cve-bin-tool: add new data sources for vulnerabilities

About Me

Name: Rhythm Jamwal

Github: rhythmrx9

Email: rhythmrx9@gmail.com

University: Shri Mata Vaishno Devi University, Katra

Program: Bachelor of Technology in Computer Science and Engineering

Year: 2nd Year

Location: India

Timezone: Indian Standard Time (GMT +5:30)

I have experience in python, databases, and using APIs, needed to work on this project. I have been contributing to cve-bin-tool since the start of this year and have learned a lot about the structure and workings of the tool. I would love to work on this project and learn with the community.

Code Contribution

Pull Requests

Project Information

Sub-org Name: cve-bin-tool

Abstract

Currently cve-bin-tool uses the NVD database as its only source of vulnerabilities. However not all vulnerabilities are captured in the NVD and there are other sources which may also be useful in understanding the vulnerability status of a product.

This project aims at integrating additional data sources and minimising duplicate reporting of vulnerabilities due to the use of multiple databases. This will improve cve-bin-tool by increasing the amount of CVEs it covers.

This is the overview of the tasks for this project:

- Identify suitable vulnerability data sources
- Create a framework to work with multiple databases
- Write tests for framework and added data sources
- Minimise duplicate reports from multiple databases

Description

Identify suitable vulnerability databases:

Currently cve-bin-tool only uses the National Vulnerability Database, but there are other vulnerability sources that can be added to increase the scope of the tool.

The databases that I am proposing to be considered to use as data sources for vulnerabilities are:

OSV : serves as an aggregator of vulnerability databases that have adopted the OSV schema.

GitLab Advisory Database : contains the security advisories used by the GitLab dependency scanners.

Red Hat CVE Database : CVE database for Red Hat products.

These data sources consist of CVEs not in the NVD Database and can be integrated to improve vulnerability detection.

Create a framework to work with multiple databases:

Currently cve-bin-tool downloads NVD data in json format and stores it on disk in an SQLite Database. All the data sources that are being proposed to be added can be used to get data in a similar manner. To achieve this, a structure (class based) is needed that can handle downloading of data from multiple data sources and storing it on disk in a cache.

These proposed data sources contain data in different schemas, eg. databases aggregated by OSV use the OSV schema. Hence, this framework would not just be limited to fetching data, but also parse the data from different data sources to a common format.

This would allow for integration of multiple data sources with minimal changes required to other parts of the tool.

Write tests for added databases and framework:

The addition of new data sources will be test driven and the framework will also be tested to make sure it is robust in its workings.

Minimise duplicate reports:

Due to the tool using multiple databases, this makes duplicate vulnerability reports very likely.

Vulnerabilities reported in the proposed data sources have data referring to CVE IDs for eg. OSV Schema has an aliases field that gives a list of IDs of the same vulnerability in other databases. This can be used to detect and minimise duplicate reporting of vulnerabilities.

Timeline

I am familiar with the codebase, so I'll be looking to start early so that things go smoothly during the coding period. I often set pre-deadlines which help in improvising in the last minutes too.

The whole project can be broadly categorised into milestones and labels :-

Milestone 1 : Create framework to support multiple databases

- Label 1.1 : Create class based structure for framework
- Label 1.2 : Refactor CVE DB to Integrate NVD into framework
- Label 1.3 : Write tests and add documentation

Milestone 2: Add OSV Database

- Label 2.1 : Integrate Database
- Label 2.2 : Implement checking for duplicate vulnerability reports
- Label 2.3 : Write tests and add documentation

Milestone 3 : Add Gitlab Advisory Database

- Label 3.1 : Integrate Database
- Label 3.2 : Extend checking for duplicate vulnerability reports
- Label 3.3 : Write tests and add documentation

Milestone 4 : Add Red Hat CVE Database

- Label 4.1 : Integrate Database
- Label 4.2 : Extend checking for duplicate vulnerability reports
- Label 4.3 : Write tests and add documentation

Milestone 5 : Improve framework and integration

- Label 5.1 : Improve loading of databases
- Label 5.2 : Improve other parts of tool for better integration with framework

Community Bonding Period

- Interact with the members of the community.
- Discuss the design system of the project.
- Clear all the doubts, and concepts regarding my project which will help me in the coding period.
- Continue participation and solve bugs/issues.
- Set up blogs for weekly/fortnight updates. Keep my work documented.

Week I (June 13 - June 20) - Week II (June 20 - June 27)

- Implement structure for framework to work with multiple databases (Label 1.1)
- Refactor CVE DB to Integrate NVD into framework (Label 1.2)
- Add tests and documentation (Label 1.3) (Accomplishment: Milestone 1)

Week III (June 27 - July 4)

- Integrate OSV Database (Label 2.1)
- Implement checking for duplicate vulnerability reports. (Label 2.2)

Week IV (July 4 - July 11)

- Add tests and documentation. (Label 2.3) (Accomplishment: Milestone 2)
- Work on feedback/reviews.
- Write blogs about work done so far.

Week V (July 11 - July 18)

- Integrate Gitlab Advisory Database (Label 3.1)
- Extend checking for duplicate vulnerability reports. (Label 3.2)

Week VI (July 18 - July 25)

- Add tests and documentation. (Label 3.3) (Accomplishment: Milestone 3)
- Refactor and optimise, code and tests.

Phase 1 Evaluation (July 25 - July 29)

- Work on reviews.
- Refine/debug.

Week VII (July 25 - August 1) - Week VIII (Aug 1- Aug 8)

- Integrate Red Hat CVE Database. (Label 4.1)
- Extend checking for duplicate vulnerability reports. (Label 4.2)
- Add tests and documentation (Label 4.3) (Accomplishment: Milestone 4)

Week IX (Aug 8 - Aug 15)

- Look into improving mechanisms for minimising duplicate reporting.
- Update blogs about work done so far.

Week X (Aug 15- Aug 22) - Week XI (Aug 22 - Aug 29)

- Improve loading of databases. (Label 5.1)
- Improve other parts of tool for better integration with framework (Label 5.2) (Accomplishment: Milestone 5)

Week XII (August 29 - September 4)

- Update blogs about work done so far.
- Buffer week for any backlog.

Final Evaluation (September 5 - September 12)

- Finalise code and documents for final submission.
- Work on feedback/review, if any.
- Wrap up everything.

Post GSOC

- Continue contributing to the project and work on bugs/issues.

Other commitments

The GSoC timeline is mostly in sync with my university's summer break and thus will allow me ample time to work on the project. I don't have any commitments during the GSOC period and cve-bin-tool is the only organisation I'm applying to.