Lab 7 Report

Jason Tolbert

The Pennsylvania State University

IST 894-001: Capstone Experience

Dr. Michael Bartolacci, Instructor

Sunday, March 9th

Table of Contents

General Overview

This lab guides demonstrates to participants several basic network and web application vulnerabilities.

The lab begins by introducing participants to packet analysis. Packet analysis, broadly speaking, is the interception and examination of data that travels across a computer network. Packet analysis allows security professionals to monitor traffic patterns and detect anomalies that might be indicative of a cyberattack (Joseph et al., 2024). Users use Wireshark (a popular graphical packet analysis program) and tcpdump (a popular command-line program for the same) to capture, analyze, and understand the contents of network packets traveling to and from their machine.

The lab continues by teaching participants of the dangers of insecure network protocols, namely Telnet (a protocol for remotely connecting to one computer's command line from another) and FTP (a protocol for wirelessly transferring files between two computers). Telnet and FTP are insecure because they are unencrypted. Unencrypted traffic is far easier for users of packet analysis tools — like Wireshark — to analyze than encrypted traffic (Sikos, 2020). This, as participants learn, is especially dangerous when sensitive information is part of that traffic. Both Telnet and FTP typically require the user to provide credentials before connecting to remote machine. Participants use preset credentials to establish authenticated connections over both protocols — only to check Wireshark and

see that their credentials have been captured in plain text. (It's worth noting that the insecurity of Telnet and FTP isn't much of a concern these days; both protocols have been obsolesced by more secure successors (Nixon & Devaraj, 2016) and are rarely used in modern computing environments.)

The lab then pivots from network security to web app security. Participants are taught about cross-site scripting (XSS). XSS is a vulnerability which allows client-side scripts to be injected into web pages and run on the machines of their visitors. Specifically, participants learn of three types of XSS: reflected XSS, where a website accepts input from a user and renders it back to them in an unsafe way; stored XSS, where a malicious script is persistently stored somewhere on the target server and is run on the user's machine whenever their browser requests information from that location on the server; and DOM-based XSS, where a malicious script is injected through vulnerabilities in the website's JavaScript code (Drissi, 2024).

Technical Overview

This lab provides participants hands-on experience with network traffic analysis and teaches them of the risks of insecure network protocols and common web vulnerabilities.

The lab begins by introducing participants to packet sniffing and traffic analysis. Most of this segment is focused on Wireshark, but participants also get exposure to tcpdump. Participants capture traffic as it goes between their own machine and a remote one and analyze their contents.

After being introduced to Wireshark, participants use it to explore firsthand the dangers of using outdated, unencrypted, network protocols — namely, Telnet and FTP. After establishing and closing a Telnet connection with a remote machine, participants return to Wireshark to see the packets captured during the connection. Examination reveals to them that their Telnet credentials were exposed in plain text. Participants then to do a similar exercise with FTP. Both Telnet and FTP have long been obsolesced by the more secure SSH and SFTP. This is also touched on in the lab — participants open a premade Wireshark capture of an SSH connection and see for themselves that unlike the packets from the Telnet connection, the SSH packets are encrypted, and no credentials are plainly exposed.

The lab then pivots to web app vulnerabilities. Participants explore vulnerabilities on both the client side and server side. All three kinds of cross-site scripting — stored, reflected,

and DOM-based — are covered. Though participants never exploit XSS to run any actual malicious code, they do use it to perform relatively harmless actions — like triggering JavaScript alerts — that allow them to get an idea of what XSS could potentially be used for in the hands of a real attacker. In addition to XSS, the lab has participants explore SQL injection and directory traversal in the hopes of stressing to them the importance of sanitizing user inputs.

References

Drissi, S. Z. (2024). *Cross-site scripting (XSS) attacks and penetration testing*.

Joseph, G., Osamor, J., & Olajide, F. (2024). A systematic review of network packet sniffing tools for enhancing cybersecurity in business applications. *International Journal of Intelligent Computing Research*, *15*(1).

Nixon, J. S., & Devaraj, D. (2016). *Vulnerability study of remote protocols: Telnet & SSH and a proposed method for secure communication using telnet over IPSec in windows platform*.

Sikos, L. F. (2020). Packet analysis for network forensics: A comprehensive survey. *Forensic Science International: Digital Investigation*, *32*, 200892. https://doi.org/10.1016/j.fsidi.2019.200892

Screenshots

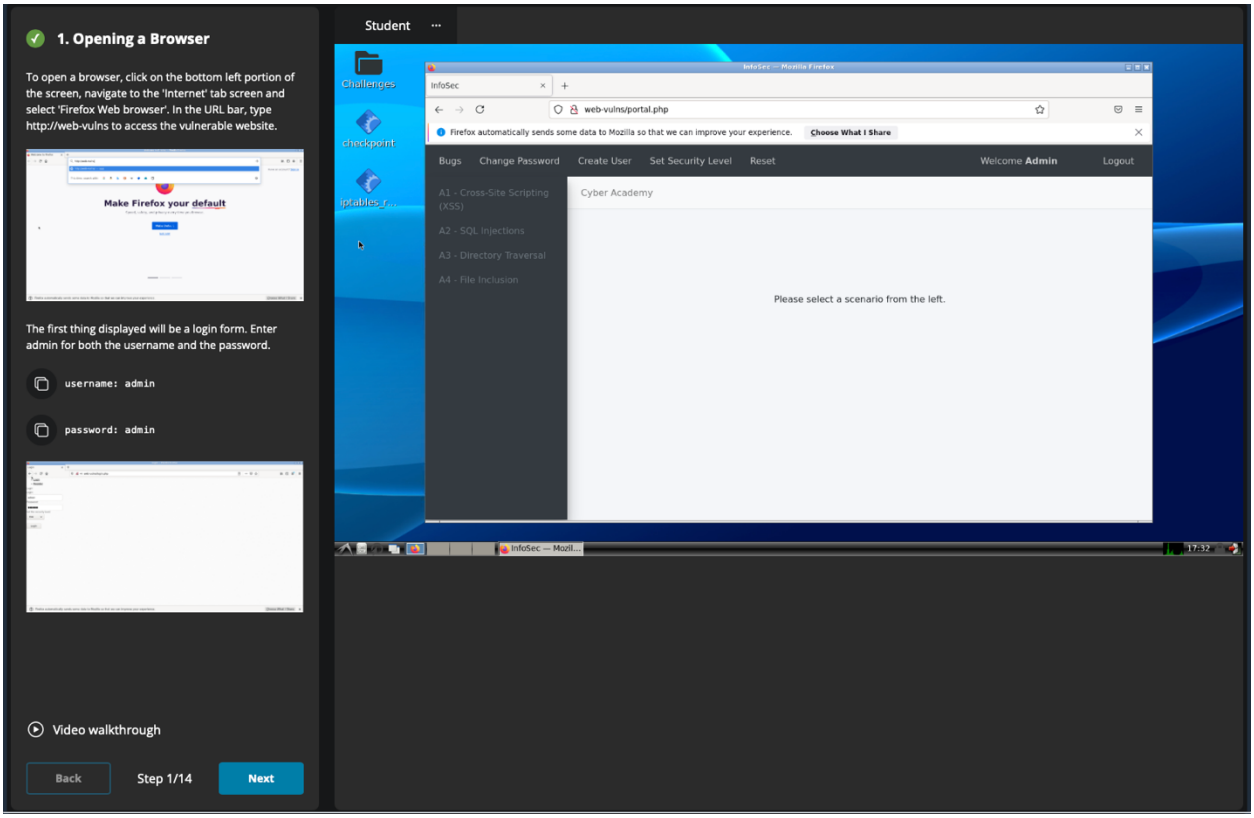

Figure 1. The web-vulns main page.

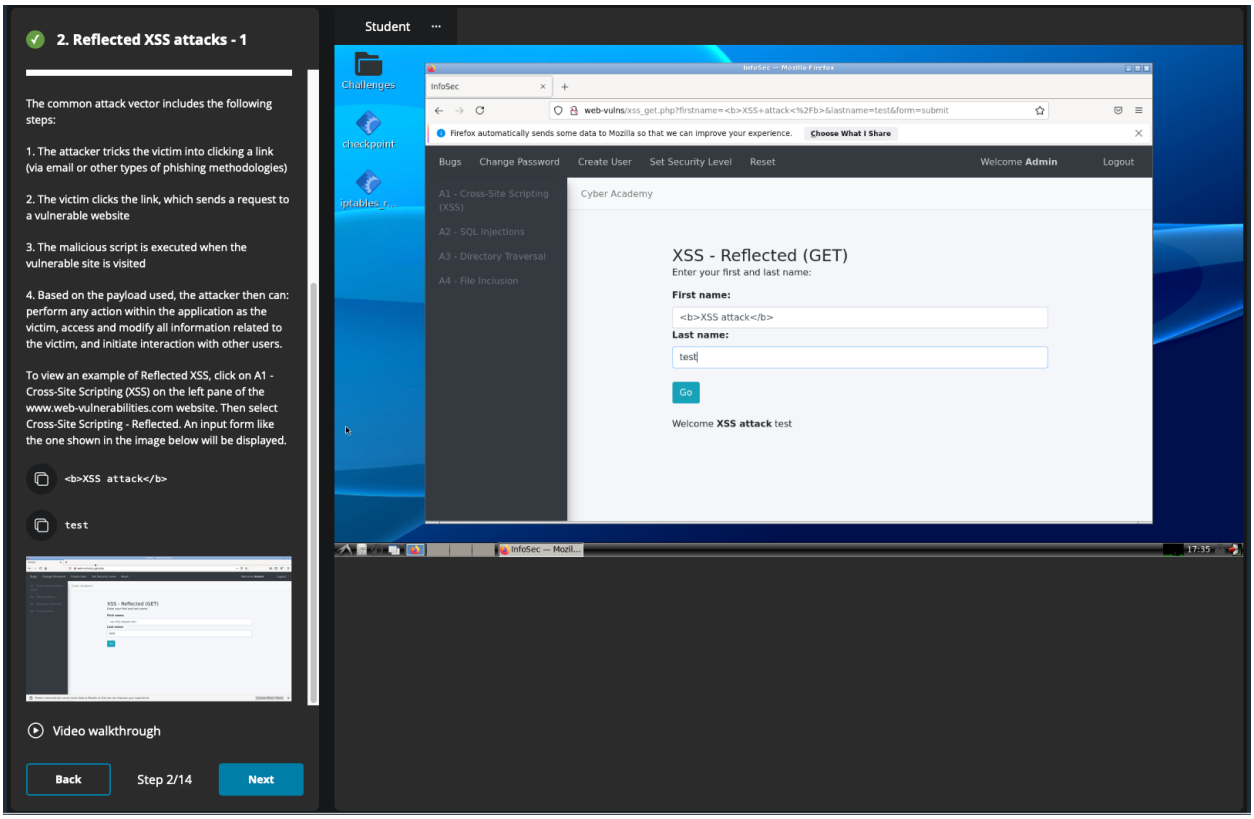

Figure 2. An example of a reflected XSS attack.

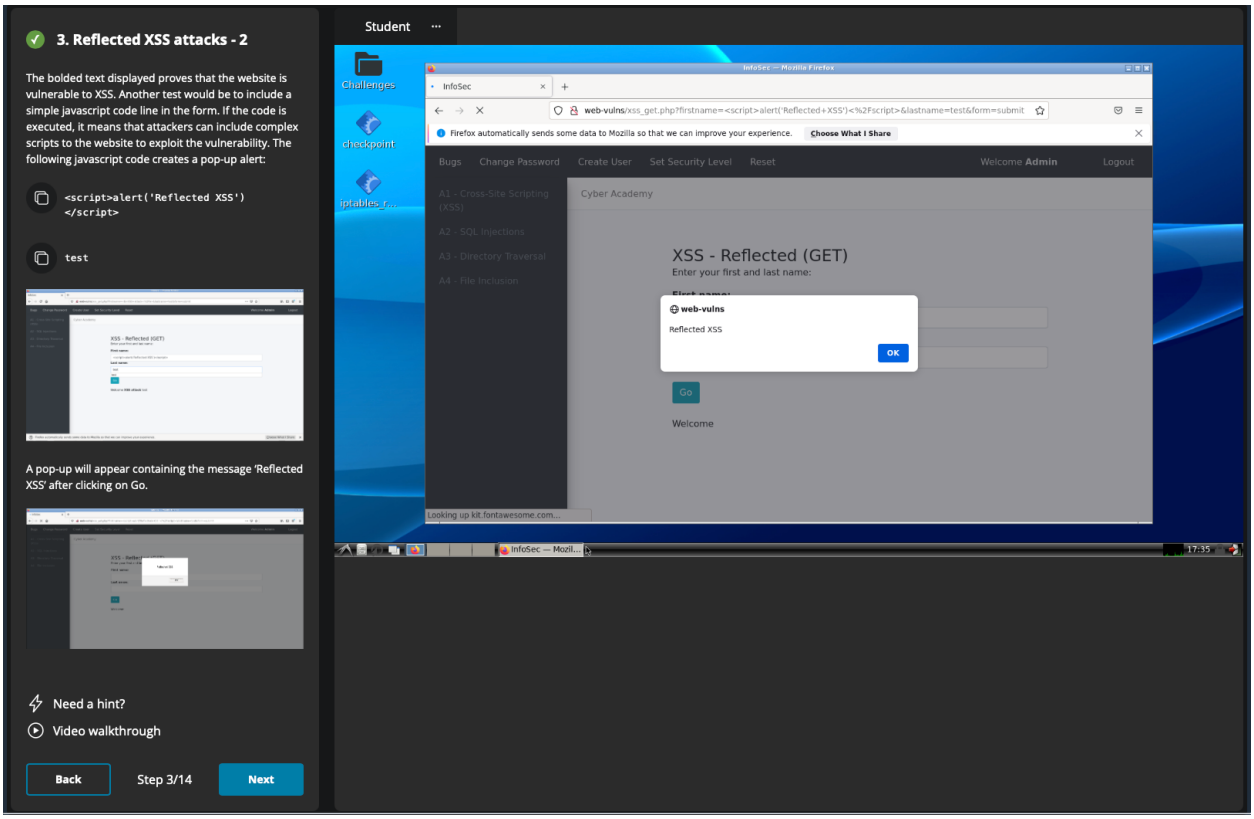

Figure 3. A second example of a reflected XSS attack (the "first name" was JavaScript code to display the pictured alert).

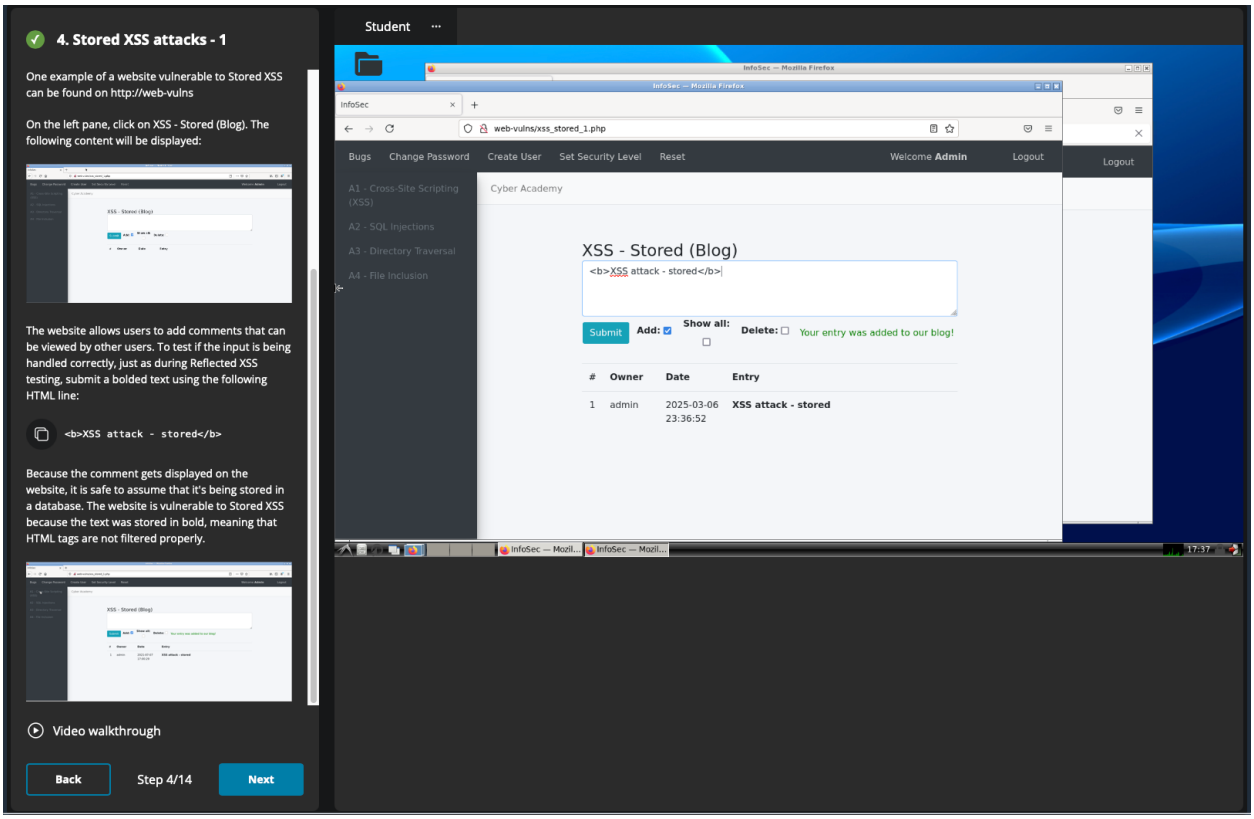

Figure 4. An example of a stored XSS attack.

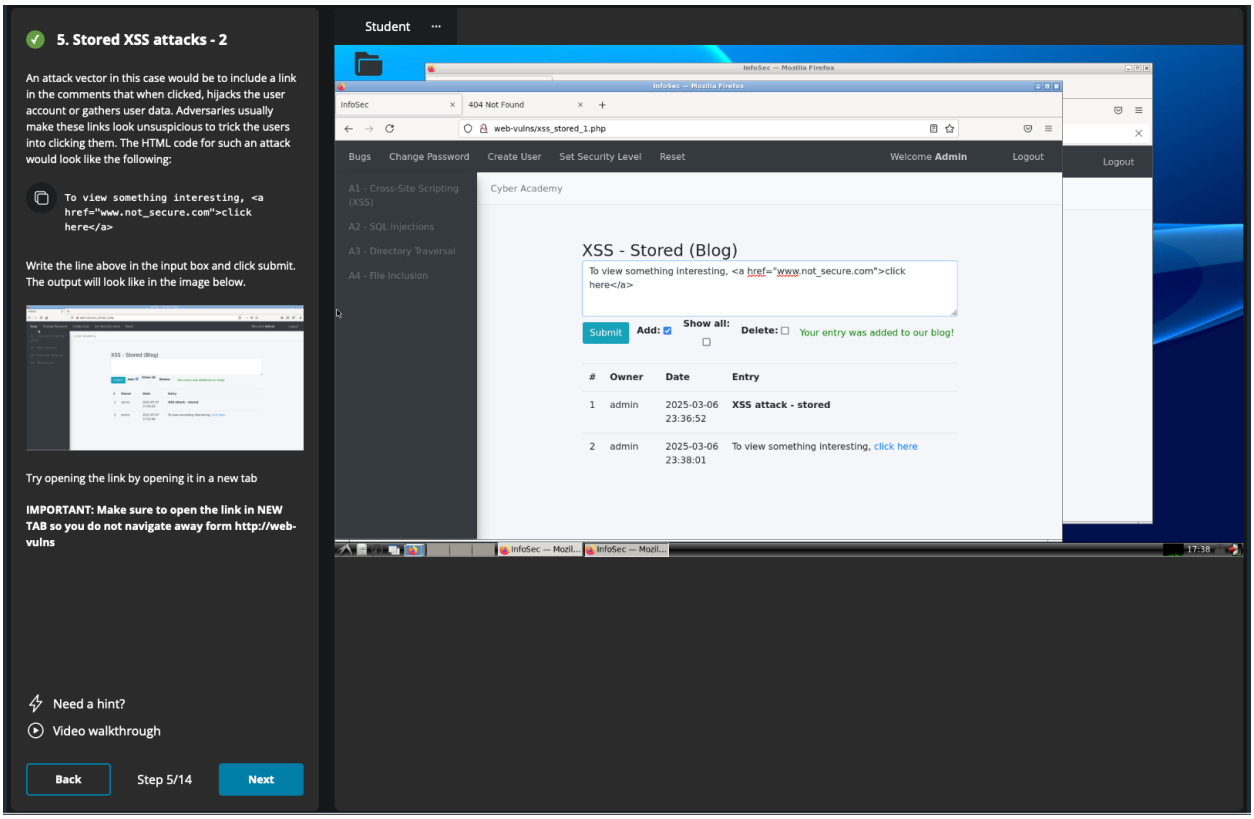

Figure 5. A second example of a stored XSS attack.

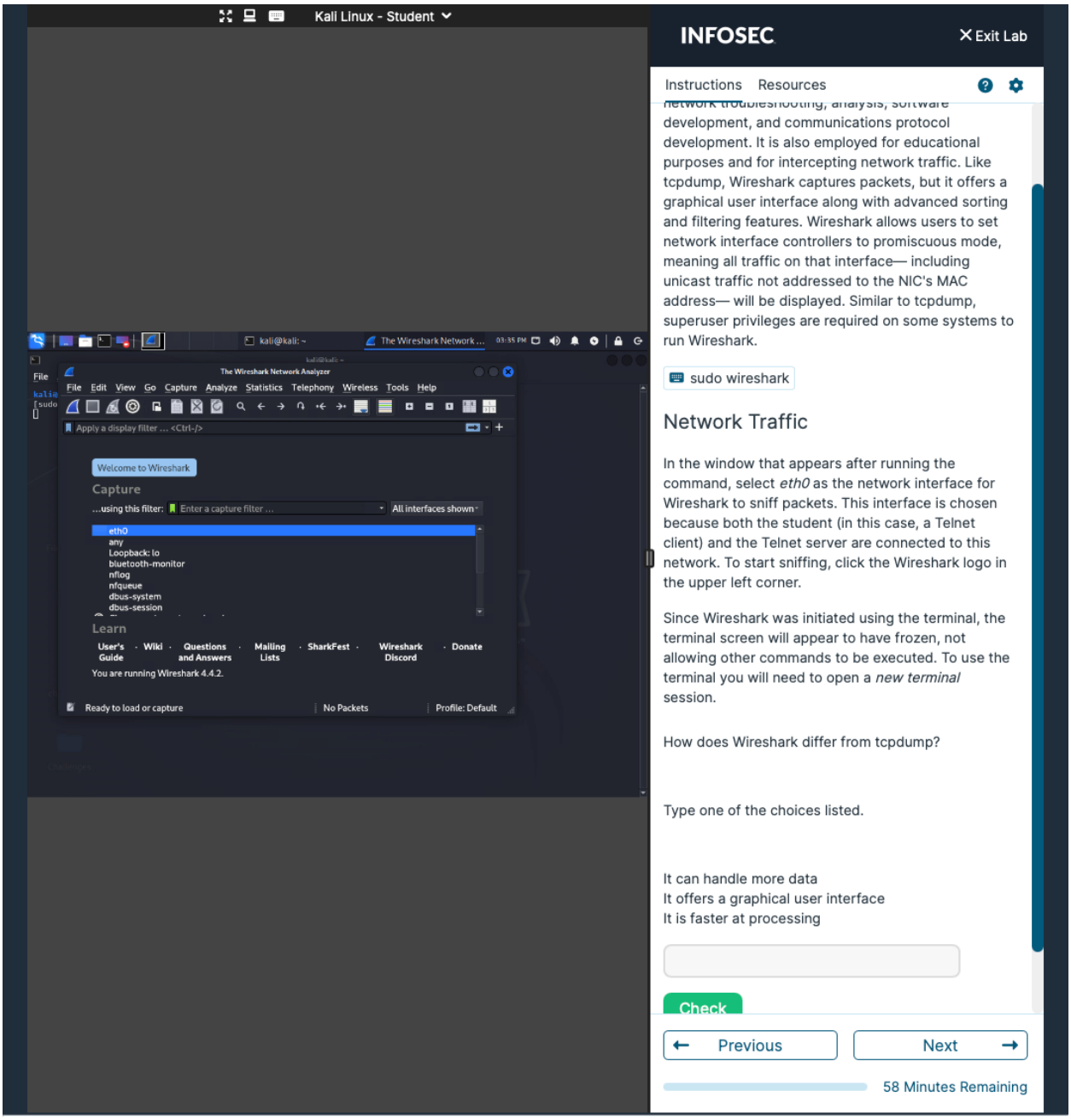

Figure 6. The Wireshark interface.

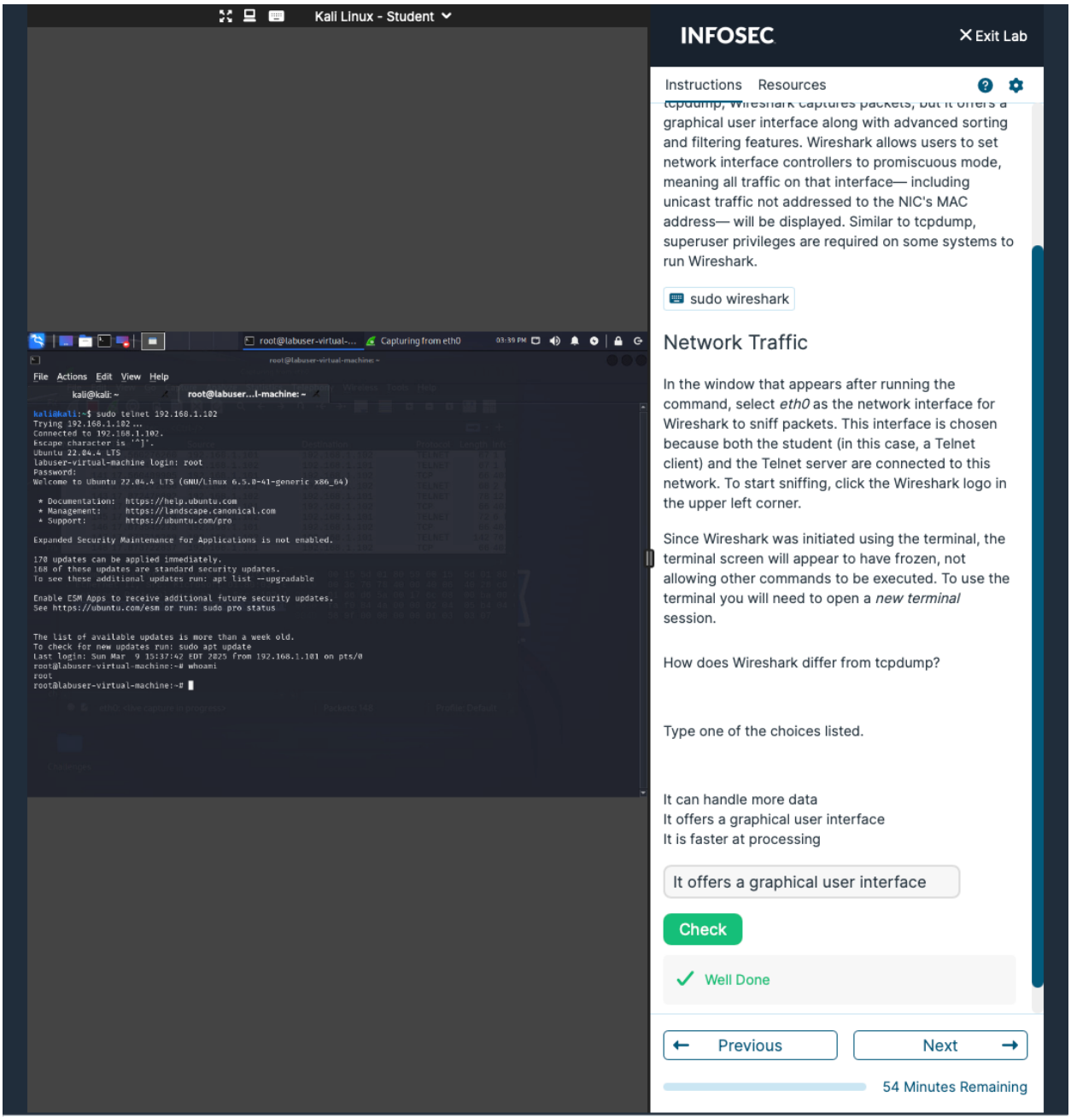

Figure 7. Accessing a remote machine over Telnet.

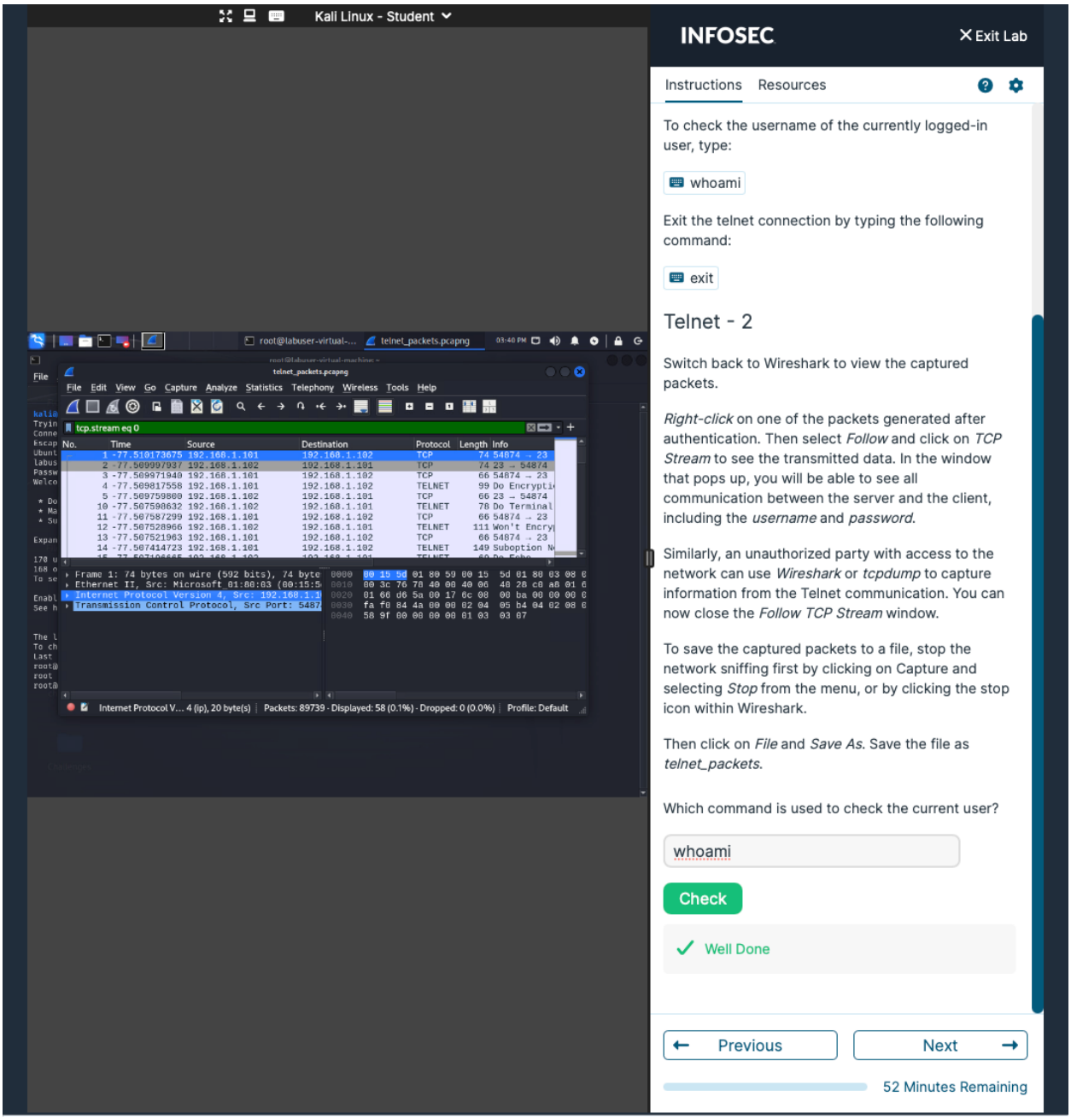

Figure 8. Packets captured from the Telnet connection.

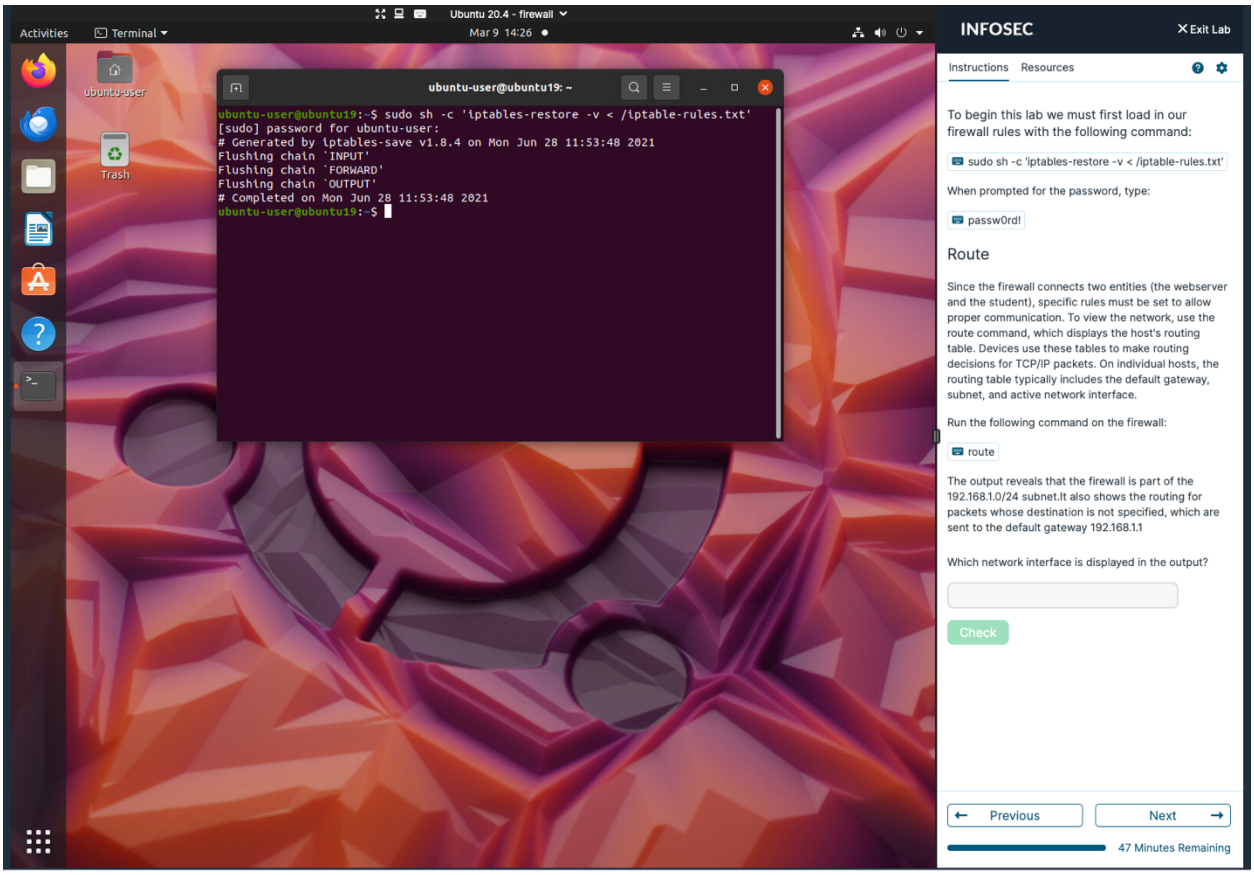

Figure 9. Restoring iptables rules from a text file.

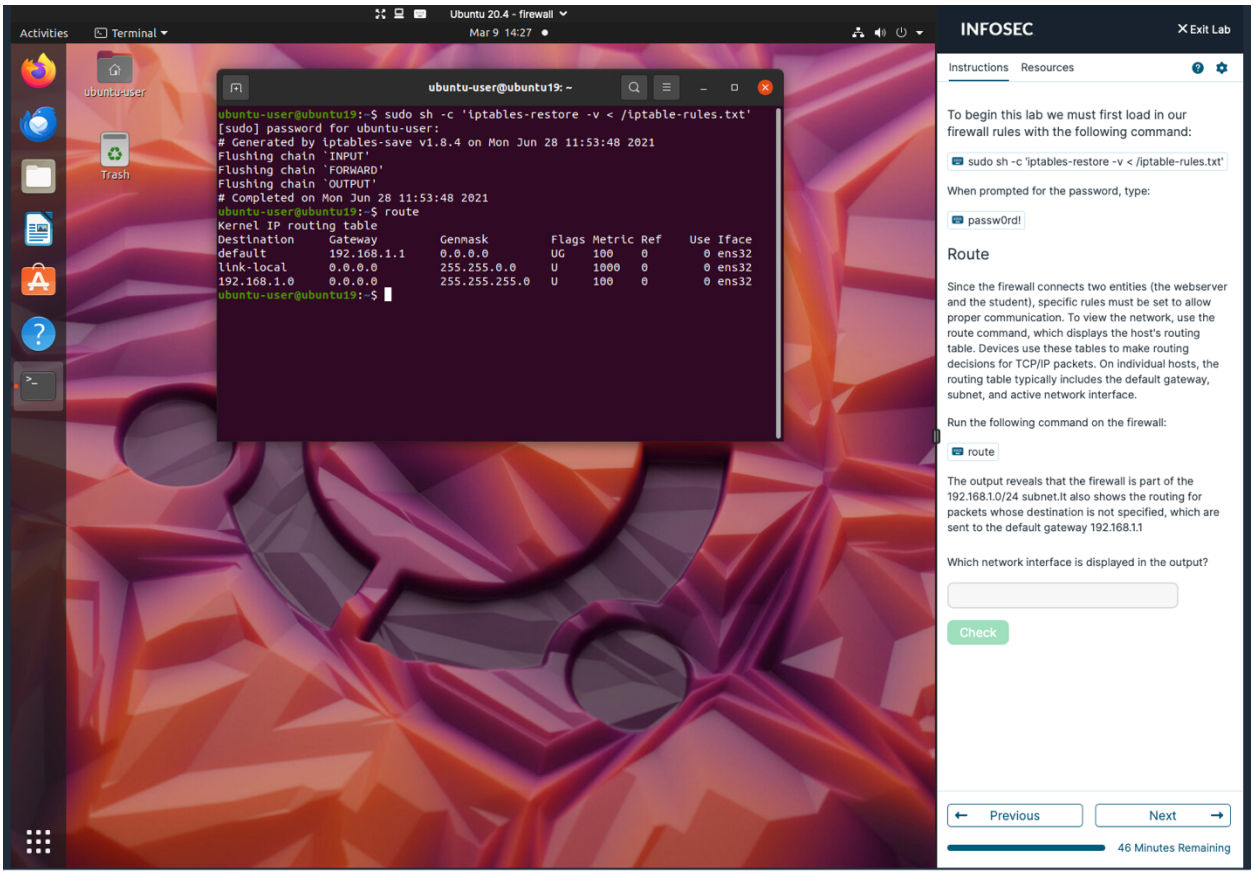

Figure 10. The output of the "route" command.