



**Empowering
developers to build
secure software faster**



Table of contents

I: Teams need a different approach to application security	3
II: The solution that empowers developers	5
III: How GitHub can help	7
IV: Conclusion	11



I: Teams need a different approach to application security

These days, developers are expected to build code and ship software faster than ever before. In fact, over 80% of top-performing engineering teams deploy software multiple times per day.¹ To keep up with this demand for speed and innovation, developers are adopting open-source software (OSS), including culture, best practices, and frictionless code reuse—which can increase productivity by up to 87%.² As an industry, we’ve recognized OSS’ power and have created DevOps tools that support developers and organizations to collaborate, innovate, and ship faster.

However, security got left behind. Despite the energy around DevOps growth and the billions of dollars invested into application security testing tools, we still don’t have effective security solutions. Today’s grim security stats bear witness: 85% of applications still contain known vulnerabilities.

1 Humanitec, DevOps Setups: A Benchmarking Study, 2021.

2 GitHub, State of the Octoverse, 2021.



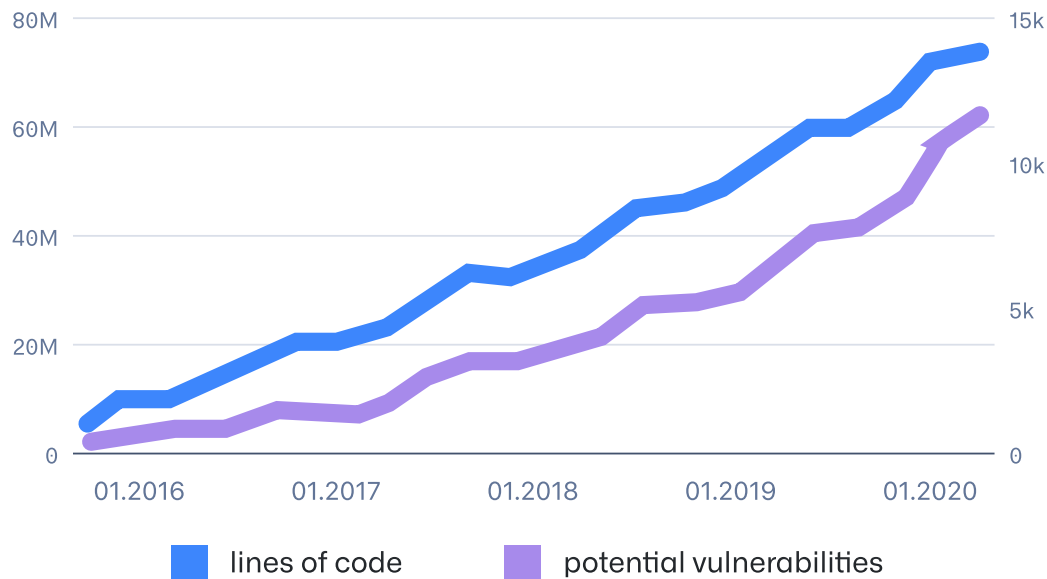
**WE FORGOT SECURITY.
DEVSECOPS IS WHAT
DEVOPS SHOULD
HAVE BEEN FROM
THE BEGINNING.”**

—
WILLIS
2021



Did you know that a line of code written in 2020 is just as likely to introduce a security vulnerability as one written in 2016?

Potential vulnerabilities found in source code scale with lines of code written



While billions of dollars have been invested in OSS security, traditional methods have failed.

85%
OF APPLICATIONS CONTAIN
KNOWN VULNERABILITIES
WITH MOST BREACHES
OCCURRING AT THE
APPLICATION LAYER

Remediation remains a challenge across all testing types and often takes longer than six months.



Our security posture is weak because our tools and approaches haven't been developer-friendly. While AppSec solutions are great at finding vulnerabilities, they haven't enabled developers to fix them. Remediation is challenging and often takes longer than six months.

Today, most organizations rely on multiple fragmented tools that are difficult to embed into the developer pipeline. When they are integrated, they're often disabled by developers because of failed testing issues and system performance impacts. This creates a large amount of technical debt that teams struggle to address.

Compounding these issues, developers and security teams remain siloed. Most developers lack security knowledge, and there are often not enough AppSec professionals to fill the gap.

II: The solution that empowers developers

To empower your teams, you need a solution to secure your software supply chain and custom code across the software lifecycle. This tool should enable you to identify



and fix security issues quickly, increase development speed, and provide seamless collaboration between engineering and security teams, no matter where in the world they are.

Your holistic application security solution should provide:



Developer-embedded experiences in all development phases, from initial planning to ship



Visibility into your security posture across your code, secrets, and supply chain



Crowd-sourced security intelligence from millions of researchers and developers



Easy collaboration and customization



An extensible and agnostic platform



Comprehensive ability to drive fixes that speed you up, not slow you down



**SECURITY AT THE EXPENSE
OF USABILITY COMES AT THE
EXPENSE OF SECURITY.”**

—
AVI DOUGLEN
OWASP Board of Directors



III: How GitHub can help

GitHub Advanced Security (GHAS) natively embeds security into the developer workflow—enabling you to secure your software supply chain and proprietary code across the software lifecycle. With GHAS, there are automated security checks at every pull request. Identified security issues are shared immediately within the familiar GitHub workflow. This empowers teams to fix vulnerabilities in minutes, not months. Having these native security capabilities integrated into the GitHub platform gives you:



Detailed changelogs across security issues and fixes



Visibility into your security posture across code, secrets, and supply chain



Crowd-sourced security intelligence from millions of developers and security researchers around the globe

This complete, native, and automated approach enhances productivity, reduces risk, and improves time-to-market. And because most developers are already familiar with GitHub, GHAS also eliminates the need to learn new tools—increasing speed, collaboration, and developer satisfaction.



**GITHUB HELPS US ENSURE
THAT WE HAVE OUR
SECURITY CONTROLS
BAKED INTO OUR PIPELINES
ALL THE WAY FROM THE
FIRST LINE OF CODE.”**

—

MIGUEL EL LAKKIS
Chief Information
Security Officer
Dow Jones



Explore the components of GHAS

Code Scanning is a developer-first static application security testing (SAST) product that examines your code for security issues as you're writing it and integrates fixes natively into your developer workflow. This makes it simple to find and fix security vulnerabilities before they ever reach production.

When code scanning is enabled, every git push is scanned for new potential vulnerabilities. Results are displayed directly in your pull request. Code scanning uses CodeQL, which includes more than 2,000 CodeQL queries written and open-sourced by the GitHub Security Lab and leading researchers. This helps you find vulnerabilities with minimal configuration.

> To learn more about code scanning, visit our [code scanning Docs page](#).

Refresher: What is SAST?



SAST is a testing method that analyzes your source code to find security vulnerabilities.



SAST uses your application source code or binary code as input and scans it for known vulnerable code patterns. This generates identification of potential vulnerabilities.

GHAS
EMPOWERS DEVELOPERS
TO FIX VULNERABILITIES IN
MINUTES, NOT MONTHS.



OUR ENGINEERING TEAMS
PLACE TREMENDOUS
VALUE IN GITHUB'S LATEST
SECURITY FEATURES LIKE
VULNERABILITY ALERTS
AND AUTOMATED FIXES."

—
JON PARISE
Engineering Architect
Pinterest



Secret scanning prevents unauthorized access and breaches by proactively scanning for secrets pre-commit and searching repositories for leaked secrets that may have accidentally been pushed into your code. This involves scanning your code for patterns from our partners, including AWS, Slack, Google Cloud, and Azure. Because scans take less than a second, we can quickly catch leaks as they occur.

Why is it important to check for secrets?

Leaked secrets allow attackers to gain access to legitimate credentials and, as a result, give them a wealth of information that can be used for malicious purposes. According to the [Verizon 2021 Data Breach Investigations Report](#), 61% of breaches are attributed to leveraged credentials. To date, GHAS has detected more than 700,000 secrets across thousands of private repositories.

If your project communicates with an external service, you can use a token or private key for authentication. If you check a secret into a repository, anyone in the repository can use the secret to access the external service with your privileges.

> To learn more about secret scanning, visit our [secret scanning Docs page](#).





Supply chain security uses software composition analysis (SCA) to catch vulnerable dependencies before you introduce them into your codebase. GitHub's proprietary tool is called Dependabot or dependency review. It helps you understand dependency changes and the security impact of these changes at every pull request. It also provides a visualization of your dependency changes with a rich diff on the "Files Changed" tab of a pull request, including information on the license, dependents, and the age of dependencies.

All alerts are powered through GitHub's Advisory Database, which is maintained by a dedicated team of curators. The data has been registered with a Creative Commons license ever since its inception—making it forever free and usable by the community.

> To learn more about supply chain security, visit our [supply chain security Docs page](#).



Dependabot catches dependencies in real time, allowing you to understand:

-  Which dependencies were added, removed, or updated, along with the release dates
-  How many projects use the respective components
-  The vulnerability data for each dependency
-  If your code is making a vulnerable call

Security overview provides a single, centralized view of the security risks in your entire organization. This is especially powerful if you're responsible for hundreds or thousands of repositories. The tool shows the known security risks, as well as unknown risks where security features haven't yet been configured.

With a comprehensive set of filters, you can focus on just the repositories you care about. You can see their risk category, which security features have been enabled, and how many active alerts they have. From there, you can drill into the repository to turn features on or view the alert details to take action.

> To learn more about security overview, visit our [security overview Docs page](#).

MANY DEVELOPERS FEAR SECURITY TECHNOLOGY BECAUSE PAST EXPERIENCES CAUSED FRICTION, DELAY, AND FRUSTRATION. BUT GHAS KEEPS CODE SECURE WITHOUT WEIGHING TEAMS DOWN—INCREASING THEIR SPEED, PRODUCTIVITY, AND SATISFACTION.

GitHub security enables:

- Decrease in fix times thanks to reusable code.³
- Roughly 50% increase in team productivity through automation and better documentation.

³ GitHub, State of the Octoverse, 2021.



Third-party security capabilities through GitHub

Actions give you the freedom and extensibility to automate, customize, and execute your software development workflows in the same place you code. With this functionality, you can use third-party SAST engines, dynamic application security testing (DAST), infrastructure as code scanning (IaC), and container scanning. You can also discover, create, and share Actions to perform any job you wish, like CI/CD—as well as combine Actions into a customized workflow. Individual Actions are reusable as code, making it easy to take advantage of the collective knowledge of millions of other developers.

IV: Conclusion

It's not hyperbole—you can design a security program developers love. GHAS helps you secure your supply chain and code with the only community-driven, native application security testing solution on the market today. Instead of using multiple tools, which cause friction and delays, GHAS offers targeted security context all within the familiar GitHub workflow. This empowers teams to fix vulnerabilities in minutes, not months—so that you can code and innovate quickly with peace of mind.

What is the Ghascompliance Action?

GHAS provides a lot of security information—some more pertinent for certain projects than others. With the Ghascompliance Action, you can set a security policy for Dependabot alerts, license compliance, and code scanning. Each repository can then be easily checked against that policy. This lets your organization define its risk threshold for each alert.



To learn more about GHAS,
connect with our [sales team](#).

WRITTEN BY GITHUB WITH ❤️