ISO 27001 Compliant Incident Management Report

SQL Injection Vulnerability in DVWA

1. Introduction

This report details the identification and exploitation of an SQL injection vulnerability in the Damn Vulnerable Web Application (DVWA). The test was conducted in a controlled environment to demonstrate a common vulnerability and its potential impact on application security.

2. Executive Summary

During a security assessment of DVWA, an SQL injection vulnerability was discovered in the "SQL Injection" module. This vulnerability allows an attacker to inject malicious SQL queries through the web application's input fields, thereby compromising the integrity and confidentiality of the data stored in the database.

3. System Information

- **Application**: Damn Vulnerable Web Application (DVWA)
- **Type**: PHP/MySQL web application
- **Purpose**: Training tool for security professionals
- **Environment**: Controlled testing environment
- **Current Username**: admin
- **Security Level**: low
- **Locale**: en
- **SQL DB**: mysql

4. Incident Description

During the security assessment of DVWA, an SQL injection vulnerability was discovered in the "SQL Injection" module. This vulnerability allows an attacker to inject malicious SQL queries through the web application's input fields, thereby compromising the integrity and confidentiality of the data stored in the database.

5. SQL Injection Method Used

To replicate and demonstrate the vulnerability, the following SQL payload was used in the "User ID" field:

```
' UNION SELECT username, password FROM users WHERE id = 2 #
```

This payload exploits the vulnerability to modify the original SQL query in such a way that it returns the usernames and passwords stored in the users table, specifically for the user with id = 2. By successfully executing this SQL injection, the target user's credentials are obtained without authorization.

The test confirmed that user data could be extracted, as demonstrated by the output displaying multiple user records:

- ID: 1' OR '1'='1
    - First name: admin
    - Surname: admin
- ID: 1' OR '1'='1
    - First name: Gordon
    - Surname: Brown
- ID: 1' OR '1'='1
    - First name: Hack
    - Surname: Me
- ID: 1' OR '1'='1
    - First name: Pablo
    - Surname: Picasso
- ID: 1' OR '1'='1
    - First name: Bob
    - Surname: Smith

6. Incident Impact

Exploiting this vulnerability could allow an attacker to:

- Access and extract confidential information from the database, including user credentials.
- Modify, delete, or compromise sensitive data stored in the application.

This represents a significant risk to the confidentiality, integrity, and availability of the data and services provided by DVWA.

7. Recommendations

Based on the findings of this security assessment, the following corrective and preventive measures are recommended:

1. **Input Validation**: Implement strict input validations for all user-supplied data, using secure parameters in SQL queries to prevent SQL injection.

2. **Penetration Testing**: Conduct regular security audits, including penetration tests, to identify and mitigate security vulnerabilities before they are exploited by attackers.

3. **Education and Awareness**: Train technical and non-technical staff on secure application development practices and raise awareness of the risks associated with security vulnerabilities.

4. **Use of Prepared Statements**: Implement prepared statements and parameterized queries to separate SQL code from user-supplied data.

5. **Least Privilege Principle**: Ensure that database users have only the minimum necessary privileges required for their function.

6. **Regular Updates**: Keep all software components, including frameworks and libraries, up to date with the latest security patches.

7. **Implement WAF**: Consider implementing a Web Application Firewall to provide an additional layer of protection against common web attacks.

8. Conclusion

The SQL injection vulnerability identified in the DVWA represents a critical security issue that could lead to unauthorized access to sensitive data. By implementing the recommended security measures, the risk associated with this vulnerability can be significantly reduced.

9. References

- https://en.wikipedia.org/wiki/SQL_injection
- https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/
- https://owasp.org/www-community/attacks/SQL_Injection

Report prepared by: Theresa Baker
Date: March 28, 2025 at 1:50pm, Central Time
Report ID: DVWA-SQLi-032825