**Parte I: Conceptos y Teoría**

**Pregunta 1: Modelos OSI y TCP/IP**

El modelo OSI (Open Systems Interconnection) y el modelo TCP/IP tienen varias diferencias clave:

* **Número de capas**: El modelo OSI tiene 7 capas (Física, Enlace de datos, Red, Transporte, Sesión, Presentación y Aplicación), mientras que el modelo TCP/IP tiene solo 4 capas (Acceso a red, Internet, Transporte y Aplicación).
* **Orientación**: El modelo OSI es más teórico y se utiliza principalmente como herramienta educativa y de referencia. En cambio, el modelo TCP/IP es más práctico y se basa en protocolos que se utilizan realmente en Internet.
* **Desarrollo**: El modelo OSI fue desarrollado por la ISO (Organización Internacional de Normalización), mientras que TCP/IP fue creado por el Departamento de Defensa de los Estados Unidos.
* **Capa de Aplicación**: En el modelo OSI, las funciones de la capa de aplicación se dividen en tres capas separadas: Aplicación, Presentación y Sesión. En TCP/IP, estas funciones se agrupan en una única capa de Aplicación.
* **Adopción práctica**: TCP/IP se ha convertido en el modelo estándar en redes reales, mientras que OSI no ha sido adoptado en la práctica, aunque sigue siendo muy útil para entender cómo funcionan las redes.

1. Explica brevemente las ventajas y limitaciones de cada uno de estos modelos.

**Modelo OSI**  
*Ventajas:*

* Proporciona una estructura clara y detallada para entender cómo funcionan las redes.
* Separa bien las funciones entre capas, lo que facilita el diseño y la implementación.
* Favorece la interoperabilidad entre distintos fabricantes y protocolos.

*Limitaciones:*

* No fue adoptado ampliamente en la práctica.
* Su complejidad puede dificultar su implementación real.

**Modelo TCP/IP**  
*Ventajas:*

* Es el modelo sobre el que se basa Internet y se utiliza en redes reales.
* Más sencillo y enfocado en la funcionalidad práctica.
* Ha demostrado ser robusto y escalable con el tiempo.

*Limitaciones:*

* Menos detallado a nivel conceptual.
* La separación de funciones entre capas no está tan claramente definida como en OSI.

**Pregunta 2: Función de la Capa de Transporte**

La **capa de transporte** tiene un papel fundamental tanto en el modelo OSI como en el modelo TCP/IP, ya que se encarga de garantizar una comunicación fiable entre dispositivos de extremo a extremo, es decir, entre el emisor y el receptor.

En ambos modelos, esta capa tiene las siguientes funciones principales:

* **Segmentación y reensamblado de datos**: Divide los datos en segmentos para su envío y los reensambla correctamente en el destino.
* **Control de errores**: Verifica que los datos lleguen correctamente al destino y solicita retransmisión en caso de errores.
* **Control de flujo**: Regula la cantidad de datos que se envían para evitar saturar al receptor.
* **Multiplexación**: Permite que múltiples aplicaciones utilicen la red al mismo tiempo, asignando puertos a cada una.

En el **modelo OSI**, la capa de transporte es la **cuarta capa**, mientras que en el **modelo TCP/IP**, también se considera la **capa de transporte**, aunque ocupa una posición relativa diferente debido al menor número de capas del modelo.

Para **garantizar la entrega de datos**, esta capa puede utilizar protocolos orientados a la conexión. Por ejemplo:

* **TCP (Transmission Control Protocol)**: Proporciona una conexión fiable, asegura la entrega ordenada de los datos y realiza control de errores y retransmisiones. Es muy utilizado en aplicaciones como navegación web (HTTP/HTTPS), correo electrónico (SMTP) o transferencia de archivos (FTP).
* **UDP (User Datagram Protocol)**: Es un protocolo más ligero que no garantiza la entrega ni el orden de los datos. Se usa cuando la velocidad es más importante que la fiabilidad, como en videollamadas, streaming o juegos en línea.

En resumen, la capa de transporte asegura que los datos lleguen correctamente y en el orden adecuado, siendo esencial para el correcto funcionamiento de las aplicaciones de red.

**Pregunta 3: TCP vs. UDP**

**1. Orientación a conexión**

* **TCP** es un protocolo **orientado a la conexión**. Antes de enviar datos, establece una conexión entre el emisor y el receptor mediante un proceso llamado *three-way handshake*.
* **UDP** es **no orientado a la conexión**, es decir, no establece una conexión previa: simplemente envía los datos directamente al receptor.

**2. Fiabilidad y control de errores**

* **TCP** garantiza la **entrega fiable** de los datos, en el orden correcto. Detecta errores, solicita retransmisión en caso necesario y controla el flujo para evitar congestión.
* **UDP** **no garantiza** la entrega ni el orden de los datos. No realiza control de errores ni retransmisiones, por lo que es más sencillo y ligero.

**3. Velocidad y orden de entrega**

* **TCP** es más **lento** debido a los mecanismos de control y verificación que usa para garantizar la fiabilidad.
* **UDP** es más **rápido**, ya que no introduce retrasos por verificación o control de flujo, pero **no asegura el orden** de los paquetes.

**4. Ejemplos de aplicaciones**

**TCP se usa en aplicaciones que necesitan fiabilidad**, como:

* Navegación web (HTTP/HTTPS)
* Correo electrónico (SMTP, IMAP, POP3)
* Transferencia de archivos (FTP)

**UDP se emplea en aplicaciones donde la velocidad es prioritaria**, como:

* Videollamadas y VoIP (Skype, Zoom)
* Streaming de vídeo y audio en tiempo real
* Juegos en línea y transmisiones en directo

**Pregunta 4: Protocolo para Transferencia de Archivos**

**a)** El protocolo de la capa de aplicación utilizado tradicionalmente para la transferencia de archivos en redes TCP/IP es el **FTP (File Transfer Protocol)**.  
FTP permite la transferencia de archivos entre un cliente y un servidor a través de una red. Utiliza normalmente el puerto **21** y funciona sobre el protocolo **TCP** para garantizar una transmisión fiable.

**b)** Dos alternativas a FTP, con diferencias importantes en cuanto a **seguridad** y **funcionalidad**, son:

1. **SFTP (SSH File Transfer Protocol)**
   * **Seguridad**: Ofrece cifrado completo durante la transmisión, ya que se basa en el protocolo SSH (Secure Shell).
   * **Puerto utilizado**: Generalmente el **22**.
   * **Funcionalidad**: Similar a FTP, pero mucho más seguro, ideal para entornos donde se requiere confidencialidad y protección de datos.
2. **FTPS (FTP Secure o FTP-SSL)**
   * **Seguridad**: Es una extensión de FTP que añade cifrado mediante SSL/TLS.
   * **Puerto utilizado**: Puede usar el **990** para conexiones implícitas, o el **21** para conexiones explícitas con cifrado.
   * **Funcionalidad**: Mantiene la estructura y comandos de FTP, pero añade una capa de seguridad.

Otras alternativas modernas, aunque más orientadas al entorno web o a servicios en la nube, incluyen **HTTP/HTTPS** (para descargas) o herramientas como **scp** (en entornos Unix/Linux).

**Pregunta 5: Resolución de Nombres en DNS**

El sistema DNS (Domain Name System) se encarga de traducir nombres de dominio legibles por humanos (como www.ejemplo.com) a direcciones IP comprensibles por los dispositivos de red.

El proceso completo de resolución de nombres funciona así:

1. **Ingreso de la URL**:  
   El usuario escribe una dirección web (por ejemplo, www.ejemplo.com) en el navegador.
2. **Consulta a la caché local**:  
   El sistema operativo del dispositivo primero consulta su **caché DNS local** para ver si ya tiene la dirección IP correspondiente al dominio.
   * Si la tiene, la devuelve inmediatamente y se establece la conexión.
   * Si no, pasa al siguiente paso.
3. **Consulta al servidor DNS configurado (generalmente del proveedor de Internet)**:  
   El dispositivo envía una consulta al **servidor DNS recursivo** configurado en la red (por ejemplo, del ISP o de Google/Cloudflare).
4. **Revisión de la caché del servidor DNS recursivo**:  
   Este servidor también comprueba si tiene la IP almacenada en su caché.
   * Si la tiene, responde al cliente.
   * Si no, inicia una serie de consultas para resolver el nombre.
5. **Consulta a los servidores raíz**:  
   El servidor DNS recursivo pregunta a uno de los **13 servidores raíz** del DNS, que no conocen la IP final, pero sí pueden dirigir al servidor **TLD (Top-Level Domain)** correspondiente al dominio .com, .org, .es, etc.
6. **Consulta al servidor TLD**:  
   El servidor TLD proporciona la dirección del **servidor autoritativo** del dominio ejemplo.com.
7. **Consulta al servidor autoritativo**:  
   Finalmente, el servidor recursivo consulta al **servidor DNS autoritativo** de ejemplo.com, que responde con la **dirección IP exacta** de www.ejemplo.com.
8. **Respuesta al cliente y almacenamiento en caché**:  
   El servidor recursivo devuelve la IP al dispositivo del usuario. Esta información se guarda tanto en la **caché local** del dispositivo como en la del **servidor recursivo**, durante un tiempo determinado (TTL), para futuras consultas.
9. **Conexión con el servidor web**:  
   El navegador ya puede usar la IP para establecer una conexión con el servidor web mediante el protocolo HTTP o HTTPS y cargar la página solicitada.

**Pregunta 6: Comunicación en el Modelo TCP/IP**

El modelo TCP/IP describe cómo se comunican dos dispositivos en una red a través de **cuatro capas**. Cada capa tiene una función específica y colabora con las demás para asegurar que los datos lleguen correctamente de origen a destino.

A continuación, se explica el proceso de comunicación, capa por capa, desde el **dispositivo emisor** hasta el **receptor**:

**1. Capa de Aplicación**

* Es la capa más cercana al usuario.
* Contiene los programas y servicios que generan o utilizan los datos (por ejemplo, un navegador web, cliente de correo, etc.).
* Define cómo se comunican las aplicaciones entre dispositivos.
* **Ejemplo de protocolos**: HTTP, FTP, SMTP, DNS.

Cuando el usuario realiza una acción (como entrar a una página web), la capa de aplicación genera los datos que se desean enviar.

**2. Capa de Transporte**

* Asegura la comunicación fiable y ordenada entre aplicaciones de ambos dispositivos.
* Divide los datos en segmentos, añade información de control (como puertos de origen y destino) y puede garantizar la entrega mediante acuses de recibo y retransmisión si es necesario.
* **Ejemplo de protocolos**: TCP (fiable), UDP (más rápido pero sin garantía).

Esta capa prepara los datos para el transporte a través de la red.

**3. Capa de Internet**

* Encapsula los segmentos en **paquetes IP**, añadiendo direcciones IP de origen y destino.
* Se encarga del **enrutamiento** de los paquetes a través de la red, incluso si hay varias redes intermedias.
* **Protocolo principal**: IP (Internet Protocol).

Gracias a esta capa, los datos pueden viajar desde el dispositivo emisor hasta el receptor, aunque estén en redes distintas.

**4. Capa de Acceso a Red (o Enlace de Datos + Física)**

* Prepara los paquetes IP para su transmisión física.
* Añade información de la red local (como direcciones MAC), y encapsula los datos en **tramas**.
* Gestiona la transmisión real a través del medio físico (cable, WiFi, etc.).
* **Protocolos y tecnologías**: Ethernet, Wi-Fi, ARP, etc.

Finalmente, los datos se transmiten por el medio físico hacia el dispositivo de destino.

**Recepción del mensaje**

En el dispositivo receptor, el proceso se realiza **en orden inverso**:

1. La capa de acceso a red recibe las tramas y extrae los paquetes IP.
2. La capa de Internet comprueba la dirección IP y pasa los datos a la capa de transporte.
3. La capa de transporte reensambla los segmentos y los pasa a la aplicación correcta mediante el puerto.
4. La capa de aplicación interpreta los datos y los presenta al usuario (por ejemplo, mostrando una página web en el navegador).

Este proceso, conocido como **encapsulación y desencapsulación**, permite una comunicación clara y ordenada entre dispositivos, independientemente del tipo de red o sistema operativo que utilicen.

**Parte II: Capa Física y Ejercicios Prácticos**  
**Pregunta 7: Cálculo de Tasa de Transmisión Máxima (Fórmula de Shannon)**

Utiliza la fórmula de Shannon:

**C = B × log₂(1 + SNR)**

Donde:

* **C** es la tasa de transmisión máxima (en bps),
* **B** es el ancho de banda (en Hz),
* **SNR** es la relación señal a ruido en escala lineal.

Recuerda que para convertir el **SNR** de decibelios a escala lineal se usa la fórmula:

**SNR (lineal) = 10^(SNR (dB) / 10)**

**Enunciado:**  
Calcula la tasa de transmisión máxima para un canal con las siguientes características:

* Ancho de banda: **500 MHz**
* SNR: **20 dB**

Muestra el proceso de conversión del SNR a escala lineal y el cálculo final de **C**.

**Resolución paso a paso:**

1. **Conversión del ancho de banda a Hz:**

500 MHz = **500.000.000 Hz**

1. **Conversión del SNR de dB a escala lineal:**

SNR (lineal) = 10^(20 / 10) = 10² = **100**

1. **Aplicación de la fórmula de Shannon:**

C = 500.000.000 × log₂(1 + 100)  
C = 500.000.000 × log₂(101)  
C ≈ 500.000.000 × 6,6582  
C ≈ **3.329.100.000 bps**  
C ≈ **3,33 Gbps**

**Resultado:**  
La tasa de transmisión máxima es de aproximadamente **3,33 Gbps**.

**Pregunta 8: Ubicación de Portadoras para Eficiencia Espectral**

**a) Frecuencia de la portadora anterior:**

1,2 GHz – 300 MHz = **0,9 GHz** o **900 MHz**

**b) Frecuencia de la portadora posterior:**

1,2 GHz + 300 MHz = **1,5 GHz** o **1500 MHz**

**Justificación:**

Las portadoras deben estar separadas una distancia equivalente al ancho de banda del canal (en este caso, 300 MHz) para evitar interferencias entre señales adyacentes y asegurar una correcta transmisión.

**Importancia para la eficiencia espectral:**

Una ubicación precisa de las portadoras permite **optimizar el uso del espectro** disponible.  
Si están demasiado próximas, puede haber **interferencia** entre canales.  
Si están demasiado separadas, se **desaprovecha espacio** en el espectro.  
Por tanto, mantener una separación igual al ancho de banda de cada canal permite una **alta eficiencia espectral** y un uso más racional de las frecuencias disponibles.

**Pregunta 9: Identificación de Modulación en Función del BER**

**Orden de mayor a menor robustez ante el ruido:**

**1. BPSK**  
**2. QPSK**  
**3. 16-QAM**  
**4. 64-QAM**  
**5. 256-QAM**

**Justificación:**

La **robustez ante el ruido** en una modulación está inversamente relacionada con el **número de símbolos** o niveles posibles. Cuantos **más símbolos** tiene una modulación, **menor es la distancia entre ellos** en el plano de constelación, lo que la hace **más sensible al ruido**.

* **BPSK (2 símbolos)**: Tiene la mayor distancia entre símbolos, por lo que es la más resistente al ruido.
* **QPSK (4 símbolos)**: Duplica la eficiencia de BPSK pero introduce algo más de vulnerabilidad al ruido.
* **16-QAM, 64-QAM y 256-QAM**: Aumentan progresivamente la **eficiencia espectral**, pero al tener más símbolos en el mismo espacio, **la probabilidad de error (BER)** crece si la SNR no es muy alta.

En resumen, se sacrifica robustez a cambio de mayor capacidad de transmisión. Por eso, en entornos con mucho ruido o baja SNR, se prefiere usar modulaciones como BPSK o QPSK, mientras que 64-QAM o 256-QAM se usan solo cuando la calidad del canal es muy buena.

**Pregunta 10: Eficiencia del Sistema de Encapsulamiento**

**a) Tamaño del mensaje tras añadir cabeceras de las capas 4 y 3**

Datos originales:  
1,5 Kbytes = 1,5 × 1024 = **1536 bytes**

Cabecera de la capa 4: **40 bytes**  
Cabecera de la capa 3: **40 bytes**

**Tamaño total del mensaje:**  
1536 + 40 + 40 = **1616 bytes**

**b) Fragmentación en tramas de 400 bytes**

Cada trama puede transportar como **máximo 400 bytes**.

Para saber cuántas tramas se necesitan: 1616 ÷ 400 = 4,04 → Se necesitan **5 tramas** (ya que no se puede enviar parte de una trama, se redondea hacia arriba).

**c) Sobrecarga de la capa 1 por cada trama**

Capa 1 añade lo siguiente por **cada 2 bytes de datos**:

* 1 byte de inicio
* 1 byte de parada
* 1 byte de CRC

Es decir, por cada 2 bytes útiles → se añaden **3 bytes de sobrecarga**

**Razón total:**

* Por cada 2 bytes enviados → se transmiten **2 + 3 = 5 bytes**
* Por tanto, cada byte útil implica 2,5 bytes en total.

Ahora, calculamos la sobrecarga para **cada trama** de 400 bytes:

→ Cada trama contiene 400 bytes de datos.  
→ Número de bloques de 2 bytes: 400 / 2 = **200 bloques**  
→ Sobrecarga por trama: 200 × 3 = **600 bytes de sobrecarga**

**d) Eficiencia del sistema de encapsulamiento**

Primero calculamos el **total de datos transmitidos incluyendo sobrecarga**:

* Número de tramas: 5
* Datos + cabeceras por trama:  
  Total mensaje = 1616 bytes → dividido en 5 tramas (4 tramas completas y 1 con 16 bytes)

Distribución:

* 4 tramas de 400 bytes
* 1 trama de 16 bytes

**Tramas de 400 bytes:**

→ Cada una tiene 200 bloques de 2 bytes → 600 bytes de sobrecarga  
→ Total transmitido por trama = 400 (datos) + 600 (sobrecarga) = **1000 bytes**

**Trama de 16 bytes:**

→ 16 ÷ 2 = 8 bloques → 8 × 3 = 24 bytes de sobrecarga  
→ Total transmitido = 16 + 24 = **40 bytes**

**Total transmitido por el sistema:**

4 × 1000 = 4000

* 40 (última trama) = **4040 bytes transmitidos**

**Datos útiles originales:**  
= **1536 bytes**

**Eficiencia del sistema:**

Eficiencia = (Datos útiles / Total transmitido) × 100  
Eficiencia = (1536 / 4040) × 100 ≈ **38,03%**

**Resultado final:**

* **Tamaño total del mensaje (tras cabeceras):** 1616 bytes
* **Número de tramas necesarias:** 5
* **Sobrecarga por trama completa (400 bytes):** 600 bytes
* **Total de datos transmitidos:** 4040 bytes
* **Eficiencia del sistema:** ≈ **38,03%**