International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 4, Issue 5, September-October 2022 |

DOI:10.15662/IJEETR.2022.0405005

Engineering Audit-Ready CI/CD Pipelines for Federally Regulated Scientific Computing

Prudhvi Raju Mudunuri

Independent Researcher, USA

ABSTRACT: When the federal regulator controls the scientific computing platforms they must strike a balance between speedy delivery of software and high-quality audit requirements that mandate the presence of evidence that can be replicated, controlled change, and evidence of tampering records. The paper proposes an engineering strategy of constructing audit ready CI/CD pipelines that generate provisable provenance, traceable change histories and cryptographically verifiable artifacts as an objective and not an ex post facto account of such. The managed DevSecOps controls of the strategy are provided across the delivery lifecycle, including policy-as-code compliance gates, automated security and quality assurance, immutable audit trail production, and secure software supply chain to sign and verify artifacts.

The research also illustrates the capability of pipeline instrumentation to offer an end to end lineage between the source commit, through build and test and package and deployment at the same time without violating separation of duties and approval traceability as is required in the IT environment of the public sector based on regulated biomedical scientific computing systems as reference implementations. Evidence is produced continuously in a machine-verifiable format, which is structured, and on which auditors and other authorizing figures can re-assemble what and why was changed, who had authorization, what controls were executed, and what exactly was placed in each environment.

Comparison of release cycles indicates that the standardized evidence package leads to an increase in audit preparedness, the reduction of the hectic authorization schedules by the avoidance of rework and avoidance of records omissions, and transparency of release governance and software assurance stakeholders. The findings show that making auditability a first-class engineering requirement can be used to improve the integrity and reproducibility without totally degrading the delivery cadence. The methodology has been extended to investigate computing infrastructure communities which must match federal specifications such as NIST control families and FISMA-based regulation and provides a plausible path forward to functioning towards operationalizing compliant continuous delivery inside regulated scientific computing setting.

KEYWORDS- Audit-Ready Systems; CI/CD Pipelines; Federal Compliance; DevSecOps; Build Provenance; Artifact Integrity; Immutable Audit Trails; Secure Software Supply Chain

I. INTRODUCTION

Modern government has achieved a core capability in scientific computing, which now gives it the ability to analyze, model, simulate, and support decisions in large volumes of data e.g., in healthcare, climate science, defense, and public safety. These platforms are rapidly ceasing to be fixed research environments and are increasingly being viewed as intricate software ecosystems, continually changing over time due to constant code, model, infrastructure, and data pipeline updates. Many organizations have embraced Continuous Integration and Continuous Delivery (CI/CD) to automate their testing, integration, and deployment to support this rate of change and accelerate the process of innovation and decrease operational risk. Nevertheless, in the cases where the scientific computing systems are run under federal control, the advantages of fast delivery should be balanced with the strict regulatory, security, and audit rules [1] [2].

The environments that are controlled on a federal level have expectations, beyond the functional correctness and performance. Agencies need to show responsibility, recurrence and control on the way software is developed, tested, approved and released. The Federal Information Security Management Act (FISMA) and supporting NIST standards demand that organizations are able to demonstrate evidence of compliance in documents like change control, separation of duties, configuration management, and continuous monitoring [3]. By default, traditional CI/CD pipelines, which are usually focused on maximizing speed and developer productivity, are not able to generate such assurance [4] [5].

The resulting incompatibility results in a continuous conflict between the engineering departments and the compliance stakeholders. Although the practices of DevOps focus on automation, decentralisation and quick iteration, federal audit processes are often viewed as slow, manual and retrospective. The collection of evidence is often viewed as a post hoc exercise and this means that teams need to reassemble release histories, approval records and test results, many years after the fact. This system will make risk and inefficiency: missing or inconsistent critical information, long authorization schedule as documentation is not done, and releases can be delayed or frozen during audit cycles. Such delays can have serious operational or societal impacts in scientific computing settings, in which the system tends to be mission-critical or provide health results to the population.

The aspect of scientific computing platforms presents its own challenge due to technical nature. Such systems can be heterogeneous systems combining custom research code, third-party libraries, containerized services and data processing workflows as well as specific hardware or cloud infrastructure..

The available literature and industry best practices regarding CI/CD and DevSecOps can offer useful practices to enhance the quality and security of software, including automated testing, static analysis, and vulnerability scanning. But these practices are mostly introduced without a background regarding the governance and audit requirements of federally regulated environments. Security tools can produce outputs that are incompatible with the existing aggregation of coherent audit evidence, and pipeline logs might not have the format and immutability necessary to allow formal compliance inspection. In addition, most of the implementations are based on a trust model that suits commercial settings but not in the systems used in the public sector that require external verification. Consequently, there is a habit of using manual compensating controls by organizations to negate the advantages of continuous delivery [6] [7].

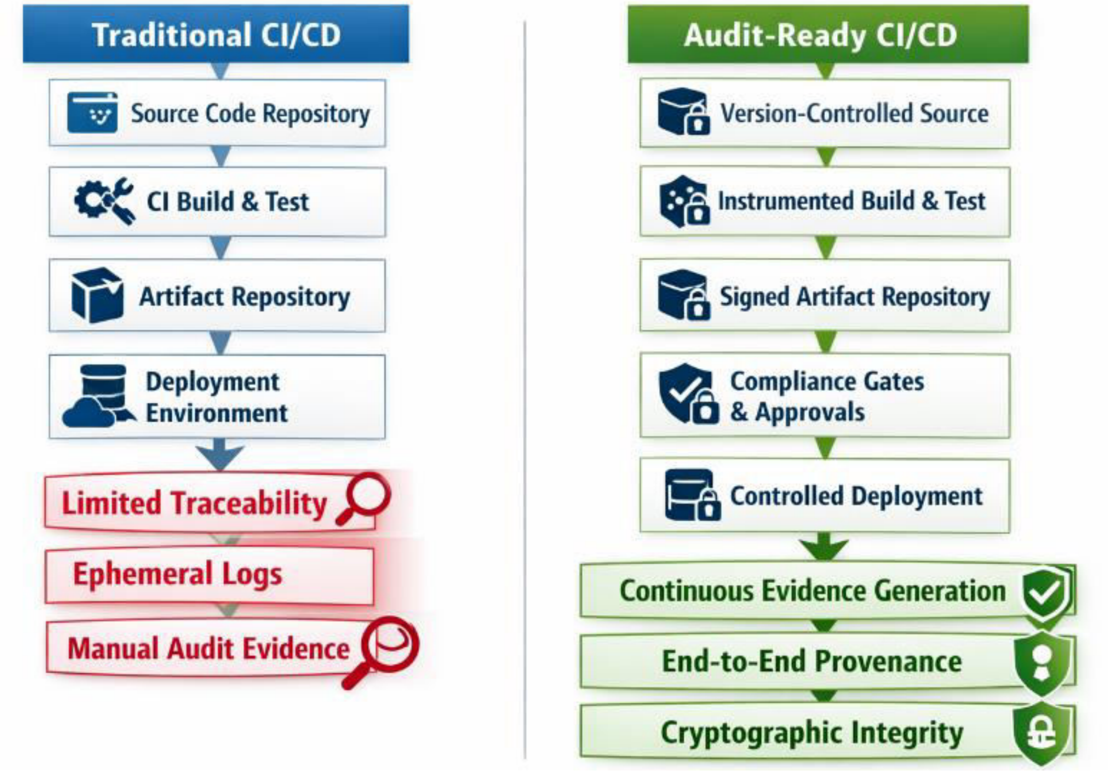

Figure 1: Traditional CI/CD vs Audit-Ready CI/CD in Federally Regulated Environments

In this paper, it is argued that auditability must be considered as a primary engineering requirement and not as an external constraint that must be imposed during post-deployment. The concept of an audit-ready CI/CD pipeline is a pipe that can be audited, but is markedly specifically prepared to produce full, verifiable, and tamper-resistant evidence as an output of the delivery process. Provenance, traceability and integrity are designed into every single stage in such a pipeline so that compliance artifacts are generated continuously and regularly along with functional software artifacts.

We introduce a set of engineering procedures of the creation of audit-ready CI/CD pipelines and adjustment to scientific computing environments that are federally regulated. Four principles form the basis of the methodology. To begin with, the pipeline has to produce end-to-end build and deployment provenance, whereby the origin of the source code, through execution in a target environment, can be traced. Second, any change should be documented in form of structured machine verifiable records, which bind together actions, actors, approvals and outcomes. Third, the integrity of software artifacts should be ensured by cryptographic signing and verification to reduce the supply-chain risk. Fourth, the evidence in audit has to be unchanging and centrally available, to enable independent checking without use of informal stories or manual fabrication.

To show the viability and effectiveness of the practice, we use the methodology to the regulated biomedical scientific computing systems that are under federal regulation. The sensitivity, complexity and strict compliance requirements of these systems such as data protection, controlled releases and formal authorization processes present the representative case. Measurements of audit readiness can be sustained in flowing policy pipelines with continuous output that does not need to reduces the pace at any point on the delivery to the client, and by instrumenting the pipelines, well produce audit packages, and the compliance gates may be enforced, through policy-as-code, the reference implementations demonstrate.

This is evaluated on the basis of practical results that not only are of interest to the engineers but also to the stakeholders of governance. These are completeness and consistency of audit evidence, effort needed to prepare release reviews in authorization, release histories transparency across environment. The findings reveal that introducing audit requirements into the pipeline design minimizes the use of manual documentation, minimizes the time of authorization, and enhances the belief in release integrity. Notably, the methodology contributes to the transition between the episodic and audit-oriented compliance to the ongoing assurance in accordance with the current DevSecOps trends.

This research has three-fold contributions. To begin with, it provides clear definition of audit-ready CI/CD pipelines and determines the engineering attributes needed to ensure compliance with the federal in scientific computing settings. Second, it offers a realistic methodology and architectural designs of using these pipelines, such as provenance capture, artifact signing, compliance gates, and unalterable audit trails. Third, it provides empirical evidence based on controlled reference implementations, which prove the effectiveness of audit readiness and efficiency in delivery.

The rest of this paper is structured in the following manner. Section 2 provides a review of the background on CI/CD, DevSecOps, and federal requirements of compliance. The section 3 explains the suggested audit-ready pipeline process and its architecture. Section 4 shows the reference implementations and evaluation results. Section 5 addresses implications, limitations and opportunities of further adoption to broader adoption of research computing applications in public-sector settings. At the end, the Section 6 has the recommendations to the future work in the field of continuous delivery with security and compliance.

II. RELEVANT BACKGROUND

Continuous Integration and Continuous Delivery in Scientific Computing

Continuous Integration and Continuous Delivery (CI/CD) refer to software engineering methods that are aimed at automating the process of code change integration, testing, and deployment. Continuous Integration focuses on high levels of committing codes to a common repository after which automated builds and automated tests are used to identify defects early. Continuous Delivery builds on this strategy by making sure that the software artifacts are continuously in a deployable form so that a reliable and repeatable release can be made. CI/CD has become significant in scientific computing environments as research platforms are becoming increasingly concerned with long-lived services, sharing in analytical workflows, and computational models, which need frequent updates. Automation assists in minimizing human error and enhancing reproducibility, as well as interdisciplinary teamwork [8].

Traditionally however, CI/CD pipelines have been optimized to be fast and productive by developers, but not governable and audit-friendly. Build outs and pipeline logs are usually temporary, unstructured, or location-specific and therefore constrained in their usefulness as formal evidence. Because of it, the standard practices in CI/CD are to be expanded so that they could address assurance demands of federally regulated settings.

DevSecOps and Secure Software Delivery

DevSecOps builds on DevOps by incorporating security controls into the software development process, instead of security being seen as an upstream and independent process. Examples of the most important DevSecOps practices are automated security testing, dependency and vulnerability scanning, infrastructure-as-code verification, and continuous monitoring. DevSecOps would decrease the security risk, though it maintains delivery speed, by integrating these controls into CI/CD pipelines [9].

DevSecOps also facilitates the security of sensitive information, intellectual property and computer resources in context of scientific computing. It is also able to provide a similar performance in enforcing security baselines in very different settings, such as the on-premises high-performance computing systems and cloud-based research systems. Even with such advantages, in practice, DevSecOps implementations do not pay much attention to audit evidence quality but rather are aimed at reducing risks. Security tools can provide reports that are not easily correlated with certain releases, approvals, or deployed artifacts thus they cannot be effectively used in compliance review [10].

Federal Compliance and Audit Requirements

The systems regulated by the federal agencies are exposed to a broad range of compliance and oversight needs to maintain security, responsibility, and trust among people. Federal Information Security Management Act (FISMA) is a regulation that provides a system of information security risk management in the case of federal agencies and associated systems in the United States. FISMA is implemented in the form of standards and guidelines released by the National Institute of Standards and Technology (NIST) which include configuration management control families, change control control families, access control families, and system integrity control families.

One of the key aspects of federal compliance is that it should provide evidence of control implementation and effectiveness. This encompasses documentation of software changes, approval procedures, test outcomes and deployment documentation. The systems are to be subject to an authorization procedure, including Authority to Operate (ATO), where reviewers should determine the presence of security and governance controls, which are regularly implemented. This in practice usually results in a great deal of documentation, especially when the evidence need to be gathered by hand in different instruments and different teams.

Scientific computing systems also add to this complexity because they are dynamic systems that depend on software that is rapidly changing. Lack of automated provenance capturing and immutable audit trail maintenance mechanisms may lead to noncompliance or a long authorization process in organizations. As a result, there is an increased interest in the methods that will allow incorporating federal compliance requirements into engineering workflows, allowing continuous evidence creation and more efficient audit.

III. METHODOLOGY

Research Approach and Design Rationale

The goal of the present research is to create and test an engineering solution to building CI/CD pipelines, which are inherently audit-ready in the context of federally regulated scientific computing solutions. The research approach is a design-oriented engineering research, integrating architectural design, pipeline instrumentation, and experimental research through real world reference implementations. Instead of suggesting compliance as an overlay of governance, the methodology considers auditability a non-functional requirement, which is deeply implicated into the software delivery lifecycle.

The three design constraints that drive the methodology include: (i) the requirement to provide continuous delivery without compromise of assurance, (ii) the requirement to produce verifiable and reproducible audit evidence and (iii) alignment with federal compliance expectations e.g. FISMA and NIST control families. These limitations define the pipeline architecture as well as the evidence capture, validation and preservation mechanisms.

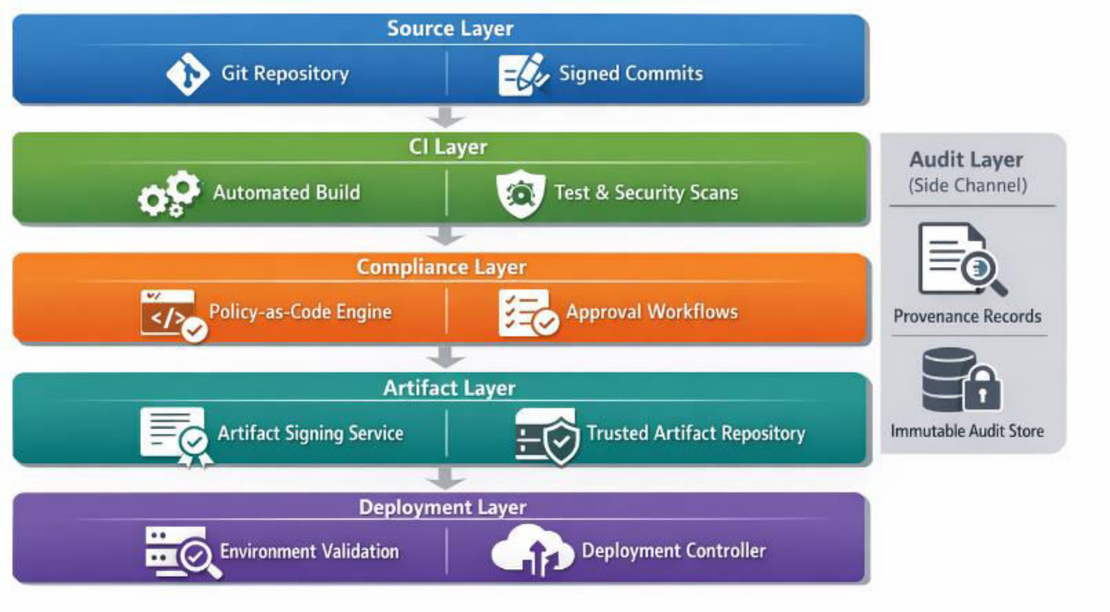

Figure 2: Audit-Ready CI/CD Pipeline Architecture

Reference System Context

It was implemented and assessed in controlled biomedical scientific computers that have been put in place under federal supervision. Such systems are generally used in data processing pipelines and analytical services and research workflows that are regularly updated to meet changing scientific needs. Applications as well as container images often contain application code and logic, definitions of infrastructure as code, configuration artifacts, and dependency updates. The systems are formally authorized and audited on a regular basis and are more representative of larger public-sector research computing environments.

The reference implementations were deployed, with the help of modern CI/CD platforms and references with version control systems, artifact repositories and deployment targets, across the development, staging and production environments. This context enabled the methodology to be assessed in the entire lifecycle of the code change, build, approval, and deployment as well as audit review.

Audit-Ready Pipeline Architecture

It is suggested that an audit-ready CI/CD pipeline is the system that generates complete, traceable, and non-tamper-evidence evidence in an output of the delivery process. The pipeline structure is divided into small phases: source, build, test, approval, package, and deploy that are specific to instrumentation in generating structured evidence artifacts. All of these artifacts create a permanent auditing trail, which can independently be used to verify compliance.

The source stage at all levels is an initial repository of the version being manipulated with identity management enforced and signed commit where available. Metadata that is recorded on this level involves the identifiers of commits, the identity of the author, times, and work links or work requests. This forms the first provenance anchor of those that follow in the pipeline activities.

The build step is the premise which turns useable or deployable artifacts out of source input. Build environments are declared to be reproducible and build metadata consists of the version of the tool used, hash of the dependency, environment identifier and build outcome. Build outputs are identified by a unique number and cryptographically connected to the source commit that is used, so that they can be traced throughout the stages.

Provenance and Traceability Mechanisms

The methodology is based on end-to-end provenance. Provenance is a term that is defined as the entire lineage of an artifact since its origin through deployment, together with all the transformations, validations and approvals. To do so, machine readables provenance information will be emitted by the pipeline at every step, including provenance information about provided input, provided output, performed actions, and the execution environment.

Traceability is implemented by connecting provenance records by immutable identifiers as opposed to informal naming conventions and manual records. As an illustration, a reference to a particular source jump can be found in a deployed artifact which traces back to a particular build and finally refers to approval record. Such connection allows auditors to have a complete lifecycle of any release in an unambiguous manner.

Provenance records are kept in an evidence store of append-only design, which is designed so as to prevent modification or deletion. Access controls help to guarantee the reviewing of evidence is not contingent upon the engineering teams and facilitate separation of duties and audit integrity.

Cryptographic Artifact Integrity

To mitigate supply-chain risk and artifact manipulation, the cryptographic validation is a mandatory pipeline control, so that the methodology can combat both. On the successful passing of pipeline stages that are required, all build artifacts such as container images and deployment packages are cryptographically signed. Signature verification before deployment is implemented with the result that only signed and non-modified artifacts are deployed to controlled environments.

The provenance records are associated with artifact signatures and stored together with audit evidence. This allows the auditors to not only ensure that what was deployed was what was actually deployed, but also that the deployed artifact is the same as that which was created and certified during the pipeline execution. The methodology will decrease the environmental trust assumptions and manual verification by integrating cryptographic integrity checks in the pipeline.

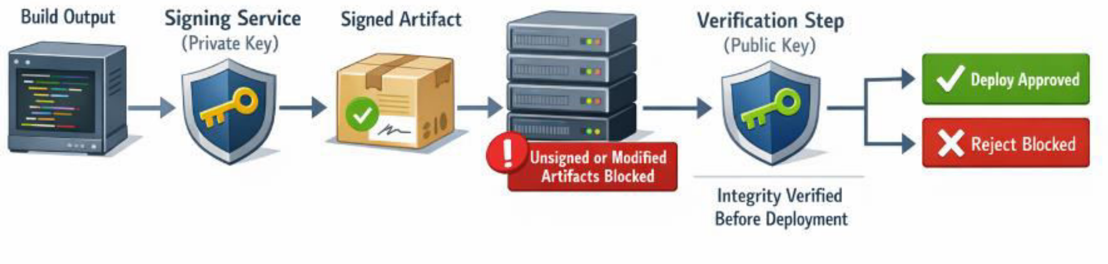

Figure 3: Cryptographic Artifact Integrity and Supply Chain Protection

Compliance Gates and Policy-as-Code

Federal compliance requirements tend to dictate clear approval procedures, separation of duties and imposition of controls before an deployment. The requirements are operationalized by the methodology by means of compliance gates that are realized as policy-as-code. Policies specify terms within which a pipeline can go through, e.g. successfully passing security scans, having necessary approvals, or signature of artifacts.

The workflows of approval are directly implemented into the pipeline with identity (identity of an approver) and approval timestamps and approval scope being a structured evidence. This also removes informal approval systems and makes the process of authorization decisions always recorded and auditable. Policies are version-managed, and it is possible to trace the changes in governance over time.

The pipeline establishes uniformity in the application of control by codifying compliance requirements, which is enforced in the release but also gives the policy the opportunity to vary with the change in regulatory expectations.

Immutable Audit Trails and Evidence Management

The formation of irrevocable audit trails is one of the core aspects of the methodology. Provenance records, test results, approvals and deployment logs are all stored in evidence repository that is tamper proof and can assist in long term retention and review. The production of evidence takes place automatically and on a continuous basis, which lowers costs of manual generation when performing audit cycles.

The artifacts of evidence are designed in a way that enables the human review and automated analysis. This two-purpose plan allows reminders, dashboards and compliance reporting without compromising the fact that auditors are able to independently audit raw records. The audit trail is immutable which promotes non-repudiation and enhances confidence in the delivery process.

Evaluation Methodology

A qualitative and operational assessment of the methodology was done through various release cycles of the reference systems. The evaluation criteria were oriented at three main dimensions that are audit readiness, authorization efficiency, and release transparency.

The level of audit preparedness was determined by observing the completeness and consistency of evidence produced by the pipeline and the amount of effort involved in creating audit packages. The efficiency of the authorization was measured by comparing the time and rework of the authorization reviews prior to and following the pipeline adoption. Transparency of release was evaluated by the capability of the stakeholders to trace released artifacts to their origin, approvals and validation outcomes.

The sources of data or information used in the task included pipeline logs, evidence depositories, and responses provided by stakeholders working in the engineering and compliance departments. Although the assessment does not make the assertion of statistical generalizability, it offers practical information of the methodology effectiveness in an actual regulated environment.

IV. IMPLEMENTATIONS AND EVALUATION RESULTS

The audit-ready CI/CD process proposed was introduced and tested on the basis of controlled biomedical scientific computing systems that were subject to federal control. The settings are typical of public-sector research infrastructure in the sense that they are formalized with respect to authorization, regularly audited, and rigidly controlled in terms of software changes and deployments.

The pipelines of CI/CD were introduced in the industry standard tooling coupled with the version control systems, artifact repository systems, and deployment platforms across development, staging, and production environment. The implementation focused on causing minimal disturbance to the current engineering operations, and gradually, audit-oriented instrumentation was implemented. Instead of replacing the existing pipes, they were extended and the comparison of pre-existing delivery practices with the audit-ready approach could be made.

All pipeline stages were instrumented to produce the structured artifacts of evidence, such as the provenance records, test and validation outcomes, the approval metadata and deployment confirmations. At the packaging stage and deployment stage artifact signing and verification schemes were introduced so that artifacts that had been cryptographically verified could only enter controlled environments. Policy-as-code was used to implement compliance gates to enforce such mandatory controls as approval requirements, security validation, and segregation of duties.

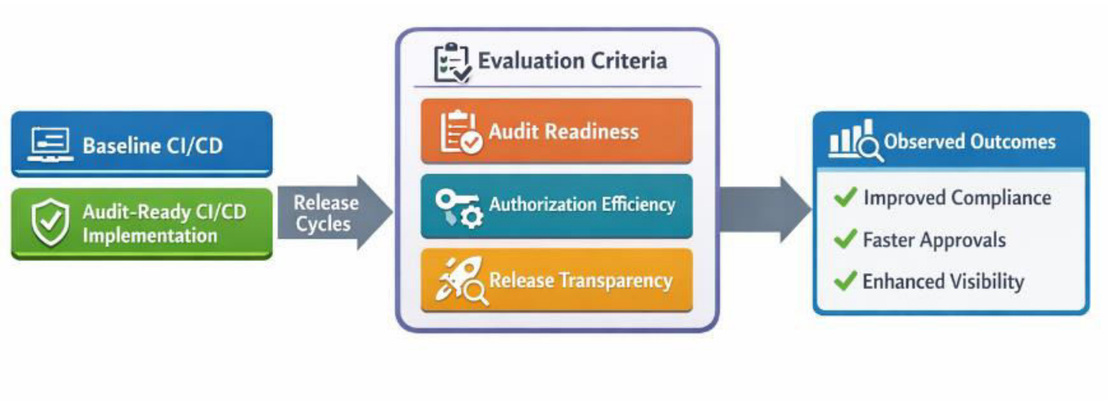

Figure 4: Implementation and Evaluation Workflow

Evidence Generation and Audit Readiness

Among the main improvements that were seen in reference implementations was a significant enhancement in audit readiness. Before switching to the methodology, audit preparation had to be done manually by combining evidence in many different systems, version control logs, CI/CD dashboards, ticketing systems, informal approval records, etc. This was a time consuming process that was prone to error and would highlight the loopholes in traceability or lack of documentation towards the end of the review process.

The pipelines ready to be audited were automatic in the creation of evidence, and they were created part of the pipeline execution. The output of every release was a complete and self-consistent ballot relating source dedication, construct metadata, examination findings, approvals, and documentation of deployments. With help of evidence store, auditors and compliance stakeholders could create release histories without extra information provided by engineering teams. This was a change in episodic to continuous evidence production that minimized doubt and enhanced trust in the honesty of the delivery process.

Authorization Efficiency

The authorization efficiency was assessed by looking into the effort and time used to facilitate the reviews of authorization and the release approvals. Although it depends on the organizational setting, formal authorization timelines were characterized by decreased rework and clarification requests among stakeholders in the process of review. Reviewers could also concentrate on the effectiveness of the control as opposed to the completeness of the evidence since the necessary evidence was always available and traceable.

There were also fewer release delays in the engineering teams that were related with late compliance findings. Gateway compliance was forced in the execution of pipelines to avoid non-conforming releases and thus lower probabilities of remedies after an application has already been deployed. Consequently, the authorization processes became more predictable and consistent with the delivery timelines and assisted a more streamlined release flow within the regulated context.

Release Transparency and Traceability

Its implementation increased transparency to a large extent with respect to release cycles. The stakeholders such as the engineers, security personnel and compliance reviewers were able to respond to critical governance questions with very little effort; what was deployed, when it was deployed, who approved it and which controls were implemented. The end-to-end traceability allowed quick problem investigation, as well as accountability without use of informal communications and tribal knowledge.

Incident response as well as rollback processes were also enhanced through deployment traceability. Due to cryptographic linking of deployed artifacts to particular builds and source commits, teams were able to find out the provenance of running parts quickly and gauge the effects of a change. This has become especially useful in scientific computing settings that are more complex and the components and workflows change at the same time.

Observed Trade-offs and Operational Impact

Although the methodology provided some good governance and assurance advantages, the implementations also showed some trade-offs. The early pipeline instrumentation and policy definition took initial engineering time and interdependent development, security, and compliance work. This investment was however compensated by a large percentage in cut back in manual effort during audits and cut back in disruptions during release cycles.

Significantly, the speed of delivery did not reduce drastically when the pipelines had settled. Compliance controls and approvals were automated and hence, the human bottlenecks were minimized and the teams could manage their regular releases and at the same time, comply with the requirements of the oversight. The pipelines, which were audit-ready, over time led to a change of culture whereby compliance was considered as part of engineering quality and not as an external constraint.

Summary of Evaluation Findings

All in all, the reference implementations indicate that audit-ready CI/CD pipelines can be practical and efficient in scientific computing environments that are regulated by the federal government. The methodology enhanced audit readiness by constantly generating evidence, transparency and traceability across releases via increased authorization efficiency by lessening review friction and improved transparency and traceability. These findings are indicative of the fact that the congruency between the needs of fast software delivery and the assurance expectations of the federal control can be achieved by implementing the requirements of audit directly into the design of the pipeline.

V. IMPLICATIONS, LIMITATIONS, AND OPPORTUNITIES

Implications for Federally Regulated Scientific Computing

This research has a number of significant implications on the engineering and governance of scientific computing systems that are run under the federal jurisdiction. First, the findings prove that continuous delivery and audit readiness do not necessarily go against each other. The treatment of auditability as a first-class engineering requirement allows organizations to transition to the reactive compliance models where documentation is a major element to the proactive, continuously assured delivery models. Such a reframing can ease tensions between the engineering team and compliance stakeholders and become the owner of governance results together.

Second, the methodology demonstrates the process in which DevSecOps practices can be generalized to deal beyond security risk mitigation to enable formal audit and authorization. Combination of provenance capturing, artifact signing and compliance gates at the policy as code level offer tangible avenue in operationalizing abstract regulatory policies like traceability, separation of duties, and configuration control. In the case of the public-sector research organizations, the given approach favors increased transparency and accountability and retains the flexibility required to serve the shifting scientific missions.

Lastly, the discoveries indicate that the pipelines ready to an audit may enhance organizational resilience. Enhanced traceability and artifact integrity are also supportive of audits in addition to strengthening incident response, rollback, and root-cause analysis. These features have found application especially within scientific computing applications, where accountability can otherwise be lost amidst a complex and rapidly changing system, and an operational risk is abetred.

Limitations of the Study

This study has a number of limitations though it has been able to do so. It is also evaluated against only a few reference implementations in controlled biomedical scientific computing environments. Although these systems are typical of more general public-sector research settings, the findings cannot be fully generalized to other setting with alternative regulatory regimes, organization structures, or technical limits.

Also, in the evaluation, there is a focus on qualitative and operational results, but not quantitative performance indicators. Although the stakeholders have claimed that audit preparedness and authorization efficiency has improved, future research might consider a more systematic measurement of the time savings, cost reduction, and error rates in a larger sample of organizations. The research also presupposes the having of an initial level of CI/CD and DevSecOps maturity; organizations that lack extensive automation facilities might have more problems with implementing the methodology.

Lastly, the interpretation and application of regulations differ among different agencies and jurisdictions. In spite of the fact that this approach is consistent with the well-established federal models, including FISMA and NIST standards, it still does not exclude the necessity of making decisions on contextual basis by authorizing officials and auditors.

Opportunities for Future Work

Findings of this study indicate that there are a number of prospects of further study and real-life development. The next step in research could be automated mapping of pipeline evidence to particular control frameworks and reading compliance reports in real-time and assessing risks. Further studies may also be used to investigate how audit ready pipelines can be used in other regulated fields, like environmental monitoring, defense research, or financial analytics.

The new software supply chain standards and tooling can also be incorporated to enhance further the artifact provenance and integrity. Lastly, longitudinal research on the effects of organizational adoption across time would give a better understanding of cultural and governance effects with how the audit-ready engineering practices affect trust, collaboration and sustainability of the system in the long run.

VI. CONCLUSION AND FUTURE WORK

The study covers a pressing problem in the federally controlled science computing settings: the necessity to reconcile high speed software delivery and stringent audit, security and control measures. The study proves that compliance and continuous delivery do not need to be conflicting priorities but instead they can be viewed as complementary. The suggested solution integrates provenance capture, verifiable change histories, cryptographic integrity of artifacts and impermanent audit trails directly into the software delivery life cycle to allow continuous assurance by design.

The methodology was demonstrated by reference implementations in controlled biomedical scientific computing environments, to increase audit preparedness, to enhance the transparency of releases, to lower the friction associated with authorization and review, etc. The pipelines create verifiable evidence at every delivery point as opposed to using manual, post release documentation and instead the pipelines automatically create such evidence. The change will facilitate predictable release cycle, accountability, and confidence amongst the engineering, security and compliance stakeholders. Notably, it was shown that these advantages could be attained without significantly compromising the delivery speed, which served as an example of the possibility to integrate governance controls into the contemporary DevSecOps work.

The results add to both research and practice since they re-conceive auditability as an engineering issue and offer specific architectural patterns of operationalization of federal compliance requirements in CI/CD pipelines. In the case of public-sector research organisations, the approach provides a feasible way to the secure, reproducible, and governable service delivery of software that is in line with the changing scientific and mission requirements.

This research can be furthered in a variety of ways in the future. A wider scope of quantitative research involving more agencies, and more scientific fields would enhance generalizability and allow achieving more rigorous measurements of efficiency outcomes and risk alleviation. Further studies would research the idea of automated mapping of evidence produced by a pipeline to particular regulatory control frameworks to aid in real-time assessment of compliance and adaptive authorization models. New software supply chain standards and attestations are also avenues of further improvement of artifact provenance and cross-organizational trust. Lastly, longitudinal research on organizational adoption and cultural influence would be informative on the determination of how audit-ready engineering practices affect cooperation, governance maturity and longevity of federally controlled scientific computing environments.

REFERENCES

[1] National Institute of Standards and Technology, *Strategies for the Integration of Software Supply Chain Security in DevSecOps CI/CD Pipelines (NIST SP 800-204D)*, Gaithersburg, MD, USA, 2021. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-204D.pdf

[2] National Institute of Standards and Technology, *Secure Software Development Framework (SSDF) Version 1.1 (NIST SP 800-218)*, Gaithersburg, MD, USA, 2021. [Online]. Available: https://csrc.nist.gov/publications/detail/sp/800-218/final

[3] Executive Office of the President, *Executive Order 14028: Improving the Nation's Cybersecurity*, Washington, DC, USA, May 12, 2021. [Online]. Available: https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/

[4] U.S. Department of Defense, *DoD Enterprise DevSecOps Playbook*, Washington, DC, USA, Oct. 2021. [Online]. Available: https://dodcio.defense.gov/Portals/0/Documents/Library/DevSecOps_Playbook.pdf

[5] Cybersecurity and Infrastructure Security Agency (CISA) and National Security Agency (NSA), *Defending Continuous Integration/Continuous Delivery (CI/CD) Environments*, Fort Meade, MD, USA, 2021. [Online]. Available: https://media.defense.gov/2021/Jul/13/2002804273/-1/-1/0/CSI_DEFENDING_CI_CD_ENVIRONMENTS.PDF

[6] National Institute of Standards and Technology, *Security and Privacy Controls for Information Systems and Organizations (NIST SP 800-53 Rev. 5)*, Gaithersburg, MD, USA, 2020. [Online]. Available: https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final

[7] Office of Management and Budget, *Federal Information Security Modernization Act (FISMA) Implementation Guidance*, Washington, DC, USA, 2020. [Online]. Available: https://www.whitehouse.gov/omb/information-for-agencies/federal-information-security-management-act/

[8] OWASP Foundation, *DevSecOps Guideline*, 2021. [Online]. Available: https://owasp.org/www-project-devsecops-guideline/

[9] OWASP Foundation, *Top 10 CI/CD Security Risks*, 2021. [Online]. Available: https://owasp.org/www-project-top-10-ci-cd-security-risks/

[10] National Institute of Standards and Technology, *NIST Risk Management Framework for Information Systems and Organizations*, Gaithersburg, MD, USA, 2020. [Online]. Available: https://csrc.nist.gov/projects/risk-management/about-rmf