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 que están interconectados, como por ejemplo computadoras, servidores, impresoras, y otros dispositivos electrónicos, que están conectados entre sí para intercambiar datos. Dichas conexiones pueden ser físicas(cables) o inalámbricas(WIFI o Bluetooth). Su propósito principal es compartir recursos e información entre los dispositivos, como por ejemplo archivos, acceso a las impresoras, aplicaciones y otros dispositivos.

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

**(WAN).**

Una red local LAN (Local Area Network), es una red de computadoras que se utilizan para conectar dispositivos en un área limitada, como por ejemplo en una oficina, en un edificio, etc. Mientras que una red WAN (Wide Area Network) es una red que conecta redes LAN y cubre un área más grande que las redes LAN, haciéndolas más compleja su administración, ya que pueden cubrir una ciudad, un país, y hasta continentes.

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

La tarjeta de red (NIC), es un componente del hardware muy importante, ya que permite que una computadora tenga conexión con una red, ya sea cableada o inalámbrica. Su función principal es actuar como un intermediario entre la computadora y la red.

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

**capa?**

El modelo OSI es un marco conceptual que estandariza las funciones de una red de telecomunicaciones o un sistema de computadoras, y dicho modelo cuenta con 7 capas:

* Capa 1: Capa Física: Se encarga de la transmisión física de datos a través de un medio de comunicación.
* Capa 2: Capa de Enlace de Datos: Proporciona un enlace fiable entre dos nodos en una red y se encarga de la detección y corrección de errores en la capa física. Maneja el direccionamiento de físico (direcciones MAC).
* Capa 3: Capa de Red: Se encarga del direccionamiento lógico (IP) y la determinación de la ruta para el envío de datos a través de diferentes redes, maneja la fragmentación y reensamblaje de paquetes.

* Capa 4: Capa de Transporte: Garantiza la transferencia de datos completa y confiable entre sistemas finales en la red. Ofrece control de flujo, control de errores y reenvío de datos si es necesario.

* Capa 5: Capa de Sesión: Administra y controla las conexiones entre computadoras, estableciendo, gestionando y terminando sesiones entre aplicaciones.

* Capa 6: Capa de Presentación: Se encarga de la traducción de datos entre la red y las aplicaciones. Actúa como traductor asegurando que los datos transmitidos por la red sean entendidos por la aplicación receptora.

* Capa 7: Capa de Aplicación: Es la capa más cercana al usuario final y proporciona servicios (Protocolos) de red directamente a las aplicaciones de software. Esta capa facilita la interacción del usuario con la red a través de aplicaciones como navegadores web (https), clientes de correo electrónico (SMTP).

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

Por ejemplo si quieres enviar un correo a una persona, el proceso por el emisor empieza desde la capa 7 hasta la capa 1:

Capa 7: Primero utilizamos un cliente de correo electrónico (como gmail) para redactar un mensaje de correo. Cuando se hace clic en “Enviar”, el cliente de correo utiliza el protocolo SMTP para crear un mensaje que incluye la dirección del remitente, la dirección del destinatario, el asunto y el cuerpo del mensaje.

Capa 6: Luego la capa 6 convierte el mensaje de correo en un formato que pueda ser transmitido a través de la red.

Capa 5: En la capa 5 se establece una sesión entre tu y el servidor de correo electrónico, para permitir la transferencia del correo electrónico que quieres enviar.

Capa 4 Transporte: El mensaje de correo es dividido en segmentos, y se le añade una cabecera que incluye la información del puerto de origen y destino.

Capa 3 de red: Cada segmento se encapsula en un paquete IP. La cabecera del paquete IP incluye la dirección IP de tu computadora como origen y la dirección IP del servidor de correo como destino.

Capa 2 Enlace de Datos: Los paquetes IP se encapsulan en tramas que contienen las direcciones MAC de tu computadora y del router o switch más cercano.

Capa 1 Física: Y por último las tramas se convierten en señales eléctricas, ópticas o inalámbricas que se transmiten a través del medio físico, desde tu computadora hasta el router, y de ahí hace muchos saltos hasta llegar al servidor de correo.

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

**OSI?**

El modelo TCP/IP (Transmission Control Protocol/Internet Protocol) es un conjunto de protocolos que permiten la comunicación en una red. Está compuesto por cuatro capas:

* Capa de Aplicación: Combina las funciones de las capas de Aplicación, Presentación y Sesión del modelo OSI. Aquí operan protocolos como HTTP, FTP, SMTP, etc.
* Capa de Transporte: Equivalente a la capa de Transporte del modelo OSI. Gestiona la comunicación de extremo a extremo, utilizando principalmente los protocolos TCP y UDP.
* Capa de Internet: Es igual a la capa de Red del modelo OSI. Maneja el direccionamiento y enrutamiento de los paquetes mediante el protocolo IP.
* Capa de Enlace de Datos (o Acceso a la Red): Combina las funciones de las capas Física y de Enlace de Datos del modelo OSI. Se encarga de la transmisión de los datos en el medio físico.

Las diferencias del modelo TCP/IP y el modelo OSI:

El modelo OSI tiene siete capas, mientras que el modelo TCP/IP tiene cuatro, además el modelo TCP/IP es más práctico y está basado en protocolos estándar que realmente se usan redes, mientras que el modelo OSI es más teórico y conceptual. También en TCP/IP, algunas capas del modelo OSI están combinadas.

**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 responsable de proporcionar servicios de red directamente a las aplicaciones del usuario. Esto incluye la entrega de datos entre aplicaciones a través de la red. Es la capa donde las aplicaciones acceden a la red y donde se encuentran los protocolos que permiten la comunicación entre ellas.

Los protocolos que trabajan en esta capa son:

* **HTTP/HTTPS**: Para la transferencia de páginas web.
* **FTP**: Para la transferencia de archivos.
* **SMTP**: Para el envío de correos electrónicos.
* **DNS**: Para la resolución de nombres de dominio.
* **Telnet/SSH**: Para la administración remota de dispositivos.

**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 se encarga de la comunicación de extremo a extremo entre dos dispositivos en una red. Garantiza que los datos se entreguen de manera fiable y ordenada, o que se entreguen rápidamente, según el protocolo que utilice.

Los principales protocolos en esta capa son:

* **TCP** (Transmission Control Protocol): Proporciona una comunicación confiable y orientada a la conexión. Se asegura de que los datos lleguen completos y en el orden correcto. Es ideal para aplicaciones donde la precisión de los datos es crucial, como en la web o el correo electrónico.
* **UDP** (User Datagram Protocol): Ofrece una comunicación sin conexión y no garantiza la entrega ni el orden de los datos. Es más rápido que TCP y se usa en aplicaciones donde la velocidad es más importante que la precisión, 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 enrutar los paquetes de datos desde el origen hasta el destino a través de múltiples redes interconectadas. Es responsable de la selección de rutas y del direccionamiento de los paquetes.

Protocolo principal:

IP (Internet Protocol): Es el protocolo principal en esta capa. Define las direcciones IP para identificar dispositivos en la red y determina la ruta que deben seguir los paquetes para llegar a su destino.

**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 se encarga de la transmisión de datos entre dos dispositivos conectados físicamente en la misma red. Esta capa maneja la comunicación a nivel de hardware, como el control de acceso al medio, la detección y corrección de errores, y el direccionamiento físico (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 en una red que utiliza el protocolo IP. Esta dirección permite que los dispositivos se identifiquen y se comuniquen entre sí en la red, facilitando el enrutamiento y la entrega de paquetes de datos desde un origen hasta un destino. Esta dirección IP es muy importante, ya que sin una dirección IP, un dispositivo no podría enviar ni recibir información dentro de la red.

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

La diferencia es que una IP Estática es una dirección IP que no cambia. Se asigna manualmente a un dispositivo y permanece constante. Mientras que una IP Dinámica es una dirección IP que cambia cada vez que un dispositivo se conecta a la red. Se asigna automáticamente por un servidor DHCP (Dynamic Host Configuration Protocol). Es común para la mayoría de los dispositivos de usuario, como computadoras personales y teléfonos móviles.

**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 a los dispositivos (hosts) dentro de esa red. La máscara de subred permite que un dispositivo en la red determine si otro dispositivo está en la misma subred o en una red diferente, lo que facilita el enrutamiento de paquetes.

**14.Explica el proceso de encapsulación de datos desde la capa de aplicación**

**hasta la capa física en el modelo OSI.**

7) **Capa de Aplicación:** Los datos se generan y se pasan a la capa de presentación.

6) **Capa de Presentación:** Se realiza la traducción de los datos si es necesario y se agregan cabeceras específicas.

5) **Capa de Sesión:** Establece, gestiona y termina sesiones. Se añade información de control para la gestión de las sesiones.

4) **Capa de Transporte:** Se añade una cabecera que incluye el número de puerto de origen y destino, así como la información de control de flujo y corrección de errores (si se usa TCP).

3) **Capa de Red:** Se añade una cabecera con la dirección IP de origen y destino, para el enrutamiento.

2) **Capa de Enlace de Datos:** Se encapsulan los datos en tramas, añadiendo una cabecera y un tráiler que incluyen las direcciones MAC y la verificación de errores.

1) **Capa Física:** Los bits se transmiten a través del medio físico como señales eléctricas, ópticas o inalámbricas.

**15.¿Qué es la concurrencia en programación y por qué es importante en el**

**contexto de redes?**

La concurrencia en programación es la capacidad de un sistema para realizar múltiples tareas de forma simultánea o en paralelo. Es importante porque permite que un sistema maneje múltiples conexiones o solicitudes al mismo tiempo, mejorando la eficiencia y el rendimiento de las aplicaciones de red.

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

**concurrencia.**

Un thread (o hilo) es la unidad más pequeña de procesamiento que puede ejecutar código de manera independiente dentro de un proceso. En otras palabras, un proceso puede contener múltiples threads, y cada uno puede ejecutar tareas simultáneamente dentro del mismo espacio de memoria compartido.

Un ejemplo de cómo se utilizan los threads en una aplicación de servidor web, un thread puede manejar la solicitud de un cliente mientras otro thread procesa una solicitud diferente al mismo tiempo.

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

La sincronización de threads es el mecanismo que asegura que dos o más threads puedan acceder y modificar recursos compartidos de manera ordenada y sin conflictos. La sincronización es importante porque de lo contrario, los threads podrían acceder al mismo recurso, como por ejemplo una variable, al mismo tiempo, entonces generaría conflictos al depender del orden en que los threads acceden a los recursos, generando corrupción de datos o errores al momento de ejecutar un programa.

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

El proceso es una instancia en ejecución de un programa. Un proceso tiene su propio espacio de memoria, recursos y estado independiente, pero la comunicación entre procesos suele ser más compleja y lenta. Mientras que, un threads es la unidad de ejecución dentro de un proceso, y su comunicación entre threads es más sencilla que el de los procesos, porque pueden compartir datos directamente en la memoria. Los threads dentro de un mismo proceso comparten el mismo espacio de memoria y recursos.

**19.Describe una situación en la que utilizarías threads para mejorar el**

**rendimiento de una aplicación en red.**

Yo lo usaría en una aplicación de descargas de películas desde internet. Porque si la aplicación sólo utiliza un thread, tendría que esperar a que una película se termine de descargar, así podrá descargar otra. Pero si se utilizan muchos threads, podrías crear un thread para cada película que se quiera descargar, permitiendo descargar varias películas a la vez.

**20.Investiga y describe brevemente un protocolo de red moderno (como HTTP/2,**

**WebSockets, gRPC) y su importancia en la programación en red.**

**gRPC** es un protocolo de comunicación de alto rendimiento desarrollado por Google. Se basa en HTTP/2 y utiliza Protocol Buffers (protobuf) para la serialización de datos, lo que permite una comunicación rápida y eficiente entre servicios en diferentes lenguajes de programación. Es importante para la programación porque es compatible con muchos lenguajes de programación, lo que lo hace ideal para sistemas distribuidos y microservicios que pueden estar escritos en diferentes lenguajes. También tiene un soporte integrado para autenticación mediante TLS/SSL, lo que lo hace seguro y adecuado para aplicaciones sensibles.

Nombre: Axel Chino