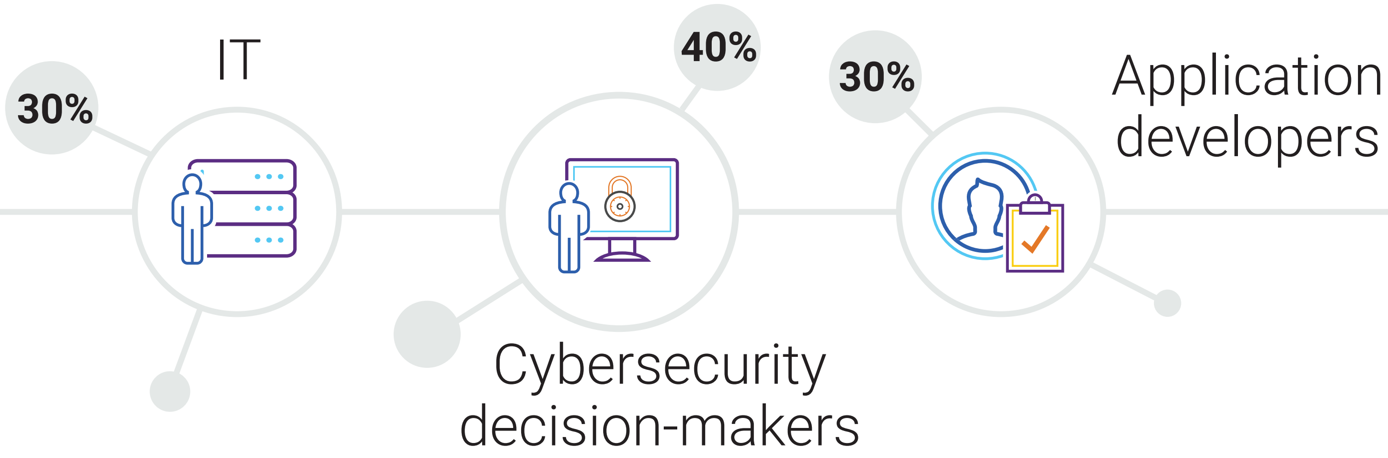
The State of API Security

The software supply chain remains very much top-of-mind for anyone responsible for or affected by application security. Taking this into account, Black Duck cosponsored a report conducted by the Enterprise Strategy Group (ESG), "Walking the Line: GitOps and Shift Left Security." This multiclient developer security research report examines the current state of application security to identify emerging trends across industry verticals and better understand how organizations are struggling and succeeding in today's climate.

ESG surveyed 350 professionals...



...at midsize (100 to 999 employees) and enterprise (1,000 or more employees) organizations in North America.

After examining the survey findings, we identified APIs as a clear topic of concern in respondents' supply chain security efforts. This eBook takes a deeper dive into the API-related findings of the ESG report and offers our analysis and recommendations for what you can do to improve your own API security strategy.

Finding 1

API security is a big concern across verticals

When asked which elements of their cloud-native application stack were most susceptible to compromise and represented the greatest risk to their organization, APIs were most commonly named as the top concern.

Our take

APIs were the most identified security concern in the survey (45%), with data storage repositories and internally developed application source code following slightly behind at 42% and 38% respectively. **Put more simply, respondents were more concerned with the security of their APIs than any other aspect of their application security efforts.** ESG commented on this finding, stating that "the cloud-native cybersecurity threat landscape is intensifying." This supports Black Duck's belief that APIs will continue to be a top concern and continual challenge for those responsible for application security.

When asked, "What are your organization's top investment priorities in cloud-native application security over the next 12 to 18 months?" respondents made clear that they intend to take immediate action to bolster their API security activities. "Discovering and inspecting APIs in source code" was identified as a top priority for 30%, and another 31% named "Applying runtime API security controls" as their main focus. This means that over 60% of security initiative priorities are centered around APIs.

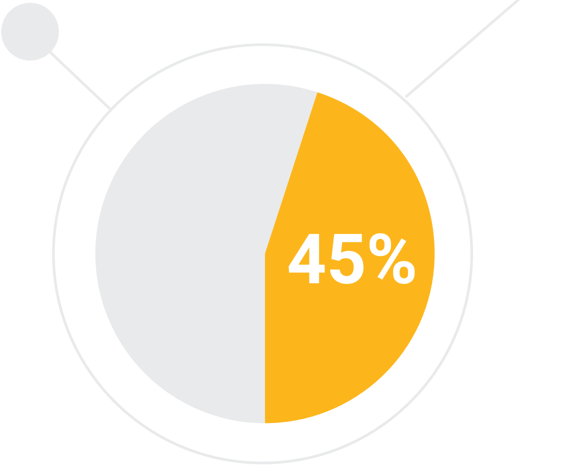
This concern is well-placed. APIs are the primary way that organizations expose their core services over the web. And although this is necessary, it also invites an abundance of opportunities for exploit.

- Careless coding/development practices allow bugs to be introduced during application development.
- The use of open source and other third-party libraries and components can unintentionally invite vulnerabilities into core systems.
- When applications are running in production, bad actors can easily target APIs if they are not adequately protected.

Recommendations

It is important to underscore that web firewalls and monitoring tools are insufficient to protect APIs. Effective API security entails much more than that.

API security should be managed and treated as its own development life cycle. The first step in this process is effective planning, which should involve designing a robust API life cycle management roadmap. This design phase should include proper API policies that are factored into the organization's overall business risk program.



APIs were the most identified security concern in the survey

Having a complete and up-to-date inventory of all API-based apps in your organization can help you control quality, determine API risk classification, and enable effective use of assessment activities. Classification also enables you to focus on the APIs with the highest risk, helping you minimize wasted time and effort.

Perhaps most critical—and often overlooked today—is continuous, real-time API testing and verification. An ideal API security program should be able to dynamically test, verify, and triage continuously. To learn more about managing API security as its own life cycle, read our recent article.

Finding 2

Orgs have had an API security breach recently

When asked whether their organization experienced a cybersecurity incident in the last 12 months related specifically to internally developed cloud-native applications, 38% said they had faced attacks that resulted in the loss of data due to their insecure APIs.

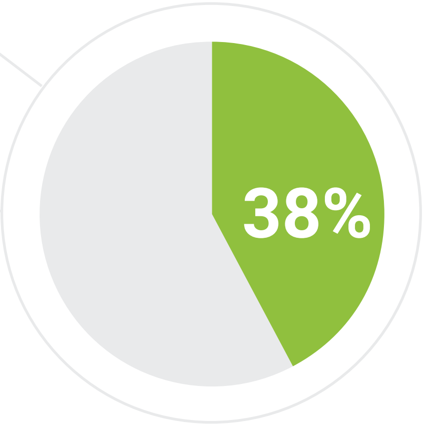
Our take

The majority of respondents noted that their organization had faced a variety of security incidents over the past year that were tied to their internally developed cloud-native applications. The most commonly cited incident type was related to the insecure use of APIs. 38% of respondents have experienced data loss due to improper use of APIs—a rate much higher than any other type of cybersecurity incident.

This finding is not surprising when you consider modern development models. Microservices development frameworks, serverless technology, containers, and so on are all composed of many small pieces of functionality that are developed using a variety of languages, frameworks, and APIs. The result is a patchwork of pieces coming together to create a whole. This means that no one-size-fits-all solution will work for securing APIs. Security programs must be multifaceted and customized to each organization's specific environment and dependencies. In this necessarily complex environment, API security will likely continue to be a pain point—and a priority for security teams.

Recommendations

From a risk management perspective, the effectiveness of an API security program stems from an overarching culture of security, including pointed activities across the entire software development life cycle (SDLC). Although there is no simple solution to solve all API security challenges, there are a few important considerations when working to secure APIs.



38% of respondents have experienced data loss due to improper use of APIs

- **API discovery/inventory.** It is critical that you are able to identify all your endpoints. Having the right tools in place can help you easily discover the APIs used in your applications. After identification, you should focus on cataloging them; a comprehensive catalog of your APIs will make it easy to assess the results of your security scans and establish a risk framework and set of policies.

  But discovery and inventory cannot rely on API documentation alone. Documentation is often incomplete or just wrong. Look for endpoints exposed by the application that were never documented. They may have been forgotten and are far less likely to be secure, especially if they weren't previously on the list of endpoints to assess.
- **API assessment coverage.** While cataloguing APIs and the endpoints within APIs is a great start, this effort is futile if you can't track your assessment of APIs. If you are unable to identify which APIs you tested and how many the endpoints in those APIs were assessed, you are undoubtedly exposing yourself to unnecessary and avoidable risk. Assessment activities should be trackable and verifiable—you must be able to verify that you scanned all your APIs and all their endpoints.
- **API assessment.** The best approach to an API assessment starts with three questions.
  - Which applications or assets are most critical to the business? Focus on these to secure and protect data from unauthorized access.
  - Do you have a way to track the callable APIs and do you test them?
  - Are your APIs internally developed? If so, you can apply more stringent control over their access. Those that are third-party are out of your control.
- **API integration.** Develop and analyze your API policies. Are you using identity and access management tools and encryption? Are you effectively implementing API security into DevOps toolchains? You should make every effort to ensure that the answer to both of these questions is yes.
- **API testing and remediation.** Do you have the means to test, detect, and prevent vulnerable APIs, from deployment to production?

Finding 3

Developers are responsible for API security

When asked who has primary responsibility for discovering and inspecting APIs in source code, 41% said developers, compared to 40% that said the security team has this responsibility. When asked who has primary responsibility for applying runtime API security controls, 44% said that was a developer responsibility.

Our take

Shifting left means teams are performing security efforts earlier and more often in the development life cycle. This has been a key driver in pushing security responsibilities onto the developer. This has not been without challenges; while 68% of respondents named developer enablement as a high priority in their organization, only 34% of security respondents actually felt confident with development teams taking on responsibility for security testing.

While there are obvious benefits of developers being more involved in security activities and processes, there are also obstacles to overcome. The most commonly cited challenges related to developers assuming more security tasks include that developers will be overburdened by these tasks (44%), and that they are underqualified to perform them (42%). Another challenge cited is that these efforts will ultimately make more work for cybersecurity teams (43%).

Traditionally, developers did not have security responsibilities, so this is a new skillset they need to acquire. Most of them do not have a proactive approach to writing secure code and treat it as an afterthought or chore to check off after they have finished development.

With APIs, the responsibility has to be on the developer, as they are responsible for building the API's specifications, functions, and calls. They are also the most qualified to fix them given their intimate knowledge of how an API was constructed. Some believe that trying to fix API issues after development is nearly impossible. We believe that while not impossible, performing fixes and finding design weaknesses after development is certainly not ideal. Early, careful consideration and planning in the design phase, in collaboration with security experts, is therefore critical.

Recommendations

There is clearly a challenge when it comes to developers tackling API security, both in developers' existing skillsets and the confidence security teams have in a developer's ability to perform. Without proper tooling that supports performing security in the design phase of the SDLC, this lack of confidence is likely well-placed.

Traditionally, static application security testing (SAST) was used at this stage, but it is not capable of securing APIs at the speed and scale necessary for most organizations. Additionally, SAST tools do not provide valuable and necessary feedback like real-time remediation guidance and reporting.

There is a debate around whether shifting left or shifting right is the best approach to reducing vulnerabilities in APIs—that is, whether testing earlier during coding activities or later during production is most effective. We believe that instead of testing everywhere all the time, you should perform the right tests, at the right time. Specifically, you should be implementing tooling that addresses vulnerabilities during coding directly in the IDE and CI pipelines, and before developers start writing their code This helps ensure that secure design practices like threat modeling are performed. We believe if your API doesn't have a threat model, it isn't truly secure.

Ultimately, the goal is to make sure you aren't running slow tools that impede your processes. Different tests at different stages in a secure SDLC will provide different information, and performing them all is fine. But that doesn't mean you can run every test, every second—or that you should want to.

Finding 4

Most orgs rely on internally developed API security solutions

When asked what control their organization primarily employs to discover and inspect APIs in source code, 38% said they use an internally developed solution.

Our take

Using internally developed solutions to address API security is an indicator of immaturity in a security program. Complex or custom requirements may also push organizations toward cobbling together their own temporary fix to larger security issues. Findings 1 and 2 make it clear that API security is of great concern, and that it isn't working well for survey respondents. It's not hard to assume, then, that these internally developed solutions are either lackluster or incapable of performing all the functionality that respondents need to successfully secure their APIs. Especially at a larger scale, organizations benefit from enterprise application security tools that can scale with their growth and speed.

Recommendations

With developers at the helm (as indicated by Finding 3), organizations are likely integrating their monitoring solutions with developer-focused security tools in order to help speed remediation efforts. This might aid efficiency as security issues can be remediated without demanding as much time from multiple teams, but these tools likely can't adequately scale. With 45% noting that API security is their #1 concern, and 38% reporting successful attacks in the past year, things don't seem to be working.

However, there are solutions that can provide the necessary coverage.

- **Interactive application security testing (IAST).** Interactive testing is tailored to customer environments based on specific languages (Go, Python, Java, etc.), has low false positives, and provides an agent that runs in the background. A good IAST tool can identify both documented and hidden API endpoints and measure the test coverage of the attack surface.
- **Fuzz testing.** Protocol or fuzz testing detects unknown vulnerabilities in specific communications technologies such as Wi-Fi, Bluetooth, and the Internet of Things (IoT).

How Black Duck can help

Black Duck Seeker® Interactive Analysis tests APIs such as REST, GraphQL, and more. It automatically detects and surfaces all the API routes and endpoints during normal development and QA tests, and also works well in DevOps CI/CD pipelines. It provides real-time alerts as well as visual data flow maps of all the inbound and outbound calls, along with detailed line-of-code insights that help ensure timely developer remediation. The continuous testing and verification provided by Seeker allows DevSecOps teams to react quickly with the least friction to their workflow.

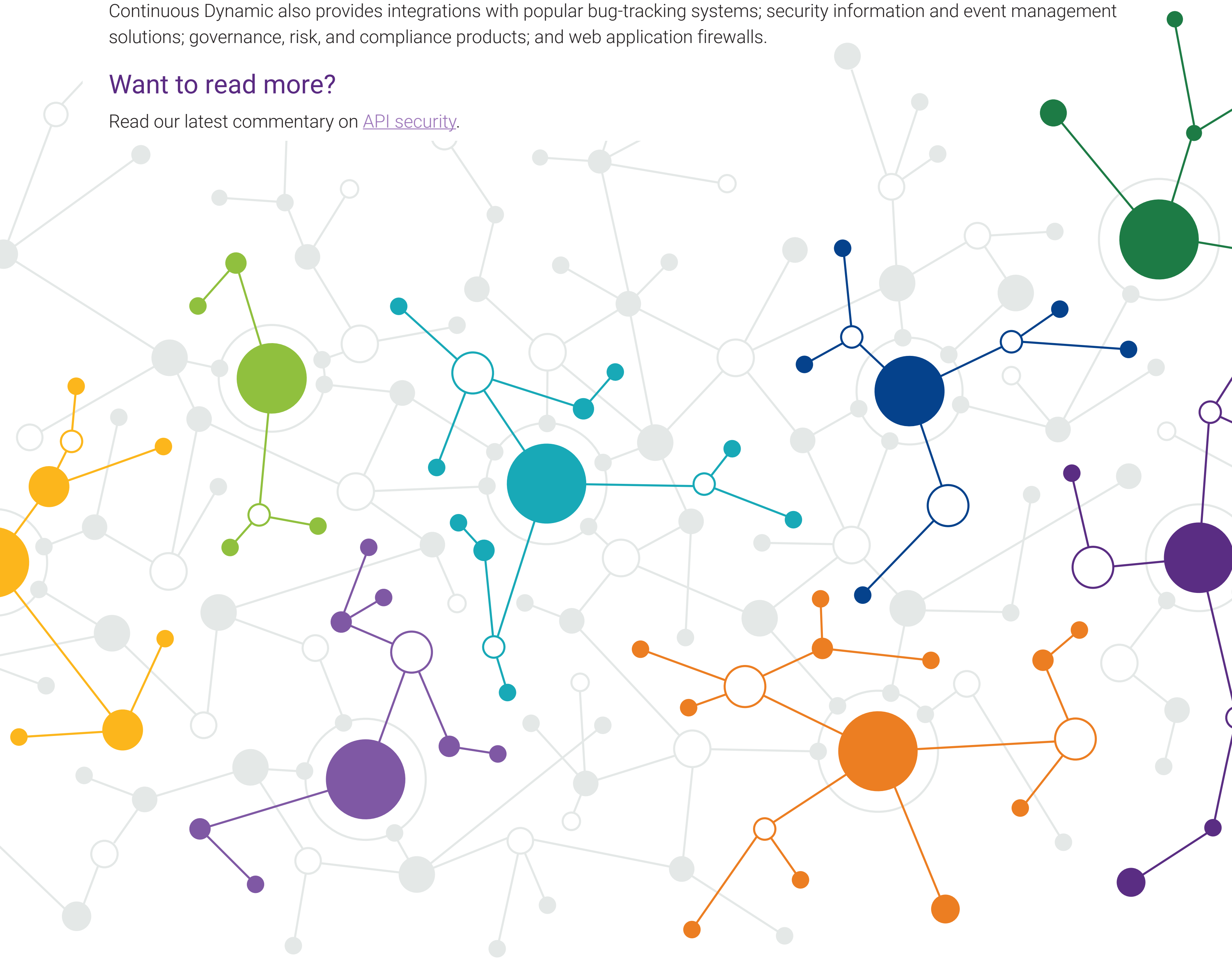
In addition to Seeker, Black Duck offers complete, end-to-end scanning technologies that help secure your cloud-native applications. Code Sight™ IDE Plug-in lightweight SAST empowers developers to instantly detect and fix vulnerable code in their IDE. Coverity® Static Analysis and Black Duck® software composition analysis help secure infrastructure-as-code (IaC), containerized apps, and images.

Black Duck® Continuous Dynamic is a software-as-a-service (SaaS) dynamic application security testing (DAST) solution that allows you to deploy a scalable web security program. No matter how many websites you have or how often they change, Continuous Dynamic can scale to meet any demand. It provides security and development teams with fast, accurate, and continuous vulnerability assessments of applications in QA and production, applying the same techniques hackers use to find weaknesses so that you can remediate them before the bad guys exploit them.

Continuous Dynamic also provides integrations with popular bug-tracking systems; security information and event management solutions; governance, risk, and compliance products; and web application firewalls.

Want to read more?

Read our latest commentary on API security.



About Black Duck

Black Duck® offers the most comprehensive, powerful, and trusted portfolio of application security solutions in the industry. We have an unmatched track record of helping organizations around the world secure their software quickly, integrate security efficiently in their development environments, and safely innovate with new technologies. As the recognized leaders, experts, and innovators in software security, Black Duck has everything you need to build trust in your software. Learn more at www.blackduck.com.