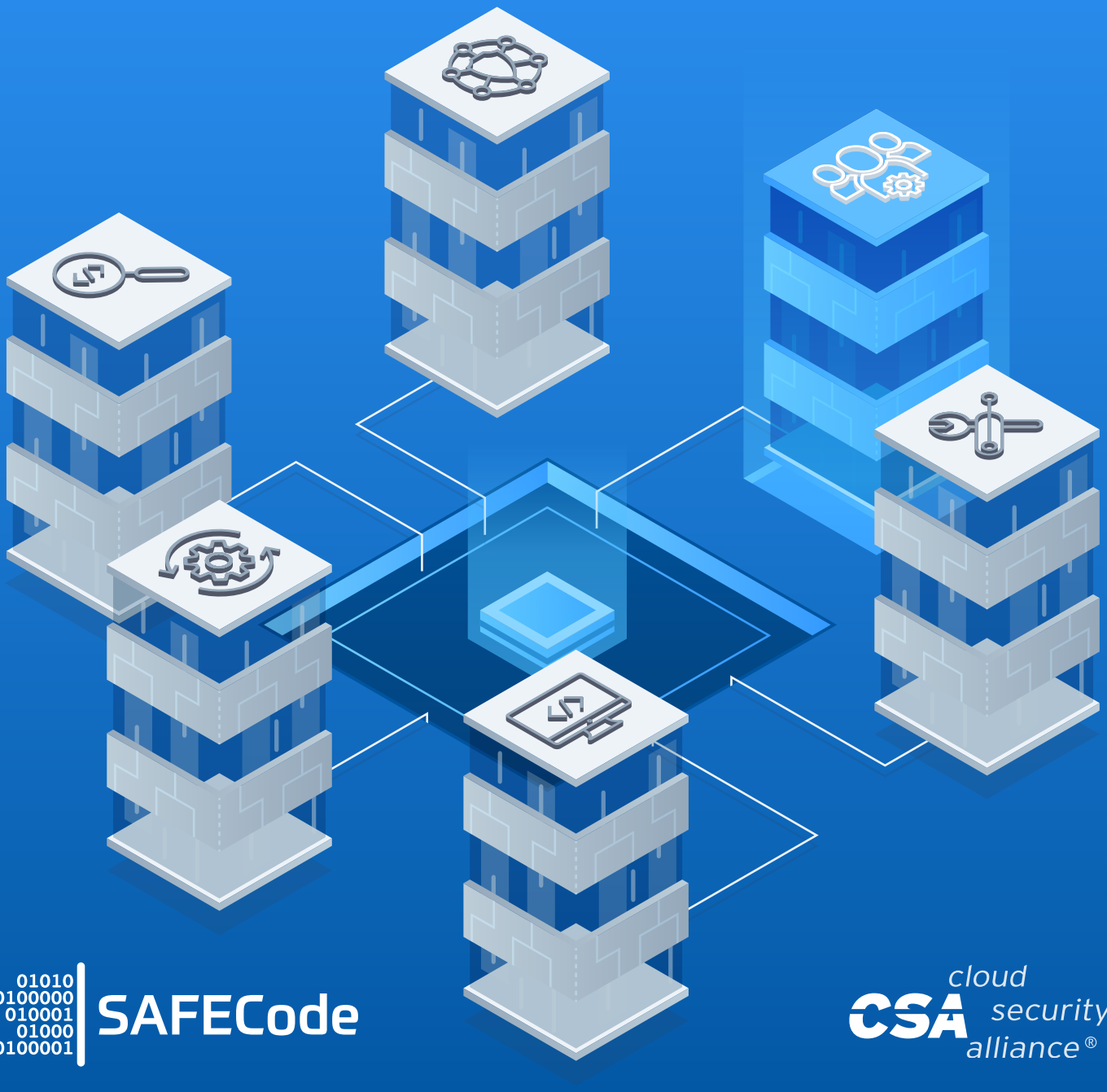
The Six Pillars of DevSecOps:

Collaboration and Integration



SAFECode

CSA cloud security alliance®

The permanent and official location for DevSecOps Working Group is
https://cloudsecurityalliance.org/research/working-groups/devsecops/

Acknowledgments

Lead Authors

Abdul Rahman Sattar

Contributors

Aristide Bouix
Amit Butail
Ivan De Los Santos
Darien Hirotsu
Nitin Kulkarni
Alexandria Leary
Michael Roza

Reviewers

Kapil Bareja
Ram Reddy
Udith Wickramasuriya

CSA Analysts

Josh Buker

Table of Contents

Acknowledgments 3
Foreword 5
Introduction 5
    Goals 6
    Audience 6
Guiding Principles for Successful DevSecOps Collaboration and Integration 7
The Why and How of a Role-Based Security Training Program 9
Collaboration and Integration in the End-to-End DevSecOps Delivery Pipeline 13
Integration Process of a New Acquisition into DevSecOps Delivery Pipeline 16
DevSecOps Case Studies 18
Convergence Between DevSecOps and Other Technology Practices 22
References 29

Foreword

The Cloud Security Alliance and SAFECode are both deeply committed to improving software security outcomes. The paper Six Pillars of DevSecOps, published in August 2019, provides a high-level set of methods and successfully implemented solutions used by its authors to build software at speed and with minimal security-related bugs. Those six pillars are:

Pillar 1: Collective Responsibility (Published 02/20/2020)
Pillar 2: Collaboration and Integration (Published 02/21/2024)
Pillar 3: Pragmatic Implementation (Published 12/14/2022)
Pillar 4: Bridging Compliance and Development (Published 02/08/2022)
Pillar 5: Automation (Published 07/06/2020)
Pillar 6: Measure, Monitor, Report, and Action (Estimated - March, 2024)

The successful solutions that underpin each of these pillars are the subjects of a much more detailed set of joint publications by the Cloud Security Alliance and SAFECode. This paper is the fifth of those follow-on publications.

Introduction

The Cloud Security Alliance DevSecOps working group (WG) issued high-level guidance in *Cloud Security Alliance, Information Security Management through Reflexive Security: Six Pillars in the Integration of Security, Development, and Operations*[1] on a new approach to application security called "Reflexive Security." The six pillars are considered to be the critical focus areas for any organization interested in implementing Reflexive Security or DevSecOps.

One of the pillars is "Pillar 2: Collaboration and Integration'' which can be summarized as "Security can only be achieved through collaboration, not confrontation." Security is a team sport that requires collaboration across the board between various organizational roles including business leaders, domain experts, the security personnel, architects, software developers, pentesters, SOC analysts, and product/project managers. Collaboration between key stakeholders and various organizational roles is required to ensure that the threat landscape relevant to the business sector is well understood, and that the organizational practices for IT activities including the software development lifecycle follow proper security hygiene. Various stakeholders also need to collaborate to make sure the organization supports continuous role-based security training. The security champions have to work with other teams in the organization to ensure that the security practises are well-documented and often communicated across the organization. The leadership and the security teams need to collaborate to ensure business continuity risks factor in relevant cyber risk and that the business has appropriate cyber incident response strategies in place.

1 https://cloudsecurityalliance.org/artifacts/information-security-management-through-reflexive-security/ page 7-9

Goals

This paper's primary focus is highlighting the criticality of integrating DevSecOps into organizational processes and fostering collaboration required for a successful DevSecOps implementation. The paper starts with a list of Guiding Principles for DevSecOps Communication. It then delves into the details of implementing a continuous role-based security training program at an organization. The paper then walks through how various organizational roles collaborate in an end-to-end DevSecOps delivery pipeline. This is followed by a section on the communication and collaboration required amongst various organizational roles to integrate a new acquisition into the existing DevSecOps processes at the organization. The paper concludes with a section on convergence between DevSecOps and other technology areas like Zero Trust, AIOps, and MLSecOps which gives an overview of how DevSecOps can be leveraged for Zero Trust, various issues in MLSecOps that have some semblance with DevSecOps, and how AIOps could be leveraged by DevSecOps.

Audience

The target audience of this document includes those who are involved in the management and operational functions of risk, information security, information technology, and knowledge management. This includes the CISO, CIO, COO, and the individuals involved in the following functional areas: Security engineering, product management, Solution and Application Architects, automation, DevOps, quality assurance, InfoSec, App Security, governance, risk management, and compliance, human resources, and training.

Guiding Principles for Successful DevSecOps Collaboration and Integration

DevSecOps is a methodology that integrates security into the DevOps process. This ensures software is secure, reliable, and efficient from the beginning of the development process. "Shift Left" is a term that is frequently used to describe this integration.

To successfully implement DevSecOps, it is important to establish a culture of collaboration, communication, and continuous improvement. Leadership, product, project, developers, security professionals, and operations teams work together seamlessly to ensure business continuity, IT security, and to create and deploy secure software. Below are some tips for crafting an effective cross-team collaboration and integration culture for DevSecOps.

Initiative and Effective Communication from Leadership

To be effective, a culture shift must come from the top down. Strong leadership buy-in will facilitate the collaboration required to implement a DevSecOps culture. Leadership is ultimately in charge of setting and communicating business continuity goals to the rest of the organization. This in turn drives funding and effort in the organization to help realize these goals set by the leadership. Planning to ensure business continuity in the face of a cyber incident is key and implementing security-first architecture requires collaboration between multiple teams to understand various internal and external cyber threats that a business faces, and the impact of these threats on the business. Once threat models are well understood, teams need to strategize on how to identify and protect organizational assets, continuously monitor and detect cyberthreats, and respond and recover business assets in case of a cyber breach. Integrating security throughout the entire IT lifecycle using DevSecOps principles is key to protect and detection functions.

No Silos

The purpose of DevSecOps is to integrate security as a shared responsibility throughout the entire IT lifecycle. This requires full transparency between all areas of the business, a willingness to embed security from the very beginning of the development process, and a deep belief that security, like quality, is not the sole duty of one team or individual but is shared across all parts of the organization. Leadership should work to ensure there is a constant security feedback loop in place and encourage open and frequent communication between teams with an effort to fully integrate security into the business processes. Engineering, security, and operations should work hand in hand and jointly own the outcomes of the DevSecOps process. One of the ways to see the biggest return on investment is to automate security to the largest extent possible. A partnership between the engineering team and the security team is a great way to ensure those automations are working and are put in place correctly.

Automation

Leveraging tools for automating security-first design in the Software Development Lifecycle (SDLC)2 can save time and provide security feedback in real-time. Automation can be leveraged for various security-related activities such as: asset identification, continuous security monitoring, code analysis, vulnerability scanning, penetration testing, incident response and recovery, and streamlining communications during a cyber breach.

A People-Focused Approach

Since the DevSecOps culture can only succeed with the full participation of many different teams, it's imperative to focus on the people aspect. A culture of continued learning is the best way to ensure your teams can stay on the cutting edge of DevSecOps technologies and processes. Measurements of success need to be created, and achievements need to be celebrated! This should be implemented for both the people and learning aspects as well as technology. Learning goals for the team can include training and certifications, which should be included in performance plans and employee reviews. Remember that you will need to provide time and incentive for these learning programs to take hold. Technological milestones should also be celebrated to keep the team motivated and ensure the DevSecOps program is on track. A few examples of technical success measurements include lead time, deployment frequency, availability, and frequency of security exploits or attacks.

Understanding of the Organizational Context

Organizations across industries are unique, each with its distinctive culture, structure, and operational methodologies. As such, the adoption of the DevSecOps approach needs to be thoughtfully tailored to each organization's specific context. This involves not just the adoption of suitable tools, but more importantly, fostering a culture that aligns with DevSecOps principles. It's quite common for conversations around DevSecOps transformation to become overly focused on tool selection and implementation. However, for organizations to reap the true benefits of DevSecOps, the emphasis should be on its foundational principles.

Clear Ownership and Accountability

For a successful DevSecOps program, there needs to be a clear stakeholder RACI. For each task, there needs to be clear stakeholder alignment on who is accountable for the task, which teams are responsible for completing the task, and which stakeholders need to be consulted and informed of task progress and completion. For example, Software engineers are responsible for the disposition of risk found at the application layer of their application or product. Infrastructure engineers are responsible for the disposition of risk found at the operating system level, and so on.

2 See also: Pillar 3: Pragmatic Implementation and Pillar 5: Automation.

Agreement on how to measure progress

The organization needs to agree on how it will measure the management of quality, performance, availability, security risk, and control effectiveness. This will help align all stakeholders to arrive at the same goal which would complement business goals and objectives.

Agreement on the methodology that will be used to prioritize work
The organization needs to agree on how risk is evaluated and prioritized. Disagreements on how risk is prioritized will adversely impact how the organization collaborates.

Embracing failures as opportunities

Organizations must embrace the inevitable fact that failures can and will occur. Rather than viewing these setbacks negatively, they should seize them as valuable opportunities for growth and enhancement.

Adopting these principles will create the right environment for a successful DevSecOps program that helps mature the secure software development practices of the organization while delivering value for their customers.

The Why and How of a Role-Based Security Training Program

Security is a team sport. Each group of players needs to be trained in accordance with the role they play. The world is a complex and diverse place and the people within it are a melting pot of abilities, skills, and attitudes. Within a company, this diversity is often put to good use by creating specific roles that leverage these different abilities and skills.

DevSecOps encompasses a multi-dimensional approach that applies throughout multiple layers in an organization and its product offerings. Given that individuals have varied roles within an organization, it's imperative to continuously offer training that's tailored to align with their specific duties and responsibilities.

Participant	Role Definition
Business Stakeholders	
C-Level Executives	• Informed about DevSecOps operations, top-level Key Performance Indicators (Decrease in Mean Time To Response and Remediate MTTR ...[3]), and Key Risk Indicators. • Allocate resources and budget for Role-Based Security training.

3 For full list please refer to Pillar 6: Measure, Monitor, Report and Action (estimated release - 3/24)

Investors, Founders, and other Business Owners	• Informed about DevSecOps operations, top-level Key Performance Indicators (Decrease in Mean Time To Response and Remediate MTTR ...[4]), and Key Risk Indicators.
Engineering Managers, Product Owners, Product Managers	• Consulted to implement DevSecOps approach in their way of work and product lifecycle.
Security Stakeholders	
CISO/CSO or CIO	• Accountable for the DevSecOps program across the organization. • Responsible for Defining DevSecOps program objectives. • Aligning the DevSecOps program with business goals and objectives.
VPs of Security	• Accountable for the DevSecOps program in their corresponding department. • Responsible for defining DevSecOps implementation functional requirements.
Security Team Leads	• Responsible for monitoring the DevSecOps process and escalating potential issues. • Responsible for defining DevSecOps implementation functional requirements.
Security Operations (PenTesters, IR Team, Threat Intel)	• Responsible for incorporating DevSecOps inputs into their operations. • Consulted by DevSecOps implementers.
DevOps/ Platform Stakeholders	
VP/Director of Engineering	• Accountable for DevSecOps processes implementation. • Consulted for defining DevSecOps implementation functional requirements. • Consulted for DevSecOps implementation across departments. • Responsible for measuring the impact of DevSecOps implementation impact on departments' productivity. • Consulted to evaluate DevSecOps implementation impact on Departments' productivity.
Platform Architects and Owners	• Accountable for the implementation of DevSecOps processes in their corresponding applications. • Consulted for DevSecOps implementation across their corresponding applications. • Responsible for measuring DevSecOps implementation impact on their corresponding applications.

4 For full list please refer to Pillar 6: Measure, Monitor, Report and Action (estimated release - 3/24)

DevOps Engineers and Team Leads	• Responsible for implementing and maintaining DevSecOps processes in development and release (CI/CD) workflows.
Software Development Engineers and Team Leads	• Responsible for following DevSecOps processes. • Consulted for DevSecOps process implementation and continuous enhancement.
Scrum Masters and Project Managers	• Responsible for informing and promoting DevSecOps processes during development workflows. • Consulted for DevSecOps process implementation and continuous enhancement.
Quality Assurance Stakeholders	
VP/Director of Quality Assurance	• Consulted for defining DevSecOps implementation functional requirements.
Quality Assurance Manager	• Accountable for DevSecOps implementation into QA processes • Consulted to evaluate DevSecOps implementation impact on applications' performances.
Quality Assurance and Team Lead	• Responsible for DevSecOps implementation into QA processes • Consulted for DevSecOps process implementation and continuous enhancement.

Security Training Program Implementation

When rolling out a security program as part of a roadmap, it's crucial to remember that many individuals may not have ample time to focus solely on security. Hence, defining the goals of your program early on is vital. For instance, the program could aim for Product owners to incorporate Threat Modeling during the discovery phase of new product features.

Once these objectives are in place, it's essential to establish a baseline training for the organization. This will acquaint members with the tools and processes introduced by the Security and Platform teams. Regular email communications or an internal newsletter about the latest security trends affecting the company can be dispatched to keep the organization updated.

Anticipate that certain team members will seek deeper insights and proactively ask questions or suggest improvements on security topics. It's crucial to be receptive and view each inquiry as if it's coming from a "security customer." Embracing DevSecOps means empowering developers to release their work securely rather than hindering them.

For those security enthusiasts or "champions" who wish to delve deeper, consider forming a dedicated working group. Use company collaboration tools to facilitate frequent discussions on security-related topics and schedule regular internal workshops or meetups at least quarterly.

How to Secure Buy-in and Support for a Security Training Program

Securing buy-in from diverse stakeholders can be challenging, especially when very few employees have dedicated time for security issues within their organization. Nonetheless, this shouldn't translate to solo efforts. It's imperative to highlight to leadership the necessity of routinely addressing security issues alongside other technical debts to ward off significant security incidents. Impacts a cyber breach could have on the organization in terms of data loss, customer churn, business downtime etc. should be highlighted to the leadership. To secure leadership buy-in for a security training program, it's important to show the leadership the impacts a security training program would have in reducing cyber incidents in an organization through a cyber aware workforce, robust devsecops program, and better preparedness of an organization to respond to a cyber breach.

Another challenge lies in maintaining employee engagement in the Security Training Program. A proven approach is to host periodic security meetups and to set aside 30 minutes each month for 1:1 interactions with each security champion. These moments offer opportunities to publicly acknowledge security champions' contributions and provide support in other aspects of their roles, such as involving them in key security decisions, like selecting and implementing a new tool, process, or policy.

How to measure the success of a Security Training Program

One straightforward way to assess the success of your training program is by tracking the number of participants who complete the security training, as well as measuring interactions with newsletters. In addition, you could do live and offline security quizzes, tabletop exercises, and simulated training sessions where you could assess their knowledge as well as how participants would act or think in such situations.

After establishing a security champion working group and cultivating a DevSecOps collaborative culture, you can introduce additional metrics such as:

- Growth in the number of Security Champions
- Engagement in workshops
- Number of security issues reported spontaneously
- Number of security concerns addressed within defined internal SLAs
- Changes in the frequency of security incidents
- Average time taken for remediation
- Number of security issues going up or down
- Mean time to detect security issues
- Mean time to remediate security issues

Collaboration and Integration in the End-to-End DevSecOps Delivery Pipeline

The figure below enumerates the key stages that make up the end-to-end Software Development Lifecycle (SDLC). Each of these stages require communication and collaboration between various stakeholders to help realize the objectives of that stage. This section iterates through each of the specified stages and, for each stage, identifies the key stakeholders that participate in that stage and the essential communication and collaboration required between those stakeholders to ensure successful execution.

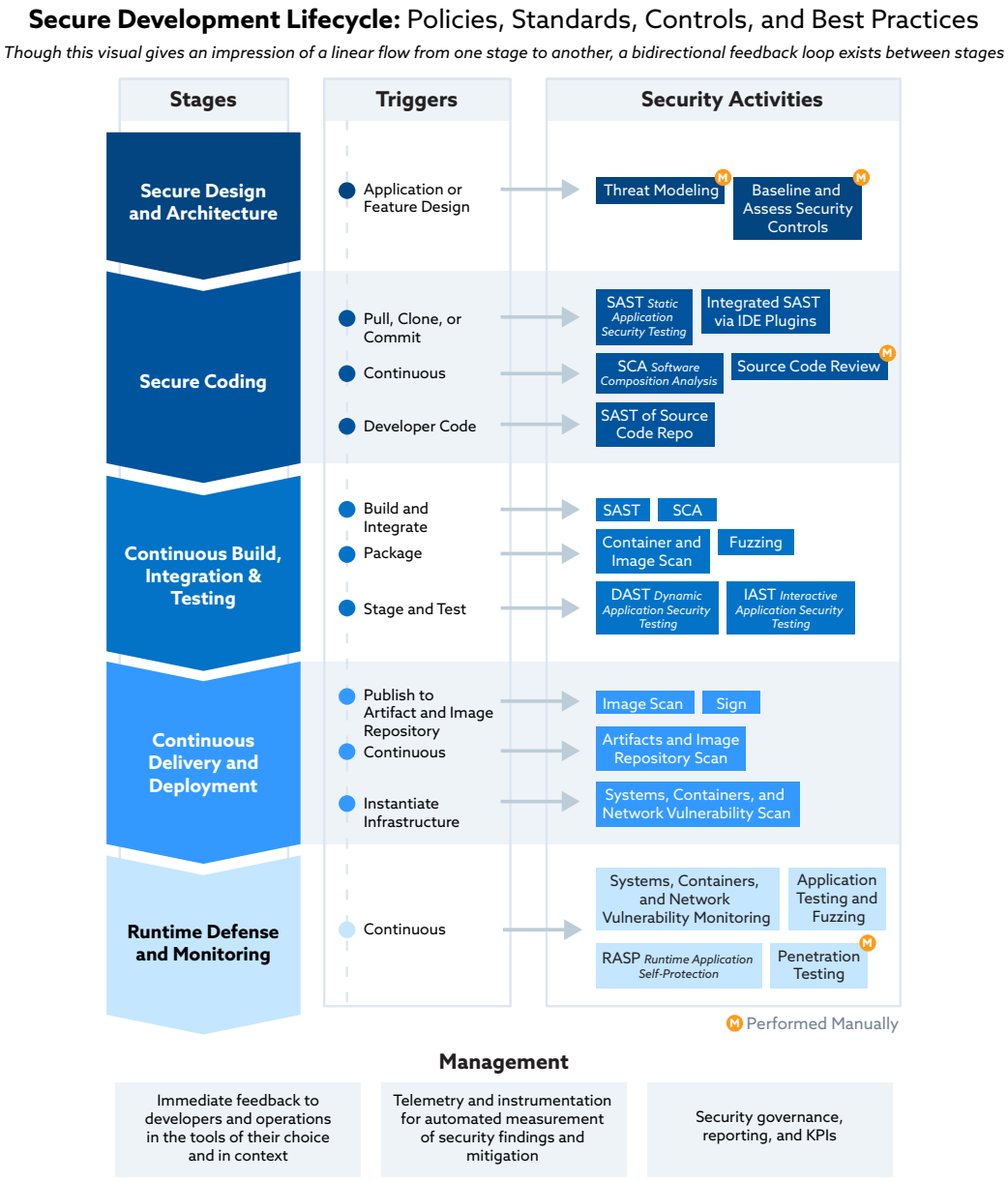


Figure 1: The CSA DevSecOps Delivery Pipeline

Secure Design and Architecture Stage

This stage is a cross-team collaboration between product teams and architects, which include system and security architects, developers, and project teams. The architects and developers team up to take Product Vision and a Product Requirements Document (PRD) developed by the product team as input and produce an aspirational System High-Level Design (HLD). The steps to get to the aspirational System HLD should also include planned work for pentesting, red teaming, blue teaming, and purple teaming exercises for the system. Architects and developers collaborate in this stage to evaluate various design choices, and come up with an aspirational design to meet various functional and non-functional requirements. The security architects develop threat models for the system to understand threat vectors, attack surfaces, security risks, and blast radius for each design choice. The security architects also evaluate and consider reusing existing security design patterns successfully used by other teams in the organization.

Secure Coding Stage

In this stage, project managers and developers collaborate to ensure the project contains the high-level milestones needed to realize the aspirational architecture from the Secure Design and Architecture stage. The development team uses agile development to break these milestones into epics, user stories, and tasks. Daily project standups and project retrospectives ensure the development is on track and moving forward, and team members are aligned. During code development, developers use secure coding practices, security standards such as OWASP, organizational coding standards, and code review processes to collaborate and develop secure code. System HLD from the earlier stage is used to develop code for authentication, authorization, auditing, and integration with build systems, chaos testing, security testing, and monitoring tools.

Continuous Build, Integration, and Testing

In this stage, developers, security testers, quality assurance testers, operations, and architecture teams collaborate to develop test automation and integrate that into the system CI pipeline so that the system can be continuously tested for performance, quality, availability, usability, and security as part of the build cycle. Development teams use feedback and test results from code coverage analysis, QA testing, SAST, DAST, IAST, container image scans, chaos testing, scalability, and stress testing, and pentesting to address gaps in code and design. Any gaps identified in the system design as a result of these various testing exercises will require developers and architects to collaborate on design adjustments and modifications in the System HLD. Agile is used to keep product and project teams in the loop on any additional design changes and bug fixes required.

Continuous Delivery and Deployment

In this stage, developers and operations collaborate to develop a Continuous Deployment (CD) pipeline for the project and integrate it into the CD tools that the organization uses. The team works together to set up continuous monitoring and monitoring dashboards for the system to track key performance indicators (KPIs) and alarms. The team also collaborates to develop project incident response playbooks to ensure the project is integrated with an incident response platform, and develop an on-call schedule for Day 2 support.

Runtime Defense and Monitoring

This is the stage where developers and operations collaborate to ensure the system runs smoothly in production on Day 2. The operations and development teams keep on top of project KPI dashboards and security events and work together to resolve system alarms and incidents. The Operations team is usually the first line of defense when an incident happens. In cases where it cannot resolve the issue, the incident is escalated to the Development team. Post-incident, there is usually an incident postmortem process where the architecture, development, and operations teams come together to brief senior leadership on root cause analysis (RCA) of the incident, the details of the incident, and whether the incident could have been avoided. This postmortem process then provides feedback on whether design and coding gaps in the system need to be fixed.

Integration Process of a New Acquisition into DevSecOps Delivery Pipeline

When a new company is acquired, the security team needs to understand its software development processes. This is important for two reasons: to measure the maturity of its processes and to learn how to best integrate them into the existing processes. The new owner can take the following steps.

Within 60 days of the acquisition

1. Compile a list of all the Source Code Management platforms (Bitbucket, GitHub, GitLab, Subversion, etc.). Include the names of the administrators for these systems.
2. List all the active code repository URLs. Align each application and other artifacts supported by each code repository. Include the product manager and engineering leads, as applicable for each application and product.
3. List the programming languages used at the acquired organization and their proportions. This could help identify gaps in controls (scanning, code reviews, etc.).
4. Identify what tools/processes are used, if any, for:
    - Secrets scanning
    - Static Code analysis
    - Dynamic analysis
    - Container/Image scanning
    - Infrastructure as Code scanning
    - Vulnerability management
    - Threat modeling
    - Configuration risk/Baseline Scanning
    - Software-as-a-Service risk management
    - Identity and Access Management governance
    - Change Management
    - Incident Management
    - Risk Management
    - Ticketing Platform integrations

The type of security tools used in the organization will give a good understanding of their current DevSecOps maturity level.

5. List all the cloud accounts (AWS, Azure, GCP, etc.) and on-premise environments with corresponding resource owners to share with the new owning organization.
6. Put in place the instrumentation required to send all security scan results and all available logs to the acquirer's organization security operations center (SOC) to be added as new data sources.
7. Discuss with executives, product managers, and software engineers from the acquired entity, how security is treated in the organization from a work prioritization standpoint.

Are time, money, and personnel allocated each sprint to meet security and architectural modernization requirements?

Based on the above information, identify gaps with the acquiring entity's processes and prioritization.

Note: At this stage, no changes should be made to the existing software development processes of the acquired company, unless the risk of this existing process exceeds the risk appetite of the new acquirer company.

Within 180 days of the acquisition

1. Use the DevSecOps Maturity Model to measure the acquired organization's across all dimensions. It should be the same model used by the acquiring organization. Popular DevSecOps Maturity Models are:
   - OWASP DevSecOps Maturity Model
   - Cloud Native Computing Foundation Maturity Model
   - Software Assurance Maturity Model
2. Use the results of the maturity model to develop a plan to incorporate and mature the acquired organization's DevSecOps program.

At least once a year

1. Evaluate and continue to mature the DevSecOps program across the organization.

   Each team will be at a different maturity level and will have a unique path to improve its DevSecOps practices. Changing the existing software development processes should be explored on a team-by-team basis.
2. Measure and report DevSecOps maturity levels at various levels (team, product, division, etc.)

DevSecOps Case Studies

This section gives a quick overview of some real-life examples of organizations, of varying sizes and at different phases of the business lifecycle, which implemented DevSecOps successfully as part of their digital transformation journey and the key facets that underpinned their DevSecOps implementation.

Large Established Corporation with Legacy Components

Capital One embarked on a DevSecOps transformation journey to enhance security, streamline processes, and accelerate software delivery. Here's an overview of how they implemented DevSecOps.

Cultural Transformation: Capital One focused on building a culture of collaboration and shared responsibility among development, operations, and security teams. They fostered open communication channels and encouraged cross-functional collaboration to break down silos and promote a collective ownership mindset.

Automated Security Controls: Capital One automated security controls across their development pipelines. They integrated security scanning tools, such as static application security testing (SAST) and dynamic application security testing (DAST), into their continuous integration and deployment processes. This helped identify and address security vulnerabilities at an early stage.

Continuous Compliance: Capital One emphasized continuous compliance by automating compliance checks and incorporating them into their CI/CD pipelines. They utilized configuration management tools and policy enforcement mechanisms to ensure adherence to industry regulations and internal security standards.

Threat Modeling and Risk Assessment: Capital One performed comprehensive threat modeling and risk assessments to identify potential security risks in their legacy components. They analyzed the attack surface, identified vulnerabilities, and implemented appropriate security measures to mitigate risks effectively.

Legacy Component Modernization: Capital One gradually modernized their legacy components by breaking them down into smaller, more manageable units. They adopted microservices architecture and containerization to improve scalability, maintainability, and security. This allowed them to apply DevSecOps practices effectively to the modernized components.

Security Automation and Orchestration: Capital One leveraged security automation and orchestration tools to streamline security processes and responses. They implemented security incident and event management (SIEM) systems, security orchestration, automation, and response (SOAR) platforms, and advanced threat detection mechanisms to enhance their security operations.

References:

- Capital One Successful DevOps Journey Secrets | Start, Fail Fast, Repeat
- DevSecOps - DevX
- How Cloud & Ops teams transform cloud adoption into success
- AWS re:Invent 2020: Capital One's Journey to being "all in" on the cloud

Fast-growing Scaleup with Higher Risk Appetite

One example of a large company successfully implementing DevSecOps is Netflix. They adopted DevSecOps practices to ensure the security of their applications and infrastructure while maintaining high agility and rapid deployment cycles. Here's an overview of how Netflix implemented DevSecOps.

Embracing the DevOps Culture: Netflix fostered a culture of collaboration and shared responsibility among development, operations, and security teams. They encouraged cross-functional teams to work together throughout the software development lifecycle.

Automating Security Processes: Netflix automated security processes to integrate security checks and controls into their continuous integration and continuous deployment (CI/CD) pipelines. They implemented a range of automated security tools and scanners to identify vulnerabilities and configuration issues early in the development process.

Continuous Security Testing: Netflix performed continuous security testing throughout the development lifecycle, including static application security testing (SAST), dynamic application security testing (DAST), and software composition analysis (SCA). This allowed them to detect and fix security issues quickly.

Infrastructure as Code (IaC): Netflix used infrastructure as code principles to manage their cloud infrastructure. They employed tools like AWS CloudFormation and Netflix's open-source tool called Spinnaker to automate the provisioning and configuration of infrastructure resources securely.

Security Monitoring and Incident Response: Netflix implemented robust security monitoring systems to detect and respond to security events in real-time. They used a combination of security information and event management (SIEM) tools, log analysis, and threat intelligence to identify and mitigate security incidents promptly.

Collaboration and Knowledge Sharing: Netflix emphasized collaboration and knowledge sharing among teams. They conducted regular security training sessions, organized hackathons, and encouraged open discussions to improve security awareness and practices across the organization.

References:

- DevSecOps - DevX
- Full Cycle Developers at Netflix — Operate What You Build
- AWS re:Invent 2015 | (SEC310) Keeping Developers and Auditors Happy in the Cloud

Private High-Flying Startup Growing into a Scaleup[5]

One example of a startup growing into a scaleup DevSecOps implementation is the special objects marketplace Catawiki. They adopted DevSecOps practices to ensure the security of their marketplace and microservices without impeding the state of their deployments and platform availability. Here's an overview of how Catawiki implemented DevSecOps.

Embracing Collaboration Between Developers and Security: One of the first milestones in Catawiki's DevSecOps journey was to dismantle the "You handle Security, I handle Coding" mentality. As a part of this process, the role of Security evolved from being a guarded quality gate to becoming an enabler that provides continuous feedback and input throughout the software development lifecycle. This new approach fostered trust and closer cooperation between the Security, Development, and Infrastructure teams.

Standardizing Components: Given the varying levels of maturity and skill sets among development teams, Catawiki recognized the importance of creating a common base that everyone can rely on and propose improvements. They established standard frameworks, templates, and Infrastructure as Code (IaC) workflows, allowing security to be easily incorporated.

Building-in Security: Catawiki automated security processes to integrate security monitoring into their continuous integration and deployment (CI/CD) pipelines. They judiciously selected automated security tools compatible with their existing frameworks and technologies to align perfectly with their current work methodology and prevent technology dispersion.

Implementing Continuous Assessment and Single Pane of Glass Visibility: To maximize the adoption of security throughout the Software Development Life Cycle (SDLC), Catawiki prioritized making security tools and reports accessible to developers. They also developed straightforward processes to simplify tool adoption and maximize value from a developer's perspective.

Running Bug Bounty Programs and Encouraging Responsible Disclosures: Recognizing that security can only be genuinely verified from an external perspective, Catawiki routinely invites independent security researchers to examine their platform. The resolution of vulnerabilities is jointly prioritized by the Security and Development teams based on the Common Vulnerability Scoring System (CVSS) and internal Service Level Agreements (SLAs) in alignment with our Vulnerability and Patch Management Policy.

Gamifying Vulnerability Management: Besides enforcing internal SLAs, Catawiki motivates non-security employees to remediate vulnerabilities by gamifying service security scores. They award team points and swags for resolving vulnerabilities. They also enable security champions to deepen their security understanding through additional training and workshops.

5 Source: https://medium.com/catawiki-engineering/catawikis-devsecops-journey-c826fe7a9030

Banking Startup Embraces DevSecOps

While numerous banks continue to operate using antiquated legacy technologies, the landscape is evolving thanks to the enhanced security and performance benefits cloud-native systems offer. Within this evolving landscape, the startup 10x Future Technologies is gaining prominence. They are pioneering the development of the cloud-native banking platform designed for large-scale banks, effectively addressing the cost and security challenges stemming from legacy systems.

Here's a glimpse into how 10x Future Technologies integrated DevSecOps into their overall Strategy.

Fostering Partnership and Collaboration Between Security and Development Teams through the usage of toolset: At 10x, fostering collaboration between security teams and development teams involves granting everyone access to a real-time dashboard that tracks vulnerabilities identified by the tools. Leveraging these toolsets, enables alerting the presence of vulnerabilities, ensuring transparency and accessibility for all team members. This transparency serves as a vital mechanism to ensure that every team comprehends their role in the collective responsibility of delivering secure software.

Shared Responsibility for Security: The organization employs essential communication strategies to enhance involvement in security practices during the development of the cloud-native banking platform. By prioritizing transparency, facilitating efficient communication, promoting teamwork in daily tasks, and proactively implementing security measures, 10x integrates security as a shared responsibility for all team members.

Putting into practice the process of Continuous Security Review of Vulnerabilities: The collaboration extends beyond mere visibility into existing vulnerabilities. By convening daily stand-up meetings that involve representatives from diverse teams across the organization, 10x guarantees that a wide range of viewpoints is taken into account when addressing ongoing security issues. This daily review of vulnerabilities across all teams during stand-up meetings represents a pivotal strategy to ensure the organization's preparedness in tackling the ever-evolving threat landscape.

Ingraining Security within the Organizational Culture: The interactions significantly enhance team effectiveness, and the presence of open communication channels among these teams has enabled for ingraining security within the organizational culture. This approach further reinforces the idea of collective responsibility for secure software, ensuring its intrinsic integration into the organization's ethos. It's a valuable method through which 10x cultivates increased empathy and mutual understanding among different roles within the company.

Through a dedicated emphasis on the pillars of people, processes, and technologies, 10x has been able to implement unique practices that have built a culture of security around software delivery. Their approach demonstrates how all three elements come together in a cooperative fashion to ensure 10x delivers software that is efficient, reliable, and secure

References:

- Integrating security into development
- DevSecOps Examples | Successes and Lessons Learned | Snyk

Convergence Between DevSecOps and Other Technology Practices

This section will give a brief overview of how some of the principles and communication processes elaborated in this whitepaper can be applied to other Technology Practices and collaboration between DevSecOps and these other technology practices for a better security outcome. The Technology Practices that we cover in this section are Zero Trust, AIOps and MLSecOps, and Data Mesh.

DevSecOps and Zero Trust

DevSecOps Overview

DevSecOps is an approach that integrates security practices into the DevOps pipeline, emphasizing security as a shared responsibility among development, operations, and security teams. Its primary principles include automation, collaboration, continuous integration, continuous delivery (CI/CD), and feedback loops.

Zero Trust Overview

Zero Trust is a security strategy that assumes no entity, whether inside or outside an organization's network, should be trusted by default. It verifies the identity and trustworthiness of every user, device, and application attempting to access resources, regardless of location. Key principles of Zero Trust include identity and access management, micro-segmentation, and continuous monitoring.
The Intersection of DevSecOps and Zero Trust
The intersection of DevSecOps and Zero Trust is where security becomes a foundational element throughout the software development lifecycle.

Secure Design and Architecture

Collaboration between the Zero Trust and DevSecOps teams during the secure design and architecture phase is essential. Zero Trust promotes the principle of "never trust, always verify." It encourages the design of network and application architectures that do not inherently trust any entity, including those within the organization's perimeter. This means micro-segmentation, strict access controls, and user identity verification from the outset. The Zero Trust team can guide how to implement effective micro-segmentation controls, while DevSecOps can ensure these controls are integrated into the architecture. This collaborative effort ensures that the design is both secure and aligns with the "never trust, always verify" philosophy of Zero Trust.

Using micro-segmentation as an example:

- Zero Trust Architect: Provides guidance to the DevSecOps Engineer on designing micro-segmentation controls for the new cloud-based application. This guidance may include defining the micro-segmentation zones, identifying the applications and services that need to be micro-segmented, and recommending micro-segmentation tools and technologies.
- DevSecOps Engineer: Works with the development team to integrate micro-segmentation controls into the application architecture. This may involve configuring cloud-based security tools, developing custom micro-segmentation solutions, and testing and validating the micro-segmentation controls.

Secure Coding

Collaboration between Zero Trust and DevSecOps teams is crucial during the secure coding phase. Zero Trust emphasizes the need to validate and verify the identity of users and applications, which aligns with DevSecOps' focus on secure coding practices. DevSecOps teams should collaborate with the Zero Trust team to define and enforce identity-based security policies, ensuring that only authenticated and authorized entities that exhibit safe behavior can access the code. This collaborative approach ensures that secure coding practices and identity verification are integrated seamlessly into the development process.

Using User Identity and applications as an example:

- Zero Trust Architect: Guides the DevSecOps Engineer on defining and enforcing identity-based security policies for the code. This guidance may include defining the roles and permissions that users need to access the code, identifying the applications and services that need to be able to access the code, and recommending identity verification and access control technologies.
- DevSecOps Engineer: Works with the development team to implement identity-based security policies for the code. This may involve configuring source control systems to enforce identity-based access controls, integrating with identity management systems to verify the identity of users and applications, and using code analysis tools to identify and fix security vulnerabilities related to identity and access control.

Continuous Build, Integration, and Testing

In the continuous build, integration, and testing phase, collaboration between Zero Trust and DevSecOps teams is essential for continuous identity validation and access controls. Zero Trust's emphasis on continuous validation aligns with DevSecOps' use of automated security testing tools. The Zero Trust team can guide how to continuously validate user and application identities while DevSecOps integrates these principles into the CI/CD pipeline. Collaboration ensures that access controls are re-evaluated for each request and automated security testing tools align with the ZT model of ongoing validation.

- Zero Trust Architect: Guides the DevSecOps Engineer on implementing continuous identity validation and access controls in the CI/CD pipeline. This guidance may include

recommending methods for validating user and application identities, identifying the tools and technologies that can be used to implement continuous identity validation and access controls, and providing guidance on integrating continuous identity validation and access controls into the CI/CD pipeline.DevSecOps Engineer: Works with the development team to implement continuous identity validation and access controls in the CI/CD pipeline. This may involve configuring CI/CD tools to perform identity validation checks, integrating with identity management systems to verify the identity of users and applications, and using automated security testing tools to identify and fix security vulnerabilities related to identity and access control.

Continuous Delivery and Deployment

Collaboration between Zero Trust and DevSecOps teams during the continuous delivery and deployment phase ensures that access to deployed resources is tightly controlled and that access is granted only to necessary resources. Zero Trust's emphasis on access control and authentication aligns with DevSecOps' use of security checks and automated responses. The Zero Trust team can guide the implementation of access control policies while DevSecOps integrates them into the deployment pipeline. Collaboration ensures that trustworthiness is verified for each access request and automated responses are triggered for suspicious behavior.

- Zero Trust Architect: Provides guidance to the DevSecOps Engineer on defining and enforcing access control policies for the deployed application. This guidance may include defining the roles and permissions that users need to have to access the application, identifying the resources that users need to be able to access, and recommending access control technologies.
- DevSecOps Engineer: Works with the development team to implement access control policies in the deployment pipeline. This may involve configuring infrastructure as code tools to enforce access control policies, integrating with identity management systems to verify the identity of users and applications, and using security scanning tools to identify and fix security vulnerabilities related to access control.

Runtime Defense and Monitoring

Collaboration between Zero Trust and DevSecOps teams is critical for continuous monitoring and behavior analytics during the runtime defense and monitoring phase. Zero Trust principles align with DevSecOps' focus on continuous monitoring and real-time feedback. The Zero Trust team can provide insights into implementing network segmentation, micro-segmentation, and behavioral monitoring while DevSecOps ensures these measures are effectively maintained. Collaboration ensures that any suspicious behavior triggers investigation and automated responses, aligning with both continuous monitoring principles and the proactive stance of Zero Trust.

- Zero Trust Architect: Guides the DevSecOps Engineer on implementing continuous monitoring and behavior analytics for the deployed application. This guidance may include recommending tools and technologies for continuous monitoring and behavior analytics,

providing guidance on configuring continuous monitoring and behavior analytics tools to detect suspicious activity, and helping to define policies for responding to suspicious activity.

- DevSecOps Engineer: Works with the operations team to implement continuous monitoring and behavior analytics for the deployed application. This may involve configuring monitoring tools to collect data on application activity, deploying security tools to detect suspicious activity in the collected data, and integrating with security orchestration, automation, and response (SOAR) tools to automate responses to suspicious activity.

Summary of DevSecOps and Zero Trust

The intersection of DevSecOps and Zero Trust is particularly powerful when considering the various phases of the software development lifecycle. DevSecOps embeds security into every stage of development, while Zero Trust reinforces the need to constantly validate and verify access, even in an increasingly dynamic and decentralized environment. Together, they create a holistic and robust security framework that safeguards applications and infrastructure from design through deployment and runtime defense.

The collaboration between Zero Trust and DevSecOps teams is key to implementing a comprehensive security framework that spans the entire software development lifecycle. Zero Trust principles reinforce the importance of continuous validation and access control, while DevSecOps ensures that security is integrated into every development phase. This collaboration results in a robust security posture that safeguards applications and infrastructure from design through deployment, runtime defense, and monitoring.

MLSecOps and AIOps

MLOps, an evolutionary branch of the well-established DevOps paradigm, is important in realizing the full potential of machine learning models. MLOps ensures the efficiency and effectiveness of machine learning models by orchestrating the operationalization process within an optimized environment, encompassing good infrastructure configurations, adequate model development, automated deployment, and ongoing performance monitoring. In parallel, the incorporation of security measures and privacy considerations into this workflow is championed by MLSecOps, which guarantees the safeguarding of data, the protection of deployed models, and underlying infrastructure against diverse threats.

The term AIOps was coined by Gartner. AIOps is the application of AI capabilities to make operational workflows more efficient. AIOps uses big data and machine learning to:

1. Collect huge volumes of data generated by infrastructure, applications, services
2. Apply various transformations to data to extract features
3. Train Machine Learning Models and build behavioral baselines
4. Apply Machine Learning models to generate signals

5. Use of machine learning, symbolic AI, AI-based Planning, and Ontological reasoning to to correlate signals
6. Apply Machine Learning models for correlating these signals and Root-Cause Analysis (RCA).

The integration of MLSecOps and MLOps signifies a big stride towards building machine learning systems that are not only proficient but also secure and reliable. While the principles of DevSecOps have proven adaptable and translatable to emerging domains like AIOps and MLSecOps, a few nuances exist within their implementation that warrant exploration.

On this note, we will consider how DevSecOps can be combined with AIOps to drive efficiency, security, reliability of IT operations and later highlight how MLSec0ps implements controls and safeguards.

Convergence in DevSecOps and AIOps

Early Threat Detection and Prevention: AIOps excels at sifting through extensive data streams, and security events. This ensures prompt discovery of unusual patterns, and potential threats swiftly. Through the application of machine learning algorithms, AIOps can pinpoint abnormal behavior, vulnerabilities in security and foresee risks, allowing proactive measures to mitigate potential security incidents before they disrupt production environments.

Intelligent Security Monitoring: The capability of AIOps extends to monitoring and analyzing security-related events, logs, and network traffic. AI-driven anomaly detection and intrusion detection, significantly reduces the lag time between detection and necessary action by security teams.

Predictive Analytics for Risk Mitigation: Through the analysis of prior security incidents and threat intelligence, AIOps provides insights into potential future risks. This equips DevSecOps teams with the information they need to prioritize tasks, allocate resources efficiently, and proactively remediate vulnerabilities, thus lowering the probability of security breaches.

Automation of Security Processes: AIOps extends its automation capability to security processes within the DevSecOps pipeline. Tasks like vulnerability scanning, security testing, compliance checks, and configuration management can be automated with AIOps.

Enhanced Incident Response and Remediation: In the unfortunate event of a security breach, AIOps facilitates incident response and remediation. AI is applied to generate summaries of Security Incidents which can streamline Security Analyst onboarding time, Real-time analysis of security events, incident correlation, and generate actionable insights to provide guidance to security teams in their response endeavors.

Distinguishing Factors that Shape MLSecOps

While the foundations align, the distinctive nature of machine learning introduces specific challenges that necessitate tailored security measures. Here are five pivotal areas where MLSecOps diverges from its DevSecOps counterpart.

Model Provenance: Model Provenance offers a comprehensive model history, spanning development, deployment, and usage. This data aids in pinpointing potential vulnerabilities by tracking alterations to a model, including contributors, methods, timing, and rationale.

In the context of evolving AI regulations, Model Provenance gains prominence in showcasing compliance during assessments. By unveiling the journey of machine learning model development, deployment, and utilization, Model Provenance brings transparency and accountability. This fosters responsible data usage and sustains model trustworthiness amid escalating cyber threats.

Governance, Risk & Compliance: A lot of emphasis is being placed on the Machine Learning Bill of Materials (MLBoM) as a way to enhance Risk, Governance, and Compliance. This framework addresses challenges in MLSecOps, encompassing GRC and Supply Chain Vulnerability, and provides an exhaustive inventory of materials in machine learning model development, such as algorithms, data sets, and frameworks. It serves to uncover potential vulnerabilities in the supply chain, like malicious code or tampered components, that could compromise model integrity.

MLBoM ensures model compliance with legal, regulatory, and ethical standards, safeguarding data privacy. A robust Bill of Materials aids in tracing data, models, and algorithms' origins, ensuring they're auditable and explainable. Adopting a comprehensive MLBoM heightens security, transparency, and accountability of machine learning systems.

Trusted AI: Bias, Fairness and Explainability: Trusted AI refers to AI systems designed for fairness, lack of bias, and transparency. The aim is equitable decision-making irrespective of personal traits. Achieving this requires the adoption and use of a well-defined Framework such as Deloitte TrustworthyAI. The framework helps to manage common risks and challenges related to AI ethics and governance, including:

- *Fair and impartial use checks:* To ensure fairness and impartiality in AI, companies must address biases introduced during coding by actively identifying and mitigating biases in algorithms and data. This involves defining fairness standards and implementing controls to prevent unexpected outcomes.
- *Implementing transparency and explainable AI:* To establish trustworthy AI, ensuring transparency and explainability is crucial. Everyone involved should comprehend data usage and AI decision-making. Organizations must be ready to make algorithms, training data, attributes, and correlations accessible for inspection.
- *Responsibility and accountability:* Creating trustworthy AI involves defining clear policies for identifying responsible parties accountable for the system's outcomes.
- *Adding Appropriate Security Controls:* Ensuring trustworthy AI entails safeguarding it from various risks, including cybersecurity threats that could result in harm. Companies must comprehensively assess and mitigate risks while transparently communicating them to users.

- *Monitoring for reliability:* To enable broad AI adoption, it should match the reliability of traditional systems and processes. Companies must verify that AI algorithms yield anticipated outcomes for different data sets and have protocols in place to address problems or inconsistencies that may emerge.
- *Safeguarding privacy:* To establish trust in AI, privacy must be upheld through adherence to data regulations and limited data use as stipulated by the GDPR, and other frameworks. Organizations must respect consumer privacy, avoid unauthorized data use, and offer opt-in/out choices for data sharing.

Adversarial Machine Learning: This field addresses safeguarding machine learning models from malicious attacks. Attacks can involve manipulating input data to mislead predictions or altering the model itself to reduce accuracy. Adversarial ML aims to create defenses against these attacks, enhancing model robustness and security. Safeguards like threat modeling and threat impact analysis, multiple classifier systems, real-time attack detection, generative models for synthetic training data, and adversarial examples in training are commonly employed by researchers to achieve these goals.

Supply Chain Vulnerability in Machine Learning: This refers to potential security risks in the components and systems of machine learning supply chains, encompassing data storage, software, hardware, and communication networks. Exploitation by hackers can lead to data breaches, operational disruptions, and data theft. Countermeasures involve robust security protocols, constant system updates, and proactive monitoring. Organizations relying on machine learning must adopt a comprehensive vulnerability assessment and prevention strategy to ensure data and operational security. Such an approach enhances compliance, reputation, and customer trust, and safeguards against financial and reputational harm. To conduct your own machine learning supply chain vulnerability assessments, consider reaching out to Protect AI and exploring their AI Radar solution.

References

Cloud Security Alliance, The Six Pillars of DevSecOps: Pillar 3, Pragmatic Implementation, December 2022, Available @ https://cloudsecurityalliance.org/artifacts/six-pillars-devsecops-pragmatic-implementation /

Cloud Security Alliance, The Six Pillars of DevSecOps: Pillar 4 - Bridging Compliance and Development, Feb 2022, Available @ https://cloudsecurityalliance.org/artifacts/devsecops-pillar-4-bridging-compliance-and-development/

Cloud Security Alliance, The Six Pillars of DevSecOps: Pillar 5 - Automation, July 2020, Available @ https://cloudsecurityalliance.org/artifacts/devsecops-automation/

Cloud Security Alliance, The Six Pillars of DevSecOps: Pillar 1 - Collective Responsibility, February 2020, Available @ https://cloudsecurityalliance.org/artifacts/devsecops-collective-responsibility/

Cloud Security Alliance, Information Security Management through Reflexive Security: Six Pillars in the Integration of Security, Development and Operations, August 2019, Available @ https://cloudsecurityalliance.org/artifacts/information-security-management-through-reflexive-security/

Cloud Security Alliance, The Six Pillars of DevSecOps, *Achieving Reflexive Security Through Integration of Security, Development and Operations*, August 2019, Available @ https://cloudsecurityalliance.org/artifacts/six-pillars-of-devsecops/

Integrating Zero Trust and DevSecOps, SOFTWARE ENGINEERING INSTITUTE | CARNEGIE MELLON UNIVERSITY, July 2021, Available @ https://apps.dtic.mil/sti/pdfs/AD1145432.pdf

NSTAC Report to the President on Zero Trust and Trusted Identity Management, February 2023, Available @ https://www.cisa.gov/sites/default/files/2023-04/NSTAC_Strategy_for_Increasing_Trust_Report_%282-21-23%29_508_0.pdf

RSA Conference, Implement ZeroTrust with Dedicated DevSecOps Pipeline, April 2023, Available @ https://www.youtube.com/watch?v=4DREGC-Z_F0

NIST Special Publication NIST SP 800-204D ipd Strategies for the Integration of Software Supply Chain Security in DevSecOps CI/CD Pipelines, August 30, 2023 Draft, Comments Open until October 2023, Available @ https://csrc.nist.gov/pubs/sp/800/204/d/ipd

MLSecOps: Defined.
What is MLSecOps

Deloitte Press Release
Deloitte Introduces Trustworthy AI Framework – Press release | Deloitte US

Scaling DevsecOps with AIOps: The Power of Artificial Intelligence in Driving Efficiency and Security
Scaling DevSecOps with AIOps: The Power of Artificial Intelligence in Driving Efficiency and Security | LinkedIn