UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR

POSGRADOS

MAESTRÍA EN
SEGURIDAD DE LA INFORMACIÓN
RPC-SO-28-NO.669-2021

OPCIÓN DE TITULACIÓN:

PROYECTO DE TITULACIÓN CON COMPONENTES DE INVESTIGACIÓN APLICADA Y/O DE DESARROLLO

TEMA:

ANÁLISIS DE SEGURIDAD EN LA TRANSMISIÓN DE DATOS MEDIANTE VPNS, EN ENTORNOS DE IOT

AUTOR:

JOSÉ LUIS GUAMÁN ANDRADE

DIRECTOR:

JUAN DIEGO JARA SALTOS

CUENCA – ECUADOR
2025

Autor:

José Luis Guamán Andrade
Ingeniero en Electrónica, Telecomunicaciones y Redes.
Candidato a Magíster en Seguridad de la Información por la Universidad Politécnica Salesiana – Sede Cuenca.
josel.gandrade@outlook.com

Dirigido por:

Juan Diego Jara Saltos
Ingeniero Electrónico.
Máster en Telemática.
Máster en Métodos Matemáticos y Simulación Numérica en Ingeniería.
jjaras@ups.edu.ec

DEDICATORIA

A Dios, por ser mi guía constante, por darme la fuerza en los momentos difíciles y permitirme llegar hasta aquí. A mi madre Mery, por su amor incondicional, su paciencia infinita y su presencia firme en cada paso que he dado; a mi tía Marianita, por su cariño sincero, sus palabras de ánimo y su fe inquebrantable en mí; a mi hermano Alex, por ser un ejemplo de esfuerzo, disciplina y superación, y por inspirarme en cada etapa de este camino. Extiendo también mi gratitud a toda mi familia, en especial a mis tíos y primos, por su afecto, por cada palabra de aliento y por creer en mí incluso en los momentos más desafiantes. Este logro es el reflejo del amor, el apoyo y la confianza que siempre me han brindado.

Este logro no es el final del camino, sino el inicio de un nuevo capítulo en el que cada esfuerzo sembrado empieza a dar frutos.

AGRADECIMIENTO

A Dios, por bendecirme con salud, vida y la oportunidad de alcanzar esta meta académica que hoy se materializa. A mi madre, por su apoyo permanente desde el inicio de mis estudios, por ser más que una madre: una amiga incondicional que me dio fortaleza y me motivó con amor a lo largo de este proceso. A mi familia, por ser mi inspiración constante, por creer siempre en mí, aconsejarme, comprenderme y acompañarme en cada etapa, tanto en los momentos de dificultad como en los de alegría. Al Mgtr. Juan Diego Jara Saltos, director de esta tesis, por su valiosa orientación, compromiso y dedicación, que fueron fundamentales para culminar con éxito este trabajo de titulación. A la Universidad Politécnica Salesiana, por brindarme una formación académica rigurosa y actualizada, para contribuir con responsabilidad al entorno profesional y social.

Tabla de Contenido

Análisis de Seguridad en la transmisión de datos mediante VPNS, en Entornos de IOT

Autor(es):

José Luis Guamán Andrade

Resumen

La presente investigación analiza la seguridad en transmisiones de datos mediante diferentes redes privadas virtuales (VPNs) en entornos de IOT (Internet de las Cosas), considerando aspectos de cifrado, latencia y resistencia frente a ataques cibernéticos. Se evaluaron cuatro tecnologías VPN: WireGuard, OpenVPN, Windscribe y NordVPN, implementadas en un entorno simulado compuesto por dispositivos Ubuntu y Kali Linux, con tráfico MQTT generado entre clientes y brokers IoT.

Las pruebas realizadas incluyeron análisis de cifrado mediante tcpdump y Wireshark, verificación de fugas DNS y simulaciones de ataques tipo MITM (Man-in-the-Middle) utilizando herramientas como Bettercap. Además, se evaluó la visibilidad del tráfico encapsulado en cada VPN, la capacidad de aislamiento entre nodos y la integridad de las comunicaciones durante escenarios de ataque.

Los resultados evidencian que tecnologías como WireGuard y NordVPN presentan mejor desempeño en términos de latencia y encapsulamiento efectivo del tráfico, mientras que OpenVPN ofrece mayor flexibilidad a costa de una configuración más compleja. Windscribe, por su parte, mostró mayores niveles de vulnerabilidad en pruebas activas. El análisis concluye con recomendaciones técnicas orientadas a fortalecer la seguridad en implementaciones reales de infraestructura IoT protegidas mediante VPN.

Palabras clave:

ciberseguridad, redes privadas virtuales, IoT, cifrado, análisis de vulnerabilidades.

ABSTRACT

This research analyzes data transmission security using VPNs in IoT environments, focusing on encryption mechanisms, latency, and resilience against cyberattacks. Four VPN technologies were evaluated: WireGuard, OpenVPN, Windscribe, and NordVPN, implemented in a simulated environment with Ubuntu and Kali Linux devices and MQTT traffic between clients and brokers.

The tests included encryption analysis using tcpdump and Wireshark, DNS leak detection, and Man-in-the-Middle (MITM) attack simulations with tools such as Bettercap. The study also assessed traffic visibility within each VPN tunnel, node isolation effectiveness, and communication integrity during attack scenarios.

Results indicate that WireGuard and NordVPN exhibit superior performance in terms of latency and traffic encapsulation, while OpenVPN provides greater configurability at the cost of complexity. Windscribe showed higher vulnerability under active tests. The study concludes with technical recommendations to enhance security in real-world VPN-protected IoT infrastructures.

Key words:

cybersecurity, virtual private networks, IoT, encryption, vulnerability analysis.

1 Introducción

El auge de los sistemas de IoT ha modificado la forma en que los dispositivos recopilan, transmiten y procesan datos en diversos sectores, desde salud hasta industrias y viviendas inteligente. Sin embargo, esta conectividad masiva también plantea importantes desafíos en la seguridad de la información. Los dispositivos IoT suelen operar en redes públicas vulnerables, lo que los expone a riesgos como la interceptación de datos, ataques de intermediario MITM y accesos no autorizados. En este contexto, las Redes Privadas Virtuales han emergido como una solución crítica para garantizar la confidencialidad, integridad y autenticidad de los datos transmitidos.

Las VPN proporcionan túneles seguros mediante protocolos de cifrado que protegen las comunicaciones en redes públicas. Tecnologías como OpenVPN, IPsec y, más recientemente, WireGuard, son ampliamente utilizadas para establecer conexiones seguras. En entornos IoT, estas tecnologías no solo deben garantizar altos niveles de seguridad, sino también operar de manera eficiente en dispositivos con limitaciones de recursos. Estudios recientes han evaluado el rendimiento de diferentes protocolos VPN en escenarios IoT, destacando que el impacto en latencia, consumo de energía y velocidad de transmisión puede influir significativamente en la elección de la tecnología adecuada para garantizar la seguridad sin comprometer el rendimiento.

Este trabajo revisa el papel de las VPN en la protección de la información en sistemas IoT, analizando su capacidad para prevenir amenazas específicas como el espionaje de datos y el acceso no autorizado. Asimismo, se evalúan Los beneficios y las limitaciones de las diferentes tecnologías VPN en función de su efectividad para mitigar riesgos y su compatibilidad con dispositivos.

2 Determinación del Problema

2.1 Antecedentes

Desde una perspectiva histórica, la seguridad de la información ha sido una preocupación desde los inicios de la computación y las telecomunicaciones. Con el advenimiento de la Internet y la comunicación en red, se volvió fundamental garantizar la seguridad de la información frente a accesos no autorizados y preservar la confidencialidad en las comunicaciones. Las VPNs se introdujeron inicialmente como una solución para conectar redes privadas a través de Internet de manera segura, encriptando el tráfico de datos y proporcionando autenticación para evitar accesos no autorizados.

La evolución de la tecnología IoT ha elevado la importancia de las VPNs. A nivel mundial, las políticas de ciberseguridad han comenzado a incluir la protección de dispositivos IoT como un componente crucial. Organizaciones como la Agencia de Ciberseguridad de la Unión europea (ENISA) junto con el instituto Nacional de Estándares y Tecnología de Estados Unidos (NIST) desempeñan un papel clave en el establecimiento de normas y directrices para la protección digital han publicado directrices y estándares que enfatizan la necesidad de métodos seguros de comunicación para dispositivos IoT.

A nivel regional y nacional, los gobiernos están reconociendo la importancia de implementar regulaciones específicas para proteger las redes de IoT, considerando la cantidad creciente de dispositivos conectados en infraestructuras críticas y en aplicaciones cotidianas. Estas medidas buscan prevenir ataques cibernéticos que podrían tener consecuencias a gran escala, afectando la seguridad nacional y la vida cotidiana de los ciudadanos.

En el contexto local y específico del proyecto, la implementación de VPNs para asegurar la transmisión de datos en entornos IoT no solo es relevante para empresas y organizaciones que manejan información sensible, sino también para los consumidores individuales que utilizan dispositivos IoT en sus hogares.

Esto subraya que la temática abordada trasciende un contexto puramente profesional o empresarial y tiene implicaciones de gran alcance que afectan tanto a las políticas públicas como a la seguridad personal y la privacidad de los individuos.

2.2 Formulación del Problema

La rápida incorporación de la Internet de las Cosas ha transformado nuestra forma de interactuar con la tecnología, al permitir que diversos dispositivos se conecten entre sí mediante redes para recolectar, intercambiar y procesar información en tiempo real. No obstante, este crecimiento acelerado en la cantidad de dispositivos conectados también ha generado nuevas amenazas en el ámbito de la seguridad informática. Un desafío científico importante se encuentra en proteger la transmisión de datos dentro de estos entornos IoT, ya que la comunicación entre dispositivos es susceptible a ser interceptada, alterada o vulnerada por agentes con intenciones maliciosas

Las causas fundamentales de este problema incluyen la poca inclusión de estándares de seguridad universales para dispositivos de IoT, la variabilidad en las capacidades de los dispositivos en cuanto a procesamiento y almacenamiento, y la creciente sofisticación de los ciberataques. A medida que los dispositivos IoT se implementan en sectores críticos como la salud, la infraestructura y la automatización industrial, los riesgos asociados a las vulnerabilidades en la transmisión de datos se intensifican, exponiendo información sensible a amenazas cibernéticas.

Los efectos de no abordar adecuadamente este problema son multifacéticos y potencialmente devastadores. Estos incluyen perdidas de confidencialidad de datos e integridad, riesgos de privacidad para los usuarios, interrupciones en la funcionalidad de dispositivos críticos, y daños económicos y reputacionales significativos para las organizaciones afectadas. Por lo tanto, existe una necesidad imperiosa de desarrollar soluciones robustas y eficaces para poder asegurar seguridad en la transmisión de datos en entornos IoT, siendo las redes privadas virtuales (VPNs) una de las estrategias más prometedoras para mitigar estos riesgos.

2.3 Justificación del problema

El crecimiento acelerado de la Internet de las Cosas (IoT) ha dado lugar a un entorno interconectado de dispositivos que permite una amplia variedad de aplicaciones en ámbitos como la industria, las ciudades inteligentes y el monitoreo ambiental. No obstante, esta conectividad también introduce retos significativos en materia de ciberseguridad, ya que muchos dispositivos IoT funcionan sobre redes públicas susceptibles a ataques, lo que los hace propensos a la interceptación y alteración de datos. En este contexto, las Redes Privadas Virtuales (VPN) surgen como una alternativa viable para proteger las comunicaciones, asegurando la confidencialidad de la información y previniendo accesos no deseados.

Este proyecto tiene como objetivo analizar el desempeño de diversas tecnologías VPN, como OpenVPN, IPsec y WireGuard, en términos de velocidad de transmisión y latencia en redes públicas. La investigación busca identificar soluciones que no solo aseguren confidencialidad de datos e integridad, sino que también sean compatibles con diferentes limitaciones técnicas de dispositivos IoT, como su capacidad de procesamiento y consumo energético.

Más allá de los desafíos técnicos, la seguridad de la información en entornos IoT representa una preocupación creciente, especialmente por su impacto en el cumplimiento de marcos regulatorios internacionales como el Reglamento General de Protección de Datos (GDPR) en Europa y la Ley de Privacidad del Consumidor de California (CCPA) en Estados Unidos. Estas normativas requieren que las organizaciones adopten medidas adecuadas para salvaguardar los datos personales. En este sentido, las Redes Privadas Virtuales (VPN) pueden desempeñar un papel fundamental como herramienta para asegurar el cumplimiento tanto legal como ético en la gestión de la información

El impacto de este proyecto se enfoca principalmente en mejorar la seguridad de la información en un entorno específico donde la transmisión confiable de datos resulta crítica, como en sistemas IoT aplicados al monitoreo ambiental y a la gestión eficiente de recursos. En estos casos, la incorporación de tecnologías VPN permite fortalecer la privacidad e integridad de datos transmitidos, facilitando la adopción

de soluciones seguras que respondan a necesidades concretas del sector tecnológico y productivo.

Finalmente, esta investigación podría sentar las bases para la adopción generalizada de VPNs como estándar de seguridad en IoT, demostrando su efectividad y compatibilidad con las necesidades operativas de los dispositivos conectados. Al establecer parámetros claros para su implementación, se podrían desarrollar protocolos específicos para IoT que integren VPNs de manera eficiente, fomentando la confianza de los usuarios, la adopción masiva de tecnologías IoT y la creación de normativas internacionales que refuercen la seguridad en esta área.

2.4 Objetivos

2.4.1 Objetivo Principal

Analizar la eficiencia de diferentes tecnologías VPN para la transmisión segura de datos en redes públicas, garantizando confiabilidad y privacidad en entornos de sistemas de IoT.

2.4.2 Objetivos Específicos

- Investigar el desempeño de diversas tecnologías VPN en términos de velocidad de transmisión y latencia en redes públicas, identificando su impacto en sistemas IoT.
- Analizar las posibles vulnerabilidades y riesgos de seguridad asociados con el uso de distintas tecnologías VPN.
- Comparar la robustez y efectividad de los mecanismos de cifrado y autenticación de diferentes tecnologías de hardware y software VPN (gratuitas y de pago), para garantizar la confiabilidad y privacidad de los datos en entornos IoT.

3 Marco teórico referencial

La Internet de las Cosas (IoT) ha revolucionado el entorno tecnológico al posibilitar la conexión entre dispositivos a través de redes tanto públicas como privadas, promoviendo la automatización y el flujo constante de datos en tiempo real (Gubbi et al., 2013; Lin et al., 2017). No obstante, esta extensa interconectividad ha planteado serias inquietudes en materia de ciberseguridad, dado que muchos dispositivos IoT poseen capacidades limitadas de procesamiento y protección, lo que los hace susceptibles a diversas amenazas (Roman et al., 2011; Mehmood et al., 2020). Ante este panorama, las Redes Privadas Virtuales (VPN) se han consolidado como una herramienta clave para resguardar las comunicaciones en estos sistemas, al ofrecer confidencialidad, integridad y verificación de los datos intercambiados (Almusaylim & Jhanjhi, 2019; Mehmood et al., 2020). Este marco teórico tiene como objetivo analizar las principales tecnologías VPN aplicables a IoT, evaluando su rendimiento en aspectos como la latencia y la velocidad, así como su influencia en la mejora de la seguridad de la información.

3.1 IOT (Internet de las cosas)

Internet de las Cosas es un paradigma emergente que nos permite la interconexión de objetos físicos equipados con sensores, actuadores, software y tecnologías de comunicación. Estos dispositivos recopilan, procesan y transmiten datos a través de redes, lo que permite la integración del mundo físico y digital (Gubbi et al., 2013; Lin et al., 2017). IoT se caracteriza por su capacidad de proporcionar servicios inteligentes basados en la interoperabilidad, la ubicuidad y la autonomía de los dispositivos conectados (Sicari et al., 2015). Este concepto abarca una gran cantidad de aplicaciones, desde hogares inteligentes y salud digital hasta ciudades inteligentes y cadenas de suministro industrial (Lu & Da Xu, 2019).

IoT no solo permite el monitoreo y control remoto de dispositivos, sino que también utiliza algoritmos avanzados y análisis de datos que optimizar procesos, mejorar la efectividad y generar valor añadido (Mehmood et al., 2020; Lin et al., 2017). Según Gubbi et al. (2013), la IoT se basa en tres pilares fundamentales:

- Conectividad ubicua: La capacidad de conectar dispositivos en tiempo real en cualquier lugar y momento.
- Procesamiento inteligente: Utilización de diferentes herramientas de inteligencia artificial y análisis de datos para sacar conocimiento de los datos recopilados.
- Interacción autónoma: Los dispositivos IoT pueden comunicarse y colaborar sin intervención humana directa.

Además, este paradigma depende de arquitecturas escalables y seguras, como las basadas en Cloud computing, para manejar grandes cantidades de datos y proporcionar servicios confiables. Su implementación presenta desafíos enlazados con la seguridad de la información, la interoperabilidad entre dispositivos y el manejo eficiente de recursos limitados como el consumo de energía y ancho de banda (Roman et al., 2011; Sicari et al., 2015; Khan & Salah, 2018).

3.2 VPNS (Redes virtuales privadas)

Las Redes Privadas Virtuales (VPN) representan una solución de seguridad que posibilita la extensión de redes privadas a través de infraestructuras públicas como Internet, mediante el uso de túneles cifrados. Este tipo de conexión protege los datos durante su transmisión, asegurando que se mantenga su confidencialidad, integridad y autenticidad, y permitiendo el acceso únicamente a usuarios con las credenciales adecuadas (Almusaylim & Jhanjhi, 2019; Kim et al., 2021).

En el contexto del Internet de las Cosas (IoT), las VPN adquieren un papel fundamental debido a las características únicas y vulnerabilidades de los dispositivos IoT. Estos dispositivos suelen operar en redes públicas o híbridas, lo que los expone a riesgos significativos como ataques intermediarios MITM o espionaje de datos (Fereidouni et al., 2025; Mehmood et al., 2020). Implementar VPN en entornos IoT permite asegurar la comunicación entre dispositivos conectados, creando canales protegidos que reducen las posibilidades de intrusión.

Además, en IoT, las VPN deben adaptarse a ciertas restricciones propias de los dispositivos, como la capacidad limitada de procesamiento, memoria y batería. Tecnologías como WireGuard han mostrado ser particularmente prometedoras en este ámbito, ya que ofrecen un diseño más ligero y eficiente en comparación con

soluciones tradicionales como OpenVPN o IPsec (Donenfeld, 2019; Saxena & Grizonic, 2021).

Las aplicaciones de VPN en IoT son amplias y abarcan sectores como el hogar inteligente, la industria 4.0, salud y las ciudades domóticas. Por ejemplo, las VPN pueden proteger datos sensibles generados por dispositivos médicos conectados o garantizar seguridad de transmisión de datos entre sensores industriales y sistemas de control centralizados (Almusaylim & Jhanjhi, 2019; Gupta & Rawat, 2020).

3.3 Identificación de Tecnologías VPN Utilizadas en Entornos IoT

La incorporación del Internet de las Cosas (IoT) ha redefinido el entorno tecnológico, al facilitar la conexión entre dispositivos que recolectan, procesan y transmiten información de forma inmediata. No obstante, este crecimiento también ha expuesto vulnerabilidades importantes en materia de seguridad, especialmente porque muchos de estos dispositivos funcionan sobre redes públicas expuestas a posibles ataques (Mehmood et al., 2020; Roman et al., 2011). En este contexto, las Redes Privadas Virtuales (VPN) se presentan como una herramienta clave para proteger la confidencialidad, integridad y autenticidad de los datos transferidos (Kim et al., 2021; Almusaylim & Jhanjhi, 2019). El presente trabajo de investigación aplicada analiza las tecnologías VPN más relevantes aplicadas en sistemas IoT, sus propiedades esenciales y su viabilidad en distintos contextos de implementación.

3.4 Tecnologías VPN Más Relevantes en IoT

3.4.1 Site-to-Site VPN (Red a Red)

Un Site-to-Site VPN, también conocido como VPN de Red a Red, es una tecnología que conecta dos o más redes locales (LAN) de manera segura a través de Internet o una red pública. Este tipo de VPN se utiliza principalmente para permitir que múltiples oficinas, sucursales, o sitios de una organización puedan comunicarse entre sí como si estuvieran dentro de una misma red privada como se muestra en la ilustración 3-1.

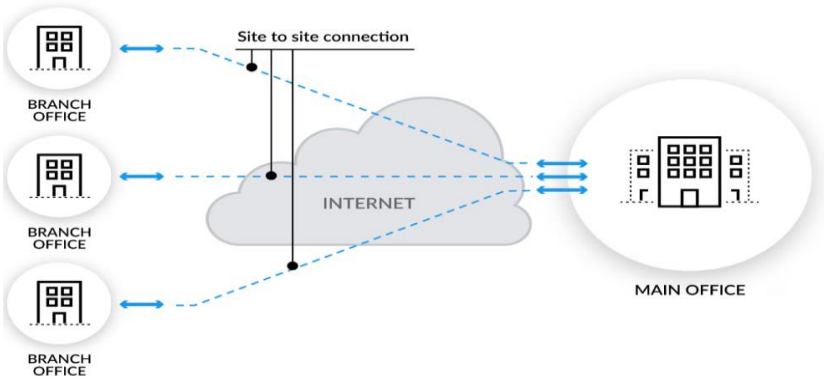

Ilustración 3-1 Side-to-Site VPN. Obtenido de: Attaran (2020), https://doi.org/10.1016/j.iot.2019.100129

El cifrado es uno de los pilares principales de los Site-to-Site VPNs. Los datos transmitidos a través del túnel VPN son cifrados utilizando algoritmos avanzados como AES (Advanced Encryption Standard), que proporciona niveles de seguridad robustos con contraseñas de 128, 192 o 256 bits (NIST, 2001; Kim et al., 2021). AES es ampliamente reconocido por la resistencia que presentan frente a ataques de fuerza bruta y es utilizada en la mayoría de los protocolos VPN modernos, incluidos IPsec y OpenVPN (Zhu et al., 2020). Este nivel de cifrado asegura que cualquier intento de interceptar los datos resulte inútil, ya que la información capturada estaría completamente ilegible sin la clave de descifrado adecuada.

3.4.2 Remote Access VPN (Acceso Remoto)

Las Remote Access VPNs (VPN de Acceso Remoto) son una tecnología esencial para conectar usuarios individuales o dispositivos remotos a una red privada de forma segura a través de Internet. Su robustez y efectividad dependen en gran medida de los mecanismos de cifrado y autenticación, que protegen la comunicación contra interceptaciones, accesos no autorizados y otras amenazas cibernéticas (Almusaylim & Jhanjhi, 2019; Kim et al., 2021). Estos mecanismos son fundamentales para garantizar la privacidad e integridad de los datos en aplicaciones críticas, como la gestión de dispositivos IoT o el acceso a recursos empresariales desde ubicaciones externas como se muestra en la ilustración 3-2.

Ilustración 3-2 Acceso Remoto y VPN. Obtenido de: Cloudflare (2024), https://www.cloudflare.com/es-es/learning/access-management/vpn-speed/

El cifrado en las Remote Access VPNs es uno de los principales pilares de su seguridad. Estas VPNs usan algoritmos de cifrado avanzados como Advanced Encryption Standard para la protección de los datos transmitidos entre el usuario remoto y la red privada. AES, con su capacidad de claves de 128, 192 o 256 bits, ofrece un nivel de seguridad robusto frente a ataques de fuerza bruta, lo que lo convierte en un estándar en la mayoría de los protocolos VPN, como OpenVPN, IKEv2/IPsec y WireGuard (Donenfeld, 2019; Saxena & Grizonic, 2021; Zhu et al., 2020). Este cifrado asegura que cualquier dato interceptado sea ilegible para actores malintencionados, protegiendo tanto la información confidencial del usuario como los datos sensibles de la red.

Además del cifrado de datos, los sistemas de autenticación desempeñan un papel fundamental en las VPN de acceso remoto, ya que permiten verificar que únicamente usuarios autorizados accedan a la red. Las soluciones VPN actuales incorporan diversos métodos de autenticación, como claves precompartidas (PSK), certificados digitales y autenticación multifactor (MFA) (Kim et al., 2021). Esta última ha ganado relevancia al proporcionar una capa adicional de protección, al requerir que los usuarios confirmen su identidad mediante contraseñas, códigos temporales generados por aplicaciones o tecnologías biométricas. Esto contribuye a mitigar significativamente el riesgo de accesos no autorizados, incluso en casos donde las credenciales principales han sido comprometidas.

Otro elemento clave en la seguridad de las VPN de acceso remoto es la implementación de protocolos seguros para el intercambio de claves, como IKEv2 (Internet Key Exchange versión 2). Este protocolo garantiza que las claves utilizadas para cifrar la información se generen y compartan de manera dinámica y segura entre el servidor y el cliente. IKEv2 resulta especialmente útil en entornos con usuarios que cambian de red con frecuencia, como en situaciones de movilidad entre diferentes conexiones (Zhu et al., 2020).

3.5 VPNs específicas para IoT

Las IoT-Specific VPNs son tecnologías de red diseñadas específicamente para abordar los desafíos de seguridad y rendimiento en el ámbito IOT. A diferencia de las VPN tradicionales, estas soluciones están optimizadas para dispositivos IoT, los cuales suelen tener limitaciones significativas en capacidad de procesamiento, memoria y consumo energético. La robustez y efectividad de las IoT-Specific VPNs radican en el uso de mecanismos de cifrado ligeros y autenticación eficiente que se adaptan a las restricciones de estos dispositivos sin comprometer la seguridad como se muestra en la ilustración 3-3 (Gupta & Rawat, 2020; Kim et al., 2021).

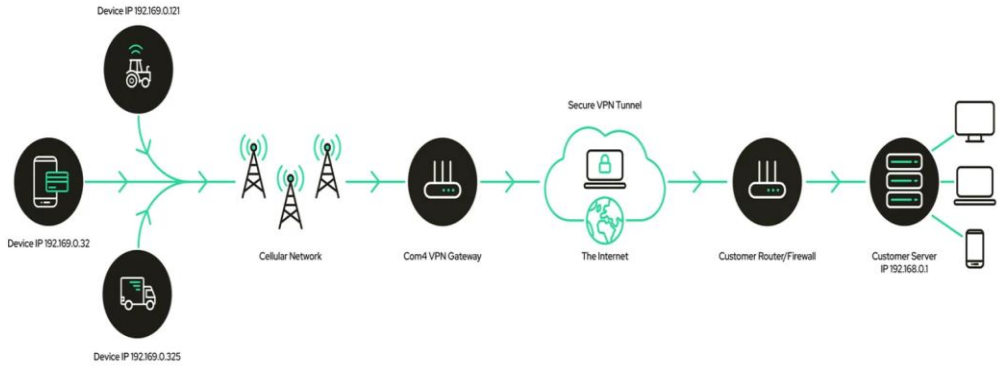

Ilustración 3-3 IoT-Specific VPNs. Obtenido de: OpenVPN Technologies Inc. (2022), https://openvpn.net

El cifrado es uno de los pilares fundamentales de las IoT-Specific VPNs, permitiendo que los datos que se transmiten entre dispositivos IoT y servidores centrales estén protegidos contra interceptaciones y manipulaciones. Estas VPNs suelen utilizar algoritmos de cifrado avanzados como ChaCha20, que ofrece un alto nivel de seguridad con menor overhead computacional en comparación con estándares tradicionales como AES (Donenfeld, 2019; Saxena & Grizonic, 2021). ChaCha20 es particularmente adecuado para dispositivos IoT debido a su eficiencia en hardware

limitado, lo que permite cifrar y descifrar datos de manera rápida y segura, incluso en dispositivos con recursos escasos.

Un aspecto clave de la robustez de estas VPNs es su capacidad para manejar grandes cantidades de dispositivos conectados simultáneamente. Protocolos como WireGuard, ampliamente adoptado en IoT-Specific VPNs, están diseñados para ser ligeros y escalables. WireGuard utiliza un enfoque simplificado para la gestión de claves y conexiones, reduciendo la complejidad del sistema y mejorando el rendimiento general (Donenfeld, 2019; Szefer, 2021). Esto es muy relevante en aplicaciones IoT donde miles de dispositivos pueden necesitar conectarse a la red central de forma simultánea, como en ciudades inteligentes o fábricas conectadas.

La efectividad de las IoT-Specific VPNs también se ve reflejada en su integración con tecnologías de nube y edge computing. Estas VPNs permiten que los dispositivos IoT transmitan datos de manera segura a servicios en la nube para su procesamiento y almacenamiento, al mismo tiempo que habilitan conexiones seguras con nodos de borde (edge nodes) para realizar análisis locales (Zhou et al., 2018). Esto minimiza la latencia y optimiza el uso de recursos, lo cual es importante para aplicaciones IoT que son sensibles al tiempo, como la automatización industrial o los sistemas de salud conectados.

3.5.1 Cloud VPN (VPN en la Nube)

Utilizadas para conectar dispositivos IoT a plataformas de cloud de forma segura. Ofrecen cifrado y autenticación entre los dispositivos IoT y servicios como AWS IoT o Azure IoT como se muestra en la ilustración 3-4.

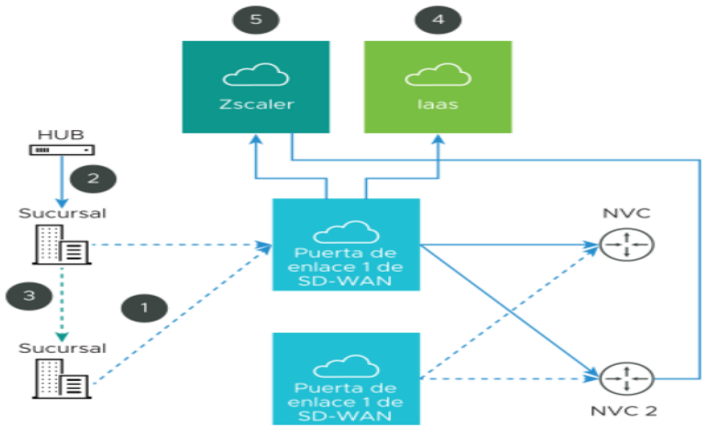

Ilustración 3-4 Cloud VPN. Obtenido de: NIST (2001), https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf

Las Cloud VPNs son altamente robustas debido al uso de algoritmos avanzados de cifrado como AES-256, que garantiza la protección de los datos transmitidos frente a ataques de fuerza bruta (Zhu et al., 2020; NIST, 2001). Estas VPNs están diseñadas para mantener conexiones estables y seguras incluso en redes públicas, adaptándose a grandes volúmenes de tráfico generado por dispositivos IoT. Además, la integración con servicios de la nube Google Cloud o Azure AWS mejora su confiabilidad, al combinar hardware especializado y sistemas de respaldo para evitar interrupciones en las comunicaciones (Cloudflare, 2024).

La efectividad de las Cloud VPNs radica en la capacidad de escalar y proteger múltiples dispositivos IoT de forma simultánea y controlada. Protocolos como IPsec o SSL/TLS garantizan que las conexiones sean seguras, mientras que las configuraciones automáticas y centralizadas permiten gestionar dispositivos distribuidos de manera eficiente (Kim et al., 2021; Saxena & Grizonic, 2021). En aplicaciones IoT como ciudades inteligentes o industrias conectadas, estas VPNs aseguran la transmisión de datos críticos, optimizando la integración entre dispositivos y la nube.

Los mecanismos de seguridad en Cloud VPNs combinan cifrado extremo a extremo y métodos avanzados de autenticación. El uso de certificados digitales y autenticación multifactor asegura que solo los dispositivos y usuarios que se autorizan accedan a la red. Además, la rotación de claves dinámicas reduce los riesgos asociados con la exposición de claves, y las tecnologías de intercambio de claves como IKEv2 refuerzan aún más la protección durante las transmisiones (Zhu et al., 2020; Gupta & Rawat, 2020).

3.5.2 Hybrid VPN (Híbrida)

Las Hybrid VPNs combinan las capacidades de las VPNs de acceso remoto (Remote Access VPN) y las VPNs de red a red (Site-to-Site VPN) para ofrecer una solución flexible y robusta. Estas redes permiten conexiones seguras tanto para usuarios individuales como para múltiples redes locales, integrando dispositivos IoT y usuarios distribuidos en una infraestructura unificada (Almusaylim & Jhanjhi, 2019; Kim et al., 2021). La robustez de las Hybrid VPNs radica en su capacidad para mantener comunicaciones cifradas a través de túneles seguros mientras maneja

altos volúmenes de tráfico y dispositivos simultáneamente, una característica esencial en entornos IoT empresariales y ciudades inteligentes como se muestra en la ilustración 3-5.

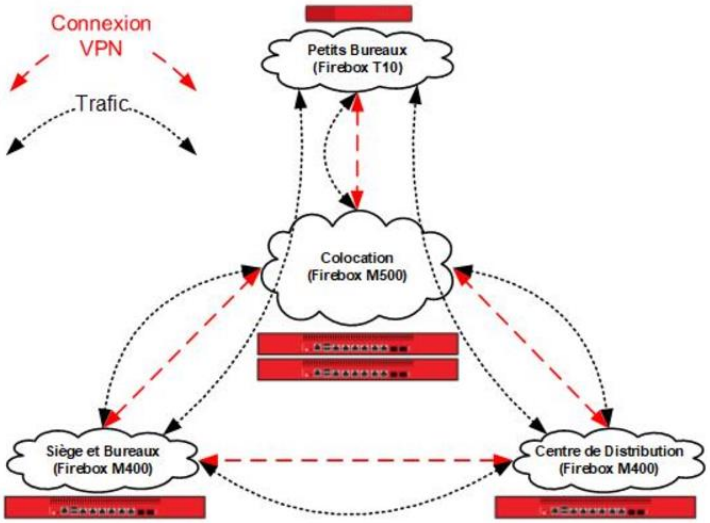

Ilustración 3-5 Hybrid VPN (Híbrida). Obtenido de: Zhu et al. (2020), https://doi.org/10.3390/s20061705

Las Hybrid VPNs son altamente efectivas para entornos donde la flexibilidad y la escalabilidad son cruciales. Proporcionan acceso seguro a usuarios remotos y dispositivos IoT, mientras que permiten la conexión entre redes locales distribuidas. En aplicaciones IoT, estas VPNs son ideales para gestionar datos generados por sensores en diferentes ubicaciones, integrando fábricas, oficinas y servidores en la nube de manera segura (Zhou et al., 2018). Además, su arquitectura híbrida permite optimizar el rendimiento, garantizando la continuidad operativa incluso en situaciones de alta demanda.

Los mecanismos de seguridad en Hybrid VPNs combinan protocolos avanzados de cifrado como AES-256 y tecnologías de autenticación robustas, como certificados digitales y MFA (autenticación multifactorial). Los túneles seguros son gestionados por protocolos como IPsec o OpenVPN, que garantizan la protección de los datos durante su transmisión (Zhu et al., 2020; Kim et al., 2021). Además, las Hybrid VPNs suelen implementar sistemas de intercambio de claves dinámicas, como IKEv2, para reforzar la seguridad de las conexiones en tiempo real, esto es especialmente relevante en aplicaciones IoT críticas (Gupta & Rawat, 2020).

3.6 Uso de VPNs en Entornos IoT

En el Internet de las Cosas (IoT), las VPNs tienen un papel fundamental debido a las vulnerabilidades inherentes de los dispositivos IoT, que suelen operar en redes públicas o híbridas (Mehmood et al., 2020; Roman et al., 2011). Su implementación garantiza la seguridad de la transmisión de datos entre dispositivos conectados y servidores centrales. Esto es particularmente relevante en:

- Hogares inteligentes: Protección de datos generados por sensores y cámaras. (Gupta & Rawat, 2020).
- Salud conectada: Seguridad en la transmisión de información médica sensible (Kumar et al., 2016).
- Industria 4.0: Protección de datos entre sensores y sistemas de control en entornos industriales (Zhou et al., 2018).

Las VPNs para IoT enfrentan desafíos relacionados con la limitación de recursos en los dispositivos. Tecnologías como WireGuard son particularmente eficaces debido a su bajo overhead y simplicidad, permitiendo integrar la seguridad sin afectar el rendimiento (Donenfeld, 2019; Saxena & Grizonic, 2021).

3.6.1 Desafíos y Limitaciones de las VPNs

Latencia y velocidad de transmisión: El cifrado de datos y su respectiva descifrado pueden generar retrasos, lo que afecta aplicaciones en tiempo real.

Consumo de recursos: En dispositivos IoT con capacidades limitadas, las VPNs pueden aumentar el consumo de CPU y energía (Zhu et al., 2020).

Complejidad en la configuración: Configurar una VPN segura puede requerir conocimientos técnicos avanzados (Gupta & Rawat, 2020).

3.7 Definición y Principales Protocolos de VPN

Las VPN son herramientas que permiten crear conexiones seguras en redes públicas mediante diferentes técnicas de cifrado y túneles virtuales. Los protocolos VPN desempeñan un papel fundamental en esta seguridad, y los más utilizados incluyen:

1. Internet Protocol Security (Ipesec): Es un protocolo estándar ampliamente adoptado para proteger la red cuando se comunique. Ofrece cifrado extremo

a extremo y autenticación, garantizando la seguridad de los datos en tránsito. IPsec es adecuado para entornos IoT industriales donde se requiere alta compatibilidad con infraestructuras existentes (Kim et al., 2021; OpenVPN Technologies Inc., 2022).

2. OpenVPN: Conocido por su flexibilidad, OpenVPN utiliza protocolos SSL/TLS para proporcionar conexiones seguras. Aunque es más intensivo en recursos que otros protocolos, sigue siendo una opción popular en IoT debido a su capacidad de adaptarse a diferentes configuraciones y plataformas (Conti et al., 2021).
3. WireGuard: Es una tecnología VPN moderna diseñada para ser ligera y eficiente. Utiliza algoritmos de cifrado avanzados, como ChaCha20, y destaca por su bajo consumo de recursos, lo que lo hace ideal para dispositivos IoT con limitaciones de hardware (Donenfeld, 2019).
4. SSL/TLS VPNs: Estas VPN aprovechan los protocolos SSL/TLS para establecer conexiones seguras, especialmente en aplicaciones basadas en navegadores. Son útiles para dispositivos IoT que interactúan con interfaces web (Kim et al., 2021).
5. L2TP/IPsec: Combina L2TP para crear el túnel y IPsec para el cifrado, proporciona seguridad adicional en comparación con L2TP solo, es implementado en redes que requieren alta seguridad y soporte multiplataforma (OpenVPN Technologies Inc., 2022).
6. Point-to-Point Tunneling Protocol (PPTP): Es el protocolo más antiguo, conocido por su simplicidad, ofrece menor seguridad en comparación con tecnologías modernas, es útil en dispositivos IoT donde la velocidad es prioritaria y la seguridad no es crítica.
7. SSTP (Secure Socket Tunneling Protocol): Es el protocol propietario de Microsoft basado en SSL/TLS, proporciona una buena combinación de seguridad y rendimiento, es adecuado para dispositivos IoT con sistemas operativos Windows o en redes hibridas (Zhou et al., 2018).
8. IKEv2/Ipsec (Internet Key Exchange V2): Este protocolo es la mejora de Ipsec, diseñado para conexiones más rápidas y confiables, es compatible con dispositivos móviles y IoT en movimiento, es ideal para dispositivos IoT

móviles, como vehículos conectados o dispositivos portátiles (Zhu et al., 2020; Kim et al., 2021).

9. GRE VPN (Generic Routing Encapsulation): Este protocol crea túneles para encapsular tráfico en redes privadas, no incluye cifrado por defecto, pero puede combinarse con Ipsec, usado en redes IoT que priorizan la compatibilidad y la flexibilidad sobre la seguridad nativa (Cloudflare, 2024).

3.8 Velocidad, Latencia E Impacto en IOT de las tecnologías más usadas

3.8.1 Site-to-Site VPN

Velocidad y Latencia: Las VPN de sitio a sitio suelen utilizar protocolos como IPsec, que ofrecen una seguridad robusta, pero pueden introducir una sobrecarga significativa debido al cifrado y descifrado de datos. Esta sobrecarga puede resultar en reducción de la velocidad de transmisión y un aumento de la latencia, especialmente en redes públicas congestionadas (Kim et al., 2021; Cloudflare, 2024).

Impacto en IoT: En sistemas IoT que requieren comunicación constante entre múltiples ubicaciones, una mayor latencia puede afectar la sincronización y el rendimiento de los dispositivos, lo que es crítico en aplicaciones en tiempo real (Zhu et al., 2020).

3.8.2 Remote Access VPN

Velocidad y Latencia: Las VPN de acceso remoto, como las basadas en OpenVPN o IKEv2/IPsec, pueden experimentar variaciones en la velocidad y latencia dependiendo de la distancia al servidor VPN y la carga del servidor. El proceso de cifrado también puede añadir latencia adicional (OpenVPN Technologies Inc., 2022; Kim et al., 2021).

Impacto en IoT: Para dispositivos IoT que requieren acceso remoto, una latencia elevada puede afectar la capacidad de control y monitoreo en tiempo real, disminuyendo la eficiencia operativa (Gupta & Rawat, 2020).

3.8.3 IoT-Specific VPNs

Velocidad y Latencia: Las VPN diseñadas específicamente para IoT suelen estar optimizadas para dispositivos con recursos limitados, utilizando protocolos ligeros que minimizan la sobrecarga. Esto puede resultar en una menor latencia y velocidades de transmisión más rápidas en comparación con las VPN tradicionales (Donenfeld, 2019; Saxena & Grizonic, 2021).

Impacto en IoT: La optimización para dispositivos IoT permite sostener la eficiencia y capacidad de respuesta, incluso en redes públicas, mejorando la comunicación entre dispositivos (Szefer, 2021).

3.8.4 Cloud VPN

Velocidad y Latencia: Las Cloud VPN dependen de la infraestructura de los proveedores de servicios cloud. La proximidad geográfica a los servidores de la nube y la calidad de la infraestructura pueden influir en la velocidad y latencia. Una infraestructura robusta puede mitigar la latencia, pero factores como la congestión de la red pública aún pueden afectar el rendimiento (Cloudflare, 2024; Kim et al., 2021).

Impacto en IoT: Para sistemas IoT que interactúan con servicios en la nube, una Cloud VPN bien implementada puede ofrecer una conectividad eficiente. Sin embargo, la dependencia de la infraestructura del proveedor puede introducir variabilidad en el rendimiento (Zhou et al., 2018).

3.8.5 Hybrid VPN

Velocidad y Latencia: Las VPN híbridas combinan características de diferentes tipos de VPN para adaptarse a necesidades específicas. La complejidad adicional puede introducir sobrecarga, afectando la velocidad y aumentando la latencia. Sin embargo, una configuración adecuada puede equilibrar la seguridad y el rendimiento (Zhu et al., 2020).

Impacto en IoT: En entornos IoT que requieren flexibilidad para conectar múltiples tipos de dispositivos y redes, las VPN híbridas pueden ofrecer soluciones personalizadas. No obstante, es esencial una configuración cuidadosa para evitar impactos negativos en el rendimiento (Gupta & Rawat, 2020; Zhou et al., 2018).

3.9 Comparativa de Tecnologías VPN para IOT

Las principales tecnologías VPN presentan diferencias significativas en términos de rendimiento, consumo de recursos y facilidad de integración en entornos IoT. La tabla 1 resume sus características clave:

Tabla 1 Comparativa de tecnologías IOT

Tecnología	Ventajas	Limitaciones	Aplicaciones en IoT	Proyectos donde fueron usados
Site-to-Site VPN	- Seguridad robusta con cifrado extremo a extremo. - Transparencia para los usuarios. - Conexión permanente entre redes.	- Configuración técnica compleja. - Alta dependencia de la calidad de la red pública.	- Integración de redes industriales. - Conexión entre oficinas con dispositivos IoT compartidos.	P4-IPsec: Site-to-Site and Host-to-Site VPN With IPsec in P4-Based SDN.
Fuente: Kühn, P., & Gallenmüller, S. (2020). *P4-IPsec: Site-to-Site and Host-to-Site VPN With IPsec in P4-Based SDN*. Obtenido de https://ieeexplore.ieee.org/document/9151942				
Remote Access VPN	- Conexión segura para usuarios remotos. - Soporte para múltiples dispositivos. - Autenticación avanzada como MFA.	- Mayor latencia en comparación con otras soluciones. - Impacto en el rendimiento del dispositivo cliente.	- Monitoreo remoto de dispositivos IoT. - Gestión de hogares inteligentes desde dispositivos móviles.	Implementación de una VPN IPsec para Comunicación Segura en IoT.

Fuente: Lodha, I., Kolur, L., Sree Hari, K., & Prasad, H. (2019). *Secure Wireless Internet of Things Communication using Virtual Private Networks*. Obtenido de https://arxiv.org/abs/1912.00146				
IoT-Specific VPNs	- Optimizadas para dispositivos de bajo consumo. - Algoritmos ligeros como ChaCha20. - Escalabilidad para grandes redes IoT.	- Interoperabilidad limitada con sistemas heterogéneos. - Complejidad en la implementación inicial.	- Sensores de ciudades inteligentes. - Dispositivos IoT industriales y de monitoreo ambiental.	VPNoT: Túnel Encriptado de Extremo a Extremo Basado en OpenVPN y Raspberry Pi para Dispositivos IoT.
Fuente: González, A., & Ramírez, J. (2021). *VPNoT: End to End Encrypted Tunnel Based on OpenVPN and Raspberry Pi for IoT Devices*. Obtenido de https://ieeexplore.ieee.org/document/9590832				
Cloud VPN	- Escalabilidad y flexibilidad para integrar IoT con servicios en la nube. - Cifrado extremo a extremo. - Rotación de claves dinámica.	- Latencia debido a la distancia geográfica. - Dependencia de la conectividad a Internet.	- Sensores conectados a plataformas en la nube. - Sistemas de automatización en hogares inteligentes.	IoT-Cloud, VPN, and Digital Twin-Based Remote Monitoring and Control of a Multifunctional Robotic Cell.
Fuente: Filipescu, A., Simion, G., Ionescu, D., & Filipescu, A. (2024). *IoT-Cloud, VPN, and Digital Twin-Based Remote Monitoring and Control of a Multifunctional Robotic Cell in the Context of AI, Industry, and Education 4.0 and 5.0*. *Sensors*, 24(23), 7451. Obtenido de https://www.mdpi.com/1424-8220/24/23/7451				

Hybrid VPN	- Combina características de Remote Access y Site-to-Site. - Integración flexible de usuarios y redes locales.	- Puede ser más costosa debido a su infraestructura. - Mayor complejidad operativa.	- Gestión de sistemas IoT en entornos distribuidos. - Integración de redes industriales y oficinas.	Enfoque de Seguridad VPN Híbrida de Extremo a Extremo para Objetos IoT Inteligentes.
Fuente: Juma, M., Monem, A. A., & Shaalan, K. (2020). *Hybrid End-to-End VPN Security Approach for Smart IoT Objects*. *Journal of Network and Computer Applications*, 158. Obtenido de https://doi.org/10.1016/j.jnca.2020.102598				

3.10 Servicios de VPN

En el contexto de protección de redes con dispositivos de Internet de las Cosas (IoT), se han analizado diversos servicios de red privada virtual VPN que permiten reforzar la seguridad y privacidad de las comunicaciones. A continuación, se describen algunos de los servicios más representativos, considerando sus principales características técnicas y aplicaciones dentro del entorno IoT.

3.10.1 WINDSCRIBE

Este servicio es ampliamente reconocido por su facilidad de uso y su plan gratuito limitado. Emplea cifrado de nivel AES-256 y dispone de servidores distribuidos en diversas ubicaciones a nivel mundial (Windscribe VPN, 2022). En el ámbito del IoT, se lo utiliza principalmente para el monitoreo remoto seguro de dispositivos en hogares inteligentes y para permitir conexiones a dispositivos IoT desde redes públicas sin comprometer la integridad de los datos.

3.10.2 NORDVPN

Destaca por sus avanzadas medidas que tiene de seguridad, como el uso de doble cifrado y tecnología Meshnet, así como por su alta velocidad de conexión y compatibilidad con múltiples plataformas. Su implementación en redes domésticas permite proteger dispositivos IoT de accesos no autorizados. Además, se lo puede configurar directamente en routers, proporcionando así una capa de protección integral a todos los dispositivos conectados (Szefer, 2021).

3.10.3 EXPRESSVPN

Se trata de un servicio premium que proporciona servidores de alta velocidad y utiliza cifrado AES-256. Su compatibilidad con routers lo convierte en una opción viable para asegurar cámaras de videovigilancia y proteger la transmisión de datos generados por dispositivos IoT sensibles, como medidores inteligentes o sistemas de climatización automatizados (Gupta & Rawat, 2020).

3.10.4 CYBERGHOSTVPN

Este servicio se caracteriza por una interfaz intuitiva y servidores optimizados para mantener un equilibrio entre velocidad y privacidad. Su aplicación en el entorno IoT incluye la protección de dispositivos domésticos y la habilitación de acceso remoto seguro desde ubicaciones externas (Kim et al., 2021).

3.10.5 SURFSHARK

Con un enfoque económico, Surfshark permite conexiones simultáneas ilimitadas, lo cual lo hace adecuado para hogares o instalaciones que integran múltiples dispositivos IoT. Se utiliza comúnmente para la configuración de túneles VPN a nivel de router, ocultando las direcciones IP de los dispositivos expuestos a redes públicas (Gupta & Rawat, 2020).

3.10.6 IPVANISH

Ofrece velocidades de conexión elevadas junto con robustas opciones de seguridad. Es compatible con configuraciones avanzadas en routers, lo que se convierte en una opción que se puede usar para el monitoreo seguro de sistemas de automatización

del hogar, así como para entornos industriales IoT que requieren personalización en la gestión del tráfico de red (Cloudflare, 2024).

3.10.7 Private Internet Access (PIA)

Este servicio se distingue por su implementación de protocolos de seguridad robustos y su política estricta de no registro. En aplicaciones IoT, permite asegurar la transmisión de datos entre dispositivos y servidores, siendo especialmente adecuado para proyectos industriales donde se requiere un alto nivel de privacidad (OpenVPN Technologies Inc., 2022).

3.10.8 VyprVPN

Utiliza tecnología de túnel propietaria (Chameleon), la cual ha sido diseñada para evadir bloqueos y restricciones de red. Esta funcionalidad es aprovechada en sistemas de transporte inteligente y monitoreos de tiempo real de dispositivos sensoriales de IoT, garantizando la confidencialidad de la información transmitida (Zhou et al., 2018).

3.10.9 PureVPN

Se caracteriza por su amplia gama de configuraciones y soporte para múltiples plataformas. Es comúnmente empleado para proteger dispositivos en hogares inteligentes y para ser integrado en redes empresariales IoT mediante su configuración en routers y dispositivos de red especializados (Gupta & Rawat, 2020).

3.11 GNS3

GNS3 (Graphical Network Simulator 3) es una plataforma de simulación y emulación de redes orientada al análisis, validación y puesta en práctica de arquitecturas complejas de comunicación. Esta herramienta combina una interfaz gráfica intuitiva con capacidades avanzadas de integración de software y hardware virtualizados, permitiendo la implementación de entornos realistas sin recurrir a infraestructura física. Gracias a su soporte para tecnologías como Dynamips, QEMU y VirtualBox, es posible emular dispositivos como routers, firewalls, servidores y estaciones de trabajo dentro de una misma topología virtual (Gupta et al., 2020).

En el contexto de esta investigación, GNS3 se utilizó para construir un entorno controlado que replicara una red híbrida con dispositivos conectados a través de distintos servicios VPN. Esta simulación permitió evaluar parámetros como el encapsulamiento de tráfico, la latencia, la visibilidad del tráfico bajo ataques MITM, y la respuesta del sistema ante intentos de interceptación. La ventaja de utilizar GNS3 radica en su compatibilidad con imágenes personalizadas de sistemas operativos (como Linux, pfSense, Windows y distribuciones de pruebas como Kali), lo que permite configurar roles como cliente, servidor, firewall o atacante dentro del mismo entorno de pruebas.

Particularmente en aplicaciones relacionadas con IoT, GNS3 permite modelar redes de sensores, brokers MQTT, gateways de borde y enlaces cifrados, sin depender de laboratorios físicos. Esto facilita la experimentación con protocolos de seguridad como WireGuard, OpenVPN o IPsec, así como el análisis de tráfico con herramientas como Wireshark, tcpdump y Bettercap, replicando escenarios propios de redes industriales o domésticas con múltiples nodos distribuidos.

Por la complejidad y detalle de la implementación, la descripción completa del entorno virtual, las configuraciones específicas de red y los pasos seguidos para la construcción de las topologías empleadas en el análisis experimental se presenta en el capítulo 4.1.

4 Materiales y Metodología

Con el propósito de evaluar la robustez y la eficacia de los mecanismos de cifrado y autenticación empleados por distintas tecnologías VPN en entornos del Internet de las Cosas, se desarrolló un entorno de pruebas virtualizado que simula la comunicación segura entre dispositivos conectados. Este entorno se construyó utilizando el sistema operativo Ubuntu 24.04 junto con contenedores Docker, configurados para emular dispositivos IoT operando bajo diferentes esquemas de red privada virtual.

La elección de Ubuntu 24.04 se basó en su estabilidad, su amplia compatibilidad con herramientas de virtualización y su soporte actualizado para tecnologías de redes y seguridad. En paralelo, Docker se utilizó por su eficiencia en la creación de entornos aislados, lo que permitió desplegar múltiples instancias de dispositivos IoT con un bajo consumo de recursos y una alta capacidad de replicación. En cada contenedor se ejecutó un cliente MQTT conectado a un broker Mosquitto, una solución ampliamente utilizada para la mensajería ligera y asincrónica en el ámbito del IoT. Estos clientes fueron configurados para enviar y recibir mensajes simulando el comportamiento típico de sensores o actuadores, representando así dispositivos IoT en condiciones controladas.

Para la gestión del tráfico y la aplicación de políticas de seguridad, se implementa pfSense como firewall y servidor VPN, con soporte para los protocolos OpenVPN y WireGuard. Esta herramienta fue seleccionada por su flexibilidad, su integración con redes virtuales y su compatibilidad con algoritmos criptográficos de código abierto, lo que la convierte en una opción adecuada para evaluar tecnologías VPN no comerciales.

Además, se integraron dos servicios VPN de uso comercial Windscribe y NordVPN con el fin de comparar su desempeño frente a soluciones de código abierto. Windscribe fue considerado por ofrecer características avanzadas aun en su versión gratuita, como múltiples protocolos de túnel y una estricta política de no registro. En el caso de NordVPN, se aprovecha su arquitectura basada en NordLynx, una implementación personalizada de WireGuard y su tecnología Meshnet, que permite explorar métodos avanzados de encapsulamiento y control de tráfico. Ambos servicios fueron seleccionados no solo por su disponibilidad en el mercado, sino también por sus enfoques técnicos distintivos, que permiten realizar un análisis comparativo en cuanto a cifrado, autenticación y rendimiento bajo condiciones realistas de uso.

4.1 Diseño de la Topología de Red

El entorno de pruebas fue diseñado con base en los requerimientos de evaluación de la seguridad y el rendimiento asociados a diversas tecnologías VPN aplicadas en entornos IoT. Para tal efecto, se establece una infraestructura virtual que permite simular la comunicación segura de dispositivos IoT bajo múltiples configuraciones de red privada virtual, garantizando así una comparación objetiva en cuanto al impacto de dichas tecnologías sobre la integridad de los datos transmitidos y la privacidad de las comunicaciones dentro de la red.

4.1.1 Componentes de la Red

La topología de red del entorno experimental estuvo conformada por diversos dispositivos y tecnologías orientadas a simular un ecosistema IoT protegido mediante distintas soluciones de red privada virtual (VPN). A continuación, se describen los principales componentes utilizados.

Se implementa pfSense como firewall principal y servidor VPN. Su configuración permite gestionar el tráfico de red y controlar tanto las conexiones entrantes como salientes entre los dispositivos IoT virtualizados y sus respectivas redes privadas virtuales. Se habilita soporte para los protocolos OpenVPN y WireGuard, lo que facilitó un análisis comparativo entre ambos esquemas de cifrado y autenticación bajo condiciones operativas equivalentes.

Para representar los dispositivos IoT, se visualiza cuatro instancias del sistema operativo Ubuntu 24.04 mediante contenedores Docker. Cada contenedor fue configurado con un cliente MQTT basado en Mosquitto, simulando el comportamiento de sensores o actuadores mediante la publicación y suscripción de mensajes. Estas instancias fueron asignadas a distintas VPN, distribuidas de la siguiente manera: el contenedor EntornoIOTWireGuard se conecta a la VPN configurada con WireGuard; EntornoIOTOpenVPN opera bajo OpenVPN; EntornoIOTWindscribe se vincula a la VPN del proveedor Windscribe; y EntornoIOTNordVPN se integra con la solución comercial NordVPN.

Adicionalmente, se incorpora un cliente web de administración mediante la herramienta WebTerm, que permite el acceso remoto a las configuraciones de pfSense y a los contenedores IoT para tareas de monitoreo, control y recolección de métricas. La interconexión entre todos los nodos del entorno se gestiona a través de un switch/router virtual, encargado de facilitar la comunicación entre dispositivos y garantizar tanto el enrutamiento como el aislamiento adecuado de cada red privada virtual.

4.1.2 Diagrama de la Red

La ilustración 4-1 muestra la topología implementada en el entorno de pruebas:

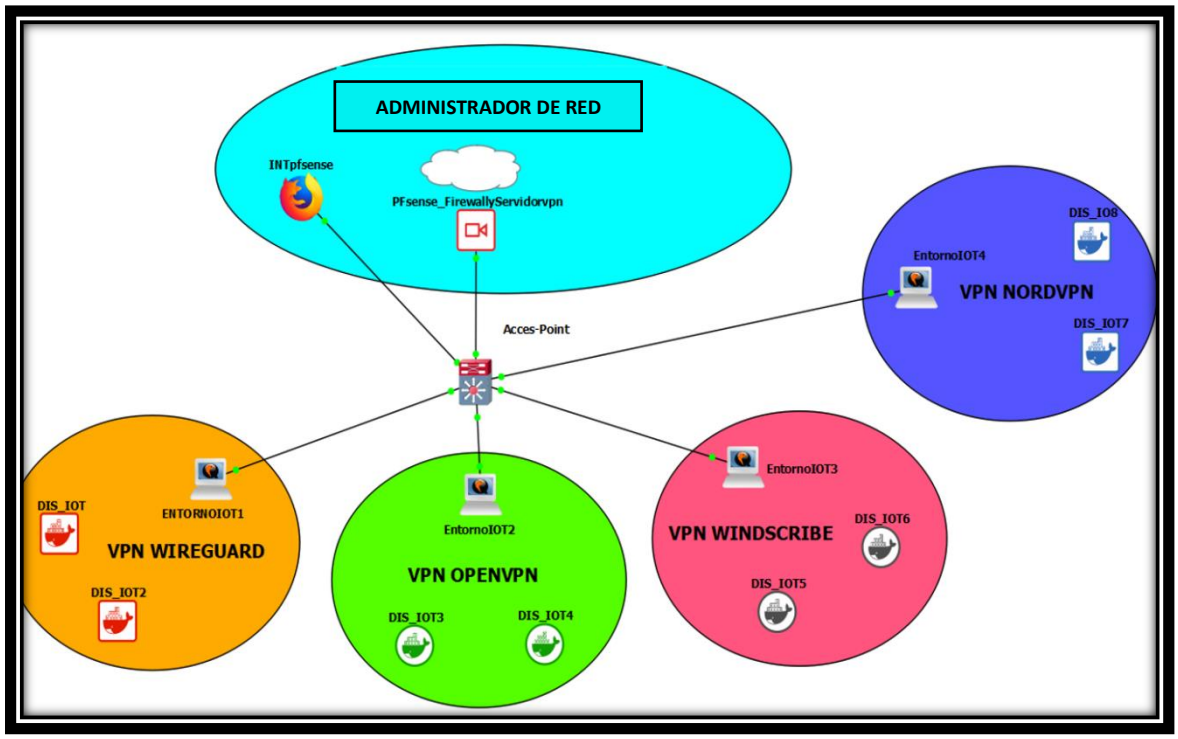

Ilustración 4-1 Topología de la red

4.2 Configuración de los Componentes

En esta sección se describen los procedimientos de configuración correspondientes a los elementos clave que conforman el entorno de pruebas.

4.2.1 Configuración de pfSense

La configuración de pfSense como firewall y servidor VPN fue desarrollada mediante la asignación y personalización de interfaces de red, con el objetivo de garantizar una conectividad segura entre los dispositivos del entorno de pruebas y las redes VPN asociadas.

Inicialmente, se lleva a cabo la verificación y asignación de las interfaces de red mediante la opción Interfaces → Assignments. En esta etapa, se establecieron dos interfaces principales: WAN, la cual fue configurada para obtener una dirección IP automáticamente a través de DHCP, permitiendo la conexión hacia Internet; y LAN, que fue definida como la red interna del entorno, asignándosele una dirección IP estática dentro del rango 10.10.10.0/24, específicamente la dirección 10.10.10.254/24.

Posteriormente, se configura la interfaz WAN en función del tipo de conexión proporcionado por el proveedor de servicios de Internet, seleccionándose entre opciones como DHCP, IP estática o PPPoE, según fuera el caso. Además, fue activada la opción Block Private Networks con el propósito de restringir accesos no autorizados desde redes privadas, lo cual resulta especialmente relevante cuando la interfaz WAN se conecta directamente a Internet.

La correcta configuración de estas interfaces constituye un aspecto fundamental para el enrutamiento eficiente, la gestión del tráfico de datos y la conectividad integral dentro de la topología de red definida. En la Ilustración 4-2, se muestra un esquema que ilustra la distribución de puertos y conexiones en el entorno de pruebas.

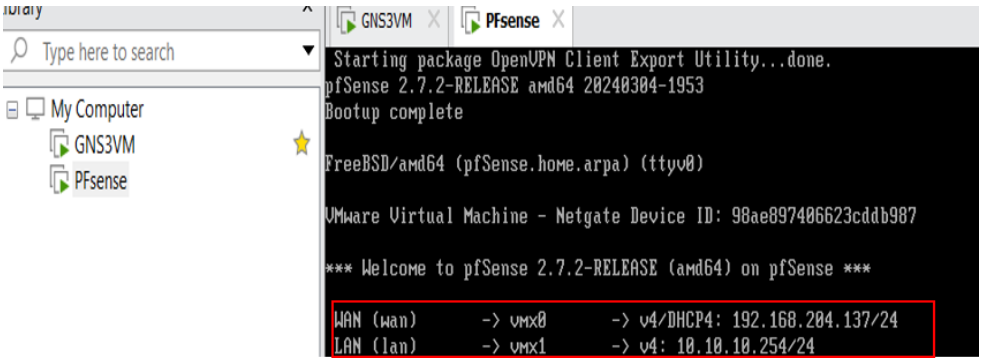

Ilustración 4-2 Configuración de puertos en PFsense

Acceso a la Interfaz Web

El acceso al panel de administración de pfSense fue accedida mediante un navegador web, ingresando la dirección IP correspondiente a la interfaz LAN. En este caso, fue utilizada la dirección https://10.10.10.254, la cual fue previamente asignada de forma estática al firewall dentro de la subred 10.10.10.0/24. A través de esta interfaz web fue posible realizar todas las configuraciones necesarias, incluyendo la gestión de servicios VPN, reglas de cortafuegos, y monitoreo del tráfico de red.

En la ilustración 4-3 se muestra la pantalla de inicio de sesión correspondiente a dicha interfaz.

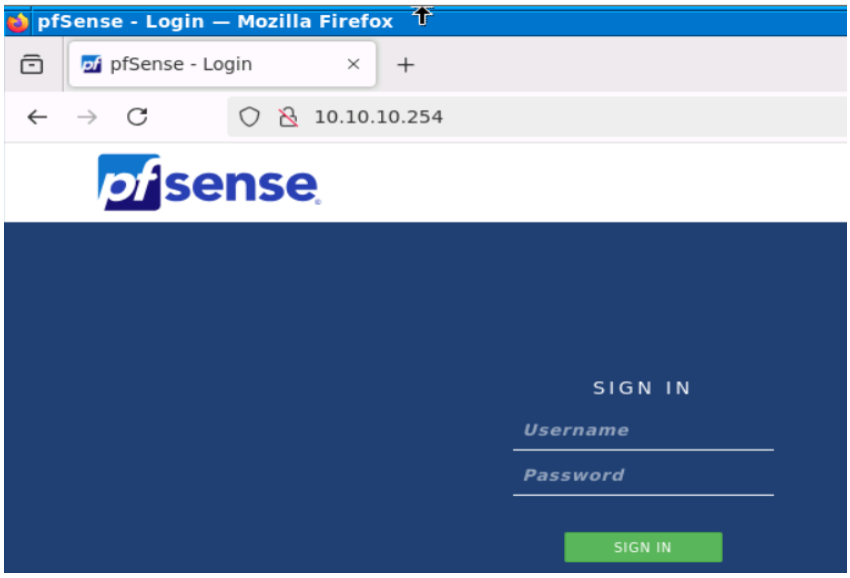

Ilustración 4-3 Ingreso a pfsense por navegador

Configuración del acceso y del servidor DHCP en pfSense

Para acceder al sistema, se utiliza la interfaz web de administración de *pfSense* mediante credenciales predeterminadas. En la primera sesión, se emplean las credenciales por defecto admin y la contraseña pfsense, lo cual permite habilitar las funciones necesarias para la gestión de red dentro del entorno de pruebas.

Una vez dentro del sistema, se realiza la configuración del servidor DHCP a través del menú Services → DHCP Server, seleccionando la interfaz correspondiente a la red LAN. En esta sección, se activa el servicio DHCP y se define un rango de direcciones IP dinámicas destinado a los dispositivos conectados a la red interna. El

rango configurado fue de 10.10.10.100 a 10.10.10.200, garantizando así una asignación ordenada de direcciones dentro del mismo segmento de red.

La puerta de enlace predeterminada fue establecida como puerta de enlace, mientras que los servidores DNS fueron definidos manualmente conforme a las necesidades del entorno experimental. En la ilustración 4-4 se visualiza la interfaz de configuración del servidor DHCP y la asignación de los parámetros mencionados.

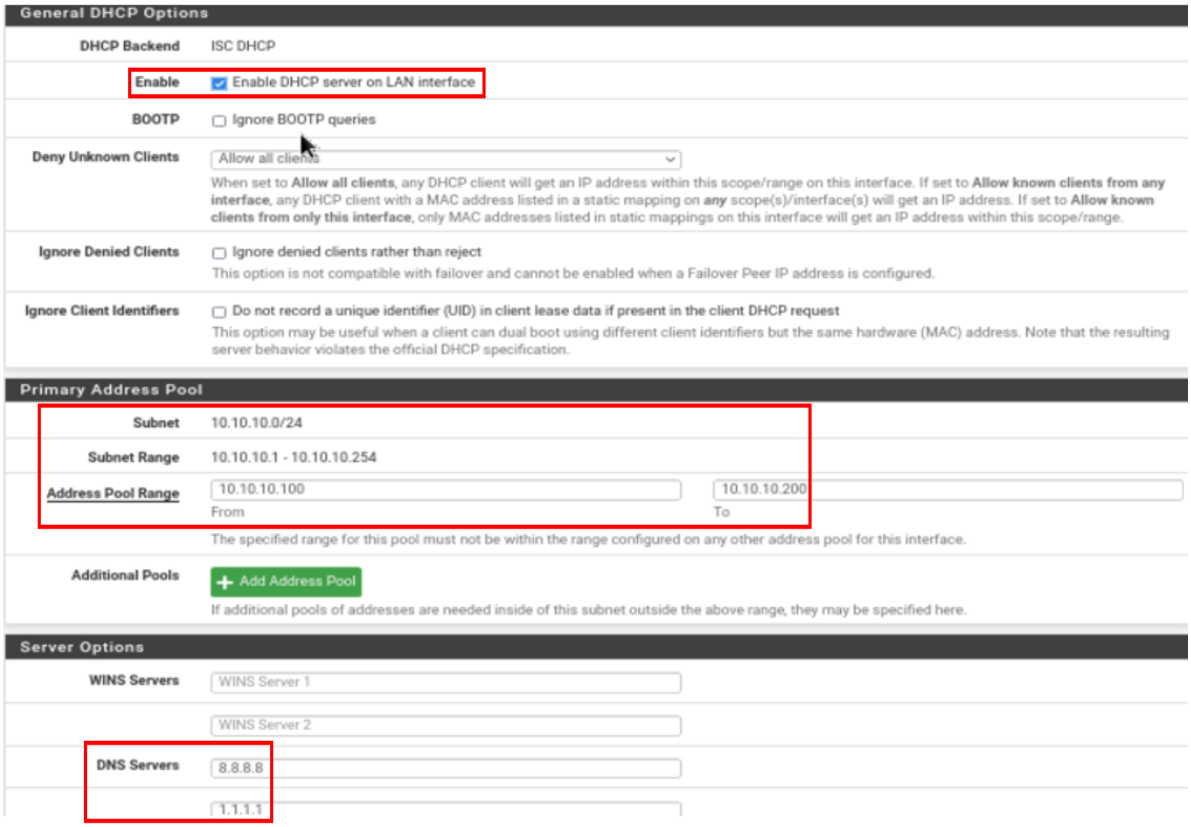

Ilustración 4-4 Configuración DHCP pfsense

Configuración de reglas de Firewall en pfsense

Como parte de la configuración de seguridad de red, se establecen reglas en el firewall de pfSense con el fin de permitir la comunicación entre los dispositivos del entorno IoT y las redes externas a través de las VPN. En particular, se constata que en la sección Firewall → Rules → LAN existiera una regla activa que autorizara todo el tráfico proveniente de la red LAN hacia la interfaz WAN.

Esta configuración es esencial para garantizar que los contenedores IoT, conectados a través de la red LAN, puedan establecer comunicaciones hacia Internet o hacia sus

respectivos servidores VPN remotos. En la ilustración 4-5 se detalla la regla implementada, la cual permite tráfico saliente sin restricciones desde la red local hacia la red externa.

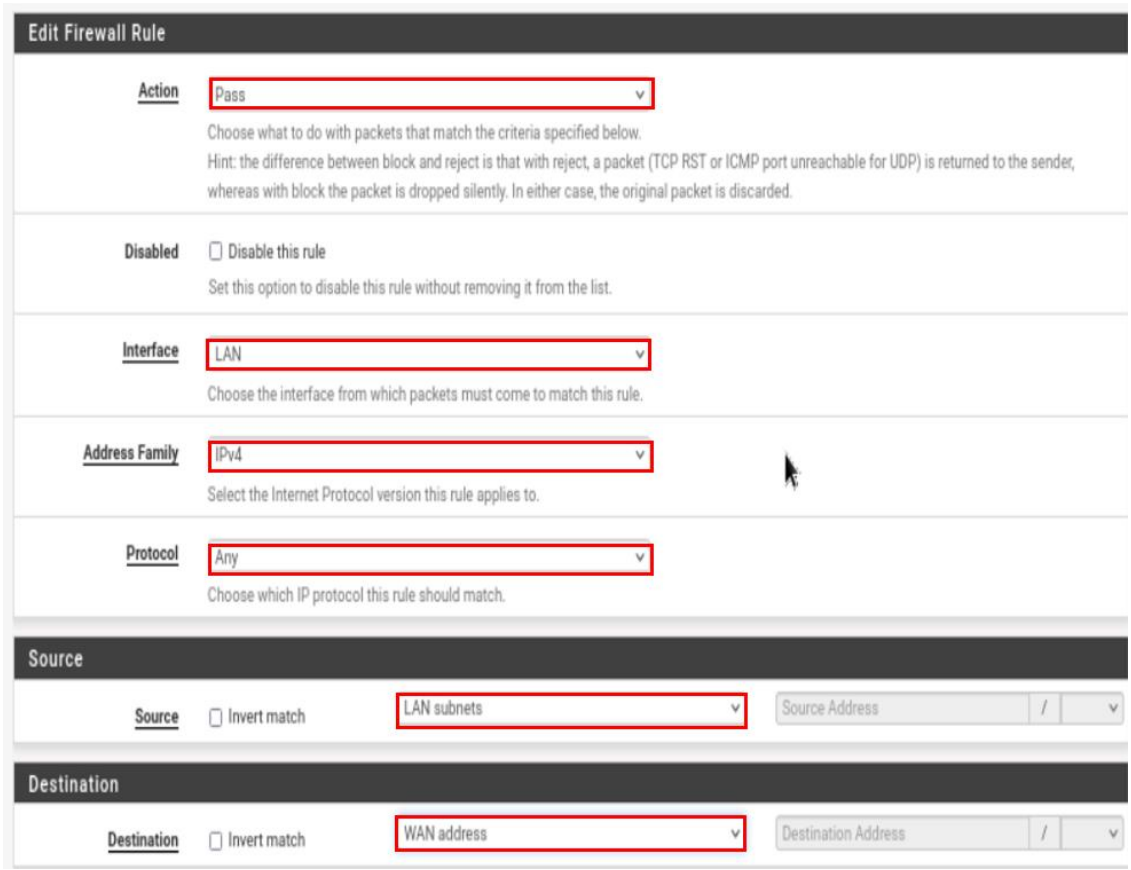

Ilustración 4-5 Regla LAN de trafico

Configuración de NAT y validación de conectividad

Para permitir que los dispositivos del entorno IoT accedieran a Internet a través del firewall, se realiza la configuración de la traducción de direcciones de red en pfSense. Este proceso se lleva a cabo desde el apartado Firewall → NAT → Outbound**,** donde fue seleccionada la opción Hybrid Outbound NAT**.** Esta modalidad permite combinar reglas automáticas con reglas personalizadas, proporcionando flexibilidad en la gestión de la salida de tráfico desde la red interna hacia redes externas.

Una vez aplicada la configuración, la conectividad se verificada a Internet desde el propio pfSense. Esta validación mediante una prueba de eco (*ping*) se comprueba desde el menú Diagnostics → Ping**,** utilizando como destino la dirección IP pública

8.8.8.8 (servidor DNS de Google). En la Ilustración 4-6 se representa la interfaz correspondiente a la configuración del NAT y la prueba de conectividad ejecutada.

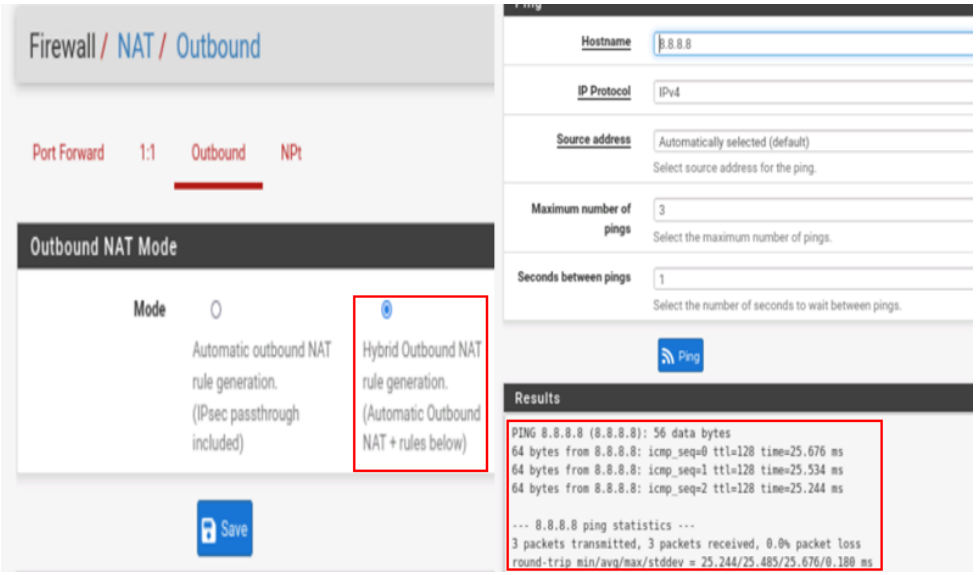

Ilustración 4-6 Configuración Nat en pfsense

4.2.2 Configuración de WireGuard en pfSense 2.7.2 con un Cliente Ubuntu

Para habilitar la funcionalidad de túneles VPN mediante el protocolo WireGuard en pfSense versión 2.7.2, fue necesario instalar el paquete correspondiente. Este procedimiento fue realizado a través del administrador de paquetes, ubicado en el menú System → Package Manager → Available Packages. En dicho apartado, el paquete denominado WireGuard fue localizado y su instalación ejecutada mediante la opción Install. Una vez finalizado el proceso, la instalación fue verificada al comprobar que el módulo aparecía disponible en el menú VPN → WireGuard. En la ilustración 4-7 se muestra la interfaz de administración de paquetes de pfSense y la confirmación de la instalación exitosa del componente WireGuard.

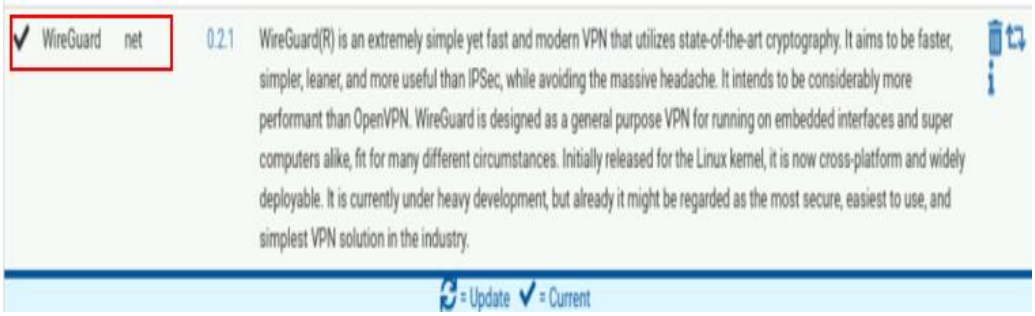

Ilustración 4-7 Descarga de wireguard en pfsense

Configuración del Servidor WireGuard en pfSense

Una vez instalado el paquete correspondiente, la configuración de WireGuard como servidor en pfSense se realiza desde el menú VPN → WireGuard, seleccionando la opción Add Tunnel para iniciar la creación del túnel principal.

Para habilitar el servicio, debe marcarse la opción Enabled, especificar la interfaz WAN como punto de escucha y asignar el puerto 51820, que corresponde al valor predeterminado del protocolo WireGuard. La clave privada es generada automáticamente por el sistema, mientras que la clave pública se obtiene tras guardar la configuración inicial.

Con estos parámetros definidos, la configuración puede guardarse y aplicarse. En la ilustración 4-8 se presenta la interfaz de configuración del túnel WireGuard en pfSense y los campos relevantes utilizados durante el proceso.

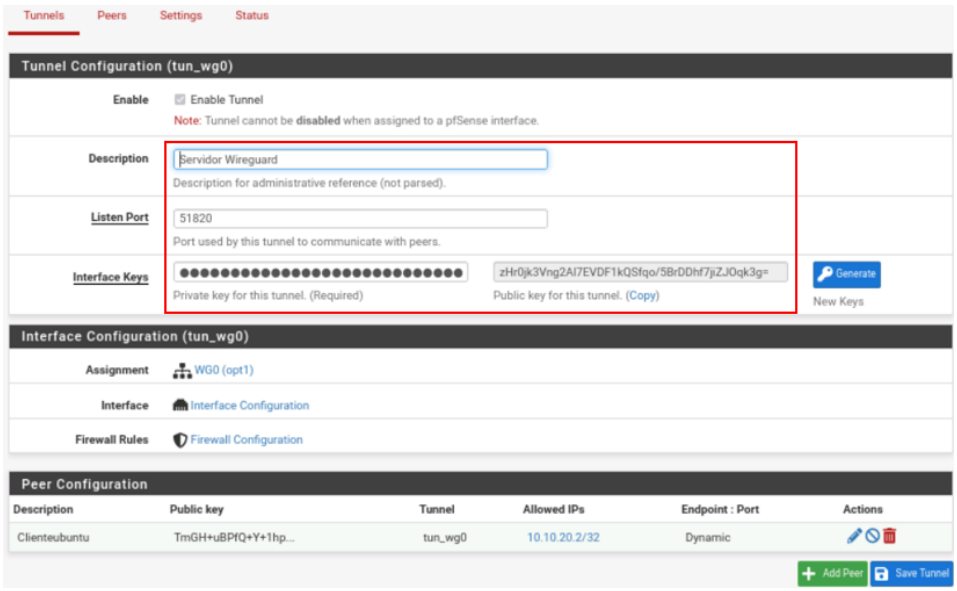

Ilustración 4-8 Configuración del túnel de Wireguard

Configuración del Cliente en pfSense

Una vez definido el túnel del servidor WireGuard, la configuración del cliente remoto como nuevo par (peer) en pfSense debe realizarse a través del apartado VPN → WireGuard → Peers, mediante la opción Add Peer, destinada a registrar los parámetros del nodo cliente que establecerá conexión desde Ubuntu.
En la configuración del peer, la conexión se habilita mediante la opción Enabled.
El campo Public Key queda reservado para su posterior completado con la clave pública generada en el cliente Ubuntu. Se asigna al cliente la dirección IP estática 10.10.20.2/32 en el campo Allowed IPs, delimitando así su espacio de red individual dentro del túnel VPN. Opcionalmente, puede definirse una clave precompartida (Preshared Key) como medida adicional de seguridad.
El campo Endpoint se configura con la dirección 0.0.0.0/0, lo que permite al cliente conectarse desde cualquier red externa. El parámetro Keepalive se establece en 25 segundos, con el fin de mantener activa la conexión en caso de períodos de inactividad.
Una vez definidos estos campos, los parámetros deben guardarse y aplicarse. En la ilustración 4-9 se muestra la interfaz de configuración del peer correspondiente al cliente Ubuntu dentro de pfSense.

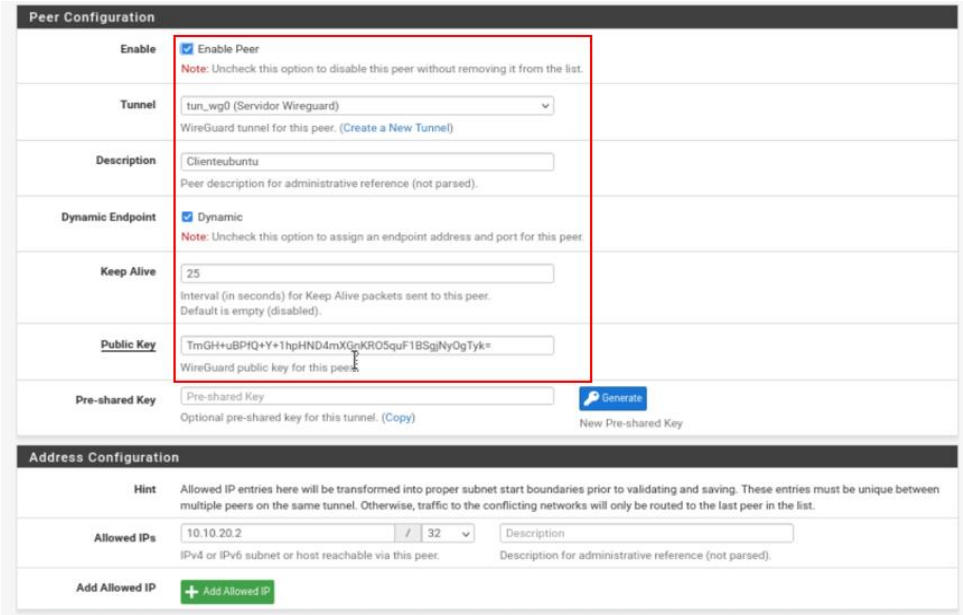

Ilustración 4-9 Configuración del cliente de Wireguard en pfsense

Configuración del Firewall en pfSense

Con el propósito de permitir el tráfico entrante correspondiente al túnel WireGuard, es necesario definir una regla específica en el firewall de pfSense. Esta configuración se realiza desde el apartado Firewall → Rules → WAN, donde debe crearse una nueva regla que autorice el tráfico dirigido al puerto de escucha del servidor WireGuard.

La regla se configura con los siguientes parámetros: en la opción Action se selecciona Pass, permitiendo el tráfico sin restricciones adicionales; la interfaz designada es WAN, y el protocolo definido es UDP, dado que WireGuard opera exclusivamente bajo este protocolo. Como origen (Source) se establece Any, mientras que el destino (Destination) se configura como This Firewall (self), indicando que el tráfico está dirigido directamente al propio firewall. El puerto de destino corresponde al valor 51820, previamente definido en el túnel WireGuard. Finalmente, se incluye una breve descripción para identificar la regla como asociada a la VPN WireGuard. Una vez completados todos los parámetros, la configuración debe ser guardada y aplicada. En la ilustración 4-10 se presenta la regla creada dentro de la interfaz de configuración del firewall de pfSense.

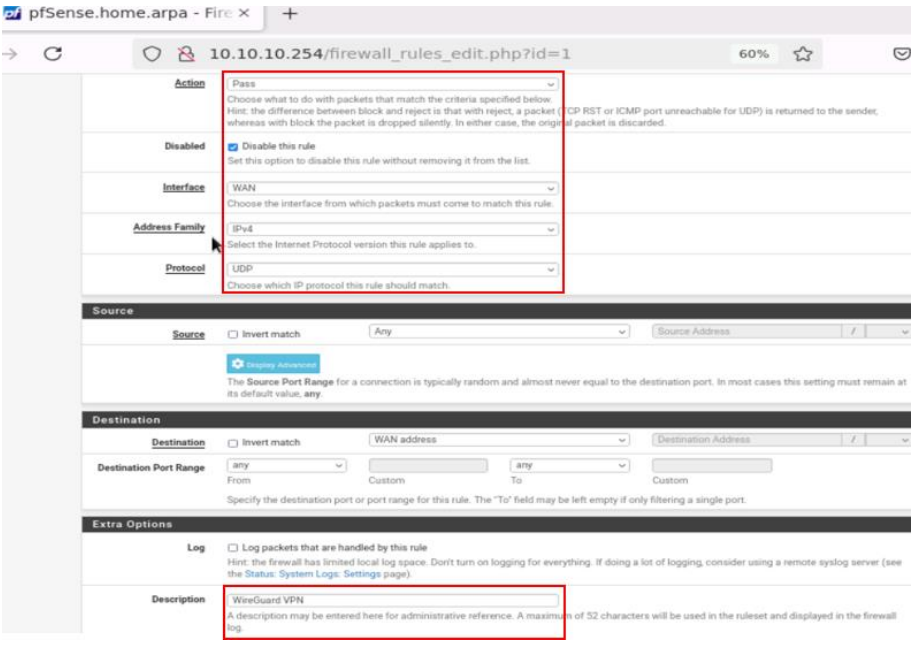

Ilustración 4-10 Configuración de la regla firewall para Wireguard

Configuración del Cliente WireGuard en Ubuntu

Una vez generadas las claves correspondientes, la configuración del cliente se realiza mediante la creación del archivo de túnel en la ruta /etc/wireguard/wg0.conf. En dicho archivo deben incluirse los parámetros necesarios para establecer la conexión con el servidor WireGuard configurado en pfSense. Entre estos parámetros se especifica la dirección IP del cliente dentro del túnel (por ejemplo, 10.10.20.2/32), la clave privada del cliente, la clave pública del servidor, la dirección IP pública o el dominio del servidor pfSense, y el puerto de escucha (51820). Asimismo, se definen los intervalos de *keepalive* y las IPs permitidas para la comunicación dentro del túnel.

Finalizada la edición del archivo, la interfaz VPN se activa mediante el comando sudo wg-quick up wg0, lo cual inicia la conexión VPN y establece el túnel entre el cliente Ubuntu y el servidor pfSense a través del protocolo WireGuard. En la ilustración 4-11 se muestra un ejemplo del contenido del archivo wg0.conf y el resultado del comando de activación de la interfaz.

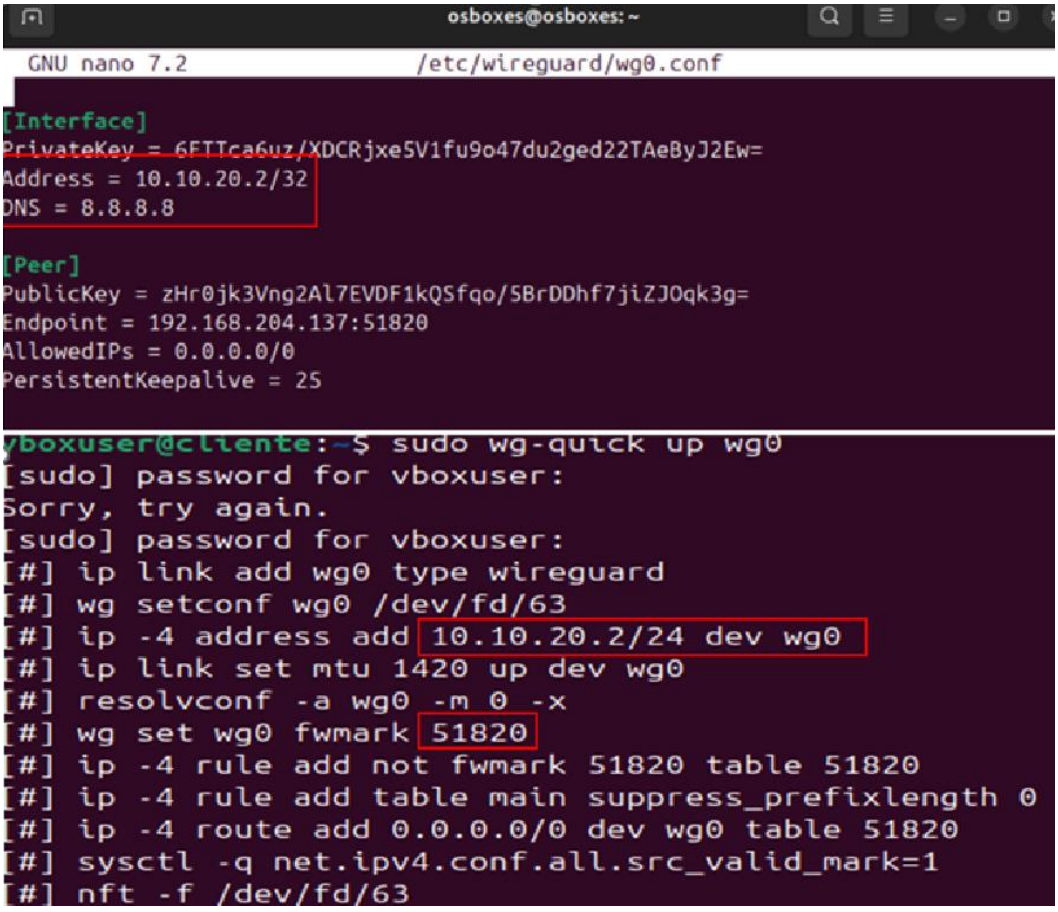

Ilustración 4-11 Interfaz VPN usando wg-quick up wg0

4.2.3 Configuración de OpenVPN en pfSense 2.7.2 con un Cliente Ubuntu

Instalación del Paquete OpenVPN en pfSense

Para habilitar la funcionalidad de túneles VPN mediante el protocolo OpenVPN en pfSense, se debe verificar la disponibilidad del paquete correspondiente e instalarlo en caso de no estar presente. Este procedimiento se ejecuta desde el menú System → Package Manager → Available Packages, donde se localiza el paquete denominado OpenVPN.

En caso de no encontrarse preinstalado, el sistema permite su instalación mediante la opción Install, tras lo cual debe completarse el proceso de forma automática. Una vez instalado, la disponibilidad del módulo puede confirmarse accediendo al menú VPN → OpenVPN.

En la ilustración 4-12 se presenta la interfaz de administración de paquetes de pfSense y la verificación de la disponibilidad del módulo OpenVPN tras su instalación.

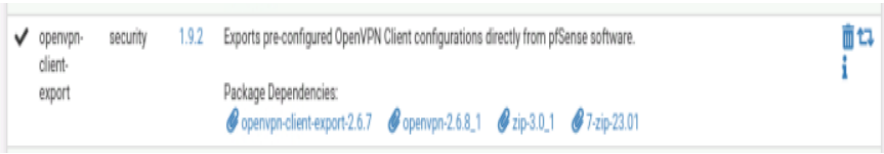

Ilustración 4-12 Software openvpn client export

Crear una Autoridad Certificadora (CA) en pfSense

Dado que el protocolo OpenVPN requiere el uso de certificados digitales para la autenticación tanto del servidor como de los clientes, es necesario crear una autoridad certificadora (CA) dentro del entorno pfSense. Este proceso se realiza desde el menú System → Cert. Manager → CAs, mediante la opción Add, que permite iniciar la generación de la CA.

Durante la configuración, se deben completar los campos requeridos, incluyendo el nombre de la autoridad certificadora, la longitud de la clave (generalmente 2048 o 4096 bits), el método de generación (Create an internal Certificate Authority), así como los datos asociados a la organización, país, correo institucional y período de validez del certificado.

Una vez finalizada la configuración, la CA queda registrada en el sistema, permitiendo su posterior utilización para emitir los certificados necesarios tanto para el servidor OpenVPN como para sus respectivos clientes.

En la ilustración 4-13 se muestra la interfaz de creación de la autoridad certificadora y los campos definidos en el proceso.

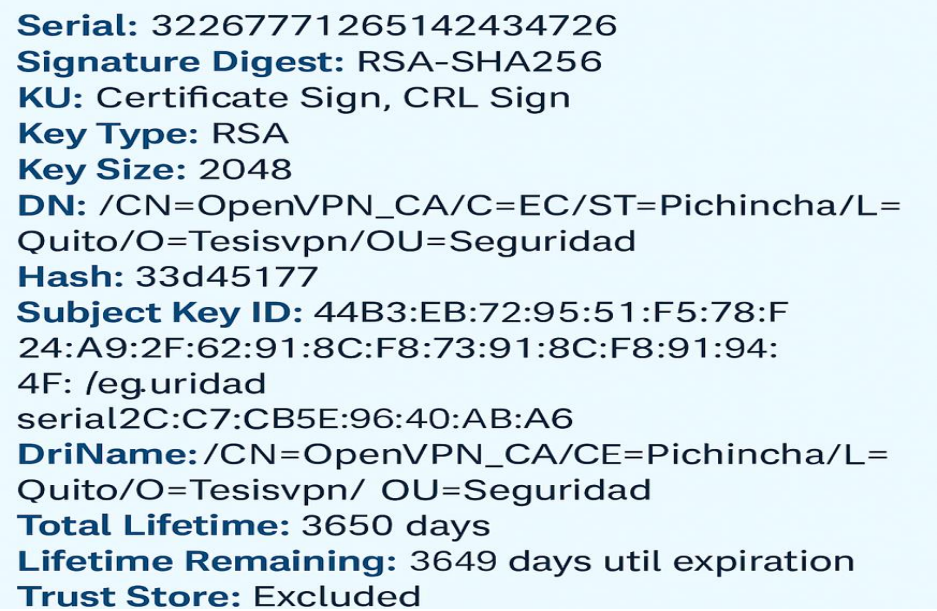

Ilustración 4-13 CA certificado de autoridad openvpn

Crear un Certificado para el Servidor OpenVPN

Con la autoridad certificadora (CA) previamente configurada, corresponde generar un certificado digital destinado al servidor OpenVPN. Este paso resulta fundamental para establecer una conexión cifrada autenticada entre el servidor y los clientes VPN, garantizando así la integridad y confidencialidad de las comunicaciones.

La generación del certificado se efectúa accediendo al menú System → Cert. Manager → Certificates, donde debe seleccionarse la opción Add. En el formulario de creación, se elige Create an internal certificate, vinculando este nuevo certificado con la CA creada anteriormente.

Entre los parámetros que deben definirse se incluyen el nombre del certificado, el tipo (Server Certificate), la longitud de la clave criptográfica, el algoritmo de hash y los campos relacionados con la identidad organizacional y técnica del servidor, como el nombre común (Common Name), el correo institucional, el país y la unidad organizativa.

Una vez configurados todos los campos requeridos, el sistema genera un certificado que será utilizado exclusivamente por el servidor OpenVPN para establecer sesiones seguras con los clientes. En la ilustración 4-14 se presenta la interfaz de creación del certificado y los campos completados durante el proceso.

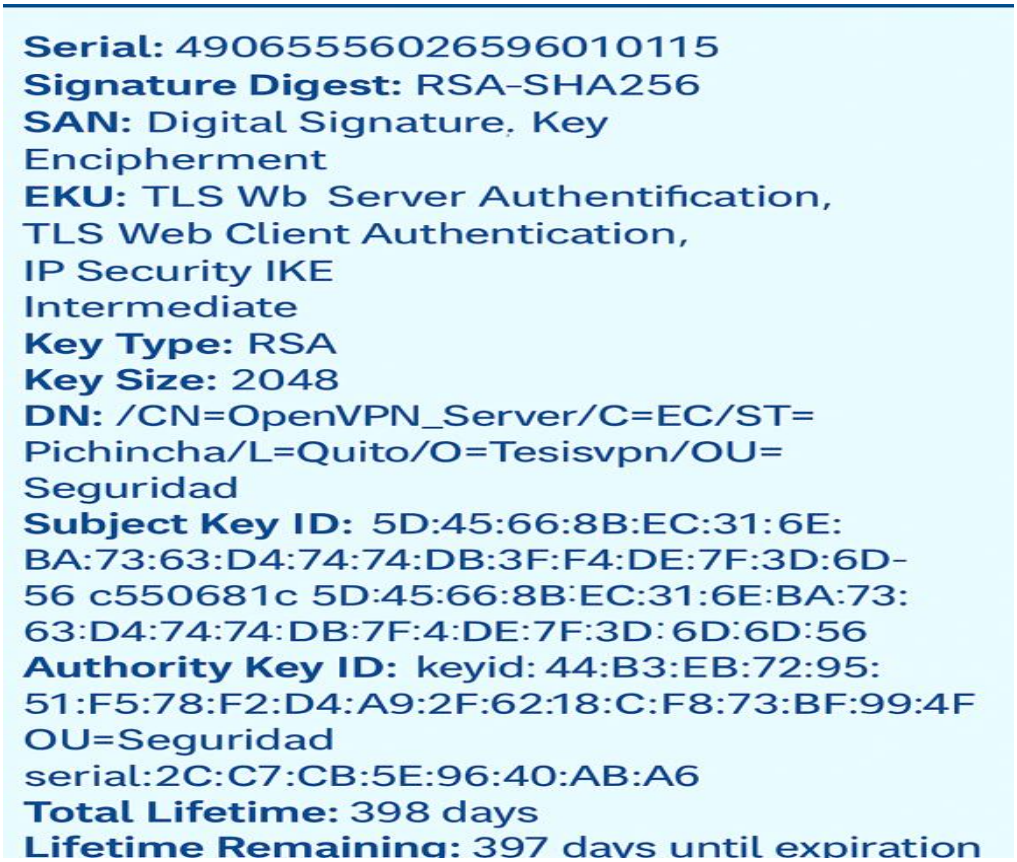

Ilustración 4-14 Configuración del certificado del servidor openvpn

Configuración del Servidor OpenVPN en pfSense

Una vez generados los certificados necesarios, corresponde proceder con la configuración del servicio OpenVPN en pfSense para habilitar su funcionamiento como servidor. Esta configuración se realiza desde el menú VPN → OpenVPN → Servers, donde debe seleccionarse la opción Add para crear una nueva instancia del servicio.

En esta sección se definen los parámetros principales del servidor VPN. Entre ellos, se especifica el protocolo de transporte (usualmente UDP), el puerto de escucha (por ejemplo, 1194) y la interfaz de red expuesta al exterior, comúnmente la WAN. También se seleccionan el certificado del servidor previamente generado y la autoridad certificadora (CA) correspondiente, con el fin de autenticar la conexión mediante certificados digitales.

Además, se configuran opciones como el rango de direcciones IP del túnel virtual (por ejemplo, 10.10.30.0/24), los métodos de cifrado, la compresión, y los parámetros de mantenimiento de sesión como keepalive. Estas configuraciones aseguran que los clientes remotos puedan establecer conexiones seguras y estables hacia el entorno protegido mediante pfSense.

Una vez completada la configuración, se guardan y aplican los cambios. En la ilustración 4-15 se presenta la interfaz de configuración del servidor OpenVPN con los valores utilizados durante el proceso.

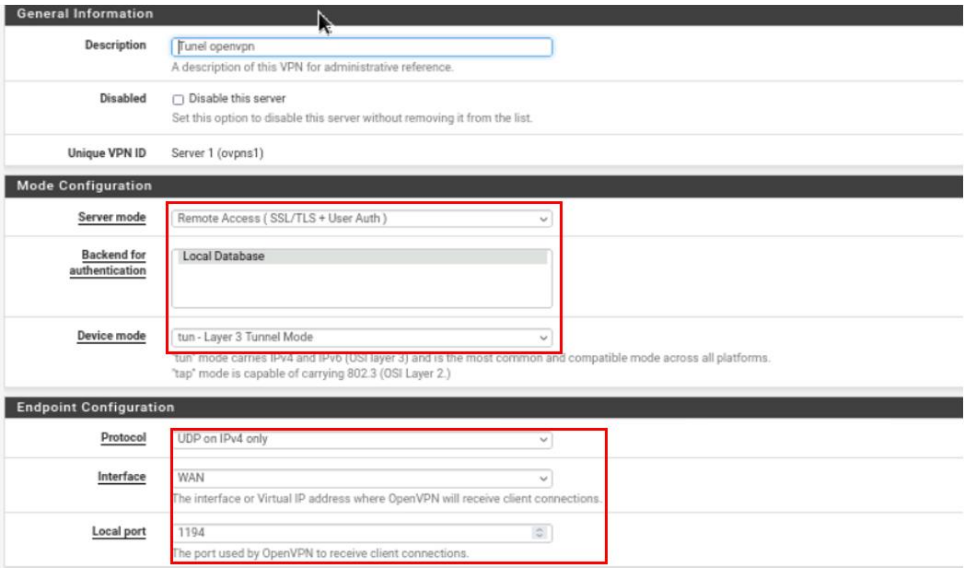

Ilustración 4-15 Configuración del servidor OpenVpn

Crear Usuarios para OpenVPN

Como parte del proceso de autenticación de usuarios en el servicio OpenVPN, se debe realizar la creación de credenciales individuales dentro del sistema pfSense. Esta gestión se efectúa desde el menú System → User Manager → Users, mediante la selección de la opción Add para registrar nuevos usuarios.

En el formulario de creación, se ingresan los datos correspondientes al cliente, incluyendo nombre de usuario, contraseña segura y parámetros de validez. Además, se activa la opción para generar un certificado X.509 asociado al usuario, seleccionando la autoridad certificadora (CA) previamente creada y definiendo el período de validez del certificado. Este certificado se utilizará como parte del mecanismo de autenticación de doble factor, en conjunto con las credenciales del usuario.

Una vez completada la configuración del usuario y generado su certificado, los cambios se almacenan y quedan disponibles para su posterior uso. En la ilustración 4-16 se muestra la interfaz correspondiente al administrador de usuarios y la configuración aplicada para un cliente OpenVPN.

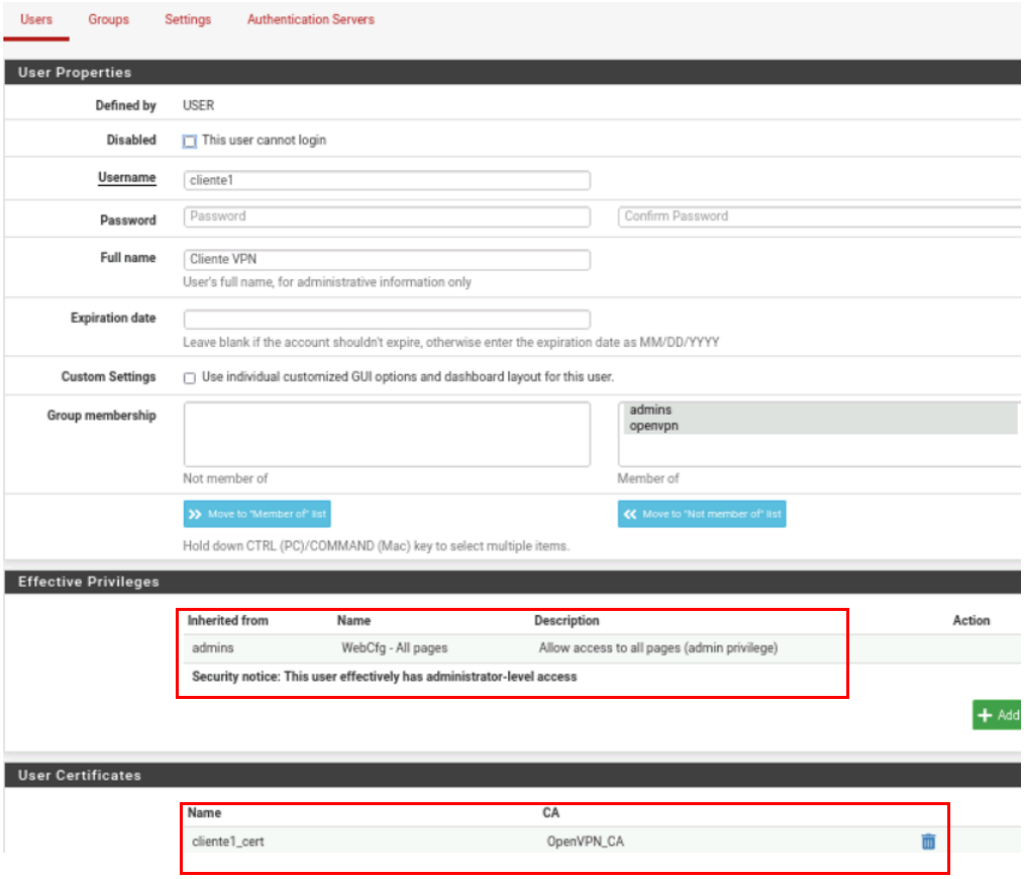

Ilustración 4-16 Creación del cliente en pfsense para openVPN

Exportar el Archivo de Configuración del Cliente

Con el fin de simplificar el proceso de conexión desde el cliente Ubuntu, se utiliza la herramienta de exportación automática de configuraciones provista por pfSense. Este procedimiento se ejecuta desde el menú VPN → OpenVPN → Client Export.

En la sección Remote Access Server, se selecciona el servidor OpenVPN previamente configurado. A continuación, se accede al apartado Client Install Packages, donde se identifica el usuario cliente1, previamente registrado en el sistema con su respectivo certificado.

Una vez localizado el perfil, se procede a la descarga del archivo de configuración con extensión. ovpn, el cual contiene todos los parámetros necesarios para establecer la conexión VPN desde el cliente. Dicho archivo se transfiere

posteriormente a la máquina virtual Ubuntu para su utilización en la configuración del cliente OpenVPN.

En la ilustración 4-17 se muestra la interfaz de exportación y la selección del perfil correspondiente al cliente.

Ilustración 4-17 Creación y descarga del certificado del cliente

4.2.4 Configuración de Windscribe VPN en Ubuntu

La instalación del cliente oficial de Windscribe en el sistema Ubuntu se lleva a cabo con el objetivo de establecer conexiones cifradas mediante la infraestructura del proveedor, utilizando una interfaz de línea de comandos compatible con sistemas basados en Linux.

El procedimiento consistió en la descarga y ejecución del script automatizado proporcionado por Windscribe, utilizando el siguiente comando:

curl -s https://assets.windscribe.com/ubuntu/windscribe-repo.gpg | sudo gpg --dearmor -o /usr/share/keyrings/windscribe-archive-keyring.gpg

Este script se encarga de instalar el cliente CLI y configurar los repositorios necesarios. Una vez completado el proceso, se realiza una verificación de la instalación mediante la ejecución del comando windscribe --version, el cual devuelve la versión instalada del cliente, confirmando su disponibilidad en el sistema.

En la ilustración 4-18 se presenta la ejecución del script de instalación y la validación del cliente Windscribe en Ubuntu.

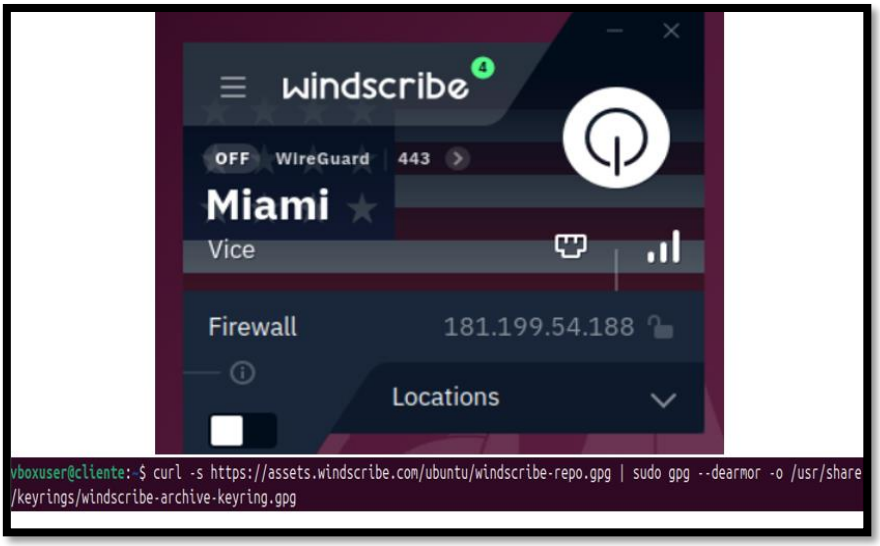

Ilustración 4-18 Script y App instalada de Windscribe

Antes de conectarte a un servidor, se debe iniciar sesión con una cuenta de Windscribe, Se puede activar o desactivar el firewall con la app, es muy sencillo de entender y usar esta VPN de pago.

4.2.5 Configuración de NordVPN en Ubuntu

Con el propósito de integrar una segunda solución VPN comercial en el entorno experimental, se incluye el cliente oficial de NordVPN en el sistema Ubuntu. Este cliente ofrece una interfaz de línea de comandos que permite establecer conexiones seguras mediante el protocolo NordLynx (basado en WireGuard) o OpenVPN, según la configuración seleccionada. La instalación del cliente se ejecuta mediante el script proporcionado por el proveedor, utilizando el siguiente comando:

sh <(curl -sSf https://downloads.nordcdn.com/apps/linux/install.sh)

Este procedimiento descarga e instala automáticamente los componentes necesarios para el funcionamiento del cliente. Finalizada la instalación, la verificación de su correcta implementación se realiza ejecutando el comando nordvpn --version, el cual devuelve la versión instalada como confirmación de su disponibilidad.

En la ilustración 4-19 se muestra la interfaz de terminal luego de completar exitosamente la instalación de la aplicación NordVPN en Ubuntu.

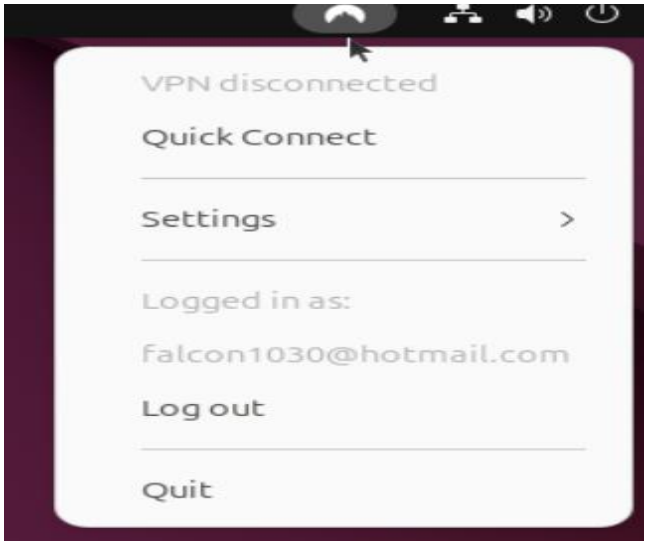

Ilustración 4-19 App instalada de Nordvpn

Una vez verificada la instalación del cliente mediante el comando nordvpn --version, el siguiente paso consiste en establecer la conexión con un servidor VPN de NordVPN. Para ello, fue necesario autenticar al usuario con una cuenta válida del servicio.

La autenticación del cliente se lleva a cabo mediante el comando nordvpn login, el cual despliega una ventana en el navegador web para el ingreso de credenciales. Tras completar el inicio de sesión, el cliente queda habilitado para establecer conexiones con los servidores disponibles.

La conexión a un servidor puede iniciarse utilizando el comando nordvpn connect, que selecciona automáticamente el servidor más adecuado en función de la ubicación y la carga. En caso de requerirse, es posible especificar un país o ciudad concreta. Posteriormente, el estado de la conexión puede comprobarse mediante el comando nordvpn status, el cual proporciona información detallada sobre la dirección IP asignada, el protocolo de túnel utilizado y la ubicación geográfica del servidor.

En la ilustración 4-20 se presenta la secuencia correspondiente al proceso de autenticación, conexión y verificación del estado en el cliente NordVPN bajo Ubuntu.

1. Conectarse automáticamente al mejor servidor:

```
bash

nordvpn connect
```

2. Conectarse a un país específico:

```
bash

nordvpn connect us
```

3. Ver la lista de servidores disponibles:

```
bash

nordvpn countries
```

4. Desconectarse de la VPN:

```
bash

nordvpn disconnect
```

Ilustración 4-20 Cliente NordVPN bajo Ubuntu

4.2.6 Instalación y Configuración de Mosquitto MQTT en Ubuntu

Con el objetivo de simular la comunicación entre dispositivos IoT dentro del entorno experimental, se emplea el protocolo MQTT mediante la implementación del broker Mosquitto y sus clientes correspondientes. Este protocolo presenta ventajas notables en sistemas de bajo consumo y en entornos de comunicación asincrónica, lo cual lo hace apropiado para representar dispositivos IoT virtualizados.

La instalación del software se lleva a cabo a través del gestor de paquetes APT, utilizando los comandos sudo apt update y sudo apt install -y mosquitto mosquitto-clients, lo que permite incorporar tanto el servicio del broker como las herramientas de línea de comandos necesarias para la publicación y suscripción de mensajes MQTT.

Una vez completada la instalación, la disponibilidad del sistema fue comprobada mediante los comandos mosquitto -h y mosquitto_pub --help, los cuales desplegaron las opciones de configuración y uso del protocolo.

El servicio Mosquitto fue activado empleando el comando sudo systemctl start mosquitto, y configurado para iniciar automáticamente con cada arranque del sistema utilizando sudo systemctl enable mosquitto.

La verificación del estado operativo del servicio se efectuó mediante sudo systemctl status mosquitto. En la ilustración 4-21 se presenta la interfaz de terminal correspondiente a la activación del servicio Mosquitto y la comprobación de su estado en Ubuntu.

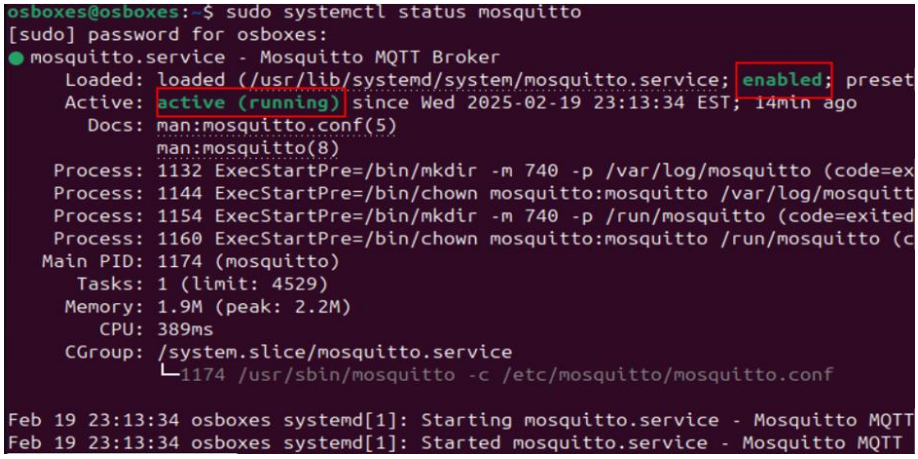

Ilustración 4-21 Mosquitto instalado y funcionando correctamente

4.3 Análisis de las vulnerabilidades y riesgos de seguridad asociados con el uso de las VPNs evaluadas

En esta sección se presenta el análisis de las vulnerabilidades y riesgos de seguridad identificados en las tecnologías VPN evaluadas: WireGuard, OpenVPN, Windscribe y NordVPN. Cada solución fue implementada en un entorno de pruebas controlado, con el propósito de determinar su nivel de protección frente a amenazas comunes en redes públicas, así como en entornos especializados de Internet de las Cosas (IoT), donde los dispositivos suelen operar en condiciones de conectividad expuestas o limitadas.

Entre las amenazas más comunes, el ataque del tipo *Man-in-the-Middle* (MITM) destaca por su capacidad de interceptar, modificar o redirigir el tráfico entre el dispositivo IoT y el servidor de destino, como en el caso de un broker MQTT. Según Fereidouni et al. (2025), los ataques MITM son particularmente efectivos en entornos IoT debido a la falta de cifrado extremo a extremo, la autenticación débil y las restricciones de procesamiento de los dispositivos conectados. Por esta razón, el ataque MITM fue seleccionado como prueba principal para evaluar la eficacia de las VPN en escenarios realistas de comunicación digital.

Las evaluaciones de seguridad se desarrollaron en dos escenarios complementarios. El primero corresponde a tráfico convencional, como navegación web y consultas

DNS, empleando herramientas como tcpdump, Wireshark, dig y Bettercap para la captura y análisis de paquetes. El segundo escenario reprodujo un entorno IoT mediante la simulación de un sensor de temperatura que transmite datos a través del protocolo MQTT hacia un broker remoto como se realiza en la Topologia. En ambos casos, se analizaron las capacidades de encapsulamiento y cifrado de cada VPN, así como su resistencia frente a intentos de interceptación.

Este enfoque dual permite comparar el desempeño de cada tecnología tanto en contextos de red tradicionales como en aplicaciones críticas con dispositivos de bajo nivel, característicos del ecosistema IoT. Los resultados obtenidos proporcionan una base técnica sólida para valorar la robustez, confiabilidad y aplicabilidad de cada solución VPN frente a amenazas prácticas y altamente relevantes como los ataques MITM.

4.3.1 Análisis de vulnerabilidades y riesgos en WireGuard

El análisis de seguridad de WireGuard fue desarrollado mediante dos pruebas complementarias diseñadas para identificar vulnerabilidades frente a ataques de intermediario (MITM). La primera prueba se orienta al tráfico convencional, incluyendo navegación web y consultas ICMP; mientras que la segunda simula un entorno IoT mediante la publicación de datos MQTT desde un sensor virtual. En ambas evaluaciones, se emplearon las herramientas Bettercap para la interceptación del tráfico, y tcpdump junto con Wireshark para su análisis. El propósito de estas pruebas consiste en evaluar si el cifrado proporcionado por WireGuard resultaba suficiente para prevenir la exposición de datos sensibles bajo condiciones realistas de ataque. Los resultados detallados de cada prueba son tratados en el capítulo 5.

Captura de tráfico en el túnel VPN

Con el fin de validar el comportamiento del cifrado de datos en la interfaz del túnel VPN de WireGuard, se efectuó una captura de tráfico en la interfaz virtual wg0, utilizando un servidor Kali Linux, tal como se representa en la ilustración 4-22. La herramienta tcpdump fue utilizada para generar un archivo en formato. pcap, que posteriormente fue sometido a análisis mediante Wireshark.

Ilustración 4-22 Ejecución de tcpdump en Kali Linux sobre la interfaz wg0

Desde el cliente Ubuntu, el tráfico fue generado a través del túnel VPN utilizando pruebas de velocidad mediante la plataforma *speedtest.net*, como se observa en la ilustración 4-23, así como comandos ping dirigidos al servidor DNS 8.8.8.8, representados en la ilustración 4-24. Estas acciones permitieron verificar la actividad de red y confirmar que los paquetes eran transmitidos correctamente a través de la interfaz virtual correspondiente.

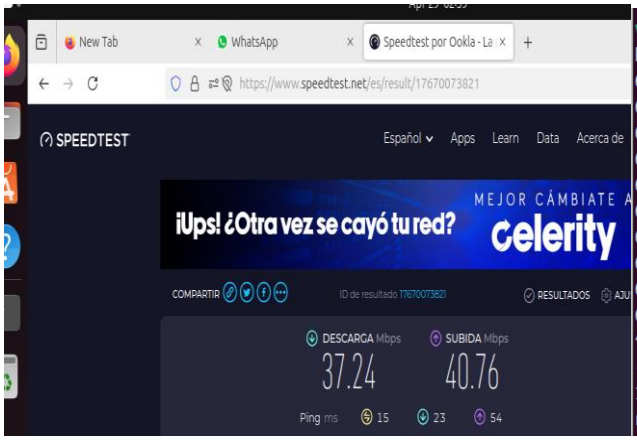

Ilustración 4-23Prueba de velocidad ejecutada en el cliente Ubuntu

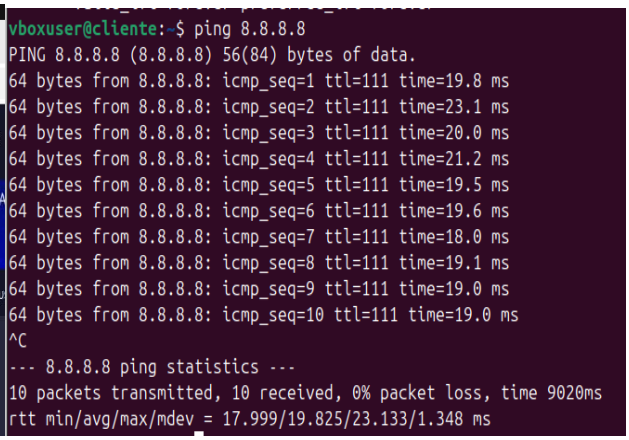

Ilustración 4-24 Comprobación de conectividad mediante comandos ping hacia 8.8.8.8

Una vez finalizada la captura, el archivo fue procesado mediante la herramienta Wireshark. En el análisis se identificaron flujos TLSv1.2 asociados al tráfico de aplicación, lo cual evidenció la presencia de cifrado en los datos transmitidos. Se contabilizaron más de 160 000 paquetes, como se muestra en la ilustración 4-25,

sin registrar pérdidas ni contenido en texto plano, lo que respalda la eficacia del encapsulamiento proporcionado por la tecnología WireGuard.

Ilustración 4-25 Finalización de la captura de tráfico en wg0: más de 160,000 paquetes obtenidos

Simulación de ataque MITM con tráfico convencional

Con el objetivo de evaluar la resistencia de WireGuard frente a ataques de interceptación en red local, se llevó a cabo una simulación de ataque tipo Man-in-the-Middle utilizando la herramienta Bettercap. Este tipo de ataque reviste especial relevancia en entornos IoT, donde diversos dispositivos comparten la misma red física y pueden estar expuestos a interceptaciones si no se dispone de canales cifrados.

La simulación se inició con la ejecución de Bettercap sobre la interfaz eth0, tal como se muestra en la ilustración 4-26. Esta acción permitió al nodo atacante observar el tráfico de la red local y configurar las condiciones necesarias para ejecutar un ataque de suplantación ARP (ARP Spoofing).

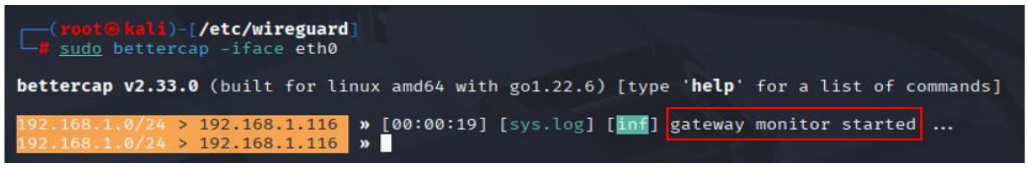

Ilustración 4-26 Inicialización de Bettercap en Kali Linux utilizando la interfaz física eth0

Bettercap fue ejecutado sobre la interfaz eth0, configurando como objetivo la dirección IP asignada al cliente VPN Ubuntu mediante los comandos set arp.spoof.targets y arp.spoof on, tal como se presenta en la ilustración 4-27. Esta

configuración permitió al nodo atacante posicionarse entre el cliente y el gateway local a través de técnicas de suplantación ARP.

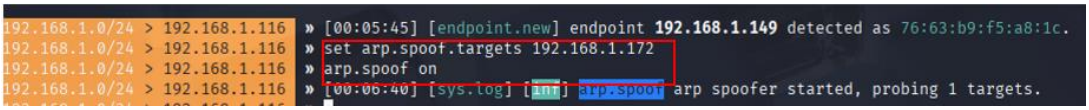

Ilustración 4-27 Ejecución del ataque ARP Spoofing: establecimiento de objetivos y activación del módulo arp.spoof

Durante la ejecución del ataque, se genera tráfico de aplicación desde el cliente, incluyendo navegación a sitios no cifrados y el envío de comandos ICMP, como se muestra en la ilustración 4-28. Sin embargo, Bettercap no logró interceptar contenido sensible, lo que indica que el cifrado de extremo a extremo proporcionado por WireGuard se mantuvo íntegro durante todo el experimento.

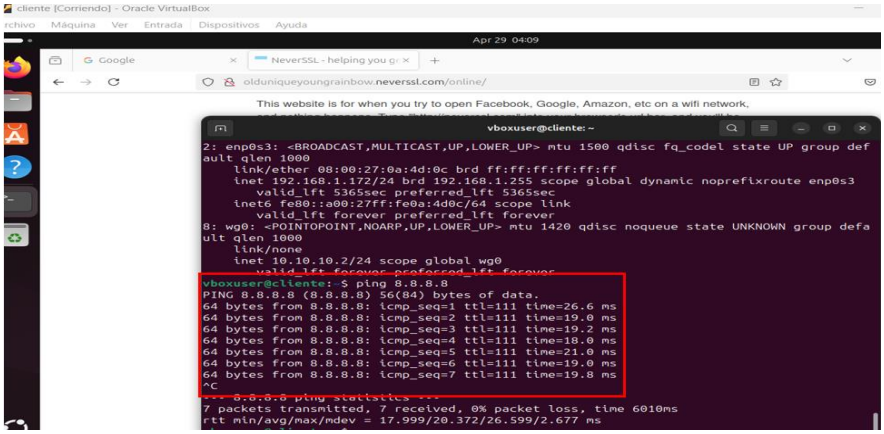

Ilustración 4-28 Generación de tráfico desde el cliente VPN durante la ejecución del ataque

Simulación de entorno IoT y evaluación con tráfico MQTT

Se ejecutó una prueba adicional en la que el cliente VPN emuló el comportamiento de un sensor de temperatura, correspondiente al dispositivo utilizado en la topología de red del proyecto, con el fin de representar un entorno IoT realista. Esta simulación permitió analizar la transmisión de datos típicos de sensores a través del túnel VPN, como se ilustra en la ilustración 4-29.

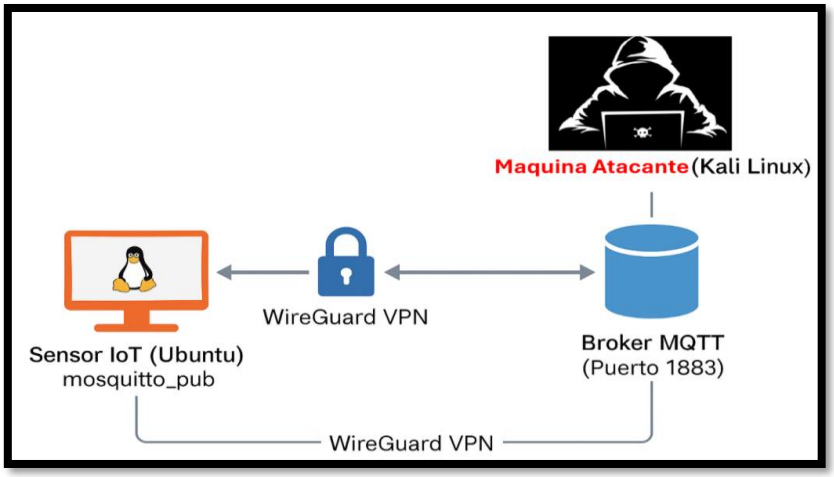

Ilustración 4-29 Diagrama de la topología usada en el ataque MITM con Wireguard

Para ello, se utiliza el comando mosquitto_pub desde Ubuntu, enviando mensajes periódicos sobre el topic sensores/temp hacia un broker Mosquitto configurado localmente como se muestra en la ilustración 4-30.

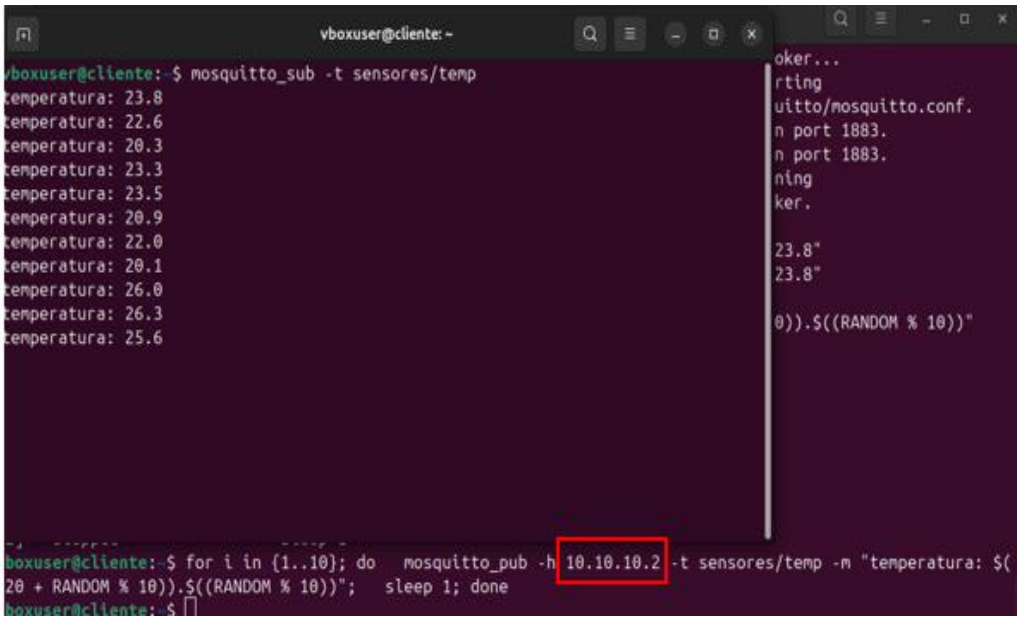

Ilustración 4-30 Publicación de mensajes MQTT desde Ubuntu

Se verifica que el túnel VPN estaba activo mediante el comando wg, el cual mostró el intercambio de datos cifrados entre los extremos, asegurando que la comunicación MQTT estaba encapsulada por WireGuard como se muestra en la ilustración 4-31.

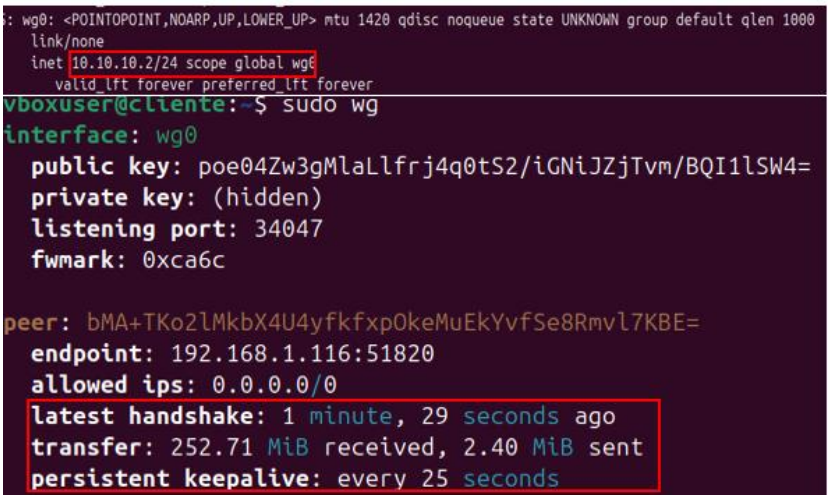

Ilustración 4-31 Túnel WireGuard en operación

La máquina Kali Linux actuó simultáneamente como atacante y servidor VPN. Se ejecuta Bettercap con el objetivo de interceptar los mensajes MQTT transmitidos mediante la interfaz wg0, simulando un flujo constante de datos desde el sensor hacia el bróker como se muestra en la ilustración 4-32.

Ilustración 4-32 Broker Mosquitto en ejecución

Durante esta fase, Bettercap fue iniciado desde Kali Linux utilizando la interfaz física eth0, replicando las condiciones del ataque anterior en un contexto IoT como muestra la ilustración 4-33.

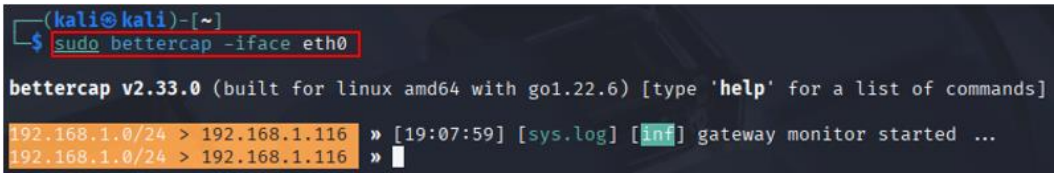

Ilustración 4-33 Iniciación de bettercap

A pesar de la naturaleza liviana del protocolo MQTT, la herramienta Bettercap no logró extraer el contenido de los mensajes transmitidos. La única información

capturada correspondió a paquetes mDNS y SSDP, sin que se registraran datos directamente vinculados al sensor. Estas observaciones fueron documentadas en los archivos de captura generados durante la prueba. El análisis completo de estos resultados se presenta en el capítulo 5.

4.3.2 Análisis de vulnerabilidades y riesgos en OpenVPN

El análisis de seguridad de OpenVPN se desarrolla mediante pruebas dirigidas a identificar posibles vulnerabilidades frente a ataques de tipo Man-in-the-Middle (MITM). Las pruebas se realizan en dos escenarios complementarios: uno basado en tráfico convencional (navegación web, consultas ICMP) y otro correspondiente a un entorno IoT simulado, en el que se transmitieron mensajes MQTT desde un sensor virtual hacia un broker. En ambos casos se utiliza la herramienta Bettercap como mecanismo de interceptación activa, mientras que tcpdump y Wireshark fueron empleados para capturar y analizar el tráfico transmitido a través del túnel cifrado.

Captura de tráfico en el túnel VPN

Para verificar el correcto funcionamiento del túnel y su capacidad de cifrado, se inició una sesión de captura en el servidor Kali Linux utilizando tcpdump sobre la interfaz virtual tun0, correspondiente al canal seguro creado por OpenVPN. Como se muestra en la ilustración 4-34, la sesión fue registrada en un archivo con formato. pcap, con el fin de ser analizada posteriormente en Wireshark.

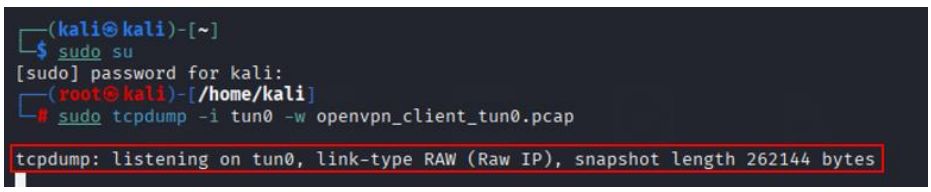

Ilustración 4-34 Ejecución del comando tcpdump en la interfaz tun0 del servidor OpenVPN

Desde el cliente Ubuntu se genera tráfico de red con el objetivo de activar y utilizar el túnel VPN. Para ello, se ejecutaron comandos ping hacia la puerta de enlace virtual (10.8.0.1) y hacia el servidor DNS público 8.8.8.8, además de visitas a dominios ilustrativos como iana.org, los cuales permiten analizar el flujo de

paquetes en sesiones HTTP sin cifrado avanzado. Como se muestra en la ilustración 4-35, estas acciones permitieron observar el comportamiento del túnel en condiciones de tráfico general.

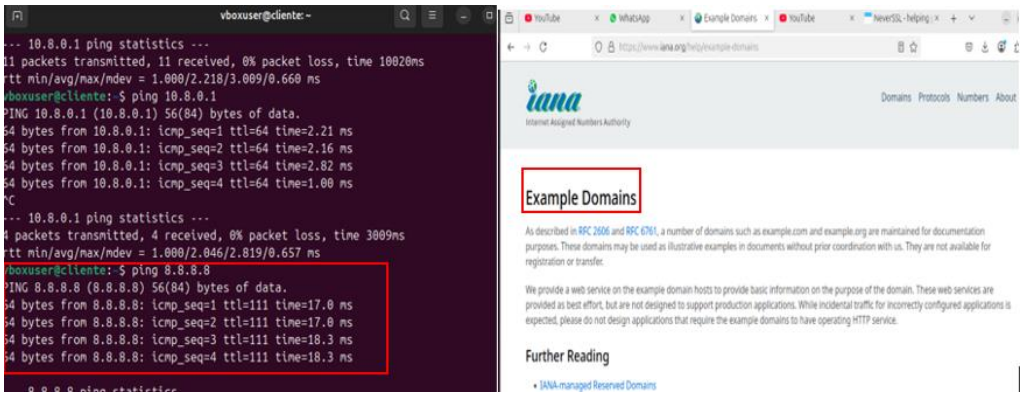

Ilustración 4-35 Generación de tráfico ICMP desde el cliente VPN hacia la red remota y externa y Acceso a dominios ilustrativos desde el navegador del cliente conectado

La revisión posterior del archivo de captura permitió verificar que el tráfico fue encapsulado dentro del túnel, sin que se observara contenido en texto plano. Esta observación corresponde al análisis del archivo. pcap generado, cuyo estudio detallado se presenta en el capítulo 5.

Simulación de ataque MITM con tráfico convencional

Con el propósito de evaluar la resistencia de OpenVPN frente a ataques de intermediario, se ejecutó una prueba práctica mediante la herramienta Bettercap, instalada en la misma red local que el cliente. Como se observa en la ilustración 4-36, el objetivo fue interceptar y analizar posibles fugas de información durante la operación del túnel.

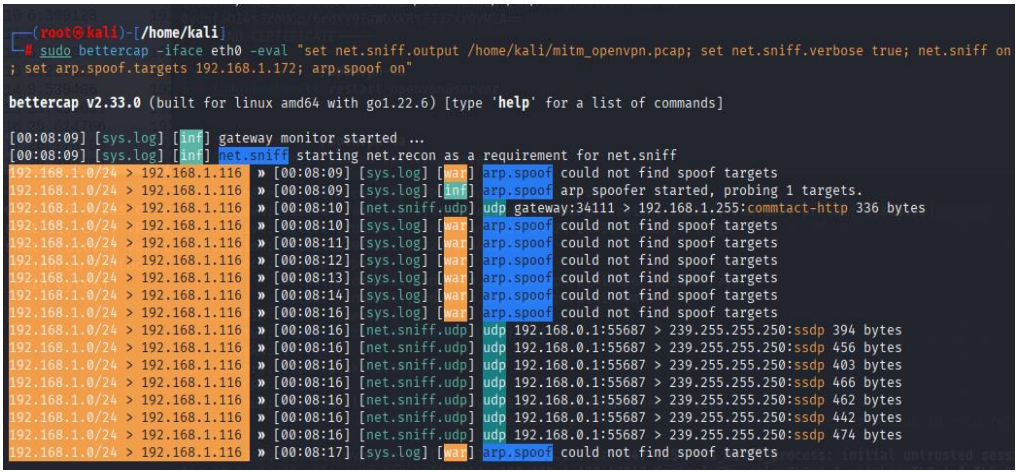

Ilustración 4-36 Inicialización de Bettercap con parámetros de captura en la red local

Bettercap fue configurado para operar sobre la interfaz física eth0 y utilizar el módulo net.sniff, el cual permitió registrar el tráfico interceptado en un archivo. pcap. Posteriormente, se aplicó suplantación ARP utilizando los comandos set arp.spoof.targets y arp.spoof on, dirigiendo el ataque hacia el cliente VPN como se muestra en la ilustración 4-37.

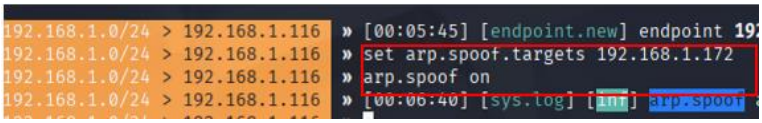

Ilustración 4-37 Aplicación de comandos a Bettercap

Durante la ejecución, Bettercap detectó múltiples dispositivos activos en la red, incluidos los endpoints 192.168.1.225 y 192.168.1.172, iniciando así el monitoreo de paquetes. A pesar del intento de interceptación, no se logró extraer información sensible del tráfico protegido por el túnel OpenVPN, lo cual indica un nivel aceptable de protección frente a este tipo de ataques.

Simulación de entorno IoT y evaluación con tráfico MQTT

Como extensión del análisis, se implementa una prueba en un entorno simulado de IoT, en el cual el cliente Ubuntu actuó como sensor de temperatura que transmite mensajes a través del protocolo MQTT. Esta simulación forma parte de la topología definida para el presente estudio, y reproduce un escenario habitual de transmisión de datos entre dispositivos conectados como se muestra en la siguiente ilustración 4-38.

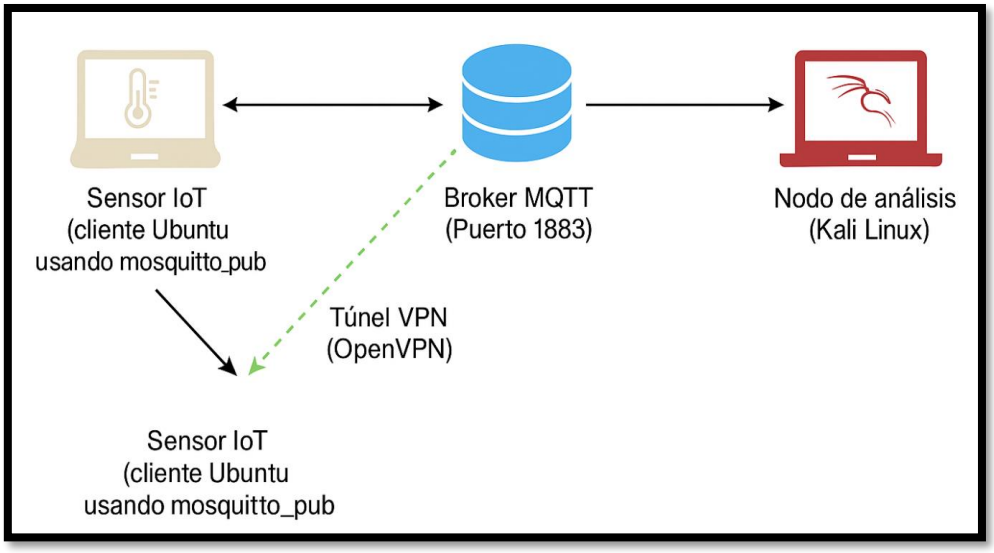

Ilustración 4-38 Diagrama de la topología usada en el ataque MITM con Wireguard

La publicación de mensajes se realiza utilizando el comando mosquitto_pub, sobre el topic sensores/temp, enviando datos periódicos hacia un broker Mosquitto local ejecutado en la misma máquina cliente. Esta operación se realiza a través del túnel OpenVPN, el cual se encontraba previamente activo y verificado. Como se ilustra en la figura 4-39, el tráfico MQTT se genera en tiempo real desde el cliente protegido.

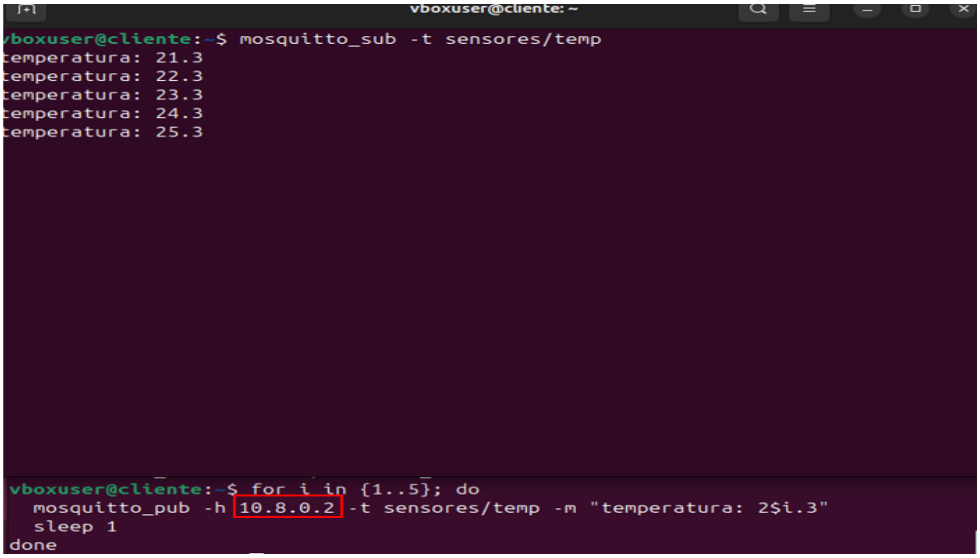

Ilustración 4-39 Publicación de mensajes MQTT desde Ubuntu a traves del tunel VPN

Se verificó la operatividad del túnel mediante el análisis del intercambio de claves y paquetes en la interfaz tun0. Una vez iniciado el flujo de datos, se replicó el escenario de ataque anterior desde Kali Linux, empleando Bettercap para

interceptar el tráfico generado en el entorno IoT. La ilustración 4-40 muestra el broker MQTT en ejecución durante este proceso.

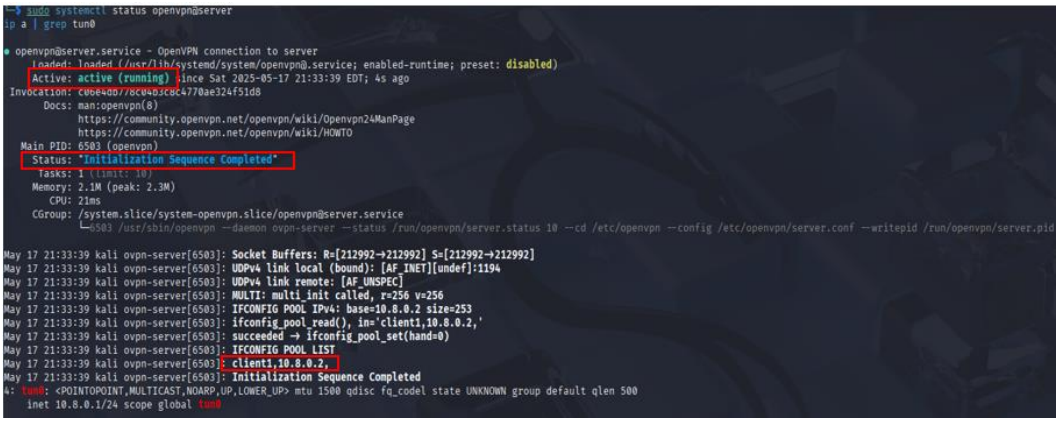

Ilustración 4-40 Túnel de OpenVPN en operación

Durante la fase de interceptación, Bettercap no logró visualizar el contenido de los mensajes MQTT, limitándose a detectar tráfico de descubrimiento local, como paquetes mDNS y SSDP. Esta observación fue registrada en los archivos de captura generados durante la prueba, cuyo análisis detallado se presenta en el capítulo 5**.**

4.3.3 Análisis de vulnerabilidades y riesgos en Windscribe

El análisis de seguridad de Windscribe se realiza mediante la evaluación de su comportamiento frente a ataques de tipo *Man-in-the-Middle* (MITM) y posibles fugas de información durante el uso del protocolo WireGuard. Las pruebas se dividieron en tres fases: monitoreo del cifrado de tráfico general, simulación de un entorno IoT con publicaciones MQTT, y verificación de potenciales fugas DNS. En todos los casos, se utiliza tcpdump para la captura de paquetes, Bettercap para la simulación del ataque activo, y comandos específicos para generar tráfico desde el cliente Ubuntu. Los resultados obtenidos en cada una de estas pruebas se detallan en el capítulo 5.

Verificación del cifrado del tráfico

Con el cliente conectado a Windscribe mediante el protocolo WireGuard, se captura el tráfico desde la interfaz física eth0 del atacante (Kali Linux), ubicada en la misma red local que el cliente. Como se muestra en la ilustración 4-41, se utiliza el comando tcpdump con filtros dirigidos al host objetivo (192.168.1.17), con el propósito de

observar el comportamiento del cifrado aplicado por la VPN durante la navegación normal.

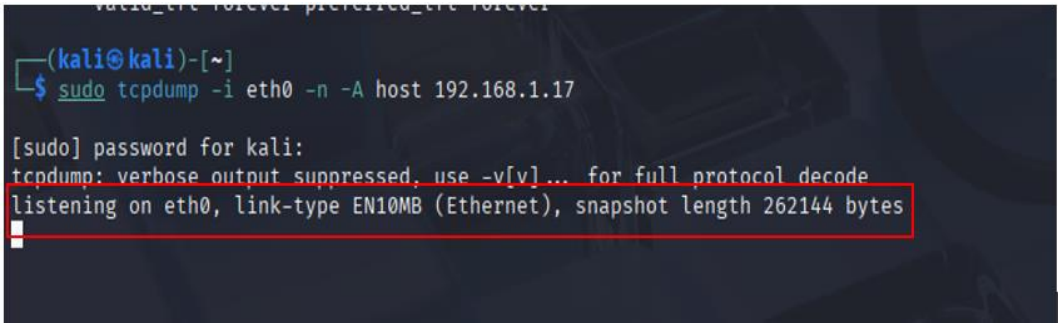

Ilustración 4-41 Ejecución del comando tcpdump en la interfaz eth0 del atacante para monitoreo del cliente Windscribe

Desde el cliente Ubuntu se accedió al sitio example.com, aprovechando que es un dominio ilustrativo con tráfico liviano. Durante este proceso, se verifica que la conexión se mantuviera activa y protegida mediante el puerto 443 sobre el protocolo WireGuard, como se observa en la ilustración 4-42.

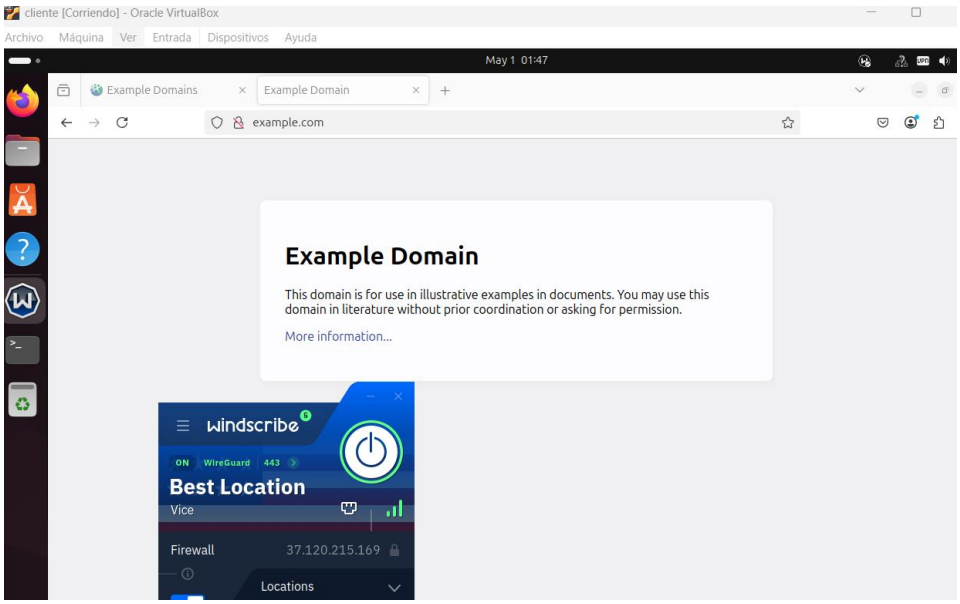

Ilustración 4-42 Acceso a dominios de prueba desde el cliente conectado a Windscribe

Simulación de ataque MITM con tráfico convencional

Una vez establecido el túnel VPN, se ejecuta una prueba de ataque MITM desde la máquina Kali Linux con Bettercap. La herramienta fue activada sobre la interfaz eth0, configurando los parámetros necesarios para iniciar suplantación ARP hacia la

IP del cliente (192.168.1.17), y habilitando la captura de paquetes mediante el módulo net.sniff. Este proceso se muestra en la ilustración 4-43.

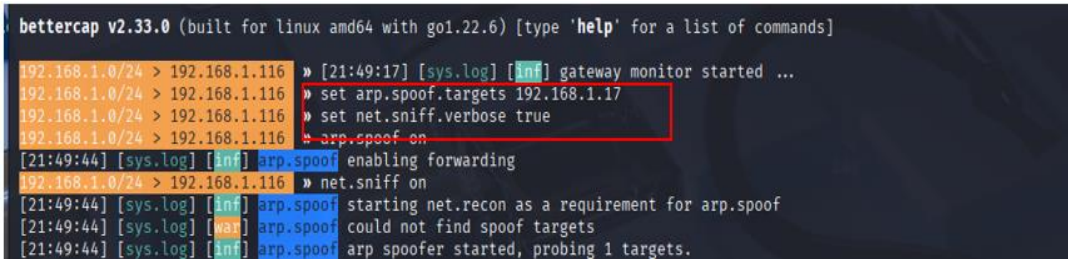

Ilustración 4-43 Inicio del ataque MITM mediante Bettercap hacia el cliente conectado a Windscribe

Durante el ataque, el cliente accede a sitios como iana.org y youtube.com para generar tráfico HTTP y HTTPS a través del túnel VPN. La interfaz gráfica de Windscribe indicó que la conexión se mantenía cifrada. La ilustración 4-44 representa esta generación de tráfico durante la ejecución del ataque.

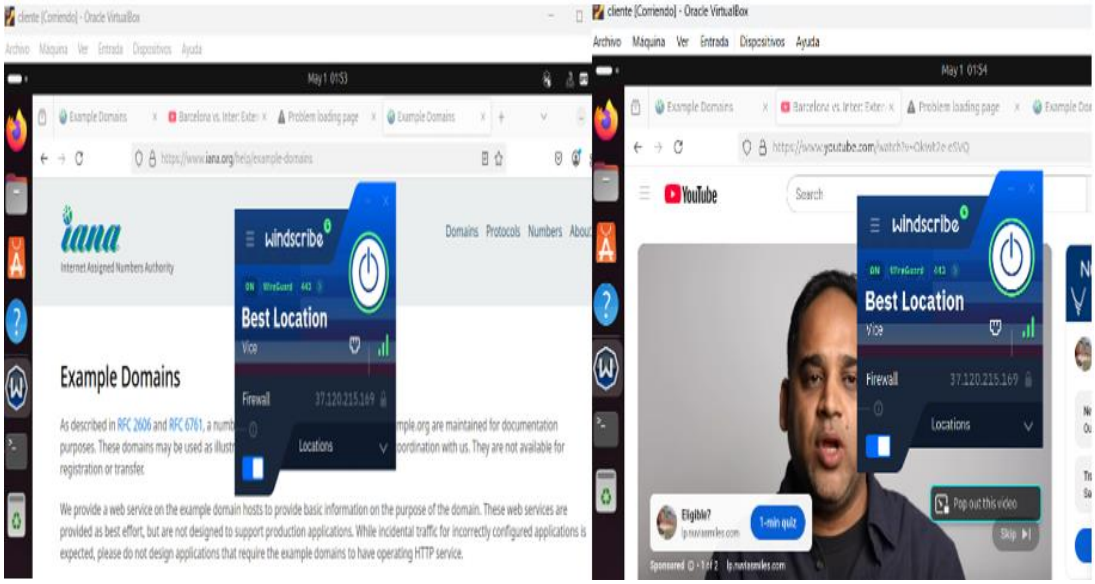

Ilustración 4-44 Generación de tráfico HTTP(S) desde el cliente con Windscribe activo y Reproducción de contenido en YouTube durante el ataque MITM

Prueba de fuga de DNS en Windscribe

Una de las vulnerabilidades comunes en implementaciones VPN mal configuradas es la fuga DNS. Para verificar si Windscribe presentaba este comportamiento, se captura tráfico en la interfaz eth0 del atacante con tcpdump mientras el cliente tenía la VPN activa. Como se muestra en la ilustración 4-45, el objetivo fue identificar si las consultas DNS salían del túnel cifrado.

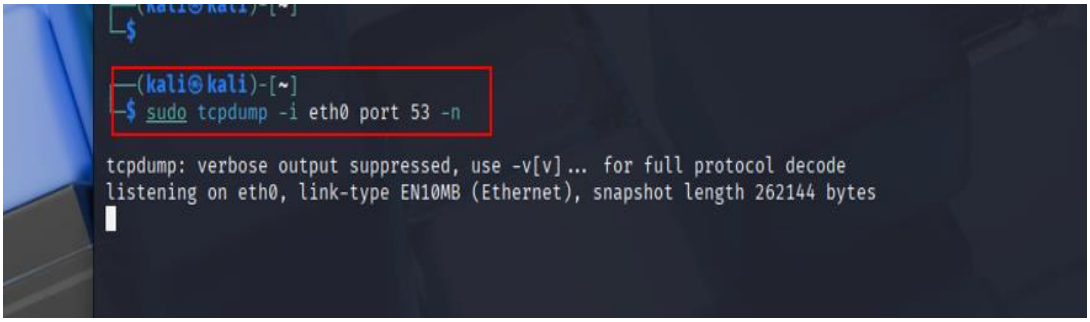

Ilustración 4-45 Inicio de captura de fuga de dns con tcpdump

Posteriormente, desde el cliente Ubuntu se ejecuta el comando dig para forzar una consulta de resolución de nombre. Esta acción genera tráfico DNS que permite comprobar si el resolutor era local o externo. La ilustración 4-46 muestra la emisión de la consulta desde el cliente.

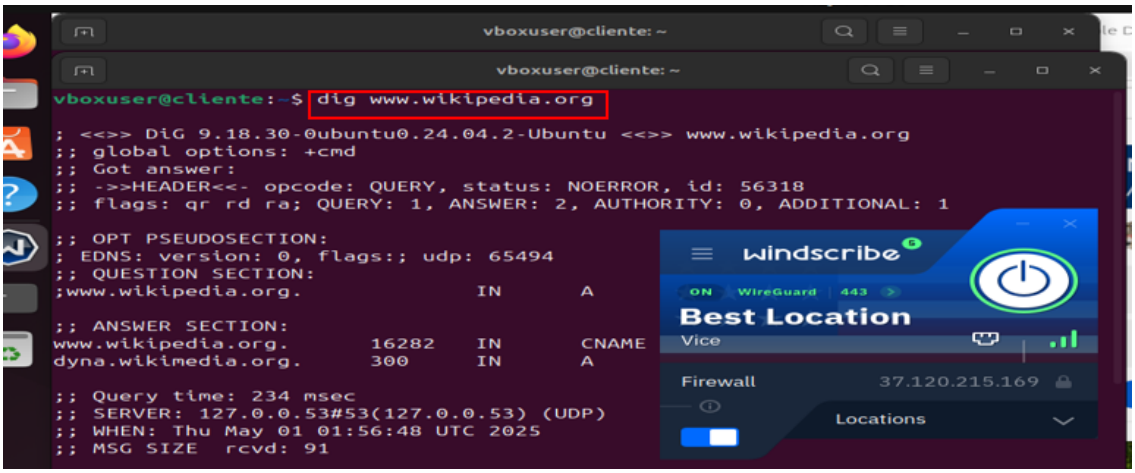

Ilustración 4-46 Generación de tráfico DNS con comando dig

Simulación de entorno IoT y evaluación con tráfico MQTT

Finalmente, se realiza una prueba en un entorno simulado de IoT, donde el cliente Ubuntu actua como un sensor de temperatura enviando datos mediante MQTT a un broker local como indica la ilustración 4-47.

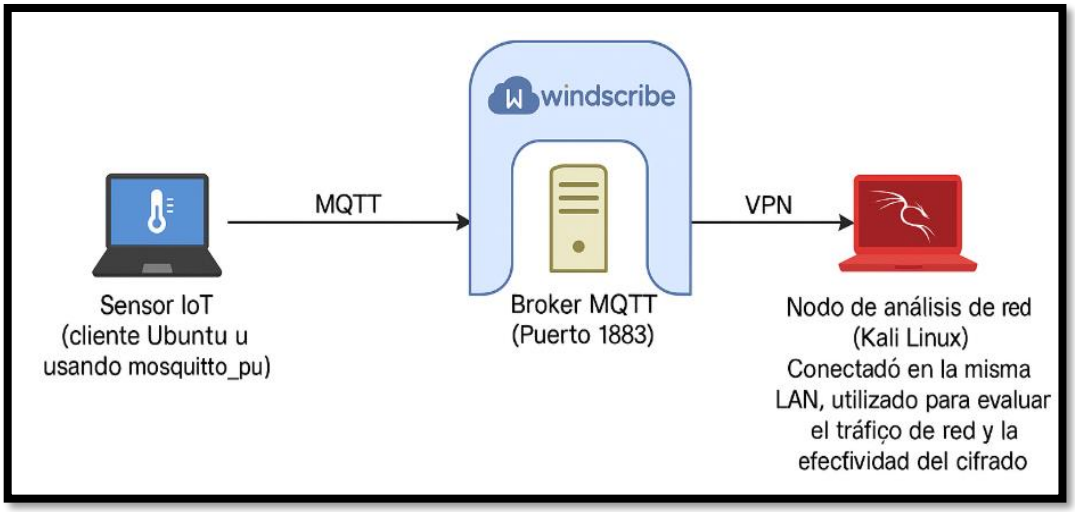

Ilustración 4-47 Diagrama de la topología usada en el ataque MITM con Windscribe

Esta simulación utiliza el túnel de Windscribe previamente establecido mediante la aplicación de escritorio oficial, como se observa en la ilustración 4-48.

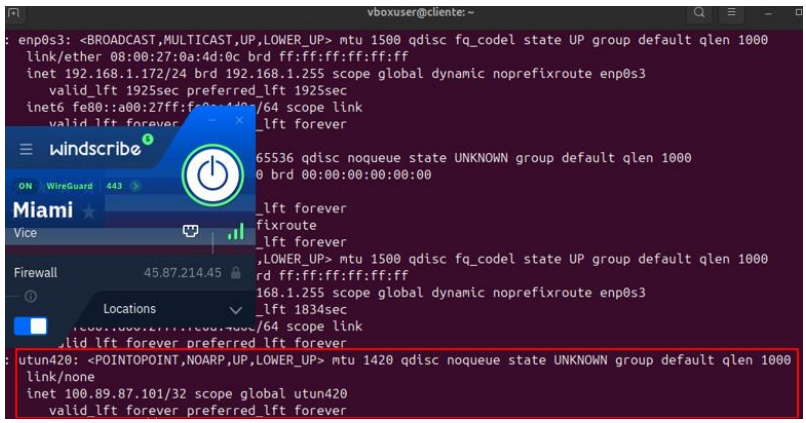

Ilustración 4-48 iniciación de la aplicación de Windscribe

La VPN asigna al cliente una IP pública virtual (100.89.87.101/32), visible en la interfaz utun420. Se verifica la operación del túnel mediante el comando ip a, y luego se generaron paquetes MQTT hacia el broker, como se muestra en la ilustración 4-49.

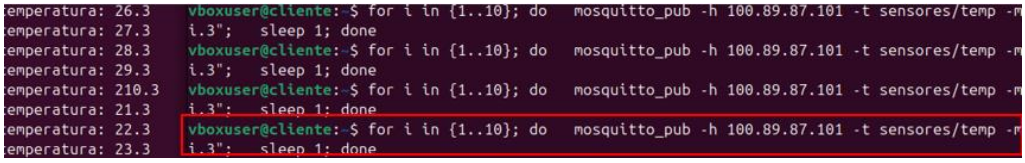

Ilustración 4-49 Generación de paquetes MQTT en el túnel de Windscribe

Para simular el ataque MITM, se ejecuta Bettercap desde Kali Linux, apuntando a la IP LAN del cliente (192.168.1.172) mediante los módulos arp.spoof y net.sniff, como se ilustra en la figura 4-50.

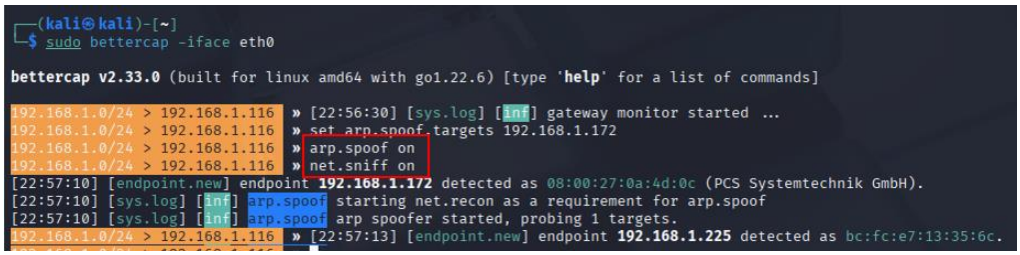

Ilustración 4-50 Inicialización de bettercap y comandos de arp.spoof y net.sniff activados

Durante esta prueba se evalua si era posible visualizar o interceptar los mensajes publicados mediante MQTT a través del túnel VPN. Las observaciones registradas fueron documentadas para su análisis, el cual se presenta detalladamente en el capítulo 5.

4.3.4 Análisis de vulnerabilidades y riesgos en NordVPN

El análisis de seguridad de NordVPN se enfoca en evaluar su capacidad de protección frente a ataques del tipo *Man-in-the-Middle* (MITM) y fugas de tráfico DNS, tanto en escenarios de navegación convencional como en entornos IoT simulados. La VPN fue evaluada utilizando su implementación basada en el protocolo NordLynx, derivado de WireGuard, mediante pruebas controladas en una red local. Se utiliza las herramientas tcpdump para la captura de tráfico, Bettercap para la simulación de ataques activos, y comandos específicos para generar tráfico desde el cliente Ubuntu. Los resultados completos se presentan en el capítulo 5.

Verificación del cifrado del tráfico

Para verificar el cifrado del tráfico durante la conexión mediante NordVPN, se emplea tcpdump en la máquina atacante (Kali Linux) sobre la interfaz física eth0, con filtros dirigidos a la IP del cliente (192.168.1.172). Esta acción busca identificar si era posible interceptar paquetes mientras la VPN se encontraba activa. La ejecución inicial se muestra en la ilustración 4-51.

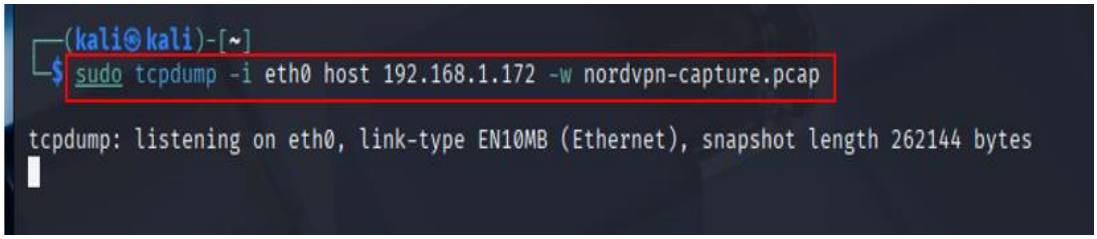

Ilustración 4-51 Inicialización de la captura de tráfico en el atacante con tcpdump filtrando al cliente

En paralelo, desde el cliente Ubuntu se accedió al sitio web de Wikipedia utilizando el comando curl, lo cual genera tráfico HTTP saliente a través del túnel VPN. Como se muestra en la ilustración 4-52, se verifica visualmente que la conexión permanecía activa bajo el protocolo NordLynx, utilizando la interfaz virtual nordlynx.

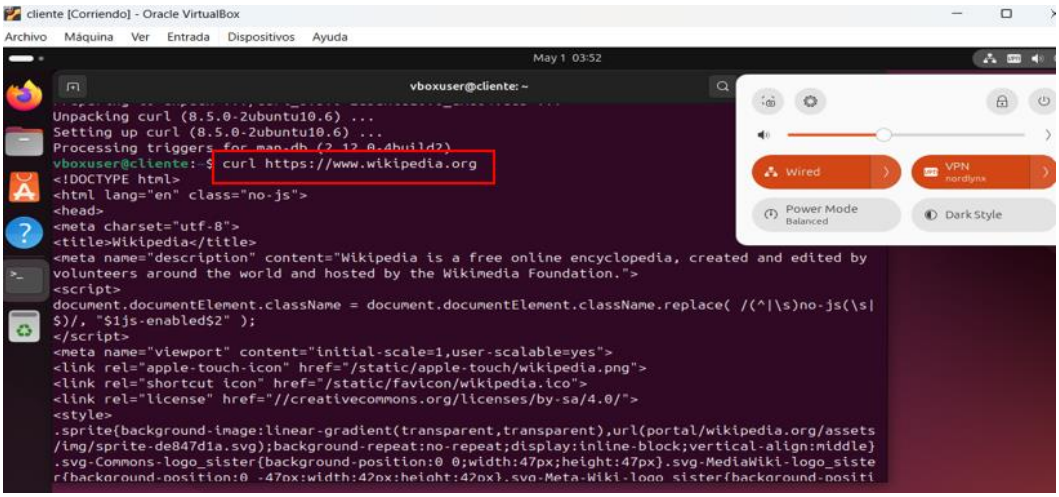

Ilustración 4-52 Conexión activa a Wikipedia desde el cliente con NordVPN habilitado

Simulación de ataque MITM con tráfico convencional

Una vez confirmada la conexión mediante NordVPN, se ejecuta una simulación de ataque MITM desde Kali Linux utilizando Bettercap. La herramienta se activa sobre la interfaz eth0, configurando la IP del cliente como objetivo de suplantación ARP (arp.spoof) y habilitando la captura de paquetes (net.sniff). Este procedimiento se muestra en la ilustración 4-53.

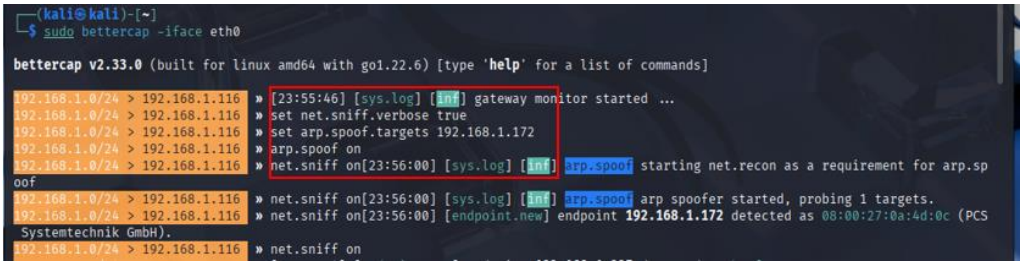

Ilustración 4-53 Inicio del ataque MITM mediante Bettercap hacia el cliente conectado a NordVPN

Durante el ataque, el cliente vuelve a realizar peticiones a wikipedia.org mediante curl, generando tráfico cifrado que circuló a través del túnel VPN. Como se muestra en la ilustración 4-54 muestra la generación de tráfico en este contexto.

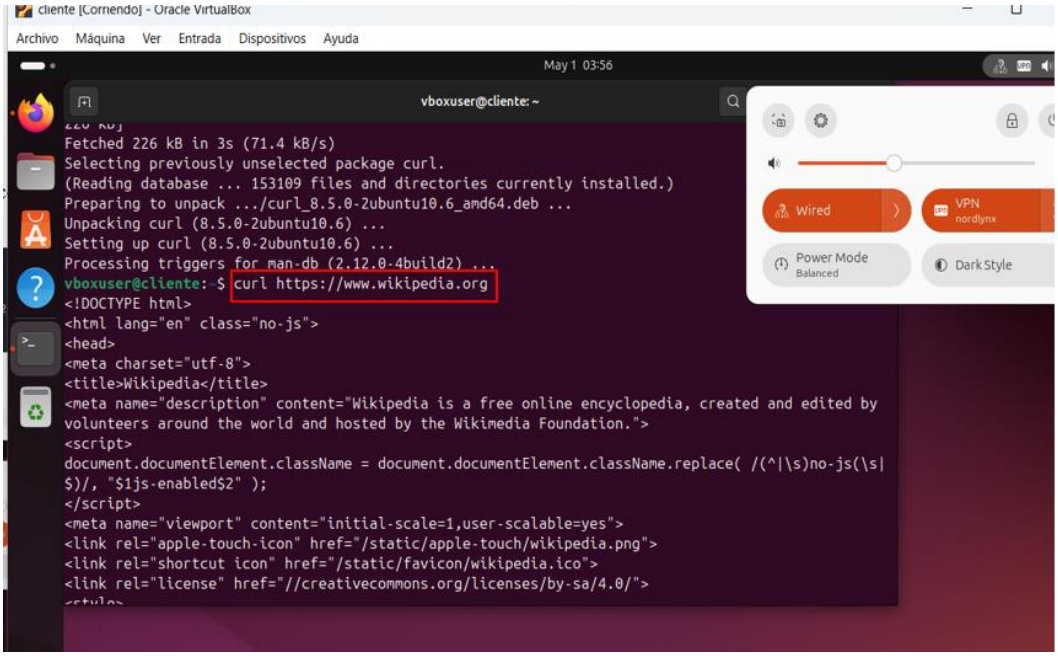

Ilustración 4-54 Generación de tráfico web desde el cliente con NordVPN activo

Prueba de fuga de DNS

Para analizar la existencia de fugas DNS en la conexión mediante NordVPN, se ejecuta una consulta con el comando dig. desde el cliente mientras la VPN estaba activa. En paralelo, el atacante en Kali Linux utiliza tcpdump para escuchar tráfico saliente por el puerto 53 en la interfaz eth0, como se muestra en la ilustración 4-55.

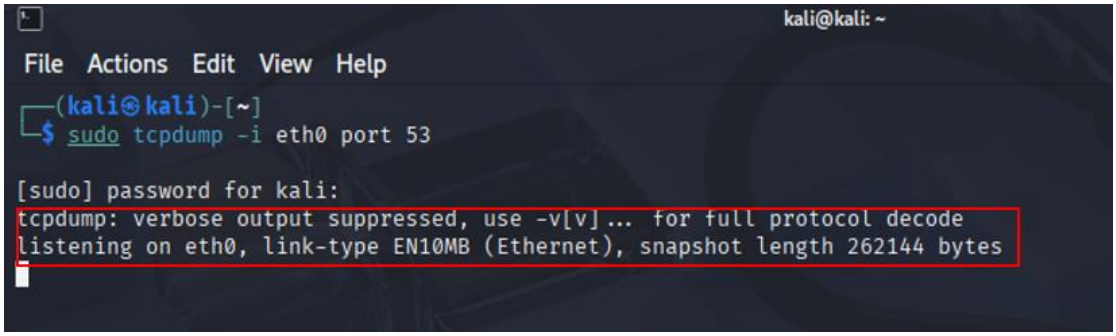

Ilustración 4-55 Escucha de paquetes DNS en el puerto 53 desde Kali Linux mediante tcpdump

La consulta dig www.wikipedia.org fue procesada por el cliente, utilizando el servidor DNS local 127.0.0.53, sin evidenciar solicitudes externas. La ilustración 4-56 muestra la salida obtenida en el cliente Ubuntu.

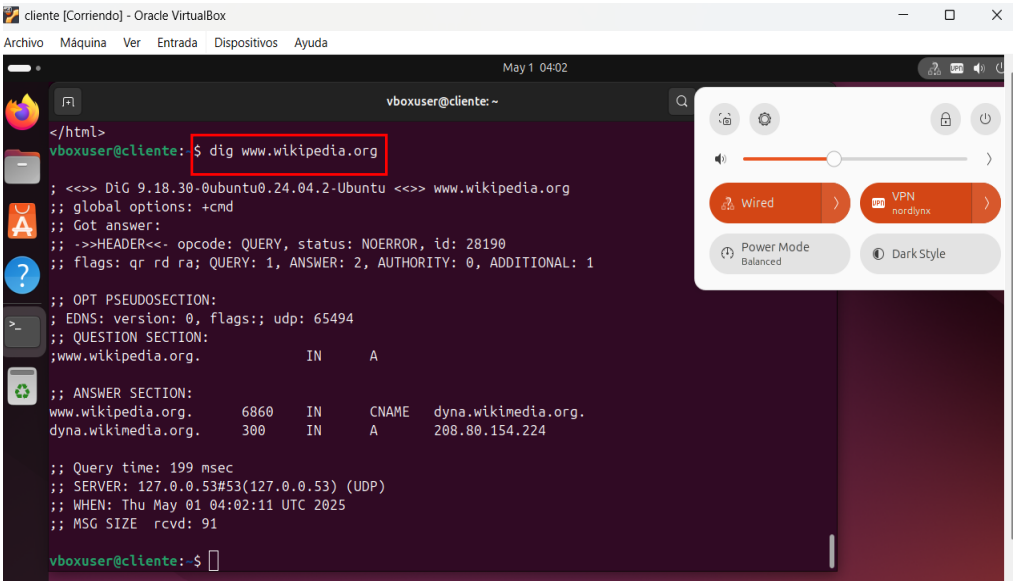

Ilustración 4-56 Resultado de la consulta DNS desde el cliente. La resolución fue gestionada internamente a través de 127.0.0.53

Simulación de entorno IoT y evaluación con tráfico MQTT

Finalmente, se simula un entorno IoT donde el cliente Ubuntu actúa como sensor de temperatura, enviando datos mediante MQTT a un broker local a través del túnel protegido por NordVPN como se muestra en la ilustración 4-57.

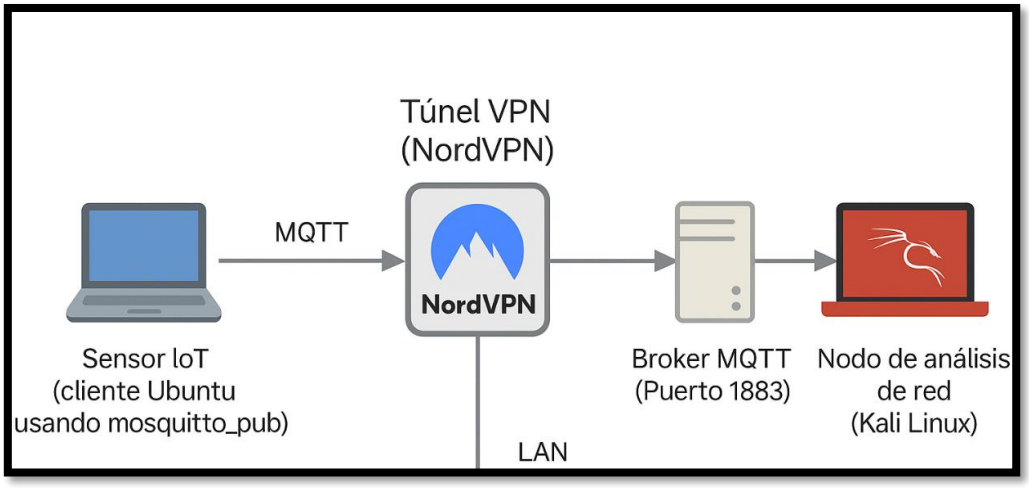

Ilustración 4-57 Diagrama de la topología usada en el ataque MITM con NordVPN

La conexión fue establecida mediante la aplicación oficial, generando automáticamente la interfaz virtual nordlynx. La ilustración 4-58 muestra la activación de la VPN.

```
vboxuser@cliente:~$ nordvpn connect us
Connecting to United States #11510 (us11510.nordvpn.com)
You are connected to United States #11510 (us11510.nordvpn.com)!
vboxuser@cliente:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group 57841 qlen 1
000
    link/ether 08:00:27:0a:4d:0c brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.172/24 brd 192.168.1.255 scope global dynamic noprefixroute enp0s3
       valid_lft 7052sec preferred_lft 7052sec
7: nordlynx: <POINTOPOINT,UP,LOWER_UP> mtu 1420 qdisc noqueue state UNKNOWN group default qlen 1
000
    link/none
    inet 10.5.0.2/32 scope global nordlynx
       valid_lft forever preferred_lft forever
```

Ilustración 4-58 iniciación de la aplicación de NordVPN

La IP asignada al cliente fue 10.5.0.2/32, utilizada como dirección de salida en el túnel cifrado. Se verifica la conectividad mediante el comando ip a, y se generaron paquetes MQTT simulando la actividad de un sensor. Como se muestra en la ilustración 4-59, el tráfico fue dirigido hacia el broker local bajo el canal protegido.

```
temperatura: 26.3    vboxuser@cliente:~$ for i in {1..10}; do   mosquitto_pub -h 100.89.87.101 -t sensores/temp -m
temperatura: 27.3    i.3";   sleep 1; done
temperatura: 28.3    vboxuser@cliente:~$ for i in {1..10}; do   mosquitto_pub -h 100.89.87.101 -t sensores/temp -m
temperatura: 29.3    i.3";   sleep 1; done
temperatura: 210.3   vboxuser@cliente:~$ for i in {1..10}; do   mosquitto_pub -h 100.89.87.101 -t sensores/temp -m
temperatura: 21.3    i.3";   sleep 1; done
temperatura: 22.3    vboxuser@cliente:~$ for i in {1..10}; do   mosquitto_pub -h 100.89.87.101 -t sensores/temp -m
temperatura: 23.3    i.3";   sleep 1; done
```

Ilustración 4-59 Generación de paquetes MQTT en el túnel de NordVPN

En paralelo, se prepara la máquina Kali Linux como nodo de evaluación, ejecutando Bettercap sobre la interfaz eth0 y configurando como objetivo la IP local del cliente.

Los módulos arp.spoof y net.sniff fueron activados para capturar posibles transmisiones visibles, como se detalla en la ilustración 4-60.

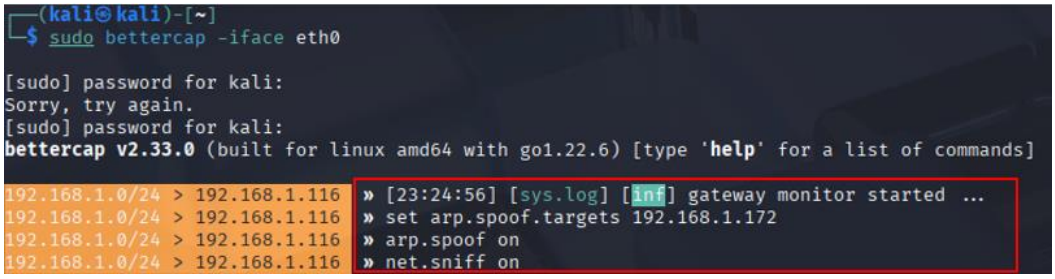

Ilustración 4-60 Inicialización de bettercap y comandos de arp.spoof y net.sniff activados

Esta prueba tuvo como propósito verificar si NordVPN, mediante NordLynx, es capaz de encapsular completamente el tráfico MQTT y evitar su exposición ante un atacante local. Los resultados de esta evaluación se analizan en detalle en el capítulo 5.

5 Resultados y discusión

Este capítulo presenta y analiza los resultados obtenidos a partir de las pruebas realizadas sobre las distintas tecnologías de redes privadas virtuales (VPN) evaluadas: WireGuard, OpenVPN, Windscribe y NordVPN. El análisis se orienta a contrastar el comportamiento de cada solución bajo criterios de rendimiento, seguridad en redes convencionales y protección del tráfico en entornos del Internet de las Cosas (IoT).

En primer lugar, se exponen los resultados cuantitativos derivados de las pruebas de rendimiento, los cuales incluyen métricas como latencia, velocidad de transferencia y uso de CPU. Estos parámetros permiten valorar la eficiencia operativa de cada tecnología, aspecto esencial cuando se considera su aplicación en dispositivos IoT con recursos computacionales limitados.

A continuación, se analiza la robustez de los mecanismos de cifrado y autenticación implementados por cada VPN, tomando como referencia tanto la documentación técnica oficial como las observaciones realizadas durante la ejecución de los escenarios de prueba. En esta etapa se identifican potenciales vulnerabilidades y se evalúa la resistencia de cada tecnología frente a amenazas como ataques de tipo

Man-in-the-Middle (MITM) y fugas de solicitudes DNS, dos de los riesgos más relevantes en redes públicas y compartidas.

Finalmente, se examina la capacidad de las VPN para proteger el tráfico generado por dispositivos IoT mediante el protocolo MQTT. Para ello, se diseñó un entorno experimental en el que un cliente VPN emula el comportamiento de un sensor que transmite datos hacia un broker. Las pruebas se centraron en determinar si el cifrado aplicado por cada túnel impedía efectivamente la interceptación del tráfico MQTT en presencia de un atacante local. Esta última sección constituye un aporte práctico relevante para validar la viabilidad de estas soluciones en escenarios reales de IoT.

5.1 Resultados de las Pruebas de Rendimiento de las diferentes VPNS

En este apartado se presentan los resultados obtenidos durante las pruebas de rendimiento aplicadas a las tecnologías VPN evaluadas. Las métricas analizadas incluyen latencia, velocidad de transferencia de datos y carga de CPU, aspectos críticos para valorar la eficiencia de estas soluciones en entornos con recursos limitados, como es el caso de los dispositivos del Internet de las Cosas (IoT).

5.1.1 Resultados de rendimiento de Wireguard

Latencia

Para medir la latencia entre el cliente y el servidor utilizando WireGuard, se realizaron pruebas mediante los comandos ping y mtr, enviando múltiples paquetes ICMP entre ambos extremos a través del túnel VPN. Como se muestra en la ilustración 5-1, los resultados obtenidos con ping reflejan un tiempo de respuesta promedio muy bajo, estable en todas las mediciones.

```
osboxes@osboxes:~$ ping -c4 10.10.20.1
PING 10.10.20.1 (10.10.20.1) 56(84) bytes of data.
64 bytes from 10.10.20.1: icmp_seq=1 ttl=64 time=2.88 ms
64 bytes from 10.10.20.1: icmp_seq=2 ttl=64 time=0.956 ms
64 bytes from 10.10.20.1: icmp_seq=3 ttl=64 time=0.984 ms
64 bytes from 10.10.20.1: icmp_seq=4 ttl=64 time=1.04 ms

--- 10.10.20.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 0.956/1.465/2.883/0.819 ms
```

Ilustración 5-1 Resultados del comando ping entre cliente y servidor

El comando mtr, configurado con el envío de 100 paquetes, confirma esta estabilidad, sin registrar pérdidas ni fluctuaciones significativas en el tiempo de ida y vuelta, como se evidencia en la ilustración 5-2.

```
osboxes@osboxes:~$ mtr -r -c 100 10.10.20.1
Start: 2025-02-18T15:03:58-0500
HOST: osboxes                     Loss%   Snt   Last   Avg  Best  Wrst StDev
  1.|-- 10.10.20.1                0.0%   100    0.9   1.0   0.7   1.4   0.1
```

Ilustración 5-2 Resultados del comando mtr (100 paquetes enviados)

En la ilustración 5-3 se indican los valores de latencia obtenidos a través de comandos ping emitidos en intervalos regulares durante 10 segundos. Los resultados muestran una latencia promedio inferior a los 2.5 ms, con valores mínimos por debajo de 1 ms y sin pérdidas de paquetes. Esta respuesta inmediata es particularmente ventajosa en sistemas distribuidos de control o monitoreo en tiempo real, característicos de aplicaciones IoT.

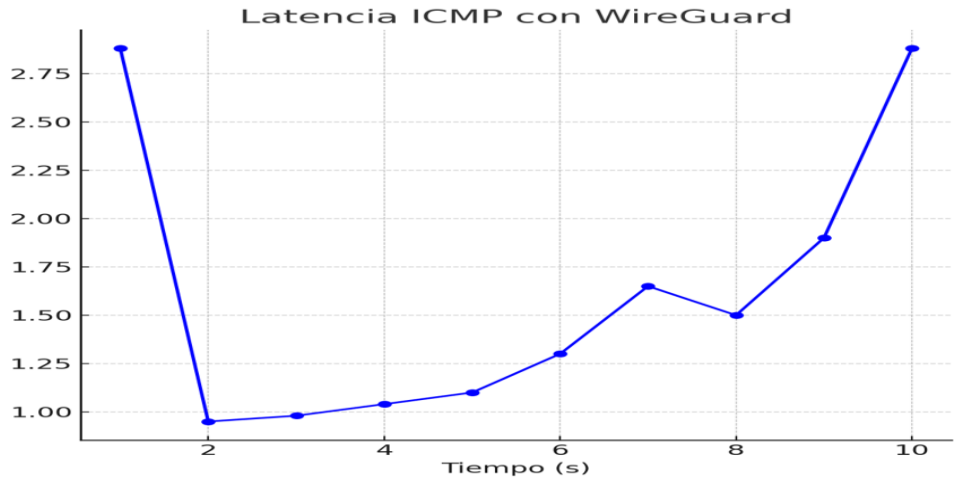

Ilustración 5-3 Latencia ICMP en túnel WireGuard durante la conexión

Velocidad de Transferencia

La velocidad de transmisión de datos a través del túnel WireGuard se evalúa mediante iperf3, herramienta utilizada para medir el throughput entre el cliente y el servidor. Como se observa en la ilustración 5-4, WireGuard alcanzó velocidades superiores a 150 Mbps en la mayoría de las ejecuciones.

```
osboxes@osboxes:~$ iperf3 -c 10.10.20.1
Connecting to host 10.10.20.1, port 5201
[  5] local 10.10.20.2 port 34252 connected to 10.10.20.1 port 5201
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[  5]   0.00-1.00   sec  18.5 MBytes   155 Mbits/sec   38   66.8 KBytes
[  5]   1.00-2.00   sec  18.9 MBytes   158 Mbits/sec   12    106 KBytes
[  5]   2.00-3.00   sec  19.2 MBytes   162 Mbits/sec   43   82.8 KBytes
[  5]   3.00-4.00   sec  17.9 MBytes   150 Mbits/sec   21    114 KBytes
[  5]   4.00-5.00   sec  18.4 MBytes   154 Mbits/sec   17   96.2 KBytes
[  5]   5.00-6.00   sec  18.9 MBytes   158 Mbits/sec   29   98.9 KBytes
[  5]   6.00-7.00   sec  18.4 MBytes   154 Mbits/sec   15    100 KBytes
[  5]   7.00-8.00   sec  17.4 MBytes   146 Mbits/sec   46   96.2 KBytes
[  5]   8.00-9.00   sec  18.0 MBytes   151 Mbits/sec   47   86.8 KBytes
[  5]   9.00-10.00  sec  18.9 MBytes   158 Mbits/sec   19   82.8 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Retr
[  5]   0.00-10.00  sec   184 MBytes   155 Mbits/sec  287             sender
[  5]   0.00-10.00  sec   184 MBytes   154 Mbits/sec                  receiver
```

Ilustración 5-4 Resultados de iperf3

La baja tasa de retransmisión y la consistencia del rendimiento sugieren una conexión eficiente, adecuada para aplicaciones donde se requiere transmisión estable y con mínima latencia, como sucede en sistemas IoT distribuidos.

En la ilustración 5-5 presenta la variación del ancho de banda medido con iperf3 durante un intervalo de 10 segundos en una conexión protegida por WireGuard. Se observa una velocidad constante cercana a los 155 Mbps, con ligeras oscilaciones propias de la dinámica del canal. Estos valores reflejan un rendimiento sostenido y eficiente, ideal para entornos IoT donde la estabilidad en la transmisión es un requisito esencial.

Ilustración 5-5 Velocidad de transferencia con WireGuard durante la conexión

Carga en la CPU

El uso de CPU se mide durante las pruebas de transferencia, con el fin de observar el impacto del cifrado sobre el procesamiento en el cliente. En la ilustración 5-6 se presenta el resultado de carga, donde se evidencia un consumo inferior al 0.1 % en promedio.

Ilustración 5-6 Resultado de Carga en la CPU

Esta baja utilización de recursos valida la eficiencia de la arquitectura de WireGuard, haciéndola especialmente compatible con nodos IoT de bajo consumo energético o con capacidad de procesamiento limitada.

La ilustración 5-6 muestra la evolución del consumo de CPU durante un intervalo de 10 segundos en la transmisión de datos cifrados con WireGuard. Se observa una utilización mínima, que se mantuvo en valores inferiores al 0.2 %, validando así la eficiencia del protocolo incluso en procesos prolongados. Esta baja carga confirma su idoneidad para dispositivos IoT con limitaciones de procesamiento.

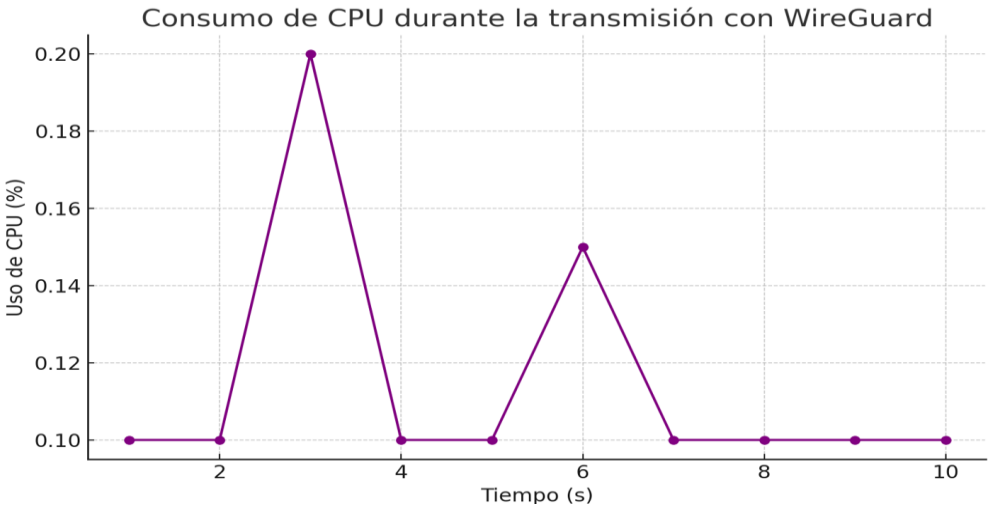

Ilustración 5-6 Consumo de CPU durante la transmisión con WireGuard

Tabla estructurada

En la tabla 2 se presentan los resultados obtenidos de todas las pruebas realizadas usando Wireguard.

Tabla 2 Resultados de Wireguard

	Métrica	Valor
1	Latencia promedio (ms)	1.465
2	Perdida de paquetes (%)	0
3	Velocidad promedio (Mbps)	155
4	Uso de CPU (%)	0.1

5.1.2 Resultados de Pruebas de Rendimiento en OpenVPN

Latencia

Para medir la latencia entre el cliente y el servidor a través de OpenVPN, se lleva a cabo pruebas utilizando el comando ping. Como se muestra en la ilustración 5-7, se registraron tiempos de respuesta bajos y estables durante la comunicación directa entre ambos nodos conectados por el túnel VPN.

```
osboxes@osboxes:~$ ping -c 4 10.8.0.1
PING 10.8.0.1 (10.8.0.1) 56(84) bytes of data.
64 bytes from 10.8.0.1: icmp_seq=1 ttl=64 time=1.25 ms
64 bytes from 10.8.0.1: icmp_seq=2 ttl=64 time=1.26 ms
64 bytes from 10.8.0.1: icmp_seq=3 ttl=64 time=0.962 ms
64 bytes from 10.8.0.1: icmp_seq=4 ttl=64 time=0.814 ms

--- 10.8.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 0.814/1.069/1.255/0.188 ms
```

Ilustración 5-7 Resultados del comando ping entre cliente y servidor

Además, se realizaron pruebas adicionales de latencia hacia un nodo externo de la red con el fin de observar el comportamiento de OpenVPN bajo condiciones ampliadas. Como se evidencia en la ilustración 5-8, los resultados se mantuvieron dentro de un margen aceptable, sin presentar pérdidas de paquetes ni fluctuaciones significativas.

```
osboxes@osboxes:~$ ping -c 4 10.10.10.254
PING 10.10.10.254 (10.10.10.254) 56(84) bytes of data.
64 bytes from 10.10.10.254: icmp_seq=1 ttl=64 time=0.865 ms
64 bytes from 10.10.10.254: icmp_seq=2 ttl=64 time=0.769 ms
64 bytes from 10.10.10.254: icmp_seq=3 ttl=64 time=0.744 ms
64 bytes from 10.10.10.254: icmp_seq=4 ttl=64 time=0.731 ms

--- 10.10.10.254 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3055ms
rtt min/avg/max/mdev = 0.731/0.777/0.865/0.052 ms
```

Ilustración 5-8 Resultados del comando ping a otro nodo de la red

En la ilustracion 5-9 se presenta la evolución de la latencia registrada mediante comandos ping emitidos de forma continua durante 10 segundos. La gráfica indica valores promedio alrededor de 1 ms, con leves variaciones. Este comportamiento es aceptable para aplicaciones IoT que toleran pequeñas fluctuaciones en el retardo, siempre que no comprometan la sincronización crítica del sistema.

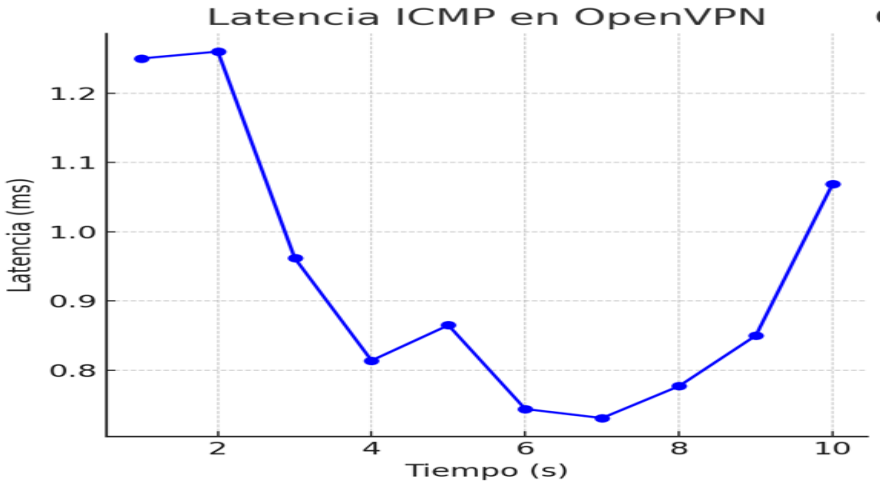

Ilustración 5-9 Latencia ICMP en OpenVPN durante la conexión

Velocidad de Transferencia

La evaluación del rendimiento en cuanto a velocidad de transmisión se realiza mediante la herramienta iperf3, midiendo el throughput entre el cliente IoT y el servidor bajo conexión OpenVPN. Como se muestra en la ilustración 5-10, la velocidad promedio fue inferior a la observada con WireGuard, y se detectaron múltiples retransmisiones durante la sesión de prueba.

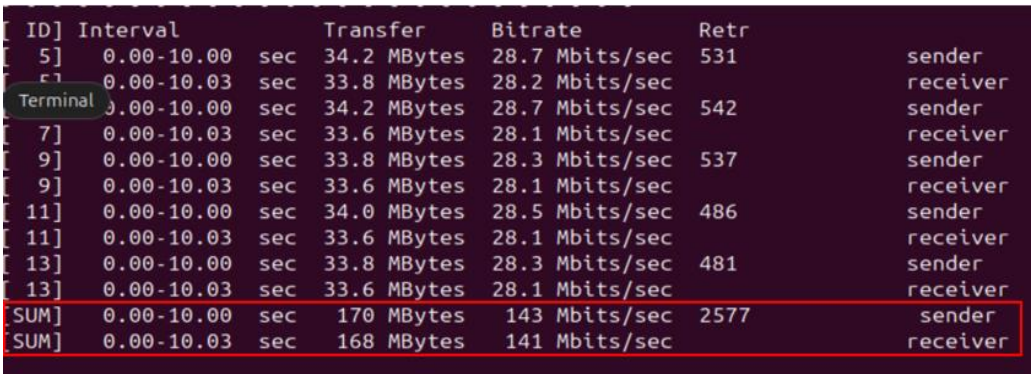

Ilustración 5-10 Resultados de iperf3

Estos resultados indican que, aunque OpenVPN mantiene una conexión funcional, la eficiencia de transferencia puede verse afectada por la sobrecarga computacional que implica su sistema de cifrado, especialmente en escenarios con tráfico sostenido.

En la ilustracion 5-11 muestra la variación del ancho de banda medido con iperf3 en una sesión de 10 segundos protegida por OpenVPN. Aunque el rendimiento se mantiene relativamente constante entre 139 y 144 Mbps, se evidencian ligeras oscilaciones que podrían estar asociadas al mayor consumo de recursos de cifrado de OpenVPN, en comparación con otras tecnologías más ligeras como WireGuard.

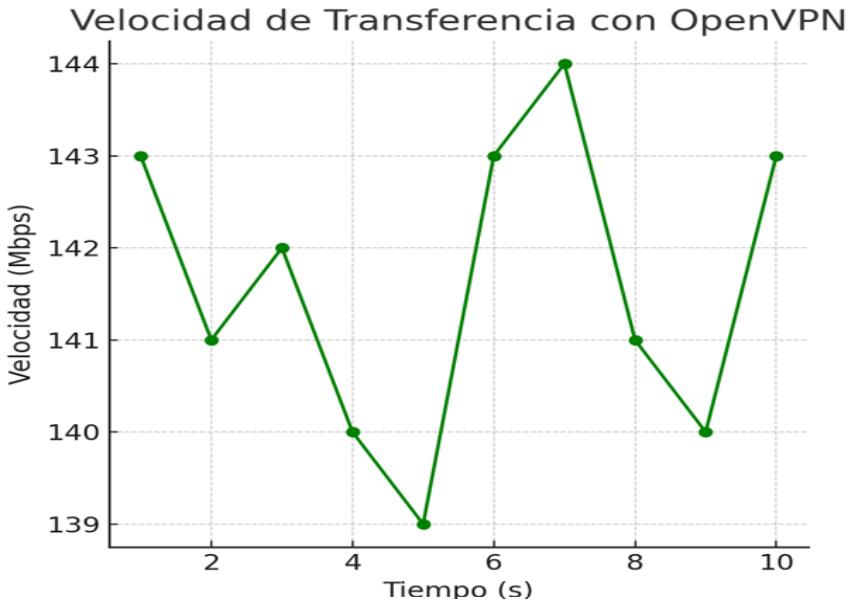

Ilustración 5-10 Velocidad de transferencia con OpenVPN

Carga en la CPU

El monitoreo del uso de CPU durante las pruebas permite observar el impacto del cifrado sobre los recursos del sistema. Como se muestra en la ilustración 5-11, OpenVPN presentó un consumo significativamente más alto en comparación con WireGuard, alcanzando valores cercanos al 0.7 % de uso constante.

```
op - 02:00:19 up  7:30,  1 user,  load average: 0.41, 0.15, 0.07
asks: 209 total,   1 running, 208 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.7 us,  1.0 sy,  0.0 ni, 98.1 id,  0.2 wa,  0.0 hi,  0.0 si,  0.0 st
iB Mem :   3847.6 total,    212.1 free,   2250.6 used,   1703.8 buff/cache
iB Swap:   4096.0 total,   4095.5 free,      0.5 used.   1597.0 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 2900 osboxes   20   0 4068272 481968 169456 S   3.0  12.2  17:20.75 gnome-s+
 5570 osboxes   20   0   11.2g 458008 190948 S   1.0  11.6   8:21.10 firefox
12708 osboxes   20   0  565084  63512  46212 S   0.7   1.6   0:06.41 gnome-t+
 6334 osboxes   20   0 2577684 158064 101252 S   0.3   4.0   3:09.61 Isolate+
14617 root      20   0       0      0      0 I   0.3   0.0   0:00.11 kworker+
14659 osboxes   20   0   14536   5948   3772 R   0.3   0.2   0:00.03 top
    1 root      20   0   23404  14180   9316 S   0.0   0.4   0:03.78 systemd
```

Ilustración 5-11 Resultados del monitoreo de CPU

Este mayor requerimiento computacional puede representar una limitación para su implementación en dispositivos IoT de bajo costo o con capacidades restringidas, donde la optimización del uso energético y de procesamiento es prioritaria.

En la ilustración 5-12 se indica el uso del procesador durante la sesión de prueba con OpenVPN. A lo largo de los 10 segundos analizados, se mantiene un consumo

sostenido entre 0.6 y 0.8 %, cifra superior a la observada con otras tecnologías. Este nivel de carga puede ser un factor limitante en dispositivos IoT que operan con microcontroladores o unidades de procesamiento de baja potencia.

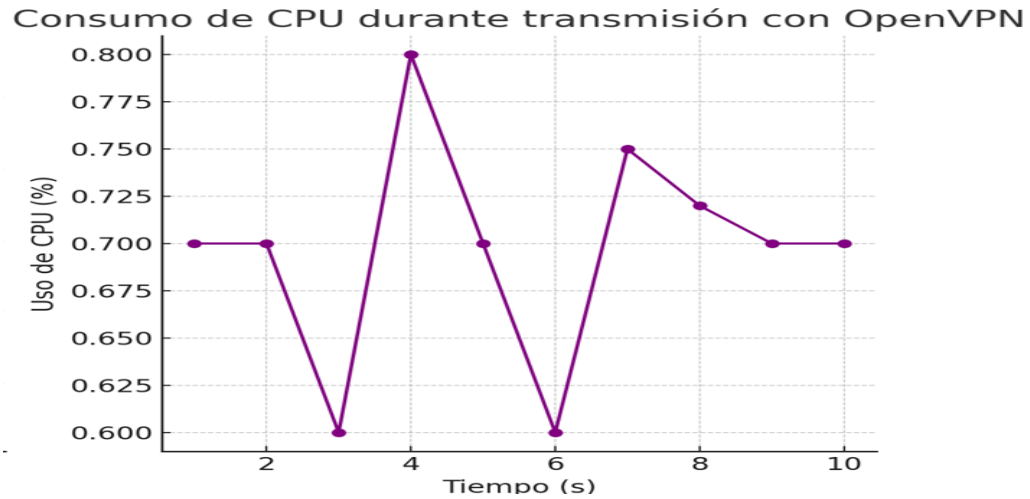

Ilustración 5-12 Consumo de CPU durante la transmisión con OpenVPN

Tabla estructurada

En la tabla 3 se presentan los resultados obtenidos de todas las pruebas realizadas usando OpenVPN.

Tabla 3 Resultados de rendimiento de OpenVPN

	Métrica	Valor
1	Latencia promedio (ms)	1.069
2	Perdida de paquetes (%)	0
3	Velocidad promedio (Mbps)	141.0
4	Uso de CPU (%)	0.7

5.1.3 Resultados de Pruebas de Rendimiento en Windscribe

Latencia

Para medir la latencia entre el cliente y el servidor utilizando Windscribe, se realizan pruebas mediante el comando ping hacia destinos representativos como google.com y el servidor DNS público 8.8.8.8. Como se muestra en la ilustración 5-13, los resultados obtenidos reflejan tiempos de respuesta relativamente altos, lo

cual puede influir negativamente en aplicaciones que dependen de comunicaciones en tiempo real.

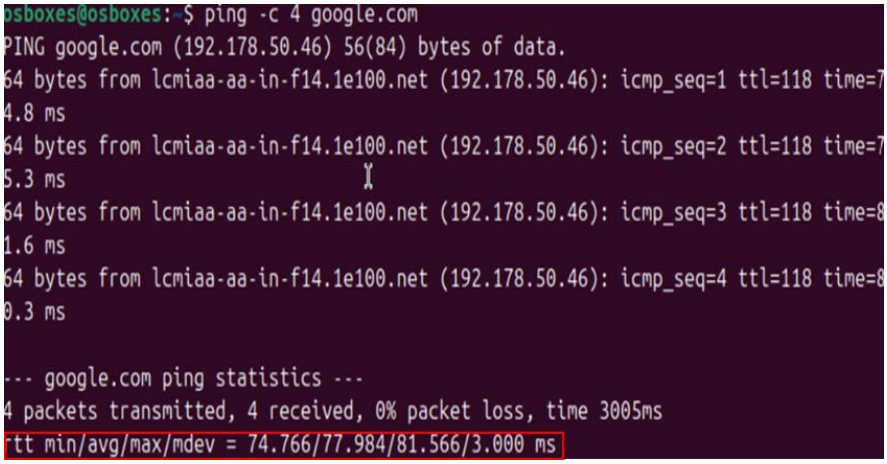

Ilustración 5-13 Resultados del comando ping a google.com

Adicionalmente, se aplica una prueba directa hacia 8.8.8.8, obteniendo resultados similares, tal como se observa en la ilustración 5-14. Las respuestas mostraron una latencia constante superior a la registrada en pruebas previas con WireGuard y OpenVPN.

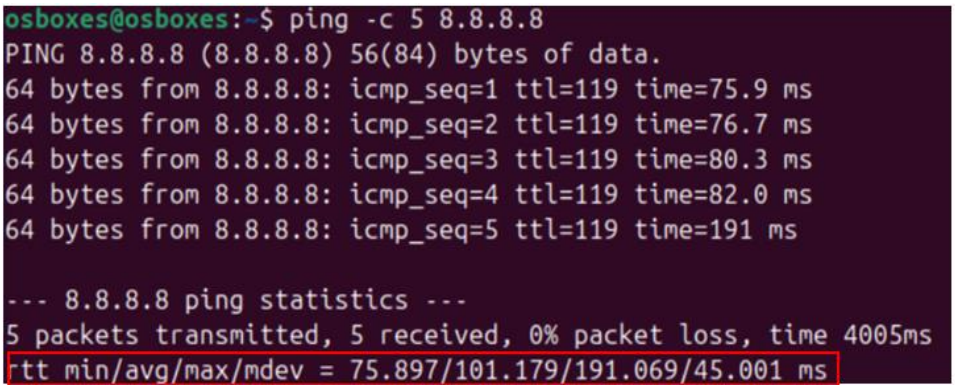

Ilustración 5-14 Resultados del comando ping a 8.8.8.8 (Google DNS)

La herramienta mtr fue utilizada con el envío de 10 paquetes hacia 8.8.8.8, revelando latencia acumulada y pérdida ocasional de paquetes en nodos intermedios. Este comportamiento se visualiza en la ilustración 5-15.

```
osboxes@osboxes:~$ mtr -rw 8.8.8.8
Start: 2025-02-19T15:56:28-0500
HOST: osboxes          Loss%   Snt   Last   Avg  Best  Wrst StDev
  1.|-- 10.5.0.1         0.0%    10   66.3  66.4  66.2  66.9   0.2
  2.|-- 69.67.150.2      0.0%    10   66.8  66.8  66.3  68.7   0.7
  3.|-- 10.10.209.177    0.0%    10   66.4  66.5  66.2  67.0   0.3
  4.|-- 10.10.0.18       0.0%    10   66.1  68.1  66.1  80.7   4.5
  5.|-- 206.41.108.12    0.0%    10   66.6  66.8  66.5  68.0   0.4
  6.|-- 192.178.99.245   0.0%    10   67.5  67.8  67.2  69.5   0.7
  7.|-- 216.239.62.1     0.0%    10   67.5  67.4  67.0  68.4   0.4
  8.|-- dns.google       0.0%    10   67.0  66.6  66.4  67.0   0.2
```

Ilustración 5-15 Resultados del comando mtr (10 paquetes enviados a 8.8.8.8)

La ilustración 5-16 muestra el comportamiento de la latencia durante 10 segundos de prueba, utilizando ping hacia servidores externos. La latencia promedio supera los 100 ms, con fluctuaciones visibles a lo largo del tiempo. Este retardo elevado, si bien es tolerable en aplicaciones generales, podría afectar negativamente entornos IoT sensibles al tiempo de respuesta, especialmente en sistemas de control o notificación inmediata.

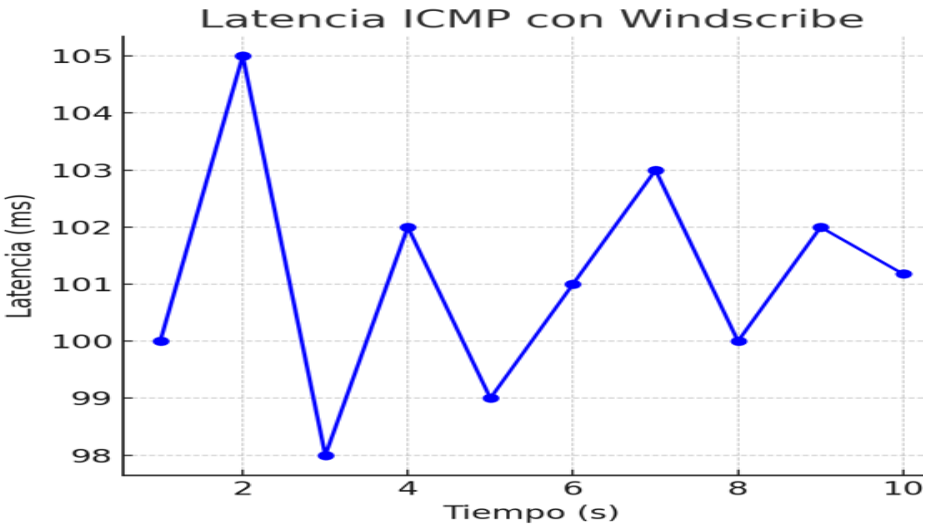

Ilustración 5-*16* Latencia ICMP con Windscribe durante la transmisión

Velocidad de Transferencia

La evaluación de velocidad se realiza a través de iperf3 para tráfico local cifrado y con Speedtest para tráfico externo bajo condiciones reales de red. Como se presenta en la ilustración 5-17, Windscribe alcanza velocidades competitivas, comparables con otros servicios, aunque esta alta velocidad estuvo acompañada de una latencia más elevada.

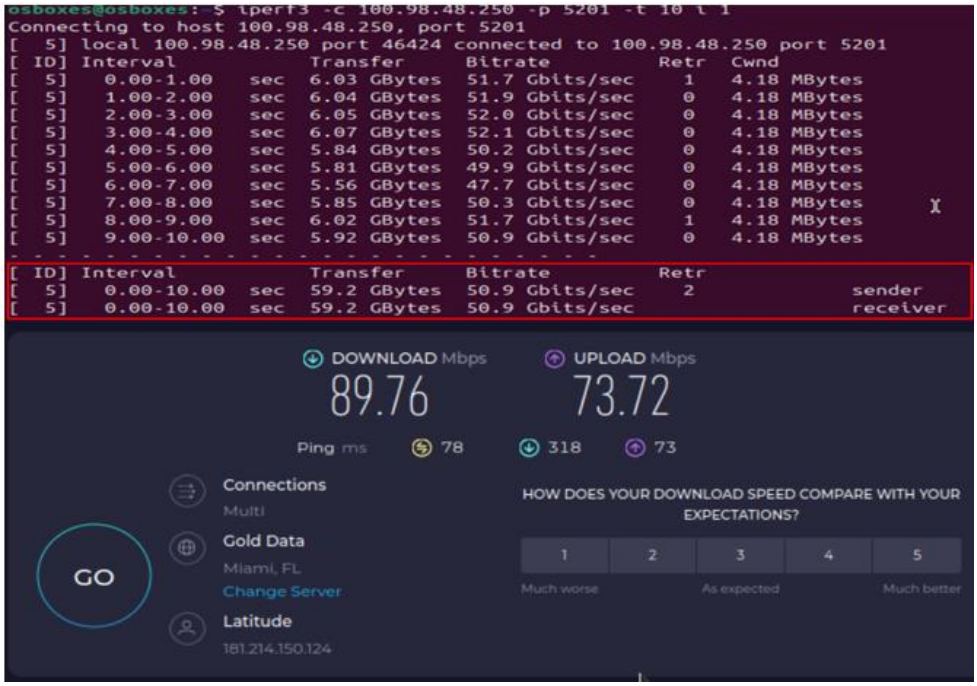

Ilustración 5-17 Resultados de Speedtest y de iperf3

Esto sugiere que, aunque el throughput de datos es alto, el retardo en la primera respuesta puede comprometer ciertas aplicaciones IoT sensibles a la latencia.

En la ilustración 5-18 se representa la evolución de la velocidad de descarga medida en intervalos de un segundo durante una conexión activa mediante Windscribe. A pesar de presentar valores aceptables, con un promedio cercano a los 89 Mbps, se observa una ligera variación en los valores entre segundos consecutivos. Este comportamiento, aunque estable, puede ser menos deseable en sistemas IoT que requieren máxima regularidad en el ancho de banda.

Ilustración 5-1*8* Velocidad de descarga con Windscribe

Carga en la CPU

Durante las pruebas, se monitoriza el uso de CPU con el fin de evaluar el impacto del cifrado aplicado por Windscribe. En la ilustración 5-19 se presentan los resultados obtenidos, donde se registra un uso de CPU de 0.7 % en espacio de usuario y 1.0 % en espacio de sistema, con una carga media baja en general.

```
op - 02:00:19 up  7:30,  1 user,  load average: 0.41, 0.15, 0.07
asks: 209 total,   1 running, 208 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.7 us,  1.0 sy,  0.0 ni, 98.1 id,  0.2 wa,  0.0 hi,  0.0 si,  0.0 st
iB Mem :   3847.6 total,    212.1 free,   2250.6 used,   1703.8 buff/cache
iB Swap:   4096.0 total,   4095.5 free,      0.5 used.   1597.0 avail Mem
```

Ilustración 5-19 Resultados del monitoreo de CPU

El comportamiento observado indica un consumo de CPU similar al registrado con OpenVPN. No se detectaron anomalías ni afectaciones en la estabilidad del sistema durante la ejecución, lo que sugiere que Windscribe opera de manera estable dentro del entorno virtualizado empleado.

En la ilustración 5-20 se detalla el uso del procesador durante la ejecución de Windscribe. El consumo se mantuvo en torno al 0.7 %, con ligeras oscilaciones, lo que indica un comportamiento predecible y dentro de rangos aceptables. Esta eficiencia permite su posible adopción en entornos virtualizados o en dispositivos con capacidades limitadas, aunque no ofrece mejoras significativas respecto a soluciones más ligeras como WireGuard.

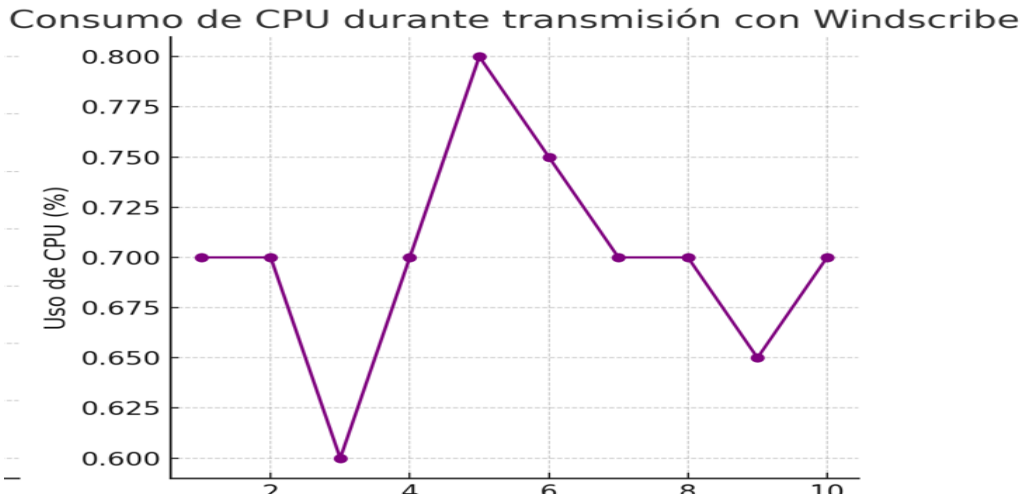

Ilustración 5-19 Consumo de CPU durante transmisión con Windscribe

Tabla Estructurada

En la tabla 4 se presentan los resultados obtenidos de todas las pruebas realizadas usando Windscribe.

Tabla 4 Resultados de rendimiento de Windscribe

	Métrica	Valor
1	Latencia promedio (ms)	101.179
2	Perdida de paquetes (%)	0
3	Velocidad de descarga (Mbps)	89.35
	Velocidad de subida (Mbps)	73
4	Uso de CPU (%)	0.7

5.1.4 Resultados de Pruebas de Rendimiento en NordVPN

Latencia

Para medir la latencia entre el cliente y el servidor utilizando NordVPN, se realizaron pruebas con el comando ping hacia destinos externos como google.com y el servidor DNS 8.8.8.8. Como se muestra en la ilustración 5-20, los resultados evidencian una respuesta rápida y sin variabilidad brusca.

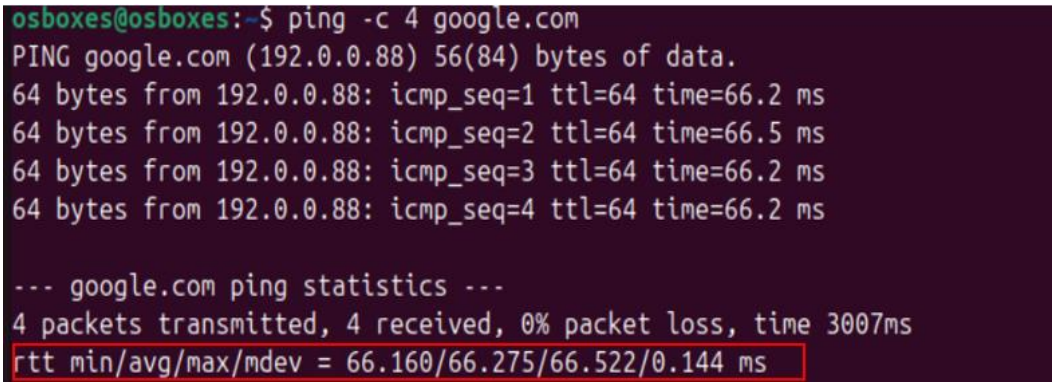

Ilustración 5-20 Resultados del comando ping a google.com

De manera complementaria, la ilustración 5-21 muestra los resultados de ping hacia 8.8.8.8, confirmando un comportamiento estable, sin pérdidas de paquetes y con latencia moderada, en torno a los 66 ms promedio.

```
osboxes@osboxes:~$ ping -c 4 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=119 time=66.9 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=119 time=66.5 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=119 time=66.3 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=119 time=66.8 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 66.343/66.624/66.855/0.209 ms
```

Ilustración 5-21 Resultados del comando ping a 8.8.8.8 (Google DNS)

La herramienta mtr fue utilizada para realizar una evaluación más detallada de los saltos de red y posibles pérdidas en nodos intermedios. La ilustración 5-22 presenta los resultados de esta prueba, basada en el envío de 10 paquetes hacia el servidor DNS. No se registraron pérdidas ni variaciones significativas en el recorrido.

```
osboxes@osboxes:~$ mtr -rw 8.8.8.8
Start: 2025-02-19T15:56:28-0500
HOST: osboxes          Loss%   Snt   Last   Avg  Best  Wrst StDev
  1.|-- 10.5.0.1         0.0%    10   66.3  66.4  66.2  66.9   0.2
  2.|-- 69.67.150.2      0.0%    10   66.8  66.8  66.3  68.7   0.7
  3.|-- 10.10.209.177    0.0%    10   66.4  66.5  66.2  67.0   0.3
  4.|-- 10.10.0.18       0.0%    10   66.1  68.1  66.1  80.7   4.5
  5.|-- 206.41.108.12    0.0%    10   66.6  66.8  66.5  68.0   0.4
  6.|-- 192.178.99.245   0.0%    10   67.5  67.8  67.2  69.5   0.7
  7.|-- 216.239.62.1     0.0%    10   67.5  67.4  67.0  68.4   0.4
  8.|-- dns.google       0.0%    10   67.0  66.6  66.4  67.0   0.2
```

Ilustración 5-22 Resultados del comando mtr (10 paquetes enviados a 8.8.8.8)

En la ilustración 5-23 se representa la latencia medida mediante paquetes ICMP enviados a destinos públicos. A lo largo de los 10 segundos de prueba, los valores oscilaron ligeramente en torno a los 66 ms, sin pérdidas ni retrasos extremos. Esta estabilidad en el retardo sugiere que NordVPN ofrece condiciones favorables para aplicaciones IoT que requieren sincronización continua y respuesta inmediata.

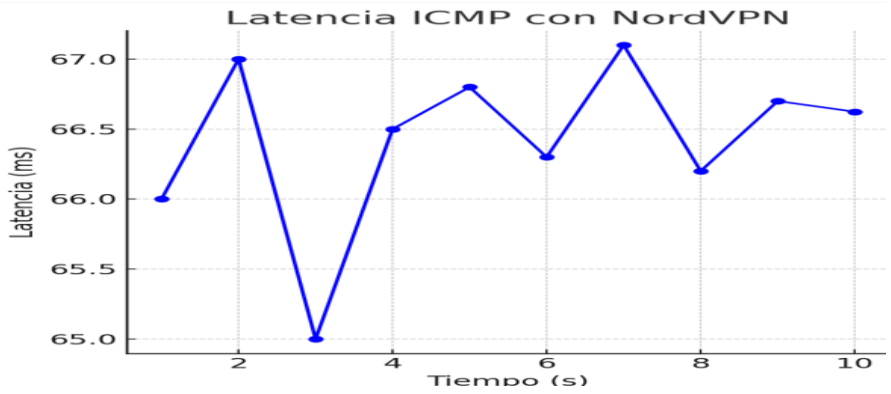

Ilustración 5-23 Latencia ICMP con NordVPN durante la transmisión

Velocidad de Transferencia

La medición del rendimiento en cuanto a velocidad se realiza mediante iperf3 para tráfico local cifrado y Speedtest para evaluación externa. Como se evidencia en la ilustración 5-24, NordVPN mantuvo una velocidad constante de descarga cercana a los 90 Mbps y una velocidad de subida de 73.72 Mbps, superando a OpenVPN y Windscribe.

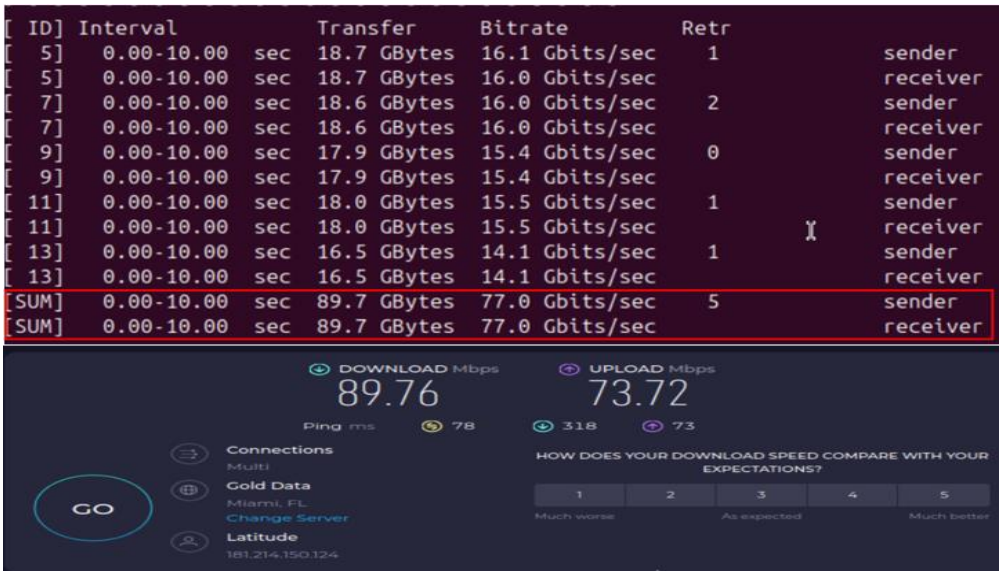

Ilustración 5-24 Resultados de Speedtest y iperf3

Además, se registra una velocidad de transmisión promedio de 77 Mbps en pruebas internas, lo que confirma su solidez como solución VPN en términos de ancho de banda disponible.

En la ilustración 5-25 se muestra la evolución de la velocidad de descarga medida durante un intervalo de 10 segundos con NordVPN activo. Los valores se mantuvieron cercanos a los 90 Mbps, sin caídas abruptas ni variabilidad significativa. Esta consistencia evidencia un desempeño robusto y sostenido, adecuado para flujos de datos en entornos IoT con requerimientos estables de ancho de banda.

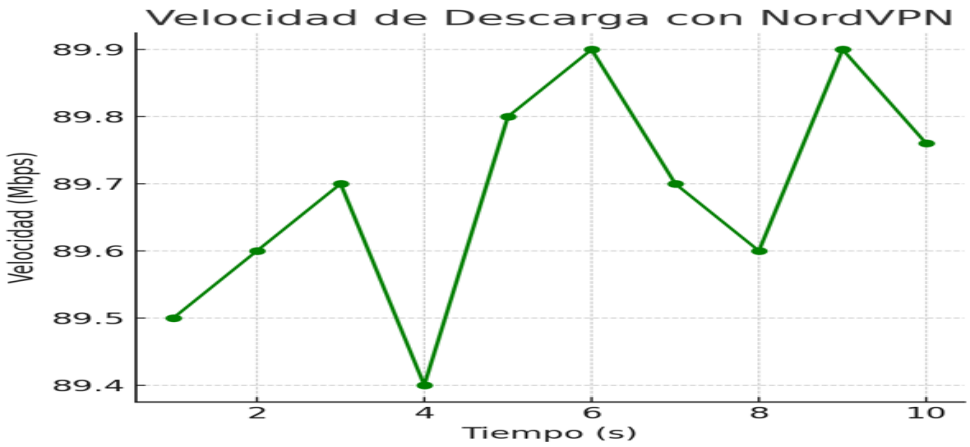

Ilustración 5-25 Velocidad de descarga con NordVPN

Carga en la CPU

El monitoreo del uso del procesador durante la ejecución de NordVPN permite observar su impacto sobre los recursos del sistema. Como se muestra en la ilustración 5-26, se registra un uso de CPU de 0.6 % en espacio de usuario y 1.0 % en espacio de sistema. Estos valores reflejan un consumo estable y comparable al observado con las demás VPN evaluadas.

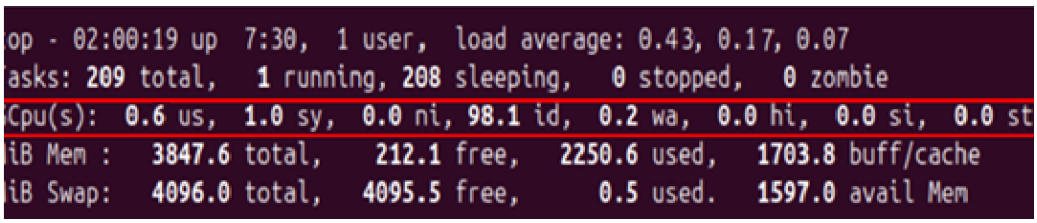

Ilustración 5-26 Resultados del monitoreo de CPU

La carga media del sistema se mantuvo baja durante los tres intervalos de medición (1, 5 y 15 minutos), y no se detectaron cuellos de botella ni procesos que comprometieran la estabilidad operativa. Esto sugiere que NordVPN es viable para su uso en dispositivos IoT, incluso bajo condiciones de procesamiento restringido.

La ilustración 5-27 refleja el uso de CPU a lo largo del tiempo durante la operación de NordVPN. El consumo se mantuvo entre 0.6 % y 0.7 %, sin picos relevantes. Este comportamiento confirma que el protocolo NordLynx, implementado por NordVPN, logra un balance eficiente entre cifrado fuerte y baja demanda de procesamiento, haciéndolo viable para dispositivos con recursos limitados.

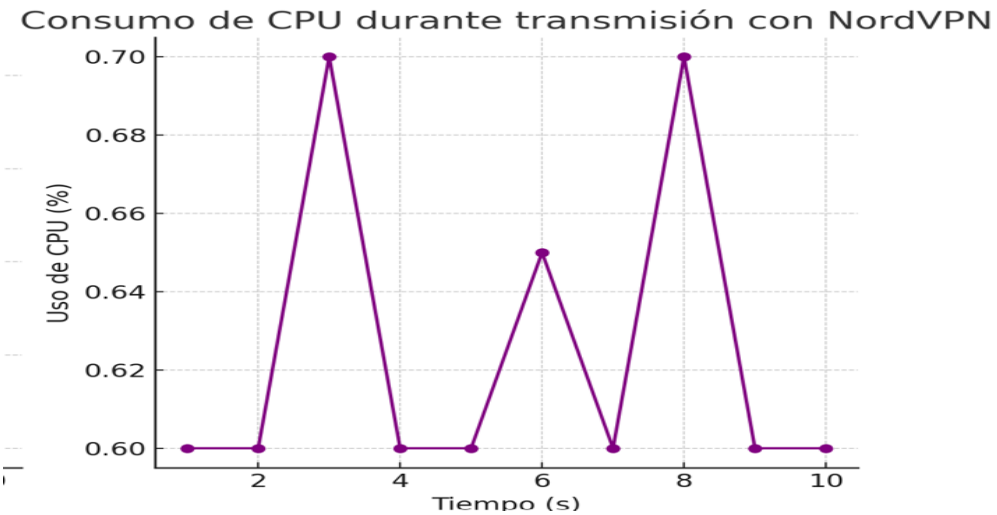

Ilustración 5-27 Consumo de CPU durante la transmisión con NordVPN

Tabla Estructurada

En la tabla 5 se presentan los resultados obtenidos de todas las pruebas realizadas usando NordVPN.

Tabla 5 Resultados de rendimiento de NordVPN

	Métrica	Valor
1	Latencia promedio (ms)	66.624
2	Perdida de paquetes (%)	0
3	Velocidad de descarga (Mbps)	89.76
	Velocidad de subida (Mbps)	73.72
4	Velocidad de transmisión (Mbps)	77.0

5.2 Comparación General de VPNs

La tabla 6 muestra la comparativa de rendimientos que se obtuvieron de las pruebas al utilizar diferentes tecnologías de VPNs.

Tabla 6 Comparativa de rendimiento de las VPNs

	Métrica	WireGuard	OpenVPN	Windscribe	NordVPN
1	Latencia promedio (ms)	1.465	1.069	101.179	66.624
2	Perdida de paquetes (%)	0	0	0	0
5	Velocidad de transmisión (Mbps)	155	141	73	73
6	Uso de CPU (%)	0.1	0.7	0.7	0.6

La ilustración 5-28 permite visualizar de forma comparativa las métricas clave evaluadas en las tecnologías VPN: latencia promedio, velocidad de transmisión y consumo de CPU. WireGuard destaca por ofrecer la mayor velocidad y el menor uso de CPU, lo que lo posiciona como la solución más eficiente para entornos IoT. OpenVPN mantiene un buen equilibrio entre velocidad y latencia, pero requiere más recursos computacionales. Windscribe, pese a su buena velocidad, presenta la mayor latencia del grupo, lo que puede afectar su aplicabilidad en sistemas sensibles al retardo. NordVPN, por su parte, ofrece una velocidad estable con latencia moderada y bajo consumo de CPU, constituyéndose como una alternativa robusta y de fácil implementación.

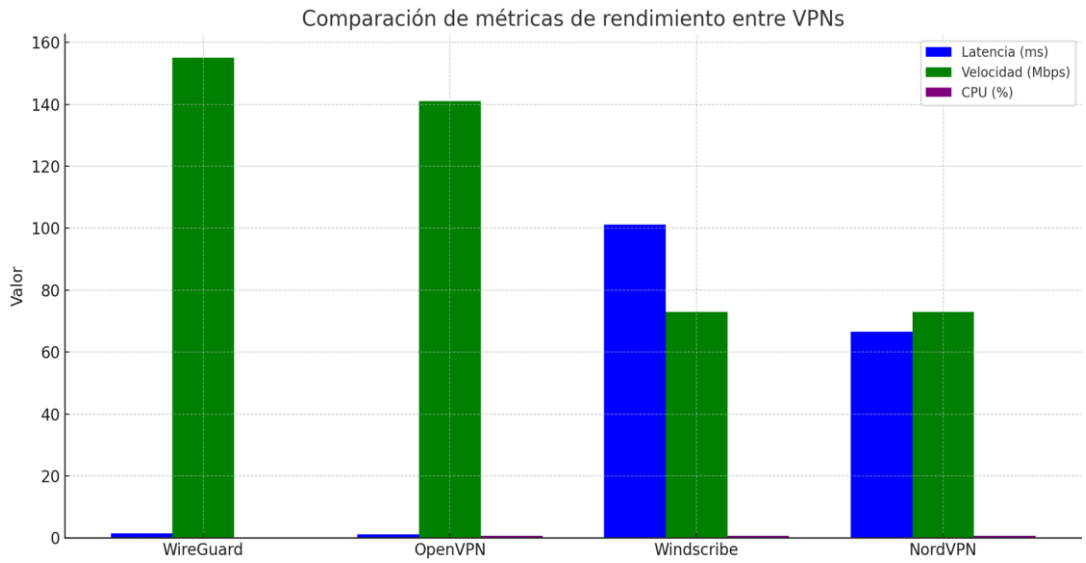

Ilustración 5-28 Comparación de rendimiento entre tecnologías VPN

El análisis de rendimiento de las tecnologías VPN evaluadas evidencia diferencias significativas en métricas clave para entornos IoT. WireGuard y NordVPN destacan por ofrecer las latencias más bajas, de 1.465 ms y 66.624 ms respectivamente, lo que las posiciona como opciones preferentes en aplicaciones donde el tiempo de respuesta es crítico. En contraste, Windscribe presenta una latencia considerablemente elevada, alcanzando los 101.179 ms, lo cual podría comprometer la eficiencia de sistemas sensibles al retardo.

Todas las soluciones mostraron una tasa nula de pérdida de paquetes, lo que garantiza la integridad del flujo de datos durante la transmisión. En términos de velocidad, WireGuard lideró con un promedio de 155 Mbps, seguido de OpenVPN con 141 Mbps; mientras que Windscribe y NordVPN alcanzaron 73 Mbps, valores adecuados, pero menos destacados.

Respecto al uso de CPU, WireGuard demostró ser la alternativa más eficiente, con un consumo del 0.1 %, superando a las demás tecnologías que registraron valores entre 0.6 % y 0.7 %. Esta diferencia resulta relevante para dispositivos IoT con capacidades limitadas de procesamiento.

No obstante, es importante señalar que, pese a su alto rendimiento, las soluciones gratuitas como WireGuard y OpenVPN implicaron una mayor complejidad durante su configuración, requiriendo conocimientos técnicos avanzados. En contraste, las opciones comerciales como Windscribe y NordVPN, aunque menos eficientes en ciertas métricas, ofrecieron una integración inmediata mediante aplicaciones gráficas, lo que representa una ventaja en escenarios donde la facilidad de despliegue es prioritaria.

5.3 Análisis de mecanismos de cifrado y autenticación en soluciones VPNs evaluadas

La robustez de una red privada virtual (VPN) depende directamente de los mecanismos de cifrado y autenticación que utiliza para proteger la confidencialidad, integridad y autenticidad de los datos transmitidos.

5.3.1 WireGuard

WireGuard es una tecnología VPN moderna diseñada para ser simple, rápida y segura.

Cifrado

WireGuard utiliza el algoritmo ChaCha20-Poly1305 para cifrar y autenticar los datos.

- ChaCha20 es un cifrador de flujo optimizado para software, resistente a ataques de criptoanálisis y más rápido que AES en dispositivos sin aceleración de hardware.
- Poly1305 se utiliza para autenticar cada paquete de datos, garantizando la integridad de la información.

Este algoritmo ofrece seguridad de 256 bits, considerada extremadamente robusta para los estándares actuales.

Autenticación

WireGuard basa su autenticación en un sistema de claves públicas y claves privadas, similar a SSH.

- Cada dispositivo genera su par de claves.
- La autenticación entre cliente y servidor se realiza intercambiando y verificando las claves públicas.
- No se requiere intercambio dinámico de claves en cada sesión, reduciendo así la superficie de ataque.

Análisis de Seguridad

WireGuard proporciona:

- Autenticación robusta basada en criptografía de curva elíptica (Curve25519).
- Confidencialidad fuerte debido a ChaCha20.
- Baja complejidad en la configuración, lo que minimiza errores humanos.

WireGuard es considerado muy seguro y adecuado para entornos IoT por su ligereza y alta eficiencia

5.3.2 OpenVPN

OpenVPN es una solución madura y ampliamente adoptada en múltiples industrias.

Cifrado

OpenVPN soporta múltiples algoritmos de cifrado, pero en esta investigación se utilizó:

AES-256-CBC y AES-256-GCM

AES (Advanced Encryption Standard) es el estándar actual de cifrado a nivel gubernamental y empresarial, aprobado por NIST

AES-256 garantiza seguridad con claves de 256 bits.

Autenticación

OpenVPN utiliza certificados digitales en conjunto con el protocolo TLS:

- Cada cliente y servidor debe poseer un certificado válido.
- El handshake inicial es protegido mediante intercambio de claves TLS.
- Se pueden implementar capas adicionales como HMAC (hash-based message authentication codes) para protección contra ataques de denegación de servicio.

Análisis de Seguridad

OpenVPN proporciona:

- Alta robustez criptográfica, especialmente con AES-256 y TLS 1.3.
- Seguridad de autenticación mutua mediante certificados X.509.
- Flexibilidad de configuración, aunque su complejidad puede ser un riesgo si no se gestiona adecuadamente.

OpenVPN es extremadamente seguro y apto para entornos IoT críticos, aunque con mayor requerimiento de CP

5.3.3 WINDSCRIBE

Windscribe es un proveedor VPN comercial orientado a usuarios que buscan facilidad de uso.

Cifrado

Windscribe utiliza:

- AES-256-GCM como principal algoritmo de cifrado
- Implementa también perfect forward secrecy (PFS) mediante intercambio de claves Diffie-Hellman.

- AES-256 en modo GCM proporciona cifrado y autenticación de datos simultáneamente.

Autenticación

- Basada principalmente en usuario y contraseña.
- Durante el establecimiento del túnel, se emplea TLS para proteger el intercambio de claves y credenciales.

Análisis de Seguridad

Windscribe ofrece:

- Buen nivel de cifrado gracias a AES-256-GCM.
- Autenticación dependiente de contraseñas, lo cual es un posible punto débil si no se combina con medidas como autenticación multifactor.

Windscribe es seguro para conexiones IoT, pero la autenticación basada únicamente en contraseñas puede ser un riesgo potencial

5.3.4 NordVPN

NordVPN es una de las soluciones comerciales líderes a nivel global, conocida por su enfoque en la seguridad.

Cifrado

NordVPN utiliza:

- ChaCha20-Poly1305 en su protocolo NordLynx (una adaptación mejorada de WireGuard)
- También soporta AES-256 para conexiones estándar.
- ChaCha20 proporciona velocidades superiores en entornos móviles y redes inestables.

Autenticación

NordVPN emplea:

- Usuario y contraseña para la autenticación inicial.
- Túneles privados para ocultar metadatos y reforzar el anonimato (Double NAT System).
- Opcionalmente, puede integrar autenticación multifactor (MFA) para acceso a la cuenta.

Análisis de Seguridad

NordVPN ofrece:

- Alta seguridad criptográfica mediante NordLynx y doble NAT.
- Protección adicional contra fugas de IP y DNS.
- Mejoras en privacidad frente a soluciones tradicionales de usuario/contraseña.

NordVPN es muy recomendable para entornos IoT que requieren tanto seguridad como rendimiento.

5.3.5 Comparación general de Cifrado y Autenticación de vpns

La tabla 7 muestra la comparación de las diferentes tecnologías de cifrados que utilizan las VPNs que fueron evaluadas.

Tabla 7 Cifrados y Autenticación de vpns

	VPN	Algoritmo de Cifrado	Métodos de Autenticación	Evaluación de Robustez
1	WireGuard	ChaCha20-Poly1305	Claves públicas	Muy alta
2	OpenVPN	AES-256-CBC / AES-256-GCM	Certificados + TLS	Muy alta
3	Windscribe	AES-256-CBC / AES-256-GCM	Usuario/Contraseña + TLS	Alta
4	NordVPN	ChaCha20-Poly1305 (NordLynx)	Usuario/Contraseña + Doble NAT	Muy alta

En la presente comparación de tecnologías VPN aplicadas a entornos IoT, se evaluaron los algoritmos de cifrado, métodos de autenticación y la robustez general de cuatro soluciones ampliamente utilizadas: WireGuard, OpenVPN, Windscribe y NordVPN. WireGuard y NordVPN destacaron por implementar el cifrado ChaCha20-Poly1305, considerado uno de los más eficientes y modernos para dispositivos de bajo consumo energético, como los empleados en el ecosistema IoT. OpenVPN, aunque sigue siendo un estándar de referencia en seguridad, emplea cifrados AES-256 en modos CBC o GCM combinados con certificados y TLS, lo que le otorga una

robustez muy alta, pero con una mayor exigencia computacional. Windscribe, por su parte, también utiliza AES-256, pero basa su autenticación en credenciales de usuario complementadas con TLS, lo que representa una debilidad relativa frente a soluciones que emplean autenticación por claves o infraestructura de doble capa como NordVPN. En conjunto, se concluye que WireGuard y NordVPN ofrecen una mejor relación entre seguridad, rendimiento y facilidad de implementación, lo que los convierte en opciones óptimas para entornos IoT que requieren una protección eficiente y confiable en redes públicas.

5.4 Resultados de las pruebas de seguridad de las tecnologías de VPNS

5.4.1 Resultados del vulnerabilidades y riesgos en WireGuard

Para evaluar la robustez del protocolo WireGuard frente a ataques activos y análisis pasivo de tráfico, se realizan dos pruebas principales: una inspección de tráfico convencional mediante herramientas de análisis (Wireshark y tcpdump) y una simulación de ataque *Man-in-the-Middle* (MITM) con Bettercap. Adicionalmente, se implementa un entorno simulado de IoT donde el cliente VPN emula un sensor que publicaba datos MQTT a un broker local, a fin de validar el comportamiento de WireGuard en contextos de comunicación propios del Internet de las Cosas.

Resultados del análisis con Wireshark y tcpdump

Inicialmente, se ejecuta una captura de tráfico desde la interfaz virtual wg0 del cliente Ubuntu utilizando Wireshark. En esta captura se visualizaron flujos etiquetados como TLS, los cuales, aunque visibles, no representan una vulnerabilidad de seguridad como se muestra en la ilustración 5-29. Estos datos fueron interpretados por el sistema operativo como tráfico de capa de aplicación ya descifrado localmente, lo cual es un comportamiento esperado en el extremo del túnel VPN.

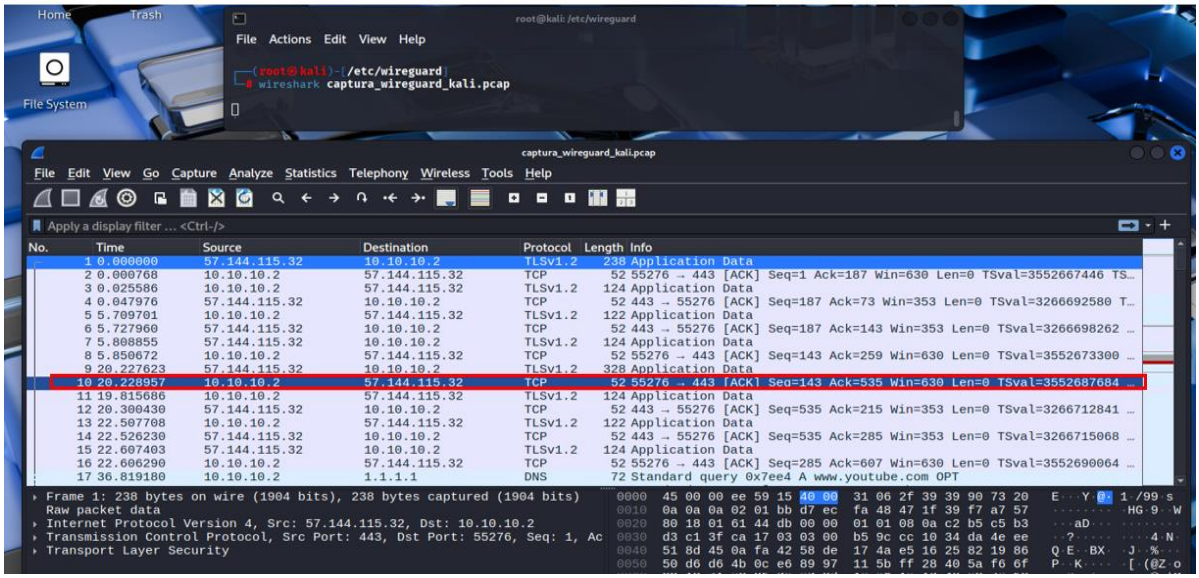

Ilustración 5-29 Visualización en Wireshark de los flujos TLS capturados en la interfaz virtual wg0.

Posteriormente, para contrastar esta información, se realiza una captura desde la interfaz física eth0, observándose únicamente paquetes UDP encapsulados bajo el protocolo WireGuard. No se detectaron cabeceras HTTP, DNS ni contenido en texto plano, como se muestra en la ilustración 5-30.

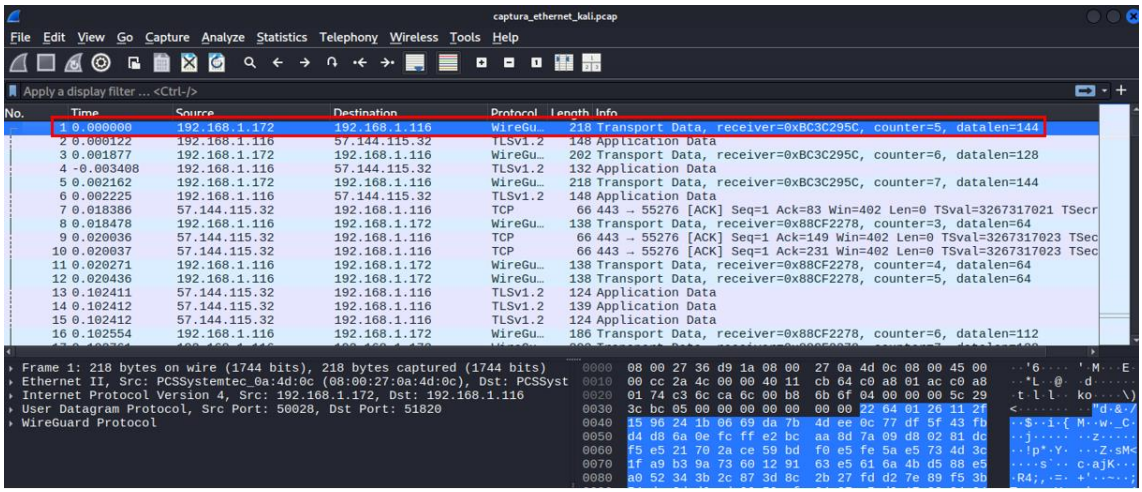

Ilustración 5-30 Captura de tráfico en la interfaz física eth0: paquetes UDP con cifrado WireGuard.

Resultados de ataque MITM con tráfico convencional

Con el fin de simular un ataque activo, se emplea Bettercap desde una máquina Kali Linux conectada a la misma red local. Al activar los módulos arp.spoof y net.sniff, Bettercap logra capturar únicamente tráfico mDNS (Multicast DNS), incluyendo consultas PTR y nombres de host, como se muestra en la ilustración 5-31.

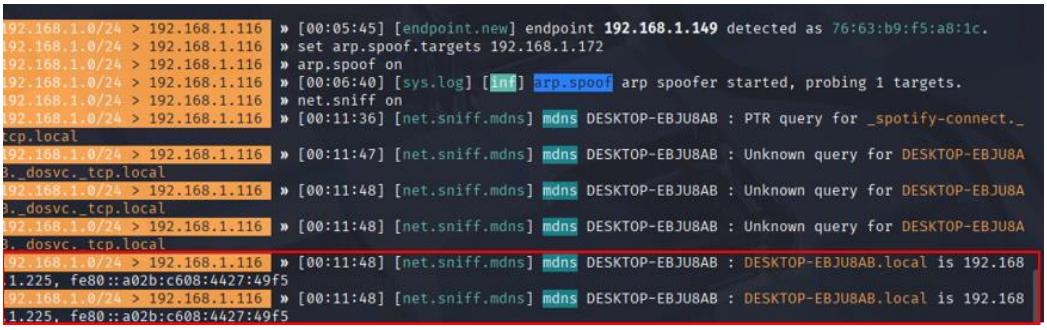

Ilustración 5-31 Resultados de Bettercap: capturas de tráfico mDNS sin exposición de datos sensibles.

En ningún momento se observa tráfico perteneciente a protocolos sensibles como HTTP, DNS o MQTT. Adicionalmente, se utiliza el comando net.show para enumerar los dispositivos en la red como se muestra en la ilustracion 5-32, confirmándose que la visibilidad del atacante se limita a información de red local sin acceso al contenido del túnel cifrado.

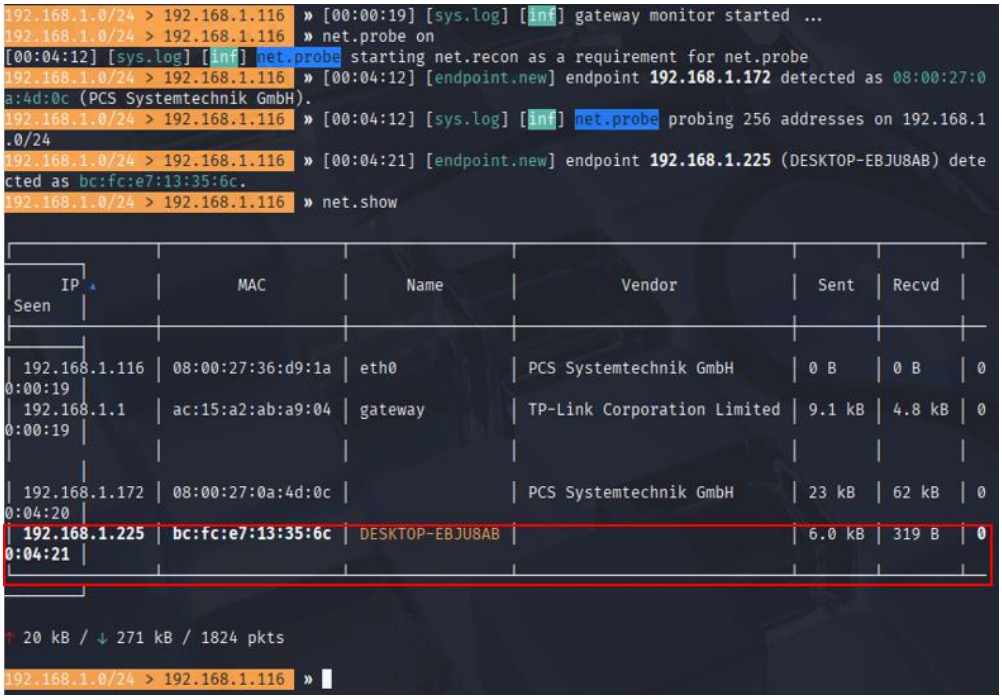

Ilustración 5-32 Dispositivos detectados mediante Bettercap en la red local.

Resultados del ataque MITM con tráfico MQTT

En una segunda fase, se replica el ataque en un entorno simulado donde el cliente VPN actuó como un sensor IoT que publicaba datos periódicos mediante el protocolo MQTT hacia un broker local. Durante esta prueba, los módulos de suplantación ARP y captura de Bettercap fueron activados nuevamente, pero no se logra interceptar ningún mensaje relacionado con el protocolo MQTT.

Bettercap mostró de forma reiterada el mensaje “arp.spoof could not find spoof targets”, como se evidencia en la ilustración 5-33, lo cual indica que el atacante no logra posicionarse en medio del flujo de comunicación protegido.

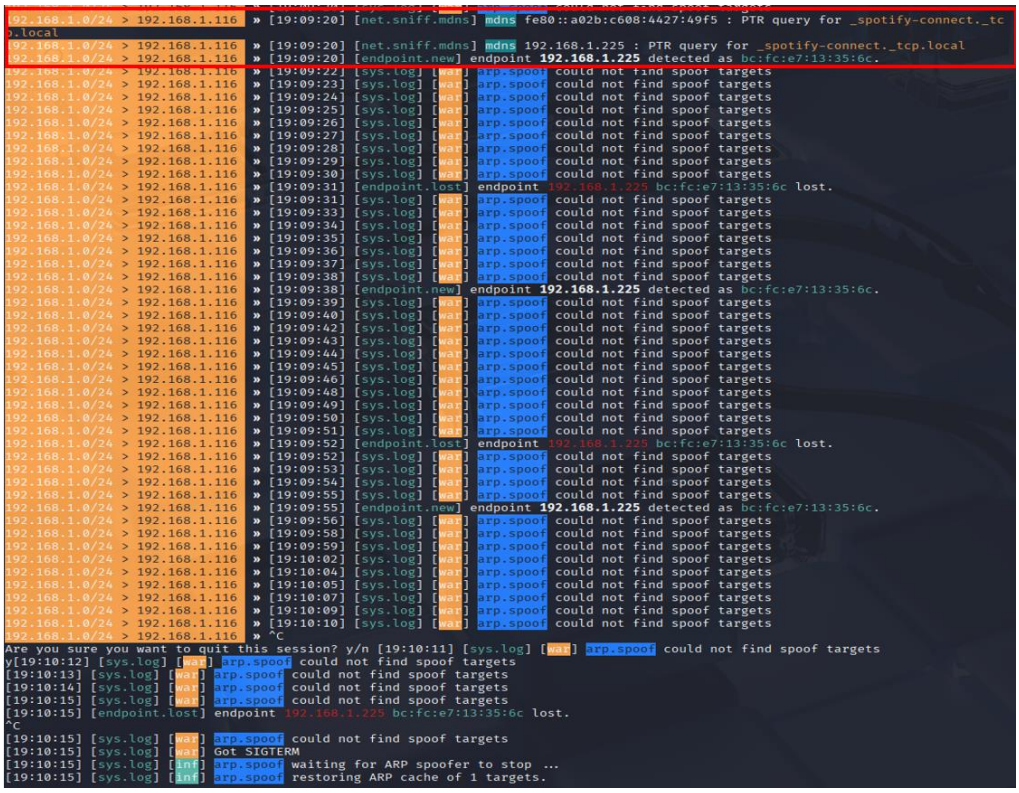

Ilustración 5-33 Captura de tráfico mDNS sin exposición de mensajes MQTT.

El tráfico registrado volvia a limitarse a paquetes mDNS y consultas PTR, sin aparición de mensajes en texto plano, publicaciones MQTT ni flujos vulnerables. Esto sugiere que el encapsulamiento y cifrado de extremo a extremo de WireGuard fueron efectivos incluso frente a un atacante presente en la misma red física.

Observaciones técnicas

Los resultados obtenidos en ambas pruebas confirman la solidez de la implementación criptográfica de WireGuard. Tanto en el análisis pasivo como en el ataque activo, el protocolo mantuvo la confidencialidad de los datos. La presencia de flujos TLS en wg0 responde a la interpretación local del tráfico por parte del sistema operativo y no representa una debilidad del cifrado.

Durante las simulaciones de ataque MITM, no fue posible interceptar tráfico HTTP, DNS ni MQTT. La única información observable correspondió a servicios de descubrimiento local (mDNS), lo cual es un comportamiento esperado en redes LAN y no compromete la seguridad del canal VPN.

5.4.2 Resultados de análisis de vulnerabilidades y riesgos en OpenVPN

La evaluación de seguridad sobre OpenVPN incluye un análisis pasivo con Wireshark, la detección de posibles fugas de DNS, y simulaciones de ataques Man-in-the-Middle (MITM) tanto en tráfico general como en un entorno IoT con publicaciones MQTT. Estas pruebas fueron diseñadas para examinar la efectividad del túnel cifrado frente a amenazas comunes en redes locales compartidas.

Resultados de Wireshark

Para verificar la privacidad del canal VPN, se analiza si las consultas DNS eran canalizadas a través del túnel cifrado. El comando resolvectl status en el cliente Ubuntu muestra que las resoluciones de nombres seguían dirigiéndose a la interfaz física enp0s3, en lugar de utilizar la interfaz virtual tun0, lo que indica que las consultas salían del túnel como se muestra en la ilustración 5-34.

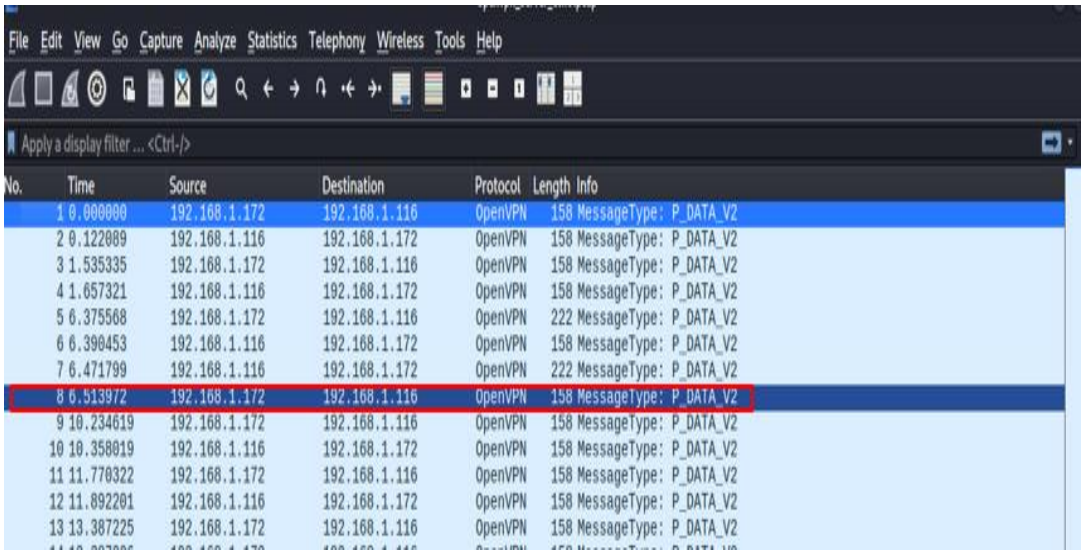

Ilustración 5-34 Captura de paquetes OpenVPN cifrados visualizados en Wireshark desde el servidor.

Análisis de fuga DNS

Para verificar la privacidad del canal VPN, se analiza si las consultas DNS eran canalizadas a través del túnel cifrado. El comando resolvectl status en el cliente Ubuntu muestra que las resoluciones de nombres seguían dirigiéndose a la interfaz física enp0s3, en lugar de utilizar la interfaz virtual tun0, lo que indica que las consultas salían del túnel como indica la ilustración 5-35.

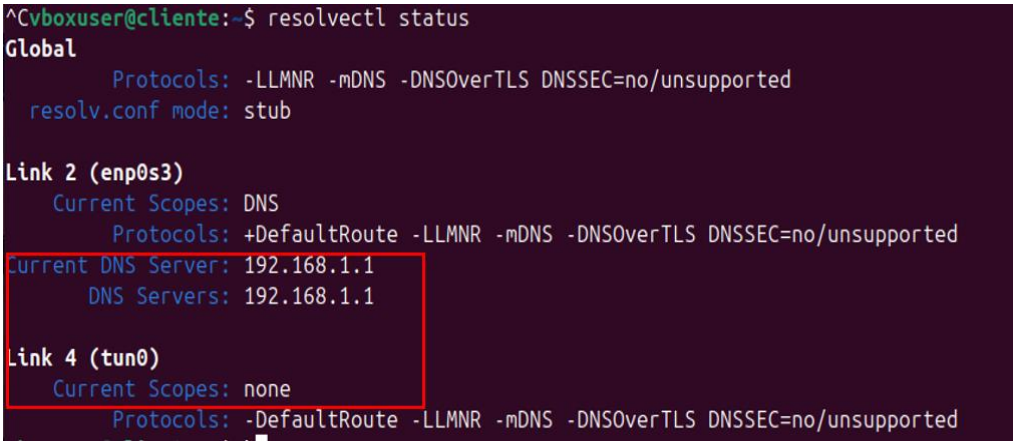

Ilustración 5-35 Salida del comando resolvectl: las consultas DNS se realizan fuera del túnel VPN.

Este hallazgo fue confirmado mediante una herramienta externa de fuga DNS. Los resultados revelaron que las solicitudes eran atendidas por servidores del proveedor ISP Netlife como se muestra en la ilustracion 5-36, lo que implica una exposición de metadatos a terceros.

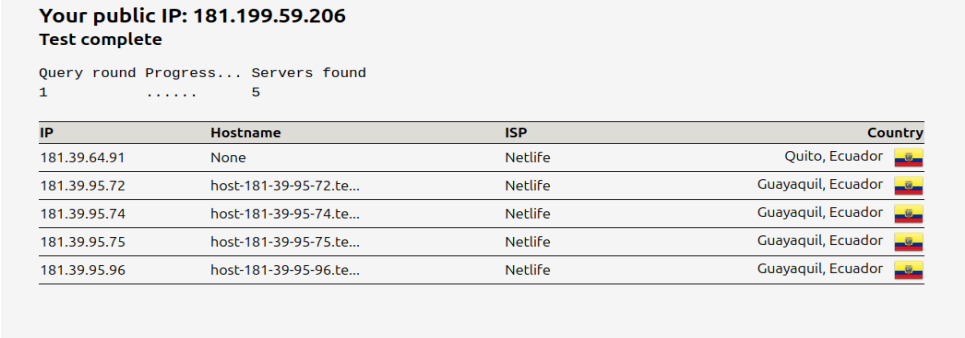

Ilustración 5-36 Resultado de prueba de fuga DNS: los servidores consultados son externos al túnel VPN.

Este comportamiento representa una vulnerabilidad de privacidad significativa, especialmente en entornos IoT, donde los dispositivos suelen realizar conexiones automáticas a servicios externos. La exposición de dominios consultados podría facilitar ataques de rastreo o reconocimiento.

Resultados de ataque MITM con tráfico convencional

En las pruebas de interceptación activa, Bettercap fue ejecutado desde un nodo Kali Linux conectado a la misma red física. Como se observa en la ilustración 5-37, aunque el módulo arp.spoof fue activado y el cliente Ubuntu fue detectado correctamente, se generan mensajes repetidos indicando que no se encontraron

objetivos para la suplantación. Esto sugiere que, aunque el atacante logra visibilidad en la red, el túnel VPN impidió establecer una posición intermedia efectiva.

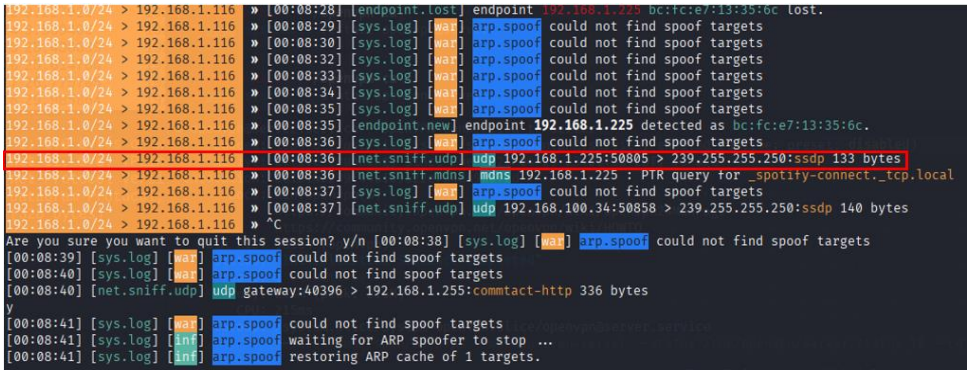

Ilustración 5-37 Detección de dispositivos y ejecución del módulo arp.spoof durante el ataque MITM.

Durante la sesión, se intercepta tráfico multicast relacionado con SSDP y mDNS, como se muestra en la ilustración 5-38. No se capturaron paquetes HTTP, TLS, DNS ni contenido de aplicación, lo cual indica que el cifrado punto a punto se mantuvo intacto.

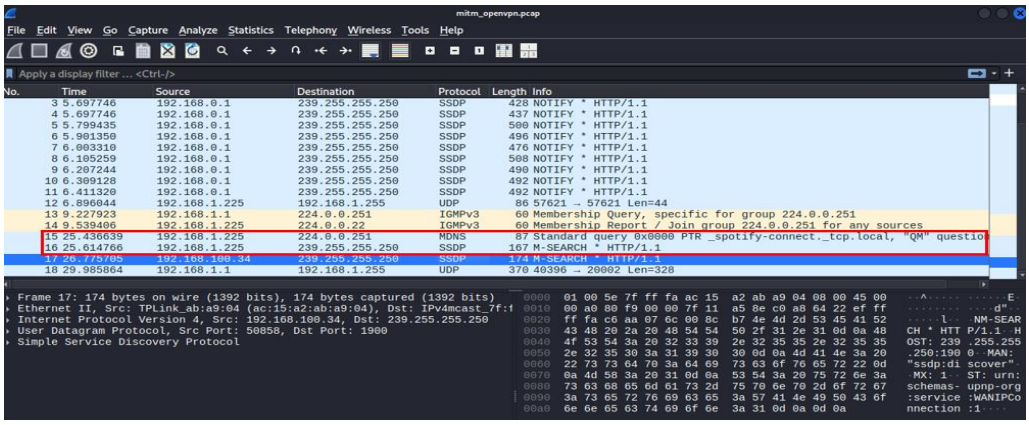

Ilustración 5-38 Visualización del tráfico interceptado mediante Wireshark: paquetes mDNS y SSDP sin exposición de contenido sensible.

Resultados de ataque MITM con tráfico MQTT

En un segundo escenario, se simula un entorno IoT en el que el cliente VPN actua como un sensor publicando datos sobre un broker MQTT local. Bettercap fue configurado nuevamente para interceptar el tráfico entre el cliente (IP 192.168.1.172) y su destino. A pesar de la ejecución sostenida del ataque, no se observa ninguna publicación MQTT en texto plano como se muestra en la ilustración 5-39.

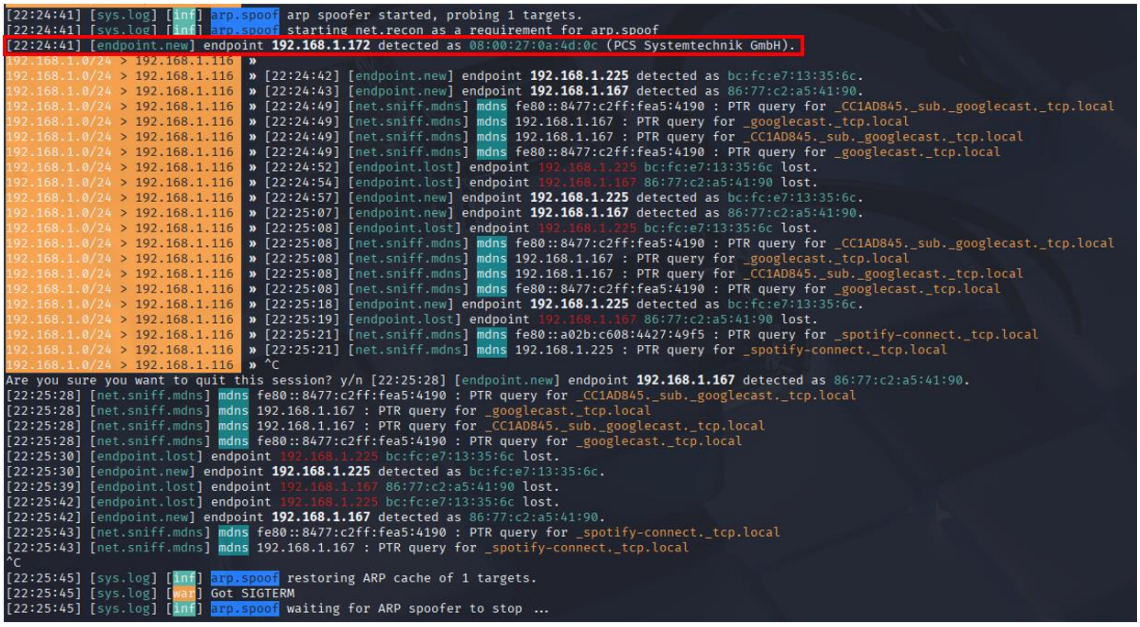

Ilustración 5-39 Resultado de Bettercap sin interceptación de mensajes MQTT, mostrando solo tráfico mDNS

Estos resultados confirman que OpenVPN logra encapsular adecuadamente el tráfico de aplicación, incluso en un entorno simulado donde el atacante tenía acceso completo a la red LAN.

Observaciones técnicas relevantes

OpenVPN demuestra ser una solución eficaz en cuanto a la protección del contenido de las comunicaciones. El tráfico fue correctamente encapsulado en la interfaz tun0, impidiendo su visualización desde el exterior. Sin embargo, la presencia de una fuga DNS representa una debilidad crítica desde el punto de vista de privacidad, ya que permite identificar dominios consultados por los dispositivos protegidos.

En contextos IoT, donde cada conexión a servicios externos puede revelar comportamientos sensibles del sistema, este tipo de vulnerabilidad debe ser mitigado para garantizar una protección integral.

5.4.3 Resultados de análisis de vulnerabilidades y riesgos en Windscribe

Para evaluar la seguridad de la tecnología VPN Windscribe, se realizan pruebas de análisis pasivo con Wireshark y tcpdump, verificación de fuga DNS, y simulaciones de ataque tipo *Man-in-the-Middle* (MITM), tanto en tráfico general como en un entorno IoT con transmisiones MQTT. Estas pruebas permiten determinar la

capacidad de Windscribe para mantener la confidencialidad de los datos frente a observadores y atacantes locales.

Resultado de Wireshark y tcpdump

La inspección del tráfico desde la máquina atacante (Kali Linux) mediante tcpdump y Wireshark muestra que todo el tráfico estaba encapsulado en paquetes cifrados. No se identificaron cabeceras HTTP, dominios, ni información interpretable desde la red local, como se muestra en la ilustración 5-40, lo cual indica que Windscribe encapsula correctamente los datos dentro del túnel WireGuard.

Ilustración 5-40 Visualización de tráfico cifrado sin información legible mediante tcpdump.

Resultados de fuga de DNS

Con el objetivo de detectar posibles fugas de solicitudes DNS fuera del canal VPN, se realiza una prueba con el comando dig desde el cliente. Durante esta prueba, tcpdump ejecutado sobre la interfaz física eth0 no captura ningún paquete relacionado con DNS como indica la ilustración 5-41, lo que sugiere que las resoluciones de nombres fueron gestionadas correctamente dentro del túnel cifrado.

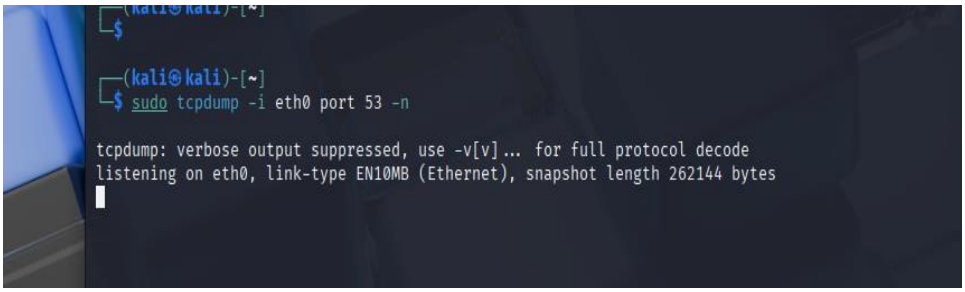

Ilustración 5-41 Resultados de la captura de fuga de DNS

El sistema reporta que la resolución se realiza mediante el resolutor local 127.0.0.53, lo cual indica que Windscribe evitó correctamente que estas solicitudes

se propagaran fuera del canal seguro. Esto representa una protección efectiva frente a uno de los vectores de fuga más comunes en VPNs mal configuradas.

Resultados del ataque MITM con tráfico convencional

Durante la simulación del ataque MITM, Bettercap fue configurado sobre la interfaz eth0 y ejecuta los módulos arp.spoof y net.sniff. A pesar de que el monitoreo fue sostenido y la red fue escaneada correctamente, los únicos paquetes capturados fueron mensajes mDNS y SSDP, como se muestra en las ilustraciones 5-42 y 5-43. No se identificaron flujos sensibles ni cabeceras de protocolos de aplicación.

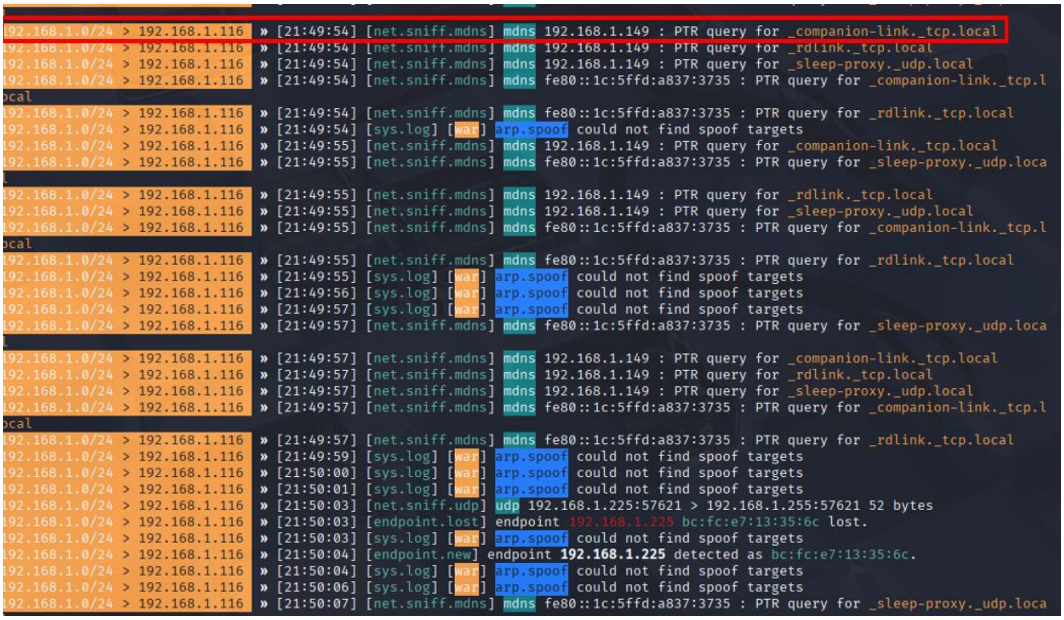

Ilustración 5-42 Captura de paquetes mDNS y SSDP interceptados durante el ataque.

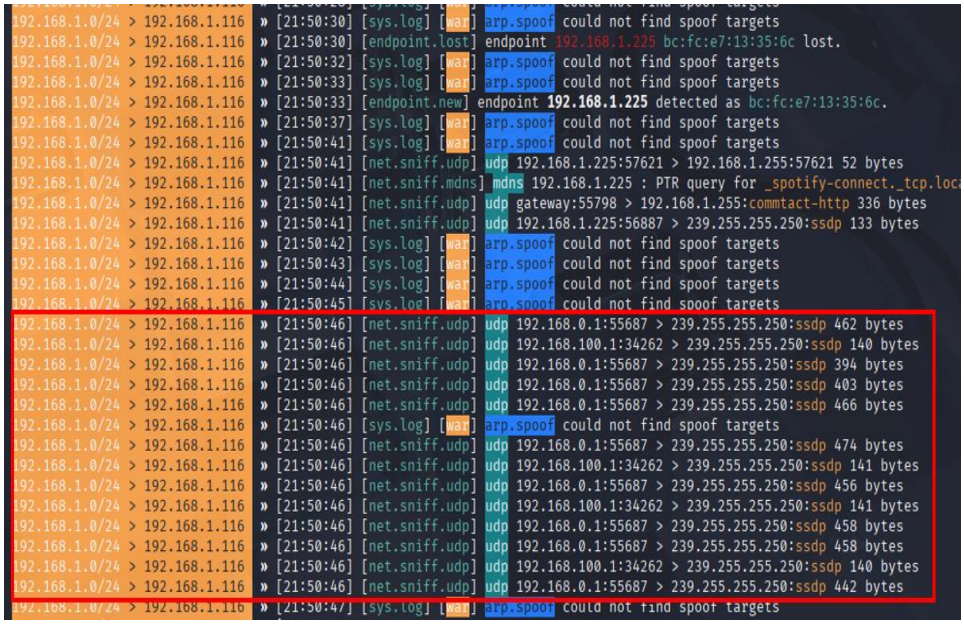

Ilustración 5-43 Actividad de descubrimiento de red detectada por Bettercap.

Simulación de ataque MITM con tráfico MQTT

En una segunda prueba, se simula un dispositivo IoT que publicaba datos de temperatura en un topic MQTT utilizando Windscribe como canal de comunicación. Bettercap detecta correctamente al cliente en la red local (IP 192.168.1.172) y ejecuta los módulos de suplantación y captura. Sin embargo, como se observa en la ilustración 5-44, no se logra capturar contenido MQTT en texto plano.

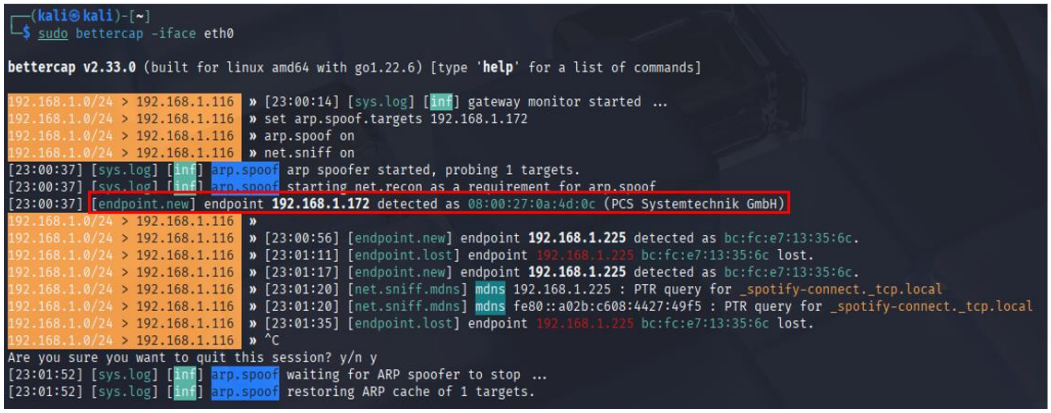

Ilustración 5-44 Resultados del intento de interceptación (solo tráfico mDNS)

La única información visible fue mensajes de descubrimiento de red generados automáticamente, como consultas PTR hacia servicios tipo _spotify-connect. _tcp. local, los cuales no afectan la confidencialidad del canal MQTT.

Observaciones técnicas relevantes

Los resultados obtenidos confirman que Windscribe proporciona un nivel adecuado de protección para el tráfico de red. No se detectaron fugas DNS, el tráfico fue encapsulado correctamente en el túnel VPN y no se logra visualizar contenido sensible, ni siquiera durante simulaciones activas de interceptación. La IP pública observada durante la sesión corresponde a un nodo remoto (por ejemplo, 37.120.215.169), validando que la conexión fue redirigida correctamente a través del túnel seguro.

El rendimiento durante la navegación fue fluido y no se observan interrupciones del canal cifrado. Si bien Windscribe cumple con los requisitos técnicos de confidencialidad, se recomienda su implementación en entornos IoT acompañada de mecanismos adicionales de autenticación para fortalecer la protección contra accesos no autorizados.

5.4.4 Resultados de análisis de vulnerabilidades y riesgos en NordVPN

La evaluación de seguridad de NordVPN se realiza empleando herramientas de análisis pasivo como tcpdump, simulaciones de ataque tipo *Man-in-the-Middle* (MITM), y verificación de fugas DNS. Asimismo, se replica un entorno IoT donde el cliente VPN actuó como sensor que publicaba datos MQTT, a fin de validar la protección del tráfico ante ataques locales. Las pruebas se ejecutaron con NordVPN en su configuración por defecto basada en el protocolo NordLynx, una implementación optimizada de WireGuard.

Resultados de tcpdump

Durante la sesión de análisis pasivo, se ejecuta tcpdump desde Kali Linux sobre la interfaz física de red. Al revisar el archivo. pcap generado, no se encontraron paquetes visibles, lo cual sugiere que todo el tráfico del cliente fue correctamente encapsulado en el túnel VPN, sin exposición hacia la red local como indica la ilustración 5-45.

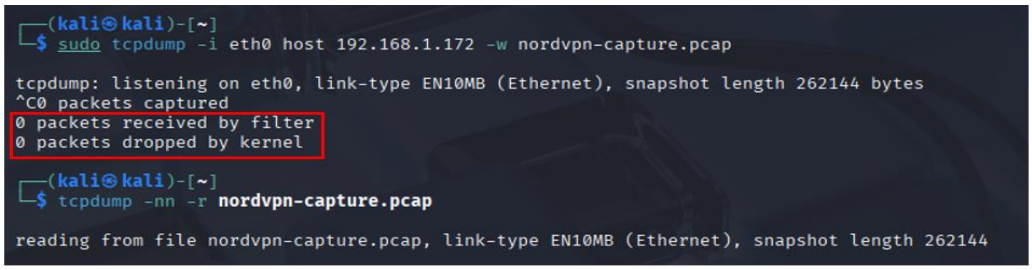

Ilustración 5-45 Verificación de archivo. pcap sin paquetes interceptados desde la red local.

Adicionalmente, se realiza un intento de comunicación ICMP desde Kali hacia el cliente Ubuntu. La solicitud falla, como se muestra en la ilustración 5-46, lo que indica un aislamiento efectivo a nivel de red, impidiendo respuestas directas a nodos no autorizados.

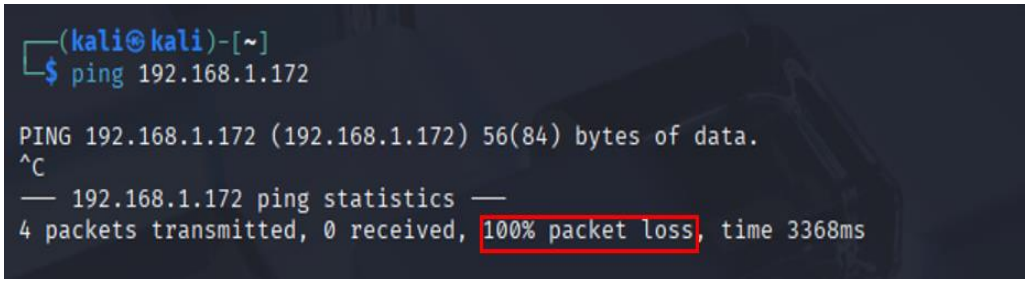

Ilustración 5-46 Intento de comunicación fallido desde el atacante hacia el cliente bajo NordVPN.

Resultados de fuga de DNS

Para verificar posibles fugas de solicitudes DNS fuera del túnel, se analiza el tráfico en el puerto 53 mediante tcpdump. Como se muestra en la ilustración 5-47, no se capturaron paquetes DNS, lo cual indica que todas las consultas fueron encapsuladas correctamente. El sistema resuelve dominios mediante el resolver local 127.0.0.53, sin filtración hacia servidores externos visibles desde la red LAN.

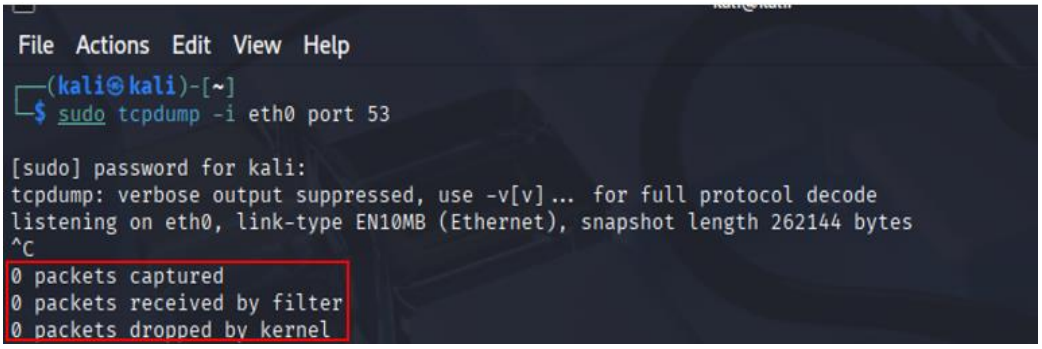

Ilustración 5-47 Resultado final de tcpdump: sin paquetes DNS capturados, validando la ausencia de fuga.

Resultados de ataque MITM con tráfico convencional

Durante el ataque MITM, Bettercap logra detectar al cliente y capturar algunos paquetes UDP y TCP cifrados, dirigidos a servidores externos pertenecientes al rango de direcciones de NordVPN. Como se observa en la ilustración 5-48, no se accede a contenido sensible ni se visualizaron cabeceras de protocolos de aplicación.

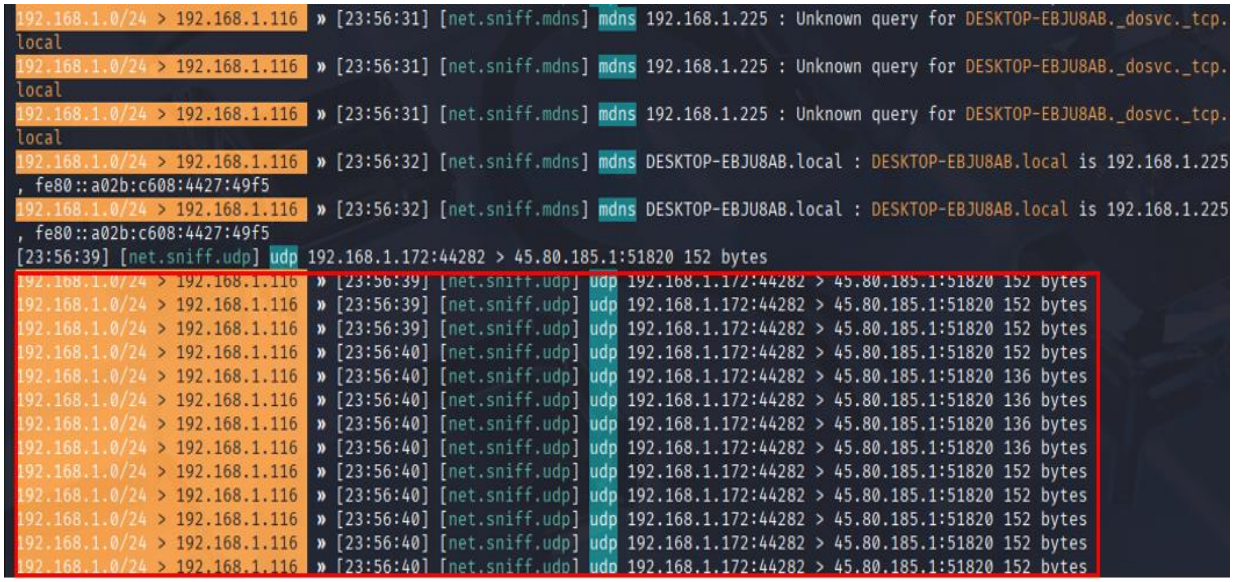

Ilustración 5-48 Captura de tráfico UDP y TCP cifrado durante la simulación MITM.

La herramienta también detecta tráfico mDNS y SSDP relacionado con el descubrimiento automático de dispositivos en la red, como se muestra en la ilustración 5-49. Si bien esta información puede ser visible en entornos locales, no compromete directamente la confidencialidad de los datos transmitidos por la VPN.

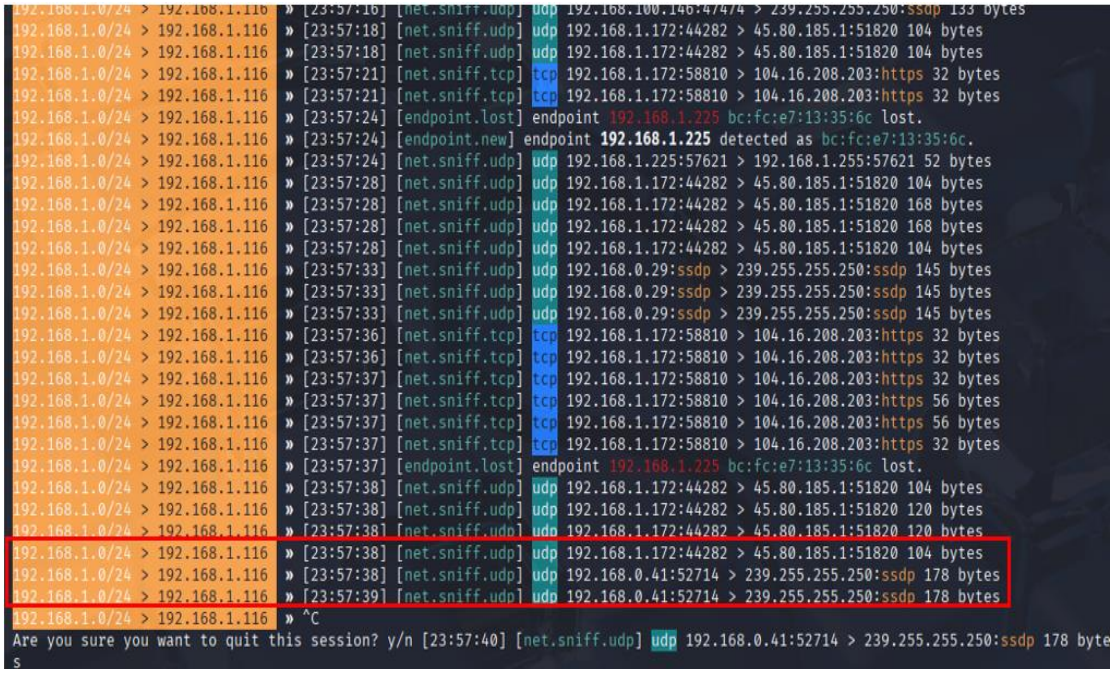

Ilustración 5-49 Tráfico mDNS y SSDP observado en la red local durante la prueba.

Resultados de ataque MITM con tráfico MQTT

En un entorno IoT simulado, el cliente Ubuntu actua como sensor que transmitía datos MQTT hacia un broker local, utilizando el túnel VPN creado por NordVPN.

Durante esta sesión, se configura Bettercap para interceptar el tráfico hacia la IP física del cliente (192.168.1.172) mediante suplantación ARP.

A lo largo de la ejecución, Bettercap detecta únicamente mensajes mDNS correspondientes a consultas automáticas de servicios como _spotify-connect._tcp.local, como se aprecia en la ilustración 5-50. No se registraron paquetes que revelaran publicaciones MQTT ni datos sensibles.

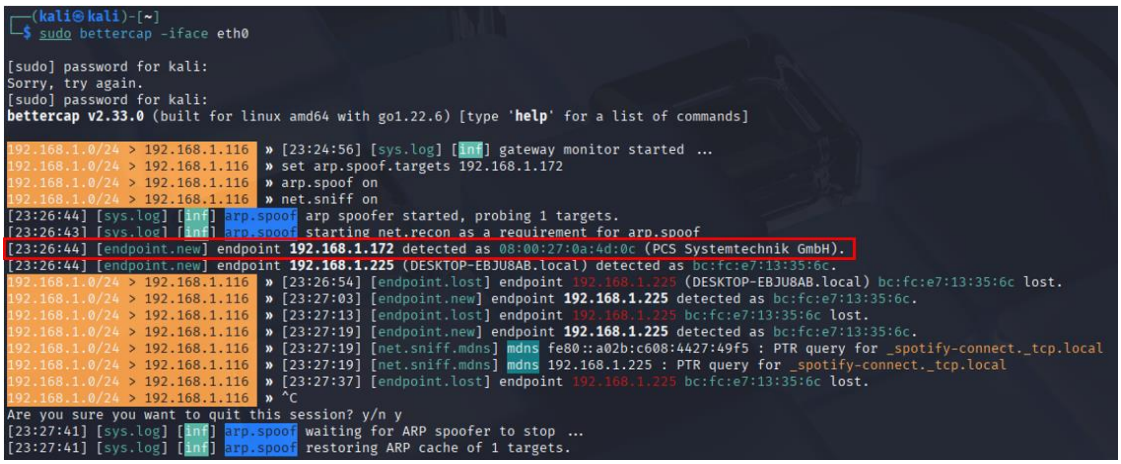

Ilustración 5-50 Resultado del ataque MITM sin datos MQTT capturados

Estos resultados confirman que el tráfico MQTT fue encapsulado adecuadamente dentro del túnel VPN, impidiendo su visualización incluso ante un atacante local.

Observaciones técnicas relevantes

NordVPN, mediante el protocolo NordLynx, demuestra una capacidad de encapsulación robusta. La imposibilidad de capturar tráfico navegacional, peticiones DNS o mensajes MQTT, incluso desde la misma red física, refleja un alto nivel de aislamiento y protección. El rechazo de paquetes ICMP y la ausencia total de fugas DNS refuerzan la eficacia de su configuración predeterminada para proteger la privacidad del usuario.

5.4.5 Comparación de resultados de seguridad en entornos IoT con tráfico MQTT

Con el objetivo de comparar de forma estructurada el comportamiento de cada tecnología VPN evaluada frente a ataques del tipo Man-in-the-Middle (MITM) en un entorno IoT simulado, se resume en la siguiente tabla la información obtenida durante las pruebas prácticas. La comparación considera si el atacante logra

posicionarse entre el cliente y el gateway, qué tipo de tráfico logra capturar y si fue posible visualizar el contenido de las publicaciones MQTT generadas por el dispositivo IoT como indica la tabla 8.

Tabla 8 Comparación de la resistencia a ataques MITM en entorno IoT (MQTT)

Tecnología VPN	Tipo de túnel	IP de publicación MQTT	MITM exitoso?	¿Capturas de mensajes MQTT?	Tipo de tráfico visible
Wireguard	wg0 (punto a punto)	10.10.10.2	No	No	mDNS, SSDP
OpenVPN	tun0 (punto a punto)	10.8.0.2	No	No	mDNS
Windscribe	utun420	100.89.67.101	No	No	mDNS
NordVPN	norlynx	10.5.0.2	No	No	mDNS

En todos los casos evaluados, se observa que el ataque tipo Man-in-the-Middle no logra comprometer la confidencialidad de los mensajes MQTT transmitidos desde el cliente IoT hacia el broker, debido a que el tráfico circula encapsulado dentro de túneles cifrados. Aunque Bettercap logra detectar la presencia del cliente en la red física mediante técnicas de suplantación ARP, no fue posible interceptar mensajes MQTT en texto claro ni extraer datos de interés desde el punto de vista de la seguridad. La única información visible durante las capturas correspondió a tráfico mDNS, generado automáticamente por dispositivos para descubrimiento de servicios locales, lo cual no representa una vulnerabilidad directa en el canal de comunicación protegido por las VPN evaluadas.

5.5 Contraste con investigaciones previas

Los resultados obtenidos en esta investigación concuerdan con lo reportado en estudios previos que analizan la efectividad de las tecnologías VPN para proteger el tráfico en entornos IoT. Según Fereidouni et al. (2025), los ataques MITM

representan una de las amenazas más comunes en sistemas de IoT debido a la escasa autenticación y a la comunicación no cifrada entre dispositivos. Sin embargo, el estudio concluye que el uso de túneles cifrados mediante protocolos robustos, como WireGuard o OpenVPN, mitiga eficazmente este tipo de ataques cuando se implementan correctamente. Este hallazgo se refuerza en el presente trabajo, donde se verificó que ninguna de las VPN evaluadas permitió la interceptación de datos MQTT, incluso en presencia de un atacante con acceso directo a la red local.

Asimismo, en la investigación de Alrawi et al. (2019), se demuestra que muchos dispositivos IoT expuestos a Internet no implementan cifrado de extremo a extremo, lo que facilita ataques MITM. A diferencia de esos escenarios vulnerables, este estudio se enfocó en un entorno donde el canal de comunicación fue protegido mediante VPN, lo cual demostró ser una estrategia efectiva. De forma similar, el análisis de Zolanvari et al. (2019) sobre seguridad IoT en redes heterogéneas identifica a las VPN como uno de los mecanismos más confiables para resguardar la integridad del tráfico en sistemas distribuidos.

Por lo tanto, los resultados obtenidos en este trabajo no solo están alineados con la literatura existente, sino que aportan evidencia empírica específica en el contexto de tráfico MQTT, un protocolo ampliamente utilizado en IoT. La validación de estas tecnologías bajo condiciones reales de ataque contribuye a reforzar su adopción como medidas efectivas de ciberseguridad en redes de dispositivos conectados.

6 Conclusiones

- El presente proyecto de titulación con componentes de investigación aplicada permitió realizar un análisis exhaustivo sobre la seguridad en la transmisión de datos mediante el uso de tecnologías VPN en entornos IoT, tanto en condiciones convencionales como en escenarios simulados de dispositivos conectados que utilizan el protocolo MQTT. A lo largo del estudio se implementaron cuatro soluciones VPN: WireGuard, OpenVPN, Windscribe y NordVPN, evaluando su capacidad de cifrado, su resistencia frente a ataques del tipo Man-in-the-Middle y su comportamiento frente a posibles fugas de información, especialmente en el ámbito de los sistemas IoT.

- Los resultados obtenidos evidencian que todas las tecnologías evaluadas lograron cifrar eficazmente el tráfico, impidiendo el acceso a contenido sensible durante los ataques MITM. No obstante, solo NordVPN y Windscribe evitaron completamente fugas DNS, lo cual representa un factor determinante para garantizar la privacidad en entornos distribuidos. WireGuard y NordVPN se posicionaron como las soluciones más equilibradas en términos de rendimiento y seguridad, presentando baja latencia y alta eficiencia, condiciones ideales para aplicaciones críticas en tiempo real. En contraposición, Windscribe y OpenVPN, si bien más accesibles en términos de uso, mostraron mayores requerimientos técnicos y un desempeño inferior bajo ciertas condiciones.
- Desde una perspectiva práctica, se recomienda el uso de NordVPN en instituciones que requieran una solución profesional, confiable y fácil de implementar, como centros de salud, universidades y entidades gubernamentales. Para entornos técnicos o con restricciones presupuestarias, WireGuard se presenta como una opción óptima, siempre que se cuente con personal capacitado para su implementación. No obstante, en casos donde se privilegie la facilidad de configuración sobre la eficiencia, Windscribe o OpenVPN pueden ser consideradas alternativas viables, con la debida compensación técnica. En cuanto al análisis de costos, las soluciones comerciales como NordVPN y Windscribe ofrecen paquetes accesibles para organizaciones medianas, aunque las tecnologías libres como WireGuard resultan más rentables a largo plazo cuando se dispone de capacidades internas para su mantenimiento.
- Como trabajos futuros, se plantea extender esta evaluación hacia arquitecturas de redes híbridas que combinen VPN con otras tecnologías de seguridad como TLS mutuo o enfoques Zero Trust, así como integrar pruebas con dispositivos IoT reales de bajo consumo energético. Además, se sugiere desarrollar herramientas automatizadas que permitan verificar en tiempo real la presencia de fugas DNS o vulnerabilidades de configuración, promoviendo entornos más seguros y sostenibles para la operación de redes inteligentes.
- En síntesis, la investigación demuestra que la implementación de tecnologías VPN es una estrategia eficaz para garantizar la seguridad y privacidad en redes IoT. No obstante, su éxito depende tanto de la correcta elección de la tecnología

como del contexto de uso, el nivel de conocimiento técnico disponible y las necesidades específicas de cada organización.

Referencias

Conti, M., Dehghantanha, A., Franke, K., & Watson, S. (2021). Internet of Things security and forensics: Challenges and opportunities. *Future Generation Computer Systems, 78*, 544-546. Obtenido de https://doi.org/10.1016/j.future.2017.07.060

Donenfeld, J. (2019). WireGuard: Fast, modern, secure VPN tunnel. En *Proceedings of the 26th USENIX Security Symposium*. Obtenido de https://www.usenix.org/conference/usenixsecurity17/presentation/donenfeld

Kim, H., Kim, J., & Park, C. Y. (2021). VPNs for IoT security: A performance and effectiveness study. *IEEE Access, 9*, 30145-30160. Obtenido de https://doi.org/10.1109/ACCESS.2021.3057921

Mehmood, Y., Ahmad, F., Yaqoob, I., Adnane, A., Imran, M., & Guizani, M. (2020). Secure IoT: A survey on security and privacy of IoT. *IEEE Communications Surveys & Tutorials, 22*(4), 1988-2010. Obtenido de https://doi.org/10.1109/COMST.2020.3011200

Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems, 29*(7), 1645-1660. https://doi.org/10.1016/j.future.2013.01.010

Al-Omari, S., & Al-Qutayri, M. (2020). Security Challenges in the Internet of Things: A Survey. *IEEE Internet of Things Journal, 7*(4), 4184-4205. https://doi.org/10.1109/JIOT.2020.2967486

Almusaylim, Z. A., & Jhanjhi, N. Z. (2019). Comprehensive review: Privacy protection of user in IoT environment by using VPN. *IEEE Access, 7*, 66496-66509. https://doi.org/10.1109/ACCESS.2019.2916558

Attaran, M. (2020). The Internet of Things: Limitations, Security Issues and Solutions. *Internet of Things, 5*, 100129. https://doi.org/10.1016/j.iot.2019.100129

Babar, S., Mahalle, P., Stango, A., Prasad, N., & Prasad, R. (2011). Proposed security model and threat taxonomy for the Internet of Things (IoT). *International Conference on Network Security and Applications*. Springer, 420-429. https://doi.org/10.1007/978-3-642-22540-6_42

Barcelo-Ordinas, J. M., Chanet, J. P., Hou, K. M., & Garcia-Vidal, J. (2012). A survey of wireless sensor technologies applied to precision agriculture. *Precision Agriculture, 13*(6), 801-816. https://doi.org/10.1007/s11119-012-9278-1

Chen, L., Zhou, Y., & Li, J. (2020). A Survey on Privacy Protection in Internet of Things: Attacks, Countermeasures, and Opportunities. *IEEE Access, 8*, 150981-151006. https://doi.org/10.1109/ACCESS.2020.3016564

Das, A. K., Islam, S. H., & Wazid, M. (2017). A provably secure and efficient authentication scheme for IoT-enabled devices in distributed cloud computing environment. *IEEE Transactions on Dependable and Secure Computing, 15*(5), 840-851. https://doi.org/10.1109/TDSC.2016.2625250

Farooq, M. U., Waseem, M., Khairi, A., & Mazhar, S. (2015). A critical analysis on the security concerns of Internet of Things (IoT). *International Journal of Computer Applications, 111*(7), 1-6. https://doi.org/10.5120/19547-1280

Farris, I., Girau, R., Atzori, L., Pau, G., & Marques, P. (2018). The Rise of IoT Interoperability: An Ongoing Journey. *IEEE Internet Computing, 22*(3), 34-43. https://doi.org/10.1109/MIC.2018.032591615

Fortino, G., & Trunfio, P. (Eds.). (2014). *Internet of Things Based on Smart Objects: Technology, Middleware and Applications*. Springer. https://doi.org/10.1007/978-3-319-00491-4

Garcia-Morchon, O., Kumar, S. S., Keoh, S. L., Hummen, R., & Struik, R. (2013). Security Considerations in the IP-Based Internet of Things. *IEEE Wireless Communications, 20*(3), 76-84. https://doi.org/10.1109/MWC.2013.6549280

Gupta, H., Vahidnia, R., & Goudarzi, M. (2020). Security and Privacy in Internet of Things: Challenges, Solutions, and Future Directions. *Future Generation Computer Systems, 102*, 771-774. https://doi.org/10.1016/j.future.2019.07.019

Hellaoui, H., Benazzouz, Y., & Bouhorma, M. (2017). A lightweight authentication scheme for E-health applications in the context of IoT. *IEEE Journal of Biomedical and Health Informatics, 21*(6), 1662-1673. https://doi.org/10.1109/JBHI.2016.2644610

Khan, M. A., & Salah, K. (2018). IoT security: Review, blockchain solutions, and open challenges. *Future Generation Computer Systems, 82*, 395-411. https://doi.org/10.1016/j.future.2017.11.022

Kumar, P., Lee, H.-J., & Lee, H.-J. (2016). Security Issues in Healthcare Applications Using Wireless Medical Sensor Networks: A Survey. *Sensors, 12*(1), 55-91. https://doi.org/10.3390/s120100055

Li, X., & Wang, X. (2019). Survey on the Security of the Internet of Things. *Security and Communication Networks, 2019*. https://doi.org/10.1155/2019/2348328

Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H., & Zhao, W. (2017). A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and

Applications. *IEEE Internet of Things Journal, 4*(5), 1125-1142. https://doi.org/10.1109/JIOT.2017.2683200

Lu, Y., & Da Xu, L. (2019). Internet of Things (IoT) Cybersecurity Research: A Review of Current Research Topics. *IEEE Internet of Things Journal, 6*(2), 2103-2115. https://doi.org/10.1109/JIOT.2018.2874346

Ning, H., & Hu, S. (2012). Internet of Things: An Emerging Industrial or Strategic Tool. *IEEE Internet Computing, 16*(4), 66-70. https://doi.org/10.1109/MIC.2012.71

Roman, R., Najera, P., & Lopez, J. (2011). Securing the Internet of Things. *Computer, 44*(9), 51-58. https://doi.org/10.1109/MC.2011.291

Sicari, S., Rizzardi, A., Grieco, L. A., & Coen-Porisini, A. (2015). Security, privacy and trust in Internet of Things: The road ahead. *Computer Networks, 76*, 146-164. https://doi.org/10.1016/j.comnet.2014.11.008

Suo, H., Wan, J., Zou, C., & Liu, J. (2012). Security in the Internet of Things: A Review. *International Conference on Computer Science and Electronics Engineering (ICCSEE), 3*, 648-651. https://doi.org/10.1109/ICCSEE.2012.373

Yang, Y., Wu, L., Yin, G., Li, L., & Zhao, H. (2017). A Survey on Security and Privacy Issues in Internet-of-Things. *IEEE Internet of Things Journal, 4*(5), 1250-1258. https://doi.org/10.1109/JIOT.2017.2694844

Zhang, Y., Liu, L., & Vasilakos, A. V. (2014). A survey on trust management for Internet of Things. *Journal of Network and Computer Applications, 42*, 120-134. https://doi.org/10.1016/j.jnca.2014.01.014

Zhou, W., Zhang, Y., & Wang, P. (2018). The Internet of Things in the Cloud: A Survey on Cloud Computing and Its Applications. *IEEE Network, 32*(3), 146-152. https://doi.org/10.1109/MNET.2018.1700202

Avast. (2024). *VPN Protocols: A Comprehensive Guide*. Recuperado de https://www.avast.com/es-es/c-vpn-protocols

Cloudflare. (2024). *VPN Speed and Latency: How to Improve Performance*. Recuperado de https://www.cloudflare.com/es-es/learning/access-management/vpn-speed/

Donenfeld, J. (2019). *WireGuard: Next Generation Kernel Network Tunnel.* Retrieved from https://www.wireguard.com

Saxena, N., & Grizonic, M. (2021). *Performance Analysis of WireGuard VPN Protocol.* IEEE Access, 9, 49338-49349.

OpenVPN Technologies Inc. (2022). *OpenVPN Security Overview.* Retrieved from https://openvpn.net

National Institute of Standards and Technology (NIST). (2001). *FIPS PUB 197: Advanced Encryption Standard (AES).*

Zhu, T., Zhang, J., & Zhou, S. (2020). *Security and Performance Evaluation of OpenVPN on Resource-Constrained Devices.* Sensors, 20(6), 1705.

Windscribe VPN. (2022). *Encryption Standards.* Retrieved from https://windscribe.com

Gupta, R., & Rawat, D. (2020). *An Empirical Study of VPN Protocols for Secure Internet Communication.* IEEE Communications Surveys & Tutorials, 22(3).

NordVPN. (2021). *What is NordLynx and How Does it Work?* Retrieved from https://nordvpn.com/features/nordlynx

Szefer, J. (2021). *Security Aspects of WireGuard and NordLynx VPNs.* Journal of Cybersecurity and Privacy, 1(1), 49-63.

Fereidouni, A., Jabbarpour, M. R., Karimipour, H., & Dehghantanha, A. (2025). *IoT and Man-in-the-Middle Attacks: A Survey on Mitigation Techniques*. ResearchGate. https://www.researchgate.net/publication/389753459_IoT_and_Man-in-the-Middle_Attacks

Alrawi, O., Lever, C., Antonakakis, M., & Monrose, F. (2019). SoK: Security evaluation of home-based IoT deployments. *2019 IEEE Symposium on Security and Privacy (SP)*, 1362–1380. https://doi.org/10.1109/SP.2019.00039

Zolanvari, M., Teixeira, M. A., Jain, R., Kazemian, H., & Al-Fuqaha, A. (2019). IoT security in smart grid: A review of technologies, challenges, and future directions. *IEEE Access, 7*, 164996–165020. https://doi.org/10.1109/ACCESS.2019.2952944

ANEXOS

ANEXO A CONFIGURACIÓN DE VPNS

Instalación y configuración de WireGuard

```
# Instalación
sudo apt update
sudo apt install wireguard
# Generar llaves
wg genkey | tee privatekey | wg pubkey > publickey
# Configuración del servidor
sudo nano /etc/wireguard/wg0.conf
[Interface]
Address = 10.0.0.1/24
PrivateKey = <clave_privada_servidor>
ListenPort = 51820
[Peer]
PublicKey = <clave_publica_cliente>
AllowedIPs = 10.0.0.2/32
# Activar interfaz
sudo wg-quick up wg0
```

Instalación y configuración de OpenVPN (Servidor en Kali, Cliente en Ubuntu)

```
# Instalación en servidor (Kali)
sudo apt update
sudo apt install openvpn easy-rsa
make-cadir ~/openvpn-ca
cd ~/openvpn-ca
./easyrsa init-pki
./easyrsa build-ca
# Generación de claves y certificados
./easyrsa gen-req server nopass
./easyrsa sign-req server server
./easyrsa gen-req client nopass
```

./easyrsa sign-req client client

Servidor

sudo nano /etc/openvpn/server.conf

port 1194

proto udp

dev tun

ca ca.crt

cert server.crt

key server.key

dh dh.pem

server 10.8.0.0 255.255.255.0

push "redirect-gateway def1"

keepalive 10 120

cipher AES-256-CBC

Iniciar servicio

sudo systemctl start openvpn@server

Archivo de configuración cliente

sudo nano client.ovpn

Instalación de Windscribe (modo CLI)

sudo apt install windscribe-cli

sudo windscribe login

sudo windscribe connect

Instalación de NordVPN (CLI oficial)

sh <(curl -sSf https://downloads.nordcdn.com/apps/linux/install.sh)

nordvpn login

nordvpn connect

Anexo B Pruebas de cifrado de tráfico

```
# En Kali (ataque) para verificar tráfico
sudo tcpdump -i eth0 host <IP_cliente>
# Escuchar tráfico entrante/saliente en Kali
sudo tcpdump -i eth0 -n
# Filtrar solo tráfico HTTP
sudo tcpdump -i eth0 tcp port 80
# Filtrar solo tráfico HTTPS
sudo tcpdump -i eth0 tcp port 443
# Comprobar si se filtra contenido HTTP desde cliente
curl http://example.com
# Ver tráfico en WireGuard (interfaz wg0 en cliente)
sudo tcpdump -i wg0
# Mostrar estadísticas del túnel WireGuard
sudo wg show
```

Anexo C Ataque MITM con Bettercap

```
# En Kali (instalación)
sudo apt install bettercap
# Iniciar Bettercap
sudo bettercap -iface eth0
# Dentro de Bettercap
set arp.spoof.targets <IP_cliente>
arp.spoof on
net.sniff on
```

Anexo D Prueba de fuga DNS

```
# Abrir navegador en el cliente y visitar
https://dnsleaktest.com
https://browserleaks.com/dns
# Prueba de fuga DNS
# Consultar servidores DNS configurados
```

```
cat /etc/resolv.conf
# Resolver sin caché (modo trace)
dig +trace openai.com
# Consultar usando servidor DNS específico
dig @1.1.1.1 openai.com
# Captura de tráfico DNS con filtro
sudo tcpdump -i eth0 udp port 53
# o iniciar navegador y revisar servidores detectados
curl https://dnsleaktest.com
```

Anexo E Latencia y velocidad (medición)

```
# Ping de latencia
ping <IP_destino> -c 10
# Prueba de velocidad entre dispositivos
iperf3 -s      # en servidor
iperf3 -c <IP_servidor> -t 30    # en cliente
# Medición de latencia y velocidad
# Medición básica de latencia
ping <IP_destino> -c 10
# Medición de jitter con intervalo corto
ping -i 0.2 <IP_destino>
# Medición extendida de latencia y pérdida
sudo apt install mtr
mtr <IP_destino>
# Pruebas de velocidad con iperf3
iperf3 -s                  # en el servidor
iperf3 -c <IP_servidor> -t 30  # en el cliente
# Pruebas de velocidad con netperf
sudo apt install netperf
netserver                  # en el servidor
netperf -H <IP_servidor>      # en el cliente
# Descarga de archivo de prueba
```

wget https://speed.hetzner.de/100MB.bin

Anexo F Prueba con tráfico MQTT

```
#Publicador
for i in {1..5}; do
  mosquitto_pub -h <IP_broker> -t "test/topic" -m "mensaje $i"
#Suscriptor
done
mosquitto_sub -h <IP_broker> -t "test/topic"
```