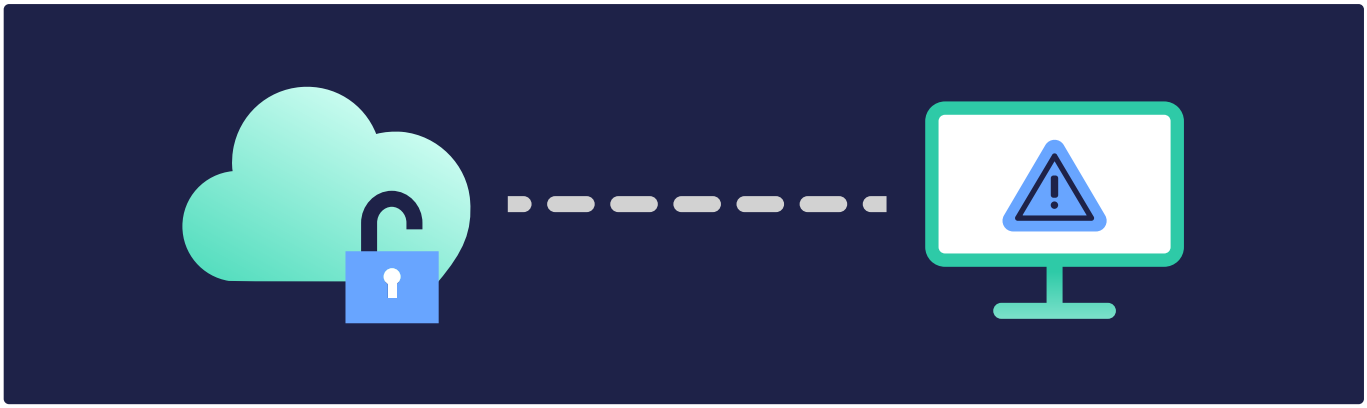
Whitepaper

noname

Continuous Security for APIs

On this Report



The API Security Problem

Enterprises manage thousands of APIs, many of which are not routed through a proxy such as an API Gateway or WAF. Which means they are not monitored, rarely audited, and are most vulnerable to mistakes, misfortune, and mischief. This has left enterprise security teams to play catch-up when it comes to API security. In fact, Gartner predicts that 'by 2025, less than 50% of enterprise APIs will be managed as explosive growth in APIs surpasses the capabilities of API management tools.' Here are some of the key reasons that explain the proliferation of APIs and why many of them are left unsecure:

Digital Transformation

Digital transformation has surfaced immense opportunity and stoked equally intense demand for new capabilities. This has been led by line-of-business initiatives driven by urgency and commercial objectives rather than by traditional IT rollouts. The imperative to act fast and seize market opportunities has eclipsed full consideration of security implications. Developers that aren't in a centralized IT organization, and may even be reporting to a line of business, can quickly spin up new websites and applications, bypassing internal controls. And in many instances, security teams don't have visibility into those projects and therefore cannot thoroughly assess the risks.

Financial Model

In tandem, budgets have changed as they move from a centralized IT spend to LOB operational expense. Organizations are moving fast, 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 then collapse. It's an environment that is significantly harder to secure than monolithic applications because of its highly dynamic nature and legacy tools designed for more static environments.

Agile Development

Timelines have shrunk, with users and customers expecting new features fast. As a result, software leaders are converging around CI/CD methodologies as their strategic accelerator. By adding new functionality iteratively and automating development, integration, and testing tasks, working software gets into the hands of users sooner.

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, in which API security testing is moved earlier in the development process. But full adoption of those practices will take time.

As a result of all these factors, security teams weren't ready. Perhaps they weren't at the table, so they don't have the tools to provide visibility. So now they are playing catch-up – and turning to continuous security as the strategic framework. The reality is, continuous delivery requires continuous security. Here are five steps to embracing the continuous paradigm to protect enterprises as they innovate at high speed.

The Five Steps to Continuous Security for APIs

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 (SDLC). This leads to better visibility, governance, and collaboration.

Here are some practical tips your organization can use 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 upon on-team context provided by these embeds.

Ensure security teams participate in all digital rollouts

Not just through policymaking, but rather 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

Identifying advocates within business units helps to 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 each other accountable.

Security training is imperative

And not just for developers and engineers– essentially everyone involved in the software development process and beyond.

Assess the API Security Posture

Many organizations underestimate the size of their API estate. They often assume the number of applications is equivalent to the number of APIs they have. The reality is, applications have many APIs within them, reflecting an API footprint upwards of 10x what was originally anticipated. Creating an accurate inventory provides an accurate snapshot for you to evaluate the API attack surface area.

The following recommendations will guide you on gaining full visibility into your API security posture:

Get a complete 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 legacy and shadow APIs.

Identify each API and its owner

Understand the types of sensitive data it 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 app sec team member must maintain. Determine whether more tech or more training can maintain or improve the posture.

Remediate, Automate, and Integrate

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

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

Integrate with existing IT workflow management systems

You need to ensure issues are assigned to appropriate teams as they are identified. Integrations should trigger automation workflows for addressing 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, leverage full automation to accelerate remediation.

Monitor for malicious behavior

Use historical knowledge of API exploitation tactics to determine anomalous behavior which may reveal attacker intent. Leverage automated or semi-automated responses to mitigate attacks.

Integrate with existing SIEM systems

This is to assure the larger team can leverage API security data.

Shift Left API Security

When it comes to API development, it's not just a matter of testing but also when you test your APIs. The traditional model places testing closer to the deployment phase. And though this is definitely a vital step, testing only during this time is insufficient and can lead to serious vulnerabilities. Shift Left is an approach of moving a variety of tasks 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 lifecycle. This allows organizations to accelerate innovation and strengthen their competitive advantage.

Below 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 small steps where testing might be placed earlier in the life cycle.

Automate security processes with security automation tools

As dev teams launch microservices, ensure embedded security experts leverage API security tools from the start to help monitor risks arising from containerization.

Use consistent tools

Encourage security teams to adopt the primary interface of the dev team and adjust to the tools, workbenches, and language preferred

by 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 app sec 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.

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 and more often in the life cycle. In contrast, a shift right approach considers testing 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. In order to minimize potential risks, an organization needs to continually test.

Your organization can use the following 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 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. 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.

Monitor for vulnerabilities and misconfigurations

Testing should be continuous, running parallel with development, and involves continuous communication between the clients, developers, and testers. Also be sure to report on changes in policies or functionality and update SIEM systems.

Log API traffic

In case there is a need to create forensic reports for specific API keys, tokens, IP addresses, and user identities, be sure to log API traffic.

Summary

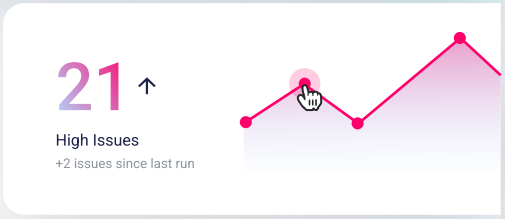
The API Security threat is real and present for many organizations. Gartner research reveals that over half of APIs are unmanaged. As a result, APIs could be misconfigured, have known vulnerabilities, or even be compromised without the organization's knowledge.

The solution to this is continuous security - building best practices into APIs as they are developed and moved into production. First, this requires a new culture to embed appsec experts into the engineering team. Second, it means discovering the full inventory of APIs to get a handle on the API security risk profile. Third, it means prioritizing remediations, automating fixes where possible, and seamlessly integrating API security into current application security systems. Finally, it requires constant vigilance and testing to ensure new vulnerabilities are rapidly identified and mitigated.

While this requires a new mindset, different processes, and cross-team collaboration, it's a challenge that can be addressed. API security may well be the perfect storm for security professionals, but they can navigate the risks with the right approach.

About
Noname Security

Noname Security is the only company taking a complete, proactive approach to API Security. Noname works with 20% of the Fortune 500 and covers the entire API security scope across three pillars — Posture Management, Runtime Security, and API Security Testing. Noname Security is privately held, remote-first with headquarters in Palo Alto, California, and offices in Tel Aviv and Amsterdam.



nonamesecurity.com

info@nonamesecurity.com

+1 (415) 993-7371