ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA SOFTWARE

ANÁLISIS DE VULNERABILIDADES Y PRUEBAS DE ESTRÉS AL SISTEMA ACADÉMICO DE LA ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

Trabajo de Integración Curricular

Tipo: Proyecto Técnico

Presentado para optar el grado académico de:

INGENIERA DE SOFTWARE

AUTORAS: ERIKA MICHELLE ASTUDILLO MUÑOZ

ANDREA ELIZABETH VIZUETE ULLOA

DIRECTOR: Ing. DIEGO FERNANDO AVILA PESANTEZ Ph.D.

Riobamba – Ecuador

2023

Nosotras, ERIKA MICHELLE ASTUDILLO MUÑOZ Y ANDREA ELIZABETH VIZUETE ULLOA, declaramos que el presente Trabajo de Integración Curricular es de nuestra autoría y los resultados del mismo son auténticos. Los textos en el documento que provienen de otras fuentes están debidamente citados y referenciados.

Como autoras asumimos la responsabilidad legal y académica de los contenidos de este Trabajo de Integración Curricular; el patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo.

Riobamba, 09 de junio 2023

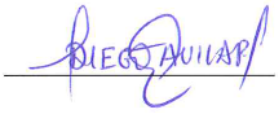
Erika Michelle Astudillo Muñoz
0105837017

Andrea Elizabeth Vizuete Ulloa
0705060614

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA SOFTWARE

El Tribunal del Trabajo de Integración Curricular certifica que: El Trabajo de Integración Curricular; tipo Proyecto Técnico **ANÁLISIS DE VULNERABILIDADES Y PRUEBAS DE ESTRÉS AL SISTEMA ACADÉMICO DE LA ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO**, realizado por las señoritas: **ERIKA MICHELLE ASTUDILLO MUÑOZ Y ANDREA ELIZABETH VIZUETE ULLOA**, ha sido minuciosamente revisado por los Miembros del Trabajo de Integración Curricular, el mismo que cumple con los requisitos científicos, técnicos, legales, en tal virtud el Tribunal Autoriza su presentación.

	FIRMA	FECHA
Ing. Miguel Ángel Duque Vaca Mgs. **PRESIDENTE DEL TRIBUNAL**		2023-06-09
Ing. Diego Fernando Avila Pesantez Ph. D. **DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR**		2023-06-09
Ing. Danilo Mauricio Pastor Ramirez Ph. D. **ASESOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR**		2023-06-09

DEDICATORIA

Dedico el presente trabajo de integración curricular a mis padres Juan Astudillo y Gladys Muñoz porque me han apoyado incondicionalmente y han estado presentes a pesar de mis equivocaciones, gracias por enseñarme a afrontar las dificultades de la vida con sabiduría y paciencia. De la misma manera, gracias a mi novio Marco Rodríguez por brindarme su paciencia, su comprensión, su apoyo y su amor incondicional, él ha sido mi pilar y mi fortaleza en todo momento, lo amo y lo admiro profundamente. A mi amiga Andrea Vizuete porque más que una compañera, ha sido una amiga incondicional y paciente en todo momento, hemos pasado momentos difíciles en la carrera y, a pesar de ello, hemos podido salir adelante. Finalmente, pero no menos importante, dedico mi tesis a mis perritas Chula, Rufis, Preciosa, Juliet y a mi gatita Michi que han estado presentes en mis triunfos y en mis derrotas, y las tengo presentes en mi corazón, especialmente a las que partieron hacia un lugar mejor.

Michelle

Dedico este trabajo a mis padres y hermana María Ulloa, Francisco Vizuete y Lisbeth con mucho cariño y gratitud, por su apoyo incondicional y su constante motivación, tan indispensable para llegar hasta aquí, gracias por ser el pilar e inspiración en los momentos más difíciles y poder celebrar este importante logro. A mis amigos Leonardo, José, Michelle, Erika y Josué por ser parte tan importante de mi vida, siempre han estado en las buenas y en las malas, así como el apoyo incondicional que me han brindado durante todo el transcurso de la carrera. A mi abuelita por alentarme siempre a seguir adelante, y con todo mi amor y cariño a mi Zeus quien ha sido la fuente de mi dedicación y esfuerzo a lo largo de mi carrera hacia esta meta.

Andrea

AGRADECIMIENTO

Con un corazón lleno de gratitud se agradece:

A nuestra querida institución la Escuela Superior Politécnica de Chimborazo y a nuestros docentes que en la trayectoria de nuestra carrera han jugado un papel crucial en cuanto a nuestra formación estudiantil y profesional, estamos muy profundamente agradecidas por su contribución.

Queremos agradecer a nuestro tutor y asesor el Ing. Diego Ávila y Ing. Danilo Pastor, por su guía y sabiduría, gracias por compartir sus conocimientos, ayudarnos con su experiencia en el desarrollo de este trabajo, por el apoyo y paciencia que han sido elementos muy valiosos y apreciados.

Finalmente, queremos agradecer a nuestros familiares y amigos por su amor, apoyo incondicional y por ser fuente de motivación y estar en nuestros momentos más difíciles. Estamos muy profundamente agradecidas con todas las personas que nos han apoyado y nos han ayudado durante nuestra carrera y sobre todo durante el desarrollo de este trabajo, gracias.

Michelle, Andrea

ÍNDICE DE CONTENIDO

CAPÍTULO III

CAPÍTULO IV

CAPÍTULO V

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

RESUMEN

El objetivo del presente trabajo de integración curricular fue realizar análisis de vulnerabilidades y ejecutar pruebas de estrés al sistema académico de la Escuela Superior Politécnica de Chimborazo (ESPOCH) para mejorar la seguridad y su rendimiento. Se utilizó el método descriptivo y se revisó varios documentos fiables para conocer las características de la metodología Open Web Application Security Project (OWASP), que permitieron definir las fases del análisis de vulnerabilidades y las etapas de las pruebas de estrés. Para la detección de vulnerabilidades se utilizó la herramienta OWASP ZAP; mientras que, para ejecutar las pruebas de estrés se utilizó Apache JMeter. Finalmente, se hizo una revisión exhaustiva de documentación confiable para proponer las mejores prácticas que permitan mitigar las vulnerabilidades encontradas y mejorar la seguridad del sistema académico. De igual manera, se efectuó un análisis de resultados de las pruebas de estrés, las que mostraron que el sistema académico responde adecuadamente pese a la cantidad de peticiones enviadas; sin embargo, no se consideraron todas las peticiones que se generan en el proceso de matrículas, de lo contrario el sistema colapsaría. Asimismo, se recomienda aumentar un nodo de respaldo para cada módulo del sistema académico. Se concluye que es importante aplicar las mejores prácticas y las estrategias propuestas en el presente trabajo para mejorar la seguridad y el rendimiento del sistema académico institucional. Se recomienda realizar análisis de vulnerabilidades periódicamente para verificar la seguridad del sistema

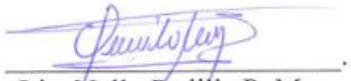
Palabras clave: <SEGURIDAD INFORMÁTICA>**,** <ANÁLISIS DE VULNERABILIDADES>, <METODOLOGÍA OWASP>, <PRUEBAS DE ESTRÉS>, <OWASP ZAP (SOFTWARE)>, <APACHE JMETER (SOFTWARE)>, <SISTEMA ACADÉMICO DE LA ESPOCH>

1120-DBRA-UPT-2023

ABSTRACT

The objective of this curricular integration work was to carry out vulnerability analysis and execute stress tests to the academic system of the Escuela Superior Politécnica de Chimborazo (ESPOCH) to improve security and performance. The descriptive method was used and several reliable documents were reviewed to know the characteristics of the Open Web Application Security Project (OWASP) methodology, which allowed defining the phases of vulnerability analysis and the stages of stress tests. The OWASP ZAP tool was used to detect vulnerabilities, while Apache JMeter was used to execute the stress tests. Finally, an exhaustive review of reliable documentation was carried out to propose the best practices that allow mitigating the vulnerabilities found and improving the security of the academic system. Likewise, an analysis of the results of the stress tests was carried out, which showed that the academic system responds adequately despite the number of requests sent, however, not all the requests that are generated in the enrollment process were considered. otherwise the system would collapse. Furthermore, it is recommended to increase a backup node for each module of the academic system. It is concluded that it is important to apply the best practices and strategies proposed in this paper to improve the security and performance of the institutional academic system. It is recommended to perform vulnerability scans periodically to verify the security of the system.

Keywords: <COMPUTER SECURITY>, <ANALYSIS OF VULNERABILITIES>, <OWASP METHODOLOGY>, <STRESS TESTING>, <OWASP ZAP (SOFTWARE)>, <APACHE JMETER (SOFTWARE)>, <ACADEMIC SYSTEM>



Lic. Nelly Padilla P. Mgs.
0603818717
DOCENTE FIE

1120-DBRA-UPT-2023

INTRODUCCIÓN

En la actualidad, el análisis de vulnerabilidades y pruebas de estrés son dos técnicas importantes para garantizar la seguridad y estabilidad de un sistema académico universitario. El análisis de vulnerabilidades permite identificar y corregir las debilidades del sistema que podrían ser explotadas por atacantes externo o internos, mientras que las pruebas de estrés ayudan a evaluar la capacidad del sistema para manejar situaciones de alta demanda o sobrecarga. En conjunto, estas técnicas permiten fortalecer la confidencialidad, integridad y disponibilidad de los datos.
El nuevo sistema académico que está desarrollando la Escuela Superior Politécnica de Chimborazo (ESPOCH) incluye proceso de matrículas, aprobación de exámenes, visualización de calificaciones, entre otros. Por lo que, es necesario realizar una serie de pruebas a este sistema, utilizando la metodología OWASP que se enfoca en mejorar la seguridad de las aplicaciones web, y se utiliza ampliamente en todo el mundo.

En este trabajo, se identifican las herramientas adecuadas para detectar las vulnerabilidades y se proponen las políticas de seguridad para reducir estos riesgos. Finalmente, se presenta una propuesta para mejorar el rendimiento y la seguridad del sistema académico de la institución.

CAPÍTULO I

1. DIAGNÓSTICO DEL PROBLEMA

En la presente sección se analizan los antecedentes del trabajo, la formulación del problema, justificación y los objetivos planteados.

1.1. Antecedentes

La Escuela Superior Politécnica de Chimborazo (ESPOCH) es una institución educativa de nivel superior en Ecuador, que cuenta con diversos sistemas informáticos para el apoyo a la gestión académica, administrativa y de investigación. Debido a la creciente cantidad de datos y la complejidad de las aplicaciones informáticas utilizadas, el análisis de vulnerabilidades y pruebas de estrés de han vuelto cada vez más importantes para garantizar la seguridad y el buen funcionamiento de los sistemas.

En el año 2019, se realizó un análisis de vulnerabilidades en el sistema académico de la ESPOCH, con el objetivo de identificar y corregir las posibles brechas de seguridad que pudieran poner en riesgo la integridad de los datos y la privacidad de los usuarios. Se utilizaron herramientas especializadas para la detección de vulnerabilidades, y se utilizaron pruebas de penetración para simular los ataques que un hacker podría realizar. Como resultado del análisis, se identificaron varias vulnerabilidades en el sistema académico, incluyendo fallas en la autenticación y autorización de usuarios, exposición de información sensible en páginas web, y falta de cifrado en la transmisión de datos. Se corrigieron las vulnerabilidades detectadas y se implementaron medidas adicionales de seguridad, como la implementación de certificados SSL para la transmisión segura de datos (Arias Paredes, 2019, p. 19).

En este sentido, se han realizado varios estudios de vulnerabilidades en los sistemas académicos universidades del Ecuador y Argentina. Por ejemplo, se realizó un análisis de vulnerabilidades, amenazas y riesgos al sistema de matriculación de la unidad académica de ciencias empresariales de la UTMACH, que tiene como finalidad mitigar las vulnerabilidades encontradas a través de una metodología descriptiva (Agila Tinoco, 2019). 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 (Cuevas et al., 2018).

En el trabajo de Jiang y Hassan (2019, p. 1) 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.

1.2. Formulación del problema

¿Cómo se puede analizar las vulnerabilidades y realizar pruebas de estrés en el sistema académico de la ESPOCH?

1.3. Sistematización del problema

- ¿Cuáles son las características de la metodología OWASP que permite detectar vulnerabilidades en un sistema?
- ¿Cuál es el procedimiento para realizar las pruebas de estrés?
- ¿Cuáles son las herramientas que permiten analizar las vulnerabilidades que existen en el sistema académico de la ESPOCH?
- ¿Cuáles son las herramientas que permiten realizar pruebas de estrés en el sistema académico de la ESPOCH?
- ¿Qué vulnerabilidades se pueden detectar en el nuevo sistema académico?
- ¿Qué resultados se podrá evidenciar con las pruebas de estrés?

1.4. Justificación

1.4.1. Justificación teórica

OWASP es una metodología transparente, innovadora y global que permite mejorar la seguridad en las aplicaciones web. A través de esta metodología se puede detectar y prevenir las vulnerabilidades encontradas en un sistema mediante un informe detallado. En este informe se enumera 10 vulnerabilidades más comunes en las aplicaciones web, incluyendo inyección SQL, problemas de autenticación, exposición de datos sensibles, entidades extremas XML(XXE), pérdida de control de acceso, configuración de seguridad incorrecta, Cross-Site Scripting(XSS), diseño inseguro, uso de componentes con vulnerabilidades conocidas, registro y monitoreo

insuficientes, siendo estas las vulnerabilidades más comunes que se pueden detectar y corregir para mejorar la seguridad en las aplicaciones web (OWASP Foundation, Inc., 2022a).

Por otro lado, las pruebas de estrés, es un proceso cíclico con varios pasos, donde cada interacción detalla los resultados y mejoras que se aplicarán en la subsiguiente iteración. Su objetivo es evitar la degradación del rendimiento y detectar errores de código interno debido a la multitarea compleja y la optimización del sistema. En aplicaciones web, estas pruebas son especialmente importantes para detectar errores de rendimiento y permite que la Institución pueda medir sus capacidades para eventos específicos relacionados con el tiempo de respuesta. Por ejemplo, se puede crean grandes cantidades de usuarios de forma remota o denegación de servicio, para observar cuanto puede tolerar el sistema en su utilización de la plataforma para comunicarse al mismo tiempo. A través de estas pruebas, es posible examinar el rendimiento y decretar su funcionamiento a este tipo de situaciones (Simba et al., 2022).

1.4.2. Justificación aplicativa

Mediante el análisis de vulnerabilidades, es posible identificar desde problemas leves q hasta los más graves, permitiendo tomar medidas correctivas y reducir cualquier intento de intrusión que pueda afectar el sistema académico de la ESPOCH. Además, las pruebas de estrés son útiles para medir el rendimiento del sistema y verificar si es capaz de manejar la cantidad de solicitudes que se presentan durante eventos críticos, como la fase de matrículas de los estudiantes, en el cual el sistema debe responder a una sobrecarga de peticiones. En este trabajo, se realiza lo siguiente:

- Describir la metodología OWASP para llevar a cabo un estudio de vulnerabilidades en el sistema académico de la ESPOCH.
- Estudiar las herramientas adecuadas para identificar vulnerabilidades en el sistema académico de la ESPOCH.
- Proponer políticas de seguridad para mitigar las vulnerabilidades identificadas en el sistema académico de la ESPOCH.
- Realizar una propuesta para la mejora del rendimiento del sistema académico de la ESPOCH

1.5. Objetivos

1.5.1. Objetivo general

Analizar las vulnerabilidades y realizar pruebas de estrés en el sistema académico de la ESPOCH que permita mejorar la seguridad.

1.5.2. Objetivos específicos

- Describir las características de la metodología OWASP para determinar las vulnerabilidades que tiene el sistema; el procedimiento y las herramientas utilizadas para realizar las pruebas de estrés.
- Analizar las vulnerabilidades del sistema académico aplicando la metodología OWASP para conocer las más críticas.
- Realizar las pruebas de estrés al sistema académico para conocer el rendimiento y proponer estrategias para mitigar en caso de fallo.
- Proponer las mejores prácticas que permitan mejorar la seguridad del sistema académico.

CAPÍTULO II

2. MARCO TEÓRICO

El presente capítulo proporciona los antecedentes teóricos esenciales para el análisis de vulnerabilidades y pruebas de estrés en el sistema académico de la ESPOCH.

2.1. Seguridad Informática

Según Romero Castro et al. (2018, pp. 13-14), la seguridad informática es una disciplina encargada de proponer y diseñar reglas, procedimientos, métodos y técnicas para lograr que un sistema de información sea seguro, confiable y esté disponible al personal autorizado. Por otro lado, Samaniego Mena y Ponce Ordóñez (2021, p. 2) define a la seguridad informática como un conjunto de medidas que imposibilitan la ejecución de operaciones no autorizadas sobre un programa, una aplicación, un sistema o una red informática; de lo contrario, dicha acción puede ocasionar consecuencias severas, entre las cuales están la pérdida o el robo de información, se pierde la confidencialidad, autenticidad o la integridad de los datos; además, puede ocasionar la disminución del rendimiento de los equipos o la pérdida de acceso de los usuarios autorizados. En cambio, Kuthnik et al. (2018, pp. 3-4) menciona que la seguridad informática es una disciplina encargada de proponer medidas de prevención y respuestas para las organizaciones que trabajan con sistemas tecnológicos, que permite mantener la información segura, protegida y confiable. Con base en las definiciones mencionadas anteriormente, se infiere que la seguridad informática es el conjunto de procesos y medidas utilizadas para resguardar los programas, sistemas o redes informáticas. Además, es usada para mantener la información precisa, segura y disponible ante las posibles amenazas que se pueden presentar causadas por los atacantes.

Kuthnik et al. (2018, p. 2) enfatiza que los tres pilares de seguridad son: a) la confidencialidad que se encarga de que la información sea reservada y protegida ante el personal no autorizado; b) la integridad que hace referencia a asegurar que los activos sean precisos e íntegros; es decir, que no haya modificación; y c) la disponibilidad, que se encarga de que se pueda acceder al dominio siempre y cuando una entidad autorizada lo solicite.

Según Samaniego Mena y Ponce Ordóñez (2021, p.2), la seguridad informática comprende los componentes hardware, software y la red; los cuales se detallan a continuación:

- **Seguridad de hardware:** tiene como propósito proteger el hardware electrónico durante el ciclo de fabricación; es decir, durante el diseño, implementación y validación de este para asegurar su confiabilidad y funcionamiento seguro; de igual manera, se encarga de proteger la información sensible almacenada en el hardware ante posibles amenazas. Por ejemplo, el robo de información de un chip a través de la medición y el análisis de la potencia y el retardo de propagación de señal. (Bhunia y Tehranipoor, 2018, p. 6).
- **Seguridad del software:** promueve un enfoque de incorporar buenas prácticas de seguridad durante el ciclo de vida del desarrollo de un software (SDLC) con la finalidad de que sea seguro antes de ser puesto en producción o, que siga funcionando correctamente a pesar de que esté siendo atacado (Venson et al., 2019, p. 1).
- **Seguridad de la red:** es responsable de proteger la red y los datos que son transportados por ese medio manteniendo su integridad, verificando el acceso y el uso autorizado de estos. Para lograr su objetivo, combina varias capas de seguridad de red para defender el perímetro y la red; cada una implementada con políticas y controles definidos (Cisco Systems, Inc., 2022). Este trabajo trata de la seguridad del software, en la siguiente sección se da mayor detalle sobre el tema.

2.1.1. Seguridad del Software

La norma ISO/IEC 25010, evalúa la calidad de software, uno de los parámetros que considera es la seguridad, su objetivo es proteger los datos e información sensible, con la finalidad de solo las personas con autorización puedan acceder a esta y modificarla en caso que sea necesario (iso25000.com, 2022). Según Payer (2021, p. 3) el objetivo de la seguridad del software es permitir el uso del sistema al personal autorizado y denegar el acceso a las personas no autorizadas. Si hay un uso no previsto, puede causar daños al sistema y a la información albergada; por ello, es esencial desarrollar e implementar un software seguro. Mientras que, Piessens (2021, p. 6) plantea que el enfoque de la seguridad de software consiste en que el sistema cumpla con los requisitos de confidencialidad, integridad y disponibilidad, para proteger los datos e información delicada y, de igual modo, las funcionalidades del sistema. Por otra parte, McGraw (2004, p. 1) enfatiza que la finalidad de la seguridad del software es conseguir que el sistema siga funcionando adecuadamente ante un ataque malicioso. Por consiguiente, se enfatiza que la seguridad del software abarca la protección de un sistema ante amenazas generadas por un atacante, cumpliendo los tres principios de seguridad: confidencialidad, integridad y disponibilidad. El presente trabajo se limita a la seguridad de aplicaciones web, en la siguiente sección se amplía la información sobre este tema.

2.1.1.1 Seguridad en aplicaciones Web

Campderrós Vilà (2019, p. 12) define que la finalidad de la seguridad en aplicaciones web es proteger los datos e información contenida en el sistema, dado que es el activo de mayor valor en la seguridad de la información. A su vez, Andrian y Fauzi (2020, p. 63) señalan que la seguridad de aplicaciones web tiene como objetivo proteger la información sensible almacenada en el sistema a través de prácticas de desarrollo seguras, las cuales están basadas en los tres principios fundamentales de seguridad: confidencialidad, integridad y disponibilidad, conocidos como la triada CIA. Por su parte, Zech et al. (2019, p. 1) manifiestan que la seguridad juega un rol muy importante en las aplicaciones web, dado que en estas se almacenan datos sensibles del usuario, los cuales son objetivo de los atacantes; por consiguiente, se debe aplicar la triada CIA para protegerlos y evitar que sean adquiridos por personas no autorizadas. De modo que, con base en los conceptos mencionados, la seguridad en las aplicaciones web es indispensable para mantener segura la información que se encuentra almacenada en un sistema web.

La seguridad y las vulnerabilidades están relacionas, ya que las vulnerabilidades son los puntos débiles o las brechas en un sistema que pueden ser explotadas para causar daño u obtener acceso no autorizado. Por otro lado, la seguridad se refiere a las medidas implementadas para proteger un sistema de estos ataques, que busca defender un sistema de las vulnerabilidades, debilidades o brechas que pueden ser explotadas. Por lo tanto, la seguridad enfoca en identificar y minimizar las vulnerabilidades para evitar un ataque, para que una aplicación sea segura, se deben conocer las vulnerabilidades que puede poseer una aplicación. Estas se detallan a continuación.

2.2. Vulnerabilidades en aplicaciones web

Arévalo Cordovilla et al. (2020, pp. 2-5) mencionan que, en el área de seguridad informática, las vulnerabilidades son brechas de un software, las cuales son utilizadas por un atacante para acceder a los datos almacenados, quebrantando los principios de seguridad. En el diseño, las vulnerabilidades son conocidas como flaws y son consecuencia de la falta de aplicación de políticas de seguridad; mientras que, en la implementación se las conoce como bugs y, se debe a existencia de puertas traseras por defectos en la programación. A su vez, como expresa Rubín Linares (2021, p. 47), las vulnerabilidades son fallas de seguridad en las aplicaciones web, lo que permite a los atacantes robar la información privada, quebrantar la integridad de los datos o, incluso afectar la disponibilidad de la aplicación web. En cambio, Al Khurafi y Al Ahmad (2015, p. 1) afirman que las vulnerabilidades son debilidades presentes en las aplicaciones web, pueden presentar a nivel físico y lógico a consecuencia de la falta de seguridad a nivel de código o, por falta de aplicación de políticas de seguridad. En conclusión, las vulnerabilidades en las

aplicaciones web son debilidades que ocurren cuando los sistemas presentan fallas no contempladas en el desarrollo del software, ocasionando que la aplicación sea vulnerable a los ataques ocasionados por hackers, provocando la modificación o sustracción de la información almacenada en estos y perjudicando a las organizaciones, como se visualiza en la Figura 1-2. Con respecto a las vulnerabilidades, son clasificadas para orientar la gestión y corrección de éstas, las cuales se detallan a continuación.

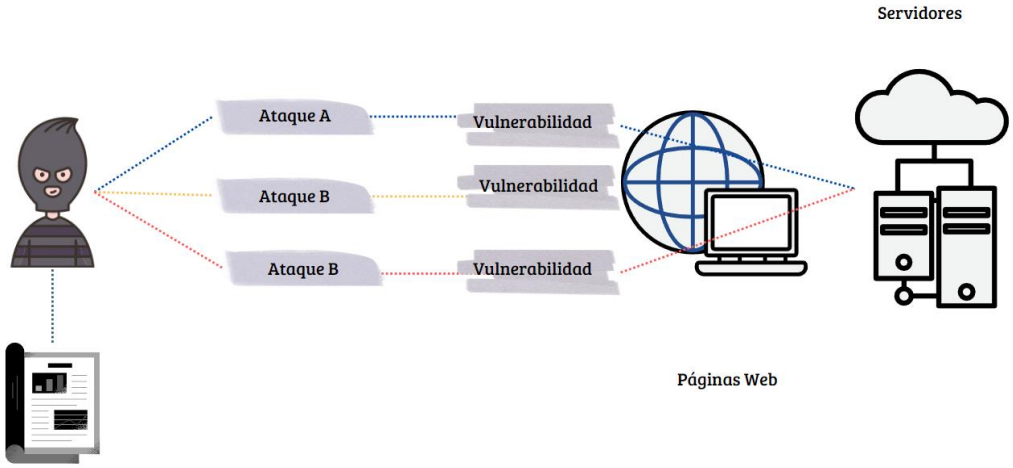


Figura 1-2: Vulnerabilidades en aplicaciones web

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

2.2.1. Clasificación de las vulnerabilidades en aplicaciones web

OWASP Top 10 team (2021b) destaca una lista que clasifica las 10 vulnerabilidades más críticas encontradas en las aplicaciones web de diversas partes del mundo. Además, es actualizada periódicamente y proporciona una visión general de las principales amenazas a los sistemas, las cuales se muestran a continuación en la Tabla 1-2.

Tabla 1-2: Clasificación de las vulnerabilidades TOP 10 2021

TOP 10 2021
A01:2021 – Pérdida de control de acceso
A02:2021 – Fallas criptográficas
A03:2021 – Inyección
A04:2021 – Diseño inseguro
A05:2021 – Configuración de seguridad incorrecta
A06:2021 - Componentes vulnerables y desactualizados
A07:2021 – Fallas de identificación y autenticación
A08:2021 – Fallas en el Software y en la integridad de los datos
A09:2021 – Fallas en el registro y monitoreo
A10:2021 – Falsificación de solicitudes del lado del servidor (SSRF)

Fuente: OWASP Top 10 team, 2021b

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

2.2.1.1. Pérdida de control de acceso

Esta vulnerabilidad hace referencia a una situación en la que el administrador no tiene el control sobre quien ingresa al sistema, a consecuencia de diversos factores, la deficiencia del control de seguridad en el software, la falta de mantenimiento o de actualización del sistema, entre otros; lo que conlleva a la exposición de información confidencial, la suspensión de servicios, entre otros daños a la aplicación (Bach Nutman, 2020, p. 2).

2.2.1.2. Fallas criptográficas

Se refiere a un error de seguridad en la protección de datos dado que, hubo dificultades en el diseño de los algoritmos de cifrado o el cifrado es obsoleto; como consecuencia, los atacantes descifran información sensible o pueden comprometer los sistemas de seguridad (Carvaca Orrala, 2022, p. 22). Existen diversos ataques utilizando esta debilidad, los más frecuentes se describen a continuación:

- **Ataque de fuerza bruta:** el atacante usa un gran número de combinaciones de claves para descifrar los datos que están protegidos (Packetlabs, 2022).
- **Ataque de texto plano seleccionado:** el atacante usa un texto plano arbitrario para compararlo con un texto codificado con la finalidad de encontrar la clave.
- **Ataque de clave y algoritmo:** los atacantes examinan el algoritmo de cifrado para encontrar la clave.

2.2.1.3. Inyección

Este ataque consiste en enviar datos no confiables a un sistema web, para que este los procese y envíe una respuesta. Existen varios tipos de ataques de inyección, los cuales se explican en la Tabla 2-2 (Pinango Bayas et al., 2022, p. 10).

Tabla 2-2: Tipos de ataques de inyección según el código que se envíe y a donde se destine

Tipos de ataques de inyección	Descripción
SQL	Consiste en introducir código SQL para obtener o modificar información en una base de datos. Se lo puede realizar de diferentes maneras, como son los ataques basados en errores que ocasionan que la base de datos falle y devuelva información; mientras que, los que son basados en uniones utilizan la cláusula UNION para dirigir distintas consultas y, finalmente, los que son a ciegas, el atacante manda diferentes queries hasta que la base de datos devuelva algún dato.

NoSQL	Consiste en introducir código malicioso en la aplicación y que realice algún proceso; para hacerlo, el atacante define el tipo de lenguaje de programación que se utilizó en la aplicación, como es PHP, Java, Ruby, entre otros.
Protocolo Ligero de Acceso a Directorios (LDAP)	Consiste en introducir código no confiable con la finalidad de modificar la capacidad de búsqueda del protocolo LDAP en la base de datos.
Xpath	Consiste en que el atacante suministre cadenas de consultas a una aplicación web, con la finalidad de que esta construya una consulta XPath para los datos XML, de tal manera que pueda extraer información delicada del sistema.
Cabeceras de Host	Consiste en inyectar un host malicioso en el encabezado del host HTTP para poder controlarlo, con la finalidad de seguir inyectando código en la aplicación web.

Fuente: Bach Nutman, 2020

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

2.2.1.4. Diseño inseguro

Hace referencia a la falta de medidas de seguridad en el diseño y arquitectura de un software, lo que ocasiona que se convierta en un defecto en la implementación, pasando a ser una vulnerabilidad en el sistema que puede ser explotada por los atacantes (OWASP Top 10 team, 2021a). Los principales ejemplos de diseño inseguro son los siguientes:

- **Falta de autenticación y autorización adecuada**: no existen controles de acceso adecuados en un sistema (OWASP Top 10 team, 2021a).
- **Falla de validaciones en las entradas**: no se validan las entradas de datos, lo que puede ocasionar que haya un ataque de inyección.
- **Falta de protección contra ataques de denegación de servicio:** un sistema que no ha sido diseñado correctamente es vulnerable a ataques de DoS.

2.2.1.5. Configuración de seguridad incorrecta

Esta vulnerabilidad se refiere a una mala configuración de seguridad en los servidores dado que, existen fallos no controlados en los componentes o en los subsistemas de un software; por ejemplo, puede ocurrir que los certificados SSL estén mal configurados, el cifrado sea inseguro, entre otros errores en el servidor que provocan que el sistema esté expuesto a ataques (Kumi et al., 2021, p. 93).

2.2.1.6. Componentes vulnerables y desactualizados

Esta debilidad se refiere a que, si los componentes están desactualizados, no son compatibles con la aplicación o, están expuestos a un exploit, son vulnerables a ataques; por tanto, si se los utiliza,

las aplicaciones que están en entornos de producción son expuestas a amenazas, las cuales pueden ser: inyección de código, desbordamiento de búfer y XSS (Carvaca Orrala, 2022, p. 23).

2.2.1.7. Fallas de identificación y autentificación

La identificación es la facultad de reconocer de manera exclusiva a un usuario dentro de un sistema que se está ejecutando; mientras que, la autenticación es la facultad de comprobar que el usuario que se encuentra en la aplicación no está siendo suplantado (IBM MQ, 2021). Si las funcionalidades de identificación o autentificación no son implementadas de manera segura, pueden producirse fallas en el sistema, las cuales son explotadas por los atacantes para robar contraseñas, claves, tokens de sesión u otros datos, con el objetivo de asumir de forma temporal o permanente las identidades de otros usuarios (Carvaca Orrala, 2022, p. 23).

2.2.1.8. Fallas en el software y en la integridad de los datos

La infraestructura y el código que no han contemplado posibles vulnerabilidades, ocasionan fallas en el software y en la integridad de los datos, los atacantes las aprovechan para explotarlas o para introducir código maligno; estos fallos pueden ocurrir cuando se utiliza software de repositorios no confiables o que han sido manipulados en su origen (Carvaca Orrala, 2022, p. 23).

2.2.1.9. Fallas en el registro y monitoreo

Cuando hay fallas en los eventos de seguridad, como no registrar los intentos de inicio de sesión en un sitio web o no supervisar de manera continua, un atacante puede explotar dichas vulnerabilidades a través de un sondeo continuo a la aplicación en busca de estas debilidades (Llamuca Quinaloa et al., 2021, p. 13).

2.2.1.10. Falsificación de solicitudes del lado del servidor (SSRF)

2.3. Metodología OWASP

El Proyecto Abierto de Seguridad de Aplicaciones Web (OWASP) es un conjunto de proyectos abiertos de seguridad para aplicaciones web que recopila métodos y procedimientos, los cuales son usados por las personas encargadas del sistema a desarrollar con el objetivo de cerciorarse de que el código del sistema es seguro y que no posee vulnerabilidades; asimismo es utilizada por los testers para realizar pruebas que son precisas para verificar que un software es confiable y, finalmente por los encargados de seguridad que están en el deber de garantizar que una aplicación es segura para poder llevarla a producción (OWASP Foundation, Inc., 2010; Zapata García, 2018, p. 26).

2.3.1. Características de la metodología OWASP

- El propósito de esta metodología es orientar a las personas que deseen testear la seguridad de un producto de software, denotar la finalidad de cada una de las pruebas y cómo se deben realizarlas, independientemente de si el programa es de su autoría o de terceros (The OWASP Foundation, 2020, p. 13).
- Gracias al gran conjunto de pruebas que describe dicha metodología, se puede garantizar la confiabilidad y seguridad del software.
- El conjunto de pruebas a realizarse hace referencia a la totalidad de posibles fallas de seguridad que pueden presentarse en una aplicación web en determinadas circunstancias. Como se mencionó anteriormente, la metodología OWASP se compone de tres módulos base:

- **Guía para Construir Aplicaciones y Servicios Web Seguros:** es una colección de procesos y buenas prácticas que se deben aplicar para desarrollar software seguro y de calidad. Los métodos que se mencionan ayudan a los programadores a prever posibles vulnerabilidades del sistema para evitar ataques de inyección SQL, suplantación de identidad, entre otros (The OWASP Foundation, 2005, p. 15; Zapata García, 2018, p. 26).
- **Guía de Revisión de Código:** es un conjunto de buenas prácticas que se deben emplear en la revisión de código para que sea seguro. La guía puede ser utilizada dentro de un ciclo de vida de desarrollo de software seguro (S-SDLC) (The OWASP Foundation, 2017, p. 7).
- **Guía de pruebas de seguridad web:** contiene una gran variedad de pruebas y de las herramientas y técnicas para ponerlas en práctica (The OWASP Foundation, 2020, p. 13).

Arya Wiradarma y Arya Sasmita (2019, p. 19) destacan que la metodología de pruebas OWASP se enfoca fuertemente en el nivel de seguridad de las aplicaciones web en todas las fases de desarrollo web, a diferencia de otras metodologías de pruebas de seguridad como ISSAF y OSSTMM, que se enfocan en la seguridad en la fase de la implementación. Mientras que, Santiago García (2021, p. 17) expresa que la metodología OWASP suministra una guía para realizar una auditoría de seguridad a una aplicación web o, en lo que respecta a realizar pruebas de seguridad, esta metodología brinda no solo la información de las pruebas sino de cómo ponerlas en práctica, de forma segura y confiable. En cambio, como lo hace notar Zapata García (2018, p. 37), uno de los beneficios de la utilización de la metodologías OWASP es el modelamiento de amenazas de un sistema, porque brinda una base para aplicar métodos de seguridad con la finalidad de mitigarlas o minimizarlas. Por tanto, gracias al análisis realizado, se utilizará la metodología OWASP, que contiene la guía de pruebas de seguridad web, la cual se utilizará en el presente trabajo. Esta guía se detalla a continuación.

2.3.2. Guía de pruebas de seguridad web de la metodología OWASP

The OWASP Foundation (2020, p. 47) menciona que la metodología de pruebas OWASP se basa en las pruebas de caja negra, en las que, la persona encargada de realizar las pruebas desconoce o tiene muy poca información sobre la aplicación a ser probada; en esta metodología se recopila todas las técnicas de comprobación posibles, explicándolas como se deben realizar y manteniendo la guía actualizada. Por otro lado, Gamboa Safla (2021, p. 20), la metodología de la Guía de Pruebas OWASP es utilizada para realizar pruebas de seguridad durante el ciclo de desarrollo de software, con el objetivo de que las vulnerabilidades que presente la aplicación web que se esté desarrollando, puedan ser prevenidas, mitigadas o minimizadas a través de métodos que se indica en la guía. Por otra parte, Fernández Sanguino (2006, p. 13) menciona que las fases de la metodología son dos: en la fase uno se da un preámbulo de la metodología para realizar pruebas de seguridad; mientras que, en la fase dos, se describe las distintas pruebas de seguridad y cómo se deben realizarlas. En la Figura 2-2 se describe el flujo de trabajo del marco de prueba de OWASP.

Figura 2-2: Flujo de trabajo del Marco de Pruebas de OWASP

Fuente: The OWASP Foundation, 2020, p. 42

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

Para el presente trabajo, las pruebas de seguridad se realizan en la fase de despliegue, utilizando las pruebas de intrusión para conocer que vulnerabilidades existen en el nuevo sistema académico, aplicando las fases de la metodología.

2.3.3. Fases para realizar el análisis de vulnerabilidades en aplicaciones web

The OWASP Foundation (2020, p. 45) recomienda un proceso de varias fases para llevar a cabo el análisis de vulnerabilidades:

- **Fase 1: Recopilación de información y diseño de escenario:** Se colecciona información sobre el sistema o aplicación que se va a evaluar, la tecnología utilizada y que recursos ocupa.
- **Fase 2: Análisis y clasificación de vulnerabilidades:** Se utilizan herramientas automatizadas y técnicas manuales para buscar vulnerabilidades en el sistema o aplicación; las cuales incluyen el escaneo de vulnerabilidades:
    - **Herramienta para detectar vulnerabilidades:** OWASP Zen Attack Proxy ZAP es una herramienta flexible y versátil diseñada para aplicaciones web, por lo que se usa para escanear vulnerabilidades en los sitios web. Entre sus principales características se destaca: es fácil de instalar, código abierto, multiplataforma, fácil de utilizar y gratuita (The OWASP Foundation, 2020, p. 76).
    - **Escaneo de vulnerabilidades:** En este proceso se busca y se detecta automáticamente las posibles debilidades o puntos débiles en el sistema mediante la herramienta (The OWASP Foundation, 2020, p. 74).
    - **Análisis de vulnerabilidades:** En este proceso se evalúa y se distinguen las debilidades detectadas durante el escaneo. Se incluye una investigación de cada una para definir conceptos, características, nivel de riesgo, entre otros aspectos de estas e identificar posibles soluciones (The OWASP Foundation, 2020, p. 75).
- **Fase 3: Reporte y resumen del análisis:** En esta fase se visualiza el reporte generado por la herramienta, para examinar los resultados del escaneo y poder elaborar un resumen detallado de las vulnerabilidades detectadas con sus respectivos parámetros (The OWASP Foundation, 2020, p. 155). Estos se describen a continuación:
    - **Identificación de alerta:** Es una etiqueta única en OWASP para identificar la alerta, la cual permite proporcionar información oportuna a los desarrolladores y administradores de seguridad, para que puedan corregir y mitigar las vulnerabilidades de una manera más efectiva (The OWASP Foundation, 2020, p. 155; the ZAP Dev Team, 2023j).
    - **Tipo de alerta:** En el contexto de OWASP, los tipos de alerta pasiva y activa se refieren a distintas formas de notificar y responder ante una amenaza de seguridad. La alerta activa puede ser un informe o una notificación enviada por correo o a través de un sistema de ticketing; mientras que, la forma pasiva es una notificación en tiempo real, como una alerta en un sistema de monitoreo de seguridad o una respuesta automatizada en un sistema de defensa.
    - **CWE ID:** Es un estándar internacional utilizado para identificar y clasificar los defectos encontrados en las aplicaciones. Cada debilidad tiene un identificador único

que se utiliza para describir y documentar las características de esta, lo que permite a los desarrolladores y administradores de seguridad comprender y mitigar estas vulnerabilidades de una manera más efectiva (Martin, 2019).

- **Nivel de Riesgo:** El nivel de riesgo se refiere a la evaluación del impacto potencial de una vulnerabilidad o amenaza en una aplicación web, se utiliza para mitigar las vulnerabilidades y permite que los desarrolladores tomen decisiones para consolidar la seguridad en las aplicaciones. OWASP utiliza ciertos factores para evaluar el nivel de riesgo: la gravedad de la vulnerabilidad, la probabilidad que ocurra un ataque y el impacto potencial que puede tener en la aplicación (The OWASP Foundation, 2020, p. 441).
- **Fase del ciclo de vida del software:** Según la metodología OWASP, se debe garantizar la seguridad del software durante el ciclo de vida, desde el diseño hasta la implementación (The OWASP Foundation, 2020, p. 12).
- **OWASP Top 10 2021:** Este documento muestra una lista de las 10 amenazas más frecuentes y nocivas encontradas en las aplicaciones web; en el resumen se relaciona la amenaza de seguridad con la vulnerabilidad detectada. (OWASP Foundation, Inc., 2021; The OWASP Foundation, 2020, p. 38)

- **Fase 4: Proponer mejores prácticas:** Se propone un plan de acción para poder mitigar las vulnerabilidades detectadas y mejorar la seguridad del sistema (The OWASP Foundation, 2020, p. 12).

2.4. Pruebas de rendimiento en aplicaciones Web

La finalidad de las pruebas de rendimiento es "determinar el rendimiento del sistema bajo una carga de trabajo" (IBM Corporation, 2021). De igual manera, Han (2021, p. 2) indica que el propósito de las pruebas de rendimiento es encontrar fallas en el rendimiento de un sistema. A su vez, Kalita y Bezboruah (2011, p. 2) mencionan que se las puede utilizar para evaluar y verificar atributos de calidad como es la escalabilidad y la fiabilidad; asimismo, se puede determinar si el software cumple ciertos criterios de rendimiento. Si se la realiza en diferentes módulos, se puede determinar que módulo es el que contribuye al bajo rendimiento del software. Entre las pruebas de rendimiento están las pruebas de carga y de estrés, las cuales se detallan a continuación:

- **Pruebas de carga:** de acuerdo con Kalita y Bezboruah (2011, p. 2), las pruebas de carga son experimentos que pretenden modelar el comportamiento de los usuarios en el mundo real con el fin de obtener una visión global del sistema. Para lograrlo, se debe tomar en cuenta cómo se comportan los usuarios; de lo contrario, la prueba no es válida. Para realizar las pruebas, el generador de carga simula a cada usuario llamado navegador; cada navegador simulado es llamado usuario virtual; la cantidad de usuarios simulados varía desde 0 hasta el valor máximo

que el sistema puede soportar y manejar aceptablemente sin colapsar. De igual manera, Fundación MTP (2022) destaca que, se ejecutan pruebas de carga para comprobar si el sistema es capaz de soportar la cantidad de solicitudes esperada, con tiempos de respuesta ideales y que el consumo de recursos no se vea comprometido; mientras que, en las pruebas de estrés la aplicación es sometida a un nivel de carga por encima del límite que soporta la aplicación, dado que, puede llegar a ocurrir bajo ciertas circunstancias.

- **Pruebas de estrés o de sobrecarga:** Kalita y Bezboruah (2011, p. 2) se ejecutan para determinar la estabilidad de un sistema; la finalidad de estas pruebas es descubrir fugas de memoria, límites de ancho de banda, limitaciones de los servidores o del hardware en donde se almacene el software, entre otros problemas ocasionados por la sobrecarga de peticiones. A diferencia de las pruebas de carga, las pruebas de estrés buscar probar el sistema con una cantidad que sobrepasa la capacidad operativa normal. En síntesis, el enfoque de las pruebas de rendimiento buscar si existen fallas de rendimiento en un sistema, mediante un análisis previo para conocer cuantas solicitudes puede soportar el servidor sin colapsar; dependiendo de la finalidad, se pueden aplicar distintas pruebas como son de carga o de estrés. En el presente trabajo se continuará con el estudio de las pruebas de estrés, dado que la finalidad es conocer qué cantidad de peticiones soporta el sistema académico sin que se haya fallos en el sistema.

2.4.1. Pruebas de estrés en Aplicaciones web

Según Maâlej et al. (2015, p. 285) las pruebas de estrés están diseñadas para someter a un sistema a una carga irrazonable de trabajo con la finalidad de colapsarlo, provocando un fallo y con ello, se evalúa la capacidad de recuperación de fallos en el sistema; de igual manera, se espera que la falla no corrompa o pierda los datos. A su vez, Pradeep y Kumar Sharma (2019, p. 1) destacan que las pruebas de estrés analizan la robustez del software; para ejecutarlas se estudia cada módulo para conocer si hay algún problema; por consiguiente, se debe evaluar cada uno en condiciones extremas; es decir, se le manda una gran carga de trabajo que excede los límites que soporta el sistema sin colapsar. En cambio, Hegde (2014, p. 3) menciona que las pruebas de estrés se enfocan en validar las características de rendimiento del sistema cuando es sometido a condiciones críticas, como son la memoria limitada, espacio en disco reducido, fallo en el servidor o, que la cantidad de peticiones sea mayor a la esperada en el sistema que está en producción. Además, con las pruebas se puede conocer qué indicadores pueden ser monitoreados para percatarse de un fallo cercano. En resumen, el objetivo de las pruebas de estrés es someter al sistema a una carga de trabajo mayor de la que puede soportar, para saber si es capaz de funcionar correctamente o si colapsa. Además, es importante seguir una metodología, la cual indica qué medidas o fases se deben seguir para realizar este tipo de pruebas.

2.4.2. Fases para realizar pruebas de estrés en una aplicación web

Las pruebas de estrés permiten conocer el comportamiento de una aplicación web y medir la solidez cuando es sometida a condiciones de carga pesada (Hamilton, 2020). Para ejecutarla se debe considerar algunos puntos, los cuales se describen a continuación:

- **Fase 1: Planificación:** Se define el alcance de la prueba y el propósito o los objetivos de la prueba, estos tienen que ser medibles para que puedan ser evaluados; además, se identifica y describe la arquitectura del software a ser probado; de igual manera, se precisa la herramienta a utilizar (Congote y Hincapie, 2020). En el presente trabajo se utilizará la herramienta Apache JMeter para ejecutar las pruebas de estrés, dado que permite simular una carga pesada en el servidor, para probar el rendimiento del sistema académico (Husufa y Prihandi, 2022).
- **Fase 2: Identificación de los recursos:** Se definen los recursos necesarios para realizar la prueba de estrés; se incluyen los recursos de hardware, software y humanos (Congote y Hincapie, 2020).
- **Fase 3: Diseño del escenario:** Se identifica y se define el escenario en donde se ejecutará la prueba. Asimismo, se determinan las condiciones para ejecutar la prueba, se deben tomar en cuenta distintos aspectos como es la carga de trabajo, el tiempo y la evaluación de la prueba.
- **Fase 4: Preparación del entorno:** Se configura y se revisa el escenario antes de ser ejecutado, para comprobar que esté bien hecho y con los parámetros definidos anteriormente.
- **Fase 5: Ejecución:** Se realizan las pruebas de estrés previamente configuradas. De igual manera, se debe tomar notas de cómo responde el sistema bajo diferentes niveles de estrés.
- **Fase 6: Análisis de resultados:** Se examinan y evalúan los resultados de las pruebas realizadas para conocer cuáles son las fallas en caso de haberlas o, si el sistema sigue funcionando correctamente a pesar de la sobrecarga de trabajo.
- **Fase 7: Estrategias para mejorar el rendimiento:** Después del análisis de resultados, se debe hacer un informe en el cual, en caso de haber fallas se debe brindar buenas prácticas para mitigar los errores encontrados.

Por consiguiente, el proceso para evaluar el comportamiento de una aplicación web bajo cargas extremas debe cumplir con las fases descritas por la metodología para garantizar un óptimo rendimiento en el entorno de producción de la aplicación. A continuación, se describen trabajos relacionados a las pruebas de seguridad descritas anteriormente.

2.5. Trabajos relacionados

Harrell et al. (2018, p. 1) manifiesta que se efectuó un escaneo a gran escala de 272 aplicaciones web pertenecientes a varias instituciones de educación superior con la finalidad de evaluar las vulnerabilidades. De igual manera, se identificó vulnerabilidades que no fueron posibles corregirlas, las cuales fueron recreadas y remediadas en un entorno virtual, desarrollando mecanismos mejorados y automatizados, los que brindaron información concreta y útil para las instituciones. Otro estudio indica que se realizó un análisis de vulnerabilidades con dos herramientas: OWASP WAP y RIPS, en páginas web que son muy vulnerables para poder comparar la fiabilidad de cada escaneo. Según los resultados, se comprobó que OWASP WAP arroja resultados más confiables a diferencia de RIPS (Tyagi y Kumar, 2018, p. 1).

En lo referente a las pruebas de estrés, se han realizado estudios que contemplan JMeter como una herramienta para poder realizarlas. Hwang et al. (2014, p. 1) menciona que empleó el modelo de computación MapReduce para implementar un Servicio de Pruebas (TaaS) para ejecutar pruebas de estrés. La plataforma TaaS fue comparada con JMeter, lo que comprobó que JMeter presenta una menor tasa de error en comparación con TaaS. Otro estudio realizó pruebas de estrés utilizando JMeter, aplicado a dos módulos de ataque DoS y generadores de tráfico de red, con la finalidad de probar el comportamiento de la aplicación web y medir el rendimiento del servidor (Grabovsky et al., 2018, p. 1).

CAPÍTULO III

3. MARCO METODOLÓGICO

En este capítulo se describe el diseño de estudio y los mecanismos a utilizar para el abordaje del análisis de las vulnerabilidades y pruebas de estrés al sistema académico de la ESPOCH.

3.1. Diseño de estudio

3.1.1. Tipo de estudio

En el presente trabajo es de tipo descriptivo, ya que detalla las fases necesarias para realizar un análisis de vulnerabilidades y pruebas de estrés en el sistema académico de la ESPOCH, mediante la metodología OWASP.

3.1.2. Métodos y técnicas

En la **Tabla 3-3,** se observa los métodos y técnicas que están relacionados con fases de las pruebas de estrés software cada uno de los objetivos específicos planteados en el presente trabajo.

Tabla 3-3: Objetivos, métodos, descripción, técnicas y fuentes para cumplir cada objetivo.

OBJETIVOS	MÉTODOS	DESCRIPCIÓN	TÉCNICAS	FUENTES
Describir las características de la metodología OWASP para determinar las vulnerabilidades que tiene el sistema; el procedimiento y las herramientas utilizadas para realizar las pruebas de estrés.	Descriptivo	Describir y explicar las características de la metodología OWASP para determinar las vulnerabilidades que tiene el sistema; el procedimiento y las herramientas utilizadas para realizar las pruebas de estrés.	Revisión de documentación	Libros, artículos científicos
Analizar las vulnerabilidades del sistema académico aplicando la metodología OWASP para conocer las más críticas.	OWASP	Es una metodología de seguridad de auditoría web, abierta y colaborativa, orientada al análisis de seguridad de aplicaciones Web, y usada como referente en auditorías de seguridad, con ello se puede analizar las vulnerabilidades del sistema académico de la ESPOCH.	• Observación • Experimentación	OWASP ZAP

Realizar las pruebas de estrés al sistema académico para conocer el rendimiento y proponer estrategias para mitigar en caso de fallo.	Analítico	Analizar los resultados de las pruebas de estrés y proponer estrategias para mejorar el rendimiento.	• Pruebas de estrés • Análisis de escenarios • Experimentación	Apache JMeter
Proponer las mejores prácticas que permita mejorar la seguridad del sistema académico.	Descriptivo	Se presenta una descripción precisa y objetiva de los resultados para proponer mejores prácticas la cual permitirá mejorar la calidad de seguridad del sistema académico.	Revisión de documentación	Libros, artículos científicos

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.1.3. Población y Muestra

Las pruebas de vulnerabilidades y pruebas de estrés se realizaron en el nuevo sistema académico de la ESPOCH, tomando como muestra las 9 carreras: Software, Diseño gráfico, Electrónica en control y redes industriales, Telecomunicaciones, Tecnologías de la Información, Telemática, Electricidad y Agroindustria.

3.2. Estudio de factibilidad

Se plantea que en el estudio de factibilidad se entrega al cliente una estimación del costo aproximado del análisis de vulnerabilidades y de las pruebas de estrés del sistema académico de la ESPOCH, determinando los recursos económicos, técnicos, operativos para decidir si el proyecto de integración curricular es viable o no. Los diversos tipos de factibilidad se describen en el Anexo D. En el presente trabajo se describe como se realizó el análisis de vulnerabilidades en el sistema académico. Esta se detalla a continuación.

3.3. Análisis de vulnerabilidades del sistema académico aplicando la metodología OWASP

Para el desarrollo del análisis de vulnerabilidades en el sistema académico se aplicó las fases establecidas en la metodología OWASP, que se detallan a continuación.

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

Se analizó como está desarrollada la aplicación, la infraestructura y su arquitectura; que se muestra en la Figura 3-3. El sistema académico de la ESPOCH cuenta con la siguiente arquitectura:

- **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 (Anibal Herrera, comunicación personal, 13 de octubre de 2022; Kornienko et al., 2021, p. 3).
- **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 (Anibal Herrera, comunicación personal, 13 de octubre de 2022).
- **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.

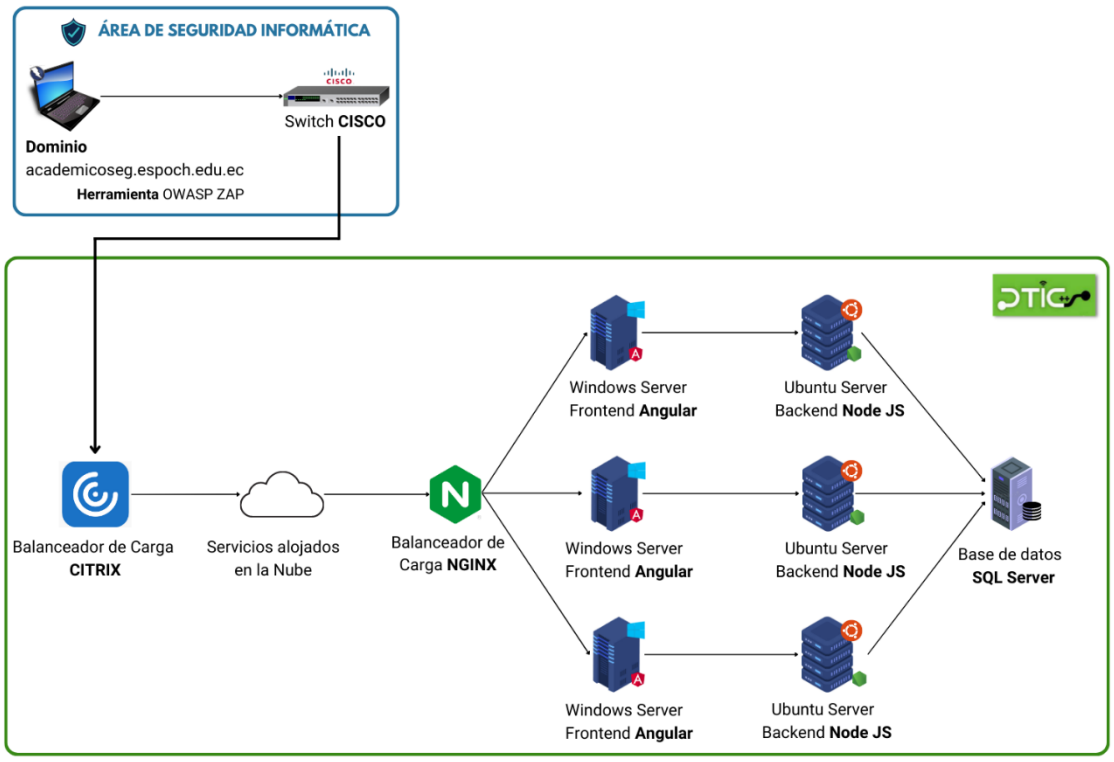


Figura 3-3: Escenario de ejecución de análisis de vulnerabilidades

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

En la Tabla 4-3, se describen los recursos utilizados en el escenario de prueba para la detección de vulnerabilidades.

Tabla 4-3: Elementos del escenario para realizar el análisis de vulnerabilidades en el sistema académico de la ESPOCH

No	HERRAMIENTA/ RECURSOS		SOFTWARE/ FRAMEWORK		FUNCIÓN
1	Laptop ASUS ROG Strix SCAR 15 G532	**Procesador:** AMD Ryzen 9 **Tarjeta Gráfica:** RTX 3070 **RAM:** 32 GB **Disco:** SSD 1 TB	Windows 10 Pro OWASP ZAP		Realizar el análisis de vulnerabilidades en el sistema académico con la herramienta OWASP ZAP
3	Servidor Frontend	**# Procesadores:** 8 **RAM:** 12 GB	Servidor CITRIX		Balancear la carga de las peticiones HTTP hechas a través de la página principal del sistema académico
			Windows Server	Angular	Mostrar los elementos de la página
3	Servidor Backend	**# Procesadores:** 8 **RAM:** 14 GB	Servidor NGINX		Equilibrar la carga de las peticiones HTTP hechas a través de la página principal del sistema académico
			Ubuntu Server	Node JS	Envía los datos y los elementos para que se muestren en la vista
1	Servidor de base de datos	**# Procesadores:** 8 **RAM:** 12 GB	SQL Server		Gestionar los datos e información de los estudiantes

Fuente: Herrera, 2022

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.2. Fase 2: Análisis y clasificación de vulnerabilidades

Como se visualiza en las Figuras 4-3, 5-3 y 6-3, se realizó el escaneo de vulnerabilidades en el sistema académico de la ESPOCH, a través de la herramienta con la herramienta OWASP Zen Attack Proxy ZAP. En el análisis se encontró 15 vulnerabilidades con riesgos medios y bajos. A continuación, se describe cada una.

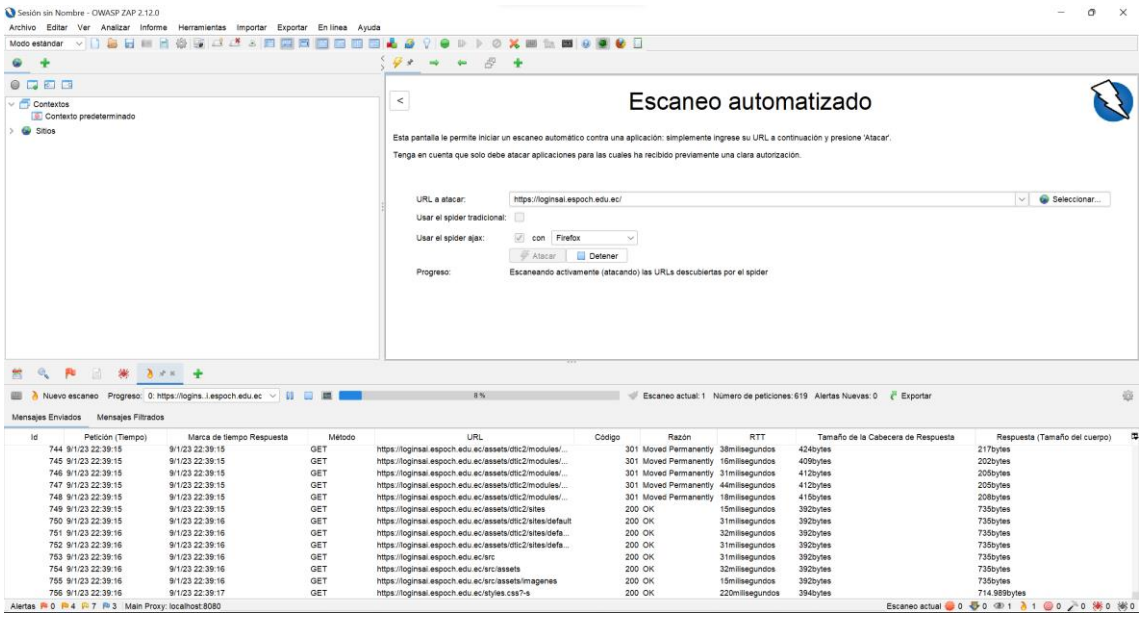


Figura 4-3: Herramienta OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

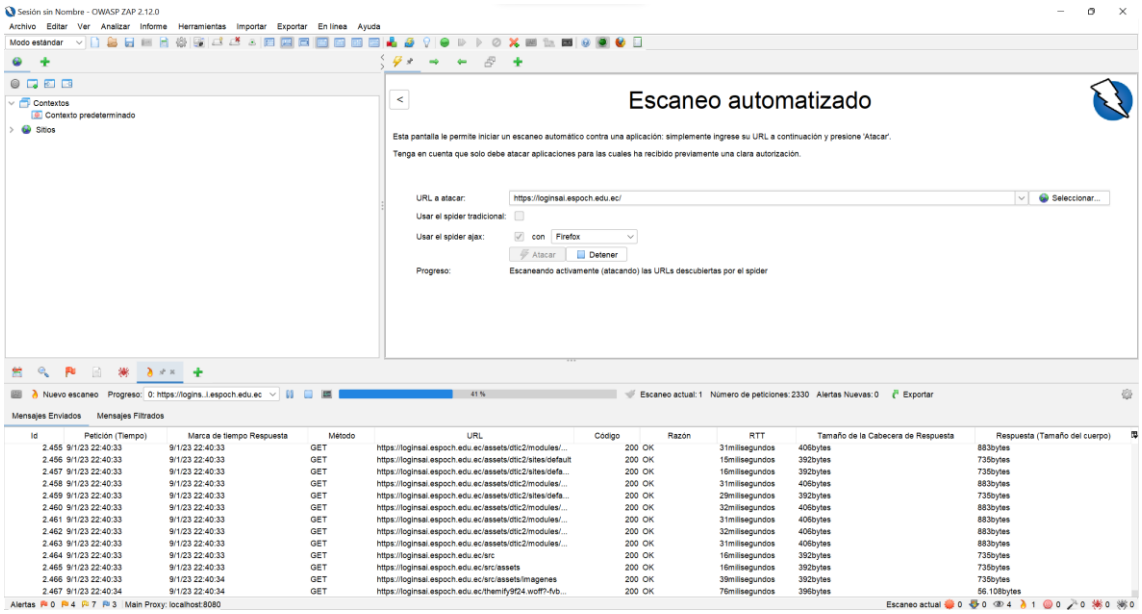


Figura 5-3: Escaneo de Vulnerabilidades - OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

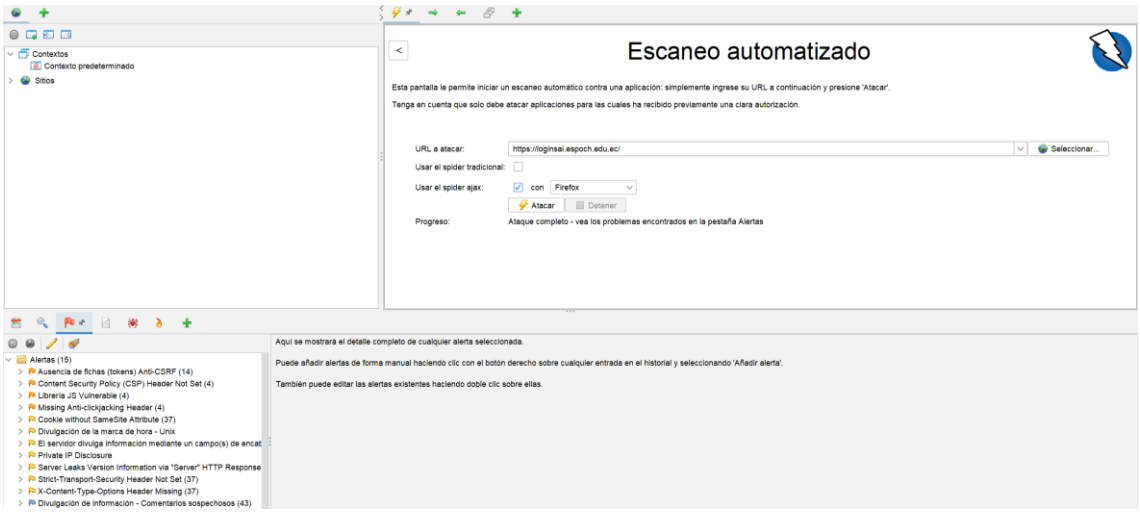


Figura 6-3: Análisis de Vulnerabilidades – Herramienta OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

3.3.2.1. Ausencia de fichas (tokens) Anti-CSRF

Se refiere a una brecha en la seguridad de la aplicación web, en la cual le permite a un atacante realizar acciones en nombre del usuario autenticado sin su conocimiento o consentimiento. Esto puede suceder cuando una aplicación no utiliza tokens anti-CSRF para validar las solicitudes entrantes, permitiendo que un atacante inyecte solicitudes maliciosas desde otro sitio web y realice acciones en el contexto de la sesión activa del usuario (The MITRE Corporation, 2023c; the ZAP Dev Team, 2023a).

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

Este defecto se refiere a la ausencia de una política de seguridad de contenido en una aplicación web. La política de seguridad de contenido es un encabezado HTTP que indica al navegador web que tipo de contino es permitido y de que origen proviene (The MITRE Corporation, 2023e; the ZAP Dev Team, 2023b).

3.3.2.3. Falta el encabezado antisecuestro de clics

Esta debilidad describe la falta de una cabecera Anti-clickjacking, la cual previene un ataque de secuestro de clics, el que se produce cuando un atacante duplique el contenido de una página web en otro sitio malicioso con el objetivo de engañar a los usuarios, pidiéndoles que ingresen información confidencial o realizando acciones no autorizadas con los datos del usuario (OWASP Foundation, Inc., 2023; The MITRE Corporation, 2023a).

3.3.2.4. Biblioteca JS vulnerable

The MITRE Corporation (2023f) afirma que esta debilidad es generada cuando se utiliza un marco de JavaSacript desactualizado, obsoleto o que contenga código malicioso, permitiendo que se comparta información confidencial albergada en la aplicación, que se otorgue permisos no autorizados a los atacantes.

3.3.2.5. Cookie sin atributo SameSite

Este defecto origina un problema de seguridad relacionado con la manera en que una página web maneja las cookies; si estas no están configuradas con el atributo SameSite, un sitio malicioso puede enviar una solicitud POST al dominio del sitio vulnerable para obtener las cookies de manera automática, lo que da lugar a un ataque CSRF (The MITRE Corporation, 2023g; the ZAP Dev Team, 2023c).

3.3.2.6. Divulgación de la marca de hora - Unix

The MITRE Corporation (2023b) indica a esta debilidad como un problema de seguridad que posibilita a los atacantes descubrir la hora exacta en que un sistema operativo Unix ha sido reiniciado, permitiendo que el atacante realice ataques más efectivos, porque conoce la fecha y hora del último reinicio del sistema.

3.3.2.7. El servidor divulga información mediante un campo de encabezado de respuesta HTTP ""X-Powered-By""

Esta vulnerabilidad genera un problema de seguridad porque se filtra información sensible a través de uno o más encabezados de respuesta X-Powered-By, que puede ser ocupada por los atacantes, la cual contiene datos detallados sobre la tecnología utilizada y el servidor web (The MITRE Corporation, 2023b; the ZAP Dev Team, 2023g).

3.3.2.8. Divulgación de IP privada

De acuerdo con the ZAP Dev Team (2023e), este fallo de seguridad en la protección de las direcciones IP privadas posibilita que los atacantes obtengan la dirección IP de un sistema o un dispositivo, a través de una solicitud HTTP o una conexión de red.

3.3.2.9. Encabezado de respuesta del servidor HTTP

Esta alerta describe que la aplicación está siendo filtrada en la respuesta de HTTP 'Servidor'; a pesar de dicho inconveniente, los datos filtrados no son útiles por sí solos; no obstante, se debe mitigar este problema por seguridad (the ZAP Dev Team, 2023h).

3.3.2.10. Encabezado de seguridad de transporte estricto

Es una cabecera HTTP que permite a un servidor web indicar a un navegador web que solo se deben realizar solicitudes HTTPS (HTTP seguro) a ese servidor y que no se deben realizar solicitudes HTTP no seguras. El uso de HSTS previene ataques como la interceptación de la red y la falsificación de certificados, ya que garantiza que todas las comunicaciones con el servidor se realicen de forma segura a través de HTTP (ScanRepeat, 2020b; the ZAP Dev Team, 2023i).

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

Esta alerta describe que falta el encabezado Content-Type; como consecuencia, las versiones desactualizadas de los navegadores Internet Explorer y Chrome pueden hacer un rastreo MIME; es decir, pueden explorar el contenido que brinda el servidor web y determinar cómo usarlo. Esto puede generar que un atacante utilice esta debilidad y realice un ataque de secuencia de comandos (ScanRepeat, 2020c; the ZAP Dev Team, 2023l).

3.3.2.12. Divulgación de información - Comentarios sospechosos

Esta menciona los comentarios en el código fuente. Se conoce que comentar el código permite a los desarrolladores trabajar de manera armoniosa; sin embargo, puede ser peligroso, porque el atacante puede entender la lógica de programación a través de estos y realizar acciones maliciosas (ScanRepeat, 2020a).

3.3.2.13. Aplicación web moderna

Esta alerta menciona que la aplicación del sistema académico es moderna; además, podría ser más efectivo explorar la aplicación con la herramienta Ajax Spider, es una forma avanzada de analizar la aplicación y detectar posibles problemas, para que esta funciona de una manera óptima (the ZAP Dev Team, 2023d).

3.3.2.14. Reexaminar las directivas de control de caché

Esta alerta se refiere a un problema de seguridad en que las directivas de control de caché en una aplicación web no están configuradas adecuadamente, es posible que la información confidencial, como tokens de autenticación o datos sensibles, se almacena en caché y sea accedida por personas no autorizada. El control de caché es un mecanismo utilizado por los navegadores, los servidores web para almacenar en caché las páginas y los recursos solicitados con frecuencia para acelerar la navegación (The MITRE Corporation, 2023d; the ZAP Dev Team, 2023f).

3.3.2.15. Fuzzer de agente de usuario

Esta alerta menciona que el atacante puede utilizar un fuzzer para enviar información inválida o mal formada a una aplicación en el campo "Agente de usuario" de la cabecera HTTP, la cual identifica el tipo de navegador o aplicación que está solicitando la página. Un fuzzer de agente de usuario puede usarse para enviar información inválida o mal intencionada, lo que puede provocar errores y puede llevar a la explotación de la vulnerabilidad en la aplicación (the ZAP Dev Team, 2023k).

3.3.3. Fase 3: Reporte y resumen de análisis

A continuación, se detalla el informe y el resumen del análisis de vulnerabilidades el cual proporcionó el documento la herramienta OWASP ZAP con una descripción detallada de las vulnerabilidades identificadas en la aplicación web durante el análisis de seguridad. Esta incluye un resumen general de cada uno de los parámetros emitidos, proporcionando una comprensión clara concisa de los riesgos y debilidades presentes en la aplicación. En la Figura 7-3 y Figura 8-3 se observa dicho reporte.

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

Figura 7-3: Reporte de escaneo - OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

Tipo de alerta	Riesgo	Contar
HTTP ""X-Powered-By""		
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

Figura 8-3: Reporte de escaneo - OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

3.3.3.1. Ausencia de fichas (tokens) Anti-CSRF

En la Tabla 5-3, se detallan los parámetros más relevantes de esta alerta, tal como, esta ha sido detectada en tiempo real, se relaciona la vulnerabilidad de pérdida de control de acceso, entre otras características de suma importancia.

Tabla 5-3: Descripción de los parámetros más relevantes de la vulnerabilidad 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

Fuente: The MITRE Corporation, 2023c; the ZAP Dev Team, 2023a

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

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

En la fase anterior se describió en qué consiste esta falla, por lo que, en la Tabla 6-3 se mencionan los detalles más importantes de la vulnerabilidad, como es el nivel de riesgo, a qué clasificación pertenece, entre otros parámetros descritos.

Tabla 6-3: Detalles de la vulnerabilidad Encabezado de política de seguridad de contenido (CSP) no establecido

Detalles	Descripción
Identificación de alerta	10038
Tipo de alerta	Pasiva
CWE ID	693 – Fallo del mecanismo de protección
Nivel de Riesgo	Medio
Fase del ciclo de vida del Software	Arquitectura y diseño, implementación y operación.
OWASP Top 10 2021	A05:2021- Configuración de Seguridad Incorrecta

Fuente: The MITRE Corporation, 2023e; the ZAP Dev Team, 2023b

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.3.3. Falta el encabezado antisecuestro de clics

En la Tabla 7-3, se pormenoriza los parámetros de la alerta descubierta, como es: en qué fase se puede detectar y a cuál vulnerabilidad del Top 10 de OWASP pertenece, entre otros relevantes.

Tabla 7-3: Pormenorización de los parámetros relevantes de la vulnerabilidad Falta el encabezado antisecuestro de clics

Detalles	Descripción
Identificación de alerta	10020-1
Tipo de alerta	Pasiva
CWE ID	1021 – Restricción incorrecta de marcos o capas de interfaz de usuario renderizados
Nivel de Riesgo	Medio
Fase del ciclo de vida del Software	Implementación
OWASP Top 10 2021	A05:2021 – Perdida de Control de Acceso

Fuente: The MITRE Corporation, 2023a

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.3.4. Biblioteca JS vulnerable

En la siguiente Tabla 8-3, se explica los datos más relevantes de la vulnerabilidad encontrada en el sistema académico, como la identificación de la alerta, cual es el tipo, a qué vulnerabilidad pertenece según el Top 10 del 2021 de OWASP, entre otros detalles.

Tabla 8-3: Descripción de los parámetros más relevantes de la alerta Biblioteca JS vulnerable

Detalles	Descripción
Identificación de alerta	10003
Tipo de alerta	Pasiva
CWE ID	829 – Inclusión de funcionalidad de esfera de control no confiable
Nivel de Riesgo	Medio
Fase del ciclo de vida del Software	Implementación
OWASP Top 10 2021	A08:2021 – Fallas de software e integridad de datos

Fuente: The MITRE Corporation, 2023f

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.3.5. Cookie sin atributo SameSite

En la Tabla 9-3, se indica la información más destacada de esta vulnerabilidad, como lo referente a la fase del ciclo de vida del software en la que se produce este error, el nivel de riesgo, entre otros parámetros descritos a continuación.

Tabla 9-3: Explicación detallada sobre la vulnerabilidad de la Cookie sin atributo SameSite

Detalles	Descripción
Identificación de alerta	10054
Tipo de alerta	Pasiva
CWE ID	1275 – Cookie confidencial con atributo de SameSite incorrecto
Nivel de Riesgo	Bajo
Fase del ciclo de vida del Software	Implementación
OWASP Top 10 2021	A01:2021 – Control de acceso roto

Fuente: The MITRE Corporation, 2023g; the ZAP Dev Team, 2023c

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.3.6. Divulgación de la marca de hora - Unix

Se describen los datos más destacables de esta vulnerabilidad encontrados en el reporte generado, como se muestra en la Tabla 10-3.

Tabla 10-3: Detalles de la vulnerabilidad Divulgación de la marca de hora - Unix

Detalles	Descripción
Identificación de alerta	10096
Tipo de alerta	Pasiva
CWE ID	200 – Exposición de información sensible a un agente no autorizado
Nivel de Riesgo	Bajo
Fase del ciclo de vida del Software	Arquitectura y Diseño – Implementación
OWASP Top 10 2021	A01:2021 – Control de acceso roto

Fuente: The MITRE Corporation, 2023b

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.3.7. El servidor divulga información mediante un campo de encabezado de respuesta HTTP ""X-Powered-By""

En la Tabla 11-3, se detallan los factores esenciales de la alerta encontrada, como la identificación de la alerta, el tipo de alerta, nivel de riesgo, entre otros.

Tabla 11-3: Detalles de la vulnerabilidad El servidor divulga información mediante un campo de encabezado de respuesta HTTP ""X-Powered-By""

Detalles	Descripción
Identificación de alerta	10037
Tipo de alerta	Pasiva
CWE ID	200 – Exposición de información sensible a un agente no autorizado
Nivel de Riesgo	Medio
Fase del ciclo de vida del Software	Arquitectura y Diseño – Implementación
OWASP Top 10 2021	A01:2021 – Control de acceso roto

Fuente: The MITRE Corporation, 2023b; the ZAP Dev Team, 2023g

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.3.8. Divulgación de IP privada

En la Tabla 12-3, se detallan los aspectos clave de esta falla, incluido el tipo de alerta utilizado para determinar si esta necesita una acción automática e inmediata.

Tabla 12-3: Especificación de los parámetros más importantes de la vulnerabilidad Divulgación de IP privada

Detalles	Descripción
Identificación de alerta	2
Tipo de alerta	Pasiva
CWE ID	200 – Exposición de información sensible a un agente no autorizado
Nivel de Riesgo	Bajo
Fase del ciclo de vida del Software	Arquitectura y Diseño – Implementación
OWASP Top 10 2021	A04:2021 – Diseño inseguro

Fuente: The MITRE Corporation, 2023b; the ZAP Dev Team, 2023e

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.3.9. Encabezado de respuesta del servidor HTTP

En la Tabla 13-3, se describen los elementos más importantes de esta alerta, que se vinculan a la vulnerabilidad de configuración de seguridad incorrecta. Se detallan aspectos como el tipo de alerta, la identificación, nivel de riesgo entre otros.

Tabla 13-3: Elementos de la vulnerabilidad Encabezado de respuesta del servidor HTTP

Detalles	Descripción
Identificación de alerta	10036-1
Tipo de alerta	Pasiva
CWE ID	200 – Exposición de información sensible a un agente no autorizado
Nivel de Riesgo	Bajo
Fase del ciclo de vida del Software	Arquitectura, diseño e implementación
OWASP Top 10 2021	A05:2021 – Configuración de seguridad incorrecta

Fuente: The MITRE Corporation, 2023b; the ZAP Dev Team., 2023h

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.3.10. Encabezado de seguridad de transporte estricto

En la Tabla 14-3, se proporciona una descripción detallada la información relacionada con esta alerta, el nivel de riesgo informativo y otros parámetros de suma importancia.

Tabla 14-3: Especificación de la vulnerabilidad Encabezado de seguridad de transporte estricto

Detalles	Descripción
Identificación de alerta	10035
Tipo de alerta	Pasiva
CWE ID	319 – Transmisión de información sensible en texto claro
Nivel de Riesgo	Informativo
Fase del ciclo de vida del Software	Arquitectura, diseño, operación y configuración del sistema
OWASP Top 10 2021	A05:2021 – Configuración de seguridad incorrecta

Fuente: the ZAP Dev Team., 2023i

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

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

En la Tabla 15-3, se explican los detalles descubiertos de esta alerta, como es el nivel de riesgo, a que vulnerabilidad OWASP TOP 10 pertenece, entre otros.

Tabla 15-3: Detalles de la vulnerabilidad falta el encabezado X-Content-Type-Options

Detalles	Descripción
Identificación de alerta	10021
Tipo de alerta	Pasiva
CWE ID	693 – Fallo del mecanismo de protección
Nivel de Riesgo	Bajo
Fase del ciclo de vida del Software	Arquitectura, diseño, implementación y operación
OWASP Top 10 2021	A05:2021 – Configuración de seguridad incorrecta

Fuente: The MITRE Corporation, 2023d; the ZAP Dev Team., 2023l

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.3.12. Divulgación de información - Comentarios sospechosos

En la Tabla 16-3, se muestra la información más relevante sobre esta vulnerabilidad, incluido el tipo de alerta, la enumeración de las debilidades más frecuentes, entre otros datos.

Tabla 16-3: Parámetros importantes de la alerta de Divulgación de información - Comentarios sospechosos

Detalles	Descripción
Identificación de alerta	10027
Tipo de alerta	Pasiva
CWE ID	200 – Exposición de información sensible a un agente no autorizado
Nivel de Riesgo	Informativo
Fase del ciclo de vida del Software	Arquitectura, diseño e implementación
OWASP Top 10 2021	A01:2021 – Pérdida de control de acceso

Fuente: The MITRE Corporation, 2023b

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.3.13. Alerta de Aplicación web moderna

En la Tabla 17-3, se destaca el tipo de alerta y la identificación, junto con otros parámetros que son de tipo informativo; cabe recalcar que es una alerta poco común de detectar.

Tabla 17-3: Elementos significativos de tipo informativo de la alerta Aplicación web moderna

Detalles	Descripción
Identificación de alerta	10109
Tipo de alerta	Pasiva
Nivel de Riesgo	Informativo

Fuente: the ZAP Dev Team., 2023d

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.3.14. Reexaminar las directivas de control de caché

En la Tabla 18-3, se describen los indicadores principales detalles de esta falla, junto con el tipo de alerta que utiliza para decidir si es necesaria una acción automática e inmediata, entre otros parámetros.

Tabla 18-3: Indicadores de la vulnerabilidad Reexaminar las directivas de control de caché

Detalles	Descripción
Identificación de alerta	10015
Tipo de alerta	Pasiva
CWE ID	525 – Uso del caché del navegador web que contiene información sensible
Nivel de Riesgo	Informativo
Fase del ciclo de vida del Software	Implementación
OWASP Top 10 2021	A04:2021 – Diseño inseguro

Fuente: The MITRE Corporation, 2023d; the ZAP Dev Team, 2023f

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

3.3.3.15. Fuzzer de agente de usuario

Se explica en la Tabla 19-3, los parámetros de forma detallada de esta vulnerabilidad, los cuales incluyen el tipo de alerta y otra información pertinente.

Tabla 19-3: Elementos informativos de la vulnerabilidad Fuzzer de agente de usuario

Detalles	Descripción
Identificación de alerta	10104
Tipo de alerta	Activa
CWE ID	Sin información
Nivel de Riesgo	Informativo
Fase del ciclo de vida del Software	Implementación
OWASP Top 10 2021	A08:2021 – Fallas en el Software y en la integridad de los datos

Fuente: the ZAP Dev Team., 2023k

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

En el Anexo E se detalla el informe completo del escaneo de vulnerabilidades, que muestran las 15 alertas, con diferentes niveles de riesgo y de confianza.

3.3.4. Fase 4: Propuesta de mejores prácticas

En el capítulo de resultados se detallan la propuesta de mejores prácticas en función de las vulnerabilidades encontradas en el sistema académico.

3.4. Pruebas de estrés del sistema académico

Es importante realizar las pruebas de estrés para conocer el rendimiento del sistema académico, este procedimiento se desarrolla siguiendo un proceso. A continuación, se describen las fases:

3.4.1. 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 (D. Palacios, comunicación personal, 13 de octubre de 2022).

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 (Anibal Herrera, comunicación personal, 13 de octubre de 2022; Kornienko et al., 2021, p. 3).

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 (Anibal Herrera, comunicación personal, 13 de octubre de 2022).

3.4.2. Fase 2: Identificar de los recursos

En la Tabla 20-3 se describe los recursos de hardware requeridos.

Tabla 20-3: Recursos de Hardware necesarios para ejecutar la prueba de estrés

CANTIDAD	DESCRIPCIÓN	RECURSOS
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

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

En la Tabla 21-3 se visualiza los recursos humanos involucrados en la realización y ejecución de la prueba.

Tabla 21-3: Recursos Humanos necesarios para realizar las pruebas de estrés en el sistema.

CANTIDAD	FUNCIÓN	FORMACIÓN	OBSERVACIÓN
2	Analista de Seguridad – Tester	Estudios de tercer nivel	Analista de seguridad y tester de aplicaciones

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

En la Tabla 22-3 se observa los recursos de software requeridos para diseñar y ejecutar la prueba.

Tabla 22-3: Recursos de Software necesarios para ejecutar las pruebas de estrés.

CANTIDAD	DESCRIPCIÓN	OBSERVACIÓN
1	Apache JMeter, herramienta para realizar las pruebas de estrés	Software libre

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

Una vez identificados los recursos para realizar la prueba de estrés, se diseña el escenario sobre el cual se ejecutarán las pruebas de estrés; de igual manera, se definen las condiciones para ejecutar la prueba considerando la cantidad de usuarios que se conectarán al sistema académico durante el período de matrículas.

3.4.3. Fase 3: Diseñar del escenario

En la Figura 9-3, se muestra el escenario en donde se ejecutarán las pruebas de estrés, considerando los recursos que ocupa el sistema académico durante el período de matrículas.

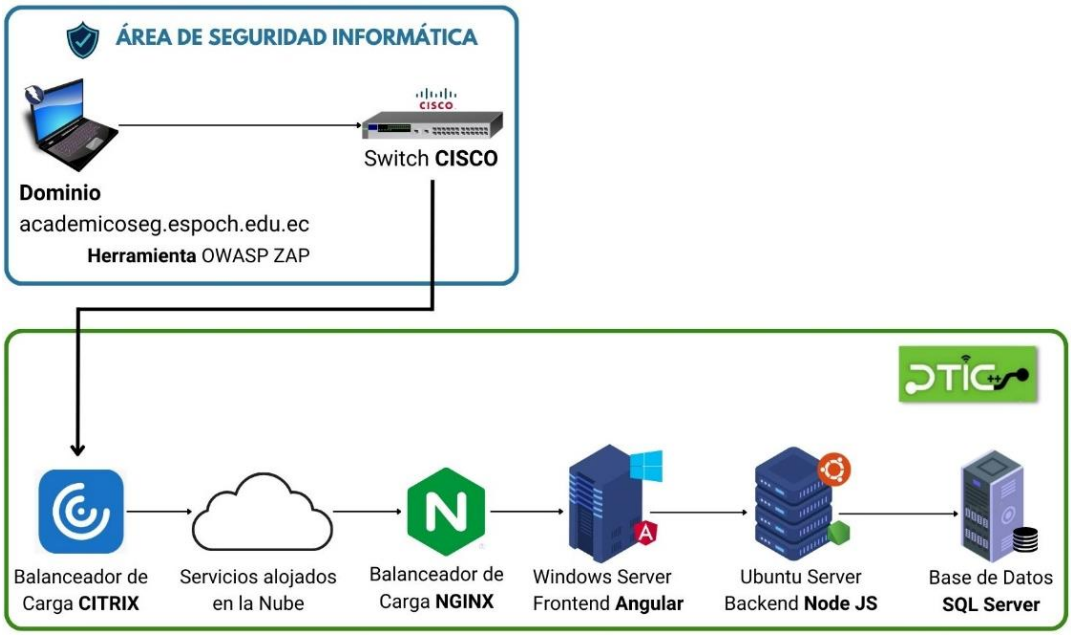


Figura 9-3: Escenario de prueba

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

En la Tabla 23-3 se describen los elementos involucrados para ejecutar la prueba de estrés en el sistema académico, el computador que se está utilizando, la herramienta para hacer la prueba y los servidores que utiliza el sistema académico para procesar las peticiones que se envían del lado del cliente.

Tabla 23-3: Elementos del escenario para ejecutar las pruebas de estrés en el sistema académico de la ESPOCH

No	HERRAMIENTA/ RECURSOS		SOFTWARE/ FRAMEWORK		FUNCIÓN
1	Laptop Lenovo Legion Y520	**Procesador:** Intel Core i7 **Tarjeta Gráfica:** NVIDIA GeForce GTX 1060 **RAM:** 16 GB **Disco:** SSD 128 GB, HD 1 TB	Windows 10 Pro JMeter		Diseñar y ejecutar las pruebas de estrés al sistema académico con la herramienta JMeter
1	Servidor Frontend	**# Procesadores:** 8 **RAM:** 12 GB	Servidor CITRIX		Balancear la carga de las peticiones HTTP hechas a través de la página principal del sistema académico
			Windows Server	Angular	Mostrar los elementos de la página
1	Servidor Backend	**# Procesadores:** 8 **RAM:** 14 GB	Servidor NGINX		Equilibrar la carga de las peticiones HTTP hechas a través de la página principal del sistema académico
			Ubuntu Server	Node JS	Envía los datos y los elementos para que se muestren en la vista
1	Servidor de base de datos	**# Procesadores:** 8 **RAM:** 12 GB	SQL Server		Gestionar los datos e información de los estudiantes

Fuente: Herrera, 2022

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

Para diseñar el escenario de la prueba de estrés, se considera que, en 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 (N. Salazar, comunicación personal, 9 de enero de 2023).

Se considera 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 (Noboa Cevallos y Cuenca Obregon, 2021, p. 65). 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 Ecuación 1.

Ecuación 1: Fórmula para calcular la cantidad de estudiantes por semestre.

$$\text{Estudiantes}/_{\text{Semestre}} = \text{Total} - (\text{Deserción} + \text{Titulados} + \text{Egresados})$$

$$\text{Estudiantes}/_{\text{Semestre}} = 448 \text{ estudiantes}$$

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; es decir, habrá un crecimiento exponencial, como se muestra en la Figura 10-3.

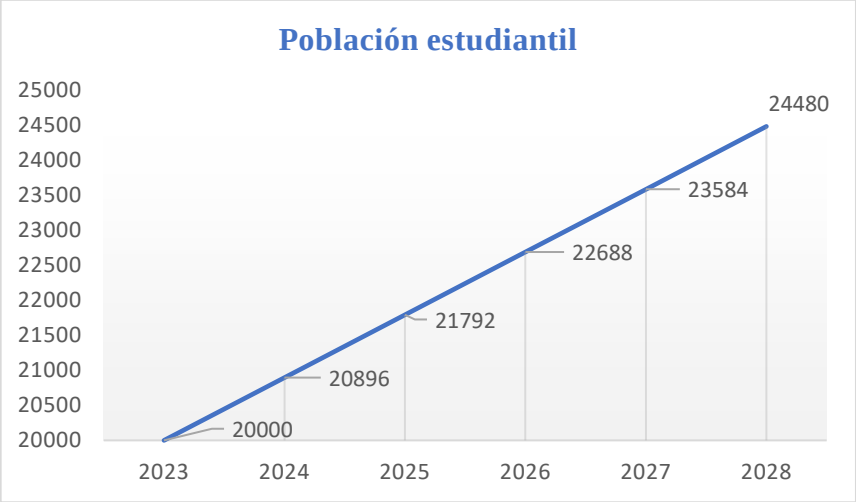


Figura 10-3: Ejecución de la primera prueba de estrés al sistema académico.

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

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 de tiempo.

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 (S. Ibarra, comunicación personal, 17 de noviembre de 2022). 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 Ecuación 2.

Ecuación 2: Proporción directa para conocer el número de estudiantes conectados en 5 años.

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

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

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 Ecuación 3.

Ecuación 3: Proporción directa para conocer el número de estudiantes conectados en 5 años.

$$\frac{18360\ \text{estudiantes}}{540\ \text{minutos}} = \frac{34\ \text{estudiantes}}{1\ \text{minuto}}$$

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 Ecuación 4. Mientras que, en el nivel alto, se toma el número total de estudiantes considerando las 9 horas, como se observa en la Ecuación 5.

Ecuación 4: Proporción directa para conocer el número de estudiantes conectados en el nivel medio.

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

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

Ecuación 5: Proporción directa para conocer el número de estudiantes conectados en el nivel medio.

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

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.

3.4.4. Fase 4: Preparar el entorno de la prueba

Para preparar el escenario de pruebas, se debe comenzar a configurar el Proxy. Como se visualiza en la Figura 11-3, 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.

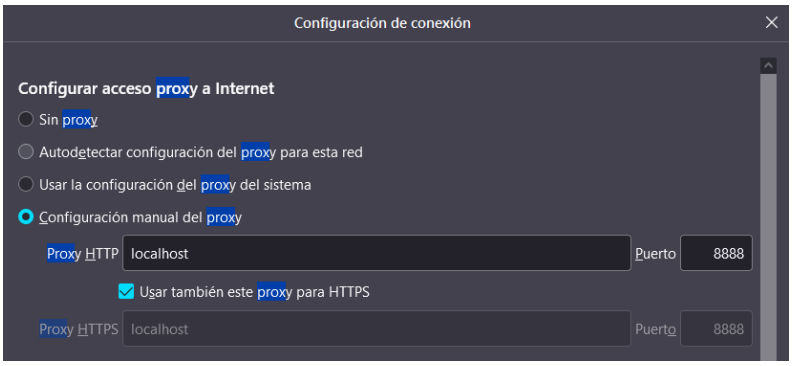


Figura 11-3: Configuración manual del Proxy en el navegador de Firefox.

Fuente: Apache Software Foundation, 2022

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, esto se observa en la Figura 12-3.
- **Controlador Objetivo**: este campo guarda la información de las peticiones que se hacen en el proceso de matrículas. Para hacerlo, se debe dar clic en Test Sistema Académico ESPOCH > Estudiantes en el proceso de matrículas, como se observa en la Figura 10-3.
- **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. Como se muestra en la Figura 10-3, la primera transacción es el HomePage.

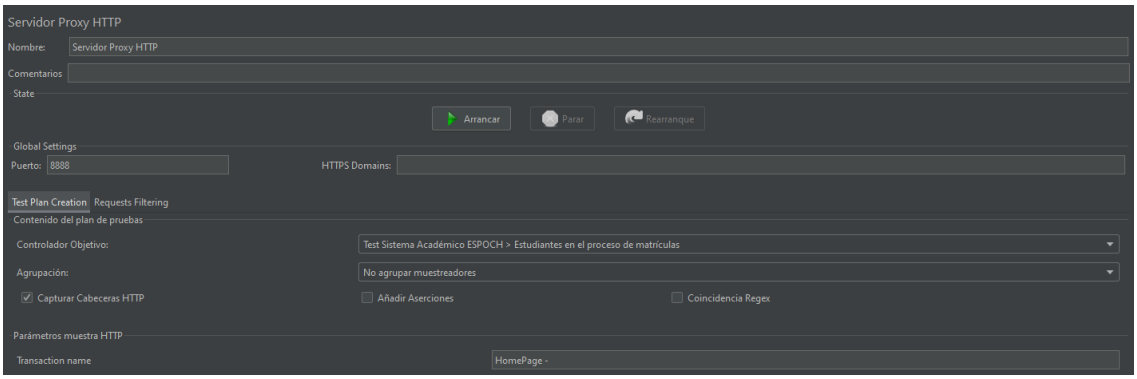


Figura 12-3: Configuración del Servidor Proxy HTTP en JMeter

Fuente: Apache Software Foundation, 2022

Antes de iniciar el proceso de grabación de la secuencia de peticiones al sistema académico, se debe considerar que el programa toma en cuenta todas las peticiones que se hacen en la aplicación, por lo que, es recomendable que no se muestre las llamadas a css, a imágenes, entre otros archivos. Para hacerlo se debe ir a ventana de *Requests Filtering*, y dar clic en *Add suggested Excludes* como se visualiza en la Figura 13-3.

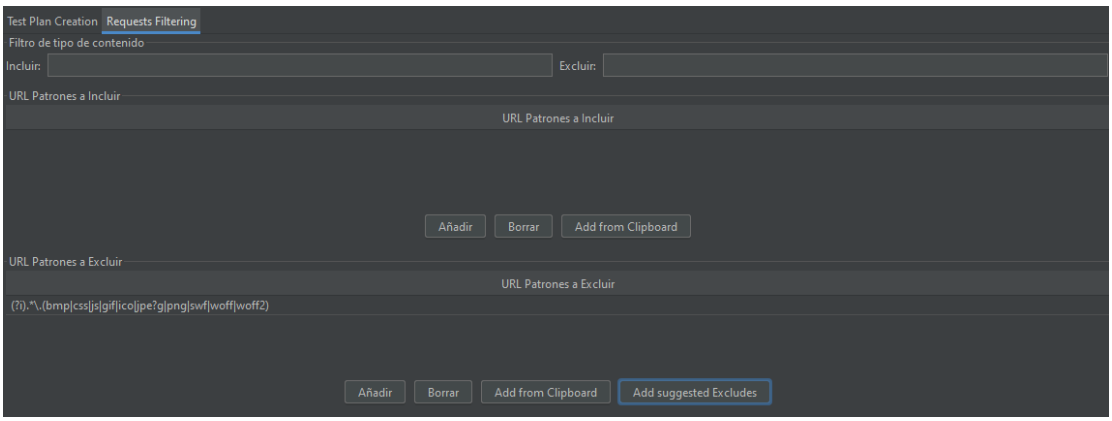


Figura 13-3: Configuración del Servidor Proxy HTTP en JMeter

Fuente: Apache Software Foundation, 2022

Al dar clic en Arrancar, se inicia la grabación de peticiones al servidor del sistema académico durante el proceso de matrículas de un estudiante. En la Figura 14-3 se muestra esta secuencia.

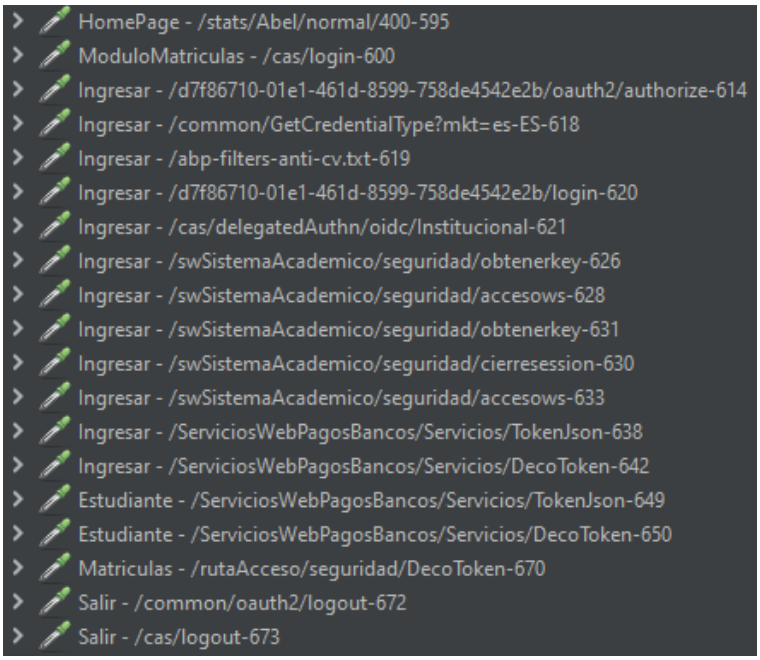


Figura 14-3: Secuencia de peticiones hechas al sistema académico.

Fuente: Apache Software Foundation, 2022

A continuación, se debe colocar un árbol de resultados y un resumen de reporte para poder visualizar los resultados, como se observa en la Figura 15-3.

Figura 15-3: Ítems Árbol de Resultados y Reporte resumen de la herramienta JMeter.

Fuente: Apache Software Foundation, 2022

En el árbol de resultados se muestra cada solicitud hecha, si se hizo correctamente o si hubo un error; esta información se muestra en la columna con el ítem Texto, como se puede observar en la Figura 16-3. Por otro lado, en el reporte se muestra la cantidad de solicitudes hechas al servidor, el camino que ha recorrido cada solicitud, el tiempo que ha tomado cada una, el porcentaje del error total de las peticiones realizadas y el promedio; esto se puede observar en la Figura 17-3.

Figura 16-3: Contenido del Árbol de Resultados de la herramienta JMeter.

Fuente: Apache Software Foundation, 2022

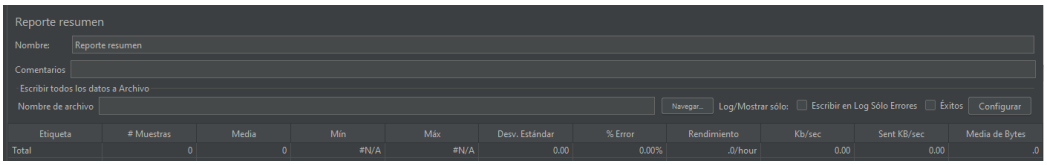


Figura 17-3: Reporte de resultados de la herramienta JMeter.

Fuente: Apache Software Foundation, 2022

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. En la Figura 18-3 se visualiza esta herramienta.

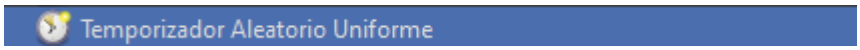


Figura 18-3: Temporizador Aleatorio Uniforme de JMeter.

Fuente: Apache Software Foundation, 2022

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 Ecuación 6.

Ecuación 6: Fórmula para calcular la cantidad de estudiantes por semestre.

$$\text{Ejecución aleatoria N: } 0.X * \text{Valor Random} + \text{Valor Constante}$$

$$\text{Ejecución aleatoria 1: } 0.8 * 5000 + 0 = 4000 \text{ ms} \rightarrow 4 \text{ segundos de retardo}$$

Tomando en cuenta estos parámetros, el máximo retardo aleatorio es de 5000.0 segundos y el desplazamiento de retraso es de 0 segundos, como se muestra en la Figura 19-3.

Figura 19-3: Configuración del Temporizador Aleatorio Uniforme.

Fuente: Apache Software Foundation, 2022

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:** $^{34\ \text{hilos}}/_{1\ \text{minuto}} \rightarrow {}^{34\ \text{hilos}}/_{60\ \text{segundos}}$
- **Medio:** $^{40\ \text{hilos}}/_{1\ \text{minuto}} \rightarrow {}^{40\ \text{hilos}}/_{60\ \text{segundos}}$
- **Alto:** $^{46\ \text{hilos}}/_{1\ \text{minuto}} \rightarrow {}^{46\ \text{hilos}}/_{60\ \text{segundos}}$

3.4.4.1. Nivel básico de peticiones enviadas al sistema académico

- **Número de Hilos:** Es la cantidad de procesos en un intervalo de tiempo. Como se observa en la Figura 20-3, 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 20-3, 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.

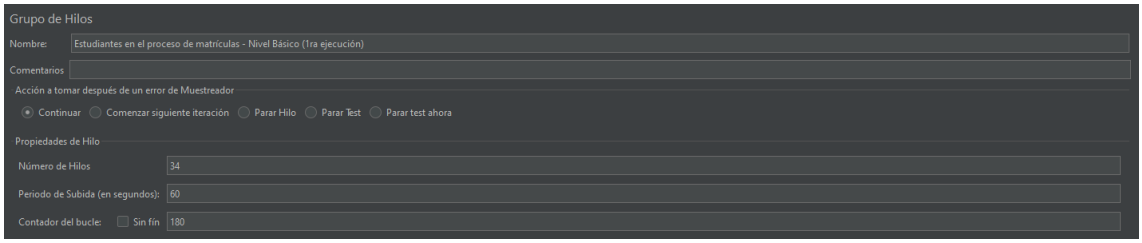


Figura 20-3: Configuración del grupo de hilos para el nivel básico.

Fuente: Apache Software Foundation, 2022

3.4.4.2. Nivel medio de peticiones enviadas al sistema académico

- **Número de Hilos:** Es la cantidad de procesos en un intervalo de tiempo. Como se observa en la Figura 21-3, son 40 hilos.
- **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 21-3, se ejecutarán 40 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.

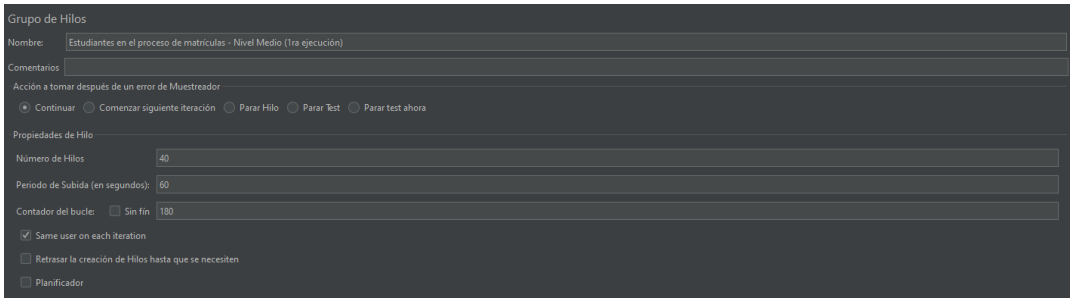


Figura 21-3: Configuración del grupo de hilos para el nivel medio.

Fuente: Apache Software Foundation, 2022

3.4.4.3. Nivel alto de peticiones enviadas al sistema académico

- **Número de Hilos:** Es la cantidad de procesos en un intervalo de tiempo. Como se observa en la Figura 22-3, son 46 hilos.
- **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 22-3, se ejecutarán 46 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.

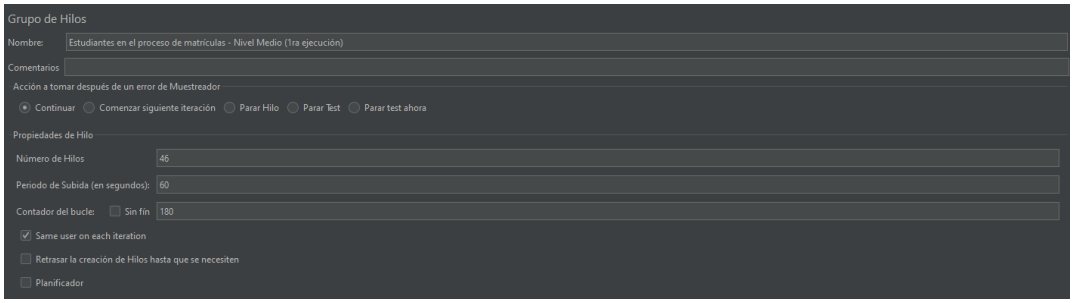


Figura 22-3: Configuración del grupo de hilos para el nivel alto.

Fuente: Apache Software Foundation, 2022

3.4.5. Fase 5: Ejecutar las pruebas de estrés al sistema académico.

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. Para hacerlo se debe dar clic en el botón de inicio en la barra de herramientas de JMeter, como se indica en la Figura 21-3.

Figura 23-3: Barra de herramientas de JMeter.

Fuente: Apache Software Foundation, 2022

En la Figura 22-3 se observa la ejecución de una prueba de estrés en el nivel básico, se muestra el número de peticiones por cada URL, el porcentaje de error y el rendimiento por cada una.

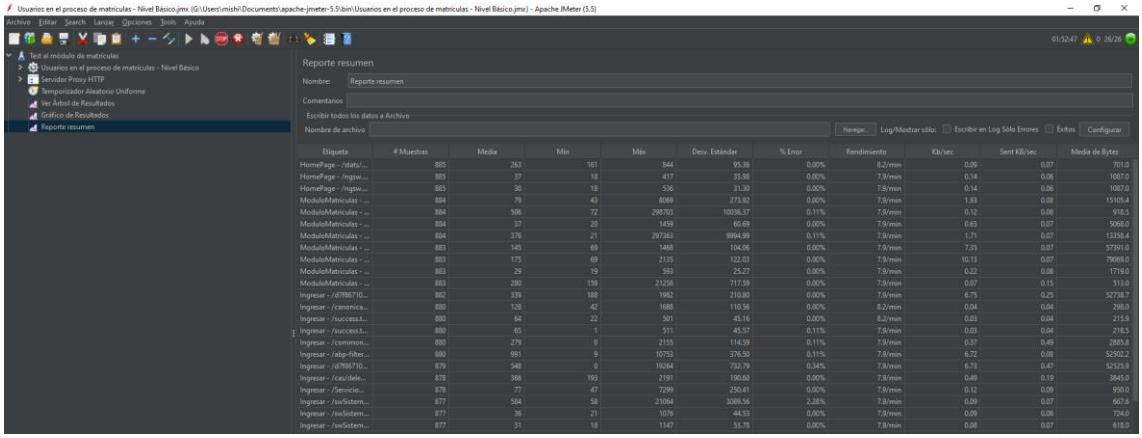


Figura 24-3: Ejecución de la primera prueba de estrés al sistema académico.

Fuente: Apache Software Foundation, 2022

Una vez detalladas las fases para la ejecución de las pruebas de estrés, se procede a describir el estudio de factibilidad, para conocer si es factible hacer las pruebas.

3.4.6. Fase 6: Análisis de resultados de las pruebas de estrés en el sistema académico de la ESPOCH

En el capítulo de resultados se detalla el análisis de resultados de las pruebas de estrés ejecutadas.

3.4.7. Fase 7: Estrategias para mejorar el rendimiento del sistema académico de la ESPOCH

En el capítulo de resultados se detallan las estrategias para mejorar el rendimiento del sistema académico.

CAPÍTULO IV

4. ANÁLISIS E INTERPRETACIÓN DE RESULTADOS

En este capítulo se describe el plan de mejores prácticas de vulnerabilidades y las estrategias para mejorar el rendimiento del sistema académico, en base al análisis y pruebas de estrés realizadas anteriormente.

4.1. Propuesta de mejores prácticas

Seguidamente, se realiza una propuesta para mejorar la seguridad del sistema académico, puesto que es esencial para garantizar la privacidad, integridad y confidencialidad de los datos y la información de los estudiantes, profesores y personal administrativo.

4.1.1. Ausencia de fichas (tokens) Anti-CSRF

Una solución para abordar está vulnerabilidad es la implementación de tokens Anti-CSFR en las solicitudes HTTP. Estos tokens son valores únicos generados para cada usuario y enviados en cada solicitud a través de una cookie o un campo oculto en un formulario HTML. Cuando el servidor recibe una solicitud, verifica si el token en la solicitud coincide con el asociado al usuario, si no coincide, la solicitud es rechazada.

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

Con la finalidad de prevenir ataques de inyección y mejorar la seguridad de la aplicación, se puede implementar la política de seguridad de contenido (CPS) en el encabezado de los servidores del sistema. Por ejemplo, se puede agregar la siguiente línea de código en el archivo de encabezado de la aplicación:

Content-Security-Policy: <policy-directive>

Donde "<policy-directive>" es la política de seguridad que se desea establecer. Para bloquear el contenido externo se puede usar la siguiente política

Content-Security-Policy: default-src 'self'

Esto permitirá que el contenido de la aplicación se cargue exclusivamente desde el origen. Cabe mencionar que, esta política puede ajustarse de acuerdo con los requerimientos de seguridad.

4.1.3. Falta el encabezado antisecuestro de clics

Para solucionar esta vulnerabilidad, se recomienda el siguiente encabezado a las respuestas del servidor:

Protección X-XSS: 1; modo = bloque

Este encabezado indica al navegador que habilita la protección contra ataques XSS y que bloquee automáticamente cualquier script sospechoso.

4.1.4. Biblioteca JS vulnerable

Para reducir esta vulnerabilidad, se requiere mantener actualizadas las bibliotecas y los frameworks utilizados en la aplicación. Asimismo, se recomienda monitorear periódicamente las aletas de seguridad y parches disponibles para las bibliotecas utilizadas. Además, es recomendable utilizar herramientas de detección de vulnerabilidades y estas pruebas deben ser periódicas. Es importante tener en cuenta que esta vulnerabilidad puede ser abordada en las fases del ciclo de vida de la aplicación como; en la arquitectura, diseño, implementación y despliegue de la aplicación.

4.1.5. Cookie sin atributo SameSite

Una forma de solucionar esta debilidad es establecer el atributo SameSite en la cookie para limitar el envío de solicitudes en el sitio web. SameSite puede ser agregado a la configuración del servidor o de la página web, como se describe en la siguiente línea de código:

Set-Cookie: <cookie_name>=<cookie_value>; SameSite=Strict;

Cabe mencionar que hay tres valores posibles para el atributo “SameSite”: “Strict”, “Lax” y “None”. Es recomendable usar “**Strict**” o “**Lax**” para mejorar la seguridad de la cookie, ya que “Strict” solo permite que la cookie se produzca con solicitudes desde el mismo sitio, mientras que “Lax” permite que la cookie se envié con solicitudes de navegación y “**None**” permite que la cookie se envié con cualquier solicitud. Es importante destacar que esta solución debería ser implementada en las fases de arquitectura, implementación y despliegue.

4.1.6. Divulgación de la marca de hora - Unix

Esta debilidad se puede mitigar, configurando el sistema para que no se divulgue la marca de tiempo en el protocolo de red o instalar un parche de seguridad para el sistema operativo. Además, se recomienda monitorear y auditar regularmente los registros de seguridad, mantener actualizado el sistema operativo y los programas instalados.

4.1.7. El servidor divulga información mediante un campo de encabezado de respuesta HTTP ""X-Powered-By""

Para solucionar esta debilidad, se recomienda desactivando o eliminando los encabezados "X-Powered-By" en la configuración del servidor web, o configurándolos para enviar un valor genérico en lugar de información específica sobre la tecnología y la versión utilizadas. Además, es importante mantener actualizadas las aplicaciones y las tecnologías en el servidor, así como monitorear regularmente la actividad y los registros de seguridad.

4.1.8. Divulgación de IP privada

Se puede solucionar esta vulnerabilidad se debe configurando el sistema o el firewall para enmascarar la dirección IP privada. Además, es importante monitorear y auditar periódicamente la actividad y los registros de seguridad. Por otra parte, se recomienda mantener actualizado el sistema.

4.1.9. Encabezado de respuesta del servidor HTTP

Para mitigar esta vulnerabilidad se debe seguir los siguientes pasos:

- **Desactivar la cabecera Server:** en la mayoría de los servidores es posible desactivar la cabecera Server para ocultar información sobre el sistema académico y el software del servidor.
- **Configurar la cabecera Server de forma segura:** en algunos casos, se puede configurar la cabecera Server para que solo muestre información básica.
- **Utilizar un servidor web seguro:** ciertos servidores web cuentan con medidas de seguridad integradas para mitigar esta vulnerabilidad, por lo que es recomendable utilizar un servidor web seguro.
- **Utilizar un intermediario de seguridad:** usar un firewall o un proxy inverso puede ayudar a ocultar información sobre la aplicación y el software del servidor.

- **Monitorear la cabecera Server:** es importante supervisar regularmente la cabecera Server para asegurarse de que no se revele información sensible.

Al usar estas recomendaciones, es posible mitigar la vulnerabilidad, proteger la aplicación y el servidor web ante posibles ataques.

4.1.10. Encabezado de seguridad de transporte estricto

Para mitigar esta vulnerabilidad, se debe seguir los siguientes pasos:

- **Implementar SSL/TLS:** es un protocolo de seguridad que permite asegurar la transmisión de información entre el cliente y el servidor.
- **Configurar el encabezado Strict-Transport-Security:** el encabezado permite que el servidor informe al navegador que solo debe conectarse mediante SSL/TLS, incluso si el usuario intenta conectarse mediante HTTP.
- **Implementar HSTS preload list:** es una lista de sitios web que se cargan automáticamente con SSL/TLS y que no permiten conexiones HTTP.
- **Monitorear el sitio web:** es importante monitorear el sitio web para detectar y asegurarse de que cumplan los requisitos de seguridad.

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

Para solucionar esta debilidad, se debe implementar lo siguiente:

- **Agregar el encabezado X-Content-Type-Option:** permite que el servidor informe al navegador que no deben ejecutarse scripts maliciosos en la página web.
- **Configurar el encabezado X-Content-Type-Option:** es considerable configurar el encabezado X-Content-Type-Option para que el navegador no ejecute scripts maliciosos en la página web.
- **Verificar la configuración del encabezado X-Content-Type-Option:** es importante verificar que esté configurado correctamente para proteger la página web.
- **Monitorear el sitio web:** es fundamental monitorear el sitio web para detectar posibles ataques y asegurarse de que se cumplan los requisitos de seguridad.

4.1.12. Divulgación de información - Comentarios sospechosos

Esta vulnerabilidad puede ser mitigada aplicando los siguiente:

- **Eliminar los comentarios innecesarios del código fuente:** estos pueden contener información sensible o sospechosa que puede ser utilizada por atacantes para comprometer la seguridad de la aplicación.
- **Utilizar herramientas de escaneo de seguridad:** estas pueden identificar y reportar posibles comentarios sospechosos o información sensible en el código fuente.
- **Establecer una política de comentarios:** es importante establecer estas políticas las cuales deben ser claras y estrictas sobre el uso de comentarios en el código fuente para asegurar que los comentarios sean útiles y no contengan información sensible.
- **Monitorear regularmente el código fuente:** es fundamental comprobar periódicamente el código fuente en busca de comentarios sospechosos o datos privados que pueden poner en peligro la seguridad de la aplicación.

4.1.13. Aplicación web moderna

Se trata de una vulnerabilidad de aplicaciones web modernas que, en algunos casos, no requiere cambios porque es una alerta informativa. No obstante, es fundamental que se apliquen medidas de seguridad adicionales, como cifrado de datos, el uso de herramientas de escaneo de seguridad y actualización regular del software, entre otros.

4.1.14. Reexaminar las directivas de control de caché

Para dar solución a esta vulnerabilidad, se debe implementar lo siguiente:

- **Configurar correctamente las directivas de control de caché:** se debe de configurar correctamente para asegurarse de que la información sensible no se almacene en la caché.
- **Utilizar encabezados HTTP:** esos encabezados deben ser los adecuados para controlar la forma en que los navegadores manejan la caché.
- **Evitar guardar información sensible en la caché:** para proteger los datos confidenciales, se debe evitar guardar en la memoria caché, especialmente las contraseñas.
- **Monitorear la caché:** Es importante realizar esta medida regularmente para detectar posibles vulnerabilidades y corregirlas.

4.1.15. Fuzzer de agente de usuario

Para solucionar esta alerta se debe poner en práctica los pasos que se mencionan a continuación:

- **Validación de entrada de datos:** Es importante validar adecuadamente todos los datos de entrada para evitar ataques de fuzzer de agente de usuario.
- **Filtrar adecuadamente los datos de entrada:** Es importante realizar esta acción para evitar que se el sistema reciba datos maliciosos.
- **Realizar pruebas de penetración:** Es necesario realizar este tipo de pruebas para detectar posibles vulnerabilidades y corregirlas.
- **Monitorear el registro de la aplicación:** Se debe realizar esta acción para detectar posibles ataques de fuzzer de agente de usuario.
- **Utilizar una solución de seguridad:** Es importante verificar que las URL de la aplicación den la misma respuesta a distintos usuarios.

Una vez propuestas las mejores prácticas para mejorar la seguridad del sistema académico. A continuación, se describen los resultados de las pruebas de estrés hechas al sistema académico de la ESPOCH.

4.2. Análisis de resultados de las pruebas de estrés en el sistema académico de la ESPOCH

Antes de analizar los resultados, se debe tomar en cuenta que se ejecutó tres veces cada prueba en función de cada nivel definido para visualizar el comportamiento del sistema académico en función a la cantidad de peticiones enviadas al servidor en total. Con respecto a lo mencionado, se continua con el análisis de resultados según cada nivel ejecutado:

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

Los resultados de las tres pruebas de estrés ejecutadas en función al nivel básico se pueden visualizar en la Tabla 24-4.

Tabla 24-4: Resultados de las tres pruebas de estrés ejecutadas en función del nivel básico

Parámetros	1ra ejecución	2da ejecución	3ra ejecución	Promedio
# procesos simulados en total	6120	6120	6120	6120
# peticiones en total	116.280	116.280	116.280	116.280
error/# total de peticiones	0.75 %	1.21%	0.66%	0.87%
Rendimiento (peticiones/minuto)	720 pet/min	726 pet/min	729 pet/min	725 pet/min

Fuente: Apache Software Foundation, 2022

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

El análisis de los resultados presentados en la tabla 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 en comparación con las otras se muestran en la Tabla 25-4.

Tabla 25-4: 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 %

Fuente: Apache Software Foundation, 2022

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

4.2.2. Análisis de resultados de las pruebas de estrés en función del nivel medio:

Los resultados de las tres pruebas de estrés ejecutadas en función del nivel medio se pueden visualizar en la Tabla 26-4.

Tabla 26-4: Resultados de las tres pruebas de estrés ejecutadas considerando el nivel medio

Parámetros	1ra ejecución	2da ejecución	3ra ejecución	Promedio
# procesos simulados en total	7200	7200	7200	7200
# peticiones en total	136.800	136.800	136.800	136.800
error/# total de peticiones	0.46 %	0.78%	0.44%	0.56%
Rendimiento (peticiones/minuto)	872 pet/min	835 pet/min	865 pet/min	857 pet/min

Fuente: Apache Software Foundation, 2022

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

El análisis de los resultados presentados en la tabla 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 27-4.

Tabla 27-4: Resultados de las tres pruebas de estrés ejecutadas considerando el nivel medio

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 %

Fuente: Apache Software Foundation, 2022

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

4.2.3. Análisis de resultados de las pruebas de estrés en función del nivel alto:

Los resultados de las tres pruebas de estrés ejecutadas en función del nivel alto se pueden visualizar en la Tabla 28-4.

Tabla 28-4: Resultados de las tres pruebas de estrés ejecutadas en función del nivel alto

Parámetros	1ra ejecución	2da ejecución	3ra ejecución	Promedio
# procesos simulados en total	8280	8280	8280	8280
# peticiones en total	157.320	157.320	157.320	157.320
error/# total de peticiones	1.73 %	0.32%	0.32%	0.79%
Rendimiento (peticiones/minuto)	995 pet/min	996 pet/min	996 pet/min	996 pet/min

Fuente: Apache Software Foundation, 2022

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

El análisis de los resultados presentados en la tabla 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 29-4.

Tabla 29-4: Resultados de las tres pruebas de estrés ejecutadas en función del nivel alto

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 %

Fuente: Apache Software Foundation, 2022

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

Se debe considerar que el rendimiento que presenta el sistema académico es relativamente bueno de acorde a los criterios de los desarrolladores del área de DTIC's; sin embargo, debe mejorar para responder correctamente a todas las peticiones.

De igual manera, presenta errores en las peticiones, como se observa, en los dos primeros niveles presentan errores en las mismas peticiones; no obstante, en el nivel alto se aumenta una petición más que presenta un error significativo.

4.3. Estrategias para mejorar el rendimiento del sistema académico de la ESPOCH

- Se recomienda aumentar un nodo de respaldo para cada uno de los procesos que maneja el sistema académico, porque si un nodo cae, el nodo de respaldo podría continuar con las peticiones y el sistema no colapsaría. Cabe recalcar que, si se aumenta un nodo, se aumenta la cantidad de recursos para el sistema académico; es decir, a nivel de interfaz de usuario, un nodo con 8 procesadores y 12 GB en memoria RAM; de igual manera, a nivel de backend se aumenta una cantidad similares de recursos.
- Se recomienda revisar las peticiones que presentan un porcentaje de error considerable, porque entre más errores se presenten en las solicitudes menor será el rendimiento del sistema académico.

CAPÍTULO V

CONCLUSIONES

- Se analizó las características de la metodología OWASP para detectar las vulnerabilidades; por otra parte, se definió el procedimiento para realizar las pruebas de estrés. Se concluye, que es una metodología efectiva y ampliamente utilizada en la industria de la seguridad del software, debido a su enfoque colaborativo y facilidad de uso en cuanto a seguridad de aplicaciones se refiere.
- En síntesis, después de aplicar la metodología OWASP en el análisis de vulnerabilidades en el sistema académico, se concluye qué es altamente eficiente para identificar las debilidades y proporcionar una visión completa y objetiva de las vulnerabilidades. La herramienta OWASP ZAP es muy eficaz, rápida y genera un informe detallado que muestra las debilidades más críticas, lo que garantizar la protección de la información sensible y la seguridad del sistema académico en el futuro.
- Se realizaron pruebas de estrés en el sistema académico para evaluar su rendimiento cuando es sometido a una gran carga de peticiones, específicamente simulando el proceso de matrículas. Se detectó que el servidor puede manejar la sobrecarga; sin embargo, presenta algunos errores con algunas URL que se cargan, lo que podría causar que el servidor colapse. Para mejorar el rendimiento, se sugiere agregar un nodo de respaldo en caso de que el nodo principal falle y revisar las páginas que dan error.
- La propuesta de mejores prácticas puede mejorar la seguridad del sistema académico. Es esencial garantizar la privacidad y la integridad de la información sensible a través de la mejora de autenticación, monitoreo, registro de actividad y actualización constante del software. Además, es necesario mantener copias de seguridad regulares y seguir las recomendaciones presentadas en el apartado de mejores prácticas para garantizar la seguridad a largo plazo del sistema académico.

RECOMENDACIONES

- En cuanto a las herramientas y procedimientos utilizados para realizar las pruebas de estrés, se recomienda utilizar herramientas que permitan hacer pruebas automatizadas y seguras. Además, es importante seguir un procedimiento estandarizado y documentado para realizar pruebas de estrés de una manera eficiente.
- Se recomienda realizar periódicamente el análisis de vulnerabilidades al sistema académico para evaluar continuamente las vulnerabilidades y priorizar las más críticas, lo que ayudará a garantizar la protección de los datos más sensibles y sobre todo a mejorar la seguridad del sistema.
- Se sugiere identificar y corregir los errores encontrados en las pruebas de estrés; de igual manera, se recomienda mejorar la infraestructura del sistema académico, aumentando un nodo de respaldo para cada módulo de la aplicación.
- Para mejorar la seguridad del sistema académico es importante aplicar las buenas prácticas para mitigar las vulnerabilidades encontradas en el sistema académico. Asimismo, es importante mantener actualizado los softwares utilizados, realizar pruebas de estrés para verificar la disponibilidad del sistema y realizar análisis de vulnerabilidades periódicamente para detectar posibles debilidades y fallos.

GLOSARIO

Agente de usuario: Es un software que actúa en nombre del usuario y se conecta con el servidor con el fin de realizar las acciones específicas en la aplicación web (BorealOS, 2018).

Autenticación: Es el proceso de verificación de la identidad de un usuario, dispositivo o sistema antes de permitir el acceso a recursos restringidos o confidenciales (Pacheco, 2010, p. 5).

Cabecera de Host: Es un identificador de una aplicación, la cual es utilizada cuando existen múltiples aplicaciones web alojadas en una sola dirección IP, con la finalidad de que cada una cumpla con su función dependiendo de la solicitud (the Mozilla Foundation, 2022a).

Código malicioso: Es un término que se refiere a cualquier software o fragmento de software diseñado para dañar, interferir o robar información de un sistema informático (García Monje, 2017, p. 3).

CWE: Sus siglas significan "Enumeración Común de Debilidades", se trata de una lista amplia, consolidada de debilidades y vulnerabilidades comunes en el software (The MITRE Corporation, 2023h).

Exploit: Es una técnica diseñada para aprovechar una debilidad o vulnerabilidad en el software (Bustamante Perez, 2019, p. 7)

HSTS: Significa "Seguridad Estricta de transporte HTTP", es un mecanismo de seguridad de la capa de transporte que permite a un sitio web informar a los navegadores que solo deben comunicarse con él a través de conexiones seguras HTTPS (CERT-PY, 2020, p. 1).

LDAP: Es un protocolo que se usa para dividir y ordenar las listas de información de los árboles de los directorios en una base de datos (Bach Nutman, 2020, p. 1).

MIME: Significa "Extensiones Multiuso para Correo Internet", es un estándar que permite a los mensajes de correo electrónico contener múltiples tipos de datos, como texto, imágenes, audio y video (the Mozilla Foundation, 2022b).

Nonce: En el campo de la criptografía, un nonce es un número aleatorio que no se repite (Centro Criptológico Nacional, 2018).

Nosniff: Es una cabecera HTTP que se utiliza para prevenir que el navegador web interprete o ejecute contenido malicioso o inesperado (the Mozilla Foundation, 2022c).

OWASP: Es una fundación sin fines de lucro que trabaja para mejorar la seguridad del software, a través de los proyectos de software de código abierto liderados por la comunidad. OWASP es la fuente para que los desarrolladores y tecnólogos protejan la web (OWASP Foundation, Inc., 2022b).

Rendimiento: El rendimiento es una métrica de calidad que mide la eficiencia de la aplicación al momento de usar los recursos hardware (Microsoft, 2022).

SDLC: Significa "Ciclo de vida del desarrollo del Software", es un proceso formalizado que se utiliza para planificar, desarrollar, probar y mantener sistemas informáticos, asegurándose que estos sean eficientes y fiables (Figueroa, 2016, p. 4).

Seguridad: La seguridad del software hace referencia a implementar mecanismos de seguridad en la construcción del sistema para que permanezca funcional ante los ataques (Thales, 2022).

Sistema en producción: Es un sistema informático que está en uso activo y en funcionamiento en un entorno de producción, el cual se utiliza para brindar servicios o realizar tareas para los usuarios finales (Quiroa, 2023).

SPA: significa "Aplicación de página única", es un patrón de diseño de aplicaciones web, el cual consiste en que se ejecuta una sola página, el contenido de esta depende de los datos que se envíen desde el backend (the Mozilla Foundation, 2023).

Tester: Es una persona encargada de planificar y ejecutar pruebas en el software que se esté trabajando, para hacerlo debe tener en cuenta las entradas, la función que realiza y lo que debe dar como resultado el sistema (Whittaker, 2000, p.71)

Vulnerabilidad: Es una debilidad a nivel de software, causada por fallos en la implementación o por errores humano, que posibilita a un atacante aprovecharla para causar daño (Rodríguez Rodríguez y Sánchez Sánchez, 2018).

XSS: Sus siglas significan "Inyección de código en sitios cruzados", es un ataque en el que un atacante inyecta código malicioso, comúnmente en forma de scripts en un sitio web legítimo, con el fin de ejecutar ese código en el navegador de un usuario, quien accede al sitio web (Caballero Quezada, 2021, p. 5).

BIBLIOGRAFÍA

AGILA TINOCO, V.P., 2019. ANÁLISIS DE VULNERABILIDADES, AMENAZAS Y RIESGOS AL SISTEMA DE MATRICULACIÓN DE LA UNIDAD ACADÉMICA DE CIENCIAS EMPRESARIALES DE LA UTMACH. Machala: Universidad Técnica de Machala.

AL KHURAFI, O. y AL AHMAD, M.A., 2015. Survey of Web Application Vulnerability Attacks. *2015 4th International Conference on Advanced Computer Science Applications and Technologies (ACSAT)* [en línea]. Kuala Lumpur, Malaysia: IEEE, pp. 154-158. [Consulta: 24 octubre 2022]. ISBN 978-1-5090-0423-2. DOI 10.1109/ACSAT.2015.46. Disponible en: http://ieeexplore.ieee.org/document/7478735/.

ANDRIAN, R. y FAUZI, A., 2020. Security Scanner For Web Applications Case Study: Learning Management System. *Jurnal Online Informatika*, vol. 4, no. 2, pp. 63. ISSN 2527-9165, 2528-1682. DOI 10.15575/join.v4i2.394.

APACHE SOFTWARE FOUNDATION, 2022. *Apache JMeter* [en línea]. Java. 2022. S.l.: Apache Software Foundation. Disponible en: https://jmeter.apache.org/download_jmeter.cgi.

ARÉVALO CORDOVILLA, F.E., ORDOÑEZ SIGCHO, I.B., PEÑAHERRERA LARENAS, M.F. y SUÁREZ MATAMOROS, V.J., 2020. Importancia de la seguridad de los sistemas de información frente el abuso, error y hurto de información. *Revista Científica Dominio de las Ciencias*, vol. 6, no. 2, pp. 12. ISSN 2477-8818. DOI http://dx.doi.org/10.23857/dc.v6i2.1197.

ARIAS PAREDES, A.S., 2019. *ANÁLISIS DE VULNERABILIDADES DE SERVIDORES VIRTUALES, CASO PRÁCTICO SERVICIOS WEB INFORMATIVOS DE LA ESPOCH* [en línea]. Propuesta Tecnológica. Riobamba: Escuela Superior Politécnica de Chimborazo. [Consulta: 10 febrero 2023]. Disponible en: http://dspace.espoch.edu.ec/bitstream/123456789/13631/1/98T00269.pdf.

ARYA WIRADARMA, A.A.B. y ARYA SASMITA, G.M., 2019. IT Risk Management Based on ISO 31000 and OWASP Framework using OSINT at the Information Gathering Stage (Case Study: X Company). *International Journal of Computer Network and Information Security*, vol. 11, no. 12, pp. 17-29. ISSN 20749090, 20749104. DOI **10.5815/ijcnis.2019.12.03.**

BACH NUTMAN, M., 2020. *Understanding The Top 10 OWASP Vulnerabilities* [en línea]. 17 diciembre 2020. S.l.: arXiv. [Consulta: 3 enero 2023]. arXiv:2012.09960. Disponible en: http://arxiv.org/abs/2012.09960.

BHUNIA, S. y TEHRANIPOOR, M., 2018. *Hardware Security: A Hands-on Learning Approach*. United States: Morgan Kaufmann Publishers. ISBN 978-0-12-812478-9.

BOREALOS, 2018. ¿Qué es un agent user o agente de usuario? *Boreal Open Systems - Agencia Digital* [en línea]. [Consulta: 5 febrero 2023]. Disponible en: https://borealos.com/post/que-es-un-agent-user-o-agente-de-usuario.html.

BUSTAMANTE PEREZ, D., 2019. *Desarrollo de exploits* [en línea]. 31 mayo 2019. S.l.: The OWASP Foundation. [Consulta: 6 febrero 2023]. Disponible en: https://owasp.org/www-pdf-archive//Exploit_development.pdf.

CABALLERO QUEZADA, A.E., 2021. *Cross-Site Scripting (XSS)* [en línea]. 12 agosto 2021. S.l.: s.n. [Consulta: 4 febrero 2023]. Disponible en: https://www.reydes.com/archivos/slides/webinars/AC_WG_Cross_Site_Scripting_XSS.pdf.

CAMPDERRÓS VILÀ, J., 2019. *Ataques y vulnerabilidades web* [en línea]. Trabajo de Final de Grado. Barcelona: Universitat de Barcelona. Disponible en: diposit.ub.edu/dspace/bitstream/2445/143419/2/memoria.pdf.

CARVACA ORRALA, A.L., 2022. *Análisis de seguridad controlado en aplicaciones web de una institución financiera utilizando herramientas de ciberseguridad y buenas prácticas de OWASP* [en línea]. Examen complexivo. La Libertad, Ecuador: Universidad Estatal Península de Santa Elena. Disponible en: https://repositorio.upse.edu.ec/bitstream/46000/8646/1/UPSE-TTI-2022-0030.pdf.

CENTRO CRIPTOLÓGICO NACIONAL, 2018. Glosario: Nonce. *GUÍA DE SEGURIDAD (CCN-STIC-401)* [en línea]. [Consulta: 5 febrero 2023]. Disponible en: https://www.ccn-cert.cni.es/publico/seriesCCN-STIC/series/400-Guias_Generales/401-glosario_abreviaturas/index.html?n=639.html.

CERT-PY, 2020. *Guía de Seguridad* [en línea]. 2 junio 2020. S.l.: CERT-PY. [Consulta: 5 febrero 2023]. Disponible en: https://www.cert.gov.py/application/files/4015/9250/1441/Guia_de_Seguridad_-__HSTS_un_mecanismo_de_seguridad_adicional_a_HTTPs.pdf.

CISCO SYSTEMS, INC., 2022. ¿Qué es la seguridad de red? *Cisco* [en línea]. [Consulta: 24 octubre 2022]. Disponible en: https://www.cisco.com/c/es_mx/products/security/what-is-network-security.html.

CONGOTE, E. y HINCAPIE, A., 2020. Pruebas Perfomance: tipos y etapas. *Academia pragma* [en línea]. [Consulta: 27 enero 2023]. Disponible en: http://www.pragma.com.co/academia/lecciones/pruebas-perfomance-tipos-y-etapas.

CUEVAS, J., MUÑOZ, R., DI GIONANTONIO, A., GASTAÑAGA, I., GIBELLINI, F., PARISI, G., BARRIONUEVO, D. y ZEA CÁRDENAS, M., 2018. Análisis de Vulnerabilidades de Sistemas Web en desarrollo y en producción. . Córdoba: Universidad Tecnológica Nacional.

FERNÁNDEZ SANGUINO, J., 2006. El proyecto OWASP Testing. *germinus*. Madrid: s.n., pp. 21.

FIGUEROA, V., 2016. Secure Software Development Life Cycle. *OWASP Latam Tour 2016* [en línea]. S.l.: The OWASP Foundation, pp. 31. Disponible en: https://owasp.org/www-pdf-archive/OWASP-LATAMTour-Patagonia-2016-rvfigueroa.pdf.

FUNDACIÓN MTP, 2022. PRUEBAS DE RENDIMIENTO. *MTP DIGITAL BUSINESS ASSURANCE* [en línea]. [Consulta: 26 octubre 2022]. Disponible en: https://www.mtp.es/aseguramiento-de-la-calidad/servicios-de-operacion/pruebas-de-rendimiento/.

GAMBOA SAFLA, D.L., 2021. *VULNERABILIDADES EN APLICACIONES WEB UTILIZANDO LA METODOLOGÍA DE «PROYECTO ABIERTO DE SEGURIDAD DE APLICACIONES WEB»*. Ambato: Pontifacia Universidad Católica del Ecuador.

GARCÍA MONJE, R.A., 2017. SEGURIDAD INFORMÁTICA Y EL MALWARE. *Universidad Piloto de Colombia*, pp. 11.

GRABOVSKY, S., CIKA, P., ZEMAN, V., CLUPEK, V., SVEHLAK, M. y KLIMES, J., 2018. Denial of Service Attack Generator in Apache JMeter. *2018 10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)* [en línea].

Moscow, Russia: IEEE, pp. 1-4. [Consulta: 25 octubre 2022]. ISBN 978-1-5386-9361-2. DOI 10.1109/ICUMT.2018.8631212. Disponible en: https://ieeexplore.ieee.org/document/8631212/.

HAMILTON, T., 2020. What is STRESS Testing in Software Testing? *GURU99* [en línea]. [Consulta: 2 febrero 2023]. Disponible en: https://www.guru99.com/stress-testing-tutorial.html.

HAN, X., 2021. A Study of Performance Testing in Configurable Software Systems. *Journal of Software Engineering and Applications*, vol. 14, no. 9, pp. 474-492. ISSN 1945-3116, 1945-3124. DOI 10.4236/jsea.2021.149028.

HARRELL, C.R., PATTON, M., CHEN, H. y SAMTANI, S., 2018. Vulnerability Assessment, Remediation, and Automated Reporting: Case Studies of Higher Education Institutions. *2018 IEEE International Conference on Intelligence and Security Informatics (ISI)* [en línea]. Miami, FL: IEEE, pp. 148-153. [Consulta: 25 octubre 2022]. ISBN 978-1-5386-7848-0. DOI 10.1109/ISI.2018.8587380. Disponible en: https://ieeexplore.ieee.org/document/8587380/.

HEGDE, V., 2014. Web Performance Testing: Methodologies, Tools and Challenges. , vol. 2, no. 1, pp. 7.

HUSUFA, N. y PRIHANDI, I., 2022. Optimizing JMeter on Performance Testing Using the Bulk Data Method. *Journal of Information Systems and Informatics*, vol. 4, no. 2, pp. 205-215. ISSN 2656-4882. DOI 10.51519/journalisi.v4i2.244.

HWANG, G.-H., WU-LEE, C., TUNG, Y.-H., CHUANG, C.-J. y WU, S.-F., 2014. Implementing TaaS-based stress testing by MapReduce computing model. *2014 IEEE 5th International Conference on Software Engineering and Service Science* [en línea]. Beijing, China: IEEE, pp. 137-140. [Consulta: 25 octubre 2022]. ISBN 978-1-4799-3279-5. DOI 10.1109/ICSESS.2014.6933530. Disponible en: http://ieeexplore.ieee.org/document/6933530/.

IBM CORPORATION, 2021. Pruebas de rendimiento. *IBM* [en línea]. [Consulta: 26 octubre 2022]. Disponible en: https://prod.ibmdocs-production-dal-6099123ce774e592a519d7c33db8265e-0000.us-south.containers.appdomain.cloud/docs/es/rtw/9.0.0?topic=phases-performance-testing.

IBM MQ, 2021. Identificación y autenticación. *IBM Documentation* [en línea]. [Consulta: 25 octubre 2022]. Disponible en: https://prod.ibmdocs-production-dal-6099123ce774e592a519d7c33db8265e-0000.us-south.containers.appdomain.cloud/docs/es/ibm-mq/7.5?topic=ssfksj-7-5-0-com-ibm-mq-sec-doc-q009740--htm.

ISO25000.COM, 2022. ISO 25010. *ISO/IEC 25010* [en línea]. [Consulta: 27 enero 2023]. Disponible en: https://iso25000.com/index.php/normas-iso-25000/iso-25010?start=6.

JIANG, Z.M. y HASSAN, A.E., 2019. A Survey on Load Testing of Large-Scale Software Systems. *IEEE Transactions on Software Engineering*, vol. 41, no. 11, pp. 1091-1118. ISSN 0098-5589, 1939-3520. DOI 10.1109/TSE.2015.2445340.

KALITA, M. y BEZBORUAH, T., 2011. Investigation on performance testing and evaluation of PReWebD: a .NET technique for implementing web application. *IET Software*, vol. 5, no. 4, pp. 357. ISSN 17518806. DOI 10.1049/iet-sen.2010.0139.

KORNIENKO, D., MISHINA, S. y MELNIKOV, M., 2021. The Single Page Application architecture when developing secure Web services - IOPscience. *Journal of Physics: Conference Series*, pp. 13. DOI 10.1088/1742-6596/2091/1/012065.

KUMI, S., LIM, C., LEE, S.-G., OKTIAN, Y.O. y WITANTO, E.N., 2021. Automatic Detection of Security Misconfigurations in Web Applications. *Proceedings of International Conference on Smart Computing and Cyber Security*. Singapore: Springer, pp. 91-99. ISBN 9789811579905. DOI 10.1007/978-981-15-7990-5_8.

KUTHNIK, T., MARTIN, D., GERARDI, T., CORTES, I. y VERGARA, A., 2018. La Criptografía y la Seguridad Informática. , pp. 7.

LLAMUCA QUINALOA, J., VER VINCENT, Y. y TAPIA CERDA, V., 2021. Análisis comparativo para medir la eficiencia de desempeño entre una aplicación web tradicional y una aplicación web progresiva. *TecnoLógicas*, vol. 24, no. 51, pp. e1892. ISSN 2256-5337, 0123-7799. DOI 10.22430/22565337.1892.

MAÂLEJ, A.J., KRICHEN, M. y JMAÏEL, M., 2015. A comparative evaluation of state-of-the-art load and stress testing approaches. *International Journal of Computer Applications in Technology*, vol. 51, no. 4, pp. 283. ISSN 0952-8091, 1741-5047. DOI 10.1504/IJCAT.2015.070491.

MARTIN, B., 2019. Common Vulnerabilities Enumeration (CVE), Common Weakness Enumeration (CWE), and Common Quality Enumeration (CQE): Attempting to systematically catalog the safety and security challenges for modern, networked, software-intensive systems. *ACM SIGAda Ada Letters*, vol. 38, no. 2, pp. 9-42. ISSN 1094-3641. DOI 10.1145/3375408.3375410.

MCGRAW, G., 2004. Software security. *IEEE Security & Privacy*, vol. 2, no. 2, pp. 80-83. ISSN 1558-4046. DOI 10.1109/MSECP.2004.1281254.

MICROSOFT, 2022. Introducción al rendimiento de aplicaciones de Windows. *Microsoft* [en línea]. [Consulta: 6 febrero 2023]. Disponible en: https://learn.microsoft.com/es-es/windows/apps/performance/introduction.

NOBOA CEVALLOS, M.C. y CUENCA OBREGON, D.E., 2021. *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. Riobamba: Escuela Superior Politécnica de Chimborazo.

OWASP FOUNDATION, INC., 2010. The Open Web Application Security Project. S.l.:

OWASP FOUNDATION, INC., 2021. OWASP Top Ten. *OWASP* [en línea]. [Consulta: 2 febrero 2023]. Disponible en: https://owasp.org/www-project-top-ten/.

OWASP FOUNDATION, INC., 2022a. Open Source Foundation for Application Security. *OWASP Foundation* [en línea]. [Consulta: 21 junio 2022]. Disponible en: https://owasp.org/.

OWASP FOUNDATION, INC., 2022b. Who is the OWASP® Foundation? *OWASP* [en línea]. [Consulta: 17 julio 2022]. Disponible en: https://owasp.org/.

OWASP FOUNDATION, INC., 2023. Clickjacking. *OWASP Foundation* [en línea]. [Consulta: 3 febrero 2023]. Disponible en: https://owasp.org/www-community/attacks/Clickjacking.

OWASP TOP 10 TEAM, 2021a. A04 Diseño Inseguro - OWASP Top 10:2021. *OWASP Top 10:2021* [en línea]. [Consulta: 4 enero 2023]. Disponible en: https://owasp.org/Top10/es/A04_2021-Insecure_Design/.

OWASP TOP 10 TEAM, 2021b. Introducción - OWASP Top 10:2021. *OWASP Top 10:2021* [en línea]. [Consulta: 24 octubre 2022]. Disponible en: https://owasp.org/Top10/es/A00_2021_Introduction/.

PACHECO, C., 2010. Implementación de Autenticación de Usuarios con Múltiples Credenciales. *Jornadas Argentinas de Informática e Investigación Operativa* [en línea]. Buenos Aires: National Scientific and Technical Research Council, pp. 13. [Consulta: 4 febrero 2023]. DOI 10.13140/2.1.1496.9927. Disponible en: https://www.researchgate.net/publication/269632012_Implementacion_de_Autenticacion_de_Usuarios_con_Multiples_Credenciales.

PACKETLABS, 2022. Cryptography Attacks: 6 Types & Prevention. *Packetlabs* [en línea]. [Consulta: 29 enero 2023]. Disponible en: https://www.packetlabs.net/posts/cryptography-attacks/.

PAYER, M., 2021. *Software Security* [en línea]. S.l.: s.n. [Consulta: 18 octubre 2022]. Disponible en: https://nebelwelt.net/SS3P/softsec.pdf.

PIESSENS, F., 2021. *Software Security Knowledge Area Version 1.0.1* [en línea]. julio 2021. S.l.: University of Bristol. [Consulta: 14 octubre 2022]. Disponible en: https://www.cybok.org/media/downloads/Software_Security_v1.0.1.pdf.

PINANGO BAYAS, Á.H., MÉNDEZ NARANJO, P.M., CAIZA MÉNDEZ, D.G. y BARRENO NARANJO, D.G., 2022. Plan de seguridad para plataformas web empleando normas iso-27001 y considerando el owasp top 10-2017. *Revista Ciencia UNEMI*, vol. 15, no. 40, pp. 1-15. ISSN 1390-4272.

PRADEEP, S. y KUMAR SHARMA, Y., 2019. A Pragmatic Evaluation of Stress and Performance Testing Technologies for Web Based Applications. *2019 Amity International Conference on Artificial Intelligence (AICAI)* [en línea]. Dubai, United Arab Emirates: IEEE, pp. 399-403. [Consulta: 25 octubre 2022]. ISBN 978-1-5386-9346-9. DOI 10.1109/AICAI.2019.8701327. Disponible en: https://ieeexplore.ieee.org/document/8701327/.

QUIROA, M., 2023. Sistema de producción. *Economipedia* [en línea]. [Consulta: 6 febrero 2023]. Disponible en: https://economipedia.com/definiciones/sistema-de-produccion.html.

RODRÍGUEZ RODRÍGUEZ, R.E. y SÁNCHEZ SÁNCHEZ, A.F., 2018. *DESARROLLO DE UN MODELO PARA CALCULAR EL NIVEL DE SEGURIDAD EN SITIOS WEB, BASADO EN EL TOP 10 DE VULNERABILIDADES MÁS EXPLOTADAS EN 2017 SEGÚN EL MARCO DE REFERENCIA OWASP*. Bogotá: Universidad Católica de Colombia.

ROMERO CASTRO, M.I., FIGUEROA MORÁN, G.L., VERA NAVARRETE, D.S., ÁLAVA CRUZATTY, J.E., PARRALES ANZÚLES, G.R., ÁLAVA MERO, C.J., MURILLO QUIMIZ, Á.L. y CASTILLO MERINO, M.A., 2018. *Introducción a la seguridad informática y el análisis de vulnerabilidades* [en línea]. 1. S.l.: Editorial Científica 3Ciencias. [Consulta: 13 octubre 2022]. ISBN 978-84-949306-1-4. Disponible en: https://www.3ciencias.com/libros/libro/introduccion-a-la-seguridad-informatica-y-el-analisis-de-vulnerabilidades/.

RUBÍN LINARES, G.T., 2021. *APORTACIONES DE LAS CIENCIAS COMPUTACIONALES DURANTE LA PANDEMIA COVID19*. Primera. Santiago Puebla: Montiel & Soriano Editores S.A. de C.V. ISBN 978-607-8728-80-0.

SAMANIEGO MENA, E.A. y PONCE ORDÓÑEZ, J.A., 2021. *Fundamentos de seguridad informática* [en línea]. Guayaquil, Ecuador: Editorial Grupo Compás. ISBN 978-9942-33-426-8.

Disponible en: https://www.researchgate.net/publication/354054517_Libro_Fundamentos_de_seguridad_informatica.

SANTIAGO GARCÍA, O.C., 2021. *OWASP COMO ELEMENTO ESTRATÉGICO EN LA IDENTIFICACIÓN DE VULNERABILIDADES Y LA VALIDACIÓN DE SEGURIDAD EN EL DISEÑO, PROGRAMACIÓN Y OPERACIÓN DE APLICACIONES SEGURAS EN LAS ORGANIZACIONES DESARROLLADORAS DE SOFTWARE EN COLOMBIA.* Villavicencio: Universidad Nacional Abierta y a Distancia -UNAD.

SCANREPEAT, 2020a. Information Disclosure - Suspicious Comments. *ScanRepeat* [en línea]. [Consulta: 30 enero 2023]. Disponible en: https://scanrepeat.com/web-security-knowledge-base/${'https://scanrepeat.com/' + path}.

SCANREPEAT, 2020b. Strict-Transport-Security Header Not Set. *ScanRepeat* [en línea]. [Consulta: 29 enero 2023]. Disponible en: https://scanrepeat.com/web-security-knowledge-base/strict-transport-security-header-not-set.

SCANREPEAT, 2020c. X-Content-Type-Options Header Missing. *ScanRepeat* [en línea]. [Consulta: 30 enero 2023]. Disponible en: https://scanrepeat.com/web-security-knowledge-base/${'https://scanrepeat.com/' + path}.

SIMBA, G., SOTO, S. y GÓMEZ, E., 2022. FACTORES CRÍTICOS EN PRUEBAS DE ESTRÉS EN SISTEMAS DE INFORMACIÓN TRANSACCIONALES. [en línea]. Sangolquí: Escuela Politécnica del Ejército. Disponible en: https://www.academia.edu/26896192/FACTORES_CRITICOS_EN_PRUEBAS_DE_ESTRES_EN_SISTEMAS_DE_INFORMACION_TRANSACCIONALES.

THALES, 2022. Seguridad del software | ¿Qué es la seguridad del software? *THALES Building a future we can all trust* [en línea]. [Consulta: 17 julio 2022]. Disponible en: https://cpl.thalesgroup.com/es/software-monetization/what-is-software-security.

THE MITRE CORPORATION, 2023a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor. *CWE - Common Weakness Enumeration* [en línea]. [Consulta: 2 febrero 2023]. Disponible en: https://cwe.mitre.org/data/definitions/200.html.

THE MITRE CORPORATION, 2023b. CWE-352: Cross-Site Request Forgery (CSRF). *CWE - Common Weakness Enumeration* [en línea]. [Consulta: 31 enero 2023]. Disponible en: https://cwe.mitre.org/data/definitions/352.html.

THE MITRE CORPORATION, 2023c. CWE-525: Use of Web Browser Cache Containing Sensitive Information. *CWE - Common Weakness Enumeration* [en línea]. [Consulta: 3 febrero 2023]. Disponible en: https://cwe.mitre.org/data/definitions/525.html.

THE MITRE CORPORATION, 2023d. CWE-693: Protection Mechanism Failure. *CWE - Common Weakness Enumeration* [en línea]. [Consulta: 31 enero 2023]. Disponible en: https://cwe.mitre.org/data/definitions/693.html.

THE MITRE CORPORATION, 2023e. CWE-829: Inclusion of Functionality from Untrusted Control Sphere. *CWE - Common Weakness Enumeration* [en línea]. [Consulta: 2 febrero 2023]. Disponible en: https://cwe.mitre.org/data/definitions/829.html.

THE MITRE CORPORATION, 2023f. CWE-1021: Improper Restriction of Rendered UI Layers or Frames. *CWE - Common Weakness Enumeration* [en línea]. [Consulta: 31 enero 2023]. Disponible en: https://cwe.mitre.org/data/definitions/1021.html.

THE MITRE CORPORATION, 2023g. CWE-1275: Sensitive Cookie with Improper SameSite Attribute. *CWE - Common Weakness Enumeration* [en línea]. [Consulta: 2 febrero 2023]. Disponible en: https://cwe.mitre.org/data/definitions/1275.html.

THE MITRE CORPORATION, 2023h. Viewing Customized CWE information. *CWE - Common Weakness Enumeration* [en línea]. [Consulta: 6 febrero 2023]. Disponible en: https://cwe.mitre.org/.

THE MOZILLA FOUNDATION, 2022a. Host. *MDN web docs* [en línea]. [Consulta: 5 febrero 2023]. Disponible en: https://developer.mozilla.org/es/docs/Web/HTTP/Headers/Host.

THE MOZILLA FOUNDATION, 2022b. Lista completa de tipos MIME - HTTP. *MDN web docs* [en línea]. [Consulta: 6 febrero 2023]. Disponible en: https://developer.mozilla.org/es/docs/Web/HTTP/Basics_of_HTTP/MIME_types/Common_types.

THE MOZILLA FOUNDATION, 2022c. X-Content-Type-Options - HTTP. *MDN web docs* [en línea]. [Consulta: 6 febrero 2023]. Disponible en: https://developer.mozilla.org/es/docs/Web/HTTP/Headers/X-Content-Type-Options.

THE MOZILLA FOUNDATION, 2023. SPA (Single-page application). *MDN web docs* [en línea]. [Consulta: 1 febrero 2023]. Disponible en: https://developer.mozilla.org/en-US/docs/Glossary/SPA.

THE OWASP FOUNDATION, 2005. *Una Guía para Construir Aplicaciones y Servicios Web Seguros*. 2.0 Black Hat. S.l.: OWASP.

THE OWASP FOUNDATION, 2017. *Code Review Guide* [en línea]. 2.0. S.l.: The OWASP Foundation. [Consulta: 24 octubre 2022]. Disponible en: https://owasp.org/www-project-code-review-guide/.

THE OWASP FOUNDATION, 2020. *Web Security Testing Guide*. 4.2. S.l.: The OWASP Foundation.

THE ZAP DEV TEAM, 2023a. Absence of Anti-CSRF Tokens. *OWASP ZAP* [en línea]. [Consulta: 29 enero 2023]. Disponible en: https://www.zaproxy.org/docs/alerts/10202/.

THE ZAP DEV TEAM, 2023b. Content Security Policy (CSP) Header Not Set. *OWASP ZAP* [en línea]. [Consulta: 2 febrero 2023]. Disponible en: https://www.zaproxy.org/docs/alerts/10038/.

THE ZAP DEV TEAM, 2023c. Cookie without SameSite Attribute. *OWASP ZAP* [en línea]. [Consulta: 3 febrero 2023]. Disponible en: https://www.zaproxy.org/docs/alerts/10054/.

THE ZAP DEV TEAM, 2023d. Modern Web Application. *OWASP ZAP* [en línea]. [Consulta: 30 enero 2023]. Disponible en: https://www.zaproxy.org/docs/alerts/10109/.

THE ZAP DEV TEAM, 2023e. Private IP Disclosure. *OWASP ZAP* [en línea]. [Consulta: 2 febrero 2023]. Disponible en: https://www.zaproxy.org/docs/alerts/2/.

THE ZAP DEV TEAM, 2023f. Re-examine Cache-control Directives. *OWASP ZAP* [en línea]. [Consulta: 30 enero 2023]. Disponible en: https://www.zaproxy.org/docs/alerts/10015/.

THE ZAP DEV TEAM, 2023g. Server Leaks Information via «X-Powered-By» HTTP Response Header Field(s). *OWASP ZAP* [en línea]. [Consulta: 3 febrero 2023]. Disponible en: https://www.zaproxy.org/docs/alerts/10037/.

THE ZAP DEV TEAM, 2023h. Server Leaks its Webserver Application via «Server» HTTP Response Header Field. *OWASP ZAP* [en línea]. [Consulta: 29 enero 2023]. Disponible en: https://www.zaproxy.org/docs/alerts/10036-1/.

THE ZAP DEV TEAM, 2023i. Strict-Transport-Security Header. *OWASP ZAP* [en línea]. [Consulta: 29 enero 2023]. Disponible en: https://www.zaproxy.org/docs/alerts/10035/.

THE ZAP DEV TEAM, 2023j. The OWASP ZAP Desktop User Guide. *OWASP ZAP* [en línea]. [Consulta: 2 febrero 2023]. Disponible en: https://www.zaproxy.org/docs/desktop/.

THE ZAP DEV TEAM, 2023k. User Agent Fuzzer. *OWASP ZAP* [en línea]. [Consulta: 30 enero 2023]. Disponible en: https://www.zaproxy.org/docs/alerts/10104/.

THE ZAP DEV TEAM, 2023l. X-Content-Type-Options Header Missing. *OWASP ZAP* [en línea]. [Consulta: 30 enero 2023]. Disponible en: https://www.zaproxy.org/docs/alerts/10021/.

TYAGI, S. y KUMAR, K., 2018. Evaluation of Static Web Vulnerability Analysis Tools. *2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC)* [en línea]. Solan Himachal Pradesh, India: IEEE, pp. 1-6. [Consulta: 25 octubre 2022]. ISBN 978-1-72810-646-5. DOI 10.1109/PDGC.2018.8745996. Disponible en: https://ieeexplore.ieee.org/document/8745996/.

VENSON, E., ALFAYEZ, R., GOMES, M.M.F., FIGUEIREDO, R.M.C. y BOEHM, B., 2019. The Impact of Software Security Practices on Development Effort: An Initial Survey. *2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)* [en línea]. Porto de Galinhas, Recife, Brazil: IEEE, pp. 1-12. [Consulta: 24 octubre 2022]. ISBN 978-1-72812-968-6. DOI 10.1109/ESEM.2019.8870153. Disponible en: https://ieeexplore.ieee.org/document/8870153/.

WHITTAKER, J., 2000. What is software testing? And why is it so hard? *IEEE Software*, vol. 17, no. 1, pp. 70-79. ISSN 07407459. DOI 10.1109/52.819971.

ZAP DEVELOPMENT TEAM, 2023. *OWASP ZAP* [en línea]. Java. 2023. S.l.: ZAP Development Team. Disponible en: https://www.zaproxy.org/download/.

ZAPATA GARCÍA, J., 2018. *USO DE TECNOLOGÍAS DE PRUEBAS DE PENETRACIÓN PARA VALIDACIÓN DE SEGURIDAD DE APLICACIONES WEB BASADO EN EL TOP 10 DE VULNERABILIDADES DE OWASP*. S.l.: Universidad Nacional Abierta y a Distancia.

ZECH, P., FELDERER, M. y BREU, R., 2019. Knowledge-based security testing of web applications by logic programming. *International Journal on Software Tools for Technology Transfer*, vol. 21, no. 2, pp. 221-246. ISSN 1433-2779, 1433-2787. DOI 10.1007/s10009-017-0472-3.

ANEXO D: MANUAL TÉCNICO

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA SOFTWARE

ANÁLISIS DE VULNERABILIDADES Y PRUEBAS DE ESTRÉS AL SISTEMA ACADÉMICO DE LA ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

MANUAL TÉCNICO

Tipo: Proyecto Técnico

Presentado para optar el grado académico de:

INGENIERA EN SOFTWARE

AUTORAS:

ERIKA MICHELLE ASTUDILLO MUÑOZ

ANDREA ELIZABETH VIZUETE ULLOA

DIRECTOR: Ing. DIEGO FERNANDO AVILA PESANTEZ Ph.D.

Riobamba – Ecuador

2023

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1. ESTUDIO DE FACTIBILIDAD

El estudio de factibilidad es el análisis de los recursos técnicos, económicos y operacionales del trabajo de integración curricular para determinar si el trabajo es viable o no, cuáles son las condiciones ideales para realizarlo y cómo se podría solucionar las dificultades que se puedan presentar (Quiroa, 2022).

1.1. Factibilidad Técnica

El análisis de factibilidad técnica considera los recursos de hardware y software que son necesarios para efectuar el trabajo de integración curricular. Asimismo, se consideran los recursos humanos implicados en el trabajo (BIBLIOTECA DIGITAL, 2020). En la Tabla 1-1 se visualiza los recursos de hardware existentes para la viabilidad del proyecto de integración curricular.

Tabla 30-1: Recursos de Hardware existentes

CANTIDAD	DESCRIPCIÓN	OBSERVACIÓN
1	Lenovo Legion Y520 Procesador: Intel Core i7-7ma Tarjeta Gráfica: NVIDIA GeForce GTX 1060 RAM: 16 GB Disco: SSD 128 GB Disco: HD 1 TB	Necesaria para realizar el análisis de vulnerabilidades y las pruebas de estrés
1	ASUS ROG Strix SCAR 15 G532 Procesador: AMD Ryzen 9 Tarjeta Gráfica: RTX 3070 RAM: 32 GB Disco: SSD 1 TB	Necesaria para realizar el análisis de vulnerabilidades y las pruebas de estrés

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

En la Tabla 2-1 se visualiza los recursos de software existentes; mientras que, en la Tabla 3-1 los recursos de software requeridos para la viabilidad del trabajo de integración curricular.

Tabla 31-1: Recursos de Software existentes

CANTIDAD	DESCRIPCIÓN	OBSERVACIÓN
2	Windows 10	Sistema operativo de paga

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

Tabla 32-1: Recursos de Software requeridos

CANTIDAD	DESCRIPCIÓN	OBSERVACIÓN

2	OWASP ZAP (Herramienta para realizar el análisis de vulnerabilidades)	Software libre
2	JMeter (Herramienta para realizar las pruebas de estrés)	Software libre

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

En la Tabla 4-1 se observa los recursos humanos presentes para realizar los análisis y pruebas correspondientes. De igual manera, en la Tabla 1-5 se visualiza los recursos humanos requeridos para que revisen y usen las recomendaciones propuestas para mejorar el sistema académico de la ESPOCH.

Tabla 33-1: Recursos Humanos existentes

CANTIDAD	FUNCIÓN	FORMACIÓN	OBSERVACIÓN
2	Analista de Seguridad - Tester	Estudios de tercer nivel	Analista de seguridad y tester de aplicaciones

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

Tabla 34-1: Recursos Humanos requeridos

CANTIDAD	FUNCIÓN	FORMACIÓN	OBSERVACIÓN
2	Desarrolladores	Estudios de tercer nivel	Programadores del área de Desarrollo de DTIC'S perteneciente a la ESPOCH

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

Como se observa, se disponen de los recursos técnicos y los recursos humanos para realizar el trabajo de integración curricular.

1.2. Factibilidad Económica

El análisis de factibilidad económica evalúa económicamente los recursos planteados anteriormente; de igual manera, determina la relación costo-beneficio del trabajo de integración curricular (BIBLIOTECA DIGITAL, 2020).

1.2.1. Costos

Los costos del análisis de vulnerabilidades y de las pruebas de estrés, está dado por la utilización de los recursos de hardware, software, humanos, luz e internet ocupados en el proceso de el desarrollo del trabajo. A continuación, en la Tabla 6-1 se muestra el listado de los recursos ocupados en conjunto con el costo unitario y el costo total del trabajo de integración curricular; mientras que, en la Tabla 7-1 se muestra el número de personas que se necesitan y el costo/mes de cada uno.

Tabla 35-1: Recursos Técnicos Utilizados

DETALLE	CANTIDAD	VALOR UNITARIO	COSTO TOTAL
Lenovo Legion Y520 Procesador: Intel Core i7-7ma Tarjeta Gráfica: NVIDIA GeForce GTX 1060 RAM: 16 GB Disco: SSD 128 GB, HD 1 TB	1	$ 1250	$ 1250
ASUS ROG Strix SCAR 15 G532 Procesador: AMD Ryzen 9 Tarjeta Gráfica: RTX 3070 RAM: 32 GB Disco: SSD 1 TB	1	$ 2300	$ 2300
Mouse	1	$ 10	$ 10
Alimentación/mes	4	$ 60	$ 300
Pago de luz/mes	4	$ 40	$ 200
Internet/mes	4	$ 30	$ 120
COSTO TOTAL			$ 4180

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

Tabla 36-1: Recursos Humanos Utilizados

DETALLE	PERSONAL	MES	COSTO MENSUAL/UNITARIO	COSTO TOTAL
Tester	2	2	$ 1000	$ 4000
Desarrollador	2	1	$ 1200	$ 2400
TOTAL				$ 6400

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

1.2.2. Beneficios

Los beneficios tangibles y no tangibles serán notorios una vez que se hayan realizado las pruebas correspondientes al sistema académico de la ESPOCH. A continuación, se describen los beneficios tangibles:

- Manual de buenas prácticas para mejorar el sistema académico de la ESPOCH.

- Mitigación de vulnerabilidades en caso de que se apliquen las buenas prácticas.

Los beneficios intangibles son:

- Mejora en el rendimiento del sistema académico de la ESPOCH si son aplicadas las buenas prácticas recomendadas.
- Identificación de las causas del cuello de botella.

1.3. Factibilidad Operativa

La factibilidad operativa se relaciona con el personal del área de DTIC's, quienes observarán y estudiarán el manual de mejores prácticas para mejorar el sistema académico de la ESPOCH, dado que, implementarán las mejoras; por lo tanto, es viable porque el manual está enfocado al área de DTIC's.

2. ANÁLISIS DE RIESGO

En esta sección se describen los riesgos que implicaría llevar a cabo el proyecto de trabajo de integración curricular mientras se avanza con el análisis de vulnerabilidad y las pruebas de estrés en el sistema académico de la ESPOCH.

- Uno de los integrantes abandone el proyecto.
- No encontrar herramientas adapta para llevar a cabo el análisis de vulnerabilidad y pruebas de estrés.
- No encontrar la documentación necesaria que explique la utilización de las herramientas.
- Los del área de desarrollo de DTIC's no permitan realizar las pruebas en el sistema académico.
- Los dispositivos por utilizar presenten fallas.
- Al usar las herramientas para el análisis de vulnerabilidad y pruebas de estrés comiencen a dar errores.

3. PLANIFICACIÓN DEL PROYECTO

En esta sección se procederá a describir la fase de planificación del proyecto de trabajo de integración curricular con la finalidad de identificar los integrantes para realizar el análisis de vulnerabilidades y las pruebas de estrés en el sistema académico de la ESPOCH.

3.1. Roles del estudio

En la Tabla 8-3, se observa los integrantes que van a realizar el análisis de vulnerabilidades y las pruebas de estrés con sus respectivos roles.

Tabla 37-3: Roles del estudio

Integrante	Rol	Contacto
Dirección de Tecnología de Información	Desarrolladores del Sistema Académico	dpalacios@espoch.edu.ec
Erika Michelle Astudillo Muñoz	Tester – Analista de seguridad	erika.astudillo@espoch.edu.ec
Andrea Elizabeth Vizuete Ulloa	Tester – Analista de seguridad	andrea.vizuete@espoch.edu.ec

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

4. VULNERABILIDADES DETECTADAS EN EL SISTEMA ACADÉMICO

La herramienta utilizada para el análisis de vulnerabilidades es OWASP Zen Attack Proxy ZAP. En la Figura 1-4, inicializa la ejecución de la herramienta OWASP ZAP versión 2.12.0. En la Figura 2-4 se describe la interfaz de usuario de ZAP Desktop, la cual se compone de los siguientes elementos:

1. **Barra de menú**: proporciona acceso a muchas de las herramientas automáticas y manuales.
2. **Barra de herramientas:** incluye botones que brindan fácil acceso a las funciones más utilizadas.
3. **Ventana de árbol**: muestra el árbol de sitios y el árbol de scripts.
4. **Ventana del área de trabajo**: muestra solicitudes, respuestas y guiones y le permite editarlos.
5. **Ventana de información**: muestra detalles de las herramientas automáticas y manuales.
6. **Pie de página:** muestra un resumen de las alertas encontradas y el estado de las principales herramientas automatizadas.

Figura 25-4: Inicio de herramienta OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

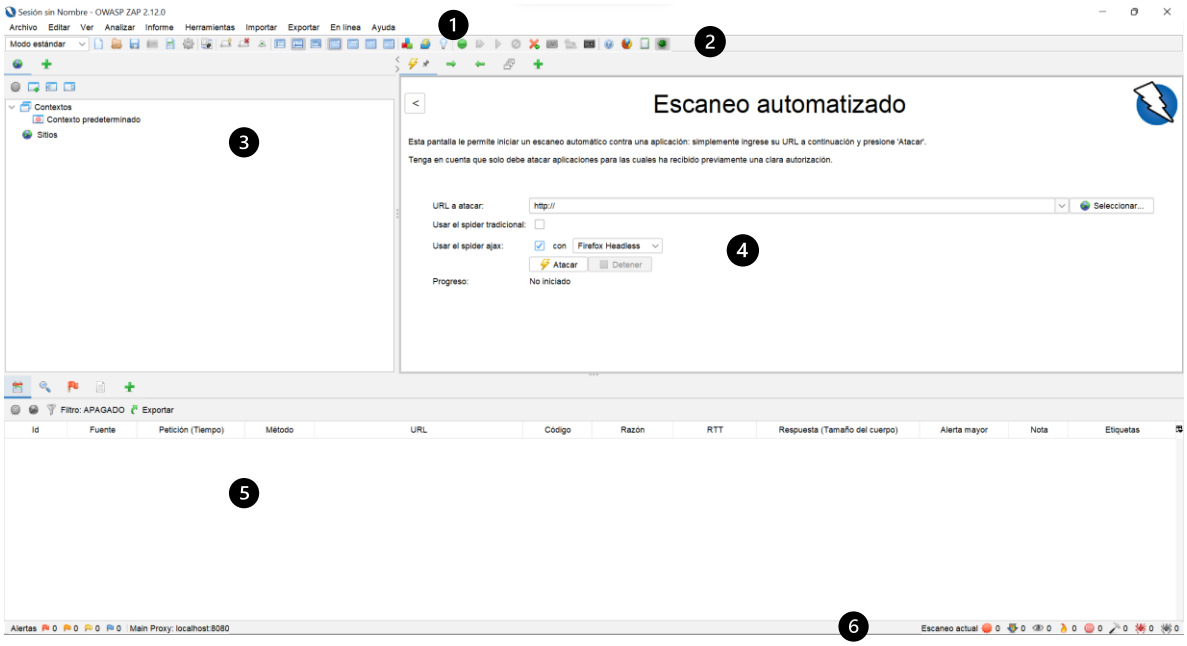


Figura 26-4: Herramienta OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

En la Figuras 3-4, 4-4 y 5-4 se muestra el utilizo de la herramienta OWASP ZAP, y la realización del escaneo de vulnerabilidades al sistema académico de la ESPOCH.

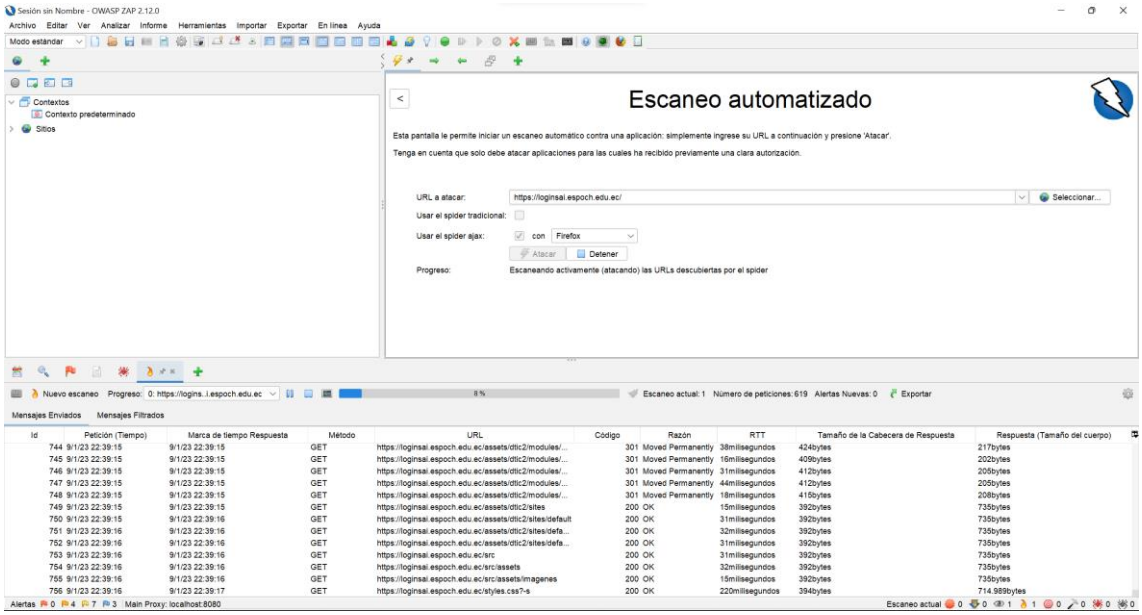


Figura 27-4: Herramienta OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

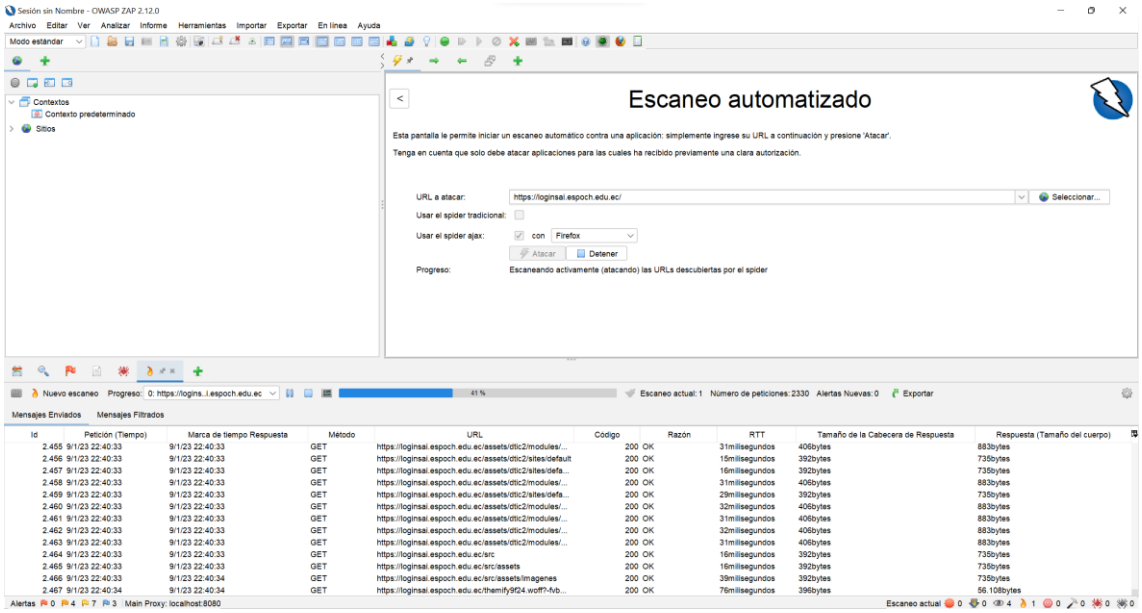


Figura 28-4: Escaneo de Vulnerabilidades - OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

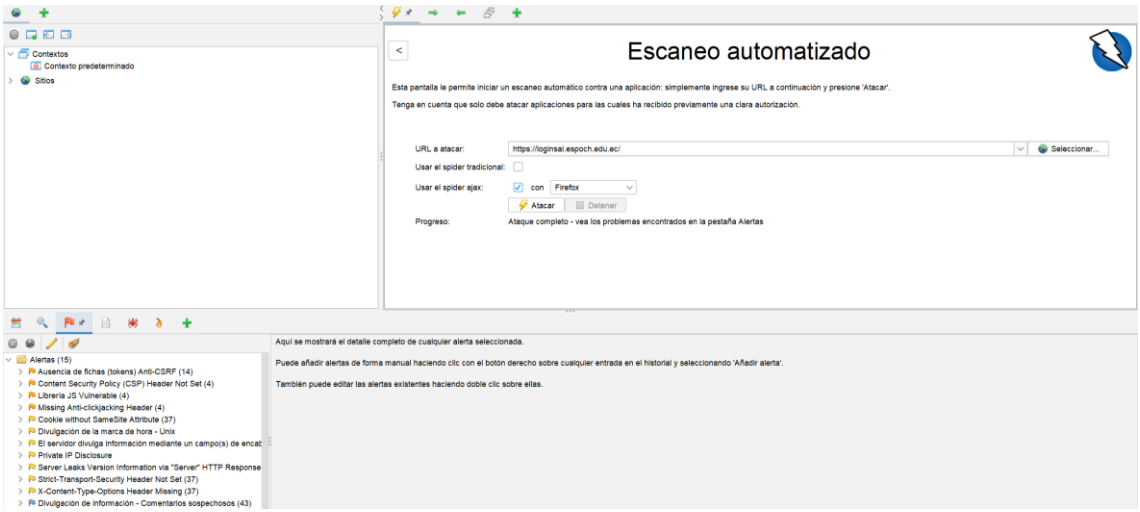


Figura 29-4: Análisis de Vulnerabilidades – Herramienta OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

Una vez realizado el análisis, la herramienta genera un reporte como se observa en las Figuras 6-4, 7-4, 8-4 y 9-4 a continuación:

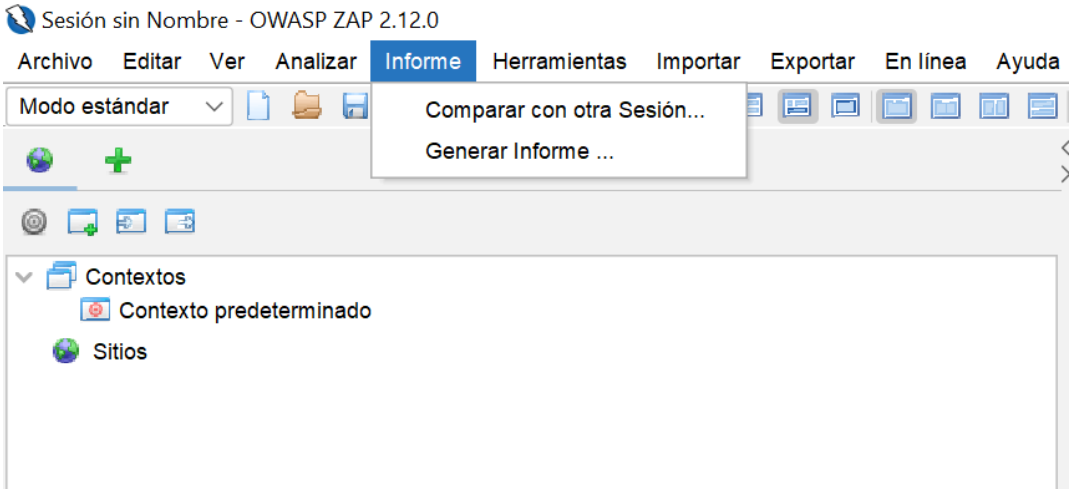


Figura 30-4: Informe – Herramienta OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

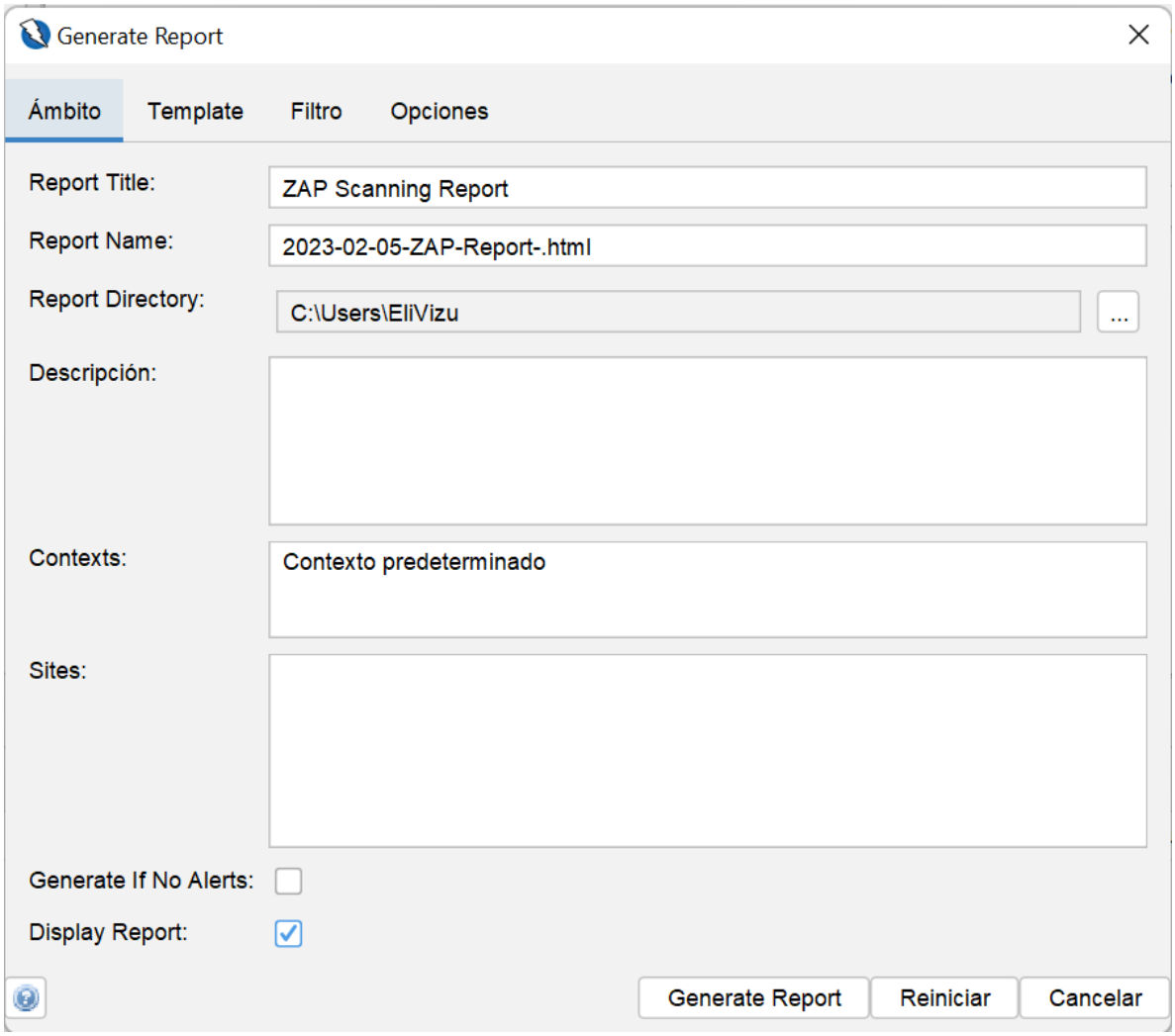


Figura 31-4: Generar Informe – Herramienta OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

Informe de escaneo ZAP

Generado con ZAP el jue. 5 ene. 2023, a las 14:33:43

Contenido

Figura 32-4: Información de informe – Herramienta OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

- Alertas
  - Riesgo = Medio , Confianza = Alto (1)
  - Riesgo = Medio , Confianza = Medio (2)
  - Riesgo = Medio , Confianza = Bajo (1)
  - Riesgo = Bajo , Confianza = Alto (2)
  - Riesgo = Bajo , Confianza = Medio (4)
  - Riesgo = Bajo , Confianza = Bajo (1)
  - Riesgo = Informativo , Confianza = Medio (2)
  - Riesgo = Informativo , Confianza = Bajo (2)
- Apéndice

- Tipos de alerta

Acerca de este informe

Figura 33-4: Información de informe – Herramienta OWASP Zen Attack Proxy ZAP

Fuente: ZAP Development Team, 2023

En la Tabla 9-4, se muestra las vulnerabilidades detectadas en el sistema académico al realizar el escaneo con la herramienta OWASP ZAP.

Tabla 38-4: Vulnerabilidades detectadas en el sistema académico

TIPO DE ALERTA	NIVEL DE RIESGO	PORCENTAJE	No. DE ALERTAS
Ausencia de fichas Anti/CSRF	Medio	93,3 %	14
Encabezado de políticas de seguridad de contenido (CSP) no establecido	Medio	26,7 %	4
Falta el encabezado antisecuestro de clics	Medio	26,7 %	4
Biblioteca JS vulnerable	Medio	26,7 %	4
Cookie sin atributo SameSite	Bajo	246,7 %	37
Divulgación de la marca de hora – Unix	Bajo	6,7 %	1
El servidor divulga información mediante un campo de encabezado de respuesta HTTP “”X-Powered-By””	Bajo	224,7	37
Divulgación de IP privada	Bajo	6,7 %	1
Encabezado de respuesta del servidor HTTP	Bajo	246,7 %	37
Encabezado de seguridad de transporte estricto	Bajo	246,7 %	37
Falta el encabezado X-Content-Type-Options	Bajo	246,7 %	37
Divulgación de información - Comentarios sospechosos	Informativo	286,7 %	43
Aplicación web moderna	Informativo	53,3 %	8
Reexaminar las directivas de control de caché	Informativo	13,3 %	2
Fuzzer de agente de usuario	Informativo	2.320,0 %	348
Total			**15**

Fuente: ZAP Development Team, 2023

Realizado por: Astudillo Muñoz, Erika y Vizuete Ulloa, Andrea, 2023

5. EJECUCIÓN DE LAS PRUEBAS DE ESTRÉS

Se ha hecho tres pruebas estrés por cada nivel; es decir, nueve pruebas en total. Los niveles dependen de la cantidad de hilos o procesos que simulan los usuarios conectados en el proceso de matrículas al sistema académico de la ESPOCH; a su vez, cada hilo contiene 19 peticiones.

5.1. Ejecución de las pruebas de estrés en función al nivel básico

A continuación, se muestran las ejecuciones de las pruebas de estrés en función al nivel básico de peticiones por minuto.

5.1.1. Primera ejecución de la prueba de estrés en función al nivel básico

En la Figura 10-5 se observa la configuración del escenario de la prueba de estrés, la cantidad de hilos son 34 por cada minuto, las veces que se repetirá este proceso es 180, simulando las 3 horas que hay más afluencia de los estudiantes.

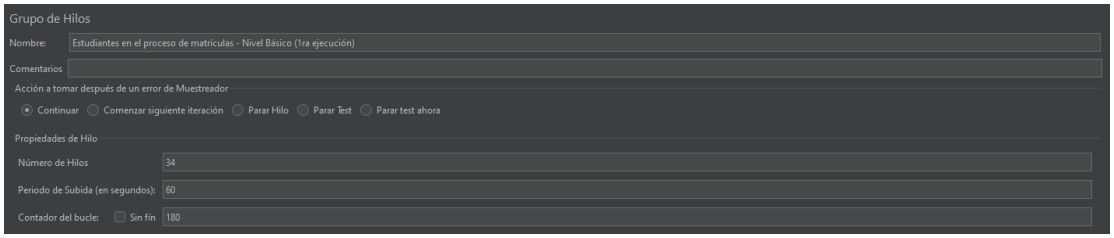


Figura 34-5: Ingreso de los parámetros considerados para el nivel básico.

Fuente: Apache Software Foundation, 2022

En la Figura 11-5 se muestra la ejecución total de la prueba de estrés, en la que se observa que el tiempo total de la ejecución es de 2 horas, 42 minutos y 13 segundos, tiempo en el que se simularon 6120 procesos y 116.280 peticiones hechas al sistema académico; asimismo, se observa que el error promediado del total de las peticiones es de 0.04%. Finalmente, el rendimiento del sistema académico en función a todas las peticiones hechas es de 720 peticiones por minuto.

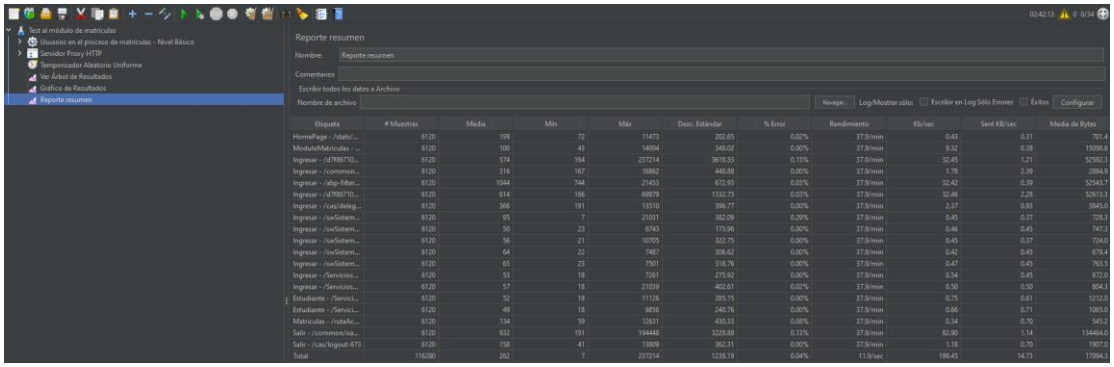


Figura 35-5: Ejecución de la prueba en función del nivel básico.

Fuente: Apache Software Foundation, 2022

En la Figura 12-5 se muestra las peticiones realizadas al sistema académico de la ESPOCH. Al dar clic en una se muestra el número de hilo que se está ejecutando en ese momento, el tiempo que se demora en ejecutarse, entre otros datos relevantes de esta petición.

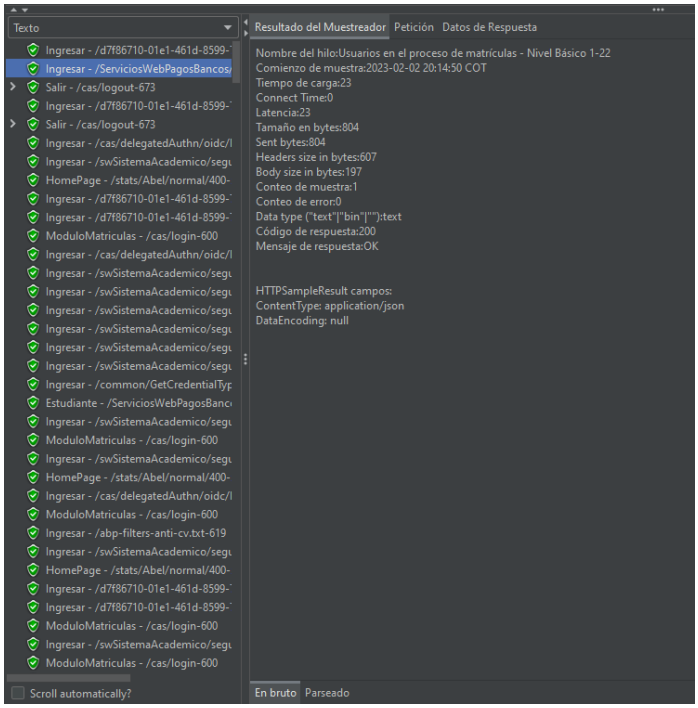


Figura 36-5: Ejecución de la prueba en función del nivel básico.

Fuente: Apache Software Foundation, 2022

5.1.2. Segunda ejecución de la prueba de estrés en función al nivel básico

En la Figura 13-5 se observa la configuración del escenario de la prueba de estrés, la cantidad de hilos son 34 por cada minuto, las veces que se repetirá este proceso es 180, simulando las 3 horas que hay más afluencia de los estudiantes.

Figura 37-5: Ingreso de los parámetros considerados para el nivel básico.

Fuente: Apache Software Foundation, 2022

En la Figura 14-5 se muestra la ejecución total de la prueba de estrés, en la que se observa que el tiempo total de la ejecución es de 2 horas, 40 minutos y 49 segundos, tiempo en el que se simularon 6120 procesos y 116.280 peticiones hechas al sistema académico; asimismo, se observa que el error promediado del total de las peticiones es de 0.02%. Finalmente, el rendimiento del sistema académico en función a todas las peticiones hechas es de 726 peticiones por minuto.

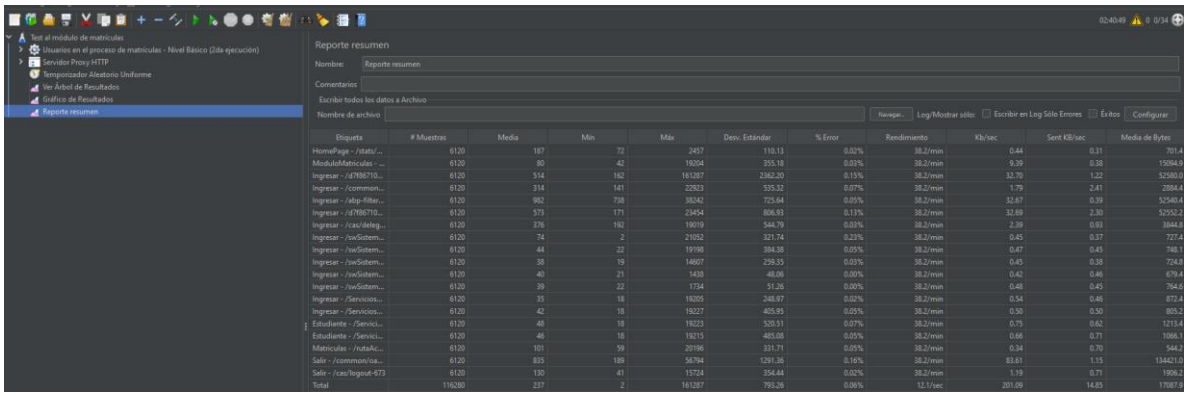


Figura 38-5: Ejecución de la prueba en función del nivel básico.

Fuente: Apache Software Foundation, 2022

En la Figura 15-5 se muestra las peticiones realizadas al sistema académico de la ESPOCH. Al dar clic en una se muestra el número de hilo que se está ejecutando en ese momento, el tiempo que se demora en ejecutarse, entre otros datos relevantes de esta petición.

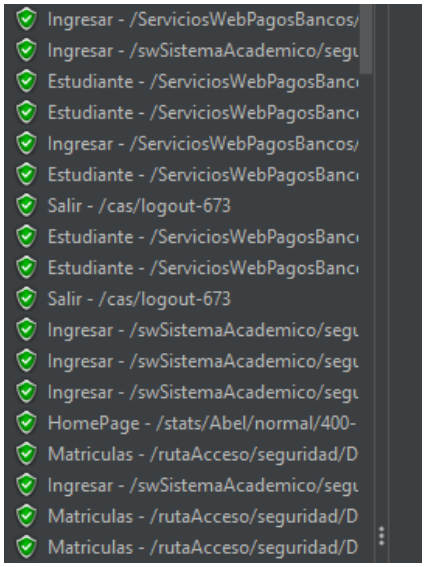


Figura 39-5: Ejecución de la prueba en función del nivel básico.

Fuente: Apache Software Foundation, 2022

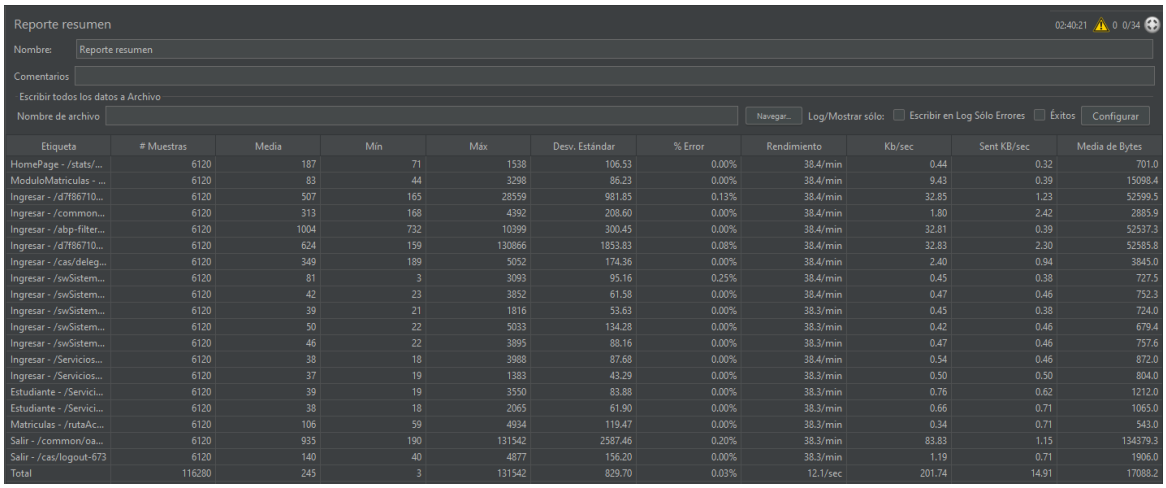
5.1.3. Tercera ejecución de la prueba de estrés en función al nivel básico

En la Figura 16-5 se observa la configuración del escenario de la prueba de estrés, la cantidad de hilos son 34 por cada minuto, las veces que se repetirá este proceso es 180, simulando las 3 horas que hay más afluencia de los estudiantes.

Figura 40-5: Ingreso de los parámetros considerados para el nivel básico.

Fuente: Apache Software Foundation, 2022

En la Figura 17-5 se muestra la ejecución total de la prueba de estrés, en la que se observa que el tiempo total de la ejecución es de 2 horas, 40 minutos y 21 segundos, tiempo en el que se simularon 6120 procesos y 116.280 peticiones hechas al sistema académico; asimismo, se observa que el error del total de las peticiones es de 0.03%. Finalmente, el rendimiento del sistema académico en función a todas las peticiones hechas es de 729 peticiones por minuto.



Etiqueta	# Muestras	Media	Min	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
HomePage - /stats/...	6120	187	71	1538	106.53	0.00%	38.4/min	0.44	0.32	701.0
ModuloMatriculas - ...	6120	83	44	3298	86.23	0.00%	38.4/min	9.43	0.39	15098.4
Ingresar - /d7f86710...	6120	507	165	28559	981.85	0.13%	38.4/min	32.85	1.23	52599.5
Ingresar - /common...	6120	313	168	4392	208.60	0.00%	38.4/min	1.80	2.42	2885.9
Ingresar - /abp-filter...	6120	1004	732	10399	300.45	0.00%	38.4/min	32.81	0.39	52537.3
Ingresar - /d7f86710...	6120	624	159	130866	1853.83	0.08%	38.4/min	32.83	2.30	52585.8
Ingresar - /cas/deleg...	6120	349	189	5052	174.36	0.00%	38.4/min	2.40	0.94	3845.0
Ingresar - /swSistem...	6120	81	3	3093	95.16	0.25%	38.4/min	0.45	0.38	727.5
Ingresar - /swSistem...	6120	42	23	3852	61.58	0.00%	38.4/min	0.47	0.46	752.3
Ingresar - /swSistem...	6120	39	21	1816	53.63	0.00%	38.3/min	0.45	0.38	724.0
Ingresar - /swSistem...	6120	50	22	5033	134.28	0.00%	38.3/min	0.42	0.46	679.4
Ingresar - /swSistem...	6120	46	22	3895	88.16	0.00%	38.3/min	0.47	0.46	757.6
Ingresar - /Servicios...	6120	38	18	3988	87.68	0.00%	38.4/min	0.54	0.46	872.0
Ingresar - /Servicios...	6120	37	19	1383	43.29	0.00%	38.3/min	0.50	0.50	804.0
Estudiante - /Servici...	6120	39	19	3550	83.88	0.00%	38.3/min	0.76	0.62	1212.0
Estudiante - /Servici...	6120	38	18	2065	61.90	0.00%	38.3/min	0.66	0.71	1065.0
Matriculas - /rutaAc...	6120	106	59	4934	119.47	0.00%	38.3/min	0.34	0.71	543.0
Salir - /common/oa...	6120	935	190	131542	2587.46	0.20%	38.3/min	83.83	1.15	134379.3
Salir - /cas/logout-673	6120	140	40	4877	156.20	0.00%	38.3/min	1.19	0.71	1906.0
Total	116280	245	3	131542	829.70	0.03%	12.1/sec	201.74	14.91	17088.2

Figura 41-5: Ejecución de la prueba en función del nivel básico.

Fuente: Apache Software Foundation, 2022

5.1.4. Primera ejecución de la prueba de estrés en función al nivel medio

En la Figura 18-5 se observa la configuración del escenario de la prueba de estrés, la cantidad de hilos son 40 por cada minuto, las veces que se repetirá este proceso es 180, simulando las 3 horas que hay más afluencia de los estudiantes.

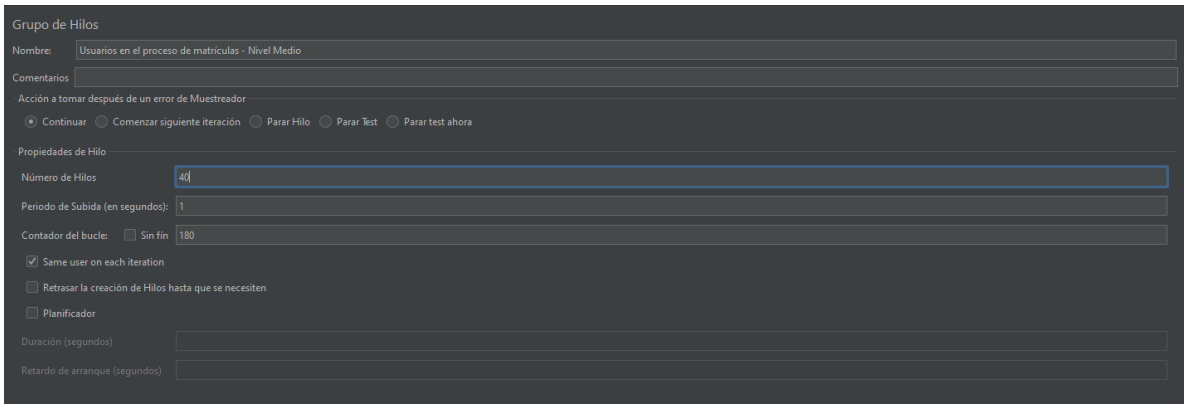


Figura 42-5: Ingreso de los parámetros considerados para el nivel medio.

Fuente: Apache Software Foundation, 2022

En la Figura 19-5 se muestra la ejecución total de la prueba de estrés, en la que se observa que el tiempo total de la ejecución es de 2 horas, 37 minutos y 33 segundos, tiempo en el que se simularon 7200 procesos y 136.800 peticiones hechas al sistema académico; asimismo, se observa que el error promediado del total de las peticiones es de 0.02%. Finalmente, el rendimiento del sistema académico en función a todas las peticiones hechas es de 872 peticiones por minuto.

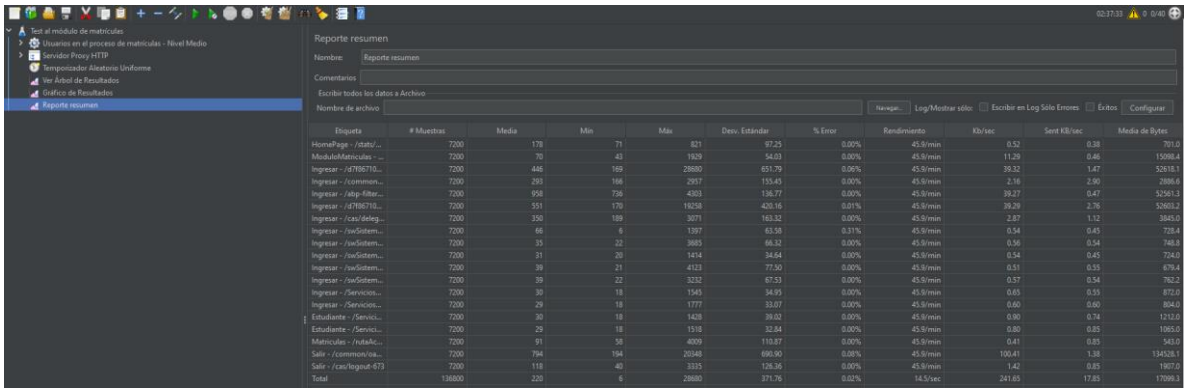


Figura 43-5: Ejecución de la prueba en función del nivel medio.

Fuente: Apache Software Foundation, 2022

5.1.5. Segunda ejecución de la prueba de estrés en función al nivel medio

En la Figura 20-5 se observa la configuración del escenario de la prueba de estrés, la cantidad de hilos son 40 por cada minuto, las veces que se repetirá este proceso es 180, simulando las 3 horas que hay más afluencia de los estudiantes.

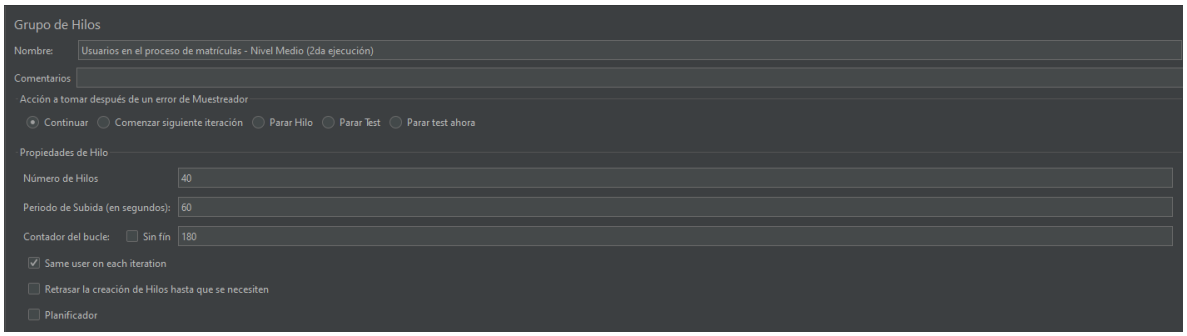
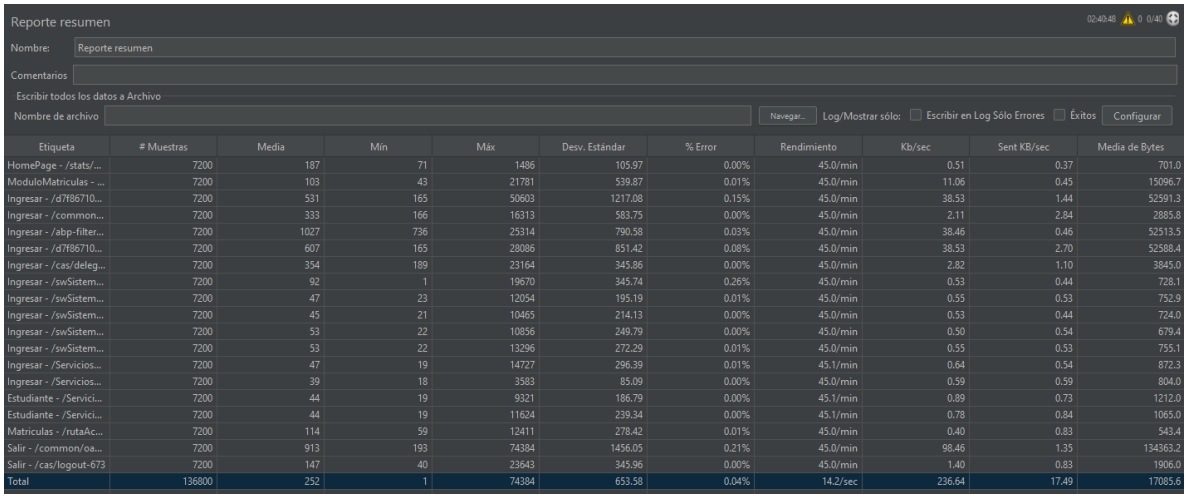


Figura 44-5: Ingreso de los parámetros considerados para el nivel medio.

Fuente: Apache Software Foundation, 2022

En la Figura 21-5 se muestra la ejecución total de la prueba de estrés, en la que se observa que el tiempo total de la ejecución es de 2 horas, 40 minutos y 48 segundos, tiempo en el que se simularon 7200 procesos y 136.800 peticiones hechas al sistema académico; asimismo, se observa que el error promediado del total de las peticiones es de 0.04%. Finalmente, el rendimiento del sistema académico en función a todas las peticiones hechas es de 835 peticiones por minuto.



Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
HomePage - /stats/...	7200	187	71	1486	105.97	0.00%	45.0/min	0.51	0.37	701.0
ModuloMatriculas - ...	7200	103	43	21781	539.87	0.01%	45.0/min	11.06	0.45	15096.7
Ingresar - /d7f86710...	7200	531	165	50603	1217.08	0.15%	45.0/min	38.53	1.44	52591.3
Ingresar - /common...	7200	333	166	16313	583.75	0.00%	45.0/min	2.11	2.84	2885.8
Ingresar - /abp-filter...	7200	1027	736	25314	790.58	0.03%	45.0/min	38.46	0.46	52513.5
Ingresar - /d7f86710...	7200	607	165	28086	851.42	0.08%	45.0/min	38.53	2.70	52588.4
Ingresar - /cas/deleg...	7200	354	189	23164	345.86	0.00%	45.0/min	2.82	1.10	3845.0
Ingresar - /swSistem...	7200	92	1	19670	345.74	0.26%	45.0/min	0.53	0.44	728.1
Ingresar - /swSistem...	7200	47	23	12054	195.19	0.01%	45.0/min	0.55	0.53	752.9
Ingresar - /swSistem...	7200	45	21	10465	214.13	0.00%	45.0/min	0.53	0.44	724.0
Ingresar - /swSistem...	7200	53	22	10856	249.79	0.00%	45.0/min	0.50	0.54	679.4
Ingresar - /swSistem...	7200	53	22	13296	272.29	0.01%	45.0/min	0.55	0.53	755.1
Ingresar - /Servicios...	7200	47	19	14727	296.39	0.01%	45.1/min	0.64	0.54	872.3
Ingresar - /Servicios...	7200	39	18	3583	85.09	0.00%	45.0/min	0.59	0.59	804.0
Estudiante - /Servici...	7200	44	19	9321	186.79	0.00%	45.1/min	0.89	0.73	1212.0
Estudiante - /Servici...	7200	44	19	11624	239.34	0.00%	45.1/min	0.78	0.84	1065.0
Matriculas - /rutaAc...	7200	114	59	12411	278.42	0.01%	45.0/min	0.40	0.83	543.4
Salir - /common/oa...	7200	913	193	74384	1456.05	0.21%	45.0/min	98.46	1.35	134363.2
Salir - /cas/logout-673	7200	147	40	23643	345.96	0.00%	45.0/min	1.40	0.83	1906.0
Total	136800	252	1	74384	653.58	0.04%	14.2/sec	236.64	17.49	17085.6

Figura 45-5: Ejecución de la prueba en función del nivel medio.

Fuente: Apache Software Foundation, 2022

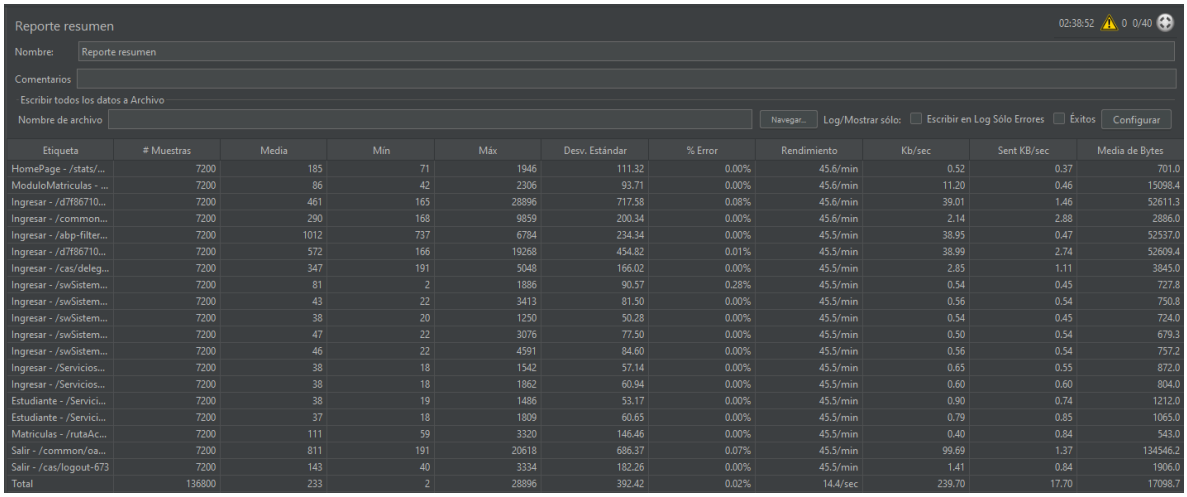
5.1.6. Tercera ejecución de la prueba de estrés en función al nivel medio

En la Figura 22-5 se observa la configuración del escenario de la prueba de estrés, la cantidad de hilos son 40 por cada minuto, las veces que se repetirá este proceso es 180, simulando las 3 horas que hay más afluencia de los estudiantes.

Figura 46-5: Ingreso de los parámetros considerados para el nivel medio.

Fuente: Apache Software Foundation, 2022

En la Figura 23-5 se muestra la ejecución total de la prueba de estrés, en la que se observa que el tiempo total de la ejecución es de 2 horas, 38 minutos y 52 segundos, tiempo en el que se simularon 7200 procesos y 136.800 peticiones hechas al sistema académico; asimismo, se observa que el error promediado del total de las peticiones es de 0.02%. Finalmente, el rendimiento del sistema académico en función a todas las peticiones hechas es de 865 peticiones por minuto.



Etiqueta	# Muestras	Media	Min	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
HomePage - /stats/...	7200	185	71	1946	111.32	0.00%	45.6/min	0.52	0.37	701.0
ModuloMatriculas - ...	7200	86	42	2306	93.71	0.00%	45.6/min	11.20	0.46	15098.4
Ingresar - /d7f86710...	7200	461	165	28896	717.58	0.08%	45.6/min	39.01	1.46	52611.3
Ingresar - /common...	7200	290	168	9859	200.34	0.00%	45.6/min	2.14	2.88	2886.0
Ingresar - /abp-filter...	7200	1012	737	6784	234.34	0.00%	45.5/min	38.95	0.47	52537.0
Ingresar - /d7f86710...	7200	572	166	19268	454.82	0.01%	45.5/min	38.99	2.74	52609.4
Ingresar - /cas/deleg...	7200	347	191	5048	166.02	0.00%	45.5/min	2.85	1.11	3845.0
Ingresar - /swSistem...	7200	81	2	1886	90.57	0.28%	45.5/min	0.54	0.45	727.8
Ingresar - /swSistem...	7200	43	22	3413	81.50	0.00%	45.5/min	0.56	0.54	750.8
Ingresar - /swSistem...	7200	38	20	1250	50.28	0.00%	45.5/min	0.54	0.45	724.0
Ingresar - /swSistem...	7200	47	22	3076	77.50	0.00%	45.5/min	0.50	0.54	679.3
Ingresar - /swSistem...	7200	46	22	4591	84.60	0.00%	45.5/min	0.56	0.54	757.2
Ingresar - /Servicios...	7200	38	18	1542	57.14	0.00%	45.5/min	0.65	0.55	872.0
Ingresar - /Servicios...	7200	38	18	1862	60.94	0.00%	45.5/min	0.60	0.60	804.0
Estudiante - /Servici...	7200	38	19	1486	53.17	0.00%	45.5/min	0.90	0.74	1212.0
Estudiante - /Servici...	7200	37	18	1809	60.65	0.00%	45.5/min	0.79	0.85	1065.0
Matriculas - /rutaAc...	7200	111	59	3320	146.46	0.00%	45.5/min	0.40	0.84	543.0
Salir - /common/oa...	7200	811	191	20618	686.37	0.07%	45.5/min	99.69	1.37	134546.2
Salir - /cas/logout-673	7200	143	40	3334	182.26	0.00%	45.5/min	1.41	0.84	1906.0
Total	136800	233	2	28896	392.42	0.02%	14.4/sec	239.70	17.70	17098.7

Figura 47-5: Ejecución de la prueba en función del nivel medio.

Fuente: Apache Software Foundation, 2022

5.1.7. Primera ejecución de la prueba de estrés en función al nivel alto

En la Figura 24-5 se observa la configuración del escenario de la prueba de estrés, la cantidad de hilos son 46 por cada minuto, las veces que se repetirá este proceso es 180, simulando las 3 horas que hay más afluencia de los estudiantes.

Figura 48-5: Ingreso de los parámetros considerados para el nivel alto.

Fuente: Apache Software Foundation, 2022

En la Figura 25-5 se muestra la ejecución total de la prueba de estrés, en la que se observa que el tiempo total de la ejecución es de 2 horas, 38 minutos y 50 segundos, tiempo en el que se simularon 8280 procesos y 157.320 peticiones hechas al sistema académico; asimismo, se observa que el error promediado del total de las peticiones es de 0.10%. Finalmente, el rendimiento del sistema académico en función a todas las peticiones hechas es de 995 peticiones por minuto.

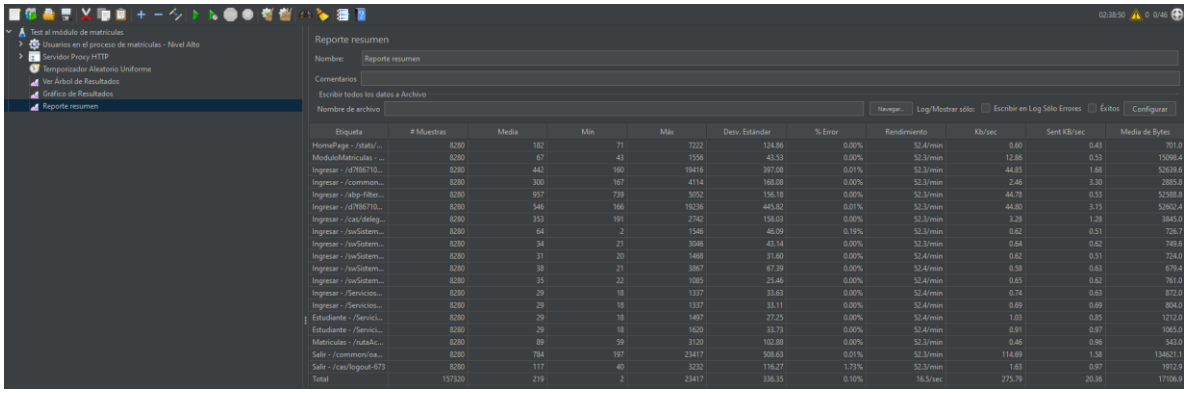


Figura 49-5: Ejecución de la prueba en función del nivel alto.

Fuente: Apache Software Foundation, 2022

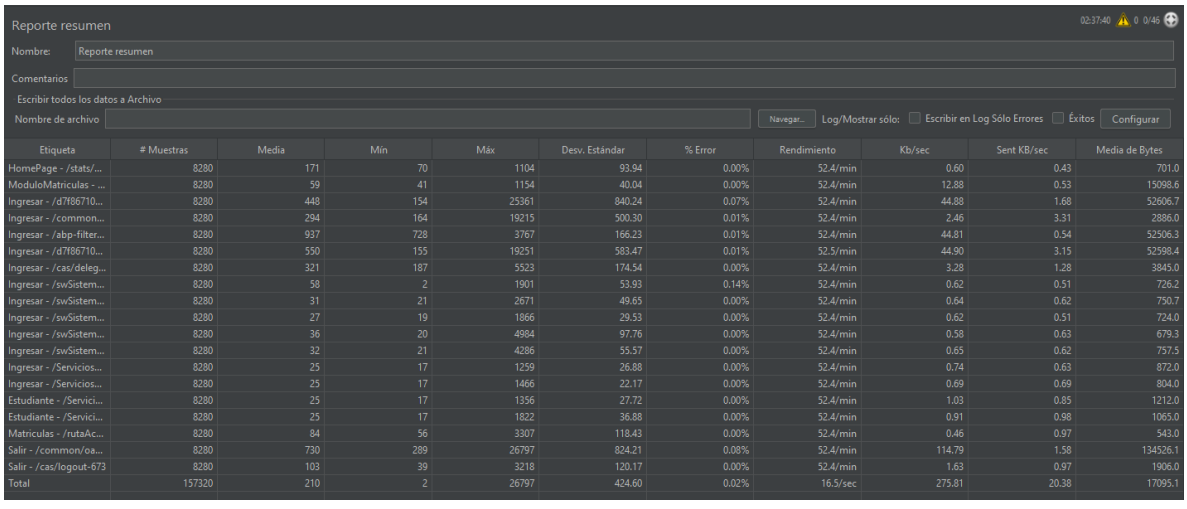
5.1.8. Segunda ejecución de la prueba de estrés en función al nivel alto

En la Figura 26-5 se observa la configuración del escenario de la prueba de estrés, la cantidad de hilos son 46 por cada minuto, las veces que se repetirá este proceso es 180, simulando las 3 horas que hay más afluencia de los estudiantes.

Figura 50-5: Ingreso de los parámetros considerados para el nivel alto.

Fuente: Apache Software Foundation, 2022

En la Figura 27-5 se muestra la ejecución total de la prueba de estrés, en la que se observa que el tiempo total de la ejecución es de 2 horas, 37 minutos y 40 segundos, tiempo en el que se simularon 8280 procesos y 157.320 peticiones hechas al sistema académico; asimismo, se observa que el error promediado del total de las peticiones es de 0.02%. Finalmente, el rendimiento del sistema académico en función a todas las peticiones hechas es de 996 peticiones por minuto.



Etiqueta	# Muestras	Media	Min	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
HomePage - /stats/...	8280	171	70	1104	93.94	0.00%	52.4/min	0.60	0.43	701.0
ModuloMatriculas - ...	8280	59	41	1154	40.04	0.00%	52.4/min	12.88	0.53	15098.6
Ingresar - /d7f86710...	8280	448	154	25361	840.24	0.07%	52.4/min	44.88	1.68	52606.7
Ingresar - /common...	8280	294	164	19215	500.30	0.01%	52.4/min	2.46	3.31	2886.0
Ingresar - /abp-filter...	8280	937	728	3767	166.23	0.01%	52.4/min	44.81	0.54	52506.3
Ingresar - /d7f86710...	8280	550	155	19251	583.47	0.01%	52.5/min	44.90	3.15	52598.4
Ingresar - /cas/deleg...	8280	321	187	5523	174.54	0.00%	52.4/min	3.28	1.28	3845.0
Ingresar - /swSistem...	8280	58	2	1901	53.93	0.14%	52.4/min	0.62	0.51	726.2
Ingresar - /swSistem...	8280	31	21	2671	49.65	0.00%	52.4/min	0.64	0.62	750.7
Ingresar - /swSistem...	8280	27	19	1866	29.53	0.00%	52.4/min	0.62	0.51	724.0
Ingresar - /swSistem...	8280	36	20	4984	97.76	0.00%	52.4/min	0.58	0.63	679.3
Ingresar - /swSistem...	8280	32	21	4286	55.57	0.00%	52.4/min	0.65	0.62	757.5
Ingresar - /Servicios...	8280	25	17	1259	26.88	0.00%	52.4/min	0.74	0.63	872.0
Ingresar - /Servicios...	8280	25	17	1466	22.17	0.00%	52.4/min	0.69	0.69	804.0
Estudiante - /Servici...	8280	25	17	1356	27.72	0.00%	52.4/min	1.03	0.85	1212.0
Estudiante - /Servici...	8280	25	17	1822	36.88	0.00%	52.4/min	0.91	0.98	1065.0
Matriculas - /rutaAc...	8280	84	56	3307	118.43	0.00%	52.4/min	0.46	0.97	543.0
Salir - /common/oa...	8280	730	289	26797	824.21	0.08%	52.4/min	114.79	1.58	134526.1
Salir - /cas/logout-673	8280	103	39	3218	120.17	0.00%	52.4/min	1.63	0.97	1906.0
Total	157320	210	2	26797	424.60	0.02%	16.5/sec	275.81	20.38	17095.1

Figura 51-5: Ejecución de la prueba en función del nivel alto.

Fuente: Apache Software Foundation, 2022

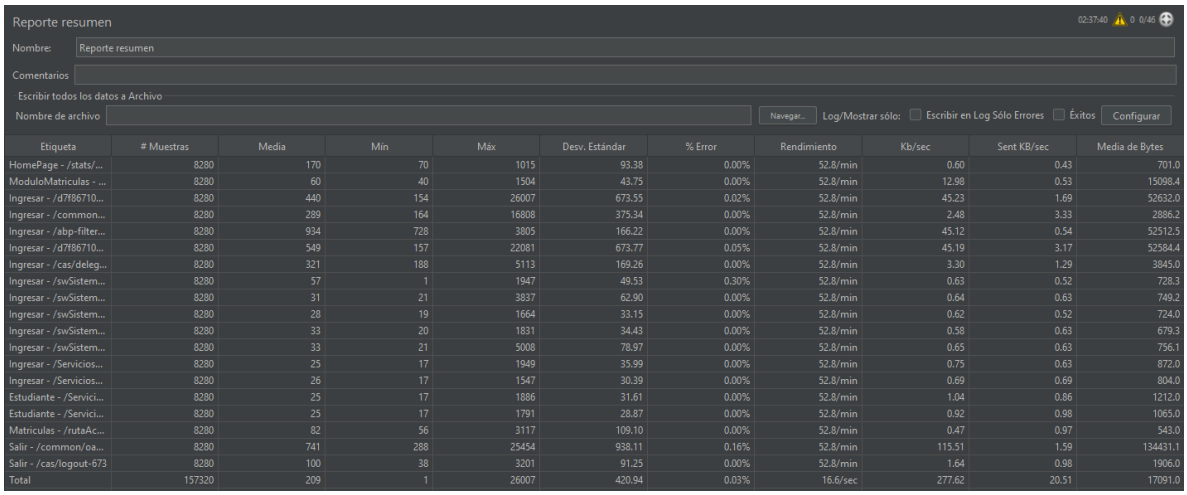
5.1.9. Tercera ejecución de la prueba de estrés en función al nivel alto

En la Figura 28-5 se observa la configuración del escenario de la prueba de estrés, la cantidad de hilos son 46 por cada minuto, las veces que se repetirá este proceso es 180, simulando las 3 horas que hay más afluencia de los estudiantes.

Figura 52-5: Ingreso de los parámetros considerados para el nivel alto.

Fuente: Apache Software Foundation, 2022

En la Figura 29-5 se muestra la ejecución total de la prueba de estrés, en la que se observa que el tiempo total de la ejecución es de 2 horas, 37 minutos y 40 segundos, tiempo en el que se simularon 8280 procesos y 157.320 peticiones hechas al sistema académico; asimismo, se observa que el error promediado del total de las peticiones es de 0.03%. Finalmente, el rendimiento del sistema académico en función a todas las peticiones hechas es de 996 peticiones por minuto.



Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
HomePage - /stats/...	8280	170	70	1015	93.38	0.00%	52.8/min	0.60	0.43	701.0
ModuloMatriculas - ...	8280	60	40	1504	43.75	0.00%	52.8/min	12.98	0.53	15098.4
Ingresar - /d7f86710...	8280	440	154	26007	673.55	0.02%	52.8/min	45.23	1.69	52632.0
Ingresar - /common...	8280	289	164	16808	375.34	0.00%	52.8/min	2.48	3.33	2886.2
Ingresar - /abp-filter...	8280	934	728	3805	166.22	0.00%	52.8/min	45.12	0.54	52512.5
Ingresar - /d7f86710...	8280	549	157	22081	673.77	0.05%	52.8/min	45.19	3.17	52584.4
Ingresar - /cas/deleg...	8280	321	188	5113	169.26	0.00%	52.8/min	3.30	1.29	3845.0
Ingresar - /swSistem...	8280	57	1	1947	49.53	0.30%	52.8/min	0.63	0.52	728.3
Ingresar - /swSistem...	8280	31	21	3837	62.90	0.00%	52.8/min	0.64	0.63	749.2
Ingresar - /swSistem...	8280	28	19	1664	33.15	0.00%	52.8/min	0.62	0.52	724.0
Ingresar - /swSistem...	8280	33	20	1831	34.43	0.00%	52.8/min	0.58	0.63	679.3
Ingresar - /swSistem...	8280	33	21	5008	78.97	0.00%	52.8/min	0.65	0.63	756.1
Ingresar - /Servicios...	8280	25	17	1949	35.99	0.00%	52.8/min	0.75	0.63	872.0
Ingresar - /Servicios...	8280	26	17	1547	30.39	0.00%	52.8/min	0.69	0.69	804.0
Estudiante - /Servici...	8280	25	17	1886	31.61	0.00%	52.8/min	1.04	0.86	1212.0
Estudiante - /Servici...	8280	25	17	1791	28.87	0.00%	52.8/min	0.92	0.98	1065.0
Matriculas - /rutaAc...	8280	82	56	3117	109.10	0.00%	52.8/min	0.47	0.97	543.0
Salir - /common/oa...	8280	741	288	25454	938.11	0.16%	52.8/min	115.51	1.59	134431.1
Salir - /cas/logout-673	8280	100	38	3201	91.25	0.00%	52.8/min	1.64	0.98	1906.0
Total	157320	209	1	26007	420.94	0.03%	16.6/sec	277.62	20.51	17091.0

Figura 53-5: Ejecución de la prueba en función del nivel medio.

Fuente: Apache Software Foundation, 2022

BIBLIOGRAFÍA

APACHE SOFTWARE FOUNDATION, 2022. *Apache JMeter* [en línea]. Java. 2022. S.l.: Apache Software Foundation. Disponible en: https://jmeter.apache.org/download_jmeter.cgi.

BIBLIOTECA DIGITAL, 2020. *METODOLOGÍA*. 2020. S.l.: BIBLIOTECA DIGITAL. PDF

QUIROA, M., 2022. Estudio de factibilidad - Qué es, definición y concepto | 2022 | Economipedia. *economipedia* [en línea]. [Consulta: 26 octubre 2022]. Disponible en: https://economipedia.com/definiciones/estudio-de-factibilidad.html.

ZAP DEVELOPMENT TEAM, 2023. *OWASP ZAP* [en línea]. Java. 2023. S.l.: ZAP Development Team. Disponible en: https://www.zaproxy.org/download/.

ANEXO E: INFORME DEL ESCANEO DE VULNERABILIDADES

Informe de escaneo ZAP

Generado con ZAP el jue. 5 ene. 2023, a las 14:33:43

Contenido

- Tipos de alerta

Acerca de este informe

Informe de parámetros

Contextos

No se seleccionó ningún contexto, por lo que todos los contextos se incluyeron de forma predeterminada.

Sitios

Se incluyeron los siguientes sitios:

- `https://loginsai.espoch.edu.ec`

(Si no se seleccionó ningún sitio, todos los sitios se incluyeron de manera predeterminada).

Un sitio incluido también debe estar dentro de uno de los contextos incluidos para que sus datos se incluyan en el informe.

Niveles de riesgo

Incluido : `Alto` , `Medio` , `Bajo` , `Informativo`

Excluido : Ninguno

Niveles de confianza

Incluido : `Usuario confirmado` , `Alto` , `Medio` , `Bajo`

Excluidos : `Usuario confirmado` , `Alto` , `Medio` , `Bajo` , `Falso positivo`

resúmenes

Recuentos de alertas por riesgo y confianza

Esta tabla muestra el número de alertas para cada nivel de riesgo y confianza incluido en el informe.

(Los porcentajes entre paréntesis representan el recuento como porcentaje del número total de alertas incluidas en el informe, redondeado a un decimal).

		Confianza				
		Usuario confirmado	Alto	medio	bajos	Total
Riesgo	Alto	0 (0,0 %)	0 (0,0 %)	0 (0,0 %)	0 (0,0 %)	0 (0,0 %)
	medio	0 (0,0 %)	1 (6,7 %)	2 (13,3 %)	1 (6,7 %)	4 (26,7 %)
	bajos	0 (0,0 %)	2 (13,3 %)	4 (26,7 %)	1 (6,7 %)	7 (46,7 %)
	informativo	0 (0,0 %)	0 (0,0 %)	2 (13,3 %)	2 (13,3 %)	4 (26,7 %)
	Total	0 (0,0 %)	3 (20,0 %)	8 (53,3 %)	4 (26,7 %)	15 (100%)

Recuentos de alertas por sitio y riesgo

Esta tabla muestra, para cada sitio para el que se generaron una o más alertas, la cantidad de alertas generadas en cada nivel de riesgo.

Las alertas con un nivel de confianza de "Falso positivo" se han excluido de estos recuentos.

(Los números entre paréntesis son el número de alertas emitidas para el sitio en o por encima de ese nivel de riesgo).

		Riesgo			
		alto (= alto)	Medio (>= Medio)	Bajo (>= Bajo)	Informativo (>= Informativo)
Sitio	https://loginsai.es	0	4	7	4
	poch.edu.ec	(0)	(4)	(11)	(15)

Recuentos de alertas por tipo de alerta

Esta tabla muestra el número de alertas de cada tipo de alerta, junto con el nivel de riesgo del tipo de alerta.

(Los porcentajes entre paréntesis representan cada recuento como un porcentaje, redondeado a un decimal, del número total de alertas incluidas en este informe).

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

Tipo de alerta	Riesgo	Contar
HTTP ""X-Powered-By""		
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

Alerts

Risk=Medio, Confidence=Alto (1)

https://loginsai.espoch.edu.ec **(1)**

Content Security Policy (CSP) Header Not Set (1)

▶ GET https://loginsai.espoch.edu.ec/

Risk=Medio, Confidence=Medio (2)

https://loginsai.espoch.edu.ec (2)

Missing Anti-clickjacking Header (1)

▶ GET https://loginsai.espoch.edu.ec/

Vulnerable JS Library (1)

▶ GET https://loginsai.espoch.edu.ec/scripts.js

Risk=Medio, Confidence=Bajo (1)

https://loginsai.espoch.edu.ec (1)

Ausencia de fichas (tokens) Anti-CSRF (1)

▶ GET https://loginsai.espoch.edu.ec/vendor.js

Risk=Bajo, Confidence=Alto (2)

https://loginsai.espoch.edu.ec (2)

Server Leaks Version Information via "Server" HTTP Response Header Field (1)

▶ GET https://loginsai.espoch.edu.ec/

Strict-Transport-Security Header Not Set (1)

▶ GET https://loginsai.espoch.edu.ec/

Risk=Bajo, Confidence=Medio (4)

https://loginsai.espoch.edu.ec (4)

Cookie without SameSite Attribute (1)

▶ GET https://loginsai.espoch.edu.ec/

El servidor divulga información mediante un campo(s) de encabezado de respuesta HTTP ""X-Powered-By"" (1)

▶ GET https://loginsai.espoch.edu.ec/

Private IP Disclosure (1)

▶ GET https://loginsai.espoch.edu.ec/main.js

X-Content-Type-Options Header Missing (1)

▶ GET https://loginsai.espoch.edu.ec/

Risk=Bajo, Confidence=Bajo (1)

https://loginsai.espoch.edu.ec (1)

Divulgación de la marca de hora - Unix (1)

▶ GET https://loginsai.espoch.edu.ec/styles.css

Risk=Informativo, Confidence=Medio (2)

https://loginsai.espoch.edu.ec (2)

Modern Web Application (1)

▶ GET https://loginsai.espoch.edu.ec/

User Agent Fuzzer (1)

▶ GET https://loginsai.espoch.edu.ec/assets

Risk=Informativo, Confidence=Bajo (2)

https://loginsai.espoch.edu.ec **(2)**

Divulgación de información - Comentarios sospechosos (1)

▶ GET https://loginsai.espoch.edu.ec/runtime.js

Re-examine Cache-control Directives (1)

▶ GET https://loginsai.espoch.edu.ec/

Appendix

Alert types

This section contains additional information on the types of alerts in the report.

Ausencia de fichas (tokens) Anti-CSRF

Source	raised by a passive scanner (Ausencia de fichas (tokens) Anti-CSRF)
CWE ID	352
WASC ID	9
Reference	▪ http://projects.webappsec.org/Cross-Site-Request-Forgery

- http://cwe.mitre.org/data/definitions/352.html

Content Security Policy (CSP) Header Not Set

Source	raised by a passive scanner (Content Security Policy (CSP) Header Not Set)
CWE ID	693
WASC ID	15
Reference	• https://developer.mozilla.org/en-US/docs/Web/Security/CSP/Introducing_Content_Security_Policy • https://cheatsheetseries.owasp.org/cheatsheets/Content_Security_Policy_Cheat_Sheet.html • http://www.w3.org/TR/CSP/ • http://w3c.github.io/webappsec/specs/content-security-policy/csp-specification.dev.html • http://www.html5rocks.com/en/tutorials/security/content-security-policy/ • http://caniuse.com/#feat=contentsecuritypolicy • http://content-security-policy.com/

Missing Anti-clickjacking Header

Source	raised by a passive scanner (Anti-clickjacking Header)

CWE ID	1021
WASC ID	15
Reference	▪ https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options

Vulnerable JS Library

Source	raised by a passive scanner (Vulnerable JS Library (Powered by Retire.js))
CWE ID	829
Reference	▪ https://github.com/jquery/jquery/issues/2432 ▪ http://blog.jquery.com/2016/01/08/jquery-2-2-and-1-12-released/ ▪ http://research.insecurelabs.org/jquery/test/ ▪ https://blog.jquery.com/2019/04/10/jquery-3-4-0-released/ ▪ https://nvd.nist.gov/vuln/detail/CVE-2019-11358 ▪ https://nvd.nist.gov/vuln/detail/CVE-2015-9251 ▪ https://github.com/jquery/jquery/commit/753d591aea698e57d6db58c9f722cd0808619b1b ▪ https://bugs.jquery.com/ticket/11974 ▪ https://blog.jquery.com/2020/04/10/jquery-3-5-0-released/

Cookie without SameSite Attribute

Source	raised by a passive scanner (Cookie without SameSite Attribute)
CWE ID	1275
WASC ID	13
Reference	▪ https://tools.ietf.org/html/draft-ietf-httpbis-cookie-same-site

Divulgación de la marca de hora - Unix

Source	raised by a passive scanner (Divulgación de la marca de hora)
CWE ID	200
WASC ID	13
Reference	▪ http://projects.webappsec.org/w/page/13246936/Information%20Leakage

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

Source	raised by a passive scanner (El servidor divulga información mediante un campo(s) de encabezado de respuesta HTTP ""X-Powered-By"")
CWE ID	200
WASC ID	13
Reference	▪ http://blogs.msdn.com/b/varunm/Archive/2013/04/23/Remove-Unwanted-http-Response-headers.aspx

http://www.troyhunt.com/2012/02/shhh-don't-deje-la-respuesta-headers.html

Private IP Disclosure

Source	raised by a passive scanner (Private IP Disclosure)
CWE ID	200
WASC ID	13
Reference	• https://tools.ietf.org/html/rfc1918

Server Leaks Version Information via "Server" HTTP Response Header Field

Source	raised by a passive scanner (HTTP Server Response Header)
CWE ID	200
WASC ID	13
Reference	• http://httpd.apache.org/docs/current/mod/core.html#servertokens • http://msdn.microsoft.com/en-us/library/ff648552.aspx#ht_urlscan_007 • http://blogs.msdn.com/b/varunm/archive/2013/04/23/remove-unwanted-http-response-headers.aspx • http://www.troyhunt.com/2012/02/shhh-dont-let-your-response-headers.html

Strict-Transport-Security Header Not Set

Source	raised by a passive scanner (Strict-Transport-Security Header)
CWE ID	319
WASC ID	15
Reference	▪ https://cheatsheetseries.owasp.org/cheatsheets/HTTP_Strict_Transport_Security_Cheat_Sheet.html ▪ https://owasp.org/www-community/Security_Headers ▪ http://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security ▪ http://caniuse.com/stricttransportsecurity ▪ http://tools.ietf.org/html/rfc6797

X-Content-Type-Options Header Missing

Source	raised by a passive scanner (X-Content-Type-Options Header Missing)
CWE ID	693
WASC ID	15
Reference	▪ http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx ▪ https://owasp.org/www-community/Security_Headers

Divulgación de información - Comentarios sospechosos

Source	raised by a passive scanner (Divulgación de información - Comentarios sospechosos)
CWE ID	200
WASC ID	13

Modern Web Application

Source	raised by a passive scanner (Modern Web Application)

Re-examine Cache-control Directives

Source	raised by a passive scanner (Re-examine Cache-control Directives)
CWE ID	525
WASC ID	13
Reference	▪ https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#web-content-caching ▪ https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cache-Control ▪ https://grayduck.mn/2021/09/13/cache-control-recommendations/

Fuzzer de agente de usuario

Fuente	generado por un escáner activo (User Agent Fuzzer)
Referencia	▪ https://owasp.org/wstg

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

DIRECCIÓN DE BIBLIOTECAS Y RECURSOS DEL APRENDIZAJE

UNIDAD DE PROCESOS TÉCNICOS

REVISIÓN DE NORMAS TÉCNICAS, RESUMEN Y BIBLIOGRAFÍA

Fecha de entrega: 16 / 06 / 2023

INFORMACIÓN DE LAS AUTORAS
Nombres – Apellidos: ERIKA MICHELLE ASTUDILLO MUÑOZ ANDREA ELIZABETH VIZUETE ULLOA
INFORMACIÓN INSTITUCIONAL
Facultad: INFORMÁTICA Y ELECTRÓNICA
Carrera: SOFTWARE
Título a optar: INGENIERA DE SOFTWARE
f. Analista de Biblioteca responsable: Ing. Fernanda Arévalo M.

UNIDAD DE PROCESOS TÉCNICOS DBRA ESPOCH

1120-DBRA-UPT-2023