WHITEPAPER

Operationalizing Trusted Mission AI

A technical blueprint for risk-based software supply chain governance across Federal missions



© 2026 JFrog Ltd. All rights reserved.

CONTENTS

Operationalizing Trusted Mission AI

Executive Summary 03
The Technical Challenge 03
Paper Objective 03

01 Architecting the Binary as the System of Record 04
The Implementation Gap 04
Establishing the Single Source of Truth (SSOT) 05
Vaulting Assets 05
Capturing Provenance 05
Scanning Opaque AI and Legacy Systems 07

02 Implementing Policy-as-Code for Active Enforcement 08
Translating Risk Thresholds into Automated Gates 08
Executing Active Enforcement 09
Centralized Administration, Delegated Execution 09
Blocking at the Gateway 09
Executing Contextual Analysis 09

03 Automating Evidence Collection and SBOM Generation 11
The Compliance Requirement 11
Automating the Audit Trail (NIST SSDF Mapping) 12
CI/CD Integration 12
Dynamic Vulnerability Mapping 12
Cryptographic Evidence 12

04 Designing Defense-Grade Delivery for Cross-Domain Architectures 13
The Boundary Challenge 13
Executing Atomic Verification 14
Immutable Trusted Applications 14
Cryptographic Signing and Air-Gapped Promotion 14
Maintaining Continuous Authority to Operate (cATO) 14

Conclusion & Implementation Strategy 15
Moving from Observation to Operations 15
A 90-Day Roadmap for Agencies & FSIs 15

EXECUTIVE SUMMARY

The Federal Software Supply Chain Has Entered a Highly Contested Era

The Technical Challenge

Federal agencies and Federal Systems Integrators (FSIs) are transitioning to compliance with decentralized, risk-based governance mandates, such as the Office of Management and Budget (OMB) Memorandum M-26-05, NSPM-12, and the June 2026 AI Executive Order 14409.[1,2,3] Most notably, OMB M-26-05 has rescinded the "one-size-fits-all" self-attestations of the past (notably M-22-18 and M-23-16) and replaces them with an agency-defined, risk-based approach, directing agencies to develop localized software and hardware assurance policies tailored to unique mission threats.[3]

Concurrently, the Promoting Advanced Artificial Intelligence Innovation and Security Executive Order directs agencies to strengthen cybersecurity and governance to safely support advanced AI capabilities across Federal systems, including National Security Systems (NSS), with an emphasis on risk-based controls.[4]

These mandates arrive as the threat landscape shifts aggressively toward the software supply chain. The rapid adoption of opaque AI models and "Shadow AI" fundamentally breaks traditional, manual compliance workflows.[5] AI Steering Committees within FSIs—tasked with securing and fielding these innovations—face severe scalability limits. Enforcing strict governance via manual intake reviews against high-speed continuous integration/continuous deployment (CI/CD) pipelines creates massive bottlenecks, impeding the delivery of trusted mission AI.

Paper Objective

This whitepaper provides a technical blueprint for operationalizing AI governance and software supply chain security.[6] Moving beyond passive Application Security Posture Management (ASPM) tools, it details the implementation of automated Policy-as-Code and binary-centric security scanning to actively eliminate risk. By shifting the System of Record (SOR) to the compiled binary, organizations can automate the generation of binary-derived Software Bills of Materials (SBOMs) and mathematically guarantee artifact integrity across classified domains via Trusted Applications.

FRAMEWORKS REFERENCED

OMB M-26-05 | NSPM-12 | EO 14409 | NIST SP 800-218 (SSDF) | EO 14028 | NTIA SBOM Minimum Elements | CNSS / Cross-Domain Policy

1. Borgia and Lewis. "OMB Rescinds Biden-Era Software Security Requirements." *Davis Wright Tremaine*, 10 Feb. 2026.
2. Brown et al. "OMB Rescinds Secure Software Development Mandate." *Wiley*, 29 Jan. 2026.
3. OMB. *Adopting a Risk-Based Approach to Software and Hardware Security* (M-26-05). 23 Jan. 2026.
4. The White House. "Promoting Advanced AI Innovation and Security" Executive Order. 2 Jun. 2026.
5. JFrog LTD. *2026 Software Supply Chain Security State of the Union*. 2026.
6. CISA. *Securing the Software Supply Chain: Recommended Practices Guide for Developers*. Aug. 2022.

01

Architecting the Binary as the System of Record

The Implementation Gap

The industry relies heavily on source-code scanners and ASPM solutions that observe and score risk by aggregating alerts. However, these tools suffer from a critical implementation gap: they cannot physically prevent malicious code from executing, and they remain fundamentally blind to threats hidden inside compiled binaries or opaque AI model files.

State-sponsored actors and cybercriminals actively exploit this blind spot. According to JFrog's 2026 Software Supply Chain Security State of the Union Report, malicious npm packages skyrocketed by approximately 451% in 2025, reaching 171,592 unique instances, fueled by major hijack campaigns that resulted in over two million compromised downloads.[5]

In addition, threat actors are weaponizing AI and agentic ecosystems directly. JFrog researchers identified roughly 495 malicious models on Hugging Face carrying live payloads like reverse shells, as well as approximately 969 malicious AI agent skills in registries like ClawHub and Skills.sh. The same pattern extends across the wider agentic supply chain: agent skills, Model Context Protocol (MCP) servers, and IDE plugins are all installed from public registries with the same implicit trust once reserved for vetted dependencies, yet they execute with the developer's own credentials and network access.[5]

451%
Growth in malicious npm packages during 2025

171,592
Unique malicious package instances identified

495
Malicious models on Hugging Face with live payloads

969
Malicious AI agent skills across public registries

Establishing the Single Source of Truth (SSOT)

Vaulting Assets

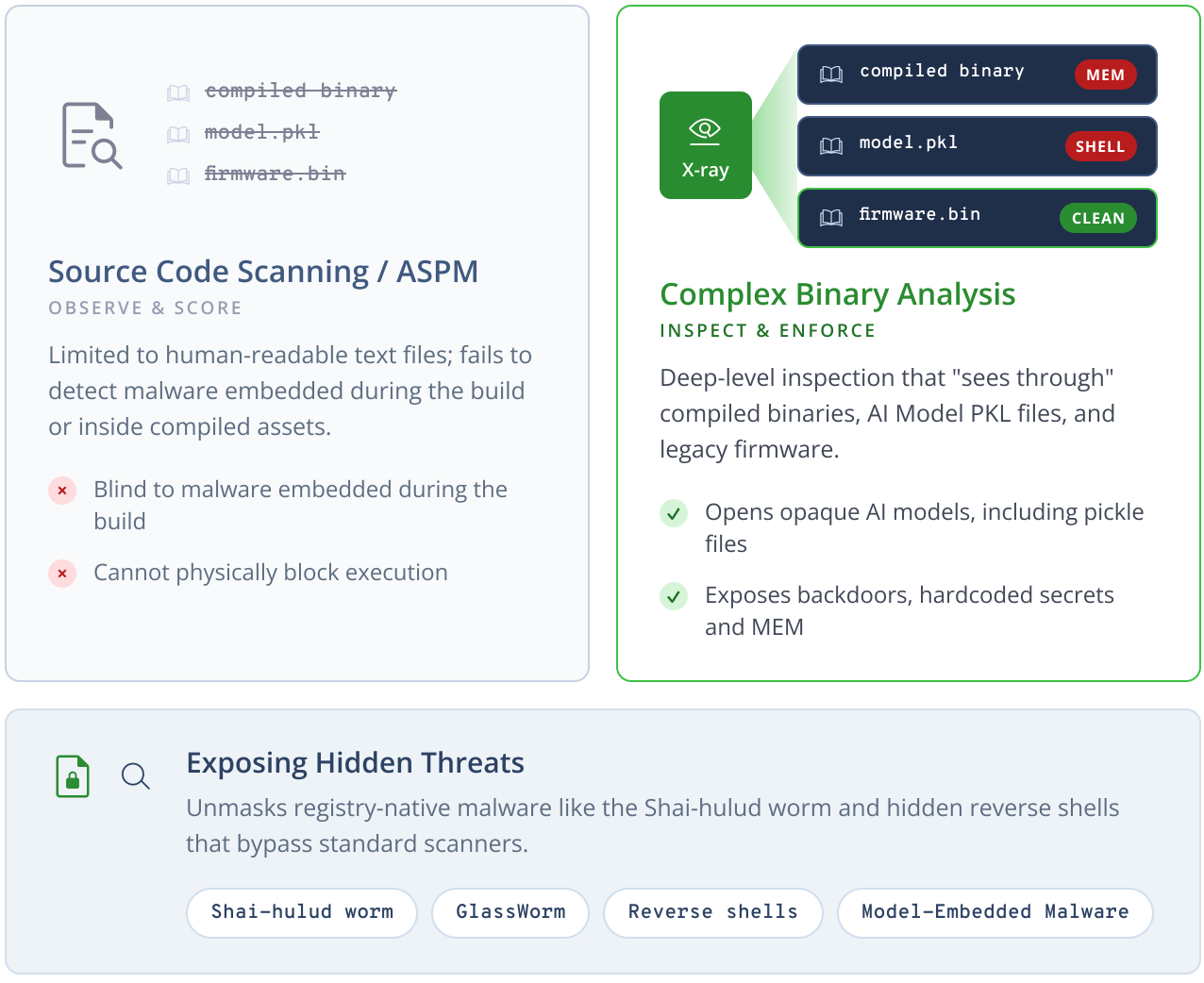
To combat these threats, organizations must shift the SOR away from the source code and onto the compiled artifact itself. Using a centralized binary vault, development teams should vault all traditional software, containers, and AI/ML models as standard binaries using checksum-based storage. The same discipline must cover the agentic supply chain: agent skills, MCP servers, and IDE plugins are executable dependencies and belong in the vault under identical controls, not in an unmanaged channel alongside it. Treating opaque AI models as standard binaries removes the blindfold, ensuring that every asset, whether a proprietary microservice or a public model from Hugging Face, is subjected to the exact same rigorous controls.

Capturing Provenance

Secure software development requires immutable provenance. To support implementation of NIST Secure Software Development Framework (SSDF) Protect Software (PS.1) controls, use a centralized binary vault that captures build metadata during compilation.[7] By capturing the exact metadata, environment variables, dependencies, and configurations required to recreate any build, you can provide structural proof of origin without relying on human data entry.[7]

5. JFrog LTD. *2026 Software Supply Chain Security State of the Union*. 2026.

FIGURE 1 See the Unseen: Source vs. Binary Scanning



While traditional source code scanners and ASPM tools suffer from critical blind spots, complex binary analysis provides complete X-ray visibility into opaque compiled binaries and AI models to expose hidden threats.

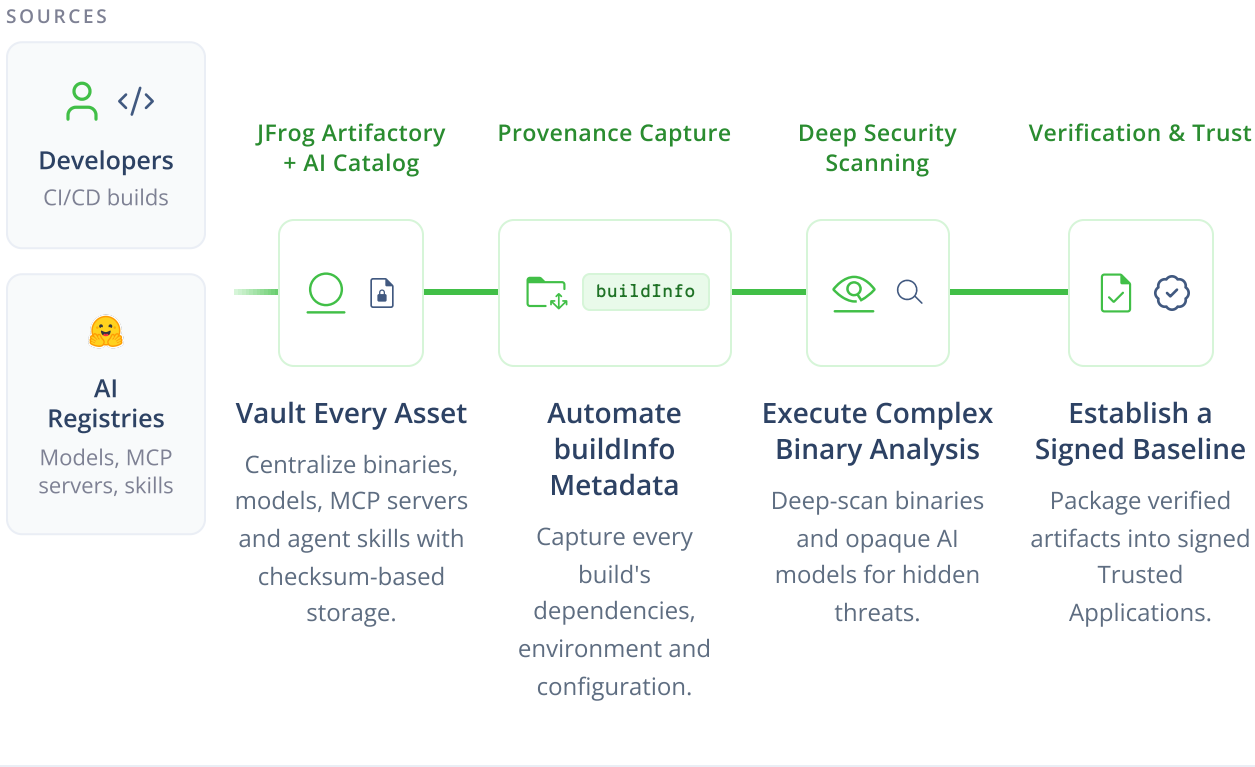
7. NIST. *Secure Software Development Framework (SSDF) Version 1.1*. SP 800-218.

Scanning Opaque AI and Legacy Systems

Once vaulted, these binaries are subjected to complex binary analysis. This capability is paramount for detecting registry-native threats like the Shai-hulud worm, which, according to public research estimates, compromised approximately 1,000 npm packages using stolen developer credentials.[5] It is also critical for uncovering threats like "GlassWorm", the first self-propagating worm targeting VS Code extensions, which reached approximately 35,800 installations by hiding malicious code inside invisible Unicode characters.[5]

By analyzing the binary that executes rather than just the code that produced it, architects can proactively expose hidden backdoors, hardcoded secrets, and Model-Embedded Malware (MEM) before they reach a runtime environment.

FIGURE 2 Establishing the Single Source of Truth



OUTPUT One System of Record — software and AI assets alike held to the same controls, with structural proof of origin.

By shifting the SOR to the compiled binary, you can securely vault traditional software and AI models alike, capturing immutable provenance (buildInfo) and subjecting opaque assets to rigorous complex binary analysis before deployment.

5. JFrog LTD. *2026 Software Supply Chain Security State of the Union*. 2026.

02

Implementing Policy-as-Code for Active Enforcement

Translating Risk Thresholds into Automated Gates

AI Steering Committees and AI Governance Boards bear the ultimate responsibility for setting risk thresholds and approving or halting high-impact AI deployments. However, relying on manual ticketing and human gatekeepers cannot survive the speed of agentic AI or operational mission velocity. Agencies and FSIs must translate these risk thresholds into automated structural enforcement.

An automated policy enforcement tool, working with an open-source dependency and vulnerability scanner, can operationalize this transition by converting theoretical frameworks into "Policy-as-Code". An active enforcement control plane allows security teams to deploy immutable policy gates that physically block binaries that fail governance standards from being promoted or downloaded.

Executing Active Enforcement

Centralized Administration, Delegated Execution

Utilizing Role-Based Access Control (RBAC), the platform supports the centralized administration of security, operational, and open-source licensing policies (mapping to NIST SSDF PO.1 and PO.2).[7,8] AI Governance Boards can define enterprise-wide baselines, while delegating the automated execution of those rules to specific CI/CD pipelines across the organization.

Blocking at the Gateway

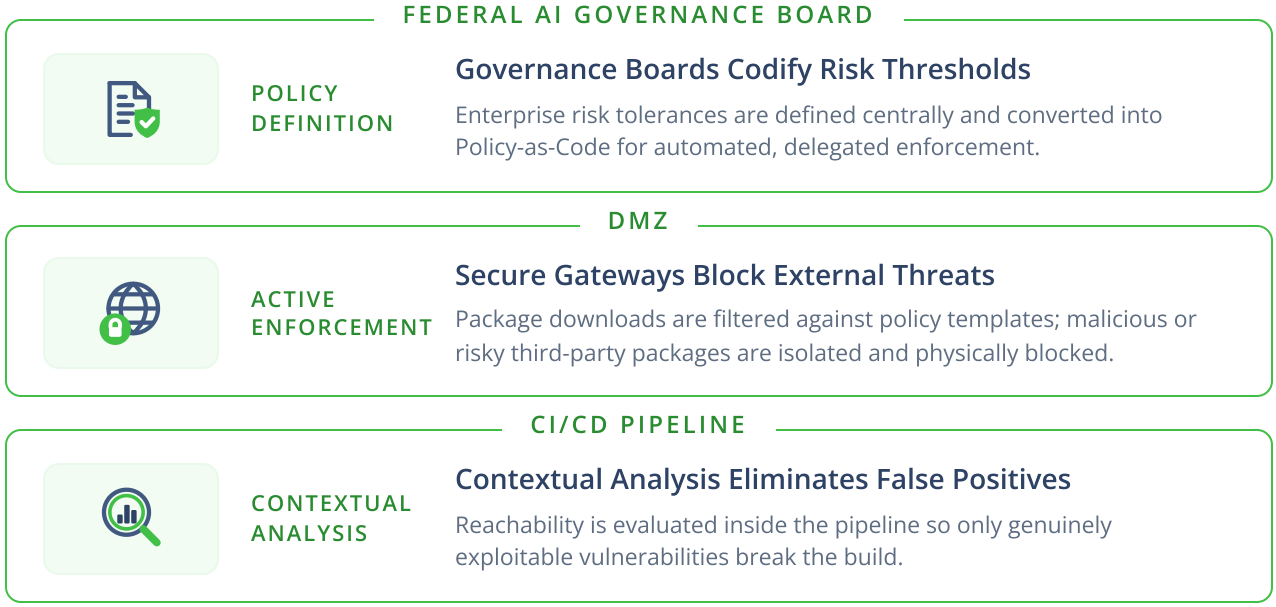
To radically reduce risk early in the SDLC process, use a secure gateway within the DMZ network segmentation.[9] Filter package downloads against defined policy templates, automatically isolate and physically block malicious or risky third-party packages before they can enter the developer environment.[9] The same gate must extend to what coding agents reach at runtime: JFrog Agent Guard restricts an agent to the MCP servers an administrator has approved, hooks every tool call inside the IDE to deny anything routed around that list, and locks the enforcement through standard device management so it cannot be switched off — producing the allow/deny audit record an Authorizing Official can inspect.

7. NIST. *Secure Software Development Framework (SSDF) Version 1.1*. SP 800-218.
8. JFrog LTD. *NIST SP 800-218 JFrog Platform Compliance Guide*. Ver. 1.3, Nov. 2024.
9. NTIA. *The Minimum Elements For a Software Bill of Materials (SBOM)*. 2021.

Executing Contextual Analysis

A primary reason developers bypass security gates is alert fatigue from false positives. An advanced binary security scanner solves this through Contextual Analysis (supporting NIST SSDF RV.2 and RV.3).[7,8] Rather than simply flagging a CVE, the engine analyzes code, configuration, and micro-services to determine actual applicability. It checks if the first-party code actually calls the vulnerable function and evaluates if the CVE is in a reachable path. If the vulnerability is not reachable, the system suppresses the alert, eliminating false positives and preventing unnecessary build breaks.[9]

FIGURE 3 Active AI Governance: From Policy to Enforcement



Federal AI Governance Boards establish enterprise risk thresholds, which are translated into automated Policy-as-Code. A secure gateway actively blocks malicious open-source packages at the DMZ, while an advanced binary security scanner executes Contextual Analysis within the CI/CD pipeline to eliminate false positives and approve clean builds.

7. NIST. *Secure Software Development Framework (SSDF) Version 1.1*. SP 800-218.
8. JFrog LTD. *NIST SP 800-218 JFrog Platform Compliance Guide*. Ver. 1.3, Nov. 2024.
9. NTIA. *The Minimum Elements For a Software Bill of Materials (SBOM)*. 2021.

03

Automating Evidence Collection and SBOM Generation

The Compliance Requirement

Federal guidance, including Executive Order 14028 and NTIA's The Minimum Elements For a Software Bill of Materials (SBOM), established expectations for machine-readable, automatically generated SBOMs in Federal software procurement. Although OMB M-26-05 has rescinded earlier blanket SBOM mandates, agencies can still require SBOMs as part of their own risk-based assurance policies.[9,10,11] Additionally, OMB M-26-05 emphasizes the need for dynamic assurance tailored to mission needs.[1] To meet these expectations, developers must be freed from the implementation friction of manual attestation paperwork, requiring an automated system that collects evidence silently in the background.

1. Borgia and Lewis. "OMB Rescinds Biden-Era Software Security Requirements." *Davis Wright Tremaine*, 10 Feb. 2026.
9. NTIA. *The Minimum Elements For a Software Bill of Materials (SBOM)*. 2021.
10. NIST. *Software Security in Supply Chains: Software Bill of Materials (SBOM)*. Updated 1 Nov. 2024.
11. The White House. "Executive Order 14028: Improving the Nation's Cybersecurity." 12 May 2021.

Automating the Audit Trail (NIST SSDF Mapping)[8]

CI/CD Integration

Employing a unified Software Composition Analysis (SCA) scanner that includes an automated SBOM generator to seamlessly integrate into the CI/CD pipeline will programmatically produce binary-derived SBOMs directly from the executing builds. It also enables the SBOMs to be exported in Federally accepted, machine-readable standard schemas, such as SPDX and CycloneDX, aligning with NTIA's minimum SBOM data fields and automation guidance.[9,10]

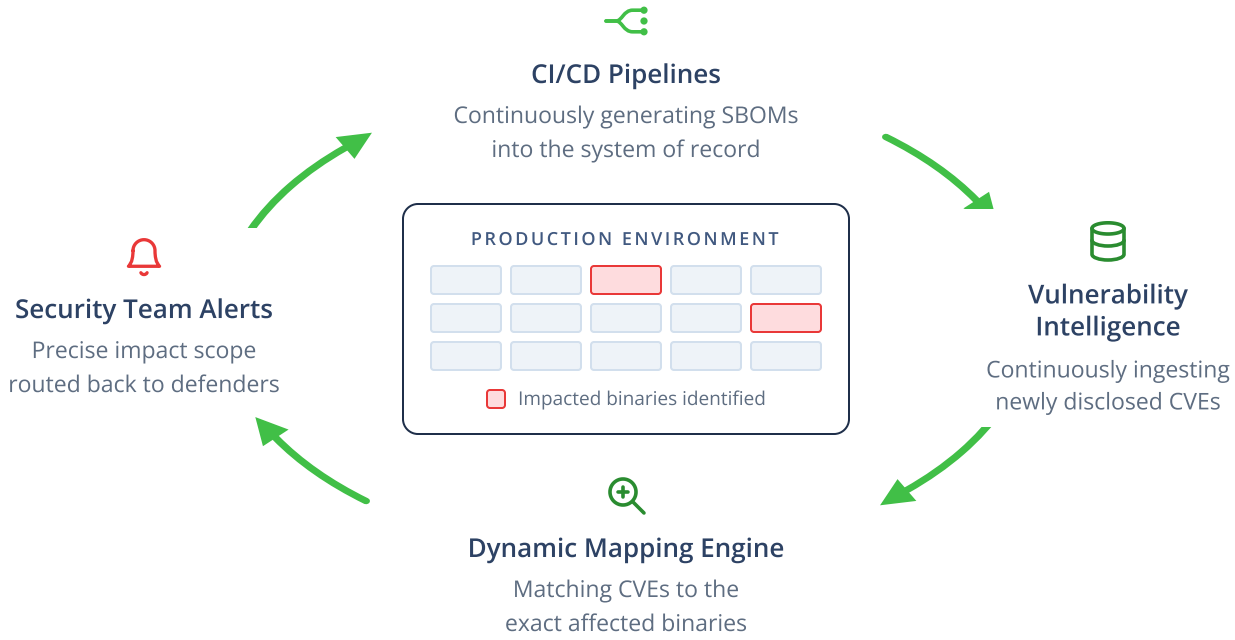
Dynamic Vulnerability Mapping

While an SBOM is a static document, the threat landscape is highly dynamic. It is essential to dynamically map generated SBOMs against continuous, real-time vulnerability intelligence. If a new zero-day exploit emerges in a third-party open-source component, your platform should be able to instantly cross-reference the component against the vaulted SBOMs to identify the exact impacted descendant production binaries across your fleet.

Cryptographic Evidence

For strongly proactive and dynamic SBOM defense, as code moves through the SDLC, your platform must automatically log events, user queries, and permissions, and generate the unalterable audit trails necessary to demonstrate alignment with NIST SP 800-218 Produce Well-Secured Software and Respond to Vulnerabilities practices.[7]

FIGURE 4 Dynamic SBOM Defense: Securing the Software Flow



The CI/CD pipeline automatically outputs a machine-readable Software Bill of Materials (SBOM) in standard formats like SPDX or CycloneDX. This static inventory is continuously mapped against a real-time vulnerability intelligence database, dynamically matching newly discovered zero-day threats to specific production binaries.

7. NIST. *Secure Software Development Framework (SSDF) Version 1.1*. SP 800-218.
8. JFrog LTD. *NIST SP 800-218 JFrog Platform Compliance Guide*. Ver. 1.3, Nov. 2024.

04

Designing Defense-Grade Delivery for Cross-Domain Architectures

The Boundary Challenge

For NSS, the Committee on National Security Systems (CNSS) as well as a series of National Security Presidential Memorandums (NSPM) establish strict technical mandates for cross-domain solutions and the separation of security domains.[12] Moving software from an unclassified development network to a Secret or TS/SCI air-gapped network presents a large-scale vulnerability. Trust cannot be assumed across these boundaries; it must be cryptographically proven.

12. The White House. "National Policy for the Cybersecurity of National Security Systems (NSPM-12)." June 2026.

Executing Atomic Verification

Immutable Trusted Applications

Instead of using orchestration tools that rely on fragmented pipeline completion checks, agencies are best served by employing a dedicated "Trusted Release" capability within your platform. A secure distribution capability packages the verified artifacts, dependencies, build metadata, and SBOMs into a single, immutable package—a "Trusted Application"—directly from the CI/CD output.

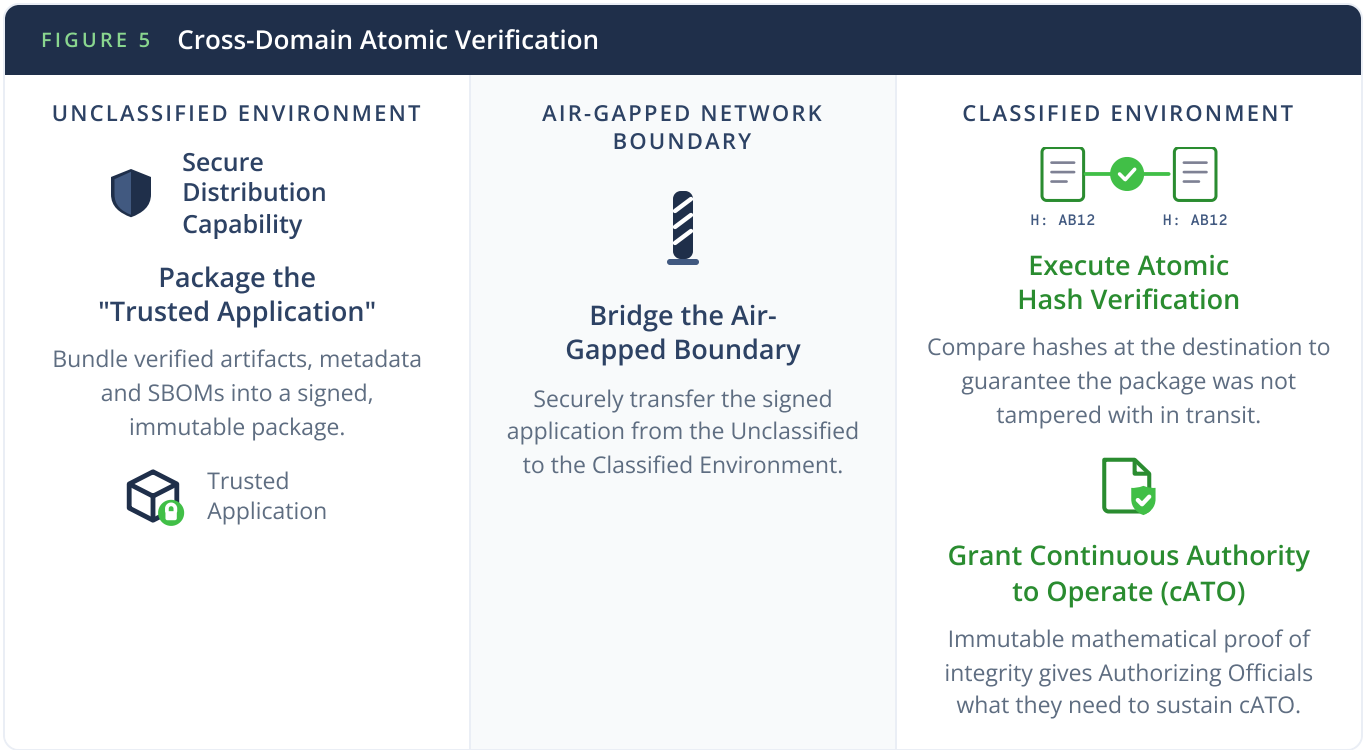
Cryptographic Signing and Air-Gapped Promotion

By utilizing a Trusted Release capability within your platform, you enable the Trusted Application to be cryptographically signed and encrypted using GPG keys.[9] Then, when this immutable application is promoted across the boundary to a target platform instance residing on the classified or air-gapped network, the receiving instance automatically initiates verification.[9]

Maintaining Continuous Authority to Operate (cATO)

Upon receipt, the target platform verifies the atomicity of the bundle by computing and comparing the hashes.[9] If the hashes match, the software is proven to be unaltered from the exact moment it left the secure development environment, and is permitted to be consumed by the classified runtime. If the hashes fail, the artifact is physically halted. This atomic verification provides Authorizing Officials (AOs) with strong, cryptographic evidence of software integrity that can support sustaining a cATO.

FIGURE 5 Cross-Domain Atomic Verification



Verified artifacts, metadata and SBOMs are packaged into an immutable, signed Trusted Application. The receiving system recomputes and compares hashes at the boundary — a match mathematically proves integrity, sustaining a cATO.

8. JFrog LTD. *NIST SP 800-218 JFrog Platform Compliance Guide*. Ver. 1.3, Nov. 2024.

9. NTIA. *The Minimum Elements For a Software Bill of Materials (SBOM)*. 2021.

Conclusion & Implementation Strategy

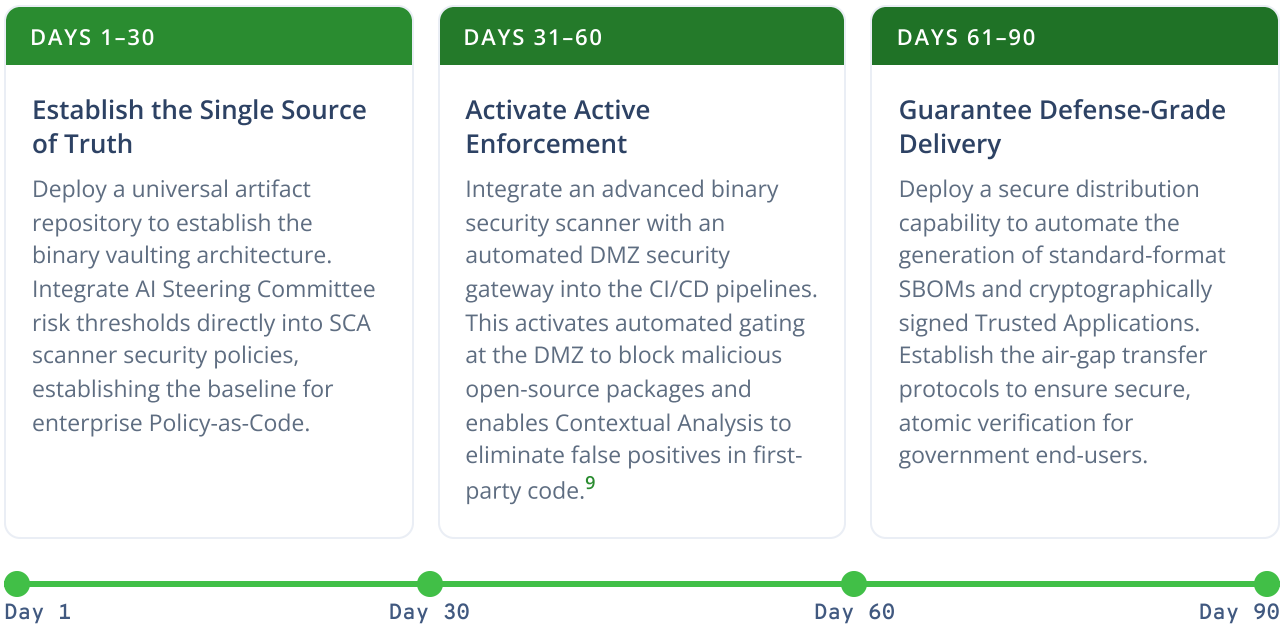
Moving from Observation to Operations

To rapidly and safely adopt agentic AI and comply with decentralized Federal risk mandates, mission-driven organizations must abandon manual frameworks. Aggregating risk scores through traditional ASPM tools is insufficient. Agencies and FSIs must implement a structural DevSecOps control plane that governs the supply chain at the binary level—actively blocking threats and automating compliance.

A 90-Day Roadmap for Agencies & FSIs

To operationalize this architecture, agencies and FSIs should follow a structured 90-day implementation roadmap:

FIGURE 6 The 90-Day Federal Implementation Roadmap



The 90-day Federal implementation roadmap — operationalize NIST-aligned security through automated binary-centric gating and distribution.

By following this blueprint, organizations can eliminate technical debt, block embedded AI malware, automate complex compliance proofs, and successfully field trusted mission AI to the warfighter at the speed of modern innovation.

9. NTIA. *The Minimum Elements For a Software Bill of Materials (SBOM)*. 2021.

Operationalize Trust in Your AI Supply Chain

JFrog works with Federal agencies and Federal Systems Integrators to operationalize binary-level supply chain governance — from vaulting and complex binary analysis through cryptographically verified, air-gapped delivery. JFrog AI Catalog extends that governance to the AI assets themselves — models, MCP servers, and agent skills — giving agencies a curated entry point where approved assets arrive with their provenance, licenses, and scan results attached.

Learn more about AI Catalog →

Contact a JFrog Fed Expert →



jfrog.com

© 2026 JFrog Ltd. All rights reserved.