Security / Python Homework - SBOM

Command line tool for finding and documenting dependencies and their versions

A Software Bill Of Materials (SBOM) is a list of software (libraries, names, version numbers) which is used in some context, for example an organization, on a server, or in a consumer electronics product.

The use of SBOMs has some advantages, notably to document to end users / buyers what software they are actually using, and to answer questions like:

> *Are we affected by the new vulnerability, CVE-2023-12345678, in libawesomejson version 1.2.3?*

SBOMs can help organizations quickly determine whether they need to upgrade some software dependency, what places they need to upgrade, and whether they have successfully upgraded.

In this project you will implement a simple version of this, using Python. We expect this project to be doable in 1-2 days, but give you plenty of time to finish it. After you have submitted your project, we will have a call together to look at and discuss your solution.

Requirements for the project

- Must be uploaded to github.com as a public repository
  - Send link to us via email
- Must be written in Python 3
  - Should only use python standard library (json, csv, os, sys, hashlib, shutil)
  - No other libraries / dependencies allowed
  - Do “everything” in python, don’t use subprocesses / os.system / shell commands
- Must include a short README.md with instructions for how to run the project
  - If anything in this task description is unclear, you can make assumptions and write them down in the README.
- Documentation, tests, CI, etc. is not required, but appreciated if you have experience creating those things
- Code, comments and documentation must be in English

Example

```
$ python3 sbom.py /home/alice/code/repos/
Found 10 repositories in ‘/home/alice/code/repos/’
```

```
Saved SBOM in CSV format to '/home/alice/code/repos/sbom.csv'
Saved SBOM in JSON format to '/home/alice/code/repos/sbom.json'
```

Functionality

1. The command takes one directory as an argument
2. In this directory we expect 1 or more subdirectories which are actually source code repositories (git repositories)
3. Source code repositories must have either a requirements.txt (python) file or package.json (javascript) file.
    - You can ignore all other files / folders in the source code directories
    - Example of what requirements.txt file looks like:
        - https://github.com/cfengine/cf-remote/blob/master/requirements.txt
    - Example of what package.json file looks like:
        - https://github.com/mendersoftware/gui/blob/master/package.json
4. The command saves the SBOM as a CSV file (sbom.csv) in the directory, with at least the following columns:
    - name
    - version
    - type (npm or pip)
    - absolute path to where it was found (requirements.txt / package.json file)
5. Save the SBOM as a JSON file (sbom.json) in the directory
    - You can choose how to structure the JSON, but it must have the same information as the CSV file

Optional features

6. Implement helpful error messages and exit codes for all ways the user can use the tool incorrectly
7. For javascript projects also use the package-lock.json file to find the indirect dependencies
8. Add a git commit column to the SBOM, with the git commit of the source code repo (not the dependency). This can be obtained by running the following shell command:
    - `(cd /path/to/repo && git log --format="%H" -n 1)`
    - You are allowed to use a subprocess or os.system for this part
9. Add some additional columns of information you think is useful to the SBOM
10. Write a list of known issues / bugs in the README.md
11. Write a list of ideas for features you could work on next, that you think could be useful, and put them in the README.md