OpsMx

WHITEPAPER

Using the CI/CD Process to Achieve NIST 800-53 Compliance:

A Guide to Secure Software Delivery

Implementing Software Supply Chain Security in Modern Business

Executive Summary

In today's digital world, the software supply chain is both an innovation engine and a target for cyber-attacks. With 61% of businesses impacted by a supply chain threat in the last year, the risks are far from theoretical. Against this backdrop, the National Institute of Standards and Technology's (NIST) 800-53 standard offers a robust framework for enhancing cybersecurity measures. This whitepaper delves into the critical areas of software delivery and deployment, pinpointing vulnerabilities in these stages and offering NIST 800-based solutions. The paper recommends focusing on key NIST control families for fortifying the 'last mile' of software delivery. Through case studies and technical insights, the paper provides a comprehensive strategy for organizations to secure their software supply chains against evolving threats.

Table Of Contents

Introduction

In today's digitally driven landscape, the software supply chain serves as both a cornerstone of innovation and a potential vulnerability. The risks are very much present, driven by a significant escalation in attacks targeting open-source software. An estimated 61% of businesses have been impacted by a software supply chain threat in the past year alone[1]. These attacks have global repercussions, causing not just substantial financial losses but also consequences that extend beyond mere monetary metrics.

Cost of software supply chain attacks could exceed $46B this year

Losses attributed to software supply chain attacks will jump 76%, reaching almost **$81** billion by 2026, according to Juniper Research.

Given these alarming trends, the National Institute of Standards and Technology (NIST) 800-53 standard stands out as a crucial guidepost. Designed to assist federal agencies and diverse institutions, NIST 800-53 offers an exhaustive framework to enhance cybersecurity measures, ensuring they remain robust and adaptable in the face of evolving threats.

Today's reality is that the software supply chain is rife with potential vulnerabilities. The intricate path of code, from its development to deployment, presents numerous opportunities for malicious exploitation if not properly managed. This is where the NIST 800-53 series comes into play, offering comprehensive guidelines that address the unique challenges posed during software delivery and deployment. From ensuring the integrity of software components, mandating rigorous audit trails, and managing third-party elements to rigorous vetting processes, following NIST 800-53 is the cornerstone for comprehensive cybersecurity, encompassing all aspects of protection, prevention, and resilience.

In the following sections, we will delve deeper into the relationship between NIST 800-53 compliance and software delivery and deployment, charting a course for organizations to use their CI//CD process to bolster their defenses against the ever-looming specter of supply chain attacks.

Securing the Software Supply Chain: Delivery and Deployment

Secure software delivery and deployment is essential in today's digital landscape, and understanding this process begins with defining the software supply chain. The software supply chain is a continuum of processes encompassing software application development, testing, delivery, deployment, and maintenance. While each phase is vital, an increasing awareness of vulnerabilities associated with the delivery and deployment stages is often regarded as the "last mile" in software release.

Before the advent of modern DevSecOps practices, security was often handled by a separate, siloed team that typically got involved late in the deployment phase. This reactive approach meant that security concerns were frequently addressed only after the development and operational stages were nearly complete, leading to delays and potential vulnerabilities. The 'Shift Left' paradigm was revolutionary in breaking down these silos, encouraging Developer and DevOps teams to incorporate security measures earlier in the software lifecycle.

[1] https://www.capterra.com/resources/software-supply-chain-attacks/

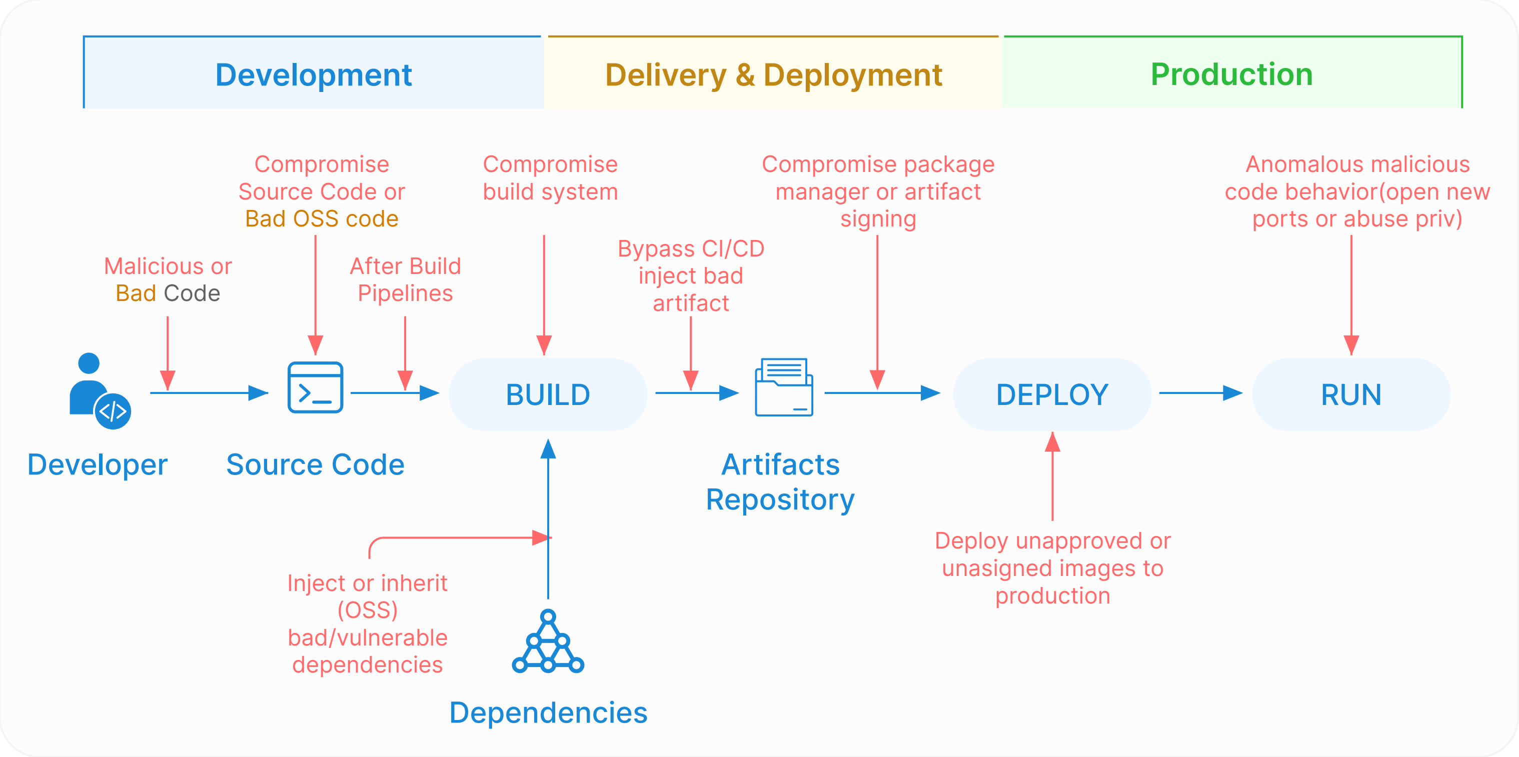
Development
Delivery & Deployment
Production
Malicious or Bad Code
Compromise Source Code or Bad OSS code
After Build Pipelines
Compromise build system
Bypass CI/CD inject bad artifact
Compromise package manager or artifact signing
Anomalous malicious code behavior(open new ports or abuse priv)
Developer
Source Code
BUILD
Artifacts Repository
DEPLOY
RUN
Inject or inherit (OSS) bad/vulnerable dependencies
Dependencies
Deploy unapproved or unasigned images to production

Image 1: Expanding attack surface demands a comprehensive approach to secure every stage of the software supply chain

Recent software supply chain attacks underscore the perils lurking in the last mile, reminding us that the software lifecycle's post “Build” stages are neither immune nor less vulnerable.

Historical Case Studies: The Perils of Neglecting End-to-End Security

The SolarWinds Incident: A Testament to Vulnerabilities
Spanning December 2020 to March 2021, the SolarWinds breach starkly illustrates these risks. Malicious actors exploited the SolarWinds Orion platform, discreetly embedding a backdoor that facilitated widespread unauthorized access to client networks, data theft, and malware seeding. The aftermath was not just technical; the reputational damage and subsequent decline in customer trust had significant business ramifications.

SSRF Attacks: Unveiling the Cloud's Underbelly
From 2022 onward, Server-side Request Forgery (SSRF) attacks emerged as another prominent threat, especially within cloud deployments. These attacks manipulate servers into executing unintended HTTP requests, opening doors for unauthorized data breaches and ransomware. Misconfigured deployment settings, like unchecked outgoing requests, often lay the foundation for these breaches, causing profound business disruptions.
These incidents emphasize a crucial truth: securing the early stages of software development, while vital, is not enough. The entire supply chain demands rigorous security measures from development to deployment.

Shift Left & Beyond: Completing the Security Journey

The shift left methodology has paved the way for proactive vulnerability detection early in the development process.

DevOps initially emphasized rapid deployments and feature releases, often relegating security to a secondary role. The "shift-left" movement integrated security into DevOps teams to address this imbalance. While a step forward, this focus on early-stage security distributes security responsibilities across many development teams that need a unified, holistic view of application security. Thus, while crucial, the "shift-left" approach inadvertently underscores the need for comprehensive, end-to-end security—particularly during the delivery and deployment stages—to ensure no unaddressed vulnerabilities.

The Software Delivery Security Gap

Spanning December 2020 to March 2021, the SolarWinds breach starkly illustrates these risks. Malicious actors exploited the SolarWinds Orion platform, discreetly embedding a backdoor that facilitated widespread unauthorized access to client networks, data theft, and malware seeding. The aftermath was not just technical; the reputational damage and subsequent decline in customer trust had significant business ramifications.

Bridging the Gap: Security Concerns Between Build and Deployment

To bridge this gap, it's essential to recognize and respond to the dynamic threat landscape that can cause new vulnerabilities to attack surfaces between build completion and release into production. The multifaceted nature of deployment environments further complicates security efforts, as each stage—from testing to staging to production—can reveal different challenges and potential risks. Furthermore, the integrity of software artifacts remains a crucial consideration; a repository's security must be uncompromised to ensure that the final deployed product is free from tampering or malicious alterations.

Closing the Security Loop: Benefits of End-to-End Protection

End-to-end security protection offers numerous advantages. It allows for real-time adaptability, ensuring that security measures are up-to-date with the latest threat intelligence throughout the delivery phase. By conducting thorough security assessments up to the very brink of release to production, organizations can significantly narrow the window of opportunity for potential attacks. This comprehensive approach does not replace the shift-left strategy but rather enhances and extends it, establishing a multi-layered defense that envelops the software from its initial development stages to its final deployment.

Closing the Security Loop: Benefits of End-to-End Protection

- **Real-time Adaptable Security:** Protecting the delivery phase ensures dynamic risk evaluations, adjusting defenses to match the latest threat intelligence. By integrating security into the delivery process, organizations can quickly adapt to new threats and vulnerabilities.
- **Minimized Attack Window:** By extending security checks to the edge of deployment, the potential breach window is notably curtailed. By deploying security measures at the last mile, organizations can minimize the amount of time that attackers have to exploit vulnerabilities.
- **A Complement to Shift Left:** While Shift Left provides a foundational security layer at the beginning, securing the subsequent stages, especially the last mile, ensures that the software is shielded from start to finish. Shift Left is a security approach that focuses on integrating security into the software development process from the very beginning. By securing the last mile, organizations can complement the security benefits of Shift Left and create a comprehensive security posture for their software supply chain.

By integrating the early strengths of Shift Left with the rigorous protection of the delivery process, a comprehensive shield is established around the software supply chain. It's about weaving these strategies together to offer an end-to-end defense against relentless cyber adversaries.

Implementing NIST 800-53 Compliance Controls during Software Delivery and Deployment

The National Institute of Standards and Technology (NIST) stands out as a globally recognized authority on information security, with its NIST 800-53 standard offering guidelines, standards, and recommendations designed to fortify the security and resilience of information systems. NIST 800-53 includes over 1100 controls organized into 20 control families that ensure an organization's infrastructure and operating processes are secure. It also lays the foundation for compliance with specific regulations such as HIPAA or FISMA. Several reasons make NIST 800-53 particularly suitable for guiding secure software delivery:

1. **Comprehensive Nature:** With over 1,100 individual controls, NIST 800-53 has the depth and breadth to address the intricate nature of contemporary information security challenges.
2. **Universality and Adaptability:** NIST's guidelines are versatile and can be applied to various industry sectors and contexts, making it a universally recognized benchmark.
3. **Continuous Evolution:** NIST's commitment to updating its standards means it stays in tandem with the evolving threat landscape.
4. **Integration with Other Standards:** As we delve deeper into the next chapter, NIST 800-53 overlaps with and seamlessly maps to other industry standards.

In short, NIST 800-53 is an efficient, high-impact starting point for security compliance.

Using the CI/CD Process to Implement NIST 800-53 Controls

While the expansive nature of NIST 800-53 is powerful, no single product or solution can implement all 1,100+ controls. Conversely, any single control can be implemented with various processes and technologies. The key is identifying which controls can be best implemented by which tools and at which stage of the software lifecycle. The "last mile" CI/CD process is particularly well suited to implementing key elements of eight NIST 800-53 control families.

Control Family	Description
AC (Access Control)	Manages and controls access to information and resources.
AU (Audit and Accountability)	Creates and reviews system audit logs.
CM (Configuration Management)	Manages security features through controlled changes.
IA (Identification and Authentication)	Validates users, processes, or devices.
SI (System and Info Integrity)	Maintains system and information integrity.
SC (System and Comm Protection)	Protects information in transit and at rest.
SA (System and Services Acquisition)	Emphasizes security in system and service acquisitions.
CA (Security Assessment and Authorization)	Regularly assesses security controls and authorizes operations.

Table 1: Control Families that can be implemented in Relevant to Software Delivery and Deployment

These eight control families are included in the NIST Special Publication 800-53B - Control Baselines for Information Systems and Organizations. They serve as our recommended control baseline for CI/CD, designed to assist organizations in aligning their software delivery processes with security and privacy risks.

They provide a well-rounded approach covering various security needs, from access control to system integrity, while leveraging CI/CD best practices. NIST's extensive framework also includes additional control families that can be employed based on your organization's specific needs or unique security landscape to achieve a more comprehensive security posture.

Let’s explore each family to provide further context to the capabilities and use cases they bring to apply the CI/CD process to achieve overall information systems security.

NIST 800-53 Access Control (AC)

Overview

The Access Control family of NIST controls is the cornerstone for maintaining a secure and reliable software delivery process by ensuring only authorized personnel can interact with the system. This is particularly crucial during the 'last mile' of software delivery, the critical stages between the completion of the build process and the deployment to a live production environment, where configurations are finalized, dependencies are verified, and the codebase undergoes its final security checks before going live. During these post-build and pre-production phases, the system is particularly vulnerable to security risks that can compromise the application being deployed as well as the underlying infrastructure.

Lack of stringent access control measures during these phases can result in multiple security and operational issues:

1. **Unauthorized Access:** Without proper controls, unauthorized users may gain access to sensitive parts of the system, posing a risk of malicious activities.
2. **Data Breaches:** Improperly managed permissions can expose sensitive data, leading to potential leaks or unauthorized data manipulation.
3. **System Tampering:** Unintentional and intentional changes can be made to the system, introducing vulnerabilities or malicious code into the production environment.
4. **Operational Disruptions:** Unrestricted access can also lead to unintentional errors, causing operational disruptions that can be costly to rectify and can delay production deployment timelines.

By integrating robust Access Control measures into the CI/CD pipeline, organizations can significantly enhance their security posture, ensuring that each phase of the delivery process—from code development and testing to deployment—is conducted in a secure, controlled environment.

The CI/CD process can be used to implement key controls within this family:

- AC-5: Separation of Duties
- AC-6: Least Privilege
- AC-8: System Use Notification

Control-by-Control Analysis

AC-5: Separation of Duties

Technical Description: AC-5 mandates the division of high-level privileges among multiple roles and individuals to reduce the risk of a single point of failure or compromise. It's designed to prevent a single individual or role from having undue influence or control over a system, thereby reducing the potential for malicious or erroneous actions.

Role-based access controls can be enforced effectively in a CI/CD pipeline, dividing permissions among different roles based on their job functions and limiting who can push software releases into which environments. For example, developers may be allowed to promote applications to a QA environment but not to staging. This control aligns well with other compliance frameworks like ISO 27001, PCI DSS, and HIPAA, and its implementation enhances security and compliance.

AC-6: Least Privilege

Technical Description: AC-6 insists that users should be granted only the most minimal accesses—or permissions—necessary to accomplish tasks.

Least Privilege is crucial for minimizing the potential attack surface by limiting access only to what is necessary for performing a specific task. In a CI/CD context, tools can be configured to restrict the ability to run pipelines that trigger application releases or system changes. Tools can also limit deployment workflows, thereby increasing operational integrity and reducing the likelihood of unauthorized changes. Implementing this control effectively narrows the potential for exploitation and aligns with other regulations like GDPR, ISO 27001, and CIS benchmarks.

AC-8: System Use Notification

Technical Description: AC-8 requires that users be notified regarding acceptable system activities and usage policies upon system login.

This awareness is particularly important in Continuous Delivery environments, where rapid code changes are the norm and a single mistake can have far-reaching implications. CI/CD systems can provide a continuous record of who has executed which stages of the process. Code scanning tools in CI/CD pipelines can automatically flag and notify developers about insecure practices, thereby improving the organization's overall security posture. This control aligns well with frameworks like GDPR and the Sarbanes-Oxley Act, which focus on data protection and integrity.

NIST 800-53 Audit and Accountability (AU)

Overview

The Audit and Accountability family of NIST controls is essential for maintaining a secure, transparent, and accountable software delivery process. The critical requirement is to capture a comprehensive record of the end-to-end process. As any software delivery and deployment must go through the CI/CD process, this is an ideal point of audit. The stages following the build process and leading up to production deployment are also where multiple stakeholders interact with the system, from developers to operations to security teams. These interactions can range from code reviews and security scans to deployment activities.

This is not merely a compliance requirement but a cornerstone of operational integrity and security. By ensuring a comprehensive Audit and Accountability structure through their CI/CD process, organizations can accomplish several critical objectives:

1. **Traceability:** Tracing every action back to its origin is invaluable for debugging and security analysis. This becomes especially important when investigating issues that may have been introduced during the final stages of the software delivery process.
2. **Forensic Readiness:** In cases of security incidents or system failures, well-structured audit logs serve as a critical resource for forensic analysis, enabling rapid identification and remediation of issues.
3. **Accountability:** By keeping a detailed record of who did what and when, accountability is enforced across all team members. This is particularly important for ensuring that only authorized changes are made during critical phases like the pre-production environment setup.
4. **Regulatory Compliance:** Detailed auditing is often required to comply with various industry standards and regulations, such as GDPR, ISO 27001, or PCI DSS. With proper auditing, organizations can avoid security risks and legal repercussions.
5. **Operational Efficiency:** Effective audit mechanisms can also improve operational efficiency by making it easier to identify bottlenecks, inefficiencies, or errors in the pipeline, leading to more streamlined and secure operations.

In summary, with the Audit and Accountability controls, organizations can ensure a secure, efficient, and accountable software delivery process, safeguarding both the product and the production environment.

The CI/CD process can be used to implement key controls within this family:

- AU-2: Audit Events
- AU-6: Audit Review, Analysis, and Reporting

Control-by-Control Analysis

AU-2: Audit Events

Technical Description: AU-2 mandates the detailed logging of specific events, particularly those with security implications, to create an audit trail that can be reviewed and analyzed.

AU-2 is centered around capturing the details of various events, especially those with security implications. This becomes increasingly important in Continuous Delivery environments where rapid deployments can occur without a proper audit trail. Within a CI/CD framework, this control can be implemented by integrating logging mechanisms that capture essential details such as who approved the deployment, when it occurred, and what artifacts were deployed. These logs are invaluable for forensic investigations and support compliance with other frameworks like ISO 27001, which similarly mandates detailed logging and monitoring.

AU-6: Audit Review, Analysis, and Reporting

Technical Description: AU-6 requires the regular review, analysis, and reporting of audit records to ensure that actions within the system can be traced and verified.

Many organizations adopt a reactive approach, conducting manual ad-hoc audit reviews. This can be made proactive to better identify and mitigate risks. Automated CI/CD tools can capture and review audit logs regularly and flag anomalies. Moreover, a final audit validation can be implemented to cross-reference deployment details with tickets or other auditable records, ensuring that only verified changes reach the production environment. This control also aligns well with PCI DSS, which requires regular audit reviews to maintain a secure environment.

The absence of robust Audit and Accountability controls in Continuous Delivery and deployments can lead to significant risks, including unauthorized changes and the inability to trace actions effectively. However, carefully implementing controls like AU-2 and AU-6 can significantly mitigate these risks and align with other regulatory frameworks.

NIST 800-53 System Integrity (SI)

Overview

System Integrity is a broad family of NIST controls. System integrity is increasingly important during the post-build and pre-production stages of software delivery, where the software artifacts, configurations, and dependencies are in their final forms, awaiting deployment into live environments.

System integrity controls for software release are essential for several reasons:

1. **Verification of Artifacts:** As artifacts move from the build stage to the deployment stage, they must remain unaltered and free from unauthorized changes. This is essential for ensuring that what gets deployed is precisely developed and tested.
2. **Configuration Integrity:** During these phases, configuration data is particularly susceptible to unauthorized modification, either accidental or intentional, which could result in the introduction of vulnerabilities into the production environment.

3. **Dependency Integrity:** Any third-party libraries or services that the software relies on must also be verified to ensure they have not been compromised, as this can indirectly introduce vulnerabilities into your system.
4. **Code Authenticity:** Ensuring that the code deployed is the original code that has passed all security checks is vital. Any deviation can be a potential security risk, affecting the application and integrated systems.
5. **Operational Consistency:** Lack of integrity can cause operational issues, such as system crashes or unexpected behavior, which can be as damaging in the long term as any security breach.

By diligently implementing System Integrity controls into your CI/CD pipeline, your organization proactively safeguards against introducing security risks through the software delivery process.

The CI/CD process can be used to implement key controls within this family:

- SI-2: Flaw Remediation
- SI-7: Software, Firmware, and Information Integrity

Control-by-Control Analysis

SI-2: Flaw Remediation

Technical Description: SI-2 requires the identification and remediation of flaws in software and configurations to prevent vulnerabilities that could be exploited.

SI-2 aims to identify and fix system flaws, such as software bugs or misconfigurations. Without this control, flawed configurations like open ports could be deployed, making the system vulnerable. This control can be enforced in the CI/CD process, for example, by integrating tools that scan Infrastructure as Code (IaC) configurations. These tools block flawed configurations from being deployed, thereby enhancing security. This control aligns well with regulatory frameworks like CIS benchmarks and HIPAA, emphasizing flaw identification and remediation.

SI-7: Software, Firmware, and Information Integrity

Technical Description: SI-7 mandates the verification of the integrity of software, firmware, and information through cryptographic means like hashing or signature checks.

Lack of this control could result in a high risk of deploying tampered code or artifacts. To mitigate this risk, hash or signature checks can be incorporated into the CI/CD pipeline to verify all artifacts before deployment. This reduces the risk of deploying malicious or altered code, enhancing security and trust. This control also aligns with frameworks like ISO 27001 and PCI DSS, which emphasize data and software integrity.

Ignoring System Integrity controls in Continuous Delivery and deployments leaves the system vulnerable to potential security incidents. Implementing controls like SI-7 and SI-2 provides a robust defense while enabling cross-compliance with other regulatory frameworks.

NIST 800-53 Configuration Management (CM)

Overview

The environment in which a software release is deployed is as critical to software delivery security as the software release itself. The Configuration Management family of NIST controls seeks to ensure the integrity of the deployment destination. As organizations move to GitOps, with configurations stored and managed through Git, changes to configuration look much like software releases and thus can be managed through the CI/CD process.

Configuration Management controls are crucial for several key reasons:

1. **Baseline Consistency:** Maintaining a stable baseline configuration is vital as you transition from the build stage to deployment. A well-defined baseline serves as a reference point, ensuring all system components are congruent and secure, mitigating the risks associated with inconsistencies and unauthorized changes.
2. **Change Control:** An effective Configuration Management strategy includes rigorous change control mechanisms. This ensures that configuration alterations are well-documented, reversible, and only implemented following a strict approval process. This level of control is crucial for preventing unauthorized or inadvertent changes that could introduce vulnerabilities or operational inconsistencies.
3. **Access Restrictions:** Determining who can make configuration changes is critical. By strictly controlling access permissions, you minimize the risk of unauthorized or erroneous changes, enhancing the overall security posture using your CI/CD pipeline.
4. **Auditability:** A robust Configuration Management setup ensures that changes are not just controlled but also logged in a manner that supports future audits or forensic activities. This is especially important for complying with various regulatory frameworks that require detailed change logs.
5. **Cross-Compliance:** By incorporating Configuration Management controls, you align your CI/CD processes with compliance frameworks such as ISO 27001, PCI DSS, and HIPAA. This not only improves your security posture but also simplifies the compliance process by addressing multiple requirements through a single control family.

By implementing Configuration Management controls into your CI/CD pipeline, you fortify the stability and security of your software delivery and deployment process.

The CI/CD process can be used to implement key controls within this family

- CM-2: Baseline Configuration
- CM-3: Configuration Change Control
- CM-5: Access Restrictions for Configuration Change

Control-by-Control Analysis

CM-2: Baseline Configuration

Technical Description: CM-2 mandates the maintenance of a baseline configuration as a point of reference for all system components.

The absence of a baseline configuration in Continuous Delivery can lead to inconsistencies across various environments, complicating the tracking and resolution of issues. Within a CI/CD pipeline, this control can be effectively implemented by maintaining an inventory of all deployed software components. This inventory is the baseline configuration and significantly enhances system reliability while simplifying troubleshooting. Adherence to this control also aligns well with other compliance frameworks like ISO 27001.

CM-3: Configuration Change Control

Technical Description: CM-3 focuses on managing changes to system configurations.

Many organizations currently lack strict controls over changes to Infrastructure as Code (IaC) and software configurations. To enforce this control in a CI/CD context, all changes to IaC and software configurations can be managed through controlled pipelines that include automated checks and approvals. This additional layer of scrutiny helps to prevent unauthorized changes and mitigates risks associated with human error. Moreover, this control is essential for compliance with frameworks like PCI DSS.

CM-5: Access Restrictions for Configuration Change

Technical Description: CM-5 aims to restrict the range of authorized personnel to change system configurations.

Without these restrictions, there's a high risk of unauthorized personnel making changes, either inadvertently or maliciously. Within a CI/CD pipeline, this control can be implemented by scanning IaC files for unauthorized changes and either flagging them for review or automatically reverting them. This proactive approach minimizes the risk of unauthorized changes infiltrating the production environment. Implementation of this control also supports compliance with other regulations like HIPAA.

NIST 800-53 Identity and Authentication (IA)

Overview

The Identity and Authentication family of NIST controls is the linchpin for ensuring authenticated and authorized interactions within the software delivery process. Identity and Authentication controls become increasingly relevant during the transitional phases of post-build and pre-production when software components are finalized and ready for deployment.

Enforcing Identity and Authentication through the CI/CD process:

1. **User Verification:** As the software advances toward the deployment stage, only verified users must be able to interact with or make changes to the system.
2. **Device and Process Authentication:** Beyond users, it is also vital to authenticate the devices and processes that are part of the CI/CD pipeline to prevent unauthorized or rogue elements from affecting the system.
3. **Credential Security:** Managing credentials securely helps prevent unauthorized access and protects against both internal and external threats.
4. **Auditability:** Like other control families, Identity and Authentication controls should maintain comprehensive logs to support future audits and forensic activities.
5. **Compliance Synergy:** Implementing these controls aligns your CI/CD operations with other compliance frameworks like GDPR, PCI DSS, and HIPAA, enhancing both security and simplifying the compliance journey.

The CI/CD process can be used to implement key controls within this family:

- IA-5: Unencrypted Credentials and Hardcoded Secrets

Control-by-Control Analysis

IA-5: Unencrypted Credentials and Hardcoded Secrets

Technical Description: IA-5 focuses on the secure management of authentication data, such as passwords and cryptographic keys.

Unencrypted credentials and hardcoded secrets often present vulnerabilities in artifacts passing through CI/CD pipelines. This control can be implemented by integrating tools that automatically scan for and flag such insecure practices. By addressing this, you significantly mitigate the risk of unauthorized system access, fortifying the security posture of the entire pipeline.

Implementing IA-5 is also required to comply with data protection regulations like GDPR and PCI DSS. It ensures that you are taking a comprehensive approach to secure authentication, which is a cornerstone of many compliance frameworks.

NIST 800-53 System and Communications Protection (SC)

Overview

The System and Communications Protection family of NIST controls serves as the defensive shield that safeguards the transmission and interaction of data within and between systems. As the last stop before production deployment, the CI/CD process is a final opportunity to verify system and communications protection.

The CI/CD process can support System and Communications Protection in a few significant ways:

1. **Boundary Integrity:** Robust boundary protections between different operational environments (Dev, Test, Staging, Production, etc.) are key to preventing unauthorized or unintended data flows.
2. **Secure Protocols:** Employing secure communication protocols ensures the integrity and confidentiality of data during transmission, mitigating risks of interception or alteration.
3. **Auditability:** Maintaining detailed logs of data transmissions and boundary traversals is necessary for future audits and complying with various regulations.

The CI/CD process can be used to implement key controls within this family:

- SC-7: Boundary Protection
- SC-8: Secure Communication Protocols

Control-by-Control Analysis

SC-7: Boundary Protection

Technical Description: SC-7 requires enforcement of stringent boundary protections between different operational environments.

Weak or non-existent boundary controls can lead to unintended or unauthorized data flow between different operational environments, including dev, staging, and production. In a CI/CD context, SC-7 can be implemented by controlling which software releases are allowed into the environment and validating Infrastructure as Code (IaC) files to strictly govern transitions between these environments. This ensures that only authorized data flows are permitted.

This control aligns closely with compliance frameworks like ISO 27001, which places a strong emphasis on the integrity and control of data boundaries.

SC-8: Secure Communication Protocols

Technical Description: SC-8 emphasizes the use of secure communication protocols to maintain the confidentiality and integrity of data during transmission.

Implementing SC-8 during the CI/CD process involves scanning configuration files for secure protocols like TLS. Any instances of insecure protocols can be flagged for immediate remediation.

Implementing this control is also required to comply with regulations like HIPAA, which mandates secure data transmission.

NIST 800-53 System and Services Acquisition (SA)

Overview

The System and Services Acquisition family of NIST controls operates as the quality assurance mechanism over systems and services integrated into the software delivery process. Within the complex software supply chain, the CI/CD process is where external systems and services, such as third-party libraries or cloud services, are integrated into the final deployment package.

Failure to apply these controls over these integrations can introduce vulnerabilities and risk non-compliance, jeopardizing the entire operation.

Implementing System and Services Acquisition in CI/CD can provide:

1. **Artifact Provenance:** Verifying the origin of software artifacts is crucial to ensure that only legitimate and secure components are included in the final deployment.
2. **Secure Engineering:** Implementing security engineering principles throughout the software lifecycle helps to identify and remediate vulnerabilities before they can be exploited.
3. **Risk Mitigation:** Effective implementation of these controls minimizes the risk of integrating compromised or insecure systems and services into the CI/CD pipeline.
4. **Cross-Compliance:** By embedding these controls in the CI/CD process, you align with regulatory frameworks such as ISO 27001, PCI DSS, and CIS benchmarks.

The CI/CD process can be used to implement key controls within this family:

- SA-22: Provenance Checks
- SA-8: Security Engineering Principles

Control-by-Control Analysis

SA-22: Provenance Checks

Technical Description: SA-22 focuses on verifying the origin of software artifacts, ensuring their integrity before inclusion in the system.

Without adequate provenance checks, the risk of deploying compromised or altered software artifacts is high. In a CI/CD setting, SA-22 can be implemented by verifying the cryptographic hashes or signatures of all software artifacts before deployment. This ensures they match the signatures generated at build to ensure the artifacts' integrity and legitimacy. CI/CD systems can also validate that builds were performed on authorized build servers, or that 3rd party artifacts came from a trusted source.

This control is harmonious with frameworks like ISO 27001 and PCI DSS, both of which mandate checks on the provenance of software components.

SA-8: Security Engineering Principles

Technical Description: SA-8 requires the application of security engineering principles throughout the system’s lifecycle.

Often, security is not fully integrated into the system acquisition process, leading to vulnerabilities. In a CI/CD context, SA-8 can be implemented by mandating that all code and configurations pass through security scans and tests before deployment. This multi-layered approach ensures that only secure, vetted components are deployed.

Adherence to this control is critical for compliance with various frameworks, including CIS benchmarks, emphasizing the importance of applying security principles throughout the system's lifecycle.

NIST 800-53 Assessment and Authorization (CA)

Overview

The Assessment and Authorization family of NIST controls acts as the governance layer that validates the security compliance of system operations within the software delivery process. With their focus on continuous monitoring and auditing, Assessment and Authorization capabilities can be implemented using CI/CD processes.

Significance of Assessment and Authorization in CI/CD:

1. **Audit Trail:** Robust logging is critical for post-mortem analysis and for demonstrating compliance during audits.
2. **Real-time Monitoring:** Continuous oversight of system activities allows for immediate detection and response to any unauthorized or suspicious actions.
3. **Compliance Validation:** These controls are essential for validating that the software delivery process meets various regulatory requirements.
4. **Cross-Compliance:** Adherence to these controls ensures that the CI/CD pipeline aligns with various compliance frameworks such as PCI DSS and SOX.

The CI/CD process can be used to implement key controls within this family:

- CA-2: Audit Records for Assessment
- CA-4: Continuous Monitoring

Control-by-Control Analysis

CA-2: Audit Records for Assessment

Technical Description: CA-2 stipulates the need for generating and maintaining audit records to facilitate assessment activities.

Mature CI/CD processes and software delivery pipelines document and automate many complex steps in the software delivery and deployment process. In this context, CA-2 can be implemented in CI/CD by maintaining comprehensive logs of all deployment activities. This should include details such as what was deployed, when, and by whom, thereby enhancing the auditability of the system. Lack of such logging can severely hamper post-incident forensic activities and compliance assessments.

This control is particularly beneficial for organizations that need to comply with regulations like the Sarbanes-Oxley Act (SOX), which mandates the need for extensive audit trails.

CA-4: Continuous Monitoring

Technical Description: CA-4 underscores the importance of continuous monitoring mechanisms to detect and report unauthorized or suspicious activities promptly.

Given that “C” in “CI/CD” is for “Continuous,” the CI/CD process can be a pivotal contributor to this control. With some organizations pushing thousands of software releases weekly or daily, the CI/CD system is uniquely placed to give a real-time view of what is running where in the environment. In support of CA-4, you can enable real-time monitoring during the deployment phase, possibly triggering alerts or halting deployments if anomalies are detected.

Implementation of this control is aligned with frameworks like PCI DSS, which also mandates continuous monitoring for unauthorized access or activities.

How to Implement NIST 800-53 Security Control in CI/CD Pipelines

Achieving compliance with NIST security controls requires a comprehensive approach. The accelerating speed and frequency of software delivery today make it challenging to monitor, enforce, and report compliance without disrupting the flow of operations. However, with the right architecture and processes in place, it's possible to use the CI/CD process to achieve agility and compliance seamlessly.

The overarching architecture involves three primary phases: Data Collection, Data Analysis, and Compliance Conformance. In the Data Collection phase, the CI/CD pipeline is integrated with a suite of DevSecOps tools to gather essential data points. This includes, but is not limited to, code changes, configuration adjustments, and system interactions. Control mapping techniques are employed to ensure that the data collected aligns with the specific NIST controls relevant to the organization.

The Data Analysis phase involves processing the collected data, typically aggregated in a centralized data store, to identify compliance gaps or potential security vulnerabilities. Advanced algorithms and real-time monitoring solutions are employed to sift through this granular data.

Lastly, in the Compliance Conformance phase, automated tools such as a deployment firewall act as gating mechanisms. These tools analyze the data against predefined compliance policies or rules and either approve or reject deployments based on this analysis.

In the following sections, we will delve deeper into each phase, providing a roadmap for effectively implementing NIST security controls into your CI/CD pipeline.

Data Collection During Software Delivery

Types of Data to Collect

For secure and compliant software delivery, aligning data collection strategies with NIST security controls is imperative. The first crucial step in this direction is the analysis of your existing CI/CD processes to identify what types of data can be collected to help achieve compliance. This is where the concept of control mapping comes into play.

Control mapping involves correlating specific NIST controls with corresponding data points collected by your CI/CD pipeline. For example, if you are focusing on the NIST control AU-2, which deals with audit events, your data collection strategy should ensure the capture of all event logs that record who has accessed what resources and when. This would encompass logs from code repositories, build systems, deployment orchestration tools, and runtime environments. It's not just about gathering information; it's about gathering the right information.

Importance of Data Granularity

Data granularity is a necessity when dealing with compliance frameworks like NIST. Granular data allows for an in-depth analysis that can highlight anomalies or non-compliant activities at a very detailed level. For instance, in AU-2, fine-grained logs that record each API call or system interaction provide the depth of visibility needed for rigorous auditability. This granularity is also pivotal when implementing protective mechanisms like a deployment firewall, ensuring that each piece of data is evaluated for compliance as it passes through the CI/CD pipeline.

Tools and Techniques for Effective Data Collection

Control mapping helps you identify what data to collect, but how you collect that data is equally important. Your CI/CD solution should offer an integration framework that seamlessly aggregates data from your existing DevSecOps toolchain. Whether it's static code analysis tools, container scanners, or configuration management databases, the integration should be seamless.

Sophisticated data collection techniques such as real-time event streaming, batch data capture, and edge data collection can be employed. These methods, when integrated into your CI/CD pipeline, offer a robust mechanism for capturing all required data types in real-time. By doing so, you not only create a resilient data collection system but also ensure that your deployment firewall has all the data it needs to effectively gate your deployments based on your security and compliance requirements.

Data Analysis and Normalization

How to Process and Analyze the Collected Data Using AI/ML Models

Once data is collected, the next critical step is its analysis and normalization. For speed and scalability, this increasingly can build on available AI/ML models. These models assess the data's security risks and impacts on an ongoing basis. This detailed analysis is the foundation for a deployment firewall, a security layer that scrutinizes all deployment-related data against predefined criteria.

Use Case: Deployment Verification Through AI/ML

For example, AI/ML models can be utilized to analyze deployment logs for abnormal variations and behaviors against a developed baseline. A "security score" can be generated based on this analysis. This score can then be used by the deployment firewall to either permit or reject a deployment to production. If the score falls below a certain threshold, the firewall can automatically alert, audit, or block the deployment, ensuring that only compliant and secure code makes it to the production environment.

Normalization and AI/ML-Driven Decision-Making

Normalization of data involves converting disparate data into a standard format or unit that can be easily compared. AI/ML algorithms can significantly help in this normalization process, making it easier to correlate data from different sources in the DevSecOps toolchain. This contributes to more accurate risk assessments and more effective deployment firewalls.

Compliance Conformance

Importance of Rigorous Conformance Measures

Compliance conformance is a cornerstone for a robust and resilient security architecture. It requires active enforcement, auditing, and alerting of compliance statuses. This multi-faceted approach is made possible through the implementation of a deployment firewall, which serves as a real-time verification and enforcement mechanism. This tool can alert or audit when security and compliance policies are not met, allowing for immediate remedial action.

Conformance Criteria, Benchmarks, and Policies

Establishing a set of well-defined criteria and benchmarks is crucial for the effective operation of your deployment firewall in enforcing compliance conformance. These criteria serve as the policies that the deployment firewall uses to make real-time decisions on whether to allow or deny a deployment. By adhering to these predefined policies, the deployment firewall can enforce rigorous compliance conformance, providing immediate alerts or audits when deviations or violations occur. This adds a real-time, proactive layer of security and compliance assurance to your CI/CD pipeline.

Deployment Firewall Example Policies

- ✓ Containers must be hardened and only use compliant base image
- ✓ Production config files must not contain UAT parameters
- ✓ Only QA can approve automated tests
- ✓ All scans with score below 80% must be manually approved
- ✓ Pipeline updates from authorized users
- ✓ A pipeline must include prescribed security checks
- ✓ An application must not be deployed on weekdays between 9 am-4 pm
- ✓ Only fully approved tickets can be deployed
- ✓ Require manual judgment in pipelines before deploying to production
- ✓ Require “Owner” and “Application” annotations to be defined
- ✓ Restrict what ports can be exposed by load balancers
- ✓ Prevent application pipelines from deploying to certain namespaces

These policies serve as actionable rules that directly influence the operational behavior of your deployment firewall, fortifying the overall security posture of your CI/CD pipeline.

Continuous vs. Event-Triggered Compliance Conformance
Continuous conformance offers the advantage of ongoing compliance monitoring and immediate flagging of deviations, thus enabling swift corrective measures. Event-triggered conformance is activated in response to specific incidents or audit requirements and is equally critical for a comprehensive conformance strategy. The deployment firewall plays a pivotal role in both, ensuring that all deployments meet defined security and compliance benchmarks while providing the flexibility to audit or alert based on different triggers.

By meticulously integrating these practices into your CI/CD pipeline, you're not merely adding another layer of security; you're creating a deployment shield fortified by a robust deployment firewall. This ensures that your software delivery process is in alignment with NIST controls and other regulatory frameworks, thereby enhancing both your security posture and compliance levels.

Conclusion

In summary, the NIST 800-53 framework provides a robust set of guidelines to secure the software supply chain, with particular emphasis on the software delivery and deployment processes. As highlighted in this paper, techniques like Shift Left have limitations in providing end-to-end security across the pipeline. By implementing controls from relevant NIST families like Access Control, Audit and Accountability, and System Integrity, organizations can establish protections that span the entire delivery process.

The CI/CD process that your organization likely has in place today can be a powerful foundation for achieving NIST 800-53 compliance. The control baseline proposed in this paper, encompassing subsets of eight key families, is an excellent starting point for most organizations. With software supply chain attacks on the rise, aligning delivery and deployment practices with NIST 800-53 is a proactive cybersecurity strategy. The framework's comprehensive nature also enables compliance with industry regulations in a streamlined manner.

Overall, for any organization seeking to enhance the security and compliance of their software delivery process, leveraging NIST 800-53 is a foundational best practice within the software supply chain landscape.

OpsMx

Streamlining NIST Compliance with OpsMx

Translating the NIST 800-53 controls into actionable policies and demonstrating compliance can be challenging for most organizations.

OpsMx can dramatically simplify NIST compliance for software delivery. OpsMx offers a turnkey compliance solution through its Deploy Shield and SecureCD products that can help organizations realize the benefits of automated compliance and audit quickly and cost-effectively. Key advantages of the OpsMx approach include:

- Comprehensive policy framework - OpsMx provides 150+ out-of-the-box policy templates spanning security, compliance, infrastructure readiness, and release quality. Policies can be easily customized and extended.
- Automated enforcement - The OpsMx Deployment Firewall continuously and automatically evaluates software releases for compliance with your defined policies, generating alerts or blocking out-of-compliance releases.
- Broad tool integrations - Deploy Shield and SecureCD integrate with all major CI/CD tools like Jenkins, Spinnaker, Argo CD, and commercial cloud platforms. This avoids a rip-and-replace approach.
- Automated compliance audits - Deployment firewall policies map to specific controls in frameworks like NIST, PCI DSS, and HIPAA. Compliance reports are auto-generated.
- Simulate before deploying - Deployment simulations allow catching policy violations early without impacting production environments.
- Unified control plane - All deployment data, approvals, and audit trails are available from a single control plane for simplified management.

By leveraging OpsMx's out-of-the-box policies, automation, and reporting tailored to NIST 800-53, organizations can not only accelerate their compliance journey but also achieve and maintain alignment with various regulatory frameworks. The pre-packaged nature of our solutions offers quick time-to-value, eliminating the complexities and costs associated with custom builds. Reach out to OpsMx to explore how we can support your compliance journey.

Appendix A - Summary of NIST 800-53 Controls Discussed

Control Family	Description	Controls Discussed
Access Control (AC)	Manages and controls access to information and resources	AC-5: Separation of Duties AC-6: Least Privilege AC-8: System Use Notification
Audit and Accountability (AU)	Creates and reviews system audit logs	AU-2: Audit Events AU-6: Audit Review, Analysis, and Reporting
System Integrity (SI)	Maintains system and information integrity	SI-2: Flaw Remediation SI-7: Software, Firmware, and Information Integrity
Configuration Management (CM)	Manages security features through controlled changes	CM-2: Baseline Configuration CM-3: Configuration Change Control CM-5: Access Restrictions for Configuration Change
Identification and Authentication (IA)	Validates users, processes, or devices	IA-5: Unencrypted Credentials and Hardcoded Secrets
System and Communications Protection (SC)	Protects information in transit and at rest	SC-7: Boundary Protection SC-8: Secure Communication Protocols
System and Services Acquisition (SA)	Emphasizes security in system acquisitions	SA-22: Provenance Checks SA-8: Security Engineering Principles
Assessment and Authorization (CA)	Assesses controls and authorizes operations	CA-2: Audit Records for Assessment CA-4: Continuous Monitoring