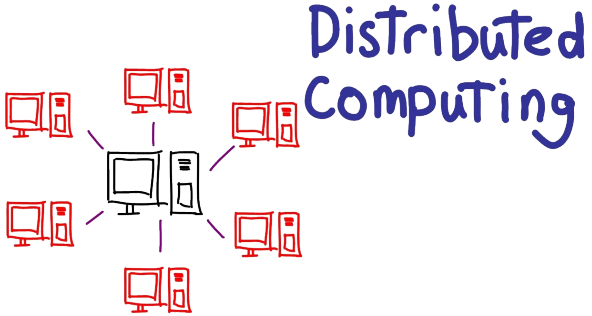
Programación de Procesos y Servicios

UT3: Programación en Red
2º DAM

Computación Distribuida

Computación Distribuida: Modelo de computación en el que los recursos (como procesadores, almacenamiento y redes) están distribuidos en diferentes ubicaciones físicas pero trabajan juntos para realizar tareas o resolver problemas. Así, los sistemas distribuidos:

- Están formados por más de un elemento computacional distinto e independiente.
- Los elementos que forman el sistema distribuido no están sincronizados entre sí.
- Están conectados a una red de comunicaciones que les permite comunicarse entre sí.



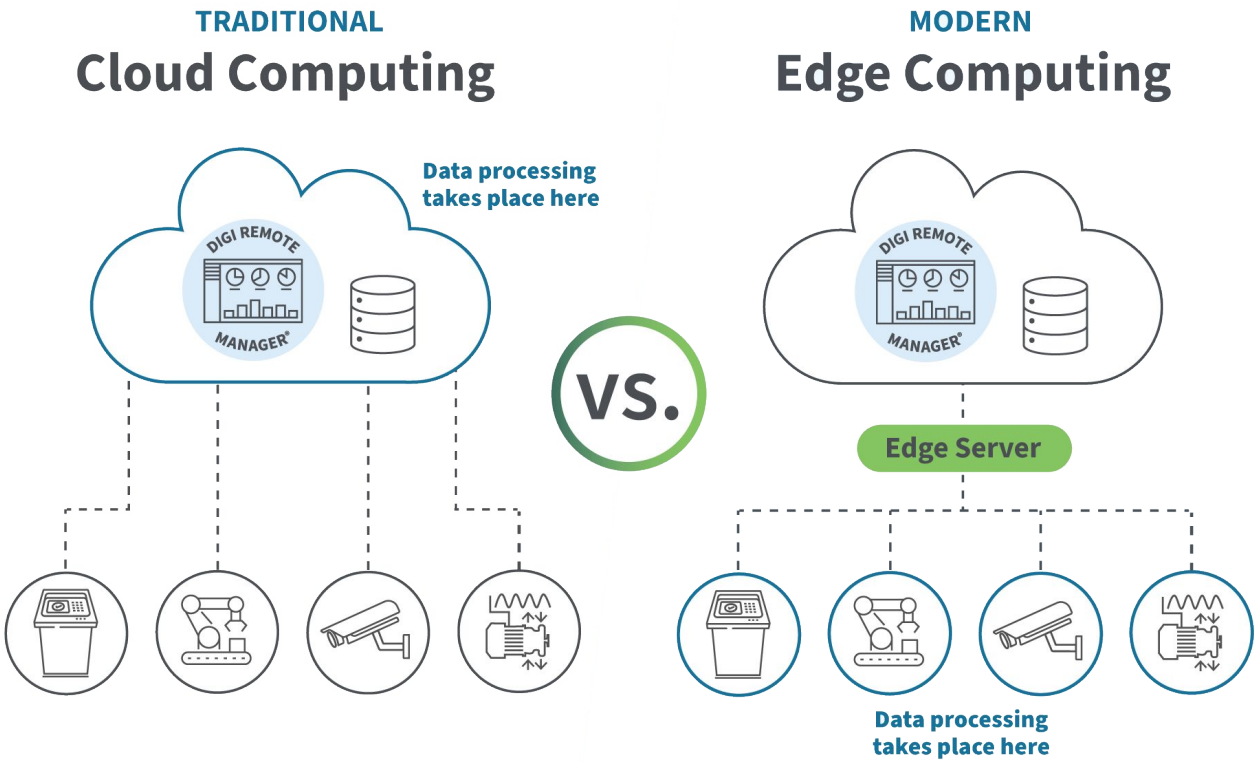
Computación Distribuida

Existen dos modelos de sistemas distribuidos:

- Modelo Cliente/Servidor: Un proceso, denominado servidor, ofrece servicios a uno o más procesos, denominados clientes.
- Peer to Peer (P2P): Todos los procesos colaboran de forma similar y con un mismo fin, no existiendo una especialización ni diferenciación entre ellos..

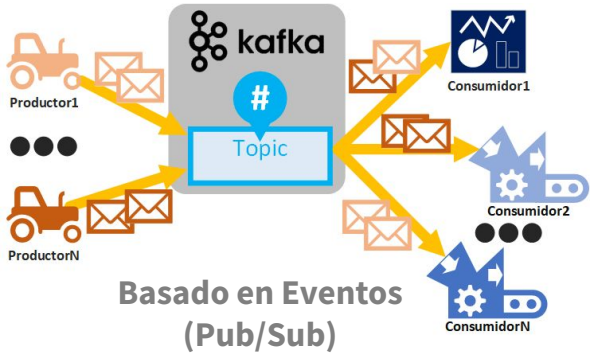
Paradigmas de programación distribuida

En función de dónde se produce el cómputo:

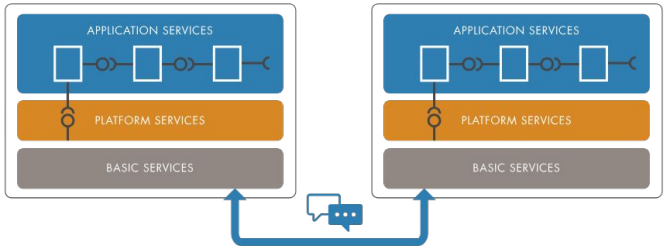


Paradigmas de programación distribuida

En función del modelo de computación:

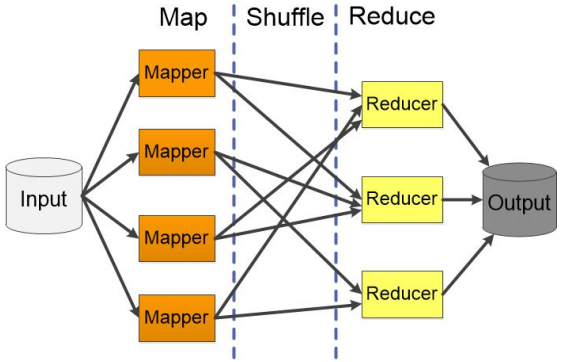


Basado en Eventos
(Pub/Sub)



Programación Basa en Servicios

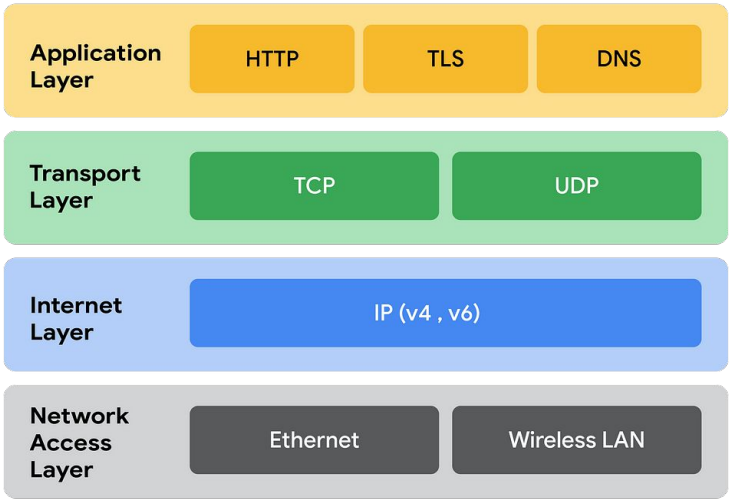
Map/Reduce
Programación Funcional



Pila de Protocolos TCP/IP

Pila de Protocolos TCP/IP: Conjunto de protocolos de comunicación utilizados para interconectar dispositivos en redes, incluyendo Internet.

Recibe el nombre de pila o stack porque está diseñado como una jerarquía de capas en las que cada capa da soporte a la capa que tiene por encima y utiliza los servicios de la capa que tiene por debajo.



Pila de Protocolos TCP/IP

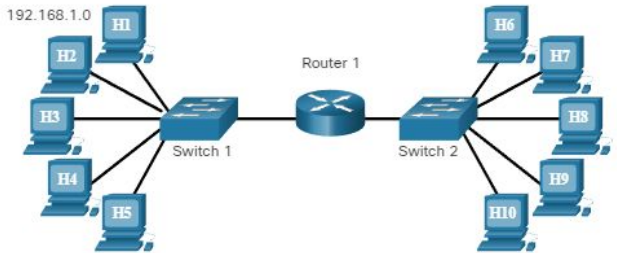
Capa de Enlace: Se ocupa de la transmisión física de datos a través de medios como Ethernet, Wi-Fi y otros. Se encarga de cómo los datos se envían a través de una red local y gestiona la conexión a los dispositivos físicos. Funciones:

- Transmisión física
- Dirección MAC
- Control de acceso al medio
- Detección y Corrección de errores
- Fragmentación y Reensamblaje
- Encapsulación

Pila de Protocolos TCP/IP

Capa de Red: Esta capa es responsable del direccionamiento y el enrutamiento de los paquetes de datos a través de la red, proporcionando la interconexión entre diferentes redes. Protocolos:

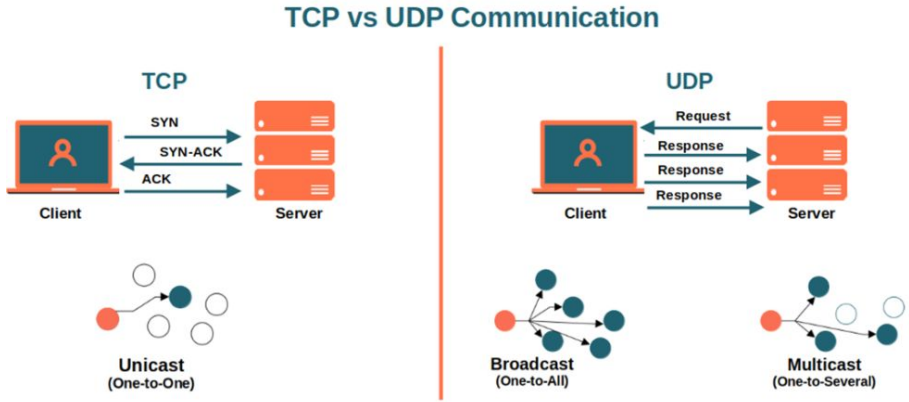
- IP: Se encarga de dirigir los paquetes de datos hacia su destino a través de la red, usando direcciones IP
- ICMP: Utilizado principalmente para enviar mensajes de error y de diagnóstico de red.
- ARP: Convierte las direcciones IP a direcciones físicas (MAC) en redes locales, permitiendo que los dispositivos se encuentren en la misma red física.



Pila de Protocolos TCP/IP

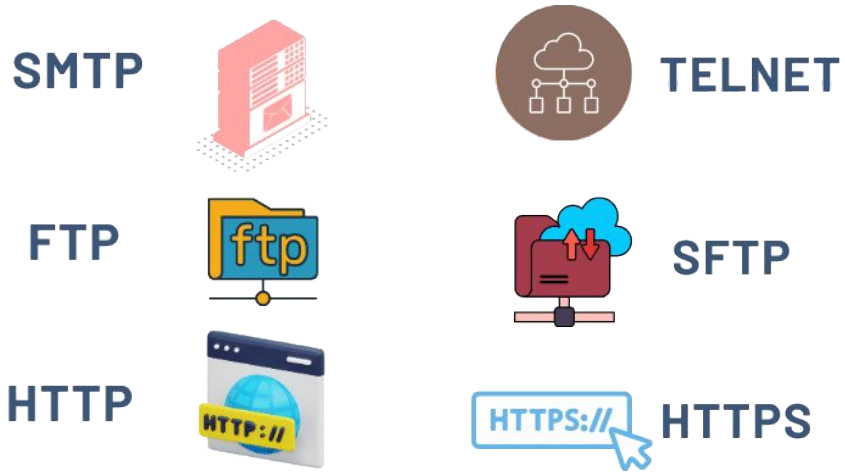
Capa de Transporte: Esta capa proporciona la comunicación fiable y ordenada entre las aplicaciones de los diferentes hosts. Aquí se encuentran dos de los principales protocolos de la pila TCP/IP:

- TCP: Proporciona una comunicación confiable, con control de flujo y corrección de errores.
- UDP: Es un protocolo de transporte no confiable y sin conexión, pero más rápido que TCP, ya que no realiza control de errores ni control de flujo.



Pila de Protocolos TCP/IP

Capa de Aplicación: Es la capa más alta del modelo y proporciona servicios de red a las aplicaciones del usuario. Se encarga de determinar cómo las aplicaciones deben comunicarse a través de la red. También define la estructura de los datos y establece el contexto para las aplicaciones que interactúan.



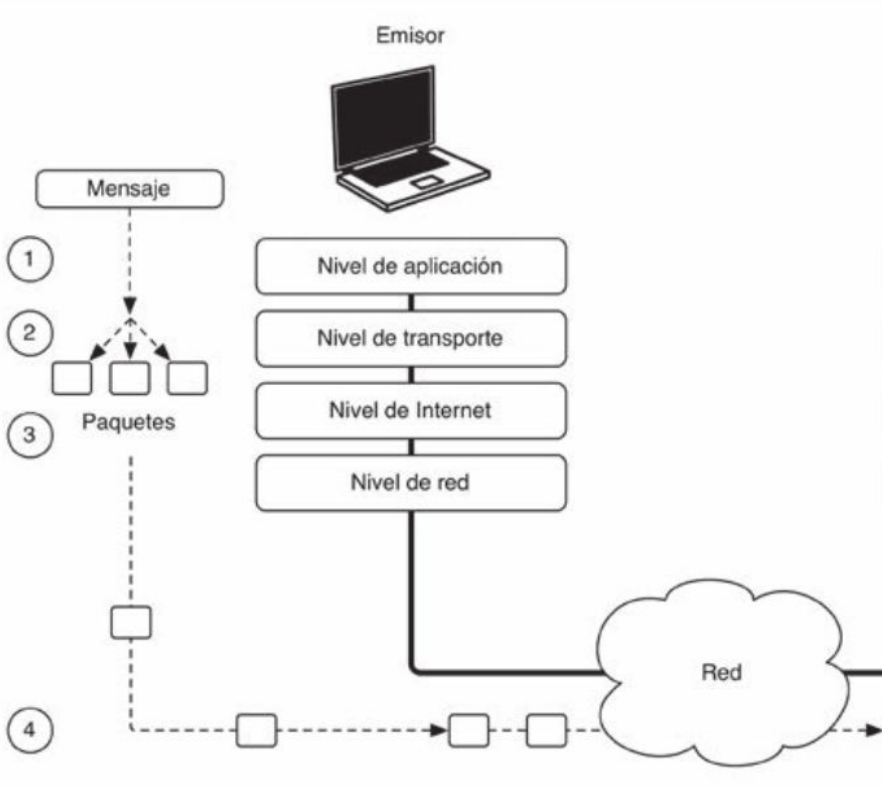
Pila de Protocolos TCP/IP: Funcionamiento

1. La aplicación en el emisor envía el mensaje al nivel de transporte, que es el nivel inmediatamente inferior.

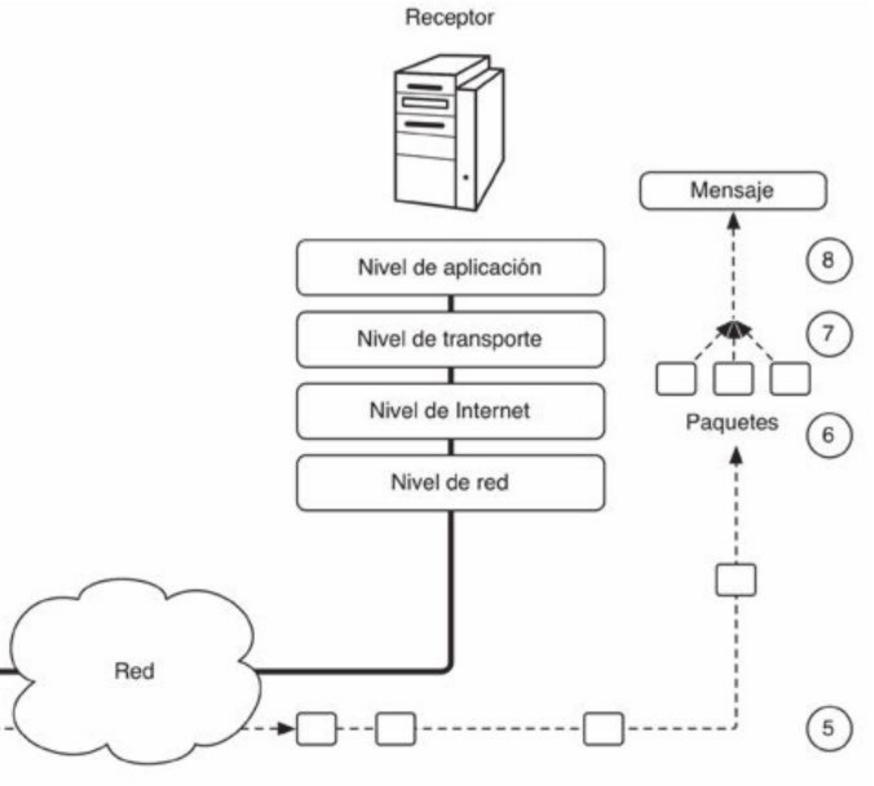
2. El protocolo de la capa de transporte fragmenta el mensaje en paquetes y los envía a la capa inferior, que es la capa de Internet.

3. La capa de Internet localiza la dirección del receptor y calcula la ruta que seguirán los paquetes para llegar a destino, luego entrega los paquetes a la capa inferior (capa de red).

4. La capa de red se encarga de transmitir los paquetes hasta que lleguen al receptor.



Pila de Protocolos TCP/IP: Funcionamiento



5. Al llegar al receptor, la capa de red recibe los paquetes y los pasa a la capa superior (capa de Internet).

6. La capa de Internet verifica que los paquetes hayan llegado al destinatario correcto y, si es así, los pasa a la capa superior (capa de transporte).

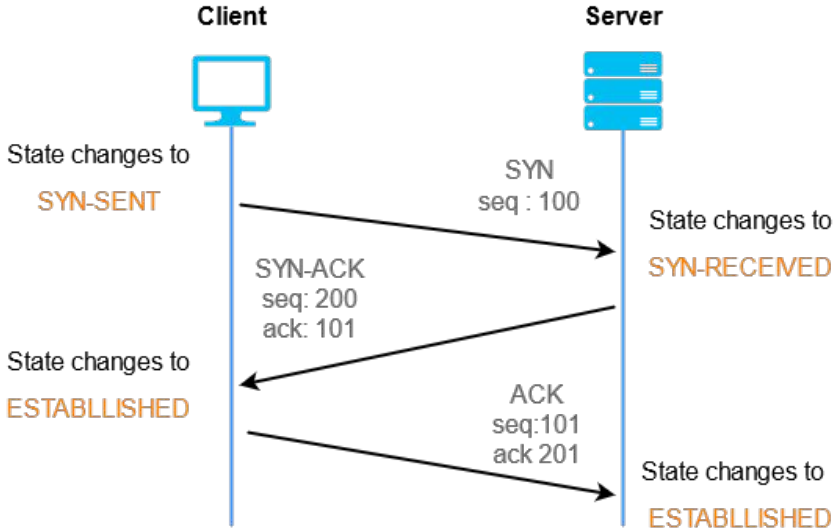
7. La capa de transporte vuelve a ensamblar los paquetes recibidos para reconstruir el mensaje completo y lo envía a la capa superior (capa de aplicación).

8. Finalmente, la aplicación receptora recibe el mensaje reconstruido.

Protocolo TCP

El **protocolo TCP** garantiza la transmisión confiable de datos.

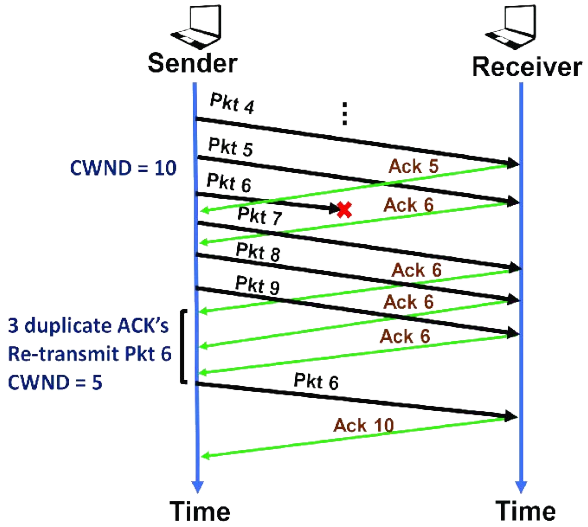
- Orientado a conexión: La conexión se realiza mediante el proceso de handshake de tres vías, que garantiza que ambas partes estén sincronizadas y listas para la comunicación.



Protocolo TCP

El **protocolo TCP** garantiza la transmisión confiable de datos.

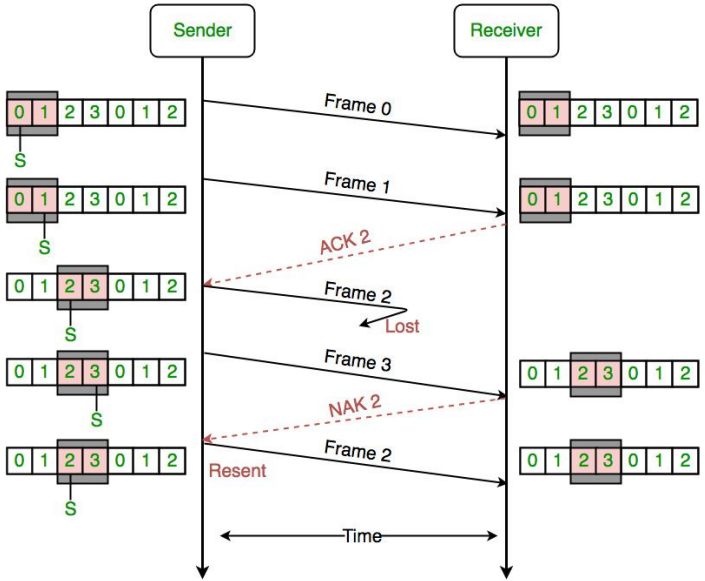
- Transmisión fiable: Garantiza que los datos lleguen en el mismo orden en que fueron enviados e implementa control de errores, verificando que todos los segmentos de datos lleguen correctamente.



Protocolo TCP

El **protocolo TCP** garantiza la transmisión confiable de datos.

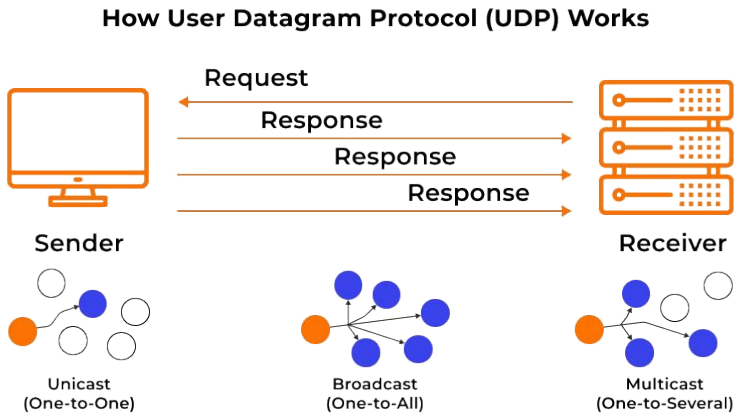
- Control de flujo y congestión: Regula la velocidad de envío de datos para no sobrecargar la red ni al receptor. Utiliza técnicas de ventana deslizante para ajustar la cantidad de datos enviados en función de la capacidad del receptor y las condiciones de la red.



Protocolo UDP

El **protocolo UDP** ofrece una comunicación rápida y de baja latencia entre dispositivos en la red.

- Sin conexión: No establece una conexión antes de enviar los datos, lo que permite una transmisión más rápida al eliminar los pasos iniciales que requiere TCP. Los paquetes se envían independientemente y no hay confirmación de recepción, lo que significa que el remitente no sabe si el receptor ha recibido el paquete.

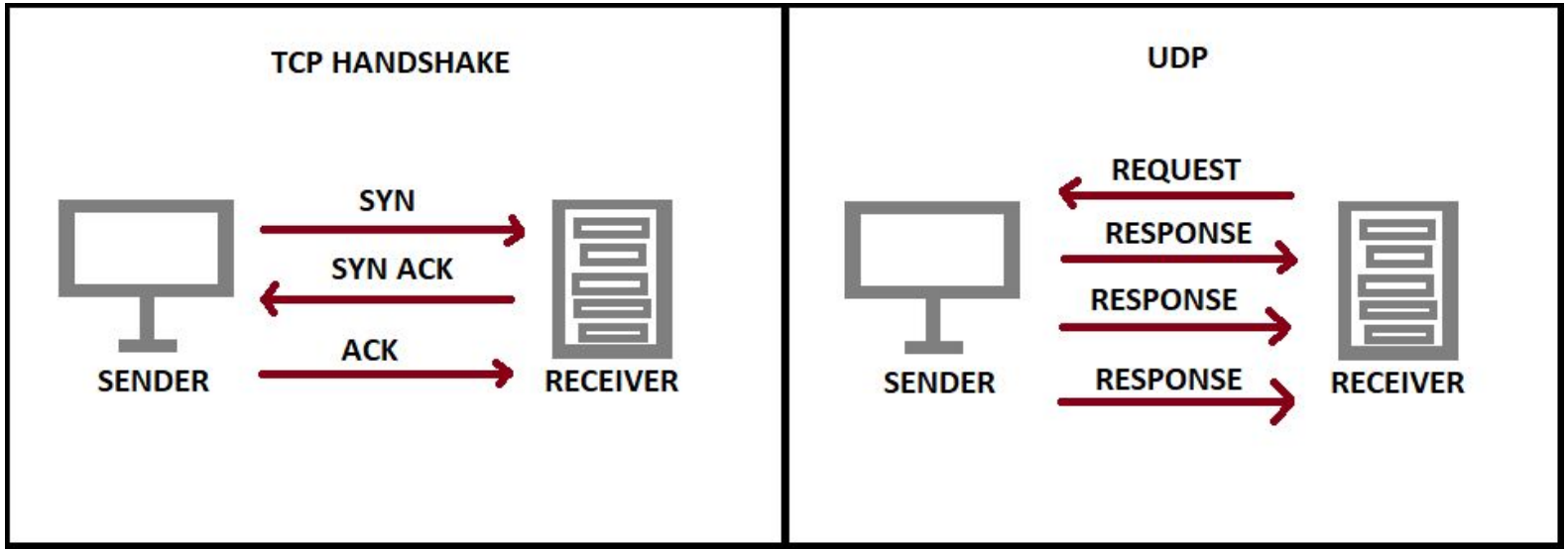


Protocolo UDP

El **protocolo UDP** ofrece una comunicación rápida y de baja latencia entre dispositivos en la red.

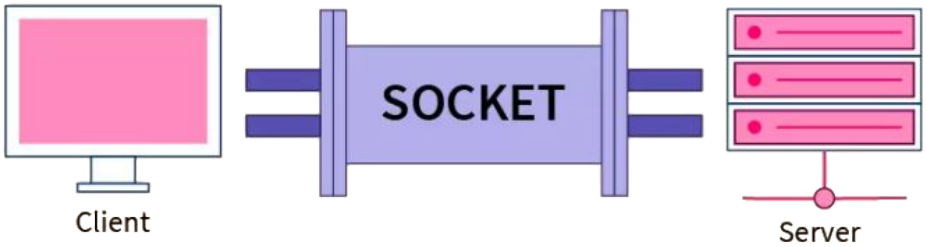
- Transmisión no fiable: No garantiza la entrega de los datos ni su orden. Esto hace que sea ideal para aplicaciones donde la velocidad y la baja latencia son más importantes que la fiabilidad.
- Sin control de flujo ni congestión: UDP no implementa mecanismos de control de flujo ni de congestión, lo que permite una transmisión continua de paquetes, pero puede resultar en una sobrecarga de la red si se envían demasiados datos demasiado rápido.

TCP vs UDP


TCP HANDSHAKE
SYN
SYN ACK
ACK
SENDER
RECEIVER
UDP
REQUEST
RESPONSE
RESPONSE
RESPONSE
SENDER
RECEIVER

Sockets

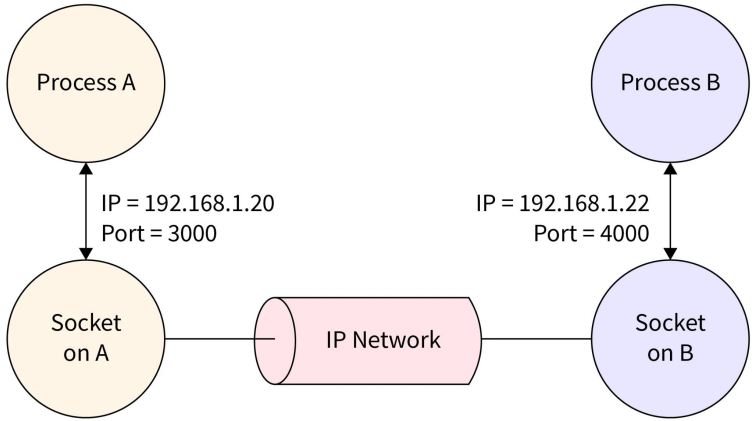
Un **socket** es la puerta de entrada a la transmisión y recepción de datos entre dispositivos, ya sea en una máquina local o a través de internet. Es una interfaz de programación de aplicaciones (API) que proporciona un mecanismo para que los procesos de software se comuniquen entre sí a través de una red



Sockets

Un **socket** representa el extremo de un canal de comunicación establecido entre un emisor y un receptor.

- Para establecer una comunicación entre dos aplicaciones, ambas deben crear sus respectivos sockets, y conectarlos entre sí.
- Una vez conectados, entre ambos sockets se crea una "tubería privada" a través de la red, que permite que las aplicaciones en los extremos envíen y reciban mensajes por ella.



Sockets: Direcciones y Puertos

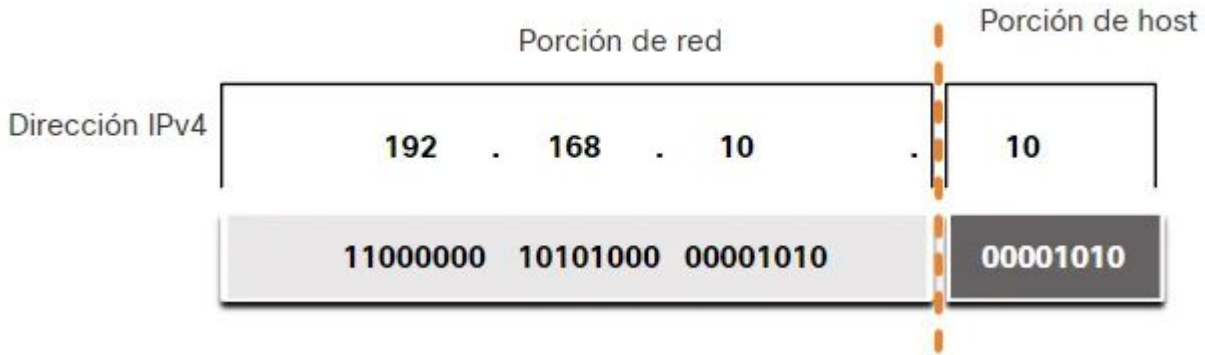
Cada host o equipo que está en una red TCP/IP tiene asignada una **dirección IP** única consistente en un número de red y un número de host.

- El número de red sirve para identificar la red en la que se encuentran los hosts.
- El número de host sirve para identificar a un host dentro de una red.

Sockets: Direcciones y Puertos

Las **direcciones Ipv4** son direcciones de 32-bits.

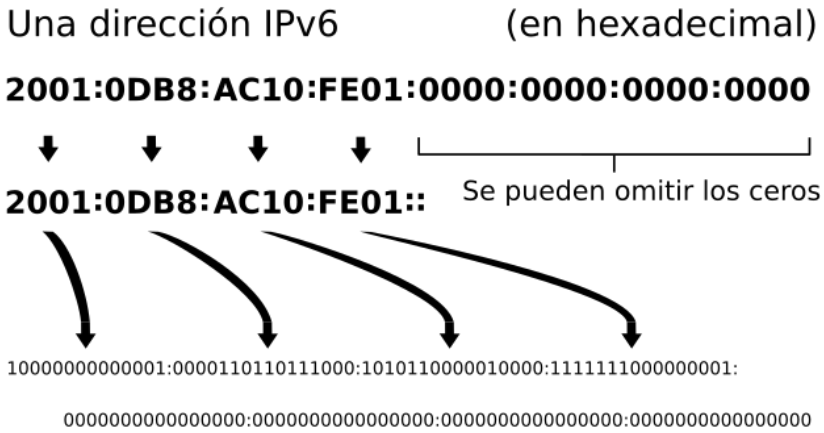
- La dirección IP se agrupa en cuatro octetos o bytes (grupos de 8 bits) y se representan usando el valor en notación decimal de cada uno de los bytes, separados por puntos.
- El valor mínimo para cada octeto es 0 y el valor máximo es 255.



Sockets: Direcciones y Puertos

Las **direcciones IPv6** están formadas por 64-bits para la dirección de red o prefijo de red, y otros 64 bits para el número de host.

- Se escriben como 8 grupos de 4 dígitos hexadecimales separados por el carácter ':'.
- Un grupo que sólo tiene ceros puede ser omitido.
- Los ceros iniciales también se pueden omitir.



Sockets: Direcciones y Puertos

Cuando una aplicación que se está ejecutando en un equipo quiere comunicarse con otra aplicación de otro equipo, se identifica a sí misma con un número de 16 bits, que denominamos **puerto**. Ese identificador es usado por los protocolos de la capa de transporte (TCP or UDP) para entregar los mensajes a la aplicación correcta dentro del equipo.

- Los puertos van de 0 a 65535, y se agrupan en tres rangos:

Grupo de puertos	Rango de puertos	Descripción
Puertos bien conocidos o puertos del sistema	0 - 1023	Los usan los protocolos estándar y los servicios del SO
Puertos registrados	1024- 49151	Reservados por empresas y organizaciones para sus propios servicios
Puertos efímeros	49152 - 65535	De libre disposición y uso para aplicaciones cliente y servidor

Sockets: Direcciones y Puertos

Los servidores de protocolos estándar como Telnet y FTP usan uno o más de estos puertos.

- La mayoría de los servidores sólo utilizan un puerto aunque hay otros, como FTP, que usan dos.
- El uso de un puerto específico permite a las aplicaciones cliente el poder comunicarse con el servidor sin tener que enviar una petición previa para determinar qué puerto se está usando.