Red Hat

Red Hat Product Security Risk Report 2024

Table of contents

facebook.com/redhatinc
@RedHat
linkedin.com/company/red-hat

Introduction

The rate of change we see in product security, open source communities, and the industry at large has never moved more expeditiously. This year's risk report will include the usual statistics and insights from Red Hat's Product Security Incident Response Team (PSIRT), a section on artificial intelligence (AI), highlights from other product security programs, and a look at the threat landscape, written in collaboration with Red Hat's Cyber Threat Intelligence team.

Unless you have been living under a rock for the last year, chances are AI has become an increasing part of your work and home life. The capabilities of the latest frontier models are now impossible to ignore. As businesses rush to adopt these new capabilities, many questions around safety and security remain open. We find ourselves once more looking to open source and community collaboration to work together in laying the foundations of trust for these developments.

Of course, AI was not the only thing happening in 2024. In February, we saw the realization of a threat that has long troubled many security-minded people working in open source, with the discovery of a fairly advanced backdoor planted in XZ. This issue has its own section in this year's report, also written in collaboration with Red Hat's Cyber Threat Intelligence team.

This may be the most content we have ever packed into a single risk report. As always, the information provided here will serve our customers and the larger community with valuable insight into our work. Enjoy.

- Garth Mollett, Product Security Lead Architect, Red Hat

Vulnerability response overview

Red Hat Security Advisory

Red Hat® Security Advisories (RHSA) continue to be the primary notification mechanism for releasing software updates containing fixes for known vulnerabilities in the Common Vulnerability and Exposure (CVE) catalog since their inception more than 20 years ago. While the format, delivery mechanism, and other details have developed over the years, the primary purpose remains.

The trendline from 2020-2024 once again shows an upward trend. There is an almost linear upward slope from 2022, reaching a new peak of 2975 security advisories in 2024.

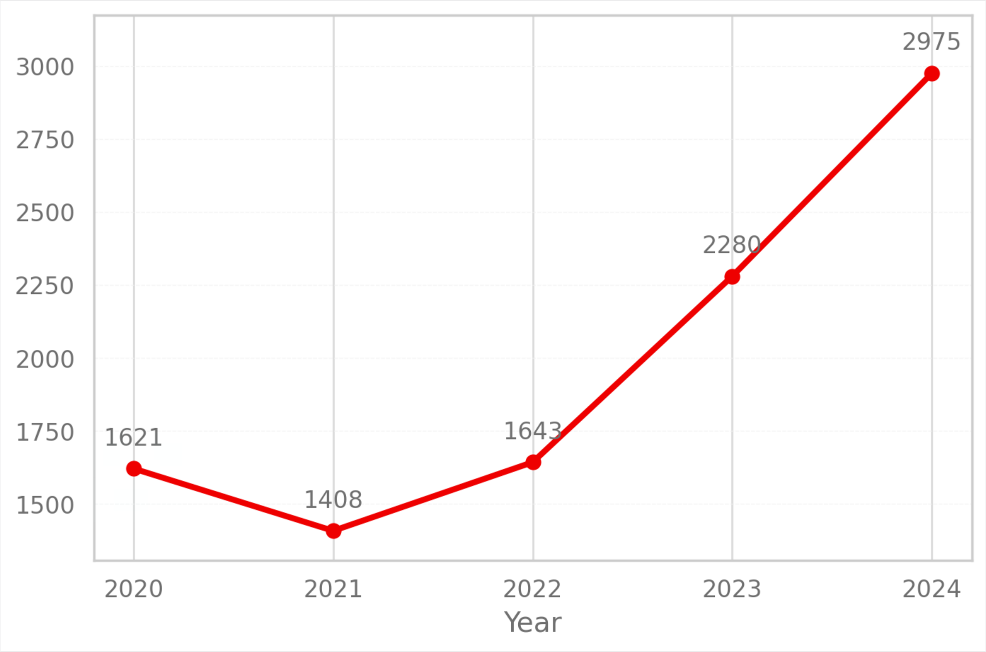


Figure 1. Red Hat Security Advisory timeline

Table 1. Red Hat Security Advisory by severity

Total	2975	+695 from 2023
Critical	55	-10 from 2023
Important	1649	+244 from 2023
Moderate	1178	+418 from 2023
Low	93	+43 from 2023

Table 2. Red Hat Security Advisory by product family (does not include all products)

Product family	Critical	Important	Moderate and Low
Red Hat Enterprise Linux®	9	1210	893
Red Hat OpenShift®	26	238	180
Red Hat Ansible® Automation Platform	0	3	18
Red Hat OpenStack® Platform	4	22	34

Product family	Critical	Important	Moderate and Low
Middleware products	5	86	51
Red Hat Satellite	1	8	9

The numbers continue to climb across the board, as they have done for many years. This is more of a reflection on the growing size and complexity of the software portfolio than any alarming trend of increased software vulnerability. However, we see a slight drop in Critical errata from last year, which we will also see reflected in the CVE data below, where the Critical CVE count was down by 2 from 2023.

Common Vulnerabilities and Exposures (CVE)

If we analyze the CVE distribution by quarter, we find that Lows and especially Moderates make up the bulk of the reports, which is usually the case.

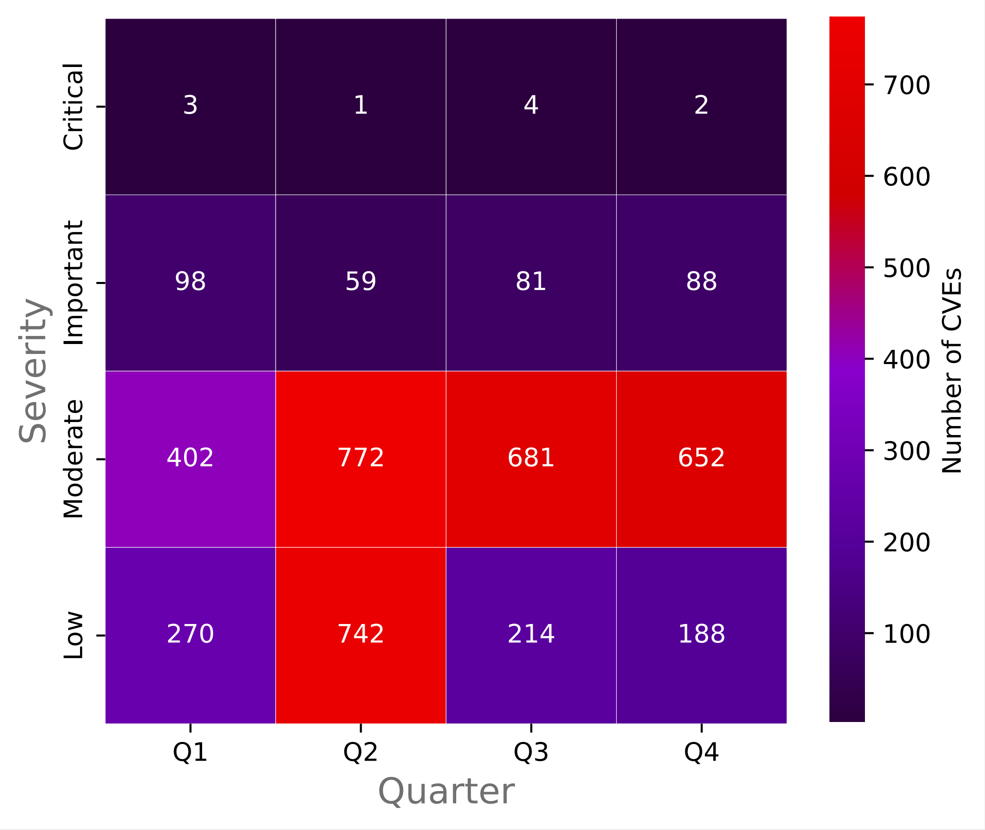


Figure 2. CVE reports by quarter

This is where things become particularly interesting. While the CVE count from 2020-2023 has been relatively flat, we noticed that according to the 5-year trend, there was a very sharp uptick for 2024.

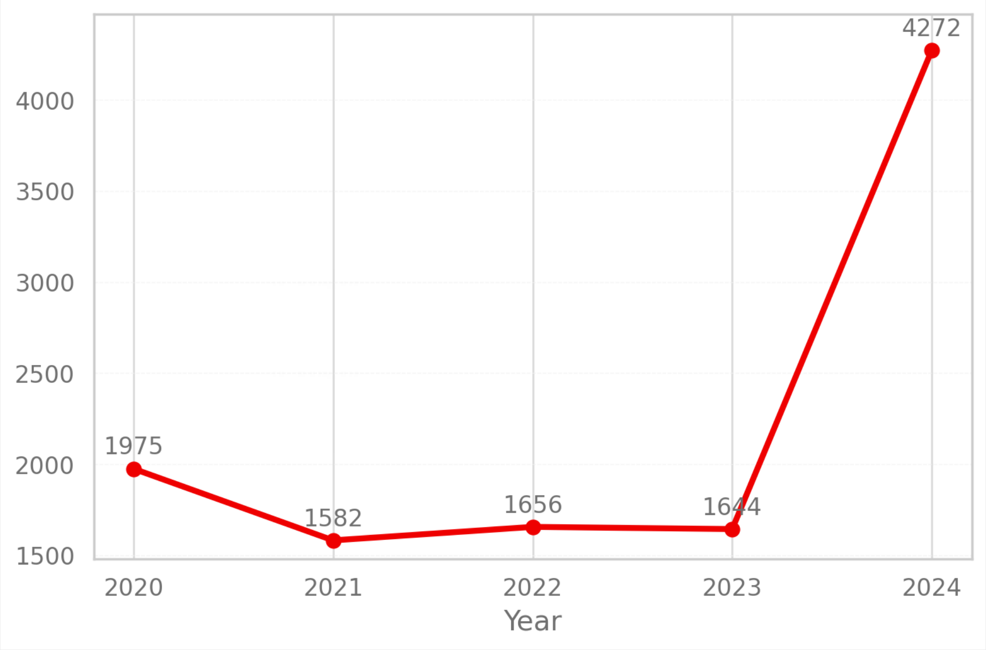


Figure 3. Red Hat CVE timeline

So, what are we seeing here? What caused the gigantic increase in CVEs? The simple answer is that the Linux kernel (kernel.org) became a CNA in February 2024 and started assigning a ton of CVEs to Linux kernel issues that would have been unlikely to get CVEs in the past. We will dig into this in a bit more detail later on, but first let's look at the CVE trend line without the kernel.org CVEs.

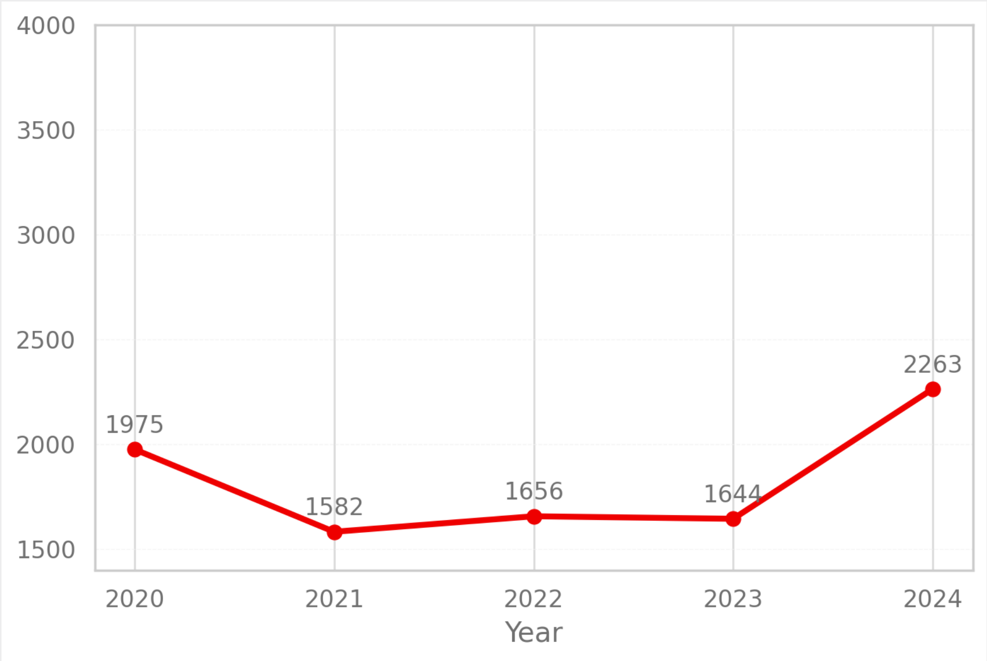


Figure 4. Red Hat CVE timeline - No kernel for 2024

That looks more like what we would expect. There is still a minor uptick, but nothing alarming, and it is in line with expectations for an expanding software portfolio. However, neither of these graphs really show the whole story. Let us dig a little deeper and compare 2023 and 2024, showing each severity divided into kernel and non-kernel CVEs.

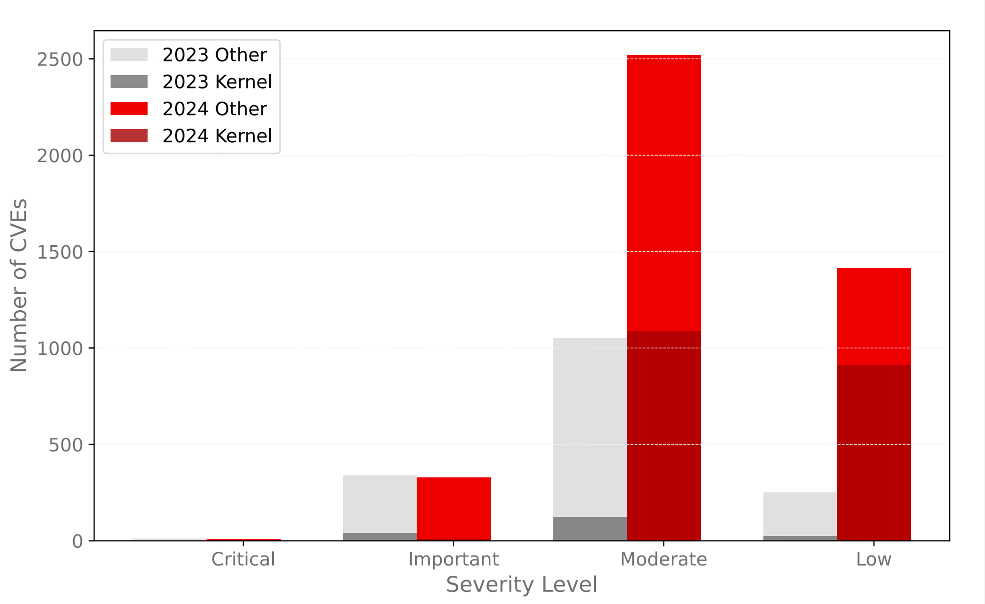


Figure 5. Kernel vs non-kernel CVE 2023/2024

Well, that is a little bit more interesting. With Critical and Important, we see almost no change, but for Moderates, nearly half the CVEs are from the kernel. For Lows, we see it is more than half.

What does this mean? For one, it means the vast majority of these bugs do not pose a risk of effective exploitation, at least in the Red Hat or Red Hat Enterprise Linux use cases. Secondly, without delving too deeply into the politics and heated debates surrounding this situation, this highlights the difficulty that kernel.org has been trying to articulate about the difficulty upstream has with classification of security issues in a kernel with so many broad use cases. There is room for improvement here, and now the community has the opportunity to step up and help improve the classification and data around these bugs.

Let us look at one more data point regarding the Linux kernel. In the figure below, we compare the total CVEs assigned by kernel.org to the number of CVEs that affected the kernels in Red Hat Enterprise Linux.

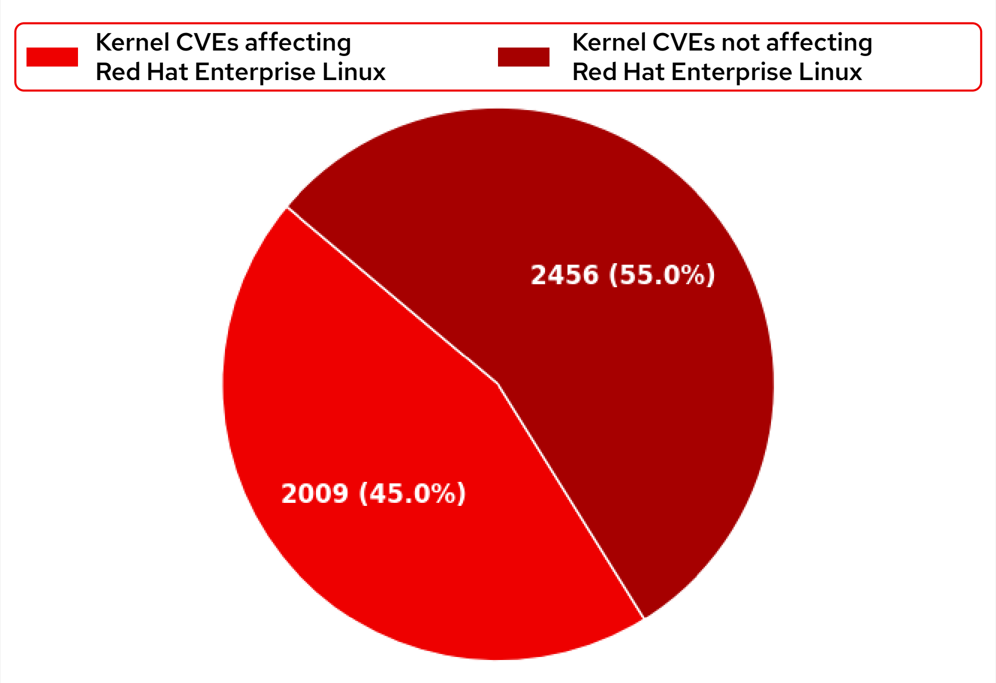


Figure 6. CVEs assigned by kernel.org in 2024

The 2009 CVEs we see in the CVE data above are only a subset (45%) of the 4465 CVEs assigned by kernel.org in 2024, the remaining 2456 CVEs did not affect the current Red Hat Enterprise Linux kernels for various reasons.

Exploitation in the wild

While we saw an increase in the amount of reported CVEs, even without the influx of kernel.org CVE's, there is a decrease of reports of exploitation in the wild in every severity rating other than Critical, which remains stable at 1.

Table 3. Red Hat Security Advisory by severity

Severity rating	Flaw count (+/- from 2023)	In the wild exploitation (% of total | +/- from 2023)
All	4272 (+2628)	11 (0.3% | -9)
Critical	10 (-2)	1 (10% | +0)
Important	328 (-8)	10 (3% | -5)
Moderate	2520 (+1473)	0 (0% | -3)
Low	1414 (+1167)	0 (0% | -1)

Important once again remains the severity level where the bulk of exploitation is occurring. This is absolutely expected and further reinforces that the severity ratings make sense.

While it is nice to see low numbers, it is important to understand real-world attacks remain private and not shared publicly. Many attacks go undetected and when they are detected, the kind of detailed analysis and insight needed is usually beyond the reach of many smaller entities. The significant difference between Low and Moderate CVEs compared to Important and Critical CVEs reflects the reality of the effectiveness of exploitability and usefulness in real-world attacks. The true numbers are probably higher, providing an opportunity for more research.

Response times

Response time, the speed at which a fix can be released, continues to be a focus for Red Hat Product Security and our customers. The biggest threat from vulnerabilities lies in the time between the details of a public disclosure about a flaw and when patches are applied.

For patches to be applied, they must first be developed, tested, and released. When Red Hat is part of the coordinated disclosure process, this begins before a vulnerability is made public. In all cases, significant testing is done by Red Hat before releases to improve confidence and minimize testing cycles required by customers.

Table 4. Response times by severity

Severity	2023	2024	Change
Critical	9 days	7 days	22% faster
Important	48 days	31 days	35% faster
Moderate	112 days	89 days	21% faster
Low	130 days	115 days	12% faster

As with previous years' reports, these numbers are point-in-time readings and the reality is much more nuanced as numbers fluctuate throughout the year depending on complexity of vulnerabilities, fixes, and the amount of components affected. These times are based on the vector between a CVE becoming public and the first fix being released. It is nice to see that efforts to deliver fixes faster appear to be working, and the numbers are trending down across all severities.

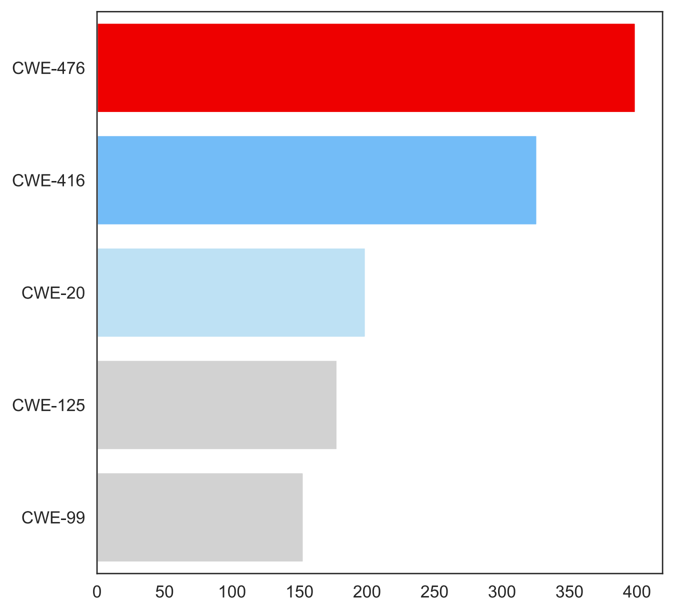
It is also worth noting that over half of the Critical CVEs had first fixes available in 7 days or less and 3 were fixed on Day 0.

Common Weakness Enumeration

While these statistics provide a good view of the volume of work required in addressing vulnerabilities, a more interesting data point is the type of vulnerabilities being addressed.

Common Weakness Enumerations (CWEs) allow us to categorize vulnerabilities, but the categorization is fairly basic. Modern exploit and malware development is increasingly complex, and more often than not, involves combining multiple bugs to provide chained exploitation primitives instead of using a single bug.

While CWE does not capture this complexity, it provides an interesting data point that gives some insight into where hardening efforts can be most effective in software development.



CWE-476

NULL Pointer dereference

CWE-416

Use after free

CWE-20

Improper input validation

CWE-125

Out of bounds read

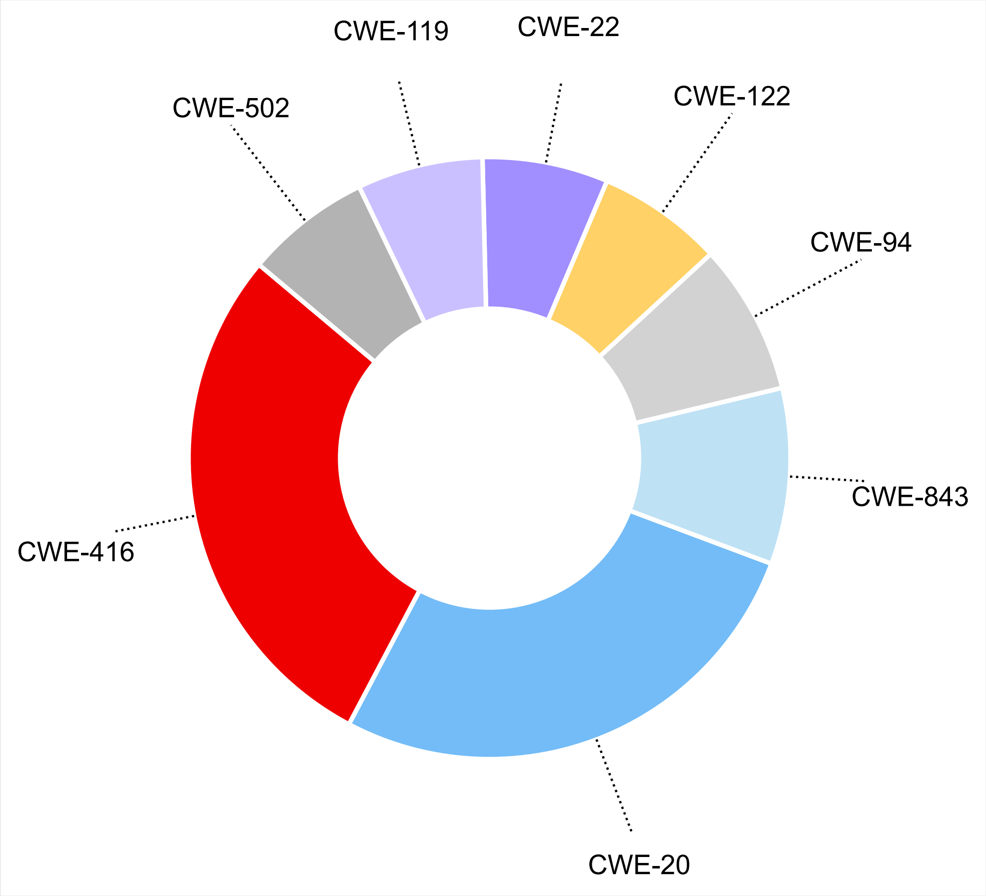
CWE-99

Improper control of resource identifiers

Figure 7. Top 5 CWE for 2024

CWE-20 has some heavy lifting here as it covers a broad area of issues. However, CWE-476, CWE-416, and CWE-125 are all classic memory safety issues and of very little surprise to anyone paying attention, the top culprit of all 3 is the Linux kernel.

Taking a long-term view, going back to when we first started collecting this data but narrowing it down to flaws known to be exploited in the wild with 5 more occurrences of a specific CWE, we observe the following.



CWE-416 Use After Free

CWE-20 Improper Input Validation

CWE-843 Access of Resource Using Incompatible Type ('Type Confusion')

CWE-94 Improper Control of Generation of Code ('Code Injection')

CWE-122 Heap-based Buffer Overflow

CWE-22 Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')

CWE-119 Improper Restriction of Operations within the Bounds of a Memory Buffer

CWE-502 Deserialization of Untrusted Data

Figure 8. CWEs Exploited in the wild

We see CWE-416 jump to first place. There are a few things at play here. We see the tension between performance and security hardening. Finding the right balance in memory allocation routines where errors are hard to exploit but still remain highly performant, is not simple. We also know that because of the complexities in implementing these routines, bugs that fall into this class can often provide multiple useful primitives for exploitation.

CWE-843 (type confusion) is also likely getting some mileage here as web browsers and JavaScript engines have become an increasingly common initial attack vector. These bugs also tend to provide some good opportunities—from an attacker's point of view.

Select major incidents

Red Hat Product Security uses many factors to determine if an event or a vulnerability should be treated as an incident. The primary factors are the risk to customers and the Red Hat brand, with the highest priority being an easily exploitable vulnerability in significant, mission-critical software. Other types of risk, such as misinformation and panic from confusing or exaggerated media coverage, are also considered.

The purpose of the incident process is to prioritize fixes that would happen regardless of whether an incident occurs or not, provide for clear and calm coordination, and keep our internal and customer-facing communication channels open. On the customer-facing side, we issue additional artifacts including Red Hat Security Bulletin (RHSB), Red Hat Insights detection rules, Ansible remediation playbooks, the usual security errata, and CVE page entries.

This section details the 2 major incidents from 2024 where Red Hat Security Bulletins were released.

RHSB-2024-001 Leaky Vessels - runc - (CVE-2024-21626)

The basics: A file descriptor leak in runc was discovered that might have led to a container escape to the underlying host operating system (OS), either by building an image using a malicious Containerfile or running a malicious image. Rory McNamara, a researcher at Snyk, found and responsibly disclosed this issue in runc and a few more related vulnerabilities in Moby BuildKit. During the disclosure, additional disclosure participants identified 3 other attacks that took advantage of CVE-2024-21626.[1]

The details: This vulnerability was rooted in how runc processed the 'WORKDIR' directive in a Dockerfile. When specifying the initial working directory for processes created during the 'build' or 'run' operations, runc changed the directory using 'chdir' before closing certain privileged host directory file descriptors. This action allowed an attacker to specify those privileged file descriptors using the 'WORKDIR' directive, such as a directory within '/proc/self/fd/'. Consequently, even after 'runc' closes the file descriptor during normal operations, it remains accessible, facilitating unauthorized access to sensitive host files and allowing the creation of arbitrary files within the host filesystem.

The statistics

Affected major product versions: Red Hat OpenShift 4, Red Hat OpenShift 3.11, Red Hat Enterprise Linux 7-9

Severity rating: Important

[1] *Github credits. "Several container breakouts due to internally leaked fds." 31 Jan. 2024.*

Embargo time: 14 Days

Time from public to first fix release: 2 Days

Time from public to all fixes released: 8 Days

Exploit code published: Yes, proofs of concepts (POCs) were available publicly in under a day.

Exploitation in the wild: No

Closing thoughts: Red Hat Enterprise Linux and Red Hat OpenShift ship with SELinux in targeted enforcing mode, which prevents the container processes from accessing host content and mitigates this attack, so this vulnerability stood out because it evoked a sense of familiarity. However, it was not because of the vulnerability or even the multiple attack scenarios discovered and shared during the embargo by coordinating participants. This event caused us to recall another container escape that was prevented thanks to SELinux.

RHSB-2024-002 - OpenPrinting cups-filters CVE-2024-47076, CVE-2024-47175, CVE-2024-47176, and CVE-2024-47177

The basics: This incident was complicated. CUPS is an open source printing system that provides tools to manage, discover, and share printers. The group of vulnerabilities identified in OpenPrinting CUPS affected all Red Hat Enterprise Linux versions with a severity impact of Important. As shipped in Red Hat Enterprise Linux, the default configuration of the service was vulnerable. However, this service was installed in a disabled state. The affected components were not installed with the vulnerable service enabled.

The researcher reported discovering this chain of vulnerabilities and posted it as an embargoed GitHub Security Advisory. Because CVE IDs had not been assigned, Red Hat was unaware of the issues. By the time we were informed, the media had already picked up on it, and the embargo was breached a few days later.

The details: The circumstances required to exploit these vulnerabilities successfully required certain conditions to be true. An attacker had to advertise a malicious Internet Printing Protocol (IPP) service accessible by a victim. The attacker-controlled service could be on the public internet or within an internal trusted network. Advertising on an internal trusted network would require a successful breach of the network by being local on another server or having the ability to be local with a malicious system, such as a laptop.

For a successful attack, the victim must have the cups-browsed service running, which scans for available printers. This allows an attacker to automatically add a temporary printer definition from a malicious IPP server. At this point, the malicious IPP server could send arbitrary code back to the victim as a part of the printer definition, which executes as the unprivileged lp user when triggered. The victim must attempt to print from the malicious device to execute the provided code.

Exploiting these vulnerabilities could lead to remote code execution as the unprivileged lp user.

The statistics

Affected major product versions: Red Hat Enterprise Linux 7-9

Severity rating: Important

Embargo time: 3 days

Time from public to first fix release: 1 day

Time from public to all fixes released: 7 days (excluding non-Important)

Exploit code published: Yes, before the embargo was lifted

Exploitation in the wild: No reports

Closing thoughts: An embargo's purpose is to protect users during the sensitive timeframe from when attackers have access to actionable information to when defenders can start protecting themselves. From the time the information was initially leaked, attackers could have pursued the narrow set of attack surfaces to try and duplicate the discovery to use the vulnerability against victims before protections are available.

Within hours of the few details of the CUPS CVEs being disclosed on social media, Red Hat was flooded with customer inquiries and escalations. While the existence of a vulnerability does not equate to risk, the lack of guidance or details behind the CVEs heightened concerns amongst Red Hat customers, partners, and the open source community, including intelligence and government entities. Because information about the vulnerabilities was unavailable, organizations began preparing for the worst.

Maintaining an embargo is hard and oftentimes stressful, especially in situations where coordination is required between stakeholders across the industry. While we do not know who or what caused the embargo to breach, we attribute it to the frustration and exhaustion that comes with this type of coordination.

Red Hat Product Security is well-versed in the effort it takes to report, resolve, and disclose security vulnerabilities and we are committed to providing assistance and support to reporters and open source communities in these situations. Red Hat acknowledges that security is a shared responsibility, and we are prepared to support those who need assistance.

A special thank you to Simone "EvilSocket" Margaritelli for identifying and reporting these vulnerabilities.

Reflecting on the XZ backdoor

Overview

On March 29 2024, Andres Freund emailed the open source security mailing list describing a backdoor in the XZ project code, leading to one of the most notable open source security incidents in recent years. If it had gone undetected, this backdoor could have allowed unauthorized access to Secure Shell (SSH) software, which is used for remote access to the majority of cloud and server systems on the internet.

The backdoor was introduced by an established maintainer of the XZ open source project who had worked more than 2 years to gain the trust of the project's founder. The identity of the malicious actor or group behind this attack remains unknown.

The backdoor employed a carefully concealed method to embed itself into the XZ utilities library (liblzma) during the build process. It was designed to avoid detection, even with inspection of the build process and artifacts. It contained a specific logic that included the malicious code only when integrated into a software package build process for Linux systems.

When executed as part of the SSH daemon process, the backdoor used a sophisticated programmatic feature specifically introduced by the threat actor to hook into the legitimate libraries of the SSH process. The code attempted to hide and obfuscate itself to avoid detection. This obfuscation also caused a delay in the startup time of SSH, ultimately leading to the discovery of the backdoor.

Open source response

One might have anticipated a large-scale supply chain incident exploiting open source software's open and inclusive nature. Still, no one could have predicted something this meticulously planned and audacious. The Log4j incidents of 2021 were similar in the sense that it was a small but critical project that was quietly maintained by a dedicated but overworked founder and became a widespread point of potential vulnerability.

An advanced persistent threat (APT) actor has repeatedly demonstrated that supply chain attacks, in both open and closed source software, are highly effective and difficult to defend against. Therefore, future supply chain attacks are inevitable. Epictetus believed that human response is more important than the associated event.[2] This is helpful to consider in the context of security incidents, with the open source response to the XZ backdoor demonstrating another advantage of the open source approach.

Linux distributions had a 2-day embargo period to assess their exposure and communicate a clear message to their customers. Upon receiving the public notification, the open source community swiftly gathered a wealth of information about the backdoor and published a technical analysis of the build processes and reverse engineering of the backdoor itself.

This analysis was facilitated using publicly available information: the malicious release, the project source repository, and the mailing list interactions of the threat actor (or actors) with the project leader. This public source data allowed review and verification, causing an in-depth and accurate analysis of the incident being available within a few days of public disclosure.

This crowdsourcing provided a clear picture of the attacks compared to the details of incidents affecting closed source software vendors, which resulted in a timely disclosure and resolution to the public.

Long term effect on open source

The XZ backdoor has and will continue to have profound and long lasting effects on open source projects and communities. Using the collaborative and inclusive nature of open source development for ill has already made open source projects less trusting of new contributors.

The incident blunts some of the efforts that projects have made over recent years, such as codes of conduct, designed to make projects and communities more accessible and welcoming for new contributors. All of this will also add more cognitive load for maintainers and reviewers who will now have to consider the potential adversarial nature of contributions.

While this is a setback, open source has proven its ability to evolve and adapt to change. The XZ incident has already inspired tactical and procedural changes to make such an attack more difficult. Future changes will evolve through trial and error, and there will likely be missteps and overcorrections, but these measures will ultimately make such attacks more difficult in the future.

[2] *Wikipedia. "Epicetus." accessed 28 March 2025.*

Part of Red Hat's success stems from not trying to control open source projects, but understanding and engaging positively with open source communities. This is fundamental in order "*To be the catalyst in communities of customers, contributors, and partners creating better technology the open source way.*"[3]

The value proposition of Red Hat

Could the XZ backdoor have been shipped as part of Red Hat Enterprise Linux? This is a complex hypothetical question without a conclusive answer. However, it's true that there were many barriers of increasing difficulty that could have prevented the compromise of Red Hat customer environments from the backdoor.

The threat actors responsible for this attack have an advanced knowledge of C software development, Linux build processes, and command line utilities. They also have an excellent understanding of how open source projects operate, including how upstream code flows downstream to distributions. However, distributions and package build systems are complex, which ended in the threat actors making some mistakes. These mistakes provided some red flags along the way. Some of these red flags were overlooked and not immediately noticed, but several were, leading to the detection of the malicious code .

The Fedora® distribution remains an integral part of Red Hat's product pipeline. Fedora is an innovative platform for hardware, clouds, and containers. It brings together a rich collection of new upstream components while providing stability, leading to a large user base of Fedora, even internally at Red Hat. A backdoor would need to be promoted throughout the gated release cycle of Fedora before it runs on most Fedora user systems. The large and diverse uses of Fedora would significantly increase the likelihood of abnormal behavior being observed, investigated, and discovered.

Assuming it remained undetected in the stable release of Fedora, which would still be a catastrophic security incident, the malicious package would need to be included in Red Hat Enterprise Linux. This inclusion typically occurs during major releases in newer software versions. The security-focused and well-resourced quality engineering in Red Hat Enterprise Linux, along with additional automated build checks, would likely have flagged this backdoor case.

Finally, if the XZ backdoor was shipped with Red Hat Enterprise Linux, then it would have to be used extremely sparingly by the threat actors. Failing to do so, or any other operational security missteps would result in it being discovered in the incident response phases, as was the case for the Solarwinds supply chain incident in 2020.

Threat landscape

Software supply chain attack (SSCA) events in 2024

According to Black Duck's 2024 Open Source Security and Risk Analysis (OSSRA) report, 97% of commercial codebases across 17 industries include applications or services containing open source components.[4] This reliance on the open source software ecosystem makes an attractive target for threat actors.

[3] *Red Hat. "Book of Red Hat." accessed 28 March 2025.*

[4] *Black Duck report. "Six takeaways from the 2025 Open Source Security and Risk Analysis report." 25 Feb. 2025.*

Red Hat data and some industry reports, including Sonatype's 10th Annual State of the Software Supply Chain Report, indicate a substantial increase in attacks detected in the software supply chain in 2024.[5] Red Hat collected information from 89 publicly reported software supply chain attacks (SSCAs) in 2024, a 68% year-on-year increase.

The following 5 events, or 6% of SSCAs, are successful adversary operations with a high impact on multiple downstream consumers:

- Fake developer jobs laced with malware
- Tornado cash theft uncovered: Malicious code drains funds for months
- Red Hat warns of backdoor in XZ tools used by most Linux distros
- Mitigating the risk of software supply chain attacks: Insights from the dropbox sign breach
- BeyondTrust says hackers breached remote support SaaS instances

Reporting on this type of attack fell from 17% in 2023. We selected 2 key events to discuss impact and mitigation.

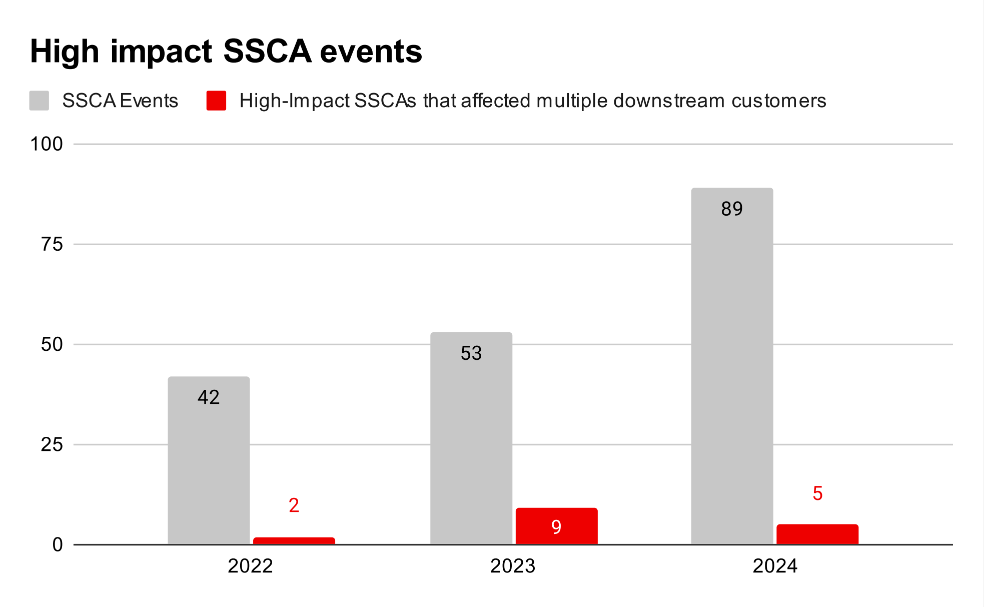


Figure 9. Comparison of reported software supply chain attacks and high-impact events between 2022 and 2024

[5] Sonatype. *"Sonatype's 10th Annual State of the Software Supply Chain Report Reveals 156% Surge in Open Source Malware." 10 Oct. 2024.*

XZ Util compromise

This event was one of the highest profile incidents in 2024 involving a significant infiltration of a critical open source project. The incident affected several Linux distributions, including Fedora. At the time of discovery, the impact was primarily limited to the Fedora Linux 40 pre-release and Fedora Rawhide, the development distribution of Fedora Linux, which serves as the basis for future Fedora Linux builds.

Impact: On 29 March 2024, the collaboration moved swiftly. Our teams delivered the first YARA rules, evaluated the Vulnerability Management aspects, and verified if our productization pipeline was using affected versions. As a CVE Numbering Authority (CNA), we assigned this vulnerability CVE-2024-3094 and published the blog, "Understanding Red Hat's response to the XZ security incident,"[6] to increase awareness.

Mitigation: Primary mitigation for attacks like the XZ Utils compromise is a shared responsibility and investment by software manufacturers and system operators for the health and security of the open source ecosystem. Building relationships for real-time collaboration with open source community members is essential. This includes working with package repositories to scale security improvements. The "Lessons from XZ Utils: Achieving a More Sustainable Open Source Ecosystem"[7] article discusses the need for these relationships.

Red Hat is well-known for contributing resources to critical Open Source Software (OSS) projects, from code to technical thought leadership. Through close collaboration and trusted relationships with the OSS community, we were able to respond quickly and prevent further tampering.

North Korea employment fraud schemes

North Korean threat actors posed as remote IT professionals to secure employment with companies worldwide. Once embedded within organizations, these threat actors engaged in various malicious activities, including deploying malware, abusing access to proprietary source code, and conducting sensitive information theft. In response to these threats, the U.S. Department of Justice indicted 14 North Korean nationals[8] involved in these schemes.

Additionally, North Korean state-aligned groups targeted software engineers through fake job interviews and coding challenges as part of the recruitment process. A successful compromise of a developer's workstation can lead to supply chain related attacks, stolen stored credentials, or performing actions that appear to be conducted by legitimate employees.

Mitigation: Successful mitigation of employment fraud was achieved through due diligence in all stages of the interview and onboarding processes to detect this type of fraud comprehensively. Best practices include identity verification, cross-referencing and validating information such as contact details, curriculum vitae (CV) content, social media profiles, GitHub contributions, and following up with professional references.

[6] *Red Hat blog. "Understanding Red Hat's response to the XZ security incident." 30 April 2024.*

[7] *CISA blog. "Lessons from XZ Utils: Achieving a More Sustainable Open Source Ecosystem." 12 April 2024.*

[8] *Justice.gov. archives. "Fourteen North Korean Nationals Indicted for Carrying Out Multi-Year Fraudulent Information Technology Worker Scheme and Related Extortions." 12 Dec. 2024.*

Notable attack trends

Threat analysis of the 89 SSCA events in 2024 revealed at least 20 different tactics, techniques, and procedures (TTPs) deployed by threat actors across the cyber kill chain framework. The TTPs that made the top 3 list of most often used were:

- Publish malicious artifacts on public registries for download, which can be used by multiple applications to infect all the systems that use them.
- Use typosquatting to create a package or a container similar to a legitimate one to gain initial access.
- Discover secret leaks, which is often found in code repositories to advance an attack or demonstrate an impact to exploit victims.

This analysis proved the most popular and effective way to execute a software supply chain attack was to target open source dependencies.

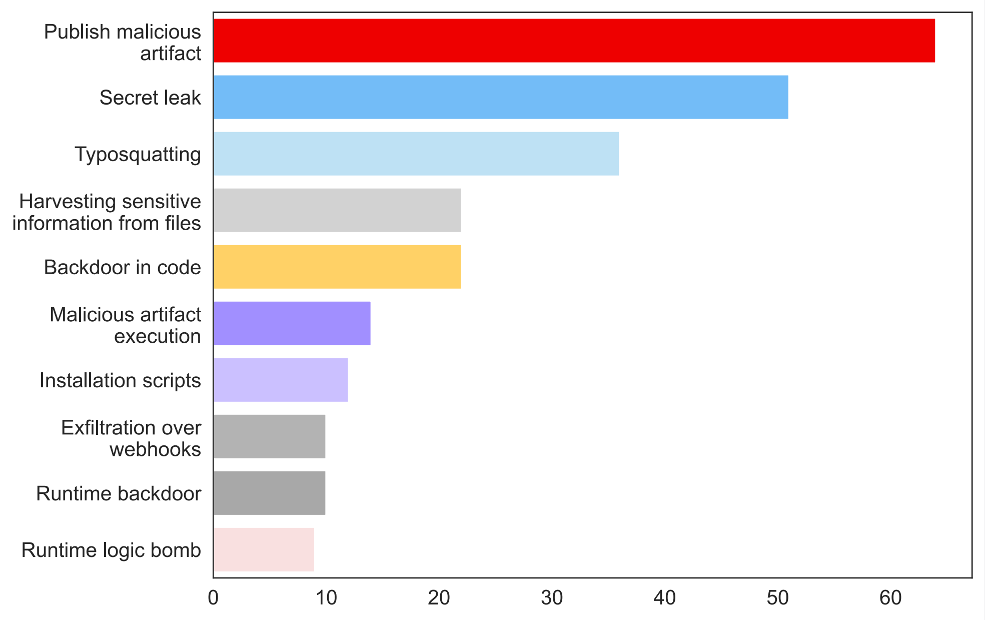


Figure 10. Top 10 most used TTPs used by threat actors based on the 89 SSCA events in 2024

Based on data collected by Red Hat in 2024, we assess that financially-motivated opportunistic attacks through the open source ecosystem are the most prominent type of software supply chain attack. The most common scenario is to upload a malicious package to a registry and use social engineering to lure developers to install it. We observed that node package manager (npm) emerged as the most abused registry, accounting for 37% of upstream victims, surpassing the Python Package Index (PyPI), which stood at 30%. GitHub ranked third with 15% and Visual Studio and VS Code fourth with 4%, surpassing NuGet that ranked fourth in 2023.

In 2024 and 2022, npm experienced the highest number of publicly reported attacks, while PyPI took the lead in 2023. The primary focus of threat actors and increasingly the 1st stage of SSCAs are npm, PyPI, and GitHub.

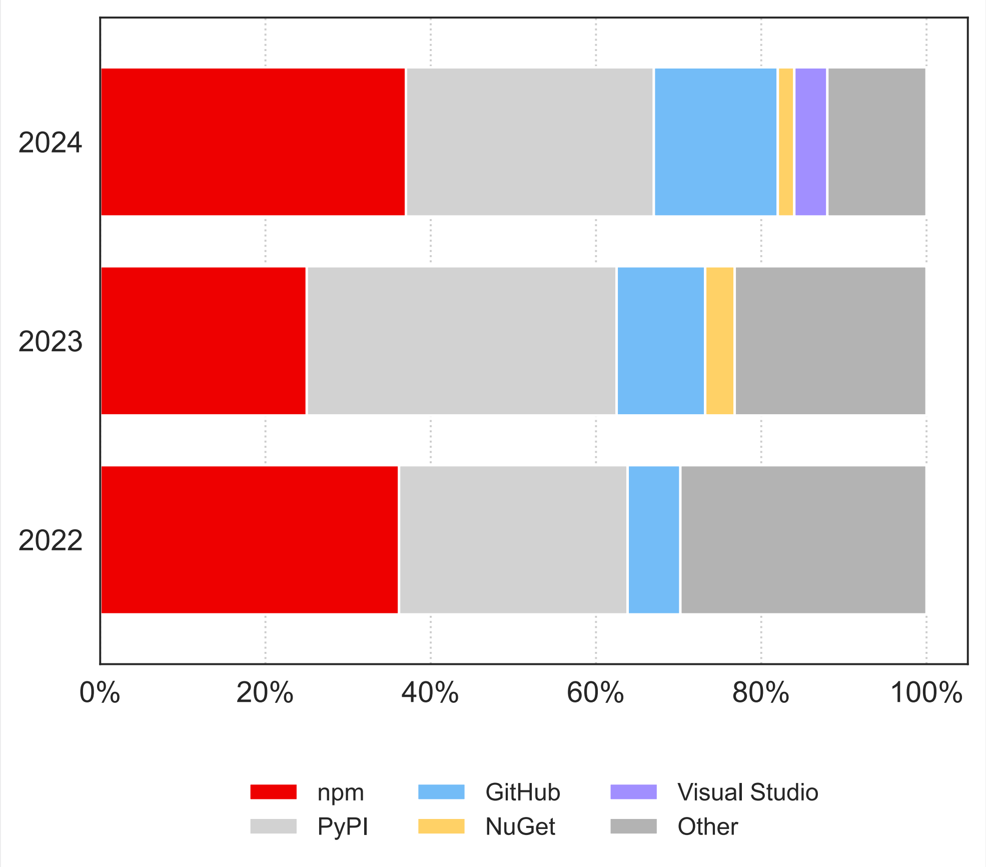


Figure 11. Highest number of publicly reported attacks 2022-2024

CVE-2024-3094 and CVE-2024-12356 were reported in 2024 as part of an SSCA operation, down from 8 events in 2023. For security professionals, this downward trend means we need to adapt from being hyper-focused on managing vulnerabilities to proactively identifying weaknesses and using that information to continuously improve and further safeguard the software development lifecycle.

Various forms of social engineering were used in the XZ Utils compromise and employment fraud-driven attacks. These attacks involved gathering victim information to gain initial access through developing and exploiting trusted relationships with software developers and open source maintainers.

Opportunistic attacks through the open source ecosystem typically result in either a leak of sensitive information or insertion of a malicious backdoor that allows a threat actor to maintain access and control over the system. In most cases, we assess this as the first stage of a more impactful attack with direct financial, reputational, and other implications for the end-user and organizations.

Motivation and attribution trends

SSCAs are predominantly motivated by 2 major factors: financial gain and espionage. Based on our collection of 89 publicly reported SSCA events in 2024, the data indicates that while in 56% of cases the motivation is unclear, 36% are financially motivated and 8% are espionage operations.

In almost 67% of the operations, the threat actors target developers and endpoints, 17% target cryptocurrency related end-users or organizations, and 2% target government entities.

In 2024, 82% of the reported SSCAs remain unattributed to known specific threat actors, 10% are linked to North Korean state-aligned groups, and 5% to groups from China. Since 2023, we observed a consistent pattern of North Korean involvement in SSCA, a slight 2% increase in attacks attributed to China-based threat actors, and a mere 1% attribution to Russian-based threat actors. The growing percentage of unattributed attacks makes it more difficult to understand the adversaries' intent and capabilities, which is often used for predictive threat analysis.

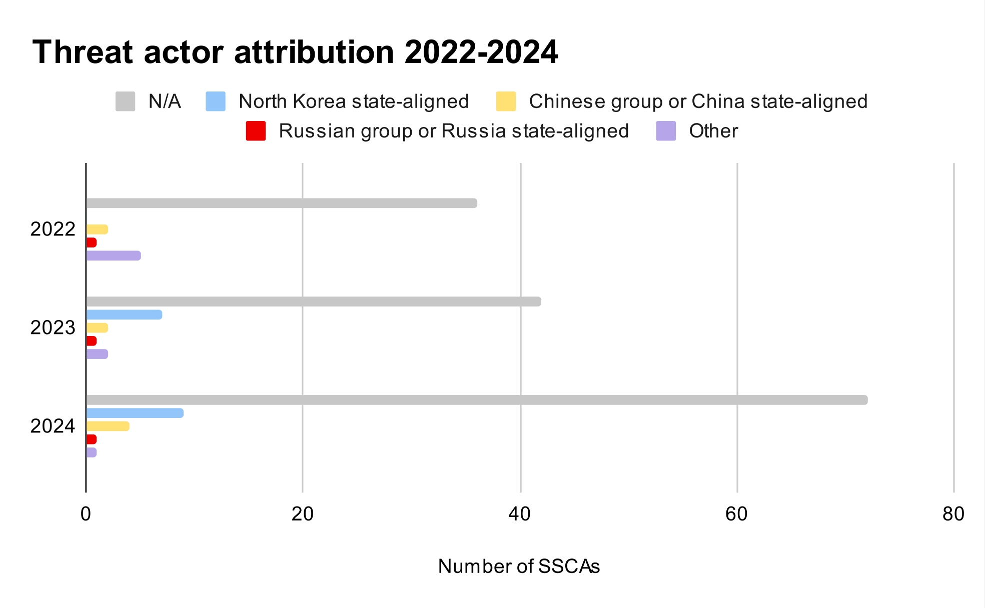


Figure 12. Threat actor attribution 2022-2024

AI security

Over the last year we have seen an explosion of demand for generative AI. Between now and 2030, an annual growth rate of 37.3%[9] is predicted. As with any new technology, it first needs to demonstrate its value. When that has been shown, many sectors, especially those that are strongly regulated, will start considering security and compliance implications. In fact, the top generative AI concern for the remainder of 2024 is data security risk.[10] In this section we discuss some of the work Red Hat Product Security is doing in the field of AI Security, Safety, and Trust.

It is important to understand and define the difference between an AI-security vulnerability and an AI-safety issue. Large language models (LLMs) can occasionally generate inaccurate, toxic, biased, or otherwise harmful content. These occurrences should be treated as safety issues rather than security vulnerabilities. Therefore, it is essential to establish a clear framework for handling safety concerns, recognizing that they require processes and protocols distinct from those of traditional security matters. Red Hat recently wrote a research paper on handling AI safety issues and proposed a way to track and manage them.[11]

We are currently working with various open forums, such as the Coalition for Secure AI, the AI Alliance, OpenSSF's AI working group, and the MITRE AI CVE and CWE working groups to design and develop processes and workflows for handling AI risks and governance specifically related to security and safety considerations.

Regarding compliance, society and governments have identified the need for robust ethical governance frameworks. The European Union (EU) and the National Institute of Standards and Technology (NIST) in the United States are leading the efforts on this matter. Some frameworks have emerged identifying and categorizing AI security risks. MITRE ATLAS and MIT AI Risk Repository are 2 of the most important to note.

The Red Hat Product Security AI team has been working on defining security reference architecture for AI software. We are also working on guidelines for engineers to reduce specific AI risks when including LLMs in their solutions, developing and publishing information and training to raise general knowledge about AI security and safety, and working directly with the engineering teams designing and developing products, including Red Hat Enterprise Linux AI and Red Hat OpenShift AI, to reduce AI specific risks to acceptable levels.

[9] *Hatchwork AI. "Generative AI Statistics: Insights and Emerging Trends for 2025." 2 Dec. 2024.*

[10] *Agility PR Solutions. "The top Generative AI concern for the remainder of 2024 is data security risk, say decision makers–what companies can do to protect data." 8 May 2024.*

[11] *Cornell University. "Building Trust: Foundations of Security, Safety, and Transparency in AI." ArXiv:2411.12275v1. submitted 19 Nov. 2024.*

Secure development update

In 2024, Red Hat further deepened its commitment to secure development by expanding and refining Secure Development Lifecycle (SDLC) activities. Our unwavering focus on the SDLC aims to mitigate risks for our customers by building every Red Hat product and service, including AI-based solutions, with inherent trust and reliability. Beyond aligning with industry standards like the National Institute of Standards and Technology (NIST) Secure Software Development Framework (SSDF), Red Hat strives to exceed these requirements, leading customer confidence and enhancing the value of our portfolio.

It is essential to identify and maintain component inventory and have proper documentation, secure configurations, and perform testing on integrated third-party tools as part of the Red Hat Secure Software Development Lifecycle (RH-SDLC). The RH-SDLC incorporates supply chain security activities, such as our Security Operating Approvals (SOA) procedure. Since publishing how Red Hat manages risk in its software supply chain in 2023, we matured our SOA procedure with verification automation across all systems identified for integration into our production pipeline. Our efforts in 2024 allow us to boost the value of security data, detect failures faster, and prioritize remediation of security issues using a risk-based approach.

Security Architecture Review

This year, Red Hat embedded Security Architecture Reviews (SARs) into the SDLC, guided by foundational secure design principles such as least privilege, defense-in-depth, and secure defaults. Our approach is grounded in the foundational principles of secure system design, which were first articulated in 1975 by Jerome Saltzer and Michael Schroeder in their seminal work, "The Protection of Information in Computer Systems."[12]

These principles, alongside more recent advancements, such as those outlined in "SafeCode Fundamental Practices for Secure Software Development,"[13] remain crucial to building and maintaining secure software systems today.

These concepts form the backbone of security in any IT system, and while they might be applied differently depending on the type of system, such as cloud services vs. on-prem solutions, they are all important in designing more trustworthy architectures. Read this Red Hat article, "Secure design principles in the age of artificial intelligence,"[14] to learn more about the key principles we use during our SDLC activities.

[12] *J. H. Saltier and M. P. Schroeder, "Protection of information in computer systems," IEEE CSIT Newsletter, vol. 3, no. 12, pp. 19-19, December 1975, doi: 10.1109/CSIT.1975.6498831.*

[13] *SafeCode. "Fundamental Practices for Secure Software Development, 3rd Edition." 28 Oct. 2019.*

[14] *Red Hat. "Secure design principles in the age of artificial intelligence." 22 Oct. 2024.*

RapiDAST

Last but by no means least, our RapiDAST tool continues to evolve as a robust open-source security testing tool, achieving significant advancements in 2024. Building on its foundation as a trusted community resource, this year saw significant improvements aimed at enhancing user experience and expanding its functionality.

One standout innovation is introducing a groundbreaking in-house feature for Kubernetes operator testing, Out-of-Band Testing for Kubernetes (OOBTKUBE). This feature allows for targeted security evaluations for meaningful vulnerabilities, specifically addressing remote command injection in Kubernetes operator services. While currently focused on addressing this specific issue, OOBTKUBE will be enhanced to cover additional security concerns in the future. With this addition, RapiDAST has significantly expanded DAST's coverage within the Red Hat Software Development Lifecycle (RH-SDLC) by enhancing its scanning capabilities to include Kubernetes operators, moving beyond only web and application programming interface (API) endpoints.

Additionally, we have prioritized usability by introducing an automated end-to-end (e2e) test pipeline and resolving issues that caused integration and automation challenges. These enhancements have made RapiDAST more stable, consistent, and user-friendly. RapiDAST now supports exporting scan results to central locations, such as Google Cloud Storage, further enhancing collaboration and accessibility.

The integration of RapiDAST into Red Hat Engineering teams' CI/CD pipelines has steadily progressed, receiving positive feedback and achieving widespread adoption. This momentum is expected to continue into next year, reinforcing RapiDAST's ongoing relevance and impact in secure software development practices.

Final words

"Red Hat has a long reputation for being a bastion of defense for our customers and the open source ecosystem. This year's risk report is the most comprehensive to date, looking beyond simple vulnerability statistics to consider the entire ecosystem of open source software, whether available directly from Red Hat or upstream projects and communities.

Last year, we saw a significant rise in CVEs assigned to the Linux kernel, predominantly in the Low and Moderate categories. While taking on the challenge of addressing a more significant number, our focus continues to be on the broadest, most applicable vulnerabilities. That said, these vulnerabilities are comparable to bugs that were not a big concern before, and they remain unlikely to pose a significant risk now. The Linux kernel is as security-focused as it has always been, and the bugs that have CVEs today are no worse than those that did not have a CVE assigned a year ago. It is extremely rare for a Low or Moderate Linux kernel bug to be used for actual exploitation. Effective exploitation is typically reserved for Important-rated vulnerabilities, according to the Cybersecurity and Infrastructure Security Agency (CISA) Known Exploited Vulnerabilities Catalog (KEV).

Further, the data continues to show that focusing on Critical and Important vulnerabilities is the right place for our attention. This year, we saw even less exploitation in Low and Moderate vulnerabilities than in prior years. Attackers will target vulnerabilities where effective exploitation is possible and that have the best chance of success, in effect looking for the best return on investment. Time and time again, we see where their focus is, and as defenders, that is also where our focus remains: Critical and Important vulnerabilities.

This is especially true when it comes to response times for truly critical issues. Last year, it was XZ. In 2021, it was log4j. The open source community response to these events is nothing short of impressive. As a long time proponent of open source, these unfortunate and disruptive events, actually highlight the benefits of the open source development model. In an emergency, the community at large makes every effort to contain the threat. I take a great sense of pride in this and believe that how an entity responds is equally, if not more, important than prevention. No matter how hard we try, there will always be security issues in both open source and proprietary software. The real hallmark of responsible vulnerability management is the speed and appropriateness of response.

The data also shows there is much work to do, and an area of focus for proactive work is the upstream repositories that hold all the modules and libraries used in everyday development projects both in-house and external, in open source and proprietary software alike. The continued attacks on these repositories for publishing malicious artifacts or taking advantage of typos continue to be a threat we must guard against. We should redirect the time and energy we save by not focusing on Low and Moderate vulnerabilities, which cannot be effectively exploited, towards protecting against these types of attacks, ultimately leading to better overall ecosystem security outcomes."

- Vincent Danen, VP Product Security, Red Hat

facebook.com/redhatinc
@RedHat
linkedin.com/company/red-hat

North America
1 888 REDHAT1

Europe, Middle East, and Africa
00800 7334 2835
europe@redhat.com

Asia Pacific
+65 6490 4200
apac@redhat.com

Latin America
+54 11 4329 7300
info-latam@redhat.com