

Portswigger Academy - SQLi APPRENTICE - ENG

SQL injection Challenges

Challenge 1 “SQL injection vulnerability in WHERE clause allowing retrieval of hidden data”

1) In the first lab, it asks us to show all items in store. Also, it shows the SQL query so we can find a payload easily.

In SQL query, the “category” part would be the place where application takes its inputs. So, purpose of lab is attack there. Since, our purpose is listing all products, we can use a simple payload like “**or true --**”.





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

2) When we open the lab page it looks like that:

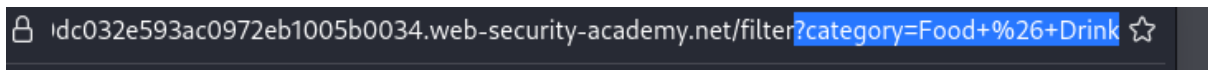
WE LIKE TO SHOP

Refine your search:

[All](#) [Clothing, shoes and accessories](#) [Corporate gifts](#) [Food & Drink](#) [Lifestyle](#)

			
Portable Hat ★★★★★ \$38.99	The Trolley-ON ★☆☆☆☆ \$22.90	First Impression Costumes ★★★☆☆ \$67.38	The Giant Enter Key ★☆☆☆☆ \$95.50
View details	View details	View details	View details

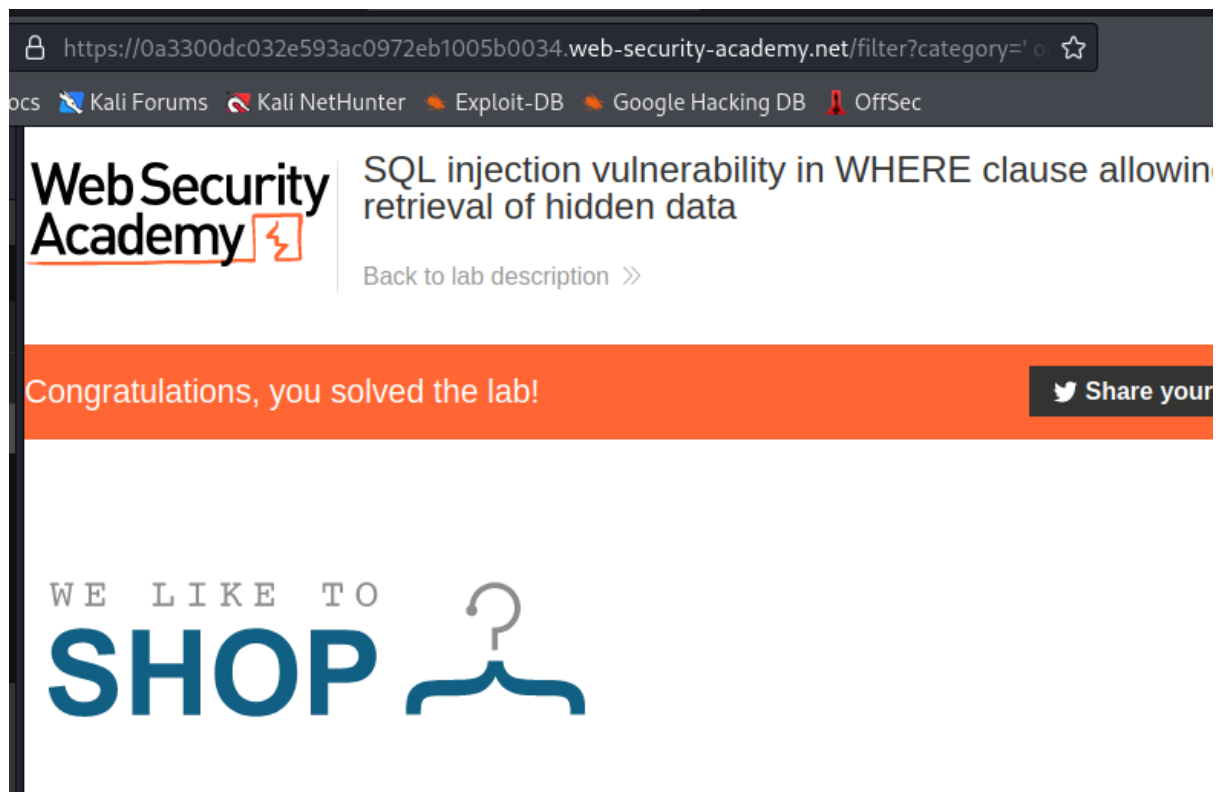
3) After we choose any category, the parameter transported over URL so it is a “GET” parameter.



4) We should manipulate the SQL query without getting an SQL error. To be able to do this, we should put our payload in “Gifts” part and use a payload like “**or 1=1 --**”. Then the SQL query will be like this:

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

5) When we add our payload, the condition part of the query always will be true and the “AND released = 1” part will be commented out so all results will be listed.



Challenge 2: SQL injection vulnerability allowing login bypass

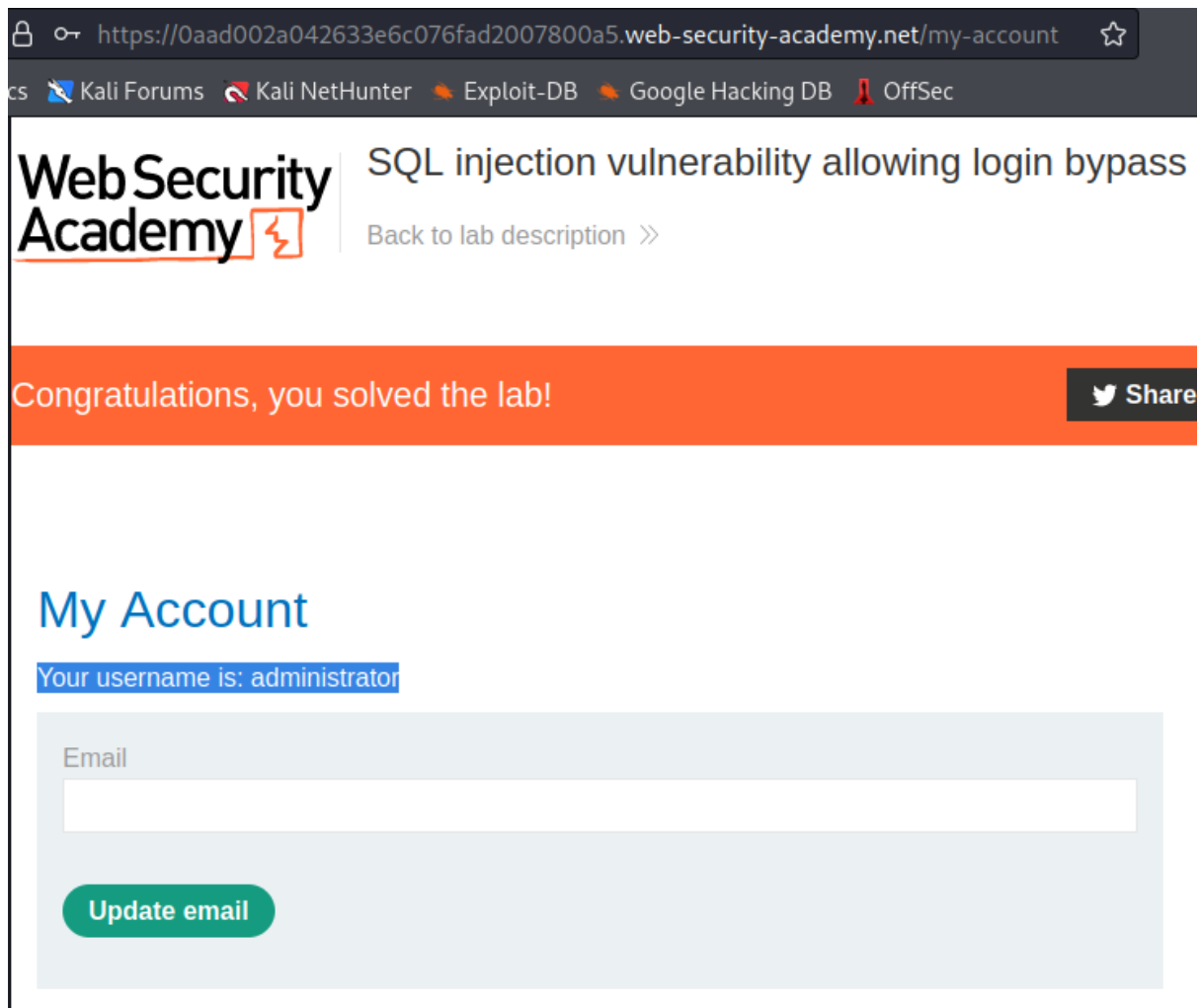
- 1) In this lab, explanation says, there is an SQLi vulnerability in login form. The goal is login as “administrator”.
- 2) The login form in “my account” page. We already know that there is an SQLi. Unlike the first lab, in this one, we don't have the SQL query but we can predict it. Probably, SQL query will be like that:

```
Select * from USERS Where username='$Username' and password='$password' Limit 1;
```

- 3) If we use a payload like that “ ‘ or ‘1’=’1” for both username and password part, we can manipulate the query like that:

```
Select * from USERS Where username=' ' or '1'='1' and password=' ' or '1'='1' Limit 1;
```

4) When we manipulate the SQL query, the result will be all table but the “Limit 1” will choose the top of it. Since the first value in the result is Administrator, we can solve it in this way.



5) But if we couldn't login as admin in this way, we should specify the admin parameter and we can comment out the rest of query. Also, if the application use some type of hashing algorithm before running the query, this will be more useful. When we try, SQL query ll be like that:

Payload: “**Administrator**” -- “

```
Select * from USERS Where username='Administrator' -- ' and password=' ' or '1'='1' Limit 1;
```