VOLUMEN 5, N°2 / JULIO - DICIEMBRE 2023 / e - ISSN: 2661

ANÁLISIS DE VULNERABILIDADES Y PRUEBAS DE ESTRÉS EN EL SISTEMA ACADÉMICO DE LA ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO: UNA EVALUACIÓN INTEGRAL

Vulnerability Analysis and Stress Testing in the Academic System of the Escuela Superior Politécnica de Chimborazo: A Comprehensive Evaluation

Astudillo Muñoz Erika Michelle [1]	erika.mastudillo@gmail.com
Vizuete Ulloa Andrea Elizabeth [2]	eli.vizuete@gmail.com
Diego Avila-Pesantez [3]	davila@espoch.edu.ec
Danilo Pástor Ramírez [4]	dpastor@espoch.edu.ec
L. Miriam Avila [5]	miriam.avila@espoch.edu.ec

[1,2] Investigador independiente,

[3] Grupo de Investigación en Innovación Científica y Tecnológica. Escuela Superior Politécnica de Chimborazo,

[4] Facultad de Informática y Electrónica. Escuela Superior Politécnica de Chimborazo

[5] Facultad de Administración de Empresas. Escuela Superior Politécnica de Chimborazo

Riobamba – Ecuador

RESUMEN

En la era digital, es importante asegurar la seguridad y el óptimo rendimiento de las aplicaciones web, para proteger la información confidencial que se maneja. Con el aumento constante de las amenazas informáticas y los ataques cibernéticos, las organizaciones deben tomar medidas proactivas para evitar posibles violaciones de seguridad y pérdidas de datos. Además, un bajo rendimiento de las aplicaciones web puede tener un impacto negativo en la experiencia del usuario y en la eficiencia de los procesos empresariales. Con este antecedente, se llevó a cabo un análisis exhaustivo de las vulnerabilidades del sistema académico de la ESPOCH, siguiendo las etapas propuestas por la metodología OWASP y utilizando la herramienta OWASP ZAP. Además, se realizaron pruebas de estrés en el dicho aplicativo para comprobar si puede manejar la cantidad de solicitudes esperadas, mediante diversos cálculos. Como resultado de este análisis, se identificaron 15 debilidades en el sistema académico y se proporcionó las mejores prácticas para su mitigación, con resultados favorables. Finalmente, se determinó que el sistema académico está preparado para atender a una gran cantidad de peticiones de usuarios durante un período de cinco años y garantizar la disponibilidad de los servicios en momentos críticos.

Palabras Clave: Análisis de Vulnerabilidades, Metodología OWASP, Pruebas de Estrés, Sistema Académico de la ESPOCH.

ABSTRACT

In the digital age, it is important to ensure the security and optimal performance of web applications to protect sensitive information. With the constant increase in cyber threats and attacks, organizations must take proactive measures to prevent potential security breaches and data loss. Additionally, poor performance of web applications can have a negative impact on user experience and the efficiency of business processes. With this background, a comprehensive analysis of vulnerabilities in the academic system of ESPOCH was conducted, following the stages proposed by the OWASP methodology and using the OWASP ZAP tool. Stress tests were also performed on the mentioned application to

Fecha de Recepción 23/jun/2023. Fecha de Aceptación: 02/jul/2023 DOI:https://doi.org/10.47187/perspectivas.5.2.196

verify if it can handle the expected number of requests through various calculations. As a result of this analysis, 15 weaknesses were identified in the academic system, and best practices were provided for their mitigation, yielding favorable results. Finally, it was determined that the academic system is prepared to handle many user requests over a period of five years and ensure the availability of services during critical moments.

Keywords: Vulnerability Analysis, OWASP Methodology, Stress Tests, ESPOCH Academic System.

I. Introducción

En la actualidad, las aplicaciones web se han convertido en una herramienta imprescindible para las empresas y organizaciones de todo el mundo, dado que ofrecen a los usuarios la posibilidad de acceder a servicios y realizar tareas de forma rápida y eficiente desde cualquier ubicación. Sin embargo, a medida que aumenta la popularidad de estas aplicaciones, también aumentan los riesgos de seguridad. Las vulnerabilidades presentes en las aplicaciones web pueden ser explotadas por los atacantes para obtener información confidencial tanto de las empresas como de los usuarios involucrados [1]. Para abordar estos riesgos, es fundamental llevar a cabo un análisis de vulnerabilidades en las aplicaciones web, para implementar medidas preventivas que protejan la integridad de los datos y la privacidad de los usuarios [2]. Este artículo se enfoca en resaltar la importancia del análisis de vulnerabilidades en las aplicaciones web utilizando la metodología OWASP, la cual es ampliamente reconocida y utilizada en todo el mundo con el enfoque de prevenir y mantener la seguridad en las aplicaciones [3].

Por consiguiente, se propone aplicar esta metodología para realizar un estudio de vulnerabilidades en el sistema académico desarrollado por la Escuela Superior Politécnica de Chimborazo (ESPOCH), con la finalidad de recomendar mejores prácticas de seguridad efectivas para reducir los riesgos de seguridad en el sistema académico.De la misma manera, es crucial realizar pruebas de estrés al sistema académico para evaluar su capacidad de respuesta ante grandes cargas de trabajo. Estas pruebas permiten determinar si el sistema puede mantener un rendimiento óptimo y una disponibilidad adecuada bajo condiciones de alta exigencia [4]. Durante la ejecución de las pruebas de estrés, se simularon múltiples usuarios conectados en un tiempo estimado durante el periodo de matrículas, con la finalidad de observar si el sistema académico soportaba una gran cantidad de peticiones al mismo tiempo.

Cabe destacar que es crucial garantizar la seguridad y el buen rendimiento en las aplicaciones web para evitar posibles fallas y asegurar la satisfacción de los usuarios. En el presente artículo se proponen mejores prácticas con el objetivo de prevenir y mitigar los fallos y debilidades encontrados en el sistema académico de la ESPOCH. Asimismo, se detalla la gran cantidad de solicitudes hechas por parte de los usuarios simulados durante el periodo de matrículas, con la finalidad de conocer si el sistema ha sido capaz de soportar el gran número de peticiones o si ha presentado errores durante las pruebas de estrés realizadas.

Este trabajo tiene un enfoque descriptivo, porque proporciona una descripción detallada de las etapas requeridas para llevar a cabo el análisis de vulnerabilidades en el sistema académico de la ESPOCH utilizando la metodología OWASP. De igual manera, para diseñar y ejecutar las pruebas de estrés se detalla el procedimiento a seguir.

El artículo sigue la siguiente estructura: primero, se realiza una revisión de los trabajos anteriores relacionados con el tema. Luego, en la sección 3, se lleva a cabo un análisis de vulnerabilidades aplicando la metodología OWASP, que proporciona una guía detallada para llevar a cabo dicho análisis en aplicaciones web. La sección 4 describe las pruebas de estrés y las diferentes fases involucradas en su ejecución. A continuación, en la sección 5, se presentan los resultados obtenidos, así como una propuesta de mejores prácticas para el análisis de vulnerabilidades y las pruebas de estrés realizadas en tres niveles. Finalmente, se presentan las conclusiones del estudio.

II. Trabajos similares

Se han realizado algunos estudios de vulnerabilidades en los sistemas académicos universidades del Ecuador y Argentina. Por ejemplo, en el sistema de matriculación de la unidad académica de ciencias empresariales de la UTMACH, se realizó un análisis de

vulnerabilidades, amenazas y riesgos al que tiene como finalidad mitigar las vulnerabilidades encontradas a través de una metodología descriptiva mediante un análisis comparativo, donde se destaca la falta de un plan integral que involucre a todas las dependencias y la insuficiencia de aplicación directa de metodologías de riesgo, por lo que es necesario renovar equipos, actualizarlos implementando controles diarios y tomar medidas para minimizar errores humanos, por lo cual se sugiere mejorar la usabilidad y funcionalidad mediante módulos adicionales, ya que el nivel de riesgo es bajo debido al control constante y la ausencia de incidentes [5]. Otro estudio realizado es un análisis de vulnerabilidades de sistema web en desarrollo y en producción, en el laboratorio de sistemas perteneciente a la Universidad Tecnológica Nacional en Argentina; para detectar las vulnerabilidades a través de pruebas de penetración en sistemas en desarrollo y en sistemas de producción, es importante tener en cuenta que obtener datos confiables para este tipo de pruebas depende de acuerdo de confidencialidad y confianza con los propietarios de los sistemas, los resultados obtenidos marcan el comienzo de un camino para establecer las bases de un sistema automatizado de pruebas de penetración. Además, se identificaron los desafíos en los términos de tiempo requerido para realizar estas acciones donde se enfatiza la necesidad de incorporar requisitos de seguridad desde la planificación de nuevos proyectos de desarrollo de software, así como gestionar múltiples pruebas de penetración en sistemas web en producción utilizando metodologías abiertas para identificar y analizar vulnerabilidades en el sistema, para que sea accesible y práctico para los expertos en seguridad [6]

En el trabajo de Jiang y Hassan realizaron pruebas de estrés en el sistema académico, con el objetivo de evaluar su capacidad de respuesta ante una carga de trabajo elevada. Se simularon situaciones de alta demanda, como la inscripción de estudiantes en línea, la consulta de notas y la generación de certificados, entre otras. Los resultados de la prueba de estrés permitieron identificar cuellos de botella y limitaciones en el sistema, que fueron corregidas para mejorar su capacidad de respuesta y garantizar la disponibilidad de los servicios en momentos críticos [6].

III. Análisis de vulnerabilidades aplicando la metodología OWASP

Para llevar a cabo el análisis de vulnerabilidades en el sistema académico ESPOCH, se siguieron las etapas definidas en la metodología OWASP, las cuales se describen a continuación.

A. Fase 1: Recopilación de información y diseño de escenario

Se realizó un análisis de la estructura de la aplicación, la infraestructura y la arquitectura del sistema académico de la ESPOCH. A continuación se presenta la arquitectura del sistema académico (ver Fig: 1):

- *Frontend:* Para la arquitectura del sistema académico se utiliza el patrón Single Page-Application (SPA), el cual consiste en mostrar una sola página web que carga el contenido dependiendo de los datos recibidos del backend a través de JavaScript y para el diseño la página se utiliza el Framework Angular [7], [8].
- *Backend:* Para la lógica de la aplicación, se utiliza la arquitectura de microservicios, porque permite dividir al sistema académico en varios servicios para cada una de las funcionalidades del sistema; por ejemplo: matrículas, notas, datos personales, entre otros servicios. Las bases de datos de la aplicación son relacionales, trabajando con SQL Server como gestor principal y PostgreSQL para otros servicios. Cabe recalcar que, cada servicio cuenta con su base de datos [7].
- *Infraestructura:* El sistema académico cuenta con recursos para cada uno de los servicios brindados por el sistema académico, los cuales se encuentran distribuidos de la siguiente manera:
  - A nivel de interfaz de usuario, el sistema académico trabaja con 3 nodos, cada uno con 8 procesadores y 12 GB en memoria RAM.
  - Para el backend, el sistema académico trabaja con 3 nodos para cada uno de los servicios web programados en Ubuntu Server; a su vez, cada nodo cuenta con 8 procesadores y 14 GB en memoria RAM.

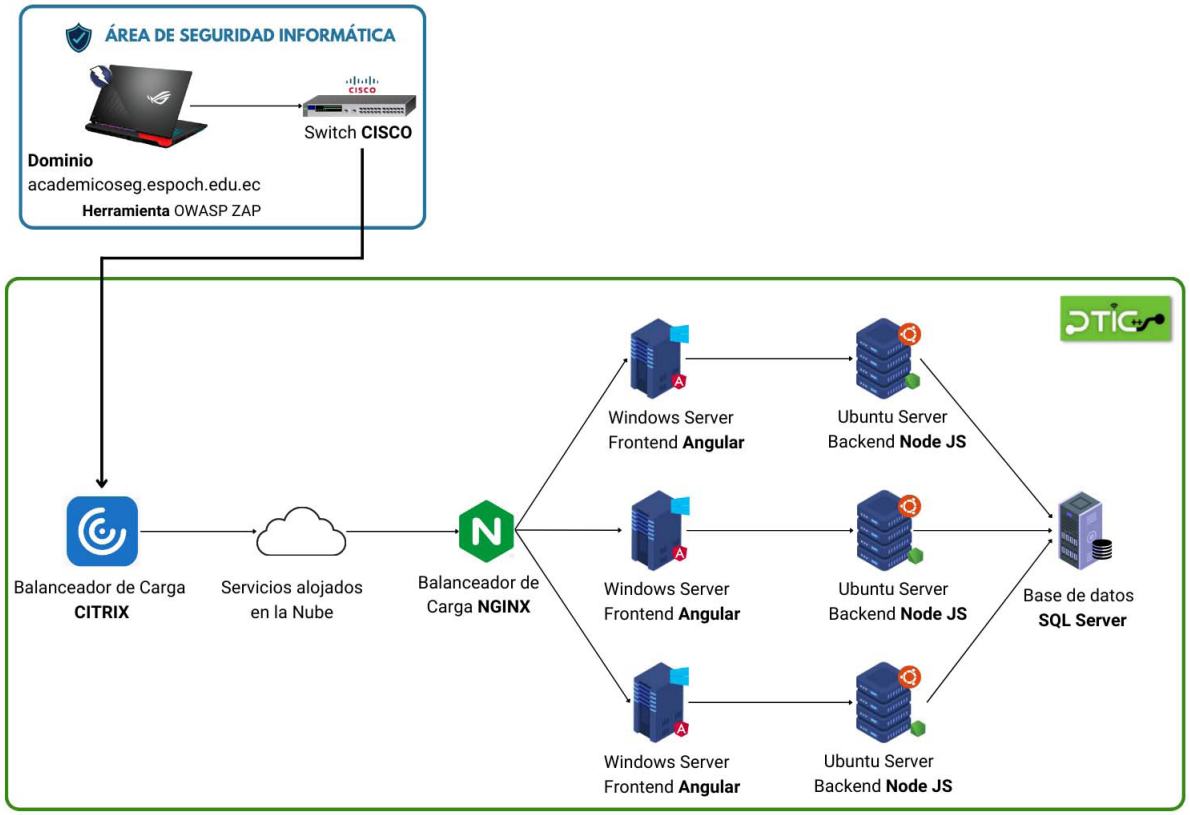

Fig. 1. *Escenario de ejecución de análisis de vulnerabilidades.*

B. Fase 2: Análisis y clasificación de vulnerabilidades

En esta fase se llevó a cabo el escaneo de vulnerabilidades en el sistema académico de la ESPOCH utilizando la herramienta OWASP ZAP (OWASP Zen Attack Proxy). Durante el análisis, se identificaron un total de 15 vulnerabilidades, las cuales fueron calificadas como de riesgo medio y bajo (ver Fig: 2):

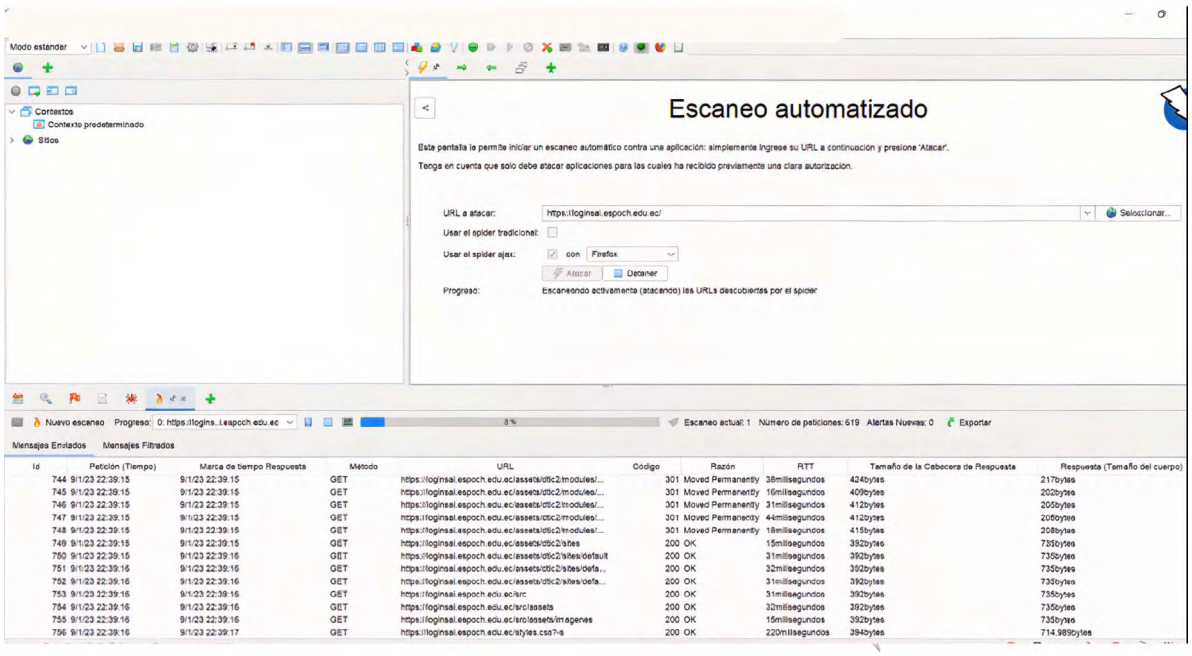

Fig. 2. *Escaneo de Vulnerabilidades y Análisis de Vulnerabilidades*

1) Ausencia de fichas (tokens) Anti-CSRF

Esta vulnerabilidad ocurre cuando una aplicación no utiliza tokens anti-CSRF (Cross-Site Request Forgery) para validar las solicitudes entrantes, lo que permite que un atacante pueda inyectar solicitudes maliciosas desde otro sitio web y realizar acciones en el contexto de la sesión activa del usuario. [9], [10].

2) Encabezado de política de seguridad de contenido (CSP) no establecido

Este problema se relaciona con la falta de una política de seguridad de contenido en el encabezado HTTP que informa al navegador web sobre qué tipo de contenido está permitido y desde qué origen puede provenir. [11], [12].

3) Falta el encabezado antisecuestro de clics

Esta vulnerabilidad se refiere a la ausencia de una cabecera Anti-clickjacking, que es utilizada para prevenir ataques de secuestro de clics. Este tipo de ataque ocurre cuando un atacante duplica el contenido de una página web en un sitio malicioso con el objetivo de engañar a los usuarios, solicitándoles que proporcionen información confidencial o realizando acciones no autorizadas utilizando los datos del usuario [13], [14].

4) Biblioteca JS vulnerable

Esta vulnerabilidad se produce cuando se emplea un marco de JavaScript desactualizado, obsoleto o que contiene código malicioso. Esto permite la divulgación de información confidencial alojada en la aplicación y otorga permisos no autorizados a los atacantes [15].

5) Cookie sin atributo SameSite

Esta vulnerabilidad genera un problema de seguridad que está relacionado con la forma en que una página web gestiona las cookies. Si estas cookies no están configuradas con el atributo SameSite, un sitio malicioso puede enviar una solicitud POST al dominio del sitio vulnerable para obtener automáticamente las cookies. Esto puede dar lugar a un ataque de falsificación de solicitudes entre sitios CSRF [16], [17].

6) Divulgación de la marca de hora – Unix

Esta vulnerabilidad se identifica como un problema de seguridad que permite a los atacantes descubrir la fecha y hora exacta en que un sistema operativo Unix ha sido reiniciado. Esto brinda a los atacantes la posibilidad de llevar a cabo ataques más efectivos, ya que tienen conocimiento de la última vez que el sistema fue reiniciado [18].

7) El servidor divulga información mediante un campo de encabezado de respuesta HTTP ““X-Powered-By””

Esta debilidad de seguridad provoca una filtración de información confidencial a través de uno o más encabezados de respuesta X-Powered-By, los cuales pueden ser aprovechados por los atacantes. Estos encabezados concluyen detalles específicos sobre la tecnología utilizada y el servidor web, lo que permite a los atacantes obtener información detallada que puede ser utilizada en su beneficio [18], [19].

8) Divulgación de IP privada

Esta vulnerabilidad de seguridad en la protección de las direcciones IP privadas permite que los atacantes obtengan la dirección IP de un sistema o dispositivo mediante una solicitud HTTP o una conexión de red [20].

9) Encabezado de respuesta del servidor HTTP

Esta alerta señala que la aplicación está filtrando información en la respuesta HTTP bajo el encabezado 'Servidor'. Aunque los datos filtrados en sí mismos no son útiles, es necesario reducir este problema por razones de seguridad [21].

10) Encabezado de seguridad de transporte estricto

Es un encabezado HTTP que permite que un servidor web indique un navegador web que solo se deben realizar solicitudes seguras HTTPS a ese servidor y que se deben evitar solicitudes HTTP no seguras. El uso de HSTS (HTTP Strict Transport Security) previene ataques como la interceptación de la red y la falsificación de certificados, ya que asegura que todas las comunicaciones con el servidor se realizarán de manera segura a través de HTTPS [22], [23].

11) Falta el encabezado X-Content-Type-Options

Esta advertencia indica la ausencia del encabezado Content-Type. Como resultado, las versiones desactualizadas de los navegadores Internet Explorer y Chrome pueden llevar a cabo un rastreo MIME, lo que les permite explorar el contenido proporcionado por el servidor web y

determinar cómo utilizarlo. Esta situación puede ser aprovechada por un atacante para realizar un ataque de secuencia de comandos [24], [25].

12) Divulgación de información - Comentarios sospechosos

Esta observación hace referencia a los comentarios presentes en el código fuente. Aunque los comentarios son útiles para que los desarrolladores trabajen de manera colaborativa, también un riesgo, ya que un atacante puede comprender la lógica de programación a través de ellos y llevar a cabo acciones maliciosas [26].

13) Aplicación web moderna

Esta advertencia indica que la aplicación del sistema académico es moderna y sugiere que sería más efectivo explorarla utilizando la herramienta Ajax Spider. Esta herramienta proporciona un enfoque avanzado para analizar la aplicación y detectar posibles problemas, lo que contribuirá a que la aplicación funcione de manera óptima [27].

14) Reexaminar las directivas de control de cache

Esta alerta señala un problema de seguridad en el que las directivas de control de caché en una aplicación web no están configuradas correctamente. Como resultado, es posible que la información confidencial, como tokens de autenticación o datos sensibles, se almacene en caché y sea accesible por personas no autorizadas. El control de caché es un mecanismo utilizado por los navegadores y servidores web para almacenar en caché páginas y recursos frecuentes, con el objetivo de acelerar la navegación [28], [29].

15) Fuzzer de agente de usuario

Esta advertencia destaca que un atacante puede emplear un fuzzer para enviar información inválida o mal formada al campo "Agente de usuario" en la cabecera HTTP de una aplicación. Este campo identifica el tipo de navegador o la aplicación que solicita la página. El uso de un fuzzer de agente de usuario permite enviar información inválida o maliciosa, lo cual puede ocasionar errores y explotar potencialmente vulnerabilidades en la aplicación [30].

Fase 3: Reporte y resumen de análisis

Se presenta un informe completo y un resumen del análisis de vulnerabilidades realizado mediante la herramienta OWASP ZAP (ver Fig: 3 y 4). El documento proporciona una descripción detallada de las vulnerabilidades identificadas en el sistema académico durante el análisis de seguridad. El informe incluye un resumen general de cada parámetro evaluado de cada vulnerabilidad, lo cual brinda una comprensión clara y concisa de los riesgos y debilidades presentes en la aplicación (ver Tabla 1):

C. Fase 3: Reporte y resumen de análisis

Se presenta un informe completo y un resumen del análisis de vulnerabilidades realizado mediante la herramienta OWASP ZAP (ver Fig: 3 y 4). El documento proporciona una descripción detallada de las vulnerabilidades identificadas en el sistema académico durante el análisis de seguridad. El informe incluye un resumen general de cada parámetro evaluado de cada vulnerabilidad, lo cual brinda una comprensión clara y concisa de los riesgos y debilidades presentes en la aplicación (ver Tabla 1):

Tipo de alerta	Riesgo	Contar
Ausencia de fichas Anti-CSRF	medio	14 (93,3 %)
Encabezado de política de seguridad de contenido (CSP) no establecido	medio	4 (26,7 %)
Falta el encabezado antisecuestro de clics	medio	4 (26,7 %)
Biblioteca JS vulnerable	medio	4 (26,7 %)
Cookie sin atributo SameSite	bajos	37 (246,7 %)
Divulgación de la marca de hora - Unix	Bajo	1 (6,7 %)
El servidor divulga información mediante un campo(s) de encabezado de respuesta	Bajo	37 (246,7 %)
Total		15

Fig. 3. *Reporte de escaneo*

Private IP Disclosure	Bajo	1 (6,7 %)
Server Leaks Version Information via "Server" HTTP Response Header Field	Bajo	37 (246,7 %)
Strict-Transport-Security Header Not Set	Bajo	37 (246,7 %)
X-Content-Type-Options Header Missing	Bajo	37 (246,7 %)
Divulgación de información - Comentarios sospechosos	Informativo	43 (286,7 %)
Modern Web Application	Informativo	8 (53,3 %)
Re-examine Cache-control Directives	Informativo	2 (13,3 %)
User Agent Fuzzer	Informativo	348 (2.320,0 %)
Total		15

Fig. 4. *Reporte de escaneo*

TABLA I: *Ausencia de fichas (tokens) Anti-CSRF*

Detalles	Descripción
Identificación de alerta	10202
Tipo de alerta	Pasiva
CWE ID	352 – Falsificación de solicitudes entre sitios (CSRF)
Nivel de Riesgo	Medio
Fase del ciclo de vida del Software	Arquitectura y diseño
OWASP Top 10 2021	A01:2021 – Pérdida de control de acceso

D. Fase 4: Propuesta de mejores prácticas

Los resultados muestran la propuesta de prácticas recomendadas basadas en las vulnerabilidades encontradas en el sistema académico.

IV. Pruebas de estrés

Para realizar las pruebas de estrés en el sistema académico se debe seguir un proceso metódico, el que se puede visualizar en la Fig. 5:

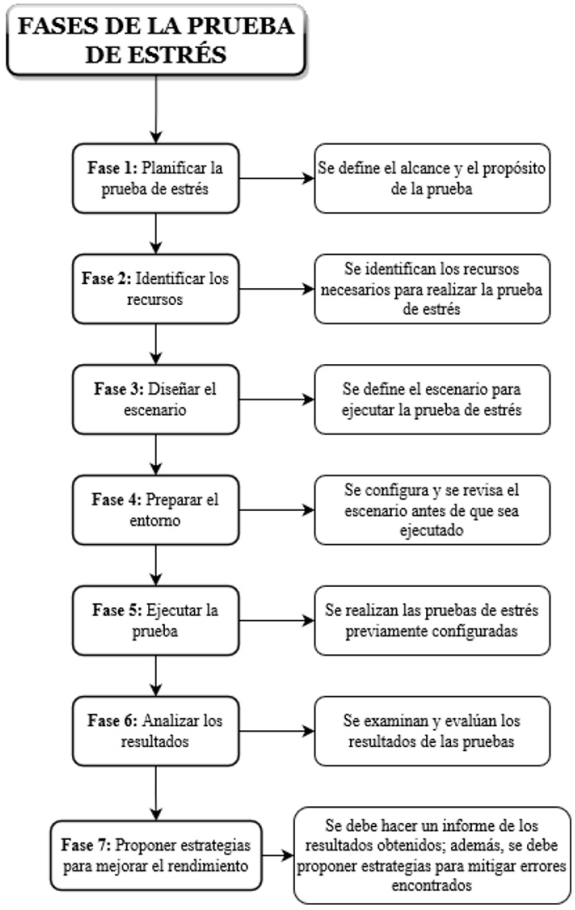

Fig. 5. *Fases para realizar una prueba de estrés*

A continuación, se describen las fases:

A. Fase 1: Planificar la prueba de estrés

El objetivo de la prueba de estrés es evaluar el rendimiento del sistema académico de la ESPOCH cuando es sometido a una gran cantidad de trabajo. Para el caso de estudio se considera el número de estudiantes en el período de matrículas en cinco años, porque se toma en cuenta que, la vida útil del sistema es de cinco años aproximadamente, lo que conlleva que la cantidad de estudiantes aumente, provocando un aumento de peticiones al sistema en el período de matrículas.

El sistema académico de la ESPOCH es una aplicación basada en el patrón de Single Page-Application (SPA), para el diseño la página se utiliza el Framework Angular y JavaScript para cargar el contenido de la vista. Para el backend utiliza una arquitectura de microservicios, la base de datos es relacional gestionada por SQL Server y se conecta con el sistema a través de NodeJS

Para el proceso de matrículas el sistema cuenta con los siguientes recursos: a nivel de interfaz de usuario, el sistema académico trabaja con un nodo que tiene 8 procesadores y 12 GB en memoria RAM. Para el backend, el sistema trabaja con un servidor Ubuntu Server con 8 procesadores y 14 GB en memoria RAM; mientras que, para el servidor de la base de datos cuenta con 8 procesadores y 12 GB en memoria RAM.

B. Fase 2: Identificar los recursos

Los recursos requeridos de Harware, Software y Humanos se describen en la Tabla XIV, Tabla XV y Tabla XVI a continuación:

TABLA XIV: *Recursos de Hardware*

Cantidad	Descripción
1	ASUS ROG Strix SCAR 15 G532 Procesador: AMD Ryzen 9 Tarjeta Gráfica: RTX 3070 RAM: 32 GB Disco: SSD 1 TB
1	Lenovo Legion Y520 Procesador: Intel Core i7-7ma Tarjeta Gráfica: NVIDIA GeForce GTX 1060 RAM: 16 GB Disco: SSD 128 GB, HD 1 TB

TABLA XV: *Recursos de Software*

Cantidad	Descripción
1	OWASP ZAP, esta herramienta se usa para buscar vulnerabilidades
1	JMeter, esta herramienta se usa para realizar pruebas de estrés

TABLA XVI: *Recursos Humanos*

Cantidad	Descripción
2	Analista de Seguridad – Tester

C. Fase 3: Diseñar el escenario

Para el diseño del escenario se identificó los recursos involucrados en el proceso de matrículas (Ver Fig. 6).

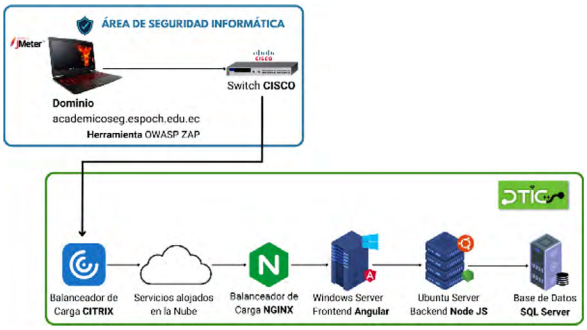

Fig. 6: *Escenario de prueba*

De igual manera, se considera que la institución hay aproximadamente 20000 estudiantes, cada semestre se abren 50 cupos por carrera, por lo que los estudiantes antes de inscribirse en la carrera pasan por un proceso de admisión y nivelación (UAN). Cada semestre, después de este proceso, a la institución ingresan aproximadamente un promedio de 35 a 40 estudiantes por carrera; cabe recalcar que la ESPOCH hay 40 carreras.

Por otra parte, se toma en cuenta el ingreso de 35 estudiantes a la institución por carrera, lo que da como promedio un aumento de 1400 estudiantes por cada semestre. No obstante, se considera la tasa de deserción, que es el 35% del total de los estudiantes que ingresan al primer semestre en la ESPOCH; es decir, 490 estudiantes que deciden retirarse de la carrera. Asimismo, se debe tomar en cuenta que la tasa de titulación es del 17%; es decir, del total de estudiantes que ingresan 238 se titulan; el 16% son estudiantes egresados; es decir, 224 estudiantes que han culminado sus estudios sin haberse titulado [31].

Por lo tanto, si se toma en cuenta todos estos factores, la cantidad aproximada de estudiantes por semestre es de 448 estudiantes, como se describe en la Ec. 1.

$$\frac{\text{Estudiantes}}{\text{Semestre}} = \text{Total - (Deserción + Titulados + Egresados)} \quad (1)$$

$$\frac{\text{Estudiantes}}{\text{Semestre}} = 448 \text{ estudiantes}$$

Por consiguiente, se toma en cuenta un período de cinco años para que el sistema cumpla su vida útil. Se considera una cantidad inicial de 20000 estudiantes y un incremento de 448 estudiantes por semestre, en cinco años habrá un total de 24480 estudiantes.

Asimismo, se toma en cuenta las horas pico en donde hay una gran afluencia de estudiantes en el período de matrículas. Según la entrevista realizada al Ingeniero Anibal Herrera, encargado del área de Desarrollo de DTIC's, mencionó que, a nivel institucional, las horas en donde hay más estudiantes ingresando al sistema son de 6 a 9 am durante este período, el número de estudiantes conectados es de 4000 a 5000 aproximadamente durante ese lapso.

Por otra parte, los estudiantes pueden matricularse en un lapso de cinco días; es decir, se considera solo una semana y, en esta, los días laborales. No obstante, se toma en cuenta solo los días: lunes, martes y miércoles, porque en estos se ha observado que hay muchos estudiantes conectados; mientras que, en los días restantes los estudiantes se conectan casualmente. Finalmente, para saber qué cantidad de estudiantes se conecta por minuto, se estima que son 5000 estudiantes conectados durante 3 horas; durante los tres días serían 15000 estudiantes conectados en un lapso de 9 horas. Para conocer la cantidad de estudiantes conectados en ese lapso en cinco años, se utiliza una proporción directa como se muestra en la Ec. 2.

$$\frac{\text{Estudiantes}}{\text{Semestre}} = 448 \text{ estudiantes}$$

$$\frac{\#\text{Estudiantes conectados (En este periodo)}}{\text{Total de estudiantes (En este periodo)}} = \frac{\#\text{ Estudiantes conectados (5 años)}}{\text{Total de estudiantes en 5 años}} \quad (2)$$

$$\frac{15000}{20000} = \frac{18360}{24480}$$

Se conoce que el número de estudiantes conectados en el período de matrículas son de 18360 en un lapso de 9 horas en total. Para saber cuántos estudiantes hay por minuto, se aplica una proporción directa, tomando en cuenta que, 1 hora tiene 60 minutos. Esto se visualiza en la Ec. 3.

$$\frac{18360 \text{ estudiantes}}{540 \text{ minutos}} = \frac{34 \text{ estudiantes}}{1 \text{ minuto}} \quad (3)$$

En función a los cálculos realizados, se concluye que hay 26 estudiantes conectados por minuto; se considera que es un número ideal de estudiantes

conectados sin que el sistema colapse. Teniendo en cuenta el valor del nivel básico, se considera dos niveles más, el nivel medio y alto de número de estudiantes conectados por minuto. Para hacer los cálculos correspondientes, se considera que en el nivel medio se toma en consideración el promedio del número de estudiantes en el nivel básico y el total, considerando el tiempo del nivel básico, como se puede visualizar en la Ec. 4

$$\text{Promedio} = 21420 \text{ estudiantes}$$

$$\frac{21420 \text{ estudiantes}}{540 \text{ minutos}} = \frac{40 \text{ estudiantes}}{1 \text{ minuto}}$$

Mientras que, en el nivel alto, se toma el número total de estudiantes considerando las 9 horas, como se observa en la Ec. 5.

$$\frac{24480 \text{ estudiantes}}{540 \text{ minutos}} = \frac{46 \text{ estudiantes}}{1 \text{ minuto}}$$

En función a los cálculos correspondientes se obtiene lo siguiente, en el nivel básico son 34 estudiantes por minuto, en el nivel medio son 40 estudiantes por minuto y en el nivel alto son 46 estudiantes por minuto. Se debe considerar que el número de estudiantes considerados en cada nivel son el número de hilos por cada minuto en las pruebas de estrés.

D. Preparar el entorno de prueba

Para preparar el escenario de pruebas, se debe comenzar a configurar el Proxy. Como se visualiza en la Figura 4, para poder hacer la prueba, se debe configurar manualmente el Proxy, para que haya un intermediario entre el cliente y el servidor. En este caso, se utiliza Firefox como navegador para poder configurar el Proxy, como se observa para HTTP y para HTTPS el acceso es local y el puerto utilizado es el 8888.

Para continuar, debe configurar en la herramienta JMeter. Primero, se configura el Servidor Proxy HTTP de la siguiente manera:

- Puerto: el puerto 8888, como se configuró en Firefox.
- Controlador Objetivo: este campo guarda la información de las peticiones que se hacen en el proceso de matrículas.
- Nombre de la Transacción: para definir cada proceso que se desee hacer; es decir, el camino a seguir simulando que es un usuario en la página.

Al dar clic en Arrancar, se inicia la grabación de peticiones al servidor del sistema académico durante el proceso de matrículas de un estudiante. A continuación, se debe colocar un árbol de resultados y un resumen de reporte para poder visualizar los resultados. Para que la prueba simule el comportamiento real de uno o varios usuarios, se debe agregar un Temporizador Aleatorio Uniforme, con la finalidad de que haya un retraso de milisegundos en cada petición, dado que, la experiencia de cada uno es diferente; a uno el sistema les carga más rápido y a otros más lento.

Para ingresar los valores de demora aleatoria y constante, se debe tomar en cuenta la fórmula que utiliza JMeter. Para el caso de estudio se considera:

- Máximo retardo aleatorio: 10 000 ms.
- Desplazamiento de retraso constante: 0 ms.

El primero detendrá a las peticiones en un número variado de milisegundos de 0 a 5000. La fórmula se visualiza en la Ecuación 6, como se muestra en esta ecuación, para cada ejecución, se considera que X tomará un valor aleatorio. Si se toma un valor de 8 para X y se remplazan los valores, la ejecución aleatoria 1 se visualiza en la Ec. 6.

(6)

Ecuación aleatoria N: 0.X * Valor Random + Valor Constante

Ecuación aleatoria 1:0.8 * 5000 + 0 = 4000ms

4 segundos de retardo

Tomando en cuenta estos parámetros, el máximo retardo aleatorio es de 5000.0 segundos y el desplazamiento de retraso es de 0 segundos. Finalmente, para configurar el grupo de hilos, se debe tener en cuenta que, cada hilo representa el proceso que sigue un estudiante al momento de matricularse en el sistema académico; es decir, cada hilo es un usuario conectado. Para la prueba de estrés considera tres niveles para el número de hilos, simulando los estudiantes conectados durante un periodo de tiempo.

- **Básico:** 34 hilos/60 segundos
- **Medio:** 40 hilos/60 segundos
- **Alto:** 46 hilos/60 segundos

- **Número de Hilos:** Es la cantidad de procesos en un intervalo de tiempo. Como se observa en la Figura 8, el número de hilos es 34.
- **Periodo de subida:** Es la cantidad de tiempo para que se ejecute cada una de las peticiones de un hilo simultáneamente. Como se observa en la Figura 8, se ejecutará 34 hilos cada 60 segundos.
- **Contador del bucle:** Es la cantidad de veces que se va a repetir el proceso, el proceso se ejecutará 180 veces para simular 3 horas en un día.

Para el nivel medio y el nivel alto, la configuración del entorno de prueba es similar. Una vez diseñado el entorno de las pruebas de estrés, se debe ejecutar tres veces cada prueba para simular los tres días de mayor afluencia de los tres estudiantes.

V. Resultados

En el presente trabajo se enfoca en los resultados obtenidos a través del análisis exhaustivo de vulnerabilidades en el sistema académico de la institución. Se identificaron 15 vulnerabilidades que representan riesgos importantes para la seguridad de la información y la integridad.

A. Ausencia de fichas (tokens) Anti-CSRF.

Se detectó que el sistema no utiliza tokens Anti-CSRF para prevenir ataques CSRF (Cross-Site Request Forgery). Para evitar este tipo de vulnerabilidad, se recomienda implementar tokens Anti-CSRF en todas las solicitudes que modifiquen los datos del usuario.

B. Encabezado de política de seguridad de contenido (CSP) no establecido

El sistema no establece una política de seguridad de contenido (CSP), lo que aumenta el riesgo de ataques XSS (Cross-Site Scripting). Se recomienda implementar una política CSP adecuada para evitar este tipo de vulnerabilidad.

C. Falta el encabezado antisecuestro de clics

Se descubrió que el sistema académico no implementa el encabezado antisecuestro de clics (CHP), lo que aumenta el riesgo de ataques de secuestro de clics. Para prevenir este tipo de vulnerabilidad, se recomienda implementar el encabezado CHP.

D. Biblioteca JS vulnerable

Para prevenir este tipo de vulnerabilidad, se recomienda actualizar la biblioteca JS a la versión más reciente que no tenga vulnerabilidades conocidas.

E. Cookie sin atributo SameSite

Se recomienda establecer el atributo SameSite en todas las cookies del sistema académico para prevenir este tipo de vulnerabilidad.

F. Divulgación de la marca de hora – Unix

Para prevenir este tipo de vulnerabilidad, se recomienda eliminar la marca de tiempo de Unix de las URL del sistema académico.

G. El servidor divulga información mediante un campo(s) de encabezado de respuesta HTTP "X-Powered-By"

Se discute la importancia de limitar la información expuesta a través de las respuestas HTTP para reducir la superficie de ataque.

H. Divulgación de IP privada

Se encontró que el sistema exponía información sobre la dirección IP privada del servidor. Se propone la implementación de medidas de seguridad para limitar el acceso a esta información.

I. Encabezado de respuesta del servidor HTTP

Se identificó que el sistema no está configurado para ocultar la información sobre el servidor web utilizado, lo que puede ser utilizado por atacantes para identificar vulnerabilidades específicas del software utilizado. Se discuten las implicaciones de esta vulnerabilidad y se proponen medidas para ocultar la información sobre el servidor.

J. Cookie sin atributo SameSite

Encabezado de seguridad de transporte estricto: se detectó que el sistema no utiliza el encabezado de seguridad de transporte estricto, lo que puede permitir a los atacantes interceptar las conexiones y robar información confidencial. Se discute la importancia de implementar este encabezado para mejorar la seguridad de las comunicaciones en línea.

K. Falta el encabezado X-Content-Type-Options

Esta vulnerabilidad se refiere a la falta de un encabezado HTTP de seguridad importante, que ayuda a prevenir ataques de tipo MIME sniffing. La solución para esta vulnerabilidad es agregar el encabezado X-Content-Type-Options en la respuesta HTTP.

L. Divulgación de información - Comentarios sospechosos

La solución para esta vulnerabilidad es revisar y eliminar cualquier comentario sospechoso en el código fuente.

M. Aplicación web moderna

La solución para esta vulnerabilidad es implementar controles de seguridad adecuados para mitigar estos riesgos.

N. Reexaminar las directivas de control de caché

La solución para esta vulnerabilidad es revisar y ajustar las directivas de control de caché de acuerdo con las necesidades de seguridad de la aplicación web.

O. Fuzzer de agente de usuario

Esta vulnerabilidad se refiere a la falta de pruebas adecuadas para detectar vulnerabilidades de seguridad en la aplicación web en relación con el comportamiento de diferentes navegadores y dispositivos. La solución para esta vulnerabilidad es utilizar herramientas de pruebas de fuzzer de agente de usuario para identificar y corregir posibles vulnerabilidades relacionadas con el comportamiento de diferentes navegadores y dispositivos.

P. Pruebas de estrés ejecutadas en función a los tres niveles

Las pruebas de estrés del sistema académico se realizaron para evaluar su capacidad de respuesta bajo cargas de trabajo pesado y determinar si el sistema puede mantener su rendimiento y disponibilidad en estas condiciones. Para las pruebas, se generaron múltiples escenarios en función de tres niveles, que simularon la actividad de los usuarios en el sistema, durante el proceso de matriculación.

1. Análisis de resultados de las pruebas de estrés en función del nivel básico

TABLA XVII: *Resultados de la ejecución de la prueba de estrés en nivel básico*

Parámetros	1ra	2da	3ra	Promedio
# Procesos	6120	6120	6120	6120
# Peticiones	116.280	116.280	116.280	116.280
Error/pet	0.75%	1.21%	0.66%	0.87%
Rendimiento (pet/min)	720	726	729	725

1) El análisis de los resultados presentados en la Tabla XVII en función a las tres ejecuciones es:

- El número de peticiones que se enviaron en total al servidor es de 116.280.
- El error en función de la cantidad de peticiones en las tres ejecuciones es 0.87% en promedio.
- El rendimiento del sistema académico es de 725 peticiones por minuto en promedio.
- El rendimiento varía de acorde con los errores que se presentan en las peticiones y la cantidad de milisegundos de retraso en cada petición.
- Las peticiones que presentaban un porcentaje de error relativamente alto se presentan en la siguiente tabla.

TABLA XVIII: *Porcentaje de error del total de peticiones por cada URL*

Transacción – URL de la petición	Porcentaje de error
Ingresar – /d7f86710-01e1-461d-8599-758de4542e2b/oauth2/authorize-614	0.14%
Ingresar – /swSistemaAcademico/seguridad/obtenerkey-626	0.26%
Salir – /common/oauth2/logout-672	0.49%

2) Análisis de resultados de las pruebas de estrés en función del nivel medio

TABLA XIX: *Resultados de la ejecución de la prueba de estrés en nivel medio*

Parámetros	1ra	2da	3ra	Promedio
# Procesos	7200	7200	7200	7200
# Peticiones	136.800	136.800	136.800	136.800
Error/pet	0.46%	0.78%	0.44%	0.56%
Rendimiento (pet/min)	872	835	865	857

El análisis de los resultados presentados en la Tabla XIX en función a las tres ejecuciones es:

- El número de peticiones que se enviaron en total al servidor es de 136.800.
- El error en función de la cantidad de peticiones en promedio de las tres ejecuciones es 0.56%.
- El rendimiento del sistema académico es en promedio de 857 peticiones por minuto.
- El rendimiento varía de acorde con los errores que se presentan en las peticiones y la cantidad de milisegundos de retraso en cada petición.
- Las peticiones que presentaban un porcentaje de error relativamente alto en comparación con las otras se muestran en la Tabla XX.

TABLA XX: *Porcentaje de error del total de peticiones por cada URL*

Transacción – URL de la petición	Porcentaje de error
Ingresar – /d7f86710-01e1-461d-8599-758de4542e2b/oauth2/authorize-614	0.10%
Ingresar – /swSistemaAcademico/seguridad/obtenerkey-626	0.28%
Salir – /common/oauth2/logout-672	0.12%

3) Análisis de resultados de las pruebas de estrés en función del nivel alto

TABLA XXI: *Resultados de la ejecución de la prueba de estrés en nivel alto.*

Parámetros	1ra	2da	3ra	Promedio
# Procesos	8280	8280	8280	8280
# Peticiones	157.320	157.320	157.320	157.320
Error/pet	1.73%	0.32%	0.32%	0.79%
Rendimiento (pet/min)	995	996	996	996

El análisis de los resultados presentados en la Tabla XXI en función a las tres ejecuciones es:

- El número de peticiones que se enviaron en total al servidor es de 157.320.
- El error en función de la cantidad de peticiones en promedio de las tres ejecuciones es 0.79%.
- El rendimiento del sistema académico es en promedio de 996 peticiones por minuto.
- El rendimiento varía de acorde con los errores que se presentan en las peticiones y la cantidad de milisegundos de retraso en cada petición.
- Las peticiones que presentaban un porcentaje de error relativamente alto en comparación con las otras se muestran en la Tabla XXII.

TABLA XXII: *Porcentaje de error del total de peticiones por cada URL*

Transacción – URL de la petición	Porcentaje de error
Ingresar – /d7f86710-01e1-461d-8599-758de4542e2b/oauth2/authorize-614	0.03%
Ingresar – /swSistemaAcademico/seguridad/obtenerkey-626	0.21%
Salir – /common/oauth2/logout-672	0.12%
Salir – Salir - /cas/logout-673	0.58%

Q. Recomendaciones en base a las ejecuciones de las pruebas de estrés.

Se recomienda implementar un nodo de respaldo adicional con 8 procesadores y 12 GB de memoria RAM para cada uno de los procesos del sistema académico. Esta medida tiene como objetivo evitar posibles colapsos del sistema y asegurar su continuidad operativa. Además, se recomienda revisar detalladamente las solicitudes que presentan un porcentaje de error considerable en el sistema académico. Esto se debe a que a medida que aumentan los errores en las solicitudes, se ve comprometido el rendimiento general del sistema. Por lo tanto, es importante identificar y solucionar las causas de estos errores para mejorar la eficiencia y el rendimiento del sistema académico.

VI. Conclusiones

En este estudio, se evaluaron las características de la metodología OWASP para la detección de vulnerabilidades y se desarrolló un procedimiento para llevar a cabo pruebas de estrés. El análisis de vulnerabilidades del sistema académico ha demostrado su alta eficiencia al identificar 15 debilidades, las cuales 4 de

riego medio, 7 de riego bajo y 4 informativas, proporcionando una visión completa y objetiva de las vulnerabilidades existentes. Los resultados e informes obtenidos brindan un análisis detallado sobre las debilidades más críticas. Para mejorar la seguridad del sistema académico a largo plazo, se recomienda implementar las mejores prácticas propuestas en base a estos hallazgos. Estas prácticas permitirán abordar de manera efectiva las debilidades identificadas, fortaleciendo la seguridad del sistema y protegiendo la información confidencial de manera más robusta.

Se realizaron múltiples pruebas de estrés en el sistema académico con el objetivo de evaluar su rendimiento durante el proceso de matrículas. Se observó que el servidor fue capaz de manejar hasta 157.320 peticiones durante 180 minutos; sin embargo, se detectaron algunos errores ocasionados por determinadas páginas web, provocando un retraso en el tiempo de respuesta del servidor, lo que generó que haya un 0.79% de error del total de peticiones. En base a estos hallazgos, se formularon recomendaciones para mejorar el rendimiento y optimizar su funcionamiento y reducir los tiempos de respuesta. Al implementar estas mejoras, se espera que el rendimiento general del sistema académico mejore significativamente durante el proceso de matrículas, brindando una experiencia más eficiente a los usuarios.

VII. Referencias

[1] F. E. Arévalo Cordovilla, I. B. Ordoñez Sigcho, M. F. Peñaherrera Larenas, y V. J. Suárez Matamoros, «Importancia de la seguridad de los sistemas de información frente el abuso, error y hurto de información», Rev. Científica Dominio Las Cienc., vol. 6, n.o 2, p. 12, jun. 2020, doi: http://dx.doi.org/10.23857/dc.v6i2.1197.

[2] R. Andrian y A. Fauzi, «Security Scanner For Web Applications Case Study: Learning Management System», J. Online Inform., vol. 4, n.o 2, p. 63, feb. 2020, doi: 10.15575/join.v4i2.394.

[3] The OWASP Foundation, Web Security Testing Guide, 4.2. The OWASP Foundation, 2020.

[4] S. Pradeep y Y. Kumar Sharma, «A Pragmatic Evaluation of Stress and Performance Testing Technologies for Web Based Applications», en 2019 Amity International Conference on Artificial Intelligence (AICAI), Dubai, United Arab Emirates: IEEE, feb. 2019, pp. 399-403. doi: 10.1109/AICAI.2019.8701327.

[5] V. P. Agila Tinoco, «ANÁLISIS DE VULNERABILIDADES, AMENAZAS Y RIESGOS AL SISTEMA DE MATRICULACIÓN DE LA UNIDAD ACADÉMICA DE CIENCIAS EMPRESARIALES DE LA UTMACH», Universidad Técnica de Machala, Machala, ago. 2019.

[6] Z. M. Jiang y A. E. Hassan, «A Survey on Load Testing of Large-Scale Software Systems», IEEE Trans. Softw. Eng., vol. 41, n.o 11, pp. 1091-1118, nov. 2019, doi: 10.1109/TSE.2015.2445340.

[7] Anibal Herrera, «Entrevista para conocer la arquitectura del nuevo Sistema Académico de la ESPOCH», 13 de octubre de 2022.

[8] D. Kornienko, S. Mishina, y M. Melnikov, «The Single Page Application architecture when developing secure Web services - IOPscience», J. Phys. Conf. Ser., p. 13, 2021, doi: 10.1088/1742-6596/2091/1/012065.

[9] The MITRE Corporation, «CWE-352: Cross-Site Request Forgery (CSRF)», CWE - Common Weakness Enumeration, 31 de enero de 2023. https://cwe.mitre.org/data/definitions/352.html (accedido 31 de enero de 2023).

[10] the ZAP Dev Team, «Absence of Anti-CSRF Tokens», OWASP ZAP, 2023. https://www.zaproxy.org/docs/alerts/10202/ (accedido 29 de enero de 2023).

[11] The MITRE Corporation, «CWE-693: Protection Mechanism Failure», CWE - Common Weakness Enumeration, 31 de enero de 2023. https://cwe.mitre.org/data/definitions/693.html (accedido 31 de enero de 2023).

[12] the ZAP Dev Team, «Content Security Policy (CSP) Header Not Set», OWASP ZAP, 2023. https://www.zaproxy.org/docs/alerts/10038/ (accedido 2 de febrero de 2023).

[13] OWASP Foundation, Inc., «Clickjacking», OWASP Foundation, 2023. https://owasp.org/www-community/attacks/Clickjacking (accedido 3 de febrero de 2023).

[14] The MITRE Corporation, «CWE-1021: Improper Restriction of Rendered UI Layers or Frames», CWE - Common Weakness Enumeration, 2023. https://cwe.mitre.org/data/definitions/1021.html (accedido 31 de enero de 2023).

[15] The MITRE Corporation, «CWE-829: Inclusion of Functionality from Untrusted Control Sphere», CWE - Common Weakness Enumeration, 31 de enero de 2023. https://cwe.mitre.org/data/definitions/829.html (accedido 2 de febrero de 2023).

[16] The MITRE Corporation, «CWE-1275: Sensitive Cookie with Improper SameSite Attribute», CWE - Common Weakness Enumeration, 31 de enero de 2023. https://cwe.mitre.org/data/definitions/1275.html (accedido 2 de febrero de 2023).

[17] the ZAP Dev Team, «Cookie without SameSite Attribute», OWASP ZAP, 2023. https://www.zaproxy.org/docs/alerts/10054/ (accedido 3 de febrero de 2023).

[18] The MITRE Corporation, «CWE-200: Exposure of Sensitive Information to an Unauthorized Actor», CWE - Common Weakness Enumeration, 31 de enero de 2023. https://cwe.mitre.org/data/definitions/200.html (accedido 2 de febrero de 2023).

[19] the ZAP Dev Team, «Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)», OWASP ZAP, 2023. https://www.zaproxy.org/docs/alerts/10037/ (accedido 3 de febrero de 2023).

[20] the ZAP Dev Team, «Private IP Disclosure», OWASP ZAP, 2023. https://www.zaproxy.org/docs/alerts/2/ (accedido 2 de febrero de 2023).

[21] the ZAP Dev Team, «Server Leaks its Webserver Application via "Server" HTTP Response Header Field», OWASP ZAP, 2023. https://www.zaproxy.org/docs/alerts/10036-1/ (accedido 29 de enero de 2023).

[22] ScanRepeat, «Strict-Transport-Security Header Not Set», ScanRepeat, 2020. https://scanrepeat.com/web-security-knowledge-base/strict-transport-security-header-not-set (accedido 29 de enero de 2023).

[23] the ZAP Dev Team, «Strict-Transport-Security Header», OWASP ZAP, 2023. https://www.zaproxy.org/docs/alerts/10035/ (accedido 29 de enero de 2023).

[24] ScanRepeat, «X-Content-Type-Options Header Missing», ScanRepeat, 2020. https://scanrepeat.com/web-security-knowledge-base/${'https://scanrepeat.com/' + path} (accedido 30 de enero de 2023).

[25] the ZAP Dev Team, «X-Content-Type-Options Header Missing», OWASP ZAP, 2023. https://www.zaproxy.org/docs/alerts/10021/ (accedido 30 de enero de 2023).

[26] ScanRepeat, «Information Disclosure - Suspicious Comments», ScanRepeat, 2020. https://scanrepeat.com/web-security-knowledge-base/${'https://scanrepeat.com/' + path} (accedido 30 de enero de 2023).

[27] the ZAP Dev Team, «Modern Web Application», OWASP ZAP, 2023. https://www.zaproxy.org/docs/alerts/10109/ (accedido 30 de enero de 2023).

[28] The MITRE Corporation, «CWE-525: Use of Web Browser Cache Containing Sensitive Information», CWE - Common Weakness Enumeration, 31 de enero de 2023. https://cwe.mitre.org/data/definitions/525.html (accedido 3 de febrero de 2023).

[29] the ZAP Dev Team, «Re-examine Cache-control Directives», OWASP ZAP, 2023. https://www.zaproxy.org/docs/alerts/10015/ (accedido 30 de enero de 2023).

[30] the ZAP Dev Team, «User Agent Fuzzer», OWASP ZAP, 2023. https://www.zaproxy.org/docs/alerts/10104/ (accedido 30 de enero de 2023).

[31] M.C.NoboaCevallosyD.E.CuencaObregon, «LEVANTAMIENTO Y ANÁLISIS ESTADÍSTICO DESCRIPTIVO DE LAS TASAS DE DESERCIÓN, RETENCIÓN Y TITULACIÓN DE LOS ESTUDIANTES DE LA ESPOCH EN LOS PERIODOS 2014-2020», Proyecto de Investigación, Escuela Superior Politécnica de Chimborazo, Riobamba, 2021.