**Que es un DNS**

Un DNS (Domain Name System) es un sistema de nomenclatura jerárquica utilizado para traducir nombres de dominio (como ejemplo.com) en direcciones IP (como 192.0.2.1) y viceversa.

En lugar de tener que recordar y escribir direcciones IP numéricas complicadas para acceder a un sitio web o servicio en línea, el DNS permite a los usuarios ingresar un nombre de dominio fácil de recordar en su navegador web. El sistema DNS luego traduce ese nombre de dominio en la dirección IP correspondiente que se necesita para acceder al servidor que aloja el sitio web o servicio.

El DNS también permite la configuración de otros registros DNS, como registros de correo electrónico (para especificar los servidores de correo electrónico para un dominio) y registros de servicio (para especificar los servicios que se ofrecen en un dominio).

**Que es una URL.**

Una URL (Uniform Resource Locator) es una cadena de texto que se utiliza para identificar y localizar un recurso en Internet, como una página web, una imagen o un archivo. Una URL se compone de varios componentes, que incluyen:

* Protocolo: el protocolo de comunicación utilizado para acceder al recurso (por ejemplo, HTTP, HTTPS, FTP, etc.)
* Dominio: el nombre de dominio del servidor que aloja el recurso (por ejemplo, google.com)
* Ruta: la ruta de acceso al recurso en el servidor (por ejemplo, /search?q=example)
* Parámetros: información adicional que se pasa al servidor junto con la solicitud (por ejemplo, para realizar una búsqueda específica)

En conjunto, estos componentes permiten a un navegador web o a otro cliente de Internet acceder a un recurso específico en la web. Por ejemplo, la URL "https://www.google.com/search?q=example" se utiliza para acceder a la página de resultados de búsqueda de Google para la consulta "example".

**Que es un Socket**

Un socket es un punto final en una conexión de red bidireccional que se utiliza para la comunicación entre procesos o aplicaciones en una red. Es un objeto que representa una conexión de red en un sistema operativo y proporciona una interfaz para la comunicación entre procesos que se ejecutan en diferentes hosts.

Un socket se identifica mediante una dirección IP y un número de puerto. La dirección IP identifica el host en la red y el número de puerto identifica el proceso en el host. La combinación de la dirección IP y el número de puerto se utiliza para establecer una conexión de red entre dos hosts.

Los sockets se utilizan en una amplia variedad de aplicaciones, como navegadores web, clientes de correo electrónico, servidores web, servicios de mensajería instantánea y transferencia de archivos. Los diferentes tipos de sockets incluyen sockets de flujo, sockets de datagrama y sockets raw.

**Capa 1: Capa física**

* Protocolos: Ethernet, Wi-Fi, Bluetooth, USB
* Función: Establecer y mantener la conexión física entre los dispositivos de red.

**Capa 2: Capa de enlace de datos**

* Protocolos: PPP, HDLC, Ethernet, Wi-Fi
* Función: Proporcionar un medio confiable para transferir datos entre dispositivos de red adyacentes.

**Capa 3: Capa de red**

* Protocolos: IP, ICMP, ARP, OSPF
* Función: Proporcionar direccionamiento lógico para la transferencia de datos entre diferentes redes.
* **IP** (Internet Protocol): es el protocolo más importante de la capa de red, ya que se encarga de enrutar los paquetes de datos entre los diferentes nodos de la red. IP proporciona una dirección lógica única a cada dispositivo en la red, conocida como dirección IP, que se utiliza para identificar y enrutar los paquetes a su destino. Además, IP utiliza el enrutamiento dinámico para ajustar las rutas de la red en tiempo real, lo que permite una mayor eficiencia y flexibilidad en la transmisión de datos.
* **ICMP** (Internet Control Message Protocol): es un protocolo utilizado para enviar mensajes de control y error en la red. Por ejemplo, cuando un paquete no llega a su destino, ICMP puede enviar un mensaje de error al dispositivo emisor para informarle del problema. ICMP también se utiliza para realizar pruebas de conectividad en la red, como el comando "ping", que envía un paquete ICMP y espera una respuesta del dispositivo de destino.
* **ARP** (Address Resolution Protocol): es un protocolo utilizado para obtener la dirección física (MAC) de un dispositivo a partir de su dirección lógica (IP). Cuando un dispositivo desea comunicarse con otro dispositivo en la red, utiliza ARP para obtener su dirección MAC, que es necesaria para la transmisión de datos en la red.
* **OSPF** (Open Shortest Path First): es un protocolo de enrutamiento que se utiliza para calcular la ruta más eficiente para enrutar los paquetes en una red. OSPF utiliza un algoritmo de enrutamiento de estado de enlace para calcular las rutas y actualizarlas dinámicamente en tiempo real. OSPF es especialmente útil en redes grandes y complejas, donde se requiere una actualización constante de las rutas.

La Capa de Red es la tercera capa del modelo OSI (Open Systems Interconnection) y del modelo TCP/IP (Transmission Control Protocol/Internet Protocol). Su principal función es la de proporcionar una comunicación lógica entre los dispositivos en una red, independientemente de su ubicación física en la misma.

La Capa de Red se encarga de enrutar los paquetes de datos entre los distintos nodos de la red, utilizando direcciones lógicas (como la dirección IP) para identificar los dispositivos y determinar la ruta óptima para que los paquetes de datos lleguen a su destino. También se encarga de controlar la congestión en la red y de proporcionar un servicio de calidad de servicio (QoS) que permita priorizar determinados tipos de tráfico sobre otros.

En resumen, la Capa de Red es esencial para la comunicación entre dispositivos en una red, ya que permite que los paquetes de datos sean enviados y recibidos de forma eficiente y confiable.

**Capa 4: Capa de transporte**

* Protocolos: TCP, UDP,SCTP
* Función: Proporcionar la entrega confiable y la gestión de flujo de datos entre procesos de aplicaciones en diferentes hosts.
* **TCP** (Transmission Control Protocol): es un protocolo orientado a conexión, lo que significa que establece una conexión antes de transferir datos. TCP garantiza la entrega confiable de los datos, asegurándose de que lleguen al destino en el orden correcto, sin errores y sin duplicados. También proporciona control de flujo, evitando que los dispositivos emisores envíen más datos de los que los dispositivos receptores pueden manejar.
* **UDP** (User Datagram Protocol): es un protocolo sin conexión que no establece una conexión antes de enviar los datos. Es más rápido y eficiente que TCP, pero no garantiza la entrega confiable de los datos, ya que no proporciona mecanismos para controlar errores ni para retransmitir los paquetes perdidos. UDP es comúnmente utilizado en aplicaciones que requieren una transmisión rápida y eficiente de datos, como videojuegos en línea y transmisiones de audio y video en tiempo real.
* **SCTP** (Stream Control Transmission Protocol): es un protocolo orientado a conexión que se diseñó para mejorar la fiabilidad y seguridad de las comunicaciones. Es similar a TCP en términos de garantizar la entrega confiable de los datos y control de flujo, pero también ofrece otras características, como la capacidad de soportar múltiples flujos de datos en una misma conexión, lo que mejora el rendimiento en redes con alta latencia y pérdida de paquetes. SCTP se utiliza en aplicaciones que requieren un alto nivel de confiabilidad y seguridad, como la telefonía IP y las transacciones financieras en línea.

La Capa de Transporte es la cuarta capa del modelo OSI (Open Systems Interconnection) y del modelo TCP/IP (Transmission Control Protocol/Internet Protocol). Su principal función es la de proporcionar servicios de comunicación de extremo a extremo entre los dispositivos finales de una red, es decir, los dispositivos que generan y reciben los datos.

La Capa de Transporte se encarga de dividir los datos que recibe de la Capa de Aplicación en segmentos más pequeños, que pueden ser transmitidos a través de la red de manera más eficiente. Además, esta capa proporciona un mecanismo para el control de flujo y de errores, garantizando que los datos lleguen al destino final de manera confiable.

Los dos protocolos más comunes de la Capa de Transporte son el Protocolo de Control de Transmisión (TCP) y el Protocolo de Datagrama de Usuario (UDP). TCP proporciona una comunicación orientada a conexión, lo que significa que se establece una conexión entre los dispositivos antes de que se transmitan los datos. TCP también garantiza que los datos lleguen al destino final en el orden correcto y sin errores. UDP, por otro lado, es un protocolo sin conexión que se utiliza para la transmisión de datos en tiempo real, como en las transmisiones de video y voz.

En resumen, la Capa de Transporte es esencial para garantizar una comunicación confiable y eficiente entre los dispositivos finales de una red, proporcionando mecanismos de control de flujo y errores, así como la división de los datos en segmentos más pequeños para su transmisión a través de la red.

**Capa 5: Capa de sesión**

* Protocolos: SSL, TLS
* Función: Establecer y mantener conexiones de sesión seguras y confiables entre procesos de aplicaciones en diferentes hosts.

**Capa 6: Capa de presentación**

* Protocolos: MIME, SSL, TLS
* Función: Proporcionar la representación y el formato de datos adecuados para su transferencia entre aplicaciones.

**Capa 7: Capa de aplicación**

* Protocolos: HTTP, FTP, SMTP, DNS
* Función: Proporcionar servicios de red específicos de la aplicación para procesos de aplicaciones en diferentes hosts.

<https://github.com/Raro1280/algorito-para-crear-chat-en-java.git>

Conclusiones de la practica:

* Para generar un chat es necesario tener una conexión inicial el cual está orientado identificado ServerSocket es que lo que nos dejar pasar la información
* Se identifica como Ip el registro único de cada computadora para establecer la conexión
* Es importante tener claro nuestro proyecto he identificar las posibles librerías que nos permiten realizar más fácil nuestro trabaja.