

SQL Injection LAB – Software Security Assessment

Name: Ravi Teja Thota

Student ID:2486986

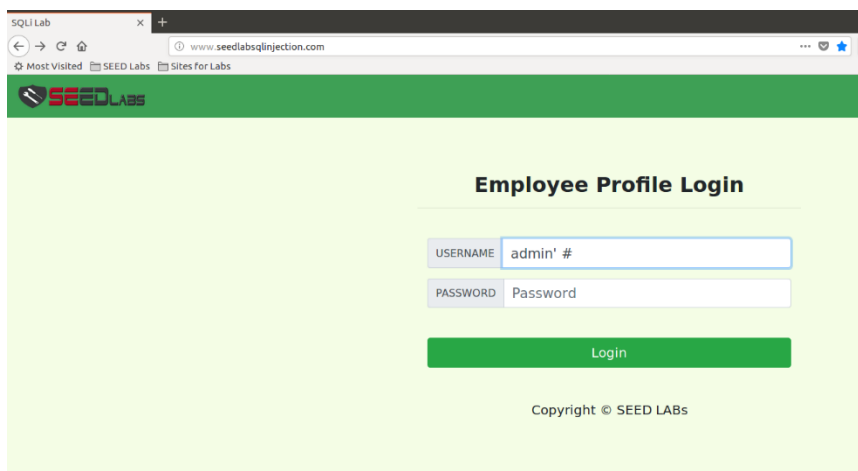
AIM: The main motto of this lab is to perform SQL injection on a website to acquire desired output in retrieving or manipulating the data using SQL Queries.

Setting up lab: in order to perform the sql injection we need to use <http://www.SEEDLabSQLInjection.com>

Webpage. To run the website in the web browser we need to host the website which can be done by adding the website to /var/www/SQLInjection/ file by giving 127.0.0.1 as local host address and www.seedlabsqlinjection.com as the website and save the website. Now, go to a browser and enter the website in address bar and the website is ready for go.

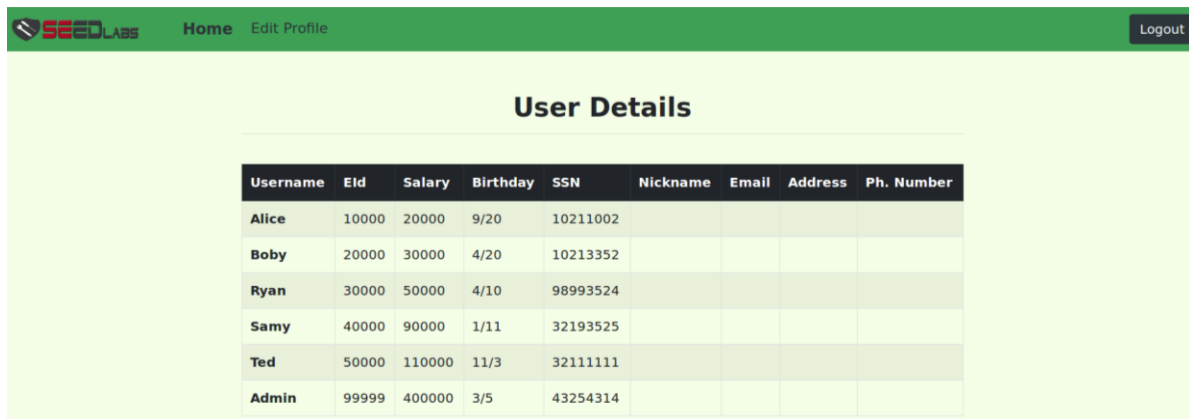
Task 3.1.1 performing SQL Injection on a website

in order to perform sql injection, I used “admin’#” command in the username. The main motto of me using the specified command is to get access to admin account. As the background sql command running is like select id,name,salary..... From credential where name ‘\$username’ and password=’\$hashed_pwd’. So inorder to only check with the username and escape the password part we need to use # in the end to comment out the remaining query so that what ever data present in the given user the data is retrieved without checking for password.



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 titled "Employee Profile Login" and contains a login form with two input fields: "USERNAME" and "PASSWORD". The "USERNAME" field contains the text "admin' #" and the "PASSWORD" field contains the text "Password". Below the input fields is a green "Login" button. At the bottom of the page, there is a copyright notice: "Copyright © SEED LABS".

Post sql injection:

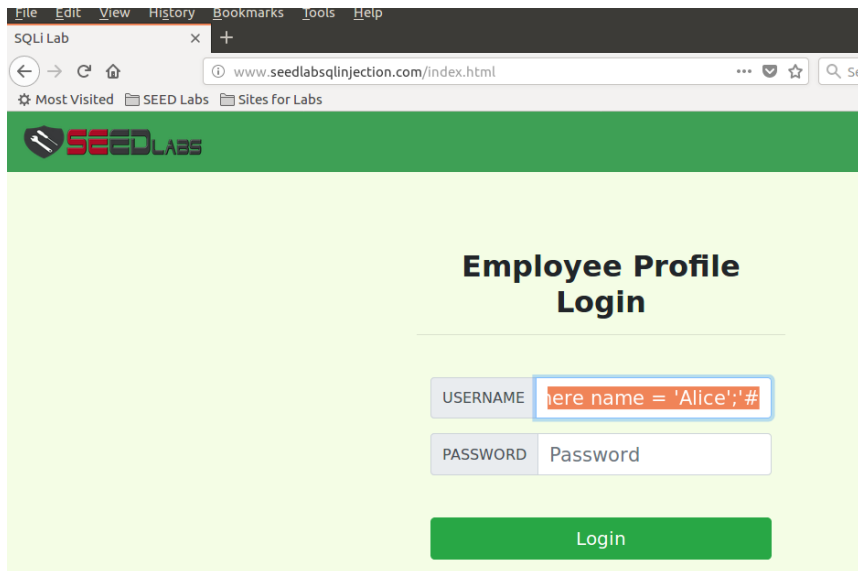


Username	Eid	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	20000	9/20	10211002				
Boby	20000	30000	4/20	10213352				
Ryan	30000	50000	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

Task 3.1.2 Append a new SQL statement

In this task we are supposed to append another sql statement with the general sql injection statement using “;” as the separator for two statement as shown below statement

admin ‘; delete from credential where name = 'alice';’#



File Edit View History Bookmarks Tools Help

SQLi Lab

www.seedlabsqlinjection.com/index.html

Most Visited SEED Labs Sites for Labs

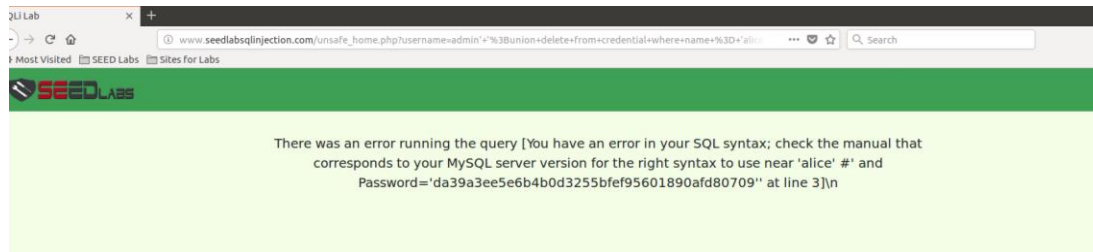
Employee Profile Login

USERNAME

PASSWORD

Login

The attack is failed due to the countermeasures offered by MySQL that prevents multiple statements from executing when invoked in php.



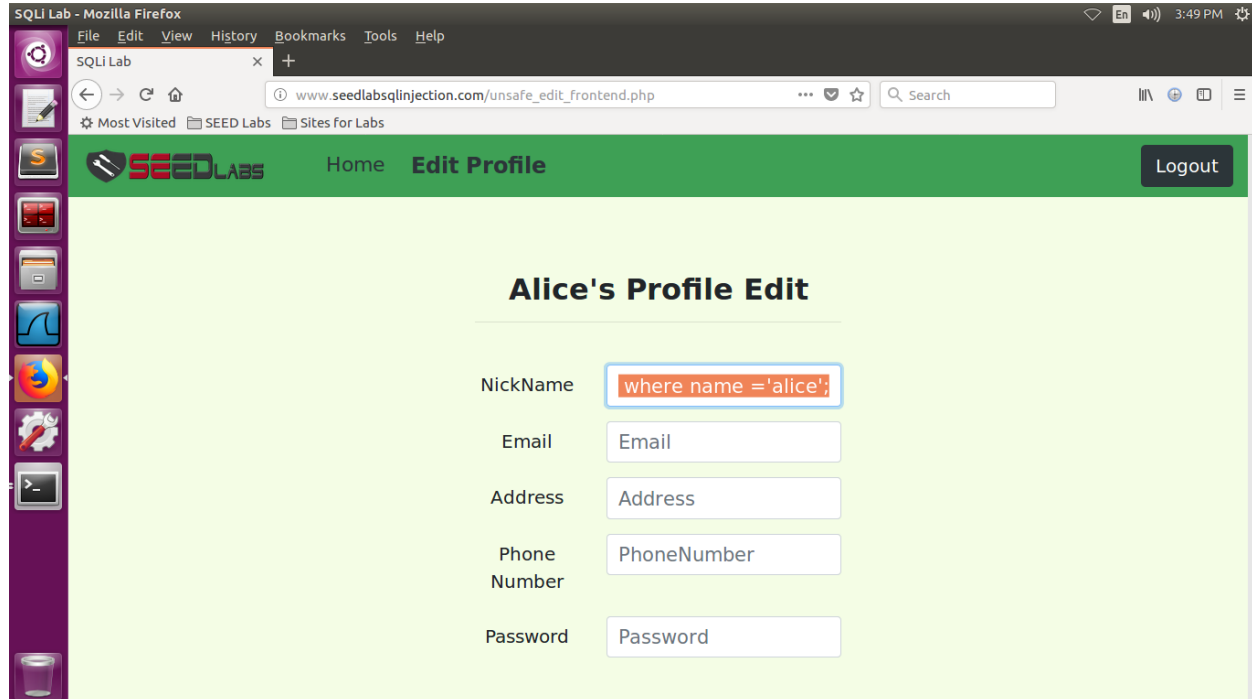
3.2.1 SQL Injection attack on update statement – modifying own and other people's salary

We are supposed to modify alice's salary once entering alice's account. For which i logged into alice's account and clicked on edit profile.

In order to perform update salary to 40000 for alice, I entered the first query and clicked enter. After increased salary of alice I decided to reduce the salary of Bobby's for which I used second statement in the nickname field as shown in the below picture..

'salary=40000 where name ='alice';

'salary=10000 where name =boby;



User Details

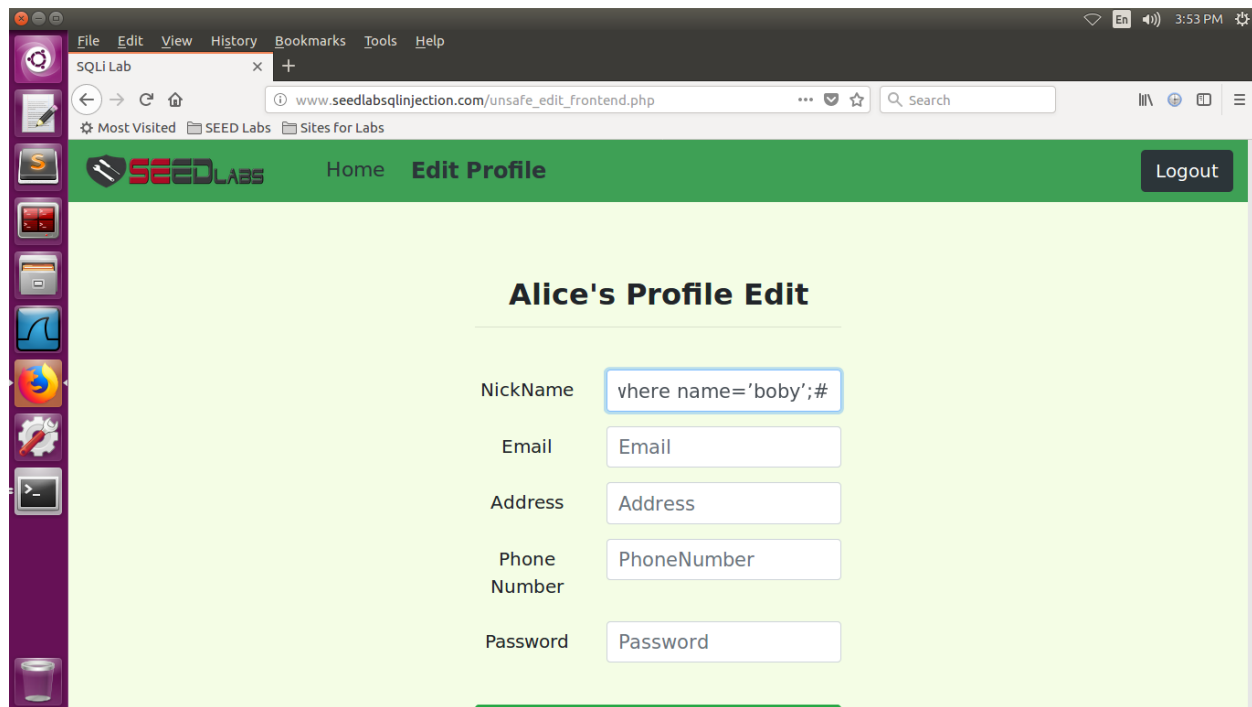
Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	40000	9/20	10211002				
Boby	20000	10000	4/20	10213352				
Ryan	30000	50000	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

We can observe that the salary of alice and boby's are changed to 40000 and 10000 respectively.

3.2.1 SQL Injection attack on update statement- modify other people's password

As the story of alice and boby's rivalry narrates alice wanted to more damage to boby like modifying the password of boby's account. So in order to do that alice's logged in to her account and clicked on edit profile and in the nick name field. Now Alice enters the following command in the nickname field. As you see the query we are entering the in the password field of query a SHA1 hash value for the word 'attacker'.

`',password='2e51cf3f58377b8a687d49b960a58dfc677f0ad' where name='boby';#`

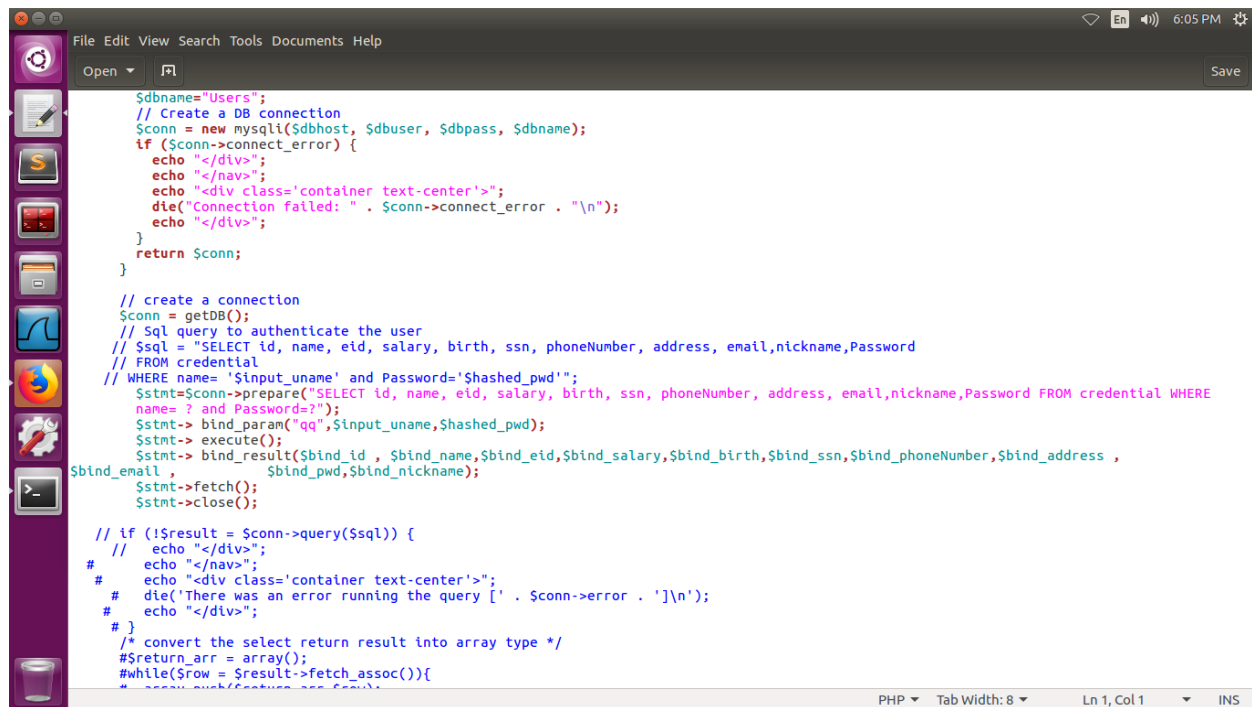


Task 3.3 Countermeasure – prepared statement

In this task we are supposed to make sure that the SQL Injection doesn't happen. So in order to do that we need to modify the code in `unsafe_home.php` present in the `/var/www/SQLInjection` directory.

While going through that statement I came up with the select statement where without any parameter check the username and the password are directly running in the sql query which caused sql injection in the first place.

We will be using prepared statement so that the code and data are sent in separate channels which makes the database understand that the code received in data channel should not be executed. For which code is shown in the below picture.



The screenshot shows a code editor window with a dark theme. The menu bar includes File, Edit, View, Search, Tools, Documents, and Help. The toolbar has buttons for Open, Save, and a search icon. The code is written in PHP and includes comments in English. It defines a database name, connects to a MySQL database, and attempts to authenticate a user. The code uses PDO for database operations. The status bar at the bottom indicates the file is a PHP script, the tab width is 8, and the cursor is at line 1, column 1.

```
File Edit View Search Tools Documents Help
Open Save

$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
// $sql = "SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password
// FROM credential
// WHERE name= '$input_uname' and Password='$hashed_pwd'";
$stmt=$conn->prepare("SELECT id, name, eid, salary, birth, ssn, phoneNumber, address, email,nickname,Password FROM credential WHERE
name= ? and Password=?");
$stmt->bind_param("qq",$input_uname,$hashed_pwd);
$stmt->execute();
$stmt->bind_result($bind_id, $bind_name,$bind_eid,$bind_salary,$bind_birth,$bind_ssn,$bind_phoneNumber,$bind_address ,
$bind_email, $bind_pwd,$bind_nickname);
$stmt->fetch();
$stmt->close();

// 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()){
    $return_arr[]=$row;
}
```

PHP Tab Width: 8 Ln 1, Col 1 INS