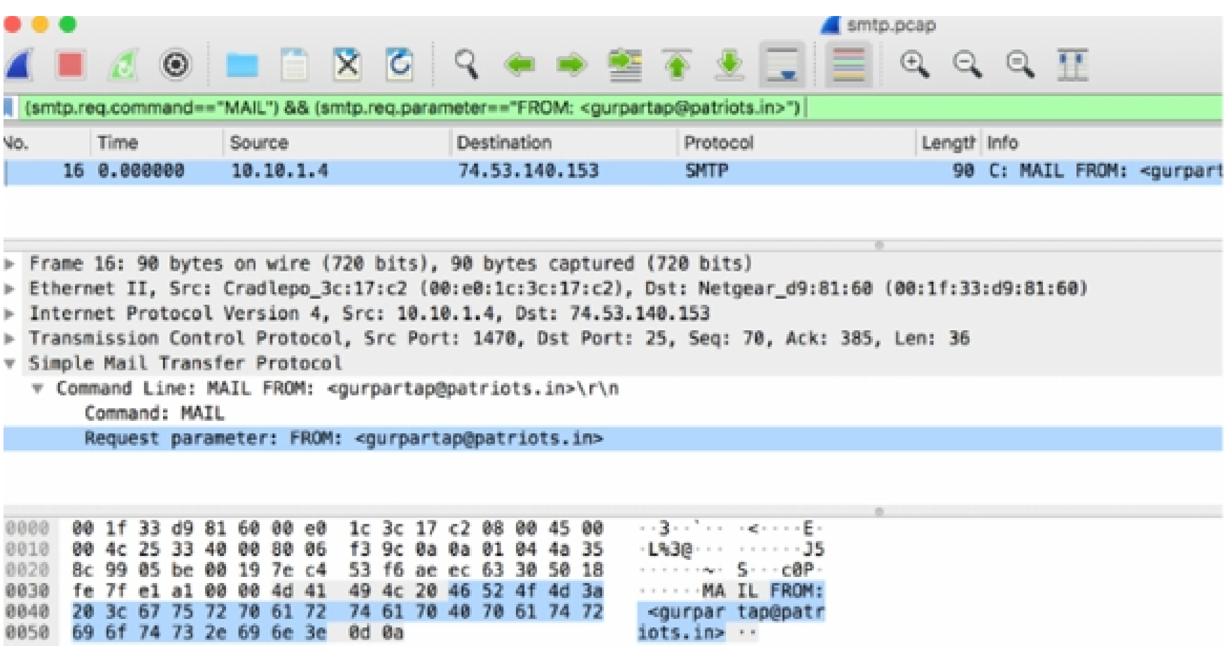
To display only those packets in which the client and the server show that they support mail extensions and are setting up an SMTP communication, in the filter toolbar, enter smtp.req.command==”EHLO” . Only one result is displayed in the Packet List pane, as shown in the figure below.



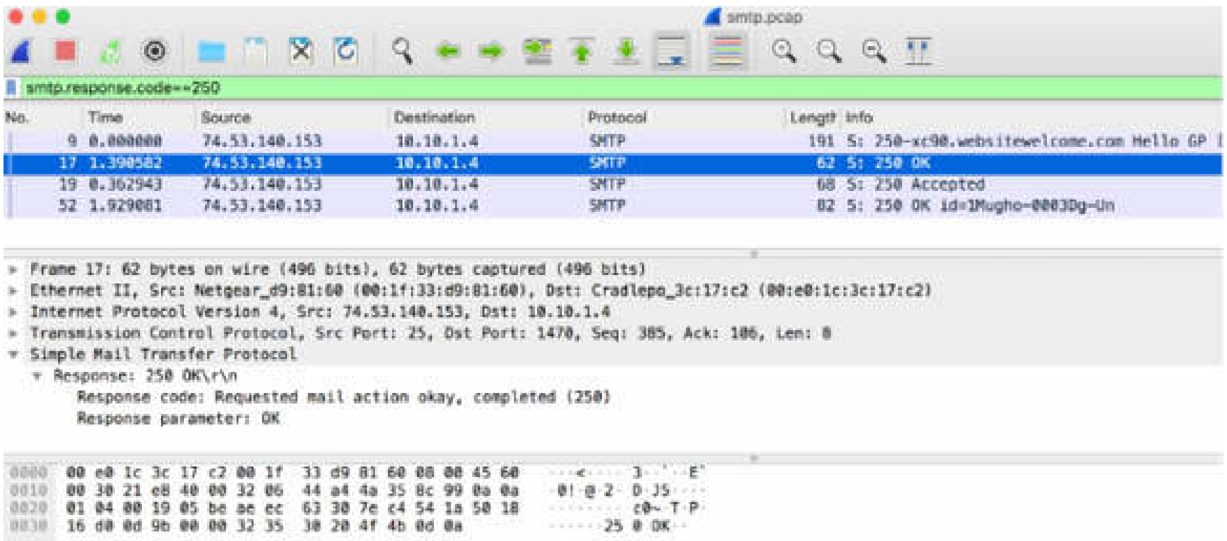
To display the SMTP packets where the recipient of the email is identified and the recipient’s email address is <raj_deol2002in@yahoo.co.in>, in the filter toolbar, enter (smtp.req.command==”RCPT”) && (smtp.req.parameter==”TO: <raj_deol2002in@yahoo.co.in>”) . Only one result is displayed in the Packet List pane, as shown in the figure below.



To display the SMTP packets where the sender of the email is identified and the sender's email address is <gurpartap@patriots.in>, in the filter toolbar, enter (smtp.req.command==”MAIL”) && (smtp.req.parameter==”FROM: <gurpartap@patriots.in>”) . Only one result is displayed in the Packet List pane, as shown in the figure below.



To display the SMTP packets where the response code of the server signals that the requested action is okay and completed, in the filter toolbar, enter smtp.response.code==250 . The results are displayed in the Packet List pane, as shown in the figure below.



Notes:

Repeat the previous steps to analyze a different SMTP capture and identify the specific fields of SMTP. To gain confidence in filtering only specific packets of interest, apply some of the filters explained in this lab.

Wireless Networking and Voice

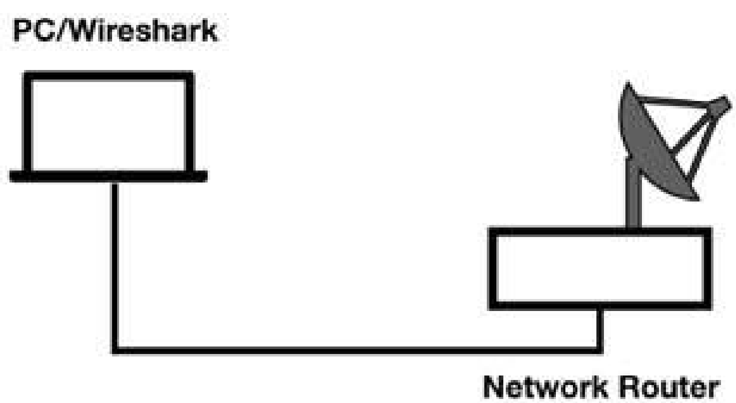
Lab 76. WLAN Capturing Modes and Decrypting

Lab Objective:
Learn how to capture on WLAN networks.

Lab Purpose:
Understand the process and modalities to capture on WLAN networks.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:

When analyzing WLAN traffic in Wireshark, it is important to capture traffic as close as possible to the users having problems with the connection to directly see the issues. If possible, it is better to capture traffic close to the access point. Some of the typical issues that can occur are low signal strength, retransmissions, problems locating the access point, access point "disappearance," problems with the authentication process, etc.

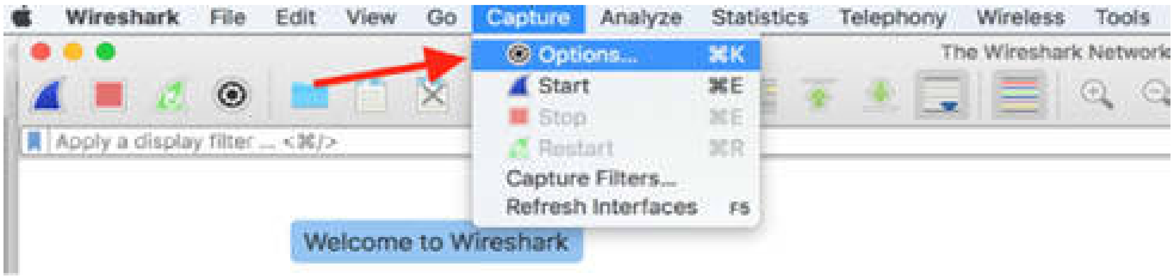
There are two modalities for capturing traffic with Wireshark through a WLAN adapter: promiscuous mode and monitor mode.

In promiscuous mode, an 802.11 adapter only captures packets of the SSID the adapter has joined. To capture all traffic that the adapter can receive, the adapter must be put into monitor mode, sometimes called rfmon mode. When using monitor mode, the driver does not make the adapter a member of any service set on the network. In monitor mode, all packets of all SSIDs from the currently-selected channel are captured.

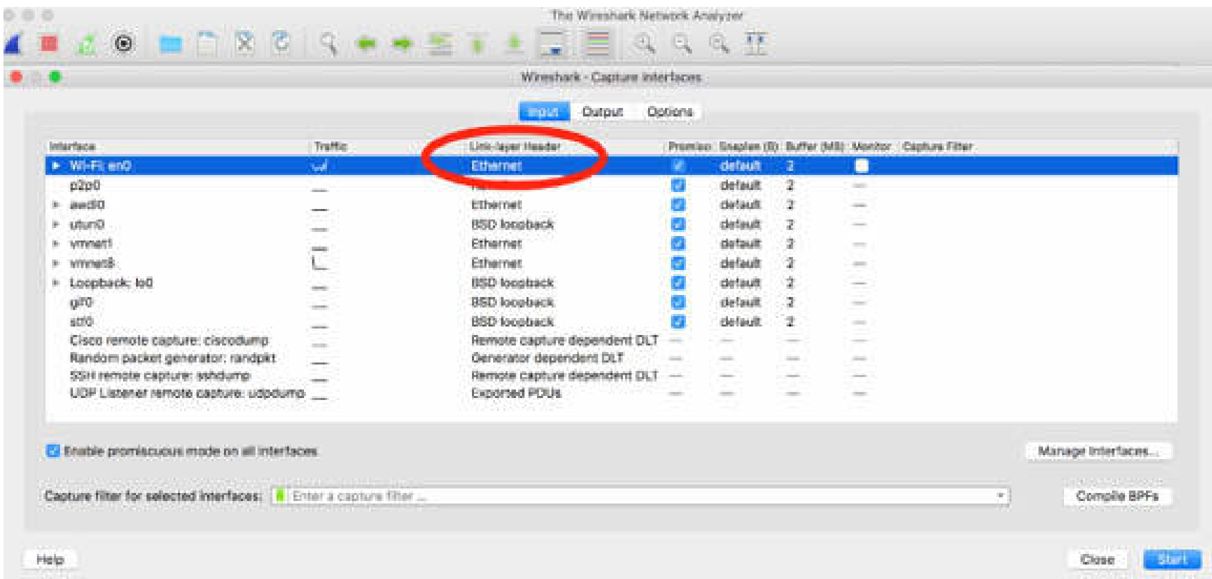
In monitor mode, the adapter doesn't support general network communications (web browsing, email, etc.) because the adapter is not part of any service set. The driver only supplies received packets to a packet capture mechanism, not to the network stack.

Monitor mode is not supported by WinPcap—this limits the WLAN analysis capabilities of Wireshark and TShark on Windows machines. It is supported, for at least some network interface cards, on some versions of Linux, FreeBSD, NetBSD, OpenBSD, and Mac OS X. No additional cards or drivers may be needed for WLAN analysis when using these operating systems.

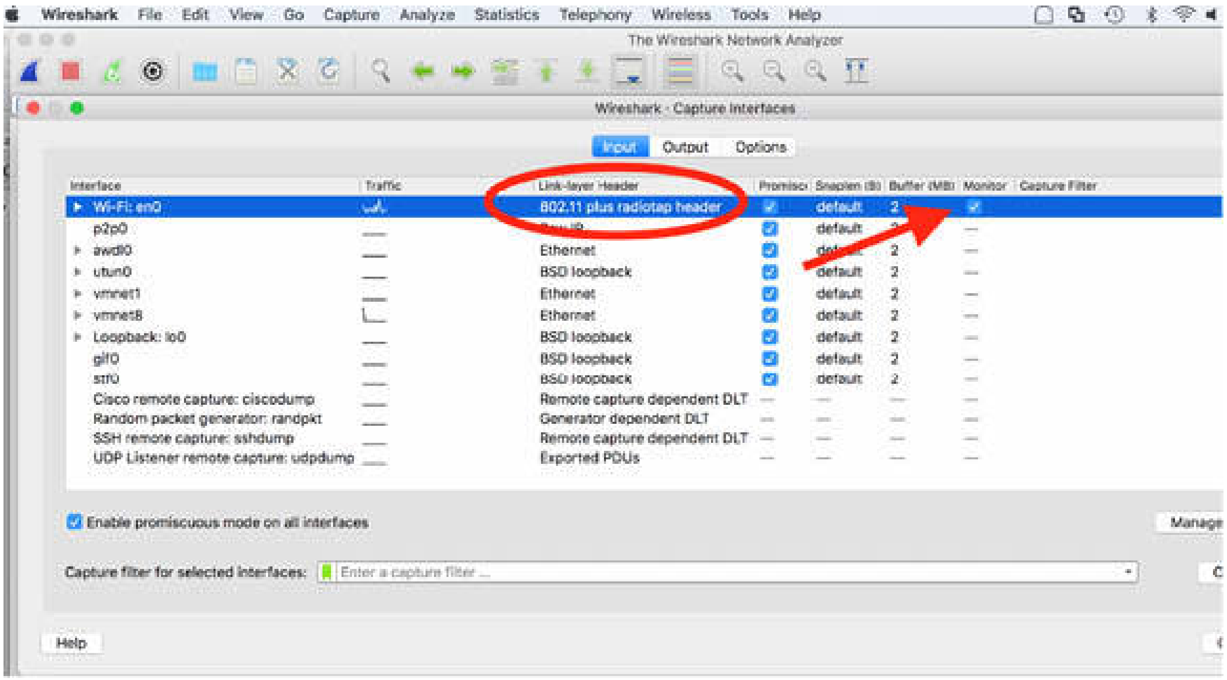
To test if your network card supports monitor mode, open Wireshark and on the main menu, select Capture > Options.



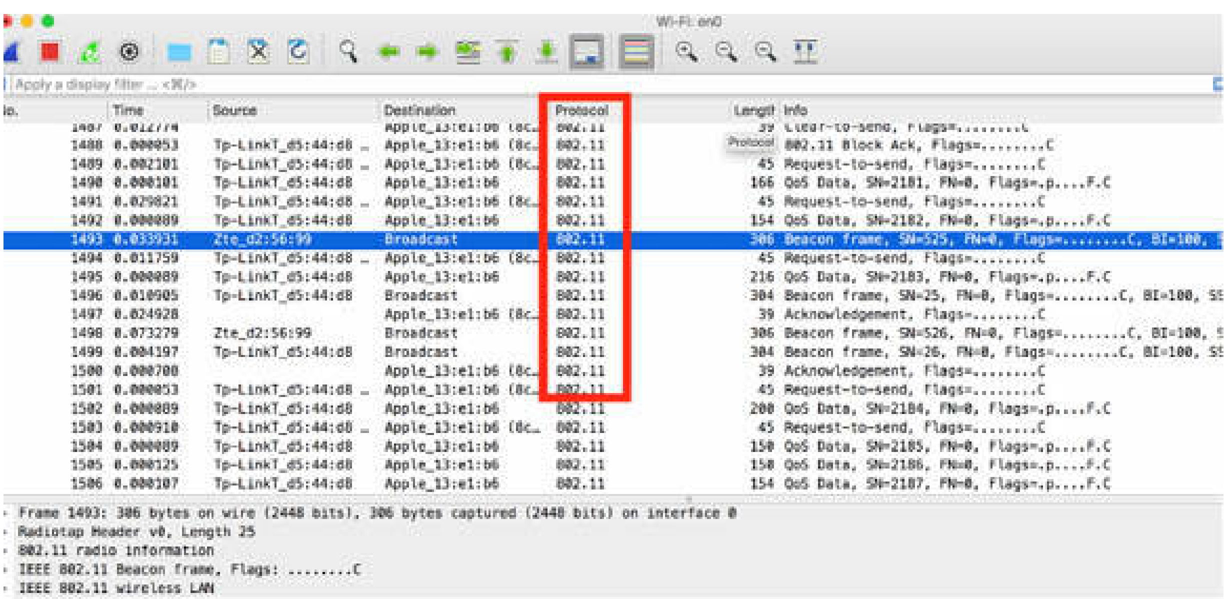
The Capture Options dialog box is displayed, showing all network interfaces available. The Wi-Fi network card is displayed with the header Ethernet, as shown in the figure below.



In the Wi-Fi network card row, select the check box in the Monitor column, as shown in the figure below, and click Start to begin the network capture.



The Packet List pane shows the list of capture packets, as shown in the figure below. See, in particular, the Protocol column, all the data available, and the management and control frames.



There are four possible combinations of promiscuous mode and monitor mode configurations. The following list describes the capabilities and possible issues for each combination:

- Promiscuous Mode On/Monitor Mode Off Capture Capabilities: Fake Ethernet header prepended to packet; no Management or Control packets captured
- Promiscuous Mode Off/Monitor Mode Off Capture Capabilities: Fake Ethernet header prepended to packet; no Management or Control packets captured Issues to Consider: Need to capture traffic on the host you're interested in
- Promiscuous Mode Off/Monitor Mode On Capture Capabilities: 802.11 header; Management and Control packets captured Issues to Consider: Need to capture traffic on the host you're interested in
- Promiscuous Mode On/Monitor Mode On Capture Capabilities: 802.11 header; Management and Control packets captured Issues to Consider: Great. Can capture traffic on various channels and from all SSIDs

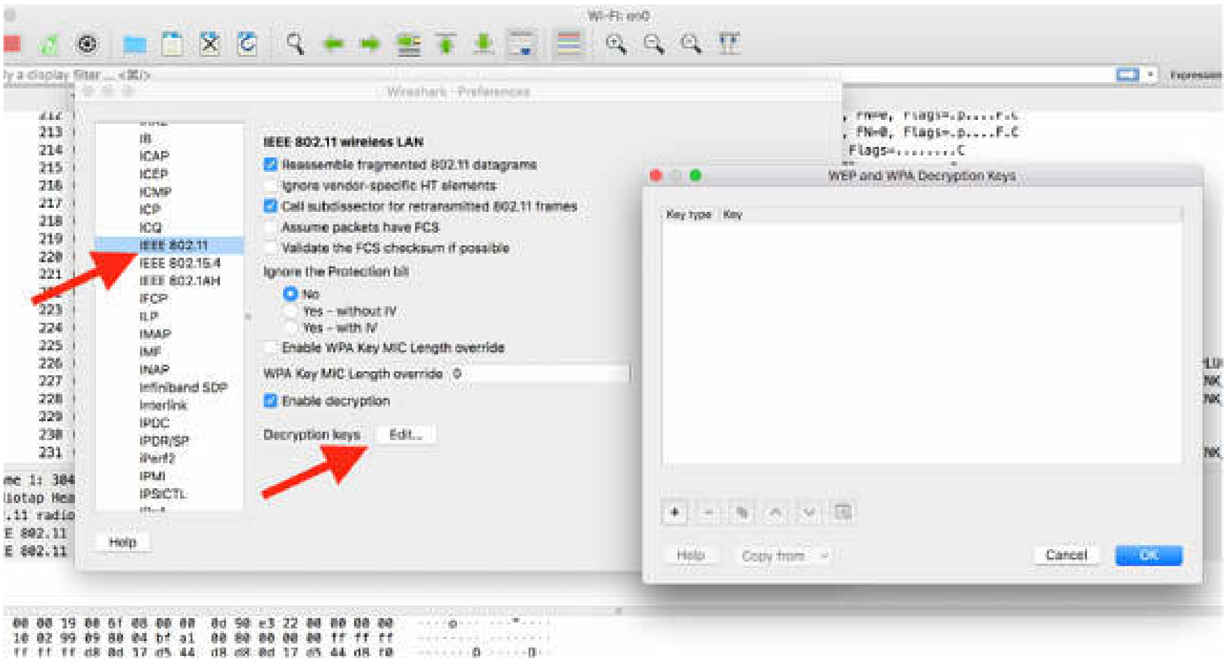
Task 2:
To decrypt WLAN traffic, it is mandatory to have the decryption key available.

Decryption keys can be input using the Decryption Mode and Decryption Key Management on the Wireless Toolbar or in the IEEE 802.11 Preferences setting.

If you are using the Wireless Toolbar (available only when you are using an AirPcap adapter), you can choose from three decryption modes: none (no decryption), Wireshark (decryption done by Wireshark), and Driver (decryption done by the AirPcap driver).

If you are decrypting by using the IEEE 802.11 Preferences setting, in Wireshark, on the main menu, select Edit > Preferences. In the left tree view, select IEEE 802.11, and click Edit, as shown in the figure below. The

WEP and WPA Decryption Keys dialog box is displayed where you can enter the keys.



Wireshark can decrypt WEP, WPA, and WPA2 traffic. When decrypting WPA traffic, you must capture the four EAPOL (Extensible Authorization Protocol) four-way handshake packets. .

Notes:

Repeat the previous steps to identify the Wi-Fi card and test its capabilities on different machine types. Test different combinations of monitor and promiscuous modes to have a complete and clear picture of driver capabilities and gain the necessary confidence in Wi-Fi capturing. Try to decrypt a Wi-FI network capture of your choice when you have the knowledge of the Crypt Key.

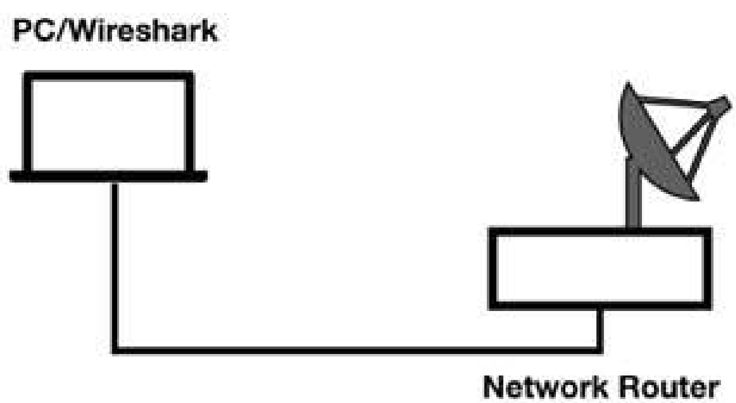
Lab 77. WLAN Header Settings

Lab Objective:
Learn how to use the WLAN header settings.

Lab Purpose:
Understand the WLAN header settings and their use.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless connection to a network router that has access to the internet.

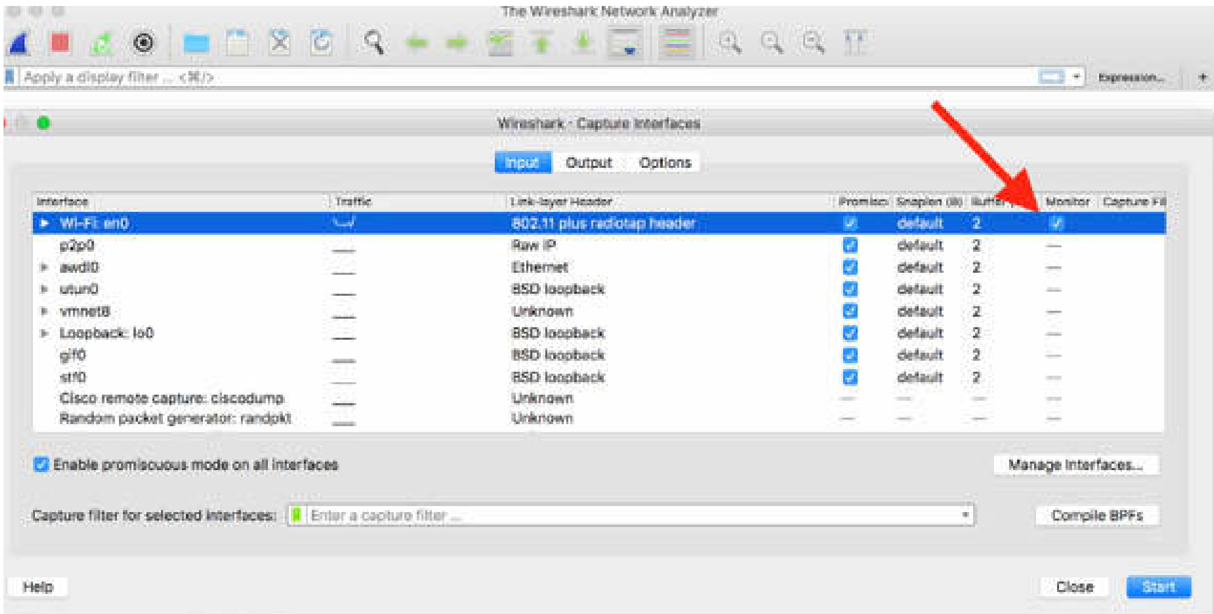


Lab Walkthrough:

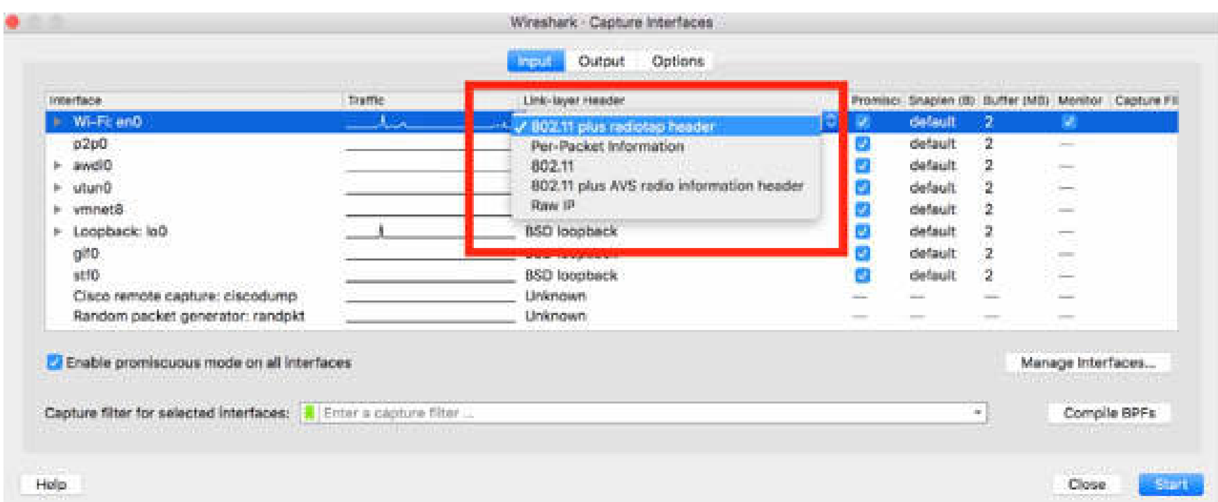
Task 1:
Open Wireshark, and on the main menu, select Capture > Options. The Capture Options dialog box is displayed, showing all network interfaces

available.

In the Wi-Fi interface row, select the check box in the Monitor column, as shown in the figure below.



In the “Link-layer Header” column, select a header to be applied to the packets, as shown in the figure below.

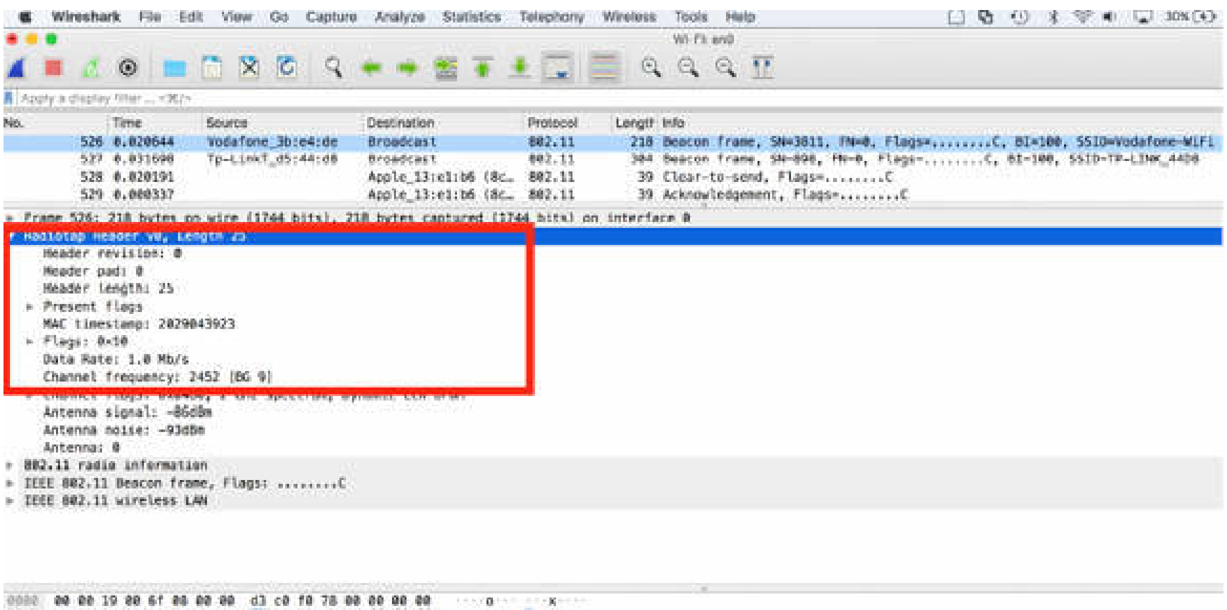


The following five choices are available in WLAN settings for headers:

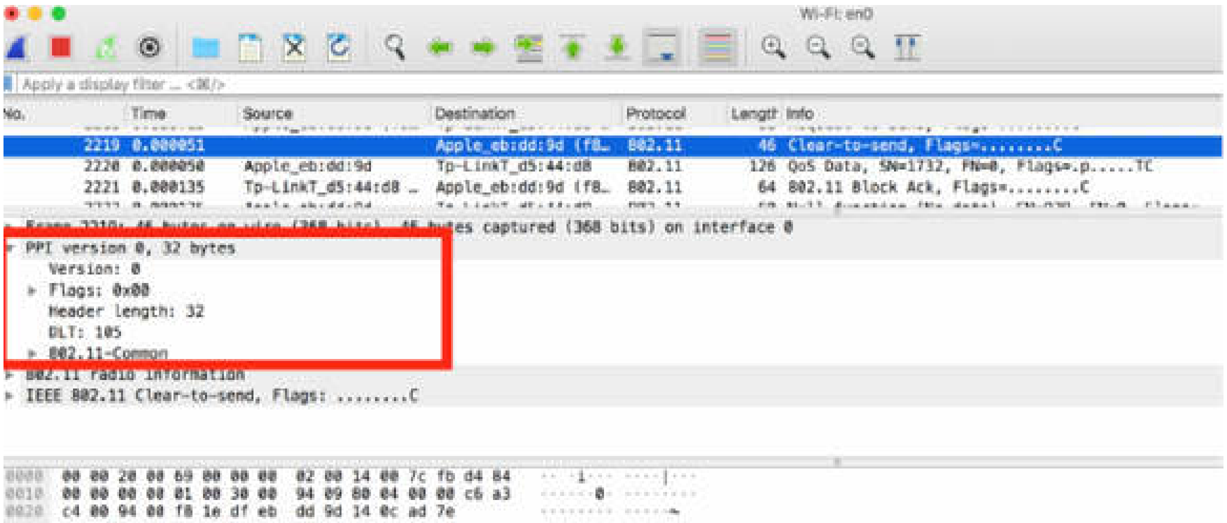
- 802.11 plus radiotap header
- Per-Packet information (PPI)
- 802.11
- 802.11 plus AVS radio information header
- Raw IP

The radiotap and PPI pseudo headers provide more information about the frames that exist only in the 802.11 header.

Select “802.11 plus radiotap header” and start capturing packets. The resultant window will look as shown in the figure below.



Repeat the same steps by selecting the PPI Header. Alternatively, you can download the sample capture file mesh.pcap from **https://wiki.wireshark.org/SampleCaptures** . The resultant window will look as shown in the figure below.

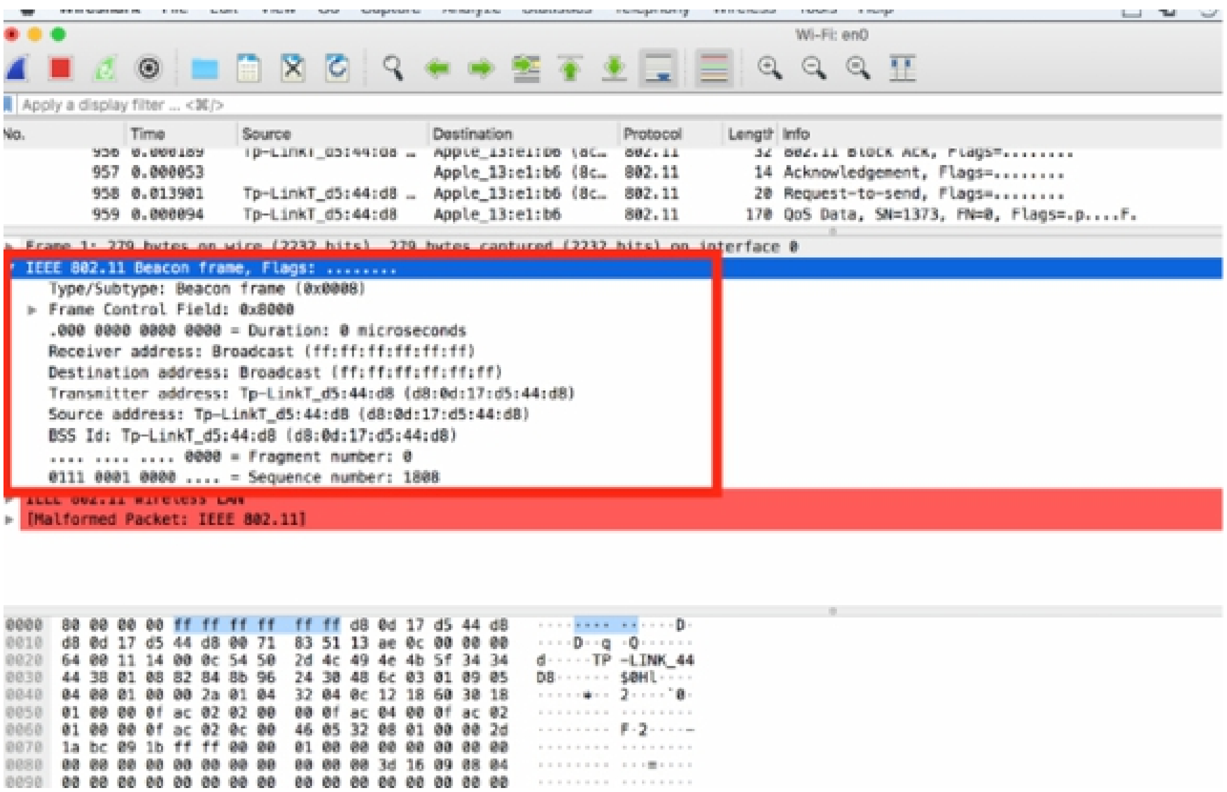


From the previous two figures, it is clear that the radiotap and PPI headers supply additional information about the frames captured. The radiotap header provides this information from the AirPcap or libpcap driver to Wireshark, because it is not in the WLAN packet when it arrives, it is added by the local capture adapter.

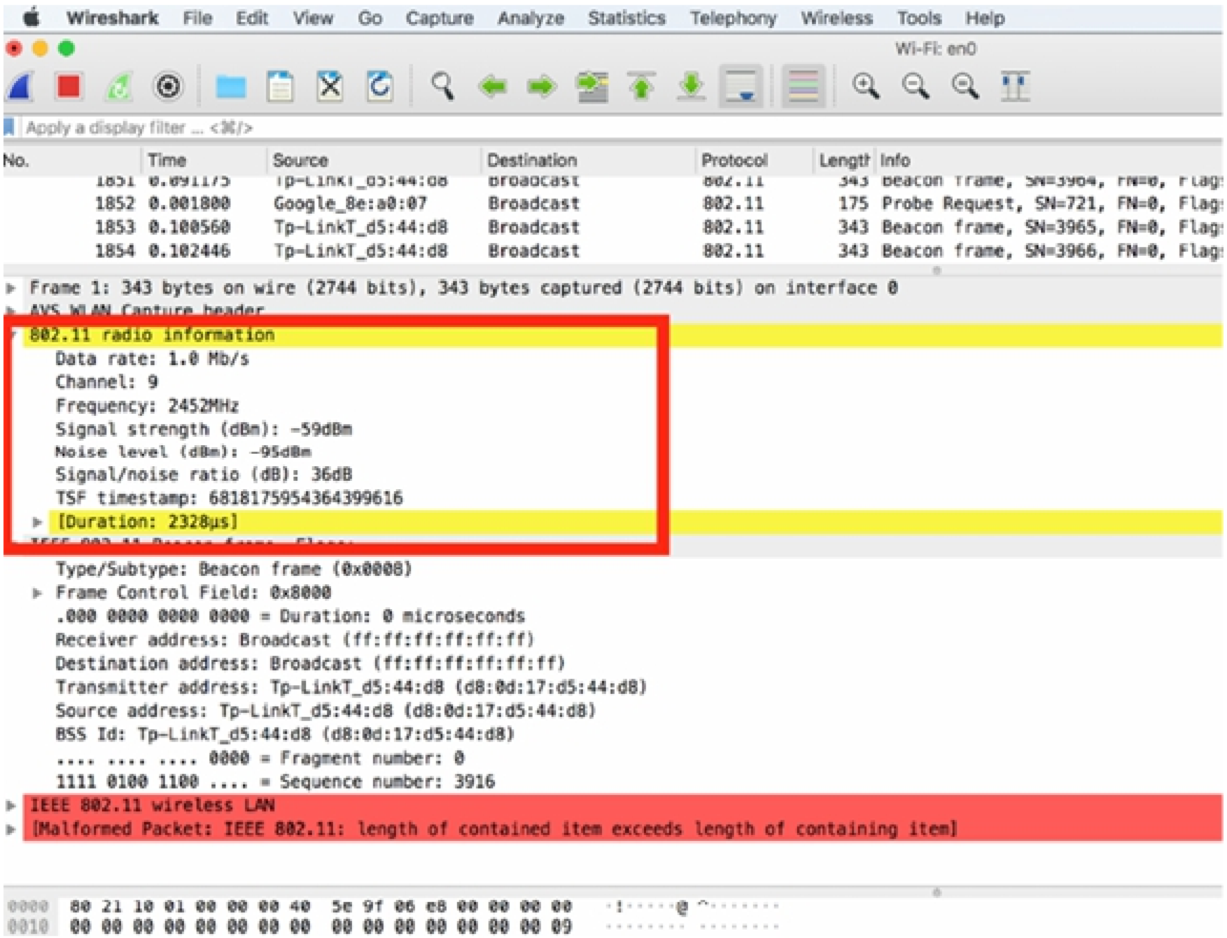
PPI header is a general and extensible meta-information header format. It was originally developed, in 2007, to provide 802.11n radio information, but it can handle other information as well.

Task 2:

To understand the difference in different packet headers, in the “Link-layer Header” column of the Capture Options dialog box, select the 802.11 option and start a WLAN capture. The resultant window will look as shown in the figure below, where you can detect all fields belonging to the selected header format.



Next, select the "802.11 plus radio information" option. As shown in the figure below, the additional radio information—related to the channel, frequency, noise level, and signal strength—is added before the 802.11 header.



To filter the 802.11 frequency/channel information, apply the radiotap or PPI header. This information is not sent with a packet as a field value. When the adapter receives the WLAN packet, the frequency that the packet was captured on is used to define which channel the packet was on. This frequency/channel information is shown in the radiotap and PPI headers.

Both radiotap and PPI headers include information about the signal strength. The signal strength value is based on the signal strength at the location and the time at which the packet was received. The radiotap header and PPI headers also contain channel/frequency values.

The signal strength indicator value defines the power but not the quality of the signal. The value is defined in dBm (power ratio in decibels referenced to one milliwatt). From 0 to —65 dBm is considered excellent to acceptable signal strength whereas the signal strength becomes an issue as it moves lower (closer to —100 dBm). Problems will likely occur when the signal strength goes below –80 dBm. The signal strength issues between the WLAN hosts and access points may lead to retransmissions and eventually loss of connectivity.

The signal-to-noise ratio defines the difference between the signal and noise values. Higher ratio numbers indicate less noise obstruction. If this value reaches as low as < 15 dB, performance is degraded.

Notes:
Repeat the previous steps to better understand the differences in the header types for WLAN packets and to identify all the relevant fields for each selected header.

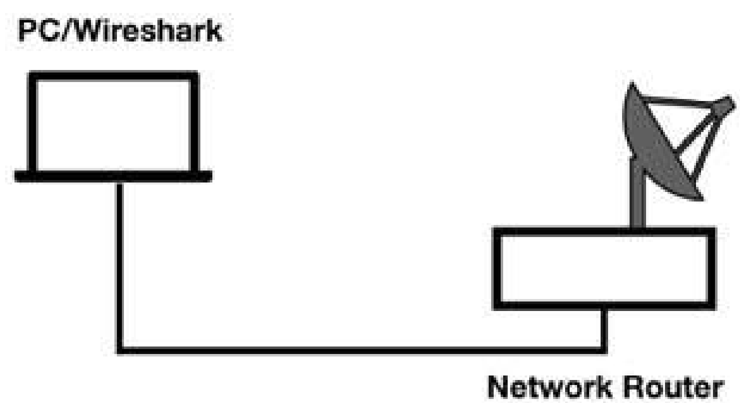
Lab 78. 802.11 Traffic Basics

Lab Objective:
Learn how to use the WLAN Header Settings.

Lab Purpose:
Understand the WLAN Header Settings and why they are used.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless connection to a network router that has access to the internet.

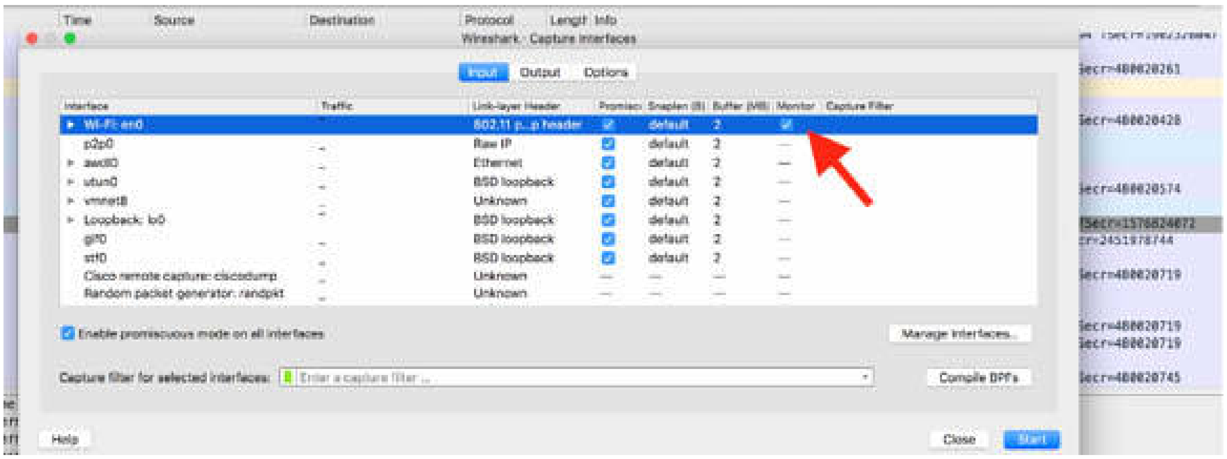


Lab Walkthrough:

Task 1:
Open Wireshark, and on the main menu, select Capture > Options. The Capture Options dialog box is displayed, showing all network interfaces

available. In the Wi-Fi interface row, select the check box in the Monitor column, as shown in the figure below.

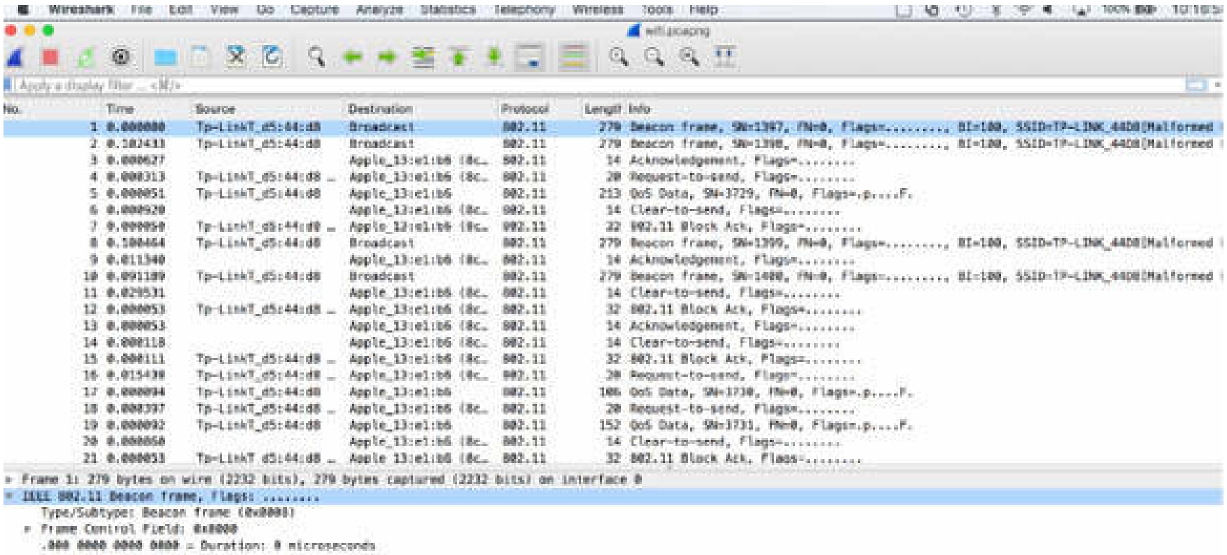
Capture the traffic for a few minutes. Stop the capture and save the file.



The results in the Packet List pane will be similar to the one displayed in the figure below. There are three types of 802.11 frames seen on WLANs:

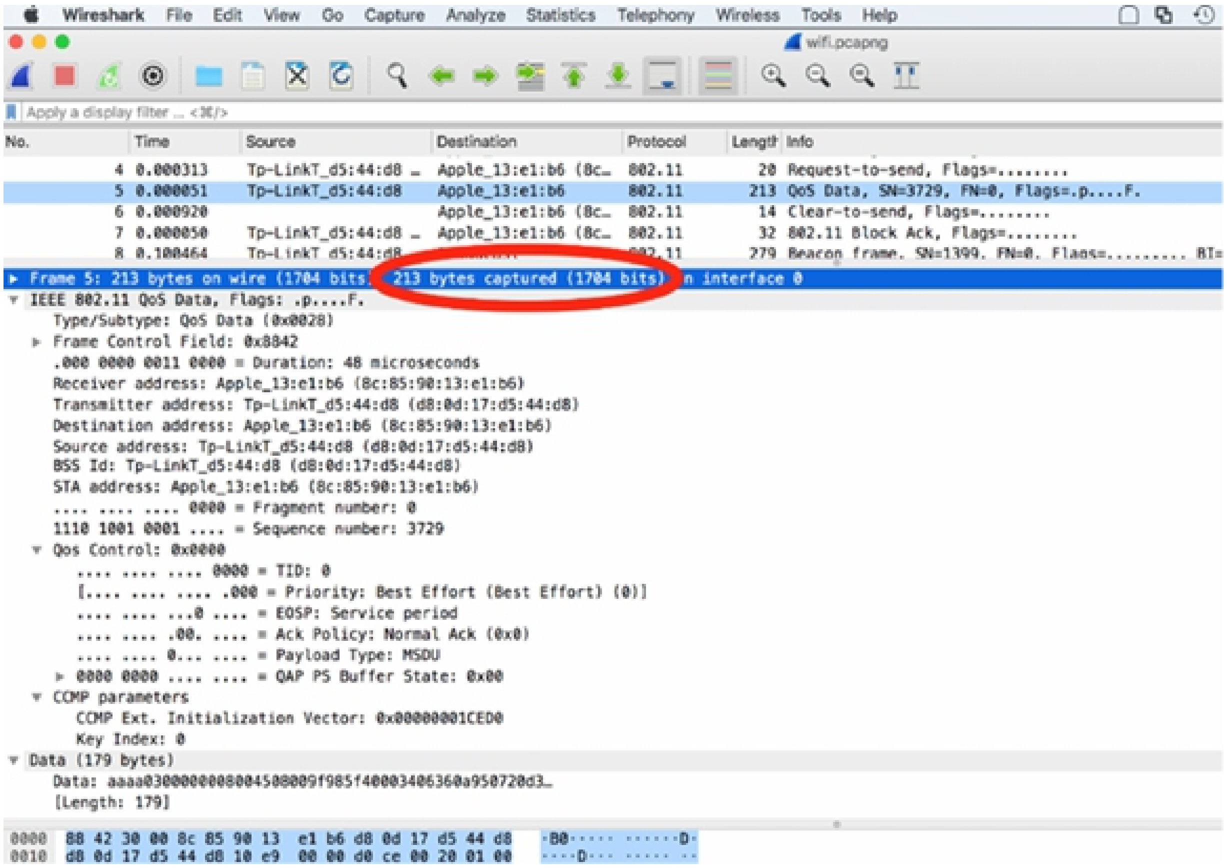
- Data: Contains data of some sort.
- Management: Used to establish MAC-layer connectivity; Association Request/Responses, Probe Requests/Responses, and Beacons are examples of management frames.
- Control: Used to enable the delivery of data and management frames; Request-to-Send (RTS), Clear-to-Send (CTS), and ACKs are control frames.

The management and control frames are used to enable the basic 802.11 processes. Data frames are quite simply used to transfer data across the WLAN.

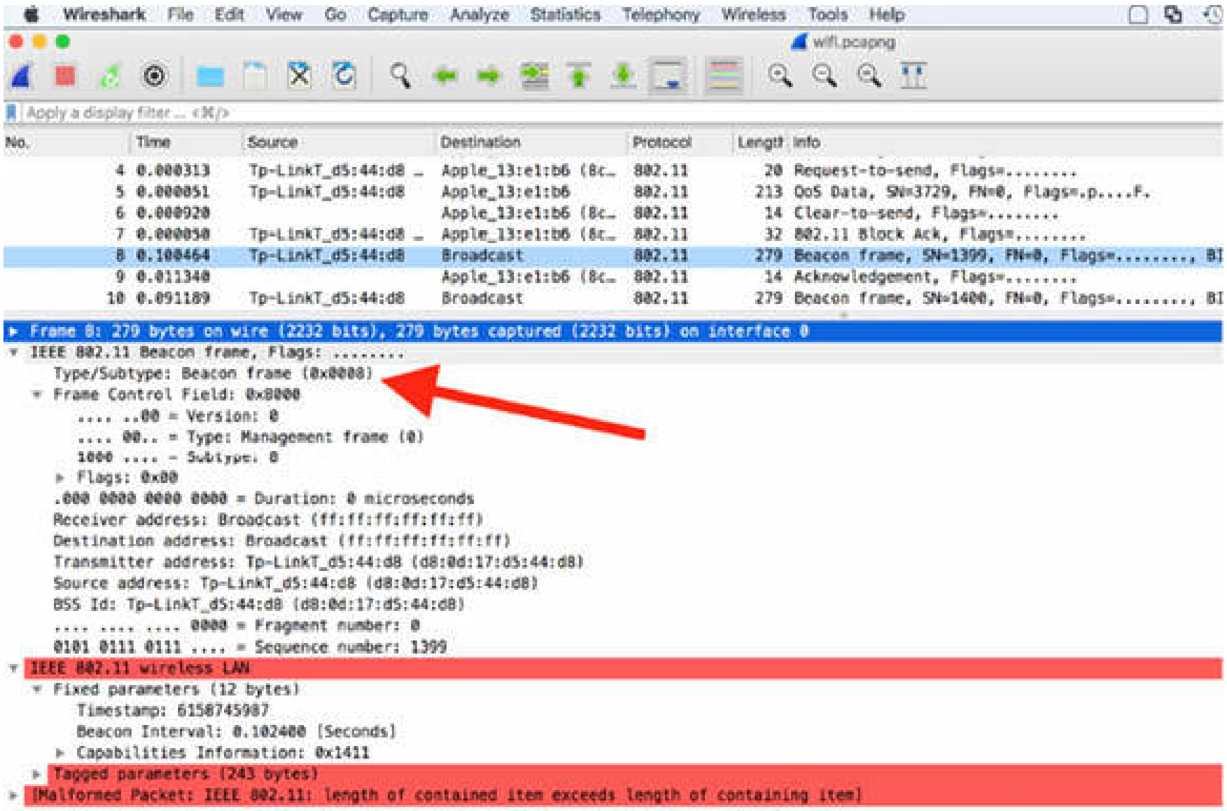


In the Packet List pane, select packet #5. In the Packet Details pane, open the tree view to see the packet details. As shown in the figure below, you can verify that this is a data frame. Data frames are the only WLAN frame types that can be forwarded to the wired network.

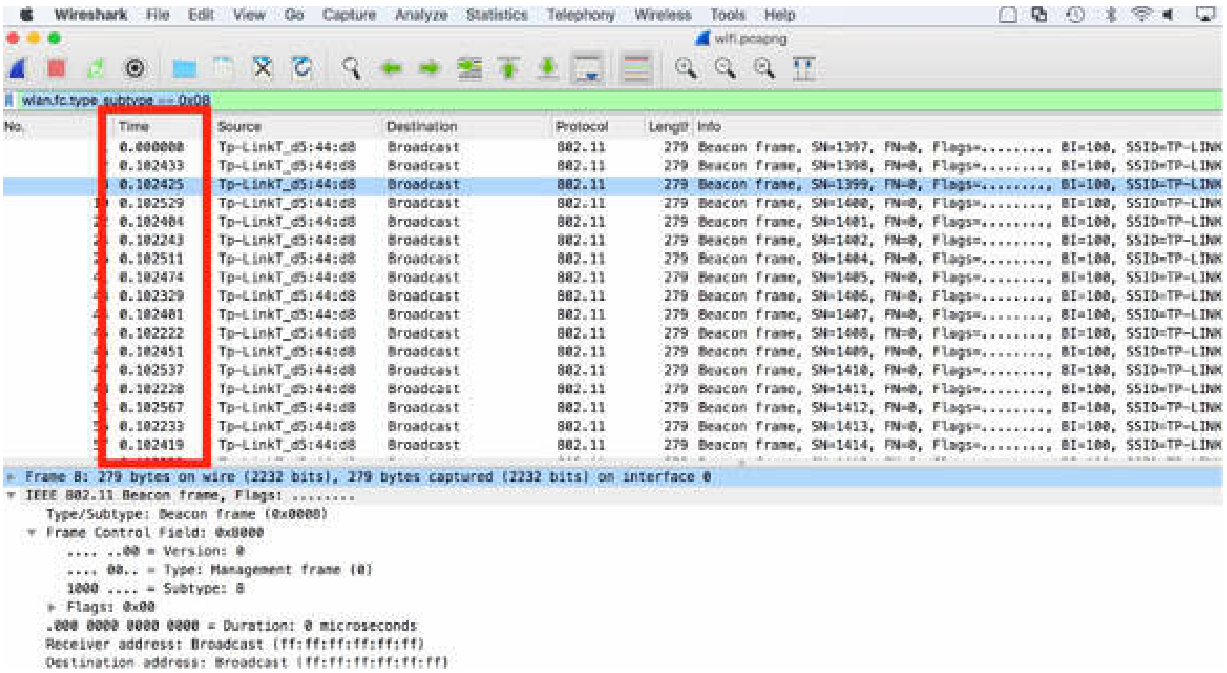
Although the IEEE 802.11 specifications state that the MAC Service Data Unit (MSDU) can be up to 2304 bytes, you will probably see smaller data frames because these frames are bridged to an Ethernet network (in this example, the length of the data frame is equal to 213 bytes). For example, if an STA makes a connection to an HTTP server on the wired network, the Maximum Segment Size (MSS) is negotiated during the TCP handshake process. This is the size of the TCP segment that will be prepended by the TCP and IP headers and encapsulated in an 802.11 header.



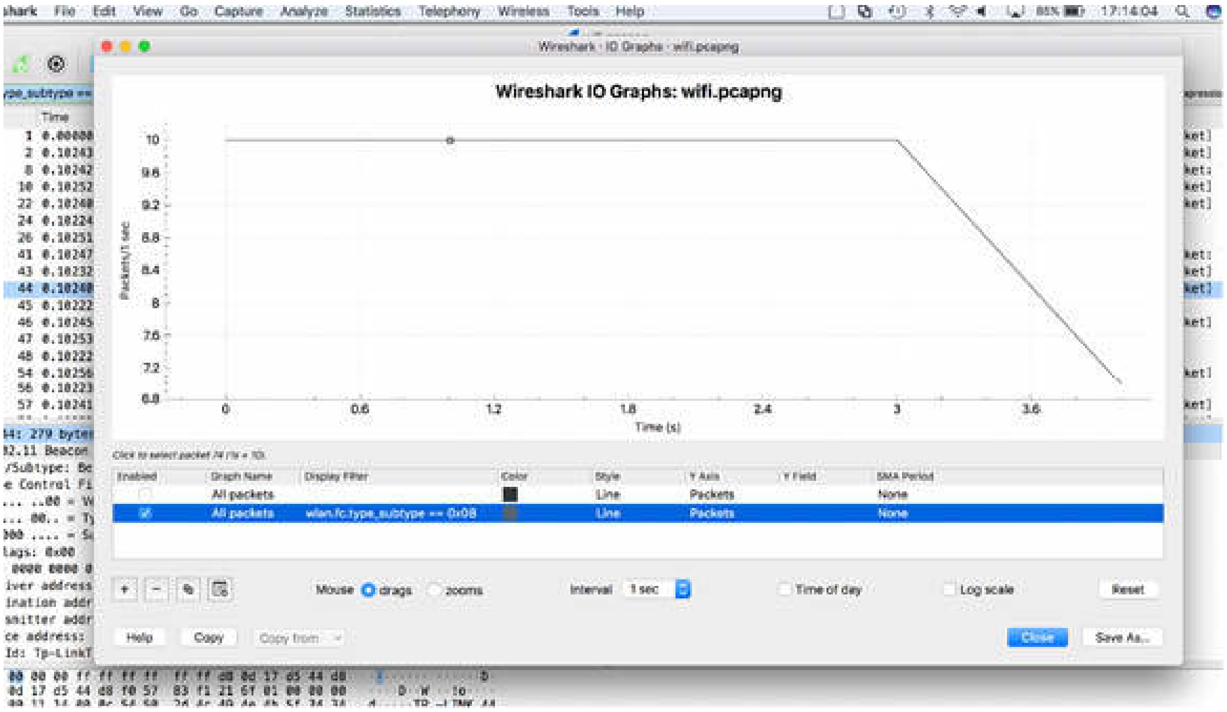
In the Packet List pane, select packet #8. In the Packet Details pane, open the tree view to see the packet details. As shown in the figure below, you can verify that this is a management frame—specifically, a beacon packet.



A beacon packet is a periodical packet that is sent by the access point (AP) on the network to provide information and to notify the AP presence in the network. In the filter toolbar, enter wlan.fc.type_subtype == 0x08 . The beacon packet cycle (in this case, equal to the default value, 100 ms) is displayed, as shown in the figure below.



Creating an I/O graph for beacon packets can be very useful because beacon packets are one of the most important management frames on the WLAN. In scenarios where users complain about the intermittent loss of connectivity to the WLAN, you can use an I/O graph—created by using a filter for beacon packets—to see if there is a problem. On the main menu, select Statistics > I/O Graph. In the display filter field of the I/O Graph dialog box, insert the display filter for the beacon packet, as shown in the figure below.



This graph shows that the period of beacon detection is quite constant until 3 seconds of capture after which either the capture is too short, or there is a problem where the access point stopped beaconing for a period.

The following list describes some of the most common 802.11 management frames:

- Authentication: STA sends to AP with identity.
- OpenSysAuth: AP sends the Authentication frame back indicating success or failure.
- SharedKey: AP sends the challenge text. NIC sends the encrypted version of the challenge text using the key. AP sends the Authentication frame indicating success or failure.
- Deauthentication: STA sends to terminate secure communications.
- Association: Used by AP to synchronize with the STA radio and define capabilities.
- Reassociation: Sent by STA to the new AP; triggers AP to get buffered data (if any) from the previous AP.
- Disassociation: Sent by STA to terminate an association with the AP.

- Beacon: Sent every 100 ms (default) by AP to announce its presence and provide info; STAs continuously scan for other APs.
- Probe: Request/Response. STA uses to obtain info from another STA; e.g., find APs in range (request/response).

The following list describes some of the most common 802.11 control frames:

- Request-to-Send: (optional) Used as a part of the two-way handshake to request transmission privileges.
- Clear-to-Send: (optional) The second part of the two-way handshake.
- ACK: Sent by the receiver to indicate that the data frame was received OK. No ACK would trigger an 802.11 retransmission by the sender.

Notes:

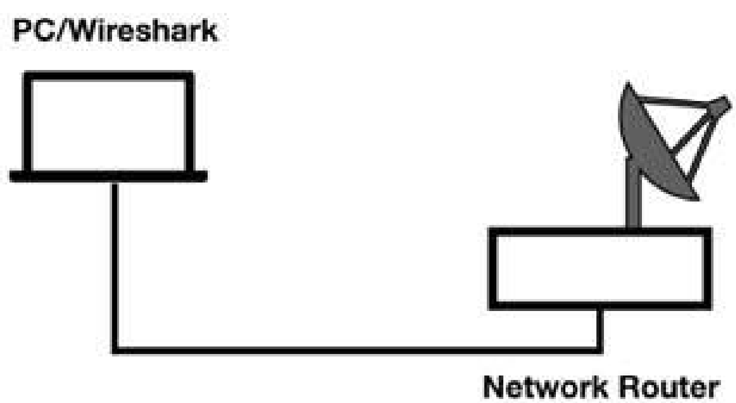
Lab 79. 802.11 Communications

Lab Objective:
Learn how to analyze 802.11 communications.

Lab Purpose:
Understand the main features of 802.11 communications.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

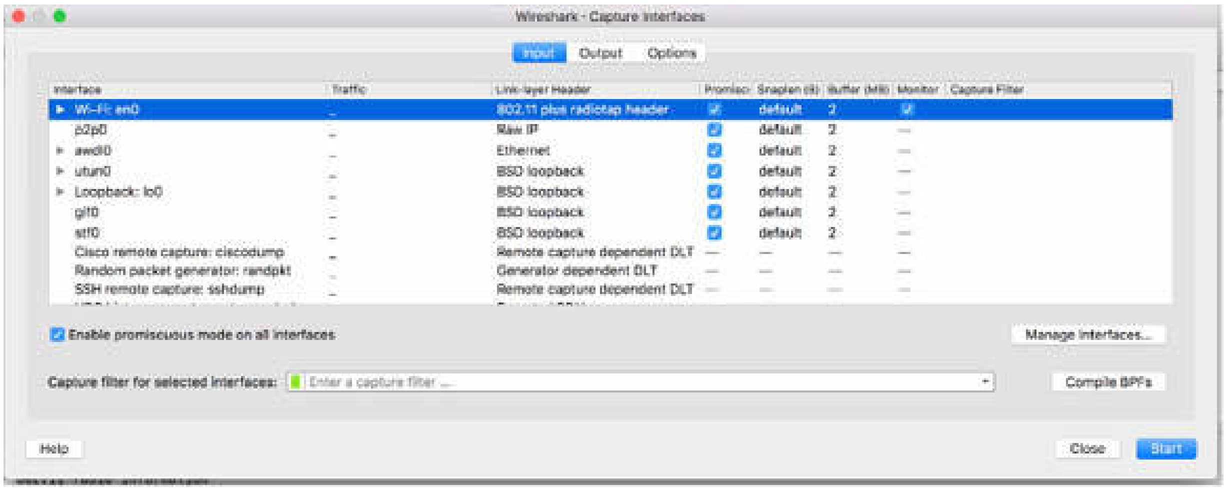
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
Open Wireshark, and on the main menu, select Capture > Options. The Capture Options dialog box is displayed, showing all network interfaces

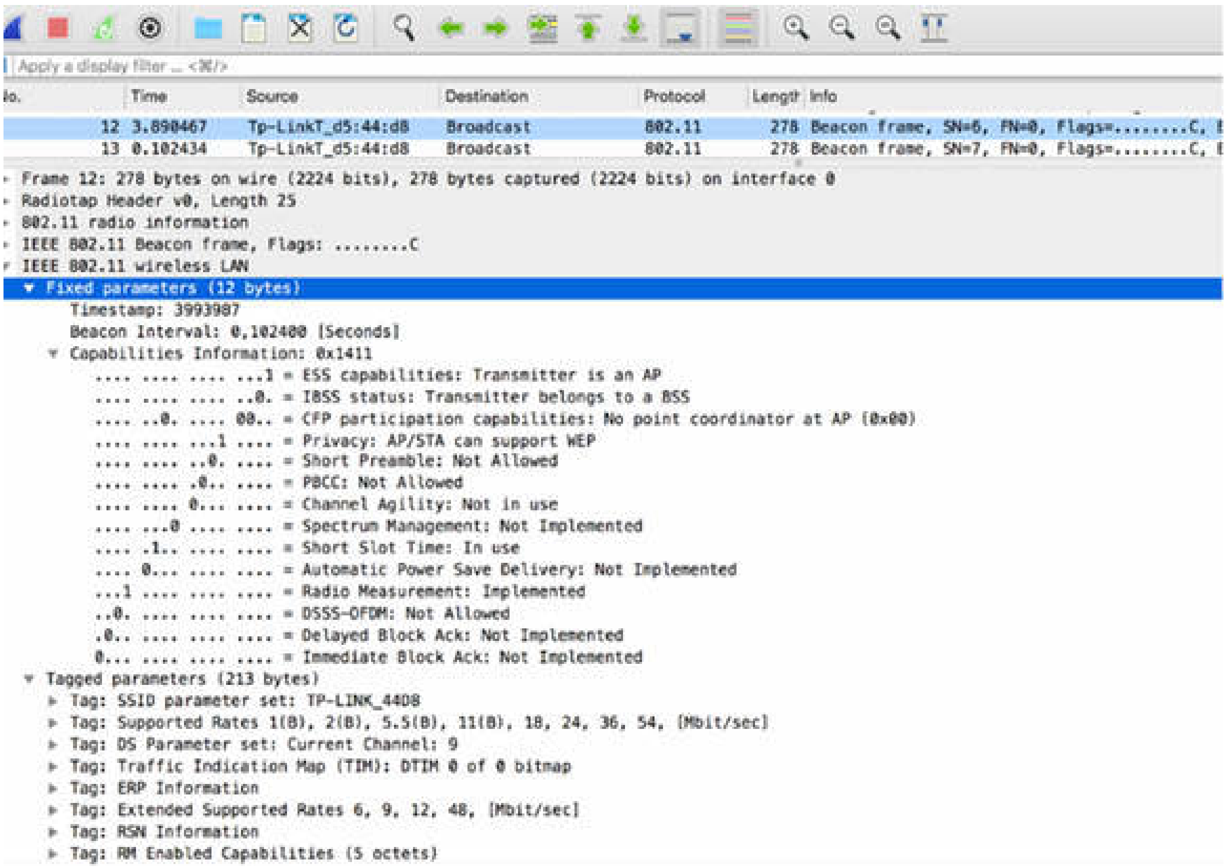
available. In the Wi-Fi interface row, select the check box in the Monitor column, as shown in the figure below. In the "Link-layer Header" column, select "802.11 plus radiotap header" and start capturing frames. Capture the traffic for a few minutes. Stop the capture and save the file.



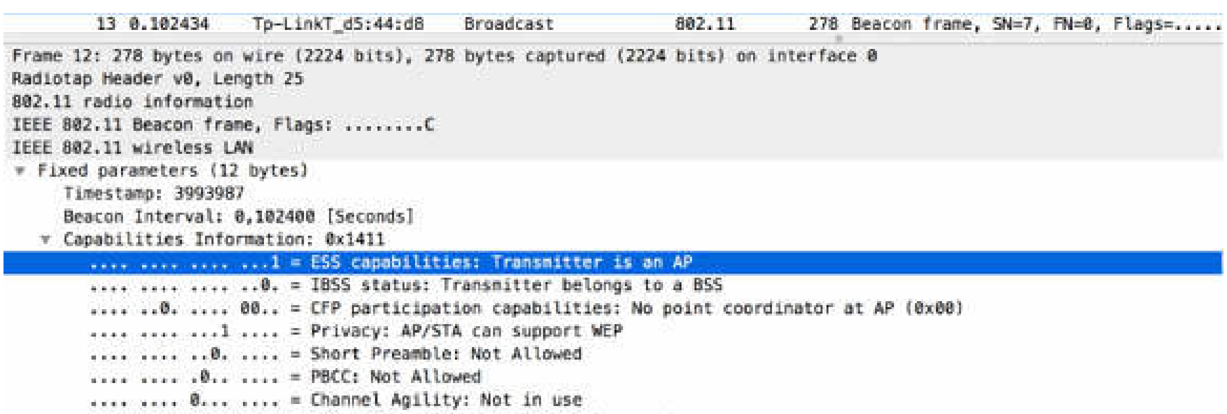
The process of connecting to a Wireless network requires that a station (the local PC) locates a wireless network, and authenticates and associates to the network. When a station wants to access a wireless network, it must connect with an access point that supports the desired SSID.

There are two modalities of access: passive mode or active mode. The station can either wait (passive mode) for a beacon frame from the access point or the station can send a probe request to find the access point (active mode). When an access point is configured not to broadcast an SSID, stations that are configured with an SSID value need to broadcast probe requests to the WLAN to find an access point with that SSID. By default, beacon frames are sent at approximately 100ms intervals.

In the Packet List pane, select a beacon frame sent to the destination "Broadcast" from the access point and inspect the details in the Packet Details pane. The result will be similar to the figure below.



In the figure above, packet #12 is a beacon frame with the “Capabilities Information” field expanded in the Packet Details pane. Based on the “ESS Capabilities” field shown in the figure below, it is clear that the transmitter is an access point (AP).

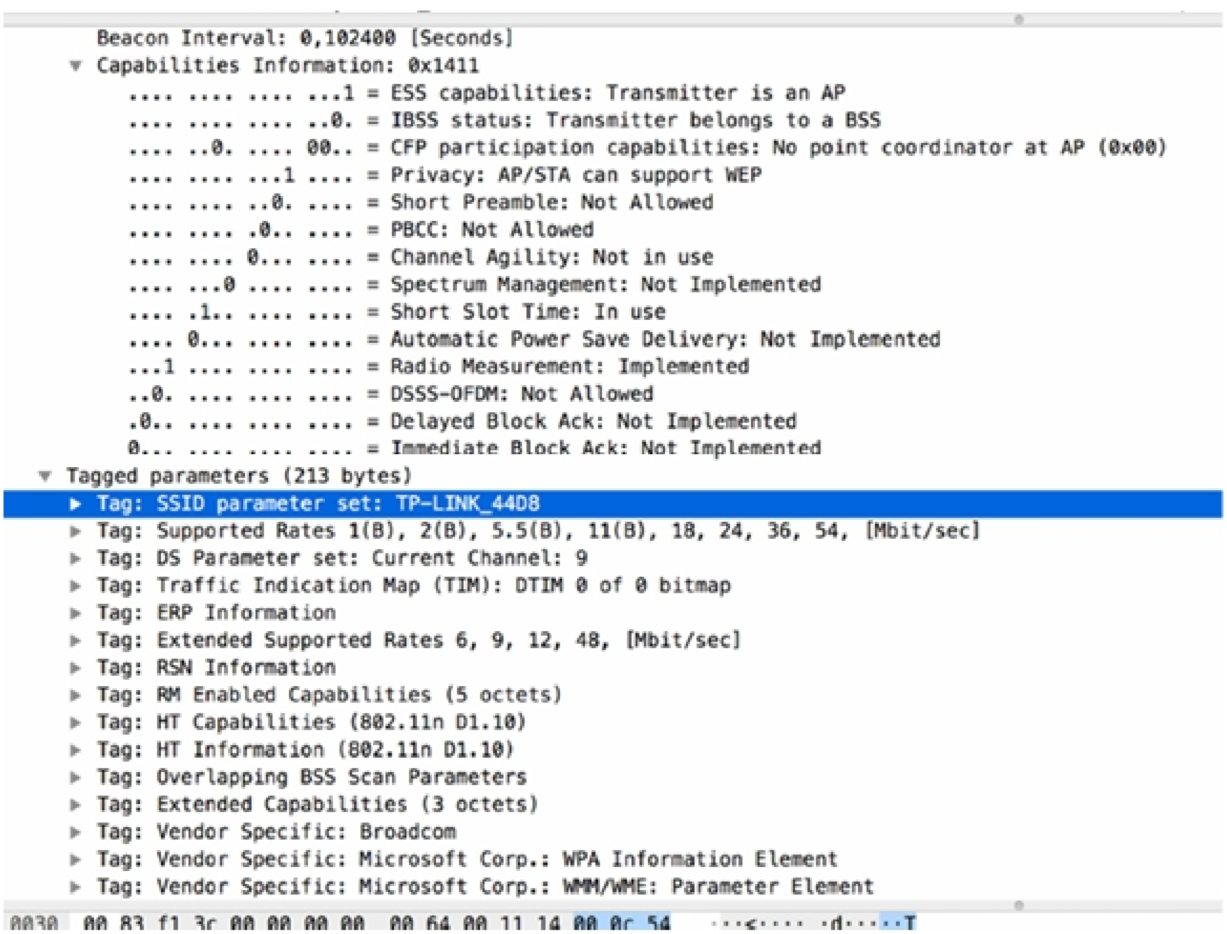


To successfully complete a WLAN connection, the following actions are required:

1. The STA (wireless station) decides which AP to join.
2. The STA successfully completes the authentication process.
3. The STA successfully completes the association process.

The “Tagged parameters” field, shown in the figure below, provides a set of information in the Packet Details pane. The following list provides some of the common parameters:

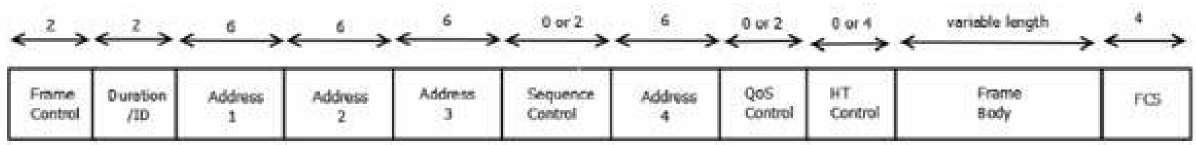
- SSID parameter set
- Supported Rates
- DS Parameter set (indicating the used channel)
- Extended Supported Rates



Task 2:
Learning to dissect the 802.11 frame structure is important, considering that 802.11 headers contain much more information than a simple Ethernet header that contains three fields (excluding the FCS at the end of the packet). For example, an 802.11 association frame contains 17 fields in the header. Many of the fields are only a single bit long. The retry flag, for example, is only 1 bit long.

The frame body is of variable length, and even the maximum size is of variable length depending upon the encryption type in use. Inspecting the IEEE 802.11 specifications, it is clear that the MAC Service Data Unit (MSDU) is 2304 bytes.

The figure below shows the basic 802.11 frame structure.



The first 32 bytes represent the MAC header. Even if the Frame Control field (the first field in the structure above) is only 2 bytes, it contains a lot of information, and it is possible to build many filters with the fields contained in it.

Because encryption routines affect the length of the 802.11 packets, it is important to remember that each kind of encryption has its own length. In fact, usually, you can find the indicated maximum frame body length for 802.11 frames as 2312, but this is only in the case of WEP encryption. The following list provides the length for each encryption type:

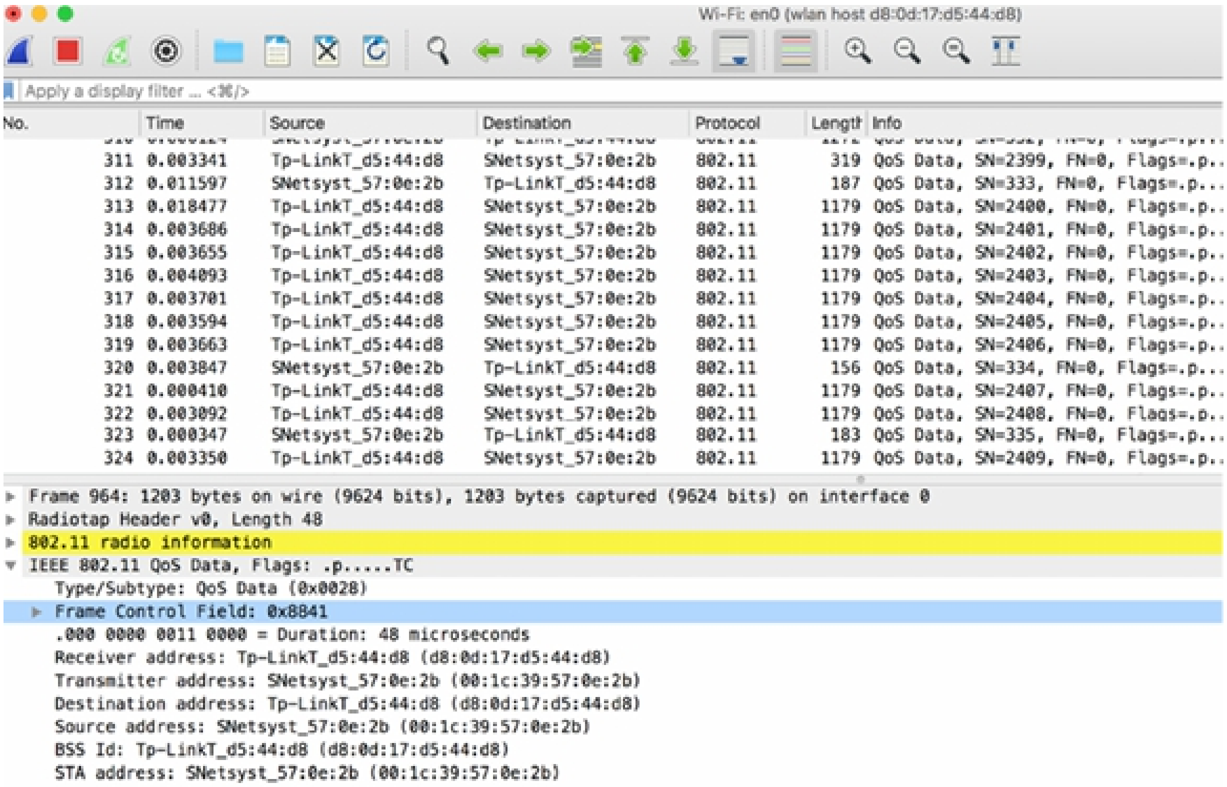
- WEP: It is necessary to add 8 bytes to the MSDU length (2312).
- WPA (TKIP): It is necessary to add 20 bytes to the MSDU length (2324).
- CCMP (WPA2): It is necessary to add 16 bytes to the MSDU length (2320).

The last field of the 802.11 structure is Frame Checking Sequence(FCS). It has a length of 4 bytes, and it provides error checking on the contents of the frame.

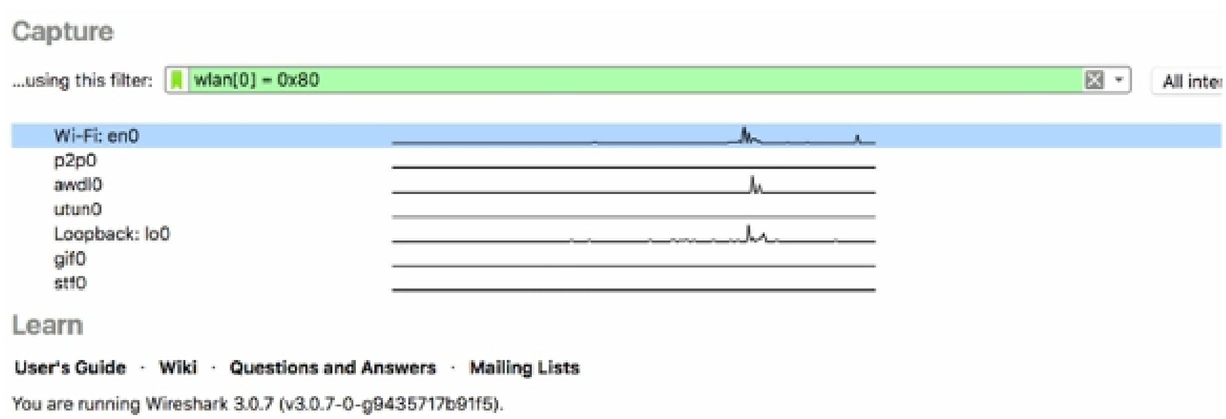
Task 3:
There are a lot of filters available for 802.11. If you need to use a capture filter, you can create one based on a specific host.

Open Wireshark, and in the Capture Filter, enter wlan host d8:0d:17:d5:44:d8 , as shown in the figure below. Your host MAC will differ of course.

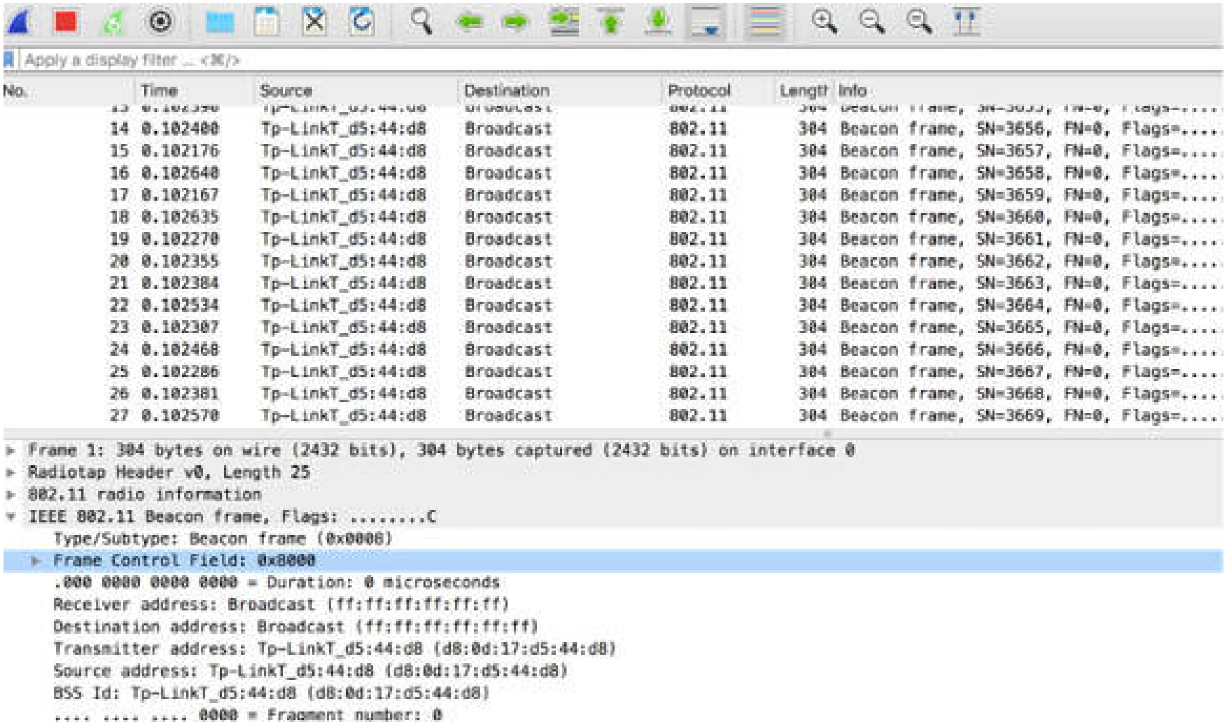
Capture the traffic for a few minutes. In the Packet List pane, only the filtered packets are displayed, as shown in the figure below.



To capture only specific frames (for example, only beacons frames), in the Capture Filter, enter wlan[0] = 0x80 , as shown in the figure below.



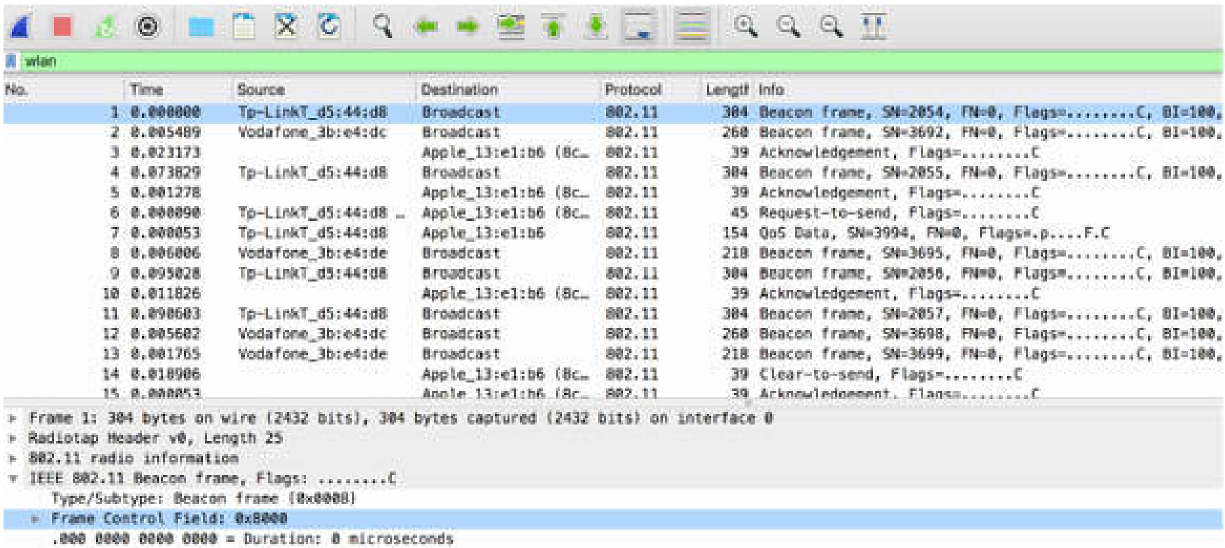
In the Packet List pane, only the specified frames are displayed, as shown in the figure below.



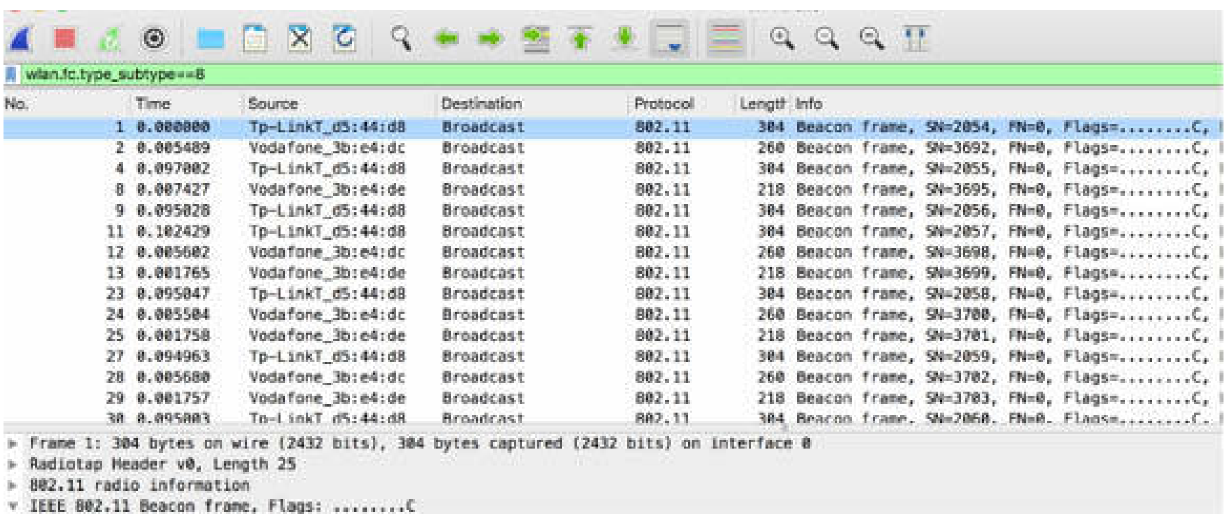
To build capture filters based on one of the four WLAN address fields, you can use the following syntax:

- `wlan addr1 d8:0d:17:d5:44:d8` for the Receiver Address capture filter (addr1)
- `wlan addr2 d8:0d:17:d5:44:d8` for the Transmitter Address capture filter (addr2)
- `wlan addr1 d8:0d:17:d5:44:d8` for the Destination Address capture filter (dst)
- `wlan addr1 d8:0d:17:d5:44:d8` for the Source Address capture filter (src)

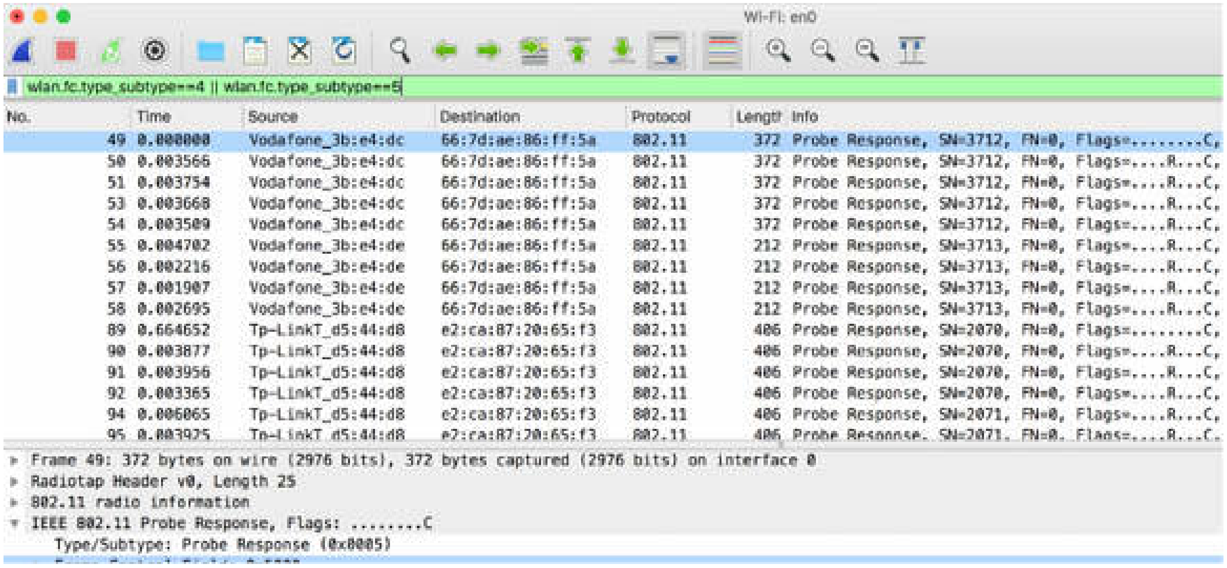
In addition to the capture filters, a lot of display filters are available for 802.11. A starting point is the basic display filter for 802.11, that is, `wlan` .



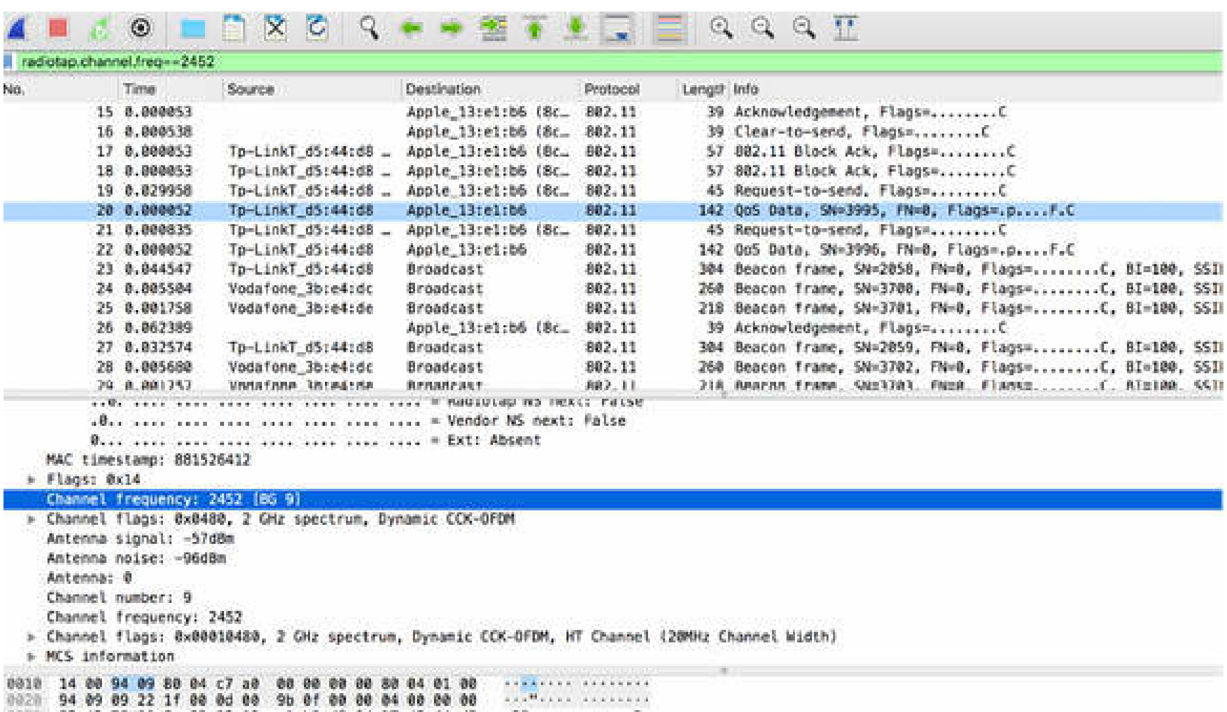
To display only beacon frames, in the filter toolbar, enter wlan.fc.type_subtype==8 , as shown in the figure below.



To display only Probe Requests or Probe Responses, in the filter toolbar, enter wlan.fc.type_subtype==4 || wlan.fc.type_subtype==5 , as shown in the figure below.



To display only a specific channel, in the filter toolbar, enter radiotap.channel.freq==2452 . This display filter is based on the radiotap header. The channel frequency information is displayed in the Packet Details pane, as shown in the figure below.



Notes:

Repeat the previous steps to capture more capture log files and try to identify the previously explained fields. Use capture filters and display filters to select only packets of interest, based on the predefined criteria.

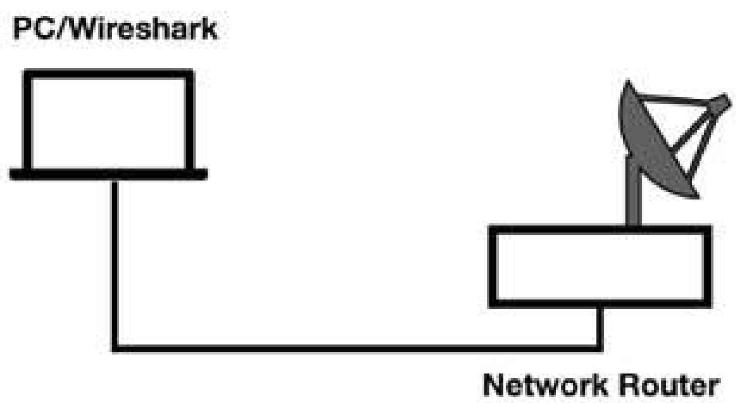
Lab 80. 802.11 Frame Control Field

Lab Objective:
Learn how to analyze the Frame Control types and subtypes.

Lab Purpose:
Understand the Frame Control field and the information contained in each of its subfields.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

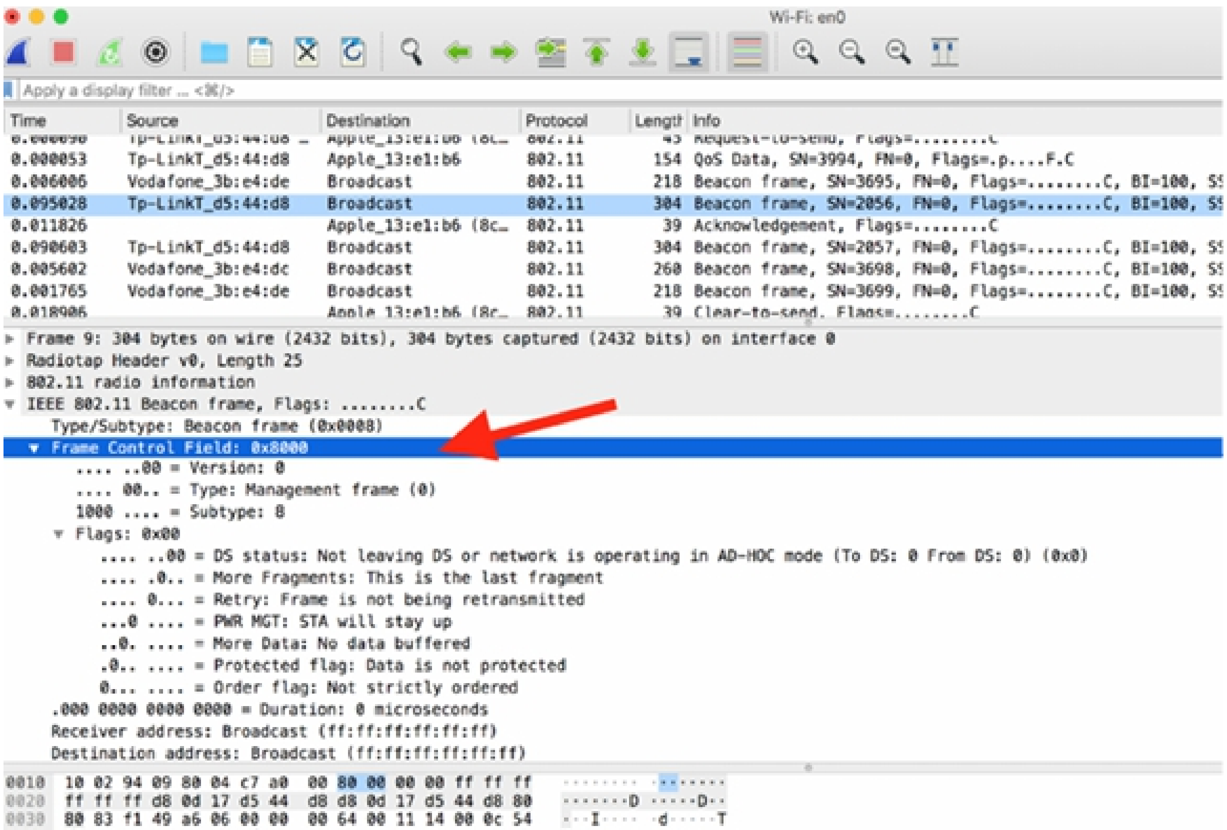
Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless connection to a network router that has access to the internet.



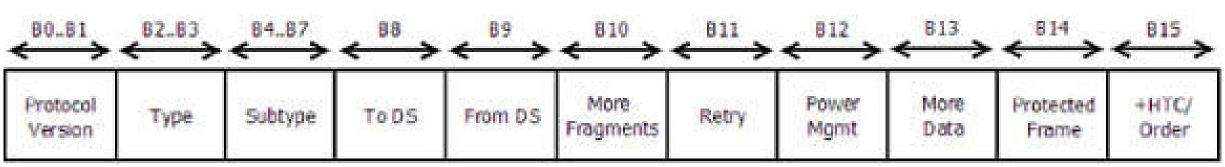
Lab Walkthrough:

Task 1:

In Wireshark, open the capture file saved in the previous lab. In the Packet List pane, select a packet. In the Packet Details pane, open the "Frame Control Field" tree view, as shown in the figure below.



The Frame Control field is the first field of the 802.11 packet. It consists of 2 octets and contains numerous individual fields, as shown in the figure below.



The first three fields (Protocol Version, Type, and Subtype) are always present in the Frame Control field. The following list describes the Frame Control fields and their sequence during transmission:

- Protocol Version: Protocol Version Number—always set to 00 at this time
- Type/Subtype: Management, Control, Data Frame
- To DS/From DS: 0,0 between stations in same BSS (DS distribution system); 0,1 to DS; 1,0 From DS; 1,1 From DS to DS
- More Fragments: Set to 1, fragmentation is set at the 802.11 MAC layer
- Retry: Set to 1, this is an 802.11 retransmission[120]
- Power Management: Set to 1, the STA is stating it is in the power save mode
- More Data: Typically used by AP to tell the STA in the power save mode that more data is buffered for it
- Protected Frame: Set to 1 when data is encrypted
- Order: Set to 1 when order is important; discard the frame when out of order

802.11 frames can use up to five address fields that are abbreviated as follows:

- BSSID—Basic Service Set identifier
- DA—Destination address
- SA—Source address
- RA—Receiver address
- TA—Transmitter address

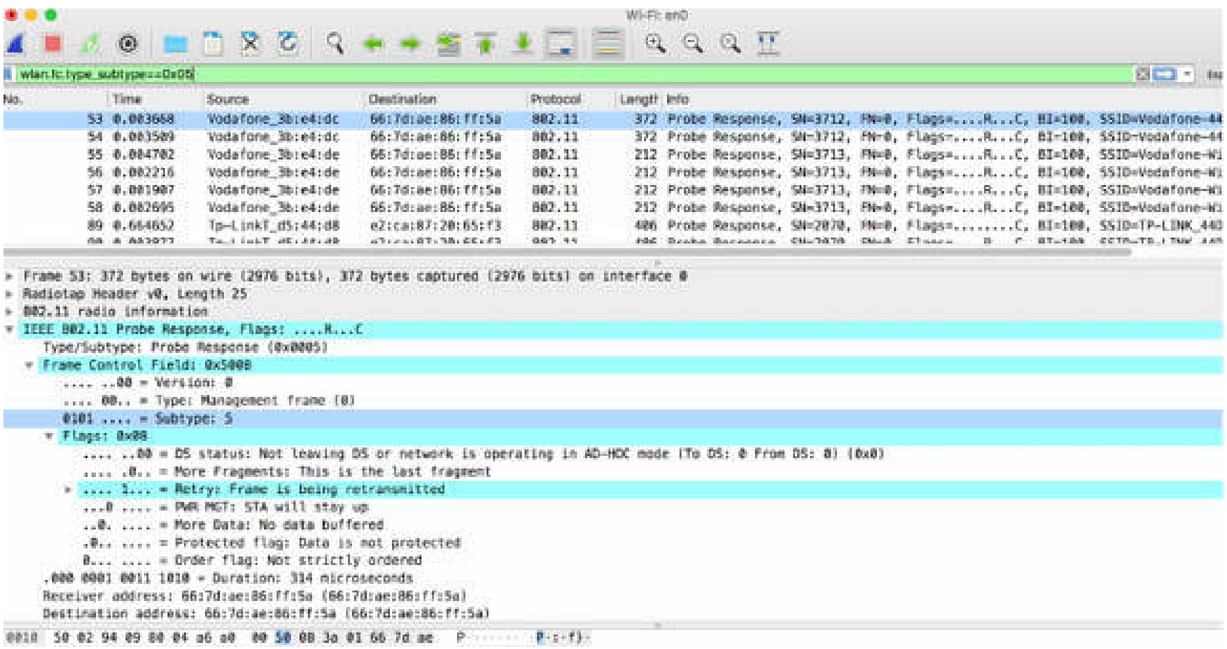
Task 2:
You can create display filters based on the Type/Subtype values of the Frame Control field.

In the filter toolbar, enter wlan.fc.type_subtype == 0x…… to translate the type/subtype values to hex format. For example, to filter only Probe

Response packets, the process is as follows:

- Convert 000101 to hex
- Take the first two bits = 00 = 0x0
- Take the next four bits = 0101 = 0x5
- Create the display filter as wlan.fc.type_subtype==0x05

The results will be similar to the ones displayed in the figure below.



To display both Probe Requests and Probe Replies, in the filter toolbar, enter wlan.fc.type_subtype==0x05 or wlan.fc.type_subtype==0x04 .

To filter out the type and subtype values of the management, control, and data frames, the following list describes various display filters for 802.11 traffic:

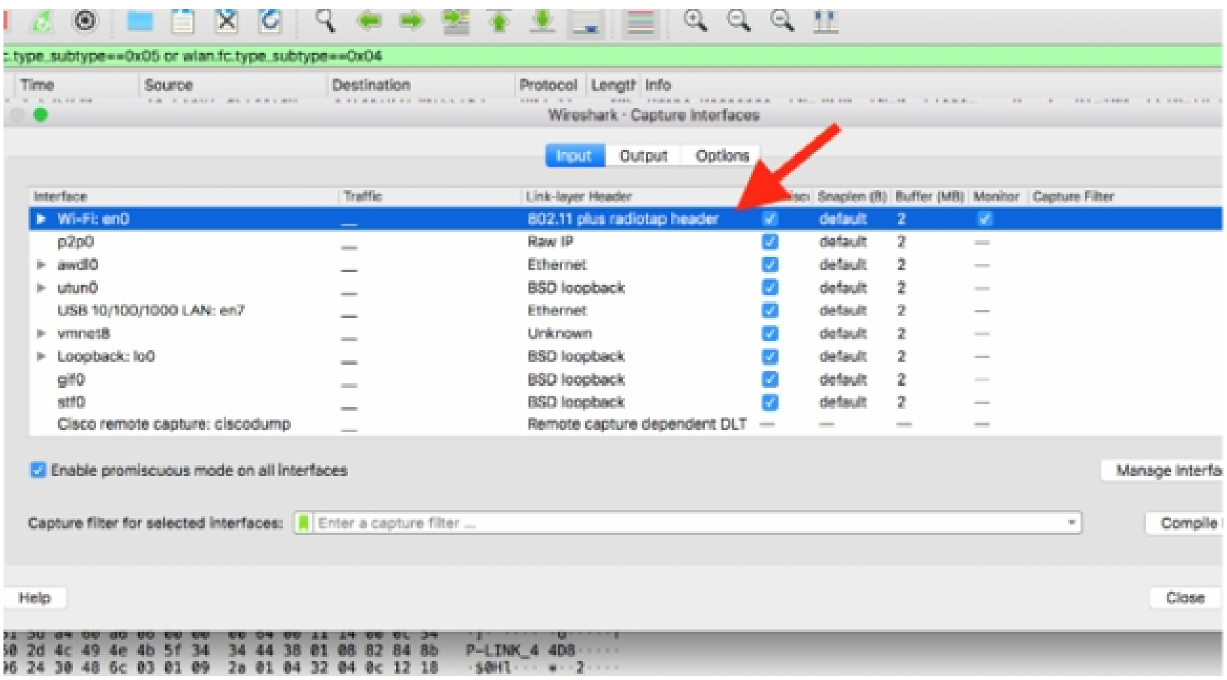
- wlan.fc.type_subtype==0 —(Management) Association request
- wlan.fc.type_subtype==1 —(Management) Association response
- wlan.fc.type_subtype==2 —(Management) Reassociation request
- wlan.fc.type_subtype==3 —(Management) Reassociation response

- wlan.fc.type_subtype==4 —(Management) Probe request
- wlan.fc.type_subtype==5 —(Management) Probe response
- wlan.fc.type_subtype==6 || wlan.fc.type_subtype==7 —(Management) Reserved
- wlan.fc.type_subtype==8 —(Management) Beacon
- wlan.fc.type_subtype==9 —(Management) ATIM
- wlan.fc.type_subtype==10 —(Management) Disassociation
- wlan.fc.type_subtype==11 —(Management) Authentication
- wlan.fc.type_subtype==12 —(Management) Deauthentication
- wlan.fc.type_subtype==13 —(Management) Action
- wlan.fc.type_subtype==14 || wlan.fc.type_subtype==15 —(Management) Reserved
- wlan.fc.type_subtype > 15 && wlan.fc.type_subtype <= 0x23 —(Control) Reserved
- wlan.fc.type_subtype==24 —(Control) Block ACK request
- wlan.fc.type_subtype==25 —(Control) Block ACK
- wlan.fc.type_subtype==26 —(Control) PS-Poll
- wlan.fc.type_subtype==27 —(Control) Request to send
- wlan.fc.type_subtype==28 —(Control) Clear to Send
- wlan.fc.type_subtype==29 —(Control) ACK
- wlan.fc.type_subtype==30 —(Control) CF-End
- wlan.fc.type_subtype==31 —(Control) CF-End + CF-ACK
- wlan.fc.type_subtype==32 —(Data) Data
- wlan.fc.type_subtype==33 —(Data) Data + CF-ACK
- wlan.fc.type_subtype==34 —(Data) Data + CF-Poll
- wlan.fc.type_subtype==35 —(Data) Data+CF-ACK + CF-Poll
- wlan.fc.type_subtype==36 —(Data) Null (no data)
- wlan.fc.type_subtype==37 —(Data) CF-Ack (no data)
- wlan.fc.type_subtype==38 —(Data) CF-Poll (no data)
- wlan.fc.type_subtype==39 —(Data) CF-ACK + CF-Poll (no data)
- wlan.fc.type_subtype==40 —(Data) QoS Data
- wlan.fc.type_subtype==41 —(Data) QoS Data + CF-ACK
- wlan.fc.type_subtype==42 —(Data) QoS Data + CF-Poll
- wlan.fc.type_subtype==43 —(Data) QoS Data + CF-ACK + CF-Poll
- wlan.fc.type_subtype==44 —(Data) QoS Null (no data)

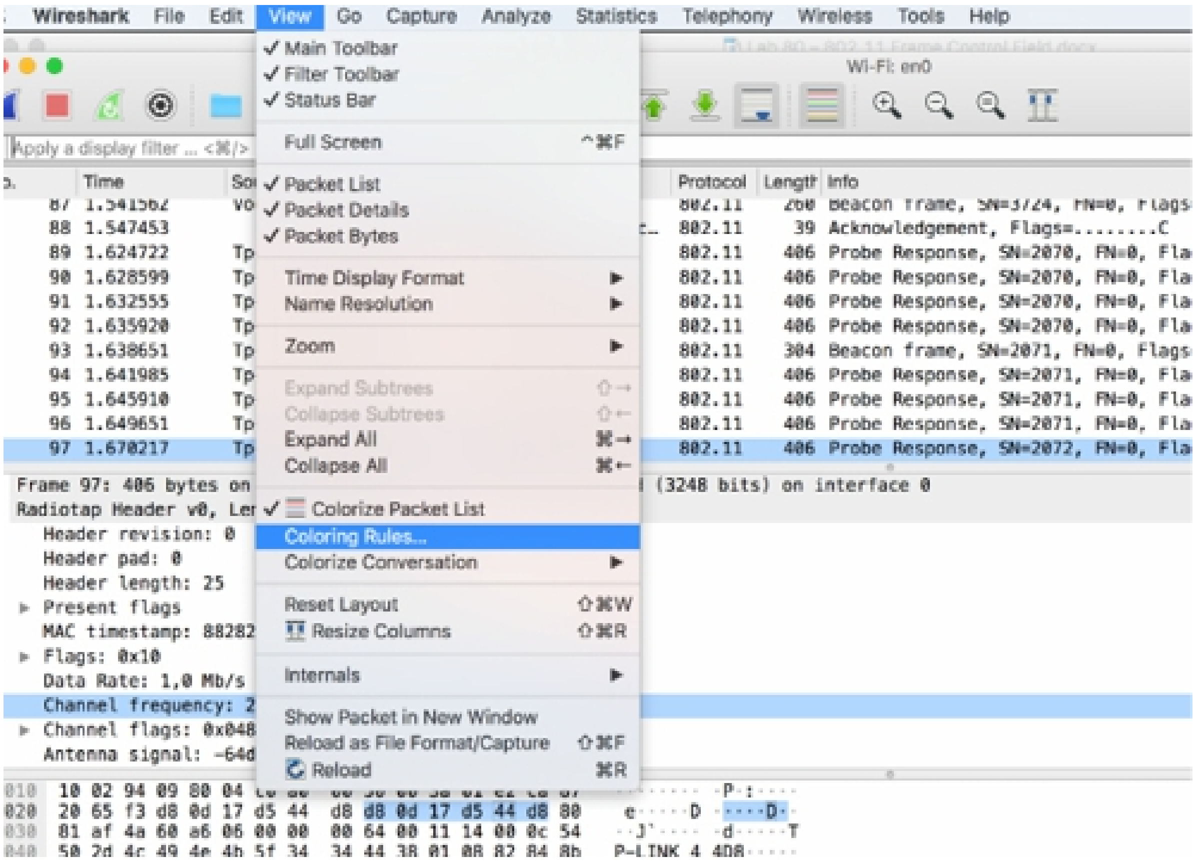
- wlan.fc.type_subtype==45 —(Data) Reserved
- wlan.fc.type_subtype==46 —(Data) QoS CF-Poll (no data)
- wlan.fc.type_subtype==47 —(Data) QoS CF-ACK + CF-Poll (no data)
- wlan.fc.type_subtype > 47 && wlan.fc.type_subtype <= 63 —(Reserved) Reserved

Task 3:
When using Wireshark for WLAN analysis, it is very useful to create an appropriate profile dedicated to WLAN analysis.

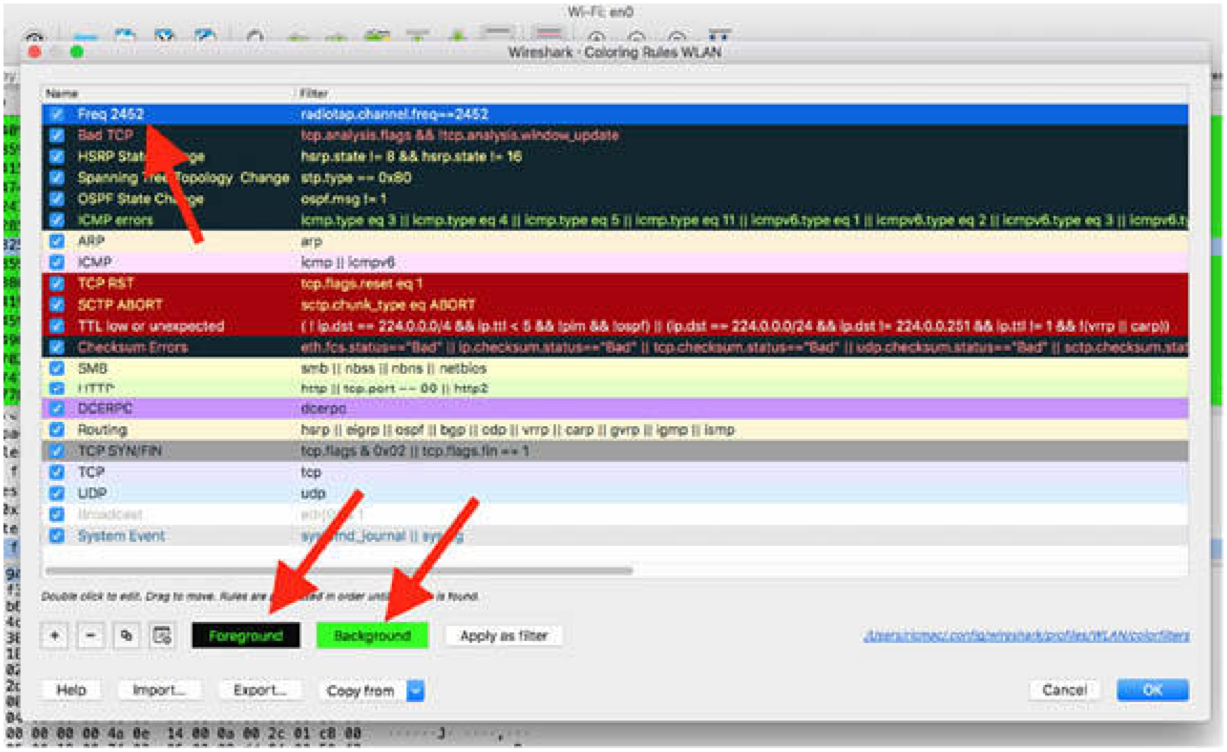
Select the radiotap header or the PPI header for the received packets, as described in Lab 77.



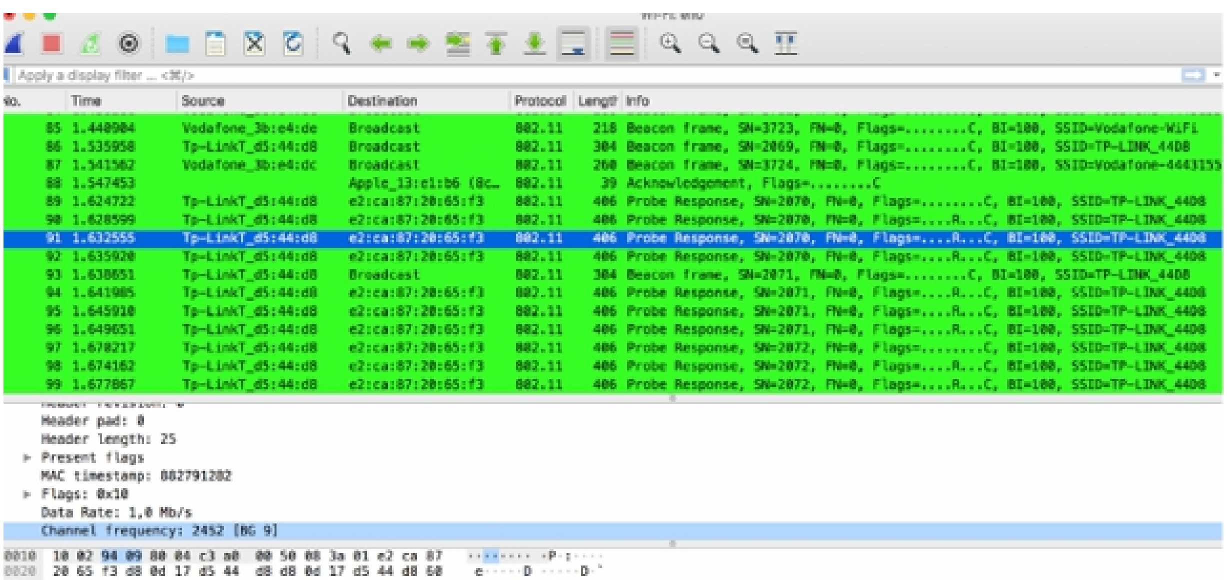
Associate some coloring rules to the packets that you are interested in; for example, if you are interested in a set of packets in channel frequency 2452, set the following coloring rule for the filter: radiotap.channel.freq==2452. To do this, on the main menu, select View > Coloring Rules.



In this case, a green background and a green foreground are associated with the filter, as shown in the figure below. Feel free to download the color images from the resources page at www.101labs.net .



Considering that all the captured packets are in frequency 2452, the result will look similar to the figure below.



Notes:

Repeat the previous steps to create a Wireshark profile that allows you to identify different frequencies and different packet types (for example, retries).

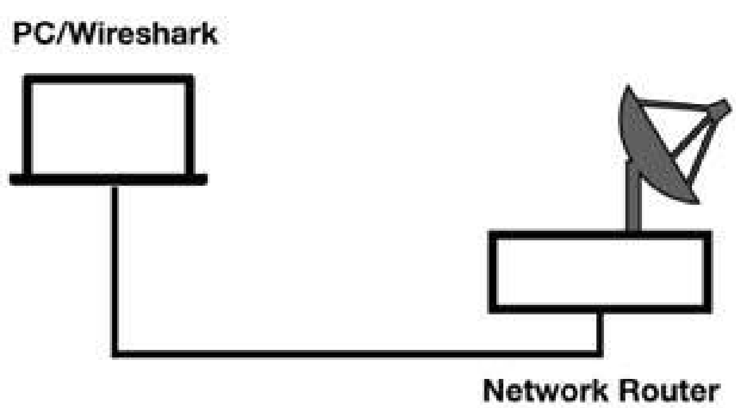
Lab 81. Voice Over Internet Protocol

Lab Objective:
Learn how Voice Over IP (VoIP) works and why is it used.

Lab Purpose:
Understand the main purpose of VoIP and its main features.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
VoIP, also called IP telephony, is a method and a group of technologies for delivering voice communications and multimedia sessions over Internet Protocol networks (IP Networks), such as the Internet network.

The steps and principles involved in originating VoIP calls through a telephone are similar to the traditional digital telephony—signaling, channel setup, digitization of the analog voice signals, and encoding. Instead of being transmitted over a circuit-switched network, the digital information is packetized and transmitted as IP packets over a packet-switched network. Media streams are transported by using special media delivery protocols that encode audio and video with audio codecs and video codecs.

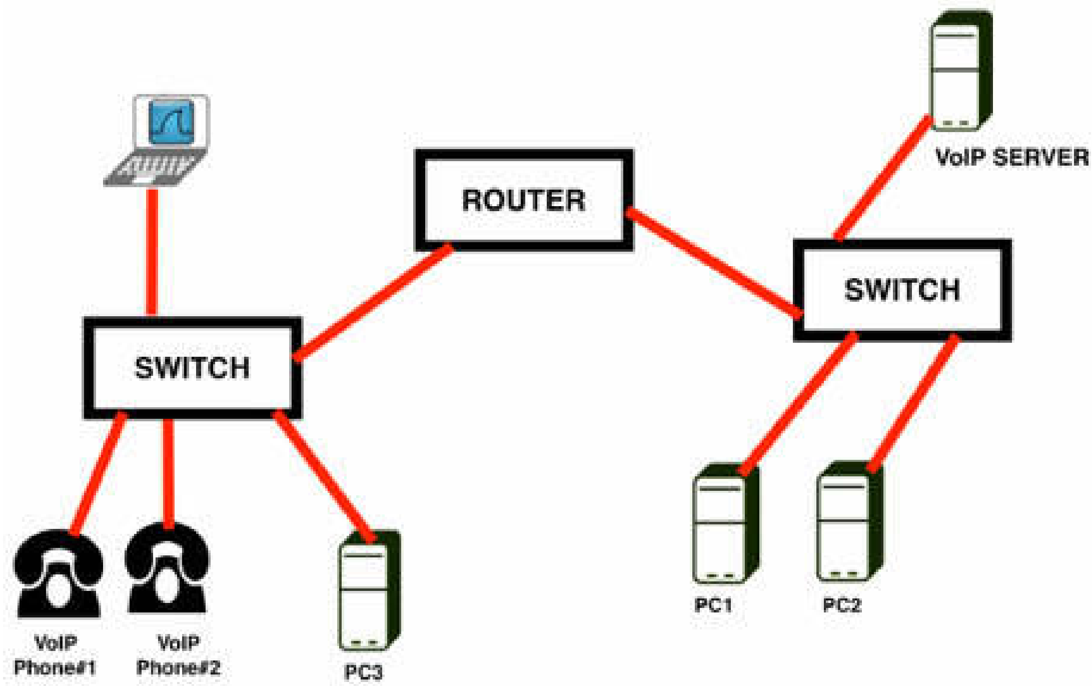
VoIP communications consist of two primary parts—the signaling protocol for call setup and teardown and the transport protocol for voice communications.

Some examples of VoIP signaling protocols are:

- Session Initiation Protocol (SIP) is an example of a VoIP signaling protocol. SIP can run over UDP or TCP port 5060. It is more common to see SIP running over UDP.
- Skinny Call Control Protocol (called SCCP or "Skinny") is a Cisco proprietary protocol used between Cisco VoIP phones and the Cisco Call Manager.
- Realtime Transport Protocol (RTP) carries the voice call itself. Wireshark includes an RTP player that enables you to play back VoIP conversations.
- Realtime Transport Control Protocol (RTCP) provides out-of-band statistics and control information for an RTP flow. RTP can run over any even port number, whereas RTCP runs over the next higher odd port number. For example, if RTP runs over port 7000, RTCP runs over port 7001.
- Dual-Tone MultiFrequency (DTMF) telephony events are the events that you may see during your VoIP analysis sessions. DTMF is the tone that is sent when you press a button on a phone, for example,

when you dial an extension number. Sometimes these signals are sent in the voice channel, which is called in-band signaling. More often, you'll see separate control packets for DTMF, which is called out-of-band signaling. Wireshark recognizes and dissects out-of-band DTMF traffic.

To analyze the VoIP traffic, consider placing Wireshark as close as possible to the VoIP phone to obtain the round trip time and packet loss from that phone's perspective, as shown in the figure below.



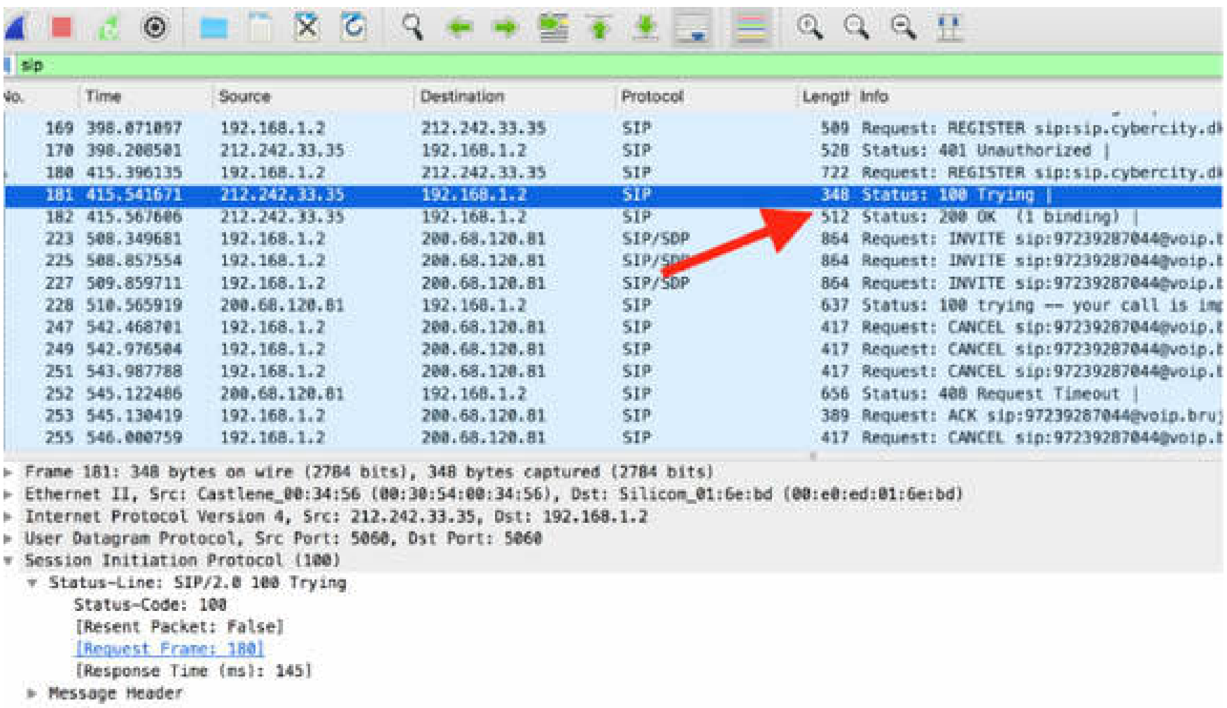
If Wireshark doesn't see the signaling protocol, it may not be able to identify the VoIP datastream and (wrongly) mark the conversation simply as UDP traffic in the protocol column of the Packet List pane.

On the main menu, select Edit > Preferences. In the Preferences dialog box, select RTP in the left tree view and then select the "Try to decode RTP outside of conversations" check box.

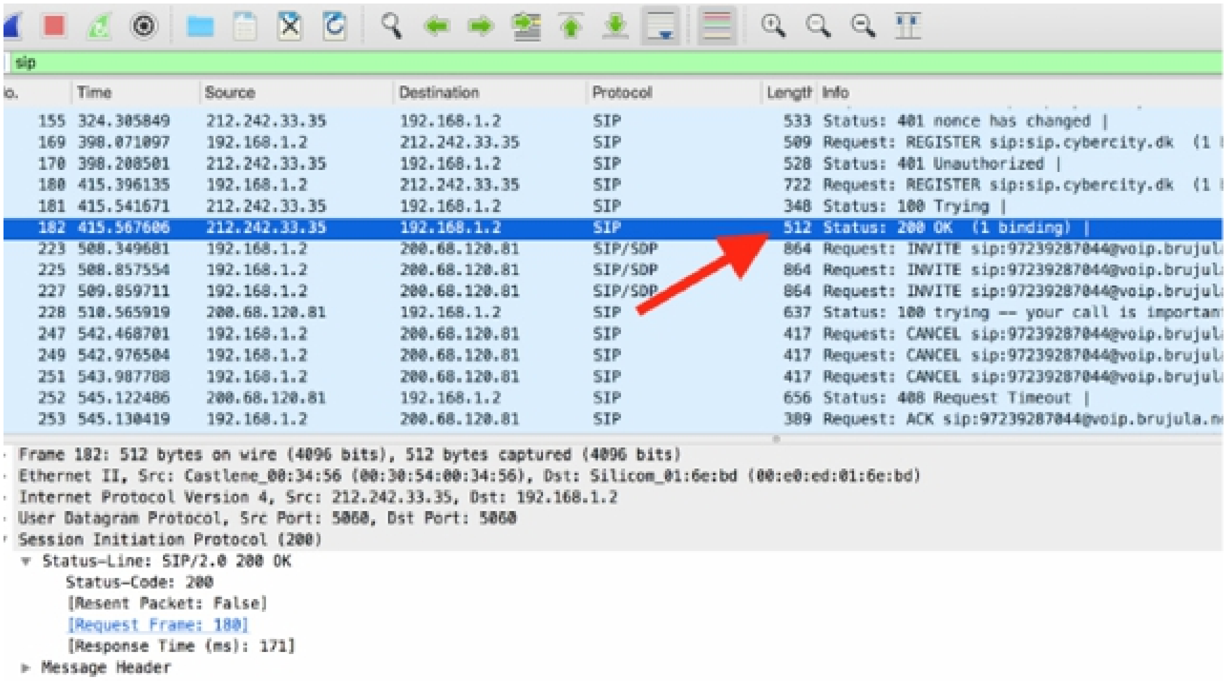
Alternatively, if you are sure that the traffic is RTP, in the Packet List pane, right-click a packet and select "Decode As". Select the UDP port option for “both” and choose RTP in the protocols list.

Task 2:
Download the free sample capture file aaa.pcap from https://wiki.wireshark.org/SampleCaptures and open it in Wireshark. This file contains sample SIP and RTP traffic.

When a VoIP call is initiated, the signaling protocol is used to set up the call. In this case, the signaling traffic flows through the telephony server when the phone sends an invite. The telephony server sends an invite to the target phone while sending a “100 Trying” message to the caller (packet #181).



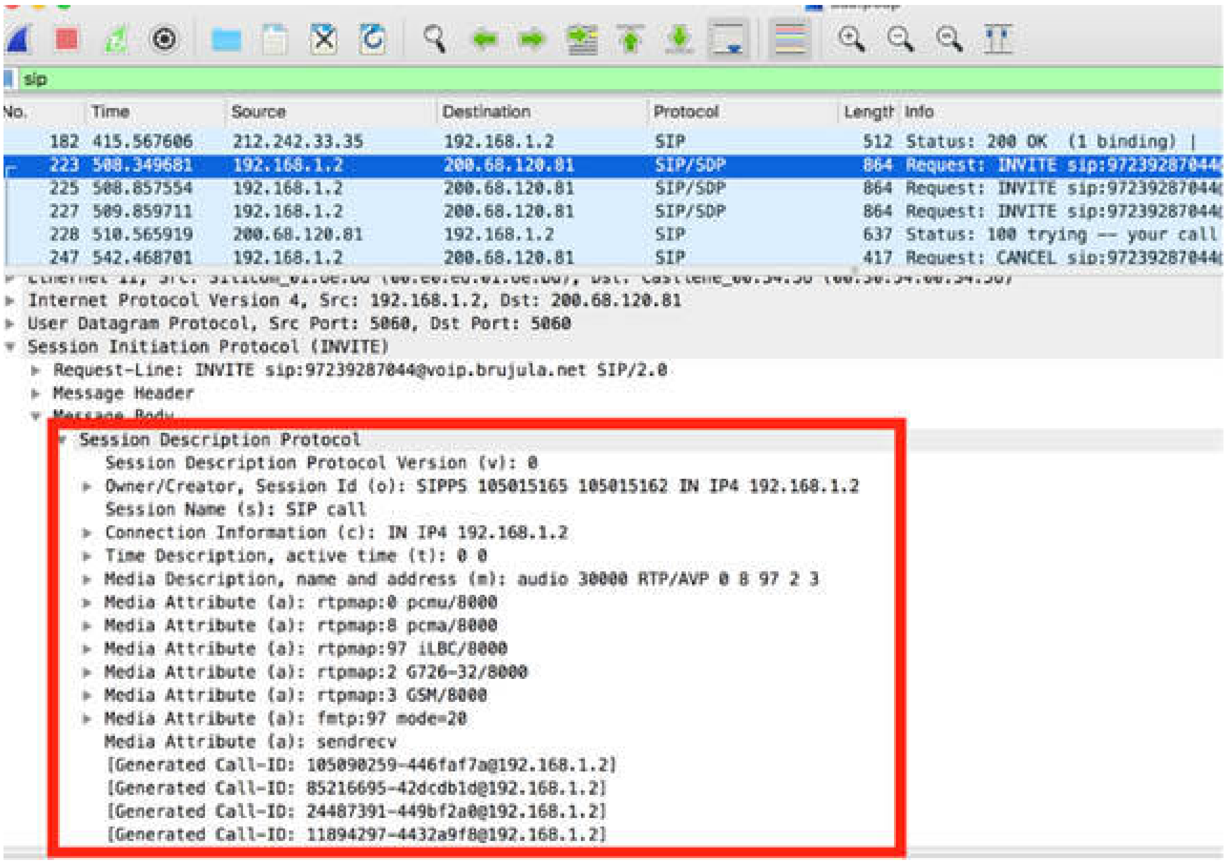
When the user picks up the phone, a “200 OK” message is sent indicating that the call has been accepted (packet #182).



Task 3:

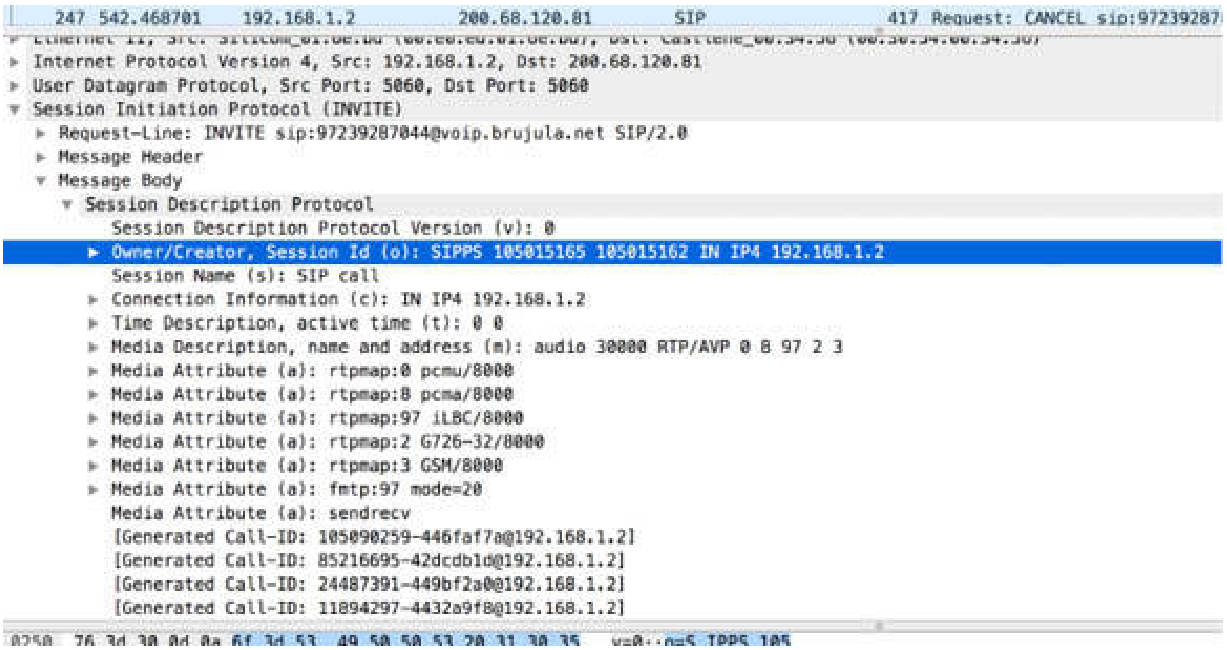
In the Packet List pane, click packet #223. In the Packet Details pane, view the information related to the “Message Body” and “Session Description Protocol” fields.

This SIP packet contains the Session Description Protocol (SDP), which is used to provide information about media streams in multimedia sessions.

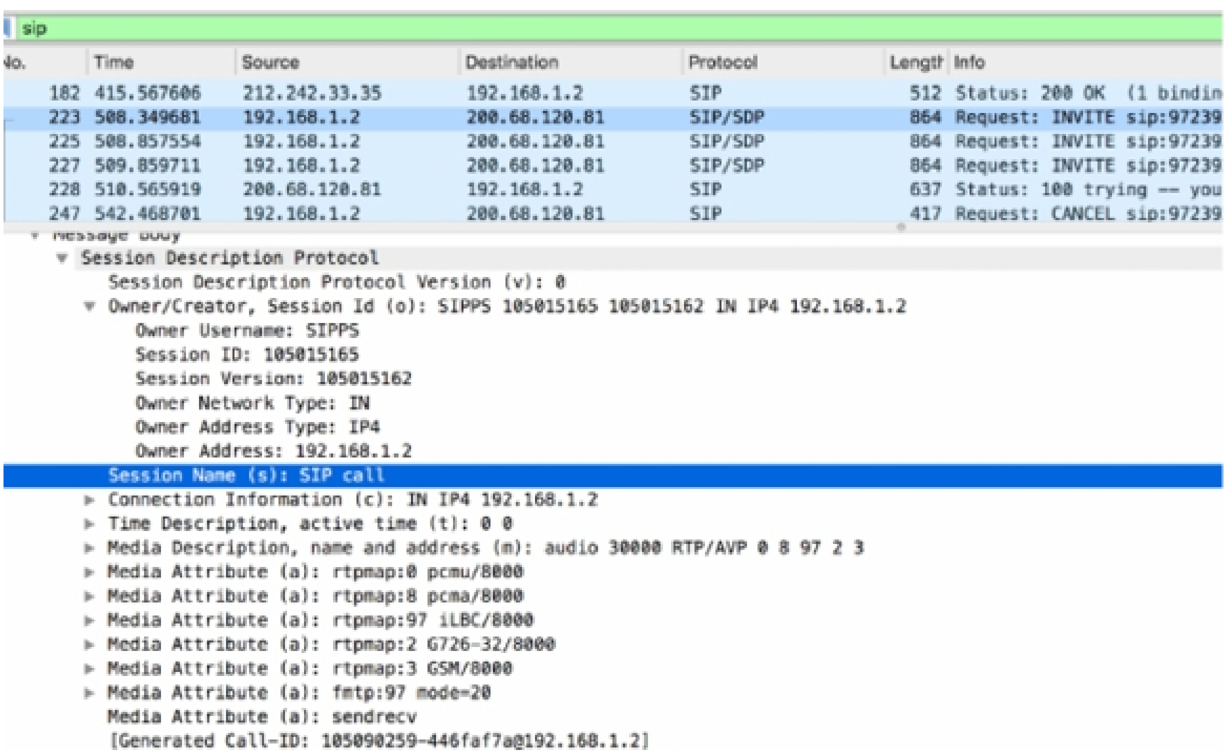


The SDP information, as displayed in the figure below, includes the following fields:

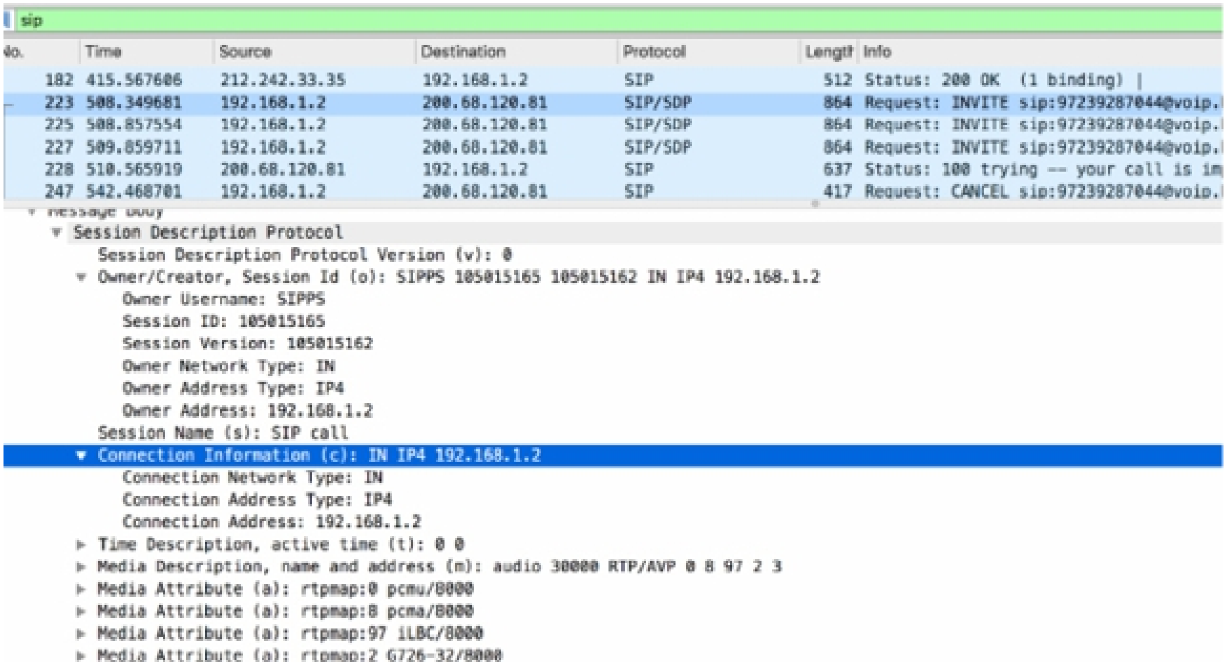
1. The owner (or creator) of the session



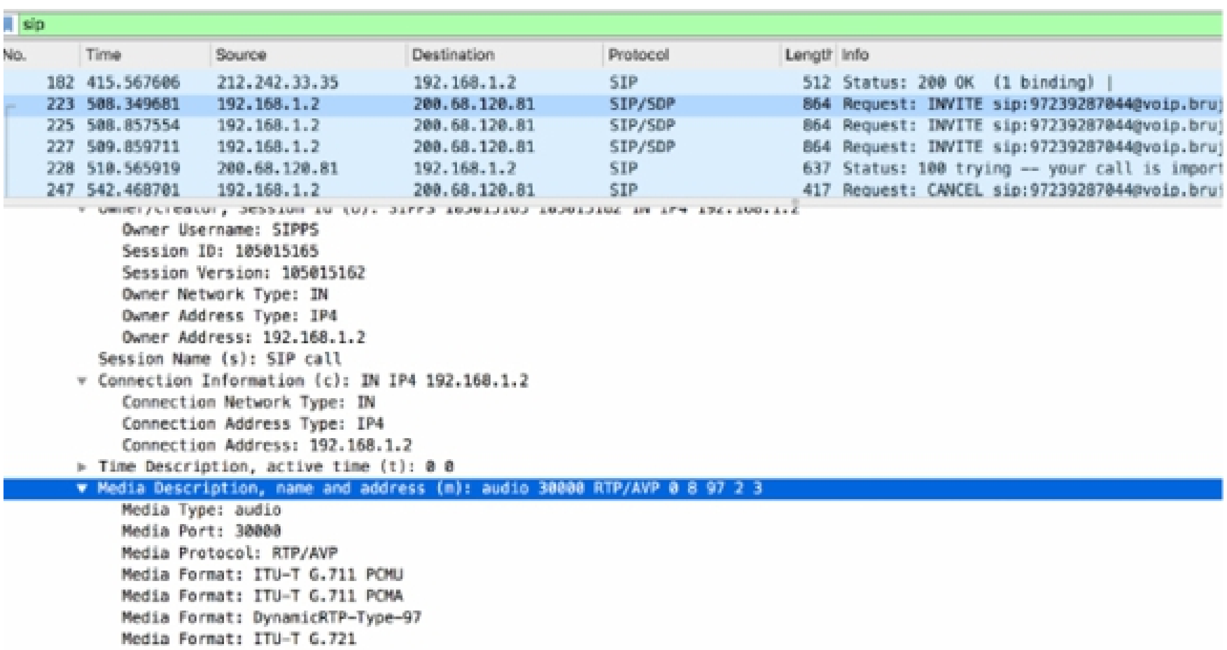
2. The session name



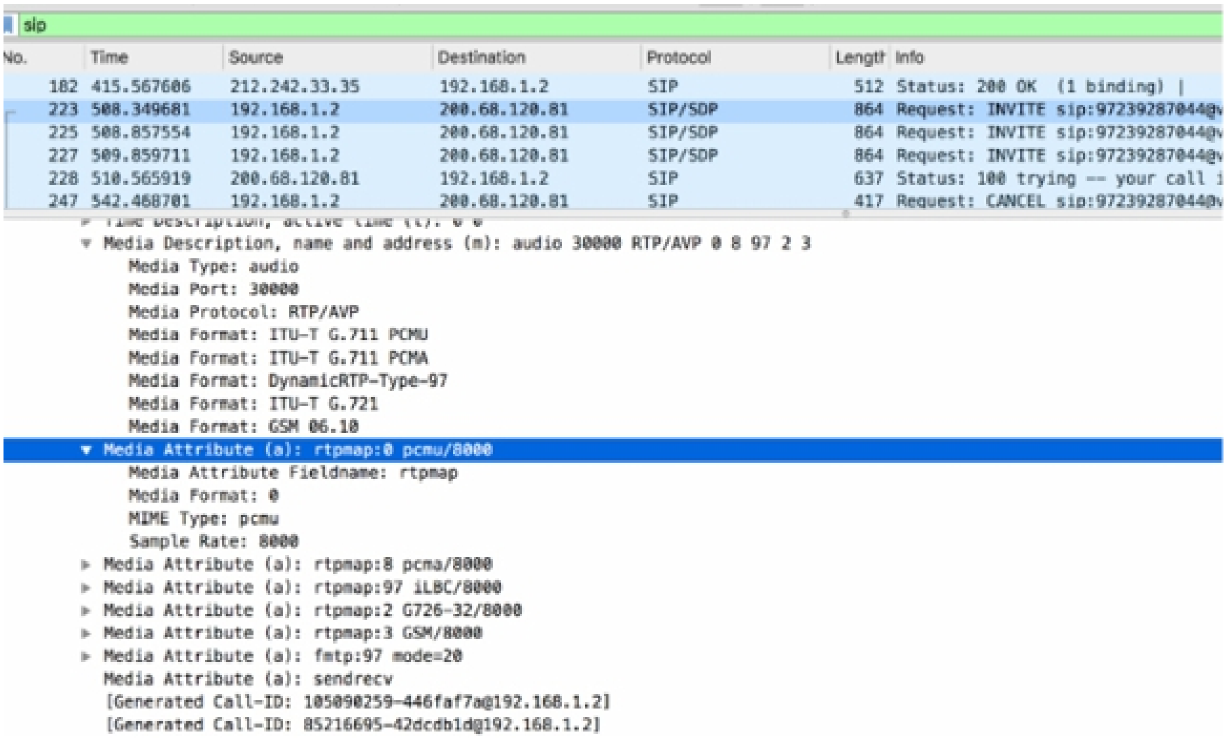
3. The connection information (i.e., IP address)



4. The media description (name and address)



5. The media attributes



Notes:

To gain confidence in understanding the VoIP traffic flow, repeat the previous steps to identify the main packets belonging to the session in a SIP capture.

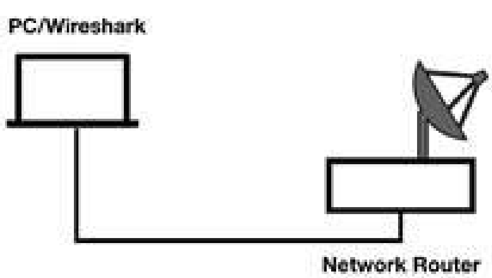
Lab 82. VoIP Problems

Lab Objective:
Learn about the more common VoIP problems.

Lab Purpose:
Learn how to detect and analyze the more common VoIP problems.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

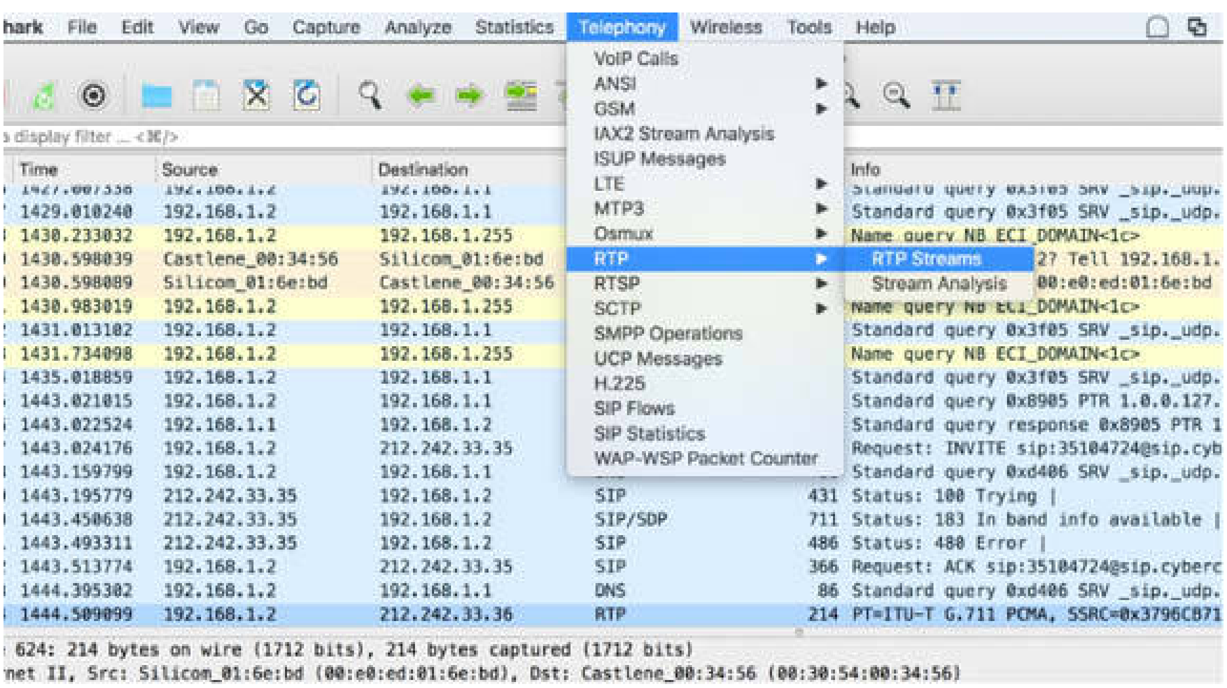
Task 1:
VoIP communications can experience problems such as the calls not going through or the degraded call quality. In addition, there might be cases when the caller may hear echoing, or their voice may drop out sporadically.

In general, packet loss and jitter are the most common causes of VoIP communications not working correctly.

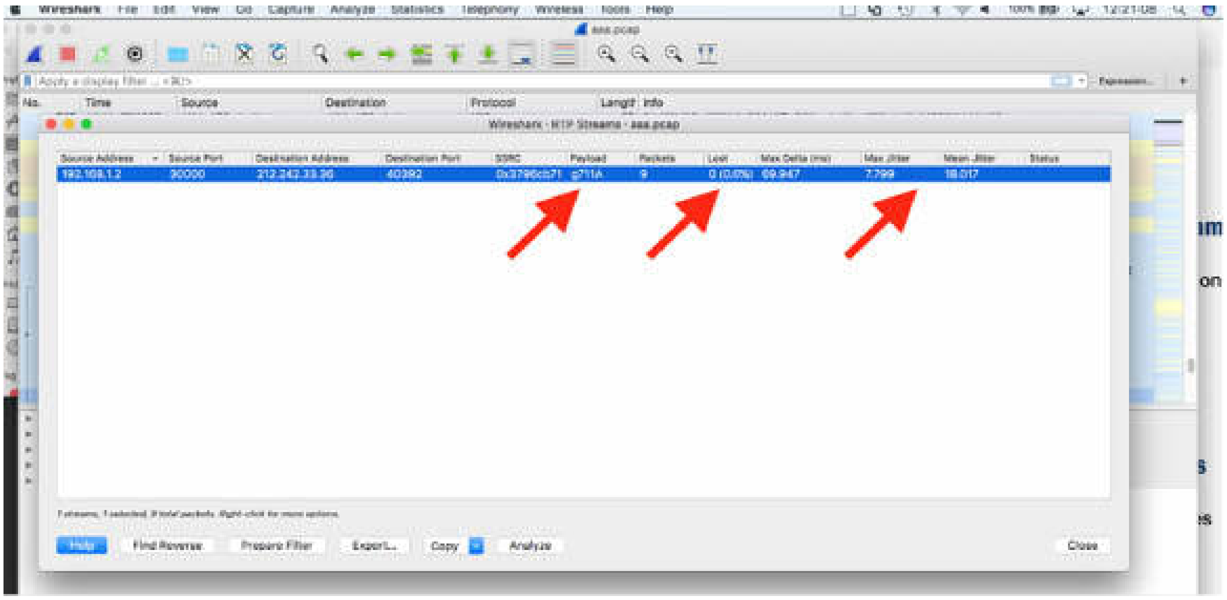
The Realtime Transport Protocol (RTP) typically uses UDP to transport data. Because UDP is a connectionless protocol, if a packet is lost, it is not retransmitted. UDP does not track packets to ensure their arrival at the destination. It is a task of the application to retransmit the lost packets. In the case of VoIP, the packet retransmission is not a good idea because one thing that can happen is that conversations can contain words in the wrong order, for example, imagine the following conversation: “Hello,… today….how you…are?”

Download the free sample capture file aaa.pcap from **https://wiki.wireshark.org/SampleCaptures** and open it in Wireshark. This file contains sample SIP and RTP traffic.

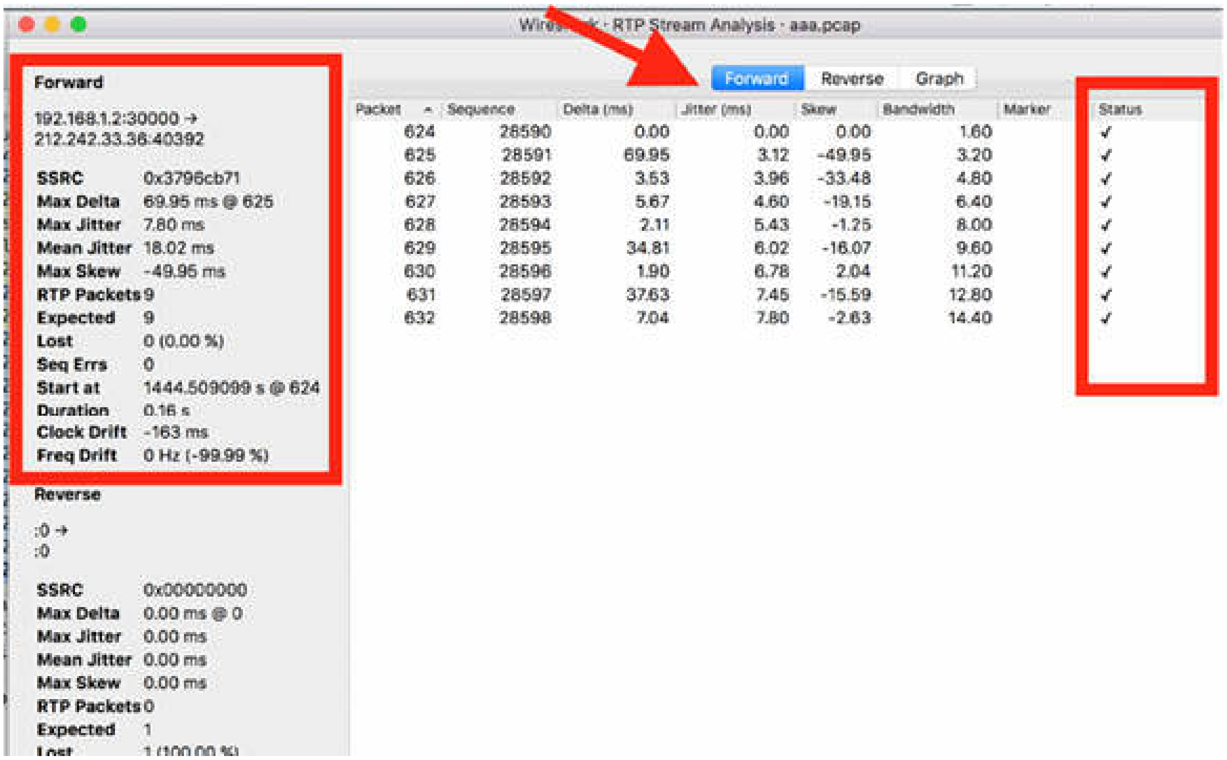
In Wireshark, on the main menu, select Telephony > RTP > RTP Stream, as shown in the figure below.



The RTP Streams dialog box is displayed, containing information related to packet loss, jitter, and payload for each communication flow, as shown in the figure below.



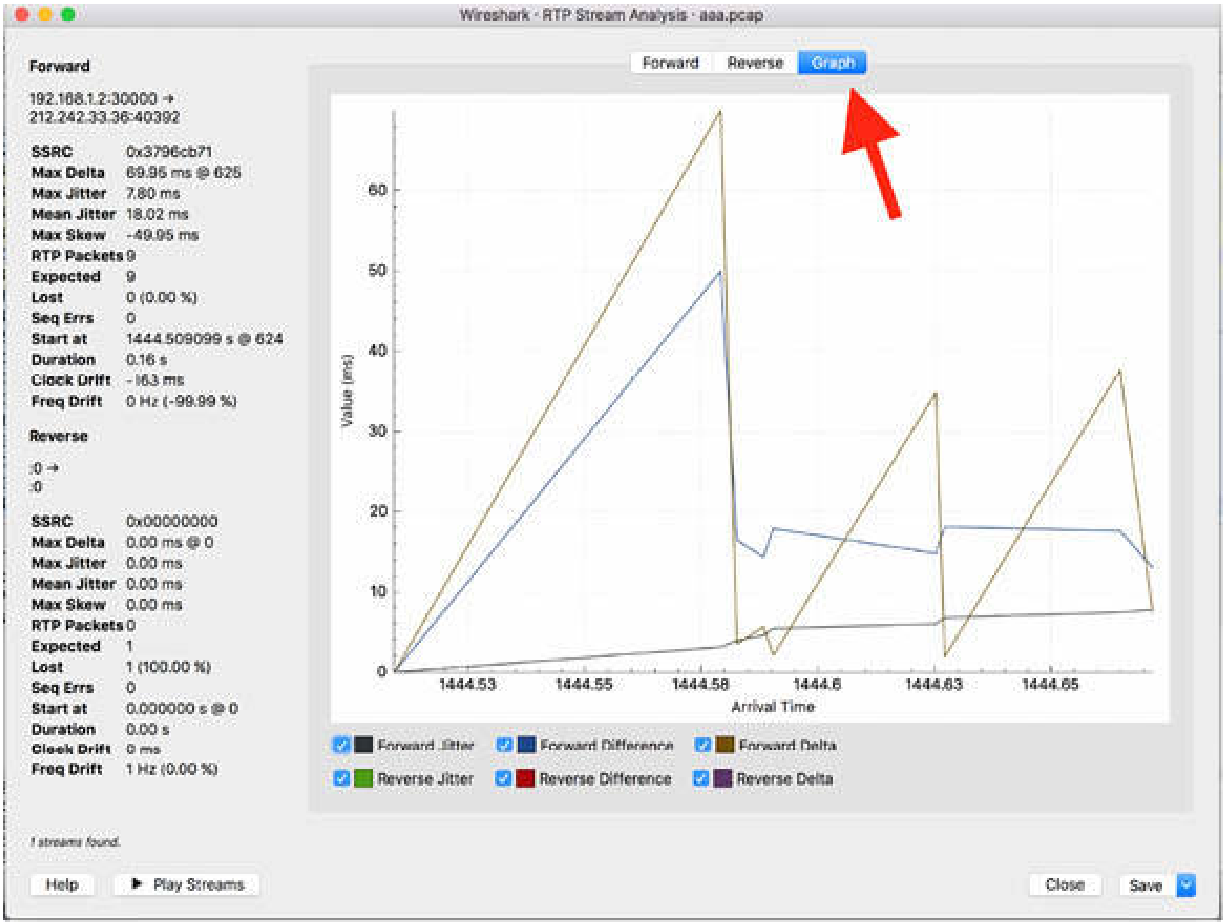
Select a communication flow, and click Analyze. Wireshark lists the packets in the RTP stream and provides additional details on the VoIP call. The Status column indicates problems in the RTP streams. The Forward tab is displayed in the figure below.



Packet loss can occur because of timing issues such as excessive jitter or clock skew between the VoIP endpoints. The Delta (ms) column in the figure above shows the time since the last RTP packet was received. This time should remain fairly consistent, but this is not the case (you can note a max of 69 ms and a min of 1.90 ms).

Jitter is a variance in the packet rate, and it is reported in the Jitter column. Wireshark calculates jitter according to RFC 3550 (RTP). Excessive jitter can be caused by congested networks, load balancing, quality of service configurations, or low bandwidth links. Jitter buffers act like “elastic bands” that help buffer packets to even out the variance in arrival times. If you observe a high jitter rate (above 20ms), you can assume that the call will be affected, and the users will get impacted. If the jitter level is excessively high, packets can be dropped by the jitter buffer in the receiving VoIP host. In this example, the jitter is quite constant—at a low level (below 10ms).

In the RTP Streams dialog box, select the Graph tab. The corresponding values, displayed in the previous figure, are plotted in a graph where you can one by one select the desired plot line, as shown in the figure below.



Notes:

Repeat the previous steps analyzing a different VoIP capture to identify possible problems related to packet loss and/or jitter. Again generate the stream analysis graph and try to identify whether the values are above or below the critical values and whether they can affect the quality of the service.

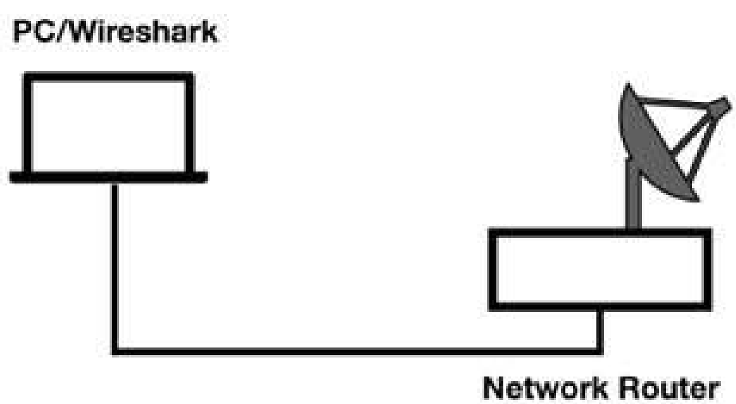
Lab 83. SIP and RTP Traffic

Lab Objective:
Learn to examine the SIP and RTP traffic.

Lab Purpose:
Learn how to identify the fields composing the SIP protocol and the RTP protocol.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

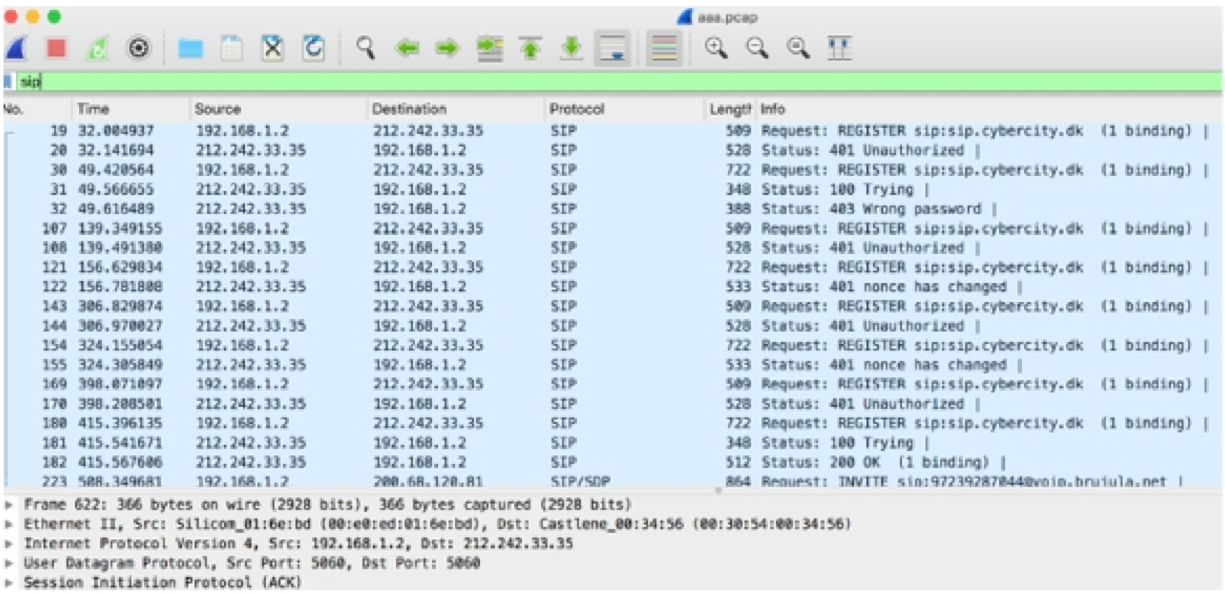


Lab Walkthrough:

Task 1:

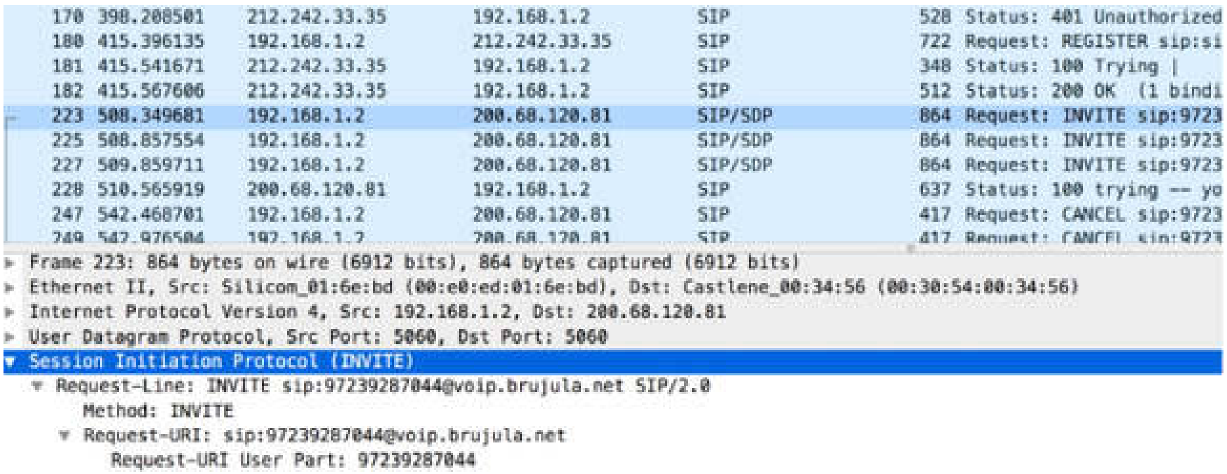
Download the free sample capture file aaa.pcap from https://wiki.wireshark.org/SampleCaptures and open it in Wireshark. This file contains sample SIP and RTP traffic.

In the filter toolbar, enter sip to display only SIP packets in the Packet List pane.



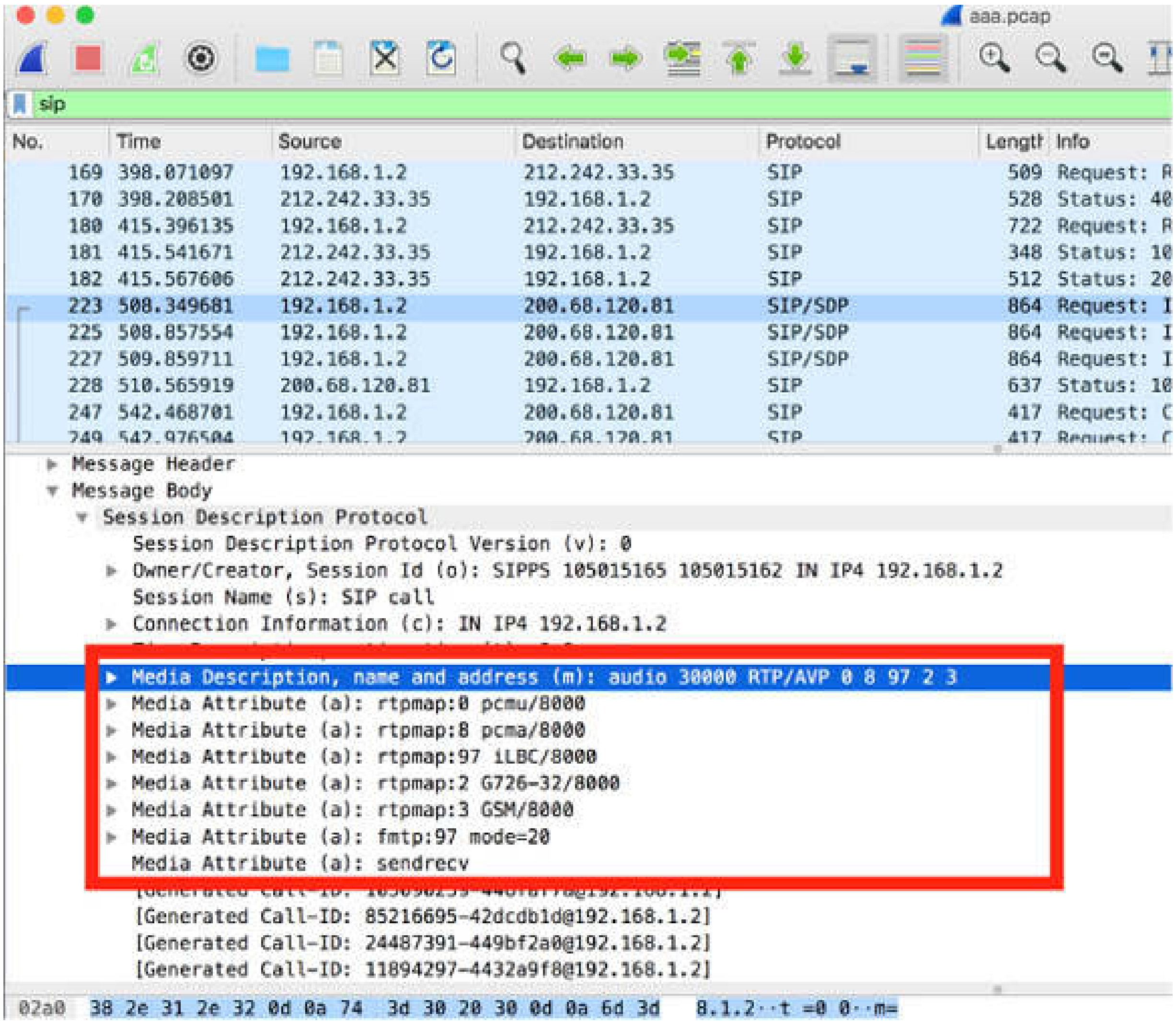
Even though SIP is usually associated with the VoIP call setup, it can be also used to set up other application sessions.

In the Packet List pane, select packet #223, and inspect the related information in the Packet Bytes pane. This packet is a SIP invite packet. This invitation is being sent for extension 97239287044 (97239287044@voip.brujula.net), as shown in the figure below.



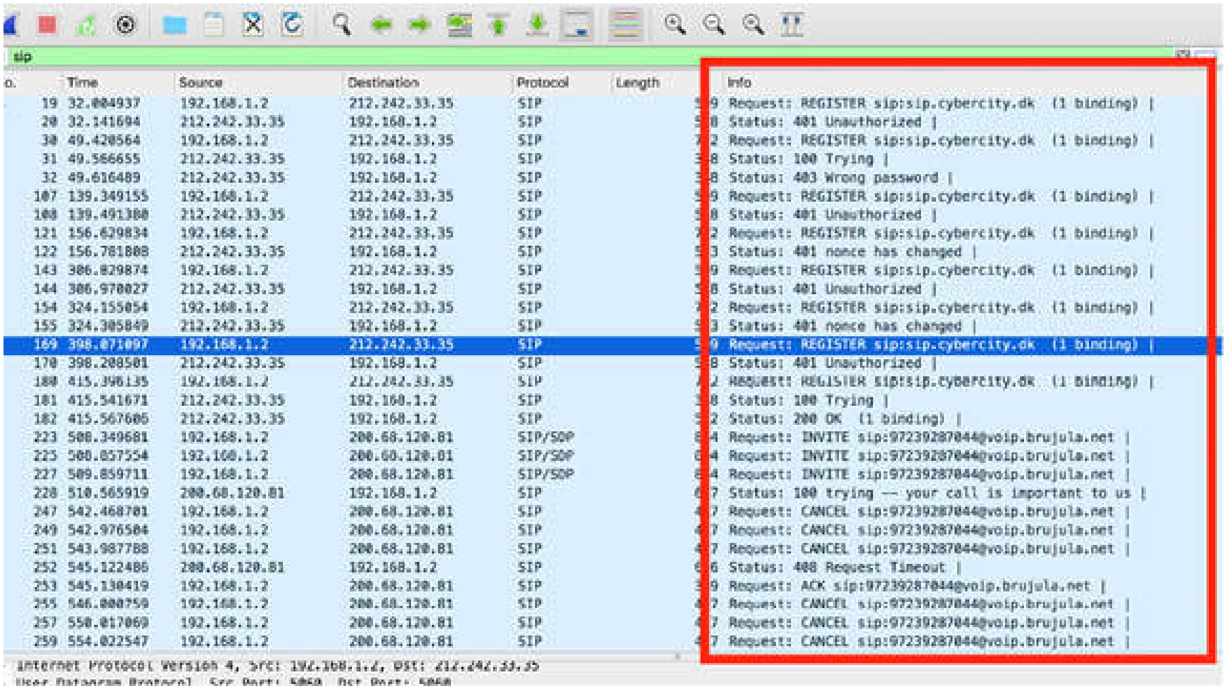
As shown in the figure below, the SDP media attribute section indicates that the RTP stream should run over UDP port 30000 (the *m* attribute) and the sender is offering the following (the *a* attribute):

- G.726 @ 8KHz (G726-32/8000)
- G.711 mu-law @ 8KHz (PCMU/8000)
- G.711 A-law @ 8Khz (PCMA/8000)
- Fmtp: 97



Task 2:

Wireshark defines the SIP commands and response codes in the Info column of the Packet List pane. The figure below shows a set of commands defined in SIP.



The following list describes the SIP commands:

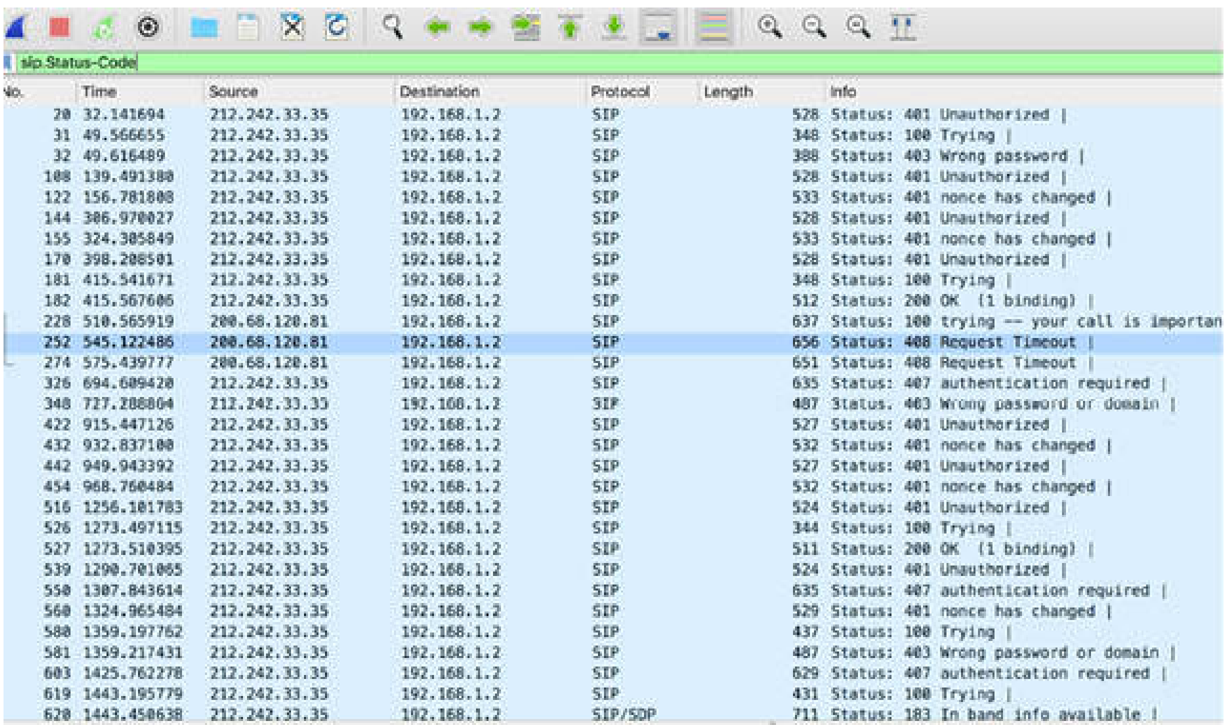
- INVITE: Invites a user to a call.
- ACK: Uses acknowledgment to facilitate reliable message exchange for INVITEs.
- BYE: Terminates a connection between users.
- CANCEL: Terminates a request, or search, for a user. It is used if a client sends an INVITE and then changes its decision to call the recipient.
- OPTIONS: Solicits information about a server's capabilities.
- SUBSCRIBE: Requests state change information regarding another host.
- REGISTER: Registers a user's current location.
- INFO: Used for mid-session signaling.

The SIP response codes are divided into the following six groups:

- 1xx: Provisional—request received, continuing to process the request

- 2xx: Success—the action was successfully received, understood, and accepted
- 3xx: Redirection—further action needs to be taken to complete the request
- 4xx: Client Error—the request contains bad syntax or cannot be fulfilled at this server
- 5xx: Server Error—the server failed to fulfill an apparently valid request
- 6xx: Global Failure—the request cannot be fulfilled at any server

In the filter toolbar, enter sip.Status-Code to display all captured response codes available in the pcap file, as shown in the figure below.

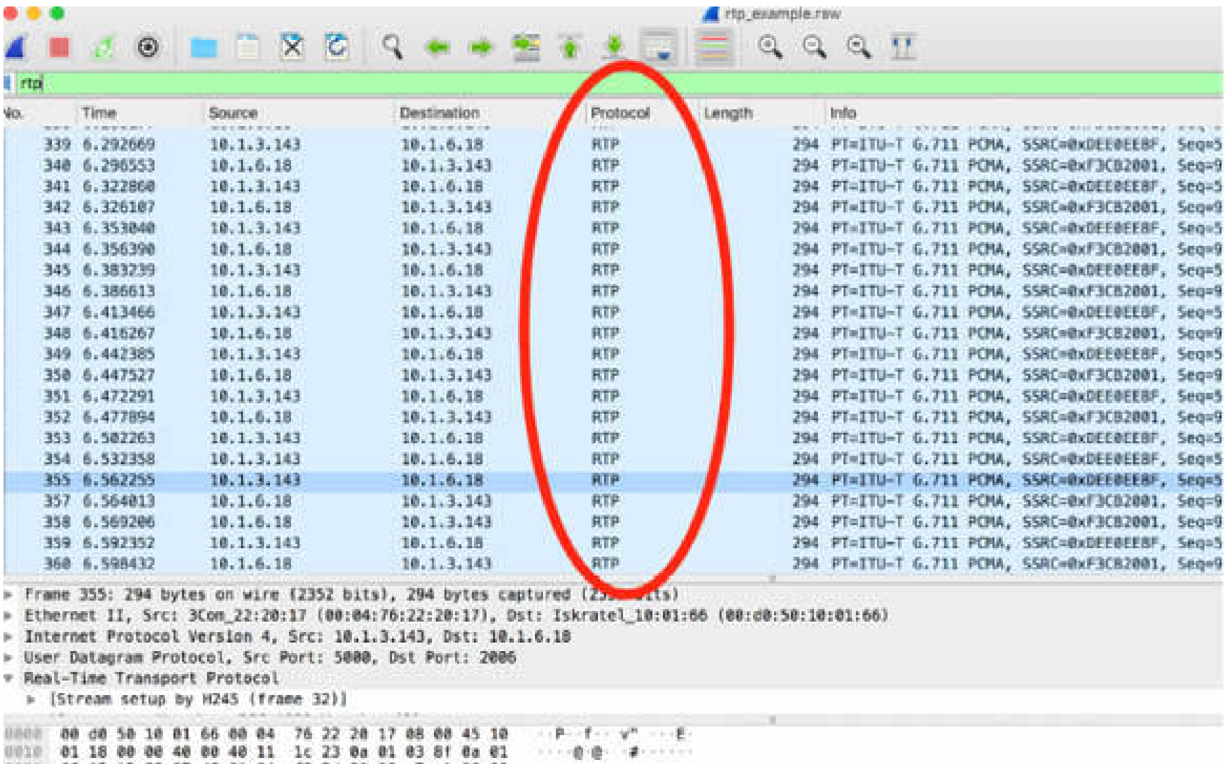


No.	Time	Source	Destination	Protocol	Length	Info
20	32.141694	212.242.33.35	192.168.1.2	SIP	528	Status: 401 Unauthorized |
31	49.566655	212.242.33.35	192.168.1.2	SIP	348	Status: 100 Trying |
32	49.616489	212.242.33.35	192.168.1.2	SIP	388	Status: 403 Wrong password |
108	139.491380	212.242.33.35	192.168.1.2	SIP	528	Status: 401 Unauthorized |
122	156.781808	212.242.33.35	192.168.1.2	SIP	533	Status: 401 nonce has changed |
144	306.970027	212.242.33.35	192.168.1.2	SIP	528	Status: 401 Unauthorized |
155	324.305849	212.242.33.35	192.168.1.2	SIP	533	Status: 401 nonce has changed |
170	398.208501	212.242.33.35	192.168.1.2	SIP	528	Status: 401 Unauthorized |
181	415.541671	212.242.33.35	192.168.1.2	SIP	348	Status: 100 Trying |
182	415.567606	212.242.33.35	192.168.1.2	SIP	512	Status: 200 OK (1 binding) |
228	510.565919	200.68.120.81	192.168.1.2	SIP	637	Status: 100 trying -- your call is importan
252	545.122486	200.68.120.81	192.168.1.2	SIP	656	Status: 408 Request Timeout |
274	575.439777	200.68.120.81	192.168.1.2	SIP	651	Status: 408 Request Timeout |
326	694.609420	212.242.33.35	192.168.1.2	SIP	635	Status: 407 authentication required |
348	727.208804	212.242.33.35	192.168.1.2	SIP	487	Status: 403 Wrong password or domain |
422	915.447126	212.242.33.35	192.168.1.2	SIP	527	Status: 401 Unauthorized |
432	932.837100	212.242.33.35	192.168.1.2	SIP	532	Status: 401 nonce has changed |
442	949.943392	212.242.33.35	192.168.1.2	SIP	527	Status: 401 Unauthorized |
454	968.760484	212.242.33.35	192.168.1.2	SIP	532	Status: 401 nonce has changed |
516	1256.101783	212.242.33.35	192.168.1.2	SIP	524	Status: 401 Unauthorized |
526	1273.497115	212.242.33.35	192.168.1.2	SIP	344	Status: 100 Trying |
527	1273.510395	212.242.33.35	192.168.1.2	SIP	511	Status: 200 OK (1 binding) |
539	1290.701065	212.242.33.35	192.168.1.2	SIP	524	Status: 401 Unauthorized |
550	1307.843614	212.242.33.35	192.168.1.2	SIP	635	Status: 407 authentication required |
560	1324.965484	212.242.33.35	192.168.1.2	SIP	529	Status: 401 nonce has changed |
580	1359.197762	212.242.33.35	192.168.1.2	SIP	437	Status: 100 Trying |
581	1359.217431	212.242.33.35	192.168.1.2	SIP	487	Status: 403 Wrong password or domain |
603	1425.762278	212.242.33.35	192.168.1.2	SIP	629	Status: 407 authentication required |
619	1443.195779	212.242.33.35	192.168.1.2	SIP	431	Status: 100 Trying |
620	1443.450638	212.242.33.35	192.168.1.2	SIP/SDP	711	Status: 183 In band info available |

Task 3:

Download the free sample capture file rtp_example.raw.gz from **https://wiki.wireshark.org/SampleCaptures** . Unpack it and open it in Wireshark.

In the filter toolbar, enter rtp to display only RTP packets, as shown in the figure below.



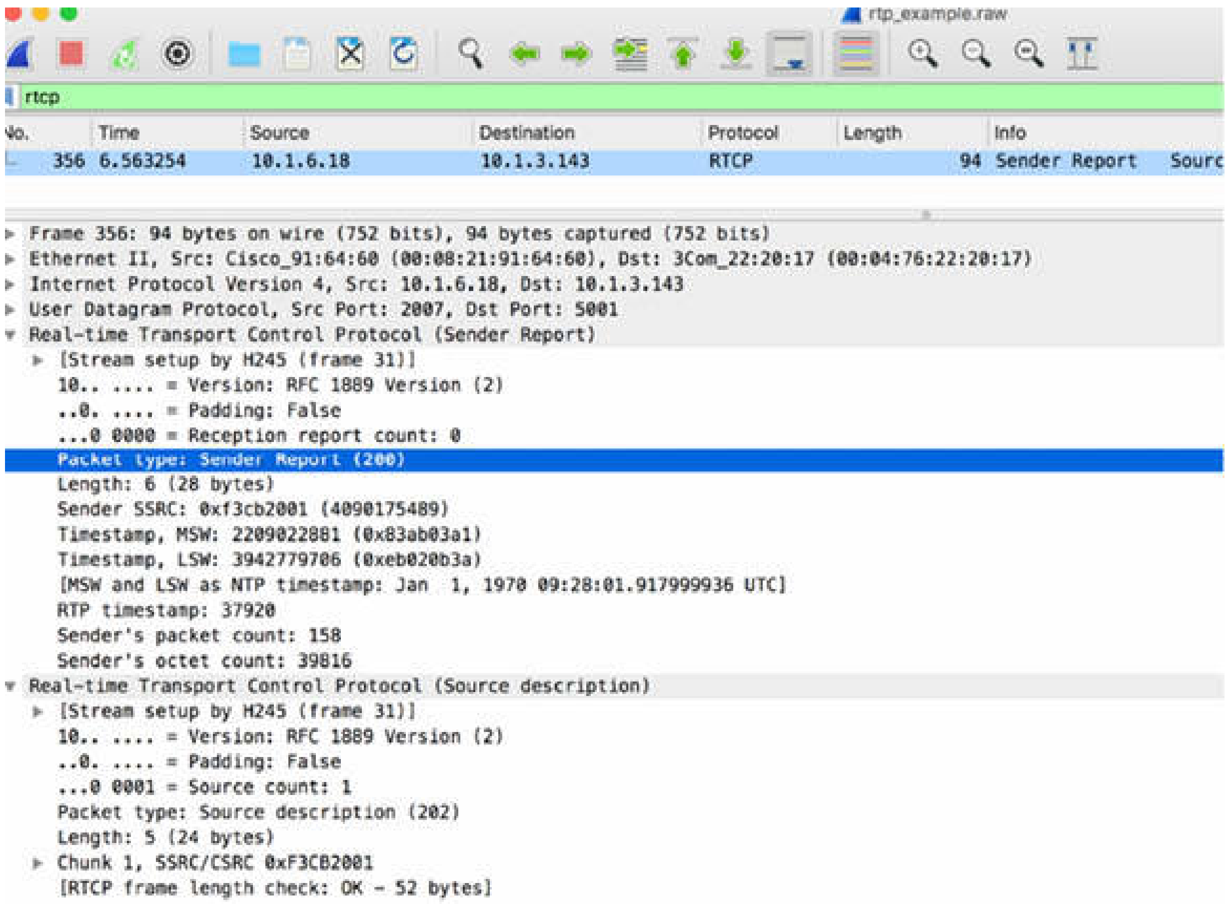
RTP provides end-to-end transport functions for real-time data such as audio, video, or simulation data over multicast or unicast network services. Realtime Transport Protocol (RTP) is defined in RFC 3550.

RTP is supplemented by a control protocol (RTCP) to allow monitoring of the RTP data delivery and to provide minimal control and identification functionality.

The five RTCP packet types are:

- SR (sender report) 200
- RR (receiver report) 201
- SDES (source description) 202
- BYE (goodbye) 203
- APP (application-defined) 204

In the filter toolbar, enter rtcp to display only RTCP packets. A single RTCP packet is displayed of the “Sender Report” packet type. You can also see it in the Packet Details pane, as shown in the figure below.



Notes:
Repeat the previous analysis on a different capture to gain confidence with the SIP and RTP traffic. Identify all available SIP packet types and response codes and all RTP/RTCP packet types.

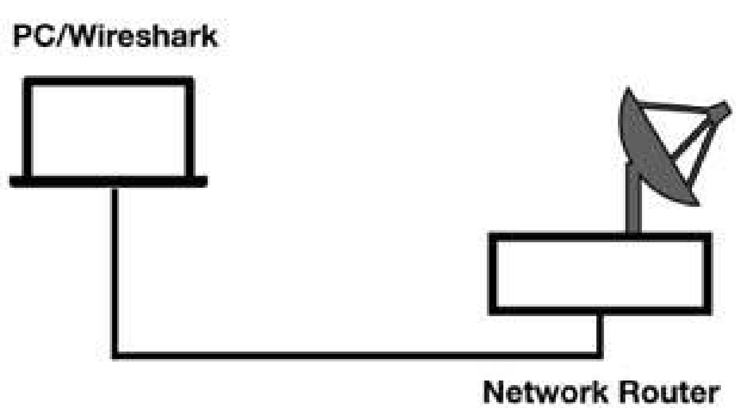
Lab 84. VoIP Playback

Lab Objective:
Learn how to play back a VoIP RTP stream.

Lab Purpose:
Learn how to use the Wireshark playback functionality to play back a VoIP stream.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. In this topology, a PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

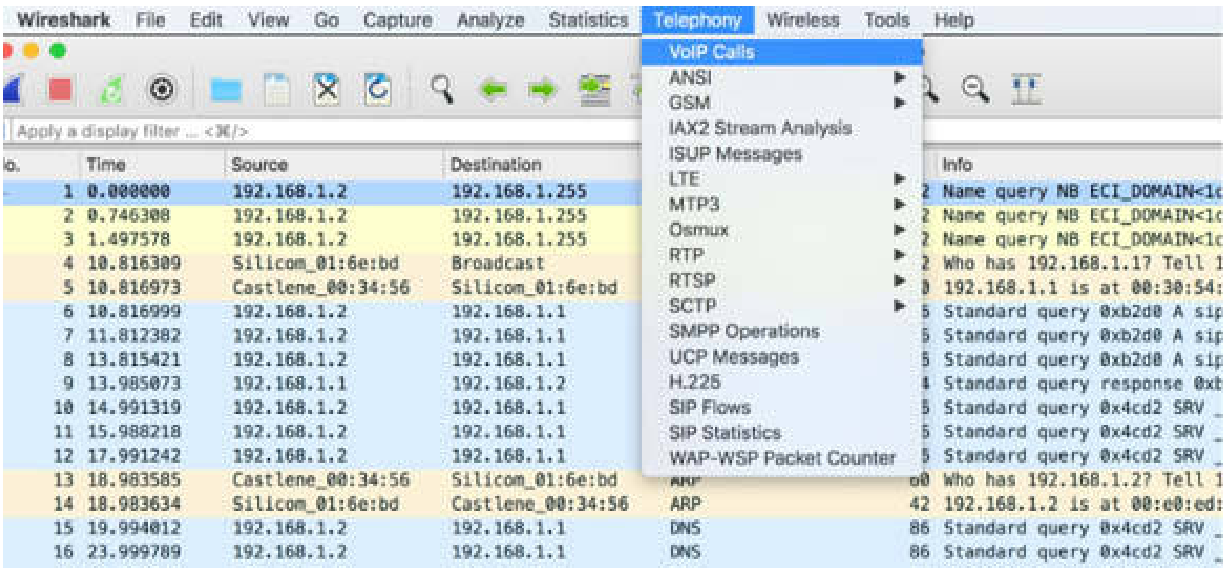


Lab Walkthrough:

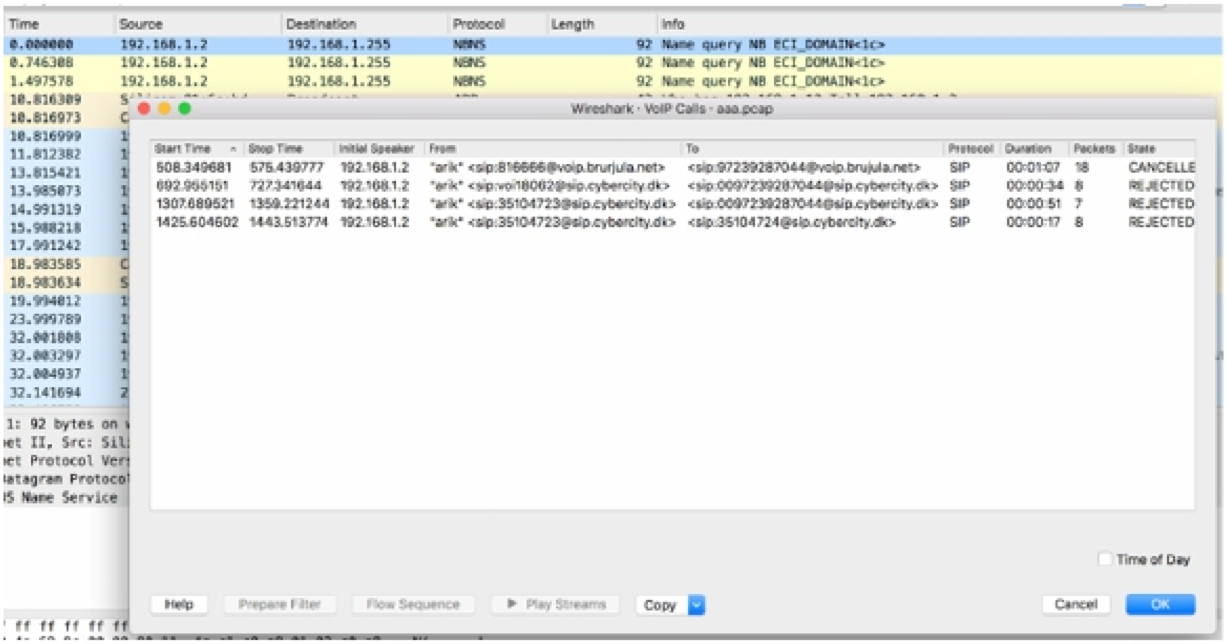
Task 1:

Download the free sample capture file aaa.pcap from **https://wiki.wireshark.org/SampleCaptures** and open it in Wireshark. This file contains sample SIP and RTP traffic.

To play back a VoIP RTP stream, on the main menu, select Telephony > VoIP Calls, as shown in the figure below.



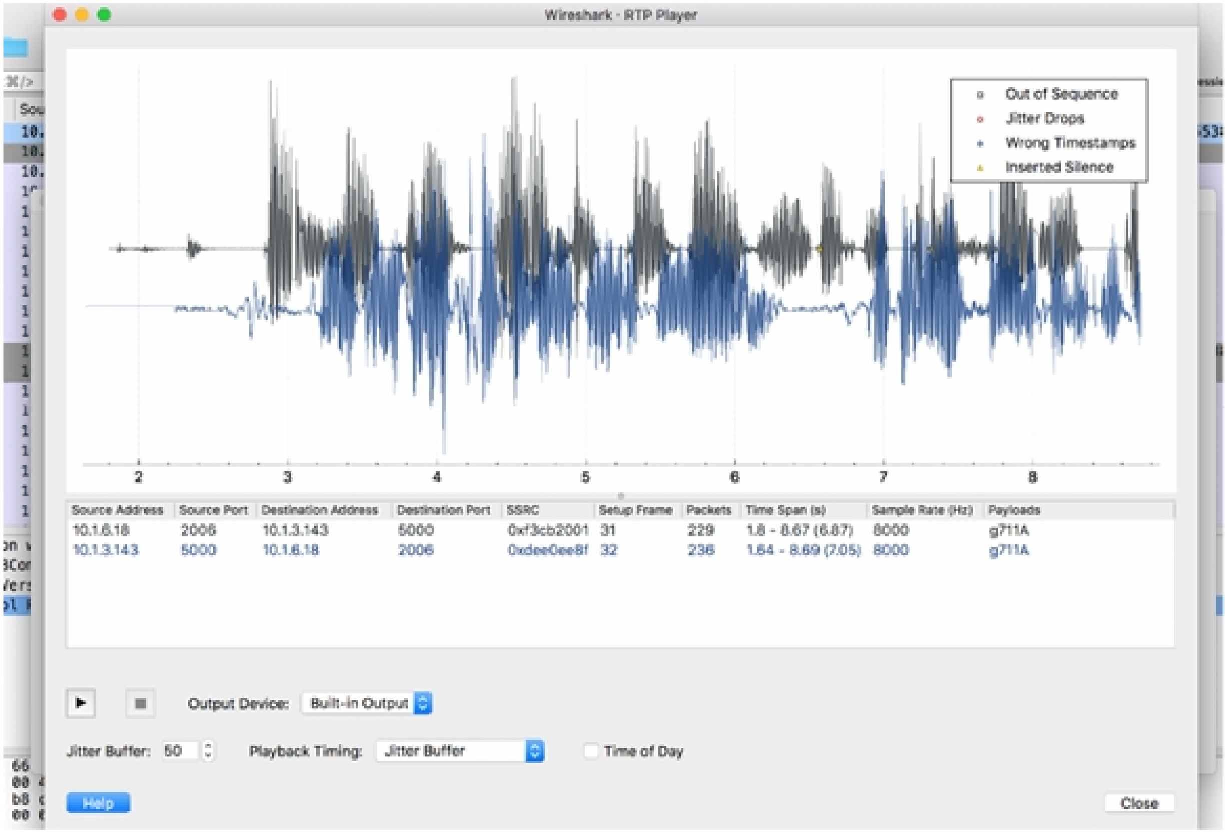
The VoIP Calls dialog box is displayed showing a list of calls, as shown in the figure below.



You can select a single call or multiple calls (by using the CTRL key). The "Play Streams" button is enabled after selecting a call.

Download the free sample capture file rtp_example.raw.gz (libpcap) from **https://wiki.wireshark.org/SampleCaptures** . This file contains a VoIP sample capture of an H323 call (including H225, H245, RTP, and RTCP). Unpack it and open it in Wireshark.

Play the contained stream by using the VoIP Calls dialog box, as described earlier. The RTP Player dialog box is displayed.

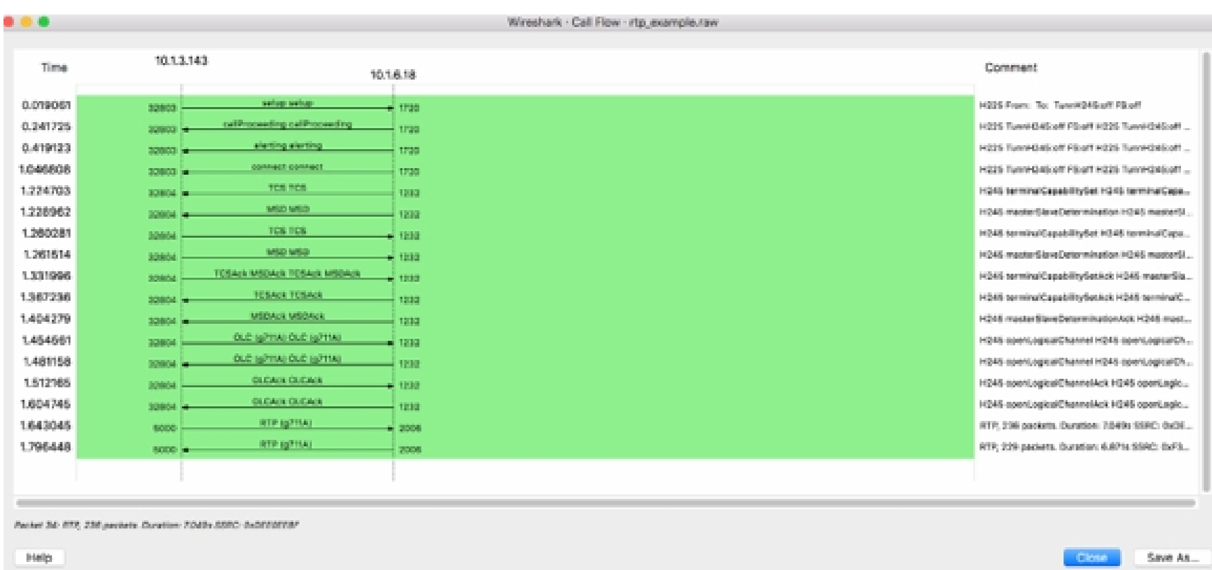


The RTP Player contains a playback graph for each stream detected. In the figure above, you can see two playback graphs—one for each direction of the call. Below the graphs, you can observe the information related to the source/destination address and port, the number of packets, the sample rate, and other useful information.

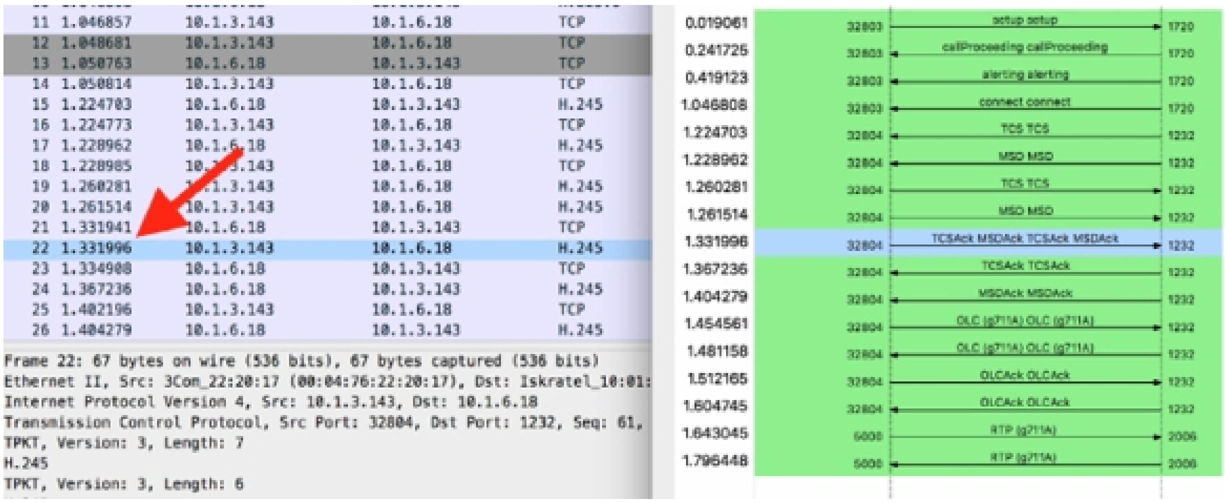
To get the actual time, select the “Time of Day” check box. To listen to the VoIP call, click the Play button.

If Wireshark indicates errors in the VoIP call, you will be able to hear the quality issues of these errors during playback. In the figure below, the “Time of Day” check box is selected and the symbols indicating a set of errors (Out of sequence, Jitter Drops, Wrong timestamp, and Inserted Silence) are highlighted by using arrows.

To view the call flow in chronological order, go back to the VoIP Call dialog box, and click the “Flow sequence” button. The Call Flow sequence related to the call above is displayed in the Call Flow dialog box, as shown in the figure below.



On the left side, the timeline is displayed which is followed by the source and destination addresses. Each message of the flow is displayed by using an arrow. Selecting a message and clicking on it displays the related packet in the Packet List pane, as shown in the figure below.

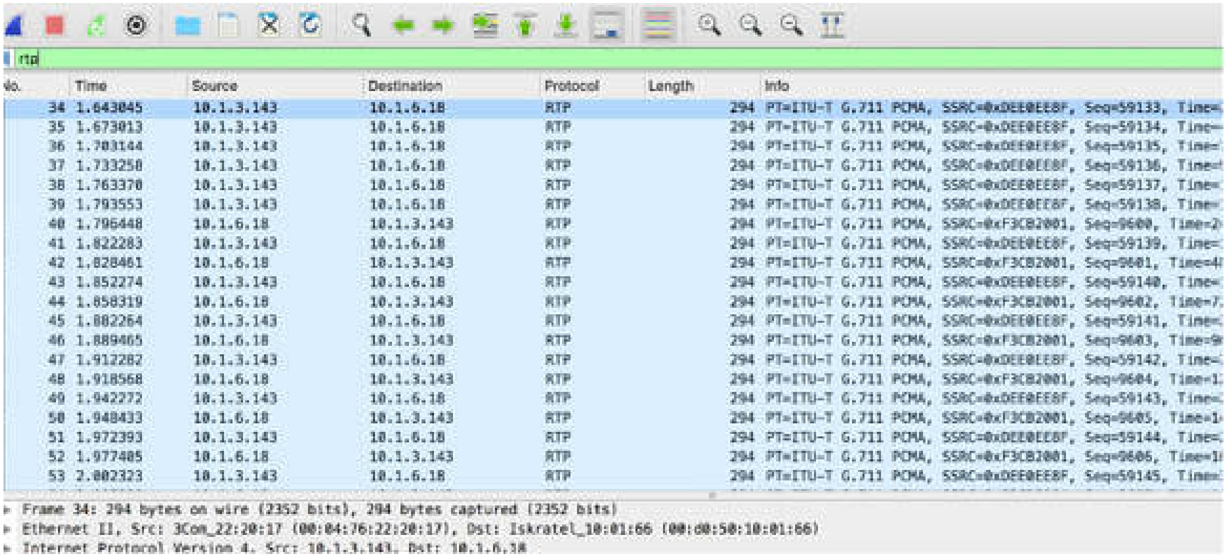


Task 2:

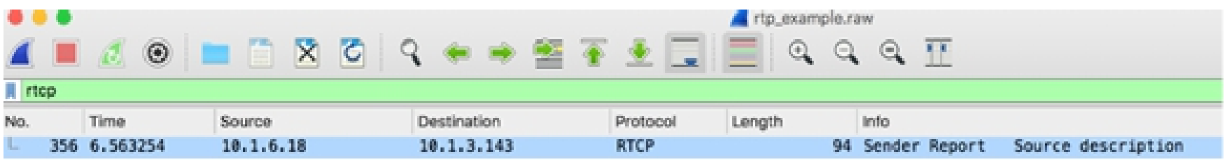
Creating a VoIP profile is useful for VoIP analysis and troubleshooting. For efficiency, associate the necessary elements to the profile. Create a profile using as elements the Time display format as "Seconds Since Previous Displayed Packet", a column for "Differentiated Services Code Point", select a color for SIP resends and a different color for SIP response codes greater than 399.

During VoIP analysis, it is important to select appropriate filters. For capture filters, remember that the filter can be built only on the ports used (for example, udp.port), even when the used port is not known. If this is the case, you can capture all the UDP traffic and then apply some VoIP display filters. Some of the most used display filters are:

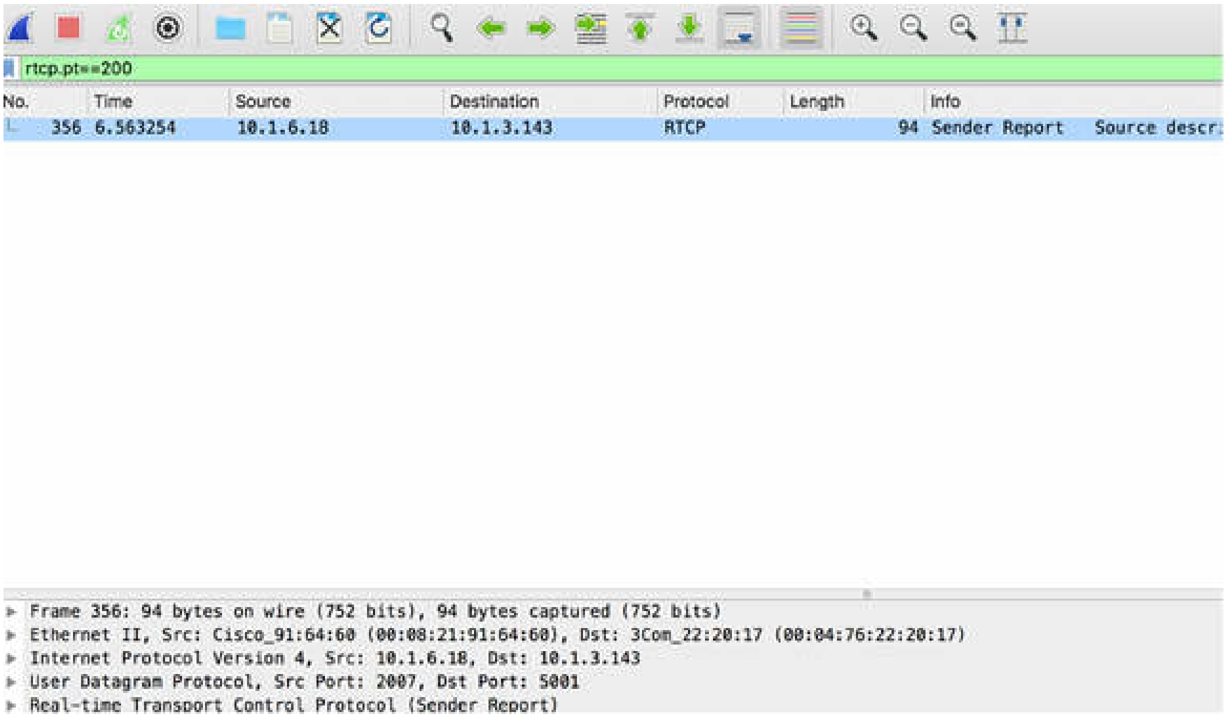
- sip —For SIP traffic only
- rtp —For RTP traffic only



- rtcp —For Realtime Transport Control Protocol



- sip.Method==”INVITE” —For SIP Invites
- sip.Method==”BYE” —For SIP Connection closings
- sip.resend==1 —For detecting when a SIP packet had to be resent
- rtcp.pt==200 —For RTCP sender report



Notes:

Repeat the previous steps on a different VoIP call capture to understand the differences in the features extracted by Wireshark from the calls. Try to generate the related graphs and to identify at what time each feature or error appeared. Gain the necessary confidence in doing the VoIP stream analysis.

Network Baselines and Security

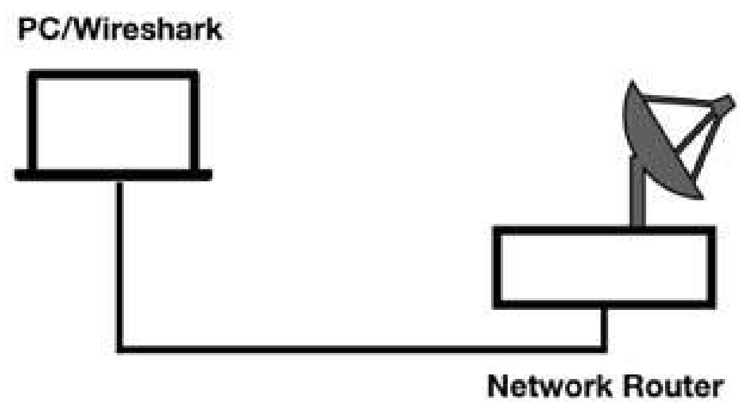
Lab 85. Baseline Traffic Pattern (Broadcast/Multicast, Protocols/Applications)

Lab Objective:
Learn what baselining is and why it's important.

Lab Purpose:
Understand the importance of the process of baselining during communication analysis.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
Baselining is the process of creating a set of trace files that depict normal communications on the network. A baseline can consist of more than one trace file, can include screenshots taken from a client/server, and can be created by gathering summary data, I/O graph information, and network maps.

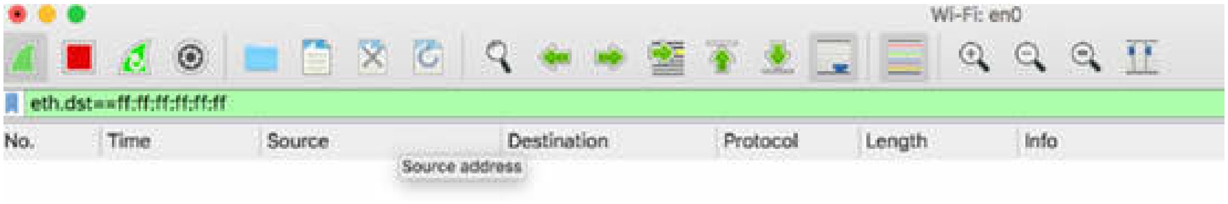
In general, it is important to create a baseline before network problems or before the occurrence of security breaches. That's because, in this way, the analysis process can speed up. Baselining enables you to resolve problems more effectively and efficiently.

Baselining is very useful for identifying the normal traffic patterns during the analysis of a problem. For example, if a user complains about the performance experienced on a particular day, you can take the trace file of the current traffic. Referring back to the baseline trace file, you can filter out the normal traffic and focus on the unusual traffic. This can significantly reduce the troubleshooting time and make the effort cost-effective.

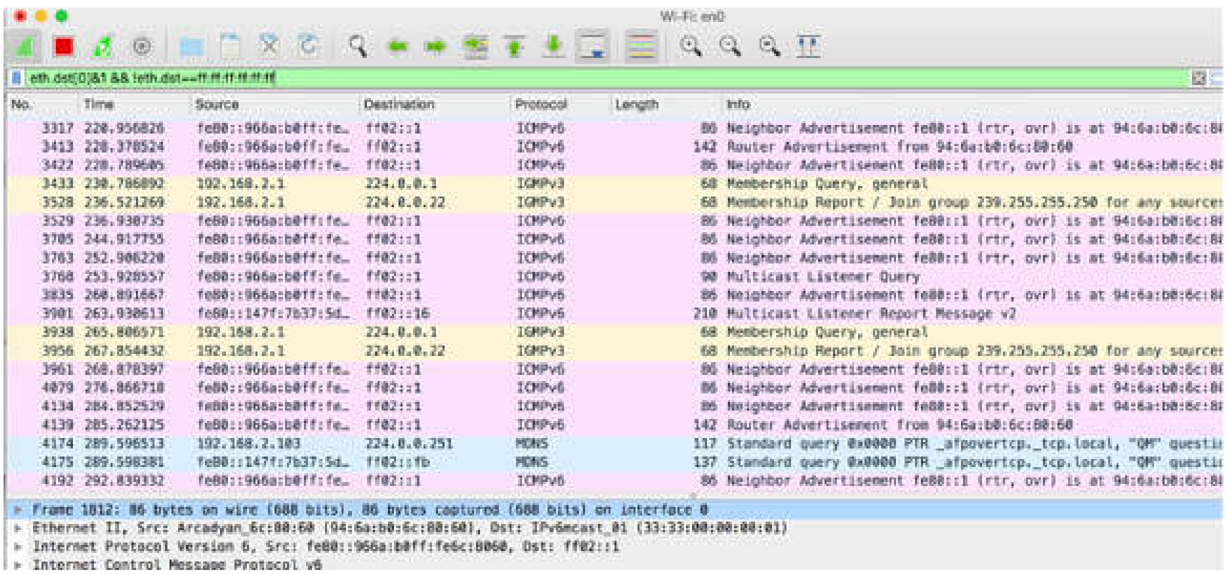
Another use of baseline is when a security breach situation occurs. In such a case, as you already know the normal protocols, applications, and traffic patterns, you can spot unusual communications. For example, if the hosts you are observing never use Internet Relay Chat but suddenly this type of traffic appears in the trace log, it could indicate a bot infection.

Task 2:
Open Wireshark, and on the main menu, select Capture > Options. Select an interface for which the line graph displays some activity in the Traffic column, and capture the traffic for a few minutes. Stop the capture and save the file.

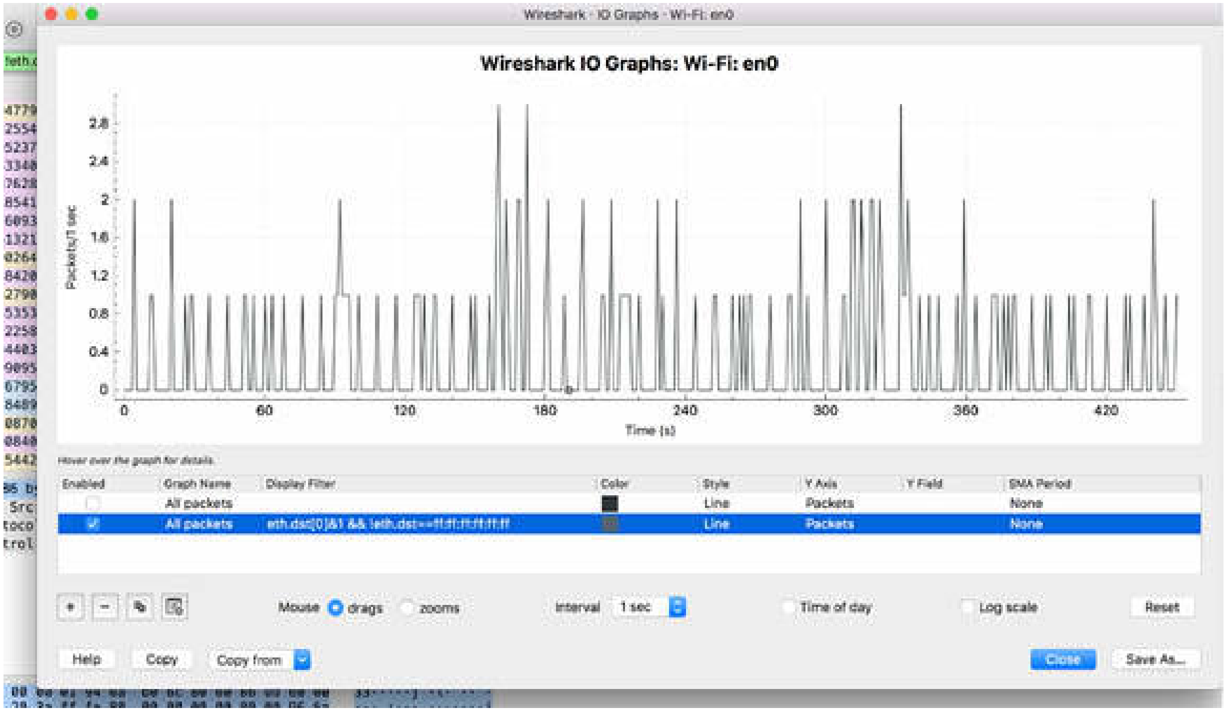
To baseline broadcast traffic, in the filter toolbar, enter eth.dst==ff:ff:ff:ff:ff:ff , as shown in the figure below (there is, currently, no broadcast traffic).



To baseline the multicast traffic, in the filter toolbar, enter eth.dst[0]&1 && !eth.dst==ff:ff:ff:ff:ff:ff , as shown in the figure below.



As shown in the figure above, there is a lot of multicast traffic, and, in particular, ICMP traffic. By observing this traffic, you can determine which rate of the traffic selected is presently generating the I/O Graph for the selected display filter, as shown in the figure below.

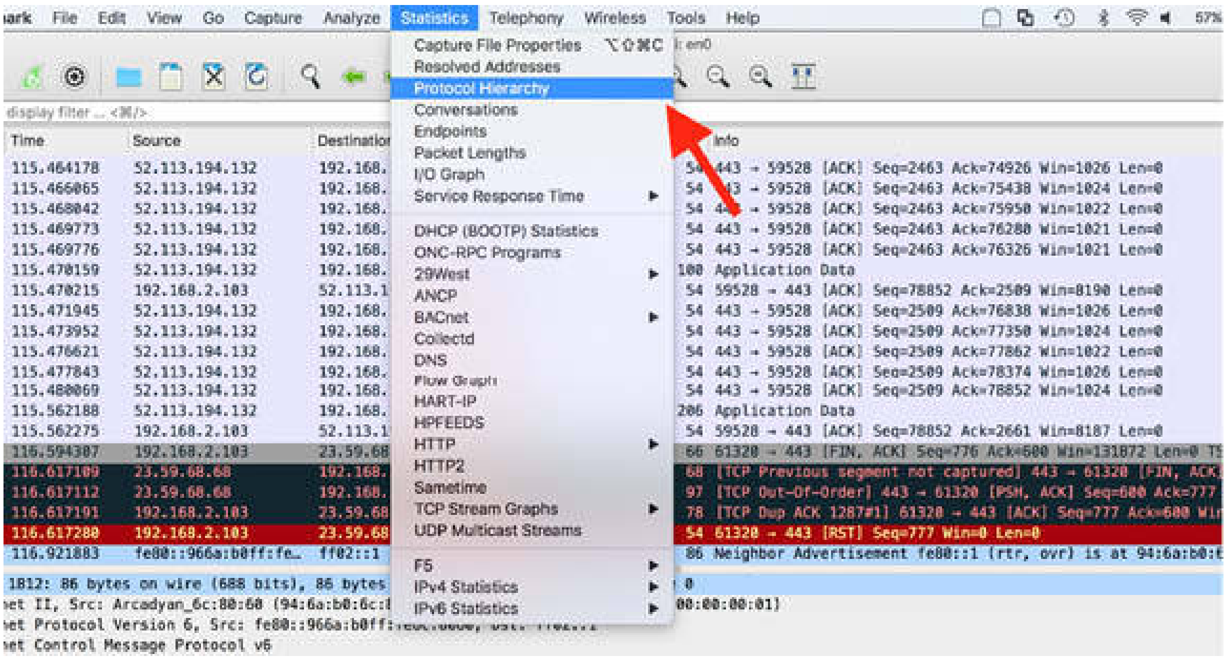


When baselining such types of traffic, you need to identify who is transmitting broadcast/multicast traffic and from which applications. The I/O graph provides the typical rate in packets per second. This is required for identifying an increase or decrease in the traffic in a future captures.

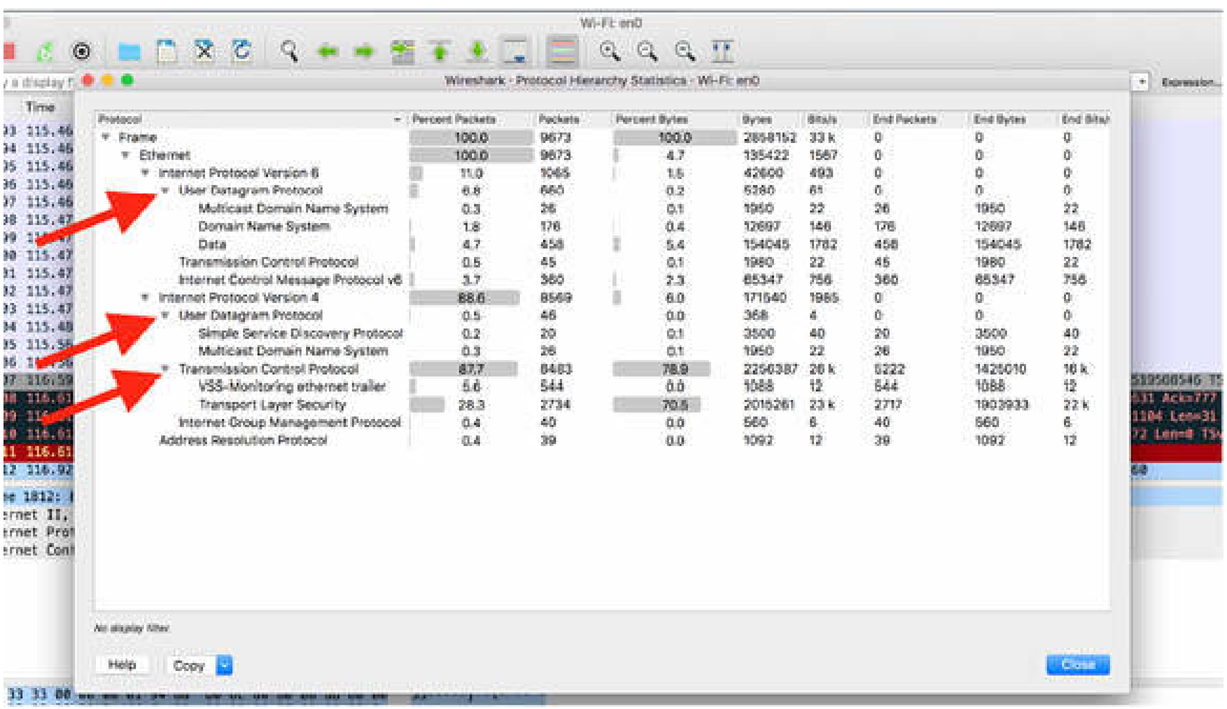
Task 3:

To baseline the protocols and applications for identifying breached hosts, compare the current traffic with a previously saved baseline.

To create the protocols and applications baseline, capture the traffic for a few minutes on an active connection. On the main menu, select Statistics > Protocol Hierarchy, as shown in the figure below.



The Protocol Hierarchy Statistics dialog box is displayed. It shows the statistics in percentage and bytes for each protocol, as shown in the figure below.



When baselining such types of traffic, you need to identify which applications are running on the network and which protocols are being used. If the applications use TCP, identify the TCP ports. Similarly, for UDP ports, identify the UDP ports.

You also need to identify which routing protocol is used and determine the characteristics of the routing update protocol to compare with later traffic acquisitions.

Notes:
Repeat the previous steps on a different network traffic capture to create a baseline for the broadcast and multicast traffic types. Create another baseline for protocols and applications. Gain the necessary confidence in using the utilities provided by Wireshark to determine a well-determined baseline.

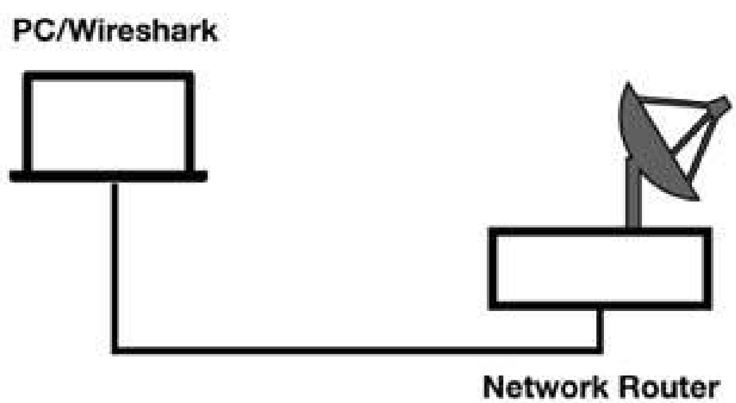
Lab 86. Baseline Traffic Pattern (Bootup—VoIP)

Lab Objective:
Learn what is the process for baselining bootup sequences.

Lab Purpose:
Understand how to implement bootup sequences baselining.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:

Analyzing the bootup sequence is important considering that this sequence sets up the client's general configuration and performance for the normal functioning uptime.

Having a bootup baseline can help you in identifying the performance changes. Doing a periodic bootup check is recommended to understand the startup process in detail (if the parameter request is involved) and list the applications involved in the initial process. When a different bootup process is implemented in a network, a comparison with a previous bootup baseline can help in discovering the changes in the performance.

The baselining bootup process cannot be captured by running Wireshark on the host currently booting. You need to tap into the existing network connection close to the booting client, and then start capturing when the host boots up.

For baselining the bootup process, tap into the network connection under analysis and start capturing in Wireshark. After that, boot up the host. Take into account the following two main things:

- What happens during the startup sequence? Is there any parameters exchange during the DHCP process?
- Which applications are generating traffic during the startup process?

Answering these questions can give you a clearer picture of the performance of the booting process.

Task 2:
Every time a new configuration is deployed on the network, you should baseline the login sequence. This process should also be repeated in the lab environment before deployment. During this process, you should identify normal patterns and acceptable behaviors.

To tap into an existing network connection (as close to the client as possible), start capturing in Wireshark, and then log in to the network from the baseline host. When done, check for the following:

- What discovery process takes place during login?
- Which server does the client connect to?
- Which processes are seen during login?
- How many packets a typical login requires?
- Are there any login dependencies?

Task 3:
The baseline procedure should also be repeated during idle times. Idle times are defined as the times when no one is using the host. Baselining during idle times helps in identifying the background traffic that automatically occurs and that is generated by the applications loaded on a host.

In Wireshark, capture the traffic for a few minutes, and answer the following:

- Which protocols or applications are seen during idle time?
- Which hosts are contacted? Try to identify the IP address and/or host name.
- How frequently does the idle traffic occur? Note down the packet rate.
- What are the signatures of this traffic that you can filter out when removing this traffic from a trace file?

Task 4:
Baselining of application sequences and key tasks is also very important. The analysis of the application sequences helps you in identifying the interdependencies and the general ports and startup procedures used. Key tasks help you in learning about how they work and what their typical response times are.

In Wireshark, capture the traffic for a few minutes, and identify the following:

- Which discovery process is the application dependent upon?
- Is the application TCP-based or UDP-based?

- (For TCP) Which TCP options are set in the handshake packets?
- Which ports are used in the application?
- Which hosts are contacted when the application starts?
- How many packets and how much time until the launch is complete?
- What is the IO rate during the launch sequence?
- What happens during application idle time?
- Are any portions of the login visible in clear text?
- What is the round trip latency during the application launch?
- Are there any failures or retries during the application launch?
- Are there any server delays in receiving the requests?
- Are there any client delays preceding the requests?
- Are there lost packets, retransmissions, or out-of-order packets during the launch?

Answering these questions should create a clear picture to reuse the baseline during different sessions.

Task 5:
To identify the typical behavior and latency times related to the most popular web hosts, you can create a baseline of web browsing sessions.

In Wireshark, capture the traffic for a few minutes, and answer the following:

- Which browser was used?
- What is the target for the name resolution process?
- What is the name resolution response time?
- What is the round trip latency time between the client and the target server?
- What is the application response time for a page request?
- Did you communicate with other hosts during the web browsing session?
- Are there any HTTP errors in the trace file?

Task 6:
The name resolution process has a significant impact on performance. For such analysis, compare the baseline of this process against future trace files. This comparison can identify some possible performance issues.

In Wireshark, capture the traffic for a few minutes, and answer the following:

- Which application is being used to test the name resolution process?
- Which name and type are being resolved?
- What is the IP address of the target name server?
- What is the round trip response time for the name resolution process?

Task 7:
To spot performance issues during their occurrence, perform a baseline of the throughput tests. Capture the trace file during the test to graph the IO rate, and answer the following:

- Which application is being used to perform the throughput test?
- What are the configurations of host 1 and host 2?
- What packet size is used for the throughput test?
- Which transport was used for the test?
- What is the Kbytes per second rate from host 1 to host 2?
- What is the Kbytes per second rate from host 2 to host 1?
- What was the packet loss rate in each direction?
- What was the latency (if measurable) in each direction?

Save the I/O graph from throughput tests.

Task 8:
Creating a baseline of the Wireless connectivity is important in those cases where the network is already in place. This helps in identifying the problems that can be possibly solved at a later date.

Analyze the following key points:

- Location of the packet capture point
- Types of packets that are involved with the connection establishment to the access point
- Identify if an encryption method is used
- Identify if there were WLAN retries, filtering on the retry bit
- Identify the beacon rate by using the I/O graph with the beacon filter

Copy and save the statistics and WLAN traffic baseline information.

Task 9:
Understanding the basic VoIP traffic patterns (including call setup and actual call processes) speeds up the comparative analysis at a later date.

In Wireshark, capture the traffic for a few minutes, and focus on the jitter rate, packet loss rate, and call setup procedures.

Create a complete picture of the captured traffic description by answering the following:

- Which type of protocol is used for the call setup procedure?
- What is the round trip latency time for the call setup procedure?
- What is the average call setup time for Telephony and SIP?
- Which codec is used for compressing the payload?
- Did Wireshark detect VoIP calls in the trace file?
- Are there any SIP error responses?
- What is the jitter rate?
- Is there any packet loss in the communication (Telephony, RTP, or Stream Analysis)?

Notes:
Repeat the previous steps for creating a baseline for different purposes on different networks. Compare baselines created on different days to understand changes in performance and to gain more confidence in analysing baselines.

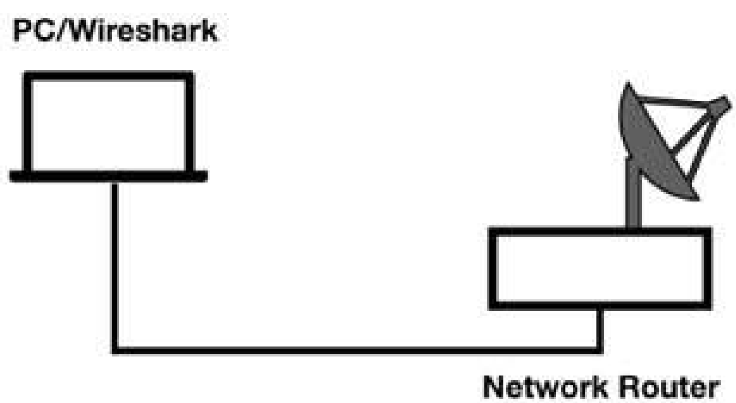
Lab 87. Troubleshoot Performance Problems

Lab Objective:
Learn about various performance problems.

Lab Purpose:
Understand how to detect and troubleshoot performance problems.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:

One of the most popular performance troubleshooting methodologies is to begin at the physical layer and move up to the application layer in an OSI model bottom-up order.

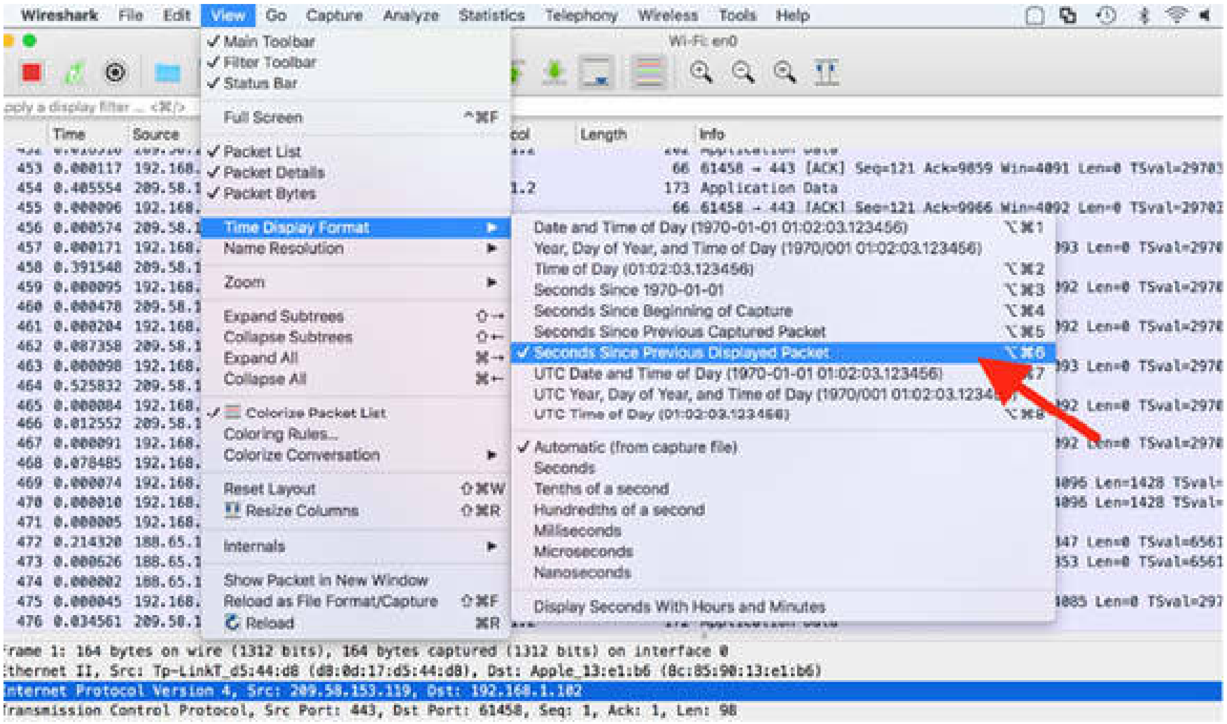
Usually, when you observe slow application loading time, slow file transfer time, or inability to connect to specific services, you suspect that there are some performance issues. Some of the cases that you can encounter during performance analysis are:

- DNS problems may prevent a host from obtaining the IP address of a target host.
- Incorrect subnet mask values may cause a host to perform discovery for a local host that is, in fact, remote.
- Incorrect route table values or unavailable gateways may isolate a host.

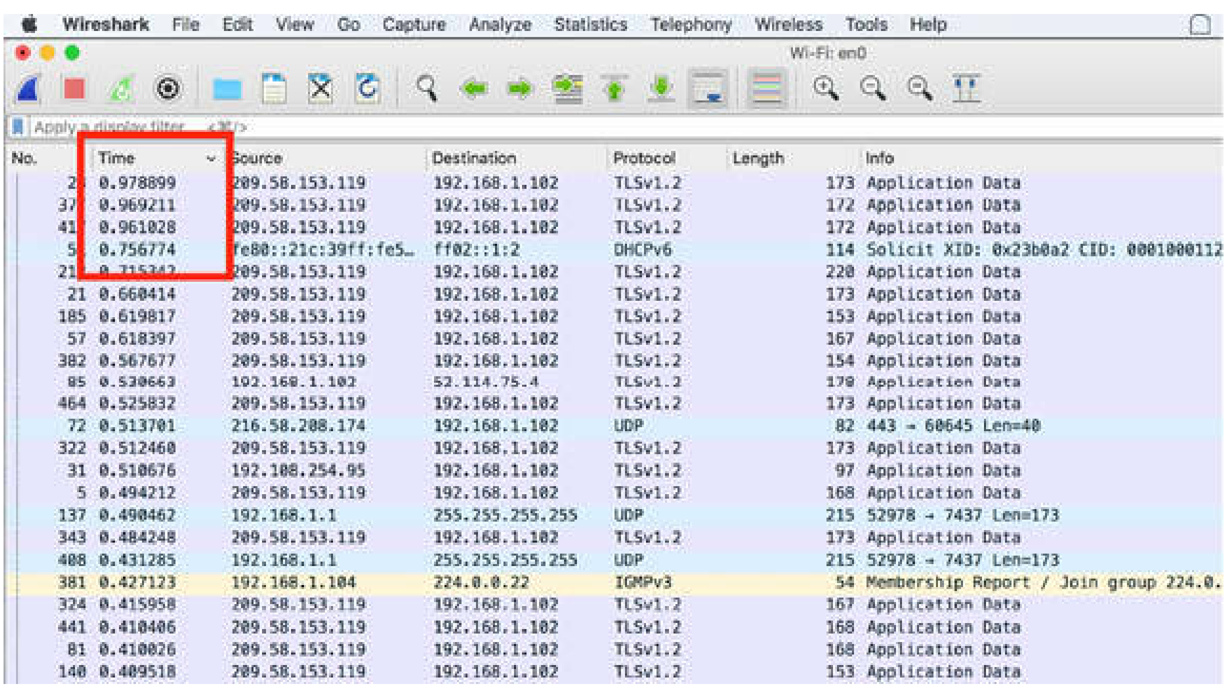
To immediately identify the source of the problem and solve it, take the baseline of normal network communications and compare it to the baseline of faulty communications to locate differences.

Task 2:
High latency times can be caused by distance, queuing delays along a path, processing delays, etc.

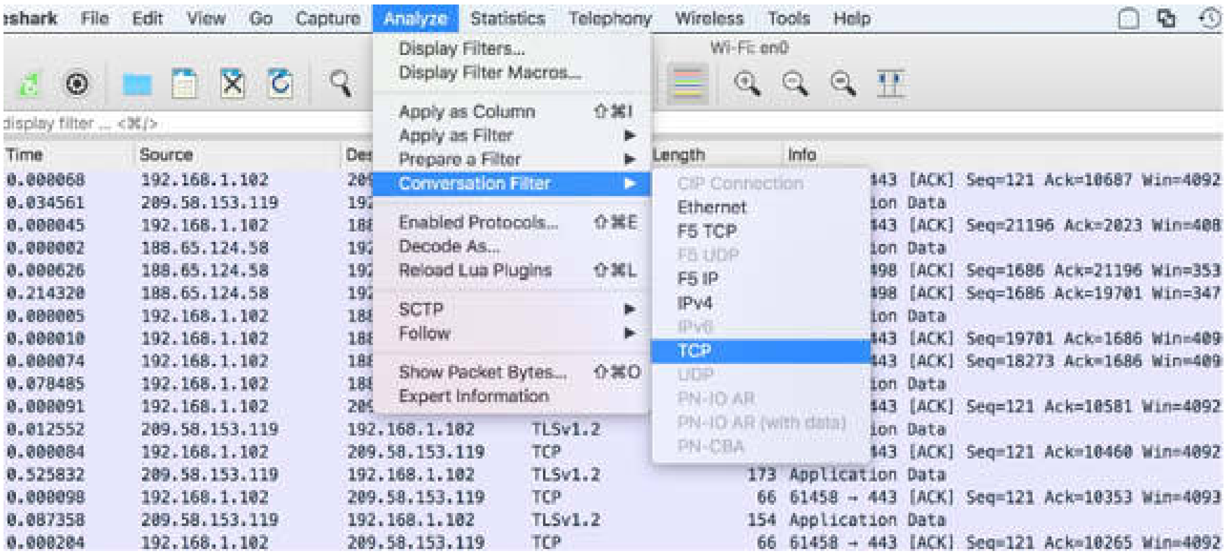
In Wireshark, capture the traffic for a few minutes on the active network interface, and save the file. On the main menu, select View > Time Display Format > Seconds Since Previous Displayed Packet, as shown in the figure below.



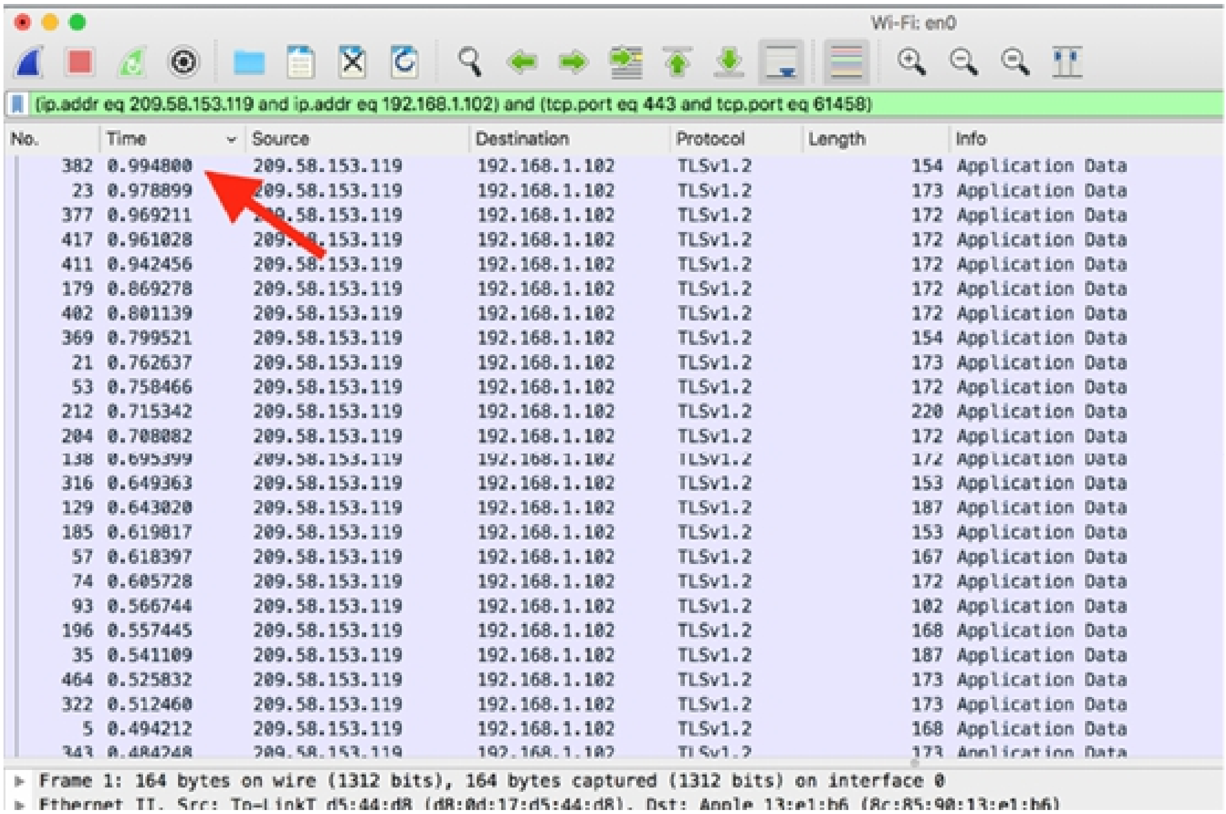
In the Packet List pane, click on the Time column to sort this column. Note the large gaps in time between packets in the capture file, as shown in the figure below. The maximum gap is shown as 0.97s.



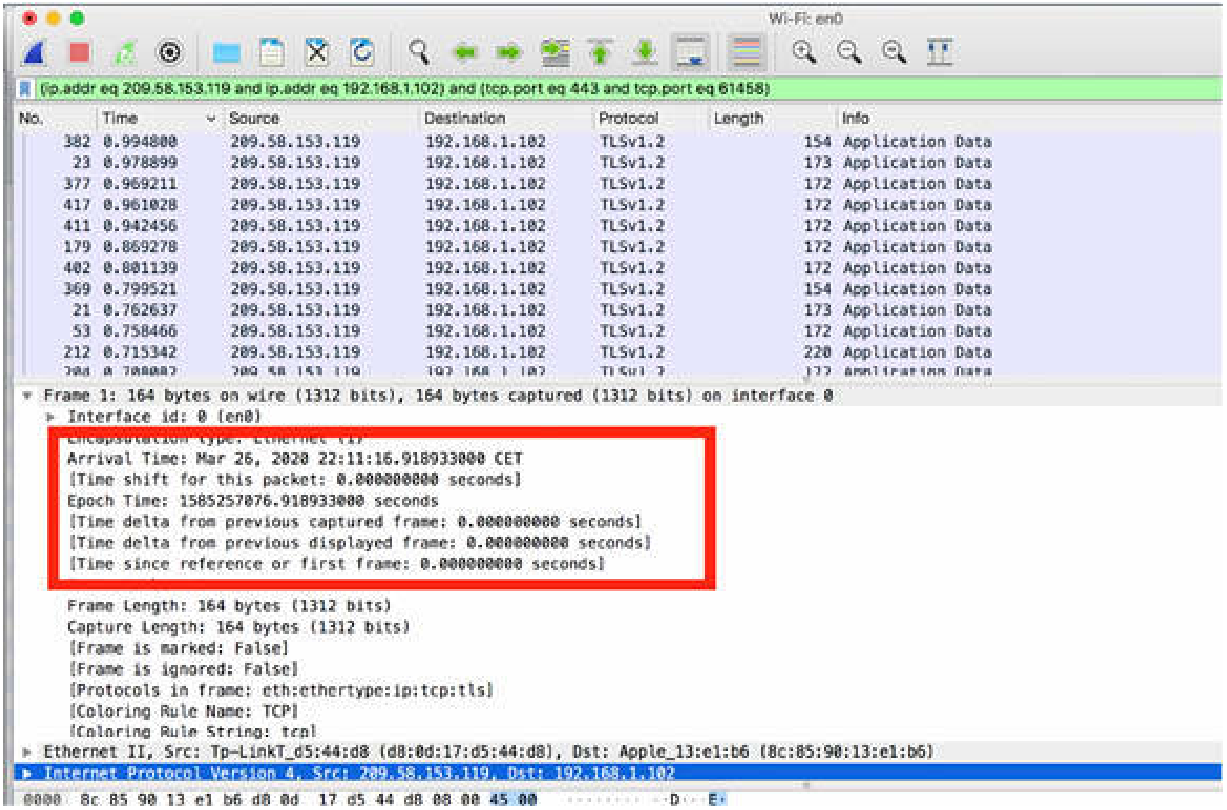
If the capture file contains numerous conversations, then before sorting the Time column, filter a conversation to ensure that you are comparing times within a single conversation. To do so, on the main menu, select Analyze > Conversation Filter, as shown in the figure below.



If you select a TCP conversation, the maximum gap is 0.99s, as shown in the figure below.



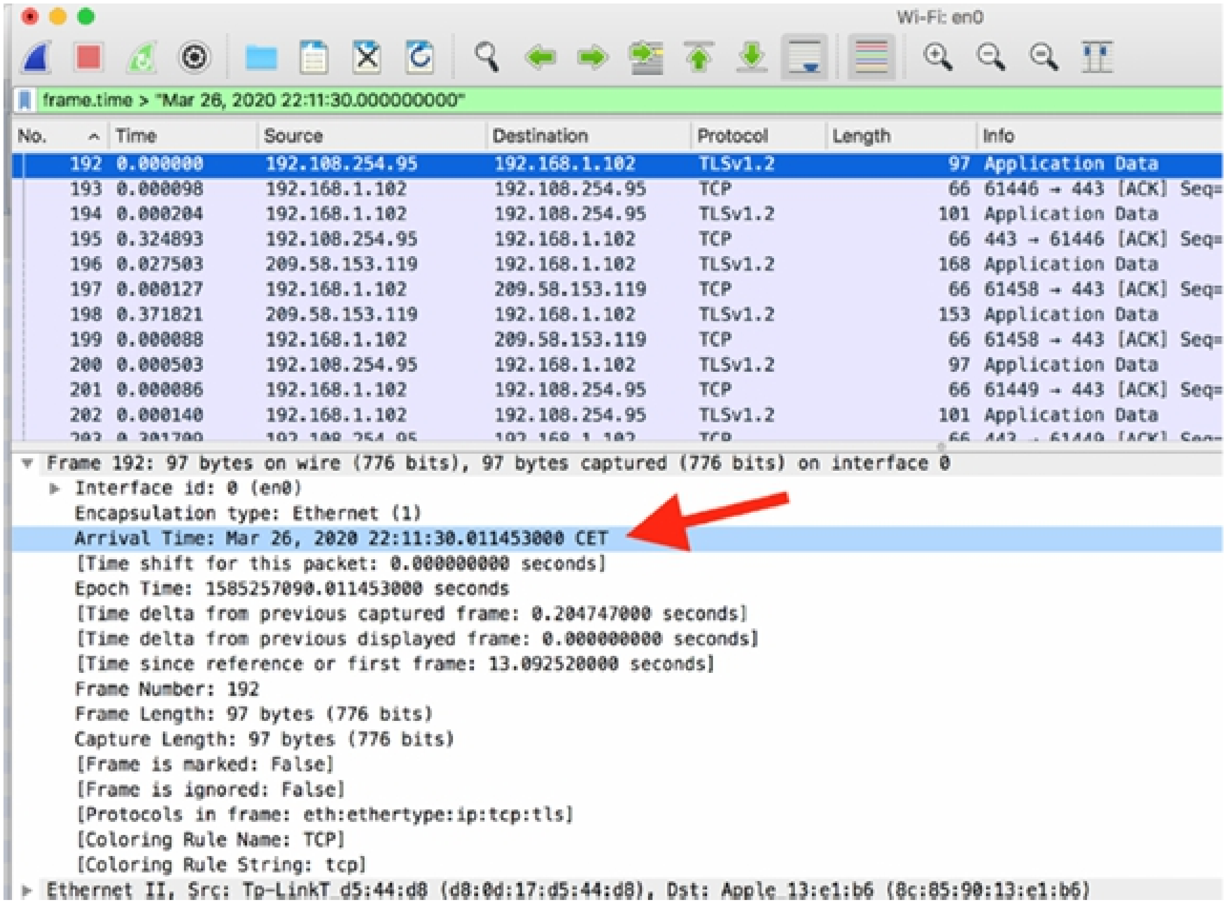
You can also add a new column by using the Preference dialog box, where you can select an additional delta time. Another option is to inspect the packet content in the Packet Details pane, as shown in the figure below.



You can see numerous time values inside the Frame field. Although these values are not the actual fields in the packet, Wireshark can find packets based on these values.

Packet timestamps are provided by the WinPcap, libpcap, or AirPcap libraries (supporting microsecond resolution) at the time of capturing the packet. These packet timestamps are saved with the capture file.

Task 3:
You can also filter packets based on the arrival time. The Arrival Time value is based on the system time at the time of packet capture. For example, to filter out the packets arrived after 10:11:30 PM on March 26, in the filter toolbar, enter frame.time > "Mar 26, 2020 22:11:30.000000000, as shown in the figure below.



Notes:

To identify time gaps in the network capture, repeat the previous steps for filtering with timestamps and delta time between packets. Gain confidence in filtering conversation times and again capture traffic to test different approaches based on the arrival time of the packets.

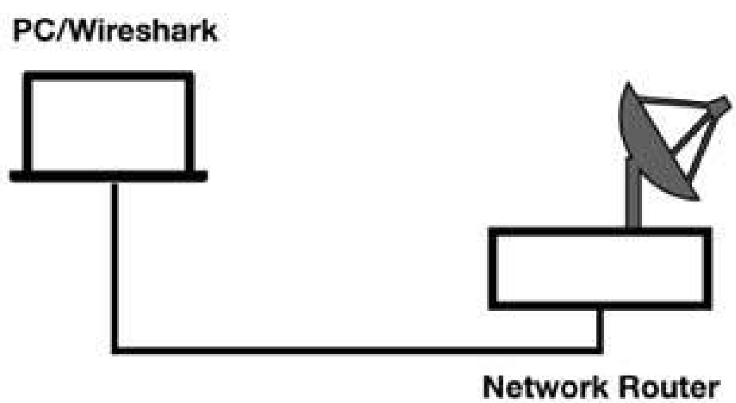
Lab 88. Slow Processing Time

Lab Objective:
Learn about the problems related to slow processing time.

Lab Purpose:
Understand how to detect and troubleshoot slow processing time issues.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

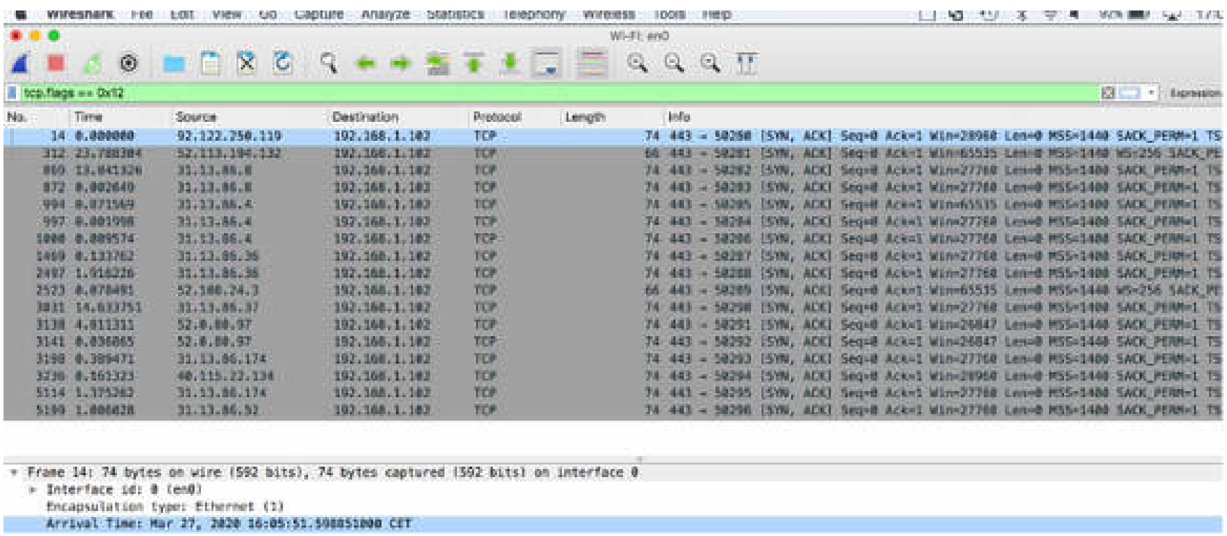
Task 1:
When a host doesn't have sufficient processing power or memory, or an application does not respond in a timely manner, you may see gaps in the

response times between the requests and replies.

These gaps may be accompanied by other indicators of the problem, such as a TCP window of size zero or a TCP window size smaller than the TCP MSS value. Alternatively, application responses may indicate an overloaded condition. Consider reassembling streams to decipher any plain text messages (if they exist). These messages may clearly define the application problem.

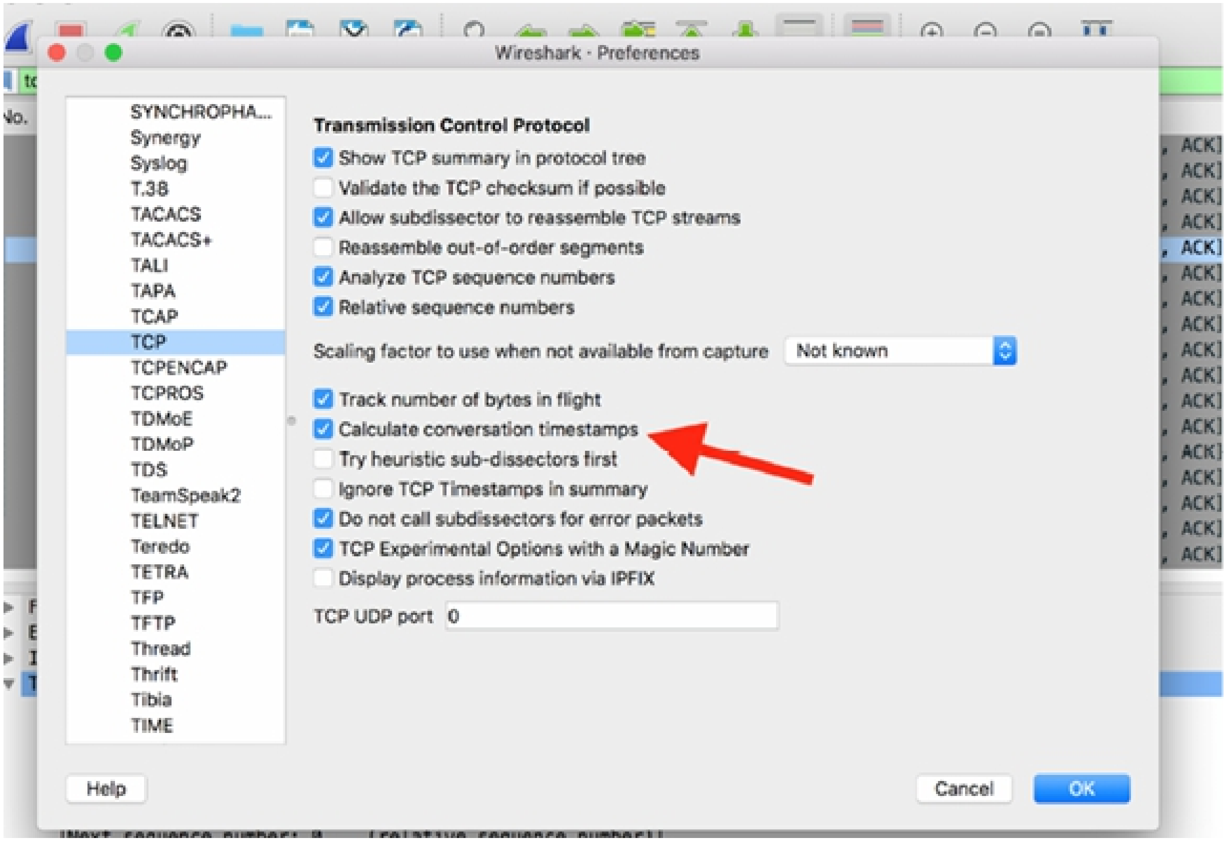
Task 2:
In Wireshark, capture the traffic for a few minutes while navigating using a web browser. Stop the capture and save the file.

To focus on the TCP roundtrip latency times in the TCP handshakes, in the filter toolbar, enter tcp.flags == 0x12 , as shown in the figure below.

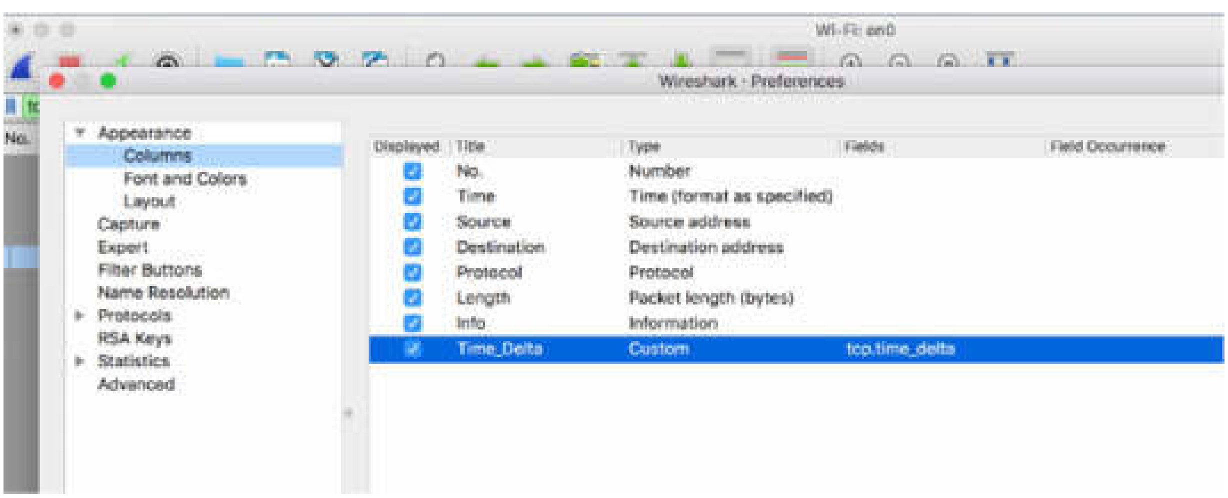


As shown in the figure above, only SYN/ACK packets are displayed.

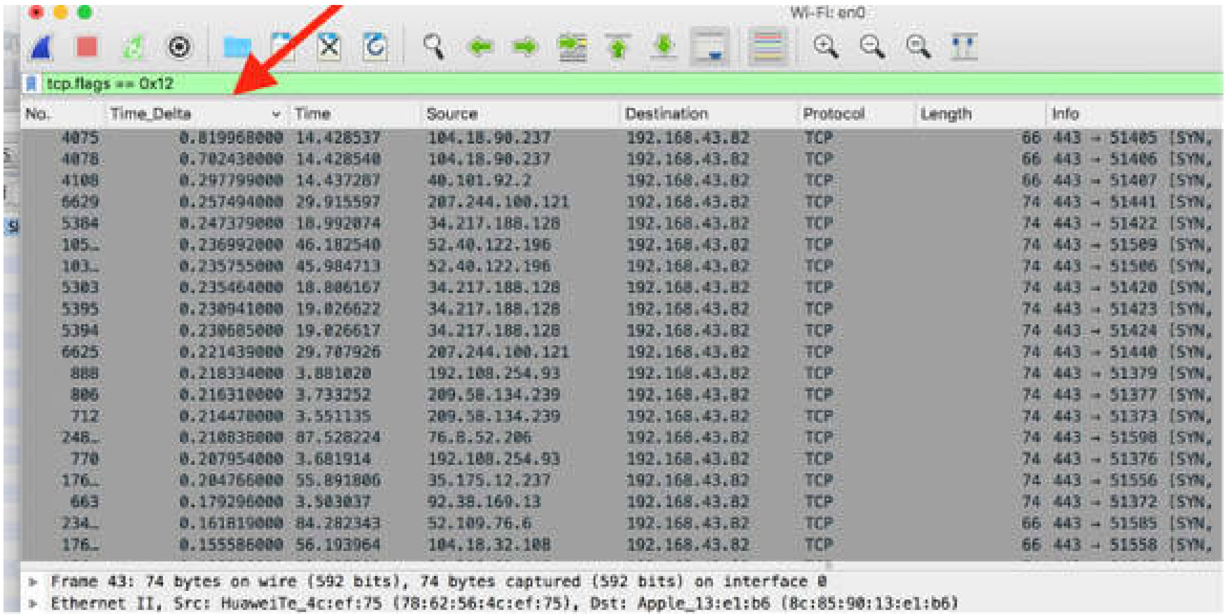
On the main menu, select Edit > Preferences. In the Preferences dialog box, select Protocol > TCP in the left tree view. Select the “Calculate conversation timestamps” check box, as shown in the figure below.



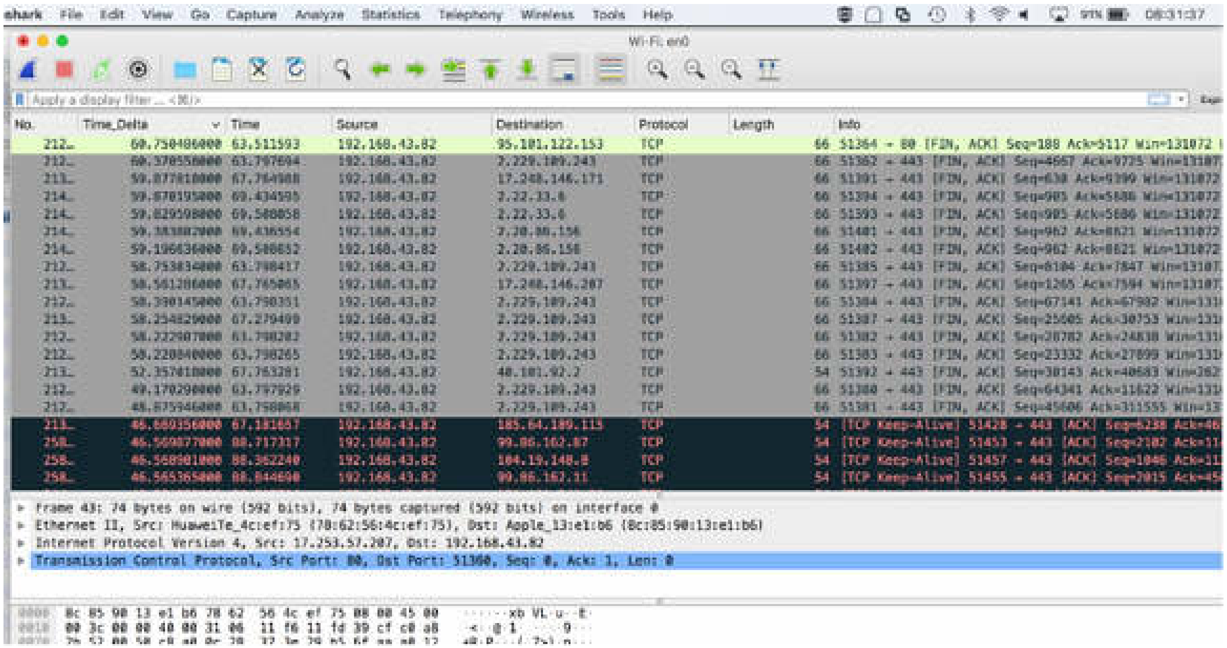
In the Preferences dialog box, add a column (Time_Delta) for displaying the time since the previous frame in this TCP stream (tcp.time_delta), as shown in the figure below.



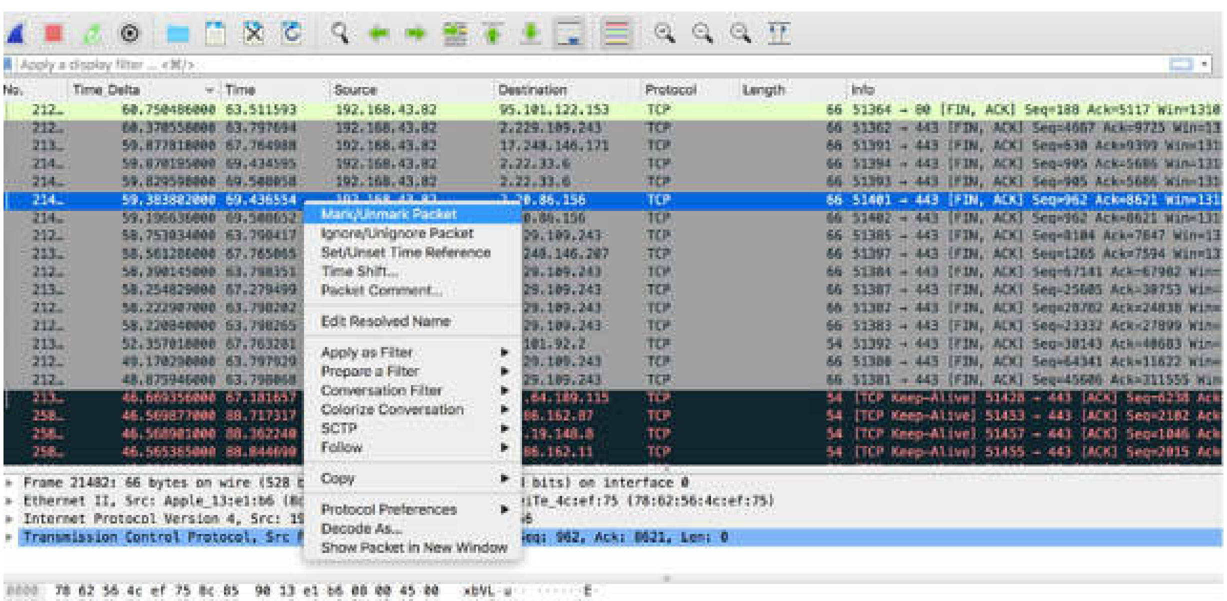
To view the roundtrip latency times of the TCP connections, sort the Time_Delta column by clicking the column header. The results, shown in the figure below, indicate that a few connections have a very high round trip latency (over 700 ms, packets #4075 and #4078).



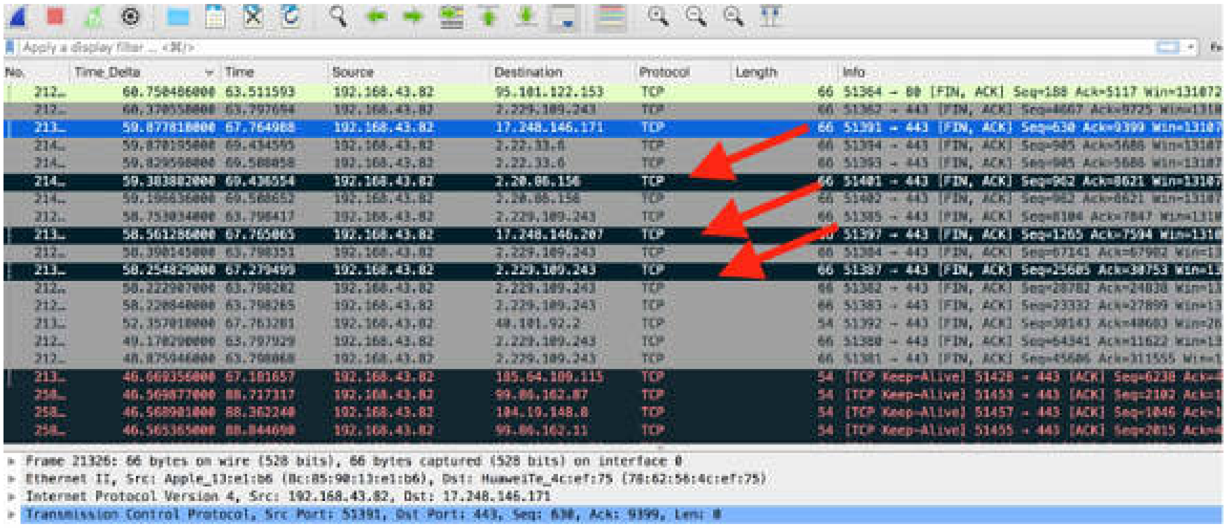
Remove the previous filter from the filter toolbar and again double-click the Time_Delta header column to sort the time delta values. This enables you to see the major delays between packets in each separate TCP stream, which is useful for considering and deciding which stream do you want to troubleshoot and which one you don't, as shown in the figure below.

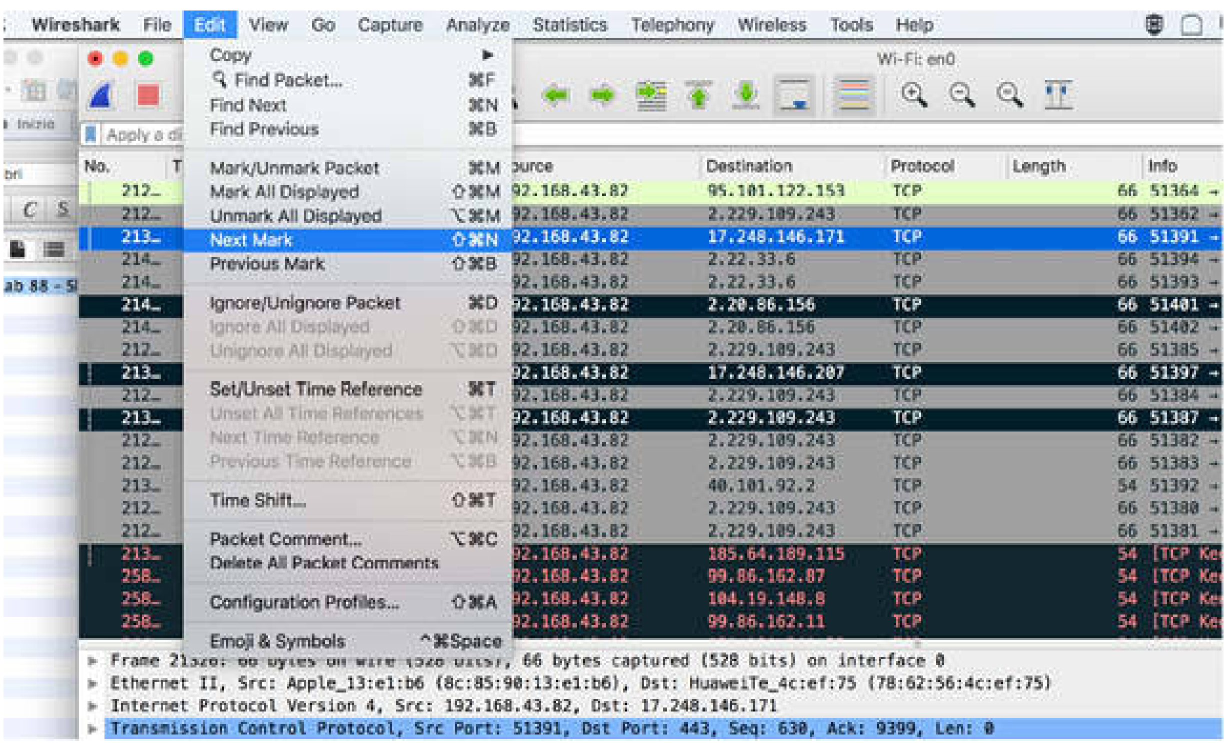


Task 3:
When you have many interesting packets that you want to focus on and also view them at a later stage, you can mark them. To do so, right-click the packets that you are interested in and then select Mark/Unmark Packet, as shown in the figure below.

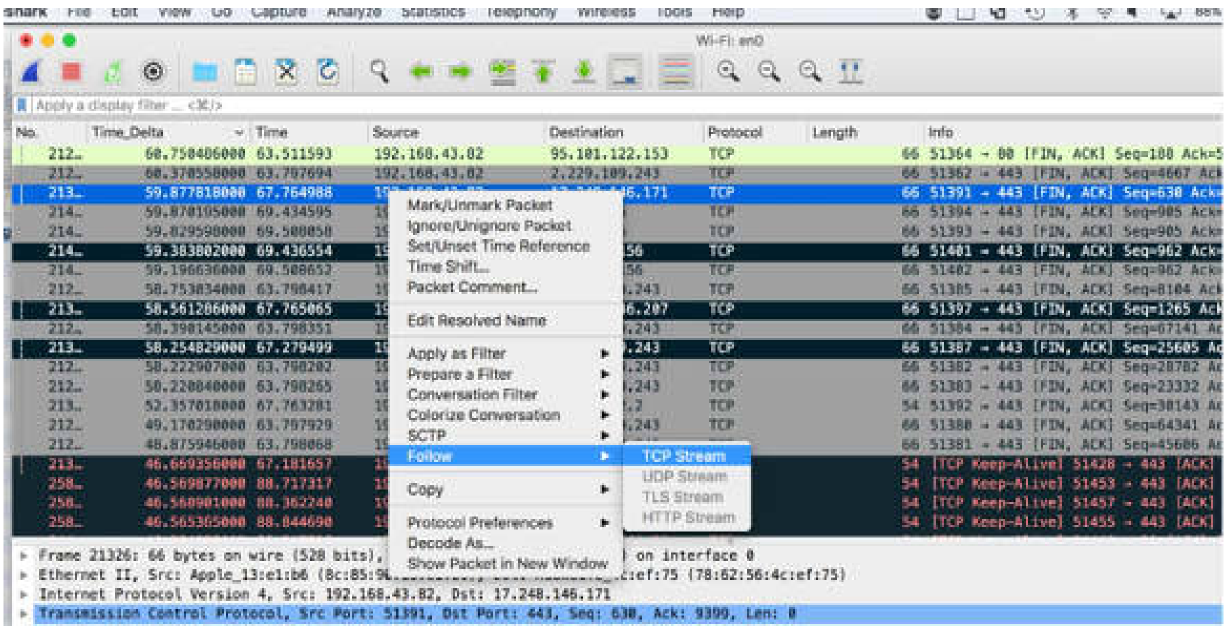


As a result, you can see all marked packets in the Packet List pane. You can also return to any of these packets at a later stage by using the Next Mark or Previous Mark options from the Edit menu.





Examine each packet with the large TCP delta time, and apply a TCP conversation filter to see what's happening, as shown in the figure below.



Notes:

Repeat the previous steps and try to identify the reasons for the slow processing time connection by using the tools offered in Wireshark. Mark the suspicious packets, and for each selected stream, find the reason for the issue.

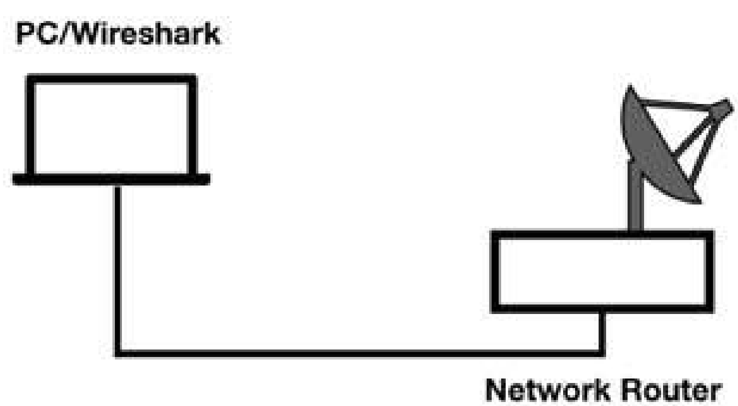
Lab 89. Packet Loss, Misconfiguration, and Redirections

Lab Objective:
Learn how to identify packet loss, misconfiguration, and redirections.

Lab Purpose:
Understand how to detect and troubleshoot issues related to packet loss, misconfiguration, and redirections.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

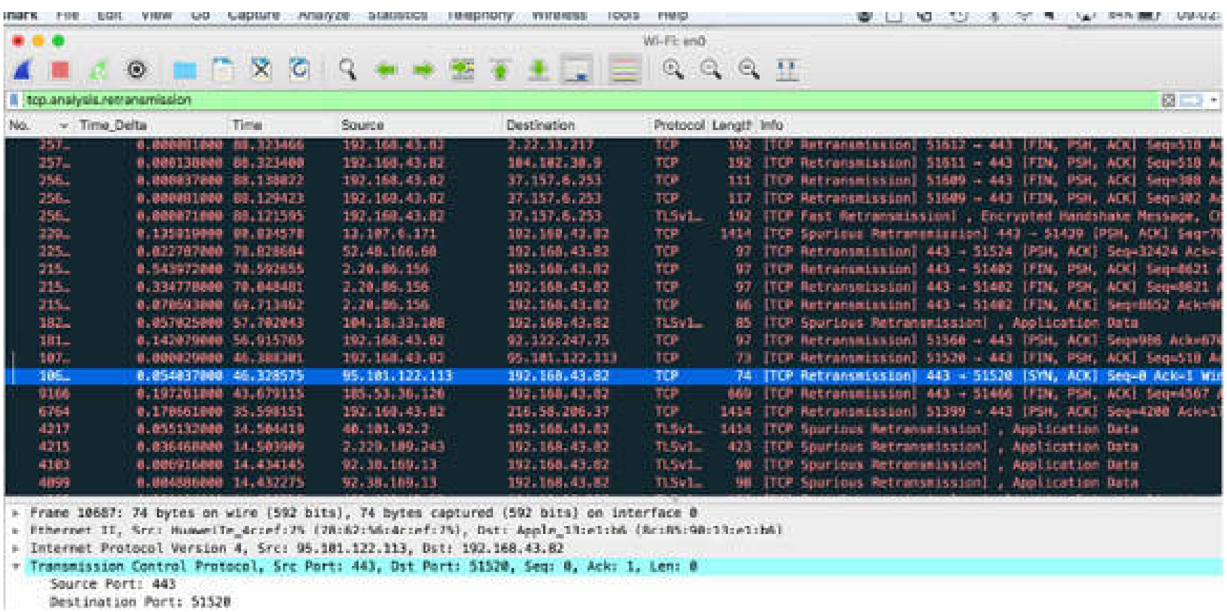


Lab Walkthrough:

Task 1:

Packet loss can affect performance when the receiver must request retransmissions and wait for those retransmissions before passing the data to the application. For example, when packet loss occurs on a TCP connection that does not support Selective ACKs, numerous packets may be retransmitted because the receiver cannot acknowledge receipt of data after the lost packet.

In Wireshark, capture the traffic for a few minutes while browsing the web using a web browser. Stop the capture and save the file. To analyze whether the communication contains retransmission packets, in the filter toolbar, enter tcp.analysis.retransmission , as shown in the figure below.

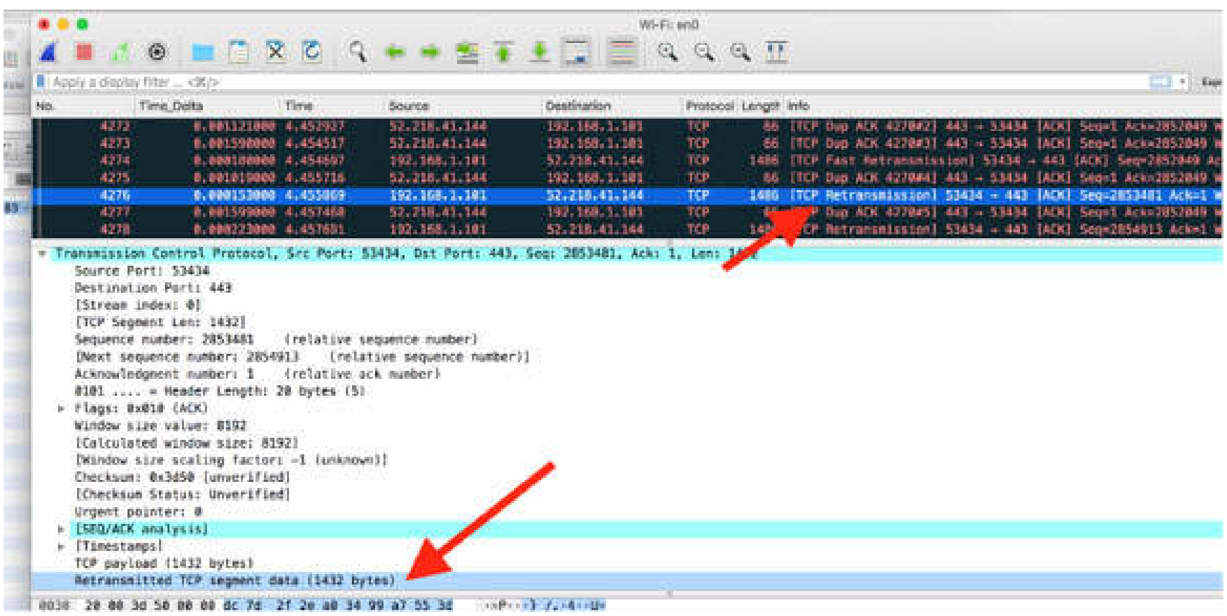


In the figure above, you can see a lot of TCP retransmission packets, indicating that the performance of the communication is getting low.

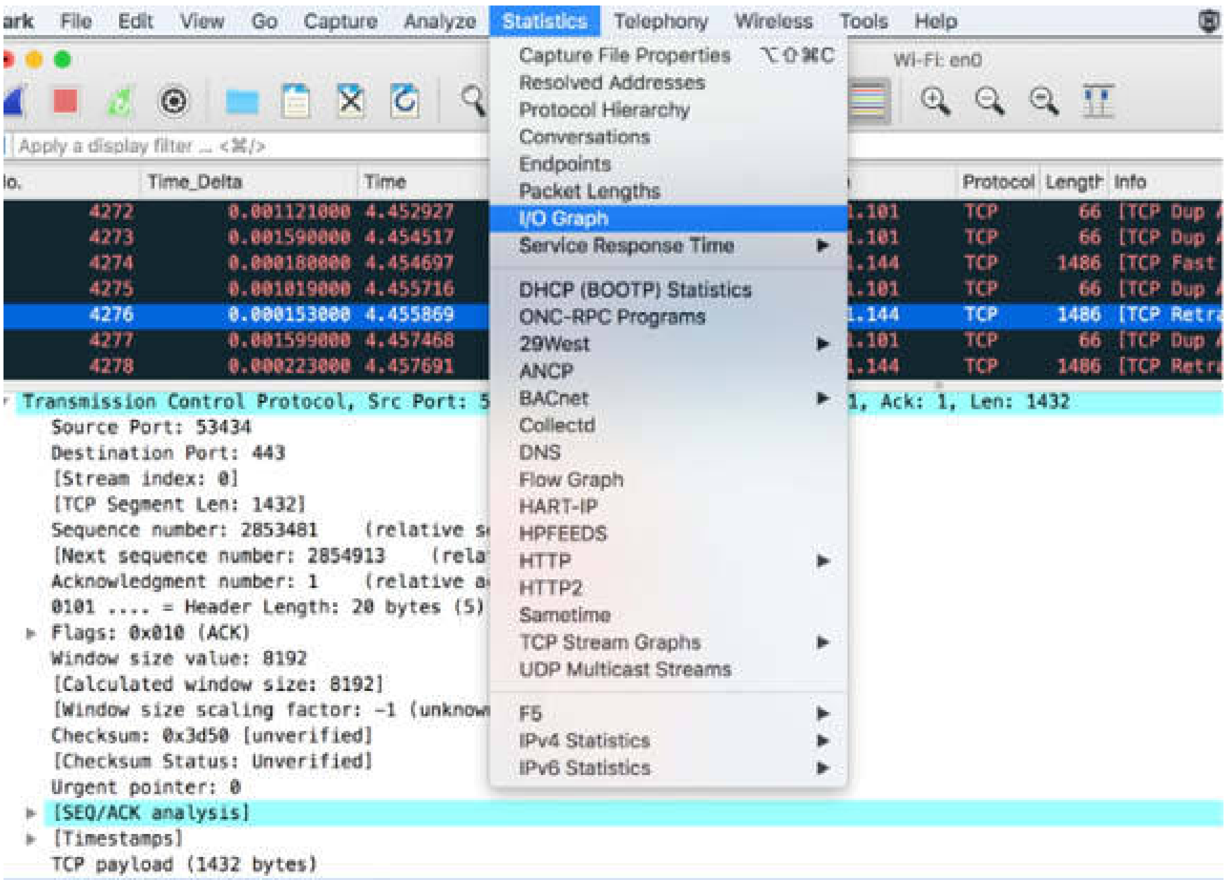
In a UDP-based application, the application decides the retransmission timeout value. An application that is slow to request retransmission affects the overall performance of the application. For example, in a retransmission process occurring on a DHCP client, when the first DHCP Discover packet goes unanswered, the DHCP client has to retransmit the Discover packet. If the client waits for a couple of seconds before retransmitting the Discover packet, it could result in a delay in recovering from the possible packet loss during the bootup process. Because the DHCP server or relay agent must be on the same network segment, a couple of seconds seem to be an excessive amount of time.

If you are capturing traffic in the infrastructure and you see the original packet and the retransmission, you are upstream (at a point before) from the point of packet loss. Upstream means you are closer to the sender of the data. To find out where the packet loss is occurring and the packets are being dropped, move along the path until you no longer see the original packet and retransmissions.

Packet loss typically occurs at interconnecting devices, such as switches and routers. This is a relatively simple process for TCP communications because Wireshark clearly indicates which packets are retransmissions. The Info field in the Packet List pane and the Packet Details pane provide this information, as shown in the figure below.



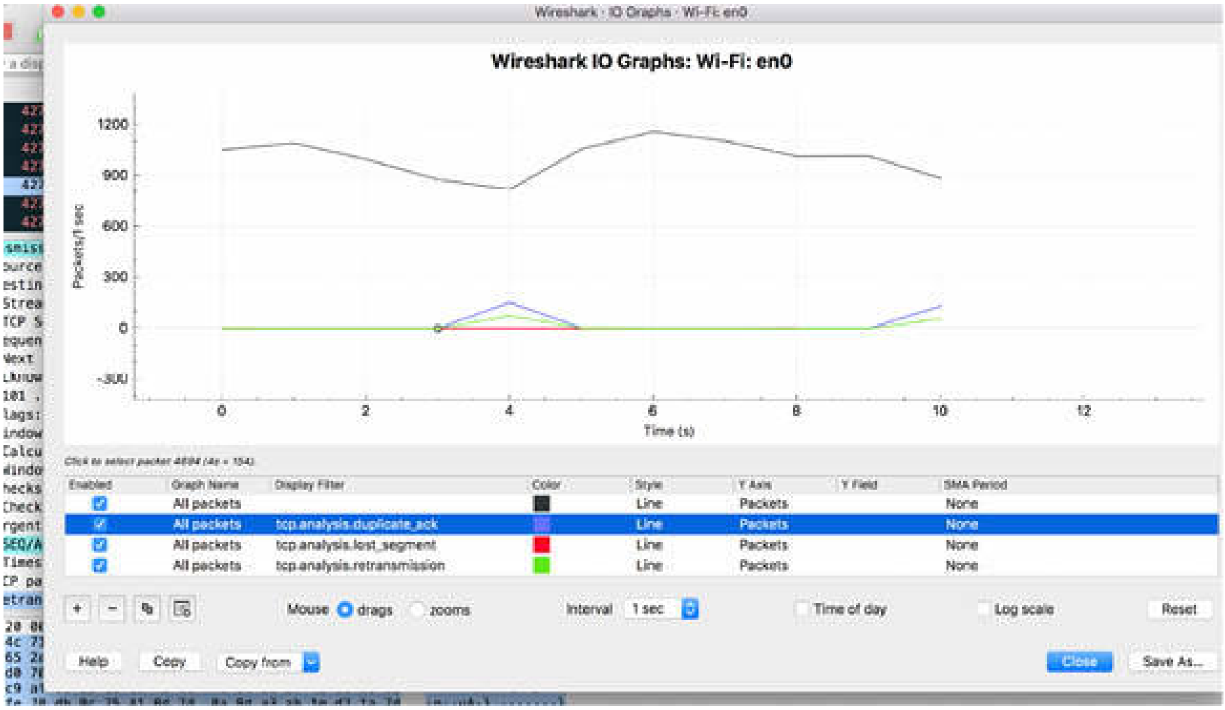
To find out whether there is any packet loss in a capture file, on the main menu, select Statistics > I/O Graph to open the I/O Graph , as shown in the figure below.



In the "Display Filter" fields for the graph, enter the following filters:

- tcp.analysis.duplicate_ack
- tcp.analysis.lost_segment
- tcp.analysis.retransmission

Assign a different color to each graph, as shown in the figure below.

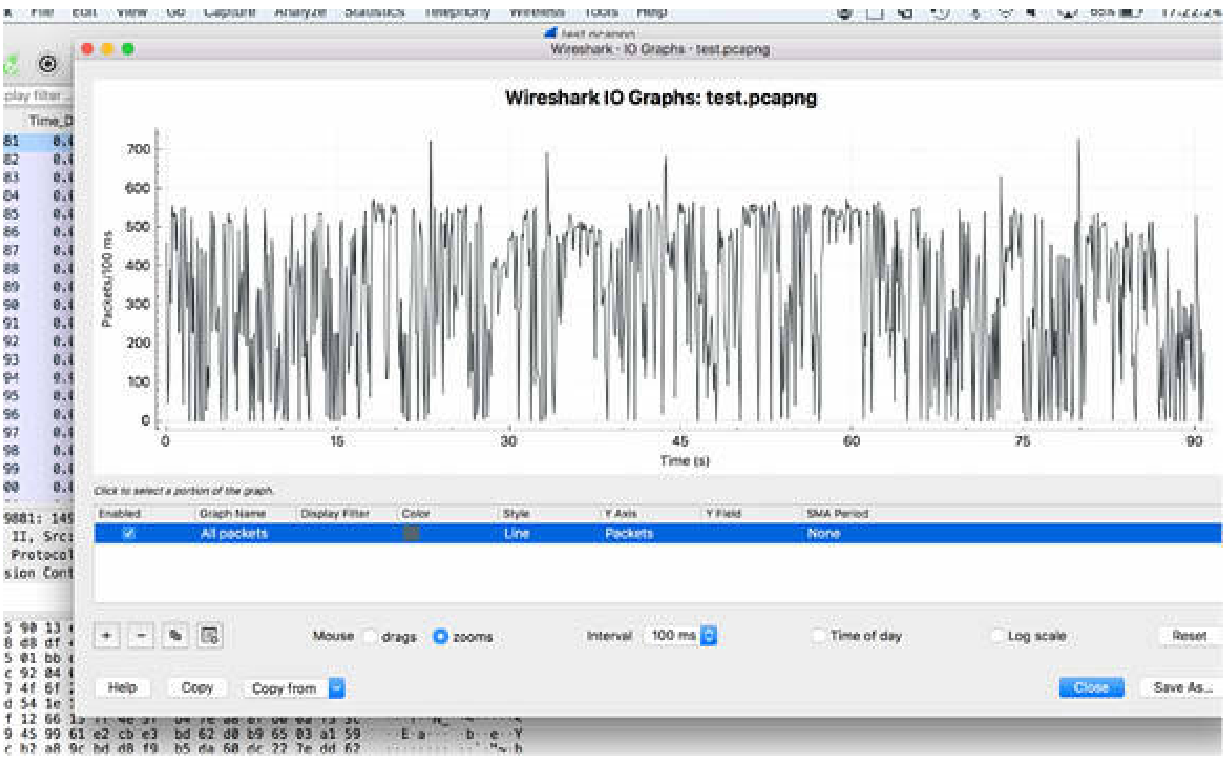


As shown in the graph above, there is no clear impact of packets lost during this quite short TCP session. In general, TCP doesn't do well with a large number of packets lost in a single congestion window, so it is important to keep an eye on this important parameter.

Task 2:

In a web browser, watch a video from a webstream. In the meantime, download a large file from the web. In Wireshark, capture the traffic for a few minutes. Stop the capture and save the file.

Open the I/O graph to display the packets/tick for a tick of 100ms, as shown in the figure below.



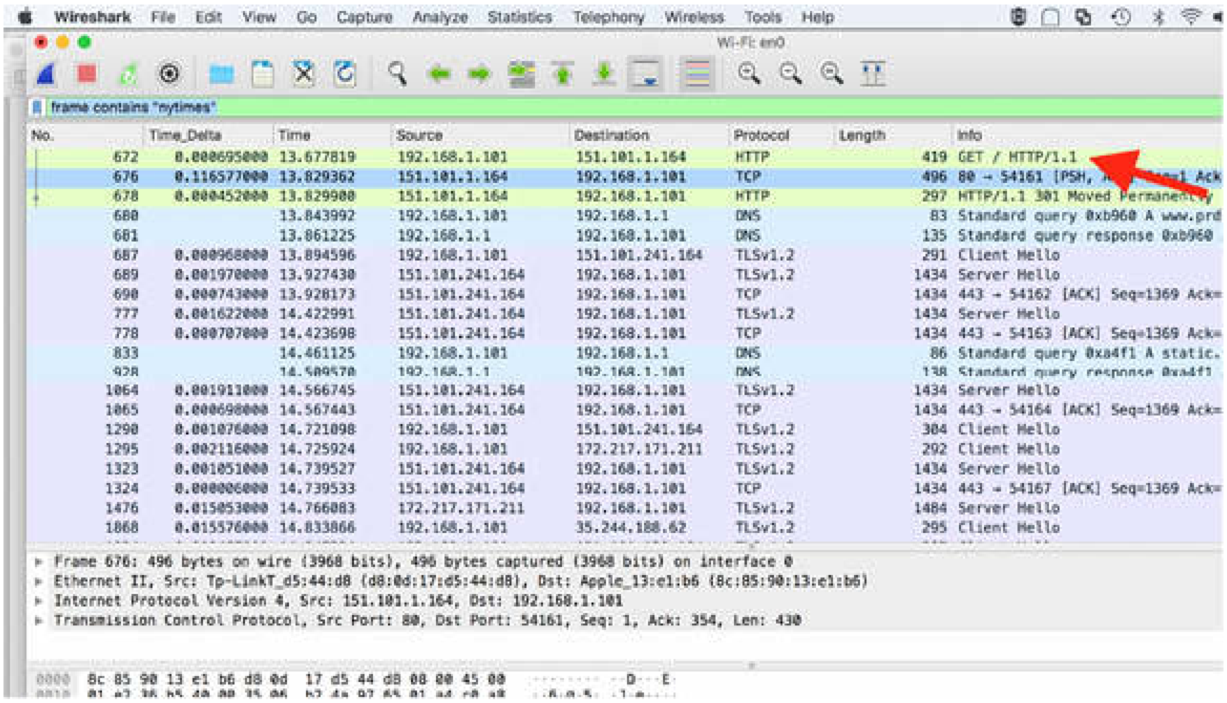
The heartbeat effect is indicated in the I/O graph. Various misconfigurations can affect network performance. For example, video multicast traffic—having a lower priority than the file transfer, voice, and email traffic—may be held in queues along a path while the higher priority traffic flows ahead of it.

Task 3:

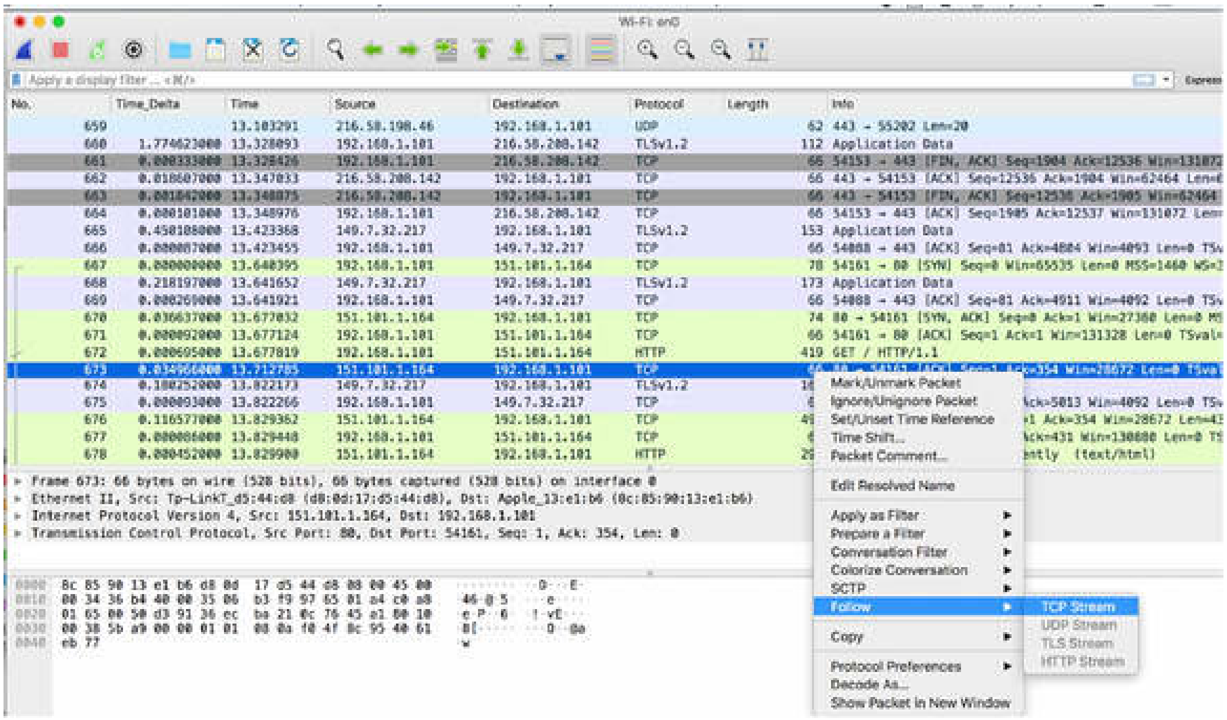
Traffic redirections can also cause common performance issues. The most common redirections seen on a network are based on the default paths that may not be optimal or available. This could be a default gateway that does not offer the best route to the target network (responding with an ICMP redirection packet).

Another common redirection is seen in web browsing sessions when a browsing client connects to a website only to be redirected to other sites to build the pages. To verify the effect of such redirection, in Wireshark, capture the traffic for a few minutes, and by using a web browser, go to **www.nytimes.org** . After some seconds, the web browser is redirected to **www.nytimes.com** .

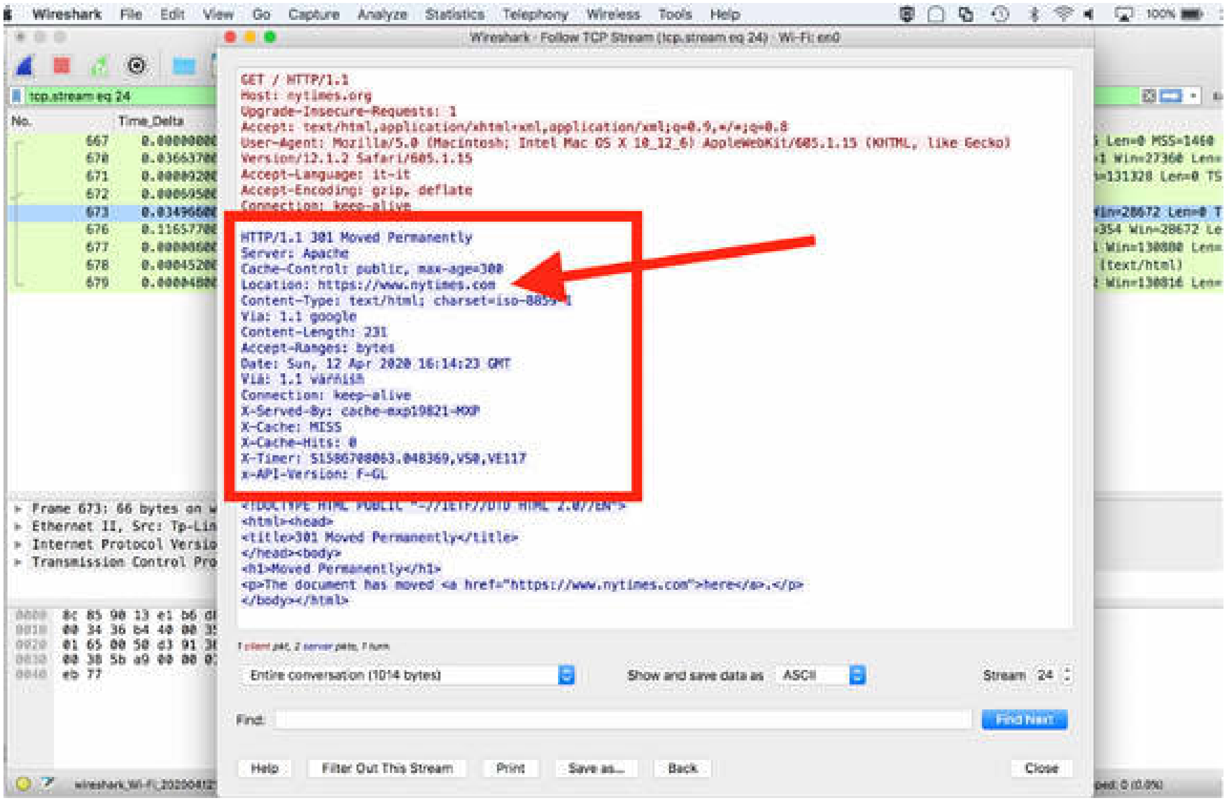
Stop the capture and save the file. To identify the first HTTP packet (#672) requesting the webpage, in the filter toolbar, enter frame contains "nytimes" , as shown in the figure below.



Remove the previous filter from the filter toolbar and select the packet immediately after the HTTP get request (packet #672). Right-click the selected packet and select Follow > TCP stream, as shown in the figure below.



The Follow TCP Stream dialog box is displayed in which the server indicates to the client that it must connect to “nytimes.com”. This prompts the client to generate a DNS query for the new website before generating a TCP handshake.



Notes:

Repeat the previous steps to capture some traffic in Wireshark and identify possible packet loss in the capture. Try to find the root cause. Generate the related I/O graphs to confirm whether the network is correctly configured or not and whether there are redirection problems.

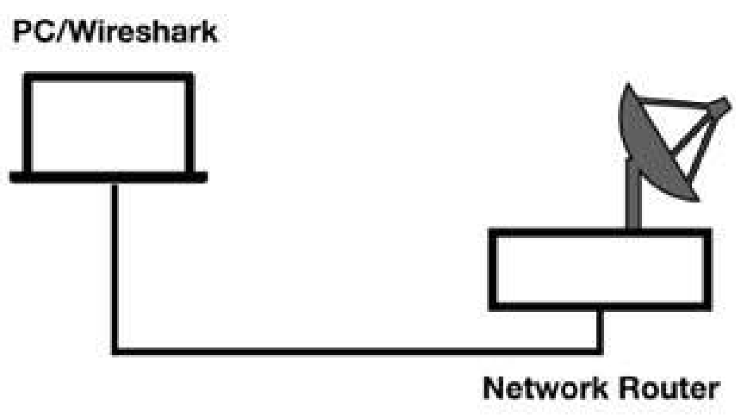
Lab 90. Payload Sizes, Congestion, and Faults

Lab Objective:
Learn how to identify problems related to small payload sizes, congestion, and application faults.

Lab Purpose:
Understand how to detect and troubleshoot issues related to small payload sizes, congestion, and application faults.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



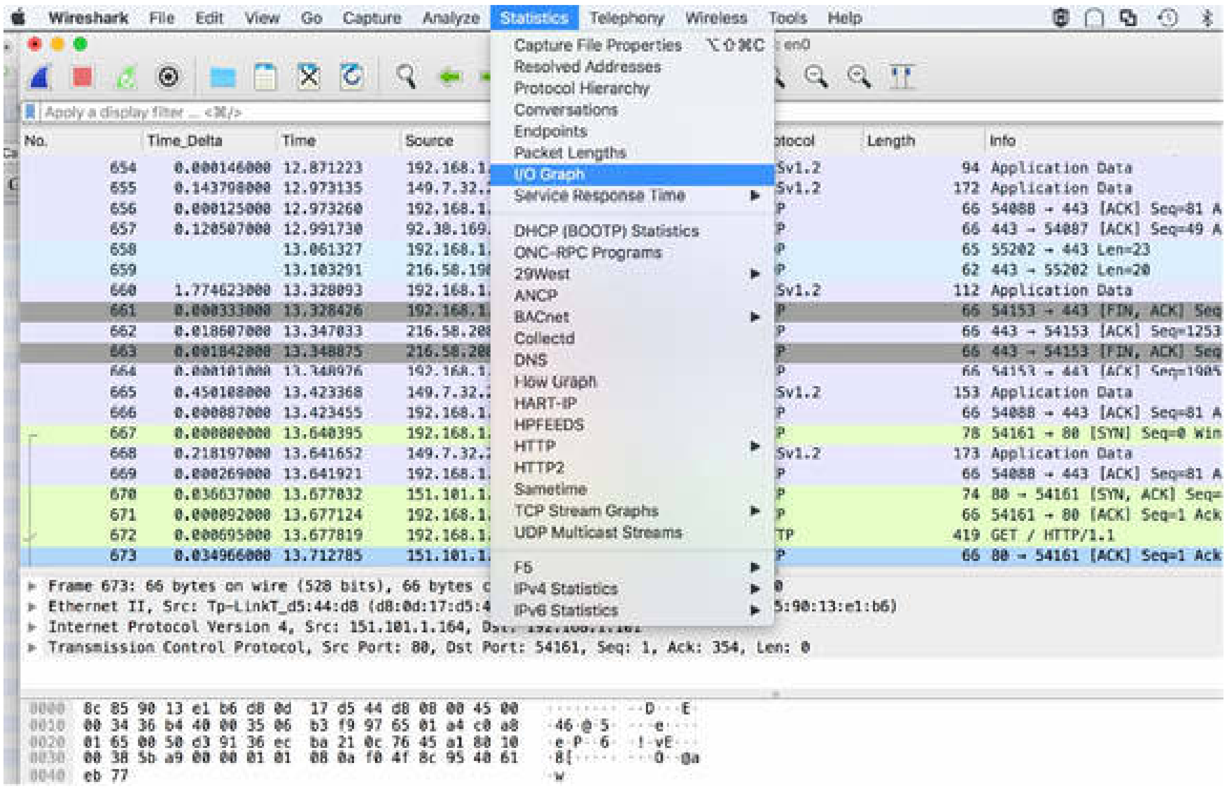
Lab Walkthrough:

Task 1:
Problems related to small payload sizes can occur when small segments are used instead of the classic sizes. For example, in a situation where a 500 MB file is exchanged in 512-byte segments, instead of 1460-byte segments, the data exchange requires 665 more data packets to complete the transfer. This overload can affect the communication in terms of the number of packets.

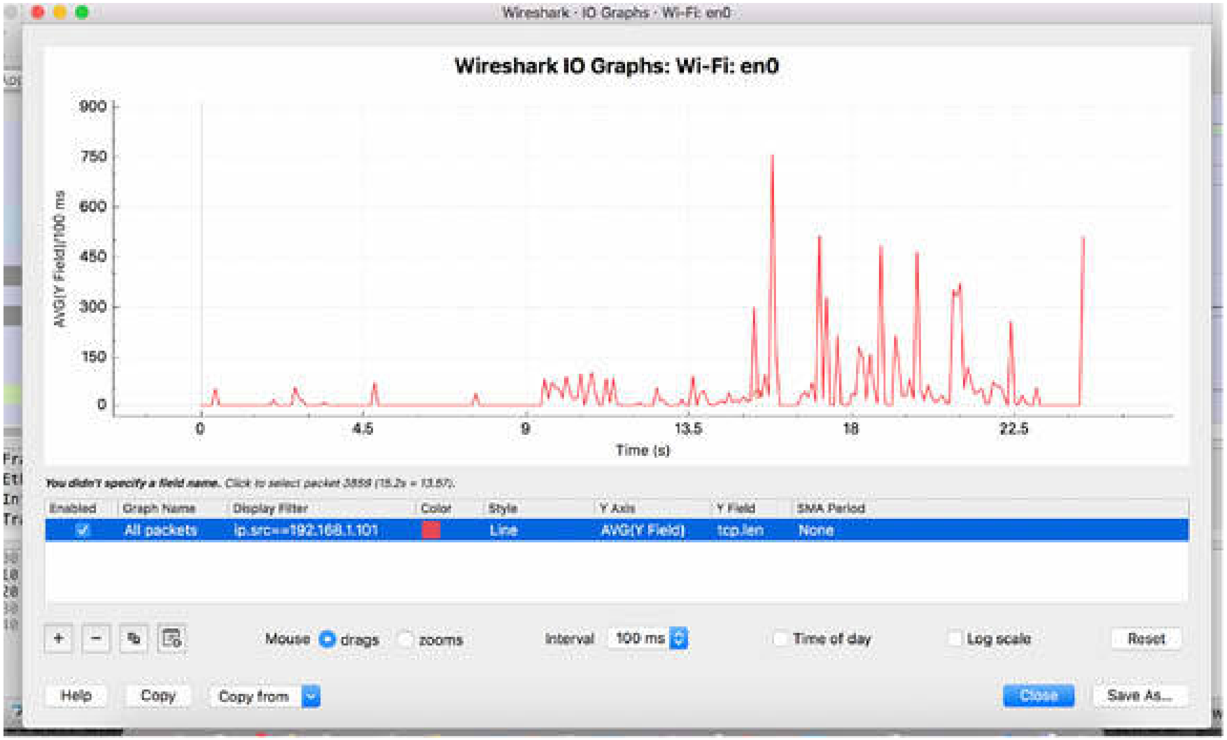
Some applications may use a smaller payload size on purpose. For example, database applications may be transferring a record or set of records at a time. The records may be non-contiguous in the file, so a steady stream of larger data segments is not possible. An example of this is when two distant TCP hosts complete the handshake process, indicating the Maximum Segment Size (MSS) value of 1460 bytes. If a part of the network path only supports an MTU size of 512, one of the following is required:

- The router adjoining the limiting segment must fragment the packets.
- The peers must use ICMP path discovery to identify the new MTU size to use when communicating.

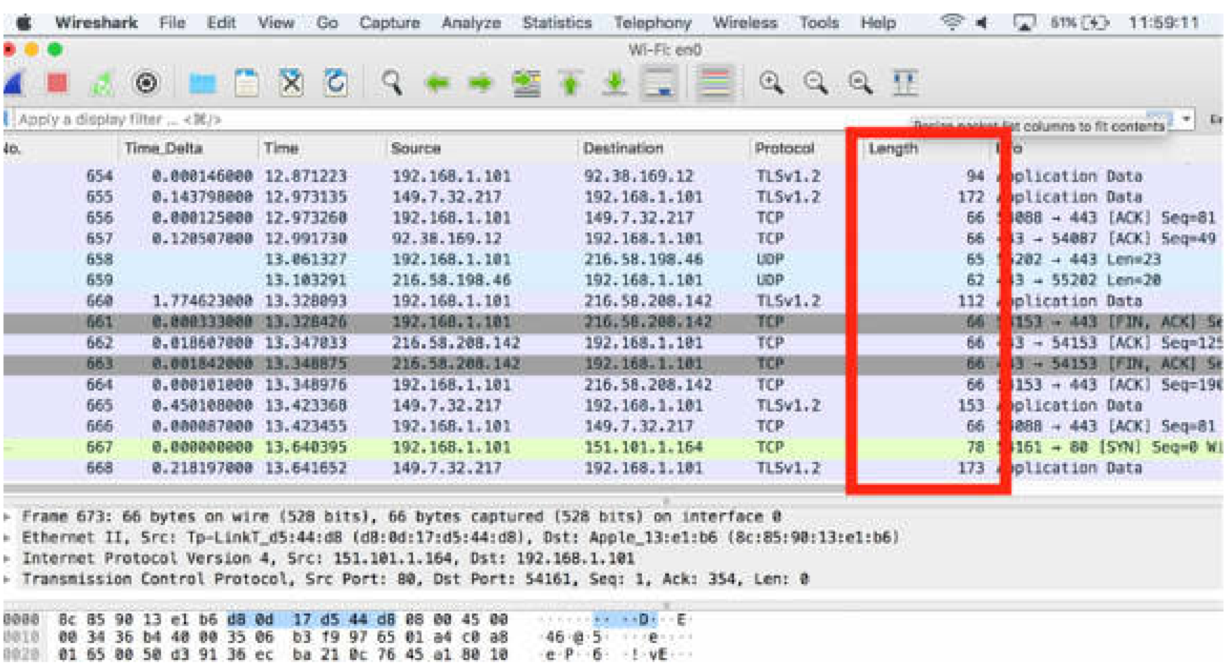
In Wireshark, capture the traffic for a few minutes. Stop the capture and save the file. On the main menu, select Statistics > I/O Graph, as shown in the figure below.



In the generated I/O graph, apply a TCP filter to identify only a single transmitter (ip.src == x.x.x.x). To depict the traffic payload, in the "Y Axis" field, select AVG(*) Y Field , and in "Y Field", enter tcp.len , as shown in the figure below.



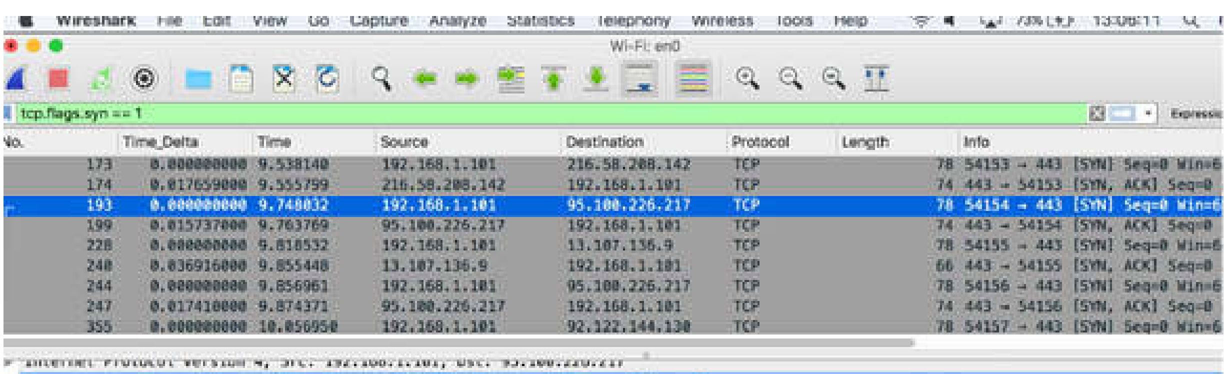
The graph above is useful even when Wireshark automatically uses the packet length column in the Packet List pane, as shown in the figure below.



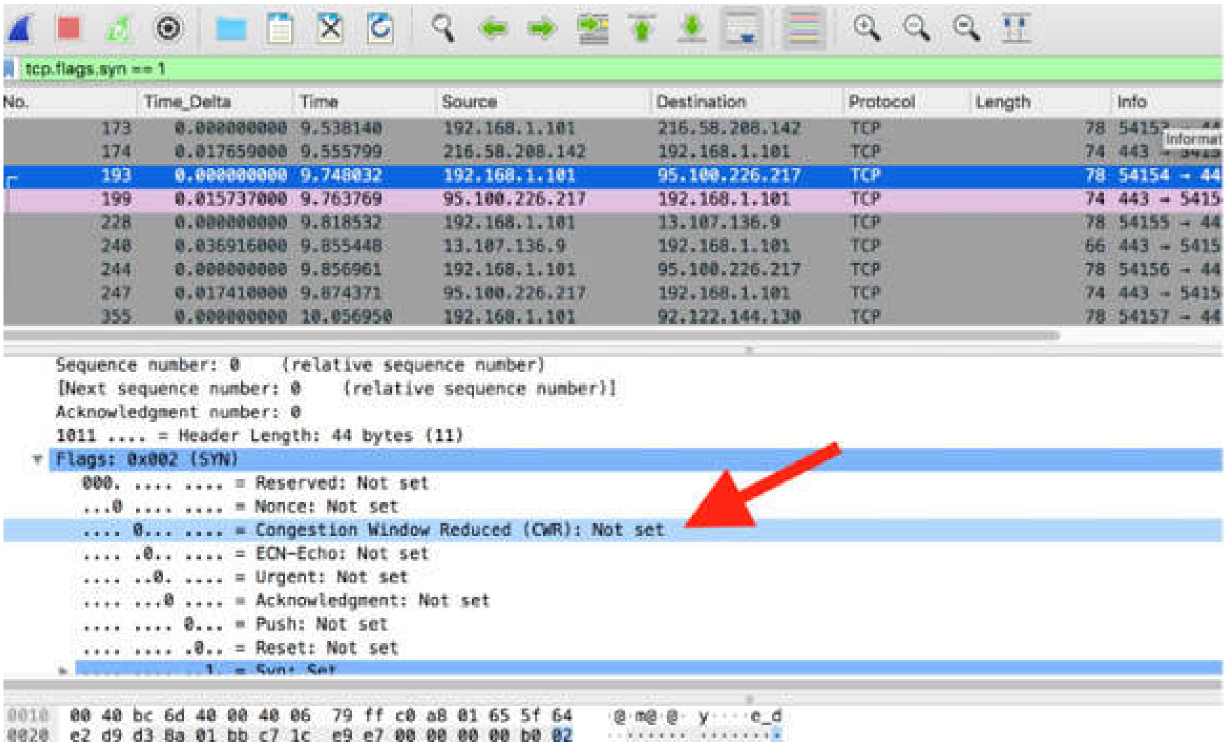
Task 2:
Congestion along a network path may cause packet loss, queuing, or throttling back of possible throughput maximums. A window zero condition is one of the common examples of possible congestion at a receiving host. Alternatively, this could be caused by an application that starts to transmit without respecting or following the configured main cycle.

When a host hits a window size of zero, no data can be sent. As a result, the IO drops to zero bytes/second. You can verify this by using the I/O graph functionality in Wireshark. One possible solution for the zero window condition is to use window scaling, which enables a host to exponentially increase the window size. Window scaling is defined as a TCP option during the handshake process.

Open the capture file saved in the previous task. In the filter toolbar, enter tcp.flags.syn==1 to display only the start of the TCP handshake process, as shown in the figure below.



In the Packet List pane, select a SYN packet. In the Packet Details pane, click the Flags field to open the tree view and verify the flags related to the window size, as shown in the figure below (where the Reduced Window is not implemented).

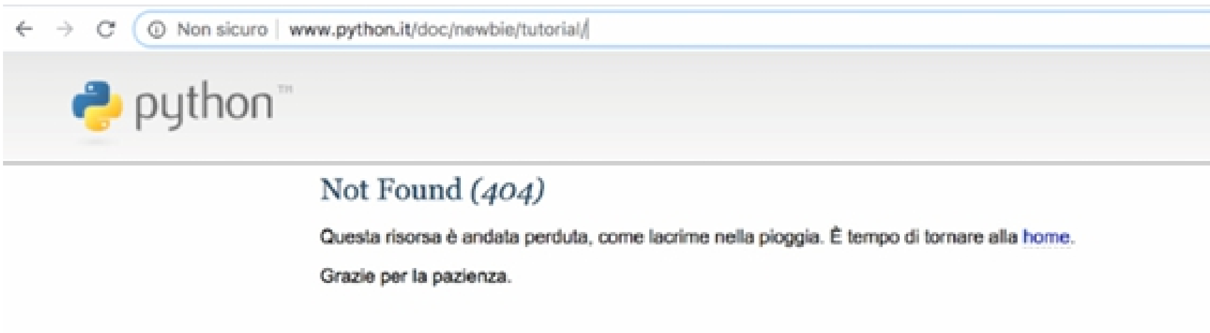


When the network experiences flooding, defined as a condition of prolonged peak packets per second or bytes per second rate, communication may suffer in different ways. In some cases, the flood traffic may overwhelm Wireshark, making it impossible to capture traffic while displaying packets in the GUI. In such a case, you can use TShark (the console-only version of Wireshark explained later) to capture the traffic and directly save it to a file. You can later examine the trace files separately.

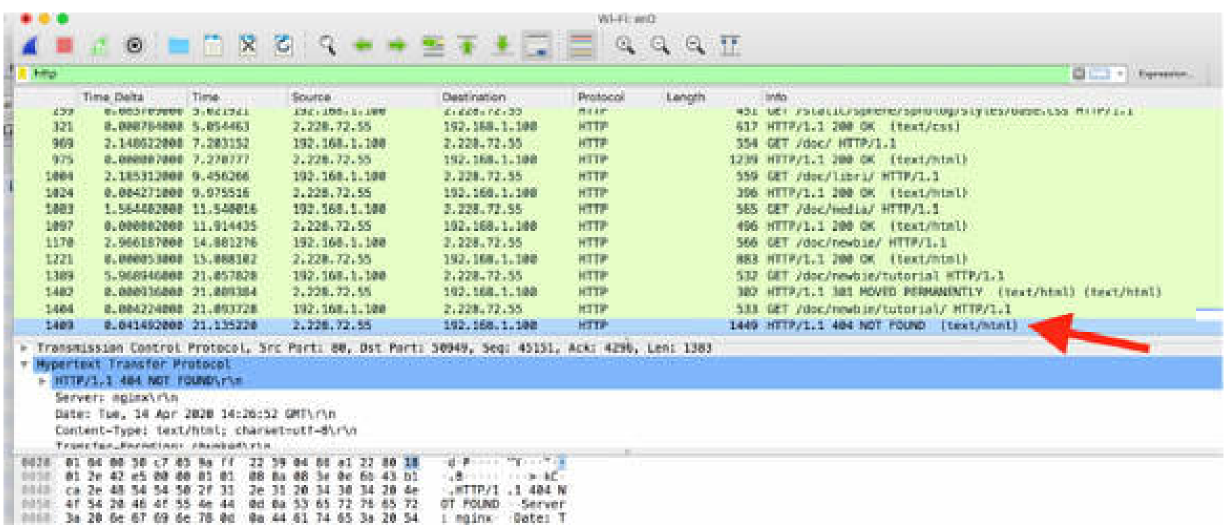
Task 3:

Another case of performance problems is when application faults occur. These faults may manifest through dissected response codes or by simply not allowing efficient data flow. The HTTP 404 Not Found response received by a web browsing client is a commonly experienced fault. No data is transferred from the target page after this condition, and no redirection takes place.

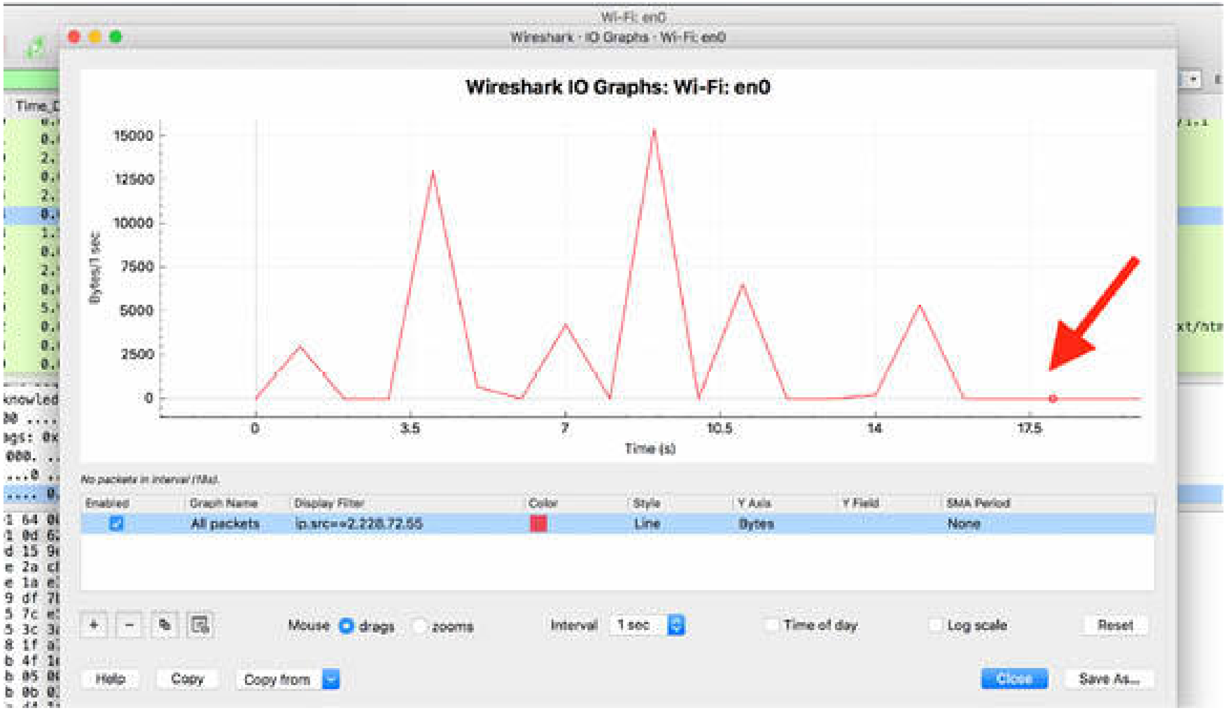
In Wireshark, capture the traffic for a few minutes, and in a web browser, go to **http://www.python.it** . Browse the website by clicking on various links on the main page and going back to the main page. At a certain point, add “/tutorial” at the end of the current web address and press the Return key. You will get the RETURN—Page Not Found result, as shown in the figure below.



Stop the capture and save the file. In the filter toolbar, enter http . You can verify the presence of the HTTP 404 response, as shown in the figure below.



Open the I/O graph and in the “Display Filter” field, enter the source IP address of the website visited (in this case, 2.228.72.55) and verify that at a certain point, the server throughput has drooped to zero because the page doesn’t exist. The result is displayed in the figure below. If you click on the point where the throughput is dropped to zero, you can jump to the related packet in the trace file.



Notes:

Repeat the previous steps to capture some traffic in Wireshark and identify possible issues because of payload sizes, congestion in the network, or fault of the server. Get the necessary confidence in using the tools provided by Wireshark and in particular, the I/O graph that is useful, especially when used with appropriate filters.

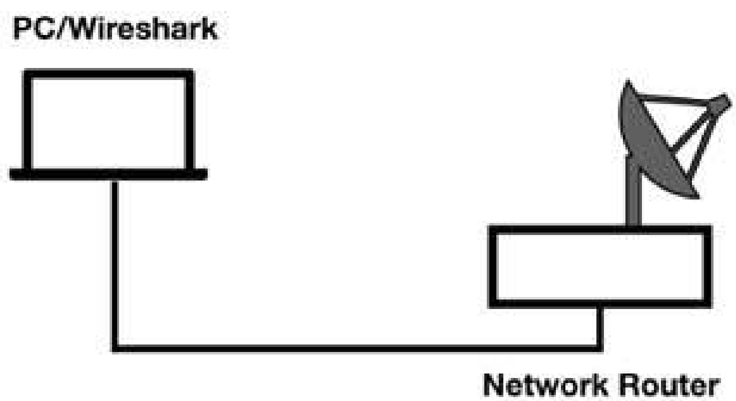
Lab 91. Network Forensics

Lab Objective:
Learn about network forensics is and which aspects of networking it covers.

Lab Purpose:
Understand how to approach network traffic and get the necessary evidence for network forensics.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:

When approaching forensics, it is important to understand the difference between host forensics and network forensics.

Host forensics is the process of investigating media storage elements, such as internal and external hard drives, to find some kind of evidence. Evidence may include searching for data files (maybe hidden or password protected), locally-stored emails, registry settings, browsing history, etc.

Network forensics is the process of examining network traffic for evidence of unusual or malicious traffic on the communication network. This traffic may include discovery processes, phone-home behavior, denial-of-service attacks, man-in-the-middle poisoning, botnet commands, etc. Network forensics can also be used to study traffic patterns of malicious activity to properly configure network defense mechanisms.

Task 2:
Network forensic evidence may be collected and divided into two main categories: proactive or reactive analysis.

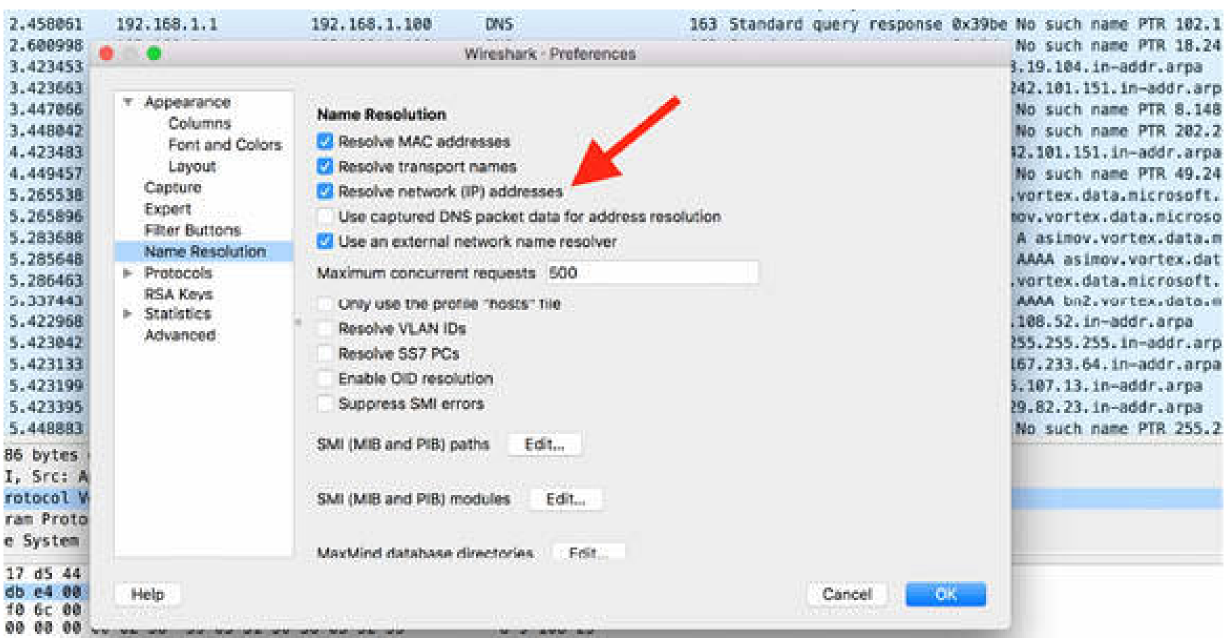
Proactive analysis techniques may require placing network capture devices at various key locations on the network and saving large volumes of traffic. An Intrusion Detection System offers the complementary capability for examining network traffic and alerting IT staff in case some unusual traffic patterns are detected. The placement of Wireshark, on the other hand, depends on the issue that needs to be investigated.

Using Wireshark to capture traffic to and from a suspect host is an example of reactive analysis. In case there is a clear sign of numerous compromised hosts that are communicating with command and control servers, it is recommended to place Wireshark close to the network exit point. You can filter the IP addresses of the suspect hosts or filter only specific traffic by using the protocol in use.

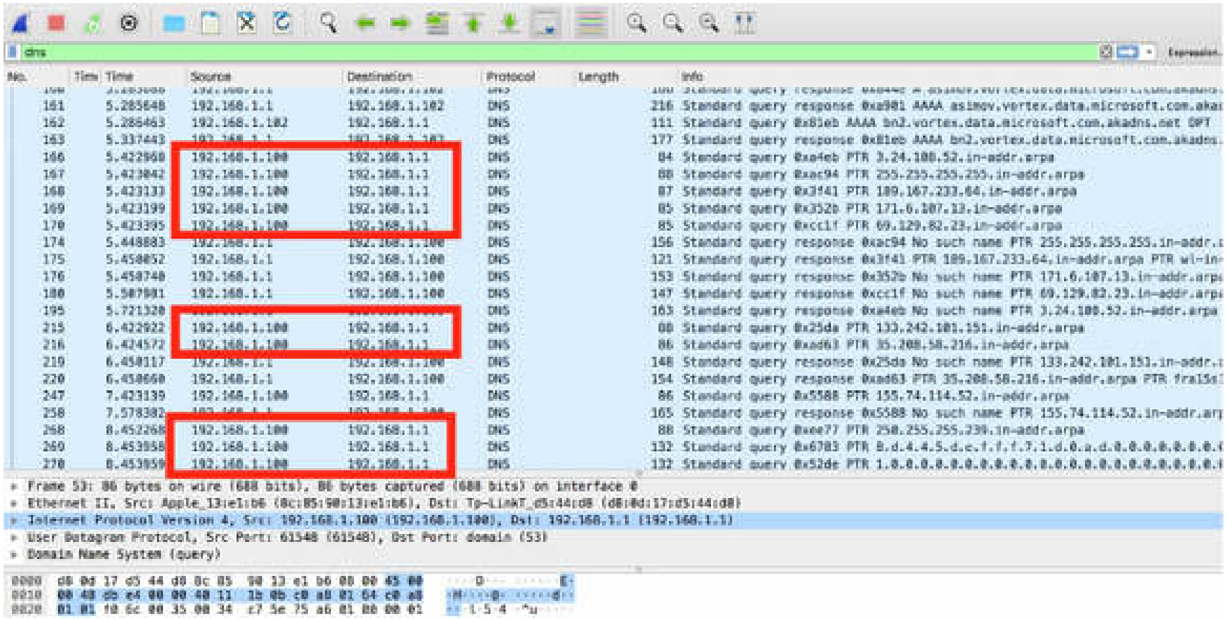
The purpose of network forensics analysis is to convert the traffic data to useful information. In fact, capturing high quantities of traffic can lead to disorganized and large files. Therefore, it is important to distill them into separate conversations, differentiate protocols, to collect and divide time slots and find a way to more easily locate possible breaches.

Task 3:
In general, by default, Wireshark does not transmit data on the network while it is being used to capture packets. Other applications running on the same host as Wireshark may, however, be communicating.

Wireshark may be detectable in case you enable network name resolution. To get this confirmation, on the main menu, click Edit > Preferences. In the Preferences dialog box, select "Name Resolution" in the left tree view and select the "Resolve network (IP) addresses" check box, as shown in the figure below.

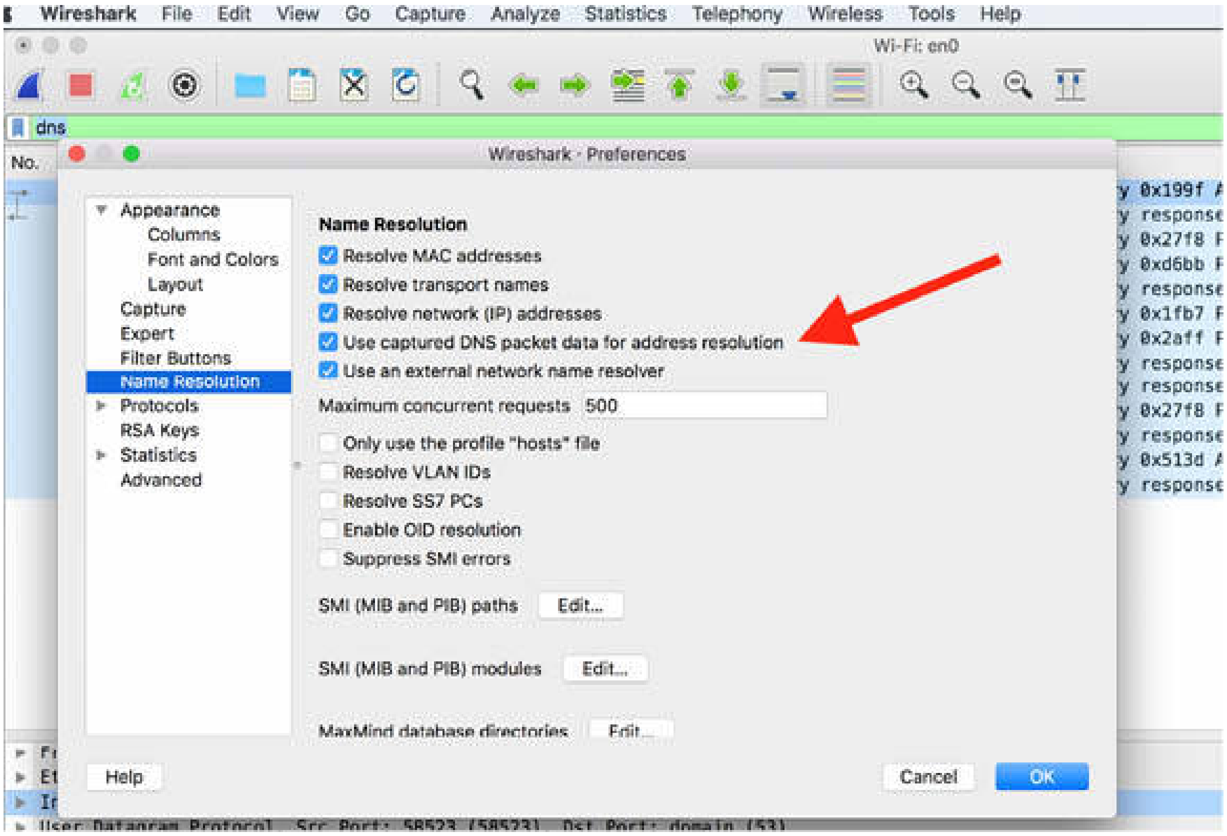


Capture the traffic on an active network interface for a few minutes. Stop the capture and save the file. In the filter toolbar, enter dns . You will observe a large number of DNS requests transmitted from the machine where Wireshark is running (i.e., ip src 192.168.1.100), as shown in the figure below.



You can see that the host running Wireshark is capturing traffic on an Ethernet network and sending DNS PTR queries for each captured IP address. Besides making Wireshark visible, this traffic may overwhelm the DNS server if the capture contains a high number of IP addresses.

There are two main solutions to this problem. The first one is to select the “Use captured DNS packet data for address resolution” check box in the Preference dialog box, as shown in the figure below. In this way, the number of forced DNS requests is extremely reduced because the “normal” DNS messages are used to resolve the names.



Another possible solution is to disable the TCP/IP stack on the host running Wireshark. It is important to note that even if the TCP/IP stack is disabled, Wireshark can still capture traffic. To apply this setting, open the Network Preference interface of the machine running Wireshark and select the TCP/IP tab, as shown in the figure below (on MAC OS).

In the Configure IPv4 list, select Off, as shown in the figure.

In Wireshark, capture the traffic for a few minutes to verify that it is still possible to capture the packets on the network.

Some software tools can perform discovery processes to identify hosts in promiscuous mode—a requirement for capturing traffic addressed to other hosts' hardware addresses. Such tools can search and identify a host running Wireshark or another packet capture tool on the network.

These software tools first send an ARP scan to discover all devices on the local network and then sort the devices based on some tests done on the MAC address.

- 31-bit Broadcast MAC Address (0xff:ff:ff:ff:ff:fe)
- 16-bit Broadcast MAC Address (0xff:ff:00:00:00:00)
- 8-bit Broadcast MAC Address (0xff:00:00:00:00:00)
- Multicast MAC Address ending in 0 (0x01:00:5e:00:00:00)
- Multicast MAC Address ending in 1 (0x01:00:5e:00:00:01)

If a couple or more of these tests are applicable to a host, the host is probably capturing the network in promiscuous mode.

Notes:

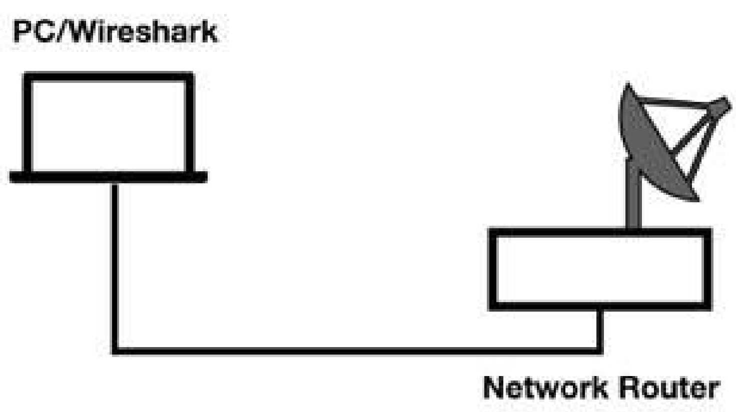
Lab 92. Handle Evidence and Unusual Traffic Patterns

Lab Objective:
Learn how to properly handle evidence and how to recognize unusual traffic patterns.

Lab Purpose:
Understand how to properly manage the evidence collected and recognize unusual traffic patterns while capturing.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:

The handling of evidence should not alter or cause concern regarding its integrity. Trace files should always be stored in a secure location, and the chain of custody documentation should define the capture process and location, trace file control, and the transfer and analysis process details. The staff responsible for IT services should have a fireproof safe for securing magnetic media that contains evidence and follows all recommended evidence-handling procedures.

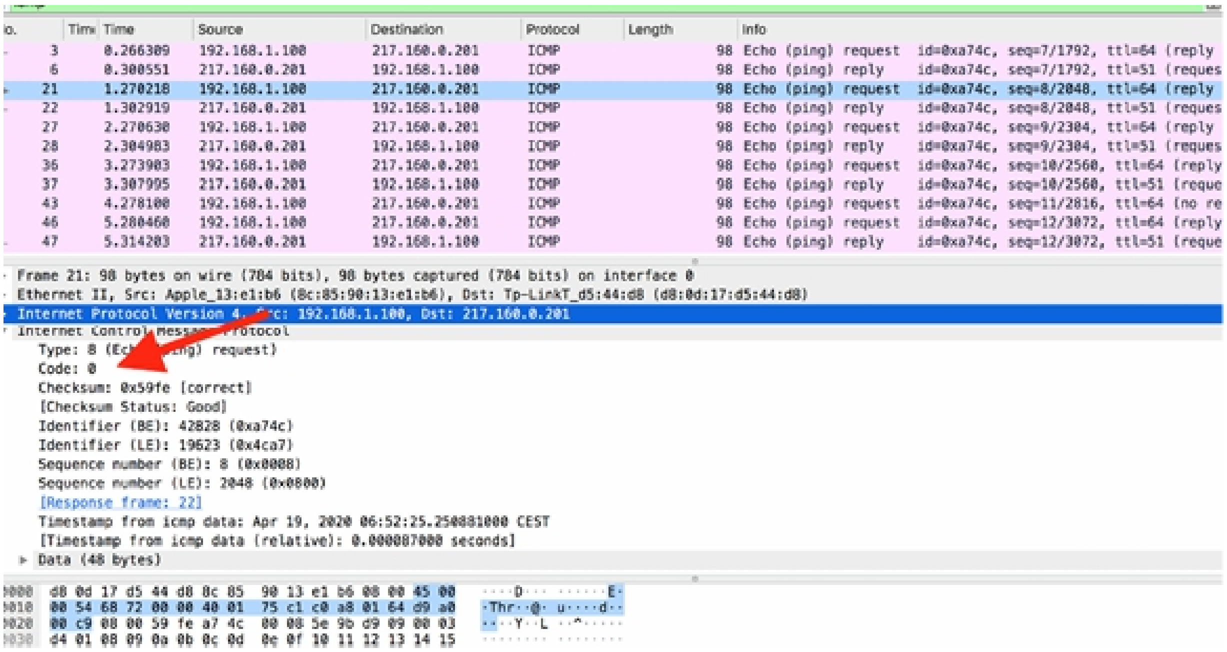
Digital evidence handling procedure recommendations can vary in different ways. Local laws and regulations should be considered to preserve the integrity and admissibility of digital evidence.

In general, it is important to be aware of the legal issues of listening to network traffic. Wireshark provides the ability to eavesdrop on network communications but unauthorized use of Wireshark may be illegal in some countries.

Task 2:

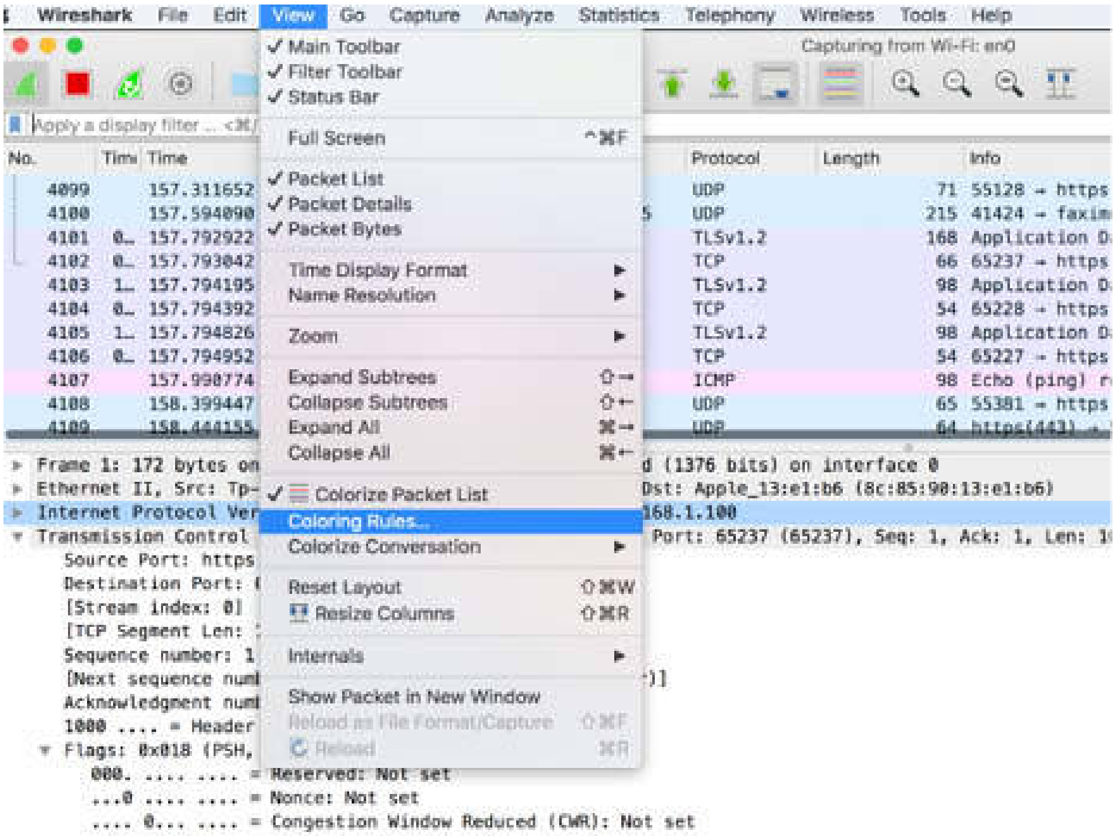
To recognize unusual traffic patterns, it is important to first recognize normal traffic patterns. The baseline process described in the previous labs is essential in differentiating traffic types.

Using penetration testing, reconnaissance, and mapping tools to generate unusual traffic enables you to correlate this type of traffic with Wireshark tools. For example, if you detect something strange in the packet structure, you can assume that something unusual is happening. A specific example is when in some of the ICMP Echo Request packets, the code field is set to 9. This is unusual because as per the specification, the code field of an ICMP Echo Request packet should be 0. In the figure below, a classic ICMP sequence is reported (for a ping request issued to a public IP address, such as google.com, in the terminal window).

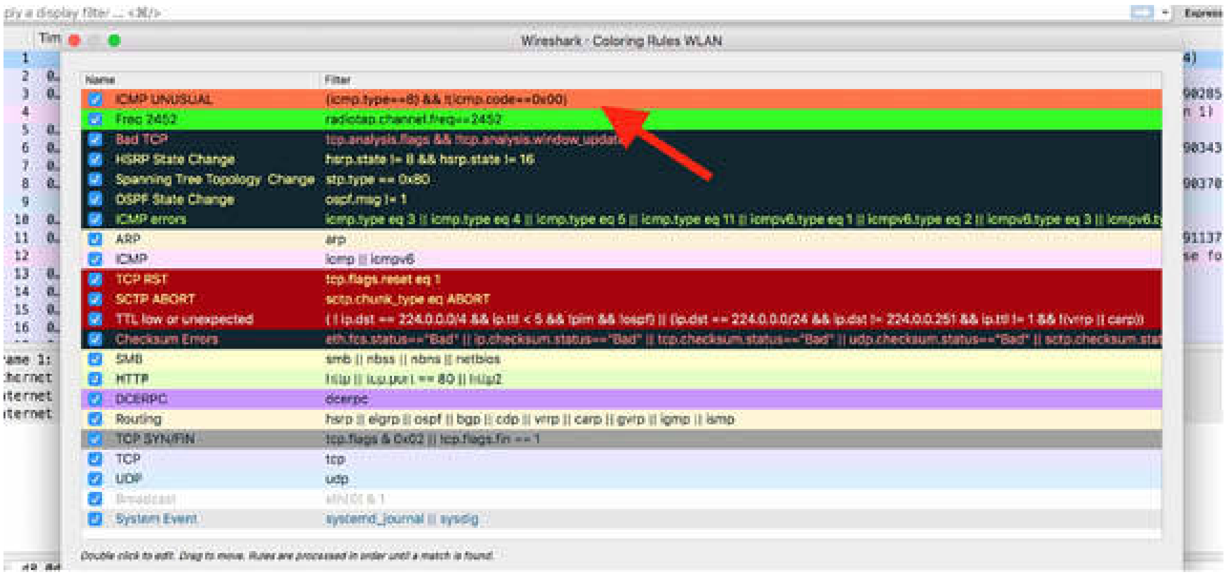


Task 3:
To spot unusual traffic more efficiently, you can create coloring rules to highlight the traffic.

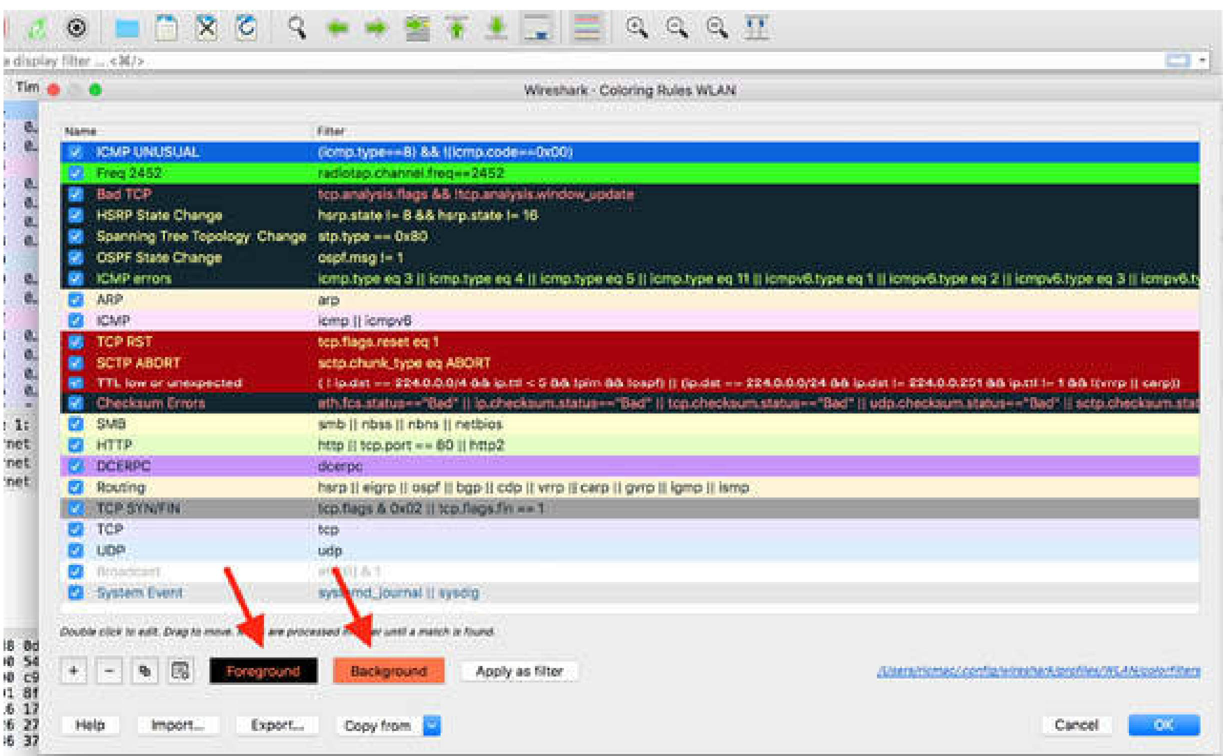
To create a new coloring rule, on the main menu, click View > Coloring Rule, as shown in the figure below.



The Coloring Rules dialog box is displayed. You can add a new filter to clearly display the unusual packets in the Packet List pane. For example, to highlight the unusual ICMP Echo Request used by NetScanTools, create a coloring rule by using the syntax (icmp.type==8) && !(icmp.code==0x00) , as shown in the figure below.



Click the rule to customize its background and foreground color in the Packet List pane, as shown in the figure below.



The following list describes some coloring rules to detect:

- Nmap General Traffic:

  (tcp.flags==0x00) || (tcp.options.wscale_val==10) || (tcp.options.mss_val < 1460) || (tcp.flags==0x29) && tcp.urgent_pointer==0 || (tcp.flags==0x02 && frame[42:4] != 00:00:00:00) || (tcp.flags==0x02 && tcp.window_size < 65535 && tcp.options.wscale_val > 0)

- Small WinSize SYN (probably indicating a discovery packet):

  tcp.window_size < 65535 && tcp.flags.syn==1

- Default IRC TCP Ports 6666–6669 (an IRCP traffic bot issue):

  tcp.port==6666 || tcp.port==6667 || tcp.port==6668 || tcp.port==6669

- DNS Answers > 5 (A bot server is probably listed in this packet. It is a suspect and not a sure problem. Look for a number of dissimilar IP addresses in the response.)

  dns.count.answers > 5

- ICMP Protocol Unreachable (An IP scan might be underway):

  icmp.type==3 && icmp.code==2

- ICMP Response to TCP Packet (The sender probably has been firewalled):

  (icmp) && (tcp)

- ICMP Type 3/Code 4 (A black hole might have been detected):

  icmp.type==3 && icmp.code==4

- ICMP Types 13, 15 or 17 (OS fingerprinting):

  icmp.type==13 || icmp.type==15 || icmp.type==17

- Non-Standard ICMP Echo Request (Look into and try to detect the application used):

  icmp.type==8 && !icmp.code==0

- TCP Window Size < 1460 (The receiver probably is trying to stop the data transfer):

  tcp.window_size < 1460 && tcp.flags.reset==0

- TCP Zero Window (The receiver is stopping the data transfer):

  (tcp.window_size==0) && (tcp.flags.reset==0)

- Look for any HTTP GET packets that have exe, zip, jar, or tar files:

  http.request.method ==”GET” && http matches “\(?i)(exe|zip|jar|tar)”

Task 4:
There are many security tools that can complement the packet capture abilities of Wireshark. The following list provides some of the security tools:

- Nessus (**www.nessus.org**)
- Snort (**www.snort.org**)
- Netcat (**netcat.sourceforge.net**)
- NetScanTools (**www.netscantools.com**)
- Metasploit Framework (**www.metasploit.com**)
- Hping2 (**www.hping.org**)
- Kismet (**www.kismetwireless.net**)
- Tcpdump (**www.tcpdump.org**)
- Cain and Abel (**www.oxid.it**)
- John the Ripper (**www.openwall.com/john**)
- Ettercap (**ettercap.sourceforge.net**)
- Nikto (**www.cirt.net/nikto2**)

On the official Wireshark Wiki page (**https://wiki.wireshark.org/Tools**), you can find a more comprehensive list of tools that complement Wireshark’s capabilities along with a concise description of cases where each tool can be useful.

Notes:
Repeat the previous steps to capture some traffic in Wireshark and try to apply some of the suggested coloring rules to gain confidence in detecting unusual traffic patterns.

Take a look at Wireshark's complementary tools and try using some of the tools to have a complete set of instruments for providing evidence in captured files.

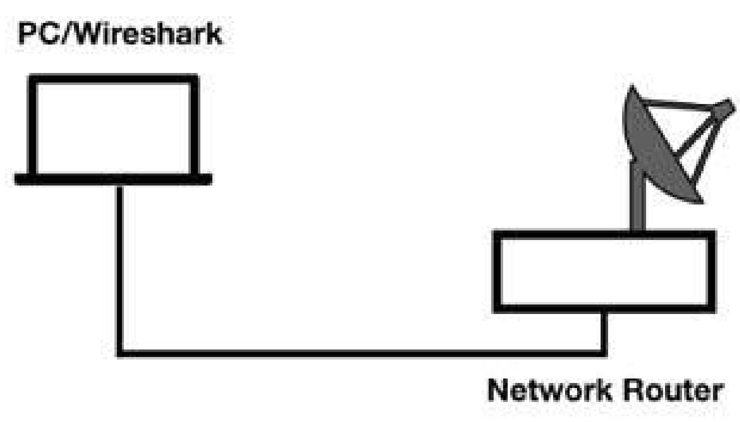
Lab 93. Detect Scanning and Discovery Processes

Lab Objective:
Learn the purpose of the discovery and reconnaissance process.

Lab Purpose:
Understand how to properly use scans and discovery tools for the discovery process.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/WiFi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



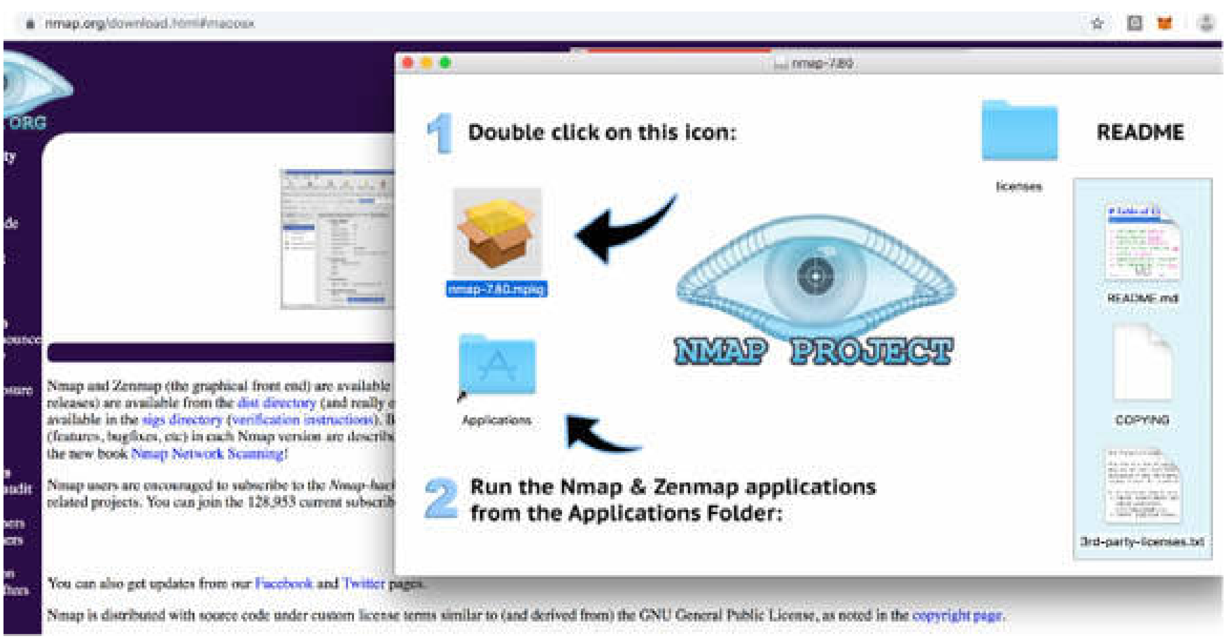
Lab Walkthrough:

Task 1:
In a certain way, just as a criminal may investigate every detail of a bank and how it works before robbing it, malicious programs and processes may investigate open ports and working hosts before attempting an exploit. This is the reason why it is important to identify the discovery and reconnaissance processes as soon as possible because avoiding an eventual attack depends on your timeliness. Understanding the purpose of these discovery methods will help you in knowing in advance what the attacker is looking for and what options are available to block the malicious traffic.

As mentioned in the previous lab, Nmap is one of the most popular tools used to discover network devices and services, and it is useful for you to understand in detail how Nmap works.

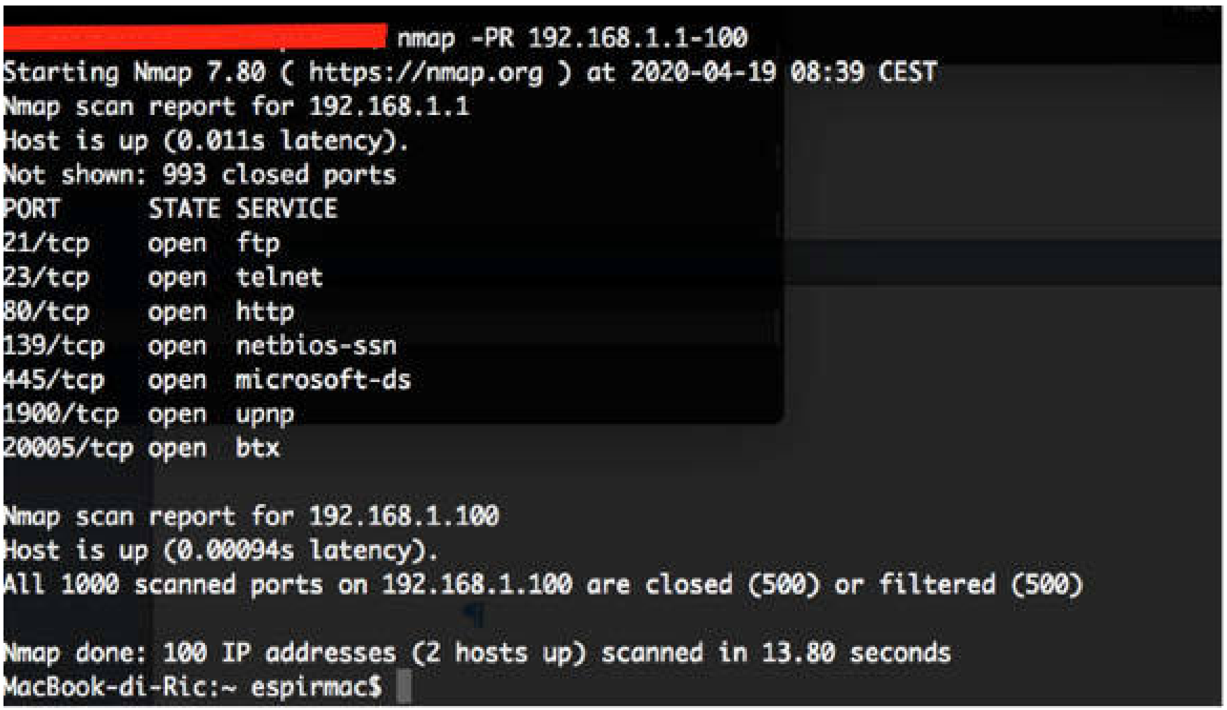
Task 2:
Nmap is a free, multi-platform (Windows, Linux/Unix, Mac OS X) security scanner available at **nmap.org** . Nmap should be run on a network with the permission of the network administrator. It is important to understand how Nmap works alone as well as with Wireshark.

To install Nmap, download an appropriate version for your operating system from **nmap.org** .



Task 3:
ARP scans (or ARP sweeps) are used to find hosts only on the local network. ARP packets do not cross a router because ARP packets are not routable—they do not have an IP header.

In Wireshark, capture the traffic for a few minutes. Open a terminal window and run the nmap command (to scan a range of 100 IPs from the starting IP address given as the argument), as shown in the figure below.



Stop the capture and save the file.

The Nmap parameter for running an ARP scan is –PR (referred to as an ARP ping), but this parameter is rarely used because Nmap automatically uses ARP scans whenever it can (e.g. when the target is on the same Ethernet segment as the source).

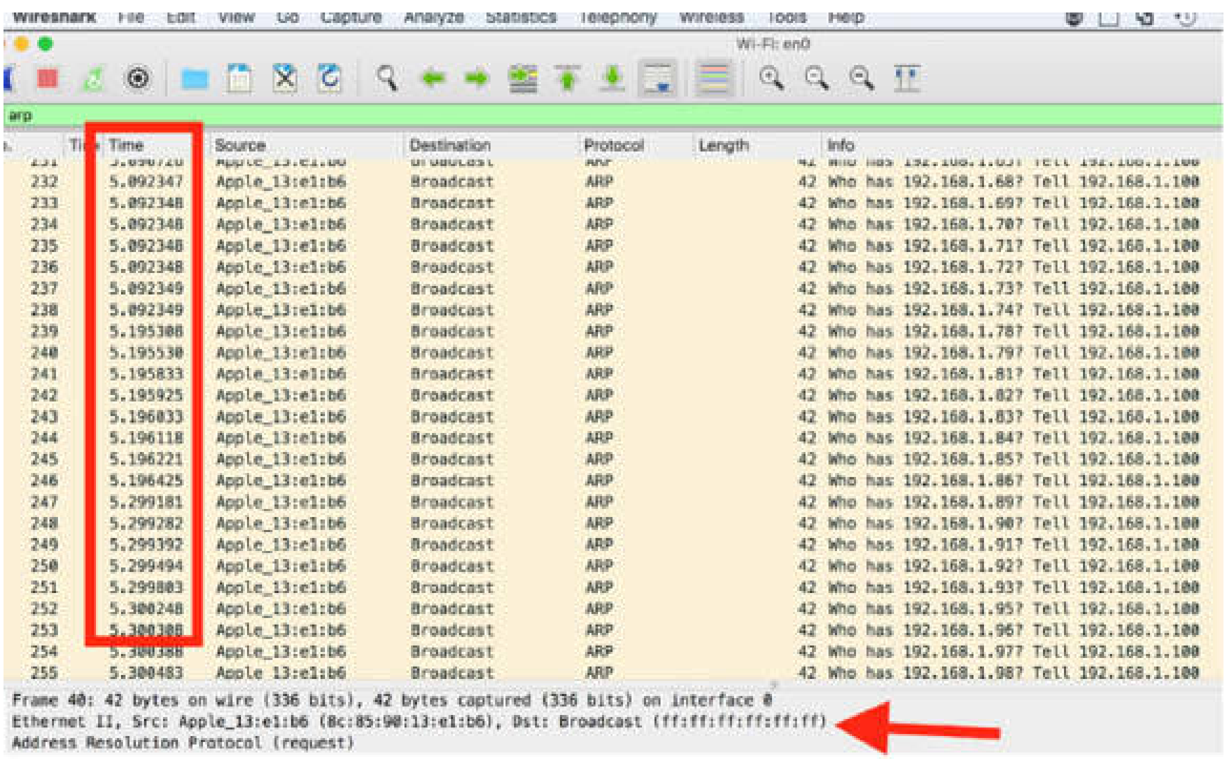
In the result of the example command above, for each "known" port, a statistic related to each IP—found on the network with an open port—is displayed. In the example above, no IP was found on the local network.

The disadvantage of using an ARP scan is that the ARP traffic can't get through a router or any layer-3 device. The advantage of running an ARP scan is that you can discover local devices that may be hidden from other discovery methods by a firewall. If the firewall blocks ICMP-based pings, you can use an ARP scan to discover the device. You cannot disable ARP responses—that would "break" the TCP/IP communications system.

Keep in mind that ARP scans do not cross a router. If you detect an ARP scan taking place, the source and targets will be on the same network on which you are capturing the traffic.

Common ARP scan processes send ARP requests to the broadcast MAC address (0xff:ff:ff:ff:ff:ff). Discovering ARP scan traffic can be difficult if the ARP traffic is not using a high packets per second rate, which would make it clearly visible in the trace file.

In the figure below, the arp display filter is used, and the high packet rate is clearly visible in the Packet List pane. The broadcast MAC address is displayed in the Packet Details pane.



Task 4:
In addition to the ARP scan, you can use ICMP ping sweeps. There are three possible variations of ping sweeps, although, in the majority of cases, you can refer to the ping sweep as a scan using an ICMP Type 8 Echo Request, followed by an ICMP Type 0 Echo Reply.

The other variations are TCP ping scans and UDP ping scans. Both TCP and UDP variations use destination port 7, the Echo port. Most hosts do not support Echo services on TCP or UDP port 7, so using TCP and UDP ping scan methods are not very useful.

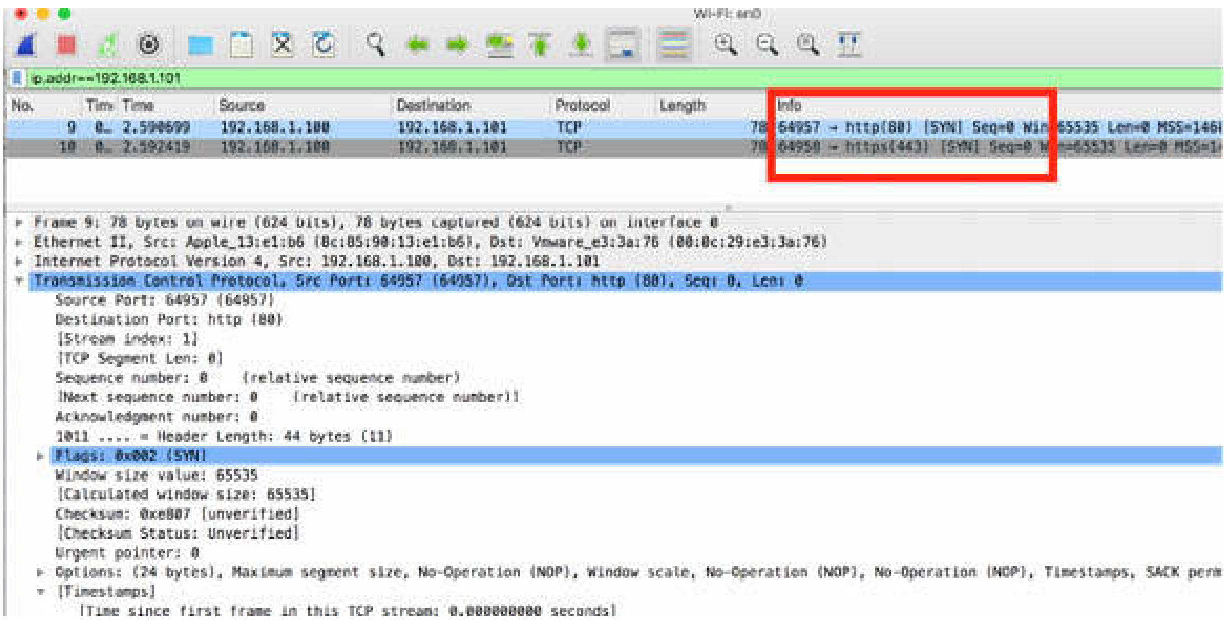
The standard ICMP ping sweeps worked great for many years until firewalls (host and network) started to be configured to block such ICMP packets.

In Wireshark, capture the traffic for a few minutes. Open a terminal window, and run the command nmap –sn TARGET-IP , where TARGET-IP is the host to be discovered, and sn is the Nmap syntax for an ICMP-based ping sweep. -sn means “skip the port scan phase,” and was previously available as -sP , with the mnemonic “Ping scan”.

Stop the capture and save the file. ICMP-based ping sweeps are easy to detect with a simple filter icmp.type==8 || icmp.type==0 . The ICMP echo requests use the ICMP Type 8 whereas the ICMP echo replies use the ICMP Type 0.

In this example, considering that a non-privileged user sent a ping sweep, by default, Nmap sends only one TCP SYN packet to port 80 and one TCP

SYN packet to port 443 of the target. The result is similar to the one displayed in the figure below by using the display filter ip.addr == TARGET-IP .



If a target blocks ICMP pings, you can use a TCP or UDP port scan to identify hosts on the network.

Notes:

Repeat the previous steps and first try to identify the topology and the host belonging to your local network by using ARP sweeps. Then try to identify the result for each host present in the network by using ICMP ping sweeps.

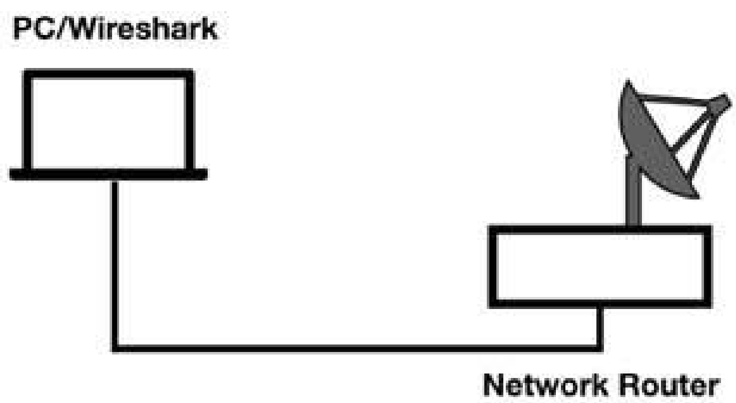
Lab 94. TCP Port Scan

Lab Objective:
Learn the purpose of TCP port scans.

Lab Purpose:
Understand how to properly use TCP port scans in the context of the discovery process.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:

Port scans are used to discover targets and detect services offered on a target. Most of the popular services, such as web browsing and email services, run over TCP.

Some of the popular services that could be scanned over a network are:

- FTP: TCP port number 21
- Secure Shell (SSH): TCP port number 22
- Telnet: TCP port number 23
- SMTP: TCP port number 25
- HTTP: TCP port number 80
- POP: TCP port number 110
- NTP: TCP port number 123
- Endpoint Mapper Resolution: TCP port number 135
- NetBIOS Session Service: TCP port number 139
- HTTP over SSL/TLS: TCP port number 443
- Microsoft Directory Services: TCP port number 445
- Microsoft SQL Server: TCP port number 1433
- VNC Server: TCP port number 5900
- HTTP Alternate: TCP port number 8080

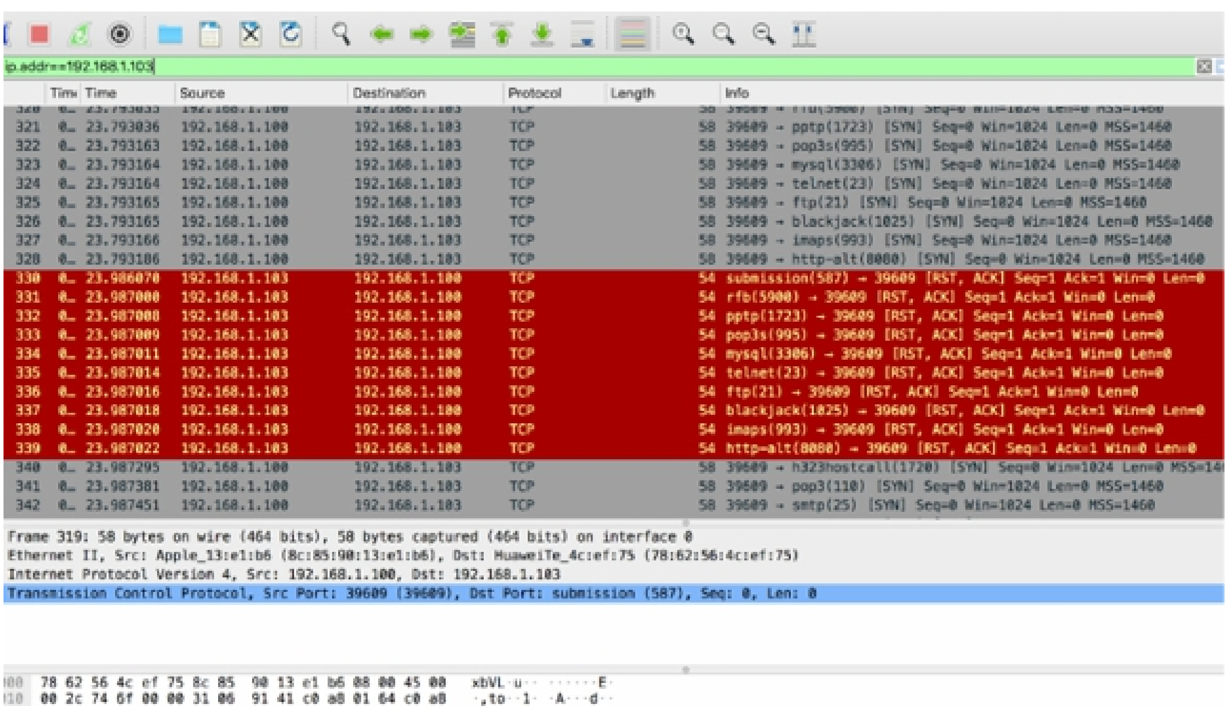
There are several variations of TCP scans. Considering a basic TCP connection establishment, one TCP host sends a TCP SYN packet to a port on a target. The target must respond with either an RST packet (port is not open) or a SYN/ACK packet (port is open). This method provides a quick connectivity test.

Task 2:
Nmap, by default, uses a TCP half-open scan—also known as "stealth scan". In fact, if a port is open, Nmap does not finish the three-way handshake (SYN/SYN-ACK/ACK) to make a complete connection. On receiving SYN/ACK from a target, the host running Nmap generates a TCP Reset to terminate the connection attempt.

In Wireshark, capture the traffic for a few minutes on an active network interface. Open a terminal window, and run the command nmap –sS TARGET-IP with root privileges, where TARGET-IP is the target IP address (192.168.1.103) to be scanned, as shown in the figure below.

As shown in the figure above, the result of the nmap command is that all ports are closed.

Stop the capture and save the file. In the filter toolbar, enter ip.addr==192.168.1.103 to display all traffic related to the target IP address. The result will be similar to as shown in the figure below.



As shown in the figure above, only RST packets are available. A TCP Reset response indicates that the target port is closed. If no response is received, you cannot assume that the port is open or closed. The TCP SYN or the response may have been dropped along the way. Advanced port scanners, such as Nmap, retransmit probe packets to distinguish intentional packet filtering from the occasional packet loss, which is expected on busy networks.

If you receive an ICMP Destination Unreachable (Type 3) response with a code 1, 2, 3, 9, 10, or 13, the port is probably firewalled.

In a TCP half-open scan, because the scanner does not complete the three-way handshake, a target can look at the list of open connections, but the scanning host will not show up. This is why it's called "stealth scan". The TCP half-open scan is the desired type of TCP scan for stealthiness and resource preservation on the target.

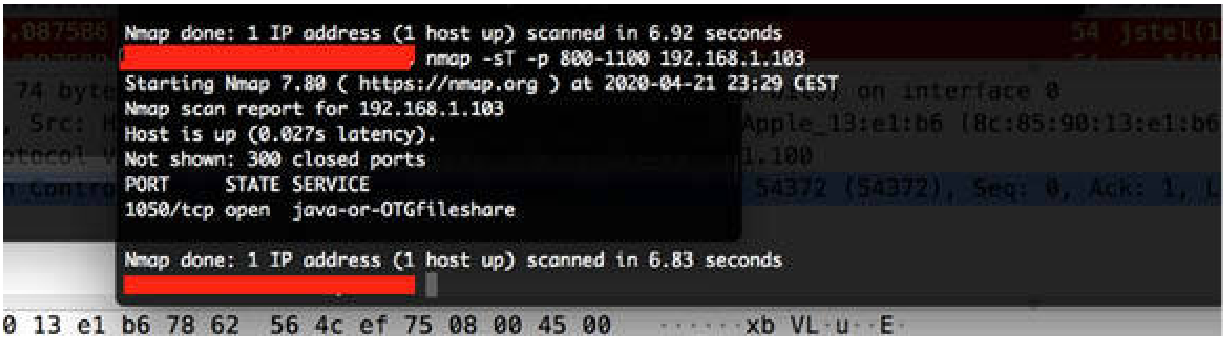
TCP scans can be difficult to detect with Wireshark unless the scans are in close proximity, and they are evident in the trace file, as shown in the figure above. An unusually high number of RSTs or a high number of SYN/ACKs without data transfer is a strong indicator of a TCP scan being underway.

Task 3:

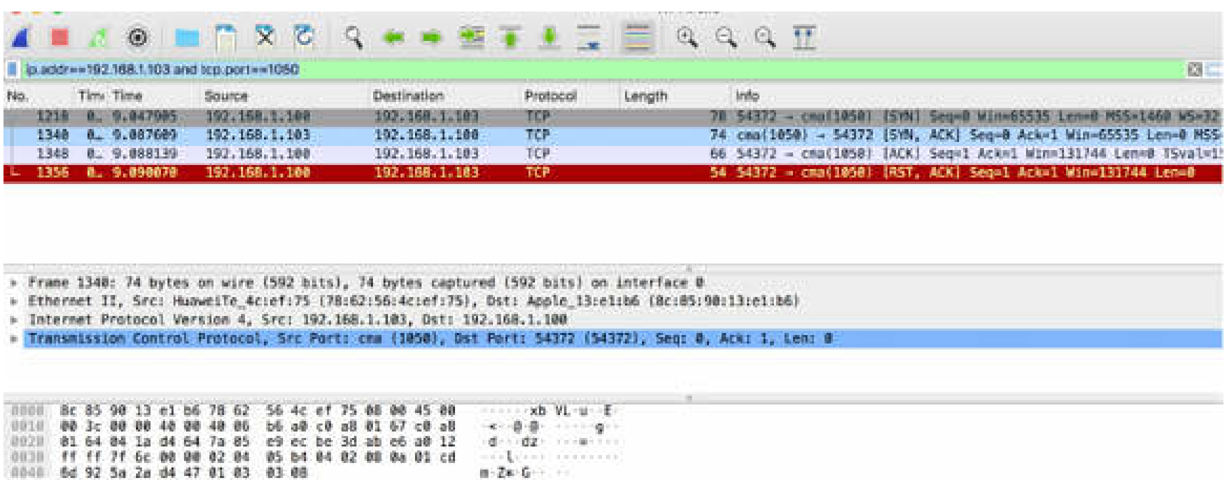
TCP full-connect scans complete the three-way handshake after receiving the SYN/ACK packet from an open port. A TCP Reset response indicates that the target port is closed. If no response is received, you cannot assume that the port is open or closed. The TCP SYN or the response may have been dropped along the way.

Identify a Linux PC target where a sample TCP port is open. In this example, the port 1050 on the target IP 192.168.1.103 has been opened with the command `sudo ufw allow 1050/tcp` .

In Wireshark, capture the traffic for a few minutes. Open a terminal window, and run the `nmap –sT –p 800-1100 192.168.1.103` command, where 192.168.1.103 is the target IP address to be scanned, and 800–1100 is the port range to be scanned. The result of the command is shown in the figure below. You can identify that the port 1050 is open.



Stop the capture and save the file. In the filter toolbar, enter ip.addr==192.168.1.103 and tcp.port==1050 to inspect the effect only on the open port. The result will be similar to the one shown in the figure below.

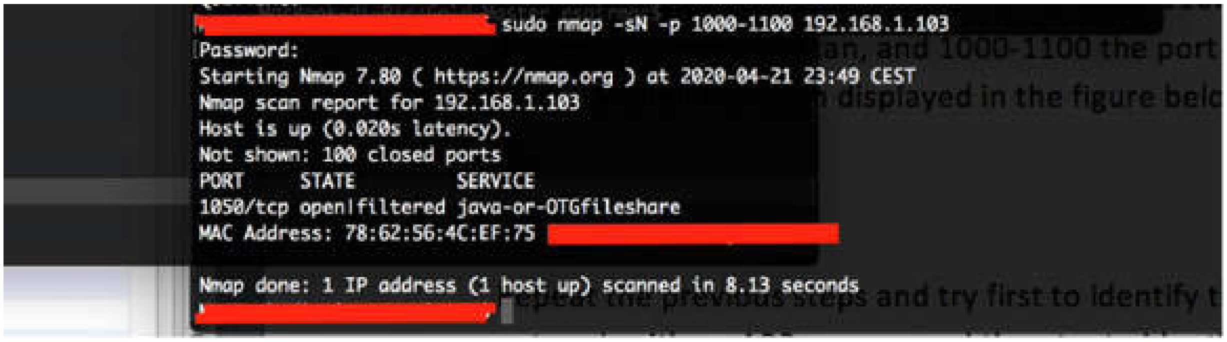


Based on the Packet List pane shown above, it is clear that the scanner completed the three-way handshake. The scanner sends the ACK packet in packet #1348. This is a classical pattern of a TCP full-connect scan.

Task 4:

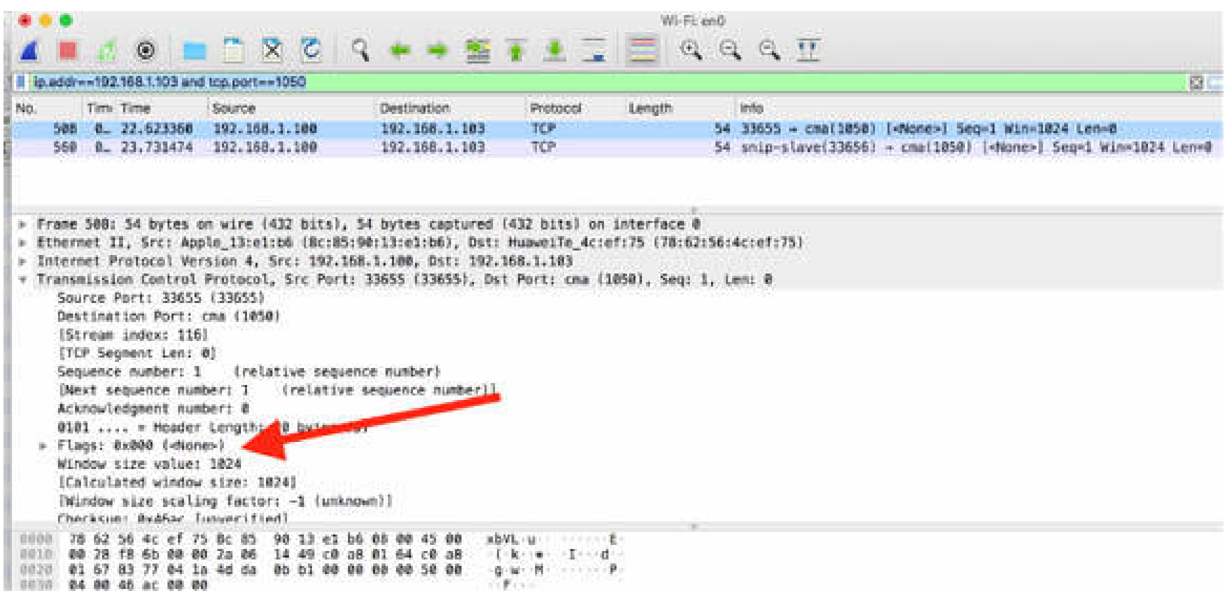
In Wireshark, capture the traffic for a few minutes on an active network interface. Open a terminal window, and run the command sudo nmap –sN –p 1000-1100 192.168.1.103 with root privileges, where 192.168.1.103 is the target

IP address to be scanned, and 1000–1100 is the port range to be scanned. The result will be similar to the one shown in the figure below.



In this case, a null scan has been applied. Null scans use an unusual TCP packet format in which none of the TCP flags is set. No response to a null scan indicates that the port is either open or filtered. A TCP Reset response indicates that the port is closed. An ICMP Destination Unreachable (Type 3) response with a code 1, 2, 3, 9, 10, or 13 indicates that the port is probably firewalled.

In the filter toolbar, enter ip.addr==192.168.1.103 and tcp.port==1050 . The result will be similar to the one displayed in the figure below, where you can identify the TCP flags with value NULL in the Packet Details pane.



To detect null scans, you can create a coloring rule or a display filter (tcp.flags==0x000) for TCP packets for which no TCP flags are set.

Task 5:
The last three possible TCP scans applicable with Nmap are Xmas scan, FIN scan, and ACK scan, as described below:

- Xmas scans have the URG, FIN, and PUSH flags set. In case of no response to a Xmas scan, it is clear that the port is either open or filtered. A TCP Reset response indicates that the port is closed. An ICMP Destination Unreachable (Type 3) response with a code 1, 2, 3, 9, 10, or 13 indicates that the port is probably firewalled.
  To start a Xmas scan with Nmap, open a terminal window, and run the command nmap –sX TARGET-IP . To detect Xmas scans, create a coloring rule or a display filter (tcp.flags==0x029) for TCP packets that have only these three flags set.
- FIN scans only have the TCP FIN bit set. No response to a FIN scan indicates that the port is either open or filtered. A TCP Reset response indicates the port is closed. An ICMP Destination Unreachable (Type 3) response with code 1, 2, 3, 9, 10, or 13 indicates that the port is probably firewalled.
  To start a FIN scan with Nmap, open a terminal window, and run the command nmap –sF TARGET-IP . Detecting FIN scans can be difficult unless the scans are in close proximity and evident in the trace file.
- ACK scans are typically used to check firewall rules to see if ports are explicitly blocked. ACK scans are not used to identify open ports unless the window scan technique (-sW with Nmap) is also used.

An ACK scan sends a TCP packet with only the ACK (Acknowledge) flag bit set to 1—there is no TCP handshake preceding the ACK scan.

A TCP RST response indicates that the port is unfiltered, which does not indicate that the port is open. A TCP scan can be used to determine whether or not the port is open. An ICMP Destination Unreachable response (Type 3, codes 1, 2, 3, 9, 10, or 13) indicates that the port is likely filtered. No response also indicates that the port is likely filtered.

Wireshark's default coloring rules contain a coloring rule for the ICMP Destination Unreachable packets (black background, vivid green foreground). The rule syntax is icmp.type eq 3 || icmp.type eq 4 || icmp.type eq 5 || icmp.type eq 11 || icmpv6.type eq 1 || icmpv6.type eq 2 || icmpv6.type eq 3 || icmpv6.type eq 4.

To start an ACK scan with Nmap, open a terminal window, and run the command nmap –sA TARGET-IP .

Notes:
Repeat the previous steps to gain confidence with all presented types of TCP scans. Try to identify a target machine by opening some TCP ports to be tested and then capture traffic with Wireshark while scanning the ports with Nmap.

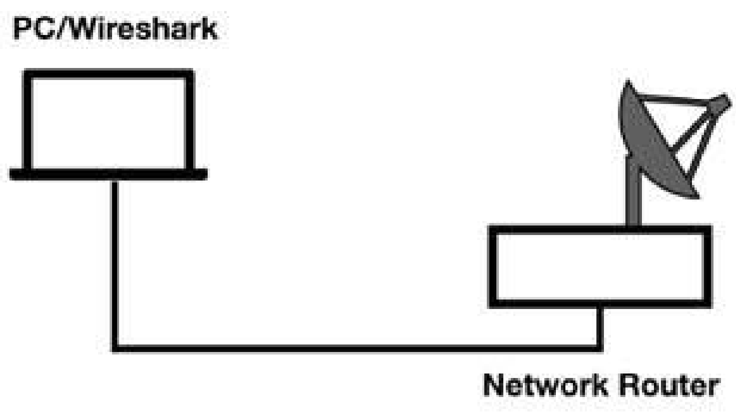
Lab 95. UDP and IP Scan

Lab Objective:
Learn the purpose of the UDP port and IP protocol scan.

Lab Purpose:
Understand how to properly use the UDP port and IP protocol scan in the context of the discovery process.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:

Although the majority of popular services, such as web browsing and email services, run over TCP protocol, there are certain specific services that run over UDP. Some of the UDP-based services are:

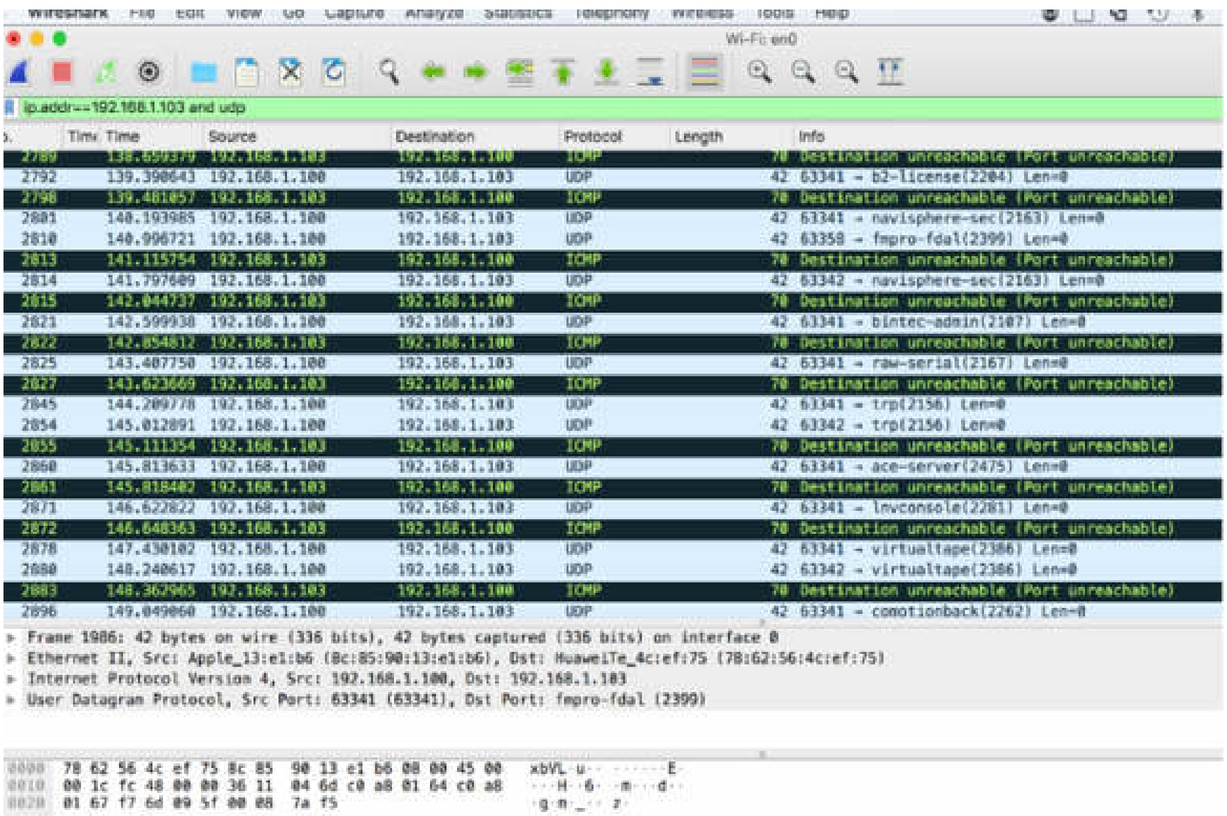
- DNS: UDP Port Number 53
- SNMP: UDP Port Number 161/162
- DHCP: UDP Port Number 67/68
- SIP: UDP Port Number 5060
- Microsoft Endpoint Mapper: UDP Port Number 135
- NetBIOS Name Service: UDP Port Number 137/139

UDP port scans can be used to find services running on UDP ports or as a simple connectivity test. An ICMP Destination Unreachable/Port Unreachable response indicates that the service is not available on the target. No response indicates that the service might be available, or the service might just be filtered. Any other ICMP response indicates that the service is filtered.

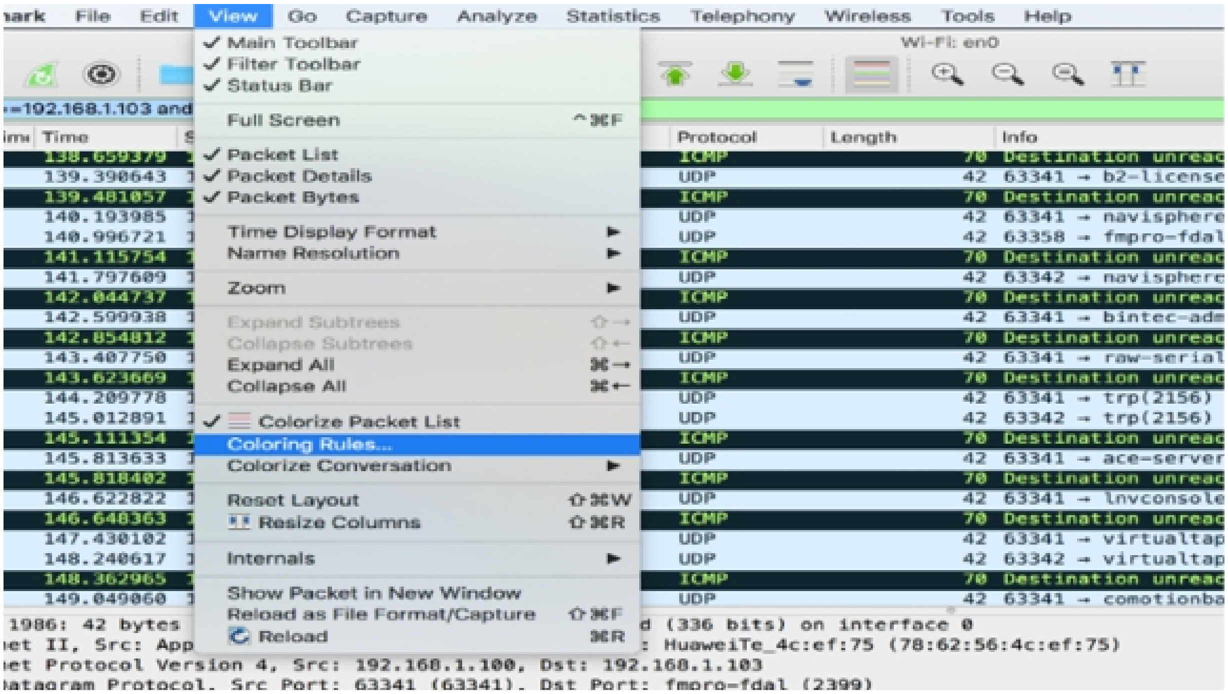
An unusually high number of ICMP Destination Unreachable/Port Unreachable packets or a high number of unanswered UDP packets is a strong indicator of a UDP scan being underway.

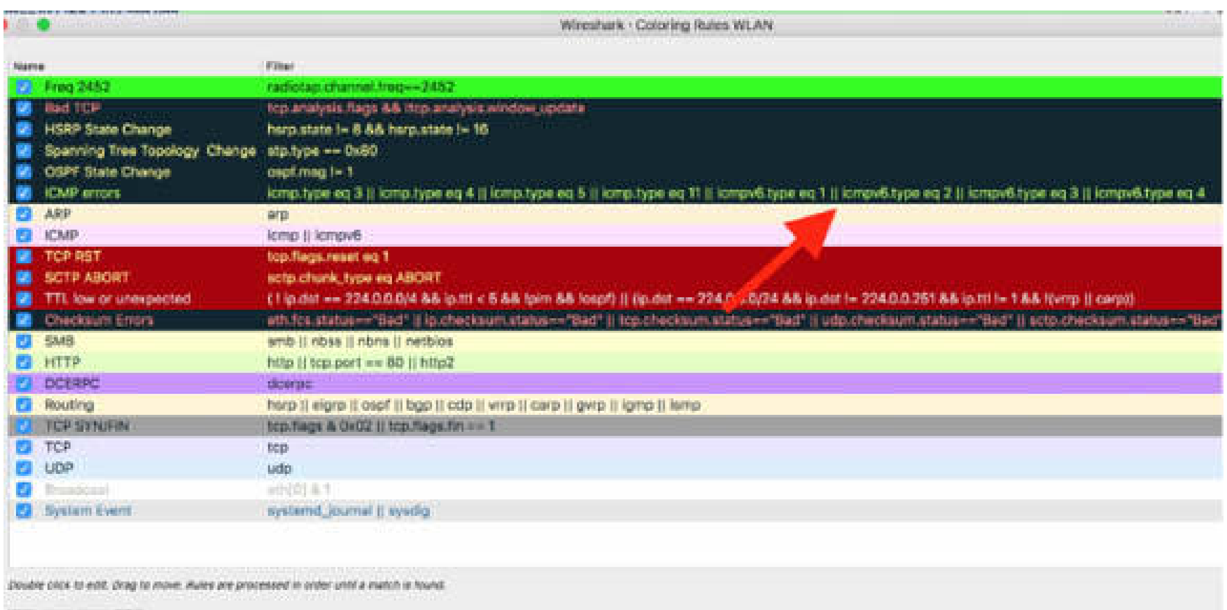
In Wireshark, capture the traffic for a few minutes on an active network connection. Open a terminal window, and run the command `sudo nmap -sU -p 2100-2500 192.168.1.103` , where 192.168.1.103 is the target IP address you need to scan, and 2100–2500 is the port range to be scanned.

Stop the capture and save the file. In the filter toolbar, enter `ip.addr==192.168.1.103 and udp` . You can observe a large number of ICMP Destination Unreachable (Port Unreachable) messages confirming that the scan is in progress, but all the ports are closed. The result will be similar to the one shown in the figure below.



Wireshark's default coloring rules contain a coloring rule for the ICMP Destination Unreachable packets (black background, vivid green foreground). The rule syntax is icmp.type eq 3 || icmp.type eq 4 || icmp.type eq 5 || icmp.type eq 11 || icmpv6.type eq 1 || icmpv6.type eq 2 || icmpv6.type eq 3 || icmpv6.type eq 4 . To verify it, on the main menu, select View > Coloring Rules, as shown in the figures below.





Task 2:

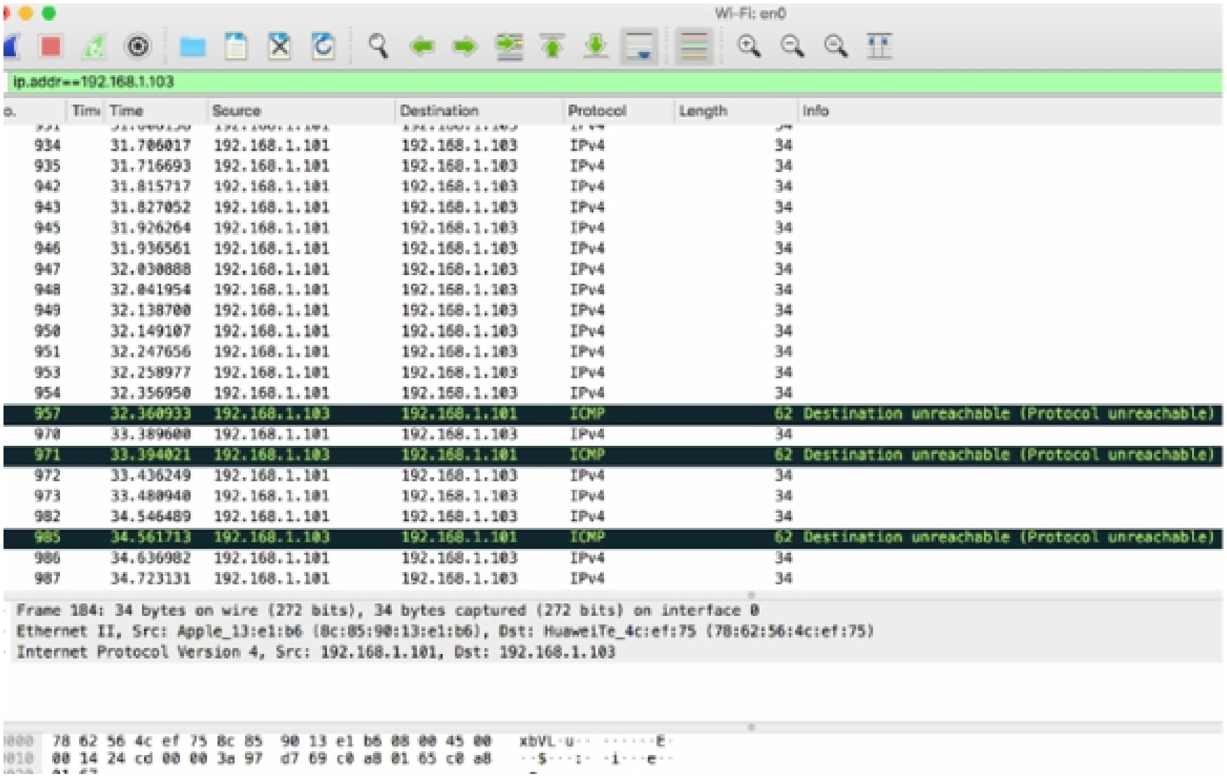
IP protocol scans are designed to locate services running directly over IP. For example, an IP scan can locate a device that supports the Enhanced Interior Gateway Routing Protocol (EIGRP). Some of the services that run directly over IP Protocol are:

- ICMP: IP Protocol Number 1
- IGMP: IP Protocol Number 2
- TCP: IP Protocol Number 6
- EGP: IP Protocol Number 8
- IGP (used for IGRP): IP Protocol Number 9
- UDP: IP Protocol Number 17

When a protocol is not supported on a target and from Nmap, you try an IP scan, the target may respond with an ICMP Destination Unreachable/Protocol Unreachable response (Type 3/Code 2). If no response is received, you can assume the service is available, or the response is filtered (open|filtered).

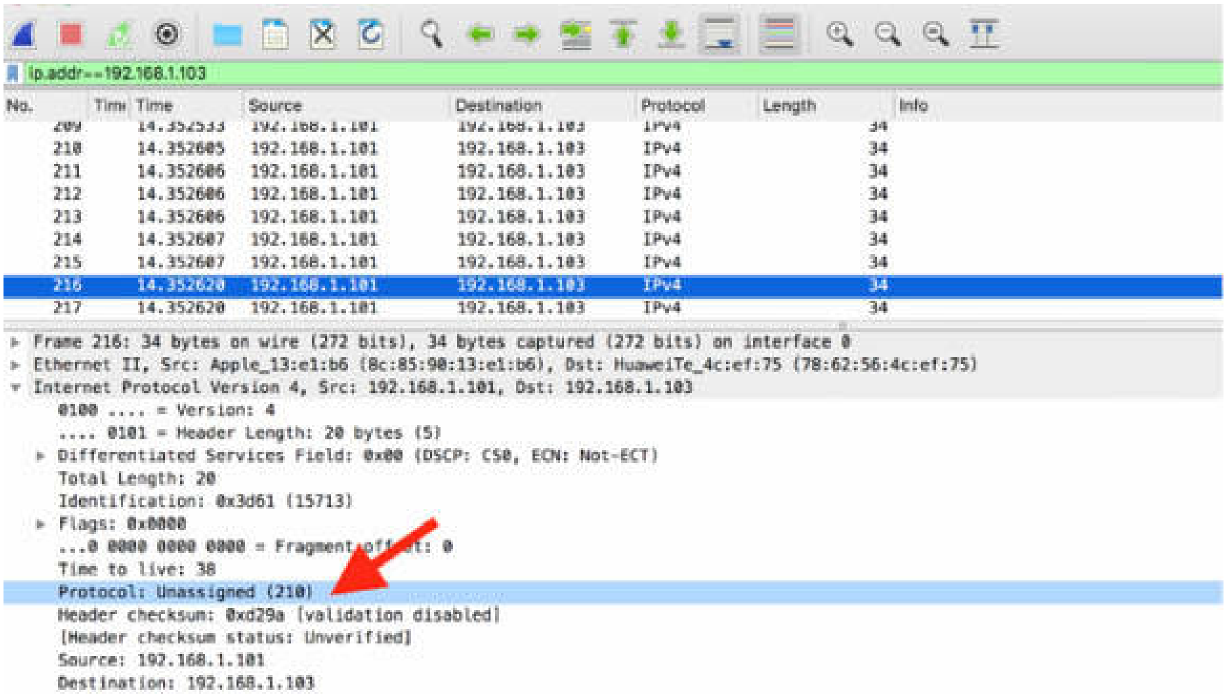
In Wireshark, capture the traffic for a few minutes on an active network interface. Open a terminal window, and run the sudo nmap –sO 192.168.1.103 command with administrative privileges, where 192.168.1.103 is the target IP address you need to scan.

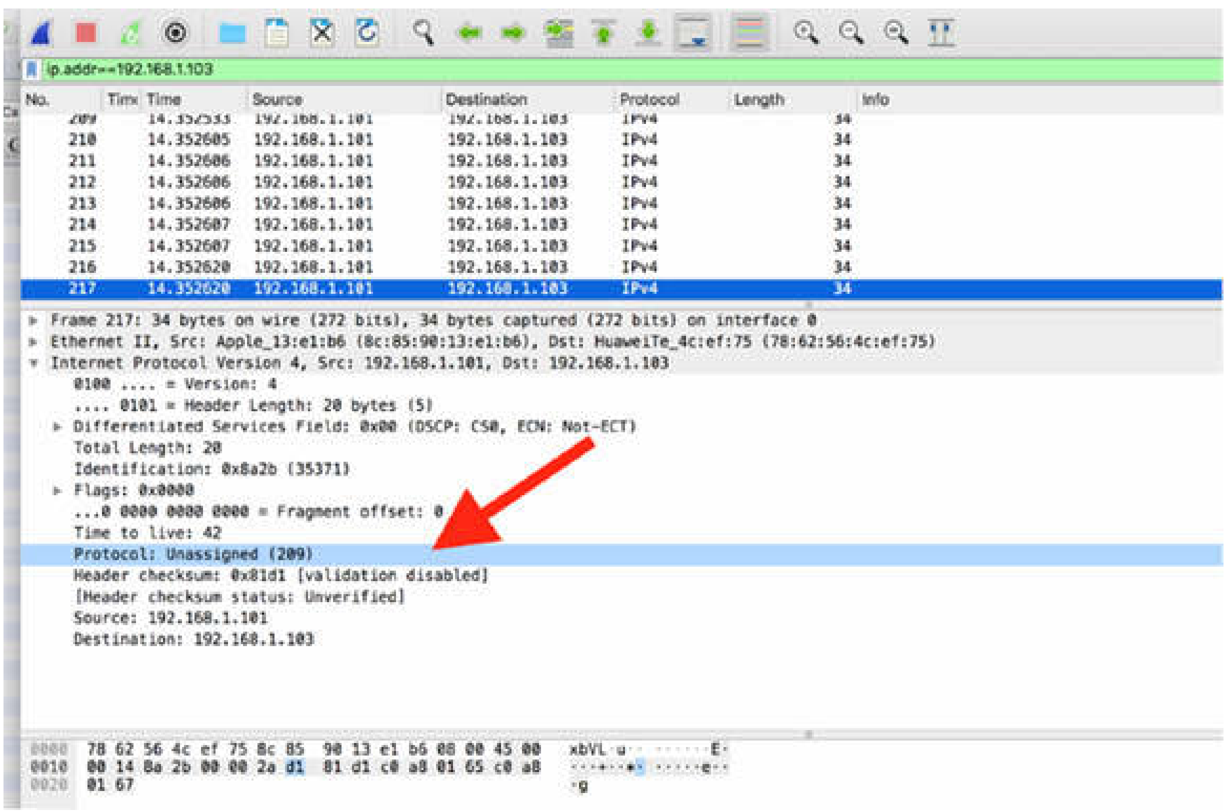
Stop the capture and save the file. In the filter toolbar, enter ip.addr == 192.168.1.103 , as shown in the figure below.



To detect IP protocol scans, you can create a coloring rule or a display filter (icmp.type==3 && icmp.code==2) for ICMP Type 3/Code 2 packets. This will allow you to detect a lot of ICMP packets in the capture.

It is important to note that the value in the IP Header of the Protocol field is altered for each packet during an IP Scan. To verify, inspect the packet details of two consecutive packets sent towards the target IP, as shown in the figures below.


ip.addr==192.168.1.103
No. Time Time Source Destination Protocol Length Info
209 14.352533 192.168.1.101 192.168.1.103 IPv4 34
210 14.352605 192.168.1.101 192.168.1.103 IPv4 34
211 14.352606 192.168.1.101 192.168.1.103 IPv4 34
212 14.352606 192.168.1.101 192.168.1.103 IPv4 34
213 14.352606 192.168.1.101 192.168.1.103 IPv4 34
214 14.352607 192.168.1.101 192.168.1.103 IPv4 34
215 14.352607 192.168.1.101 192.168.1.103 IPv4 34
216 14.352620 192.168.1.101 192.168.1.103 IPv4 34
217 14.352620 192.168.1.101 192.168.1.103 IPv4 34
Frame 216: 34 bytes on wire (272 bits), 34 bytes captured (272 bits) on interface 0
Ethernet II, Src: Apple_13:e1:b6 (8c:85:90:13:e1:b6), Dst: HuaweiTe_4c:ef:75 (78:62:56:4c:ef:75)
Internet Protocol Version 4, Src: 192.168.1.101, Dst: 192.168.1.103
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 20
Identification: 0x3d61 (15713)
Flags: 0x0000
...0 0000 0000 0000 = Fragment offset: 0
Time to live: 38
Protocol: Unassigned (210)
Header checksum: 0xd29a [validation disabled]
[Header checksum status: Unverified]
Source: 192.168.1.101
Destination: 192.168.1.103


ip.addr==192.168.1.103
No. Time Time Source Destination Protocol Length Info
209 14.352533 192.168.1.101 192.168.1.103 IPv4 34
210 14.352605 192.168.1.101 192.168.1.103 IPv4 34
211 14.352606 192.168.1.101 192.168.1.103 IPv4 34
212 14.352606 192.168.1.101 192.168.1.103 IPv4 34
213 14.352606 192.168.1.101 192.168.1.103 IPv4 34
214 14.352607 192.168.1.101 192.168.1.103 IPv4 34
215 14.352607 192.168.1.101 192.168.1.103 IPv4 34
216 14.352620 192.168.1.101 192.168.1.103 IPv4 34
217 14.352620 192.168.1.101 192.168.1.103 IPv4 34
Frame 217: 34 bytes on wire (272 bits), 34 bytes captured (272 bits) on interface 0
Ethernet II, Src: Apple_13:e1:b6 (8c:85:90:13:e1:b6), Dst: HuaweiTe_4c:ef:75 (78:62:56:4c:ef:75)
Internet Protocol Version 4, Src: 192.168.1.101, Dst: 192.168.1.103
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 20
Identification: 0x8a2b (35371)
Flags: 0x0000
...0 0000 0000 0000 = Fragment offset: 0
Time to live: 42
Protocol: Unassigned (209)
Header checksum: 0x81d1 [validation disabled]
[Header checksum status: Unverified]
Source: 192.168.1.101
Destination: 192.168.1.103
0000 78 62 56 4c ef 75 8c 85 90 13 e1 b6 08 00 45 00 xbVL·u·· ······E·
0010 00 14 8a 2b 00 00 2a d1 81 d1 c0 a8 01 65 c0 a8 ···+··*· ·····e··
0020 01 67 ·g

Notes:

Repeat the previous steps to scan a target of your choice through UDP and find some running UDP services. Gain confidence in using coloring rules to discover when a scan is occurring. Perform an IP Protocol scan on a target and use Nmap to identify the type of packets sent during the scan.

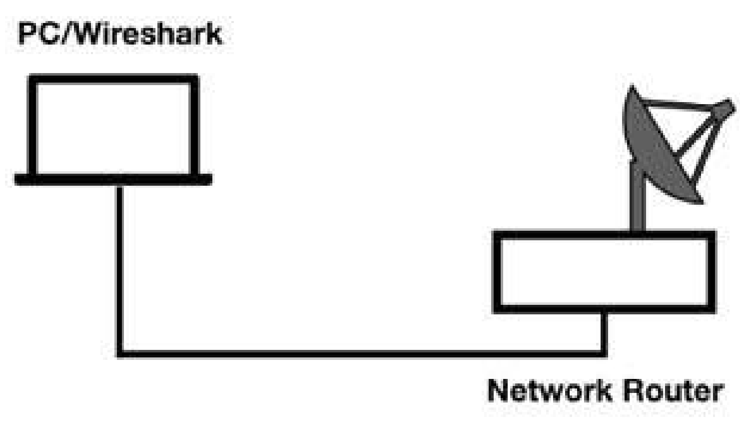
Lab 96. Idle Scan and ICMP Traceroute

Lab Objective:
Learn about idle scan and its purpose, and traceroute discovery.

Lab Purpose:
Understand how to properly use idle scan in the context of the discovery process and the ICMP traceroute discovery.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



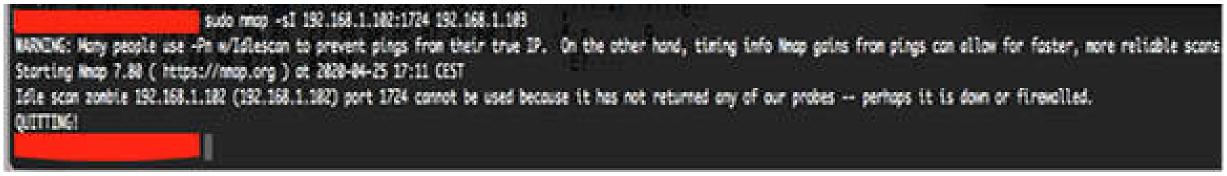
Lab Walkthrough:

Task 1:
Idle scans are used when a scanner is not allowed to talk directly to a target (where a firewall is blocking the traffic based on the scanner's IP address). Idle scans, in general, use another host that can reach the target. This host is referred to as the zombie.

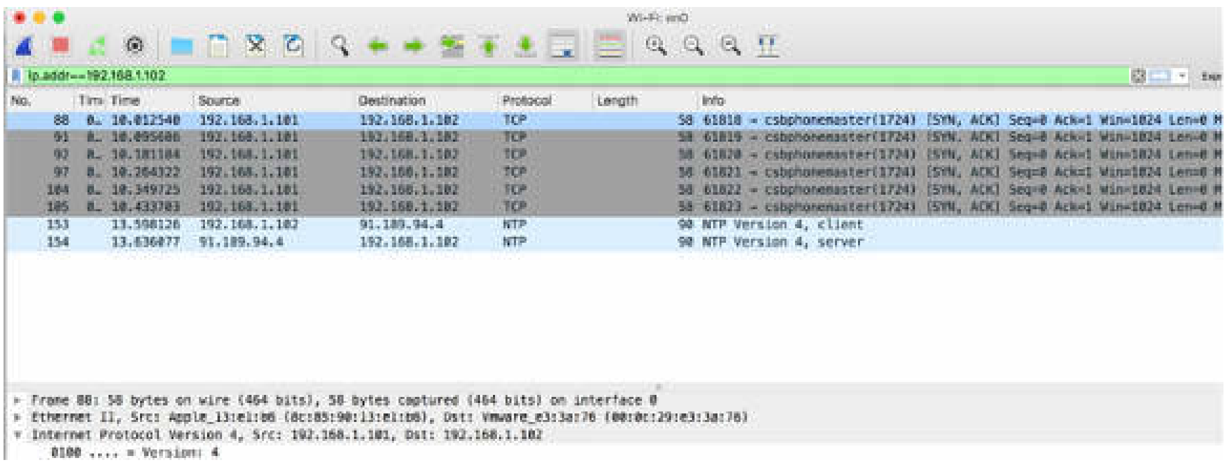
The predefined process for the idle scan can be outlined in the following three steps:

1. The scanner sends a TCP scan to the zombie on a TCP port that is expected to be closed. When the TCP Reset response is received, the scanner notes the IP header ID field value (ID=n). This value typically counts up sequentially for each IP packet transmitted through the TCP/IP stack.
2. The scanner sends a TCP scan to the target by using the zombie's IP address as the source IP address.
3. If the target port is closed, the target responds to the zombie with a TCP Reset packet. The zombie discards this TCP Reset packet. The next IP packet from the zombie is incremented by 1 (ID=n+1).
4. If the target port is open, the target sends SYN/ACK to the zombie. The zombie did not initiate the handshake, and it sends a TCP Reset packet to the target. This causes the zombie's IP ID value to increment by 1 (ID=n+1). The next IP packet from the zombie would be incremented by 2 (ID=n+2).
5. Step 1 is repeated.
6. If the zombie's IP ID field is incremented by 1, you can assume it received a TCP RST from the target, and the target port is not open. If the zombie's IP ID value is incremented by 2, you can assume the port is open at the target.

Task 2:
In Wireshark, capture the traffic for a few minutes on an active network interface. Identify a target IP and a zombie IP in your network. Open a terminal window, and run the command `nmap –sI 192.168.1.102:1724 192.168.1.103` , where 192.168.1.102 is the IP address of the zombie and 192.168.1.103 is the IP address of the target, as shown in the figure below.

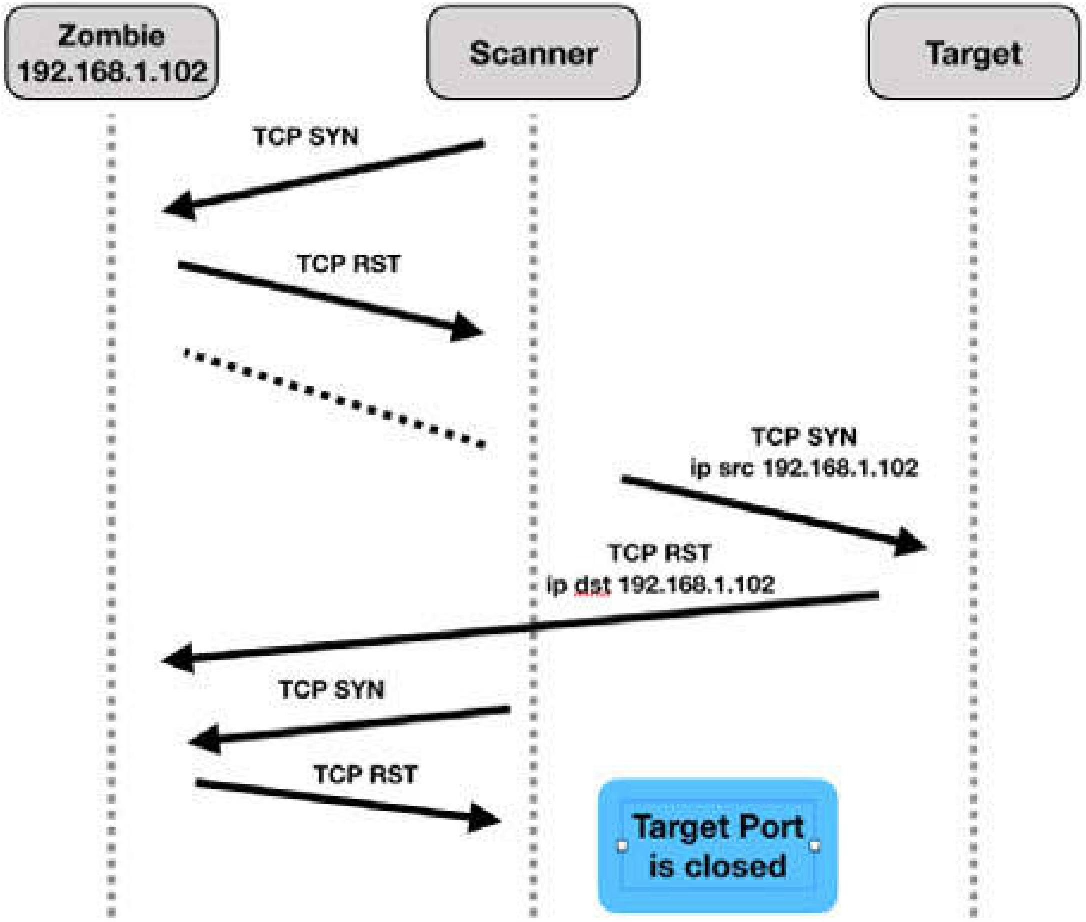


Stop the capture and save the file. As shown in the figure below, the zombie IP didn't respond to any of the probes sent by the scan IP. It is quite clear in the figure below where the display filter ip.addr == 192.168.1.102 is applied. This is also indicated in the output of the command above.

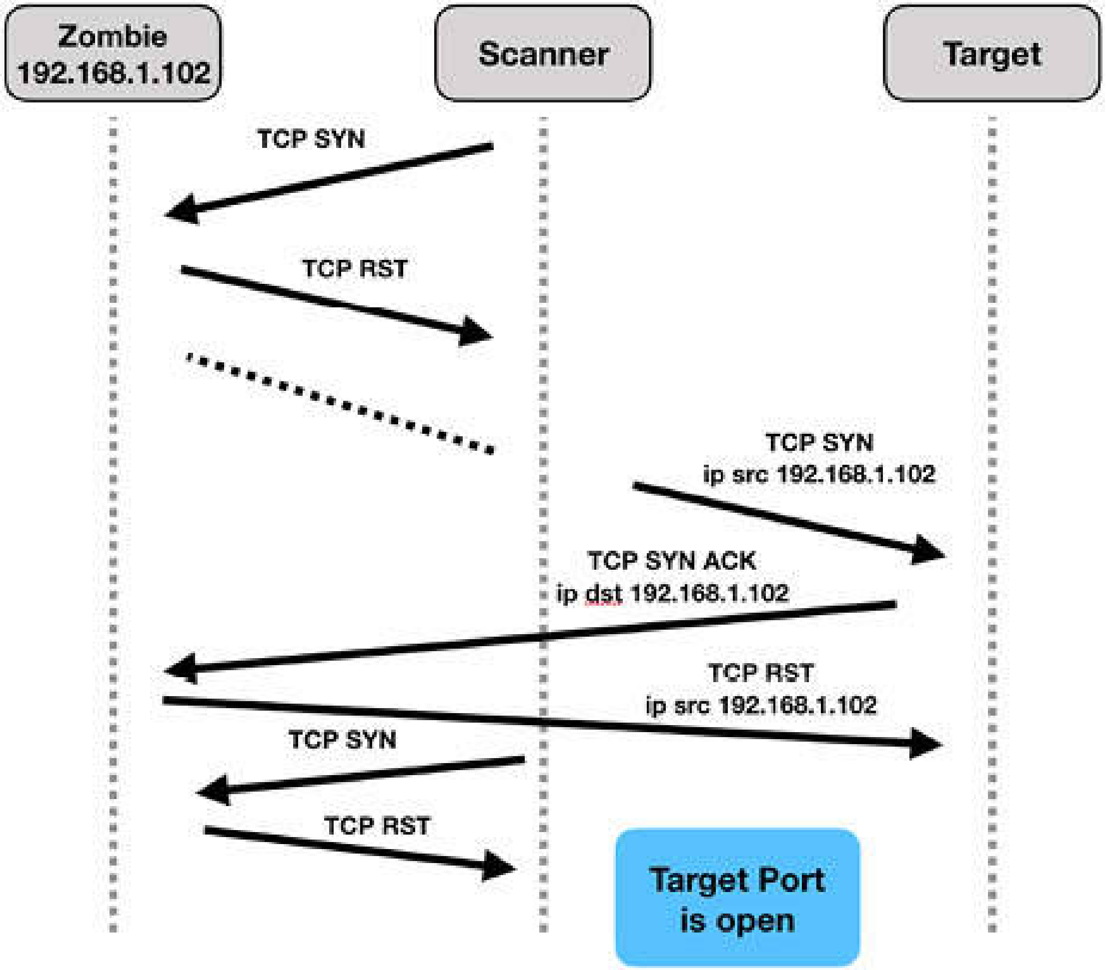


If the zombie responds, there can be two cases:

- Case#1: The target port was not open, and the target sent a TCP RST to the zombie, as shown in the figure below.



- Case#2: The target port was open, and the target sent a TCP ACK to the zombie.



Task 3:
ICMP packets are very important in the case of Destination Unreachable scenarios. ICMP may also be sent for many other reasons, such as route redirection (ICMP Type 5).

If ICMP Type 3 packets are sent, the following codes are available:

- Code 0: Net unreachable
- Code 1: Host unreachable
- Code 2: Protocol unreachable
- Code 3: Port unreachable
- Code 4: Fragmentation Needed and Don't Fragment was set
- Code 5: Source route failed

- Code 6: Destination network unknown
- Code 7: Destination host unknown
- Code 8: Source host isolated
- Code 9: Communication with destination network is administratively prohibited
- Code 10: Communication with destination host is administratively prohibited
- Code 11: Destination network unreachable for Type of Service
- Code 12: Destination host unreachable for Type of Service
- Code 13: Communication administratively prohibited
- Code 14: Host precedence violation
- Code 15: Precedence cutoff in effect

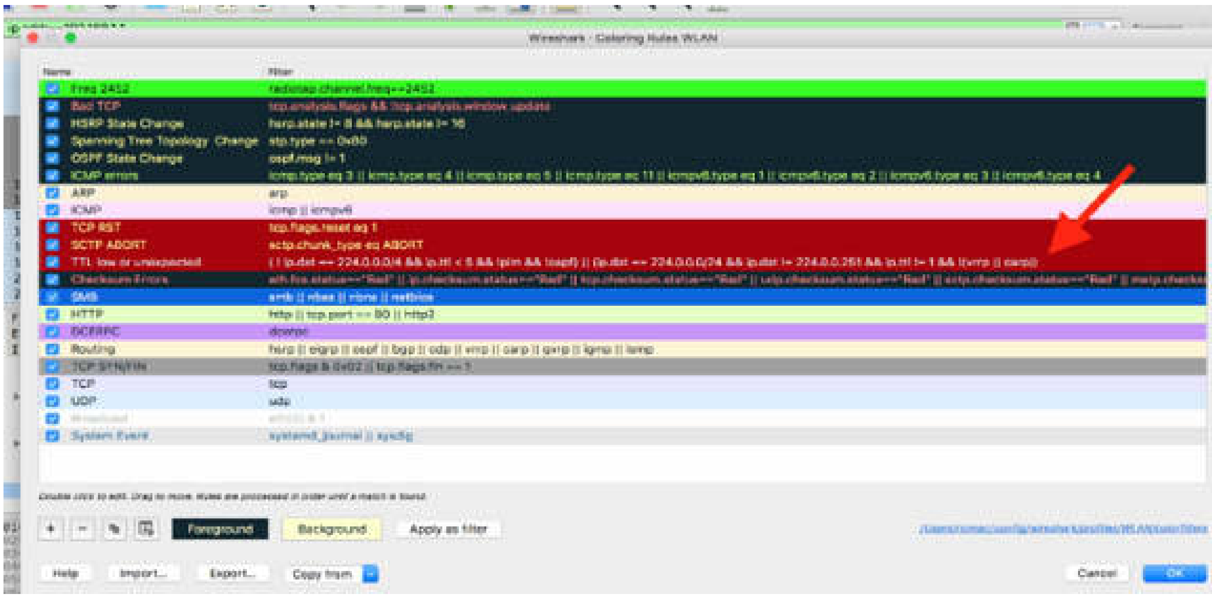
Although many of the above ICMP packets may be blocked or hosts may be configured to not generate them, ICMP Type 3/Code 4 should never be blocked. In fact, an ICMP packet alerts a host that its packet was too large to traverse a link and the “Don’t Fragment” bit in the IP header was set to 1. On receiving this ICMP Type 3/Code 4 packet, the transmitting host should automatically split the original TCP segment data into smaller packets and resend the data.

Task 4:
ICMP is also commonly used as a sort of path discovery mechanism using an ICMP Echo Request (Type 8) packet and an Echo Reply (Type 0) packet (a well-known mechanism for ping request/reply). Windows hosts use ICMP as the default traceroute method. Unix hosts use UDP for traceroute path discovery.

Consider a situation where a source system increments the IP header TTL field value in consecutive ping packets to discover the route to a target. Each router along the path then decrements the TTL value by 1, at every step. When the packet arrives at a router and its TTL value has been decremented to 1, the receiving router discards the packet (because it is not possible to subtract 1 from TTL value). The router generates an ICMP Time

Exceeded in Transit (Type 11) packet to the originator of the discarded packet to notify it.

In general, packets with a TTL value lower than 5 are considered suspicious. Wireshark includes a default coloring rule for packets that contain a low TTL value. The syntax of the coloring rule is ((! ip.dst == 224.0.0.0/4 && ip.ttl < 5 && !pim && !ospf) || (ip.dst == 224.0.0.0/24 && ip.dst != 224.0.0.251 && ip.ttl != 1 && !(vrrp || carp))) , as shown in the Coloring Rules dialog box below.



This coloring rule examines the destination IP address field to look for multicasts. Traffic is colored with a red background and a white foreground if it is not multicast but has an IP TTL value lower than 5 or if it is multicast and the IP TTL value is not equal to 1.

Two other commonly used variations of traceroute include UDP traceroute and TCP traceroute. UDP traceroute sends UDP packets to a closed UDP port. The “Time-to-Live Exceeded in Transit” responses from routers along the path are used to discover the path to the target. The expected response is an ICMP Type 3/Code 3—Destination Unreachable/Port Unreachable.

TCP traceroute sends TCP packets to any TCP port. The “Time-to-Live Exceeded in Transit” responses from routers along the path are used to discover the path to the target. The expected response is a TCP Reset or TCP SYN/ACK.

In an IPv4 environment, detecting ICMP-based, UDP-based, or TCP-based traceroute can be simple if the routers along the path respond with the "Time-to-Live Exceeded in Transit" ICMP packets. Consider creating a coloring rule or a display filter (icmp.type==11) && (icmp.code==0) . This coloring rule must be placed above the default ICMP Errors coloring rule to correctly detect ICMPs.

Notes:
Repeat the previous steps to gain confidence in using idle scan and try to test if a target has any open ports even if you are not able to get direct access to it (use zombie).

To gain more confidence in using ICMP traceroute, correctly detect the responses of the Windows target and the Unix target.

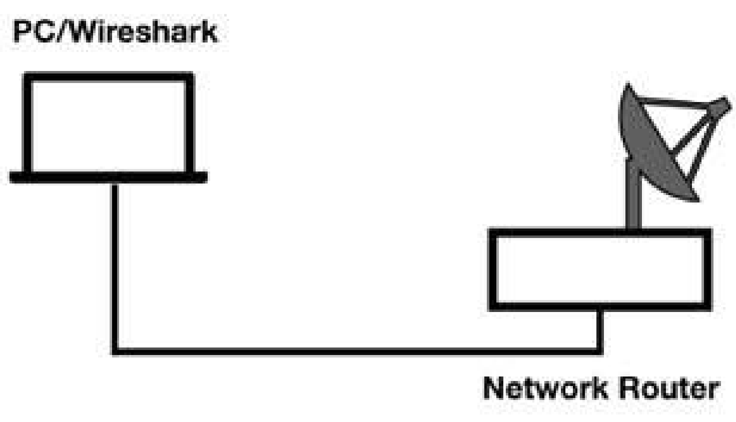
Lab 97. Application Mapping, OS Fingerprinting, and IP Spoofing

Lab Objective:
Learn what application mapping and OS fingerprinting are and what's their use during the scan process.

Lab Purpose:
Understand how to properly use the application mapping process, OS fingerprinting, and the necessary tools.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
Application mapping identifies services on a target even when these services are not using the standard ports. For example, if someone runs an FTP server process on port 80 (the default port for HTTP), an application mapping tool identifies that FTP is running on that port, not HTTP.

Nmap offers excellent application mapping capabilities, and it is used as an example tool in this lab. Amap is another useful tool (**https://tools.kali.org/information-gathering/amap**).

By default, while scanning ports, Nmap references the nmap-services file (an internal configuration file) to correlate a port number with a service.

Application mapping relies on two distinct functions—probing and matching. Probes are messages sent to a target to generate responses. Responses are matched to predefined response patterns to identify the service discovered.

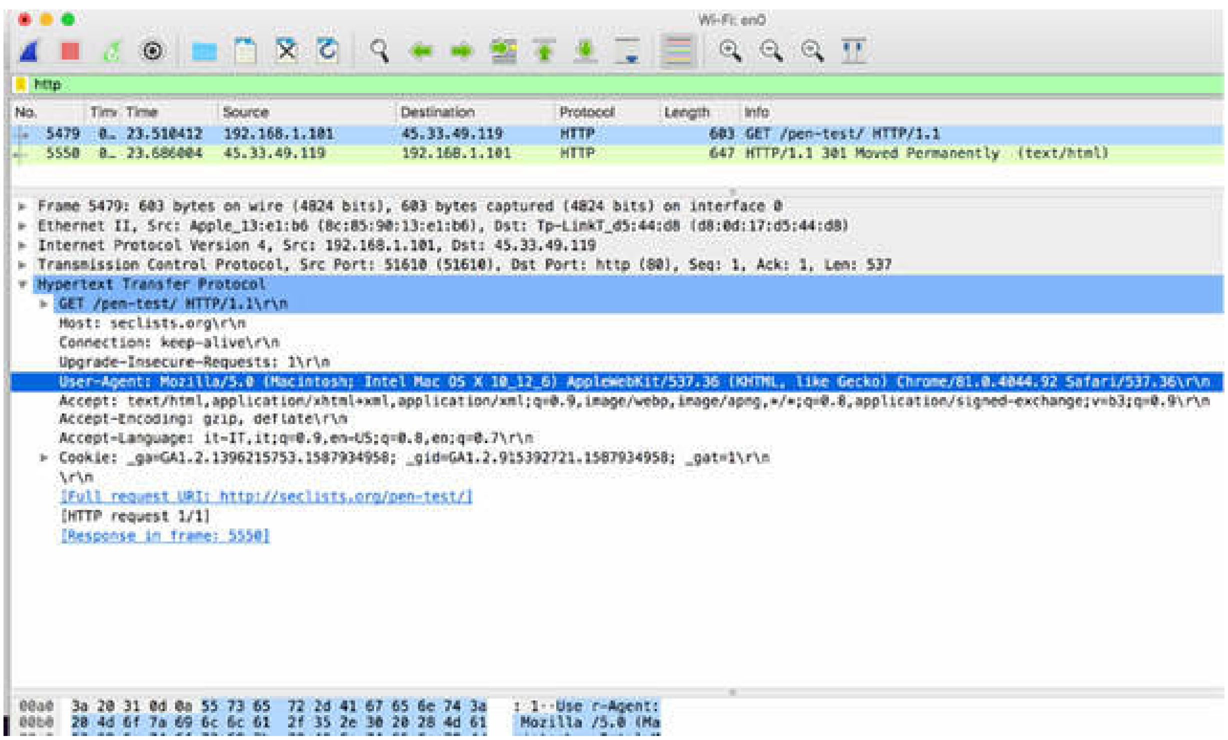
In some cases though, probes are not required to identify a service. For example, after a TCP connection is established with a port, Nmap listens for five seconds. Many applications—such as FTP, POP3, and SMTP—offer a banner immediately after the connection. Nmap compares the response received, if any, to the contents of the nmap-services-probes file. This process of listening is called a NULL probe, but it is not related to a TCP null scan that generates a packet without any TCP flags set. The probing process sends packets out with a protocol definition and a string to trigger a response. The response is compared with the matched lines in the nmap-services-probes file.

Task 2:
OS fingerprinting is the process where you can determine the operating system of a target through either active scanning or passive listening. Wireshark can be used as a passive listening device, and it can identify active OS fingerprinting processes.

Trace files taken by Wireshark can be used to make some assumptions regarding the operating system running on different hosts. For example, if traffic travels to and from ports 135, 137, 139, and 445 on a host, you can make some basic assumptions that the host is a Windows host. You can also assume that the host is not on a Windows version before Windows 2000 because those Windows versions did not support services on port 445 (usually SMB over TCP/IP).

In general, numerous packets on the network also contain evidence of a host’s operating system. For example, HTTP GET requests contain a UserAgent definition.

In Wireshark, capture the traffic for a few minutes on an active network interface while you browse different websites with your web browser. Stop the capture and save the file. In the filter toolbar, enter http . The result will be similar to the one shown in the figure below.



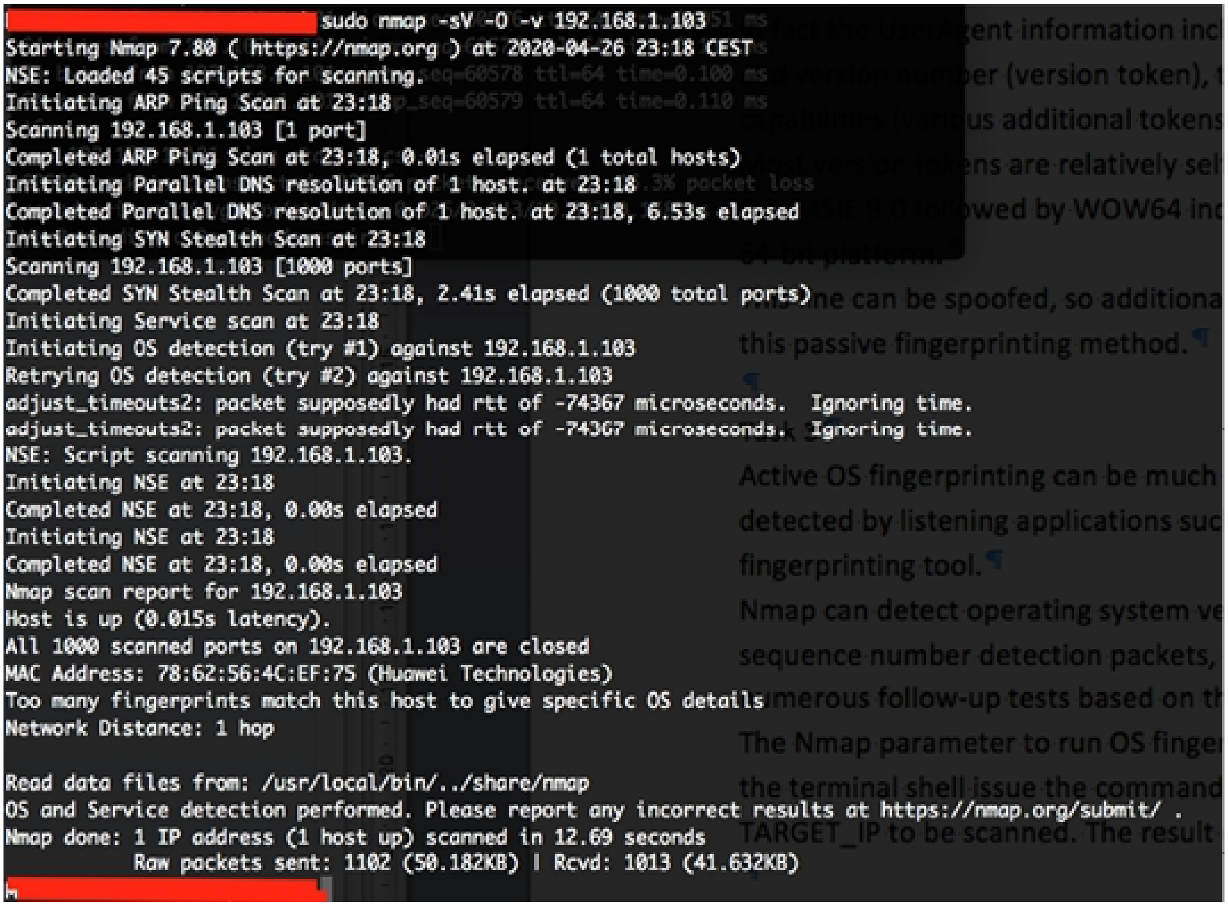
In the figure above, the User-Agent field in the Packet Details pane indicates that probably the browsing client is a Macintosh host using Chrome v.81.0.4044.92. The User-Agent information includes several components such as the browser application name and version number (version token), the operating system information (platform token), and additional capabilities (various additional tokens).

Most version tokens are relatively self-explanatory—for example, MSIE 9.0 is Internet Explorer version 9.0. MSIE 9.0 followed by WOW64 indicates that the 32-bit version of Internet Explorer is running on a 64-bit platform. This line can be spoofed, so additional OS fingerprinting techniques should be used in conjunction with this passive fingerprinting method.

Task 3:
Active OS fingerprinting can be much more efficient than passive OS fingerprinting, and it can also be detected by listening applications such as Wireshark. Nmap is an excellent example of an OS fingerprinting tool.

Nmap can detect operating system version information based on a series of port scans, ICMP pings, sequence number detection packets, TCP Explicit Congestion Notification tests, closed port tests, and numerous follow-up tests based on the responses received.

The Nmap parameters to run OS fingerprinting with verbosity and version detection are –sV –O –v . In Wireshark, capture the traffic for a few minutes on an active network interface. In the terminal window, run the command nmap -sV -O -v 192.168.1.103 , where 192.168.1.103 is the target IP to be scanned. The result of this command is shown in the figure below.



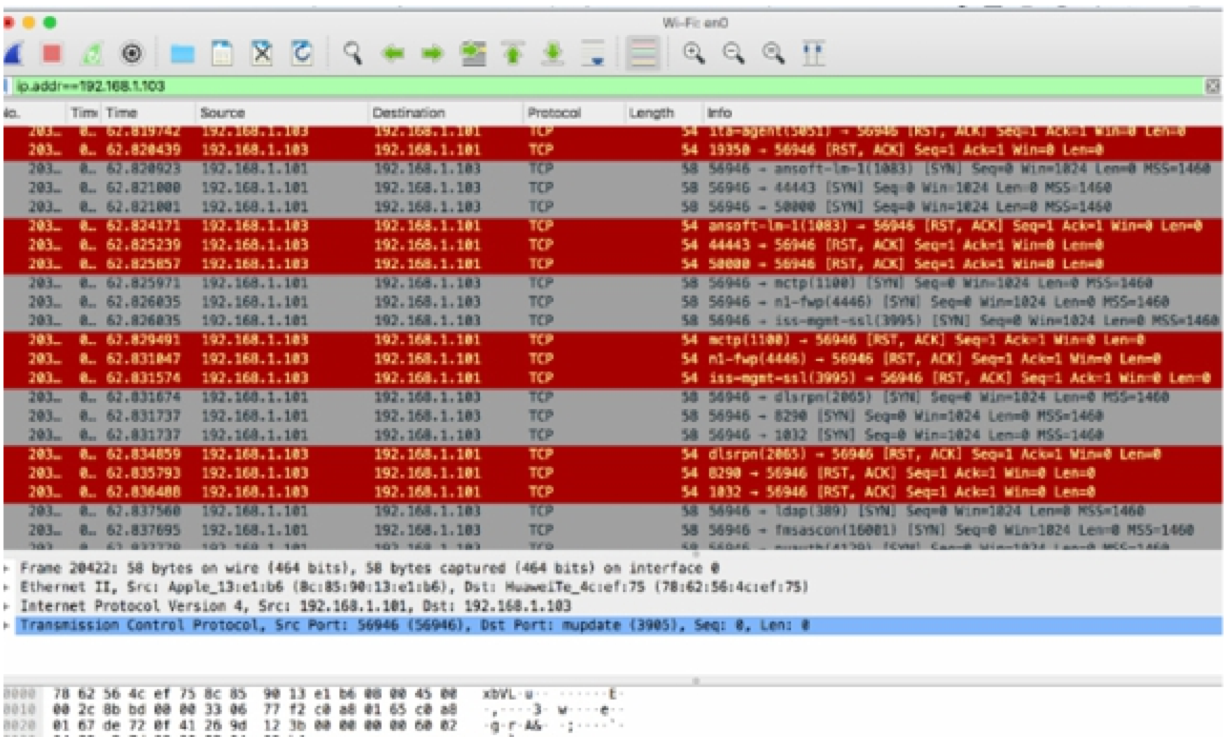
Stop the capture and save the file. You can observe that even if multiple discovery attempts are made, it is impossible to correctly detect the OS.

Examining Nmap's process of OS fingerprinting provides many signatures of its traffic:

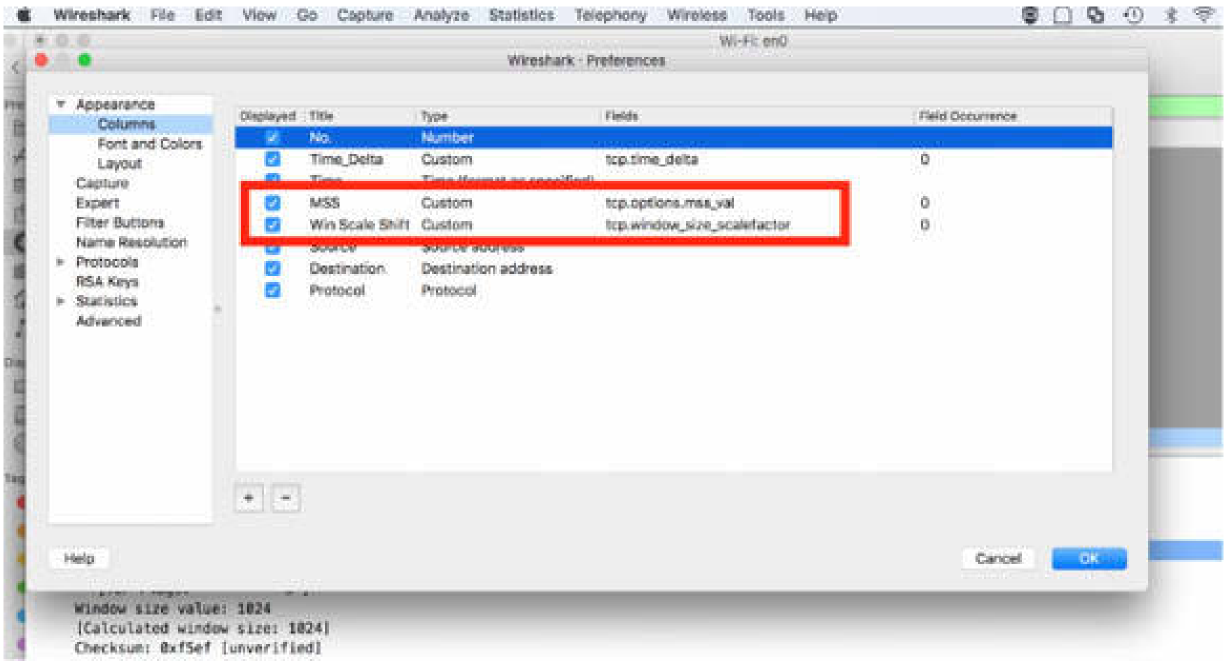
- ICMP Echo Request (Type 8) with no payload
- ICMP Echo Request (Type 8) with 120-byte or 150-byte payload of 0x00s
- ICMP Timestamp Request with Originate Timestamp value set to 0
- TCP SYN with 40-byte options area
- TCP SYN with Window Scale Shift Count set to 10
- TCP SYN with Maximum Segment Size set to 256
- TCP SYN with Timestamp Value set to 0xFFFFFFFF

- TCP packet with options and SYN, FIN, PSH, and URG bits set
- TCP packet with options and no flags set
- TCP Acknowledgment Number field non-zero without the ACK bit set
- TCP packets with unusual TCP Window Size field values

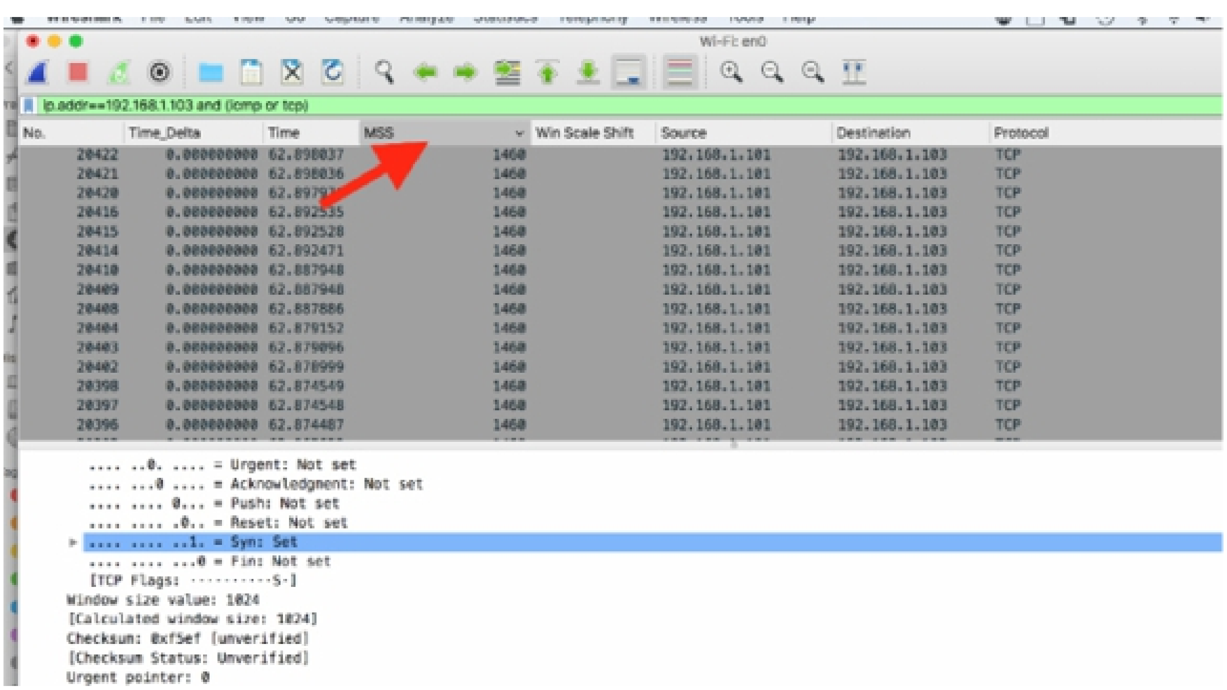
To show all relevant packets in the Packet List pane, in the filter toolbar, enter ip.addr==192.168.1.103 , as shown in the figure below.



To highlight some of the unique packets of the Nmap OS detection process, you can add two columns—a TCP Maximum Segment Size (MSS) column and a Window Scale Shift column by using the Preferences dialog box, as shown in the figure below.



Apply a TCP or ICMP filter, as shown in the figure below, and then sort the results based on the MSS column.



You can also create coloring rules for some of the unique packets to make Nmap's OS detection process much easier. The following list describes some of the coloring rules with distinctive coloring:

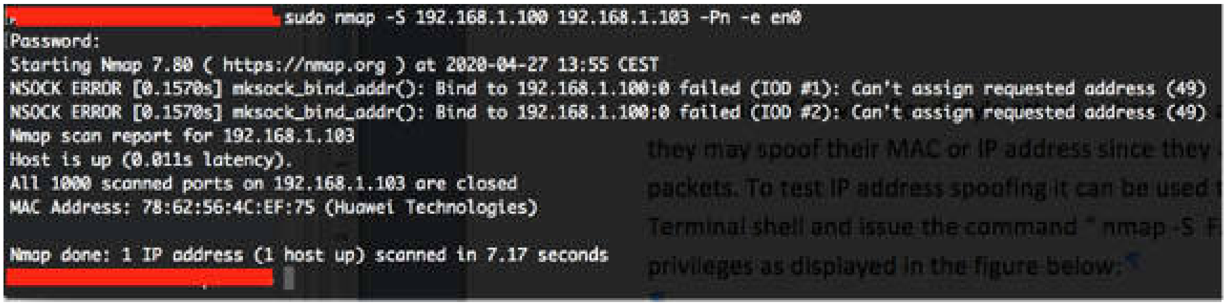
- TCP SYN/ACK with a TCP Window Size field value less than 1025:
  (tcp.flags==0x02) && (tcp.window_size < 1025)
- TCP SYN, FIN, PSH, and URG bits set:
  tcp.flags==0x2b
- No TCP flags set:
  tcp.flags==0x00
- ICMP Timestamp Request with Originate Timestamp Value set to 0 (Ethernet II header structure):
  (icmp.type==13) && (frame[42:4]==00:00:00:00)
- TCP Window Scale Option set to 10
  tcp.options.wscale_val==10
- TCP Maximum Segment Size value set to less than 1460
  tcp.options.mss_val < 1460

Task 4:
Usually, attackers and scanners may use MAC or IP address spoofing to hide their actual hardware or network addresses or to appear to be another system to get through filtering devices on the network.

In a denial-of-service flood style attack where the attackers do not rely on two-way communications, the attackers may spoof their MAC or IP address because they are not reliant upon receiving responses to their packets. To test IP address spoofing, you can use the -S option with the nmap command.

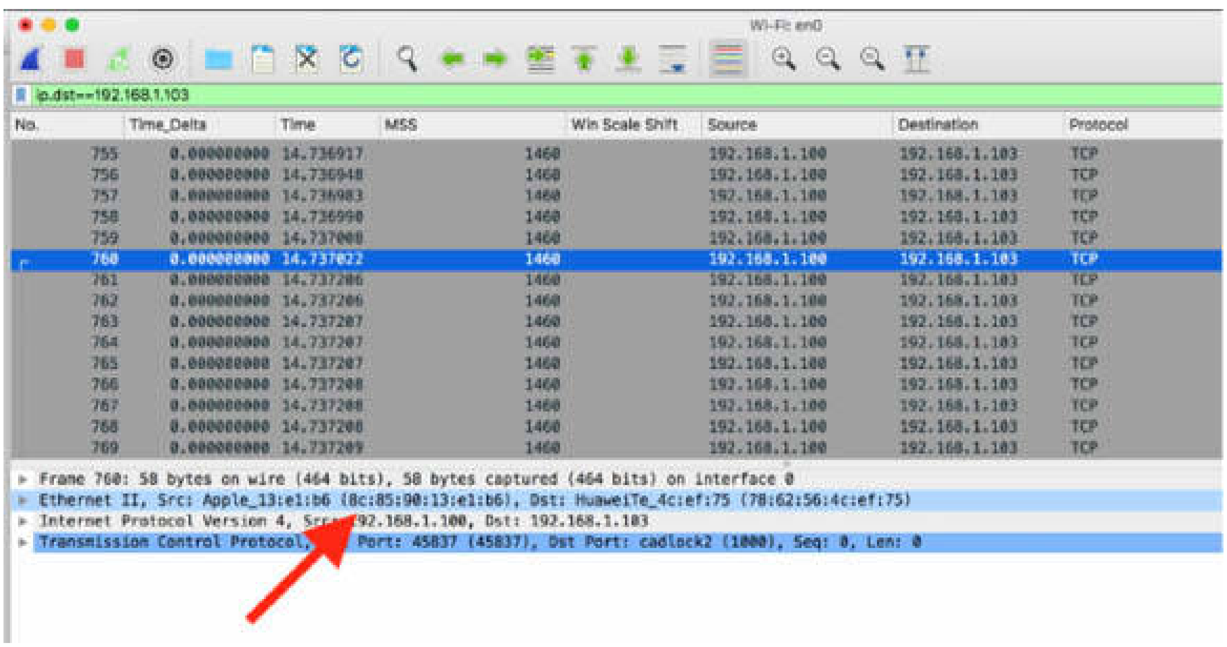
In Wireshark, capture the traffic for a few minutes on an active network interface. Open a terminal window, and run the nmap -S FAKE_IP TARGET_IP -Pn

-e ACTIVE_ETH_ITF command with root privileges, as shown in the figure below.

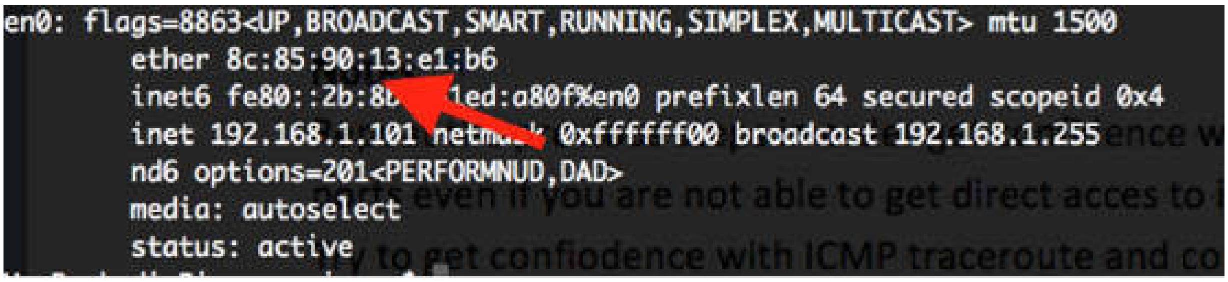


The real IP address of the scanning machine is 192.168.1.101 but you will see the packets transmitted on the network with the source IP address 192.168.1.100.

Stop the capture and save the file. In the filter toolbar, enter ip.dst==192.168.1.103 . The result will be similar to the one shown in the figure below.



From the figure above, you can confirm that the only packets that are transmitted to IP address 192.168.1.103 (TARGET_ADDR) are coming from IP address 192.168.1.100, which is not the IP address of the local machine. In addition, you can confirm that the SRC MAC Address present in the packets is our MAC address. In the terminal window, type ifconfig , as shown in the figure below, even if obviously the real IP address is different.



Notes:

Repeat the previous steps to gain confidence in using the application mapping procedure and applying it to different hosts. Try to detect the OS fingerprint by using Wireshark, capturing HTTP traffic in the network.

Identify a target host and try IP spoofing by using the Nmap tool.

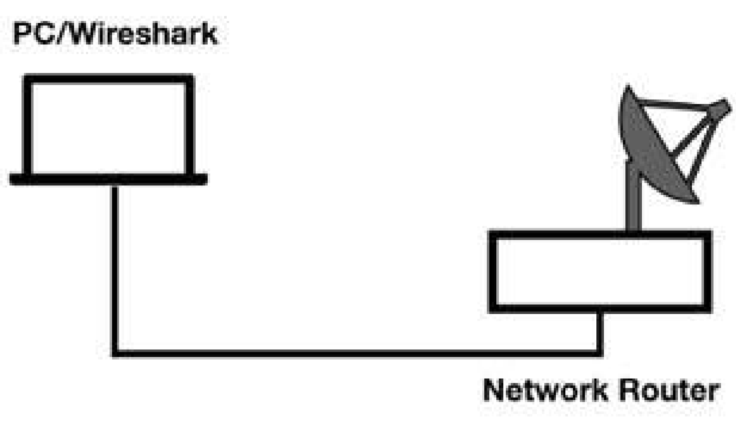
Lab 98. Vulnerabilities, Malformed Packets, and Dark Addresses

Lab Objective:
Learn how to detect vulnerabilities in the resolution process.

Lab Purpose:
Understand how to properly identify vulnerabilities in the TCP/IP communication.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
Suspect traffic is some sort of traffic that does not match network baselines. It is either out of place because the protocol type is not right, the used port is not correct, packet frequency is strange, etc. Sometimes normal network communications that we are not familiar with or traffic that has unusual patterns can be considered suspect traffic.

Suspect traffic may simply be caused by poorly-behaving applications, misconfigurations, innocent mistakes, or faulty devices. To rule out the causes of suspect traffic, you must first understand what is normal. This is where the baselines become a precious resource.

Understanding normal TCP/IP communications is important for identifying abnormal communications. The standard flow for TCP/IP communications is based on the following steps:

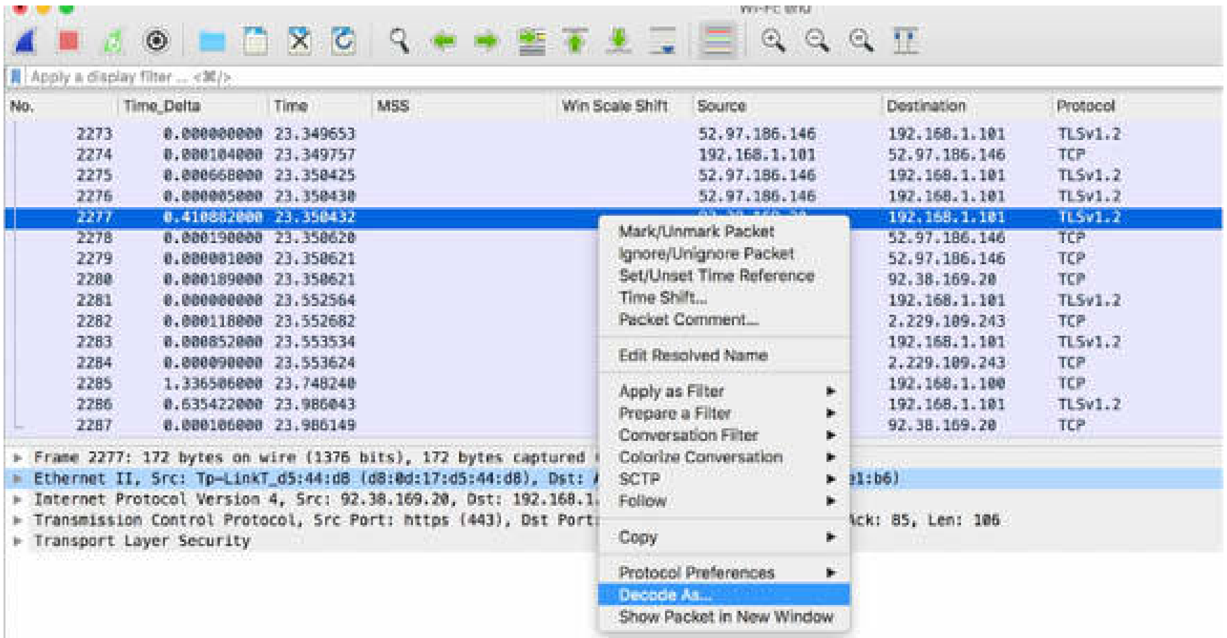
Port Resolution → Name Resolution → Mac Address Resolution (or, if the target is remote, → Route Resolution → Mac Address Resolution)

For each step, there is at least one security issue to be considered.

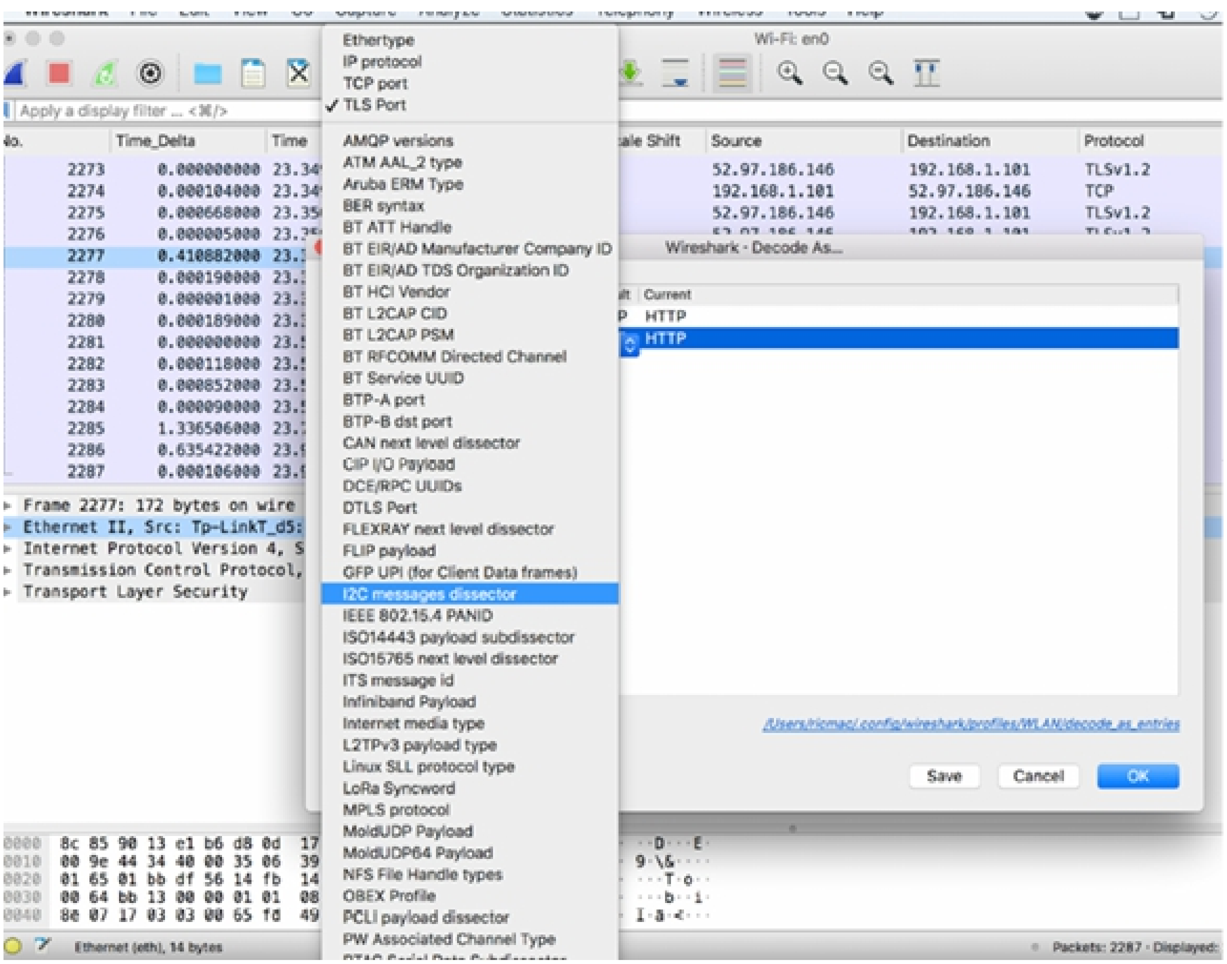
If you consider the Port Resolution vulnerabilities, you must know that port resolution relies on the integrity of the services file and the application requesting to use a particular port number. If a malicious user or program has altered the content of the services file, the port resolution process may be affected. Applications can also define the ports they use. A malicious FTP program might use port 80 knowing that many companies do not block outbound traffic to this port.

Bot-infected hosts could use non-standard ports to communicate through standard protocols. For example, it can be use Internet Relay Chat (IRC) to communicate with Command and Control (C&C) servers. In this case, the bot-infected host connects to the IRC server on a non-standard port and Wireshark defines the IRC communications as simply “Data”.

To handle such issues in Wireshark, in the Packet List pane, select a packet, right-click it, and then select Decode As. This forces Wireshark to temporarily dissect traffic to and from a non-standard port as different protocol traffic, as shown in the figure below.

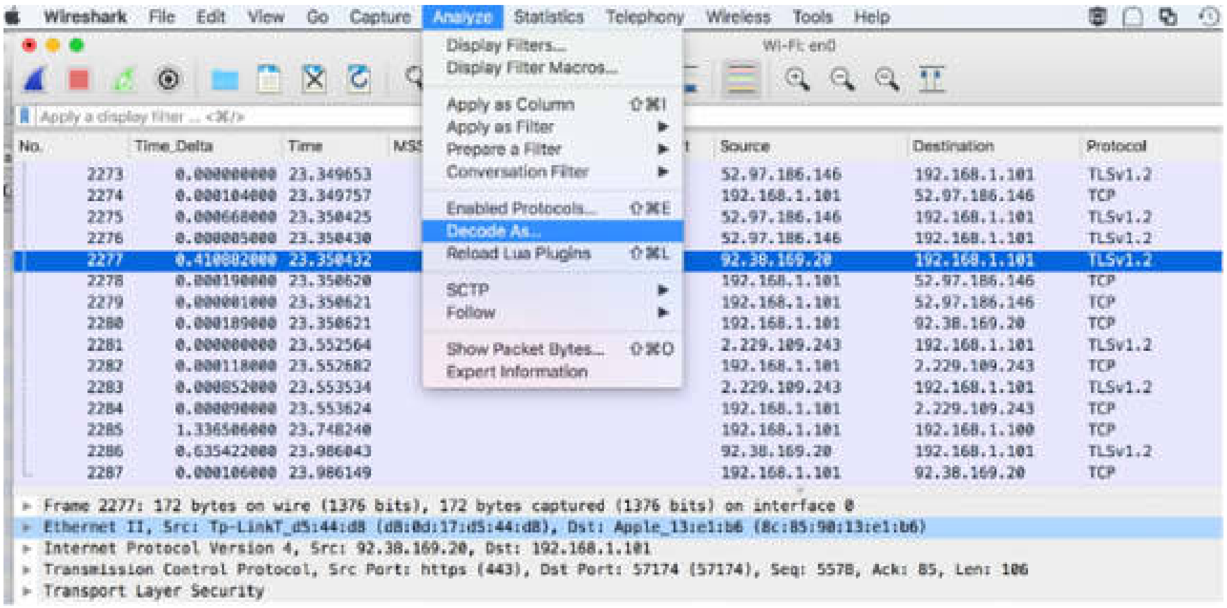


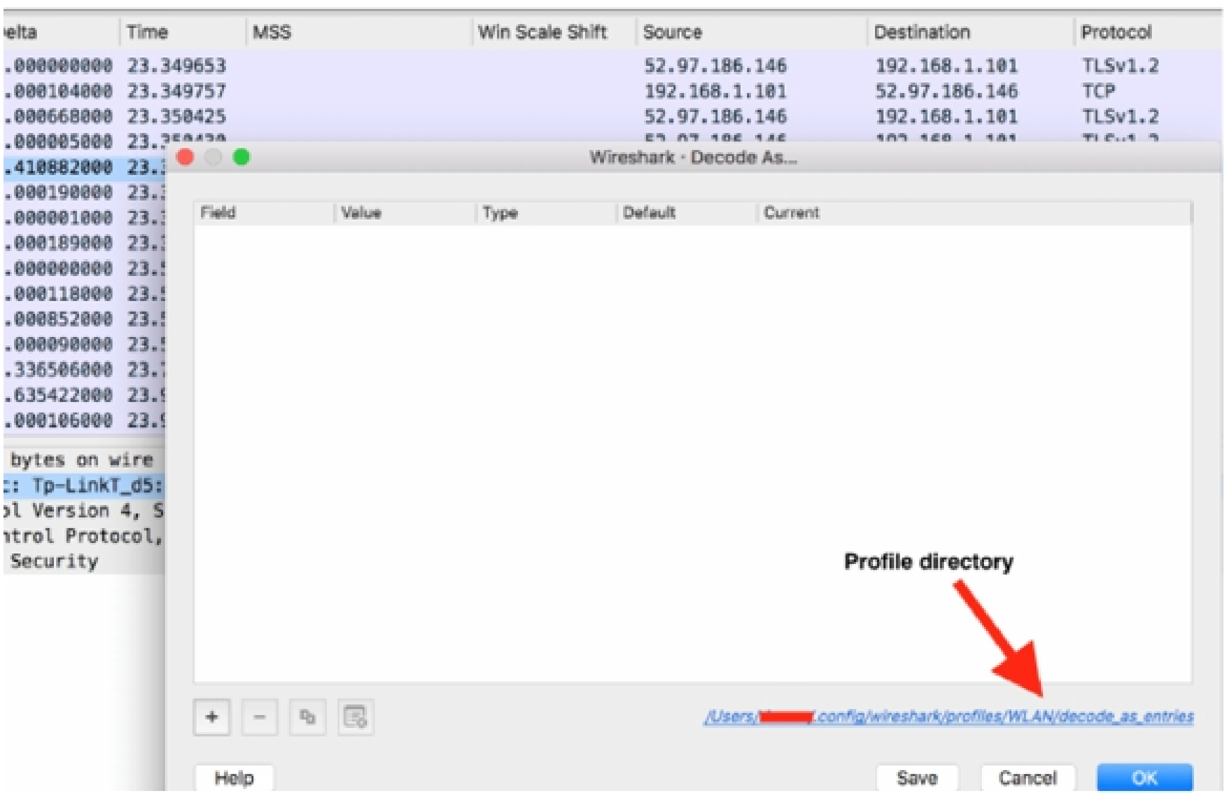
From the list of provided protocols, select the protocol that you are interested in.



When you restart Wireshark or change to another profile, the dissector will not be in place.

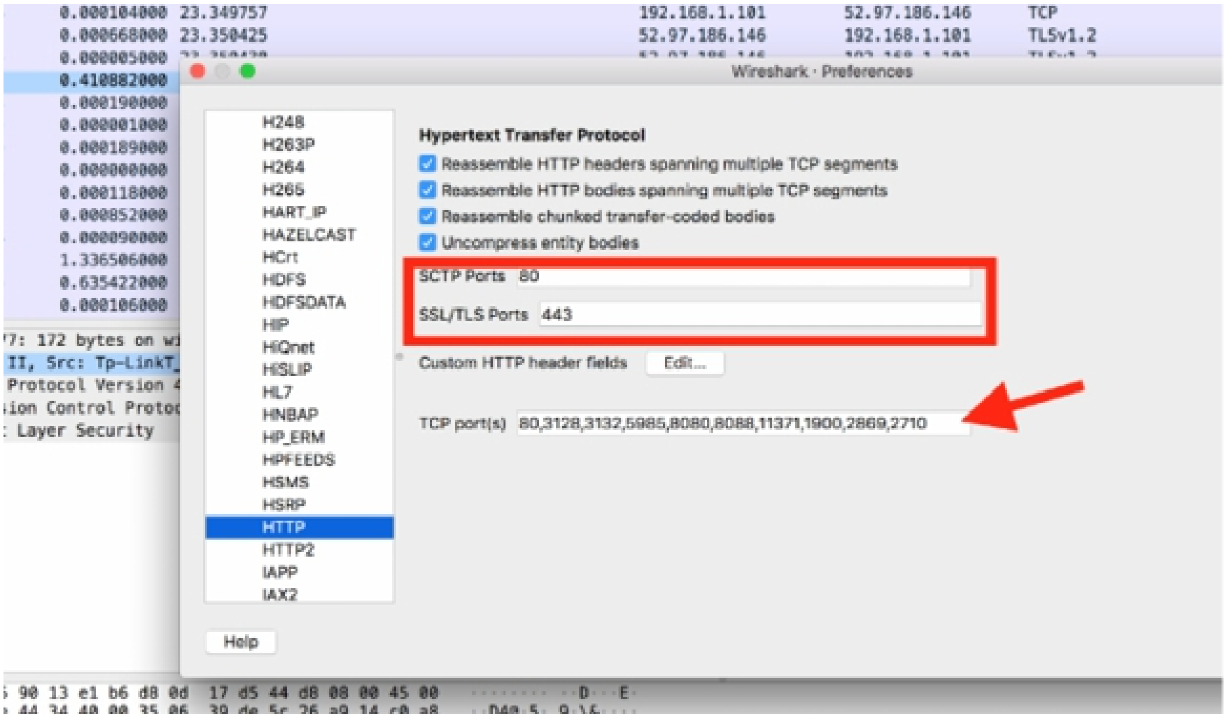
You can also save the “Decode As” settings in a profile. After you have applied a temporary decode, on the main menu, select Analyze > Decode as. Click Save. Wireshark retains your new decode setting in a decode_as_entries file in your profile directory.





You can also define preferences for some applications, such as HTTP, and configure Wireshark to recognize additional or alternate port numbers for applications. On the main menu, click Edit > Preferences. In the

Preferences dialog box, select a protocol in the left tree view (for example, HTTP) and then in the port settings, set TCP ports to be decoded as HTTP traffic. Also, set the port that will be dissected as SSL/TLS traffic.

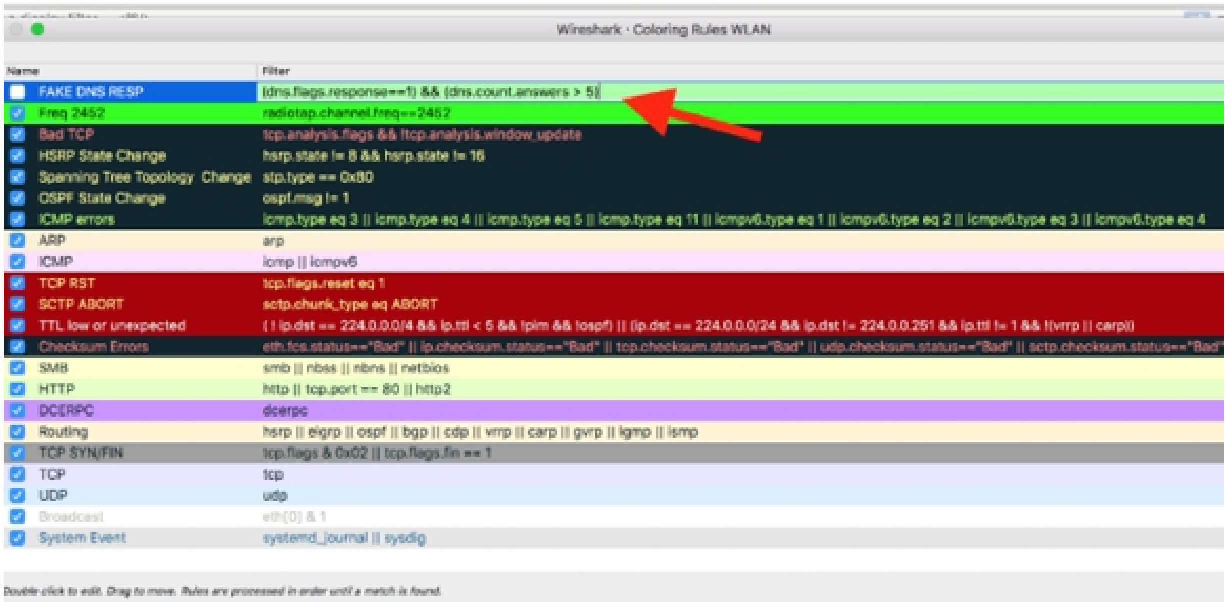


Task 2:

Another possibility is the Name Resolution process vulnerability. If a malicious application has altered the client's hosts file, the client's system will use the information in that file before generating a DNS query. Unless a secure form of DNS is used to validate responses and the responding DNS server, clients accept any DNS responses as long as the transaction ID number and the restated query match the original request.

If the DNS information supplied is not correct or leads to an alternate host, the client continues the resolution processes to connect to the incorrect host. If this information is kept in the DNS cache, the client uses it again (until the information has expired). In fact, unless you know the IP address that corresponds to a hostname, it is difficult to spot traffic with malicious intent.

In the case of bot-infected hosts, however, it is not uncommon to see a DNS query that generates canonical name responses with many IP addresses. To spot this situation, you can create a coloring rule to identify DNS responses that contain more than 5 IP addresses. Create a coloring rule with the syntax (dns.flags.response==1) && (dns.count.answers > 5) , as shown in the figure below.



Another vulnerability is related to the MAC Address Resolution process. When resolving the hardware address of a local target or a router, the client depends on the validity of the ARP response or entries that exist in the local ARP cache to use the proper MAC address in the subsequent packets.

MAC address redirection can be used by some attackers to perpetrate a man-in-the-middle attack. ARP poisoning is an example of unusual ARP traffic.

The last example of vulnerability is related to the Route Resolution. When a client needs to send data to a target on a remote network, the client checks its routing tables to identify the best gateway or a default gateway if present. If the local route table has been poisoned, the client sends the packets in the wrong direction, and it can’t reach the desired target. This route redirection can be used for the man-in-the-middle attacks.

Task 3:
Wireshark usually can reveal unusual patterns of network scans, attempted logins, insecure communications, strange protocols, or unusual application behavior. You can make unusual traffic easier to identify by colorizing the traffic that is of concern. The syntax used by display filters and coloring rules should be chosen appropriately to make this traffic more visible in Wireshark.

Scanning traffic is typically considered unacceptable on the network, but in some cases, it is possible to find out that the scans are generated by network monitoring devices that build and maintain a database of network devices.

Some examples of unacceptable traffic on the network are:

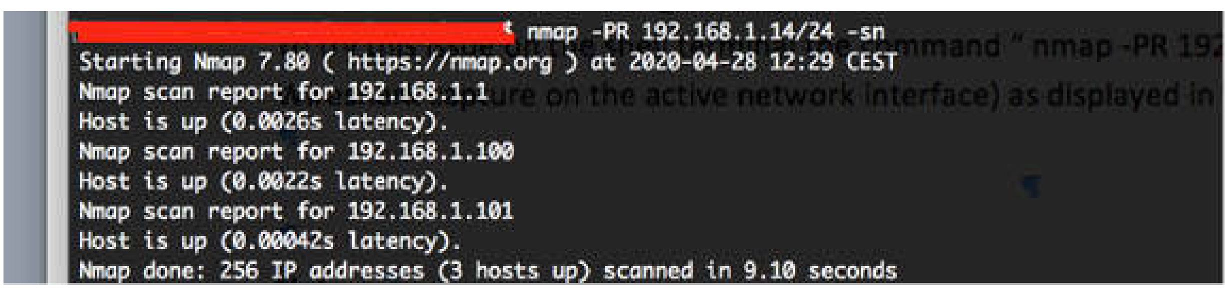
- Maliciously malformed packets—intentionally malicious packets
- Traffic to invalid or 'dark' addresses—packets addressed to unassigned IP or MAC addresses
- Flooding or denial-of-service traffic—traffic sent at a high packet per second rate to a single, group or all hosts
- Clear text passwords—passwords that are visible and therefore, unsecure
- Clear text data—data that is visible or able to be reconstructed
- Phone home traffic—traffic patterns indicating an application is checking in periodically with a remote host
- Unusual protocols and applications—protocols and applications that are not commonly seen or allowed on the network
- Route redirections—ICMP-based route redirections in preparation for man-in-the-middle attacks
- ARP poisoning—altering target ARP tables for redirection of local traffic through another host—used for man-in-the-middle attacks
- IP fragmentation and overwriting—using the IP fragment offset field setting to overwrite previous data sent to a target
- TCP splicing—obscuring the actual TCP data to be processed at the peer

- Password cracking attempts—repeated attempts to guess an account password over a single connection or multiple connections

Task 4:
Given the numerous resolution processes for host and hardware addresses, it is considered unusual to see traffic destined to addresses that are not assigned. For example, consider your network configured as 192.168.1.0/16 and you have assigned the addresses 192.168.1.1 through 192.168.1.20, you would not expect to see traffic destined to 192.168.1.99.

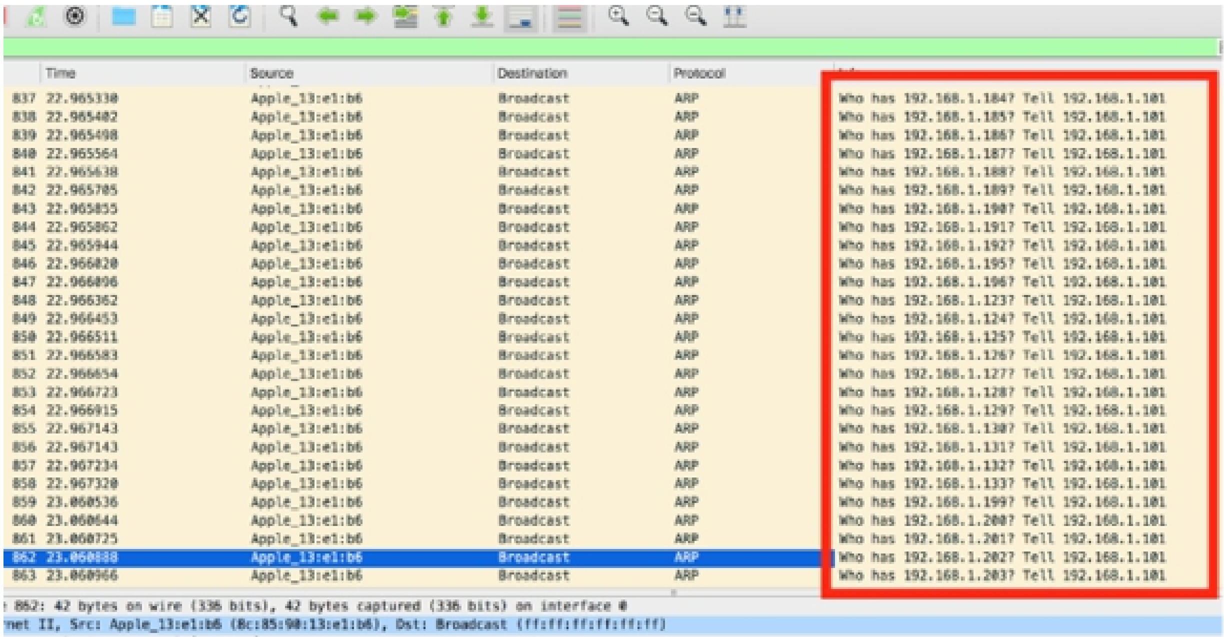
Unassigned MAC addresses are also called "dark MAC addresses and unassigned IP addresses are also called "dark IP addresses." Traffic sent to or referencing unassigned addresses may be indications of blind discovery processes, i.e., someone is trying to find hosts on the network by doing a scan of those host addresses and listening for responses.

To try this issue, in Wireshark, capture the traffic for a few minutes on an active network interface. Open a terminal window, and run the command nmap –PR 192.168.1.14/24 –sn , as shown in the figure below.



Stop the capture and save the file.

In the figure below, a lot of subsequent ARP requests in the Packet List pane indicate an issue that must be investigated.



Traffic sent to unusual target addresses is also an indication of a possible configuration or application problem. For example, traffic sent to 127.0.0.1 (the loopback address) would be considered quite unusual. You can locate traffic to or from addresses that are not in use, but the display filter may be quite long if you use non-contiguous addressing.

The following can be an example of non-contiguous address if your network is configured to use these IP address ranges:

- 192.168.1.1–4 is assigned to routers
- 192.168.1.100–112 is assigned to servers
- 192.168.1.140–211 is assigned to clients

To display the packets of your interest in an efficient way, create a display filter like the following one:

```
ip.dst > 192.168.0.4 && ip.dst < 192.168.0.100) ||
   (ip.dst > 192.168.0.112 && ip.dst < 192.168.0.140) ||
   (ip.dst > 192.168.0.211 && ip.dst <= 192.168.0.255)
```

The parentheses group together the addresses that you want to display.

Notes:

To gain confidence in discovering vulnerabilities and malformed packets, repeat the previous steps capturing on your local network and simulating some attacks with the Nmap tool. Try to scan the network for dark addresses. Get the necessary confidence in using the display filters and coloring rules to make the packets you are interested in more visible.

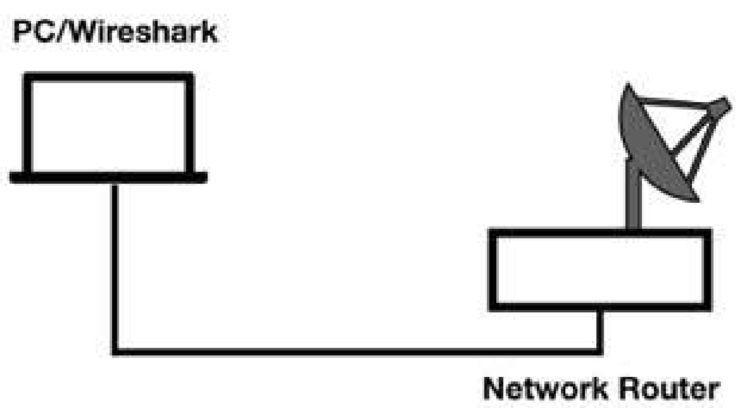
Lab 99. Flood, Clear Text Password, and Unusual Applications

Lab Objective:
Learn how to detect flood during network capture and what are the risk considerations when passwords are sent in clear text.

Lab Purpose:
Understand how to properly identify flood and discriminate it from standard denial of service and the risks associated with using passwords in clear text.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
Floods are a form of a denial-of-service attack. Consistent connection requests are another form. Denial-of-service attacks are created to make a resource unavailable to others. The attack may be focused on a target host, group of hosts, or even the network infrastructure itself.

There are a variety of flooding attacks that can be used to saturate a network link, a TCP connection table, the buffer on a network interface card, switch tables, routing tables, or other elements of a network.

When analyzing network floods, the first thing to be verified is that a configuration mistake could be the cause of the flood. In the majority of cases, the flood is because of a loop in the network, In such a case, the IP ID field of all flooding packets would likely be the same because it is the same packet that is circulating in the network. This type of flood is typically caused by a layer 2 loop, for example, when someone connects a hub into two switches. Spanning Tree is a protocol designed to resolve layer 2 loops.

If the IP ID field value (or other packet value) is different in each packet, then it is not a loop situation. In this case, each packet is generated separately through the IP stack element.

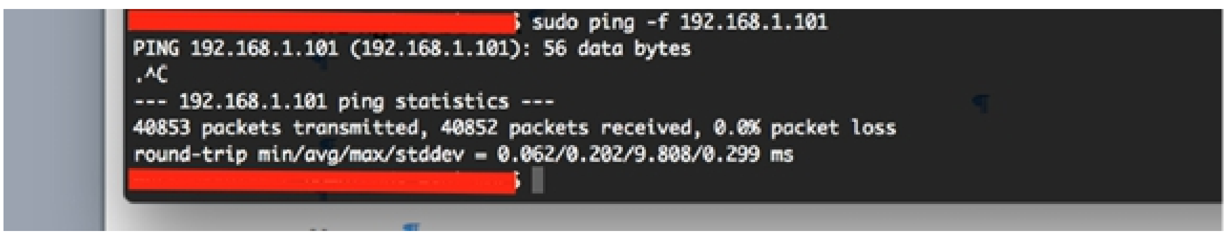
There is a tool that purposefully floods a network, and it is called Macof (**https://kalilinuxtutorials.com/macof/**). The purpose of Macof is to overload a switch's MAC address table to cause the switch to stop making forwarding decisions and forward all packets to all ports (in this case, it is the functionality of a hub) or stop forwarding packets altogether (thereby, becoming a brick).

Wireshark may not be able to keep up with the traffic on a flooded network. If the packet per second rate is high enough, you may find that Wireshark drops packets because of performance limitations on the network interface. Dropped packets may be indicated in the Status Bar depending upon whether the operating system enables the driver to determine if packets are lost.

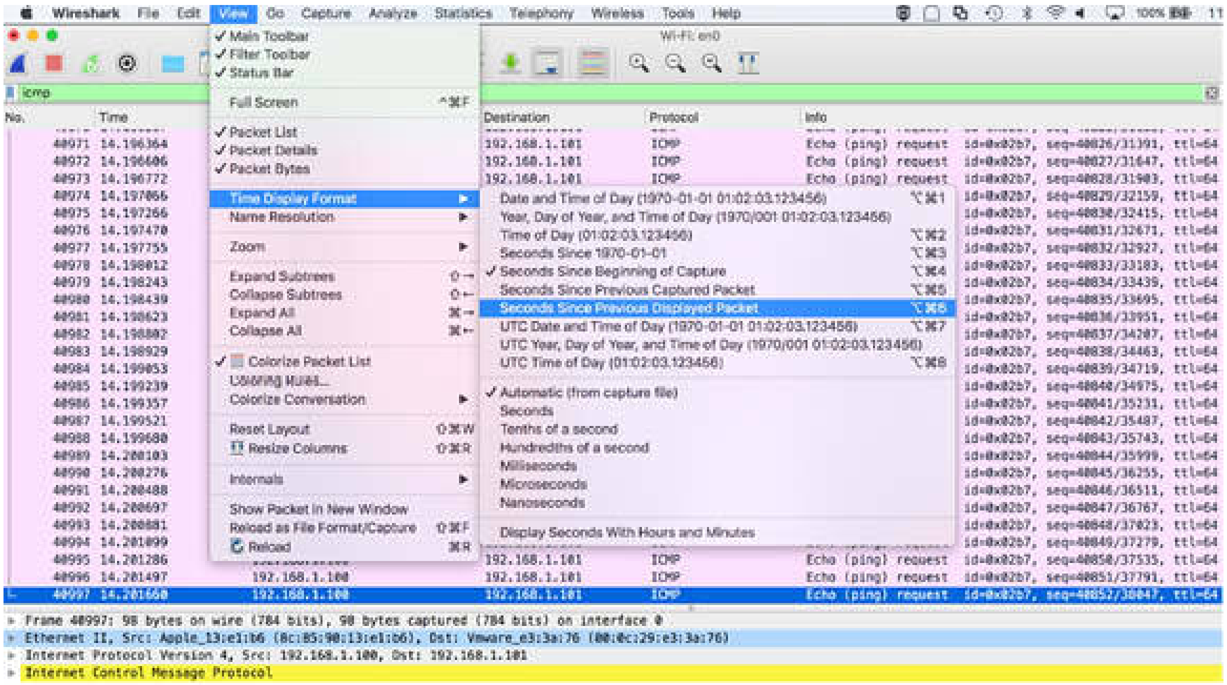
Several optimization techniques can be used when capturing on a flooded network. The first, and most efficient method, is to use TShark (the command-line version without GUI) or Dumpcap instead of Wireshark to capture the traffic. In fact, on a flooded network, you may not need to capture many packets to identify the characteristics of the flood.

If you choose to use Wireshark to capture the traffic, an option could be to turn off unnecessary features, such as updating the list of packets in real time, colorization, and network name resolution.

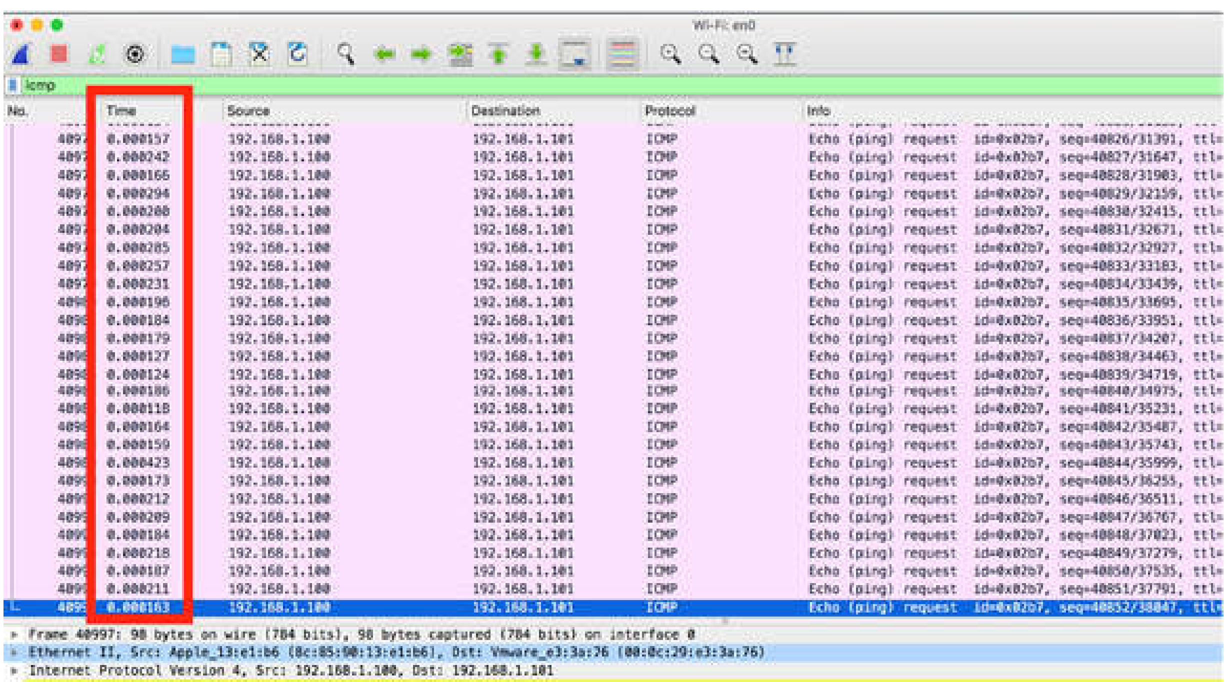
You can see some effect similar to a flood by using the flood version of the standard ping command. In Wireshark, capture the traffic for a few minutes on an active network interface. Open a terminal window, and run the command sudo ping -f TARGET-IP with administrative privileges, where TARGET-IP is a known IP belonging to a known host, as shown in the figure below.



Stop the capture and save the file. In the filter toolbar, enter icmp to display only the ping packets. On the main menu, select View > Time Display Format > Seconds Since Previous Displayed Packet, as shown in the figure below.



You can observe that the majority of the ping/flood packets are sent after 200 microseconds (millionths of a second) gap, as shown in the figure below.



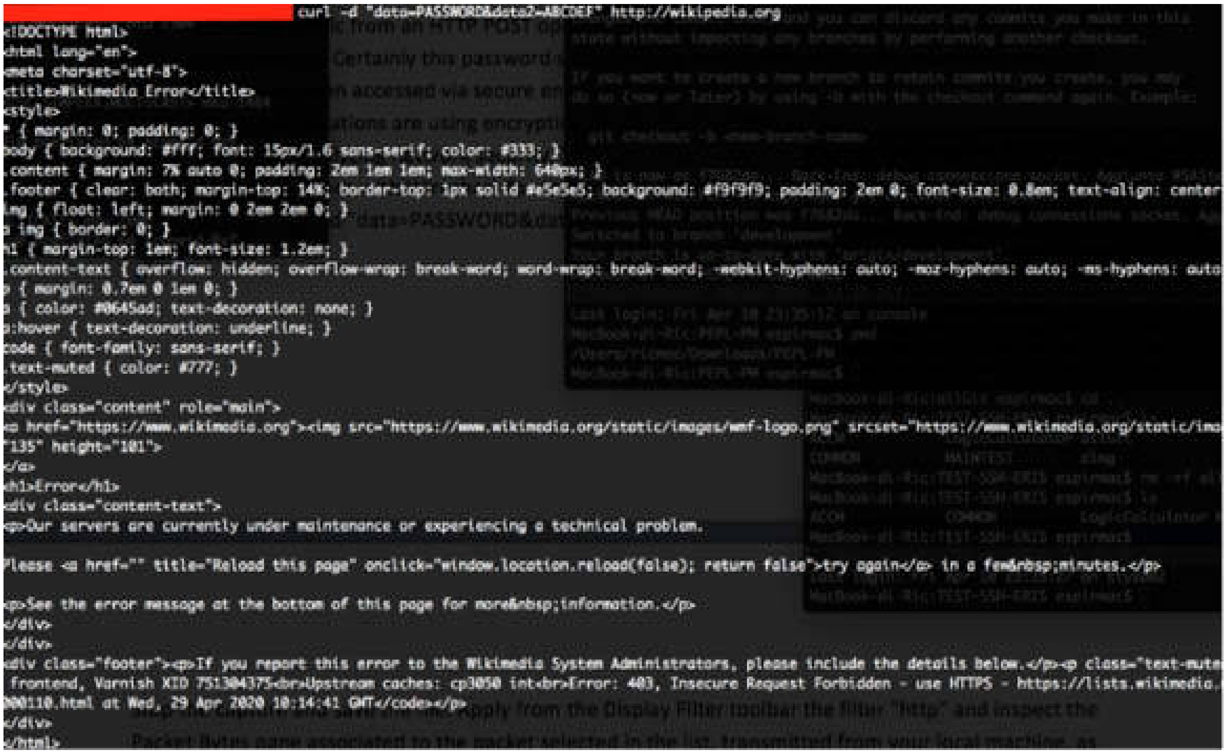
Such time values in the Time column are a clear indication of a flooding attack. If the flooding attacker is using Macof, it is important to know that it sends SYN packets to random target addresses. Macof has a signature in the TCP header of each SYN packet, so you can build the following coloring rule: tcp.window_size==512 && tcp.flags.syn==1 .

Task 2:

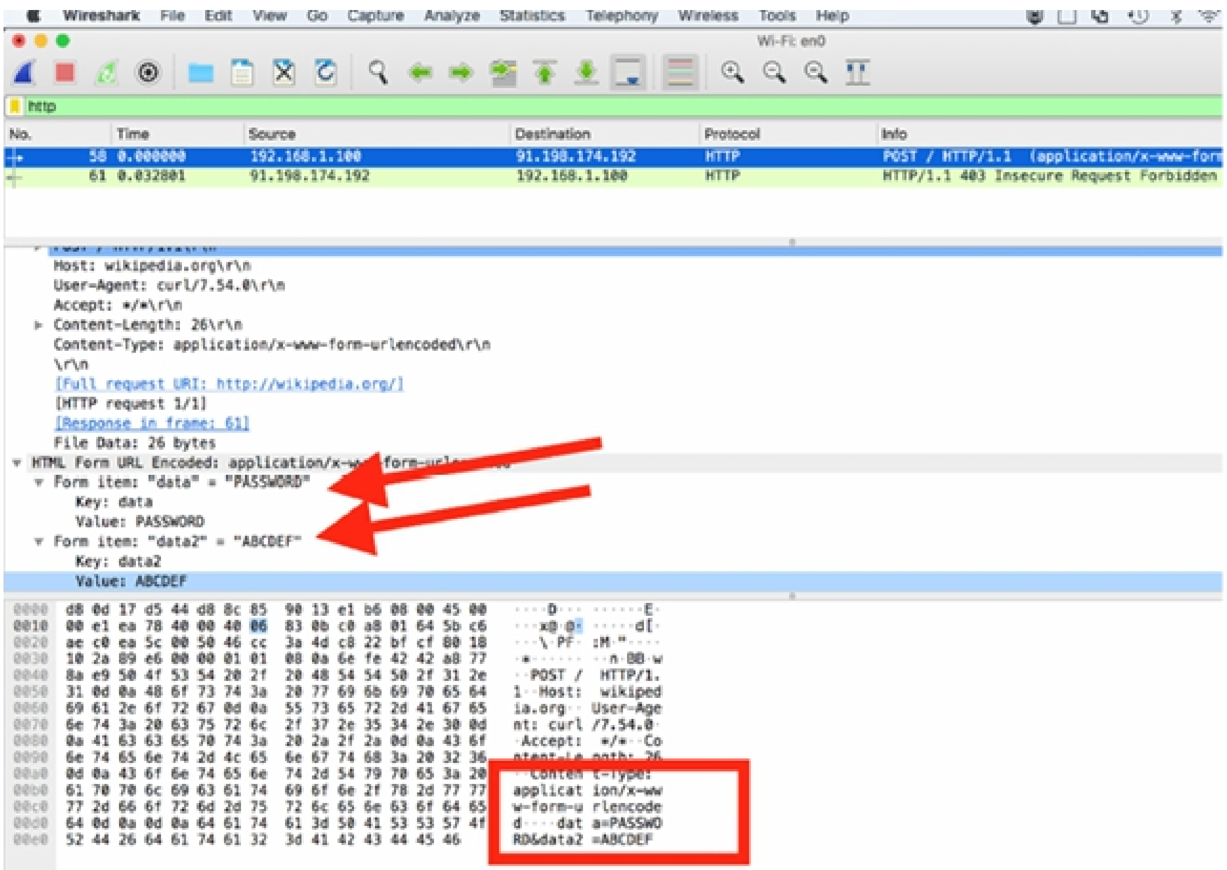
Some applications are known to use clear text passwords, and Wireshark can easily capture and display those passwords. These visible passwords are obviously a security concern.

Wireshark can be used to display any clear text communication transmitted on the network. From the network security standpoint, these applications should be examined to determine if they are releasing sensitive data on the network. From an intruder's standpoint, this information may be used to exploit network vulnerabilities. For example, the traffic from an HTTP POST operation that is setting a user password is a security vulnerability. Certainly, this password should not be sent in clear text—the password prompting page should have been accessed via a secure encrypted connection.

Validating that applications are using encryption for password settings and password input is an important step in analyzing network security. In Wireshark, capture the traffic for a few minutes on an active network interface. Open a terminal window, and run the curl -d "data=PASSWORD&data2=ABCDEF" http://wikipedia.org command, as shown in the figure below.



Stop the capture and save the file. In the filter toolbar, enter http . Inspect the Packet Details and Packet Bytes panes associated with the packet selected in the list, transmitted from your local machine, as shown in the figure below.



As shown in the figure above, you can easily spot the password "ABCDEF" in plain text. This confirms that one method to identify whether clear text passwords are crossing the network is to begin packet capture and then access the host using a password. You can also use the Find feature to look for a string in the trace file or follow the TCP Stream (or UDP Stream) to look for the password in a readable format.

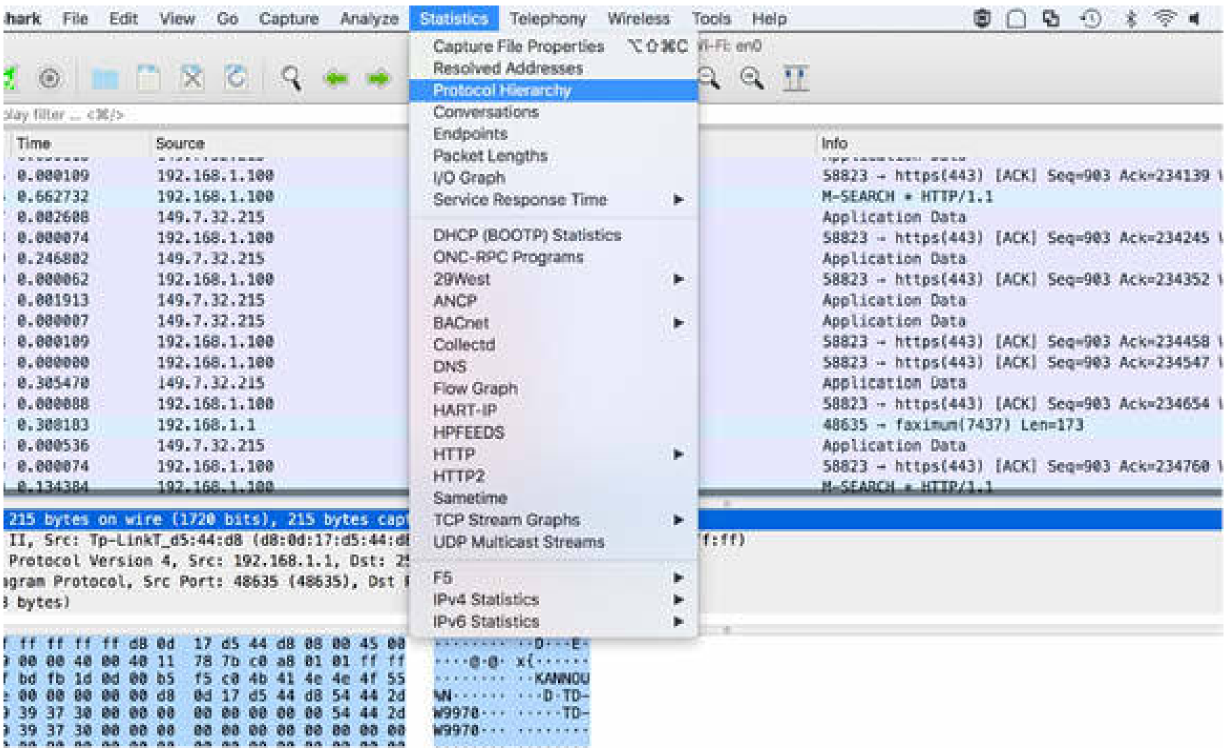
Clear text data is a concern as well. For example, if financial information is crossing the network, you would want to know this and possibly alter the data transfer process to a more secure method. Again, to detect clear text data, you might capture the traffic and then reassemble the stream to identify the clear text data.

Task 3:

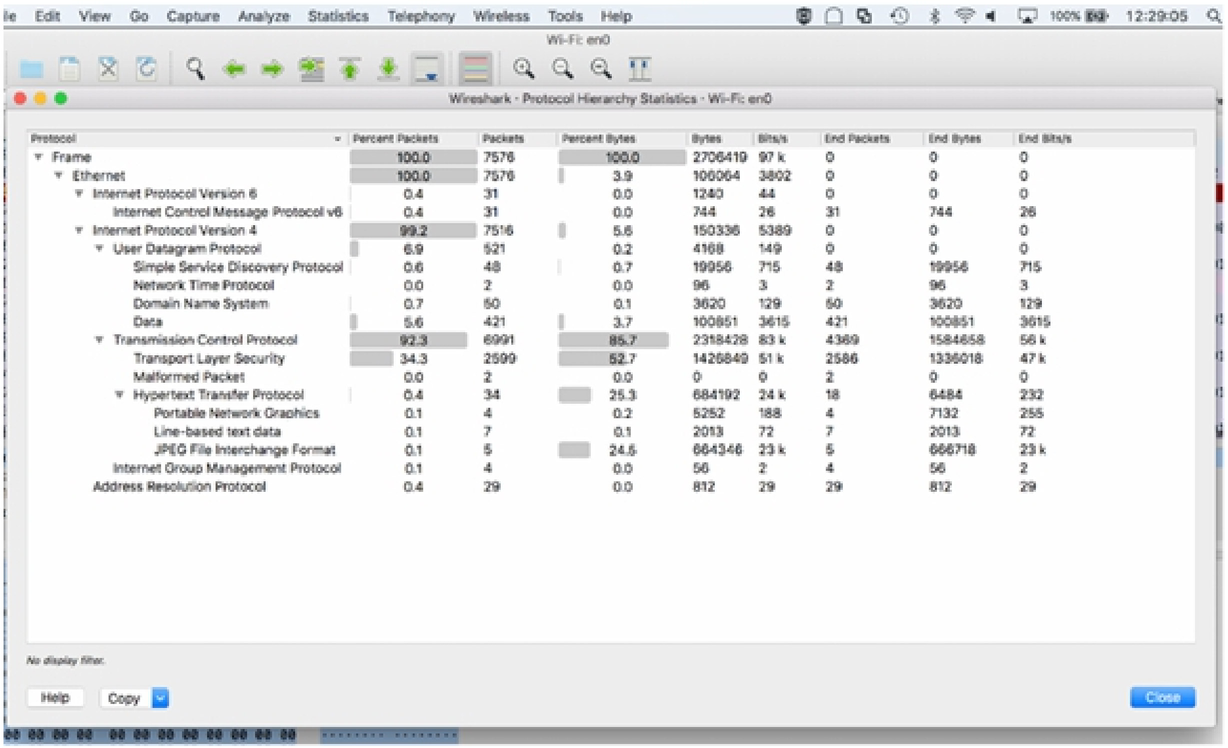
A solid baseline of normal communications assists in locating unusual protocols and applications on the network in the majority of cases. The

Protocol Hierarchy Statistics dialog box helps in identifying unusual protocols and applications in the traffic.

In Wireshark, capture the traffic for a few minutes on an active network interface. Stop the capture and save the file. On the main menu, select Statistics > Protocol Hierarchy, as shown in the figure below.



The Protocol Hierarchy Statistics dialog box is displayed, showing the hierarchical view of the protocols used in the capture saved, as shown in the figure below.



One thing that can be suspicious is that you might find traces of IRC and Trivial File Transfer Protocol (TFTP) traffic if you are not using them. In this example, it is not normal for our host.

By right-clicking the IRC or TFTP line, you can see the option to filter this traffic directly in the Packet List pane for further investigation. If you applied a display filter to the traffic before opening the Protocol Hierarchy Statistics dialog box, you won’t be able to see the statistics for the complete traffic. The applied display filter is listed just below the title bar of the dialog box.

Another thing to be considered is creating coloring rules for unusual traffic on standard ports (irc || tftp) to easily identify the traffic when you scroll through the trace file.

If the unusual traffic is using a port that Wireshark does not recognize, the Protocol Hierarchy Statistics dialog box may have a high percentage of packets listed as “Data”, immediately below the UDP or TCP main line. In such a case, right-click to filter this unrecognized traffic and reassemble the stream to look for something that identifies the purpose of the traffic.

Notes:
Repeat the previous steps and identify a target host that you can use to flood. Capture the related traffic on your local network, and verify the flood characteristics in the trace file.

Capture the traffic from and to some real hosts inside your local network to gain confidence in using the Protocol Hierarchy Statistics dialog box. Try to identify if some suspect protocols are used or if any unknown protocols are present.

Identify if there are client/server connections where the credential data is exchanged in clear text.

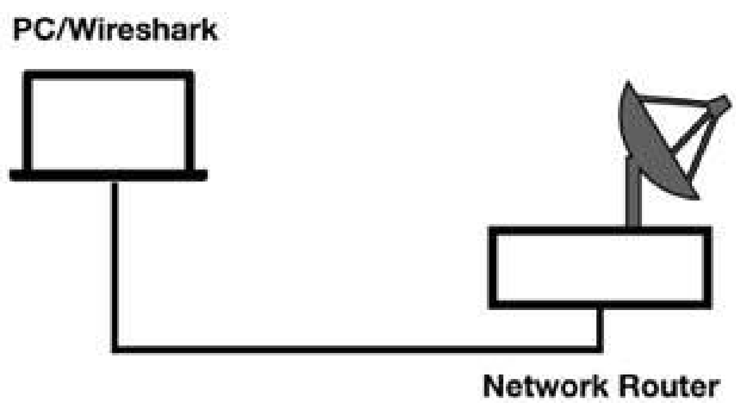
Lab 100. Route Redirection, ARP Poisoning, and TCP Splicing

Lab Objective:
Learn how to detect route redirection attacks and about ARP poisoning and TCP splicing.

Lab Purpose:
Understand how to properly identify route redirection attacks based on ICMP, attacks based on fake ARP requests, and TCP methods to bypass firewalls.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.

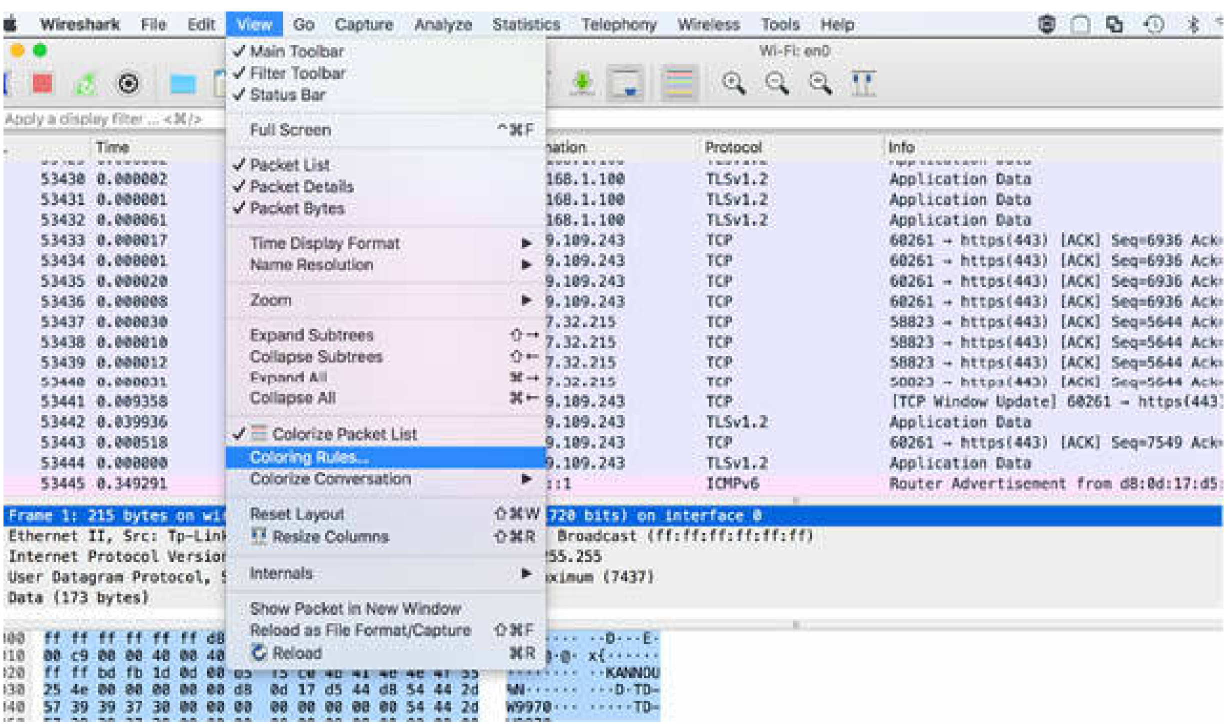


Lab Walkthrough:

Task 1:

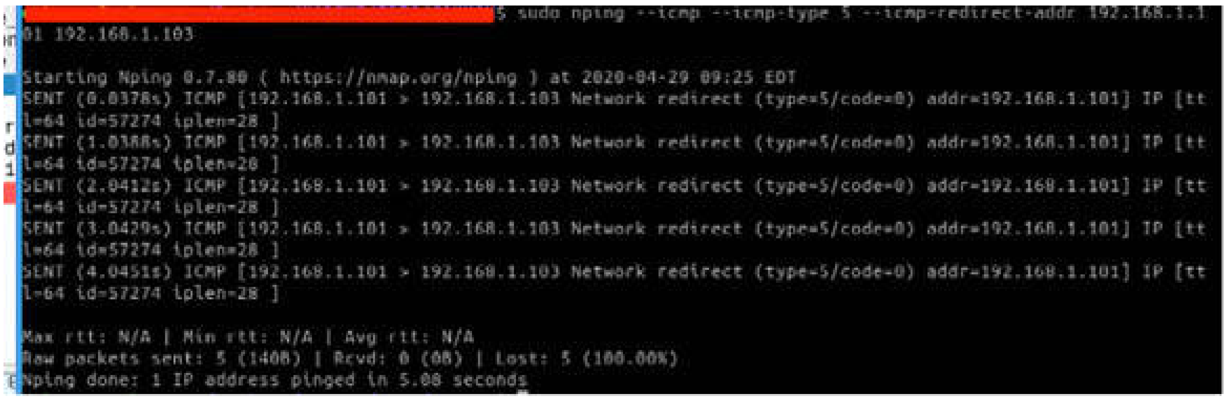
Route redirection is one of the methods used for man-in-the-middle attacks. ICMP, in the standard functionality, offers a method to dynamically discover the best router when more than one router is available on the network. When a host sends the packets to a gateway that knows of a better router to use (closer to the target network or host), it sends an ICMP Redirect (Type 5) containing the IP address of the gateway offering a better path. On receiving the packet, a host should update its routing tables to add an entry.

An attacker can use this redirection to intercept and forward the traffic that normally would not be directed to the attacker's IP address. You can create a special coloring rule (icmp.type==5) for ICMP redirections that have invalid gateway address entries. On the main menu, select View > Coloring Rules, as shown in the figure below.

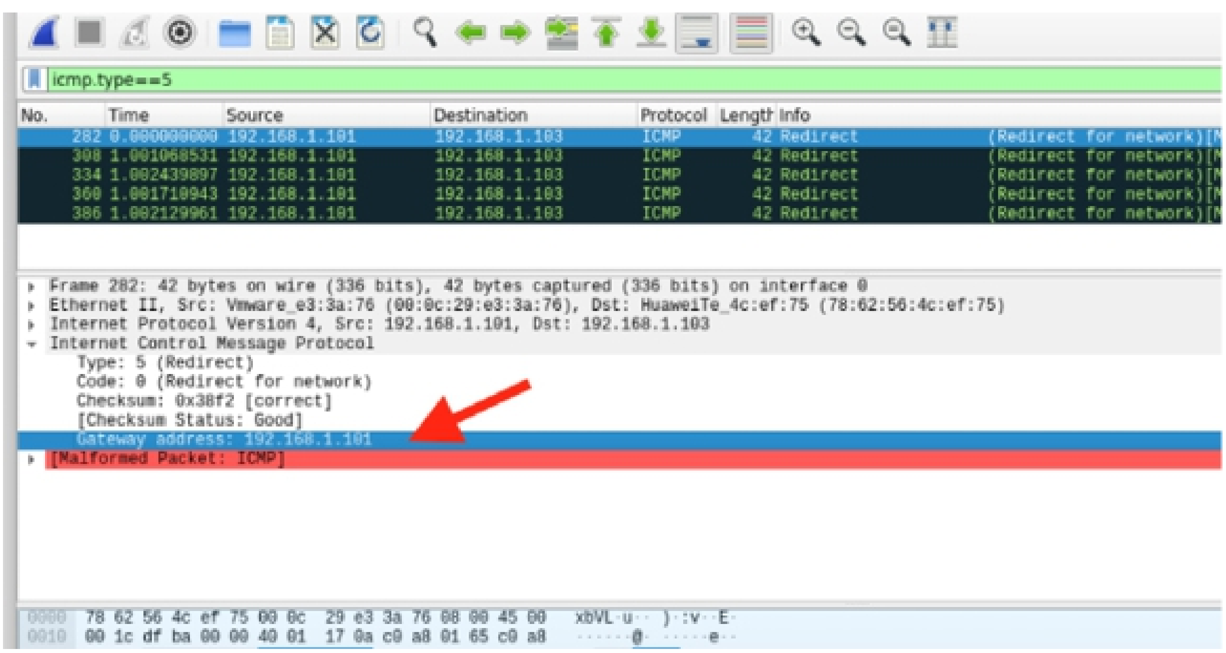


ICMP Redirect packets are also easy to spot during a live capture with a display filter (icmp.type==5).

To verify this, in Wireshark, capture the traffic for a few minutes on an active network interface. Open a terminal window (having installed the Nmap toolset), and run the command sudo nping --icmp --icmp-type 5 --icmp-redirect-addr 192.168.1.101 192.168.1.103 with administrative privileges. The first IP address is the redirect address and the second IP address is the target address, as shown in the figure below.



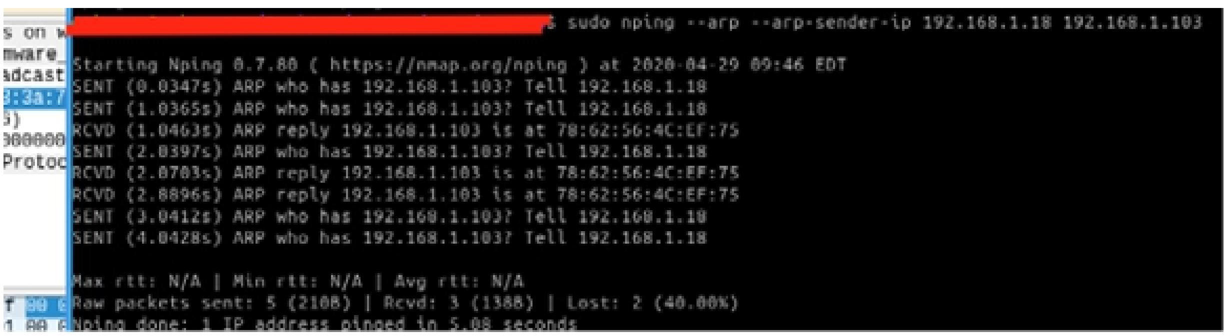
Stop the capture and save the file. In the filter toolbar, enter icmp.type==5 to verify the presence of the ICMP redirect packets. In the figure below, the highlighted fields display the gateway address specified in the command above.



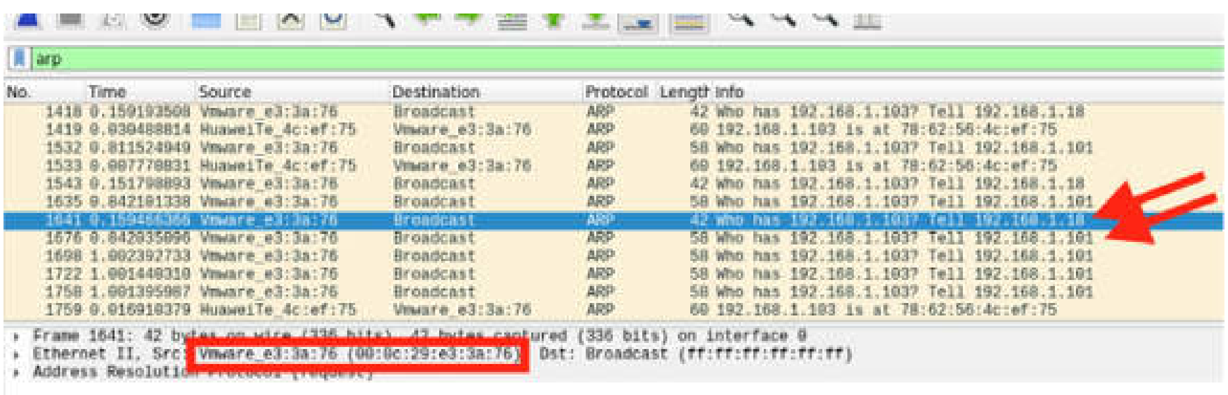
Task 2:

ARP poisoning is typically used for man-in-the-middle attacks. The attacker generates a series of ARP packets with false information that alters the ARP tables of victim hosts.

"Ettercap", "Cain and Abel", and the Nmap tool "nping" can be used to perform ARP poisoning. In Wireshark, capture the traffic for a few minutes on an active network interface. Open a terminal window, and run the command `sudo nping --arp --arp-sender-ip 192.168.1.18 192.168.1.103` , as shown in the figure below.



During the ARP poisoning process, the poisoning host is using MAC address 00:0c:29:e3:3a:76. The poisoning host states that both 192.168.1.101 and 192.168.1.18 are at MAC address 00:0c:29:e3:3a:76. You can verify this by inspecting packets #1641 and #1676, as shown in the figure below, where the display filter `arp` has been applied.

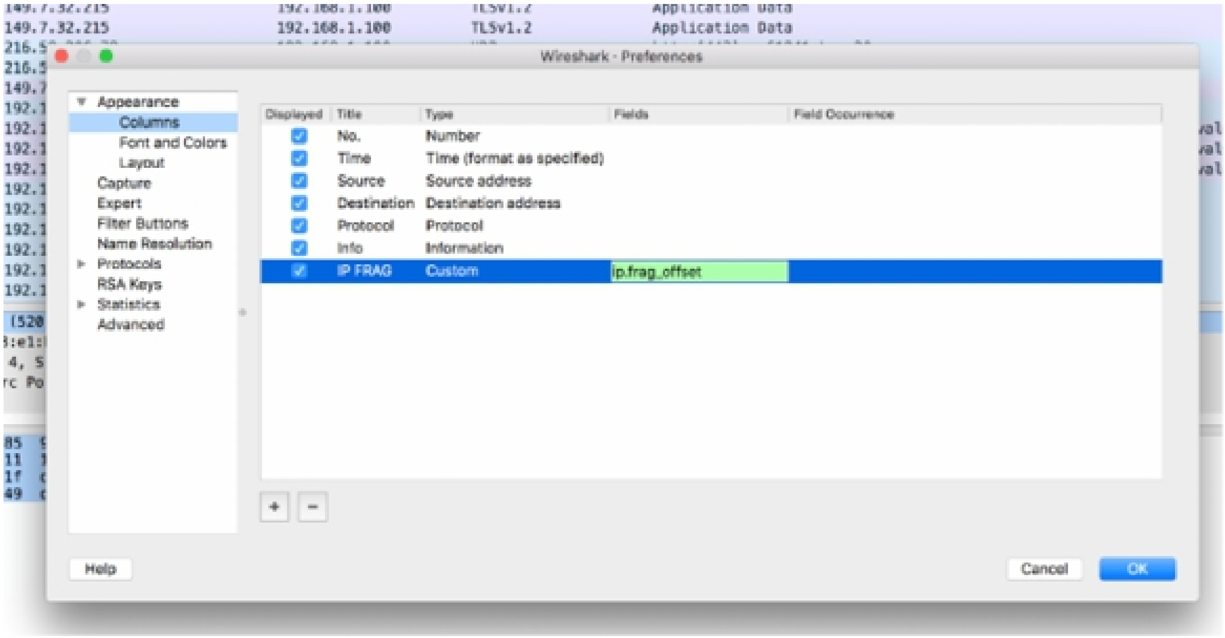


In this case, a host updates its ARP tables based on the information learned during the ARP poisoning process. When it wants to send data to another IP address, it consults its ARP table and forwards the packets to the MAC address associated with the target IP address.

Task 3:
IP fragmentation is a process that is used to split a packet into smaller sizes to traverse network segments that support smaller MTU sizes. The IP header contains three fields that define whether an IP packet may be fragmented and whether an IP packet has been fragmented. These fields are listed below:

- May Fragment field (one bit long): 0 = may fragment; 1 = don't fragment
- More Fragments field (one bit long): 0 = no more fragments to come; 1 = more fragments to come
- Fragment Offset field (13 bits long): This field is used to reassemble the fragmented data in the correct order.

Fragmentation overwriting occurs when the data coming later in a fragmented set overrides the previous data based on its Fragmentation Offset field value when the data is reassembled. To spot fragmentation override, add an ip.frag_offset column to the Packet List pane, as shown in the figure below.

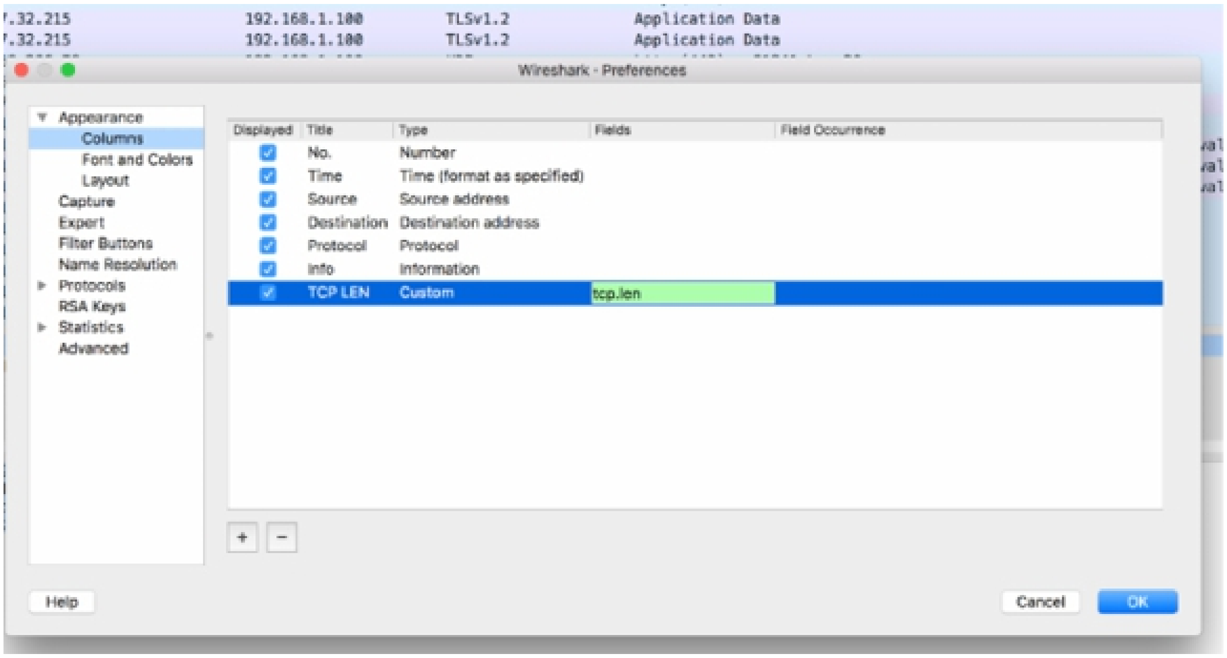


If the fragmentation offset value does not increment with each new fragment, the entire communication is considered suspicious. For example, if the ip.frag_offset column indicates that the fragment offset values are 0, 1480, 2960, 4400, 1480, 5920, 7400, and 8880, you must wonder about the fifth packet in the set. If it is not retransmission, perhaps you have the IP fragment override situation.

Task 4:

Numerous TCP evasion techniques are used to bypass firewalls, intrusion detection systems, and intrusion prevention systems.

One method of TCP evasion is TCP splicing—splitting TCP segments over multiple packets. Each packet may contain only one byte of data. This is easy to detect in Wireshark by adding a tcp.len column, as shown in the figure below.



Right-click one of the packets to reassemble these spliced packets into a single stream. To run protection checking on TCP payload values, the firewall must reassemble the TCP segments and examine the payload before forwarding or triggering events.

A continuous stream of small packets traveling in both directions on a TCP connection can indicate that splicing may be underway. Extremely small packets are expected to come from the receiving host that is sending ACK packets. Extremely small packets are not expected from the host that is sending the data.

There are numerous methods that are used to manipulate TCP communications to bypass IDS or firewall elements on the network. The following list describes some of the TCP packets that are considered unusual and possibly malicious on the network:

- TCP Segment Overwrite: One or more TCP segments in a stream overwrite one or more segments occurring earlier in the stream.
- TCP Options Occurring after an End of Options Indicator: Additional TCP options are seen in the TCP header options area after the End of Options (0) indicator.

- TCP SYN Packet Contains Data: The initial TCP SYN handshake packets contain data.
- TCP Bad Flags Combination: An illogical combination of TCP flags is seen.
- TCP URG Bit Set with Illogical Urgent Pointer Value: The Urgent Pointer bit is set, and the Urgent Pointer field points to non-existent data.
- TCP Timestamp Not Allowed: A packet contains a TCP Timestamp value when that option is not allowed in the connection.
- TCP SYN/ACK but Not SYN Window Scale Option: The second handshake packet (SYN/ACK) contains the Window Scale Option setting when the SYN packet did not contain this option.

Notes:
Repeat the previous steps to generate route redirections on your personal network and gain confidence in identifying the problem in Wireshark.

Repeat the simulation of the ARP poisoning on your local network and try to correctly detect the packets with Wireshark and gain the necessary confidence in using the tools provided by Wireshark.

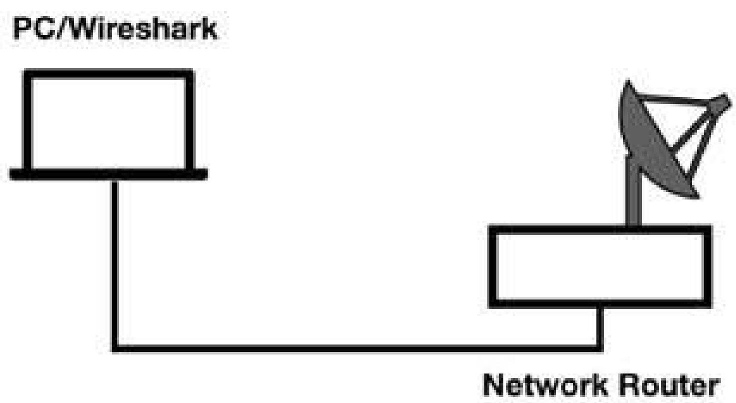
Lab 101. Command-Line Tools

Lab Objective:
Learn about the command-line tools provided by Wireshark.

Lab Purpose:
Understand how to properly use the command-line tools provided by Wireshark and when to use the command-line interface.

Lab Tool:
Wireshark installed on a PC, Ethernet switch or router (cable/Wi-Fi).

Lab Topology:
Use the topology shown in the figure below to complete this lab exercise. A PC (equipped with Wireshark) is connected through a wireless or cable connection to a network router that has access to the internet.



Lab Walkthrough:

Task 1:
Wireshark includes the following command-line tools:

- TShark
- Capinfos
- Editcap
- Mergecap
- Text2pcap
- Dumpcap
- Rawshark

In addition, in the installation folder, there is a Wireshark executable that launches GUI with different parameters available.

Add the Wireshark program file directory to your path so that you can run these tools from any directory and path. The Wireshark command syntax is `wireshark [options]` , as shown in the figure below. To view all the options, run the command `wireshark -h` .