Maestría en

CIBERSEGURIDAD

Trabajo previo a la obtención de título de
Magister en Ciberseguridad

AUTORES:

AYOVI ANGULO JACKSON NERY

LLUMIHUASI QUISPE JUAN MIGUEL

PARDO ALDAS JUAN CARLOS

YEPEZ HERDOIZA JUAN PABLO

TUTORES:

Paulina Vizcaíno

Iván Reyes Chacón

TEMA

Diseño e implementación de un pipeline de DevSecOps para el análisis automático de vulnerabilidades en imágenes Docker

Quito - Ecuador

Junio – 20206

REINVENTEMOS EL FUTURO

Certificación de autoría

Nosotros, **Jackson Nery Ayovi Angulo, Juan Miguel Llumihuasi Quispe, Juan Carlos Pardo Aldas y Juan Pablo Yépez Herdoiza,** declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido presentado anteriormente para ningún grado o calificación profesional y que se ha consultado la bibliografía detallada.

Cedemos nuestros derechos de propiedad intelectual a la Universidad Internacional del Ecuador (UIDE), para que sea publicado y divulgado en internet, según lo establecido en la Ley de Propiedad Intelectual, su reglamento y demás disposiciones legales.

Firma
Jackson Nery Ayovi Angulo

Firma
Juan Miguel Llumihuasi Quispe

Firma
Juan Carlos Pardo Aldas

Firma
Juan Pablo Yépez Herdoiza

Autorización de Derechos de Propiedad Intelectual

Nosotros, **Jackson Nery Ayovi Angulo, Juan Miguel Llumihuasi Quispe, Juan Carlos Pardo Aldas y Juan Pablo Yépez Herdoiza**, en calidad de autores del trabajo de investigación titulado ***Diseño e implementación de un pipeline de DevSecOps para el análisis automático de vulnerabilidades en imágenes Docker***, autorizamos a la Universidad Internacional del Ecuador (UIDE) para hacer uso de todos los contenidos que nos pertenecen o de parte de los que contiene esta obra, con fines estrictamente académicos o de investigación. Los derechos que como autores nos corresponden, lo establecido en los artículos 5, 6, 8, 19 y demás pertinentes de la Ley de Propiedad Intelectual y su Reglamento en Ecuador.

D. M. Quito, julio de 2026

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
JACKSON NERY AYOVI ANGULO

Firma
Jackson Nery Ayovi Angulo

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
JUAN MIGUEL LLUMIHUASI QUISPE

Firma
Juan Miguel Llumihuasi Quispe

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
JUAN CARLOS PARDO ALDAS

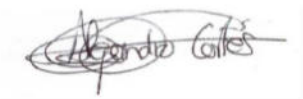
Firma
Juan Carlos Pardo Aldas

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
JUAN PABLO YEPEZ HERDOIZA

Firma
Juan Pablo Yépez Herdoiza

APROBACIÓN DE DIRECCIÓN Y COORDINACIÓN DEL PROGRAMA

Nosotros, Alejandro Cortés López e Iván Galo Reyes Chacón, declaramos que: Jackson Nery Ayovi Angulo, Juan Miguel Llumihuasi Quispe, Juan Carlos Pardo Aldas y Juan Pablo Yépez Herdoiza, son los autores exclusivos de la presente investigación y que esta es original, auténtica y personal de ellos.



Alejandro Cortés
Director/a de la
Maestría en Ciberseguridad

Iván Reyes
Coordinador/a de la
Maestría en Ciberseguridad

DEDICATORIA

Jackson Nery Ayovi Angulo:

Dedico este Trabajo Fin de Máster, en primer lugar, a Dios, por darme la fortaleza y la sabiduría para alcanzar esta meta.

A mi familia y amigos, por su cariño, apoyo incondicional y palabras de aliento durante este camino. Su confianza y motivación fueron fundamentales para hacer posible este logro.

Juan Miguel Llumihuasi Quispe:

Porque cada hora robada al tiempo contigo ha valido la pena si sirve para enseñarte que con esfuerzo todo es posible. Este trabajo es para ti, César Julián, con todo mi amor.

Juan Carlos Pardo Aldas:

Con todo mi amor, a mi esposa, por caminar siempre a mi lado, creer en mí incluso en los momentos más difíciles y ser mi apoyo. A mis hijos, quienes son la razón de cada esfuerzo y la mayor inspiración para seguir creciendo como persona y como profesional. Todo lo que hago busca dejarles el mejor ejemplo: que con fe, dedicación y perseverancia no existen metas imposibles de alcanzar. Este logro también es de ustedes.

Juan Pablo Yépez Herdoiza:

Dedico este Trabajo Fin de Máster, en primer lugar, a Dios, por brindarme salud, fortaleza y perseverancia para alcanzar esta meta.

A mi familia, por su amor incondicional, apoyo constante y confianza en cada etapa de mi formación. Gracias por ser mi inspiración y el motor que me impulsa a seguir creciendo personal y profesionalmente.

AGRADECIMIENTOS

Jackson Nery Ayovi Angulo:

A Dios, por su guía, fortaleza y bendiciones, que hicieron posible alcanzar esta meta.

A Sugey Guilcapi y a mis queridos hijos, por su amor, comprensión y apoyo incondicional. Ustedes son mi mayor motivación y la razón de este logro.

A mis compañeros de grupo, por el trabajo en equipo, el apoyo y los conocimientos compartidos durante este proceso académico.

Juan Miguel Llumihuasi Quispe:

Quiero agradecer profundamente a mi esposa y a mi hijo. Gracias por su paciencia infinita durante los meses de estudio y por recordarme siempre la importancia de este objetivo. Su impulso constante fue la fuerza que necesité para llegar a la meta. Gracias por estar siempre a mi lado.

Juan Carlos Pardo Aldas:

En primer lugar, agradezco a Dios, por ser mi guía, darme fortaleza y sabiduría para superar cada desafío presentado.

A mi esposa e hijos, por su amor incondicional, paciencia y comprensión durante los momentos ausentes que el estudio y el desarrollo de este proyecto demandaron gran parte de mi tiempo. Su apoyo y motivación fueron el impulso para no rendirme.

A mis padres y hermanos, por sus enseñanzas, valores y constante respaldo. Gracias por creer en mí, por sus palabras de aliento sin ustedes no tendría el crecimiento personal y profesional.

A mis compañeros un gracias por el compromiso, la colaboración. El trabajo en equipo fue fundamental para alcanzar los objetivos planteados.

Finalmente, expreso mi sincero agradecimiento a la Universidad Internacional del Ecuador (UIDE), a sus docentes y autoridades, por brindar una formación de excelencia y los conocimientos necesarios para culminar con éxito esta nueva etapa en mi carrera profesional.

Juan Pablo Yépez Herdoiza:

Quiero expresar mi más sincero agradecimiento a todas las personas e instituciones que hicieron posible la realización de este Trabajo Fin de Máster.

En primer lugar, agradezco a Dios por brindarme la fortaleza, la sabiduría y la perseverancia necesarias para superar cada desafío y alcanzar esta importante meta académica.

A mi familia, por su amor incondicional, apoyo constante y comprensión durante todo este proceso. Su confianza en mis capacidades ha sido una fuente permanente de motivación para seguir adelante.

RESUMEN

El presente Trabajo diseña e implementa un pipeline de DevSecOps automatizado con la finalidad de realizar un análisis continuo de vulnerabilidades en imágenes de contenedores Docker, mitigando los riesgos de seguridad en arquitecturas cloud-native antes de su paso a producción. Ante el vacío técnico existente en organizaciones con infraestructuras reguladas que no pueden externalizar datos a nubes públicas, se propone una arquitectura híbrida que descentraliza las operaciones: utiliza GitHub Actions en la nube como plano de control y orquestación, y un entorno local (on-premises) virtualizado en VMware bajo el sistema operativo Rocky Linux como plano de ejecución (Self-Hosted Runner). El núcleo de seguridad está conformado por las herramientas de código abierto Aqua Trivy que realiza las detecciones de malas configuraciones estructurales y fallos del sistema operativo y Anchore Grype para realizar el análisis de composición de software (SCA). Los controles de calidad se basan en el estándar de criticidad CVSS v3.1, generando políticas de bloqueo frente a riesgos altos.

La solución demostró su efectividad cuantitativa en un escenario inicial multiproyecto basado en una imagen obsoleta de Ubuntu 20.04 y microservicios en Python/Flask. Durante la ejecución, el pipeline capturó exitosamente las alertas estructurales de fin de soporte de la distribución base y catalogó en un tiempo de ejecución local de 3 minutos y 12 segundos más de 200 vulnerabilidades activas.

En consecuencia, la implementación de una arquitectura híbrida en el pipeline DevSecOps permite integrar con éxito la agilidad del desarrollo y el control estricto de la infraestructura. El mantener el plano de control en la nube (**GitHub Actions**) y el plano de ejecución local (*Self-Hosted* sobre **VMware/Rocky Linux**) permite automatizar el ciclo de vida del software maximizando el rendimiento del hardware preexistente de la organización.

Asimismo, a través de este modelo descentralizado se presenta un mecanismo viable para garantizar la **soberanía de los datos**, asegurando que los metadatos de código, inventarios de dependencias y reportes de seguridad se procesen y retengan de forma inmutable dentro del centro de datos local.

Palabras Claves: DevSecOps, Docker, análisis de vulnerabilidades, seguridad de contenedores, CI/CD, automatización de seguridad, NIST, SBOM, CVE, YAML, CVSS v3.1, LOPDP, OWASP, Anchore, GitHub Actions, SARIF, self-hosted runner, Dockerfile, workflow, VMware, Aqua Trivy, Anchore Grype

ABSTRACT

This paper designs and implements an automated DevSecOps pipeline to perform continuous vulnerability analysis on Docker container images, mitigating security risks in cloud-native architectures before their deployment to production. Given the existing technical gap in organizations with regulated infrastructures that cannot outsource data to public clouds, a hybrid architecture is proposed that decentralizes operations: it uses GitHub Actions in the cloud as a control and orchestration plane, and a virtualized on-premises environment in VMware under the Rocky Linux operating system as an execution plane (Self-Hosted Runner). The security core consists of the open-source tools Aqua Trivy, which detects structural misconfigurations and operating system failures, and Anchore Grype, which performs software composition analysis (SCA). Quality controls are based on the CVSS v3.1 criticality standard, generating blocking policies for high-risk applications.

The solution demonstrated its quantitative effectiveness in an initial multi-project scenario based on an outdated Ubuntu 20.04 image and Python/Flask microservices. During execution, the pipeline successfully captured end-of-support structural alerts for the base distribution and cataloged over 200 active vulnerabilities in a local runtime of 3 minutes and 12 seconds.

Consequently, implementing a hybrid architecture in the DevSecOps pipeline successfully integrates development agility with tight infrastructure control. Maintaining the control plane in the cloud (GitHub Actions) and the local execution plane (self-hosted on VMware/Rocky Linux) automates the software lifecycle while maximizing the performance of the organization's existing hardware. Furthermore, this decentralized model presents a viable mechanism to guarantee data sovereignty, ensuring that code metadata, dependency inventories, and security reports are processed and retained immutably within the local data center.

Keywords: DevSecOps, Docker, vulnerability analysis, container security, CI/CD, security automation, NIST, SBOM, CVE, YAML, CVSS v3.1, LOPDP, OWASP, Anchore, GitHub Actions, SARIF, self-hosted runner, Dockerfile, workflow, VMware, Aqua Trivy, Anchore Grype

Indice

CAPÍTULO 1

1. Introducción

1.1. Definición del proyecto

El presente Trabajo Fin de Máster (TFM) tiene como finalidad el diseño e implementación de un pipeline de Development, Security and Operations (DevSecOps) orientado al análisis automático de vulnerabilidades en imágenes Docker, integrando herramientas de escaneo de seguridad dentro del ciclo de vida del desarrollo de software. La solución se integrará en un flujo de Integración y Despliegue Continuo (CI/CD) utilizando GitHub Actions como orquestador principal en la nube, este se integrará con un servidor self-hosted runner Rocky Linux 10 instalado on-premise, con base en esta infraestructura se integrará el pipeline DevSecOps que incluirán las herramientas Trivy y Grype encargadas de realizar el análisis de composición de software y las auditorias estructurales sobre las imágenes Docker que llevarán empaquetadas aplicaciones web demo en Python (Flask)

El proyecto se basa en la filosofía DevSecOps, que promueve la integración de la seguridad como un componente transversal que inicia en las primeras etapas del desarrollo del software (shift-left security) conforme a los estándares del Institute of Electrical and Electronics Engineers (IEEE). En este contexto, se diseñará un flujo automatizado alineado con la norma National Institute of Standards and Technology (NIST) Special Publication 800-190 (Application Container Security Guide) el cual exige que se realice la inspección y mitigación de riesgos críticos en la aplicación de contenedores. Logrando reducir la superficie de ataque en entornos basados en microservicios y arquitecturas cloud-native.

1.2. Justificación e importancia del trabajo de investigación

El uso de contenedores, especialmente a través de plataformas como Docker, se ha convertido en un estándar en el desarrollo y despliegue de aplicaciones modernas. Sin

embargo, las imágenes Docker pueden contener vulnerabilidades heredadas de sistemas base, dependencias obsoletas o configuraciones inseguras.

Diversos informes de ciberseguridad como la firma Karspersky y Sysdig Cloud-Native Security Report indican que una gran parte de las imágenes disponibles en repositorios públicos presentan vulnerabilidades críticas. Esto supone un riesgo significativo para organizaciones que adoptan metodologías ágiles y despliegue continuo.

- La incorporación de prácticas DevSecOps permite:
- Detectar vulnerabilidades de forma temprana (shift-left security).
- Reducir costes asociados a fallos de seguridad en producción.
- Mejorar la trazabilidad y cumplimiento normativo.
- Automatizar controles de seguridad sin frenar el ritmo de desarrollo.

Este trabajo es relevante porque resuelve la necesidad de implementar controles automatizados previo al despliegue en los entornos on-premises, con esto se aporta la necesidad de varias instituciones públicas de realizar los análisis sin subir datos a las nubes públicas.

El uso de herramientas sin costo como Trivy y Grype integrado con la visión del modelo Self-Hosted permiten a las instituciones fortalecer su postura de seguridad en entornos de integración y despliegue continuo además de reducir costos de implementación.

Alcance

El alcance del TFM incluye:

El proyecto se desarrollará bajo una arquitectura híbrida permitiendo desacoplar el plano de ejecución del plano de control mediante la combinación de nube pública (GitHub) e infraestructura on-premises (VMware)

Infraestructura: Se utilizará un entorno local basado en VMware para la simulación de red y servidor Self-Hosted Runner (Rocky Linux 10), combinado con GitHub Actions para la orquestación del flujo del pipeline.

Herramientas de Escaneo: Se han preseleccionado Trivy y Grype para el análisis de vulnerabilidades, ya que estas herramientas cuentan con una amplia base de datos (Common Vulnerabilities and Exposures (CVE), National Vulnerability Database (NVD)), velocidad de ejecución y facilidad de integración a través de archivos Yet Another Markup Language (YAML) justificada a través de los estudios del Cloud Native Computing Foundation (CNCF).

Fuera del alcance

- ✓ Implementación en entorno experimentales y no productivo.
- ✓ Desarrollo de nuevas herramientas de escaneo.
- ✓ Auditoría completa de infraestructuras empresariales.

1.3. Objetivos

1.3.1. Objetivo general

Diseñar e implementar un pipeline DevSecOps automatizado que permita detectar y gestionar vulnerabilidades en imágenes Docker durante el ciclo de integración continua, mejorando la seguridad en entornos de desarrollo basados en contenedores.

1.3.2. Objetivo específico

- Evaluar y caracterizar técnicamente herramientas de escaneo de vulnerabilidades para imágenes Docker como Trivy y Grype, evaluando su funcionamiento, cobertura, bases de datos utilizadas y capacidades de integración en pipelines CI/CD.
- Comparar la efectividad de la solución implementada, considerando:
    - Cantidad de fallos detectados por imagen y nivel de severidad.
    - Impacto temporal del análisis sobre el tiempo total del pipeline.
    - Impacto del uso de contenedores con versiones sin soporte y versiones con soporte
- Diseñar la arquitectura de un pipeline DevSecOps que incorpore controles de seguridad automatizados para el análisis de imágenes Docker dentro de procesos de integración continua.
- Implementar el pipeline propuesto en un entorno de pruebas, integrando herramientas de escaneo y configurando políticas de seguridad basadas en el estándar Common Vulnerability Scoring System (CVSS).
- Aplicar la solución a un conjunto representativo de imágenes Docker (públicas y privadas), obteniendo métricas cuantitativas sobre número, tipo y criticidad de vulnerabilidades detectadas.
- Formular recomendaciones para el manejo de imágenes privadas, asegurando el cumplimiento de la Ley Orgánica de Protección de Datos Personales (LOPDP) respecto a la trazabilidad y acceso a metadatos de los desarrolladores.

CAPÍTULO 2

2. Revisión de literatura

2.1. Estado del arte

2.1.1. Eficacia y tasas de detección de Trivy vs. Grype

La seguridad en contenedores adquirió gran importancia en entornos Development, Security and Operations (DevSecOps) debido al crecimiento de Docker y Kubernetes, promoviendo el uso de herramientas especializadas para detectar vulnerabilidades antes del despliegue de aplicaciones. Entre las herramientas más utilizadas destacan Trivy y Grype, ambos escáneres de código abierto orientados a detectar vulnerabilidades antes del despliegue de aplicaciones Trivy utiliza fuentes como National Vulnerability Database (NVD), GitHub Security Advisories y repositorios Linux oficiales para identificar fallas tanto en paquetes del sistema operativo como en dependencias de lenguajes de programación; por otro lado, Grype suele detectar vulnerabilidades adicionales en proyectos complejos, aunque puede generar una mayor cantidad de alertas que requieren validación manual.

Actualmente, muchas organizaciones utilizan ambos escáneres de manera complementaria para ampliar la cobertura de análisis y disminuir riesgos en producción.

2.1.2. El dilema de los falsos positivos y triaje automatizado

El aumento de falsos positivos representa una de las principales limitaciones de los escáneres de vulnerabilidades modernos, ya que existen sistemas que generan alertas sobre componentes cuya explotación no es viable dentro del entorno analizado, provocando una carga operativa adicional para los equipos de seguridad. Ante esta situación surgieron mecanismos de triaje automatizado enfocados en priorizar vulnerabilidades según el contexto operativo, considerando factores como el proceso de la evaluación de riesgo del activo, nivel de exposición y condiciones reales de explotación. Dentro de estas soluciones destaca el

estándar Vulnerability Exploitability eXchange (VEX), un mecanismo diseñado para comunicar el estado de explotación y el impacto de vulnerabilidades en productos específicos. VEX se encuentra formalizado como parte de la especificación Common Security Advisory Framework (CSAF) versión 2.0, desarrollada por la Organization for the Advancement of Structured Information Standards (OASIS) y reconocida por organismos como Cyberseguirity and Infrastructure Security Agency (CISA). Asimismo, su integración con los Software Bill of Materials (SBOM) permite correlacionar componentes de software con vulnerabilidades conocidas, facilitando la reducción de falsos positivos y la priorización de riesgos según el contexto operativo.

Además, VEX puede integrarse con tecnologías como SBOM y pipelines de Integración y Despliegue Continuo (CI/CD) automatizados para mejorar la trazabilidad y reducir alertas innecesarias. Diversos estudios coinciden en que el objetivo actual no consiste únicamente en detectar más vulnerabilidades, sino en identificar aquellas que representan un riesgo real para la organización.

2.1.3. Optimización del tiempo de ejecución (Pipeline Bottlenecks)

Los últimos benchmarks entre los años 2024 y 2026 muestran que, si bien Trivy 0.50 presenta un excelente rendimiento general y menor consumo de recursos en análisis iniciales, Grype 0.70 logra ser entre un 15% y 40% más rápido en entornos de escaneo continuo y repetido dentro de los pipelines DevSecOps.

Según estudios de caso documentados, el impacto cuantitativo en pipelines CI/CD es un factor decisivo para el análisis de la viabilidad de implementaciones DevSecOps.

Los pipelines bottlenecks críticos identificados son:

- Build times: Tiempos de compilación o tiempos de construcción (el tiempo que tarda

el código fuente en convertirse en un artefacto ejecutable).

- Context switching: Cambio de contexto (la pérdida de productividad o el esfuerzo mental al alternar entre diferentes tareas o procesos).
- P99: Percentil 99 (métrica de rendimiento que indica que el 99% de las solicitudes se procesan en ese tiempo o menos, descartando valores atípicos extremos).

1. Build times largos (15-30 min) causan context-switching y pérdida de momentum en los equipos de desarrollo

2. Escaneos de seguridad no optimizados que representar un máximo de 42% del costo de computo

3. La p99 del tiempo de escaneo en pipelines puede reducirse de 14 min a 12.1 min con migración a Grype

2.1.4. Seguridad en la cadena de suministro y generación de SBOM

El SBOM actualmente es un requisito regulatorio a partir de la Orden Ejecutiva 14028 de los EEUU (National Institute of Standards and Technology Special Publication (NIST SP) 800-218) y de la Cyber Resilience Act (CRA) de la Unión con implicaciones legales y penales para organizaciones.

Software Package Data Exchange (SPDX) es mantenido por la Linux Foundation desde 2010 mantiene un enfoque original en licencia y cumplimiento jurídico. CycloneDX es mantenido por Open Worldwide Application Security Project (OWASP) y cuenta con enfoque en seguridad aplicada y administración de las vulnerabilidades.

Al combinar Grype para escaneo rápido **y** SBOM obligatorio en SPDX/CycloneDX se garantiza un alto rendimiento y seguridad de cadena de suministro para pipelines DevSecOps.

2.1.5. Estandarización de reportes mediante formato SARIF

El formato Static Analysis Results Interchange Format (SARIF) se ha consolidado como un estándar fundamental para la interoperabilidad entre herramientas de análisis estático y plataformas modernas de desarrollo como GitHub. Gracias a su estructura basada en JavaScript Object Notation (JSON), SARIF permite centralizar y visualizar alertas de seguridad directamente en los repositorios, facilitando la automatización de procesos DevSecOps y mejorando la trazabilidad de vulnerabilidades dentro de los pipelines CI/CD. Además, este estándar favorece la integración entre múltiples herramientas de escaneo, eliminando la dependencia de formatos propietarios y optimizando la gestión de resultados de seguridad.

2.1.6. Limitaciones de las GitHub Actions oficiales en el Marketplace

Las GitHub Actions oficiales más utilizadas para el análisis de vulnerabilidades en imágenes de contenedores son Trivy Action, desarrollada por Aqua Security, y Anchore Scan Action, desarrollada por Anchore. Estas soluciones permiten integrar de forma rápida el análisis de seguridad dentro de los flujos de CI/CD de GitHub, facilitando la detección automática de vulnerabilidades conocidas durante el proceso de desarrollo.

Sin embargo, presentan ciertas limitaciones. Por ejemplo, la Trivy Action está orientada principalmente a la ejecución estándar del escáner y a la generación de reportes, ofreciendo opciones limitadas para realizar correlación de resultados con otras herramientas de seguridad dentro del mismo flujo de trabajo. De manera similar, Anchore Scan Action se enfoca en la identificación de vulnerabilidades y cumplimiento de políticas, pero no incorpora mecanismos nativos para combinar resultados de múltiples escáneres o aplicar reglas de decisión personalizadas más allá de las configuraciones definidas por la herramienta.

Frente a estas restricciones, los scripts personalizados permiten implementar procesos más flexibles, tales como la ejecución secuencial de Trivy y Grype, la consolidación de hallazgos en un único reporte SARIF, la aplicación de umbrales específicos de criticidad y la integración con plataformas corporativas de monitoreo o gestión de incidentes. Esta capacidad de adaptación resulta especialmente relevante en entornos empresariales donde las políticas de seguridad requieren controles adicionales que no están disponibles de forma predeterminada en las Actions oficiales.

Diversos estudios y la documentación técnica de Aqua Security (2025) y Anchore (2025) indican que las GitHub Actions oficiales están diseñadas para simplificar la adopción de prácticas DevSecOps, mientras que las organizaciones con mayores requerimientos de seguridad suelen extender estas capacidades mediante automatizaciones propias para lograr una mayor personalización y control del proceso.

2.1.7. Políticas de Build Breakers en Pipelines Empresariales

Las políticas de build breakers forman parte de un mecanismo de control dentro de los pipelines de integración y entrega continua (CI/CD), encaminadas a garantizar la seguridad, calidad y cumplimiento del software durante su ciclo de vida. Su función principal se basa en interrumpir automáticamente los procesos de construcción o despliegue, cuando existen errores críticos, se declaran infracciones o vulnerabilidades de políticas organizacionales.

La integración con herramientas y automatización se componen con herramientas de escaneo de vulnerabilidades como Trivy, Grype o Snyk, Se bloquearía cuando se detectan claves, tokens o contraseñas en el código, librerías con licencias incompatibles, por esta razón el pipeline se detiene por razones de seguridad, tal como con motores de políticas establecidos en código, como HashiCorp Sentinel Open Policy Agent (OPA) y plataformas DevSecOps, son los que toman la decisión.

2.1.8. Soluciones alternativas frente a Trivy/Grype

Las Herramientas Trivy y Grype son fáciles de integrar y populares en entornos CI/CD y rapidez en el escaneo, existen varias alternativas como: Snyk, Aqua Security, Anchore, Qualys, Prisma Cloud, DevSecOps, JFrog Xray, que visualización completa de riesgos, cumplimiento a la normativa, priorización por vulnerabilidades, sin embargo, no son escogidas por costo, compleja integración, dependencia fuerte en ecosistemas comerciales.

Las tendencias actuales en el análisis de vulnerabilidades se reflejan en la evolución de abordar la seguridad:

El SBOM se está transformando en un elemento esencial para conocer y identificar los componentes de software, ayudando en el cumplimiento de la normativa, detectar vulnerabilidades y los requisitos regulatorios.

Actualmente las herramientas integran factores como el impacto en el entorno especifico, probabilidad real de explotación y el nivel de exposición.

La integración de DevSecOPS son las soluciones modernas que buscan integrarse por completo en el ciclo de CI/CD, incluyendo desde fases tempranas en el desarrollo (shift-left) hasta a operación (shift-right), certificando una seguridad continua a lo largo de todo el ciclo de vida del software.

2.2. Marco teórico

2.2.1. Arquitectura interna de Docker

La plataforma Docker permite ejecutar aplicaciones dentro de contenedores, ofreciendo un entorno ligero e independiente, sin necesidad de instalación de un sistema operativo completo, Por lo cual facilita que una aplicación funcione de igual manera en diferentes sistemas y equipos.

La arquitectura está formada principalmente por Docker Engine, que se enfoca en la administración, para la creación y ejecución de contenedores. Su motor recibe las instrucciones del usuario mediante el cliente Docker (CLI, Command Line Interface) además utiliza imágenes en la generación de contenedores, dichas imágenes comprenden el sistema de archivo, dependencias y configuraciones que hacen que una aplicación se ejecute correctamente

Los contenedores se aíslan entre sí mediante tecnología kernel de Linux, los namespaces, separan los procesos y recursos, los cgroups controlan el uso de CPU, memoria y otros recursos del sistema, permitiendo que varios contenedores se ejecuten de forma simultánea, segura y eficiente.

La arquitectura permite que sea una herramienta ampliamente utilizada en entornos DevOps (Development Operations) y DevSecOps (Development Security Operations), ya que simplifica el despliegue de aplicaciones, mejora la portabilidad y optimiza el uso de los recursos del servidor.

2.2.2. Arquitectura del motor de GitHub Actions

GitHub Actions es una plataforma de integración y entrega continua (CI/CD) integrada en GitHub. Permite automatizar flujos de trabajo (workflows) mediante archivos

Yet Another Markup Language (YAML) que ejecutan tareas automáticas (jobs y steps) en entornos aislados (runners), desencadenados por eventos específicos como push o pull requests. En entornos DevSecOps, actúa como el orquestador estratégico para centralizar pruebas y análisis de seguridad.

Riesgos Introducidos en el Pipeline:

- Ataques a la Cadena de Suministro: Uso de acciones de terceros (Marketplace) vulnerables o modificadas maliciosamente si apuntan a versiones mutables (ej. @main).
- Inyección de Código: Manipulación de variables de contexto dinámicas (como títulos de PR) para ejecutar comandos no autorizados en el runner.
- Fuga de Credenciales: Exposición accidental de secretos y llaves Application Programming Interface (API) en los logs de ejecución.

Controles de Mitigación Implementados:

- Inmutabilidad de Acciones: Fijación obligatoria de versiones mediante el hash criptográfico Secure Hash Algorithm (SHA) (ej. actions/checkout@a5ac7e..) en lugar de etiquetas mutables.
- Principio de Menor Privilegio: Configuración estricta del bloque permissions en el YAML para restringir el token por defecto (GITHUB_TOKEN) a solo lectura (contents: read).
- Inyección Segura de Secretos: Uso exclusivo de GitHub Secrets o integración OpenID Connect (OIDC) con bóvedas externas, evitando credenciales embebidas.
- Análisis Estático (Linting): Inclusión de Actionlint en el pipeline para detectar automáticamente configuraciones inseguras antes de su ejecución.

2.2.3. Cultura DevSecOps y el principio Shift Left

DevSecOps es una evolución de DevOps que "integra la seguridad como componente nativo del pipeline desde sus etapas tempranas, transformando la seguridad de control de fase final a actividad continua y automatizada" (Abigail, 2025, párr. 1). El principio fundamental de DevSecOps es *shift-left security*, que establece que "la seguridad debe integrarse lo más pronto posible en el ciclo de desarrollo y así reducir costos de remediación y realizar una rápida detección de vulnerabilidades" (Abigail, 2025, párr. 2).

El principio Shift Left se fundamenta en la teoría clásica del costo exponencial de remediación formulada por Boehm (1981), la cual demuestra que mitigar un defecto en la fase de requisitos cuesta (1x), en diseño de (3-6x), en codificación de (6-10x), en pruebas (testing) de (15-20x), y en producción de (30-100x) (p. 40). *Shift Left* aplica esta teoría moviendo la detección de vulnerabilidades a fases tempranas donde el costo de remediación es mínimo. La implementación práctica incluye "seguridad en Integrated Development Environment (IDE) (plugins que detectan vulnerabilidades en tiempo real), seguridad en *commit* (*pre-commit hooks* con SAST/SCA), seguridad en *build* (Grype/Trivy escanean contenedores), seguridad en *test* (DAST, *fuzzing*) y seguridad en *release* (*policy-as-code* automatizado)" (Abigail, 2025, párr. 4).

2.2.4. Criptografía y gestión de secretos en CI/CD

La criptografía "protege secretos (contraseñas, tokens API, claves privadas) mediante la transformación matemática de datos legibles a ilegibles" (WeeSec, 2025, párr. 1). En CI/CD, "la criptografía simétrica (AES-256) cifra secretos en repositorios, la criptografía asimétrica (RSA-2048, ECC) firma artefactos y SBOM, y el hashing criptográfico (SHA-256) verifica integridad" (WeeSec, 2025, párr. 2).

Los secretos en CI/CD incluyen "credenciales de autenticación (bases de datos, SSH keys), tokens de autorización (GitHub tokens, OAuth, JWT), claves criptográficas (claves privadas para firma, certificados SSL/TLS) y API Keys de servicios cloud" (WeeSec, 2025, párr. 3). La exposición accidental de secretos genera riesgos críticos: "aproximadamente el 30% de repositorios públicos en GitHub contienen secretos expuestos, permitiendo acceso no autorizado a recursos, robo de datos y contribuye a que la infraestructura quede comprometida" (WeeSec, 2025, párr. 3).

Las mejores prácticas incluyen: "principio de mínimo privilegio (solo Secrets estrictamente necesarios), rotación regular de secrets (cada 30-90 días, o dynamic secrets con rotación automática), segregar secrets por ambiente (development, staging, production), nunca hardcodear secrets en código fuente o Dockerfiles, redacción automática en logs (GitHub Actions redacta automáticamente, otras plataformas requieren configuración manual), auditoría periódica de accesos, scanning de secrets en código con herramientas (trufflehog, gitleaks) integradas en pre-commit hooks" (OpsMx, 2025; WeeSec, 2025, párr. 4).

La gestión de secretos es fundamental para firma criptográfica de SBOM: "las claves privadas para firmar SBOM con cosign/Sigstore deben almacenarse en bóvedas digitales, no en código fuente. Solo pipelines autorizados acceden a claves de firma, con logs de auditoría de quién usó la clave y cuándo, y rotación automática según política definida" (WeeSec, 2025, párr. 5).

2.2.5. Funcionamiento del análisis estático de imágenes

El análisis estático de imágenes de contenedores representa una de las técnicas más utilizadas dentro de los entornos modernos de DevSecOps y ciberseguridad preventiva.

Herramientas como Trivy y Grype permiten detectar vulnerabilidades en imágenes de contenedores sin necesidad de ejecutar activamente el sistema analizado. Este enfoque reduce riesgos operativos y evita la exposición de servicios potencialmente comprometidos durante el proceso de evaluación.

Estas herramientas realizan un montaje lógico y desempaquetado “en frío” del sistema de archivos contenido dentro de la imagen. Cada imagen Docker o Open Container Initiative (OCI) se encuentra compuesta por múltiples capas que almacenan paquetes, dependencias, configuraciones y binarios. Durante el análisis, Trivy y Grype extraen las capas, reconstruyen el árbol del sistema de archivos y examinan los componentes presentes utilizando metadatos y firmas de paquetes.

De acuerdo con Anchore (2025), Grype realiza análisis de imágenes de contenedores y sistemas de archivos con el objetivo de detectar vulnerabilidades conocidas.

Además, estas herramientas poseen integración con pipelines CI/CD, permitiendo automatizar verificaciones de seguridad antes del despliegue en producción. Esto fortalece las prácticas al incorporar controles de vulnerabilidades desde etapas tempranas del desarrollo.

Trivy es una herramienta de análisis de vulnerabilidades que permite identificar fallos de seguridad en paquetes del sistema operativo y dependencias de aplicaciones (Aqua Security, 2025).

2.2.6. Teoría y anatomía de un SBOM

Un SBOM constituye un inventario estructurado de todos los componentes, bibliotecas y dependencias que conforman una aplicación o sistema de software. Su objetivo

principal es proporcionar visibilidad sobre los elementos internos del software para facilitar la gestión de riesgos, auditorías de seguridad y cumplimiento normativo.

"An SBOM is a nested inventory of software components."
(CISA, 2024).

Un SBOM puede representarse como un árbol de dependencias o un grafo de paquetes. Cada nodo representa un componente de software, mientras que las relaciones entre nodos describen dependencias directas o transitivas. Esta estructura permite rastrear vulnerabilidades heredadas y comprender cómo un componente vulnerable puede impactar otros módulos dentro del ecosistema de software.

La implementación de SBOMs se ha convertido en una práctica fundamental dentro de arquitecturas DevSecOps modernas, debido a que incrementa la transparencia del software y facilita la identificación rápida de componentes afectados frente a nuevas vulnerabilidades publicadas.

Para la implementación propuesta en este trabajo se utilizará el formato CycloneDX como estándar para la generación de SBOM. La elección se fundamenta en que CycloneDX fue diseñado específicamente para el ámbito de la ciberseguridad y la gestión de vulnerabilidades, proporcionando información detallada sobre componentes, dependencias y metadatos relevantes para los procesos DevSecOps. Además, herramientas como Trivy y Grype ofrecen soporte nativo para este formato, facilitando su integración dentro del pipeline automatizado de análisis de imágenes Docker. Asimismo, CycloneDX permite una mejor correlación entre los componentes identificados y las vulnerabilidades detectadas, contribuyendo a la trazabilidad y gestión eficiente de riesgos de seguridad durante el ciclo de vida del software.

2.2.7. Sistemas de puntuación de vulnerabilidades

Los sistemas de puntuación de vulnerabilidades son fundamentales en la gestión de la seguridad, ya que permiten identificar, clasificar y priorizar los riesgos asociados a fallos en software, sistemas e infraestructuras. Además, proporcionan un lenguaje común estandarizado que facilita la comunicación entre desarrolladores, equipos de seguridad y organizaciones. A nivel global, destacan estándares como Common Vulnerabilities and Exposures (CVE), que cataloga vulnerabilidades conocidas; Common Weakness Enumeration (CWE), que clasifica debilidades comunes; y Common Vulnerability Scoring System (CVSS), que permite medir su nivel de severidad.

El estándar CVSS permite medir la gravedad de una vulnerabilidad mediante un puntaje que va de 0.0 a 10.0, facilitando así su evaluación y priorización en función de su nivel de riesgo. Su estructura se compone de tres grupos de métricas: como base, que consiste en calcular el nivel de riesgo de una vulnerabilidad; temporales, se usan para refinar la evaluación del riesgo, dependiendo de la vulnerabilidad actual; y ambientales, utilizadas para personalizar la evaluación del riesgo según el contexto existente de la organización, la versión más usada y estable seria CVSS v3.1, se bloque según su severidad, basándose en los umbrales ≥ 7.0 o ≥ 9.0.

2.2.8. Estándares internacionales de seguridad para contenedores

El avance de las arquitecturas basadas en contenedores ha impulsado la necesidad de contar con estándares internacionales que orienten su seguridad. Estos establecen buenas prácticas, controles y lineamientos para reducir riesgos en entornos cloud-native.

Entre los principales se encuentran:

- NIST SP 800-190
- OWASP Top 10
- Otros marcos de ciberseguridad que funcionan como complemento

Los estándares como NIST SP 800-190 y OWASP Top 10 contribuyen como una guía para proteger los entornos de contenedores, definiendo buenas prácticas y controles que permiten mitigar riesgos de manera sistemática. Su correcta aplicación dentro de prácticas DevSecOps contribuye a reducir riesgos de forma organizada y endureciendo a la seguridad integral del ciclo de vida del software.

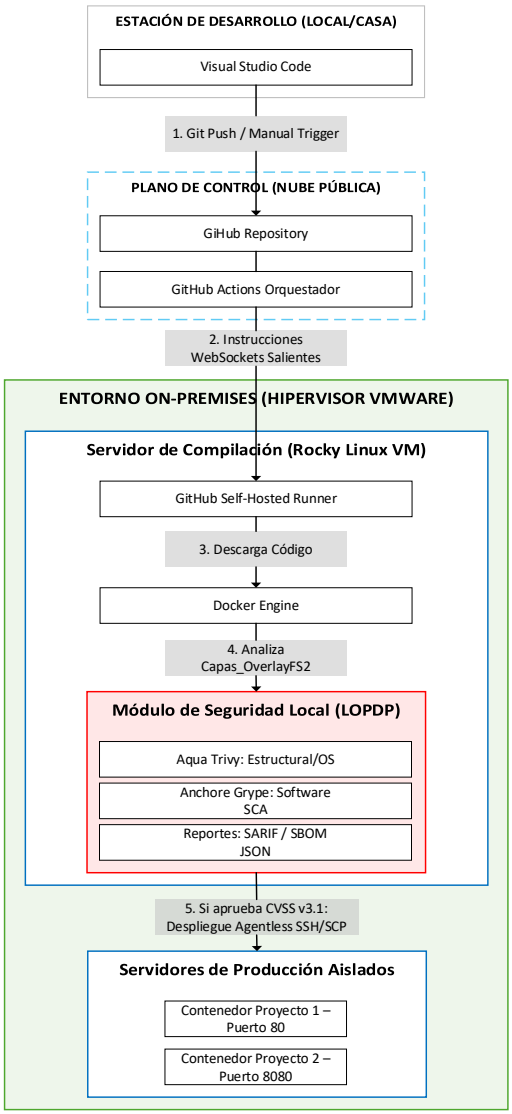
CAPÍTULO 3

3. Desarrollo.

Como parte del trabajo realizado el grupo ha definido el siguiente flujo híbrida para la implementación en el entorno de pruebas:

Figura 1

Desarrollo del proyecto



Nota. Elaboración propia. El diagrama representa la arquitectura híbrida del pipeline de Development, Security and Operations (DevSecOps) implementado para el análisis automático de vulnerabilidades en imágenes Docker.

Este flujo se encuentra soportado por la siguiente infraestructura:

- Servidor de Compilación Rocky Linux
- Plano de control (Nube Pública) GITHUB
- Servidor de publicación de servicios.

Las características de estos componentes son los siguientes:

Servidor Rocky Linux:

Creado sobre el hipervisor Vmware ESXi (On-premises)

Memoria RAM: 16 GB

Virtual CPU (vCPU): 4

Almacenamiento: 40 GB

Versión de Sistema Operativo: Rocky Linux 10 (Red Quartz)

Tipo de instalación: Mínima.

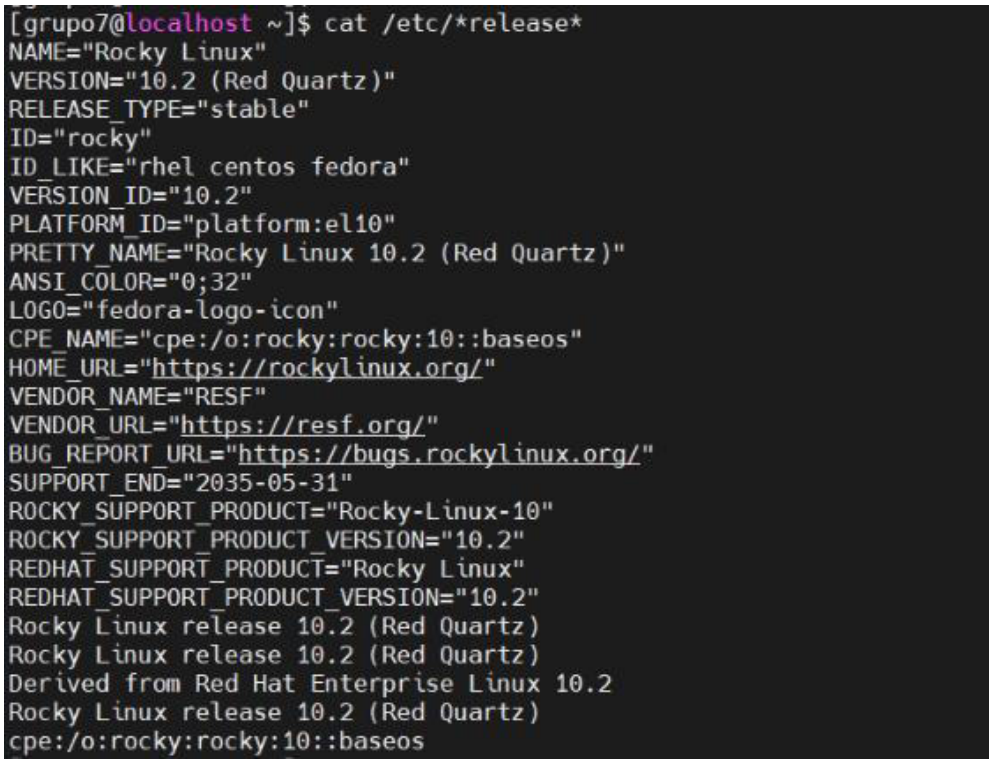
Figura 2

Configuración de hardware de la máquina virtual Rocky Linux

Nota. Detalle de la configuración de hardware de la máquina virtual utilizada para implementar el servidor Rocky Linux 10.

Figura 3

Identificación de la versión del sistema operativo Rocky Linux 10.2



Nota. Salida del comando `cat /etc/*release*` utilizada para verificar la versión del sistema operativo Rocky Linux 10.2 instalado en el servidor self-hosted runner.

Se procede a instalar todas las dependencias necesarias para que el servidor Rocky Linux realice las tareas de compilación y aplicación de los módulos de seguridad Anchore Grype y Aqua Trivy.

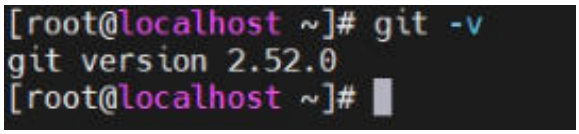
Se realiza la instalación de git a través del comando:

```
dnf install git
```

Se valida la versión que se instaló en el servidor:

Figura 4

Verificación de la versión Git 2.52.0



Nota. Salida del comando `git -v` utilizada para verificar la versión instalada de Git en el servidor `self-hosted runner`.

La versión de Git 2.52.0 integra parches para vulnerabilidades de desbordamiento de buffer y ejecución remota de código, ya que Trivy y Grype dependen de la integridad de archivos planos y de los manifiestos descargados para realizar el Análisis de Composición de Software esta versión mitiga el riesgo de manipulación de la cadena de suministro antes de que los artefactos a ser analizados ingresen al servidor Rocky Linux.

Se procede a realizar la instalación de Docker en el servidor:

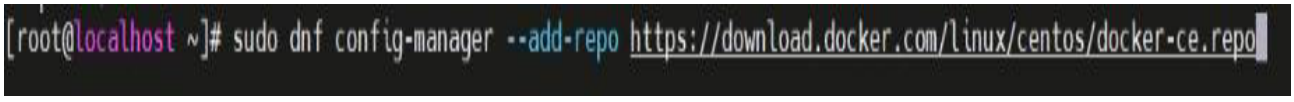
Agregar el repositorio de Docker para su instalación a través del comando:

```
sudo dnf config-manager –add-repo
```

https://download.docker.com/linux/centos/docker-ce.repo

Figura 5

Adición del repositorio oficial de Docker



Nota. Ejecución del comando `sudo dnf config-manager --add-repo` para agregar el repositorio oficial de Docker al sistema operativo Rocky Linux 10.

Una vez agregado el repositorio se procede a realizar la instalación de los siguientes componentes de Docker:

`docker-ce`

`docker-ce-cli`

`containerd.io`

`docker-buildx-plugin`

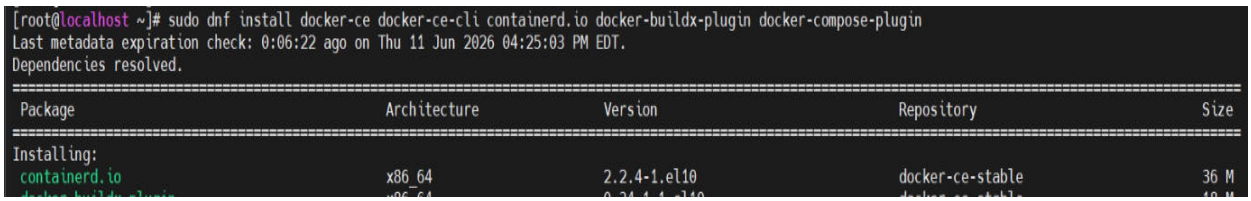
`docker-compose-plugin`

Esta instalación se realiza con el siguiente comando:

`dnf install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin`

Figura 6

Instalación de Docker Engine



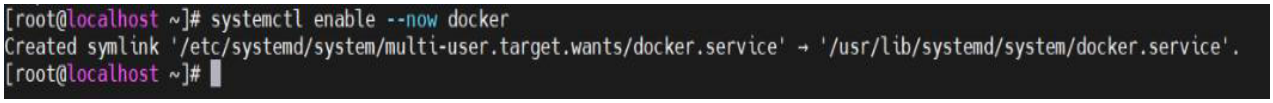
Nota. Ejecución del comando sudo `dnf install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin` para instalar Docker Engine y sus componentes en el servidor Rocky Linux 10.

Habilitamos el inicio automático para el servicio Docker:

`systemctl enable --now Docker`

Figura 7

Habilitación e inicio automática del servicio Docker

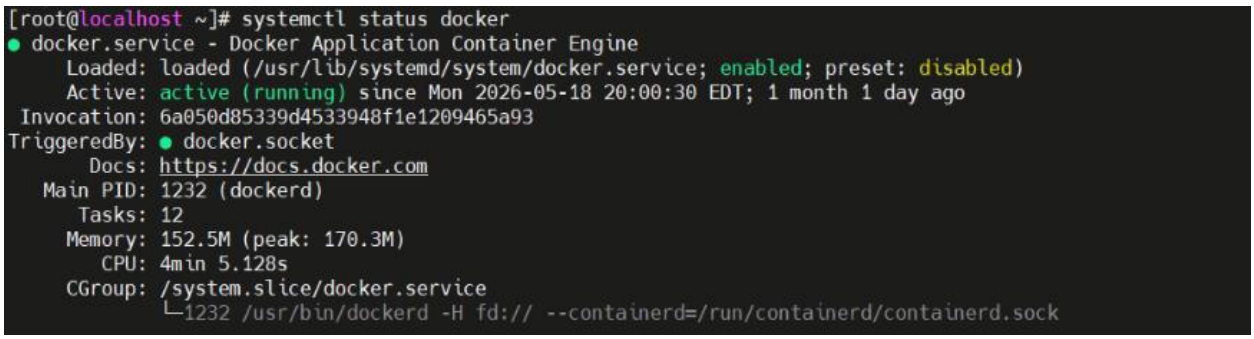


Nota. Ejecución del comando `systemctl enable --now docker` para habilitar el inicio automático e iniciar el servicio Docker en el servidor Rocky Linux 10.

Validamos que se encuentre habilitado el servicio:

Figura 8

Verificación del estado del servicio Docker



Nota. Elaboración propia. Captura de pantalla que muestra la salida del comando `systemctl status docker`, utilizada para verificar que el servicio Docker se encuentra habilitado (*enabled*) y en ejecución (*active (running)*) en el servidor Rocky Linux 10.

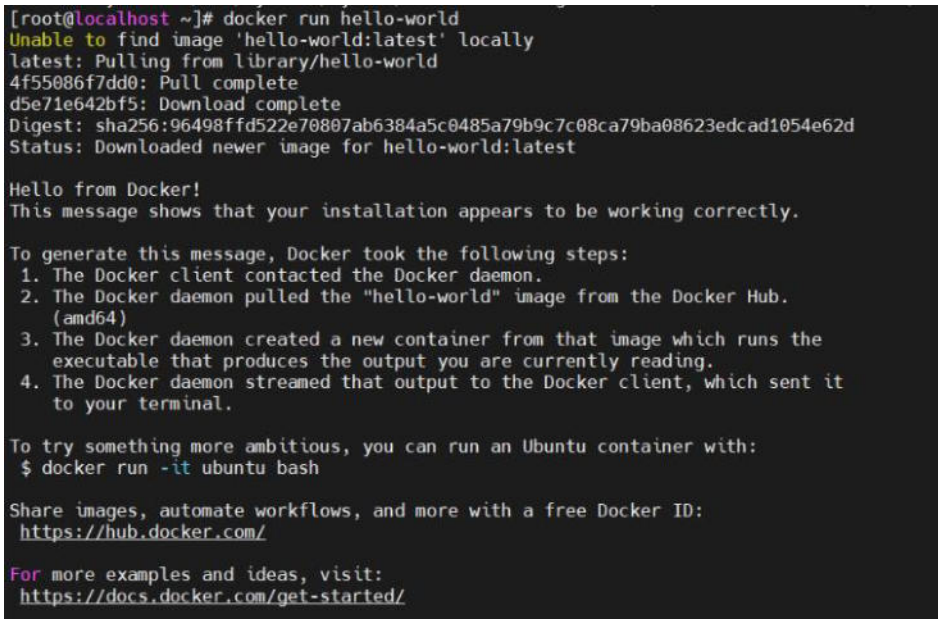
Realizamos la validación de que Docker se encuentre ejecutándose correctamente con la ejecución del comando:

```
docker run hello-world
```

Este comando busca la imagen local hello-world, al no encontrarla realiza la descarga desde Docker Hub en el internet, finalmente crea y levanta el contenedor a partir de esa imagen.

Figura 9

Validación de la instalación de Docker mediante el contenedor Hello World



Nota. Elaboración propia. Captura de pantalla que muestra la ejecución del comando `docker run hello-world`, utilizada para validar el correcto funcionamiento de Docker mediante la descarga y ejecución del contenedor de prueba **Hello World**.

Con la finalidad de lograr que el pipeline de GitHub se ejecute dentro de nuestro entorno VMware (Rocky Linux) debemos enlazar GittHub y Rocky Linux a través de la configuración del puente híbrido GitHub Self-Hosted Runner.

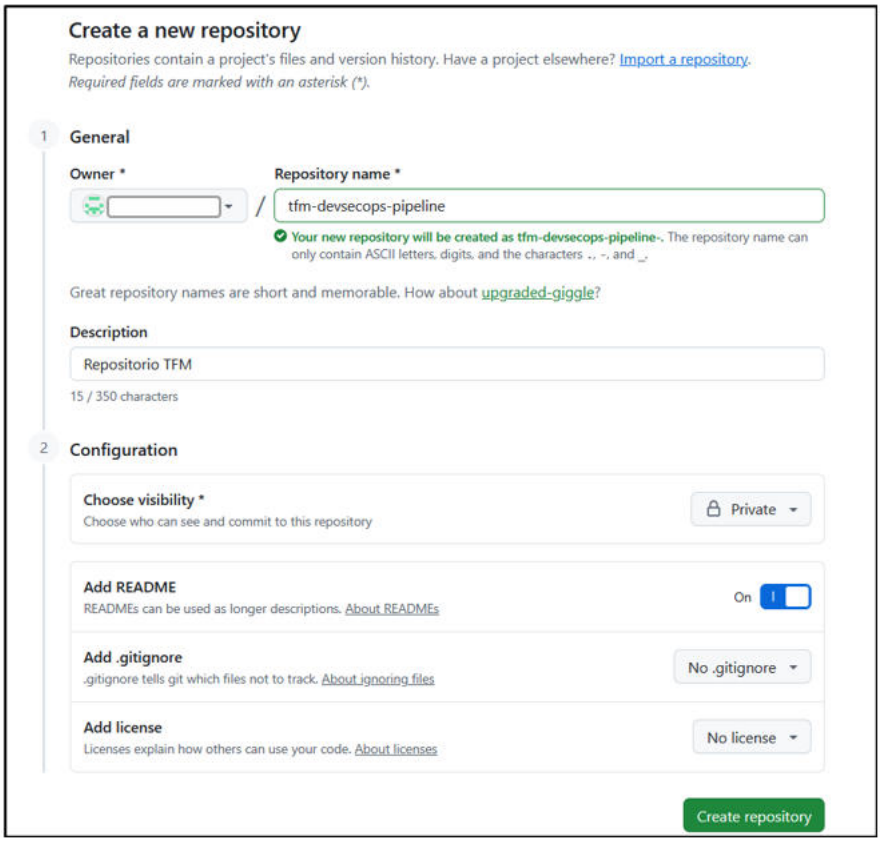
Para obtener las credenciales debemos realizar los siguientes pasos:

1. Crear el repositorio para nuestro código del Trabajo Fin de Máster (TFM), ingresando con nuestra cuenta al sitio https://github.com.

Una vez dentro de nuestra cuenta elegimos la opción “Crear nuevo repositorio”, llenamos los datos solicitados y finalmente damos click en el botón “Create repository”

Figura 10

Creación del repositorio del proyecto en GitHub

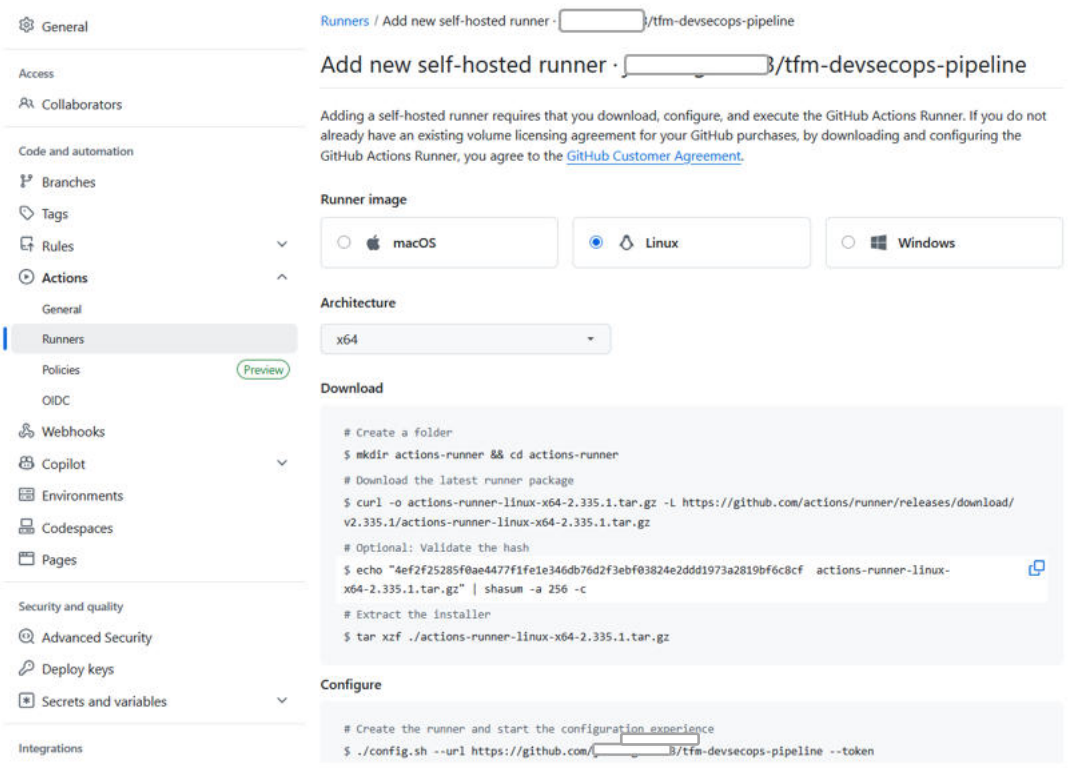


Nota. Elaboración propia. Captura de pantalla que muestra la creación del repositorio `tfm-devsecops-pipeline` en GitHub, utilizado para almacenar el código fuente y los archivos de configuración del pipeline DevSecOps.

2. Una vez creado nuestro repositorio, ingresamos a este, luego click en la opción Settings (Configuración) que se encuentra ubicado en la parte superior derecha, se despliega un menú en la parte izquierda, buscamos la sección Action y damos click en New self-hosted runner y seleccionamos “Linux”

Figura 11

Creación de un self-hosted runner en GitHub



Nota. Elaboración propia. Figura de GitHub que muestra la creación de un **self-hosted runner** para Linux, utilizado para ejecutar de forma local los flujos de trabajo (*workflows*) del pipeline DevSecOps.

A partir de esto GitHub nos muestra los pasos detallados uno por uno que debemos configurar en nuestro servidor Rocky Linux para realizar la integración:

Crear una carpeta para el agente de enlace:

```
mkdir actions-runner && cd actions-runner
```

Descargar el instalador oficial de GitHub:

```
curl -o actions-runner-linux-x64-2.334.0.tar.gz -L
```

https://github.com/actions/runner/releases/download/v2.334.0/actions-runner-linux-x64-2.334.0.tar.gz

Figura 12

Creación de la carpeta y descarga del paquete del self-hosted runner

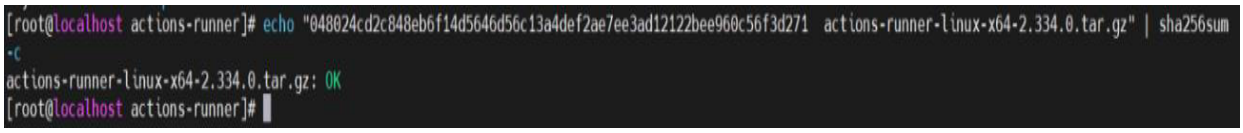
Nota. Elaboración propia. La figura anterior que muestra la creación del directorio actions-runner y la descarga del paquete oficial del *self-hosted runner* desde GitHub para su posterior configuración.

Se realiza la validación del hash del archivo descargado.

```
echo
"048024cd2c848eb6f14d5646d56c13a4def2ae7ee3ad12122bee960c56f3d271  actions-runner-linux-x64-2.334.0.tar.gz" | sha256sum -c
```

Figura 13

Verificación de la integridad del paquete del self-hosted runner mediante SHA-256



Nota. Elaboración propia. Captura de pantalla que muestra la verificación del hash SHA-256 del paquete del *self-hosted runner*, utilizada para comprobar la integridad del archivo descargado antes de su instalación.

Se procede a extraer el instalador:

```
tar xzf ./actions-runner-linux-x64-2.334.0.tar.gz
```

Finalmente se configura el enlace:

```
./config.sh --url https://github.com/juanmiguel113/tfm-devsecops-pipeline --token xxxxxxxxxxxxxxxxxxxxxx
```

Figura 14

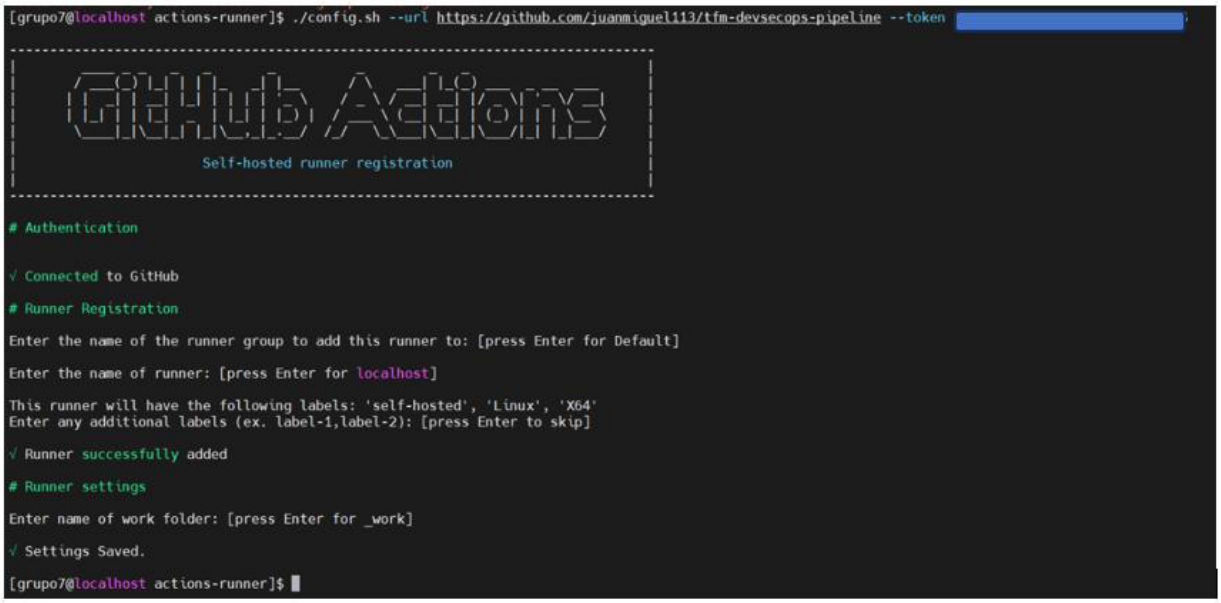
Registro del self-hosted runner en GitHub Actions

Nota. Elaboración propia. Captura de pantalla que muestra la ejecución del script config.sh para registrar el *self-hosted runner* en GitHub Actions, permitiendo su asociación con el repositorio del proyecto y la ejecución local de los *workflows* del pipeline DevSecOps.

Luego se presentarán varias preguntas (como el nombre del runner y las etiquetas) en las cuales daremos enter para dejarlos con los valores por defecto.

Figura 15

Registro exitoso del self-hosted runner en GitHub Actions



Nota. Elaboración propia. Captura de pantalla que muestra la finalización del proceso de registro del *self-hosted runner* en GitHub Actions, confirmando su correcta asociación con el repositorio mediante los mensajes **Connected to GitHub**, **Runner successfully added** y **Settings Saved**.

Por seguridad tenemos habilitado SELinux en el servidor, SELinux (Security-Enhanced Linux) es una capa de seguridad en los servidores Linux derivados de Fedora, tales como: RedHat, CetOS, Rocky Linux, Alma Linux, etc, esta capa de seguridad brinda estrictos controles sobre los procesos, aplicativos y archivos que se ejecutan en el servidor, es decir un control de acceso obligatorio para los usuarios del sistema.

SELinux tiene tres modos de operación:

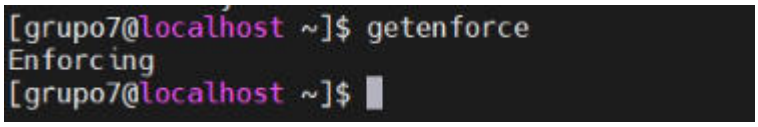
Enforcing: Bloquea cualquier acción no autorizada.

Permissive: Solo advierte y registra, no bloquea ninguna acción

Disabled: Apagado

Figura 16

Verificación del modo de operación de SELinux



Nota. Elaboración propia. Captura de pantalla que muestra la ejecución del comando getenforce, utilizada para verificar que SELinux se encuentra configurado en modo **Enforcing**, garantizando la aplicación de las políticas de seguridad del sistema operativo.

Se verificó que SELinux permaneciera en modo **Enforcing**, ya que este modo aplica activamente las políticas de control de acceso obligatorio (MAC), contribuyendo al fortalecimiento de la seguridad del entorno donde se ejecuta el pipeline DevSecOps.

Por lo tanto, debemos configurar SELinux para ejecutar archivos que están dentro de la carpeta personal (grupo7). Al cambiar el tipo a bin_t, estamos configurando en SELinux para que trate al script de inicio del servicio runner de usuario como si fuera un binario legítimo del sistema, y así este no sea bloqueado.

```
sudo chcon -t bin_t /home/grupo7/actions-runner/runsvc.sh
```

Y procedemos a levantar el servicio:

```
sudo ./svc.sh start
```

Figura 17

Inicio del servicio del self-hosted runner

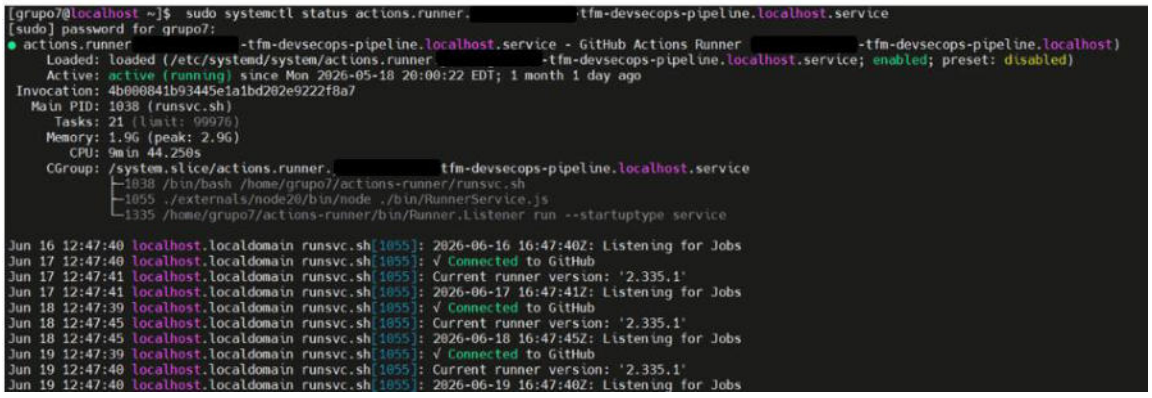
Nota. Elaboración propia. Captura de pantalla que muestra la ejecución del script svc.sh para iniciar el servicio del *self-hosted runner* y verificar que permanece en estado **active (running)** dentro del sistema operativo Rocky Linux.

Una vez que el servicio ha subido, se verifica que este se encuentre activo a través del siguiente comando:

```
sudo systemctl status actions.runner.juanmiguel113-tfm-devsecops-pipeline.localhost.service
```

Figura 18

Verificación del estado del servicio del self-hosted runner



Nota. Elaboración propia. Captura de pantalla que muestra la salida del comando `systemctl status`, utilizada para verificar que el servicio del *self-hosted runner* se

encuentra en estado **active (running)** y conectado correctamente a GitHub, permaneciendo disponible para ejecutar los *workflows* del pipeline DevSecOps.

Con esto se valida la correcta integración de nuestro servidor Self-hosted runner Rocky Linux con GitHub Actions.

El siguiente paso es instalar dentro del servidor los motores de escaneo Aqua Trivy y Anchor Grype, con la finalidad de que el pipeline las utilice localmente:

Aqua Trivy

Para realizar la instalación de Aqua Trivy agregamos el repositorio al servidor a través del siguiente comando:

```
cat <<EOF | sudo tee /etc/yum.repos.d/trivy.repo
[trivy]
name=Trivy repository
baseurl=https://github.io\$basearch/
gpgcheck=1
enabled=1
gpgkey=https://github.io
EOF
```

A través de este comando se crea el archivo trivy.repo en la ubicación `/etc/yum.repos.d` cono todo el contenido necesario para que le sistema operativo sepa de donde descargar el software al momento de solicitar la instalación.

Figura 19

Configuración del repositorio de Trivy

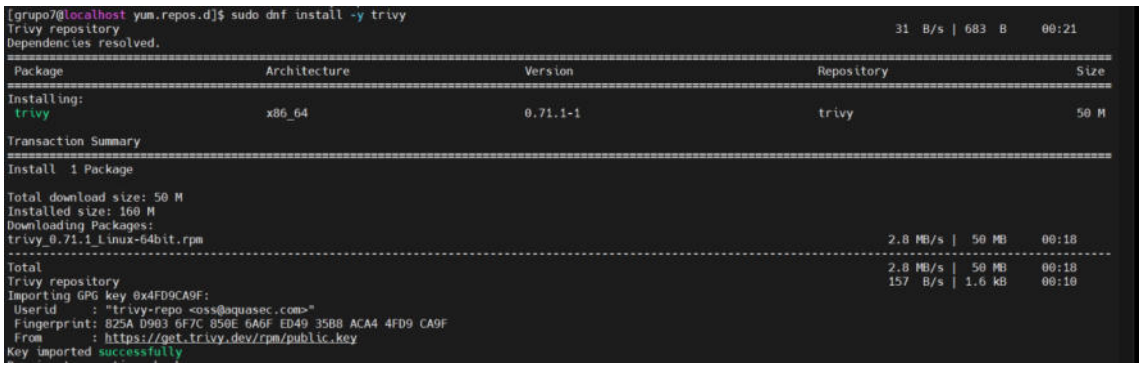
Nota. Elaboración propia. Captura de pantalla que muestra el contenido del archivo trivy.repo, utilizado para configurar el repositorio oficial de Trivy en el sistema operativo Rocky Linux antes de su instalación.

Una vez que se ha configurado el repositorio se procede a realizar la instalación del motor Trivy:

```
sudo dnf install -y Trivy
```

Figura 20

Instalación de Trivy



Nota. Elaboración propia. Captura de pantalla que muestra la ejecución del comando `sudo dnf install -y trivy`, utilizado para instalar el escáner de vulnerabilidades Trivy desde su repositorio oficial en el servidor Rocky Linux 10.

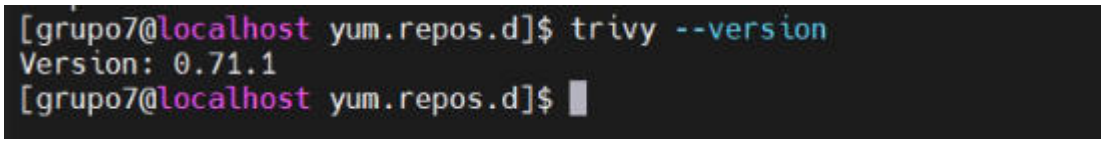
Durante la instalación se importó automáticamente la clave GNU Privacy Guard (GPG) del repositorio oficial, garantizando la autenticidad e integridad del paquete instalado.

Para validar que la instalación se haya realizado correctamente ejecutamos el siguiente comando:

```
trivy -version
```

Figura 21

Verificación de la versión de Trivy



Nota. Elaboración propia. Captura de pantalla que muestra la ejecución del comando `trivy --version`, utilizada para verificar la correcta instalación del escáner de vulnerabilidades Trivy y confirmar la versión instalada (**0.71.1**) en el servidor Rocky Linux 10.

Para finalizar con la instalación de Trivy procedemos a descargar la base de datos de vulnerabilidades de Trivy en el servidor con la finalidad de actualizar la base de datos previo el escaneo.

```
trivy image --download-db-only
```

Figura 22

Descarga de la base de datos de vulnerabilidades de Trivy

Nota. Elaboración propia. Captura de pantalla que muestra la ejecución del comando `trivy image --download-db-only`, utilizado para descargar la base de datos de vulnerabilidades de Trivy antes de realizar el análisis de imágenes Docker.

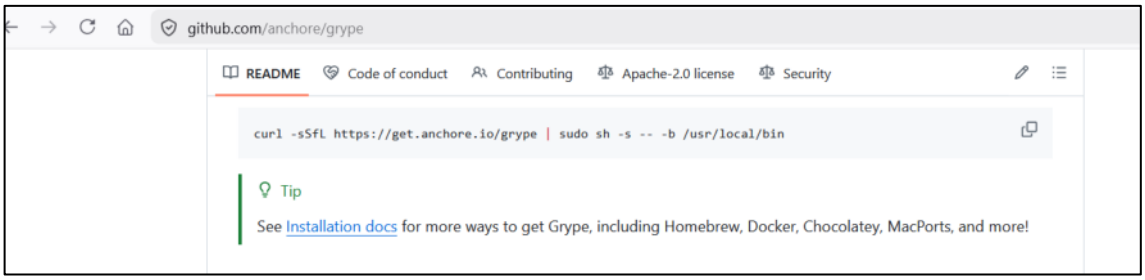
Antes de ejecutar los análisis de vulnerabilidades, se descargó la base de datos oficial de Trivy mediante el parámetro `--download-db-only`, garantizando que las detecciones se realizaran con información actualizada sobre vulnerabilidades conocidas.

De igual forma procedemos a instalar el motor de Anchor Grype en el servidor Rocky Linux.

Con el siguiente comando se procede a descargar e instalar Grype automáticamente conforme el script detallado en su página web.

Figura 23

Obtención del comando de instalación de Grype desde la documentación oficial

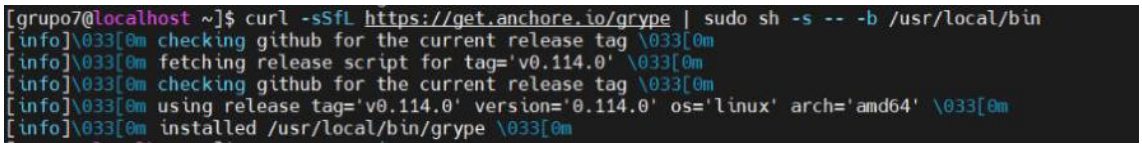


Nota. Elaboración propia. Captura de pantalla de la documentación oficial de Grype que muestra el comando recomendado para su instalación en sistemas Linux, utilizado como referencia para la implementación del escáner de vulnerabilidades.

```
curl -sSfL https://get.anchore.io/grype | sudo sh -s -- -b
/usr/local/bin
```

Figura 24

Instalación de Grype



Nota. Elaboración propia. Captura de pantalla que muestra la ejecución del comando de instalación de Grype mediante el script oficial de Anchore, utilizada para instalar el escáner de vulnerabilidades en el servidor Rocky Linux 10.

La instalación finalizó correctamente, quedando el ejecutable disponible en la ruta `/usr/local/bin/grype`, lo que permitió su utilización en las etapas posteriores del análisis de vulnerabilidades.

Igual que el motor anterior se procede a realizar la validación de la versión instalada.

```
grype –versión
```

Figura 25

Validación de la versión de Grype

Nota. Elaboración propia. Captura de pantalla que muestra la ejecución del comando `grype --version`, utilizada para verificar la correcta instalación del escáner de vulnerabilidades Grype y confirmar la versión instalada (**0.114.0**) en el servidor Rocky Linux 10.

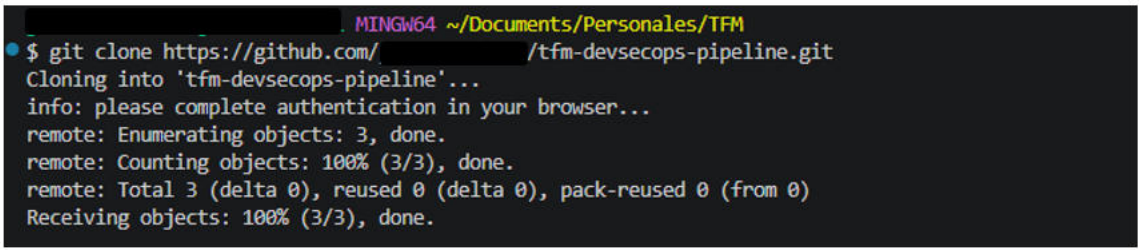
Una vez instalados los motores de escaneo de seguridad procedemos a crear la estructura del proyecto para ser analizado por el pipeline.

Clonamos el repositorio creado en GitHub a la maquina local, para realizar todos los cambios y crear la estructura del proyecto.

El grupo ha decido trabajar con Visual Studio Code para generar el sitio de prueba y el push al repositorio de GitHub.

Figura 26

Clonación del repositorio del proyecto desde GitHub

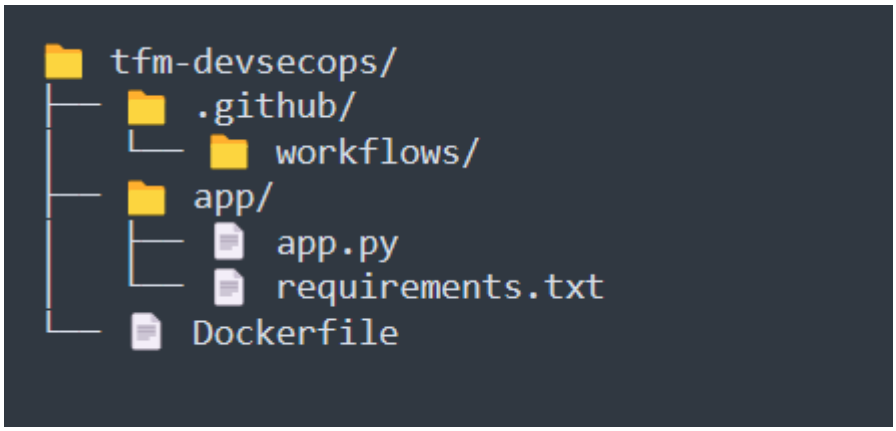


Nota. Elaboración propia. Captura de pantalla que muestra la ejecución del comando git clone, utilizado para clonar el repositorio del proyecto desde GitHub hacia la estación de trabajo local, con el fin de realizar el desarrollo y las pruebas del pipeline DevSecOps.

Una vez clonado el repositorio se crea la siguiente estructura:

Figura 27

Estructura del repositorio del proyecto



Nota. Elaboración propia. Figura que muestra la estructura del repositorio del proyecto, incluyendo los directorios y archivos principales utilizados para el desarrollo de la aplicación, la construcción de la imagen Docker y la automatización del pipeline mediante GitHub Actions.

El repositorio fue organizado siguiendo una estructura modular. El directorio. `github/workflows` almacena los flujos de trabajo de GitHub Actions; la carpeta app contiene el código fuente de la aplicación y sus dependencias; mientras que el archivo Dockerfile define el proceso de construcción de la imagen Docker utilizada durante el análisis de vulnerabilidades.

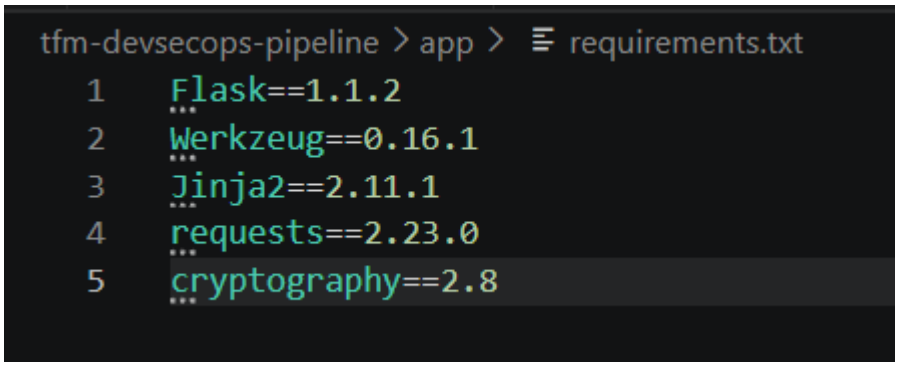
`app.py: Aplicación web en Python usando Flask.`

`requirements.txt`

Define las librerías de Python necesarias para el funcionamiento de la aplicación, para este diseño se usa librerías antiguas con fallos de seguridad.

Figura 28

Archivo requirements.txt con las dependencias de la aplicación



Nota. Elaboración propia. Captura de pantalla del archivo `requirements.txt`, que contiene las dependencias de Python requeridas para la ejecución de la aplicación utilizada en el pipeline DevSecOps.

El archivo requirements.txt define las bibliotecas necesarias para la ejecución de la aplicación. Estas dependencias son utilizadas posteriormente durante la construcción de la imagen Docker mediante el Dockerfile, garantizando un entorno reproducible para el análisis automatizado de vulnerabilidades.

Las dependencias fueron seleccionadas en versiones con vulnerabilidades conocidas para validar la capacidad de detección de Trivy y Grype durante la ejecución del pipeline.

Dockerfile

Es el archivo que se utiliza para empaquetar la aplicación, este archivo detalla el paso a paso para generar la imagen la imagen de nuestra aplicación. Para este TFM lo diseñamos incumpliendo reglas de seguridad a apropósito.

Usamos un Sistema Operativo antiguo Ubuntu 20.04

No usamos la opción USER para que el contenedor se ejecute como root.

Figura 29

Archivo Dockerfile para la construcción de la imagen Docker

m-devsecops-pipeline > Dockerfile > ...

```
# Instrucción 1: Usar un sistema operativo viejo (Ubuntu 20.04) lleno de parches sin resolver
FROM ubuntu:20.04

# Instrucción 2: Evitar que el sistema haga preguntas interactivas durante la instalación
ENV DEBIAN_FRONTEND=noninteractive

# Instrucción 3: Descargar e instalar Python en el contenedor
# NOTA DE TFM: Ponemos los comandos por separado para engordar las "capas de Docker" a proposi
RUN apt-get update
RUN apt-get install -y python3 python3-pip curl

# Instrucción 4: Crear una carpeta de trabajo dentro del contenedor
WORKDIR /workspace

# Instrucción 5: Copiar la lista de ingredientes (librerías vulnerables)
COPY app/requirements.txt /workspace/requirements.txt

# Instrucción 6: Instalar las librerías viejas que pusimos en el Paso 4
RUN pip3 install --no-cache-dir -r requirements.txt

# Instrucción 7: Copiar el código de la aplicación web
COPY app/ /workspace/
```

Nota. Elaboración propia. Captura de pantalla del archivo Dockerfile, utilizado para definir el proceso de construcción de la imagen Docker que será empleada durante el análisis automatizado de vulnerabilidades en el pipeline DevSecOps.

El archivo Dockerfile define las instrucciones necesarias para construir la imagen Docker utilizada en el experimento. Se empleó una imagen base Ubuntu 20.04 y se instalaron dependencias de Python especificadas en requirements.txt. Además, las instrucciones fueron organizadas en capas independientes para incrementar el número de capas de la imagen y facilitar el análisis realizado por Trivy y Grype.

`devsecops-pipeline.yml`: es el archivo de instrucciones de automatización el cual integra la GitHub con nuestro servidor Rocky Linux.

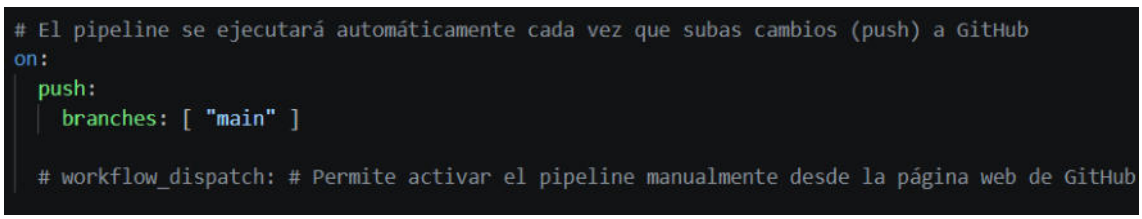
En este archivo contempla los siguientes bloques:

Activación automática cada vez que se suba código nuevo a la rama principal en GitHub.

- Se agrega comentando la instrucción `workflow_dispath` para activar el pipeline manualmente, con esto se previene que cada vez que se realice un push al main del proyecto en Github se dispare el análisis automaticamente.

Figura 30

Configuración del evento de activación del workflow



Nota. Elaboración propia. Captura de pantalla del archivo de configuración del *workflow* de GitHub Actions, donde se define el evento push sobre la rama **main** como mecanismo para iniciar automáticamente la ejecución del pipeline DevSecOps.

- Es este bloque se obliga a GitHub no usar la nube pública, más bien que las instrucciones sean enviadas a nuestro servidor Rocky Linux.

Figura 31

Configuración del trabajo de escaneo de seguridad en GitHub Actions

```
jobs:
  escaneo-seguridad:
    # OBJETIVO TFM: Obliga a GitHub a usar tu servidor local de VMware (Self-Hosted)
    runs-on: self-hosted
```

Nota. Elaboración propia. Captura de pantalla del archivo de configuración del *workflow* de GitHub Actions, donde se define el trabajo (*job*) de escaneo de seguridad y se especifica la ejecución sobre un **runner self-hosted**, permitiendo que el pipeline se ejecute en la infraestructura local del laboratorio.

El workflow define un trabajo denominado escaneo-seguridad, configurado para ejecutarse sobre un runner self-hosted. Esta configuración permite que todas las etapas del pipeline se ejecuten en la infraestructura local del laboratorio, donde se encuentran instaladas las herramientas Docker, Trivy y Grype.

- Este bloque copia los archivos del proyecto app.py, requirements.txt y Dockerfile desde GitHub a nuestro servidor Rocky Linux, todo esto dentro de una carpeta temporal.

Figura 32

Obtención del código fuente mediante actions/checkout

```yaml
steps:
# 1. Descarga el código fuente del repositorio hacia el servidor Rocky Linux
- name: Descargar codigo fuente
  uses: actions/checkout@v4
```

Nota. Elaboración propia. Captura de pantalla del archivo de configuración del *workflow* de GitHub Actions que muestra el uso de la acción oficial actions/checkout@v4, encargada de obtener el código fuente del repositorio antes de ejecutar las etapas del pipeline.

La primera etapa del workflow utiliza la acción oficial actions/checkout@v4, cuya función es descargar el contenido del repositorio Git al entorno de ejecución del runner self-hosted. Esto permite que las etapas posteriores del pipeline accedan al código fuente, al Dockerfile y a los archivos de configuración necesarios para construir la imagen Docker y ejecutar los análisis de seguridad.

- Este bloque solicita al servidor Rocky Linux que ejecute el contenido del archivo dockerfile para empaquetar la aplicación. Es decir, crea el contenedor que será base para este estudio.

Figura 33

Construcción de la imagen Docker mediante el archivo Dockerfile

```yaml
# 2. Construye la imagen Docker utilizando el Dockerfile inseguro del proyecto
- name: Construir Imagen Docker Insegura
  run: |
    docker build -t app-vulnerable:latest .
```

Nota. Elaboración propia. Captura de pantalla del archivo de configuración del *workflow* de GitHub Actions que muestra la ejecución del comando docker build, utilizado para construir la imagen Docker del proyecto a partir del archivo Dockerfile.

En esta etapa del pipeline se ejecuta el comando docker build, el cual interpreta las instrucciones definidas en el archivo Dockerfile para generar la imagen Docker denominada app-vulnerable:latest. Esta imagen constituye el objeto de análisis de vulnerabilidades por parte de Trivy y Grype en las etapas posteriores del pipeline.

- En este bloque se lanza el escaner de Trivy, con la opción –severity CRITICAL para que realice la búsqueda de vulnerabilidades extremas según el estándar CVSS 3.1

Figura 34

Configuración del análisis de vulnerabilidades con Trivy

```
# Objetivo TFM: Establecer políticas de bloqueo basadas en el estándar CVSS v3.1
- name: Escaneo Estructural con Trivy (Bloqueo por Fallos Críticos)
  run: |
    echo "=== Iniciando escaneo con Trivy ==?"
    # --exit-code 1: Si encuentra una vulnerabilidad de la severidad indicada, rompe el
    # --severity CRITICAL: Filtra y evalúa únicamente los fallos con métrica CVSS v3.1
    trivy image --exit-code 1 --severity CRITICAL app-vulnerable:latest
```

Nota. Elaboración propia. Captura de pantalla del archivo de configuración del *workflow* de GitHub Actions que muestra la ejecución del escáner Trivy para analizar la imagen Docker y detectar vulnerabilidades críticas mediante la política de severidad definida.

En esta etapa del pipeline se ejecuta Trivy para analizar la imagen Docker construida previamente. Se configuró el parámetro --severity CRITICAL para considerar únicamente vulnerabilidades críticas y --exit-code 1 para interrumpir la ejecución del pipeline cuando se detecten vulnerabilidades de ese nivel, implementando así una política de seguridad basada en el estándar CVSS.

- En este bloque se realiza el escaneo de dependencias con Grype con la finalizar de analizar fallas específicas de la librería Python.

Figura 35

Configuración del análisis de dependencias con Grype

```
# Esto te permitirá recolectar métricas simultáneas para las tablas comparativas.
- name: Escaneo de Dependencias de Aplicacion con Grype (SCA)
  if: always()
  run: |
    echo "=== Iniciando escaneo con Grype ==="
    # Ejecuta el escaneo de software y muestra los resultados detallados en la consola
    grype app-vulnerable:latest
```

Nota. Elaboración propia. Captura de pantalla del archivo de configuración del *workflow* de GitHub Actions que muestra la ejecución del escáner Grype para realizar el análisis de dependencias (*Software Composition Analysis*, SCA) sobre la imagen Docker construida previamente.

Después del análisis realizado por Trivy, el pipeline ejecuta Grype para identificar vulnerabilidades presentes en las dependencias de software incluidas en la imagen Docker. La condición if: always() garantiza que este análisis se ejecute incluso si el escaneo anterior detecta vulnerabilidades y finaliza con un código de error.

La condición if: always() fue incorporada para asegurar que Grype se ejecute independientemente del resultado obtenido por Trivy, permitiendo recopilar información de ambos escáneres y realizar posteriormente una comparación de los resultados.

3.1. Validación del funcionamiento del pipeline.

Una vez que se clono localmente a través de Visual Studio Code se realizan los cambios en el archivo devsecops-pipeline.yml y se sube todos los cambios al repositorio de GitHub (git push) y estos son exitosos se realiza el siguiente proceso:

- Se activa el archivo devsecops-pipeline.yml, GitHub Actions envía una señal en tiempo real al servidor Rocky Linux.

- El servidor Rocky Linux se conecta a GitHub y descarga automaticamente una copia de todos los archivos (app.py, requirements.txt y Dockerfile) dentro de la carpeta temporal.
- Posteriormente el servidor ejecuta el paso de crear la imagen del contenedor.
- Se realiza los escaneos de seguridad con las herramientas Trivy y Grype.

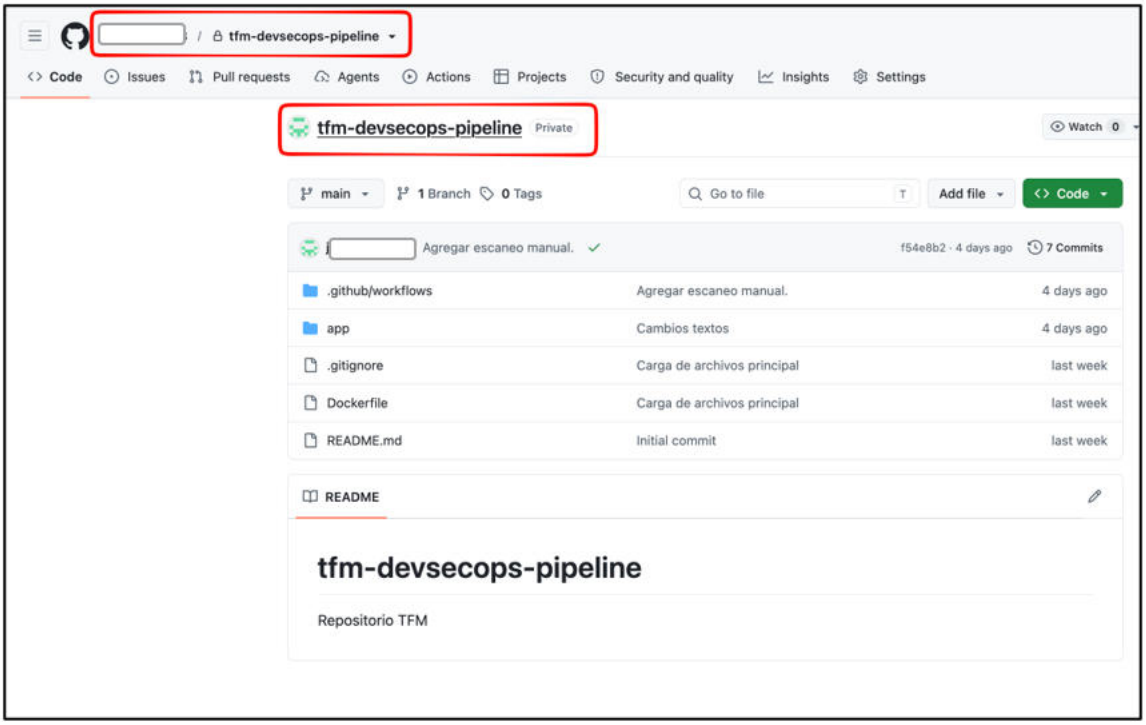
Validación de Ejecución del Pipeline

Para realizar la validación de que el pipeline se ejecutó realizamos los siguientes pasos:

- Entramos a nuestro repositorio en GitHub.

Figura 36

Repositorio del proyecto en GitHub



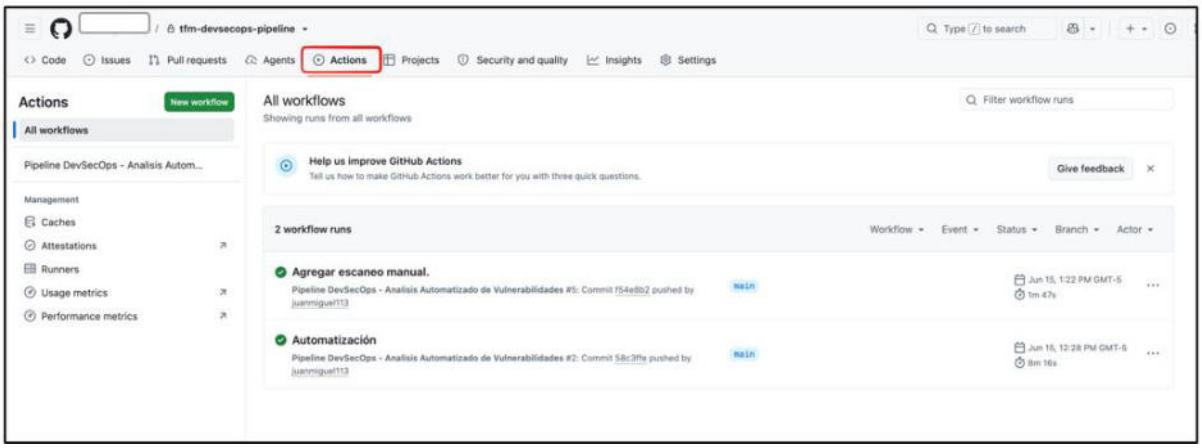
Nota. Elaboración propia. Captura de pantalla del repositorio del proyecto alojado en GitHub, donde se almacenan el código fuente, los archivos de configuración y el *workflow* utilizado para la ejecución del pipeline DevSecOps.

El repositorio del proyecto se encuentra alojado en GitHub y constituye el punto central para el control de versiones del código fuente. Además de almacenar los archivos de la aplicación, contiene la configuración del pipeline implementado mediante GitHub Actions, permitiendo la ejecución automática del análisis de vulnerabilidades tras cada actualización del repositorio.

- Hacemos click en la pestaña Actions.

Figura 37

Visualización de la ejecución del pipeline en GitHub Actions



Nota. Elaboración propia. Captura de pantalla de la pestaña **Actions** del repositorio en GitHub, donde se visualiza el historial de ejecuciones del pipeline DevSecOps y el estado de los workflows ejecutados.

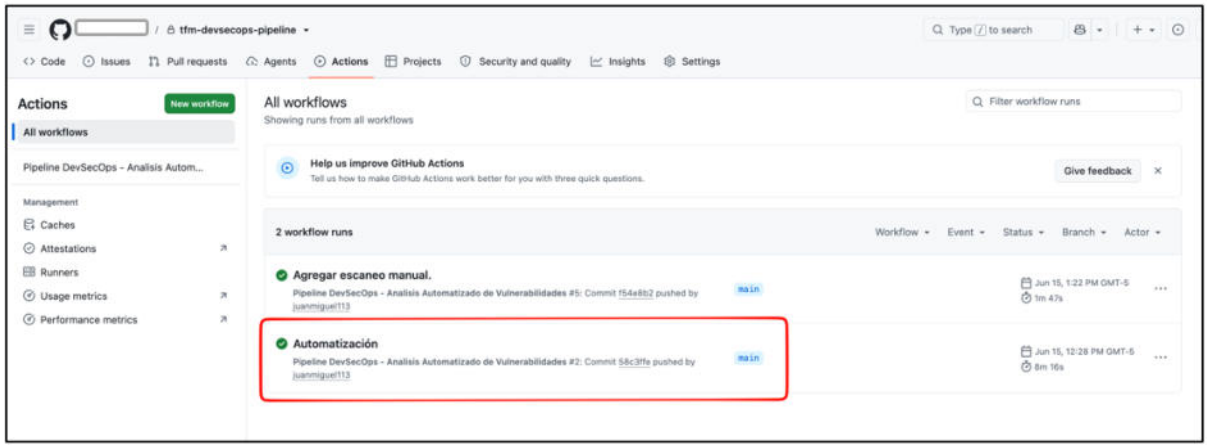
GitHub Actions proporciona un panel de seguimiento donde se registran todas las ejecuciones del pipeline. En esta vista es posible verificar el estado de cada workflow, el

evento que originó su ejecución, la rama utilizada y el tiempo empleado, facilitando el monitoreo del proceso automatizado.

- Visualizamos la ejecución del pipeline.

Figura 38

Ejecución exitosa del pipeline en GitHub Actions



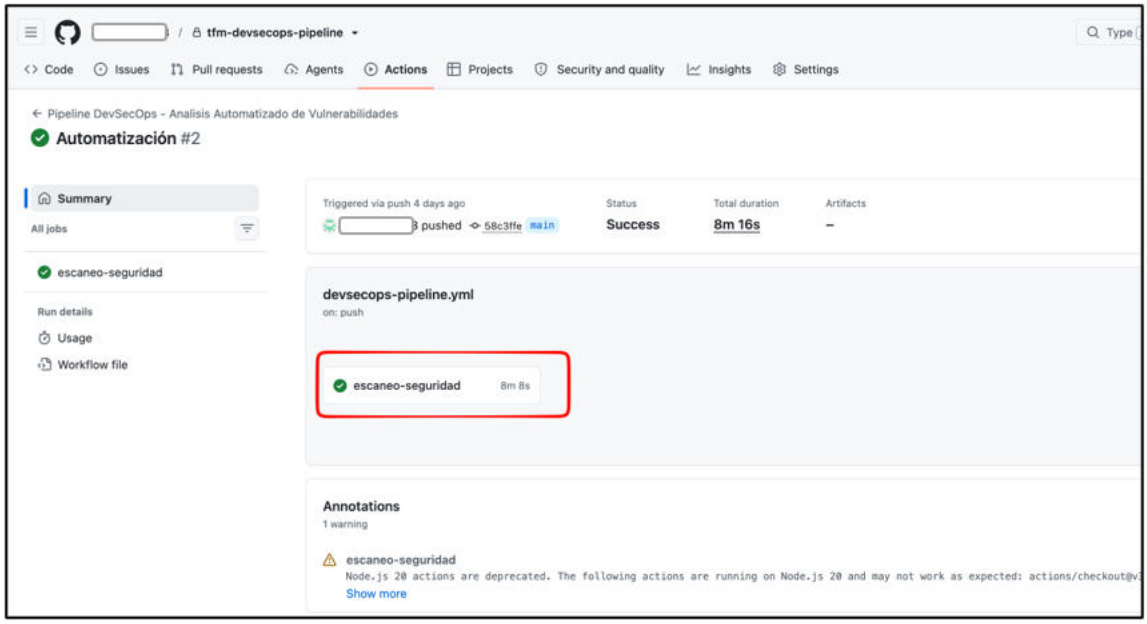
Nota. Elaboración propia. Captura de pantalla de GitHub Actions que muestra la ejecución exitosa del pipeline DevSecOps, evidenciando que el workflow fue ejecutado correctamente sobre la rama principal del repositorio.

Una vez configurado el workflow, GitHub Actions ejecutó automáticamente el pipeline tras un cambio en la rama principal. El estado exitoso de la ejecución confirma que todas las etapas definidas en el flujo de trabajo fueron procesadas correctamente antes de revisar el detalle de cada una de ellas.

- Damos click sobre la palabra Automatización y luego sobre el icono escaneo-seguridad.

Figura 39

Detalle de la ejecución del trabajo de escaneo de seguridad



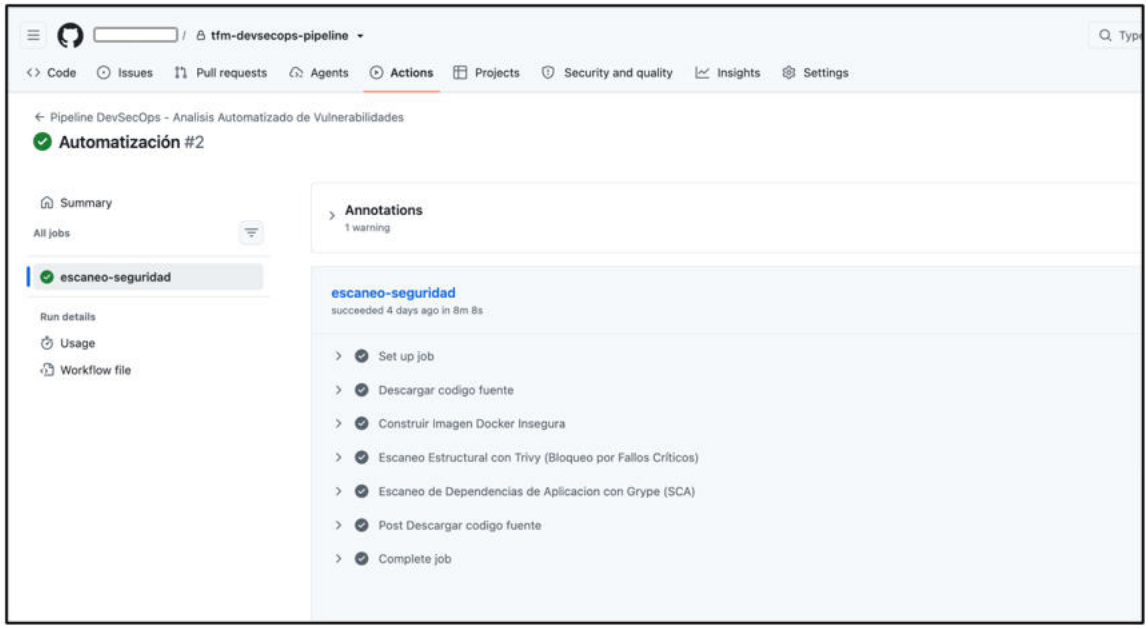
Nota. Elaboración propia. Captura de pantalla del resumen de ejecución del *workflow* en GitHub Actions, donde se muestra el trabajo **escaneo-seguridad** ejecutado correctamente, incluyendo su estado y duración.

GitHub Actions presenta un resumen de la ejecución del pipeline, donde se verifica que el trabajo denominado **escaneo-seguridad** fue completado exitosamente. Esta vista proporciona información sobre el estado del proceso, el tiempo de ejecución y el acceso al detalle de cada una de las etapas ejecutadas.

Se visualiza todos los pasos que le ejecutaron en el pipeline.

Figura 40

Resumen de las etapas ejecutadas del pipeline DevSecOps



Nota. Elaboración propia. Captura de pantalla del resumen de ejecución del *workflow* en GitHub Actions, donde se muestran las etapas ejecutadas durante el pipeline DevSecOps, incluyendo la construcción de la imagen Docker y los análisis de vulnerabilidades realizados con Trivy y Grype.

GitHub Actions presenta un resumen de todas las etapas ejecutadas durante el workflow. Se observa que el pipeline realizó correctamente la preparación del entorno, la descarga del código fuente, la construcción de la imagen Docker, el análisis estructural mediante Trivy, el análisis de dependencias con Grype y la finalización del proceso, confirmando la correcta automatización del flujo DevSecOps propuesto.

A continuación, revisamos cada uno de los pasos generados por nuestro pipeline.

Como primer paso prepara el entorno de ejecución antes que las instrucciones comiencen a ejecutarse.

Figura 41

Preparación del entorno de ejecución del pipeline

Nota. Elaboración propia. Captura de pantalla de la etapa **Set up job** del pipeline en GitHub Actions, donde se prepara el entorno de ejecución, se validan los permisos del repositorio y se descargan las acciones necesarias para iniciar el workflow.

La primera etapa del pipeline corresponde a la preparación automática del entorno de ejecución. Durante este proceso, GitHub Actions identifica el runner configurado, asigna los permisos requeridos para el acceso al repositorio y descarga las acciones necesarias antes de iniciar las tareas definidas en el workflow.

Se muestran los registros detallados de la descarga del código al servidor Rocky Linux, estos archivos son clonados desde el repositorio de GitHub para que las herramientas de escaneo puedan trabajar sobre ellos.

Se evidencia el path temporal en el servidor donde se clonarán los archivos.

`/home/grupo7/actions-runner/_work/_temp/fc009c41-2f9c-4882-9ab2-fae6a68d9f5d`

Figura 42

Descarga del código fuente del repositorio

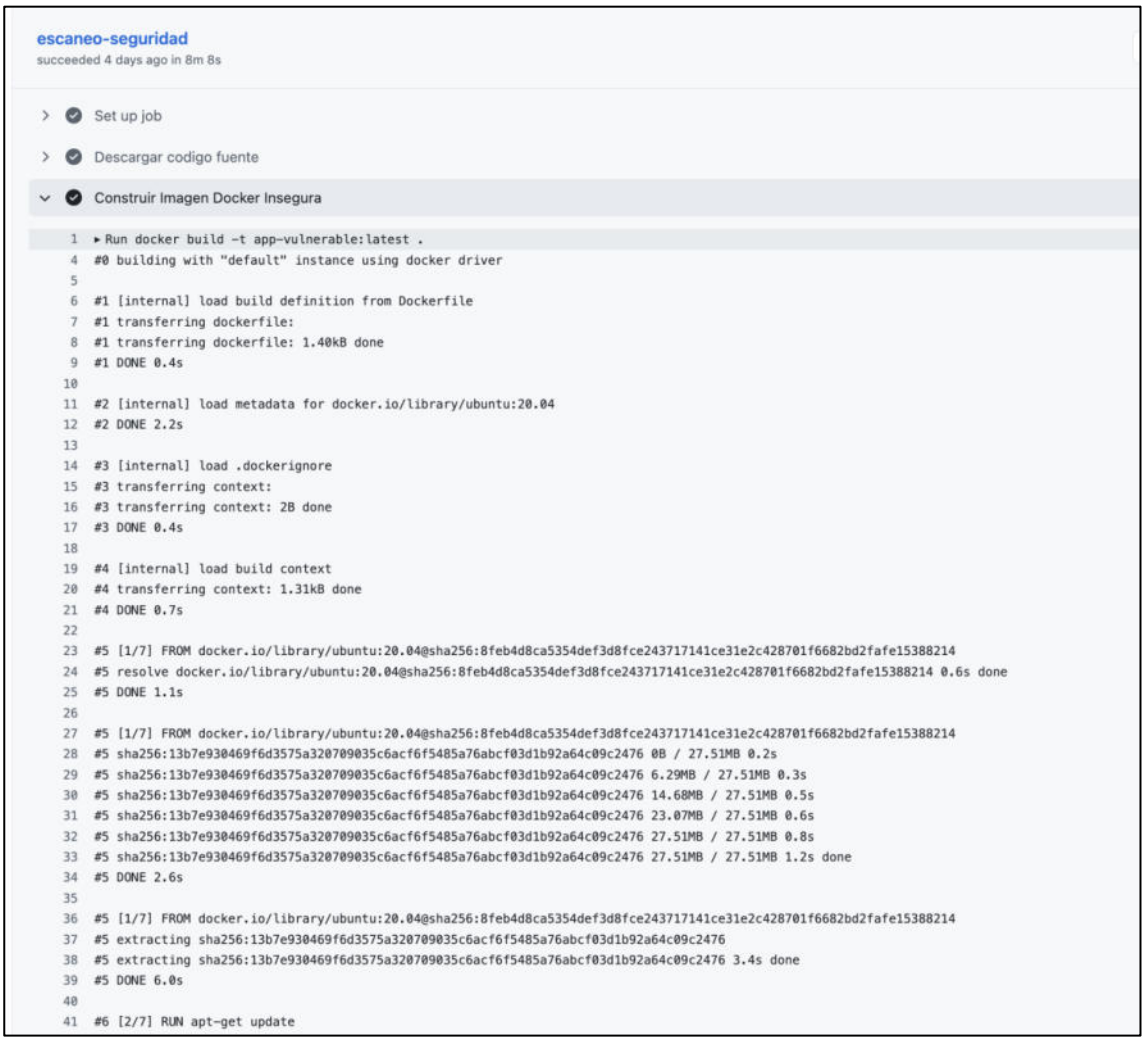
Nota. Elaboración propia. Captura de pantalla de la etapa **Descargar código fuente** del pipeline, donde la acción **actions/checkout@v4** sincroniza el contenido del repositorio de GitHub con el servidor *self-hosted* para continuar con la ejecución del workflow.

En esta etapa del pipeline se ejecuta la acción oficial **actions/checkout@v4**, encargada de descargar el contenido del repositorio hacia el runner self-hosted. Este proceso inicializa el repositorio Git en el entorno de ejecución y deja disponible el código fuente para las etapas posteriores de construcción de la imagen Docker y análisis de vulnerabilidades.

Se muestran los registros del tercer paso de la construcción de la imagen Docker, a través de la compilación del archivo Dockerfile con la finalidad de empaquetar la aplicación en el contenedor.

Figura 43

Construcción de la imagen Docker mediante el pipeline



Nota. Elaboración propia. Captura de pantalla de la etapa de construcción de la imagen Docker dentro del pipeline DevSecOps, donde se ejecuta el comando docker build para generar la imagen **app-vulnerable:latest** a partir del Dockerfile del proyecto.

Después de descargar el código fuente, el pipeline ejecuta la construcción automática de la imagen Docker utilizando el archivo Dockerfile del proyecto. Durante esta etapa se descargan las capas base del sistema operativo, se ejecutan las instrucciones definidas por el desarrollador y se genera la imagen que posteriormente será analizada mediante las herramientas Trivy y Grype.

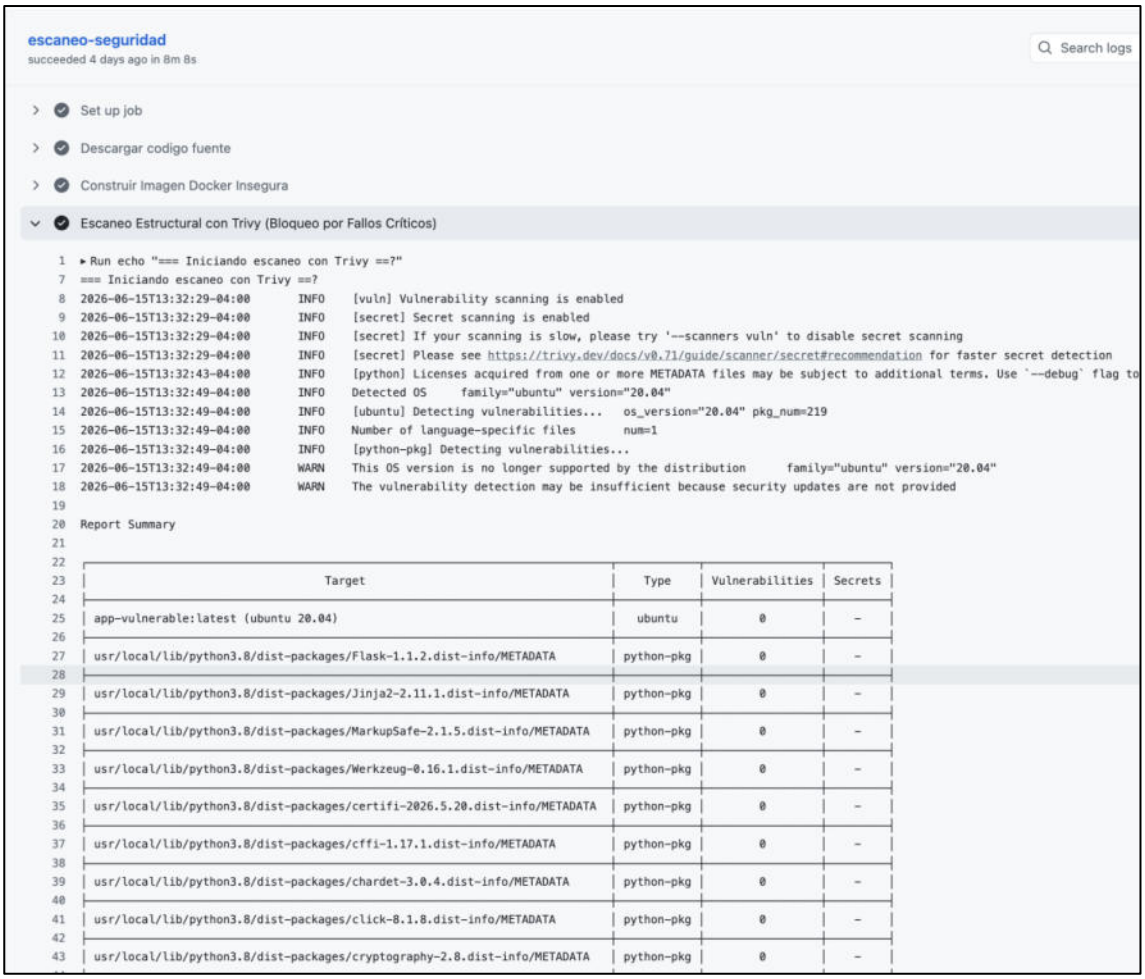
En el siguiente paso se muestra el escaneo con Trivy, aquí la herramienta realiza una auditoria detallada de la imagen Docker construida en el paso anterior.

Como observación podemos ver que nos avisa que la imagen Ubuntu 20.04 ya no cuenta con el soporte completo y nos advierte que la detección de vulnerabilidades puede ser insuficiente por cuanto ya no se proporcionan actualizaciones de seguridad para esa imagen.

Aquí se valida que al tener esta versión obsoleta Trivy reporta 0 vulnerabilidades y secretos, por lo tanto, el pipeline continuo sin bloquearse.

Figura 44

Reporte del análisis de vulnerabilidades con Trivy



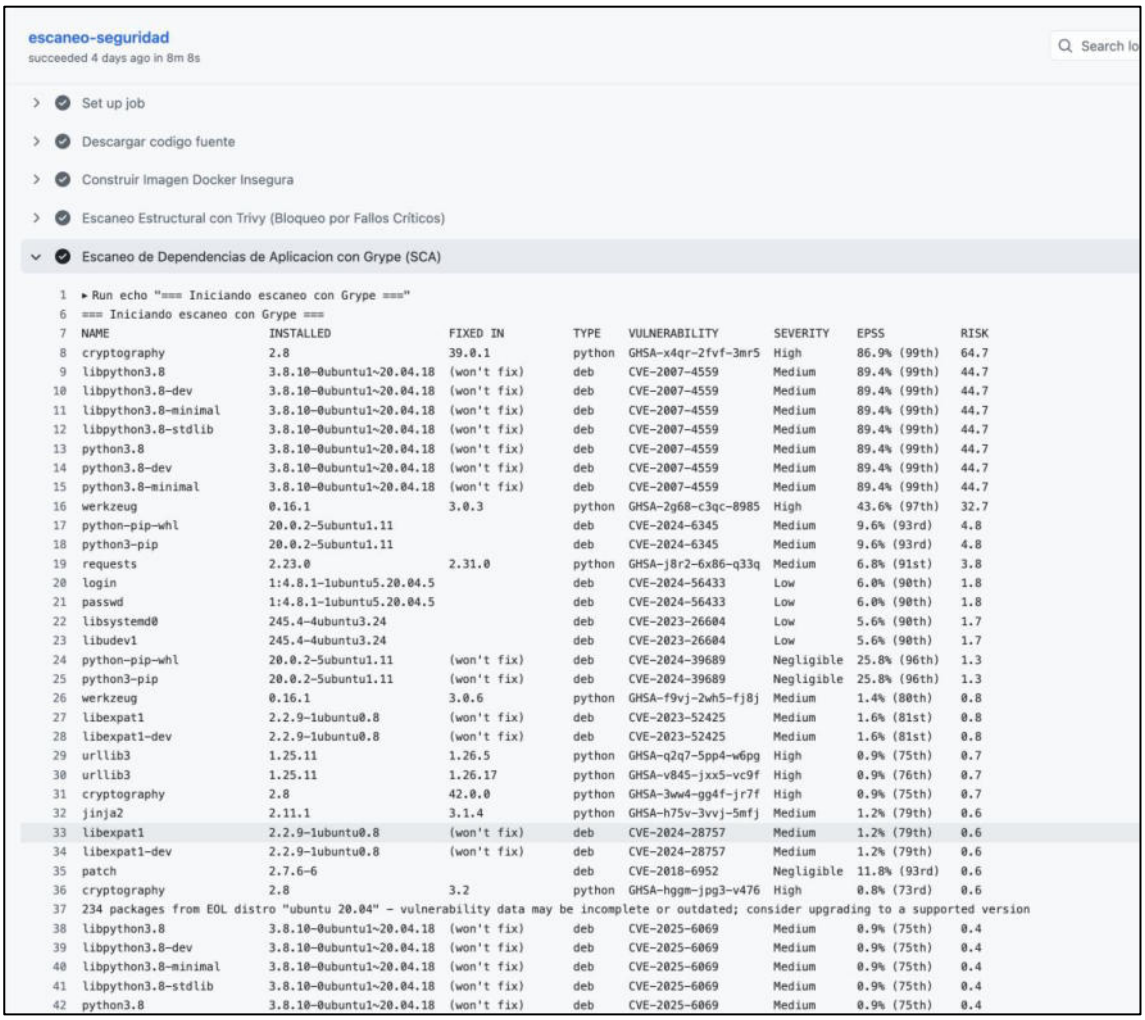
Nota. Elaboración propia. Captura de pantalla del reporte generado por Trivy durante la ejecución del pipeline DevSecOps, donde se presenta el resumen del análisis estructural de la imagen Docker y el número de vulnerabilidades detectadas en cada componente evaluado.

Una vez construida la imagen Docker, el pipeline ejecuta Trivy para realizar un análisis estructural de vulnerabilidades. La herramienta actualiza automáticamente su base de datos, inspecciona el sistema operativo y las bibliotecas instaladas, y genera un reporte que resume los componentes analizados y el número de vulnerabilidades identificadas, permitiendo evaluar el nivel de exposición de la imagen.

En esta quinta etapa del pipeline la herramienta Grype realiza el análisis de la composición de software, enfocado en encontrar vulnerabilidades conocidas (Common Vulnerabilities and Exposures (CVE) – GitHub Security Advisory (GHSA)) en las librerías del sistema operativo y las del código Python.

Figura 45

Reporte del análisis de vulnerabilidades con Grype



escaneo-seguridad

succeeded 4 days ago in 8m 8s

Search lo

- Set up job
- Descargar codigo fuente
- Construir Imagen Docker Insegura
- Escaneo Estructural con Trivy (Bloqueo por Fallos Criticos)
- Escaneo de Dependencias de Aplicacion con Grype (SCA)

```
▸ Run echo "=== Iniciando escaneo con Grype ==="
=== Iniciando escaneo con Grype ===
NAME                    INSTALLED                FIXED IN         TYPE    VULNERABILITY        SEVERITY    EPSS           RISK
cryptography            2.8                      39.0.1           python  GHSA-x4qr-2fvf-3mr5  High        86.9% (99th)   64.7
libpython3.8            3.8.10-0ubuntu1~20.04.18 (won't fix)      deb     CVE-2007-4559        Medium      89.4% (99th)   44.7
libpython3.8-dev        3.8.10-0ubuntu1~20.04.18 (won't fix)      deb     CVE-2007-4559        Medium      89.4% (99th)   44.7
libpython3.8-minimal    3.8.10-0ubuntu1~20.04.18 (won't fix)      deb     CVE-2007-4559        Medium      89.4% (99th)   44.7
libpython3.8-stdlib     3.8.10-0ubuntu1~20.04.18 (won't fix)      deb     CVE-2007-4559        Medium      89.4% (99th)   44.7
python3.8               3.8.10-0ubuntu1~20.04.18 (won't fix)      deb     CVE-2007-4559        Medium      89.4% (99th)   44.7
python3.8-dev           3.8.10-0ubuntu1~20.04.18 (won't fix)      deb     CVE-2007-4559        Medium      89.4% (99th)   44.7
python3.8-minimal       3.8.10-0ubuntu1~20.04.18 (won't fix)      deb     CVE-2007-4559        Medium      89.4% (99th)   44.7
werkzeug                0.16.1                   3.0.3            python  GHSA-2g68-c3qc-8985  High        43.6% (97th)   32.7
python-pip-whl          20.0.2-5ubuntu1.11                        deb     CVE-2024-6345        Medium      9.6% (93rd)    4.8
python3-pip             20.0.2-5ubuntu1.11                        deb     CVE-2024-6345        Medium      9.6% (93rd)    4.8
requests                2.23.0                   2.31.0           python  GHSA-j8r2-6x86-q33q  Medium      6.8% (91st)    3.8
login                   1:4.8.1-1ubuntu5.20.04.5                  deb     CVE-2024-56433       Low         6.0% (90th)    1.8
passwd                  1:4.8.1-1ubuntu5.20.04.5                  deb     CVE-2024-56433       Low         6.0% (90th)    1.8
libsystemd0             245.4-4ubuntu3.24                         deb     CVE-2023-26604       Low         5.6% (90th)    1.7
libudev1                245.4-4ubuntu3.24                         deb     CVE-2023-26604       Low         5.6% (90th)    1.7
python-pip-whl          20.0.2-5ubuntu1.11       (won't fix)      deb     CVE-2024-39689       Negligible  25.8% (96th)   1.3
python3-pip             20.0.2-5ubuntu1.11       (won't fix)      deb     CVE-2024-39689       Negligible  25.8% (96th)   1.3
werkzeug                0.16.1                   3.0.6            python  GHSA-f9vj-2wh5-fj8j  Medium      1.4% (80th)    0.8
libexpat1               2.2.9-1ubuntu0.8         (won't fix)      deb     CVE-2023-52425       Medium      1.6% (81st)    0.8
libexpat1-dev           2.2.9-1ubuntu0.8         (won't fix)      deb     CVE-2023-52425       Medium      1.6% (81st)    0.8
urllib3                 1.25.11                  1.26.5           python  GHSA-q2q7-5pp4-w6pg  High        0.9% (75th)    0.7
urllib3                 1.25.11                  1.26.17          python  GHSA-v845-jxx5-vc9f  High        0.9% (76th)    0.7
cryptography            2.8                      42.0.0           python  GHSA-3ww4-gg4f-jr7f  High        0.9% (75th)    0.7
jinja2                  2.11.1                   3.1.4            python  GHSA-h75v-3vvj-5mfj  Medium      1.2% (79th)    0.6
libexpat1               2.2.9-1ubuntu0.8         (won't fix)      deb     CVE-2024-28757       Medium      1.2% (79th)    0.6
libexpat1-dev           2.2.9-1ubuntu0.8         (won't fix)      deb     CVE-2024-28757       Medium      1.2% (79th)    0.6
patch                   2.7.6-6                                   deb     CVE-2018-6952        Negligible  11.8% (93rd)   0.6
cryptography            2.8                      3.2              python  GHSA-hggm-jpg3-v476  High        0.8% (73rd)    0.6
234 packages from EOL distro "ubuntu 20.04" - vulnerability data may be incomplete or outdated; consider upgrading to a supported version
libpython3.8            3.8.10-0ubuntu1~20.04.18 (won't fix)      deb     CVE-2025-6069        Medium      0.9% (75th)    0.4
libpython3.8-dev        3.8.10-0ubuntu1~20.04.18 (won't fix)      deb     CVE-2025-6069        Medium      0.9% (75th)    0.4
libpython3.8-minimal    3.8.10-0ubuntu1~20.04.18 (won't fix)      deb     CVE-2025-6069        Medium      0.9% (75th)    0.4
libpython3.8-stdlib     3.8.10-0ubuntu1~20.04.18 (won't fix)      deb     CVE-2025-6069        Medium      0.9% (75th)    0.4
python3.8               3.8.10-0ubuntu1~20.04.18 (won't fix)      deb     CVE-2025-6069        Medium      0.9% (75th)    0.4
```

Nota. Elaboración propia. Captura de pantalla del reporte generado por Grype durante la ejecución del pipeline DevSecOps, donde se presentan las vulnerabilidades detectadas en las dependencias de la aplicación, incluyendo la severidad, los identificadores CVE y las versiones corregidas disponibles.

Posteriormente, el pipeline ejecuta Grype para realizar un análisis de las dependencias incluidas en la imagen Docker. La herramienta identifica vulnerabilidades conocidas mediante los identificadores CVE, clasifica su nivel de severidad y muestra las versiones en

las que los problemas han sido corregidos, proporcionando información complementaria al análisis estructural realizado por Trivy.

Este penúltimo paso de limpieza se ejecuta al final de todo el trabajo con la finalidad de garantizar la seguridad del servidor local Rocky Linux.

Figura 46

Limpieza del entorno al finalizar el pipeline

Nota. Elaboración propia. Captura de pantalla de la etapa **Post Descargar código fuente** del pipeline DevSecOps, donde GitHub Actions elimina configuraciones temporales y realiza la limpieza del entorno de ejecución para finalizar correctamente el workflow.

Al concluir las tareas de construcción y análisis de vulnerabilidades, GitHub Actions ejecuta automáticamente una etapa de limpieza del entorno. Durante este proceso se eliminan configuraciones temporales de Git y otros elementos utilizados durante la ejecución del workflow, garantizando que el runner quede preparado para futuras ejecuciones del pipeline.

En el siguiente paso (final), se muestra los registros de log y cierra el proceso, adicional nos muestra una advertencia de obsolescencia de herramientas.

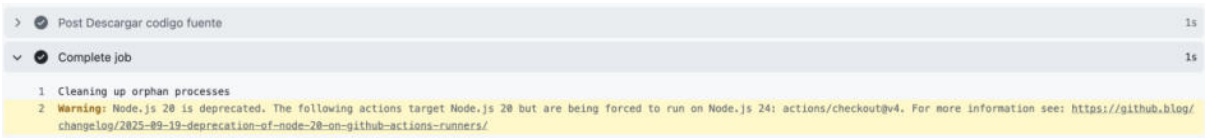
Esta advertencia en nuestro pipeline esta solventada es solo informativa, por cuanto en el archivo Yet Another Markup Language (YAML) se usa la acción actions/checkout@v4 ya

que la acción actions/checkout@v3 está construida sobre Node.js 20 y GitHub la ha marcado como obsoleto.

Adicional podemos observar que el proceso se completa con éxito en un tiempo de 8 minutos y 8 segundos. (Esto se visualiza bajo el título escaneo-seguridad).

Figura 47

Finalización exitosa del pipeline DevSecOps



Nota. Elaboración propia. Captura de pantalla de la etapa **Complete job** del pipeline DevSecOps en GitHub Actions, donde se evidencia la finalización exitosa del workflow y la ejecución completa de todas las etapas definidas.

La última etapa del pipeline corresponde a la finalización del workflow. GitHub Actions verifica que todas las tareas hayan concluido correctamente, ejecuta las acciones de limpieza correspondientes y marca el proceso como exitoso, confirmando la correcta automatización del flujo DevSecOps implementado.

CAPÍTULO 4

4. Análisis de resultados

4.1. Pruebas de concepto

Con la finalidad de evaluar la efectividad y viabilidad del modelo Development, Security and Operations (DevSecOps) con arquitectura híbrida planteado en esta investigación, se realizó un entorno de pruebas controlado bajo el hipervisor VMware ESXi, este entorno cuenta con un servidor de compilación Rocky Linux 10 el cual se encuentra atado al plano de control de la nube pública GitHub a través del agente Self-Hosted Runner.

A continuación, se exponen los resultados obtenidos en la ejecución de la prueba de concepto diseñada intencionalmente a partir de parámetro inseguros establecidos para la aplicación y construcción de la imagen del contenedor.

La primera validación consistió en comprobar la sincronización del control en la nube ante el evento de integración continua (git push) desde el computador local con Visual Studio Code, el flujo se ejecutó de acuerdo a lo esperado enviando las tareas de cómputo al servidor local Rocky Linux lo que aseguro que el almacenamiento de los datos se mantuviera de forma local (on-premises), y así preservar la privacidad del código fuente.

Como resultado se obtuvo el aislamiento en un dierctorio temporal local del servidor (`/home/grupo7/actions-runner/_work/_temp/`) donde se clonaron los archivos fuente, para posteriormente generar la imagen del contenedor y los respectivos escaneos de seguridad.

4.2. Análisis de Resultados.

La herramienta Trivy fue configurada para realizar un escaneo de caja blanca sobre la imagen construida, aplicando la directiva estricta (`--exit-code 1`) que es vulnerabilidad de severidad critica bajo el estándar Common Vulnerability Scoring System CVSS v3.1

A continuación, se presenta el log de escaneo.

```
2026-06-15T17:32:29.7565843Z === Iniciando escaneo con Trivy ==?
2026-06-15T17:32:29.9482789Z 2026-06-15T13:32:29-04:00     INFO
[vuln] Vulnerability scanning is enabled
2026-06-15T17:32:29.9484494Z 2026-06-15T13:32:29-04:00     INFO
[secret] Secret scanning is enabled
2026-06-15T17:32:29.9487167Z 2026-06-15T13:32:29-04:00     INFO
[secret] If your scanning is slow, please try '--scanners vuln' to
disable secret scanning
2026-06-15T17:32:29.9501880Z 2026-06-15T13:32:29-04:00     INFO
[secret] Please see
https://trivy.dev/docs/v0.71/guide/scanner/secret#recommendation for faster
secret detection
2026-06-15T17:32:43.0553160Z 2026-06-15T13:32:43-04:00     INFO
[python] Licenses acquired from one or more METADATA files may be
subject to additional terms. Use `--debug` flag to see all affected
packages.
2026-06-15T17:32:49.2695576Z 2026-06-15T13:32:49-04:00     INFO
Detected OS family="ubuntu" version="20.04"
2026-06-15T17:32:49.2698379Z 2026-06-15T13:32:49-04:00     INFO
[ubuntu] Detecting vulnerabilities...    os_version="20.04"
pkg_num=219
2026-06-15T17:32:49.3513729Z 2026-06-15T13:32:49-04:00     INFO
Number of language-specific files  num=1
2026-06-15T17:32:49.3515388Z 2026-06-15T13:32:49-04:00     INFO
[python-pkg] Detecting vulnerabilities...
2026-06-15T17:32:49.3645543Z 2026-06-15T13:32:49-04:00     WARN  This
OS version is no longer supported by the distributionfamily="ubuntu"
version="20.04"
2026-06-15T17:32:49.3648721Z 2026-06-15T13:32:49-04:00     WARN  The
vulnerability detection may be insufficient because security updates are
not provided
2026-06-15T17:32:49.4557267Z
2026-06-15T17:32:49.4557783Z Report Summary
2026-06-15T17:32:49.4558281Z
2026-06-15T17:32:49.4559914Z
┌──────────────────────────────────────────────────────────────────────────
──┬──────────────┬──────────────────────┬─────────┐
2026-06-15T17:32:49.4562458Z │
Target                                  │     Type     │ Vulnerabilities │
Secrets │
2026-06-15T17:32:49.4564867Z
├──────────────────────────────────────────────────────────────────────────
──┼──────────────┼──────────────────────┼─────────┤
2026-06-15T17:32:49.4567857Z │ app-vulnerable:latest (ubuntu 20.04)
│   ubuntu   │        0        │    -    │
2026-06-15T17:32:49.4570432Z
├──────────────────────────────────────────────────────────────────────────
──┼──────────────┼──────────────────────┼─────────┤
2026-06-15T17:32:49.4573124Z │ usr/local/lib/python3.8/dist-
packages/Flask-1.1.2.dist-info/METADATA        │ python-pkg │        0
│    -    │
2026-06-15T17:32:49.4575904Z
├──────────────────────────────────────────────────────────────────────────
──┼──────────────┼──────────────────────┼─────────┤
2026-06-15T17:32:49.4578987Z │ usr/local/lib/python3.8/dist-
packages/Jinja2-2.11.1.dist-info/METADATA      │ python-pkg │        0
│    -    │
```

```
      2026-06-15T17:32:49.4581834Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4584538Z | usr/local/lib/python3.8/dist-
packages/MarkupSafe-2.1.5.dist-info/METADATA   | python-pkg |        0
|     -    |
      2026-06-15T17:32:49.4587630Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4590307Z | usr/local/lib/python3.8/dist-
packages/Werkzeug-0.16.1.dist-info/METADATA    | python-pkg |        0
|     -    |
      2026-06-15T17:32:49.4594062Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4597188Z | usr/local/lib/python3.8/dist-
packages/certifi-2026.5.20.dist-info/METADATA  | python-pkg |        0
|     -    |
      2026-06-15T17:32:49.4600008Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4603270Z | usr/local/lib/python3.8/dist-
packages/cffi-1.17.1.dist-info/METADATA        | python-pkg |        0
|     -    |
      2026-06-15T17:32:49.4606047Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4609150Z | usr/local/lib/python3.8/dist-
packages/chardet-3.0.4.dist-info/METADATA      | python-pkg |        0
|     -    |
      2026-06-15T17:32:49.4611993Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4614614Z | usr/local/lib/python3.8/dist-
packages/click-8.1.8.dist-info/METADATA        | python-pkg |        0
|     -    |
      2026-06-15T17:32:49.4617619Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4620341Z | usr/local/lib/python3.8/dist-
packages/cryptography-2.8.dist-info/METADATA   | python-pkg |        0
|     -    |
      2026-06-15T17:32:49.4623131Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4625798Z | usr/local/lib/python3.8/dist-
packages/idna-2.10.dist-info/METADATA          | python-pkg |        0
|     -    |
      2026-06-15T17:32:49.4628805Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4632090Z | usr/local/lib/python3.8/dist-
packages/itsdangerous-2.2.0.dist-info/METADATA | python-pkg |        0
|     -    |
      2026-06-15T17:32:49.4634975Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4637991Z | usr/local/lib/python3.8/dist-
packages/pycparser-2.23.dist-info/METADATA     | python-pkg |        0
|     -    |
```

```
      2026-06-15T17:32:49.4641489Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4644066Z | usr/local/lib/python3.8/dist-
packages/requests-2.23.0.dist-info/METADATA    | python-pkg |        0
|    -    |
      2026-06-15T17:32:49.4647097Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4649727Z | usr/local/lib/python3.8/dist-
packages/six-1.17.0.dist-info/METADATA         | python-pkg |        0
|    -    |
      2026-06-15T17:32:49.4652551Z
├──────────────────────────────────────────────────────────────────────────────
──┼──────────────┼──────────────────────┼──────────┤
      2026-06-15T17:32:49.4655149Z | usr/local/lib/python3.8/dist-
packages/urllib3-1.25.11.dist-info/METADATA    | python-pkg |        0
|    -    |
      2026-06-15T17:32:49.4658448Z
└──────────────────────────────────────────────────────────────────────────────
──┴──────────────┴──────────────────────┴──────────┘
      2026-06-15T17:32:49.4659929Z Legend:
      2026-06-15T17:32:49.4660567Z - '-': Not scanned
      2026-06-15T17:32:49.4661379Z - '0': Clean (no security findings
detected)
      2026-06-15T17:32:49.4662078Z
```

Los resultados arrojaron las siguientes métricas.

- Alerta de obsolescencia: Se emitió una alerta WARM indicando que el sistema operativo de la imagen en el archivo Dockerfile (Ubuntu 20.04), ya no cuenta con soporte, lo que impacto directamente en la detección de parches de seguridad activos.
- Falsos Negativos: A pesar de realizar una construcción del contenedor sin restricciones en el privilegio, es decir se ejecuta con el usuario root, la herramienta Trivy reportó o vulnerabilidades en las capas del software del sistema operativo. Lo cual demostró empíricamente como la imagen base obsoleta es capaz de generar reportes sesgados ya que sus bases de datos no generar fallos.

Con la evaluación de auditoría realizada por la herramienta Grype la cual está enfocada a la inspección de dependencias a nivel de aplicación se evaluaron los paquetes de Python proporcionados en la prueba de concepto, esta herramienta demostró una mayor efectividad a nivel lógico.

A continuación, se presentan los resultados de esta prueba:

```
      2026-06-15T17:32:49.4799006Z ##[group]Run echo "=== Iniciando escaneo
con Grype ==="
      2026-06-15T17:32:49.4800449Z [36;1mecho "=== Iniciando escaneo con
Grype ==="[0m
      2026-06-15T17:32:49.4802011Z [36;1m# Ejecuta el escaneo de software y
muestra los resultados detallados en la consola[0m
      2026-06-15T17:32:49.4803545Z [36;1mgrype app-vulnerable:latest[0m
      2026-06-15T17:32:49.4820283Z shell: /usr/bin/bash -e {0}
      2026-06-15T17:32:49.4821121Z ##[endgroup]
      2026-06-15T17:32:49.4908187Z === Iniciando escaneo con Grype ===
      2026-06-15T17:36:00.9688366Z NAME                        INSTALLED
FIXED IN          TYPE    VULNERABILITY        SEVERITY    EPSS
RISK
      2026-06-15T17:36:00.9691211Z cryptography                2.8
39.0.1            python  GHSA-x4qr-2fvf-3mr5  High        86.9% (99th)
64.7
      2026-06-15T17:36:00.9693700Z libpython3.8                3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2007-4559        Medium
89.4% (99th)   44.7
      2026-06-15T17:36:00.9696365Z libpython3.8-dev            3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2007-4559        Medium
89.4% (99th)   44.7
      2026-06-15T17:36:00.9699185Z libpython3.8-minimal        3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2007-4559        Medium
89.4% (99th)   44.7
      2026-06-15T17:36:00.9701708Z libpython3.8-stdlib         3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2007-4559        Medium
89.4% (99th)   44.7
      2026-06-15T17:36:00.9704128Z python3.8                   3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2007-4559        Medium
89.4% (99th)   44.7
      2026-06-15T17:36:00.9706475Z python3.8-dev               3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2007-4559        Medium
89.4% (99th)   44.7
      2026-06-15T17:36:00.9710360Z python3.8-minimal           3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2007-4559        Medium
89.4% (99th)   44.7
      2026-06-15T17:36:00.9712879Z werkzeug                    0.16.1
3.0.3             python  GHSA-2g68-c3qc-8985  High        43.6% (97th)
32.7
      2026-06-15T17:36:00.9715138Z python-pip-whl              20.0.2-
5ubuntu1.11                         deb     CVE-2024-6345        Medium
9.6% (93rd)    4.8
      2026-06-15T17:36:00.9718173Z python3-pip                 20.0.2-
5ubuntu1.11                         deb     CVE-2024-6345        Medium
9.6% (93rd)    4.8
      2026-06-15T17:36:00.9720469Z requests                    2.23.0
2.31.0            python  GHSA-j8r2-6x86-q33q  Medium      6.8% (91st)
3.8
      2026-06-15T17:36:00.9722660Z login                       1:4.8.1-
1ubuntu5.20.04.5                    deb     CVE-2024-56433       Low
6.0% (90th)    1.8
      2026-06-15T17:36:00.9724876Z passwd                      1:4.8.1-
1ubuntu5.20.04.5                    deb     CVE-2024-56433       Low
6.0% (90th)    1.8
      2026-06-15T17:36:00.9727482Z libsystemd0                 245.4-
4ubuntu3.24                           deb     CVE-2023-26604       Low
5.6% (90th)    1.7
      2026-06-15T17:36:00.9729711Z libudev1                    245.4-
4ubuntu3.24                           deb     CVE-2023-26604       Low
5.6% (90th)    1.7
```

```
      2026-06-15T17:36:00.9732019Z python-pip-whl             20.0.2-
5ubuntu1.11        (won't fix)       deb     CVE-2024-39689
Negligible  25.8% (96th)   1.3
      2026-06-15T17:36:00.9734421Z python3-pip                20.0.2-
5ubuntu1.11        (won't fix)       deb     CVE-2024-39689
Negligible  25.8% (96th)   1.3
      2026-06-15T17:36:00.9736987Z werkzeug                   0.16.1
3.0.6             python  GHSA-f9vj-2wh5-fj8j  Medium      1.4% (80th)
0.8
      2026-06-15T17:36:00.9739288Z libexpat1                  2.2.9-
1ubuntu0.8          (won't fix)       deb     CVE-2023-52425       Medium
1.6% (81st)    0.8
      2026-06-15T17:36:00.9742568Z libexpat1-dev              2.2.9-
1ubuntu0.8          (won't fix)       deb     CVE-2023-52425       Medium
1.6% (81st)    0.8
      2026-06-15T17:36:00.9745011Z urllib3                    1.25.11
1.26.5            python  GHSA-q2q7-5pp4-w6pg  High        0.9% (75th)
0.7
      2026-06-15T17:36:00.9748573Z urllib3                    1.25.11
1.26.17           python  GHSA-v845-jxx5-vc9f  High        0.9% (76th)
0.7
      2026-06-15T17:36:00.9750839Z cryptography               2.8
42.0.0            python  GHSA-3ww4-gg4f-jr7f  High        0.9% (75th)
0.7
      2026-06-15T17:36:00.9753054Z jinja2                     2.11.1
3.1.4             python  GHSA-h75v-3vvj-5mfj  Medium      1.2% (79th)
0.6
      2026-06-15T17:36:00.9755332Z libexpat1                  2.2.9-
1ubuntu0.8          (won't fix)       deb     CVE-2024-28757       Medium
1.2% (79th)    0.6
      2026-06-15T17:36:00.9757969Z libexpat1-dev              2.2.9-
1ubuntu0.8          (won't fix)       deb     CVE-2024-28757       Medium
1.2% (79th)    0.6
      2026-06-15T17:36:00.9760186Z patch                      2.7.6-6
deb     CVE-2018-6952        Negligible  11.8% (93rd)   0.6
      2026-06-15T17:36:00.9762340Z cryptography               2.8
3.2               python  GHSA-hggm-jpg3-v476  High        0.8% (73rd)
0.6
      2026-06-15T17:36:00.9765257Z 234 packages from EOL distro "ubuntu
20.04" - vulnerability data may be incomplete or outdated; consider
upgrading to a supported version
      2026-06-15T17:36:00.9768545Z libpython3.8               3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2025-6069        Medium
0.9% (75th)    0.4
      2026-06-15T17:36:00.9771009Z libpython3.8-dev           3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2025-6069        Medium
0.9% (75th)    0.4
      2026-06-15T17:36:00.9774080Z libpython3.8-minimal       3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2025-6069        Medium
0.9% (75th)    0.4
      2026-06-15T17:36:00.9778327Z libpython3.8-stdlib        3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2025-6069        Medium
0.9% (75th)    0.4
      2026-06-15T17:36:00.9780844Z python3.8                  3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2025-6069        Medium
0.9% (75th)    0.4
      2026-06-15T17:36:00.9783189Z python3.8-dev              3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2025-6069        Medium
0.9% (75th)    0.4
```

```
      2026-06-15T17:36:00.9785616Z python3.8-minimal          3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2025-6069        Medium
0.9% (75th)    0.4
      2026-06-15T17:36:00.9788274Z cryptography               2.8
39.0.1            python  GHSA-w7pp-m8wf-vj6r  Medium      0.7% (72nd)
0.4
      2026-06-15T17:36:00.9790606Z libexpat1                  2.2.9-
1ubuntu0.8          (won't fix)       deb     CVE-2024-8176        Medium
0.8% (74th)    0.4
      2026-06-15T17:36:00.9793158Z libexpat1-dev              2.2.9-
1ubuntu0.8          (won't fix)       deb     CVE-2024-8176        Medium
0.8% (74th)    0.4
      2026-06-15T17:36:00.9795400Z idna                       2.10
3.7               python  GHSA-jjg7-2v4v-x38h  Medium      0.7% (72nd)
0.4
      2026-06-15T17:36:00.9797890Z libperl5.30                5.30.0-
9ubuntu0.5          (won't fix)       deb     CVE-2023-31486       Medium
0.8% (74th)    0.4
      2026-06-15T17:36:00.9800158Z perl                       5.30.0-
9ubuntu0.5          (won't fix)       deb     CVE-2023-31486       Medium
0.8% (74th)    0.4
      2026-06-15T17:36:00.9802412Z perl-base                  5.30.0-
9ubuntu0.5          (won't fix)       deb     CVE-2023-31486       Medium
0.8% (74th)    0.4
      2026-06-15T17:36:00.9804810Z perl-modules-5.30          5.30.0-
9ubuntu0.5          (won't fix)       deb     CVE-2023-31486       Medium
0.8% (74th)    0.4
      2026-06-15T17:36:00.9808036Z python-pip-whl             20.0.2-
5ubuntu1.11                           deb     CVE-2024-3651        Medium
0.7% (72nd)    0.3
      2026-06-15T17:36:00.9810408Z python3-pip                20.0.2-
5ubuntu1.11                           deb     CVE-2024-3651        Medium
0.7% (72nd)    0.3
      2026-06-15T17:36:00.9812610Z jinja2                     2.11.1
3.1.5             python  GHSA-q2x7-8rv6-6q7h  Medium      0.6% (69th)
0.3
      2026-06-15T17:36:00.9814774Z werkzeug                   0.16.1
2.2.3             python  GHSA-xg9f-g7g7-2323  High        0.4% (59th)
0.3
      2026-06-15T17:36:00.9817279Z libpython3.8               3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-1795        Low
0.8% (73rd)    0.2
      2026-06-15T17:36:00.9819594Z libpython3.8-dev           3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-1795        Low
0.8% (73rd)    0.2
      2026-06-15T17:36:00.9822146Z libpython3.8-minimal       3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-1795        Low
0.8% (73rd)    0.2
      2026-06-15T17:36:00.9824554Z libpython3.8-stdlib        3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-1795        Low
0.8% (73rd)    0.2
      2026-06-15T17:36:00.9827125Z python3.8                  3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-1795        Low
0.8% (73rd)    0.2
      2026-06-15T17:36:00.9829392Z python3.8-dev              3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-1795        Low
0.8% (73rd)    0.2
      2026-06-15T17:36:00.9831717Z python3.8-minimal          3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-1795        Low
0.8% (73rd)    0.2
```

```
      2026-06-15T17:36:00.9833998Z libgcrypt20                1.8.5-
5ubuntu1.1                            deb     CVE-2024-2236        Low
0.7% (71st)    0.2
      2026-06-15T17:36:00.9837170Z python-pip-whl             20.0.2-
5ubuntu1.11        (won't fix)       deb     CVE-2018-20225
Negligible  3.7% (88th)    0.2
      2026-06-15T17:36:00.9839608Z python3-pip                20.0.2-
5ubuntu1.11        (won't fix)       deb     CVE-2018-20225
Negligible  3.7% (88th)    0.2
      2026-06-15T17:36:00.9841879Z flask                      1.1.2
2.2.5             python  GHSA-m2qf-hxjv-5gpq  High        0.2% (44th)
0.2
      2026-06-15T17:36:00.9844102Z libc-bin                   2.31-
0ubuntu9.17          (won't fix)       deb     CVE-2019-1010024     Low
0.5% (66th)    0.2
      2026-06-15T17:36:00.9846471Z libc-dev-bin               2.31-
0ubuntu9.18          (won't fix)       deb     CVE-2019-1010024     Low
0.5% (66th)    0.2
      2026-06-15T17:36:00.9849017Z libc6                      2.31-
0ubuntu9.18          (won't fix)       deb     CVE-2019-1010024     Low
0.5% (66th)    0.2
      2026-06-15T17:36:00.9851428Z libc6-dev                  2.31-
0ubuntu9.18          (won't fix)       deb     CVE-2019-1010024     Low
0.5% (66th)    0.2
      2026-06-15T17:36:00.9853967Z libsystemd0                245.4-
4ubuntu3.24                           deb     CVE-2023-7008        Low
0.5% (65th)    0.1
      2026-06-15T17:36:00.9856274Z libudev1                   245.4-
4ubuntu3.24                           deb     CVE-2023-7008        Low
0.5% (65th)    0.1
      2026-06-15T17:36:00.9858803Z libtasn1-6                 4.16.0-
2ubuntu0.1                            deb     CVE-2021-46848       Low
0.4% (61st)    0.1
      2026-06-15T17:36:00.9861014Z jinja2                     2.11.1
2.11.3            python  GHSA-g3rq-g295-4j3m  Medium      0.2% (43rd)
0.1
      2026-06-15T17:36:00.9863202Z requests                   2.23.0
2.32.4            python  GHSA-9hjg-9r4m-mvj7  Medium      0.2% (43rd)
0.1
      2026-06-15T17:36:00.9865508Z cryptography               2.8
42.0.2            python  GHSA-9v9h-cgj8-h64p  Medium      0.2% (42nd)
0.1
      2026-06-15T17:36:00.9868652Z libpython3.8               3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-4516        Medium
0.2% (43rd)    0.1
      2026-06-15T17:36:00.9871040Z libpython3.8-dev           3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-4516        Medium
0.2% (43rd)    0.1
      2026-06-15T17:36:00.9873428Z libpython3.8-minimal       3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-4516        Medium
0.2% (43rd)    0.1
      2026-06-15T17:36:00.9875885Z libpython3.8-stdlib        3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-4516        Medium
0.2% (43rd)    0.1
      2026-06-15T17:36:00.9878472Z python3.8                  3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-4516        Medium
0.2% (43rd)    0.1
      2026-06-15T17:36:00.9880857Z python3.8-dev              3.8.10-
0ubuntu1~20.04.18                     deb     CVE-2025-4516        Medium
0.2% (43rd)    0.1
```

```
      2026-06-15T17:36:00.9883245Z python3.8-minimal          3.8.10-
0ubuntu1~20.04.18                    deb     CVE-2025-4516        Medium
0.2% (43rd)    0.1
      2026-06-15T17:36:00.9885547Z urllib3                    1.25.11
1.26.19           python  GHSA-34jh-p97f-mpxf  Medium      0.2% (44th)
0.1
      2026-06-15T17:36:00.9888018Z cpp                        4:9.3.0-
1ubuntu2          (won't fix)       deb     CVE-2020-23026       Low
0.3% (56th)    0.1
      2026-06-15T17:36:00.9890584Z cpp-9                      9.4.0-
1ubuntu1~20.04.2    (won't fix)       deb     CVE-2020-23026       Low
0.3% (56th)    0.1
      2026-06-15T17:36:00.9892789Z g++                        4:9.3.0-
1ubuntu2          (won't fix)       deb     CVE-2020-23026       Low
0.3% (56th)    0.1
      2026-06-15T17:36:00.9895021Z g++-9                      9.4.0-
1ubuntu1~20.04.2    (won't fix)       deb     CVE-2020-23026       Low
0.3% (56th)    0.1
      2026-06-15T17:36:00.9898032Z gcc                        4:9.3.0-
1ubuntu2          (won't fix)       deb     CVE-2020-23026       Low
0.3% (56th)    0.1
      2026-06-15T17:36:00.9900256Z gcc-9                      9.4.0-
1ubuntu1~20.04.2    (won't fix)       deb     CVE-2020-23026       Low
0.3% (56th)    0.1
      2026-06-15T17:36:00.9902524Z gcc-9-base                 9.4.0-
1ubuntu1~20.04.2    (won't fix)       deb     CVE-2020-23026       Low
0.3% (56th)    0.1
      2026-06-15T17:36:00.9904825Z libasan5                   9.4.0-
1ubuntu1~20.04.2    (won't fix)       deb     CVE-2020-23026       Low
0.3% (56th)    0.1
      2026-06-15T17:36:00.9907437Z libgcc-9-dev               9.4.0-
1ubuntu1~20.04.2    (won't fix)       deb     CVE-2020-23026       Low
0.3% (56th)    0.1
      2026-06-15T17:36:00.9909824Z libstdc++-9-dev            9.4.0-
1ubuntu1~20.04.2    (won't fix)       deb     CVE-2020-23026       Low
0.3% (56th)    0.1
      2026-06-15T17:36:00.9912401Z libpython3.8               3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2021-23336       Low
0.3% (54th)    < 0.1
      2026-06-15T17:36:00.9914816Z libpython3.8-dev           3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2021-23336       Low
0.3% (54th)    < 0.1
      2026-06-15T17:36:00.9917535Z libpython3.8-minimal       3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2021-23336       Low
0.3% (54th)    < 0.1
      2026-06-15T17:36:00.9920048Z libpython3.8-stdlib        3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2021-23336       Low
0.3% (54th)    < 0.1
      2026-06-15T17:36:00.9922466Z python3.8                  3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2021-23336       Low
0.3% (54th)    < 0.1
      2026-06-15T17:36:00.9924810Z python3.8-dev              3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2021-23336       Low
0.3% (54th)    < 0.1
      2026-06-15T17:36:00.9927479Z python3.8-minimal          3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2021-23336       Low
0.3% (54th)    < 0.1
      2026-06-15T17:36:00.9930439Z libc-bin                   2.31-
0ubuntu9.17          (won't fix)       deb     CVE-2019-1010023     Low
0.3% (53rd)    < 0.1
```

```
      2026-06-15T17:36:00.9932845Z libc-dev-bin               2.31-
0ubuntu9.18          (won't fix)       deb     CVE-2019-1010023     Low
0.3% (53rd)    < 0.1
      2026-06-15T17:36:00.9935126Z libc6                     2.31-
0ubuntu9.18          (won't fix)       deb     CVE-2019-1010023     Low
0.3% (53rd)    < 0.1
      2026-06-15T17:36:00.9937640Z libc6-dev                 2.31-
0ubuntu9.18          (won't fix)       deb     CVE-2019-1010023     Low
0.3% (53rd)    < 0.1
      2026-06-15T17:36:00.9939979Z krb5-locales              1.17-
6ubuntu4.11          (won't fix)       deb     CVE-2018-5709
Negligible  1.6% (82nd)    < 0.1
      2026-06-15T17:36:00.9942512Z libgssapi-krb5-2          1.17-
6ubuntu4.11          (won't fix)       deb     CVE-2018-5709
Negligible  1.6% (82nd)    < 0.1
      2026-06-15T17:36:00.9944942Z libk5crypto3              1.17-
6ubuntu4.11          (won't fix)       deb     CVE-2018-5709
Negligible  1.6% (82nd)    < 0.1
      2026-06-15T17:36:00.9947532Z libkrb5-3                 1.17-
6ubuntu4.11          (won't fix)       deb     CVE-2018-5709
Negligible  1.6% (82nd)    < 0.1
      2026-06-15T17:36:00.9949980Z libkrb5support0           1.17-
6ubuntu4.11          (won't fix)       deb     CVE-2018-5709
Negligible  1.6% (82nd)    < 0.1
      2026-06-15T17:36:00.9952370Z libsqlite3-0              3.31.1-
4ubuntu0.7         (won't fix)       deb     CVE-2021-45346       Low
0.3% (50th)    < 0.1
      2026-06-15T17:36:00.9954625Z jinja2                    2.11.1
3.1.3             python  GHSA-h5c8-rqwp-cp95  Medium      0.2% (35th)    <
0.1
      2026-06-15T17:36:00.9957113Z werkzeug                  0.16.1
2.2.3             python  GHSA-px8h-6qxv-m22q  Low         0.3% (50th)    <
0.1
      2026-06-15T17:36:00.9959904Z binutils                  2.34-
6ubuntu1.11                       deb     CVE-2017-13716       Low
0.2% (47th)    < 0.1
      2026-06-15T17:36:00.9962158Z binutils-common           2.34-
6ubuntu1.11                       deb     CVE-2017-13716       Low
0.2% (47th)    < 0.1
      2026-06-15T17:36:00.9964548Z binutils-x86-64-linux-gnu  2.34-
6ubuntu1.11                       deb     CVE-2017-13716       Low
0.2% (47th)    < 0.1
      2026-06-15T17:36:00.9967184Z libbinutils               2.34-
6ubuntu1.11                       deb     CVE-2017-13716       Low
0.2% (47th)    < 0.1
      2026-06-15T17:36:00.9969472Z libctf-nobfd0             2.34-
6ubuntu1.11                       deb     CVE-2017-13716       Low
0.2% (47th)    < 0.1
      2026-06-15T17:36:00.9971838Z libctf0                   2.34-
6ubuntu1.11                       deb     CVE-2017-13716       Low
0.2% (47th)    < 0.1
      2026-06-15T17:36:00.9974083Z libc-bin                  2.31-
0ubuntu9.17          (won't fix)       deb     CVE-2018-20796
Negligible  1.3% (80th)    < 0.1
      2026-06-15T17:36:00.9976459Z libc-dev-bin              2.31-
0ubuntu9.18          (won't fix)       deb     CVE-2018-20796
Negligible  1.3% (80th)    < 0.1
      2026-06-15T17:36:00.9979037Z libc6                     2.31-
0ubuntu9.18          (won't fix)       deb     CVE-2018-20796
Negligible  1.3% (80th)    < 0.1
```

```
      2026-06-15T17:36:00.9981320Z libc6-dev                  2.31-
0ubuntu9.18          (won't fix)       deb     CVE-2018-20796
Negligible  1.3% (80th)    < 0.1
      2026-06-15T17:36:00.9983602Z jinja2                     2.11.1
3.1.6             python  GHSA-cpwx-vrp4-4pq7  Medium      0.1% (30th)    <
0.1
      2026-06-15T17:36:00.9985856Z python-pip-whl             20.0.2-
5ubuntu1.11                          deb     CVE-2025-47273       Medium
0.1% (30th)    < 0.1
      2026-06-15T17:36:00.9988617Z python3-pip                20.0.2-
5ubuntu1.11                          deb     CVE-2025-47273       Medium
0.1% (30th)    < 0.1
      2026-06-15T17:36:00.9991414Z libsystemd0                245.4-
4ubuntu3.24                           deb     CVE-2025-4598        Medium
0.1% (29th)    < 0.1
      2026-06-15T17:36:00.9993682Z libudev1                   245.4-
4ubuntu3.24                           deb     CVE-2025-4598        Medium
0.1% (29th)    < 0.1
      2026-06-15T17:36:00.9995850Z cpp-9                      9.4.0-
1ubuntu1~20.04.2                      deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:00.9998278Z g++-9                      9.4.0-
1ubuntu1~20.04.2                      deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0000444Z gcc-10-base                10.5.0-
1ubuntu1~20.04                        deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0002686Z gcc-9                      9.4.0-
1ubuntu1~20.04.2                      deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0004878Z gcc-9-base                 9.4.0-
1ubuntu1~20.04.2                      deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0007345Z libasan5                   9.4.0-
1ubuntu1~20.04.2                      deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0009559Z libatomic1                 10.5.0-
1ubuntu1~20.04                        deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0011889Z libcc1-0                   10.5.0-
1ubuntu1~20.04                        deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0014107Z libgcc-9-dev               9.4.0-
1ubuntu1~20.04.2                      deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0016320Z libgcc-s1                  10.5.0-
1ubuntu1~20.04                        deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0019371Z libgomp1                   10.5.0-
1ubuntu1~20.04                        deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0021598Z libitm1                    10.5.0-
1ubuntu1~20.04                        deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0023775Z liblsan0                   10.5.0-
1ubuntu1~20.04                        deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0026018Z libquadmath0               10.5.0-
1ubuntu1~20.04                        deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
```

```
      2026-06-15T17:36:01.0028566Z libstdc++-9-dev            9.4.0-
1ubuntu1~20.04.2                      deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0030790Z libstdc++6                 10.5.0-
1ubuntu1~20.04                        deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0032983Z libtsan0                   10.5.0-
1ubuntu1~20.04                        deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0035215Z libubsan1                  10.5.0-
1ubuntu1~20.04                        deb     CVE-2023-4039        Low
0.2% (40th)    < 0.1
      2026-06-15T17:36:01.0037665Z binutils                   2.34-
6ubuntu1.11                           deb     CVE-2025-1179        Medium
0.1% (28th)    < 0.1
      2026-06-15T17:36:01.0039910Z binutils-common            2.34-
6ubuntu1.11                           deb     CVE-2025-1179        Medium
0.1% (28th)    < 0.1
      2026-06-15T17:36:01.0042432Z binutils-x86-64-linux-gnu  2.34-
6ubuntu1.11                           deb     CVE-2025-1179        Medium
0.1% (28th)    < 0.1
      2026-06-15T17:36:01.0044829Z libbinutils                2.34-
6ubuntu1.11                           deb     CVE-2025-1179        Medium
0.1% (28th)    < 0.1
      2026-06-15T17:36:01.0047312Z libctf-nobfd0              2.34-
6ubuntu1.11                           deb     CVE-2025-1179        Medium
0.1% (28th)    < 0.1
      2026-06-15T17:36:01.0050113Z libctf0                    2.34-
6ubuntu1.11                           deb     CVE-2025-1179        Medium
0.1% (28th)    < 0.1
      2026-06-15T17:36:01.0052385Z libc-bin                   2.31-
0ubuntu9.17          (won't fix)      deb     CVE-2019-1010022     Low
0.1% (35th)    < 0.1
      2026-06-15T17:36:01.0054736Z libc-dev-bin               2.31-
0ubuntu9.18          (won't fix)      deb     CVE-2019-1010022     Low
0.1% (35th)    < 0.1
      2026-06-15T17:36:01.0057315Z libc6                      2.31-
0ubuntu9.18          (won't fix)      deb     CVE-2019-1010022     Low
0.1% (35th)    < 0.1
      2026-06-15T17:36:01.0059594Z libc6-dev                  2.31-
0ubuntu9.18          (won't fix)      deb     CVE-2019-1010022     Low
0.1% (35th)    < 0.1
      2026-06-15T17:36:01.0061836Z binutils                   2.34-
6ubuntu1.11                           deb     CVE-2019-1010204     Low
0.1% (35th)    < 0.1
      2026-06-15T17:36:01.0064109Z binutils-common            2.34-
6ubuntu1.11                           deb     CVE-2019-1010204     Low
0.1% (35th)    < 0.1
      2026-06-15T17:36:01.0066844Z binutils-x86-64-linux-gnu  2.34-
6ubuntu1.11                           deb     CVE-2019-1010204     Low
0.1% (35th)    < 0.1
      2026-06-15T17:36:01.0069363Z libbinutils                2.34-
6ubuntu1.11                           deb     CVE-2019-1010204     Low
0.1% (35th)    < 0.1
      2026-06-15T17:36:01.0071780Z libctf-nobfd0              2.34-
6ubuntu1.11                           deb     CVE-2019-1010204     Low
0.1% (35th)    < 0.1
      2026-06-15T17:36:01.0073995Z libctf0                    2.34-
6ubuntu1.11                           deb     CVE-2019-1010204     Low
0.1% (35th)    < 0.1
```

```
      2026-06-15T17:36:01.0076174Z coreutils                  8.30-3ubuntu2
deb     CVE-2025-5278        Low         0.1% (34th)    < 0.1
      2026-06-15T17:36:01.0079281Z libsystemd0                245.4-
4ubuntu3.24          (won't fix)       deb     CVE-2020-13776       Low
0.1% (33rd)    < 0.1
      2026-06-15T17:36:01.0081607Z libudev1                   245.4-
4ubuntu3.24          (won't fix)       deb     CVE-2020-13776       Low
0.1% (33rd)    < 0.1
      2026-06-15T17:36:01.0083841Z binutils                   2.34-
6ubuntu1.11                            deb     CVE-2025-1180        Medium
< 0.1% (24th)  < 0.1
      2026-06-15T17:36:01.0086091Z binutils-common            2.34-
6ubuntu1.11                            deb     CVE-2025-1180        Medium
< 0.1% (24th)  < 0.1
      2026-06-15T17:36:01.0088974Z binutils-x86-64-linux-gnu  2.34-
6ubuntu1.11                            deb     CVE-2025-1180        Medium
< 0.1% (24th)  < 0.1
      2026-06-15T17:36:01.0091322Z libbinutils                2.34-
6ubuntu1.11                            deb     CVE-2025-1180        Medium
< 0.1% (24th)  < 0.1
      2026-06-15T17:36:01.0093580Z libctf-nobfd0              2.34-
6ubuntu1.11                            deb     CVE-2025-1180        Medium
< 0.1% (24th)  < 0.1
      2026-06-15T17:36:01.0095798Z libctf0                    2.34-
6ubuntu1.11                            deb     CVE-2025-1180        Medium
< 0.1% (24th)  < 0.1
      2026-06-15T17:36:01.0098255Z urllib3                    1.25.11
2.5.0             python  GHSA-pq67-6m6q-mj2v  Medium      < 0.1% (23rd)  <
0.1
      2026-06-15T17:36:01.0100430Z binutils                   2.34-
6ubuntu1.11                            deb     CVE-2025-5245        Medium
< 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0102682Z binutils-common            2.34-
6ubuntu1.11                            deb     CVE-2025-5245        Medium
< 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0105052Z binutils-x86-64-linux-gnu  2.34-
6ubuntu1.11                            deb     CVE-2025-5245        Medium
< 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0107658Z libbinutils                2.34-
6ubuntu1.11                            deb     CVE-2025-5245        Medium
< 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0110477Z libctf-nobfd0              2.34-
6ubuntu1.11                            deb     CVE-2025-5245        Medium
< 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0112853Z libctf0                    2.34-
6ubuntu1.11                            deb     CVE-2025-5245        Medium
< 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0115028Z requests                   2.23.0
2.32.0            python  GHSA-9wx4-h78v-vm56  Medium      < 0.1% (22nd)  <
0.1
      2026-06-15T17:36:01.0117564Z libc-bin                   2.31-
0ubuntu9.17           (won't fix)       deb     CVE-2021-33574       Low
0.1% (32nd)    < 0.1
      2026-06-15T17:36:01.0119879Z libc-dev-bin               2.31-
0ubuntu9.18           (won't fix)       deb     CVE-2021-33574       Low
0.1% (32nd)    < 0.1
      2026-06-15T17:36:01.0122147Z libc6                      2.31-
0ubuntu9.18           (won't fix)       deb     CVE-2021-33574       Low
0.1% (32nd)    < 0.1
```

```
      2026-06-15T17:36:01.0124466Z libc6-dev                   2.31-
0ubuntu9.18          (won't fix)       deb     CVE-2021-33574       Low
0.1% (32nd)    < 0.1
      2026-06-15T17:36:01.0126947Z binutils                    2.34-
6ubuntu1.11                            deb     CVE-2025-5244        Medium
< 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0129211Z binutils-common             2.34-
6ubuntu1.11                            deb     CVE-2025-5244        Medium
< 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0131590Z binutils-x86-64-linux-gnu   2.34-
6ubuntu1.11                            deb     CVE-2025-5244        Medium
< 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0133984Z libbinutils                 2.34-
6ubuntu1.11                            deb     CVE-2025-5244        Medium
< 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0137068Z libctf-nobfd0               2.34-
6ubuntu1.11                            deb     CVE-2025-5244        Medium
< 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0139893Z libctf0                     2.34-
6ubuntu1.11                            deb     CVE-2025-5244        Medium
< 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0142195Z python-pip-whl              20.0.2-
5ubuntu1.11          (won't fix)       deb     CVE-2024-35195       Medium
< 0.1% (22nd)  < 0.1
      2026-06-15T17:36:01.0144599Z python3-pip                 20.0.2-
5ubuntu1.11          (won't fix)       deb     CVE-2024-35195       Medium
< 0.1% (22nd)  < 0.1
      2026-06-15T17:36:01.0147139Z binutils                    2.34-
6ubuntu1.11                            deb     CVE-2025-1148        Medium
< 0.1% (22nd)  < 0.1
      2026-06-15T17:36:01.0149405Z binutils-common             2.34-
6ubuntu1.11                            deb     CVE-2025-1148        Medium
< 0.1% (22nd)  < 0.1
      2026-06-15T17:36:01.0151782Z binutils-x86-64-linux-gnu   2.34-
6ubuntu1.11                            deb     CVE-2025-1148        Medium
< 0.1% (22nd)  < 0.1
      2026-06-15T17:36:01.0154146Z libbinutils                 2.34-
6ubuntu1.11                            deb     CVE-2025-1148        Medium
< 0.1% (22nd)  < 0.1
      2026-06-15T17:36:01.0156456Z libctf-nobfd0               2.34-
6ubuntu1.11                            deb     CVE-2025-1148        Medium
< 0.1% (22nd)  < 0.1
      2026-06-15T17:36:01.0159034Z libctf0                     2.34-
6ubuntu1.11                            deb     CVE-2025-1148        Medium
< 0.1% (22nd)  < 0.1
      2026-06-15T17:36:01.0161388Z binutils                    2.34-
6ubuntu1.11          (won't fix)       deb     CVE-2021-20197       Low
0.1% (30th)    < 0.1
      2026-06-15T17:36:01.0163841Z binutils-common             2.34-
6ubuntu1.11          (won't fix)       deb     CVE-2021-20197       Low
0.1% (30th)    < 0.1
      2026-06-15T17:36:01.0166364Z binutils-x86-64-linux-gnu   2.34-
6ubuntu1.11          (won't fix)       deb     CVE-2021-20197       Low
0.1% (30th)    < 0.1
      2026-06-15T17:36:01.0169332Z libbinutils                 2.34-
6ubuntu1.11          (won't fix)       deb     CVE-2021-20197       Low
0.1% (30th)    < 0.1
      2026-06-15T17:36:01.0172472Z libctf-nobfd0               2.34-
6ubuntu1.11          (won't fix)       deb     CVE-2021-20197       Low
0.1% (30th)    < 0.1
```

```
      2026-06-15T17:36:01.0174832Z libctf0                    2.34-
6ubuntu1.11          (won't fix)       deb     CVE-2021-20197       Low
0.1% (30th)    < 0.1
      2026-06-15T17:36:01.0177353Z urllib3                    1.25.11
1.26.18           python  GHSA-g4mx-q9vg-27p4  Medium      < 0.1% (17th)  <
0.1
      2026-06-15T17:36:01.0179571Z binutils                   2.34-
6ubuntu1.11                           deb     CVE-2025-1149        Medium
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0182012Z binutils-common            2.34-
6ubuntu1.11                           deb     CVE-2025-1149        Medium
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0184449Z binutils-x86-64-linux-gnu  2.34-
6ubuntu1.11                           deb     CVE-2025-1149        Medium
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0187080Z libbinutils                2.34-
6ubuntu1.11                           deb     CVE-2025-1149        Medium
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0189352Z libctf-nobfd0              2.34-
6ubuntu1.11                           deb     CVE-2025-1149        Medium
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0191550Z libctf0                    2.34-
6ubuntu1.11                           deb     CVE-2025-1149        Medium
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0193722Z werkzeug                   0.16.1
3.1.4             python  GHSA-hgf8-39gv-g3f2  Medium      < 0.1% (13th)  <
0.1
      2026-06-15T17:36:01.0195888Z coreutils                  8.30-3ubuntu2
deb     CVE-2016-2781        Low         < 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0198446Z libc-bin                   2.31-
0ubuntu9.17          2.31-0ubuntu9.18  deb     CVE-2025-4802        Medium
< 0.1% (13th)  < 0.1
      2026-06-15T17:36:01.0205949Z libpam-modules             1.3.1-
5ubuntu4.7                            deb     CVE-2024-10041       Medium
< 0.1% (13th)  < 0.1
      2026-06-15T17:36:01.0208883Z libpam-modules-bin         1.3.1-
5ubuntu4.7                            deb     CVE-2024-10041       Medium
< 0.1% (13th)  < 0.1
      2026-06-15T17:36:01.0211255Z libpam-runtime             1.3.1-
5ubuntu4.7                            deb     CVE-2024-10041       Medium
< 0.1% (13th)  < 0.1
      2026-06-15T17:36:01.0213486Z libpam0g                   1.3.1-
5ubuntu4.7                            deb     CVE-2024-10041       Medium
< 0.1% (13th)  < 0.1
      2026-06-15T17:36:01.0215663Z urllib3                    1.25.11
2.6.0             python  GHSA-gm62-xv2j-4w53  High        < 0.1% (7th)   <
0.1
      2026-06-15T17:36:01.0218180Z login                      1:4.8.1-
1ubuntu5.20.04.5                    deb     CVE-2013-4235        Low
< 0.1% (20th)  < 0.1
      2026-06-15T17:36:01.0220382Z passwd                     1:4.8.1-
1ubuntu5.20.04.5                    deb     CVE-2013-4235        Low
< 0.1% (20th)  < 0.1
      2026-06-15T17:36:01.0222703Z libpython3.8               3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2020-10735
Negligible  0.4% (60th)    < 0.1
      2026-06-15T17:36:01.0225180Z libpython3.8-dev           3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2020-10735
Negligible  0.4% (60th)    < 0.1
```

```
      2026-06-15T17:36:01.0227953Z libpython3.8-minimal       3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2020-10735
Negligible  0.4% (60th)    < 0.1
      2026-06-15T17:36:01.0230490Z libpython3.8-stdlib        3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2020-10735
Negligible  0.4% (60th)    < 0.1
      2026-06-15T17:36:01.0233067Z python3.8                  3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2020-10735
Negligible  0.4% (60th)    < 0.1
      2026-06-15T17:36:01.0235521Z python3.8-dev              3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2020-10735
Negligible  0.4% (60th)    < 0.1
      2026-06-15T17:36:01.0239443Z python3.8-minimal          3.8.10-
0ubuntu1~20.04.18  (won't fix)       deb     CVE-2020-10735
Negligible  0.4% (60th)    < 0.1
      2026-06-15T17:36:01.0241919Z libperl5.30                5.30.0-
9ubuntu0.5                           deb     CVE-2025-40909       Medium
< 0.1% (9th)   < 0.1
      2026-06-15T17:36:01.0244125Z perl                       5.30.0-
9ubuntu0.5                           deb     CVE-2025-40909       Medium
< 0.1% (9th)   < 0.1
      2026-06-15T17:36:01.0246307Z perl-base                  5.30.0-
9ubuntu0.5                           deb     CVE-2025-40909       Medium
< 0.1% (9th)   < 0.1
      2026-06-15T17:36:01.0248897Z perl-modules-5.30          5.30.0-
9ubuntu0.5                           deb     CVE-2025-40909       Medium
< 0.1% (9th)   < 0.1
      2026-06-15T17:36:01.0251166Z libc-bin                   2.31-
0ubuntu9.17                            deb     CVE-2016-20013
Negligible  0.3% (54th)    < 0.1
      2026-06-15T17:36:01.0253426Z libc-dev-bin               2.31-
0ubuntu9.18                            deb     CVE-2016-20013
Negligible  0.3% (54th)    < 0.1
      2026-06-15T17:36:01.0255624Z libc6                      2.31-
0ubuntu9.18                            deb     CVE-2016-20013
Negligible  0.3% (54th)    < 0.1
      2026-06-15T17:36:01.0258288Z libc6-dev                  2.31-
0ubuntu9.18                            deb     CVE-2016-20013
Negligible  0.3% (54th)    < 0.1
      2026-06-15T17:36:01.0260532Z libncurses6                6.2-
0ubuntu2.1                             deb     CVE-2023-50495       Low
< 0.1% (16th)  < 0.1
      2026-06-15T17:36:01.0262784Z libncursesw6               6.2-
0ubuntu2.1                             deb     CVE-2023-50495       Low
< 0.1% (16th)  < 0.1
      2026-06-15T17:36:01.0265210Z libtinfo6                  6.2-
0ubuntu2.1                             deb     CVE-2023-50495       Low
< 0.1% (16th)  < 0.1
      2026-06-15T17:36:01.0268304Z ncurses-base               6.2-
0ubuntu2.1                             deb     CVE-2023-50495       Low
< 0.1% (16th)  < 0.1
      2026-06-15T17:36:01.0270540Z ncurses-bin                6.2-
0ubuntu2.1                             deb     CVE-2023-50495       Low
< 0.1% (16th)  < 0.1
      2026-06-15T17:36:01.0272785Z werkzeug                   0.16.1
3.1.6             python  GHSA-29vq-49wr-vm6x  Medium      < 0.1% (8th)   <
0.1
      2026-06-15T17:36:01.0274972Z binutils                   2.34-
6ubuntu1.11                            deb     CVE-2025-1151        Low
< 0.1% (16th)  < 0.1
```

```
      2026-06-15T17:36:01.0277467Z binutils-common            2.34-
6ubuntu1.11                              deb     CVE-2025-1151        Low
< 0.1% (16th)  < 0.1
      2026-06-15T17:36:01.0279889Z binutils-x86-64-linux-gnu  2.34-
6ubuntu1.11                              deb     CVE-2025-1151        Low
< 0.1% (16th)  < 0.1
      2026-06-15T17:36:01.0282235Z libbinutils                2.34-
6ubuntu1.11                              deb     CVE-2025-1151        Low
< 0.1% (16th)  < 0.1
      2026-06-15T17:36:01.0284483Z libctf-nobfd0              2.34-
6ubuntu1.11                              deb     CVE-2025-1151        Low
< 0.1% (16th)  < 0.1
      2026-06-15T17:36:01.0287090Z libctf0                    2.34-
6ubuntu1.11                              deb     CVE-2025-1151        Low
< 0.1% (16th)  < 0.1
      2026-06-15T17:36:01.0289320Z libpcre2-8-0               10.34-
7ubuntu0.1                             deb     CVE-2022-41409       Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0291514Z binutils                   2.34-
6ubuntu1.11                              deb     CVE-2025-1150        Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0293670Z binutils                   2.34-
6ubuntu1.11                              deb     CVE-2025-1152        Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0295949Z binutils-common            2.34-
6ubuntu1.11                              deb     CVE-2025-1150        Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0325125Z binutils-common            2.34-
6ubuntu1.11                              deb     CVE-2025-1152        Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0328162Z binutils-x86-64-linux-gnu  2.34-
6ubuntu1.11                              deb     CVE-2025-1150        Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0330638Z binutils-x86-64-linux-gnu  2.34-
6ubuntu1.11                              deb     CVE-2025-1152        Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0332980Z libbinutils                2.34-
6ubuntu1.11                              deb     CVE-2025-1150        Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0335203Z libbinutils                2.34-
6ubuntu1.11                              deb     CVE-2025-1152        Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0337932Z libctf-nobfd0              2.34-
6ubuntu1.11                              deb     CVE-2025-1150        Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0340289Z libctf-nobfd0              2.34-
6ubuntu1.11                              deb     CVE-2025-1152        Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0342481Z libctf0                    2.34-
6ubuntu1.11                              deb     CVE-2025-1150        Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0344611Z libctf0                    2.34-
6ubuntu1.11                              deb     CVE-2025-1152        Low
< 0.1% (15th)  < 0.1
      2026-06-15T17:36:01.0347051Z binutils                   2.34-
6ubuntu1.11                              deb     CVE-2025-3198        Medium
< 0.1% (8th)   < 0.1
      2026-06-15T17:36:01.0349320Z binutils-common            2.34-
6ubuntu1.11                              deb     CVE-2025-3198        Medium
< 0.1% (8th)   < 0.1
```

```
      2026-06-15T17:36:01.0351691Z binutils-x86-64-linux-gnu  2.34-
6ubuntu1.11                             deb     CVE-2025-3198        Medium
< 0.1% (8th)   < 0.1
      2026-06-15T17:36:01.0354616Z libbinutils                2.34-
6ubuntu1.11                             deb     CVE-2025-3198        Medium
< 0.1% (8th)   < 0.1
      2026-06-15T17:36:01.0357405Z libctf-nobfd0              2.34-
6ubuntu1.11                             deb     CVE-2025-3198        Medium
< 0.1% (8th)   < 0.1
      2026-06-15T17:36:01.0359650Z libctf0                    2.34-
6ubuntu1.11                             deb     CVE-2025-3198        Medium
< 0.1% (8th)   < 0.1
      2026-06-15T17:36:01.0361846Z urllib3                    1.25.11
2.6.0             python  GHSA-2xpw-w6gg-jr37  High        < 0.1% (4th)   <
0.1
      2026-06-15T17:36:01.0364047Z werkzeug                   0.16.1
3.1.5             python  GHSA-87hc-h4r5-73f7  Medium      < 0.1% (6th)   <
0.1
      2026-06-15T17:36:01.0366236Z login                      1:4.8.1-
1ubuntu5.20.04.5                     deb     CVE-2023-29383       Low
< 0.1% (12th)  < 0.1
      2026-06-15T17:36:01.0368756Z passwd                     1:4.8.1-
1ubuntu5.20.04.5                     deb     CVE-2023-29383       Low
< 0.1% (12th)  < 0.1
      2026-06-15T17:36:01.0370957Z urllib3                    1.25.11
2.6.3             python  GHSA-38jv-5279-wg99  High        < 0.1% (2nd)   <
0.1
      2026-06-15T17:36:01.0373120Z idna                       2.10
3.15              python  GHSA-65pc-fj4g-8rjx  Medium      < 0.1% (4th)   <
0.1
      2026-06-15T17:36:01.0375265Z urllib3                    1.25.11
2.7.0             python  GHSA-qccp-gfcp-xxvc  High        < 0.1% (2nd)   <
0.1
      2026-06-15T17:36:01.0377953Z cryptography               2.8
46.0.5            python  GHSA-r6ph-v2qm-q3c2  High        < 0.1% (1st)   <
0.1
      2026-06-15T17:36:01.0380116Z patch                      2.7.6-6
deb     CVE-2021-45261       Negligible  0.1% (35th)    < 0.1
      2026-06-15T17:36:01.0382221Z dirmngr                    2.2.19-
3ubuntu2.5                              deb     CVE-2022-3219        Low
< 0.1% (6th)   < 0.1
      2026-06-15T17:36:01.0384940Z gnupg                      2.2.19-
3ubuntu2.5                              deb     CVE-2022-3219        Low
< 0.1% (6th)   < 0.1
      2026-06-15T17:36:01.0387632Z gnupg-l10n                 2.2.19-
3ubuntu2.5                              deb     CVE-2022-3219        Low
< 0.1% (6th)   < 0.1
      2026-06-15T17:36:01.0389881Z gnupg-utils                2.2.19-
3ubuntu2.5                              deb     CVE-2022-3219        Low
< 0.1% (6th)   < 0.1
      2026-06-15T17:36:01.0392062Z gpg                        2.2.19-
3ubuntu2.5                              deb     CVE-2022-3219        Low
< 0.1% (6th)   < 0.1
      2026-06-15T17:36:01.0394182Z gpg-agent                  2.2.19-
3ubuntu2.5                              deb     CVE-2022-3219        Low
< 0.1% (6th)   < 0.1
      2026-06-15T17:36:01.0396401Z gpg-wks-client             2.2.19-
3ubuntu2.5                              deb     CVE-2022-3219        Low
< 0.1% (6th)   < 0.1
```

```
      2026-06-15T17:36:01.0398976Z gpg-wks-server             2.2.19-
3ubuntu2.5                          deb     CVE-2022-3219        Low
< 0.1% (6th)   < 0.1
      2026-06-15T17:36:01.0401194Z gpgconf                    2.2.19-
3ubuntu2.5                          deb     CVE-2022-3219        Low
< 0.1% (6th)   < 0.1
      2026-06-15T17:36:01.0403325Z gpgsm                      2.2.19-
3ubuntu2.5                          deb     CVE-2022-3219        Low
< 0.1% (6th)   < 0.1
      2026-06-15T17:36:01.0405424Z gpgv                       2.2.19-
3ubuntu2.5                          deb     CVE-2022-3219        Low
< 0.1% (6th)   < 0.1
      2026-06-15T17:36:01.0408032Z python-pip-whl             20.0.2-
5ubuntu1.11        (won't fix)      deb     CVE-2023-37920
Negligible  0.1% (30th)    < 0.1
      2026-06-15T17:36:01.0410451Z python3-pip                20.0.2-
5ubuntu1.11        (won't fix)      deb     CVE-2023-37920
Negligible  0.1% (30th)    < 0.1
      2026-06-15T17:36:01.0413376Z patch                      2.7.6-6
deb     CVE-2019-20633       Negligible  0.1% (29th)    < 0.1
      2026-06-15T17:36:01.0415520Z libpcre3                   2:8.39-
12ubuntu0.1                         deb     CVE-2017-11164
Negligible  0.1% (29th)    < 0.1
      2026-06-15T17:36:01.0418112Z libperl5.30                5.30.0-
9ubuntu0.5         (won't fix)      deb     CVE-2023-47039
Negligible  < 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0420409Z perl                       5.30.0-
9ubuntu0.5         (won't fix)      deb     CVE-2023-47039
Negligible  < 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0422679Z perl-base                  5.30.0-
9ubuntu0.5         (won't fix)      deb     CVE-2023-47039
Negligible  < 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0425102Z perl-modules-5.30          5.30.0-
9ubuntu0.5         (won't fix)      deb     CVE-2023-47039
Negligible  < 0.1% (23rd)  < 0.1
      2026-06-15T17:36:01.0427656Z flask                      1.1.2
3.1.3             python  GHSA-68rp-wp8r-4726  Low         < 0.1% (2nd)   <
0.1
      2026-06-15T17:36:01.0429879Z cryptography               2.8
46.0.6            python  GHSA-m959-cc7f-wv43  Low         < 0.1% (1st)   <
0.1
      2026-06-15T17:36:01.0432157Z requests                   2.23.0
2.33.0            python  GHSA-gc5v-m9x4-r6x2  Medium      < 0.1% (0th)   <
0.1
      2026-06-15T17:36:01.0434357Z binutils                   2.34-
6ubuntu1.11                         deb     CVE-2022-48064
Negligible  < 0.1% (0th)   < 0.1
      2026-06-15T17:36:01.0436850Z binutils-common            2.34-
6ubuntu1.11                         deb     CVE-2022-48064
Negligible  < 0.1% (0th)   < 0.1
      2026-06-15T17:36:01.0439293Z binutils-x86-64-linux-gnu  2.34-
6ubuntu1.11                         deb     CVE-2022-48064
Negligible  < 0.1% (0th)   < 0.1
      2026-06-15T17:36:01.0441829Z libbinutils                2.34-
6ubuntu1.11                         deb     CVE-2022-48064
Negligible  < 0.1% (0th)   < 0.1
      2026-06-15T17:36:01.0444680Z libctf-nobfd0              2.34-
6ubuntu1.11                         deb     CVE-2022-48064
Negligible  < 0.1% (0th)   < 0.1
```

```
      2026-06-15T17:36:01.0447208Z libctf0                    2.34-
6ubuntu1.11                              deb     CVE-2022-48064
Negligible  < 0.1% (0th)   < 0.1
      2026-06-15T17:36:01.0449477Z cryptography               2.8
41.0.0            python  GHSA-5cpq-8wj7-hf2v  Low         N/A
N/A
      2026-06-15T17:36:01.0451747Z cryptography               2.8
41.0.3            python  GHSA-jm77-qphf-c4w8  Low         N/A
N/A
      2026-06-15T17:36:01.0454008Z cryptography               2.8
41.0.4            python  GHSA-v8gr-m533-ghj9  Low         N/A
N/A
      2026-06-15T17:36:01.0456237Z libncurses6                6.2-
0ubuntu2.1                               deb     CVE-2023-45918       Low
N/A            N/A
      2026-06-15T17:36:01.0458752Z libncursesw6               6.2-
0ubuntu2.1                               deb     CVE-2023-45918       Low
N/A            N/A
      2026-06-15T17:36:01.0460982Z libtinfo6                  6.2-
0ubuntu2.1                               deb     CVE-2023-45918       Low
N/A            N/A
      2026-06-15T17:36:01.0463171Z ncurses-base               6.2-
0ubuntu2.1                               deb     CVE-2023-45918       Low
N/A            N/A
      2026-06-15T17:36:01.0465359Z ncurses-bin                6.2-
0ubuntu2.1                               deb     CVE-2023-45918       Low
N/A            N/A
```

- Identificación de librerías críticas: Se identificaron de forma precisa dependencias comprometidas, mostrando vulnerabilidades de criticidad alta (High) en varios componentes.
- Análisis de remediación: El reporte de Grype no solo expuso las fallas sino provee una trazabilidad para la solución de estas sugiriendo versiones más estables de mitigación.
- Paquetes del sistema: También se presentaron fallos en las librerías del sistema .deb con el estado de won’t fix por cuanto el sistema operativo de la prueba ya no tiene soporte y ha superado su vida útil.

Tabla 1. Matriz comparativa de hallazgos.

Dimensión de la Evaluación	Métrica Cuantitativa	Herramienta: Trivy	Herramienta: Grype
Volumetría de Vulnerabilidades (Clasificación según CVSS v3.1)	Total, de CVEs Detectados	0	264
	Severidad: Crítica (Critical)	0	0
	Severidad: Alta (High)	0	13
	Severidad: Media (Medium)	0	112
	Severidad: Baja (Low)	0	139
Eficiencia Temporal	Tiempo de ejecución del escaneo	20 segundos	191 segundos (3 min 12 s)
Capa de Cobertura Principal	Foco de detección observado	Imagen Docker / Vulnerabilidades del contenedor	Dependencias de aplicación y paquetes del sistema (SCA)
Resultado del Control	Acción del Pipeline	COMPLETADO (Informativo)	COMPLETADO (Informativo)

El comportamiento del pipeline responde a una arquitectura diseñada deliberadamente con la finalidad de no interrumpir el flujo con finalidad académica.

El flujo realiza un análisis multicapa a través de Aqua Trivy el cual está configurado de manera restrictiva y se encarga del proceso estructural realizando el análisis y evaluación del sistema operativo base y las malas configuraciones críticas que pueda encontrar como, por ejemplo: ejecución a nivel de privilegio root o vulnerabilidades de kernel. En cuanto Grype se utilizó con el propósito de mapear el inventario de dependencias de la capa de aplicación (requirements.txt), al mantener separados los criterios de bloqueos nos ayuda a

prevenir que fallos en las librerías secundarias detengan la fase de compilación de la infraestructura.

Al unificar los criterios de bloqueo de ambas herramientas se puede correr el riesgo de saturar el flujo de trabajo, ya que por ejemplo las 13 vulnerabilidades de severidad alta encontradas por Grype en las librerías Python requieren una validación para determinar si realmente son explotables en el entorno de microservicios locales puestos en práctica. Si realizamos el bloqueo del pipeline de forma masiva si el filtro previo de explotabilidad estamos contradiciendo los principios de agilidad y eficiencia en la metodología DevSecOps.

Por lo tanto, el diseño presentado muestra un modelo balanceado el cual tiene un control mandatorio en la capa de estructura (Trivy) y un control informativo y de validación en la capa del software (Grype), con esto podemos realizar una correcta toma de decisiones y aplicaciones de parches mediante un ciclo de remediación planificado.

CAPÍTULO 5

5. Conclusiones y recomendaciones

5.1. Conclusiones

- Se diseñó e implementó con éxito un pipeline DevSecOps híbrido automatizado dentro del ciclo de integración continua (CI) en GitHub Actions, logrando de esta forma generar seguridad en las fases tempranas del desarrollo (shift left) mediante el uso de contenedores y la orquestación en un servidor local Self-Hosted Runner.
- La evaluación de Trivy y Grype demostró que, aunque ambas herramientas se enfocan en la gestión de vulnerabilidades estas operan bajo enfoques complementarios, Trivy en el análisis estructural de la imagen Docker, mientras que Grype muestra más granularidad en el Análisis de Composición de Software y se enfoca en las librerías de aplicación.
- Con base en la comparación cuantitativa se determinó que existe diferencias: Trivy reportó 0 vulnerabilidades con filtro de severidad y Grype identificó 13 vulnerabilidades altas (High), 112 vulnerabilidades medias (Medium) y 139 vulnerabilidades bajas (Low).
- Se verificó empíricamente el riesgo que representa el uso de imágenes base obsoletas y sin soporte, ya que la utilización de Ubuntu 20.04 en el Dockerfile generó advertencias explícitas de fin de ciclo de vida (End of Life), provocando falsos negativos en la herramienta Trivy, esto debido a la falta de parches activos.
- La arquitectura propuesta demostró ser robusta al realizar una integración de controles basados en el estándar CVSS v3.1, y permitiendo la inyección de directivas como --

exit-code 1 con esto se genera el bloqueo y toma de decisiones ante el descubrimiento de hallazgos críticos.

- El entorno híbrido implementado en la arquitectura con un runner local garantizó que el procesamiento de metadatos y código fuente se limitará solo a la infraestructura de la organización on-premises), asegurando el principio de minimización de datos y el cumplimiento de la trazabilidad exigida por la Ley Orgánica de Protección de Datos Personales (LOPDP).

5.2. Recomendaciones

- Se recomienda modificar el uso de sistemas operativos bases obsoletos o sin soporte (como Ubuntu 20.04) por distribuciones modernas y con soporte de largo plazo (LTS) o imágenes minimalistas tipo Distroless o Alpine, de esta forma se reduce la superficie de ataque y se elimina las alertas de sistemas operativos obsoletos que generan falsos positivos en la herramienta Trivy.
- Se recomienda en una fase productiva mantener los integrados los dos motores de escaneo de seguridades ya que se ha demostrado que estos son complementarios.

6. Referencias bibliográficas

- Abigail. (2025, diciembre 1). *DevSecOps Pipeline Guide: Best practices for secure CI/CD in 2025*.

  https://blog.abigail.co.il/devsecops-pipeline-guide-best-practices-for-secure-ci-cd-in-2025

- Anchore. (2026). *Grype Documentation*.

  https://oss.anchore.com/docs/

- Aqua Security. (2026).

  Trivy: Open Source Vulnerability Scanner.

  https://trivy.dev/

- Aqua Security. (2026).

  Trivy – Vulnerability and Misconfiguration Scanner.

  https://www.aquasec.com/products/trivy/

- Berton, L. (2026). *Trivy vs Grype 2026: Container Scanner*.

  https://lucaberton.com

- Boehm, B. (1981). *Software engineering economics*. Prentice Hall.

- Chowdary, D. (2026). *SBOM Generation and Scanning with Syft and Grype*.

  https://techbytes.app/posts/sbom-generation-scanning-syft-grype-tutorial/

- CISA. (2024). *Software Bill of Materials (SBOM)*.

  https://www.cisa.gov/sbom

- CRA Evidence. (2026, mayo 9). Plazos y sanciones SBOM bajo el CRA: fechas para 2027.

  https://craevidence.com/es/cumplimiento-cra/sbom/requisitos-cra

- Dashen Tech (2026). *Trivy Complete Guide 2026*. https://dashen-tech.com/en/dev-tools/trivy-security-scanner-guide/
- Dev.to. (2026, abril 28). Performance Test: Grype 0.70 vs Trivy 0.50 Scan Times. https://dev.to/johalputt/performance-test-grype-070-vs-trivy-050-scan-times-15-faster-for-alpine-images-31j9
- DevOpsBoys (2026). *Trivy vs Grype vs Snyk — Container Image Scanning Comparison*. https://devopsboys.com/blog/trivy-vs-grype-vs-snyk-container-scanning-2026
- Executive Office of the President. (2021). *Executive Order 14028: Improving the Nation's Cybersecurity*. https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/
- Explica criterios clave como velocidad, integración y consumo en pipelines CI/CD. [secure-pipelines.com]
- Fluid Attacks. (2026, mayo 18). ¿Qué es SBOM? https://fluidattacks.com/es/cybersecurity-essentials/que-es-sbom
- FIRST.org, Inc. (2019). *Common Vulnerability Scoring System v3.1: Specification Document.* https://www.first.org/cvss/v3-1/specification-document
- GitHub. (2026). *GitHub Actions Documentation*. https://docs.github.com/en/actions
- MITRE. (s.f.). *Common Vulnerabilities and Exposures (CVE)*. https://cve.mitre.org

- National Institute of Standards and Technology (NVD). (2022). *Vulnerability Metrics (CVSS).* https://nvd.nist.gov/vuln-metrics/cvss
- National Institute of Standards and Technology (NIST). (2021). *Secure Software Development Framework (SP 800-218).* https://csrc.nist.gov/publications/detail/sp/800-218/final
- National Institute of Standards and Technology (NIST). (2017). *Application Container Security Guide (SP 800-190)*. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf
- OASIS. (2022). *Common Security Advisory Framework (CSAF) Version 2.0*. https://docs.oasis-open.org/csaf/csaf/v2.0/csaf-v2.0.html
- Open Policy Agent (OPA). (2024). *Open Policy Agent Documentation.* https://www.openpolicyagent.org/docs
- OpsMx. (2025, enero 29). *How to eliminate security bottlenecks in your CI/CD pipeline*. https://www.linkedin.com/pulse/how-eliminate-security-bottlenecks-your-cicd-pipeline-opsmx-hpq6c
- Oulmakhzoune, S. (2026). *CI/CD Security Scanners Compared: Trivy vs Grype vs Snyk vs Checkov*. https://secure-pipelines.com
- OWASP Foundation. (2021). *OWASP Top 10: The Ten Most Critical Web Application Security Risks.* https://owasp.org/Top10/

- Prabhakaran, S. (2026). *End-to-End SBOM & Vulnerability Scanning with Syft and Grype*. https://medium.com/@santhoshprabhakaran03/end-to-end-sbom-vulnerability-scanning-with-syft-and-grype-dc15a5ff40e7
- Souppaya, M., Morello, J., & Scarfone, K. (2017). *Application Container Security Guide (NIST SP 800-190).* National Institute of Standards and Technology (NIST). https://doi.org/10.6028/NIST.SP.800-190
- WeeSec. (2025, septiembre 14). *SBOM en 2026: générer, signer, distribuer (SPDX vs CycloneDX)*. https://www.weesec.com/blog/sbom-2026-spdx-cyclonedx-pratique.html