SBOM Sprint #3

Thomas Eby, Pranav Gonepalli, Evan Hellersund, Manel Leong,
Camron Rule, Skyler Walker, Duohan Xu, Rachel Zheng

College of William & Mary
CSCI 435: Software Engineering
October 23, 2024

Sprint Goal

- Implement SBOM parsing and tree visualization for SPDX 2.2 standard

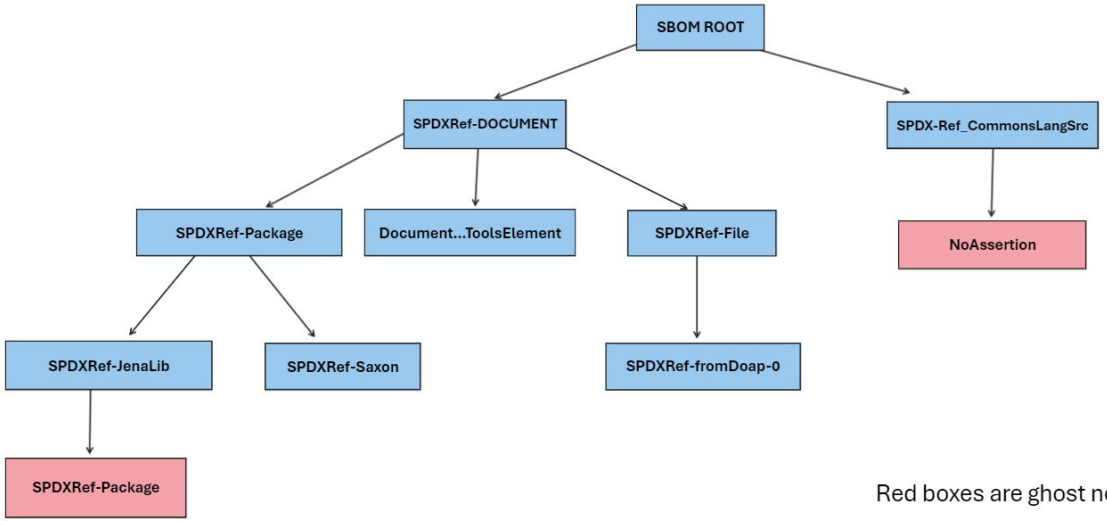
Sprint Backlog - Front end

- Set up initial tree
- Set up sidebar in initial visualization page
- Add ability to expand/collapse tree
- Connect tree to backend interface

Sprint Backlog - Backend

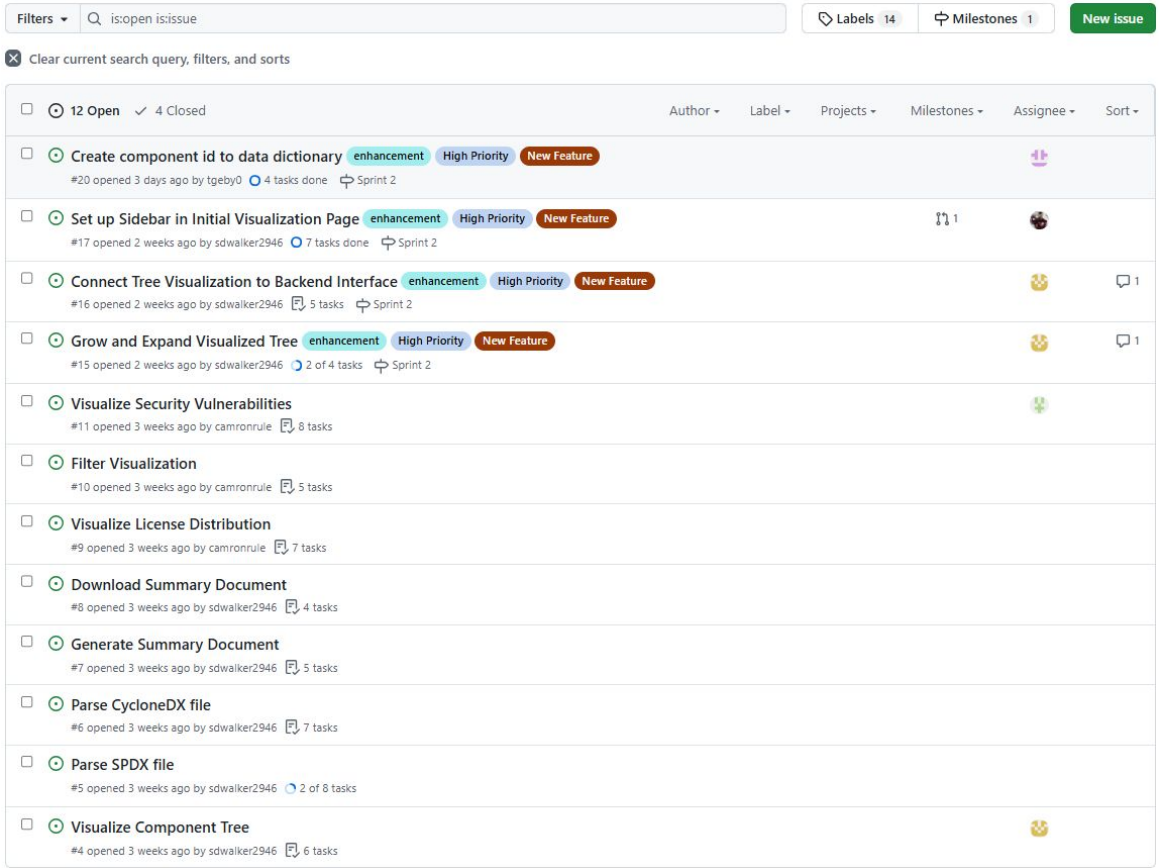
- Created mock tree data for visualization in the frontend
- Made SBOM component id to data map
- Made significant progress towards building the relationship tree from the SBOM

Sprint Backlog - Backend

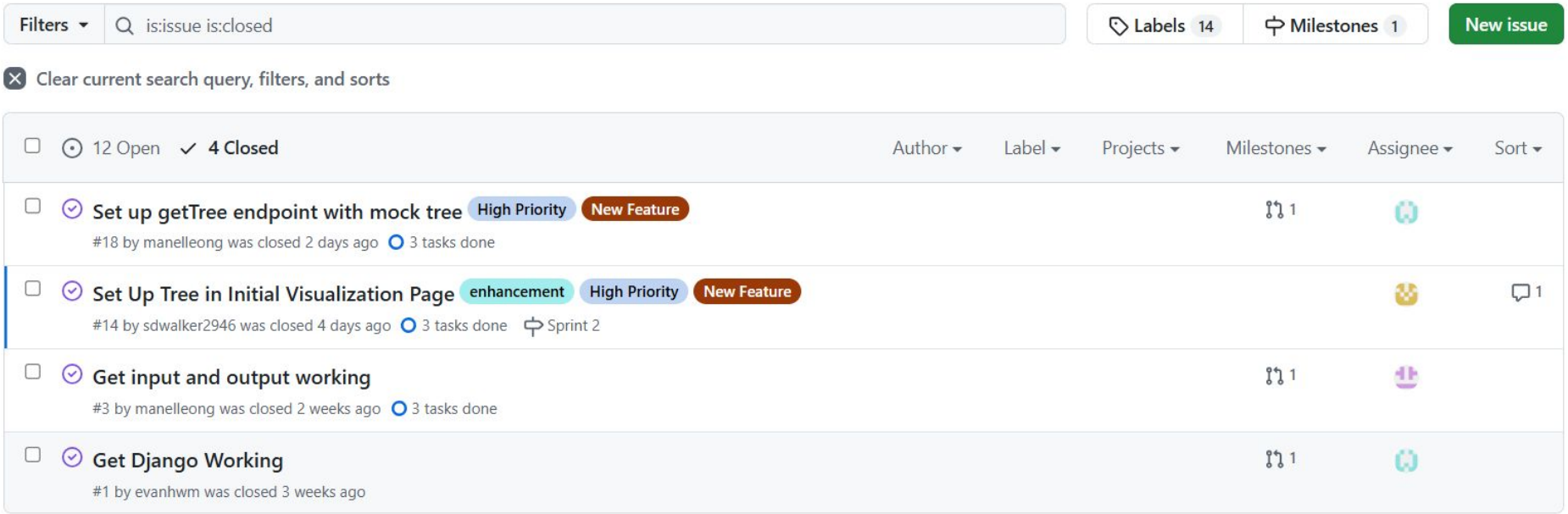


Red boxes are ghost nodes

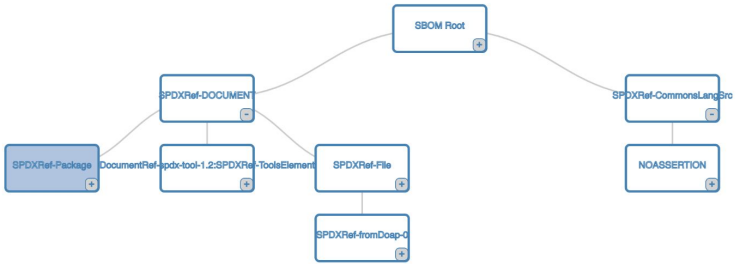
Issue Tracker



Issue Tracker (Continued)



Tree Visualization Page



Sidebar

Clear All

SPDXRef-CommonsLangSrc

This is the content for "SPDXRef-CommonsLangSrc".

NOASSERTION

This is the content for "NOASSERTION".

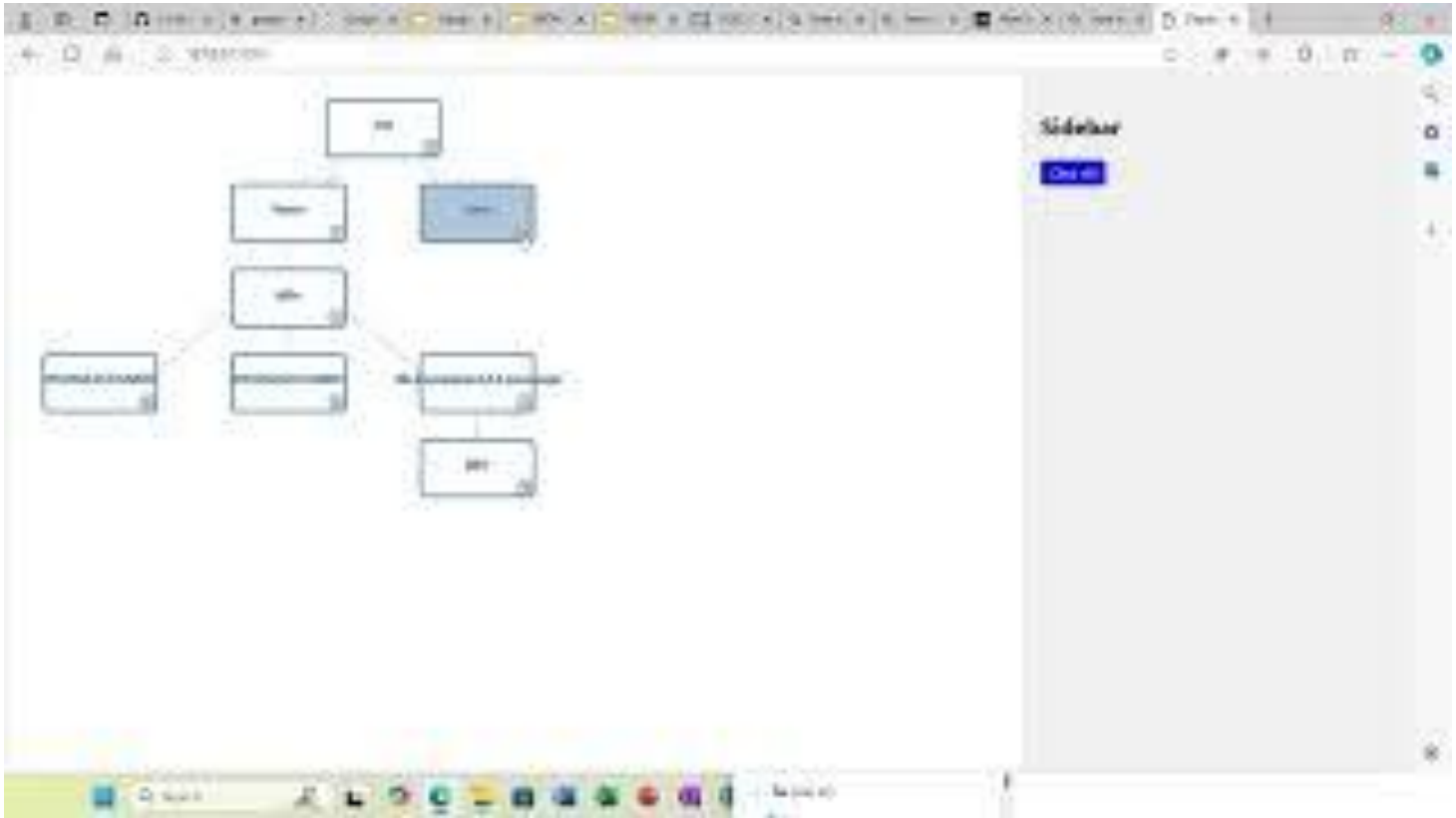
SPDXRef-File

This is the content for "SPDXRef-File".

SBOM Root

This is the content for "SBOM Root".

Demo



Scanning for Security Vulnerabilities

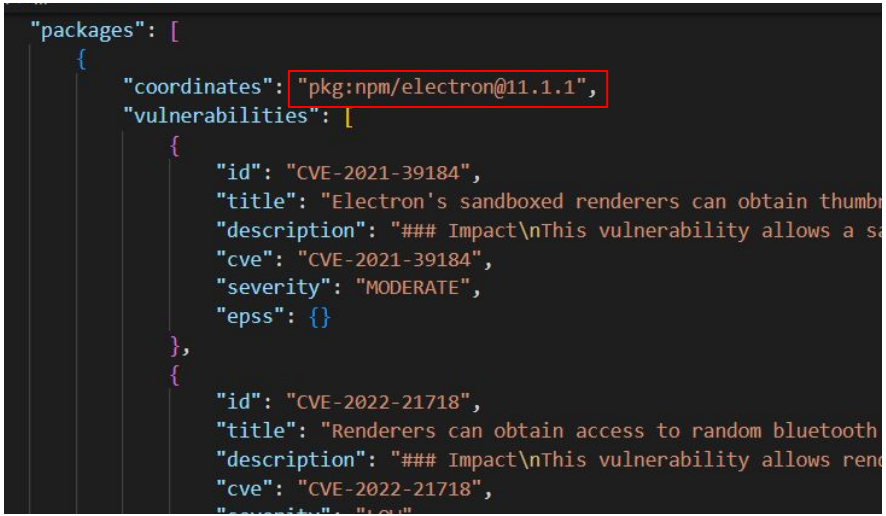
- Decided to use github repository bomber to scan for security vulnerabilities
- Bomber takes as input a sbom file and can output a json file of the security vulnerabilities it found
- Some issues and thoughts:
  - Doesn't return the exact cvss score but rather the level of severity
  - Shows the purl and related vulnerabilities but doesn't connect back to the specific sbom component (working on solution)
  - Which cvss score to use?
  - Bomber is not entirely comprehensive
  - Do we want to add other info? (ie. related cwe, links to resources, version number affected, etc)

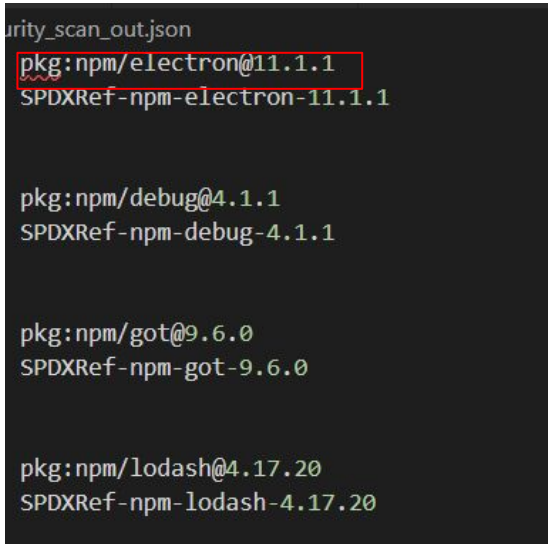
Bomber (osv version) only identifies vulnerabilities from these ecosystems:

- AlmaLinux
- Alpine
- Android
- Bitnami
- crates.io
- Curl
- Debian GNU/Linux
- Git (including C/C++)
- GitHub Actions
- Go
- Haskell
- Hex
- Linux kernel
- Maven
- npm
- NuGet
- OSS-Fuzz
- Packagist
- Pub
- PyPI
- Python
- R (CRAN and Bioconductor)
- Rocky Linux
- RubyGems
- SwiftURL
- Ubuntu OS

Bomber output example

Finding the corresponding spdx id





Next Sprint Backlog

- Generate summary document
- Create universal header for navigation to different pages
- Visualize license distribution
- Parse SBOMs ourselves
- Implement session-based storage of SBOM information

Lessons

- Improve communication between front-end and back-end so that expectations for tree visualization are agreed upon
- We need to work earlier in the week so our branches can be merged before the sprint review
    - Additionally, too many branches leads to confusion between team members
- The lib4sbom parser omits some information, so we will need to implement parsing ourselves

Contributions

Manel: Set up getTree endpoint and filled it with mock tree data

Skyler: With Camron, implemented and debugged the tree visualization. Connected frontend and backend. Also created issues, labels, and milestones.

Thomas: Developed an algorithm to build a relationship tree based on the lib4sbom parser interface. Set up endpoint for retrieving the sbom id to data dictionary.

Pranav: Implemented sidebar and combined with tree visualization page, ensuring functionality worked between the two.

Camron: Styled main page. Implemented and took part in modifying an existing D3 tree for our purposes.

Rachel: Worked on identifying security vulnerabilities given a sbom file

Evan: Worked on looking for existing tools that could be adapted and researched licensing

Duohan: Discussed over the tree structure in the backend. Not making much progress due to time conflicts

Everyone: Worked on slides and performed research