ActiveState

SOFTWARE SUPPLY CHAIN SECURITY FOR FINSERV

Executive Summary

For the Financial Services (FinServ) industry, the software supply chain is a critical security risk. Unfortunately, most banking, trading, and insurance organizations continue to approach the problem in a fragmented and reactive way, leaving significant gaps that expose them to cyberattack.

FinServ organizations too often treat software supply chain security as a subset of cybersecurity by implementing siloed tools that generate alerts which need to be investigated. This kind of reactive approach increases the AppSec team's workload, exacerbating cybersecurity burnout, and slowing down response times.

Without a comprehensive, collaborative, and proactive approach, organizations face multiple potential attack vectors, including:

- **Development Environments,** which can be compromised by the introduction of malware and vulnerabilities.
- **Build Environments,** which can be jeopardized by the injection of compromised components.
- **Software Development Process,** where using built artifacts that are not signed and/or verified using a mechanism like Provenance Attestations raises the risk of compromise.

FinServ organizations can take a proactive, collaborative approach to securing their software supply chain by implementing an end-to-end, automated platform that can ensure the security and integrity of open source components imported, built, and consumed within the enterprise.

Introduction

The Financial Services (FinServ) industry, including banking, trading, and insurance have been leaders in digital transformation, including the adoption of emerging trends like Generative AI (GenAI) applications, which in the form of chatbots are seen as the key to increasing productivity and personalization.

But the implementation of new technologies without first securing the software supply chain that these kinds of applications rely on exposes FinServ to increased risk. And the cost of addressing these risks is only going to grow.

> "The cost of software supply chain attacks will grow from $46B in 2023 to $138B by 2031 – Cybersecurity Ventures"

A balance needs to be struck between the imperative of securing the software supply chain of new and business-critical applications and the likelihood of being exposed to security risks given the ongoing AI arms race between cybercriminals, who are leveraging AI in novel ways, and FinServ organizations who are trying to close emerging attack vectors as fast as possible.

In a sign that FinServ is soon to be hit with a perfect storm of increasing risk, Statista is estimating that worldwide losses associated with cybercrimes are expected to top $10T for the first time in 2024. Cybersecurity is key, but often too reactive to effectively address the software supply chain, which is both wide and deep, and affects multiple stakeholders.

For example, the software supply chain typically includes multiple open source ecosystems (Java, JavaScript, Python, etc) and spans a number of key processes, including:

- **Import –** the process of making available open source packages, code snippets, tools, or other third-party software brought into the organization in order to streamline the software development process.
- **Build –** the process of compiling, building, and/or packaging code, usually via an automated system that also executes tests on built artifacts.
- **Consume –** the process of working with, testing, and running built artifacts in dev, test, and production.

While these processes generally focus on software development, they affect more than just coders. Security personnel, compliance officers, IT workers, and vendor risk managers all have a role to play, from making sure that third-party code is "fit for use" to ensuring the organization is complying with corporate standards and industry regulations.

A collaborative platform that can provide all stakeholders with insight into the entire software supply chain is the key to addressing cybersecurity risk in a comprehensive manner. And if that platform can also automate many of the import-build-consume supply chain security tasks, FinServ organizations will be able to take a proactive approach to decreasing the risk of cyberattacks.

Proactive Versus Reactive Software Supply Chain Security

FinServ organizations too often treat software supply chain security with the same mindset as they approach cybersecurity, implementing tools designed to generate alerts that need to be investigated. While this approach can fit well with Security Operations Center (SOC) oriented FinServ organizations, in practice it just ends up adding to their workload, exacerbating cybersecurity burnout. WIth burnout comes slower response times when alarms go off, or worse, alarms may go uninvestigated altogether.

> "Two-thirds of respondents are either implementing or adopting software supply chain security, but most are failing to see success[1]"

While Gartner suggests these results may be due to Software Supply Chain Security (SSCS) initiatives being uncoordinated across the organization, ActiveState has found that a significant driver of failure can be the result of taking a reactive rather than a proactive approach to supply chain security.

For example:

- Most FinServ organizations store their open source dependencies in an artifact repository and restrict access to public repositories to prevent the download of compromised components. But the security, compliance, and IT approval process required prior to adding a new dependency (or even a new version of a dependency) to the repository can significantly slow down development cycles. As a result, FinServ organizations can fall into the trap of standardizing on a set of "golden dependencies" that become vulnerable and out of date.
  - » A proactive approach enables evaluating and approving dependencies on a continual basis BEFORE they become vulnerable or outdated. Alternatively, FinServ organizations should work with a third party provider that adheres to their OSS governance guidelines.
- Statistics show that billions of older, vulnerable dependencies continue to be downloaded year after year from public repositories, despite the fact that a fixed version is available, according to Sonatype's latest State of the Software Supply Chain. These older versions are often flagged during the CI/CD pipeline, creating more work for DevOps.
  - » A proactive approach ensures that runtime environments are always built with up-to-date dependencies.
- Artifacts built in a non-reproducible manner may differ from one build to the next, raising the security and integrity risk of the built artifact.
  - » A proactive approach builds all artifacts with a hardened, tamper-proof, reproducible build system instead.

1. 2023 Gartner "Technology Adoption Roadmap for Large Enterprises Survey"

- Prebuilt open source dependencies sourced from a public repository are unsigned and lack a Provenance Attestation to help verify the security and integrity of the component. Typically, binary scanners are employed to identify whether they are fit for use, but scanners inevitably generate alerts that must then be investigated. These alerts can be numerous, especially at the start of a new project, but cannot be ignored since one in eight open source downloads includes a known risk.[2]
    - » A proactive approach sources signed dependencies and their attestations from a trusted vendor.
- Vulnerabilities can emerge at any point in the lifecycle of a project, but are typically actioned only when found during a CI/CD run. This delays deployment since the vulnerability must now be investigated by developers.
    - » A proactive approach incorporates a Vulnerability Exploitability eXchange (VEX) document in the CI/CD pipeline, allowing DevOps to automatically validate whether a previously discovered vulnerability is actually exploitable in the context of the software product.

And so on. Whether at the import, build, or consumption stage, taking a proactive approach to ensuring the security and integrity of your software supply chain can dramatically improve outcomes.

> “Taking a reactive approach to SSCS inevitably results in rework, increasing friction and slowing down development cycles.”

While automation can help, automating reactive processes typically ends up creating more follow up work than it resolves. Automated proactive tools, on the other hand, can significantly decrease the costs and resources required to secure your software supply chain. For example, some solutions allow for policy to be preset as part of a workflow that automatically evaluates and prevents exceptions to corporate guidelines

This is what the ActiveState Platform can provide you: an automated way to create secure. reproducible builds of open source components from vetted source code using a hardened, tamper-proof build system. Provenance Attestations and Software Bills Of Material (SBOMs) are created for every built artifact, including each component and the overall runtime environment, as well. On use, each component is verified by the ActiveState command line tool (State Tool) to ensure its security and integrity have not been compromised.

In this way, you can take a proactive approach to securing your open source supply chain from import through the build process to consumption by developer and DevOps teams.

2. 9th Annual State of the Software Supply Chain, Sonatype.

The FinServ Threat Landscape

FinServ organizations are subject to multiple types of cyber threats on a regular basis because, simply put, that's where the money is. While phishing (i.e., fraudulent emails) continues to be the main catalyst in 93%[3] of all data breaches, the threat is well known, and mitigation techniques should already be in place.

Securing the software supply chain for applications (whether developed in-house or via a partner) should be the focus of any proactive FInServ organization that wants to get ahead of bad actors. While many threats exist across the entire import-build-consume supply chain, the ones FinServ should be primarily concerned with include malware and vulnerabilities.

Importing Malware Components

Malware (short for malicious software) refers to hacker-created intrusive software designed to steal data, damage computers or compromise computer systems. While malware vectors of attack can include phishing and vulnerability exploits, more and more malware is simply being embedded in the open source components of your software supply chain. This is especially true of the most common kind of malware FinServ is likely to encounter: ransomware. Ransomware is typically designed to encrypt data across the enterprise, locking users out, but can also exfiltrate that data prior to encryption.

> "FinServ organizations are hit by ransomware attacks at a higher rate (65%) than the cross-industry average (59%), with ransoms averaging $2M[4]"

While 51% of FinServ firms pay the ransom to regain access to their data (ibid), they may also face demands from the cybercriminal to pay a second ransom in order to prevent leaking the exfiltrated data on the dark web.

3. https://www.getastra.com/blog/security-audit/phishing-attack-statistics/
4. Sophos "State of Ransomware in Financial Services 2024"

Combating ransomware in the software supply chain requires:

- **Automated Code Scanning –** open source components should always be scanned using common threat detection tools such as Software Composition Analysis (SCA) for binaries and/or Static Application Security Testing (SAST) for source code.
    - » Keep in mind that while scanning works well on non-obfuscated code, hackers typically use compression, encryption and other techniques to hide their malware from scanners. Consider using only vetted, securely built and signed open source components from a trusted vendor.
- **Dynamic Analysis –** Dynamic Application Security Testing (DAST) tools can be successful at catching obfuscated malware when the application is run.
    - » Care must be taken to not only exercise the code thoroughly, but to do so in an isolated, ephemeral network segment so as to avoid infecting the organization.
- **Repositories & Firewalling –** a typical FinServ strategy is to deploy an artifact repository from which vetted open source packages are made available to developers. Coupled with firewall blocking of open source repositories (as well as other sites that may pose a risk), this method can be effective in limiting risk.
    - » As noted previously, however, open source dependencies in artifact repositories can quickly become out of date and vulnerable. FinServ stakeholders must be proactive in approving new versions of dependencies for use, and replacing older versions.
- **Signed Code –** rather than obtaining open source from public repositories and applying the practices listed above, some FinServ organizations use only open source code that has been securely built and signed by a trusted partner.
    - » Ensure that artifacts such as Provenance Attestations are collected and maintained as proof of trustworthiness of the components.

Building Components Securely

Secure software development best practices extend from developer desktops through code repositories to the CI/CD build system. Each of these environments should be implemented with secure controls to ensure code is being developed, checked in/out, and built in a manner that minimizes risk.

Build systems are far too often given short shrift when it comes to security for a number of reasons, not least of which is the complexity of creating a declarative pipeline that allows control over every step of the build process.

> “However, as Solarwinds proved in December 2020, build systems must be hardened to eliminate attack vectors such as the ability to inject compromised dependencies into the build process prior to the signing step.”

When creating a build system, FinServ organizations should implement a number of best practices, including:

- **Hardening –** create a dedicated build service that runs on a minimal set of predefined, locked down resources rather than a developer’s desktop or other arbitrary system that can offer a larger attack surface to hackers.
- **Automation –** execute only scripted builds using scripts that cannot be accessed and modified within the build service, preventing exploits.
- **Ephemeral, Isolated Build Steps –** every step in a build process must execute in its own container, which is discarded at the completion of each step. In other words, containers are purpose-built to perform a single function, reducing the potential for pollution or compromise.
- **Hermetic Environments –** containers must have no internet access, preventing (for example) dynamic packages from including remote resources.
- **Signing –** all output artifacts must be digitally signed to ensure they haven’t been tampered with between creation and use.
- **Reproducibility –** all built artifacts must be verified against a hash to ensure that the same “bits” input always result in the same “bits” output.

Consuming Vulnerable Components

Modern FinServ software is primarily built from open source code, which typically forms ~80% of the codebase. While OSS vulnerabilities continue to escalate year on year, this is primarily due to the fact that more and more organizations are not only using more OSS, but also reporting vulnerabilities in a more systematic manner.

> "Less than half of cybersecurity professionals claim to have high (35%) or complete visibility (11%) into vulnerabilities within their deployed applications. At best, only around half of all organizations (51%) can claim to have moderate visibility into their vulnerabilities[5] "

While FinServ tends to be more proactive about vulnerabilities than non-regulated industries, the key issues remain the same:

- **Discoverability –** creating a comprehensive catalog of all open source components deployed across the extended enterprise is key to decreasing Mean Time To Identification (MTTI) of vulnerabilities. Software Composition Analysis (SCA) tools are commonly used to automate the catalog creation process (replacing the previous solution of manually populated spreadsheets), but since they essentially reverse engineer a compiled artifact, no two SCA tools generate the same catalog. SBOMs are no substitute since they are often generated by those same tools.
  - » Generating an open source catalog from an application's source code will not work if that source code contains precompiled open source components. By contrast, ActiveState builds all open source components from source code, including linked C libraries, providing a truly comprehensive catalog.
- **Remediabiility –** Mean Time To Exploit (MTTE) has decreased to as little as 22 minutes from the announcement of a vulnerability until the first instance of its exploitation in the wild. Unfortunately, the lengthy process of investigating a vulnerability, and then patching/upgrading, recompiling, retesting and redeploying the affected application exposes FinServ to significant risk. Employing a CI/CD system that features extensive test suites as well as automated deployment to production dramatically decreases Mean Time To Remediation (MTTR), but not if there is a need to wait for an upstream OSS patch.
  - » By comparison, ActiveState continually sources downstream OSS component patches, and then rebuilds the affected open source software in minutes, ready to be pulled into your CI/CD process for testing and redeployment.

5. Tripwire, 2022 Vulnerability Management Report

Conclusions

The innovation, cost and accelerated development benefits of using OSS still outweigh the risks, despite the fact that (by some estimates) security and compliance risks have almost tripled over the past year. Collaboration across software development, security, compliance, IT, vendor management and legal departments is the key to minimizing risks and maximizing benefits.

It has never been more imperative to secure the software supply chain to decrease the risk of cyberattacks against vulnerable applications, especially given that malware-infected packages in 2023 totaled more than twice the total number discovered for all years combined since 2019.[6]

While the marketscape is complex, involving both traditional AppSec vendors and emerging software supply chain vendors, very few vendors offer a comprehensive solution that spans the entire import-build-consume chain. But cobbling together multiple point solutions increases complexity, maintenance overhead, and ultimately costs.

Instead, consider acquiring a single, centralized platform like ActiveState that can provide a collaboration point for all stakeholders, and can be easily integrated with existing software development processes. When that platform can also provide an automated, end-to-end solution that allows for a proactive approach to securing the software supply chain, the benefits far outweigh the costs.

6. Sonatype, State of the Software Supply Chain

ActiveState

Tame open source complexities

Ready to Tame the Complexity of your Open Source?

Contact us to learn more and get a demo

www.activestate.com
solutions@activestate.com