Lab 2.pdf

Sonal Chandrakant Hasbi

***Abstract*—In this lab, I explored how websites communicate and how attackers can take advantage of weak spots in that communication. Using tools like Burp Suite, browser developer tools, and WebGoat, I learned how to intercept and analyze HTTP traffic, understand how cookies and sessions work, and spot common security issues—like poor encryption and broken access controls. This hands-on experience helped me connect theory to real-world web security problems and gave me a better understanding of how to protect web applications.**

I. Introduction

In this lab, I learned about the principles of HTTP communication and the security considerations that occur in web applications. The purpose was to obtain practical expertise with HTTP headers, sessions, cookies, proxies, and cryptography using a variety of tools, including Burp Suite Community Edition, Firefox Developer Tools, and WebGoat. Through actual exercises, I gained a better knowledge of how attackers exploit web application flaws and how security professionals detect and mitigate these risks.

II. Tools Used

Burp Suite Community Edition

Burp Suite was used as the main proxy tool to intercept, analyze, and manipulate HTTP requests between the browser and WebGoat. After installing Burp Suite, I configured FoxyProxy in Firefox to route traffic through the Burp listener running on 127.0.0.1:8081

FoxyProxy (Firefox Add-on)

This add-on helped automate proxy switching in the browser, enabling seamless integration with Burp Suite. It allowed me to selectively route traffic through Burp without affecting all browser traffic.[3]

Firefox Developer Tools

The browser's built-in tools were used to inspect HTTP requests and responses, observe cookies, and track session behavior. This was especially helpful during the Developer Tools and CIA Triad sections of WebGoat.

WebGoat Platform

A deliberately insecure application used to demonstrate common web vulnerabilities. I completed the following modules[2]:

- **HTTP Basics** – Explored GET/POST methods, status codes, and headers.
- **HTTP Proxies** – Used Burp Suite to intercept and modify requests.
- **Developer Tools** – Investigated cookies, headers, and storage.
- **CIA Triad** – Answered questions based on observed data.
- **Crypto Basics (A2 - Parts 1–4)** – Cracked weak hashes using public tools like 10015 Tools [1] (fig. 1)

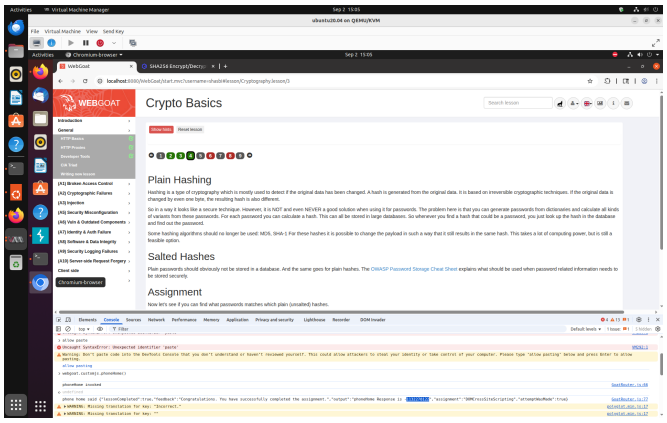

Fig. 1: Tasks completed on Webgoat

Online Hash Cracking Tools

Used platforms like 10015 Tools to reverse engineer MD5 and SHA-1 hashes in the cryptographic exercises.

III. Procedure Summary

- Launched WebGoat and accessed lessons through Firefox with the FoxyProxy extension (Fig. 2).
- Configured Burp Suite to act as an intercepting proxy and successfully captured HTTP traffic (Fig. 3)
- Identified and modified request parameters in Burp, e.g., changing form inputs or headers.
- Used the browser developer tools to validate how headers and cookies were handled.
- Completed "The Quiz" in the HTTP Basics section after analyzing intercepted traffic.(Fig. 4)
- Used Burp's Repeater feature to resend manipulated requests and observe responses.

IV. Findings / Discussion

- Burp Suite provides a powerful interface for intercepting and replaying requests, giving clear visibility into how data flows through a web application. Its Proxy, Repeater, and Decoder tools were especially useful.
- HTTP methods such as GET and POST have distinct use cases. The exercises emphasized how GET parameters

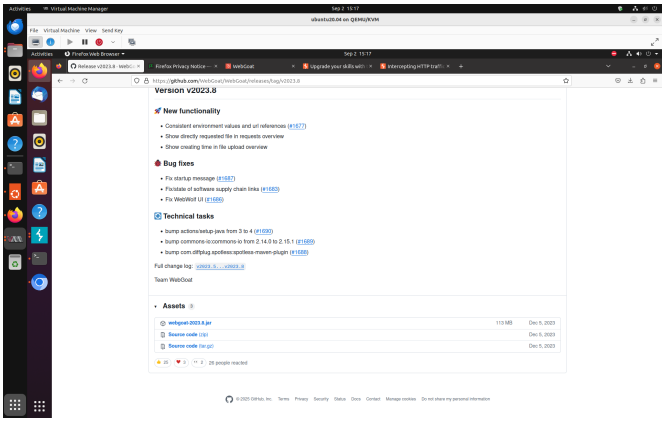

Fig. 2: Launched WebGoat

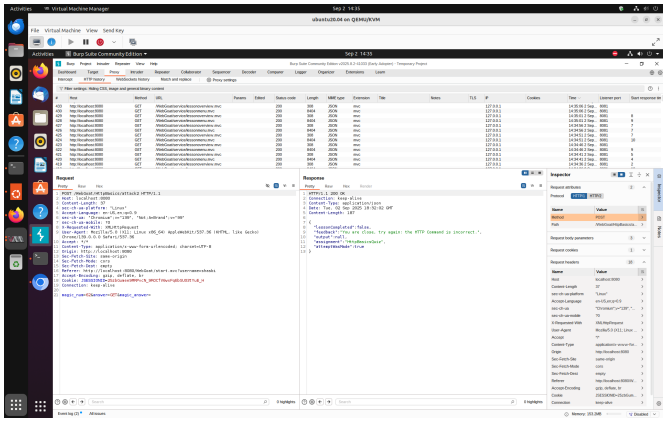

Fig. 4: Found Magic Number

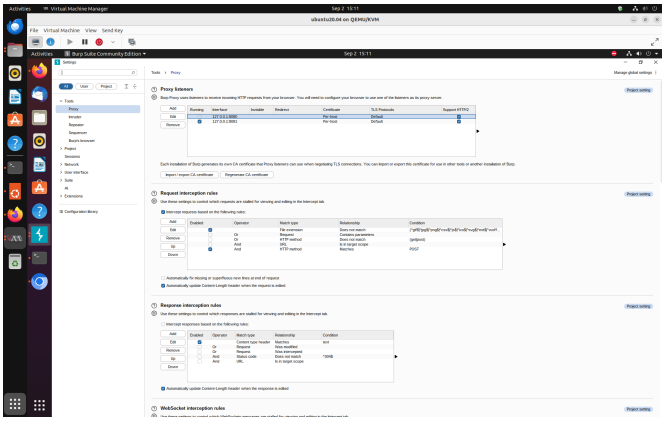

Fig. 3: Configuration of Burp Suite

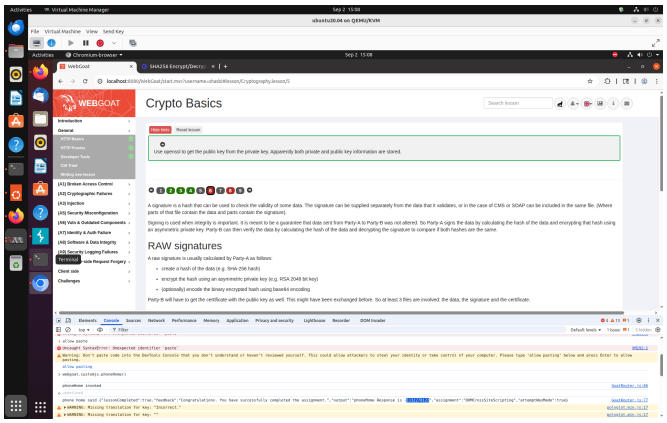

Fig. 5: Completed all Tasks

are exposed in URLs while POST data is not, which has implications for security[5].

- Cookies and Sessions can be manipulated or stolen if not secured with proper flags (Secure, HttpOnly, SameSite). I was able to see session cookies transmitted over unencrypted connections during exercises, highlighting risks of session hijacking[6].
- Developer Tools showed how web storage can store sensitive data insecurely[7].
- Cryptographic Failures were easy to exploit. In some exercises, plaintext passwords were hashed using outdated algorithms like MD5 or SHA-1, which could be reversed in seconds using online tools.

V. Security Implications

These exercises demonstrated how developers frequently forget critical security configurations. Even minor HTTP misconfigurations can result in data leakage, session hijacking, and broken access controls. Tools such as Burp Suite are critical for detecting these flaws before they are exploited in the wild.

VI. Conclusion

This lab provided valuable hands-on exposure to web application security tools and techniques. I gained practical experience in HTTP analysis, proxy configuration, and cryptographic evaluation. The use of Burp Suite for intercepting traffic and WebGoat for practicing vulnerabilities offered a realistic view into attacker behavior and defense strategies. Understanding these tools and concepts is vital for any future cybersecurity role, especially in penetration testing and application security.

VII. Answers to CIA Triad Questions

1) Question 1 - Solution 3: By stealing a database where names and emails are stored and uploading it to a website.
2) Question 2 - Solution 1: By changing the names and emails of one or more users stored in a database.
3) Question 3 - Solution 4: By launching a denial of service attack on the servers.
4) Question 4 - Solution 4: Solution 2: The system's security is compromised even if only one goal is harmed.

References

[1] https://10015.io/tools/sha256-encrypt-decrypt#google_vignette
[2] https://portswigger.net/burp/releases#community
[3] https://owasp.org/www-project-webgoat/
[4] https://addons.mozilla.org/en-US/firefox/addon/foxyproxy-standard/
[5] https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/Overview.
[6] https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies
[7] https://developer.mozilla.org/en-US/docs/Tools