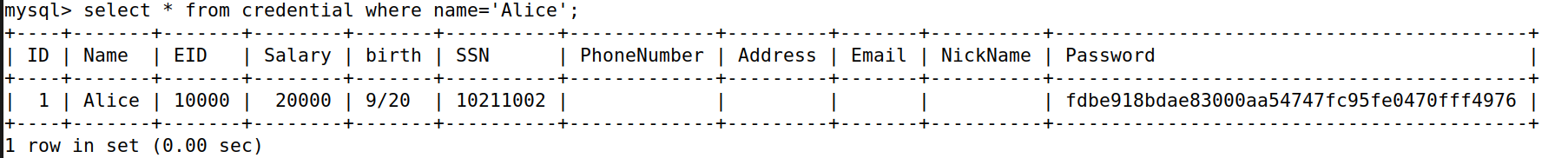
**Lab 2 : SQL Injection Attack**

Due: July 21, 2024 at 11:59 PM | Total points: 100

**Task 1 (20 Pts): Get Familiar with SQL Statements**

Question 1: After running the commands in task 1, you need to use a SQL command to print all the profile information of the employee Alice. Please provide the screenshot of your results.

**Select \* from credential where name = ‘Alice’;**

Question 2: What fields are available in the credential table of the SQL database?

**ID, Name, EID, Salary, birth, SSN, PhoneNumber, Address, Email, NickName, Password**

**Task 2 (40 Pts): SQL Injection Attack on SELECT Statement**

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описаниеQuestion 3: Provide a screenshot of a successful SQL injection attack from the webpage. (Output of task 2.1). What commands did you use for the SQL injection attack? Explain the command usage and provide observation. Command: admin’; #

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

**The command inputs admin’s username and terminates the original SQL statement.**

**It does it right before the and clause, bypassing the password validation by commenting it with #. This implies that there is no separation between data and the code.**

Question 4: Provide a screenshot of a successful SQL injection attack from the command line. (Output of task 2.2). What command did you use from the command line? Explain the command usage and provide observation.

**curl 'www.seed-server.com/unsafe\_home.php?username=****admin%27%20%23&Password='**

**This command does the same operation, except it is now done through a terminal. Here we are addressing the website, typing our user login information straight into the input files, username, and Password. However, our command also bypasses the password field, using the percent encoding, where %27 – ‘ %20 - ; %23 - #.**

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Question 5: Provide the screenshot output to show the failed execution of two SQL statements (Output of task 2.3). Also provide the statements.

**Using these two commands:**

**admin'; DELETE FROM credential where name= 'Alice'; #**

**We tried deleting the user with the name Alice**

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Question 6: What countermeasure prevented the execution of two SQL statements? Describe your observation. (Refer to SQL overview pdf provided in the Canvas), (answer related to task 2.3)

**According to the SEED manual, the website uses prepared statements, to separate the code from its data. Therefore, none of the standard SQL code would work since it is treated as data by the website rather than as the code. This means that neither user input nor potentially malicious SQL code, can impact the intended execution of the SQL statement.**

**Task 3 (30 Pts): SQL Injection Attack on UPDATE Statement**

Question 7: Provide screenshot of successful modification of your own salary. (Increase Alice’s salary, Task 3.1). What command/changes did you introduced for this step? Explain the command usage and provide observation.

Command: **', salary = 60000 where EID = 10000; #**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

**Using the command above, we managed to edit Alice’s salary from 30000 to 60000. We terminated the default SQL statement, injected our new salary value and directly targeted Alice’s account by her EID.**

**The fact that we managed to change Alice’s salary means that the Edit profile page doesn’t have any security in terms of the code and data separation, such as prepared statement.**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описаниеQuestion 8: Provide screenshot of successful modification of other’s salary. (Decrease Bob’s salary, and show Boby’s original salary, Task 3.2). What command/changes did you introduced for this step? Explain the command usage and provide observation.

**Original salary**

**Decreased salary to $1 (using Alice’s account)**

Command: **',salary = 1 where EID = 20000; #**

**Using the command above, we managed to edit Boby’s salary. We terminated the default SQL statement, injected our new salary value and directly targeted Boby’s account by his EID.**

**Same as with the previous task, this means that the Edit profile page doesn’t have any security in terms of the code and data separation, such as prepared statement. Moreover, any user can easily target other users without any access validation.**

Question 9: Provide screenshot of successful modification of other’s password. (Successful login to Boby’s account, Task 3.3). Explain the command usage and provide observation.

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описаниеИзображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описаниеOriginal

Modified password

Command: **', password = sha1('pas') where EID = 20000; #**

**In this case the command, cuts off the SQL statement, and inserts our own statement with a new password for Boby. To be more specific, it terminates the current SQL command, sets the password field to the SHA-1 hashed value of the string 'pas' and gets Boby’s profile using his EID.**

**Task 4 (10 Pts): Countermeasure – Prepared Statement**

Question 10: Provide screenshot of failed SQL injection attack after applying the prepared statement mechanism to the vulnerable codes. What changes did you make to the unsafe.php? Explain your code(Output of Task 4)

Using Boby’s new password – ‘pas’

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Using the standard SQL Injection command, did not work out this time: Boby’; #.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

**In the PHP code below, we have implemented the prepared statement approach. Instead of inserting the logging data straight into the database search, we have first prepared the statement, bound the parameters, executed the query, and then processed the results:**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание