The Peerian Journal

Open Access | Peer Reviewed

Volume 58 September 2026

ISSN (E): 2788-0303

Website: www.peerianjournal.com

Email: editor@peerianjournal.com

Shifting Security Left in Software Development: "A Systematic Review of Strategies, Challenges, and Tools"

Asst. Lect. Alaa Basil M. Sh.

Open Educational College / Nineveh Center / Ministry of Education

E-mail: alaa.20csp10@student.uomosul.edu.iq

https://orcid.org/0009-0009-5772-6324

Abstract

With the advent of Agile, DevOps and CI/CD, software systems are becoming more complex and developing faster, making traditional reactive security methods less effective. Many people are familiar with the paradigm of Shift-Left Security, a strategy that seeks to incorporate security activities into the initial stages of the Software Development Lifecycle (SDLC) as opposed to a last check step. A systematic evaluation of 28 peer-reviewed papers from Scopus, IEEE Xplore, ACM Digital Library and other well-known academic databases, this research pulls together some of the existing strategies, barriers, supporting tools, and benefits that have been reported in the field for implementing Shift-Left Security and DevSecOps. Selected studies were grouped into five themes: (1) DevSecOps Adoption and AI Integration, (2) Shift-Left Security Practices, (3) Organizational and Cultural Challenges, (4) Security Requirements Engineering, and (5) Security Automation Tools (SAST, DAST, IAST, SCA). The results show that early security integration helps minimize remediation costs, enhance software quality, and boost cyber resilience, and automated security testing in CI/CD pipelines allows for ongoing validation without negatively impacting delivery speed. However, persistent challenges remain, including cultural resistance among development teams, the prevalence of false positives from automated tools, limited security expertise among developers, and the absence of standardized implementation frameworks. The review concludes that successful Shift-Left Security adoption requires a synergistic combination of automated tooling, continuous developer training, executive leadership support, and a fundamental cultural transformation toward shared security responsibility. Future research should focus on AI-driven security automation, standardization of Shift-Left frameworks, and empirical studies measuring the long-term return on investment of DevSecOps practices.

The Peerian Journal

Open Access | Peer Reviewed

Volume 58 September 2026
Website: www.peerianjournal.com

ISSN (E): 2788-0303
Email: editor@peerianjournal.com

Keywords: Shift-Left Security, DevSecOps, Secure Software Development, Software Development Life Cycle (SDLC), Threat Modeling, Secure Coding, Security Testing Automation, Cybersecurity.

نقل الأمن إلى اليسار في تطوير البرمجيات: مراجعة منهجية للاستراتيجيات والتحديات والأدوات

م.م. الاء باسل محمد شريف
وزارة التربية – مديرية تربية نينوى – الكلية التربوية المفتوحة
E-mail: alaa.20csp10@student.uomosul.edu.iq
https://orcid.org/0009-0009-5772-6324

ملخص البحث:

مع تزايد تعقيد الأنظمة البرمجية وتسارع دورات التطوير من خلال منهجيات Agile وDevOps وCI/CD، أصبحت نماذج الأمن التقليدية رد الفعل غير كافية. تستعرض هذه المراجعة المنهجية نموذج نقل الأمن إلى اليسار (Shift-Left Security)، الذي يدمج الأنشطة الأمنية في المراحل المبكرة من دورة حياة تطوير البرمجيات (SDLC) بدلاً من معاملتها كمرحلة تحقق نهائية. من خلال تحليل منهجي لـ 28 دراسة محكمة من قواعد بيانات أكاديمية مرموقة، تُركِّز هذه الدراسة على استراتيجيات التنفيذ الحالية، والتحديات، والأدوات الداعمة، والفوائد المبلّغ عنها. صُنِّفت الدراسات المختارة إلى خمس فئات موضوعية: (1) اعتماد DevSecOps ودمج الذكاء الاصطناعي، (2) ممارسات نقل الأمن إلى اليسار، (3) التحديات التنظيمية والثقافية، (4) هندسة المتطلبات الأمنية، و(5) أدوات أتمتة الأمن. تكشف النتائج أن الدمج المبكر للأمن يقلل تكاليف المعالجة بشكل كبير ويحسّن جودة البرمجيات، في حين أن الاختبار الآلي داخل خطوط CI/CD يتيح التحقق المستمر دون الإخلال بسرعة التسليم. ومع ذلك، لا تزال هناك تحديات مستمرة، بما في ذلك المقاومة الثقافية بين فرق التطوير، وانتشار النتائج الإيجابية الكاذبة من الأدوات الآلية، ومحدودية الخبرة الأمنية بين المطورين، وغياب أطر تنفيذ موحدة. تؤكد المراجعة إلى أن اعتماد أمن التحول إلى اليسار بنجاح يتطلب مزيجا تآزريا من الأدوات الآلية، والتدريب المستمر للمطورين، ودعم القيادة التنفيذية، وتحول ثقافي جذري نحو المسؤولية الأمنية المشتركة. يجب أن تركز الأبحاث المستقبلية على أتمتة الأمن المدعومة بالذكاء الاصطناعي، وتوحيد أطر Shift-Left، والدراسات التجريبية التي تقيس العائد طويل الأمد على الاستثمار لممارسات DevSecOps.

الكلمات المفتاحية :نقل الأمن إلى اليسار، تطوير البرمجيات الآمنة، دورة حياة تطوير البرمجيات(SDLC) ، نمذجة التهديدات، البرمجة الآمنة، أتمتة اختبار الأمن، الأمن السيبراني.

1. Introduction

The rapid evolution of information technology is driving organizations and companies to develop custom software to support their operational activities. This process requires a structured methodology to ensure the production of high-quality software that meets user needs within the specified budget and timeframe. This is known as the Software Development Lifecycle (SDLC). The traditional SDLC consists of six essential, sequential, and integrated phases (Fiska & Pratama, 2026):

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

Requirements Analysis: which serves as the cornerstone of the project, Analysts collaborate with stakeholders to accurately identify and document system requirements in a Software Requirements Specification (SRS) (Quah et al., 2025).
Design: This entails creating the databases, user interfaces, and system architecture to serve as a blueprint prior to implementation and code development. (Davila-Campos et al., 2025).
Implementation/Coding: developers transform the approved software design into executable source code using suitable programming languages and development tools (Hossain, 2023).
Testing: This phase aims to run the system and identify viruses and vulnerabilities, to ensure the integrity of its results, and to evaluate security and performance to ensure compliance with quality standards (Mohapatra, 2025).
Deployment: This includes software deployment and installation to prepare for use in the target work environment (Olorunshola & Ogwueleka, 2021).
Maintenance: It is a continuous process of resolving bugs found after deployment, enhancing performance and upgrading the system to catch up with the rapid advancements and changing requirements (Anthony et al., 2020).
Strong security has become more challenging in today's software development landscape, which operates in an agile and fast-paced environment where speed and agility are paramount. The quick release cycles of today's software development processes are too much for the old security approach, which is frequently defined by isolated teams and reactive procedures. DevSecOps, a revolutionary methodology that smoothly incorporates security into each phase of the software development lifecycle, was born out of this gap (Rajapakse et al., 2021). It is an emerging approach that integrates extensive security into the DevOps software development paradigm (Manchana, 2024). DevSecOps considers security as one of the most important parts of the DevOps process. This results in quicker releases, more consistent implementation and security. It is based on principles of people-centric software development that allow for productive cooperation between software development and operations and security teams to achieve a shared objective (Rajapakse et al., 2022a). The DevSecOps philosophy revolves around the idea of "Shifting Left." It promotes including security procedures as early in the development process as feasible, ideally from the planning and design phases. By identifying and addressing security issues before they get ingrained in the codebase, this proactive method seeks to make remediation simpler and less expensive (Nicho et al., 2025).

1.1 Problem Statement

With the advent of Agile, DevOps and Continuous Integration/Continuous Delivery (CI/CD), software security is an emerging problem as software systems have been getting more complex and software development has been quickening. Although software development has become more secure, there is still a tendency to adopt the ‘reactive' attitude to software security by discovering vulnerabilities during testing or after deployment. These can result in increased remediation costs, delays and higher exposure to security threats.

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

To overcome these limitations, the concept of Shifting Security Left has also appeared, a proactive approach to embedding security activities into the early stages of the Software Development Lifecycle (SDLC). If vulnerabilities are discovered, automated testing, secure coding and continuous security validation can be a part of the development process to help minimize them from creating major problems.

However, shift-left security has not been a bed of roses, even as it grows in popularity. However, challenges such as integrating security into CI/CD pipelines, balancing speed and security, lack of security knowledge among developers, and the need for change in the security cultures and absence of common CI/CD security implementation frameworks persist in limiting its effectiveness. Moreover, the literature on the topic generally presents a variety of strategies, tools and good practices, with limited ability to provide a synthesis for systematic comparison of the effectiveness of the good practices, research gaps and implementation issues. This study aims at systematically reviewing the available literature on shift-left security in software development, analyzing strategies and challenges in implementing them, and the tools that support them and the reported benefits. Results will provide researchers and practitioners with a panoramic view of the state of the art, and allow them to define future research directions and how the practice of secure software development can be improved.

2. Literature Review

2.1 Evolution of Software Security Practices

The traditional software security methods are called Reactive Security or Late-Stage Security Testing Model, which can be summed up as a separate project to be applied towards the end of the Software Development Life Cycle (SDLC). Security assessments are usually performed at the end of the development process and just prior to deployment. This model was accepted generally, but has been heavily attacked because of structural and operational deformities (Macarthy & Bass, 2020).

One of the drawbacks is the security constraint, which means that penetration testing and vulnerability assessments are carried out after the entire system has been built. As a result, it can take some time for vulnerabilities to be discovered and for a significant amount of code to be modified, resulting in a bit of a tug of war between security and development teams to meet both deadlines and security needs (Teixeira et al., 2020). Moreover, remediation costs increase significantly when vulnerabilities are detected late. Fixing vulnerabilities at this point could involve architectural redesign, changing database definitions, massive regression testing, and validation of interdependent systems components, which will likely be significantly more costly than discovering vulnerabilities in previous stages of development (Rafi et al., 2020).

The adoption and application of traditional security methods further exacerbates the developer vs security dichotomy. Developers tend to be taught about functional requirements but not necessarily secure coding.When security is passed to specialised groups, developers may not learn secure coding skills. This separation undermines a culture of security and frequently means the security results are seen as criticism instead of a chance to enhance software quality and resilience (Makani, 2022).

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

Furthermore, the reactive security models aren't effective in an agile and DevOps-oriented software industry that requires continuous integration and continuous deployment to improve quick and frequent software releases. Because of these development cycles (Villamizar et al., 2018), traditional security processes, based on lengthy manual security reviews, are becoming much less practical. Moreover, traditional approaches provide only a limited insight into the software supply chain risks, since they capture only the internal software development risks and do not consider vulnerabilities associated with third-party libraries and open software sources. Organizations are also likely to opt for quick fixes rather than redesigning architecture, which provides the setting for later attacks (Nägele et al., 2022). The restrictions have resulted in the shift towards Shift-Left Security, an approach that embeds security in the SDLC process, rather than focusing it at the end of the development process (Malik, 2025).

Shift-Left Security is based on a number of fundamental ideas. The first one is proactive prevention, where one defines the security needs and eliminates vulnerabilities at the planning and design phase before they're implemented (Kaithe, 2025). The second principle is shared responsibility, which means that security is a shared responsibility among development, operations and security teams, following the DevSecOps concept. The method encourages software developers to be more secure regarding their coding methods and also to contribute to the software security (Malik & Prashasti, 2025).

Another key principle is continuous automation, achieved by integrating automated security tools such as Static Application Security Testing (SAST) and Software Composition Analysis (SCA) into Continuous Integration and Continuous Deployment (CI/CD) pipelines. These tools enable continuous assessment of source code and software dependencies without reducing development efficiency (Kaithe, 2025). Complementing this is continuous feedback which will alert developers to vulnerabilities, coding mistakes and compliance issues as they are being developed. This ongoing feedback increases security awareness, improves secure coding habits and diminishes the traditional knowledge gap between developers and security experts.

All of these principles work together to make security a part of the engineering process in software development, rather than a late-stage verification step, and to minimize remediation expenses—both operational and financial—by making software development more resilient. In the end, proactively, with shared responsibility, by automating, and with constant feedback, Shift-Left Security enables organizations to provide secure, reliable, and cyber-resilient software systems faster (Asprion et al., 2023).

2.2 Classification of Previous Studies

A careful analysis of the selected literature has been classified under five broad theme groups, which have emerged from the previous studies. The classification is useful to consider the methods, challenges, resources, and processes of implementing Shift-Left Security, and makes for a structured overview of the state of the art.

2.2.1 DevSecOps Adoption and AI Integration

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

This involves studying the integration of security into DevSecOps workflows and the potential of AI in enhancing security automation, vulnerability assessment, monitoring, and decision-making.

Table (1) Summary of Studies on DevSecOps Adoption and AI Integration

Study	Research Objective	Methodology	Key Findings
Rahman, A., & Williams, L. (2016)	To identify security practices used to integrate security into DevOps.	Analysis of 34 online artifacts and a survey of 9 DevOps organizations.	Found four key practices: non-automated security activities, teamwork, security training, and security automation. The study found that Automation and teamwork are essential to successful DevSecOps adoption.
Desai & Nisha (2021)	To identify best practices for secure DevOps implementation.	Literature review and multiple case studies involving three technology companies.	Early integration of security through Shift-Left practices reduces downtime and improves operational efficiency.
Prates & Pereira (2025)	To identify DevSecOps practices and tools reported in the literature.	Multivocal Literature Review (MLR) of 103 academic and industrial studies.	Identified 39 DevSecOps practices, with SAST, DAST, static code analysis, and continuous collaboration being the most widely adopted.
Binbeshr & Imam (2025)	To assess the effectiveness of AI techniques in DevSecOps environments.	Systematic Literature Review of 18 studies with comparative technical analysis.	AI enhances automation and anomaly detection; however, scalability and computational resource requirements remain key challenges.
Pimentel, Fonseca, & Carmona Cortes (2025)	To identify and analyze the security technologies and tools used in DevSecOps and provide a comprehensive knowledge base for researchers and practitioners.	Systematic Literature Review (SLR). Reviewed 87 studies published up to the first half of 2024, with 42 studies selected for detailed analysis based on three research questions	The review revealed a total of 13 categories of DevSecOps tools, with the most common supporting technology being Artificial Intelligence (AI). The majority of studies pointed to the same set of tools, with each study revealing a well-established and stable DevSecOps toolset as well as the

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

Study	Research Objective	Methodology	Key Findings
			potential for additional research on emerging technologies.
Fu et al. (2025)	To review AI techniques that support DevSecOps security automation.	Systematic review of 99 research papers.	Identified AI applications across 12 DevSecOps security tasks and highlighted future opportunities for intelligent security automation.

2.2.2 Shift-Left Security

This classification involves research on the merits of embedding security aspects at the initial stages of software development, particularly for proactive risk mitigation, early identification of vulnerabilities and periodic security validation.

Table (2) Summary of Studies on Shift-Left Security

Study	Research Objective	Methodology	Key Findings
Gonzalez, Peralta Perez, & Mirakhorli (2021)	To identify the barriers developers face when implementing Shift-Left Security through automated security unit and integration testing.	Mixed-method empirical study analyzing 525 Stack Overflow and Security Stack Exchange posts using Latent Dirichlet Allocation (LDA) and qualitative analysis.	The study revealed 7 major pain points in security unit and integration testing, primarily connected to authentication, security components, test configuration and security services. It makes the following recommendations to overcome this challenge: improve tools, guidance, and automated support to ensure successful adoption of Shift-Left Security.
Rajapakse et al. (2021)	To identify the challenges and solutions associated with adopting DevSecOps practices,	Systematic Literature Review of 54 peer-reviewed studies.	Based on the review, Shift-Left Security and continuous security assessment were identified as the best practices to enhance software security. The automation of tools and the cooperation of developers

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

Study	Research Objective	Methodology	Key Findings
	including Shift-Left Security.		were mentioned as key success factors.
Rajapakse et al. (2022b)	To investigate collaborative application security testing in DevSecOps environments.	Empirical qualitative study based on 48 practitioner webinars and technical discussions.	The study emphasized that integrating security testing early in development improves collaboration between development, security, and operations teams while enabling earlier vulnerability detection.
Seknametla & Sunkara (2024)	To examine the integration of threat modeling into DevSecOps pipelines using Shift-Left approaches.	Multi-organization empirical study conducted across ten organizations over 17 months.	Continuous threat modeling enabled organizations to identify architectural security risks much earlier in the SDLC and reduced design-level vulnerabilities discovered after deployment.
Kaithe (2025)	To examine the evolution of Shift-Left Security in modern software development.	Literature review supported by industry case studies.	The study concluded that early security integration improves development efficiency, reduces vulnerability remediation effort, enhances collaboration, and strengthens security throughout the SDLC.

2.2.3 Organizational Challenges

This category deals with the organizational, cultural and governance issues surrounding the adoption of security principles in enterprise software development projects, specifically in large-scale agile organizations.

Table (3) Summary of Studies on Organizational Challenges

Study	Research Objective	Methodology	Key Findings
Myrbakken & Colomo-Palacios (2017)	To provide a comprehensive overview of DevSecOps by defining the concept,	examining its benefits and challenges, and exploring its evolution.	DevSecOps makes security a core part of the DevOps lifecycle by bringing the development, operations, and security teams

The Peerian Journal

Open Access | Peer Reviewed

Volume 58 September 2026

ISSN (E): 2788-0303

Website: www.peerianjournal.com

Email: editor@peerianjournal.com

Study	Research Objective	Methodology	Key Findings
	identifying its characteristics, examining its benefits and challenges, and exploring its evolution.	Multivocal Literature Review (MLR) based on academic and grey literature (e.g., blogs, white papers, and industry reports). A total of 52 relevant sources published after 2014 were analyzed.	together. The study revealed that DevSecOps bolstered software quality, security, and automation, and mitigated risks and costs. But there are organizational challenges like cultural change, they have fewer security tools, and they need new skills and practices.
Macak et al. (2020)	To develop a game-based cybersecurity training platform that identifies attack vectors caused by unintentional insiders and enhances employees' security awareness through simulated workplace scenarios	A design and development study proposing a simulation-game platform. It tracks the employees' behavior in cybersecurity scenarios and uses process mining to analyzes the event logs and uncover possible attack vectors.	The proposed platform allows organizations to leverage process mining to detect potential security threats before they happen, conduct interactive security training, and understand how security issues stem from employees' unintentional actions.
Rajapakse et al. (2021)	To identify organizational and technical challenges in adopting DevSecOps.	Systematic Literature Review (54 studies).	People-related issues, lack of collaboration, security culture, and automation readiness were identified as major organizational barriers.
Nägele et al. (2023)	To understand how security is managed in Large-Scale Agile Development (LSAD) environments.	Semi-structured interviews with nine security and software development experts from various industries.	Organizations commonly employ shared security roles across teams. However, security governance often remains centralized, creating potential conflicts with agile team autonomy.
Thool & Brown (2024)	To evaluate the impact of security activities on team productivity and development speed.	Survey of 34 practitioners supported by open coding analysis.	Participants reported that security activities do not negatively affect team velocity

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

Study	Research Objective	Methodology	Key Findings
			when appropriately planned before sprint execution.
Schelehoff (2025)	To identify challenges encountered by agile teams when integrating security practices.	Systematic Literature Review involving 46 studies.	Cultural challenges, particularly insufficient security awareness, were found to be the most prevalent obstacles, followed by operational, technical, and regulatory compliance challenges.
Malik & Prashasti (2025)	Explore proactive security as organizational culture	Exploratory research and industry case studies	Security as culture; continuous awareness reduces production vulnerabilities

2.2.4 Security Requirements Engineering

This category is for frameworks, models, methodologies and requirements engineering approaches that can be used to integrate security requirements into agile software development processes.

Table (4) Summary of Studies on Security Requirements Engineering

Study	Research Objective	Methodology	Key Findings
Villamizar et al. (2018)	To characterize research trends concerning security requirements in agile software development.	Systematic Mapping Study of 21 studies.	Scrum emerged as the most frequently studied agile methodology. Research predominantly focuses on requirements elicitation and specification, while empirical validation studies remain limited.
Sharma & Bawa (2020)	To integrate traditional security activities into agile development lifecycles.	Literature review, survey of 97 professionals, and statistical analysis.	Identified 20 security activities compatible with agile development and proposed an algorithm for dynamically integrating security practices based on cost-benefit considerations.
Tashtoush et al. (2021)	To compare agile methodologies used in cybersecurity,	Systematic literature review and comparative analysis	Proposed a framework for selecting suitable agile methodologies according to project characteristics

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

Study	Research Objective	Methodology	Key Findings
	IoT, and intelligent transportation systems.	of six agile models based on twelve evaluation criteria.	and identified Scrum as the most widely adopted approach in cybersecurity projects.
Ardo et al. (2022)	To develop a taxonomy of agile security practices.	Practitioner interviews analyzed using Grounded Theory.	Discovered 26 security practices including four security roles, twelve ceremonies and ten security artifacts. Results underscored the use of automated security methods over manual testing strategies.
Handri et al. (2024)	To develop an agile cybersecurity framework integrating technological and organizational dimensions.	Literature review followed by Q Methodology analysis.	Proposed a framework combining DSDM and FDD methodologies as an alternative to Scrum for security-sensitive environments requiring a balance between agility and governance.

2.2.5 Security Automation Tools for Shift-Left Security

This category includes research projects that explore the adoption of automated security testing tools in DevSecOps and Continuous Integration/Continuous Deployment (CI/CD) pipelines. These studies analyze Static Application Security Testing (SAST), Dynamic Application Security Testing (DAST), Interactive Application Security Testing (IAST), and Software Composition Analysis (SCA) to understand how each of them can help in implementing Shift-Left Security and enhance the effectiveness of vulnerability detection during the Software Development Life Cycle (SDLC).

Table (5) Summary of Studies on Security Automation Tools for Shift-Left Security

Study	Research Objective	Methodology	Testing Tools (SAST, DAST, IAST)	Security Automation in CI/CD and DevSecOps Tools
Singh (2020)	To explore the seamless integration of security practices into DevOps	Literature review and analysis of security automation practices and tools within	SAST: SonarQube, Checkmarx, and Fortify for source code analysis. DAST: Burp Suite and Acunetix for	The study focused on sustained security automation across CI/CD processes, and underscored the need to connect automated

The Peerian Journal

Open Access | Peer Reviewed

Volume 58 September 2026

ISSN (E): 2788-0303

Website: www.peerianjournal.com

Email: editor@peerianjournal.com

Study	Research Objective	Methodology	Testing Tools (SAST, DAST, IAST)	Security Automation in CI/CD and DevSecOps Tools
	pipelines through Shift-Left Security principles.	DevSecOps environments.	runtime application scanning. IAST: It is used for monitoring the behaviour of the application and to detect suspicious transactions instantly while the application is running.	security testing tools with development processes to ensure proactive detection and remediation of vulnerabilities.
Rajapakse et al. (2021)	To investigate how security testing tools can be integrated into DevOps workflows without negatively affecting deployment speed.	Thematic analysis of 31 industry webinars involving experts discussing security tool integration in DevOps environments.	SAST: Implemented at three levels: instantaneous feedback in IDEs, incremental scanning at commit time and deep analysis during build time. DAST: Conducted concurrently in the testing environments. IAST: Proposed as a hybrid approach combining SAST and DAST capabilities to reduce false positives.	Security automation was integrated across IDEs, source code management systems, build servers, and testing environments. Additional technologies included Software Composition Analysis (SCA), Runtime Application Self-Protection (RASP), Web Application Firewalls (WAF), and Security Information and Event Management (SIEM) systems.
Raza et al. (2023)	To implement and evaluate automated	Case study implementing an automated	SAST: Snyk Code. DAST: StackHawk. SCA: Snyk Open	Automated security scanning was integrated into the GitHub CI/CD

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

Study	Research Objective	Methodology	Testing Tools (SAST, DAST, IAST)	Security Automation in CI/CD and DevSecOps Tools
	security scanning within a DevSecOps CI/CD pipeline.	security pipeline using GitHub and containerized applications, followed by performance evaluation.	Source. (The study does not implement IAST.)	pipeline. Security checks were executed during every build, vulnerabilities were detected automatically, and remediation was accelerated through continuous scanning and automated reporting.
Feio et al. (2024)	To evaluate a practical DevSecOps framework focused on continuous security testing within a real-world software project.	Case study of the GRACE microservices project utilizing 15 different security tools.	SAST: SonarLint for dev, SonarQube for build server analysis. OWASP ZAP and OWASP Wapiti vulnerability scanners for Web Apps. IAST: Highly effective for automation, but not yet mature and open-source solutions.	Full integration with Jenkins provided automatic security scans in every build process. A further use of the framework was to provide support for SCA with Dependency-Check to detect vulnerabilities in third party libraries and dependencies.
Zhao et al. (2024)	To identify the main dimensions, practices, tools, and metrics of DevSecOps.	Multivocal Literature Review.	SAST, DAST, SCA, Continuous Monitoring.	Proposed a Challenge–Practice–Tool–Metric model and highlighted automation of security testing as a core DevSecOps practice.

2.3 Comparing the traditional approach to security with the modern approach (Shift-Left Security).

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

Traditional security view security as a final phase of SDLC, which is based on penetration testing and manual testing at the end of SDLC (Rafi et al., 2020). This reactive approach will only detect vulnerabilities when the architectural decisions are completed, which is very expensive and disruptive (Rajapakse et al., 2022; Nicho et al., 2025).

Shift-Left Security, on the other hand, integrates security requirements, threat modelling, secure coding, and automated testing into early development stages.Shift-Left Security, however, incorporates security requirements, threat modelling, secure coding and automated testing at the earliest stages of development (Manchana, 2024). It brings a new model of shared responsibility, where developers, operations and security experts work side by side and continuously (Teixeira et al., 2020; Nägele et al., 2022). The shift-left approach, in addition, enables automated security evaluation without slowing down releases, while maintaining the Agile and DevOps pace, unlike traditional approaches that are at odds with this rapid deployment time (Rajapakse et al., 2022). In conclusion, this forward-looking approach can help lower remediation expenses, boost software quality, and reinforce cyber resilience (Kaithe, 2025). The table below contrasts Traditional Security with Shift-Left Security.

Table (6) Comparison Between Traditional Security and Shift-Left Security

Aspect	Traditional Security	Shift-Left Security
Security Timing	Late stages of SDLC	Early and continuous throughout SDLC
Security Approach	Reactive	Proactive
Vulnerability Detection	After development completion	During development
Remediation Cost	High	Lower
Team Responsibility	Security team only	Shared responsibility (DevSecOps)
Testing Method	Manual and periodic	Automated and continuous
Compatibility with Agile/DevOps	Limited	High
Feedback Cycle	Delayed	Immediate and continuous
Development Impact	Creates bottlenecks	Supports continuous delivery
Security Culture	Isolated security function	Organization-wide security culture

3. Methodology

3.1 Database Selection and Search Strategy

This study used a structured literature review method to identify, synthesize, analyze and review literature on Shift-Left Security and DevSecOps practices. Based on the great number of popular publications that are spread across different regions of the world, the research was conducted comprehensively in few reputable digital libraries and academic databases, which would span the entire

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

academic world from all over the world. The selected databases included: Scopus, IEEE Xplore, ACM Digital Library, SpringerLink, ScienceDirect, Wiley Online Library and Google Scholar. The selection of these databases was based on their access to a broad range of peer-reviewed journal articles, conference papers and systematic literature reviews and industrial studies in software engineering, cyber security and DevSecOps.

This literature search was focused on literature covering integrating security into Software Development Processes (SDPs), DevSecOps adoption, security and Shift-Left Security automation. First priority was given to peer reviewed publications, however when the industrial reports and multi-vocal literature reviews could give useful practical knowledge, they were incorporated.

3.2 Search String and Refinement

The search process was carried out with combinations of keywords linked with Boolean operators to maximize the accuracy of the retrieval. The main keywords used were:

- "Shift-Left Security"
- "DevSecOps"
- "Secure Software Development"
- "Security Automation"
- "Security Testing"
- "Static Application Security Testing (SAST)"
- "Dynamic Application Security Testing (DAST)"
- "Interactive Application Security Testing (IAST)"
- "Software Composition Analysis (SCA)"
- "Continuous Integration"
- "Continuous Deployment"
- "CI/CD"
- "Artificial Intelligence"
- "Threat Modeling"
- "Security Requirements Engineering"

An example of the search query used is presented below:

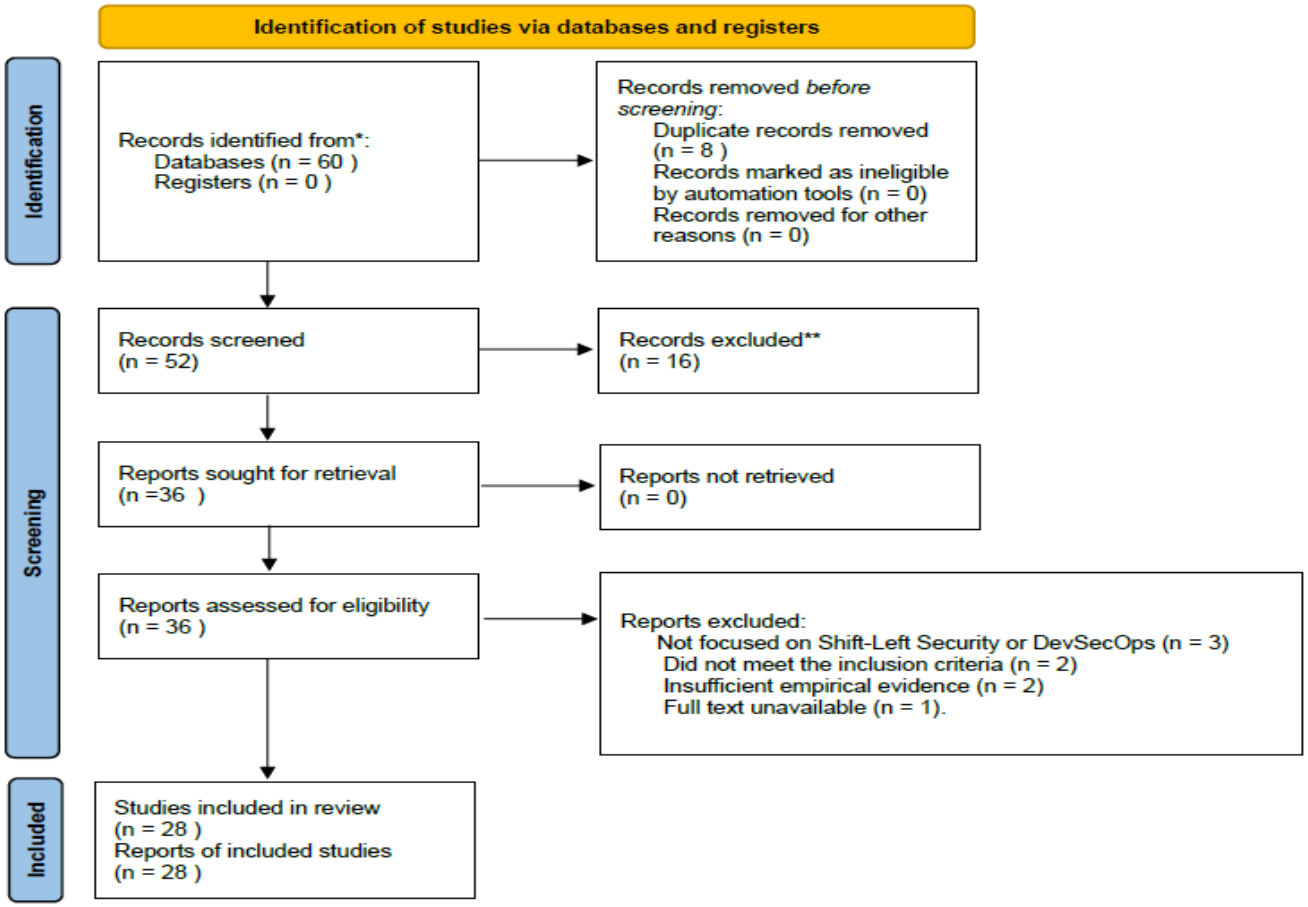
("Shift-Left Security" OR "DevSecOps") AND ("Security Automation" OR SAST OR DAST OR IAST OR SCA) AND ("CI/CD" OR "Continuous Integration" OR "Continuous Deployment")

The retrieved publications were subjected to several stages of filtering. Duplicate records were sorted out and then the titles and abstracts were screened to exclude studies beyond the scope of the review. The remaining papers were then subjected to full text analysis to identify their relevance to the goal of this study.

Study selection was done in compliance with the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) 2020 guidelines to ensure a transparent and systematic study selection process. The 60 records were found by searching databases in the first place. Duplicate records were eliminated and the remaining studies were categorised using title, abstract and full text screening with the predefined inclusion and exclusion criteria. Ultimately, 28 studies met the eligibility criteria and

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

were included in the final systematic literature review. The detailed PRISMA 2020 flow diagram of the selection process of the studies is presented in Figure 1. (Page et al., 2021).



3.3 Inclusion Criteria

The following criteria were used to include studies:

- The study is on the topic of DevSecOps, Shift-Left Security, secure software development, or security automation.
- It is a publication that contains empirical evidence, systematic reviews, mapping studies, case studies, industrial experiences or methodological frameworks.
- The publication is in English language.
- Study must be published in a peer-reviewed journal, conference proceeding or a well-recognized academic source.

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

- The publication offers adequate methodological information and results are presented in a clear manner.
- The study directly supports one or more of the research themes explored in this review.

Studies were excluded if they:

A cybersecurity solution limited to the traditional approach and not aligned to the software development lifecycle. Did editorials, tutorials, presentation slides, posters or opinion papers lack a research methodology? Lacked technical and/or methodological information. Did duplicate studies of previously published articles. After the screening procedure, 28 studies met the inclusion criteria and were included in a detailed analysis.

3.4 Study Selection

The study selection process consisted of four sequential stages. Relevant publications were identified by the predefined search strings in a search of databases (first). Second, duplicate records were eliminated. Third, titles and abstracts were screened to determine the relevance to the research goals. Lastly, articles with full text were carefully screened to see if they met the inclusion and exclusion criteria

The final cohort of 28 studies were grouped into five thematic areas based on the main focus of their research:

1. DevSecOps Adoption and Artificial Intelligence Integration.
2. Shift-Left Security.
3. Organizational Challenges.
4. Security Requirements Engineering.
5. Security Automation Tools for Shift-Left Security.

This thematic classification allowed for a systematic comparison between studies and helped identify current trends and existing practices, research gaps and possible directions in the future.

3.5 Quality Assessment

A quality assessment process of each study was carried out and the following measures were used to assess the reliability of the selected literature:

- Appropriateness to the research objectives.
- Research Methodology is clear.
- Appropriateness of the data gathering and analysis techniques.
- Results of reported findings that are valid and credible.
- Support of Shift-Left Security and DevSecOps understanding.
- Publication quality and peer review status.
- Research with sound methodology and clear reporting and relevance to the topic was given priority in the synthesis process.

3.6 Data Extraction Strategy

A structured data extraction approach was adopted to ensure consistency across all selected studies. For each publication, the following information was extracted:

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

- Authors and publication year.
- Research objective.
- Research methodology.
- Application domain.
- Security practices or technologies investigated.
- Tools and techniques employed.
- Major findings.
- Reported challenges and limitations.
- Contribution to Shift-Left Security or DevSecOps implementation.

The obtained data were clearly tabulated and presented side by side to facilitate comparative analysis and thematization of the data. The process helped to identify common patterns, emerging technologies, trends in the existing literature and gaps in knowledge about Shift-Left Security and DevSecOps.

4. Advantages and disadvantages of Shift-Left Security

In recent software engineering, Shift-Left Security has become a popular solution to integrate security problems in the initial stages of the Software Development Life Cycle (SDLC). Security activities get closer to the requirements, the design and the implementation phase, which helps organizations discover and fix any potential vulnerabilities before they become well-entrenched in software systems. Although Shift-Left Security offers a number of advantages, there are some issues that organizations need to address to ensure that Shift-Left Security can be successfully implemented (Malik & Prashasti ,2025), (Kaithe, 2025).

4.1 Advantages of Shift-Left Security

The ability to detect and address vulnerabilities early is one of the key advantages of the "Shift-Left Security" approach in modern software engineering. Research confirms that security flaws identified in early stages, such as requirements analysis, system design, or programming, are easier and significantly less costly to fix (by a factor of up to 100) compared to vulnerabilities discovered during testing or after the actual product deployment. This proactive intervention also reduces the complexity of software fixes and significantly lowers the risk of security incidents in production environments, thereby enhancing confidence in the final product (Dachepally ,2025). Another important benefit of Shift-Left Security is the substantial decrease in both development and maintenance expenditures. Research within software engineering confirms that the expense associated with remediating defects escalates exponentially as the project moves from the design stage toward post-deployment maintenance. Eliminating security issues at their source enables organizations to avoid the time and expense of the frequent large-scale testing and emergency hotfixes and architectural redesigns. More than that, it is a proactive approach that enhances software quality and system reliability by incorporating security aspects from the beginning of the development process. In turn, security becomes ever more integral part of the quality assurance process rather than an independent or stand-alone quality assurance test at the end of the pipeline (Kaithe, B. K., 2025). Consequently, software applications are more resilient to cyber threats and closer to conforming to the existing security rules. Moreover, it fosters ongoing cooperation and responsibility between developers, ops, and security teams. It is a key component of

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

DevSecOps and breaks down silos within organizations and contributes to building security awareness across development teams (Manchana, 2024). One of the main benefits of the “left-shifted security” approach is that it is naturally compatible with the Agile and DevOps frameworks. Using automated security testing tools in CI/CD production lines allows for ongoing security testing without slowing down development cycles. By allowing the technical integration, organizations can strike a balance between speed of software delivery and effective security governance without compromising their security and inhibiting their ability to innovate (Ardo et al., 2021). Furthermore, with this proactive approach, the efforts towards security risks and regulatory compliance are also enhanced by ensuring that security needs are systematically and extensively integrated into the Software Development Lifecycle (SDLC). Besides, 24/7 security audit processes help organizations meet best practices and legal obligations efficiently since they provide automated and auditable proof of the integrity of security processes (Handri et al., 2024).

4.2 Disadvantages and Challenges of Shift-Left Security

While there are benefits to Shift Left Security, it does not come without its drawbacks. One of the challenges is some particular experience in security is needed in the developers. Implementing them successfully demands that the developer understands secure coding, threat modelling and security testing techniques. An organisation may, at that point, have to spend quite a bit of money upon preparing and talent creation. The other challenge is the cost of initial implementation and organisational change. The introduction of security tools into the development process, process redesign, and new security practices often involve significant investments in technology, people and process change. With the greater implementation of automated security testing, there could be false positives, which would be security tools that incorrectly detect a vulnerability that is not a threat. Too many alerts can cause developers to become "alert fatigued" and diminish trust in automated security systems. Additionally, the inclusion of several security instruments within complicated development environments can add operational complexity. It's important to plan for and maintain the security scanners and dependency analysis tools, as well as compliance checks and CI/CD tool integrations to ensure they do not interfere with development workflows. Early security testing has another constraint that it does not ensure the complete removal of vulnerabilities (Manchana, 2024). Some problems in security can only be identified at runtime, under certain conditions, or by using some complex attack techniques (Feio et al. 2024). The "Shift-Left Security" strategy is not an alternative to traditional security assessments, but rather an approach that is used in conjunction with traditional assessments so that early activities can be combined with activities like penetration testing and continuous monitoring in production environments to ensure comprehensive security coverage. But, cultural barriers can be a major obstacle in transitioning to this shift and research indicates that the most frequent and frequent barrier to DevSecOps transitions is cultural resistance. Traditional teams are typically known to think that security and its related operations are “bottlenecks” or an added burden in a software delivery system and that is slowing down the speed of release. Strategically, it is important to create a culture in the organisation based on security, whereby all teams will be responsible, and to have high level commitment from the senior leaders who would resist and accept that responsibility

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

(Feio et al. 2024). Finally, Shift-Left Security benefits greatly in terms of the early detection of vulnerabilities, cost reduction, software quality and modern program development. However, its implementation should go along with organization commitment, adequate tools, competent staff, and organization of shared responsibility for security (Handri et al., 2024).

5. Discussion and Future Outlook

This detailed review reveals a shifting (but still divided) landscape of implementation of shift left security. An analysis of 28 studies reveals good agreement that early incorporation of security activities in the SDLC results in concrete paybacks on software QA, cost savings and vulnerability mitigation. The Shift-Left is economically well motivated as the costs of patching vulnerabilities at a point in time after deployment can be 100 times what it costs to do so at design time because of the exponential escalation of costs. In the Agile and DevOps culture, automated security testing can be a key component of the CI/CD pipeline, ensuring that security is always considered.

5.1 Synthesis of Thematic Findings

Five thematic classes enable the identification of research streams, which are different, but also overlapping. Research on DevSecOps adoption and the shift towards Artificial Intelligence shows that although AI is widely recognized as a promising enabler of security automation, anomaly detection, and intelligent decision-making, its actual application is hindered by scalability concerns, computational requirements, and the availability and quality of training data. As AI becomes the most common supporting technology in DevSecOps tool ecosystems, there is still a way to go before it is fully developed in terms of standardized AI-driven security frameworks.

All the articles in the Shift-Left Security category are focused on proactive security measures: threat modelling, secure coding and early detection of vulnerabilities. But, the empirical research by Gonzalez et al. (2021) and Rajapakse et al. (2021, 2022b) have identified a number of tensions, including the “pain points,” developers experience in implementing automated security unit testing. The results point to the need for a significant improvement in developer experience and usability of security tools, in addition to the use of tooling.

5.2 Organizational and Cultural Dimensions

Of all the barriers that have been pointed out in several of the studies, the one that might be the most important is not technical, but cultural. The analysis of organizational challenges shows that the main challenges are cultural resistance, lack of security awareness and weak accountability models. This corroborates the findings of Schelehoff (2025) and Malik & Prashasti (2025), which reveal cultural barriers as the topmost challenges, outpacing technical and regulatory challenges. The shift from narrow security forces to a shared-responsibility approach has completely turned the tide on security organizations and needs continuous executive sponsorship. As mentioned by Nägele et al. (2023), there is no resolution on the conflict between the centralized security governance and agile team autonomy in large-scale agile environments.

The Peerian Journal

Open Access | Peer Reviewed

Volume 58 September 2026

ISSN (E): 2788-0303

Website: www.peerianjournal.com

Email: editor@peerianjournal.com

5.3 Tooling and Automation Maturity

The analysis of security automation tools shows that the security automation tools and security automation for security vulnerabilities (SAST and SCA) are well developed and adopted tools, including SonarQube, Checkmarx, Snyk, and OWASP Dependency-Check, have been in use for long. DAST and especially IAST are not as integrated into the standard CI/CD flows, often because they are more complex to operate, take longer to run, and lack of mature open-source IAST solutions. The high number of false-positive alerts from SAST tools is causing "alert fatigue," which is detrimental to developer trust and may be missing the critical vulnerabilities in the code.

5.4 Research Gaps and Limitations

There are a number of important gaps, even with the new literature. However, there is a clear lack of consistent frameworks and architectures available that organizations can follow to systematically integrate Shift-Left Security in a wide range of technology stacks. Second, there are few studies that have adopted a longitudinal design to measure ROI of DevSecOps practices and the majority of the existing studies are based on a cross-sectional design or qualitative interviews. Third, the intersection of AI-generated code (such as GitHub Copilot, ChatGPT) and Shift-Left Security is almost untapped and has not been examined to determine how automated security testing applies to machine-generated code, which may have new types of vulnerabilities. Fourth, there is a lack of studies for Small and Medium Enterprises (SMEs) and resource-constrained development contexts, while studies focus mainly on enterprise and large scale environments.

5.5 Future Research Directions

The following are areas that should be considered for future research:

1. **The standardisation of Shift-Left Frameworks:** the creation of domain-specific reference architectures and maturity models for the implementation of Shift-Left Security in various organisational contexts with prescriptive guidance.
2. **AI-Augmented Security Automation:** Explore how machine learning is used to minimize false positives in SAST/DAST tools, prioritize vulnerabilities intelligently and automate security patching in CI/CD pipelines.
3. **Developer-Centric Security Tooling:** This involved human-computer interaction research aimed at making security tools easier to use, reducing the cognitive load, and embedding security feedback into integrated development environments (IDEs).
4. **Economic and Longitudinal Impact Studies:** Quantitative longitudinal studies that capture the total cost of ownership, productivity impacts, and security posture gains with extended DevSecOps adoption durations.
5. **Supply Chain Security Integration:** Extending the Shift-Left principles to software supply chains, including automated generation of the Software Bill of Materials (SBOM) and regular monitoring of third-party software and open-source code dependencies, and the containerized environment.

The Peerian Journal
Open Access | Peer Reviewed
Volume 58 September 2026
ISSN (E): 2788-0303
Website: www.peerianjournal.com
Email: editor@peerianjournal.com

6. **6. Cultural Transformation Metrics:** Creating standardized tools to quantify and assess security culture, shared responsibility adoption, and organizational readiness to implement DevSecOps transitions.

6. Conclusion

This systematic review aims to take an overall picture of Shift-Left Security in the software development and has summarized the current status of 28 studies grouped into five thematic domains. The evidence is fairly clear that, in fact, shifting security activities to earlier stages of the SDLC is a paradigm shift from best responding to “test once” in the late stages of the SDLC, to "engineering security" all the time. Security testing tools like SAST, DAST, IAST and SCA can be integrated into CI/CD pipelines, enabling organizations to continue development without disruption and ensure they identify and remediate security vulnerabilities in a systematic and proactive way.

As a result of these findings, Shift-Left Security is showing great benefit in metrics including reduced remediation costs, improved software quality and reliability, greater software regulatory compliance and enhanced integration with Agile and DevOps methods and tools. DevSecOps becomes “shared responsibility,” eliminating silos between Dev, Ops and Sec.

It also points out persistent challenges to widespread adoption of these technology power races. However, culture and security awareness among the development teams are the strongest barriers to cross-cultural collaboration, and this involves strong commitment from the leaders and continued organizational learning. Multiple implementation of security solutions, false positives and less security knowledge among developer heighten implementation difficulties. In addition, there is no standard literature available about the long-term economic outcomes, and very little empirical evidence do exist regarding long-term economic outcomes.

Practitioners should be aware that it is important that they pursue a holistic approach which embraces both technology and human support – such as training developers, selecting appropriate technology that is accessible to users, fostering collaboration between departments and securing leadership’s assistance. The study provides interesting possibilities for future research focused on AI in security automation, architectural standardized approaches to Shift-Left and more long-term research on the impact of DevSecOps practices.

Shift-Left Security is not only a different way to approach software security, thinking, building and maintaining it, but it's a different way to approach software. There needs to be a synergic gathering of automated tools, transformation of culture and ongoing learning to create a cyber resilient software system that can adapt to the evolving threatscape.

The Peerian Journal

Open Access | Peer Reviewed

Volume 58 September 2026

ISSN (E): 2788-0303

Website: www.peerianjournal.com

Email: editor@peerianjournal.com

Reference

- Anthony, A. R. V., Prasad, G. D., Randunuge, S. U., Alahakoon, S. R., Wijendra, D. R., & Krishara, J. (2020). *Software development automation: An approach to automate the processes of SDLC. International Journal of Computer Applications, 175*(37), 44–51.
- Ardo, A. A., Bass, J. M., & Gaber, T. (2021, December). An empirical investigation of agile information systems development for cybersecurity. In *European, Mediterranean, and Middle Eastern Conference on Information Systems* (pp. 567–581). Springer.
- Asprion, P. M., Giovanoli, C., Scherb, C., & Bhat, S. (2023). *Agile management in cybersecurity. Proceedings of Society, 93*, 21–32.
- Binbeshr, F., & Imam, M. (2025, June). Comparative analysis of AI-driven security approaches in DevSecOps: Challenges, solutions, and future directions. In *Proceedings of the 29th International Conference on Evaluation and Assessment in Software Engineering Companion* (pp. 142–151).
- Dachepally, R. (2025). Devsecops: Shifting security left with automated scanning tools. *IJAIDR-Journal of Advances in Developmental Research*, *16*(1).
- Davila-Campos, R., Mora, M., Reyes-Delgado, P. Y., Galván-Cruz, S., & Lopez-Torres, G. C. (2025). *Design and evaluation of AgileBPM SDLC: An agile SDLC for business process management systems. IEEE Access.*
- Desai, R., & Nisha, T. N. (2021, July). Best practices for ensuring security in DevOps: A case study approach. In *Journal of Physics: Conference Series* (Vol. 1964, No. 4, Article 042045). IOP Publishing.
- Feio, C., Santos, N., Escravana, N., & Pacheco, B. (2024, July). An empirical study of DevSecOps focused on continuous security testing. In *2024 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)* (pp. 610–617). IEEE.
- Fiska, R. R., & Pratama, M. A. (2026). *Analysis of the simple additive weighting and similarity methods in decision support systems for SDLC method selection. Sistemasi: Jurnal Sistem Informasi, 15*(4), 1539–1547.
- Fu, M., Pasuksmit, J., & Tantithamthavorn, C. (2025). *AI for DevSecOps: A landscape and future opportunities. ACM Transactions on Software Engineering and Methodology.*

The Peerian Journal

Open Access | Peer Reviewed

Volume 58 September 2026

ISSN (E): 2788-0303

Website: www.peerianjournal.com

Email: editor@peerianjournal.com

- Gonzalez, D., Perez, P. P., & Mirakhorli, M. (2021, October). Barriers to shift-left security: The unique pain points of writing automated tests involving security controls. In *Proceedings of the 15th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)* (pp. 1–12).
- Handri, E. Y., Sensuse, D. I., & Tarigan, A. (2024). *Developing an agile cybersecurity framework with organizational culture approach using Q methodology*. *IEEE Access, 12*, 108835–108850.
- Hossain, M. I. (2023). *Software development life cycle (SDLC) methodologies for information systems project management*. *International Journal for Multidisciplinary Research, 5*(5), 1–36.
- Kaithe, B. K. (2025). *Shift left security: A paradigm shift in software development security integration*. *European Journal of Computer Science and Information Technology, 13*(24), 96–102.
- Macak, M., Kruzikova, A., Daubner, L., & Buhnova, B. (2020, June). Simulation games platform for unintentional perpetrator attack vector identification. In *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops* (pp. 222–229).
- Macarthy, R. W., & Bass, J. M. (2020, August). An empirical taxonomy of DevOps in practice. In *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)* (pp. 221–228). IEEE.
- Makani, S. T. (2022). *DevOps security tools evaluating effectiveness in detecting and fixing security holes*. *DevOps—An Open Access Journal, 1*(1), 18–21.
- Malik, G., & Prashasti. (2025). *Shift Left Security*. *The Eastasouth Journal of Information System and Computer Science, 2*(3), 219–245.
- Manchana, R. (2024). *DevSecOps in cloud native cybersecurity: Shifting left for early security, securing right with continuous protection*. *International Journal of Science and Research, 13*(8), 1374–1382.
- Marandi, M., Bertia, A., & Silas, S. (2023, July). Implementing and automating security scanning to a DevSecOps CI/CD pipeline. In *2023 World Conference on Communication & Computing (WCONF)* (pp. 1–6). IEEE.
- Mohapatra, P. (2025). *The evolution of software testing: Manual testing, verification methods, and SDLC integration in software engineering*. *Intelligent Assurance: Artificial Intelligence-Powered Software Testing in the Modern Development Lifecycle, 4*, 1.
- Myrbakken, H., & Colomo-Palacios, R. (2017, September). DevSecOps: A multivocal literature review. In *International Conference on Software Process Improvement and Capability Determination* (pp. 17–29). Springer.
- Nägele, S., Schenk, N., & Matthes, F. (2023, June). The current state of security governance and compliance in large-scale agile development: A systematic literature review and interview study. In *2023 IEEE 25th Conference on Business Informatics (CBI)* (pp. 1–10). IEEE.
- Nägele, S., Watzelt, J. P., & Matthes, F. (2022, June). Investigating the current state of security in large-scale agile development. In *International Conference on Agile Software Development* (pp. 203–219). Springer.

The Peerian Journal

Open Access | Peer Reviewed

Volume 58 September 2026

ISSN (E): 2788-0303

Website: www.peerianjournal.com

Email: editor@peerianjournal.com

- Nicho, M., Effiong, I., & McDermott, C. D. (2025, April). Shift-left security: Integrating security in the initial phase of the DevOps methodology. In *2025 9th International Conference on Cryptography, Security and Privacy (CSP)* (pp. 182–191). IEEE.
- Olorunshola, O. E., & Ogwueleka, F. N. (2021). Review of system development life cycle (SDLC) models for effective application delivery. In *Information and Communication Technology for Competitive Strategies (ICTCS 2020): ICT Applications and Social Interfaces* (pp. 281–289). Springer.
- Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., ... & Moher, D. (2021). *The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. BMJ, 372*, n71. https://doi.org/10.1136/bmj.n71
- Pimentel, F. C. S., Fonseca, L. C. C., & Cortes, O. A. C. (2025). *Tecnologias e Ferramentas Utilizadas em DevSecOps: Uma revisão sistemática da literatura. iSys – Brazilian Journal of Information Systems, 18*(1).
- Prates, L., & Pereira, R. (2025). *DevSecOps practices and tools. International Journal of Information Security, 24*(1), 11.
- Quah, J. K., Chiam, Y. K., & Sari, N. A. M. (2025). *Personalized explainability requirements analysis framework for AI-enabled systems. Malaysian Journal of Computer Science, 38*(1), 55–80.
- Rafi, S., Yu, W., & Akbar, M. A. (2020, April). Towards a hypothetical framework to secure DevOps adoption: Grounded theory approach. In *Proceedings of the 24th International Conference on Evaluation and Assessment in Software Engineering* (pp. 457–462).
- Rajapakse, R. N., Zahedi, M., & Babar, M. A. (2021, October). An empirical analysis of practitioners' perspectives on security tool integration into DevOps. In *Proceedings of the 15th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)* (pp. 1–12).
- Rajapakse, R. N., Zahedi, M., & Babar, M. A. (2022a). *Collaborative application security testing for DevSecOps: An empirical analysis of challenges, best practices and tool support. arXiv.* https://arxiv.org/abs/2211.06953
- Rajapakse, R. N., Zahedi, M., Babar, M. A., & Shen, H. (2022b). *Challenges and solutions when adopting DevSecOps: A systematic review. Information and Software Technology, 141*, 106700.
- Schelehoff, N. (2025). *Securing agile development: Analysing the intersection of cybersecurity and agile methodologies—A systematic literature review.*
- Seknametla, P. R., & Sunkara, R. (2024). *Threat modeling integration in DevSecOps pipelines: Early-stage security risk identification using Shift-Left approaches. International Journal of Emerging Research in Engineering and Technology, 5*(1), 126–133. https://doi.org/10.63282/3050-922X.IJERET-V5I1P115
- Sharma, A., & Bawa, R. K. (2022). *Identification and integration of security activities for secure agile development. International Journal of Information Technology, 14*(2), 1117–1130.

The Peerian Journal

Open Access | Peer Reviewed

Volume 58 September 2026

ISSN (E): 2788-0303

Website: www.peerianjournal.com

Email: editor@peerianjournal.com

- Singh, B. (2020). *Shifting security left: Integrating DevSecOps into agile software development lifecycles*. *The Research Journal (TRJ)*.
- Tashtoush, Y. M., Darweesh, D. A., Husari, G., Darwish, O. A., Darwish, Y., Issa, L. B., & Ashqar, H. I. (2021). *Agile approaches for cybersecurity systems, IoT and intelligent transportation*. *IEEE Access, 10*, 1360–1375.
- Teixeira, D., Pereira, R., Henriques, T. A., Silva, M., & Faustino, J. (2020). *A systematic literature review on DevOps capabilities and areas*. *International Journal of Human Capital and Information Technology Professionals, 11*(3), 1–22.
- Thool, A., & Brown, C. (2024, June). Securing agile: Assessing the impact of security activities on agile development. In *Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering* (pp. 668–678).
- Ur Rahman, A. A., & Williams, L. (2016, April). Security practices in DevOps. In *Proceedings of the Symposium and Bootcamp on the Science of Security* (pp. 109–111).
- Villamizar, H., Kalinowski, M., Viana, M., & Fernández, D. M. (2018, August). A systematic mapping study on security in agile requirements engineering. In *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)* (pp. 454–461). IEEE.
- Wu, X., Li, Z., Hu, F., Lin, T., Wang, R., Zhao, X., & Guo, X. (2026). *Shift-left techniques in electronic design automation: A survey*. *ACM Computing Surveys, 58*(13), 1–35.