

Lab 3: Performing Packet Capture and Traffic Analysis

Osamudiamen Eweka

Cyb-605-Z2 Principles of Cybersecurity

Utica University

Introduction

Modern networking communication aims to be transparent to end-users, facilitating the seamless exchange of vast information between interconnected computers and devices. Despite this user-friendly design, most individuals remain oblivious to the intricate details of how their applications facilitate data retrieval. The concealment of communication intricacies is intended to provide a smoother user experience, as fewer network-related decisions typically lead to greater user satisfaction (Hofstede et al., 2014). Nevertheless, this concealment poses challenges for testers, troubleshooters, and security analysts, hindering their ability to comprehend underlying network activities and address issues effectively. Security analysts require a comprehensive understanding of normal network traffic patterns and the capability to identify anomalies that may signify a security threat.

Packet capture and analyzers, commonly known as packet "sniffers," play a crucial role in addressing these challenges. Wireshark, a widely used open-source software, stands out as a premier packet capture and analysis tool (Lotfollahi et al., 2019). Despite lacking some features found in commercial alternatives, Wireshark's accessibility and cost-effectiveness make it a staple for various IT professionals, including network administrators, software developers, penetration testers, and even ethical hackers. This lab introduces the practical application of Wireshark by guiding users through a session of capturing network packets and analyzing the traffic at the packet level, emphasizing its importance in understanding, and securing modern networks.

Objective

Upon successful completion of this lab, participants will acquire the essential skills needed to navigate and analyze network traffic using Wireshark. The overarching learning objectives encompass the ability to capture live network data, analyze the captured traffic, discern crucial information corresponding to each layer within the TCP/IP protocol stack, apply filters to selectively isolate specific types of network traffic, and differentiate between the protocol stacks of wired and wireless network data. Adhering to the provided guidelines in scholarly writing is imperative in the pursuit of effective knowledge transfer. Employing a clear, direct, and simple language ensures that learners comprehend these objectives without ambiguity or unnecessary complexity. The avoidance of biased language and colloquialisms fosters an inclusive learning environment, while steering clear of rhetorical questions and contractions maintains a formal and scholarly tone. This approach aligns with the academic standards of clarity and precision, facilitating a more effective transfer of knowledge to the participants engaging in this educational endeavor.

Lab Setup

Lab Environment Details

This section presents crucial information about the computer hardware, software, and network environment essential for replicating the results of the lab exercise based on the provided instructions.

Computer Hardware:

- Manufacturer: Jones & Bartlett Learning
- Model: Remote
- Operating System: Windows 10

Software Tools:

The successful completion of this lab requires the utilization of various tools and software, each serving a specific purpose in enhancing participants' understanding of network traffic analysis. The primary tool employed is Wireshark, an open-source packet capture and analysis software, enabling users to scrutinize live network data comprehensively. PuTTY, a versatile terminal emulator, facilitates secure communication with network devices, contributing to the lab's practical aspects. FileZilla, a fast and reliable FTP client, is utilized for file transfer activities within the network environment.

In addition to these tools, the lab incorporates Mininet-WiFi, a network emulator that enables the creation of virtual wireless networks for testing and analysis. Airodump-ng and Aireplay-ng, both components of the Aircrack-ng suite, are essential for wireless network security assessments, providing functionalities such as packet capture and injection.

Section 1: Hands-On Demonstration

Part 1:

Configure Wireshark and Generate Network Traffic

Wireshark is a packet sniffer and analysis tool. It captures network traffic from ethernet, Bluetooth, wireless (IEEE.802.11), token ring, and frame relay connections, among others, and stores that data for offline analysis (Petters, 2019). In this phase of the lab, the focus is on initiating a Wireshark packet capture while employing various tools to generate diverse network traffic. Wireshark will continuously capture packets in the background until manually halted, setting the stage for subsequent analysis in the later part of the lab.

For a comprehensive understanding, a brief revisiting of the OSI Reference Model is recommended. This model comprises seven layers, with the Application Layer (Layer 7) at the top, facilitating application communication, and the Physical Layer (Layer 1) at the bottom, managing physical connections. Layer 2, the Data Link Layer, organizes bits into frames and assigns MAC addresses, typically associated with Ethernet. The Network Layer (Layer 3) encapsulates frames into packets, introducing IP addressing for inter-network communication. The Transport Layer (Layer 4) ensures reliable host communication, utilizing ports for effective data exchange. It is noteworthy that the TCP/IP Model consolidates Layers 5-7 into a single Application Layer, simplifying the traditional OSI model. In this lab, the discussion will individually cover Layers 1-4, while Layers 5-7 will be collectively addressed as the Application Layer. These layers are graphically represented below in Figure 1 (Imperva, 2021).

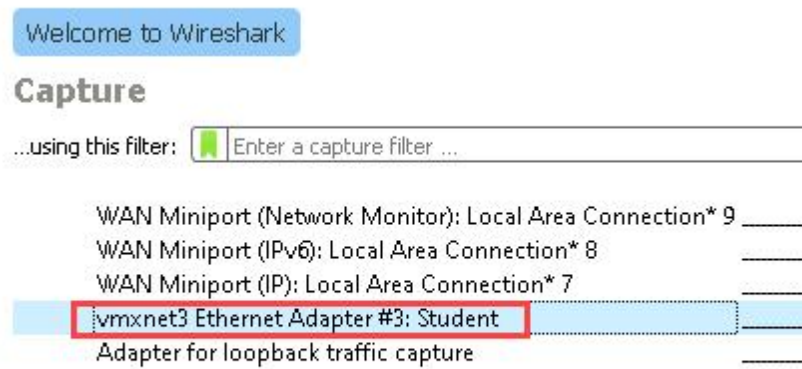
Figure 1*OSI Model Explained: The OSI 7 Layers*

7	Application Layer	Human-computer interaction layer, where applications can access the network services
6	Presentation Layer	Ensures that data is in a usable format and is where data encryption occurs
5	Session Layer	Maintains connections and is responsible for controlling ports and sessions
4	Transport Layer	Transmits data using transmission protocols including TCP and UDP
3	Network Layer	Decides which physical path the data will take
2	Data Link Layer	Defines the format of data on the network
1	Physical Layer	Transmits raw bit stream over the physical medium

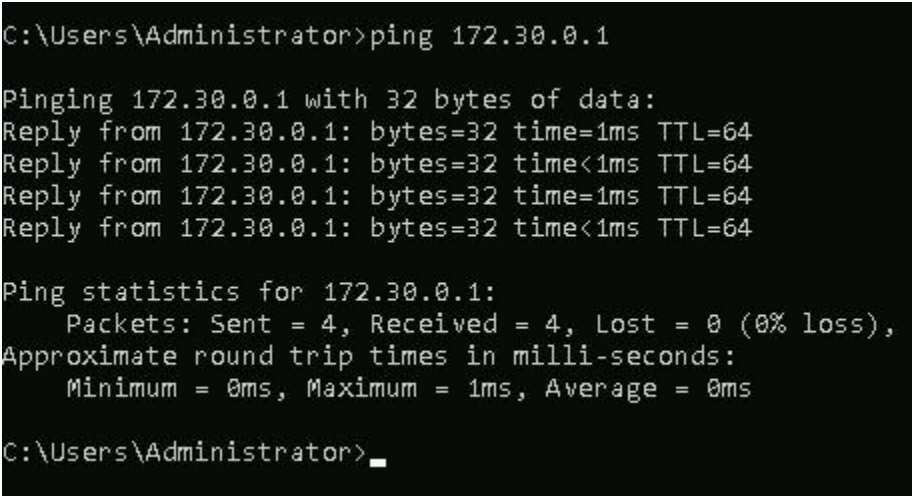
The subsequent segment of the lab involves initiating network traffic generation and utilizing Wireshark for a detailed examination of protocols. To commence, access the vWorkstation machine (172.30.0.2) using the provided administrator credentials:

- Username: Administrator
- Password: P@ssw0rd!

Proceed by double-clicking the Wireshark icon on the vWorkstation desktop to launch the Wireshark application. In the Wireshark window's Capture section, select the vmxnet3 Ethernet Adapter #3: Student interface as the designated capture interface as shown below in Figure 2 (Jones & Bartlett, 2024). This marks the inception of the lab, paving the way for practical exploration and analysis of network traffic dynamics.

Figure 2*Student interface*

To capture comprehensive network traffic, activate Wireshark in promiscuous mode, allowing the designated interface to capture all Ethernet-based frames within the LAN. After starting the capture session, perform activities to generate traffic for analysis. Begin by pinging 172.30.0.1 from the vWorkstation's command prompt to pfSense, confirming connectivity and generating ICMP Echo Requests. This ICMP protocol, associated with the Network Layer, underpins effective communication across Transport and Application Layers. This sets the stage for Wireshark analysis in the lab.

Figure 3*Ping the default Gateway*

```
C:\Users\Administrator>ping 172.30.0.1

Pinging 172.30.0.1 with 32 bytes of data:
Reply from 172.30.0.1: bytes=32 time=1ms TTL=64
Reply from 172.30.0.1: bytes=32 time<1ms TTL=64
Reply from 172.30.0.1: bytes=32 time=1ms TTL=64
Reply from 172.30.0.1: bytes=32 time<1ms TTL=64

Ping statistics for 172.30.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\Users\Administrator>_
```

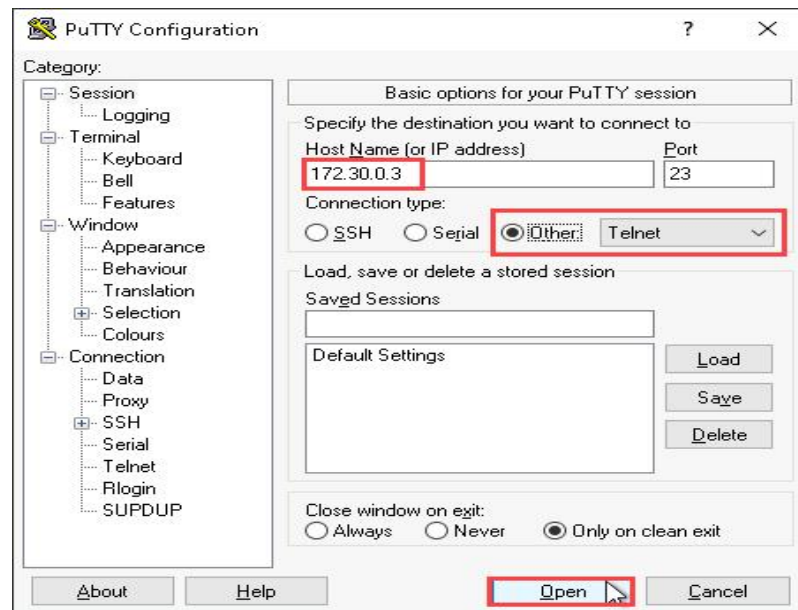
In Figure 3 (Jones & Bartlett, 2024), the procedural step involves executing commands in the command prompt to verify network connectivity. Initially, the command `ping 172.30.0.3` is employed, initiating an ICMP Echo Request directed towards the Switch01 network switch. Subsequently, the command `ping 172.30.0.5` is executed to generate an ICMP Echo Request targeting the FileServer01 host. The Ping statistics displayed in Figure 3 (Jones & Bartlett, 2024). indicate that four packets were sent, four packets were received, and zero packets were lost, affirming successful communication with both the Switch01 network switch and FileServer01 host.

Proceeding to the next steps, the utilization of PuTTY to establish a remote connection is outlined. The Command Prompt window is closed, and on the vWorkstation desktop, the PuTTY application is launched by double-clicking its icon. Within the PuTTY configuration window, `172.30.0.3` is entered in the Host Name field. By selecting the "Other" radio button and clicking "Open," a Telnet connection (port 23) is established to the Switch01 device. This progression

aligns seamlessly with the lab objectives, fostering a practical understanding of network communication and connectivity, as illustrated in Figure 4 (Jones & Bartlett, 2024).

Figure 4

PuTTY configuration window



Upon reaching the Switch01 login prompt in PuTTY, input the following credentials and press Enter to authenticate:

- Username: user
- Password: password

Please be aware that for security reasons, PuTTY will not display your password inputs on-screen, but rest assured that your entries are being logged. It is essential to enter your credentials within 30 seconds of opening the session, as PuTTY will automatically close the session if authentication is not completed within this timeframe. This login step is crucial for establishing a secure Telnet connection to the Switch01 device, facilitating subsequent network management activities in the lab.

Close the PuTTY window for the user@switch1 session. Open a new PuTTY window, type 172.30.0.1 in the Host Name field, and click Open for an SSH connection. Log in to the router with the credentials: Username: admin, Password: pfsense. Close the pfSense PuTTY window when done and click OK to confirm closure.

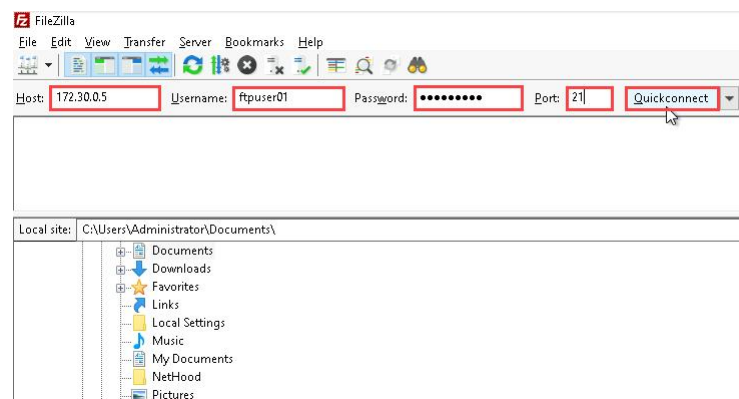
Open FileZilla by double-clicking its icon on the vWorkstation desktop. In the FileZilla Client window, input the following details and click Quickconnect to initiate an FTP connection with FileServer01:

- Host: 172.30.0.5
- Username: ftpuser01
- Password: P@ssw0rd!
- Port: 21

Figure 5 (Jones & Bartlett, 2024). should reflect the above steps These steps will facilitate an FTP file transfer between the vWorkstation (acting as the client) and the FileServer01 host (acting as the server).

Figure 5

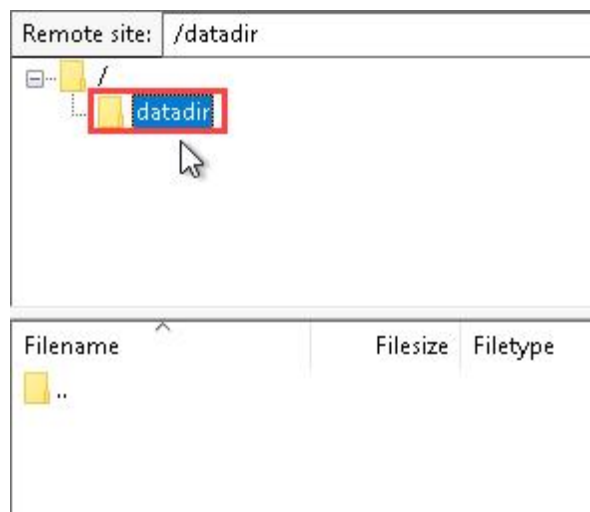
FileZilla Client.



Click OK when prompted to acknowledge that the FTP session will not be encrypted. In the Remote site section of FileZilla, double-click the datadir directory to designate it as the target directory for the FTP file transfer. As shown in Figure 6 (Jones & Bartlett, 2024).

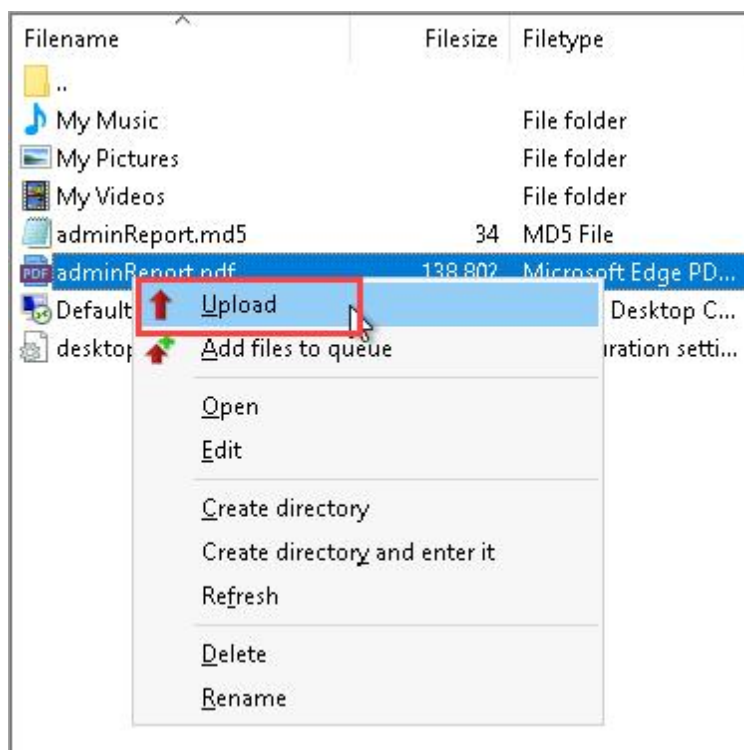
Figure 6

Datadir Directory



The Local and Remote site sections in FileZilla consist of two panes each. The upper pane displays directories on the filesystem, providing a hierarchical view of the folder structure. Simultaneously, the lower pane exhibits the files contained within the presently selected directory. This dual-pane setup enhances user navigation and simplifies the process of transferring files between the local machine and the remote server, offering a clear visual representation of both directory structures and file contents.

In the Local site section, navigate to the C:\Users\Administrator\Documents directory; then, in the lower pane, right-click the adminReport.pdf file and select Upload from the context menu to send the adminFile.txt file to the FTP server running on FileServer01. Illustrated in Figure 7 (Jones & Bartlett, 2024).

Figure 7***Upload adminfile.dpf***

In this phase, participants will engage in a secure file transfer using SFTP between the vWorkstation and FileServer01, prioritizing enhanced security through payload encryption, a feature distinct from FTP. In the FileZilla window, the following details are entered to establish an SFTP connection with FileServer01:

- Host: 172.30.0.5
- Username: ftpuser01
- Password: P@ssw0rd!
- Port: 22

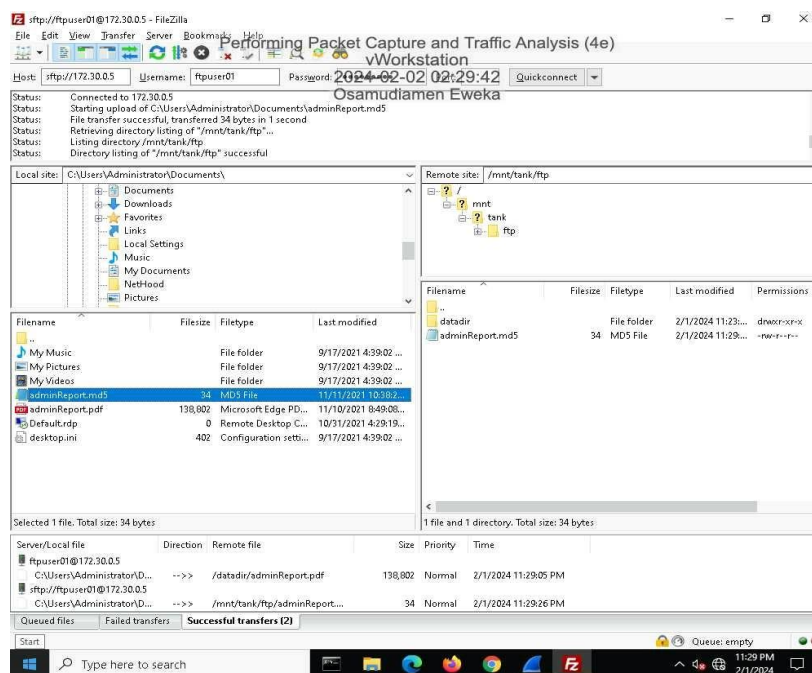
Upon successful connection, participants navigate to the Remote site section, designating the datadir directory as the destination for the upcoming SFTP file transfer.

In the Local Site section, participants initiate the SFTP transfer by right-clicking the adminReport.md5 file and select Upload from the context menu. This action sends the adminFile.txt file to the SFTP server on FileServer01. To track the transfer status, participants can access the bottom pane and select the Successful Transfers tab, providing a comprehensive log of all successful file transfers and serving as a record of the completed SFTP operation. This structured approach ensures participants follow secure file transfer protocols while maintaining transparency regarding transfer outcomes.

Participants to Make a screen capture showing the successful FTP and SFTP file transfers. This screen capture is shown in Figure 8 (Eweka, 2024).

Figure 8

Screen capture showing the successful FTP and SFTP file transfers.



Section 1: Hands-On Demonstration

Part 2:

Analyze Traffic Using Wireshark

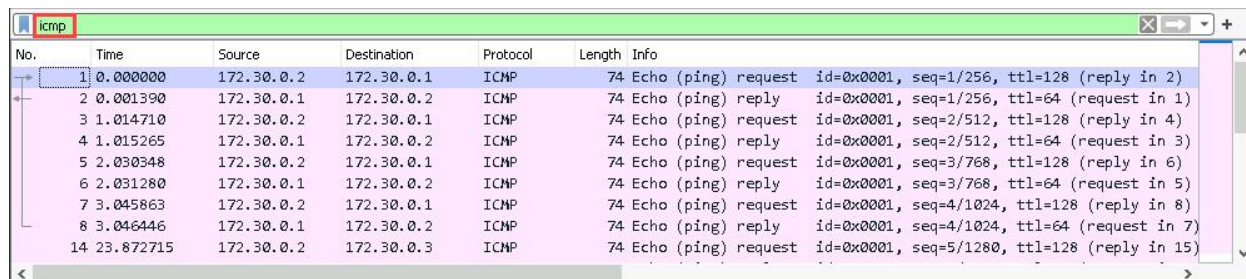
In this segment of the lab, participants engage in the analysis of generated traffic from Part 1 using Wireshark. Wireshark's analysis interface encompasses three distinct panes: the Packet List pane at the top, presenting captured packets in sequence with a human-readable summary; the middle Packet Details pane, showcasing the structure and contents of selected packet fields; and the bottom Packet Bytes pane, displaying hexadecimal data for in-depth inspection, particularly useful for uncovering unencrypted passwords.

The upcoming steps guide participants through the inspection of packets associated with each traffic type generated, extracting crucial details from these three panes. The process begins with the application of a display filter to isolate traffic related to the Ping utility. It's important to note that due to the live capture of traffic for analytical purposes, the screenshots provided in this lab section may not precisely mirror the content observed in participants' individual lab capture files. This approach empowers participants to explore the intricacies of captured packets, extracting valuable insights from the diverse perspectives offered by each of the three panes in Wireshark's analysis view.

- On the Wireshark toolbar, type icmp in the display filter bar and press Enter to apply a filter that will display only ICMP packets. Illustrated in Figure 9 (Jones & Bartlett, 2024).

Figure 9

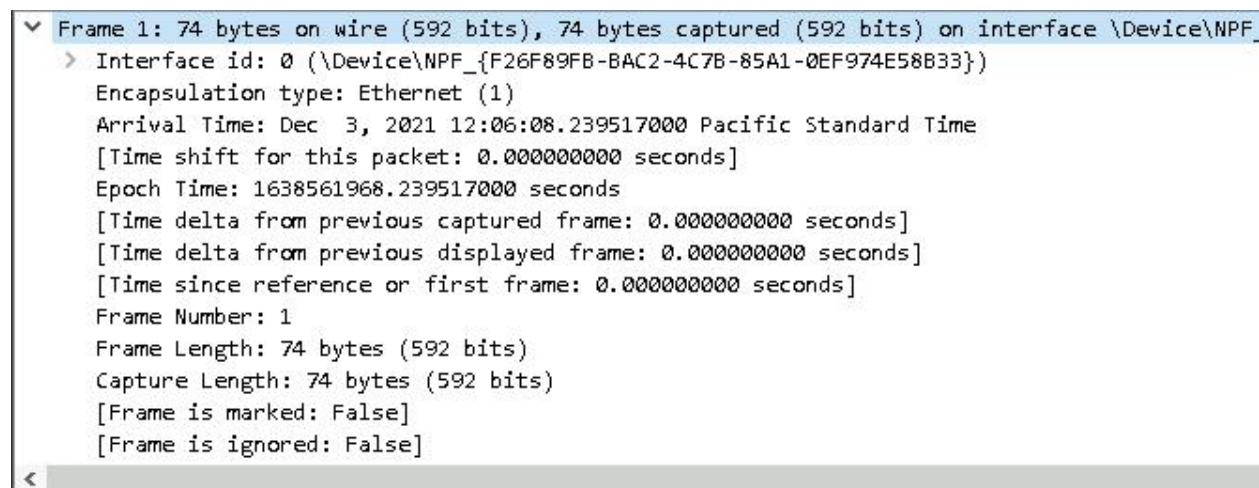
Apply ICMP display filter



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.30.0.2	172.30.0.1	ICMP	74	Echo (ping) request id=0x0001, seq=1/256, ttl=128 (reply in 2)
2	0.001390	172.30.0.1	172.30.0.2	ICMP	74	Echo (ping) reply id=0x0001, seq=1/256, ttl=64 (request in 1)
3	1.014710	172.30.0.2	172.30.0.1	ICMP	74	Echo (ping) request id=0x0001, seq=2/512, ttl=128 (reply in 4)
4	1.015265	172.30.0.1	172.30.0.2	ICMP	74	Echo (ping) reply id=0x0001, seq=2/512, ttl=64 (request in 3)
5	2.030348	172.30.0.2	172.30.0.1	ICMP	74	Echo (ping) request id=0x0001, seq=3/768, ttl=128 (reply in 6)
6	2.031280	172.30.0.1	172.30.0.2	ICMP	74	Echo (ping) reply id=0x0001, seq=3/768, ttl=64 (request in 5)
7	3.045863	172.30.0.2	172.30.0.1	ICMP	74	Echo (ping) request id=0x0001, seq=4/1024, ttl=128 (reply in 8)
8	3.046446	172.30.0.1	172.30.0.2	ICMP	74	Echo (ping) reply id=0x0001, seq=4/1024, ttl=64 (request in 7)
14	23.872715	172.30.0.2	172.30.0.3	ICMP	74	Echo (ping) request id=0x0001, seq=5/1280, ttl=128 (reply in 15)

In this lab, display filters are a crucial tool for refining the displayed packets in the packet list based on specific protocols or field values. By applying a filter for ICMP protocol, the packet list exclusively shows ICMP-related packets. Columns provide key details such as packet number, timestamp, source (vWorkstation for Echo requests), destination (pinged systems for Echo replies), protocol (exclusively ICMP), length, and additional information. This concise overview enhances understanding of ICMP traffic dynamics, revealing successful ping commands and key packet details within the limited character count.

- Next In the Packet Details pane, double-click the first row (Frame 1) to display additional information about the selected packet. Illustrated in Figure 10 (Jones & Bartlett, 2024).

Figure 10*Packet Details for ICMP packet*

 The image shows the 'Packet Details' pane in Wireshark. The top line is expanded, showing 'Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{F26F89FB-BAC2-4C7B-85A1-0EF974E58B33}'. Below this, a list of details is shown: 'Interface id: 0 (\Device\NPF_{F26F89FB-BAC2-4C7B-85A1-0EF974E58B33})', 'Encapsulation type: Ethernet (1)', 'Arrival Time: Dec 3, 2021 12:06:08.239517000 Pacific Standard Time', '[Time shift for this packet: 0.000000000 seconds]', 'Epoch Time: 1638561968.239517000 seconds', '[Time delta from previous captured frame: 0.000000000 seconds]', '[Time delta from previous displayed frame: 0.000000000 seconds]', '[Time since reference or first frame: 0.000000000 seconds]', 'Frame Number: 1', 'Frame Length: 74 bytes (592 bits)', 'Capture Length: 74 bytes (592 bits)', '[Frame is marked: False]', and '[Frame is ignored: False]'. A scroll bar is visible at the bottom of the pane.


```

  ▼ Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_
    > Interface id: 0 (\Device\NPF_{F26F89FB-BAC2-4C7B-85A1-0EF974E58B33})
      Encapsulation type: Ethernet (1)
      Arrival Time: Dec 3, 2021 12:06:08.239517000 Pacific Standard Time
      [Time shift for this packet: 0.000000000 seconds]
      Epoch Time: 1638561968.239517000 seconds
      [Time delta from previous captured frame: 0.000000000 seconds]
      [Time delta from previous displayed frame: 0.000000000 seconds]
      [Time since reference or first frame: 0.000000000 seconds]
      Frame Number: 1
      Frame Length: 74 bytes (592 bits)
      Capture Length: 74 bytes (592 bits)
      [Frame is marked: False]
      [Frame is ignored: False]
  
```

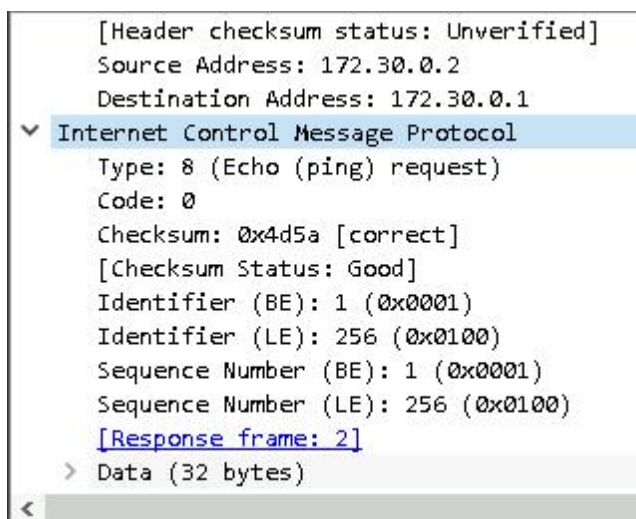
In network models, encapsulation is a key concept where each layer adds information as a message descends. The Application Layer includes payload and addresses, while the Transport Layer organizes messages into segments with source/destination ports. The Network Layer converts segments into packets, appending IP addresses for routing. The Data Link Layer encapsulates packets into frames with MAC addresses for network transmission. Wireshark illustrates this correlation, with each packet corresponding to a frame, sharing the same number. Packet info pertains to the Network Layer, and frame info relates to the Data Link Layer, showcasing the layered approach optimizing network functionality.

- In the Packet Details pane, double-click the second row (Ethernet II) to display additional information about the Data Link Layer protocol for the selected packet.

The Data Link Layer is specifically concerned with communication within a Local Area Network (LAN). In this layer, the crucial address is the MAC (Media Access Control) address. The source and destination addresses within frames are represented by MAC addresses. A MAC

address is composed of six hexadecimal pairs. The first three pairs are unique to the manufacturer, identifying the device's vendor, while the last three pairs are unique to the hardware device itself. In the context mentioned, the MAC addresses signify that VMware is the manufacturer of the Network Interface Card (NIC). These MAC addresses play a pivotal role in LAN communication, ensuring that data frames reach the intended devices within the local network.

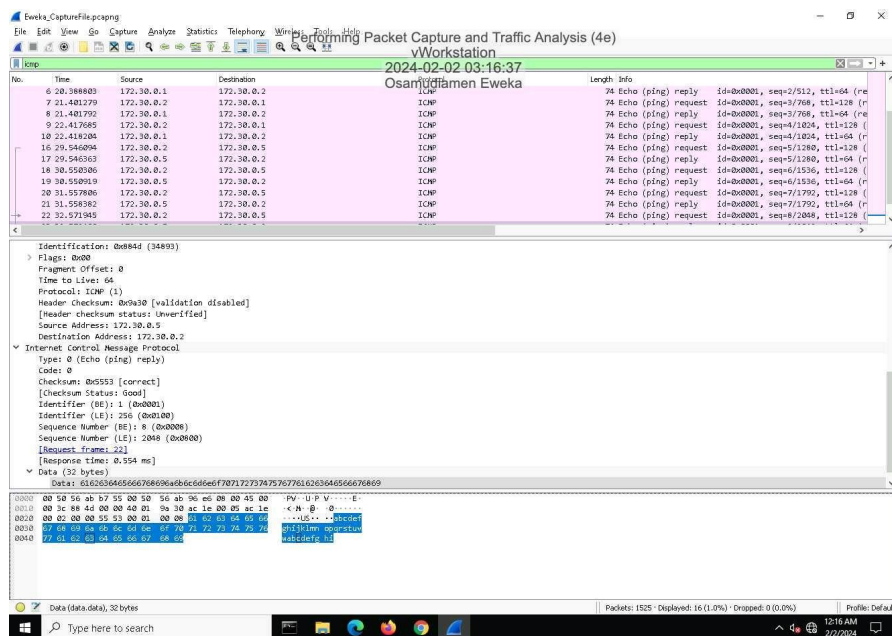
In the Packet Details pane, double-click the third row (Internet Protocol Version 4) to access additional Network Layer protocol information, showcasing IP addresses as source and destination. Since you're inspecting an ICMP message, a Network Layer protocol, there is no further Transport Layer encapsulation. Observe a fourth row for ICMP-specific details in the Packet Details pane, reflecting Wireshark's specialization for ICMP packets. Double-clicking the fourth row (Internet Control Message Protocol) unveils additional ICMP protocol details shown in Figure 11 (Jones & Bartlett, 2024). The listed information parallels what's visible in the Info column of the Packet List pane. Note that the ICMP packet's data payload is dummy data, and upcoming steps involve a direct examination of this payload for verification.

Figure 11*ICMP details*

In Wireshark's Packet Details pane, I expand the "Data" sub-header, highlighting the ICMP packet's payload. The Packet Bytes pane then displays the hexadecimal and translated data, showing alphabet letters ("abcdefgh..."). ICMP messages typically use dummy data as they focus on control messages, not extensive payloads. Participants to Make a screen capture showing ICMP payload.. This screen capture is shown in Figure 12 below (Eweka, 2024).

Figure 12

A screen capture showing ICMP payload.



Next In the Wireshark toolbar, input "telnet" into the Filter bar and press Enter. This action filters and displays only the packets related to the Telnet session with Switch01 (172.30.0.3) from Part 1 of the lab. If required, choose the first packet in the Packet List pane for further analysis. Shown in Figure 13 below.

Figure 13

TCP display filter

No.	Time	Source	Destination	Protocol	Length	Info
37	116.660493	172.30.0.2	172.30.0.3	TELNET	75	Telnet Data ...
39	116.719870	172.30.0.3	172.30.0.2	TELNET	66	Telnet Data ...
40	116.720201	172.30.0.2	172.30.0.3	TELNET	57	Telnet Data ...
41	116.720533	172.30.0.3	172.30.0.2	TELNET	66	Telnet Data ...
42	116.720728	172.30.0.2	172.30.0.3	TELNET	63	Telnet Data ...
43	116.720885	172.30.0.3	172.30.0.2	TELNET	72	Telnet Data ...
44	116.721054	172.30.0.2	172.30.0.3	TELNET	71	Telnet Data ...
45	116.721151	172.30.0.2	172.30.0.3	TELNET	60	Telnet Data ...
46	116.721225	172.30.0.2	172.30.0.3	TELNET	65	Telnet Data ...

In the Packet details, expand the fourth row (Transmission Control Protocol) to analyze Telnet, an Application Layer protocol utilizing TCP (Transmission Control Protocol). Unlike ICMP, Telnet operates at the Application Layer, employing TCP as a connection-oriented Transport Layer protocol. TCP ensures delivery through mechanisms like the three-way handshake, acknowledgment for each received packet, and sequence numbers for tracking and proper reassembly. Key fields within the TCP row include ports and sequencing information, with source and destination ports facilitating proper service addressing, and sequence numbers aiding in segment reassembly to reconstruct Application Layer messages. Telnet commonly uses TCP port 23, but any port can be employed, provided both endpoints are configured accordingly.

Figure 14

Telnet Detail

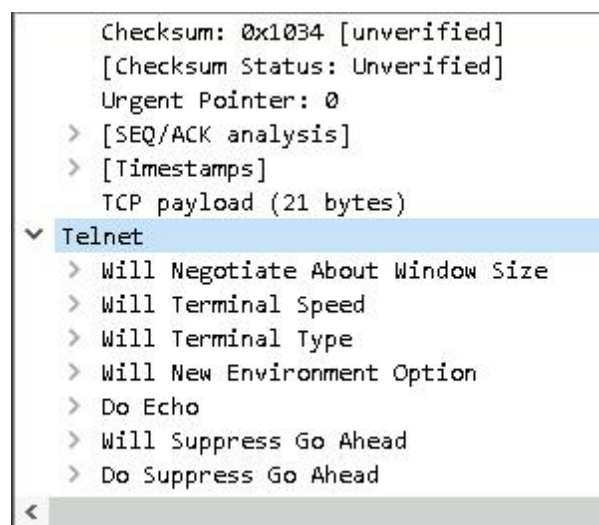


Figure 14 (Jones & Bartlett, 2024). In the Packet Details pane, expand the fifth row to delve into Telnet configuration. Telnet negotiates window size, terminal speed, type, and echoes characters to the destination. Despite the potential for customization, Telnet defaults lack transparency, exposing transmitted data. Telnet is insecure as it doesn't encrypt the payload, making user IDs and passwords vulnerable to interception by protocol analyzers like Wireshark.

Due to its inherent security risks, it is strongly advised against using Telnet for confidential communications.

Next, participant will confirm Telnet's lack of encryption by examining plaintext credentials from your session. Select the first packet in the Packet List pane, using the down arrow key to analyze Telnet row details. As the session initiates, you'll encounter familiar prompts like "Switch01 login:", exposing entered username and password characters.

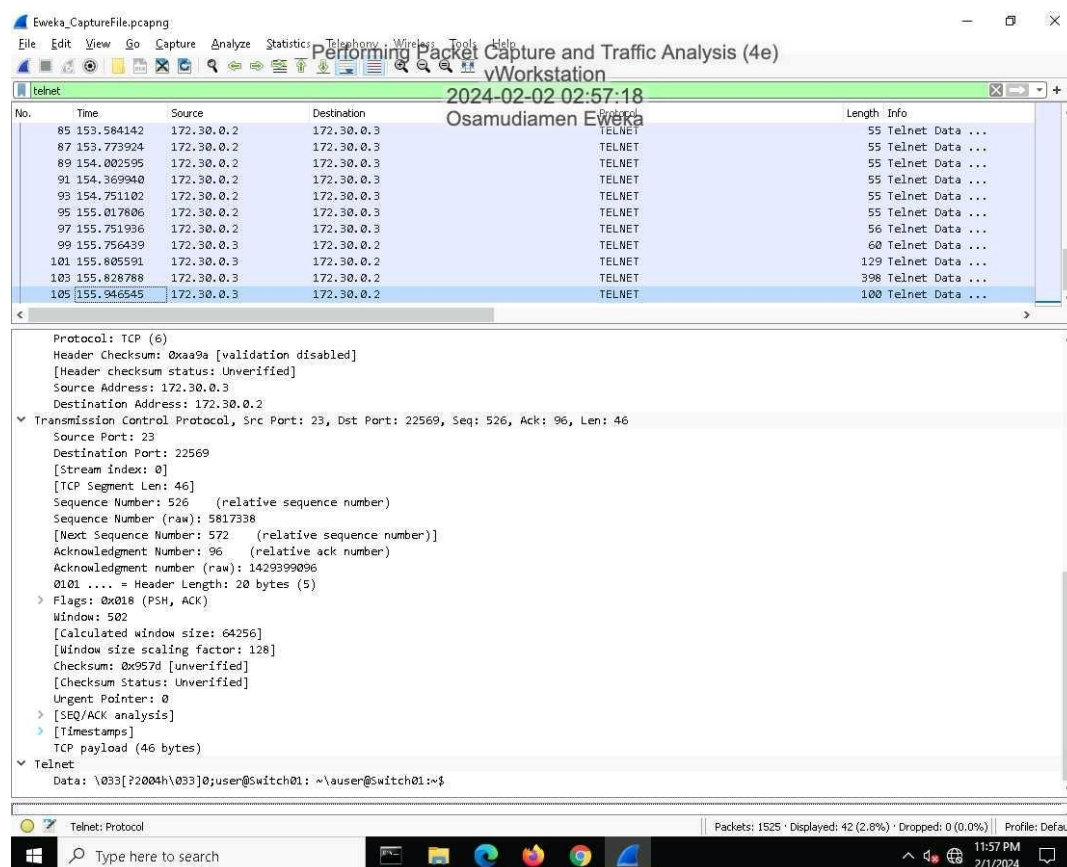
PuTTY defaults to sending each typed character immediately, but the linemode configuration sends the entire string after pressing Enter, minimizing traffic. Observe potential username character duplication, clarified by IP addresses showing one sent to the server and one echoed to the PuTTY console. Password characters, however, remain singular for confidentiality.

Choose the last Telnet packet, revealing the data payload with the command prompt, "`033)0;user@Switch01: ~\auser@Switch01:~$`". This echo post-login emphasizes Telnet's insecure nature, transmitting data in plaintext, exposing login details.

Participants to Make a screen capture showing the Last Login: information in the Packet Details pane. This screen capture is shown in Figure 15 (Eweka, 2024).

Figure 15

Screen capture showing the Last Login: information in the Packet Details pane.

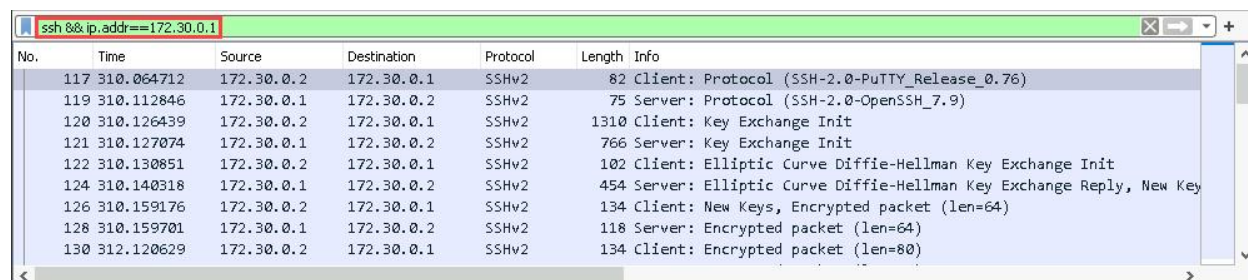


On the Wireshark toolbar, select and type "ssh" in the Filter bar, then press Enter, isolating packets related to SSHv2 sessions with pfSense (172.30.0.1) and SFTP with FileServer01 (172.30.0.5). SSHv2, utilizing TCP, employs the three-way handshake, acknowledgments, and packet sequencing. Unlike Telnet, SSHv2 encrypts payload, ensuring user credentials' confidentiality. SFTP, an extension of SSH, secures file transfers between client and server. These SSH-filtered packets, including SFTP sessions, manifest on TCP port 22. Refine the filter by appending "&& ip.addr==172.30.0.1" and pressing Enter, highlighting

entries linked to the SSH connection with the pfSense gateway router. Illustrated in Figure 16 (Jones & Bartlett, 2024).

Figure 16

Update display filter with gateway router IP address



No.	Time	Source	Destination	Protocol	Length	Info
117	310.064712	172.30.0.2	172.30.0.1	SSHv2	82	Client: Protocol (SSH-2.0-PuTTY_Release_0.76)
119	310.112846	172.30.0.1	172.30.0.2	SSHv2	75	Server: Protocol (SSH-2.0-OpenSSH_7.9)
120	310.126439	172.30.0.2	172.30.0.1	SSHv2	1310	Client: Key Exchange Init
121	310.127074	172.30.0.1	172.30.0.2	SSHv2	766	Server: Key Exchange Init
122	310.130851	172.30.0.2	172.30.0.1	SSHv2	102	Client: Elliptic Curve Diffie-Hellman Key Exchange Init
124	310.140318	172.30.0.1	172.30.0.2	SSHv2	454	Server: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Key
126	310.159176	172.30.0.2	172.30.0.1	SSHv2	134	Client: New Keys, Encrypted packet (len=64)
128	310.159701	172.30.0.1	172.30.0.2	SSHv2	118	Server: Encrypted packet (len=64)
130	312.120629	172.30.0.2	172.30.0.1	SSHv2	134	Client: Encrypted packet (len=80)

In the Packet List pane, choose the first packet and use the down arrow key to navigate through packets, analyzing the ASCII column in the Packet Bytes pane to identify payload contents. Similar to Telnet, initial SSH packets negotiate connection parameters. Differently, SSH negotiates a transform set, determines encryption and hashing algorithms, and engages in a key exchange to establish a session key for future encryption. Only specific packets, like "Client: Protocol," "Server: Protocol," "Client: Key Exchange Init," and "Server: Key Exchange Init," display readable data in the ASCII portion. Others remain encrypted, requiring decryption for content interpretation. Select the "Server: Key Exchange Init" packet for further analysis. Figure 17 below (Jones & Bartlett, 2024) , shows the above actions

Figure 17

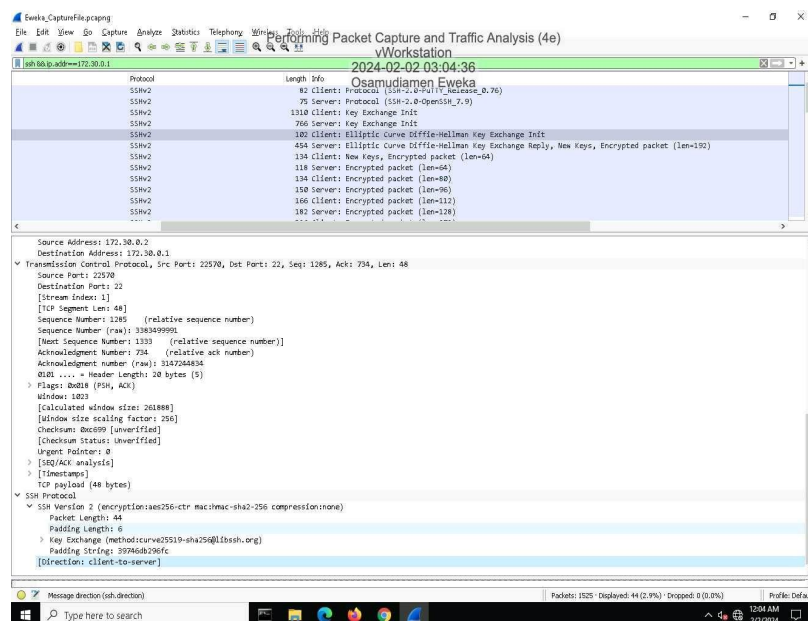
Server: Key Exchange Init packet

No.	Time	Source	Destination	Protocol	Length	Info
117	310.064712	172.30.0.2	172.30.0.1	SSHv2	82	Client: Protocol (SSH-2.0-PuTTY_Release_0.76)
119	310.112846	172.30.0.1	172.30.0.2	SSHv2	75	Server: Protocol (SSH-2.0-OpenSSH_7.9)
120	310.126439	172.30.0.2	172.30.0.1	SSHv2	1310	Client: Key Exchange Init
121	310.127074	172.30.0.1	172.30.0.2	SSHv2	766	Server: Key Exchange Init
122	310.130851	172.30.0.2	172.30.0.1	SSHv2	102	Client: Elliptic Curve Diffie-Hellman Key Exchange Init
124	310.140318	172.30.0.1	172.30.0.2	SSHv2	454	Server: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Key
126	310.159176	172.30.0.2	172.30.0.1	SSHv2	134	Client: New Keys, Encrypted packet (len=64)
128	310.159701	172.30.0.1	172.30.0.2	SSHv2	118	Server: Encrypted packet (len=64)
130	312.120629	172.30.0.2	172.30.0.1	SSHv2	134	Client: Encrypted packet (len=80)

Double-click the SSH Protocol row in the Packet Details pane to reveal SSH details, showcasing the negotiated encryption and hashing algorithms. In this context, "mac" doesn't refer to the physical MAC address but denotes a Message Authentication Code. This uses hashing for one-way encryption to verify the sender's authenticity and ensure data integrity during transmission. You are Capture a screenshot to document the SSHv2 encryption and mac configurations for the SSH connection. Figure 18 below shows this screen capture (Eweka, 2024)

Figure 18

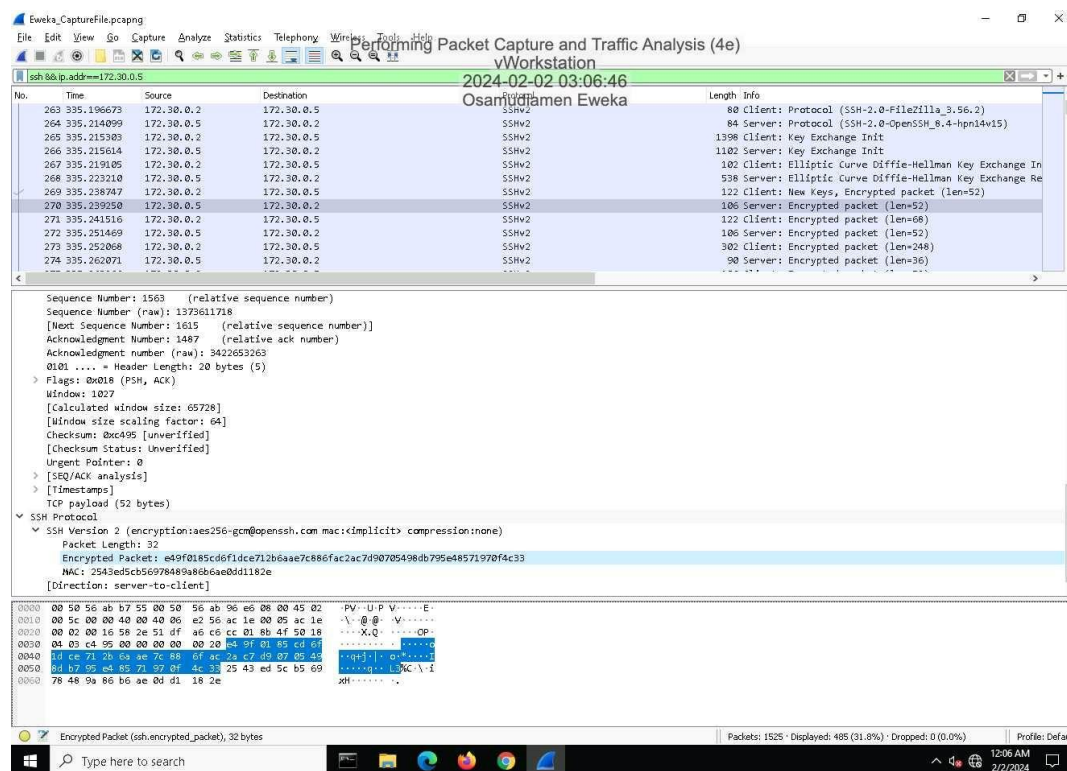
Make a screen capture SSHv2 encryption and mac selections for the SSH connection.



In the Filter bar, highlight 172.30.0.1, then type 172.30.0.5 and press Enter to inspect SSH packets related to the SFTP session with FileServer01. The packet sequence mirrors the default gateway session, with variations only in the executed commands, considering this session involves file transfers from the vWorkstation to FileServer01. However, encrypted payloads obscure specific commands. In the Packet List pane, choose the first "Server: Encrypted Packet" packet. In the Packet Details pane, double-click the SSH Version 2 row to explore SSHv2 details. Click on the Encrypted Packet row to highlight it, revealing all encrypted message data from the vWorkstation (client) to FileServer01 (server). Wireshark's data highlighting simplifies pinpointing pertinent information during the investigation. In the lab, participants are to Make a screen capture showing the highlighted (encrypted) data in the Packet Bytes pane. This is shown in Figure 19 below (Eweka, 2024)

Figure 19

Highlighted (encrypted) data in the Packet Bytes pane.



On Wireshark, filter FTP traffic by entering "ftp" in the Filter bar. This displays packets from the FTP session with FileServer01. FTP, using TCP, lacks encryption, unlike SFTP. FTP employs two channels: one for control (commonly port 21) and another for data transfer (usually port 20). In contrast, SFTP utilizes a single channel on SSH port 22.

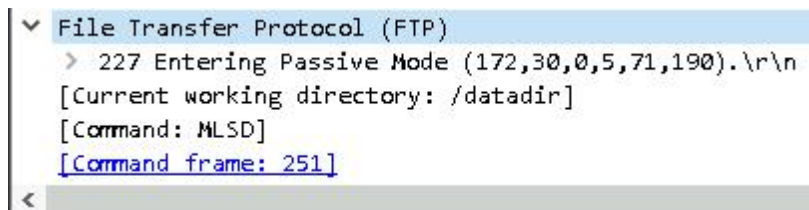
Analyze initial packets, indicating server type and TLS disallowance, associated with FTPS. TLS in FTPS offers security akin to SFTP. FTPS typically uses port 990 for control and 989 for data transfer but may use port 21 like FTP.

Explore subsequent packets, showing authentication prompts and visible login credentials due to FTP's non-encryption. Select the packet with Response: 227 Entering Passive Mode. In the Packet Details pane, double-click the File Transfer Protocol row for FTP details. Locate and select the packet with Response: 227 Entering Passive Mode in the Info column. in

the Packet Details pane, open the File Transfer Protocol row to view the FTP details. Shown below in Figure 20 (Jones & Bartlett, 2024).

Figure 20

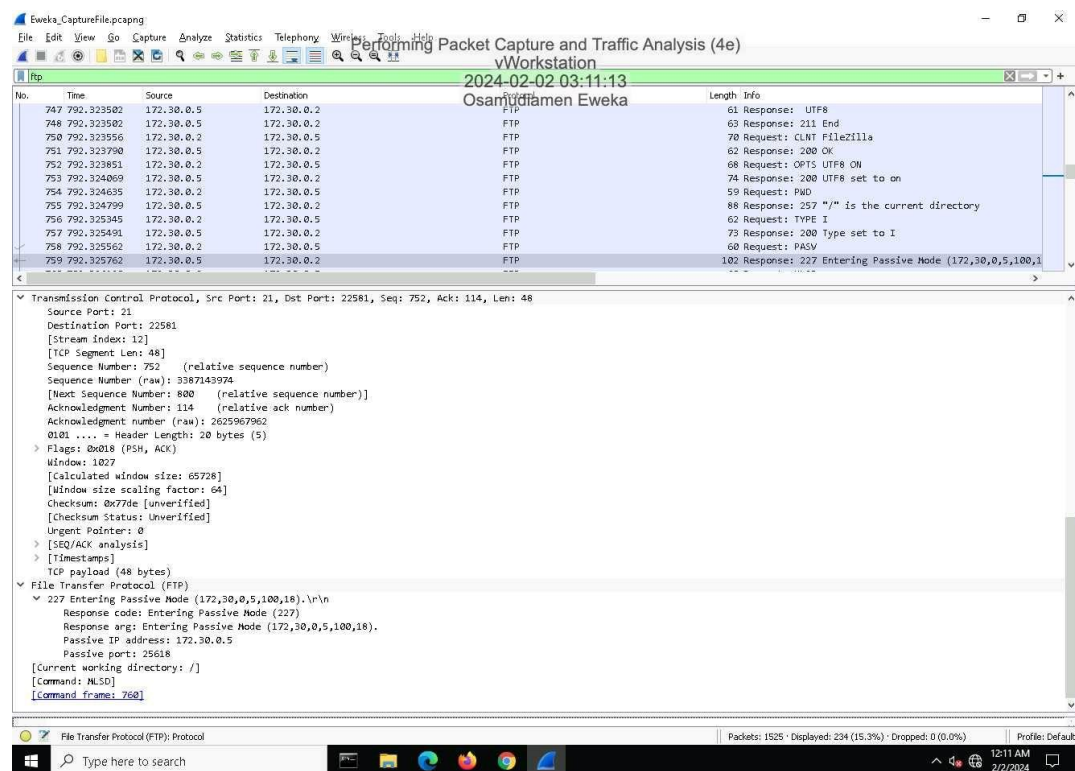
FTP details.



In FTP connections, Active mode involves the client specifying a random listening port for data on the control port 21. The server initiates a connection back to the client. However, this mode is less common due to firewalls. Passive mode, widely used today, has the client request ports from the server. The server responds with a string, and by manipulating the values, the client determines the port for the data connection. Figure 21 (Eweka, 2024) Capture and illustrate this Passive port information from the Packet Details pane for a comprehensive understanding.

Figure 21

Make a screen capture passive port specified by the FTP server in the Packet Details pane



Next using Wireshark to filter and inspect packets related to the FTP data connection. By applying a filter for "ftp-data," the display is narrowed down to only those packets involved in FTP data transmission. Subsequently, selecting the first FTP data packet sent from the vWorkstation to FileServer01 allows for a detailed examination of the Transmission Control Protocol (TCP) details, specifically focusing on the Destination Port value. The screen capture of this Destination Port field in the Packet Details pane in Figure 22 below (Eweka, 2024), is intended to capture and document the specific port associated with the FTP data connection, providing valuable insights into the communication between the client (vWorkstation) and the server (FileServer01).

Figure 22

Destination Port field value in the Packet Details pane.

The image shows a Wireshark packet capture analysis. The top pane displays a list of captured packets. The bottom pane shows the detailed view of a selected packet (No. 819).

No.	Time	Source	Destination	Protocol	Length	Info
233	298.457303	172.30.0.5	172.30.0.2	FTP-DATA	553	FTP Data: 499 bytes (PASV) (NLSD)
250	303.002038	172.30.0.5	172.30.0.2	FTP-DATA	572	FTP Data: 518 bytes (PASV) (NLSD)
765	792.327240	172.30.0.5	172.30.0.2	FTP-DATA	553	FTP Data: 499 bytes (PASV) (NLSD)
782	795.676645	172.30.0.5	172.30.0.2	FTP-DATA	756	FTP Data: 702 bytes (PASV) (NLSD)
819	803.517671	172.30.0.2	172.30.0.5	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (STOR adminReport.pdf)
820	803.517671	172.30.0.2	172.30.0.5	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (STOR adminReport.pdf)
821	803.517671	172.30.0.2	172.30.0.5	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (STOR adminReport.pdf)
822	803.517671	172.30.0.2	172.30.0.5	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (STOR adminReport.pdf)
823	803.517671	172.30.0.2	172.30.0.5	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (STOR adminReport.pdf)
824	803.517671	172.30.0.2	172.30.0.5	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (STOR adminReport.pdf)
825	803.517671	172.30.0.2	172.30.0.5	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (STOR adminReport.pdf)
826	803.517671	172.30.0.2	172.30.0.5	FTP-DATA	1514	FTP Data: 1460 bytes (PASV) (STOR adminReport.pdf)

The detailed view of packet 819 shows the following information:

- [Header checksum status: Unverified]
- Source Address: 172.30.0.2
- Destination Address: 172.30.0.5
- Transmission Control Protocol, Src Port: 22585, Dst Port: 38144, Seq: 1, Ack: 1, Len: 1460
- Source Port: 22585
- Destination Port: 38144
- [Stream index: 16]
- [TCP Segment Len: 1460]
- Sequence Number: 1 (relative sequence number)
- Sequence Number (raw): 4270601298
- [Next Sequence Number: 1461 (relative sequence number)]
- Acknowledgment Number: 1 (relative ack number)
- Acknowledgment number (raw): 2519493485
- 0101 ... = Header Length: 20 bytes (5)
- Flags: 0x010 (ACK)
- Window: 32768
- [calculated window size: 4194304]
- [Window size scaling factor: 128]
- Checksum: 0x9a34 [Unverified]
- [Checksum Status: Unverified]
- Urgent Pointer: 0
- [SEQ/ACK analysis]
- [Timestamps]
- TCP payload (1460 bytes)
- FTP Data (1460 bytes data)
- [Setup frame: 814]
- [Setup method: PASV]
- [Command: STOR adminReport.pdf]
- [Command frame: 815]
- [Current working directory: /datadin]

The bottom status bar shows: Packets: 1525 · Displayed: 200 (13.1%) · Dropped: 0 (0.0%) · Profile: Default

Section 2: Applied Learning

Part 1:

Configure Wireshark and Generate Network Traffic

In this lab section, Wireshark is used on the vWorkstation to capture network traffic on an emulated WLAN. Later, in Part 2, the captured packets will be examined and compared. The following steps involve using PuTTY to remotely connect to the sta1 wireless station, and running Kali Linux. This connection enables scanning the airwaves from sta1 to confirm its presence within the broadcast range of the local access point.

On the vWorkstation, launch PuTTY, then open an SSH connection to sta1.securelabsondemand.local. Log in with the credentials: User: sta1, Password: password. Password input won't be visible. If a container stopped warning appears, reset the lab. Execute iwlist sta1-wlan0 scan at the command prompt to scan the airwaves for AP beacons using the wireless interface sta1-wlan0. Illustrated in Figure 24 below (Jones & Bartlett, 2024).

Figure 24

Iwlist output

```

# iwlist sta1-wlan0 scan
sta1-wlan0 Scan completed :
    Cell 01 - Address: 00:02:00:00:00:10
        Channel:1
        Frequency:2.412 GHz (Channel 1)
        Quality=70/70  Signal level=-36 dBm
        Encryption key:off
        ESSID:"SecureLabs-WiFi"
        Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 6 Mb/s
                   9 Mb/s; 12 Mb/s; 18 Mb/s
        Bit Rates:24 Mb/s; 36 Mb/s; 48 Mb/s; 54 Mb/s
        Mode:Master
        Extra:tsf=0005d244d650e3a2
        Extra: Last beacon: 24ms ago
        IE: Unknown: 000F5365637572654C6162732D57694669
        IE: Unknown: 010882848B960C121824
        IE: Unknown: 030101
        IE: Unknown: 2A0104
        IE: Unknown: 32043048606C
        IE: Unknown: 3B025100
        IE: Unknown: 7F080400400200000040

(root@sta1) ~[/]
#
```

Participant should Detect and identify a wireless network, "SecureLabs-WiFi," broadcasting on channel 1 with no WEP encryption. The network includes devices: Wireless Access Point (ap1) at 172.20.0.254/24, Wireless Station 1 (sta1) at 172.20.0.1/24, Wireless Station 2 (sta2) at 172.20.0.2/24, and Wireless Station 3 (sta3) at 172.20.0.3/24. Ap1 has no encryption, while sta2 and sta3 are connected, and sta1 is in broadcast range.

Execute a script to start a Wireshark capture on the WLAN using the Wireshark-Remote.vbs shortcut on the vWorkstation. Assume monitor mode for the vWorkstation's wireless NIC, capturing all airborne traffic in range. Beacons in Wireshark indicate AP presence; even if SSID is hidden, it's a weak security measure. Minimize Wireshark, restore the sta1/Kali PuTTY term.

- At the command prompt, execute `iwconfig` to review the station's current wireless network configuration. As illustrated below, in Figure 24 (Jones & Bartlett, 2024).

Figure 24

iwconfig output

```
(root@sta1)-[/]
# iwconfig
lo          no wireless extensions.

eth0        no wireless extensions.

sta1-wlan0  IEEE 802.11  ESSID:off/any
            Mode:Managed  Access Point: Not-Associated   Tx-Power=20 dBm
            Retry short limit:7   RTS thr:off   Fragment thr:off
            Encryption key:off
            Power Management:on

(root@sta1)-[/]
#
```

Run ``iwconfig sta1-wlan0 essid SecureLabs-WiFi`` to associate `sta1` with the SecureLabs-WiFi WLAN. The ``iwconfig`` command allows viewing and setting wireless interface parameters. After executing, check the output to ensure that `sta1-wlan0` is now associated with the access point, confirming a successful connection. Figure 25 shows this connection (Jones & Bartlett, 2024).

Figure 25

Associate sta1 with the WLAN titled SecureLabs-WiFi

A terminal window with a black background and white text. The prompt is (root@sta1)-[/]. The command # iwconfig sta1-wlan0 essid SecureLabs-WiFi is entered. The prompt returns to (root@sta1)-[/]. The command # is entered again, followed by a green cursor.

```
(root@sta1)-[/]  
# iwconfig sta1-wlan0 essid SecureLabs-WiFi  
(root@sta1)-[/]  
#
```

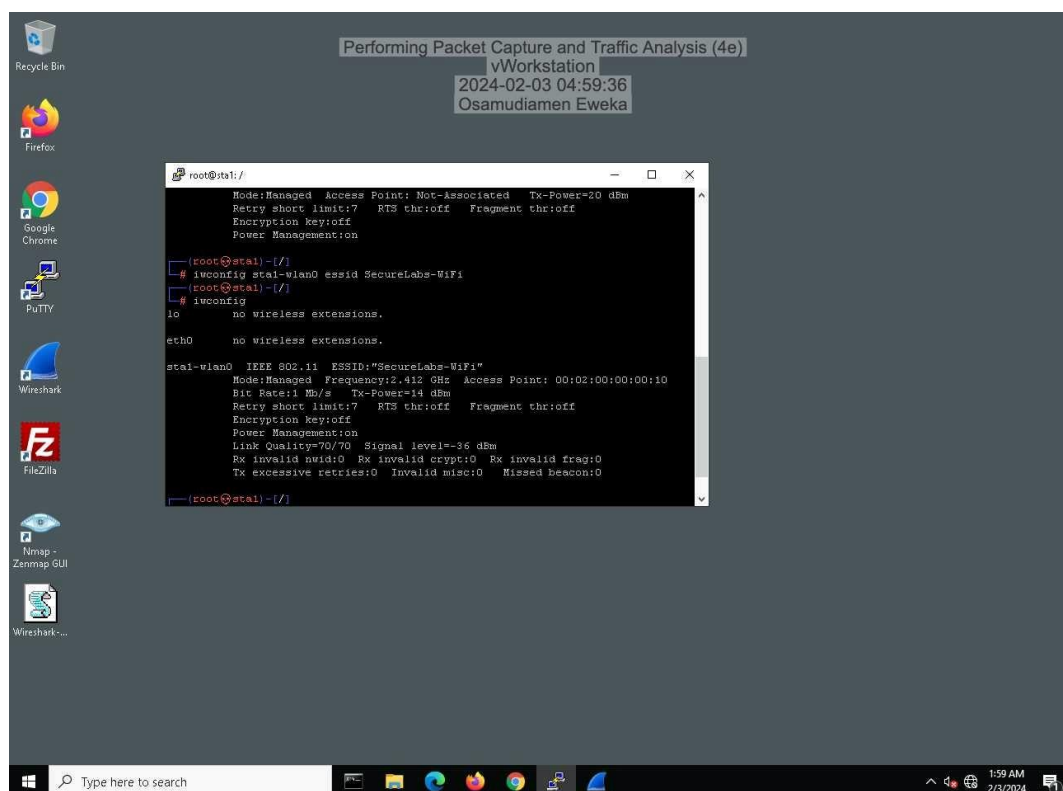
Run ``iwconfig`` at the command prompt to confirm that your station is now associated with the SecureLabs-WiFi WLAN. Check the output for information about the wireless interface and ensure that the association status with the access point is displayed as "Associated" for `sta1-wlan0`. To validate that `sta1-wlan0` is successfully connected to the SecureLabs-WiFi network, execute the ``iwconfig`` command at the prompt. Verify that the output displays "SecureLabs-WiFi" as the ESSID, the broadcast frequency is 2.412 GHz, and the access point's MAC address is 00:02:00:00:00:10. If the output does not match these criteria, retry the ``iwconfig`` command or return to previous steps to attempt reassociation, ensuring accurate command input. Capture a screenshot of the successful association output for reference.

Capture a screenshot of the command output after running `iwconfig` to showcase that the `sta1-wlan0` interface is successfully associated with the SecureLabs-WiFi network. Ensure the ESSID is displayed as "SecureLabs-WiFi," the broadcast frequency is 2.412 GHz, and the access

point's MAC address is 00:02:00:00:00:10 Figure 26 illustrates this screen capture (Eweka, 2024).

Figure 26

Make screen capture showing sta1-wlan0 connected to the SecureLabs-WiFi network.



Next

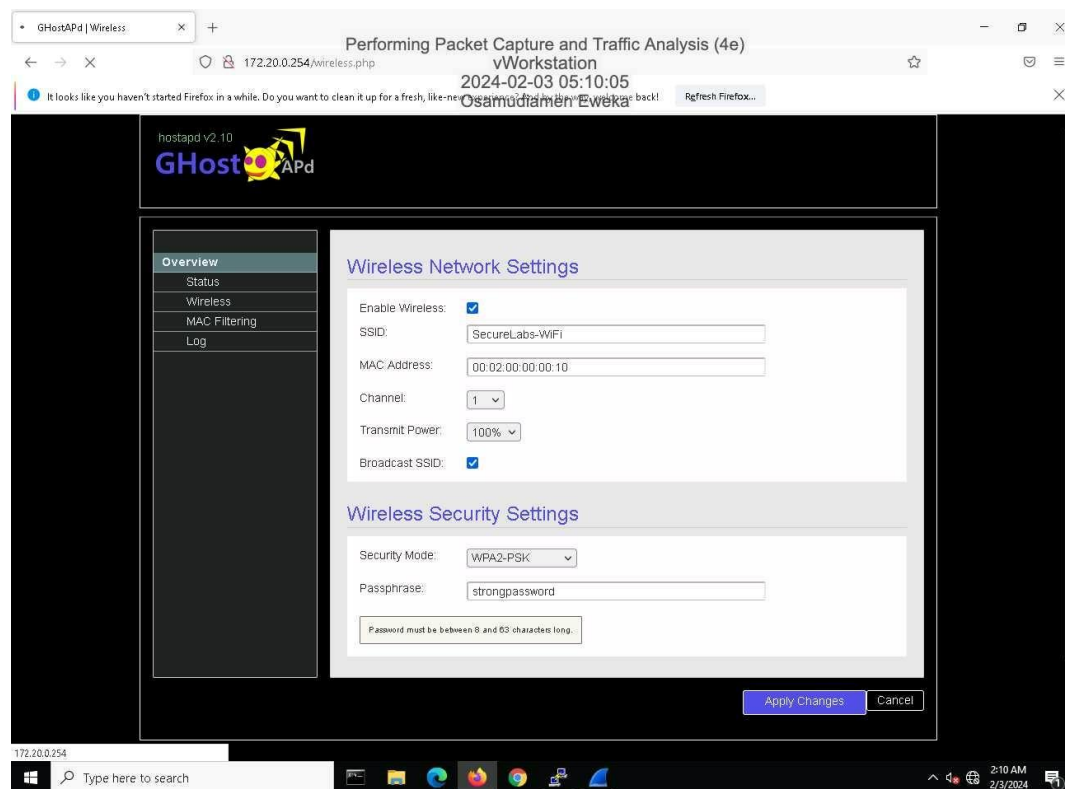
- Participants will generate traffic on the SecureLabs-WiFi network using sta1.
- At the command prompt, execute ping -c 4 172.20.0.2 to ping sta2.
- At the command prompt, execute curl http://172.20.0.254/ to generate HTTP traffic to ap1.

In the executed instructions, the process began with the generation of network traffic on the SecureLabs-WiFi network using sta1. A ping operation was performed to sta2, confirming successful communication through ICMP echo request and reply packets. Subsequently, HTTP

traffic was initiated to ap1, the wireless access point, ensuring the network's capability to handle HTTP requests. Following the traffic generation phase, attention shifted to enhancing wireless security on the ap1 access point. This involved accessing the ap1 management GUI through Firefox, navigating to the Wireless page, and configuring WPA2-PSK encryption. The security mode was updated, and a strong password, "strongpassword," was applied. These measures aimed to fortify the wireless network's security by safeguarding data in transit, providing an additional layer of protection against potential threats on the SecureLabs-WiFi network. Figure 27 shows a screen capture of updated security mode on the Status page (Eweka, 2024).

Figure 27

Screen capture of updated security mode on the Status page.



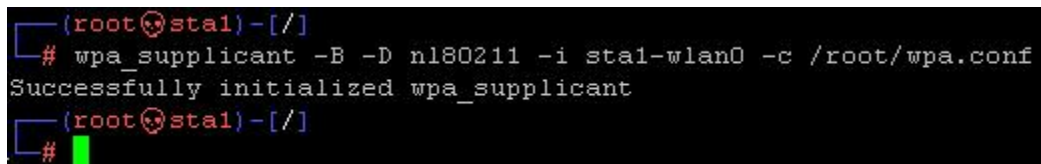
In the context of Linux shell connections to WPA2-enabled networks, iwconfig limitations led to the adoption of wpa_supplicant, a command-line WPA client. This supplicant handles login requests for wireless 802.11i-supported networks, ensuring continuous connection maintenance and reauthentication if disconnected. The process involves two steps: adding the WPA passphrase and connecting to the network. Executing "wpa_passphrase SecureLabs-WiFi > /root/wpa.conf" at the command prompt configures a WPA passphrase for the SecureLabs-WiFi network and stores the configuration in the wpa.conf file. The user inputs the passphrase, in this case, "strongpassword," enhancing security by facilitating secure access to the WPA2-protected network.

Linux shell, lacking iwconfig support for WPA2, utilizes wpa_supplicant for 802.11i networks. Executing "wpa_passphrase SecureLabs-WiFi > /root/wpa.conf" configures a secure passphrase, enhancing WPA2-protected network access with user-inputted "strongpassword."

wpa_supplicant converts ASCII key to PSK, recorded in wpa.conf. Executing "wpa_supplicant -B -D nl80211 -i sta1-wlan0 -c /root/wpa.conf" connects sta1-wlan0 to SecureLabs-WiFi using configured SSID and WPA passphrase. Shown in the figure 28 below (Jones & Bartlett, 2024).

Figure 28

Connects to the network.



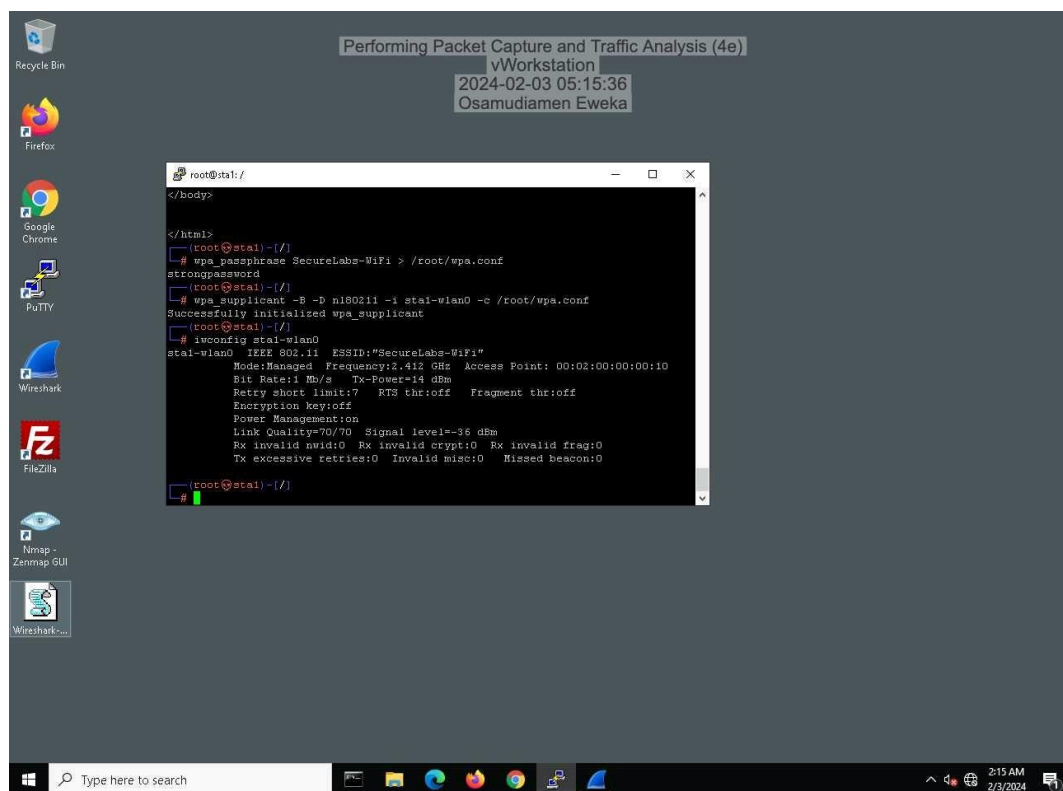
```
(root@sta1)-[/]
# wpa_supplicant -B -D nl80211 -i sta1-wlan0 -c /root/wpa.conf
Successfully initialized wpa_supplicant
(root@sta1)-[/]
#
```

The wpa_supplicant process is backgrounded (-B) to run independently. The -D option specifies nl80211 drivers for Linux softMAC. Executing "iwconfig sta1-wlan0" confirms reassociation with SecureLabs-WiFi. Although iwconfig can't configure WPA2 associations, it

checks interface status. Figure 29 (Eweka, 2024), Capture stal's connection to the encrypted WLAN and generate HTML traffic with "curl http://172.20.0.254". Close stal terminal, restore Wireshark, end the capture, and save it as "yourname_wificapture" on the vWorkstation desktop.

Figure 29

Make a screen capture showing the connection to the now-encrypted WLAN.



Section 2: Applied Learning

Part 2:

Analyze Traffic Using Wireshark

In this lab phase, participants engage in the analysis of packets generated on the SecureLabs-WiFi WLAN, focusing on the distinctive characteristics of wireless network transmissions. The Packet List Analysis begins with participants observing an abundance of packets labeled as "beacon frames" in the Info column. These frames, broadcast by Wireless Access Points (WAPs), serve the purpose of advertising the presence of the wireless network and contain vital information such as SSID, supported rates, channel utilization parameters, and a traffic indication map.

The subsequent step involves understanding the significance of beacon frames in wireless network communication. These frames, transmitted at regular intervals (every 100ms in the lab network), play a crucial role in facilitating devices and software adhering to 802.11 standards to establish wireless communication at the Link Layer (TCP/IP Model) or Data Link Layer (OSI Model).

Further exploration takes participants into the anatomy of a broadcast beacon by selecting any beacon frame from NetSys_00:00:10 in the Packet List pane. The focus shifts to the Packet Details pane, where a new wireless stack unfolds with four Protocol Fields/rows: Radiotap, 802.11 radio information, 802.11 Beacon frame, and 802.11 Wireless Management data. Attention is directed towards expanding the IEEE 802.11 Beacon frame row, followed by expanding the Frame Control Field row. This step delves into the specific structure of the beacon

frame, providing insights into the physical layer implementations and the current configuration of the wireless network.

The overarching goal of this process is to familiarize participants with the intricacies of wireless communication captured in beacon frames. By delving into the protocols and configurations governing the wireless network, participants gain a comprehensive understanding of the underlying mechanisms at play.

In the Packet Details pane, expand the IEEE 802.11 Beacon frame row, then expand the Frame Control Field row. Illustrate in figure 30 (Jones & Bartlett, 2024).

Figure 30

Frame control field

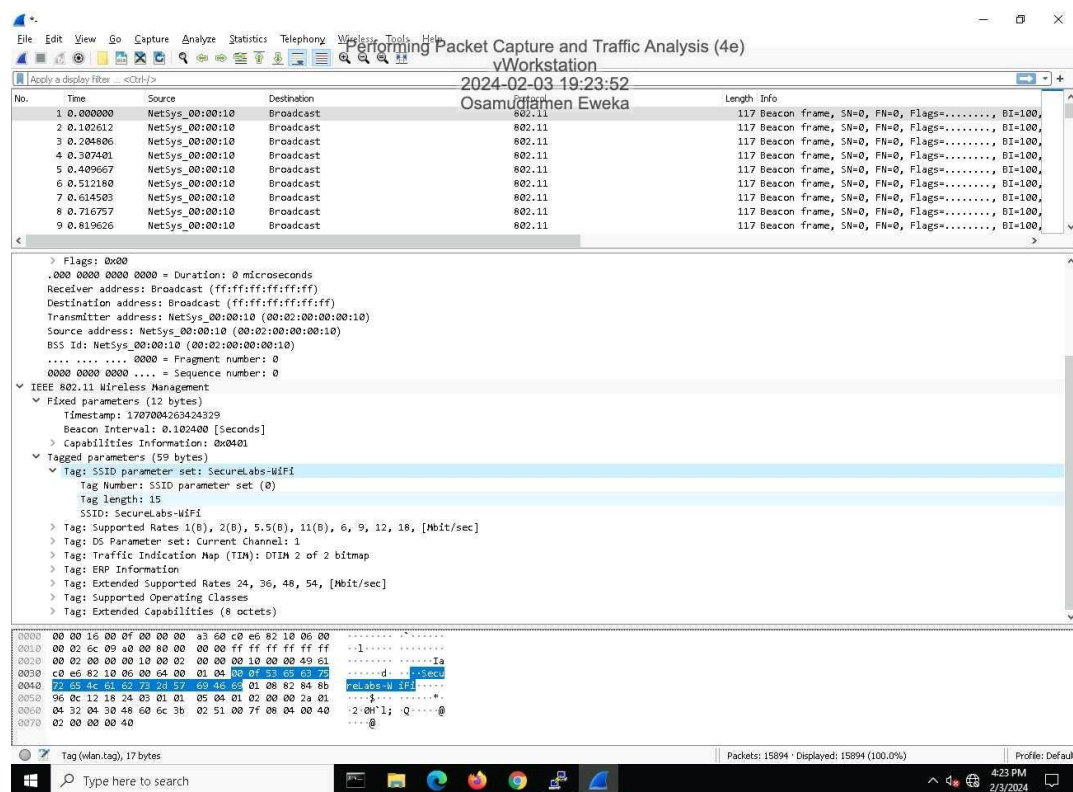
```
> Frame 4447: 139 bytes on wire (1112 bits), 139 bytes captured (1112 bits) on interface -, id 0
> Radiotap Header v0, Length 22
> 802.11 radio information
▼ IEEE 802.11 Beacon frame, Flags: .....
  Type/Subtype: Beacon frame (0x0008)
  ▼ Frame Control Field: 0x8000
    .... ..00 = Version: 0
    .... 00.. = Type: Management frame (0)
    1000 .... = Subtype: 8
    > Flags: 0x00
    .000 0000 0000 0000 = Duration: 0 microseconds
    Receiver address: Broadcast (ff:ff:ff:ff:ff:ff)
    Destination address: Broadcast (ff:ff:ff:ff:ff:ff)
    Transmitter address: NetSys_00:00:10 (00:02:00:00:00:10)
    Source address: NetSys_00:00:10 (00:02:00:00:00:10)
    BSS Id: NetSys_00:00:10 (00:02:00:00:00:10)
    .... .... 0000 = Fragment number: 0
    0000 0000 0000 .... = Sequence number: 0
  > IEEE 802.11 Wireless Management
```

Observing the beacon frame in Wireshark, we note that it falls under the Management frame type, one of the three 802.11 frame types. The beacon interval is set at 100ms, as previously mentioned. Expanding the Tagged Parameters row, we find that the SSID parameter is "SecureLabs-WiFi," serving as the identifier for the wireless network.

In the DS parameter set, the current channel is indicated as 1. Channels, in the context of wireless networks, help segregate frequency ranges for communication. This channelization allows multiple wireless networks to coexist with minimal interference. Overall, the beacon frame provides essential information about the wireless network, including its identifier, channel configuration, and beacon interval, contributing to a comprehensive understanding of the network's characteristics. Figure 31 show the screen capture of SSID and channel in the Packet Details pane (Eweka, 2024)

Figure 31

Screen capture of SSID and channel in the Packet Details pane.

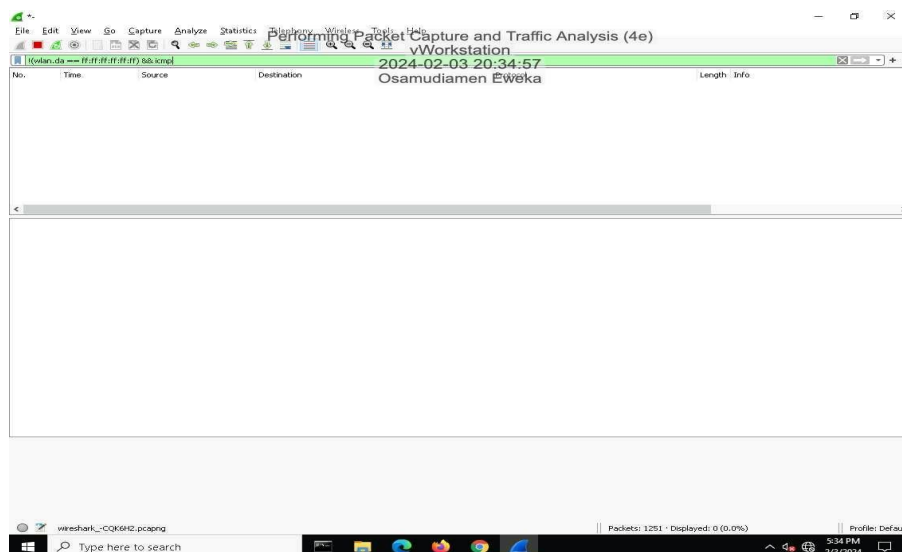


In Wireshark, a broadcast packet filter is applied to exclude all broadcast packets, enhancing focus on specific packet types. Selecting a packet with "Acknowledgement" in the Info column, preceded by a "Probe Request," unveils a notable aspect of wireless devices. When not connected to a network, devices broadcast probe packets containing previously associated SSIDs, attempting to connect transparently to known networks. Upon receiving an acknowledgment, the device endeavors to establish a connection with the acknowledged wireless network.

Inspecting the Frame Control Field row in the Packet Details pane reveals a Control frame, used for medium access and acknowledgment. Introducing a display filter with "&& icmp" isolates packets related to ping requests. Reviewing the Frame Control Field row for these filtered packets exposes Data frames. In the 802.11 frame hierarchy, Data frames facilitate the transfer of information, providing insights into the dynamics of wireless communication and the exchange of data in the observed network. Figure 32 show the screen shoot of the packet details for the ICMP packet (Eweka, 2024)

Figure 32

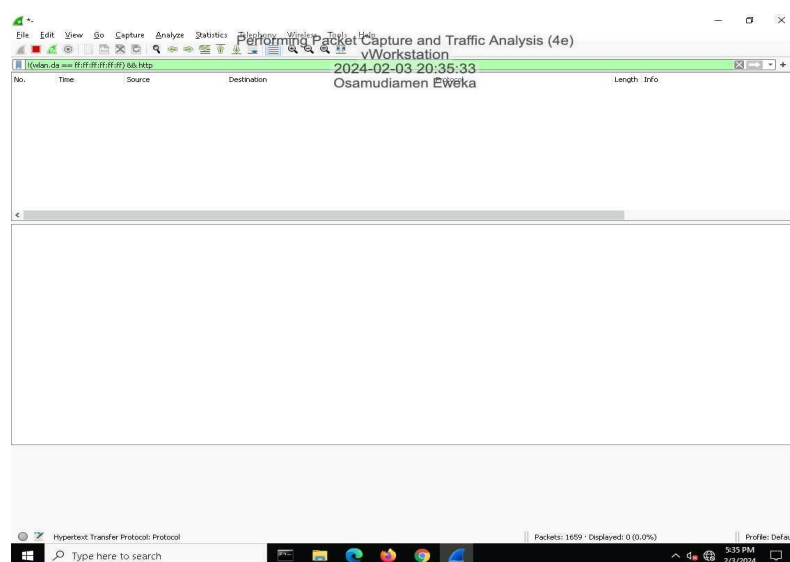
Make a screen capture showing the Packet Details for the ICMP packet.



A new display filter in Wireshark isolates HTTP protocol traffic, revealing two packets related to the initial curl command. The Packet List pane allows selection of any HTTP packet, showcasing the OSI Data Link Layer's composition and highlighting familiar fields in IP, TCP, and HTTP communications. The screen capture of Packet Details for the HTTP packet captures the encapsulation process and protocol fields in wireless communication in Figure 33 (Eweka, 2024).

Figure 33

Screen capture of Packet Details for the HTTP packet.

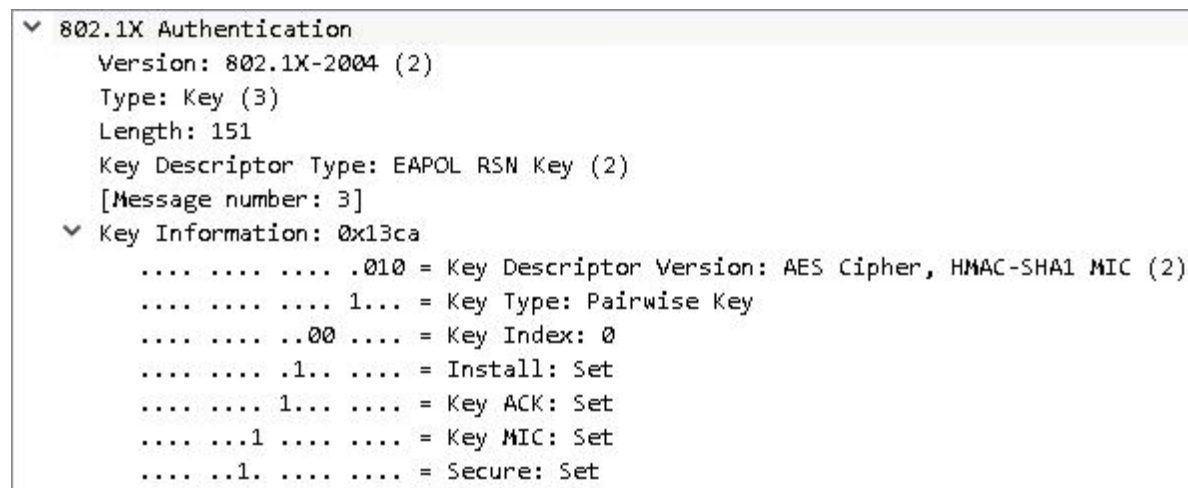


Applying a new display filter in Wireshark isolates EAPOL protocol traffic, exposing packets related to WPA2 encryption authentication. EAPOL establishes trust between a WAP and a station through a four-way handshake. The process involves the exchange of random numbers, generation of temporary encryption keys, and verification messages. The packet with "Key (Message 3 of 4)" in the Packet List pane represents the critical step where the WAP sends the encryption key to the station, marking a key phase in securing the connection. The

subsequent expansion of the 802.11X Authentication row and Key Information row in the Packet Details pane reveals insights into the authentication process defined by the IEEE 802.11X standard. Shown in figure 34 (Jones & Bartlett, 2024).

Figure 34

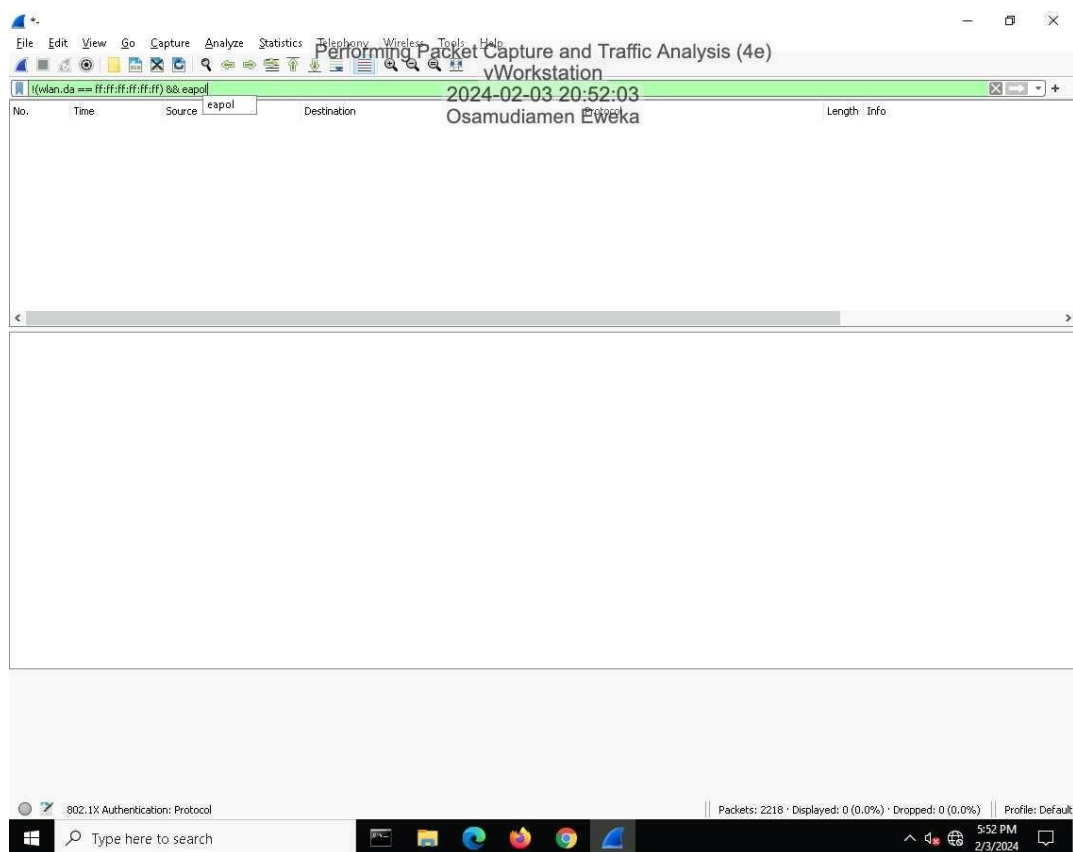
Key information



Notice the AES cipher and HMAC info at the top. These are cryptographic algorithms used for message encryption and authentication, respectively. Pay attention to the bits that are set. In Message 3, bits are set for Install, Key ACK, Key MIC, and encrypted key data. This particular frame is an important one in the setup process, and was actually taken advantage of in the KRACK vulnerability, because it allowed key to be resent over and over. Below in Figure 35 shows the key information of message 3 in the four-way handshake (Eweka, 2024)

Figure 35

Make a screen capture showing the key information for Message 3 in the four-way handshake.



Section 3: Challenge and Analysis

Part 1:

Generate Malicious Network Traffic

Step 1: Navigate to the C:\ directory on the vWorkstation and launch the Section3_WLAN.cmd file. This script prepares the lab environment by initiating a Wireshark capture session on a new WLAN topology.

Step 2: Utilize the Airodump-ng module and Aireplay-ng module from the Aircrack-ng suite to simulate malicious network traffic on the WLAN. Airodump-ng functions as a packet sniffer, while Aireplay-ng generates wireless traffic. Security professionals must identify signs of unusual and potentially malicious network activity, making these tools crucial for assessing wireless security.

By executing these steps, you ensure that the lab environment is set up for Section 3, where you simulate malicious traffic to understand and recognize potential security threats.

Note: In this lab exercise, participants may encounter errors with aireplay-ng commands stem from incorrect syntax and the system's inability to locate files or directories named "bssid" and "bssid." The solution involves replacing placeholders like <bssid> and <00:02:00:00:00:12> with the actual MAC addresses of the target access point and station.

Example Syntax:

```
aireplay-ng --deauth 10 -a 00:02:00:00:00:10 -c 00:02:00:00:00:12 wlan0
```

Ensure to substitute "00:02:00:00:00:10" with the target access point's MAC address and "00:02:00:00:00:12" with the station's MAC address. Also, replace "wlan0" with the correct interface name for your setup.

Verify that you've accurately gathered MAC addresses from your network scan, and the specified interface is available. Double-check the syntax, incorporating actual values, and rerun the command.

By addressing these issues and using the correct syntax with real MAC addresses and interface names, you should successfully resolve errors and execute the aireplay-ng command in your lab exercise. Participants are to make screenshots as shown in Figure 36 below (Eweka, 2024).

Figure 36

Make a screen capture showing the aireplay-ng --deauth output.

```

root@stali: /
CH 1 ][ Elapsed: 1 min ][ 2024-02-04 03:03:37
BSSID      PWR RXQ Beacons  #Data, #/s CH  MP  Prio  Prio  Prio  Prio
00:02:00:50:00:3E -34 100   755      0  0  1  50  WPA2 CCMP PSK Sec3-WiFi
00:02:00:00:00:10 -34  0    755      7  0  1  54  WPA2 CCMP PSK Sec3-WiFi

BSSID      STATION      PWR  Rate  Lost  Frames  Notes  Probes
00:02:00:00:00:10 00:02:00:00:00:12 -35  0 -54    0      3      Sec3-WiFi
00:02:00:00:00:10 00:02:00:00:00:13 -35  0 -24    0      3      Sec3-WiFi
Quitting...

(root@stali) - [/]
# aireplay-ng --deauth 10 -a 00:02:00:00:00:10 -c 00:02:00:00:00:12 stali-wlan0 --lg
03:07:11 Waiting for beacon frame (BSSID: 00:02:00:00:00:10) on channel 1
03:07:12 Sending 64 directed DeAuth (code 7). STMAC: [00:02:00:00:00:12] [ 0] 0 ACKs]
03:07:12 Sending 64 directed DeAuth (code 7). STMAC: [00:02:00:00:00:12] [ 0] 0 ACKs]
03:07:12 Sending 64 directed DeAuth (code 7). STMAC: [00:02:00:00:00:12] [ 0] 0 ACKs]
03:07:13 Sending 64 directed DeAuth (code 7). STMAC: [00:02:00:00:00:12] [ 0] 0 ACKs]
03:07:13 Sending 64 directed DeAuth (code 7). STMAC: [00:02:00:00:00:12] [ 0] 0 ACKs]
03:07:14 Sending 64 directed DeAuth (code 7). STMAC: [00:02:00:00:00:12] [ 0] 0 ACKs]
03:07:14 Sending 64 directed DeAuth (code 7). STMAC: [00:02:00:00:00:12] [ 0] 0 ACKs]
03:07:15 Sending 64 directed DeAuth (code 7). STMAC: [00:02:00:00:00:12] [ 0] 0 ACKs]
03:07:15 Sending 64 directed DeAuth (code 7). STMAC: [00:02:00:00:00:12] [ 0] 0 ACKs]
03:07:16 Sending 64 directed DeAuth (code 7). STMAC: [00:02:00:00:00:12] [ 0] 0 ACKs]

(root@stali) - [/]
#

```

Section 3: Challenge and Analysis

Part 2:

Analyze Malicious Network Traffic

In Part 1, the simulated attack aimed to disrupt access or potentially capture EAPOL data for a passphrase crack. As a security analyst, Wireshark becomes crucial for packet-level analysis of such attacks. Follow these steps:

Apply a Display Filter to Hide Broadcast Packets and Find De-authentication Packets

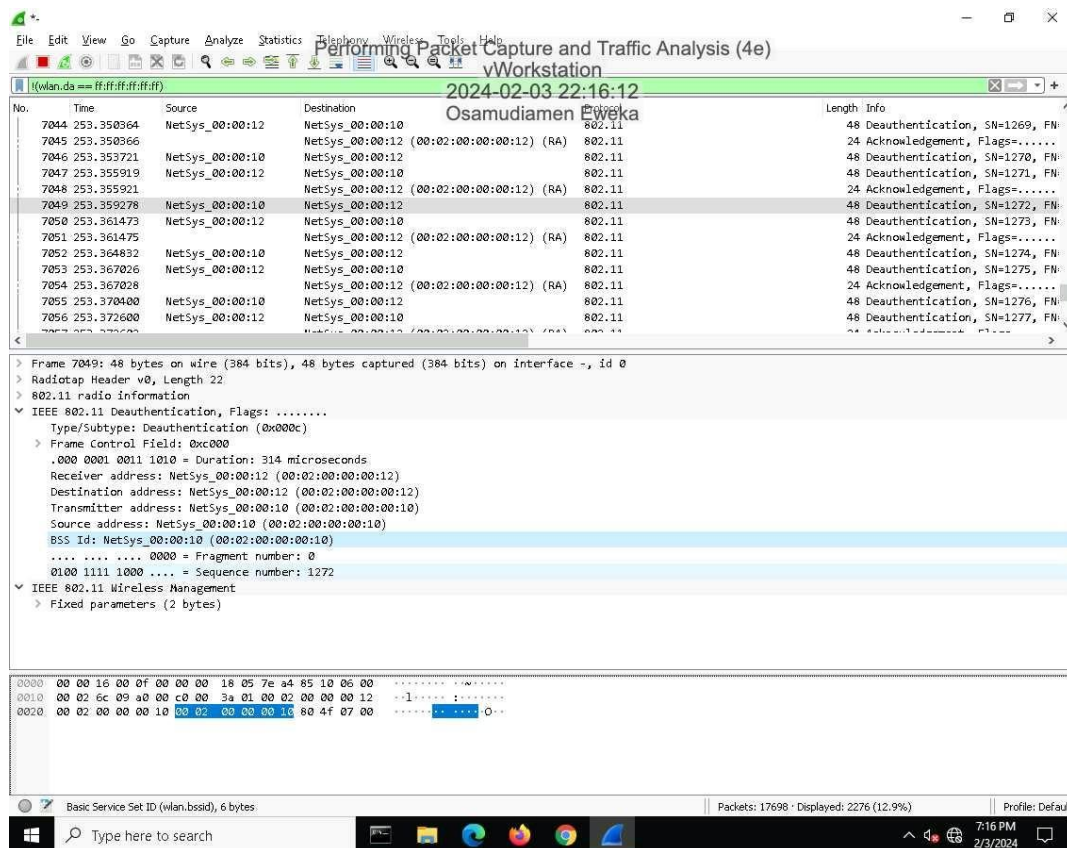
Let's simplify the steps:

1. Open Wireshark and load the captured packet trace file from Part 1 or begin capturing packets if not done already.
2. Apply a display filter to hide broadcast packets. Use "not broadcast" as the filter.
3. Search for deauthentication packets in the Packet List pane.
4. Identify deauthentication packets marked as "802.11" in the protocol column.

These steps help focus on deauthentication packets, aiding in the analysis of potential security threats. Figure 37 (Eweka, 2024) shows screen capture showing one of the deauth packets that you generated between the BSSID and your selected station.

Figure 37

Make a screen capture showing one of the deauth packets that you generated between the BSSID and your selected station.



Next Isolate Packets Related to the Four-Way Handshake: To isolate packets related to the four-way handshake that occurred when the client station reauthenticated to the access point, follow these steps:

1. Modify your display filter to focus on packets related to the four-way handshake. The four-way handshake is part of the WPA/WPA2 authentication process. Use the filter EAPOL to isolate these packets:

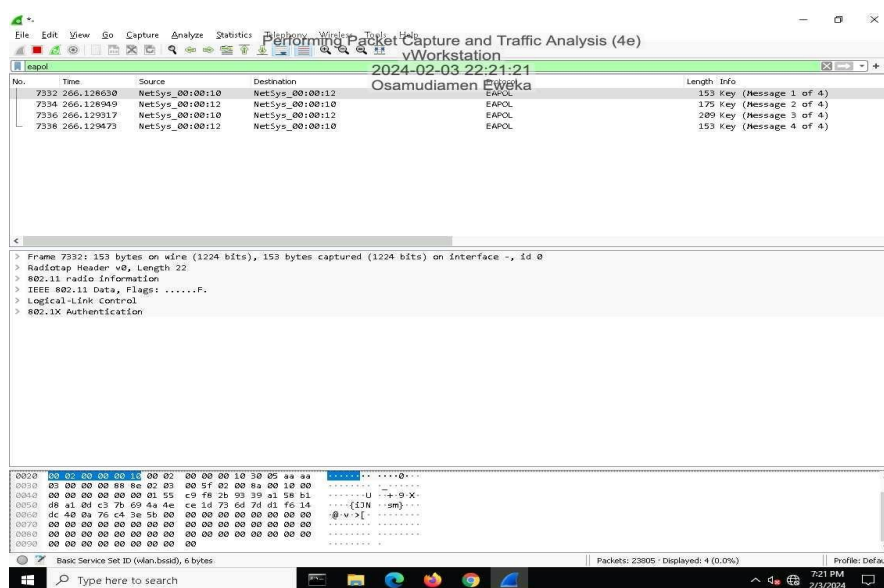
This filter will display only EAPOL (Extensible Authentication Protocol over LAN) packets, which are used in the WPA/WPA2 handshake.

2. After applying the filter, you should see packets related to the four-way handshake between the client station and the access point. The four-way handshake typically consists of the following four steps: 1) Request, 2) Response, 3) Confirm, and 4) Confirm.
3. Examine the packet details to ensure that the four-way handshake packets are present and correctly ordered.

Analyzing the four-way handshake packets can help you understand the authentication process and ensure the security of the wireless network. If you find any suspicious or unexpected behavior during the analysis, it may indicate a security concern, and further investigation may be necessary. Figure 38 make a screen capture the packets related to the four-way handshake (Eweka, 2024).

Figure 38

Make a screen capture showing the packets related to the four-way handshake.



Conclusion

This report on "Performing Packet Capture and Traffic Analysis" provides a comprehensive analysis of network security through practical exercises using Wireshark. It covers configuring Wireshark, generating and analyzing both regular and malicious network traffic, and understanding security mechanisms within network communications. The lab exercises demonstrated real-world applications of packet capture techniques for identifying vulnerabilities and securing network traffic. This hands-on approach enhances understanding of network protocols, encryption methods, and security practices essential for protecting against cyber threats. Through detailed analyses and screen captures, the report underscores the importance of continuous monitoring and analysis for maintaining network integrity and security.

References

Eweka O. (2024). Performing Packet Capture and Traffic Analysis (Figure 12).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 15).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 18).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 19).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 21).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 22).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 26).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 27).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 29).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 30).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 31).

- Jones and Bartlett Learning Virtual Lab.* URL: <https://jbl-lti.hatsize.com/startlab>
- Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 32).
- Jones and Bartlett Learning Virtual Lab.* URL: <https://jbl-lti.hatsize.com/startlab>
- Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 33).
- Jones and Bartlett Learning Virtual Lab.* URL: <https://jbl-lti.hatsize.com/startlab>
- Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 35).
- Jones and Bartlett Learning Virtual Lab.*URL: <https://jbl-lti.hatsize.com/startlab>
- Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 36).
- Jones and Bartlett Learning Virtual Lab.* URL: <https://jbl-lti.hatsize.com/startlab>
- Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 37).
- Jones and Bartlett Learning Virtual Lab.* URL: <https://jbl-lti.hatsize.com/startlab>
- Eweka O.(2024). Performing Packet Capture and Traffic Analysis (Figure 38).
- Jones and Bartlett Learning Virtual Lab.* URL: <https://jbl-lti.hatsize.com/startlab>
- Hofstede, R., Celeda, P., Trammell, B., Drago, I., Sadre, R., Sperotto, A., & Pras, A. (2014). Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX. *IEEE Communications Surveys & Tutorials*, 16(4), 2037–2064.
<https://doi.org/10.1109/comst.2014.2321898>
- Imperva. (2021). *What Is OSI Model | 7 Layers Explained*. Imperva.
<https://www.imperva.com/learn/application-security/osi-model/>
- Jones K. & Bartlett S. (2024). Performing Packet Capture and Traffic Analysis (Figure 12).
- Jones and Bartlett Learning Virtual Lab.* URL: <https://jbl-lti.hatsize.com/startlab>
- Jones K. & Bartlett S. (2024). Performing Packet Capture and Traffic Analysis (Figure 13).
- Jones and Bartlett Learning Virtual Lab.*URL: <https://jbl-lti.hatsize.com/startlab>

Jones K. & Bartlett S. (2024). Performing Packet Capture and Traffic Analysis (Figure 14).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Jones K. & Bartlett S. (2024). Performing Packet Capture and Traffic Analysis (Figure 16).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Jones K. & Bartlett S. (2024). Performing Packet Capture and Traffic Analysis (Figure 17).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Jones K. & Bartlett S. (2024). Performing Packet Capture and Traffic Analysis (Figure 20).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Jones K. & Bartlett S. (2024). Performing Packet Capture and Traffic Analysis (Figure 25).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Jones K. & Bartlett S. (2024). Performing Packet Capture and Traffic Analysis (Figure 28).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Jones K. & Bartlett S. (2024). Performing Packet Capture and Traffic Analysis (Figure 30).

Jones and Bartlett Learning Virtual Lab. URL: <https://jbl-lti.hatsize.com/startlab>

Jones K. & Bartlett S. (2024). Performing Packet Capture and Traffic Analysis (Figure 34).

Jones and Bartlett Learning Virtual Lab.

URL: <https://jbl-lti.hatsize.com/startlab>

Lotfollahi, M., Jafari Siavoshani, M., Shirali Hossein Zade, R., & Saberian, M. (2019). Deep packet: a novel approach for encrypted traffic classification using deep learning. *Soft*

Computing. <https://doi.org/10.1007/s00500-019-04030-2>

Petters, J. (2019, August 29). *How to Use Wireshark: Comprehensive Tutorial + Tips* | Varonis.

Inside out Security. <https://www.varonis.com/blog/how-to-use-wireshark>