PennState
College of Information
Sciences and Technology

Lab Assignment Report

COURSE:	**IST 894**
ASSIGNMENT:	**LAB ASSIGNMENT REPORT 7**
SUBMITTED BY:	**BRANTLEY PRICE**
DUE DATE:	**JULY 20TH, 2025**
INSTRUCTOR:	**DR. BARTOLACCI**

Contents

General Context

The first lab in the Security+ cyber range, “Web Vulnerabilities,” focused on web application vulnerabilities, which are commonly exploited due to poor input validation or insecure coding practices. Some of the attack vectors the student interacted with were cross-site scripting (XSS), SQL injection, directory traversal, and file inclusion. These attack vectors are especially pertinent as OWASP’s Top 10:2021 had XSS and SQL injections ranked #3, and just behind in #4 was insecure design (OWASP, 2022). Each of the vulnerabilities allowed the student to see how easily web apps can be exploited if the necessary care is not taken regarding these vulnerabilities.

The second lab, “Secure and Insecure Protocols,” focused on four commonly used network protocols: Telnet, FTP, SSH, and SFTP. Using tcpdump and Wireshark, the student successfully captured and analyzed packets as they traversed the network. Once packets were captured, specifically over Telnet and FTP, the student was able to see usernames and passwords in plaintext. Telnet is a highly vulnerable legacy protocol because it does not encrypt data in transit (UK Cyber Security Ltd., 2023). In contrast, SSH and SFTP were demonstrated to be secure protocols that encrypt data in transit. The student was unable to view any data within the packets sent over SSH and SFTP due to the use of Diffie-Hellman encryption.

These cyber range exercises reinforce the criticality of secure software design, and input validation, and data in transit encryption. Vulnerabilities can be found anywhere in the stack, but when found at the application or transport layer, they can quickly lead to significant security incidents. Though the vulnerabilities mentioned in the exercises seem like low-hanging fruit, they continue to top annual lists of the most exploited vulnerabilities. IT professionals must understand this and ensure they can deploy the appropriate safeguards to protect against them.

Technical Context

The “Web Vulnerability” exercise demonstrated how attackers can exploit gaps on both the client and server sides if they know what they are doing. As the student stepped through the reflected, stored, and DOM-based XSS exploits, input validation was a theme. In the absence of proper input validation, malicious scripts were executed in the browser. This can lead to stolen sessions or the injection of malicious content. DOM-based XSS exploits are more complex than reflected and stored XSS exploits, as they utilize JavaScript to exploit the vulnerability (PortSwigger, n.d.). JavaScript is well-known for its vulnerable nature, so it is important to pay close attention to this exploit.

Moving from web app vulnerability to network traffic, the student used tcpdump and Wireshark to capture packets and confirmed that Telnet and FTP are not suitable for passing confidential information. This is because of their inherent lack of data-in-transit encryption, making them vulnerable to eavesdropping attacks. SSH and SFTP, however, showed the student encrypted sessions. The packets intercepted by the student were unreadable to anyone without the keys necessary to decrypt the data. Additionally, Wireshark excels at providing a substantial amount of metadata, which can be invaluable when analyzing network data. Data from the top to the bottom of the TCP/IP stack can be collected and analyzed using Wireshark (Garn, 2024).

A key point the student should take away from these exercises is the need for layered defenses. In the development of web apps, software developers must apply rigorous security standards to their craft; it all starts with software development. After that piece is handled, where applicable, data-in-transit encryption must be a standard. In reality, there is virtually no reason to use legacy protocols like Telnet in 2025; it is best to do away with them unless absolutely necessary. Applying these principles aligns with OWASP and broader cybersecurity frameworks, which makes the difficult task of network defense a bit easier.

Screenshots

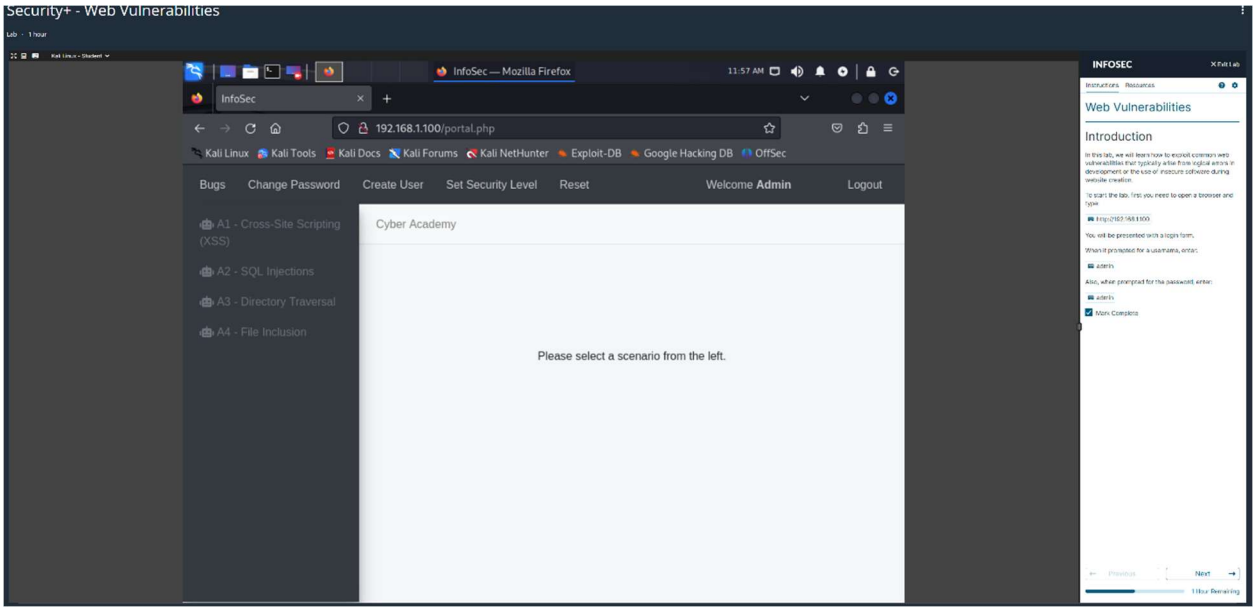

Figure 1: Web Vulnerabilities Introduction

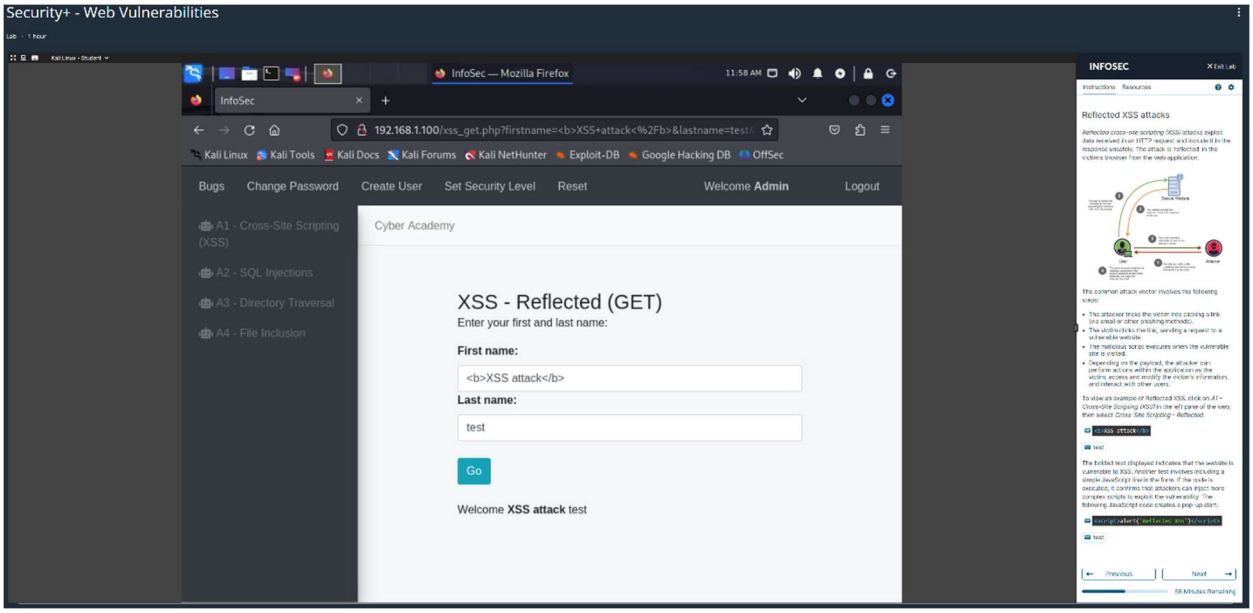

Figure 2: Testing the XSS Reflected Vulnerability

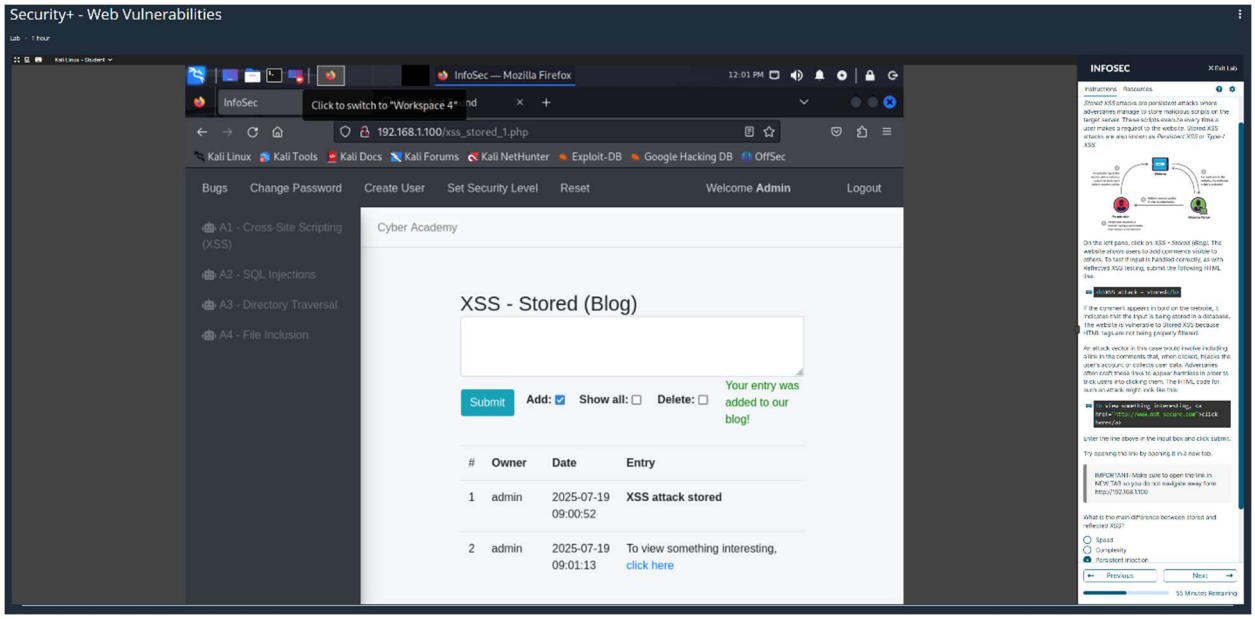

Figure 3: Testing the XSS Stored Vulnerability

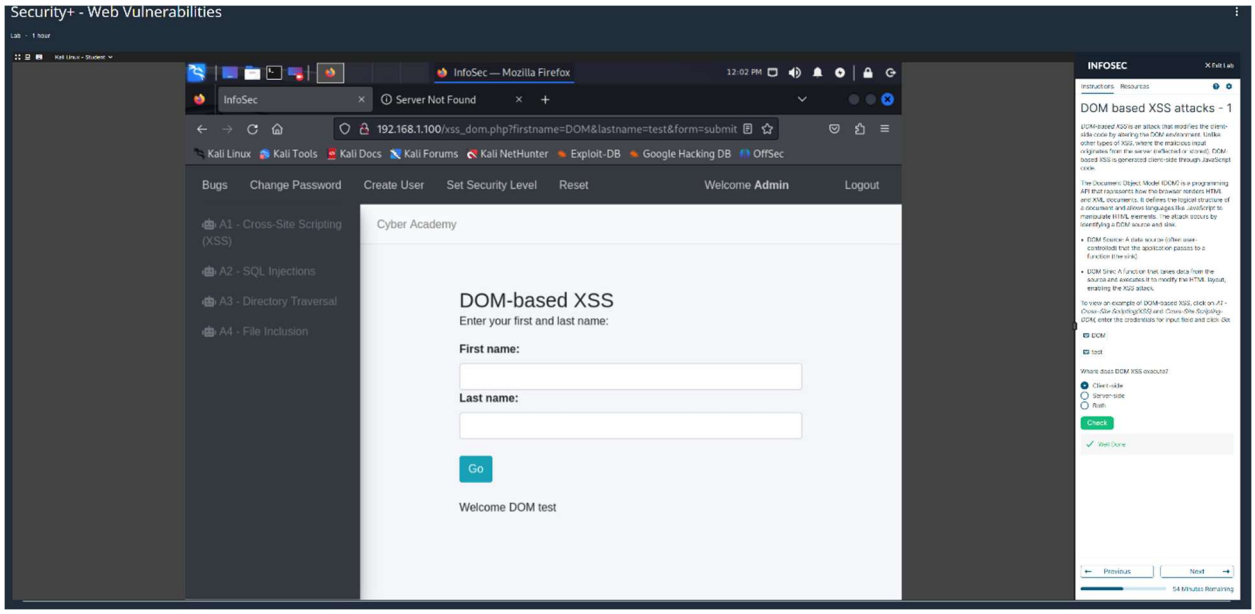

Figure 4: Testing the DOM-based XSS Vulnerability

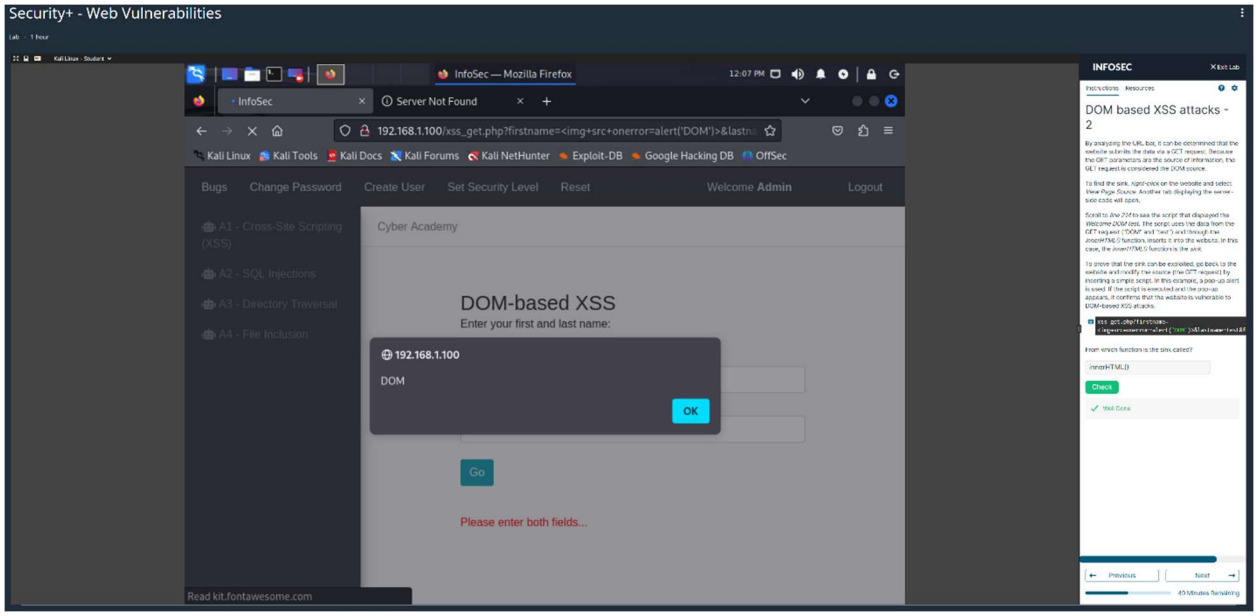

Figure 5: DOM-based JavaScript Exploitation

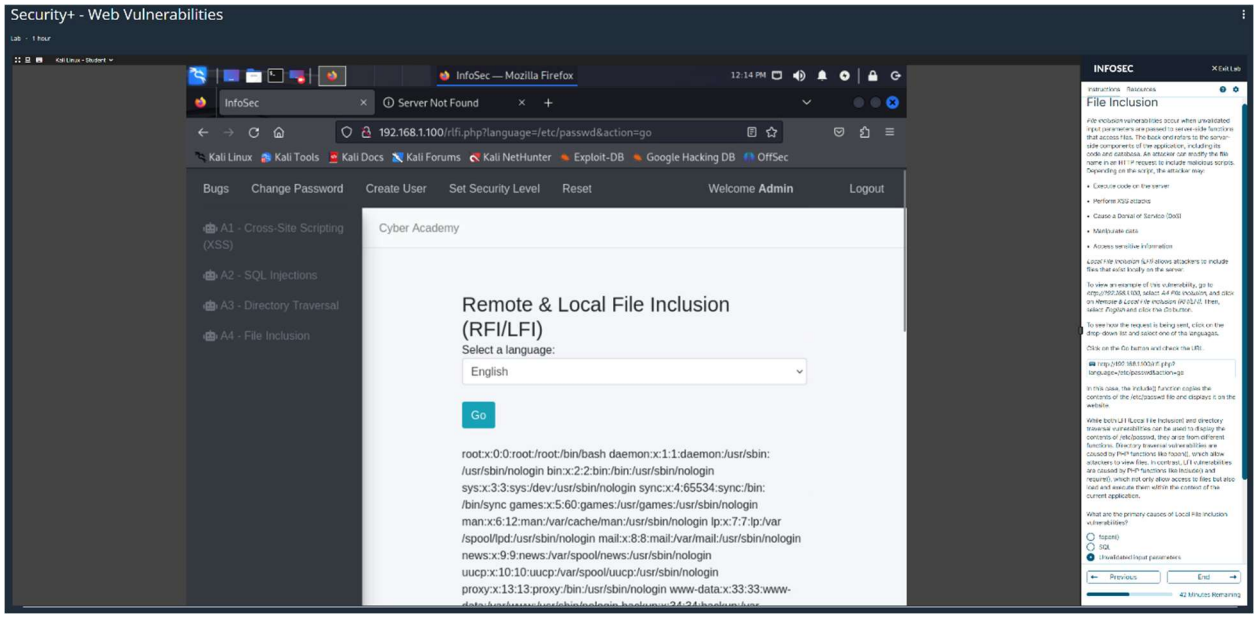

Figure 6: Final Web Application Screenshot

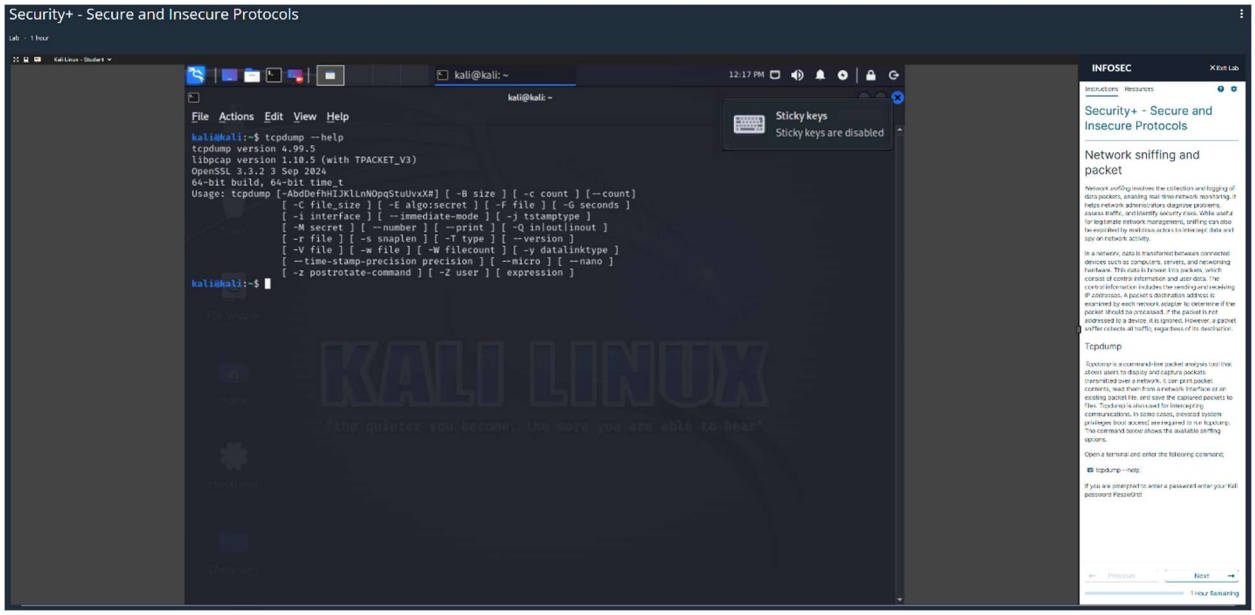

Figure 7: tcpdump and Secure/Insecure Protocols Introduction

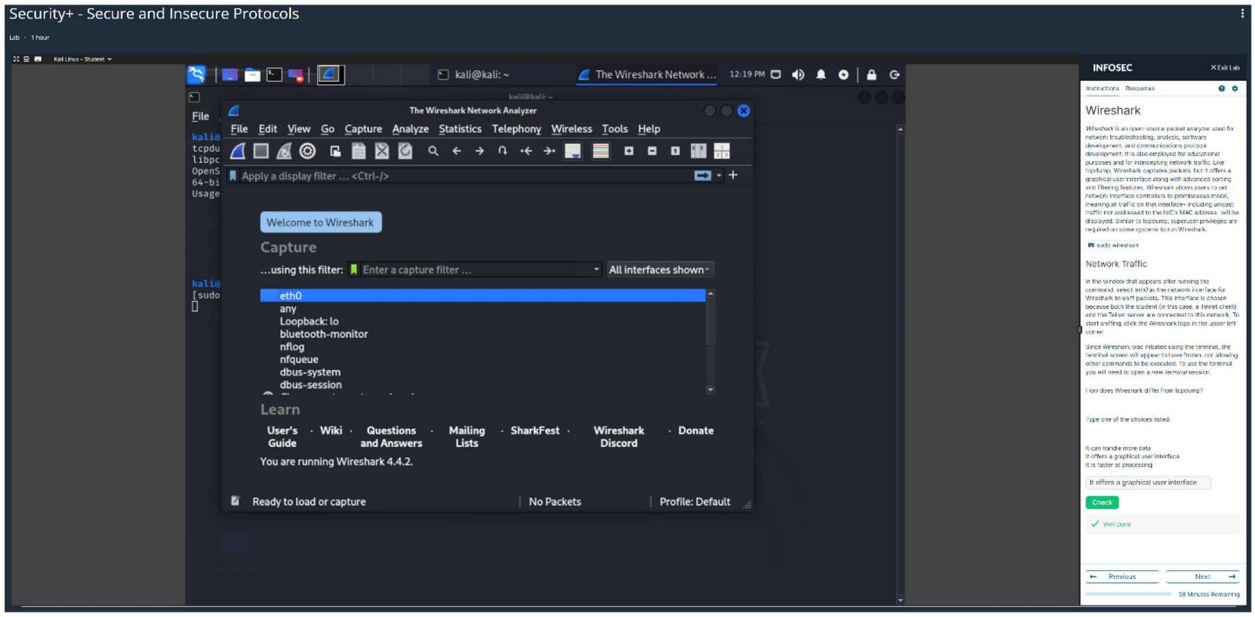

Figure 8: Starting Wireshark

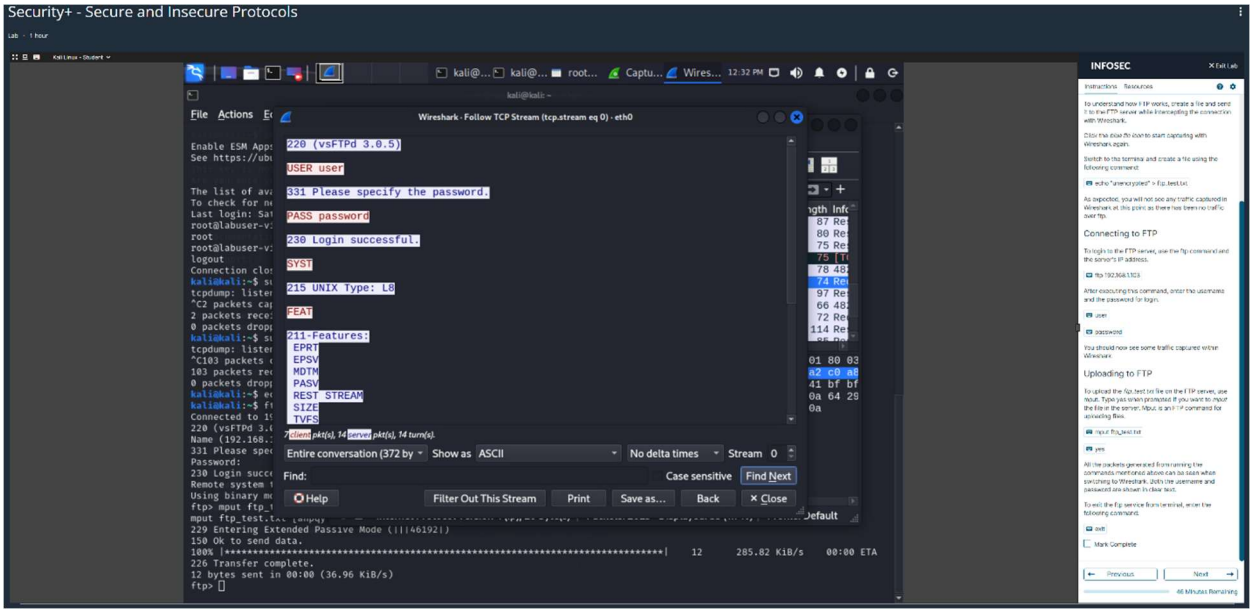

Figure 9: Plaintext Credentials Captured

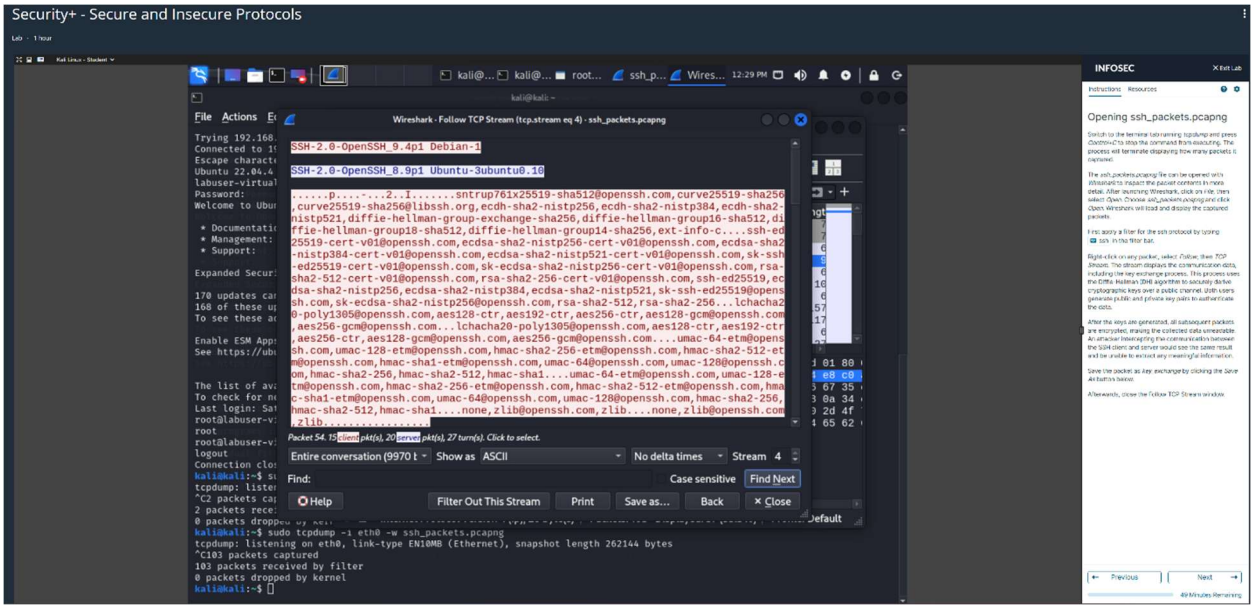

Figure 10: Data-in-Transit Encrypted Credentials

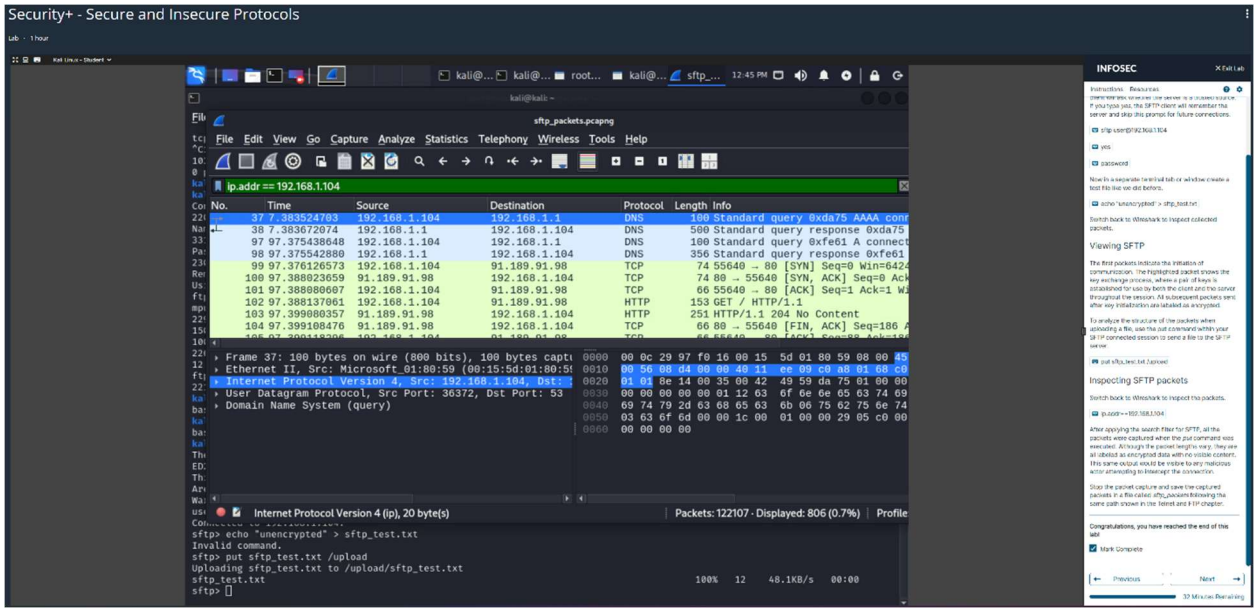

Figure 11: Final Secure and Insecure Protocols Screenshot

References

Garn, D. (2024, August 7). *Examine a captured packet using Wireshark*. Search Networking. https://www.techtarget.com/searchnetworking/tutorial/Examine-a-captured-packet-using-Wireshark

OWASP. (2022). *OWASP Top 10:2021*. Retrieved July 19, 2025, from https://owasp.org/Top10/A03_2021-Injection/

PortSwigger. (n.d.). *What is DOM-based XSS (cross-site scripting)?* Retrieved July 19, 2025, from https://portswigger.net/web-security/cross-site-scripting/dom-based

UK Cyber Security Ltd. (2023, July 11). *What makes Telnet vulnerable?* UK Cyber Security Group Ltd. https://www.ukcybersecurity.co.uk/blog/news-advice/what-makes-telnet-vulnerable/