Lab 2: Security Tools and HTTP Basics

Shivali Mate

I. Introduction

THIS lab introduced the core concepts of HTTP communication, proxy-based request interception, and cryptographic failures in web applications. The use of Burp Suite and WebGoat helped gain practical experience in analyzing requests and responses, understanding session handling, modifying HTTP traffic, and applying cryptographic decoding techniques. The exercises highlighted common vulnerabilities in web applications and demonstrated how attackers can exploit weaknesses in insecure implementations.

II. Methodology

The Burp Suite Community Edition was installed and its built-in browser was used to avoid additional proxy configuration. After launching WebGoat inside the Burp browser, the assigned exercises were performed covering where tasks involved intercepting traffic, analyzing parameters, or decoding cryptographic data. A Burp Suite window with performing exercise is demonstrated in Fig. 1

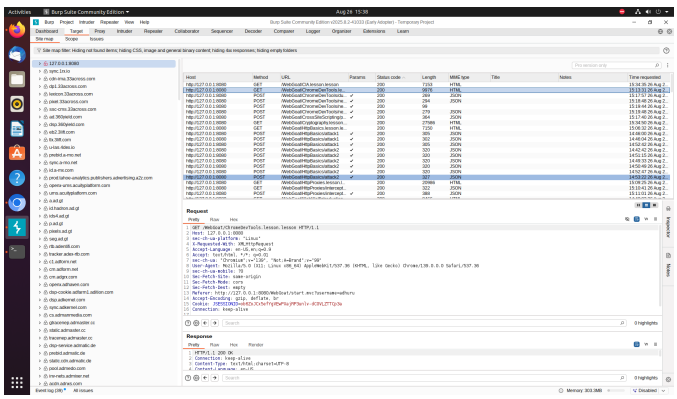

Fig. 1. Burp Suite Interface

A. HTTP Basics

The exercise required identifying whether an HTTP request was a GET or POST command and discovering a “magic number” used in the communication. Burp Suite was used to capture the request and analyze its details. The correct method and value were identified, as shown in Fig. 2.

B. HTTP Proxies

This task demonstrated the interception and modification of HTTP requests. After clicking the submit button on the WebGoat page, the request was intercepted in Burp. The method was changed to GET, a custom header x-request-intercepted:true was added, and the request body was removed.

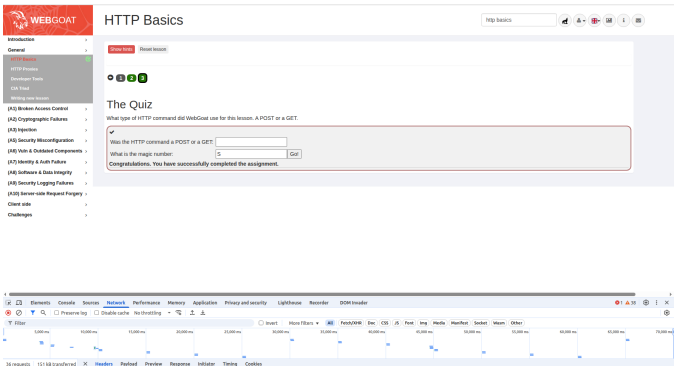

Fig. 2. HTTP Basics

A query string parameter ChangeMe was added with the value Requests are tampered easily. The modified request was forwarded, and the page reflected the tampered value as in Fig. 3.

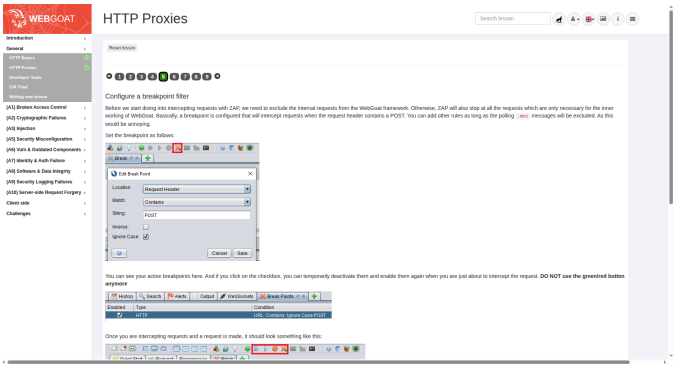

Fig. 3. HTTP Proxies Request Modification

C. Developer Tools

This exercise involved working with the Network tab in browser developer tools. The goal was to locate a specific HTTP request containing the randomized field ’networkNum:’. The correct number was identified by inspecting the network traffic in the developer console. Fig. 4 documents the result.

D. CIA Triad

The CIA triad—Confidentiality, Integrity, and Availability defines the core goals of information security: protecting data from unauthorized access, ensuring its accuracy, and keeping it accessible when needed. It highlighs its role as a foundational model of information security. Four multiple-choice questions from the exercise are answered based on the principles of CIA.

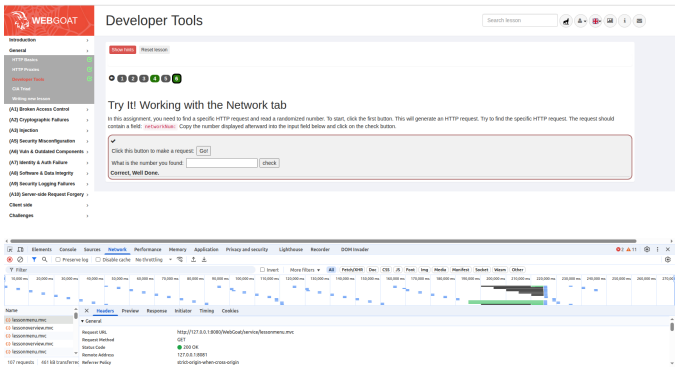

Fig. 4. Developer Tools Network Analysis

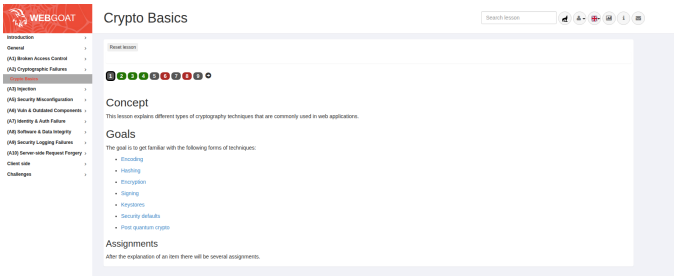

Fig. 5. Cryptographic Failures Solutions

1) Question 1: How could an intruder harm the security goal of confidentiality?
   Solution 3: Stealing a database where names and emails are stored and uploading it.
2) Question 2: How could an intruder harm the security goal of integrity?
   Solution 1: Changing the names and emails of users in the database.
3) Question 3: How could an intruder harm the security goal of availability?
   Solution 4: Launching a denial of service attack on the servers.
4) Question 4: What happens if at least one of the CIA security goals is harmed?
   Solution 2: The system's security is compromised even if only one goal is harmed.

E. Cryptographic Failures

This section introduced common cryptographic issues. The first four exercises were completed using online hash crackers and decoders:

- **Base64 decoding:**
  It is a method that converts binary data into text using 64 ASCII characters. The intercepted header was decoded to reveal the username and password.
- **XOR decoding:**
  It hides data by applying the XOR logical operation between the data and a key. The encoded password was successfully de-obfuscated.
- **MD5 and SHA256 cracking:**
  They are one-way cryptographic functions that generate fixed-length digests of input data. Given hash values were reversed using publicly available hash-cracking tools.

III. Tools and Learning Experience

Burp Suite was the primary tool used, serving as a proxy to intercept and manipulate HTTP traffic. Browser developer tools provided insights into sessions, cookies, and headers. Online decoders and hash-cracking tools facilitated cryptographic exercises. The lab experience highlighted the ease with which insecure implementations can be exploited, deepening understanding of vulnerabilities in modern web applications.

The lab also reinforced key security concepts:

- **HTTP and cookies:** Essential to application communication and session management.
- **Cryptographic failures:** Common weaknesses that can expose credentials.
- **Traffic interception:** Demonstrated how attackers can manipulate requests.

While effective, tools like Burp Suite and WebGoat are limited to educational environments and simulated scenarios. Improving realism in exercises, expanding coverage of modern encryption schemes, and integrating real-world attack case studies could enhance their applicability.

IV. Findings and Discussion

The lab demonstrated that fundamental aspects of HTTP traffic, when not properly secured, can be easily manipulated. Cookies and sessions were shown to be critical for maintaining security, while cryptographic failures highlighted the risks of weak or outdated algorithms. The hands-on tasks with Burp Suite provided practical reinforcement of theoretical concepts, bridging the gap between academic study and industry relevance.

V. Conclusion

This lab successfully introduced key concepts in web communication and security testing. By using Burp Suite, browser developer tools, and online hash decoders, vulnerabilities in HTTP and cryptographic practices were explored. The exercises highlighted the importance of secure coding, robust encryption, and strong traffic inspection methods in safeguarding modern web applications.

References

[1] https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/How_the_Web_works
[2] https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview
[3] https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies
[4] https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers
[5] https://developer.mozilla.org/en-US/docs/Web/HTTP/Session
[6] https://www.w3schools.com/tags/ref_urlencode.ASP
[7] https://www.urlencoder.org/
[8] https://addons.mozilla.org/en-US/firefox/addon/foxyproxy-standard/
[9] https://portswigger.net/burp/documentation/desktop/getting-started