Módulo 2. Perímetro y acceso remoto

CÓRDOBA CREA FUTURO

Este módulo aborda la protección del perímetro de red y los servicios de acceso remoto. Se presentan los *firewalls* (cortafuegos) y el NAT (*network address translation*, o «traducción de direcciones de red»), junto con sus topologías y reglas básicas; la integración de sistemas de detección y prevención de intrusiones (IDS/IPS); métodos para la publicación segura de servicios en la DMZ (zona desmilitarizada); y la gestión de registros hacia un sistema SIEM.

En la segunda unidad se analiza la creación de VPN seguras: protocolos y herramientas (como OpenVPN, WireGuard, IPSec), autenticación y gestión de claves, modos de acceso (túnel dividido versus total), así como medidas de rendimiento y alta disponibilidad.

Se incluyen actividades prácticas en entornos virtuales o físicos con herramientas gratuitas de uso extendido (pfSense, OPNsense, OpenVPN, WireGuard) y un cuestionario final para consolidar el aprendizaje. Los conceptos técnicos se ilustran con referencias actualizadas y buenas prácticas.

Unidad 1. Firewall y NATdeo conceptual

Introducción

Los *firewalls* son componentes fundamentales en la defensa del perímetro de una red, ya que permiten controlar el tráfico entrante y saliente en función de reglas definidas.

Esta unidad explora los distintos tipos de *firewall*, su ubicación dentro de la arquitectura de red, la aplicación de reglas efectivas y el uso de NAT y alias para optimizar la gestión de la seguridad.

Los *firewalls* son sistemas basados en hardware o software que filtran el tráfico entre redes de distinta confianza, aplicando políticas de seguridad. Pueden presentarse como hardware dedicado a funciones de *firewall* (por ejemplo, *appliance* o router avanzado), o como *software* instalado en servidores o equipos personales. También existen *firewalls* personales en equipos finales, como Windows Defender Firewall. En general, se ubican en el perímetro de la red, separando Internet de la red interna (LAN).

1.1. Tipos de *firewall*, topologías perimetrales, reglas de *firewall*, NAT, alias y herramientas prácticas

Tipos de *firewall*

Existen diversas clasificaciones de *firewalls* según su funcionamiento y ubicación. Los cortafuegos pueden clasificarse en varias categorías:

- ***Firewall* de red o de filtrado de paquetes (*packet filter*) (*stateless*)** Inspecciona los paquetes según su dirección IP, puerto y protocolo. Opera en un nivel bajo, analizando los encabezados de los paquetes (IP de origen/destino, puertos, protocolo) conforme a reglas estáticas. Es muy simple y rápido, pero no rastrea el estado de las conexiones, lo que implica el riesgo de permitir tráfico no deseado al abrir puertos genéricos. Entre las herramientas más comunes para este tipo de filtrado se encuentran *iptables* y *pf*.

- ***Firewall* con seguimiento de estado (*stateful*)**: rastrea las conexiones activas y toma decisiones basadas en su estado. Mantiene información contextual de las sesiones (TCP, UDP), lo que permite decisiones más precisas. Es más seguro que el filtrado

simple, ya que detecta tráfico ilegítimo según patrones de conexión, aunque requiere mayor uso de recursos. Este enfoque es utilizado por soluciones como pfSense u OPNsense.

- ***Firewall* de aplicaciones (capa 7)**: inspecciona y filtra el tráfico a nivel de aplicación (como HTTP o DNS), interpretando protocolos específicos para detectar ataques más avanzados. Opera en la capa 7 del modelo OSI. Un ejemplo común de esta categoría es Squid, que permite filtrar tráfico HTTP con alto nivel de detalle.

- ***Firewall* basado en host (HIPS)**: se instala directamente en los equipos finales (*endpoints*) y protege el dispositivo de forma local, analizando el tráfico antes de que alcance o salga del sistema operativo.

- ***Firewall* de nueva generación (NGFW, *next-generation firewall*)**: combina múltiples funciones en un solo dispositivo. Estas funciones pueden incluir inspección profunda de paquetes (DPI), detección y prevención de intrusiones (IDS/IPS), inspección de tráfico cifrado SSL/TLS, control granular de aplicaciones, antivirus y filtrado web. Algunos NGFW funcionan desde la nube como servicio y, además, incorporan funcionalidades de

WAF (*web application firewall*), ampliando su alcance a la protección de aplicaciones web.

Topologías perimetrales

Una topología perimetral define cómo se ubican los dispositivos de seguridad en relación con la red interna, la red externa (Internet) y las zonas desmilitarizadas (DMZ). Existen diversos enfoques, pero todos persiguen el mismo objetivo: aislar segmentos con distintos niveles de confianza, aplicando controles específicos en cada frontera.

Entre las configuraciones más comunes se encuentra el uso de un único *firewall* que actúa como frontera entre la red interna (LAN) y la red externa (Internet). Este diseño básico, aunque simple de implementar, ofrece un nivel de aislamiento limitado, ya que todo el control perimetral recae en un solo punto de filtrado.

Para la publicación de servicios accesibles desde Internet —como servidores web, de correo o DNS—, es habitual recurrir a una DMZ (zona desmilitarizada), una subred intermedia aislada entre la LAN e Internet. La topología recomendada consiste en ubicar la DMZ entre dos *firewalls*: el primero filtra el tráfico proveniente de Internet hacia la DMZ, mientras que el segundo protege la LAN interna del tráfico que proviene de dicha zona intermedia. De esta manera, incluso si un atacante logra comprometer un servidor

ubicado en la DMZ, todavía deberá atravesar una segunda capa de defensa para alcanzar la red interna.

En la DMZ se alojan típicamente servicios públicos como servidores web, FTP, DNS, VoIP o de correo electrónico. Este diseño asegura que dichos servicios puedan comunicarse con el exterior sin exponer directamente los activos internos de la organización.

En entornos más complejos, se pueden desplegar *firewalls* en múltiples capas, combinando dispositivos perimetrales e internos para lograr una segmentación más profunda. Por ejemplo, algunas arquitecturas usan *firewalls* adicionales dentro de la LAN para separar áreas críticas, como bases de datos o servidores de gestión.

Para pequeñas y medianas empresas, una recomendación práctica es utilizar un *firewall* con soporte para DMZ —como pfSense— y configurar reglas que separen claramente la red de usuarios de los servicios expuestos al exterior. Así se obtiene un equilibrio entre simplicidad, aislamiento y control granular del tráfico.

Reglas de *firewall*

Cada cortafuegos aplica reglas que permiten, bloquean o rechazan el tráfico en función de direcciones IP, puertos y protocolos. Estas reglas son el núcleo del control de acceso, ya que definen qué comunicaciones están autorizadas y cuáles deben ser detenidas. Cada regla contiene parámetros clave que determinan su funcionamiento:

- **Interfaz** sobre la que se aplica la regla.
- **Dirección del tráfico**, que especifica si es tráfico entrante o saliente.
- **Protocolo**, como TCP, UDP o ICMP.
- **IP/máscara de origen** (origen del tráfico).
- **IP/máscara de destino** (destino al que se dirige el tráfico).
- **Puerto de origen/destino** aplicable principalmente a protocolos TCP y UDP.
- **Acción**, es decir, si se debe permitir, bloquear o registrar el tráfico.

Veamos un ejemplo concreto: una regla que dice *Permit TCP from 192.168.1.0/24 to 192.168.2.100 port 443* indica que se permite el

tráfico TCP proveniente del segmento 192.168.1.0/24 hacia el host *192.168.2.100*, siempre que utilice el puerto 443 (HTTPS).

En los *firewalls* modernos, las acciones básicas asociadas a una regla son las siguientes:

- **Permitir (*pass*)**. Autoriza el paso del tráfico.
- **Bloquear (*drop*)**: descarta los paquetes sin notificar al emisor. Esta opción es más segura en redes no confiables.
- **Rechazar (*reject*)**: deniega el tráfico, pero notifica al emisor con un mensaje (como RST en TCP o ICMP Unreachable).

El orden de las reglas es fundamental. Estas se procesan de arriba hacia abajo, por lo que las reglas más específicas o de mayor prioridad deben colocarse primero. Una regla genérica colocada al principio podría anular las que siguen, afectando la seguridad y funcionalidad del sistema.

Buenas prácticas

Una recomendación fundamental es aplicar el modelo *deny-by-default* o principio de privilegios mínimos. Esto implica negar todo el tráfico por defecto y luego permitir solo aquello que sea

estrictamente necesario para el funcionamiento previsto (*deny-all, allow-specific*). Otras buenas prácticas incluyen las siguientes:

REVISAR PERIÓDICAMENTE LAS REGLAS ACTIVAS:

DOCUMENTAR REGLAS Y CAMBIOS:

eliminar reglas obsoletas o que ya no cumplen una función clara reduce la superficie de ataque.

REVISAR PERIÓDICAMENTE LAS REGLAS ACTIVAS:

DOCUMENTAR REGLAS Y CAMBIOS:

registrar quién hizo qué cambio, cuándo y por qué, permite mantener trazabilidad y facilita la resolución de problemas.

Estas medidas no solo fortalecen la seguridad, sino que también permiten una gestión ordenada, sostenible y auditable de las políticas de control de tráfico.

NAT (*network address translation*)

La traducción de direcciones de red (NAT) permite modificar direcciones IP en los encabezados de los paquetes para posibilitar

la comunicación entre redes con distintos esquemas de direccionamiento. Su uso más habitual es permitir que múltiples dispositivos internos compartan una única dirección IP pública para acceder a Internet.

Existen diferentes tipos de NAT, según el objetivo de la traducción. El más común es el **source NAT (SNAT)**, que convierte las direcciones IP privadas de origen en la IP pública del *firewall* o *router*. Esto se utiliza, por ejemplo, cuando un equipo de la LAN accede a Internet: el tráfico saliente parece provenir de una sola IP pública. Dentro de este grupo se encuentra también **masquerade**, una forma dinámica de SNAT común en redes domésticas o pequeñas, donde no se asigna una IP pública fija.

Otro tipo es el **destination NAT (DNAT)**, también conocido como redirección de puertos (*port forwarding*). Esta técnica permite mapear una dirección IP o puerto público hacia un servidor interno. Por ejemplo, es posible redirigir el tráfico entrante en el puerto 80 de la IP pública hacia un servidor web ubicado en la LAN. En sistemas basados en *iptables*, esto se implementa mediante la tabla *nat*, usando la cadena *PREROUTING* y una regla de tipo DNAT. En esencia, esta práctica permite exponer servicios internos hacia el exterior de manera controlada.

Por último, existe el **NAT 1:1**, que asigna una dirección IP pública fija a una dirección IP interna fija. Este tipo se usa cuando se necesita

una traducción directa, sin compartir la IP pública con otros dispositivos o servicios.

En pfSense u OPNsense, la configuración de NAT se realiza mediante su interfaz gráfica. Allí es posible crear una regla de redirección de puertos en la interfaz WAN (mediante DNAT) o bien habilitar el enmascaramiento automático (*masquerade*) para la LAN, lo cual corresponde a un SNAT. "Bajo el capó", esto se traduce en reglas gestionadas por la tabla *nat* del subsistema Netfilter, utilizando la cadena *PREROUTING* para las redirecciones entrantes (DNAT) y *POSTROUTING* para los enmascaramientos salientes (SNAT o *masquerade*).

Por ejemplo, para redirigir el tráfico entrante en el puerto 22 (SSH) desde la interfaz WAN (*eth0*) hacia un host interno con dirección 192.168.1.10, se podría utilizar una regla como la siguiente:

iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 22 -j DNAT --to-destination 192.168.1.10:22

La correcta implementación de NAT resulta clave para el funcionamiento de toda la arquitectura de red, ya que:

- asegura que las redes privadas puedan comunicarse con Internet sin exponer sus direcciones internas;
- actúa como una primera capa de seguridad al ocultar la topología interna;
- permite definir qué zonas están autorizadas a comunicarse con la red externa (WAN);
- facilita la aplicación posterior de reglas de *firewall* o filtrado a través de *proxies;*
- y contribuye a mitigar riesgos, evitando que los servidores ubicados en la DMZ queden expuestos directamente a Internet.

NAT permite que múltiples dispositivos internos accedan a recursos externos utilizando una sola dirección IP pública, gestionando así el tráfico entre la LAN, la DMZ y la red WAN. Su implementación adecuada es crucial en cualquier diseño perimetral.

Por ejemplo, soluciones como pfSense o Endian permiten configurar NAT de salida —mediante *masquerade*— para que el tráfico de la LAN pueda salir a través de la WAN utilizando la IP pública del *firewall*. Del mismo modo, se puede configurar NAT

de entrada (DNAT) para exponer servicios internos, como servidores web o de correo, a través de redirecciones de puerto.

El uso de NAT es esencial en *firewalls* perimetrales, ya que actúa como punto de traducción entre redes con diferentes rangos de direcciones. Es importante tener en cuenta que las reglas de *firewall* se procesan después de que ocurre la traducción de direcciones; por ello, se recomienda nombrar y organizar las reglas de forma clara, por ejemplo, agrupándolas según la interfaz de entrada o salida, para evitar confusiones operativas.

Un caso de uso frecuente consiste en redirigir el puerto 443 del *firewall* a un servidor HTTPS ubicado en la red interna. De este modo, el tráfico cifrado que llega desde Internet es reenviado de forma controlada hacia el servicio correspondiente en la LAN, sin exponer directamente su dirección IP real.

Alias

Para simplificar la gestión de las reglas de *firewall*, es habitual utilizar alias: nombres o etiquetas que agrupan múltiples direcciones IP, rangos de red o puertos. Esta práctica mejora tanto la legibilidad como el mantenimiento de la configuración, ya que evita repetir direcciones o rangos en múltiples reglas.

Por ejemplo, en sistemas como pfSense u OPNsense, se puede definir un alias que represente a una lista de servidores internos o a un conjunto de puertos. Luego, en lugar de escribir cada dirección

o puerto individualmente en cada regla, se hace referencia al alias. Esto no solo facilita la lectura, sino que también permite modificar múltiples reglas al actualizar un solo alias.

Un ejemplo típico sería definir un alias llamado «Red_TRUSTED» que agrupe los rangos *192.168.10.0/24* y *192.168.20.0/24*. Al utilizar este alias en una regla, se puede aplicar la política de forma simultánea a ambos segmentos, sin necesidad de duplicar la configuración.

Herramientas prácticas

En entornos de laboratorio o en pequeñas y medianas empresas, es posible implementar *firewalls* funcionales utilizando herramientas gratuitas de amplia adopción. Entre las opciones más recomendadas se encuentran pfSense y OPNsense, ambos basados en FreeBSD, que ofrecen una interfaz web intuitiva y soporte para paquetes adicionales que amplían sus funcionalidades.

También se pueden utilizar soluciones basadas en Linux, como *iptables* o *nftables*, complementadas con interfaces más accesibles como UFW o GUFW en distribuciones como Ubuntu. Otras alternativas incluyen VyOS (un *router* basado en Linux), Endian o IPFire, todas con enfoques similares para la administración de redes y seguridad.

Para la creación de topologías de prueba, se puede recurrir a entornos virtuales utilizando herramientas como VirtualBox o VMware Player. Por ejemplo, es posible montar un laboratorio donde dos máquinas virtuales Linux actúan como clientes, y una tercera VM con pfSense cumple el rol de puerta de enlace (*gateway*), simulando así un entorno de red real.

En escenarios físicos, también es viable reutilizar un equipo antiguo como *firewall*, instalando pfSense o alguna de las distribuciones mencionadas. Otra opción es usar una Raspberry Pi como cortafuegos básico, especialmente útil en entornos domésticos o educativos, gracias a su bajo consumo y facilidad de configuración.

Tabla 1. Herramientas prácticas y gratuitas

Herramienta	Descripción	Sitio oficial
pfSense	*Firewall open source* con GUI web, soporta NAT, alias, reglas avanzadas. Ideal para pymes.	https://www.pfsense.org/
OPNsense	Derivado de pfSense, con foco	https://opnsense.org/

	en UI y plugins modernos.	
iptables / nftables	*Firewall* para sistemas Linux. Potente, *scriptable*.	https://wiki.nftables.org/
Firewalld (Linux)	*Frontend* para *iptables/nftables* con zonas y servicios predefinidos.	https://firewalld.org/

Fuente: elaboración propia.

Actividades prácticas complementarias

Estas actividades no son obligatorias, pero permiten reforzar los conceptos adquiridos a través de la práctica directa. La propuesta es experimentar con herramientas reales y entornos simulados para familiarizarse con la implementación de firewalls, NAT, DMZ y reglas de red.

- Instalar *pfSense* en *VirtualBox* con tres interfaces y configurar los segmentos LAN, WAN y DMZ. Verificar que un cliente en la LAN puede acceder a Internet mediante NAT.

- Definir alias en *pfSense* (por ejemplo, un grupo de direcciones IP internas) y utilizarlos en reglas de cortafuegos para simplificar la gestión de puertos.

- En un entorno físico, reutilizar una PC como cortafuegos (por ejemplo, con *pfSense*), conectarla a la LAN interna y simular una DMZ mediante otro switch.

Laboratorio complementario

Este laboratorio no es obligatorio ni será objeto de evaluación, pero se propone como ejercicio práctico para afianzar los conocimientos adquiridos mediante la implementación de un entorno realista de red segmentada con NAT y reglas de filtrado.

Se sugiere montar un escenario en VirtualBox utilizando *pfSense* (o *Endian*), creando tres redes: Verde (LAN), Naranja (DMZ) y Roja (WAN). Las actividades incluyen:

- Configurar NAT para que la red LAN acceda a Internet (LAN → WAN) mediante MASQUERADE.

- Aplicar reglas de firewall para permitir únicamente tráfico saliente HTTP/HTTPS desde la LAN.

- Configurar NAT para permitir tráfico saliente desde la DMZ (DMZ → WAN).
- Publicar un servidor web en la DMZ mediante DNAT, exponiendo solo los puertos 80 y 443 hacia la WAN
- Verificar la conectividad y el funcionamiento usando herramientas como *ping*, *curl* y consultando la tabla NAT.

Como variante, se puede desplegar un entorno con dos redes LAN internas y una DMZ, conectadas a través de uno o más cortafuegos. También es posible replicar estas topologías en simuladores como GNS3 o EVE-NG. En todos los casos, se ensayarán reglas de filtrado, configuración de NAT y uso de alias para simplificar la administración.

1.2 IDS/IPS integrados – Herramientas

Un IDS (sistema de detección de intrusiones) monitoriza el tráfico en busca de patrones maliciosos y alerta cuando detecta actividad sospechosa, sin bloquear directamente. Analiza el flujo de red o eventos de hosts para identificar firmas conocidas o comportamientos anómalos, y notifica al administrador para que actúe.

Por su parte, un IPS (sistema de prevención de intrusiones) opera en tiempo real para bloquear automáticamente el tráfico malicioso cuando se reconoce como tal. A diferencia del IDS, que es pasivo y solo notifica, el IPS es activo y descarta o rechaza paquetes para detener ataques en curso.

En la práctica, muchas plataformas de *firewall* o de *unified threat management* (UTM) ofrecen soluciones que integran funciones de IDS e IPS. Por ejemplo, en pfSense es posible habilitar estas capacidades instalando paquetes como Snort o Suricata. OPNsense incluye Suricata de fábrica, y Endian dispone de módulos basados en Snort. Estas herramientas permiten importar conjuntos de reglas basadas en firmas o en análisis de anomalías y operar en modo IDS (solo alerta) o IPS (bloqueo activo de tráfico malicioso).

Entre las herramientas libres más utilizadas se encuentran *Suricata* y *Snort*, ambas muy extendidas y con soporte de la comunidad, así como sistemas especializados orientados a la detección y análisis de seguridad, como Security Onion, una distribución de Linux que integra herramientas de IDS junto con soluciones de análisis y visualización como ELK. Para la detección en hosts, existen proyectos como Wazuh y OSSEC, que se enfocan en la supervisión de integridad de archivos, análisis de *logs* y correlación de eventos.

En entornos de laboratorio es recomendable instalar *Suricata* en *pfSense* u OPNsense y habilitar reglas básicas para familiarizarse

con su funcionamiento. Por ejemplo, se puede simular un ataque con una herramienta como *nmap* para generar alertas y observar cómo el sistema IDS/IPS responde ante esos eventos.

Consejos de laboratorio

Se puede configurar Snort o Suricata en un *firewall* gratuito como pfSense, cargando un conjunto de reglas conocidas, como las del proyecto Emerging Threats. A continuación, se genera tráfico sospechoso desde una máquina cliente (por ejemplo, intentos de escaneo o explotación de servicios SSH) y se observan las alertas en la interfaz del IDS. Luego, se puede probar el modo IPS habilitando el bloqueo activo, por ejemplo, denegando peticiones ICMP de *ping* maliciosas.

Herramientas

Entre las herramientas de código abierto más utilizadas en la comunidad se encuentran Snort, muy popular para detección basada en firmas, y Suricata, que ofrece alto rendimiento mediante multihilo, inspección profunda de paquetes (*deep packet inspection*, DPI) y compatibilidad con las reglas de Snort. Ambos analizan el tráfico en tiempo real y pueden tanto generar alertas como, en modo IPS, bloquear ataques automáticamente.

Otra alternativa es Zeek (anteriormente Bro), que se enfoca en el análisis de protocolos y la detección de comportamientos anómalos. También existen IDS basados en host, como OSSEC o su

bifurcación Wazuh, que inspeccionan registros del sistema y controlan la integridad de archivos.

En la práctica, es común instalar el sensor IDS/IPS en el mismo *firewall* o en un dispositivo cercano. Por ejemplo, pfSense y OPNsense permiten instalar fácilmente Snort o Suricata como paquetes adicionales. Una opción más avanzada es Security Onion, una distribución Linux que incluye herramientas como Snort, Suricata, Zeek, ELK y otras, pensada para el monitoreo integral del tráfico en redes. Los eventos generados por estos sistemas pueden enviarse a un SIEM centralizado para su análisis y correlación.

Ejemplo de uso integrado

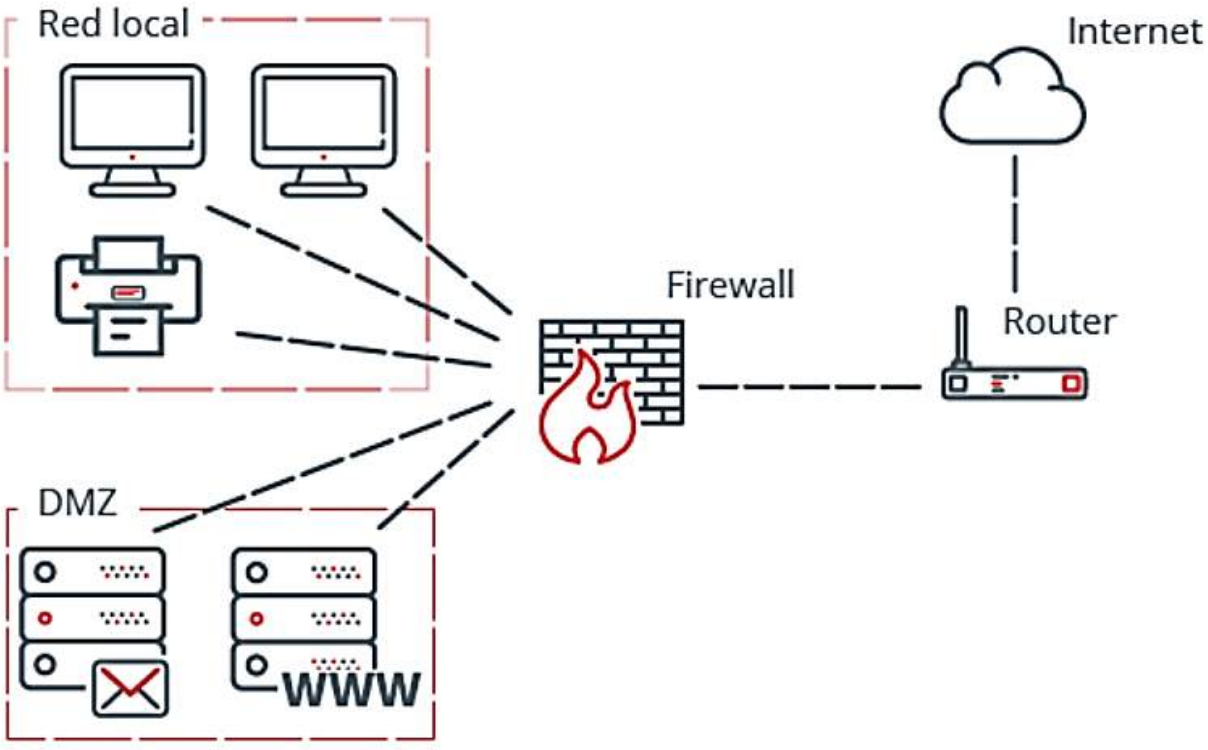
Un *firewall* UTM puede incluir una regla que redirija el tráfico hacia Suricata para su inspección antes de decidir si permitirlo o bloquearlo. De este modo, se combinan el filtrado básico mediante reglas del *firewall* con la prevención de ataques complejos a través del sistema IPS, todo dentro de una misma plataforma.

1.3. Publicación segura de servicios

Publicar servicios internos en Internet (como sitios web, correo electrónico, FTP, entre otros) aumenta significativamente el riesgo de sufrir incidentes de seguridad. Por ello, la práctica recomendada es exponer estos servicios dentro de una zona desmilitarizada (DMZ), que se encuentra aislada de la red interna mediante uno o

dos *firewalls*. En la DMZ se ubican exclusivamente los servidores que deben ser accesibles desde el exterior; de este modo, si un atacante logra comprometer alguno de estos sistemas, su avance hacia la LAN queda contenido gracias a la separación y protección que ofrece el *firewall*. El siguiente diagrama ilustra esta arquitectura de publicación segura:

Figura 1. Esquema de publicación segura de servicios mediante DMZ y firewalls perimetrales



Fuente: García, 2021, https://goo.su/ryuut

Como se observa en el esquema, la LAN interna (ubicada en la parte superior) y la DMZ (en la parte inferior) están separadas por un *firewall* perimetral. Solo los servicios ubicados en la DMZ —como los de web (WWW) o correo electrónico (Mail)— son accesibles desde Internet, y únicamente a través de reglas de NAT controladas.

Es importante señalar que publicar cualquier servicio en Internet incrementa significativamente el riesgo de sufrir incidentes de seguridad. Para reducir esa exposición, se recomienda crear una zona desmilitarizada (DMZ) donde ubicar exclusivamente los servicios que deban ser accesibles desde el exterior. Si se va a publicar un servidor en una red corporativa, debe situarse dentro de una DMZ correctamente aislada. Asimismo, en esta zona resulta fundamental aplicar sistemas IDS/IPS y mantener actualizados los servidores para minimizar vulnerabilidades.

Actividad práctica sugerida

En esta actividad, que no es obligatoria ni requiere valoración, se propone simular una DMZ en VirtualBox. Para ello, se debe configurar en pfSense tres zonas (LAN, WAN y DMZ), ubicar un servidor web en la DMZ y crear reglas de NAT/DNAT que permitan exponer únicamente el puerto 80 desde Internet hacia dicho servidor. Luego, se debe probar la conexión desde un host externo (simulando el acceso desde Internet) y

verificar que la red interna (LAN) no sea accesible desde el exterior.

Cuando se exponen servicios internos a Internet, se busca hacerlo de forma controlada. La estrategia habitual consiste en colocar dichos servidores en la DMZ, aislados de la LAN. Desde el exterior se accede a ellos mediante reglas de NAT (DNAT) en el *firewall*, abriendo solo los puertos imprescindibles, como los puertos 80 o 443 en el caso de un servidor web. Esto impide que los clientes externos alcancen directamente la red interna. Además, para reforzar la publicación de servicios, se pueden emplear *proxy* inversos o WAF, aplicando autenticación y filtrado.

Un *proxy* inverso (*reverse proxy*) se ubica en la DMZ, donde recibe las peticiones externas y las reenvía a los servidores *backend* internos. Actúa como intermediario que puede aplicar filtros, balanceo de carga, almacenamiento en caché y cifrado (TLS). Por ejemplo, un servidor Nginx o HAProxy en la DMZ puede recibir todo el tráfico HTTPS, verificar los certificados, aplicar controles de acceso (o módulos de seguridad como *mod_security*) y luego reenviar la petición al servidor web real en la red interna. De este modo, los servidores *backend* quedan protegidos tras el *proxy* y el *firewall*, evitando ataques directos.

El *firewall* debe tener reglas estrictas que permitan únicamente el tráfico entrante hacia direcciones IP y puertos específicos en la DMZ, negando todo lo demás. Como señala la documentación: “un *reverse proxy* protege tus servidores públicos actuando como intermediario: acepta las peticiones de los clientes, las reenvía a los servidores *backend* y entrega la respuesta al usuario”. Suele desplegarse en la DMZ junto con un *firewall* adicional.

Alternativamente, algunas infraestructuras emplean *jump hosts* o puertas de enlace seguras para la administración remota (RDP/SSH), en lugar de exponer estos servicios directamente. También es una buena práctica exigir cifrado TLS en los servicios públicos y aplicar listas de acceso estrictas.

En resumen, la publicación segura de servicios combina el uso de DMZ, reglas de NAT estrictas y, en muchos casos, *proxies* o WAF. Por ejemplo, en la DMZ se ubicaría un servidor web accesible desde Internet, y el *firewall* se configuraría con una regla DNAT que permita únicamente el acceso a los puertos 80 o 443 de ese servidor. De forma opcional, se podría interponer un *proxy* inverso para gestionar el cifrado TLS y validar las solicitudes entrantes. Este enfoque garantiza que, incluso si el servicio público es atacado, la red interna no se vea directamente comprometida.

1.4. *Logs* y exportación a SIEM

Los *firewalls* y sistemas de seguridad generan información valiosa como registros de conexiones, alertas del IDS o eventos de autenticación. Esta telemetría debe ser recolectada de forma centralizada mediante un sistema de *security information and event management* (SIEM), que permite analizar, correlacionar y conservar estos datos de manera segura. El SIEM centraliza eventos provenientes de distintos dispositivos —*firewalls*, servidores, *endpoints* y servicios en la nube— para facilitar su monitoreo conjunto.

El registro de eventos es una función esencial: todo *firewall* debe generar *logs* detallados sobre conexiones, cambios de configuración y alertas de seguridad. En la práctica, sistemas como pfSense permiten configurar el envío remoto de registros desde la sección «Status > System Logs > Settings», activando la opción de «Remote Logging» e indicando la IP del servidor receptor.

Estos registros se envían normalmente a un servidor *syslog* o a una solución SIEM, donde se almacenan, se indexan y se visualizan mediante paneles de control. Es importante establecer políticas de retención y cifrar los canales de envío —por ejemplo, mediante *stunnel* o una VPN— si los *logs* contienen información sensible.

El SIEM permite además correlacionar eventos de distintas fuentes para identificar incidentes complejos. Al combinar registros de varios dispositivos, facilita la detección de patrones anómalos o comportamientos sospechosos.

Por ejemplo, pfSense puede generar registros tanto del *firewall* como del IDS. Estos registros se envían por UDP o TCP a un servidor Graylog instalado en una máquina virtual. Graylog se encarga de recibirlos, normalizarlos y visualizarlos en paneles gráficos, lo que permite un análisis más eficiente y detallado de la actividad de red.

Cabe señalar que existen opciones libres para la centralización y análisis de *logs*. Entre ellas, se destacan Graylog, que ofrece una interfaz web para búsquedas y alertas; ELK (ElasticSearch, Logstash y Kibana); Wazuh; OSSIM/AlienVault; Splunk en su versión gratuita; y Security Onion, que integra múltiples herramientas como ElasticSearch, Kibana y motores de correlación.

En entornos Windows, herramientas como Kiwi Syslog Server (versión gratuita) también permiten recibir registros de red.

Es recomendable habilitar el envío de *logs* relacionados con traducción de direcciones (NAT), sistemas de prevención de intrusiones (IPS) y eventos de autenticación hacia ese servidor centralizado. Además, se sugiere configurar filtros que prioricen el almacenamiento de *logs* críticos —como cambios en reglas, accesos denegados o alertas del IDS— para facilitar auditorías y asegurar el cumplimiento normativo.

Laboratorio sugerido

En este laboratorio, que no es obligatorio ni requiere valoración, se propone desplegar un entorno básico de recolección de logs. Para ello, se debe levantar una máquina virtual con Graylog o ELK (puede utilizarse Docker para simplificar la instalación) y configurar una entrada *syslog*.

Luego, en el *firewall* (pfSense u OPNsense), se debe habilitar el envío de logs mediante syslog hacia el servidor configurado. A continuación, se deben generar eventos —por ejemplo, un intento de conexión bloqueada— y verificar que dichos registros aparecen correctamente en el sistema SIEM.

En resumen, un esquema robusto de perímetro debe incluir *firewalls*, una DMZ, sistemas IDS/IPS y un SIEM para centralizar y visualizar la actividad de red. De acuerdo con las recomendaciones del NIST SP 800-41, la combinación de NAT, filtrado y *proxy* permite construir un entorno seguro, alineado con buenas prácticas de seguridad en redes.

CONTINUAR

Lección 2 de 3

Unidad 2. VPN segura

Introducción

Una *VPN* (virtual private network) es un tipo de conexión que permite cifrar y mejorar la seguridad de la comunicación entre un usuario y una red. Para ello, crea túneles seguros cifrados para conectar redes o usuarios remotos. Los protocolos y herramientas utilizados varían según las necesidades.

Una *VPN* (red privada virtual) es una tecnología que establece una conexión segura y cifrada entre un dispositivo e internet, funcionando como un «túnel» privado que oculta la dirección *IP* y protege los datos del proveedor de servicios de internet, anunciantes y atacantes, lo que permite una navegación con mayor privacidad, seguridad y sin restricciones geográficas o censura. Redirige el tráfico a través de un servidor remoto, haciendo que la conexión parezca originarse desde la ubicación de ese servidor, y cifra los datos para que resulten ilegibles ante cualquier intento de interceptación.

Una *VPN* se utiliza con diversos fines. Entre ellos, se encuentran los siguientes:

Privacidad. —

Impide que proveedores de internet, anunciantes o entidades gubernamentales rastreen la actividad en línea.

Seguridad: —

protege los datos al utilizar redes Wi-Fi públicas o vulnerables.

Acceso a contenido: —

posibilita el ingreso a sitios web y servicios con restricciones geográficas (por ejemplo, por cuestiones de censura o licencias de distribución).

Trabajo remoto: —

permite a los empleados conectarse de forma segura a la red interna de su empresa.

En síntesis, una *VPN* brinda una experiencia en línea más segura, privada y sin restricciones territoriales.

2.1 Herramientas e implementaciones

Las VPN ofrecen un nivel considerable de anonimato sin requerir el uso de TOR. Si bien no emplean mecanismos de ocultamiento tan sofisticados, proporcionan un umbral de seguridad adecuado para un uso general.

Entre las ventajas que pueden presentar las VPN, se destacan las siguientes:

- Posibilidad de conexión y desconexión de manera sencilla.
- Mejora en la seguridad de la conexión.
- Capacidad de modificar la ubicación geográfica aparente.
- Impide que el proveedor de internet acceda al historial de navegación.
- Permite eludir mecanismos de censura.

Por otro lado, también presentan algunas limitaciones:

- Disminución de la velocidad de navegación.
- Reducción de la velocidad de descarga.
- Costos asociados.
- No garantizan una seguridad absoluta.

Herramientas e implementaciones

Las VPN se implementan con distintos protocolos, y cada uno se adapta mejor a casos de uso específicos.

- **IPsec (IKEv2/IPsec)**. Protocolo estándar ampliamente soportado por *routers* y

sistemas operativos como Windows y Linux. Es adecuado para interconexiones entre sitios (*site-to-site*) y para acceso remoto corporativo. Utiliza algoritmos como AES y modos túnel o transporte. Requiere, por lo general, certificado o clave precompartida (PSK) para la autenticación. Ejemplos de herramientas que lo implementan son *strongSwan* o *Libreswan* en Linux.

- **OpenVPN**: protocolo basado en *SSL/TLS*, versátil y compatible con Windows, Linux, macOS y dispositivos móviles. Funciona sobre UDP o TCP y atraviesa *NAT* y cortafuegos con facilidad. Su configuración es flexible y permite tunelizar en capa 2 o 3. Se utiliza con frecuencia para acceso remoto de empleados mediante *clientes OpenVPN* instalados en equipos personales.

- **WireGuard**: protocolo reciente incluido en Linux a partir del *kernel* 5.6. Destaca por su simplicidad y velocidad, gracias al uso de cifrados modernos como Curve25519 y ChaCha20, y a una base de código reducida que facilita su auditoría. Generalmente, ofrece mayor rendimiento y menor consumo de CPU que OpenVPN. Es ideal cuando se prioriza el rendimiento y se trabaja principalmente con Linux, Android o iOS. También cuenta con *clientes* para Windows y macOS. Actualmente,

no incluye soporte avanzado como RADIUS, pero en muchos casos reemplaza a OpenVPN o IPsec.

- **SoftEther VPN**: *software* libre multiprotocolo que admite conexiones mediante *SSL-VPN*, L2TP, OpenVPN, entre otros. Permite crear redes privadas virtuales compatibles con distintos estándares desde un mismo entorno.

- **L2TP/IPsec**: combinación comúnmente utilizada, con soporte nativo en muchos sistemas. Es más simple de configurar en ciertos entornos, aunque puede ser menos eficiente debido a la doble encapsulación y ofrece menor flexibilidad que OpenVPN o WireGuard.

- **PPTP**: protocolo antiguo y actualmente desaconsejado por sus vulnerabilidades.

¿Cuándo utilizar cada uno?

La elección del protocolo depende del contexto y los requerimientos específicos de uso.

- OpenVPN es una opción sólida cuando se busca amplia compatibilidad y funciones

como túnel dividido (*split-tunneling*) o autenticación mediante certificados.

- WireGuard resulta adecuado cuando se priorizan la velocidad y la eficiencia, por ejemplo, en enlaces con alto volumen de tráfico.

- Tanto OpenVPN como WireGuard son recomendables para acceso remoto de empleados.

- IPsec es preferible para enlaces permanentes entre *routers* o sedes, como en el caso de la configuración de un túnel *site-to-site* en *firewalls* como pfSense.

- SoftEther o L2TP/IPsec se utilizan en escenarios donde el *cliente* presenta restricciones, como ocurre con ciertos dispositivos móviles.

- SSL-VPN (OpenVPN o SoftEther) es útil en entornos con *NAT* estricto.

- Soluciones como pfSense u OPNsense incluyen servidores IPsec y OpenVPN integrados, lo que permite su implementación sin costo adicional.

- L2TP/IPsec es una alternativa habitual para equipos con clientes integrados en sistemas Windows.

Herramientas gratuitas

Todos los protocolos mencionados (OpenVPN, WireGuard, strongSwan) son de código abierto. Además, pfSense y OPNsense incluyen servidores OpenVPN e IPsec listos para habilitar, junto con paquetes de WireGuard. *Algo VPN* es un *script* libre que permite desplegar fácilmente una red WireGuard en la nube.

Laboratorio sugerido

A modo de sugerencia, se recomienda realizar las siguientes prácticas para afianzar los conceptos abordados:

- **Configurar un servidor OpenVPN en pfSense, generar certificados con *Easy-RSA*, crear usuarios y verificar el acceso desde un *cliente* con Windows o Linux.**
- **Configurar una conexión WireGuard instalando el protocolo en dos sistemas Linux o en OPNsense, intercambiar claves públicas y probar la conectividad mediante *ping* a través del túnel.**

- **Montar un enlace *site-to-site* mediante IPsec, por ejemplo, entre dos dispositivos pfSense o entre dos sistemas Linux con strongSwan, y transferir tráfico interno a través de dicho enlace.**

2.2. Autenticación y gestión de claves

Las VPN seguras requieren una adecuada gestión de claves y autenticación de identidades. En cualquier implementación que priorice la seguridad, resulta fundamental autenticar correctamente a las partes y administrar las llaves utilizadas.

- **Certificados X.509 (PKI)**

Frecuentemente se emplea infraestructura de clave pública (PKI) combinada con TLS y certificados X.509. El proceso implica:

- Crear una autoridad certificadora (CA) interna, mediante herramientas como easy-rsa o XCA.
- Emitir y firmar certificados para el servidor VPN y los clientes (o routers).

De este modo, el servidor puede verificar el certificado del *cliente* y viceversa, estableciendo una relación de confianza mutua. Este

enfoque permite además la revocación de claves y favorece la escalabilidad. Es común en implementaciones de OpenVPN e IPsec. Herramientas como *easy-rsa*, incluida con OpenVPN, simplifican la creación de CA y certificados. Entre sus ventajas se encuentran una autenticación robusta, la posibilidad de implementar listas de revocación de certificados (CRL) y la integración con mecanismos de autenticación en dos pasos, como TOTP.

- **Claves precompartidas (PSK)**

En entornos reducidos, algunas implementaciones optan por utilizar una clave precompartida (PSK). Este método es habitual en escenarios pequeños con IPsec o L2TP. Aunque presenta una gestión más simple, también implica mayores riesgos: el uso de un único secreto compartido limita la escalabilidad y expone la red si la clave es comprometida. En este caso, se configura la misma clave manualmente en ambos extremos. Es útil para enlaces simples, en los que no se dispone de una infraestructura de certificados.

- **Autenticación de usuario**

En OpenVPN, la autenticación puede complementarse con credenciales de usuario y contraseña en el canal TLS, o bien

integrarse con servicios como RADIUS o LDAP. De este modo, se verifica tanto la identidad del usuario como la validez del certificado.

Protocolo de autenticación extensible (EAP)

EAP es un marco flexible para la autenticación en redes, utilizado para permitir el acceso seguro de usuarios o dispositivos mediante distintos métodos, como contraseñas, certificados o tokens. Es común su aplicación en entornos wifi y VPN, especialmente a través del estándar 802.1X.

Su funcionamiento se basa en facilitar una comunicación entre un cliente (par) y un servidor de autenticación, como RADIUS. Esta interacción es mediada por un autenticador, que puede ser un punto de acceso o un switch, el cual transmite los mensajes necesarios. Este esquema garantiza que solo los usuarios autorizados accedan a la red, mediante un proceso de verificación de credenciales que, en la mayoría de los casos, está cifrado.

Entre los métodos más utilizados dentro del marco EAP se encuentran EAP-TLS, PEAP y EAP-TTLS, los cuales ofrecen distintos niveles de seguridad y complejidad en la experiencia del usuario.

- **Autenticación en WireGuard**

WireGuard no utiliza certificados X.509 ni cuenta con infraestructura CA o TLS de forma nativa. Su modelo de autenticación se basa en el uso de claves públicas, sin mecanismos centralizados como RADIUS. Cada par (*peer*) dispone de una clave pública y una clave privada. La autenticación se establece mediante el intercambio directo de estas claves entre el *cliente* y el servidor, lo que simplifica considerablemente la configuración y gestión. Este intercambio puede realizarse de forma manual o con herramientas como *wg-quick*, facilitando su implementación en entornos donde se prioriza la simplicidad y el rendimiento.

En las VPN de acceso remoto, como OpenVPN o WireGuard, es habitual contar con una autoridad certificadora (CA) y un certificado para el servidor, generando además certificados individuales para cada *cliente*.

En el caso de IPsec (particularmente con IKEv2), pueden utilizarse claves precompartidas (PSK) o certificados. Sin embargo, en entornos empresariales se recomienda emplear certificados emitidos por una CA interna, ya que ofrecen un nivel superior de seguridad.

La gestión de claves incluye también su rotación periódica. Una buena práctica es implementar *perfect forward secrecy* mediante el uso de claves efímeras de Diffie-Hellman (DH ephemeral) en los túneles.

Para facilitar la administración, existen herramientas como *easy-rsa* (en OpenVPN) u OpenSSL, que permiten generar CA, solicitudes de firma de certificado (CSR) y listas de revocación (CRL).

En entornos corporativos de mayor escala, es posible integrar un servidor RADIUS o aplicar autenticación multifactor (2FA) en los *clientes* VPN. Muchas soluciones admiten métodos como Google Authenticator o *tokens* físicos para reforzar la seguridad.

Aplicación práctica en laboratorio

En los ejercicios de laboratorio se recomienda practicar ambos enfoques de autenticación:

- **Generar una autoridad certificadora (CA) con easy-rsa y emitir certificados TLS para el servidor y los clientes en OpenVPN.**
- **Crear pares de claves con el comando wg genkey para configurar túneles en WireGuard.**

En el caso de IPsec, utilizar strongSwan para implementar configuraciones tanto con certificados como con autenticación EAP, según el escenario requerido.

2.3. Posturas de acceso y *split-tunnel*

Las posturas de acceso se refieren a las condiciones que debe cumplir un dispositivo *cliente* antes de permitirle el acceso mediante VPN, como parte de una política de seguridad. Estas condiciones pueden incluir la presencia de un antivirus activo, un sistema operativo actualizado o configuraciones específicas.

Si bien estas verificaciones suelen gestionarse mediante soluciones de control de acceso a la red (NAC) o arquitecturas de acceso zero trust (ZTNA), también pueden implementarse en algunas VPN avanzadas mediante *scripts* o *software*. Por ejemplo, pfSense permite ejecutar un *script* de comprobación al establecer la conexión, como verificar la presencia de un antivirus antes de asignar una dirección IP.

La postura de acceso también influye en la forma en que se enruta el tráfico una vez establecida la conexión VPN. Un concepto central en este contexto es el del túnel dividido (*split-tunnel*).

El enfoque tradicional es el túnel completo (*full-tunnel*), en el que todo el tráfico del *cliente*, incluido el acceso a internet, se enruta de

forma cifrada a través de la VPN. Este modo es preferido cuando se busca máxima privacidad y control del tráfico.

> Por otro lado, el túnel dividido permite que solo parte del tráfico utilice la VPN, mientras que el resto se enruta directamente a través del proveedor de internet local. Es decir, únicamente el tráfico destinado a la red corporativa pasa por la VPN, mientras que las conexiones a servicios externos, como plataformas de entretenimiento o búsquedas en línea, se realizan de forma directa.

Este enfoque permite reducir la carga sobre el túnel VPN, ya que el tráfico que no requiere seguridad no consume ancho de banda cifrado. Además, mejora el rendimiento y permite al usuario seguir accediendo a recursos locales, como impresoras o servidores en la red LAN, sin necesidad de desconectar la VPN. También puede ayudar a reducir la latencia y el uso de recursos del servidor VPN.

Sin embargo, el uso de túneles divididos implica ciertos riesgos. Al permitir que parte del tráfico no esté cifrado ni pase por los controles corporativos, se pierde la capacidad de aplicar medidas como filtros web, *firewalls* o *proxies*. En caso de que el dispositivo del *cliente* esté comprometido, podría representar una vía de entrada a la red interna, ya que parte del tráfico no es inspeccionado. En contraste, el túnel completo garantiza que todo el tráfico pase por la red central, donde puede ser monitoreado y

protegido, aunque a costa de un mayor consumo de ancho de banda.

La elección entre uno y otro depende de la política de seguridad de la organización. En entornos que requieren altos niveles de protección, se suele forzar el uso de túnel completo. En cambio, organizaciones con menos restricciones de ancho de banda pueden habilitar el túnel dividido como forma de equilibrar seguridad y rendimiento.

En cuanto a su configuración, tanto pfSense como OPNsense permiten activar o desactivar el túnel dividido en los *clientes* OpenVPN o IPsec. En dispositivos móviles, como Android o iOS, esta opción suele configurarse manualmente, evitando redirigir todo el tráfico a través de la VPN. Una de las ventajas más prácticas del túnel dividido es que permite mantener el acceso a dispositivos locales, como impresoras o servidores de archivos, mientras se utiliza la conexión VPN para acceder a recursos remotos.

2.4. Medición de desempeño y *fallback*

El desempeño y la calidad de servicio de una VPN se evalúan a través de parámetros como el *throughput* (ancho de banda efectivo), la latencia, la estabilidad de la conexión y el consumo de recursos del túnel.

El ancho de banda representa la capacidad máxima teórica de una red, es decir, la cantidad total de datos que podría transmitirse por un enlace en condiciones ideales. Es comparable al tamaño de una tubería. Por su parte, el *throughput* o rendimiento corresponde a la cantidad real de datos que efectivamente se transmiten con éxito en un período determinado. Este valor se ve afectado por factores como congestión, latencia, pérdidas de paquetes o errores de transmisión, y equivale al caudal real de agua que fluye por la tubería. En síntesis, el ancho de banda es el potencial de transmisión, mientras que el *throughput* refleja el resultado práctico alcanzado bajo condiciones reales de operación.

Para ello, se emplean herramientas estándar:

- *ping* para medir la latencia entre los extremos del túnel,
- *iperf* para evaluar el *throughput* en protocolos TCP o UDP,
- pruebas de velocidad, y
- supervisión continua mediante SNMP, MRTG u otras soluciones de monitoreo.

Es recomendable probar el comportamiento de la VPN bajo diferentes condiciones de carga y utilizando distintos algoritmos de cifrado (por ejemplo, AES-256 frente a ChaCha20), así como probar

modos de transporte como TCP o UDP. Estas variables influyen directamente en la velocidad y la eficiencia, ya que tanto el cifrado como la compresión introducen sobrecarga. En este sentido, WireGuard suele ofrecer mayor rendimiento que OpenVPN, especialmente cuando se utiliza con UDP.

También se sugiere monitorear el uso de CPU en el servidor VPN, dado que el proceso de cifrado puede representar una carga significativa. En equipos con recursos limitados, este aspecto puede convertirse en un factor restrictivo para el desempeño general.

Para entornos que requieren alta disponibilidad, es posible implementar mecanismos de conmutación por error (*fallback*). Esto permite que, en caso de falla de la VPN principal o del enlace WAN, el tráfico se redirija automáticamente a una ruta alternativa.

Este tipo de configuración puede lograrse mediante esquemas *multi-WAN* en *firewalls* como pfSense u OPNsense, que admiten múltiples *gateways* y conexiones VPN redundantes. También es posible emplear grupos de *gateways* VPN, donde se alterna entre rutas según su disponibilidad.

En conexiones sitio a sitio, se puede aplicar enrutamiento dinámico (OSPF o BGP) entre *routers* o *firewalls* para permitir la redirección automática en caso de caída del túnel principal. Un ejemplo de alta disponibilidad es el uso de dos proveedores de internet con túneles

IPsec redundantes y la activación del protocolo CARP en pfSense, lo que permite una conmutación automática efectiva.

En el caso de acceso remoto, los *clientes* pueden configurarse con múltiples servidores VPN —por ejemplo, uno principal ubicado en la oficina central y otro secundario en una sucursal—, de modo que, ante una falla del primero, se intente conexión con el siguiente.

En entornos móviles, también se puede implementar un *kill-switch*, que bloquea el acceso a internet si la conexión VPN se interrumpe, evitando así filtraciones accidentales de tráfico sin cifrar.

Además de monitorear el rendimiento general, es recomendable analizar indicadores específicos como los registros de renegociación de claves (rekey), los eventos de caída de túnel y los tiempos de inactividad. El ajuste de parámetros como *keepalive* y los intervalos de retransmisión puede ayudar a minimizar interrupciones.

Como ejemplo práctico, se puede montar una configuración con pfSense utilizando dos enlaces WAN —uno primario y otro secundario— y establecer dos túneles IPsec hacia la oficina central. Al definir uno como activo y otro en espera (*takeover*), se puede simular la caída del enlace principal y verificar que el tráfico migra automáticamente al túnel alternativo.

Para completar esta estrategia, es recomendable implementar herramientas de supervisión, como Zabbix o las opciones de

observabilidad integradas en pfSense, con el fin de generar alertas ante caídas de túneles o degradación del servicio, ya sea por pérdida de paquetes o aumento de *jitter*.

Actividades prácticas complementarias

Estas actividades, que no son obligatorias, permiten reforzar los contenidos trabajados. Se proponen prácticas tanto en entornos virtuales como físicos:

- **Laboratorio VPN sitio a sitio**. En VirtualBox o GNS3, montar dos *routers* virtuales (por ejemplo, pfSense), cada uno conectado a su red LAN correspondiente. Configurar un túnel IPsec IKEv2 entre ambos, con clave precompartida o certificados. Verificar la conectividad entre subredes mediante *ping*. Medir el *throughput* con *iperf*. Luego, simular una falla del proveedor de internet en uno de los extremos y comprobar la conmutación (*fallback*) utilizando *multi-WAN* o enrutamiento dinámico.

- **Laboratorio OpenVPN para acceso remoto**: instalar un servidor OpenVPN en pfSense en modo UDP con autenticación TLS. Generar una autoridad certificadora (CA) y certificados con *easy-rsa*. Configurar un *cliente* OpenVPN en Linux o Windows con perfil basado en

certificado y usuario. Establecer la conexión desde un *cliente* externo (simulando acceso desde internet) y verificar el acceso a la red LAN. Medir la velocidad de descarga a través del túnel. Activar el modo túnel dividido (*split-tunnel*), configurando que solo el tráfico hacia la red corporativa utilice la VPN. Confirmar el acceso mixto: internet local sin cifrar y LAN protegida por la VPN.

- **Pruebas con WireGuard**: configurar un servidor WireGuard (por ejemplo, en Ubuntu) y un *cliente*, intercambiando las claves públicas correspondientes. Comparar el rendimiento de transferencia con la prueba anterior realizada en OpenVPN (utilizando los mismos equipos y iperf). Observar el uso de CPU durante la prueba. Realizar también una conexión desde un dispositivo móvil mediante la aplicación WireGuard en Android.

- **Autenticación y gestión de claves**: utilizar *easy-rsa* para crear una CA y generar certificados para OpenVPN. Habilitar autenticación de dos factores (por ejemplo, mediante *Google Authenticator*) en el servidor OpenVPN. Probar el inicio de sesión con usuario, contraseña y código TOTP. Para IPsec IKEv2, generar certificados para servidor y *cliente* con *strongSwan* y verificar la conexión.

- **Análisis de registros y fallos**: provocar desconexiones, como la baja de una interfaz de red, y revisar los registros de VPN en pfSense (menú «Status > System Logs > IPsec/OpenVPN»). Configurar alertas por correo en caso de caída del túnel. Analizar el comportamiento de la conexión (latencia y *jitter*) mediante *ping* continuo.

- **Actividad en entorno virtual**: utilizar VirtualBox (u otro hipervisor gratuito) para crear un entorno de laboratorio con múltiples máquinas virtuales. Incluir un servidor pfSense u OPNsense, varios *clientes* (Linux o Windows) y un servidor ubicado en una zona desmilitarizada (DMZ). Configurar cortafuegos, VPN y servicios conforme a los contenidos de cada unidad. Documentar todo el proceso con capturas de pantalla y descripciones paso a paso.

- **Actividad en entorno físico**: reutilizar hardware disponible (una computadora antigua o una Raspberry Pi) como cortafuegos o servidor VPN. Por ejemplo, instalar pfSense en un mini-PC con dos interfaces de red y montar una DMZ física. Establecer una conexión OpenVPN entre este cortafuegos y un *cliente* externo. Esta práctica permite aplicar los conocimientos en una red real, sin

necesidad de virtualización ni equipos de alto rendimiento.

Herramientas gratuitas recomendadas

A continuación, se presentan algunas herramientas gratuitas que pueden utilizarse en laboratorios o entornos reales para implementar y reforzar los conceptos trabajados.

- **Cortafuegos (*firewall*)**: pfSense, OPNsense, IPFire, Endian.
- **Sistemas de detección y prevención de intrusiones (IDS/IPS)**: Suricata, Snort, Security Onion.
- **VPN**: OpenVPN, WireGuard, strongSwan/IPsec, SoftEther.
- **Gestión de eventos e información de seguridad (SIEM) y registro de eventos**: Graylog, Wazuh, OSSIM, Elasticsearch junto con Kibana.
- **Laboratorio**: para entornos virtuales, es posible utilizar VirtualBox junto con imágenes gratuitas de pfSense o VyOS. En laboratorios físicos, se pueden emplear *routers* basados en

Linux o equipos dedicados compatibles con soluciones de cortafuegos.

CONTINUAR

Referencias

García, A. (2021). *¿Es recomendable usar el modo DMZ de tu conexión para jugar en consola o PC?* ADL Zone. https://www.adslzone.net/noticias/redes/usar-modo-dmz-conexion-jugar-pc-consola-recomendable/

Referencias bibliográficas de referencia

Achirou. (2024). *Curso gratis de redes #20 – Configuración de firewall*. https://achirou.com/curso-gratis-de-redes-20-configuracion-de-firewall/

Cisco. (s. f.). *Cisco SAFE: Un modelo de seguridad para las redes de las empresas*. https://www.cisco.com/c/dam/global/es_es/assets/docs/safe_wp1.pdf

Fortinet. (s. f.). *Redes DMZ*. https://www.fortinet.com/lat/resources/cyberglossary/what-is-dmz

Fortinet. (s. f.). *Túnel dividido VPN*. https://www.fortinet.com/lat/resources/cyberglossary/vpn-split-tunneling

Incibe. (2020). *¿Qué son y para qué sirven los SIEM, IDS e IPS?* https://www.incibe.es/empresas/blog/son-y-sirven-los-siem-ids-e-ips

Incibe. (2019). *Qué es una DMZ y cómo te puede ayudar a proteger tu empresa.* https://www.incibe.es/empresas/blog/dmz-y-te-puede-ayudar-proteger-tu-empresa

IONOS. (s. f.). *¿Qué es un reverse proxy?* https://www.ionos.com/es-us/digitalguide/servidores/know-how/que-es-un-servidor-proxy-inverso/

LightNode. (s. f.). *WireGuard vs. OpenVPN: Un análisis comparativo detallado*. https://go.lightnode.com/es/tech/wireguard-vs-openvpn

Netgate. (s. f.). *IDS / IPS*. https://docs.netgate.com/pfsense/en/latest/packages/snort/index.html

OpenVPN. (s. f.). *Setting up your own certificate authority (CA) and generating certificates and keys for an OpenVPN server and multiple clients*. https://openvpn.net/community-docs/setting-up-your-own-certificate-authority--ca--and-generating-certificates-and-keys-for-an-openvpn-server-and-multiple-clients.html

Palo Alto Networks. (s. f.). *IPS vs. IDS vs. firewall: ¿Cuáles son las diferencias?* https://www.paloaltonetworks.lat/cyberpedia/firewall-vs-ids-vs-ips

Blanco Cárdenas, H. C., Zamora Méndez, L. A., Cardozo Campos, E. A., & Restrepo Salamanca, B. F. (2020). *Implementación de seguridad perimetral en GNU/Linux mediante firewall Endian en entornos LAN, WAN y DMZ [Trabajo de grado, Universidad Nacional Abierta y a Distancia]*. Repositorio Institucional UNAD. https://repository.unad.edu.co/jspui/bitstream/10596/77203/1/eacardozoca.pdf

Tecnoseguro. (s. f.). *¿Qué es la seguridad perimetral informática? Guía completa*. https://www.tecnoseguro.com/faqs/seguridad-perimetral-informatica-que-es

Trend Micro. (s. f.). *¿Qué es la gestión de eventos e información de seguridad (SIEM)?* https://www.trendmicro.com/es_mx/what-is/security-operations/security-information-and-event-management.html

WireGuard. (s. f.). *WireGuard vs. OpenVPN: ¿Qué protocolo de VPN es mejor?* Surfshark. https://surfshark.com/es/blog/wireguard-vs-openvpn?srsltid=AfmBOoqkgxcYTY8tKadw1Oa0sSUfVbgObah5U2VCxMez8Z6oLBk48YFt

4Geeks. (s. f.). *Herramientas para la prevención o detección de intrusos (IDS/IPS)*. https://4geeks.com/es/lesson/herramientas-para-la-prevencion-o-deteccion-de-intrusos-ids-ips

IPP. (s. f.). *¿Qué es rollback en el sector TI?* https://ipp.cl/tecnologia-y-desarrollo/que-es-rollback-en-el-sector-ti/

CONTINUAR