SBOMs:
Software inventory für die komplette Softwarekette

Dr. Marco Bungart

Senior Software Engineer

2024-02-22

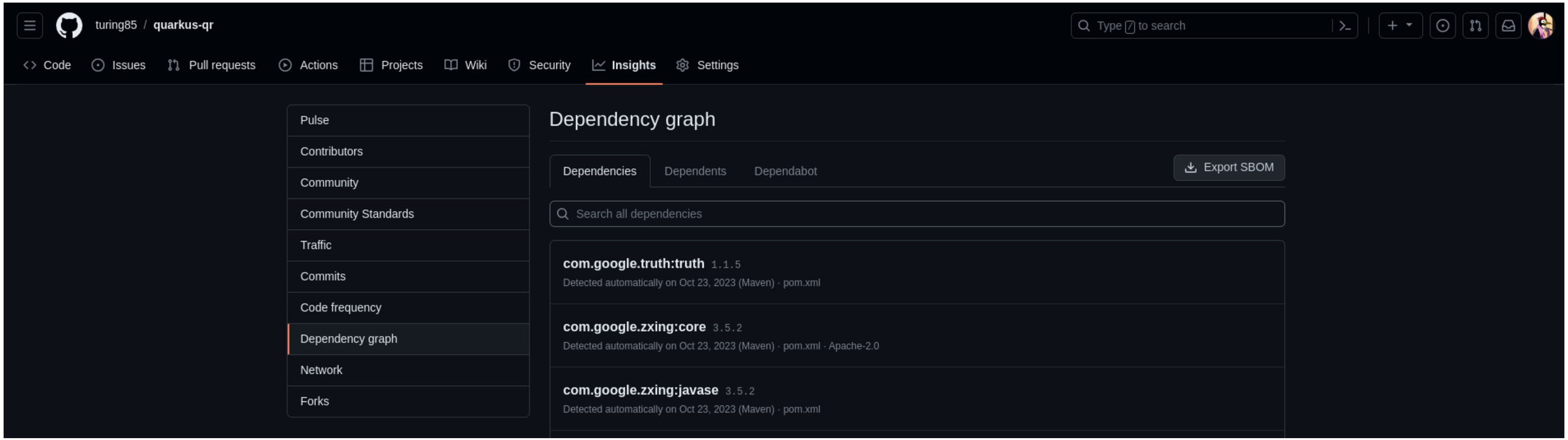
$> whoami

Dr. Marco Bungart

- 2008 – 2013: Studium Bioinformatik/Informatik in Jena
- 2013 – 2018: Promotion in Kassel
- Since 2018: Software Engineer bei ConSol
- Twitter, github, bitbucket, stackoverflow, … : turing85
- Interessen: Cloud-native Architekturen, GraalVM, Quarkus, Observability

Eine kleine Zeitreise

Eine andere Art der Zeitreise



- Was ist es?
- Was kann es (was andere nicht können)
- Wie kann ich es nutzen?

SBOM: Was ist es?

SBOM: Was ist es?

- Software Bill Of Materials
- (Sprach- und Framework) unabhängige Beschreibung der Software-Komponenten
- Verschiedene Formate: CycloneDX, SPDX, CPE
- Verschiedene Encodings: XML und JSON

SBOM: Was kann es (was andere nicht können)?

SBOM: Was kann es?

- Sprach- und Framework- unabhängige Beschreibung der Software-Komponenten
- Auch für OS-Komponenten geeignet
- Insbesondere: Für Container geeignet

SBOM: Wie kann ich es nutzen?

SBOM: Wie kann ich es nutzen?

Drei Unterkategorien:

- Wie kann ich eine erzeugen?
- Wie kann ich sie konsumieren?
- Wie kann ich sie publizieren? (behandeln wir heute nicht)

SBOM: Wie kann ich es nutzen?

Wie kann ich eine erzeugen?

- Verschiedene Tools:

  Sprachspezifisch, z.B.:

  - CycloneDX Maven Plugin
  - Cyclone-node-npm

  - Sprachunabhängig, z.B.:
    - syft und grype von Anchore (ersteres wird von docker genutzt)
    - trivy von Aquasecurity
    - cyclonedx-cli

SBOM: Wie kann ich sie nutzen?

Wie kann ich sie konsumieren?

- docker scout Subkommando
- Erweitern der SBOM, wenn ich Komponenten hinzufüge
- Vulnerability-Scan:
  - grype von Anchore (CLI)
  - dependencytrack Web UI (nur cyclonedx, Plugins für z.B. Maven)
  - iqsererver von Sonatype

SBOM: dependencytrack ohne Dependency-Tree

SBOM: dependencytrack mit Dependency-Tree

SBOM: dependencytrack kombinierte BOM

Weiterführende Literatur

Weiterführende Literatur

- https://github.com/anchore/syft
- https://aquasecurity.github.io/trivy
- https://github.com/anchore/grype
- https://docs.docker.com/engine/sbom/
- https://docs.docker.com/scout/
- https://cyclonedx.org/
- https://github.com/sigstore/cosign
- https://www.endorlabs.com/blog/sbom-vs-sbom-comparing-sboms-from-different-tools-and-lifecycle-stages
- https://www.endorlabs.com/blog/how-to-quickly-measure-sbom-accuracy-for-free

Hilfreiche Kommandos

Hilfreiche Kommandos

- Erzeugen einer SBOM für einene Container mit grype:

```
grype \
  <image-name> \
  --output cyclonedx-json \
  --file sbom.json
```

Hilfreiche Kommandos

- Erzeugen einer SBOM für ein maven-basiertes Projekt:

```
mvn \
  --also-make \
  --define projectType=application \
  --define schemaVersion=1.5 \
  --define outputFormat=json \
  --define outputName=sbom/sbom \
  org.cyclonedx:cyclonedx-maven-plugin:2.7.11:makeBom
```

Hilfreiche Kommandos

- Erzeugen einer SBOM für ein maven-basiertes Projekt und Analyse durch dependency-track:

```
mvn \
  --also-make \
  --define projectType=application \
  --define schemaVersion=1.5 \
  --define outputFormat=json \
  --define outputName=sbom/sbom \
  \
  --define dependency-track.dependencyTrackBaseUrl=<url-to-depenency-track> \
  --define dependency-track.apiKey=<api-token>  \
  --define dependency-track.bomLocation=target/sbom/sbom.json \
  --define dependency-track.projectName=\${project.groupId}:\${project.artifactId} \
  --define dependency-track.projectVersion=\${project.version} \
  --define dependency-track.failOnError=true \
  \
  --define findingThresholds.critical=0 \
  --define findingThresholds.high=0 \
  --define findingThresholds.medium=0 \
  --define findingThresholds.low=10 \
  --define findingThresholds.unassigned=0 \
  \
  org.cyclonedx:cyclonedx-maven-plugin:2.7.11:makeBom \
  io.github.pmckeown:dependency-track-maven-plugin:1.7.0:upload-bom \
  io.github.pmckeown:dependency-track-maven-plugin::1.7.0:findings
```

Hilfreiche Kommandos

- Zusammenführen mehrerer SBOMs zu einer:

```
cyclonedx \
  merge \
  --input-files <sbom1> <sbom2> … <sbomN> \
  --hierarchical \
  --group <group-name-here> \
  --name <name-here> \
  --output-format json \
  --output-file <output-file-name-here>
```

Fragen?

Danke!

ConSol
Consulting & Solutions Software GmbH

Office Düsseldorf
Kanzlerstraße 8
D-40472 Düsseldorf
Deutschland
Tel.: +49-89-45841-100
Marco.Bungart@consol.de
www.consol.de
Twitter: @turing85