WhiteSource

THE STATE OF OPEN SOURCE VULNERABILITIES MANAGEMENT

Open source usage is a mainstream practice in this day and age. You simply cannot develop software in today's competitive go-to-market timelines without it. The rise in open source usage has lead to a dramatic rise in open source vulnerabilities, bringing to the fore interesting developments in open source security.

The State of Open Source Vulnerabilities Management drills down into the deeper layers of the open source phenomena. Surveying over 650 developers from the US and Europe and collecting data from multiple data sources including the NVD, security advisories, peer-reviewed vulnerability databases, and popular open source issue trackers, this report brings the latest in open source security management. Our mission is to determine where we are as an industry to know where we can go in years to come.

KEY FINDINGS:

1. A significant rise in the number of open source vulnerabilities presents a serious challenge to development and security teams striving to meet security objectives.

2. Developers spend a lot of time addressing open source vulnerabilities, but the absence of standard practices and developer-focused tools result in an inefficient use of time.

3. A prioritization strategy for open source vulnerabilities is critical to ensure companies address the most critical issues on time.

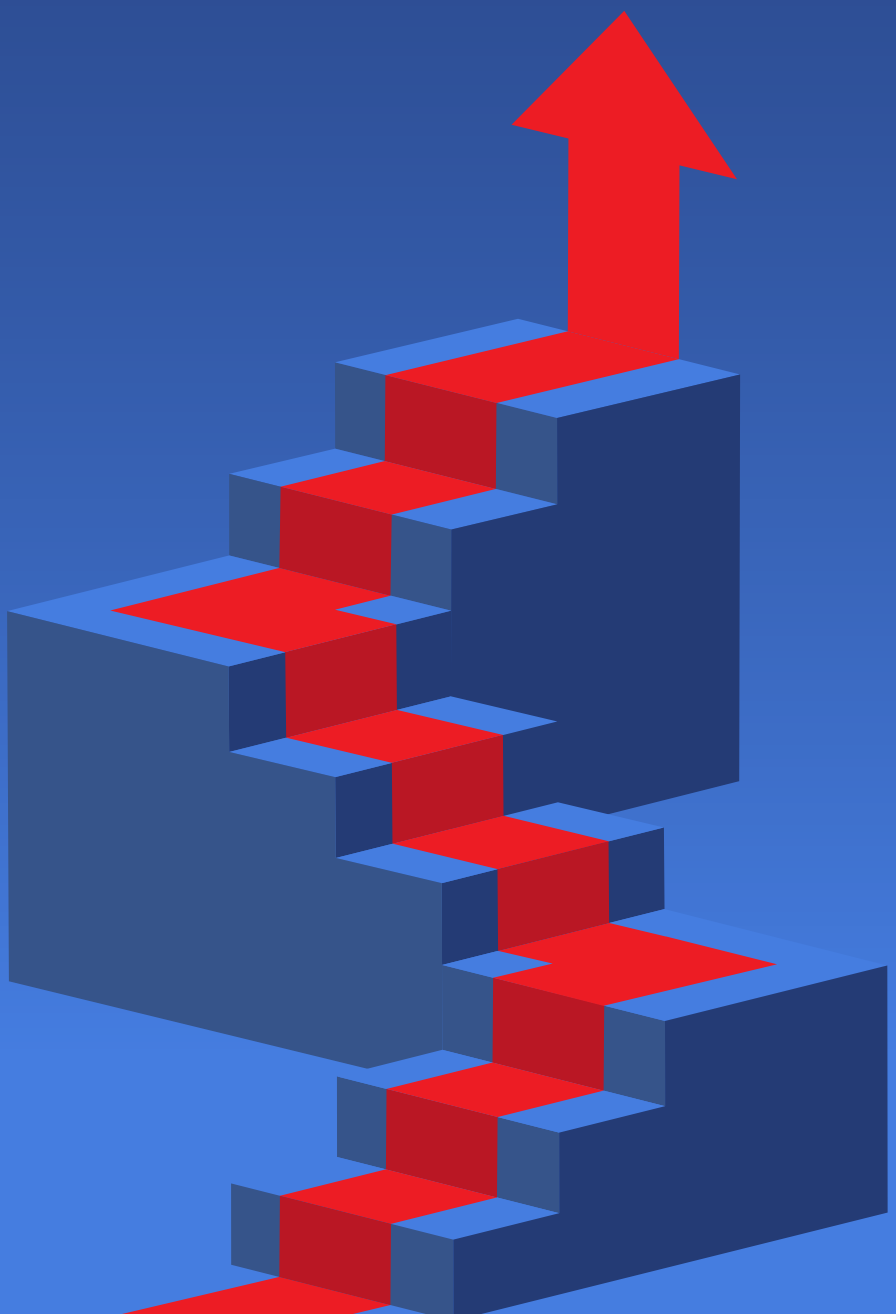
4. A solid prioritization practice for open source vulnerability remediation can reduce security alerts by 70% to 80% due to the nature of open source.

1

OPEN SOURCE SECURITY VULNERABILITIES ARE ON THE RISE

Key Takeaway:

A significant rise in the number of open source vulnerabilities presents a serious challenge to development and security teams striving to meet security objectives.

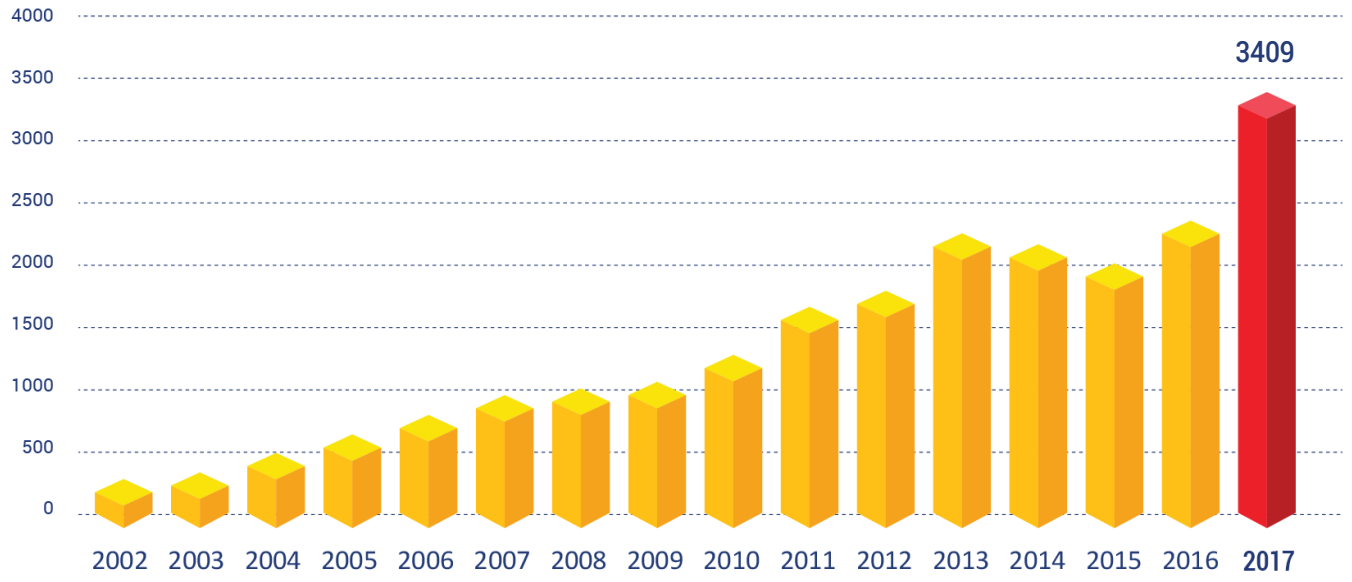


The number of disclosed open source vulnerabilities skyrocketed in 2017, reaching almost 3,500 reported vulnerabilities.

According to the WhiteSource database, aggregated from the NVD, security advisories, peer reviewed vulnerability databases, and popular open source issue trackers, the number of disclosed open source vulnerabilities in 2017 rose by over 60% as compared to 2016. We can see this trend continues in 2018.

This can be attributed to the software development community's focus on open source security following the widespread adoption of open source components and heightened awareness to security vulnerabilities due to publicized data breaches.

NUMBER OF REPORTED OPEN SOURCE VULNERABILITIES ROSE BY 61.2% IN 2017

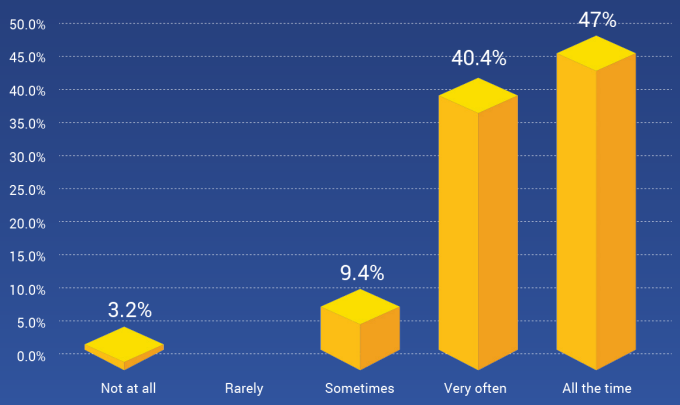


WhiteSource conducted a survey encompassing 650 developers from the US and Western Europe, asking about their practices and challenges of open source usage.

According to survey results, only a negligible percentage of developers do not use open source components, probably as a matter of policy. And the ones who do, rely upon it regularly.

96.8% of developers rely on open source components and are therefore affected by the recent rise in the number of known vulnerabilities.

FREQUENCY OF USE OF OPEN SOURCE COMPONENTS



The risks are even more severe, since the majority of disclosed open source vulnerabilities lie in a limited number of projects – the most popular ones!

The more popular an open source project is, the larger its community and the more 'eyeballs' it garners from security researchers. With more contributors looking at it, more security and quality issues are discovered and made public every month.

According to the WhiteSource database, 7.5% of all open source projects are vulnerable. Of the 100 most popular projects, 32% are vulnerable.

While one vulnerability is enough to put multiple libraries at risk, vulnerable open source projects contain 8 vulnerabilities on average.

The list of top 10 open source projects with the highest number of known open source vulnerabilities includes projects we are all familiar with and which many of our products depend on.

It's no coincidence that the majority of these projects are internet facing front-end components with large attack surfaces that are very exposed, making them relatively easy to exploit and therefore attracting a lot of focus.

Another interesting fact here is that there are commercial companies behind most of these projects. Remember that a high number of reported vulnerabilities usually implies that a project is properly maintained by its community, and not an indication of poor security standards.

TOP 10 VULNERABLE OPEN SOURCE PROJECTS BASED ON NUMBER OF VULNERABILITIES

Open source projects	Open source projects
Mozilla Firefox	1470
Linux	1249
Chromium	602
Wireshark	549
Oracle Java SE	531
PHP	486
Moodle	451
ImageMagick	415
FFmpeg	281
WordPress	245

In our list of most vulnerable languages, C/C++ gets first place by a considerable margin with an overwhelming 41% of all vulnerabilities in our database.

The most popular programming language in use today, JavaScript, only takes 4th place, with a mere 7% of vulnerabilities.

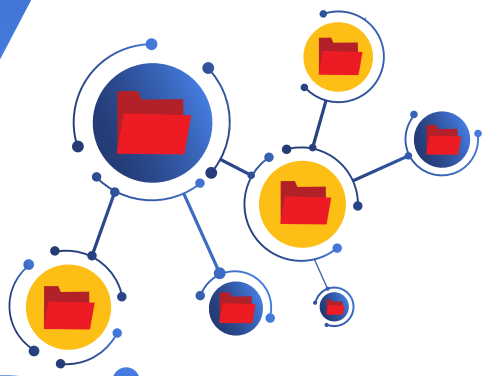
TOP 7 LANGUAGES WITH THE HIGHEST NUMBER OF VULNERABILITIES

Languages	% of Vulnerabilities
C/C++	41%
PHP	17%
Java	8%
JavaScript	7%
Python	6%
Ruby	4%
C#	1%

But, it's not all bad.

The rise in awareness that led to the detection of open source vulnerabilities also led to a sharp rise in the number of suggested fixes offered by the community, usually published within days of the release date.

97.4% OF ALL REPORTED VULNERABILITIES HAVE AT LEAST ONE SUGGESTED FIX IN THE OPEN SOURCE COMMUNITY



Unfortunately, while the open source community is doing a great job securing open source projects, users are unable to fully benefit from their efforts.

The problem is that information about vulnerabilities is not published in one centralized location, it is rather scattered across hundreds of resources and usually poorly indexed and therefore unsearchable.

This presents an ongoing challenge for developers with detecting open source components with known vulnerabilities.

OVER **86%** OF REPORTED OPEN SOURCE VULNERABILITIES APPEAR IN THE CVE DATABASE

2

DEVELOPERS ARE NOT EFFICIENTLY MANAGING OPEN SOURCE VULNERABILITIES

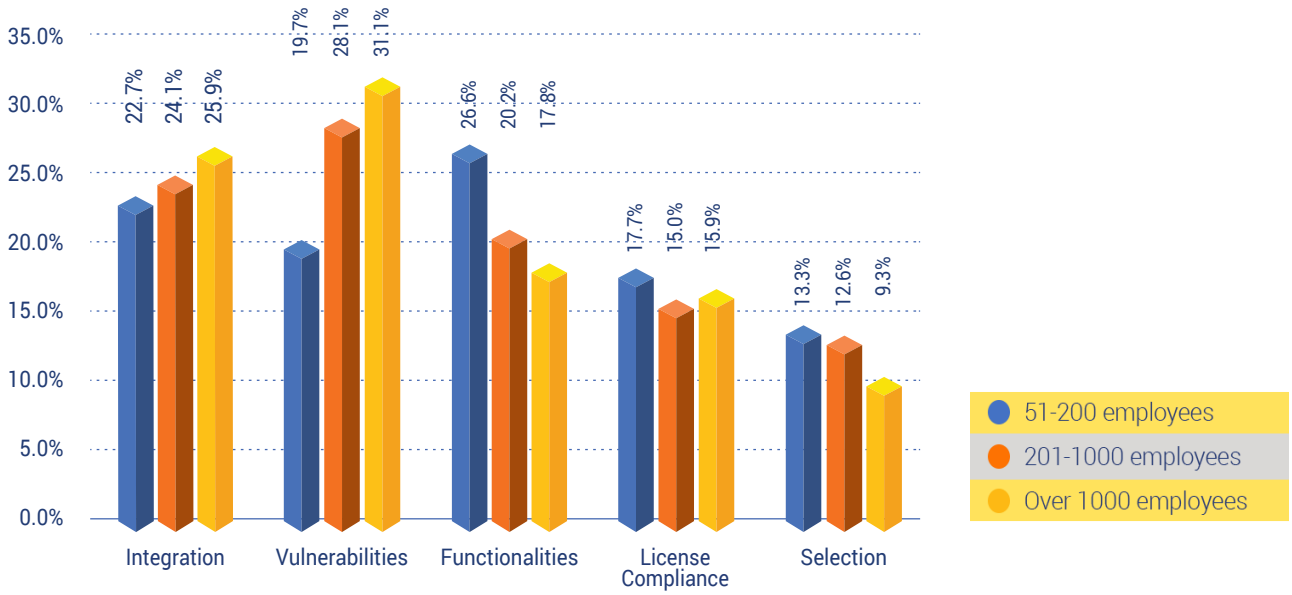
Key Takeaway:

Developers spend a lot of time addressing open source vulnerabilities, but the absence of standard practices and developer-focused tools result in an inefficient use of time.

The rise in open source vulnerabilities has not gone unnoticed by developers. Our survey clearly shows that developers have come to consider open source security vulnerabilities as their #1 challenge when using open source.

26% of developers rated security vulnerabilities as the top challenge posed by open source components. Open source vulnerabilities have been ranked above integration, functionality, licensing and selection. Larger organizations, in particular, are more concerned with open source security.

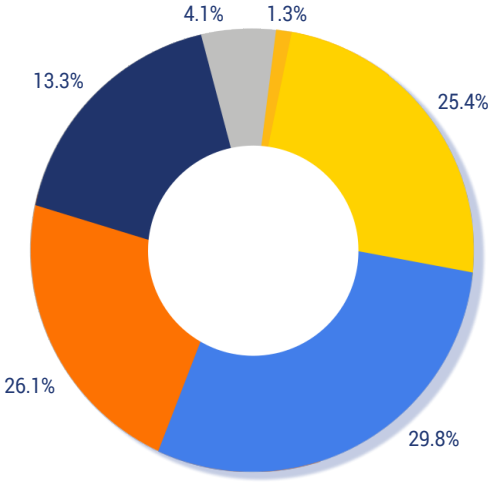
#1 CHALLENGES IN USING OPEN SOURCE COMPONENTS



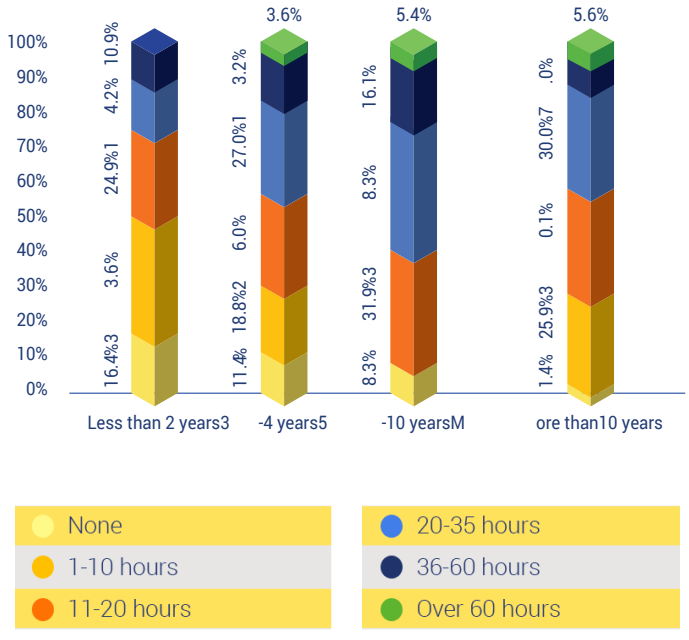
This concern has become a costly issue, with developers spending almost 15 hours each month dealing with open source vulnerabilities (e.g. reviewing, discussing, addressing and remediating).

The cost is even higher when we consider that the more experienced developers are usually the ones tasked with remediating open source vulnerabilities.

HOURS SPENT PER MONTH HANDLING OPEN SOURCE VULNERABILITIES



HOURS SPENT ON OPEN SOURCE VULNERABILITIES PER DEVELOPERS' EXPERIENCE

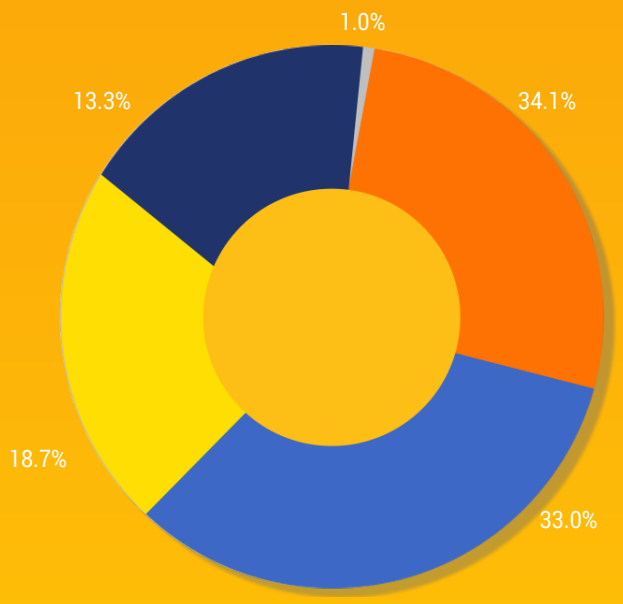


The WhiteSource survey also shows that 15 hours per month of developers' time go into addressing open source vulnerabilities, as compared with only 3.8 hours monthly for their actual remediation.

The absence of standard practices became apparent when developers were asked what they do when a vulnerability is discovered, and provided numerous answers with no clear practice.

The lack of set practices when attending to newly discovered vulnerabilities explains the inefficient use of time when addressing open source vulnerabilities.

WHAT DO YOU DO WHEN A VULNERABILITY IS FOUND?



- Nothing
- Research to better understand the vulnerability and its impact
- Remediate based on the open source community recomendation
- Report to the other teams (Security/DevOps) or a manager
- Remediate only through patches (if available)

3

PRIORITIZATION IS KEY TO OPEN SOURCE VULNERABILITY MANAGEMENT

Key Takeaway:

A prioritization strategy for open source vulnerabilities is critical to ensure companies address the most critical issues on time.

Inefficient remediation practices for security vulnerabilities is becoming a prime concern considering security teams get overwhelmed with alerts on a daily basis.

Top industry experts agree that attempting to solve every single issue is impossible, and that prioritization is key to efficient vulnerability management.

> Perfect security is impossible. Zero risk is impossible. We must bring continuous risk and trust-based assessment and prioritization of application vulnerabilities to DevSecOps. In a futile attempt to remove all possible vulnerabilities from applications, we are slowing developers down and wasting their time chasing issues that aren't real (false positives) or addressing lower-risk vulnerabilities that are real, but not directly or easily exploitable.
>
> 10 Things to Get Right for Successful DevSecOps
> Neil MacDonald, Gartner

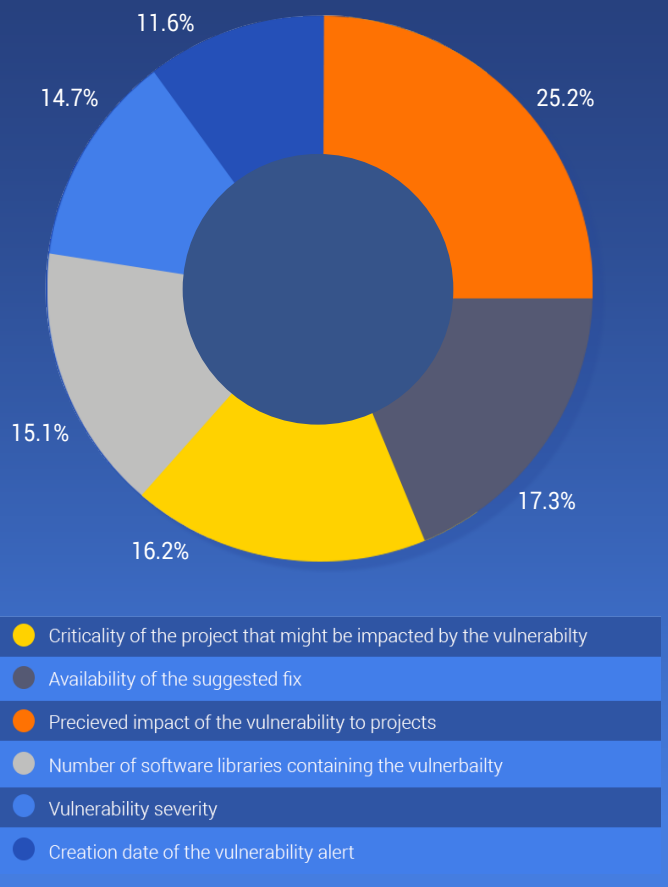
Survey results show that developers lack standardized best practices for prioritizing open source vulnerabilities.

The results also indicate that developers often look to the most readily available data when prioritizing remediation, such as the criticality of an application or the availability of a fix. However, developers tend to prioritize based on available data and not necessarily on the correct one.

In a race against hackers, time is of the essence - especially when it comes to open source vulnerabilities where data is public.

Therefore, lack of vulnerability prioritization leads to inefficient use of resources, when developers are investing time in the "wrong" vulnerabilities.

DEVELOPERS ARE LACKING STANDARD PRACTICES TO PRIORITIZE OPEN SOURCE VULNERABILITIES

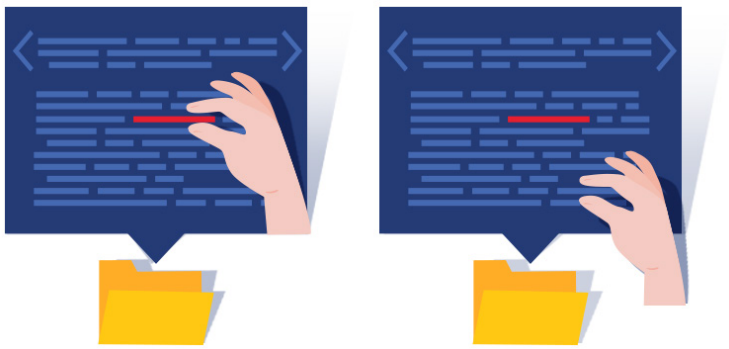


A new approach to prioritizing open source vulnerabilities should be based on the impact on the product's security.

A vulnerable functionality does not necessarily make a project vulnerable, since the proprietary code may not be making calls to that functionality (i.e. making it ineffective).

Determining whether a vulnerability poses an actual risk by understanding its effectiveness, can save security and development teams precious time.

EFFECTIVE vs INEFFECTIVE
VULNERABILITIES IN A COMPONENT



EFFECTIVE VULNERABILITY
If the proprietary code is making calls to the vulnerable functionality

INEFFECTIVE VULNERABILITY
If the proprietary code is NOT making calls to the vulnerable functionality

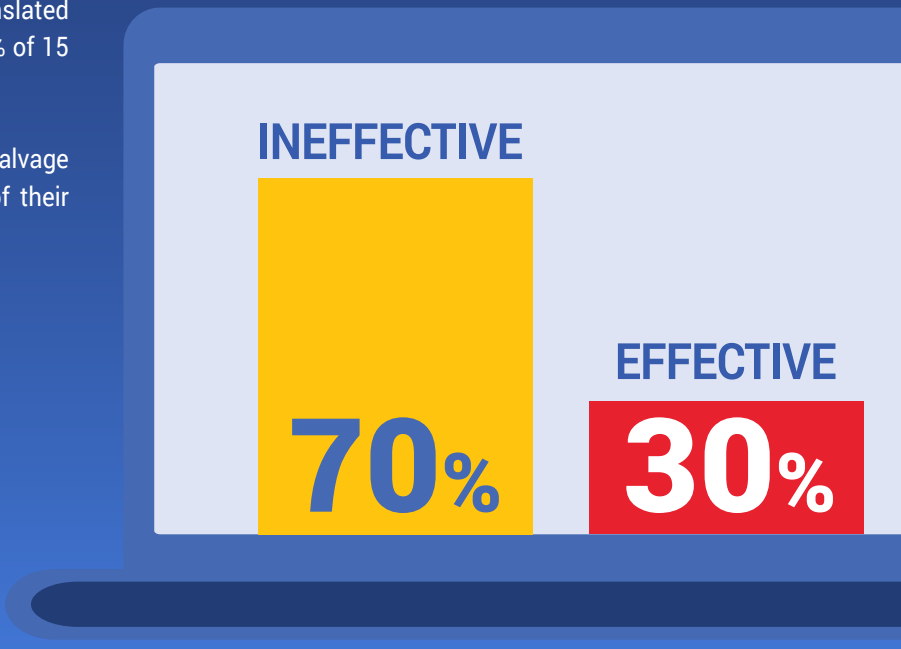
After testing 2,000 Java applications, WhiteSource found that 72% of all vulnerabilities detected in these applications were deemed ineffective.

Analyzing effective vs. ineffective vulnerabilities demonstrates how powerful prioritization based on effectiveness is. Data proves that the number of open source vulnerabilities alerts can be reduced by 70%.

Based on the data collected in our survey, this can be translated to saving 10.5 hours per month per each developer (70% of 15 monthly hours).

Organizations that adopt prioritization practices can salvage precious development time and improve the security of their products with quicker remediation of critical issues.

PRIORITIZATION BASED ON EFFECTIVE VULNERABILITIES REDUCES ALERTS BY 70% IN JAVA APPLICATIONS



4

EFFECTIVE USAGE ANALYSIS

A solid prioritization practice for open source vulnerability remediation can reduce security alerts by 70% to 80% due to the nature of open source.

WhiteSource recently launched a new technology for prioritizing open source vulnerabilities based on the way they are used by the application - **Effective Usage Analysis.**

Our beta testing on 25 commercial applications from 12 organizations showed that:

ALL analyzed projects were found to be vulnerable, with at least one open source vulnerability. On average, each project contained 8.2 vulnerable libraries.

90% of the vulnerabilities (effective and ineffective) were found in transitive dependencies, which emphasizes the importance of accurate open source inventory tracking for security purposes.

86% of all open source vulnerabilities alerts were found to be ineffective, having no impact on the security of the projects.

64% of all analyzed projects were found to contain only ineffective open source vulnerabilities, and therefore did not require any remediation efforts.

The volume of ineffective open source vulnerabilities found shows that there is great potential for reducing the time and effort that organizations currently invest in research and remediation of open source vulnerabilities.

The Effective Usage Analysis tool helped development teams save time by locating and focusing their resources on the effective vulnerabilities that required immediate attention. Team leaders and managers testified that the new technology:

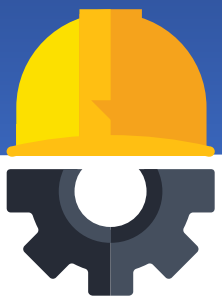
- Enables effortless prioritization of open source vulnerabilities by classifying them visually according to their impact on the project's security.
- Facilitates remediation of open source vulnerabilities by identifying the file and line numbers of direct and indirect calls made from proprietary code to open source vulnerabilities.
- In addition to handling a significantly lower number of vulnerabilities, an additional 10%-20% of the time developers invest in remediation was saved.

> "The best thing about this new technology is being able to prioritize which vulnerabilities need to be remediated first. It's been easier tackling the vulnerabilities in our products with a technology that is so easy and scalable."
>
> Senior Director Application Information Security, IGT

> "After seeing the dramatic results of our beta testing, we have no doubt that implementing EUA on our entire code will be a breakthrough in both reducing effort and improving our results when dealing with open source security vulnerabilities."
>
> CIO, ECI

> "Effective Usage Analysis gives us the added value of faster remediation, with trace analysis that pinpoints the exact location of vulnerable dependencies. This new capability enables us to significantly cut down on the time our developers spend dealing with open source vulnerability alerts."
>
> Senior Release Manager, ForgeRock

ABOUT WHITESOURCE



WhiteSource's vision is to empower businesses to develop better software, by securing and managing the open source components in their software.

A trusted leader in Software Composition Analysis, WhiteSource helps industry giants like Microsoft, IBM, and hundreds more to harness the power of open source with their continuous open source security and license compliance management solution.

Founded in 2011.
Offices in NY,
Boston & Tel-Aviv

500+
Customers

Empowering over
1.2M
developers

Supporting
23%
of Fortune 100
companies

Over
300%
growth YOY