Vol.10 No.2 (2026): Journal Scientific MQRInvestigar ISSN: 2588–0659
https://doi.org/10.56048/MQR.2026.e263

WebSocket Security: Vulnerabilities, Threats, and Theoretical Guidelines for Best Practices in Web Applications

Seguridad en WebSockets: vulnerabilidades, amenazas y lineamientos teóricos de buenas prácticas en aplicaciones web

Autores:

Zhunaula-Lucero, Michelle Andrea
UNIVERSIDAD CATÓLICA DE CUENCA
Unidad Académica de Informática, Ciencias de la Computación, e Innovación Tecnológica
Cuenca–Ecuador

michelle.zhunaula.75@est.ucacue.edu.ec

https://orcid.org/0009-0008-4710-6087

Quevedo-Sacoto, Andrés Sebastián
UNIVERSIDAD CATÓLICA DE CUENCA
Unidad Académica de Informática, Ciencias de la Computación, e Innovación Tecnológica
Cuenca –Ecuador

asquevedos@ucacue.edu.ec

https://orcid.org/0000-0001-5585-0270

Fechas de recepción: 07-MAR-2026 aceptación: 07-ABR-2026 publicación: 30-JUN-2026

https://orcid.org/0000-0002-8695-5005
http://mqrinvestigar.com/

Resumen

La presente investigación analiza las vulnerabilidades asociadas al protocolo WebSocket en aplicaciones web, considerando su creciente adopción en sectores críticos como salud, finanzas e IoT. A través de una metodología documental con enfoque cualitativo, se realizó una revisión sistemática de literatura científica y documentación técnica especializada con el fin de identificar amenazas recurrentes y mecanismos de mitigación reportados en implementaciones reales. El análisis permitió sistematizar las principales vulnerabilidades, entre ellas Cross-Site WebSocket Hijacking (CSWSH), ataques Man-in-the-Middle y escenarios de denegación de servicio, evidenciando que muchas de ellas derivan de configuraciones inseguras, ausencia de validación del encabezado Origin y mecanismos de autenticación insuficientes. Asimismo, se analizaron los controles de seguridad reportados en la literatura, como el uso de TLS, autenticación basada en tokens, validación estricta del origen y estrategias de monitoreo continuo, determinando que su efectividad depende de una implementación integral y no aislada. Como resultado, se estructuró un conjunto de lineamientos teóricos de buenas prácticas orientados a reducir la superficie de ataque en entornos WebSocket. La contribución del estudio radica en la sistematización crítica de riesgos y controles de seguridad, proporcionando un marco conceptual que facilita una implementación más segura y fundamentada del protocolo en aplicaciones web.

Palabras clave: WebSockets; Ciberseguridad; Vulnerabilidades

Abstract

This research analyzes the vulnerabilities associated with the WebSocket protocol in web applications, considering its growing adoption in critical sectors such as healthcare, finance, and IoT. Using a qualitative documentary methodology, a systematic review of scientific literature and specialized technical documentation was conducted to identify recurring threats and mitigation mechanisms reported in real implementations. The analysis allowed us to systematize the main vulnerabilities, including Cross-Site WebSocket Hijacking (CSWSH), Man-in-the-Middle attacks, and denial-of-service scenarios, showing that many of them stem from insecure configurations, lack of Origin header validation, and insufficient authentication mechanisms. Likewise, the security controls reported in the literature were analyzed, such as the use of TLS, token-based authentication, strict origin validation, and continuous monitoring strategies, determining that their effectiveness depends on comprehensive and not isolated implementation. As a result, a set of theoretical guidelines for best practices aimed at reducing the attack surface in WebSocket environments was structured. The contribution of the study lies in the critical systematization of risks and security controls, providing a conceptual framework that facilitates a more secure and informed implementation of the protocol in web applications.

Keywords: WebSockets; Cybersecurity; Vulnerabilities

Introducción

La evolución de las tecnologías de comunicación en la web ha transformado la forma en que los usuarios interactúan con servicios en línea (Álvarez, 2021). Desde los primeros sitios estáticos hasta las aplicaciones dinámicas actuales, los protocolos de comunicación han sido esenciales para brindar experiencias más fluidas y en tiempo real (Picado y Pérez, 2023). En este contexto, la estandarización del protocolo WebSocket por el IETF en la RFC 6455 (2011) marcó un avance decisivo al permitir conexiones bidireccionales y persistentes entre cliente y servidor, reduciendo la latencia y optimizando recursos (Johnson, 2024).

A diferencia del modelo de solicitud y respuesta de HTTP, los WebSockets mantienen un canal abierto que posibilita la comunicación continua, esta característica ha impulsado el desarrollo de aplicaciones que dependen de la inmediatez, como mensajería, videojuegos, sistemas financieros, telemedicina e infraestructuras del Internet de las Cosas (IoT) (Smith, 2025). Gracias a ello, el protocolo se ha consolidado como un elemento esencial en servicios que requieren rapidez y eficiencia.

La persistencia de las conexiones WebSocket, aunque favorece el rendimiento, incrementa el consumo de recursos y la complejidad en la gestión del estado, ampliando la superficie de ataque del sistema (Gourko, 2024). En entornos de comunicación en tiempo real, esta característica puede aumentar la exposición operativa y exigir controles de seguridad adicionales (Rakib, 2025), especialmente considerando que el protocolo WebSocket no incorpora mecanismos de seguridad completos por diseño (Raciti y Vitello, 2022).

Al carecer de mecanismos nativos de autenticación y autorización, la responsabilidad de establecer controles adecuados recae en la capa de aplicación (Jones, 2024). Esto ha derivado en vulnerabilidades como secuestro de sesiones, manipulación de mensajes y accesos no autorizados (Pandey y Kumar, 2024).

En sectores críticos como la salud, las finanzas o los sistemas gubernamentales, donde la confidencialidad y disponibilidad de los datos son vitales, estos riesgos son especialmente preocupantes (Awuor, 2023). Además, la ausencia de estándares claros genera una brecha entre la velocidad de adopción del protocolo y el nivel de protección real (Murley et al., 2021). Incluso controles básicos, como la validación del encabezado Origin, suelen aplicarse de forma deficiente, exponiendo las aplicaciones a ataques prevenibles (Ghasemshirazi y Heydarabadi, 2021).

Ante ello surge la necesidad de equilibrar la eficiencia de los WebSockets con la obligación de garantizar la seguridad de las aplicaciones críticas. De este contexto deriva la pregunta científica que guía el estudio: ¿cuáles son las principales vulnerabilidades y amenazas de seguridad en WebSockets y qué lineamientos teóricos de buenas prácticas pueden establecerse, a partir de la literatura existente, para su implementación segura?

La revisión del estado del arte evidencia que la investigación en torno a este protocolo ha crecido, aunque persisten vacíos importantes. Estudios recientes identifican tres ejes de

vulnerabilidad: el secuestro de sesión mediante Cross-Site WebSocket Hijacking (CSWSH), favorecido por controles inadecuados del encabezado Origin; los ataques Man-in-the-Middle (MitM), cuando no se usa cifrado TLS; y los ataques de denegación de servicio (DoS), asociados a la persistencia de múltiples conexiones (Subramanian, 2025). También se señalan configuraciones inseguras, como cookies no protegidas o ausencia de monitoreo en tiempo real (Flow, 2021).

Ante este panorama, algunos autores proponen un enfoque de seguridad por capas que incluya cifrado, control del handshake, validación de mensajes y supervisión continua (Bussey, 2020). Otros sugieren el uso de inteligencia artificial, modelos Zero Trust o tecnología blockchain para detectar anomalías y fortalecer la autenticación (González, 2025). No obstante, aunque la literatura identifica diversas vulnerabilidades y propone mecanismos de mitigación de forma dispersa, se observa la ausencia de una integración sistemática que articule dichas recomendaciones en un marco conceptual coherente. En respuesta a esta brecha, el presente estudio no solo sistematiza las vulnerabilidades, amenazas y estrategias de mitigación reportadas, sino que las analiza críticamente para derivar un conjunto estructurado de lineamientos teóricos de buenas prácticas orientados a una implementación segura del protocolo WebSocket. Su propósito general es analizar las vulnerabilidades y amenazas asociadas al protocolo WebSocket en aplicaciones web, examinando los mecanismos de mitigación reportados en la literatura, con el fin de establecer lineamientos teóricos de buenas prácticas que orienten su implementación segura.

La investigación se justifica por la necesidad de enfrentar los desafíos de seguridad que acompañan la rápida adopción del protocolo. A diferencia de HTTP o HTTPS, que cuentan con guías consolidadas, los WebSockets carecen de marcos normativos uniformes (Awuor, 2023). Esto provoca implementaciones basadas en experiencias aisladas y recomendaciones dispersas, lo que genera prácticas inseguras (Muhammad et al., 2024).

En consecuencia, la contribución de este estudio se delimita en dos ámbitos claramente definidos. En el ámbito académico, aporta una sistematización crítica de las vulnerabilidades y mecanismos de mitigación asociados al protocolo WebSocket, integrando aportes dispersos en la literatura en un marco conceptual coherente. En el ámbito teórico-metodológico, propone un conjunto estructurado de lineamientos de buenas prácticas derivados del análisis documental, que pueden servir como referencia para futuras investigaciones y desarrollos normativos.

En este sentido, el análisis realizado permite organizar y examinar de manera estructurada los principales riesgos de seguridad asociados al protocolo WebSocket, contribuyendo a una comprensión más sistemática del tema en la literatura especializada. Por ello, analizar sus vulnerabilidades y proponer buenas prácticas no solo es una contribución académica, sino una necesidad para garantizar su uso seguro en un entorno digital cada vez más interconectado.

Material y métodos

El estudio se desarrolla mediante una Revisión Sistemática de la Literatura (SLR), con enfoque cualitativo y analítico, orientada a examinar las vulnerabilidades, amenazas y mecanismos de mitigación asociados al protocolo WebSocket en aplicaciones web. La revisión se llevó a cabo siguiendo las directrices del modelo PRISMA con el propósito de garantizar transparencia, trazabilidad y reproducibilidad en el proceso de selección y análisis de fuentes.

1. **Estrategia de búsqueda:** La búsqueda bibliográfica se realizó en bases de datos científicas indexadas, incluyendo IEEE Xplore, Scopus, ACM Digital Library, SpringerLink y Google Scholar, considerando el periodo comprendido entre 2020 y 2026, con el fin de abarcar estudios recientes sobre la seguridad del protocolo WebSocket. Se emplearon combinaciones estructuradas de palabras clave en inglés, tales como “WebSocket security”, “WebSocket vulnerabilities”, “CSWSH”, “WebSocket attacks” y “WebSocket mitigation”, utilizando operadores booleanos (AND, OR) para refinar los resultados. Se aplicaron filtros por tipo de documento (artículos científicos, actas de congresos y documentación técnica revisada) e idioma (inglés y español), garantizando coherencia con los criterios establecidos en el protocolo de revisión sistemática.
2. **Criterios de selección:** Se incluyeron estudios académicos y documentos técnicos que abordaran explícitamente aspectos de seguridad en implementaciones de WebSockets. Se excluyeron trabajos duplicados, publicaciones sin revisión técnica o estudios que no trataran directamente vulnerabilidades o mecanismos de mitigación. Asimismo, se aplicó la técnica de bola de nieve (snowballing), tanto retrospectiva como prospectiva, para ampliar la identificación de literatura relevante.
3. **Extracción y análisis de datos:** Los estudios seleccionados fueron sometidos a un análisis cualitativo comparativo, orientado a identificar patrones recurrentes de vulnerabilidad, entre ellos Cross-Site WebSocket Hijacking (CSWSH), ataques Man-in-the-Middle (MitM) y Denegación de Servicio (DoS), así como configuraciones inseguras asociadas. Paralelamente, se examinaron los mecanismos de mitigación reportados, incluyendo controles técnicos (uso de TLS, validación del encabezado Origin, autenticación basada en tokens) y organizativos (monitoreo continuo y auditorías de seguridad).
4. **Síntesis y derivación de lineamientos:** A partir del análisis temático de la información recopilada, se realizó una síntesis conceptual que permitió agrupar las vulnerabilidades y sus respectivos mecanismos de defensa en categorías analíticas. Sobre esta base, se derivó un conjunto estructurado de lineamientos teóricos de buenas prácticas orientados a una implementación más segura del protocolo WebSocket.

El estudio es de carácter no experimental, dado que no implica manipulación de variables ni intervención en entornos reales. Se fundamenta exclusivamente en el análisis crítico de información secundaria. Se respetaron las consideraciones éticas relativas a la propiedad intelectual mediante la correcta citación y referencia de las fuentes conforme a la normativa APA (7.ª edición).

1. Antecedentes y estado del arte

1.1 Evolución del protocolo WebSocket y su relevancia en la web contemporánea

La evolución de las tecnologías web ha estado impulsada por la necesidad de soportar comunicaciones interactivas y en tiempo real, especialmente en aplicaciones multimedia donde la gestión eficiente de los flujos de datos es un aspecto clave (Grande, 2024). Los primeros modelos de interacción basados en HTTP respondían a una lógica estrictamente síncrona de solicitud–respuesta, lo que limitaba la capacidad de las aplicaciones para ofrecer interacciones dinámicas y continuas (Murley et al., 2021). Ante esta restricción, surgieron mecanismos alternativos como el polling y el long polling, los cuales, si bien permitieron una comunicación más frecuente, introdujeron una sobrecarga significativa en los servidores y un uso ineficiente del ancho de banda (Johnson, 2024).

En este contexto, la estandarización del protocolo WebSocket por el IETF en la RFC 6455 marcó un punto de inflexión en el diseño de aplicaciones web modernas (Adegbamigbe, 2021). WebSocket permitió establecer conexiones persistentes y bidireccionales sobre TCP, habilitando el intercambio continuo de datos entre cliente y servidor sin necesidad de renegociar la conexión en cada interacción (Ziemann, 2023). Esta característica técnica convirtió al protocolo en un componente clave para aplicaciones que requieren baja latencia y alta frecuencia de actualización, como plataformas de mensajería instantánea, videojuegos en línea, sistemas de monitoreo en tiempo real y soluciones colaborativas en la nube (Smith, 2025).

Diversos autores coinciden en que la adopción de WebSockets se ha acelerado de manera exponencial en los últimos años, impulsada por el auge de arquitecturas reactivas, microservicios y el Internet de las Cosas (IoT) (Avasthi et al., 2025). Sin embargo, esta rápida expansión no ha estado acompañada por un desarrollo proporcional de estándares de seguridad específicos, lo que ha generado un escenario donde el rendimiento y la eficiencia suelen priorizarse sobre la protección de la información (Awuor, 2023).

1.2 Diferencias estructurales entre HTTP y WebSocket desde la perspectiva de seguridad

Desde una perspectiva arquitectónica, WebSocket se diferencia de HTTP por su comunicación persistente y bidireccional, con implicaciones directas en la seguridad del canal (Paweł y Badurowicz, 2021). Mientras que HTTP se basa en conexiones efímeras y

estados transitorios, WebSocket mantiene un canal abierto durante períodos prolongados, permitiendo el intercambio continuo de mensajes (Tsoukalos, 2024). Esta persistencia, aunque ventajosa en términos de rendimiento, introduce nuevos desafíos relacionados con la gestión de sesiones, la autenticación y el control de acceso.

La literatura señala que HTTP ha evolucionado durante décadas incorporando mecanismos de seguridad consolidados, como cookies con atributos de protección, encabezados de seguridad y una integración madura con TLS (Jones, 2024). En contraste, la seguridad en WebSocket depende en gran medida de las decisiones de diseño adoptadas en la arquitectura de la aplicación, siendo habitual la incorporación de mecanismos externos como autenticación basada en tokens y cifrado mediante TLS para garantizar la confidencialidad y el control de acceso en comunicaciones persistentes (Rakib, 2025). Esta delegación incrementa la probabilidad de errores de implementación, especialmente en entornos donde los desarrolladores no cuentan con guías claras o estandarizadas (Ghasemshirazi y Heydarabadi, 2021).

Murley et al. (2021) destacan que esta diferencia estructural ha generado una brecha de seguridad significativa: mientras que las aplicaciones HTTP suelen beneficiarse de prácticas ampliamente documentadas, las implementaciones WebSocket dependen en gran medida de decisiones ad hoc, lo que conduce a configuraciones inconsistentes y, en muchos casos, vulnerables.

1.3 Superficie de ataque ampliada en conexiones persistentes

La literatura reciente destaca que los protocolos basados en conexiones persistentes, como WebSocket, amplían la superficie de ataque al mantener canales de comunicación activos durante periodos prolongados, incrementando la exposición a comportamientos maliciosos y abusos del canal (Tommasi et al., 2022).

Baloch (2024) sostiene que cada conexión activa representa un canal potencial de abuso si no se establecen límites adecuados de tiempo, concurrencia y validación de mensajes. A diferencia de HTTP, cuyo modelo de comunicación se basa en solicitudes independientes, la naturaleza persistente de WebSocket permite que un atacante que compromete una sesión mantenga acceso prolongado al sistema, incrementando el impacto potencial del ataque (Paweł y Badurowicz, 2021).

Flow (2021) documenta múltiples escenarios en los que configuraciones deficientes permiten a actores maliciosos explotar conexiones abiertas para enviar mensajes malformados, saturar recursos del servidor o manipular la lógica de negocio, estas amenazas se agravan cuando las aplicaciones no implementan mecanismos de monitoreo en tiempo real ni estrategias de cierre automático de sesiones sospechosas.

Asimismo, la persistencia de la conexión dificulta la detección temprana de comportamientos anómalos, ya que el tráfico WebSocket suele evadir controles tradicionales diseñados para tráfico HTTP (Łasocha y Badurowicz, 2021). Este factor ha sido señalado como una de las principales razones por las que los WebSockets se han convertido en un vector atractivo para ataques.

1.4 Vulnerabilidades recurrentes identificadas en la literatura

El análisis del estado del arte revela un conjunto de vulnerabilidades que se repiten de forma consistente en diferentes estudios. Entre las más relevantes se encuentran:

a) Cross-Site WebSocket Hijacking (CSWSH):

Pandey y Kumar (2024) describen este ataque como una extensión del Cross-Site Request Forgery (CSRF), en el cual un atacante aprovecha sesiones autenticadas y encabezados Origin mal configurados para establecer conexiones WebSocket no autorizadas. Ghasemshirazi y Heydarabadi (2021) enfatizan que la validación inadecuada del encabezado Origin constituye uno de los puntos más frágiles del protocolo, ya que muchos desarrolladores asumen erróneamente que este encabezado es confiable por defecto.

b) Ataques Man-in-the-Middle (MitM):

Los ataques MitM representan una amenaza constante para las conexiones WebSocket que no emplean el esquema seguro wss://, es decir, aquellas que no establecen la conexión sobre un canal TLS cifrado (Gourko, 2024). Muhammad et al. (2024) evidencian que la ausencia de cifrado TLS permite la interceptación y modificación de mensajes, comprometiendo la confidencialidad e integridad de la información transmitida.

c) Denegación de servicio (DoS):

Subramanian (2025) señala que la capacidad de mantener múltiples conexiones simultáneas facilita ataques DoS, especialmente en servidores que no implementan límites de concurrencia ni control de recursos. Este tipo de ataque resulta particularmente crítico en aplicaciones que dependen de la disponibilidad continua del servicio.

d) Deficiencias en autenticación y autorización:

Jones (2024) destaca que la falta de mecanismos nativos de autenticación en WebSocket conduce a enfoques heterogéneos, muchos de los cuales carecen de controles adecuados de expiración, revocación y validación de tokens.

1.5 Casos documentados y evidencia empírica

La literatura no solo aborda vulnerabilidades desde una perspectiva teórica, sino que también documenta incidentes reales derivados de implementaciones inseguras. Muhammad et al. (2024) analizan el caso de la plataforma Keepa.com, donde la combinación de encabezados mal validados y controles de autenticación débiles permitió a atacantes evadir restricciones y acceder a información sensible.

Honbo et al. (2024) demuestran que herramientas de análisis de tráfico y proxies pueden explotar fácilmente estas debilidades cuando no se aplican controles estrictos durante el handshake y la comunicación posterior. Estos casos refuerzan la idea de que las vulnerabilidades en WebSocket no son meramente hipotéticas, sino que tienen consecuencias tangibles en entornos productivos.

1.6 Contexto regional: América Latina y Ecuador

En América Latina, la adopción acelerada de tecnologías digitales ha ido acompañada de un incremento sostenido de incidentes de ciberseguridad (Batista, 2021). Informes del Banco Interamericano de Desarrollo (BID, 2020) y de ESET (2024) indican que la región enfrenta desafíos significativos relacionados con la madurez de sus capacidades de defensa, la capacitación técnica y la formulación de políticas públicas en ciberseguridad.

En el caso ecuatoriano, estudios recientes evidencian deficiencias en la seguridad de aplicaciones web, particularmente en pequeñas y medianas empresas, donde la falta de recursos y conocimiento técnico limita la adopción de buenas prácticas (Lucio y Campaña, 2024). Diagnósticos institucionales también han señalado debilidades en la implementación de herramientas OWASP y en la gestión de riesgos asociados a protocolos emergentes (Ministerio de Telecomunicaciones y de la Sociedad de la información, 2022).

Estos hallazgos sugieren que las vulnerabilidades de WebSocket adquieren una dimensión aún más crítica en contextos donde las capacidades técnicas y regulatorias son limitadas, lo que refuerza la necesidad de propuestas teóricas adaptables al entorno regional.

1.7 Ausencia de estándares y vacíos en la investigación

A pesar de su adopción generalizada, la implementación de WebSocket carece de un estándar unificado, como lo evidencia la diversidad de enfoques arquitectónicos (puros, híbridos o basados en librerías intermediarias) y la ausencia de soporte nativo en frameworks ampliamente utilizados (Raciti y Vitello, 2022). Awuor (2023) sostiene que la responsabilidad de la seguridad recae casi por completo en los desarrolladores, sin una guía universal que integre las mejores prácticas de manera coherente.

Si bien existen recomendaciones dispersas en documentos técnicos y estudios académicos, estas suelen abordarse de forma fragmentada, sin un marco teórico integrador que articule vulnerabilidades, amenazas y mecanismos de mitigación (Murley et al., 2021). Este vacío lleva a las organizaciones a desarrollar proyectos de modernización específicos y complejos para lograr escalabilidad y seguridad, tal como se documenta en estudios de migración de aplicaciones web (Gourko, 2024).

Estudios recientes que analizan aplicaciones basadas en WebSockets evidencian que muchas de las decisiones relacionadas con su diseño y operación segura se abordan de manera contextual y específica al dominio, sin apoyarse en marcos teóricos unificados que puedan generalizarse a otros escenarios (Faishal y Sutopo, 2025). En este sentido, el presente estudio se posiciona como un aporte orientado a cerrar dicha brecha, proporcionando una base conceptual para la seguridad en WebSockets.

2. Análisis de vulnerabilidades y amenazas en WebSockets

2.1 Consideraciones generales sobre el modelo de amenazas en WebSockets

El análisis de seguridad de WebSocket requiere un enfoque distinto al aplicado tradicionalmente a HTTP, debido a las particularidades de su modelo de comunicación (Tommasi et al., 2022). La persistencia del canal, la bidireccionalidad de la comunicación y

la ausencia de mecanismos nativos de autenticación generan un modelo de amenazas más complejo, donde los riesgos no se limitan al momento inicial de la conexión, sino que se extienden durante todo el ciclo de vida de la sesión (Baloch, 2024).

Desde esta perspectiva, la literatura coincide en que los WebSockets deben evaluarse como canales activos y prolongados, susceptibles de ser explotados tanto a nivel técnico como organizativo (Awuor, 2023). El modelo de amenazas incluye no solo ataques externos, sino también errores de configuración, malas prácticas de desarrollo y deficiencias en los procesos de monitoreo y respuesta a incidentes (Flow, 2021).

2.2 Cross-Site WebSocket Hijacking (CSWSH)

La literatura sobre seguridad en WebSocket identifica el Cross-Site WebSocket Hijacking (CSWSH) como una amenaza relevante derivada de la falta de mecanismos nativos de autenticación y de una validación insuficiente del origen de la conexión (Raciti y Vitello, 2022). Este ataque se produce cuando un atacante logra establecer una conexión WebSocket no autorizada desde un origen distinto al legítimo, aprovechando sesiones previamente autenticadas y una validación inadecuada del encabezado Origin (Pandey y Kumar, 2024).

Ghasemshirazi y Heydarabadi (2021) señalan que uno de los errores más comunes en implementaciones reales es asumir que el encabezado Origin es confiable por defecto, en realidad, dicho encabezado puede ser manipulado o enviado desde contextos maliciosos si el servidor no implementa controles estrictos. Esta debilidad convierte a CSWSH en un ataque particularmente peligroso, ya que permite reutilizar credenciales válidas sin necesidad de comprometer directamente al usuario (Murley, 2023).

Desde una perspectiva técnica, el impacto de CSWSH puede variar desde la ejecución de acciones no autorizadas hasta la exfiltración de datos sensibles, dependiendo de la lógica implementada en la aplicación (Jones, 2024). En este tipo de aplicaciones críticas en tiempo real, una conexión WebSocket no autorizada puede comprometer la integridad de los procesos, permitiendo la ejecución de acciones indebidas o la alteración de información transmitida de manera continua, lo que incrementa significativamente el riesgo operativo del sistema (Cahyo, 2025).

La literatura propone diversas estrategias de mitigación, entre ellas la validación estricta del encabezado Origin, el uso de tokens antifalsificación (CSRF tokens) y la autenticación explícita durante el handshake inicial (Pandey y Kumar, 2024). No obstante, varios autores advierten que estas medidas suelen aplicarse de forma parcial o incorrecta, reduciendo su efectividad (Ahmed et al., 2024).

2.3 Ataques Man-in-the-Middle (MitM)

Las comunicaciones WebSocket sin mecanismos de cifrado robustos exponen a las aplicaciones en tiempo real a ataques de intermediación, especialmente debido a la naturaleza continua y bidireccional del intercambio de datos (Kumar, 2024). A diferencia de HTTP, donde el uso de HTTPS está ampliamente difundido, aún existen implementaciones

WebSocket que operan sobre ws:// en lugar de wss://, especialmente en entornos de desarrollo o sistemas heredados (Muhammad et al., 2024).

En este escenario, un atacante puede intervenir el canal activo y manipular el flujo de información durante toda la sesión, incrementando el riesgo de compromiso de datos y de la lógica de la aplicación (Kumar, 2024). Este riesgo se ve amplificado en WebSockets debido a la transmisión continua de datos, lo que permite al atacante observar patrones de comunicación y explotar información sensible a lo largo del tiempo (Subramanian, 2025).

La literatura coincide en que el uso obligatorio de TLS es la principal medida de mitigación frente a este tipo de ataques (Johnson, 2025). Sin embargo, Dynowski y Dulak (2025) advierten que el simple uso de TLS no es suficiente si la configuración del servidor es débil, por ejemplo, mediante el uso de certificados autofirmados, algoritmos obsoletos o una gestión inadecuada de claves.

Desde una perspectiva crítica, algunos estudios señalan que la falsa percepción de seguridad asociada al cifrado puede generar una relajación en otros controles, como la validación de mensajes o la autenticación robusta (Sathya y Swetha, 2025). Esto refuerza la idea de que la seguridad en WebSockets debe abordarse de forma integral.

2.4 Ataques de denegación de servicio (DoS)

Los ataques de denegación de servicio constituyen una de las amenazas más relevantes para aplicaciones que utilizan WebSockets, debido a la capacidad del protocolo de mantener múltiples conexiones abiertas de manera simultánea (Raciti y Vitello, 2022). Subramanian (2025) explica que un atacante puede explotar esta característica abriendo un gran número de conexiones persistentes o enviando mensajes de forma intensiva, saturando los recursos del servidor.

A diferencia de los ataques DoS tradicionales basados en HTTP, los ataques contra WebSockets pueden ser más difíciles de detectar, ya que el tráfico malicioso se mezcla con comunicaciones legítimas y mantiene un estado activo durante largos periodos (Baloch, 2024). Esta situación se agrava cuando las aplicaciones no implementan límites claros de concurrencia ni mecanismos de cierre automático de sesiones inactivas o sospechosas.

La literatura propone estrategias como el rate limiting, la definición de cuotas por usuario y el monitoreo del comportamiento de las conexiones en tiempo real (Manglik, 2023). No obstante, varios autores advierten que estas medidas pueden afectar negativamente el rendimiento si no se diseñan cuidadosamente, lo que plantea un desafío adicional para los desarrolladores (Bux et al., 2021).

2.5 Vulnerabilidades en autenticación y autorización

Una de las debilidades estructurales más relevantes de WebSocket es la ausencia de mecanismos nativos de autenticación y autorización (Raciti y Vitello, 2022). Esta carencia obliga a las aplicaciones a implementar soluciones personalizadas, lo que da lugar a enfoques heterogéneos y, en muchos casos, inseguros (Jones, 2024).

La literatura documenta errores frecuentes como el uso de cookies sin atributos de seguridad, tokens de larga duración sin mecanismos de revocación y la falta de validación del estado de autenticación durante toda la sesión WebSocket (Johnson, 2025). Estos errores facilitan ataques de reutilización de sesiones y escalamiento de privilegios.

Desde una perspectiva comparativa, los estudios coinciden en que los mecanismos basados en tokens, como JWT u OAuth2, ofrecen un mayor nivel de control y trazabilidad, siempre que se implementen correctamente (Pandey y Kumar, 2024). Sin embargo, su efectividad depende de una gestión adecuada del ciclo de vida de los tokens, aspecto que suele ser descuidado en implementaciones reales (Neru y Yudertha, 2020).

2.6 Errores de configuración y malas prácticas de desarrollo

Además de las debilidades propias del protocolo, los principales riesgos asociados al uso de WebSockets suelen originarse en configuraciones inadecuadas y en prácticas de desarrollo deficientes, que incrementan la superficie de ataque y facilitan la explotación de los sistemas (Chowdary, 2021).

Flow (2021) identifica configuraciones comunes como la exposición innecesaria de endpoints WebSocket, la falta de autenticación en entornos productivos y la ausencia de validación de mensajes entrantes.

Estos errores suelen estar relacionados con la presión por reducir tiempos de desarrollo y la falta de capacitación especializada en seguridad (Lucio y Campaña, 2024). En este sentido, la literatura resalta que muchas vulnerabilidades no son consecuencia de limitaciones técnicas, sino de decisiones organizativas y culturales que subestiman la importancia de la seguridad (Awuor, 2023).

2.7 Factores organizativos y humanos como amplificadores del riesgo

El análisis de amenazas en WebSockets no puede limitarse a aspectos técnicos. Diversos estudios destacan que factores organizativos, como la ausencia de políticas de seguridad, la falta de auditorías periódicas y la escasa formación del personal, amplifican significativamente los riesgos asociados al protocolo (Dementyev, 2023).

En entornos donde la seguridad no se integra desde las fases iniciales del diseño, las vulnerabilidades tienden a acumularse y a manifestarse en etapas avanzadas del ciclo de vida del sistema, cuando su corrección resulta más costosa y compleja (Bussey, 2020). Este fenómeno refuerza la necesidad de adoptar enfoques proactivos y sistemáticos.

2.8 Síntesis crítica del análisis de vulnerabilidades

El análisis profundo de las vulnerabilidades y amenazas en WebSockets evidencia que los riesgos asociados al protocolo son el resultado de una combinación de factores técnicos, organizativos y contextuales. Ataques como CSWSH, MitM y DoS no surgen únicamente por deficiencias del protocolo, sino por implementaciones que no consideran adecuadamente su modelo de amenazas (Murley et al., 2021).

La literatura coincide en que no existe una solución única para mitigar estos riesgos; por el contrario, se requiere la integración de múltiples controles que aborden la seguridad desde diferentes capas (Dementyev, 2023). Esta conclusión sienta las bases para el análisis de marcos, herramientas y estrategias de mitigación, que se abordará en la siguiente sección del desarrollo.

3. Marcos, herramientas y estrategias de mitigación en WebSockets: efectividad y limitaciones

3.1 Enfoques generales para la mitigación de riesgos en WebSockets

La literatura especializada coincide en que la mitigación de vulnerabilidades en WebSockets no puede abordarse mediante una única técnica o herramienta, la naturaleza persistente del protocolo, junto con la diversidad de amenazas identificadas, exige la adopción de enfoques integrales que combinen controles técnicos, organizativos y procedimentales (Bussey, 2020). En este sentido, los marcos de mitigación más efectivos se fundamentan en el principio de defensa en profundidad, donde múltiples capas de protección actúan de forma complementaria para reducir la probabilidad y el impacto de los ataques (Dementyev, 2023). Este enfoque reconoce que ningún mecanismo es infalible por sí mismo. Por ejemplo, el uso de cifrado TLS protege la confidencialidad de la información, pero no previene ataques derivados de una autenticación deficiente o de una validación inadecuada de los mensajes (Johnson, 2025). Por ello, la literatura propone integrar distintos controles a lo largo de todo el ciclo de vida de la conexión WebSocket, desde el handshake inicial hasta la finalización de la sesión (Raciti y Vitello, 2022).

3.2 Uso de TLS y el esquema wss://

El uso del esquema seguro wss://, basado en Transport Layer Security (TLS), constituye una de las medidas más fundamentales para proteger las comunicaciones WebSocket frente a ataques de tipo Man-in-the-Middle (MitM) (Gourko, 2024). Diversos estudios coinciden en que la adopción obligatoria de TLS es un requisito mínimo para cualquier implementación segura del protocolo (Pandey y Kumar, 2024; Muhammad et al., 2024).

La efectividad de TLS radica en su capacidad para cifrar los datos transmitidos y autenticar al servidor mediante certificados digitales confiables (Gourko, 2024). No obstante, Dynowski y Dulak (2025) advierten que la seguridad proporcionada por TLS depende en gran medida de una configuración adecuada, por ende, el uso de versiones obsoletas del protocolo, algoritmos criptográficos débiles o certificados autofirmados puede comprometer seriamente la protección esperada.

Entre las limitaciones identificadas, la literatura señala que TLS no protege contra ataques que explotan vulnerabilidades en la lógica de la aplicación, como CSWSH o errores de autorización (Awuor, 2023). Además, la implementación incorrecta de TLS puede generar una falsa sensación de seguridad, llevando a los desarrolladores a descuidar otros controles críticos (Paweł y Badurowicz, 2021).

3.3 Autenticación basada en tokens: JWT, OAuth2 y OpenID Connect

Ante la ausencia de mecanismos nativos de autenticación en WebSocket, los enfoques basados en tokens se han consolidado como una de las estrategias más recomendadas para la gestión de identidad y control de acceso (Ghasemshirazi y Heydarabadi, 2021). El uso de JSON Web Tokens (JWT), OAuth2 y OpenID Connect permite desacoplar la autenticación del canal de comunicación y aplicar controles más flexibles y escalables (Jones, 2024).

La literatura destaca que estos mecanismos facilitan la implementación de autenticación sin estado, reduciendo la dependencia de cookies y mitigando riesgos asociados al secuestro de sesiones (Pandey y Kumar, 2024). Además, los tokens permiten incorporar información contextual, como roles y permisos, lo que contribuye a una autorización más granular (Ghasemshirazi y Heydarabadi, 2021).

Sin embargo, varios estudios advierten que la efectividad de los tokens depende de una gestión rigurosa de su ciclo de vida (Giovanni, 2020). El uso de tokens de larga duración sin mecanismos de revocación incrementa el riesgo de abuso en caso de compromiso (Johnson, 2025). Asimismo, la exposición de tokens en clientes inseguros o su transmisión sin cifrado anula gran parte de sus beneficios (Oktavia et al., 2020).

3.4 Validación del encabezado Origin y protección contra CSWSH

La validación del encabezado Origin constituye un mecanismo de control inicial frente a ataques Cross-Site WebSocket Hijacking (CSWSH); sin embargo, la literatura advierte que su efectividad es limitada, ya que dicho encabezado tiene un carácter meramente informativo y no representa una prueba robusta de autenticación (Raciti y Vitello, 2022).

Ghasemshirazi y Heydarabadi (2021) subrayan que esta validación debe realizarse siempre del lado del servidor y basarse en listas explícitas de orígenes permitidos, evitando enfoques permisivos o dependientes del cliente.

De forma complementaria, diversos estudios destacan la necesidad de incorporar mecanismos de autenticación explícita durante el handshake inicial, así como proteger dicho proceso frente a ataques de tipo CSRF y Cross-Site WebSocket Hijacking, dado que estas capacidades no están contempladas de manera nativa en el protocolo WebSocket (Raciti y Vitello, 2022). Estas medidas refuerzan la protección al exigir que cada conexión WebSocket demuestre su legitimidad más allá de la simple presencia de una sesión activa.

No obstante, varios autores señalan que la validación del encabezado Origin presenta limitaciones prácticas, especialmente en aplicaciones distribuidas que interactúan con múltiples dominios o servicios de terceros (Awuor, 2023). En estos escenarios, una configuración incorrecta puede provocar falsos positivos o, por el contrario, abrir la puerta a ataques si se adoptan listas demasiado amplias (Khodayari et al., 2024).

3.5 Gestión de sesiones y control del ciclo de vida de la conexión

La naturaleza persistente de las conexiones WebSocket introduce desafíos específicos en la gestión de sesiones, ya que una administración inadecuada del ciclo de vida de la conexión

puede derivar en un uso excesivo de recursos, afectar la disponibilidad del servicio y ampliar la superficie de ataque frente a abusos o comportamientos maliciosos (Chowdary, 2021).
La literatura recomienda definir tiempos de vida claros para las conexiones, implementar mecanismos de expiración y renovación de credenciales, y cerrar automáticamente sesiones inactivas o sospechosas (Manglik, 2023).
Estos controles permiten reducir la ventana de oportunidad para los atacantes y limitar el impacto de una posible intrusión. Sin embargo, Subramanian (2025) advierte que una gestión demasiado restrictiva puede afectar la experiencia del usuario y el rendimiento del sistema, especialmente en aplicaciones que requieren conexiones persistentes prolongadas.
Este dilema evidencia una de las tensiones recurrentes en la seguridad de WebSockets: el equilibrio entre disponibilidad y protección, la literatura sugiere que este equilibrio debe evaluarse en función del contexto y la criticidad del sistema, adoptando políticas diferenciadas según el perfil de riesgo (Obonguko, 2025).

3.6 Mitigación de ataques de denegación de servicio (DoS)

Para enfrentar los ataques DoS dirigidos a WebSockets, los estudios proponen una combinación de técnicas que incluyen rate limiting, cuotas de uso por cliente, control de concurrencia y monitoreo del comportamiento de las conexiones (Subramanian, 2025). Estas estrategias buscan identificar patrones anómalos y limitar el consumo excesivo de recursos.
La efectividad de estas medidas depende en gran medida de su correcta implementación y ajuste. Baloch (2024) señala que configuraciones demasiado permisivas pueden resultar ineficaces, mientras que restricciones excesivas pueden degradar el servicio para usuarios legítimos. Además, la implementación de mecanismos de mitigación DoS suele requerir infraestructura adicional, como balanceadores de carga o sistemas de detección de intrusiones, lo que puede representar una barrera para organizaciones con recursos limitados (Lucio y Campaña, 2024).

3.7 Validación y sanitización de mensajes

El control estricto del formato y contenido de los mensajes transmitidos mediante WebSockets resulta fundamental para preservar la integridad de la aplicación, ya que permite detectar entradas inesperadas y prevenir la ejecución de datos malformados o manipulados durante la comunicación bidireccional (Chowdary, 2021).
Johnson (2025) destaca la importancia de definir esquemas de datos estrictos y validar cada mensaje recibido antes de procesarlo.
Esta práctica reduce significativamente el riesgo de que datos maliciosos afecten la lógica de la aplicación. Sin embargo, su implementación puede introducir sobrecarga computacional y complejidad adicional en el desarrollo, especialmente en aplicaciones de alta frecuencia de mensajes (Delgado, 2025).
Diversos estudios indican que el impacto en el rendimiento asociado a la validación de mensajes puede reducirse mediante el empleo de arquitecturas optimizadas y herramientas

especializadas, las cuales permiten equilibrar la eficiencia del procesamiento con los requisitos de seguridad en sistemas de comunicación en tiempo real (Chowdary, 2021).

3.8 Monitoreo, registro y detección de anomalías

La implementación de medidas como validación de entradas y limitación de tasa en WebSockets forma parte esencial de un monitoreo continuo para la detección temprana de incidentes de seguridad (Raciti y Vitello, 2022). Dementyev (2023) propone integrar métricas, registros y alertas que permitan visualizar el comportamiento de las conexiones en tiempo real y responder rápidamente ante actividades sospechosas.

En años recientes, algunos estudios han explorado el uso de técnicas de inteligencia artificial y aprendizaje automático para detectar anomalías en el tráfico WebSocket (González, 2025). Estos enfoques resultan ideales en sistemas híbridos WebSocket-REST, donde se prioriza la baja latencia para notificaciones, paneles en vivo y streaming de datos en tiempo real (Chowdary, 2021).

En este contexto, el uso de WebSockets como canal de comunicación bidireccional facilita la monitorización continua y la retroalimentación inmediata, permitiendo que los modelos basados en IA analicen flujos de datos en vivo y optimicen la respuesta del sistema sin comprometer su desempeño operativo (Kayatas, 2025).

3.9 Limitaciones de los marcos y herramientas actuales

Si bien se han propuesto múltiples enfoques y herramientas para fortalecer la seguridad en aplicaciones basadas en WebSockets, los estudios revisados evidencian que su efectividad depende en gran medida del modelo arquitectónico adoptado y del contexto operativo en el que se implementan, lo que dificulta la aplicación de mecanismos de protección homogéneos en todos los escenarios (Chowdary, 2021).

Muchas de las medidas analizadas dependen de configuraciones específicas y del contexto de uso, lo que limita su generalización (Awuor, 2023).

Asimismo, varios autores señalan que la fragmentación de recomendaciones y la ausencia de estándares unificados dificultan la adopción de enfoques sistemáticos (Murley et al., 2021). Esta situación refuerza la necesidad de integrar las estrategias de mitigación en un marco teórico coherente que oriente a desarrolladores y responsables de seguridad.

3.10 Síntesis del análisis de mitigación

El análisis de los marcos, herramientas y estrategias de mitigación evidencia que la seguridad en WebSockets requiere una combinación equilibrada de controles técnicos y organizativos. Si bien existen mecanismos efectivos para reducir riesgos específicos, su implementación aislada resulta insuficiente para abordar la complejidad del modelo de amenazas del protocolo (Bussey, 2020).

Esta síntesis sienta las bases para la formulación de lineamientos teóricos de buenas prácticas, que integren las recomendaciones más sólidas de la literatura y consideren tanto su

efectividad como sus limitaciones. Dichos lineamientos serán desarrollados en la siguiente sección del artículo.

4. Síntesis teórica y formulación de lineamientos de buenas prácticas para WebSockets

4.1 Integración de hallazgos del estado del arte

El análisis sistemático de la literatura evidencia que la seguridad en WebSockets constituye un campo en evolución, caracterizado por una rápida adopción tecnológica y una maduración incompleta de sus mecanismos de protección. Los estudios revisados coinciden en que las vulnerabilidades más críticas (CSWSH, ataques Man-in-the-Middle y denegación de servicio) no derivan exclusivamente de fallas del protocolo en sí, sino de su implementación en contextos reales donde predominan configuraciones inseguras, controles parciales y ausencia de lineamientos uniformes (Murley et al., 2021; Awuor, 2023).

Desde una perspectiva teórica, se observa que la mayoría de los enfoques propuestos abordan problemas específicos de forma aislada, sin integrar los distintos niveles del modelo de amenazas. Esta fragmentación limita la efectividad global de las medidas de mitigación y dificulta su adopción sistemática por parte de desarrolladores y responsables de seguridad (Bussey, 2020). En consecuencia, la síntesis de la literatura pone de manifiesto la necesidad de articular un conjunto coherente de principios que orienten la implementación segura de WebSockets más allá de soluciones puntuales.

4.2 Principios teóricos para la seguridad en WebSockets

La revisión de los marcos conceptuales y técnicos permite identificar una serie de principios transversales que sustentan las prácticas más efectivas en materia de seguridad.

En primer lugar, el principio de defensa en profundidad se consolida como eje central, al reconocer que ninguna medida individual es suficiente para proteger un canal de comunicación persistente y bidireccional como WebSocket (Dementyev, 2023).

En segundo lugar, emerge el principio de mínimo privilegio, aplicado tanto a la autenticación como a la autorización; los estudios destacan la importancia de limitar el alcance de las credenciales utilizadas en las conexiones WebSocket, reduciendo la superficie de impacto ante un posible compromiso (Pandey y Kumar, 2024). Este principio se complementa con la validación continua y automatizada de los controles de seguridad, gracias a la integración de escaneos estáticos y dinámicos en pipelines DevSecOps durante el desarrollo y despliegue de aplicaciones WebSocket (Interrante, 2025). En arquitecturas donde la infraestructura es administrada por el proveedor de la plataforma, esta práctica permite detectar vulnerabilidades a nivel de aplicación, reforzando la seguridad del canal de comunicación bidireccional (Ahlawat, 2023).

Finalmente, la literatura resalta la necesidad de incorporar el principio de seguridad por diseño, integrando los controles desde las etapas iniciales del desarrollo y evitando enfoques reactivos basados únicamente en la corrección de incidentes (Hilbing et al., 2023).

4.3 Lineamientos teóricos de buenas prácticas

A partir de la síntesis realizada, se proponen los siguientes lineamientos teóricos de buenas prácticas para la implementación segura de WebSockets en aplicaciones web:

- Uso obligatorio de cifrado TLS: Todas las comunicaciones WebSocket deben establecerse mediante el esquema wss://, empleando configuraciones criptográficas robustas y certificados digitales confiables (Raciti y Vitello, 2022). Este lineamiento constituye la base para garantizar la confidencialidad e integridad de los datos transmitidos (Muhammad et al., 2024).
- Autenticación y autorización robustas: Se recomienda emplear mecanismos de autenticación mediante tokens JWT, ya que permiten una validación de sesiones segura y sin estado, complementados con políticas de expiración de acceso que reduzcan el riesgo de uso indebido de credenciales y fortalezcan la protección de datos sensibles (Ashfi, 2025). La autorización debe realizarse de forma explícita en la capa de aplicación, evitando la reutilización implícita de sesiones HTTP (Jones, 2024).
- Validación estricta del encabezado Origin: El servidor debe verificar el encabezado Origin durante el handshake inicial y permitir únicamente dominios previamente autorizados (Raciti y Vitello, 2022). Este control debe complementarse con tokens antifalsificación para mitigar ataques CSWSH (Ghasemshirazi y Heydarabadi, 2021).
- Gestión segura del ciclo de vida de la conexión: Las conexiones WebSocket deben contar con políticas claras de expiración, cierre automático por inactividad y revocación de credenciales, este lineamiento reduce el riesgo asociado a sesiones persistentes no controladas (Manglik, 2023).
- Control de recursos y mitigación DoS: Es fundamental establecer límites de concurrencia, aplicar rate limiting y monitorear el consumo de recursos por conexión, estas medidas permiten preservar la disponibilidad del sistema frente a intentos de saturación (Subramanian, 2025).
- Validación y sanitización de mensajes: Todo mensaje recibido debe ser validado conforme a esquemas predefinidos y sometido a procesos de sanitización antes de su procesamiento, este enfoque previene ataques de inyección y manipulación de datos (Johnson, 2025).
- Monitoreo continuo y respuesta a incidentes: La implementación de mecanismos de registro, monitoreo en tiempo real y alertas automatizadas resulta esencial para detectar comportamientos anómalos y responder oportunamente a incidentes de seguridad (Dementyev, 2023).

4.4 Limitaciones y consideraciones contextuales

Si bien los lineamientos propuestos se fundamentan en la literatura existente, su aplicación práctica enfrenta diversas limitaciones. Entre ellas se encuentran la heterogeneidad de los entornos tecnológicos, las restricciones de recursos en organizaciones pequeñas y la falta de personal especializado en seguridad (Lucio y Campaña, 2024). Además, la adopción de

controles avanzados, como detección de anomalías basada en inteligencia artificial, puede resultar inviable en ciertos contextos debido a su complejidad y costo (González, 2025).
Por ello, se enfatiza que los lineamientos deben interpretarse como un marco orientador y no como un conjunto rígido de reglas. En este sentido, su implementación debe ser flexible y adaptada a las capacidades de cada organización, permitiendo priorizar controles esenciales y reforzar la seguridad sin afectar la operatividad del sistema (Mitrović et al., 2023).

4.5 Implicaciones teóricas y proyección futura

Desde el punto de vista académico, esta síntesis contribuye a consolidar un marco conceptual que integra las principales recomendaciones dispersas en la literatura sobre seguridad en WebSockets. Al articular vulnerabilidades, amenazas y estrategias de mitigación en un conjunto coherente de lineamientos, el estudio aporta una base teórica que puede ser utilizada como referencia en futuras investigaciones.
Asimismo, investigaciones recientes sobre el uso de WebSockets en aplicaciones financieras y de criptoactivos destacan la necesidad de fortalecer los marcos teóricos en torno a la confiabilidad, disponibilidad y gestión eficiente de comunicaciones persistentes en tiempo real, dada la alta criticidad de estos entornos, lo que abre nuevas líneas de investigación para mejorar el diseño y la seguridad de estos sistemas (Rokhim et al., 2024).

4.6 Síntesis final del desarrollo

En conjunto, el desarrollo teórico realizado evidencia que la seguridad en WebSockets requiere un enfoque integral que equilibre eficiencia, disponibilidad y protección. La formulación de lineamientos teóricos de buenas prácticas responde a la necesidad de cerrar la brecha existente entre la adopción del protocolo y la madurez de sus mecanismos de seguridad, ofreciendo una guía estructurada para su implementación segura en aplicaciones web modernas.

Conclusiones

El análisis documental realizado evidencia que WebSocket, aunque eficiente para la comunicación en tiempo real, introduce riesgos significativos cuando no se implementa bajo criterios de seguridad sólidos. La ausencia de mecanismos nativos de autenticación y la dependencia de la correcta configuración por parte de los desarrolladores incrementan la superficie de ataque. Vulnerabilidades como CSWSH, ataques Man-in-the-Middle y denegación de servicio representan amenazas recurrentes en aplicaciones modernas. La literatura revisada coincide en que el uso obligatorio de TLS, la autenticación mediante tokens y la validación estricta del encabezado Origin constituyen controles fundamentales. Asimismo, la gestión adecuada de sesiones, la limitación de recursos y el monitoreo continuo permiten fortalecer la disponibilidad y la detección temprana de incidentes. Se evidencia también la falta de estándares universales que orienten la implementación segura del protocolo. Por ello, resulta pertinente proponer lineamientos teóricos que sirvan como

referencia para desarrolladores y responsables técnicos. La investigación demuestra que la seguridad en WebSockets debe abordarse desde un enfoque integral y por capas. Finalmente, se concluye que el fortalecimiento de buenas prácticas contribuirá a reducir riesgos y mejorar la confianza en aplicaciones web basadas en este protocolo.

Referencias bibliográficas

Adegbamigbe, A. (2021). *Mint 709 Capstone Project - Final Report: Implementation of a Video-Conferencing Web- Application using the WebRTC and WebSockets APIs within a Local Area Network* [Tesis de Mestría, University of Alberta] U Alberta. https://ualberta.scholaris.ca/server/api/core/bitstreams/1110be1c-692d-4190-a0b1-07a3b3186161/content

Ahlawat, D. (2023). *Automating Security Test-cases using DevSecOps approach for AWS Serverless application with WebSockets* [Tesis de Maestría, National College of Ireland] NVI Website. https://norma.ncirl.ie/6504/

Ahmed, R., Biradar, P., Bharadwaj, R., y Rishith, P. (2024). Real Time Communication using Web Sockets. *Social Science Research Network.* 1-5. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4912638

Álvarez, M. (2021). La contratación electrónica mediante plataformas en línea: modelo negocial (B2C), régimen jurídico y protección de los contratantes (proveedores y consumidores). *Reus Editorial*. https://www.google.com.ec/books/edition/La_contratación_electrónica_mediante_p/ZhI5EAAAQBAJ?hl=en&gbpv=0

Ashfi, N. R. (2025). A Real-Time Emergency Communication Framework Using WebSockets and the MERN Stack. *Journal of Computer, Software, and Program (JCSP), 2(*2), 34-43. https://journals.stecab.com/jcsp/article/view/1159

Avasthi, A., Pathrikar, A., Rajesh, D., y Ravishankar, V. (2025). *IoT (Internet of things) and its Application.* RK Publication. https://n9.cl/lfnlpd

Awuor, O. (2023). Review of the security challenges in web-based systems. *World Journal of Advanced Engineering Technology and Sciences, 08*(02), 204-216. https://www.researchgate.net/publication/369775890_Review_of_the_security_challenges_in_web-based_systems

Baloch, R. (2024). *Web Hacking Arsenal: A Practical Guide to Modern Web Pentesting*. CRC Press. https://n9.cl/82nx4

Banco Interamericano de Desarrollo (BID). (2020). *Ciberseguridad 2020: Riesgos, avances y el camino a seguir en América Latina y el Caribe*. BID. https://publications.iadb.org/publications/spanish/document/Reporte-Ciberseguridad- 2020-riesgos-avances-y-el-camino-a-seguir-en-America-Latina-y-el-Caribe.pdf

Batista, R. (2021). Enfoque transcultural de la ciberseguridad. Desafíos para el balance regional en América Latina. *IV Conferencia Científica Internacional UCIENCIA 2021*, 1-18. https://repositorio.uci.cu/handle/123456789/9695

Bussey, S. (2020). Real-Time Phoenix: Build Highly Scalable Systems with Channels. *Pragmatic Programmers.* https://n9.cl/qe5xl

Bux, K., Yousaf, M., Shahzad, M., Jamilo, R., Hussain, A., y Memon, Z. (2021). Poor Coding Leads to DoS Attack and Security Issues in Web Applications for Sensors. *Security and Communication Networks,* 1-11. https://onlinelibrary.wiley.com/doi/abs/10.1155/2021/5523806

Cahyo, E. (2025). A Horizontally Scalable WebSocket Architecture for Cost-Effective Online Examination Proctoring System on AWS Cloud Infrastructure. *JURNAL EMACS (Engineering, MAthematics and Computer Science), 7*(1), 109-120. https://journal.binus.ac.id/index.php/EMACS/article/view/12770

Chowdary, G. V. K. (2021). A Hybrid WebSocket-REST Approach for Scalable Real-Time API Design. *International Journal of Emerging Trends in Computer Science and Information Technology, 2*, 60-69. https://www.ijetcsit.org/index.php/ijetcsit/article/view/174/141

Delgado, M. (2025). *Automated Fuzzing Framework for Vulnerability Detection in Complex Systems* [Tesis de Maestría, Univerdidade Do Porto] Proquest. https://www.proquest.com/openview/d2338b728d68e89f3e558bcc491bfe3d/1?pq-origsite=gscholar&cbl=2026366&diss=y

Dementyev, V. (2023). Layered Design for Ruby on Rails Applications. *Packt Publishing*. https://n9.cl/4v8w8

Dynowski, L., y Dulak, M. (2025). *Learning API Styles: Understanding the Trade-Offs of Common APIs and Choosing the Correct Solutions*. O'Reilly Media. https://n9.cl/hwsyz

ESET. (2024). *Las amenazas más detectadas en Latinoamérica en el primer semestre 2024*. ESET Latinoamérica. https://www.eset.com/py/acerca-de-eset/sala-de-prensa/comunicados-de-prensa/articulos-de-prensa/las-amenazas-mas-detectadas-en-latinoamerica-en-el-primer-semestre-2024

Faishal, M., y Sutopo, J. (2025). Comparative Performance Analysis Between the MQTT and WebSocket Protocols. *Bit-Tech(Binary Digital -Technology), 8*(2), 1-11. https://jurnal.kdi.or.id/index.php/bt/article/view/3223/1701

Flow, S. (2021). How to Hack Like a Ghost: Breaching the Cloud. *No Starch Press*. https://n9.cl/1jl4w

Ghasemshirazi, S., y Heydarabadi, P. (2021). Exploring the Attack Surface of WebSocket. *Cornell University,* 1-5. https://arxiv.org/abs/2104.05324

Giovanni, B. (2020). Token Based Authorization. *Bonn: Gesellschaft für Informatik, 1*(1), 179-184. https://dl.gi.de/items/bf34d487-7396-4f8e-a873-a98258f35dee

González, O. (2025). *o3 Mini: La revolución del razonamiento en la IA.* RedUSERS. https://n9.cl/c3yi4

Gourko, A. (2024). *WebSocket Communication between Multiple Users in Scalable Web-application Environment* [Tesis de Maestría, Metropolia University of Applied Sciences] Theseus. https://www.theseus.fi/bitstream/handle/10024/860920/Gourko_Anatoli.pdf?sequence=5&isAllowed=y

Grande, J. (2024). *Entorno inmersivo 3D con tecnologías web* [Tesis de Grado, Escuela de Ingeniería de Fuenlabrada] Burjc Digital. https://burjcdigital.urjc.es/server/api/core/bitstreams/cf17b743-fd1f-4d4d-b4b9-a74b4a1c46bc/content

Hilbing, T., Schreck, T., y Limmer, T. (2023). 'State of the Union': Evaluating Open Source Zero Trust Components. En Rios, R., y Posegga, J (Eds.). *Security and Trust Management: 19th International Workshop, STM 2023, The Hague, The Netherlands, September 28, 2023, Proceedings*. Springer. https://n9.cl/x8hjv8

Honbo, X., Zhenyu, C., Shuhao, L., Chenxu, W., Peishuai, S., Jiang, X., y Qingyun, L. (2024). ProxyKiller: An Anonymous Proxy Traffic Attack Model Based on Traffic Behavior Graphs. En Garcia, J., Kozik, R., Choraś, M., y Katsikas, S (Eds.). *Computer Security – ESORICS 2024.* Springer. https://n9.cl/hb05s

Interrante, B. A. (2025). *APIOps and DevSecOps: Automating API Deployment and Security Scanning in Cloud Environments* [Tesis de Master, Politecnico di Torino] Biblioteche d'ateneo. https://webthesis.biblio.polito.it/37730/

Johnson, R. (2024). The WebSockets Handbook: Seamless Communication for Web, Mobile, and IoT. *HiTeX Press.* https://n9.cl/jxu4g

Johnson, R. (2025). *Jetty Essentials: Definitive Reference for Developers and Engineers*. HiTeX Press. https://n9.cl/edgyh

Johnson, R. (2025). *PrimeFaces Solutions: Definitive Reference for Developers and Engineers*. HiTeX Press. https://n9.cl/jigjn

Jones, A. (2024). In-Depth Exploration of Spring Security: Mastering Authentication and Authorization. *Walzone Press*. https://n9.cl/uk3jg9

Kayatas, K. (2025). *Enhancing smart environments through an ai-assisted iort agent* [Tesis de Maestría, Istanbul Technical University] Polen. https://polen.itu.edu.tr/server/api/core/bitstreams/e2dfcd53-a634-4f8f-8ee3-6d703a49378d/content

Khodayari, S., Barber, T., y Pellegrino, G. (2024). The Great Request Robbery: An Empirical Study of Client-side Request Hijacking Vulnerabilities on the Web. *2024 IEEE Symposium on Security and Privacy (SP),* 166-184. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10646795

Kumar, M. (2024). Designing resilient front end architectures for real-time web application. *International Journal of Engineering Technology Research & Management, 8*(8), 229-240. https://ijetrm.com/issues/files/May-2024-22-1747887028-AUG202430.pdf

Łasocha, W., & Badurowicz, M. (2021). Comparison of WebSocket and HTTP protocol performance. *Journal of Computer Sciences Institute, 19,* 67–74. https://ph.pollub.pl/index.php/jcsi/article/view/2452

Lucio, M., y Campaña, E. (2024). Desafíos y estrategias de ciberseguridad para pequeñas empresas. *Revista Electrónica de Gestión Ciencias Gerenciales, 6*(11), 18–36. https://iieakoinonia.org/ojs3/index.php/gestioep/article/view/151/235

Manglik, R. (2023). *Internet of Things*. EduGorilla Publication. https://n9.cl/h4ex0

Ministerio de Telecomunicaciones y de la Sociedad de la información. (2022). *Diagnóstico de las capacidades de Ciberseguridad*. Ministerio de Telecomunicaciones y de la Sociedad de la información: República del Ecuador. https://www.telecomunicaciones.gob.ec/wp-content/uploads/2023/05/DIAGNO%CC%81STICO-DE-LAS-CAPACIDADES-DE-CIBERSEGURIDAD-Ecuador-Diciembre-2022_compressed.pdf

Mitrović, N., Đorđević, M., Veljković, S., y Danković, D. (2023). Implementation and testing of websocket protocol in ESP32 based IOT systems. *Facta Universitatis, 36*(2), 267 – 284. https://casopisi.junis.ni.ac.rs/index.php/FUElectEnerg/article/view/11223

Muhammad, M., Raza, Y., Sajid, A., Razzaq, H., Malik, R., y Vidanagamachchi, S. (2024). Review Analysis of Web Socket Security: Case Study, *iTDAF, 2*(2), 56-75. https://www.researchgate.net/publication/384105299_Review_Analysis_of_Web_Socket_Security_Case_Study

Murley, P. (2023). *Exploring the trade-offs in web page behavioral abstractions* [Tesis de Doctorado, University of Illinois Urbana-Champaign] Illinois Library. https://www.ideals.illinois.edu/items/127485

Murley, P., Ma, Z., Mason, J., Bailey, M., y Kharraz, A. (2021). Websocket adoption and the landscape of the real-time web. *In Proceedings of the Web Conference 2021*, 1192-1203. https://dl.acm.org/doi/abs/10.1145/3442381.3450063

Murley, P., Ma, Z., Mason, J., Bailey, M., y Kharraz, A. (2021). Websocket adoption and the landscape of the real-time web. *In Proceedings of the Web Conference 2021*, 1192-1203. https://dl.acm.org/doi/abs/10.1145/3442381.3450063

Neru., y Yudertha., A. (2020). A Proposed Design of Realtime Patient Monitoring System Using Websocket as a Basis of Telemedicine. *Atlantis Press SARL, 172,* 263-268. https://www.atlantis-press.com/proceedings/siconian-19/125939990

Obonguko, U. (2025). Websockets vs. Http polling: implications for high-frequency data streaming applications. *International journal of modern technology and engineering research, 3*(1), 59-72. http://caarnjournals.com/index.php/IJMTER/article/view/174/182

Oktavia, I., Dwitya, y., Atho, H., y Kuswanto, D. (2020). Secure Data Flow Messaging on Web Socket using Rivest Code 6. *In Proceedings of the International Conference on Culture Heritage, Education, Sustainable Tourism, and Innovation Technologies (CESIT 2020).* 151-157. https://www.scitepress.org/Papers/2020/103054/103054.pdf

Pandey, B., y Kumar, P. (2024). Security Risks and Best Practices for Securing WebSocket Communications. *International journal of novel research and development (IJNRD), 9*, 753-755. https://www.ijnrd.org/papers/IJNRD2407274.pdf

Paweł, W., y Badurowicz., M. (2021). Comparison of WebSocket and HTTP protocol performance. *Journal of Computer Sciences Institute, 19,* 67-74. https://ph.pollub.pl/index.php/jcsi/article/view/2452

Picado, F., y Pérez, M. (2023). Administración de servicios web: Anatomía del Internet. *Marcombo.* https://www.google.com.ec/books/edition/Administración_de_servicios_web_Anatom/dtavEAAAQBAJ?hl=en&gbpv=0

Raciti, M., y Vitello, F. (2022). WebSocket Integration in Django. *Rapporti Tecnici INAF, 143,* 1-14. https://openaccess.inaf.it/server/api/core/bitstreams/2aadcb8a-6b1a-405c-8336-9133301ec099/content

Rakib, N. (2025). A Real-Time Emergency Communication Framework Using WebSockets and the MERN Stack. *Journal of Computer, Software, and Program (JCSP), 2*(2), 34-43. https://journals.stecab.com/jcsp/article/view/1159

Rokhim, A., Alimin., y Mustofa. Implementasi extreme programming untuk monitoring harga cryptocurrency antar exchanger dengan menggunakan websocket. *Jurnal spirit, 16*(2), 318 – 327. http://www.jurnal.stmik-yadika.ac.id/index.php/spirit/article/view/356

Sathya, S., y Swetha, S. (2025). Real-Time Chat Application with Web Socket for Network Efficiency. *International Journal of Scientific Research in Science and Technology, 12*(4), 830-835. https://www.researchgate.net/publication/394388196_Real-Time_Chat_Application_with_Web_Socket_for_Network_Efficiency

Smith, W. (2025). Godot 4 WebAssembly Export Guide: The Complete Guide for Developers and Engineers. *HiTeX Press.* https://n9.cl/exmkz

Subramanian, R. (2025). Mastering APIs for Enterprise Applications: Practical Guide to Building Robust, Scalable, and Secure APIs (English Edition). *Bpb Publications*. https://n9.cl/hz72j0

Tommasi, F., Catalano, C., y Taurino, I. (2021). Browser-in-the-Middle (BitM) attack. *International Journal of Information Security, 21*, 179–189. https://link.springer.com/article/10.1007/s10207-021-00548-5

Tsoukalos, M. (2024). Mastering Go: Leverage Go's Expertise for Advanced Utilities, Empowering You to Develop Professional Software. *Packt Publishing*. https://n9.cl/b48do

Ziemann, V. (2023). *A Hands-On Course in Sensors Using the Arduino and Raspberry Pi*. CRC Press. https://n9.cl/sddmx

Conflicto de intereses:
Los autores declaran que no existe conflicto de interés posible.
Financiamiento:
No existió asistencia financiera de partes externas al presente artículo.
Agradecimiento:

Nota:
El artículo no es producto de una publicación anterior.