

Informe Proyecto Final

IMPLEMENTACION DE CONTROLES DE SEGURIDAD EN UN CI/CD (INTEGRACION CONTINUA / DESPLIEGUE CONTINUO) PARA UNA ARQUITECTURA DE MICROSERVICIOS SOBRE KUBERNETES EN AMAZON WEB SERVICES

Angel Andres Vargas, Miguel Angel Lemus, Juan Pablo Barbosa S

Curso MSIN 4301 – Proyecto Final

Universidad de los Andes, Bogotá, Colombia

{ ma.lemus72, aa.vargas, jp.barbosa10 }@uniandes.edu.co

Fecha de presentación: diciembre 10 de 2019

1. Resumen

El problema para abordar en este trabajo de grado consiste en la identificación e implementación de los controles de seguridad que buscan reducir, mitigar o eliminar los riesgos asociados a un proceso de CI/CD (Integración Continua y Despliegue Continuo) para una arquitectura de microservicios sobre contenedores en la nube.

En este contexto se identificaron las tecnologías y herramientas más reconocidas y utilizadas y sobre estas se identificó en la literatura las amenazas más comunes, brindando un marco de trabajo para identificar inicialmente los componentes de seguridad que deben ser desarrollados e implementados y que se muestran en el diagrama de componentes como solución a la problemática identificada.

2. Conceptos básicos

2.1 Imagen

Una imagen de contenedor es un software autónomo que tiene todo lo que necesita para ejecutarse: código, herramientas y recursos . (Aquasec, 2018)

2.2 Contenedor

Un contenedor es una unidad de software estándar que empaqueta el código y todas sus dependencias para que la aplicación se ejecute de manera rápida y confiable de un entorno informático a otro. Una imagen de contenedor Docker es un paquete de software ligero, independiente y ejecutable que incluye todo lo necesario para ejecutar una aplicación: código, tiempo de ejecución, herramientas del sistema, bibliotecas del sistema y configuraciones. (Docker, 2019)

2.3 Orquestador

Es una plataforma portable y extensible de código abierto para administrar cargas de trabajo y servicios, como se mencionaba anteriormente en el documento, la plataforma más popular para controlar el flujo de vida de una aplicación es Kubernetes liberada por Google en el año 2014; facilitando la automatización y configuración declarativa.

2.4 Flujo despliegue de un servicio y/o aplicación

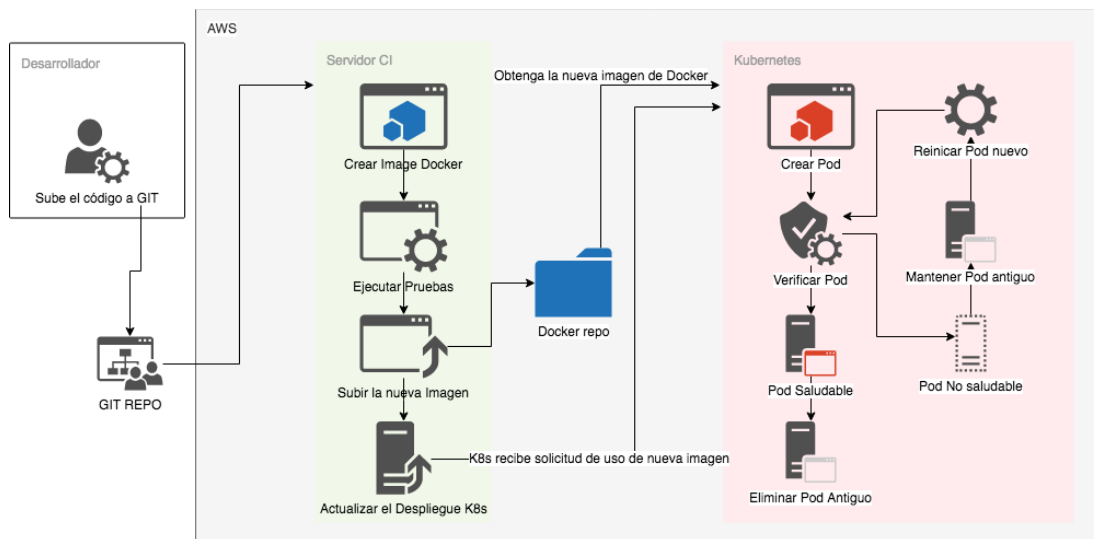


Imagen 1 Diagrama base de flujo de información CI/CD con orquestador de contenedores K8s

La imagen 1 ilustra de forma básica un flujo de información de un ciclo de integración y despliegue continuo:

- El desarrollador confirma un cambio en su código fuente al gestor de control de versiones, en este caso representado por GIT.
- El cambio en el repositorio es detectado por el servicio de integración el cual trae el código fuente incluyendo el cambio.
- Inicia el proceso de construcción del artefacto completo que incluye tomar la imagen base requerida por el artefacto y construir una imagen.
- Se realizan las pruebas establecidas por la organización que pueden incluir pruebas funcionales y de unidad.
- Confirmación de la correcta construcción de todas las imágenes relacionadas a dichos artefactos. En este punto el servicio de integración informa al gestor y orquestador de contenedores (Kubernetes) que se ha realizado un cambio por lo que se requiere un nuevo despliegue de las imágenes.
- Kubernetes prepara la creación del pod y verificación, posteriormente procede a desplegar las imágenes en nuevos contenedores, dando de baja los que esté relacionados y actualmente en producción.

3. Descripción del Problema

Debido a la necesidad de las organizaciones para dar respuesta rápida y oportuna a las exigencias del cliente y del mercado (time to market) a través de la transformación digital de sus servicios y la alta adopción del uso de nuevas tecnologías en la nube, se evidencia un alto crecimiento en el uso de microservicios y contenedores orquestados a través de kubernetes. Este escenario supone nuevas amenazas, vectores de ataque, riesgos sobre las herramientas y procesos de integración, distribución e implementación continua en el ciclo de vida de los servicios ofrecidos por las organizaciones.

En el paper publicado en IEEE Container Security: Issues, Challenges, and the Road Ahead (Sultan, Ahmad, & Dimitriou, 2019), se propone un modelo de amenazas resumido en 4 casos de uso que se indican a continuación:

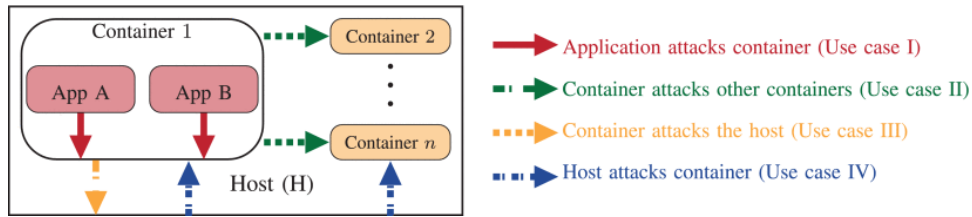


Imagen 2 Resumen de los requerimientos de protección de seguridad en contenedores

De acuerdo a los cuatro escenarios identificados en el paper Container Security, se identifica las posibles amenazas y vulnerabilidades para los contenedores en las diferentes capas de su stack, incluyendo todas las tecnologías que permiten su realización, en estos se incluyen los mencionados en la guía de aseguramiento para contenedores publicada por el NIST (NIST SP -800-190, 2017), que es la guía recomendada para la implementación en la industria.

Componente	Amenaza	Vulnerabilidad	Posible solución
Imagen	Vulnerabilidades de la imagen	Ejecución de código remoto (CVE-2014-6271)	Componente valoración pruebas estaticas de seguridad
	Defectos de configuración de imagen	Acceso no autorizado	Componente valoración de imágenes base
	Malware embebido	Virus, troyanos, gusanos	Componente valoración pruebas dinámicas de seguridad
	Secretos en texto plano embebidos	Divulgación de información confidencial	Componente valoración pruebas dinámicas de seguridad
	Uso de imágenes no confiables	Ejecución de dienfretes ataques debido a Backdoors	Componente valoración de imágenes base
Registro	Imagenes obsoletas en el registro	Ataque a los contenedores	Componente valoración de imágenes base
Contenedor	Vulnerabilidades durante el tiempo de ejecución	Escalación de privilegios (CVE-2017-5123)	Componente valoración pruebas dinámicas de seguridad
	Configuración insegura del tiempo de ejecución	Permitir a un atacante escapar de un contenedor al host y además hacerlo como usuario privilegiado (CVE-2019-5736)	Componete valoración Pod
Orquestador	Conexiones inseguras registros		Validar los estandares y lineamientos de seguridad (NIST 800-190) (NIST 800-144)
	Autenticación Insuficiente		
	Acceso a la red sin limites desde contenedores		
	Acceso administrativo sin límites		
	Acceso no autorizado	Escalación de privilegios	
	Tráfico entre contenedores mal separado		
	Mezcla de niveles de sensibilidad en las cargas de trabajo		
	Confianza del nodo orquestador		

Tabla 1 Amenazas de contenedores, vulnerabilidades y contramedidas por componente (Sultan, Ahmad, & Dimitriou, 2019)

De igual forma se enumeran las posibles soluciones que se implementarían para remediar las vulnerabilidades identificadas, agregando los siguientes componentes a un flujo de despliegue de un servicio y /o aplicación:

COMPONENTE VALORADOR DE IMÁGENES BASE: Es el componente que permitirá revisar el cumplimiento de las imágenes con los estándares de la industria definidos.

COMPONENTE DE PRUEBAS ESTATICAS DE SEGURIDAD: Es el componente que permitirá realizar de manera automatizada las pruebas SAST al código de la aplicación y al código como infraestructura.

COMPONENTE DE PRUEBAS DINAMICAS DE SEGURIDAD: Este componente permite ejecutar las pruebas dinámicas DAST sobre la aplicación desplegada en la infraestructura de contenedores.

COMPONENTES DE VALORACION DE PODS DESPLEGADOS: Este componente realiza la detección de vulnerabilidades en tiempo de ejecución, y garantiza el cumplimiento de la configuración del contenedor en tiempo de ejecución

4. Objetivo General

Integrar controles de seguridad a un CI/CD (integración continua / despliegue continuo) desplegado en modelo IaaS en el proveedor de servicios Amazon Web Services, que permita el diseño, desarrollo y puesta en producción de microservicios con una plataforma de orquestación de contenedores Kubernetes garantizando la confidencialidad, integridad y disponibilidad de la información.

5. Evaluación de riesgo de la implementación del proyecto

Para la construcción de la evaluación de riesgo se toma como base las amenazas y vulnerabilidades indicadas en la sección de justificación del proyecto (Sultan, Ahmad, & Dimitriou, 2019), adicionalmente se consultan los siguientes estándares ISO 27000, NIST:

5.1 Supuesto para el análisis de riesgo

El presente proyecto de integración y despliegue continuo es desarrollado para una entidad financiera que requiere automatizar su proceso de desarrollo de software para las aplicaciones web de la banca digital para personas y empresas.

5.2 Clasificación de la información

Se identifico la siguiente información relacionada con información personal de identificación y financiera e información confidencial para la entidad financiera, que será procesada, almacenada o transmitida por los componentes del proyecto, y que se indican a continuación:

5.2.1 Método de valoración de los activos de información

Se establece la siguiente medida de criticidad para la valoración de los activos de información en Integridad, Confidencialidad y Disponibilidad, (NIST SP 800-60 Vol. 1 Rev. 1, 2008) (Guía para la Gestión y Clasificación de Activos de Información., 2016):

- **Integridad:** Guardar en contra de modificación o destrucción de información inapropiada, asegurando no repudio de información y autenticidad. (NIST FIPS 199, 2004). Se refiere a la completitud, exactitud, autorización y oportunidad de la información.
- **Confidencialidad:** Preservar restricciones autorizadas en acceso y divulgación de información, incluyendo los medios para proteger privacidad personal e información propietaria. (NIST FIPS 199, 2004).
- **Disponibilidad:** Asegurar acceso oportuno y confiable a la información y a su uso. (NIST FIPS 199, 2004)

5.2.2 Clasificación de la información

Según lo indicado en el anterior numeral, se determina la siguiente clasificación para los activos de información del proyecto:

Integridad: Media.

- La corrupción de datos puede generar pérdidas financieras serias, pueden causar una degradación significativa en la capacidad de la misión de la entidad en una o más de sus funciones primarias.

- La falta de disponibilidad por corrupción de datos puede ser solo aceptable por un periodo limitado de tiempo.
- Los datos pueden contener información personal del cliente, incluyendo información financiera.
- La entidad es responsable por daños legales si se toman decisiones con base en información corrupta.

Confidencialidad: Media.

- La información contiene información detallada del cliente: Nombre, usuario, contraseña, datos financieros.
- La información incluye código fuente que son propiedad de la entidad protegidos por derechos de autor.
- La información incluye documentos que describen el diseño de los sistemas de información.
- La divulgación de información puede causar una violación a la privacidad de los clientes de la entidad.
- La divulgación de la información puede causar una litigación en contra de la entidad.

Disponibilidad: Alta.

- El Sistema proporciona servicio a un gran número de clientes.
- El sistema procesa transacciones por alto valor.
- El sistema proporciona un mercado a clientes con alto valor comercial.
- La entidad es sujeta a sanciones antes entes de vigilancia gubernamentales si el sistema no funciona por más de 1 hora.

Integridad	Confidencialidad	Disponibilidad
Media	Media	Alta

Tabla 2 Clasificación de la Información

5.2.3 Análisis de Amenazas

Según los objetivos del análisis de riesgo, se establece lo siguiente para el análisis de amenazas:

- Para la implementación de un CICD en la nube:

Se realiza un análisis de amenazas basado en la metodología STRIDE, como se observa en la tabla a continuación, para aplicar esta metodología se toma como base en el diagrama *“Flujo de trabajo de un CI/CD con Kubernetes con Controles de Seguridad detallado”*.

- Para el despliegue en una infraestructura Kubernetes:

Se utiliza como fuente las amenazas y vulnerabilidades resultante de la aplicación del análisis de amenazas documentado en el paper (Sultan, Ahmad, & Dimitriou, 2019) y que es resumido en la imagen: *“Resumen de los requerimientos de protección de seguridad en contenedores”*, con los casos de uso que se indican a continuación:

- La aplicación ataca los contenedores (Caso de uso I).
- Los contenedores atacan los contenedores (Caso de Uso II).
- Los contenedores atacan los Host (Caso de Uso III).
- El host ataca los contenedores (Caso de uso IV).

6. Solución Planteada

6.1 Arquitectura de componentes

Para la arquitectura de un CI/CD para microservicios sobre Kubernetes, se tomó como referencia el documento de Microsoft construyendo una canalización de un CI/CD para microservicios en Kubernetes (MICROSOFT, 2019), y la publicación de New Stack (THENEWSTACK, 2018). Con base en estas arquitecturas, se adicionaron los siguientes componentes:

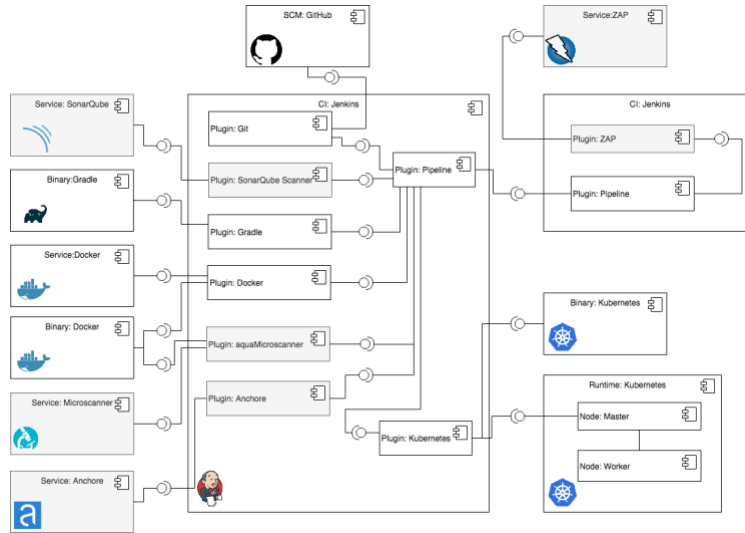


Imagen 3 Componentes implementados CI/CD

6.2 Flujo de CI/CD con controles implementados

Las herramientas ilustradas en el diagrama de componentes se integran en el flujo de CI/CD como se muestra en la siguiente imagen:

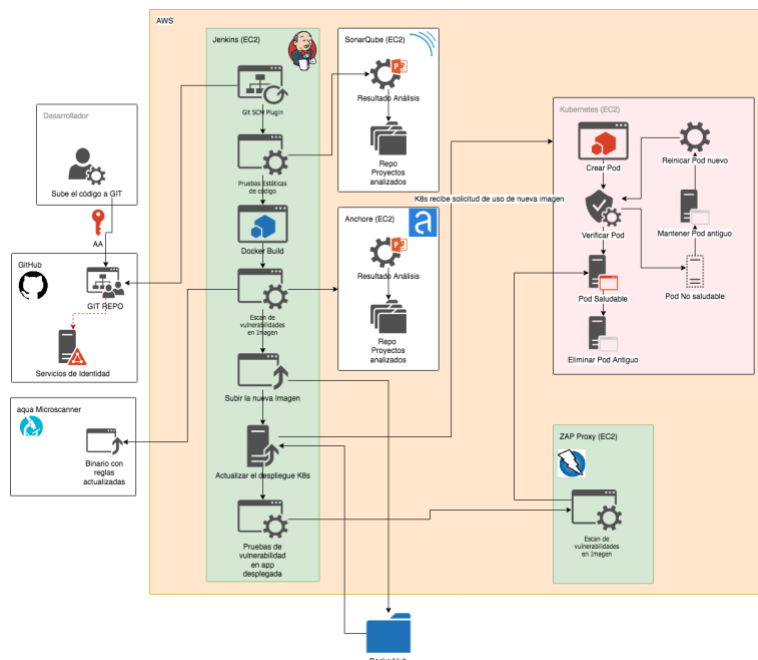


Imagen 4 Flujo Ci/CD con componentes implementados

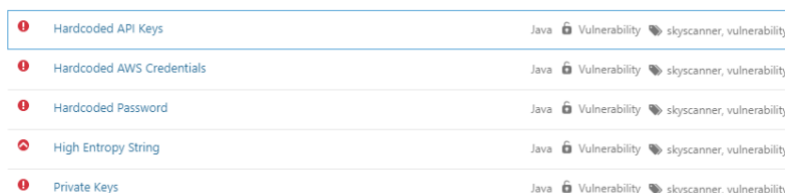
6.3 Implementación de controles

A continuación, se describe la implementación de los controles sobre el CI/CD haciendo uso de las herramientas seleccionadas en este proyecto.

6.3.1 Componentes de pruebas estáticas

A través de la herramienta SonarQube integrada en el CI/CD se implementa este control mitigante de las vulnerabilidades:

- **Vulnerabilidades tipo XSS, SQL Injection, CSRF, malas prácticas de desarrollo:** SonarQube incorpora dentro de sus reglas de calidad más de 600 expresiones para el análisis de vulnerabilidades tipo XSS, SQL Injection, CSRF las cuales incluyen capacidad de identificación de vulnerabilidades del Top 10 de OWASP, SANS Top 25 y la lista de CWE. Adicional a estas se cuenta con alrededor de dos mil expresiones que buscan identificar malas prácticas de desarrollo y problemas en producción. Jenkins es configurado para detener el ciclo de despliegue si reporte de calidad de SonarQube indica estado fallido, de esta forma no es posible continuar con el flujo hasta que las vulnerabilidades identificadas sean ajustadas en el código fuente por parte del desarrollador. La configuración llevada a cabo para ambos componentes es detallada en el manual de instalación y configuración de CI/CD anexo a este documento.
- **Secretos o información confidencial almacenada en texto plano en la aplicación:** Se incorpora a SonarQube el complemento (plugin) Sonar Secrets el cual, a través de las nuevas reglas incorporadas y asociadas al perfil de calidad o “Quality Profile”, permite identificar y marcar como críticas y bloqueantes para los lenguajes de Java y JavaScript:



Hardcoded API Keys	Java	Vulnerability	skyscanner, vulnerability
Hardcoded AWS Credentials	Java	Vulnerability	skyscanner, vulnerability
Hardcoded Password	Java	Vulnerability	skyscanner, vulnerability
High Entropy String	Java	Vulnerability	skyscanner, vulnerability
Private Keys	Java	Vulnerability	skyscanner, vulnerability

Imagen 5 Reglas Skyscanner agregadas en SonarQube

Este complemento refuerza el conjunto de reglas de tipo “Credentials should not be hardcoded” incorporadas en la herramienta y que se aplican a todos los lenguajes de programación soportados por SonarQube. En el evento que una de las reglas anteriores haga “match” SonarQube, finaliza el análisis y genera el reporte indicando un estado bloqueante lo que detendrá los pasos posteriores en Jenkins. El flujo no continuará hasta que el código sea ajustado y actualizado en GitHub. La instalación del complemento sonar secrets y configuración de las reglas en la política se detalla en el manual de configuración de CI/CD anexo a este documento.

Como resultado de la implementación de este control SonarQube, y a manera de ejemplo, al ejecutar el flujo con una aplicación de prueba, en su reporte genera el siguiente estado:

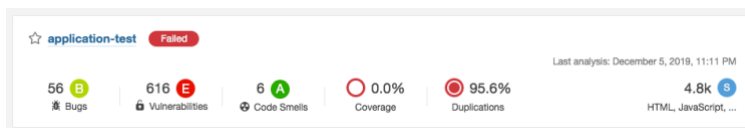


Imagen 6 Resultado de un análisis en SonarQube

El estado “Failed” generado por el “Quality Gate” de SonarQube para esta aplicación de prueba se da por el puntaje “E” obtenido por las pruebas de seguridad las cuales encontraron 616 vulnerabilidades entre las cuales se destacan:



Imagen 7 Hallazgo reportado en SonarQube

La cual identifica una vulnerabilidad asociada a una regla de anti-phishing y anti-XSS cuyo nivel de criticidad es marcado como “Blocker” o bloqueante. De esta forma la ejecución del flujo de CI/CD en Jenkins es detenida en el paso 2 con el error que se muestra a continuación:



Imagen 8 Evento de detención en flujo de Jenkins

Finalmente, el flujo de proceso de Jenkins queda detenido por ERROR:

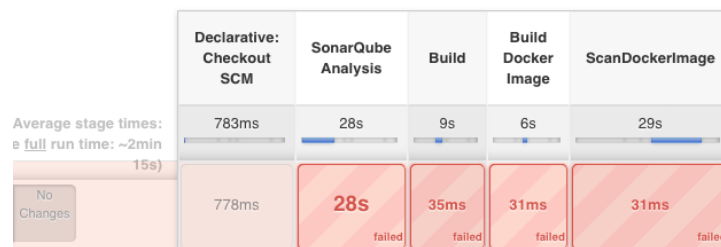


Imagen 9 Resultado fallido de flujo en Jenkins

Este flujo solo construirá la aplicación en el evento que el código sea ajustado siguiendo las recomendaciones de SonarQube y actualizando el repositorio de GitHub con los respectivos cambios.

6.3.2 Componentes de Escáner de Imágenes Contenerizadas

Anchore Engine es una herramienta de código abierto que permite el análisis estático y el cumplimiento basado en políticas que automatiza la inspección, análisis y evaluación de imágenes en contra de la configuración definida por el usuario y que se muestra a continuación. (Anchore, 2019)

Anchore ejecuta las siguientes fases para cada imagen analizada:

Extraer: El contenido de la imagen, pero no lo ejecuta.

Analiza: La imagen corriendo un grupo de analizadores, clasificando y extrayendo meta data.

Guarda: El resultado del análisis en la base de datos para uso futuro.

Evalúa: Políticas contra el resultado, incluyendo vulnerabilidades encontradas en los artefactos.

Actualiza: los datos de fuentes externas usados para evaluación de políticas y descubrimiento de vulnerabilidades y actualiza los resultados de las imágenes en contra de los últimos datos descargados.

- **Análisis de Imágenes:**

Durante el análisis de imágenes cada paquete, librería de software y archivo son inspeccionados y sus datos almacenados en las bases de datos de Anchore. Anchore incluye un número de módulos analizadores que extraen datos de la imagen como (Anchore, 2019):

- Meta data de la imagen
- Capas de la imagen
- Paquetes de Sistema Operativo (RPM, DEB)
- Archivos de datos
- Ruby
- Node.js NPM
- Archivos de Java
- Paquetes Python
- Contenido de Archivos

- **Política:**

Cuando la imagen ha sido analizada y su contenido ha sido descubierto, categorizado y procesado, los resultados son evaluados en contra de un set de checks para dar una recomendación final para pasar o bloquear. Las políticas son como los usuarios describen que chequeos realizar en las imágenes y como los resultados son interpretados. (Anchore, 2019)

Una política es expresada como un conjunto de políticas que es hecha de un grupo de reglas para realizar la evaluación de una imagen contenerizada. Las reglas definen:

- Vulnerabilidades de Seguridad.
- Listas blancas y negras.
- Contenidos de archivos de configuración.
- Presencia de credenciales en la Imagen.
- Cambios de manifiesto en las imágenes.
- Puertos expuestos abiertos.

Una política puede devolver dos resultados: “Passed” o “Failed”

Blacklisted Images: Sobrescribe imágenes específicas para fallar sin importar el resultado de la política.

- Respuesta del flujo de CI/CD con la integración para la valoración de imágenes contenerizadas
 - Ejecución del paso de valoración de imagen con Anchore por Jenkins en el flujo de despliegue continuo.

```
2019-12-09T03:22:25.572 INFO AnchoreWorker Jenkins version: 2.190.3
2019-12-09T03:22:25.572 INFO AnchoreWorker Anchore Container Image Scanner Plugin version: 1.0.21
2019-12-09T03:22:25.572 INFO AnchoreWorker [global] debug: false
2019-12-09T03:22:25.572 INFO AnchoreWorker [build] engineurl: http://127.0.0.1:8228/v1
2019-12-09T03:22:25.572 INFO AnchoreWorker [build] engineuser: admin
2019-12-09T03:22:25.572 INFO AnchoreWorker [build] enginepass: ****
2019-12-09T03:22:25.572 INFO AnchoreWorker [build] engineverify: false
2019-12-09T03:22:25.572 INFO AnchoreWorker [build] name: anchore_images
2019-12-09T03:22:25.572 INFO AnchoreWorker [build] engineRetries: 300
2019-12-09T03:22:25.572 INFO AnchoreWorker [build] policyBundleId: anchore_security_only
2019-12-09T03:22:25.572 INFO AnchoreWorker [build] bailOnFail: true
2019-12-09T03:22:25.572 INFO AnchoreWorker [build] bailOnPluginFail: true
2019-12-09T03:22:25.578 INFO AnchoreWorker Submitting docker.io/juanpab/train-schedule for analysis
2019-12-09T03:22:26.941 INFO AnchoreWorker Analysis request accepted, received image digest
sha256:a4567f1a73c2266aa5b84a77ffc601f37b5f93cdb36465d1d9d6f5deedf54199
2019-12-09T03:22:26.946 INFO AnchoreWorker Waiting for analysis of docker.io/juanpab/train-schedule, polling
status periodically
```

Imagen 10 Ejecución de la valoración de imagen con Anchore en Jenkins

- Respuesta del flujo de Jenkins fallida ante la detección de problemas en la valoración de la imagen vulnerable.

```

2019-12-09T03:49:58.966 INFO AnchoreWorker Completed analysis and processed policy evaluation result
2019-12-09T03:49:58.973 INFO AnchoreWorker Policy evaluation summary for docker.io/juanpab/train-
schedule:latest - stop: 1 (+0
2019-12-09T03:49:58.976 INFO AnchoreWorker Anchore Container Image Scanner Plugin step result - FAIL
2019-12-09T03:49:58.977 INFO AnchoreWorker Querying vulnerability listing for docker.io/juanpab/train-schedule
Archiving artifacts
2019-12-09T03:50:07.136 WARN AnchorePlugin Failing Anchore Container Image Scanner Plugin step due to final
result FAIL
2019-12-09T03:50:07.137 INFO AnchorePlugin Completed Anchore Container Image Scanner step

```

Imagen 11 Resultado fallido del análisis de la imagen vulnerable, en Jenkins salida consola.



Imagen 12 Respuesta fallida del CI/CD en el paso de análisis de imagen contenerizada.

- Resultado del reporte generado por la ejecución de la política secure-only por anchore

Anchore Policy Evaluation Summary

Show 10 entries Search:

Repo Tag	Stop Actions	Warn Actions	Go Actions	Final Action
docker.io/juanpab/train-schedule:latest	1	0	0	STOP

Showing 1 to 1 of 1 entries Previous 1 Next

Anchore Policy Evaluation Report

Show 10 entries Search:

Image Id	Repo Tag	Trigger Id	Gate	Trigger	Check Output	Gate Action	Whitelisted	Policy Id
c0d22679140ee4f80b901e6de697b02fb9ce54944fd36c2d31c3045cd413a41f	docker.io/juanpab/train-schedule:latest	68e630ce4fa8533b139875aa5fc54da5	dockerfile	effective_user	User root found as effective user, which is explicitly not allowed list	STOP	false	48e6f7d6-1765-11e8-b5f9-8b6f228548b6

Imagen 13 Sección de resultado de políticas generado por Anchore en el flujo del CI/CD.

En este reporte se puede observar que una de las políticas configuradas fue activada al detectar un usuario root en los archivos de configuración en el gate dockerfile.

6.3.3 Kubernetes como orquestador de contenedores

La intención de este trabajo es además incluir como componente de orquestación de contenedores a Kubernetes (K8s) dada su popularidad y robustez, y sobre este servicio implementar el modelo de despliegue Canary o canario el cual permite la ejecución controlada de la aplicación construida en pasos anteriores, habilitándola para el escán de vulnerabilidades y verificación de salud. A continuación, se da una explicación breve del despliegue de Kubernetes realizado en AWS como parte de este proyecto.

Amazon Web Services en colaboración con VMWare ofrecen una plantilla de despliegue de Kubernetes en su servicio EC2 la cual puede ser usada en un entorno productivo. Esta plantilla se encuentra disponible en el servicio de CloudFormation de AWS. (VMware Amazon Web Services, 2019)

6.3.4 Componentes pruebas dinámicas

En el desarrollo de aplicaciones Web cada vez que se crea una nueva versión o actualización, es necesario el apoyo por parte del equipo de seguridad para ejecutar por demanda y manualmente evaluaciones de seguridad con herramientas DAST, de esta manera identificar vulnerabilidades para asegurar el despliegue de las aplicaciones de acuerdo con la metodología de desarrollo de software seguro de la organización. OWASP Zed Proxy Attack – ZAP es uno de los escáneres de seguridad más utilizados por parte de los profesionales de seguridad para identificar vulnerabilidades en el código y en la configuración instancia/servidor. A través de la herramienta Zed Attack Proxy de OWASP se realiza el análisis automático de vulnerabilidades sobre la aplicación Web, teniendo como base el documento OWASP Top 10 (Commons, 2017).

Política de escaneo

La solución permite la parametrización de reglas activas para identificar vulnerabilidades potenciales mediante el uso de ataques conocidos, de esta manera:

- Número de hosts analizados
- Subprocesos de escaneo
- Duración máxima de la regla
- Duración máxima de escaneo

La configuración del recurso Spider permite descubrir nuevos recursos automáticamente sobre la aplicación, analizando los hipervínculos identificados.

7. Recomendaciones

Como parte de la ejecución de este proyecto es importante tener en cuenta los siguientes aspectos y recomendaciones:

- En la coyuntura actual, el enfoque de seguridad debe estar alineado con las nuevas tecnologías y modelos de integración y despliegue continuo, sirviendo como habilitante en los procesos organizacionales.
- Sin embargo, no se debe perder de vista los esquemas tradicionales de aseguramiento de infraestructura (Hardeninig, mínimo privilegio, separación de zonas de seguridad y componentes de borde) ya que los vectores clásicos de ataque y vulnerabilidades siguen estando vigentes en estos modelos.
- Las herramientas open source pueden constituir un poderoso componente en el aseguramiento de nuevas tecnologías, la tendencia de uso de las mismas está en auge debido a su rápido ciclo de desarrollo y adaptación, y el creciente número de organizaciones que soportan y contribuyen a las mismas.
- En el análisis de riesgo de la implementación, se identificó que el uso de herramientas Open Source y nuevas tecnologías puede representar un riesgo para el manejo de información crítica si esta no es correctamente configurada, se recomienda siempre implementar los componentes de seguridad con los que cuenta la herramienta, así esto implique el uso de la versión empresarial, y hacer uso de herramientas con trayectoria y reconocimiento. En caso de no contar con mecanismos de aseguramiento suficientes, se deben implementar controles complementarios con herramientas externas que garanticen un nivel de riesgo aceptable para la organización.
- El uso de nuevas tecnologías de desarrollo y despliegue continuo permiten agilizar y dar cumplimiento a las exigencias de los clientes digitales en un entorno altamente competitivo, es una obligación de los profesionales y áreas de seguridad de las organizaciones adaptarse y evolucionar con este cambio, haciendo uso de estas nuevas tecnologías y herramientas para mejorar los procesos de aseguramiento y calidad de las aplicaciones.
- El uso de contenedores y orquestadores de contenedores, representan un nuevo paradigma y un nuevo escenario de ataque, con nuevas vulnerabilidades como se evidencia en este documento, es necesario un entendimiento y correcto aseguramiento de su implementación en diferentes capas, como lo son el sistema operativo, la

infraestructura y segmentación de red, su configuración siguiendo estándares y mejores prácticas, y finalmente registrando y monitoreando el comportamiento del sistema en tiempo real.

- La viabilidad financiera del proyecto está asociado a los costos fijos, costos variables y punto de equilibrio, de esta manera al tratarse de un servicio interno es importante identificar los valores a tenerse en cuenta como ingresos para obtener un modelo financiero.
- Es necesario en el momento de definir las herramientas a integrar como controles al flujo CI/CD, analizar la infraestructura que se va a adquirir como servicio en el proveedor de nube y de esta manera no acarrear con gastos innecesarios.

8. Referencias

NIST SP -800-190. (25 de Septiembre de 2017). *National Institute for Standards and Technology*. Obtenido de Application Container Security Guide: <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-190.pdf>

THENEWSTACK. (5 de Julio de 2018). *CI/CD WITH KUBERNETES: TOOLS AND PRACTICES*. Obtenido de The New Stack: <https://thenewstack.io/ci-cd-with-kubernetes-tools-and-practices/>

MICROSOFT. (10 de Abril de 2019). *Building a CI/CD pipeline for microservices on Kubernetes*. Obtenido de Microsoft Azure: <https://docs.microsoft.com/en-us/azure/architecture/microservices/ci-cd-kubernetes>

Aquasec. (2018). *Aquasec*. Obtenido de Aqua Security: <https://www.aquasec.com/wiki/display/containers/What+is+a+Container+Image>

Docker. (2019). *Docker*. Obtenido de Docker Inc.: <https://www.docker.com/resources/what-container>

NIST SP 800-60 Vol. 1 Rev. 1. (01 de Agosto de 2008). *Guide for Mapping Types of Information and Information Systems to Security Categories*. Obtenido de National Institute of Standard and Technology: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=152106

Guía para la Gestión y Clasificación de Activos de Información. (15 de marzo de 2016). *Guía para la Gestión y Clasificación de Activos de Información*. Obtenido de Ministerio de las Telecomunicaciones MIntIC: https://www.mintic.gov.co/gestionti/615/articles-5482_G5_Gestion_Clasificacion.pdf

NIST FIPS 199. (01 de Febrero de 2004). *Standards for Security Categorization of Federal Information and Information Systems*. Obtenido de National Institute of Standards and Technology: <https://csrc.nist.gov/publications/detail/fips/199/final>

Sultan, S., Ahmad, I., & Dimitriou, T. (1 de Mayo de 2019). *Container Security: Issues, Challenges, and the Road Ahead*. Obtenido de IEEE Xplore Digital Library: <https://ieeexplore.ieee.org/document/8693491>

Anchore. (03 de septiembre de 2019). *Overview*. Obtenido de Anchore Enterprise Documentation: <https://docs.anchore.com/current/docs/overview/>

Commons, C. (2017). *OWASP Top 10 - Los diez riesgos más críticos en aplicaciones WEB*.

VMware Amazon Web Services. (Febrero de 2019). Obtenido de <https://aws-quickstart.s3.amazonaws.com/quickstart-vmware/doc/vmware-kubernetes-on-the-aws-cloud.pdf>