Challenge 1

Target: 10.5.5.12 & 192.168.0.10

Status: Successfully Exploited

1. Executive Summary

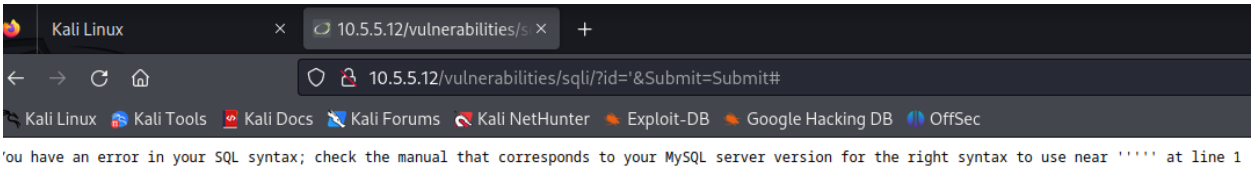
The objective of this test was to identify and exploit vulnerabilities within the DVWA environment to gain unauthorized access to user credentials and retrieve a protected challenge code. A critical SQL Injection vulnerability was discovered, allowing for a full database compromise of the user table.

2. Vulnerability Discovery: SQL Injection

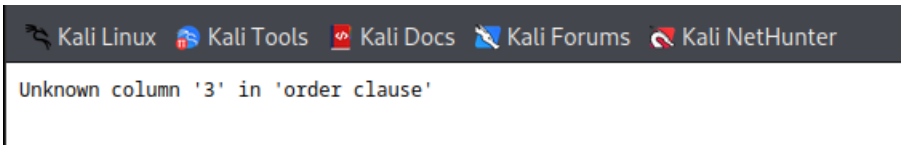
- **Vulnerability Type:** SQL Injection (Union-Based)
- **Severity:** Critical
- **Location:** 10.5.5.12 (DVWA SQL Injection Module)
- **Security Level:** Low

Steps Taken:

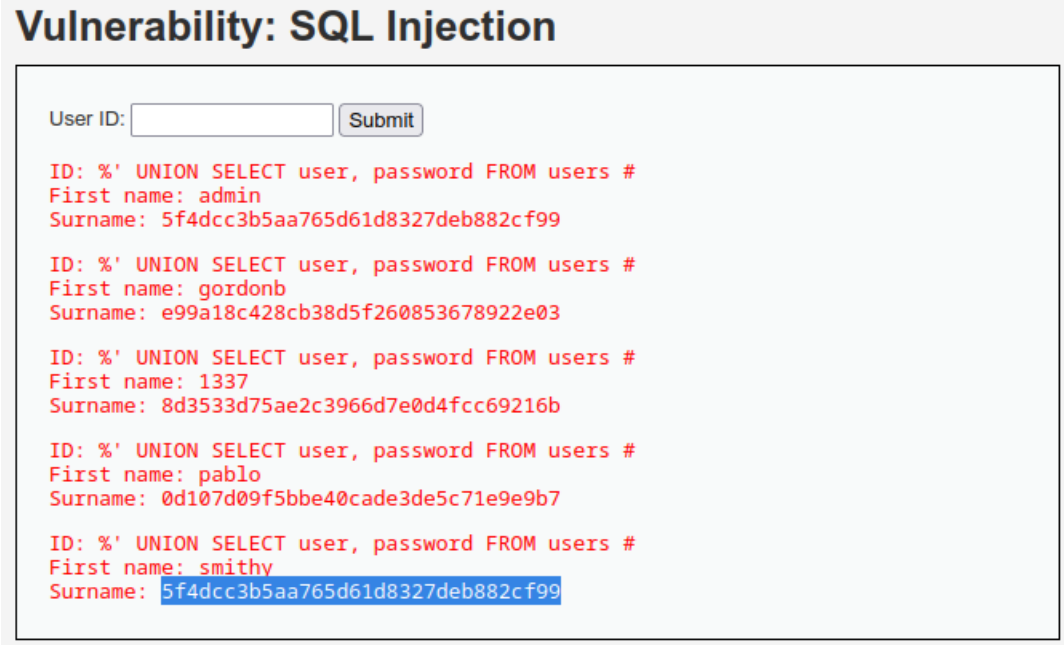
1. **Detection:** Inputting a single quote (`'`) resulted in a database syntax error, confirming the field was unsanitized.



2. **Enumeration:** Used `' ORDER BY 2 #` to determine the query returns two columns.



3. **Extraction:** Executed the payload `%' UNION SELECT user, password FROM users #` to dump the user database.

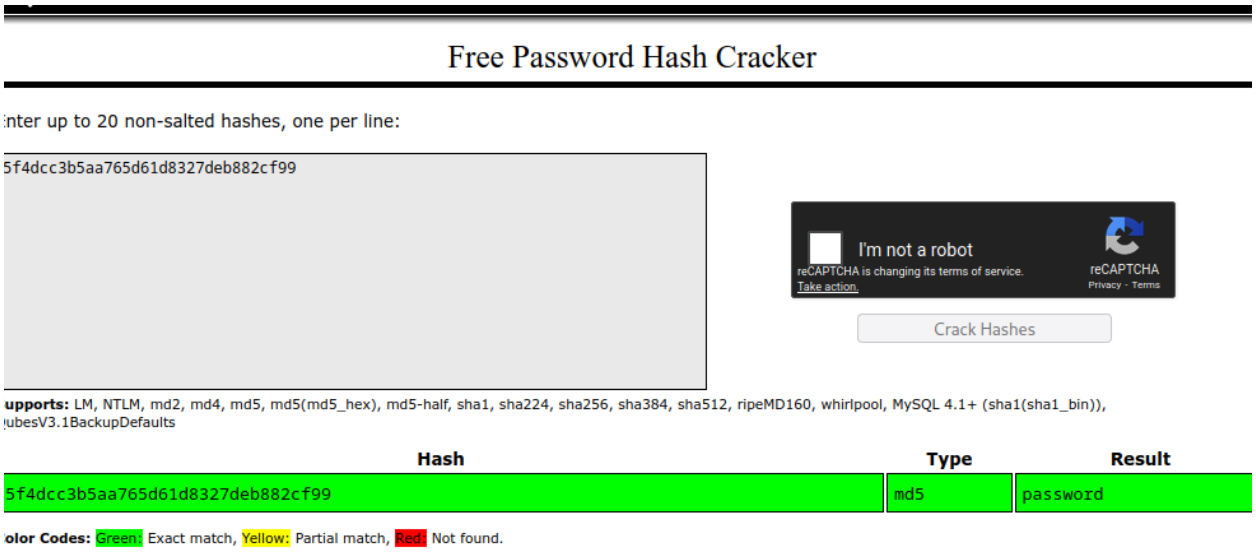


3. Credential Cracking

- **Target Account:** Bob Smith
- **Username found:** `smithy`
- **MD5 Hash:** 5f4dcc3b5aa765d61d8327deb882cf99
- **Cracked Password:** password

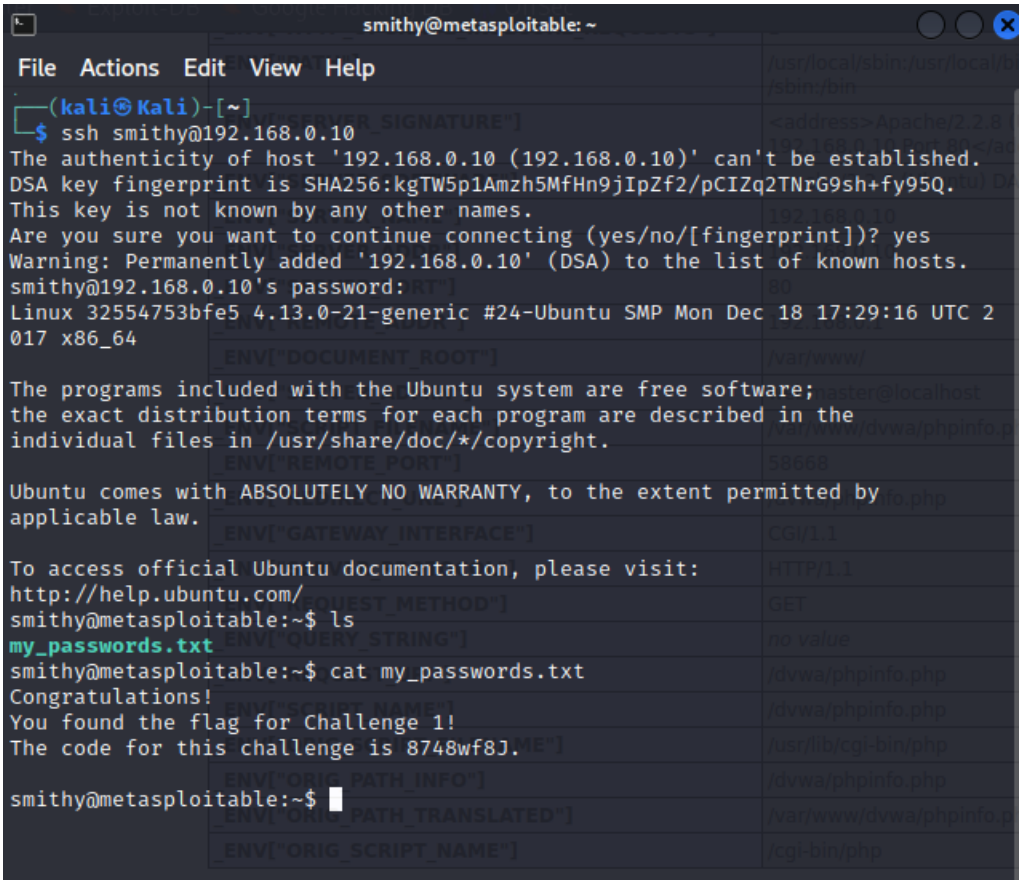
Tools Used:

- CrackStation



4. Successful Exploitation & Proof of Access

Using the cracked credentials, access was attempted at the secondary target `192.168.0.10`. The login was successful, providing access to the restricted "Challenge 1" code.



5. Remediation Recommendations

To prevent future SQL Injection attacks, the following measures are recommended:

1. **Prepared Statements:** Use parameterized queries to separate data from code.
2. **Input Validation:** Implement strict allow-lists for all user-supplied data.
3. **Least Privilege:** Ensure the database user has minimal permissions.
4. **Stored Procedures:** Utilize secure stored procedures for database interactions.
5. **Error Handling:** Disable verbose database error messages to prevent information leakage.