**6-1 Coding Assignment: Modifying Vulnerability Reporting**

Devante Crenshaw

Southern New Hampshire University

CS -305 Software Security

Professor Jonathan Norman

October 14, 2024

Abstract

This report presents the process of conducting a dependency check using the OWASP dependency scanner to identify vulnerabilities in the provided code base. The primary goal of this assignment was to assess potential security risks, reconfigure the tool to suppress false-positive results, and ensure that the code base remains secure without alarming developers unnecessarily. The report outlines the steps taken for static testing, reconfiguration of the OWASP tool using Maven, and validation through the suppression of false positives.

Introduction

In software development, integrating security measures into the DevOps lifecycle is crucial, which is often referred to as DevSecOps. Vulnerability assessments play a key role in this process. Web applications rely heavily on external libraries, which may introduce dependency vulnerabilities. This report provides a detailed account of identifying and suppressing false-positive vulnerabilities using the OWASP dependency scanner, as well as modifying the project configuration files to mitigate potential risks.

Static Testing

Configuring Maven for OWASP Dependency Check

The OWASP dependency scanner was integrated with Maven to perform a static analysis of the code base. The configuration involved modifying the pom.xml file to include the OWASP dependency check plugin. The following code snippet shows the changes made:

<build>

<plugins>

<plugin>

<groupId>org.owasp</groupId>

<artifactId>dependency-check-maven</artifactId>

<version>6.5.0</version>

<executions>

<execution>

<goals>

<goal>check</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

</build>

After running the Maven command, the initial dependency-check report identified several vulnerabilities, as shown in Appendix A.

Reconfiguration to Suppress False Positives

Identifying False Positives

The dependency check revealed vulnerabilities that were false positives, meaning they had no available fixes and did not pose immediate risks. To prevent these warnings from misleading developers in future scans, the false positives were suppressed using the following process:

Accessing the HTML Report:

The initial report was opened in a web browser.

Generating the Suppression XML:

The Suppress button next to each false positive was clicked, and the Complete XML Doc button generated the necessary XML code.

Creating suppression.xml:

A new file named suppression.xml was created in the same directory as the pom.xml file. The following is an example of the generated suppression entry:

<suppress>

<dependency>

<name>com.example:library</name>

</dependency>

<note>This library has no known fix, but it does not affect the application security directly.</note>

</suppress>

Updating pom.xml Configuration:

The pom.xml file was further updated to point to the newly created suppression.xml file:

<configuration>

<suppressionFiles>

<suppressionFile>suppression.xml</suppressionFile>

</suppressionFiles>

</configuration>

Conclusion

This assignment involved integrating the OWASP dependency check into a Maven-based project, identifying vulnerabilities, and reconfiguring the tool to suppress false positives. The reconfiguration was achieved by creating a suppression.xml file and updating the Maven configuration in the pom.xml file. The final reports confirm that the false positives were suppressed, ensuring that developers are not misled by unnecessary warnings.

References

OWASP Foundation. (2023). OWASP Dependency-Check. https://owasp.org

Apache Maven Project. (2023). Maven: The Definitive Guide. https://maven.apache.org