GitHub Advanced Security

Proactive vs Reactive Security

For many organizations, security has taken a back seat. The explosive growth of DevOps has allowed teams to build and ship software faster than ever before—but 85% of applications still contain vulnerabilities1. And it isn't for lack of trying. Organizations have poured billions of dollars into security tooling over the past 10 years, yet secure code remains elusive.

This is because teams have been approaching security reactively, not proactively. This paper will explore the shortcomings of a reactive strategy and cover why proactive security is the best way to secure your code and stay competitive.

What's the difference between reactive and proactive security approaches?

A reactive security approach is when teams respond to past and present threats. A proactive approach, on the other hand, prevents issues from happening in the first place. After all, it's much more effective to prevent attacks from happening altogether rather than trying to undo the damage. Of course, there are potential advantages to reactive security—like the rare situation in which a proactive approach in a particular circumstance would be very costly. However, a proactive security approach generally offers a higher degree of control.

And yet, reactive security has been the name of the game. Most security problems take longer than six months to remediate2. Given that teams are under pressure to ship multiple times per day, waiting months to fix security issues is painfully ineffective.

Why does remediation take so long?

- **Number of vulnerabilities.** From 2020 to 2021, vulnerabilities increased by 33%.
- **Lack of automation.** Developers test their code with disparate, third-party apps that are slow, create noise, and are not integrated into their native developer environments.
- **Silos between IT and security.** Historically, IT and security teams have been siloed, with not nearly enough security experts to help fix security issues.
- **Poor orchestration.** Even with remediation tools, it can be challenging for developers to patch across all related systems.

Ultimately, reactive security blocks you from being able to innovate quickly, delight your customers, and meet your business goals.

How a proactive approach can help

Taking a proactive security approach involves empowering your developers. By providing developers with the pre-commit findings and intelligence to quickly fix issues themselves, you won't be reliant on security experts and have to manage the accompanying bottlenecks.

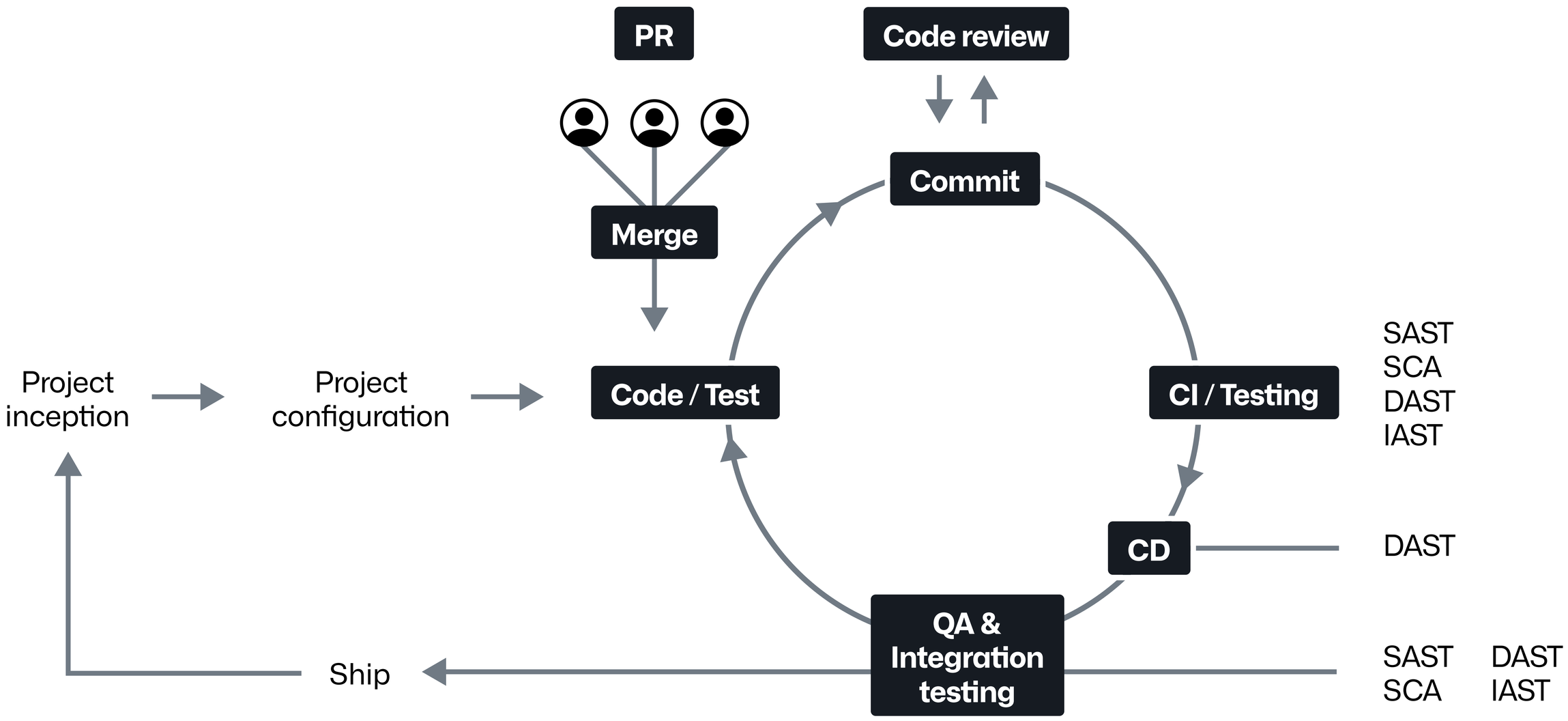
What are the key components of proactive security?

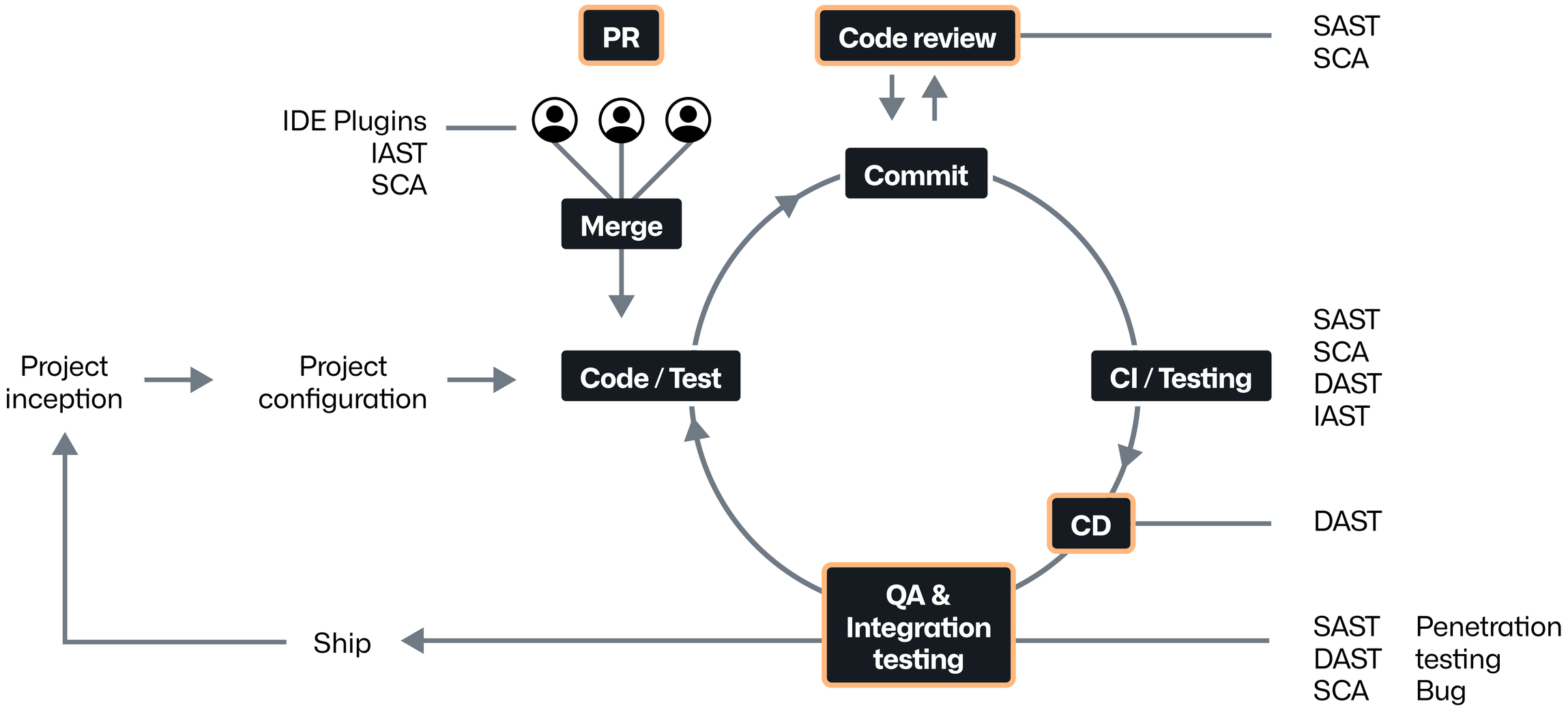
- **Tools embedded directly into the developer workflow** enabling developers to prevent issues from arising in the first place.
- **Visibility into your security posture** across your code, secrets, and supply chain.
- **Security awareness training,** so developers know how to spot signs of attacks.
- **Secret scanning** to detect secrets pre-commit and identify leaked secrets.
- **Penetration testing** to identify, test, and highlight vulnerabilities in your security posture.
- **Proactive endpoint and network monitoring** with technologies like machine learning.

A proactive security approach puts your developers in the driver's seat and empowers them to secure their code, easily and quickly. This is how you can gain a competitive advantage and meet your business objectives.

Proactive security pays off. This approach enables you to:

- **Prevent data breaches** by ensuring that you have all the barriers in place to prevent vulnerabilities from being exploited.
- **Remain in compliance with data protection laws** using the most up-to-date security protections available.
- **Ensure your teams aren't constantly reacting,** which allows them to spend more time building software
- **Find mistakes that might expose private data,** so you can get ahead of the bad actors and recover any lost ground





The GitHub way

GitHub Advanced Security (GHAS) enables developers to proactively fix security issues in real time, as they code. Gone are the days of testing right before the software is about to ship, and having to stop because of a security issue, which is a waste of time and money.

How does GHAS enable a proactive approach to security? It unleashes developer-optimized automation, always-on protection, and a community-powered foundation, which we'll discuss in the next section.

I. Developer-optimized automation

While other solutions' security findings aren't always actionable, understandable, integrated, or trustworthy, GHAS automatically embeds security alerts side-by-side with your code in a way that minimizes friction and drives developer efficiency. This creates a better experience and allows developers to **fix vulnerabilities in minutes, not months.**

GHAS also automates security tasks, allowing developers to automatically test their code at every git push. Plus, they can see security issues in their pull requests as part of the code review process. This prevents security issues from ever making it into the main branch.

With GHAS, developers also have full access to **GitHub Actions**, where they can automate a multitude of tasks. From automatically pulling customer quotes to reusing CI/CD, GHAS takes the busy work out of security and software development—empowering you to meet your shipping goals.

Did you know that 32% of developers spend up to 10 hours a week fixing bugs instead of writing code?

Get your time back with GitHub Advanced Security.

II. Always-on protection

GHAS continually monitors your code and surfaces findings immediately. The solution also features **secret scanning**, which proactively scans for secrets pre-commit. Furthermore, if push protection is enabled, secret scanning searches for 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, GHAS can quickly catch leaks as they occur.

To date, GHAS has detected more than 700,000 secrets across thousands of private repositories.

III. Community-powered foundation

GHAS is built from the community. It relies on crowd-sourced security intelligence from millions of developers and security researchers around the globe. The **GitHub Advisory Database,** which powers GHAS' **Dependabot or dependency review** features, is maintained by a dedicated team of full-time curators and supported by contributions from the entire GitHub community.

These features provide developers with just the right security information at just the right time. Whether the alerts are through the SAST, SCA, or secret scanning, you can be confident that your teams will be given the latest, most cutting-edge security intelligence, along with the cleanest suggestions available for fixing those issues. You'll also be able to customize the platform, so it aligns with your specific business needs. With GHAS, the power of the entire developer community is at your fingertips.

Scale your security team with the world's leading security researchers. Use GHAS to:

- → Leverage the world's expert community and GitHub's millions of developers to increase the accuracy and quality of security detection.
- → Take advantage of open source contributions from GitHub's security coalition, including security researchers from Google, Microsoft, and Uber.

"I can't remember a single security tool where I've heard a developer say, 'I like it, it helped me avoid this mistake.' Usually, security tools feel negative. GitHub Advanced Security actually creates pull requests so that developers can fix issues fast, while the code is still fresh in their minds, and lets them move on to the next thing. It helps our developers learn and improve."

Robert Kugler, Senior Security Manager Cresta

Conclusion

Proactive security can help you go further, faster. By empowering developers to quickly and easily secure their code themselves, you can radically decrease the industry average remediation time of six months to just a few minutes. GitHub's GHAS solution is the only tool that flips the security trajectory from reactive to proactive, with native automation, always-on protection, and a community-powered foundation—enabling your organization to ship secure code faster, innovate more, and increase revenue.

To learn more about GHAS, connect with our **sales team.**

Highest User Adoption
FALL
2022

GitHub's Static Application Security Testing (SAST) software was given the Highest User Adoption recognition by G2.

Written by GitHub with ❤️

1. Osterman Research Report, Uncovering the Presence of Vulnerable Open-Source Components in Commercial Software, 2021.
2. Security Magazine, Security leaders must proactively remediate vulnerabilities to combat modern threats, 2022.
3. IBM, X-Force Threat Intelligence Index 2022.
4. Rollbar, The 2021 State of Software Code Report.

Questions about DevSecOps?
We're here to help.

sales@github.com
github.com/enterprise