Universidad Internacional de La Rioja (UNIR)

ESIT

Máster universitario en Seguridad Informática

Ataques al protocolo de autenticación OPENID CONNECT + OAUTH2

Trabajo Fin de Máster

presentado por: Quiñónez Murillo, Carlos Andrés

Director/a: Coz Fernandez, José Ramón

Ciudad: Durán – Ecuador

Fecha: 07/02/2022

RESUMEN

El trabajo ha realizado un análisis en profundidad de las metodologías y patrones de ataque a los protocolos de autenticación OPENID Connect con OAUTH2 en entornos de prueba que simulan implementaciones reales, así como la evaluación de riesgo que representa para las aplicaciones web

Como parte del trabajo se han identificado recomendaciones para realizar implementaciones seguras de estos protocolos minimizando el riesgo de afectaciones a entornos de producción de las organizaciones, aportando una mejora de los actuales esquemas de ciberseguridad.

Palabras Clave: Autenticación, Ataque, Ciberseguridad, Protocolo, Riesgo

ABSTRACT

It has been performed an In-depth analysis of the methodologies and attack patterns to the OPENID Connect authentication protocols with OAUTH2 in test environments that simulate real implementations, as well as the risk assessment that it represents for web applications

As part of the activities, it has been identified recommendations to carry out secure implementations of these protocols minimizing the risk of damage to production environments of organizations contributing to the improvement of current cybersecurity schemes.

Keywords: Authentication, Attack, Cybersecurity, Protocol, Risk

INDICE

Tabla de Ilustraciones

Tablas

Anexos

1. INTRODUCCIÓN

1.1. Motivación

El protocolo de autenticación OPENID Connect están en auge a raíz de la necesidad de implementar mecanismos de autenticación robustos y simples desde la perspectiva y experiencia de usuario.

Existe un importante interés en evaluar y analizar en profundidad estos esquemas, ya que la autenticación es uno de los principales procesos que se buscan vulnerar por parte de atacantes malintencionados que han logrado vulnerar derechos fundamentales de las personas en la sociedad como la privacidad y confidencialidad.

En el ámbito profesional las motivaciones van de la mano de los deseos de aportar a la comunidad de ciberseguridad e informática en general, buscando generar conciencia sobre los riesgos que involucran el mal uso e implementación de los esquemas de autenticación. Desde el punto de vista profesional, he sido testigo de cómo muchos desarrolladores realizan implementaciones en las aplicaciones de software de manera irresponsable. Por otro lado, el desarrollo de la tecnología requiere que estos mecanismos de control de acceso para identificar y autorizar a los usuarios no solo sean seguros, sino que también sean simples y sencillos de usar, siendo ese el principal desafío.

1.2. Planteamiento del trabajo

1.2.1. Metodología

La metodología que se aplicará consta de tres fases

1. Estudio del Estado del Arte
2. Creación de Laboratorio de trabajo
3. Pruebas de Penetración

El objetivo de la fase inicial, **el estudio del estado del arte** es lograr un entendimiento suficiente y necesario para dominar teóricamente los conceptos para cumplir los propósitos del trabo fin de máster (TFM). Para ello se analizarán las siguientes etapas referentes al protocolo OpenId Connect:

1. Marco Metodológico y de riesgos.
2. Técnicas de Ataque y principales vulnerabilidades
3. Historia del protocolo y su evolución

La segunda fase consiste en la preparación del laboratorio o entorno de evaluación en la que se definirán las plataformas de hardware y/o software, procedimientos que serán utilizados para los análisis correspondientes, incluyendo la ejecución de las Pruebas de Penetración al protocolo de autenticación OPENID Connect que permitan determinar y medir su robustez.

1.2.2. Objetivos

El objetivo general del presente TFM consiste en aportar con la mejora en la implementación de los esquemas de autenticación de manera segura. Se definen los siguientes objetivos específicos:

1. Realizar el estudio del arte sobre el protocolo de autenticación OPENID Connect analizando sus componentes, estructura, características e historia.
2. Identificar y evaluar las técnicas de ataque a los protocolos OPENID Connect
3. Identificar y evaluar el riesgo y exposición a las que pueden estar sujetos los protocolos estudiados.
4. Identificar y analizar herramientas para realizar test de penetración al protocolo de autenticación.
5. Implementar un ambiente para realizar la prueba de penetración y poder evaluarlos sin perjuicio a terceros.
6. Recomendar consideraciones para la implementación segura del protocolo de autenticación.

1.3. Estructura del trabajo.

El presente trabajo se realiza con una estructura organizada con la intención de facilitar su lectura y generar interés sobre el tema, así como lograr que cada tópico sea incluido en el momento propicio y el detalle suficiente para que la información sirva para la interpretación adecuada evitando datos irrelevantes para la investigación.

En primer lugar, se incluirá el contexto técnico de investigación referente a la definición, uso e historia del protocolo de autenticación objetos de este estudio. Luego se pretende exponer las herramientas y técnicas de implementación tradicionales de los protocolos OPENID Connect en un ambiente de evaluación controlado que permita realizar análisis sin perjudicar a terceros.

Dentro del desarrollo del TFM se describen el objetivo general y los específicos, así como las metodologías de estudio y evaluación, seguido del desarrollo del experimento, presentación de resultados, conclusiones y recomendaciones.

Los capítulos del presente documento son:

1. Introducción
    1.1. Motivación
    Se describen los motivos de la elección del tema, y las razones por las que considero que se debe tratar, así como el aporte que se puede dar a la comunidad de ciberseguridad.
    1.2. Planteamiento del trabajo
    Se resume brevemente cómo se analizará el problema y se describen los objetivos que se esperan cumplir al término de la investigación
    1.3. Estructura del Trabajo
    Se narra de forma breve el desarrollo de los capítulos, incluyendo un resumen del documento.
2. Contexto y Estado del ArteMarco Metodológico y de riesgos
    Describir los fallos generales en el proceso de autenticación y el contexto de evaluación de los riesgos más comunes
    2.2. Ataques y Vulnerabilidades
    Esta sección es para describir las técnicas y metodologías más importantes usadas para realizar ataques al protocolo de autenticación a evaluarse en el presente trabajo.
    2.3. Historia del protocolo de autenticación OPENID Connect
    Se desarrolla contextualmente el origen y evolución. Se pretende plantear las bases teóricas y situación actual del protocolo que será analizado en la investigación.

3. Objetivos y metodología del trabajo
    3.1. Objetivo general
        Se explica el objetivo principal del TFM
    3.2. Objetivos específicos
        Se detallan los aspectos fundamentales para alcanzar el objetivo general
    3.3. Metodología del trabajo
        Se describe de manera general los pasos a seguir para alcanzar los objetivos y el entorno en que se realizará la investigación y el experimento.
4. Desarrollo específico de la contribución
    4.1. Descripción detallada del experimento
        Se detalla y justifica las tecnologías utilizadas para la investigación, organización del piloto experimental, como se llevó a cabo y las técnicas utilizadas en su análisis.
    4.2. Presentación de los resultados
        En este segmento se presentarán detalladamente los resultados obtenidos mediante gráficos y tablas que permitan referenciar y comprender claramente el experimento realizado.
    4.3. Discusión de los resultados
        En esta sección se pretende justificar los resultados obtenidos durante el experimento explicando y resaltando aquellos aspectos más relevantes como objeto de consideración para las conclusiones del experimento.
5. Conclusiones y trabajo futuro
    5.1. Análisis sobre el TFM
        Se presenta un resumen comparativo de la situación polémica y los resultados del experimento.
    5.2. Contribuciones del trabajo
        Especificar lo que se considera como aporte a la ciberseguridad sobre el estudio a los ataques del protocolo de autenticación estudiados en el presente TFM, así como la comparativa entre los objetivos y resultados
    5.3. Líneas de trabajo futuro

Se realiza una valoración del aporte y como se puede mejorar los esquemas de autenticación en las implementaciones de software.

2. CONTEXTO Y ESTADO DEL ARTE

2.1. Marco metodológico y de riesgos

(López, 2017) en su artículo menciona que el crecimiento y la evolución de la red de internet ha obtenido un impacto de vital importancia en la forma de establecer canales de comunicación por los usuarios para comunicarse con personas a través de redes sociales, sistemas de video conferencia y demás por medio de una aplicación web y la manera de realizar operaciones en línea, generando una gran cantidad de información confidencial.

La inclusión de una gran cantidad de sitios de E-Commerce, bancarios, redes sociales, entre otros, están volviéndose una necesidad y cada vez son más populares, ya que podemos ver miles de usuarios interactuando con estos sitios y aplicaciones web llamando poderosamente la atención de cibercriminales quienes buscan obtener la información que les permita acceder de manera fraudulenta a los destinos con diferentes finalidades. Además, (López, 2017) indica que los dispositivos conectados a internet son susceptibles a intrusiones maliciosas ejecutadas por ciberdelincuentes quienes tienen la capacidad de poner en riesgo las aplicaciones web para tener el acceso fraudulento a la información más valiosa de las personas y empresas, y al objeto de malograr la mayoría de los activos de información.

En el ámbito tecnológico, comúnmente, las exposiciones y vulnerabilidades de las aplicaciones web son provocadas por malas prácticas de los responsables del desarrollo y arquitectura de los sitios, por lo tanto, se vuelve fundamental promover la relevancia e importancia de la seguridad y medidas de prevención que deben tener los sitios web, y no solo deben cumplir las funciones u objetivos específicos necesarios para la operación de las empresas.

Un estudio realizado por la empresa especialista en desarrollo seguro de aplicaciones web (VERACODE, 2021) a 130.000 aplicaciones escaneadas durante los últimos 12 meses, determinó que el 76 por ciento de las aplicaciones tienes al menos una vulnerabilidad, mientras que el 24% corresponde a vulnerabilidades de mayor nivel de

riesgo. La mitad de los hallazgos siguen vigentes 6 meses después del descubrimiento.

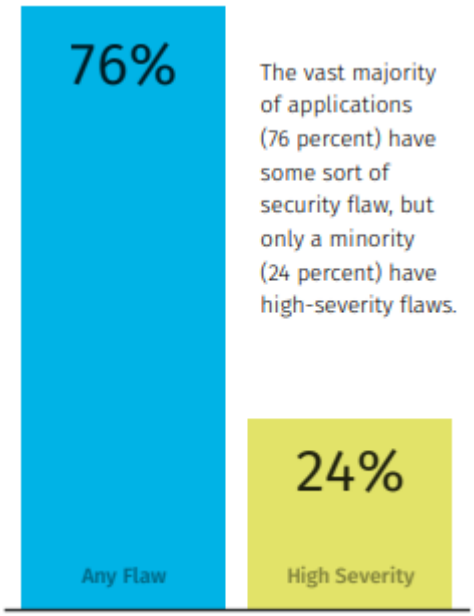

Ilustración 1 - Flaw Severity
Fuente: https://www.veracode.com/sites/default/files/pdf/resources/sossreports/state-of-software-security-volume-11-veracode-report.pdf

En el reporte (VERACODE, 2021) expone que el objetivo de la seguridad del software no es escribir aplicaciones a la perfección la primera vez, sino remediar las fallas de manera integral y oportuna.

Hoy en día es muy común escuchar noticias y comunicados sobre ataques cibernéticos a instituciones, es por esta razón que la necesidad de establecer medidas preventivas se ha vuelto necesario para las empresas para evitar afectar a su imagen corporativa. Según (VERACODE, 2021) el programa de seguridad denominado OWASP Top 10 es una lista de los 10 riesgos de seguridad de aplicaciones web más comunes. Al escribir código y realizar pruebas sólidas con estos riesgos en mente, los desarrolladores pueden crear aplicaciones seguras que mantienen los datos confidenciales de sus usuarios a salvo de los atacantes.

El OWASP Top 10 no solo lista las principales vulnerabilidades de las aplicaciones web, sino que también evalúa los diferentes fallos mediante la denominada "Metodología de Calificación de riesgo de OWASP" que proporciona herramientas y mejores prácticas para prevenir ataques cibernéticos, permitiendo a los desarrolladores tomar medidas para mantener a los usuarios seguros cuando acceden a las aplicaciones.

2.1.1. OWASP

El proyecto de seguridad de aplicaciones Web Open Source (OWASP, por sus siglas en inglés) es una comunidad abierta dedicada a permitir que las organizaciones desarrollen, compren y mantengan aplicaciones y API en las que se pueda confiar creado por (OWASP Foundation, 2021). OWASP es una fundación sin fines de lucro que trabaja para mejorar la seguridad del software. A través de proyectos de software de código abierto liderados por la comunidad, cientos de capítulos locales en todo el mundo, decenas de miles de miembros y conferencias educativas y de capacitación líderes, la Fundación OWASP es la fuente para que los desarrolladores y tecnólogos protejan la web mediante herramientas, recursos tecnológicos, blogs, capacitación.

Los informes de estos proyectos se emiten generalmente cada 4 años en los que se detallan las principales vulnerabilidades tanto para aplicaciones web como para aplicaciones móviles, así como manuales de buenas prácticas con argumentos para que los desarrolladores y empresas se basen con el objetivo de proteger los datos de sus aplicaciones.

La última entrega es el OWASP Top 10: 2021 que a diferencia de su antecesor emitido en 2017 contiene nuevas categorías, y otras con cambios en nomenclatura y alcance. (OWASP Foundation, 2021)

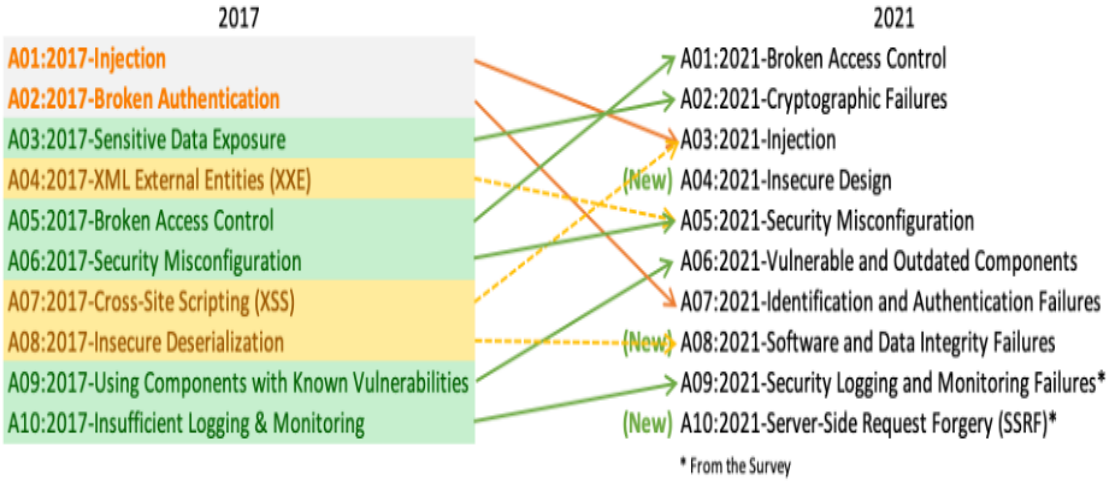

Ilustración 2 - Cambios del OWASP Top10 2021

Fuente: https://OWASP.org/Top10/A00_2021_Introduction/

A continuación, se detallan los principales cambios (Borghello, 2021):

- A01: 2021 - Broken Access Control se posiciona en el primer lugar luego de haber estado en el quinto puesto, en las pruebas realizadas el 94% de las aplicaciones presentaron algún tipo de problema de control de acceso.
- A02: 2021 – Los Fallos criptográficos escalan al segundo lugar. En la versión anterior se denominaba "Exposición de datos confidenciales". Se relaciona a un nuevo enfoque relacionado con la criptografía, ya que generalmente, exponen datos confidenciales o comprometer a los sistemas.
- A03: 2021 - La inyección de código se posiciona de manera sorpresiva en la tercera posición, luego de evaluar, el 94% de las aplicaciones presentaron alguna vulnerabilidad de inyección, que incluye Cross Site Scripting y los CWE utilizados en esta categoría presenta la segunda mayor cantidad de ocurrencias en aplicaciones.
- A04: 2021 – En el cuarto lugar se posiciona una nueva categoría denominada “Diseño inseguro” basado en un enfoque de riesgos relacionados con fallas de la etapa del diseño de los sistemas recomendando un mayor uso del modelado de amenazas, patrones y principios y arquitecturas de diseño seguro.
- A05: 2021 - La categoría relacionada a la “configuración incorrecta de seguridad” se ubica en el quinto punto escalando una posición en relación a la versión

anterior de OWASP con un 90% de aplicaciones vulnerables que fueron evaluadas. Las entidades externas XML (XXE) se unifica en esta categoría.

- A06: 2021 – Los “componentes vulnerables y obsoletos” se refiere a dependencias de los sistemas que no son actualizados de manera oportuna heredando sus problemas de seguridad a los sistemas, es un problema conocido difícil de probar y evaluar el riesgo. Escala desde la posición novena en 2017 a la sexta en 2021.
- A07: 2021 - Fallas de identificación y autenticación, bajo desde la segunda posición en 2017 a la séptima en esta versión. Anteriormente se denominaba "pérdida de autenticación". La presencia y uso de estándares como OpenId Connect entre otros… esta dando sus frutos, permitiendo a las aplicaciones simplificar de manera segura el esquema de autenticación.
- A08: 2021 – En la octava posición aparece una nueva categoría a la que denominaron “Fallas de software e integridad de datos”. La deserialización insegura es parte de esta categoría que unifica a los problemas relacionados con actualizaciones de software, datos críticos y canalizaciones de integración y entrega continua (CI/CD) sin verificar la integridad.
- A09:2021 - Las fallas de registro y monitoreo de seguridad, que en 2017 eran conocidas como "Registro y monitoreo insuficientes". La categoría es ampliada para incluir otros tipos de fallas que pueden afectar directamente la visibilidad de amenazas de ciberseguridad, alertas de incidentes y análisis forenses.
- A10:2021 - La falsificación de solicitudes del lado del servidor (Server-Side Request Forgery - SSRF). Esta categoría es incluida por la evaluación a las empresas e industrias.

Se puede determinar que OWASP Top10: 2021 pasó de ser una lista de clasificación de vulnerabilidades a una lista de los riesgos más comunes detectados en el último periodo de tiempo. ***Anexo A - Análisis detallado de OWASP 2021***

En el Anexo B – Factores Análisis OWASP Top 10 – 2021 se resume mediante una tabla las ponderaciones utilizada sen la evaluación de riesgos del Top10 de Amenazas del presente año.

2.1.2. Metodología OWASP

Según (OWASP Foundation, 2021) en su última versión, ocho de las categorías están basadas en los datos de las investigaciones y dos en una encuesta a la comunidad realizada a expertos en seguridad y desarrollo de aplicaciones de primer nivel.

Las categorías del reporte del 2017 se basaron en aproximadamente 30 debilidades comunes (CWE); el problema fue que los desarrolladores y empresas se limitaron a la prevención evitando la inclusión de otras debilidades en sus análisis. Para el reporte 2021 se utilizaron datos de aproximadamente 400 CWE incluyendo datos para el análisis sin limitarse al código CWE. Este aumento impacta en la forma en que se estructuran las categorías, centrándose en la causa raíz y no solamente en los síntomas. ***Anexo D - Proceso CWE de Mitre.***

2.1.3. Recopilación y Análisis de Datos

Para (Borghello, 2021) la nueva entrega del Top 10 está más basada en datos que nunca. Se seleccionaron ocho de las diez categorías desde los datos aportados y dos categorías desde una encuesta de la industria de alto nivel. Se hace de esta forma por una razón fundamental: mirar los datos aportados es "mirar el pasado". Los investigadores de AppSec se toman su tiempo para encontrar nuevas vulnerabilidades y formas de probarlas. Se necesita tiempo para integrar estas pruebas en herramientas y procesos. Para cuando se puede probar de manera confiable una debilidad a escala, es probable que hayan pasado años. Para equilibrar ese punto de vista, se utiliza una encuesta de la industria para preguntar a las personas en primera línea qué ven como debilidades esenciales que los datos pueden no mostrar aún. (Owasp Foundation, 2021)

2.1.4. Uso OWASP Top 10 como estándar.

El OWASP Top10 es un documento para concientizar a programadores y personal de seguridad de la información, sin embargo (Owasp Foundation, 2021) menciona que las empresas lo utilicen como un estándar para alinear los objetivos de control de seguridad en el desarrollo de las aplicaciones, permitiendo priorizar basado en los riesgos más importantes y relevantes para la seguridad de las empresas.

La versión 2021, OWASP Top10 está basada en la identificación de riesgos mediante una metodología de evaluación propia que contiene además esquemas para modelar las amenazas convirtiéndose en una importante herramienta para prevención de seguridad que ponen a disposición de los especialistas de seguridad en todo el mundo. Uno de los principales problemas de utilizarlo como estándar es interpretar de forma adecuada cada categoría ya al estar basada en riesgo, cada categoría puede involucrar una gran cantidad de vulnerabilidades (CWE).

En la siguiente imagen se listan las principales recomendaciones sobre cuando es factible adoptar este Estándar.

Caso de uso	OWASP Top 10 2021	Estándar de verificación de seguridad de aplicaciones OWASP
Conciencia	sí	
Capacitación	Nivel Básico	Exhaustivo
Diseño y arquitectura	De vez en cuando	sí
Estándar de codificación	Mínimo	sí
Revisión de código seguro	Mínimo	sí
Lista de verificación de revisión por pares	Mínimo	sí
Examen de la unidad	De vez en cuando	sí
Pruebas de integración	De vez en cuando	sí
Pruebas de penetración	Mínimo	sí
Soporte de herramientas	Mínimo	sí
Cadena de suministro segura	De vez en cuando	sí

Ilustración 3 - Casos de Uso para adoptar OWASP como Estándar.
Fuente: https://OWASP.org/Top10/A00_2021_How_to_use_the_OWASP_Top_10_as_a_standard/

El estándar de verificación seguridad de OWASP (ASVS) está diseñado para utilizarse en todo el ciclo de vida de desarrollo seguro, su última versión es 4.0.2 liberada en octubre de 2020 y está dividida en 14 dominios como se detallan en el ***Anexo C – Objetivos de Control OWASP***

Este documento sirve como guía para la evaluación de seguridad de las aplicaciones tanto para organizaciones que desarrollan, proveen e inclusive para los responsables de seguridad de las organizaciones.

Según (OWASP Foundation, 2021), el Estándar de verificación de seguridad de aplicaciones define tres niveles de verificación de seguridad. El primero para niveles de seguridad bajos estableciendo medidas necesarias para comprobar la penetración.

El segundo nivel es para aplicaciones que contienen datos sensibles, que requieren protección siendo el nivel más recomendado para la mayoría de las aplicaciones. El tercer nivel de ASVS es para las aplicaciones más críticas, aplicaciones que realizan transacciones de alto valor, contienen datos sensibles y confidenciales como la historia clínica de pacientes o cualquier aplicación que requiera el más alto nivel de confianza.

Los niveles de ASVS contienen una lista de requisitos de seguridad como se muestra en la siguiente imagen.

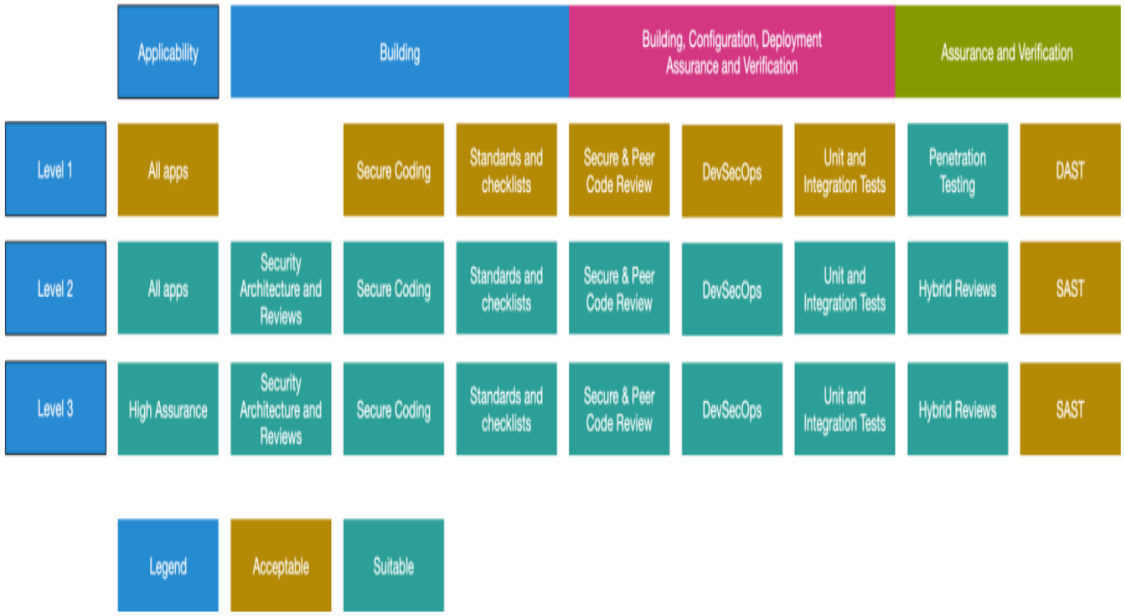

Ilustración 4 - Niveles del Estándar de verificación de Seguridad OWASP
Fuente: https://OWASP.org/www-project-application-security-verification-standard/

Los requisitos también se pueden asignar a funciones y capacidades específicas de seguridad que los desarrolladores deben integrar en el software.

2.1.5. Marco de gestión de riesgos para aplicaciones Web

Según (Williams, 2021), el descubrimiento de las vulnerabilidades es importante, pero ser capaz de estimar el riesgo asociado a la empresa es igualmente importante. Por tal razón se considera que durante las etapas iniciales del ciclo de vida de desarrollo de una aplicación es completamente factible identificar fallos de seguridad en la

arquitectura o diseño mediante el uso de herramientas para modelar amenazas, el uso de herramientas para revisar el código fuente es otra fuente de detección temprana de posibles problemas de seguridad en etapas tempranas del ciclo de vida siempre que estén correctamente enfocadas, caso contrario ciertos fallos no podrán ser descubiertos hasta que la aplicación sea completamente funcional.

De igual manera (Williams, 2021), menciona que es posible estimar la gravedad de los riesgos para el negocio y tomar una decisión acertada sobre qué hacer con los mismos, por lo tanto, el uso de sistemas para calificar los riesgos permitirá reducir tiempos y evitará situaciones para tomar decisiones sobre priorizar la implementación de controles de seguridad o distracciones con problemas de menor riesgo y criticidad ignorando los más relevantes. OWASP propone una metodología para evaluación del riesgo de las amenazas. Se encuentra detallado en el ***Anexo E - EVALUACION DE RIESGOS EN APLICACIONES WEB***

2.1.6. Autenticación

En contexto, en los últimos años, y especialmente en 2021 gracias a factores externos, la transformación digital de las empresas provocó una acelerada demanda de los servicios públicos y privados al internet, incrementado las necesidades de prevenir incidentes para evitar la modificación de los datos y poder garantizar la confidencialidad, integridad y disponibilidad.

Según (Deitel, Deitel, & Deitel, 2014), la autenticación y la autorización son los primeros parámetros de seguridad que las aplicaciones web consideran para protegerse de accesos no autorizados

Según (Electronic Identification, 2021) La autenticación es el proceso de verificar que una persona, entidad o sitio web es quien dice ser. Para las aplicaciones web, autenticar significa enviar identificadores y datos privados que solo el propietario puede conocer o tener.

La gestión de sesiones según (OWASP Foundation, 2021) es un proceso mediante el cual un servidor mantiene el estado de una entidad que interactúa con él. Esto es necesario para que un servidor recuerde cómo reaccionar a las solicitudes posteriores

a lo largo de una transacción. Las sesiones se mantienen en el servidor mediante un identificador de sesión que se puede pasar hacia atrás y hacia adelante entre el cliente y el servidor al transmitir y recibir solicitudes.

A continuación, desarrollaremos unos conceptos que son críticos para nuestro análisis:

Autenticación mediante usuario y contraseñas.

Según (Kukic, 2020) la autenticación de nombre de usuario y contraseña es el método probado y verdadero de protección aplicaciones. En este tipo de autenticación, un usuario simplemente proporciona su nombre de usuario o identificador y contraseña, que se compara con una base de datos de usuarios, y si las credenciales coinciden, el usuario está autenticado.

Hoy en día es muy común ver que las aplicaciones web utilizan como identificador el correo electrónico o el número de teléfono móvil pero el esquema y flujo de autenticación es el mismo. En aplicación críticas que requieren altos niveles de seguridad estos datos pueden ser gestionados de manera confidencial, es decir que ni el usuario los conoce. Sin embargo, los problemas de seguridad generalmente asociados a descuidos de los programadores como, por ejemplo, el almacenamiento de credenciales en texto plano o canales de comunicación con esquemas vulnerables aumentan la probabilidad de que esos datos caigan en manos de personas inescrupulosas para cometer delitos informáticos.

Mejorar el flujo de autenticación de nombre de usuario y contraseña viene en el sabor de hacer cumplir requisitos de contraseña estrictos, forzar cambios de contraseña de vez en cuando, y evitar la reutilización de contraseñas. Los requisitos de contraseña segura pueden variar desde contraseña longitud según los requisitos de caracteres especiales, números, etc. (Kukic, 2020)

Implementar directivas para gestionar la autenticación permiten reducir la probabilidad de que estos datos sean mal utilizados por atacantes, entre los más importantes tenemos:

- Política de contraseñas. Tener la capacidad de controlar las características para la generación de contraseñas de una manera segura como definir la cantidad mínima de caracteres, garantizar la rotación de credenciales, incluir medidores de seguridad para ayudar a los usuarios a determinar el nivel de complejidad de la contraseña que está registrando, evitar el uso de contraseñas de fácil deducción
- Mecanismo seguro de recuperación de contraseña. Uso de mecanismos para que el usuario pueda recuperar la contraseña de manera segura minimizando la posibilidad de compromisos de las cuentas.
- Almacenamiento seguro de contraseñas. Utilizar funciones criptográficas robustas para almacenar las credenciales en repositorios de datos como Bases de datos.
- Comparación de Hashes mediante funciones seguras. Consiste en la comparación del hash que se obtiene en el campo de ingreso de las credenciales versus el hash de la contraseña almacenada, si esto no es posible por limitantes externas es recomendable definir umbrales para el tiempo y longitud para evitar ataques de denegación de servicios
- Transmisión segura de contraseñas. El uso de canales de comunicación con algoritmos de cifrado robustos como TLS para evitar que atacantes puedan ver el ID de Usuario o sesión y comprometer la privacidad del usuario final.
- Re-autenticación para funciones sensibles. En transacciones críticas que requieren mayores niveles de seguridad es importante implementar esquemas de Re-autenticación para evitar ataques de inyección de credenciales especializados que están diseñados para evitar los esquemas de autenticación tradicionales. Ejemplo, registrar nuevos destinos de compras.
- Autenticación de Cliente. Según (OWASP Foundation, 2021) consiste en que el navegador y el servidor envíen sus respectivos certificados TLS durante el proceso de negociación TLS. Es un proceso muy robusto y recomendado, sin embargo, es importante evaluar la necesidad de este tipo de esquemas para aplicaciones públicas de alta transaccionalidad ya que la instalación de certificados es un proceso que dependerá de personal especializado. Está

recomendada para aplicaciones muy críticas como implementación de firma electrónica.

- Mensajes de autenticación y error. El manejo de errores de autenticación es crítico para el diseño de una aplicación. Un mal manejo puede proporcionar información a los atacantes que le permita configurar ataques, por ejemplo, una respuesta de error de autenticación incorrecta que generalmente se utiliza es: *"Error de inicio de sesión, ID de usuario no válido".* Este mensaje permitirá al atacante identificar los ID de usuario válidos y enumerarlos. Lo correcto es mostrar el siguiente mensaje: "Error de inicio de sesión; ID de usuario o contraseña no válidos"

Otros métodos de autenticación generalmente utilizados son la delegación de autenticación a terceros mediante los denominados Proveedores de Identidad (IDP) en el que terceros (Facebook, Google) se encargan de coordinar los procesos inicio de sesión de las aplicaciones web y autenticación sin credenciales que consisten en el uso de tokens que se envían a dispositivos previamente registrados que incluyen mensajes SMS, correo electrónico e inclusive el uso de sensores biométricos haciendo que esta tecnología sea más robusta que el tradicional esquema de usuario y contraseñas

Autorización

La autorización se encarga de garantizar que los usuarios tengan los niveles correctos de acceso dentro de un sistema. Una vez que un usuario se autentica exitosamente, se le otorgan ciertos permisos por lo que pueden hacer dentro de una aplicación. (Kukic, 2020)

Gestor de Identidad

La gestión de identidades es responsable de crear, actualizar y eliminar usuarios a medida que moverse por una organización. (Kukic, 2020)

Generalmente cuando una persona ingresa a trabajar en una organización, y entre sus responsabilidades se encuentra el uso de herramientas tecnológicas, seguramente el área de recursos humanos solicita a las áreas de TI la creación de

accesos en las diferentes aplicaciones necesarias para las actividades. Los usuarios sin conciencia de seguridad reutilizarán la misma contraseña en todas las aplicaciones o en su defecto recordar credenciales por cada aplicación se vuelve un problema que provoca acciones no recomendadas como anotar claves en medios inseguros o provocando dificultades para acceder a los sistemas informáticos. Como medida para minimizar estos problemas se implementan los denominados gestores de identidad que son herramientas que centralizan la gestión de accesos a las diferentes plataformas, proporcionando un inicio de sesión unificado y simple. (SSO). Estas plataformas utilizan diferentes protocolos para tal efecto como los siguientes:

- OAuth. Es un estándar abierto que permite flujos simples de autorización para sitios web o aplicaciones informáticas. Se trata de un protocolo propuesto por Blaine Cook y Chris Messina, que permite autorización segura de una API de modo estándar y simple para aplicaciones de escritorio, móviles y web (Parecki, 2021)
- OpenId. Es un estándar de identificación digital descentralizado, con el que un usuario puede identificarse en una página web a través de una URL (OpenID Foundation, 2020)
- SAML. Es un estándar de código abierto basado en XML que permite el intercambio de información, tanto de autenticación como de autorización entre diferentes partes: un proveedor de identidad y un proveedor de servicios. (OASIS, 2014)
- Federación de Identidad. Es un sistema de confianza entre dos partes con el fin de autenticar a los usuarios y transmitir la información necesaria para autorizar su acceso a los recursos (Amazon Web Services, 2021)
- Open ID Connect. Permite a los Clientes verificar la identidad del Usuario Final basándose en la autenticación realizada por un Servidor de Autorización, así como obtener información de perfil básico sobre el Usuario Final de una manera interoperable (OpenID, 2020)

2.2. Ataques y Vulnerabilidades a Protocolos de Autenticación

2.2.1. Ataques Informáticos.

Según (Mieres, 2009), los ataques informáticos se refieren al aprovechamiento de una debilidad o falla en el software, hardware, o en las personas que componen un ambiente informático; para obtener algún tipo de beneficio, especialmente de tipo económico, provocando un efecto negativo en la seguridad del sistema, afectando directamente en los activos de la organización.

Los atacantes generalmente son personas con conocimientos técnicos avanzados en los campos de programación, comunicaciones y seguridad informática, por lo que utilizan metodologías para establecer estrategias que le permitan tener éxito en sus intenciones. La intención con la que realizan sus actividades son las que hacen la diferencia y los clasifica en Maliciosos o Educativos.

En la siguiente imagen se muestra las fases comunes que según (Mieres, 2009) utilizan los atacantes:

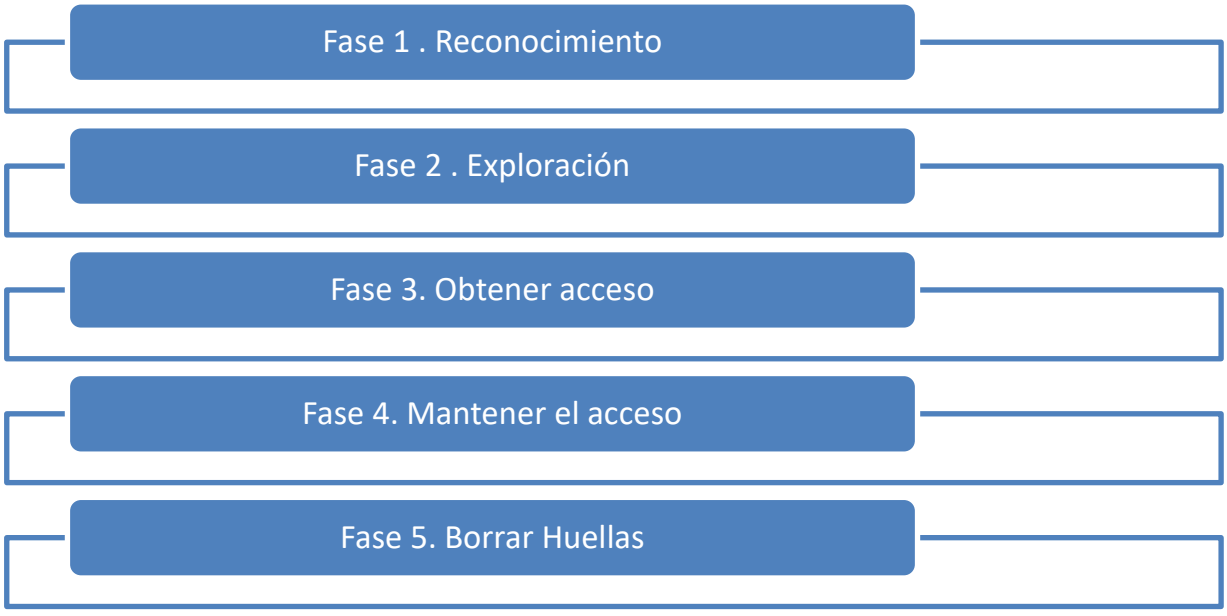

Ilustración 5 - Fases comunes de ataques informáticos
Fuente: https://www.evilfingers.net/publications/white_AR/01_Ataques_informaticos.pdf

Fase 1. Reconocimiento. Esta fase incluye el proceso de obtener datos e información de una víctima potencial que puede incluir personas, empresas y cualquier tipo de organización. Durante esta fase se utiliza diferentes recursos de Internet como para recolectar datos del objetivo. Entre las técnicas utilizadas se involucra la Ingeniería Social, el Dumpster Diving, el sniffing. (Mieres, 2009)

Fase 2. Exploración. Se utiliza la información obtenida en la fase de reconocimiento para investigar y evaluar el objetivo para tratar de obtener información relacionada al sistema víctima como direcciones IP, nombres de host, datos para autenticación, entre otros. Las herramientas más utilizadas por los atacantes son: el escáner y mapeadores de red o puertos y escáneres de vulnerabilidades. (Mieres, 2009)

Fase 3. Obtener acceso. En esta fase se materializa el ataque por medio de la explotación de las vulnerabilidades y los defectos de seguridad del sistema que se hayan descubierto en las fases de reconocimiento y exploración. Entre las técnicas más conocidas que los atacantes pueden utilizar tenemos: Ataques de Buffer Overflow, Denegación de Servicios simple o distribuido (DoS, DDos), Filtrado de contraseñas y Secuestro de sesiones. (Mieres, 2009)

Fase 4. Mantener el acceso. Si los ataques de la fase anterior obtienen acceso a los sistemas, el atacante buscará instalar herramientas que le permitan volver a acceder en el futuro desde cualquier lugar donde tenga acceso a Internet por medio de utilitarios como backdoors, rootkits y troyanos. (Mieres, 2009)

Fase 5: Borrar las huellas. Finalmente, como en toda acción maliciosa el atacante buscará eliminar todo rastro o huellas que fue dejando durante la intrusión para evitar ser detectado por el profesional de seguridad o los administradores de la red, eliminando los archivos de registro o alarmas del Sistema de Detección de Intrusos. (Mieres, 2009)

2.2.2. Ataques conocidos a protocolos de autenticación

Hay varios tipos diferentes de ataques automatizados que los atacantes pueden utilizar para intentar comprometer las cuentas de los usuarios. Los tipos más comunes se enumeran a continuación:

Tipo de ataque	Descripción
Fuerza bruta	Prueba de varias contraseñas de un diccionario u otra fuente en una sola cuenta.
Relleno de credenciales	Prueba de pares de nombre de usuario / contraseña obtenidos de la infracción de otro sitio.
Pulverización de contraseña	Prueba de una única contraseña débil contra una gran cantidad de cuentas diferentes.

Ilustración 6 - Tipos de Ataques Automatizados
Fuente: https://cheatsheetseries.OWASP.org/cheatsheets/Authentication_Cheat_Sheet.html

2.2.3. Vulnerabilidades de los protocolos de Autenticación

- Según el Top Ten OWASP 2021 (OWASP Foundation, 2021). las debilidades de autenticación más comunes son:
- Permitir ataques automatizados para ingresar credenciales, a través de diccionarios donde el atacante tiene una lista de nombres de usuario y contraseñas válidos.
- Permitir el ingreso y uso de contraseñas predeterminadas, débiles o conocidas.
- Utilizar procesos de recuperación de credenciales débiles o ineficaces y de olvido de contraseñas, como "respuestas basadas en el conocimiento", que no pueden protegerse.
- Utilizar repositorios de datos de contraseñas de texto sin formato, cifradas o con un hash débil.
- Falta o falencias de mecanismos de múltiple factor de autenticación.
- Exponer el identificador de sesión en la URL.
- Reutilizar el identificador de sesión después de iniciar sesión correctamente.
- No invalidar correctamente los ID de sesión. Las sesiones de usuario o los tokens de autenticación no se invalidan correctamente durante el cierre de sesión o un período de inactividad.

2.2.4. Medidas de Prevención y protección

Para prevenir los ataques a los protocolos de autenticación como fuerza bruta, relleno de credenciales y propagación de contraseñas se pueden considerar diferentes mecanismos para protegerse contra estos ataques. Para tener una protección más robusta, generalmente es necesario implementar combinaciones de estas haciendo relación al enfoque de defensa en profundidad, aplicando varias capas de protección que permita lograr un alto nivel de seguridad.

(OWASP Foundation, 2021) recomienda implementar las siguientes recomendaciones para minimizar los problemas de autenticación:

- Autenticación multifactor para evitar el relleno automatizado de credenciales, fuerza bruta y los ataques de reutilización de credenciales robadas.
- Renombrar y cambiar las credenciales predeterminadas, especialmente para usuarios con máximos privilegios.
- Comprobar la complejidad de las contraseñas para evitar el uso de credenciales de fácil deducción.
- Implementar políticas de complejidad, longitud, expiración y bloqueo de la contraseña por intentos erróneos considerando un umbral y duración de bloqueo y definir el tiempo que pueden ocurrir.
- Evitar ataques de enumeración aplicando un adecuado registro de errores para aspectos como recuperación de credenciales.
- Control de intentos fallidos excesivos, implementando bloqueos con periodos de tiempo incrementales según la cantidad de eventos.
- Tener cuidado con el diseño de la autenticación, evitando que el identificador de sesión sea visible en la URL de petición.
- Almacenarse las credenciales de forma segura
- Invalidarse las sesiones después del cierre de sesión, periodos de inactividad y tiempos de espera absolutos.
- Implementación de preguntas de seguridad y/o Captcha para ayudar a prevenir ataques automatizados.

- Habilitar el registro y la supervisión de eventos de autenticación para detectar anomalías en tiempo real, asegurando que todas las fallas se registren.

2.3. Historia y evolución del protocolo de autenticación OPENID Connect

Vivimos tiempos en que la tecnología se está volviendo parte de la vida de las personas, hoy en día escuchamos conceptos como “Redes Sociales”, “Internet de las cosas”, “BigData”, “Nube”, al final todos son diferentes sistemas de información implementados como servicios.

Estos servicios utilizan el internet para que los usuarios u otros servicios puedan acceder, lo que ha provocado una gran demanda de consumo en las aplicaciones. Cada servicio expuesto en internet está obligado a identificar y autorizar el acceso, los motivos son diversos, se pueden mencionar las estadísticas del uso, cumplimiento de regulaciones, entre otros, por lo tanto, la multiplicidad de mecanismos de autenticación de usuarios pone en riesgo los datos personales, y dio origen a la implementación de estándares y protocolos de autenticación y autorización en internet.

Para contextualizar de otra manera, los servicios requieren que el usuario se identifique previo a acceder al servicio demostrando de alguna manera que es quien dice ser, la práctica más común es hacerlo mediante un identificador único y un secreto o contraseña, pero también se requiere almacenar datos personales como dirección de correo, fecha de nacimiento, dirección domiciliaria, fotografías, entre otros. Por lo general, cada aplicación los gestiona de manera independiente, provocando que los usuarios tengan la necesidad de recordar múltiples identificadores y contraseñas, lo que lleva a recurrir a prácticas inseguras para recordarlos como valores triviales para contraseñas, anotarlos en lugares que cualquier persona de su entorno puede visualizar, o utilizar los mismos datos en múltiples sistemas.

La problemática dio origen al uso de mecanismos de autenticación que se denominaron como “Inicio de Sesión único”, Según (Kukic, 2020), el inicio de sesión único (SSO por sus siglas en inglés) es un tipo de autenticación en el que un usuario inicia sesión en un sistema y se le otorga automáticamente acceso a otros servicios proporcionando una experiencia perfecta para los usuarios cuando utilizan sus aplicaciones y servicios.

Para continuar contextualizando el origen de OPENID Connect, es necesario tener clara las diferentes entre servicios de **autenticación** y **autorización.**

(Fernandez, 2020) Menciona que la **autenticación** verifica la identidad del usuario que quiere acceder a un recurso mientras que la **autorización** valida si efectivamente el usuario tiene el permiso para acceder al mismo o realizar alguna función. De igual manera define a la autenticación como proceso de identificar a los usuarios y garantizar que los mismos sean quien dice ser, y la autorización es lo que define a qué recursos de sistema, el usuario puede acceder.

Los protocolos o tecnologías de autenticación y autorización unificadas como OPEN ID Connect están diseñadas para mejorar la experiencia del usuario al acceder a las aplicaciones de software, evitando que tengan que recordar múltiples contraseñas mediante un sistema de inicio de sesión único (SSO) y almacenar sus datos personales en diferentes repositorios autorizando que las aplicaciones accedan a ellos en uno común, donde estén actualizados.

2.3.1. Reseña Histórica

OpenID. (OpenID Foundation, 2020) OpenID fue creado en el verano de 2005 por una comunidad de código abierto que intentaba resolver un problema que no se corregía fácilmente con otras tecnologías de identidad existentes. Como tal, OpenID está descentralizado y no es propiedad de nadie. Hoy en día, cualquiera puede optar por utilizar un OpenID o convertirse en un proveedor de OpenID de forma gratuita sin tener que registrarse ni ser aprobado por ninguna organización. Fue desarrollado por Brad Fitzpatrick (Fitzpatrick, LiveJournal, 2005), fundador de LiveJournal, inicialmente conocido como Yadis y fue nombrado OPENID luego de que le asignen el dominio openid.net para usarlo en el proyecto por parte de la empresa SIX APART para la cual trabajaba, fue implementado en el portal LiveJournal y sus publicaciones en el blog DeadJournal llamaron rápidamente la atención de la comunidad digital. (Fitzpatrick, Danga, 2005). Según Fitzpatrick, JanRain fue uno de los primeros desarrolladores en apoyar esta iniciativa, proporcionando librerías de software, e implementando en sus portales OPENID (Fitzpatrick, LiveJournal, 2005).

En junio del mismo año, usuarios de OPENID y los desarrolladores de la empresa NETMESH empezaron discusiones en diferentes foros sobre la interoperabilidad de OPENID y LID (Light Weight Identity de NETMESH dando como resultado la creación del protocolo YADIS, nombre utilizado originalmente para OPENID, siendo anunciado el 24 de octubre del 2005 por Fitzpatrick y Johannes Ernst. (Recordon, David RecordonDavid Recordon, 2005). Posteriormente XRI se une al proyecto asignado desarrolladores y contribuyendo con su secuencia de descriptores de recursos extensibles (XRDS) (Reed, 2005). (Reed, Equals Drummond, 2008)

En diciembre del mismo año se une SXip Identity a la comunidad OPENID/Yadis, otra compañía de desarrollo de gestión de identidades, luego anunciar la versión 2.0 de sus protocolos de identidad extensible SXIP basada en solicitudes a URL como LID y OPENID. (SXIP, 2005).

Según (Hoyt, 2006), A inicios del año 2006, JanRain modifica las librerías de Python y Ruby, creando una extensión de Registro simple que permite el intercambio de perfiles, siendo un mecanismo muy enfocado que proporciona información que comúnmente se necesita para registrarse con un Sitio web, luego se publicarían extensiones en otros lenguajes para OpenID, paralelamente se había iniciado el desarrollo para anexar el soporte completo de XRI (Grey, 2006) cuyo objetivo era modificar los pasos de descubrimiento del servidor de identidad. La diferencia es que OpenID estándar presenta una URL como un identificador y la URL se obtiene leyendo la información en el encabezado HTTP que devuelve la URL de identidad y el perfil XRI presenta los parámetros i-name o i-number y la URL se obtiene mediante resolución XRI.

En mayo del 2006, Brad (Fitzpatrick, 2006), publica que uno de los desarrolladores clave del proyecto, David Recordon, se une a VeriSign, enfocando su trabajo en la identidad digital orientándolo a la especificación de Open ID, siendo un paso importante en el diseño de seguridad del protocolo.

David (Recordon, 2006) en junio del mismo año publica lo siguiente: "*Vemos OpenID como un paraguas para el marco que abarca las capas para identificadores, descubrimiento, autenticación y una capa de servicios de mensajería que se*

encuentra en la parte superior y todo esto se ha denominado "OpenID 2.0". Es decir, que a partir de esta versión se permitiría el intercambio de perfiles usando datos únicos como el correo electrónico directamente entre un proveedor de Identidad (IDP) y un consumidor, o desde un IDP a otro IDP. A finales del año, (Becker, 2006) publica en el portal ZDNet, el caso de OpenID a usuarios, empresarios y operadores de sitios web detallando y explicando los beneficios que estos tendrían al adoptar este protocolo en sus aplicaciones de software de internet.

Esto llama la atención de grandes empresas, siendo hitos importantes el anuncio de Symantec (Garretson, 2007) con la iniciativa “Symantec Identity” que incluye servicios y software que aprovechan la línea de productos Norton para ayudar a los consumidores a administrar sus identidades en línea y facilitar transacciones seguras a través de Internet a través de los protocolos de intercambio de identidades como OpenID y Cardspace de Microsoft.

En febrero del mismo año, (Verisign, 2007) publica en su blog, el acuerdo entre JanRain, Microsoft, Sxip y VeriSign para la colaboración en la interoperabilidad entre OpenID y Windows CardSpace (TM) para hacer que Internet sea más seguro y fácil de usar. Este acuerdo consiste básicamente en:

a. OpenID se ampliará para permitir que las partes de confianza soliciten explícitamente y se les informe sobre el uso de credenciales resistentes al phishing.
b. Microsoft reconoce el crecimiento de la comunidad OpenID y cree que OpenID juega un papel importante en la infraestructura de identidad de Internet.
c. JanRain, Sxip y VeriSign reconocerían que las tarjetas de información brindan a los usuarios importantes beneficios contra el phishing, la privacidad y la comodidad
d. JanRain y Sxip, proveedores líderes de bibliotecas de código fuente abierto para blogs y sitios web.
e. JanRain, Sxip y VeriSign planean agregar soporte de tarjeta de información a futuras soluciones de identidad.
f. Microsoft planea admitir OpenID en futuros productos de servidor de identidad.

El siguiente en anunciar la adopción de OpenID es (AOL, 2007) en su blog corporativo confirmando:

g. Cada usuario de AOL / AIM ahora tiene al menos un URI de OpenID, http://openid.aol.com/ <sn>.
h. Este servicio de proveedor experimental OpenID 1.1 ya está disponible y estamos realizando pruebas de compatibilidad.
i. Estamos trabajando con partes de confianza de OpenID para resolver problemas de compatibilidad.
j. Nuestra plataforma de blogs ha habilitado OpenID 1.1 básico en beta, por lo que cada URI de blog beta también es un identificador OpenID básico. (Aún no hay Yadis).
k. Todavía no aceptamos identidades OpenID en nuestros productos como parte de confianza, pero estamos trabajando activamente en ello. Es probable que ese despliegue sea gradual.
l. Estamos rastreando el esfuerzo de estandarización de OpenID 2.0 y planeamos admitirlo después de que sea definitivo.

En junio del 2007, se crea la (OpenID Foundation, 2007) una corporación de beneficio público con sede en Oregon para administrar la marca y la propiedad OpenID. Luego, a finales de este año se confirman las especificaciones y la liberación de OpenId Authentication 2.0 y OpenID Attribute Exchange 1.0 (Recordon, 2007) mencionando a los participantes de la fundación que aportaron en su desarrollo como AOL, Cordance, JanRain, Microsoft, NetMesh, Six Apart, Sxip, Sun Microsystems, Symantec, Verisign y Yahoo. Recalcando el importante papel desempeñado por Microsoft en ayudar con el apoyo legal y la orientación combinados con el conocimiento de Sun y Yahoo con su trabajo conjunto en el desarrollo del lenguaje.

Para el año 2008, (Yahoo! C, 2008) anuncia su apoyo al marco de identidad digital OpenID 2.0 para los millones de usuarios en todo el mundo, permitiendo la consolidación de su identidad en internet a través de sus servicios, eliminando la necesidad de crear identificadores por separado en todos sus portales web, blogs, noticias y confirmando la posibilidad de usarlo en cualquier sitio compatible con OpenID 2.0.

En mayo, (SourceForge, Inc, 2008), líder en medios de comunicación y comercio electrónico impulsados por la comunidad, anunció la inclusión de la funcionalidad OpenID en su sitio web SourceForge.net, en el mismo año (MySpace, 2008), Google y Microsoft integran la funcionalidad en sus plataformas (OpenID, 2008) siendo un hito muy importante ya que se marca el inicio de un cambio en la gestión de identidades de internet.

El 14 de Noviembre de 2008, JanRain anuncia una versión de código abierto de OpenID denominada RPX Basic que permitiría a los consumidores y empresas en general implementarlo de forma rápida y sencilla.

Para el 2009, se anuncia la integración de OpenID en Facebook, (Shepard, 2009), y sus requerimientos (Facebook, 2009) que permitieron a los usuarios registrarse con las credenciales de las cuenta de Gmail como lo podemos observar en la siguiente imagen

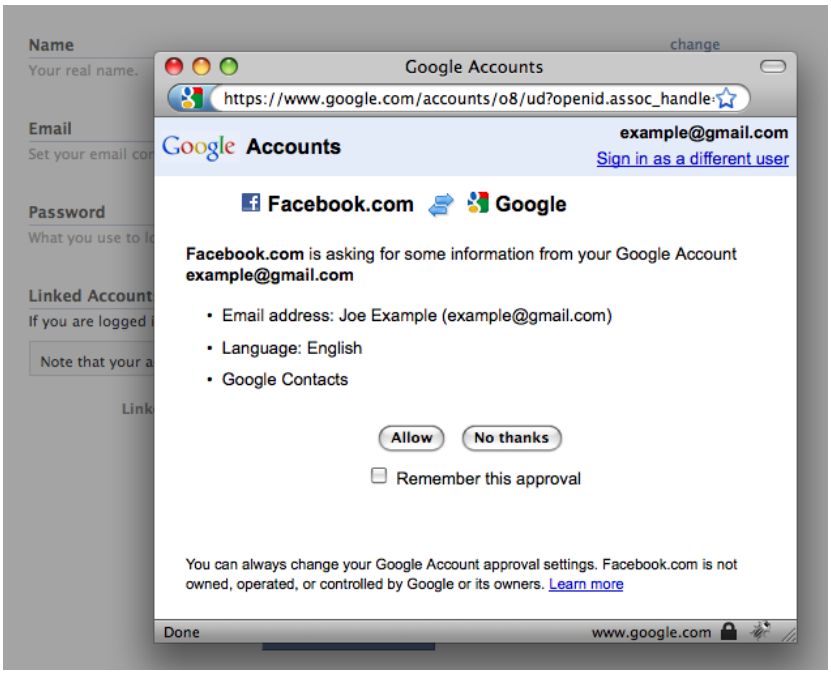

Ilustración 7 - Autenticación en Facebook mediante OpenID de Google
Fuente:
https://web.archive.org/web/20090701093324im_/http://wiki.developers.facebook.com/images/8/89/OpenIDex.png

Para el año 2013, una publicación (JanRain, 2013) oficializa el cierre del servidor gratuito de OpenID Identity denominado MyOpenID, creado por JainRain debido a la reducción de sitios compatibles con OpenID apuntando la falta de información, la complejidad de la implementación y a las grandes industrias del internet por su falta de apoyo y confirmando que la guerra de la identidad estaba siendo liderada por Facebook quien a partir de ese momento dejo de patrocinar a OpenID.

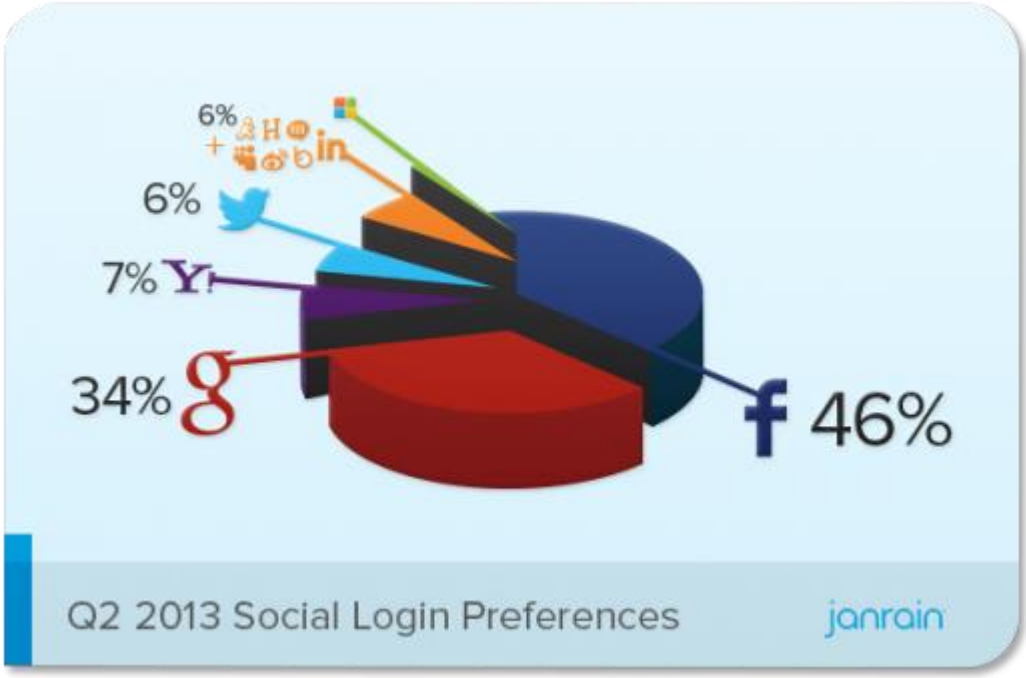

Ilustración 8 - Preferencias de Gestión de Identidad.
Fuente: https://cdn0.tnwcdn.com/wp-content/blogs.dir/1/files/2013/09/Q2-2013-Social-Login-Preferences-520x346.png

En 2014, OpenID Foundation (OpenID, 2014) publica la tercera generación de OpenID, denominada OpenId Connect que integra el flujo de autorización OAUTH 2.0 permitiendo a los clientes informáticos verificar la identidad de un usuario final basándose en la autenticación realizada por un servidor de autorización y obtener información básica del perfil utilizando el estándar de intercambio datos de Javascript (JSON). A partir de este anunció Las organizaciones y las empresas tienen la oportunidad de utilizar OpenID Connect para el desarrollo de ecosistemas de identidad seguros, flexibles e interoperables, para que las identidades digitales se puedan usar fácilmente en sitios web y aplicaciones a través de cualquier computadora o dispositivo móvil.

Para el 2015, (Thibeau, 2015), se lanza el programa de certificación para OpenIdConnect en la conferencia RSA, este programa permite a las empresas certificar que sus implementaciones están correctamente alineadas a las especificaciones, siendo empresas como Microsoft, Google, ForgeRock, entre otras las primeras obtener las cerficaciones de OpenId Connect.

En 2015 se aprueba la especificación final del modo de respuestas posterior al formulario en formato OAUTH 2.0 (Jones, OpenID, 2015), La misma que proporciona protecciones de propiedad intelectual a los implementadores.

En 2018, (StackExchange, 2018) Stack Overflow anuncia el fin del soporte para OpenID por la preferencia de los usuarios al uso de otras fuentes de gestión de identidad como Google, Facebook. En la siguiente imagen podemos visualizar las estadísticas del uso de este estándar demostrando sus decrecimientos a partir del 2014.

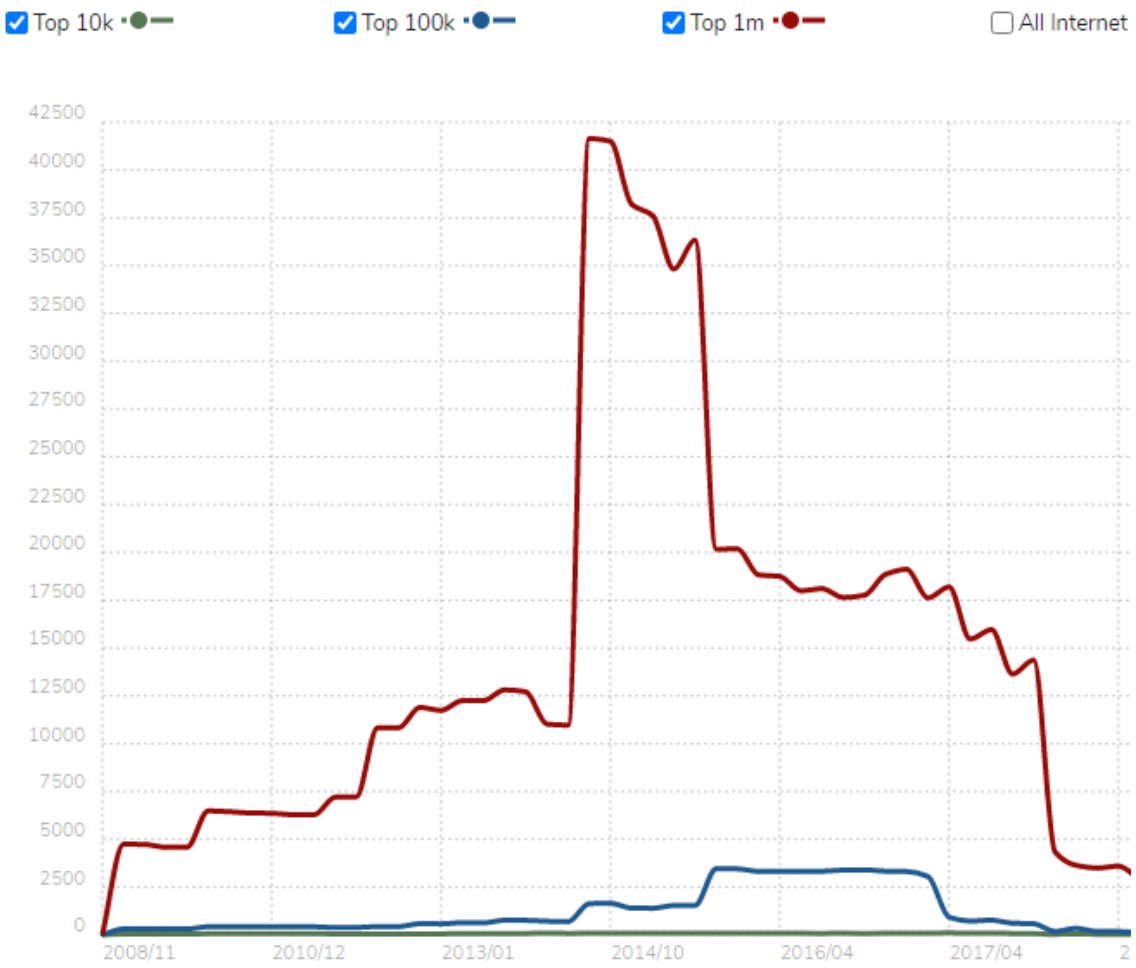

Ilustración 9 - Estadísticas de OpenID
Fuente: https://trends.builtwith.com/docinfo/OpenID

Para inicios del 2020, la Fundación OpenID aprueba el borrador del implementador de la especificación de la federación de OpenID Connect (Jones, OpenId, 2020). El Borrador de un implementador es una versión estable de una especificación que brinda protección de propiedad intelectual a los implementadores de la especificación. Este es el segundo borrador del implementador de esta especificación. Esta especificación es un producto del grupo de trabajo OpenID Connect.

En la siguiente imagen se resume, la reseña histórica con los hitos más importantes del desarrollo del protocolo OpenID, hasta la creación de OPENID Connect como la conocemos ahora.

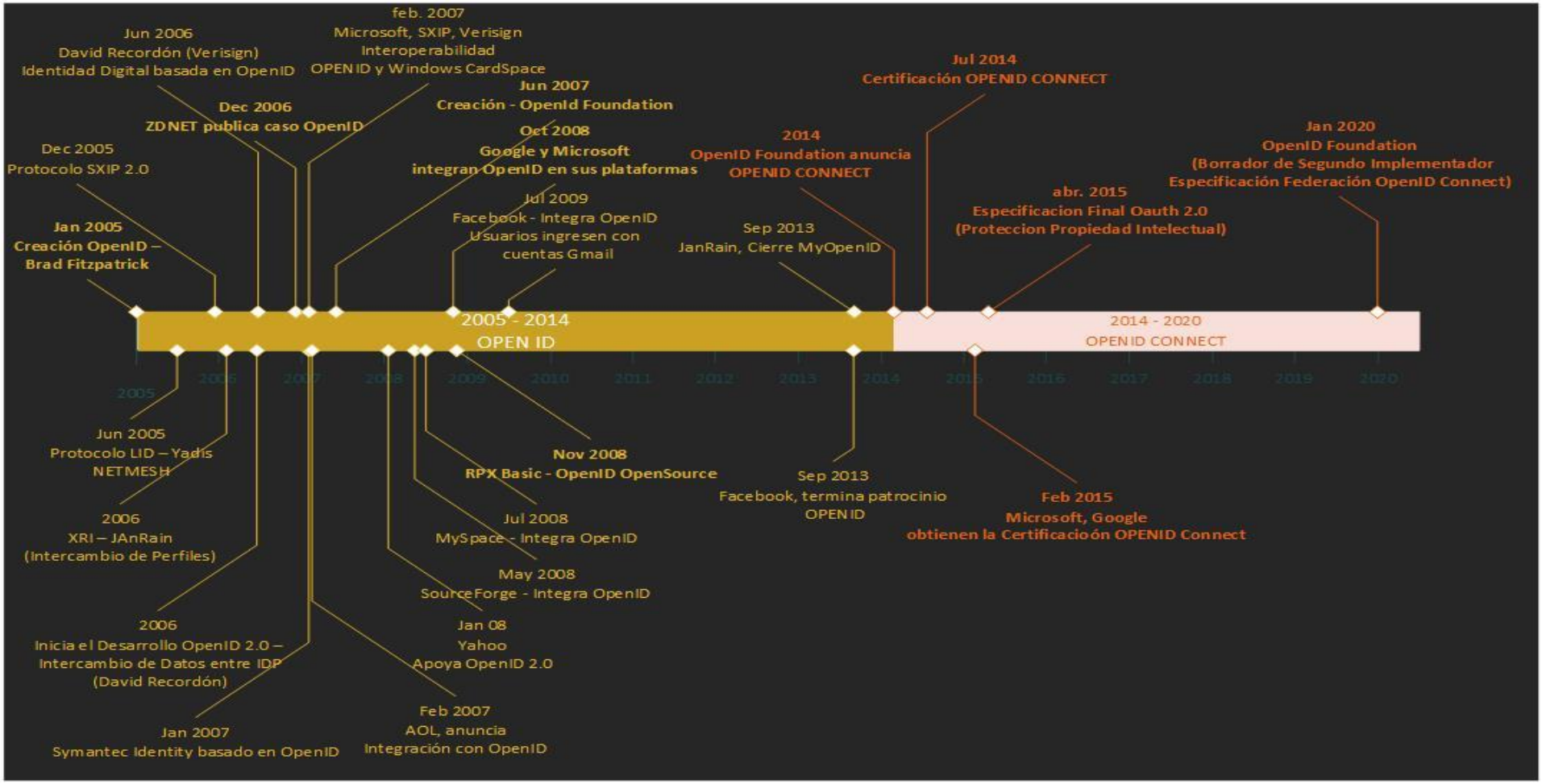

Ilustración 10 - Reseña histórica OpenID, OpenID Connect
Fuente: Elaboración Propia

3. OBJETIVOS Y METODOLOGÍA DE TRABAJO

3.1. Objetivo General

El objetivo general del presente TFM consiste en **aportar con la mejora en la implementación de los esquemas de autenticación** permitiendo aumentar el nivel de seguridad de las aplicaciones y la integración con otros sistemas de información mediante la ejecución de los objetivos específicos que pretenden analizar de manera detallada los protocolos, evaluar el nivel de exposición o riesgo y presentar las consideraciones necesarias para asegurar las aplicaciones.

3.2. Objetivos Específicos

Se han definido los siguientes objetivos específicos que permitan analizar minuciosamente el estado del arte para entender la situación problemática y establecer recomendaciones para su uso en las organizaciones

Los objetivos específicos en el presente TFM son:

1. Realizar el estudio del arte sobre el protocolo de autenticación OPENID Connect analizando sus componentes, estructura, características e historia.
2. Identificar y evaluar las técnicas de ataque a los protocolos OPENID Connect
3. Identificar y evaluar el riesgo y exposición a las que pueden estar sujetos los protocolos estudiados.
4. Identificar y analizar herramientas para realizar test de penetración al protocolo de autenticación.
5. Implementar un ambiente para realizar el test de penetración y poder evaluarlos sin perjuicio a terceros.
6. Recomendar consideraciones para la implementación segura del protocolo de autenticación.

3.3. Metodología de Trabajo

La metodología para este TFM consiste en abarcar el objetivo general mediante el análisis de información o estudio del estado del arte y la implementación de un ambiente controlado que permita realizar pruebas sin afectar a terceros.

A. Estudio del Estado del Arte

Para el presente TFM se desarrollará las siguientes etapas

1. Desarrollar la estructura del documento.
2. Organización de fuentes de información.
3. Búsqueda de información relevante y concerniente al tema.
4. Lectura y comprensión de la información
5. Redacción del documento.

Al finalizar las tareas anteriores, se realiza el diseño del entorno de pruebas para la evaluación mediante ataques o pruebas de penetración al protocolo de autenticación OPENID Connect.

B. Implementación del entorno de pruebas

Para el ambiente de pruebas se va a requerir de un entorno de servidores virtuales para implementar una aplicación demo y dos servicios web que se comuniquen mediante una API

Se utilizará:

m. AZURE AD B2C como proveedor de OPENID Connect
n. BURP como proxy para capturar las peticiones y poder analizarlas
o. Se utilizará una aplicación de Login de demo

C. Herramientas de soporte

AZURE AD B2C

Es una herramienta de gestión de identidades de Microsoft disponible en el portal POWERAPPS basada en código abierto que permite mejorar la seguridad de las empresas y gestionar los roles mediante interfaces que controlan el acceso a las aplicaciones incluyendo las integraciones como API, Webservices, entre otras.

OWASP ZAP (Zed Attack Proxy)

Es una herramienta utilizada para realizar evaluaciones de seguridad mediante diferentes tipos de pruebas y metodologías permitiendo realizar análisis profundos y tener control total de las interacciones y respuestas de aplicaciones con el objetivo de detectar fallos o vulnerabilidades que los atacantes podrían explotar.

D. Metodología de Realización de las Pruebas

Para realizar el experimento, se realizará un ciclo de pruebas por cada protocolo OPENID Connect siguiendo las siguientes etapas:

1. Probar la robustez de las suites criptográficas soportadas en OPENID Connect
2. Probar la robustez de los flujos de autorización de OPENID Connect

4. DESARROLLO ESPECÍFICO DE LA CONTRIBUCIÓN

El capítulo representa la finalización del trabajo realizado durante el TFM, que corresponde al análisis de información relacionada con el protocolo OpenID Connect, para profundizar en su entendimiento y poder determinar su grado de seguridad y confianza que brinda a los esquemas de autenticación actuales.

Primero se analiza el protocolo y sus características principales, posteriormente se presenta el piloto experimental detallando paso a paso las pruebas realizadas y sus resultados, finalmente se presenta el análisis de riesgos siguiendo la metodología OWASP.

4.1. Introducción al Protocolo y sus principales características.

4.1.1. Protocolo OpenID Connect

El protocolo OpenID Connect es una estándar de identificación implementada sobre OAUTH 2.0, y utilizada para comprobar la identidad del usuario final mediante un servidor de autenticación externo a la aplicación, también denominado proveedor de Identidad, que además proporciona información básica de una forma muy similar al protocolo REST. Es utilizado mayormente para entornos web, móviles y javascript para obtener información de las sesiones y de los usuarios finales. Sus especificaciones se consideran extensibles, permitiendo el manejo de funciones adicionales como el descubrimiento de proveedores de autenticación OpenID, gestores de sesiones e inclusive cifrar datos de identificación de usuarios para robustecer el esquema de seguridad.

Según (Biehl, 2019) OpenId Connect (OIDC) es un protocolo para proporcionar identidad como servicio. Es un protocolo técnico que permite que los datos del usuario final se pasen de forma segura de un proveedor a un cliente, con el consentimiento explícito del usuario.

Entre las causas de mayor consideración para adoptar el uso de este protocolo es que está basado en estándares de uso libre lo que proporciona grandes posibilidades de adaptación a otras plataformas, por otro lado, se debe considerar desde el punto de

vista de desarrollo que es más fácil integrarse a un servicio que crearlo y diseñarlo, lo cual permite optimizar el tiempo en proyectos de desarrollo de aplicaciones, especialmente en entornos empresariales, y esto tiene sentido, ya que un programador se despreocupa de implementar desde el proceso de registro de usuarios hasta el almacenamiento seguro de contraseñas y los datos del usuario.

Desde el punto de vista de la administración y operación del sistema se evita estar duplicando y manteniendo la información de la identidad del usuario en diferentes bases de datos, es decir que la integración de este servicio permite agilizar el proceso de autenticación, pero lo interesante es que su diseño y arquitectura incluye aspectos de seguridad para la validación de la identidad del usuario segregando perfiles y permitiendo que el usuario final exprese el consentimiento para el uso de las aplicaciones.

Otro aspecto importante a tener en cuenta para la adopción de OPENID Connect es el control que se proporciona a los usuarios finales sobre la identidad digital, por estar registrado en un repositorio común, es decir que el registro es realizado una sola vez. Esta identidad puede ser reutilizada en múltiples sitios o aplicaciones web como se requiera y el usuario lo permita. Lo interesante es que el mismo usuario tiene la capacidad de interrumpir las sesiones iniciadas en múltiples aplicaciones y sitios web de manera simultánea.

Desde el punto de vista de la seguridad, la reducción de riesgos es inminente ya que se evita el traspaso de contraseña como parámetro por los diferentes aplicativos reduciendo su exposición ya que lo que se transmite por los diferentes sitios es un token de acceso para validar la identidad del usuario final, es decir que este protocolo las contraseña no será expuesta en las transacciones de autenticación.

De igual manera la adopción de este protocolo permite y facilita el uso de múltiples algoritmos de autenticación incluyendo tecnologías de doble factor como mensajes de texto, correos electrónicos, claves dinámicas, huellas dactilares, reconocimiento facial, etc.

Según (OpenID Foundation, 2021) en su sitio web, se define al protocolo OpenID Connect 1.0 como *una capa de identidad simple sobre el protocolo OAuth 2.0 que*

permite a los Clientes verificar la identidad del Usuario Final basándose en la autenticación realizada por un Servidor de Autorización, así como obtener información de perfil básica sobre el Usuario Final de una manera interoperable y similar a REST.

4.1.2. Características técnicas de OPENID Connect

Previo a realizar una explicación de las características de este protocolo, es necesario tener claros los conceptos sobre Autenticación y Autorización. Para empezar la autenticación se refiere al proceso de identificar y confirmar la identidad de una persona o servicio mediante un parámetro único que identifique al cliente y un valor único que el cliente conoce, posee o una mezcla de los dos, mientras que la autorización se encarga de determinar los privilegios o permisos autorizados para dicha entidad.

Se debe tener claro que el uso de un esquema de autenticación y autorización no es obligatorio, tanto es así que la mayoría de los sitios web en internet no requieren autenticación para su acceso, como portales informativos, buscadores, etc.; en cambio, aplicaciones y sitios transaccionales o que manejen información confidencial, prácticamente están obligados a implementarlo. Esto conlleva a la necesidad de proteger los datos confidenciales para autenticar y autorizar mediante servicios criptográficos que cada vez son más robustos y esto significa que dichos datos deben ser convertidos de manera tal que se garantice su privacidad siendo los protocolos SSL y SSH los más utilizados.

OpenId Connect integra en su diseño el proceso de autenticación, autorización y puede integrarse con un sin número de aplicaciones proporcionando **Tokens de identidad de fácil consumo,** esto significa que las aplicaciones y/o servicios clientes pueden recibir los datos de la identidad del usuario de forma codificada mediante un token web en formato JSON (JWT) seguro denominado token de identificación. Estos archivos JWT están estructurados para ser portátiles y admitir varios tipos algoritmos de firma y cifrado.

Al utilizar el protocolo **OAuth** las aplicaciones y/o servicios que usan OpenId Connect utilizan sus flujos para obtener los tokens de identificación, tanto para aplicaciones

web, móviles como para APIS de integración. En otras palabras, significa que tendrá un protocolo único para autenticarse basado en la obtención de token de acceso, razón por la cual este protocolo se está convirtiendo en un estándar para que las aplicaciones logren autenticarse mediante proveedores externos, como por ejemplo Facebook, google u Office365 de Microsoft debido a su sencillez y rendimiento: ya que es lo suficientemente simple como para integrarse con aplicaciones, al mismo tiempo de ofrecer funciones y opciones de seguridad que pueden cumplir con los exigentes requisitos de las organizaciones.

OpenID Connect se ha vuelto un estándar cuando las aplicaciones requieren identificar a sus usuarios evitando el desarrollo de esquema de autenticación basado en una base de datos de la información personal utilizados como credenciales para acceder a los servicios o aplicaciones. Sin embargo, las personas encuentran tedioso el registro y la creación de cuentas por el hecho de tener que proteger múltiples datos por cada aplicación obligando a caer en malas prácticas como anotaciones en notas o simplemente en papel. En aplicaciones comerciales, esto puede representar perdidas de ventas en línea, por el hecho de la fatiga al momento de autenticarse que pueden sufrir los consumidores, y no solo eso, sino la responsabilidad en la seguridad que se debe aplicar a los datos para evitar que caigan en manos de terceros.

La solución provista por este protocolo a estos problemas es la delegación de la autenticación y el aprovisionamiento del usuario a un servicio dedicado especialmente diseñado, llamado proveedor de identidad (IdP) cuyas principales características son:

- Uso de Tokens de identidad.
- Basado en el protocolo OAuth 2.0
- Simplicidad.

4.1.3. Flujos de Código

Según la guía de implementación (OpenID, 2020) el flujo se resume en los siguientes pasos:

1. El RP (Cliente) envía una solicitud al Proveedor OpenID (OP).
2. El OP autentica al usuario final y obtiene la autorización.

3. El OP responde con un token de identificación y generalmente un token de acceso.
4. El RP puede enviar una solicitud con el token de acceso al punto final de UserInfo.
5. UserInfo Endpoint devuelve reclamaciones sobre el usuario final.

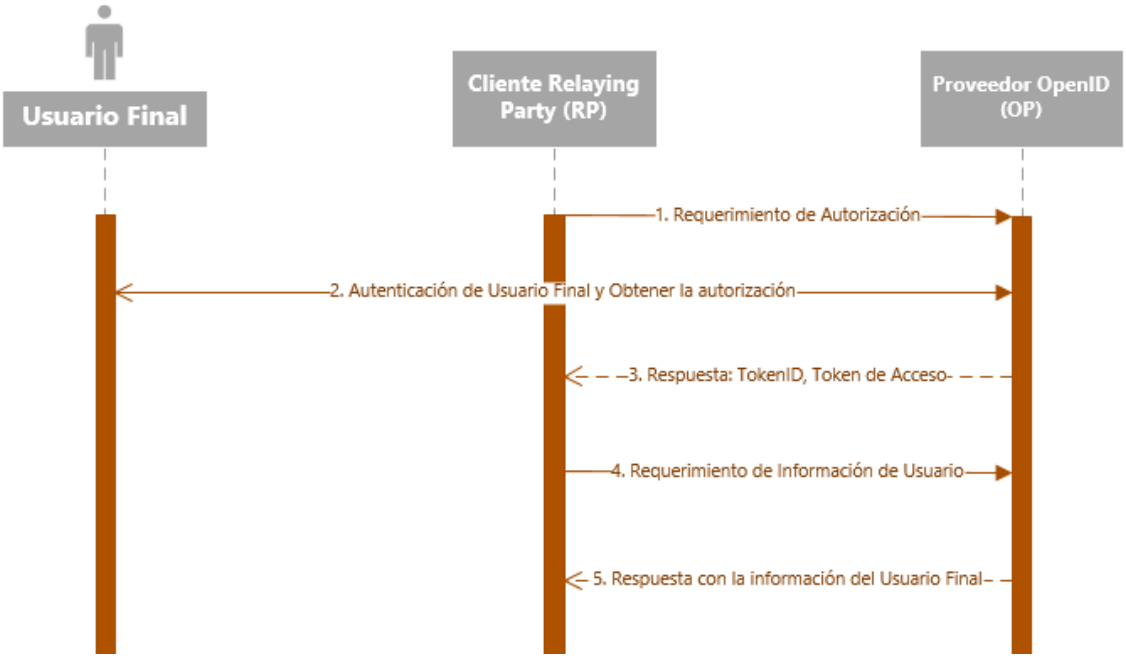

Ilustración 11 - Flujo Abreviado Open ID Connect

Fuente: Elaboración Propia

Los flujos válidos para OpenID Connect son:

Flujo de código de autorización: Utilizado generalmente por aplicaciones web y móviles redireccionando el navegador desde o hacia el OP para la autenticación y el consentimiento del usuario, luego mediante una segunda solicitud denominada de canal de retorno, recupera el token ID ofreciendo una seguridad robusta y optimizada, sin revelar los tokens al navegador, permitiendo que el cliente también pueda autenticarse.

El flujo de código consta de los siguientes pasos:

1. Un usuario accede a una aplicación de la RP.
2. La RP prepara una solicitud de autenticación y redirige al usuario al OP.

3. El OP autentica al usuario, solicitándole las credenciales. El usuario autoriza a la RP a acceder a la información necesaria para la aplicación. El OP genera un código de autorización de un solo uso para la RP.
4. El OP redirige al usuario a la RP con el código de autorización.
5. La RP llama al punto final de señal del OP para intercambiar el código de autorización por una señal de acceso, señal de ID y una señal de renovación.
6. La RP utiliza la señal de ID para autorizar al usuario final.

En el siguiente gráfico se muestra el flujo de autorización:

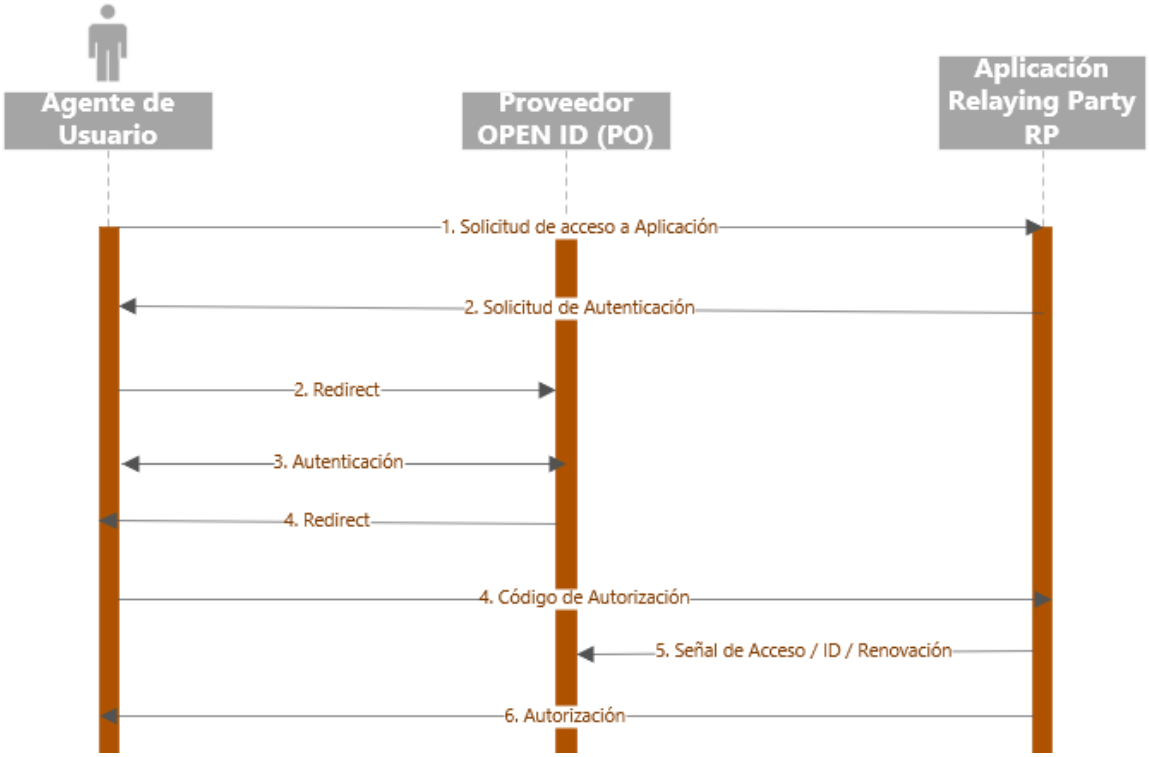

Ilustración 12 - Flujo de Autorización de OpenId Connect
Fuente: Realización Propia

Flujo implícito: Es utilizado para aplicaciones basadas en navegador (JavaScript) que no tienen backend. El token ID se recibe con la respuesta de redirección del OP sin requerir petición de canal de retorno.

1. Un usuario accede a una aplicación de la RP.
2. La RP prepara una solicitud de autenticación y redirige al usuario al OP.

3. El OP autentica al usuario, solicitándole las credenciales. El usuario autoriza a la RP a acceder a la información necesaria para la aplicación.
4. El OP redirige al usuario a la RP con una señal de ID y una señal de acceso.
5. La RP utiliza la señal de ID para autorizar al usuario.

En la siguiente imagen se detalla el flujo de manera gráfica

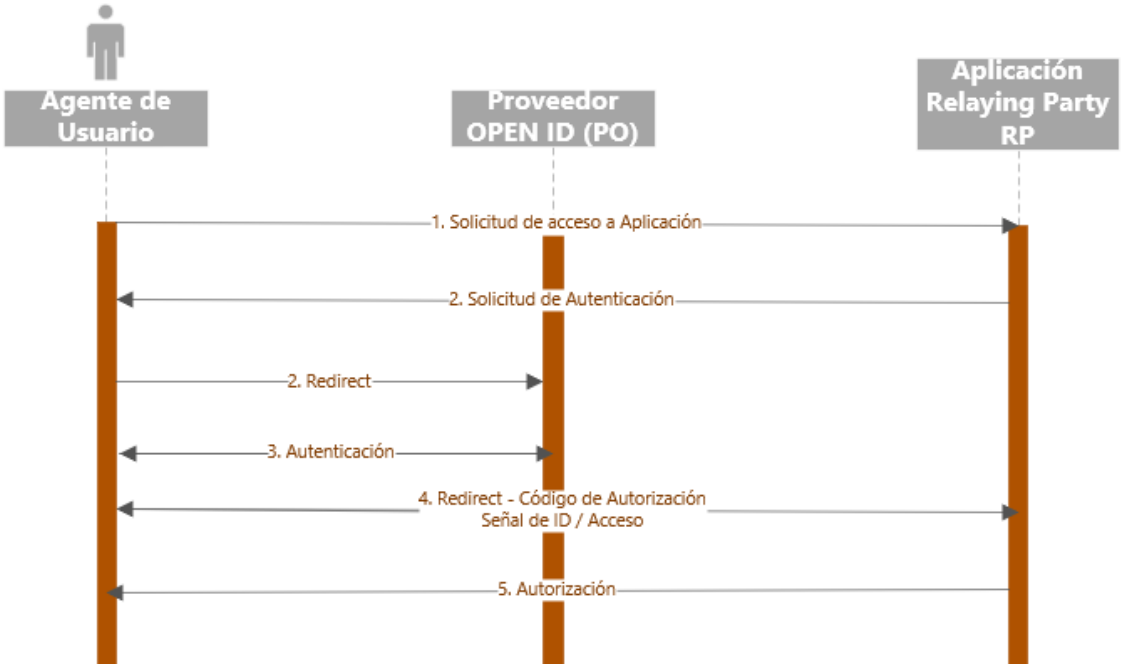

Ilustración 13 - Flujo de Autorización Implícito de Open Id Connect
Fuente: Realización Propia

4.1.4. El token de Identidad

El token de identidad es una tarjeta de identidad digital en un formato JWT (JSON Web Token) estándar firmado por el proveedor de OpenID (OP). Para obtener un token de este tipo el cliente debe enviar al usuario a su OP con una solicitud de autenticación.

Características del token de identificación:

a. Confirma la identidad del usuario.
b. Se especifica la entidad emisora
c. Se genera para una audiencia en particular.

d. Puede contener un código arbitrario único.
e. Se puede definir un tiempo de autorización y como se debe autenticar al usuario.
f. Permite definir el tiempo de vencimiento y organizar los problemas de autenticación.
g. Puede incluir detalles adicionales sobre el usuario final como el correo electrónico, nombre de usuario.
h. Se firma digitalmente, por lo que los destinatarios pueden hacer validaciones adicionales.
i. Se pueden cifrar o codificar.

Ejemplo de Petición en formato JWT:

```
{
  "sub"       : "carlos",
  "iss"       : "https://facebook.com",
  "aud"       : "client-98765",
  "nonce"     : "n-856_WasdfT2Mj",
  "auth_time" : 1311280476,
  "acr"       : "pepito.com",
  "iat"       : 1311278459,
  "exp"       : 1311245580
}
```

La solicitud debe realizarla un cliente que se denomina RP (Relaying Party) y el proceso de autenticación la debe realizar el proveedor de Identidad (IdP) verificando la sesión o credenciales del usuario, en el que se necesitará un agente de confianza que generalmente se utilizan los navegadores web mediante una ventana emergente para tal efecto, en el que la aplicación redirige la petición. La autenticación del usuario siempre debe ocurrir en un contexto confiable que esté separado de la aplicación.

El protocolo no detalla como el proveedor de identidad debe gestionar la autenticación, ya que esta acción es delegada. los tokens de identificación son llamados a través del protocolo OAuth 2.0, que tiene flujos diseñados para todo tipo de aplicaciones: aplicaciones web tradicionales basadas en servidor, aplicaciones solo para navegadores (JavaScript) y aplicaciones nativas / móviles.

La estructura de los datos del TokenID, se constituye en uno de los principales aportes del protocolo OpenID Connect sobre OAuth 2.0 al permitir la autenticación de los usuarios finales. Contiene requerimientos o reclamos (realms) para gestionar el flujo

de información entre el usuario final, el proveedor de identidad y cliente RP representado como JWT (JSON Web Token). En la siguiente tabla se muestra los reclamos o realms posibles para el tokenID

Claims	Tipo	Descripción	Características
iss	Requerido	Identificador de emisor de la respuesta.	URL Sensitiva (Distingue Mayúsculas y Minúsculas). Protocolo: https Contiene: esquema, host, componentes de ruta (opcional) y número de puerto (opcional)
sub	Requerido	Identificador de sujeto.	Un identificador local único que no se puede reasignar que solo puede ser consumido por el Cliente. Longitud: 255 caracteres ASCII (Cadena de caracteres).
aud	Requerido	Audiencia(s). Destinatarios del token	Debe contener el identificador (client_id) de OAuth 2.0 del "Relaying Party" u otras audiencias. Es una matriz de cadenas de caracteres sensible a mayúsculas y minúsculas.
exp	Requerido	Tiempo de caducidad del token	La fecha / hora actual debe ser anterior a la de vencimiento. Formato: JSON que representa el número de segundos desde 1970-01-01T0: 0: 0Z medido en UTC hasta la fecha / hora. Referencia: RFC 3339
iat	Requerido	Hora a la que se emitió el JWT.	Su valor es un número JSON que representa el número de segundos desde 1970-01-01T0: 0: 0Z medido en UTC hasta la fecha / hora.
auth_time	Requerido	Hora en que se produjo la autenticación del usuario final.	Su valor es un número JSON que representa el número de segundos desde 1970-01-01T0: 0: 0Z medido en UTC hasta la fecha / hora. Cuando se realiza una solicitud o reclamo "max_age" o "auth_time"
	Opcional		Si el reclamo "auth_time" corresponde semánticamente con el parámetro de respuesta auth_time de OpenID 2.0 PAPE
nonce		Valor de cadena utilizada para asociar una sesión de cliente con un token de identificación	El valor se pasa sin modificar desde la solicitud de autenticación al token de identificación. Si está presente en el token ID, los clientes verifican que el valor de reclamo sea igual al valor del parámetro enviado en la solicitud de autenticación. Si está presente en la solicitud de autenticación, los servidores de autorización deben incluir

Claims	Tipo	Descripción	Características
		y para mitigar los ataques de reproducción.	un reclamo "nonce" en el token ID. Los servidores de autorización no deberían realizar ningún otro procesamiento en los valores nonce utilizados. El nonce value es una cadena sensible a mayúsculas y minúsculas.
acr	Opcional	Referencia de clase de contexto de autenticación.	Cadena que especifica un valor referencial que identifica la clase de contexto de autenticación realizada. El valor "0" indica que la autenticación del usuario final no cumplió con los requisitos de ISO / IEC 29115 nivel 1. La autenticación mediante una cookie de navegador de larga duración, es un ejemplo en el que el uso de "nivel 0" es apropiado. Las autenticaciones con nivel 0 no deben utilizarse para autorizar el acceso a ningún recurso de valor monetario. Los nombres registrados no deben utilizarse con un significado diferente al registrado. El valor acr es una cadena sensible a mayúsculas y minúsculas.
amr	Opcional	Referencias de métodos de autenticación.	Matriz JSON que corresponde a los identificadores de los métodos de autenticación utilizados. Por ejemplo, los valores pueden indicar que se utilizaron métodos de autenticación de contraseña y OTP. Las partes que utilicen esta afirmación deberán acordar los significados de los valores utilizados, que pueden ser específicos del contexto. El valor amr es una matriz de cadenas que distinguen entre mayúsculas y minúsculas.
azp	Opcional	Authorized party: la parte a la que se emitió el token de identificación.	Si está presente, DEBE contener el ID de cliente OAuth 2.0 de esta parte. Este reclamo solo es necesario cuando el token de identificación tiene un valor de audiencia único y esa audiencia es diferente a la parte autorizada. PUEDE incluirse cuando la parte autorizada sea la misma que la única audiencia. El valor azp es una cadena sensible a mayúsculas y minúsculas que contiene un valor StringOrURI.

Tabla 1 - Reclamaciones o Realms del Token de Identificación
Fuente: https://openid.net/connect/

4.1.4.1. Validación del TokenID

El cliente debe validar el token de identificación en la respuesta de la siguiente manera:

j. El identificador del emisor para el proveedor de OpenID debe coincidir exactamente con el valor del claim “ISS”.
k. El cliente valida que el claim “aud” contiene el valor de “client_id” registrado en el emisor identificado por el claim “ISS” como una audiencia. El token de identificación debe ser rechazado si el token de identificación no incluye al cliente como una audiencia válida, o si contiene públicos adicionales en los que el cliente no confía.
l. Si el token de ID contiene varias audiencias, el cliente debe verificar que haya un claim “azp” .
m. Si un claim “azp” está presente, el cliente debe verificar que su client_id es el valor del claim.
n. La hora actual debe ser anterior a la hora representada por el claim “exp”
o. El claim “iat” se puede usar para rechazar tokens que se emitieron demasiado lejos de la hora actual, lo que limita la cantidad de tiempo que los nonces deben almacenarse para evitar ataques.
p. Si se solicitó el claim “acr”, el cliente debe verificar que el valor del claim declarado sea apropiado.
q. Cuando se realiza una solicitud de “max_age”, el cliente debe verificar el valor de reclamo auth_time y solicitar una nueva autenticación si determina que ha transcurrido demasiado tiempo desde la última autenticación del usuario final.

4.1.4.2. Claims del Usuario Final

El protocolo permite la ejecución de un conjunto de Claims que permitirán obtener atributos de usuario o información como correo electrónico, nombres, cedula, teléfono, dirección.

El Cliente puede utilizar dos esquemas para realizar estas peticiones por valor del Claim o por su categoría. Adicionalmente permite la inclusión de nuevos atributos para personalizar información que puede ser obtenida por el cliente.

Tipos de Proveedores de Endpoints

r. Autorización de Endpoint (Authorization). El Servidor Endpoint del OP en el que se solicita al usuario autenticarse y se le otorgue acceso al token de identidad u otros datos solicitados como correo, nombre, etc. Es el único endpoint estándar que interactúa con el OP vía UserAgent.
s. Token de Endpoint. Permite a la aplicación cliente intercambiar el código obtenido desde el Endpoint Authorization por un Token ID y un Token de acceso. Para los clientes confidenciales se solicitará autenticar el token del endpoint. También puede aceptar otras entidades "grant Types" que manejen Oauth 2.0 como: "JWT Assertion" y "SAML 2.0 Assertion"
t. Información de Usuarios (UserInfo). Devuelve información del perfil de usuario previamente autorizada. Puede transmitirse empaquetado como un JWT firmado o cifrado.

Adicionalmente existen otros tipos de Proveedores de Endpoints opcionales:

u. WebFinger. Permite el descubrimiento dinámico del proveedor de OpenID Connect para un usuario determinado, por su dirección de correo electrónico u otros datos.
v. Provider metadata. Es un documento en formato JSON que contiene las URL de los extremos del OP y las características de OpenID Connect / OAuth 2.0 soportadas.
w. Provider JWT set. Es un documento JSON que contiene las claves públicas del proveedor en formato JSON Web Key utilizadas para la protección de los tokens ID emitidos.
x. Client Registration. Es un API Web RESTful utilizada para registrar aplicaciones de cliente con el OP que puede ser abierta o pública y de manera protegida mediante una autorización previa.
y. Session Management. Permite que las aplicaciones del cliente validen si el usuario que ha iniciado sesión con el proveedor de OpenID Connect, todavía la mantiene activa

4.1.5. Especificaciones

La guía de especificaciones detallada que ofrece el portal (OpenID Foundation, 2021) esta resumida en la siguiente tabla:

Especificación	Descripción	Tipo
Core	Describe la funcionalidad principal de OpenID Connect	
Descubrimiento	Describe cómo los clientes descubren de forma dinámica información sobre los proveedores de OpenID	Opcional
Registro dinámico	Describe cómo los clientes se registran dinámicamente con proveedores OpenID.	Opcional
Tipos de respuesta múltiple de OAuth 2.0	Describe varios tipos nuevos de respuesta de OAuth 2.0 específicos	
Modo de respuesta de publicación de formulario OAuth 2.0	Describe cómo devolver los parámetros de respuesta de autorización de OAuth 2.0 (incluidos los parámetros de respuesta de autenticación de OpenID Connect	Opcional
Cierre de sesión iniciado por RP	Describe cómo una parte de confianza solicita que un proveedor de OpenID cierre la sesión del usuario final.	Opcional
Gestión de sesiones	Describe cómo gestionar las sesiones de OpenID Connect, incluido el cierre de sesión basado en mensajes posteriores y la función de cierre de sesión iniciado por RP.	Opcional
Cierre de sesión del canal frontal	Describe un mecanismo de cierre de sesión del canal frontal que no utiliza un iframe OP en las páginas de RP	Opcional
Cierre de sesión del canal trasero	Describe un mecanismo de cierre de sesión que utiliza la comunicación directa del canal trasero entre el OP y los RP que se desconectan	Opcional
Federación de OpenID Connect	Describe cómo los conjuntos de OP y RP pueden establecer la confianza mediante la utilización de un operador de federación	Opcional

Tabla 2. Especificaciones OpenID Connect.
Fuente: https://openid.net/connect/

4.2. Piloto Experimental

4.2.1. Descripción detallada del experimento

El experimento consiste en implementar un ambiente de evaluación en Microsoft Azure publicando una página web simple con esquema de autenticación OpenID Connect para someterlo a varias pruebas de seguridad para evaluar el nivel de seguridad que provee a una aplicación web.

El experimento se divide en varias fases:

- Fase 1. Preparación del Ambiente
- Fase 2. Descubrimiento de Amenazas
- Fase 3. Pruebas de Seguridad

4.2.1.1. Primera Fase: Preparación del Ambiente

Para la configuración del ambiente de evaluación se realizan las siguientes actividades:

- Mediante el Módulo PowerApps se crea y registra una aplicación web “Portal Test”

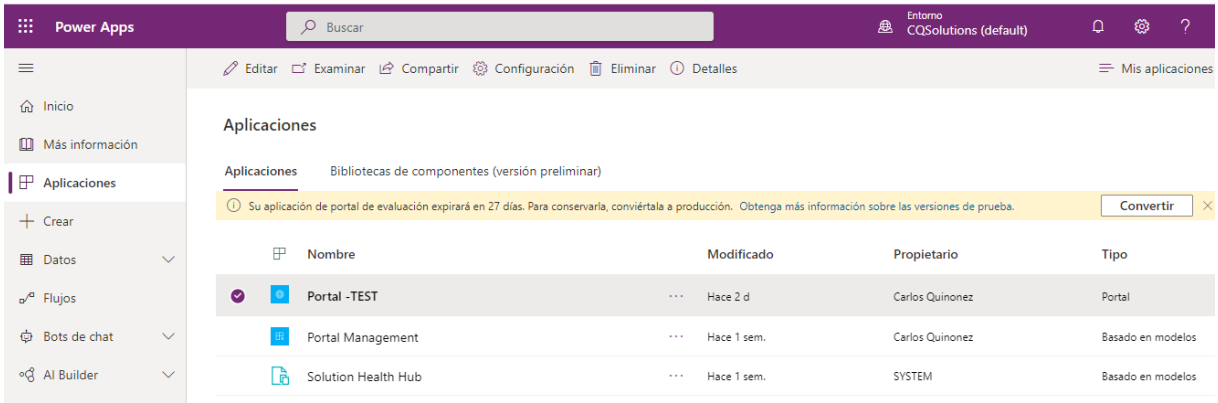

Ilustración 14 - Preparación del Ambiente de Evaluación
Fuente: Portal de Aplicaciones Microsoft – Realización propia

- El sitio es accesible mediante la siguiente dirección URL https://cqtest.powerappsportals.com

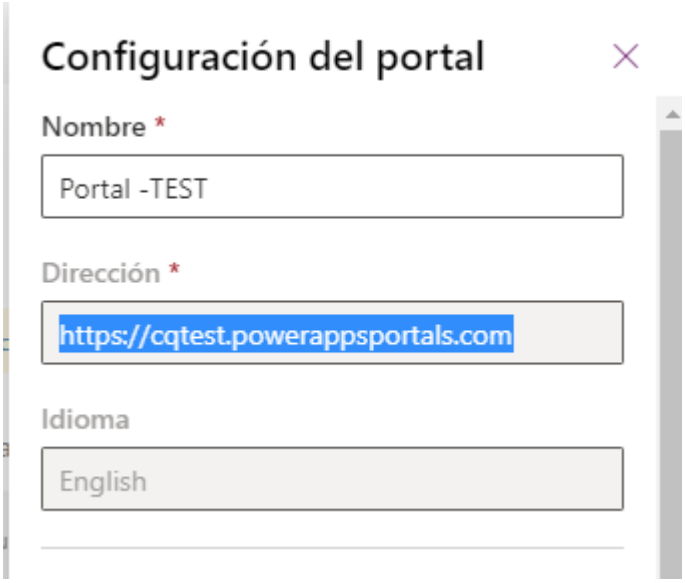

Ilustración 15 - URL del Ambiente de Evaluación
Fuente: Portal de Aplicaciones Microsoft – Realización Propia

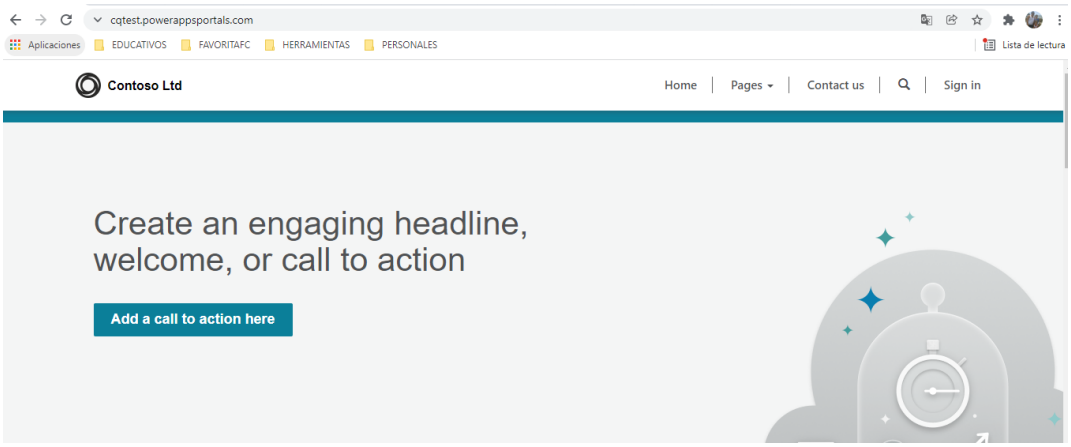

Ilustración 16 - Vista del Portal de Pruebas
Fuente: Portal de Aplicaciones Microsoft – Realización propia

- Se configura como método de autenticación al sitio el protocolo OpenIdConnect mediante el gestor de identidad de Azure Active Directory B2C

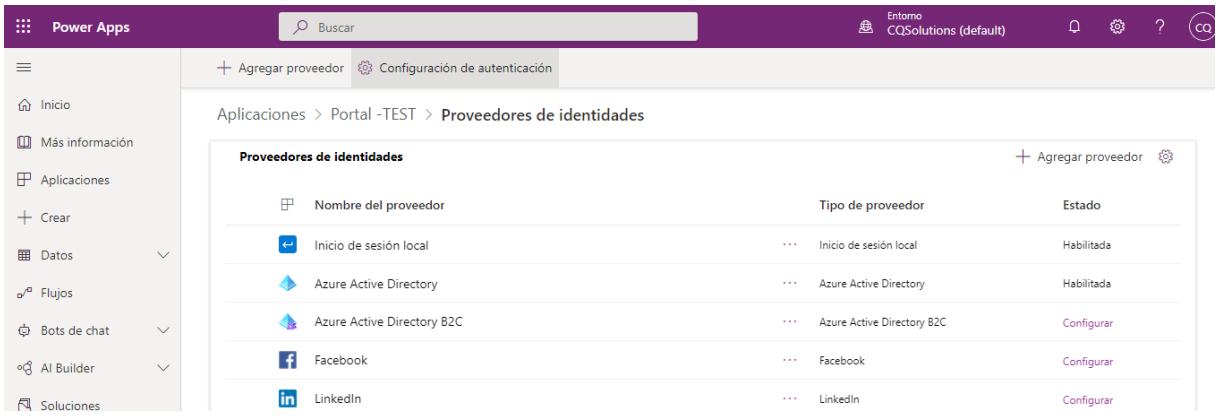

Ilustración 17 - Configuración de Proveedor de Identidad
Fuente: Portal de Aplicaciones Microsoft – Realización Propia

Ilustración 18 - Selección de Proveedor de Identidad
Fuente: Portal de Aplicaciones Microsoft – Realización propia

- Se procede a registrar la aplicación en el Gestor de Identidad

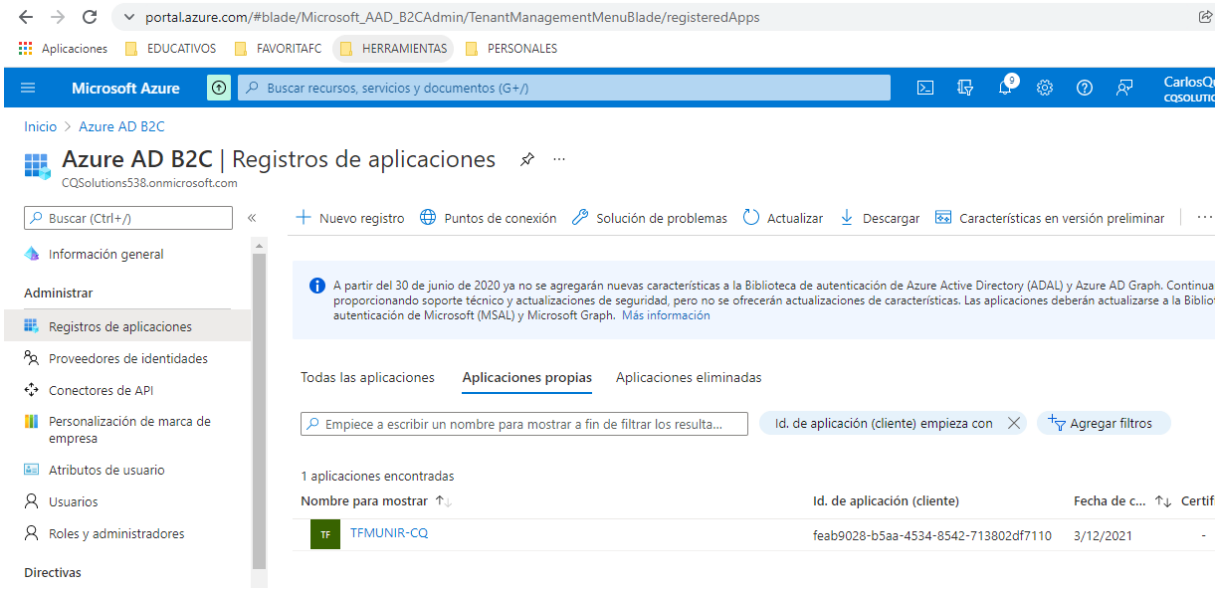

Ilustración 19 - Muestra de Configuración final
Fuente: Portal de Aplicaciones Microsoft – Realización propia

Una vez registrado, se identifican los parámetros de la aplicación

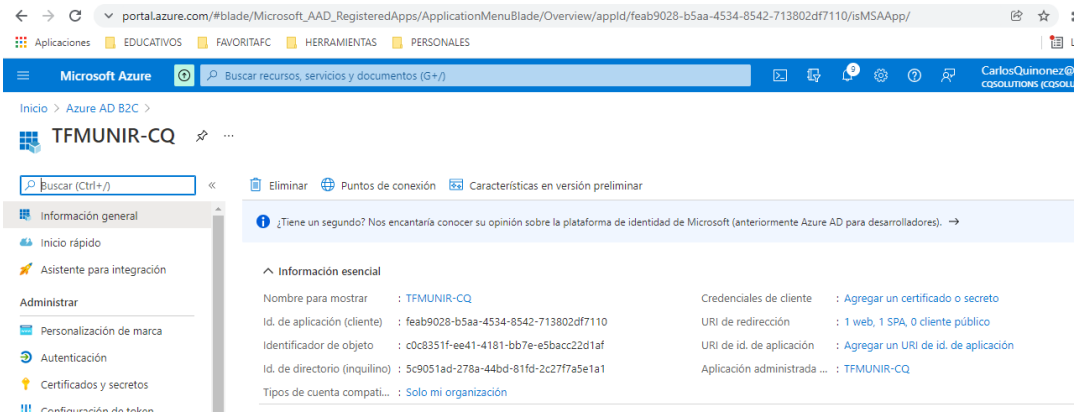

Ilustración 20 - Estado del Método de Autenticación del Sitio
Fuente: Portal de Aplicaciones Microsoft – Realización propia

Se configura el flujo de autorización del protocolo que se va a utilizar para la aplicación

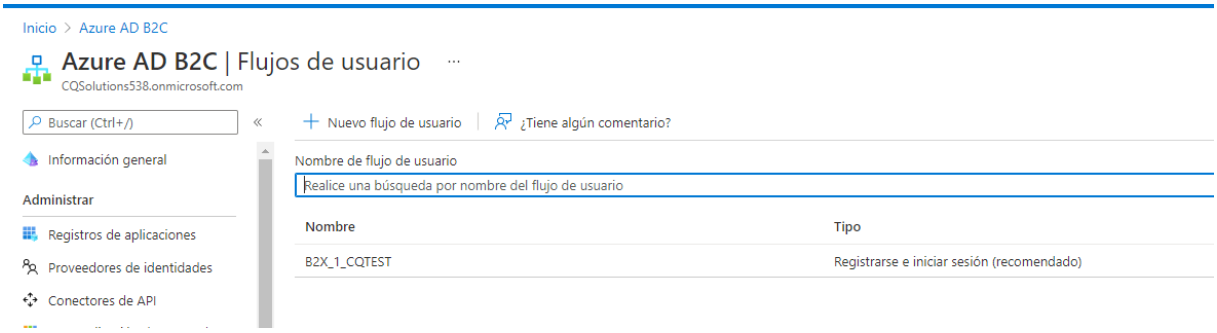

Ilustración 21 - Configuración del Flujo de Autorización
Fuente: Portal de Aplicaciones Microsoft – Realización propia

Se definen los métodos de autenticación que serán habilitados para el portal

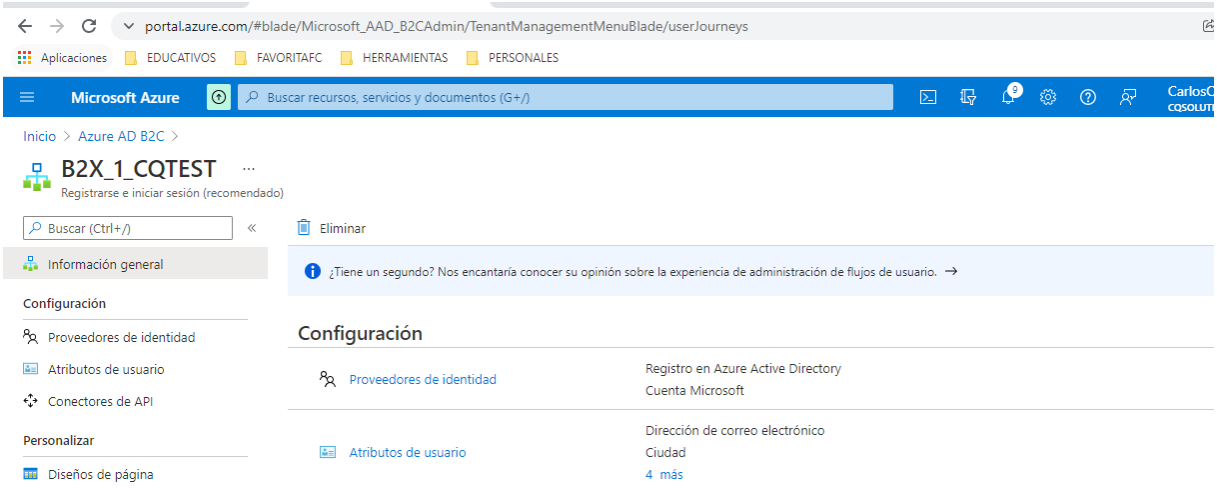

Ilustración 22 - Estado de Flujo de Autorización
Fuente: Portal de Aplicaciones Microsoft – Realización propia

Se definen los reclamos o realms que estarán disponibles para el protocolo

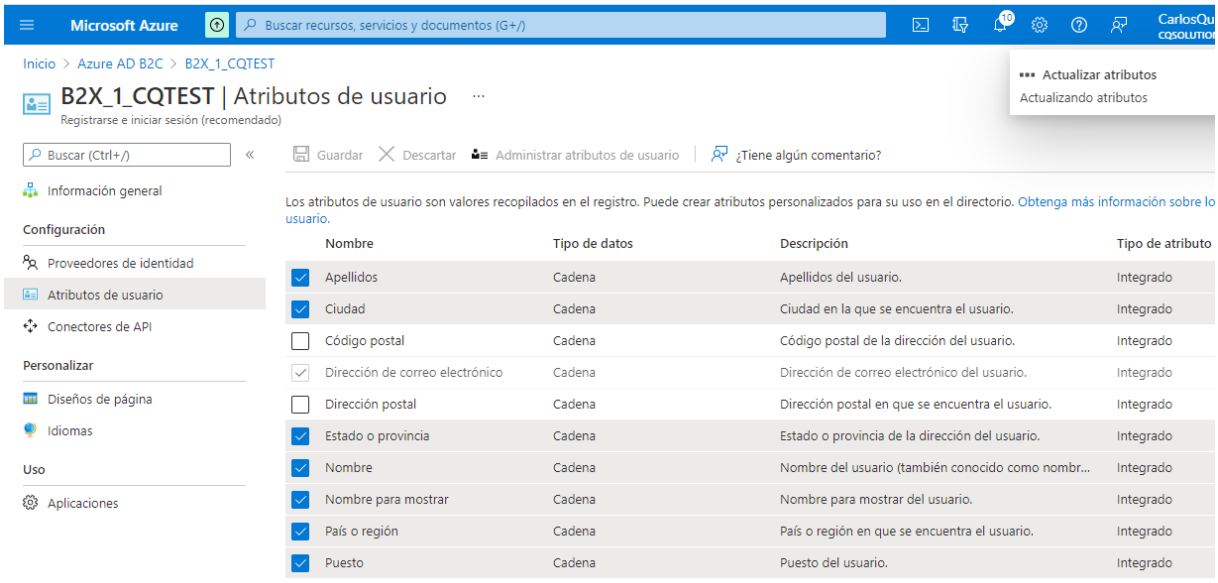

Ilustración 23 - Selección de Reclamos o Realms
Fuente: Portal de Aplicaciones Microsoft – Realización propia

Se asigna el flujo configurado a la aplicación

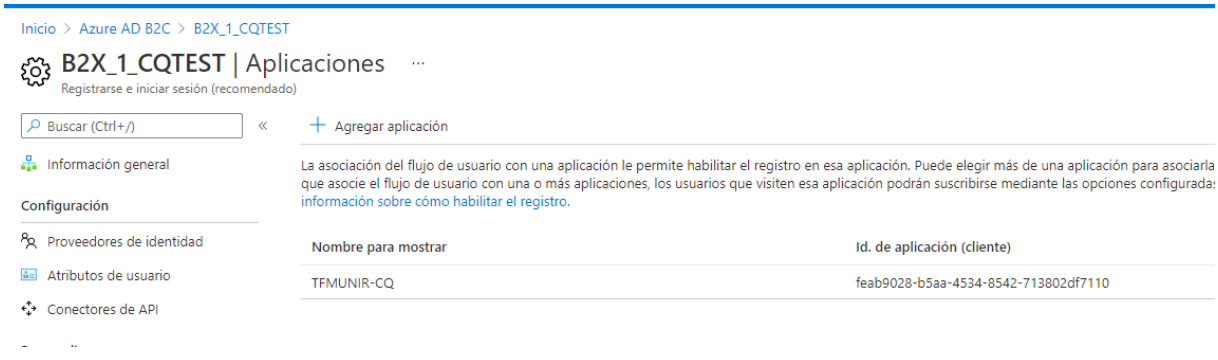

Ilustración 24 - Asignación de Flujo al Sitio de Evaluación
Fuente: Portal de Aplicaciones Microsoft – Realización propia

Una vez registrada y publicada la aplicación son accesibles los diferentes puntos de conexión que procederemos a evaluar según el caso que corresponda

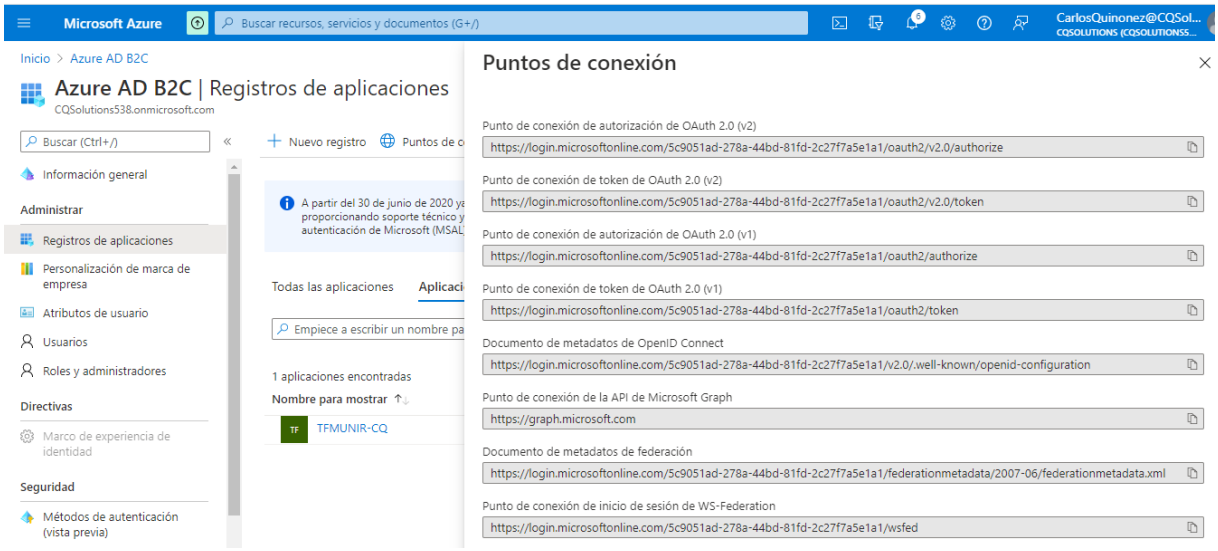

Ilustración 25 - Configuración Final del Protocolo OpenIDConnect + Oauth2
Fuente: Portal de Aplicaciones Microsoft – Realización propia

Posteriormente, se realiza la preparación del escritorio de trabajo para la evaluación del protocolo de autenticación

- Instalación de OwasZAP para realizar la fase descubrimiento de amenazas

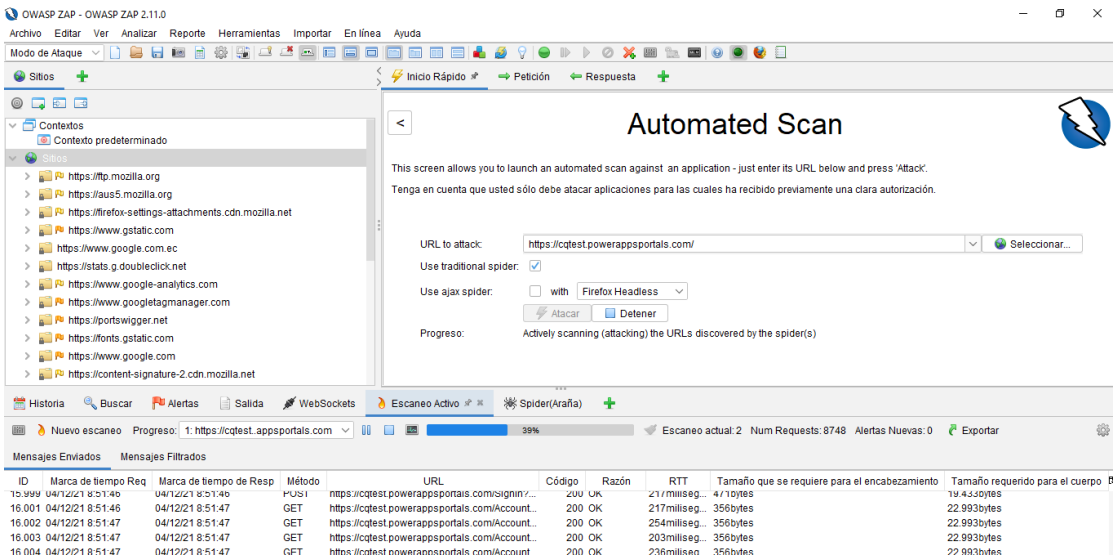

Ilustración 26 - Análisis Dinámico a la URL del protocolo
Fuente: Realización Propia

En el siguiente gráfico se detalla el diagrama del ambiente de evaluación

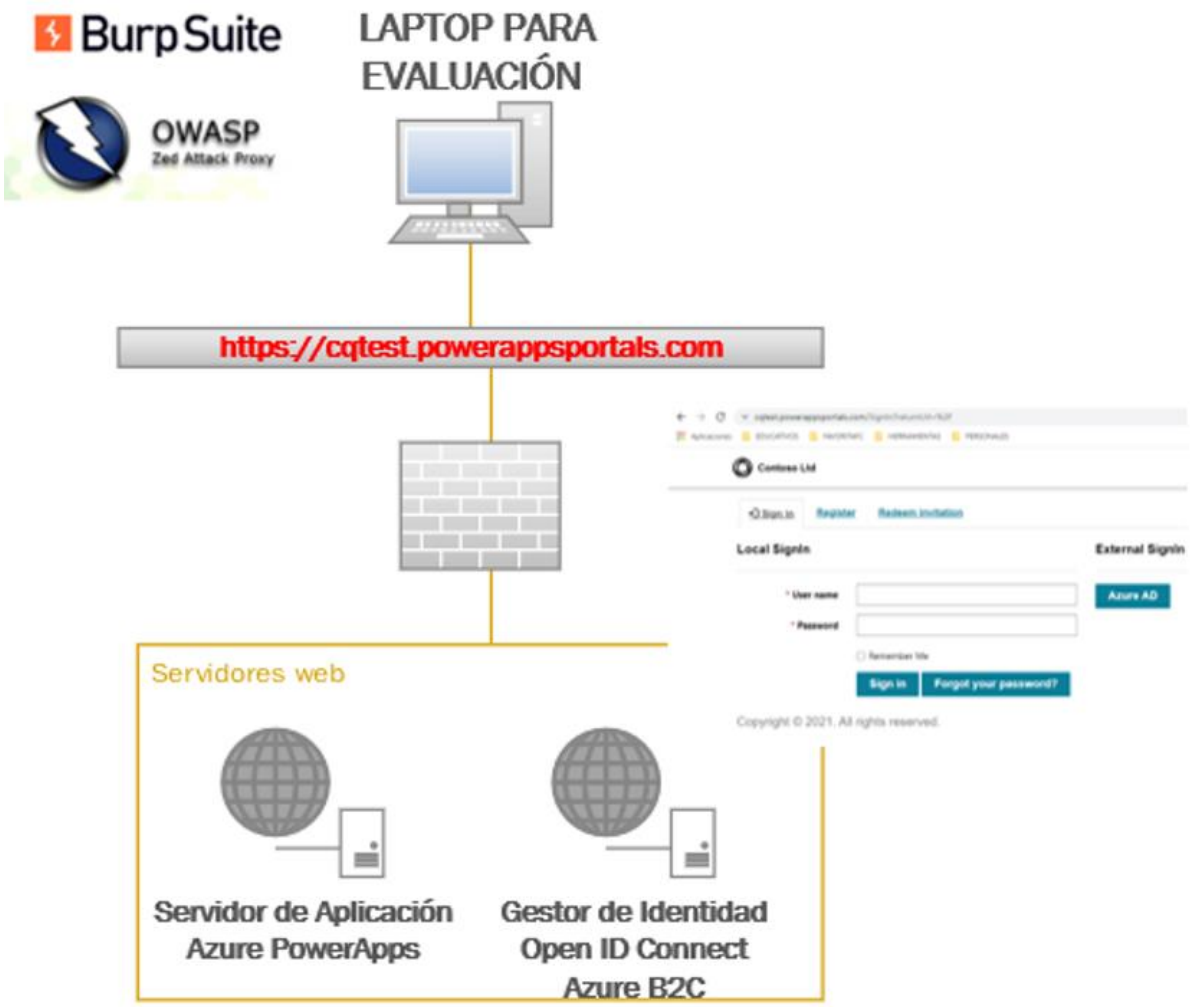

Ilustración 27 - Diagrama de Ambiente de evaluación
Fuente: Realización Propia

4.2.1.2. Segunda Fase: Descubrimiento de Amenazas

Las actividades para el descubrimiento de amenazas se detallan a continuación:

- **Crawling.** Este componente analiza la estructura de la aplicación web y su esquema de autenticación en búsqueda de los nombres de directorios, formularios de autenticación, y toda la información necesaria para posteriormente poder realizar el ataque.

- **Análisis Estático o Pasivo.** Se realiza el análisis automatizado de las cabeceras de las peticiones GET/POST sobre el protocolo de autenticación en

búsqueda de los diferentes métodos de seguridad que tenga la aplicación implementada como el acceso a cookies, métodos HTTP, entre otros.
Durante esta fase, el escáner tiene acceso completo al código de la aplicación objetivo para detectar fallas de implementación en la lógica de verificación o flujos de autorización.

- **Análisis Dinámico o Activo.** Se inyectan porciones de código malicioso predefinidos (payloads) de manera automatizada en los campos de entrada para determinar el nivel de compromiso y riesgo de la aplicación.
  Este análisis busca probar el nivel de seguridad en la transmisión de los datos simulando peticiones de autenticación con datos validos o inválidos, cuyas respuestas serán inspeccionadas para determinar las vulnerabilidades posibles.

- **Análisis Manual de Variantes.** Con la información recolectada en las pruebas anteriores se captura y analiza las interacciones entre mensajes y parámetros modificables para analizar diferentes amenazas a los protocolos de autenticación buscando focalizar la evaluación de seguridad y determinar el nivel de riesgo de la aplicación ejecutando una serie de ataques personalizados

4.2.1.3. Tercera Fase: Pruebas de Seguridad

En este apartado se describen las diferentes pruebas realizadas al protocolo en el ambiente de evaluación que permitan determinar su nivel de seguridad. Las pruebas definidas son las siguientes:

- Validar el uso del protocolo OpenIDConnect
- Validar que el flujo de autorización es OAUTH 2
- Confirmar si el protocolo de cifrado TLS y las cadenas de certificados están configuradas y validadas correctamente.
- Revisar si el proveedor de OpenID realiza es vulnerable a Open Redirection comprobando las URL de los Endpoints.

- Confirmar si se invalidan o expiran las concesiones de autorización posterior al uso.

4.3. Resultados de la Evaluación.

4.3.1. Descripción de Resultados.

- Validar el uso del protocolo OpenIDConnect

  Se proceder a realizar la captura de una petición de autenticación

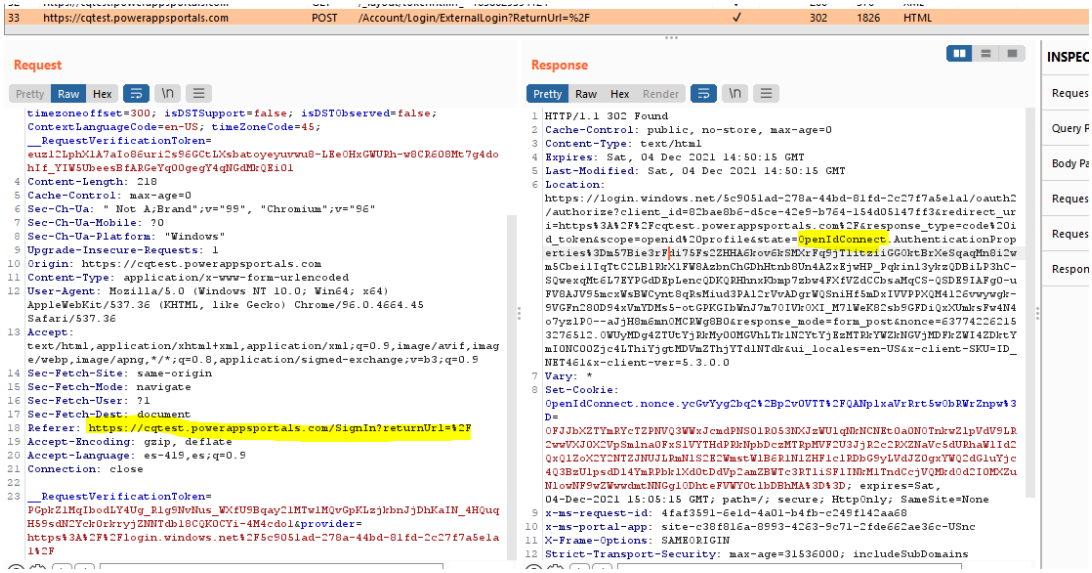

Ilustración 28 - Captura de Petición OpenId Connect al portal de pruebas
Fuente: Realización propia

En la imagen se puede identificar la respuesta en la que se muestra el protocolo utilizado y en la petición confirmamos que el destino es el sitio de pruebas

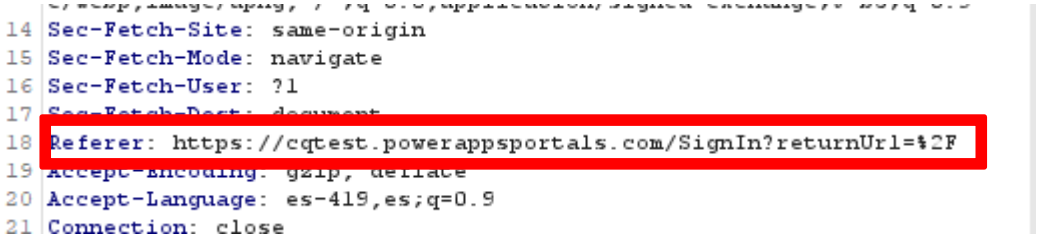

Ilustración 29 - Confirmación del sitio de pruebas
Fuente: Realización propia

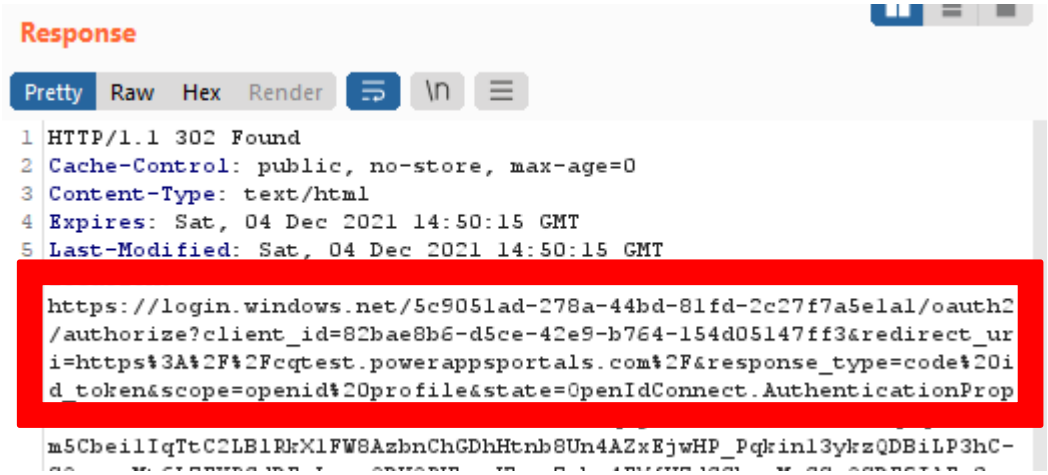

Ilustración 30 - Confirmación del protocolo en la petición
Fuente: Realización propia

- Validar que el flujo de autorización es OAUTH 2

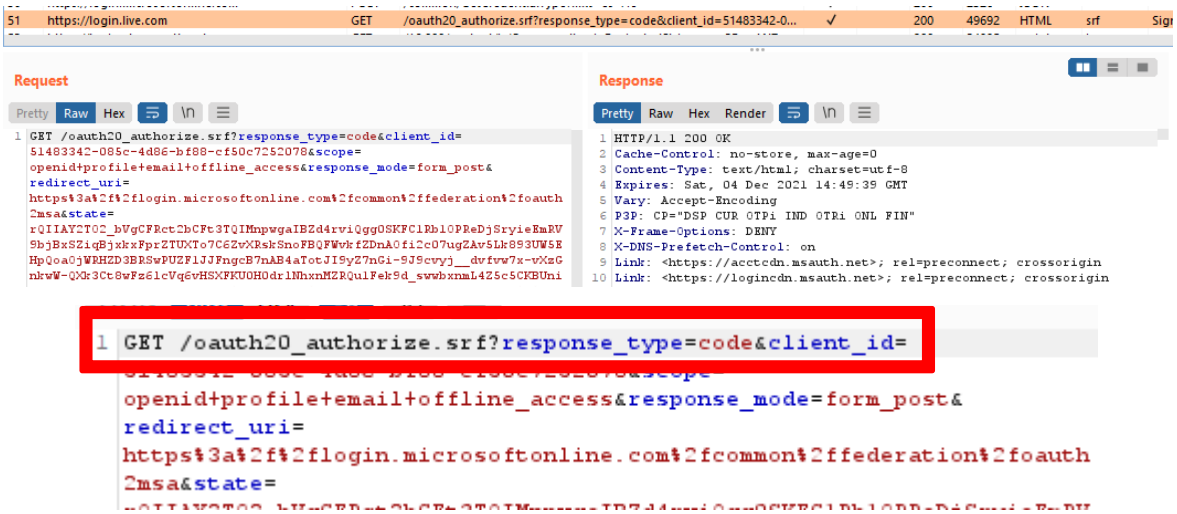

Ilustración 31 - Confirmación del Flujo de Autorización
Fuente: Realización propia

- Confirmar si el protocolo de cifrado TLS y las cadenas de certificados están configuradas y validadas correctamente.

  Se identifica la existencia de la cabecera contiene “Strict Transport Security” para garantizar que la comunicación se realiza por canal cifrado

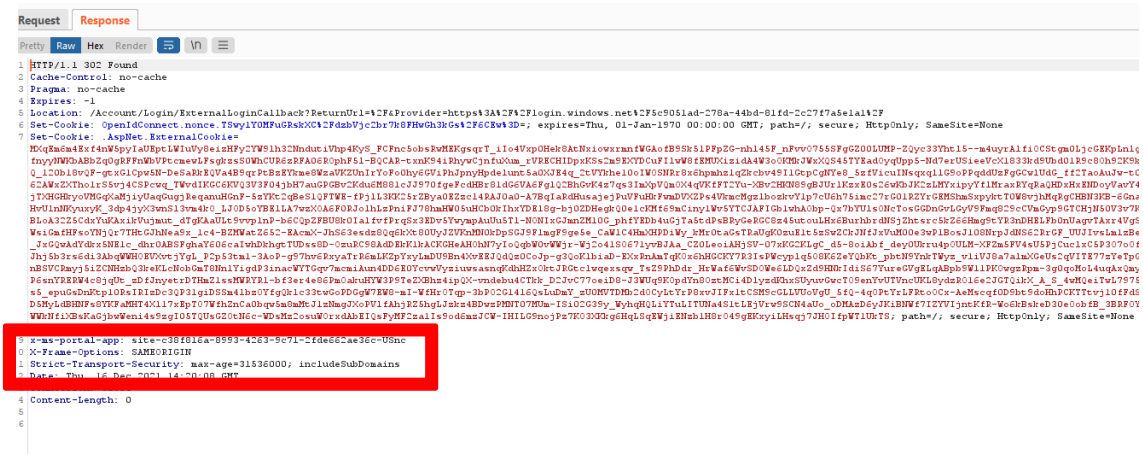

Ilustración 32 - Confirmación de cabeceras utilizadas
Fuente: Realización propia

Para esta prueba, vamos a utilizar el portal SSLLABS que realiza un análisis del nivel de seguridad de las suites criptográficas.

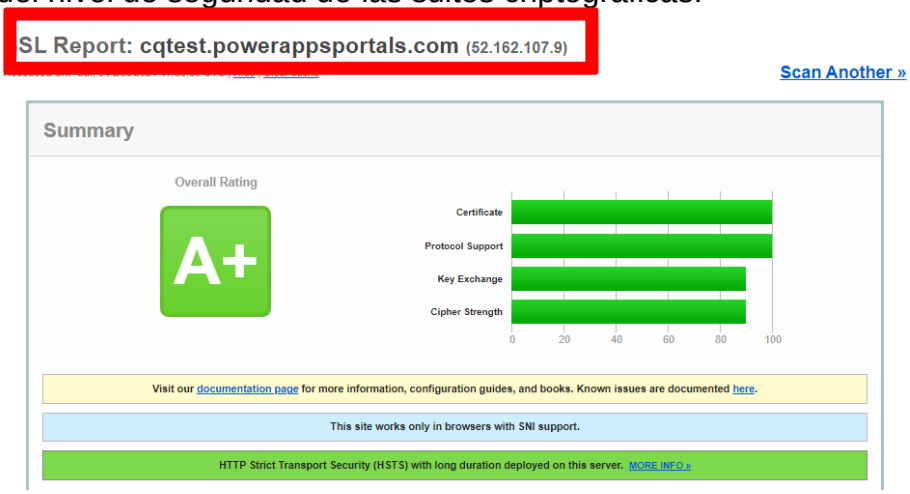

Ilustración 33 - Validación del nivel de seguridad de cifrado
Fuente: Portal SSL Labs

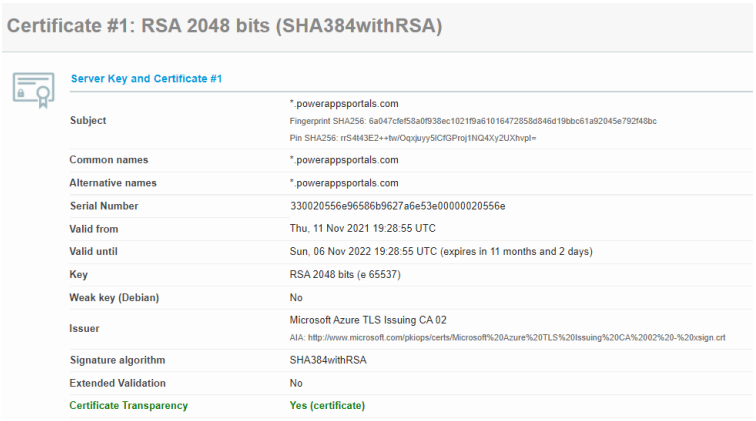

Ilustración 34 - Validación del Certificado utilizado
Fuente: Portal SSL LAbs

- Revisar si el proveedor de OpenID realiza es vulnerable a Open Redirection comprobando las URL de los Endpoints.

Al realizar la petición al service provider "login.windows.net" utiliza redireccionamientos HTTP mediante el parámetro "redirect_uri" para dirigir el navegador al proveedor de OpenID después de un inicio de sesión exitoso.

Ilustración 35 - Comprobación de Redirección URL
Fuente: Realización propia

El objetivo de esta prueba es modificar el parámetro para redirigir la petición a un sitio externo. Si se logra falsificar el enlace y redireccionar la petición a un sitio malicioso, el sitio es vulnerable a ataques tipo phishing. En la siguiente imagen se presenta la petición al OpenID Provider

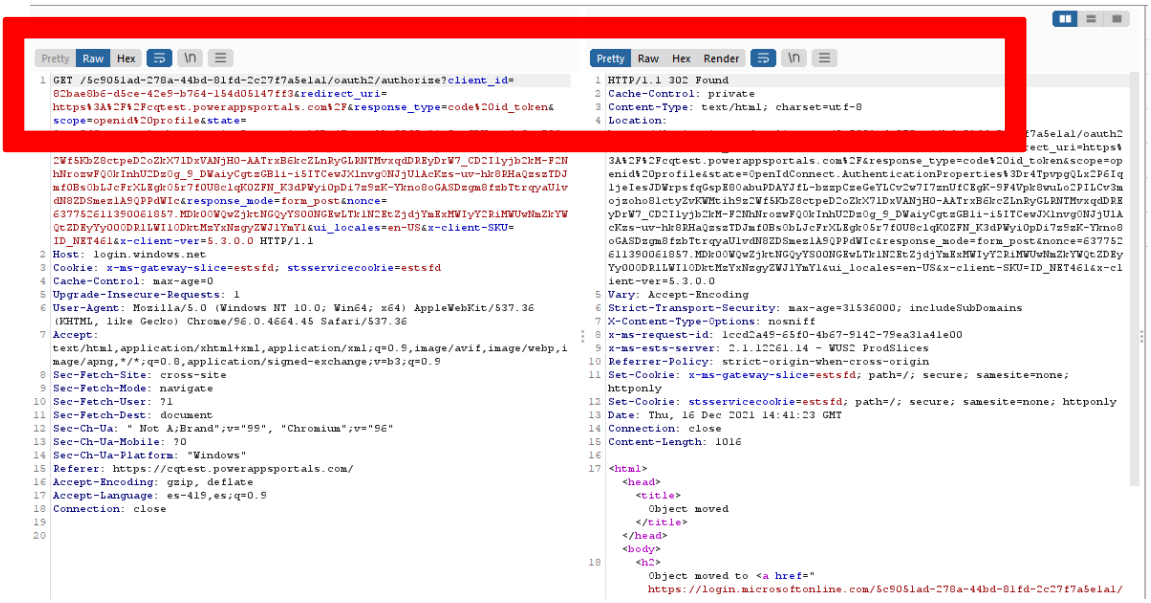

Ilustración 36 - Prueba de Seguridad Redirect Url
Fuente Realización propia

Se modifica la url de redirección. Se visualiza que el OpenID Provider bloquea la petición maliciosa confirmando que el flujo valida la URL para minimizar el riesgo de este tipo de ataques.

- Validar que los tokens de acceso y las concesiones de autorización son generadas aleatoriamente y son impredecibles.

El flujo de autorización incluye un flujo de validación de tokens como se puede demostrar en la siguiente imagen

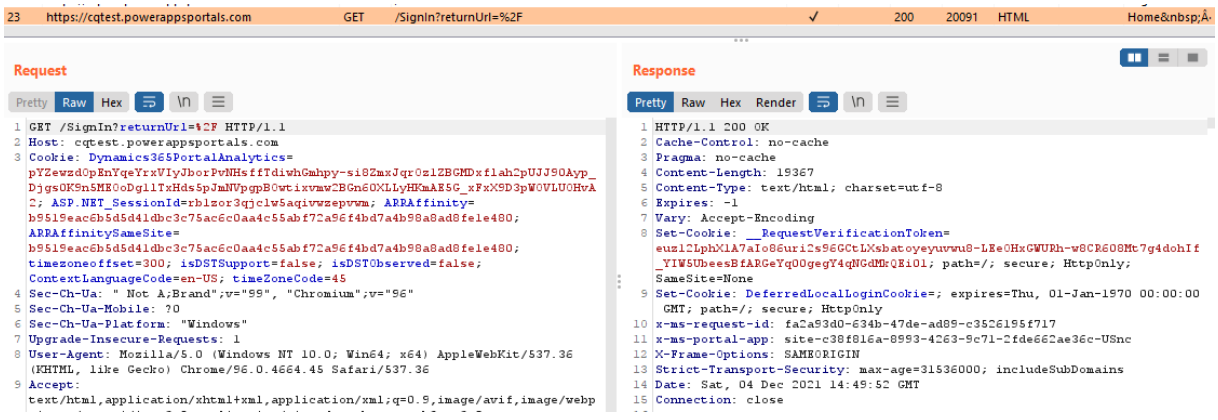

Ilustración 37 - Prueba de Seguridad al Token
Fuente: Realización propia

El flujo de validación se encarga de confirmar la vigencia y uso de los tokens para garantizar que el mismo no pueda ser reutilizado para alterar otras peticiones.

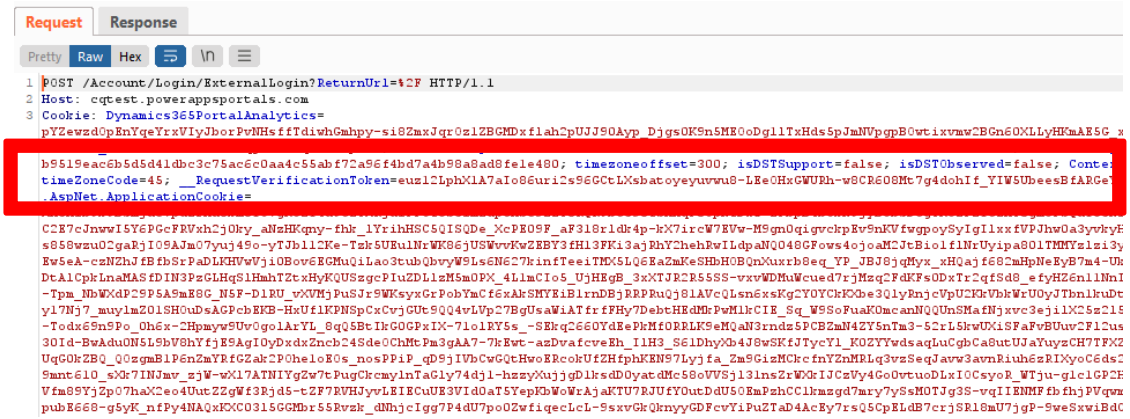

Ilustración 38 - Resultado de la prueba de Token
Fuente: Realización propia

4.3.2. Evaluación de Riesgos.

Para la evaluación del riesgo se tomarán en cuenta los siguientes parámetros de Impacto vs Probabilidad

			GRAVEDAD (IMPACTO)				
			MUY BAJO **1**	BAJO **2**	MEDIO **3**	ALTO **4**	MUY ALTO **5**
PROBABILIDAD	MUY ALTA	**5**	5	10	15	20	25
	ALTA	**4**	4	8	12	16	20
	MEDIA	**3**	3	6	9	12	15
	BAJA	**2**	2	4	6	8	12
	MUY BAJA	**1**	1	2	3	4	5

Tabla 3 - Tabla de Riesgos
Fuente: Realización propia

La siguiente escala representará el código de colores seleccionado para establecer la matriz de riesgos

	Riesgo muy grave. Requiere medidas preventivas urgentes. No se debe iniciar el proyecto sin la aplicación de medidas preventivas urgentes y sin acotar sólidamente el riesgo.
	Riesgo importante. Medidas preventivas obligatorias. Se deben controlar fuertemente llas variables de riesgo durante el proyecto.
	Riesgo apreciable. Estudiar económicamente ai es posible introducir medidas preventivas para reducir el nivel de riesgo. Si no fuera posible, mantener las variables controladas.
	Riesgo marginal.Se vigilará aunque no requiere medidas preventivas de partida.

Ilustración 39 - Niveles de Riesgo
Fuente: Realización propia

En la siguiente Tabla se detalla los riesgos identificados y las valoraciones correspondientes tomando en consideración la exposición de una aplicación en la nube.

RIESGO	Probabilidad	Gravedad	Valoración	Nivel de Riesgo
Afectación a la Privacidad	3	15	45	Importante
Aparición de nuevas vulnerabilidades	4	16	64	Importante
Perdida de Confidencialidad	2	20	40	Importante
Perdida de Integridad	1	12	12	Importante
Perdida de Disponibilidad	3	16	48	Importante
Explotación de una Vulnerabilidad	2	12	24	Muy grave

Tabla 4 - Valoración del Riesgo
Fuente: Realización propia

Una vez que se ha identificado y valorado el riesgo inherente se analizan los controles que permitirán establecer el riesgo residual tal como se muestra en la siguiente tabla

Riesgo	Valor del Riesgo	Nivel de Riesgo	Controles	Efectividad		Peso		Promedio	Riesgo Residual	
Afectación a la Privacidad	45	Importante	Realizar peticiones a los flujos de autorización - método POST.	Destacado	5.00	Necesario	2.00	10	5	Apreciable
			Validación del parámetro "Nonce"	Destacado	5.00	Necesario	2.00			
			Las suites criptográficas deben utilizar claves superiores o iguales a 2048 bits.	Destacado	5.00	Necesario	2.00			
Aparición de nuevas vulnerabilidades	64	Importante	Utilizar algoritmos de firma digital de 256 bits o superiores.	Alto	4.00	Necesario	2.00	8	8	Apreciable
Perdida de Confidencialidad	40	Importante	Validar las firmas digitales del Token de Identificación	Alto	4.00	Necesario	2.00	8	5	Apreciable
			Hay que confirmar que la audiencia corresponda al identificador de la aplicación.	Alto	4.00	Necesario	2.00			
			Utilizar protocolo TLS 1.2 o superior para cifrar el canal de comunicación y evitar suites criptográficas débiles.	Alto	4.00	Necesario	2.00			
			Los perfiles TLS deberían ser correspondientes entre los clientes y el protocolo.	Alto	4.00	Necesario	2.00			
Perdida de Integridad	12	Importante	Validar la expiración de la aserción	Medio	3.00	Necesario	2.00	8	2	Marginal
			Hay que confirmar que el token de acceso es el apropiado.	Destacado	5.00	Necesario	2.00			
Perdida de Disponibilidad	48	Importante	Caducar los tokens de identificación por tiempo	Medio	3.00	Necesario	2.00	6	8	Apreciable
Explotación de una Vulnerabilidad	24	Muy grave	En caso de usar “Secret ID” deben ser superiores o iguales a 128 bits.	Medio	3.00	Relevante	1.00	7	4	Apreciable
			Las suites criptográficas deben utilizar claves superiores o iguales a 2048 bits.	Destacado	5.00	Necesario	2.00			

Tabla 5 - Análisis del Riesgo
Fuente: Realización propia

5. CONCLUSIONES Y TRABAJOS FUTUROS

5.1. Análisis sobre el TFM

El protocolo OpenID Connect ha sido la base para el desarrollo de los protocolos y federaciones de autenticación más utilizadas en la actualidad porque permita manejar conceptos de escalabilidad en el desarrollo de aplicaciones y servicios, así como la delegación de los procesos de autenticación permitiendo a los desarrolladores ahorrar tiempo en las implementaciones. En este trabajo se pretendió evaluar el nivel de seguridad que provee este protocolo en la actualidad y desarrollar los criterios necesarios para establecer recomendaciones para una implementación segura y con mínimos riesgos de seguridad.

A continuación, se resumen las recomendaciones para una implementación segura del protocolo OpenID Connect.

- Realizar peticiones de autenticación a los flujos de autorización mediante método POST.
- Validar que el valor del parámetro “nonce” coincida con el enviado en la solicitud
- Validar la expiración de la aserción
- Validar las firmas digitales del Token de Identificación.
- Confirmar que la audiencia corresponda al identificador de la aplicación.
- Confirmar que el token de acceso es el apropiado.
- Caducar los tokens de identificación por tiempo
- Utilizar protocolo TLS 1.2 o superior para cifrar el canal de comunicación y evitar suites criptográficas débiles.
- Los perfiles TLS deberían ser correspondientes entre los clientes y el protocolo.
- Las suites criptográficas deben utilizar claves de 2048 bits o superiores
- En caso de usar “Secret ID” deben ser superiores o iguales a 128 bits.

- Utilizar algoritmos de firma digital de 256 bits o superiores.

Otras consideraciones importantes que se deben tener al implementar son:

- Implementar mecanismos de múltiple factor de autenticación para minimizar ataques de relleno de credenciales, fuerza bruta, entre otros.
- Utilizar políticas de complejidad de contraseñas para evitar el uso de credenciales de fácil deducción.
- Tener controles para bloquear intentos recurrentes de acceso a las credenciales, así como desafíos por imagen o ecuaciones para minimizar ataques recurrentes o automatizados por robots.
- Utilizar flujos estándares y comprobados evitando la personalización de flujos que expongan datos de autenticación.
- Invalidar sesiones por inactividad o tiempos absolutos.
- Registrar y controlar patrones de acceso recurrentes satisfactorios y fallidos mediante un SIEM.

Durante el presente trabajo se logró demostrar el alto nivel de confiabilidad y seguridad que presenta el protocolo cuando es implementado siguiendo los lineamientos recomendados por lo que se puede concluir que las pruebas resultaron satisfactorias en todo sentido y el nivel de riesgo inherente es aceptable.

5.2. Contribuciones del trabajo

El presente trabajo realiza un análisis minucioso y exhaustivo del protocolo OpenId Connect, su historia y evolución, funcionamiento, riesgos y como realizar pruebas de seguridad sobre una aplicación implementada en la nube de Microsoft Azure configurada para que este protocolo gestione la autenticación y autorización.

De igual manera se describe la implementación del protocolo de manera que los riesgos asociados al proceso de control de accesos sean mínimos.

5.3. Líneas de trabajo futuro

El alcance del presente trabajo imposibilitó profundizar el análisis de las interacciones del protocolo para definir una estrategia que permita robustecer los flujos que permita reducir las probabilidades de ataques que afecten los servicios de autenticación en ambientes híbridos, es decir, aplicaciones que complementan servicios en implementaciones que combinan DataCenters locales y en Nube.

Los ámbitos que pueden complementar este trabajo son:

- Esquemas de monitoreo para alerta temprana de amenazas a OpenID Connect en nube.
- Analizar las mejores prácticas para contrarrestar ataques dedicados.
- Realizar pruebas de Penetración utilizando tecnologías que contengan Inteligencia Artificial

6. BIBLIOGRAFIA

Amazon Web Services. (2021). Obtenido de https://aws.amazon.com/es/identity/federation/

AOL. (14 de 02 de 2007). Obtenido de http://dev.aol.com/aol-and-63-million-openids

Becker, P. (4 de 12 de 2006). Obtenido de https://www.zdnet.com/article/the-case-for-openid/

Biehl, M. (2019). *OpenID Connect y JWT* (Vols. Volumen 6 of the API-University Series). API-University Press.

Borghello, C. (09 de Septiembre de 2021). *Segu-Info: Noticias sobre Seguridad de la información*. Obtenido de https://blog.segu-info.com.ar/2021/09/bienvenido-OWASP-top-10-2021-i.html

Boston University. (2020). *Undestanding Authentication, Authorization and Encryption*. Obtenido de http://www.bu.edu/tech/about/security-resources/bestpractice/auth/

Cantón, D. (20 de 05 de 2014). *Seguridad en OAuth 2.0*. Obtenido de https://www.incibe-cert.es/blog/seguridad-oauth-2-0

Deitel, P., Deitel, H., & Deitel, A. (2014). *COMO PROGRAMAR INTERNET & WORLD WIDE WEB.* España: Pearson.

Electronic Identification. (26 de Julio de 2021). Obtenido de https://www.electronicid.eu/es/blog/post/verificacion-de-identidad/es

Facebook. (01 de 07 de 2009). *Facebook Developers*. Obtenido de https://web.archive.org/web/20090701093750/http://wiki.developers.facebook.com/index.php/OpenID_Requirements

Fernandez, L. (27 de Junio de 2020). *RZ Redes Zone*. Obtenido de https://www.redeszone.net/tutoriales/seguridad/diferencias-autenticacion-autorizacion/

Fitzpatrick, B. (07 de 2005). *Danga*. Obtenido de http://www.danga.com/openid/

Fitzpatrick, B. (16 de 05 de 2005). *LiveJournal*. Obtenido de https://lj-dev.livejournal.com/683939.html

Fitzpatrick, B. (07 de 2005). *LiveJournal*. Obtenido de http://brad.livejournal.com/2226738.html

Fitzpatrick, B. (30 de 05 de 2006). Obtenido de http://brad.livejournal.com/2226738.html

Garretson, C. (31 de 01 de 2007). Obtenido de https://www.networkworld.com/article/2303615/demo--07--symantec-launches-online-identity-initiative.html

Grey, V. (02 de 04 de 2006). Obtenido de http://lists.danga.com/pipermail/yadis/2006-April/002388.html

Hoyt, J. (15 de 03 de 2006). Obtenido de http://lists.danga.com/pipermail/yadis/2006-March/002304.html

JanRain. (04 de 09 de 2013). Obtenido de https://thenextweb.com/insider/2013/09/04/myopenid-to-shut-down/

Jones, M. (27 de 04 de 2015). *OpenID*. Obtenido de https://openid.net/2015/04/27/final-oauth-2-0-form-post-response-mode-specification-approved/

Jones, M. (08 de 01 de 2020). *OpenId*. Obtenido de https://openid.net/2020/01/08/second-implementers-draft-of-openid-connect-federation-specification-approved/

Kukic, A. (2020). *Auth0.* Obtenido de https://auth0.com/resources/whitepapers/definitive-guide-to-single-sign-on

López, R. E. (Diciembre de 2017). PRUEBAS DE PENETRACIÓN EN APLICACIONES WEB. *REVISTA TECNOLÓGICA ITCA-FEPADE*, 7. Obtenido de http://redicces.org.sv/jspui/bitstream/10972/3018/1/Articulo2.pdf

Microsoft. (21 de 07 de 2020). *Protocolos OAuth 2.0 y OpenID Connect en la plataforma de identidad de Microsoft.* Obtenido de https://docs.microsoft.com/es-es/azure/active-directory/develop/active-directory-v2-protocols

Mieres, J. (Enero de 2009). Obtenido de https://www.evilfingers.net/publications/white_AR/01_Ataques_informaticos.pdf

Mitre Corporation. (30 de Agosto de 2021). *VISTA CWE: Desarrollo de software*. Obtenido de https://cwe.mitre.org/data/definitions/699.html

MySpace. (22 de 07 de 2008). Obtenido de https://www.businesswire.com/news/home/20080722006024/en

Nascimento, A. E. (2017). *OAuth 2.0 Cookbook.* Birmingham: Packt Publishing Ltd.

OASIS. (2014). Obtenido de http://saml.xml.org/history

OpenID. (30 de 10 de 2008). Obtenido de https://openid.net/2008/10/30/microsoft-and-google-announce-openid-support/

OpenID. (26 de 02 de 2014). Obtenido de https://openid.net/2014/02/26/the-openid-foundation-launches-the-openid-connect-standard/

OpenID. (2020). *Open ID Foundation*. Obtenido de https://openid.net/specs/openid-connect-basic-1_0.html

OpenID Foundation. (s.f.). Obtenido de https://openid.net/developers/specs/

OpenID Foundation. (06 de 2007). Obtenido de https://openid.net/foundation/

OpenID Foundation. (2020). *OPENID*. Obtenido de https://openid.net/what-is-openid/

OpenID Foundation. (2021). Obtenido de https://openid.net/connect/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/Top10/A01_2021-Broken_Access_Control/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/Top10/A02_2021-Cryptographic_Failures/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/Top10/A03_2021-Injection/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/Top10/A06_2021-Vulnerable_and_Outdated_Components/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/Top10/A09_2021-Security_Logging_and_Monitoring_Failures/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/Top10/A00-about-OWASP/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/Top10/A00_2021_Introduction/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/Top10/A04_2021-Insecure_Design/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/Top10/A05_2021-Security_Misconfiguration/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/Top10/A07_2021-Identification_and_Authentication_Failures/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/Top10/A08_2021-Software_and_Data_Integrity_Failures/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/Top10/A10_2021-Server-Side_Request_Forgery_%28SSRF%29/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/www-project-application-security-verification-standard/

OWASP Foundation. (2021). Obtenido de https://OWASP.org/www-community/OWASP_Risk_Rating_Methodology

OWASP Foundation. (2021). Obtenido de https://cheatsheetseries.OWASP.org/cheatsheets/Authentication_Cheat_Sheet.html

OWASP Foundation. (2021). *A01_2021-Broken_Access_Control/*. Obtenido de https://OWASP.org/Top10/A01_2021-Broken_Access_Control/

OWASP Foundation. (2021). *Cómo utilizar el Estándar OWASP Top 10*. Obtenido de https://OWASP.org/Top10/A00_2021_How_to_use_the_OWASP_Top_10_as_a_standard/

Parecki, A. (2020). *OAuth 2.0 Simplified.* San Francisco: Okta.

Parecki, A. (2020). *Oauth Community Site*. Obtenido de https://oauth.net/

Parecki, A. (2021). Obtenido de https://www.oauth.com/oauth2-servers/differences-between-oauth-1-2/

Recordon, D. (24 de 10 de 2005). *David RecordonDavid Recordon*. Obtenido de http://lists.danga.com/pipermail/yadis/2005-October/001511.html

Recordon, D. (16 de 06 de 2006). Obtenido de http://lists.danga.com/pipermail/yadis/2006-June/002631.html

Recordon, D. (05 de 12 de 2007). Obtenido de http://openid.net/2007/12/05/openid-2_0-final-ly/

Reed, D. (31 de 10 de 2005). Obtenido de http://lists.danga.com/pipermail/yadis/2005-October/001544.html

Reed, D. (30 de 11 de 2008). *Equals Drummond*. Obtenido de https://equalsdrummond.name/2008/11/30/xrd-begins/

Richer, J. (s.f.). Obtenido de https://oauth.net/articles/authentication/

Shepard, L. (18 de 05 de 2009). *Facebook Developers*. Obtenido de https://web.archive.org/web/20090701093321/http://developers.facebook.com/news.php?blog=1&story=246

SourceForge, Inc. (13 de 05 de 2008). Obtenido de https://web.archive.org/web/20080513100231/http://www.primenewswire.com/newsroom/news.html?d=142213

StackExchange. (2018). Obtenido de https://meta.stackexchange.com/questions/307647/support-for-openid-ended-on-july-25-2018

SXIP. (10 de 12 de 2005). *Identity 2.0*. Obtenido de http://identity20.com/?p=44

The Mitre Corporation. (30 de Agosto de 2021). *CWE*. Obtenido de https://cwe.mitre.org/

Thibeau, D. (17 de 04 de 2015). *OpenID*. Obtenido de https://openid.net/2015/04/17/openid-connect-certification-program/

VERACODE. (2021). Obtenido de https://www.veracode.com/sites/default/files/pdf/resources/sossreports/state-of-software-security-volume-11-veracode-report.pdf

VERACODE. (2021). Obtenido de https://www.veracode.com/security/OWASP-top-10

Verisign. (02 de 2007). Obtenido de http://blogs.verisign.com/infrablog/2007/02/verisign_microsoft_partners_to_1.php

Williams, J. (2021). Obtenido de https://OWASP.org/www-community/OWASP_Risk_Rating_Methodology

Yahoo! C. (17 de 01 de 2008). Obtenido de https://web.archive.org/web/20080304014817/http://biz.yahoo.com/bw/080117/20080117005332.html

Anexos A - ANÁLISIS DETALLADO DE OWASP

En el presente Anexo, se describe el Top 10 de Amenazas de Seguridad descritas por OWASP en el más reciente análisis realizado en el 2021 y los factores determinantes de su estadio.

En la siguiente tabla se resume el Top 10 de la Amenazas:

Amenaza	DESCRIPCIÓN	RIESGOS
A01: 2021 - Control de acceso roto.	La categoría involucra todos los riesgos asociados a la evasión de controles de acceso a recursos permitiendo la divulgación y mal uso de información no autorizada, así como la modificación o eliminación de datos.	• Violación del principio de mínimo privilegio o denegación por defecto, en el que los accesos solo deben asignarse según la necesidad, a través de roles o usuarios específicos, pero están accesibles sin restricciones. • Evasión de comprobaciones de control de acceso alterando parámetros para modificar la URL de peticiones de manera forzada. • Permitir referenciar objetos de manera insegura permitiendo ver o editar la cuenta de otra persona, proporcionando su identificador único. • Acceso a API con controles de acceso faltantes para POST, PUT y DELETE. • Elevación de privilegios. Acceder como usuario sin iniciar sesión o actuar como administrador cuando inicie sesión como usuario.

Amenaza	DESCRIPCIÓN	RIESGOS
		• Manipulación de metadatos, consiste en reproducir o alterar un token de control de acceso JSON Web Token (JWT), cookies o campo ocultos manipulados para elevar privilegios invalidando el control de sesión de JWT.
		• Configuración inapropiada de CORS que permita el acceso a la API desde orígenes no autorizados o no confiables.
		• Permitir forzar la navegación a páginas autenticadas como usuario no autenticado o páginas privilegiadas como usuario estándar.
		• Uso de algoritmos o protocolos criptográficos antiguos o débiles de forma predeterminada
A02: 2021 - Fallos criptográficos	Las prioridades deben incluir el análisis de las necesidades de protección de los datos en tránsito y en reposo especialmente aquellos que están sujetos a legislaciones de protección de datos personales como GDPR o regulaciones para información financiera como PCI DSS.	• Mal uso de claves criptográficas. Uso de llaves de manera predeterminada Generación de claves criptográficas débiles Fallo en la gestión o rotación de claves Uso de llaves en el código fuente
		• Falta de seguridad en el cifrado. Directivas de seguridad de los encabezados HTTP.
		• Certificados publicados o cadenas de confianza validados incorrectamente o falta de validación.

Amenaza	DESCRIPCIÓN	RIESGOS
		• Mal uso de claves criptográficas. Las contraseñas se utilizan como claves criptográficas en ausencia de una función de derivación de claves base de contraseñas. • Utilización de funciones de aleatoriedad con fines criptográficos. Las funciones que generan caracteres aleatorios no se diseñaron para cumplir con los requisitos criptográficos. • Uso de funciones hash obsoletas e inseguras, como MD5 o SHA1. • Uso de métodos de relleno criptográfico vulnerables, como PCKS en sus versiones obsoletos. • Explotación de los mensajes de error criptográficos.
A03: 2021 – Inyección	Involucra todas aquellas vulnerabilidades o fallos en las que el vector es la entrada de código externo siendo ejecutado de en el cliente (navegador) y en el servidor. Ejemplo: Cross-site scripting, Inyección SQL. Una aplicación es vulnerable a ataques de inyección cuando:	• La aplicación no realiza validaciones de entrada y/o salida de información. • Uso del intérprete de programación para realizar las consultas dinámicas o las llamadas no parametrizadas. • Los datos hostiles se utilizan o concatenan directamente. El lenguaje SQL o de comandos puede incluir sentencias maliciosas que permitan acceder a información confidencial o almacenada.

Amenaza	DESCRIPCIÓN	RIESGOS
A04: 2021 - Diseño inseguro	Corresponde a todos los defectos generados en etapas de diseño e implementación de los controles lógicos que deberían asegurar el correcto funcionamiento de la aplicación. Ejemplo, permitir cantidades negativas en aplicaciones de carro de la compra, almacenar las credenciales en texto claro, etc.	• Diseño seguro. Cultura y metodología que evalúa constantemente las amenazas garantizando que el código esté diseñado y probado de manera sólida para prevenir métodos de ataque conocidos. El modelado de amenazas debe integrarse en los análisis, identificar cambios en los flujos de datos y el control de acceso u otros controles de seguridad • Ciclo de vida de desarrollo seguro. El software seguro requiere un ciclo de vida de desarrollo seguro, establecer patrones de diseño seguro, metodología de hardening, biblioteca de componentes seguros, herramientas y modelado de amenazas. Comuníquese con sus especialistas en seguridad al comienzo de un proyecto de software durante todo el proyecto y el mantenimiento de su software. Incluir Modelo de madurez de software (SAMM) de OWASP para ayudar a estructurar sus esfuerzos de desarrollo de software seguro.
A05: 2021 - Configuración incorrecta de seguridad.	Se refiere a los aspectos de la seguridad relevantes al contexto de la configuración más que del diseño e implementación, por ejemplo:	• Falta de aseguramiento adecuado, incluyendo permisos configurados incorrectamente en los servicios. • Funciones o servicios innecesarias habilitadas. • Las cuentas predeterminadas y sus contraseñas aún están habilitadas y sin cambios. • El manejo de errores es inadecuado, revela información de los servicios. • Funciones de seguridad deshabilitadas o no implementadas de forma segura.

Amenaza	DESCRIPCIÓN	RIESGOS
A06: 2021 - Componentes vulnerables y obsoletos.	Los componentes de los servicios y las aplicaciones deben ser parcheados periódicamente para minimizar los problemas de seguridad. A continuación, se listan las causas generales de vulnerabilidades:	• La configuración de seguridad en los servidores web de aplicaciones, no se establecen con valores seguros. • Las cabeceras no incluyen directivas de seguridad, o no están configurados para valores seguros. • El software está desactualizado o es vulnerable • Falta de un proceso de configuración de seguridad de aplicaciones coordinado y recurrente, los sistemas corren un mayor riesgo. • Desconocimiento de las versiones de todos los componentes que utiliza (tanto del lado del cliente como del lado del servidor). Esto incluye los componentes que usa directamente, así como las dependencias anidadas. • Uso de software vulnerable, incompatible o desactualizado, incluyendo el sistema operativo, el servidor web o de aplicaciones, el motor de bases de datos, aplicaciones, API y todos los componentes, los entornos de ejecución y las bibliotecas. • Falta de procesos de detección de vulnerabilidades periódicos. Es recomendable suscribirse a los boletines de seguridad relacionados con los componentes que utiliza. • Falta de esquemas de implementación constante de parches para los componentes de un software que permitan reparar o actualizar la plataforma y las dependencias de manera oportuna y basada en el riesgo, evitando exponer la seguridad durante días o meses de manera innecesaria.

Amenaza	DESCRIPCIÓN	RIESGOS
A07: 2021 - Fallos de identificación y autenticación.	Esta categoría identifica riesgos en el momento de la autenticación, adicionalmente involucra todo el ciclo de vida de la sesión del usuario como la confirmación de la identidad, la autenticación y la administración de sesiones del usuario siendo consideradas como fundamentales para proteger contra los ataques relacionados con la autenticación.	• Falta de pruebas la compatibilidad de las bibliotecas actualizadas, o parcheadas. • Falta de esquemas de aseguramiento de las configuraciones de los componentes Permite ataques automatizados como el relleno de credenciales, donde el atacante tiene una lista de nombres de usuario y contraseñas válidos. Permite la fuerza bruta u otros ataques automatizados. Permite contraseñas predeterminadas, débiles o conocidas, como "Password1" o "admin / admin". Utiliza procesos de recuperación de credenciales débiles o ineficaces y de olvido de contraseñas, como "respuestas basadas en el conocimiento", que no pueden protegerse. Utiliza almacenes de datos de contraseñas de texto sin formato, cifradas o con un hash débil. Tiene autenticación multifactor falta o ineficaz. Expone el identificador de sesión en la URL. Reutilice el identificador de sesión después de iniciar sesión correctamente. No invalida correctamente los ID de sesión. Las sesiones de usuario o los tokens de autenticación (principalmente los tokens de inicio de sesión único (SSO)) no se invalidan correctamente durante el cierre de sesión o un período de inactividad.

Amenaza	DESCRIPCIÓN	RIESGOS
A08: 2021 - Fallos de integridad de datos y software.	Esta categoría concierne a lo relacionado con la integridad y verificación de fuentes cuando instalamos, actualizamos o poseemos infraestructura relacionada como integraciones a servicios de terceros.	Los fallos en la integridad del software y los datos se relacionan directamente con el código y la infraestructura no protegidas contra las violaciones de la integridad.
A09: 2021 - Fallos de monitoreo y registro de seguridad.	La categoría cubre desde las ausencias de registro de eventos hasta su almacenamiento y gestión adecuada. Sin registro y monitoreo, las infracciones no se pueden detectar y pueden ocurrir en cualquier momento. Las fallas generales referente a este apartado son:	• Falta de registro de los eventos auditables, como inicios de sesión, inicios de sesión fallidos y transacciones críticas. • Las advertencias y los errores generan mensajes de registro inexistentes, inadecuados o poco claros. • Los registros de aplicaciones y API no se controlan para detectar actividades sospechosas. • Los registros solo se almacenan localmente. • Los umbrales de alerta apropiados y los procesos de escalamiento de la respuesta no están establecidos o no son efectivos. • Las pruebas de penetración y los escaneos mediante herramientas de prueba de seguridad de aplicaciones dinámicas no activan alertas. • La aplicación no puede detectar, escalar ni alertar sobre ataques activos en tiempo real o casi en tiempo real. • Falta de controles para prevenir la fuga de información permitiendo que los eventos de registro y alerta sean visibles para un usuario o un atacante.

Amenaza	DESCRIPCIÓN	RIESGOS
A10: 2021 - Falsificación de solicitudes del lado del servidor (SSRF).	Esta categoría expone una vulnerabilidad especifica que trata sobre el alto riesgo del descubrimiento de servicios que no son públicos como archivos o bases de datos e inclusive la ejecución de código arbitrario, ya que los desarrolladores y responsables de centros de cómputo generalmente consideran que los servicios internos son inaccesibles.	Las fallas de SSRF generalmente ocurren cuando las aplicaciones o servicios web tratan de acceder a un recurso remoto sin realizar las validaciones correspondientes permitiendo que un atacante manipule la aplicación para que envíe solicitudes a un destino inesperado, inclusive si está protegido por sistemas de seguridad como firewall, VPN u otro tipo de ACLs. La gravedad e impacto de SSRF es cada vez mayor debido a los servicios en la nube y la complejidad de sus arquitecturas.

Tabla 6 - OWASP Top 10 – 2021
Fuente: Realización Propia

Anexos B - FACTORES ANALISIS OWASP TOP 10 - 2021

Categoría	CWEs mapeados	Tasa de incidencia máxima	Tasa de incidencia media	Exploit ponderado promedio	Impacto ponderado medio	Cobertura máxima	Cobertura promedio	Incidencias totales	CVE totales
A01	34	**55,97%**	3,81%	6,92	5,93	**94,55%**	47,72%	**318,487**	19,013
A02	29	46,44%	4,49%	7.29	6,81	79,33%	34,85%	233,788	3,075
A03	33	19,09%	3,37%	7.25	7.15	94,04%	47,90%	274,228	**32.078**
A04	**40**	24,19%	3,00%	6,46	6,78	77,25%	42,51%	262,407	2.691
A05	20	19,84%	4,51%	**8.12**	6.56	89,58%	44,84%	208,387	789
A06	3	27,96%	**8,77%**	5.00	5.00	51,78%	22,47%	30,457	0
A07	22	14,84%	2,55%	7.4	6,50	79,51%	45,72%	132,195	3.897
A08:	10	16,67%	2,05%	6,94	**7,94**	75,04%	45,35%	47,972	1,152
A09	4	19,23%	6,51%	6,87	4,99	53,67%	39,97%	53,615	242
A10	1	2,72%	2,72%	8.28	6,72	67,72%	**67,72%**	9,503	385

Tabla 7 - Comparativa de Factores OWASP Top 10 - 2021

Anexos C - OBJETIVOS DE CONTROL OWASP

Dominio	Requerimiento	Objetivo de Control
V1	Arquitectura, Diseño y modelado de amenazas.	Ciclo de vida de desarrollo seguro
		Arquitectura de autenticación
		Arquitectura de gestión de sesiones
		Arquitectura de control de accesos
		Arquitectura de entrada y salida de datos
		Arquitectura Criptográfica
		Arquitectura de manejo de errores, logs y auditoría
		Arquitectura para protección de datos
		Arquitectura de comunicaciones
		Arquitectura de prevención de software malicioso
		Arquitectura de Lógica de negocio
		Arquitectura API
		Arquitectura de configuración
V2	Verificación de Autenticación.	Seguridad de contraseñas
		Autenticador general
		Ciclo de vida del Autenticador
		Almacenamiento de credenciales
		Recuperación de credenciales
		Búsqueda de verificador secreto
		Verificación fuera de banda
		Simple o Múltiple factor de autenticación
		Software y dispositivos criptográficos

Dominio	Requerimiento	Objetivo de Control
		Autenticación de servicios
V3	Verificación de manejo de sesiones.	Administración fundamental de sesiones.
		Enlaces de sesión
		Cierre de sesión y tiempo de espera
		Sesión basada en cookies
		Sesión basada en Token
		Re-autenticación desde federación o aserción
		Manejo de exploits y defensas contra la gestión de sesiones
V4	Verificación de control de accesos	Diseño general de control de accesos.
		Control de acceso a nivel de operación
		Otras consideraciones del control de accesos
V5	Verificación de Validación, Sanitización y Codificación.	Validación de entradas
		Desinfección y filtrado de caja de arena
		Codificación de salida y prevención de inyección de código
		Manejo de memoria, cadenas de texto y código no administrado
		Prevención de deserialización
V6	Verificación de Criptografía almacenada	Clasificación de datos
		Algoritmos
		Valores aleatorios
		Manejo de secretos
V7	Verificación de registro y manejo de errores	Logs de contenido
		Logs de procesamiento
		Logs de protección
		Manejo de errores

Dominio	Requerimiento	Objetivo de Control
V8	Verificación de protección de datos	Data General
		Data de Clientes internos
		Datos sensibles y privados
V9	Verificación de Comunicaciones	Requisitos de seguridad para comunicaciones de clientes
		Requisitos de seguridad para comunicaciones de servidores
V10	Verificación de código malicioso	Control de integridad en el código
		Búsqueda de código malicioso
		Controles de integridad en el despliegue de aplicaciones
V11	Verificación de Lógica de Negocio	Requisitos de seguridad para la lógica de negocio
V12	Verificación de Archivos y Recursos	Carga de archivos
		Integridad de archivos
		Ejecución de archivos
		Almacenamiento de archivos
		Descarga de archivos
		Protección SSRF
V13	Verificación de API y WebService	Verificación de seguridad para servicios web genéricos
		Verificación para servicios REST
		Verificación para servicios SOAP
		Verificación para GraphQL y otros webservice de la capa de datos
V14	Verificación de configuraciones	Versión de Build
		Dependencias
		Divulgación de seguridad no intencionada
		Cabeceras de seguridad HTTP

Tabla 8 - Objetivos de Control OWASP . Fuente: Realización Propia

Anexos D - PROCESO CWE DE MITRE

Según (The Mitre Corporation, 2021) CWE es una lista desarrollada por la comunidad de tipos de debilidades de software y hardware. Sirve como un lenguaje común, una vara de medir para las herramientas de seguridad y como una línea de base para los esfuerzos de identificación, mitigación y prevención de debilidades. Para acceder al contenido Mitre proporciona el sitio web **https://cwe.mitre.org** en el que se puede acceder al contenido clasificado por el punto de vista especifico clasifico en Vistas según la plataforma de desarrollo, diseño de hardware y bajo los conceptos de investigación.

A continuación, se realiza un breve resumen de las diferentes vistas, por su relevancia para el trabajo.

VISTA CWE: Plataforma de Desarrollo. El objetivo de esta vista es organizar las debilidades en torno a conceptos que se utilizan o se encuentran con frecuencia en el desarrollo de software. Esto incluye todos los aspectos del ciclo de vida del desarrollo de software, incluida la arquitectura y la implementación. Esta vista está dirigida a Desarrolladores de Software, arquitectos, diseñadores, codificadores, evaluadores para ayudarlos a comprender mejor los posibles errores que se pueden cometer en áreas específicas de su aplicación de software. (Mitre Corporation, 2021).

En la siguiente Ilustración, se muestran las relaciones en forma de árbol mostrando las diferentes debilidades que existen por cada plataforma de desarrollo de software.

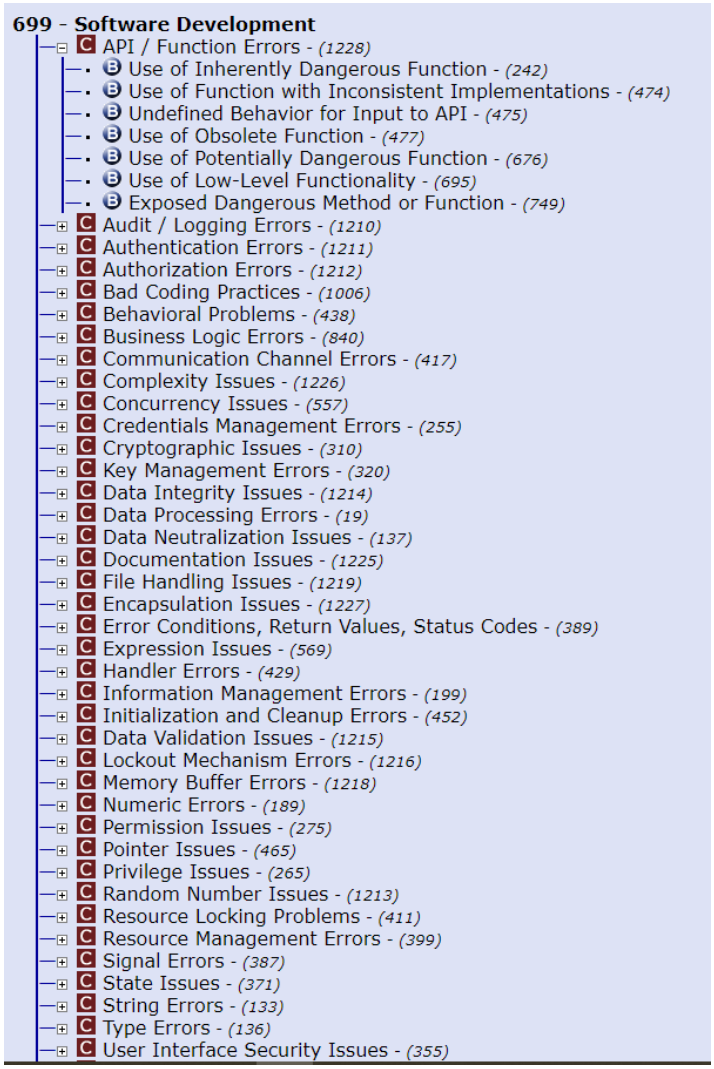

Ilustración 40 - Árbol CWE: Software Development.
Fuente https://cwe.mitre.org/cgi-bin/viewgen.cgi?id=699&qs=_0_0_0_0

VISTA CWE: Diseño de Hardware. El objetivo de esta vista organiza las debilidades en torno a conceptos que se utilizan o se encuentran con frecuencia en el diseño de hardware. En consecuencia, esta visión puede alinearse estrechamente con las perspectivas de diseñadores, fabricantes, educadores y proveedores de evaluaciones. Proporciona una variedad de categorías destinadas a simplificar la navegación, la navegación y la creación de mapas. (The Mitre Corporation, 2021).

En la siguiente imagen se muestran en forma de árbol las debilidades más comunes en el diseño de hardware

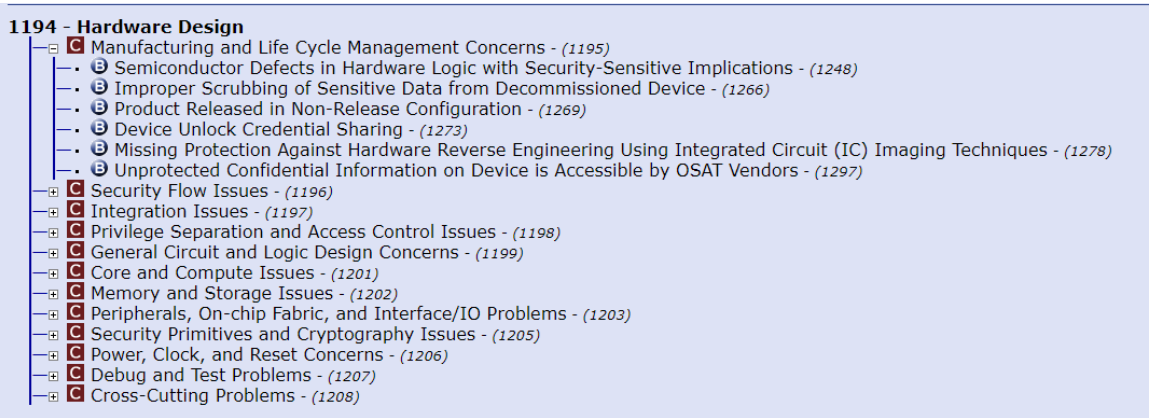

Ilustración 41 - Vista CWE Hardware Design
Fuente: https://cwe.mitre.org/data/definitions/1194.html

VISTA CWE: Conceptos de Investigación. Esta visión está destinada a facilitar la investigación de las debilidades, incluidas sus interdependencias, y puede aprovecharse para identificar sistemáticamente las lagunas teóricas dentro de CWE (The Mitre Corporation, 2021).

La vista está dirigida a investigadores académicos para identificar áreas potenciales de futuros análisis, analistas de vulnerabilidades para identificar debilidades relacionadas que podrían aprovecharse siguiendo las relaciones de sus codificaciones y para los proveedores de herramientas de evaluación, quienes podrán identificar debilidades adicionales que un análisis automatizado pueda detectar.

En el siguiente gráfico, se presenta el árbol que muestra cómo se relacionan las debilidades de acuerdo con el ámbito de investigación.

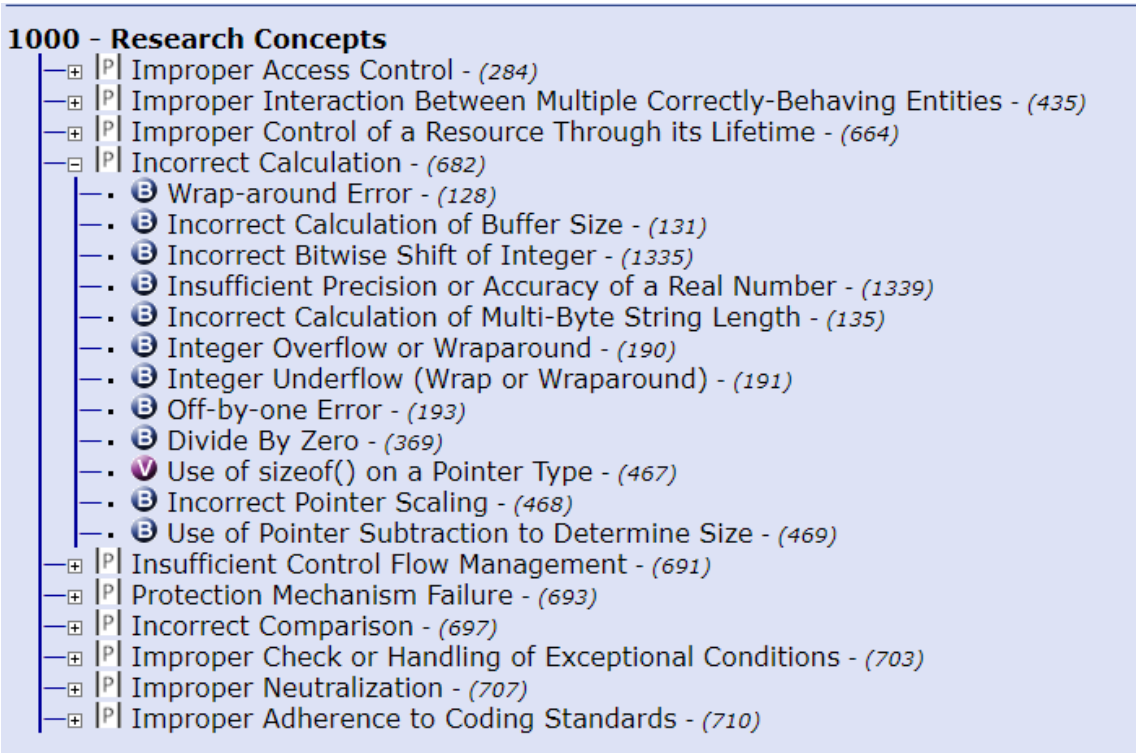

Ilustración 42 - Vista CWE Research Concepts
Fuente: https://cwe.mitre.org/data/definitions/1000.html

Los principales objetivos de CWE son:

- Proporcionar un lenguaje de uso común que permita describir los principales problemas defectos y debilidades de seguridad de software en la arquitectura, diseño y codificación.
- Estandarizar la comparación de herramientas de auditoría seguridad de software.
- Establecer una línea base para la identificación y documentación de vulnerabilidades, esquemas para la mitigación y medidas de mitigación.

Cada identificador CWE incluye los siguientes aspectos

- Nombre del tipo de vulnerabilidad
- Descripción del tipo
- Términos alternativos
- Descripción del comportamiento
- Descripción de la vulnerabilidad

- Probabilidad de explotar la vulnerabilidad
- Descripción de las consecuencias de la explotación
- Posibles mitigaciones
- Taxonomías de las fuentes
- Ejemplos de código para los lenguajes/arquitecturas
- Patrones de Ataque relacionados
- Referencias

Los principales factores de datos seleccionados por (OWASP Foundation, 2021) para cada una de las 10 categorías principales, son detallados a continuación:

- CWE mapeados: corresponde al número de CWE mapeados a una categoría por los investigadores de Top 10.
- Tasa de incidencia: Es el porcentaje de aplicaciones vulnerables del CWE que correspondientes a la muestra analizada.
- Exploit ponderado: la puntuación de severidad del Exploit obtenido de los CVSS asignadas a los CVE (Common Vulnerabilites and exposures) y CWE, en una escala de 10 puntos.
- Impacto ponderado: Puntaje obtenido de la medición de impacto de los CVSS asignados a las CVE y CWE, en escala de 10 puntos.
- Cobertura (de prueba): El porcentaje de aplicaciones analizadas por todas las organizaciones para un CWE especifico
- Total de ocurrencias: número total de aplicaciones que tienen los CWE asignados a una categoría.
- Total de CVE: número total de CVE en la base de datos NVD (National Vulnerability Database) que se asignaron a los CWE

Anexos E - EVALUACIÓN DE RIESGOS EN APLICACIONES WEB

OWASP se basa en un modelo estándar y alineado para evaluar riesgos de la seguridad de la información calculando de valoración de riesgo de la siguiente manera:

Riesgo = Probabilidad de Ocurrencia * Impacto

Los pasos para determinar y calcular la probabilidad y el impacto se detallan en las siguientes fases:

1. **Identificación de riesgos.**

Consiste en obtener información sobre las amenazas, ataques, vulnerabilidades que pueden ser explotadas, determinar si la amenaza es generada por múltiples orígenes e impacto que provoca a la organización.

2. **Factores de estimar la probabilidad.**

Una vez que se tiene toda la información de la amenaza es necesario estimar la probabilidad de que se materialice es decir que el o los atacantes logren explotar los fallos de seguridad. Para (Williams, 2021), en el nivel más alto, esta es una medida aproximada de la probabilidad de que un atacante descubra y explote esta vulnerabilidad en particular. No es necesario ser demasiado preciso en esta estimación. Generalmente, es suficiente identificar si la probabilidad es baja, media o alta, sin embargo, es recomendable siempre estimar en el peor de los escenarios para que los resultados de la evaluación sean lo más cercanos a la realidad posible, tomando en cuenta que cada factor tiene varios escenarios. Esta metodología propone asignar valores del 0 al 9 para determinar la probabilidad.

Tipo	Objetivo	Factores	Ponderación	Valor
Amenaza	Estimar la probabilidad de un ataque exitoso por parte de un grupo de amenazas.	Nivel de Habilidad: Conocimiento técnico	Sin habilidades técnicas	1
			Algunas habilidades técnicas	3
			Usuario avanzado	5
			Habilidades de programación y redes	7
			Habilidades de penetración de seguridad	9
		Motivo: Aspectos que promueven la detección y explotación de vulnerabilidades	Recompensa baja o nula	1
			Recompensa posible	4
			Recompensa Alta	9
		Oportunidad: Según los recursos utilizados	Acceso total o recursos costosos	0
			Acceso o recursos especiales	4
			Algún acceso o recursos	7
			No se requiere acceso o recursos	9
		Tamaño:	Desarrolladores	2
			Administradores de sistemas	2

Tipo	Objetivo	Factores	Ponderación	Valor
			Usuarios de intranet	4
			Socios	5
			Usuarios autenticados	6
			Usuarios de Internet anónimos	9
Vulnerabilidad	Estimar cuan probable es que la vulnerabilidad sea explotada. Los factores son:	Facilidad de descubrimiento	Prácticamente imposible	1
			Difícil	3
			Fácil	7
			Herramientas automatizadas disponibles	9
		Facilidad de explotación	Teórico	1
			Difícil	3
			Fácil	5
			Herramientas automatizadas disponibles	9
		Conciencia: En función de popularidad	Desconocido	1
			Oculto	4

Tipo	Objetivo	Factores	Ponderación	Valor
			Obvio	6
			Conocimiento público	9
		Detección de intrusiones	Detección activa en la aplicación	1
			Registrada y revisada	3
			Registrada sin revisión	8
			No registrada	9
Impacto Técnico	Determinar con mayor exactitud el éxito de un ataque en función del impacto a nivel técnico	Pérdida de confidencialidad: Se refiere a cuantificación de los datos que se podrían divulgarse y el nivel de criticidad.	Información mínima no confidencial divulgada	2
			información mínima crítica divulgada	6
			divulgación de datos no confidenciales extensos	6
			divulgación de datos críticos extensos	7
			divulgación de todos los datos	9
		Pérdida de integridad: Medir el nivel de corrupción y afectación de los datos	Datos mínimos levemente corruptos	1
			mínimos datos muy corruptos	3
			datos extensos ligeramente corruptos	5
			datos extensos muy corruptos	7

Tipo	Objetivo	Factores	Ponderación	Valor
			todos los datos totalmente corruptos	9
		Pérdida de disponibilidad: Determinar la importancia y proporción de la afectación al servicio.	Servicios secundarios mínimos interrumpidos	1
			servicios primarios mínimos interrumpidos	5
			servicios secundarios extensos interrumpidos	5
			servicios primarios extensos interrumpidos	7
			todos los servicios completamente perdidos	9
		Pérdida de responsabilidad: Evaluar si los datos son identificables y rastreables	Totalmente rastreable	1
			posiblemente rastreable	7
			completamente anónimo	9
Impacto **empresarial.**	Determinar con mayor exactitud el éxito de un ataque en función del impacto a nivel empresarial	Daño financiero: Perjuicios financieros	Menos que el costo de reparar la vulnerabilidad	1
			efecto menor en la ganancia anual	3
			efecto significativo en la ganancia anual	7
			quiebra	9
			Daño mínimo	1

Tipo	Objetivo	Factores	Ponderación	Valor
		Daño a la reputación: Perjuicio a la imagen de la empresa	pérdida de cuentas importantes	4
			pérdida de buena voluntad	5
			daño a la marca	9
		Incumplimiento: Valoración de la exposición	Violación menor	2
			violación clara	5
			violación de alto perfil	7
		Violación de la privacidad: Valoración en función de los usuarios afectados	Un individuo	3
			cientos de personas	5
			miles de personas	7
			millones de personas	9

Tabla 9 - Factores para estimar la probabilidad de una amenaza
Fuente: Realización Propia

1. Determinación de la gravedad del Riesgo

Se calcula la gravedad del riesgo en función de la estimación de la probabilidad y el impacto.

El siguiente cuadro referencia los rangos de 0-9 para medir el impacto medido en 3 niveles

Niveles de probabilidad e impacto	
0 a <3	BAJO
3 a <6	MEDIO
6 a 9	ELEVADO

Ilustración 43 - Severidad del impacto
Fuente: https://OWASP.org/www-community/OWASP_Risk_Rating_Methodology

Los métodos se detallan a continuación:

Método informal

Consiste en asignar los valores en función del conocimiento y experiencia del evaluador, sin embargo, este mecanismo puede acarrear consecuencias en los resultados si los valores asignados no están sujetos a la realidad.

Método repetible

Establecer un proceso formal de calificación de los que puede estar respaldado por herramientas automatizadas para facilitar el cálculo y obtención de resultados.

a. Se debe establecer los valores por cada factor de probabilidad ingresando la calificación de cada opción para luego calcular el promedio y obtener la probabilidad tal como se muestra en el siguiente grafico a modo de ejemplo:

'"Threat agent factors"'					'"Vulnerability factors"'			
Skill level	Motive	Opportunity	Size		Ease of discovery	Ease of exploit	Awareness	Intrusion detection
5	2	7	1		3	6	9	2
Overall likelihood=4.375 (MEDIUM)								

Ilustración 44 - Ejemplo de medición de Probabilidad
Fuente: https://OWASP.org/www-community/OWASP_Risk_Rating_Methodology

b. Determinar el impacto, de manera similar se registran las puntuaciones según el aspecto de cada factor para luego obtener mediante el promedio el valor del impacto.

Technical Impact					**Business Impact**			
Loss of confidentiality	Loss of integrity	Loss of availability	Loss of accountability		Financial damage	Reputation damage	Non-compliance	Privacy violation
9	7	5	8		1	2	1	5
Overall technical impact=7.25 (HIGH)					Overall business impact=2.25 (LOW)			

Ilustración 45 - Ejemplo de medición de Impacto
Fuente: https://OWASP.org/www-community/OWASP_Risk_Rating_Methodology

c. Por último, se debe crear una matriz de impacto vs probabilidad para obtener el nivel de severidad o riesgo inherente, tal como se muestra en la siguiente imagen

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

Ilustración 46 - Ejemplo de Matriz de Riesgos
Fuente: https://OWASP.org/www-community/OWASP_Risk_Rating_Methodology

El ejemplo se resume de la siguiente manera:

- Probabilidad: Media
- Impacto técnico: Alto
- Impacto empresarial: Bajo

Si analizamos técnicamente, se podría determinar que la gravedad del riesgo es alta, pero es importante considerar apropiadamente el impacto empresarial para poder tomar las mejores decisiones. El objetivo es evitar el impacto al negocio.

2. Toma de decisiones

En esta etapa se deben evaluar los resultados del análisis y establecer comparativas para determinar prioridades considerando que los riesgos más graves deben corregirse primero, y se deben incluir actividades para medir el retorno de inversión ROI de las decisiones tomadas. Ejemplo: Si la mitigación de un riesgo el muy costosa, y el impacto económico es mejor, el valor del ROI será muy alto.

3. Personalización del modelo de calificación de riesgo

Es recomendable aterrizar el marco de clasificación de riesgos según las necesidades del negocio, los aspectos más comunes para hacerlo son los siguientes:

a. **Sumando factores.** Consiste en agregar factores específicos en función del giro de negocio. Por ejemplo, una empresa bananera podría agregar factores de impacto relacionados con la producción permitiendo determinar la afectación al negocio.
b. **Opciones de personalización.** La efectividad del modelo dependerá de cuan ajustado a la realidad al giro de negocios se establecen los factores de riesgo comparando las calificaciones producidas por el modelo con las calificaciones producidas por un equipo de expertos permitiendo.
c. **Factores de ponderación.** Se puede establecer métricas para dar mayor ponderación a ciertos factores ajustando el modelo de evaluación de riesgos