Prepared By: Alexander Marquis

Supervisor: Douglas Lundin

Injection Vulnerability Mitigation

SQL Injection

Abstract

Bug# 123ABC was initially captured in the defect tracking system by an OWASP Zap scan performed by the cyber team. Bug# 123ABC was identified as a possible SQL injection vulnerability to the page residing at the base URL of <http://localhost:8080/JavaVulnerableLab/vulnerability/UserDetails.jsp?username=>. Further confirmation of the vulnerability was accomplished through more OWASP Zap scans, source code audit, and controlled exploitation of the vulnerability. The vulnerability was demonstrated to be capable of leaking users’ “about me” data without requiring proper authentication though further and more damaging exploitation was possible. This injection vulnerability stemmed from HTTP GET request parameters that are stored in the URL being directly concatenated into a SQL query that is then executed. This vulnerability has been effectively mitigated with the use of query parameterization accomplished via the usage of Java’s prepared statements. While the SQL injection vulnerability has been effectively mitigated there appear to be broken authentication issues remaining on the page. This means that the same data spills as before are still possible and that further work will be required to effectively stop all vulnerabilities of data spills on the affected page.

Table of Contents

[**Abstract** 1](#_Toc68616656)

[SQL Injection Vulnerability Detection 3](#_Toc68616657)

[Pre-Mitigation Code 4](#_Toc68616658)

[Pre-Mitigation URL 4](#_Toc68616659)

[Impacts to Company 5](#_Toc68616660)

[Confirmation of Vulnerability Mitigation 5](#_Toc68616661)

[Post-Mitigation Code 6](#_Toc68616662)

[Post-Mitigation URL 6-7](#_Toc68616663)

[**References** 8](#_Toc68616664)

SQL Injection Vulnerability Detection

Bug# 123ABC was initially detected and captured within the defect tracking system via an OWASP Zap scan performed by the cyber team. Later OWASP Zap scans performed for confirmation that the vulnerability still exists did not detect the exact same exploit, however, they did point to the possibility of a SQL injection vulnerability on the same webpage. The presence of the vulnerability was then confirmed by a source code audit and controlled exploitation of the vulnerability. The SQL injection vulnerability detected by the OWASP ZAP scan performed to confirm Bug# 123ABC on November 3rd, 2021 is shown below.Text

Description automatically generated

## Pre-Mitigation Code

The code responsible for this vulnerability was sourced to the UserDetails.jsp file residing in JavaVulnerableLab-master\src\main\webapp\vulnerability\ directory. The offending lines of code were found to be on lines eleven through thirteen of the file. This vulnerability was caused in code by the usage of HTTP parameters stored in the URL from a GET request as direct input to a SQL query. This is shown below with the username variable storing the username parameter passed in as part of the HTTP GET request being concatenated into the query.**Graphical user interface, text

Description automatically generated with medium confidence**

Pre-Mitigation URL

The User Details page when accessed via the URL <http://localhost:8080/JavaVulnerableLab/vulnerability/UserDetails.jsp?username=Anderson%27+OR+%271%27%3D%271%27+--+> will spill the admin “about me” data due to a SQL injection attack. This data should not be accessible without proper authentication and this SQL injection attack allows for direct access to the content.

A screenshot of a computer

Description automatically generated

# Impacts to Company

Data spills of users’ “about me” data can lead to a loss of trust from the user base and in turn to loss of users. Data spills caused by SQL injection attacks require time and resources to rectify and damage to users is often permanent and unrepairable. Beyond the leakage of data, data loss or corruption is also possible. This may lead to the loss of company assets that may or may not be recoverable. SQL injection attacks are number three on the OWASP 2021 Top 10 list. They are very common, and their impacts can be devastating (*A03:2021 – injection* 2021).

# Confirmation of Vulnerability Mitigation

After receiving direct approval to proceed from the project supervisor Douglas Lundin on November 5th a mitigation based on query parameterization was employed. This mitigation operates by constructing a SQL query beforehand with placeholder values and then feeding the HTTP parameters into the query as purely values to the query and not code. This effectively stops the data spills from SQL injection attacks on this page. Further OWASP zap scans run on November 11th to confirm the mitigation of Bug# 123ABC failed to find a SQL injection attack on the page at URL <http://localhost:8080/JavaVulnerableLab/vulnerability/UserDetails.jsp?username=Anderson%27+OR+%271%27%3D%271%27+--+>.

## Post-Mitigation Code

The rectified code now occupies lines two, and eleven through fourteen of the UserDetails.jsp file in the JavaVulnerableLab-master\src\main\webapp\vulnerability\ directory. The query parameterization accomplished in this mitigation is done via the use of Java’s prepared statements. This is the method recommended by OWASP in their injection prevention cheat sheet (*Injection prevention cheat sheet¶* 2021). The username parameter can now only be interpreted as a value to the SQL query and specifically the “username=?” portion of it. This effectively stops the injection of malicious SQL code on this page.

Text

Description automatically generated

## Post-Mitigation URL

The webpage no longer suffers from SQL injection attacks and this was verified via the use of the same SQL attack as before and having a lack of data spillage. This failed attack using the <http://localhost:8080/JavaVulnerableLab/vulnerability/UserDetails.jsp?username=-Anderson%27+OR+%271%27%3D%271%27+--+> URL is demonstrated in the screenshot below

A screenshot of a computer

Description automatically generated with medium confidence

## Post-Mitigation URL (Continued)

The webpage does still suffer from data spillage though due to a lack of authentication (*A07:2021 – identification* 2021). This can be shown by navigating to the home page at the URL <http://localhost:8080/JavaVulnerableLab/> and then clicking on the forum tab. From there, click on the forum users link at the bottom of the page and then click on a user link to their about me page. The data spillage is shown in the screenshot below. Of particular note is that the login tab is still available signifying that the user does not have to be logged in to get other users’ “about me data.” Further effort and resources will be required to completely stop data spills on this webpage.A screenshot of a computer

Description automatically generated with medium confidence

**References**

Open Web Application Security Project. (2021). *Injection prevention cheat sheet¶*. Injection Prevention - OWASP Cheat Sheet Series. Retrieved November 15, 2021, from https://cheatsheetseries.owasp.org/cheatsheets/Injection\_Prevention\_Cheat\_Sheet.html.

Open Web Application Security Project. (2021, September 24). *A03:2021 – injection*. A03 Injection - OWASP Top 10:2021. Retrieved November 15, 2021, from https://owasp.org/Top10/A03\_2021-Injection/.

Open Web Application Security Project. (2021, September 24). *A07:2021 – identification and authentication failures*. A07 Identification and Authentication Failures - OWASP Top 10:2021. Retrieved November 15, 2021, from https://owasp.org/Top10/A07\_2021-Identification\_and\_Authentication\_Failures/.