Homework-2

Francesco Penasa - 27/09/2019

Find SQLi vulnerabilities

In order to find the SQL vulnerabilities we tested both the input fields of the script. We exploited the first field (*user-id*) using the following String. `' OR 1 = 1 /*`

```
// So the query send to the DB would be like the following.
SELECT * FROM credentials WHERE user-id = '' OR 1 = 1 /* user ' AND password = 'passwor
```

Such query select all the data from credentials and doesn't check if the password is correct. In this way we have access to all the data in the TABLE *credentials* and even in the TABLE *account*. Of course the same can be done with the second field but this will return only the data about the user specified in the first input field.

Fix the vulnerabilities.

To fix such vulnerabilities, python3 prepared statements has been used in the two pieces of code that contains the DB queries, following the piece of code of the first fix.

vulnerable

```
retrieve_user = "SELECT * FROM credentials WHERE user_id = '" + user_id + "' and passwo
cursor = conn.execute(retrieve_user)
```

fixed

```
retrieve_user = "SELECT * FROM credentials WHERE user_id = ? and password = ?;"
params = (user_id, password, )
cursor = conn.execute(retrieve_user, params)
```

While for the other piece of code the fix is substantially the same, it is necessary only to use a parameter instead of two.

fixed

```
retrieve_user = "SELECT * FROM accounts WHERE user_id = ?;"
params = (user_id, )
cursor = conn.execute(retrieve_user, params)
```