ActiveState

Navigating Your Software Supply Chain Journey

5 Stages to Success

Introductions

Nicole Schwartz

Security Product Manager
ActiveState

Dana Crane

Product Marketing Mgr
ActiveState

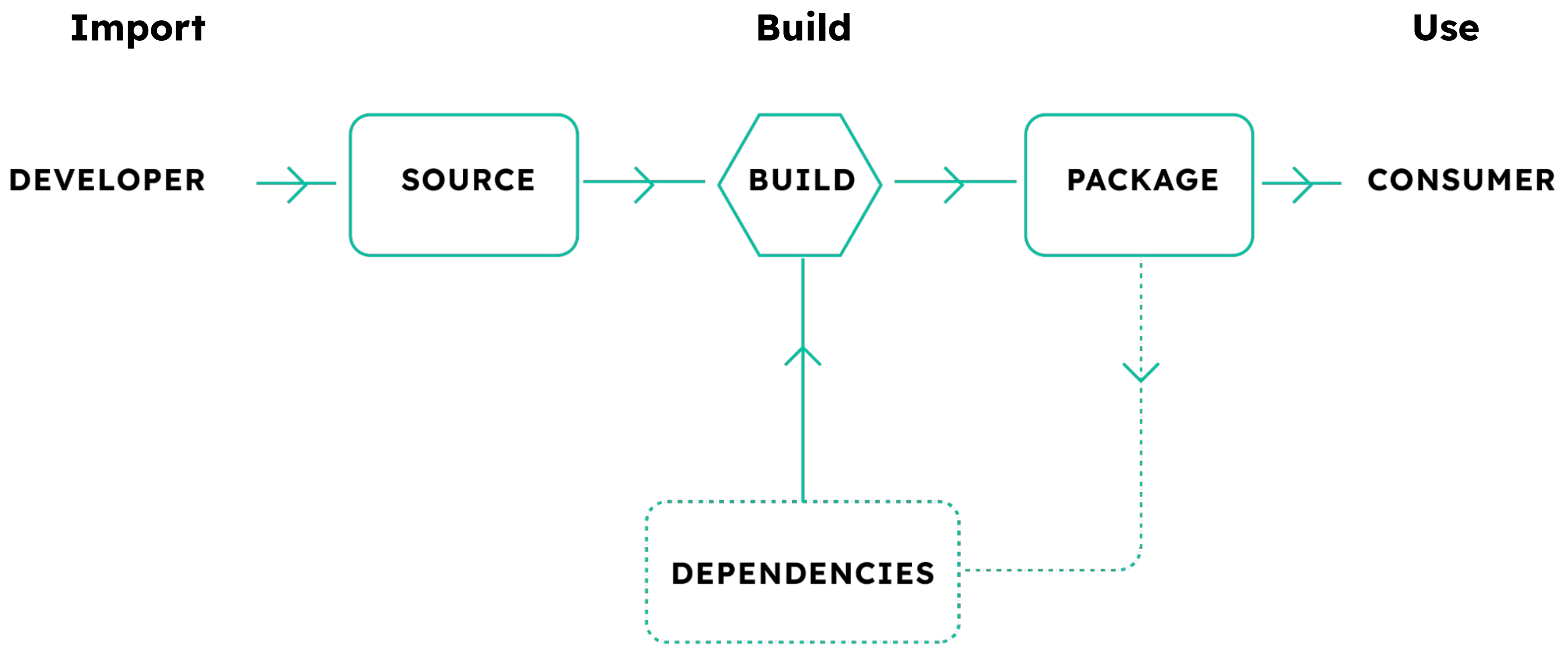
Housekeeping

- We will host polls throughout the webinar
- We will be emailing everyone the slides after the webinar
- Submit your questions in the Q&A tab and we will answer at the end
- Use the Chat to pitch in any comments or have technical issues

ActiveState

Threats in the Software Supply Chain

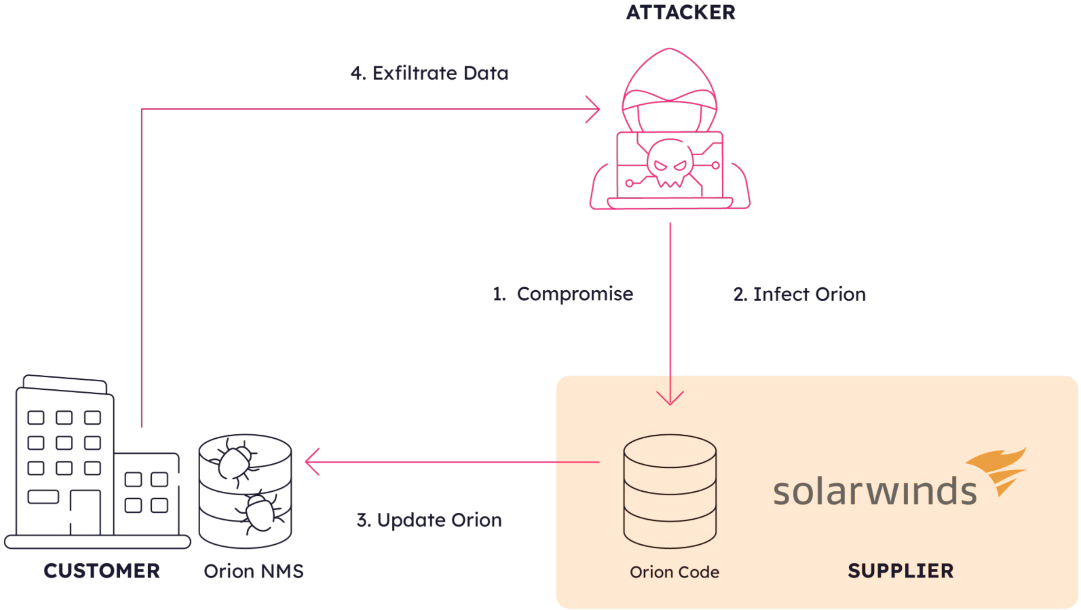
What is the Software Supply Chain?



ActiveState

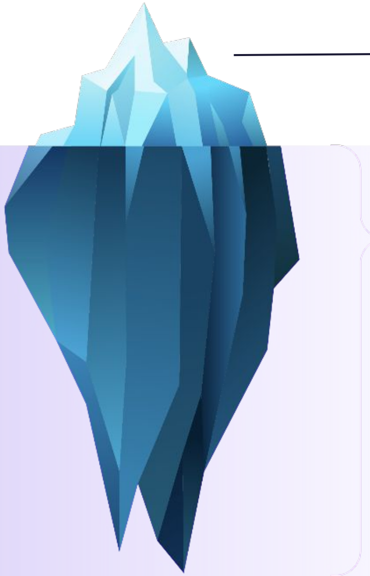
Solarwinds – December 2020

The real value of signed code is the establishment of trust, which is why this hack was so pernicious: it effectively undermined trust in signed software.

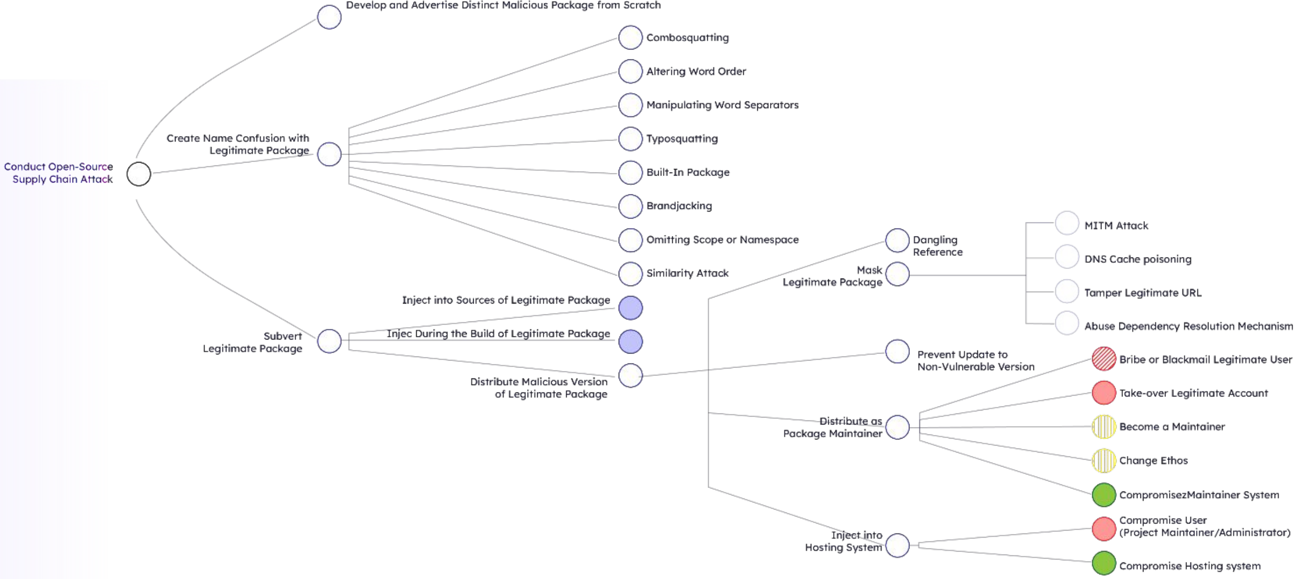


ActiveState

Vulnerabilities vs Supply Chain Threats



Vulnerabilities

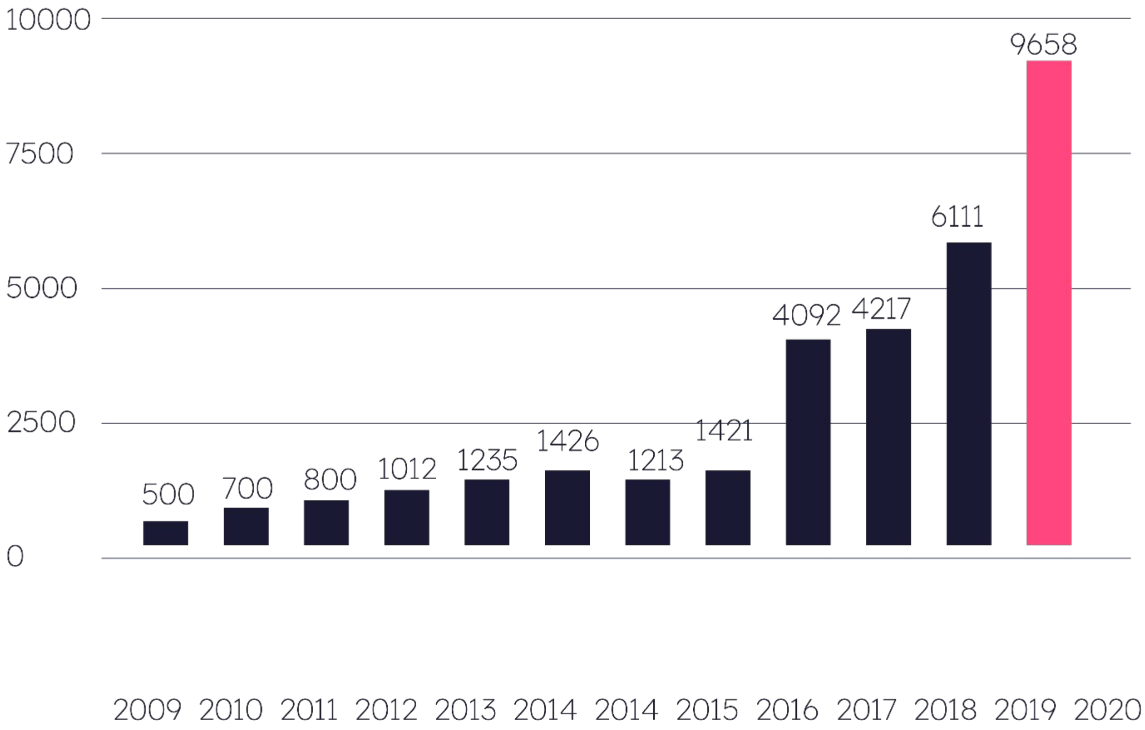


- Conduct Open-Source Supply Chain Attack
  - Develop and Advertise Distinct Malicious Package from Scratch
  - Create Name Confusion with Legitimate Package
    - Combosquatting
    - Altering Word Order
    - Manipulating Word Separators
    - Typosquatting
    - Built-In Package
    - Brandjacking
    - Omitting Scope or Namespace
    - Similarity Attack
  - Subvert Legitimate Package
    - Inject into Sources of Legitimate Package
    - Injec During the Build of Legitimate Package
    - Distribute Malicious Version of Legitimate Package
      - Mask Legitimate Package
        - Dangling Reference
        - MITM Attack
        - DNS Cache poisoning
        - Tamper Legitimate URL
        - Abuse Dependency Resolution Mechanism
      - Prevent Update to Non-Vulnerable Version
      - Distribute as Package Maintainer
        - Bribe or Blackmail Legitimate User
        - Take-over Legitimate Account
        - Become a Maintainer
        - Change Ethos
        - CompromisezMaintainer System
      - Inject into Hosting System
        - Compromise User (Project Maintainer/Administrator)
        - Compromise Hosting system

Source: SAP Supply Chain Risk Explorer

ActiveState

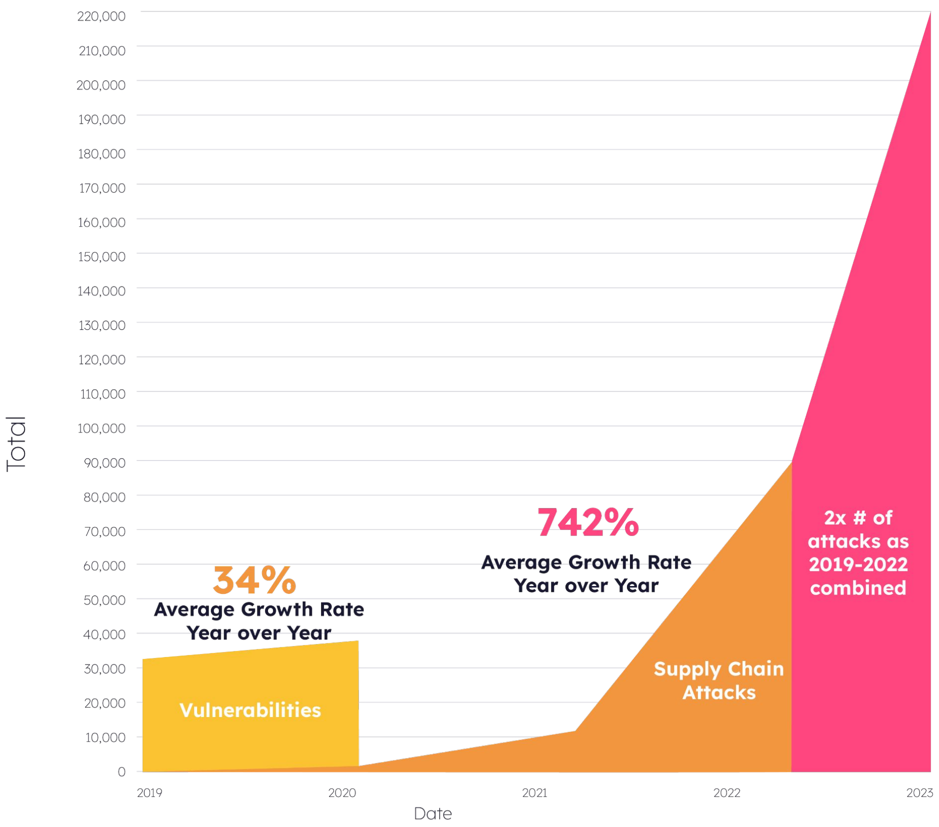
Vulnerabilities are Growing



Source: mend.io

ActiveState

Supply Chain Attacks vs Vulnerabilities



Regulations

- The **US** Government (EO 14028 & M-22-18)
- The **US** Securities and Exchange Commission (SEC) Cybersecurity Disclosure Rules
- **EU** Cyber Resilience Act [proposed]
- **Canada** – Bill C-26 (telecom only)
- **UK** – Network and Information Systems Regulations last updated in 2020 pre major attacks
- **Germany** – The Law on the Federal Office for Information Security (BSIG) aligns closely with EU
- **ASEAN** – ten member countries target of 2025 for a set of cybersecurity regulations
- **Japan** – Draft Law Concerning Promotion of Ensuring Security through Integrated Economic Measures

ActiveState

5 Stages to a Secure Software Supply Chain

Stage 0
Complete Anarchy

- Ignorant of best practices like the NIST Secure Software Development Framework (SSDF)
- Everyone uses their own development tools
- No standardized processes
- No governance

ActiveState

Stage 1
Observable Chaos

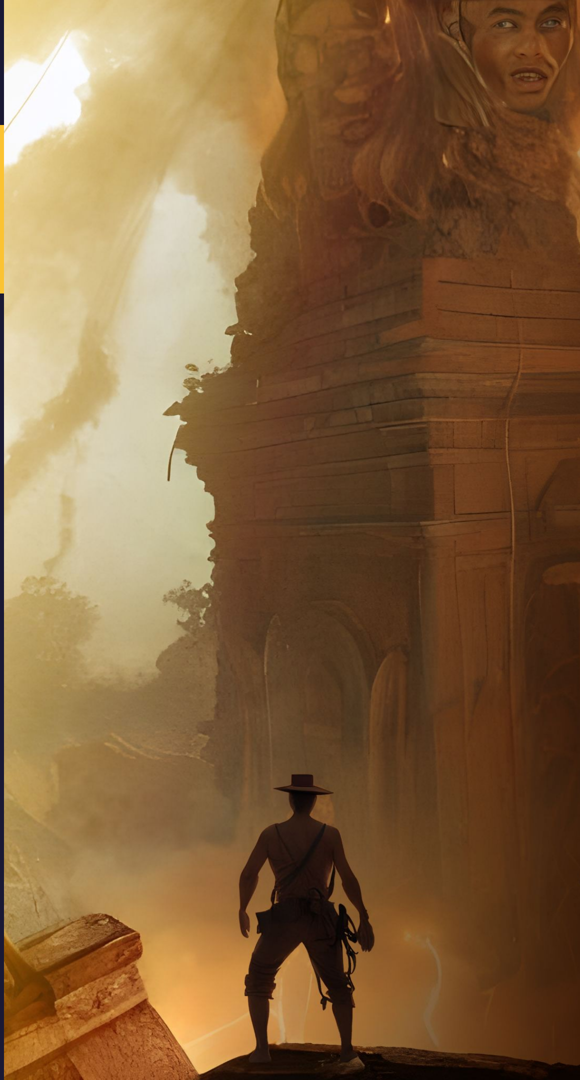
There is awareness of issues and a desire to gain visibility into the problems

Standardized Tooling:

- SCA tools such as Snyk, Sonatype Nexus, Synopsys Black Duck, etc
- SBOMs solutions supporting CycloneDX and/or SPDX standards

No / minimal best processes

No / minimal governance



Poll

What tools do you currently have in place?

Check all that apply:

■ Software Composition Analysis (SCA) / Dependency Scanners

■ Software Bill of Materials (SBOM)

■ Static Application Security Testing (SAST)

■ Others, such as GUAC

ActiveState

Stage 2
Automated Security

- Standardized tooling supports observability
  - eg., Software Attestations
- Standardized “import & build” best practices
  - Supply-chain Levels for Software Artifacts or SLSA ("salsa")
- No / minimal governance

Poll

Does your import and/or build routine use:

Check all that apply:

- ■ Best practices such as 2fa/mfa, 2 person code review, etc
- ■ Emerging practices like Attestations
- ■ Hardened CI/CD pipelines
- ■ Secure development best practices like SSDF
- ■ Emerging frameworks like SLSA

ActiveState

Stage 3
Verifiable Safety

- Standardized Tooling supports observability
  - Intrusion Detection System (IDS): Barracuda, Check Point
  - Endpoint Detection Systems (EDS) / (EDR): Trellix, CrowdStrike, Sentinel One
  - Exposed system port monitoring: Shodan
- Standardized best practices followed
- Governance in place (company / regulatory policy)
  - i.e. Open Policy Agent (OPA) by styra, minder by stacklok

Poll

How are your best practices enforced?

Check all that apply:

- ■ Automate vulnerability and/or package license checks
- ■ Automate SBOMs/Attestations
- ■ Build dependencies from source
- ■ Create reproducible builds
- ■ No, we trust that best practices are followed, but don’t verify

ActiveState

Stage 4
Anti Entropy

- Your team is proactive about security
- Standardized Tooling supports observability
  - Threat Modeling: Microsoft Threat Modeling Tool, Cairis, OWASP Threat Dragon
  - Intrusion Prevention System (IPS): Snort, Fortinet
  - Pentesting
- Standardized best practices followed
- Governance: security is everyone’s priority

ActiveState

Poll

What stage is your organization at today?

Choose one:

- ■ Stage 0 - Complete Anarchy
- ■ Stage 1 - Observable Chaos
- ■ Stage 2 - Automated Security
- ■ Stage 3 - Verifiable Safety
- ■ Stage 4 - Anti Entropy

The Current State of the Industry

40%

SBOMs
in place

47%

Implementing
SLSA

48%

Implementing
SSDF

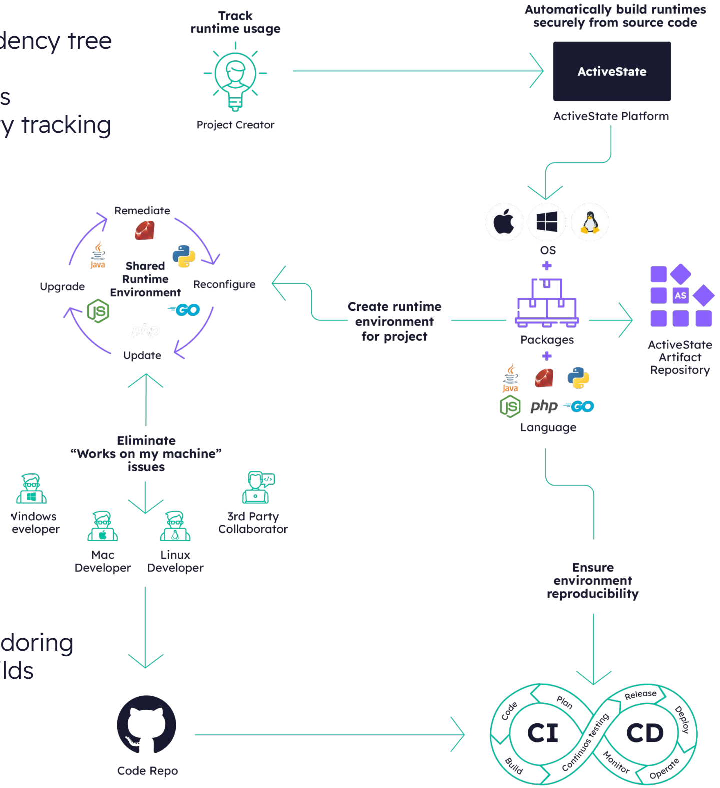
ActiveState

Where does ActiveState fit in?

ActiveState Platform

Stage 1: **Observability**

- Full dependency tree
- SBOMs
- Attestations
- Vulnerability tracking



Track runtime usage

Project Creator

Automatically build runtimes securely from source code

ActiveState

ActiveState Platform

OS

+

Packages

+

Language

ActiveState Artifact Repository

Create runtime environment for project

Remediate

Reconfigure

Update

Upgrade

Shared Runtime Environment

Eliminate “Works on my machine” issues

vindows eveloper

Mac Developer

Linux Developer

3rd Party Collaborator

Code Repo

Ensure environment reproducibility

CI

CD

Code

Plan

Build

Continuous testing

Release

Deploy

Monitor

Operate

Stage 2: **Automated Security**

- SLSA build level 3

Stage 3: **Verifiable Safety**

- Dependency vendoring
- Reproducible builds

Stage 4: **Anti Entropy**

- Built-in security

ActiveState

ActiveState Platform Demo

Q&A

Next Steps

Schedule a discussion to get an assessment:

https://www.activestate.com/solutions/contact-sales/

Take our Supply Chain Security Survey & find out how you rate:

https://www.surveymonkey.com/r/BNGZPH6

Try the ActiveState Platform for free:

https://platform.activestate.com/