



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

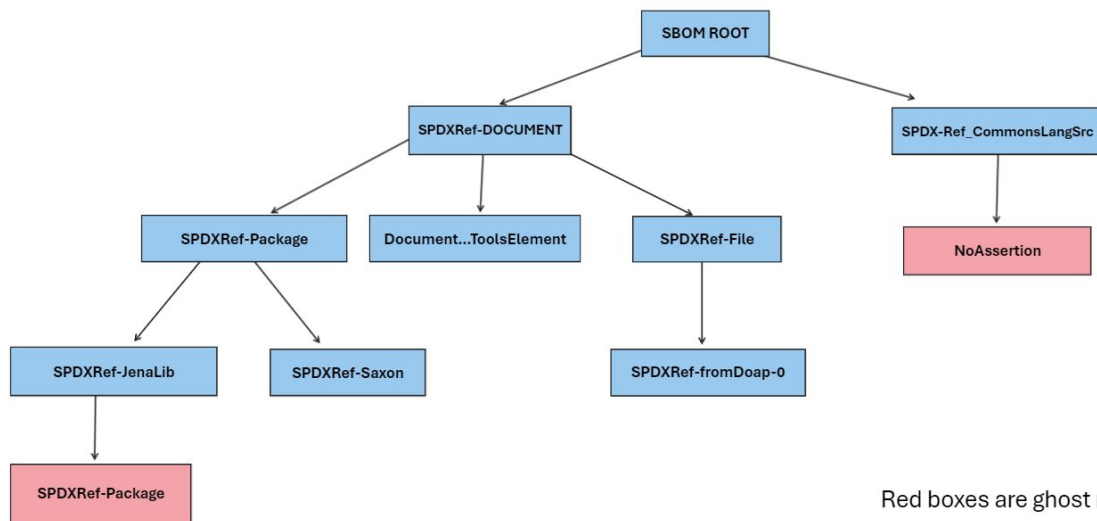
- Set up initial tree
- Set up sidebar in initial visualization page
- Add ability to expand/collapse tree
- Connect tree to backend interface



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



Issue Tracker

Filters

is:open is:issue

Labels 14

Milestones 1

New issue

Clear current search query, filters, and sorts

☐

12 Open

4 Closed

Author

Label

Projects

Milestones

Assignee

Sort

☐

Create component id to data dictionary

enhancement

High Priority

New Feature

#20 opened 3 days ago by tgeby0

4 tasks done

Sprint 2

☐

Set up Sidebar in Initial Visualization Page

enhancement

High Priority

New Feature

#17 opened 2 weeks ago by sdwalker2946

7 tasks done

Sprint 2

1

☐

Connect Tree Visualization to Backend Interface

enhancement

High Priority

New Feature

#16 opened 2 weeks ago by sdwalker2946

5 tasks

Sprint 2

1

☐

Grow and Expand Visualized Tree

enhancement

High Priority

New Feature

#15 opened 2 weeks ago by sdwalker2946

2 of 4 tasks

Sprint 2

1

☐

Visualize Security Vulnerabilities

#11 opened 3 weeks ago by camronrule

8 tasks

☐

Filter Visualization

#10 opened 3 weeks ago by camronrule

5 tasks

☐

Visualize License Distribution

#9 opened 3 weeks ago by camronrule

7 tasks

☐

Download Summary Document

#8 opened 3 weeks ago by sdwalker2946

4 tasks

☐

Generate Summary Document

#7 opened 3 weeks ago by sdwalker2946

5 tasks

☐

Parse CycloneDX file

#6 opened 3 weeks ago by sdwalker2946

7 tasks

☐

Parse SPDX file

#5 opened 3 weeks ago by sdwalker2946

2 of 8 tasks

☐

Visualize Component Tree

#4 opened 3 weeks ago by sdwalker2946

6 tasks

Issue Tracker (Continued)

Filters

Labels 14

Milestones 1

New issue

☐ Clear current search query, filters, and sorts

☐ 12 Open ☒ 4 Closed

Author ▾

Label ▾

Projects ▾

Milestones ▾

Assignee ▾

Sort ▾

☐ ☒ Set up getTree endpoint with mock tree High Priority New Feature

1

#18 by manelleong was closed 2 days ago ☐ 3 tasks done

☐ ☒ Set Up Tree in Initial Visualization Page enhancement High Priority New Feature

#14 by sdwalker2946 was closed 4 days ago ☐ 3 tasks done ☐ Sprint 2

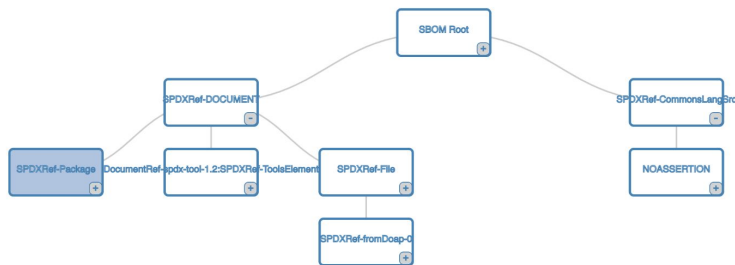
☐ ☒ Get input and output working

#3 by manelleong was closed 2 weeks ago ☐ 3 tasks done

☐ ☒ Get Django Working

#1 by evanhwm was closed 3 weeks ago

Tree Visualization Page



Sidebar

Clear All

SPDXRef-CommonsLangSrc

This is the content for "SPDXRef-CommonsLangSrc".

NOASSERTION

This is the content for "NOASSERTION".

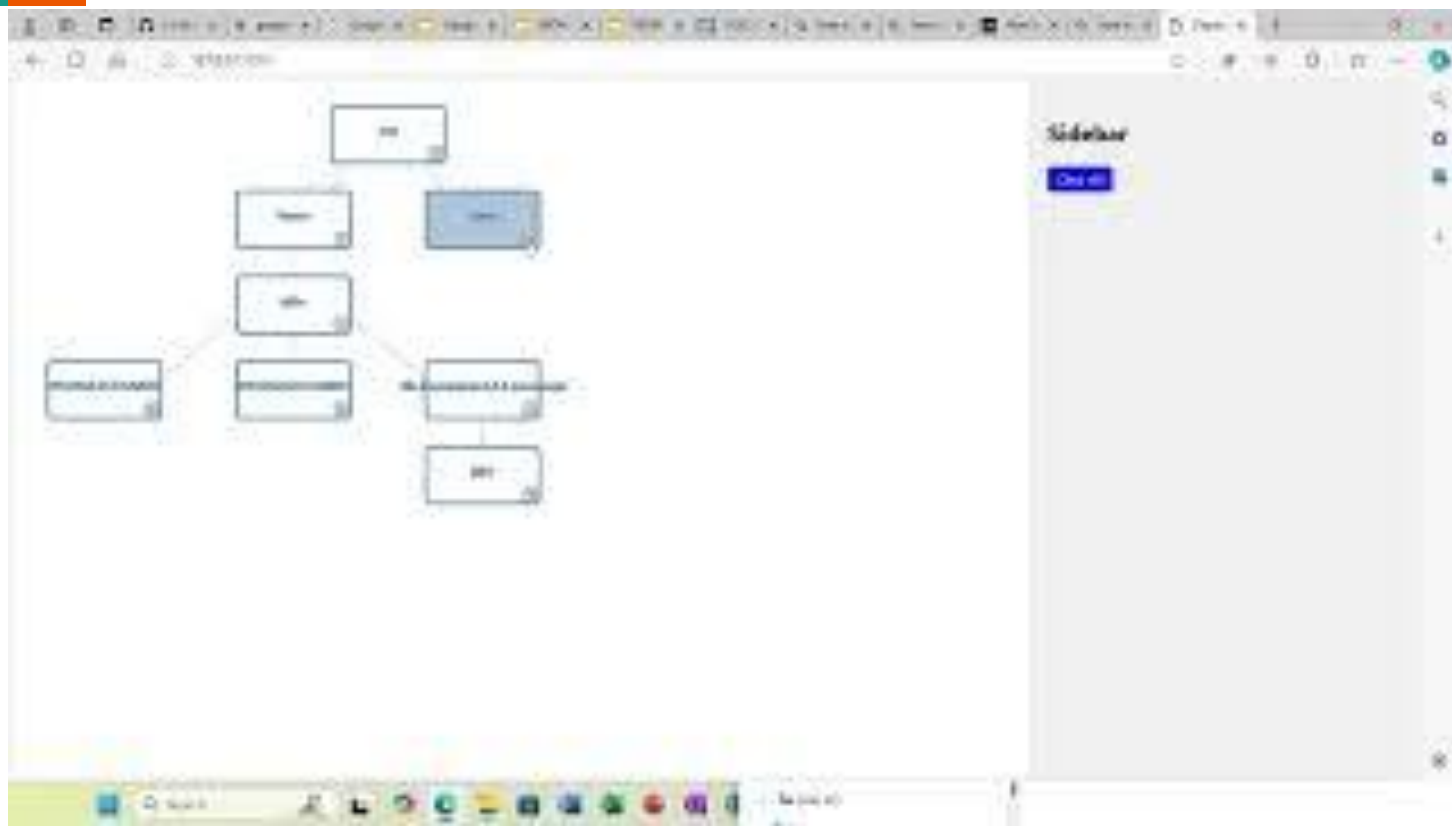
SPDXRef-File

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

```

23   "packages": [
24     {
25       "coordinates": "pkg:npm/electron@11.1.1",
26       "vulnerabilities": [
27         {
28           "id": "CVE-2021-39184",
29           "title": "Electron's sandboxed renderers can obtain thumbnails of arbitrary files through the nativeImage API",
30           "description": "### Impact\nThis vulnerability allows a sandboxed renderer to request a \"thumbnail\" image of an arbitrary file on the host system",
31           "cve": "CVE-2021-39184",
32           "severity": "MODERATE",
33           "epss": {}
34         },
35         {
36           "id": "CVE-2022-21718",
37           "title": "Renderers can obtain access to random bluetooth device without permission in Electron",
38           "description": "### Impact\nThis vulnerability allows renderers to obtain access to a random bluetooth device via the [web bluetooth] API",
39           "cve": "CVE-2022-21718",
40           "severity": "LOW",
41           "epss": {}
42         },
43         {
44           "id": "CVE-2022-29257",
45           "title": "AutoUpdater module fails to validate certain nested components of the bundle",
46           "description": "### Impact\nThis vulnerability allows attackers who have control over a given apps update server / update storage to execute arbitrary code",
47           "cve": "CVE-2022-29257",
48           "severity": "MODERATE",
49           "epss": {}
50         }
51       ]
52     }
53   ]

```

Bomber output example

Finding the corresponding spdx id

```
"packages": [  
  {  
    "coordinates": "pkg:npm/electron@11.1.1",  
    "vulnerabilities": [  
      {  
        "id": "CVE-2021-39184",  
        "title": "Electron's sandboxed renderers can obtain thumbna  
        "description": "### Impact\nThis vulnerability allows a sa  
        "cve": "CVE-2021-39184",  
        "severity": "MODERATE",  
        "epss": {}  
      },  
      {  
        "id": "CVE-2022-21718",  
        "title": "Renderers can obtain access to random bluetooth  
        "description": "### Impact\nThis vulnerability allows rend  
        "cve": "CVE-2022-21718",  
        "severity": "High"
```

urity_scan_out.json

```
pkg:npm/electron@11.1.1  
SPDXRef-npm-electron-11.1.1
```

```
pkg:npm/debug@4.1.1  
SPDXRef-npm-debug-4.1.1
```

```
pkg:npm/got@9.6.0  
SPDXRef-npm-got-9.6.0
```

```
pkg:npm/lodash@4.17.20  
SPDXRef-npm-lodash-4.17.20
```



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