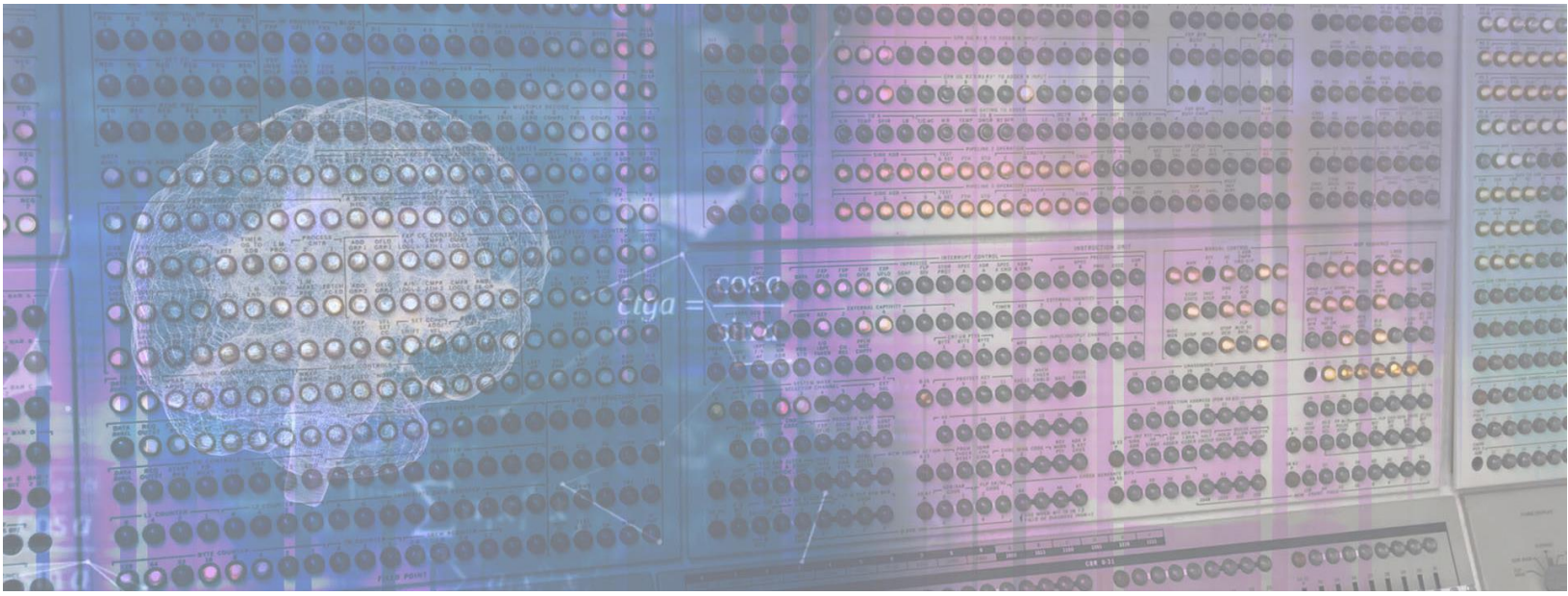


How to Bulletproof Your Open Source and Proprietary Software Supply Chain

An Intellyx Whitepaper for CIQ
by Jason Bloomberg, Managing Partner
August 2023

Modern software, both proprietary and open-source, depends upon open-source components. Those components typically depend upon other open-source elements, leading to complex dependencies of interconnected open-source components, libraries, and packages.

These software supply chains offer bad actors numerous vulnerabilities ripe for compromise. If they can compromise an upstream component, then any downstream software product that includes that component is automatically compromised.

Securing software supply chains, therefore, is a top priority for enterprises and government agencies around the world. Given the inherently dynamic nature of software as well as the constantly evolving threat landscape, however, securing such supply chains is exceptionally difficult.

CIQ has tackled this problem with Mountain, its hybrid data center management and security platform. Mountain provides a repository of secure open-source components across all software dependency chains, all the way down to the operating system components that power the Linux ecosystem.

With Mountain, enterprises can build, manage, and secure the supply chains for any software product that leverages open-source components.

Understanding the Software Supply Chain Challenge

Nobody writes software from scratch anymore. Up to 99% of the code running in any enterprise consists of third-party open-source components and libraries.

Furthermore, those third-party components and libraries themselves consist of code written elsewhere, and so on. It's turtles all the way down.

We call this opaque, intertwined series of dependencies the *software supply chain*. Every link in this chain – or turtle in the pile – presents opportunities for bad actors. Compromise a single link, and every downstream component automatically includes that vulnerability.

Every link in the software supply chain presents opportunities for bad actors. Compromise a single link, and every downstream component automatically includes that vulnerability.

In other words, software supply chains are disasters waiting to happen.

From the hackers' perspective, the more complexity a software supply chain has, the better – and complex they are. Such supply chains not only consist of code, but also configurations, binaries (both proprietary and open-source), libraries, plugins, and dependencies on containers.

The tools that developers use to create and assemble the various software components in the supply chain can also be points of compromise, including

compilers, assemblers, repositories, code analyzers, and various ops tools for security, monitoring, and observability.

People are also a critical part of any supply chain – and an obvious point of compromise. Bad actors target individuals, organizations, and the various processes those organizations use to create, test, deploy, and manage the software they are building.

The SolarWinds Breach Ups the Ante

With all these potential vulnerabilities, it's no wonder modern software supply chains are under attack. The best known of these breaches targeted SolarWinds, an IT management vendor which is ironically "setting the new standard in secure software development," according to its web site.

In December 2020, researchers discovered a backdoor in SolarWinds' Orion software that bad actors had inserted by compromising SolarWinds' build server. None the wiser, SolarWinds distributed the compromised software to its customers – a long list of enterprises, including several agencies of the US Federal Government.

The SolarWinds software supply chain attack convinced the Biden Administration to institute new cybersecurity guidelines. In an executive order, the government called for a range of cybersecurity improvements, including better collaboration between the government and the private sector.

The order also called upon NIST to establish best practices and guidelines that any software supplier would need to comply with in order to sell to the Federal Government.

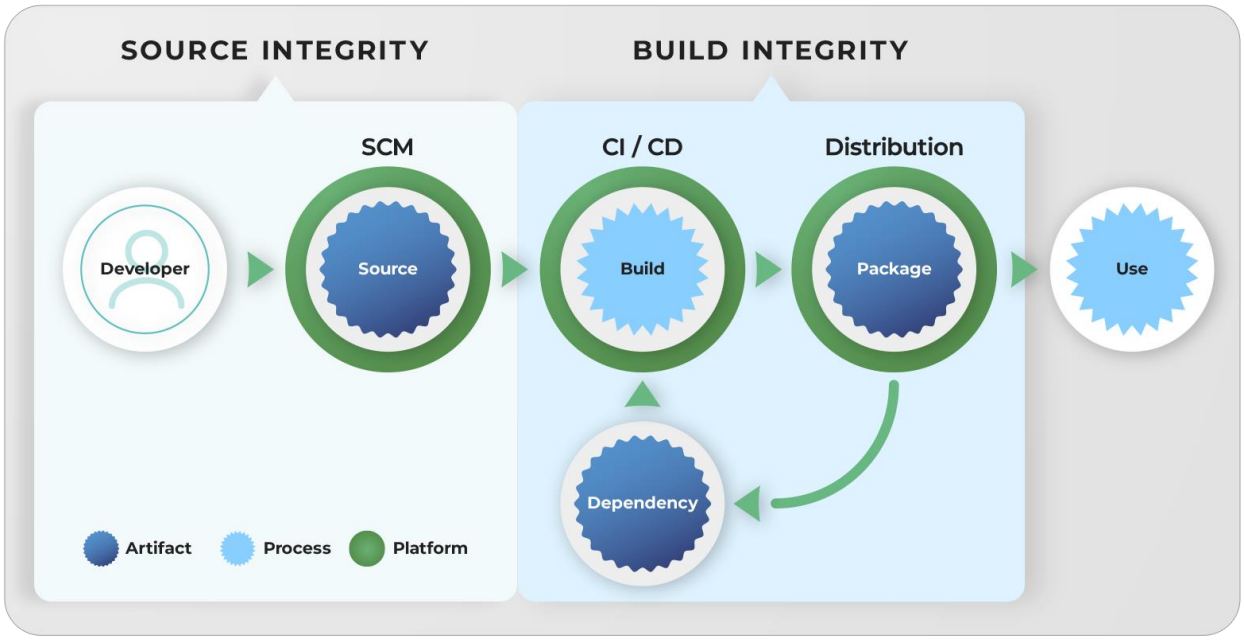
This executive order has been driving change across the entire business landscape, prompting both software vendors and enterprises to get their software supply chain acts together.

No longer can they look the other way, only to be caught by surprise by some vulnerability buried deep in their software supply chains. The time to get a handle on this problem is now.

Why Software Supply Chains Are So Vulnerable

The reasons why software supply chains are so vulnerable is because there are so many points of compromise, and furthermore, most of those points of compromise fall outside of the organization building the software.

Securing software supply chains divides generally into two areas: source integrity and build integrity, as the figure below illustrates.



The Software Supply Chain

The first point of compromise is the developers themselves. If they submit vulnerable code, then the entire supply chain becomes vulnerable.

Also impacting source integrity: compromising the source control mechanism (typically Git). Git repo attacks fall into this category, allowing for the modification of code as it moves to the build platform.

Build integrity centers on CI/CD tools and practices as well as package distribution. Attackers may compromise the build platform, bypass CI/CD processes, or compromise the package repository.

Build integrity centers on CI/CD tools and practices as well as package distribution. Attackers may compromise the build platform, bypass CI/CD processes, or compromise the package repository.

The feedback loop in the diagram above represents the dependencies in the supply chain itself. The build process would typically incorporate a third-party software package to create a new package.

Attackers can compromise this use of dependencies, often by simply publishing malware with the same name as a well-known package.

In many cases, a software package traverses this dependency loop several times, as one package includes others that in turn include additional third-party code.

The dependencies in such cases are transitive, in that any package will include dependencies several layers down, as each link in the chain transfers its dependencies to any package that includes it.

While engineering teams typically have visibility into the packages they are incorporating into the software, such visibility generally doesn't extend to the

underlying dependencies. This lack of visibility is at the heart of why software supply chains are so vulnerable.

Patches: Part of the Solution but Also Part of the Problem

Most commercial and enterprise applications include dozens or even hundreds of software components and libraries in these dependency chains – and every one of them is subject to compromise.

Zero-day attacks crop up all the time. Even in the absence of a planned attack, security personnel and researchers uncover common vulnerabilities and exposures (CVEs) for such components on an ongoing basis – and every CVE represents a possible point of compromise subject to attack.

The primary approach to addressing zero-days and other CVEs is for the developer of the component in question to issue a patch.

The patching process itself becomes a point of compromise. Is the patch effective? Is there a vulnerability in the patching process? How can an organization ensure that it has applied all the necessary patches properly?

The patching process itself becomes a point of compromise. Is the patch effective? Is there a vulnerability in the patching process? How can an organization ensure that it has applied all the necessary patches properly?

For these reasons as well as many others, securing software supply chains is far more difficult than people realize.

Why Solving the Software Supply Chain Problem Is So Difficult

Mitigating software supply chain vulnerabilities involves several human and process-centric activities. It's essential for engineering leadership to build a qualified and trustworthy team. That team must create threat models of the software under development to define and implement security test plans.

The team must also define release criteria and then test the software under production against those criteria. It's also essential to establish product support and vulnerability mitigation policies and procedures for all software under development.

In many cases, engineering leadership must arrange for the appropriate training for engineering staff to get them up to speed on these vulnerability mitigation strategies. In addition, leadership must arrange for the documentation and publication of security procedures and processes for each software release of each software product.

All these processes and procedures, while necessary, are onerous, time-consuming, and prone to errors, shortcuts, and omissions. Furthermore, adversaries are aware of these shortcomings and are all too eager to take advantage of them.

Hardening Software Supply Chains Falls Short

To address these process concerns, many organizations seek to harden their software supply chains by limiting the number of third-party components they

include in their software products and locking down those components they do incorporate by rigorously testing and approving specific versions of them.

This hardening approach, however, is often counterproductive, as it doesn't take into account the dynamic nature of open-source software supply chains.

In reality, threats continue to appear and evolve on an ongoing basis. In response, maintainers of open-source components are in a constant process of detecting, managing, and patching newly discovered vulnerabilities.

Hardening a software supply chain can actually limit the ability of third-party maintainers to properly patch the components an organization deploys in production.

In sum, traditional process-based approaches to addressing software supply chain weaknesses fall short, while hardening them is often counterproductive. Organizations require a more effective approach.

CIQ Mountain Transforms Software Supply Chain Management

At the heart of modern software supply chain management is the Software Bill of Materials (SBOM), which provides all the necessary details about the components of any software product, including open-source dependencies, containers, build tools, and any other piece of the software development puzzle.

To be an effective tool, SBOMs must be *complete* and *accurate* – even though the software supply chain is inherently dynamic. As a result, organizations must use automation to maintain their SBOMs.

CIQ provides the necessary level of automation, starting with a list of all known issues with all open-source components available on the market. CIQ describes this list as *errata*.

Mountain, CIQ's software supply chain management product, includes these errata in a global repository that it delivers via an as-a-service model.

In addition to the errata, Mountain includes a catalog of open-source assets, including operating system images, Singularity image file (SIF) and Open Container Initiative (OCI) containers, custom Linux kernels, and curated software packages.

CIQ continually maintains the assets in the Mountain repository, along with all associated errata and instructions for their use.

As a result, any users of Mountain can ensure that their software products have properly managed supply chains by ensuring that each component conforms to the specs in the Mountain repository.

Users of Mountain can ensure that their software products have properly managed supply chains by ensuring that each component conforms to the specs in the Mountain repository.

Ascender, CIQ's product that it based on the open-source Ansible AWX, governs automation of Rocky Linux workloads and other infrastructure at an enterprise level.

With Ascender, a single administrator can apply complex patches and security hardening standards to a fleet of servers or other devices with the click of a button or an API call.

Using Ascender allows teams to reduce time spent on monotonous and repetitive tasks, simplify the execution of complex tasks, and create audit trails to track changes.

Ascender also allows teams to create automations separately but makes bringing together disparate automations easier for more complex tasks. All these features reduce costs and increase productivity for the enterprise.

CIQ: Champions of Rocky Linux

Mountain contains all relevant information and software components for the entire Linux ecosystem, all the way down to the kernels and other low-level Linux components. In fact, CIQ is an expert in Linux, as it provides best-in-class escalation support for Rocky Linux, the popular successor to CentOS.

Not only does Rocky Linux have full API and application binary interface (ABI) compatibility with Red Hat Enterprise Linux (RHEL), thus making it compatible with most flavors of Linux in production today, but it also benefits from CIQ's support for the OS, and the company's deep understanding of the inner workings of Linux.

Mountain's catalog and associated repository of standard operating systems and curated open-source packages gives its customers a way to automatically manage their software supply chains following every dependency chain down to the lowest level operating system components.

Mountain, however, is more than a software supply chain management tool. It is a complete hybrid data center management and security platform that helps organizations secure, manage, and optimize their Enterprise Linux deployments.

The Intellyx Take

Before Mountain arrived on the scene, software supply chain management essentially consisted of manual processes that were necessarily expensive, time-consuming, and error-prone. Disasters like the SolarWinds breach were the result.

With Mountain, CIQ brings automation to software supply chain management. In turn, this automation brings trust, convenience, and control to its customers across the software supply chain landscape.

The open-source world is a bazaar of all sorts of software components and libraries. Some of these components are safely maintained, but others are not.

Given most open-source components themselves depend upon other components from the bazaar, the chance that any product that includes such components will have points of compromise.

CIQ brings order and trust to the bazaar – without limiting its value. Organizations can fetch any open-source component they like from the Mountain repository and feel confident that it is as free of vulnerabilities as is possible.

This repository, however, is never static. New vulnerabilities crop up all the time – and CIQ tracks them in its list of errata. As soon as a fix is available, CIQ includes it in Mountain.

The result: any software supply chain that takes advantage of Mountain is as free from vulnerabilities as possible – and CIQ provides the visibility that organizations need to be confident in the security of their supply chains.

After all, it's not good enough for software supply chains to be secure. Organizations must be able to prove they're secure. Mountain makes such proofs possible.

About the Author

Jason Bloomberg is the founder and managing partner of enterprise IT industry analysis firm Intellyx. He is a leading IT industry analyst, author, keynote speaker, and globally recognized expert on multiple disruptive trends in enterprise technology and digital transformation.

He is #13 on the *Top 50 Global Thought Leaders on Cloud Computing 2023* and #10 on the *Top 50 Global Thought Leaders on Mobility 2023*, both by Thinkers 360. He is a leading social amplifier in Onalytica's *Who's Who in Cloud?* for 2022 and a *Top 50 Agile Leaders of 2022* by Team leadersHum.

Mr. Bloomberg is the author or coauthor of five books, including *Low-Code for Dummies*, published in October 2019.

About CIQ

CIQ partners are empowered to utilize trusted open source tools to solve their customers' challenges with confidence, knowing the history of Linux and HPC expertise at CIQ. Additionally, CIQ partners are encouraged to disrupt the status quo of traditional enterprise Linux licenses. Our industry-changing "by the person" support model allows CIQ partners and customers to have current and scaling infrastructure conversations without having to stop and count licenses for machines. CIQ lowers the cost of entry for production-ready, stable, secure and automated full stack deployments, and we want our partners to be right by our side with every engagement.