INVESTIGACIÓN

Integración de prácticas de seguridad para pipelines DevSecOps en entornos contenerizados

Integration of security practices for DevSecOps pipelines in containerized environments

Ing. Mary Nelsa Bonne Cuza[*], Téc. Meliza León Bencomo[2]

Recibido: 11/2025 | Aceptado: 11/2025 | Publicado: 12/2025

Resumen

La seguridad en el desarrollo de *software* es una preocupación primordial en la era digital actual, especialmente en entornos de integración y entrega continua. A medida que las amenazas cibernéticas evolucionan, es necesario que los equipos de desarrollo integren la seguridad desde las fases iniciales del ciclo de vida del *software*. La investigación se centra en implementar de prácticas DevSecOps, que buscan incorporar la seguridad desde el inicio del *pipeline*. Se emplearon métodos cualitativos y cuantitativos como análisis de casos y revisión documental, para identificar mejores prácticas y desafíos en el campo. Los hallazgos indican que la falta de alineación entre los equipos de desarrollo, operaciones y seguridad es un obstáculo crítico. Sin embargo, la implementación efectiva de herramientas automatizadas mejora de forma significativa seguridad. Las conclusiones destacan la necesidad de fomentar una cultura colaborativa y un

[*] Facultad de Ciberseguridad. Universidad de las Ciencia Informáticas. Carretera a San Antonio Km 2 ½ Torrens. Boyeros. La Habana. Cuba. mary@uci.cu

[2] Dirección de Seguridad Informática. Universidad de las Ciencia Informáticas. Carretera a San Antonio Km 2 ½ Torrens. Boyeros. La Habana. Cuba. mary@uci.cu

enfoque proactivo hacia la seguridad. Se resalta la urgencia de abordar las preocupaciones de seguridad en el desarrollo de *software* y se propone un proceso claro hacia una integración exitosa de prácticas DevSecOps.

Palabras clave: contenerización, entrega continua, integración continua, seguridad

Abstract

Security in software development is a paramount concern in today's digital age, especially in continuous integration and continuous delivery environments. As cyber threats evolve, it is necessary for development teams to integrate security from the early stages of the software lifecycle. The research focuses on the implementation of DevSecOps practices, which seek to incorporate security from the beginning of the pipeline. Qualitative and quantitative methods, such as case analysis and document review, were used to identify best practices and challenges in the field. The findings indicate that the lack of alignment between development, operations and security teams is a critical obstacle. However, effective implementation of automated tools significantly improves the security posture. The findings highlight the need to foster a collaborative culture and a proactive approach to security. It highlights the urgency of addressing security concerns in software development, proposing a clear process towards successful integration of DevSecOps practices.

Keywords: *containerization, continuous delivery, continuous integration, security*

Introducción

La evolución de las prácticas de desarrollo de *software* ha llevado a la creación de metodologías que buscan mejorar la eficiencia y la colaboración entre los equipos involucrados. Una de las innovaciones más significativas en el ámbito es el concepto de DevSecOps, que ha transformado la manera en que los desarrolladores y los equipos de operaciones interactúan y trabajan juntos. Su propósito es optimizar el proceso de entrega de *software*. Esta metodología promueve la

colaboración entre operadores y desarrolladores, con el objetivo de igualar su agilidad hacia un objetivo común (Muñoz, 2023).

La seguridad de la información y el cumplimiento normativo son fundamentales en el desarrollo de *software*. Estadísticas recientes revelan un aumento notable en los costos asociados a las violaciones de datos, así como en la frecuencia y sofisticación de los ataques dirigidos a las aplicaciones y sistemas. La gestión de vulnerabilidades se ha convertido en un desafío urgente, con tiempos de reparación que superan los 200 días, según datos del Instituto Nacional de Estándares y Tecnología (NIST, por sus siglas en inglés) (Madnick, 2023).

El panorama muestra una tendencia creciente hacia la automatización y la mejora continua en las operaciones de *software*. Entre 2015 y 2025 se ha observado una adopción significativa de DevSecOps, cuyo crecimiento exponencial refleja la madurez progresiva de las prácticas de seguridad integradas. En este contexto, DevSecOps se consolida como el enfoque clave para abordar los desafíos de seguridad. La metodología se establece como una necesidad imperativa para mitigar riesgos y asegurar la integridad de las aplicaciones (Chandramouli et al., 2024; *GitLab*, 2025). Para mejorar la velocidad de implementación, la aplicación de parches, el escalado y la eficiencia en el desarrollo del *software*, se utilizan las tecnologías tales como Docker y Kubernetes. En esta investigación las autoras se centraron en la tecnología de contenerización Kubernetes.

Kubernetes permite organizar y gestionar el trabajo en DevSecOps de manera efectiva al proporcionar una infraestructura escalable y automatizada que facilita el despliegue de aplicaciones. Muchas organizaciones han adoptado esta tecnología para implementar políticas de seguridad específicas y reducir la superficie de ataque, mejorando la resistencia ante posibles amenazas. La integración de DevSecOps en Kubernetes se materializa a través de un *pipeline* de seguridad continua que abarca múltiples capas de protección (Kubernetes, 2024).

No obstante, el entorno presenta una serie de desafíos críticos relacionados con la seguridad del *pipeline*. Una de las problemáticas más significativas se manifiesta cuando un despliegue reciente se ve comprometido por una vulnerabilidad no detectada en una de

las imágenes del contenedor. La ausencia de estándares específicos y mejores prácticas establecidas impide que el equipo de desarrollo cuente con un marco referencial que dirija la implementación de medidas de seguridad a lo largo del SDLC (*System Development Life Cycle*).

Este vacío normativo retrasa la detección de los problemas y compromete la integridad de la aplicación, lo que afecta de manera negativa la confianza del cliente y la reputación corporativa. La situación genera incertidumbre con respecto a la conformidad regulatoria que podría conducir a sanciones económicas significativas y la pérdida de licencias operativas para el funcionamiento del negocio. La carencia de visibilidad respecto al estado y comportamiento del *pipeline* también representa un desafío crítico. Sin herramientas apropiadas para la supervisión resulta casi imposible detectar anomalías o comportamientos sospechosos en tiempo real. Dicha situación conduce a un aumento en los tiempos de respuesta ante incidentes, lo que afecta gravemente la continuidad operativa. En ausencia de prácticas estandarizadas para gestionar configuraciones seguras, el equipo de desarrollo identifica que muchas aplicaciones son implementadas con configuraciones inseguras por defecto, lo que genera un entorno propenso a ataques cibernéticos y vulnerabilidades susceptibles de ser explotadas por actores maliciosos.

Por lo antes expuesto, las autoras decidieron realizar una investigación sobre las prácticas más idóneas en materia de seguridad que se pueden integrar en el *pipeline* DevSecOps en entornos contenerizados con el fin de garantizar la protección integral del *software* desde su concepción hasta su despliegue en producción.

Materiales y métodos

Para la realización de la propuesta de solución, las autoras tuvieron en cuenta varios métodos científicos que permitieron una mayor organización en el trabajo. Se utilizó el método **análisis-síntesis** para descomponer el problema de investigación en sus componentes principales y, posteriormente, integra estos elementos en la propuesta del procedimiento. Este proceso facilitó la identificación de las in-

terrelaciones entre las herramientas, estándares y mejores prácticas de seguridad utilizadas. El método de **observación** contribuyó a la definición del objeto de estudio y las herramientas empleadas; permitió captar información relevante sobre el tema de investigación. El método de **inducción-deducción** se utilizó para tener una visión clara del procedimiento que se quiere elaborar.

Se tomaron en cuenta los fundamentos necesarios sobre los elementos que intervienen en los *pipelines* DevSecOps en Kubernetes como la entrega e integración continua (CI/CD) en la implementación del *pipeline* DevSecOps, la cual garantiza que cada etapa del ciclo de vida del *software* esté respaldada por medidas robustas (Muñoz, 2023; Xygeni, 2024). Según Kev Zettler (en su publicación de Atlassian, 2024), la implementación de DevSecOps cuenta con cinco etapas cíclicas para su desarrollo: Planificación, Compilación, Prueba, Despliegue y Monitoreo (Atlassian, 2024). Estas etapas se convierten en un marco fundamental para garantizar que las aplicaciones sean seguras y alineadas con los objetivos estratégicos de la organización. Por su parte, Luca Antoniotti, desarrollador de *software*, define, en su tesis de maestría para el Politécnico de Torino, cuatro etapas fundamentales para el *pipeline* DevSecOps con diferencias sutiles respecto a la metodología general: Construcción, Prueba, Entrega y Despliegue (Antoniotti, 2021).

Se consideraron estándares y regulaciones para DevSecOps. Entre las principales se encuentran:

• *ISO/IEC 27001:2022:* proporciona un marco adaptable para alinear la seguridad con los objetivos empresariales (Zhu & Zou, 2021).

• *NIST SP 800-53:* brinda lineamientos específicos para proteger sistemas y datos en entornos dinámicos, como *pipelines* de CI/CD (Zayni & Nassereddine, 2020).

Se efectuó un análisis de soluciones para la integración de estándares y mejores prácticas de seguridad para el *pipeline* DevSecOps en Kubernetes a partir del diseño de un protocolo de búsqueda que facilitó la observación de artículos y estudios relacionados con el tema de investigación y se sentaron las bases para confección de la propuesta de solución.

La solución presentada integra herramientas y tecnologías para la conformación de estándares y mejores prácticas de seguridad para el *pipeline* DevSecOps en entornos contenerizados. Se utilizó Tekton, un marco de automatización de código abierto (específicamente en entornos Kubernetes), en particular para crear sistemas de CI/CD, que permite a los desarrolladores construir, probar e implementar en proveedores de nube y sistemas locales. La elección de Tekton se justifica por su capacidad para proporcionar una plataforma extensible y modular que facilita la automatización de flujos de trabajo (Tekton, 2023).

Resultados y discusión

En el ámbito de la gestión de procesos y controles de seguridad, la estructuración de procedimientos definidos es fundamental para garantizar la consistencia y el cumplimiento de los objetivos establecidos. El procedimiento propuesto en esta investigación se compone de tres fases principales, las cuales agrupan un total de seis pasos que permiten definir, implementar y validar controles de seguridad de manera sistemática, lo que asegura su alineación con estándares reconocidos y la correcta asignación de responsabilidades. En la Figura 1 se muestra el proceso organizado en tres fases principales:

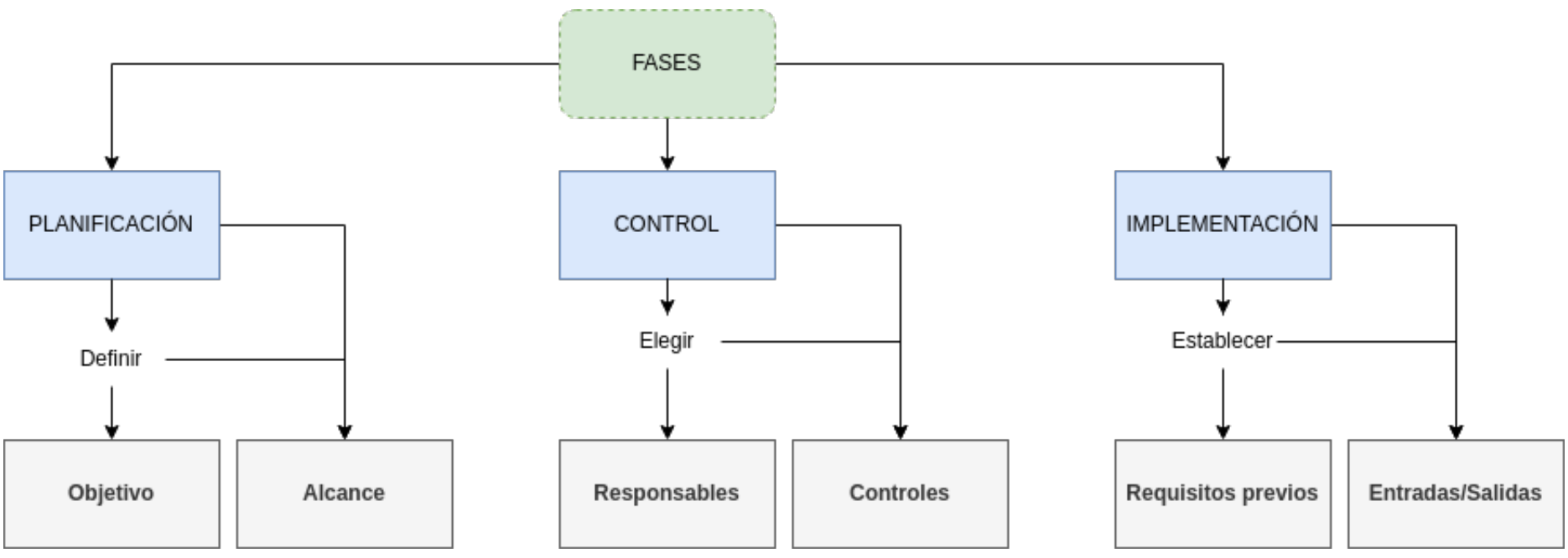

Figura 1. Estructura del procedimiento a desarrollar

1. Fase de Planificación

- Comienza con la definición del objetivo, que establece el propósito principal.
- El alcance se deriva del objetivo y define los límites del procedimiento.

2. Fase de Control

- La selección de controles de seguridad proporciona las medidas necesarias.
- La asignación de responsables garantiza la ejecución adecuada.

3. Fase de Implementación

- Se establecen los requisitos específicos para cada control.
- Por último, se definen las entradas y salidas adaptadas al proceso.

A través de este enfoque, se busca no solo establecer un marco de acción definido, sino también facilitar la adaptabilidad a diferentes contextos organizacionales, integrando requisitos previos, acciones esperadas y mecanismos de retroalimentación. La aplicación metodológica de estos pasos contribuye a la mejora continua, minimiza riesgos y optimiza la eficacia de los controles implementados.

Descripción de las fases para la implementación del procedimiento

1. La **fase de Planificación** es fundamental en cualquier procedimiento, su objetivo principal es establecer de manera clara y precisa qué se desea lograr. Este proceso proporciona una dirección específica y define un propósito concreto que guiará todas las acciones derivadas (véase Cuadro 1).

Cuadro 1. Fase de Planificación del procedimiento

Objetivo del procedimiento	Alcance del procedimiento
• Integrar controles de seguridad automatizados en el *pipeline* de CI/CD mediante estándares reconocidos (*NIST SP 800-53* e *ISO/IEC 27001/2022*), lo que combina la implementación técnica. • El enfoque unificado busca garantizar que la seguridad no sea una fase posterior, sino un proceso automatizado desde la construcción hasta el despliegue, reduciendo riesgos sin comprometer agilidad.	• Alcance multidimensional: La solución propuesta tiene un impacto en tres ámbitos fundamentales: en el empresarial, que abarca todos los niveles organizacionales desde el desarrollo hasta las operaciones; en el técnico, que cubre la infraestructura de Kubernetes y sus componentes; y en el de seguridad, que implementa controles asegurando el cumplimiento de estándares internacionales como la *ISO/IEC 27001:2022* y el *NIST SP 800-53*.

2. El propósito principal de la **fase de Control** es definir un conjunto de medidas de seguridad y verificación que aseguren el cumplimiento riguroso de los estándares establecidos. Implica la selección de mecanismos que monitoreen la calidad del trabajo y la designación de personas específicas y responsables para las tareas asignadas. De esta forma, se garantiza que cada miembro del equipo tenga definido su rol para mantener la integridad del proceso.

Estándares y controles. Criterio de selección

La elección de *ISO/IEC 27001:2022* junto con *NIST SP 800-53* como marcos de seguridad para un *pipeline* DevSecOps se justifica debido a la cobertura integral y enfoque complementario que poseen respectivamente. La base normativa para el procedimiento de integración de seguridad en *pipelines* DevSecOps incluye los siguientes elementos específicos, cada uno con una función generalizada, según las normativas (Wilbur L. Ross, Jr., Secretary, 2020; *ISO/IEC 27001, 2022*)

ISO/IEC 27001/2022

- A.8.1: se enfoca en la gestión y seguridad de dispositivos de usuario finales.
- A.8.2: establece los requisitos para gestionar y controlar los accesos privilegiados dentro de la organización.
- A.12.1.3: establece los componentes necesarios para mantener la capacidad adecuada para garantizar que los servicios sean seguros y estén disponibles.
- A.8.23: implementa medidas para controlar y monitorear el acceso a contenido web.
- A.8.28: se centra en garantizar que el desarrollo de *software* se realice de manera segura.

NIST SP 800-53

- AC-2.7: establece un marco de autenticación único y consistente, lo que permite la integración automática en el *pipeline* mediante la automatización de pruebas de autenticación y la validación continua de identidades. Establece el control de acceso a los recursos.

- AC-3: aporta una capa adicional de seguridad mediante la gestión de acceso, lo que es esencial en entornos DevSecOps donde los cambios son frecuentes y necesitan ser consistentes.
- AC-6: proporciona un enfoque de gestión de privilegios mínimos, lo que permite la automatización de roles y permisos.
- AC-4: establece políticas de red seguras que pueden ser versionadas y aplicadas automáticamente, esto posibilita la segmentación de red y el control de tráfico en cada etapa del *pipeline*.
- AC-10: aporta un enfoque integral de gestión de eventos de seguridad que puede ser automatizado y monitoreado en tiempo real.

Estos controles son particularmente valiosos porque pueden ser automatizados y versionados, lo que permite su integración fluida en el *pipeline* CI/CD, a diferencia de otros controles que requieren intervención manual o que no son de manera directa aplicables a un entorno automatizado. Su implementación conjunta crea una capa de seguridad integral que abarca los aspectos principales de un entorno DevSecOps: autenticación, control de acceso, seguridad de red y monitoreo de eventos.

Estrategia de selección para los controles de seguridad

Para el desarrollo de la investigación se diseñó un protocolo de selección que facilitó el análisis de estándares y controles relacionados con la integración de medidas de seguridad en el *pipeline* DevSecOps en el clúster de k8s que utilizó Tekton. La combinación de estos criterios permitió seleccionar un conjunto de controles que no solo cumplen con los requisitos de seguridad, sino que también son implementables y mantenibles en un entorno DevSecOps moderno, donde la automatización y la velocidad de entrega son fundamentales.

Objetivo del análisis: identificar estándares y controles que describen métodos aplicables para la integración de mejores prácticas para el *pipeline* DevSecOps en Kubernetes. Además, identificar las técnicas de implementación seguras que se pueden integrar al *pipeline* DevSecOps.

Fuente de información: normativas gubernamentales o verificadas en el ámbito nacional. *ISO/IEC 27001* y *NIST SP 800-53*.

Estrategia de búsqueda: Controles que abarquen la implementación de prácticas de seguridad que complementen la robustez de Kubernetes.

Criterios de inclusión y exclusión: Inclusión: capacidad de automatización a través de la posibilidad de implementación mediante *scripts*. Alineación con medidas de seguridad establecidas en el marco regulatorio cubano. Integración con herramientas de automatización utilizadas. Capacidad de evolución con la organización según su adaptabilidad a diferentes entornos empresariales.

Procedimientos de revisión: Selección inicial, Evaluación detallada y Síntesis de resultados.

Responsables de la ejecución del procedimiento: El enfoque DevSecOps trasciende los equipos de desarrollo, operaciones y seguridad. Involucra grupos multidisciplinarios que incluyen equipos de control de calidad (QA), arquitectos de soluciones y especialistas en cumplimiento normativo. La colaboración entre estos grupos resulta fundamental para garantizar la integración de la seguridad en cada fase del ciclo de vida del desarrollo del *pipeline*.

El equipo DevSecOps es el responsable del proceso. Su función principal es garantizar la integridad y protección de los entornos de desarrollo y despliegue. Sus responsabilidades comprenden la definición y mantenimiento de políticas de seguridad, basadas en las normas *NIST SP 800-53*. Es el encargado de la implementación de controles automatizados en la plataforma Tekton, en conformidad con los estándares seleccionados.

Clientes finales: el proceso cuenta con diversos clientes que se dividen en dos grupos: internos, que participan activamente en el proceso DevSecOps; y los externos, que son los beneficiarios finales con una relación independiente del proceso. La implementación de *pipelines* en DevSecOps es fundamental porque integra de forma automática la seguridad en cada etapa del desarrollo de *software*, lo que ayuda a identificar y corregir fallos desde el inicio del proceso. Al automatizar las pruebas de seguridad y la evaluación de vulnerabilidades, las organizaciones pueden entregar *software* más seguro y de mayor calidad de manera más rápida.

Requisitos previos para la implementación de cada control: organizar y aislar los recursos de manera segura permite la creación de una base sólida para el desarrollo y despliegue de los controles seleccionados. A continuación se presentan los requisitos que aseguran que la implementación sea exitosa para crear y gestionar *pipelines*:

Requisito 1: Instalar y configurar Tekton: se debe comenzar con la ejecución de las instrucciones correspondientes a su documentación inicial. El objetivo es asegurar que el código y las dependencias cumplan con estándares de seguridad antes de generar artefactos.

Requisito 2: Configurar el entorno de trabajo

Salidas esperadas: Un *pipeline* de DevSecOps operativo que integre controles de seguridad automatizados, estableciendo un ciclo de retroalimentación continua para la mejora iterativa de las medidas de seguridad a lo largo de todas las etapas, desde el desarrollo hasta la producción.

Implementación en la etapa de construcción del *pipeline*

La etapa de construcción es fundamental para verificar la integridad del código y detectar vulnerabilidades de seguridad. Se lleva a cabo su desarrollo a través de los controles AC-3 y AC-2.7 de *NIST 800-53*.

Control AC-2(7) del *NIST SP 800-53* Esquema Basado en Roles (RBAC)

Objetivo: restringir accesos al *pipeline* de construcción (ejemplo: solo el equipo de DevSecOps puede modificar *Tasks*).

Implementación

• Crear 3 Roles en formato YAML para definir los roles esenciales (un rol de administrador total, un rol de desarrollador y un rol de solo lectura).

Control AC-3 del NIST SP 800-53 (Políticas de red)

Objetivo: garantizar que solo los recursos autorizados puedan interactuar con el *pipeline*, aplicando el principio de mínimos privilegios y seguridad por defecto. Crear *NetworkPolicies* para aislar los *pods*.

Implementación

• Crear *NetworkPolicies* para aislar el *pipeline*.

Implementación en la etapa de prueba del *pipeline*

La implementación del I*SO/IEC 27001:2022* como control de seguridad en el *pipeline* DevSecOps se realizó mediante una arquitectura integrada que comenzó con la definición de dos controles fundamentales: el A.8.1 para la identificación de activos mediante escaneo de secretos, conforme a la legislación cubana que recomienda a las entidades a implementar medidas de seguridad para la información crítica según su Artículo 17 del *Decreto-Ley 370/2018* y el A.8.2 para el control de acceso elevado a sistemas y datos en dependencias, siguiendo las disposiciones del *Decreto-Ley 360/2019* que requiere la gestión de riesgos en infraestructuras críticas según su Artículo 8 (Saber, 2018).

Control A.8.1 y A.8.2 del ISO/IEC 27001-2022 (Gestión de acceso y privilegios)

Objetivos: fortalecer la seguridad durante el desarrollo y despliegue de aplicaciones mediante la asignación controlada de derechos administrativos y el escáner de elementos privados o de acceso restringido.

Implementación:

- Crear *Task* para el escáner.

La tarea está diseñada para escanear secretos en un directorio de código fuente utilizando la herramienta Trivy de Aqua Security que ayuda a identificar vulnerabilidades y problemas de seguridad en diferentes componentes.

Implementación en la etapa de entrega del *pipeline*

Para establecer un entorno seguro en *Tekton Pipelines* en la etapa de entrega se implementaron varios aspectos de distintos controles del *NIST SP 800-53*, en cumplimiento con el marco legal cubano que establece controles de seguridad para entidades estatales, que incluyen la gestión de accesos y protección de datos en la *Resolución 105/2020*.

Control AC-3 AC-6 AC-4 AC-10 de NIST SP 800-53

Implementación

- Crear un *namespace* y una cuenta de servicio con permisos mínimos.

Se crea una cuenta de servicio (*ServiceAccount*) dentro del denominado *namespace* previamente creado. La cuenta de servicio será

utilizada por los *pods* para acceder a los recursos necesarios dentro del clúster. Al establecer esta cuenta de servicio, se facilita la ejecución de tareas automatizadas y la implementación de procesos de integración continua.

- Aplicar en el clúster con *kubectl* apply -f <service-account.yaml>
- Definir un rol que solo permita ejecutar *pipelines* sin permisos administrativos.

Implementación en la etapa de despliegue del pipeline

En los entornos Kubernetes para el despliegue se utilizan múltiples tecnologías y herramientas (como Docker, Helm, Istio, entre otros) que pueden variar entre organizaciones. En esta etapa se proponen mejores prácticas de seguridad, según las características necesarias de cada proyecto o *software* desplegado en función de la organización. Recomendar estándares generales permite que cada equipo elija las herramientas más adecuadas para su contexto sin limitarse a una solución específica que puede no ser inclusiva para sus necesidades. La etapa de despliegue en un *pipeline* DevSecOps es crítica para garantizar la seguridad operacional, donde se aplicarán controles específicos del estándar *ISO/IEC 27001:2022*. Los controles seleccionados abarcan el A.12.1.3 (gestión de capacidad), A.8.23 (seguridad en servicios en red) y A.8.28 (protección en sistemas públicos).

Control A.12.3 del ISO/IEC 27001:2022 (Gestión de capacidad)

El control establece que el uso de los recursos debe monitorizarse, ajustarse y proyectarse para anticipar necesidades futuras, garantizando así el rendimiento requerido del sistema. Asimismo, enfatiza en la importancia de escanear y evaluar los componentes del sistema para identificar vulnerabilidades conocidas. Para implementarlo se recomienda añadir pasos en el *Task* de despliegue que cumplan funciones propuestas.

Objetivo: Limitar CPU por *pod* para evitar consumo excesivo.

Control A.8.23 del ISO/IEC 27001:2022 (Seguridad en red)

El control se enfoca en garantizar que las comunicaciones dentro de los sistemas estén protegidas contra accesos no autorizados, ataques y exposición de datos sensibles.

Objetivo: Restringir tráfico entre *pods* (solo *frontend* a *backend*).

Control A.8.28 del ISO/IEC 27001:2022 (Protección en sistemas públicos)

El control se refiere a la gestión de configuraciones y cambios en sistemas de información. Establece que las configuraciones de los sistemas deben ser documentadas, controladas y revisadas con regularidad para garantizar que no introduzcan vulnerabilidades. Además, los cambios en las configuraciones deben ser evaluados, probados y aprobados antes de su implementación para minimizar riesgos.

Objetivo: Asegurar comunicaciones con HTTPS (*Hyper Text Transfer Protocol Secure*), y evitar datos en claro. Forzar HTTPS en *Ingress* redirigiendo HTTP a HTTPS y usar HSTS.

Estrategia de validación del procedimiento propuesto

La validación de procedimientos es fundamental en el campo de la seguridad informática, ya que es un proceso riguroso que asegura que los métodos establecidos cumplan de forma efectiva con los objetivos propuestos. A través de este procedimiento, se verifica la capacidad de cada elemento, se identifican posibles fallos y se corrigen, demostrando su idoneidad para proteger los activos informáticos. Para esta investigación, se emplearon dos enfoques: simulación y métricas.

La Figura 2 detalla el procedimiento para desarrollar **la simulación en VirtualBox**. Esta guía describe cada etapa del proceso con el fin de asegurar la correcta aplicación de las técnicas y lograr una validación exitosa en un entorno real.

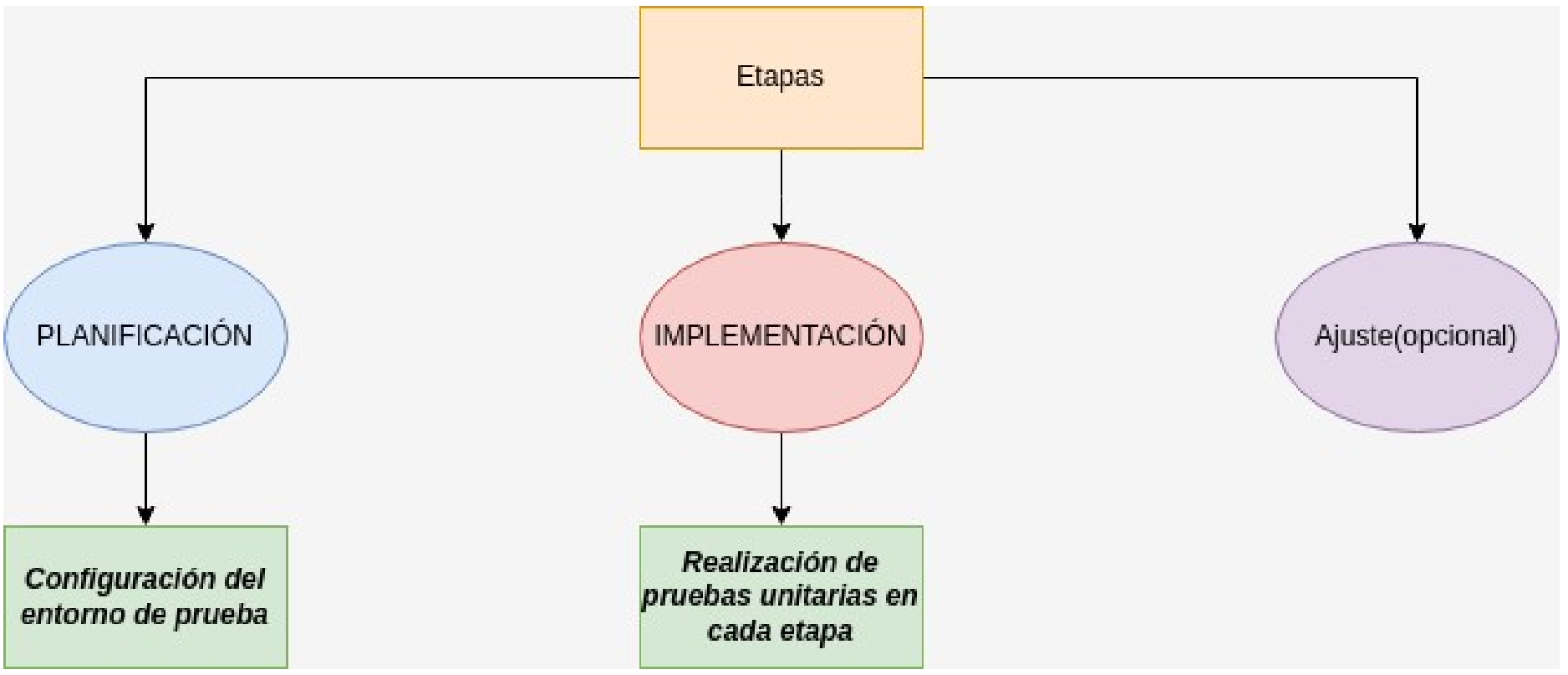

Figura 2. Etapas que componen el procedimiento empleado en la estrategia de validación simulación

Las dos etapas principales, junto con una fase opcional, facilitan el establecimiento de objetivos claros, la identificación de los recursos necesarios y la definición de criterios de éxito. La fase de ajuste incluye la corrección de hallazgos identificados, ajuste de configuraciones según resultados, identificación de áreas de mejora, documentación de cambios implementados y priorizar acciones correctivas según la necesidad empresarial.

Las fases materializan el procedimiento en acciones concretas. Esto facilita la reproducción controlada de escenarios de seguridad y la verificación práctica de los procedimientos. Además, contribuyen al análisis de los resultados obtenidos y generan conclusiones relevantes que sirven de base para recomendaciones orientadas a la mejora continua. La secuencia lógica de las etapas asegura que la simulación sea un proceso riguroso, en el que cada fase se apoya en la anterior para producir resultados confiables y aplicables.

El uso de **métricas** para determinar el cumplimiento normativo es fundamental para garantizar que la organización cumpla con los estándares de seguridad y regulaciones aplicables, lo que facilita la identificación de brechas y la mejora continua en las políticas de seguridad. Maverix es una solución clave para implementar DevSecOps debido a su capacidad de automatizar y centralizar métricas críticas de seguridad, integrando distintas herramientas en los *pipelines* CI/CD. Su modelo cuantificable permite priorizar vulnerabilidades, reducir riesgos y demostrar cumplimiento normativo, mientras fomenta una cultura de seguridad ágil (Maverix, 2021).

Maverix calcula el CP (cumplimiento de política) mediante la fórmula $CP = \frac{PC}{PT} \times 100$, donde PC son los puntos de control de seguridad implementados y PT el total de controles requeridos según estándares reconocidos. Esta métrica permite evaluar brechas de seguridad (ejemplo: un CP del 80 % indica un 20 % de controles pendientes) y priorizar su implementación para cumplir con normativas o mejorar la postura de seguridad. Se determina que el umbral de cumplimiento aceptable se establece cuando CP es mayor o igual al 75 %.

El umbral de cumplimiento del 75 % fue seleccionado como aceptable, en línea con recomendaciones de diferentes marcos internacionales:

• Según la sección 6.1.2 de la norma *ISO/IEC 27001*, los controles de seguridad pueden ajustarse según el nivel de riesgo.

• El marco del NIST CSF clasifica los niveles de madurez en cuatro categorías (Tier 1 a Tier 4), donde Tier 3 («Consistente») indica que la mayoría de los controles relevantes están en práctica. Un cumplimiento del 75 % puede corresponder con un nivel de madurez Tier 3, lo que significa que la organización tiene una postura de seguridad estable, pero aún no alcanza la excelencia, que sería Tier 4.

• Los *CIS Controls* (conjunto de mejores prácticas diseñadas para ayudar a las organizaciones a mejorar su ciberseguridad) priorizan los controles básicos y fundamentales. Para sistemas de baja criticidad, se recomienda implementar al menos el 75 % de los controles básicos.

Desde una perspectiva económica, alcanzar el 100 % de cumplimiento implicaría un incremento sustancial en los costos, sin ofrecer beneficios proporcionales en la reducción de riesgos. Por lo tanto, se concluye que un nivel de cumplimiento entre el 75 % y el 80 % es común en entornos similares (de prueba o de baja criticidad), lo que respalda la elección de este umbral como una opción razonable y práctica.

Conclusiones

Como resultado de la investigación realizada, se concluye que:

La comprensión de los conceptos fundamentales de DevSecOps es crucial para implementar prácticas de seguridad efectivas en el desarrollo de *software*. El análisis detallado de los estándares y mejores prácticas permite establecer un marco teórico-metodológico sólido que sustenta la implementación exitosa de procedimientos de seguridad, lo que garantiza la integridad y confiabilidad del proceso de desarrollo.

La implementación de estándares y mejores prácticas de seguridad en *pipelines* DevSecOps es fundamental para proteger la infraestructura empresarial y garantizar la integridad del *software*. El procedimiento

estructurado brindado permite identificar y mitigar vulnerabilidades para fortalecer la seguridad de la organización mediante un enfoque sistemático y controlado.

La validación de la propuesta consolida su credibilidad por medio de un enfoque reflexivo que permite ajustar los resultados según los datos obtenidos, lo que contribuye al avance del conocimiento científico. Las simulaciones efectuadas validaron que el flujo de trabajo es sólido y que el procedimiento alcanza los objetivos planteados, y esto asegura su aplicabilidad en la práctica y su capacidad para generar mejoras.

Integrar estándares y mejores prácticas de seguridad en el *pipeline* DevSecOps es fundamental para proteger proactivamente el *software* desde las etapas iniciales del desarrollo. Permite identificar y abordar vulnerabilidades antes de que se conviertan en problemas costosos, lo que asegura el cumplimiento normativo y mejorando la calidad del *software*.

Referencias bibliográficas

Antoniotti, L. (2021). *Tesi di Laurea Magistrale Pipeline DevSecOps* [Máster en Ingeniería Informática, POLITECNICO DI TORINO]. Electronico. http://webthesis.biblio.polito.it/id/eprint/20537

Atlassian. (2025, febrero 20). *What is SDLC? Software Development Life Cycle Explained*. Atlassian. https://www.atlassian.com/agile/software-development/sdlc

Chandramouli, R., Kautz, F., & Torres-Arias, S. (2024). *Strategies for the integration of software supply chain security in DevSecOps CICD pipelines*: (No. NIST SP 800-204D; Número NIST SP 800-204D, p. NIST SP 800-204D). National Institute of Standards and Technology (U.S.). https://doi.org/10.6028/NIST.SP.800-204D

GitLab 2024 *Global DevSecOps Report* | GitLab. (2025, febrero 26). Nuxt-App. https://about.gitlab.com/developer-survey/

ISO/IEC27001:2022.(2022).ISO.https://www.iso.org/standard/27001

Kubernetes. (2024). *Overview*. Kubernetes. https://kubernetes.io/docs/concepts/overview/

Madnick, P. S. E. (2023). *The Continued Threat to Personal Data: Key Factors Behind the 2023 Increase*.

Maverix. (2021). DevSecOps automation platform: Metrics-driven security compliance. Maverix Security Solutions. https://www.maverix.com/resources/devsecops-automation-whitepaper

Muñoz, C. D. (2023). DevOps en el desarrollo de software: Integración y entrega continua. *Polo del Conocimiento: Revista científico-profesional*, *8*(9 (SEPTIEMBRE 2023)), 603-615.

Saber, H. (2018). *Consejo de Estado*.

Tekton. (2023, noviembre 13). *How it all started The Tekton Story—Tekton*. https://blog.tektonlabs.com/tekton-labs-blog/how-it-all-started-the-tekton-story/

Wilbur L. Ross, Jr., Secretary. (2020). *Security and Privacy Controls for Information Systems and Organizations*.

Xygeni. (2024). *Mejores prácticas de CI/CD: Transformando el desarrollo de software*| xygeni. https://xygeni.io/es/blog/cicd-best-practices-transforming-software-development/

Zayni, M., & Nassereddine, B. (2020). (PDF) *DevSecOps Practices for an agile and secure it service management*. https://www.researchgate.net/publication/338659928_DevSecOps_PRACTICES_FOR_AN_AGILE_AND_SECURE_IT_SERVICE_MANAGEMENT

Zhu, B., & Zou, E. (2021). *Design of intelligent bedroom system based on pure off-line speech recognition technology*. https://ieeexplore.ieee.org/document/9407425/references#references