NIST SPECIAL PUBLICATION 1800-44A

Secure Software Development, Security, and Operations (DevSecOps) Practices

High-Level Document

Volume A

Alper Kerman
Michael Ogata
National Institute of Standards and Technology

Parisa Grayeli
Phillip Millwee
Allen Tan
The MITRE Corporation

William Barker
Dakota Consulting

Sudan Ayanam
Stefano Righi
AMI

Al Bessey
Chrissa Constantine
Tim Mackey
Black Duck

Rahul Dubey
James Imanian
Evan Litwak
CyberArk

Daniel Carroll
Daniel Jackson
Dell Technologies

Corey Bonnell
Dean Coclin
Dave Roche
DigiCert

Tom Gleason
Ron Harnik
Karl Mattson
Shruti Sundaresh
Endor Labs

Sameer Kamani
Paul Pickhardt
MaryGrace Wajda
GitLab

Isaac Hepworth
Brandon Lum
Leah Rivers
Google

Pradeep Balachandran
Jen Gilbert
Philippe Mulet
Ritchie Schacher
Harmeet Singh
IBM

Adrian Diglio
Toddy Mladenov
Segu Riluvan
Mark Svancarek
Microsoft

Tony Berning
Keng Lim
Sameer Shukla
NextLabs

Sean Morgan
Alfredo Motta
Norman Wong
Palo Alto Networks

Jose Palazon
Michael Smith
Sagittal AI

Rubi Arbel
Daniel Nebenzahl
Nir Peleg
Scribe Security

July 2025

FIRST PRELIMINARY DRAFT

This publication is available free of charge from https://www.nccoe.nist.gov/projects/software-supply-chain-and-devops-security-practices

NIST NATIONAL CYBERSECURITY CENTER OF EXCELLENCE

DISCLAIMER

Certain commercial entities, equipment, products, or materials may be identified by name or company logo or other insignia in order to acknowledge their participation in this collaboration or to describe an experimental procedure or concept adequately. Such identification is not intended to imply special status or relationship with NIST or recommendation or endorsement by NIST or NCCoE; neither is it intended to imply that the entities, equipment, products, or materials are necessarily the best available for the purpose.

National Institute of Standards and Technology Special Publication 1800-44A, Natl. Inst. Stand. Technol. Spec. Publ. 1800-44A, 18 pages, (July 2025), CODEN: NSPUE2

FEEDBACK

You can view or download the first preliminary draft guide at the NCCoE Software Supply Chain and DevOps Security Practices project page.

NIST will use an agile process to make updates available as the project continues. We are asking for feedback on this first preliminary draft.

Comments on this publication may be submitted to: nccoe-devsecops@list.nist.gov

Public comment period: July 30, 2025, through September 14, 2025

All comments are subject to release under the Freedom of Information Act.

NIST is particularly interested in your feedback on the following questions:

1. What practices would be most helpful in addressing the challenges with secure software development?
2. How do you expect this guide to influence your future practices and processes? What guidance would be most helpful to meet your expectations?
3. How do you envision using this guide? What changes would you like to see to increase/improve that use?
4. What suggestions do you have on changing the format of the provided information?

National Cybersecurity Center of Excellence
National Institute of Standards and Technology
100 Bureau Drive
Mailstop 2002
Gaithersburg, MD 20899
Email: nccoe@nist.gov

NATIONAL CYBERSECURITY CENTER OF EXCELLENCE

The National Cybersecurity Center of Excellence (NCCoE), a part of the National Institute of Standards and Technology (NIST), is a collaborative hub where industry organizations, government agencies, and academic institutions work together to address businesses' most pressing cybersecurity issues. This public-private partnership enables the creation of practical cybersecurity solutions for specific industries, as well as for broad, cross-sector technology challenges. Through consortia under Cooperative Research and Development Agreements (CRADAs), including technology partners—from Fortune 50 market leaders to smaller companies specializing in information technology security—the NCCoE applies standards and best practices to develop modular, adaptable example cybersecurity solutions using commercially available technology. The NCCoE documents these example solutions in the NIST Special Publication 1800 series, which maps capabilities to the NIST Cybersecurity Framework and details the steps needed for another entity to re-create the example solution. The NCCoE was established in 2012 by NIST in partnership with the State of Maryland and Montgomery County, Maryland.

To learn more about the NCCoE, visit https://www.nccoe.nist.gov/. To learn more about NIST, visit https://www.nist.gov.

NIST CYBERSECURITY PRACTICE GUIDES

NIST Cybersecurity Practice Guides (Special Publication 1800 series) target specific cybersecurity challenges in the public and private sectors. They are practical, user-friendly guides that facilitate the adoption of standards-based approaches to cybersecurity. They show members of the information security community how to implement example solutions that help them align with relevant standards and best practices and provide users with the materials lists, configuration files, and other information they need to implement a similar approach.

The documents in this series describe example implementations of cybersecurity practices that businesses and other organizations may voluntarily adopt. These documents do not describe regulations or mandatory practices, nor do they carry statutory authority.

ABSTRACT

Development Operations (DevOps) bring together software development and operations to shorten development cycles, allow organizations to be agile and maintain the pace of innovation while taking advantage of cloud-native technology and practices and the increasing industry use of rapidly evolving artificial intelligence (AI) capabilities. Development, Security, Operations (DevSecOps) emphasizes this philosophy by continuously addressing security throughout all phases of the software development lifecycle. Modern software is the synthesis of a wide array of components and processes, some of which are under the direct control of the software producer while others are part of a large, interconnected, and often opaque supply chain. Much of the DevSecOps methodology relies on automated production flows, which can quickly propagate security risks directly into production if they are not caught and corrected early in the development process. To help improve the security of DevSecOps practices, the NCCoE is executing a project that focuses on demonstrating and documenting applied risk-based approaches and recommendations for DevSecOps practices consistent with the NIST Secure Software

Development Framework (SSDF) [1][2]. This project demonstrates these DevSecOps practices by implementing example software development processes.

KEYWORDS

Cybersecurity supply chain risk management; *DevOps; DevSecOps; Secure software development; Secure Software Development Framework (SSDF); Software Supply Chain; Supply chain security.*

ACKNOWLEDGMENTS

We are grateful to the following individuals for their generous contributions of expertise and time.

- AMI: Veerajothi Ramasamy
- Black Duck: Rod Musser, Brendon Rizzolo, Rick Smith
- CyberArk: David Dennenberg, Joshua Freeman, Oana Garnett, Steve Judd, Darren Khan, Grayson Miller, Gram Slingbam, Bryan Sowell, Riaz Vellamparambil, Nathan Whipple
- Dell Technologies: Mukund Khatri, Sean Schwoerer, Sam Sehgal
- DigiCert: Mohan Dattatreya, Tim Hollebeek, Sam Merrill, Bob Vogt
- GitLab: Joel Krooswyk*
- Google: Bob Callaway, Chris Cornillie, Wendy Dembowski, Kim Gajewski, Tom Hennen, Shmuel Herzberg, Al Huizenga, Dustin Ingram, Stephanie Kiel
- IBM: Anamika Agrawal, Piyush Mundra, Smith Naik, Nic Sauriol
- Microsoft: Andrew Brenner, Nick Couraud, Pedro Enriquez, Lara Goldstein, Dick Lake, Tony Rice, Zachary Steindler, Betty Tso, Michael Yomokoh, Yi Zha
- MITRE: Jonathan Davis, John Kent, Lauren Swan, Theresa Suloway, Thomas Walters
- NextLabs: Johnwin Johnrose, Emmanuel Thioux
- NIST: Cherilyn Pascoe, Kevin Stine
- Palo Alto Networks: David Kubicki, Neil Roxburgh, Eden Tesfay, Jeff Yuetter
- Sagittal: Sam Lacey
- Scribe Security: Guy Chernobrov

* *Former employee; all work for this publication was done while at that organization*

The Technology Partners/Collaborators who are participating in this NCCoE project submitted their capabilities in response to a notice in the Federal Register. Respondents with relevant capabilities or product components were invited to sign a Cooperative Research and Development Agreement (CRADA) with NIST, allowing them to participate in a consortium to build this example solution. We are working with:

Technology Collaborators		
AMI	Endor Labs	NextLabs
Black Duck	GitLab	Palo Alto Networks
CyberArk	Google	Sagittal AI
Dell Technologies	IBM	Scribe Security
DigiCert	Microsoft	

DOCUMENT CONVENTIONS

The terms “shall” and “shall not” indicate requirements to be followed strictly to conform to the publication and from which no deviation is permitted. The terms “should” and “should not” indicate that among several possibilities, one is recommended as particularly suitable without mentioning or excluding others, or that a certain course of action is preferred but not necessarily required, or that (in the negative form) a certain possibility or course of action is discouraged but not prohibited. The terms “may” and “need not” indicate a course of action permissible within the limits of the publication. The terms “can” and “cannot” indicate a possibility and capability, whether material, physical, or causal.

CALL FOR PATENT CLAIMS

This public review includes a call for information on essential patent claims (claims whose use would be required for compliance with the guidance or requirements in this Information Technology Laboratory (ITL) draft publication). Such guidance and/or requirements may be directly stated in this ITL Publication or by reference to another publication. This call also includes disclosure, where known, of the existence of pending U.S. or foreign patent applications relating to this ITL draft publication and of any relevant unexpired U.S. or foreign patents.

ITL may require from the patent holder, or a party authorized to make assurances on its behalf, in written or electronic form, either:

a) assurance in the form of a general disclaimer to the effect that such party does not hold and does not currently intend holding any essential patent claim(s); or

b) assurance that a license to such essential patent claim(s) will be made available to applicants desiring to utilize the license for the purpose of complying with the guidance or requirements in this ITL draft publication either:

1. under reasonable terms and conditions that are demonstrably free of any unfair discrimination; or
2. without compensation and under reasonable terms and conditions that are demonstrably free of any unfair discrimination.

Such assurance shall indicate that the patent holder (or third party authorized to make assurances on its behalf) will include in any documents transferring ownership of patents subject to the assurance, provisions sufficient to ensure that the commitments in the assurance are binding on the transferee, and that the transferee will similarly include appropriate provisions in the event of future transfers with the goal of binding each successor-in-interest.

The assurance shall also indicate that it is intended to be binding on successors-in-interest regardless of whether such provisions are included in the relevant transfer documents.

Such statements should be addressed to: nccoe-devsecops@list.nist.gov

Contents

List of Figures

Executive Summary

DevOps embodies a modern approach to software development by abolishing the traditional silos between development and operations teams. It emphasizes collaboration, automation, and continuous improvement by removing delays and inefficiencies while boosting efficiency and quality in the software development lifecycle. However, with the evolving cybersecurity risks and rising security breaches, most organizations that produce software have been increasingly integrating security into their DevOps environments, tools, and processes – a practice known as DevSecOps.

DevSecOps is a software development paradigm that prioritizes integration of security practices from the very beginning of the development process (a "before-thought" approach) rather than addressing them as a separate concern in later stages or post-deployment (an "after-thought" approach). There are various reasons for growing adoption of DevSecOps:

- The growing complexity and volume of cyber threats have made software development environments prime targets for various forms of attacks and cybersecurity breaches. Bad actors are preying on overlooked security practices in software development infrastructures and tools, misconfigurations in cloud environments, and weak access controls that lead to vulnerabilities. DevSecOps practice can help organizations build more resilient systems to protect against these types of threats.

- Most software today relies on one or more third-party components, yet organizations often have little or no visibility into or understanding of how these components are developed, integrated, deployed, and maintained, as well as the practices used to ensure the components' security. DevSecOps tools can scan, identify, discover, and help eliminate vulnerabilities in third-party components early in the development lifecycle.

- The growing adoption of artificial intelligence (AI) in DevSecOps is making significant impacts in enhancing threat detection and prevention, security testing and remediation, preserving proactive security posture while also enhancing efficiency, speed, and quality in the software development process by automating tasks such as code integration, testing, deployment and monitoring that lead to consistently reliable and quicker software production.

The National Cybersecurity Center of Excellence (NCCoE) is demonstrating and documenting risk-based security practices for DevSecOps, aligning with the NIST Secure Software Development Framework [1][2] Initially, this project will focus on securing cloud-based environments that resemble typical closed-source software development settings, highlighting dynamic enforcement of least privileged access through a practical Zero Trust Architecture (ZTA) implementation. Moreover, the project will showcase a holistic approach to secure software development, embedding security considerations and best practices as well as leveraging AI throughout the phases of the secure software development process to automate builds, integrations, deliveries, and deployments that lead to trustworthy and faster software development.

This project will result in vital and novel details concerning the possible implementations of DevSecOps that are often unavailable to the community as a whole. It will allow practitioners to evaluate, compare,

and identify gaps in existing software factories, which enables and enhances cybersecurity for both software producers and consumers.

1 Introduction

This document, in its preliminary draft state, only provides an overview of a National Cybersecurity Center of Excellence (NCCoE) project at this time. The primary goals of this project are to implement software development processes, document their construction using currently available technologies, and illustrate how their design can improve cybersecurity in software development. For the purposes of this project, a software development process includes the activities surrounding the development of software from inception to its delivery to a software consumer. A key objective of this project is to demonstrate software development processes that have the following characteristics:

- integrates security practices into existing processes and toolchains used by developers and managed by operations teams that manage continuous integration/continuous delivery (CI/CD)
- automates security and compliance artifact generation to enhance efficiency
- reduces vulnerabilities and mitigates risks throughout the software development process
- addresses root causes of vulnerabilities to prevent future occurrences
- enhances collaboration between development, operations, and security teams to maintain agility and innovation while strengthening security
- adheres to zero trust principles and approaches
- uses AI capabilities as assistance tools to proactively identify and assist to remediate security threats, automate repetitive tasks, and provide actionable insights for continuous improvement of the security posture throughout the software development lifecycle

This NIST NCCoE project brings together and normalizes content regarding DevSecOps practices from existing guidance and practices publications. Selected NIST guidance most closely related to DevOps and supply chain security, such as NIST Special Publication (SP) 800-218 [1] and NIST Special Publication (SP) 800-218A [2], has been leveraged to inform the use case implementations in this project. Findings from this project may be utilized by future efforts to update NIST guidance. Additional existing security guidelines, practice recommendations, and publications from NIST (e.g., those relating to zero trust architecture (ZTA) and artificial intelligence (AI)) will inform this project.

1.1 Background

This section provides background information about DevSecOps, the role of AI in software development and the role of Zero Trust in software development.

1.1.1 Development, Security, and Operations (DevSecOps)

Development Operations (DevOps) is an organizational model that brings together software development and operations teams to improve collaboration, coordination, and efficiency through concepts such as shared ownership, software development and operational automation, and constant, rapid feedback. DevOps activities are designed to shorten development cycles, promote agile software development practices, and increase the pace of remediations and new features. DevOps practices have gained access to new tools and techniques with the rise of cloud-native technologies, microservice architectures, and serverless frameworks while also integrating the rapidly evolving technology space surrounding artificial intelligence (AI) tools and capabilities.

DevSecOps is the addition of security as a first-party contributor to the DevOps organizational model that was described earlier. By adding security to the model, this addition helps integrate security practices starting from the earliest phase in attempt to "shift left" [3], which ensures that security is included as fundamental component of any DevOps practices. DevSecOps practices implement security practices as part of software development processes, build and test automation artifact packaging and distribution, and software release or deployment management. Integration of DevSecOps has the potential to:

- reduce vulnerabilities and support the detection of malicious code and other security issues (e.g. exposed secrets/credentials) in released software without slowing down code production and releases
- mitigate the potential impact of vulnerability exploitation throughout the software lifecycle, including when the software is being developed, built, packaged, distributed, deployed, and executed on dynamic hosting platforms
- address the root causes of vulnerabilities to prevent recurrences, such as strengthening test tools and methodologies in the toolchain and improving practices for developing code and operating hosting platforms
- facilitate and enhance collaboration, communication, and consistency between the members of the development, operation, and security teams in order to maintain the speed and agility needed to support the organization's mission while taking advantage of modern and innovative technology
- improve incident response time and quality by enabling faster detection through comprehensive monitoring, automated alerting, and fostering rapid and collaborative remediation

This project illustrates how the NIST Secure Software Development Framework [1] practices and tasks can be implemented in DevSecOps to aid organizations in improving the security of the software they develop and operate. Additionally, the project demonstrates how to generate specific artifacts that can support and inform organizations' attestation and declaration conformance.

This project addresses DevSecOps in the context of current and emerging secure development frameworks, practices, and tools. NIST will share lessons learned during the project with security and software development communities with the intent of informing improvements to secure software development frameworks, practices, and tools. These lessons can also inform standards development organizations' DevSecOps-specific activities.

1.1.2 The Role of AI in Software Development

Throughout the software development lifecycle, AI is increasingly being used for the automation of the processes enhancing the security and the organization's effectiveness. AI powered tools are used to facilitate the automation of coding, security analysis, and even automated analysis for detecting and remediating vulnerabilities. The use of AI technology in software development not only improves the work efficiency but also could bring higher quality software in more timely manner. While AI can deliver significant efficiencies and other advantages, the software development teams still need to ensure AI-generated content is monitored and validated by a human and verifiable processes in place to ensure its accuracy and trust. Within DevSecOps, both human users and automated processes must monitor AI's

influence. AI-based suggestions may inadvertently be accepted by human actors without sufficient scrutiny. There is a real need to employ oversight necessary to prevent insecure and non-functional code from being inserted into the software development process.

Identifying where AI is being used, including use by third-party models, in source code, and in agents, is challenging. It is necessary to provide mechanisms for tracing models, modifications, and annotations to ensure that AI-assisted processes can be subjected to a level of review matching for human modification of software systems and applications. This project explores the responsible use of AI to extend existing DevSecOps tools and capabilities throughout the software lifecycle.

1.1.3 The Role of Zero Trust in Software Development

A Zero Trust security strategy, if adopted, can significantly strengthen the resiliency of DevSecOps environments by shrinking implicit trust zones and mitigating breach risks by imposing every access request to rigorous authentication and authorization and device security posture. Escalating cybersecurity threats, rise in remote and hybrid working options, increased reliance on cloud services and multi-cloud environments, and data protection requirements and measures, along with federal mandates for cybersecurity, push organizations to adopt zero trust as their most resilient cybersecurity strategy for today's hybrid environments. Many organizations are actively pursuing Zero Trust, some in the initial planning stages and others well into implementation across various sectors.

To strengthen the security of DevSecOps process, this project will explore the use of zero trust principles and approaches and demonstrate utilizing security checks and controls, continuous monitoring and scanning for vulnerabilities, ensuring the integrity of artifacts, verifying code commits and signatures, and employing other proactive security measures throughout the development lifecycle.

1.2 Audience

The audience for this publication is technology leaders and practitioners responsible for developing, delivering, and operating secure software systems. This group includes software developers, software systems designers, software development managers, software security specialists, software acquisition specialists and managers, and systems managers and owners. Furthermore, this document will be of interest to those responsible for enhancing collaboration between software development, operations, and security teams to maintain agility and innovation while strengthening security. Readers are assumed to understand basic DevOps and secure software development concepts.

1.3 Scope

There are many methodologies used to produce software. This project focuses on demonstrating and documenting risk-based approaches for software development using DevSecOps processes. The capabilities being demonstrated are applicable to information technology (IT) development in medium to large enterprises in multiple sectors. Our initial focus will be constructing environments that mimic representative closed-source software development environments, that is, environments that resemble those implemented by organizations that have a vested interest in protecting their intellectual property from outside tampering/observation. The development of open-source software may be addressed in a later phase of this project. Furthermore, Zero Trust principles and approaches will be also integrated to elevate the security posture of the DevSecOps lifecycle.

This project will not focus on the development of any particular technology type (i.e. the output of the software development process). Certain domains such as Operational Technology (OT) and Internet of Things (IoT) are out of scope of this project. While the project will investigate the use of AI tools in the DevSecOps lifecycle, it will not specifically address machine learning operations (MLOps) or AI bill of material (AI BOM).

1.4 Challenges

Modern software is complex and fast moving, which requires focusing on innovation and cybersecurity. This project intends to address the following challenges that apply to producing software more securely:

1. Integration: Modern software is developed and delivered using a variety of tools, automations, ecosystems, and services. Orchestrating these systems to provide the appropriate feedback and enforce security requirements is a complex and time-consuming task.

2. Attestation: Improving and evaluating security is a continuous effort. Producing evidence that security requirements have been met is vital to ensure proper application of those requirements. However, the volume, format, and content of artifacts that support security claims is poorly defined.

3. Challenges of visibility of third-party components: Another aspect of today's software is that it often uses one or more software components, either commercial or open source, that are developed by other organizations, groups, or individuals. Some of those components may also use components from other organizations. Managing cybersecurity risk from third party software components, as part of cybersecurity supply chain risk management (C-SCRM), involves identifying, assessing, selecting, and implementing processes and mitigating controls. This risk management can largely be integrated into DevSecOps through its automation capabilities, such as constant automated testing of software and software components to identify vulnerabilities, breaches of integrity, and other security issues.

4. Emergence of AI tools: AI tools are being increasingly employed throughout the software development process. While there are potential applications in code generation, code evaluation, security monitoring, and other aspects, defining the security requirements for employing these technologies is not yet fully understood.

This project will address these challenges by documenting example implementations to help demonstrate solutions to them.

2 Methodology – A Notional Reference Model for DevSecOps

This project's foundational technical task requires the construction of software development processes that implement DevSecOps practices, which necessitates the selection, configuration, and integration of many different IT technologies. The aggregate of these systems must fulfill both the technical requirements needed to deliver software and ensure that the security practices are continuously being applied throughout the process.

To guide this task, NCCoE developed a notional reference model, illustrated below in Figure 1, that implements a phased secure software development process and was developed and refined in conjunction with project's collaborators. Each phase of the development process represents a logical stage in the path as software is developed. Furthermore, it highlights our emphasis on feedback through the use of control gates that must be enforced before software changes are permitted to advance to the next stage in the lifecycle. Control gates consist of technical or organizational controls that are designed to provide the necessary feedback or evidence that enables development, operational, or security tasking for resolving discovered vulnerabilities or defects. By leveraging technologies to enforce and automate these controls, this project will demonstrate real-world, repeatable, and verifiable mechanisms to increase the security of software development.

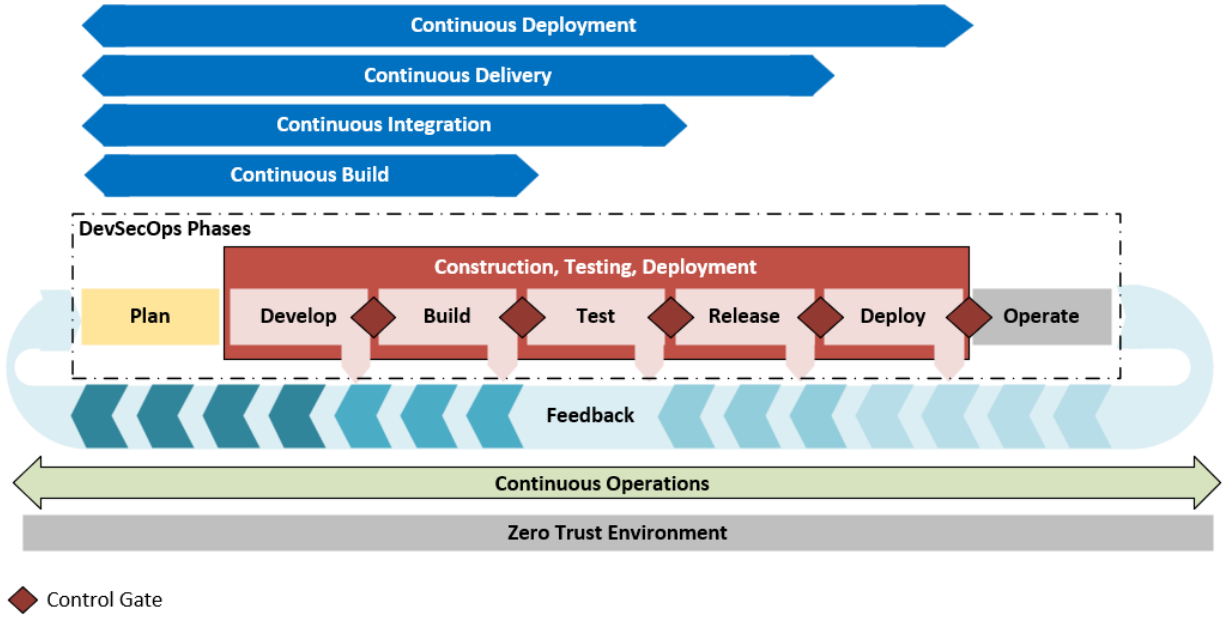

Figure 1 DevSecOps Notional Reference Model

Figure above shows the DevSecOps Notional Reference Model and five supporting concepts and technologies in support of the model used in the project:

1. DevSecOps Phases: The development lifecycle begins with the plan phase, where security considerations are incorporated from the start. The process then moves through develop, build, test, release, deploy, and operation phases as shown below:

360 a. Plan - Development, operations, and security teams collaboratively define
361 requirements, establish secure practices, and create roadmaps, updating them as
362 feedback from later phases informs new functional or security needs.
363 b. Develop - Teams collaboratively create and proactively review code, infrastructure, and
364 policies with approved tools to meet requirements and ensure quality, security, and
365 compliance early in the process.
366 c. Build- Automated pipelines transform code and configurations into deployable artifacts,
367 using ephemeral environments and integrated analysis to ensure quality, security, and
368 compliance while providing actionable feedback to all teams.
369 d. Test- Teams use automated testing and ephemeral environments to independently
370 evaluate deployable artifacts for functional, operational, and security requirements with
371 CI/CD feedback guiding future improvements.
372 e. Release - Automated processes package and transfer software to production
373 environments while teams coordinate releases, verify readiness and security, document
374 changes, and collect feedback to improve future releases.
375 f. Deploy - Automated pipelines install and configure software on production
376 infrastructure while teams monitor deployments, verify security and performance, and
377 collect feedback to ensure reliable, secure, and consistent releases.
378 g. Operate - Teams use automated monitoring and evidence collection to ensure the
379 integrity, security, and reliability of production systems, using feedback to address
380 issues and guide continuous improvement.

381 2. Feedback Cycle – While not a technology or processing component, this concept covers how
382 events or signals emitted from control gates or learned from external events (new technologies,
383 attack patterns or regulations) are fed back into earlier phases of the DevSecOps lifecycle to
384 inform changes in subsequent phases of the lifecycle.

385 3. Continuous Operations – This consists of a collection of components and activities that appear
386 throughout the DevSecOps lifecycle. These ever-present components provide phase-specific
387 integrations to enable the goals and objectives of a given phase.

388 4. Continuous Integration/Continuous Delivery (CI/CD) Pipeline – This technology provides the
389 automated pipelines (e.g., continuous build, integration, delivery and deployment) that are used
390 to run many of the components that are unique to each of the phases in the DevSecOps
391 lifecycle. Different phases of the DevSecOps lifecycle have direct influence on components and
392 activities conducted by the automated pipelines previously described. Each automated pipeline
393 functions independently from other pipelines but are combined in sequence to produce the full
394 automated process of building and deploying software to production.

395 5. Zero Trust Environment – This underlying environment includes all aspects of Zero Trust
396 Architecture (ZTA) implemented at NCCoE as part of the "Implementing a Zero Trust
397 Architecture" project. It includes ZTA components such as Identity Credential Access
398 Management (ICAM), Policy Decision Point (PDP), Policy Enforcement Point (PEP), Data Security,
399 Security Analytics, Endpoint Security and Resource Protection.

3 Next Steps

NIST intends to hold a workshop to discuss this project on August 27, 2025. Based on the feedback from this paper, NIST will demonstrate solutions consistent with the notional reference model shown in Section 2.

With the objective to exemplify SSDF security practices, the project effort will continue to focus on installing and integrating commercially offered technologies provided by the collaborators to construct multiple cloud-based platforms for demonstration. In addition, the participants will conduct recurring meetings to collaboratively determine use cases and scenarios that align with projects demonstration objectives.

Since the main output of the project is an 1800 series NIST special publication, project effort will continue to focus on producing additional revisions of the document to be released in intervals during the life of the project. The document will provide a project overview, intricate details of implementation and technologies used for demonstration of security practices, and steps needed for organizations to be able to implement the same security practices in their own environments. Furthermore, since a practical Zero Trust Architecture (ZTA) will be used to manage policy-driven access throughout DevSecOps, this document will also detail the embedded security controls and their alignment with existing guidance.

Appendix A List of Acronyms

AI	Artificial Intelligence
AI BOM	Artificial Intelligence Bill of Materials
CBOM	Crypto Bill of Materials
CI/CD	Continuous Integration/Continuous Delivery
C-SCRM	Cybersecurity Supply Chain Risk Management
DevOps	Development Operations
DevSecOps	Secure Development Operations
IoT	Internet of Things
MLOps	Machine Learning Operations
NCCoE	National Cybersecurity Center of Excellence
NIST	National Institute of Standards and Technology
OT	Operational Technology
SDLC	Software Development Lifecycle
SP	Special Publication
SSDF	Secure Software Development Framework

Appendix B References

[1] M. Souppaya et al., "Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities," *National Institute of Standards and Technology (NIST)* Special Publication (SP) 800-218, Gaithersburg, Md., February 2022, pp. 36. Available: https://doi.org/10.6028/NIST.SP.800-218.

[2] M. Souppaya et al., Secure Software Development Practices for Generative AI and Dual-Use Foundation Models: An SSDF Community Profile, *National Institute of Standards and Technology (NIST)* Special Report (SP) 800-218A Internal Report (IR) 7864, Gaithersburg, Md., July 2024, pp. 22. Available: https://doi.org/10.6028/NIST.SP.800-218A.

[3] Larry Smith, "Shift-left testing,", *Dr. Dobb's Journal*, vol. 26, Issue 9, CMP Media, Inc., Boulder, Co., September 2001, pp. 56.