NED University of Engineering and Technology
Department of Computer and Information Systems Engineering

Lab 6

Sheikh Muhammad Farjad
CS-16059
Section A
Batch 2016-17

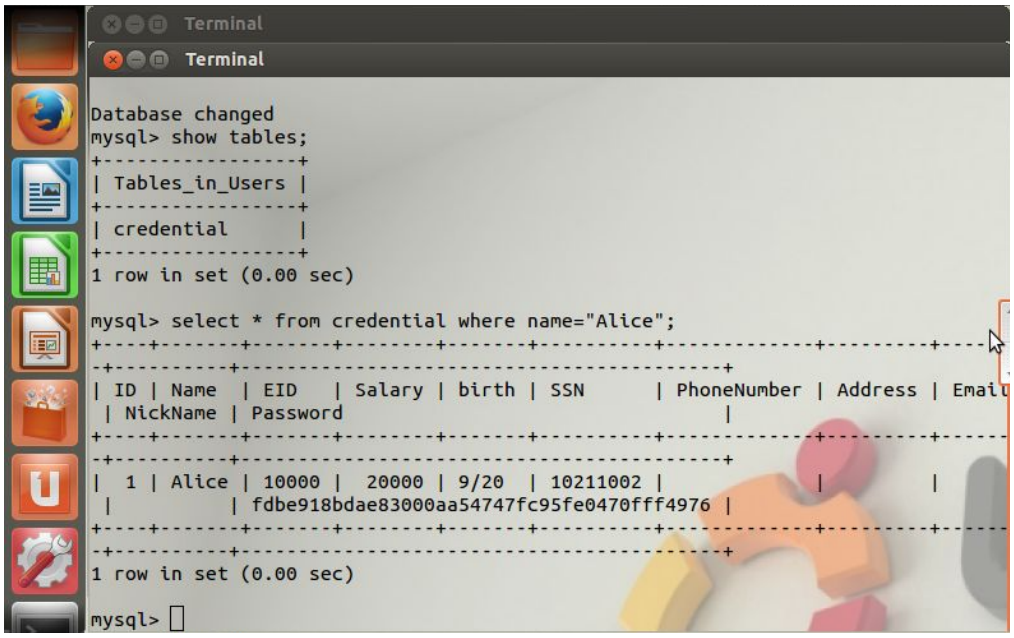
Course Instructor: Sir Umar Iftikhar
Computer Systems Security

Prerequisite

A new patch was required to be downloaded from Seed Ubuntu's site for performing lab 6. The patch contained an Employee Management web application. This web application contained SQL injection vulnerabilities and the students were asked to exploit these vulnerabilities as directed by the tasks.

Task 1

The downloaded patch contained the Users database. The tables of the database are shown in the below screenshot. An SQL query was also run to further elucidate the existing database.



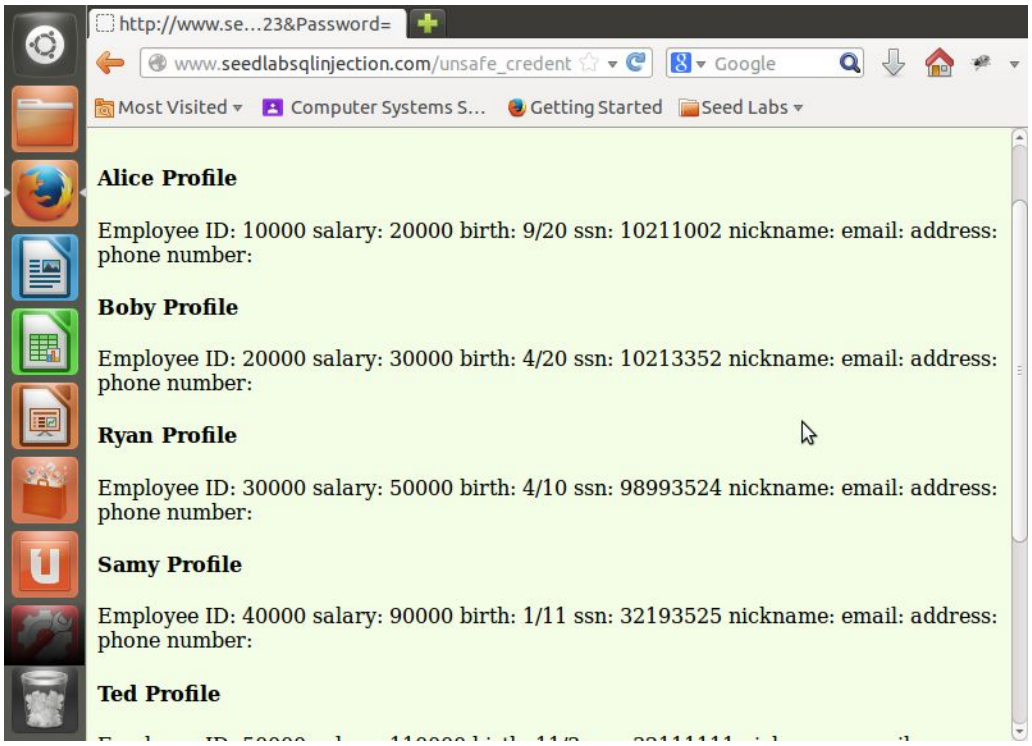
Task 2

Task 2.a: SQL Injection Attack from webpage.

The code for the database schema of the web application was mentioned in the manual. A thorough analysis of the code resulted in the information that the authentication mechanism compares only the ID, not password. While considering this flaw, the Employee ID field was filled provided the ID of admin along with a single quotation mark. Following string was provided to the Employee ID field of the login page of the web application:

```
99999' #
```

99999 is the ID for admin and single quotation mark encapsulated the result of Employee ID field and this is how the login page was easily bypassed.

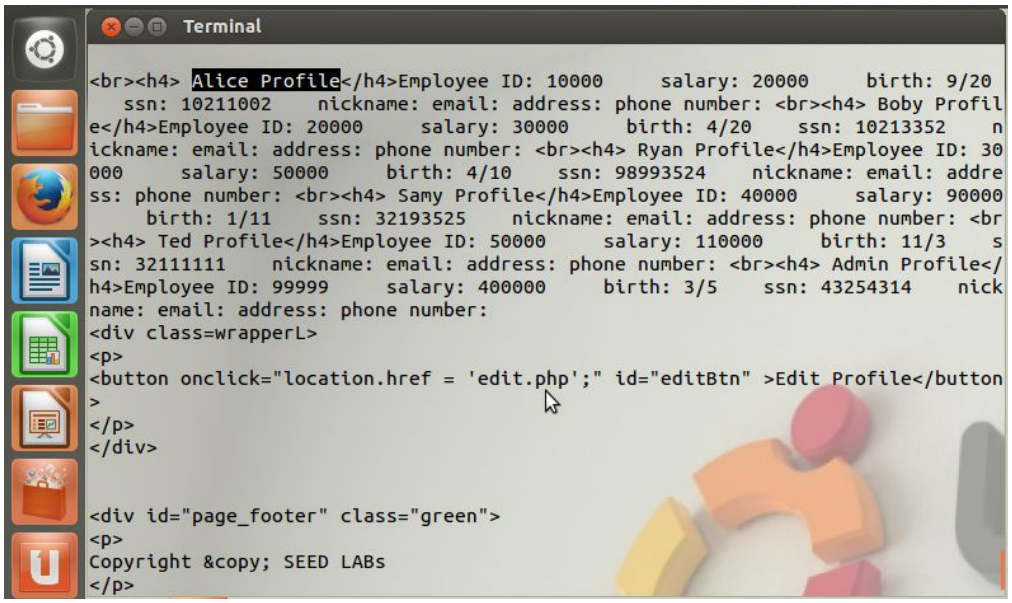


Task 2.b: SQL Injection Attack from command line

The task accomplished in Task 2.a was executed in this task but through command line interface. The curl command-line utility is used for fetching the web pages. For passing the SQL query in via command-line interface, the special characters like space, ‘, and # were converted unto their respective ascii values. Following command was run in terminal:

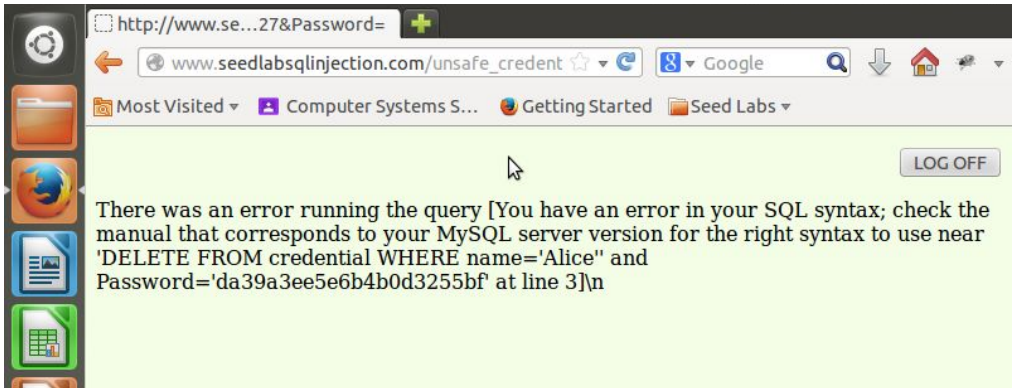
```
curl
'www.SeedLabSQLInjection.com/unsafe_credential.php?EID=99999%27%
20%23'

where,
%27 ---> ascii for '
%20 ---> ascii for space
%23 ---> ascii for #
```



Task 2.c: Append a new SQL statement.

In this task, a new SQL statement was appended into the payload being sent to the Employee ID field but this resulted in an error in the next page. It implies that there are filtering mechanism deployed for filtering the strings passed to the Employee ID field.



Task 3

Task 3.a: SQL Injection Attack on UPDATE Statement

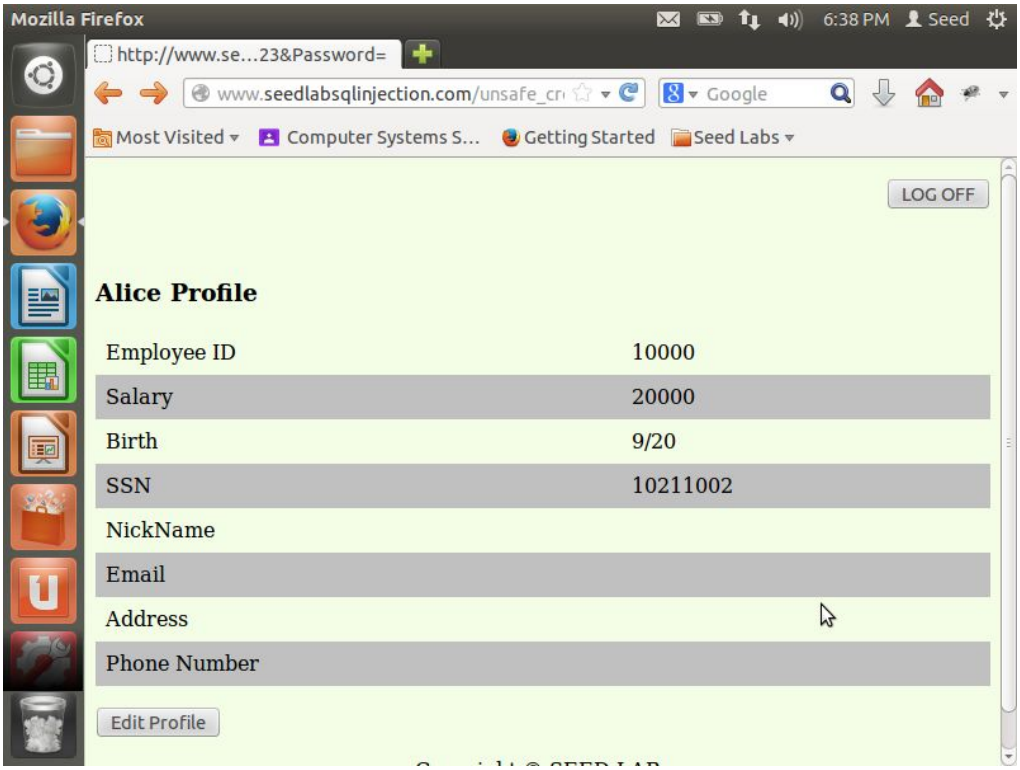
(modify salary)

In order to do this task, I logged in as Alice (EID: 10000) using the trick used in Task 2.a. After logging into Alice's account, I clicked on the the Edit Profile button and inserted the following string in any input field:

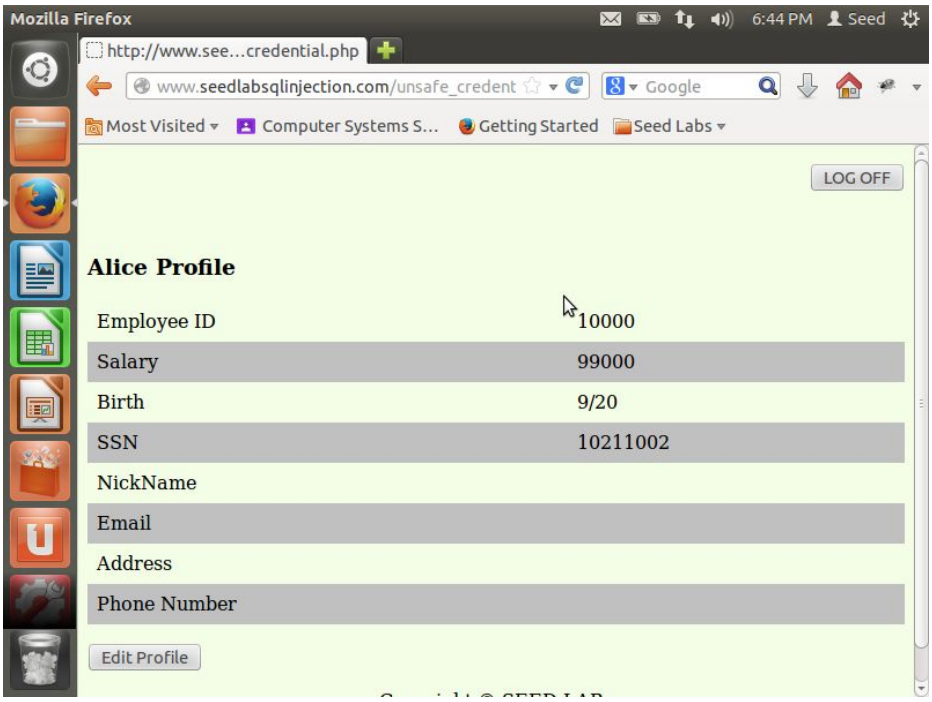
```
', salary=99000 where EID=10000, #
```

The above string changes the salary or Alice from previous salary to 99000. The screenshots can be observed below.

Before SQL Injection:



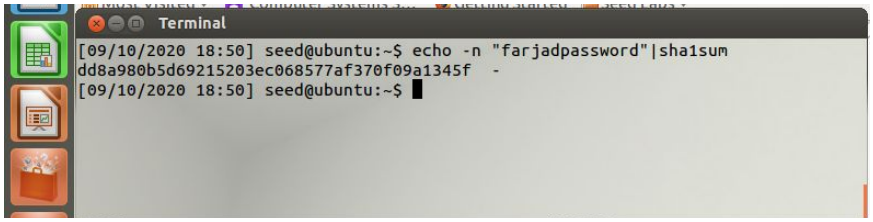
After SQL Injection:



Task 3.b: SQL Injection Attack on UPDATE Statement

(modify other people' password)

In this task, a password of any user was to be modified. For the accomplishment of this task, the initial step was to obtain the hashing value of our desired password. The hashed value can be obtained using the command mentioned in the following screenshot:



The following command was injected through Alice's input field mentioned in the "Edit Profile" settings:

```
', password=<HASH Value> where name='Ted', #
```

In the above command, I attempted to change the password of user “Ted”. Next, I logged in using the EID of Ted, that is, 50000. And, I used my own password for logging into the account, that is, “**farjadpassword**”.