

# Lab 9.1 SQLi Review

💡 You were exposed to basic SQL Injection in SEC260 - Web and Application Security. It is a very common method of exploitation and deserves some consideration and review in this class.

## Preparation

### Mysqld configuration on Kali

```
sudo systemctl enable mysqld
sudo systemctl start mysqld
sudo mysql_secure_installation
```

- Note, current root password for mysql is nothing so hit [Enter]

```
Switch to unix_socket authentication [Y/n] n
Change the root password? [Y/n] Y
Remove anonymous users?
Disallow root login remotely? [Y/n] y[Y/n] y
Remove test database and access to it? [Y/n] y
Reload privilege tables now? [Y/n] y
```

### clone the sqli-labs-php git repository

```
mkdir -p ~/sec335/week9
cd ~/sec335/week9
git clone https://github.com/skyblueeee/sqli-labs-php7.git
cd sqli-labs-php7
```

Updated 4/5/22

edit sql-connections/db-creds.inc

```
GNU nano 5.4 sql-connections/db-creds.inc
<?php

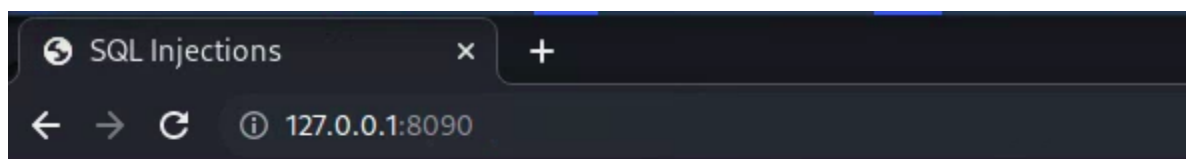
//give your mysql connection username n password
$dbuser = 'root';
$dbpass = ' ';
$dbname = "security";
$host = 'localhost';
$dbname1 = "challenges";
```

start the application

access the application and click on the Setup/reset Database for labs link.

```
File Actions Edit View Help
champuser@kali: ~/sec335/week9/sqli-labs-php7 x champuser@kali: ~/sec335/week9/sqli-labs-php7/Less-1 x

(champuser@kali)-[~/sec335/week9/sqli-labs-php7]
$ pwd
/home/champuser/sec335/week9/sqli-labs-php7
(champuser@kali)-[~/sec335/week9/sqli-labs-php7]
$ php -S 127.0.0.1:8090 -t .
[Tue Apr 5 16:30:15 2022] PHP 8.1.2 Development Server (http://127.0.0.1:8090) started
```



## **SQLi-LABS Page-1(Basic Challenges).**

[Setup/reset Database for labs](#)

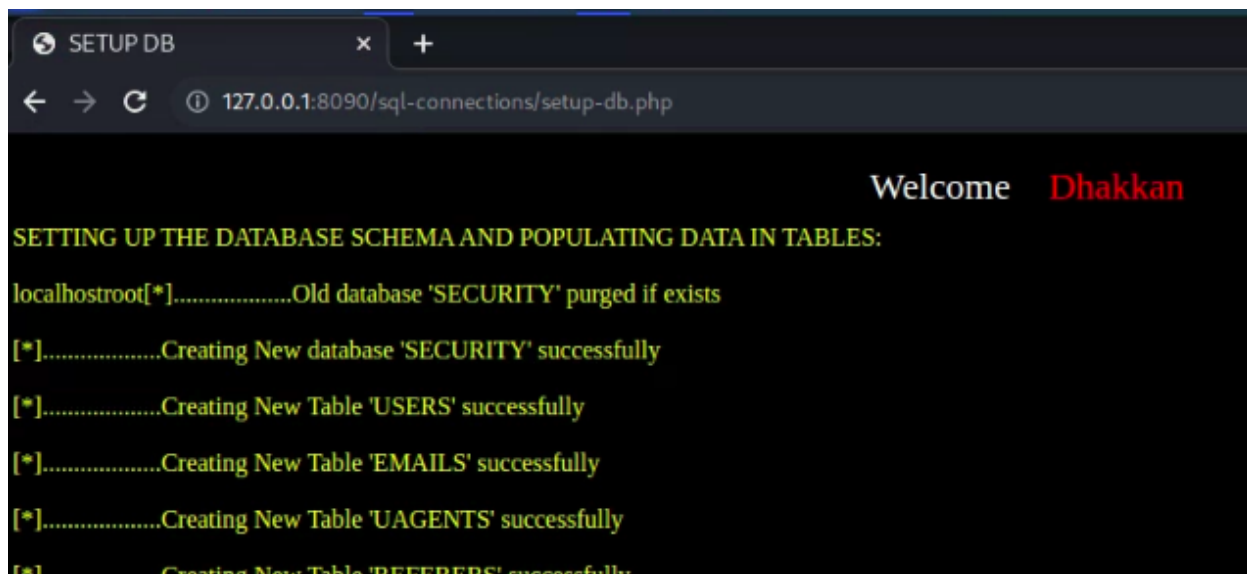
1

[Page-2 \(Advanced Injections\)](#)

[Page-3 \(Stacked Injections\)](#)

[Page-4 \(Challenges\)](#)

Updated 4/5/22



Updated 4/5/22

## Deliverables

Add increased error handling and a debug statement for the raw sql as shown below. By default php8 will suppress many of the error messages.



```
champuser@kali: ~/sec335/week9/sqli-labs-php7 x champuser@kali: ~/sec335/week9/sqli-labs-php7/Less-1 x
GNU nano 6.2 index.php
welcome vulnerable

<?php
//including the Mysql connect parameters.
include("../sql-connections/sqli-connect.php");
//SEC335
error_reporting(E_ALL);
ini_set('display_errors', 1);

// take the variables
if(isset($_GET['id']))
{
$id=$_GET['id'];
//logging the connection parameters to a file for analysis.
$fp=fopen('result.txt','a');
fwrite($fp,'ID:'.$id."\n");
fclose($fp);

// connectivity

$sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
// $sql="SELECT * FROM users WHERE id='0' union select 1,2,3 -- ' LIMIT 0,1";
// $sql="SELECT * FROM users WHERE id='0' union select 1,2,3 # ' LIMIT 0,1";
$result=mysqli_query($con1, $sql);
$row = mysqli_fetch_array($result, MYSQLI_BOTH);
//SEC335
printf("<br>raw_sql: %s<br>", $sql);
    if($row)
    {
        echo "<font size='5' color= '#99FF00'>";
        echo 'Your Login name:'. $row['username'];
        echo "<br>";
        echo 'Your Password:'. $row['password'];
        echo "</font>";
    }
    else
    {

```

Updated 4/5/22

Go through the following [SQLi walk through](#)  
Capture and label screenshots for:

1. Display the Login name and password for arbitrary user
2. Error condition when number of columns are exceeded
3. A Union select that displays your own value for login name and password
4. Another union that displays the mysql user and database
5. A union that dumps all the tables in the current database
6. A union that dumps all the usernames and passwords

Deliverable 7. Figure out how to run sqlmap against the vulnerable uri: <http://127.0.0.1:8090/Less-1?id=1>

- Run this using Medium Difficulty and Intermediate Enumeration.
- Figure out how to dump the contents of the users table in the security database.
- Provide a screenshot showing the results of dumping the user's table.