# Arapahoe Community College

# CSC245 Project 5

# Scan a project with a SAST Tool

# Overview

In this project, instead of identifying vulnerabilities manually, you will use a tool to scan software. The tool we will use is Parasoft Jtest which is a plug-in for IntelliJ Ultimate.

Install license

Configure JTest to use CERT

* Open Preferences
* Select Parasoft | Configuration
* Open Builtin | Compliance Packs | Security Packs
* Select CERT for Java
* Configure where you want your reports stored

You will be scanning one of two large software packages that are freely downloadable. You will not be asked to modify the code. You can choose Netflix Conductor or Spotify Java Client or you may also choose one from this list: https://medium.com/javarevisited/8-best-popular-projects-on-java-e1a663ab3cc1.

Netflix Conductor

https://github.com/Netflix/conductor

When I ran Jtest, I discovered 1243 findings

Spotify

https://github.com/spotify/github-java-client

When I ran Jtest, I discovered 368 findings

Run a JTest scan at the project-level to identify SEI CERT vulnerabilities.

* Select project
* Right click and choose Static Analysis
* Choose Run CERT for Java
* You may have to Import the Findings and generate a report (check the box)

Perform some analysis of the vulnerabilities. Here are some ideas:

* Group the vulnerabilities by code and draw some conclusions (by author?) based on the type of vulnerability and frequency of occurrence
* Choose some small quantity of vulnerabilities (say 10) and do a deep dive into each. Identify the code pointed to by the report and build a deliverable around the code and your analysis. You don’t have to mitigate any code.

***Only individual submittals will be permitted for this assignment.***

# Deliverables

Provide your final report describing how you identified each issue. You should provide subsets of the code as needed to support your findings and improvements. Do not use screenshots.

One very popular approach is to include the original source code and then provide analysis as commentary. Note this is not sufficient as your report will also require beginning and ending sections.

Be sure your report is neat, well-organized and is well-written with minimal spelling and grammar errors. All references used should be included in your document.

# Grading rubric

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Meets** | **Does not meet** |
| Demonstrates baseline vulnerabilities | **160 points**  Imports insecure Java project into IntelliJ Ultimate edition. (10 pts)  Includes the date and time stamp when the application was run. (10 pts)  Demonstrates the code runs properly within IntelliJ environment. (10 pts)  Clearly identifies vulnerabilities in the insecure Java project with attribution to SEI CERT – use MLA format and include a References section. (130 pts) | **0 points**  Does not demonstrate successful import.  Does not include the date and time stamp when the application was run.  Does not demonstrate your code runs properly within the IntelliJ environment.  Does not clearly identify vulnerabilities. |
| Documentation and Submission | **40 points**  Document is neat, well-organized and is well-written with minimal spelling and grammar errors. (30 pts)  All references used should be included in your document. (10 pts) | **0 points**  Document is not neat, well-organized or well-written with minimal spelling and grammar errors.  References were not included. |