George, Levi

CS44500 – Computer Security

03/06/2023

Lab 6 – Web SQL Injection

Env Setup.

My hosts file is still previously configured from the last lab.

Text

Description automatically generated

I ran dcbuild

Text

Description automatically generated

I ran dcup

Text

Description automatically generated

I confirmed the docker containers were live

Text

Description automatically generated

I then confirmed that I could access the web page

Graphical user interface, application

Description automatically generated

TASK 1. FAMILIARIZING YOURSELF WITH SQL

I jumped into the SQL container

A picture containing timeline

Description automatically generated

I get into the mysql client

Text

Description automatically generated

I load and how the tables in the SQQLLAB\_USERS database

Text

Description automatically generated

I show the full table

Graphical user interface, text

Description automatically generated

Then I isolate Alice’s credentials and info

Graphical user interface

Description automatically generated

TASK 2 SQL INJECT VIA SELECT STMT

2.1 – Injection via web page

I entered the phrase admin’; # as the username and used that for the login. Admin ensures we are using the admin user account, the single quote closes the string being input, the semicolon closes the sql statement and the pound symbol comments the rest of the statement out.

Without closing the quotes, we get an error about the account existing.

Without the semicolon we need to provide a password

Without the pound sign we get a SQL error (see image below entry.)

Graphical user interface, application

Description automatically generated

Text

Description automatically generated

Table

Description automatically generated

Above we see the user details that we gained access to.

2.2 – SQL INJECTION FROM COMMAND LINE

I ran the following curl command, using a similar method as the previous task.

It works essentially the same, but I considered the formatting of the HTTP request.

%27 – single apostrophe, %3B – semicolon, %23 - #



I got this back from the curl command.

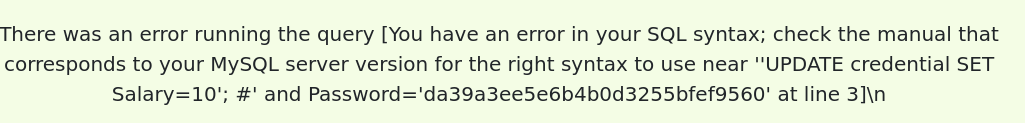
A black screen with white text

Description automatically generated with low confidence

2.3 – APPEND NEW SQL STMT



This shows that I ran the original command



However, with my appended update statement, I keep getting the previous error.

Referencing the book, this countermeasure is not something implemented by the developer but instead is baked into MYSQL itself. The API for MySQL more specifically the query() command disallows multiple statement executions for the fear of injection.

Text

Description automatically generated

TASK 3 SQL INJECTION VIA UPDATE STMT

3.1 – Modify your own salary

Using my previous methods, I gain access to Alice’s account.

I got to the edit profile page.

I enter “Ali, salary=150000; #”

Graphical user interface, application

Description automatically generated

Table

Description automatically generated

This shows the modified salary, that I set.

3.2 – Modify Boby’s salary

I ran the following command in Alice’s profile edit

Boby', salary='1' where name='boby'; #

Graphical user interface, text, application, chat or text message, email

Description automatically generated

Which changed Boby’s salary.

Table

Description automatically generated

3.3 – Modify Boby’s password

Boby', password=sha1('passwd') where name='boby'; #

I ran the above command in Alice’s profile edit

I then used it to login to Boby’s profile

Table

Description automatically generated with low confidence

TASK 4 SQL COUNTERMEASURES: PREPARED STMTS

I updated the code to include a prepared statement.

Text

Description automatically generated

Then I tried running my first attack from the beginning.

Graphical user interface, text, application, email

Description automatically generated

This is what I got returned to me,

Text

Description automatically generated with medium confidence

Which shows clearly that the defense was successful, since in a normal unprepared statement will only show alice.