**Modifying Vulnerability Reporting**

Justin Starr

Department of STEM

CS 305 – Software Security

Dr. Sarah North

October 8, 2023

**Modifying Vulnerability Reporting**

The following is an assessment of the potential security vulnerabilities identified within the libraries used in conjunction with the rest-service application. To identify potential false positives, we first need to run a dependency check to identify all potential vulnerabilities. Once this has been done, we can determine which vulnerabilities are false positives. The following is a screenshot of the pom.xml file before any changes are made.

A screenshot of a computer

Description automatically generated

This also shows that the suppression configuration has not been implemented yet. The OWASP plugin has been updated to perform the most up-to-date dependency check.

A summary of the known vulnerabilities related to the rest-service application is as follows:

A screenshot of a computer

Description automatically generated

The above picture shows that there are 13 vulnerable dependencies with 118 vulnerabilities identified. We also see the first dependency listed as bcprov-jdk15on-1.46.jar. This is the bouncy-castle dependency, which is a Java library that pairs with the default Java Cryptographic Extension (Baeldung, 2023). This is the dependency that needs to be suppressed. From here we are going to create a file called suppression.xml and add the configuration into the pom so that the identified false positives will be suppressed for our final dependency-check report.

The following is a picture of the pom file with the suppression configuration added as well as a picture of where the suppression.xml file has been created.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

The suppression.xml file has been in the rest-service directory which is the same location as the pom.xml file.

Now that we have created the suppression file and the configuration for suppressions for our dependency-check we need to add all of the suppressions for the 18 CVEs that were identified and associated with the bouncy-castle dependency. The following is a screenshot of the process of adding a suppression into the suppression.xml file:

A screenshot of a computer

Description automatically generated

The picture shows that we have identified the bouncy-castle dependency. We then go to each of the associated CVEs and click suppress next to each one. A box appears with the XML information we need to add to the suppression.xml to be able to suppress this CVE and all other CVEs associated with the bouncy-castle dependency. Additionally, for the first suppression, we want to press “Complete XML DOC” because it is going to give us all the necessary initialization tags that we need for the suppression.xml file. To completely suppress the bouncy-castle dependency, we repeat the process until we have added all 18 CVEs to the suppression file.

The following is a picture showing CVEs that have been added to the suppression file:

A screenshot of a computer program

Description automatically generated

The list continues on with all 18 CVEs. These stop the alarms for all of the false positives. Now that all suppressions have been added to the suppression.xml file, we can run the dependency check again to verify that the bouncy-castle dependency has been suppressed.

The following is a picture that shows the bcprov-jdk15on-1.46.jar dependency is no longer on the summary of identified vulnerable dependencies:

A screenshot of a computer

Description automatically generated

Additionally, we see now that there are only 12 vulnerable dependencies identified instead of 13, and that 18 vulnerabilities have been suppressed. If we go to the bottom of the report, it displays a summary of all the dependencies that have been suppressed.

The following is a screenshot showing that the bouncy-castle dependency has been suppressed:

A close-up of a computer screen

Description automatically generated

This shows that we have successfully suppressed our identified false positives and that the remaining dependencies are valid.

The following is a picture showing the target location of the before and after dependency check reports:

A screenshot of a computer

Description automatically generated

The following is the link for the dependency-check-report before false positives were suppressed: <file:///C:/Users/justi/eclipse-workspace/rest-service/target/dependency-check-report-before-suppressions.html>

The following is the link for the dependency-check-report after false positives were suppressed: <file:///C:/Users/justi/eclipse-workspace/rest-service/target/dependency-check-report-after-suppressions.html>

**References**

Baeldung, W. (2023, February 9). *Introduction to BouncyCastle with Java*.

<https://www.baeldung.com/java-bouncy-castle>

OWASP. (2023). How To Read the Reports. dependency-check – How To Read The Reports.

<https://jeremylong.github.io/DependencyCheck/general/thereport.html>