Security Boulevard

ActiveState

3 Steps to Software Supply Chain Security Success in 2023

February 22, 1 pm ET

Loreli Cadapan
Vice President, Product
ActiveState

Mitch Ashley
CTO, Techstrong Group
Principal
Techstrong Research

POWERED BY Techstrong | Learning

ActiveState

Introductions

MITCH ASHLEY

CTO, Techstrong Group
Principal, Techstrong Research

- Leads team of preeminent experts in digital transformation, DevOps, cloud-native, and cybersecurity.
- Works with companies to align digital transformation and technology strategies to achieve disruptive goals and high impact results.
- Host of popular DevOps Chats podcast engaging with digital leaders.

ActiveState

Introductions

LORELI CADAPAN

Vice President, Product
ActiveState

- 20+ years in enterprise software at enterprises and startups, focused on DevOps and DevSecOps.
- Held different roles from coding, architecture, development management to product management.
- Currently leads Product team at ActiveState, powering the world's software development teams and accelerating their application security.

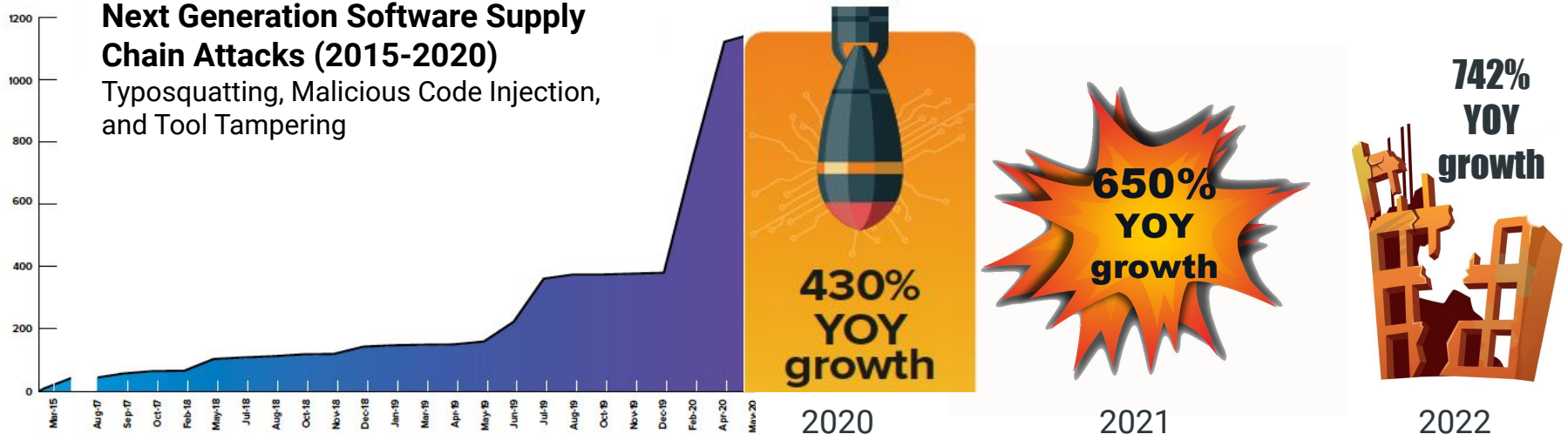
ActiveState

ActiveState's Mission

- Purpose
  - Create technology that just works: open source software that's easy and safe for Enterprises and open source communities.
- What We Deliver
  - A secure open source software supply chain for the modern enterprise

ActiveState

Why Software Supply Chain Security Matters?



Source: [Sonatype State of the Software Supply Chain]

ActiveState

The Consequences of Supply Chain Weaknesses

- Business Impact:
  - Millions in direct losses
  - Billions in cleanup costs
  - SWI stock dropped 40% in a day

- 18,000 customer affected, including:
  - 80% of the Fortune 500
  - The top 10 US telecos
  - The top 5 US accounting firms
  - The CISA, FBI & NSA
  - All 5 branches of the US military

ActiveState

The Open Source Supply Chain Challenge

- Breadth
  - Multiple ecosystems: Java, JavaScript, Python, Rust, Perl, Ruby, PHP, etc
- Depth
  - Tens of thousands of authors/maintainers of millions of packages
- Complexity
  - High rate of change: new vulnerabilities, new package versions, new authors/maintainers

ActiveState

US Government Response

MAY 12, 2021

Executive Order on Improving the Nation's Cybersecurity

BRIEFING ROOM › PRESIDENTIAL ACTIONS

EXECUTIVE OFFICE OF THE PRESIDENT
OFFICE OF MANAGEMENT AND BUDGET
WASHINGTON, D.C. 20503

THE DIRECTOR

September 14, 2022

M-22-18

MEMORANDUM FOR THE HEADS OF EXECUTIVE DEPARTMENTS AND AGENCIES

ActiveState

US Government Agency Requirements

1. Obtain **SBOMs** from their software vendors, along with documented processes in order to help validate code integrity.
2. Obtain **a software attestation** from the software producer.
3. Only use software that meets the NIST guidance for **secure software development** practices.

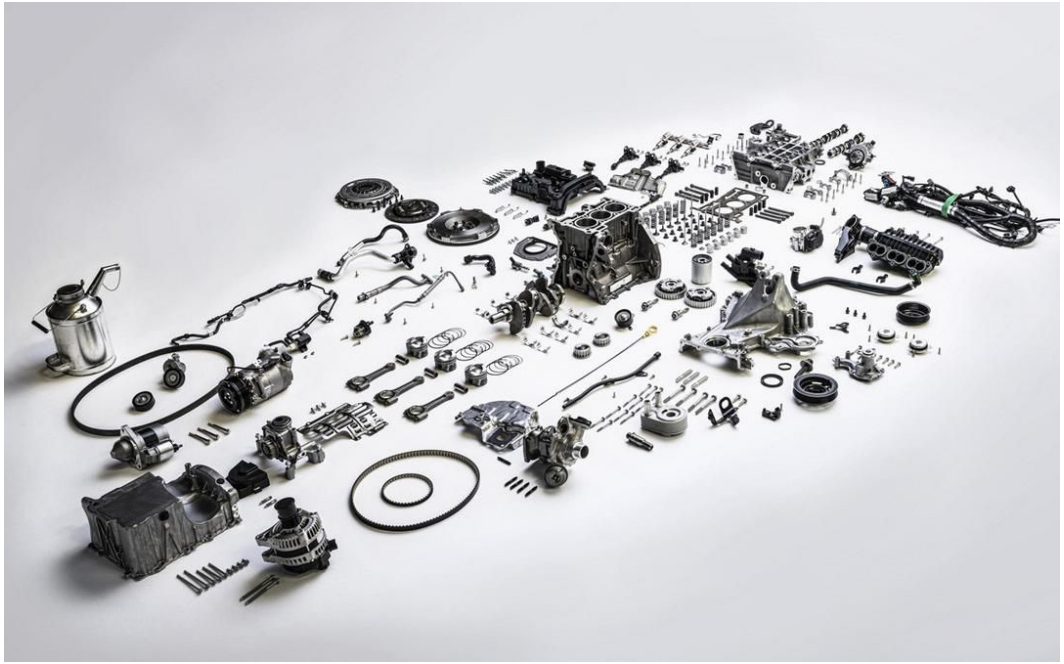
ActiveState

Deadlines

- **June 11, 2023** - Agencies shall collect attestation letters for “critical software”
  - Exec Order Memorandum: **III.A.3**
- **September 13, 2023** - Agencies shall collect attestation letters for all software
  - Exec Order Memorandum: **III.A.4**

ActiveState

Requirement #1: SBOMs



ActiveState

SBOM Benefits

- Identify components that are not allowed within a compliance framework like PCI-DSS, SOX, HIPAA, etc.
- Identify compromised components known to be targeted by cyberattacks
- Identify licenses that don’t comply with corporate guidelines
- Provide compatibility between older software packages and OSS updates by helping to identify transient dependencies that may have shifted

ActiveState

SBOM Options

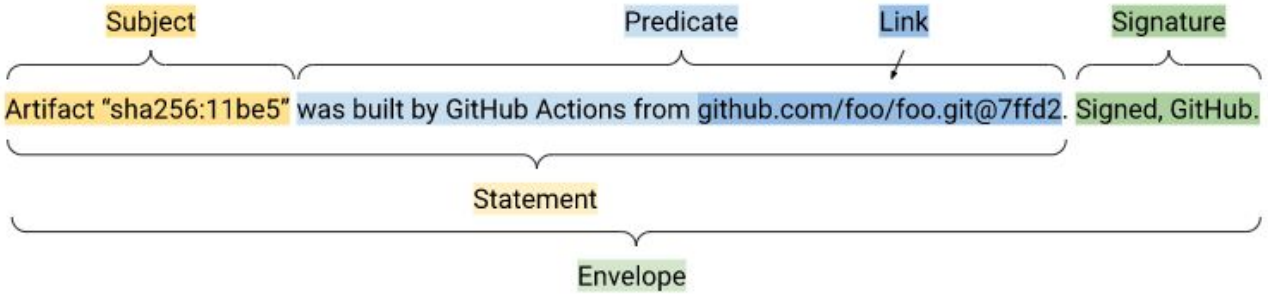
Your development platform may already be able to generate SBOMs using existing tools, such as:

- Microsoft's SPDX sbom-tool
- GitLab's CycloneDX generator
- Anchore's SBOM GitHub Action
- Linux Foundation SPDX SBOM Tool

ActiveState

Requirement #2: Software Attestations

- Allow providers to establish trust for their software with their consumers.



ActiveState

Software Attestation Benefits

- Can be used to assert (for example) that an application:
  - Was built securely
  - Is not currently compromised by malicious code
  - Was built using an approved set of dependencies
  - Contains only dependencies that were built securely from source code, etc
- Promotes trust

ActiveState

Software Attestation Options

Check if your development platform can generate a software attestation for your proprietary code using tools like:

- TestifySec Witness
- Google Cloud Build
- GitHub Actions
- GitLab Runner Attestations

ActiveState

Requirement #3: Secure Software Development

- Specifically, secure development that follows the US National Institute of Standards and Technology (NIST) guidelines:
  - Secure Software Development Framework (SSDF)
  - NIST Software Supply Chain Security Guidance

ActiveState

Pathway to Secure Development

- Prepare the organization/developers to perform secure software development.
- Protect the software from tampering and unauthorized access.
- Produce well-secured software that features minimal security vulnerabilities.
- Respond to vulnerabilities in a timely manner, and implement processes to prevent similar vulnerabilities from occurring in the future.

ActiveState

Secure Software Frameworks

Review your existing development process against:

- The Supply chain Levels for Software Artifacts (SLSA)
- The US government matrix of SSDF requirements

1	Define Security Requirements for Software Development
2	Implement Roles and Responsibilities
3	Implement Supporting Toolchains
4	Define and Use Criteria for Software Security Checks
5	Implement and Maintain Secure Environments for Software Development
6	Protect All Forms of Code from Unauthorized Access and Tampering
7	Provide a Mechanism for Verifying Software Release Integrity
8	Archive and Protect Each Software Release
9	Design Software to Meet Security Requirements and Mitigate Security Risks
10	Review the Software Design to Verify Compliance with Security Requirements and Risk Information
11	Reuse Existing, Well-Secured Software When Feasible Instead of Duplicating Functionality
12	Create Source Code by Adhering to Secure Coding Practices
13	Configure the Compilation, Interpreter, and Build Processes to Improve Executable Security
14	Review and/or Analyze Human-Readable Code to Identify Vulnerabilities and Verify Compliance with Security Requirements
15	Test Executable Code to Identify Vulnerabilities and Verify Compliance with Security Requirements
16	Configure Software to Have Secure Settings by Default
17	Identify and Confirm Vulnerabilities on an Ongoing Basis
18	Assess, Prioritize, and Remediate Vulnerabilities
19	Analyze Vulnerabilities to Identify Their Root Causes

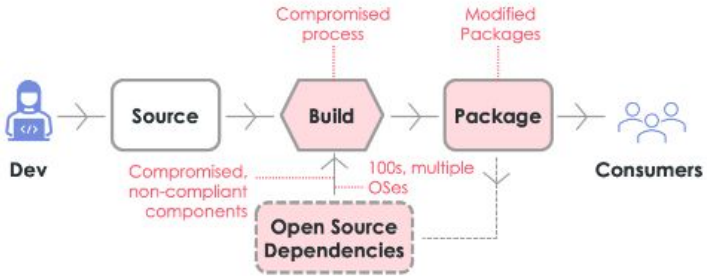
SLSA Requirements
Source - Version controlled
Source - Verified history
Source - Retained indefinitely
Source - Two-person reviewed
Build - Scripted build
Build - Build service
Build - Build as code
Build - Ephemeral environment
Build - Isolated
Build - Parameterless
Build - Hermetic
Build - Reproducible
Provenance - Available
Provenance - Authenticated
Provenance - Service generated
Provenance - Non-falsifiable
Provenance - Dependencies complete
Common - Security
Common - Access
Common - Superusers

ActiveState

How ActiveState Can Help

Challenging State of Affairs for Open Source

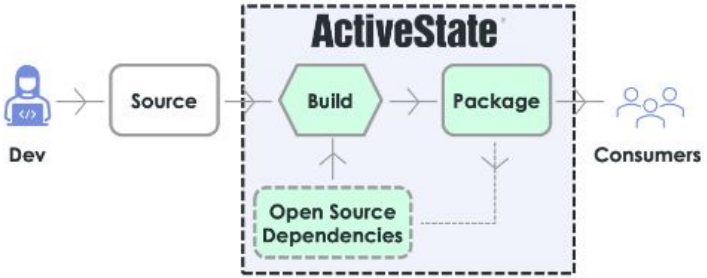
Everyone loves working with open source but hates **building, managing** and **securing** it



- Managing Open Source Components Is Complex
- Creates Licensing / Compliance Issues
- Introduces Security Vulnerabilities

ActiveState Makes Using Open Source Easy and Secure

We **shift left** to deliver **secure open source builds** so developers don't need to worry about the **drudgery** and **security** of building open source



- End-to-End Open Source Build and Package Management
- Consistent Enterprise-Wide Open Source Compliance Policy
- Security and Provenance of Open Source Components

ActiveState

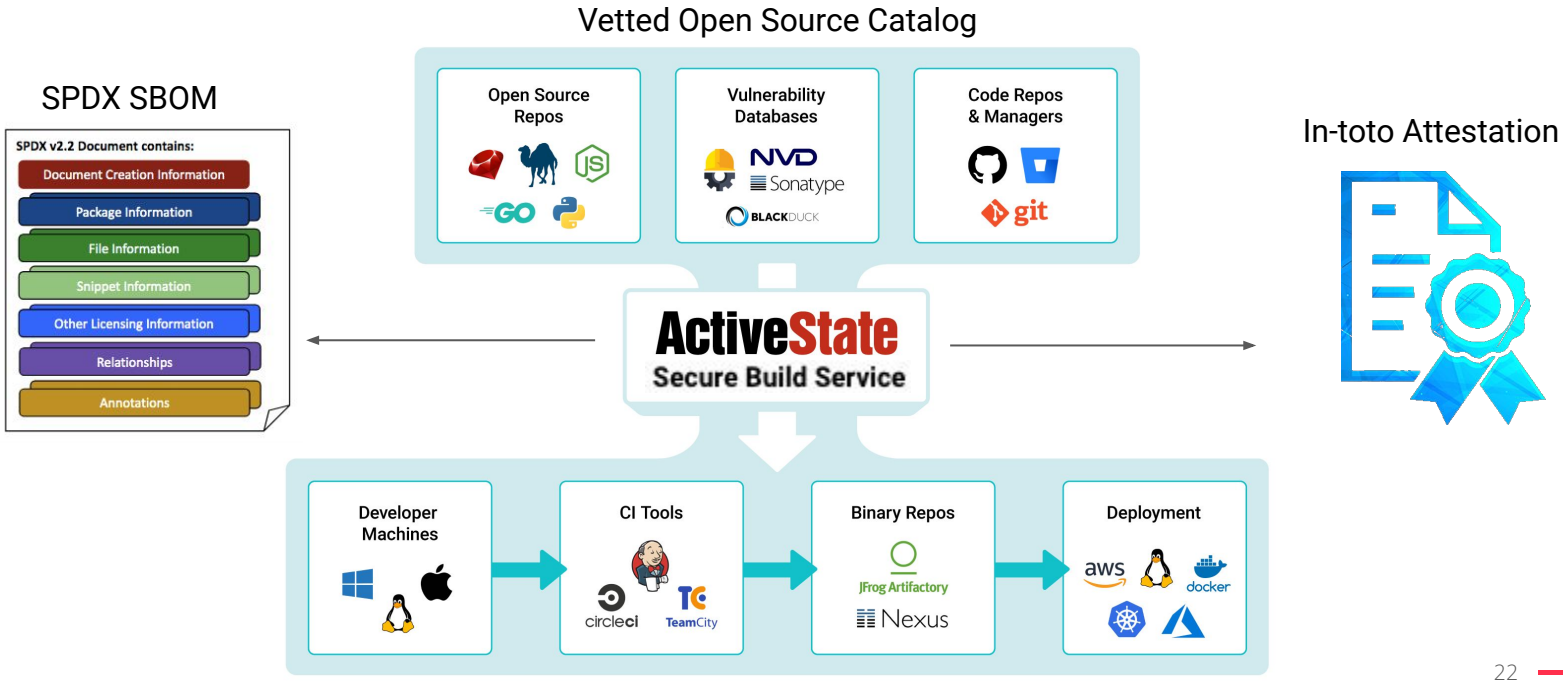
Working with Open Source is Time Consuming & Challenging

	Market Challenge	ActiveState Solution
 **Developers**	• *Open Source not always available in binary form needed* • *Difficult to reproduce environments* • *Fear updating because of unforeseen consequences*	• *Convert Open Source to artifacts you trust with no effort* • *Create and deploy reproducible environments* • *Switch between environments at will*
 **DevOps/IT**	• *Difficulty managing the necessary tools and languages* • *Expensive to maintain secure build infrastructure* • *Long-term support of applications and environments*	• *Universal tooling across operating and language ecosystems* • *A cloud native Open Source supply chain out-of-the-box* • *On demand language and operating experts*
 **Security**	• *What Open Source is being used, by who, where it's from* • *Need teams to stay current and update quickly* • *No single source of origin for each Open Source binary*	• *Cross project dependency management* • *Seamless, efficient point and click dependency updates* • *Auditable trail of Open Source ingestion to deployment*

Enterprises Buy ActiveState To Enable Their Developers, Engineers And Security Teams To Operationalize Open Source In A Single, Unified Platform

ActiveState

A Single SaaS Solution



ActiveState

Requirement #1: ActiveState SBOM

- Programmatically created in SPDX format via the ActiveState Platform GraphQL API
- Mitigates risk by:
  - Keeping track of transitory dependencies and related native libraries. (turtles all the way down)
  - Identifying and examining the relationships between the dependencies. (ancestors and descendants)
  - Showing changes to the dependency tree BEFORE you commit to any update.

ActiveState

Requirement #2: ActiveState Software Attestation

- DevOps systems (GitHub Actions, Azure DevOps, etc) provide attestations for **proprietary** software.
- ActiveState automatically builds all **open source binaries** from source code, and provides an attestation for each.

ActiveState

Requirement #3: SLSA-Compliant Secure Build Service

- Supply chain Levels for Software Artifacts (SLSA) is a security framework designed to help organizations improve the security & integrity of their code

Requirement	SLSA 1	SLSA 2	SLSA 3	SLSA 4
Build - Scripted build	✓	✓	✓	✓
Build - Build service		✓	✓	✓
Build - Build as code			✓	✓
Build - Ephemeral environment			✓	✓
Build - Isolated			✓	✓
Build - Parameterless				✓
Build - Hermetic				✓
Build - Reproducible				✓

ActiveState

ActiveState Platform Demo

ActiveState

Q&A

ActiveState

Next Steps

Schedule a demo with our product experts:
https://www.activestate.com/get-demo/

Learn more about ActiveState's Secure Supply Chain Solutions:
https://www.activestate.com/solutions/enterprise-security/

Try the ActiveState Platform for free:
https://platform.activestate.com/