
Working
Connections
IT Faculty Development Institute
July 15-19, 2024
NITIC
National Information Technology
Innovation Center

Protect Your Python

Summer Working Connections 2024
Lunchtime Presentation

Prof. Pamela Brauda

Prof. David Singletary

Agenda

1. Open Source Software risks
2. The Software Bill of Materials (SBOM)
3. Open source scanning tools
4. Using scanning tools and SBOMs to enhance security and compliance
5. Example
6. Conclusion/Q&A

Open Source Software Risks

- Vulnerabilities can occur in any programming language
- Python is a great example because it is so widely used, especially in data science
  - People of all skill levels are using it, so risks are more pronounced
- Examples of recently reported vulnerabilities in open source Python tools:

  https://thehackernews.com/2024/02/new-malicious-pypi-packages-caught.html

  New Malicious PyPI Packages Caught Using Covert Side-Loading Tactics

  https://www.sonatype.com/blog/top-8-malicious-attacks-recently-found-on-pypi

  RAT (Remote Access Trojan) Mutants

  PyTorch Namespace Confusion Attack

  GTA 5 Multihack Site

The Software Bill of Materials (SBOM)

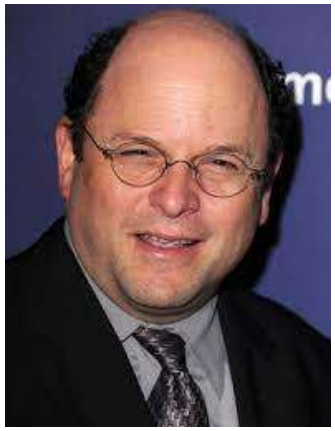
- U.S. Executive Order on Improving the Nation's Cybersecurity (14028): "Understanding the supply chain of software, obtaining an SBOM, and using it to analyze known vulnerabilities are crucial in managing risk."
- A **Software Bill of Materials (SBOM)** is a detailed inventory of all components, libraries, and dependencies used by a software application
- It provides a comprehensive record which lists open-source, proprietary, and third-party components
- It contains component metadata, including version numbers, licenses, and source information
- SBOMs promote visibility into the software supply chain
- Used in conjunction with scanning tools to identify components with known security issues

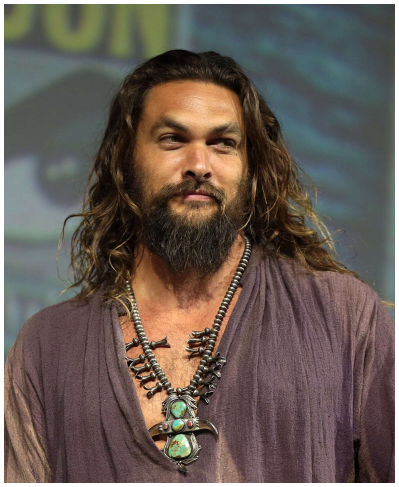
Generating an SBOM

- Popular SBOM generators include **CycloneDX**, **SPDX**, **OWASP Dependency-Track**, **Syft**, **Anchore**, and **FOSSA**
- Output formats vary, but JSON (JavaScript Object Notation, pronounced "Jason") is one of the most popular
- JSON is a lightweight, text-based, human-readable format used to represent data as key-value pairs and arrays
- https://www.json.org/json-en.html
- A simple JSON example

```
{"name": "John Doe",  "age": 30,  "city": "New York"}
```

More JSON examples





Open Source Scanning Tools

- Scanning tools examine software codebases to identify open-source components and their licenses
- The tools find known vulnerabilities in open-source libraries and dependencies
- They also ensure compliance with open-source licenses and legal requirements
- The tools assess and manage potential risks associated with using open-source software
- Popular scanning tools include **Sonatype Nexus IQ**, **Snyk**, **Black Duck**, **OWASP Dependency-Check**, **WhiteSource**, **Trivy**, and **Clair**
- U.S. Executive Order 14028 (again) mandates the verification of open source software components using these types of tools

Scanning + SBOMs For Security and Compliance

- Create the project SBOM (includes components, dependencies, and metadata)
- Configure the scanner to use the generated SBOM
- The scanner cross-references SBOM data with vulnerability databases to identify known issues

  e.g., CVE (Common Vulnerabilities and Exposures), National Vulnerability Database (NVD), Aqua Vulnerability Database, OSS Index, GitHub Advisory Database, Snyk Vulnerability Database

- A report is generated highlighting vulnerabilities and providing actionable insights for remediation and updates

Example/Walkthrough

- The following example demonstrates how to perform a scan for a Python program which uses TensorFlow, a widely-used open source machine learning library.
    1. The scenario is that we are developing a Python application which uses several popular data science libraries (numpy, pandas, etc.)
    2. A JSON-based SBOM is created using the **cyclonedx** generator library
    3. The **trivy** scanner is executed against the generated SBOM to identify known vulnerabilities in the installed modules

```
c:> type sbom.json
{
    "components": [
        ...
        {
            "bom-ref": "BomRef.8266599203400378.1777748697270512 5",
            "name": "scikit-learn",
            "purl": "pkg:pypi/scikit-learn@1.3.1",
            "type": "library",
            "version": "1.3.1"
        }
    ],
```

```
"dependencies": [
        ...
        {
            "ref": "BomRef.8266599203400378.17777486972705125"
        }
    ],
    "metadata": {
        "timestamp": "2024-07-10T18:00:27.973509+00:00",
        "tools": [
            {
                "name": "cyclonedx-python-lib",
                "vendor": "CycloneDX",
                "version": "7.5.1"
            }
        ]
    },
```

```
    "serialNumber": "urn:uuid:75861a84-ed1e-40a2-bb8f-3d634297a627",
    "version": 1,
    "$schema": "http://cyclonedx.org/schema/bom-1.3a.schema.json",
    "bomFormat": "CycloneDX",
    "specVersion": "1.3"
}
```

- "serialNumber": "urn:uuid:75861a84-ed1e-40a2-bb8f-3d634297a627" is a globally unique identifier for the SBOM
- "$schema": "http://cyclonedx.org/schema/bom-1.3a.schema.json" is a URL which refers to the structure and validation rules for the document

```
C:\> trivy sbom sbom.json
2024-07-10T14:02:01-04:00 INFO Vulnerability scanning is enabled
2024-07-10T14:02:01-04:00 INFO Detected SBOM format
                               format="cyclonedx-json"
2024-07-10T14:02:01-04:00 WARN Third-party SBOM may lead to
                               inaccurate vulnerability detection
2024-07-10T14:02:01-04:00 WARN Recommend using Trivy to generate
                               SBOMs
2024-07-10T14:02:01-04:00 INFO Number of language-specific files
                               num=1
2024-07-10T14:02:01-04:00 INFO [python-pkg] Detecting
                               vulnerabilities...
```

```
Total: 2 (UNKNOWN: 0, LOW: 0, MEDIUM: 2, HIGH: 0, CRITICAL: 0)
==================
Library: requests
Vulnerability: CVE-2024-35195
Sev: MEDIUM
Installed Version: 2.31.0
Fixed Version: 2.32.0
Title: requests: subsequent requests to the same host ignore cert
Verification: https://avd.aquasec.com/nvd/cve-2024-35195
==================
Library: scikit-learn
Vulnerability: CVE-2024-5206
Sev: MEDIUM
Installed Version: 1.3.1
Fixed Version: 1.5.0
Title: scikit-learn: Possible sensitive data leak
Verification: https://avd.aquasec.com/nvd/cve-2024-5206
```

Conclusion/Q&A

- Based on the scan results, the modules can be updated to versions where the vulnerabilities are patched
- Scanning tools can also check for licensing issues, ensuring all components comply with project license policies.
- Languages other than Python are also vulnerable, e.g. JavaScript/Node.js (npm), Java (Maven Central), and others
- Continuous scanning can be implemented in a CI/CD pipeline to monitor for new vulnerabilities.
- Active community maintenance efforts help in promptly addressing vulnerabilities, but the risk is still non-zero.
- To mitigate risks, use latest versions, apply security patches, perform regular vulnerability scans
  - Beware of complacency: maintaining compatibility can introduce vulnerabilities

This material Is based upon work supported by the National Science Foundation under Grant No. 2300188. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.