ActiveState

Business Leader's Guide to Establishing Software Supply Chain Trust

Executive Summary

Leaders concerned with the security of the software they produce and purchase need to be aware of an emerging attack vector. SolarWinds, Codecov and many other software vendors have been subject to bad actors who are exploiting weaknesses within the digital supply chain to penetrate internal development environments and compromise software development processes. The result is tens of thousands of end customers compromised by simply installing a software update from a trusted vendor.

Business leaders need new ways to manage software supply chain risks, whether they're buying software or creating it. Two key approaches stand out:

- **Software Security** - managing software vulnerabilities
- **Process Integrity** - managing the security of software development that spans the:
  - Import Process - software that enters the organization requires strict validation controls
  - Build Process - code must be built with a secure build service to prevent compromise
  - Run Process - code needs to be validated for integrity and security at runtime

For software vendors, the ActiveState Platform provides a turnkey secure supply chain for open source languages by implementing key import, build and run controls. Unlike point solutions that typically focus only on security, the ActiveState Platform ensures both the security and integrity of the entire software development lifecycle.

A New Generation of Cyber Attacks

The alarming rise in ransomware designed to hold your organization hostage has resulted in leaders taking their eye off the fast-moving threat of software supply chain attacks. It's one thing to have your digital assets held for ransom, and quite another to lose 40% of your stock value and precipitate an Executive Order from President Biden, as happened with SolarWinds.

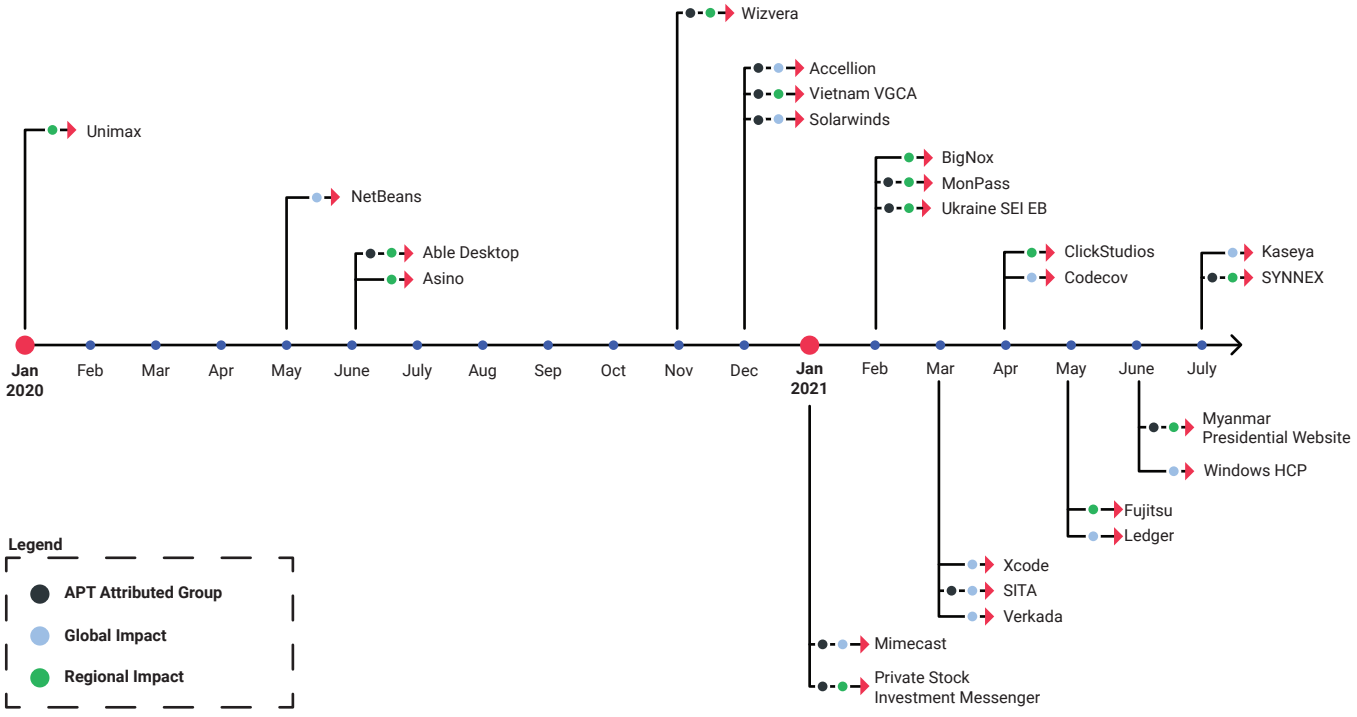
With a 650% increase in supply chain attacks in 2021 (compared to a 430% increase year on year in 2020), these kinds of attacks are increasing exponentially for the simple reason that a single breach of a software vendor can potentially compromise thousands, or even tens of thousands of downstream customers.

This kind of leverage is possible due to the inherent weaknesses in open source ecosystems, which typically feature hundreds of thousands of independent developers that submit millions of assets to a central repository that provides little to no guarantees of their security or integrity. The message is clear: user beware.

Anatomy of a Supply Chain Attack

Typically, a supply chain attack starts with the compromise of an open source asset, which, once it enters an organization, provides a potential vector of compromise. Alternatively, since most development environments are connected to the internet, developer and/or build systems may be compromised directly. Software produced by a compromised organization is a risk to all of their customers, and anyone else downstream. The routine task of distributing, installing or updating software from a trusted vendor has now become a significant risk. In effect, software vendors are now the frontline of security for their customers.

A visualization of recent software supply chain attacks is instructive in demonstrating both their accelerating reach and rate:



Source: ENISA Threat Landscape for Supply Chain Attacks

A description of a few key attacks will serve to illuminate the three most popular ways attackers are currently undermining the software supply chain, including hijacking updates, undermining code signing and compromising open-source code:

- **SolarWinds** - spyware was implanted into Orion, their IT management and security platform, infecting thousands of corporate customers, as well as the U.S. Treasury, the State Department, and the Department of Homeland Security when they installed it.
  - **Vector of Attack** - a compromised software build system allowed the attacker to add a malicious version of a DLL, which was then signed by the legitimate SolarWinds code signing certificate.
- **Codecov** - as a supplier of code management and audit solutions, they had their public-facing bash uploader script hacked, granting the attacker access to customers' Continuous Integration (CI) environments.
  - **Vector of Attack** - a compromised upload script granted attackers access to the build systems of software vendors like IBM, HPE and others.
- **Apache Netbeans IDE hack** - malware uploaded to public GitHub repositories targeted the Netbeans IDE with the goal of installing a Remote Access Trojan (RAT), which would grant hackers access to developer environments.
  - **Vector of Attack** - public code repositories were infected with the Octopus Scanner malware.
- **NotPetya Cyberattack** - power plants, banks, metro systems, and the world's largest container shipping company were just some of the victims of malware delivered through the updating process of an accounting software package (created by a small firm, Linkos Group) commonly used by companies in Ukraine.
  - **Vector of Attack** - a software delivery environment was compromised by attackers that hijacked the Linkos Group's update servers.
- **ShadowPad Backdoor** - a backdoor dubbed ShadowPad was injected into NetSarang's network management software suite, and was pushed through a software update to banks, utilities, and other customers.
  - **Vector of Attack** - similar to SolarWinds, a compromised build system allowed the attacker to add a malicious version of a DLL, which was then signed by the legitimate NetSarang code signing certificate.

Changing Security Priorities

When organizations purchase software, whether applications for deployment or tools for code development, the expectation is that they are buying secure software. Software supply chain security is actually the responsibility of both vendors and customers, but at this time it isn't a priority for either of them because:

- Security does not directly contribute to revenue
- Security is a secondary concern for software vendors who typically promote a culture of "move fast and break things"

After all, software is usually purchased for it's features and functionality, rather than its security. Security is expected to be "built in" rather than a feature itself. However, this attitude is changing. Companies are beginning to request proof that the software supply chain of their vendors is secure by having the vendor fill out complicated security questionnaires. President Biden's Executive Order has also made this a requirement for US government departments as of October 2022. Other organizations perform Proof Of Concepts (POCs) that include independent security testing as part of their acceptance process.

As a result, software vendors should increasingly expect contracts to include language that holds them accountable for the security risks of their software, and by extension, the supply chain used to create it. Features may still be the driver of a sale, but security is more and more becoming the blocker.

Mitigating Software Supply Chain Risks

There are many best practices that organizations can put in place to help mitigate the risk of supply chain compromise. The US Government's National Institute of Standards and Technology (NIST) provides guidelines that help define a Secure Software Development Framework (SSDF), which organizations can not only adopt within their own development processes, but can also use as a framework to help identify which of their vendors take their software supply chain security seriously, as well.

The SSDF details a number of policies, practices, and tasks that any organization can integrate within their existing processes across three main categories: Preparation, Protection and Response.

Securing the Supply Chain - Preparation

Preparation refers to the policies and procedures that need to be put in place in order to ensure that a software supply chain is secure BEFORE coding begins. For example:

- Establish policies that define the architecture, development infrastructure, and software life cycle security requirements, and review/update them annually.
- Integrate security roles and responsibilities into the software design and development teams, and empower them with sufficient training.
- Evaluate existing tools for security, and purchase new, (preferably automated) secure tooling where necessary.

Securing the Supply Chain - Protection

Protection refers to the software tools and practices that must be put in place to ensure the software supply chain is secure during code development and testing. For example:

- Store all code in a secured versioning system, and ensure all code creation/updates follow secure coding practices, including peer reviews prior to acceptance.
- Create a Software Bill of Materials (SBOM) that identifies all the components the software is built with, as well as each component's vulnerability status.
- Ensure all components used by developers are built with a secure build service designed to generate reproducible builds.
- Use an access-restricted repository to store all your approved components, including third party/open source components, which must be vetted on import.
- Employ cryptographic hashing and code signing to ensure the integrity and provenance of each component.
- Create threat models, attack models, and attack surface maps, and then implement testing/tooling (penetration testing, static code analysis tooling, etc) to assess the security risk of every build.

Securing the Supply Chain - Response

Response refers to the ability of the organization to identify and respond in a timely manner to software supply chain risks. For example:

- Monitor for and flag vulnerabilities as they are found in the components of your SBOM.
- Establish a process (preferably automated) to resolve identified vulnerabilities in a timely manner.

ActiveState Platform: End-to-End Supply Chain Security

The ActiveState Platform provides a next generation solution to securing your software supply chain. It follows many of the recommendations suggested in the NIST secure development framework, but simplifies implementation by offering an out-of-the-box service that fits with your organization's existing software development processes.

Integrating multiple “best in class” solutions is both cost prohibitive and time consuming. In the meantime, your software supply chain remains vulnerable. The ActiveState Platform provides a turnkey solution that ensures the security and integrity of your open source supply chains with minimal time and effort. As a result, organizations can quickly benefit from:

- **A Comprehensive SBOM** for each of your projects.
- A **Secure Build Service** that not only creates reproducible builds, but also provides provenance for all built components.
- **Software Integrity** that ensures your existing import, build and run processes haven’t been compromised.
- **Automated Vulnerability Remediation** that not only identifies vulnerabilities, but also allows developers to resolve them in minutes, not days.

As a result, organizations can ensure the integrity and security of the open source software they use to develop their digital products and services.

Conclusions

The current software supply chain cannot be blindly trusted. Its integrity is being actively exploited by bad actors, and its security is being unintentionally undermined by the introduction of vulnerabilities. However, vendors and customers can mitigate supply chain risk through the use of technical controls as well as timely, two-way communications that push actionable information to customers and pull security requirements back to the vendor.

Software vendors should look to physical supply chain best practice case studies from companies like Toyota, who have realized greater control and reliability by limiting their supply chain. By adopting fewer suppliers that offer higher quality, software vendors can gain the same kind of control over their supply chain that automotive companies have realized. That will mean allocating budget for next generation solutions that can consolidate your open source software sources while ensuring their integrity and security. But by adopting a secure, open source supply chain, your organization can:

- **Make security a selling point.** Software that's more secure by design is a feature you can market, getting ahead of your competitors.
- **Involve the development team.** The "shift left" movement of DevSecOps has never really caught on because of perceived disruption to the Dev teams. Bake in security from the start in your supply chain to avoid the disruption.
- **Examine your software vendors' supply chain.** The software you create can only ever be as secure as the software supply chain of the development tools you use.

Trustworthiness has always been a critical component of any brand, but it's increasingly becoming a differentiator for software vendors. Trustworthiness means not only that the software and services you provide do what you say they will, but also that they won't jeopardize your customer's business by compromising their security. No software vendor wants to be the next SolarWinds. Leaders have a special role in ensuring the integrity and security of their software supply chain, including both the software they sell and the development tools they use.

ActiveState

www.activestate.com

Toll-free in NA: 1-866.631.4581

solutions@activestate.com