White Paper

DevSecOps Modernization Through Integration and Complexity Reduction

Sponsored by: JFrog

Jim Mercer
April 2026

EXECUTIVE SUMMARY

Across every industry, organizations are increasingly defined by their ability to build, secure, and deliver software at scale. As the growth of net-new applications continues to skyrocket, these applications will increasingly integrate with AI agents, and IDC forecasts that the total number of AI agents will exceed 1 billion by 2029 (source: IDC's Agent Economics Adoption and Delivery Model, December 2025). Applications now underpin customer experiences, operational processes, data analytics, and emerging AI-driven services and agentic AI workflows. As a result, the speed of software delivery has become a strategic differentiator for enterprises competing in digital markets.

However, the complexity of modern software supply chains has increased significantly. Applications incorporate large volumes of open source components, containerized services, infrastructure configurations, and other third-party dependencies. Managing these components securely requires visibility and control across increasingly complex development environments.

In response, organizations have adopted DevSecOps practices to integrate security directly into development pipelines. However, many enterprises implemented DevSecOps by adding layers of security tools to existing workflows rather than rethinking the underlying architecture of their development platforms.

This IDC research paper examines the challenges organizations face in securing their growing application estate and how they can address them through an integrated DevSecOps platform. It includes a customer case study of a large financial institution that addressed these issues using JFrog Curation and other security capabilities included in the JFrog Software Supply Chain Platform.

April 2026, IDC #US54469926

SITUATION OVERVIEW

Increasing complexity

Modern application architecture has shifted from custom, internally developed code to a heavy reliance on open source frameworks and third-party services. While this allows teams to leverage community innovation and build complex features quickly, it means that much of an application's code base often originates outside the organization. This trend is extending to modern intelligent applications, and IDC data shows that 61% of enterprises favor open-weight models over proprietary ones to address generative AI use cases (source: IDC's *U.S. Open Source Software Survey,* July 2025). These dependencies create a complex software supply chain in which every external dependency introduces potential security vulnerabilities, licensing risks, and operational hurdles that require careful management.

Despite the need for oversight, many organizations struggle to gain visibility into the security and components used within their development pipelines. Because environments often grow organically, different teams frequently use fragmented tools and inconsistent processes rather than a unified DevSecOps platform. This tool sprawl makes it difficult to track how software artifacts move from development to production, leaving many companies unable to secure or govern their application software ecosystems.

Operational impact

The explosion of DevSecOps tools has forced developers into a cycle of constant context switching. In addition to prioritization concerns, investigating a single vulnerability often requires jumping between security scanners, dependency logs, and artifact repositories, each with its own interface. This fragmentation creates cognitive overhead and wastes time on false positives, ultimately slowing down development and frustrating engineering teams.

Beyond productivity losses, these disconnected systems create dangerous security blind spots. Without a centralized, security-integrated way to manage artifacts, organizations struggle to track where vulnerable components are actually deployed. Corporate mergers or team silos often worsen these gaps because a lack of unified tooling makes it nearly impossible to maintain a clear view of the security of the DevOps software pipeline.

Security findings fatigue

Modern security testing tools are essential for finding and identifying vulnerabilities, but when they operate in silos, they often generate a mountain of overlapping or conflicting alerts. This toil leads to "findings fatigue," where a high volume of notifications for the same components overwhelms developers. When critical issues are buried under a sea of low-priority or redundant security findings, the risk increases that organizations will ignore, or miss entirely, truly dangerous vulnerabilities.

Furthermore, many tools fail to provide the context needed to prioritize fixes, such as whether a vulnerability is exploitable in production. Without this clarity, developers must spend valuable time manually investigating each alert rather than building features. This lack of automated intelligence creates a major security bottleneck, draining engineering resources and stalling development.

Software supply chain risk

The software supply chain's growing complexity has expanded the potential attack surface available to malicious actors. Rather than targeting individual applications directly, attackers increasingly focus on upstream dependencies. By compromising widely used libraries, developer tools, and AI models, attackers can potentially introduce malicious code into thousands of downstream applications. High-profile incidents in recent years have demonstrated the effectiveness of this strategy.

Organizations lacking centralized visibility into their software supply chains may require days or weeks to identify affected systems after discovering vulnerabilities. During this time, production environments may remain exposed, putting the entire organization at risk of an attack from a malicious actor.

Preventing vulnerable components from entering development pipelines is significantly more effective than attempting remediation after deployment. Achieving this level of control requires centralized management of software artifacts and dependency ingestion.

Toolchains to DevSecOps platforms

As software supply chains have grown more complex, organizations increasingly recognize the limitations of these fragmented toolchains. Development leaders are beginning to prioritize platform architectures that can manage multiple stages of the software life cycle through integrated capabilities.

Several industry trends are accelerating this transition:

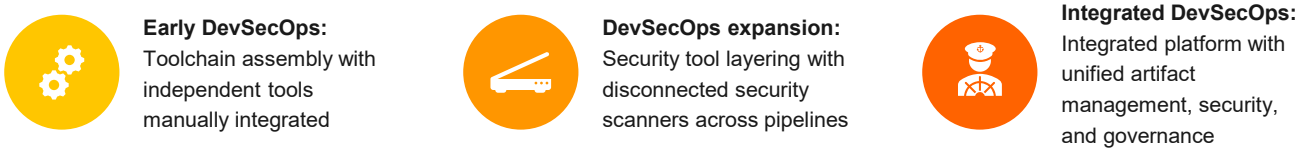
- The rise of software supply chain attacks has highlighted the importance of visibility across the entire development life cycle. According to the 2025 Verizon DBIR report, software supply chain attacks account for 53% of all breaches, nearly double the 27% reported in the previous year. Organizations are seeking platforms that can track the origin of software artifacts from development to deployment.
- Generative and agentic AI tools are dramatically increasing development velocity. IDC's October 2025 *Generative AI Survey* found that 80% of developers reported productivity gains of 20% or more when using AI coding assistants. As code creation accelerates, organizations require automated governance processes that scale alongside development pipelines.

- Platform engineering teams are emerging within enterprises as internal service providers responsible for building standardized development environments. IDC's June 2025 *Platform Engineering and DevOps Survey* indicates that 93% will pilot, use, or expand their use of an internal developer platform for platform engineering over the next 12 months. These teams increasingly rely on integrated DevSecOps platforms as foundational infrastructure.

Figure 1 illustrates the evolving DevSecOps architectural landscape.

FIGURE 1

DevSecOps architectural landscape evolution



Source: IDC, 2026

CASE STUDY

A global financial service company unifies the SDLC to eliminate sprawl

IDC recently spoke with an IT leader at one of the world's largest financial services firms. His team builds and supports a centralized development platform serving roughly 25,000 developers and technologists. The company supports an enterprise with 80,000+ employees operating across more than 40 countries. Every line of code written across that vast landscape, whether for mobile banking, trading platforms, analytics engines, or modernized mainframe systems, ultimately depends on the platform his team provides. And at the center of that platform sits the JFrog Platform.

Life before a unified binary backbone

In the early days of DevOps, the company's toolchains were fragmented, stitched together manually, and heavily reliant on manual processes. These toolchains evolved, moving to multicloud, with different lines of business using different CI/CD tools. A flurry of acquisitions brought different technical stacks, policies, and security products.

From an artifact and binary perspective, the company's world was disjointed. Its application development teams typically built software in one environment and deployed it to another. Artifacts moved through a patchwork of scripts, homegrown distribution mechanisms, and

ticket-driven workflows. Onboarding open source software was a multiday, or even multiweek, process involving ticket queues and manual checks for licensing and security.

That manual approach created tension because developers wanted speed, while compliance teams demanded control to reduce risk. And every acquisition compounded the problem because each new company brought its own tools and software composition analysis (SCA) products. Security vendors often overlapped, but there was no standardization.

The organization could have built its own unified binary distribution fabric, but the IT leader put it plainly: "We're a bank. It doesn't make sense for us to build these things."

JFrog Artifactory

JFrog Artifactory, which the company deployed more than a decade ago, was the first inflection point. It became the enterprise's federated space for binaries, serving as a single distribution backbone spanning on-premises datacenters and multiple cloud providers.

With JFrog's enterprise capabilities, including edge distribution, the company could build software anywhere and deploy it everywhere. That mattered not only for day-to-day engineering but also for integration. When the company acquired new businesses, each adding new developers and technical stacks, JFrog Artifactory provided a common substrate. Even when acquired entities used their own tooling, most were already accustomed to using JFrog Artifactory for binaries, making standardization easier.

Today, even the company's modernized mainframe teams follow a Git-based workflow, publish artifacts through JFrog as a waypoint, and then deploy to mainframe targets. From a pipeline and controls perspective, publishing to the mainframe now resembles publishing a Docker image or a Maven module. Metadata collection is consistent, and risk visibility and control are centralized.

The IT leader stressed that application developers don't need to understand how it works. They publish artifacts according to convention, and JFrog Artifactory handles the replication, distribution, and scale behind the scenes. The result is pipelines that are simpler than they were a decade ago and are now plug-and-play across CI/CD systems, clouds, mobile targets, and legacy platforms.

From JFrog Xray to JFrog Curation: Shifting security left

While JFrog Artifactory unifies distribution and JFrog Xray provides artifact scanning, the recent advent of JFrog Curation has transformed the company's software supply chain security. Before JFrog Xray, discovering vulnerabilities after artifacts had entered the environment was disruptive. Remediation meant hunting down binaries across

environments, tracing dependencies, and reacting under pressure, which was especially painful during industrywide events, such as the Log4j incident.

JFrog Xray introduced systematic scanning and policy enforcement. However, JFrog Curation extended and scaled that model. Instead of managing multiple funnels across environments (e.g., on premises, cloud, and acquisitions), the company now directs everything through a single control point with centrally defined policies. JFrog policies are automated, rule-based governance mechanisms within JFrog Xray and JFrog Curation that define security, license compliance, and operational standards for software artifacts.

With JFrog Curation, the onboarding funnel became authoritative. Application developers no longer need to fetch packages directly from third-party endpoints. All open source modules flow through the JFrog Curation Catalog. The policies enforce license checks, vulnerability thresholds, and even timing rules, such as blocking packages released too recently to be trusted. JFrog Curation applies these policies before pipeline ingestion to ensure they never reach their internal development systems.

The operational shift is transformative, and what was once a ticket-based process that took days or weeks has, in many cases, shrunk to minutes. The IT leader noted that automation doesn't generate headlines the way AI coding assistants do, but the productivity gains when using the JFrog Platform are substantial.

A recent test occurred during a wave of Node Package Manager (npm) zero-day vulnerabilities that affected the organization in late 2025 and early 2026, revealing hundreds of vulnerabilities over a matter of weeks. By leveraging JFrog's capabilities to manage policies in a single place, the firm significantly reduced its application security exposure. The company never onboarded vulnerable packages, eliminating the need for downstream remediation work. The IT leader shared that even a modest reduction in onboarding vulnerable components justifies the investment, but the actual reduction in risk exposure for the affected applications was much greater. JFrog Curation, in effect, moved the control point to ingestion. Preventing a bad component from entering the environment is exponentially easier than extracting it from production.

Standardization through acquisition

The benefits of the tool consolidation enabled by the JFrog Platform were most visible during acquisitions. Each newly acquired company arrived with its own onboarding tools, SCA products, secret scanners, and security vendors. By standardizing on JFrog, the financial services company reduced tool sprawl and licensing costs. More importantly, it established a consistent SDLC story across the combined organization.

The IT leader emphasized that standardization is not merely about efficiency; it's about risk. In a regulated industry, inconsistent pipelines create inconsistent controls. JFrog became a

cornerstone in enforcing a uniform approach across development, infrastructure as code, containers, and even modernized mainframe workflows.

The firm's platform team retains the flexibility to plug-in other solutions where it makes sense, but the JFrog Platform serves as a foundation seated within the binary flow, centralizing security capabilities and providing efficiency and security.

Velocity without compromise

The organization has tracked DORA metrics for years, and deployment frequency, throughput, and change failure rates have all trended in the right direction through using the JFrog Platform, even before AI coding assistants entered the picture. While causality is hard to assign to any single tool, the IT leader was clear that without a scalable, reliable binary distribution backbone, those gains would not have been possible. Despite significant increases in build and release volume over the past two years, the binary substrate has scaled up to meet the challenge.

The application developers often don't realize what the JFrog Platform is doing for them. For example, when they run an npm install, JFrog Curation ensures compliant packages are delivered without application developers needing to think about security. They push code through a standardized pipeline, and JFrog Artifactory manages distribution. The plumbing is invisible but essential, and the IT leader stressed that without it, moving fast and safely would be impossible.

The IT leader noted that AI may accelerate code creation, but without a reliable binary platform, it cannot quicken business outcomes. Eventually, those bits must run in production, and they will pass through the JFrog Platform.

The company is also exploring JFrog's ML and AI catalog capabilities. It is not yet using these tools at scale, but there is clear interest in model governance, detecting model API calls in binaries, and life-cycle management.

Business impact

Across the enterprise, JFrog has provided the following benefits:

- Reduced tool sprawl and licensing costs through consolidation
- Standardized onboarding and security policies across acquired companies
- Significantly reduced exposure to zero-day vulnerabilities via JFrog Curation
- Streamlined pipelines across cloud, on-premises, mobile, and mainframe environments
- Improved DORA metrics and developer satisfaction
- Enabled safe use of the open source ecosystem and innovation at scale

Perhaps most importantly, it has allowed the organization to scale release velocity without increasing head count.

While JFrog may not be the tool developers talk about in stand-ups, it is one of two "sticky" platforms, alongside GitHub, that the organization views as foundational.

Customer challenges

The global financial service company faced several notable challenges with standardizing its tools and processes:

- **Tool sprawl and a lack of standardization:** Acquired companies introduced multiple DevOps tools, different onboarding and security solutions, and inconsistent pipelines and policies. Sprawl created operational inefficiencies, increased licensing costs, and made governance enforcement challenging.
  - **Solution:** The company standardized on the JFrog Platform as the common backbone, using JFrog Artifactory for binaries and JFrog Xray coupled with JFrog Curation for security and governance. The organization migrated acquired companies onto a consistent pipeline model, reducing tool sprawl, enabling faster onboarding, and creating a uniform pipeline and policy framework.
- **Scaling JFrog to enterprise complexity:** Implementing the JFrog Platform required several customizations, with some iterative tuning for performance and distribution.
  - **Solution:** The company received strong support from JFrog and established a tight partnership with JFrog engineering team. This helped it improve its enterprise JFrog Artifactory configuration, edge distributions, and performance. The IT leader indicated that JFrog Platform can now run with a relatively lean ops team that requires little administrative work to keep running.
- **Normalizing ways of working across different teams and acquisition cultures:** Post-acquisition environments had different pipelines, onboarding, and security tools. Even with JFrog in use internally, rolling it out consistently was difficult. Cultural change, process alignment, and migration efforts were required.
  - **Solution:** The company used the JFrog Platform as the anchor for software changes and security. The platform helped enforce common open source curation practices, unify binary management, and provide centralized policy management. The result was the integration of the acquired companies' development environments, leading to a uniform pipeline and processes across the organization.

Integrated DevSecOps

Integrated DevSecOps platforms solve the headache of fragmented tools by consolidating everything into a unified architecture. At the core is centralized artifact management, which

acts as the official system of record for all software binaries. This foundation allows organizations to track exactly how components move from the initial code phase through to final deployment, providing the visibility needed to manage a complex software supply chain.

By embedding security scanning and policy-based ingestion directly into this workflow, companies can catch vulnerabilities before they enter the development pipeline. Instead of developers pulling unchecked files from public sources, the platform automatically enforces compliance and security rules. This integration ensures that software is distributed securely across environments, turning a chaotic mix of tools into a streamlined, high-speed delivery engine.

ADVICE FOR TECHNOLOGY BUYERS

Organizations seeking to modernize their DevSecOps environments should prioritize platform architectures that can manage the software supply chain end to end and provide:

- Centralized artifact management that enables organizations to track software artifacts as they move through development pipelines and deployment environments
- Security scanning capabilities that integrate directly into development workflows to detect vulnerabilities early without introducing delays
- Reduced cognitive load for developers, minimizing context switching and reducing friction within the DevOps pipeline
- Coverage across the DevOps pipeline, enabling the consolidation of DevSecOps tools into an integrated platform that can manage multiple stages of the software life cycle

FUTURE OUTLOOK

Rapid advancements in AI-assisted coding tools and increasingly complex software supply chains will shape the next phase of software development. These technologies will dramatically increase the volume of software artifacts entering development pipelines and introduce new artifacts, such as AI models and agentic AI skills.

Organizations will need automated governance and security processes that scale alongside development velocity. Platform-based DevSecOps architectures will become foundational infrastructure for modern software development.

CONCLUSION

DevSecOps tool sprawl has emerged as one of the most significant barriers to efficient and secure software development. Fragmented toolchains create developer inefficiencies, increase security risk, and slow innovation. Organizations are increasingly recognizing that they cannot solve these challenges simply by adding more tools.

Instead, enterprises should evaluate platform-based DevSecOps architectures that unify artifact management, security scanning, and governance. Platforms designed to manage the software supply chain from development to deployment, or from prompt to production, can simplify pipelines, strengthen the security posture, and improve developer productivity.

As software supply chains grow more complex and AI accelerates development velocity, the importance of unified DevSecOps platforms will only increase. Organizations that invest in these platforms today will be better positioned to deliver secure, reliable software at the speed of modern digital enterprises.

ABOUT IDC

International Data Corporation (IDC) is the premier global market intelligence, data, and events provider for the information technology, telecommunications, and consumer technology markets. With more than 1,300 analysts worldwide, IDC offers global, regional, and local expertise on technology and industry opportunities and trends in over 110 countries. IDC's analysis and insight help IT professionals, business executives, and the investment community make fact-based technology decisions and achieve their key business objectives.

Global headquarters

One Beacon Street
Suite 33100
Boston, MA 02108
USA
508.872.8200
X: @IDC
blogs.idc.com
www.idc.com