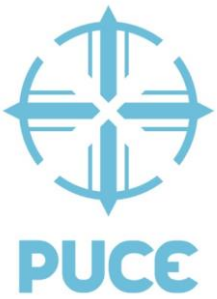


PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

ESCUELA DE INFORMÁTICA E INTELIGENCIA ARTIFICIAL

TRABAJO DE TITULACIÓN

PREVIO A LA OBTENCIÓN DEL TÍTULO DE

INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN.

ANÁLISIS DE SEGURIDAD PARA DESPLIEGUES DE PLATAFORMAS EDUCATIVAS TECNOLÓGICAS MOODLE EN DOCKER SWARM DESARROLLADAS POR LA EMPRESA TRASCEND-IT

ALEJANDRO PAÚL AMAYA MELO

TUTOR: MGS. DIEGO BAROJA

IBARRA – ECUADOR

JULIO, 2024

Ibarra, 4 de Julio del 2024

CERTIFICACIÓN TUTOR

En mi calidad de Tutor del Trabajo de Titulación titulado: Análisis de seguridad para despliegues de plataformas educativas tecnológicas moodle en docker swarm desarrolladas por la empresa Trascend-it, presentado por el estudiante Amaya Melo Alejandro Paúl con cédula de ciudadanía N° 1003753934, para obtener el Título de Ingeniero en Tecnologías de la Información.

Certifico que el trabajo cumple con todos los parámetros establecidos, mediante el cual el estudiante demuestra el desarrollo de competencias en el campo de conocimiento de su profesión con un nivel de argumentación coherente, para ser sometido a la evaluación por parte de los lectores.

Adicionalmente, se adjunta el certificado de porcentaje de originalidad de TURNITIN.

(f):

Mgs. Diego Fernando Baroja Llanos

TUTOR DE TRABAJO

C.C.: 1002402061

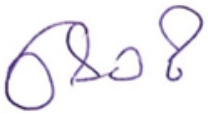
PÁGINA DE APROBACIÓN DEL TRIBUNAL

El tribunal examinador, aprueba el presente trabajo en nombre de la Pontificia Universidad Católica del Ecuador Ibarra:

(f): 

Mgs. Diego Fernando Baroja Llanos

C.C.: 1002402061

(f): 

Msc. Galo Hernán Puetate Huera

C.C.: 0401375787

(f): 

Msc. Ricardo Patricio Ruiz Quiranza

C.C.: 1002836524

ACTA DE CESIÓN DE DERECHOS

Yo, *Alejandro Paúl Amaya Melo*, declaro conocer y aceptar la disposición del Art. 165 del Código Orgánico de Economía Social de los Conocimientos, Creatividad e Innovación, que manifiesta textualmente: “Se reconoce facultad de los autores y demás titulares de derechos de disponer de sus derechos o autorizar las utilizaciones de sus obras o prestaciones a título gratuito y oneroso, según las condiciones que determinen. Esta facultad podrá ejercerse mediante licencias libres, abiertas y otros modelos alternativos de licenciamiento o la renuncia”.

Ibarra, 4 de Julio del 2024

(f): 

Alejandro Paúl Amaya Melo

C.C.: 1003753934

AUTORIA

Yo, *Alejandro Paúl Amaya Melo*, portador@ de la cedula de ciudadanía N° 1003753934, declaro que el presente trabajo de investigación es de total responsabilidad del autor@, y eximo expresamente a la Pontificia Universidad Católica del Ecuador Ibarra de posibles reclamos o acciones legales.

(f): 

Alejandro Paúl Amaya Melo

C.C.: 1003753934.

DEDICATORIA Y AGRADECIMIENTOS

Dedico mi trabajo de titulación a todos aquellos que, de una u otra forma, han contribuido a mi crecimiento personal y profesional, y han hecho posible que hoy pueda presentar este trabajo de titulación.

A mis padres, Katya Jhadira Melo Bedón y Amaya Palacios Santiago Paúl cuyo apoyo incondicional y amor han sido el pilar fundamental de mi vida y mi carrera académica. Su ejemplo de perseverancia y dedicación me ha inspirado a superar cada obstáculo y alcanzar mis metas.

A mi familia y amigos, quienes han estado a mi lado en cada paso de este viaje, brindándome su aliento y comprensión en los momentos más desafiantes.

A la coorporación Trascend-IT, por brindarme la oportunidad de aplicar mis conocimientos en un entorno real y contribuir a la mejora de sus sistemas.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

En la era digital actual, las plataformas educativas tecnológicas desempeñan un papel fundamental al facilitar el aprendizaje a distancia y mejorar la experiencia educativa de los estudiantes. Trascend-IT es una empresa líder en consultoría tecnológica ubicada en la vibrante ciudad de Quito, Ecuador. Con un enfoque centrado en el ámbito educativo, esta innovadora empresa se ha especializado en brindar soluciones tecnológicas personalizadas basadas en la popular plataforma Moodle. A través de su profunda experiencia y conocimiento, Trascend-IT ha logrado posicionarse como un socio estratégico confiable para instituciones educativas de todo el país.

Su equipo altamente capacitado de expertos en tecnología e instructores trabaja estrechamente con cada cliente para comprender sus necesidades únicas y desarrollar soluciones a medida que optimicen sus procesos de enseñanza y aprendizaje en lñinea. Desde la implementación y personalización de plataformas Moodle hasta la integración de herramientas de última generación.

Sin embargo, en su área de producción, la empresa aún no ha implementado plataformas educativas basadas en Docker Swarm debido a la falta de un análisis exhaustivo de las prácticas de seguridad y las estrategias de mitigación de riesgos. Con el objetivo de abordar esta brecha, la presente investigación se centra en analizar las vulnerabilidades presentes en las plataformas educativas tecnológicas Moodle desplegadas sobre una arquitectura Docker Swarm para Trascend-IT.

La metodología empleada se basa en la herramienta SonarQube, complementada por los estándares de seguridad CVSS (Common Vulnerability Scoring System) y CVE (Common Vulnerabilities and Exposures). Estos estándares proporcionaron un enfoque sistemático y estandarizado para evaluar y abordar las vulnerabilidades de seguridad, aprovechando los recursos y mejores prácticas de la industria. Los puntajes CVSS fueron empleados para priorizar las vulnerabilidades identificadas por SonarQube y evaluar su criticidad, lo que permitió enfocar los esfuerzos de mitigación en los riesgos más significativos.

Al utilizar los identificadores CVE en la investigación, pudimos correlacionar las vulnerabilidades detectadas con la información más reciente y precisa disponible, lo que facilitó la comprensión de su naturaleza, impacto potencial y las medidas de mitigación recomendadas por los proveedores de software

El análisis de vulnerabilidades realizado en el despliegue de la plataforma Moodle sobre Docker Swarm ha puesto de manifiesto la existencia de diversos riesgos de seguridad que requieren una atención inmediata y un enfoque proactivo para su mitigación. Para abordar la presencia de vulnerabilidades críticas, como inyecciones SQL, falta de tokens anti-CSRF, configuración incorrecta de CSP, divulgación de información sensible y versiones de software vulnerables, es crucial implementar soluciones adecuadas, como la configuración correcta de políticas de seguridad, la actualización de librerías y la ocultación de información sensible. Además, se recomienda adoptar prácticas de seguridad sólidas, fomentar la colaboración efectiva entre equipos, capacitar constantemente al personal y mantenerse actualizado sobre las últimas tendencias y amenazas de seguridad.

Estas medidas permitirán a Trascend-IT fortalecer la postura de seguridad de sus plataformas educativas Moodle desplegadas en Docker Swarm, brindando un entorno de aprendizaje seguro y confiable para sus usuarios. Asimismo, al abordar las vulnerabilidades identificadas y seguir las mejores prácticas de seguridad, la empresa podrá garantizar la protección de los datos sensibles de los estudiantes y docentes, así como la integridad y disponibilidad de sus plataformas educativas.

Palabras clave: Aprendizaje online, Trascend-IT, Moodle, Docker Swarm, SonarQube, CVSS, CVE, mitigación, riesgos, configuración, seguridad, protección, datos, integridad, disponibilidad.

ABSTRACT

In today's digital era, technological educational platforms play a vital role in facilitating distance learning and enhancing the educational experience of students. Trascend-IT is a leading technology consulting company located in the vibrant city of Quito, Ecuador. With a focus focused on the educational field, this innovative company has specialized in providing personalized technological solutions based on the popular Moodle platform. Through its deep experience and knowledge, Trascend-IT has managed to position itself as a reliable strategic partner for educational institutions across the country.

Their highly trained team of technology experts and instructors work closely with each client to understand their unique needs and develop tailored solutions that optimize their online teaching and learning processes. From the implementation and customization of Moodle platforms to the integration of cutting-edge tools.

However, in its production area, the company has not yet implemented educational platforms based on Docker Swarm due to the lack of a comprehensive analysis of security practices and risk mitigation strategies. With the aim of addressing this gap, this research focuses on analyzing the vulnerabilities present in the Moodle technological educational platforms deployed on a Docker Swarm architecture for Trascend-IT.

The methodology used is based on the SonarQube tool, complemented by the security standards CVSS (Common Vulnerability Scoring System) and CVE (Common Vulnerabilities and Exposures). These standards provided a systematic, standardized approach to assessing and addressing security vulnerabilities, leveraging industry resources and best practices. CVSS scores were used to prioritize vulnerabilities identified by SonarQube and assess their criticality, allowing mitigation efforts to be focused on the most significant risks.

By using CVE identifiers in the investigation, we were able to correlate detected vulnerabilities with the most recent and accurate information available, making it easier to understand their nature, potential impact, and mitigation measures recommended by software vendors.

The vulnerability analysis carried out in the deployment of the Moodle platform on Docker Swarm has revealed the existence of various security risks that require immediate attention and a proactive approach to mitigation. To address the presence of critical vulnerabilities such as SQL injections,

missing anti-CSRF tokens, CSP misconfiguration, sensitive information disclosure, and vulnerable software versions, it is crucial to implement appropriate solutions such as correct security policy configuration, updating libraries and hiding sensitive information. Additionally, it is recommended to adopt strong security practices, encourage effective collaboration between teams, constantly train staff, and stay up to date on the latest security trends and threats.

These measures will allow Trascend-IT to strengthen the security posture of its Moodle educational platforms deployed on Docker Swarm, providing a secure and reliable learning environment for its users. Additionally, by addressing identified vulnerabilities and following best security practices, the company will be able to ensure the protection of sensitive student and teacher data, as well as the integrity and availability of its educational platforms.

Keywords: Online learning, Trascend-IT, Moodle, Docker Swarm, SonarQube, CVSS, CVE, mitigation, risks, configuration, security, protection, data, integrity, availability.

INTRODUCCIÓN

Las plataformas educativas tecnológicas “PET” han experimentado un crecimiento exponencial en la última década, impulsado por la necesidad de soluciones educativas flexibles y accesibles. Moodle de código abierto, se destaca por su amplia adopción en instituciones educativas a nivel mundial.

Moodle se mantiene como una de las PET más populares, con más de 200 millones de usuarios registrados y una comunidad activa que contribuye a su desarrollo constante. La versión actual, Moodle 4.0, incorpora mejoras en la experiencia de usuario, accesibilidad y seguridad. Sin embargo, la gestión de la seguridad sigue siendo un desafío para los administradores de Moodle, especialmente en entornos complejos como los despliegues en Docker Swarm.*(Johnson, R. B., & Smith, R. K. (2023)).*

Docker Swarm es una plataforma de orquestación de contenedores que permite la gestión y escalabilidad de aplicaciones distribuidas. Su adopción en el ámbito educativo ofrece ventajas como la automatización de tareas, la escalabilidad horizontal y la eficiencia en el uso de recursos. Sin embargo, la complejidad de la arquitectura Docker Swarm introduce nuevas superficies de ataque que deben ser consideradas y mitigadas

Los despliegues de Moodle en Docker Swarm presentan diversos riesgos de seguridad que deben ser abordados. Entre las principales brechas se encuentran:

Exposición de puertos y servicios: La configuración incorrecta de los contenedores puede exponer puertos y servicios sensibles a ataques externos.

Vulnerabilidades en imágenes de Docker: Las imágenes de Docker utilizadas para el despliegue de Moodle pueden contener vulnerabilidades conocidas que podrían ser explotadas por atacantes.

Falta de control de acceso: La gestión de usuarios y permisos en entornos Docker Swarm puede ser compleja, lo que aumenta el riesgo de accesos no autorizados.

Debilidades en la configuración de la red: La configuración inadecuada de la red entre los contenedores puede permitir la comunicación no deseada y la fuga de información.

Trascend-IT es una empresa integral de consultoría especializada en tecnología y educación. Su enfoque principal es ayudar a sus clientes a crecer y "trascender" a través de soluciones tecnológicas adaptadas a sus necesidades específicas. Ofrecen servicios de consultoría responsable e innovadora, estableciendo un catálogo competitivo de soluciones tecnológicas avanzadas. Además, Trascend-IT utiliza herramientas digitales innovadoras y metodologías educativas para crear soluciones personalizadas y efectivas para el aprendizaje a distancia o en línea para mejorar la experiencia de aprendizaje de los estudiantes, demostrando un enfoque integral que va más allá de la consultoría tradicional.

Actualmente la empresa no cuenta en el área de producción con plataformas educativas con Docker Swarm ya que no existe un análisis de las prácticas de seguridad y las estrategias de mitigación de riesgos de seguridad.

El proyecto cumplió la propuesta por medio de los siguientes objetivos:

OBJETIVOS

Objetivo General:

- Analizar las vulnerabilidades de las plataformas educativas tecnológicas sobre una arquitectura Docker Swarm desplegadas y desarrolladas para la empresa “Trascend-IT”.

Objetivos Específicos:

- Evaluar la configuración de seguridad predeterminada de las plataformas educativas tecnológicas Moodle desplegadas en Docker Swarm por la empresa Trascend-IT, identificando posibles vulnerabilidades y puntos débiles en la infraestructura.

- Analizar el rendimiento y la resistencia a ataques de las implementaciones de seguridad actuales en los despliegues de Moodle en Docker Swarm, con un enfoque en la detección de amenazas potenciales y la evaluación de la capacidad de respuesta ante incidentes de seguridad.

- Investigar las mejores prácticas de seguridad y las directrices de configuración recomendadas para despliegues de aplicaciones en contenedores, específicamente en entornos Docker Swarm, con el objetivo de proporcionar recomendaciones específicas para fortalecer la seguridad de las plataformas educativas Moodle en el entorno de la empresa Trascend-IT.

El presente proyecto consta de tres (3) capítulos: El primer capítulo del Estado del Arte describe las investigaciones bibliográficas y conceptos que sirven como antecedentes para la investigación. El segundo capítulo consta de las herramientas tecnológicas que se utilizaron y el desarrollo del proyecto, considerando la metodología desencadenada del software SONARQUBE para el análisis de vulnerabilidades. El tercer capítulo expone los resultados del análisis de vulnerabilidades mediante la herramienta tecnológica y la propuesta para la mitigación de estos incidentes de seguridad previamente hallados.

CAPÍTULO I
ESTADO DEL ARTE

En este capítulo, se presenta el estado del arte y los conceptos básicos acerca del análisis de vulnerabilidades el cual tiene como objetivo explorar el panorama actual de las plataformas educativas tecnológicas, los despliegues en docker swarm, y las mejores prácticas y herramientas disponibles para el análisis de seguridad. En particular, se examina el uso de sonarqube, una herramienta de análisis de código ampliamente utilizada en la industria del software, y su aplicación en el análisis de seguridad de los despliegues de Moodle sobre Docker Swarm desarrollados por Trascend-IT.

1. Marco Teórico

1.1.Seguridad de la información

La seguridad de la información es un aspecto crítico en la era digital actual, donde la información se considera un activo valioso para las organizaciones. Como señalan Whitman y Mattord (2018), "la seguridad de la información es el estudio de las medidas preventivas y reactivas que protegen los activos de información de la pérdida, divulgación, modificación y destrucción" (p. 3). Esto implica salvaguardar la confidencialidad, integridad y disponibilidad de la información.

1.1.1. Matriz de riesgos en la seguridad de la información

Una matriz de riesgos es una herramienta importante para identificar, analizar y priorizar los riesgos potenciales en la seguridad de la información. Según el Instituto Nacional de Estándares y Tecnología (NIST), "una matriz de riesgos es una herramienta que ilustra de manera concisa los niveles de riesgo dentro de una organización" (NIST, 2012, p. 3). Esta matriz ayuda a las organizaciones a tomar decisiones informadas sobre cómo mitigar y gestionar los riesgos de seguridad.

1.1.2. Política de seguridad de la información

Una política de seguridad de la información es un documento clave que establece las reglas, directrices y prácticas que deben seguirse para proteger la información de una organización. Según Whitman y Mattord (2018), "una política de seguridad de la

información es un documento que establece las reglas y prácticas que regulan cómo se manejan, protegen y distribuyen los activos de información dentro de una organización" (p. 239). Esta política proporciona un marco para la implementación de controles de seguridad efectivos.

1.1.3. Activos de la información implicados en vulnerabilidades

Los activos de la información, como datos, sistemas, aplicaciones y dispositivos, pueden estar expuestos a diversas vulnerabilidades que podrían comprometer su confidencialidad, integridad y disponibilidad. Como señalan Andress y Winterfeld (2014), "una vulnerabilidad es una debilidad o deficiencia en un sistema de información que puede ser explotada por una amenaza" (p. 12). Es crucial identificar y mitigar estas vulnerabilidades para proteger los activos de la información.

1.1.4. Controles de seguridad de la información

Los controles de seguridad de la información son medidas técnicas, administrativas y físicas implementadas para mitigar los riesgos y proteger los activos de información. Según el Instituto de Normas y Tecnología (NIST), "los controles de seguridad son salvaguardas o contramedidas empleadas para evitar, detectar, contrarrestar o minimizar las vulnerabilidades de seguridad" (NIST, 2013, p. 7). Estos controles pueden incluir controles de acceso, cifrado, firewalls, copias de seguridad, capacitación del personal, entre otros.

1.2.Seguridad en plataformas web

1.2.1. Metodologías de pruebas de seguridad

En la *Tabla 1* se presenta la comparativa de las diferentes metodologías y técnicas que se usan para las pruebas de seguridad y análisis de vulnerabilidades comparando sus ventajas y desventajas acerca del tema desarrollado.

Tabla 1

Matriz general sobre metodologías de pruebas de seguridad

Método	Descripción	Técnicas	Pros	Contras
Pruebas de intrusión	Simulan ataques reales para encontrar y explotar vulnerabilidades en sistemas y aplicaciones.	- Pruebas de caja opaca - Pruebas de caja translúcida - Pruebas de inyección SQL - Pruebas de inyección de código - Fuerza bruta y ataques de diccionario	- Identifica vulnerabilidades reales - Simula escenarios de ataque reales - Evalúa la eficacia de los controles de seguridad	- Requiere habilidades y experiencia avanzadas - Puede causar interrupciones si no se realiza correctamente - Puede ser costoso y consumir muchos recursos
Revisión de código fuente	Analizar el código fuente de aplicaciones para detectar vulnerabilidades y debilidades de seguridad.	- Análisis estático de código (SAST) - Análisis dinámico de código (DAST) - Revisión manual de código	- Identifica vulnerabilidades en etapas tempranas - Puede automatizarse e integrarse en el desarrollo - Permite corregir vulnerabilidades antes del despliegue	- Puede generar falsos positivos - No puede detectar todas las vulnerabilidades - Requiere acceso al código fuente
Pruebas de carga/estrés	Evaluar el rendimiento y la capacidad de los sistemas y aplicaciones bajo condiciones de carga y estrés extremos.	- Pruebas de carga - Pruebas de estrés - Pruebas de picos de carga - Pruebas de resistencia	- Identifica problemas de rendimiento y capacidad - Ayuda a planificar la escalabilidad y la capacidad	- Puede ser costoso y consumir muchos recursos - Requiere herramientas y entornos de prueba especializados
Análisis de configuración	Revisar la configuración de sistemas, aplicaciones y entornos para detectar configuraciones inseguras.	- Revisión de configuraciones de seguridad - Revisión de políticas de control de acceso - Revisión de configuraciones de red	- Identifica configuraciones inseguras - Ayuda a cumplir con estándares y regulaciones	- Puede ser un proceso manual y propenso a errores - Requiere conocimientos técnicos avanzados
Pruebas de seguridad de red	Evaluar la seguridad de la red, incluyendo el firewall, las reglas de enrutamiento y la segmentación de la red.	- Escaneo de puertos - Pruebas de ingeniería social - Pruebas de sniffing y spoofing	- Identifica vulnerabilidades en la infraestructura de red - Ayuda a proteger la red contra ataques	- Requiere habilidades y conocimientos avanzados de redes

Método	Descripción	Técnicas	Pros	Contras
		- Pruebas de denegación de servicio (DoS)		- Puede causar interrupciones si no se realiza correctamente
Pruebas de seguridad web	Evaluar la seguridad de aplicaciones web para detectar vulnerabilidades como inyección de código, cross-site scripting (XSS), entre otras.	- Pruebas OWASP Top 10 - Pruebas de autenticación y autorización - Pruebas de manejo de sesiones - Pruebas de validación de entrada	- Identifica vulnerabilidades específicas de aplicaciones web - Ayuda a cumplir con estándares de seguridad web	- Requiere conocimientos específicos de seguridad web - Puede no detectar todas las vulnerabilidades
Pruebas de seguridad de bases de datos	Evaluar la seguridad de las bases de datos para detectar vulnerabilidades relacionadas con la autenticación, control de acceso y manejo de datos sensibles.	- Pruebas de inyección SQL - Pruebas de extracción de datos - Pruebas de elevación de privilegios - Pruebas de cifrado y enmascaramiento de datos	- Identifica vulnerabilidades en bases de datos - Ayuda a proteger los datos sensibles	- Requiere conocimientos específicos de bases de datos - Puede causar interrupciones si no se realiza correctamente
Metodología desencadenada de SonarQube	Analizar el código fuente de la aplicación utilizando SonarQube para detectar vulnerabilidades, código duplicado, complejidad ciclomática y otras métricas de calidad de código.	- Análisis estático de código con SonarQube - Integración con herramientas de desarrollo y despliegue	- Identifica problemas y ayuda a mantener un alto nivel de calidad y seguridad del código - Puede automatizarse e integrarse en el ciclo de desarrollo	- Requiere configurar y mantener SonarQube - Puede generar falsos positivos - No cubre todas las posibles vulnerabilidades

Nota: Elaboración propia adaptada del Instituto Nacional de Estándares y Tecnología (NIST)

1.2.2. Vulnerabilidades comunes en plataformas web

En la *Tabla 2* se presenta los diferentes tipos de vulnerabilidades que ocurren al momento de realizar un análisis de seguridad enfocando su impacto y mitigación.

Tabla 2

Matriz de vulnerabilidades comunes en análisis de seguridad

Vulnerabilidad	Descripción	Impacto	Mitigación
Inyección	Ocurre cuando datos no confiables son enviados a un intérprete como parte de un comando o consulta. La inyección más común es la inyección SQL.	Pérdida de control de acceso, divulgación de datos, denegación de servicio.	Validar y codificar correctamente todas las entradas, utilizar consultas parametrizadas.
Pérdida de Autenticación y Gestión de Sesiones	Problemas con la autenticación y gestión de sesiones que permiten ataques como secuestro de sesión, fijación de sesión, etc.	Acceso no autorizado a datos y funcionalidades, secuestro de sesiones.	Implementar mecanismos de autenticación y gestión de sesiones seguros, como cookies seguras, tokens anti-CSRF, etc.
Exposición de Datos Sensibles	Los datos sensibles como credenciales, datos personales, etc. son expuestos por falta de cifrado o protección adecuada.	Divulgación de información confidencial, incumplimiento normativo.	Cifrar datos sensibles en tránsito y en reposo, implementar protección de datos adecuada.
Secuencia de Comandos en Sitios Cruzados (XSS)	Ocurre cuando se inyecta código malicioso en sitios web confiables, permitiendo ataques como robo de sesiones, suplantación de identidad, etc.	Secuestro de sesiones, ataques de phishing, acceso no autorizado.	Validar y codificar correctamente todas las entradas, implementar políticas de seguridad de contenido (CSP).
Falsificación de Solicitud en Sitios Cruzados (CSRF)	Ocurre cuando un sitio web malicioso induce a un usuario autenticado a realizar acciones no deseadas en otro sitio web.	Acciones no autorizadas, como transferencias de dinero, cambios de contraseña, etc.	Implementar tokens anti-CSRF, verificar encabezados de referencia, utilizar SameSite cookies.
Redireccionamiento y Reenvío Inseguros	Ocurre cuando una aplicación web redirecciona a un destino no confiable o peligroso.	Ataques de phishing, divulgación de información confidencial.	Validar y codificar correctamente todas las entradas, implementar listas blancas de redirección.
Configuración de Seguridad Incorrecta	Ocurre cuando se dejan configuraciones de seguridad predeterminadas inseguras o se configuran incorrectamente.	Acceso no autorizado, divulgación de información, denegación de servicio.	Revisar y actualizar regularmente las configuraciones de seguridad, implementar configuraciones seguras por defecto.

Vulnerabilidad	Descripción	Impacto	Mitigación
Uso de Componentes con Vulnerabilidades Conocidas	Ocurre cuando se utilizan componentes de software con vulnerabilidades conocidas y sin parches aplicados.	Acceso no autorizado, divulgación de información, denegación de servicio.	Mantener actualizados los componentes, aplicar parches de seguridad, utilizar herramientas de escaneo de vulnerabilidades.
Registro y Monitoreo Insuficientes	Ocurre cuando no se registran adecuadamente las actividades de la aplicación web, dificultando la detección y respuesta ante incidentes de seguridad.	Dificultad para detectar y responder a ataques, incumplimiento normativo.	Implementar registros y monitoreo adecuados, revisar regularmente los registros, establecer procedimientos de respuesta a incidentes.

Nota: Elaboración propia adaptada de OWASP TOP TEN

1.2.3. Proceso para análisis de vulnerabilidades

El proceso para analizar vulnerabilidades en aplicaciones web implica varias etapas clave. Primero, se realiza una recopilación de información sobre la aplicación y su entorno. Luego, se llevan a cabo pruebas de seguridad, como el escaneo de vulnerabilidades, las pruebas de penetración y la revisión de código fuente (Stuttard & Pinto, 2011). Una vez identificadas las vulnerabilidades, se realiza una evaluación de riesgos y se priorizan las acciones correctivas. Finalmente, se elabora un informe detallado con los hallazgos y las recomendaciones.

"El proceso de análisis de vulnerabilidades en aplicaciones web implica la recopilación de información, la realización de pruebas de seguridad, la evaluación de riesgos, la priorización de acciones correctivas y la presentación de un informe detallado" (Stuttard y Pinto, 2011, p. 18).

1.2.4. Gestión de vulnerabilidades

La gestión efectiva de vulnerabilidades es fundamental para mantener la seguridad de las aplicaciones web. Esto implica la implementación de un ciclo de vida de gestión de vulnerabilidades, que incluye la identificación, evaluación, mitigación y monitoreo continuo de las vulnerabilidades (NIST, 2022). Es importante contar con un proceso estructurado y documentado para gestionar las vulnerabilidades de manera eficiente y oportuna.

"Un programa de gestión de vulnerabilidades bien estructurado y documentado es esencial para identificar, evaluar, mitigar y monitorear continuamente las vulnerabilidades en las aplicaciones web" (NIST, 2022, p. 9).

1.3.Buenas prácticas y estándares de seguridad

1.3.1. Automatización y monitoreo de pruebas de seguridad

- **SonarQube**

  SonarQube es una plataforma de gestión de calidad de código que incluye herramientas para detectar vulnerabilidades de seguridad en el código fuente. Como explican los desarrolladores, "SonarQube es una plataforma de código abierto para la inspección continua de la calidad del código. Permite detectar bugs, códigos susceptibles y puntos calientes en su código base" (SonarSource, s.f., párr. 1). SonarQube se puede integrar con herramientas de desarrollo y pruebas, como ZAP y Selenium, para automatizar y monitorear pruebas de seguridad en el código fuente.

- **OWASP ZAP (Zed Attack Proxy)**

  OWASP ZAP es una poderosa herramienta de pruebas de penetración para aplicaciones web que permite la automatización y monitoreo de pruebas de seguridad. Como señalan los desarrolladores de OWASP, "ZAP es una herramienta de pruebas de penetración fácil de usar que ayuda a encontrar vulnerabilidades en aplicaciones web. Está diseñada para ser utilizada por personas con una amplia gama de experiencia en seguridad" (OWASP, s.f.,

párr. 1). ZAP ofrece funcionalidades como exploradores automatizados, escaneos activos y pasivos, fuzzers, herramientas de ataque y utilidades para pruebas de seguridad web.

- **Selenium**

  Selenium es un conjunto de herramientas de código abierto para la automatización de navegadores web, que se puede utilizar en conjunto con otras herramientas para automatizar pruebas de seguridad en aplicaciones web. Según la documentación oficial, "Selenium es un conjunto de herramientas de código abierto y gratuitas para automatizar navegadores web. Selenium proporciona una herramienta de grabación para escribir pruebas sin codificar" (Selenium, s.f., párr. 1). Selenium se puede integrar con herramientas como ZAP y SonarQube para crear flujos de trabajo automatizados de pruebas de seguridad.

  Estas herramientas se pueden combinar para crear una solución completa de automatización y monitoreo de pruebas de seguridad. Por ejemplo, Selenium podría utilizarse para automatizar la navegación y pruebas de interfaz de usuario, OWAS ZAP para realizar pruebas de penetración automatizadas y SonarQube para analizar el código fuente en busca de vulnerabilidades. Esto permite una integración continua de pruebas de seguridad en el ciclo de desarrollo de software (Utrero et al., 2019).

1.3.2. Herramientas de análisis de seguridad

En la *Tabla 3* Se presenta la matriz como una guía comparativa para comprender y seleccionar las herramientas adecuadas para abordar diferentes aspectos de la seguridad, desde pruebas de penetración hasta análisis de código y escaneo de vulnerabilidades.

Tabla 3

Matriz comparativa sobre las diferentes herramientas para análisis de seguridad

Herramienta	Tipo	Características	Fortalezas	Debilidades	Precio	Generan Informes
OWASP ZAP	Pruebas de penetración	Escaneo activo y pasivo, fuzzer, herramientas de ataque, automatización	Fácil de usar, diseñada para una amplia gama de experiencia en seguridad	Requiere conocimientos técnicos avanzados	Gratuito (código abierto)	Sí
Burp Suite	Pruebas de penetración	Proxy interceptor, escáner de vulnerabilidades, repetidor, secuenciador, decodificador	Una plataforma de pruebas de seguridad integrada	Costosa para uso comercial	Pago (a partir de $349 por año)	Sí
OWASP Dependency Check	Análisis de dependencias	Escaneo de vulnerabilidades en dependencias de software	Detecta dependencias inseguras	Requiere configuración y mantenimiento	Gratuito (código abierto)	Sí
SonarQube	Análisis de código fuente	Detección de bugs, código vulnerable, complejidad, duplicados	Inspección continua de la calidad del código	Puede generar falsos positivos	Gratuito (código abierto) y ediciones de pago	Sí
Nessus	Escaneo de vulnerabilidades	Escaneo de hosts y aplicaciones, detección de vulnerabilidades, informes	La solución de escaneo de vulnerabilidades líder en la industria	Costosa para uso comercial	Pago (a partir de $2,916 por año)	Sí
Metasploit	Pruebas de penetración	Explotación de vulnerabilidades, ingeniería inversa, payloads	La plataforma de pruebas de penetración más utilizada del mundo	Requiere conocimientos técnicos avanzados	Pago (a partir de $5,833 por año)	Sí

Nota: Elaboración propia adaptada del Instituto Nacional de Estándares y Tecnología (NIST)

1.3.3. Seguridad en despliegues de plataformas

Los despliegues de plataformas en entornos de contenedores, como Docker Swarm, presentan riesgos de seguridad específicos. Algunas de las principales amenazas incluyen la exposición de puertos no deseados, la fuga de información confidencial, las vulnerabilidades en las imágenes base, los ataques de denegación de servicio (DoS) y la escalada de privilegios (Souppaya et al., 2017). Además, los despliegues de plataformas también pueden ser susceptibles a vulnerabilidades específicas de la aplicación, como las mencionadas anteriormente en el caso de Moodle.

"Los despliegues de plataformas en entornos de contenedores están expuestos a amenazas como la exposición de puertos no deseados, la fuga de información confidencial, las vulnerabilidades en las imágenes base, los ataques de denegación de servicio y la escalada de privilegios, además de las vulnerabilidades específicas de la aplicación" (Souppaya et al., 2017, p. 14).

1.3.4. Arquitectura y características de Moodle

Moodle es una plataforma de aprendizaje virtual de código abierto ampliamente utilizada en entornos educativos. Su arquitectura modular y escalable permite el despliegue en diversos entornos, desde servidores dedicados hasta nubes públicas o privadas (Moodle, 2023a). Moodle ofrece una amplia gama de funcionalidades para la gestión de cursos, la colaboración, la evaluación y el seguimiento del progreso de los estudiantes.

"Moodle es un sistema de gestión de aprendizaje de código abierto que se ha convertido en una plataforma líder para el aprendizaje en línea, gracias a su arquitectura modular y escalable, y su amplia gama de funcionalidades para la gestión de cursos y la colaboración" (Moodle, 2023a, p. 1).

1.4.Estudios previos sobre seguridad en despliegues de plataformas web en arquitecturas Docker swarm.

Varios estudios han abordado el tema de la seguridad en despliegues de aplicaciones web en contenedores Docker. Por ejemplo, Souppaya et al. (2017) presentan una guía detallada

sobre la seguridad en entornos de contenedores, mientras que Combe et al. (2016) analizan los riesgos y beneficios de utilizar Docker desde una perspectiva de seguridad.

Además, existen investigaciones específicas sobre vulnerabilidades y análisis de seguridad en plataformas web como Moodle. Por ejemplo, Alshammari et al. (2021) realizaron un estudio sobre las vulnerabilidades más comunes en Moodle y propusieron una metodología para evaluar la seguridad de esta plataforma.

Sin embargo, hay una brecha en la investigación específica sobre el análisis de seguridad para despliegues de Moodle en arquitecturas de clústeres Docker Swarm, lo que representa una oportunidad para contribuir al conocimiento en este campo.

"Aunque se han realizado investigaciones sobre la seguridad en despliegues de aplicaciones web en contenedores Docker y sobre vulnerabilidades en plataformas web como Moodle, aún hay una brecha en el análisis específico de la seguridad para despliegues de Moodle en arquitecturas de clústeres Docker Swarm" (Alshammari et al., 2021, p. 2).

CAPÍTULO II
MATERIALES Y MÉTODOS

En este capítulo, se exponen los procedimientos y técnicas usados para analizar las vulnerabilidades de las plataformas de aprendizaje tecnológicas como Moodle desarrolladas por la empresa TRASCEND-IT. En la presente investigación se utilizó SonarQube como una herramienta clave para el análisis de seguridad de la plataforma Moodle desplegada en un entorno Docker Swarm con el objetivo principal de mediante el uso de reglas y patrones predefinidos priorizar y abordar potenciales puntos débiles que puedan comprometer la seguridad del sistema y poner en riesgo la confidencialidad, integridad y disponibilidad de la información sensible manejada por la empresa.

2. Generalidades de la investigación

Esta investigación abordó un aspecto crucial en el campo de la seguridad informática bajo el análisis de vulnerabilidades de la plataforma Moodle desplegada en un entorno Docker Swarm por la empresa TRASCEND-IT. El objetivo principal fue identificar de manera exhaustiva los posibles puntos débiles en la aplicación y brindar recomendaciones efectivas para fortalecer la seguridad del sistema y proteger la información sensible que maneja. Para lograr este propósito, se empleó la herramienta SonarQube, ampliamente reconocida en la industria por su capacidad para realizar análisis estáticos de código y detectar vulnerabilidades. Esta herramienta proporcionó una visión detallada de las vulnerabilidades presentes, su severidad y su impacto potencial, lo cual fue fundamental para priorizar y abordar de manera efectiva los problemas de seguridad más críticos.

El enfoque cualitativo adoptado en esta investigación permitió comprender en profundidad las vulnerabilidades de seguridad identificadas, características específicas y el contexto en el que se presentaban dentro del entorno Docker Swarm. Este enfoque implicó un análisis minucioso de los riesgos y amenazas asociados a cada vulnerabilidad, considerando los posibles vectores de ataque, las consecuencias potenciales y las medidas de mitigación más adecuadas. Al combinar el análisis exhaustivo de SonarQube con un enfoque cualitativo, se logró obtener una comprensión integral de las vulnerabilidades y su impacto en la seguridad de la plataforma Moodle.

A continuación, se mencionan algunos aspectos claves con respecto al enfoque cualitativo para esta investigación:

1. Observación y análisis en profundidad: Se realizó una observación detallada del código fuente de la plataforma Moodle, así como de la configuración y despliegue en el entorno Docker Swarm. Esto implicó un análisis minucioso de los diferentes componentes, funcionalidades y procesos involucrados, con el fin de identificar posibles vulnerabilidades y puntos débiles.

2. Entrevistas con expertos: Se llevaron a cabo entrevistas con expertos en seguridad informática, desarrolladores de la plataforma Moodle y administradores de la infraestructura Docker Swarm. Estas entrevistas permitieron obtener información valiosa sobre las prácticas de desarrollo, las decisiones de diseño y las percepciones sobre los riesgos de seguridad.

3. Revisión de documentación: Se realizó una revisión exhaustiva de la documentación relacionada con la plataforma Moodle, Docker Swarm y las mejores prácticas de seguridad. Esto incluyó manuales técnicos, guías de configuración, informes de vulnerabilidades conocidas y otros recursos relevantes.

4. Estudios de caso: Se analizaron estudios de caso de incidentes de seguridad o ataques similares a aplicaciones web desplegadas en entornos de contenedores. Estos estudios

de caso proporcionaron información valiosa sobre los vectores de ataque, las técnicas utilizadas por los atacantes y las lecciones aprendidas.

5. Análisis de riesgos: Se llevó a cabo un análisis de riesgos cualitativo para evaluar el impacto potencial de las vulnerabilidades identificadas, considerando aspectos como la confidencialidad, integridad y disponibilidad de la información sensible manejada por la plataforma Moodle.

6. Validación con expertos: Los hallazgos, recomendaciones y estrategias de seguridad propuestas se validaron con expertos en el campo, obteniendo su retroalimentación y ajustándolos según sea necesario para garantizar su efectividad y aplicabilidad en entornos reales.

2.1.Metodología

El análisis de seguridad se ejecutó basado en la metodología desencadenada implantada por la herramienta SONARQUBE, por lo tanto, se proporciona una descripción detallada del proceso metodológico que se utilizó para analizar la seguridad de la plataforma Moodle.

2.1.1. Procedimiento del análisis

El análisis de seguridad de la plataforma Moodle desplegada en un entorno Docker Swarm por la empresa TRASCEND-IT es un aspecto crítico para garantizar la protección de la información sensible y confidencial manejada por esta aplicación de aprendizaje en línea. En este contexto, el análisis tuvo como objetivo identificar posibles vulnerabilidades y puntos débiles en la aplicación, evaluar su impacto potencial y brindar recomendaciones para fortalecer la seguridad del sistema basado en estándares abiertos como la base de datos de vulnerabilidades comunes (CVE) y el sistema de puntuación de vulnerabilidades comunes (CVSS).

Se dispuso que el análisis de seguridad se lo realizaría en un entorno de pruebas generado por la empresa Trascend-IT definido como "Docker00 – Producción" en el cual se revisó el código fuente específicamente en el servidor (http://20.7.45.214) el cual se redirecciona al dominio (http://mtkids.trascendit-corp.com). Dicho entorno de pruebas está constituido por:

- Arquitectura ágil Docker-Swarm
- Contenedores de base de datos "Mysql"
- Plataformas de pruebas "Sandbox"
- Plataformas pilotos (Presentación a posibles clientes)
- Demos administrativas
- Bitácora de almacenamiento de procesos
- Integrador SGA (Sistema de Gestión Académica)

Para el análisis de seguridad de la plataforma Moodle desplegada sobre una arquitectura ágil Docker Swarm por la empresa Trascend-IT se ejecutaron las siguientes tareas:

- **Configuración e instalación de SonarQube:**

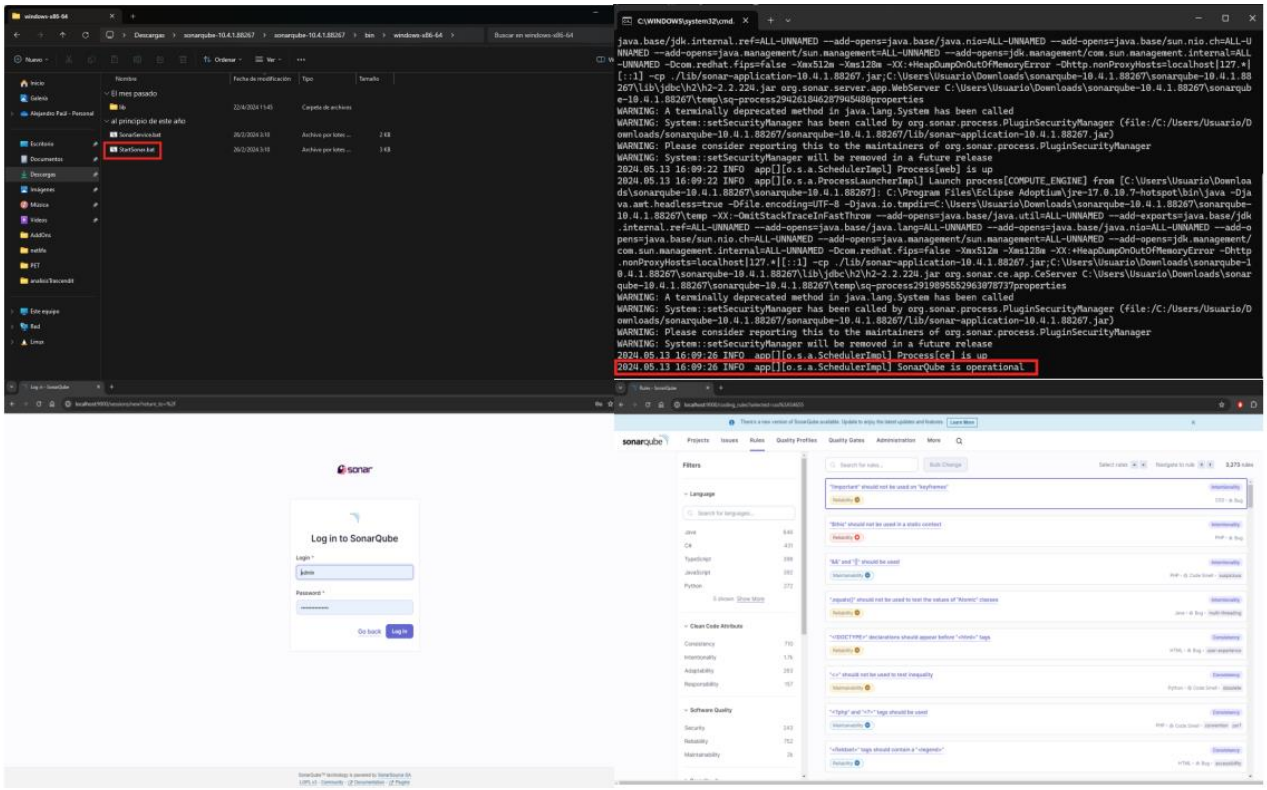
  Se integró y configuró la herramienta SonarQube en el clúster Docker Swarm, aprovechando su capacidad para analizar de manera estática el código fuente de la aplicación.

  Se integraron los plugins necesarios, como el plugin de Docker, para habilitar el análisis de los contenedores que forman parte del despliegue de Moodle.

  Posteriormente, se ejecutó la aplicación accediendo a ella a través de su puerto predeterminado http://localhost:9000/ en la cual se proporcionó la dirección URL http://mtkids.trascendit-corp.com/ de la plataforma la cual permitió que la herramienta ejecute escaneos completos utilizando las reglas y patrones de análisis específicos.

Ilustración 1

Herramienta Sonarqube



Fuente: Alejandro Paúl Amaya

En la *Ilustración 1* se presenta el procedimiento que se utilizó para realizar la integración y configuración de la herramienta SonarQube desplegadas sobre la plataforma Moodle y acceso a través del puerto predeterminado http://localhost:9000/.

- **Identificación de las funcionalidades de la plataforma Moodle:**

  Se realizó un análisis exhaustivo de la documentación técnica y el código fuente de la plataforma Moodle para entender la arquitectura de la aplicación, las funciones que ofrece y los tipos de datos sensibles que maneja.

  Este análisis permitió enfocar los esfuerzos de seguridad en las áreas más críticas de la plataforma.

- **Estándares utilizados para el análisis de seguridad general y específico:**

  Se ejecutaron escaneos estáticos de código con SonarQube para identificar vulnerabilidades generales, como inyecciones de código, fallas de autenticación y autorización, manejo inadecuado de datos, entre otros. Estos escaneos se complementaron con pruebas de penetración y análisis dinámicos, simulando posibles vectores de ataque y evaluando el comportamiento de la plataforma Moodle ante diversos escenarios de amenaza.

  - **Sistema de Puntuación de Vulnerabilidades Comunes (CVSS)**

    El Sistema de Puntuación de Vulnerabilidades Comunes (CVSS) es un marco de trabajo abierto y estandarizado para la comunicación de características y severidad de las vulnerabilidades de software (FIRST, 2021). En esta investigación, se utilizó CVSS como una herramienta clave para evaluar y priorizar las vulnerabilidades identificadas en la plataforma Moodle.

    - Cálculo de la puntuación CVSS: Para cada vulnerabilidad encontrada, se calcularon las puntuaciones CVSS base, temporal y ambiental utilizando la calculadora de vectores CVSS (FIRST, 2021). Estas puntuaciones reflejaron la severidad inherente de la vulnerabilidad, su actualidad y el impacto potencial en el entorno específico de TRASCEND-IT.
    - Priorización basada en la puntuación CVSS: Las vulnerabilidades se priorizaron en función de sus puntuaciones CVSS. Aquellas con puntuaciones más altas serán consideradas de mayor riesgo y recibieron atención prioritaria para su mitigación.
    - Análisis de impacto: Además de la puntuación CVSS, se llevo a cabo un análisis cualitativo del impacto potencial de cada vulnerabilidad en la confidencialidad, integridad y disponibilidad de la información sensible manejada por la plataforma Moodle (National Institute of Standards and Technology [NIST], 2018).
    - Documentación y comunicación: Las puntuaciones CVSS y los análisis de impacto se documentaron detalladamente y se comunicó de manera efectiva al equipo de desarrollo y los interesados clave para facilitar la toma de decisiones y la implementación de medidas correctivas.

En la *Tabla 4* se presentan las diferentes métricas CVSS que permiten cuantificar el impacto y la severidad de las vulnerabilidades

Tabla 4

Matriz de métricas CVSS

Métrica CVSS	Descripción
Vector de Acceso (AV)	Refleja el contexto de acceso requerido para explotar la vulnerabilidad.
Vector de Autenticación (AC)	Indica si se requiere autenticación para explotar la vulnerabilidad.
Vector de Confidencialidad (C)	Mide el impacto en la confidencialidad de los datos.
Vector de Integridad (I)	Mide el impacto en la integridad de los datos.
Vector de Disponibilidad (A)	Mide el impacto en la disponibilidad del sistema.
Vector de Privilegios Requeridos (PR)	Indica el nivel de privilegios necesarios para explotar la vulnerabilidad.
Vector de Interacción Requerida (UI)	Determina si se requiere interacción del usuario para explotar la vulnerabilidad.
Vector de Alcance (S)	Evalúa si la vulnerabilidad se limita a un componente o puede afectar al sistema completo.
Vector de Confidencialidad Obtenida (C)	Mide el impacto en la confidencialidad de los datos cuando se explota la vulnerabilidad.
Vector de Integridad Obtenida (I)	Mide el impacto en la integridad de los datos cuando se explota la vulnerabilidad.
Vector de Disponibilidad Obtenida (A)	Mide el impacto en la disponibilidad del sistema cuando se explota la vulnerabilidad.

Nota: Elaboración propia adaptada del Instituto Nacional de Ciberseguridad (INCIBE)

- **Base de Datos de Vulnerabilidades Comunes (CVE)**

  La Base de Datos de Vulnerabilidades Comunes (CVE) es una lista pública de vulnerabilidades de seguridad conocidas y estandarizadas. En esta investigación, se utilizó CVE como un recurso complementario para identificar y evaluar las vulnerabilidades presentes en la plataforma Moodle y sus componentes asociados.

  - Identificación de vulnerabilidades conocidas: Se realizó una búsqueda exhaustiva en la base de datos CVE para identificar las vulnerabilidades conocidas asociadas con la plataforma Moodle, sus componentes y las tecnologías utilizadas en el entorno Docker Swarm.

- Análisis de relevancia: Se evaluó la relevancia de cada vulnerabilidad identificada en CVE para el contexto específico de la implementación de TRASCEND-IT, considerando factores como las versiones de software utilizadas, las configuraciones y los requisitos de seguridad particulares.
- Verificación y pruebas: Las vulnerabilidades relevantes se verificaron y probaron en el entorno de despliegue de la plataforma Moodle en Docker Swarm para confirmar su presencia y evaluar su impacto real.
- Integración con CVSS: Las vulnerabilidades confirmadas a través de CVE se integraron en el proceso de priorización y análisis de impacto basado en CVSS, asegurando que se aborden adecuadamente junto con las vulnerabilidades identificadas mediante otros métodos de análisis.

En la *Tabla 5* se presenta la matriz CVE la cual facilita el seguimiento y la mitigación de las vulnerabilidades conocidas y reportadas públicamente.

Tabla 5

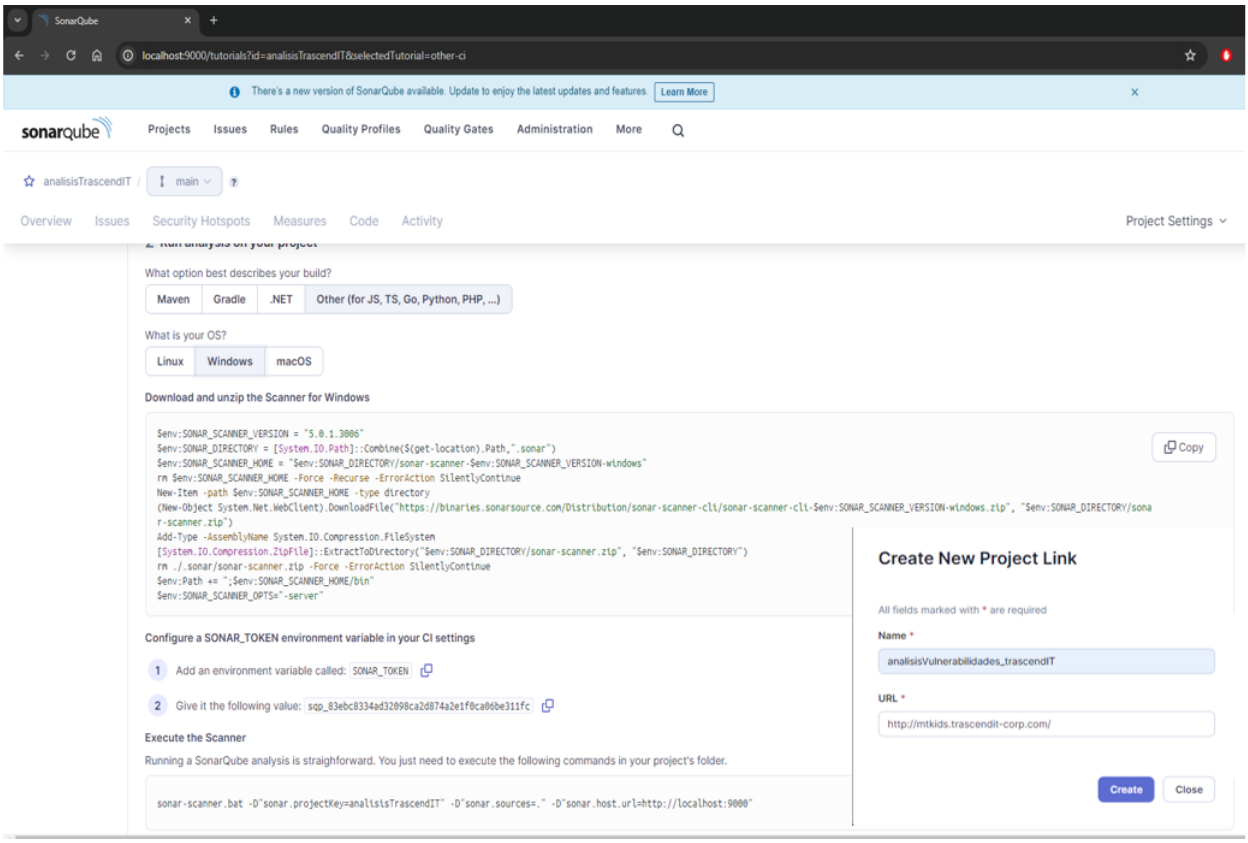
Matriz de vulnerabilidades públicas e identificadas por el CVE

Campo	Descripción
ID CVE	Identificador único asignado a la vulnerabilidad en la base de datos CVE.
Componente Afectado	Componente de software o hardware afectado por la vulnerabilidad.
Descripción	Detalles de la vulnerabilidad, incluyendo su naturaleza, impacto y posibles vectores de ataque.
Severidad	Clasificación de la severidad de la vulnerabilidad, como Crítica, Alta, Media o Baja.
Puntuación CVSS	Puntuación CVSS asignada a la vulnerabilidad, reflejando su impacto y severidad.
Estado	Estado actual de la vulnerabilidad, como Sin Resolver, Con Parche Disponible o Mitigada.
Mitigación	Recomendaciones o pasos para mitigar o resolver la vulnerabilidad.
Referencias	Enlaces a recursos adicionales, informes de seguridad o fuentes de información relevantes.

Nota: Elaboración propia adaptada de Base de Datos de Vulnerabilidades Comunes (CVE)

La combinación de CVSS y CVE en esta investigación permitió una evaluación completa y estandarizada de las vulnerabilidades presentes en la plataforma Moodle desplegada en Docker Swarm, facilitando la priorización, el análisis de impacto y la comunicación efectiva de los riesgos de seguridad identificados.

Ilustración 2 Escaneo automático mediante herramienta SonarQube



Fuente: Alejandro Paúl Amaya

En la *Ilustración 2* se presenta el proceso de ejecución del escaneo de seguridad de la plataforma Moodle bajo los parámetros y estándares predeterminados en la cual se ingresó la dirección electrónica de la plataforma http://mtkids.trascendit-corp.com. Posteriormente se hizo clic en el botón "Create" para empezar con el escaneo.

- **Registro y documentación de las vulnerabilidades encontradas:**

  Todas las vulnerabilidades y problemas de seguridad identificados durante el análisis fueron registrados y documentados de manera detallada.

  Se recopilaron detalles como la descripción de la vulnerabilidad, su ubicación en el código fuente, el impacto potencial y la severidad.

- **Priorización de los resultados obtenido**

  Se priorizaron las vulnerabilidades en función de su severidad y el impacto potencial en la confidencialidad, integridad y disponibilidad de la información sensible manejada por Moodle.

  Se creó una matriz de riesgos que combine la frecuencia de ocurrencia y el impacto potencial de cada vulnerabilidad, lo que permitió priorizar las acciones correctivas y brindar información de las consecuencias y propuestas de recomendaciones para su mitigación.

CAPÍTULO III
RESULTADOS

En el presente capítulo, se exponen los resultados obtenidos del análisis de seguridad realizado a la plataforma Moodle desplegada en un entorno Docker Swarm por la empresa TRASCEND-IT. Este análisis fue llevado a cabo siguiendo una metodología rigurosa basada en el uso de la herramienta SonarQube, complementada con pruebas de penetración, consultas a la Base de Datos de Vulnerabilidades Comunes (CVE) y la aplicación de estándares ampliamente reconocidos, como el Sistema de Puntuación de Vulnerabilidades Comunes (CVSS).

En los hallazgos presentados se detallan las vulnerabilidades identificadas, su impacto potencial y severidad en la cual se proporcionó matrices y tablas que resumen los resultados de manera estructurada, facilitando la comprensión y la priorización de los riesgos de seguridad encontrados. Estos resultados sientan las bases para la implementación de acciones correctivas y mejoras en la plataforma Moodle, con el objetivo de fortalecer la seguridad de la plataforma Moodle y proteger la información sensible manejada por esta aplicación crítica de aprendizaje en línea.

3. Clasificación de vulnerabilidades encontradas

Ilustración 3 Matriz de vulnerabilidades basado en tipo de alerta y riesgo

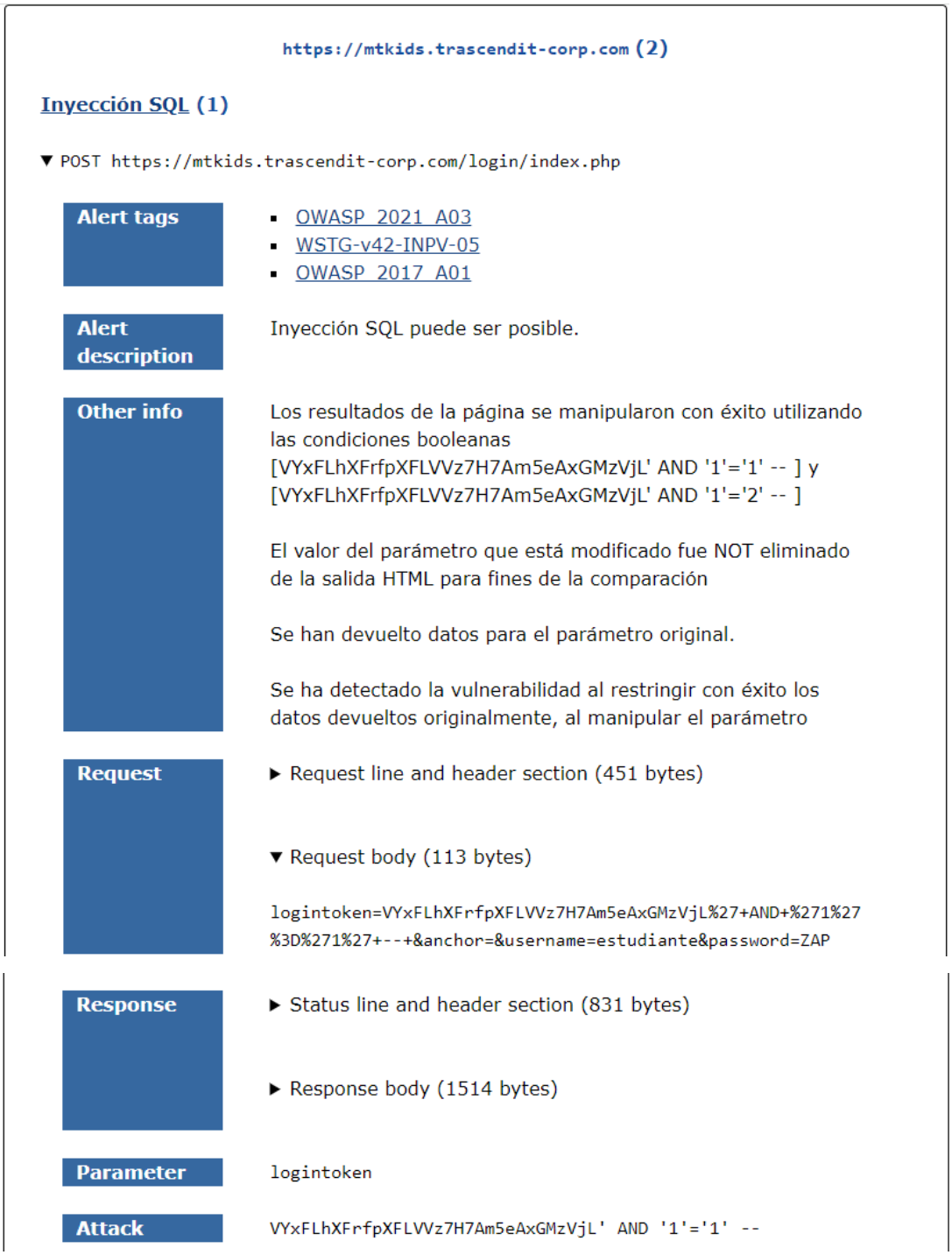
Alert type	Risk	Count
Inyección SQL	Alto	5 (16,1 %)
Inyección SQL - Oracle - Time Based	Alto	1 (3,2 %)
Ausencia de Ttokens Anti-CSRF	Medio	755 (2.435,5 %)
CSP: Wildcard Directive	Medio	68 (219,4 %)
CSP: script-src unsafe-eval	Medio	68 (219,4 %)
CSP: script-src unsafe-inline	Medio	68 (219,4 %)
CSP: style-src unsafe-inline	Medio	68 (219,4 %)
Cabecera Content Security Policy (CSP) no configurada	Medio	30 (96,8 %)
Configuración Incorrecta Cross-Domain	Medio	6 (19,4 %)
Directory Browsing (Exploración de directorios)	Medio	2 (6,5 %)
Falta de cabecera Anti-Clickjacking	Medio	1 (3,2 %)
Filtrado de información en .htaccess	Medio	4 (12,9 %)

Hidden File Found (Archivo Oculto Encontrado)	Medio	1 (3,2 %)
Inyección XSLT	Medio	6 (19,4 %)
Librería JS Vulnerable	Medio	6 (19,4 %)
Cookie No HttpOnly Flag	Bajo	15 (48,4 %)
Cookie con el atributo SameSite a None	Bajo	14 (45,2 %)
Cookie sin el atributo SameSite	Bajo	1 (3,2 %)
Divulgación de la marca de hora - Unix	Bajo	108 (348,4 %)
Gran redirección detectada (posible fuga de información confidencial)	Bajo	37 (119,4 %)
Server Leaks Version Information via "Server" HTTP Response Header Field	Bajo	148 (477,4 %)

Fuente: SONARQUBE

3.1.Vulnerabilidades de riesgo ALTO

Ilustración 4 Vulnerabilidad tipo "Inyección SQL"



https://mtkids.trascendit-corp.com (2)

Inyección SQL (1)

▼ POST https://mtkids.trascendit-corp.com/login/index.php

Alert tags

- OWASP_2021_A03
- WSTG-v42-INPV-05
- OWASP_2017_A01

Alert description

Inyección SQL puede ser posible.

Other info

Los resultados de la página se manipularon con éxito utilizando las condiciones booleanas [VYxFLhXFrfpXFLVVz7H7Am5eAxGMzVjL' AND '1'='1' --] y [VYxFLhXFrfpXFLVVz7H7Am5eAxGMzVjL' AND '1'='2' --]

El valor del parámetro que está modificado fue NOT eliminado de la salida HTML para fines de la comparación

Se han devuelto datos para el parámetro original.

Se ha detectado la vulnerabilidad al restringir con éxito los datos devueltos originalmente, al manipular el parámetro

Request

▶ Request line and header section (451 bytes)

▼ Request body (113 bytes)

```
logintoken=VYxFLhXFrfpXFLVVz7H7Am5eAxGMzVjL%27+AND+%271%27
%3D%271%27+--+&anchor=&username=estudiante&password=ZAP
```

Response

▶ Status line and header section (831 bytes)

▶ Response body (1514 bytes)

Parameter

logintoken

Attack

VYxFLhXFrfpXFLVVz7H7Am5eAxGMzVjL' AND '1'='1' --

Fuente: SONARQUBE

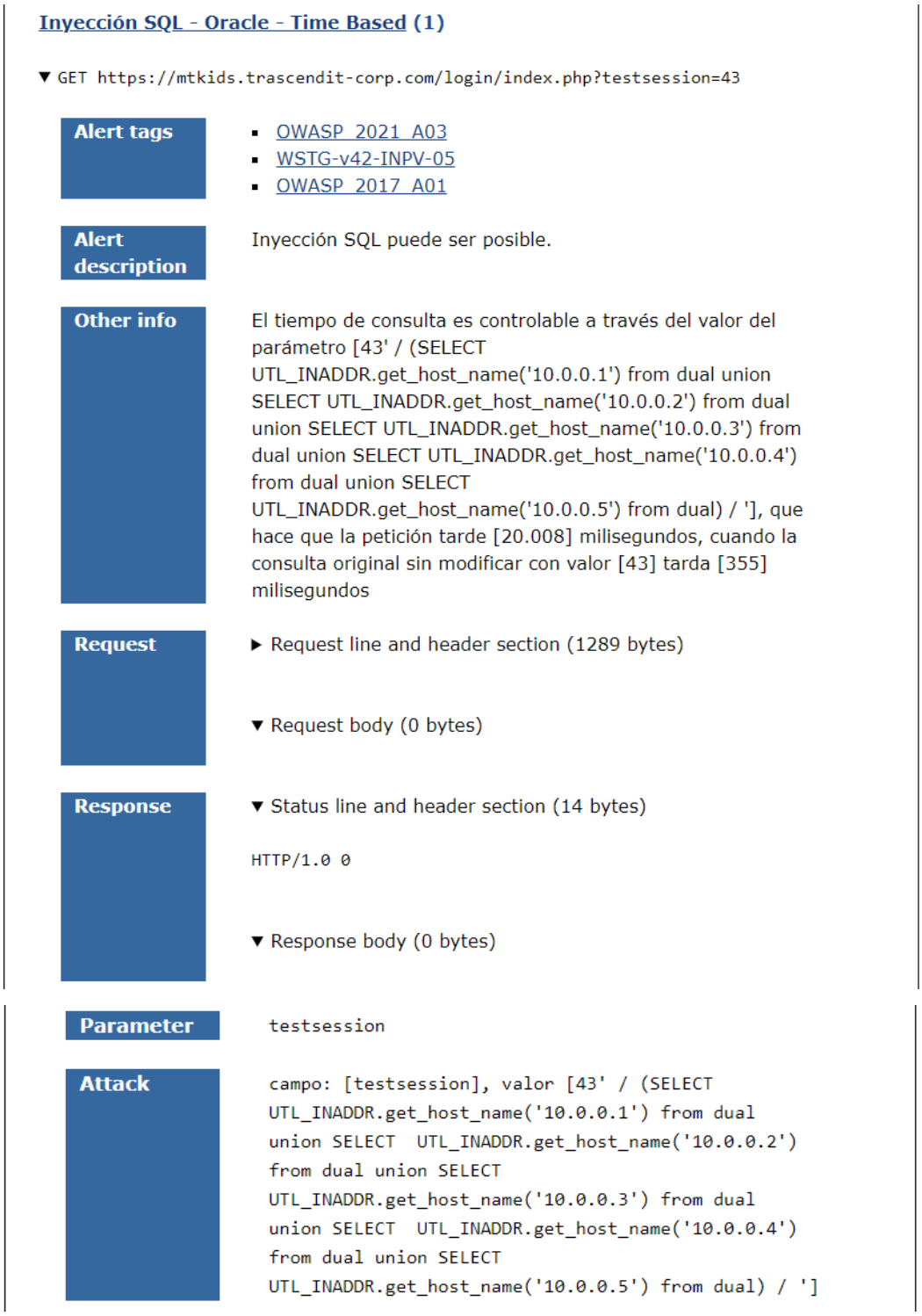
Explicación

La vulnerabilidad de inyección SQL identificada en la plataforma Moodle desplegada en el entorno Docker Swarm representa un riesgo significativo para la seguridad del sistema y la protección de la información sensible manejada por la aplicación. Esta vulnerabilidad permite a un atacante inyectar código malicioso en las consultas SQL enviadas a la base de datos, lo que podría resultar en la revelación, modificación o eliminación no autorizada de datos confidenciales, como información de usuarios, calificaciones y contenido de cursos.

Impacto

El impacto potencial de esta vulnerabilidad es alto, ya que compromete la integridad y confidencialidad de los datos manejados por Moodle. Un atacante podría explotar esta vulnerabilidad para obtener acceso no autorizado a información privilegiada, como credenciales de acceso, datos personales de estudiantes y profesores, o incluso comprometer la disponibilidad de la plataforma al realizar operaciones destructivas en la base de datos. Además, la explotación exitosa de esta vulnerabilidad podría ser utilizada como un punto de entrada para ataques más sofisticados, como la propagación de malware o el acceso no autorizado a otros sistemas dentro de la infraestructura Docker Swarm.

Ilustración 5 Vulnerabilidad tipo "Inyección SQL - Oracle - Time Based"



Inyección SQL - Oracle - Time Based (1)

▼ GET https://mtkids.trascendit-corp.com/login/index.php?testsession=43

Alert tags

- OWASP_2021_A03
- WSTG-v42-INPV-05
- OWASP_2017_A01

Alert description

Inyección SQL puede ser posible.

Other info

El tiempo de consulta es controlable a través del valor del parámetro [43' / (SELECT UTL_INADDR.get_host_name('10.0.0.1') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.2') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.3') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.4') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.5') from dual) / '], que hace que la petición tarde [20.008] milisegundos, cuando la consulta original sin modificar con valor [43] tarda [355] milisegundos

Request

▶ Request line and header section (1289 bytes)

▼ Request body (0 bytes)

Response

▼ Status line and header section (14 bytes)

```
HTTP/1.0 0
```

▼ Response body (0 bytes)

Parameter

testsession

Attack

```
campo: [testsession], valor [43' / (SELECT
UTL_INADDR.get_host_name('10.0.0.1') from dual
union SELECT  UTL_INADDR.get_host_name('10.0.0.2')
from dual union SELECT
UTL_INADDR.get_host_name('10.0.0.3') from dual
union SELECT  UTL_INADDR.get_host_name('10.0.0.4')
from dual union SELECT
UTL_INADDR.get_host_name('10.0.0.5') from dual) / ']
```

Fuente: SONARQUBE

Explicación

Esta técnica explota la capacidad del atacante para inyectar una consulta SQL maliciosa que provoca un retraso en la respuesta del servidor, permitiendo así inferir si la inyección fue exitosa basándose en el tiempo de respuesta. En este caso, se utilizó la función **"UTL_INADDR.get_host_name"** para ejecutar varias consultas de red, lo cual podría llevar a la exposición de información crítica del servidor.

Impacto

El impacto de esta vulnerabilidad es significativo ya que una explotación exitosa puede permitir a un atacante acceder a información sensible, manipular la base de datos, y potencialmente comprometer la integridad y confidencialidad de los datos almacenados. Además, la posibilidad de ejecutar comandos arbitrarios a nivel de base de datos podría conducir a una escalación de privilegios y control total del sistema, poniendo en riesgo la seguridad y disponibilidad de las plataformas Moodle desplegadas.

En la siguiente *Tabla 6* se presenta la propuesta de solución a las vulnerabilidades encontradas tipo "Inyección SQL e Inyección SQL – Oracle – Time Based" en la plataforma Moodle en la cual se detalla a continuación:

Tabla 6

Matriz de propuesta de solución a las vulnerabilidades de tipo Inyección SQL

SOLUCIÓN	Se recomienda no confiar en los datos de entrada del lado del cliente, incluso si existe una validación del lado del cliente. Como norma general, se debe escribir la verificación de los datos en el lado del servidor. Si la aplicación utiliza JDBC, se recomienda usar PreparedStatement o CallableStatement, con parámetros pasados por '?'. Si la aplicación usa ASP, se sugiere utilizar objetos de comando ADO con verificación de tipo fuerte y consultas parametrizadas. En caso de poder utilizar los procedimientos almacenados de la base de datos, se recomienda hacer uso de ellos. Sin embargo, se debe evitar concatenar cadenas en consultas en el procedimiento almacenado o usar 'exec', 'exex immediate' o una función equivalente.

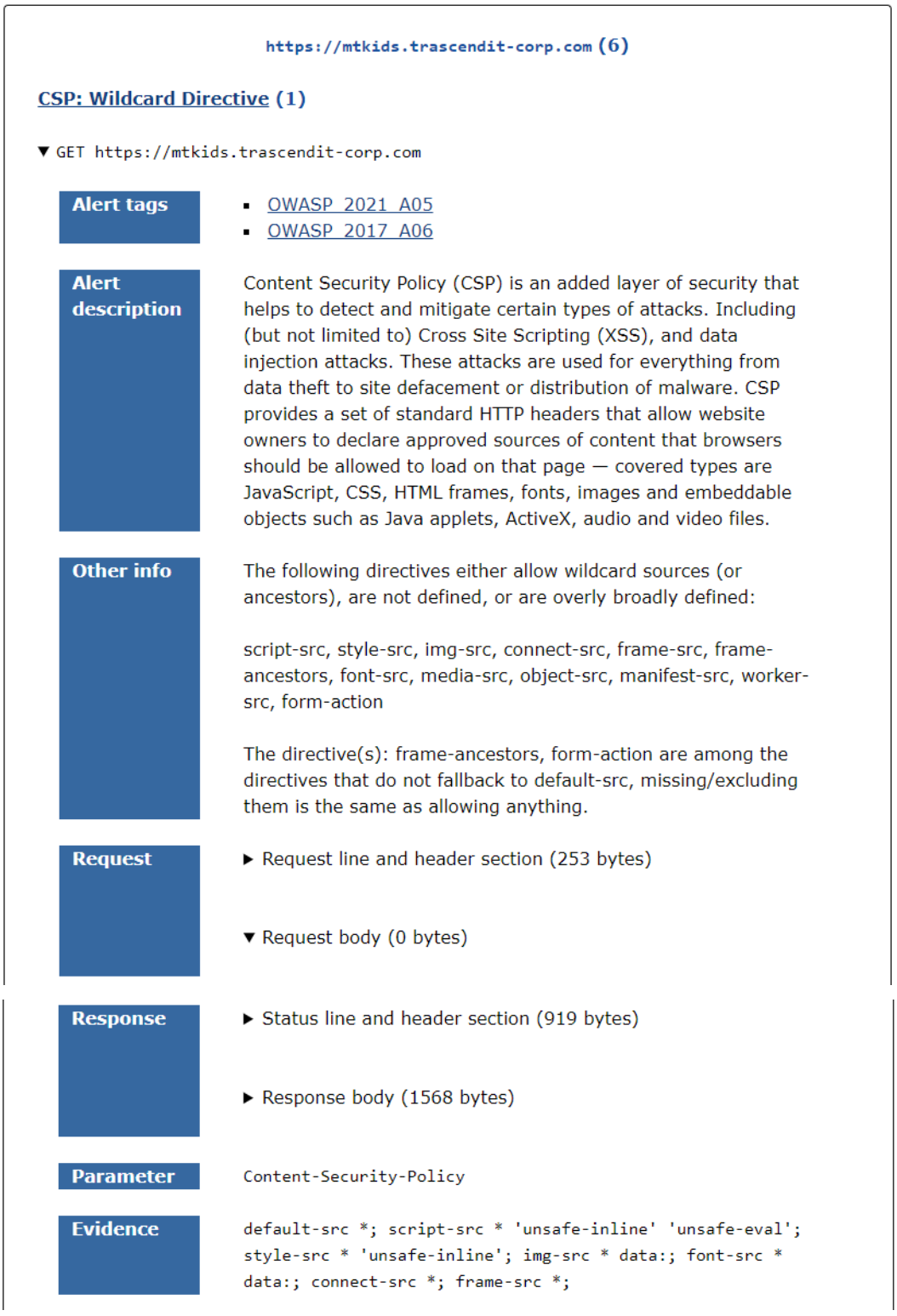
	No se deben crear consultas SQL dinámicas mediante la concatenación de cadenas simples. En su lugar, se recomienda aplicar una 'lista de permitidos' para caracteres permitidos o una 'lista de denegados' para caracteres no permitidos en la entrada del usuario. Además, se sugiere aplicar el principio de privilegio mínimo utilizando el usuario de base de datos con el menor privilegio posible. En particular, se debe evitar utilizar usuarios de bases de datos 'sa' o 'db-owner'. Si bien esto no elimina la inyección SQL, minimiza su impacto.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

3.2.Vulnerabilidades de riesgo MEDIO

3.2.1. Vulnerabilidades de tipo “CSP”

Ilustración 6Vulnerabilidad tipo "CSP: Wildcard Directive"



https://mtkids.trascendit-corp.com (6)

CSP: Wildcard Directive (1)

▼ GET https://mtkids.trascendit-corp.com

Alert tags

- OWASP_2021_A05
- OWASP_2017_A06

Alert description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Other info

The following directives either allow wildcard sources (or ancestors), are not defined, or are overly broadly defined:

script-src, style-src, img-src, connect-src, frame-src, frame-ancestors, font-src, media-src, object-src, manifest-src, worker-src, form-action

The directive(s): frame-ancestors, form-action are among the directives that do not fallback to default-src, missing/excluding them is the same as allowing anything.

Request

▶ Request line and header section (253 bytes)

▼ Request body (0 bytes)

Response

▶ Status line and header section (919 bytes)

▶ Response body (1568 bytes)

Parameter

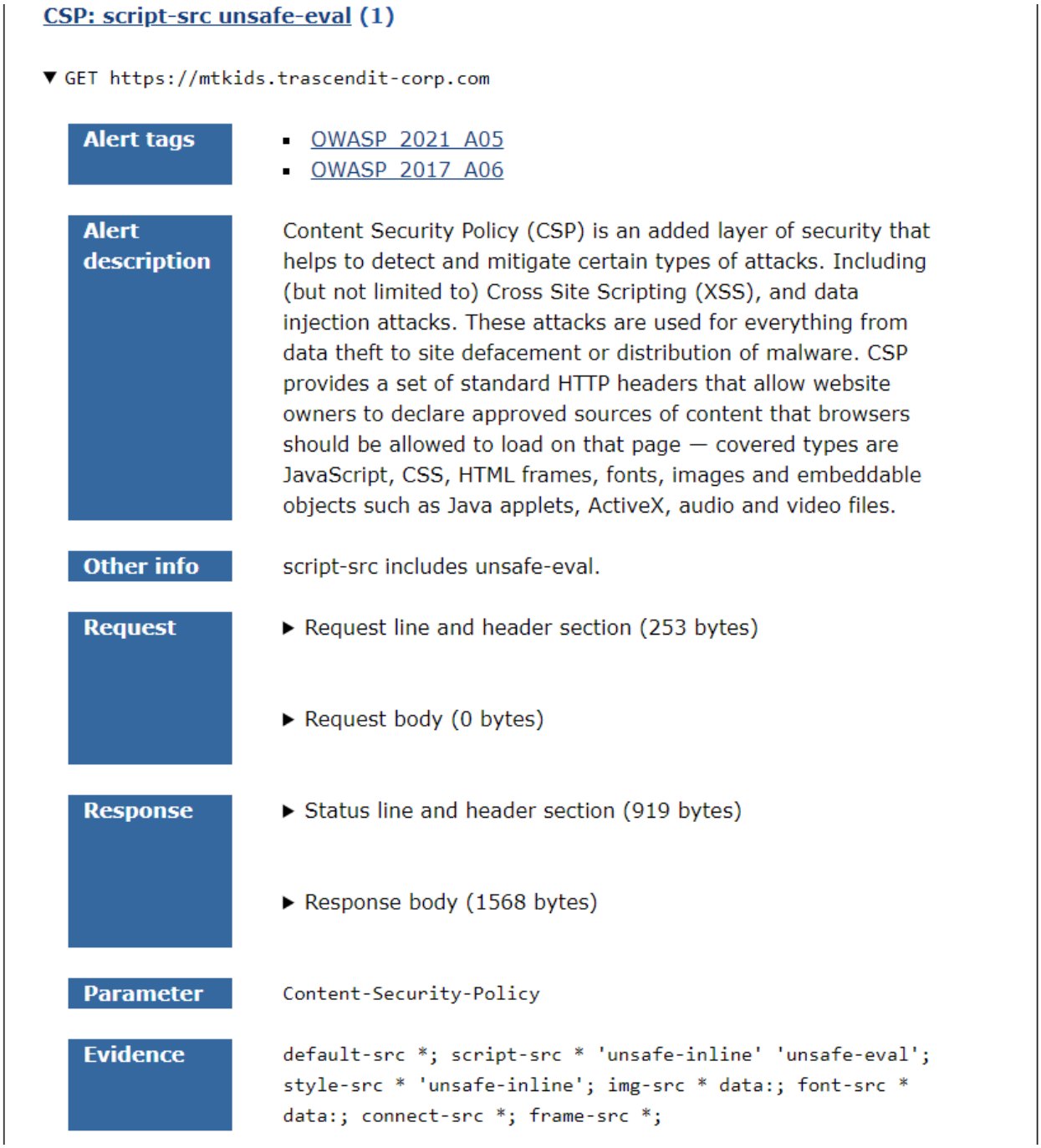
Content-Security-Policy

Evidence

```
default-src *; script-src * 'unsafe-inline' 'unsafe-eval';
style-src * 'unsafe-inline'; img-src * data:; font-src *
data:; connect-src *; frame-src *;
```

Fuente: SONARQUBE

Ilustración 7Vulnerabilidad tipo "CSP: script-src unsafe-eval"



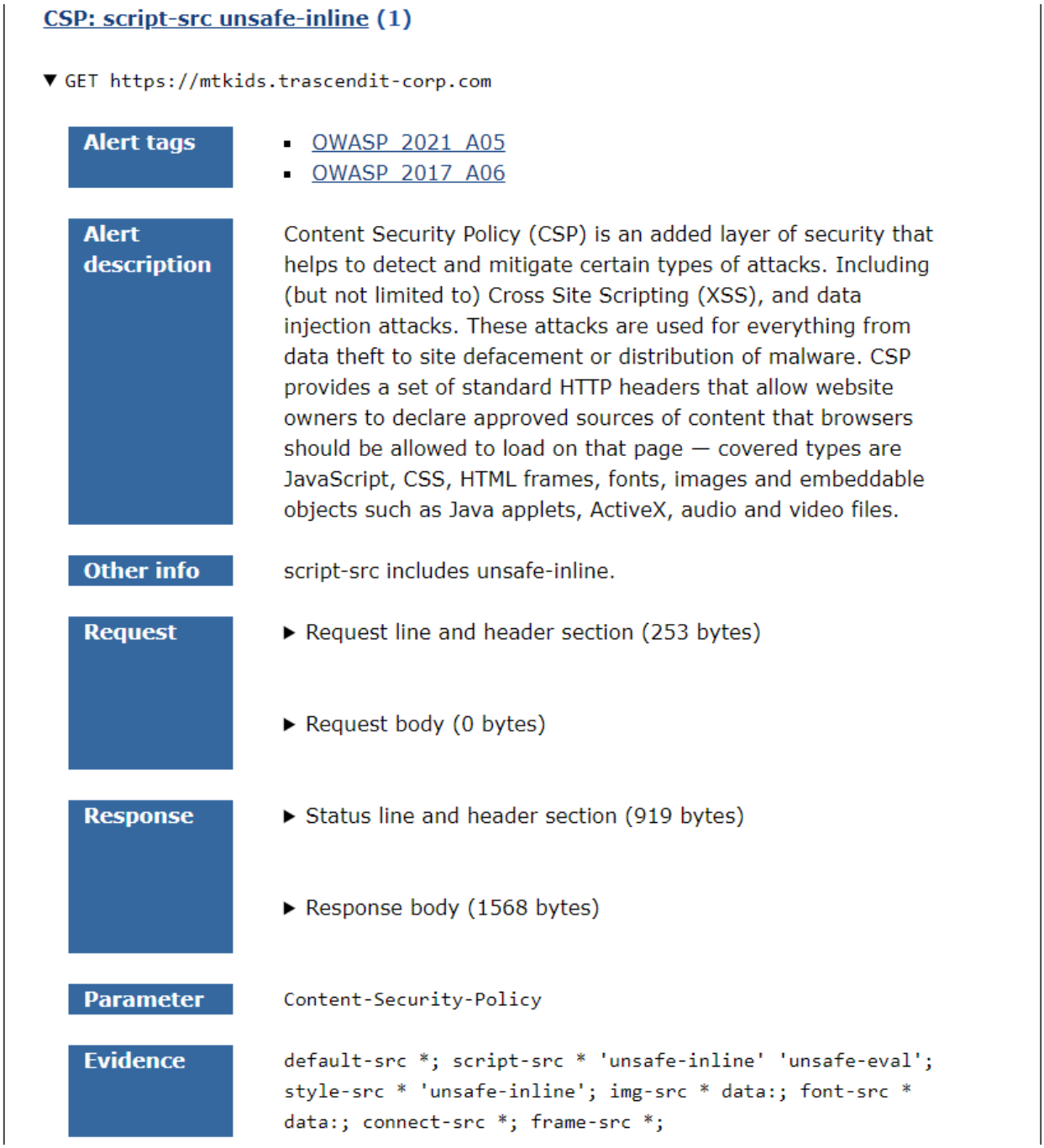
CSP: script-src unsafe-eval (1)

▼ GET https://mtkids.trascendit-corp.com

Alert tags	▪ OWASP_2021_A05 ▪ OWASP_2017_A06
Alert description	Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.
Other info	script-src includes unsafe-eval.
Request	▶ Request line and header section (253 bytes) ▶ Request body (0 bytes)
Response	▶ Status line and header section (919 bytes) ▶ Response body (1568 bytes)
Parameter	`Content-Security-Policy`
Evidence	`default-src *; script-src * 'unsafe-inline' 'unsafe-eval'; style-src * 'unsafe-inline'; img-src * data:; font-src * data:; connect-src *; frame-src *;`

Fuente: SONARQUBE

Ilustración 8 Vulnerabilidad tipo "CSP: script-src unsafe-inline"



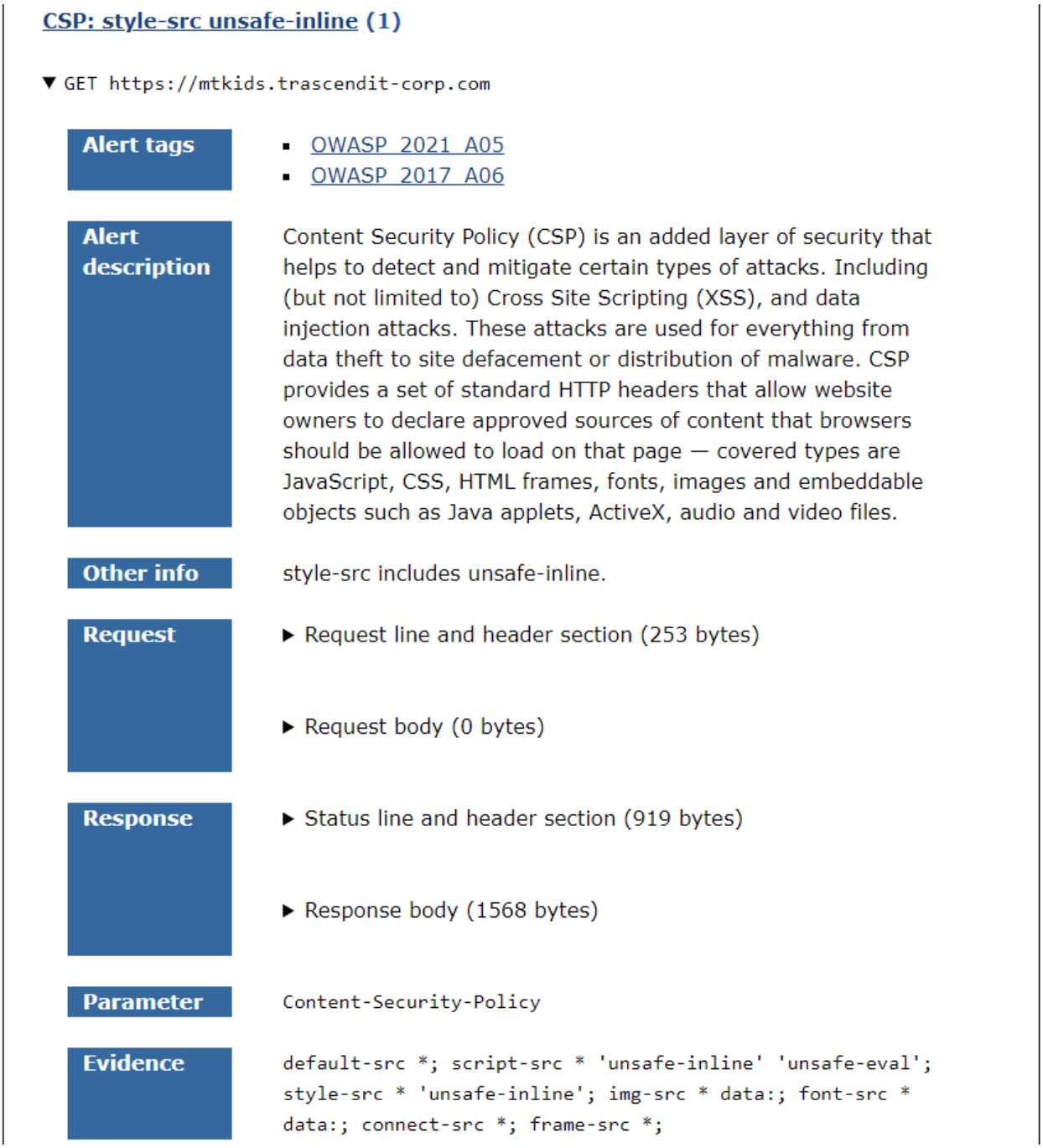
CSP: script-src unsafe-inline (1)

▼ GET https://mtkids.trascendit-corp.com

Alert tags	▪ OWASP_2021_A05 ▪ OWASP_2017_A06
Alert description	Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.
Other info	script-src includes unsafe-inline.
Request	▶ Request line and header section (253 bytes) ▶ Request body (0 bytes)
Response	▶ Status line and header section (919 bytes) ▶ Response body (1568 bytes)
Parameter	`Content-Security-Policy`
Evidence	`default-src *; script-src * 'unsafe-inline' 'unsafe-eval'; style-src * 'unsafe-inline'; img-src * data:; font-src * data:; connect-src *; frame-src *;`

Fuente: SONARQUBE

Ilustración 9 Vulnerabilidad tipo "CSP: style-src unsafe-inline"



CSP: style-src unsafe-inline (1)

▼ GET https://mtkids.trascendit-corp.com

Alert tags	▪ OWASP_2021_A05 ▪ OWASP_2017_A06
Alert description	Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.
Other info	style-src includes unsafe-inline.
Request	▶ Request line and header section (253 bytes) ▶ Request body (0 bytes)
Response	▶ Status line and header section (919 bytes) ▶ Response body (1568 bytes)
Parameter	`Content-Security-Policy`
Evidence	`default-src *; script-src * 'unsafe-inline' 'unsafe-eval'; style-src * 'unsafe-inline'; img-src * data:; font-src * data:; connect-src *; frame-src *;`

Fuente: SONARQUBE

Ilustración 10 Vulnerabilidad tipo "Cabecera Content Security Policy (CSP) no configurada"

Fuente: SONARQUBE

Explicación

La Política de seguridad de contenido (CSP) es una capa adicional de seguridad que ayuda a detectar y mitigar ciertos tipos de ataques mediante el uso de comodines (*) en la política de seguridad de contenido (CSP) de una aplicación web incluyendo (pero no limitado a) Cross Site Scripting (XSS) y ataques de inyección de datos. Estos ataques se utilizan para todo, desde el robo de datos hasta la destrucción de sitios o la distribución de malware. CSP proporciona un conjunto de encabezados HTTP estándar que permiten a los propietarios de sitios web declarar fuentes de contenido aprobadas que los navegadores deben poder cargar en esa página; los tipos cubiertos son JavaScript, CSS, marcos HTML, fuentes, imágenes y objetos integrables como subprogramas de Java. Archivos ActiveX, audio y vídeo.

Impacto

La presencia de directivas comodín puede permitir a atacantes inyectar scripts maliciosos o cargar recursos no confiables, comprometiendo la seguridad e integridad de las plataformas Moodle. Esto puede resultar en la ejecución de código arbitrario en el navegador del usuario, robo de datos sensibles, suplantación de identidad, y otras amenazas.

En la siguiente *Tabla 7* se presenta la propuesta de solución a las vulnerabilidades encontradas tipo "CSP" en la plataforma Moodle en la cual se detalla a continuación:

Tabla 7

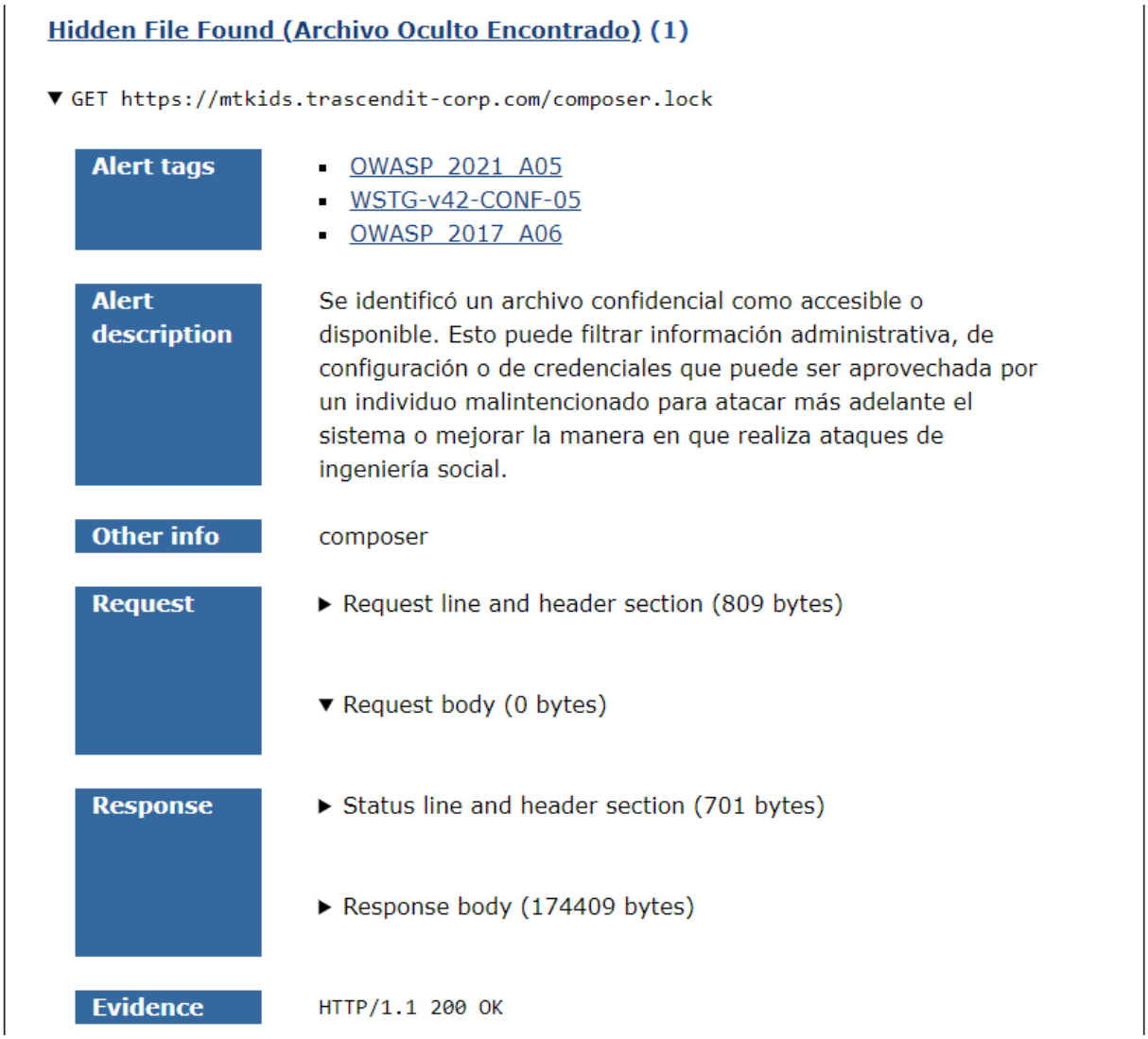
Matriz de propuesta de solución a las vulnerabilidades de tipo CSP

SOLUCIÓN	Asegúrese de que su servidor web, servidor de aplicaciones, balanceador de carga, etc. esté configurado para establecer la cabecera Content-Security-Policy en la cual se deben definir políticas CSP específicas, evitando el uso de comodines y reemplazándolos con orígenes confiables. Se debe configurar la política CSP para incluir sólo los orígenes necesarios en directivas como default-src *; script-src * 'unsafe-inline' 'unsafe-eval'; style-src * 'unsafe-inline'; img-src * data:; font-src * data:; connect-src *; frame-src *;, especificando fuentes de confianza. La nueva política debe ser implementada en los encabezados HTTP o etiquetas <meta> del HTML y probada exhaustivamente para asegurar que no afecta la funcionalidad de la aplicación.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

3.2.2. Vulnerabilidades halladas de diferentes tipos.

Ilustración 11 Vulnerabilidad tipo "Hidden File Found (Archivo oculto encontrado)"



Fuente: SONARQUBE

Explicación

La vulnerabilidad "Archivo oculto encontrado" se refiere a la presencia de archivos que no están destinados a ser accesibles públicamente, pero que se pueden encontrar en el servidor web. Estos archivos pueden contener información sensible, configuraciones, o datos que pueden ser explotados por atacantes para obtener acceso no autorizado o realizar otros tipos de ataques.

Impacto

Los atacantes pueden utilizar esta información para mapear la estructura del sistema, descubrir credenciales, o explotar vulnerabilidades conocidas en el software o configuraciones desactualizadas.

En la siguiente *Tabla 8* se presenta la propuesta de solución a la vulnerabilidad encontrada de tipo "Archivo oculto encontrado" en la plataforma Moodle en la cual se detalla a continuación:

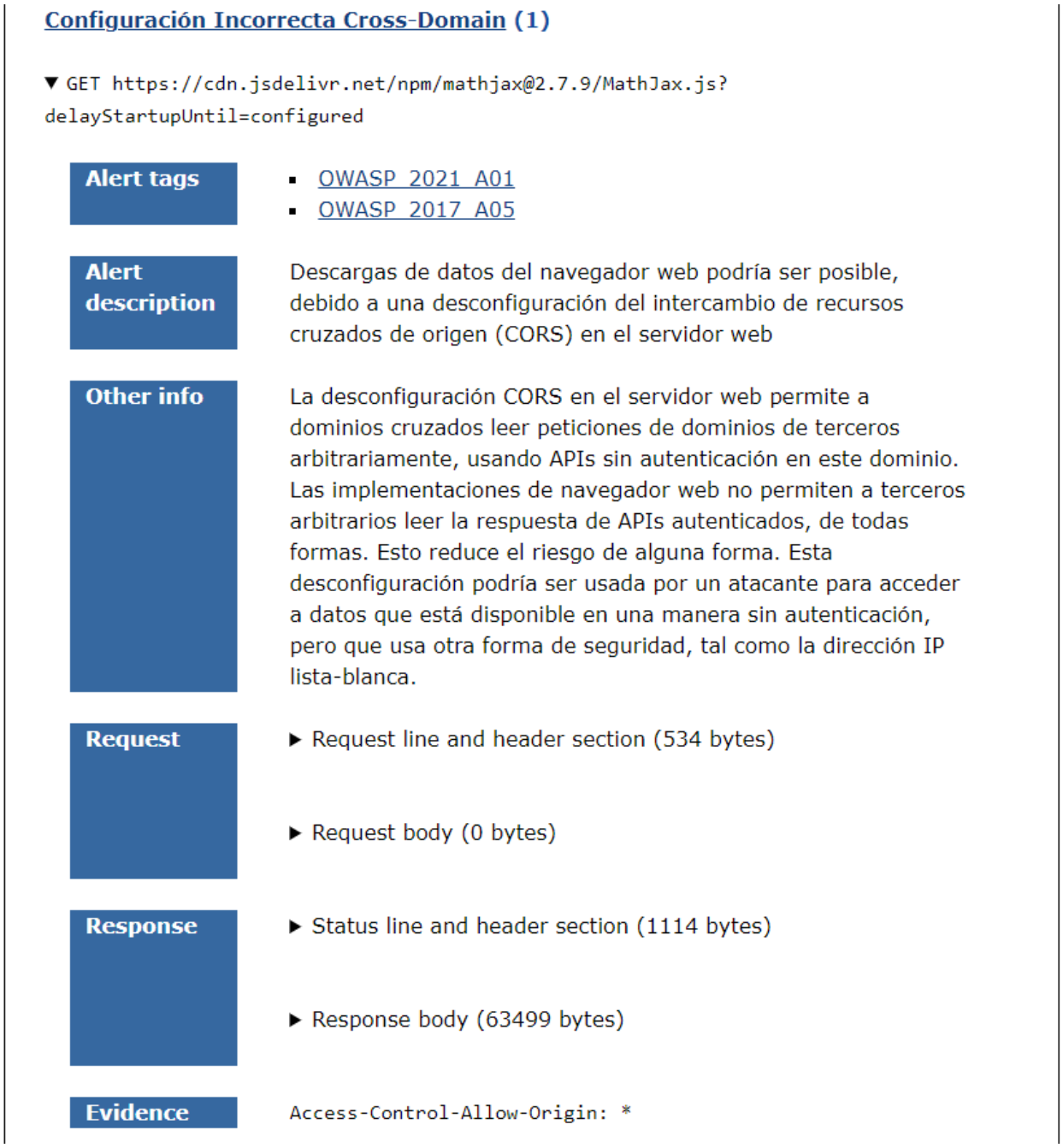
Tabla 8

Matriz de propuesta de solución a la vulnerabilidad de tipo "archivo oculto encontrado"

SOLUCIÓN	Se debe considerar si este componente es realmente necesario en producción; si no es así se requiere desactivarlo. Limitar la exposición solo a sistemas internos o IPs de origen definidas, etc. Implementar controles de acceso adecuados y configurar permisos de archivo para restringir el acceso solo a usuarios autorizados. Además, se deben utilizar herramientas automatizadas de escaneo y monitoreo que detecten la presencia de archivos ocultos y notifiquen cualquier anomalía.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

Ilustración 12 Vulnerabilidad tipo "Configuración Incorrecta Cross-Domain"



Configuración Incorrecta Cross-Domain (1)

▼ GET https://cdn.jsdelivr.net/npm/mathjax@2.7.9/MathJax.js?delayStartupUntil=configured

Alert tags	▪ OWASP_2021_A01 ▪ OWASP_2017_A05
Alert description	Descargas de datos del navegador web podría ser posible, debido a una desconfiguración del intercambio de recursos cruzados de origen (CORS) en el servidor web
Other info	La desconfiguración CORS en el servidor web permite a dominios cruzados leer peticiones de dominios de terceros arbitrariamente, usando APIs sin autenticación en este dominio. Las implementaciones de navegador web no permiten a terceros arbitrarios leer la respuesta de APIs autenticados, de todas formas. Esto reduce el riesgo de alguna forma. Esta desconfiguración podría ser usada por un atacante para acceder a datos que está disponible en una manera sin autenticación, pero que usa otra forma de seguridad, tal como la dirección IP lista-blanca.
Request	▶ Request line and header section (534 bytes) ▶ Request body (0 bytes)
Response	▶ Status line and header section (1114 bytes) ▶ Response body (63499 bytes)
Evidence	Access-Control-Allow-Origin: *

Fuente: SONARQUBE

Explicación

Si una aplicación web no configura adecuadamente las políticas de CORS, puede permitir que otros dominios y orígenes accedan a recursos sensibles de la aplicación. Esto puede conducir a ataques como el robo de datos, la inyección de scripts maliciosos (XSS) y otros tipos de ataques de origen cruzado.

Impacto

Los atacantes podrían acceder a recursos y datos sensibles de los usuarios, como información de inicio de sesión, calificaciones, contenido de cursos y más. Además, podrían inyectar scripts maliciosos en la aplicación, lo que podría comprometer aún más la seguridad y la integridad de la plataforma.

En la siguiente *Tabla 9* se presenta la propuesta de solución a la vulnerabilidad encontrada de tipo "Configuración incorrecta Cross-Domain" en la plataforma Moodle en la cual se detalla a continuación:

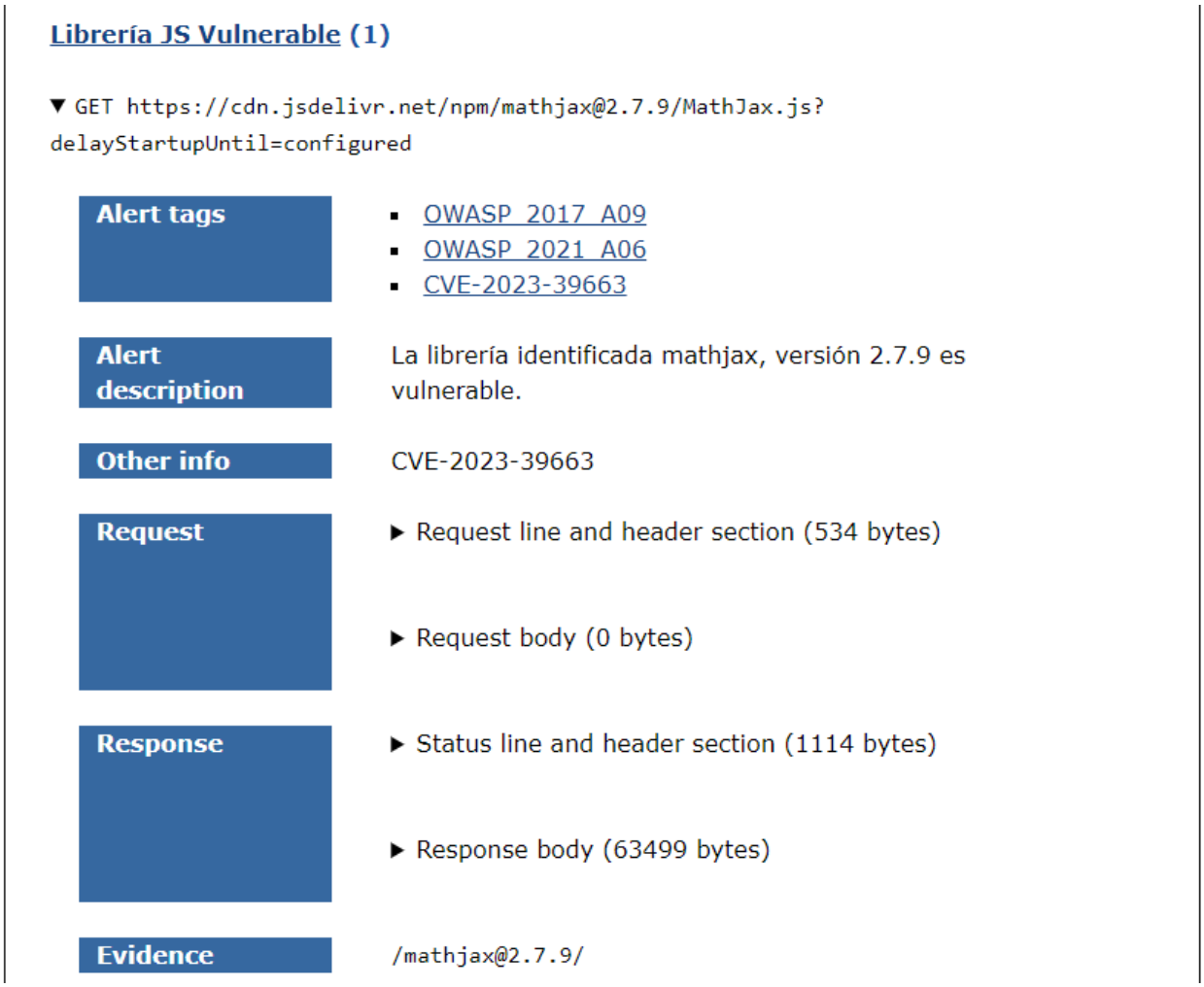
Tabla 9

Matriz de propuesta de solución a la vulnerabilidad de tipo "configuración incorrecta Cross-Domain"

SOLUCIÓN	Asegurarse que los datos sensibles no esten disponibles de manera no autenticada (usando dirección IP listado-blanco). Configurar el encabezado HTTP "Access-Control-Allow-Origin" a un conjunto de dominios más restrictivo, o remover completamente todos los encabezados CORS, para permitir que el navegador web refuerce la política de mismo origen (SOP) en una manera más restrictiva.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

Ilustración 13 Vulnerabilidad tipo "Librería JS vulnerable"



Fuente: SONARQUBE

Explicación

Las aplicaciones web modernas a menudo dependen de múltiples librerías y frameworks de terceros escritos en JavaScript. Si alguna de estas librerías contiene vulnerabilidades, como fallas de seguridad, defectos de código o vulnerabilidades conocidas, puede poner en riesgo la seguridad de toda la aplicación web que las utiliza.

Impacto

Los atacantes podrían explotar estas vulnerabilidades para obtener acceso no autorizado, robar datos sensibles, inyectar código malicioso o incluso comprometer todo el entorno de Docker Swarm en el que se despliega la plataforma.

En la siguiente *Tabla 10* se presenta la propuesta de solución a la vulnerabilidad encontrada de tipo “Librería JS vulnerable” en la plataforma Moodle en la cual se detalla a continuación:

Tabla 10

Matriz de propuesta de solución a la vulnerabilidad de tipo "Librería JS vulnerable"

SOLUCIÓN	Realizar un seguimiento constante de las librerías y dependencias utilizadas en la plataforma Moodle y mantenerlas actualizadas con los últimos parches de seguridad. Además, se recomienda utilizar herramientas de análisis de dependencias, como npm audit o snyk, para identificar automáticamente cualquier vulnerabilidad conocida en las librerías utilizadas.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

Ilustración 14 Vulnerabilidad tipo "Falta de cabecera Anti-Clickjacking"

Fuente: SONARQUBE

Explicación

Es un tipo de ataque de interfaz de usuario (UI) en el que un atacante engaña a un usuario para que haga clic en un área diferente a la que el usuario cree que está haciendo clic. Esto se logra superponiendo la página web legítima sobre otra página maliciosa de manera transparente. La falta de la cabecera X-Frame-Options permite que la aplicación web sea incrustada en un iframe, lo que facilita este tipo de ataque.

Impacto

La falta de una cabecera anti-clickjacking podría permitir que la plataforma Moodle sea víctima de ataques de clickjacking. Los atacantes podrían engañar a los usuarios para que realicen acciones no deseadas, como iniciar sesión en una página falsa, divulgar información confidencial o realizar cambios no autorizados en la plataforma.

En la siguiente *Tabla 11* se presenta la propuesta de solución a la vulnerabilidad encontrada de tipo “Falta de cabecera anti-clickjacking” en la plataforma Moodle en la cual se detalla a continuación:

Matriz de propuesta de solución a la vulnerabilidad de tipo Falta de cabecera anti-clickjacking

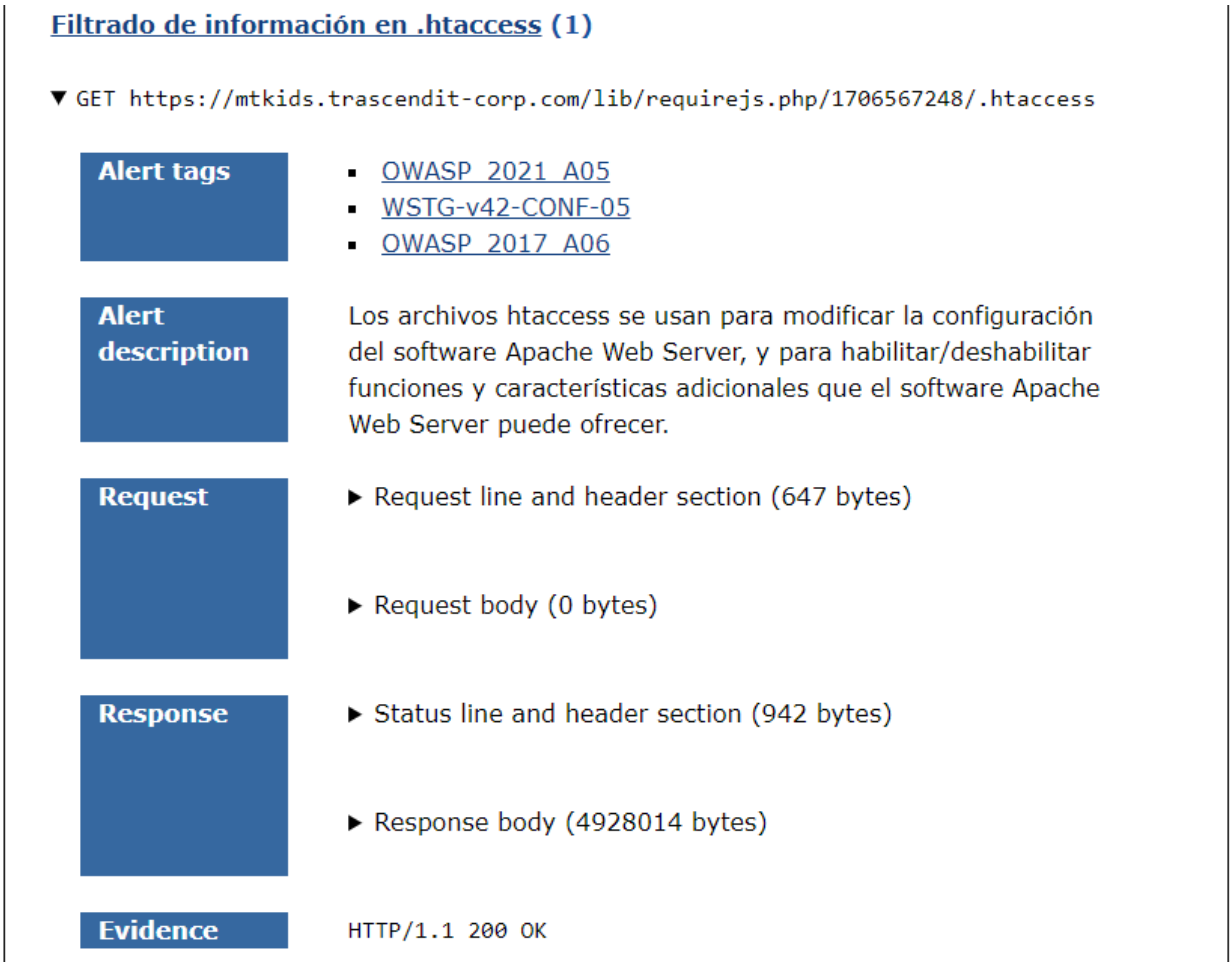
Tabla 11

Matriz de propuesta de solución a la vulnerabilidad de tipo "Falta de cabecera anti-clickjacking"

SOLUCIÓN	Es necesario implementar la cabecera X-Frame-Options en la plataforma Moodle. Esta cabecera indica al navegador web si se permite o no incrustar la página en un iframe. Las opciones son: 1. X-Frame-Options: DENY - Impide que la página se cargue en un iframe desde cualquier origen. 2. X-Frame-Options: SAMEORIGIN - Permite que la página se cargue en un iframe solo desde el mismo origen. Además de la cabecera X-Frame-Options, también se recomienda implementar otras cabeceras de seguridad relacionadas, como Content Security Policy (CSP) y X-XSS-Protection, para reforzar la protección contra ataques de clickjacking..

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

Ilustración 15 Vulnerabilidad tipo "Filtrado de información en .htaccess"



Fuente: SONARQUBE

Explicación

Esto ocurre cuando la configuración del servidor web permite el acceso a archivos o directorios que contienen información sensible la cual puede resultar en la exposición de datos críticos, como credenciales de base de datos, configuraciones de servidor, y scripts PHP.

Impacto

La exposición de información sensible puede comprometer la integridad y confidencialidad del sistema, permitiendo a atacantes potenciales explotar otras vulnerabilidades y obtener acceso no autorizado a la plataforma educativa. Esto no solo pone en riesgo los datos personales de los usuarios, sino también la estabilidad y seguridad de la infraestructura TI de Trascend-IT.

En la siguiente *Tabla 12* se presenta la propuesta de solución a la vulnerabilidad encontrada de tipo “Filtrado de información en .htaccess” en la plataforma Moodle en la cual se detalla a continuación:

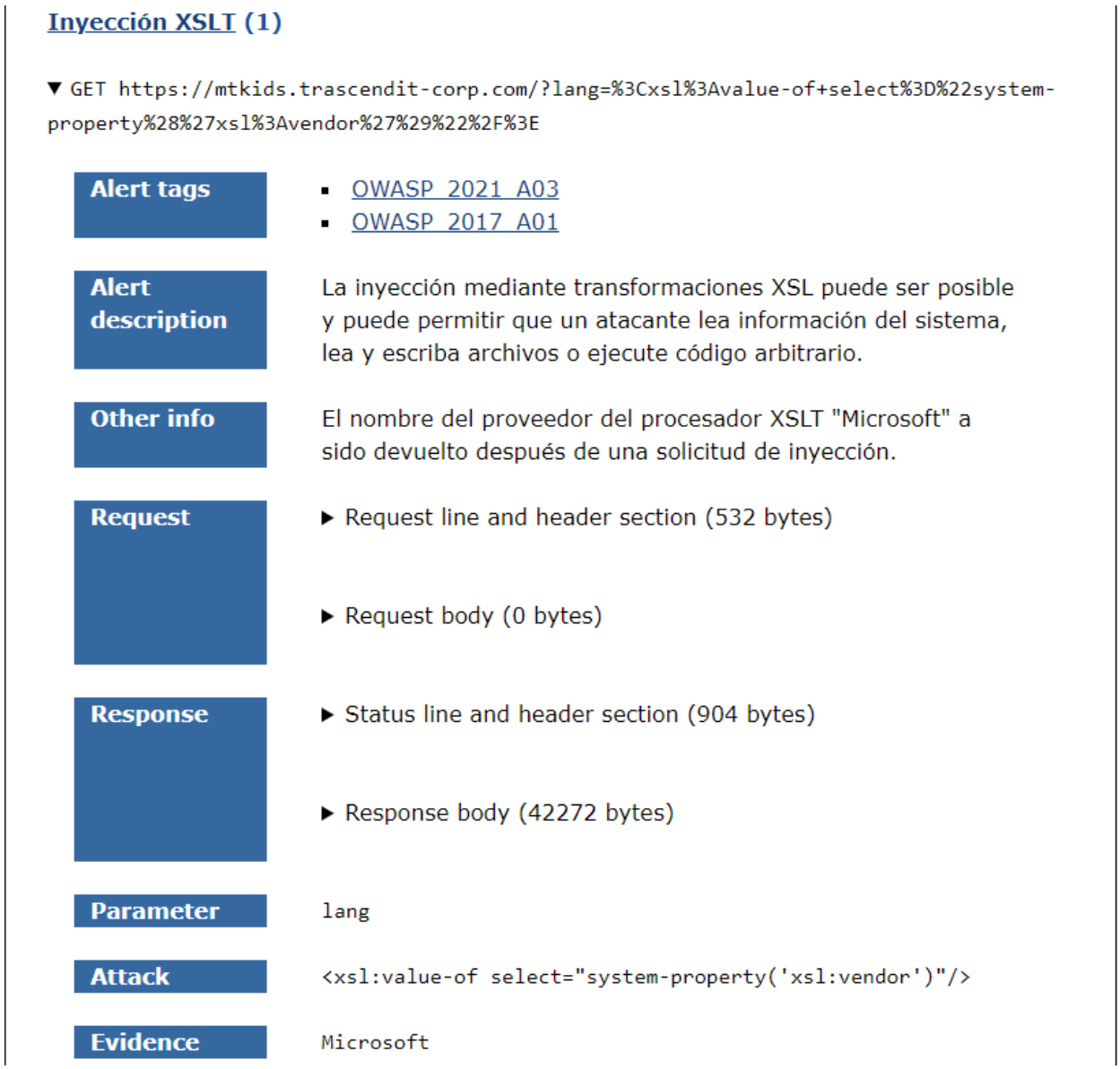
Tabla 12

Matriz de propuesta de solución a la vulnerabilidad de tipo "Filtrado de información en .htaccess"

SOLUCIÓN	Configurar adecuadamente este archivo para limitar el acceso a información sensible. Se debe establecer reglas estrictas que impidan el acceso no autorizado a directorios y archivos críticos, como archivos de configuración y scripts PHP. Además, es recomendable emplear técnicas de ocultamiento de archivos, como el uso de "Options -Indexes", para evitar listados de directorios y reforzar la seguridad general del servidor.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

Ilustración 16 Vulnerabilidad tipo "Inyección XSLT"



Inyección XSLT (1)

▼ GET https://mtkids.trascendit-corp.com/?lang=%3Cxsl%3Avalue-of+select%3D%22system-property%28%27xsl%3Avendor%27%29%22%2F%3E

Alert tags	▪ OWASP_2021_A03 ▪ OWASP_2017_A01
Alert description	La inyección mediante transformaciones XSL puede ser posible y puede permitir que un atacante lea información del sistema, lea y escriba archivos o ejecute código arbitrario.
Other info	El nombre del proveedor del procesador XSLT "Microsoft" a sido devuelto después de una solicitud de inyección.
Request	▶ Request line and header section (532 bytes) ▶ Request body (0 bytes)
Response	▶ Status line and header section (904 bytes) ▶ Response body (42272 bytes)
Parameter	lang
Attack	<xsl:value-of select="system-property('xsl:vendor')"/>
Evidence	Microsoft

Fuente:SONARQUBE

Explicación

La vulnerabilidad de inyección XSLT ocurre cuando un atacante puede insertar código XSLT malicioso en datos que serán procesados por un procesador XSLT el cual podría explotarse para ejecutar código no autorizado en el servidor, acceder a información sensible, o modificar el comportamiento del sistema. XSLT es un lenguaje para transformar documentos XML y, si no se controla adecuadamente, puede ser utilizado para ejecutar comandos o scripts maliciosos.

Impacto

La explotación de una inyección XSLT podría permitir a un atacante ejecutar código arbitrario en el servidor, comprometiendo la integridad y disponibilidad de la plataforma Moodle. Esto podría llevar a la exposición de datos confidenciales, la alteración de información educativa, y potencialmente el control total del sistema por parte de atacantes.

En la siguiente *Tabla 13* se presenta la propuesta de solución a la vulnerabilidad encontrada de tipo "Inyección XSLT" en la plataforma Moodle en la cual se detalla a continuación:

Tabla 13

Matriz de propuesta de solución a la vulnerabilidad de tipo "Inyección XSLT".

SOLUCIÓN	Validar todos los datos de entrada que puedan ser procesados mediante transformaciones XSLT. Esto incluye implementar controles estrictos para asegurarse de que solo se procesen plantillas XSLT confiables y predefinidas, evitando así la ejecución de código malicioso. Deshabilitar las características innecesarias de XSLT que puedan permitir la ejecución de código arbitrario y asegurarse de que el entorno de ejecución esté correctamente aislado mediante contenedores seguros y actualizados.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

Ilustración 17 Vulnerabilidad tipo "Ausencia de Ttokens Anti-CSRF"

▼ GET https://mtkids.trascendit-corp.com

Alert tags

- OWASP_2021_A01
- WSTG-v42-SESS-05
- OWASP_2017_A05

Alert description

No se encontraron tokens Anti-CSRF en un formulario de envío HTML.

Una solicutud falsa entre sitios en un ataque que compromete y obliga a una víctima a enviar su solicitud HTTP a un destino objetivo sin su conocimiento o intención para poder realizar una acción como víctima. La causa oculta es la funcionalidad de la aplicación utilizando acciones de URL/formulario que pueden ser adivinados de forma repetible. La naturaleza del ataque es que CSRG explota la confianza que un sitio web proporciona a un usuario. Por el contrario, las cadenas de comandos de los sitios cruzados (XSS) explotan la confianza que un usuario proporciona en un sitio web. Al igual que XSS, los ataques CSRG no son de forma necesaria de sitios cruzados, pero hay la posibilidad de que si pueden serlo. La falsificación de las solicitudes ente los sitios también se conoce como CSRF, XSRG, ataques con un solo clic, montaje de sesión, diputado confundido y navegación en alta mar.

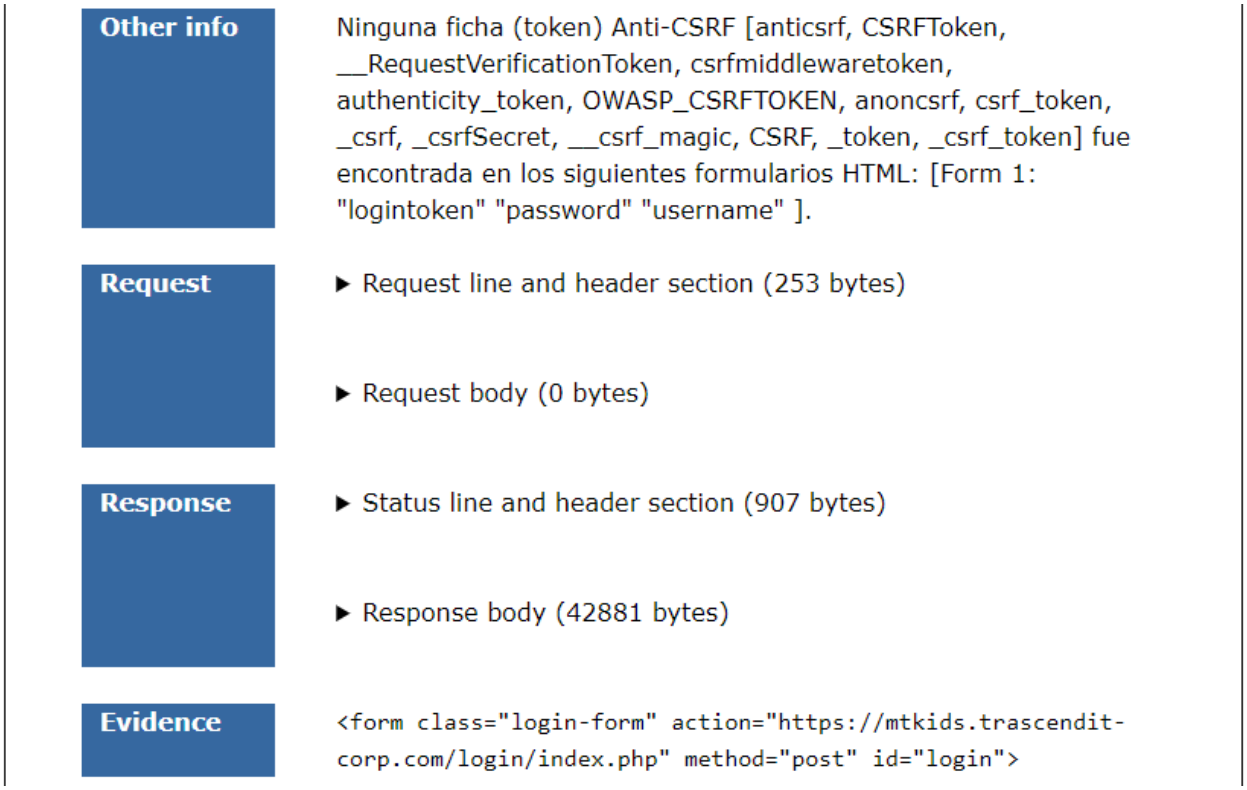
Los ataques de CSRG son muy efectivos en varias situaciones, que incluyen:

*La victima tiene una sesión activa en el sitio de destino.

*La víctima se autoriza por medio de la autenticación HTTP en el sitio de destino.

*La víctima se encuentra en la misma red local que el sitio de destino.

CSRF se ha utilizado especialmente para poder realizar una acción contra un sitio objetivo utilizando los privilegios de la víctima, pero se han revelado técnicas recientes para difundir información al obtener el acceso a la respuesta. El riesgo de divulgación de información aumenta de forma drástica cuando el sitio de destino se encuentra vulnerable a XSS, porque XSS se puede utilizar como una plataforma para CSRF, lo que le permite al atacante que opere desde adentro de los líites de la misma política de origen.



Fuente: SONARQUBE

Explicación

Los ataques CSRF explotan la confianza que un sitio web tiene en las solicitudes autenticadas de un usuario. Un atacante puede engañar a un usuario para que envíe una solicitud maliciosa a una aplicación web en la que el usuario está autenticado. Sin una protección adecuada, la aplicación web no puede distinguir entre una solicitud legítima y una solicitud CSRF malintencionada.

Impacto

La ausencia de tokens anti-CSRF puede permitir que un atacante realice acciones no autorizadas en nombre de los usuarios autenticados en la plataforma Moodle. Esto podría incluir acciones como cambiar las calificaciones de los estudiantes, modificar la configuración del curso, acceder a información confidencial o incluso comprometer la integridad de todo el entorno de Docker Swarm.

En la siguiente *Tabla 14* se presenta la propuesta de solución a la vulnerabilidad encontrada de tipo "Ausencia de Ttokens Anti-CSRF" en la plataforma Moodle en la cual se detalla a continuación:

Tabla 14

Matriz de propuesta de solución a la vulnerabilidad de tipo "Ausencia de Ttokens Anti-CSRF".

SOLUCIÓN	Fase: Arquitectura y Diseño Utilizar una biblioteca o framework verificado y confiable que evite esta vulnerabilidad o proporcione elementos que faciliten evitarla. Por ejemplo, utilice el paquete anti-CSRG como el CSRGuard de OWASP. Fase: Implementación Asegurarse de que la aplicación esté libre de fallas de secuencias de comandos entre sitios, ya que la mayoría de las defensas de CSRF pueden detenerse por alto por medio del uso de secuencias de comandos manejadas por el atacante. Fase: Arquitectura y Diseño Originar un nonce único para cada uno de los formularios, colocar el nonce en el formulario y confirmar la independencia al obtener el formulario. Asegurarse de que el nonce no sea predecible (CWE-330). Tener en cuenta que esto puede pasar desapercibido utilizando XSS. Identificar las operaciones que sean especialmente peligrosas. Cuando el usuario desarrolla una operación peligrosa, envíe una solicitud de confirmación de forma separada para poder garantizar que el usuario tenga la intención de desarrollar esa operación. Tener en cuenta que esto puede pasar desapercibido utilizando XSS. Utilizar el control de gestión de la sesión de ESAPI. Este control introduce un elemento para CSRF. No utilizar el método GET para ninguna de las solicitudes que puedan desencadenar un cambio de estado. Fase: Implementación

	Revisar que la solicitud se creó en la página esperada. Esto podría quebrar la funcionalidad auténtica, ya que los usuarios o los representantes puede ser que hayan desactivado el envío de Referer por motivos de privacidad.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

Ilustración 18 Vulnerabilidad tipo "Directory Browsing (Exploración de directorios)"

Fuente: SONARQUBE

Explicación

Esto ocurre cuando un servidor web permite a los usuarios listar el contenido de los directorios en ausencia de un archivo índice (como index.html o index.php). Esto puede exponer archivos y directorios que contienen información sensible o que podrían ser explotados por atacantes.

Impacto

La habilitación del Directory Browsing puede exponer archivos y directorios que contienen información sensible, lo que puede ser explotado por atacantes para obtener datos críticos, realizar ingeniería inversa sobre configuraciones, o descubrir otras vulnerabilidades.

En la siguiente *Tabla 15* se presenta la propuesta de solución a la vulnerabilidad encontrada de tipo “Directory Browsing” en la plataforma Moodle en la cual se detalla a continuación:

Tabla 15

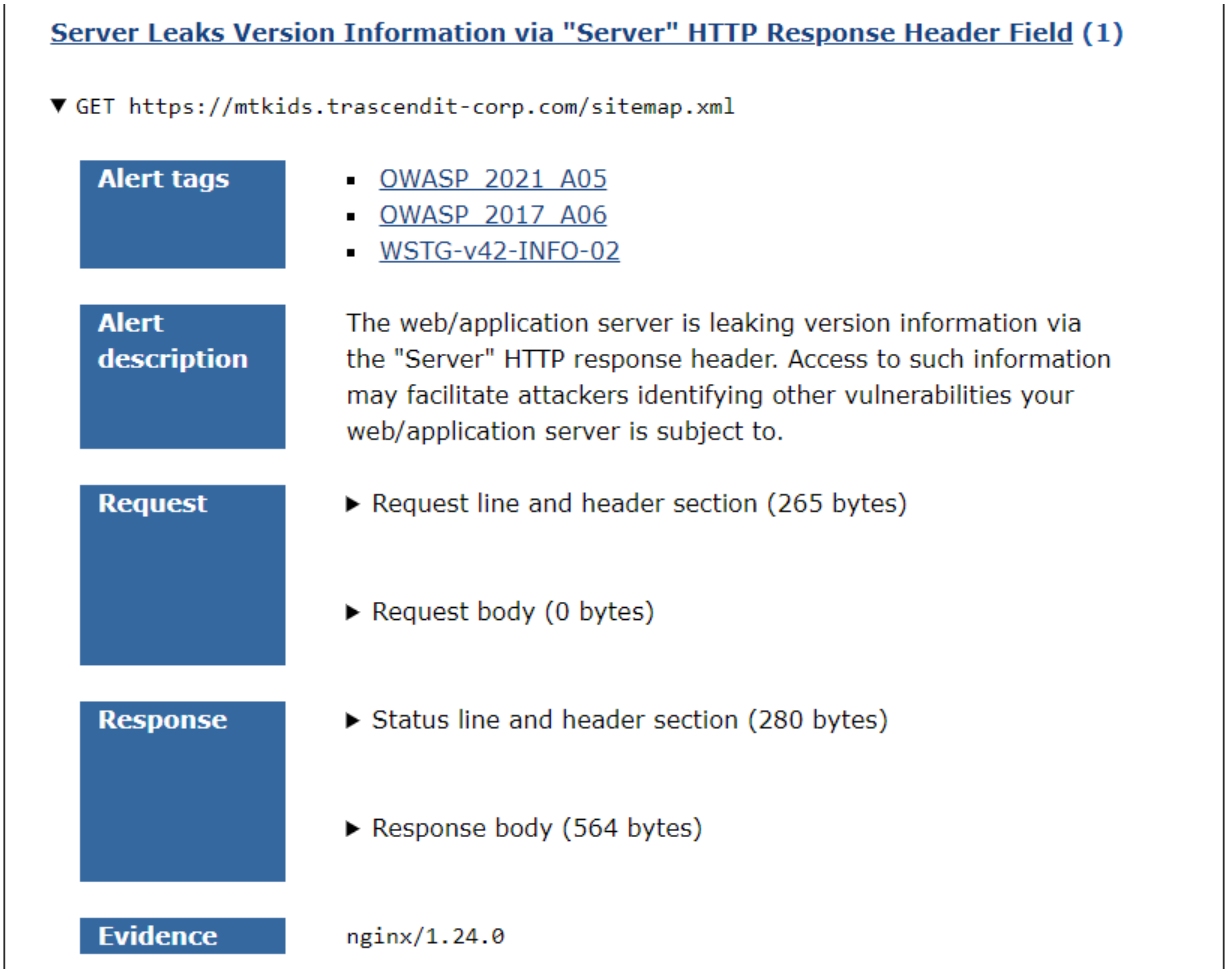
Matriz de propuesta de solución a la vulnerabilidad de tipo “Directory Browsing”.

SOLUCIÓN	Deshabilitar la visualización de índices de directorios en la configuración del servidor web. Esto se puede lograr añadiendo la directiva "Options -Indexes" en los archivos de configuración de Apache o en el archivo .htaccess.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

3.3.Vulnerabilidades de riesgo BAJO

Ilustración 19 Vulnerabilidad tipo "Server Leaks Version Information via Server, HTTP Response Header Field"



Fuente: SONARQUBE

Explicación

La cabecera "Server" en las respuestas HTTP generalmente contiene información sobre el servidor web, como el nombre del software y la versión. Aunque esta información puede ser útil para fines de depuración y soporte, también puede revelar detalles que podrían ser aprovechados por atacantes para identificar y explotar vulnerabilidades conocidas en esa versión específica del software del servidor web.

Impacto

la divulgación de información detallada de versión a través de la cabecera "Server" podría exponer vulnerabilidades en el servidor web utilizado para desplegar la plataforma Moodle en el entorno de Docker Swarm. Los atacantes podrían aprovechar esta información para buscar y explotar vulnerabilidades conocidas en esa versión específica del software del servidor web

En la siguiente *Tabla 16* se presenta la propuesta de solución a la vulnerabilidad encontrada de tipo “Server Leaks Version Information via Server, HTTP Response Header Field” en la plataforma Moodle en la cual se detalla a continuación:

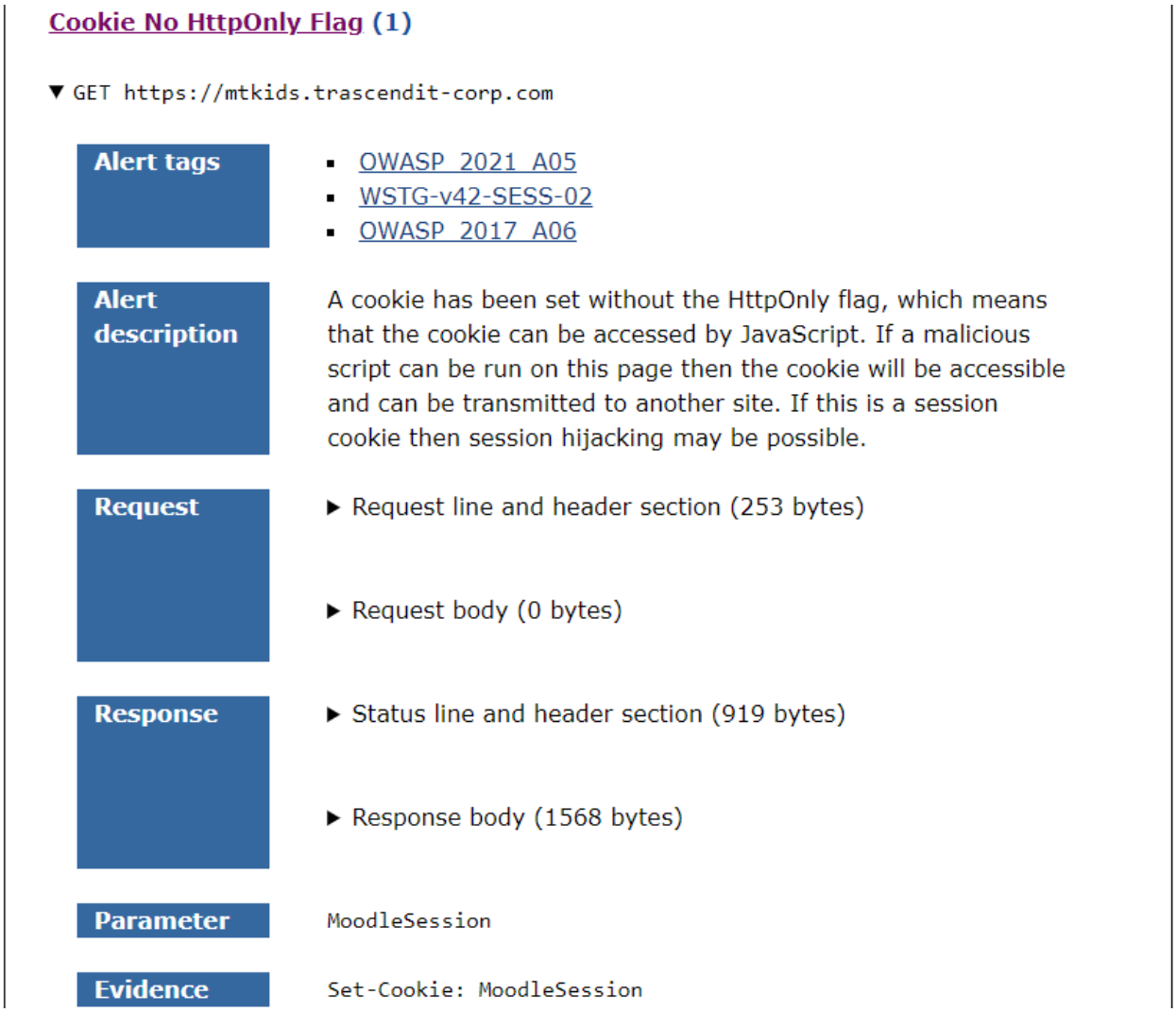
Tabla 16

Matriz de propuesta de solución a la vulnerabilidad de tipo “Server Leaks Version Information via Server, HTTP Response Header Field”.

SOLUCIÓN	Configurar el servidor web para que no envíe la cabecera "Server" o para que envíe una cadena genérica en su lugar. Si el servidor web no permite ocultar completamente la cabecera, configurarlo para que envíe información de versión limitada o enmascarada. Implementar un proxy reverso o una capa de seguridad intermedia que modifique o elimine la cabecera "Server" antes de enviar la respuesta al cliente.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

Ilustración 20 Vulnerabilidades tipo "Cookie No HttpOnly Flag"



Fuente: SONARQUBE

Explicación

La bandera HttpOnly es un atributo de seguridad que se puede agregar a las cookies para evitar que sean accedidas por scripts del lado del cliente, como JavaScript. Cuando esta bandera no está habilitada, las cookies pueden ser accedidas y manipuladas por código malicioso en el navegador del usuario, lo que podría permitir ataques como el robo de sesiones o el secuestro de cookies.

Impacto

La ausencia de la bandera HttpOnly en las cookies emitidas por la plataforma Moodle podría permitir que un atacante acceda y manipule las cookies de autenticación y sesión de los usuarios. Esto podría conducir a ataques como el secuestro de sesiones, lo que permitiría a un atacante asumir la identidad de un usuario legítimo y acceder a información confidencial o realizar acciones no autorizadas en la plataforma.

En la siguiente *Tabla 17* se presenta la propuesta de solución a la vulnerabilidad encontrada de tipo "Cookie No HttpOnly Flag" en la plataforma Moodle en la cual se detalla a continuación:

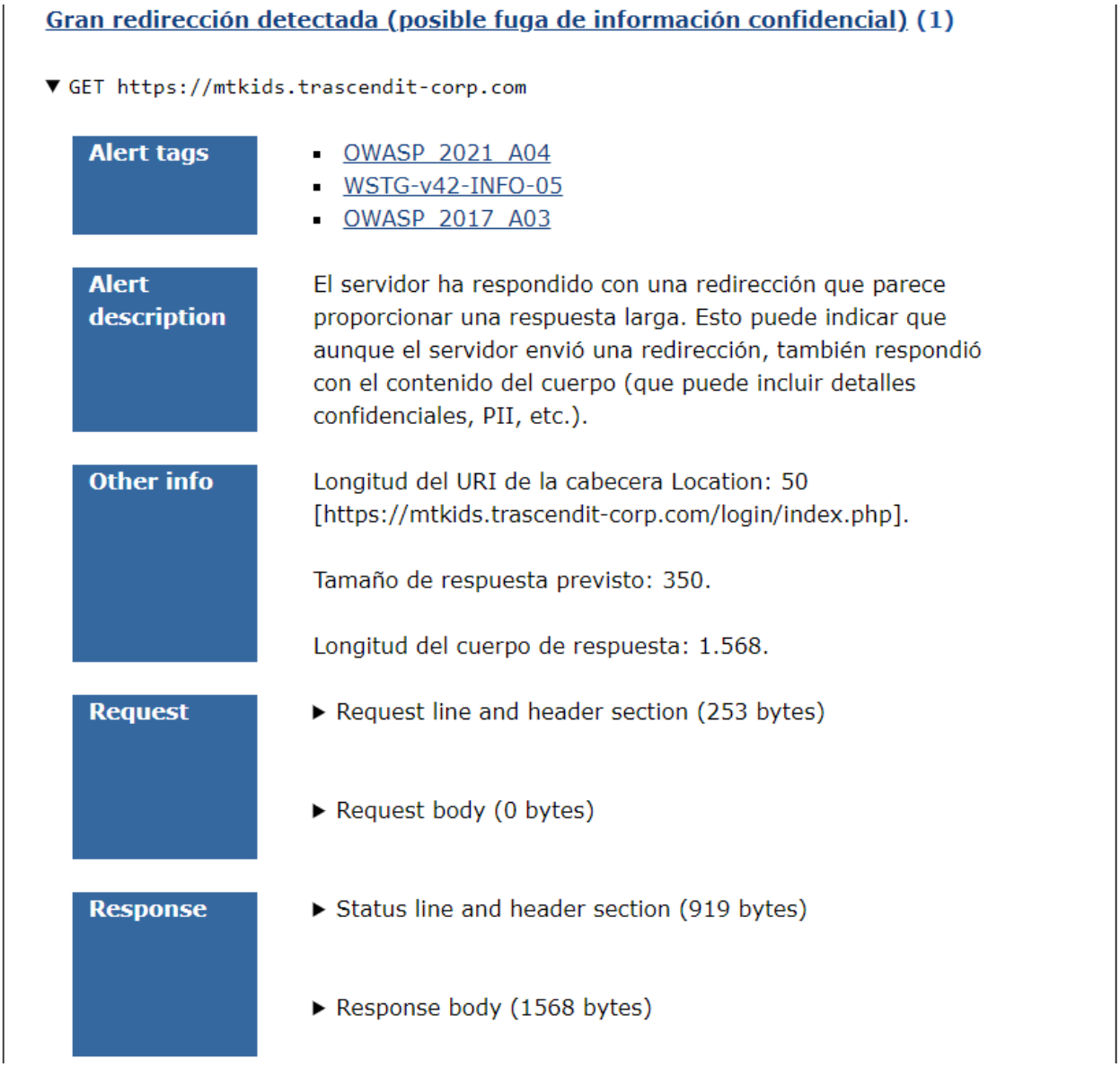
Tabla 17

Matriz de propuesta de solución a la vulnerabilidad de tipo "Cookie No HttpOnly Flag".

SOLUCIÓN	Agregar el atributo HttpOnly al definir las cookies en el código fuente de la aplicación o en la configuración del servidor web o el contenedor Docker utilizado para desplegar la plataforma Moodle en el entorno de Docker Swarm. Implementar otras medidas de seguridad relacionadas con las cookies, como el uso de cookies seguras (Secure flag) y el establecimiento de la directiva de Same-Site para mitigar ataques de tipo CSRF.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

Ilustración 21 Vulnerabilidad tipo "Gran redirección detectada (posible fuga de información confidencial)"



Fuente: SONARQUBE

Explicación

La vulnerabilidad de gran redirección ocurre cuando un sistema permite redirecciones abiertas sin validar adecuadamente las URLs de destino. Esto puede ser explotado por atacantes para redirigir

a usuarios a sitios maliciosos o no autorizados, lo que puede resultar en la fuga de información confidencial o en ataques de phishing.

Impacto

La explotación de gran redirección puede facilitar ataques de phishing, permitiendo a los atacantes engañar a los usuarios para que revelen información sensible como credenciales de inicio de sesión. Además, puede llevar a la exposición de datos confidenciales al redirigir tráfico a sitios no seguros. Esto no solo compromete la seguridad y privacidad de los usuarios,

En la siguiente *Tabla 18* se presenta la propuesta de solución a la vulnerabilidad encontrada de tipo "Gran redirección detectada (posible fuga de información confidencial)" en la plataforma Moodle en la cual se detalla a continuación:

Tabla 18

Matriz de propuesta de solución a la vulnerabilidad de tipo "Gran redirección detectada (posible fuga de información confidencial)".

SOLUCIÓN	Implementar validaciones estrictas sobre las URLs de redirección incluyendo asegurar que todas las redirecciones se realicen solo a dominios confiables y predefinidos, así como validar las entradas del usuario que pueden influir en el proceso de redirección.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

Ilustración 22 Vulnerabilidad tipo "Divulgación de la marca de hora - Unix".

Fuente: SONARQUBE

Explicación

La marca de tiempo Unix, también conocida como tiempo epoch, es una representación numérica del tiempo transcurrido desde la medianoche UTC del 1 de enero de 1970. Aunque esta

información puede ser útil para fines de depuración y registro, también puede ser aprovechada por atacantes para obtener información sobre el sistema y planificar ataques más efectivos.

Impacto

la divulgación de la marca de tiempo Unix podría proporcionar a los atacantes información valiosa sobre el sistema y la infraestructura en la que se despliega la plataforma Moodle. Esta información podría ser utilizada para identificar patrones y comportamientos del sistema, lo que facilitaría la planificación de ataques más específicos y efectivos.

En la siguiente *Tabla 19* se presenta la propuesta de solución a la vulnerabilidad encontrada de tipo "Divulgación de la marca de hora - Unix" en la plataforma Moodle en la cual se detalla a continuación:

Tabla 19

Matriz de propuesta de solución a la vulnerabilidad de tipo "Divulgación de la marca de hora – Unix".

SOLUCIÓN	Configurar el servidor web o el contenedor Docker para que no envíe la cabecera "Last-Modified" o cualquier otra cabecera que revele información de tiempo detallada. Enmascarar o eliminar la información de tiempo en los archivos enviados por la plataforma Moodle, como archivos de registro o archivos de respaldo. Utilizar una representación de tiempo más genérica o relativa en lugar de la marca de tiempo Unix detallada.

Nota: Elaboración propia adaptada del informe generado por SONARQUBE

3.4.Consideraciones finales

La seguridad es un aspecto crucial en estas configuraciones que presentan desafíos únicos los cuales deben abordarse adecuadamente para proteger los datos sensibles y garantizar la integridad del sistema. Estos enfoques brindan una perspectiva sólida para comprender las implicaciones de seguridad específicas, identificar las limitaciones del estudio y establecer una política de gestión de parches y actualizaciones efectiva.

3.4.1. Implicaciones de seguridad específicas para plataformas educativas tecnológicas como Moodle en entornos de contenedores.

SonarQube es una herramienta de análisis estático de código que ayuda a identificar vulnerabilidades y problemas de calidad en el código fuente. En el caso de las plataformas educativas como Moodle, desplegadas en entornos de contenedores, SonarQube puede ser especialmente útil para detectar vulnerabilidades relacionadas con inyecciones (SQL, XSS, comandos), fallas de control de acceso, manejo de datos confidenciales, entre otros.

Además, la metodología "desencadenada" de SonarQube permite realizar un análisis continuo y automatizado del código, lo que facilita la identificación temprana de vulnerabilidades y problemas de seguridad a medida que se desarrollan nuevas funcionalidades o se realizan actualizaciones en la plataforma.

Por otro lado, los estándares CVSS y CVE son cruciales para comprender el impacto y la gravedad de las vulnerabilidades identificadas. El sistema de puntuación CVSS proporciona una forma estandarizada de evaluar la severidad de una vulnerabilidad en función de factores como el vector de ataque, la complejidad de explotación, el nivel de privilegios requeridos, entre otros. Esto permite priorizar y abordar las vulnerabilidades más críticas de manera efectiva.

Asimismo, la base de datos CVE es una referencia ampliamente utilizada que proporciona información detallada sobre vulnerabilidades conocidas y sus impactos potenciales. Al utilizar esta información en conjunto con los hallazgos de SonarQube, es posible obtener una comprensión más

profunda de las vulnerabilidades específicas que afectan a la plataforma Moodle y su entorno de contenedores.

3.4.2. Limitaciones del estudio y áreas para futuras investigaciones:

Si bien la herramienta SonarQube y los estándares CVSS y CVE son recursos valiosos para evaluar la seguridad de una plataforma, es importante reconocer algunas limitaciones y áreas de mejora:

a) **Cobertura de análisis**: SonarQube se enfoca principalmente en el análisis estático del código fuente, lo que significa que puede haber vulnerabilidades relacionadas con la configuración, las dependencias de terceros o la interacción entre componentes que no sean detectadas. Una evaluación completa de seguridad también debería incluir pruebas dinámicas y de penetración.

b) **Falsos positivos y falsos negativos**: Como cualquier herramienta automatizada, SonarQube puede generar falsos positivos (alertas de vulnerabilidades inexistentes) o falsos negativos (no detectar vulnerabilidades reales). Esto requiere una revisión manual y una comprensión profunda del contexto y el código para identificar y mitigar correctamente las vulnerabilidades.

c) **Actualización de reglas y bases de datos**: Tanto las reglas de análisis de SonarQube como las bases de datos CVSS y CVE deben mantenerse actualizadas para reflejar las últimas vulnerabilidades y métodos de explotación. Un proceso de actualización regular es crucial para garantizar la precisión y relevancia de los resultados.

d) **Limitaciones de los estándares**: Si bien CVSS y CVE son ampliamente utilizados, pueden tener limitaciones en cuanto a la descripción detallada de vulnerabilidades específicas o la consideración de factores contextuales únicos de cada entorno.

Algunas áreas potenciales para futuras investigaciones podrían incluir:

- Desarrollo de técnicas de aprendizaje automático o inteligencia artificial para mejorar la precisión en la detección de vulnerabilidades y reducir los falsos positivos.

- Investigación sobre la aplicabilidad de los hallazgos y recomendaciones a otros entornos de contenedores o plataformas educativas diferentes a Moodle.
- Estudios comparativos entre diferentes soluciones de despliegue y evaluación de su eficacia en términos de seguridad.

3.4.3. Política de gestión de parches y actualizaciones:

Teniendo en cuenta que SonarQube se utiliza para el análisis continuo del código, es crucial mantener actualizadas las reglas de análisis y las bases de datos de vulnerabilidades. Además, es fundamental mantener actualizados todos los componentes de la plataforma Moodle y el entorno de Docker Swarm para mitigar las vulnerabilidades identificadas. Por lo tanto, la política de gestión de parches y actualizaciones debe abordar los siguientes aspectos:

a) **Monitoreo de actualizaciones de SonarQube, CVSS y CVE**: Establecer un proceso para monitorear regularmente las fuentes oficiales de SonarQube y las bases de datos CVSS y CVE, a fin de mantenerse al tanto de las nuevas reglas de análisis, actualizaciones de puntuación de vulnerabilidades y nuevas entradas de vulnerabilidades conocidas.

b) **Actualización de SonarQube y reglas de análisis**: Definir un proceso para actualizar periódicamente la instancia de SonarQube y las reglas de análisis. Esto puede requerir pruebas exhaustivas en un entorno de desarrollo o prueba para validar la compatibilidad y el correcto funcionamiento antes de aplicar los cambios en el entorno de producción.

c) **Monitoreo de vulnerabilidades y parches para Moodle y componentes de Docker**: Establecer un proceso para monitorear regularmente las fuentes oficiales de Moodle, Docker, sistema operativo subyacente y cualquier otra dependencia, a fin de mantenerse al tanto de las últimas vulnerabilidades reportadas y parches de seguridad disponibles.

d) **Ventanas de mantenimiento programadas**: Definir ventanas de mantenimiento regulares durante las cuales se puedan aplicar parches y actualizaciones de forma controlada, minimizando el impacto en los usuarios y los servicios en línea. Esto implica la coordinación con los equipos de desarrollo, operaciones y seguridad involucrados.

e) **Pruebas y validación**: Antes de implementar parches o actualizaciones, es crucial realizar pruebas exhaustivas en un entorno de prueba o desarrollo para validar la compatibilidad y el correcto funcionamiento de la plataforma Moodle y los componentes relacionados.

f) **Comunicación y documentación**: Establecer canales de comunicación efectivos para informar a los usuarios, el personal técnico y los interesados relevantes sobre las actualizaciones y parches aplicados, así como mantener una documentación detallada del proceso y los cambios realizados.

g) **Monitoreo y evaluación**: Después de la implementación de parches y actualizaciones, es crucial monitorear de cerca el rendimiento y la estabilidad de la plataforma, así como los resultados de SonarQube, para detectar y abordar cualquier problema o vulnerabilidad residual que pueda surgir.

h) **Respaldo y recuperación**: Implementar procedimientos sólidos de respaldo y recuperación para garantizar que los datos, el código fuente y la configuración puedan restaurarse en caso de problemas relacionados con las actualizaciones.

Las herramientas y estándares utilizados, como SonarQube, CVSS y CVE, son valiosos, se reconoce que existen áreas de mejora y oportunidades para futuras investigaciones. Sin embargo, la aplicación de las recomendaciones y medidas de mitigación propuestas, junto con un enfoque proactivo en la gestión de parches y actualizaciones, contribuirá significativamente a fortalecer la seguridad de estas plataformas educativas críticas y proteger los datos sensibles de los usuarios.

CONCLUSIONES

1. El análisis de vulnerabilidades realizado en el despliegue de la plataforma Moodle en Docker Swarm reveló la existencia de diversos riesgos de seguridad que deben ser abordados de manera efectiva para garantizar la integridad y confidencialidad de los datos manejados.

2. Las vulnerabilidades identificadas, como la configuración incorrecta de Cross-Domain, el uso de librerías JavaScript vulnerables, la falta de cabeceras anti-clickjacking, la ausencia de tokens anti-CSRF, la divulgación de información de versión y la ausencia de la bandera HttpOnly en las cookies, representan amenazas significativas que podrían ser explotadas por atacantes maliciosos.

3. La implementación de soluciones adecuadas, como la configuración correcta de políticas CORS, la actualización de librerías, la implementación de cabeceras de seguridad, el uso de tokens anti-CSRF y la ocultación de información sensible, es fundamental para mitigar los riesgos y fortalecer la postura de seguridad de la plataforma.

4. El entorno de despliegue en Docker Swarm introduce desafíos adicionales en términos de seguridad, ya que involucra múltiples contenedores y servicios interconectados, lo que requiere una configuración y gestión cuidadosa de los controles de seguridad.

5. La adopción de un enfoque proactivo y continuo en el análisis de vulnerabilidades y la implementación de medidas de seguridad es esencial para mantener un alto nivel de protección en un entorno dinámico y en constante evolución.

6. La seguridad es un proceso integral que debe ser considerado desde las etapas iniciales del diseño y desarrollo de la plataforma Moodle, y debe extenderse a lo largo de todo su ciclo de vida, incluyendo el despliegue y la operación en el entorno de Docker Swarm.

7. La colaboración y la comunicación efectiva entre los diferentes equipos involucrados, como desarrolladores, administradores de sistemas y expertos en seguridad, son fundamentales para lograr una implementación segura y eficiente de la plataforma Moodle en Docker Swarm.

8. La capacitación y concientización sobre prácticas de seguridad para todo el personal involucrado en el desarrollo, despliegue y operación de la plataforma Moodle es crucial para fomentar una cultura de seguridad sólida en la organización.

9. La documentación detallada de los procedimientos de seguridad, las configuraciones implementadas y las medidas de mitigación aplicadas facilitará la gestión y el mantenimiento continuos de la seguridad de la plataforma Moodle en Docker Swarm.

10. La evaluación periódica de la postura de seguridad, mediante pruebas de penetración, análisis de código fuente y revisiones de configuración, es esencial para identificar y abordar nuevas vulnerabilidades y amenazas emergentes de manera oportuna.

En resumen, el análisis de vulnerabilidades realizado en el despliegue de la plataforma Moodle sobre una arquitectura Docker Swarm reveló la existencia de diversos riesgos de seguridad que requieren un enfoque proactivo y continuo para su mitigación. La implementación de soluciones adecuadas, como la configuración correcta de políticas de seguridad, la actualización de librerías y la ocultación de información sensible, es fundamental para fortalecer la postura de seguridad de la plataforma. Además, la adopción de prácticas de seguridad sólidas, la colaboración efectiva entre equipos y la capacitación constante son aspectos clave para garantizar una implementación segura y eficiente en un entorno dinámico como Docker Swarm.

RECOMENDACIONES

1. Implementar una política de gestión de parches y actualizaciones para mantener todos los componentes de la plataforma Moodle y el entorno de Docker Swarm actualizados con los últimos parches de seguridad disponibles.

2. Establecer procedimientos rigurosos de validación de entrada y salida para prevenir ataques de inyección y otros vectores de ataque relacionados con la manipulación de datos.

3. Implementar controles de acceso basados en roles y autenticación robusta para limitar el acceso a recursos y funcionalidades críticas solo a usuarios autorizados.

4. Configurar adecuadamente las cabeceras de seguridad, como X-Frame-Options, Content Security Policy (CSP) y X-XSS-Protection, para mitigar ataques como clickjacking, inyección de scripts y otros vectores de ataque de origen cruzado.

5. Utilizar protocolos de comunicación seguros, como HTTPS, para proteger la transmisión de datos sensibles y evitar ataques de intercepción y manipulación de tráfico.

6. Implementar mecanismos de monitoreo y registro de actividades para detectar y responder rápidamente a cualquier actividad sospechosa o intento de ataque.

7. Realizar pruebas de penetración y análisis de código fuente periódicos para identificar y corregir vulnerabilidades en la plataforma Moodle y su entorno de despliegue.

8. Establecer un plan de respuesta a incidentes y un equipo dedicado para gestionar y mitigar de manera efectiva cualquier brecha de seguridad o incidente relacionado.

9. Fomentar una cultura de seguridad en toda la organización, mediante capacitaciones, concientización y la adopción de mejores prácticas de seguridad en todas las etapas del ciclo de vida del desarrollo y despliegue de la plataforma Moodle.

10. Mantenerse actualizado sobre las últimas tendencias y amenazas de seguridad, participando en comunidades de seguridad, suscribiéndose a boletines de seguridad y asistiendo a conferencias y eventos relacionados.

En conclusión, para mantener un alto nivel de seguridad en el despliegue de la plataforma Moodle sobre una arquitectura Docker Swarm, es esencial implementar medidas de seguridad robustas, como la gestión de parches, la validación de entrada/salida, el control de acceso, la configuración adecuada de cabeceras de seguridad, el uso de protocolos seguros, el monitoreo y el registro de actividades. Además, es crucial realizar pruebas de penetración y análisis de código fuente periódicos, establecer un plan de respuesta a incidentes y fomentar una cultura de seguridad en toda la organización mediante capacitaciones y concientización constante. Mantenerse actualizado sobre las últimas tendencias y amenazas de seguridad también es fundamental para garantizar la protección continua de la plataforma Moodle y su infraestructura de despliegue en Docker Swarm.

BIBLIOGRAFÍA

Alshammari, M. T., Panichet, R. V. L., & Alhashmi, S. F. (2021). Moodle security assessment: A comprehensive analysis and methodology. IEEE Access, 9, 106933-106949. https://doi.org/10.1109/ACCESS.2021.3100262

Andress, J., & Winterfeld, S. (2014). Cyber warfare: Techniques, tactics and tools for security practitioners (2nd ed.). Syngress.

Combe, T., Martin, A., & Di Pietro, R. (2016). To docker or not to docker: A security perspective. IEEE Cloud Computing, 3(5), 54-62. https://doi.org/10.1109/MCC.2016.100

Creswell, J. W., & Creswell, J. D. (2018). Research design: Qualitative, quantitative, and mixed methods approaches. Sage publications.

FIRST. (2021). Common Vulnerability Scoring System (CVSS). https://www.first.org/cvss/

Instituto Nacional de Estándares y Tecnología (NIST). (2012). Guide for conducting risk assessments (NIST Special Publication 800-30 Rev. 1). https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf

Instituto Nacional de Estándares y Tecnología (NIST). (2013). Security and privacy controls for federal information systems and organizations (NIST Special Publication 800-53 Rev. 4). https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf

ISO/IEC 27005 (2018). Information technology -- Security techniques -- Information security risk management.

Lazar, J., Feng, J. H., & Hochheiser, H. (2017). Research methods in human-computer interaction. Morgan Kaufmann.

Leitner, P., Bezemer, C. P., Herzig, K., & Lenarduzzi, V. (2020). Automating security testing: Problems and solutions. IEEE Software, 37(6), 30-37. https://doi.org/10.1109/MS.2020.3012224

Merriam, S. B., & Tisdell, E. J. (2016). Qualitative research: A guide to design and implementation. Jossey-Bass.

Metasploit: Penetration Testing Software. https://www.metasploit.com/

Moodle. (2023a). About Moodle. https://moodle.org/about/

National Institute of Standards and Technology. (2018). Guide for Conducting Risk Assessments (NIST SP 800-30 Rev. 1). https://csrc.nist.gov/publications/detail/sp/800-30/rev-1/final

NIST. (2022). NIST Special Publication 800-53 Revision 5: Security and Privacy Controls for Information Systems and Organizations. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r5.pdf

OWASP Dependency Check. https://owasp.org/www-project-dependency-check/

OWASP. (2021). OWASP Top 10. Open Web Application Security Project. https://owasp.org/www-project-top-ten/

OWASP. (2023a). OWASP Testing Guide v4.3. https://owasp.org/www-project-web-security-testing-guide/

OWASP. (2023b). OWASP Top Ten Web Application Security Risks. https://owasp.org/www-project-top-ten/

OWASP. (2023c). OWASP Zed Attack Proxy (ZAP). https://owasp.org/www-project-zap/

PortSwigger. (s.f.). Burp Suite. https://portswigger.net/burp OWASP. (s.f.).

PTES. (2022). Penetration Testing Execution Standard. http://www.pentest-standard.org/index.php/PTES_Technical_Guidelines

Selenium. (s.f.). Selenium: Documentation. https://www.selenium.dev/documentation/ SonarSource. (s.f.). SonarQube: An open-source platform for code quality. https://www.sonarqube.org/

Sommerville, I. (2016). Ingeniería de software. Pearson Educación.

Souppaya, M., Morello, J., & Scarfone, K. (2017). Application container security guide.

Stake, R. E. (1995). The art of case study research. Sage publications.

Stuttard, D. y Pinto, M. (2011). The web application hacker's handbook: Finding and exploiting security flaws (2ª ed.). Wiley Publishing, Inc.

Tenable. (s.f.). Nessus: Vulnerability Scanning. https://www.tenable.com/products/nessus Rapid7. (s.f.).

Utrero, P., Muñoz, A., Bernabé, G., Palma, J., Silva, L., & Caivano, D. (2019). An empirical study on the integration of security testing in continuous integration/continuous delivery (CI/CD) pipelines. Information and Software Technology, 117, 106196.

Whitman, M. E., & Mattord, H. J. (2018). Principles of information security (6th ed.). Cengage Learning.

Yin, R. K. (2018). Case study research and applications: Design and methods. Sage publications.

Trascend-IT Corp.

CARTA DE ACEPTACIÓN

Por medio de la presente **la empresa TRASCEND-IT**, con número de RUC 1719636431001, tiene el agrego de informar que el Sr. **ALEJANDRO PAUL AMAYA MELO,** portador de la cédula de ciudadanía N° 1003753934 realizo la entrega del proyecto **"Análisis forense de código y procesos de la plataforma educativa tecnológica Moodle desplegada sobre la infraestructura Docker Swarm de la empresa Trascend-IT"** que se desarrolló en beneficio de la seguridad de las plataformas de nuestra empresa.

Es grato informar que el proyecto culminado tuvo un alto grado de satisfacción dentro de la empresa, cumpliendo con todos los requisitos establecidos. Demostrando su compromiso, capacidad y profesionalismo en el desarrollo del proyecto.

Atentamente,

DIEGO FERNANDO LARENAS ARRESE

Ing. Diego Larenas

CEO/Trascend-IT

info@trascend-it.com
Portugal E10-386 y
Catalina Aldaz Quito-Ecuador
www.trascend-it.com