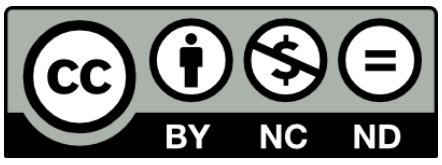
# Fundamentos de Redes


**Los Del DGIIM**, losdeldgiim.github.io

Doble Grado en Ingeniería Informática y Matemáticas
Universidad de Granada



# Fundamentos de Redes

Los Del DGIIM, losdeldgiim.github.io

José Juan Urrutia Milán

Granada, 2023-2024

# Índice general

# 1. Relaciones de Problemas

## 1.1. Introducción

**Ejercicio 1.1.1.** Explique brevemente las funciones de cada una de las capas del modelo de comunicación de datos OSI.

El modelo de comunicación de datos OSI cuenta con 7 niveles o capas:

1. Capa física: Se encarga de la parte física de la transmisión de los datos. Encontramos distintas formas de codificar los bits para su envío.

   Realiza funciones adicionales, como la codificación del canal.

2. Capa de enlace: Se encarga de los mecanismos de acceso al medio. En caso de haber un medio compartido por varios dispositivos, debe encargarse de no transmitir datos cuando otro medio lo está haciendo y de hacerlo cuando el canal se encuentre libre.

   En esta capa nos encontramos con los protocolos MAC y LLC.

3. Capa de red: Se encarga principalmente del direccionamiento de equipos (saber dar una dirección y tener un identificador dentro de la red) y del encaminamiento de datos (saber cómo mandar los paquetes al destinatario).

4. Capa de transporte: Se encarga principalmente de la fiabilidad de las comunicaciones:

   - Corrección de errores.
   - Manejar el congestionamiento de la red.
   - Controlar el flujo de datos (reducir velocidades si el receptor no es capaz de adecuar la velocidad de recibo a la de envío).
   - Realizar la multiplexación de los datos (ya que un mismo equipo puede tener varias aplicaciones que estén recibiendo datos a la vez).

5. Capa de sesión.

6. Capa de presentación[1].

7. Capa de aplicación: Se encarga de decidir qué datos envía a qué equipo, así como de interpretar los datos recibidos por otros equipos.

[1] No se han mencionado en clase las funcionalidades de las capas de presentación ni de sesión.

**Ejercicio 1.1.2.** Si la unidad de datos de protocolo en la capa de enlace se llama trama y la unidad de datos de protocolo en la capa de red se llama paquete, ¿son las tramas las que encapsulan los paquetes o son los paquete los que encapsulan las tramas? Explicar la respuesta.

Son las tramas las que encapsulan a los paquetes, ya que son las capas inferiores (en este caso, la de enlace) las que encapsulan la información de las capas superiores (en este caso, la de red) para su envío.

De esta forma, los paquetes son la PDU (*Protocol Data Unit*) de la capa de red, que se convierte en el SDU (*Servie Data Unit*) de la capa de enlace, la cual añade su cabecera al mismo convirtiéndolo en su PDU.

**Ejercicio 1.1.3.** Averigüe qué son los sistemas de representación de datos "*Little Endian*" y "*Big Endian*". ¿Puede un host que utilice representación *Little Endian* interpretar mensajes de datos numéricos provenientes de un host que utilice representación *Big Endian* y viceversa? Discuta la respuesta.

Sí que puede, para ello, debe haber un determinado protocolo que permita indicar qué codificación llevan los datos en binario. De esta forma, en alguna parte de la cabecera de los paquetes enviados, debe haber un bit que indique si los valores numéricos que se envían estén en *Big Endian* o en *Little Endian*.

**Ejercicio 1.1.4.** Cuando se intercambia un fichero entre dos hosts se pueden seguir dos estrategias de confirmación. En la primera, el fichero se divide en paquetes que se confirman individualmente por el receptor, pero el fichero en conjunto no se confirma. En la segunda, los paquetes individuales no se confirman individualmente, es el fichero entero el que se confirma cuando llega completo. Discutir las dos opciones.

Suponiendo que enviamos $n$ paquetes de datos, la primera forma envía de vuelta al emisor $n$ paquetes de confirmación, uno por cada paquete. De la segunda forma, el receptor espera a unir todos los paquetes en un fichero completo (y a verificar que no se ha perdido ningún paquete del mismo) para enviar el mensaje de verificación.

De la segunda forma se envían menos mensajes de verificación al emisor, por lo que la posibilidad de congestión de red por paquetes de verificación es menor. Sin embargo, en caso de que un paquete no consiga llegar o llegue en mal estado, no será hasta el final del envío de todos los paquetes que el receptor no genere el mensaje al emisor, por lo que en caso de errores en la comunicación, hay un mayor tiempo en la comunicación, al tener que esperar a que el receptor tenga todos los paquetes. Además, en el caso de error, el receptor no informará de qué paquete ha llegado mal, por lo que deberá pedir al emisor que reenvíe todos los paquetes de nuevo.

Resumiendo, ambas estrategias de confirmación tienen sus pros y sus contras. Dependiendo de la situación (si queremos mayor velocidad en la comunicación o si queremos menor saturación de red), puede interesarnos una u otra.

**Ejercicio 1.1.5.** ¿Para qué sirve el programa *ping*? ¿y el programa *traceroute*?

El programa *ping* usa el protocolo ICMP para enviar un paquete a un equipo, el cual tratará de responder con un paquete de confirmación de recepción del primer paquete. Sirve para comprobar la conexión y el buen funcionamiento de la red existente entre dos equipos. También sirve para calcular empíricamente la latencia de la conexión.

Por otra parte, el programa *traceroute* sirve para consultar todos los nodos intermedios por los que pasan los paquetes que salen de un emisor y llegan a un receptor, junto con la latencia de cada salto. Se usan varios paquetes ICMP con un valor creciente del campo TTL (*Time To Live*) para que cada salto intermedio devuelva un paquete de error ICMP con el valor del TTL que ha recibido. De esta forma, el emisor puede saber cuántos saltos intermedios hay entre él y el receptor, así como la latencia de cada uno de ellos.

**Ejercicio 1.1.6.** ¿Qué protocolos de un paquete puede cambiar un router? ¿En qué circunstancias?

Puede cambiar el TTL (*time to life*) del paquete[2].

**Ejercicio 1.1.7.** Averigue qué ISPs operan en España.

Algunos de los ISPs (*Internet Service Providers*) que operan en España son:

- Movistar.
- Vodafone.
- Orange.
- Jazztel.
- MásMóvil.
- Yoigo.
- Digi.

**Ejercicio 1.1.8.** ¿Qué es una aplicación cliente-servidor? ¿y una aplicación *peer-to-peer*?

Una aplicación cliente-servidor es una aplicación que depende de otra que probablemente esté en un equipo remoto (llamado servidor) para su funcionamiento.

Un ejemplo de aplicación cliente-servidor es una página web: tenemos aplicaciones que se ejecutan en local en cada equipo que accede a una determinada url. Dicha aplicación solicita datos a una aplicación que se encuentra en un equipo remoto, la cual proporciona datos (por ejemplo, accediendo a una base de datos) como respuesta a los datos solicitados por la aplicación que se ejecuta en cada equipo de forma local.

[2]Todavía no se ha mencionado en clase nada más acerca de esto.

Por otra parte, una aplicación *peer-to-peer*[3] es una aplicación que se distribuye entre varios equipos (que pueden estar muy lejanos entre sí) de forma que todas las aplicaciones tienen la misma relevancia en el buen funcionamiento del sistema.

**Ejercicio 1.1.9.** Describa brevemente la diferencia entre un *switch*, *router* y un *hub*.

Para responder a la pregunta, usamos además información que hemos aprendido en el Tema 2:

- Un switch es un nodo en una red que permite conectar tantos equipos como deseemos (normalmente, estos tienen 48 bocas de entrada RJ45 en el caso de conectar los equipos por ethernet) a una red. Funcionan a nivel de enlace, luego no tienen una dirección IP asociada.

- Un router es un nodo en una red que permite conectar redes distintas entre sí. Para ello, disponen de distintas tarjetas de red, cada una asociada a una red que se encuentra conectada al router. Disponen por tanto de varias direcciones IP, una por cada red a la que se conecta. Funciona a nivel de red.

  Presenta en su interior la tabla de enrutamientos, que permite el encaminamiento en la capa de red. Cuenta además con el NAT, que permite traducir direcciones de IP privadas a públicas y viceversa.

- Un *hub* es un nodo en una red que permite implementar la difusión. Se trata de un conjunto de bocas ethernet que internamente funcionan como un bus. Cada vez que un paquete se envía por una de las bocas, este es reenviado a todas las demás bocas, por lo que todos los equipos conectados al *hub* reciben el paquete.

**Ejercicio 1.1.10.** ¿Qué diferencia, en el contexto de una red de computadores, existe entre la tecnología de difusión y la tecnología punto-a-punto?

La tecnología de difusión permite enviar un paquete desde un equipoo y hacer que este sea recibido por el resto de equipos que estén conectados a la misma red (o hacer llegar estos a equipos en distintas redes). Es cada dispositivo el que decide si el paquete es para él o no.

Por otra parte, la teconología punto-a-punto permite el envío de paquetes desde un equipo a otro usando un medio directo, por lo que el destino está implícito desde que se envía el paquete. Esta tecnología es más rápida y segura, pero su escalabilidad es mucho menor.

**Ejercicio 1.1.11.** Un sistema tiene una jerarquía de protocolos de $n$ capas. Las aplicaciones generan mensajes de $M$ bytes de longitud. En cada capa se añade una cabecera de $h$ bytes. ¿Qué fracción del ancho de banda de la red se llena con cabeceras? Aplique el resultado a una conexión a 512 kbps con tamaño de datos de 1500 bytes y 4 capas, cada una de las cuales añade 64 bytes cabecera. ¿Qué velocidad real de envío de datos resulta?

[3]En español, podemos pensar en "entre pares".

Debemos sumar a los $M$ bytes iniciales que proporcionan las aplicaciones $n$ veces (la capa de aplicación también incluye una cabecera) $h$ bytes, por lo que la longitud de los mensajes que de verdad se envían es de $n \cdot h + M$ bytes. Por tanto, por cada $n \cdot h + M$ bytes enviados, $n \cdot h$ de ellos son de cabeceras:

$$\frac{n \cdot h}{n \cdot h + M} \cdot 100 \ \% \text{ de ancho de banda que se llena de cabeceras}$$

Si ahora partimos de 1500 bytes iniciales y añadimos 4 veces (una por capa) 64 bytes, estamos en realidad enviando mensajes de longitud:

$$4 \cdot 64 + 1500 = 256 + 1500 = 1756 \text{ bytes}$$

Por tanto, el ($^{256}/_{1756} = 0{,}145786$) 14.58 % de la red se emplea para enviar cabeceras, luego se aprovecha el ($1 - 0{,}145786 = 0{,}854214\,\%$) 85.42 % de la red para el envío de datos.

Si enviamos paquetes a una velocidad de 512kbps, en realidad estaremos enviando datos a una velocidad real de:

$$0{,}854214 \cdot 512 = 437{,}357568 \text{ kbps}$$

**Ejercicio 1.1.12.** Clasifique como de *difusión* o *punto a punto* cada uno de los siguientes sistemas de transmisión:

1. Radio y TV: Difusión, ya que es un emisor (en este caso, una cadena de televisión o radio) que difunde paquetes a cualquiera que tenga sintonizado dicho canal.

2. Redes inalámbricas (WLAN): Difusión, ya que cualquier equipo puede conectarse a la red y recibir los paquetes que se envían de forma inalámbrica.

3. ADSL: Punto a punto, ya que la conexión se establece mediante un cable. Usa en medio único.

4. Redes de cable: Puede implementar ambas tecnologías, ya que puede ser punto a punto (si cada equipo tiene su propio cable) o de difusión (si todos los equipos comparten el mismo cable).

5. Comunicaciones móviles (por ejemplo, GSM, UMTS, ...): Difusión, ya que (de nuevo) cualquier equipo puede recibir los paquetes que se envían por la red.

**Ejercicio 1.1.13.** Clasifique los siguientes servicios como orientados a conexión/no orientados a conexión y confirmados/sin confirmación. Justifique la respuesta.

Recordamos que los medios orientados a conexión son aquellos que comprueban si el receptor está disponible antes de enviar la información, y que los confirmados son aquellos que confirman la recepción del mensaje.

1. Correo postal ordinario: No es orientado a conexión (ya que no comprobamos anteriormente que el destinatario esté disponible, simplemente enviamos la carta) y es sin confirmación, ya que tras enviar la carta nada nos garantiza que el destinatario nos responda, pese a pedirlo explícitamente.

2. Correo certificado: No es orientado a conexión por la misma razón que el correo normal. Sin embargo, es confirmado, ya que al recibir el destinatario la carta debe firmar para que el emisor sea consciente de que la carta ha sido recibida.

3. Envío y recepción de fax.

4. Conversación telefónica. Es orientado a conexión, puesto que no se puede establecer una llamada si la otra persona no coge el teléfono. Además, es confirmado, ya que la otra persona ha de responder para que se produzca una comunicación.

5. Domiciliación bancaria de recibos.

6. Solicitud de certificado de empadronamiento. Es no orientado a conexión, ya que no se comprueba si el destinatario está disponible antes de enviar la solicitud. Además, es con confirmación, ya que el ayuntamiento envía un documento que certifica que el solicitante está empadronado.

**Ejercicio 1.1.14.** ¿Cuál es el tiempo necesario en enviar un paquete de 1000 Bytes, incluidos 50 Bytes de cabecera, por un enlace de 100 Mbps y 10Km? ¿cuál es el tiempo mínimo desde que se envía hasta que se recibe confirmación? ¿qué relación hay entre este tiempo y los temporizadores en, por ejemplo, las capas de enlace y transporte?

En primer lugar, hemos de calcular el retardo de transmisión $T_t$, que es el tiempo que se tarda en enviar el paquete por el enlace. Para ello, tenemos que:

$$T_t = 10^3\,\mathrm{B} \cdot \frac{8\,\mathrm{b}}{1\,\mathrm{B}} \cdot \frac{1\,\mathrm{s}}{100 \cdot 10^6\,\mathrm{b}} = 80 \cdot 10^{-6}\,\mathrm{s} = 80\,\mu\mathrm{s}$$

Por otra parte, el retardo de propagación $T_p$ es el tiempo que se tarda en enviar el paquete por los 10Km de cable. Para ello, suponiendo que la velocidad de transmisión es $^2/_3c = 2 \cdot 10^8\,\mathrm{m/s}$, tenemos que:

$$T_p = 10 \cdot 10^3\,\mathrm{m} \cdot \frac{1\,\mathrm{s}}{2 \cdot 10^8\,\mathrm{m/s}} = 50 \cdot 10^{-6}\,\mathrm{s} = 50\,\mu\mathrm{s}$$

Por tanto, el tiempo total que se tarda en enviar el paquete es de $T_t + T_p = 130\,\mu\mathrm{s}$.

Veamos ahora el tiempo mínimo desde que se envía hasta que se recibe confirmación. Además de los tiempos anteriores, hemos de tener en cuenta el tiempo de procesamiento del paquete, el retardo de transmisión del paquete de confirmación y el retardo de propagación del paquete de confirmación. Tenemos que:

- No se proporciona información del tiempo de procesamiento del paquete. No obstante, en los dispositivos modernos, este retardo es de varios órdenes de magnitud menor que los otros, por lo que se puede considerar despreciable.

- El retardo de propagación del paquete de confirmación es el mismo, puesto que la distancia recorrida es la misma.
- EL retardo de transmisión sí difiere, puesto que el tamaño del paquete de confirmación difiere. Normalmente, estos solo incluyen una cabecera, por lo que este tiempo, notado por $T_{\text{ACK}}$, es:

$$T_{\text{ACK}} = 50\,\text{B} \cdot \frac{8\,\text{b}}{1\,\text{B}} \cdot \frac{1\,\text{s}}{100 \cdot 10^6\,\text{b}} = 4 \cdot 10^{-6}\,\text{s} = 4\,\mu\text{s}$$

  Un temporizador de control de flujo en capa de enlace o de transporte debe ser suficientemente mayor a este tiempo mínimo para evitar un re-envío inmediato de paquetes ante cualquier eventualidad mínima en la red, como un retardo en las colas (mayor retardo de procesamiento) por un cierto nivel de congestión.

Por tanto, el tiempo total mínimo que se tarda en enviar el paquete y recibir confirmación es de:

$$T_{\text{total}} = T_t + T_p + T_{\text{ACK}} + T_p = 80\,\mu\text{s} + 50\,\mu\text{s} + 4\,\mu\text{s} + 50\,\mu\text{s} = 184\,\mu\text{s}$$