

LAB1: SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

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



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

To solve the lab, perform an SQL injection attack that causes the application to display details of all products in any category, both released and unreleased.

The screenshot shows a web browser window with the URL <https://portswigger.net/web-security/sql-injection/lab-retrieve-hidden-data>. The lab title is "Lab: SQL injection vulnerability in WHERE clause allowing retrieval of hidden data". It is categorized as "APPRENTICE" and "LAB" and is marked as "Solved".

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

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

The goal is: "To solve the lab, perform an SQL injection attack that causes the application to display details of all products in any category, both released and unreleased."

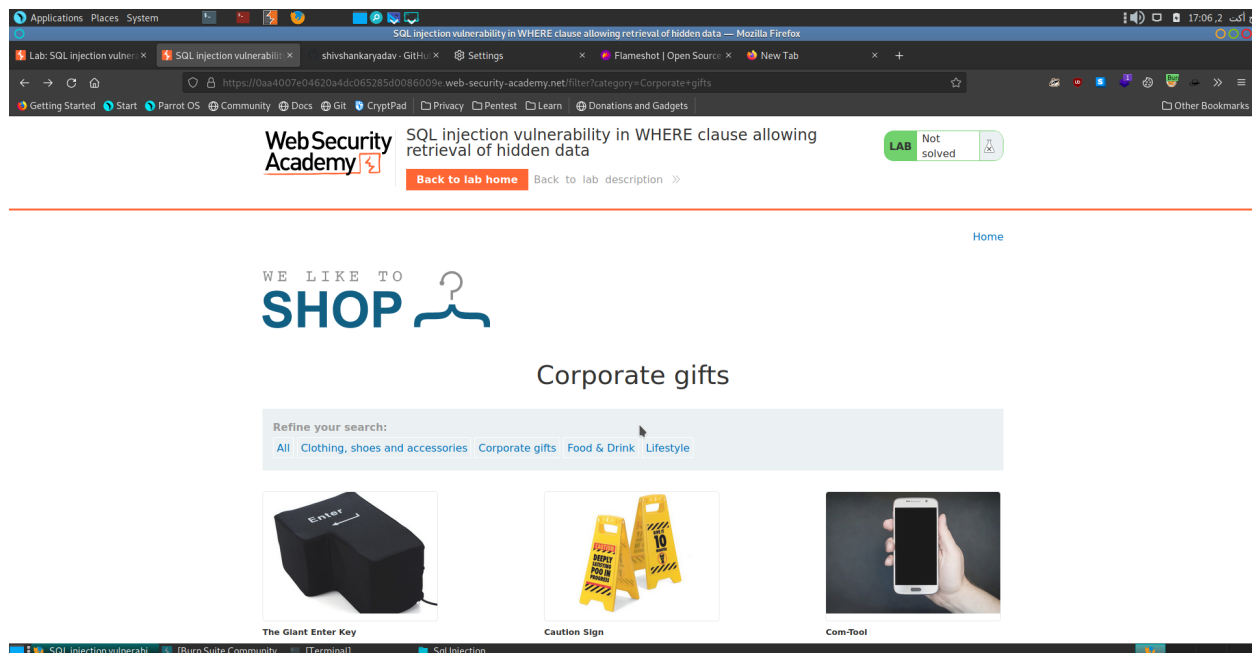
There is a green button labeled "Access the lab". Below it are sections for "Solution" and "Community solutions".

On the right side, there is a "Track your progress" sidebar showing:

- Learning materials: 0% (View all)
- Vulnerability labs: 3% (View all)
- Level progress: 2 of 32 (Apprentice), 5 of 137 (Practitioner), 0 of 35 (Expert)
- Your level: NEWBIE (Solve 50 more labs to become an apprentice)
- See where you rank on our Hall of Fame >>

STEP 1 :

Click on the access the lab



We can see the URL having a category parameter. It has three input clothing, shoe and accessories, Corporate gifts, Foods & Drink and Lifestyle. First I click on all tabs and check the result on page. When I click on All tab it shows all image but other tab only show four image.

Lets understand the logic behind the lab,

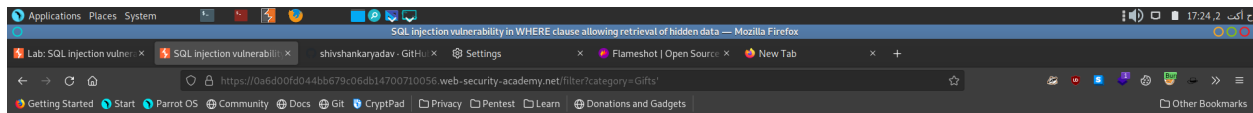


```
SELECT * FROM products WHERE category = 'Gifts' AND released = 1
```

when released =1 its show all product in any category but when released = 0 it shows only product of particular category. Let's apply sql injection in category parameter.

Step 2 :

I just put single quote at the end of URL in the category parameter. It throw an internal server error on the application. It clearly means this application having a SQL injection vulnerability.



SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

[Back to lab home](#)

[Back to lab description](#)

LAB Not solved

Internal Server Error
Internal Server Error

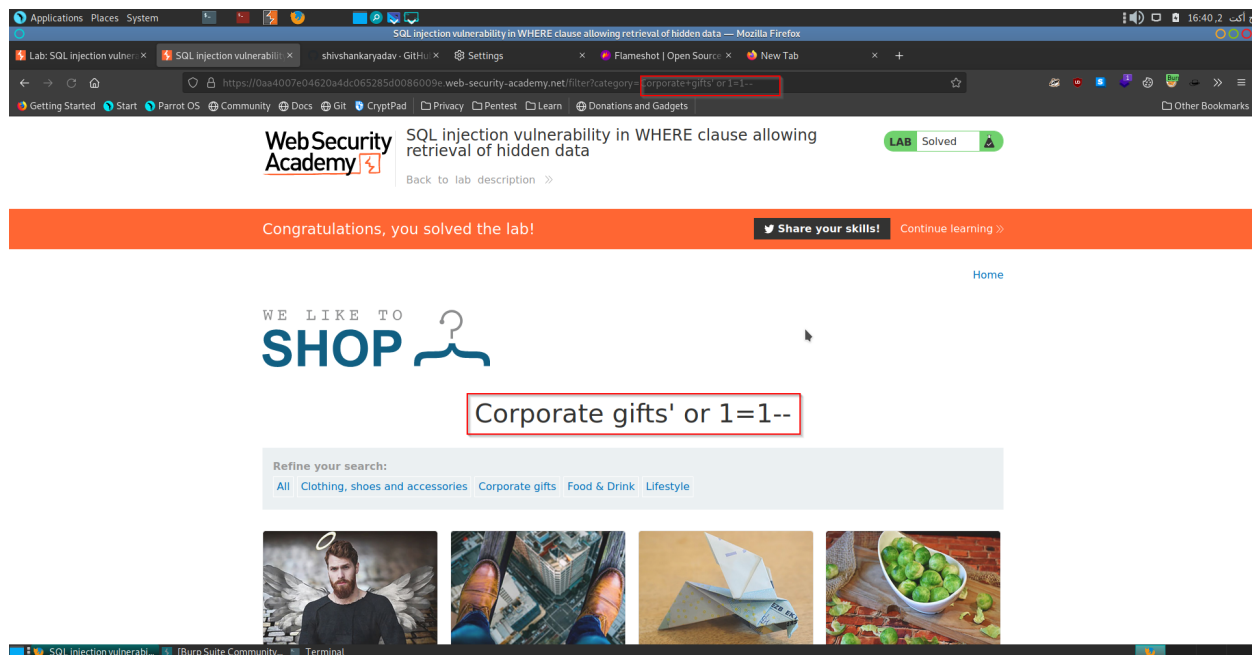


STEP 3 :

So now its time to exploit the vulnerability. I put a SQL injection common payload after the URL.



' or 1=1—



Hurrah! Lab has solved