Business & Enterprise Systems

Integrity - Service - Excellence

PEO BES
HIE DevSecOps

Software Bill of Materials (*SBOM*)

DAFITC 2022

DEVSECOPS

Agenda

- Current state of BES programs … what we know (or think we know)
- SBOM - What is it and why should we care?
- HIE DSO CI/CD Pipeline As-Is
- Demo: BOB-DB SBOM scan and results
- Contact Info

What do we know?

Some questions we should all ask about our programs and what we put into production.

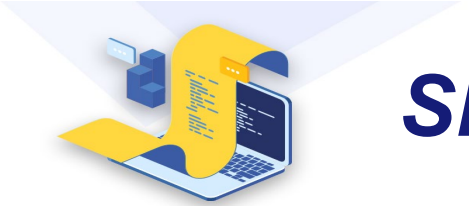
- Do we really know all the components that make up our deliverable?
  - Open Source Direct, Transitive Dependencies and Provenance
  - Third party libraries? How and what was used to create those?
  - Are your dependencies up-to-date?
- Do we really know and share the risk we are introducing?
- Not tracking your open source dependencies is making you lose visibility and control over your software.

In 2008 Gartner published a report on the state of open source software stating that
"…if you don't think you use it, then you use it; and if you think you do use it, then you use lots more of it than you know."

SBOM – What is it?

A Software Bill of Materials (SBOM) is a ***formal*** record containing the details and *supply chain* relationships of various components used in building software. These components, including libraries and modules, can be *open source or proprietary*, free or paid.

- Formal is important because the SBOM needs to be a precise software development artifact designed to be interchanged with other companies, the government and available for security audits.
- Supply chain relationships are important because we need to understand where software comes from, is it open source? Commercial? What versions?
- Lastly, SBOMs apply to more than just open source software as commercial off the shelf (COTS) software may actually be even more important because of hidden risks.



SBOM – Why is it important?

The heightened awareness of supply chain security and SBOMs are a good thing but software organizations might still be questioning their usefulness. They may agree to supply them but fail to see downstream benefits. So why should your organization insist on SBOMs from your software suppliers? Why should you start creating them for your products or the applications you are using today? Consider these benefits:

•**Identifying and avoiding vulnerabilities** in reused components in your own developed software and software purchased by your organization.

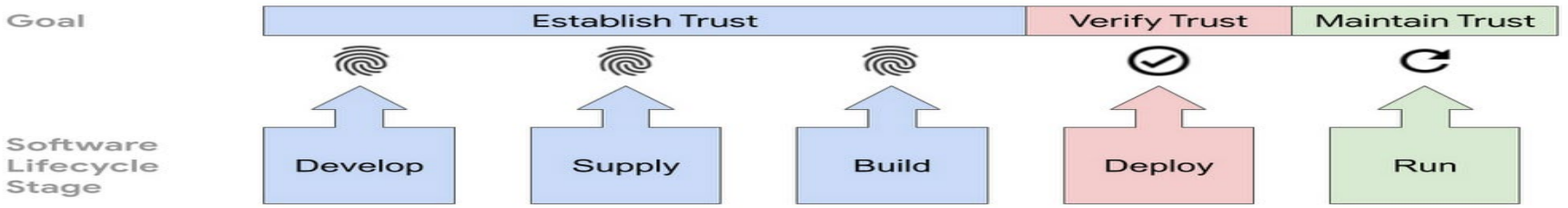
•**Managing software supply chain risk** to remove and reduce the unknown security risk in reused software. SBOMs provide data for business decisions on software purchases and open source reuse.

•**Supply chain qualification** to ensure consistency and accountability from suppliers. Suppliers than meet the SBOM requirements during procurement are given preferential treatment.

•**Improved security and downstream benefits** that come with risk management and mitigation. Avoiding and catching security risks before they become embedded in their product pays huge dividends during development and deployment of your products.

•**Common understanding of software assets** that comes with a standardized SBOM amongst software developers, suppliers and open source projects. SBOMs become a way to communicate software contents and dependencies within and outside an organization.

Secure your software development lifecycle



concepts for securing the software development lifecycle:

- Artifact - any file produced as the result of a build pipeline, such as container images, language packages, compiled binaries, etc.
- Provenance - metadata about how an artifact was built, including the build process, top-level source, and dependencies
- Digest - the result of a cryptographic hash function which produces a fixed-size value uniquely identifying an artifact, such as a SHA-256 hash of a container image
- Attestation - a cryptographically signed file recording the provenance of the build pipeline at the time a specific artifact or set of artifacts was produced
- Attestor - any system or process that produces an attestation, often included as part of a build pipeline after artifact creation and prior to deployment
- Immutable references - an identifier, such as a URL, that is guaranteed to always point to the same, immutable artifact, such as a specific container image or language package
- Build integrity - the verification of the output of a build pipeline via attestations

https://cloud.google.com/blog/products/application-development/google-introduces-slsa-framework

HIE DSO Pipeline Components

CI/CD Engine
- Jenkins

Source Control
- BES BitBucket
- AWS Code Commit
- MS TFS/ADO Repos
- Other...

Automated Build
- Maven
- Ant
- Gradle
- MSBuild
- Other...

SBOM
- CycloneDX
- Dependency track

COMING SOON

Artifact Repo
- JFrog Artifactory

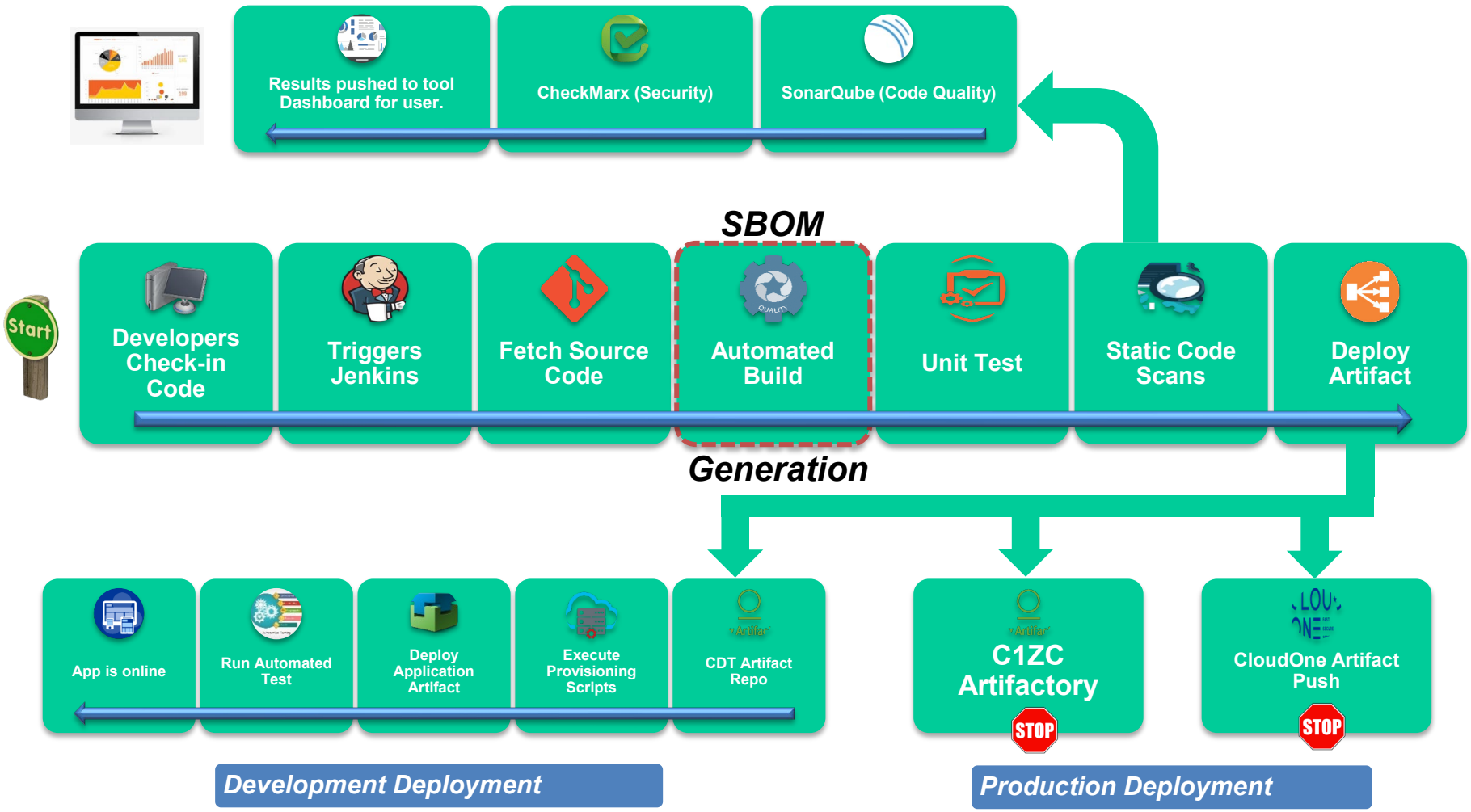
Automated Testing
- JUNit
- JMeter
- Selenium
- Microfocus ALM/UFT
- Microfocus LoadRunner

Static Code Analysis - Quality
- SonarQube/SonarLint
- CAST AIP (POC)

Static Code Analysis – Security
- Checkmarx

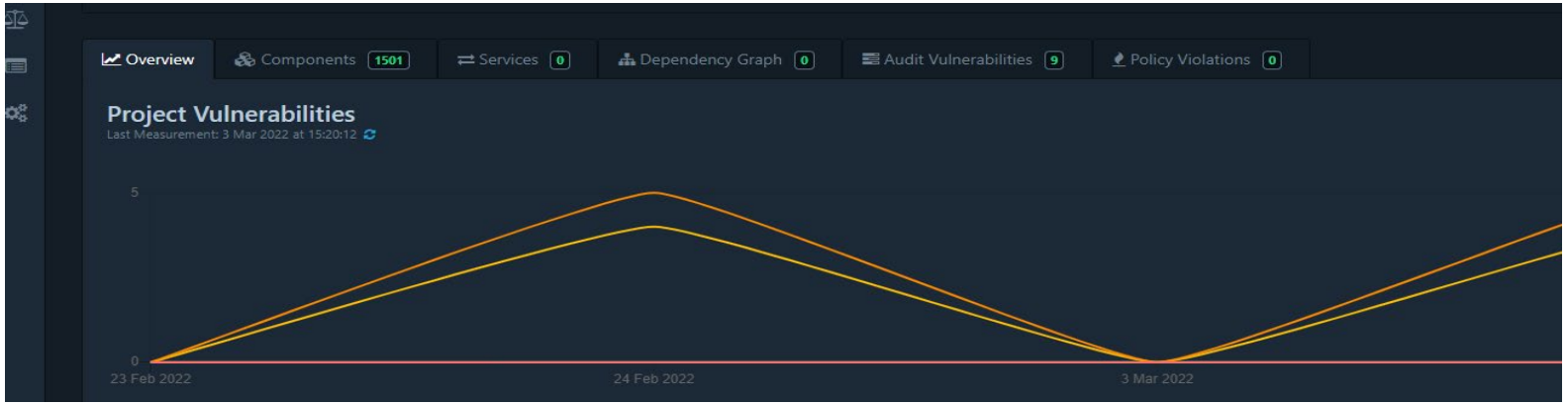
Security/Cyber Dynamic Testing
- AppDetectivePRO
- AppScan
- Burp Suite

HIE DSO Pipeline Flow



Demo: BOB-DB

- **BOB-DB – BES Operational Baseline-Database**
  - Tracks Infrastructures, Platforms, Software
- Build SBOM with CycloneDX Tools (CycloneDX Tool Center)
- Publish to Dependency Track (dependencytrack.org)
- View Results



When Do We Act?

We need to be ***proactive*** in securing our IT systems and the programs we create. We can't afford to be ***reactive*** and take action after potential vulnerabilities or risk have been exploited by hackers from across the globe.

- ✓ **President Biden's Executive Order**

Supply chain attacks are now expected to multiply by 4 compared to last year.

Contact Info

- **HIE DSO Service Desk/Get Help:**

  https://bes-atlassian.cce.af.mil/jira/servicedesk/customer/portal/9

- **HIE DSO Org Box:**

  AFLCMC.HIQA.DevSecOps@us.af.mil

- **HIE DSO Docs & How-Tos:**

  https://bes-atlassian.cce.af.mil/confluence/x/HoAg