Practical - 8

Aim: - Automated SQL injection with sqlmap.

Sqlmap: -

- 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.
- Full support for **MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase, SAP MaxDB, HSQLDB and Informix** database management systems.
- Support to **directly connect to the database** without passing via a SQL injection, by providing DBMS credentials, IP address, port and database name.

Usage: -

Step #1 Start sqlmap

- First, fire up Kali and go to Applications -> Database Assessment ->sqlmap, as shown in the screenshot below.

Step #2 Find a Vulnerable Web Site

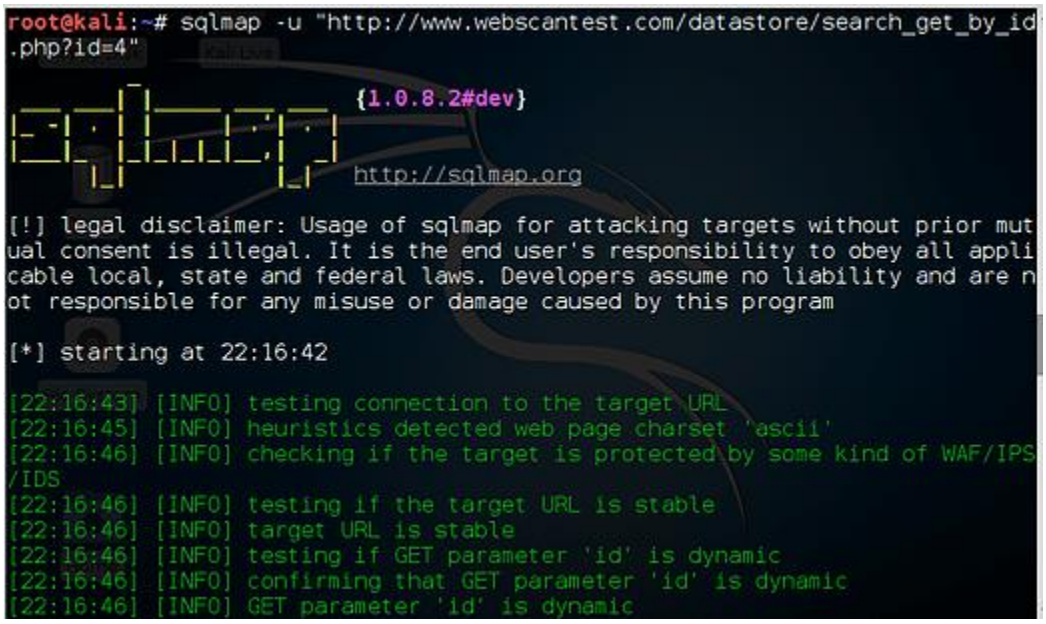
- In order to get "inside" the web site and, ultimately the database, we are looking for web sites that end in "php?id=xxx" where xxx represents some number. Those who are familiar with google hacks/dorks can do a search on google by entering:
  inurl:index.php?id=
  inurl:gallery.php?id=
  inurl:post.php?id=
  inurl:article?id=
  ...among many others.
- These dorks will bring up literally millions of web sites with this basic vulnerability criteria. If you are creative and ambitious, you can find numerous web sites that list vulnerable web sites. You might want to check these out.
- For our purposes here and to keep you out of the long reach of the law, we will be hacking a website designed for this purpose, www.webscantest.com. We can practice on this web site and refine your skills without worrying about breaking any laws and having to make bail money for you.

Step #3 Open sqlmap

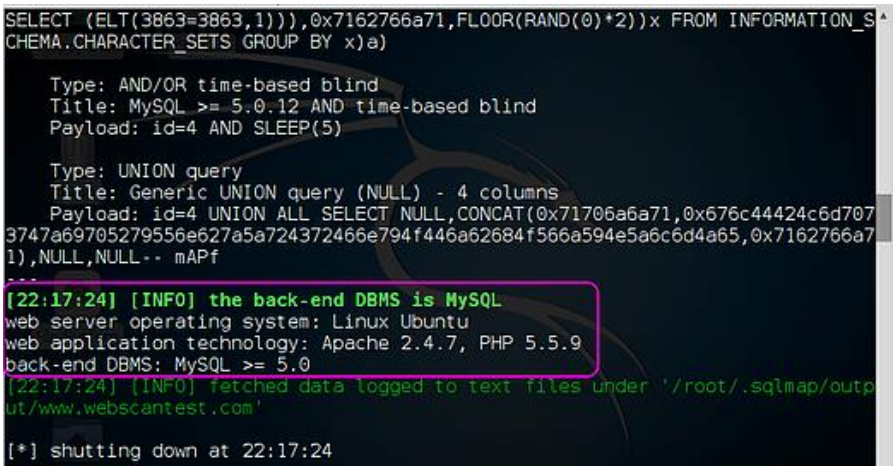
- When you click on sqlmap, you will be greeted by a screen like that below.

Step #4 Determine the DBMS Behind the Web Site

- Before we begin hacking a web site, we need to gather information. We need to know WHAT we are hacking. As I have said many times before, most exploits are very specific to the OS, the application, services, ports, etc.
- Let's begin by finding out what the DBMS is behind this web site.
- The start sqlmap on this task, we type:
- kali> sqlmap -u "the entire URL of the vulnerable web page"
  or this case:
  kali> sqlmap -u
  "http://www.webscantest.com/datastore/ search_get_by_id.php?id=4"
- Note that the entire URL is enclosed in double quotation marks (").

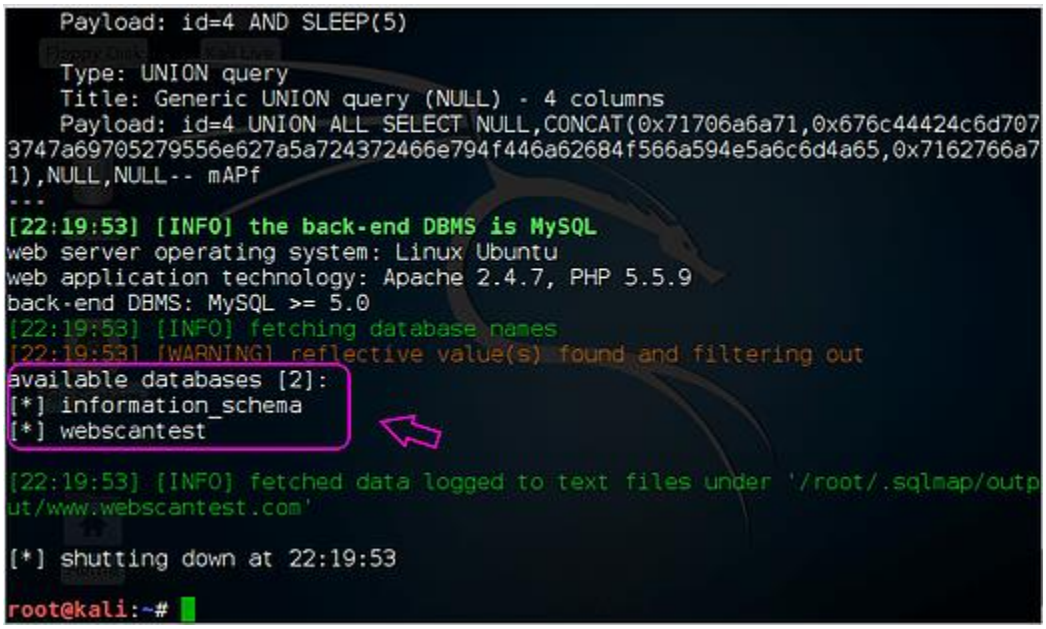


- When we do so, sqlmap will return results like that below. Notice where I highlighted that the web site backend is using MySQL 5.0



Step #5 Find the Databases

- Now that we know what the database management system (DBMS) is MySQL 5.0, we need to know what databases it contains. sqlmap can help us do that. We take the command we used above and append it with --dbs, like this:
- **kali > sqlmap -u "http://www.webscantest.com/datastore/ search_get_by_id.php?id=4" --dbs**
- When we run this command against www.webscantest.com we get the results like those below.



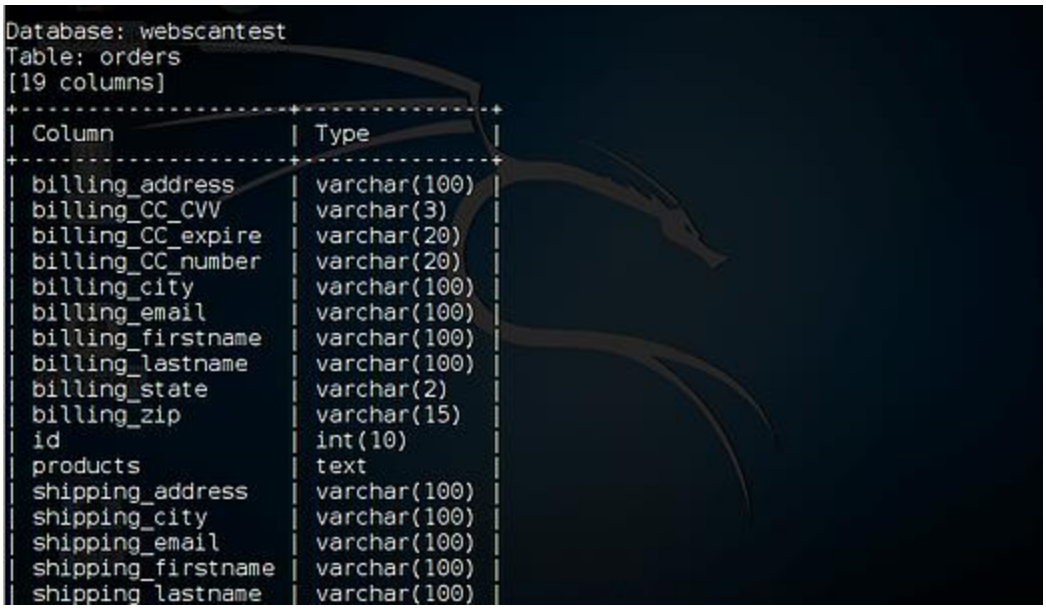
- Notice that I have circled the two available databases, information schema and webscantest. Information schema is included in every MySQL installation and it includes information on all the objects in the MySQL instance.
- Although it can be beneficial to explore that database to find objects in all the databases in the instance, we will focus our attention on the other database here, webscantest, that may have some valuable information. Let's explore it further.

Step #6 Get More Info from the Database

- So, now we know what the DBMS is (MySQL 5.0) and the name of a database of interest (webscantest). The next step is to try to determine the tables and columns in that database.
- In this way, we will have some idea of (1) what data is in the database, (2) where it is and (3) what type of data it contains (numeric or string).
- All of this information is critical and necessary to extracting the data. To do this, we need to make some small revisions to our sqlmap command.

- Everything else we have used above remains the same, but now we tell sqlmap we want to see the tables and columns from the webscantest database.
- We can append our command with --columns -D and the name of the database, webscantest such as this:
- **kali > sqlmap -u "http://www.webscantest.com/datastore/search_get_by_id.php?id=4" --dbs --columns -D webscantest**

- When we do so, sqlmap will target the webscantest database and attempt to enumerate the tables and columns in this database.
- As we can see below, sqlmap successfully was able to enumerate three tables; (1) accounts, (2) inventory, and (3) orders, complete with column names and datatypes.



```
Database: webscantest
Table: orders
[19 columns]
+--------------------+--------------+
| Column             | Type         |
+--------------------+--------------+
| billing_address    | varchar(100) |
| billing_CC_CVV     | varchar(3)   |
| billing_CC_expire  | varchar(20)  |
| billing_CC_number  | varchar(20)  |
| billing_city       | varchar(100) |
| billing_email      | varchar(100) |
| billing_firstname  | varchar(100) |
| billing_lastname   | varchar(100) |
| billing_state      | varchar(2)   |
| billing_zip        | varchar(15)  |
| id                 | int(10)      |
| products           | text         |
| shipping_address   | varchar(100) |
| shipping_city      | varchar(100) |
| shipping_email     | varchar(100) |
| shipping_firstname | varchar(100) |
| shipping_lastname  | varchar(100) |
```