Universidad Nacional Autónoma de México

Facultad de Ciencias

...

T E S I S

QUE PARA OBTENER EL GRADO DE:

Licenciatura en Ciencias de la Computación

P R E S E N T A :

Pérez Romero Natalia Abigail

TUTOR

Dr. José David Flores Peñaloza

Ciudad Universitaria, Cd. Mx., 2023

Capítulo 1

title

Índice general

Capítulo 2

Introducción

En la actualidad, y desde hace tiempo, las empresas al tener una presencia global con múltiples sucursales y empleados remotos requieren establecer conexiones seguras entre sus dispositivos. Para esto hay dos opciones: una línea de alquiler privada y dedicada o compartir una parte del ancho de banda con una línea existente, como Internet. La segunda opción es más económica y flexible, pero menos segura. De ahí surgen las redes privadas virtuales (VPN) que permiten establecer un túnel seguro dentro de una red pública tal como si estuvieran conectados en una red local. Adicionalmente, usuarios finales han encontrado en las VPN una forma de proteger su información y privacidad, además de acceder a contenido restringido geográficamente.

Diferentes protocolos de VPN han sido desarrollados para responder a estos requerimientos, cada uno con sus propias características y funcionalidades, entre ellos OpenVPN y IPsec. Sin embargo, estos protocolos presentan problemas de seguridad, complejidad y rendimiento. Por ejemplo, OpenVPN es un protocolo de VPN de código abierto que utiliza el protocolo SSL/TLS para cifrar el tráfico de red, pero es lento y complejo de configurar. Por otro lado, IPsec es un protocolo de VPN que utiliza el protocolo IKEv2 para establecer un túnel seguro, pero es difícil de configurar y no es compatible con todos los dispositivos.

En respuesta a estos problemas, se ha desarrollado un protocolo de VPN llamado WireGuard que es más simple, más rápido y más seguro que otros protocolos de VPN. WireGuard es más simple que otros protocolos de VPN porque utiliza un enfoque basado en claves públicas para establecer un túnel seguro, en lugar de utilizar certificados SSL/TLS como OpenVPN. WireGuard es más rápido que otros protocolos de VPN porque utiliza un enfoque basado en el kernel para cifrar y descifrar el tráfico de red, en lugar de utilizar un enfoque basado en el usuario como OpenVPN. WireGuard es más seguro que otros protocolos de VPN porque utiliza un enfoque basado en claves públicas para establecer un túnel seguro, en lugar de utilizar un enfoque basado en contraseñas como IPsec.

Si bien WireGuard es un protocolo de VPN simple, la mayor de sus desventajas es la configuración de los pares dentro de la red. Por ejemplo, si se desea configurar una red privada virtual con 10 pares, se debe configurar manualmente cada par con la dirección IP y la clave pública de cada par. Esto puede ser un proceso tedioso y propenso a errores, especialmente si se desea configurar una red privada virtual con

un gran número de pares.

Una solución para esta dificultad es propuesa por Tailscale, el cual es un servicio VPN que hace que sus dispositivos y aplicaciones sean accesibles en cualquier parte del mundo, de forma segura y sin esfuerzo. Este software actúa en combinación con el kernel para establecer una comunicación VPN peer-to-peer o retransmitida con otros clientes utilizando el protocolo WireGuard. Tailscale puede abrir una conexión directa con el peer utilizando técnicas de NAT traversal como STUN o solicitar el reenvío de puertos a través de UPnP IGD, NAT-PMP o PCP. [7] Si el software no consigue establecer una comunicación directa, recurre al protocolo DERP (Designated Encrypted Relay for Packets) proporcionados por la empresa. [6] Las direcciones IPv4 asignadas a los clientes se encuentran en el espacio reservado NAT de nivel operador. Esto se eligió para evitar interferencias con las redes existentes[9] ya que el enrutamiento de tráfico a redes detrás del cliente es posible.

Tailscale es una gran solución para la configuración de pares en WireGuard, pero no es una solución de código abierto. Al ofrecer más funcionalidades que las necesarias para la configuración de pares en WireGuard, Tailscale puede ser una solución costosa para empresas pequeñas o individuos que solo requieren configurar pares en WireGuard. Además, Tailscale no permite a los usuarios tener control total sobre la configuración de pares en WireGuard, ya que la configuración de pares en WireGuard se realiza a través de la interfaz de usuario de Tailscale y no a través de la línea de comandos. Y finalmente, Tailscale aumenta la superficie de ataque de la red privada virtual, ya que no solicita la autenticación al usar un servicio crítico como SSH entre los pares.

Existe una alternativa de código abierto del servidor de control de Tailscale llamado Headscale. El objetivo de Headscale es proporcionar a un servidor de código abierto que puedan utilizar para proyectos y laboratorios. Implementa un alcance estrecho, una sola Tailnet, adecuada para un uso personal o una pequeña organización de código abierto. [10]

Aunque Headscale sea una opción open-source no cuenta con un cliente en el dispositivo final y únicamente permite crear un red privada.

Por lo que en este trabajo se propone el desarrollo de un prototipo open-source de un sistema de control de configuración de pares minimalista usando el protocolo WireGuard, inspirado en Tailscale, el cual se adherirá a los principios de UNIX.

Este prototipo de sistema permitirá automatizar la configuración de pares, mediante:

- **Cliente**: Programa que se ejecutará en el dispositivo final, el cual consiste de:
  - *Cliente daemon*: Un servidor XML-RPC que actuará como daemon guardando, actualizando y manteniendo la configuración actual de los pares y el cliente.
  - *Interfaz de cliente*: Un programa que actuará como interfaz recibiendo instrucciones por CLI e interactuando con el cliente daemon.
- **Servidor Orquestador (LinkGuard)**: Un servidor XML-RPC que se encargará de orquestar y automatizar la configuración de los pares dentro de las redes

privadas.

Se espera que este prototipo automatice la configuración de los pares en una VPN WireGuard, permitiendo que el servidor orquestador actúe como directorio conociendo la dirección IP del endpoint, puertos, llaves públicas, lista de IPs permitidas por cada par en cada red privada. También permitirá la comunicación entre dispositivos finales que no pueden comunicarse directamente, ya que actuará como intermediario en la comunicación.

Se espera que este prototipo reduzca la superficie de ataque de la red privada virtual al no realizar más funcionalidad que la orquestación de los pares en WireGuard.

Para evaluar el prototipo se propondrán tres escenarios con dos dispositivos finales:

- **Todos los dispositivos finales son alcanzables**: Es decir, tanto el orquestador como los dispositivos finales pueden comunicarse entre sí porque están en la misma red o cuentan con IPs públicas. No existen restricciones como firewalls o NATs.
- **Uno de los dispositivos es alcanzable**: En este escenario, el orquestador y los dispositivos final pueden comunicarse entre sí, pero uno dispositivo final no tiene una IP pública o está detrás de un NAT. De forma que el orquestador actuará como intermediario en la comunicación.
- **Solo el Orquestador es alcanzable**: Los dispositivos finales no pueden comunicarse entre sí directamente, pero pueden comunicarse con el orquestador, que actuará como intermediario en la comunicación.

Capítulo 3

Desarrollo

El objetivo principal es desarrollar un prototipo open-source de un sistema de control de configuración de pares minimalista usando el protocolo WireGuard, inspirado en Tailscale y adherido a los principios de UNIX de modularidad, claridad, composición, separación, simplicidad, transparencia, solidez, representación, menor sorpresa, silencio y reparación. Este sistema tiene la finalidad de automatizar la configuración y sincronización de pares en una VPN WireGuard.

Permitirá a los usuarios automatizar la configuración de los pares de una VPN WireGuard mediante un servidor orquestador y un cliente que se ejecutará en el dispositivo final. De forma que el cliente en el dispositivo final recibirá una vez la configuración por CLI de los pares y la mantendrá actualizada gracias a que enviará esta información al servidor orquestador que actuará como directorio conociendo la dirección IP del endpoint, puertos, llaves públicas, lista de IPs permitidas por cada par en cada red privada. Además, el servidor orquestador permitirá la comunicación entre dispositivos finales que no pueden comunicarse directamente, ya que actuará como intermediario en la comunicación.

Finalmente en este trabajo se validara la funcionalidad y robustez del prototipo mediante pruebas en diferentes escenarios, evaluando su impacto en la simplificación de la configuración y gestión de redes VPN WireGuard.

3.1. Casos de uso

Registro de usuario

En el registro del usuario se deberá solicitar al usuario su nombre, correo electrónico y contraseña, que será enviada al orquestrador en texto plano para su registro, si es exitoso el orquestrador deberá enviar un mensaje de confirmación al usuario.

3.1.1. Identificación del usuario

En esta primera version no nos preocuparemos de que la información del usuario se transmita de forma segura, por lo que el usuario deberá identificarse con un nombre

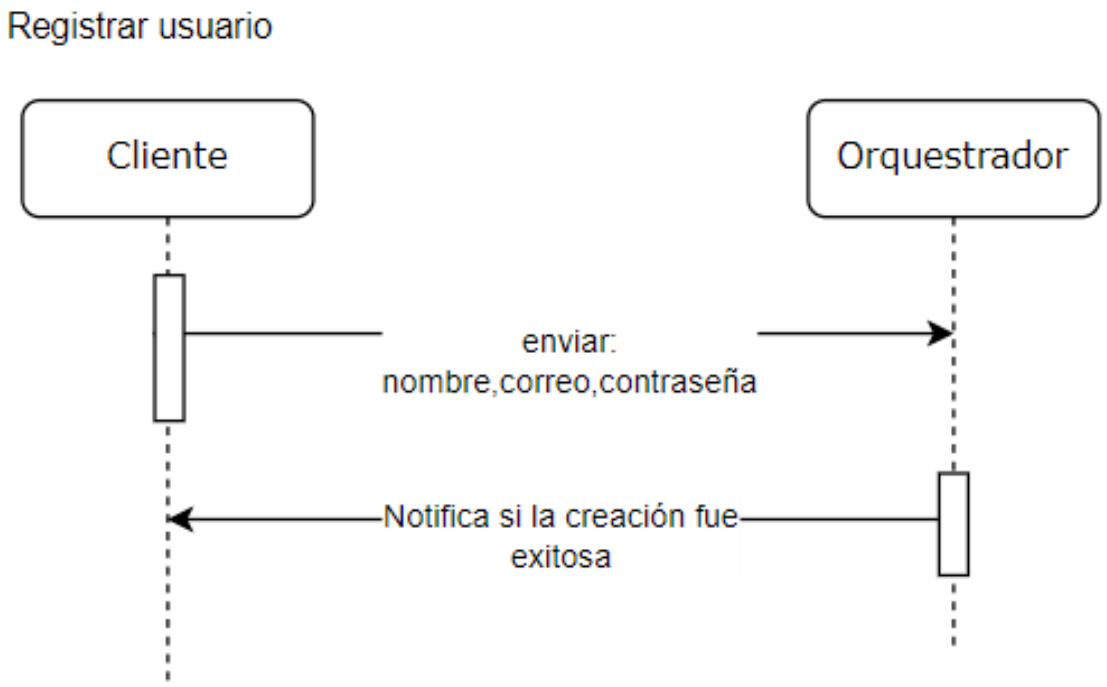

Figura 3.1: Diagrama de caso de uso: registro de usuario

de usuario, correo electrónico y contraseña. Que será enviada al orquestrador en texto plano para su verificación.

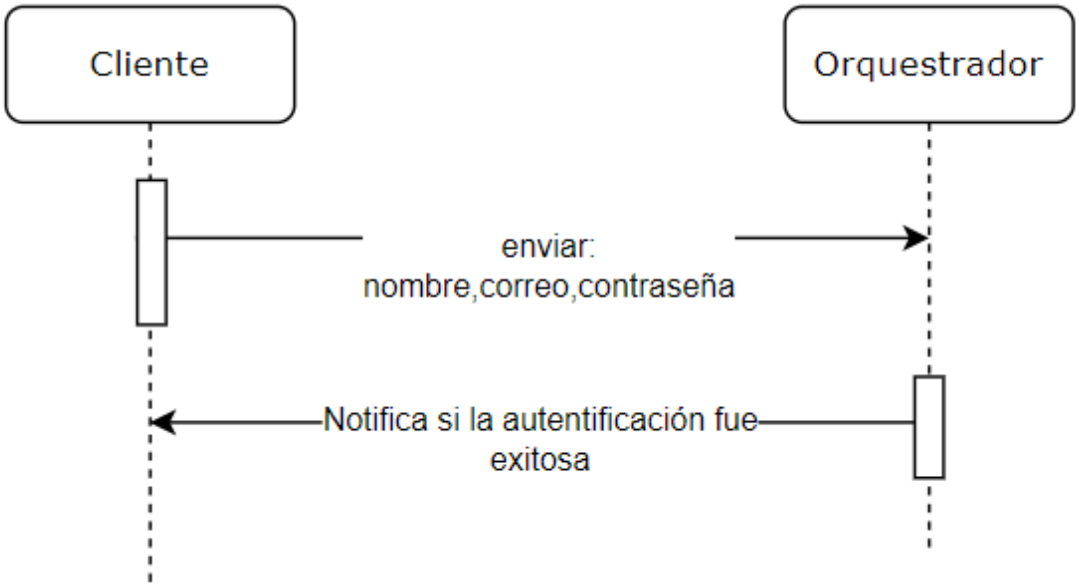

Figura 3.2: Diagrama de caso de uso: identificación del usuario

3.1.2. Registro de una red privada

Deseamos que el usuario pueda registrar las redes privadas a las que desea conectarse, para ello se deberá implementar un mecanismo de registro de redes privadas. El cliente deberá enviar un mensaje al orquestrador con el nombre de la red privada que desea crear, si la red privada existe el orquestrador deberá notificar al cliente, si la red privada no existe, el orquestrador deberá crearla y enviar un mensaje de confirmación con la información siguiente: IP asignada, rango, mascara de la red privada creada.

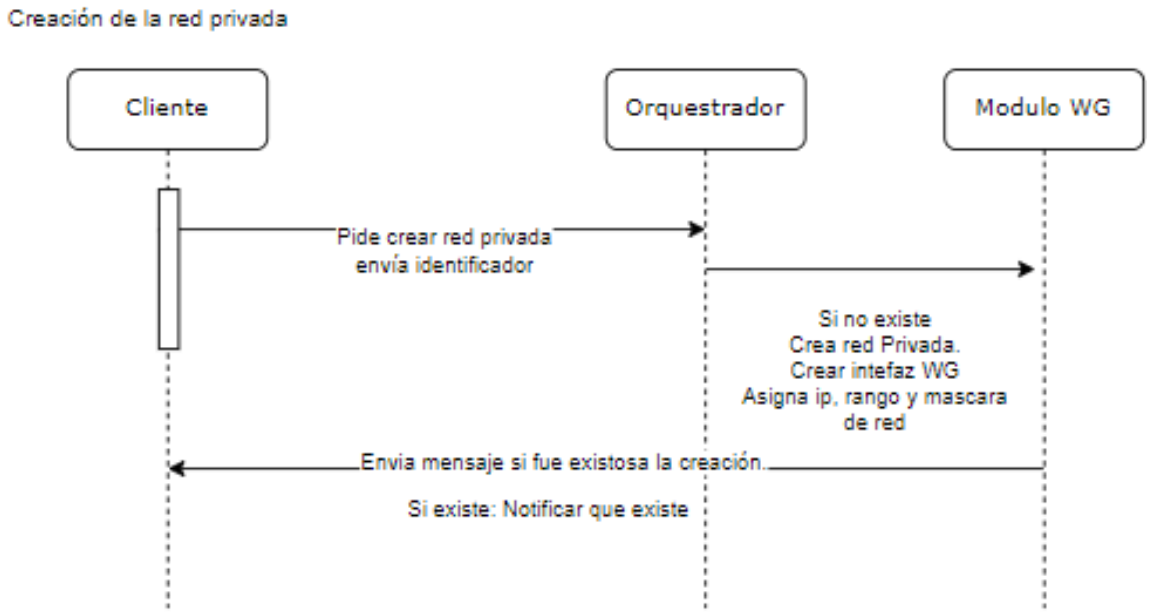

Figura 3.3: Diagrama de caso de uso: registro de red privada

Una red privada desde el punto de vista del orquestrador es un objeto que contiene la siguiente información:

- Identificador
- Nombre de la red privada
- Lista de dispositivos finales
- Lista de conexiones

La idea es crear este objeto para almacenar y consultar datos sobre sobre las redes privadas de un cliente. Luego el razonamiento tras crear un interfaz virtual de Wireguard es que en caso de que la comunicación entre los dispositivos finales no sea posible, el orquestrador pueda actuar como relay dentro de la red privada virtual.

3.1.3. Registro de un dispositivo final

Uno de los objetivos del orquestrador será registrar los dispositivos finales conectados a la red privada, para ello se deberá implementar un mecanismo de registro de dispositivos.

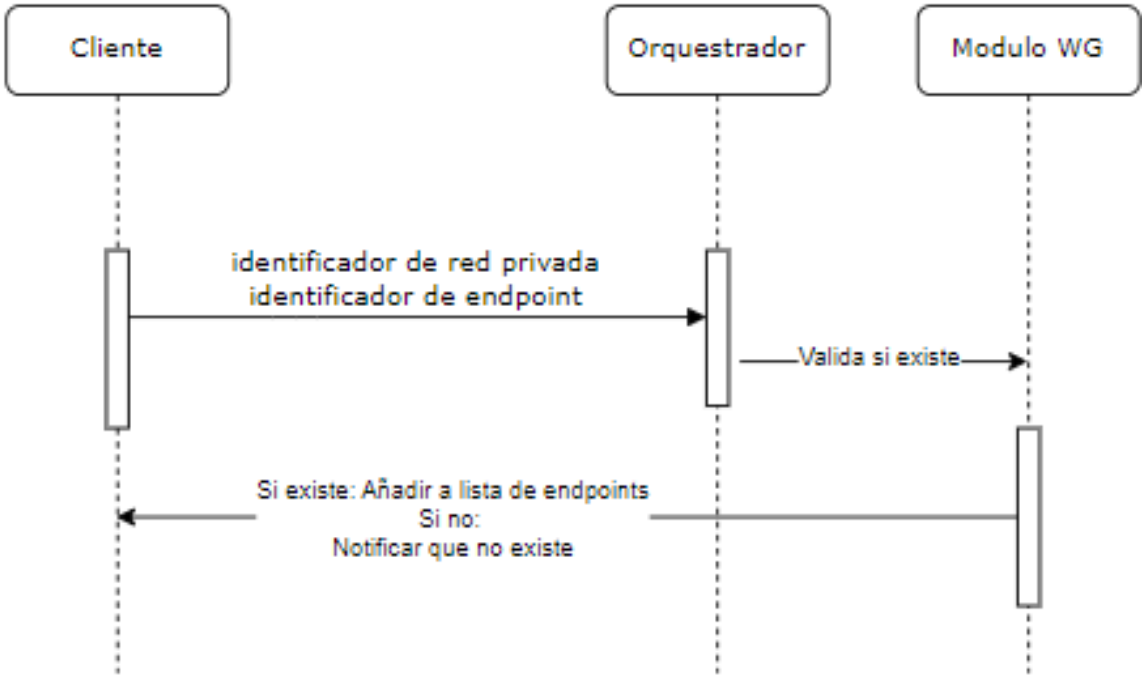

Figura 3.4: Diagrama de caso de uso: registro de dispositivo final

El cliente enviara al orquestrador el identificador de la red privada y el dispositivo final que desea registrar. El orquestrador deberá validar que la red privada exista y que el dispositivo final no esté registrado en la red privada, si es así el orquestrador deberá registrar el dispositivo final y enviar un mensaje de confirmación al cliente.

El orquestrador sera quien asigne la IP de los dispositivos finales dentro de una red privada de los clientes. Así que tambien deberá asignar el rango y la mascara de la red privada.

3.1.4. Cliente verifica conectividad con sus endpoint registrados

El cliente deberá informar al orquestrador si es posible que se comunique con los demás dispositivos finales de la red privada, para ello deberá enviar un mensaje al orquestrador para conocer que dispositivos finales supone que están conectados a la red privada. Luego este cliente verificara si alcanza a los demás dispositivos finales de la red, mediante un mensaje enviado desde la interfaz Wireguard, es decir, usando las IP asignadas por el orquestrador.

Este caso de uso se deberá hacer con cierta periodicidad, para que el cliente pueda tener información actualizada de la red privada. Y en caso de que el orquestrador la solicite el cliente tendrá la información de la red privada actualizada.

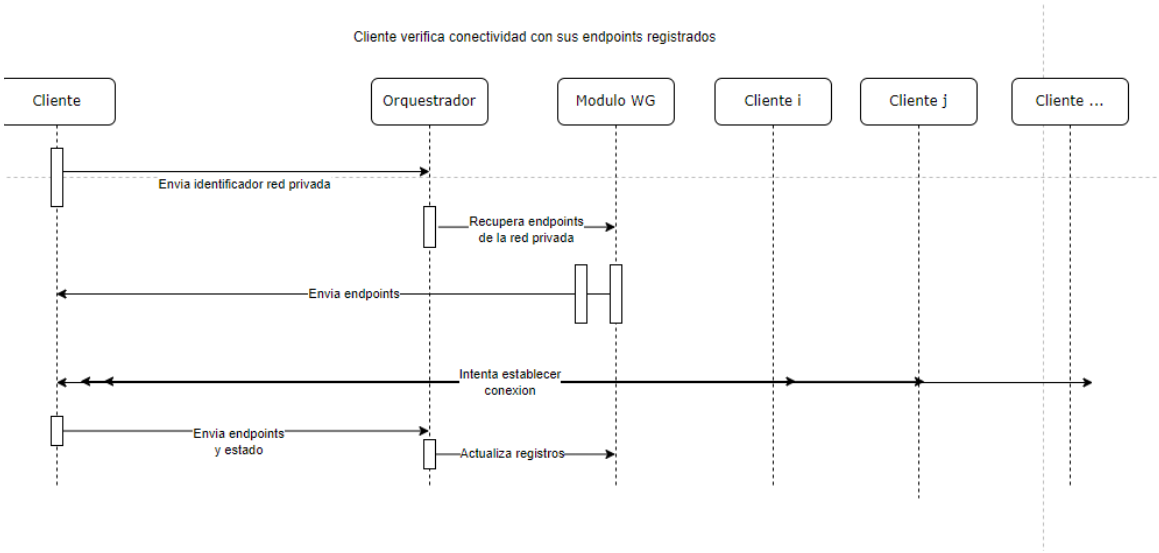

Figura 3.5: Diagrama de caso de uso: cliente verifica conectividad con sus endpoint registrados

3.1.5. Cliente consulta información de red privada al orquestrador

El cliente deberá poder consultar la información de la red privada a la que está conectado, para ello deberá enviar un mensaje al orquestrador con el identificador de la red privada, el orquestrador deberá responder con la información de la red privada.

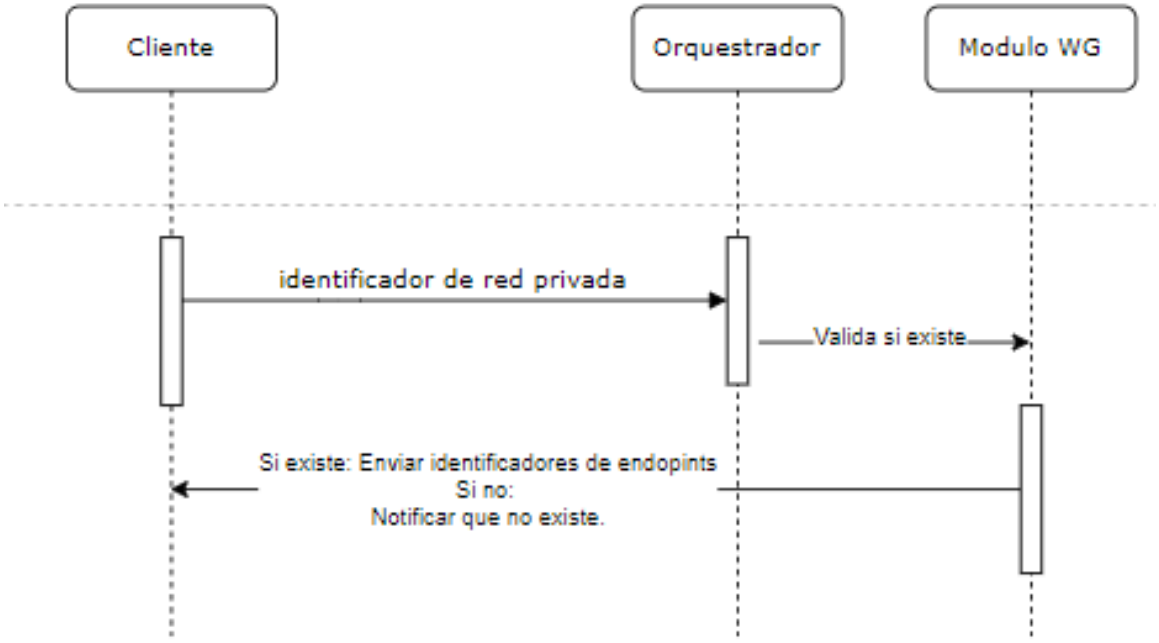

Figura 3.6: Diagrama de caso de uso: cliente consulta información de red privada al orquestrador

3.1.6. Cliente consulta redes privadas disponibles

El cliente deberá poder consultar las redes privadas disponibles, para ello deberá enviar un mensaje al orquestrador, el orquestrador deberá responder con la lista de redes privadas conocidas.

3.1.7. Orquestrador divulga tablero de red privada

Si el cliente envia una solicitud del estado de una red privada al orquestrador, este deberá responder con un tablero de la red privada, que contiene la información de los dispositivos finales conectados a la red privada, las conexiones entre los dispositivos finales y la alcanzabilidad de los dispositivos finales desde el punto de vista del orquestrador.

Esto se deberá hacer con cierta periodicidad, para que el cliente pueda tener información actualizada de la red privada.

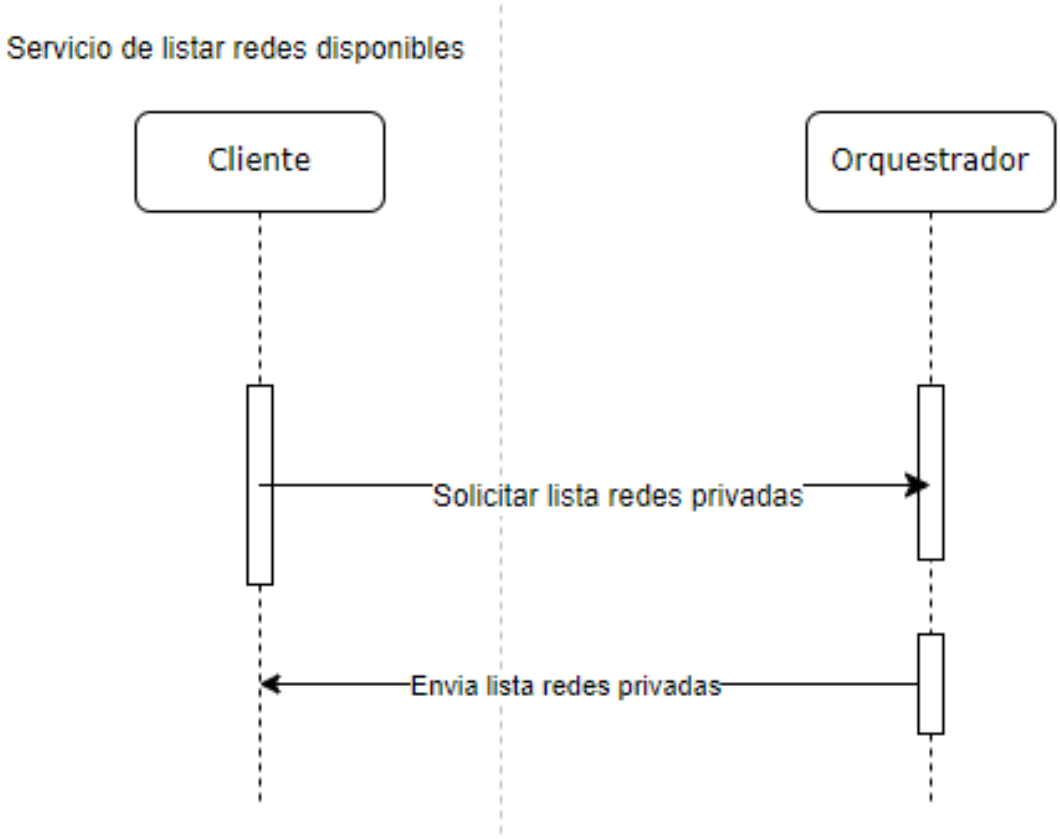

Figura 3.7: Diagrama de caso de uso: cliente consulta redes privadas disponibles

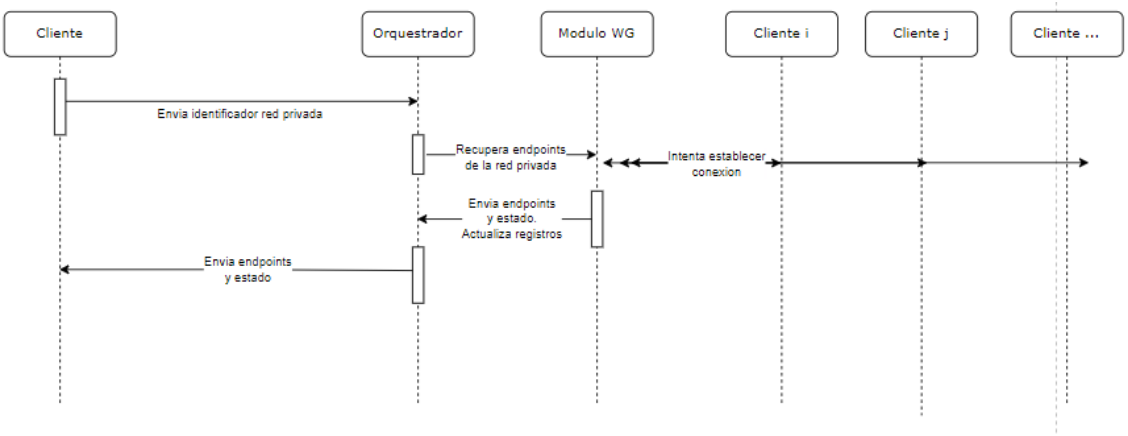

Figura 3.8: Diagrama de caso de uso: orquestrador divulga tablero de red privada

3.1.8. Conexión de dispositivos finales

Tendremos dos casos en cuando se quieran conectar dos o más dispositivos finales, el primero es cuando es posible que se comuniquen entre si por que tienen direcciones IP ruteables. El segundo caso es cuando los dispositivos finales no pueden comunicarse entre si por que no tienen direcciones IP ruteables, en este caso el orquestrador deberá

ofrecer un mecanismo para que los dispositivos finales puedan conectarse entre sí mediante un relay.

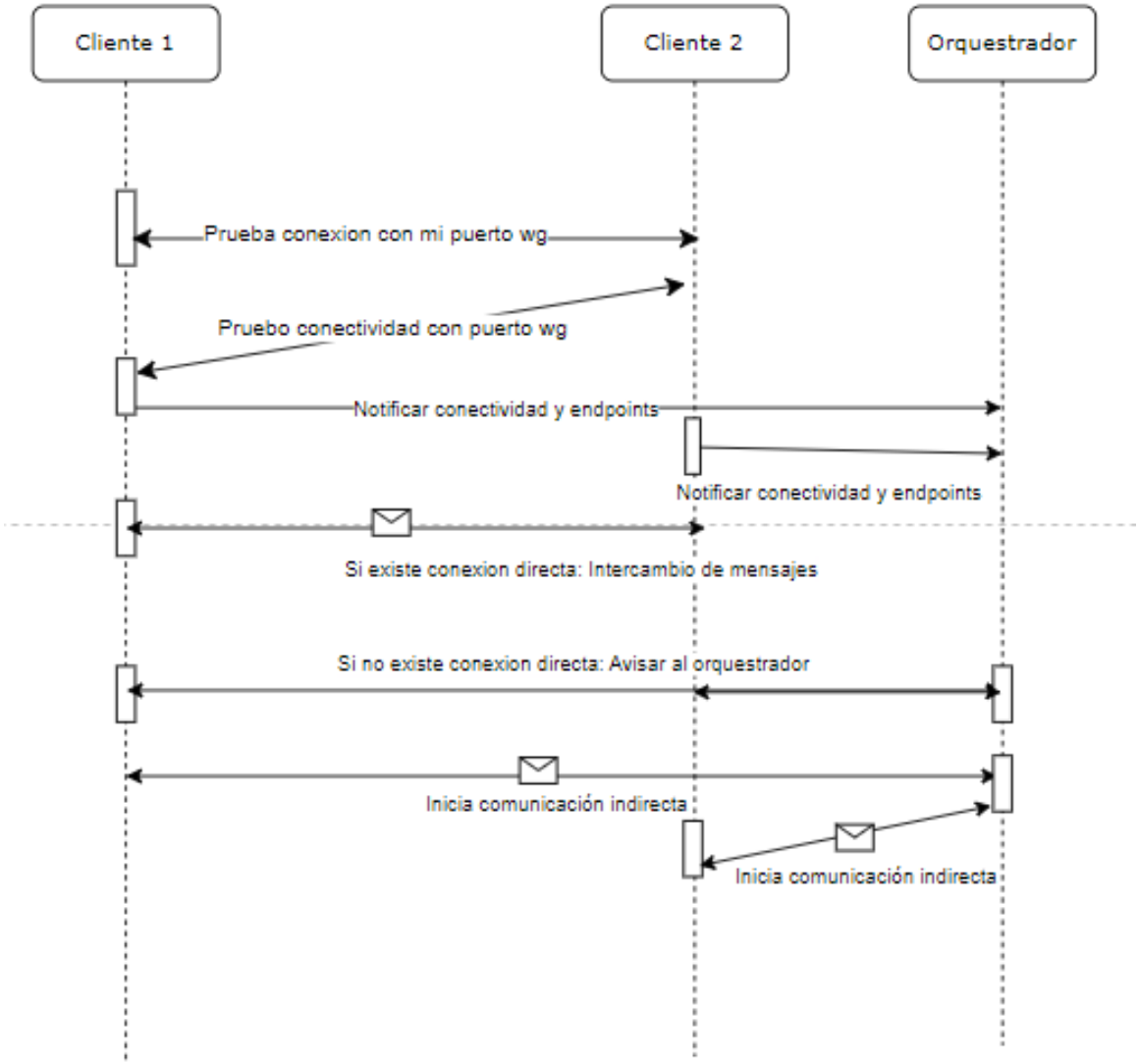

Figura 3.9: Diagrama de caso de uso: conexión de dispositivos finales

Conexión de dispositivos finales con direcciones IP ruteables. Directa

En este caso el uno de los clientes se comunica directamente con el otro cliente, el orquestrador deberá enviar un mensaje de confirmación al cliente que solicita la conexión. Para esto el cliente A enviara mediante ping al cliente B a la dirección IP y puerto que el orquestrador le proporcionó para la interfaz Wireguard.

Si se obtiene una respuesta entonces consideramos que la conexión fue exitosa (los dispositivos son alcanzables).

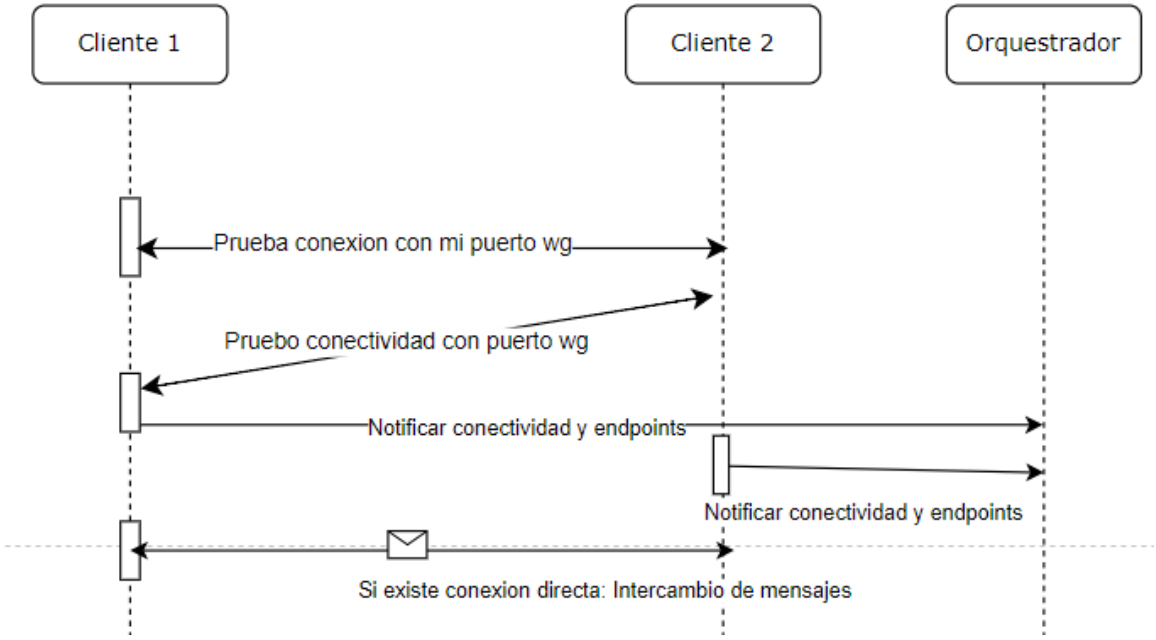

Figura 3.10: Diagrama de caso de uso: conexión de dispositivos finales con direcciones IP ruteables. Directa

Conexión de dispositivos finales con direcciones IP no ruteables. Relay

En este caso el orquestrador deberá ofrecer un mecanismo para que los dispositivos finales puedan conectarse entre sí mediante un relay. El cliente A se comunica con el orquestrador para solicitar la conexión con el cliente B, el orquestrador deberá enviar un mensaje de confirmación al cliente A con la dirección IP y puerto del orquestrador, el cliente A deberá enviar un mensaje al orquestrador para que este se comunique con el cliente B.

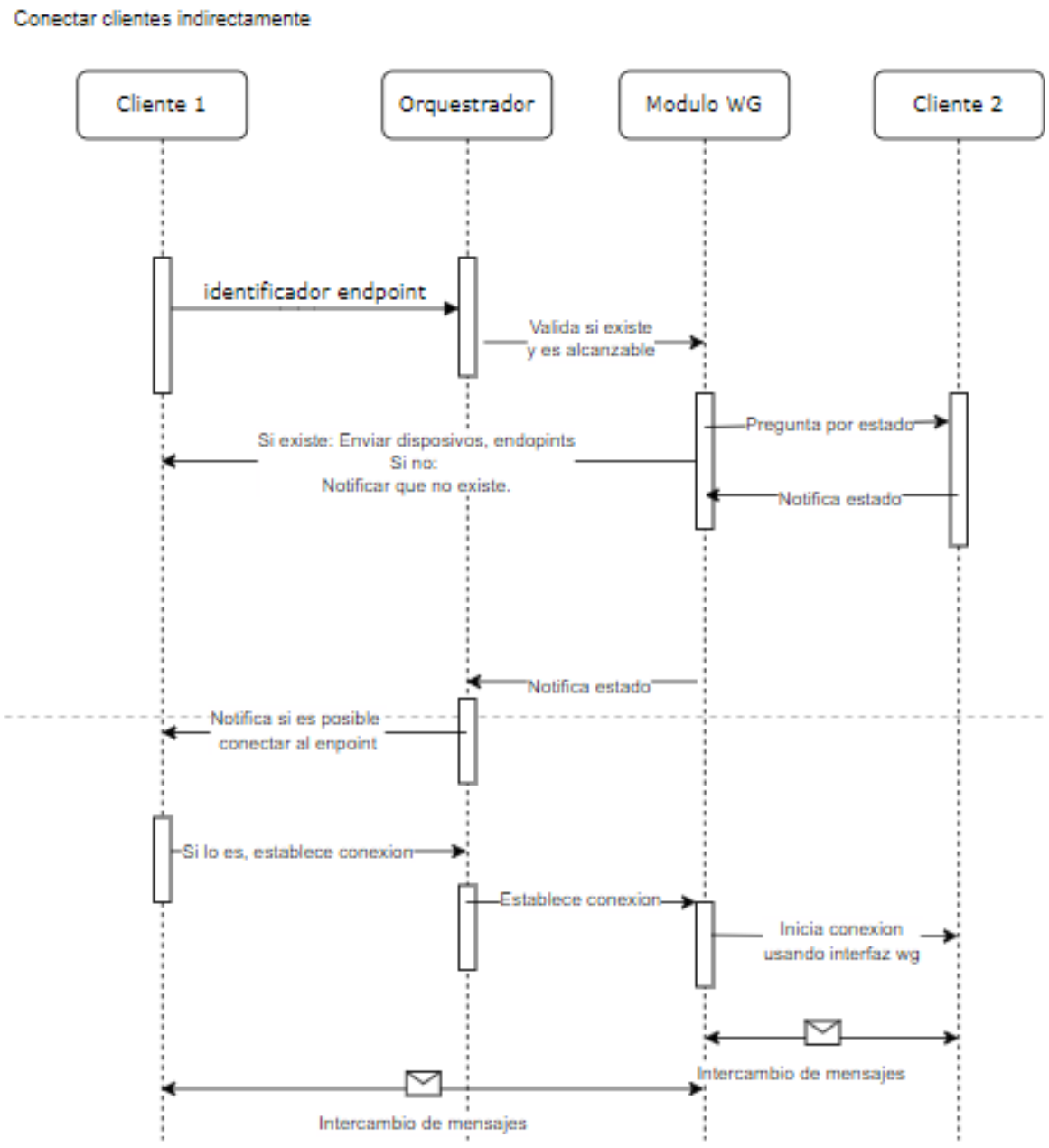

Figura 3.11: Diagrama de caso de uso: conexión de dispositivos finales con direcciones IP no ruteables. Relay

3.2. Diagrama de clases

Para el orquestrador tendremos las siguientes clases:

Bajo la idea de que el orquestrador es el encargado de orquestar la conexión entre los dispositivos finales dentro de una red privada de un cliente, tendremos las siguientes clases:

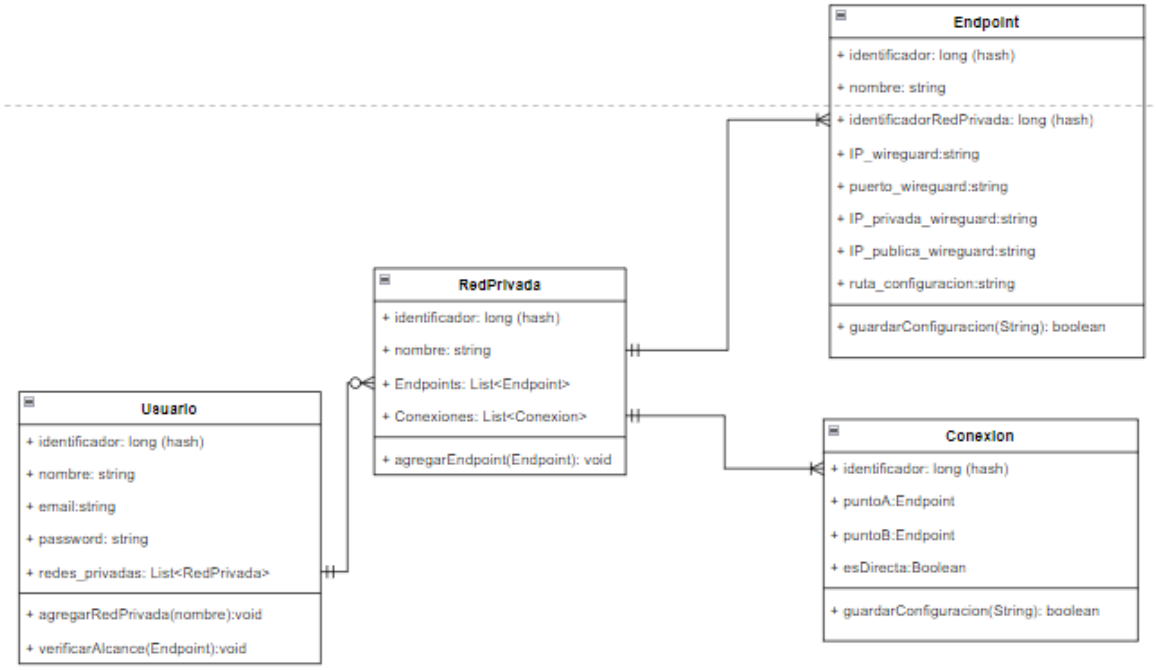

Figura 3.12: Diagrama de clases

- **Cliente**: Clase que representa a un cliente que se conecta a la red privada de otro cliente.
- **Endpoiny**: Clase que representa a un dispositivo final que se conecta a la red privada de un cliente.
- **Conexión**: Clase que representa una conexión entre dos dispositivos finales. La idea de esta clase es que el orquestrador sepa que dispositivos finales están conectados entre si y para cuales es necesario relay.

Capítulo 4

Resultados

Capítulo 5

Conclusiones

Bibliografía

[1] Abdulazeez, A., Salim, B., Zeebaree, D., y Doghramachi, D. (2020). Comparison of VPN Protocols at Network Layer Focusing on WireGuard Protocol. *International Association of Online Engineering*. Recuperado de https://www.learntechlib.org/p/218341/

[2] Bautts, M., y Dawson, M. (2000). *Linux Network Administrator's Guide*. O'Reilly Media, 3ra edición.

[3] Bautts, M., y Dawson, M. (2000). *Linux IP Masquerade HOWTO*. Disponible en: `https://tldp.org/HOWTO/IP-Masquerade-HOWTO/`

[4] Headscale. *An Open Source, Self-Hosted Implementation of the Tailscale Control Server*. Disponible en: `https://github.com/juanfont/headscale`

[5] Kurose, J. F., y Ross, K. W. (2017). *Computer Networking: A Top-Down Approach*. Pearson, 7ma edición.

[6] Linux Documentation Project. *Linux Advanced Routing and Traffic Control HOWTO*. Disponible en: `https://tldp.org/HOWTO/Adv-Routing-HOWTO/`

[7] Narayan, S., Williams, C. J., Hart, D. K., y Qualtrough, M. W. (2015). Network performance comparison of VPN protocols on wired and wireless networks. *2015 International Conference on Computer Communication and Informatics (ICCCI)*. doi:10.1109/iccci.2015.7218077

[8] Basics of the Unix Philosophy. Disponible en: `https://cscie2x.dce.harvard.edu/hw/ch01s06.html`

[9] Tailscale. *Terminology and Concepts*. Disponible en: `https://tailscale.com/kb/1155/terminology-and-concepts#relay`

[10] Tailscale. *IP Pool*. Disponible en: `https://tailscale.com/kb/1304/ip-pool`

[11] Tailscale. *Troubleshooting Device Connectivity*. Disponible en: `https://tailscale.com/kb/1411/device-connectivity#nat-types`

[12] Tailscale. *What is Tailscale?* Disponible en: `https://tailscale.com/kb/1151/what-is-tailscale/`

[13] Tailscale. *How NAT Traversal Works*. Disponible en: `https://tailscale.com/blog/how-nat-traversal-works`

[14] WireGuard. *WireGuard: Fast, Modern, Secure VPN Tunnel*. Disponible en: `https://www.wireguard.com/`