Lab 9

Venkata Diwakar Reddy Kashireddy

***Abstract*—In this lab, we learned about authentication flaws in Webgoat 7.1 and the Identity & Auth Failure section in Webgoat 8. We learned about password strength, password reset, and also tried to break into another account. We also got to work with JWT tokens.**

I. Introduction

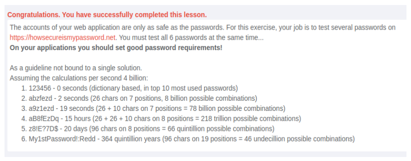
IN this lab, learned about authentication flaws and Identity & Auth Failure and we performed tasks on them. Broken Authentication occurs when attackers successfully exploit vulnerabilities to get control of passwords, keys, session tokens, user account data, and additional information, other details to assume user identities.

II. Tools

• KVM (Kernel-based Virtual Machine): It is a full virtualization solution for Linux. [1]
• WebGoat: A deliberately insecure web application maintained by OWASP designed for teaching web application security concepts. [2]
• Overleaf: It is a collaborative cloud-based LaTeX editor that helps to create documents easily by providing standard formats. [3]
• GitHub: GitHub is an Internet hosting service for software development and version control using Git. [4]
• Red: – IU Research Desktop (RED) is a virtual desktop service for users with accounts on the Carbonate research supercomputer at IU. [5]
• ZAP: OWASP ZAP (Zed Attack Proxy) is an open-source web application security scanner. It is used by those new to application security and professional penetration testers. [6]
• Firefox DevTools: Firefox Developer Tools is a set of web developer tools built into Firefox. It can be accessed to inspect the web page. [7]

III. Authentication Flaws

To safeguard passwords from brute-force attacks, it's critical that your application enforces robust password requirements. To evaluate password robustness, we utilized the security.org website, which calculates the time it might take to crack a given password.

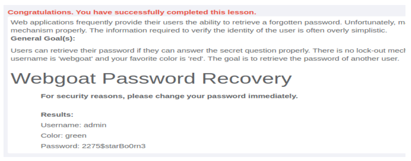


Congratulations. You have successfully completed this lesson.
The accounts of your web application are only as safe as the passwords. For this exercise, your job is to test several passwords on https://howsecureismypassword.net. You must test all 6 passwords at the same time...
On your applications you should set good password requirements!

As a guideline not bound to a single solution.
Assuming the calculations per second 4 billion:
1. 123456 - 0 seconds (dictionary based, in top 10 most used passwords)
2. abzfezd - 2 seconds (26 chars on 7 positions, 8 billion possible combinations)
3. a9z1ezd - 19 seconds (26 + 10 chars on 7 positions = 78 billion possible combinations)
4. aB8fEzDq - 15 hours (26 + 26 + 10 chars on 8 positions = 218 trillion possible combinations)
5. z8!E?7D$ - 20 days (96 chars on 8 positions = 66 quintillion possible combinations)
6. My1stPassword!:Redd - 364 quintillion years (96 chars on 19 positions = 46 undecillion possible combinations)

Fig. 1. Time needed to crack different types of password

Users have the option to recover their passwords by correctly answering a secret question. There is no lock-out mechanism on the 'Forgot Password' page. The objective was to acquire another user's password. Attempting to access the admin account, I passed 'admin' as the username and took a guess at the secret question, guessing the favorite color to be green, which allowed me to obtain the admin's password.



Congratulations. You have successfully completed this lesson.
Web applications frequently provide their users the ability to retrieve a forgotten password. Unfortunately, m
mechanism properly. The information required to verify the identity of the user is often overly simplistic.
General Goal(s):
Users can retrieve their password if they can answer the secret question properly. There is no lock-out mech
username is 'webgoat' and your favorite color is 'red'. The goal is to retrieve the password of another user.

Webgoat Password Recovery

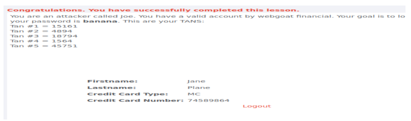
For security reasons, please change your password immediately.

Results:
Username: admin
Color: green
Password: 2275$starBo0rn3

Fig. 2. Admin's password

Effective Multi-Level Login systems provide enhanced security through two-factor authentication. During this lesson, our task was to bypass this robust security measure. Given the username and password for Joe I broke into Jane's account. I intercepted the request and then I manipulated the HTML input element by changing the username value from Joe to Jane, which tricked the system into granting me access to Jane's account without requiring her password.

For the next task, the problem is the first tan is already used but we tried to break into the system anyway. I intercepted the request and then I manipulated the HTML input element to make the system believe that the first TAN was being used for the first time. By changing the hidden TAN input value from "2" to "1" and entering the TAN 1 value in the text box, I was able to successfully complete the lesson.



Congratulations. You have successfully completed this lesson.
You are an attacker called Joe. You have a valid account by webgoat financial. Your goal is to lo
your password is **banana**. This are your TANS:
Tan #1 = 15161
Tan #2 = 4894
Tan #3 = 18794
Tan #4 = 1564
Tan #5 = 45751

Firstname:	Jane
Lastname:	Plane
Credit Card Type:	MC
Credit Card Number:	74589864

Logout

Fig. 3. Successfully logged into the system as Jane.

IV. Authentication Bypass

Authentication Bypass is tampering to achieve the right conditions to achieve required conditions by tweaking hidden input fields, taking out parameters, or forcing our way into browsing. The task was to get around some security questions. This was achieved by intercepting the request in ZAP and modifying the parameter names for SecurityQuestion0 and SecurityQuestion1, as deleting them proved ineffective.

V. JWT Tokens

For the first task (Decoding a JWT token), I figured out the contents of the JWT token by using WebWolf. I observed

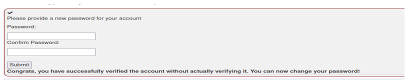

✓
Please provide a new password for your account
Password:
Confirm Password:
Submit
Congrats, you have successfully verified the account without actually verifying it. You can now change your password!

Fig. 4. Verified the account successfully without verifying the account.

that it's secured with the HS256 algorithm and the payload contains extensive information, including the 'user' username that I input in the text field to complete the task.

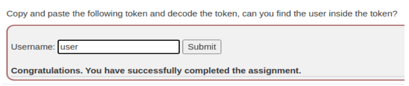

Copy and paste the following token and decode the token, can you find the user inside the token?
Username: user
Submit
Congratulations. You have successfully completed the assignment.

Fig. 5. Retreived username from JWT token.

For the next task (JWT Signing), we had to login as admin o reset votes, initially we are logged in as guest, who doesn't have admin privileges, so when we change to user from guest to tom, a JWT access token cookies is sent in the response, when we click on 'Reset votes' button, the browser sends a POST request to the server endpoint and adds the JWT previously given in the request identification, so reset doesn't work. So, to solve it we cracked the JWT token, we now have the payload and it has user as "Tom" and admin:"False", we changed the admin here to true and alg to None, thereby giving admin previleges to "Tom" and thus resetting the votes.

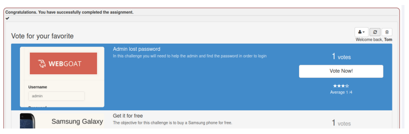

Congratulations. You have successfully completed the assignment.
✓
Vote for your favorite
Welcome back, **Tom**
Admin lost password
In this challenge you will need to help the admin and find the password in order to login
1 votes
Vote Now!
★★★☆
Average 1 /4
Username
admin
Samsung Galaxy
Get it for free
The objective for this challenge is to buy a Samsung phone for free.
1 votes

Fig. 6. Reset the votes

There were two multiple-choice questions. What is the result of the first code snippet? Ans: Throws an exception in line 12. What is the result of the second code snippet? Ans: Logs an error in line 9.

VI. Password Reset

I understood the Email feature in WebWolf and used it to reset the password of vkashir@webgoat.org to 'kayvid'. The next task was to figure out another user's password. I tried with the usernames "tom," "admin," and "Larry." Through a process of trial and error, I discovered that the 'admin' username was associated with the favorite color 'green,' which was too common and therefore vulnerable to hacking.

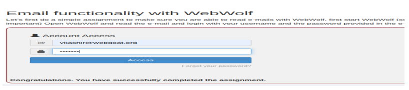

Email functionality with WebWolf
Let's first do a simple assignment to make sure you are able to read e-mails with WebWolf, first start WebWolf (se
important) Open WebWolf and read the e-mail and login with your username and the password provided in the e-
Account Access
@ vkashir@webgoat.org
••••••••
Access
Forgot your password?
Congratulations. You have successfully completed the assignment.

Fig. 7. Email functionality using WebWolf.

The next task helped us in identifying the types of security questions that aren't easily answerable, hinting at better security measures. For instance, asking for the name of a college or a job that one applied to but never attended. Such details aren't easily recalled and a hacker might just attempt to guess based on local institutions. Choosing challenging security questions is a more secure approach.

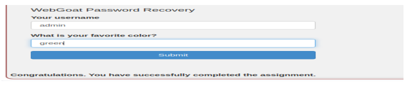

WebGoat Password Recovery
Your username
admin
What is your favorite color?
green
Submit
Congratulations. You have successfully completed the assignment.

Fig. 8. Password reset using security question.

VII. JWT Cracking

The JWT was initially provided as part of the WebGoat lesson. The first step involved saving the token to a local file named token.txt. To crack the JWT, HashCat (available at HashCat) was used. After cracking the token, the JWT was decoded using the online tool jwt.io. The key obtained from the cracking process enabled the crafting of a new JWT by changing username, sub, email values, and expiration date. The newly crafted JWT was successfully accepted by WebGoat.

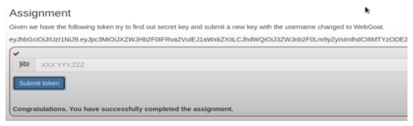

Assignment
Given we have the following token try to find out secret key and submit a new key with the username changed to WebGoat.
eyJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJXZWJHb2F0IFRva2VuIEJ1aWxkZXIiLCJhdWQiOiJ3ZWJnb2F0Lm9yZyIsImlhdCI6MTYzODE2
✓
XXX.YYY.ZZZ
Submit token
Congratulations. You have successfully completed the assignment.

Fig. 9. Successfully cracked JWT Token.

VIII. Conclusion

The lab highlights critical vulnerabilities in password security, authentication bypass methods, and JSON Web Token (JWT) manipulation within a WebGoat. Password recovery was found to be susceptible to brute-force attacks due to weak secret questions and an absence of lock-out mechanisms, as demonstrated by gaining unauthorized access to an admin account by guessing a common favorite color. Multi-level login systems were bypassed by HTML input manipulation, allowing unauthorized access to accounts without the required credentials. Authentication bypass was achieved by tweaking hidden input fields and modifying security questions' parameters. JWT vulnerabilities were exposed by decoding using WebWolf, revealing that tokens secured with HS256 could be cracked and manipulated to elevate privileges or reset passwords. This was further evidenced by changing token payload values and exploiting weak security questions to reset user passwords, illustrating the urgent need for more robust security measures in authentication systems.

References

[1] KVM
https://www.linux-kvm.org/page/MainPage..

[2] WebGoat
https://owasp.org/www-project-webgoat/..

[3] OverLeaf
https://www.overleaf.com/.

[4] Github
https://en.wikipedia.org/wiki/GitHub.

[5] RED
https://kb.iu.edu/d/apum.

[6] OWASP ZAP
https://www.zaproxy.org/

[7] Firefox DevTools
https://firefox-source-docs.mozilla.org/devtools-user/

[8] Information Schema Tables
https://www.mssqltips.com/sqlservertutorial/196/
information-schema-tables/.