COURSE: CLOUD AND NETWORK SECURITY

NAME: DENISE SOPHY ONDISO MUTAYI

STUDENT NO: CS-CN09-25047

NETWORK TRAFFIC ANALYSIS

Contents

INTRODUCTION

The network traffic analysis lab provides a practical environment to develop essential skills in dissecting and interpreting captured network data. Utilizing Wireshark, a leading network protocol analyzer, this lab focuses on analyzing HTTP and FTP traffic within packet capture (pcap) files and live network sessions.

Through a series of tasks including filtering traffic, following TCP streams, and extracting embedded objects such as images, the lab enables participants to identify suspicious activity, such as data exfiltration attempts hidden within legitimate network communications. This hands-on experience bridges theoretical knowledge with real-world application, enhancing understanding of protocol behavior, network conversations, and forensic investigation techniques.

NETWORKING BASICS: LAYERS 1 TO 4

When we talk about networking, it helps to think of communication as a structured process, with different “layers” handling specific tasks to make sure data travels smoothly from one device to another. The two key models that guide us here are the OSI model and the TCP/IP model.

- The OSI model divides the communication process into 7 distinct layers, from the physical cables up to the software applications.
- The TCP/IP model simplifies this into 4 layers by combining some of the OSI functions, reflecting how networks operate in the real world.

The first four layers — from physical transmission to reliable data transfer — are crucial for understanding how data physically moves through the network and how errors are detected and corrected.

Protocol Data Units (PDU)

At each layer of the networking model, data is wrapped inside a “protocol data unit,” or PDU. Think of it as a package that’s labeled and prepared for the next step in its journey. When analyzing network traffic with tools like Wireshark, you’re essentially seeing these PDUs as they travel through the layers.

Addressing: MAC and IP Addresses

- MAC Address: A hardware-level address unique to every network interface card. It’s how devices identify each other within the same local network.
- IP Address: A logical address used to identify devices across networks, allowing data to be routed beyond local boundaries. IPv4 is still common, but IPv6 is gaining traction due to its vastly larger address space.

IPv6

IPv6 was created to address the limitations of IPv4, mainly the shortage of available addresses. With its longer hexadecimal format and improved features like built-in security, IPv6 is gradually becoming the future of networking.

Transport Layer: TCP vs UDP

At this layer, the focus is on how data is delivered:

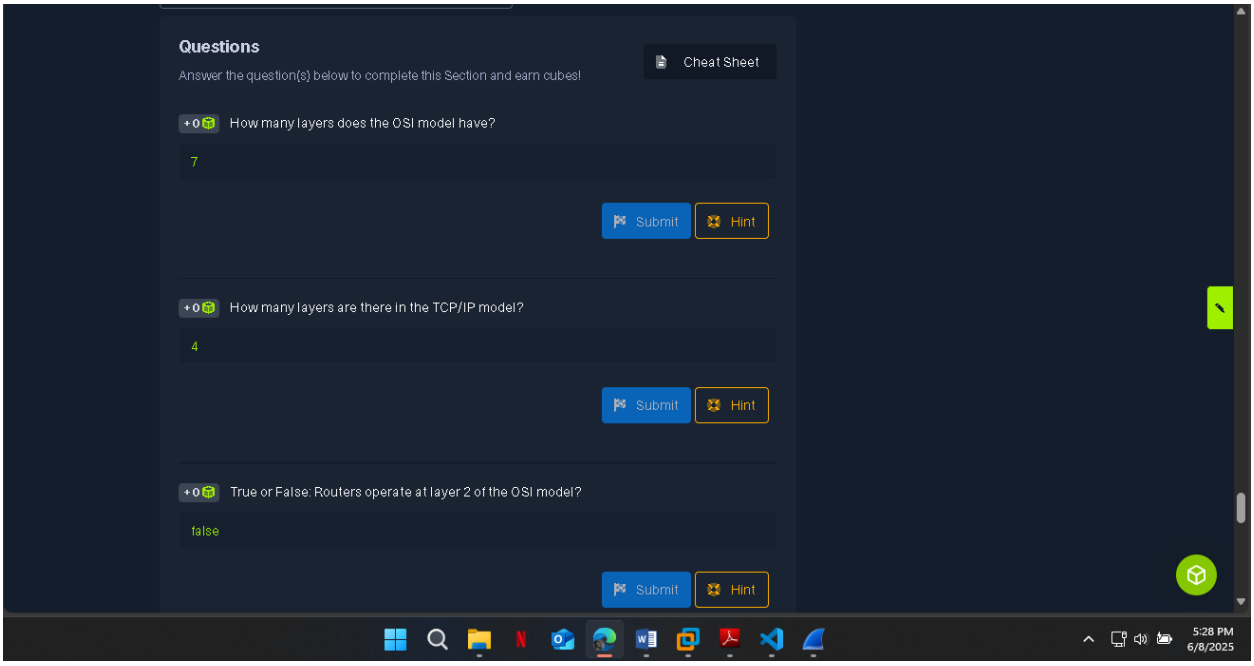
- TCP (Transmission Control Protocol): Ensures reliable, ordered delivery with error checking. It uses a “three-way handshake” to establish a connection before data transfer, making it ideal for web browsing, emails, and file transfers.
- UDP (User Datagram Protocol): A connectionless protocol that sends data without guaranteeing delivery or order, prioritizing speed over reliability. It’s often used in streaming, gaming, and DNS queries.

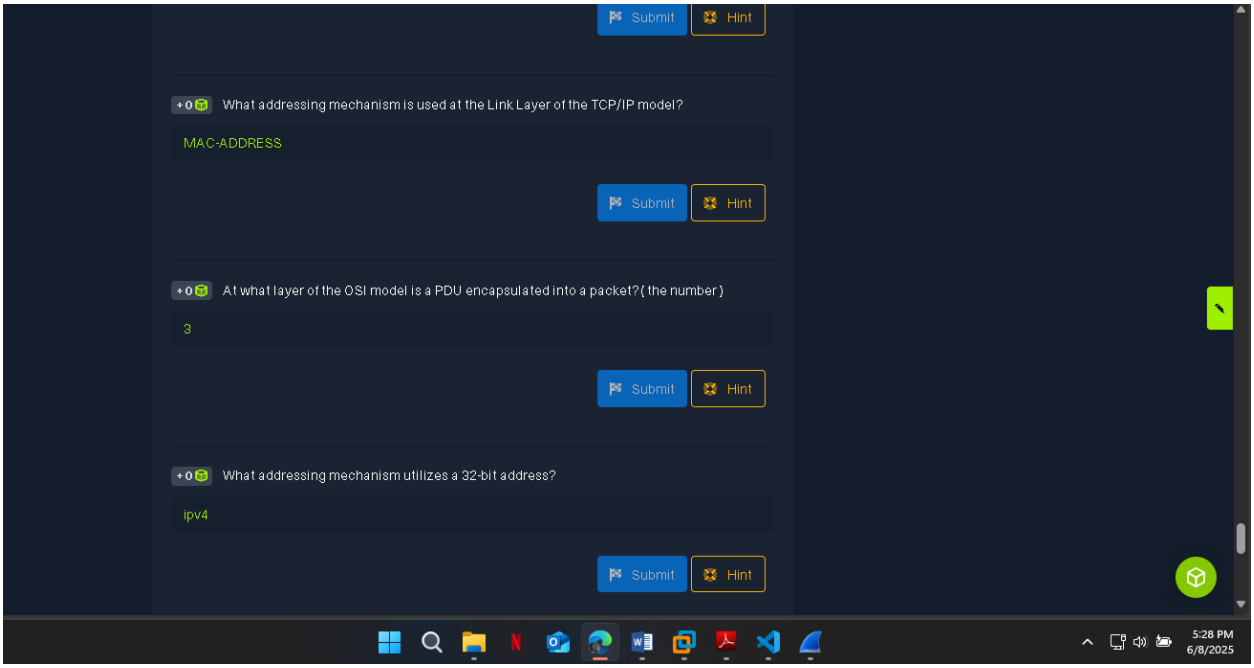
TCP’s Three-Way Handshake

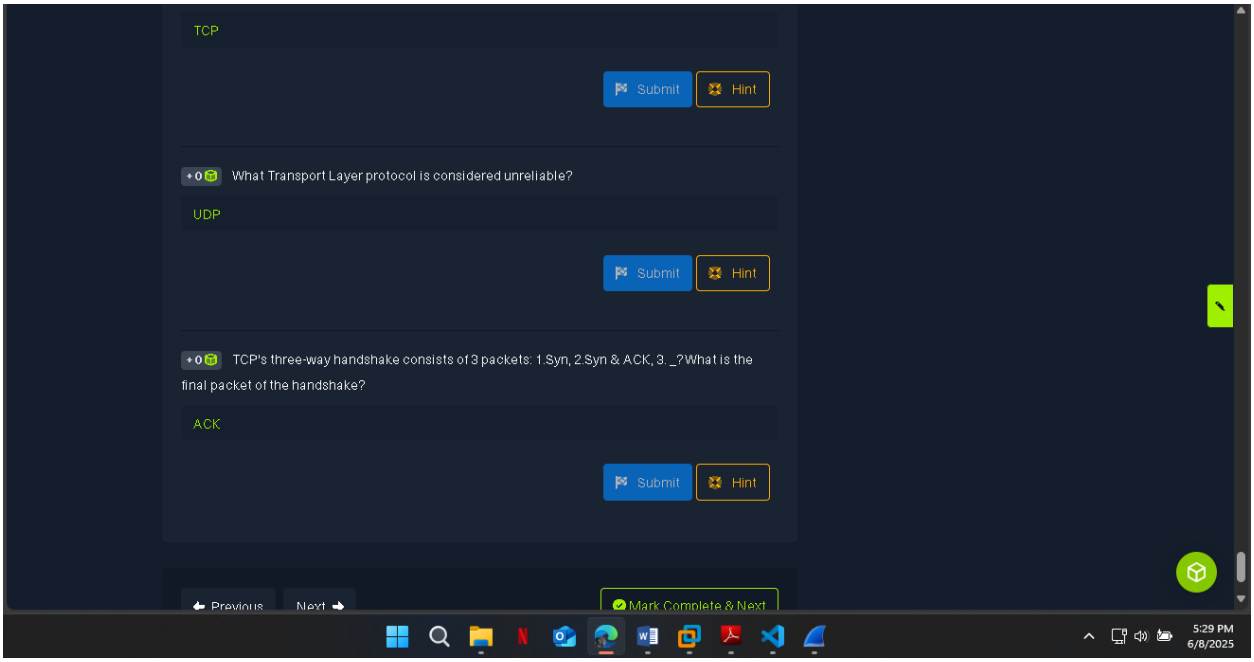
To establish a connection, TCP uses a simple, three-step process:

1. SYN: Client requests to start a connection.
2. SYN-ACK: Server acknowledges and agrees.
3. ACK: Client confirms, and the connection is set.

This handshake ensures both sides are ready before data is exchanged.




Submit
Hint
+0 What addressing mechanism is used at the Link Layer of the TCP/IP model?
MAC-ADDRESS
Submit
Hint
+0 At what layer of the OSI model is a PDU encapsulated into a packet?(the number)
3
Submit
Hint
+0 What addressing mechanism utilizes a 32-bit address?
ipv4
Submit
Hint
5:28 PM
6/8/2025


TCP
Submit
Hint
+0 What Transport Layer protocol is considered unreliable?
UDP
Submit
Hint
+0 TCP's three-way handshake consists of 3 packets: 1.Syn, 2.Syn & ACK, 3. _?What is the final packet of the handshake?
ACK
Submit
Hint
Previous
Next
Mark Complete & Next
5:29 PM
6/8/2025

NETWORKING PRIMER: UPPER LAYERS (LAYERS 5-7)

While the lower layers manage the movement of data packets across networks, the upper layers focus on how applications communicate and exchange data over those networks. Key protocols at these layers enable essential services like web browsing, secure communication, file transfers, and resource sharing.

HTTP (Hypertext Transfer Protocol)

- A stateless protocol used since 1990 for transferring web content between clients (browsers) and servers.
- Operates over TCP, typically on ports 80 or 8000.
- Uses various **methods** to define client actions:
    - **GET:** Retrieve content.
    - **HEAD:** Like GET but fetches only headers.
    - **POST:** Submit data (e.g., form submissions).
    - **PUT:** Create or update a resource.
    - **DELETE:** Remove a resource.
    - **OPTIONS:** Query supported methods.
    - **TRACE:** Diagnostic echo of the request.
    - **CONNECT:** Used for proxy tunnels (like SSL).
- GET and HEAD are mandatory for HTTP servers; others are optional depending on server configuration.

HTTPS (HTTP Secure)

- Secures HTTP using **TLS** (Transport Layer Security), encrypting all traffic between client and server to prevent eavesdropping or tampering.
- Runs on TCP ports 443 or 8443.
- Establishes a secure session via a **TLS handshake** that:
    - Negotiates cryptographic parameters.
    - Authenticates the server using X.509 certificates.
    - Generates shared secrets to encrypt data.
- Enables safe transactions for banking, messaging, and private data over the web.

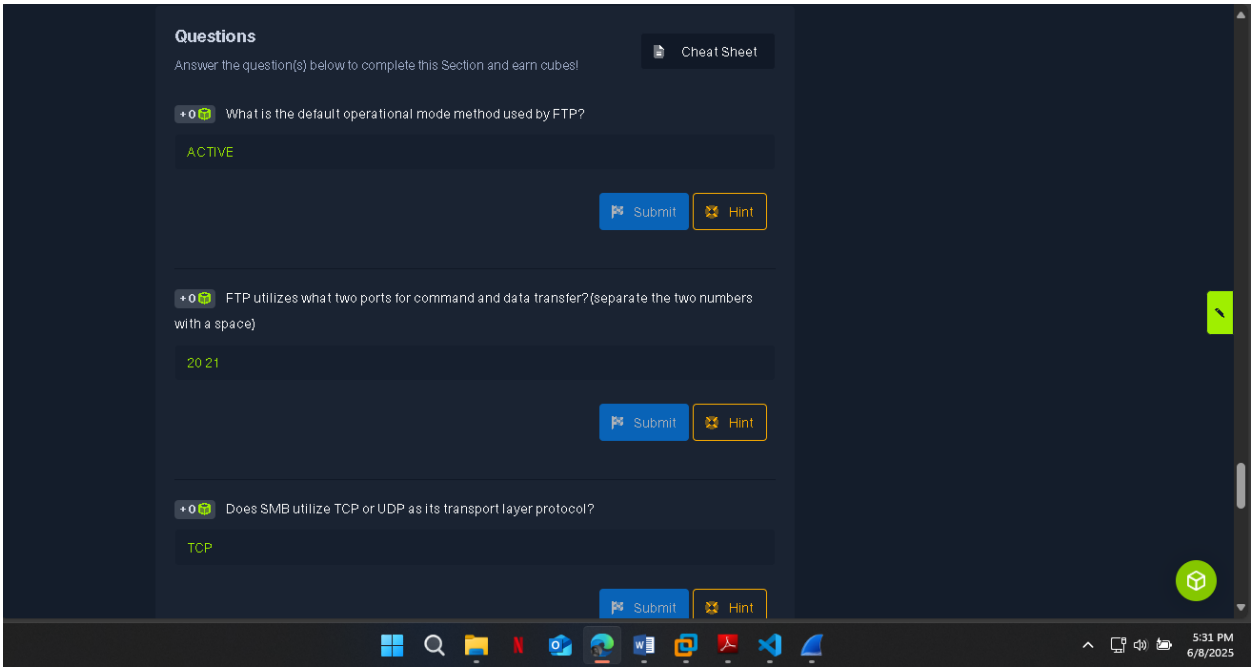
FTP (File Transfer Protocol)

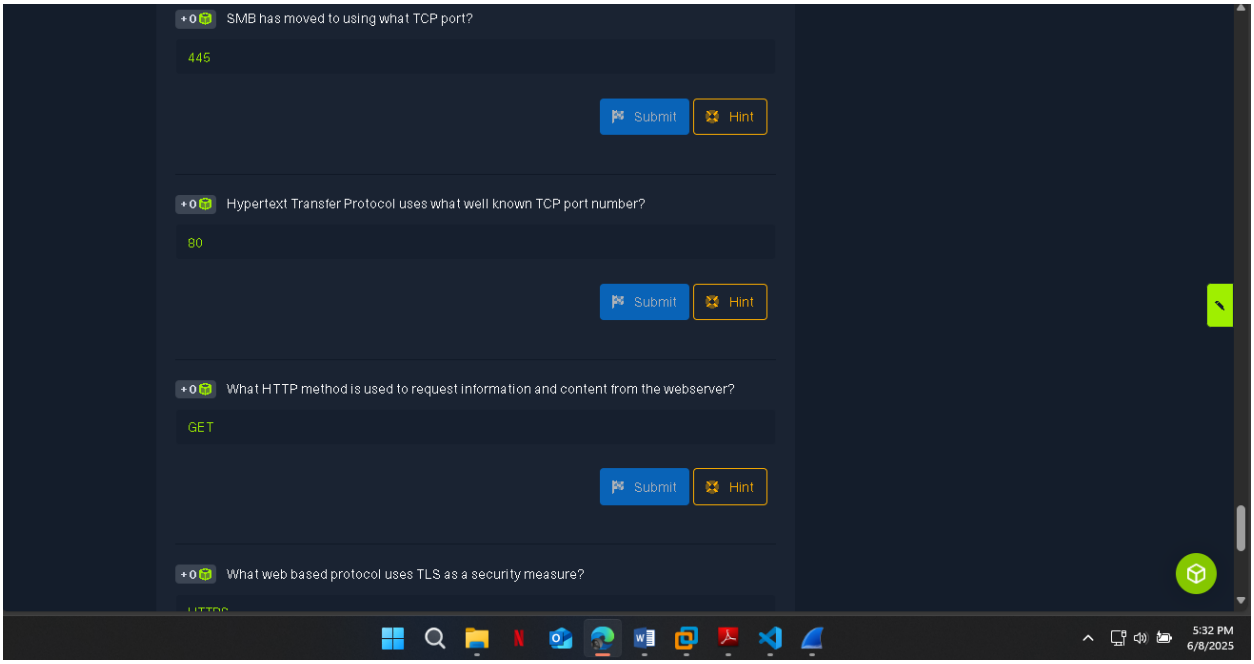
- An older protocol for transferring files over TCP, using two ports:
    - Port 21 for control commands.

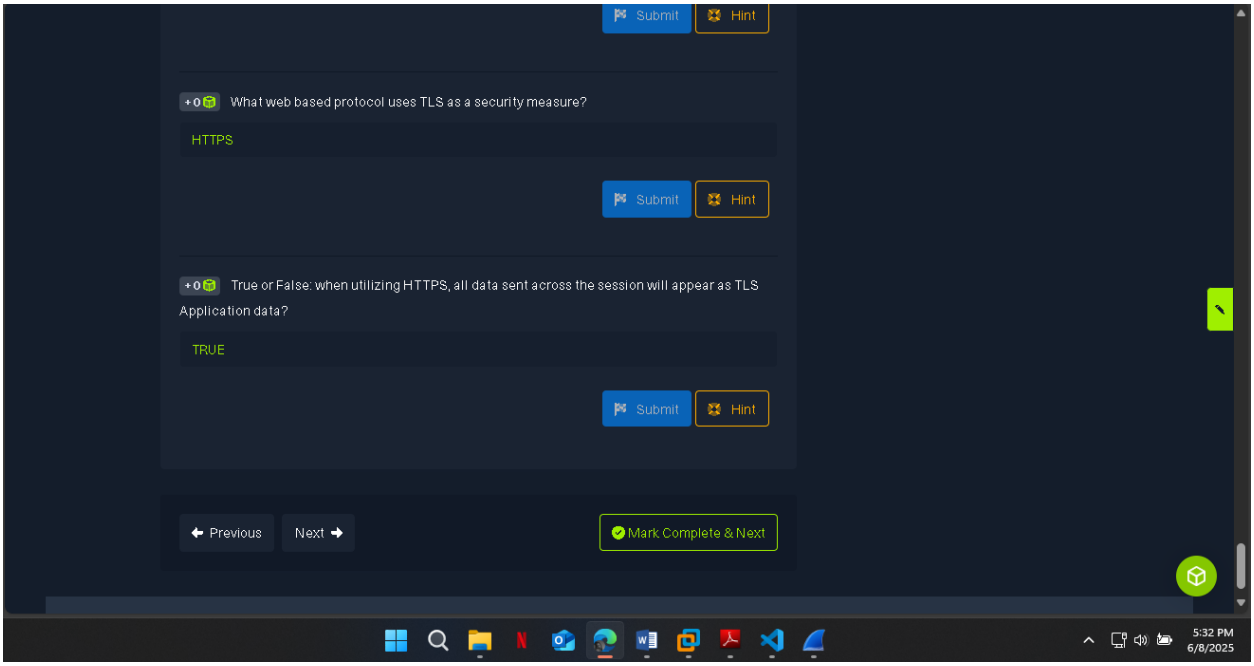
    - Port 20 for data transfer.
- Supports two modes:
    - **Active:** Server connects back to client for data transfer.
    - **Passive:** Client initiates all connections (better for firewalls).
- Commands include USER (login), PASS (password), LIST (directory listing), RETR (download file), and QUIT (end session).
- FTP is inherently insecure; modern alternatives like **SFTP** or **FTPS** are preferred.
- Browser support for FTP has mostly been deprecated.

SMB (Server Message Block)

- A protocol primarily for Windows environments to share files, printers, and other resources over a network.
- Requires user authentication to access resources.
- Uses TCP port 445 for direct communication; older versions used NetBIOS over UDP or TCP ports 137-139.
- SMB traffic includes file shares, printer access, and authentication processes.
- Because SMB allows access to critical resources, it is a common target for attackers attempting lateral movement within networks.
- Monitoring SMB traffic for unusual login failures or unexpected resource access is important for security.




+0 SMB has moved to using what TCP port?
445
Submit
Hint
+0 Hypertext Transfer Protocol uses what well known TCP port number?
80
Submit
Hint
+0 What HTTP method is used to request information and content from the webserver?
GET
Submit
Hint
+0 What web based protocol uses TLS as a security measure?
5:32 PM
6/8/2025


Submit
Hint
+0 What web based protocol uses TLS as a security measure?
HTTPS
Submit
Hint
+0 True or False: when utilizing HTTPS, all data sent across the session will appear as TLS Application data?
TRUE
Submit
Hint
Previous
Next
Mark Complete & Next
5:32 PM
6/8/2025

TCPDUMP FUNDAMENTALS

Tcpdump is a powerful, command-line network packet analyzer used primarily on Unix-like operating systems. It enables real-time capturing and inspection of network traffic by directly accessing data frames via interfaces running in promiscuous mode. This capability allows Tcpdump to monitor packets destined for any device on the local network, making it an indispensable tool for network diagnostics, security analysis, and traffic troubleshooting.

Originally designed for Unix environments, Tcpdump is commonly preinstalled on most Linux distributions (e.g., Ubuntu, Debian) and supports platforms like AIX, BSD, and Solaris. On Windows, a now-discontinued counterpart called WinDump existed, but modern users typically run Tcpdump through Linux environments like WSL (Windows Subsystem for Linux).

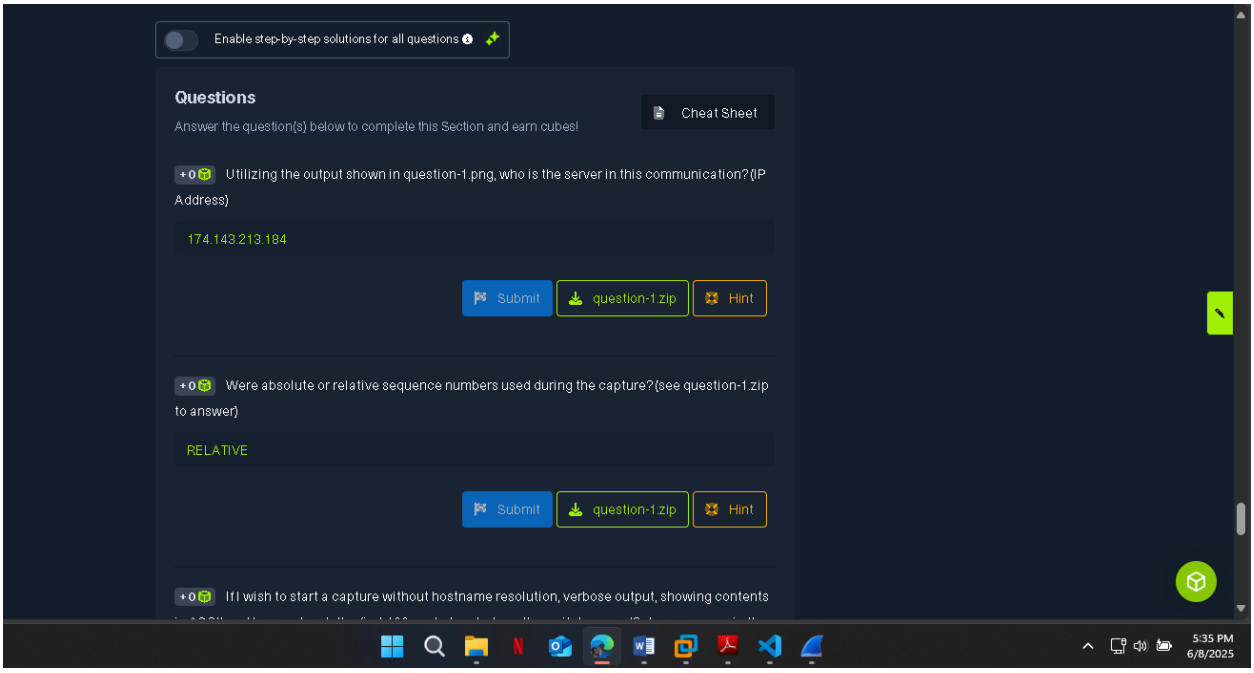
Key features and usage essentials include:

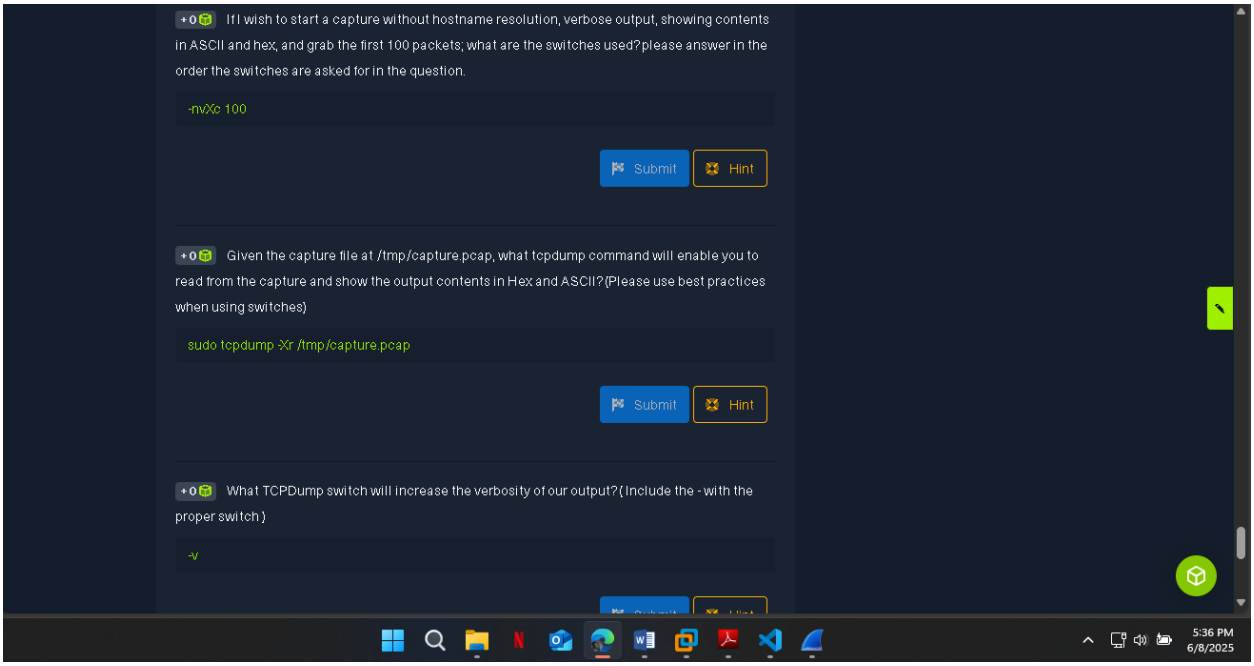
- Requires root or administrator privileges to access network interfaces.
- Relies on libpcap libraries to capture packets.
- Offers extensive filtering options to capture specific traffic patterns.
- Command-line based, no GUI needed—ideal for remote sessions via SSH.
- Captured packets can be saved into PCAP files for offline analysis.

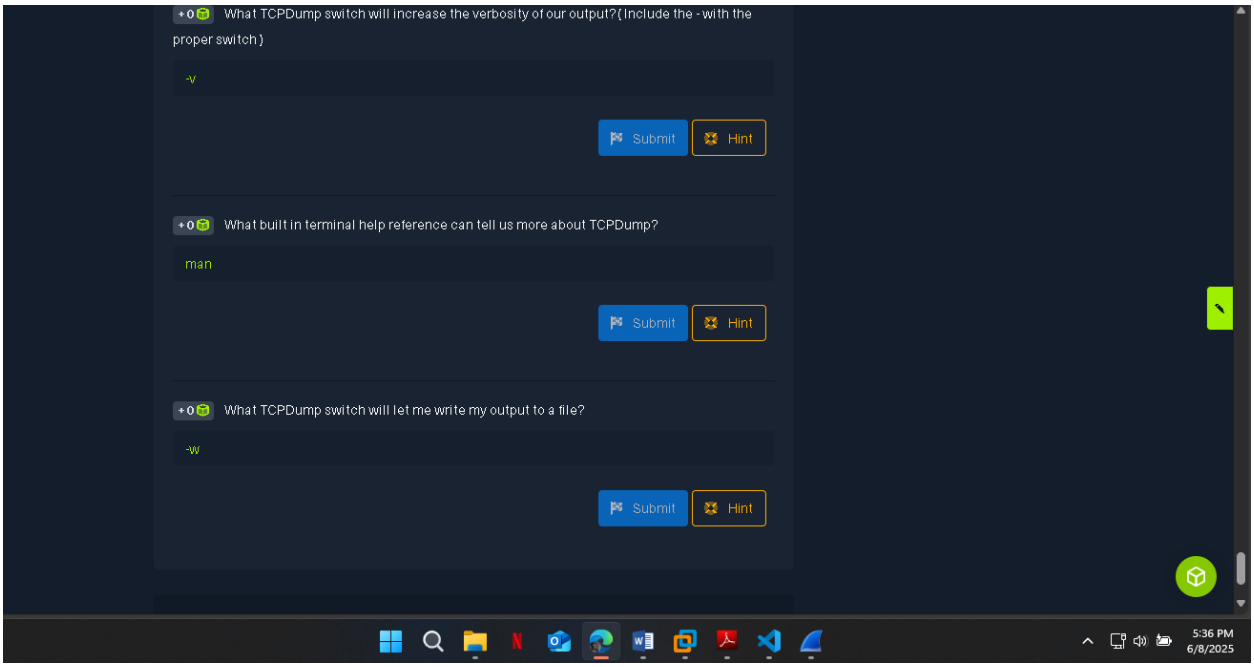
Basic Tcpdump options:

- -D: Lists available capture interfaces.
- -i <interface>: Selects the interface to capture traffic from.
- -nn: Disables hostname and port number resolution for cleaner output.
- -e: Includes Ethernet header information in packet output.
- -X: Displays packet contents in both hex and ASCII for deeper packet inspection.
- -w <file>: Writes captured traffic to a file.
- -r <file>: Reads packets from a saved capture file.
- Verbosity flags (-v, -vv, -vvv) increase detail in output.

Tcpdump excels in its simplicity and raw power, making it a staple for network professionals who need a reliable, flexible tool without the overhead of graphical environments. Mastering its core commands unlocks the ability to monitor, debug, and secure network infrastructure with precision and speed.


Enable step-by-step solutions for all questions
Questions
Answer the question(s) below to complete this Section and earn cubes!
Cheat Sheet
+0 Utilizing the output shown in question-1.png, who is the server in this communication? (IP Address)
174.143.213.184
Submit
question-1.zip
Hint
+0 Were absolute or relative sequence numbers used during the capture? (see question-1.zip to answer)
RELATIVE
Submit
question-1.zip
Hint
+0 If I wish to start a capture without hostname resolution, verbose output, showing contents
5:35 PM
6/8/2025


+0 If I wish to start a capture without hostname resolution, verbose output, showing contents in ASCII and hex, and grab the first 100 packets; what are the switches used? please answer in the order the switches are asked for in the question.
-nvXc 100
Submit
Hint
+0 Given the capture file at /tmp/capture.pcap, what tcpdump command will enable you to read from the capture and show the output contents in Hex and ASCII? (Please use best practices when using switches)
sudo tcpdump -Xr /tmp/capture.pcap
Submit
Hint
+0 What TCPDump switch will increase the verbosity of our output? (Include the - with the proper switch)
-v
5:36 PM
6/8/2025


+0 What TCPDump switch will increase the verbosity of our output? (Include the - with the proper switch)
-v
Submit
Hint
+0 What built in terminal help reference can tell us more about TCPDump?
man
Submit
Hint
+0 What TCPDump switch will let me write my output to a file?
-w
Submit
Hint
5:36 PM
6/8/2025

FUNDAMENTALS LAB

As the newly appointed network administrator for the Corporation, I conducted an initial network traffic capture and analysis to establish a baseline of local broadcast domain traffic and validate the deployment of essential network monitoring tools. The primary focus was on utilizing tcpdump, a powerful command-line packet analyzer, to capture, filter, and analyze network packets on our Linux systems.

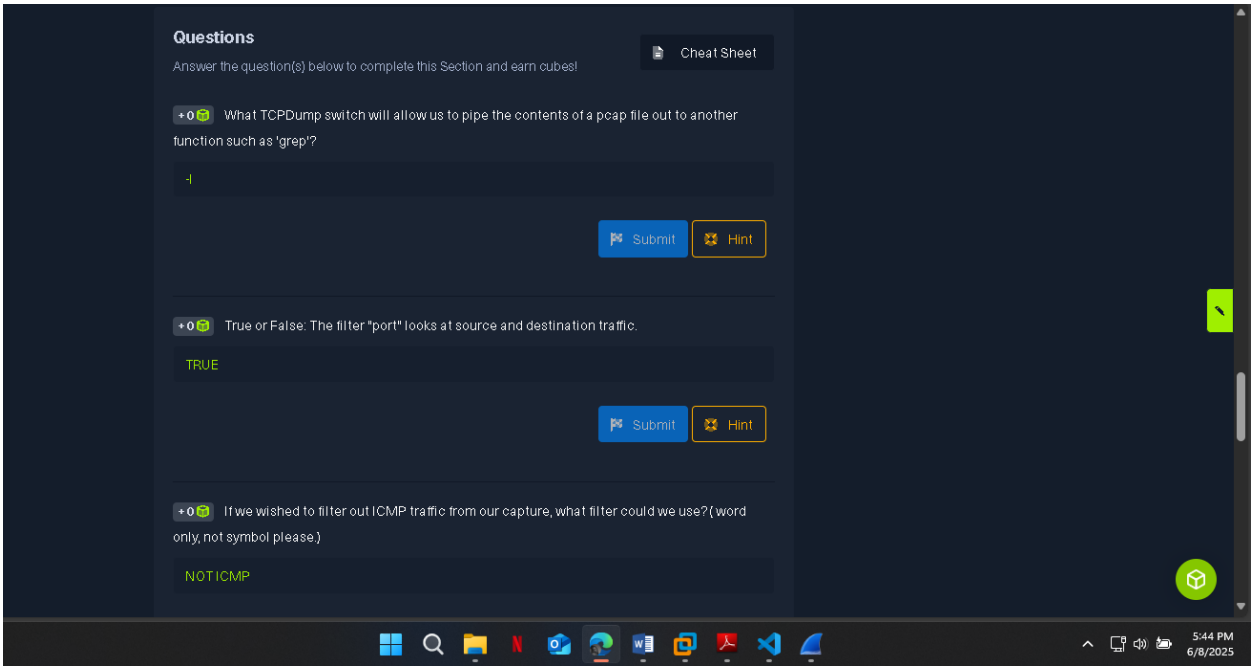
Key Activities:

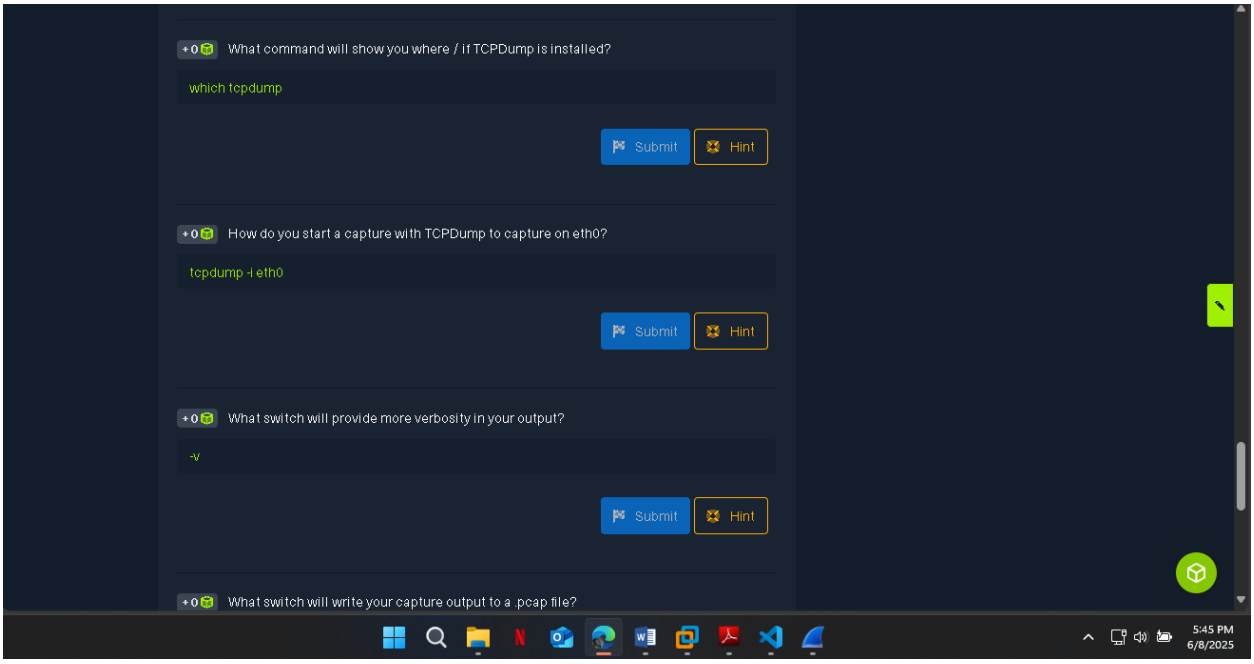
- **Tcpdump Installation Validation:** Confirmed the presence of tcpdump on our system using package verification commands to ensure readiness for packet capturing.
- **Interface Discovery:** Identified all available network interfaces on the host machine to select the appropriate listening device for capturing live traffic.
- **Packet Capture with Enhanced Verbosity:** Initiated network captures applying various filters and switches, including disabling DNS name resolution and enabling verbose output, to observe packet details in ASCII and hexadecimal formats. This approach enhanced insight into packet contents and communication patterns.
- **Data Persistence:** Saved network captures into .pcap files for subsequent offline analysis, enabling thorough examination and sharing of traffic data with the team.
- **Reading and Analyzing Captures:** Leveraged tcpdump's ability to read from saved capture files, adjusting output to disable hostname and port resolution, and displaying TCP sequence and acknowledgment numbers in absolute terms for clarity.

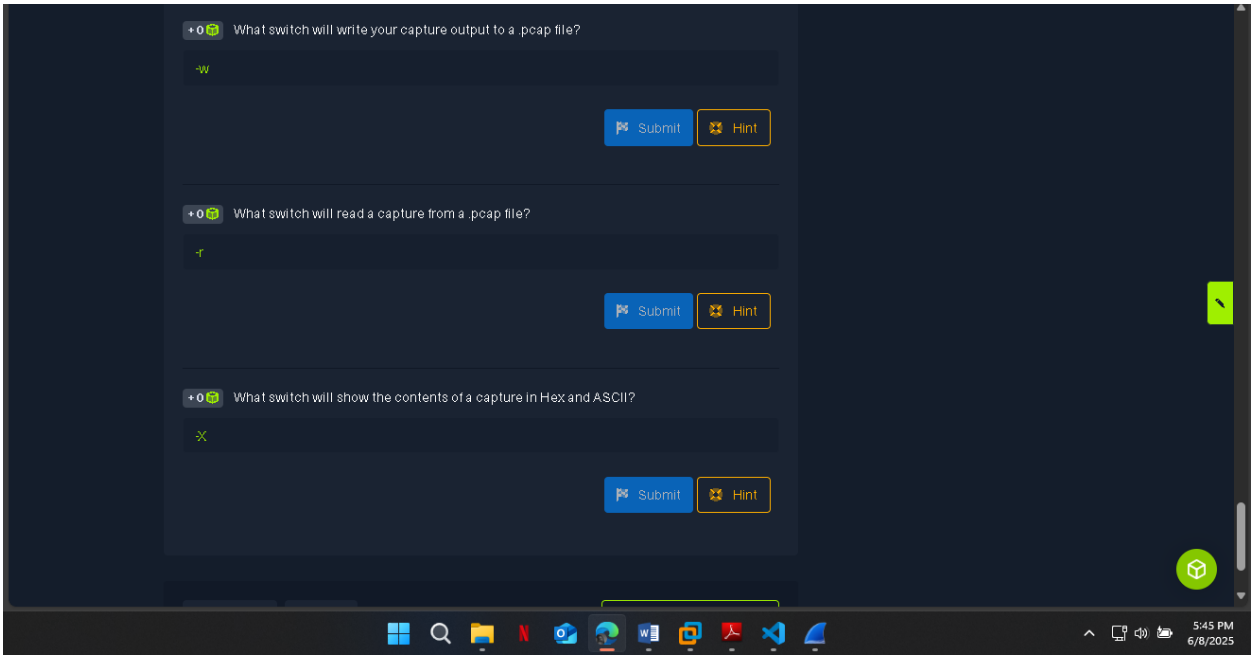
Outcomes and Next Steps:

- Successfully demonstrated the capability to capture, store, and analyze network traffic, establishing a foundational skillset and workflow for ongoing network monitoring.
- Laid groundwork for more advanced traffic filtering, pattern recognition, and identification of suspicious activity such as backdoors or unauthorized access attempts.
- Prepared to collaborate with security teams by providing detailed packet logs and filtering techniques to support proactive network defense.

This hands-on exposure to tcpdump commands, switches, and filters significantly bolstered network troubleshooting and security readiness, ensuring the Corporation's infrastructure can be monitored effectively and threats detected early.


Questions
Answer the question(s) below to complete this Section and earn cubes!
Cheat Sheet
+0 What TCPDump switch will allow us to pipe the contents of a pcap file out to another function such as 'grep'?
-l
Submit
Hint
+0 True or False: The filter "port" looks at source and destination traffic.
TRUE
Submit
Hint
+0 If we wished to filter out ICMP traffic from our capture, what filter could we use? (word only, not symbol please.)
NOT ICMP
5:44 PM
6/8/2025


+0 What command will show you where / if TCPDump is installed?
which tcpdump
Submit
Hint
+0 How do you start a capture with TCPDump to capture on eth0?
tcpdump -i eth0
Submit
Hint
+0 What switch will provide more verbosity in your output?
-v
Submit
Hint
+0 What switch will write your capture output to a .pcap file?
5:45 PM
6/8/2025


+0 What switch will write your capture output to a .pcap file?
-w
Submit
Hint
+0 What switch will read a capture from a .pcap file?
-r
Submit
Hint
+0 What switch will show the contents of a capture in Hex and ASCII?
-X
Submit
Hint
5:45 PM
6/8/2025

TCPDUMP PACKET FILTERING

Filtering and Advanced Syntax Options – Tcpdump

Helpful Tcpdump Filters

Tcpdump filters allow you to capture or display only the traffic relevant to your analysis. These filters use Berkeley Packet Filter (BPF) syntax and include a wide range of expressions to match hosts, ports, protocols, and more. They enhance both performance and precision in packet capturing.

Host Filter

The host filter restricts capture to traffic associated with a specific IP address.

Syntax:
tcpdump host 192.168.1.5

Purpose:
Used to monitor all inbound and outbound traffic related to a particular host.

Source/Destination Filter

These filters allow the capture of traffic based on source (src) or destination (dst) IP addresses.

Syntax:

- tcpdump src 10.0.0.1
- tcpdump dst 192.168.1.10

Purpose:
Ideal for isolating communication direction in traffic analysis.

Utilizing Source with Port as a Filter

Combining source/destination IP with a specific port enables more granular captures.

Syntax:
tcpdump src 10.0.0.5 and port 443

Purpose:
To isolate specific traffic from a host to a particular service.

Protocol Filter – Number

The proto filter uses protocol numbers defined in the IP header (e.g., TCP is 6, UDP is 17).

Syntax:
tcpdump ip proto 6

Purpose:
Used when specific protocol numbers are required over names, especially in low-level traffic analysis.

Port Filter
Filters packets by a single TCP or UDP port.

Syntax:
tcpdump port 80

Purpose:
To capture all traffic on a particular service port (e.g., HTTP).

Port Range Filter
Specifies a range of ports to monitor.

Syntax:
tcpdump portrange 1000-2000

Purpose:
Useful for observing traffic across a range of dynamic or custom application ports.

Less/Greater Filter
Filters packets by length, allowing you to capture only packets larger or smaller than a specified byte size.

Syntax:

- tcpdump less 128
- tcpdump greater 512

Purpose:
Helps in identifying anomalies such as very large or very small packets.

Utilizing Greater
The greater keyword filters out packets smaller than the specified byte size.

Syntax:
tcpdump greater 1000

Purpose:
Effective for locating potentially malicious large payloads or file transfers.

AND Filter
Combines multiple filter expressions where all conditions must be true.

Syntax:
tcpdump src 10.0.0.1 and port 22

Purpose:
Narrowing down the capture to very specific traffic scenarios.

Basic Capture With No Filter

Running Tcpdump with no filter captures all packets on the selected interface.

Syntax:
tcpdump

Purpose:
Used for broad traffic analysis or initial recon before applying specific filters.

OR Filter

Captures traffic that matches any of the specified conditions.

Syntax:
tcpdump port 80 or port 443

Purpose:
Useful for capturing multiple services or scenarios in a single run.

NOT Filter

Excludes packets that meet the specified condition.

Syntax:
tcpdump not port 22

Purpose:
Used to filter out noise or known safe traffic from the capture.

Pre-Capture Filters vs. Post-Capture Processing

- **Pre-Capture Filters**: Applied during capture; more efficient as they prevent irrelevant packets from being saved.
- **Post-Capture Processing**: Performed after capture using tools like grep or analysis in Wireshark; more flexible but requires more resources.

Purpose:
Pre-capture filters are ideal for real-time monitoring and storage conservation, while post-capture processing supports deeper offline analysis.

Piping a Capture to Grep

You can pipe the output of Tcpdump into grep to search for specific content in real-time.

Syntax:
tcpdump -A | grep "HTTP"

Purpose:
Helps in identifying application-level content (e.g., URLs, headers) during inspection.

Looking for TCP Protocol Flags

Tcpdump can display TCP flags such as SYN, ACK, FIN, etc. Using these flags can help detect handshake attempts, terminations, or anomalies.

Syntax:
tcpdump 'tcp[tcpflags] & tcp-syn != 0'

Purpose:
Used for detecting connection attempts, port scanning, or SYN flood attacks.

Hunting for a SYN Flag

Targeting SYN packets identifies TCP connection initiations.

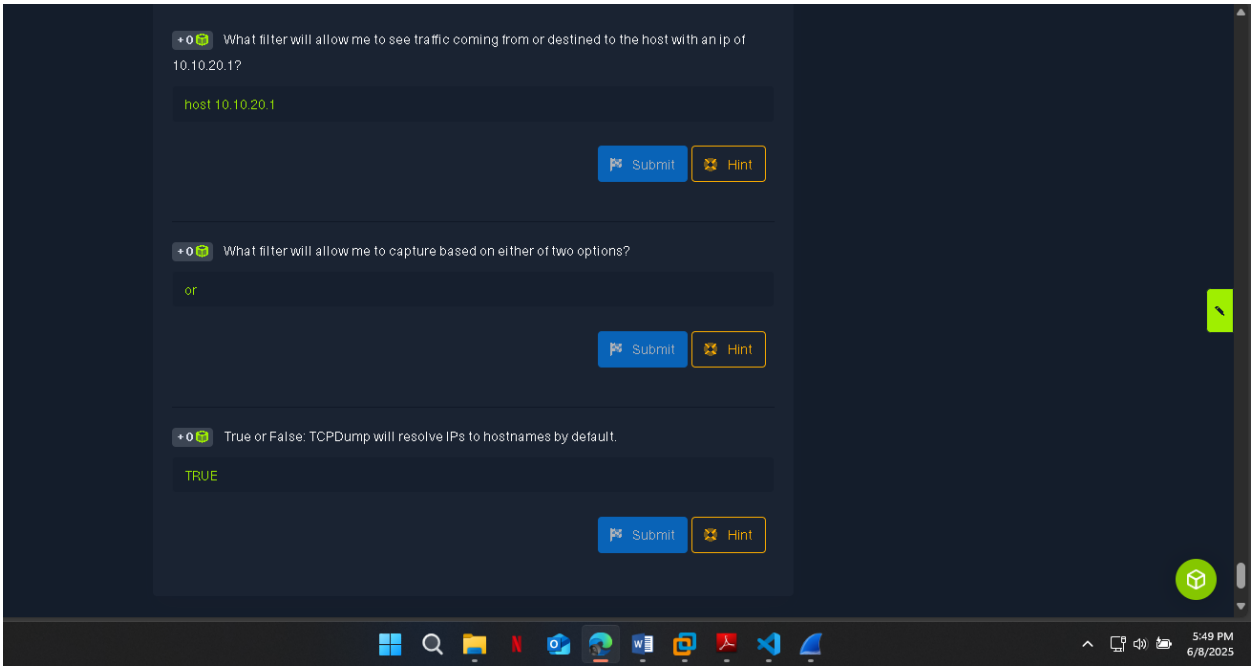
Syntax:
tcpdump 'tcp[tcpflags] & tcp-syn != 0 and tcp[tcpflags] & tcp-ack == 0'

Purpose:
Specifically captures the first packet in the TCP 3-way handshake, often used for scanning detection.

Protocol RFC Links

Each protocol's behavior, port usage, and expected traffic patterns are defined in official RFCs (Request for Comments). Understanding these documents helps in interpreting what “normal” traffic should look like.

Purpose:
Provides the foundation for accurate analysis, anomaly detection, and secure network architecture.


+0
What filter will allow me to see traffic coming from or destined to the host with an ip of 10.10.20.1?
host 10.10.20.1
Submit
Hint
+0
What filter will allow me to capture based on either of two options?
or
Submit
Hint
+0
True or False: TCPDump will resolve IPs to hostnames by default.
TRUE
Submit
Hint
5:49 PM
6/8/2025

INTERROGATING NETWORK TRAFFIC WITH CAPTURE AND DISPLAY FILTERS USING TCPDUMP

This lab provided an in-depth, hands-on exercise in network traffic analysis using tcpdump, focusing on interrogation techniques through both capture and display filters applied to a pre-captured .pcap file. The objective was to build familiarity with TCPDump filtering syntax while extracting meaningful insights from observed traffic—specifically identifying DNS and HTTP/HTTPS communications within a local enterprise network.

Key Activities and Learnings:

1. Unfiltered Packet Inspection:

The lab began by reading the .pcap file without applying any filters, which allowed for a broad overview of all captured traffic. This step served to establish a baseline understanding of the traffic structure and volume.

2. Protocol and Port Identification:

The analysis revealed the presence of DNS, HTTP, and HTTPS protocols, commonly associated with ports 53, 80, and 443 respectively. This step highlighted how TCPDump filters based on protocol and port can refine visibility into specific traffic types.

3. Traffic Pattern Recognition and Conversation Analysis:

Using filters and options such as -S (for absolute TCP sequence numbers), the lab facilitated the identification of host-server communication patterns. Analysts distinguished between client and server roles based on port usage—servers typically operated on well-known ports, while clients used ephemeral high ports. The first full TCP three-way handshake was examined in detail, including port numbers and endpoint roles.

4. Temporal and Protocol-Specific Analysis:

Timestamping helped determine the chronology of conversations. DNS queries and responses were parsed to uncover IP address resolutions (e.g., resolving apache.org) and to identify DNS servers. Further, the relevant protocols and port numbers were linked to application-level insights.

5. Targeted Traffic Filtering:

The exercise emphasized selective visibility using precise filters. Analysts isolated DNS traffic using udp and port 53 (with attention to TCP where applicable) and extracted requested domain names, A-record queries, and DNS responses. ASCII and hexadecimal output (-X flag) supported deeper inspection of data payloads.

6. Web Traffic Interrogation:

Filters were refined to display only HTTP and HTTPS traffic, revealing browsing behavior. Analysts reviewed HTTP request methods (e.g., GET, POST) and typical server responses (e.g.,

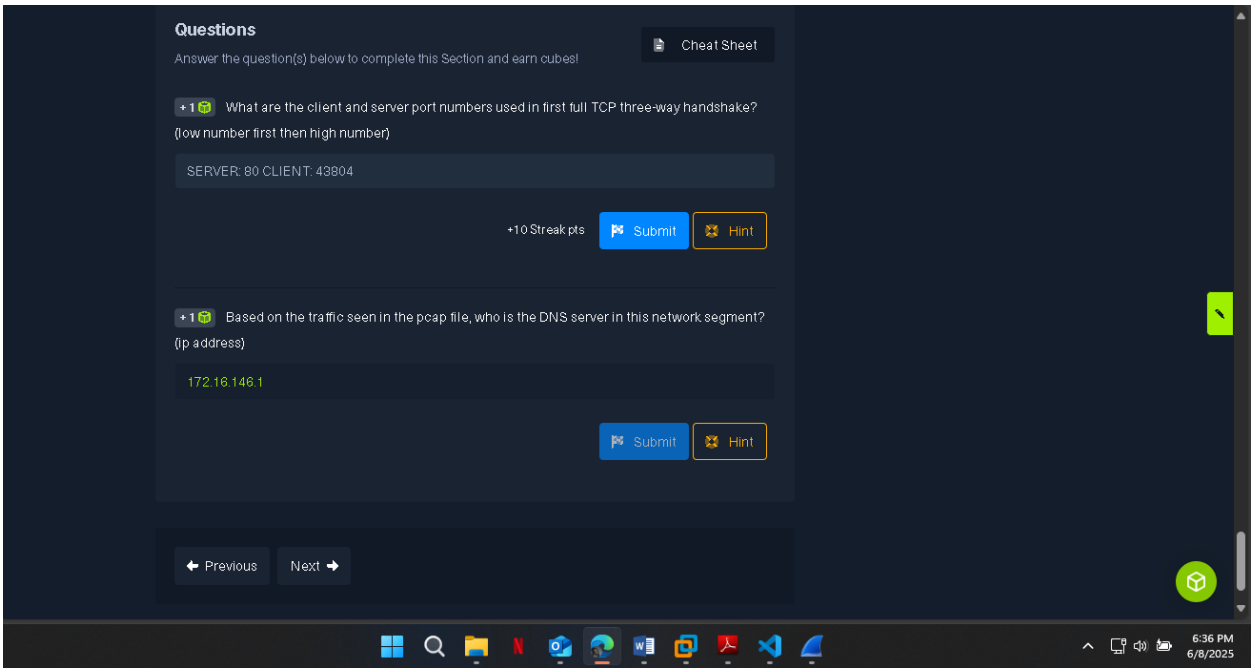
HTTP 200 OK). By examining packet content, information about visited web pages and server behavior was gathered.

7. Application Fingerprinting:

The final stage involved profiling the webserver involved in the first TCP conversation. Using hex and ASCII analysis of HTTP headers and server responses, the underlying application or platform running the webserver (e.g., Apache, Nginx) was inferred.

Summary of Outcomes:

- Gained practical experience in applying both pre-capture and post-capture filters to dissect .pcap files effectively.
- Developed a strong understanding of protocol behavior, port association, and DNS resolution mechanisms.
- Successfully used TCPDump to identify and profile critical infrastructure components such as DNS and web servers.
- Learned to extract both metadata (IP addresses, ports, timestamps) and application-level data (HTTP content, DNS answers) for deeper network forensic analysis.



ANALYSIS WITH WIRESHARK

This summary is a structured consolidation of all the key points discussed, suitable for academic or professional documentation.

Wireshark Overview

Wireshark is a free and open-source network protocol analyzer used for network troubleshooting, analysis, software and protocol development, and education. It allows users to capture and interactively browse traffic running on a computer network.

Key Features:

- Live packet capture and offline analysis
- Multi-platform support (Windows, Linux, macOS)
- Deep inspection of hundreds of protocols
- Export of packet data in .pcap or .pcapng formats
- Decryption support for many protocols (SSL/TLS, WPA/WPA2, etc.)
- Extensive filtering capabilities
- Three-pane GUI for efficient analysis

Wireshark GUI Components:

Pane Name	Description
Packet List Pane	Displays all captured packets with metadata (e.g., time, protocol, source).
Packet Details Pane	Shows the layered protocol breakdown (Ethernet, IP, TCP, HTTP, etc.).
Packet Bytes Pane	Displays raw data (hex and ASCII) of the selected packet.

TShark (Terminal Wireshark)

TShark is the command-line version of Wireshark. It provides similar capture and analysis capabilities in a text-based format, ideal for automated workflows, headless servers, or remote analysis via SSH.

Basic Command Syntax:

tshark -i <interface> -w <output_file.pcap> -f "<capture_filter>"

Key Flags:

Flag	Description
-i	Specify interface to capture from

Flag	Description
-w	Write captured packets to a file
-f	Apply a capture filter (BPF syntax)
-r	Read from a saved .pcap file
-Y	Apply a display filter
-c	Capture a specific number of packets
-T fields	Extract specific fields (custom reporting)

Termshark (Text-based Wireshark Interface)

Termshark is a terminal user interface (TUI) for TShark that provides a pseudo-GUI experience within the terminal, simulating Wireshark's layout and functionality.

Key Benefits:

- Useful in environments without a graphical interface
- Built-in packet inspection via keystrokes
- Ideal for quick inspections on remote systems

Filtering and Analysis

Capture Filters (BPF Syntax)

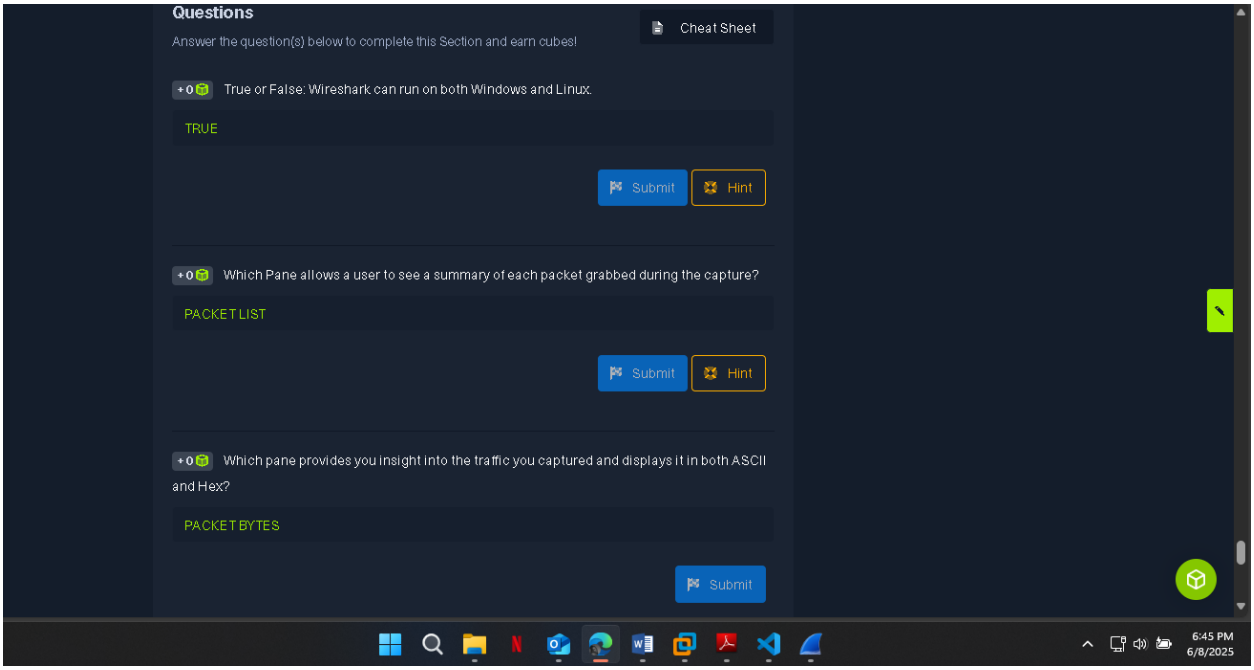
Applied *before* capturing; examples:

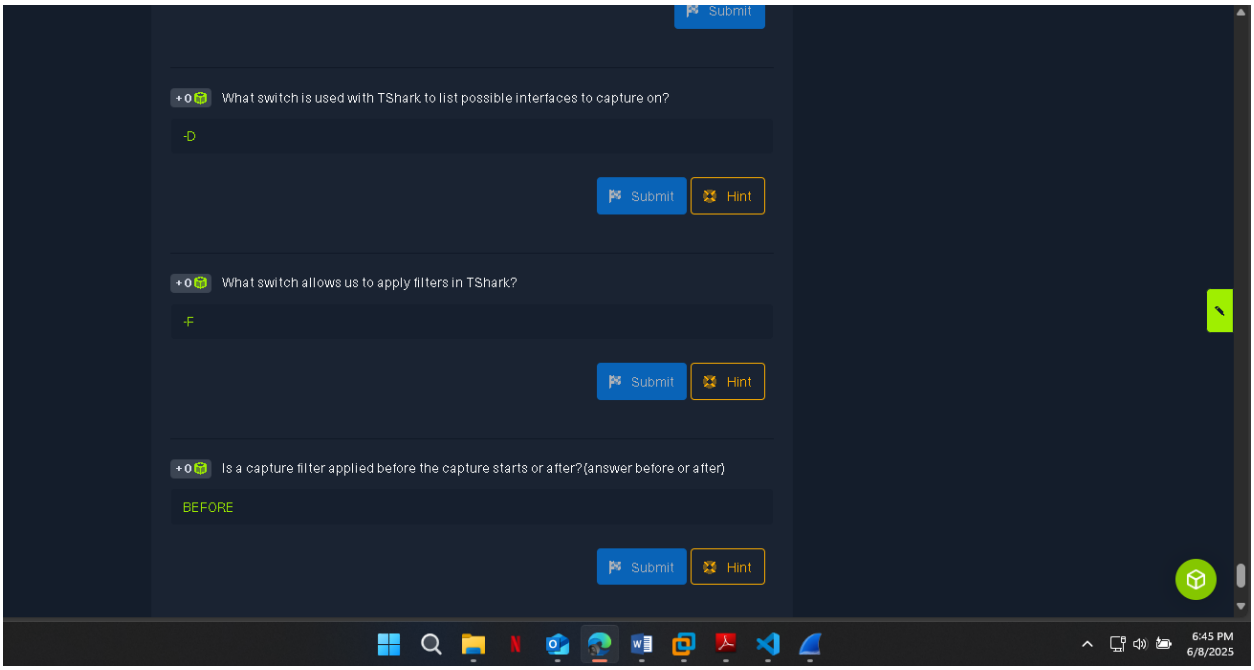
- port 80
- host 192.168.1.1
- tcp and dst port 443

Display Filters (Wireshark Syntax)

Applied *after* capture; examples:

- http
- ip.addr == 192.168.0.1
- tcp.flags.syn == 1 and tcp.flags.ack == 0


Questions
Answer the question(s) below to complete this Section and earn cubes!
Cheat Sheet
+0 True or False: Wireshark can run on both Windows and Linux.
TRUE
Submit
Hint
+0 Which Pane allows a user to see a summary of each packet grabbed during the capture?
PACKET LIST
Submit
Hint
+0 Which pane provides you insight into the traffic you captured and displays it in both ASCII and Hex?
PACKET BYTES
Submit
6:45 PM
6/8/2025


Submit
+0 What switch is used with TShark to list possible interfaces to capture on?
-D
Submit
Hint
+0 What switch allows us to apply filters in TShark?
-F
Submit
Hint
+0 Is a capture filter applied before the capture starts or after?(answer before or after)
BEFORE
Submit
Hint
6:45 PM
6/8/2025

WIRESHARK: ADVANCED USAGE

Wireshark includes advanced features and built-in plugins that support deep analysis of network traffic. These tools enhance its capability beyond simple packet capture, enabling thorough inspection of conversations, protocol behavior, and even file extraction.

Plugins via Statistics and Analyze Tabs

Wireshark offers various plugins accessible through the **Statistics** and **Analyze** tabs, allowing for in-depth traffic inspection:

- **Statistics Tab**: Provides aggregated views such as:
  - Protocol Hierarchy
  - Conversations by IP and port
  - Endpoints
  - IO graphs
- **Analyze Tab**: Offers features like:
  - Follow TCP/UDP streams
  - Expert Info diagnostics
  - Enabled Protocols
  - Display filter utilities

Following TCP Streams

Wireshark can reconstruct TCP streams for protocols using TCP (e.g., HTTP, FTP):

- Steps:
  1. Right-click a TCP packet → Select **Follow → TCP Stream**
  2. View full reassembled conversation
  3. Apply filter tcp.stream eq <number> to isolate the stream

File Extraction from Captures

Wireshark supports data/file extraction if full sessions are captured:

Steps:

1. Stop capture
2. Go to File → Export Objects
3. Select protocol (e.g., HTTP, DICOM, SMB) to export files

Note: Partial capture may prevent successful reconstruction.

FTP Traffic Analysis

FTP uses:

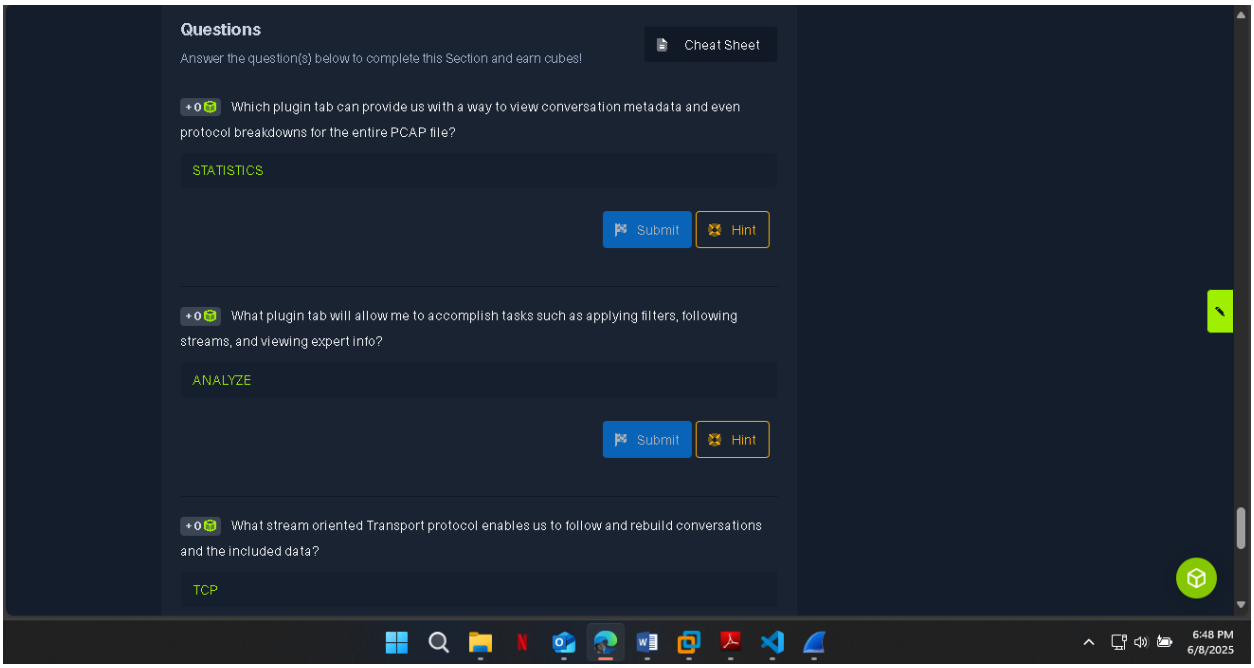
- **Port 21** for control commands (login, list, etc.)
- **Port 20** for actual data transfer

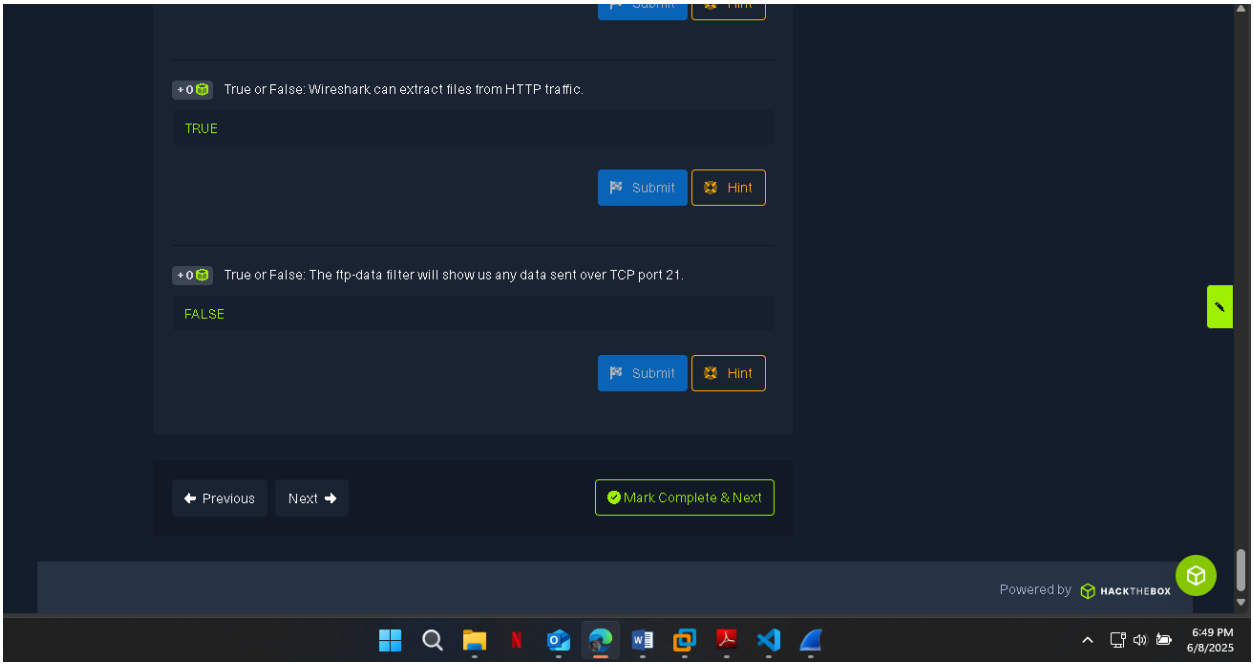
Useful Display Filters:

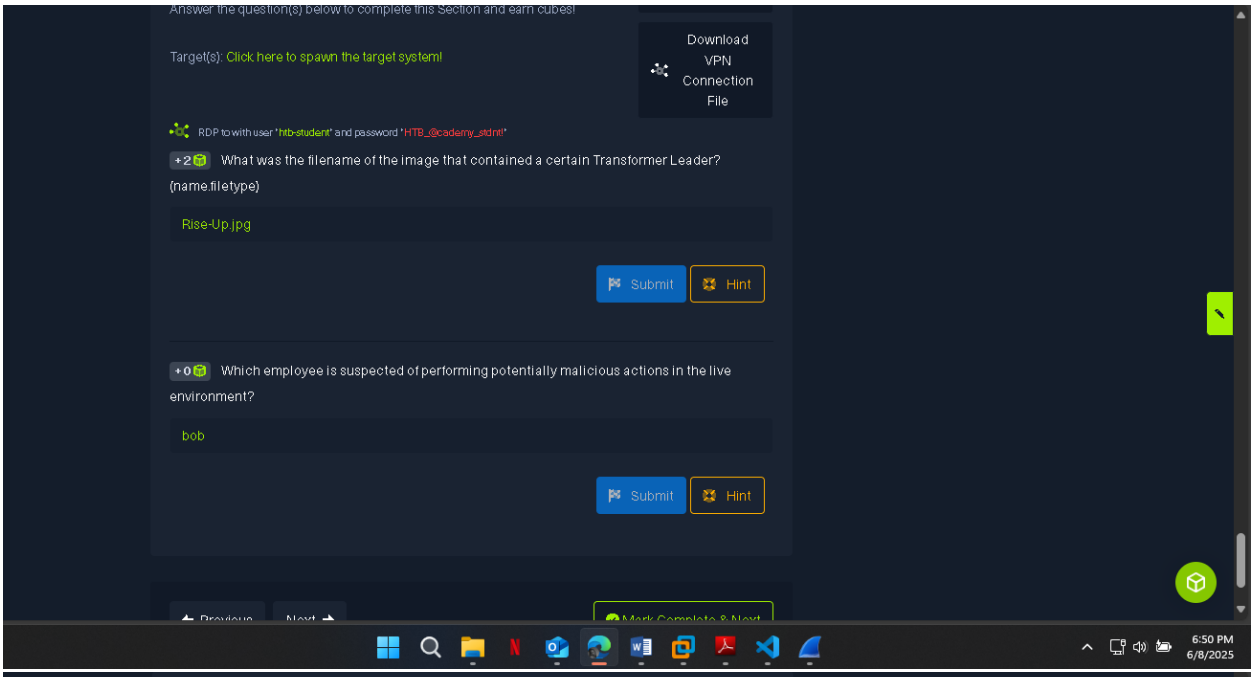
- ftp: General FTP traffic
- ftp.request.command: Control commands (reveals usernames, filenames, etc.)
- ftp-data: Actual file data transfer

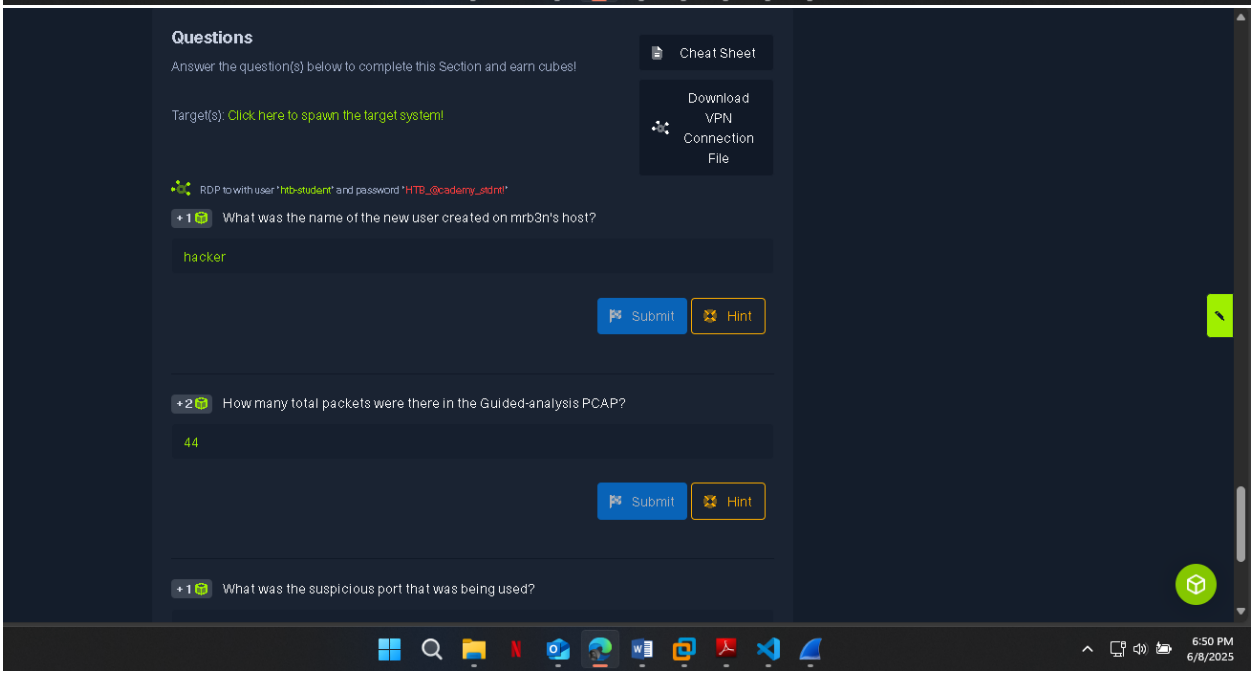
Extraction Process:

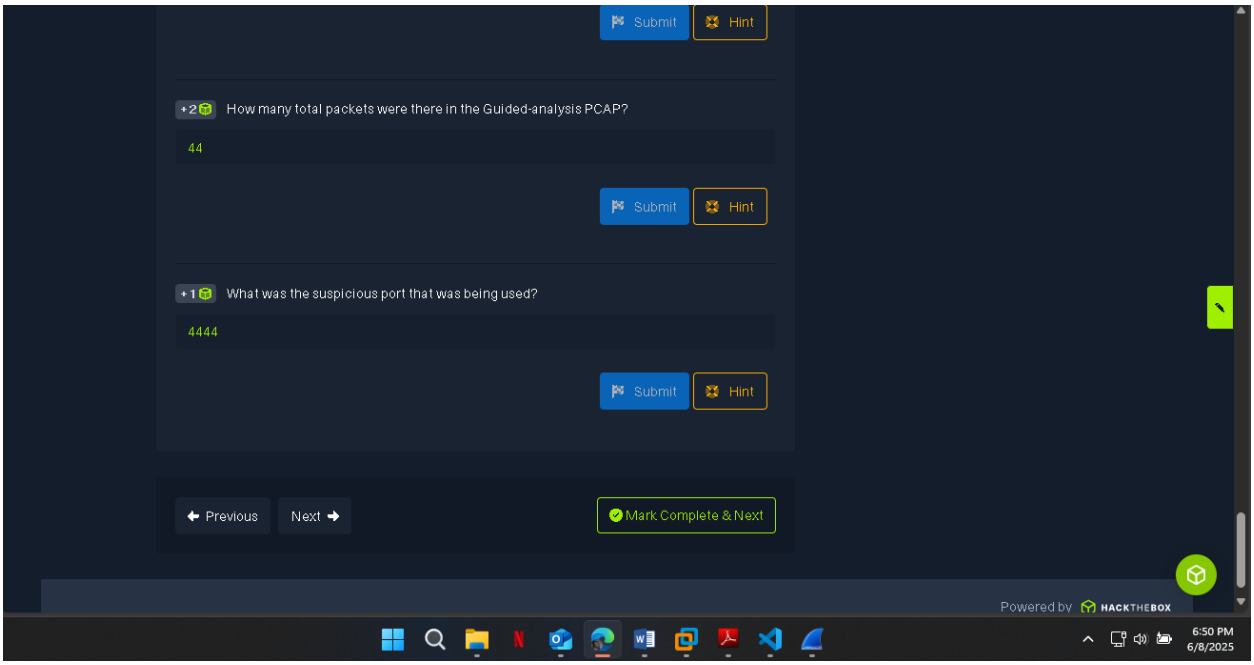
1. Identify FTP sessions using ftp
2. Use ftp.request.command to spot relevant commands and files
3. Apply ftp-data filter and follow TCP stream
4. Export data using **"Show and save data as: Raw"** and rename appropriately
5. Confirm file validity by checking file format after saving

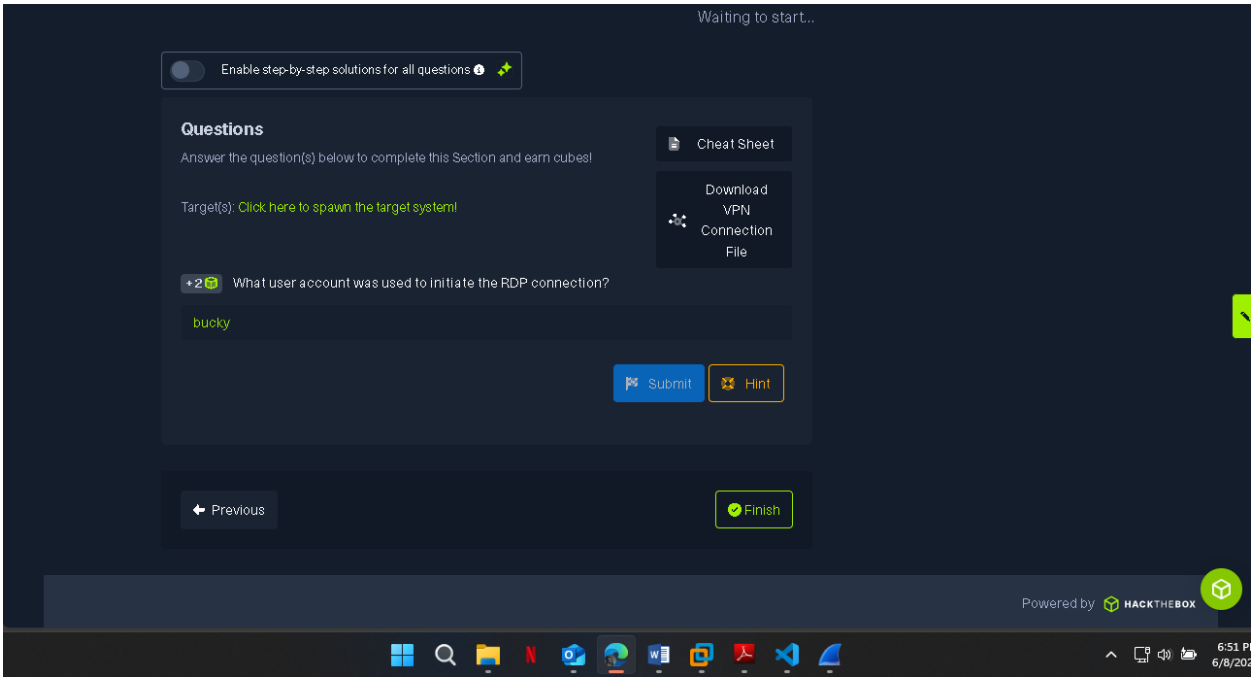



+0 True or False: Wireshark can extract files from HTTP traffic.
TRUE
Submit
Hint
+0 True or False: The ftp-data filter will show us any data sent over TCP port 21.
FALSE
Submit
Hint
Previous
Next
Mark Complete & Next
Powered by HACKTHEBOX
6:49 PM
6/8/2025


Answer the question(s) below to complete this Section and earn cubes!
Target(s): Click here to spawn the target system!
Download VPN Connection File
RDP to with user "htb-student" and password "HTB_@cademy_stdnt!"
+2 What was the filename of the image that contained a certain Transformer Leader? (name.filetype)
Rise-Up.jpg
Submit
Hint
+0 Which employee is suspected of performing potentially malicious actions in the live environment?
bob
Submit
Hint
6:50 PM
6/8/2025


Questions
Answer the question(s) below to complete this Section and earn cubes!
Cheat Sheet
Target(s): Click here to spawn the target system!
Download VPN Connection File
RDP to with user "htb-student" and password "HTB_@cademy_stdnt!"
+1 What was the name of the new user created on mrb3n's host?
hacker
Submit
Hint
+2 How many total packets were there in the Guided-analysis PCAP?
44
Submit
Hint
+1 What was the suspicious port that was being used?
6:50 PM
6/8/2025


Submit
Hint
+2 How many total packets were there in the Guided-analysis PCAP?
44
Submit
Hint
+1 What was the suspicious port that was being used?
4444
Submit
Hint
Previous
Next
Mark Complete & Next
Powered by HACKTHEBOX
6:50 PM
6/8/2025


Waiting to start...
Enable step-by-step solutions for all questions
Questions
Answer the question(s) below to complete this Section and earn cubes!
Target(s): Click here to spawn the target system!
Cheat Sheet
Download VPN Connection File
+2 What user account was used to initiate the RDP connection?
bucky
Submit
Hint
Previous
Finish
Powered by HACKTHEBOX

CONCLUSION

The completion of the Hack The Box network traffic analysis lab highlights the critical role of packet inspection and protocol analysis in modern cybersecurity operations. By effectively applying Wireshark filters, examining communication streams, and extracting files from network traffic, the lab demonstrated how hidden data and unauthorized transfers can be uncovered.