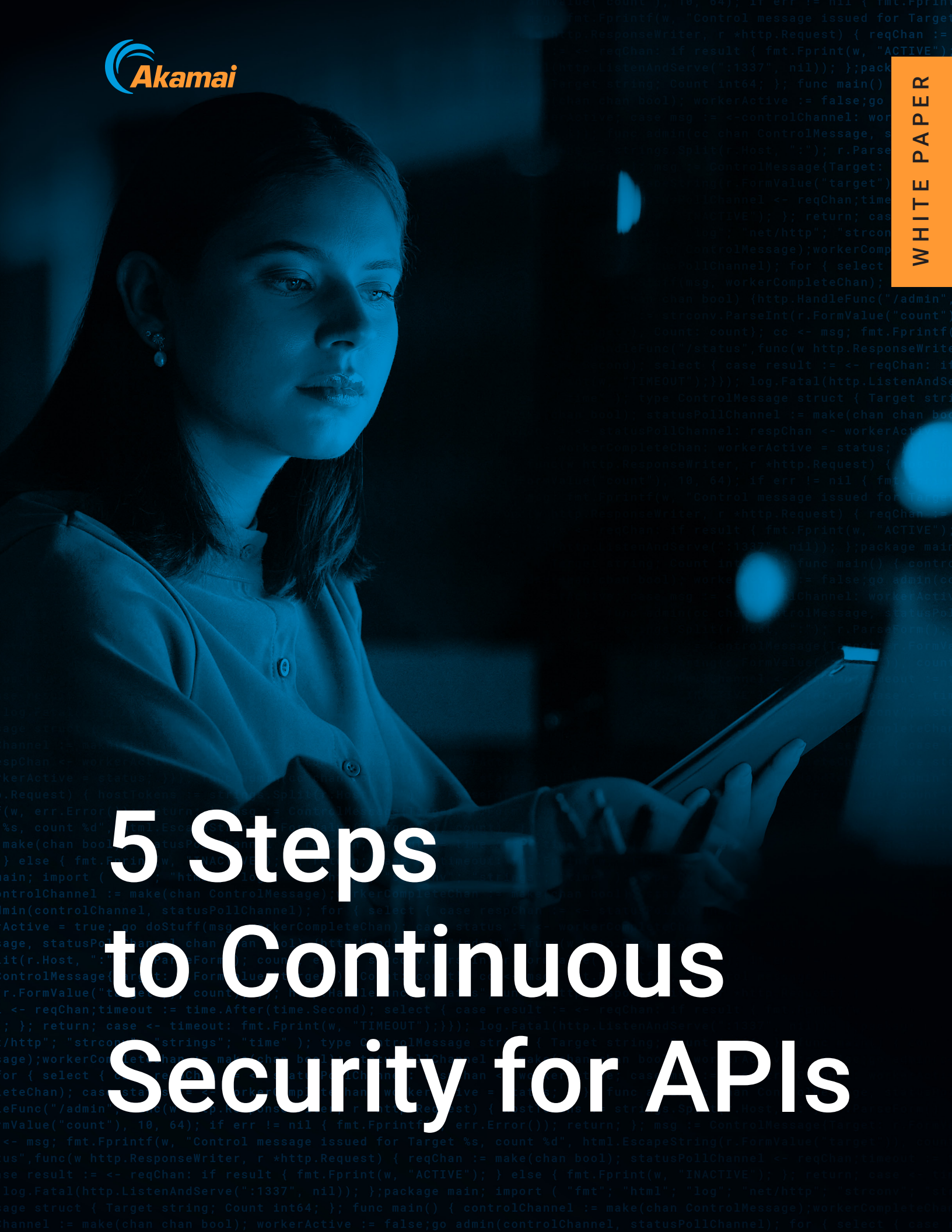
Akamai

WHITE PAPER



5 Steps to Continuous Security for APIs

In this report

The API security challenge

Embedded in every digital or cloud initiative your organization launches, there's a growing ecosystem of application programming interfaces (APIs) that enable revenue-driving innovation. The trouble is: Your APIs have also quickly emerged to become a key vector for attacks.

Threat actors know APIs can provide a fast, direct path to a company's most sensitive data. The vulnerabilities are many: APIs often go into production with misconfigurations, lax authentication controls, and unintended exposure to the internet — all of which an attacker can easily exploit.

Why aren't these API vulnerabilities spotted and remediated before applications are released publicly to end users? Let's investigate the root causes and discuss a way forward.

You may have heard the old adage "Every business is a software business." Today's version is more like "Every business unit across your organization is an independent application developer, racing to meet customer demand." Less catchy for sure, but often the reality.

While traditional rollouts from the centralized IT department still exist, many enterprises are seeing a swirl of innovation led instead by line-of-business initiatives and driven by urgency and commercial objectives versus measured processes. The imperative to act fast and seize market opportunities has eclipsed full consideration of security implications, leading to what some call cybersecurity debt. Developers who aren't in a centralized IT organization (who may even be reporting to a line of business) can quickly spin up new applications, website tools, and generative AI–enhanced services and bypass internal controls. And, in many instances, security teams don't have visibility into those projects and therefore cannot thoroughly assess the risks.

This flux has been the reality for well-known attack vectors for several years, leading industry experts to implore companies to ramp up defenses against ransomware, to secure passwords, and more. However, many companies haven't applied that urgency toward protecting APIs. And that's problematic because APIs are embedded in every application and online service an organization creates, constantly exchanging data and rarely getting adequate protection.

Financial model

Budgets have changed as they've moved from a centralized IT spend to line-of-business operational expenses, but the budget processes have not evolved to reflect the increased security risk. Business units may not fully understand how much to allocate to security, and, as a result, oftentimes nothing is allocated to ensure data is protected.

Cloud migration

Applications are being moved to public and private cloud environments. Moving data and workloads adds complexity, reduces control, and adds third parties to the environment. Organizations need additional security practices to mitigate these risk factors and may not have the skills, experience, or resources to implement them effectively.

Containerization

The move to microservices causes an exponential increase in the attack surface. These instances can instantiate rapidly and then collapse. It's an environment that is difficult to secure because of its highly dynamic nature — and because the legacy tools many organizations rely on were designed for more static environments. This is another case of APIs being omnipresent and full of risk. Today's container- and microservices-based application architecture relies on multiple APIs to function. And even if enterprises with API inventories know the exact number of APIs across their environments, they often don't know which ones return sensitive data.

Agile development

The speed at which developers are expected to roll out new applications, services, and features is a major driver of risk. Development teams meet the deadline pressure head-on with continuous integration and continuous delivery (CI/CD) methodologies and automated functions for development, integration, and testing that enable efficient work.

But who's managing the risk? The move to DevOps means more frequent code changes that are beyond the security team's control. More organizations are transitioning to a shift-left model for developing applications overall. That's a step in the right direction. But that same "test early, test frequently" mindset also needs to be applied to the APIs within the applications — and organizations have substantial catching up to do.

Where should you start? Continuous delivery requires continuous security. Here are five steps for getting started on comprehensive, always-on API protection, as your organization continues to innovate at high speed.

The 5 steps to continuous security for APIs

1. Seed the continuous security culture

Understanding and managing API security isn't an easy task. It requires your leadership team to foster a security culture within your entire organization, but especially across the software development lifecycle. If you've already made gains in building such a security culture, the next step is using it to address the intricacies of APIs and the operational risks they pose. This leads to better visibility, governance, and collaboration.

Here are some practical steps your organization can take to develop and maintain a lasting security culture:

- **Decentralize the security team.** Embed experts inside development groups and product lines to improve visibility and governance. Encourage more flexible policies that rely on the context provided by these embedded experts.
- **Ensure security teams participate in all digital rollouts**, not just through policymaking, but actively from the launch of each service. Line-of-business owners, their teams, and developers should have open lines of communication with security personnel.
- **Designate security champions.** Identify advocates within business units to help build and sustain key relationships for working at speed. Having dedicated security champions will allow you to continuously reinforce the security message and empower cross-functional teams to hold one another accountable.
- **Include everyone.** Security training is imperative — and not just for developers and engineers. Everyone involved in the software development process and beyond should have security training.

2. Assess the API security posture

Many organizations underestimate the size of their API estate. Those who have inventories may not only be missing a large subset of APIs, but also may not know which ones pose the highest risks. Creating a complete and accurate inventory allows you to evaluate the entire API attack surface area.

The following recommendations will help you gain full visibility into your API security posture:

- **Build a full inventory.** Acquire a clear and accurate picture of your organization's potential exposures and what its true surface looks like across APIs and web applications. Use an API discovery tool that comprehensively finds and inventories all APIs, including:
  - Shadow APIs
  - Zombie APIs
  - Dormant APIs

- **Identify each API and its risks.** Understand the types of sensitive data each API interacts with, how it is routed, its associated physical resources, and to which business unit or application it belongs.

- **Examine security team resource allocations.** Analyze the number of APIs each AppSec team member must maintain. Determine whether more tech or more training can maintain or improve the posture.

3. Remediate, automate, and integrate

Enterprises need to understand API access, use, and behavior. However, APIs are complex to analyze. Understanding the API security landscape in all its complexity typically requires processes such as parsing logs, ingesting catalog data, reviewing configurations, testing security, and assessing device configurations. Without the proper tools, remediation can be onerous, either because it is technically challenging or because it requires considerable time and effort. However, remediation can often be automated or semi-automated to eliminate known vulnerabilities and mitigate immediate risk and require little or no human interaction after that.

Here are some pointers to help you prevent attacks and resolve misconfigurations:

- **Integrate with existing IT workflow management systems.** You need to ensure that issues are assigned to appropriate teams as they are identified. Integrations should trigger automation workflows that will address any issues with APIs within the organization.
- **Move into automated remediation in stages.** Initially, rely upon humans to approve new remediation actions before taking them. Ensure coordination with business units to achieve semi-automated remediation. Simply telling developers the code is bad won't suffice. They need actionable insights that they can use. Otherwise you're wasting your time and the developers' time. When issues become known to recur, employ full automation to accelerate remediation.
- **Monitor for malicious behavior.** Use historical knowledge of API exploitation tactics to determine anomalous behavior that may reveal attacker intent. Employ automated or semi-automated responses to mitigate attacks.
- **Integrate with existing security information and event management (SIEM) systems.** This integration assures that the larger team can use API security data.

4. Shift-left API security

When it comes to API development, it's not just a matter of testing but also a matter of *when* you test your APIs. The traditional model places testing closer to the deployment phase, which is vital but insufficient and can lead to serious vulnerabilities. "Shift left" is an approach that moves a variety of tasks to earlier in the development process. With security and testing baked into each step of the API development, a shift-left approach ensures developers will be monitoring for vulnerabilities throughout the API's lifecycle. This allows organizations to accelerate innovation and strengthen their competitive advantage — with API security at the foundation.

Here are a few suggestions that will help you adopt shift-left API testing:

- **Define goals.** Since shift-left demands organizational and cultural change, management should first define goals for the process to ensure any new tool or process introduced into the development cycle will work for the team's existing development and testing methodologies.
- **Understand the supply chain.** Know how and where your organization develops apps and software before architecting a comprehensive shift-left security program. The security risk posture of the supply chain is largely dependent on the security proficiency of others in the chain. This also helps your developers identify where testing might be placed earlier in the lifecycle.
- **Automate security processes.** As development teams launch microservices, ensure that your embedded security experts use API security tools from the start to help monitor risks that occur with containerization.
- **Use consistent tools.** Encourage security teams to adopt the primary interface of the development team and adjust to the tools, workbenches, and language preferred by the developers. For example, vulnerabilities and findings can be entered into the same product backlogs for new application functional requirements as regular user stories.
- **Make the AppSec team a source of innovation.** Leverage continuous delivery principles to develop security-specific microservices that mitigate risks to enable your organization to do what peers cannot.

5. Continually test

As we just established, a shift-left security approach moves testing to the left on the timeline, so the team performs tests earlier in the lifecycle. In contrast, a shift-right approach conducts tests with real users and scenarios that aren't possible in the development environment. Shift-right testing ensures real-world software stability and performance by testing in production environments, and improving user experience by collecting feedback and reviews from application users. The truth is, neither approach is better than the other. To minimize potential risks, an organization needs to continually test.

Your organization can use the following tips to develop and maintain a lasting security culture:

- **Actively conduct API testing.** As part of the API software development lifecycle, API security testing should be used to remediate any potential issues both pre- and post-production. Validate the integrity of each API before and after they are deployed.
- **Continually monitor API traffic.** Track API consumption and analyze API traffic metadata. Real-time traffic analysis identifies new APIs and changes in existing APIs. The analysis process must be automated, repeatable, and actionable.
- **Monitor for vulnerabilities and misconfigurations.** Testing should be continuous, running parallel with development, and involve continuous communication among the clients, developers, and testers. It needs to identify issues so they can be remediated before they can be exploited. Delays in analysis provide additional time for hackers to take advantage of the vulnerabilities. Also important: reporting on changes in policies or functionality and updating SIEM systems.
- **Log API traffic.** Be sure to log API traffic in case there is a need to create forensic reports for specific API keys, tokens, IP addresses, and user identities.

Summary

The API security threat is a real and present danger for many organizations. APIs are often unmanaged, slipping past the radar of traditional tools and living in perpetuity with misconfigurations, missing authentication, and coding errors. As a result, unmanaged APIs are a prime target for attackers and can be compromised without an organization's knowledge.

The solution to this is continuous security — built into developers' processes and into the APIs themselves as they are created and moved into production. Organizations should keep in mind four best practices:

1. Develop a new culture to embed AppSec experts into your organization's engineering team
2. Discover the full inventory of APIs to get a handle on your organization's API security risk profile
3. Prioritize remediations, automate fixes where possible, and seamlessly integrate API security into current application security systems
4. Maintain continuous vigilance and API testing to ensure that new vulnerabilities are rapidly identified and mitigated

Although this approach requires a new mindset, different processes, and cross-team collaboration, it's a challenge that can be conquered.

Learn more about API attack methods, common API vulnerabilities, and how to secure your organization**.**

Learn how we can help you by scheduling a customized Akamai API Security demo**.**

Akamai Security protects the applications that drive your business at every point of interaction, without compromising performance or customer experience. By leveraging the scale of our global platform and its visibility to threats, we partner with you to prevent, detect, and mitigate threats, so you can build brand trust and deliver on your vision. Learn more about Akamai's cloud computing, security, and content delivery solutions at akamai.com and akamai.com/blog, or follow Akamai Technologies on X, formerly known as Twitter, and LinkedIn. Published 10/24.