Assignment-2 Report

Akhilesh Gowda Mandya Ramesh

***Abstract*—The Assignment-2 introduced us to additional sections on JWT Tokens. There were two tasks to be completed task 10 and task 11(bonus).**

I. Introduction

THE insecure application that provides functionalities to observe the vulnerabilities of applications, such as WebGoat can be used to learn and get exposure to different types of vulnerabilities in the area of injection and authentication and broken Authentication such as Authentication Bypass, JWT tokens and password reset.

II. JWT tokens

This section introduced JSON Web Tokens (JWT) which can be used for authentication. It introduced how to securely implement the usage and validation of the tokens. The structure of a JWT token consists of three parts header, claims and signature. both the header and claims are represented by JSON Object. The header describes the cryptographic operations applied to he JWT and optionally, additional properties of JWT. The claims represent a JSON object whose members are the claims conveyed by the JWT.

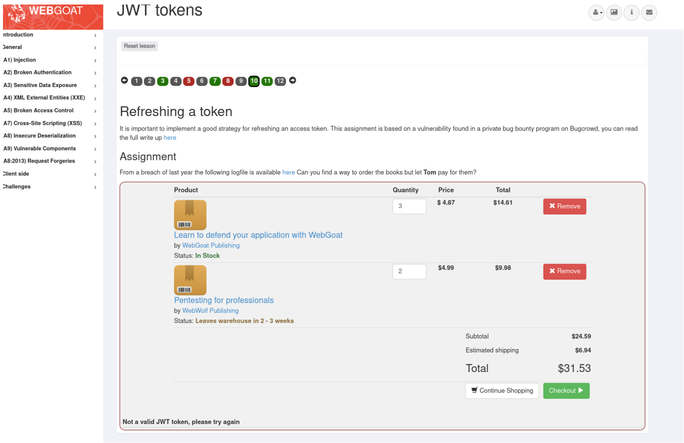
III. Task-10

The task-10 which is the task that needs to be completed as part of this assignment. This tasks demands us to pay for the items in the cart as Tom, but the user currently logged in is Jerry. I used Burp suite to intercept the request while sending a request for checkout. Then i used the JWT token decoder to decode the token. Then after the JWT token is decoded i changed The user to Tom and algorithm to none and then used this updated JWT token in the intercepted request, which completed the task of paying for the items in the cart as TOM

IV. Task-11 (Bonus)

This is the bonus task of Assignment-2. The Bonus task was to complete the step-11 of the JWT token section from OWSAP webogoat application. Here the task presented 2 twitter accounts which were named as Jerry and Tom. The task here wanted Jerry to remove Tom's twitter account, even though it can only delete his own account. So to Delete Tom's account i clicked on the remove button and intercepted the request using Burp suite. Here i extracted the JWT token from the request and decoded the JWT Token using an online JWT token decoder. Here i changed the credentials of Jerry to Tom's in the decoded JWT token and used this new updated JWT token, in the intercepted request and sent it. This resulted in deleting the Tom's account and completed the bonus task of the assignment,

Advisor: Luyi Xing, Jiale Guan.


WEBGOAT
JWT tokens
ntroduction
General
A1) Injection
A2) Broken Authentication
A3) Sensitive Data Exposure
A4) XML External Entities (XXE)
A5) Broken Access Control
A7) Cross-Site Scripting (XSS)
A8) Insecure Deserialization
A9) Vulnerable Components
A8:2013) Request Forgeries
Client side
Challenges
Reset lesson
1 2 3 4 5 6 7 8 9 10 11 12

Refreshing a token

It is important to implement a good strategy for refreshing an access token. This assignment is based on a vulnerability found in a private bug bounty program on Bugcrowd, you can read the full write up here

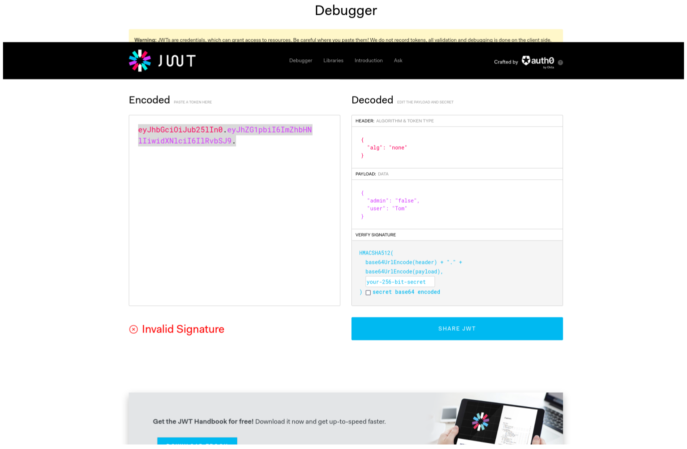
Assignment

From a breach of last year the following logfile is available here Can you find a way to order the books but let **Tom** pay for them?

Product	Quantity	Price	Total	
Learn to defend your application with WebGoat by WebGoat Publishing Status: In Stock	3	$ 4.87	$14.61	✖ Remove
Pentesting for professionals by WebWolf Publishing Status: Leaves warehouse in 2 - 3 weeks	2	$4.99	$9.98	✖ Remove
		Subtotal	$24.59	
		Estimated shipping	$6.94	
		Total	$31.53	

Not a valid JWT token, please try again

Fig. 1. Task-10 completion



Warning: JWTs are credentials, which can grant access to resources. Be careful where you paste them! We do not record tokens, all validation and debugging is done on the client side.

```
eyJhbGciOiJub25lIn0.eyJhZG1pbiI6ImZhbHN
lIiwidXNlciI6IlRvbSJ9.
```

```
{
  "alg": "none"
}
```

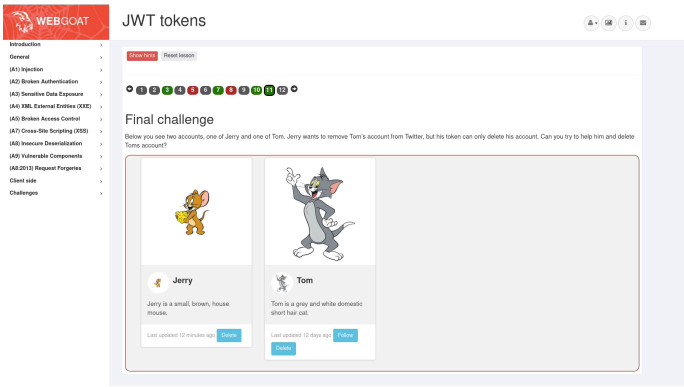
```
{
  "admin": "false",
  "user": "Tom"
}
```

```
HMACSHA512(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  your-256-bit-secret
) □ secret base64 encoded
```

Invalid Signature

Get the JWT Handbook for free! Download it now and get up-to-speed faster.

Fig. 2. Decoding the JWT Token


WEBGOAT
JWT tokens
Introduction
General
(A1) Injection
(A2) Broken Authentication
(A3) Sensitive Data Exposure
(A4) XML External Entities (XXE)
(A5) Broken Access Control
(A7) Cross-Site Scripting (XSS)
(A8) Insecure Deserialization
(A9) Vulnerable Components
(A8:2013) Request Forgeries
Client side
Challenges
Show hints
Reset lesson
1 2 3 4 5 6 7 8 9 10 11 12

Final challenge

Below you see two accounts, one of Jerry and one of Tom. Jerry wants to remove Tom's account from Twitter, but his token can only delete his account. Can you try to help him and delete Toms account?

Jerry

Jerry is a small, brown, house mouse.

Last updated 12 minutes ago Delete

Tom

Tom is a grey and white domestic short hair cat.

Last updated 12 days ago Follow Delete

Fig. 3. Bonus task - 11 completion

V. Conclusion

In Summary as WebGoats vulnerabilities can be exploited to study and bolster the attacks, We can also use it to study and mitigate the attacks such caused by SQL injection as the CIA Triads can be compromised by performing SQL Injection. This lab we studied Broken Authentication includes Authentication Bypasses, JWT tokens and password reset. This was we can decide to try and develop a website that are less vulnerable to all of these. This way we can try to develop websites that do not compromise a CIA triads.Authentication Bypasses can occur in many ways but they usually take advantage of some flaws that is present in he configuration or the logic. They involve tampering to achieve the right conditions. They make use of hidden inputs which relies on simple inputs that is in the Web page or DOM. Authentication is one of the most fundamental thing to be implemented in the website. We need to make sure that there is no flaw or vulnerability that the attacker can use to attack the website. We need to make sure that the websites are not vulnerable to phishing.

References

[1] https://github.com/WebGoat/WebGoat
[2] https://owasp.org/
[3] https://www.zaproxy.org