MD5( ) BYPASSING THROUGH SQL INJECTION

Rakesh pv[1] ,Prof. Selwyn Paul J[2]

[1]*Department of Computer science*
St.Josephs Arts and Science (Autonomous)
Bangalore, India

[2]*Assistant Professor, Department of Computer science*
St.Josephs Arts and Science (Autonomous)
Bangalore, India

[1]*rakeshpvofficial@gmail.com*,[2]*selwynpaul438@gmail.com*

Abstract

Web-based application attacks are growing dramatically in number and severity. They found that web applications that are poorly validated and verified are susceptible to attacks by the attacker. network related SQL attacks. The MD5 (message digest algorithm) hashing method is a one-way cryptographic function that accepts any text field as input and produces a fixed-length digest value that may be used to identify the original message as output. Most security experts advise replacing the MD hash algorithm with a much more secure message digest. "Because of these collisions, a hacker or malicious user may construct files with almost the same exact hash as another, making it difficult to be certain that the file has not been interfered with. As a result, it should not be utilised for anything. Developers should instead use a Solid Cryptographic Hash function or a Symmetric Cryptographic Algorithm. this research paper demonstrates how md5 function in php can be bypassed when its parameter is set to "TRUE" ie.,[md5('x', TRUE)], this makes the hashing value(x) to be raw bytes than hexa-coded value which is much more easier to inject a SQL Statement and retrieve the original String. The above is demonstrated using SQL Fiddle in which a sample php code value fields are hashed and when its md5() is set to TRUE ,how SQLI bypasses md5().The best approach for solving above problem is to use symmetric hash function like Sha1() , sha2(),CRC which does multiple layer of hashing and when using md5() not to set its parameter to"TRUE".

.

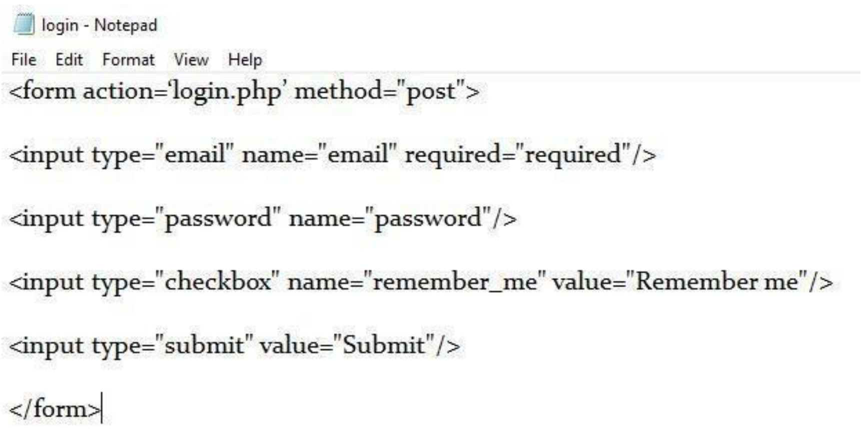
Keywords: SQLI, OWASP,, md5, Hash- basher,SHA1,SHA2 ,CRC,SQl Fiddle

1. Introduction

Web Applications are widespread today as their need becomes a necessity for lifestyle. There are thousands of security breaches that happen in a day. On websites, we feed our data which gets stored in its centralized database. We will access it from anywhere using the network. The attackers can find many tools like Botnet to generate a list of vulnerable websites. Once the web page is detected, attackers begin to steal the information [1]. The SQL injection attack web page detection is improbably done to get into the info. A hacker or an intruder will use the simple login page in a web-based application as an input entry to the vulnerable web page. Here, an attacker can inject malicious content [3] into the SQL statement to perform a login credential to log on to the web page in web application security implications depending on many mechanisms towards user authentication to save user data. The attacker typically implements the malicious codes in the form of a SQL statement such that the intruder can obtain and use the user's application. In contrast, the user does not know that the application is run by hackers using various variables

2. Demonstration of SQLI

A. Consider a form which accepts the user email and user password which gets submitted to a file named login.PHP. It uses the post method to submit data.



```
<form action='login.php' method="post">

<input type="email" name="email" required="required"/>

<input type="password" name="password"/>

<input type="checkbox" name="remember_me" value="Remember me"/>

<input type="submit" value="Submit"/>

</form>
```

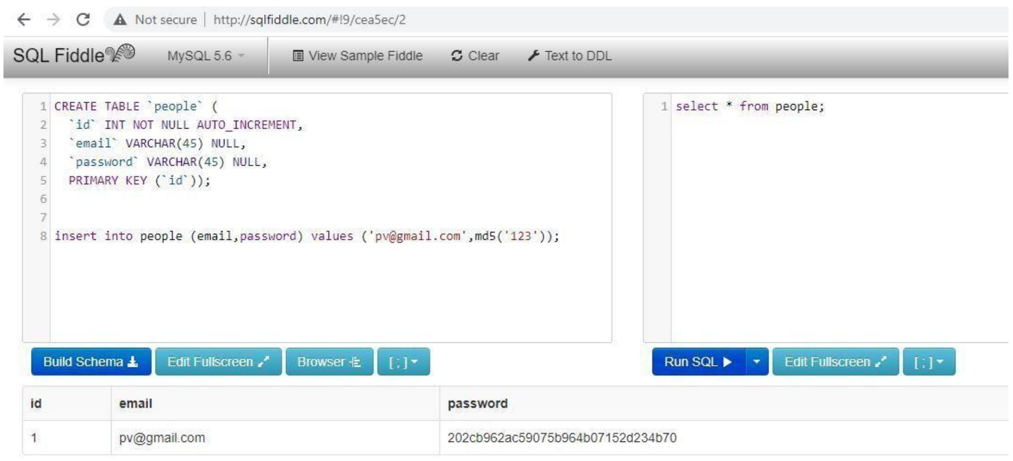
Figure 1. PHP CODE

2.1. The Statement For Checking User ID at Back End Is:

SELECT * FROM people WHERE email = $ POST['email'] AND password = md5($ POST['password']);

2.2. Suppose User Gives Input as "pv@gmail.com " and "123" As The Password.

SELECT * FROM people WHERE email = 'pv@gmail.com' AND password = md5('123');



```
CREATE TABLE `people` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `email` VARCHAR(45) NULL,
  `password` VARCHAR(45) NULL,
  PRIMARY KEY (`id`));


insert into people (email,password) values ('pv@gmail.com',md5('123'));
```

```
select * from people;
```

id	email	password
1	pv@gmail.com	202cb962ac59075b964b07152d234b70

Figure 2. SQL STRUCTURE FOR ” PEOPLE” DATABASE

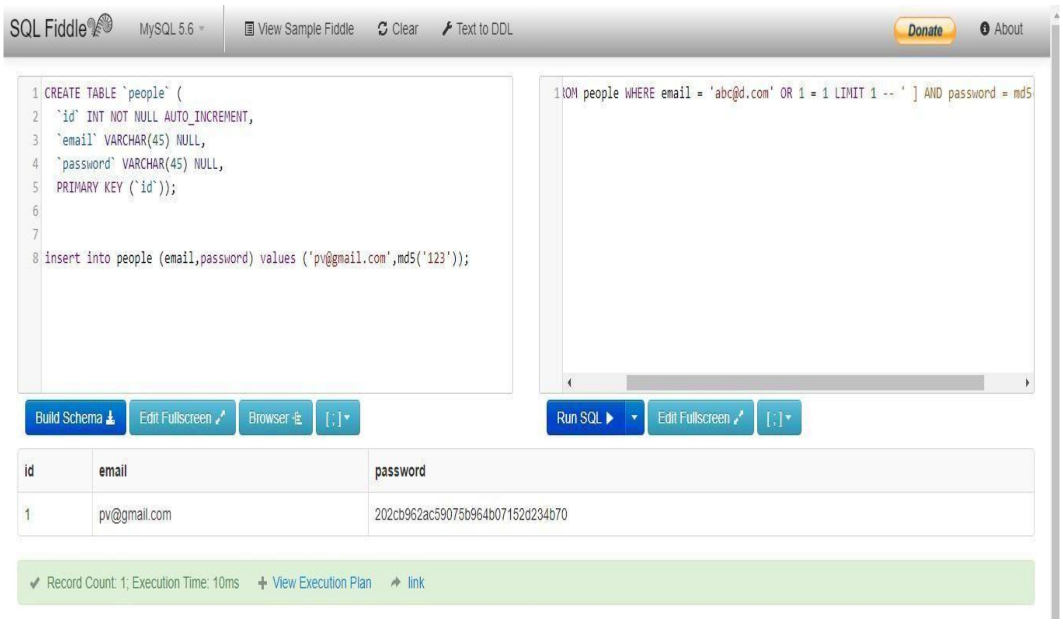
The above statement can be exploited by using comment delimiter to removed out the password part and add a new condition that will always be true.

2.3. SQL Injection Implementation

Now consider giving this statement:

SELECT * FROM people WHERE email = ‘pv@gmail@.com' OR 1 = 1 LIMIT 1 -- '] AND password = md5('123');

Here in above construct mail ends with a single quote completing the string.OR 1 = 1 LIMIT 1 is a condition which will be TRUE always.-- ' AND … is a statement which nullifies the password part



```
CREATE TABLE `people` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `email` VARCHAR(45) NULL,
  `password` VARCHAR(45) NULL,
  PRIMARY KEY (`id`));


insert into people (email,password) values ('pv@gmail.com',md5('123'));
```

```
ROM people WHERE email = 'abc@d.com' OR 1 = 1 LIMIT 1 -- ' ] AND password = md5
```

id	email	password
1	pv@gmail.com	202cb962ac59075b964b07152d234b70

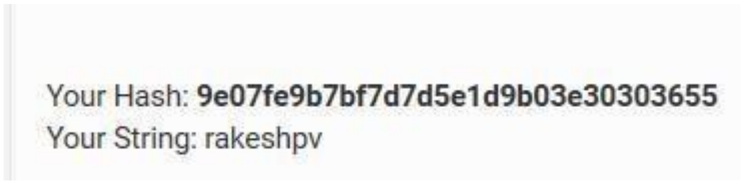
✓ Record Count: 1; Execution Time: 10ms + View Execution Plan ➚ link

Figure 3. With SQLI Injection SQL Structure

3. Md5() HASH CRYPTOLOGY

information. It is essentially a cryptography method that receives material of varying lengths and returns a set of fixed length values known as a ”digest,” which provides an authentication mechanism for the original message or content. Instead of assuming that the values of two distinct data sets are the same, MD5 creates a checksum on both data sets and verifies whether or not the checksums on both data sets are the same. The length of the md5 hashes created on the original message is 128 bits, and they are represented in a 32-digit hexadecimal number corresponding to the original message.MD5 creates non-reversible values

3.1 md5 Example



```
Your Hash: 9e07fe9b7bf7d7d5e1d9b03e30303655
Your String: rakeshpv
```

4. RAW MD5() HASH

The md5 uses raw output = proper parameter, which results in $code being raw bytes instead of a hash, which is hex coded string, and the query isn’t using prepared statement for that parameter resulting in the generation raw bytes of the MD5(code) to be populated into the string. Now an attacker can easily brute force the string using tools like Hasher Basher and perform an SQLI attack. An attacker attempts to brute force strings that would encrypt the raw result of MD5, a series that would contain an insertion of SQL to circumvent authentication used by the above query

4.1 Consider an PHP SQL query code:

Now the below query is a normal SQL statement which selects user whose mail is ‘mail’ Specified.

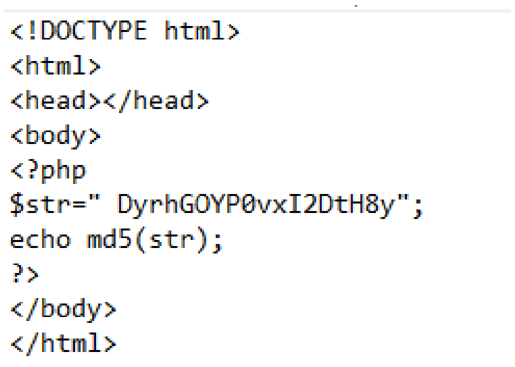
$stmt= “SELECT * FROM people WHERE mail = ‘$_POST[“mail”]'”

The statement becomes complex when we hash a user input field, here we have used hash function to password filed called code

$mail = mySQL_real_escape_string($_POST[“mail”]);
$code = md5($_POST[“code”], true);
$srmt= “SELECT * FROM people WHERE mail = ‘$mail’ AND password_hash = ‘$code

4.2 Demonstration of raw md5() bypassing:

4.2.1. .without raw md5() :consider a php code having string $str value to be hashed.



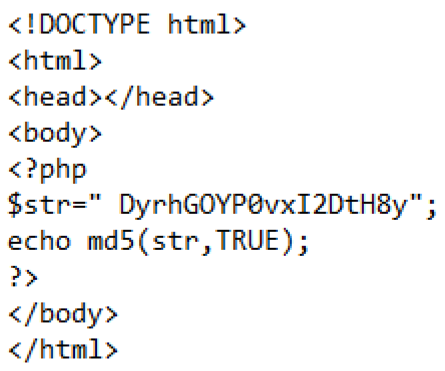
```
<!DOCTYPE html>
<html>
<head></head>
<body>
<?php
$str=" DyrhGOYP0vxI2DtH8y";
echo md5(str);
?>
</body>
</html>
```

Figure 4. PHP CODE WITH MD5()

Result :
6c0e97fda5c2252766522735b381a25b

The above query $str is sanitized with md5() and thus can't be Inject.,but when we use raw md5() we can actually perform and SQL Injection attack since its output will be raw bytes instead of digest hexa coded value

4.2.2. raw md5() :consider a php code having string $str value to be hashed.

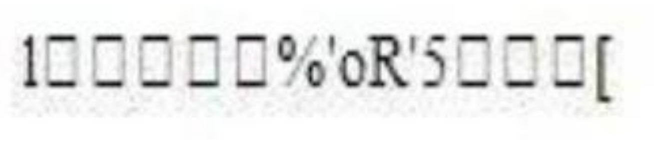


```
<!DOCTYPE html>
<html>
<head></head>
<body>
<?php
$str=" DyrhGOYP0vxI2DtH8y";
echo md5(str,TRUE);
?>
</body>
</html>
```

)

Figure 5. PHP CODE WITH RAW_MD5()

Result :



```
1□□□□□%'oR'5□□□[
```

Figure 6. HASHED VALUE WITH RAW BYTES

Here u can see the true value of the $str input, in middle of those its are " 4 , ,d,G,o,[,w " When we submit $ POST['code'] having value" DyrhGOYP0vxI2DtH8y", the final query to be executed or ends up like :

SELECT * FROM people WHERE mail = '$mail' AND password hash = '...' OR '5'

Which will be true for the where condition, so as long as a mail of a an authorized user, you can login as that user

5. CONCLUSION

SQLI attack is a pretty typical website attack that retrieves database information by identifying vulnerabilities in cross domain policies. This is usually done by modifying SQL statements used in web applications. Researchers are conscious of essential SQL injection attacks, but Numerous SQL Injection attacks are yet to be prevented and detected. md5 is one such hashing algorithm that is used to hash a file or information. It is essentially a cryptography method that receives material of varying lengths and returns a set of fixed length values known as a "digest," which provides an authentication mechanism for the original message or content. We can't revert the hash generated to get back the original string. Still, through SQLI, bypassing the injection statement, nullifying the soup, and determining the unabashed value from table input is possible. Various papers have been referred which gives brief working of MD5 hash, SQL, SQLI Attacks, how md5 is a very good hasher for protection of data. Many research papers cited that data (password or any characters) that are md5 hashed are hard to hack or retrieve its original survey. Find out is that through SQLI on POST and GET variables, we can extract individual Characters in a hash field and form a dictionary to determine hashed string. when we set md5 parameter to TRUE , the resulting string will be raw bytes instead of Hexadecimal coded string .we can easily pass an sql command to extract particular bits from resulting string which is impossible in hashed string.to justify the above statement, a simple example is demonstrated. Poorly validated statement of hash can be exploited, but stronger the mechanism and variables, it is impossible for SQLI to bypass any statement. even though md5 is strong it is now considered broken for various reason and advised to use advanced mechanism like sha1() and sha2() to overcome this issue. by conducting more research on this field we can try to mitigate this problem by making raw bytes of string to hexa-coded using another hashing mechanism on the resulting raw bytes thus i conclude by saying that we can bypass md5 when its in raw form , so a proper precaution needs to be taking while coding for prevention of any malicious attack

References

[1] A STUDY ON SQL INJECTION TECHNIQUES 1Rubidha Devi.D* , 2R.Venkatesan, 3Raghuraman.K Rubidha Devi.D* et al. /International Journal of Pharmacy Technology IJPT— Dec-2016 — Vol. 8 — Issue No.4 — 22405-22415 Page 22405 ISSN: 0975-766X CODEN: IJPTFI

[2] Tadeusz Pietraszek and Dhris Vanden Berghe., "Defending against Injection Attacks through Context-Sensitive String Evaluation", Proceedings of Recent Advances in Intrusion Detection (RAID2005)

[3] Mei Junjin, "An Approach for SQL Injection Vulnerability Detection," Proc. of the 6th Int. Conf. on Information Technology: New Generations, Las Vegas, Nevada, pp. 1411-1414, Apr. 2009

.

[4] Z. Yong-Xia and Z. Ge, "MD5 Research," 2010 Second International Conference on Multimedia and Information Technology, 2010, pp. 271- 273, doi: 10.1109/MMIT.2010.186.

[5] SQLiDDS: SQL injection detection using document similarity measure August 2016 ,Journal of Computer Security

[6] Shrivastava, Gaurav, and Kshitij Pathak. "SQL injection attacks: Technique and prevention mechanism." International Journal of Computer Applications 69.7 (2013)

[7] Andodariya, Vishal, and Shaktisinh Parmar. "A Tokenization and Encryption based Multi-Layer Architecture to Detect and Prevent SQL Injection Attack." American International Journal of Contemporary Scientific Research 2.5 (2015): 01-06