mend.io

October 2023

Practical Approaches

for Reducing Security Debt

Practical Approaches for Reducing Security Debt

We've all scrolled by life-hack videos of better ways to fold fitted sheets (start at the corners) or stack clothes in drawers (vertically for visibility), but where are the practical tips for really important challenges, such as reducing open-source security debt? Too often, tips for reducing security debt can be more strategic or visionary instead of providing tactical advice.

There's no doubt that development teams could use the help. The trick for any organization using open-source components (and that's the vast majority of organizations these days) is to find ways to reduce security debt and overall risk without negatively impacting either the development process or software functionality.

While it may not be equivalent to life hacks like using a long string of dental floss to cut a cake into slices cleanly, there are a handful of practical tips organizations can use to reduce open-source security debt without compromising functionality.

The Development vs. Security Open-Source Tradeoff

Although security debt typically matters most to cybersecurity teams, it is frequently incurred by developers, who may be shielded from its ultimate impact. Fixing a security risk (such as updating an out-of-date component) that could be a problem down the road may unintentionally cause functional pain immediately if it breaks something in the application.

As a result, developers may accrue security debt as they put off security updates in favor of not negatively impacting functionality. But that can expose organizations to potential security, licensing, and other risks. The challenge for organizations is to improve open-source security without burdening the developers more than they already are.

Strategically Solving the Security Debt Problem

That's why forward-looking AppSec programs are finding ways to reduce security debt by instituting a more strategic approach to managing potential vulnerabilities. This preventive approach to application security starts by reducing the attack surface during development. It focuses on enabling an organization to prevent future problems and prepare for efficient remediation if issues occur.

By integrating such tactics, they can move from remediating security issues after they've been discovered to proactively preparing for and preventing future security issues. As a result, organizations can solve their security debt problem more strategically.

Taking a more proactive approach helps organizations reduce the impact of security vulnerabilities, shrink the application attack surface, decrease corporate risk, and improve overall security.

Reducing open-source security debt starts with two steps that, although obvious, are not always easy to put into practice.

Preparation

Even the most secure organization can't prevent all security attacks from happening, so the next best option is to ensure if something does happen, that the organization can respond and resolve it quickly. Organizations should automate the development, remediation, and deployment processes whenever possible. This allows them to streamline processes so that if (or when) security problems need to be remediated, it can be done easily and quickly.

Prevention

Preparing for inevitable problems is good, but preventing problems is better. And the best way to do that when using open-source software is to proactively update open-source dependencies, because this typically reduces the application attack surface. Organizations should ensure their deployed applications are using the latest open-source components to eliminate any potential known security risks and reduce the number of possible problems from out-of-date libraries if emergency updates are needed.

Organizations doing open-source development need to balance new development efforts against investing resources in increasing security. However, by taking the two steps above, organizations can reduce security debt immediately, making their applications more secure. And if done right, the steps won't negatively impact developers' workload and may even free up development resources.

Practical Tips for Reducing Security Debt

Internet life hacks can often make day-to-day tasks easier once you learn the secret tip or process. It's the same with open-source security. These four tactics are a great place to start:

TIP 1

Automate dependency management

An out-of-date open-source component is a security risk. Automated dependency checking reduces that risk while ensuring security consistency and reducing the impact on development resources. Organizations should implement an automated dependency management routine that basically runs on autopilot—checking open-source dependencies consistently, flagging issues, and assisting in the remediation process.

How it works?

A good example of how automation can simplify dependency management is the Smart Merge Control feature within Mend SCA and Mend Renovate Enterprise Edition. Mend's Smart Merge Control is like autopilot for component updates. It can examine dependencies within a project and batch only the updates that have a high confidence level that they will pass build tests and not break the build.

Smart Merge Control provides a high degree of automation, including identifying all the high confidence updates, generating the associated pull requests, and then merging them, all automatically but with ultimate developer oversight and control.

TIP 2

Assess confidence levels

Any time an organization updates open-source components for newer, potentially more secure versions, there is a risk of functional problems. That's where confidence levels come in. By using and assessing confidence levels associated with new versions of open-source components, organizations can make a pretty accurate estimate of whether updating a given component will negatively impact application functionality or cause other issues.

Open-source component confidence levels can reduce the potential for unexpected problems or downtime while making it easier to upgrade—and, ultimately, shrinking application attack surface.

How it works?

Mend's Merge Confidence ratings, available as a standard feature in both Mend SCA and Mend Renovate Enterprise Edition, are a real-world example of how assessing confidence levels of open-source components can streamline the software development lifecycle. Merge Confidence levels are based on peer-sourced data from over 25 million dependency updates tracked by Mend Renovate. Developers can merge updated components with high confidence levels with assurance that they are unlikely to break the build. And when a component has a lower confidence level, it can flag to the developers that extra work may be required to merge it, so they can plan appropriately.

TIP 3

Create batch updates

Most organizations doing open-source development may be consuming dozens, hundreds, or even thousands of open-source libraries in their applications. Keeping them all up-to-date is time consuming and resource intensive. That's why one way to reduce security debt while not negatively impacting application functionality is to use the confidence levels mentioned above to create batch updates of high-confidence open-source updates. That way, multiple components can be updated at once, streamlining the process, simplifying updates, and increasing security without lots of hands-on vetting.

How it works?

Open-source development projects are getting increasingly complex, with more and more components. That means checking dependencies and updating applications tends to take more time and effort. That's why the Batch Update capability is especially important. Mend's Smart Merge Control functionality provides a way for developers to batch updates (typically high-confidence updates) into a single collection that can be applied all at once. Even though it's not complex work, manually generating 10, 20, or 50 pull requests for components that need updating can be time-consuming and boring. Mend's Smart Merge Control functionality eliminates the need to do that manually and further automates the update process.

TIP 4

Prioritize remediations and updates within the repository

Another way to reduce security debt is to fix security problems earlier in the development cycle. By implementing a way to prioritize remediations and updates within an organization's repository, companies can reduce the burden on developers while simultaneously lowering security debt.

How it works?

Sometimes the easiest and fastest way to fix a problem is at the source. That's why Mend SCA and Mend Renovate Enterprise Edition's ability to update components directly within the repository, as part of a developer's typical workflow, can make it significantly easier for organizations to keep components up to date. Mend SCA provides integrations with all major repositories, enabling developers to create pull requests themselves, without ever leaving the repository environment. Mend SCA gives automated prioritization advice, helping developers understand which risks need to be addressed ASAP. There's no need for them to switch to a different UI or learn a new tool—the update process happens within their existing environment.

Open-source Life Hacks

Just as you can iron a button-down shirt inside-out to save the time and hassle of having to iron around the buttons, you can also take a proactive approach to reducing open-source security debt. These real-world open-source life hacks will have a substantial impact on security debt and help companies build a more proactive security stance.

Luckily, organizations don't have to reinvent the wheel to implement any of these hacks. They are all currently available as part of Mend's dependency management solutions.

Take a first step in reducing your organization's security debt by checking out Mend SCA and Mend Renovate Enterprise Edition.

mend.io