REVERSINGLABS

REPORT

The State of Software Supply Chain Security

ATTACKS ON SOFTWARE SUPPLY CHAINS SOARED IN 2022.
HERE ARE THE MAJOR TRENDS — AND WHAT LIES AHEAD IN 2023.

Contents

Executive Summary

Almost two years after word of the SolarWinds hack first spread, software supply chain attacks show no sign of abating.

In the commercial sector, attacks that leverage malicious, open source modules continue to multiply. Enterprises saw an exponential increase in supply chain attacks since 2020, and a slower, but still steady rise in 2022. The popular open source repository npm, for example, saw close to 7,000 malicious package uploads from January to October of 2022 — a nearly 100 times increase over the 75 malicious packages discovered in 2020 and 40% increase over the malicious packages discovered in 2021.

The Python Package Index (PyPi) was also flooded with tainted open source modules designed to mine cryptocurrency and plant malware, among other things. These attacks were consistent with what researchers observed in 2021, when attackers commonly used techniques such as dependency confusion and typosquatting. As in previous years, high-profile organizations including Samsung and Toyota found themselves embarrassed by secrets exposed through open source repositories that were maintained internally or by third-party contractors.

The attacks have increased the focus on software supply chain security. Following the issuance of the Biden Administration's May 2021 Executive Order on Improving the Nation's Cybersecurity (EO 14028), the past year saw new federal guidance for tightening supply chain security. That included a practice guide for software suppliers to the federal government issued by the Enduring Security Framework (ESF) Software Supply Chain Working Panel. Also issued: a September, 2022 memorandum from the Office of Management and Budget (M-22-18) that requires software firms to attest to the security of software and services they license to Executive Branch agencies. (See "New policy mandates: Uncle Sam wants YOU! (to secure your code)" below).

In the coming year, software publishers with federal contracts will need to clear higher bars for software security to meet the new guidelines, including having to attest to the security of their code and — in some cases — produce SBOMs that provide a roadmap for tracking down supply chain threats. Given that the threat of supply chain attacks goes beyond publishers that sell to the federal government, all organizations that develop software will need to take similar steps to keep ahead of these threats.

They will need new tools and approaches to do that, and this report offers recommendations to prevent supply chain compromises. These include increased scrutiny of open source risks and better coordination between development teams and security operations centers (SOCs) to bridge the gaps in both the monitoring and detection of supply chain threats and attacks.

What counts as a supply chain attack?

A software supply chain attack is an attempt to exploit a weakness at a given stage in the software supply chain — the sequence of steps leading to the creation of a piece of software (a.k.a "software artifact"). Supply chain attacks try to access and manipulate source code, build processes, or update mechanisms of legitimate applications as a means to an end: planting malware; stealing data; sowing disruption; and so on. Note that software supply chain attacks aren't the same as attacks on software. For example: attempts to exploit software vulnerabilities for malicious purposes (privilege escalation, installing malware, etc.) are not software supply chain attacks because they target the finished software artifact, not the supply chain.

A recent history of software supply chain attacks

Software supply chain compromises date back more than four decades, when a covert CIA operation allegedly fed compromised industrial control system (ICS) software to a Soviet gas pipeline operator, resulting in a massive explosion. But for most of the modern Internet era, attacks transmitted by way of tampered software were the exception, rather than the rule.

Instances of software supply chain attacks are increasing. Half of the 42 supply chain attacks ReversingLabs has documented occurred in 2021 and 2022. To view the full history, read the article, *A Partial History of Software Supply Chain attacks*

Much of the recent history of cyber threats, attacks and compromises centers on the exploitation of software vulnerabilities, such as the notorious "Eternal Blue" exploit of the MS17-010 vulnerability in Microsoft's Server Message Block that powered the explosive WannaCry and NotPetya malware infections. Or it hinges on the placement of designed-malicious wares — like ransomware — on high-value endpoints and networks, often as a result of successful phishing and social engineering attacks on privileged users.

But that history of exploits and malicious attachments is ceding ground as malicious actors adapt their methods and strategies to find new avenues into sensitive private- and public-sector environments. According to the ReversingLabs report, A Partial History of Software Supply Chain Attacks, attacks on software development organizations and software supply chains are increasing at a dramatic rate. In fact, of the 42, half occurred in 2021 and 2022.

The logic behind the increase in these attacks is easy to understand: open source software libraries and components form the foundation of, by some estimates, 75% of applications. With an increasing reliance on open source packages, the attacks on open source repositories have become a matter of "fishing where the fish are," while also sidestepping many of the security and detection tools that have been deployed to protect more traditional targets.

Uncle Sam wants YOU! (to secure your code)

For software firms that do business with the U.S. Government, new guidelines raise the bar for software and supply chain security. Here are some recent developments that firms should pay attention to in 2023.

EXECUTIVE ORDER 14028

The Biden Administration's Executive Order (EO) for Improving the Nation's Cybersecurity, released in May of 2021, laid out new guidelines for securing software used by federal agencies. Among other things, it set new guidelines for software supply chain security, and empowered the Office of Management and Budget (OMB) to require agencies to comply with those guidelines.

MEMORANDUM M-22-18

Following a directive in the Executive Order, the Office of Management and Budget released a memo in September directing federal agencies to comply with NIST guidance on software supply chain security, including compliance with NIST Special Publication 800-218 and subsequent NIST guidance on software supply chain security. The memo sets a timeline for federal agencies to communicate new software security requirements to their vendors, and for software publishers that sell to federal agencies to attest to the security of their wares. It also opened the door to federal agencies requiring the creation of SBOMs that they can use to identify, track and monitor individual components within larger applications and services.

Trends associated with digital transformation are also increasing organizations' exposure to software supply chain risk. Organizations in both the public and private sectors have embraced hardware, software and services characterized by always-on Internet connections, remote, cloud-based management and extensive software supply chains that consist of both open source and third-party code.

Headlines point to the security cost of such changes. Here are just a few:

- Russian state-sponsored actors used compromised remote management software to push out wiper malware embedded in software updates to satellite modems used by the Ukrainian military.
- Hackers planted information-stealing malware in rogue Javascript npm packages that mimicked common open source packages used by developers for Microsoft's Azure cloud platform.
- A major car manufacturer exposed information on hundreds of thousands of customers by exposing a private key for accessing a customer database in a public source code repository.

These attacks feed on practices and behaviors that are ubiquitous. Among them: a heavy reliance on centralized, cloud-based infrastructure; fast-moving DevOps practices that have greatly increased the cadence of software releases, in part through heavy use of third-party commercial off-the-shelf and open source modules to speed development; and an increased reliance on centralized auto-update mechanisms to facilitate the rapid release cycles of modern, cloud-based applications and services.

Supply Chain Security in 2022: Threats Multiply

Attacks on open source repositories have skyrocketed in the past decade, outpacing vulnerabilities found in those repositories. Between 2018 and 2021, for example, attacks on npm and PyPI increased by 271% and 414%, respectively. That trend continued in 2022.

The sheer scale of supply chain attacks threatens to overwhelming platform providers. For example, malicious actors leveraged automatic submission features in npm to submit more than 900 malicious npm packages through a single submitter account in August, 2022. And researchers discovered similar attacks involving dozens or scores of malicious packages on other platforms, including PyPi.

As threats multiplied in 2022, here are the key trends ReversingLabs researchers observed over the last 12 months:

SUPPLY CHAIN ATTACKS SURGED

The distributed and ubiquitous nature of software development, and the absence of a governing body responsible for monitoring the security and integrity of development organizations, made compiling a comprehensive report on software supply chain threats and attacks virtually impossible.

The Enduring Security Framework

Also in 2022, the NSA, CISA and ODNI released The Enduring Security Framework (ESF), a new set of practice guidelines that is a roadmap for software vendors that do business with the federal government on how to implement a secure software development framework (SSDF) as envisioned by the EO.

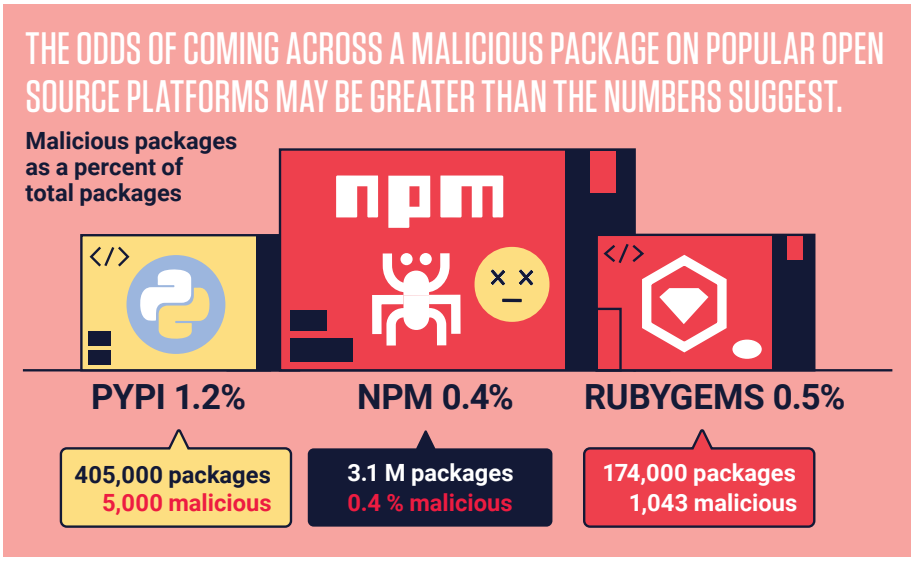
The ESF practice guidelines call on federal agencies and their software suppliers to develop proficiency in areas like binary scanning and software composition analysis (SCA), which can detect unknown files and open source components, and their associated security weaknesses.

The Framework also includes practice guidelines devoted to developing secure code such as:

- Focusing on critical code, such as that requiring elevated privileges, accessing sensitive resources or using or implementing cryptographic functions
- Removing "ease of development" features like temporary back doors that find their way into production code
- Addressing risks posed by malicious insiders, rogue developers and compromised development systems
- Mapping newly created code back to clearly identified features, and implementing authentication for code check-ins to guard against compromised development systems

However, there are some objective measures researchers can use as a rough assessment of the state of software supply chain security and the prevalence of malicious activity targeting development organizations.

One is the prevalence of malicious packages posted to prominent open source repositories such as npm, PyPi, and RubyGems. A ReversingLabs analysis of supply chain attacks like IconBurst and Material Tailwind shows that malicious actors are leveraging trust in open source software to plant malicious code within organizations. So malicious packages on platforms such as npm, while only a small part of the overall supply chain threat landscape, are telling — a possible "canary in the coal mine" indicating that more sophisticated, harder-to-detect attacks may be out there.



MALICIOUS NPM PACKAGES JUMPED BY MORE THAN 40%

Malicious package submissions to both the npm and PyPi repositories increased substantially in the last two years, according to ReversingLabs research data. From January to October, 2022, 6,977 malicious packages were uploaded to npm and 1,493 to PyPi (The data researchers used is based on the timestamp for package creation on both npm and PyPi).

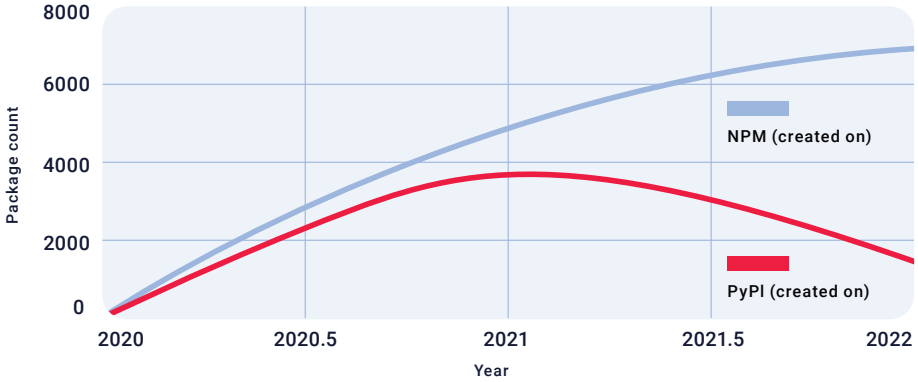


Figure 1. Malicious package uploads to the npm code repository showed a 41% increase in 2022 over 2021, when researchers detected 4,940 packages. And the 2022 numbers represent more than a 9,000% increase over 2020, when researchers detected just 75 malicious npm packages. *Source: ReversingLabs*

Five ESF Recommendations for securing executable code

The ESF also includes recommendations development organizations can follow to secure executable code against exploits. These include calls to:

- **Develop comprehensive security requirements that include compliance regulations.**
- **Create threat models for all critical software components and elements of your build pipeline, including source code repositories, build systems, and so on.**
- **Develop test plans to assess each requirement providing good "code coverage."**
- **Provide adequate staffing and testing resources to execute test plans**
- **Perform security testing of each software component in line with NIST SSDF guidelines, including:**
  - **Static and dynamic application security testing of all source code**
  - **Fuzzing of all software components to verify expected behaviors**
  - **Periodic penetration testing on a regular basis**
  - **Documentation of the results of all security tests**

MALICIOUS PYPI PACKAGES: A QUIETER YEAR BUT STILL WAY UP FROM 2020

In contrast to npm, the PyPi repository saw a nearly 60% decrease in malicious package uploads in 2022, going from 1,493 packages to 3,685 in 2021. But malicious activity since 2020 is still up more than 18,000% over 2020, when just eight malicious packages were detected.

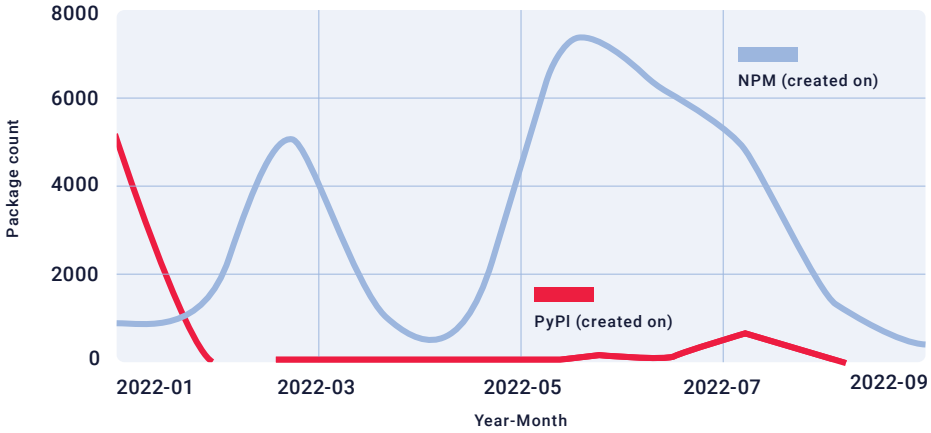


Figure 2. Malicious package additions on npm and PyPi by month.
Source: ReversingLabs

Summertime brought a big spike in malicious activity in 2022: Of the nearly 7,000 malicious npm packages added to the repository, more than 86% were added during the months of June, July and August. Many of those have been linked to large-scale campaigns such as the July "CuteBoi" campaign, first reported by Checkmarx, that placed 1,200 malicious modules on npm that contained Eazyminer crypto mining software.

The PyPi repository also saw a modest increase in malicious package submissions in August. Overall, however, 85% of malicious PyPi packages were found in January. Those were connected to an apparent "proof of concept" dependency confusion attack, linked to a common account, that targeted prominent commercial and open source projects, Sonatype reported.

Those campaigns exposed weaknesses in existing open source platforms. For example, both the January and August surges in malicious packages exploited automatic submission features that allowed a single user to submit large numbers of malicious packages, sidestepping security features such as two-factor authentication. That's a big reason why the mass publication of malicious packages to public repositories requires urgent attention.

MALICIOUS NPM PACKAGES WERE THE BIGGEST SLICE OF THE PIE

Overall, malicious packages submitted to the npm repository represent the lion's share of supply chain attacks that target open source repositories. (Note that supply chain attacks on third-party, commercial software are harder to document, since many aren't publicly disclosed.)

According to ReversingLabs data, more than 12,000 malicious npm packages were submitted through October, 2022, while less than half that number were submitted to PyPi and just over 1,000 were submitted to the RubyGems repository.

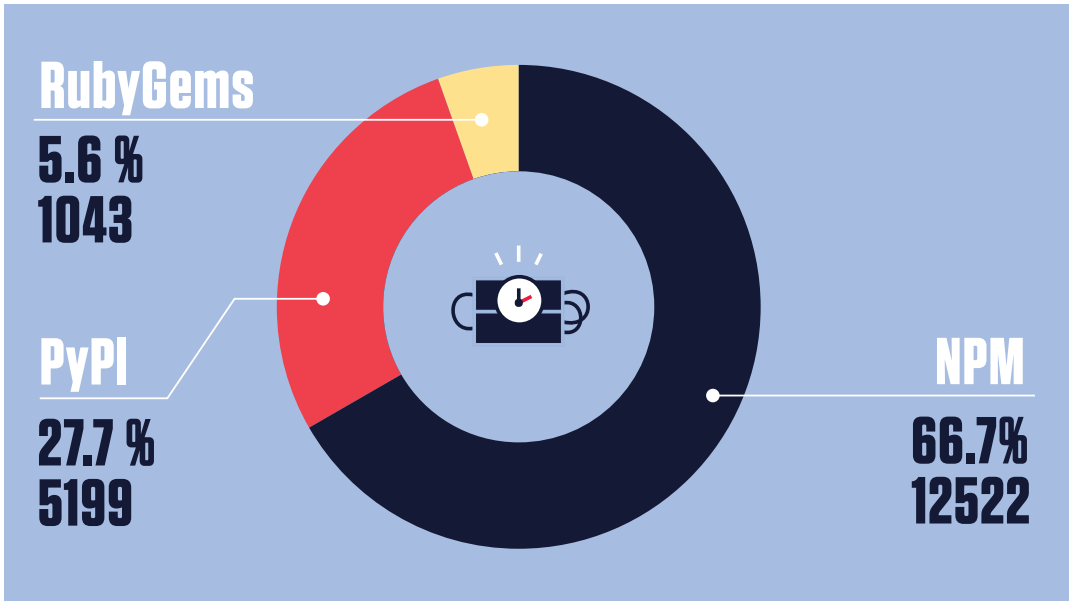


Figure 3. Total malicious packages to date on npm, PyPi and RubyGems. *Source: ReversingLabs*

Why the big discrepancy in where malicious packages were placed (or at least where they were detected)? As with the story of the thief who said he robbed banks because "that's where the money is," malicious actors are focused on npm because that's where the code is. Npm currently hosts more than 3.1 million projects, compared to just 407,000 on PyPi and 173,000 on RubyGems.

> More projects also means more developers, more accounts and more potential avenues of attack.

Karlo Zanki
Threat Researcher, ReversingLabs

However, malicious packages still make up a small slice of the overall open source ecosystem. For example, there are about 3.1 million npm packages, of which just 0.4% were malicious. For PyPi, the tally was 5,000 malicious packages out of a population of more than 405,000, just over 1% of the total. For RubyGems, the number is 0.5% (1,043 malicious packages detected out of a population of around 174,000 packages).

Malicious open source packages are the (rare) exception rather than the rule. Unfortunately, it takes just one malicious package to cause a major supply chain disruption. And attackers' efforts to impersonate popular packages on these platforms means that coming across a malicious module may be easier than you might think.

Supply Chain Red Flag: Obfuscated Code

When looking for suspicious packages in massive open source repositories, one natural question is: "what to look for?" One telltale sign, based on ReversingLabs research, is the presence of obfuscated code in public code repositories. For example, the Material Tailwind supply chain attack came to the attention of our Titanium Platform's behavior indicator because it contained code obfuscated with JavaScript Obfuscator.

Upon close inspection, ReversingLabs researcher Karlo Zanki noted that the package description used was, in fact, copied from another npm package named tailwindcss-stimulus-components. The threat actor took special care to modify the entire text and code snippets to replace the name of the original package with Material Tailwind. The malicious package also successfully implements all of the functionality provided by the original package.

POST-INSTALL SHENANIGANS

But behavior indicators don't lie. One of the JavaScript files present in the package contains obfuscated code. It also happens that this *tailwindcss-stimulus-scripts.min.cjs* file is also a declared as postinstall script in *package.json* file, which gets executed immediately after package installation. This is a popular mechanism for achieving code execution among threat actors. In brief: an obfuscated script that is set to run immediately after installation is a (big) red flag.

TYPOSQUATTING ATTACKS PROLIFERATED

Researchers discovered many typosquatting attacks on common open source repositories in 2022, including the Material Tailwind attack, which ReversingLabs reported in September. In that attack, malicious actors posted a package that played on the names of two massively popular libraries, Tailwind and Material Design, each of which netted millions of downloads. The Material Tailwind package posed as a helpful development tool, but included a post-install script that downloaded a password-protected zip file containing a custom-packed Windows executable capable of running PowerShell scripts.

Likewise, in March, researchers at DevOps software vendor J-Frog discovered of more than 200 packages on the npm repository that targeted developers using packages under the @azure scope, as well as @azure-rest, @azure-tests, @azure-tools and @cadl-lang, with typosquatting attacks. The legitimate packages targeted in the attack were downloaded tens of millions of times each week, J-Frog noted at the time.

But npm wasn't the only open source repository to suffer typosquatting attacks. In August, ReversingLabs researchers discovered 40 malicious PyPi packages that, once installed, beaconed to a malicious command-and-control domain. That domain, python-release.com, pushed malicious code to the infected systems, including a file containing the Parallax RAT malware. As with npm, the malicious PyPi modules often used names (e.g. statmodel, statmodels) as well as command-and-control domains that mimicked the names of commonly used PyPi packages and support infrastructure.

LEAKED SECRETS PLAGUED SECURITY TEAMS

Sensitive information leaked through open source modules and platforms was another prominent theme in 2022 that affected organizations ranging from Fortune 500 companies to prominent government agencies.

In September, for example, the U.S. Department of Veterans Affairs acknowledged that a federal IT contractor accidentally published source code containing sensitive information, including hard-coded administrator account privileges, encrypted key tokens and database table information, to a public GitHub repository. Public reports indicated that secret keys used to access at least 12 applications were exposed after the contractor copied source code from a VA-managed GitHub account and published it on a personal GitHub account. The breach persisted for about two months before being detected, during which time entities associated with six foreign IP addresses copied and reproduced the code.

Around the same time, Toyota Motor Corp. disclosed that a contract web developer for its T-Connect telematics system published code for the T-Connect website, including a private database access key, to a public code repository, leaking customer emails and other data. The leak, which happened in 2017, wasn't detected until 2022.

Typosquatting up close: the IconBurst Attack

IconBurst was one of the more prominent typosquatting attacks that ReversingLabs uncovered in 2022. Here's what the researchers found:

WHAT: **IconBurst was a widespread software supply chain attack involving malicious packages offered via the npm package manager. ReversingLabs researchers identified more than 50 npm packages containing obfuscated Javascript designed to steal form data from applications or websites using the malicious packages.**

WHO: **The actors responsible for the attack are not known. However, the malicious packages use exfiltration domains, with a consistent naming pattern suggesting a common actor and a unified campaign.**

HOW: **The malicious modules collected form data using jQuery Ajax functions, then exfiltrated that data to domains controlled by malicious actors such as ionicio.com, a play on the legitimate framework domain ionic.io. One module, icon-package, achieved over 17,000 downloads by typosquatting on the ionicons package before finally being removed.**

WHEN: **The attack was first detected in May, 2022, but dates to December 2021. New, malicious packages are posted periodically to npm.**

WHY: **Clues buried in some of the malicious modules suggest they were designed to steal account credentials from players of PUBG, a popular online-multiplayer video game.**

Such incidents demonstrate the persistent problems and security lapses that characterize development practices at even sophisticated, well-resourced organizations.

In June, ReversingLabs researchers analyzed 3 million releases in over 300,000 different projects hosted on the Python Package Index (PyPi) platform looking for leaks of sensitive information. They detected nearly 30,000 access tokens for Google's API services and more than 25,000 for Amazon's — the most frequently leaked credentials. While in this case many of the detected tokens were only used for internal tests and demos, that wasn't always true for the tens of thousands of other leaked tokens hosted on PyPi.

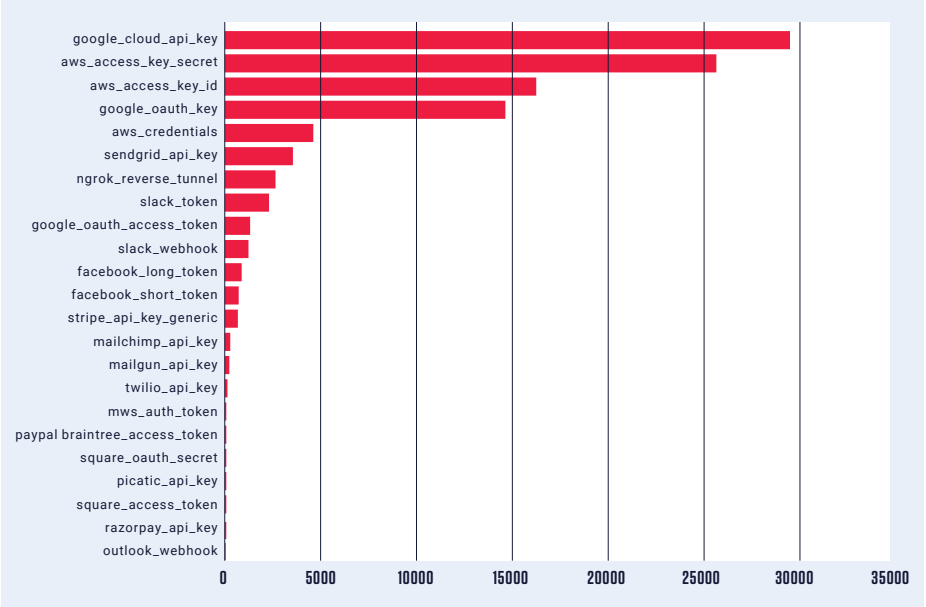


Figure 4. Number of leaked credentials for projects hosted on the PyPi platform. Sensitive data leaked in PyPi packages included credentials for API services for both Google and Amazon. *Source: ReversingLabs*

Even development organizations that are attuned to the risk of secrets leaks can still find themselves exposing sensitive information on public repositories. For example, investigations of supply chain breaches reveals that tools used to prepare packages for publication can inadvertently include files containing secrets, even when those files have been marked as exempt from publishing to a public repository.

"Secrets being unknowingly scattered across enterprise environments is not a new concept. For years, red teams and malicious actors alike have been scanning network file shares for plaintext credentials to assist them in escalating their privileges and facilitating lateral movement," said Charlie Jones, a Security Evangelist at ReversingLabs. "It is only recently that we have seen malicious attackers turning their attention to the software supply chain as they began to recognize source code as an abundant source of unintentionally embedded secrets which can be used to further attacks."

> It is only recently that we have seen malicious attackers turning their attention to the software supply chain as they began to recognize source code as an abundant source of unintentionally embedded secrets which can be used to further attacks.

Charlie Jones
Security Analyst, ReversingLabs

The continued appearance of hard-coded credentials in leaked code suggests that organizations need to do more to enforce basic security hygiene within development organizations. It also suggests that organizations aren't undertaking basic security measures, such as scanning code for credentials, signing keys, and other sensitive information prior to committing it.

Finally, incidents like the Veterans Administration and Toyota leak underscore the security risks of decentralized and outsourced development organizations. It also suggests that the unmanaged migration of proprietary code to open source repositories, often by way of contractor accounts, poses a risk to the security of development organizations and visibility challenges to development and security groups. If nothing else, the long intervals between an initial leak and its detection (months in case of the VA, years in the case of Toyota) suggest that organizations are not actively monitoring for the presence of proprietary code or secrets on open source repositories.

VULNERABLE DEPENDENCIES OPENED DOORS TO ATTACKS

Finally, the emergence of the Log4j vulnerability in 2021, and a wave of subsequent attacks targeting that vulnerable Apache library, underscore the continued risk from vulnerable software dependencies, as well as the difficulty development organizations have policing the problem.

Key technology players are trying to tip the scales. These efforts include:

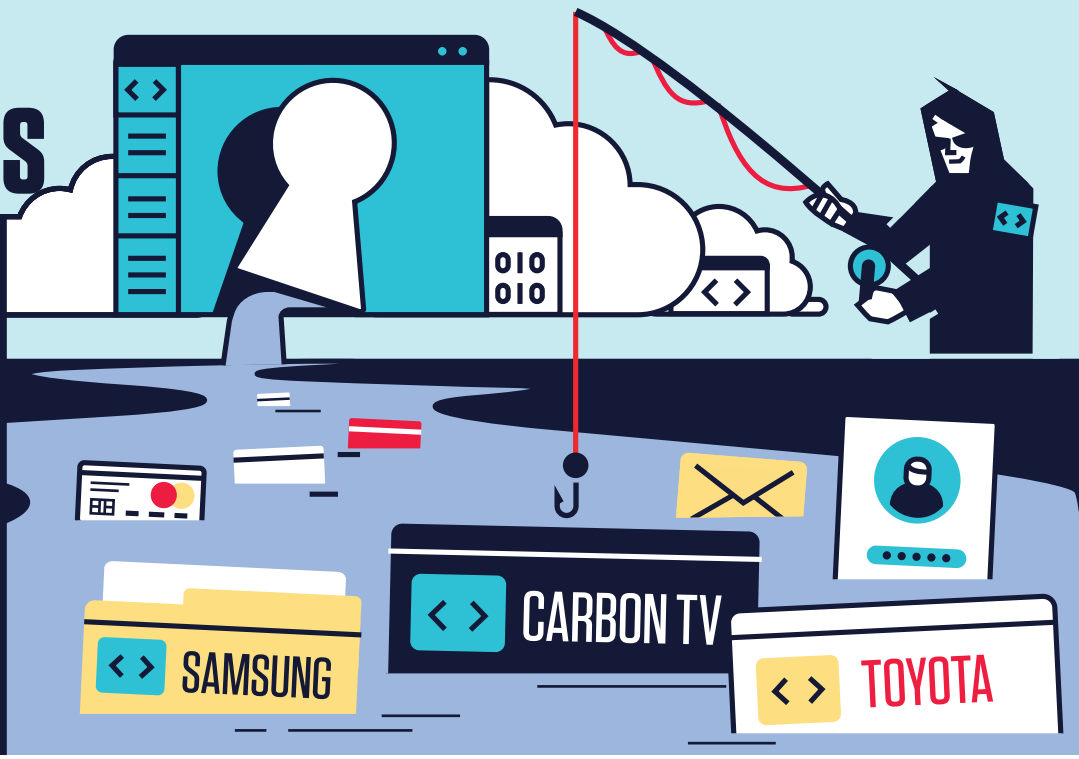
- Google announced its Open Source Software Vulnerability Reward Program (OSS VRP) in August to reward the discovery of vulnerabilities in Google-managed open source projects like Golang, Angular and Fuschia.
- The Open Source Software Foundation (OSSF)'s Alpha-Omega project has donated $1.5 million for "critical security work" to the Rust Foundation, the Python Software Foundation (PSF) and Eclipse Foundation; as well as for Node.js.
- GitHub unveiled Dependabot, a program to keep developers using that platform informed when a vulnerability is discovered in a software dependency on which they rely.

However, the sheer scale of code development taking place in both open source and proprietary ecosystems makes human-led efforts to police code for exploitable software flaws nearly impossible.

GitHub's Advisory Database demonstrates the challenge. Its security team has reviewed and issued advisories for almost 9,300 vulnerabilities in GitHub modules across all languages, but more than 177,000 advisories related to GitHub modules remain unreviewed, many with "critical" ratings. These advisories, which constitute 95% of the total vulnerability count, aren't connected to Github's Dependabot service, so no warning will be issued for them.

A YEAR OF SPILLED SECRETS

Hacking web applications is so "last year." These days, raw source code is a major target for malicious actors, who have discovered that code shared to public repositories often contains sensitive information such as system credentials and access tokens for protected, internal resources. Here are some of the prominent incidents from 2022 in which organizations found sensitive information exposed via their software supply chain:



INCIDENT #1: SAMSUNG, MARCH 2022

- The Lapsus$ hacking group leaked Samsung's source code.
- After scanning it, GitGuardian found 6,695 secrets in the leaked source code.
- GitGuardian's results also showed that just over 600 authentication tokens are exposed in the source code.

INCIDENT #2: CARBON TV, SEPTEMBER 2022

- CarbonTV, a U.S.-based streaming service, left a server containing its source code open to compromise via an insecure API.
- Researchers at Cybernews found that poor access control of the .git folder provided access to secrets contained in the code.
- The leaked source code contained several kinds of secrets, including full access credentials, giving attackers the ability to modify content shown on the streaming service.

INCIDENT #3: TOYOTA, OCTOBER 2022

- Toyota discovered that a portion of its T-Connect web site source code was accidentally published to a GitHub repository belonging to a third-party contractor hired to author the site.
- The leaked source code contained an access key to the data server that stored customer email addresses and phone numbers.
- An unauthorized third-party then accessed the details of close to 300,000 Toyota customers over the course of nearly five years.

Sources: GitGuardian, Cybernews, Security Boulevard

As scrutiny of both open source and common development tools and platforms grows, the security picture for development organizations and their customers is becoming increasingly messy, according to a ReversingLabs analysis of the National Vulnerability Database (NVD).

Vulnerabilities in platforms such as GitLab have created openings for impersonation attacks and account takeovers that hold code repositories hostage. Beyond that, account hijacking subsequent to phishing or other attacks on maintainers has stung prominent firms and resulted in the theft of proprietary code and sensitive data.

PROTESTWARE SUPPLY CHAIN ATTACKS EMERGED AS A NEW THREAT

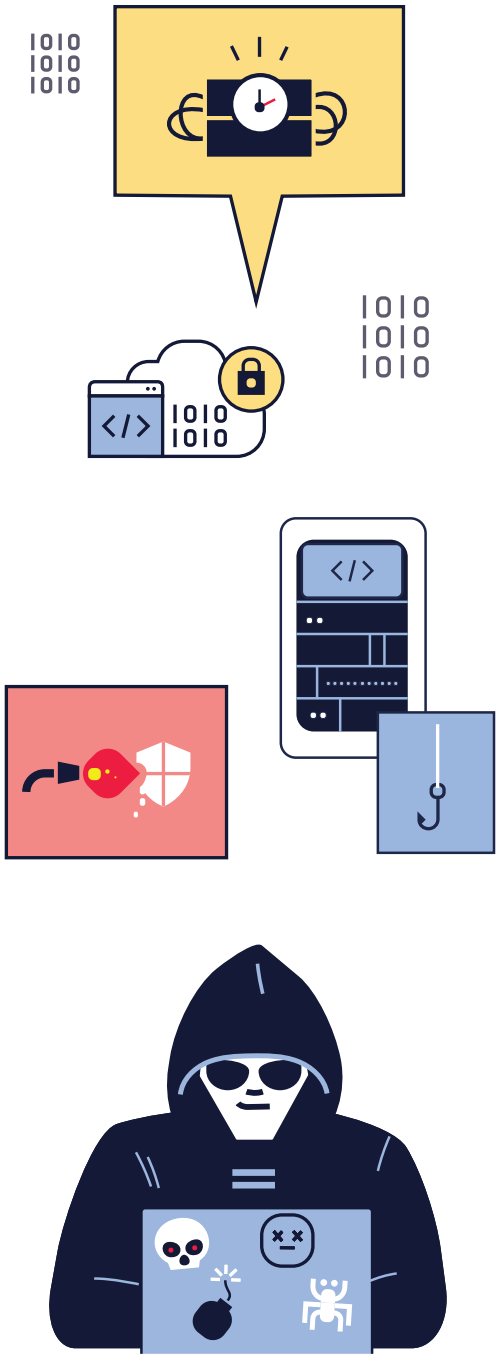
In the last year, manipulation of open source modules has sown chaos among downstream developers and applications. In 2022, so-called "protestware," emerged, in which maintainers of legitimate applications decide to weaponize their software in service of some larger cause (be it personal or political).

In January, for example, downstream applications with dependence on the popular npm libraries 'colors.js' and 'faker.js' found their applications caught in an infinite loop, printing 'LIBERTY 'LIBERTY LIBERTY' followed by a sequence of gibberish non-ASCII characters. The incident was intentional — an act of protest by the maintainer "Squires" for what he perceived as uncompensated use of his libraries by for-profit firms.

Then, in March, Brandon Nozaki Miller, the developer of node.ipc, pushed an update of his popular open source library that sabotaged computers in Russia and Belarus in retaliation for Russia's invasion of Ukraine (and Belarus's support for that invasion). The new release included an obfuscated function that checked the IP address of developers who used the node.ipc module in their projects. IP addresses that geolocated to either Russia or Belarus saw node.ipc wipe files from their machines and replaced them with a heart emoji, according to published reports.

In July, the developer Markus Unterwaditzer temporarily deleted code for his popular and widely used atomicwrites Python library from the popular code registry PyPI in protest over mandated two-factor authentication for maintainers of what are deemed "critical" projects—a requirement that is in no small part due to incidents of maintainers' accounts being hijacked and abused. Unterwaditzer said he found the requirement "annoying" and "entitled."

Such incidents, though rare, make a strong case for the need for increased security and scrutiny of the code hosted on platforms like GitLab, GitHub or npm, that goes beyond research on software vulnerabilities and exposures.



Fears of supply chain attacks are growing

The reality of software supply chain attacks hasn't been lost on developers and those working for software firms.

To assess organizations' levels of awareness about supply chain risks, ReversingLabs commissioned a survey of 307 executives, as well as technology and security professionals at software publishers. The survey, conducted by Dimensional Research, found that concerns about software supply chain attacks and the risks that accompany greater reliance on open source and third-party libraries are growing.

For example, when asked which software security issues posed a risk to their organizations, 63% of respondents said that threats and malware lurking in open source repositories or exploits such as the attacks on SolarWinds and CodeCov were a concern. That's just behind the 66% who said "exploitable software vulnerabilities" posed a risk. Similarly, when asked what is increasing software security risk to their organizations, nearly all (98%) pointed to third-party software, open source software, and software tampering as contributors.

Other supply-chain related issues also ranked high among respondents. More than half (51%) said that the inability to detect software tampering posed a security risk, while 40% cited vulnerabilities in CI/CD toolchains as a concern. But the disconnect between that perception of risk and the embrace of effective supply chain risk management and mitigation strategies was stark. Thorough software audits during and after development were a rarity, with fewer than four in 10 respondents saying their organizations were capable of detecting tampering with developed code, and less than 10% reviewed software at each stage of the production lifecycle for evidence of tampering, leaks or compromises.

Adoption of common software supply chain security tools, such as SBOMs, was also low. A minority of respondents said they used them. Those who did not cited fears about complexity and administrative overhead associated with SBOMs. That helps explain the "why" behind many of the supply chain breaches that make the headlines, but it doesn't bode well for organizations worried about the prospect of further SolarWinds-style attacks.

What's next? Looking ahead to 2023

SUPPLY CHAIN ATTACKS WILL ACCELERATE — AND DIVERSIFY

One unmistakable trend in 2022 has been the increasing cadence of attacks and compromises targeting software supply chains, including attacks targeting both open source and proprietary third-party software. To date, the growth in these attacks has been relatively linear, while incidents of malicious packages have declined on some platforms (such as PyPi).

Still, the rate of growth compared to 2020 remains high. In 2023 the number of malicious packages detected on those platforms will increase again, though security measures taken by platform providers may slow that growth. That, in turn, may push malicious actors to look for other vulnerable links in software supply chains.

PLATFORM OWNERS WILL BOOST THEIR DEFENSES

Recent months have seen development tools and platform providers introduce features to harden their environments and developer accounts from attack. GitHub is a great example. The Microsoft-owned code repository embraced OAuth in 2020, and in 2021 banned the use of passwords for use in authenticating Git operations altogether. In 2022 GitHub continued to refine its security posture, introducing fine-grained personal access tokens for use in scripts to replace more broad access that has proven problematic when tokens have been exposed via public repositories. And PyPi, facing the scourge of account takeovers, in July made two-factor authentication mandatory for use with critical Python projects.

Expect to see a continuation of these hardening efforts in 2022, as both the scope and level of supply chain threats increase. Among the changes to look for are features such as:

- **Malicious package detection.** As supply attacks proliferate, publishers will introduce new features designed to counter known risks. Scanning for anomalous version numbering (such as sudden jumps in version) could prevent dependency confusion attacks. Checking manifest files for significant changes, such as the addition of new pre- or post installation scripts, could also thwart a wide range of supply chain attacks via malicious or altered packages.

- **Integrated third-party package scanners.** Tighter integration with package scanning platforms can prevent malicious code from infiltrating commons platforms like npm, PyPi, RubyGems, Dockerhub and so on. Expect to see more platforms partnering with third-party scanning providers to help keep bad code and images out.

- **IP range locks for package publishing.** These are already popular outside of the development world. (Most people are familiar with the "someone is trying to log into your account from an unknown device" warnings.) They could have the same benefit for development teams by limiting access to repositories to a set of predefined IP addresses from which package updates can be published. That would raise the bar for attackers who try to upload malicious code to package repositories, requiring them to know the allowed IP addresses and then get one of those allowed addresses assigned to them. It might also give development organizations an early warning of attempts to infiltrate CI/CD pipelines.

- **Integrated password-reuse checks.** So-called "layer 8" (aka "humans") is still a big source of risk for development organizations. Features like password re-use checks for accounts would help pick low-hanging fruit like developer accounts that are reusing passwords exposed in prior breaches or are already available on the dark web.

SUPPLY CHAIN SECURITY AUTOMATION WILL TAKE HOLD

As application development has become less centralized and faster paced, responsibility for security has become more challenging. With the embrace of agile "DevOps" development methods, work and responsibility are distributed. It is unlikely that one person or group has a holistic view and understanding of an entire application. At the same time, legacy, manual security review processes aren't suited to fast-paced, modern CI/CD pipeline release cycles. The solution is automation. In the coming year more supply chain security processes will be automated as development organizations increase scrutiny of native, third-party, and open source code while simultaneously increasing the pace of development.

FEDERAL GUIDANCE WILL START TO BITE

Expect an increased focus on the cyber risks lurking in open source code in the next 12 months.

In the public sector, the OMB Memorandum calls out the need for software vendors doing business with the federal government to attest to the security of open source software, for example by using a certified FedRAMP Third Party Assessor Organization (3PAO).

In addition, pending Securing Open Source Software Act of 2022 (PDF) legislation would charge federal CIOs with focusing on open source risk, as well as empower the CISA to produce a framework for handling open source code risk and to perform automated analysis of open source software components used by federal systems (read more about the Act here).

MORE ORGANIZATIONS WILL CREATE OPEN SOURCE PROGRAM OFFICES

In the private sector, growing awareness of the extent of open source risk driven by high-profile flaws like Log4j/Log4Shell and of organizations' deep reliance on open source code is likely to inspire investments in shoring up critical open source projects. But it's an open question as to whether that will spur organizational shifts aimed at reducing exposure to open source risks.

The challenge is that many software supply chain risks, including those centered on open source software, boil down to cultural, rather than technical issues. The embrace of agile development and DevOps methodologies has accelerated development and release cycles, while security is too often still treated as an afterthought.

With those cultural impediments in mind, one development that may begin to get traction, Open Source Program Offices (OSPOs), will be created by more development organizations in the coming year as a way to assess open source exposure and formalize security practices.

As described by the Open Source Security Foundation, an OSPO can help spearhead open source security initiatives, including those related to open source code use, distribution, selection, auditing, and so on. It can also take a lead role in training developers, ensuring legal compliance, and promoting engagement with the larger open source community.

The Open Source Software Act envisions OSPOs for federal agencies, but they're also applicable to private-sector organizations that need a governing body to oversee the use and management of open source software.

Guidance: Four steps to address supply chain security

The growing threat of software supply chain attacks demands new approaches to securing applications and services, both in development and in deployment. Here are four techniques organizations should leverage to combat growing software supply chain risks:

BROADEN YOUR FOCUS TO INCLUDE SUPPLY CHAIN RISKS

Historically, software security has focused on improving the quality of the underlying software, for example by embracing secure software development practices that make it less likely that developers will create common vulnerabilities such as buffer overflows or injection flaws.

In the last decade, organizations across both public and private sectors have embraced secure software development and application testing technologies including static- and dynamic application security testing (SAST and DAST), and software composition analysis (SCA). These tools are invaluable parts of modern, agile software development.

But focusing narrowly on vulnerability management and code quality falls short in the larger context of software supply chain security, which must also encompass growing supply chain threats like malware, malicious insiders and other CI/CD compromises.

"Over 50% of the practices defined within the NIST Secure Software Development Framework (SSDF) focus on the protection, identification, and remediation of vulnerabilities within software," says ReversingLabs' Charlie Jones.

"Although the detection and remediation of vulnerabilities is critical to uplifting the security posture of software, the presence of vulnerabilities within a software package does not necessarily indicate that the package has been compromised, and presents an immediate threat to an acquiring organization," Jones said.

As the attacks and supply chain incidents of the last 12 months reveal, organizations involved in software creation need ways to identify vulnerabilities. But given the scale of vulnerabilities being reported, they also need an easy way to triage and prioritize the vulnerabilities that matter most: remotely exploitable flaws affecting critical systems. That's the idea behind CISA's Vulnerability Exploitability eXchange (VEX) security advisory mechanism.

Beyond finding and fixing vulnerabilities, however, development organizations need to stay attuned to other exposures (the "E" in "CVE"). One example: unauthorized code changes to software that introduce potentially malicious behaviors such as being able to change account privileges or reboot the system. Such changes often indicate compromises and precede attacks on production systems.

To address these risks, expand your organization's threat detection capabilities to encompass software tampering, malicious or compromised dependencies and other CI/CD compromises that may result in the introduction of malicious code or behaviors. Binary scanning and other behavioral detections that focus on compiled code will help prevent more SolarWinds-style compromises.

SHIFT LEFT TOGETHER: FOSTER DEV AND SOC COORDINATION

In the wake of the volume of supply chain attacks that occurred in 2022 it's clear that threats to development organizations and pipelines are growing, and the response to those threats can't be confined to development teams.

After all, a cursory review of supply chain attacks, from SolarWinds to more recent campaigns leveraging malicious PyPi modules, shows that attacks on development supply chains and pipelines are merely a means to an end for malicious actors. And the end in question is usually indistinguishable from non-supply chain attacks: lateral movement, data theft, and the placement of malware for persistence or extortion (e.g. ransomware).

That's why ReversingLabs researchers expect to see greater coordination between development and security operations teams to share intelligence about supply chain threats in 2023.

The advent of dependency confusion and typosquatting attacks exposes a weakness in the DevSecOps paradigm: the inherent trust placed in the integrity of third-party and open-source software supply chains by individual developers and larger development organizations.

Already there is evidence that such attacks are in the toolbelt of offensive actors. To ensure that such initial forays don't escape the notice, security operations centers need to follow attackers as they shift left, broadening their mandate to encompass monitoring of software supply chain threats as part of their overall risk monitoring.

How to accomplish that shift is a matter for debate. J. Paul Reed, a DevOps expert and specialist on Netflix's Critical Operations and Reliability Engineering (CORE) team, advocates bringing release engineers and security engineers together to coordinate their activities.

"A lot of the concerns that build and release engineers have are similar to the ones that security engineers have. The problems that keep the security engineer up at night are the same ones that keep the release engineer up at night," he says.

For example, both security engineers and release engineers have the same interest in ensuring the provenance of open source and third-party software components, keeping the software components updated and vetting them for vulnerabilities and other issues that could affect application confidentiality, integrity, and availability.

With insights into supply chain threats and attacks, security engineers are already becoming de facto release engineers, Reed notes. Besides, closer coordination earlier in the design and development stages can avoid last-minute holdups imposed by security teams — the kind of "scan and scold" dynamic that can antagonize development organizations.

HOME IN ON OPEN SOURCE RISKS

Threats and attacks that come by way of open source software are not new. As far back as 2003, an unknown hacker added a backdoor to the Linux kernel, while "prototype pollution," where attackers introduce malicious code to otherwise trustworthy software artifacts, has become more common. One study found that prototype pollution was detectable in more than 25% of all open source projects the authors reviewed. More recently, the North Korean state-sponsored hacking group known as Lazarus introduced Trojan horse code into open source software, including apps like PuTTY, KiTTY, TightVNC, Sumatra PDF Reader for use in targeted attacks aimed at cyberespionage, according to Microsoft.

In 2023, development organizations need to make open source risk a priority. That means developing a detailed understanding of your organization's use and dependence on open source modules, and closely monitoring those open source modules for evidence of tampering or compromise.

INVEST IN PROACTIVE THREAT HUNTING

Guidance from the federal government is encouraging development organizations to think more broadly about their software supply chain risk and to expand the scope of their defensive measures beyond vulnerability discovery and patching. That's a natural response to the current threat landscape, where the breadth of attack techniques that the supply chain is exposed to spans the entire software development, packing, delivery, and use lifecycle.

Investing in technologies that enable proactive threat hunting within the software supply chain will be increasingly important for identifying sophisticated supply chain attacks before they can materialize.

Development organizations need to invest in tools that are capable of identifying malicious components that could be hiding in open source and third party software packages. Threat hunting technologies such as binary analysis, file reputation and threat intelligence services can help identify those risks, while applying open-source YARA rules internally can help organizations detect malicious software components such as malware downloaders, viruses, trojans, exploits and ransomware.

Conclusion: To detect software supply chain tampering, look beyond the code

If data from the past three years is any indication, attacks on software supply chains will increase in both frequency and severity in 2023, as they have in each of the last three years. That, along with new regulations and guidance intended to address supply chain risk, will put new pressure on development organizations and enterprises.

Going forward, ReversingLabs researchers anticipate a shift in both security thinking and investment. With the Log4Shell exploit barely a year old, organizations need to invest the resources and time to assess their exposure to risk from open source components, with an eye to spotting targeted dependency confusion attacks and avoiding more passive typosquatting scams that look to push malicious code into sensitive organizations.

Following a spate of incidents in 2022, expect to see increased scrutiny of both internal and shared code for evidence of secrets such as access credentials for cloud-based services like AWS and Azure; SSH, SSL and PGP keys, and assorted other access tokens and API keys.

Finally, organizations should look beyond mere code vulnerabilities and secrets to assess the risk posed by software tampering. Binary scanning and behavioral analysis of compiled binaries will need to become a standard part of security reviews and quality assurance. All of this will require substantial changes in how development teams approach software security, as well as the acquisition of new tools and talent capable of carrying out fine-grained security assessments at each stage in the development pipeline. As the stakes and cost of failure increases, however, these investments will more than pay for themselves over time.

Additional Resources:

Why Malware Detection Isn't Enough Protection Against Software Supply Chain Attacks

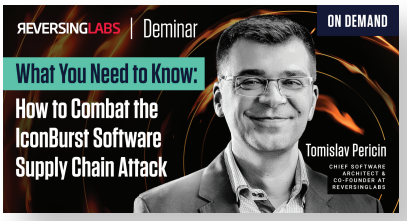
[Download Solution Brief]

Flying Blind: Software Firms Struggle To Detect Supply Chain Hacks

[Download Report]

What You Need to Know: How to Combat the IconBurst Software Supply Chain Attack

[Watch Deminar]



Not all SBOMs Are the Same. Choose Wisely!

[Read Blog]

ReversingLabs supports many languages and repository packages to deliver software supply chain protection for CI/CD workflows, containers and release packages.

Learn how ReversingLabs secure.software can help you perform critical security checks before you deploy software

LEARN MORE

Worldwide Sale:
+1.617.250.7518
sales@reversinglabs.com

REVERSINGLABS