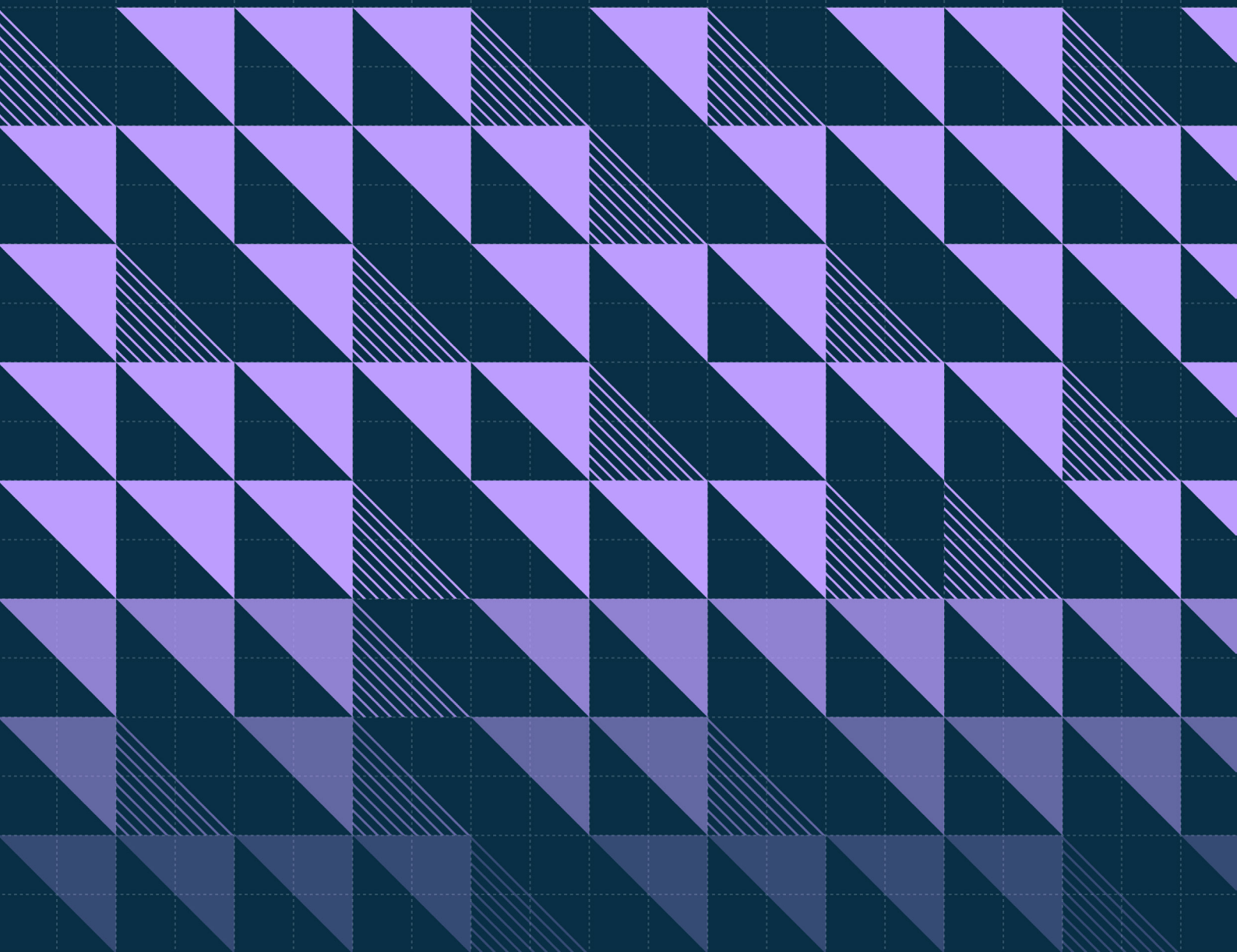
cloudsmith

Artifact Management Report 2026



Contents

03 Executive summary by Glenn Weinstein

05 Key findings

06 Modern software development creates an enforcement gap for security and compliance

10 The impact of AI on the software supply chain

14 Infrastructure limitations increase operational burden

17 2026 strategy: From complexity to consolidation

19 Appendix

Executive summary by Glenn Weinstein

Our second annual Artifact Management Report arrives at a huge inflection point in the history of software development. We write these reports to understand the way this world is changing, and it's changing fast. In a matter of months, we've gone from 'how can AI help me write better code' to 'how can I help AI write better code.'

I love the craft of programming. I grew up with it, and take pride in my ability to solve tricky problems and write beautiful code. We're all learning together to take that sense of craftsmanship, and apply it to new and powerful tools that let us build more software, build it faster, and ultimately build it better.

This year's survey tells us these powerful AI tools are expanding the attack surface, with more dependencies, more hallucinated packages, and inevitably, more CVEs. Security that worked at human speed is buckling under the pace of AI. The underlying artifact management infrastructure powering CI/CD pipelines across distributed teams is struggling to keep up. Companies are running legacy systems that were never designed for this threat environment.

So scale and security problems are exacerbated by generative AI coding tools. But as a community, we still want to use them, because they massively increase productivity, and open up new and incredible ways of developing software applications. We just want to know that our developers can use them safely. Artifact management has a big part to play here. Let's start with some baseline assumptions regarding the industry-wide transition to AI-generated code:

Software re-use is more important than ever. LLM costs are heavily subsidized today, but the costs are real, and it makes no sense to re-write the same libraries millions of times over.

AI isn't perfect. AI-generated code has bugs and needs iteration to get it right. Open-source libraries and frameworks are battle-tested over years, and there's real value in their robustness and stability.

The bad guys are using AI too. Our existing global collection of shared software — the millions of open-source libraries and other artifacts — is a ripe target. Even battle-tested software likely has latent vulnerabilities no one has discovered yet. But AI can be a very persistent bug hunter. Expect a ton more CVEs to be coming our way soon.

Platforms like Cloudsmith can help developers select the 'best' packages — the most stable, the most commonly-used, the best versions to satisfy dependencies — but developers don't always take advantage of this service at coding time, so problems show up later, at PR merge time. AI agents are more compliant. If we tell them 'check with Cloudsmith to make sure we're using the best package,' they'll do it.

Software architects using AI tools will write a lot more software. We're seeing a massive acceleration in velocity.

So building software on top of other software – the essence of the open-source trend of the past two decades – is here to stay. Automated guardrails and context are the new keys to unlock the production of safer, more efficient code. We have an incredible opportunity to change how the profession of software programming works. We all know that the world needs better software. Finally, we're getting the tools that may allow us to deliver on that promise.

Glenn Weinstein
CEO, Cloudsmith

Key findings

01

There is a gap between vulnerability detection and policy enforcement

Many organizations now generate security data and software bills of materials (SBOMs), but there is a growing 'enforcement gap' – the time between vulnerability disclosure and automated policy application. The EU Cyber Resilience Act (CRA) imposes strict reporting mandates later this year. With this deadline in mind, we explore why scanning for vulnerabilities is not sufficient. This report identifies a critical disconnect: while many organizations have visibility into composition and security data for their software, only 25% are actually using that data to automate security gatekeeping.

02

More AI demands more governance

93% of organizations utilize AI to accelerate development, creating two distinct security challenges. First, AI-generated code frequently introduces unverified dependencies, including 'slopsquatting' and hallucinated dependencies, which expand the attack surface. Second, as AI models become a primary artifact type, only 12% of organizations manage these models using the same security policies and provenance tracking as traditional binaries – like language packages and operating system libraries. To secure the software supply chain, organizations must implement a unified control plane that automates the validation of AI-generated code and applies consistent governance to AI models.

03

Outdated infrastructure hurts performance

Software development in the age of AI sits on a fragile foundation of complex infrastructure and institutional inertia. 83% of organizations run outdated artifact management systems, often because of the perception that upgrading is risky or painful, leaving them vulnerable to outages and performance issues, especially for distributed teams. We look at the 'operational tax' teams pay in manual toil and explain why organizations should consider trading legacy technical debt for cloud-native, automated topologies to ensure long-term delivery resilience.

Current estimates suggest that

90%

of software is composed of open source software (OSS) components that developers assemble.

Modern software development creates an enforcement gap for security and compliance

AI now often generates the code that connects those pieces together. Both OSS and AI have their own inherent risks. The average application has over 1,200 dependencies and each of these dependencies represents a software artifact built by someone else. This fragmented ownership can create security gaps in the development process.

Using AI in software development may compound these risks by accelerating development, often without the full context or the protective guardrails inherent in manual reviews. When AI coding agents write more software, they tend to pull in more dependencies, expanding the unmanaged attack surface.

Our **2025 artifact management report** last year anticipated a surge in software supply chain attacks. This prediction seems to have come true. Notable 2025 attacks on the npm ecosystem included **s1ngularity,** the chalk and debug packages, and multiple versions of the **Shai-Hulud** worm. Data show that 44% experienced a major or minor dependency-related security incident, and another 37% had a near miss in the past 12 months.

As attacks on the software supply chain grow more frequent, organizations are looking to shift from reactive remediation to proactive gatekeeping. Reactive security identifies threats already inside the customer's environment; proactive security prevents these threats from entering. To proactively secure their software supply chain, teams generally need to automate scanning and quarantine at ingestion time, and may consider using approaches like package age policies to mitigate zero-day risks before dependencies reach the codebase.

Most teams have visibility into their software supply chain; 79% of respondents can identify whether a specific vulnerability affects their environment in six hours or less after disclosure. However, visibility doesn't always translate into action.

The enforcement gap: Operationalizing security data

More mature risk management programs tend to automate threat mitigation before artifacts reach the development environment. This automation typically requires security policies that block vulnerabilities (as identified in CVEs) and malicious packages based on real-time data. Ideally, these policies can also be used to evaluate packages that have gone into previous builds when a new vulnerability is discovered, to protect applications that are already in production.

Achieving this automation can be difficult for enterprises. Tooling fragmentation and multi-source package ingestion — such as from public registries — hinder threat identification and policy enforcement.

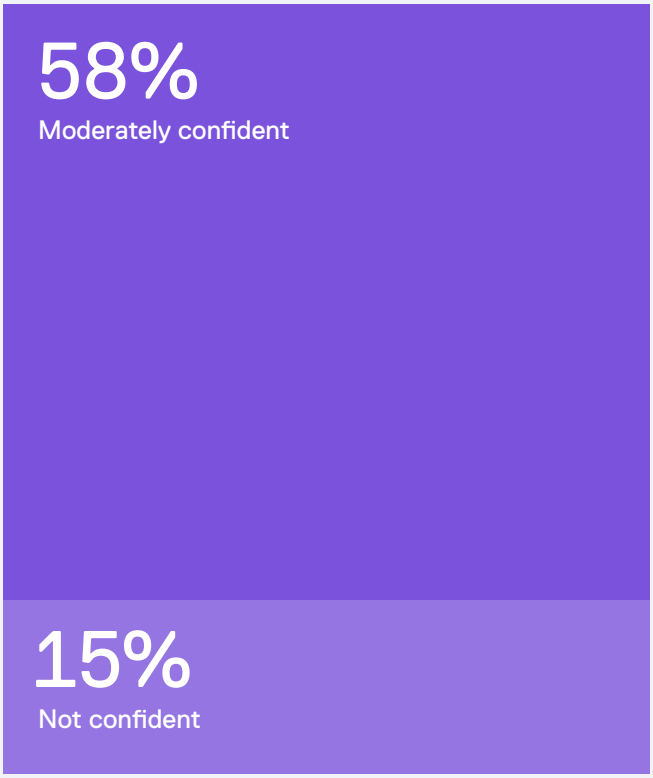
Organizations face an enforcement gap between identifying threats and stopping them. Manual remediation efforts slow threat response and impede efficient software delivery. When encountering a new CVE or malicious package, organizations must assess exploitability, identify downstream dependencies, and define remediation paths.

Organizations recognize the visibility gaps in their existing solutions. When asked to rate their confidence in their current artifact management tooling to prevent a breach of their internal environment 58% of respondents are 'moderately confident' in their current tooling, citing visibility gaps. An additional 15% report that fragmented or manual tools cannot withstand sophisticated software supply chain attacks.

Following a breach, 37% of organizations can automatically identify, block, and trace the blast radius of a compromised artifact within minutes. 48% are able to identify the presence of the artifact, but rely on manual effort to quarantine and remediate it. 12% can identify the artifact in some repositories, but lack visibility into where it is deployed.

At the enterprise level, manual investigation is hard to scale, especially as AI increases the speed and volume of software development. Automation can serve as a key safety lever to ensure software supply chain security.

RATE YOUR CONFIDENCE IN CURRENT ARTIFACT MANAGEMENT TOOLING TO PREVENT A BREACH



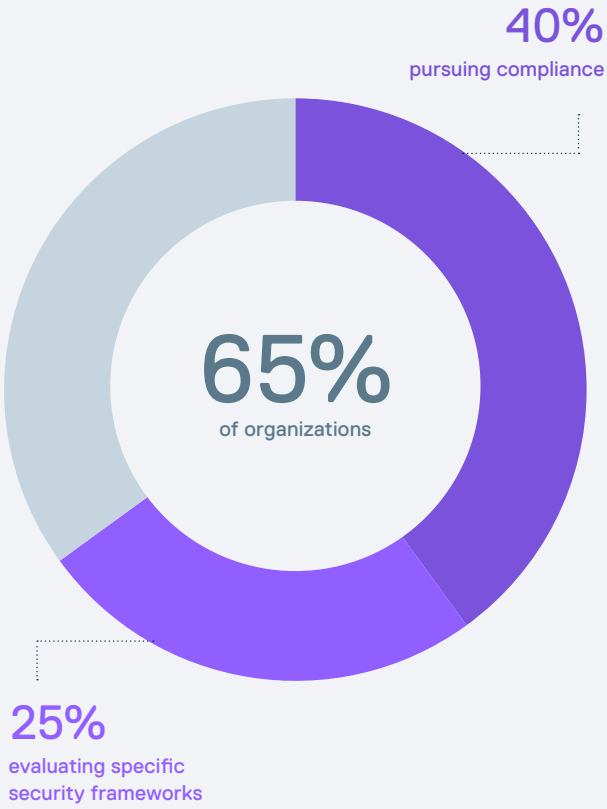
FOLLOWING A BREACH



Compliance: Scaling security through regulatory requirements

Frameworks like ISO 27001, SOC 2, and GDPR mandate risk reporting and threat mitigation. For technology vendors, software supply chain attacks cause service disruptions, data leaks, and outages. Maintaining compliance, reducing the potential for fines, and upholding marketplace trust all rely on securing the software supply chain.

Compliance requires significant resources for license checks, SBOM generation, and provenance tracking. Automating these tasks can help organizations to meet requirements efficiently. 65% of organizations are pursuing compliance (40%) or evaluating specific security frameworks (25%).



The EU Cyber Resilience Act enters enforcement in September 2026. The CRA mandates a 'secure by design' posture, requiring producers to report active vulnerabilities within strict windows.

Like GDPR, CRA applies to **any** company selling products with digital elements (hardware or software) in the EU.

Table 01: EU Cyber Resilience Act reporting windows

CRA REPORTING STAGE	REQUIRED TIMELINE
Early warning (Upon discovery of exploited vulnerability)	24 Hours
Full notification (Following incident discovery)	72 Hours
Final report (After corrective measures are available)	14 Days

Generating compliance reports requires provenance data, which details the history of artifacts and where they're used, and rapid execution.

For example, an SBOM provides critical information about package dependencies and visibility into how and where an application uses packages. Operationalizing SBOM data means proactively generating SBOMs and then using that data to trace packages through the software supply chain. When an attack occurs, having this data accelerates containment and remediation efforts because teams can quickly identify impacted areas and quarantine risky packages to prevent deployment.

While 95% of respondents generate SBOM data, 1 in 4 engineering teams actually integrate and automate SBOM verification into their security gatekeeping.

For the remaining 75%, SBOMs are static artifacts for ad-hoc compliance rather than real-time governance.

This lack of operationalized data forces many organizations to rely on manual effort. Manual compliance does not scale under the CRA's reporting timelines.

27% of respondents are 'very confident' they could pass an unexpected software supply chain audit

74% lack the visibility to produce reports quickly.

The CRA raises the stakes for unprepared organizations, with strict reporting windows and substantial financial penalties for non-compliance. Teams that already struggle to quickly produce audit data manually, will face increased pressure under the CRA's accelerated timelines. Automation accelerates the data discovery process when speed is a factor. Operationalizing artifact data ensures compliance information is available and consistent, replacing manual investigation with automated discovery.

An overview of the current compliance landscape:

Table 02: Common security standards and frameworks surveyed

FRAMEWORK	WHAT IT IS	MAJOR INDUSTRY VERTICALS	GEOGRAPHIC DOMAIN	RELEVANCE TO ARTIFACT MANAGEMENT
ISO 27001	An international standard for managing Information Security Management Systems (ISMS). Focuses on risk management and continuous improvement.	All (Global standard)	Global	Ensures artifacts are stored in environments with strict access controls, logging, and encryption.
SOC 2	Evaluates a service organization's controls based on 'Trust Services Criteria' (security, availability, processing integrity, confidentiality, and privacy).	SaaS, Cloud Service Providers	Primarily North America	Requires proof of 'integrity' for artifacts—ensuring that what you build is what you deploy without unauthorized tampering.
GDPR	Legal framework for data protection and privacy. Requires organizations to protect the personal data of individuals.	All (handling EU data)	European Union (Extra-territorial)	Relevant if artifacts (like database backups or logs) contain Personally Identifiable Information (PII).
Cyber Resilience Act (CRA)	EU regulation establishing cybersecurity requirements for products with digital elements (hardware/software).	Software/ Hardware Manufacturers	European Union (Extra-territorial)	Mandates an **SBOM** for artifacts to ensure vulnerability transparency.
HIPAA	Regulates the protection of Protected Health Information (PHI) to ensure data privacy and security.	Healthcare, HealthTech	United States	Requires strict auditing and access logs for any artifact repository containing medical data or applications processing PHI.
Supply-chain Levels for Software Artifacts (SLSA)	A security framework (not a law) to prevent tampering, improve integrity, and secure packages and infrastructure.	Software Development, DevOps	Global	**The gold standard for artifact management.** It focuses on generating 'provenance' (metadata showing how an artifact was built).
Digital Operational Resilience Act (DORA)	Ensures financial entities can withstand and recover from ICT-related disruptions.	Finance, Banking, Insurance	European Union	Requires 'ICT risk management,' meaning artifact repositories must be highly available and resilient to supply chain attacks.
NIST Secure Software Development Framework (SSDF)	A set of fundamental, sound, and secure software development practices based on established NIST standards.	Government, Public Sector	United States	Focuses on protecting the integrity of all software components and verifying third-party artifacts before use.
PCI-DSS	Security standards to ensure all companies that accept, process, store, or transmit credit card information maintain a secure environment.	Retail, E-commerce, Finance	Global	Requires 'File Integrity Monitoring' (FIM) to ensure that payment-processing artifacts haven't been modified by attackers.

The impact of AI on the software supply chain

In 2026, autonomous AI agents build and deploy entire applications rather than just suggesting code. As AI agents accelerate development cycles, machine-generated code and third-party dependencies increase in volume. For engineering teams, the challenge is no longer just writing code, but managing and securing the massive surge of artifacts these agents produce.

AI-generated code and trust-verify concerns

Software development requires organizations to balance development speed and security. A 'trust then verify' approach favors speed and assumes that packages are safe until a CVE emerges. This enables developers to build and ship software fast, and shifts security to the right. However, identifying threats late in the cycle increases remediation time.

In 2025, we found that 42% of developers reported that AI-generated code made up at least half of their codebase.

By 2026, AI-generated code has become near-universal, with 93% of organizations reporting its use.

AI-generated code requires validation because slopsquatting and an AI-fueled software supply chain create more attack surfaces, from hallucinating malicious and vulnerable packages to dangerous dependencies. The increasing reliance on AI-generated code introduces significant risk, making the 'trust, then verify' approach increasingly unsafe.

Confidence in AI-generated code

When asked about their confidence in AI code to not introduce security vulnerabilities, 16% of respondents admit they are 'not confident,' yet continue to use it for speed while acknowledging it introduces significant technical debt and security noise. 61% are 'moderately confident in AI-generated code. Only 17% of organizations automate scanning and governance to mitigate AI-specific risks, such as serialization or pickling issues.

A likely contributor to this overall lack of confidence is that AI models lack essential context for an organization's specific environment and risk tolerance. The speed afforded by AI comes with a hidden cost: 58% of teams now spend between 11 and 40+ hours validating and securing AI code monthly. As dependency on AI-generated code continues to increase, organizations face a growing challenge in their ability to verify it.

In short, AI-authored output needs review because while it may be functionally correct, there is no guarantee that it is secure. AI platforms and the organizations utilizing them are already implementing automated security reviews to address these risks. This collective focus confirms that while tools are improving, functional correctness is not a substitute for security.

From ‘trust then verify’ to ‘verify then trust’

Organizations must adopt a ‘verify then trust’ posture. This requires proactive governance policies that shift enforcement left. By identifying threats and risks in AI-generated code and enforcing governance policies earlier in the development process, developers can maintain their development speed without sacrificing security.

The ‘verify then trust’ posture helps preserve developer productivity, so they continue to be code creators and not just code validators.

Governing AI models

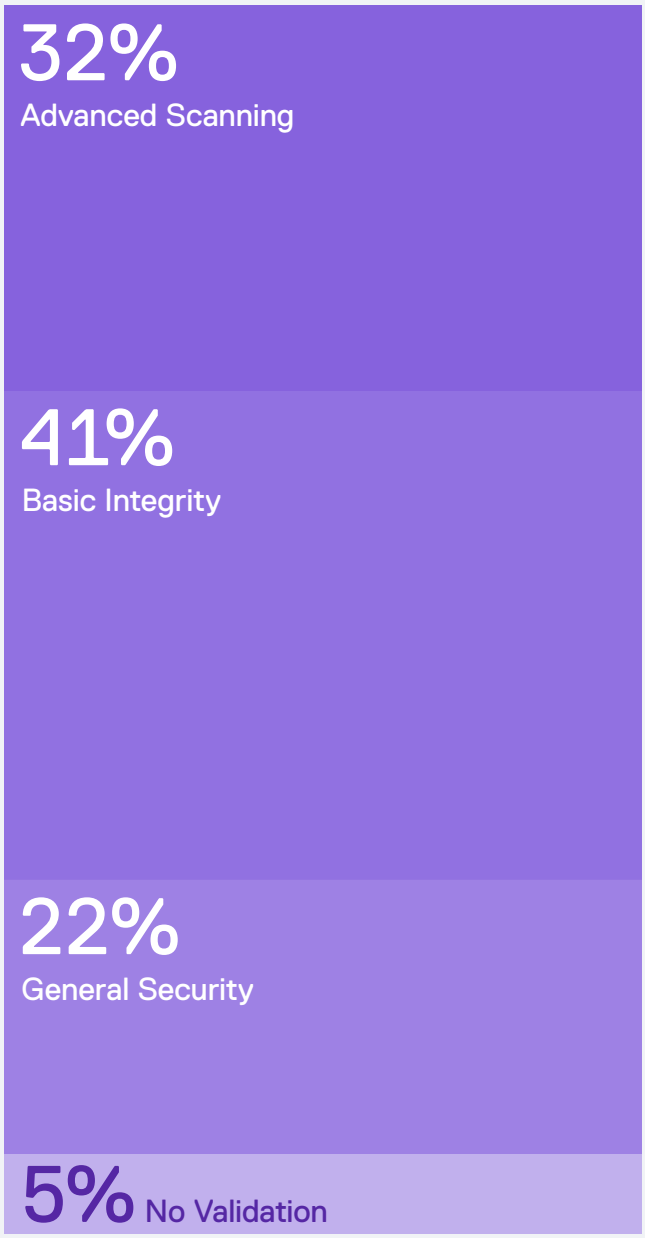
Transitioning to a ‘verify then trust’ posture requires a fundamental shift in how organizations treat machine learning assets in the software supply chain. Verification is a continuous process that begins at model selection and persists through storage and deployment. However, many organizations undermine this process by treating AI models as ‘outsiders’ to their standard security protocols, treating vendor reputation as a proxy for technical validation.

The illusion of ‘reputational security’

The move toward ‘verify then trust’ begins with the realization that artifacts from ‘trusted’ sources require internal vetting. Currently, organizations demonstrate a clear preference for established vendors when selecting AI assets; data shows that the most commonly adopted open source AI/ML models originate from Google (**69% for Gemma**) and Meta (**47% for LLaMA variants**).

While these models reduce exposure to unverified public registries, they can create a false sense of security that hinders deeper inspection. This ‘reputational security’ can lead to a fragmented and inconsistent approach to verification.

HOW ARE YOU CURRENTLY VALIDATING THE SECURITY OF AL MODELS BEFORE USING THEM IN PRODUCTION?



Advanced Scanning: We scan model files for specialized threats (e.g. hidden code execution, malicious weights, or PII leaks).

Basic Integrity: We verify checksums and provenance (signatures) but do not ‘inspect’ the model’s internal safety

General Security: We treat the model as a ‘black box’ and rely on runtime monitoring/WAFs to catch issues.

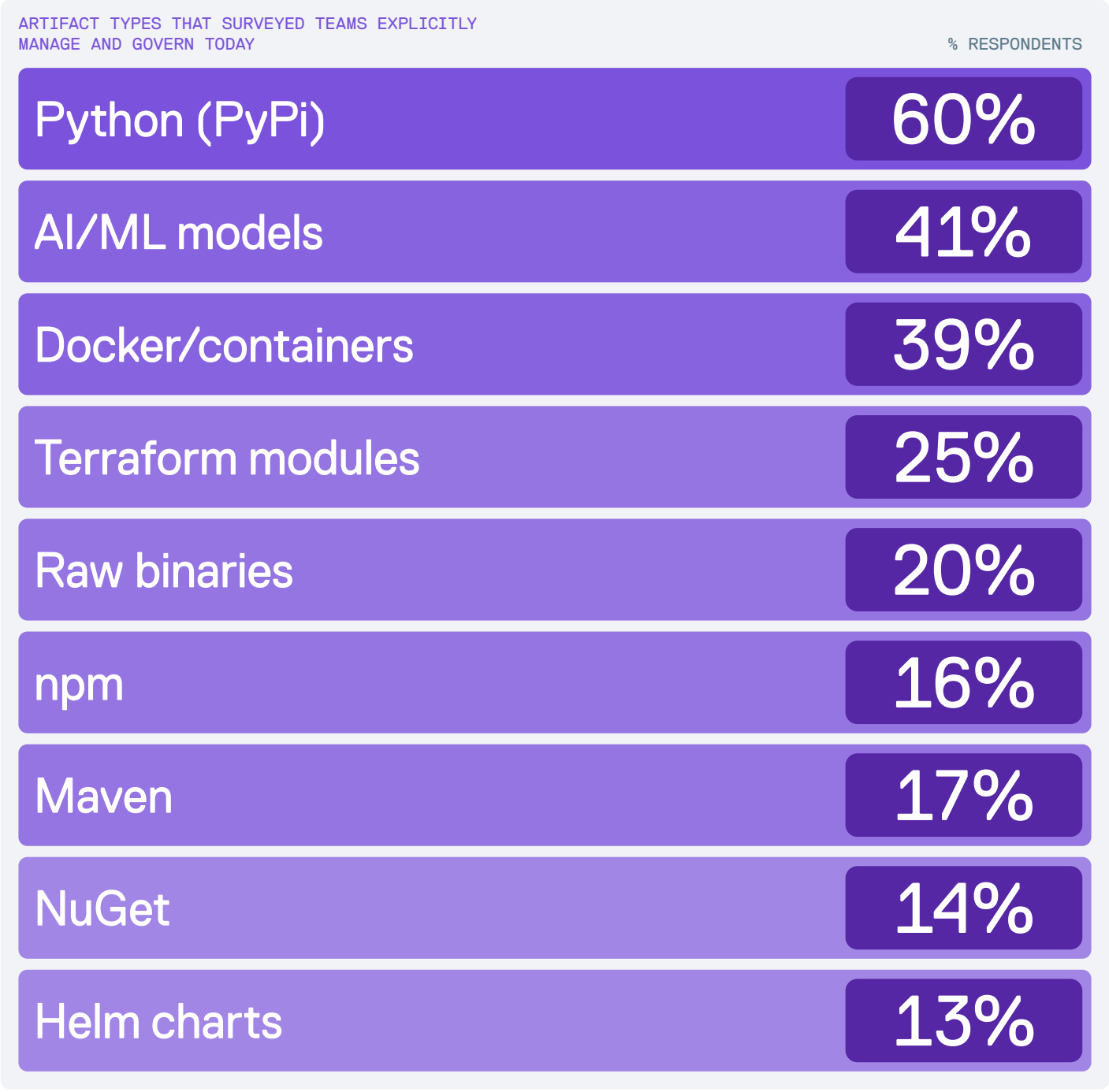
No Validation: We trust the source of the model (internal or reputable open source) and do not perform security checks.

Governance in a fragmented landscape

Visibility diverges between standard artifacts and AI/ML models. Security policies cannot govern what they cannot see and data source fragmentation across the industry for AI models makes visibility and policy enforcement challenging. AI/ML models constitute the second most managed artifact type at 41% (following Python at 60% and exceeding Docker/containers at 39%), but the rest of the software supply chain remains disconnected due to the infrastructure used to house these models.

The fragmented manner in which organizations store AI/ML models directly impacts their ability to govern them, creating risk. Isolating models in separate repositories hinders consistent governance across the software supply chain.

Organizations should adopt the same rigorous security practices for these models as they do for other binary artifacts. Therefore, developers require specialized tools to guarantee the security of AI model components.



A unified source of truth must include AI

Isolating AI models from standard development artifacts creates enforcement challenges and increases manual overhead. **Fragmented storage leads to inconsistent governance.** Ingesting models from unverified sources without oversight introduces security gaps. To mitigate these risks, organizations must implement a unified control plane. Consolidating AI models with traditional binaries extends security policies across the software supply chain. This ensures consistent governance and provenance tracking for all artifact types.

As supply chain attacks scale, enterprises require AI governance with built-in guardrails. The non-deterministic nature of AI models means prioritizing speed over centralized verification expands the attack surface. Integrating AI into a unified source of truth enables a proactive, secure-by-design development lifecycle.

How do you currently store and govern internal AI/ML models?

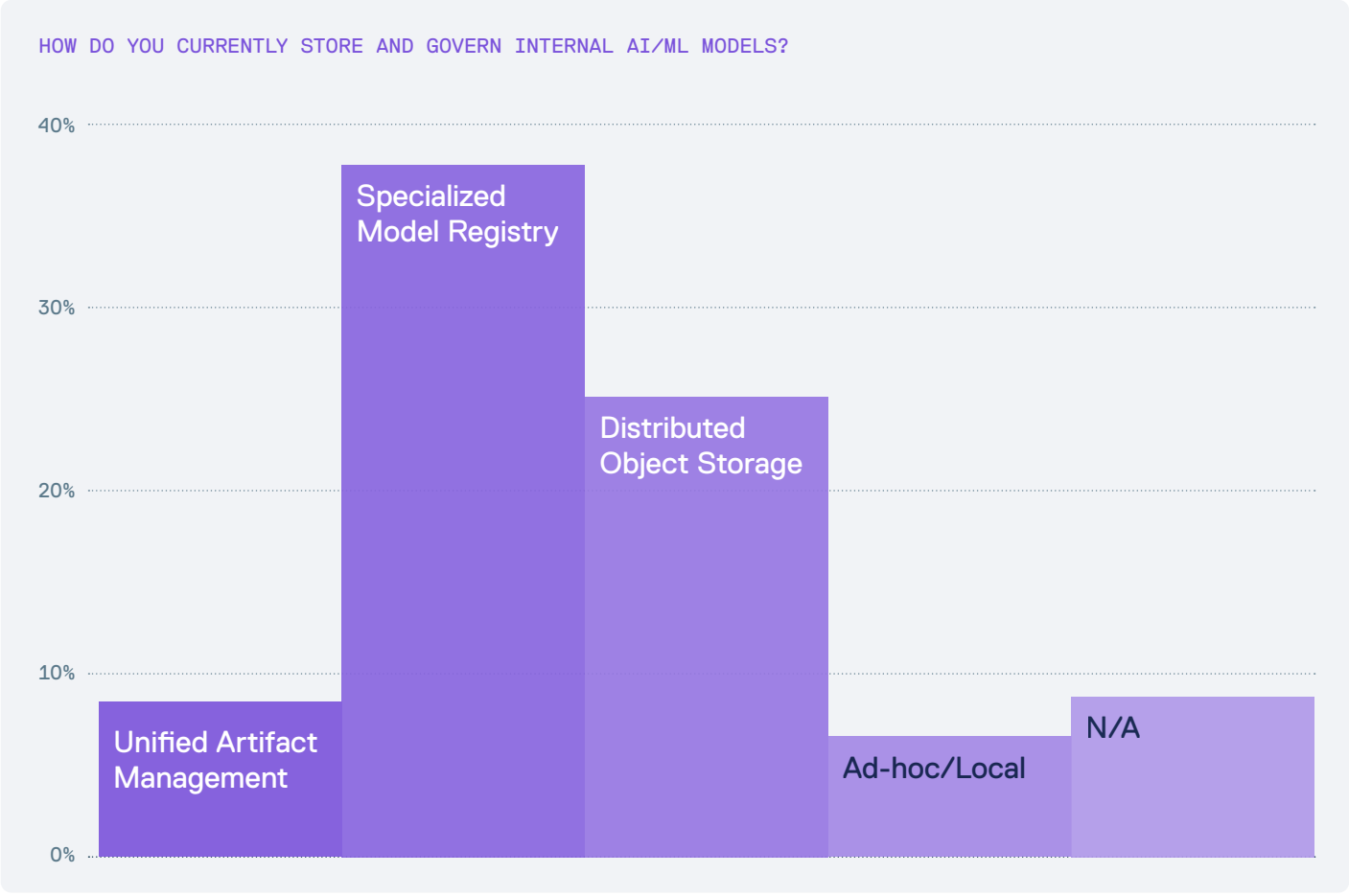
Unified Artifact Management: Models are stored in our standard artifact repository (e.g., as OCI artifacts) alongside our code and containers.

Specialized Model Registry: We use a dedicated AI tool (e.g. MLflow, Hugging Face Enterprise, Vertex AI Model Registry).

Distributed Object Storage: They are in raw cloud storage (S3, Azure Blob, GCS) with manual naming conventions.

Ad-hoc/Local: Stored on developer machines, shared network drives, or in Git LFS without formal governance.

N/A: We primarily consume AI via third-party APIs (OpenAI, Anthropic) and do not manage internal models.



Infrastructure limitations increase operational burden

Security, compliance, and AI initiatives fail if organizations cannot scale application delivery without interruption. Reliable package ingestion is critical for development. Performance bottlenecks during builds and releases delay high-level engineering initiatives.

The data for how current artifact management solutions perform vary according to several key variables tied to infrastructure and the design and implementation of those systems. **For 25% of respondents in platform engineering, infrastructure fragility is a primary failure point.** Data suggests that infrastructure has a measurable drag on developer experience and global operational reliability.

83%

of organizations use artifact management systems not built for AI-driven scale, distribution, or security.

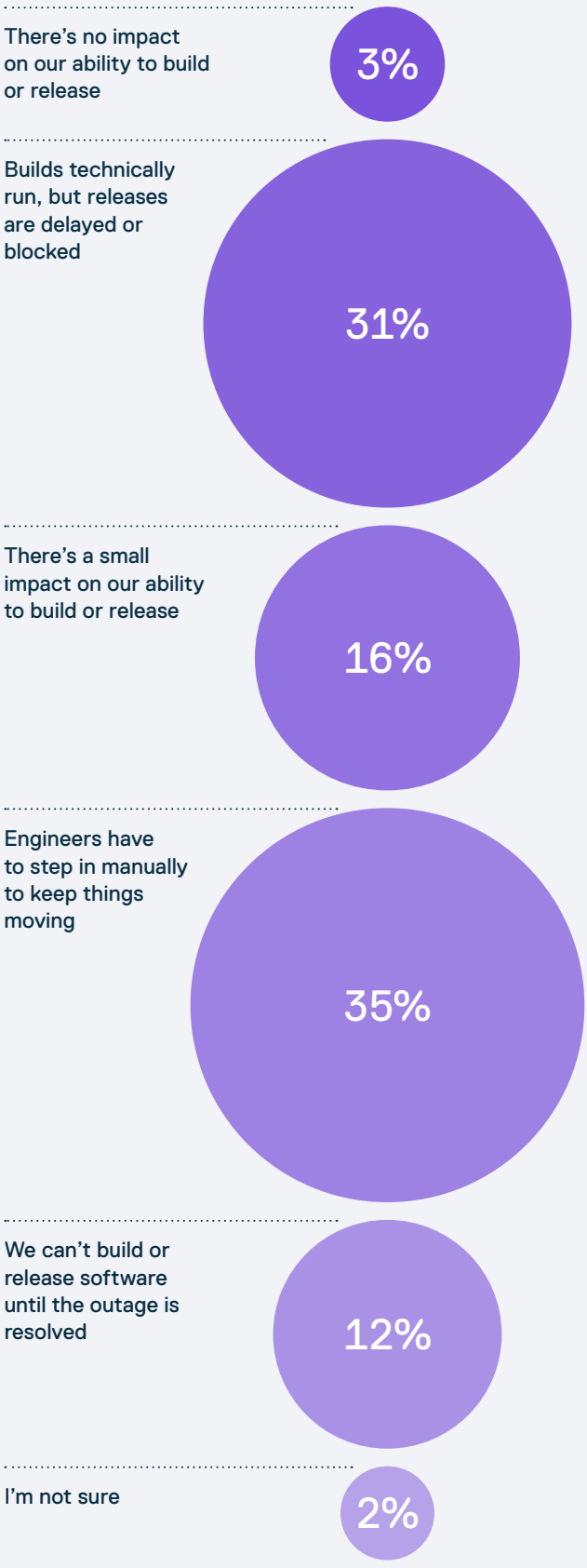
59%

of distributed team members experience frequent performance issues

51%

manually provision infrastructure in response to usage spikes

WHEN A MAJOR THIRD-PARTY INFRASTRUCTURE PROVIDER YOUR BUILDS DEPEND ON HAS AN OUTAGE, WHAT TYPICALLY HAPPENS TO YOUR SOFTWARE BUILDS AND ARTIFACT ACCESS?



The center-edge performance imbalance

Performance parity between the center and edge is rare. **59% of distributed team members experience frequent performance issues.** Within these teams, 10% report frequent timeouts and build failures. Compare this to 21% of teams that report frequent performance issues with artifact management infrastructure in the past 12 months that caused build delays or CI/CD pipeline errors during peak usage periods.

Unanticipated usage spikes

Usage spikes degrade performance for traditional artifact management systems. **51% of organizations must manually provision storage or compute, diverting engineering resources to infrastructure maintenance.** Another 16% experience immediate service degradations. These operational efforts are necessary to keep systems running, but they also prioritize low-value toil over strategic engineering, just one of the many hidden costs of having to manage infrastructure.

The fragility of the status quo

Outages are an inevitability that all vendors need to plan for. The design of an organization's artifact management tooling can temper the impact of an outage from a critical third-party infrastructure provider. System design can also expose its weaknesses, like a lack of redundancy or failover logic that is necessary for resilience.

‘Operational tax’, migration, and institutional inertia

Manual maintenance to keep artifact management solutions operational and performant imposes an operational tax of engineering hours and increased risk. 21% of respondents cite cost as a barrier to migration, reflecting a sunk-cost bias toward existing systems.

61% of respondents intend to migrate to managed cloud solutions. While 43% are moving to SaaS, 18% remain restricted by concerns over security, compliance, or technical debt.

Institutional inertia creates a security paradox as well. While 39% cite security and data sovereignty as reasons to avoid migrating to cloud, their current on-premise systems are arguably their greatest security risk.

How often do you delay or skip implementing new security/developer features (like OCI support for AI or SBOM generation)?

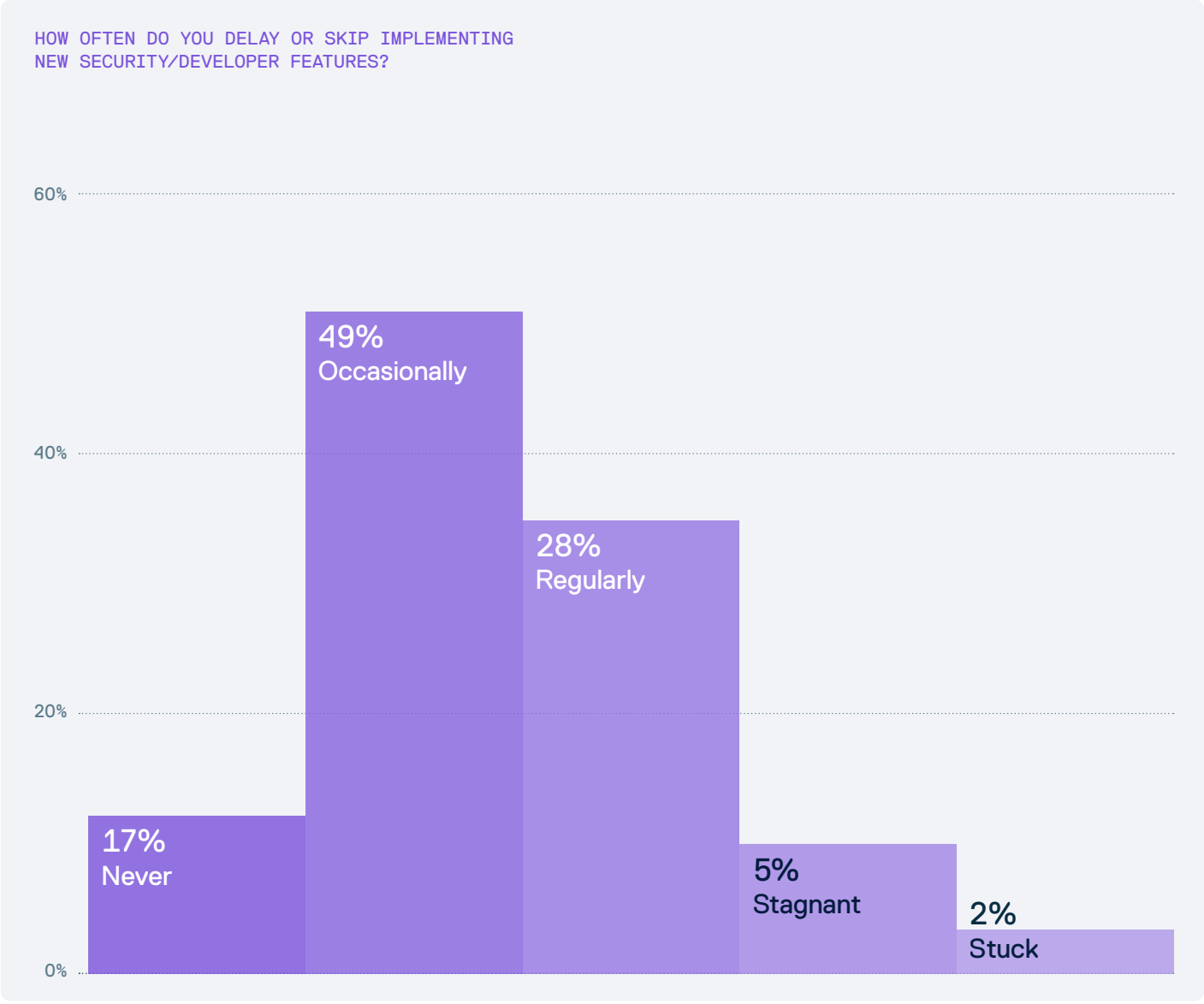
Never: We are always on the latest version and adopt new features instantly.

Occasionally: We are usually one or two versions behind due to the complexity of the upgrade path.

Regularly: Upgrades are a major ‘event’ that we only do once or twice a year, leaving us behind the feature curve.

Stagnant: We are on a legacy version and haven’t upgraded in over a year because of the risk of breaking things.

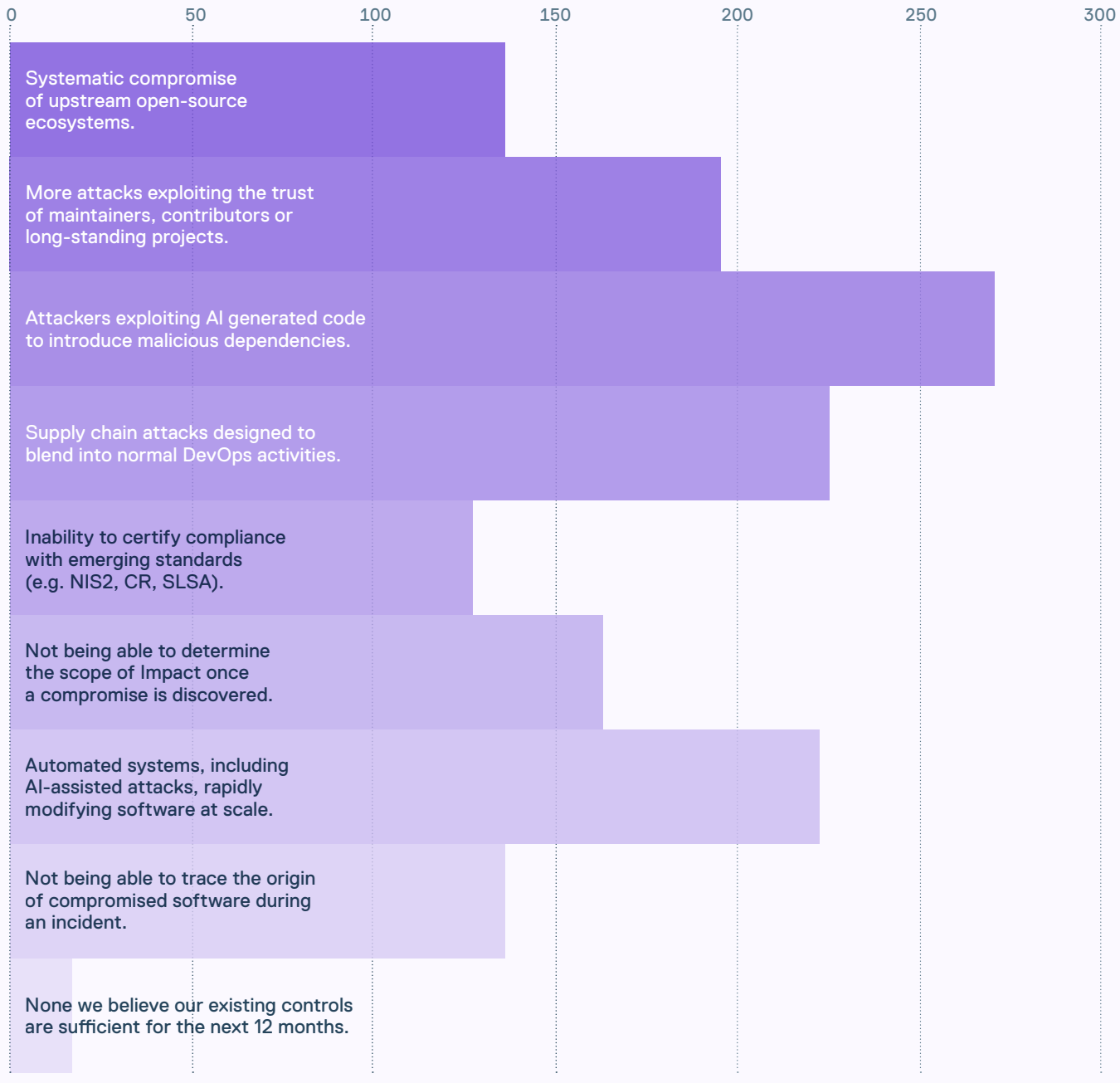
Stuck: Our current self-hosted setup is too difficult to upgrade.



2026 strategy: From complexity to consolidation

The strategic mandate for 2026 continues to shift as AI adoption scales. Fragmented toolchains create operational overhead. Instead, leaders are prioritizing toolchain consolidation to align security with AI operations between immediate security threats and the long-term operationalization of AI.

LOOKING AHEAD, WHICH SOFTWARE SUPPLY CHAIN RISK IS YOUR TEAM MOST CONCERNED ABOUT?



Fragmentation and complexity are key barriers to speed

The data shows correlations between the challenges that companies face. As organizations adopt more tools to increase speed, they create an operational tax that slows them down.

The primary challenges facing teams identified are not isolated issues:

operational resiliency (21%)

cost optimization (18%)

They are the direct result of a fragmented ecosystem where toolchain complexity and AI operationalization **(challenges tied at 14%)** impede developer productivity.

Organizations are finding that the more they layer AI on top of legacy artifact management infrastructure, the more they struggle with manual toil, technical debt, and institutional inertia.

Platform managers and executives alike are moving toward integrated systems that provide a single repository for all artifacts, ensuring that they can govern non-deterministic AI models with the same rigor as traditional code.

2026 investment priorities

Operational friction drives 2026 investment. Budgets are shifting from tool acquisition to infrastructure integrity. Allocations for 2026 reveal the highest concentration of investment in security and AI operationalization.

Security scanning remains the top investment priority for 29% of respondents. This isn't just 'business as usual'; it's a direct response to the surge in supply chain attacks, due in part to the proliferation of AI, and a requirement for audit readiness under the Cyber Resilience Act.

Closely following security is the need to operationalize AI/ML infrastructure, at 29%. This massive investment intent signals that organizations recognize the unsustainability of fragmented AI tools. Organizations are investing to resolve fragmentation, ensuring that AI-driven scaling doesn't come at the cost of safety.

Overcoming institutional inertia

Organizations must evaluate how institutional inertia affects their entire software supply chain in the face of security and AI-driven changes. The reality of modern software development, where most software is assembled, makes securing the software supply chain more important than ever. Performant access to those critical building blocks—at scale—is equally important to shipping code at speed.

Organizations must stop viewing security, AI, and infrastructure as separate budget items. The winners will be those who consolidate their toolchains into a single source of truth, managing AI models with the same policies as traditional code to reduce the cycle of technical debt and operational overhead.

Methodology

At Cloudsmith, we prioritize sharing accurate, data-driven insights. This report is based on real-world survey responses, reflecting genuine trends and challenges in software development and artifact management — rather than opinions or selectively filtered results. Our data spans a broad spectrum of industries and organization sizes. All findings are derived from anonymized responses collected through Cloudsmith-led research.

Participants

Total respondents: 505

Geographic distribution: Respondents are primarily based in the United Kingdom (50.3%) and the United States (49.7%).

Organization size: Over 64% of participants represent large-scale enterprises with more than 1,000 employees. Specifically, 35% work at organizations with over 5,000 employees (Fig. 01).

The participants in this report represent the cross-functional mix of roles responsible for modern software delivery. This variety provides a well-rounded perspective on the requirements and pain points associated with current artifact management solutions (Fig. 02).

Note: Percentages may not sum to exactly 100% due to rounding.

Fig. 01: What is the size of your organization?

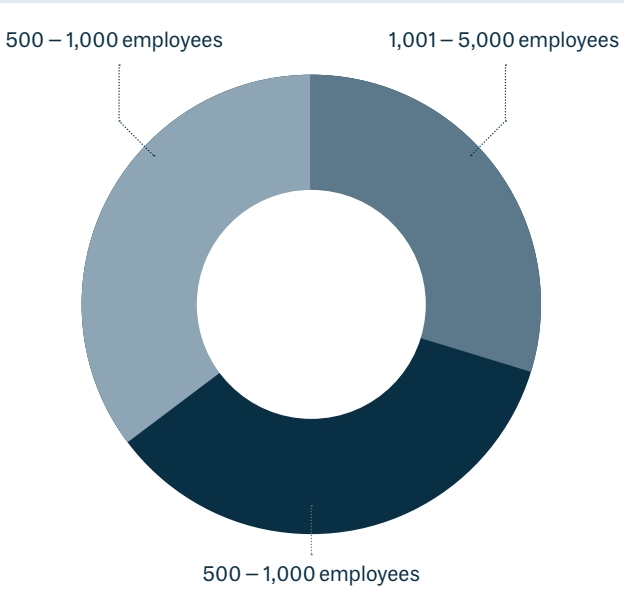


Fig 02: What is your job title or role?

JOB TITLES OR ROLE	PERCENTAGE
Software Architect / Lead Engineer	37%
Platform Engineering / DevOps	25%
Security Engineer / SecOps	21%
C-Level Exec (CTO, CIO, CISO)	17%