snyk

BUYERS GUIDE

OPTIMIZING APPSEC IN THE FINANCIAL SERVICES SECTOR

A buyer's guide for meeting the unprecedented speed and complexity of today's development practices

Customer expectations for banking and insurance are at an all-time high. Today, 70% of customers expect personalized insights from their financial providers.[1] Financial services teams now face a clear choice: deliver great, customer-focused experiences or risk losing out. As a result, traditional banking models are being disrupted, and even long-established businesses need to innovate to keep up with the competition.

While many financial services development teams are adapting to meet new challenges, their security often isn't keeping up. AppSec teams are still using outdated methods to find and fix vulnerabilities, which don't align with the fast-paced development cycles needed for modern cutting-edge features. Now, with stricter compliance regulations and the continued introduction of novel threats, it's more important than ever for financial services organizations to step up their security efforts.[2]

So, what does it look like to adapt to the changing needs of today's development teams and the financial services industry? This buyer's guide will walk you through the essentials for modernizing application security to fit current development practices. We'll cover:

- The main drivers of increased application security risk
- How to choose AppSec tooling that aligns with modern requirements
- DevSecOps considerations for choosing AppSec solutions

[1] MX: "Consumer Research on Digital and Mobile Banking" (2022)

[2] Dark Reading: "New Cyber Threats to Challenge Financial Services Sector in 2024" (2024)

Chapter 1: The typical development environment today

Today's development pipelines look drastically different than they did even a few years ago. One key difference is the vast array of third-party resources that play some role in the development process. From open source libraries to container images to AI-generated code, today's developers lean far more on resources that originated outside the organization.

Because development processes have evolved so much, FinServ teams must ask an honest question about their existing tooling and techniques: ***Are our current controls keeping pace with new levels of velocity and complexity?*** If the answer isn't yes, they risk being left in the dust as their development teammates race forward to offer customized, feature-rich apps. And as customization increases, so does the collection and management of sensitive customer data, making these applications subject to frameworks like PCI-DSS, GDPR, CCPA, and more. It will only become more and more crucial for security teams to keep pace with their developer counterparts — or else lose out on opportunities due to increased risk and lack of compliance.

Let's dive into a few of the key development areas that have evolved in recent years to better understand how and why they need to be secured:

Open source components

Snyk's State of Open Source Security 2023 report found that a majority of organizations use a developer tool stack comprising at least 50% open source tools. Yet many security teams still struggle to keep up with these development practices. Only 40% of respondents reported using formal security rating tools to check the safety of their open source packages, even though almost all respondents were impacted by one or more supply chain security issues.

In many cases, we see these issues caused by a disconnect between the speed at which developers bring new components into the software supply chain and the abilities of the AppSec team's existing tools and processes. Many rely on a daily or even weekly audit to identify vulnerabilities in the third-party code being brought into the organization. But, the longer security teams wait to check for security issues, the more deeply engrained the component becomes in the application's functionality.

Financial companies in the public sector must comply with Software Bill of Materials (SBOM) requirements, such as those outlined in Executive Order 14028. At the same time, more private-sector businesses are starting to ask their vendors for current SBOMs. To create an accurate SBOM, it's important to know exactly which open source libraries are in your applications and make sure they are secure.

Infrastructure-as-code (IaC)

Many financial services organizations have transitioned from using traditional hardware and data centers to cloud-based IaC. This change has important implications for AppSec teams. Now, infrastructure is no longer managed by operations teams. Instead, development teams are responsible for configuring and managing infrastructure through code, along with their other tasks. As a result, securing infrastructure has become as important as securing any other type of code.

Another implication is that it allows attackers to tamper with infrastructure using software-based vulnerabilities and issues — something that would've been virtually impossible about a decade ago. For most businesses, especially those that house sensitive financial data, overlooking IaC as an essential part of application security can be detrimental.

Cloud-native apps

Cloud-native applications bring increased complexity, making security more challenging. In large financial institutions, teams often work across multiple cloud environments, each using different tools for source control, CI/CD, and container orchestration. For instance, one team may rely on AWS CodeCommit, while another uses GitHub or Google Cloud Source Repositories. This variation in tools adds to the difficulty of ensuring consistent application security across the organization.

With so many different workflows to manage, AppSec teams often struggle to adapt to each development team's needs. The problem gets worse when developers, who are already short on time, are asked to switch environments just to secure their code. But developer adoption is essential for securing applications, especially those handling sensitive financial information. Without their help, security teams can't find and fix critical vulnerabilities — an even bigger challenge when there's an average of 100 developers for every security professional.[3]

Containers/microservices

It's now common practice for developers to deploy applications using a microservices architecture. However, this approach introduces certain risks, such as vulnerabilities in container base images and overly permissive communication between containers, which can increase security exposure.

The complexity of microservices architecture can make it challenging for AppSec teams to manage potential risks effectively. Identifying the root cause of security issues becomes difficult with so many components in motion. Traditional security tools often misidentify the real threats, generating excessive alerts while failing to address the core problem. This information overload can overwhelm both development and security teams, ultimately increasing the risk for customers relying on these applications to manage their banking and finances.

Continuous Integration/Continuous Deployment

Modern development teams — including those building apps for financial institutions — depend heavily on CI/CD pipelines for efficiency. With automation driving these pipelines, unauthorized access to any part of the process can compromise the entire system. However, using traditional, manual security tools in a CI/CD environment is impractical. It forces teams to slow down or stop the seamless flow of development to perform time-consuming audits and checks. In a competitive industry like financial services, where customer expectations are high, development teams can't afford to pause their automated processes for manual security measures.

Generative AI coding

The rise of AI coding assistants has reshaped modern development practices, with in-house developers increasingly relying on these tools to generate first-party code. Most of these assistants are powered by large language models (LLMs), which are trained on a wide range of source code collected from across the web.

This shift introduces two major security challenges. First, development teams at FinServ companies frequently use AI-generated code, which often raises concerns about quality and safety — yet developers tend to trust it more than human-written code. Second, AI coding assistants are producing code at an unmatched speed, significantly increasing the volume of code flowing into repositories.

In short, generative AI coding security teams to "shift left" even more aggressively, as traditional daily or hourly audits can't keep pace with the increased speed of development. Without this shift, these audits risk slowing innovation — critical for staying competitive in the fast-moving FinTech space.

[3] Forbes: "Stop Checking Boxes And Start Effectively Securing Development Pipelines" (2020)

> “FinServ development environments were already some of the most complex in the world, but that trend has only accelerated in recent years,” said Charles Henderson, EVP of Cyber Security Services, Coalfire. “Further complicating the matter, organizations must contend with securing legacy code while also implementing new technologies like generative AI. Snyk’s platform is an essential part of any organization that seeks to keep code remediation efforts ahead of attackers.”

Modern requirements for AppSec tooling

There’s a clear gap between traditional AppSec methods and the requirement demanded of development teams in financial services. Manual processes such as audits and late-stage checkpoints can’t keep up with the required speed, often causing friction between security and development teams.

Automated tools like static application security testing (SAST) and software composition analysis (SCA) can help bridge this gap by streamlining first and third-party code security testing. However, not all SAST and SCA tools are created equal. As your team evaluates AppSec solutions, here are a few recommendations to help ensure your tooling aligns with your business's fast-moving development practices.

Speed and accuracy

Security tools must align with FinServ teams' rapid development cycles and innovation demands. Lengthy delays due to extended scan times or a high volume of false positives can stall progress, delaying crucial updates and features from reaching customers who rely on them.

As AI-generated code becomes increasingly common in repositories, organizations should look for tools that work swiftly and precisely. Since most developers use AI coding assistants to enhance productivity, introducing security measures that slow down workflows or require additional effort can undermine this efficiency.

Full SDLC coverage

As mentioned earlier, organizations typically have multiple pipelines, each with its own set of development tools and processes. This means that SAST and SCA tools need to be flexible enough to adapt to these diverse SDLCs. While many tools offer customization options, choosing solutions that don’t require extensive manual adjustments is important. Otherwise, your security team could end up spending excessive time integrating security tools into each pipeline — a challenging task, especially in a large financial institution with numerous development teams.

Developer-centricity

Developer-centricity is a critical factor to consider when selecting SAST and SCA tools. If your tools don’t align with developers' workflows and allow them to maintain their pace while building high-quality financial experiences, adoption can become a challenge.

A few tenets of developer-centricity include:

- **Minimal context shifting.** Developers work most efficiently when they stay in a flow state — coding, pulling in resources, and testing — all within a single integrated development environment (IDE). If your security tools force them to step outside this environment, it can lead to frustration and slow them down.

- **Adequate educational resources.** Most developers don't have the time or interest to become security experts. They often lack the bandwidth to dive deep into research before addressing a security issue. That's why it's crucial to provide quick, accessible security education and remediation guidance — whether it's an immediate fix for a first-party code vulnerability or a suggestion to swap out an insecure component for a more secure option. Ideally, these resources should be available directly within the IDE.
- **Issue prioritization.** Some security tools lack developer-centricity because they can't prioritize what matters and filter out what doesn't, causing a lot of unnecessary noise. Developers aren't SOC engineers; it's not their job to monitor ongoing issues and decide which ones to prioritize. So, finding a tool that can accurately prioritize vulnerabilities based on actual risk to the business and seamlessly present a fix is essential.

Streamlined implementation with minimal startup time

The value of your chosen SCA and SAST solutions should outweigh the startup costs and time. After all, your company's software development teams will want to invest in competitive financial solutions and minimize resources spent on security controls for these apps. It's become more common for solutions to prioritize a speedy and straightforward implementation, minimizing the learning curve for developers and security practitioners alike. Look for a solution that powers easy setup and requires minimal manual intervention.

Successful DevSecOps for today's FinServ businesses

In addition to the technical requirements, your selected SAST and SCA tools must support the cultural shift toward DevSecOps. The right tools empower teams to modernize their collaboration, moving beyond simple handoffs at checkpoints to true alignment around shared goals. By fostering better communication and continuous cooperation, these technologies help deliver new smart banking features to customers more quickly — without sacrificing security.

As you consider the next steps for modernizing your organization's AppSec tools and techniques, look for solutions that empower you to do the following:

Communicate risk clearly.

The leaders at your FinServ organization will want to see a birds-eye view of risk posture rather than dive into the minutia of your security processes. It's essential to find tools that can measure and meet KPIs that point to overall lowered risk. Look for solutions that clearly align security activities to leadership priorities. The following features point to a risk-focused approach:

Asset visibility: Gaining complete visibility into all assets within your company's purview, including first- and third-party code, containers, IaC, cloud environments, etc.

Software estate coverage: Implementing consistent security controls across more and more of the software estate over time.

Risk-based prioritization: Triaging issues based on the actual business risk they pose rather than a generic score like CVSS.

Remediation action: Taking the next step to remediate pressing issues over time and preventing further issues from entering the SDLC by pushing for secure development from the start.

Drive a “fix culture.”

As you modernize your AppSec strategy, it's essential to ask a direct question: Are our current tools helping us actually resolve issues, or are they merely meeting the bare minimum for compliance? Being compliant doesn't always mean being secure. Protecting your customers' sensitive financial data requires actionable security measures, not just a “check-the-box” approach.

To go beyond compliance, your business must consider solutions that enable you to actionably identify and fix issues. It’s a good sign when a solution has proven value in meeting KPIs associated with a “fix culture,” such as mean-time-to-remediation (MTTR).

Focus on the overlap between AppSec investment and business success.

As we’ve mentioned, innovation in the FinServ industry heavily depends on the speed and quality of application development. Since software velocity directly impacts business bottom lines, there is a clear connection between application security efficiency and financial performance. As such, it’s essential to look into solutions that directly improve velocity — deeply embedding AppSec into development workflows, minimizing developers' cognitive load by streamlining remediation processes, and reducing noise through better, risk-based prioritization.

Snyk’s developer-first approach to AppSec in FinServ

Snyk takes a holistic, risk-based approach to securing applications. We focus on providing visibility, context, and control across your entire application ecosystem. Our approach gives your leaders the full picture they need to understand risk, your security teams the automation and accurate prioritization to focus on the most business-critical issues, and your developer teams the user-friendly tools for finding and fixing vulnerabilities as they build.

We have partnered with a variety of financial institutions to help them reduce application risk at scale, including:

Revolut, an English FinServ company that enabled rapid developer adoption and met updated PCI standards with Snyk Open Source.

Asurion, an American FinServ company that seamlessly integrated Snyk Open Source with its teams’ fast-paced and varied CI/CD pipelines, powering stronger DevSecOps collaboration as a result.

Helvetia, a Swiss FinServ company that enabled its development teams to detect third-party security issues within their software with Snyk Container.

See why Snyk is the chosen AppSec solution for developers and security teams alike — and what it can do for your team.
See Snyk in action.