Lab 8: Authentication

Thangamuthu Balaji

I. Introduction

Authentication mechanisms play a critical role in ensuring the security of applications by verifying the identity of users and protecting sensitive data from unauthorized access. In this lab, like weak passwords, issues with resetting passwords, multi-level logins, and token-based systems like JSON Web Tokens (JWTs). We used WebGoat and WebWolf to test these issues in real-life situations. This helped us find weaknesses, learn how they can be taken advantage of, and see what kind of damage they could cause. By intercepting and changing network traffic, checking the source code, and decoding tokens, we got hands-on experience with the tools and techniques that hackers might use to attack weak authentication systems. The main goal of this lab was to help us understand these vulnerabilities better and highlight how important it is to use secure practices to reduce these risks.

II. 1. Authentication Flaws

a. Password Strength

The task was all about checking how strong different passwords are and figuring out how long it would take to break them. We used a special tool to analyze the passwords, looking at things like how complicated they were, how long they were, and the variety of characters used. For instance, a really easy password like 123456 could be cracked almost instantly because it's super common and easy to guess. On the other hand, a more complicated password like My1stPassword!:Redd, which has a mix of letters, numbers, and symbols, would take an incredibly long time to crack because there are so many possible combinations. This project showed just how important it is to have strong password rules, focusing on making them complex and long to keep our information safe from attacks.

b. Forgot Password

This task demonstrated how poorly implemented password recovery mechanisms can be exploited. In the scenario, the password reset functionality allowed access by answering a simple security question: ”What is your favorite color?” with the answer red. The simplicity and predictability of the question made it trivial to bypass the mechanism. Additionally, the absence of account lockout after multiple incorrect attempts exposed a major flaw. The exercise highlighted the risks of relying on predictable or guessable security questions and the necessity for stronger recovery mechanisms like multi-factor authentication.

c. Multi Level Login 1

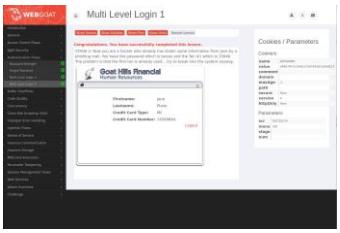
This task examined vulnerabilities in a two-step login process involving Transaction Authentication Numbers (TANs). After entering valid credentials and a TAN, the intercepted network traffic revealed that the TAN was being sent in plain text. By manipulating the intercepted parameters, a previously used TAN was resubmitted, allowing unauthorized access. This demonstrated the risks of inadequate validation on the server-side and insecure transmission of sensitive information. The task illustrated the importance of implementing secure and robust mechanisms for multi-level authentication.

d. Multi Level Login 2

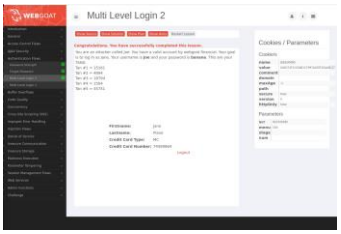
In this case, an additional layer of multi-level authentication was compromised through an attack on the TAN system. The individual initiated the process by logging in as Joe with legitimate credentials and subsequently intercepted the data during the TAN submission phase. By altering the intercepted request to replace the username from Joe to Jane, the attacker gained unauthorized access to Jane’s account.

(a) Password Strength

(b) Forgot Password



(c) Multi Level Login 1



(d) Multi Level Login 2

III. 2. Identity and Authentication Failures

a. Authentication Bypass

This showed how some security systems can be tricked by taking advantage of weak spots in how sessions or inputs are checked. By looking into session tokens, URL parameters, and headers, we found problems in the way authentication works. For example, changing certain tokens or parameters allowed people to get into areas they shouldn't have access to. This highlighted how crucial it is to check input data carefully and manage sessions securely to stop attackers from using these weaknesses.

b. Decoding a JWT Token

In this part, i looked at a JSON Web Token (JWT) to find out more about the user. We used a base64 decoder to decode the token and see its payload, which had information like the username. This showed us how JWTs function by breaking down information into three parts: the header, the payload, and the signature. But if JWTs aren't encoded or signed correctly, they can leak private information or be changed, which can create security problems. This step highlighted how crucial it is to use safe methods and algorithms when creating tokens.

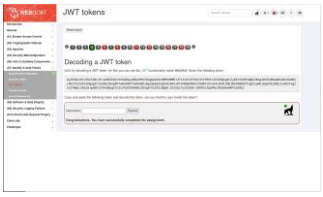
c. JWT Signing

When someone changes the information inside a token and then signs it again with a key that is already known, they can create a fake token that seems real. This showed that if the signing keys are weak or easy to guess, it can put the whole security system at risk. The project highlighted how crucial it is to use strong cryptographic keys and methods for signing JWTs to stop hackers from making fake tokens.

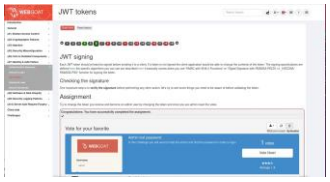
d. Code Review

In this step, we looked at the source code to find any weaknesses in how JWTs were managed. We discovered some problems, like not checking token signatures properly and mishandling claims. Because of these issues, someone could trick the system by creating fake tokens or changing them without being noticed. This showed us how important it is to use secure coding methods and to carefully review code to spot and fix problems related to authentication.

(a) Authentication Bypass



(b) Decoding a JWT Token



(c) JWT Signing

(d) Code Review

IV. 3. Password Reset

a. Email Functionality with WebWolf

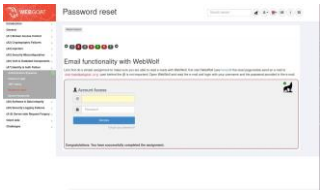
In this method the user checked how the email password reset system worked with WebWolf. They entered an email address on the reset page, which then sent out a password reset email. This email included a new password that could be used to log in. This task showed the dangers of depending only on email for resetting passwords because if someone intercepts or hacks the email, they could get into the account without permission. It highlighted the importance of using extra verification methods, like temporary codes or multi-factor authentication, to keep accounts safer.

b. Security Questions

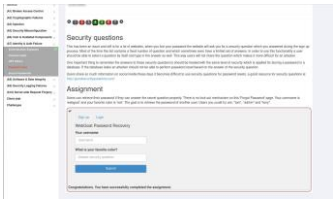
This part showed how weak security questions can be in password recovery systems. For example, if someone answered a question like "What is your favorite color?" with the right answer (red), they could get access. This example proved that easy questions and answers can be taken advantage of to get around security checks. It pointed out that we should stop using security questions and look for better ways to recover accounts that use stronger methods to verify identity.

c. The Problem with Security Questions

The last part of the password reset section looked into the bigger issues with security questions. We talked about weak questions like "What is your favorite animal?" and how people can easily guess or find out the answers. This showed how tough it is to come up with "perfect" security questions that are easy to remember but hard for others to figure out.



(a) Email Functionality with WebWolf



(b) Security Questions

(c) The Problem with Security Questions

V. 4. JWT Tokens Step 10

This task involved analyzing source code and reviewing a related video to identify vulnerabilities in JWT handling. The focus was on understanding how weak or improperly validated tokens could be exploited. For instance, using the ”none” algorithm for token signing allows attackers to bypass validation entirely. The task demonstrated the importance of

enforcing strict validation, using strong signing algorithms, and avoiding insecure practices in token management.

VI. Conclusion

This lab showed how important it is to have secure ways to log in and pointed out the dangers of common mistakes in authentication systems. By looking at how strong passwords are, using easy-to-guess security questions, messing with tokens, and getting around authentication, the activities revealed the different ways hackers can break into systems. These exercises stressed the need for strong password rules, good session management, proper token checks, and safe coding methods. They also highlighted the problems with old recovery methods, like security questions, and the need for better options like multi-factor authentication. By spotting these weaknesses and figuring out why they happen, this lab gave useful tips that are crucial for creating and keeping secure login systems in real-life situations.

References

[1] https://docs.cycubix.com/application-security-series/web-application-security-essentials/.
[2] https://spencerdodd.github.io/2017/01/25/webgoat_part_4/.
[3] https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html.
[4] https://cheatsheetseries.owasp.org.
[5] https://security.org/how-secure-is-my-password/.
[6] Parthasarathi, S. (2018). The Importance of Strong Passwords in Cyber-security. Journal of Cyber Defense, 10(2), 45-53.