**Programación sobre redes**

**Trabajo Práctico teórico**

**Consignas  
  
1. ¿Qué es una red de computadoras y cuál es su propósito principal?**

Una red de computadoras es un conjunto de dispositivos conectados entre sí para compartir recursos, como archivos, impresoras, o acceso a internet. Su propósito principal es permitir la comunicación y la transferencia de datos entre dispositivos de manera rápida y eficiente, sin importar la distancia física entre ellos. Por ejemplo, una red puede conectar computadoras en una misma oficina o a nivel mundial a través de internet.

**2. Describe las diferencias entre una red local (LAN) y una red de área amplia (WAN).**

Las redes LAN (Local Area Network) son redes que cubren un área pequeña, como una casa, oficina, o un edificio. Son rápidas y tienen baja latencia, porque los dispositivos están físicamente cercanos entre sí. Por otro lado, las redes WAN (Wide Area Network) abarcan áreas mucho más grandes, como ciudades, países o incluso continentes. Estas redes conectan múltiples LANs y suelen tener mayores latencias debido a la distancia y la cantidad de dispositivos intermedios.

**3. Explica la función de una tarjeta de red (NIC) en una computadora.**

La tarjeta de red, o NIC (Network Interface Card), es un componente que permite que una computadora se conecte a una red. Esta tarjeta convierte los datos de la computadora en señales que pueden ser transmitidas a través de un medio físico, como cables Ethernet, o de manera inalámbrica. Sin una NIC, una computadora no podría comunicarse con otras en la red.

**4. Describe el modelo OSI y sus siete capas. ¿Cuál es el propósito de cada capa?**

El modelo OSI (Open Systems Interconnection) es un marco de referencia que divide el proceso de comunicación en redes en siete capas diferentes. Cada capa tiene una función específica y se comunica con la capa superior e inferior.

1. **Capa física**: Maneja la transmisión de bits a través de medios físicos (cables, ondas de radio).
2. **Capa de enlace de datos**: Asegura una transmisión de datos libre de errores entre dos dispositivos directamente conectados.
3. **Capa de red**: Se encarga del direccionamiento y enrutamiento de los datos en la red.
4. **Capa de transporte**: Proporciona transferencia confiable de datos entre dispositivos finales.
5. **Capa de sesión**: Administra las sesiones de comunicación entre aplicaciones.
6. **Capa de presentación**: Traduce datos entre el formato utilizado por la red y el formato usado por la aplicación.
7. **Capa de aplicación**: Interactúa con el software de la aplicación para proporcionar servicios de red.

**5. Explica cómo se comunican dos dispositivos utilizando el modelo OSI.**

Cuando dos dispositivos se comunican utilizando el modelo OSI, los datos pasan por cada una de las siete capas, comenzando en la capa de aplicación del dispositivo emisor y descendiendo hasta la capa física, donde se transmiten los bits por el medio físico. Al llegar al dispositivo receptor, los datos ascienden por las capas del modelo OSI, desde la capa física hasta la capa de aplicación, donde finalmente son procesados y entendidos por la aplicación.

**6. Describe el modelo TCP/IP y sus capas. ¿En qué se diferencia del modelo OSI?**

El modelo TCP/IP es un conjunto de protocolos de red que permite la comunicación entre dispositivos en internet. Está compuesto por cuatro capas:

1. **Capa de enlace**: Similar a las capas física y de enlace de datos del modelo OSI.
2. **Capa de red**: Equivalente a la capa de red del modelo OSI, maneja el direccionamiento y enrutamiento.
3. **Capa de transporte**: Similar a la capa de transporte del OSI, asegura una transferencia de datos confiable.
4. **Capa de aplicación**: Combina las capas de sesión, presentación y aplicación del modelo OSI.

La principal diferencia entre el modelo TCP/IP y el modelo OSI es que el primero es más simple y está basado en estándares prácticos usados en internet, mientras que el modelo OSI es más teórico y detallado.

**7. Explica la función de la capa de aplicación en el modelo TCP/IP. ¿Qué protocolos trabajan en esta capa?**

La capa de aplicación en el modelo TCP/IP es donde ocurren las interacciones más cercanas al usuario. Esta capa proporciona los servicios que permiten a las aplicaciones de software comunicarse a través de la red. Es responsable de protocolos que soportan actividades como la transferencia de archivos, el envío de correos electrónicos, y la navegación web.

Algunos de los protocolos que operan en esta capa incluyen:

* **HTTP/HTTPS**: Utilizados para la navegación web.
* **FTP**: Utilizado para la transferencia de archivos.
* **SMTP**: Utilizado para enviar correos electrónicos.
* **DNS**: Encargado de la resolución de nombres de dominio.

**8. Describe la capa de transporte en el modelo TCP/IP. ¿Cuáles son los dos principales protocolos en esta capa y en qué se diferencian?**

La capa de transporte en el modelo TCP/IP es responsable de la entrega confiable de datos entre dispositivos. Maneja la fragmentación y la reensamblaje de datos, asegurando que lleguen correctamente a su destino.

Los dos principales protocolos en esta capa son:

* **TCP (Transmission Control Protocol)**: Proporciona una entrega de datos confiable y en orden. Establece una conexión antes de transmitir datos y garantiza que cada paquete sea recibido correctamente. Si se detecta alguna pérdida, retransmite los datos.
* **UDP (User Datagram Protocol)**: Es menos formal que TCP y no garantiza la entrega o el orden de los datos. Es útil en situaciones donde la velocidad es más importante que la confiabilidad, como en streaming de video o juegos en línea.

**9. Explica la función de la capa de red en el modelo TCP/IP. ¿Cuál es el protocolo principal en esta capa?**

La capa de red en el modelo TCP/IP se encarga de la dirección y enrutamiento de los paquetes de datos a través de la red. Es responsable de decidir el camino que los datos deben tomar para llegar a su destino.

El protocolo principal en esta capa es el **IP (Internet Protocol)**, que se encarga de asignar direcciones a los dispositivos y de fragmentar y dirigir los paquetes de datos hacia la dirección correcta.

**10. Describe la función de la capa de enlace de datos en el modelo TCP/IP.**

La capa de enlace de datos en el modelo TCP/IP es la encargada de la comunicación directa entre dispositivos en la misma red física. Esta capa gestiona el acceso al medio de transmisión (como un cable Ethernet) y detecta y corrige errores que pueden ocurrir durante la transmisión de datos. También maneja las direcciones físicas, como las direcciones MAC.

**11. ¿Qué es una dirección IP y por qué es importante en la comunicación de redes?**

Una dirección IP (Internet Protocol) es un identificador único asignado a cada dispositivo que se conecta a una red que utiliza el protocolo IP. Esta dirección permite que los dispositivos puedan encontrarse y comunicarse entre sí en la red. Sin direcciones IP, no habría manera de enviar datos de un dispositivo a otro, ya que no habría forma de identificar el destino correcto.

**12. Explica la diferencia entre una dirección IP estática y una dinámica.**

Una dirección IP estática es una dirección que no cambia y se asigna de forma permanente a un dispositivo. Esto es útil para servidores o dispositivos que necesitan una dirección constante. Por otro lado, una dirección IP dinámica es asignada temporalmente a un dispositivo por un servidor DHCP (Dynamic Host Configuration Protocol) y puede cambiar cada vez que el dispositivo se conecta a la red.

**13. ¿Qué es una máscara de subred y cómo se utiliza en una red?**

Una máscara de subred es un número que divide una dirección IP en dos partes: la parte que identifica la red y la parte que identifica los dispositivos individuales dentro de esa red. Esto permite que grandes redes IP se dividan en subredes más pequeñas, mejorando la eficiencia y seguridad del tráfico de red.

**14. Explica el proceso de encapsulación de datos desde la capa de aplicación hasta la capa física en el modelo OSI.**

La encapsulación de datos es el proceso mediante el cual la información pasa por las capas del modelo OSI y se le añade la información necesaria para que los datos puedan ser transmitidos y comprendidos en la red. Aquí te lo explico capa por capa:

* **Capa de aplicación**: Los datos son generados por la aplicación, como un correo electrónico o una solicitud de web.
* **Capa de presentación**: Los datos se convierten a un formato estándar y, si es necesario, se comprimen o encriptan.
* **Capa de sesión**: Se establecen y gestionan las sesiones de comunicación.
* **Capa de transporte**: Los datos se dividen en segmentos, y se añade información para la entrega confiable.
* **Capa de red**: Los segmentos se convierten en paquetes y se les asigna una dirección IP para el enrutamiento.
* **Capa de enlace de datos**: Los paquetes se encapsulan en tramas, añadiendo direcciones MAC y control de errores.
* **Capa física**: Finalmente, los datos se convierten en señales eléctricas o de radio y se transmiten a través del medio físico.

**15. ¿Qué es la concurrencia en programación y por qué es importante en el contexto de redes?**

La concurrencia en programación se refiere a la capacidad de ejecutar múltiples tareas al mismo tiempo. En el contexto de redes, es crucial porque permite que un sistema maneje múltiples conexiones y procesos simultáneamente. Por ejemplo, un servidor web necesita manejar múltiples solicitudes de clientes al mismo tiempo, y la concurrencia permite que esto sea posible sin que el sistema se ralentice.

**16. Explica qué es un thread en programación y cómo se utiliza para manejar la concurrencia.**

Un thread es la unidad más pequeña de procesamiento que puede ser ejecutada de manera concurrente dentro de un programa. Los threads permiten que diferentes partes de un programa se ejecuten al mismo tiempo, lo cual es fundamental para manejar la concurrencia. Por ejemplo, en una aplicación de red, un thread puede manejar la recepción de datos mientras otro thread procesa esos datos, mejorando así la eficiencia y la velocidad del programa.

**17. ¿Qué es la sincronización de threads y por qué es importante?**

La sincronización de threads es el proceso de coordinar la ejecución de threads para asegurar que no interfieran entre sí, especialmente cuando comparten recursos. Es crucial porque, sin sincronización, los threads podrían intentar modificar el mismo recurso al mismo tiempo, causando errores o resultados inesperados.

**18. Explica las diferencias entre procesos y threads.**

Un proceso es un programa en ejecución que tiene su propio espacio de memoria y recursos asignados por el sistema operativo. Un thread, en cambio, es una unidad de ejecución dentro de un proceso y comparte el mismo espacio de memoria y recursos que otros threads del mismo proceso. La principal diferencia es que los threads son más ligeros y rápidos de crear y manejar que los procesos, ya que comparten el mismo espacio de memoria.

**19. Describe una situación en la que utilizarías threads para mejorar el rendimiento de una aplicación en red.**

Imagina que estás desarrollando un servidor web que debe manejar múltiples solicitudes de usuarios al mismo tiempo. Si utilizas un solo proceso para manejar todas las solicitudes, el servidor se volvería lento, ya que tendría que procesar cada solicitud de manera secuencial. Sin embargo, si utilizas threads, cada solicitud podría ser manejada por un thread diferente, permitiendo que el servidor procese múltiples solicitudes de manera simultánea y mucho más rápida.

**20. Investiga y describe brevemente un protocolo de red moderno (como HTTP/2, WebSockets, gRPC) y su importancia en la programación en red.**

Uno de los protocolos modernos más importantes es **HTTP/2**, que es una evolución del protocolo HTTP/1.1 utilizado para la navegación web. HTTP/2 mejora la eficiencia del envío de datos mediante la multiplexación, lo que permite que múltiples solicitudes y respuestas se transmitan a través de una única conexión TCP. Esto reduce la latencia y acelera la carga de las páginas web, lo cual es crucial para mejorar la experiencia del usuario en la web moderna. Además, HTTP/2 también utiliza compresión de cabeceras, lo que reduce la cantidad de datos que necesitan ser transmitidos.