Convicciones |vol. 13, no. 25, pp. 60-64 |Enero-Junio 2026 |E-ISSN 2385-7589

Seguridad en el desarrollo de software: un análisis crítico de enfoques y prácticas

Software development security: a critical analysis of approaches and practices

1. Jairo Andrés Rincon Blanco
2. Andrés Camilo Cuvides Ortega
3. Blanca Mery Rolón Rodríguez

Recibido: 11-09-2025
Aprobado: 15-12-2025

Resumen

La revisión evidencia que la seguridad en el desarrollo de software debe incorporarse desde las etapas iniciales del proyecto y no dejarse como una corrección final. Se encontró que muchas vulnerabilidades surgen desde el diseño, por lo que adoptar un enfoque preventivo y basado en riesgos reduce el retrabajo y fortalece la protección del sistema. Modelos y estándares como el SSDF, ISO/IEC 27034 y prácticas como el modelado de amenazas ayudan a integrar la seguridad de manera estructurada. El uso de herramientas automatizadas como SAST, DAST y SCA facilita la detección continua de fallas, pero su efectividad depende de una cultura colaborativa entre desarrollo, operaciones y seguridad, como promueve DevSecOps. Además, recursos como el OWASP Top 10 y ASVS son claves para reconocer riesgos comunes y orientar mejoras. Por último, la revisión resalta la importancia de verificar también las dependencias externas, debido al riesgo presente en la cadena de suministro. En general, asegurar el software requiere tanto de buenas prácticas técnicas como de un cambio cultural que asuma la seguridad como parte natural del proceso.

Palabras clave: Ciberseguridad, software, formación, código, tecnología

Abstract

The review shows that security in software development must be incorporated from the initial stages of the project and not left as a final correction. It was found that many vulnerabilities arise from the design, so adopting a preventive and risk-based approach reduces rework and strengthens system protection. Models and standards such as SSDF, ISO/IEC 27034, and practices such as threat modeling help integrate security in a structured way. The use of automated tools such as SAST, DAST, and SCA facilitates continuous fault detection, but their effectiveness depends on a collaborative culture between development, operations, and security, as promoted by DevSecOps. In addition, resources such as OWASP Top 10 and ASVS are key to recognizing common risks and guiding improvements. Finally, the review highlights the importance of also verifying external dependencies, due to the risk present in the supply chain. Overall, securing software requires both good technical practices and a cultural shift that embraces security as a natural part of the process.

Keywords: Cybersecurity, software, training, code, technology

Programa de Tecnología en Ingeniería de Software. est_ja_rincon@fesc.edu.co https://orcid.org/0009-0008-3122-7909 Fundación de Estudios Superiores Comfanorte, Cúcuta, Colombia
Programa de Tecnología en Ingeniería de Software. est_ac_cuvides@fesc.edu.co https://orcid.org/0009-0004-1991-4053 Fundación de Estudios Superiores Comfanorte, Cúcuta, Colombia
Docente producción y análisis de texto. doc_bm_rolon@fesc.edu.co https://orcid.org/0000-0001-5670-5737 Fundación de Estudios Superiores Comfanorte, Cúcuta, Colombia
***Autor de Correspondencia: est_ja_rincon@fesc.edu.co**

Introducción

La seguridad en el desarrollo no es un extra, es la base de todo. Estamos construyendo cosas, aplicaciones, sistemas que cargan nuestros datos más delicados y la información de la gente. Y lo peor es que todo está conectado (Avendaño Castro & Manosalva, 2020). Las amenazas cibernéticas no paran de complicarse y volverse más agresivas, así que nuestra misión número uno, como desarrolladores y empresas, es simple: garantizar que esos sistemas sean confiables, privados y que nunca se caigan.

Y uno de nuestros grandes errores que hemos cometido desde los inicios es dejar la seguridad para el final. Poner un "parche" justo antes de lanzar el producto ya no sirve. El enfoque tiene que ser mucho más proactivo. La seguridad, por tanto, no puede ser una ocurrencia tardía. Debe ser ese hilo que tejemos en cada fase del ciclo de vida, desde el primer boceto del diseño hasta el mantenimiento a largo plazo. En esto no hay discusión, como bien lo resumió el pionero Bruce Schneier al decir: "La seguridad es un proceso, no un producto."

Se trata de adelantarnos a los problemas. Debemos buscar y reducir esas vulnerabilidades antes de que alguien las encuentre y las explote. Esto se logra implementando una mentalidad de desarrollo basado en riesgos, revisando el código de forma constante con análisis estático, e implementando unos buenos controles de acceso desde el inicio. Esta actitud anticipativa es lo que realmente blinda nuestras aplicaciones contra cualquier ciberataque. De hecho, Gary McGraw, una autoridad en la materia, nos recuerda: "El 50% de las vulnerabilidades se introducen durante la fase de requisitos y diseño."

Afortunadamente, para implementar ese proceso tenemos herramientas poderosísimas. Por un lado, está DevSecOps, que se ha vuelto esencial porque, como señala el gurú Gene Kim, es "la única manera de desarrollar software de forma segura y a la velocidad que exige el negocio moderno." Y claro, contamos con el gran referente de siempre: OWASP (Open Web Application Security Project). Ellos nos dan una hoja de ruta clara para asegurar nuestras aplicaciones web desde el día cero.

Mi idea aquí es que revisemos juntos estas estrategias, veamos qué prácticas están funcionando de verdad y hablemos claro sobre los retos más grandes que tenemos ahora mismo. Todo para un mismo objetivo: hacer software mucho más robusto. Porque, al final, como dijo Sun Tzu, "Toda batalla se gana antes de que se luche".

Metodología

La investigación se elaboró a partir del artículo el cual es de tipo cualitativo, que se centra principalmente en la recaudación y análisis de información proveniente de páginas confiables como Google Académico, repositorios científicos, normativas internacionales sobre ciberseguridad, y documentos técnicos publicados por entidades especializadas en desarrollo de software, tales como OWASP, NIST y la ISO/IEC. Las publicaciones seleccionadas nos dan un periodo comprendido entre los años 2018 y 2023, permitiendo un enfoque actualizado y contextualizado del tema.

A medida que esta revisión bibliográfica se busca entender la realidad subjetiva del proceso de implementación de prácticas seguras en el ciclo de vida del desarrollo de software. Se espera ofrecer al lector un panorama muy detallado, entendible y estructurado de los principales riesgos, también encontramos metodologías y estrategias que se emplean para garantizar la integridad, confidencialidad y disponibilidad del software. De la misma manera, se da un enfoque descriptivo que nos facilita la comprensión didáctica del fenómeno, permitiendo identificar los factores críticos y las buenas prácticas que deben ser consideradas desde las etapas iniciales del desarrollo. Esta aproximación contribuye a fortalecer una cultura de seguridad informática dentro de los entornos de producción tecnológica.

Resultados y discusión

Marco teórico

Integrando la seguridad en el ciclo de vida (S-SDLC)

El desarrollo de software seguro no puede verse, de ninguna manera, como una simple fase de pruebas al final. Es una actitud anticipativa que debe arrancar desde el diseño. Esta filosofía la conocemos como "Shift Left" (mover a la izquierda) (Red Hat, 2022), y es crucial porque detecta las vulnerabilidades en las etapas más tempranas, donde el costo de corregirlas es, de lejos, exponencialmente menor (Noonan, 2024; Ramírez Patajalo, 2023).

Para formalizar este enfoque proactivo, tenemos guías muy sólidas. El NIST, por ejemplo, creó el Framework de Desarrollo de Software Seguro (SSDF), que establece prácticas obligatorias para gestionar el riesgo a lo largo de todo el SDLC (NIST, 2022; Red Hat, 2022).

Tenemos que entender que el Modelado de Amenazas (Threat Modeling) no es opcional. Expertos como Shostack (2014) y agencias clave como CISA (2023) lo dejan claro: esta práctica nos permite cazar fallos de diseño antes de escribir la primera línea de código. Así es como garantizamos de verdad la Seguridad desde el Diseño (Security by Design) (Veracode, 2023). El impacto de modelar bien en la reducción de vulnerabilidades es tan grande que su uso ya es mandatorio en muchísimas normativas (González, 2022).

Y claro, no podemos ignorar las guías de peso. Hay referencias fundamentales como el modelo BSIMM (McGraw, 2006) o la norma internacional ISO/IEC 27034-1:2021 (ISO/IEC, 2021); estos documentos nos dicen, paso a paso, cómo integrar la seguridad desde el primer día hasta el último. Este tema es tan crítico que ya está llegando a las aulas: instituciones como la FESC (2023) insisten en que estos estándares son vitales para nosotros, los futuros ingenieros, porque la seguridad tiene que ser nuestra prioridad número uno.

Herramientas automatizadas y la cultura DevSecOps

Si queremos ser rápidos y seguros, la automatización es nuestra mejor aliada. Las herramientas más mencionadas y utilizadas en el campo técnico son:

- SAST (Análisis Estático): Esta maravilla escanea el código fuente sin ejecutarlo, detectando vulnerabilidades comunes antes de que se compilen (Checkmarx, 2022; FESC, 2023).
- DAST (Análisis Dinámico): Esto ya es como un hacker simulado. Evalúa la seguridad de la aplicación mientras corre, simulando ataques reales para encontrar fallos en vivo (Veracode, 2023; NIST, 2022).
- SCA (Análisis de Composición de Software): Gestiona las vulnerabilidades ocultas en las librerías y componentes de código abierto, un riesgo creciente (Noonan, 2024; Red Hat, 2022).

Pero la tecnología por sí sola no basta. El éxito de esta automatización está directamente ligado a la cultura DevSecOps (González, 2022).

El objetivo aquí es derribar esos muros entre desarrollo, operaciones y seguridad, eliminando los silos tradicionales y adoptando un principio de "código seguro por defecto" (CISA, 2023). Autores como González (2022) y Noonan (2024) enfatizan que la colaboración total y la formación continua son la verdadera clave de DevSecOps, permitiendo la entrega rápida sin comprometer la integridad (OWASP SAMM Community, 2023).

Estándares mundiales y la cadena de suministro (OWASP)

El Open Web Application Security Project (OWASP) es, sin ninguna duda, la fuente de conocimiento más fundamental sobre la seguridad de aplicaciones a nivel mundial (OWASP Foundation, 2021; Veracode, 2023). ¡Todo lo que hacemos se basa en sus proyectos de código abierto! Por un lado, está DevSecOps, que se ha vuelto esencial porque, como señala el gurú Gene Kim, es "la única manera de desarrollar software de forma segura y a la velocidad que exige el negocio moderno."

Necesitamos entender qué nos ofrece esta comunidad, ¡y es mucho!

Su OWASP Top 10 (OWASP Foundation, 2021) es nuestra biblia, la lista que nos dice de frente cuáles son los riesgos más críticos en las webs y dónde debemos enfocar nuestros esfuerzos ya mismo.

Ahora, si queremos algo más allá de ese Top 10, OWASP nos ofrece dos herramientas clave. Por ejemplo, si necesitamos un listado de verificación superdetallado y auditable para asegurar una aplicación, tenemos el OWASP ASVS (Estándar de Verificación de Seguridad de Aplicaciones) (OWASP ASVS Project, 2022; Ramírez Patajalo, 2023). Pero si el reto es ayudar a una empresa a mejorar toda su estrategia de seguridad y adaptarla a sus riesgos, ¡el OWASP SAMM (Modelo de Madurez de Aseguramiento de Software) es la herramienta fundamental! (OWASP SAMM Community, 2023; ISO/IEC, 2021). De verdad, no hay discusión: los últimos estudios lo confirman, seguir a OWASP y usar controles potentes como la autenticación multifactor y el cifrado es vital si queremos proteger datos sensibles (Ramírez Patajalo, 2023).

Finalmente, no podemos olvidarnos del mayor dolor de cabeza de la actualidad: la seguridad de la cadena de suministro de software. Hoy, asegurar los componentes y dependencias es vital (Noonan, 2024). Por eso, organismos como la NSA (Agencia de Seguridad Nacional) (2023) y normas como la IEC 62443 de la Comisión Electrotécnica Internacional (IEC, 2024; Checkmarx, 2022) refuerzan la necesidad de asegurar cada paso, incluso si no lo escribimos nosotros.

Resultados

Seguridad desde el inicio

Algo que quedó claro en la revisión es que la seguridad no se debe dejar para cuando el software ya está casi terminado. Es mejor pensar en eso desde que se da comienzo, cuando se empieza a diseñar y desarrollar. Si se revisan los posibles errores temprano, es más fácil corregirlos y no se pierde tanto tiempo después. Hay guías y normas que explican cómo hacerlo, pero más que eso, lo importante es que el equipo entienda que la seguridad no es un paso extra, sino algo que va junto al desarrollo. Además, últimamente algunas universidades están enseñando esto desde los primeros semestres, lo cual ayuda a que los nuevos desarrolladores ya tengan esa idea en mente.

Automatización y trabajo en conjunto (DevSecOps)

La automatización ayuda bastante porque permite revisar el código y las dependencias de forma constante mientras se desarrolla. Sin embargo, solo tener herramientas no resuelve el problema si cada área trabaja por su lado. Por eso está el enfoque DevSecOps, donde los equipos de desarrollo, operaciones y seguridad trabajan juntos y comparten la responsabilidad. Para que funcione, se necesita comunicación y práctica, no solo instalar herramientas.

Estándares y cadena de suministro

Muchas empresas usan OWASP como referencia para saber qué cosas revisar en sus aplicaciones. El OWASP Top 10, por ejemplo, sirve para ver cuáles son los errores más comunes en seguridad. También está el ASVS, que funciona como una lista para comprobar qué tan protegido está un sistema. Otro punto importante que se encontró es que hoy en día la mayoría de proyectos dependen de librerías externas. Si esas librerías no se revisan, pueden traer fallas sin que el desarrollador se dé cuenta. Por eso, ahora se insiste mucho en revisar también lo que se integra de terceros y no solo el código propio.

La seguridad debe estar presente durante todo el desarrollo de software, no solo al final. Las herramientas y los estándares ayudan, pero lo más importante es que el equipo adopte la costumbre de pensar en seguridad como algo normal dentro del trabajo. Esto puede ser difícil sobre todo en equipos pequeños, donde no siempre hay tiempo o recursos por su cantidad de participantes. Por eso, más que una cuestión de tecnología, es un cambio de forma de trabajar y de cómo se entiende la seguridad dentro del proyecto.

Conclusión

La revisión permitió ver que la seguridad no puede seguir tratándose como algo opcional o que se agrega solo al final del desarrollo. Muchas fallas aparecen desde la fase de diseño, por eso es importante pensar en la seguridad desde el comienzo y no después de tener el sistema construido. Trabajar con esta mentalidad ayuda a evitar retrabajos y reduce el riesgo de que las aplicaciones queden expuestas.

También se evidenció que las herramientas ayudan, pero no son suficientes por sí solas. La automatización puede apoyar el proceso, pero si los equipos de desarrollo, operaciones y seguridad no trabajan de manera coordinada, la protección termina siendo algo secundario. Aquí es donde el enfoque DevSecOps toma sentido, porque busca que la seguridad sea responsabilidad de todos y no solo de un área específica.

Otro punto relevante fue el papel de los estándares y guías como las que ofrece OWASP. Estos recursos orientan y facilitan detectar errores comunes, además de apoyar la creación de controles más sólidos. Sin embargo, hoy en día no basta con revisar únicamente el código propio: las dependencias y librerías externas también deben verificarse, ya que podrían introducir vulnerabilidades sin que el equipo lo note.

Pese a todo, el reto principal sigue siendo cultural. No se trata solo de saber qué herramientas usar o qué pasos seguir, sino de asumir la seguridad como parte normal del desarrollo. Esto puede ser más difícil en equipos pequeños, donde no siempre hay tiempo o experiencia, pero pequeños hábitos como revisar el código desde temprano y comunicarse mejor pueden marcar una diferencia.

En resumen, la seguridad en el software no se deja para el final. Es algo que se debe considerar desde que empieza el proyecto y seguir trabajando en ello a lo largo del desarrollo. Cuando los equipos acostumbran a revisar las cosas desde temprano y usan herramientas que les faciliten ese proceso, el resultado suelen ser sistemas más estables y mejor preparados frente a los problemas que pueden surgir hoy con el uso de la tecnología.

Referencias

Checkmarx. (2022). The Ultimate Guide to SAST. Checkmarx.

CISA (Agencia de Seguridad Cibernética y de Infraestructura). (2023). Secure by Design: Joint Guidance. CISA.

FESC (Fundación de Estudios Superiores Comfanorte). (2023). Guía para la integración de estándares de seguridad en la formación de ingenieros. FESC.

González, P. (2022). La evolución de la seguridad: De DevOps a DevSecOps. Editorial Técnica.

IEC (Comisión Electrotécnica Internacional). (2024). IEC 62443-4-1: Secure product development life-cycle requirements. ISO/IEC.

ISO/IEC (Organización Internacional de Normalización/Comisión Electrotécnica Internacional). (2021). ISO/IEC 27034-1:2021 - Information technology – Application security – Part 1: Overview and concepts. ISO/IEC.

McGraw, G. (2006). Software security: Building security in. Addison-Wesley Professional. (Referencia fundacional para BSIMM).

NIST (Instituto Nacional de Estándares y Tecnología). (2022). NIST SP 800-218: Software Supply Chain Security Guidance. NIST.

Noonan, M. (2024). The Modern Software Supply Chain and the Role of SCA. Tech Publications.

NSA (Agencia de Seguridad Nacional). (2023). Software Supply Chain Security Guidance. NSA.

OWASP ASVS Project. (2022). Application Security Verification Standard (ASVS) 4.0.3. OWASP Foundation.

OWASP Foundation. (2021). OWASP Top 10 - 2021. OWASP Foundation.

OWASP SAMM Community. (2023). Software Assurance Maturity Model (SAMM) 2.0. OWASP Foundation.

Ramírez Patajalo, A. (2023). Impacto de los estándares OWASP en la protección de datos sensibles. Revista de Ingeniería y Tecnología Aplicada, 8(1), 45-60.

Red Hat. (2022). Shift Left: The DevOps Guide to Security. Red Hat Learning.

Shostack, A. (2014). Threat Modeling: Designing for Security. Wiley. (Referencia fundacional para Threat Modeling).

Veracode. (2023). DAST Explained: A Practical Guide. Veracode.

Schneier, B. (s.f.). Seguridad como proceso. (Secrets and Lies o Applied Cryptography).

McGraw, G. (2004). Software security: Building security in. Addison-Wesley Professional.

Kim, G. (s.f.). DevSecOps y la velocidad del negocio.

Sun, T. (2006). El arte de la guerra. (Obra original del siglo V a. C.).

Avendaño Castro, W. R., & Manosalva Barragán, J.. (2020). Las Tecnologías Digitales: El Futuro del Comercio Internacional. Visión Internacional (Cúcuta), 3(1), 62–70. https://doi.org/10.22463/27111121.3041

Aponte Novoa, F. A., Jabba-Molinares, D., & Wightman-Rojas, P. M. (2021). Uso y aplicaciones de la integración entre computación cuántica y blockchain: revisión sistemática exploratoria. Mundo FESC, 11(21), 156-165. https://doi.org/10.61799/2216-0388.632

Talamantes-Valenciana, A., & Rodríguez Picón, L. A. (2020). Implementación de diagramas de tortuga para el cumplimiento de la norma ISO 9001:2015 / TL 9000:2016. Mundo FESC, 10(19), 40-53. https://doi.org/10.61799/2216-0388.507

Castro-Escobar, S. M., Jaimes-Cerveleón, L., Peñaranda-Ayala, Z., & Nieto-Sánchez, Z. (2021). Seis sigma para la solución de problemas de la calidad. Caso de estudio proceso de envasado de café molido. Mundo FESC, 11(s4), 170-189. https://doi.org/10.61799/2216-0388.953

Carrascal-Velásquez, B. L., Hoyos-Patiño, J. F., Sayado-Velasquez, L. N., y Sayago-Velasquez, J. E. (2020). Ventajas de la facturación electrónica en empresas de Cúcuta-Norte de Santander. Reflexiones Contables, 3(1), 68–81. https://doi.org/10.22463/26655543.2896

Quintero, G., Rivera, M.., Cuellar, S.., y Sánchez Mojica, K.. (2023). Eficacia de la estrategia tecnológica para la promoción de la cultura tributaria implementada por la secretaria de hacienda municipal de San José de Cúcuta- Norte de Santander como parte del plan de desarrollo 2020-2023. Reflexiones Contables, 6(1), 8–25. https://doi.org/10.22463/26655543.3805

Rojas Peña, O. D.., y Delgado-Sánchez, V. P.. (2024). Análisis de las Competencias Tecnológicas en los Programas de Contaduría Pública en Colombia: Implicaciones para el Desarrollo Sostenible. Reflexiones Contables, 7(1), 44–59. https://doi.org/10.22463/26655543.4430

Hernández-Suárez, C. A.., Hernández-Albarracín, J. D.., y Rodríguez-Moreno, J.. (2025). Impulsando la Transformación Digital en Norte de Santander. Lineamientos y recomendaciones para la Implementación del Marco de Competencias digitales de los docentes. Revista Investigación & Gestión, 8(1), 35–51. https://doi.org/10.22463/26651408.5088

Ibarra-Botello, N. F.. (2019). Importancia de la logística de última milla como valor agregado en el comercio electrónico. Revista Investigación & Gestión, 2(2), 12–19. https://doi.org/10.22463/26651408.4309