Lab Assignments (7 & 8)

Course: CS202 Software Tools and Techniques for CSE
Lab Topic: Vulnerability Analysis on Open-Source Software Repositories
Date: (20th & 27th) February 2025

Objective
The purpose of this lab is to familiarize yourself with **bandit**, a static code analysis tool designed to find security vulnerabilities in Python code. This lab will guide through installation, configuration and execution of **bandit** on GitHub-hosted python projects. Students will get hands-on training on improving their research communication skills.

Learning Outcomes
By the end of this lab, students will be able to:

- Understand the purpose and functionality of the **bandit** tool and setup **bandit** in local environment.
- Run **bandit** on python projects and interpret results.
- Perform a study on vulnerabilities in the open-source ecosystem.
- Learn to prepare a publication quality research report with in-depth analysis.

Pre-Lab Requirements

- Operating System: Windows/Linux/MacOS
- Programming Language: Python (setup a venv for this)
- Required Tools: latest version of **bandit** (vulnerability scanning tool)
- **Read**: https://en.wikipedia.org/wiki/Common_Weakness_Enumeration
- **Read**: https://cwe.mitre.org/about/index.html
- **Read**: https://cwe.mitre.org/top25/archive/2024/2024_cwe_top25.html
- **Read**: https://github.com/PyCQA/bandit
- **Read**: https://bandit.readthedocs.io/en/latest

Lab Activities:

1. Choose **TEN very large** scale open-source repositories to analyze with **bandit**. Make sure this is a **real-world** project (**in Python**) and not toy projects on GitHub. You must not reuse[1] the same repository (in Python) that you may have already selected in your previous assignments.
2. Define Selection Criteria:
    - Establish your own criteria for selection (inclusion/exclusion) of repositories and include this information in your report. Basically, you need to specify how you settled with the final set of selected repositories. Recall the hierarchical funnel diagram from the slides from Lecture 2.

[1] This MUST be strictly followed.

- Examples of selection criteria may include metrics such as the number of GitHub stars, forks, etc.
- You may use the SEART GitHub Search Engine to perform this task.

3. If needed, set up all the dependencies required for the repositories in isolated virtual environments for each project, separately.
4. For each project, execute `bandit` on the last 500 non-merge commits to the main branch.

5. [*Individual Repository-level Analyses*] For each repository, analyze `bandit`'s output by scanning all the python files across commits.
   (a) Report the number of HIGH, MEDIUM and LOW **confidence** issues the tool identifies per commit.
   (b) Report the number of HIGH, MEDIUM and LOW **severity** issues the tool identifies per commit.
   (c) Report the unique CWEs[2] the tool identifies per commit.
6. [*Overall Dataset-level Analyses*] Answer the following Research Questions (RQs):
   (a) **RQ1 (*high severity*)**: When are vulnerabilities with high severity, introduced and fixed[3] along the development timeline in OSS repositories?
   (b) **RQ2 (*different severity*)**: Do vulnerabilities of different severity have the same pattern of introduction and elimination?
   (c) **RQ3 (*CWE coverage*)**: Which CWEs are the most frequent across different OSS repositories?

   The answer to each RQ must be structured as follows:

Answering RQ
Purpose:
State the purpose of the RQ here as to what this RQ is doing, what aspect of evaluation it addresses.
Approach:
Here mention the steps that you have followed while answering this RQ.
Results:
Here mention results, explain plots, tables, figures. Report results from individual analyses here.
Mention the takeaway here… This should clearly highlight the quantitative overall summary to the specific statement asked in the research question.

Resources

- https://en.wikipedia.org/wiki/Common_Weakness_Enumeration
- https://cwe.mitre.org/about/index.html

[2] CWE (Common Weakness Enumeration)
[3] The definition of how a vulnerability is fixed, must be defined before answering this RQ.

- https://cwe.mitre.org/top25/archive/2024/2024_cwe_top25.html
- https://github.com/PyCQA/bandit
- https://bandit.readthedocs.io/en/latest