Policy Research Journal
ISSN (E): 3006-7030 ISSN (P) : 3006-7022 Volume 4, Issue 3, 2026

SECURE SOFTWARE ENGINEERING: EMBEDDING CYBERSECURITY REQUIREMENTS THROUGHOUT THE DEVELOPMENT LIFECYCLE

Ume Aksa[1], Nasir Umar[*2], Zainab Naveed[3], Iram Shafique[4], Naeem Aslam[5], Jamil Ullah[6]

[1,*2,3,4,5]*NFC Institute of Engineering and Technology Multan, Pakistan*
[6]*Department of Computer Science, NCBA&E, Multan, Pakistan*

[1]umeaksa@gmail.com, [2]nasir.umar@nfciet.edu.pk, [3]zainabnaveed4554@gmail.com,
[4]shafiqiram57@gmail.com, [6]jamilullah@gmail.com

DOI: https://doi.org/10.5281/zenodo.19246859

Keywords
Secure Software Development Lifecycle (SSDLC), Cybersecurity Requirements Engineering, Secure Coding Practices, Security Testing and Validation, Security Governance

Article History
Received: 26 January 2026
Accepted: 09 March 2026
Published: 27 March 2026

Corresponding Author: *
Nasir Umar

Abstract
As cyber threats evolve quickly, modern software engineering requires a security-focused strategy that covers the entire project and development lifecycle. This research looks at how incorporating cybersecurity requirements from the earliest planning stages through design, coding, testing, deployment, and ongoing maintenance improves the overall security of software systems. By blending secure-by-design principles with modern project management methods, the study emphasizes that security should be a continuous, fundamental part of the process rather than just a technical task. Using guidance from well-known standards and frameworks, such as NIST SSDF, OWASP, ISO 27001, CMMI, SSE-CMM, Microsoft SDL, TSP-Secure, SAMM, and other maturity-model-based approaches, the analysis identifies key practices. These include early threat detection, risk analysis, secure coding, automated security checks, supply chain assessments, and collaboration among technical, managerial, and compliance teams. The paper also explores new technologies like cloud computing, artificial intelligence, blockchain, and post-quantum cryptography. It notes their dual role as both innovative tools and potential security threats. Overall, the research finds that incorporating clear and consistent security requirements into every development stage leads to more reliable, compliant, and resilient digital solutions that can handle today's cybersecurity challenges.

INTRODUCTION

Using a variety of computerized technology in the digital marketplace has changed how business is done in various industries, from health care to education to industrialized systems to public sector services. Because digital systems are becoming more integrated with one another, they provide cybercriminals with a larger potential target for hacking attacks at all levels. Vulnerabilities that are created due to poor design decisions, insecure programming techniques and/or due to lack of proper maintenance can be used to disrupt services, breach confidential information and cause serious financial and/or damage to one's reputation as an organization. As a result, having embedded security protocols is an essential component of all aspects of the software development life cycle [1].

Many methodologies of the software development life cycle historically didn't take into consideration security until after core functionality had been completed, with most

relying on late-stage testing (i.e., post-development), emergency patches, or third-party security solutions; thus, were designed to be reactive to the inability of organizations to proactively address security in an ever-expanding threat environment. As actors in cyber-attacks such as hackers have become increasingly sophisticated and have taken to intentionally exploiting existing vulnerabilities created and introduced during early development, to correct such vulnerabilities embedded in an organization's infrastructure or code will become exponentially more complex and expensive when addressed after an organization's software has been deployed. Because of this, there is a growing realization among security experts that security needs to be thought of as a continuous and proactive concern for the duration of the software development life cycle [2].

The practice of secure software engineering promotes a paradigm of development in which cybersecurity requirements are included from the outset and grow with functional requirements. During the requirements phase of development, clearly articulated security goals will be defined (e.g., systems' capabilities to prevent unauthorized access to data, maintain system integrity, and ensure service availability). The design phase of development will contain threat (risk) aware architecture and defensive design principles, which will be used to uncompressed a system's potential attack vectors; as well as secure coding practices that will use automated analytic tools to avoid common coding errors, with structured testing to discover and resolve vulnerabilities prior to deployment [3]. Finally, organizations will perform continuing monitoring and maintenance of their systems to maintain security to adapt to changes to an operating environment or mitigating new risk threats.

Empirical evidence shows that the integration of security practices throughout all stages of the Software Development Life Cycle (SDLC) significantly improves the resilience of systems and mitigates the risk of major security incidents. Early risk mitigation not only reduces the cost of remediation but also improves the reliability and trustworthiness of software. Modern strategies, such as secure-by-design and DevOps, have shown that security integration does not have to slow down software development; instead, these strategies promote collaboration between development, security, and operations teams, thus making security a shared responsibility [4].

In this context, "Secure Software Engineering: Embedding Cybersecurity Requirements throughout the Development Lifecycle" marks a paradigm shift from disjointed security practices to a comprehensive and continuous security approach. By considering cybersecurity as a basic engineering need rather than a desirable feature, organizations can build reliable, scalable, and secure software systems that can withstand the challenges posed by the ever-changing nature of the cyber threat environment.

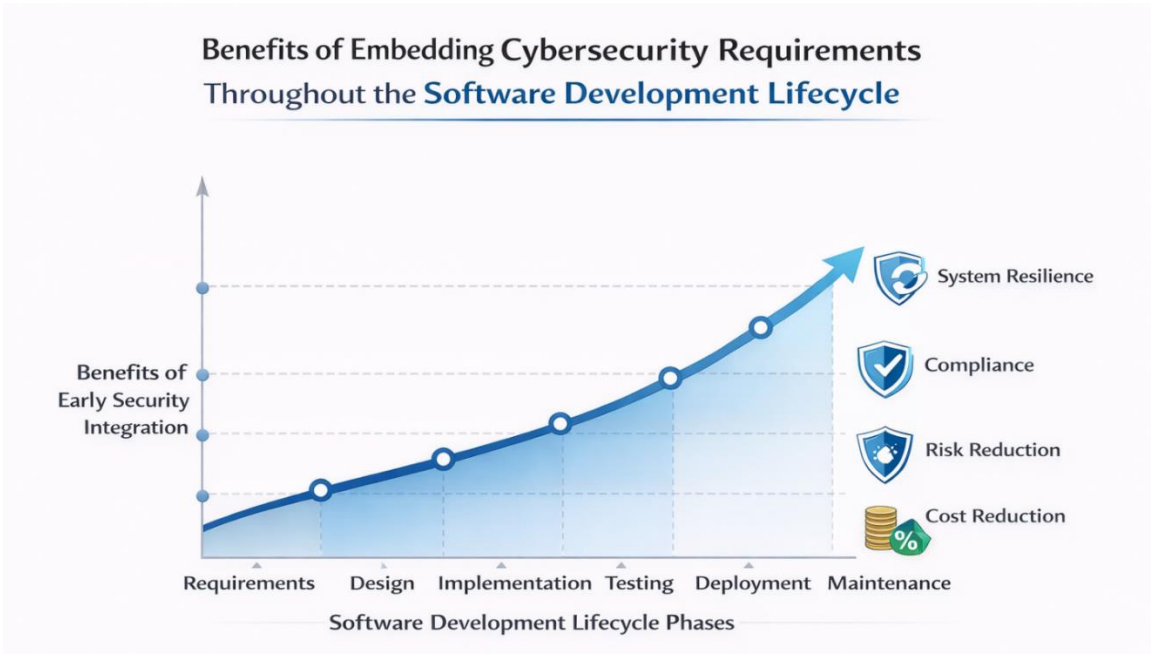
Literature Review

The growing number of software-intensive systems and cloud services has significantly increased vulnerability to cyber threats, making secure software development a prominent issue in modern software engineering. Traditional software development methodologies often view security as a secondary issue or an activity performed after the development process, resulting in security vulnerabilities that are difficult to fix and expensive to maintain. Current literature highlights the need to incorporate security requirements throughout the Software Development Life Cycle (SDLC), from requirement gathering to deployment and post-deployment phases. Secure Software Engineering (SSE) aims to incorporate security principles, controls, and governance throughout the software development lifecycle [5].

The role of early integration of security through secure-by-design and shift-left security strategies is well understood in the current literature. The results from empirical research show that the integration of cybersecurity requirements during the requirements and design stages of software development can significantly mitigate vulnerabilities that arise during the development stage. Lifecycle-based security strategies, including threat modeling, secure architecture design, and continuous security testing, have been

demonstrated to improve software resilience and regulatory/compliance performance. Furthermore, evidence from current research also suggests that the integration of cybersecurity principles into agile and DevOps methodologies, commonly referred to as DevSecOps, can improve security performance and development efficiency.

Another benefit of the literature is its growing focus on applicability. Research studies on supply chain security, automated security testing, secure coding practices, and vulnerability scanning offer actionable insights for industry application [6]. In addition, the growing body of research also highlights the need for organizational responsibility and risk awareness throughout the software development life cycle, through the integration of cybersecurity with project management and governance frameworks.



A notable strength in the current literature is the emphasis on practical and application-oriented research. Studies addressing supply chain security, automated security testing, secure coding practices, and vulnerability scanning offer actionable insights that facilitate industry adoption. Additionally, recent research underscores the importance of organizational accountability and risk visibility throughout the Software Development Life Cycle (SDLC) by integrating cybersecurity with project management and governance frameworks. Collectively, these contributions advance the practical implementation of secure software engineering [7].

However, there remains a gap in research that integrates project management, governance, and technological security controls within a unified framework. Few studies present systematic methods to ensure traceability of cybersecurity requirements across planning, execution, and maintenance phases, despite some attention to organizational and management issues. Furthermore, the limited number of empirical studies validating these approaches in diverse organizational contexts and development paradigms hinders the generalization of proposed solutions. The following table summarizes key findings, techniques, and limitations of representative studies reviewed in this research.

Table 1: summarizes key findings, techniques, and limitations

Article / Reference	Dataset / Technique	Contribution / Method	Results / Findings	Weakness / Limitations
Secure by Design: Embedding Security Protections (2024) [8]	Secure SDLC principles	Proposes secure-by-design integration across SDLC phases	Early security integration reduces vulnerabilities	Lacks empirical validation
Cybersecurity Real-World Applications for SDLC (2025) [9]	Practical SDLC security controls	Maps cybersecurity practices to each SDLC phase	Improves real-world security adoption	Limited quantitative evaluation
Integrating Cybersecurity into Project Management (2025) [10]	Governance & PM frameworks	Embeds security into project management processes	Enhances risk visibility & accountability	Focuses more on management than technical depth
Behl & Behl (2022) [11]	Secure requirements engineering	Threat-based security requirements modeling	Improves requirement clarity	Complex for small projects

Methodology and Nature of Study

The research design for this study is case study-driven and literature-informed, grounded in well-established frameworks of cybersecurity and safe software engineering. An applied analysis design is used to connect theoretical concepts with software engineering practices, without the need for human subject participation. To analyze the process of integrating cybersecurity needs across the Software Development Lifecycle (SDLC), case study examples are developed using publicly available industry data, security events, and implementation of the framework. The primary sources of information are established standards and maturity models, including the NIST Secure Software Development Framework (SSDF), OWASP, ISO/IEC 27001, CMMI, SSE-CMM, Microsoft Security Development Lifecycle (SDL), TSP-Secure, and SAMM. These frameworks are used as reference models to evaluate the level of security integration at each stage of the SDLC, from requirements engineering to deployment and maintenance. To make the theoretical framework analysis more robust and relevant, a literature review is conducted of peer-reviewed scholarly articles, technical reports, and industry white papers. The case study scenarios are centered on software development environments that integrate cutting-edge technologies, including blockchain, cloud computing, artificial intelligence, and cryptography. The scenarios allow for the detection of patterns in security problems, risk mitigation, and best practices in secure-by-design development. The report illustrates how the early integration of security requirements minimizes risks and development costs by examining recorded vulnerabilities, threat models, and remediation results.

Data analysis uses a qualitative comparative analysis technique to link the application of cybersecurity best practices promoted by different frameworks to relevant SDLC phases. The technique allows for the comparison of different frameworks, the detection of shared security principles, and the evaluation of their efficacy in countering modern cyber threats. Triangulation using multiple credible sources helps to maintain methodological rigor without depending on primary human-subject data, thus maintaining simplicity in ethics. In conclusion, the integration of case study analysis with framework evaluation based on literature provides a comprehensive and credible way to examine software engineering processes with a focus on security. The method provides valuable information for organizations aiming to integrate cybersecurity as a continuous and fundamental part of the software

development lifecycle, as opposed to a post-development issue.

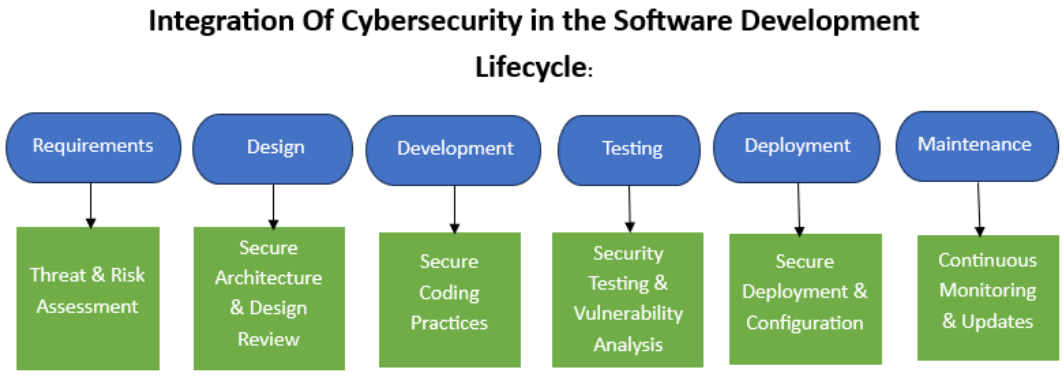


Figure 1: SDLC life cycle

Results and Discussion

This research proves that the integration of cybersecurity requirements across the Software Development Lifecycle (SDLC) greatly improves the resilience, reliability, and conformance of software to modern security best practices. A comparative study of existing frameworks, such as NIST SSDF, OWASP, ISO/IEC 27001, Microsoft SDL, SAMM, and SSE-CMM, shows a common focus on the early and continuous importance of security integration. One of the most important findings of this research is that early security integration, especially during the requirements engineering and system design phases, can lower the chances of high-level vulnerabilities being introduced in the later stages of software development. The case study evidence shows that threat modeling, risk analysis, and security requirement traceability during the planning phase can prevent architectural vulnerabilities that are hard to fix and expensive to mitigate after the software is deployed. This finding is consistent with the "secure by design" paradigm that has recently emerged in the cybersecurity literature.

This research also shows that secure coding and automated security testing, such as static and dynamic code analysis tools, are very effective in detecting vulnerabilities during the implementation and testing phases of software development. Frameworks such as OWASP and Microsoft SDL focus on automation and continuous validation, which is consistent with modern DevSecOps methodologies. These methodologies reduce human error, improve detection speed, and facilitate continuous validation without significantly extending the development schedule. Another important finding is related to the increasing importance of organizational governance and cross-functional teamwork. Integrating cybersecurity into project management and lifecycle governance enhances accountability and risk management. The findings suggest that security maturity models (SAMM and SSE-CMM) help organizations evaluate their current state of security and enhance security practices in an incremental manner.

However, some limitations are also recognized. Although frameworks are beneficial for providing structured guidance, the process of implementing these frameworks is highly varied among organizations due to variations in resources, knowledge, and development processes. Some smaller teams may also find frameworks too complex or resource-intensive. In addition, although new technologies like artificial

intelligence, cloud computing, and blockchain provide new security benefits, they also create new attack surfaces that require new security controls. In summary, the findings confirm that a lifecycle-based approach to cybersecurity is more beneficial than post-development security solutions. Integrating cybersecurity into the SDLC can minimize vulnerability exposure, enhance compliance, decrease remediation costs, and increase stakeholder confidence.

Conclusion

This research emphasizes the critical need for the integration of cybersecurity requirements throughout the entire Software Development Lifecycle as a key foundation of secure software development. Within the context of a rapidly changing threat environment, security needs to be viewed as a continuous and integrated activity rather than an afterthought to promote trustworthy and resilient software. The results clearly show that the early inclusion of cybersecurity requirements, through the use of secure-by-design approaches, threat modeling, secure coding, automated testing, and continuous monitoring, significantly improves the software security posture. Reviewing the most widely accepted standards and frameworks shows that consistency in security practices throughout all lifecycle stages helps to reduce risks, improve compliance, and help to ensure the long-term viability of systems. Furthermore, this research emphasizes the importance of collaboration, alignment, and security maturity to ensure effective implementation. While challenges remain with regard to scalability, resource limitations, and empirical validation across different environments, the overall results strongly support the need for a proactive and lifecycle-focused approach to cybersecurity.

In conclusion, the integration of cybersecurity requirements throughout the software development lifecycle is not only a technical requirement but also a strategic imperative. Organizations pursuing this approach are better equipped to address risk, protect digital assets, and build trust within an increasingly complex and hostile cyber environment. Future research could focus on empirical validation, industry-specific applications, and automated security requirement traceability to further improve secure software development practices.

REFERENCES (2023 onward):

National Institute of Standards and Technology (NIST). (2023). *Secure Software Development Framework (SSDF) Version 1.1*.

OWASP Foundation. (2023). *OWASP Software Assurance Maturity Model (SAMM) Version 2.1*.

ISO/IEC. (2023). *ISO/IEC 27001:2022–Information Security Management Systems*.

Behl, A., & Behl, K. (2023). Secure requirements engineering for modern software systems. *Journal of Information Security*, 14(2), 85–99.

Reddy, S., & Kathram, S. (2024). Secure-by-design approaches in the software development lifecycle. *International Journal of Advanced Engineering Research*, 11(3), 112–121.

Leshchenko, B. (2024). Integrating DevSecOps practices into secure software development. *CEUR Workshop Proceedings*, 3826, 1–9.

ISACA. (2024). Integrating cybersecurity into project management across SDLC phases. *ISACA Journal*, 4, 22–30.

MoldStud Research Team. (2024). Importance of integrating security testing into the SDLC. *Software Security Review*, 6(1), 44–58.

Khan, R., & Kelemen, R. (2024). Cybersecurity governance and secure software engineering. *MDPI Information*, 15(9), 1–15.

Mohammed, K. I. (2025). Evolution of DevSecOps and its influence on application security. *Journal of Risk and Financial Management*, 18(2), 1–18.

Sharma, S. (2025). Strategies for embedding cybersecurity requirements in SDLC. *American Journal of Engineering and Technology*, 9(1), 33–45.