SQL Injection Exploitation Techniques (DVWA)

Abstract

Introduction

In the era of rapidly growing web technologies, cybersecurity plays a key role in protecting sensitive data and ensuring secure communication. SQL Injection (SQLi) is one of the most prevalent and dangerous vulnerabilities in web applications, allowing attackers to gain unauthorized access to database contents. This project focuses on identifying and exploiting SQL Injection vulnerabilities using DVWA (Damn Vulnerable Web Application), while also referencing leading cybersecurity learning platforms and tools for support.

Problem Statement and Overview

SQL Injection vulnerabilities occur when user input is improperly sanitized and directly included in SQL queries, allowing attackers to manipulate backend database operations. The aim of this project is to demonstrate how SQLi vulnerabilities can be detected and exploited at different security levels and how this can inform the development of more secure web applications. The project also emphasizes practical, hands-on experience in ethical hacking and penetration testing within a safe environment.

Tools and Applications Used:

- **DVWA (Damn Vulnerable Web Application)**
- **Burp Suite (Community Edition)**
- **Kali Linux**
- **PortSwigger Web Security Academy**
- **CrackStation**
- **Firefox Browser along with foxyproxy**

Existing System and Proposed Plan

The existing DVWA application offers different security levels (Low, Medium, High , Impossible) for SQL Injection vulnerabilities. Each level has different input validation and security mechanisms. The proposed plan was to analyze each level by entering test payloads, intercepting traffic using Burp Suite, and crafting custom SQL queries to bypass filters. References from PortSwigger were used to understand and enhance payload crafting. During advanced stages, any hashed data retrieved was decoded using CrackStation to demonstrate full exploitation capability.

Design and Flow of the Project

User Input Form

↓

Intercept with Burp (Capture Request)

↓

Modify Payload (SQLi String)

↓

Send Modified Request

to DVWA Web Server

↓

DVWA Backend MySQL Database

↓

Retrieve Data / Hashes

↓

Crack Hash

↓

Final Output

Conclusion / Expected Output

By the end of the project, all three difficulty levels were successfully exploited using customized SQL injection payloads. The project highlighted how different levels of input validation can be circumvented using tools like Burp Suite and knowledge from platforms like PortSwigger. Cracked password hashes from the database served as a demonstration of the severity of SQLi when not properly mitigated.

The expected outcome is to increase awareness of common web vulnerabilities, promote secure coding practices, and gain practical experience in identifying and responsibly exploiting such issues for ethical hacking and cybersecurity learning.