CS 305 Module Two Coding Assignment Guidelines and Rubric

Overview

In this assignment, you'll conduct a dependency check. A dependency check is a type of static testing that detects vulnerabilities associated with library dependencies needed for the application. Static testing allows you to identify vulnerabilities in the code without executing the code. In this assignment, you'll do the following actions:

- Identify software security vulnerabilities by running code through a static tester.
- Identify potential mitigation techniques that have been used to mitigate vulnerabilities associated with known exploits.

Scenario

You're a senior software developer on a team of software developers. The team is responsible for a large web application that uses the Spring framework.

The software development team discussed the vulnerabilities in the code base from your manual code review and plans to mitigate them. The team also supports a new functionality that requires the addition of a new library. A best practice for making certain the code is secure is to use a dependency check to check the refactored code base and the additional library. There are tools to help with dependency checks. You'll integrate a dependency-check tool into your vulnerability assessment workflow.

Directions

To begin, open the Module Two Coding Assignment Code Base in Eclipse. Refer to the Uploading Files to Eclipse Desktop Version Tutorial for testing the code base in Eclipse. You can find the code base and tutorial in the Supporting Materials section.

Note: Integrating the static testing tool was a non-graded task that you should have completed in Module One. You may have already completed these steps.

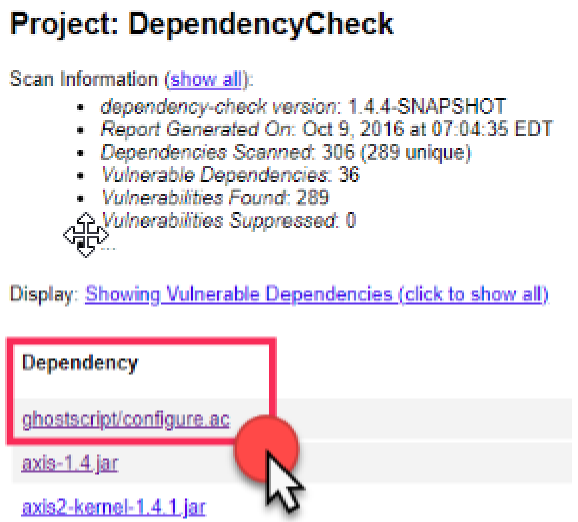
Review this module's Resources section and the supporting materials linked below to complete this assignment. After you have opened the code base, integrate the Maven dependency-check plug-in for the code base. Follow the instructions in the Integrating the Maven Dependency-Check Plug-In Tutorial linked in Supporting Materials to learn how to integrate and run the dependency-check plug-in into Maven for conducting static testing. Use the instructions in the tutorial to identify the software security vulnerabilities and document them in the Module Two Coding Assignment Template. The template is linked in the What to Submit section.

Specifically, you must address the following rubric criteria:

1. **Run the dependency check** on the code base. Include a screenshot of the resulting HTML report in your Module Two Coding Assignment Template. Make certain the screenshot includes the scan information at the top of the dependency-check report.
2. **Document the results** from the dependency check. In your Module Two Coding Assignment Template, make certain to include the codes and descriptions of each dependency that you found.

3. **Analyze the results** to identify the best solutions for addressing dependencies in the code base. Summarize your findings in your Module Two Coding Assignment Template. You can refer to industry standard guidelines, such as the Common Vulnerabilities and Exposures (CVE) and the National Vulnerability Database (NVD), by reviewing the resources in this module's Resources section.
   A. Also consider why you should filter false positives from the dependency-check tool
   B. Discuss this in the Module Two Coding Assignment Template.

To learn about the dependencies and interpret the results from the report, click on each dependency listed as shown below.



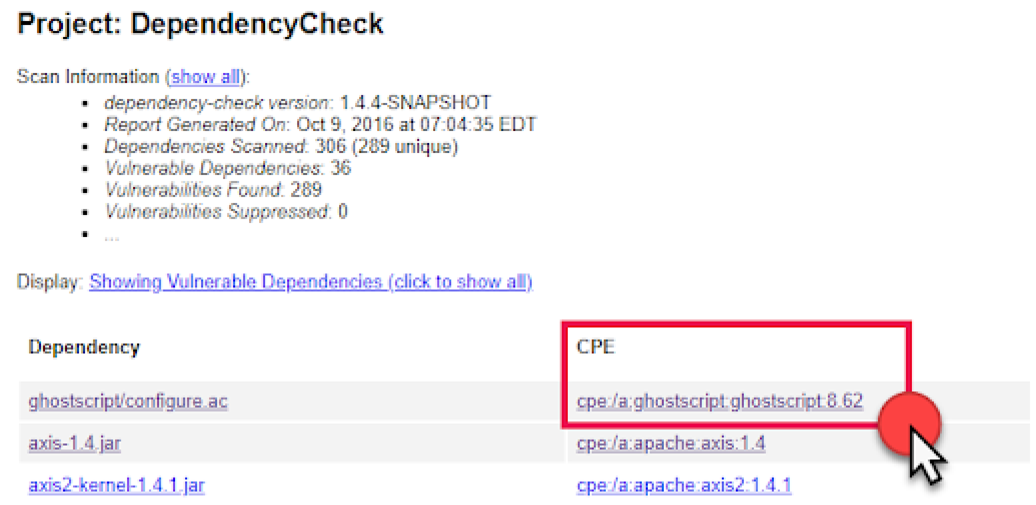
Project: DependencyCheck

Scan Information (show all):
- *dependency-check version*: 1.4.4-SNAPSHOT
- *Report Generated On*: Oct 9, 2016 at 07:04:35 EDT
- *Dependencies Scanned*: 306 (289 unique)
- *Vulnerable Dependencies*: 36
- *Vulnerabilities Found*: 289
- *Vulnerabilities Suppressed*: 0
- ...

Display: Showing Vulnerable Dependencies (click to show all)

Dependency
ghostscript/configure.ac
axis-1.4.jar
axis2-kernel-1.4.1.jar

The NVD also provides information about the dependency description, its severity, and potential solutions. You can access this information by clicking on the matching common platform enumeration (CPE) and then selecting the vulnerability ID.



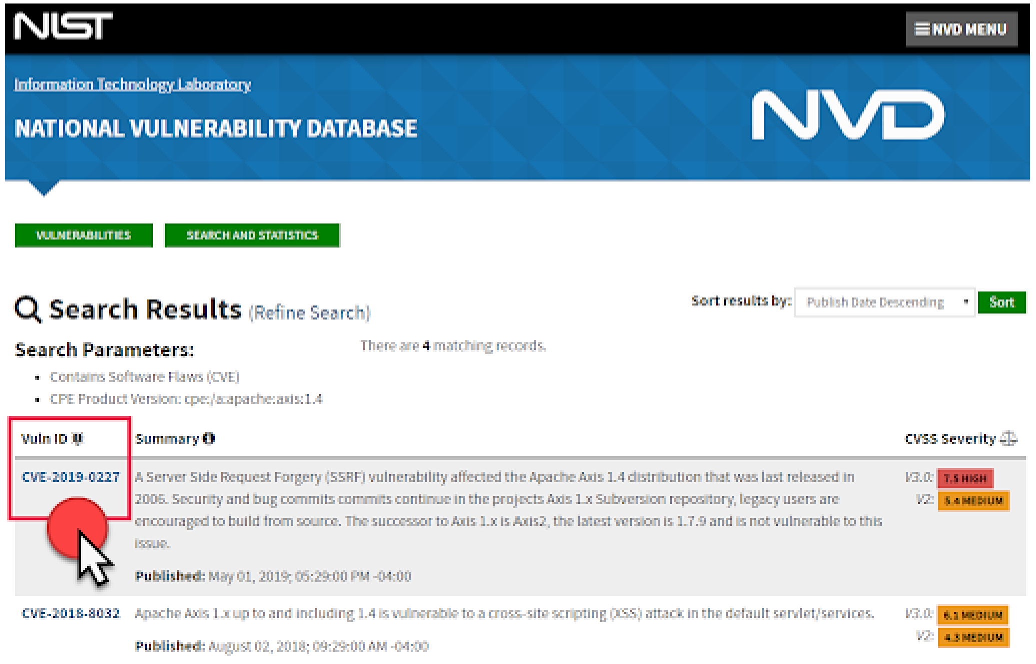
Project: DependencyCheck

Scan Information (show all):

- *dependency-check version*: 1.4.4-SNAPSHOT
- *Report Generated On*: Oct 9, 2016 at 07:04:35 EDT
- *Dependencies Scanned*: 306 (289 unique)
- *Vulnerable Dependencies*: 36
- *Vulnerabilities Found*: 289
- *Vulnerabilities Suppressed*: 0
- ...

Display: Showing Vulnerable Dependencies (click to show all)

Dependency	CPE
ghostscript/configure.ac	cpe:/a:ghostscript:ghostscript:8.62
axis-1.4.jar	cpe:/a:apache:axis:1.4
axis2-kernel-1.4.1.jar	cpe:/a:apache:axis2:1.4.1



Search Results (Refine Search)

Search Parameters:

- Contains Software Flaws (CVE)
- CPE Product Version: cpe:/a:apache:axis:1.4

There are **4** matching records.

Vuln ID	Summary	CVSS Severity
CVE-2019-0227	A Server Side Request Forgery (SSRF) vulnerability affected the Apache Axis 1.4 distribution that was last released in 2006. Security and bug commits commits continue in the projects Axis 1.x Subversion repository, legacy users are encouraged to build from source. The successor to Axis 1.x is Axis2, the latest version is 1.7.9 and is not vulnerable to this issue. **Published:** May 01, 2019; 05:29:00 PM -04:00	V3.0: 7.5 HIGH V2: 5.4 MEDIUM
CVE-2018-8032	Apache Axis 1.x up to and including 1.4 is vulnerable to a cross-site scripting (XSS) attack in the default servlet/services. **Published:** August 02, 2018; 09:29:00 AM -04:00	V3.0: 6.1 MEDIUM V2: 4.3 MEDIUM

What to Submit

Submit your completed Module Two Coding Assignment Template. Your completed assignment should be 1 to 2 pages long. Make certain to include a screenshot of the HTML output from the dependency check. Summarize the dependency-check results and potential solutions. Sources should be cited according to APA style.

Supporting Materials

The following resources support your work on this assignment:

Code Base: Module Two Coding Assignment Code Base

This ZIP file is the code base you will use for the Module Two assignments.

Tutorial: Uploading Files to Eclipse Desktop Version Tutorial

This tutorial highlights how to upload files to Eclipse.

Tutorial: Integrating the Maven Dependency-Check Plugin Tutorial

This tutorial demonstrates how to integrate the Maven dependency-check plug-in.

Module Two Coding Assignment Rubric

Criteria	Exceeds Expectations	Meets Expectations	Partially Meets Expectations	Does Not Meet Expectations	Value
Run Dependency Check	N/A	Runs the dependency check on the code base and provides a screenshot of the resulting report (100%)	Shows progress toward meeting expectations, but with errors or omissions (55%)	Does not attempt criterion (0%)	15
Document Results	Exceeds expectations in an exceptionally clear, insightful, sophisticated, or creative manner (100%)	Documents the results from the dependency-check report (85%)	Shows progress toward meeting expectations, but with errors or omissions (55%)	Does not attempt criterion (0%)	40
Analyze Results	N/A	Analyzes the results to identify the best solutions for addressing dependencies in the code base (100%)	Shows progress toward meeting expectations, but with errors or omissions (55%)	Does not attempt criterion (0%)	40
Clear Communication	Exceeds expectations with an intentional use of language that promotes a thorough understanding (100%)	Consistently and effectively communicates in an organized way to a specific audience (85%)	Shows progress toward meeting expectations, but communication is inconsistent or ineffective in a way that negatively impacts understanding (55%)	Shows no evidence of consistent, effective, or organized communication (0%)	5
				Total:	100%