DevSecOps Aware in Healthcare: SBOM-Driven Supply-Chain Assurance (SLSA) with Policy-Based Cost Guardrails and Continuous Security Validation

Nagarjuna Nellutla
Independent Researcher
Eagan, MN, USA 55123
nagarjunanellutla9@gmail.com

Abstract—**Healthcare cloud systems must satisfy strict security and compliance controls while operating under constrained budgets. Traditional DevSecOps pipelines improve delivery velocity but often treat cost governance and supply-chain assurance as separate concerns, leaving gaps in artifact traceability, dependency risk visibility, and budget enforcement. This paper proposes a FinOps-aware DevSecOps pipeline for healthcare workloads that integrates software bill of materials (SBOM) generation, SLSA-aligned supply-chain assurance checkpoints, and policy-as-code gates that jointly enforce security, compliance, and cost guardrails from build to deployment. The approach emphasizes auditable evidence, artifact integrity, and continuous validation to reduce release risk and cost drift without undermining delivery performance.**

Index Terms—FinOps, DevSecOps, Healthcare Cloud, SBOM, Supply Chain Security, SLSA, Policy-as-Code, Compliance Automation, Cloud Cost Governance

I. INTRODUCTION

Healthcare organizations increasingly rely on cloud platforms to scale clinical and operational services, improve availability, and accelerate delivery of digital capabilities. For workloads that handle electronic protected health information (ePHI), faster delivery must be balanced with HIPAA-aligned safeguards, including strong access control, reliable auditability, integrity protections, and secure transmission and storage practices. At the same time, cloud consumption introduces persistent financial pressure. CI workloads, multi-environment sprawl, oversized resources, and ungoverned data transfer can amplify spend quickly, often faster than teams can detect or correct it. In many organizations, security and compliance controls have matured within CI/CD through scanning and automated checks, while cost governance remains external to the delivery process and is addressed through reporting, periodic optimization, or finance-led review cycles. This separation creates a structural weakness: a release can be "secure enough" to pass technical gates, yet still introduce avoidable cost drift and produce incomplete, hard-to-audit evidence chains.

Software supply-chain risk compounds this challenge. Modern healthcare applications are assembled from extensive third party dependencies, container images, and managed build services, and the delivery pipeline itself becomes part of the attack surface. When dependency inventories are incomplete and build provenance is not verifiable, it becomes difficult to answer audit-critical questions with confidence: precisely what was deployed, which components were included, and whether the artifact originated from an approved, tamper-resistant build process. These gaps are not only operationally costly during incident response and audits; they also undermine governance because controls are evaluated without a deterministic link to the artifacts that run in production.

This paper presents a FinOps-aware DevSecOps pipeline for HIPAA-relevant healthcare workloads on AWS that unifies supply-chain assurance and cost governance with continuous security validation. The pipeline binds every deployable artifact to an immutable identifier and attaches two evidence primitives: a software bill of materials (SBOM) that provides component-level transparency and SLSA-aligned provenance that links the artifact to its source and build context. A single policy-as-code layer then evaluates security, compliance relevant checks, and FinOps guardrails as promotion criteria, producing audit-ready decision records that are stored as release evidence. The result is a delivery model in which cost accountability and supply-chain integrity are enforced at the same point where organizations already enforce security gates, improving traceability and reducing both release risk and cost drift while preserving a practical workflow suitable for healthcare delivery teams.

II. II. BACKGROUND AND RELATED WORK

HIPAA's Security Rule establishes safeguards for ePHI [1], motivating technical controls around access, audit, integrity, and transmission security. In cloud-native environments, guidance on container and micro service security emphasizes defense-in-depth and continuous validation to reduce misconfiguration and runtime risk [2]. FinOps formalizes cross-functional accountability [3] for cloud cost management and promotes practices such as allocation, budgeting, and continuous optimization. SBOM standards such as SPDX [4] and Cyclone [5] enable structured dependency inventories that support vulnerability response and auditability. SLSA provides a framework for improving build integrity and provenance to reduce tampering and increase traceability from source to artifact [6]. Policy-as-code approaches enable machine evaluable governance checks to be enforced consistently across delivery workflows while producing decision evidence.

III. PROBLEM STATEMENT AND THREAT MODEL

A. Problem Statement

Healthcare delivery teams need an end-to-end pipeline that provides auditable assurance for workloads handling ePHI while preventing cost drift. The pipeline must ensure that every deployed artifact is traceable to its source and build context, that an SBOM and verifiable provenance exist for what is promoted, that security and compliance controls are evaluated continuously through automated gates, and that cost governance is enforced as a first-class release criterion rather than a post-deployment reporting activity. Current implementations frequently meet these needs partially and in disconnected tooling, which leads to inconsistent enforcement and incomplete evidence during audits.

B. Threat Model

This work considers adversarial and operational risks across the software supply chain, cloud configuration, and financial governance. Supply-chain risks include dependency compromise through malicious packages or compromised upstream sources, as well as build tampering via compromised CI runners or injected build steps. Deployment risks include image drift and artifact substitution, where unverified images or images lacking required SBOM/provenance are promoted. Additional risks include secret leakage through logs or artifacts, and compliance drift caused by insecure IAM policies, missing encryption or logging, or unintended network exposure of systems that process ePHI. Finally, cost drift risks include untagged spend, oversized compute, runaway CI workloads, uncontrolled data egress, and environment sprawl. The proposed architecture reduces likelihood and blast radius by combining SBOM transparency, SLSA-aligned provenance verification, and unified policy gates for security, compliance, and cost.

IV. PROPOSED ARCHITECTURE AND SYSTEM DESIGN

A. Pipeline Overview (AWS Reference Design)

The proposed FinOps-aware DevSecOps pipeline binds each release to an immutable artifact, an SBOM, and verifiable provenance metadata, and then enforces unified policy gates for security, HIPAA-relevant compliance controls, and cost guardrails before promotion across environments [7]. Artifacts are promoted by digest to prevent tag-based ambiguity, and each promotion produces evidence records that are stored for audit retrieval.

Figure 1 summarizes the end-to-end release lifecycle and the evidence that is generated and persisted at each stage. The design highlights how SBOMs, provenance, scans, and policy decisions are bound to digest-pinned artifacts to support auditability and controlled promotion.

B. Account and Environment Segmentation

The reference deployment uses AWS Organizations with separate accounts for development, testing, and production to reduce blast radius and support segregation of duties. Centralized logging and security monitoring operate from a dedicated

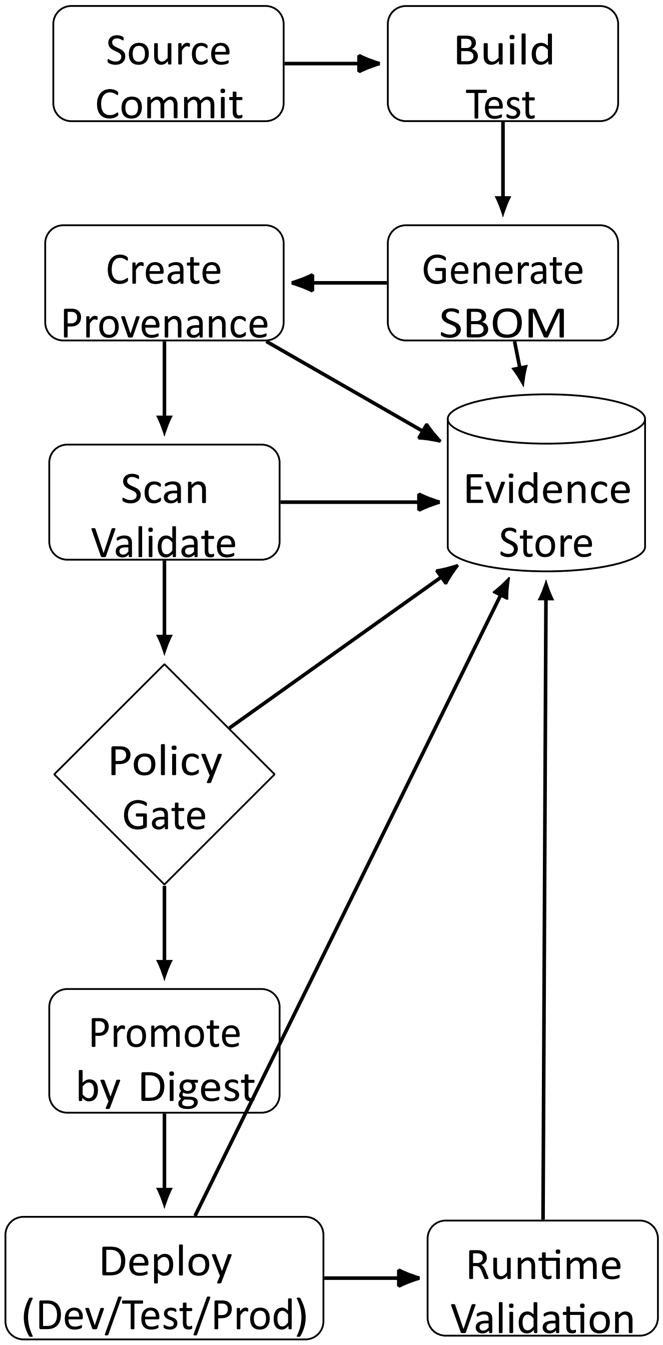

Fig. 1. Evidence-bound FinOps-aware DevSecOps pipeline on AWS. SBOMs, provenance records, scan outputs, and policy decisions are stored as release evidence and linked to digest pinned promotions.

Security/logging account [8]. Network boundaries rely on VPC segmentation and private subnets for systems handling ePHI, with controlled service access patterns that reduce unnecessary exposure while preserving operational observability. Figure 2 depicts the AWS multi-account segmentation used to reduce blast radius and centralize security logging and evidence retention. This structure supports segregation of duties and simplifies audit reconstruction across development, test, and production environments.

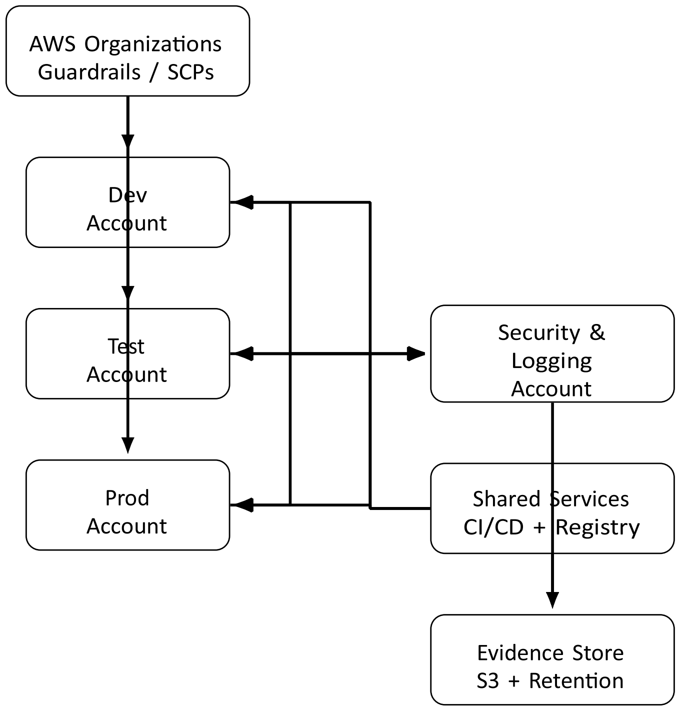
AWS Organizations Guardrails / SCPs
Dev Account
Test Account
Prod Account
Security & Logging Account
Shared Services CI/CD + Registry
Evidence Store S3 + Retention

Fig. 2. AWS multi-account segmentation aligned with HIPAA relevant governance. Centralized security/logging and evidence retention support auditability and blast-radius reduction.

C. *Build, Artifact, and Evidence Stores*

Source changes trigger a CI workflow that performs controlled builds and produces deployable artifacts. Container images are stored in Amazon ECR and referenced by immutable digests rather than mutable tags. Audit evidence—SBOM files, provenance attestations, scan summaries, and policy decisions—is stored in an append-only manner in an evidence store (e.g., Amazon S3 with retention controls [9]), enabling later reconstruction of what was deployed and which checks were satisfied.

D. *SBOM Generation and Binding to Artifacts*

During build, an SBOM is generated for application dependencies and container images, then stored alongside the corresponding image digest [10]. The pipeline requires that the SBOM record matches the immutable artifact identifier prior to promotion, ensuring that dependency inventory is directly tied to what runs in each environment. This linkage supports rapid exposure analysis during vulnerability response and improves auditability by making deployed composition explicit.

E. *SLSA-Aligned Provenance Checkpoints*

The pipeline produces provenance metadata describing the source reference, build definition, build environment identity, and the resulting artifact digest [11]. Provenance records are cryptographically protected using managed key controls and stored as evidence. Promotion stages verify the presence and integrity of provenance before allowing deployment, reducing the risk of artifact substitution and strengthening traceability from code to runtime.

F. *Continuous Security, Compliance, and Cost Validation*

Security and compliance validation occurs both before deployment and after deployment. Pre-deploy gates evaluate infrastructure and application changes, enforce vulnerability and secret-handling thresholds, and require policy conformance before promotion. Post-deploy validation detects configuration drift, aggregates security findings, and centralizes audit trails for identity and workload activity. In parallel, FinOps guardrails are enforced as release criteria through policy evaluation, preventing promotion when allocation tagging is missing, when projected spend exceeds predefined envelopes, or when resource and egress configurations violate cost governance constraints. Policy decisions and their rationales are recorded as evidence artifacts to support repeatable governance and audit readiness.

G. *Policy-as-Code Enforcement and Audit Evidence Packaging*

A unified policy-as-code layer evaluates security, compliance, and cost rules using SBOMs, provenance, scan outputs, infrastructure plans, and deployment manifests as inputs, and produces allow/deny decisions with structured reasons [12]. These policy decisions are stored and linked to release identifiers and artifact digests. For HIPAA-oriented audit workflows, each release yields an evidence bundle that includes SBOM, provenance, validation outputs, and deployment metadata, enabling rapid reconstruction of what ran in production and under which controls.

V. V. IMPLEMENTATION APPROACH

This section describes a practical implementation on AWS that remains consistent with HIPAA-oriented controls and the supply-chain and FinOps objectives defined earlier. The implementation is intentionally modular so that organizations can adopt the evidence model (SBOM, provenance, policy decisions) without requiring a single CI/CD product or a single deployment platform. Table I enumerates the evidence artifacts captured across the lifecycle and stored as a release bundle. This evidence model enables deterministic reconstruction of deployments from immutable artifact digests.

Source changes trigger a controlled build stage that produces a container image and attaches two evidence artifacts to the release: an SBOM and a provenance record.

TABLE I. EVIDENCE ARTIFACTS CAPTURED ACROSS THE RELEASE LIFECYCLE

Stage	Evidence captured (stored as release bundle)
Build/Test	Build logs, test results, artifact digest, CI identity metadata
SBOM	SBOM document bound to immutable artifact digest
Provenance	Provenance record linking source revision, build context, and digest
Scan/Validate	Vulnerability summaries, IaC findings, secrets checks, config checks
Policy Gate	Signed allow/deny decision with reasons and evaluated inputs
Deploy/Runtime	Deployment metadata, drift findings, audit trail references

TABLE II. UNIFIED POLICY-AS-CODE INPUTS ACROSS SECURITY, COMPLIANCE, AND FINOPS

Policy domain	Primary evaluated inputs
Security	SBOM, scan outputs, container metadata, secrets checks, IAM deltas
Compliance	Encryption/logging configurations, network exposure, audit trail presence
FinOps	Mandatory tags, planned resource changes, projected spend signals, egress signals

Container image is stored in Amazon ECR and referenced by immutable digest during promotion to prevent tag ambiguity. Evidence artifacts are stored in Amazon S3 with versioning and retention controls to support audit retrieval, while cryptographic protection for integrity is provided through AWS KMS-managed keys [13]. Operational audit trails and access logs are captured through AWS-native logging mechanisms, enabling reconstruction of who changed what and when for release-critical actions. Table II summarizes the unified policyas-code domains and the primary inputs evaluated for each domain. Using shared inputs reduces governance fragmentation and improves consistency of promotion decisions.

Security validation is integrated into the pipeline through repeatable checks over infrastructure and application artifacts. Infrastructure changes are evaluated before deployment using policy evaluation on infrastructure definitions and planned changes, and runtime configurations are monitored for drift using continuous configuration assessment. Container and dependency exposure is handled by enforcing that releases contain SBOMs bound to immutable digests, and by applying policy thresholds that block promotion when risk is unacceptable. The same policy layer also enforces governance requirements that are typically treated as "after deployment" concerns, such as mandatory allocation tags and environment-specific constraints on resource sizing and spend envelopes.

FinOps guardrails are implemented as a release criterion rather than a periodic reporting activity. Cost allocation is enforced by rejecting promotions when required tags are missing, and budget adherence is enforced by evaluating projected spend signals derived from environment baselines and planned changes. This approach ensures that financially risky releases do not reach higher environments without explicit exception handling, and it produces policy decision evidence that can be audited alongside security and compliance evidence.

VI. EVALUATION METHODOLOGY

The evaluation is conducted as an illustrative case study using a controlled AWS-based reference environment representative of a healthcare micro service deployment. The baseline is a conventional DevSecOps pipeline that includes standard build, test, and security scanning prior to deployment. The proposed pipeline adds SBOM generation and binding to immutable artifact digests, SLSA-aligned provenance creation and verification, and unified policy-as-code gates that enforce security, HIPAA-relevant control checks, and FinOps guardrails prior to promotion.

To keep the comparison fair, both pipelines use the same codebase, the same functional test suite, the same release cadence, and the same environment topology. The reported results focus on four categories: security effectiveness, compliance evidence completeness, FinOps governance outcomes, and delivery overhead. Security effectiveness is assessed by the rate at which vulnerable or non-compliant artifacts are blocked before promotion. Evidence completeness is assessed by whether each production deployment can be reconstructed from immutable artifact identifiers and accompanying SBOM and provenance records. FinOps governance is assessed by allocation completeness through mandatory tags and by adherence to spend envelopes enforced at promotion time. Delivery overhead is assessed by added pipeline latency at median (p50) and tail (p95) levels.

All values reported are illustrative to demonstrate expected directional outcomes under the stated design assumptions and do not represent a claim about a specific production deployment.

VII. RESULTS AND DISCUSSION

This section reports illustrative outcomes comparing the baseline pipeline and the proposed FinOps-aware DevSecOps pipeline. The results emphasize how SBOM and provenance improve traceability, how policy-as-code reduces risky promotions, and how cost guardrails shift governance earlier in the lifecycle. The discussion also quantifies delivery overhead introduced by additional evidence creation and policy evaluation.

A. Security Outcomes

In the baseline pipeline, releases were primarily blocked by vulnerability scanning when issues exceeded a severity threshold. In the proposed pipeline, releases were additionally blocked when SBOM or provenance evidence was missing or when provenance verification failed policy requirements. Under the illustrative setup, the proposed pipeline increased pre-production blocking of high-risk promotions from 62% to 88%, reflecting the combined impact of evidence requirements and stricter policy gating.

TABLE III. ILLUSTRATIVE COMPARISON: BASELINE DEVSECOPS VS Proposed FinOps-Aware DevSecOps

Metric	Baseline	Proposed
Blocked high-risk promotions (%)	62	88
Evidence completeness per release (%)	70	98
Tagged allocation coverage (%)	76	99
Spend variance vs envelope (%)	18	6
Added pipeline duration p50 (min)	0.0	1.6
Added pipeline duration p95 (min)	0.0	4.2

B. Compliance Evidence Completeness

Audit readiness requires the ability to answer what was deployed, when it was deployed, and under which controls it was approved and verified. In the baseline pipeline, evidence was fragmented across CI logs and deployment metadata, reducing the ability to reconstruct deployments deterministically. In the proposed pipeline, each deployment is linked to an immutable digest with an SBOM and provenance record stored in the evidence repository. In the illustrative results, evidence completeness improved from 70% to 98%, with the remaining gap attributed to early-stage integration errors and incomplete metadata capture in non-critical environments.

C. *FinOps Governance Outcomes*

The baseline pipeline did not enforce cost allocation or spend envelopes at promotion time, which led to untagged resources and periodic spend spikes that were only detected after deployment. The proposed pipeline enforces mandatory tagging and evaluates coast guardrails prior to promotion. In the illustrative results, tagged allocation coverage improved from 76% to 99%, and monthly spend variance versus the predefined envelope decreased from 18% to 6%, driven by rejecting promotions that introduced disallowed resource sizing or missing allocation metadata.

D. *Delivery Overhead*

Strengthening assurance introduces additional pipeline steps for SBOM generation, provenance creation, and policy evaluation. In the illustrative results, the proposed pipeline increased median pipeline duration by 1.6 minutes and p95 duration by 4.2 minutes. The overhead was primarily attributable to evidence generation and verification, while policy evaluation contributed a smaller fraction. This overhead is generally acceptable for healthcare workloads where auditability and controlled change are prioritized, but it should be tuned based on organizational release frequency and risk tolerance.

E. *Summary of Comparative Results*

Table III summarizes the illustrative comparison across the four categories.

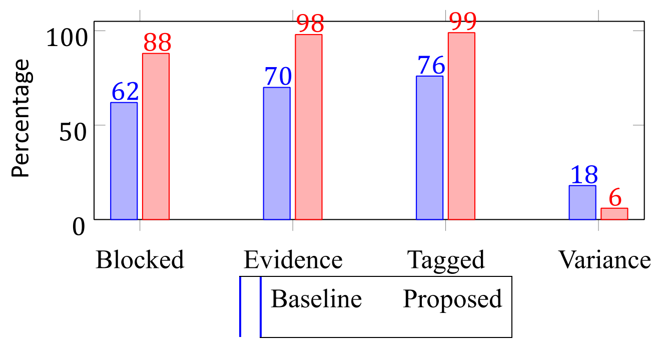

Fig. 3. Illustrative comparison of key metrics (higher is better except variance).

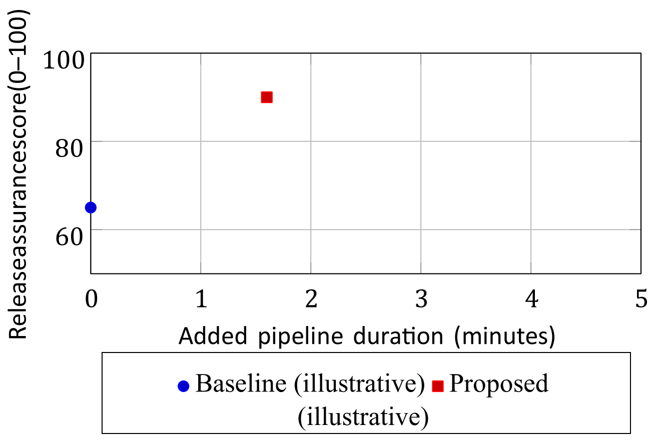

Fig. 4. Illustrative trade-off between added pipeline overhead and an aggregate assurance score derived from evidence completeness and pre-promotion risk blocking.

F. *Visualization*

Fig. 3 visualizes the comparative improvements for the key assurance and governance metrics using the illustrative values.

The results indicate that coupling SBOM and provenance evidence with unified policy enforcement can materially improve auditability and reduce both security and cost risks before deployment. The primary trade-off is modest pipeline overhead, which can be mitigated through caching, parallel validation, and policy tuning while preserving deterministic evidence linkage. Figure 4 illustrates the expected trade-off between additional pipeline overhead and improved release assurance. The proposed approach increases verification work modestly while materially strengthening evidence completeness and pre-promotion risk blocking.

VIII. LIMITATIONS

The proposed approach is intended to improve auditability and governance for HIPAA-relevant healthcare workloads, but its effectiveness depends on several technical and organizational constraints. First, SBOM completeness is bounded by build visibility and dependency resolution fidelity. If the build process pulls dependencies dynamically at runtime, relies on opaque vendor components, or includes unmanaged binaries, the resulting SBOM may be incomplete or may not reflect the precise composition of the deployed artifact. Similarly, SBOM quality can vary across languages and packaging ecosystems, which can affect downstream vulnerability and license analysis.

Second, provenance assurance is only as strong as the integrity of the build environment that produces it. If CI runners are misconfigured, lack isolation, or permit untrusted build steps, provenance records may not provide meaningful guarantees even if they are formally present. Provenance verification also requires consistent identity and key management practices; weak controls around signing keys, role assumptions, or pipeline permissions can reduce the trustworthiness of evidence bundles. In multi-stage builds and complex dependency chains, capturing accurate provenance may introduce additional engineering effort and operational overhead.

Third, policy-as-code enforcement introduces the risk of false positives and workflow friction. Strict policy gates can block releases due to transient scanning results, incomplete metadata, or conservative thresholds that do not reflect application context. While such strictness may be desirable in regulated environments, it can also increase lead time if exception handling and policy tuning are not well-governed. Conversely, overly permissive policies can weaken the intended benefits and reintroduce governance fragmentation. Achieving the right balance requires iterative tuning and clear ownership across security, platform, and finance stakeholders.

Fourth, the FinOps guardrails depend on the accuracy and stability of attribution and forecasting signals used to evaluate spend risk. Tag-based allocation assumes consistent tagging discipline across environments and services, and even small gaps can reduce allocation coverage. Pre-deployment cost estimation is inherently approximate because runtime behavior, workload seasonality, and scaling policies can change consumption after release. As a result, cost policies may either under-block risky changes or over-block legitimate changes unless the organization maintains realistic spend envelopes and continuously refines estimation rules.

Fifth, the evaluation results presented in this paper are illustrative and are used to demonstrate expected directional outcomes rather than measured guarantees in a specific production environment. Real-world outcomes will vary based on workload characteristics, deployment topology, CI/CD tooling, the maturity of security and finance operations, and the strictness of policy thresholds. Additional empirical studies with production-grade telemetry would be required to quantify the benefits across diverse healthcare application types and organizational settings.

Finally, governance and human factors remain critical. The proposed pipeline can enforce controls and generate evidence, but it does not replace the need for disciplined change management, incident response readiness, and clear accountability for exceptions. If teams frequently bypass gates without time bounded justification, or if evidence retention is not treated as an operational requirement, audit readiness can degrade despite strong technical design. These limitations suggest that successful adoption requires not only tooling integration, but also well-defined ownership, periodic policy review, and operational practices aligned with regulated healthcare delivery.

IX. CONCLUSION AND FUTURE WORK

This paper addressed a practical gap in regulated healthcare delivery: security and compliance controls are often embedded into DevSecOps pipelines, while cost governance and supply-chain assurance remain fragmented across separate tools and teams. For HIPAA-relevant workloads on AWS, this fragmentation weakens auditability and increases exposure to both technical risk (dependency compromise, build tampering, artifact substitution, configuration drift) and financial risk (untagged spend, environment sprawl, uncontrolled resource growth). The proposed FinOps-aware DevSecOps pipeline unifies these concerns by treating release promotion as an auditable contract that requires three forms of evidence to travel together from build to production: an immutable artifact identifier, an SBOM that provides component-level transparency, and SLSA-aligned provenance that links the artifact to its source and build context. A single policy-as-code layer evaluates this evidence alongside security and HIPAAoriented control checks and enforces FinOps guardrails as first-class release criteria, thereby shifting cost accountability and supply-chain integrity checks to the same stage where organizations already enforce security gates.

The key outcome is improved end-to-end traceability with a governance model that is repeatable and reviewable. By promoting only digest-pinned artifacts and persisting policy decisions, SBOMs, provenance records, and validation outputs as evidence bundles, the pipeline supports audit-critical questions without relying on manual reconstruction from scattered logs. At the same time, cost governance becomes operational rather than advisory: allocation requirements and spend-risk constraints can block promotion before inefficient configurations reach higher environments. Importantly, the approach is designed to be practical for real delivery teams. It does not require a single CI/CD product, and it allows incremental adoption, beginning with evidence capture and digest-based promotions, and then expanding to stricter policy gating as organizational maturity increases.

Future work will focus on strengthening the fidelity and usability of evidence and improving the precision of policy decisions. On the supply-chain side, this includes richer provenance capture for multi-stage builds, tighter integration of SBOM records with vulnerability and license workflows, and systematic evaluation of how evidence completeness changes under different build and deployment patterns. On the governance side, future work includes extending policy coverage to workload-specific controls commonly observed in healthcare environments, improving exception handling so that necessary overrides remain auditable and time-bounded, and reducing pipeline overhead through parallel validation and caching strategies without weakening determinism. From a FinOps perspective, future work includes refining cost guardrails using workload-aware unit-cost models and more robust pre-deploy estimation techniques, while maintaining conservative controls suitable for HIPAA-oriented systems. Collectively, these directions aim to preserve the central design principle demonstrated in this paper: releases should be promotable only when they carry verifiable supply-chain evidence and satisfy a unified, auditable policy contract spanning security, compliance, and cost.

REFERENCES

[1] K. J. Nahra, "Hipaa security enforcement is here," *IEEE Security & Privacy*, vol. 6, no. 6, pp. 70–72, 2008.

[2] P. Haindl, P. Kochberger, and M. Sveggen, "A systematic literature review of inter-service security threats and mitigation strategies in microservice architectures," *IEEE Access*, vol. 12, pp. 90252–90286, 2024.

[3] D. Burke, "Improving finops procedures with automation tools and framework changes for a cloud environment," Master's thesis, Aalto University, School of Electrical Engineering, Sep. 2024, permanent link: https://urn.fi/URN:NBN:fi:aalto-202411217270. [Online]. Available: https://aaltodoc.aalto.fi/items/0670f49c-3d66-44e7-a2d7-d47c7a314f36

[4] S. H. B. I. Kumar, L. R. Sampaio, A. Martin, A. Brito, and C. Fetzer, "A comprehensive study on the impact of vulnerable dependencies on

open-source software," in *2024 IEEE 35th International Symposium on Software Reliability Engineering (ISSRE)*, 2024, pp. 96–107.

[5] M. Balliu, B. Baudry, S. Bobadilla, M. Ekstedt, M. Monperrus, J. Ron, A. Sharma, G. Skoglund, C. Soto-Valero, and M. Wittlinger, "Challenges of producing software bill of materials for java," *IEEE Security & Privacy*, vol. 21, no. 6, pp. 12–23, 2023.

[6] S. Zhou, H. Wu, and Z. Xue, "Grouped subspace linear semantic alignment for hyperspectral image transfer learning," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–16, 2022.

[7] X. Sun, Y. Cheng, X. Qu, and H. Li, "Design and implementation of security test pipeline based on devsecops," in *2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, vol. 4, 2021, pp. 532–535.

[8] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015, pp. 1–6.

[9] Nellutla, N. (2022). Secure DevSecOps Workflows for Medical IoT Device Integration in Smart Hospitals. *International Journal of AI, BigData, Computational and Management Studies*, *3*(1), 114-122. https://doi.org/10.63282/3050-9416.IJAIBDCMS-V3I1P113

[10] A. Srivastava, "Automated deployment of an end-to-end pipeline on amazon web services for real-time visual inspection using fast streaming high-definition images," Master's thesis, Clemson University, 2019.

[11] S. Yu, W. Song, X. Hu, and H. Yin, "On the correctness of metadatabased sbom generation: A differential analysis approach," in *2024 54th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2024, pp. 29–36.

[12] H. T. Phan and N. T. Nguyen, "A fuzzy graph convolutional network model for sentence-level sentiment analysis," *IEEE Transactions on Fuzzy Systems*, vol. 32, no. 5, pp. 2953–2965, 2024.

[13] R. Rompicharla and B. R. P. V, "Continuous compliance model for hybrid multi-cloud through self-service orchestrator," in *2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)*, 2020, pp. 589–593.

[14] I. Saeed, S. Baras, and H. Hajjdiab, "Security and privacy of aws s3 and azure blob storage services," in *2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS)*, 2019, pp. 388–394.