Resumen Teórica 12 : Redes Privadas Virtuales (VPN)

Tomás F. Melli

September 2025

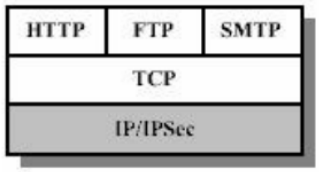
Índice

1 Redes Privadas Virtuales (VPN)

Las **Redes Privadas Virtuales (VPN)** permiten establecer conexiones seguras a través de redes públicas o no confiables, garantizando **confidencialidad, integridad y autenticidad** de la información transmitida. Se pueden definir VPN utilizando protocolos de distintos niveles del modelo OSI, dependiendo del tipo de seguridad requerida y del alcance de la protección.

1.1 Tipos de VPN y Protocolos más utilizados

- **IPSec (Internet Protocol Security)**:

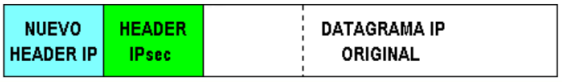


  - Protocolo de seguridad que opera en la capa de red.
  - **IPv6:** implementación obligatoria.
  - **IPv4:** uso opcional, pero ampliamente adoptado.
  - Proporciona autenticación, cifrado y protección de integridad para el tráfico IP entre dispositivos o redes.
  - Comúnmente utilizado para crear túneles VPN *site-to-site* o acceso remoto seguro.

  IPSec opera en la **capa de red** y define dos modos principales: *Transport* y *Tunnel*.

  - **Modo Transport:** se cifra únicamente la carga útil del paquete IP, dejando la cabecera IP original intacta.

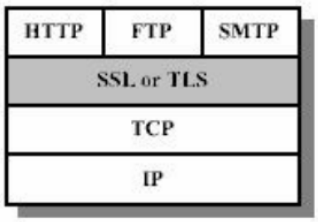
    * **ESP header:** Security Parameters Index (SPI) y número de secuencia.
    * **ESP trailer:** padding y longitud.
    * **ESP Auth:** código de autenticación opcional.
  - **Modo Tunnel:** encapsula todo el paquete IP original dentro de un nuevo paquete IP.



  - **AH (Authentication Header):** protege integridad y autenticidad, pero no cifra.

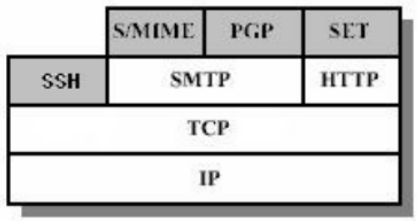
    [IP header][AH header][Payload]

- **VPN basada en TLS (Transport Layer Security)**:



  - Opera en la capa de transporte, asegurando la comunicación punto a punto.
  - Permite proteger un servicio específico (por ejemplo, HTTPS) o toda la comunicación entre dos redes.
  - Ejemplo: OpenVPN, que puede encapsular todo el tráfico IP sobre TLS.
  - El **TLS record** incluye:
    * Tipo de contenido (Handshake, Application Data, Alert)
    * Longitud
    * Versión TLS
    * Payload cifrada (datos de la aplicación o de la VPN)
  - En OpenVPN, se puede encapsular tráfico IP completo dentro del payload TLS:

- **VPN basada en SSH (Secure Shell)**:



  - **SSH packet** contiene:
    * Longitud del paquete
    * Tipo de mensaje (Data, Channel Open, Channel Close, etc.)
    * Payload cifrada
    * MAC opcional para integridad
  - Tipos de túneles:
    * **Túneles locales:** redirigen un puerto local hacia un servidor remoto.
    * **Túneles dinámicos:** usando SOCKS, permiten redirigir múltiples conexiones de manera flexible.
    * **Túneles completos de capa 2 y 3:** encapsulan tráfico de red completo, simulando una red privada sobre la pública.

- **Otros mecanismos y protocolos:** existen protocolos y técnicas especializadas según necesidades de seguridad, escalabilidad y compatibilidad con dispositivos.

Protocolos y estándares de seguridad en capa 2

En entornos corporativos o de centros de datos, la seguridad en **capa 2 (enlace de datos)** es fundamental para complementar las VPN y proteger el tráfico interno:

- **802.1Q VLANs:** permite segmentar redes físicas en redes lógicas independientes, aislando el tráfico entre segmentos.
- **802.1X Port-Based Network Access Control:** controla el acceso a la red a nivel de puerto, autenticando dispositivos antes de permitir comunicación.
- **802.1AE (MACsec):** provee cifrado y autenticación de tramas Ethernet, asegurando integridad y confidencialidad entre nodos en la misma LAN.

Para mayor detalle sobre protocolos de seguridad en capa 2, se puede consultar la documentación de Juniper Networks: Understanding Media Access Control Security.

1.2 Protocolos IPSec

Los siguientes son los principales protocolos utilizados en IPSec para construir VPN seguras:

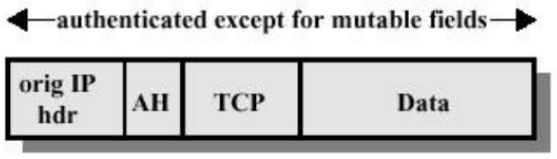
1.2.1 Internet Key Exchange (IKE)

- **Función:** negociar parámetros de seguridad e intercambiar claves criptográficas de manera segura entre los extremos de la VPN.
- Configura automáticamente:
  - Algoritmos de cifrado y autenticación.
  - Claves de sesión temporales.
- Opera sobre UDP (puerto 500).
- **Fases:**
  1. **Fase 1:** establece un canal seguro (Security Association, SA) para negociar la fase siguiente.
  2. **Fase 2:** negocia las políticas de seguridad para el cifrado y autenticación del tráfico IPSec real.
- **Mecanismos de autenticación mutua:**
  - Preshared-keys (claves compartidas previamente)
  - Pares de claves pública/privada
  - Certificados digitales

1.2.2 Authentication Header (AH)

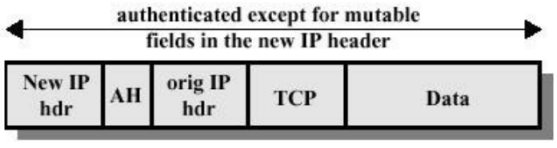
2. Authentication Header (AH)

- **Número de protocolo IP:** AH se define como **protocolo 51**.
- **Objetivo:** proporcionar autenticación y protección de integridad.
  - Verificación de la fuente (autenticidad)
  - Integridad de los paquetes
  - Protección contra ataques de repetición (anti-replay)
- **Limitación:** AH no cifra los datos ni protege la confidencialidad del flujo.
- **Posición del header de autenticación:**
  - **Modo Transport:** entre el header IP original y el payload de datos IP:



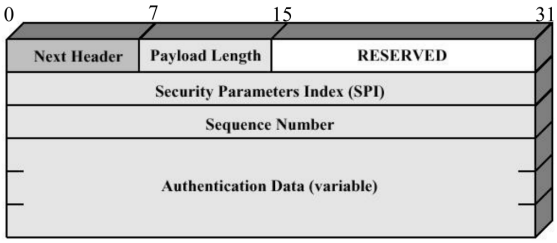
(b) Transport Mode

  - **Modo Tunnel:** como prefijo del paquete original con un nuevo header IP:



(c) Tunnel Mode

- **AH header:**

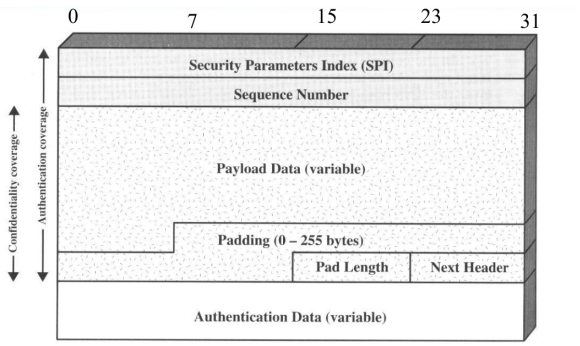


- **Next Header (8 bits):** tipo de protocolo del paquete siguiente.
- **Payload Length (8 bits):** longitud del AH.
- **Reserved (16 bits):** reservado para uso futuro, debe ser 0.
- **SPI (32 bits, Security Parameters Index):**
  - Valores 1-255 reservados para uso futuro.
  - SPI "0" reservado para uso local y no debe enviarse a la red.
- **Sequence Number (32 bits):** protege contra replays.
  - Limita la cantidad de paquetes a aproximadamente mil millones; al superarse, si la SA usa protección anti-replay, debe reestablecerse.
- **Authentication Data:** cadena de bits de longitud arbitraria usada para autenticar el paquete.

1.2.3 Encapsulating Security Payload (ESP)

ESP es un protocolo de IPSec (IP protocolo 50) diseñado para proteger el tráfico IP proporcionando **confidencialidad, integridad y autenticación**. Dependiendo del modo de la Security Association (SA), puede proteger tanto la carga útil como la cabecera original.

- **Posición del header:** similar a AH, dependiendo del modo de la Security Association (SA):
  - **Modo Transport:** entre el header IP original y el payload.
  - **Modo Tunnel:** prefijo del paquete original con un nuevo header IP.
- **Servicios de seguridad:**
  - Integridad
  - Autenticación del origen de los datos
  - Anti-replay (opcional)
    Estos servicios son los mismos que proporciona AH.
  - Confidencialidad
  - Resistencia al análisis de tráfico (modo túnel)
- **Cifrado:**
  - Algoritmos obligatorios: DES (RFC 2405) y NULL (RFC 2410)
  - Algoritmos opcionales: 3DES, CAST-128, RC5, IDEA, Blowfish y AES (RFC 2451)
  - Modo de operación: CBC
  - ESP puede usarse para proveer la misma funcionalidad que AH usando cifrado *NULL*.
- **Campos del ESP header:**



  - **SPI (32 bits):** Security Parameters Index
  - **Sequence Number (32 bits):** protege contra ataques de repetición
  - **Payload Data:** longitud variable, contiene el payload cifrado
  - **Padding:** 0-255 bytes, protege el payload contra análisis criptoanalítico
  - **Pad Length (8 bits):** indica la longitud del padding
  - **Next Header (8 bits):** tipo de protocolo del paquete siguiente
  - **Authentication Data:** checksum o MAC para autenticar el paquete

- **Método de autenticación:** emplea HMAC para integridad y autenticación.

2 NAT Traversal (NAT-T)

NAT (Network Address Translation) es una técnica utilizada para permitir que múltiples dispositivos de una red privada compartan una única dirección IP pública. Para lograr esto, NAT modifica los paquetes IP originales, cambiando direcciones IP y puertos según sea necesario. Aunque esto resuelve problemas de direccionamiento, introduce incompatibilidades con protocolos que dependen de la integridad completa del paquete, como IPSec. IPSec espera que los paquetes IP se mantengan íntegros, por lo que la modificación de los paquetes por NAT rompe la integridad que IPSec verifica y normalmente impide que la conexión se establezca.

- **Solución - NAT Traversal (NAT-T):**
  - NAT-T es un mecanismo definido en los RFC 3947 y RFC 3948.
  - Permite que el tráfico IPSec atraviese dispositivos NAT sin romper la integridad del paquete.
  - **Método:**
    * Se encapsula la trama cifrada por IPSec (antes de añadir la cabecera IP final) dentro de un paquete UDP.
    * Las operaciones de NAT se realizan sobre esta nueva cabecera UDP.
    * Esto mantiene la integridad y autenticación del payload original mientras atraviesa routers o firewalls que aplican NAT.

- **Ventajas de NAT-T:**
  - Compatibilidad de IPSec con redes que usan NAT.
  - Evita fallos en la negociación de Security Associations (SA) causados por la modificación de los paquetes.
  - Mantiene cifrado y autenticación intactos para el payload original.

3 Transport Layer Security (TLS)

Transport Layer Security (TLS) es el protocolo estándar actual para proteger comunicaciones en redes inseguras, como Internet. Es la evolución directa de SSL (Secure Sockets Layer), desarrollado por Netscape; mientras la última versión de SSL fue la 3.0, TLS se identifica como SSL v3.1. Aunque TLS y SSL comparten muchos conceptos, no son directamente compatibles, y TLS incorpora mejoras significativas en seguridad y flexibilidad. Este protocolo permite cifrar y autenticar la comunicación entre un cliente y un servidor, típicamente entre navegadores web y servidores HTTP, garantizando confidencialidad, integridad y autenticación de los datos transmitidos. TLS requiere un protocolo de transporte confiable, como TCP, para establecer correctamente sus sesiones seguras.

- **Ventajas sobre SSL:**
  - Puede iniciarse a partir de una conexión TCP ya existente, usando los mismos puertos que los protocolos no cifrados.
  - SSL no es compatible directamente con TCP, lo que requiere puertos distintos; TLS elimina esta limitación.
  - La conexión TLS se establece normalmente a través de TCP y luego activa el protocolo TLS sobre el mismo canal.

- **Mecanismos de seguridad de SSLv3 y TLS:**
  - Numeración secuencial de los paquetes que contienen datos.
  - Incorporación de la numeración al cálculo de los MAC.
  - Protección frente a ataques que intentan forzar el uso de versiones antiguas del protocolo o cifrados más débiles.

  - El mensaje que finaliza la fase de establecimiento de la conexión incorpora un hash de todos los datos intercambiados por ambos interlocutores.

- **Mecanismos adicionales de TLS:**
  - Se utilizan funciones de hash (MD5 y SHA-1 en versiones antiguas) para calcular el MAC.
  - La construcción de MAC está basada en HMAC (RFC 2104).

- **Especificaciones y RFCs:**
  - RFC 2246 (1999): TLS 1.0, basado en SSL 3.0.
  - RFC 3546 (2003): extensiones para TLS.
  - RFC 5246: TLS 1.2.
  - RFC 8446 (2018): TLS 1.3.
  - RFC 8996 (2021): deprecación de TLS 1.0 y 1.1.

- **TLS 1.2 y TLS 1.3:**
  - **TLS 1.2:** utiliza SHA-256 en lugar de MD5 o SHA-1; no puede negociar SSLv2.
  - **TLS 1.3:**
    * Más eficiente en el inicio de sesiones.
    * Deshabilita múltiples algoritmos inseguros.
    * Permite ocultar el FQDN.
    * Incluye mecanismos para dificultar ataques de downgrade.
    * Soportado en navegadores modernos.

- **Referencias adicionales:**
  - Buenas prácticas de despliegue: SSL Labs TLS Deployment Best Practices.

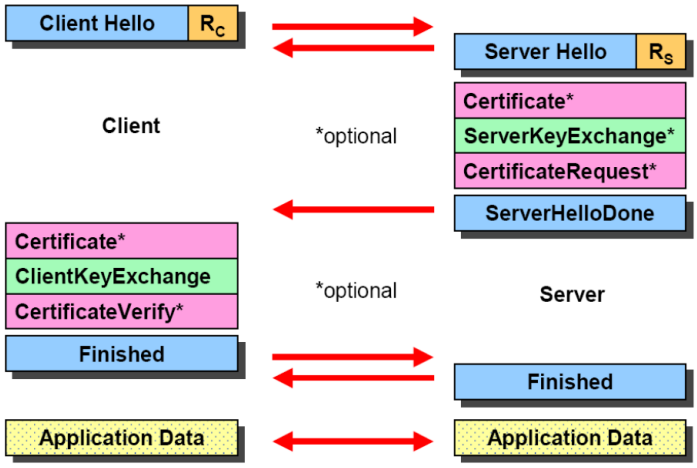
3.1 Fases de TLS

Una comunicación a través de TLS implica tres fases principales:

1. **Establecimiento de la conexión y negociación de algoritmos criptográficos:**
   - Durante esta fase, el cliente y el servidor negocian el conjunto de algoritmos criptográficos que van a utilizarse para la sesión, considerando los algoritmos soportados por cada uno de los interlocutores.
   - Esto incluye algoritmos de cifrado simétrico, funciones hash y algoritmos de clave pública.
2. **Intercambio de claves y autenticación:**
   - Se emplea un mecanismo de clave pública (como RSA o ECDHE) para intercambiar un secreto compartido.
   - Los interlocutores se autentican utilizando certificados digitales.
   - Esta fase garantiza que ambas partes conozcan la identidad de su contraparte y establezcan un canal seguro para derivar las claves de sesión.
3. **Cifrado simétrico del tráfico:**
   - Una vez derivadas las claves de sesión, todo el tráfico posterior se cifra simétricamente.
   - Se asegura la confidencialidad, integridad y autenticación de los datos intercambiados.

3.2 Handshake

El estado de la sesión TLS es controlado por el **protocolo Handshake**, responsable de negociar los atributos de la sesión. Durante el Handshake, un cliente TLS y un servidor TLS acuerdan:

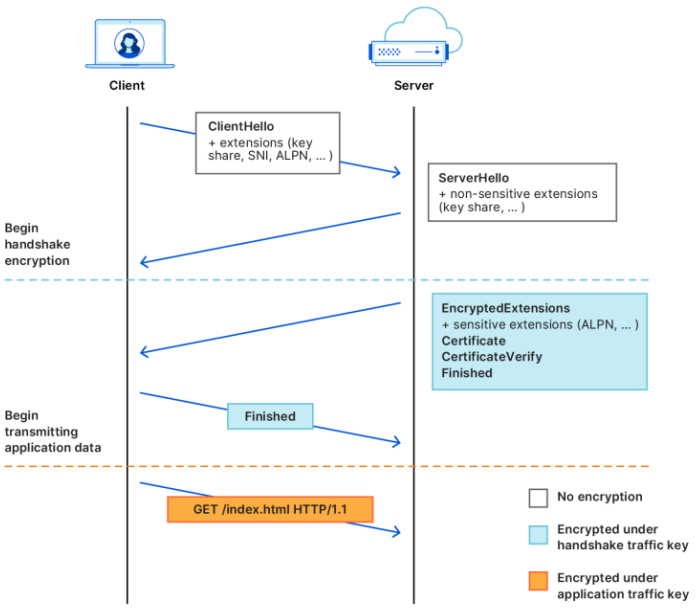


- Algoritmos criptográficos a utilizar.
- Versión del protocolo TLS.
- Autenticación opcional de uno o ambos interlocutores.
- Generación de claves de sesión mediante algoritmos de cifrado asimétrico.
- Integridad de todo el intercambio mediante hashes y MAC.

3.3 Transferencia de datos

- Tras el Handshake, se inicia la fase de transferencia de datos:
  - Se utilizan las claves derivadas durante el Handshake.
  - Se provee integridad de los datos transmitidos.
  - Opcionalmente, se cifra la información para garantizar confidencialidad.
  - Se autentica el cierre de cada conexión para evitar terminaciones no autorizadas.

3.4 Handshake en TLS 1.3

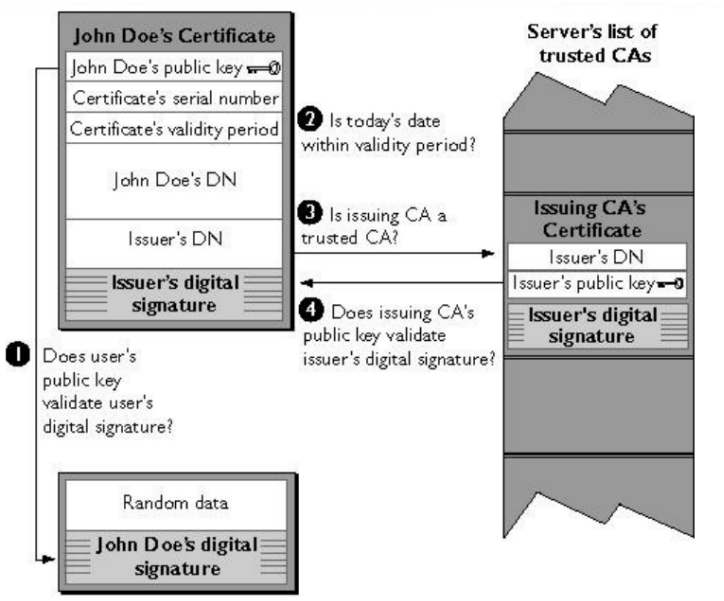


- En TLS 1.3, el Handshake se simplifica y optimiza para reducir el número de rondas necesarias.
- Incluye mecanismos para:
  - Acordar algoritmos criptográficos de manera más eficiente.
  - Autenticación mutua opcional.
  - Generación de claves de sesión derivadas de manera segura.
  - Proteger la integridad de todo el intercambio incluso durante la negociación.

3.5 Autenticación del Cliente en TLS

En TLS, además de la autenticación del servidor, es posible implementar la **autenticación del cliente** con el propósito de controlar el acceso a los recursos del servidor. Esta autenticación se basa en certificados de clave pública y la verificación de la posesión de la clave privada correspondiente.

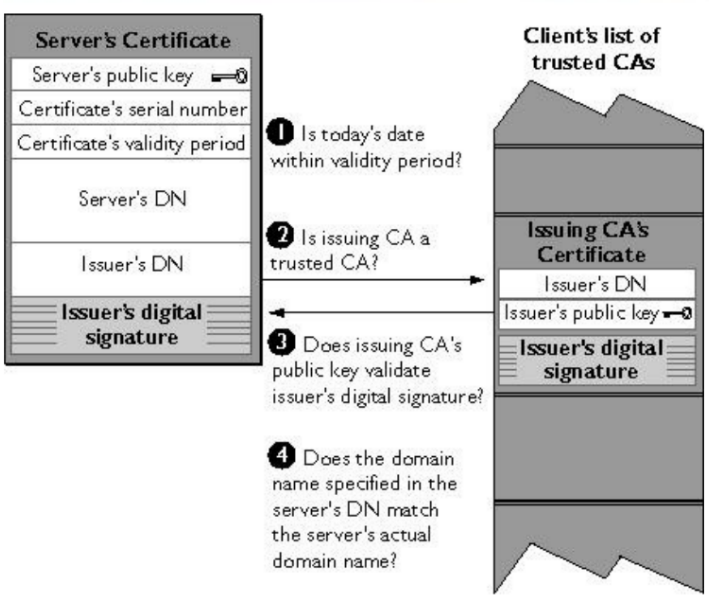
- **Flujo de autenticación del cliente:**



1. Después de que el servidor presenta su propio certificado, puede solicitar la autenticación del cliente mediante el mensaje `CertificateRequest`.
2. El cliente responde enviando su certificado de clave pública mediante el mensaje `Certificate`.
3. Para verificar que el cliente realmente posee la clave privada asociada a su certificado (KRA), el cliente envía el mensaje `CertificateVerify`.
   - Este mensaje consiste en la firma de todos los mensajes de Handshake intercambiados hasta ese momento.
   - La verificación de esta firma permite al servidor confirmar que el cliente posee la clave privada correspondiente al certificado presentado.

- **Objetivo:**
  - Garantizar que solo clientes autorizados, que posean la clave privada correspondiente a un certificado válido, puedan establecer la sesión TLS.
  - Aumentar la seguridad del acceso a los recursos del servidor mediante autenticación mutua opcional.

Verificación del certificado del servidor



- Antes de autenticar al cliente, el cliente también verifica el certificado del servidor.
- Esto asegura que la comunicación se establece con el servidor legítimo y evita ataques de tipo *man-in-the-middle*.
- El certificado del servidor debe ser válido y emitido por una autoridad de certificación confiable.

4 Perfect Forward Secrecy (PFS)

Perfect Forward Secrecy (PFS), también conocido como Forward Secrecy, es una propiedad criptográfica fundamental en protocolos de establecimiento de claves. Su objetivo principal es garantizar que la comprometida de una clave de sesión o de una clave privada a largo plazo no afecte la seguridad de sesiones anteriores. Esto significa que incluso si un atacante obtiene las claves privadas de un servidor en el futuro, no podrá descifrar las comunicaciones pasadas que se realizaron bajo sesiones protegidas con PFS.

- **Definición formal (RFC 4251, SSH):**

  "PFS is essentially defined as the cryptographic property of a key-establishment protocol in which the compromise of a session key or long-term private key after a given session does not cause the compromise of any earlier session."

- **Beneficios de PFS:**
  - Aumenta la seguridad de las comunicaciones cifradas en caso de compromiso de claves privadas a largo plazo.
  - Reduce significativamente el riesgo asociado al almacenamiento de claves privadas.
  - Es especialmente relevante en entornos donde los datos confidenciales deben permanecer seguros a lo largo del tiempo, incluso si el servidor es comprometido.
- **Implementación:**
  - PFS se logra mediante protocolos de intercambio de claves efímeras, como Diffie-Hellman efímero (DHE) o ECDH efímero (ECDHE).
  - Cada sesión genera claves de sesión únicas y temporales que no se derivan directamente de claves de largo plazo.
- **Referencias:**
  - SSL Labs: Deploying Forward Secrecy.

5 OpenVPN

OpenVPN es una solución de red privada virtual (VPN) que busca implementar comunicaciones seguras de manera más **portable y sencilla** que IPSec, aprovechando TLS para el cifrado y la autenticación de la sesión. Está diseñada para ser flexible, eficiente y fácil de desplegar en distintos entornos de red.

- **Ventajas sobre IPSec:**
  - No requiere la complejidad del protocolo IKE, simplificando la configuración y el mantenimiento.
  - Facilita la interoperabilidad entre distintos sistemas operativos y plataformas.
- **Modos de operación:**
  - **Modo Routing:** Se crean túneles IP punto a punto, permitiendo el enrutamiento de paquetes entre redes remotas.
  - **Modo Bridging:** Se crean túneles Ethernet (Layer 2), permitiendo que dispositivos de distintas redes se comporten como si estuvieran en la misma subred física.
- **Cifrado y seguridad:**
  - Utiliza la implementación de OpenSSL para manejar cifrado simétrico y asimétrico, autenticación y certificados digitales.
  - Aprovecha TLS para establecer un canal seguro y derivar claves de sesión efímeras, facilitando la compatibilidad con Perfect Forward Secrecy (PFS).
- **Referencias:**
  - Página oficial: www.openvpn.net.

6 WireGuard

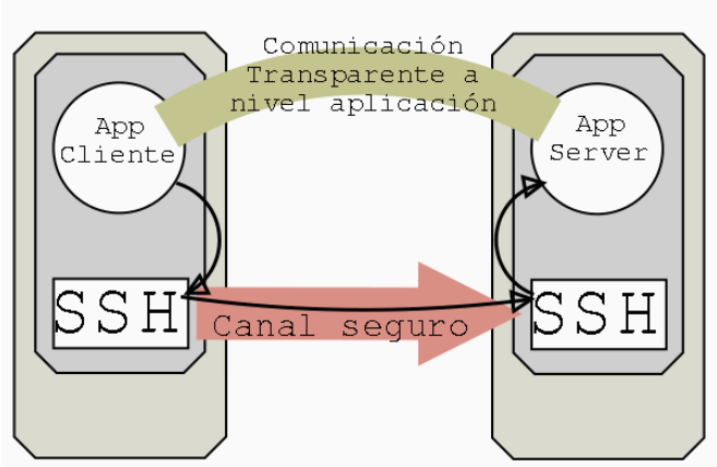
WireGuard es una solución de red privada virtual (VPN) moderna que busca simplicidad, eficiencia y seguridad, incorporando un diseño minimalista y de alto rendimiento. Está diseñado para ser más rápido y fácil de usar que soluciones tradicionales como IPSec o OpenVPN.

- **Características principales:**
  - Incorporado en el **kernel de Linux** desde la versión 5.6, lo que mejora significativamente el rendimiento.
  - **Connection-less**: funciona sin mantener estados complejos de conexión, simplificando la gestión y aumentando la eficiencia.
  - Diseño minimalista: menos líneas de código, menos algoritmos y menor complejidad, facilitando la auditoría y mantenimiento.
- **Seguridad y simplicidad:**
  - Utiliza criptografía moderna y segura de manera predeterminada.
  - Más sencillo de configurar y desplegar que soluciones tradicionales, con menos riesgos de errores de configuración.
- **Referencias:**
  - Documentación oficial: WireGuard Protocol.
  - Paper técnico: WireGuard Whitepaper.

7 Túneles SSH y Port Forwarding

Los túneles SSH permiten establecer conexiones seguras a través de redes inseguras utilizando el protocolo SSH, sin necesidad de desplegar una VPN completa. Esto es especialmente útil para proteger credenciales, atravesar firewalls restrictivos y acceder a servicios internos de manera segura.

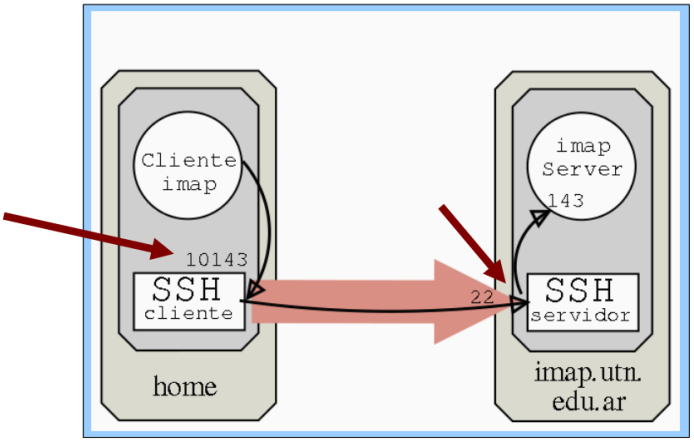
- **Usos principales:**



  - Evitar la instalación y configuración de software complejo de VPN.
  - Proteger credenciales planas de servicios como FTP, HTTP, Telnet, POP3, IMAP o SMTP autenticado.
  - Atravesar firewalls donde únicamente el servicio SSH está permitido.
  - Acceder a servicios TCP internos de una LAN con direcciones privadas.

- **Limitaciones:**
  - Los túneles SSH básicos solo permiten el reenvío de conexiones TCP.
  - No permiten forwardear paquetes UDP ni protocolos no basados en IP.

- **Forwarding local simple:**



  - Permite redirigir un puerto local hacia un host remoto a través del túnel SSH.
  - Ejemplo:

```
home:~$ ssh -L 10143:localhost:143 imap.utn.edu.ar
```

  - En este ejemplo, el puerto local 10143 se conecta al puerto 143 del servidor remoto, permitiendo acceder al servicio IMAP de manera segura.

- **SOCKS / Forwarding dinámico:**
  - SOCKet Secure (SOCKS) permite que el cliente SSH funcione como un proxy dinámico.
  - El modificador `-D` genera un túnel local dinámico, convirtiendo al cliente SSH en un servidor SOCKS.
  - Ejemplo para emular un SOCKS server en el puerto 8080:

```
$ ssh -D8080 <user>@server
```

  - Las aplicaciones que utilicen este túnel deben ser *SOCKS aware*, o se puede emplear un proxifier como proxychains-ng.

7.1 Túneles SSH con Interfaces Virtuales: TUN vs. TAP

Además de los túneles SSH simples y el forwarding dinámico, es posible crear túneles SSH que empleen interfaces virtuales para transportar tráfico de manera más transparente. Estas interfaces permiten encapsular paquetes de red completos a través de un túnel seguro, emulando una red física.

- **TUN (Network TUNnel):**
  - Trabaja en **capa 3** (nivel IP).
  - Permite encapsular paquetes IP dentro del túnel SSH.
  - Ideal para enrutar tráfico IP entre redes privadas remotas.
  - Cada extremo del túnel se comporta como una interfaz de red IP.
  - No transporta tramas Ethernet completas, solo paquetes IP.
- **TAP (Network TAP):**
  - Trabaja en **capa 2** (nivel de enlace / Ethernet).
  - Permite encapsular tramas Ethernet completas dentro del túnel SSH.
  - Ideal para transportar protocolos no-IP o para crear una LAN virtual completa.
  - Cada extremo del túnel se comporta como un switch virtual, pudiendo unir múltiples hosts como si estuvieran en la misma red física.
- **Comparación práctica:**
  - **TUN:** más eficiente, menos overhead, solo IP.
  - **TAP:** más versátil, transporta todo tipo de protocolos, mayor overhead.

8 Zero Trust

El modelo **Zero Trust** es un enfoque de seguridad de red que parte del principio fundamental de que ninguna entidad, interna o externa, debe ser automáticamente confiable. Su diseño se basa en la verificación continua y en políticas estrictas de control de acceso.

- **Acceso seguro a todos los recursos:**
  - Garantiza que todos los recursos, ya sean locales, en la nube o híbridos, estén protegidos independientemente de su ubicación.
  - Se aplica tanto a usuarios humanos como a servicios automatizados o dispositivos.
- **Mínimo privilegio y control de acceso:**
  - Se sigue estrictamente el principio de *least privilege*, otorgando únicamente los permisos necesarios para realizar tareas específicas.
  - Se implementan políticas de acceso dinámicas y contextuales, evaluando factores como ubicación, dispositivo y comportamiento del usuario.
- **Inspección y registro continuo:**
  - Todo el tráfico y actividad en la red se inspecciona y registra para detectar anomalías, intrusiones o comportamientos sospechosos.
  - Esto permite la validación continua de usuarios y dispositivos, incluso después de que hayan obtenido acceso inicial.