Redes y sistemas para defender

Las redes son el sistema circulatorio de la información en cualquier organización. Cada correo enviado, cada archivo compartido y cada acceso remoto viajan por protocolos y capas que, si no se configuran correctamente, pueden convertirse en puertas abiertas para atacantes. En las pymes, donde la infraestructura suele ser más simple y los recursos, limitados, los errores básicos en la red son responsables de la mayoría de los incidentes.

Según un informe de INCIBE (2025), el 65 % de los ataques a pymes en Argentina involucraron vulnerabilidades en redes: puertos abiertos, servicios sin cifrado, contraseñas débiles en *routers* y falta de segmentación.
Este módulo tiene como objetivo que el alumno comprenda cómo funciona la red, identifique puntos críticos de seguridad y aplique controles básicos para reducir riesgos.

Unidad 1. Pila TCP/IP y Modelo OSI para seguridad

Tanto el modelo OSI como la pila TCP/IP están relacionados con las comunicaciones entre computadores heterogéneos a través de una o varias redes. Estas comunicaciones pueden implicar que los sistemas involucrados no pertenezcan a la misma red; en ese caso, los datos transferidos deben atravesar al menos dos redes, que incluso pueden ser bastante diferentes entre sí.

Ambos modelos se basan en el concepto de capas y protocolos, y comparten muchas similitudes. Sin embargo, existen diferencias tanto filosóficas como físicas entre el modelo OSI y la pila TCP/IP. Por eso, los profesionales de la ciberseguridad deben comprender ambos enfoques.

El modelo de referencia OSI

El modelo de referencia OSI tiene siete capas, que se enumeran a continuación junto con un breve comentario sobre las funciones que debe cumplir un sistema para poder comunicarse. Como se necesitan dos sistemas para establecer una comunicación, el mismo conjunto de capas funcionales debe estar presente en ambos. La comunicación se realiza cuando las entidades del mismo nivel (es decir, de la misma capa) en dos sistemas diferentes se comunican mediante un protocolo.

Capa física

La capa física se encarga de la transmisión de flujos de bits no estructurados a través del medio físico. Aborda las características mecánicas, eléctricas, funcionales y procedurales necesarias para acceder a ese medio.

Esta capa define la interfaz física entre los dispositivos y las reglas que determinan cómo se transfieren los *bits* de un equipo a otro. Sus características se agrupan en cuatro aspectos principales:

- Mecánicos.
- Eléctricos.

- Funcionales.
- Procedurales.

Capa de enlace de datos

Esta capa permite la transferencia confiable de información a través del enlace físico. Envía bloques de datos (paquetes o *frames*) con la sincronización necesaria, control de errores y control de flujo.

Mientras que la capa física ofrece un servicio básico de transmisión de bits, la capa de enlace de datos busca garantizar la confiabilidad del enlace. Para ello, proporciona los medios necesarios para activar, mantener y desactivar dicho enlace. Su principal función, en relación con las capas superiores, es la detección y control de errores. De este modo, la capa superior inmediata puede asumir que la transmisión está libre de errores.

Capa de red

La capa de red ofrece a las capas superiores independencia respecto de la transmisión de datos y de las tecnologías de conmutación utilizadas para conectar los sistemas. Es responsable del establecimiento, mantenimiento y finalización de las conexiones.

Esta capa facilita la transferencia de información entre sistemas terminales a través de alguna red de comunicaciones. Su función es abstraer a las capas superiores de los detalles relacionados con la transmisión subyacente y las tecnologías de conmutación. En este nivel, el sistema de computación interactúa con la red para indicar la dirección de destino y solicitar ciertos recursos, como la asignación de prioridad.

Las comunicaciones entre procesos en sistemas diferentes, a través de una red, pueden comprenderse como la interacción entre dos tipos de entidades:

- Los sistemas terminales (*end systems*), que contienen los procesos e implementan alguna arquitectura de comunicación, como el modelo OSI de siete capas.
- Los nodos o sistemas intermedios, como los sistemas de conmutación (*switching systems*) y los enrutadores (*routers*), que gestionan la comunicación a lo largo de la red.

Capa de transporte

La capa de transporte proporciona una transferencia de datos confiable y transparente entre los puntos terminales, además de recuperación de errores y control de flujo entre esos mismos puntos.

Su propósito principal es ofrecer un mecanismo seguro para el intercambio de datos entre procesos que se ejecutan en sistemas distintos. Garantiza que las unidades de datos se entreguen sin errores, en el orden correcto y sin pérdidas ni duplicaciones. También puede encargarse de optimizar el uso de los servicios de red y de ofrecer una cierta calidad de servicio a las entidades usuarias, como una tasa de error aceptable, un retardo máximo, o un determinado nivel de prioridad y seguridad.

El tamaño y la complejidad del protocolo de transporte dependen del tipo de servicio que proporcione la capa de red (capa 3). Si esta capa es confiable y permite circuitos virtuales, la capa de transporte puede ser más simple. En cambio, si la capa de red no es confiable o solo ofrece datagramas, la capa de transporte deberá incorporar funciones de detección y recuperación de errores.

Entre otras funciones, esta capa se encarga de segmentar y reensamblar los paquetes que contienen la información, así como del reenvío de aquellos que se pierdan durante la transmisión.
En la capa de transporte se implementan los protocolos TCP y UDP:

- **TCP** ***(Transmission Control Protocol)***. Protocolo orientado a la conexión, con control de flujo y de errores. Garantiza la entrega de los paquetes mediante acuse de recibo. Permite distinguir múltiples aplicaciones en un mismo nodo mediante el uso de puertos. Cada proceso de una aplicación se identifica por una dirección IP y un número de puerto.
- **UDP** ***(User Datagram Protocol)*****:** protocolo sin conexión, sin control de errores ni acuse de recibo. La fiabilidad de la transmisión depende de las aplicaciones que lo utilicen. Este protocolo es habitual en servicios que requieren inmediatez por sobre fiabilidad, como la transmisión en red de contenidos de radio o televisión.

Capa de sesión —

La capa de sesión proporciona la estructura de control para la comunicación entre aplicaciones. Se encarga de establecer, gestionar y finalizar las conexiones (o sesiones) entre ellas.

Las cuatro capas inferiores del modelo OSI permiten el intercambio confiable de datos y brindan un servicio de datos básico. Sin embargo, para muchas aplicaciones, este servicio resulta insuficiente. Por ejemplo:

- Una aplicación de acceso a un terminal remoto puede requerir un diálogo en modo *half-duplex*.
- Un proceso de transacción puede necesitar puntos de verificación (*checkpoints*) en el flujo de datos, que permitan realizar copias de respaldo en disco (*backup*) y llevar a cabo una recuperación (*recovery*) si fuera necesario.
- Otra aplicación puede requerir la capacidad de interrumpir un diálogo para preparar una nueva porción de mensaje y luego retomarlo desde donde se había suspendido.

Si bien estas funciones pueden estar implementadas dentro de aplicaciones específicas en la capa 7, dado que tienen un uso amplio, se agrupan en una capa independiente: la capa de sesión. Los principales servicios que esta capa ofrece son los siguientes:

- **Disciplina de diálogo**: permite establecer comunicaciones en dos vías simultáneas (*full-duplex*) o en dos vías alternadas (*half-duplex*).
- **Agrupamiento**: posibilita la señalización del flujo de datos para definir conjuntos o grupos de información.
- **Recuperación**: ofrece un mecanismo de *checkpoints* que permite, en caso de fallo, retransmitir todos los datos desde el último punto de verificación.

La norma ISO/IEC 7498-1 establece los servicios de la capa de sesión como opciones dentro del modelo de referencia OSI.

Capa de presentación

La capa de presentación aísla los procesos de aplicación de las diferencias en la sintaxis de los datos. Su función es gestionar la forma en que se representan los datos intercambiados entre entidades de aplicación. Se encarga de resolver las diferencias de formato y representación, mediante la definición de una sintaxis común y la transformación del formato según sea necesario.

Entre los protocolos asociados a esta capa se encuentran los de codificación (*encryption*) y los protocolos de terminal virtual. Un protocolo de terminal virtual convierte las características específicas de distintos terminales en un modelo genérico o virtual, que puede ser interpretado por los programas de aplicación.

Capa de aplicación

La capa de aplicación proporciona el acceso de los usuarios al entorno OSI y ofrece servicios de información distribuida.

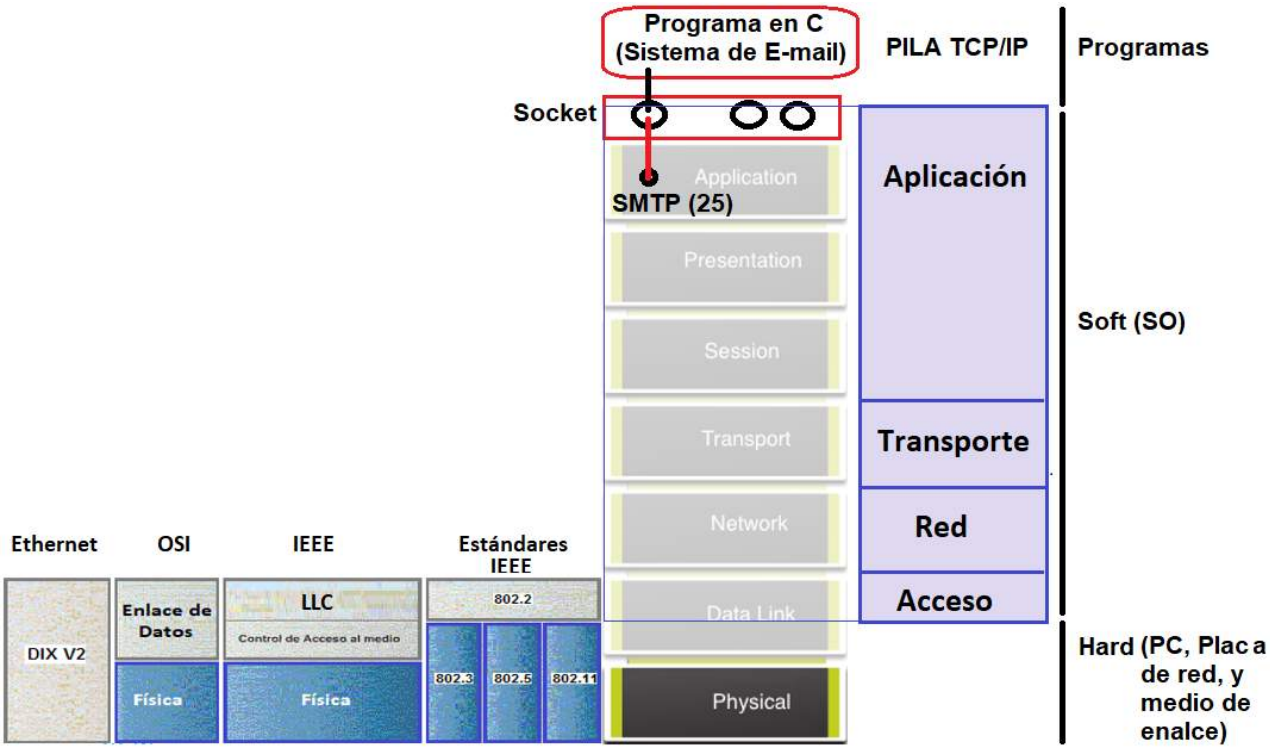
Su función es permitir que los procesos de aplicación accedan a los servicios del modelo OSI. Incluye funciones de administración y mecanismos de uso general que dan soporte a las aplicaciones distribuidas. Ejemplos de protocolos que operan en este nivel son la transferencia de archivos y el correo electrónico.

En la siguiente figura se muestra la correspondencia entre las capas de los modelos OSI y TCP/IP. Se observa que la capa física no forma parte explícita del modelo TCP/IP. Esto se debe a que la pila TCP/IP no se basa en un estándar formal, sino que representa la manera en que la industria implementa los sistemas en la práctica.

A la derecha, se establece la relación entre las capas del modelo y los componentes de un sistema computacional típico: programas (*software*), sistema operativo (*software*) y *hardware*. Por otro lado, a la izquierda, se representa la correspondencia con las implementaciones de las capas inferiores. Ethernet, la tecnología cableada más antigua y ampliamente implementada, abarca toda la parte baja.

El estándar IEEE divide la capa de enlace de datos del modelo OSI en dos subcapas: la capa de control de acceso al medio (MAC) y la capa de control de enlace lógico (LLC). También se indican los estándares IEEE correspondientes a tecnologías de red como Ethernet (802.3), wifi (802.11) y Token Ring (802.5), aunque esta última ya no se utiliza.

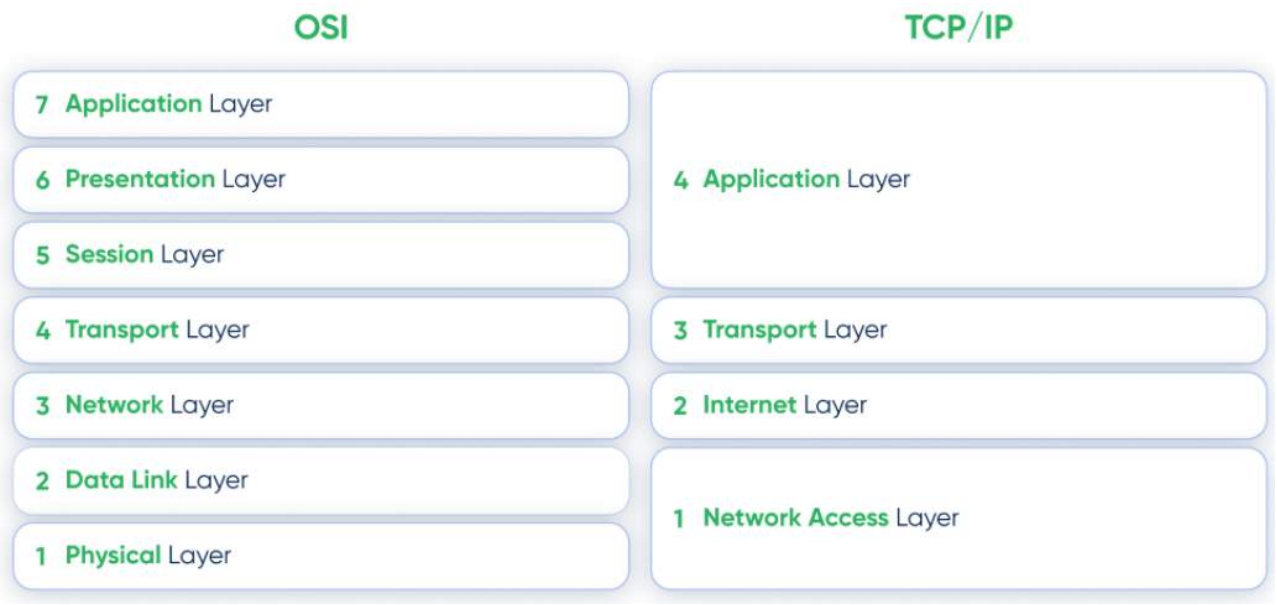
Figura 1. Correspondencia entre las capas de los modelos OSI y TCP/IP, las tecnologías de red y los componentes de un sistema computacional



Fuente: Elaboración propia.

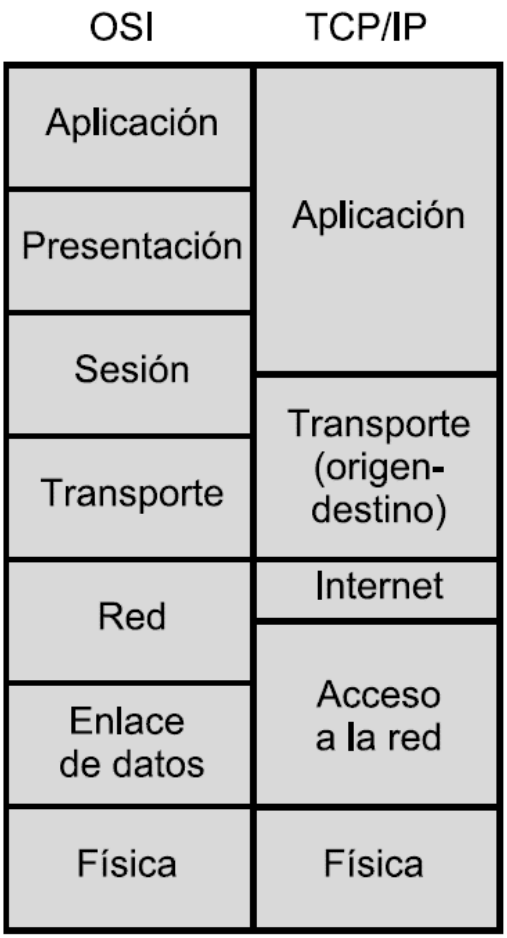
A continuación, se presentan distintas representaciones de la pila TCP/IP, que muestran algunas variaciones en su interpretación. A pesar de estas diferencias, la estructura del modelo OSI se mantiene invariable en todas ellas, ya que se trata de un estándar formal definido por la norma ISO/IEC 7498-1.

Figura 2. Comparación entre las arquitecturas de protocolos TCP/IP y OSI



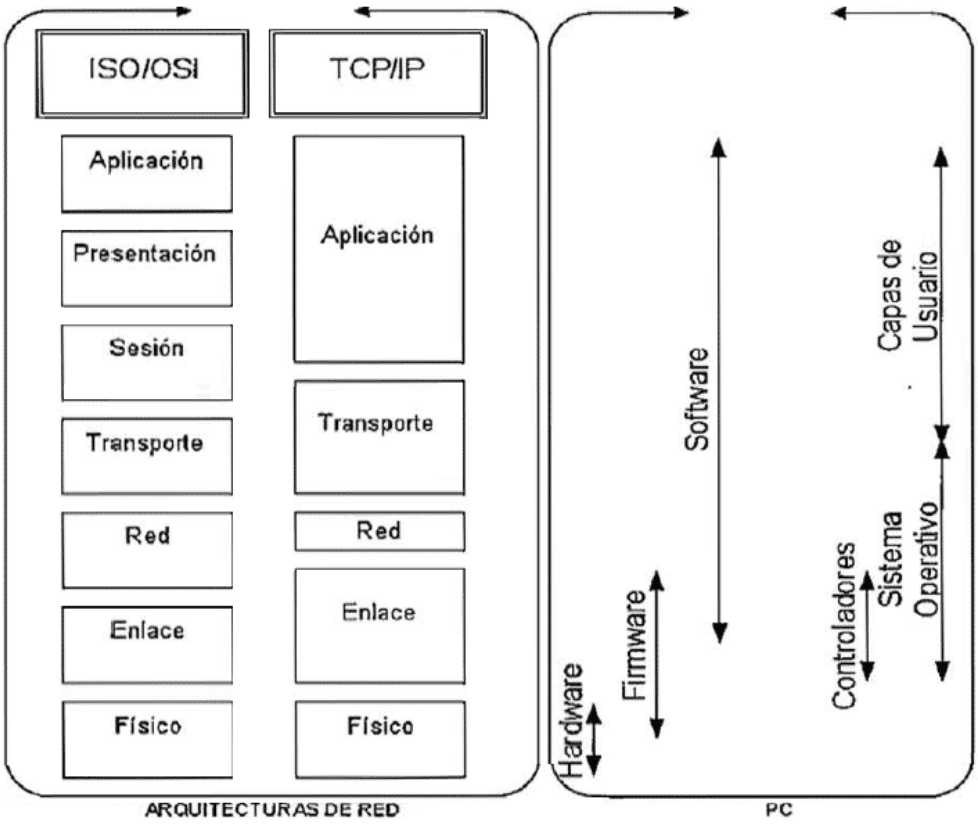
Fuente: SormWall, s.f., https://goo.su/JNcUf

Figura 3. Comparación entre las arquitecturas de protocolos TCP/IP y OSI



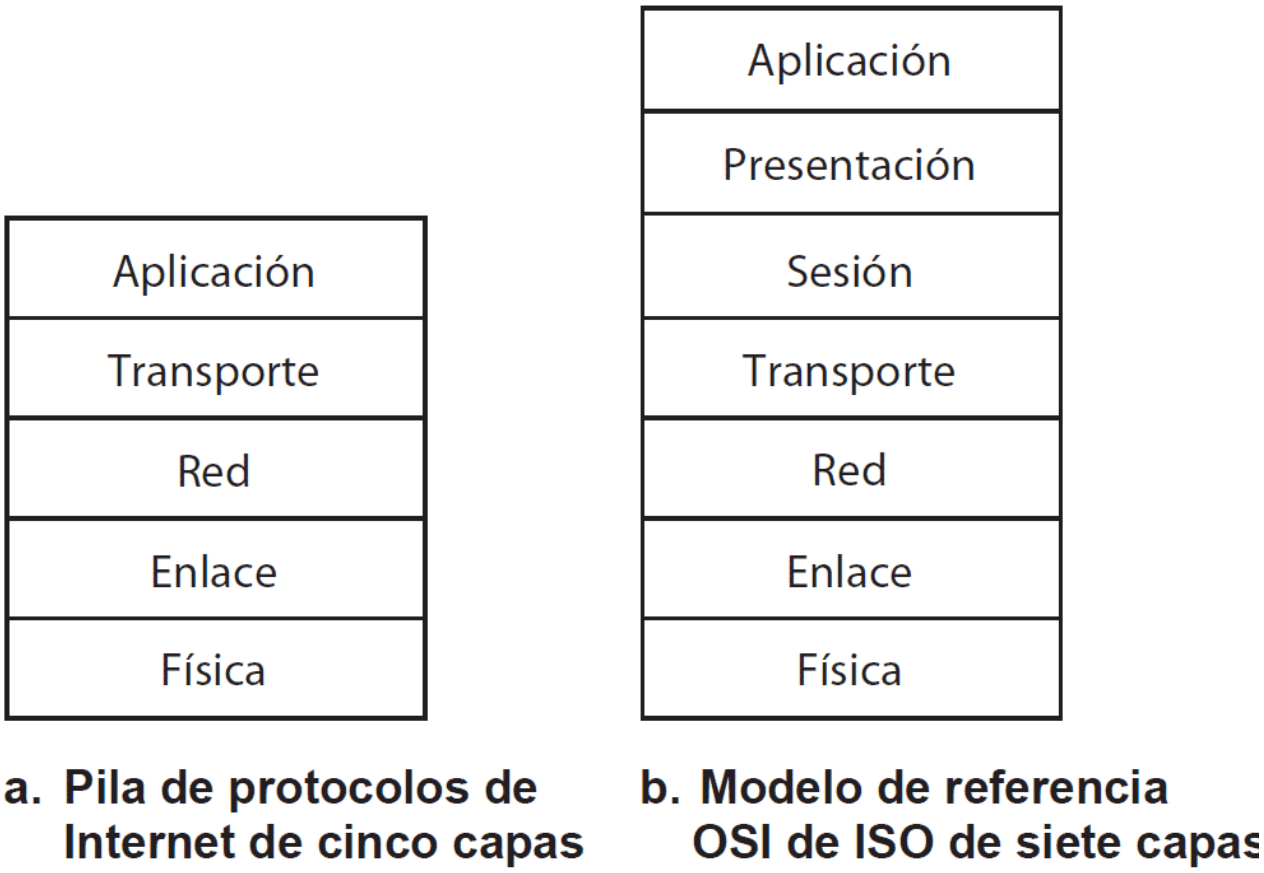
Fuente: Stallings, 2000, p. 15.

Figura 4. Comparación entre las arquitecturas de protocolos TCP/IP y OSI



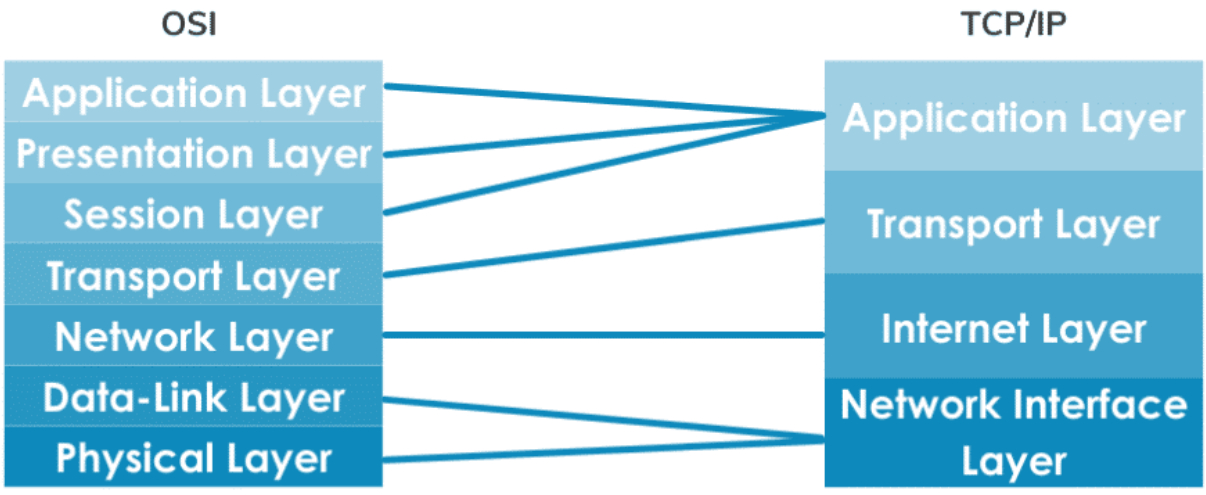
Fuente: Gil et al., 2010, p. 23.

Figura 5. Comparación entre las arquitecturas de protocolos TCP/IP y OSI



Fuente: Kurose y Ross, 2017, p. 42.

Figura 6. Comparación entre las arquitecturas de protocolos TCP/IP y OSI



Fuente: Tech Buyer, s.f., https://goo.su/CCCNxPn

Figura 7. Comparación entre las arquitecturas de protocolos TCP/IP y OSI

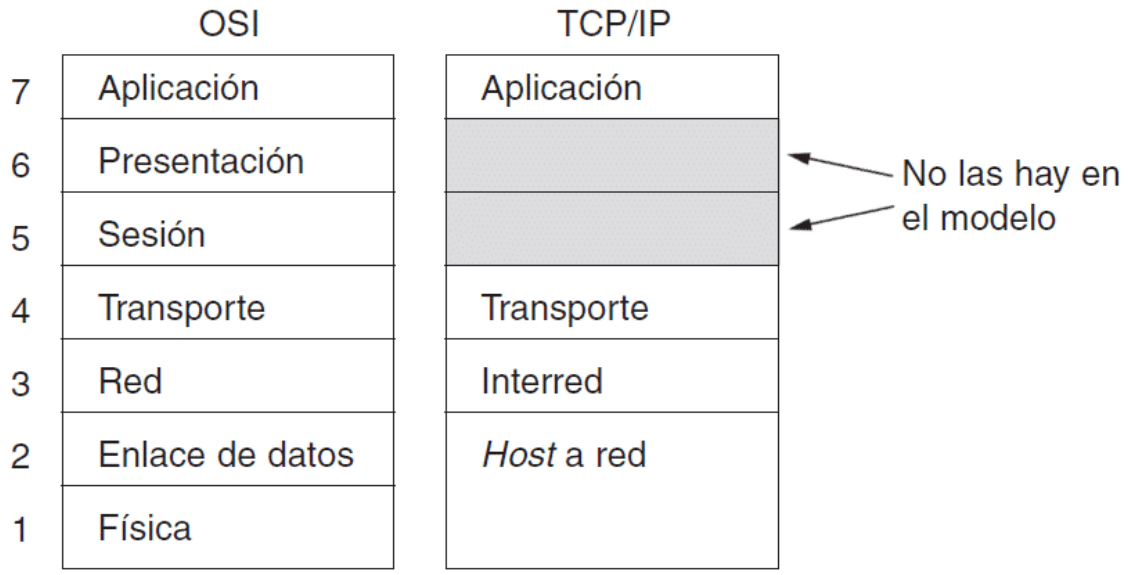


Figura: Tanenbaum, 2003, p. 43.

Figura 8. Comparación entre las arquitecturas de protocolos TCP/IP y OSI

OSI Layer	*Devices Found*	*Protocols/Standards working in the Layer*	TCP/IP Layer
7-Application	Firewall, Gateway	SMTP, POP3, IMAP, DNS, DHCP, FTP, HTTP, TFTP, SNMP, VoIP, NNTP, NTP	Application
6-Presentation	N/A	JPEG, JPG, TIFF, PNG, GIF, MIME, MP3, MP4	
5-Session	N/A	SQL, NFS, ASP, RPC	
4-Transport	Firewall	TCP, UDP	Transport
3-Network	Router	IP	Internet
2-Data Link	Switch, Bridge	Ethernet, PPP, HDLC, Frame Relay, ATM	Network Access
1-Physical	Hub, Repeater, Transciever	RJ45, ST/SC, V series (modem standards)	

Fuente: Cisco, s.f., https://goo.su/pwbU8

La pila TCP/IP

La pila TCP/IP es el modelo estándar de facto para las comunicaciones en red. Es el más utilizado y abarca casi todos los servicios comunes, como el acceso remoto (*telnet*), la navegación web, el correo electrónico, la transferencia de archivos (FTP), entre otros. Todos los dispositivos que usamos para conectarnos a una red utilizan este modelo. El modelo OSI, en cambio, es un modelo de referencia principalmente teórico, que rara vez se implementa en la práctica, salvo en sistemas muy específicos.

Las siglas TCP/IP provienen de *Transmission Control Protocol* (protocolo de control de transmisión) e *Internet Protocol* (protocolo de internet), es decir, los dos protocolos más importantes de esta pila. No obstante, el término «pila TCP/IP» se utiliza para referirse al modelo completo, que consta de cuatro capas.

Este modelo define cómo se comunican los dispositivos en internet. Está compuesto por cuatro capas principales, cada una con funciones específicas. La información viaja a través de estas capas en un orden determinado al enviarse, y en orden inverso al recibirse.

Las cuatro capas del modelo TCP/IP son las siguientes:

CAPA DE ACCESO A LA RED	CAPA DE INTERNET	CAPA DE TRANSPORTE	CAPA DE APLICACIÓN

Es la capa inferior del modelo TCP/IP. Se encarga de gestionar la comunicación dentro de una misma red local y se relaciona directamente con la conexión física. Su función incluye el direccionamiento físico (como la dirección MAC) y aspectos del hardware, como los estándares de Ethernet, wifi, entre otros. En el modelo OSI, esta capa equivale a la combinación de las capas física y de enlace de datos.

CAPA DE ACCESO A LA RED	CAPA DE INTERNET	CAPA DE TRANSPORTE	CAPA DE APLICACIÓN

Se ocupa del direccionamiento y del enrutamiento de los paquetes de datos para garantizar que lleguen correctamente a su destino, incluso al atravesar múltiples redes. El protocolo principal de esta capa es el protocolo de internet (IP), que asigna una dirección lógica (dirección IP) a cada dispositivo.

CAPA DE ACCESO A LA RED	CAPA DE INTERNET	CAPA DE TRANSPORTE	CAPA DE APLICACIÓN

Es responsable de la comunicación fiable y orientada a la conexión entre dispositivos ubicados en redes diferentes. Esta capa segmenta los datos en paquetes, los numera y los reensambla en el destino. Los principales protocolos utilizados son el protocolo de control de transmisión (TCP), que proporciona una entrega ordenada y segura, y el protocolo de datagramas de usuario (UDP), que ofrece una transmisión más rápida, aunque menos fiable.

CAPA DE ACCESO A LA RED	CAPA DE INTERNET	CAPA DE TRANSPORTE	CAPA DE APLICACIÓN

Es la capa superior del modelo y se encarga de la interacción directa con las aplicaciones de software y los usuarios. Proporciona servicios como la navegación web (HTTP), la transferencia de archivos (FTP) y el correo electrónico (SMTP). A diferencia del modelo OSI, la capa de aplicación en el modelo TCP/IP también abarca las funciones que, en el modelo OSI, corresponden a las capas de sesión y presentación.

Direccionamiento y subredes

En ciberseguridad, comprender el direccionamiento IP y el uso de subredes es fundamental para proteger las redes y segmentar el tráfico según roles o niveles de riesgo. Una segmentación deficiente facilita la propagación de amenazas, como *ransomware* o accesos no autorizados. Esta sección introduce los conceptos básicos de direccionamiento, la diferencia entre redes públicas y privadas, y cómo las subredes contribuyen a mejorar la visibilidad y el control dentro de la red.

IP son las siglas de **protocolo de internet** (*Internet protocol*), el conjunto de reglas que permite la comunicación entre dispositivos a través de internet. Miles de millones de personas acceden diariamente a la red pública, por lo que se requiere un identificador único para registrar quién hace qué. El protocolo de internet resuelve esta necesidad mediante la asignación de direcciones IP a todos los dispositivos que se conectan a la red.

La dirección IP de un ordenador funciona como la dirección de una casa. Si alguien llama a una pizzería para hacer un pedido, necesita proporcionar su dirección. Sin ella, el repartidor no sabría adónde entregar la pizza.

Direcciones IP: IPv4 vs IPv6

Las direcciones IP permiten identificar de forma única a cada dispositivo dentro de una red. La versión más utilizada actualmente es IPv4, que emplea direcciones de 32 bits representadas en formato decimal con puntos, como por ejemplo: 192.168.0.1. Sin embargo, debido a la escasez de direcciones disponibles, comenzó a adoptarse IPv6, que utiliza direcciones de 128 bits expresadas en hexadecimal, como 2001:0db8:85a3::8a2e:0370:7334.

IPv4 e IPv6 son versiones diferentes del protocolo de internet. IPv4 fue implementado en 1983 y todavía está en uso. Su formato se compone de cuatro conjuntos de números separados por puntos (por ejemplo, 192.0.2.1) y permite un total de 2^{32} direcciones únicas, es decir, aproximadamente 4.300 millones. Aunque parecía suficiente en su momento, el crecimiento explosivo de dispositivos conectados demostró lo contrario.

La necesidad de ampliar ese espacio llevó a la creación de IPv6. Esta versión utiliza un formato más extenso, con números y letras separados por dos puntos (simples o dobles), como: 2001:0db8:85a3:0000:0000:8a2e:0370:7334. Al ser de 128 bits, permite generar 2^{128} direcciones únicas, lo que representa un número extremadamente grande (de 39 dígitos).

Además de ampliar el espacio disponible, IPv6 incorpora mejoras en seguridad, privacidad y eficiencia. A pesar de sus diferencias, ambas versiones conviven desde hace más de una década. Para lograr la interoperabilidad, fue necesario implementar mecanismos de compatibilidad, ya que gran parte de la infraestructura de internet aún funciona con direcciones IPv4.

¿Qué pasó con IPv5?

IPv5 fue un protocolo experimental diseñado para la transmisión de datos en tiempo real, pero nunca llegó a implementarse de forma general. Usaba también direcciones de 32 bits, por lo que no resolvía el problema de escasez de direcciones. Por esta razón, IPv6 fue desarrollado como el verdadero sucesor de IPv4.

Generalidades de IPv4

El direccionamiento IP representa la dirección lógica de un *host* y se utiliza para identificarlo dentro de una red. En el caso de IPv4, cada dirección está compuesta por 32 bits, es decir, cuatro bytes. Sin embargo, estas direcciones se expresan habitualmente en notación decimal con puntos, donde cada byte se representa como un número decimal separado por un punto, por ejemplo: 192.132.234.102.

Cada uno de estos bytes se denomina con frecuencia **octeto**, por lo que una dirección IP consta de cuatro octetos, cuyos valores posibles van de 0 a 255.

LAS DIRECCIONES IP SE DIVIDEN PRINCIPALMENTE EN DOS PARTES:

- **Porción de red**: identifica al conjunto de *hosts* que pertenecen a la misma red.
- **Porción de *host***: identifica a un dispositivo específico dentro de esa red.

Todos los dispositivos de una misma red lógica deben compartir la misma dirección de red, pero cada uno debe tener un identificador de *host* único.

Una analogía útil para entender esta estructura es la de los números telefónicos: una parte del número —el prefijo— identifica una región o país (equivalente a la red), mientras que la otra parte distingue a cada línea individual (equivalente al host). El prefijo es común a todos los números de una misma zona, pero cada teléfono tiene un número único.

Direcciones IP estáticas y las dinámicas

La disponibilidad limitada de direcciones IPv4 impulsó el uso de la asignación dinámica de direcciones IP, una práctica que sigue siendo muy común. La mayoría de los dispositivos conectados a internet reciben direcciones IP temporales.

LAS DIRECCIONES IP PUEDEN ASIGNARSE A CADA HOST DE DOS MANERAS:

- **IP estáticas**. Se configuran manualmente por el administrador de red.
- **IP dinámicas**: se asignan automáticamente mediante un servicio llamado DHCP (*Dynamic Host Configuration Protocol*).

Por ejemplo, cuando un usuario se conecta a internet desde su computadora portátil, su proveedor de servicios de internet (ISP) le asigna una dirección IP temporal tomada de un conjunto compartido. Esto se conoce como dirección IP dinámica. Para el ISP, resulta más rentable utilizar este método que asignar una dirección IP permanente (o estática) a cada usuario.

Clases de direcciones IP

Como se ha explicado, las direcciones IP se dividen en dos partes: una porción destinada a identificar la red y otra asignada al *host* dentro de esa red.

Para facilitar la administración y optimizar el uso del espacio de direccionamiento, se definieron cinco clases de direcciones IP: A, B, C, D y E. Sin embargo, solo las clases A, B y C se utilizan con fines comerciales. La siguiente tabla muestra cómo se distribuyen los bits entre la porción de red y la de *host* en cada una de estas tres clases:

Tabla 1. Distribución de los bits en las clases de direcciones IP A, B y C

	1 byte	*1 byte*	*1 byte*	*1 byte*
	8 bits	*8 bits*	*8 bits*	*8 bits*
Clase A	Red	Host	Host	Host
Clase B	Red	Red	Host	Host
Clase C	Red	Red	Red	Host

Fuente: elaboración propia.

Redes clase A —

Las redes clase A utilizan el primer octeto para identificar la red y los tres octetos restantes para identificar los hosts dentro de esa red. El primer bit del primer octeto siempre es 0, por lo que las direcciones de red de clase A abarcan el rango de 0 a 127. Sin embargo, las direcciones 0 y 127 están reservadas para fines especiales, por lo tanto, el número total de redes clase A disponibles es 126.

- **Direcciones de red clase A**

Existen 126 redes clase A. Son aquellas en las que el primer octeto (expresado en decimal) está en el rango de 1 a 126. Por ejemplo:

- 1.x.x.x
- 9.x.x.x
- 10.x.x.x
- 11.x.x.x
- ...
- 126.x.x.x

La red que comienza con 10 es un caso particular: se encuentra dentro del rango reservado para uso privado, lo que significa que puede utilizarse para el direccionamiento interno en organizaciones, sin necesidad de coordinación con autoridades externas.

- **Direcciones de *host* clase A**

Cada red clase A permite más de 16 millones de hosts, en concreto: 224−2=16.777.2142^{24} - 2 = 16.777.214224−2=16.777.214 (se restan dos direcciones: una para la dirección de red —todos los bits en 0— y otra para la de broadcast —todos los bits en 1—). Algunos ejemplos de direcciones posibles para hosts en una red clase A son los siguientes:

- (id de red).0.0.1
- (id de red).0.0.254
- (id de red).0.1.1
- (id de red).0.255.254
- (id de red).1.0.1
- ...
- (id de red).255.255.254

Redes clase B

Las redes clase B emplean los dos primeros octetos para identificar la red y los dos restantes para identificar los *hosts* dentro de esa red. En este caso, los primeros dos bits del primer octeto están fijados en 1 y 0, lo que determina que el rango de direcciones para redes clase B va de 128.0.0.0 a 191.255.0.0.
Este formato permite definir hasta 16.384 redes clase B. Cada una de estas redes puede alojar hasta 65.534 *hosts*.

- **Direcciones de red clase B**

Corresponden a direcciones en las que el primer octeto se encuentra entre 128 y 191. A continuación, se presentan algunos ejemplos:

- 128.0.x.x
- 128.255.x.x
- 129.0.x.x
- ...
- 191.255.x.x

- **Direcciones de *host* clase B**

Cada red clase B tiene $2^{16} - 2 = 65.534$ direcciones posibles para hosts (se restan la dirección de red y la de *broadcast*). Algunos ejemplos de direcciones válidas de *host* dentro de una red clase B son los siguientes:

- (id de red).0.1

- (id de red).0.254
- (id de red).1.1
- ...
- (id de red).255.254

Redes clase C —

Las redes clase C utilizan los tres primeros octetos para identificar la red, y el cuarto octeto para identificar a los *hosts* dentro de esa red. Los tres primeros bits del primer octeto están fijados en 1, 1 y 0, lo que define un rango de direcciones que va desde 192.0.0.0 hasta 223.255.255.0.

- **Direcciones de red clase C**

Corresponden a direcciones en las que el primer octeto está en el rango de 192 a 223. Ejemplos:

- 192.0.0.x
- 192.0.255.x
- 192.1.0.x
- 192.167.255.x
- 192.169.0.x
- 193.0.0.x
- 194.1.0.x
- ...
- 223.255.255.x

- **Direcciones de *host* clase C**

Cada red clase C permite hasta 254 hosts, ya que se utilizan 8 bits para esta porción y se excluyen las direcciones reservadas (todos ceros para la red y todos unos para broadcast). Algunos ejemplos de direcciones válidas son los siguientes:

- (id de red).1
- (id de red).2
- ...
- (id de red).254

Direcciones de red clase D y E

Estas clases de dirección están reservadas para *multicast* (envío de información desde un origen hacia varios destinos) y fines investigativos. No son utilizadas comercialmente para la asignación de direcciones a los *hosts*.

Direcciones reservadas y privadas

Dentro de los rangos definidos para cada clase de direcciones IP, existen subrangos especiales que no se utilizan para el direccionamiento público. Estos rangos pueden estar **reservados** para fines específicos o ser designados como **privados**, lo que significa que se emplean únicamente en redes internas y no son válidos para su uso en internet.

DIRECCIONES RESERVADAS | DIRECCIONES PRIVADAS

En una dirección IP, el campo correspondiente a los *hosts* nunca debe estar compuesto solo por ceros cuando se asigna a un dispositivo individual. En cambio, una dirección con todos los bits del campo de *hosts* en 0 se utiliza para identificar la red en sí. Por ejemplo: 11.0.0.0 hace referencia a la red clase A identificada por el número 11.

El direccionamiento IP también incluye una dirección especial llamada **dirección de difusión** o *broadcast*, que permite enviar un mensaje a todos los *hosts* de una red. Por convención, esta dirección se representa con todos los bits del campo de *hosts* en 1 (su equivalente en decimal es 255 en cada octeto correspondiente). Por ejemplo, la dirección 11.255.255.255 alcanza todos los *hosts* de la red 11.0.0.0.

Además, existe la dirección **255.255.255.255**, conocida como *broadcast* de red local. Esta dirección se utiliza para enviar mensajes a todos los dispositivos de la red local y, por norma, los *routers* no reenvían estos paquetes a otras redes.

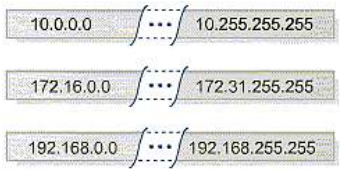
Otro rango reservado es 127.0.0.0, que no puede utilizarse para el direccionamiento convencional, ya que está destinado al ***loopback***. Este mecanismo permite realizar pruebas internas del protocolo TCP/IP y facilita la comunicación entre procesos dentro de una misma máquina. Cuando un programa envía datos a una dirección de *loopback* (por ejemplo, 127.0.0.1), el sistema operativo intercepta esos paquetes y los devuelve localmente, sin que se genere tráfico en la red.

Dentro de cada clase de direcciones IP existen ciertos rangos que no están asignados para uso público. Estas se conocen como **direcciones privadas** y se utilizan comúnmente en redes que no están conectadas directamente a internet, o en aquellas en las que no hay suficientes direcciones públicas disponibles.

Es importante tener en cuenta que, si una red privada necesita conectarse a internet, las direcciones privadas deben ser traducidas a direcciones públicas mediante mecanismos como NAT (*Network Address Translation*). Esto se debe a que cualquier paquete con una dirección de destino incluida en los rangos de direcciones privadas **no será enrutado a través de internet**.

La siguiente figura muestra los rangos de direcciones reservadas para uso privado en las clases A, B y C:

Figura 9. Rangos de direcciones IP privadas según la clase de red



Subredes

Por razones de flexibilidad, costos y administración, especialmente en organizaciones grandes, suele ser conveniente dividir una red en segmentos más pequeños. Estas divisiones se denominan **subredes**.

El concepto de subred permite fraccionar una red principal en partes más manejables, cada una con su propio rango de direcciones IP. Aunque cada subred funciona como una minired independiente, todas permanecen conectadas al sistema general.

Esta segmentación facilita una asignación más eficiente de direcciones IP, permite agrupar dispositivos de forma lógica (por área o función) y mejora la administración general. Además de reducir la congestión del tráfico, también simplifica la resolución de problemas y mejora la seguridad al contener el flujo de datos dentro de límites definidos.

Por ejemplo, si una empresa cuenta con 500 dispositivos distribuidos en áreas como Recursos Humanos (RR. HH.), Finanzas y Tecnología de la Información (TI), puede crear subredes específicas para cada sector.

ASÍ, PODRÍA ASIGNAR:

- 192.168.1.0/24 para RR. HH.
- 192.168.2.0/24 para Finanzas
- 192.168.3.0/24 para TI

De esta manera, cada departamento opera dentro de su propio segmento de red, lo que mejora el rendimiento, facilita la administración y refuerza la seguridad.

UNA SUBRED OFRECE VARIOS BENEFICIOS IMPORTANTES PARA EL DISEÑO Y LA GESTIÓN DE REDES:

- **Mejora del rendimiento**. Al limitar la comunicación a los dispositivos dentro de la misma subred, se reduce el tráfico de difusión.
- **Aumento de la seguridad**: permite aislar segmentos sensibles, como las áreas de Finanzas o Recursos Humanos, restringiendo el acceso desde otras partes de la red.
- **Simplificación de la solución de problemas**: ante una falla, es más sencillo localizar y diagnosticar el problema si la red está segmentada, en lugar de analizar una única red masiva.
- **Escalabilidad**: facilita el crecimiento planificado, ya que es posible reservar subredes sin utilizar para incorporarlas en el futuro.

Por ejemplo, una universidad puede decidir separar la red wifi de los estudiantes de la del profesorado y de la red administrativa. Al crear subredes independientes para cada grupo —por ejemplo, 10.10.1.0/24 para estudiantes, 10.10.2.0/24 para personal y 10.10.3.0/24 para invitados—, se mejora el control sobre el enrutamiento del tráfico y la asignación de ancho de banda.

Las subredes también pueden formarse tomando bits prestados de la porción correspondiente a los *hosts* dentro de una dirección IP. Esta estrategia permite mantener sin cambios la parte de la dirección que identifica la red base.

El uso de bits adicionales para subredes tiene dos efectos principales: por un lado, aumenta la cantidad de subredes disponibles; por otro, reduce la cantidad de *hosts* que pueden ser direccionados en cada subred.

Al utilizar direccionamiento con subredes, se introduce un nuevo campo dentro de la dirección IP: el campo de subred, formado por los bits prestados, además de los campos de red y *host* ya existentes.

La cantidad mínima de bits que se pueden tomar prestados es 2. Si se utilizara solo un bit, se obtendrían únicamente dos combinaciones posibles (0 y 1); sin embargo, por convención, el valor 0 se reserva para identificar la subred y el valor 1 para referirse a todas las subredes de la red, por lo que no quedarían subredes

utilizables. En cambio, la cantidad máxima de bits prestables será aquella que permita conservar al menos dos direcciones para el campo de *hosts* (una para la dirección de red y otra para *broadcast*).

Por lo tanto, una empresa que cuenta con cuatro áreas internas y dispone de una red privada clase C (por ejemplo, 192.168.0.0/24), puede tomar tres bits del cuarto octeto —el correspondiente a los *hosts*— y destinarlos a la creación de subredes. De esta forma, es posible dividir la red original en ocho subredes, cada una con su propio rango de direcciones, lo que facilita la segmentación lógica y el control del tráfico entre sectores.

Tabla 2. Segmentación de una red clase C en ocho subredes con máscara /27

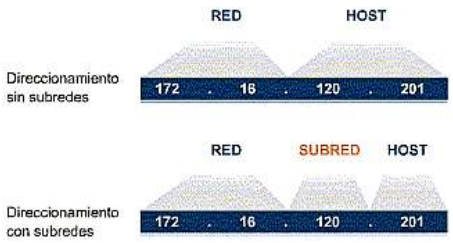
Octeto 1 (red) - Dec	**Octeto 2 (red) - Dec**	**Octeto 3 (red) - Dec**	**Octeto 4 (subredes) - Bin**								**IP de subredes (bits de hosts = 0)**	**Mask**	**IP de Broadcast (bits de hosts = 1)**
			subred			*host*							
192	168	0	**0**	**0**	**0**	0	0	0	0	0	192.168.0.**0**	/27	192.168.0.31
172	16	0	**0**	**0**	**1**	0	0	0	0	0	192.168.0.**32**	/27	192.168.0.63
172	16	0	**0**	**1**	**0**	0	0	0	0	0	192.168.0.**64**	/27	192.168.0.95
172	16	0	**0**	**1**	**1**	0	0	0	0	0	192.168.0.**96**	/27	192.168.0.127
172	16	0	**1**	**0**	**0**	0	0	0	0	0	192.168.0.**128**	/27	192.168.0.159
172	16	0	**1**	**0**	**1**	0	0	0	0	0	192.168.0.**160**	/27	192.168.0.191
172	16	0	**1**	**1**	**0**	0	0	0	0	0	192.168.0.**192**	/27	192.168.0.223
172	16	0	**1**	**1**	**1**	0	0	0	0	0	192.168.0.**224**	/27	192.168.0.255

Fuente: elaboración propia.

Aunque con 3 bits se obtienen 8 combinaciones posibles, solo 6 son utilizables para asignar direcciones a *hosts*, ya que la combinación con todos los bits en cero se reserva para identificar la subred, y la de todos los bits en uno, para la dirección de *broadcast*. Así, cada subred permite direccionar hasta 30 *hosts* válidos.

Un segundo ejemplo corresponde a una empresa mediana en crecimiento que decide segmentar su red interna para aislar sus diferentes áreas funcionales. Parte de una red clase B privada: **172.16.0.0/16**, y con el objetivo de prever futuras expansiones, toma un octeto completo —es decir, ocho bits— de la porción reservada a los *hosts* para utilizarlo como campo de subred. De esta manera, se pueden definir **254 subredes**, cada una con **254 *hosts*** válidos, manteniendo una estructura flexible y escalable para la administración de la red.

Figura 10. Ejemplo de segmentación de una red clase B utilizando un octeto para subredes



Fuente: Elaboración propia.

A continuación, se presenta el diseño del direccionamiento de las subredes resultante de esta estrategia:

Tabla 3. Segmentación de una red clase B en 254 subredes con máscara /24

Octeto 1 (red) - Dec	**Octeto 2 (red) - Dec**	**Octeto 3 (subredes) - Bin**								**Octeto 4 (*host*) - Dec**	**IP de subredes**	**Mask**	**IP de broadcast**
172	16	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	0	172.16.**0**.0	/24	172.16.**0**.255
172	16	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	0	172.16.**1**.0	/24	172.16.**1**. 255
172	16	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	0	172.16.**2**.0	/24	172.16.**2**. 255
172	16	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	0	172.16.**3**.0	/24	172.16.**3**. 255
172	16	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	0	172.16.**4**.0	/24	172.16.**4**. 255

172	16	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	0	172.16.**5**.0	/24	172.16.**5**. 255
172	16	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**	0	172.16.**6**.0	/24	172.16.**6**. 255
172	16	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	0	172.16.**7**.0	/24	172.16.**7**. 255
...	...	**...**	**...**	**...**	**...**	**...**	**...**	**...**	**...**	...	...		...
172	16	**0**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	0	172.16.**120**.0	/24	172.16.**120**. 255
...	...	**...**	**...**	**...**	**...**	**...**	**...**	**...**	**...**	...	...		...
172	16	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	0	172.16.**248**.0	/24	172.16.**248**. 255
172	16	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	0	172.16.**249**.0	/24	172.16.**249**.255
172	16	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**0**	0	172.16.**250**.0	/24	172.16.**250**.255
172	16	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	0	172.16.**251**.0	/24	172.16.**251**.255
172	16	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	0	172.16.**252**.0	/24	172.16.**252**.255
172	16	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	0	172.16.**253**.0	/24	172.16.**253**.255
172	16	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	0	172.16.**254**.0	/24	172.16.**254**.255
172	16	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	172.16.**255**.0	/24	172.16.**255**.255

Fuente: elaboración propia.

Máscara de subred (mask)

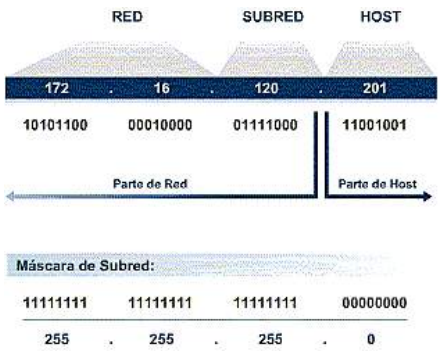
Una máscara de subred tiene una longitud de 32 bits, dividida en cuatro octetos, al igual que una dirección IP. Su función es determinar qué parte de la dirección IP corresponde al identificador de red (o subred) y qué parte corresponde al identificador del *host*.

PARA ESTO, SE APLICA LA SIGUIENTE LÓGICA:

- Si el bit de la máscara es **1**, el bit correspondiente en la dirección IP se interpreta como parte de la red.
- Si el bit de la máscara es **0**, el bit correspondiente se interpreta como parte del *host*.

En la siguiente figura —correspondiente al ejemplo previamente desarrollado— la máscara utilizada es **255.255.255.0**, lo que implica que los tres primeros octetos de la dirección IP identifican a la red, mientras que el cuarto octeto identifica al *host* dentro de dicha red.

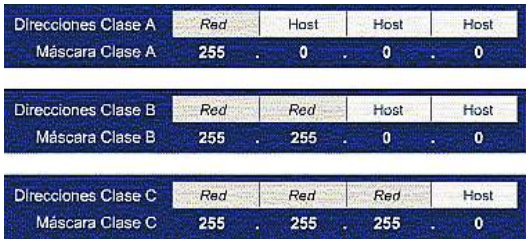
Figura 11. Relación entre dirección IP y máscara de subred para identificar red y *host*



Fuente: Elaboración propia.

En la figura siguiente se presentan las máscaras predeterminadas de las clases A, B y C.

Figura 12. Máscaras predeterminadas de las clases A, B y C



Fuente: Elaboración propia.

Observando la submáscara de red, es posible deducir el tipo de red en cuestión: la cantidad de octetos con valor **255** indica cuántos bits están reservados para la identificación de red, y, por lo tanto, permite estimar cuántas direcciones IP están disponibles para los *hosts*.

ESTO OFRECE UNA FORMA RÁPIDA DE IDENTIFICAR EL TIPO DE RED:

- **255.0.0.0** → red clase A
- **255.255.0.0** → red clase B
- **255.255.255.0** → red clase C

En la siguiente tabla se presentan las máscaras más comunes utilizadas, en función de la cantidad de bits ocupados por el campo de subred.

Tabla 4. Máscaras de subred típicas según cantidad de bits reservados para subredes

Cantidad de bits en la porción de red y subred	Máscara de subred en formato decimal
8	255 . 0 . 0 . 0
9	255 . 128 . 0 . 0
10	255 . 192 . 0 . 0
11	255 . 224 . 0 . 0
12	255 . 240 . 0 . 0
13	255 . 248 . 0 . 0
14	255 . 252 . 0 . 0
15	255 . 254 . 0 . 0
16	255 . 255 . 0 . 0
17	255 . 255 . 128 . 0
18	255 . 255 . 192 . 0
19	255 . 255 . 224 . 0
20	255 . 255 . 240 . 0
21	255 . 255 . 248 . 0
22	255 . 255 . 252 . 0
23	255 . 255 . 254 . 0
24	255 . 255 . 255 . 0
25	255 . 255 . 255 . 128
26	255 . 255 . 255 . 192
27	255 . 255 . 255 . 224
28	255 . 255 . 255 . 240
29	255 . 255 . 255 . 248
30	255 . 255 . 255 . 252

Fuente: Elaboración propia.

Una manera compacta de expresar la máscara de red es mediante la cantidad de bits en **1** que contiene la máscara. A continuación, se presentan algunos ejemplos:

Tabla 5. Ejemplos de máscaras de subred expresadas en notación CIDR

Dirección IP	**Mascará subred**		**Comentarios**
192.168.0.0	255.255.255.0	/24	[255.255.255].[0] indica red clase C (tiene disponible 1 octeto para definir hosts de la red)
192.168.1.37	255.255.255.0	/24	[255.255.255].[0] indica red clase C (tiene disponible 1 octeto para definir hosts de la red)
1.0.0.0	255.0.0.0	/8	[255].[0.0.0] indica red clase A (tiene disponible 3 octetos para definir hosts de la red)
128.0.0.0	255.255.0.0	/16	[255.255].[0.0] indica red clase B (tiene disponible 2 octetos para definir hosts de la red)
128.0.0.0	255.255.255.0	/24	[255.255.255].[0] indicaría red clase C, pero la IP no está en el rango homologado de la clase C.
192.168.0.255	255.255.255.0	/24	[255.255.255].[0] indica red clase C. La IP termina en 255, indica *broadcarts* (paquete enviado a todos los

			hosts de la red).

Fuente: elaboración propia.

Puertos y servicios

En redes, un puerto es un punto virtual, basado en software, donde comienzan y terminan las conexiones de red. Todos los ordenadores conectados a la red exponen una serie de puertos para poder recibir tráfico. Cada puerto está asociado a un proceso o servicio específico, y los diferentes protocolos utilizan puertos distintos.

Los puertos permiten que múltiples servicios operen sobre una misma dirección IP. Cada puerto representa una «puerta de entrada» a un servicio de red, como web, correo o archivos. Los atacantes escanean puertos en busca de servicios expuestos con fallos.

A CONTINUACIÓN, SE PRESENTAN LOS DISTINTOS TIPOS DE PUERTOS SEGÚN SU RANGO NUMÉRICO:

- **Bien conocidos (0–1023)**. Usados por protocolos estándar como HTTP (80), HTTPS (443) y SSH (22).
- **Registrados (1024–49151):** usados por aplicaciones personalizadas.
- **Dinámicos (49152–65535):** usados temporalmente.

Protocolos comunes

En la siguiente tabla se enumeran algunos de los protocolos más utilizados junto con sus respectivos puertos:

Tabla 6. Protocolos comunes y sus puertos asociados

Servicio	Puerto TCP	Protocolo
HTTP	80	TCP

HTTPS	443	TCP
FTP	21	TCP
SSH	22	TCP
DNS	53	UDP/TCP

Fuente: elaboración propia.

A CONTINUACIÓN, SE ENUMERAN ALGUNOS SERVICIOS HABITUALES EN REDES Y UNA BREVE DESCRIPCIÓN DE SU FUNCIÓN:

- **Telnet:** terminal virtual remota; transporta datos no cifrados.
- **SSH (*secure shell*):** terminal virtual remota; transporta datos cifrados.
- **FINGER:** proporciona información sobre los usuarios conectados.
- **FTP (*file transfer protocol*):** protocolo de transferencia de archivos.
- **TFTP (*trivial file transfer protocol*):** protocolo de transferencia de archivos simple.
- **DHCP (*dynamic host configuration protocol*):** protocolo de asignación de direcciones IP.
- **SNMP (*simple network management protocol*):** protocolo simple de administración de redes.
- **DNS (*domain name system*):** protocolo que relaciona nombres simbólicos con direcciones IP.
- **NETBIOS:** servicios de comunicación para redes LAN.
- **SMB:** permite la compartición de recursos.
- **WINS:** servicio de nombres.

Telnet

El protocolo Telnet es un protocolo de capa de aplicación que emula una terminal remota a través de una red TCP/IP. Su objetivo es definir una interfaz estándar para la intercomunicación entre sistemas finales,

mediante una terminal virtual.

Antiguamente, para iniciar una sesión remota, se conectaba al servidor una terminal sin capacidad de procesamiento, llamada «terminal boba», por medio de un cable conectado al puerto serie. La comunicación se establecía a través de un protocolo de transmisión serie. El servidor actuaba como centro de una topología en estrella, donde convergían todas las terminales.

Algunas desventajas de este sistema eran:

- la distancia entre las terminales y el servidor, limitada por el alcance del cable serie;
- el costo asociado a mantener en el servidor un puerto serie por cada terminal remota.

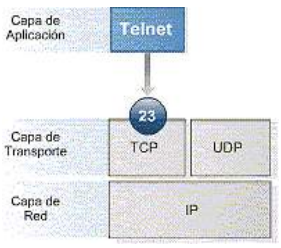
Con la llegada de las estaciones de trabajo y la expansión de las redes TCP/IP, las terminales físicas fueron reemplazadas por un software de terminal virtual, que se ejecuta en la estación de trabajo y se comunica con el servidor utilizando el protocolo Telnet. Esta evolución eliminó la limitación de distancia entre terminales y servidor.

El protocolo Telnet está definido principalmente por los siguientes RFC:

- RFC 854 (*Telnet Protocol Specifications*)
- RFC 855 (*Telnet Option Specifications*)

Telnet se ejecuta sobre TCP y tiene asignado el puerto 23. Funciona en la capa de aplicación de la pila de protocolos TCP/IP.

Figura 13. Funcionamiento de Telnet dentro del modelo TCP/IP



El protocolo se basa en tres ideas principales:

- el concepto de «terminal virtual de red» (*network virtual terminal*, NVT);

- una visión simétrica de terminales y procesos;
- la negociación de opciones.

En relación con la seguridad, Telnet permite acceder a una cuenta en un servidor y ejecutar las acciones autorizadas según los privilegios asignados. Para ello, es necesario contar con un nombre de usuario (*login*) y una contraseña válidos.

El protocolo proporciona autenticación mediante el ingreso de esas credenciales al iniciar una sesión. Sin embargo, su principal problema es que transmite los datos en texto claro, es decir, sin cifrado. En redes de difusión como Ethernet, Token Ring o FDDI, el tráfico enviado por una estación puede ser recibido por todas las demás que forman parte de la LAN. Como los datos no viajan cifrados, cualquier estación conectada podría interceptar y leer las transacciones de otra.

Para remediar esta debilidad, se desarrolló el protocolo SSH (*secure shell*), que ofrece una funcionalidad similar a Telnet, pero con cifrado de datos durante toda la comunicación.

FTP (file transfer protocol)

El protocolo FTP es el estándar actual para la transferencia de archivos en redes TCP/IP. Utiliza el puerto TCP 21 para establecer conexiones y está definido en el RFC 959. Funciona bajo un modelo cliente-servidor y emplea TCP como protocolo de transporte.

Entre sus principales características, se destacan las siguientes:

- **Permite un acceso interactivo:** Aunque puede utilizarse software de aplicación para realizar las transferencias, la mayoría de los sistemas operativos incluyen una interfaz basada en comandos para acceder a servidores remotos.
- **Ofrece configuración del formato de transferencia**: el usuario puede optar entre modo texto o binario, y especificar el formato de texto que utilizará.
- **Requiere autenticación:** antes de iniciar la transferencia, el cliente debe identificarse en el servidor con un nombre de usuario y una contraseña válidos.

Existen dos implementaciones del protocolo, ambas sobre TCP: FTP estándar y FTP pasivo.

- **FTP estándar:** el cliente establece una conexión hacia el servidor a través del puerto 21. Luego, el servidor inicia una segunda conexión desde su puerto 20 hacia un puerto definido por el cliente. Esta modalidad fue la primera en desarrollarse y requiere que el servidor tenga permiso para iniciar conexiones hacia el cliente.
- **FTP pasivo:** en lugar de iniciar la conexión de datos desde el servidor, es el cliente quien lo hace. Para ello, envía el comando PASV y recibe del servidor el número de puerto al que debe conectarse. Este método mejora la seguridad y facilita el uso en redes protegidas, ya que todas las conexiones se originan desde el cliente.

En cuanto al **proceso de conexión**, tanto en el modo estándar como en el pasivo, el cliente inicia una conexión TCP hacia el servidor a través del puerto 21, asignado convencionalmente a este servicio. Esta comunicación se denomina canal de control (o conexión de control).

Las diferencias entre ambos modos radican en cómo se establece la conexión de transferencia de datos:

- **FTP estándar.** El cliente utiliza el canal de control para enviar un paquete con el comando «PORT», mediante el cual indica el número de puerto que usará para la transferencia. El servidor, entonces, inicia una conexión desde su puerto local 20 hacia el puerto especificado por el cliente. Esta nueva conexión se denomina canal de transferencia de datos.
- **FTP pasivo:** en lugar de enviar el comando «PORT», el cliente envía «PASV» para conocer el puerto que utilizará el servidor para la transferencia. A partir de esta información, el cliente inicia la conexión hacia ese puerto del servidor.

Las conexiones de transferencia se crean dinámicamente cuando son necesarias, mientras que la conexión de control permanece activa durante toda la sesión. Una vez que esta se cierra, la sesión finaliza y ambos extremos concluyen todos los procesos de transferencia.

En lo que respecta al **acceso**, las características del servicio FTP hacen que la autenticación sea obligatoria, por lo que solo los clientes que cuenten con un nombre de usuario y una contraseña válidos en el servidor pueden conectarse.

Para facilitar el acceso a ciertos archivos públicos, la mayoría de los servidores permiten el acceso al **FTP anónimo**. Esto significa que el cliente no necesita contar con credenciales propias, sino que ingresa como invitado, generalmente identificado como *anonymous*, y sujeto a las restricciones que imponga el administrador del servicio.

En relación con la **seguridad**, es importante conocer bien este servicio, ya que además de utilizarse para transferir archivos, también puede ser empleado con fines maliciosos.

Dado que FTP requiere autenticación, podemos aprovechar esta característica para definir con cuidado qué datos y directorios estarán disponibles para cada usuario, en particular para quienes accedan de forma anónima. Si los accesos anónimos no son necesarios, convendrá deshabilitarlos. En caso de habilitarlos, lo recomendable es que dichos usuarios solo puedan descargar información del servidor.

Otro aspecto relevante es que la transferencia de datos se realiza en texto claro. Por lo tanto, si nos encontramos en una red de difusión, como Ethernet, Token Ring o FDDI, cualquier estación conectada podría interceptar e interpretar los datos transmitidos, incluidos nombres de usuario y contraseñas.

TFTP (trivial file transfer protocol)

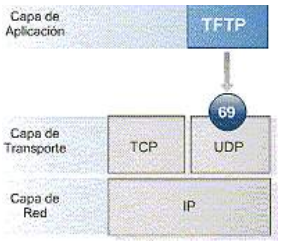
El protocolo TFTP es una versión muy simple para la transferencia de archivos. Utiliza el protocolo UDP y presenta las siguientes características:

- provee control de errores para cada datagrama;
- no establece una conexión entre cliente y servidor;
- no incluye control de flujo;
- no permite acceso interactivo;
- no implementa autenticación.

Su simplicidad facilita la implementación en dispositivos con recursos limitados de memoria y almacenamiento. Por ejemplo, algunos modelos de *routers* lo utilizan para actualizar sus sistemas operativos. Generalmente, se emplea en entornos controlados, como dentro de una LAN, ya que no es confiable para redes más amplias.

TFTP permite transferencias de archivos entre un cliente y un servidor. El cliente realiza la solicitud inicial al puerto UDP número 69 del servidor. Debido a su diseño simple, cualquier error interrumpe la transferencia, con la excepción de la pérdida de un segmento, que desencadena la retransmisión del último bloque enviado.

Figura 14. Ubicación de TFTP en la pila de protocolos TCP/IP



Para iniciar una transferencia, el cliente envía una petición de lectura o escritura al puerto UDP 69. Si el servidor acepta el requerimiento, responde desde un puerto diferente, que se utilizará para el resto de la transferencia. El archivo se transfiere en bloques de tamaño fijo de 512 bytes, numerados secuencialmente desde 1. Cada bloque debe ser confirmado con un acuse de recibo antes de enviar el siguiente. Cuando se recibe un bloque de tamaño inferior a 512 bytes, se interpreta como el final de la transferencia.

En relación con la **seguridad**, TFTP no ofrece autenticación de usuarios ni cifrado de datos, por lo que se considera un protocolo no seguro. Por este motivo, se recomienda no implementarlo, salvo en casos muy específicos.

Si su uso resulta indispensable para transferencias entre dispositivos, es fundamental controlar qué archivos y directorios estarán expuestos. Por lo general, solo se habilita la lectura, no la escritura. Las versiones más recientes se configuran por defecto para restringir el acceso únicamente al directorio «/tftpboot». Esta es una medida adecuada, aunque insuficiente: un atacante podría acceder a cualquier archivo en ese directorio si conoce su nombre. Esto incluye archivos sensibles de configuración de *routers*, como aquellos que suelen llamarse «<nombre del router>.cfg», los cuales pueden contener contraseñas o datos de comunidades SNMP.

Algunas implementaciones de TFTP permiten definir una lista de *hosts*, mediante direcciones IP o nombres DNS, con permisos específicos de lectura o escritura. En caso de no contar con esta posibilidad, será necesario aplicar mecanismos complementarios para limitar el acceso al servidor.

DHCP (dynamic host configuration protocol)

El protocolo de configuración dinámica de hosts (DHCP) fue diseñado por la IETF como sucesor de BOOTP para la configuración automática de dispositivos en redes TCP/IP. Su función principal es asignar direcciones IP a las estaciones de trabajo.

DHCP representa una mejora respecto de su antecesor en dos aspectos:

- Permite que el host reciba toda la información necesaria para su configuración de red en un solo mensaje (como dirección IP, máscara de subred, puerta de enlace predeterminada y dirección de un servidor DNS).
- Utiliza un mecanismo de asignación dinámica de direcciones IP, manteniendo un conjunto de direcciones disponibles que se asignan temporalmente a medida que los clientes las solicitan.

Está implementado bajo un modelo cliente-servidor, por lo que es necesario contar con un servidor DHCP configurado en la red local. Actualmente, es el protocolo de configuración dinámica más utilizado. En redes TCP/IP extensas, resulta una herramienta muy útil para los administradores, ya que:

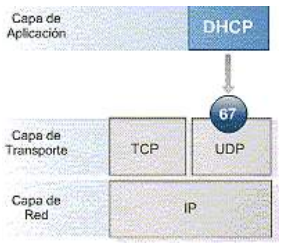
- reduce la probabilidad de errores en la configuración de los *hosts*;
- permite un uso más eficiente del espacio de direcciones IP disponible.

La posibilidad de automatizar la configuración de red mediante DHCP fue un avance significativo en comparación con BOOTP. Un servidor puede configurarse tanto para asignar direcciones IP estáticas basadas en la dirección MAC del cliente como para entregarlas dinámicamente, sin importar quién las solicite. Estas asignaciones tienen una duración determinada, que puede variar según las necesidades de cada red local, desde minutos hasta horas.

Toda la comunicación entre cliente y servidor se realiza mediante el protocolo UDP, utilizando el puerto 67 en el servidor. La siguiente figura ilustra la ubicación de DHCP en la pila de protocolos TCP/IP.

En cuanto al funcionamiento, las estaciones pueden tener una dirección IP fija o estar configuradas como clientes DHCP. Cuando un dispositivo necesita configurarse automáticamente, se inicia un proceso de intercambio de mensajes entre el cliente y uno o varios servidores DHCP.

Figura 15. Ubicación de DHCP en la pila de protocolos TCP/IP



Los pasos del proceso son los siguientes:

1. El cliente intenta descubrir un servidor DHCP disponible.
2. Los servidores que reciben la solicitud responden con ofertas de configuración.
3. El cliente selecciona una de las ofertas y notifica su elección.
4. El servidor confirma la configuración seleccionada por el cliente.

El primer mensaje que envía el cliente (generalmente al encenderse) es un *broadcast* a toda la red local para localizar un servidor DHCP. Ese mensaje se conoce como «DHCPDISCOVER». En la siguiente tabla se presentan los valores posibles para los diferentes tipos de mensajes utilizados por este protocolo:

Tabla 7. Tipos de mensajes utilizados en el protocolo DHCP

Tipo de Mensaje	**Descripción**
1	DHCPDISCOVER
2	DHCPOFFER
3	DHCPREQUEST
4	DHCPDECLINE
5	DHCPACK
6	DHCPNACK
7	DHCPRELEASE

Fuente: elaboración propia.

Una vez que el cliente envía el paquete «DHCPDISCOVER», queda a la espera de respuestas por parte de los servidores DHCP presentes en la red. Estos paquetes se dirigen a una dirección *broadcast* para

asegurar que todos los hosts locales los reciban.

Aunque todos los hosts de la red reciben el mensaje, solo los servidores DHCP lo interpretan y responden al cliente mediante un paquete «DHCPOFFER». El cliente puede recibir una o varias respuestas, y cada una de ellas puede incluir más de una dirección IP (en general, hasta cuatro).

Cuando recibe las ofertas, el cliente selecciona una —por ejemplo, la primera que llegue— y responde al servidor correspondiente informando qué dirección IP acepta. En esa respuesta también se negocia el tiempo de concesión de la dirección IP y otros parámetros de configuración. Para ello, el cliente envía un paquete «DHCPREQUEST» al servidor DHCP seleccionado.

La información que un cliente generalmente obtiene de un servidor DHCP incluye:

- dirección IP;
- máscara de subred;
- dirección del *router* por defecto (puerta de enlace o *default gateway*);
- nombre de dominio;
- dirección de servidores DNS;
- dirección de servidores WINS;
- tiempo de asignación de la dirección.

El servidor marca la dirección IP otorgada al cliente para evitar asignarla a otro dispositivo. Para confirmar la asignación, le envía al cliente un paquete «DHCPACK».

Si un cliente que recibió toda su configuración TCP/IP ya no necesita acceso a la red antes de que finalice el tiempo de concesión, puede enviar un paquete «DHCPRELEASE» al servidor para informarle que liberó la dirección IP.

En cambio, si necesita conservar el acceso más allá del período inicialmente asignado, deberá renegociar la concesión.

En relación con la **seguridad**, es necesario tomar ciertos recaudos en entornos que utilizan DHCP para evitar usos indebidos.

Al analizar el proceso de comunicación, se observa que el primer paquete enviado por un cliente DHCP —el mensaje «DHCPDISCOVER»— se transmite mediante *broadcast* a toda la red. Este tipo de mensaje puede ser respondido por cualquier servidor que se encuentre en la red local. Por este motivo, una

estación que implemente un servidor DHCP no autorizado podría asignar direcciones IP arbitrarias, generando conflictos y problemas de comunicación en la red.

Además, deben considerarse los riesgos asociados a cada tipo de asignación:

- si se utiliza asignación dinámica de direcciones, una estación externa a la red podría conectarse y recibir una dirección IP válida, obteniendo los mismos permisos y privilegios que una estación autorizada;
- si se utiliza asignación estática, una estación maliciosa podría suplantar la identidad de otra modificando la dirección MAC de sus tramas, con el objetivo de obtener una dirección IP previamente reservada.

DNS, HTTP(S), TLS

El sistema DNS (*domain name system*) convierte nombres de dominio en direcciones IP. Los protocolos HTTP y HTTPS permiten el intercambio de información en la web, y TLS se encarga de cifrar ese intercambio para proteger la confidencialidad de los datos.

DNS (*domain name service*) o sistema de nombres de dominio

En los comienzos de Internet, los distintos hosts se identificaban únicamente por sus direcciones IP. Con el tiempo, esto evolucionó hacia el uso de nombres simbólicos, lo que permitió referirse a un host mediante una cadena de caracteres en lugar de su dirección numérica.

Esta solución trajo consigo un nuevo problema: la necesidad de mantener, de forma centralizada y coherente, la correspondencia entre nombres y direcciones IP. Inicialmente, esta

relación era gestionada por el Network Information Center (NIC), que la registraba en un único archivo llamado «HOSTS.TXT». Este archivo era distribuido a todas las estaciones mediante FTP.

En la siguiente figura se muestra un ejemplo de dicho archivo.

Figura 16. Ejemplo de un archivo hosts.txt

Fuente: Elaboración propia.

Debido al crecimiento explosivo de Internet, el mecanismo anterior dejó de ser práctico y fue reemplazado por un nuevo concepto: el sistema de nombres de dominio (*domain name system*, DNS), que convierte un nombre de dominio en una dirección IP.

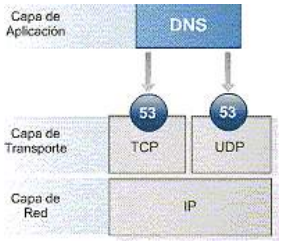
Este sistema permite que una estación obtenga de forma dinámica la dirección IP correspondiente a un nombre dado, sin necesidad de mantener un archivo centralizado con todas las asociaciones.

El sistema de nombres de dominio es una estructura distribuida, conformada por miles de servidores. Cada uno de ellos mantiene únicamente una parte del espacio de nombres total, lo que permite escalar y gestionar la red de forma eficiente.

Aspectos generales de DNS

Los mensajes DNS pueden transmitirse mediante TCP o UDP, y en ambos casos se utiliza el puerto 53. En la siguiente figura se muestra la ubicación del protocolo DNS en la pila de protocolos TCP/IP.

Figura 17. Ubicación de DNS en la pila de protocolos TCP/IP



Fuente: Elaboración porpia.

UDP impone un tamaño máximo al segmento, mientras que TCP no. En el caso de utilizar UDP, el tamaño máximo permitido es de 512 bytes. En cambio, con TCP, el tamaño total se especifica al inicio del segmento.

EL ESTÁNDAR QUE DEFINE EL FUNCIONAMIENTO DE DNS ESTABLECE QUE:

- para consultas comunes, se recomienda el uso de UDP;
- si la respuesta supera el límite de 512 bytes y debe dividirse, debe utilizarse TCP;
- se prefiere UDP sobre TCP debido a su menor *overhead*; en la práctica, es poco frecuente que una respuesta supere los 512 bytes;
- para realizar transferencias de zonas, se debe utilizar TCP, ya que el volumen de datos transferido supera ampliamente ese límite.

El espacio de nombres de dominio es una estructura jerárquica compuesta por dominios y subdominios, que en conjunto conforman un árbol de nombres.

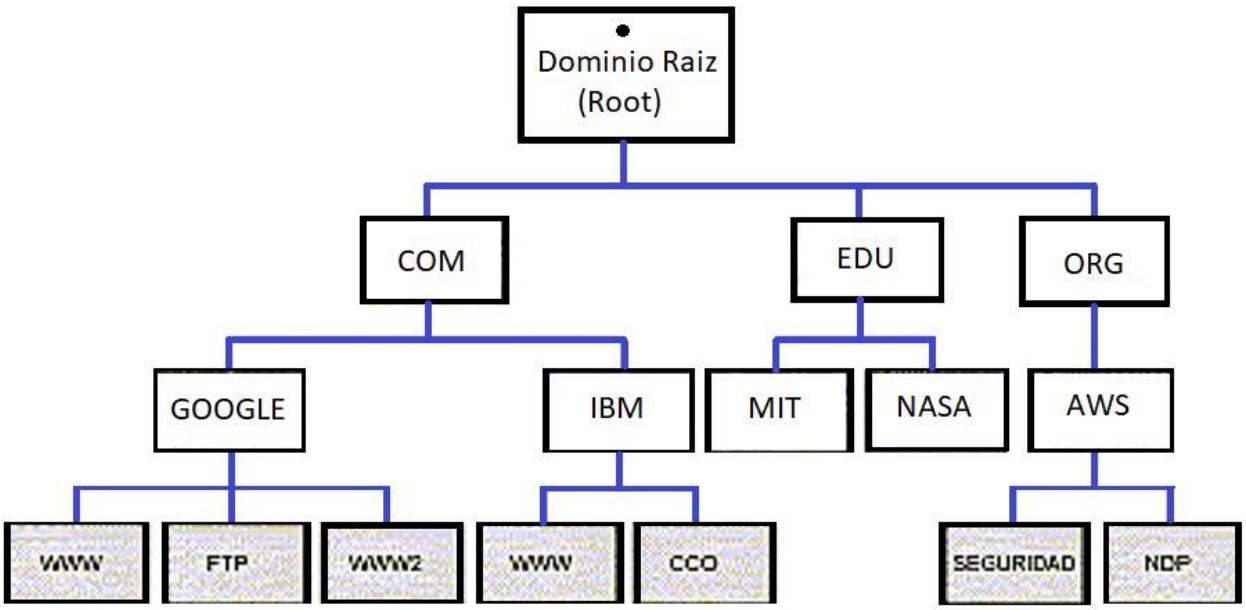
POR EJEMPLO, EN EL NOMBRE: PC1.AULA3.DEPTOINF.FACULTAD.EDU, SE OBSERVA LA SIGUIENTE JERARQUÍA:

- «aula3.deptoinf.facultad.edu» es el dominio de nivel más bajo (nombre completo);
- «aula3» es un subdominio de «deptoinf.facultad.edu»;
- «deptoinf.facultad.edu» es un subdominio de «facultad.edu»;

- «facultad.edu» es, a su vez, un subdominio de «edu».

Esta jerarquía puede representarse mediante un árbol de dominios.

Figura 18. Ejemplo de árbol jerárquico del espacio de nombres de DNS



Fuente: Elaboración propia.

Dominios genéricos y dominios nacionales

Los dominios de nivel superior pueden clasificarse en dos grandes grupos: aquellos que identifican el tipo de organización y aquellos que están asociados a un país determinado.

Dominios genéricos

Los nombres compuestos por tres caracteres que conforman los dominios de nivel superior son conocidos como dominios genéricos o dominios organizacionales. En la siguiente tabla se presentan algunos ejemplos:

Tabla 8. Dominios genéricos de nivel superior

Nombre de dominio	Significado

com	Organizaciones comerciales
edu	Instituciones educativas
gov	Instituciones gubernamentales
int	Instituciones internacionales
mil	Milicia estadounidense
net	Centros de soporte de red
org	Organizaciones sin fines de lucro

Fuente: elaboración propia.

Dominios nacionales —

Existen también dominios de nivel superior asociados a países, definidos conforme al estándar ISO 3166. Estos son conocidos como dominios nacionales o geográficos, y van desde «.ae» para los Emiratos Árabes Unidos hasta «.zw» para Zimbabue.

La mayoría de los países adoptan, dentro de sus dominios nacionales, una estructura similar a la de los dominios genéricos. Así, es posible encontrar extensiones como «.edu.ar», «.com.uk» o «.gov.uy».

Administración de dominios

El sistema de nombres de dominio utiliza un espacio de nombres distribuido. Los nombres simbólicos se agrupan en «zonas de autoridad», comúnmente llamadas zonas.

En cada zona, uno o más hosts tienen la tarea de mantener la base de datos que asocia nombres con direcciones IP, y de operar como servidores que responden consultas de otras estaciones. Estos servidores de nombres locales están conectados a la jerarquía del sistema de nombres de dominio.

Cada zona contiene una parte —o una rama— del árbol jerárquico, y los nombres dentro de ella se administran de forma independiente respecto del resto de las zonas. El inicio de todo el árbol corresponde a la zona raíz, representada por «.» (también llamada *root*).

Dentro de una zona pueden existir subdominios, los cuales pueden ser delegados a servidores distintos, responsables de su administración.

Consideremos, por ejemplo, una consulta para el nombre «seguridad.proydesa.org.ar», y que nuestro servidor no tenga esa información en caché. En ese caso, la solicitud será reenviada al servidor raíz («.»), que derivará la consulta al servidor con la delegación del dominio «ar».

Ese servidor, a su vez, reenviará la consulta al servidor responsable del dominio «org», y así sucesivamente, hasta llegar al servidor que contenga la información solicitada. También puede ocurrir que, en alguno de estos pasos, un servidor tenga en caché la respuesta. En tal caso, responderá directamente con la dirección IP correspondiente, sin necesidad de reenviar la consulta a otro servidor.

ESTE ESQUEMA OFRECE VARIAS VENTAJAS:

- En lugar de mantener una base de datos centralizada, la carga de mantenimiento se distribuye entre múltiples servidores.
- La autoridad y la responsabilidad sobre la creación y modificación de los nombres simbólicos se delega en los propietarios de las organizaciones correspondientes.
- Desde el punto de vista del usuario, el proceso es completamente transparente: se envía una solicitud y un servidor responde con la información requerida.

Resolución de consultas DNS

EL PROCESO DE RESOLUCIÓN DE NOMBRES DE DOMINIO PUEDE RESUMIRSE EN LOS SIGUIENTES PASOS:

1. Un programa de usuario realiza una solicitud de resolución.
2. El cliente DNS (denominado *resolver*) revisa su tabla de caché para comprobar si ya ha realizado la consulta. Si no la encuentra, envía la solicitud al servidor DNS que tenga configurado.

3. El servidor DNS recibe la solicitud y verifica si la respuesta corresponde a su zona de autoridad. Si es así, responde directamente. En caso contrario, consulta a otros servidores disponibles, comenzando por los servidores raíz, hasta obtener una respuesta.
4. El programa de usuario recibe la dirección IP asociada al nombre solicitado o un mensaje de error si no se pudo resolver. Por lo general, el programa no recibe la lista de servidores consultados durante el proceso.

Los mensajes de consulta y respuesta se transportan mediante UDP o TCP. Este proceso sigue un modelo cliente-servidor.

La función cliente es transparente para el usuario y es invocada por las aplicaciones cuando necesitan realizar resoluciones de nombres. Por su parte, el servidor de nombres es una aplicación que se encarga de traducir nombres simbólicos a direcciones IP.

La resolución de un nombre de dominio se realiza de derecha a izquierda. Primero se identifica el dominio de nivel superior —por ejemplo, «.ar»— y se consulta el servidor correspondiente, como «nic.ar», administrado por la Cancillería, donde se gestionan los dominios argentinos.

Luego se analiza el siguiente tramo, como «.org», «.edu», «.com», «.mil» o «.gov», y por último se interpreta el nombre específico del host. Con esta información se obtiene finalmente la dirección IP correspondiente.

Operación del resolver

Las consultas de nombres pueden ser de dos tipos: recursivas o iterativas. Un bit en la solicitud indica qué tipo de consulta está realizando el cliente.

LA DIFERENCIA ENTRE AMBOS TIPOS APARECE CUANDO EL SERVIDOR CONSULTADO NO PUEDE RESOLVER DIRECTAMENTE LA SOLICITUD:

- si el cliente solicita una consulta recursiva (lo más habitual), el servidor se encargará de obtener la respuesta completa, consultando a otros servidores si es necesario;

- si la consulta es iterativa, el servidor responderá con la información que tenga disponible y proporcionará una lista de servidores adicionales para que el cliente continúe la búsqueda por su cuenta.

Las respuestas de nombres de dominio también pueden clasificarse en dos tipos: autoritativas o no autoritativas. Un bit en la respuesta indica si el servidor tiene autoridad sobre el nombre consultado o si la información proviene de su caché.

La d**iferencia entre las respuestas autoritativas y no autoritativas** radica en si el servidor tiene o no autoridad sobre la zona del dominio consultado.

Cuando el servidor tiene autoridad sobre la zona correspondiente, responde con una respuesta autoritativa. Es decir, si recibe una consulta para un dominio perteneciente a una zona bajo su control, puede contestar directamente con datos oficiales.

SI EL SERVIDOR NO TIENE AUTORIDAD SOBRE LA ZONA DEL DOMINIO, SU COMPORTAMIENTO DEPENDERÁ DEL TIPO DE CONSULTA RECIBIDA:

- Si se trata de una consulta recursiva, reenviará la solicitud a otro servidor con autoridad o a un servidor raíz. Si ese segundo servidor tampoco tiene autoridad y ha delegado la zona, el proceso se repetirá hasta encontrar una respuesta válida. La respuesta obtenida, en este caso, será no autoritativa. Cuando un servidor o *resolver* obtiene una respuesta, la almacena en caché para agilizar futuras consultas. Esta información se guarda por un período determinado, especificado por el servidor de origen en un campo llamado «TTL» (*time to live*), que típicamente es de 172800 segundos (dos días).
- Si la consulta es iterativa, el servidor devolverá la información que tenga en caché, junto con una lista de servidores de nombres que puedan proporcionar una respuesta autoritativa.

Principales registros de recursos (RR)

La base de datos distribuida del sistema de nombres de dominio está compuesta por registros de recursos (RR), que se organizan en diferentes clases según el tipo de red. En este caso, se analizarán únicamente los registros correspondientes a la clase Internet.

Los registros de recursos proporcionan un mapeo entre nombres de dominio y objetos de red. El objeto más común es la dirección de los hosts de Internet, aunque el sistema está diseñado para localizar una variedad más amplia de elementos.

Una zona se compone de un conjunto de registros de recursos, comenzando con un registro «start of authority» (SOA), que identifica el nombre de dominio correspondiente a la zona.

Debe existir un registro «name server» (NS) para el servidor de dominio primario. También pueden incluirse registros NS para los servidores secundarios. Estos registros permiten identificar qué servidores tienen autoridad sobre la zona.

A continuación, se incorporan otros registros que pueden establecer correspondencias entre nombres y direcciones IP, o entre alias y nombres reales.

El formato general de un registro de recurso se muestra en la siguiente figura.

Figura 19. Formato general de un registro de recurso (RR) en DNS

Fuente: Elaboración propia.

El campo **«nombre»** indica el nombre de dominio que se desea definir. Aunque DNS es flexible en las reglas para la composición de nombres, recomienda una sintaxis que reduzca la posibilidad de que las aplicaciones interpreten erróneamente los dominios.

UN NOMBRE QUE SIGUE ESTAS RECOMENDACIONES DEBE CUMPLIR LOS SIGUIENTES REQUISITOS:

- estar compuesto por una serie de etiquetas formadas por caracteres alfanuméricos o guiones;
- cada etiqueta debe tener entre 1 y 63 caracteres de longitud;

- comenzar con un carácter alfabético.

Las etiquetas se separan mediante puntos («.»). Los nombres de dominio no distinguen entre mayúsculas y minúsculas.

El campo **«ttl»** (time to live) indica el tiempo, en segundos, durante el cual este registro puede permanecer en caché.

El campo **«clase»** identifica la familia de protocolos; el valor más común es «IN» (Internet).

El campo **«tipo»** especifica el tipo de recurso contenido en el registro. Los distintos tipos están definidos en los RFC 1034, 1035 y 1706.

EL CAMPO «RDATA» CONTIENE EL VALOR DEL RECURSO, QUE VARÍA SEGÚN EL TIPO:

- A. Una dirección IP (si la clase es IN).
- CNAME: un nombre de dominio.
- MX: un número de 16 bits que indica la preferencia (valores más bajos indican mayor prioridad), seguido por un nombre de dominio.
- NS: un nombre de host.
- PTR: un nombre de dominio.

En la siguiente tabla se enumeran los diferentes tipos de registros de recursos.

Tabla 8. Tipos de registros de recursos en DNS

Tipo	Valor	Significado
A	1	Una dirección de *host*.

MX	15	*Mail exchanger*. Servidor de *mail* del dominio; mapea a una casilla de *host*.
NS	2	*Name server*. Servidor de nombres autoritativo del dominio.
PTR	12	*Pointer*. Puntero hacia otra parte del espacio de nombres de dominio.
SOA	6	*Start of authority*. Inicio de una zona de autoridad.
WKS	11	Servicios bien conocidos. Especifica algunos servicios (por ejemplo, SMTP) que se espera estén activos en el *host*.
HINFO	13	Información del *hardware* y sistema operativo del *host*. Campo de comentario.

El dominio IN-ADDR.ARPA para mapeos reversos —

El sistema de nombres de dominio permite realizar mapeos tanto de nombres simbólicos a direcciones IP como en sentido inverso. La búsqueda directa de un nombre dentro de la base de datos resulta sencilla, gracias a la estructura jerárquica del sistema. Sin embargo, el proceso inverso no puede seguir esa jerarquía, por lo que se utiliza un espacio de nombres distinto destinado exclusivamente a los mapeos reversos.

Este espacio se denomina «in-addr.arpa». El sufijo «arpa» hace referencia a ARPAnet, la red precursora de Internet. Las direcciones IP se representan como cuatro números separados por puntos, y existe un subdominio para cada nivel jerárquico. En el caso de los mapeos reversos, la dirección IP se escribe en orden invertido.

Esto se debe a que los nombres de dominio se estructuran desde lo particular hacia lo general (es decir, de host a dominio), mientras que en una dirección IP los bytes más significativos aparecen al comienzo (de red a host). Por ejemplo, la dirección IP 129.34.139.30 se representa en este espacio como «30.139.34.129.in-addr.arpa».

Dada una dirección IP, el sistema puede utilizarse para obtener el nombre de host asociado. Una consulta de este tipo se denomina consulta «pointer».

Uso de WHOIS para obtener información de un dominio —

En internet existe una gran cantidad de información disponible sobre los diferentes dominios registrados. Una herramienta desarrollada para acceder a esa información es WHOIS. Existen múltiples bases de datos

WHOIS que pueden consultarse y que permiten obtener datos sobre un dominio específico o sobre un bloque de direcciones IP.

Para acceder a estas bases, se pueden utilizar distintos mecanismos. Una opción es consultar el sitio web de la entidad ARIN (American Registry for Internet Numbers). Otra posibilidad es realizar consultas desde una estación que tenga instalado un cliente WHOIS.

La información obtenida para un dominio puede incluir los siguientes datos:

- Registro. Datos del registro y de los servidores WHOIS.
- Empresa: información sobre la entidad propietaria del dominio.
- Dominio: detalles técnicos y administrativos del dominio consultado.
- Bloque de direcciones IP: rangos de direcciones asignados.
- Punto de contacto: datos de la persona o entidad responsable del dominio.

En la siguiente tabla se presenta una lista de las entidades regionales de registración donde pueden realizarse consultas WHOIS.

Tabla 9. Entidades regionales de registración que ofrecen servicios WHOIS

Entidad	Descripción	Dirección
ARIN	American Registry for Internet Numbers	http://www.arin.net
APNIC	Asia Pacific Network Information Centre	http://www.apnic.net
LACNIC	Latin American and Caribbean IP address Regional Registry	http://www.lacnic.net
RIPE NCC	RIPE Network Coordination Centre	http://www.ripe.net
AfriNIC	African Network Information Center	http://www.afrinic.org

Fuente: Elaboración propia.

Consideraciones sobre seguridad

Existen distintos tipos de amenazas dirigidas al sistema de nombres de dominio. En general, estas amenazas no tienen como objetivo principal al servicio DNS, sino que lo utilizan como parte del proceso de un ataque más amplio.

EL PRINCIPAL INCONVENIENTE DE DNS ES QUE:

- no realiza autenticación;
- no garantiza confidencialidad, ya que no cifra ni las consultas ni las respuestas;
- usualmente, utiliza solo un mensaje UDP para la consulta y otro para la respuesta, lo que facilita la manipulación o interceptación.

A CONTINUACIÓN, SE ENUMERAN ALGUNAS DE LAS AMENAZAS MÁS FRECUENTES AL SISTEMA DE NOMBRES DE DOMINIO:

- **Modificación de un paquete.** Una de las amenazas más simples contra DNS consiste en la alteración de paquetes utilizados en consultas o respuestas. En estos casos, el atacante modifica un paquete para inyectar información falsa, como una dirección IP incorrecta, con el fin de desviar la resolución.
- **Ataques basados en nombres.** Estos ataques, conocidos como *cache poisoning* (envenenamiento de caché), consisten en agregar información falsa a una respuesta legítima con el objetivo de contaminar la caché del *resolver*. De esta manera, el atacante logra insertar asociaciones falsas entre nombres y direcciones IP, que serán utilizadas en futuras consultas.
- **DNS *spoofing*.** Este ataque implica la falsificación de una dirección IP en respuesta a una consulta de resolución de nombre, o viceversa, la falsificación de un nombre asociado a una dirección IP. Esto es posible debido a la falta de autenticación del servicio DNS. Puede lograrse mediante la modificación de entradas en servidores comprometidos, el envío de respuestas falsas a peticiones legítimas o la infección de cachés mediante ataques indirectos, lo que se conoce como *DNS poisoning*.
- **Denegación de servicio.** Como cualquier otro servicio, DNS es vulnerable a ataques de denegación de servicio (DoS). Peor aún, los servidores DNS pueden ser explotados como amplificadores en este tipo de ataques, ya que los paquetes de respuesta suelen ser mucho más grandes que los de solicitud, incrementando así el volumen de tráfico dirigido a la víctima.

Seguridad relacionada con puertos —

En relación con la seguridad de los puertos, se deben tener en cuenta las siguientes recomendaciones:

- cerrar los puertos innecesarios mediante la configuración del *firewall*;
- evitar el uso de servicios por defecto que no incluyan mecanismos de autenticación;
- monitorear de forma continua la actividad inusual en puertos sensibles.

HTTP vs. HTTPS

HTTP (puerto 80) es el protocolo de transferencia de hipertexto, utilizado para la carga de páginas web mediante hiperenlaces. La comunicación en HTTP no está cifrada.

HTTPS (puerto 443) es la versión segura de HTTP, ya que incorpora cifrado mediante el protocolo TLS (*Transport Layer Security*). Permite el envío de datos cifrados entre el navegador y el servidor, garantizando la confidencialidad y autenticidad de la información transmitida.

Desde el punto de vista técnico, HTTPS no constituye un protocolo distinto, sino que es HTTP operando sobre TLS/SSL. La seguridad de HTTPS se basa en el uso de certificados digitales que permiten verificar la identidad del servidor.

Cuando un usuario accede a un sitio web mediante HTTPS, el servidor le envía su certificado digital, que contiene la clave pública necesaria para establecer una sesión segura. A continuación, cliente y servidor ejecutan el protocolo de enlace TLS, una secuencia de intercambios que permite negociar una conexión segura antes de iniciar la transferencia de datos.

HTTP

El protocolo de transferencia de hipertexto (HTTP) constituye la base de la *World Wide Web* y se utiliza para cargar páginas web mediante enlaces de hipertexto. HTTP es un protocolo de capa de aplicación diseñado para transferir información entre dispositivos conectados en una red, y funciona sobre otras capas del conjunto de protocolos TCP/IP.

Un flujo típico sobre HTTP involucra a una máquina cliente que envía una solicitud a un servidor, el cual responde con un mensaje que contiene la información solicitada.

Solicitud HTTP

Una solicitud HTTP es el mecanismo mediante el cual plataformas de comunicación en *internet*, como los navegadores web, solicitan la información necesaria para cargar un sitio web.
Cada solicitud HTTP incluye una serie de datos codificados que contienen distintos tipos de información. Una solicitud típica está compuesta por:

- la versión del protocolo HTTP;
- una URL;
- un método HTTP;
- encabezados de solicitud HTTP;
- un cuerpo opcional.

A continuación, se analizará el funcionamiento de las solicitudes HTTP y cómo se puede utilizar su contenido para compartir información entre cliente y servidor.

Método HTTP

Un método HTTP, también llamado verbo HTTP, indica la acción que se espera del servidor al realizar una solicitud. Dos de los métodos más comunes son los siguientes:

- «GET». Utilizado para solicitar información al servidor, como una página web u otro recurso.
- «POST»: empleado cuando el cliente envía datos al servidor, por ejemplo, la información de un formulario, como un nombre de usuario y una contraseña.

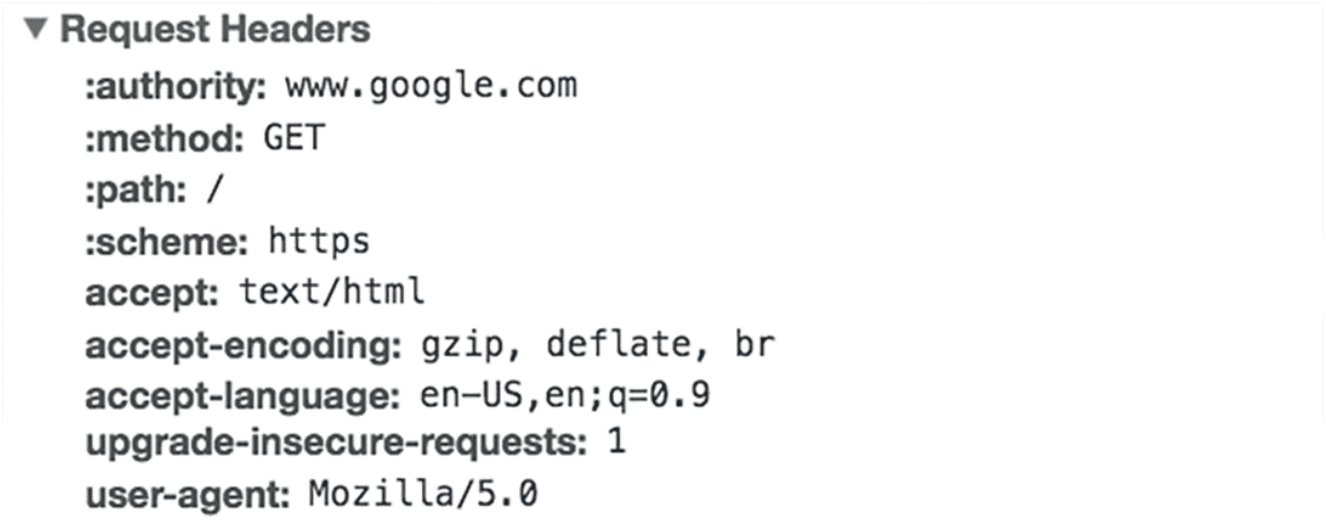
Cada método define el tipo de operación que el servidor debe realizar sobre el recurso solicitado.

Encabezados de solicitud HTTP

Los encabezados HTTP contienen información en formato de texto, organizada en pares clave-valor, y se incluyen en cada solicitud HTTP. También aparecen en las respuestas, tema que se abordará más adelante.

Estos encabezados permiten comunicar datos esenciales sobre la solicitud, como el navegador utilizado por el cliente, el tipo de contenido aceptado o los detalles del recurso solicitado.

Figura 20. Ejemplo de encabezados de solicitud HTTP de la pestaña de red de Google Chrome



Cuerpo de solicitud HTTP

El cuerpo de una solicitud HTTP es la sección que contiene la información enviada por el cliente al servidor web. Puede incluir datos como un nombre de usuario, una contraseña o cualquier otra información ingresada en un formulario.

Respuesta HTTP

Una respuesta HTTP es la información que un servidor web devuelve a un cliente (por lo general, un navegador) tras recibir una solicitud HTTP. Estas respuestas contienen datos relevantes en función del contenido solicitado.

Una respuesta HTTP típica incluye:

- un código de estado HTTP;
- encabezados de respuesta HTTP;

- un cuerpo opcional.

A continuación, se describen estos componentes.

Código de estado HTTP

Los códigos de estado HTTP son valores numéricos de tres dígitos que indican el resultado de una solicitud realizada por el cliente. Estos códigos se agrupan en cinco clases principales:

- 1xx: informativos
- 2xx: éxito
- 3xx: redirección
- 4xx: error del cliente
- 5xx: error del servidor

Las «xx» representan un rango de números entre 00 y 99.

Los códigos que comienzan con «2» señalan que la solicitud se completó correctamente. Por ejemplo, una respuesta con el código «200 OK» indica que el servidor procesó la solicitud de forma exitosa.

Por el contrario, si la respuesta comienza con «4» o «5», indica que hubo un error.

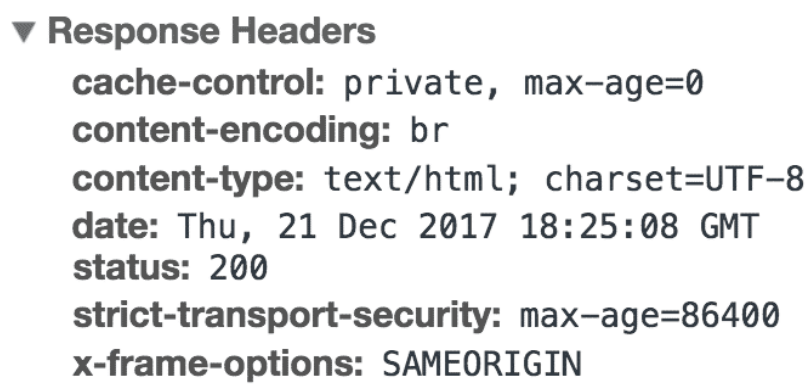
- Un código que comienza con «4», como «404 Not Found», señala un error del lado del cliente, por ejemplo, cuando se escribe incorrectamente una URL.
- Un código que comienza con «5», como «500 Internal Server Error», indica un problema en el servidor.

Los códigos que empiezan con «1» o «3» representan, respectivamente, mensajes informativos y redirecciones.

Encabezados de respuesta HTTP

Al igual que en las solicitudes, las respuestas HTTP incluyen encabezados que transmiten información adicional sobre los datos enviados. Estos encabezados pueden indicar, por ejemplo, el tipo y formato del contenido, el idioma, la fecha de expiración o las políticas de almacenamiento en caché.

Figura 21. Ejemplo de encabezados de respuesta HTTP de la pestaña de red de Google Chrome



Cuerpo de respuesta HTTP

Las respuestas HTTP a solicitudes de tipo «GET» suelen incluir un cuerpo que contiene la información solicitada. En la mayoría de los casos, se trata de datos en formato HTML, que el navegador interpreta para mostrar una página web.

Ataques DDoS a través de HTTP

HTTP es un protocolo sin estado, lo que significa que cada solicitud se ejecuta de forma independiente, sin conservar información sobre solicitudes anteriores. En su especificación original, cada solicitud HTTP abría y cerraba una conexión TCP individualmente. A partir de HTTP/1.1, se introdujo el concepto de conexión persistente, que permite enviar múltiples solicitudes a través de una misma conexión TCP, lo que mejora la eficiencia en el uso de recursos.

En el contexto de los ataques de denegación de servicio (DoS) o de denegación de servicio distribuida (DDoS), el envío masivo de solicitudes HTTP puede ser utilizado para saturar un servidor o dispositivo objetivo. Este tipo de ataques corresponde a la capa de aplicación (capa 7) del modelo OSI.

HTTPS —

El protocolo de transferencia de hipertexto seguro (HTTPS) es la versión segura de HTTP y se utiliza para el envío cifrado de datos entre un navegador web y un sitio web. HTTPS protege la información durante la transferencia, lo que resulta especialmente importante cuando se transmiten datos confidenciales, como credenciales bancarias, correos electrónicos o información de servicios médicos.

Cualquier sitio web que requiera inicio de sesión debe emplear HTTPS. En los navegadores modernos, como Chrome, los sitios que no utilizan HTTPS se indican de manera distinta a los que sí lo hacen. La presencia de un candado en la barra de direcciones señala que la página es segura. Los navegadores web priorizan el uso de HTTPS, y plataformas como Google Chrome marcan todas las páginas que no emplean este protocolo como no seguras.

Funcionamiento de HTTPS

HTTPS utiliza un protocolo de cifrado para proteger las comunicaciones: el *Transport Layer Security* (TLS), anteriormente conocido como *Secure Sockets Layer* (SSL). Este protocolo garantiza la seguridad mediante una infraestructura de clave pública asimétrica. En este sistema se utilizan dos claves diferentes:

- **Clave privada.** Controlada por el propietario del sitio web y mantenida en secreto. Se almacena en el servidor y se utiliza para desencriptar la información cifrada con la clave pública.
- **Clave pública:** disponible para cualquier usuario que desee comunicarse de forma segura con el servidor. La información cifrada con esta clave solo puede ser descifrada mediante la clave privada correspondiente.

Importancia de HTTPS

HTTPS evita que la información transmitida por los sitios web sea fácilmente accesible a terceros que intercepten la comunicación. Cuando se utiliza HTTP, los datos se dividen en paquetes que pueden ser interceptados o visualizados mediante herramientas disponibles libremente. Esto hace que la comunicación a través de medios inseguros, como redes wifi públicas, sea vulnerable a ataques de intercepción. Todas las comunicaciones en HTTP se transmiten en texto plano, lo que las hace fácilmente legibles para cualquiera con las herramientas adecuadas y expuestas a ataques en ruta.

Con HTTPS, el tráfico se cifra de manera que, aunque los paquetes sean interceptados, su contenido aparece como caracteres incomprensibles. Veamos un ejemplo.

- **Antes de la encriptación:** una cadena de texto completamente legible.
- **Después de la encriptación:** ITM0IRyiEhVpa6VnKyExMiEgNveroyWBPlgGyfkflYjDaaFf/Kn3bo3OfghBPDWo6AfSHINtL8N7ITEwIXc1gU5X73xMsJormzzXlwOyrCs+9XCPk63Y+z0=.

Además, en sitios web sin HTTPS, proveedores de servicios de Internet (ISP) u otros intermediarios podrían inyectar contenido en las páginas sin autorización del propietario, generalmente en forma de publicidad. Esto puede afectar los ingresos y el control de calidad de los anuncios para el propietario del sitio. HTTPS elimina esta posibilidad, garantizando que solo el contenido autorizado por el propietario sea entregado al usuario final.

Cómo comienza un sitio web a utilizar HTTPS

Muchos proveedores de alojamiento web y servicios relacionados ofrecen certificados TLS/SSL, generalmente de pago. En muchos casos, estos certificados se comparten entre varios clientes. Existen también certificados más costosos que se registran de forma individual para propiedades web específicas, ofreciendo mayor control y validación sobre el dominio.

TLS (transport layer security) —

TLS es un protocolo de seguridad que garantiza la privacidad y la integridad de los datos en las comunicaciones por internet. Su implementación es un estándar en el diseño de aplicaciones web seguras, ya que asegura la autenticidad del servidor y cifra la información transmitida.

El uso más frecuente de TLS es la encriptación de la comunicación entre navegadores y servidores web, aunque también puede aplicarse a correo electrónico, VPN, mensajería y voz sobre IP (VoIP). En este contexto, TLS protege la información frente a interceptaciones y modificaciones no autorizadas.

TLS fue desarrollado por el Internet Engineering Task Force (IETF), una organización internacional de estandarización. La primera versión del protocolo se publicó en 1999, y la versión más reciente, TLS 1.3, se publicó en 2018.

Diferencia entre TLS y SSL

TLS evolucionó a partir de un protocolo de encriptación anterior llamado *secure sockets layer* (SSL), desarrollado por Netscape. La versión 1.0 de TLS comenzó a desarrollarse como la versión 3.1 de SSL, pero el nombre se cambió antes de su publicación para indicar que ya no estaba asociado con Netscape. Debido a esta historia, los términos TLS y SSL a veces se usan de manera indistinta.

Diferencian TLS y HTTPS

HTTPS es una implementación del cifrado TLS sobre el protocolo HTTP, utilizado por todos los sitios web y otros servicios web. Por lo tanto, todos los sitios web que emplean HTTPS utilizan la encriptación proporcionada por TLS.

¿Por qué las empresas y las aplicaciones web usan el protocolo TLS?

La encriptación TLS ayuda a proteger las aplicaciones web frente a fugas de datos y otros tipos de ataques. Hoy en día, el uso de HTTPS protegido por TLS es una práctica estándar en sitios web. Google Chrome

implementó gradualmente medidas contra los sitios que no utilizan HTTPS, y otros navegadores han seguido su ejemplo. Los usuarios habituales de Internet desconfían cada vez más de los sitios que no muestran el ícono de candado que indica HTTPS.

¿Qué hace el protocolo TLS?

El protocolo TLS tiene tres funciones principales: encriptación, autenticación e integridad.

- **Encriptación.** Oculta los datos transferidos para que terceros no puedan acceder a ellos.
- **Autenticación:** asegura que las partes que intercambian información sean realmente quienes dicen ser.
- **Integridad:** verifica que los datos no hayan sido falsificados o modificados durante la transmisión.

Certificado de TLS

Para que un sitio web o una aplicación utilice TLS, debe contar con un certificado TLS instalado en su servidor de origen (este certificado también se denomina a veces «certificado SSL» debido a la confusión histórica entre los términos). Los certificados TLS son emitidos por una autoridad de certificación y están vinculados a la persona o empresa propietaria del dominio. El certificado contiene información clave sobre la propiedad del dominio, así como la clave pública del servidor. Ambos elementos son necesarios para validar la identidad del servidor y establecer comunicaciones seguras.

Funcionamiento del protocolo TLS

La conexión TLS se inicia mediante una secuencia conocida como protocolo de enlace TLS. Cuando un usuario accede a un sitio web que utiliza TLS, se establece este protocolo entre el dispositivo del usuario (dispositivo cliente) y el servidor web.

Durante el protocolo de enlace TLS, el dispositivo del usuario y el servidor web realizan las siguientes acciones:

- especifican la versión de TLS que se utilizará (TLS 1.0, 1.2, 1.3, etc.);
- acuerdan las *suites* de cifrado que se emplearán;
- autentican la identidad del servidor utilizando su certificado TLS;
- generan claves de sesión para cifrar los mensajes intercambiados después de completar el protocolo de enlace.

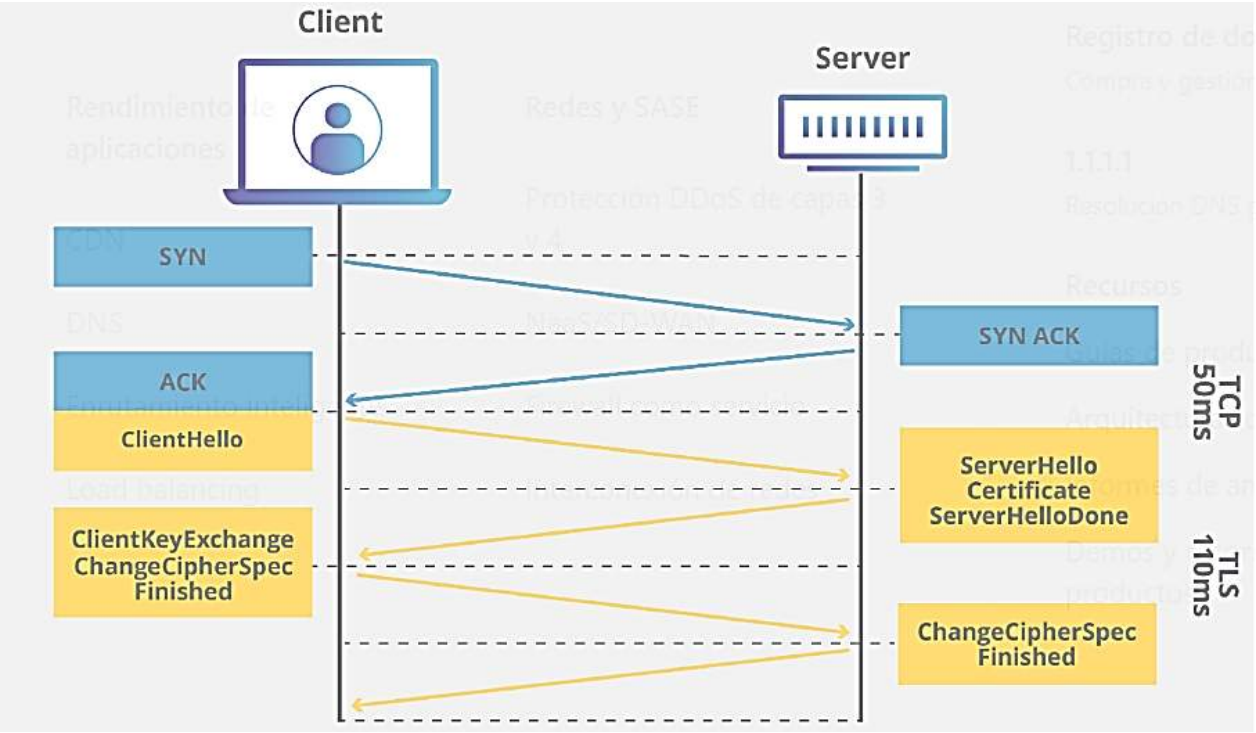
El protocolo de enlace TLS define un **conjunto de cifrado** para cada sesión de comunicación. Este conjunto es un conjunto de algoritmos que determina, entre otros aspectos, qué claves de cifrado y de sesión se utilizarán para esa comunicación. TLS puede asignar las claves de sesión incluso a través de un canal no cifrado, gracias al uso de criptografía de clave pública.

El protocolo de enlace TLS también se encarga de la autenticación, que normalmente consiste en que el servidor demuestre su identidad al cliente mediante el uso de claves públicas. Estas claves son unidireccionales y permiten que cualquier receptor pueda descifrar los datos previamente encriptados con la clave privada del servidor para verificar su autenticidad. Sin embargo, solo el servidor puede cifrar los datos con su clave privada. La clave pública del servidor forma parte de su certificado TLS.

Una vez que los datos han sido cifrados y autenticados, se firman mediante un código de autenticación de mensajes (MAC). El destinatario puede verificar el MAC para asegurarse de que los datos no han sido

alterados. Este mecanismo puede compararse con el cierre de garantía de un medicamento: el consumidor sabe que nadie ha abierto el envase porque el cierre se mantiene intacto al momento de la compra.

Figura 22. Flujo de autenticación y verificación de integridad en TLS



Impacto del protocolo TLS al rendimiento de las aplicaciones web

La versión más reciente de TLS apenas afecta el rendimiento de las aplicaciones web. Debido a la complejidad del proceso para establecer una conexión TLS, su utilización implica cierto tiempo de carga y consumo de recursos. El cliente y el servidor deben intercambiar varios mensajes antes de transmitir datos, lo que suma milisegundos al tiempo de respuesta y requiere algo de memoria en ambos extremos.

Existen, sin embargo, tecnologías que reducen la latencia generada por el protocolo de enlace TLS. Una de ellas es *TLS False Start*, que permite que el servidor y el cliente comiencen a transmitir datos antes de finalizar el protocolo de enlace. Otra es *TLS Session Resumption*, que posibilita que clientes y servidores que ya se han comunicado previamente utilicen un protocolo de enlace abreviado.

Estas optimizaciones han logrado que TLS sea muy rápido y que su impacto en los tiempos de carga sea prácticamente imperceptible. En cuanto a los recursos computacionales necesarios para su funcionamiento, resultan casi insignificantes con respecto a los estándares actuales.

TLS 1.3, publicado en 2018, es aún más eficiente. Sus protocolos de enlace requieren un único recorrido de ida y vuelta, en lugar de dos, reduciendo algunos milisegundos del proceso. Si el usuario ya se conectó anteriormente al sitio web, el protocolo de enlace no requiere ningún recorrido adicional, acelerando todavía más la conexión.

Rastreo y latencia

Comprender el tiempo de respuesta de la red y el recorrido de los paquetes es fundamental para detectar problemas de conectividad o comportamientos anómalos. La latencia es el tiempo que tarda un paquete en viajar desde el origen hasta el destino y se mide en milisegundos (ms). Cuando la latencia es alta, el rendimiento de la conexión disminuye y las comunicaciones se vuelven más lentas.

Para medir y analizar estos parámetros se utilizan herramientas específicas. El comando «ping» permite obtener el tiempo promedio de ida y vuelta hacia un destino, por ejemplo, «ping google.com». Para conocer el recorrido que realiza un paquete a través de Internet se emplean herramientas de rastreo, como «tracert» en Windows o «traceroute» en Linux y macOS, las cuales muestran cada uno de los saltos intermedios.

El análisis de la latencia y del rastreo resulta útil para detectar cuellos de botella en redes empresariales o para verificar si una conexión está siendo direccionada a través del proveedor de servicios de Internet previsto.

CONTINUAR

Unidad 2. Hardering inicial

Windows: usuarios y políticas de seguridad (Microsoft Defender)

Windows es ampliamente utilizado en entornos de pequeñas y medianas empresas. Su seguridad depende de una configuración adecuada de usuarios, políticas y mecanismos internos de protección. El «hardening» consiste en reducir la superficie de ataque mediante configuraciones seguras.

Windows incorpora funciones de seguridad como Microsoft Defender, el Firewall de Windows y el Control de cuentas de usuario (UAC), que ayudan a prevenir accesos no autorizados y a detectar software malicioso. La gestión de usuarios se realiza desde la configuración del sistema, donde es posible crear, eliminar o modificar cuentas con distintos niveles de privilegios. La práctica recomendada es utilizar una cuenta estándar para las tareas cotidianas y reservar la cuenta de administrador para las acciones que realmente lo requieran.

LAS PRINCIPALES HERRAMIENTAS DE SEGURIDAD DE WINDOWS INCLUYEN LAS SIGUIENTES:

- **Seguridad de Windows.** Es el centro de control para la protección del sistema. Desde allí se gestiona el rendimiento y el estado del dispositivo, el firewall, el control de aplicaciones y del navegador, además de la seguridad de la cuenta.
- **Microsoft Defender.** Es el programa de seguridad integrado que protege contra virus, malware, spyware y otras amenazas.
- ***Firewall* de Windows.** Ayuda a impedir accesos no autorizados a la red y al dispositivo, bloqueando conexiones no seguras.
- **Control de cuentas de usuario (UAC).** Solicita la aprobación de un administrador antes de permitir cambios que puedan afectar la configuración del sistema, lo que evita que aplicaciones maliciosas realicen modificaciones sin permiso.
- **BitLocker.** Es la herramienta de cifrado que protege los datos del disco. Resulta útil para resguardar la información en caso de robo o pérdida del dispositivo.

- **Protección del sistema.** Permite crear y restaurar copias de seguridad de archivos y configuraciones, facilitando revertir cambios no deseados o recuperarse de fallos graves.

Gestión de usuarios

En sistemas Windows, los usuarios se agrupan según distintos niveles de permisos. En entornos con baja madurez de seguridad es habitual encontrar malas prácticas, como trabajar de forma permanente desde cuentas administrativas. Windows permite crear y administrar diversos tipos de cuentas, cada una con privilegios específicos:

- **Cuenta de administrador.** Otorga control total sobre el sistema, incluyendo la instalación de software, la gestión de otras cuentas y la modificación de la configuración general.
- **Cuenta de usuario estándar.** Es adecuada para el uso cotidiano. Permite utilizar aplicaciones y administrar la configuración personal, pero no realizar cambios de sistema. Para instalar programas o modificar configuraciones del equipo requiere la contraseña de un administrador.
- **Cuenta de invitado.** Está destinada a usuarios temporales y ofrece un acceso muy limitado al sistema.
- **Perfiles de usuario.** Consisten en un conjunto de archivos y configuraciones que definen el entorno de una cuenta al iniciar sesión, como el escritorio, los documentos y los favoritos.

La gestión de las cuentas se realiza desde la configuración del sistema, en el apartado Cuentas, donde se encuentra la sección «Familia y otros usuarios», que permite crear, eliminar o modificar los permisos de las cuentas existentes.

Buenas prácticas:

Para mantener un entorno seguro en Windows, es recomendable aplicar las siguientes prácticas:

- utilizar cuentas administrativas solo para tareas de configuración;
- crear cuentas estándar para las actividades diarias;
- proteger todas las cuentas mediante contraseñas robustas y, cuando sea posible, habilitar autenticación multifactor.

Políticas de seguridad

En entornos empresariales o avanzados es posible aplicar políticas de seguridad más estrictas para usuarios y grupos. Entre las herramientas disponibles se encuentran las siguientes:

- **Políticas de grupo.** Permiten definir un conjunto de reglas que controlan el entorno de trabajo de los usuarios y de los equipos. Son útiles para establecer restricciones de software, políticas de contraseña y otras configuraciones de seguridad.

- **Directiva de seguridad local.** Es una consola avanzada para administrar reglas y políticas en el ámbito local, como la gestión de claves, la auditoría y la asignación de derechos de usuario.
- **Grupos de usuarios locales.** Permiten organizar usuarios para simplificar la administración de permisos. Al asignar privilegios a un grupo, todos sus integrantes heredan esos permisos.

Políticas mínimas de seguridad

Las políticas establecen el comportamiento del sistema en relación con el acceso, los permisos y la gestión de contraseñas. Entre los lineamientos mínimos se incluyen los siguientes:

- **Acceso al sistema.** Controlar quién puede iniciar sesión y bajo qué condiciones.
- **Contraseñas.** Definir requisitos de complejidad, longitud mínima y períodos de vigencia.
- **Bloqueo de cuenta.** Establecer bloqueos automáticos después de varios intentos fallidos de inicio de sesión.

La herramienta recomendada para administrar estas configuraciones es «secpol.msc» (directiva de seguridad local).

Windows Defender —

Microsoft Defender es la solución nativa de protección contra software malicioso integrada en Windows y complementa sus funciones mediante Microsoft Defender para Endpoint. Esta plataforma combina aprendizaje automático, análisis de grandes volúmenes de datos, investigación sobre resistencia a amenazas y la infraestructura en la nube de Microsoft, con el fin de proteger tanto dispositivos domésticos como equipos de organizaciones.

A partir de 2015 dejó de utilizar un motor estático basado en firmas para adoptar tecnologías predictivas, como el aprendizaje automático, la ciencia aplicada y la inteligencia artificial. Este cambio permite enfrentar con mayor eficacia la complejidad del panorama actual del *malware*, caracterizado por amenazas dinámicas y en constante evolución.

Entre sus capacidades principales se encuentra la detección de anomalías, una capa de protección que no depende de patrones predefinidos. Esta función monitoriza la creación de procesos y la descarga de archivos desde Internet con el objetivo de identificar comportamientos inusuales. Gracias a esta combinación de análisis en la nube y aprendizaje automático, el sistema puede bloquear la mayoría de las amenazas en milisegundos e incluso detener ataques que ya han comenzado su ejecución, como ocurre en el caso del malware sin archivos. Todas estas funciones operan de forma conjunta para detectar y bloquear actividades maliciosas basadas en comportamiento.

Además, ofrece funcionalidades adicionales como:

- análisis en tiempo real;
- *firewall* integrado;
- protección basada en reputación;
- control de aplicaciones y del navegador.

En cuanto a la configuración recomendada, conviene activar el análisis automático diario, definir acciones automáticas para archivos sospechosos e integrar estas configuraciones con las políticas de grupo para consolidar el control en entornos corporativos.

Finalmente, es fundamental mantener actualizado Microsoft Defender, al igual que cualquier solución antivirus o *antimalware* utilizada en el sistema.

Modo activo, modo pasivo y modo deshabilitado: comparación —

A continuación, se describen los modos de funcionamiento de Microsoft Defender y su impacto en la protección del dispositivo.

Cuando el sistema se encuentra en **modo activo**, Microsoft Defender funciona como la aplicación antivirus principal. En este estado analiza archivos, corrige amenazas y registra los incidentes detectados tanto en los informes de seguridad de la organización como en la aplicación Seguridad de Windows.

En **modo pasivo**, el sistema realiza análisis de archivos y notifica las amenazas encontradas, pero no las corrige. Este modo solo está disponible en equipos integrados en Microsoft Defender para Endpoint y resulta útil cuando se utiliza otra solución antivirus como protección principal, aunque se quiera conservar la capacidad de supervisión ofrecida por Microsoft Defender.

Por último, en **modo deshabilitado o desinstalado**, Microsoft Defender no realiza análisis ni acciones de corrección. Debido a que el dispositivo queda sin esta capa de protección nativa, no se recomienda deshabilitarlo ni desinstalarlo.

Linux: Usuarios, servicios y políticas UFW (*uncomplicated firewall*)

Linux es habitual en servidores y plataformas web. Su seguridad inicial depende de una configuración adecuada de cuentas, servicios y del firewall básico. A diferencia de Windows, la administración se realiza principalmente mediante línea de comandos.

La seguridad en Linux se fundamenta en la gestión de usuarios y de servicios, definiendo quién puede acceder y qué puede ejecutar dentro del sistema. El *firewall* UFW es una herramienta que simplifica la configuración de reglas de red, ya que facilita la administración de *iptables* para permitir o bloquear tráfico según las necesidades del entorno. De manera predeterminada, UFW establece una política que deniega todo el tráfico entrante y permite el saliente, aunque estas reglas pueden ajustarse para limitar la superficie de ataque y autorizar únicamente el tráfico necesario.

El sistema operativo Linux ofrece un enfoque de seguridad robusto y flexible, basado en un control estricto de usuarios, permisos y servicios, complementado con herramientas de firewall

como UFW.

- **Seguridad.** Se refuerza aplicando permisos restrictivos a archivos y directorios, gestionando la autenticación de usuarios y utilizando herramientas como firewalls para controlar el tráfico de red.

- **Usuarios.** Linux cuenta con un sistema de gestión de usuarios con distintos niveles de privilegios. Se deben mantener separadas las cuentas administrativas (root) de las cuentas de uso cotidiano, evitando utilizar la cuenta de administrador para tareas diarias.

- **Servicios.** Cada servicio que se ejecuta en un servidor —como un servicio web, una base de datos o SSH— debe mantener abiertos solo los puertos necesarios. UFW permite habilitar o deshabilitar puertos específicos para controlar qué servicios son accesibles desde la red.

Seguridad de usuarios y grupos —

Linux es un sistema multiusuario, lo que significa que varios usuarios pueden interactuar con el sistema al mismo tiempo, cada uno con un conjunto específico de privilegios. La gestión adecuada de usuarios y grupos es fundamental para mantener un entorno seguro.A continuación, se mencionan los archivos principales involucrados en la administración de usuarios y grupos:

- **/etc/passwd.** Contiene información de los usuarios del sistema, como el nombre de usuario, el ID de usuario (UID), el ID de grupo (GID), el directorio de inicio y el *shell* predeterminado.
- **/etc/shadow.** Almacena las contraseñas cifradas y los parámetros de caducidad. Solo el superusuario (root) tiene permiso de lectura sobre este archivo.
- **/etc/group.** Lista los grupos definidos en el sistema y los usuarios que pertenecen a cada uno.

Tipos de usuarios —

En Linux existen distintos tipos de usuarios, definidos por su nivel de privilegios y su función dentro del sistema:

- **Superusuario (*root*).** Tiene un UID de 0 y acceso ilimitado a todos los recursos. Debe utilizarse con extrema precaución, preferiblemente a través del comando «sudo».
- **Usuarios del sistema.** Poseen UID bajos y se utilizan para ejecutar servicios y aplicaciones que no requieren una interfaz de usuario.
- **Usuarios normales.** Tienen UID altos y permisos limitados a su directorio de inicio y a los archivos para los que se les otorgue acceso.

Permisos de archivos

Los permisos en Linux se basan en un modelo de control de acceso discrecional, dividido en tres categorías:

- **Usuario (u).** Permisos del propietario del archivo.
- **Grupo (g).** Permisos para los miembros del grupo asignado al archivo.
- **Otros (o).** Permisos para todos los demás usuarios del sistema.

Cada categoría puede contar con permisos de lectura (r), escritura (w) y ejecución (x). Estos permisos se administran con los comandos «chmod», para modificar permisos, y «chown», para cambiar el propietario o el grupo.

A continuación, se presenta una tabla con ejemplos habituales de combinaciones de permisos:

Tabla 10. Ejemplos comunes de permisos de archivos en Linux

Notación simbólica	Notación numérica	Descripción
-rw-r--r--	644	El propietario puede leer y escribir; el grupo y otros solo pueden leer. Es un permiso común para archivos de texto.
-rwxr-xr-x	755	El propietario puede leer, escribir y ejecutar; el grupo y otros solo pueden leer y ejecutar. Común para directorios y *scripts* ejecutables.
-rw-------	600	Solo el propietario puede leer y escribir. Los demás no tienen ningún permiso. Ideal para archivos privados o de configuración.
drwxr-xr-x	755	Similar al anterior, pero para un directorio. Permite al propietario crear, eliminar y renombrar archivos en el directorio, mientras que el grupo y otros pueden ver su contenido y entrar en él.
-rwxrwxrwx	777	Permite a todos (propietario, grupo y otros) leer, escribir y ejecutar el archivo. Es un permiso muy inseguro y no se recomienda en la

		mayoría de los casos.

Fuente: elaboración propia.

Cómo interpretar los permisos

Para comprender cómo funcionan los permisos en Linux, es importante saber que el sistema representa los accesos mediante tres grupos de caracteres o tres dígitos en notación octal. Cada uno de estos grupos indica qué puede hacer el propietario, el grupo y los demás usuarios. Por ejemplo, los primeros tres caracteres o el primer dígito representan los permisos del propietario; los tres caracteres centrales o el segundo dígito corresponden al grupo; y los últimos tres caracteres o el último dígito indican los permisos para otros usuarios del sistema.

Los permisos se expresan con letras que reflejan acciones concretas sobre un archivo o directorio:

- **r (*read*).** Permite leer el contenido.
- **w (*write*).** Permite modificar el contenido.
- **x (*execute*).** Permite ejecutar el archivo o, si se trata de un directorio, acceder a él.

Cuando se utiliza la notación octal, cada tipo de permiso se traduce en un valor numérico. Estos valores se suman para formar el permiso final de cada categoría:

- **4** para lectura
- **2** para escritura
- **1** para ejecución
- **0** cuando no se asigna ningún permiso

De esta forma, un valor como 7 indica lectura, escritura y ejecución (4+2+1), mientras que un valor como 5 indica lectura y ejecución (4+1). Esta representación es la que se usa en el comando «chmod», muy frecuente en la administración diaria.

A continuación, se presentan algunos ejemplos habituales de asignación de permisos mediante «chmod», junto con una breve explicación de su uso más común:

- **chmod 755 archivo**
  - Propietario: lectura, escritura y ejecución.
  - Grupo: lectura y ejecución.
  - Otros: lectura y ejecución.
  - Uso: apropiado para *scripts* y directorios que deben ser accesibles, pero solo editables por el propietario.
- **chmod 644 archivo**
  - Propietario: lectura y escritura.

    - Grupo: solo lectura.
    - Otros: solo lectura.
    - Uso: típico en archivos de configuración o documentos que solo el propietario debe modificar.
- **chmod 700 archivo**
    - Propietario: lectura, escritura y ejecución.
    - Grupo: sin permisos.
    - Otros: sin permisos.
    - Uso: adecuado para archivos o directorios completamente privados.
- **chmod 600 archivo**
    - Propietario: lectura y escritura.
    - Grupo: sin permisos.
    - Otros: sin permisos.
    - Uso: común en archivos privados que no requieren ejecución, como claves o información sensible.

Ejemplos de permisos con notación simbólica

Además de la notación octal, Linux permite administrar permisos mediante una notación simbólica. Esta utiliza letras para identificar a cada categoría de usuarios y símbolos para añadir, quitar o definir permisos. Es una forma especialmente útil cuando se busca modificar permisos de manera precisa sin recalcular valores numéricos.En esta notación, se emplean las siguientes letras:

- **u** para el propietario,
- **g** para el grupo,
- **o** para otros usuarios, y
- **a** para todos (u+g+o).

Los símbolos usados para modificar permisos son:

- **+** para añadir,
- **-** para eliminar,
- **=** para establecer permisos exactos.

A continuación, se presentan los ejemplos más comunes:

- **chmod a+x archivo.** Agrega el permiso de ejecución a todos los usuarios. Se utiliza cuando se necesita que un script sea ejecutable sin importar quién lo ejecute.
- **chmod g-w,o-w archivo.** Elimina el permiso de escritura tanto para el grupo como para otros usuarios, reforzando la protección del archivo.

- **chmod u=rwx,go= archivo.** Asigna al propietario lectura, escritura y ejecución, mientras que al grupo y a otros no les otorga ningún permiso. Es útil cuando un archivo debe mantenerse completamente privado.
- **chmod -R g+r directorio.** Añade, de forma recursiva, el permiso de lectura para el grupo en todos los archivos y subdirectorios del directorio indicado. Resulta práctico para otorgar acceso de lectura en estructuras de carpetas completas.

Seguridad de servicios

En Linux, los servicios —también llamados demonios— son procesos que se ejecutan en segundo plano y, con frecuencia, escuchan solicitudes a través de puertos de red. Asegurarlos es fundamental para reducir la superficie de ataque y evitar accesos indebidos. En este contexto, conviene tener en cuenta varias medidas esenciales:

- **Desactivar servicios innecesarios.** Es recomendable mantener activos únicamente los servicios indispensables para el funcionamiento del sistema. Esto disminuye la cantidad de posibles puntos de entrada.
- **Actualizar regularmente.** Tanto los servicios como el sistema operativo deben mantenerse actualizados con los parches de seguridad más recientes para corregir vulnerabilidades conocidas.
- **Ejecutar servicios con privilegios mínimos.** Siempre que sea posible, los servicios deben funcionar bajo cuentas sin privilegios. De esta forma, si un servicio comprometido es explotado, el impacto se reduce.
- **Hardening.** Es conveniente aplicar configuraciones de seguridad más estrictas al sistema operativo y a los componentes de red, proceso conocido como *hardening*, para reforzar su protección.
- **Control de acceso.** Herramientas como SELinux o AppArmor permiten implementar un control de acceso obligatorio que limita estrictamente las acciones de cada proceso, incluso si este se ejecuta con permisos elevados.

En la mayoría de las distribuciones modernas, los servicios se gestionan mediante systemd. Algunos comandos habituales son:

- «systemctl status apache2» para consultar el estado de un servicio;
- «systemctl start nginx» para iniciarlo;
- «systemctl disable telnet» para impedir que se inicie de manera automática.

Como medida general, es aconsejable deshabilitar servicios que no se utilicen y revisar qué puertos están abiertos mediante el comando «ss -tulnp», lo que permite identificar servicios activos y evaluar si deben continuar habilitados.

UFW - uncomplicated firewall

UFW es una interfaz sencilla e intuitiva de línea de comandos que permite gestionar las reglas del *firewall iptables* en Linux. Su objetivo es simplificar la administración del firewall, de manera que resulte accesible incluso para usuarios con poca experiencia en seguridad de red. Por este motivo, es especialmente común en distribuciones como Ubuntu y Debian.

El sistema *iptables* es la herramienta de firewall que administra las tablas de filtrado de paquetes del kernel, y sus reglas determinan cómo debe manejarse el tráfico entrante, saliente o reenviado. UFW actúa como una capa que facilita la creación y modificación de estas reglas, evitando la complejidad de trabajar directamente con la sintaxis tradicional de *iptables*.

La correcta gestión de usuarios, servicios y la configuración del firewall son pilares para mantener la seguridad en un sistema Linux, y UFW contribuye a este objetivo proporcionando un mecanismo simple para definir qué tráfico se permite y cuál se bloquea.

Funcionamiento de UFW

UFW funciona como una interfaz para un sistema de firewall más complejo, ya que traduce comandos simples a la sintaxis extensa de *iptables*. Este diseño permite configurar un firewall basado en reglas de manera accesible, sin perder potencia ni flexibilidad. Entre sus características más importantes, se encuentran las siguientes:

- **Comandos sencillos.** UFW se administra mediante órdenes claras y fáciles de interpretar. Por ejemplo, «ufw allow 22» basta para habilitar el puerto 22, a diferencia de la sintaxis más elaborada de «iptables».
- **Políticas predeterminadas.** De manera predeterminada, UFW deniega todas las conexiones entrantes y permite todas las salientes. Este esquema de denegación general crea una base segura que luego puede ajustarse permitiendo únicamente el tráfico necesario.
- **Perfiles de aplicación.** Algunas aplicaciones —como servidores web o servicios remotos— incluyen perfiles específicos para UFW que abstraen los números de puerto. Por ejemplo, «ufw allow OpenSSH» permite habilitar el acceso al servicio SSH estándar sin necesidad de recordar el número de puerto.
- **Compatibilidad con IPv4 e IPv6.** UFW puede gestionar reglas para ambos tipos de tráfico, lo que facilita su uso en entornos mixtos.

- **Interfaz gráfica (GUFW).** Para quienes prefieren una herramienta visual, GUFW ofrece una interfaz gráfica intuitiva para administrar las reglas de UFW.

Algunos comandos básicos

UFW cuenta con un conjunto de comandos simples que permiten administrar el firewall de forma clara y directa. A continuación, se presentan los más utilizados y su función específica:

- **Verificar el estado.** «sudo ufw status» muestra si el firewall está activo o inactivo, además de listar las reglas configuradas. La variante «sudo ufw status verbose» ofrece información más detallada.
- **Activar o desactivar UFW:**
  - «sudo ufw enable» activa el firewall y aplica las reglas disponibles.
  - «sudo ufw disable» desactiva el firewall por completo.
- **Restablecer reglas.** «sudo ufw reset» elimina todas las reglas activas y devuelve UFW a su configuración inicial.
- **Permitir el servicio SSH.** «sudo ufw allow ssh» habilita el acceso al servicio SSH utilizando su perfil de aplicación estándar.

Políticas

UFW permite definir políticas predeterminadas que determinan cómo debe manejarse el tráfico entrante y saliente antes de aplicar cualquier regla explícita. Estas políticas establecen el comportamiento básico del firewall y pueden modificarse según las necesidades del entorno.

Políticas predeterminadas

De manera inicial, UFW adopta un esquema seguro que bloquea todo el tráfico entrante y permite todo el saliente. Las políticas por defecto se gestionan mediante los siguientes comandos:

- «sudo ufw default deny incoming». Deniega todo el tráfico entrante.
- «sudo ufw default allow outgoing». Permite todo el tráfico saliente.

Modificar políticas

En algunos entornos, es necesario aplicar reglas más estrictas y controlar también el tráfico saliente. Para ello, puede emplearse «sudo ufw default deny outgoing», que deniega todo el tráfico saliente por defecto; lo que permite habilitar únicamente las conexiones que sean indispensables.

Reglas de puertos

UFW permite definir reglas para autorizar o bloquear el tráfico en puertos específicos, lo que facilita controlar qué servicios pueden recibir conexiones desde la red. Estas reglas pueden aplicarse tanto a puertos individuales como a direcciones IP concretas.

Para permitir tráfico hacia servicios conocidos, pueden utilizarse los siguientes comandos:

- «sudo ufw allow ssh». Permite conexiones SSH a través del puerto 22.
- «sudo ufw allow http». Permite el tráfico HTTP en el puerto 80.
- «sudo ufw allow 443/tcp». Permite el tráfico HTTPS en el puerto 443 mediante TCP.

También es posible denegar tráfico hacia un puerto determinado usando «sudo ufw deny <puerto>».Cuando se necesita autorizar solo a una dirección IP específica, puede utilizarse «sudo ufw allow from 192.168.1.100 to any port 22».

Gestión avanzada de reglas

UFW permite administrar las reglas de forma más precisa mediante numeración. Para ello, se pueden emplear los siguientes comandos:

- «sudo ufw status numbered» para mostrar las reglas junto a sus números de referencia.
- «sudo ufw delete <número>» para borrar una regla utilizando su número asignado.

Casos de uso para UFW

UFW resulta especialmente adecuado en situaciones donde se requiere una configuración de firewall simple y clara, sin la complejidad habitual de *iptables*. Su diseño lo convierte en una herramienta práctica en distintos escenarios:

- **Computadoras personales y portátiles.** Proporciona una forma sencilla de proteger el dispositivo frente a conexiones entrantes no autorizadas, manteniendo una configuración segura sin necesidad de conocimientos avanzados.
- **Servidores de propósito único.** Cuando un servidor ejecuta una única función —como un servidor web o una base de datos— UFW permite bloquear fácilmente todos los puertos y servicios que no sean necesarios, reduciendo la superficie de ataque.
- **Usuarios nuevos en Linux.** Su sintaxis simple lo convierte en una herramienta ideal para quienes necesitan un firewall básico y seguro sin tener que aprender comandos complejos de *iptables*.

Parches y gestión de cambios

La mayoría de los ciberataques aprovechan vulnerabilidades conocidas para las que ya existen parches. Por este motivo, la gestión de actualizaciones se vuelve esencial para mantener la seguridad operativa. Tanto la gestión de parches como la gestión de cambios forman parte de los procesos de tecnología de la información que garantizan que los sistemas operativos permanezcan seguros y estables.Una gestión integrada de parches y cambios es fundamental para los siguientes objetivos:

- **Ciberseguridad.** Permite proteger los sistemas frente a vulnerabilidades conocidas que podrían ser explotadas por atacantes.
- **Estabilidad y rendimiento.** Contribuye a corregir errores y mejorar la funcionalidad del sistema, asegurando un funcionamiento fluido.
- **Cumplimiento.** Mantiene los sistemas actualizados para cumplir con regulaciones y estándares de seguridad vigentes.
- **Reducción de riesgos.** Minimiza el impacto negativo de los cambios en el entorno, gracias a una planificación adecuada y a la realización de pruebas antes de aplicar modificaciones.

Gestión de parches

La gestión de parches en sistemas operativos consiste en adquirir, probar e instalar actualizaciones de software para corregir fallos, mejorar el rendimiento y, especialmente, solucionar vulnerabilidades de seguridad. Esta práctica forma parte de la gestión de cambios dentro de un entorno de TI, cuyo objetivo es introducir modificaciones con el menor impacto posible en la operación diaria.

A continuación, se mencionan los principales aspectos de la gestión de parches:

- **Proceso.** Incluye identificar, adquirir, probar e implementar los parches necesarios para el sistema.
- **Finalidad.** Busca corregir fallos de seguridad, errores de programación y añadir mejoras de funcionamiento o nuevas funcionalidades.
- **Beneficios.** Protege frente a ciberataques, reduce tiempos de inactividad y contribuye a cumplir normas de seguridad como el RGPD.
- **Diferencias entre sistemas.** Aunque el concepto es el mismo, la implementación puede variar. Por ejemplo, en Linux suele requerirse el uso de comandos en una terminal, mientras que en Windows la gestión puede realizarse mediante una interfaz gráfica.

Tipos de parches según su criticidad

Los parches pueden clasificarse según el nivel de urgencia con el que deben aplicarse. Esta clasificación ayuda a priorizar la implementación y a reducir la exposición a riesgos innecesarios.

- **Críticos.** Corrigen fallos que podrían permitir la ejecución remota de código, lo que representa uno de los riesgos más elevados para cualquier sistema.
- **Importantes.** Abordan vulnerabilidades de gravedad alta que, si bien no permiten ejecución directa de código, pueden comprometer la seguridad o la estabilidad del sistema.
- **Opcionales.** Incluyen mejoras menores o ajustes de estabilidad que no afectan directamente la protección del sistema.

Tipos de parches según sus características

Además del nivel de criticidad, los parches pueden dividirse según su propósito o alcance dentro del sistema:

- **Parches de seguridad.** Son los más relevantes desde el punto de vista de la protección del sistema. Corrigen vulnerabilidades que podrían ser explotadas para acceder a información sensible o alterar la integridad del entorno.
- **Parches de corrección (*hotfixes*).** Resuelven errores específicos que afectan la funcionalidad o la estabilidad del software, permitiendo restaurar su correcto funcionamiento.
- **Parches de actualización (*service packs*).** Añaden nuevas características, mejoran funciones existentes o agrupan varios parches ya publicados en un único paquete de instalación.

- **Parches acumulativos.** Incluyen todas las actualizaciones de seguridad y corrección liberadas hasta ese momento, de modo que reemplazan versiones anteriores y simplifican el mantenimiento.

Ciclo de vida de la gestión de parches

Una gestión eficaz de parches requiere un proceso estructurado que garantice que las actualizaciones se apliquen de forma segura y ordenada. A continuación, se describen las etapas principales:

- **Inventario.** Se elabora un registro actualizado de todos los dispositivos, sistemas operativos y aplicaciones que requieren mantenimiento.
- **Identificación.** El equipo de TI monitorea las notificaciones de los proveedores y las fuentes de seguridad para detectar nuevos parches o vulnerabilidades que puedan afectar a los sistemas.
- **Evaluación.** Se clasifican los activos y los parches según su nivel de riesgo y prioridad. En esta etapa se analiza la criticidad del parche, las vulnerabilidades que corrige y los sistemas involucrados.
- **Adquisición.** Se obtiene el parche directamente del proveedor o mediante herramientas específicas de gestión.
- **Prueba.** Antes de su implementación, el parche se prueba en un entorno controlado para verificar que no genere conflictos, errores o inestabilidad con otras aplicaciones o configuraciones.
- **Implementación.** Se aplican los parches de acuerdo con el plan definido, ya sea en grupos específicos o en toda la red. Esta aplicación puede hacerse de forma manual o, preferentemente, de manera automatizada para asegurar una cobertura consistente.
- **Verificación.** Se confirma que los parches se instalaron correctamente y que el sistema funciona según lo esperado.
- **Monitoreo y documentación.** Se supervisan continuamente los sistemas y se documentan todas las etapas del proceso, generando informes que permiten verificar el cumplimiento y mejorar las prácticas a futuro.

Buenas prácticas

A continuación, se mencionan algunas pautas recomendadas para aplicar parches de manera segura y ordenada:

- Automatizar actualizaciones en los endpoints.
- Validar parches en un entorno de prueba, cuando sea posible.
- Mantener una bitácora de actualizaciones críticas.
- Revisar semanalmente los parches críticos.
- Aplicar los parches en horarios de baja actividad.

- Confirmar que el sistema funcione correctamente después del parcheo.

Gestión de cambios

La gestión de cambios abarca el proceso mediante el cual se administran las modificaciones realizadas en los sistemas de producción, con el fin de asegurarse de que se apliquen de manera planificada y controlada. Dentro de este proceso se incluyen mejoras, actualizaciones y, de forma central, la aplicación de parches. A continuación, se mencionan sus aspectos principales:

- Gestiona todas las mejoras, actualizaciones, correcciones y parches aplicados a los sistemas de producción, incluidos los cambios en el código de las aplicaciones.
- La gestión de parches constituye una parte esencial dentro de la gestión de cambios.
- Su finalidad es garantizar que las actualizaciones se implementen de forma ordenada para evitar interrupciones.
- Requiere comunicación entre equipos, procesos documentados para aplicar o revertir cambios y revisiones periódicas de las políticas.

En cuanto a las buenas prácticas vinculadas a este proceso, se recomienda:

- documentar qué se modifica, cuándo y por qué;
- contar con un plan de reversión en caso de que el cambio genere fallos.

Ciclo de vida de la gestión de cambios

El ciclo de vida de la gestión de cambios describe las etapas que deben seguirse para aplicar modificaciones en un entorno de TI de manera controlada. A continuación, se mencionan sus fases principales:

- **Planificación.** Antes de aplicar un parche, realizar una configuración, reiniciar un sistema crítico o efectuar una reparación, se define un plan que detalle el alcance, los objetivos, el cronograma y los riesgos asociados.
- **Comunicación.** Se informa a los equipos afectados acerca del cambio, sus motivos y los tiempos previstos, con el fin de evitar sorpresas y reducir resistencia.
- **Control de versiones.** Se mantiene un registro de las versiones de software y de los parches instalados, permitiendo revertir cualquier cambio problemático.
- **Seguimiento.** Una vez implementado el cambio, se monitorea su comportamiento para verificar que no aparezcan fallos nuevos y se documentan los resultados obtenidos.
- **Reversión.** Se dispone de un plan de contingencia para deshacer el cambio en caso de efectos adversos, como la inestabilidad del sistema.

Backups 3-2-1 y restauración

Backups **(copias de seguridad)**

La copia de seguridad es la última línea de defensa en la protección de datos. Consiste en crear duplicados de la información para resguardarla frente a pérdida, corrupción, robo, errores humanos, fallos de hardware o ciberataques. Un *backup* es una réplica almacenada en un lugar distinto del origen, que permite restaurar los datos cuando ocurre un incidente. La implementación de *backups* es fundamental tanto para usuarios individuales como para organizaciones, ya que garantiza la continuidad operativa y minimiza el impacto ante cualquier contingencia.

TIPOS DE BACKUPS | OTROS TIPOS DE BACKUPS | ESTRATEGIA DE BACKUP 3-2-1

Existen varios tipos de *backups*, cada uno con un enfoque diferente para equilibrar la velocidad, el espacio de almacenamiento y la complejidad de la restauración.

- *Backup* **completo**

El *backup* completo, o *full backup*, consiste en copiar absolutamente todos los archivos y datos del sistema. Es el método más confiable y sencillo, ya que toda la información necesaria se encuentra en un único respaldo, lo que permite restaurar los datos de manera rápida y directa. Sin embargo, requiere mucho tiempo para realizarse y consume una gran cantidad de espacio de almacenamiento, por lo que no suele emplearse todos los días cuando se manejan grandes volúmenes de información. Generalmente, se utiliza como punto de partida en una estrategia de copias de seguridad o de forma periódica —por ejemplo, semanal o mensualmente— para establecer una base sólida sobre la cual se apoyarán otros tipos de *backups*.

- *Backup* **incremental**

El *backup* incremental guarda únicamente los datos que han sido creados o modificados desde el último *backup* realizado, sea completo o incremental. Este método destaca por su rapidez y

por el uso eficiente del espacio de almacenamiento, ya que solo registra los cambios ocurridos desde la copia más reciente. Por su naturaleza, es ideal para realizar copias frecuentes sin sobrecargar recursos. Su principal desventaja aparece al momento de la restauración: para recuperar todos los datos se necesita disponer de la última copia completa y de cada una de las copias incrementales realizadas en orden secuencial. Si una de ellas falla o está incompleta, toda la restauración puede verse comprometida.

- ***Backup* diferencial**

El *backup* diferencial toma como referencia exclusivamente el último *backup* completo y copia todos los archivos que hayan cambiado desde ese momento. Esto simplifica el proceso de restauración, ya que solo se necesita la copia completa inicial y el último *backup* diferencial para recuperar toda la información. La contracara es que, a medida que pasan los días, el tamaño del *backup* diferencial crece continuamente hasta que se realiza un nuevo *backup* completo, por lo que ocupa más espacio que los incrementales. Aun así, representa un equilibrio adecuado entre velocidad de copia y facilidad de restauración, por lo que suele emplearse como alternativa intermedia en estrategias de respaldo diarias.

Además de los métodos principales, existen otros enfoques que complementan y optimizan la gestión de copias de seguridad.

- El ***backup* en espejo** crea una réplica exacta de los datos en su estado actual, manteniendo los mismos archivos sin conservar versiones anteriores. La restauración es rápida, pero cualquier cambio o eliminación en el origen se refleja inmediatamente en la copia.
- El ***backup* sintético completo** genera una copia completa combinando un respaldo total previo con los incrementales posteriores, sin volver a copiar toda la información desde el sistema original. Esto acelera la creación de copias completas y reduce la carga sobre el equipo.
- El ***backup* continuo** realiza copias de seguridad de forma constante, registrando los cambios casi en tiempo real. Es adecuado para sistemas críticos donde no se puede tolerar pérdida de datos, ya que mantiene siempre una versión actualizada de la información.

Para facilitar la elección de la estrategia adecuada, la siguiente tabla presenta una comparación clara entre los distintos tipos de *backups* y sus características principales.

Tabla 11. Comparación entre los distintos tipos de *backups*

Tipo de *backup*	**Velocidad de copia**	**Espacio de almacenamiento**	**Velocidad de restauración**	**Complejidad de restauración**
Completo	Lenta	Grande	Rápida	Baja

Incremental	Rápida	Pequeño	Lenta	Alta
Diferencial	Media	Medio (creciente)	Media	Baja
Espejo	Rápida	Grande	Muy rápida	Baja

Fuente: elaboración propia.

La estrategia de backup 3-2-1 es una de las más utilizadas por su equilibrio entre simplicidad y alta protección. Propone mantener varias copias de los datos distribuidas en distintos medios y ubicaciones para reducir al mínimo el riesgo de pérdida, ya sea por fallos de *hardware*, errores humanos, desastres físicos o ataques como el *ransomware*. Veamos en qué consiste esta regla:

- **3 copias de los datos:** Incluyen el archivo original y dos copias adicionales. Tener múltiples duplicados aporta redundancia y garantiza alternativas si una copia se daña o se vuelve inaccesible.
- **2 formatos de almacenamiento distintos:** Las copias deben guardarse en medios diferentes (por ejemplo, disco local y nube). Esto evita que un fallo en un único tipo de dispositivo afecte a todas las copias.
- **1 copia fuera del sitio:** Una de las copias debe ubicarse en un lugar físico separado, como almacenamiento en la nube o en otro edificio. Así se protege la información ante eventos que puedan afectar al entorno local, como incendios, robos o inundaciones.

Consideraciones adicionales

Dado el surgimiento de amenazas avanzadas como el *ransomware*, la clásica regla 3-2-1 ha evolucionado para reforzar la resiliencia de las copias de seguridad. Estas variantes incorporan conceptos como la inmutabilidad y la verificación continua, elementos que aumentan considerablemente la protección ante ataques que buscan cifrar o destruir datos.

- **Regla 3-2-1-1-0.** Añade dos principios: mantener al menos **una copia inmutable**, que no pueda modificarse ni eliminarse, y garantizar **cero errores** en las pruebas de recuperación. Esto asegura que, incluso ante un ataque, siempre exista una copia íntegra y verificable.
- **Regla 3-2-1-M:** introduce la «M», que hace referencia a una copia mejorada o inmutable, reforzando aún más la protección frente a manipulaciones o borrado malintencionado.

- ***Ransomware*:** estas variantes se vuelven esenciales dado que los ataques modernos suelen cifrar, alterar o incluso robar información. Contar con copias en distintos medios, ubicaciones y estados (incluyendo inmutabilidad) es la forma más fiable de garantizar la recuperación completa.

Ejemplo de aplicación de la regla 3-2-1

Imaginemos una empresa que gestiona sus datos más críticos del siguiente modo:

- **Copia principal.** Los datos se almacenan en un servidor local dentro de la oficina.
- **Primera copia de seguridad (medio diferente):** se genera una copia automática en un dispositivo de almacenamiento conectado a la red (NAS).
- **Segunda copia de seguridad (fuera de sitio):** una copia adicional se sincroniza de forma automática con un servicio de almacenamiento en la nube, como «AWS S3», «Google Drive» o un servicio privado.

Con este enfoque, la empresa mantiene su información protegida incluso si:

- el servidor principal falla;
- el dispositivo NAS sufre un daño;
- un incendio o un robo destruye tanto el servidor como el dispositivo local.

Restauración

La restauración es el proceso mediante el cual se recuperan los datos desde las copias de seguridad, con el fin de volver a un estado operativo tras un fallo, un ataque o cualquier incidente que afecte la información. Para que este proceso sea fiable, intervienen varios aspectos:

- **Comprobación regular:** Consiste en probar la restauración de forma periódica para confirmar que las copias están íntegras y que el procedimiento funciona correctamente.
- **Proceso de recuperación:** Implica seleccionar la copia de seguridad adecuada y restaurar los datos en el sistema correspondiente, ya sea de forma parcial o completa.
- **Flexibilidad del sistema:** Según la solución implementada, la restauración puede limitarse a un archivo puntual o abarcar todo un sistema operativo con su configuración.

La importancia de la restauración en la regla 3-2-1

La estrategia 3-2-1 solo resulta eficaz si los datos pueden recuperarse cuando ocurre un incidente. La utilidad real de una copia de seguridad no está en su creación, sino en su capacidad de restauración. Por ello, un plan de recuperación debe considerar algunos aspectos fundamentales:

- **Pruebas periódicas**

Es necesario realizar simulacros de restauración con regularidad para comprobar que las copias de seguridad son válidas y restaurables. Esta verificación permite asegurarse de que el proceso funciona correctamente y de que los datos no presentan daños.

- **Objetivos de recuperación (RTO y RPO)**
  - **Tiempo objetivo de recuperación (RTO).** Se debe establecer el tiempo máximo aceptable para restaurar los datos después de una pérdida.
  - **Punto objetivo de recuperación (RPO).** Se debe definir la cantidad de datos que puede perderse, es decir, cuán recientes deben ser las copias de seguridad.
- **Procedimientos claros:** Es recomendable contar con un procedimiento documentado y accesible que indique cómo y desde dónde realizar la restauración en caso de emergencia.
- **Versiones de *backup*:** Conviene disponer de una estrategia de retención de versiones que permita restaurar datos de distintos momentos, lo que resulta útil para resolver casos de corrupción o eliminación accidental replicados en las copias más recientes.
- **Automatización:** Es aconsejable utilizar software que automatice las tareas de respaldo y restauración para reducir errores humanos y mantener la consistencia del proceso.

CONTINUAR

Referencias

Cloudflare, (s.f.). *¿Qué ocurre durante un protocolo de enlace TLS? | Protocolo de enlace* SSL. https://www.cloudflare.com/es-es/learning/ssl/what-happens-in-a-tls-handshake/

Gil Vázquez, P., Pomares Baeza, J., & Candelas Herías, F. A. (2010). *Redes y transmisión de datos*. Universidad de Alicante — Servicio de Publicaciones.

Instituto Nacional de Ciberseguridad [INCIBE]. (2025). *Balance de ciberseguridad 2024*. https://www.incibe.es/incibe/sala-de-prensa/incibe-presenta-su-balance-de-ciberseguridad-2024-con-mas-de-97000-incidentes

Kurose, J. F., & Ross, K. W. (2017). *Redes de computadoras: un enfoque descendente* (7.ª ed.). Pearson

Organización Internacional de Normalización. (1994). *Tecnología de la información: Interconexión de sistemas abiertos: Modelo de referencia básico* (Norma ISO/IEC 7498-1). https://www.iso.org/standard/20269.html

Stallings, W. (2000). *Comunicaciones y redes de computadores* (6.ª ed.). Prentice Hall.

StormWall, (s.f.). *OSI vs TCP/IP Models: What's the Difference?* https://stormwall.network/resources/blog/osi-vs-tcp-ip

Tech Buyer, (s.f.). *OSI vs. TCP/IP*. https://www.techbuyer.com/uk/blog/osi-vs-tcp-ip

Tanenbaum, A. S. (2003). *Redes de computadoras* (4.ª ed.). Prentice Hall.

CONTINUAR