Lab1 - Red Humana e Intro a Wireshark

Silvia Illescas 22376 – Michelle Mejía 22596
Otra Pareja: Ruth de León – Isabella Mirayes

Descripción de la práctica

Durante la práctica se exploraron distintas formas de comunicación codificada como Morse y Baudot, comparando su velocidad y precisión. También se enviaron mensajes por notas de voz, enfrentando retos como pausas ambiguas. Luego se simuló un sistema de conmutación, donde un participante actuó como intermediario, reflexionando sobre la gestión del tráfico en redes.

Finalmente, se utilizó Wireshark para analizar el tráfico de red en tiempo real, identificando protocolos, tamaño de datos y configuración del navegador. También se discutió la importancia de no usar Wireshark en servidores en producción por razones de seguridad y rendimiento.

1.1 Primera parte: transmisión de códigos

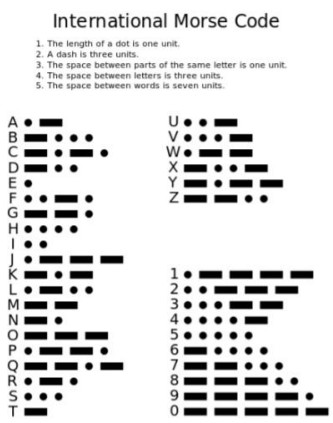


Imagen 1: *Código Morse internacional. Fuente*

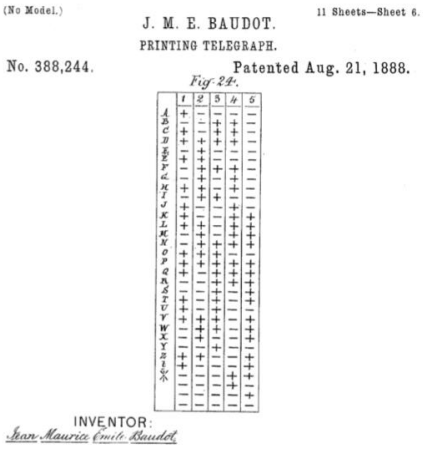


Imagen 2: *Código de Baudot. Fuente*

¿Qué esquema es más fácil? ¿Más difícil?

Consideramos que el esquema más fácil de utilizar es el del código Morse, ya que permite transmitir el mensaje de forma más rápida y su descifrado es más sencillo. Esto se debe a que las letras y los números están representados de una manera más simple y directa.

¿Con cuál ocurren menos errores?

Por otro lado, el código Baudot nos parece más completo y estructurado. Su forma de transmisión e interpretación resulta más lógica, lo que reduce la probabilidad de errores al momento de enviar o recibir el mensaje correctamente.

1.2 Segunda parte: transmisión “empaquetada”

¿Qué dificultades involucra el enviar un mensaje de esta forma "empaquetada"?

Una de las principales dificultades fue no distinguir claramente las pausas entre letras, lo que dificultaba la correcta interpretación del mensaje. Además, si nos perdíamos al escuchar el audio, era necesario reproducirlo nuevamente para intentar identificar en qué parte se encontraba el error y cuál letra no se había comprendido bien. Esto hacía el proceso más lento y propenso a confusiones.

1.3 Tercera parte: conmutación de mensajes

Para la transmisión de mensajes dentro del grupo, se estableció el siguiente protocolo con el objetivo de organizar el envío y la entrega de la información entre los clientes a través de un conmutador.

Cada cliente que desea enviar un mensaje debe comenzar indicando la letra inicial del nombre del destinatario. Esta letra actúa como un identificador del cliente que debe recibir el mensaje. Por ejemplo:

- La letra I representa a Isabella,
- La letra S representa a Silvia,
- La letra R representa a Ruth.

A continuación, se graba el mensaje en código Morse, utilizando combinaciones de puntos y rayas (pulsos cortos y largos), que representan cada letra del mensaje original.

El mensaje completo, compuesto por la inicial del destinatario y el contenido codificado en Morse, se envía en una nota de voz a través de WhatsApp al conmutador del grupo. En este caso, la persona que cumple el rol de conmutador es Michelle.

El conmutador escucha el mensaje recibido, identifica a quién va dirigido gracias a la inicial mencionada, y luego se encarga de reenviar el mensaje al cliente correspondiente. El conmutador puede también interpretar el mensaje si es necesario antes de enviarlo.

Antes de cada nuevo envío, el conmutador debe indicar si está "libre" o "ocupado", para que los clientes sepan si pueden enviar su mensaje. Si está ocupado, los clientes deberán esperar hasta que esté disponible, lo cual simula una red saturada o en proceso.

Este sistema permite organizar mejor la comunicación, distribuir el uso del canal de forma ordenada, y reflexionar sobre el funcionamiento de redes conmutadas en la práctica.

¿Qué posibilidades incluye la introducción de un conmutador en el sistema?

La introducción de un conmutador en nuestro protocolo permite centralizar la comunicación, organizando el flujo de mensajes entre los clientes. Esto facilita el control sobre a quién se dirige cada mensaje y permite simular una red más realista, en la que no todos los dispositivos están conectados directamente entre sí. Además, al exigir que el conmutador indique si está libre u ocupado, se introduce la posibilidad de gestionar la disponibilidad, evitar la sobrecarga y ordenar el turno de los envíos. También permite que el conmutador verifique o incluso descifre el mensaje, añadiendo una capa de validación.

¿Qué ventajas/desventajas se tienen al momento de agregar más conmutadores al sistema?

Ventajas:

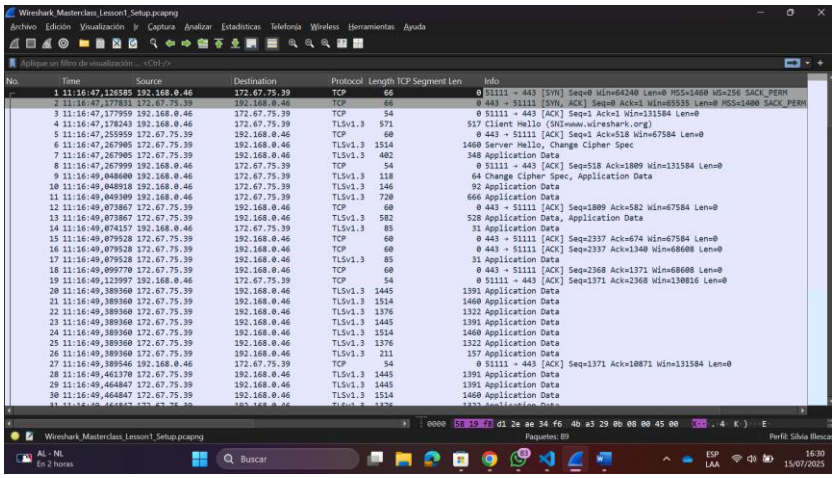
- **Mayor eficiencia:** Se pueden procesar más mensajes simultáneamente, reduciendo los tiempos de espera para los clientes.
- **Distribución de carga:** Al dividir los mensajes entre varios conmutadores, se evita la saturación de uno solo.
- **Redundancia:** Si un conmutador está ocupado o falla, otro puede tomar su lugar, lo cual aumenta la robustez del sistema.

Desventajas:

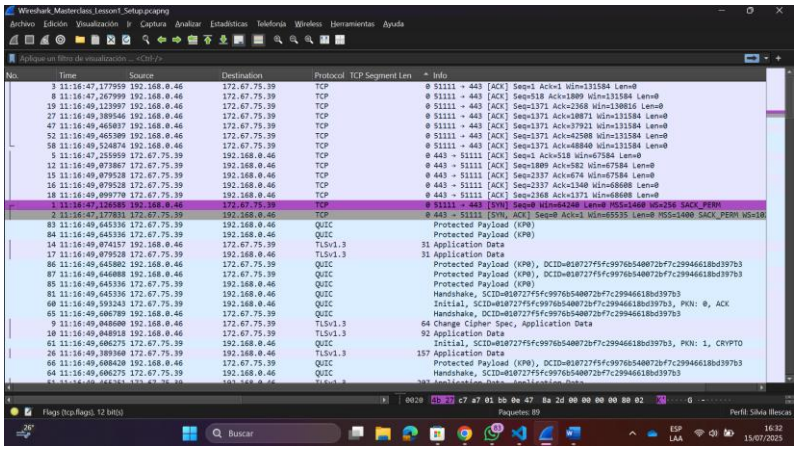
- **Coordinación más compleja:** Requiere establecer reglas claras para saber qué conmutador se encarga de cada cliente o mensaje.
- **Posibles confusiones:** Si no se indica correctamente a qué conmutador enviar, puede haber errores o retrasos.
- **Duplicación o pérdida de mensajes:** Si dos conmutadores no están sincronizados, podrían reenviar el mismo mensaje o dejar uno sin procesar.

Introducción a Wireshark

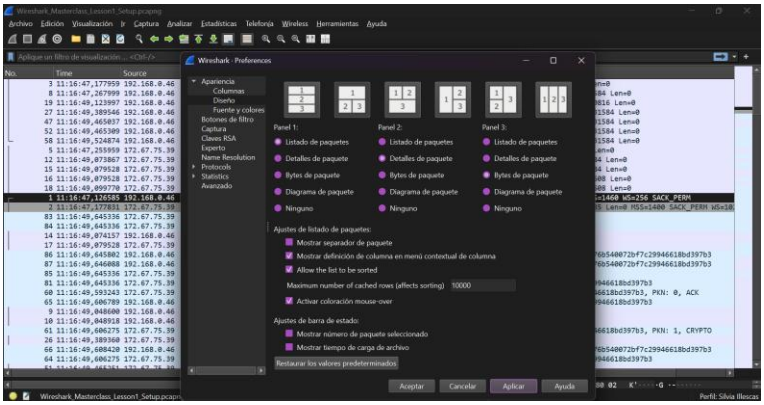
1.1 Primera parte: personalización del entorno



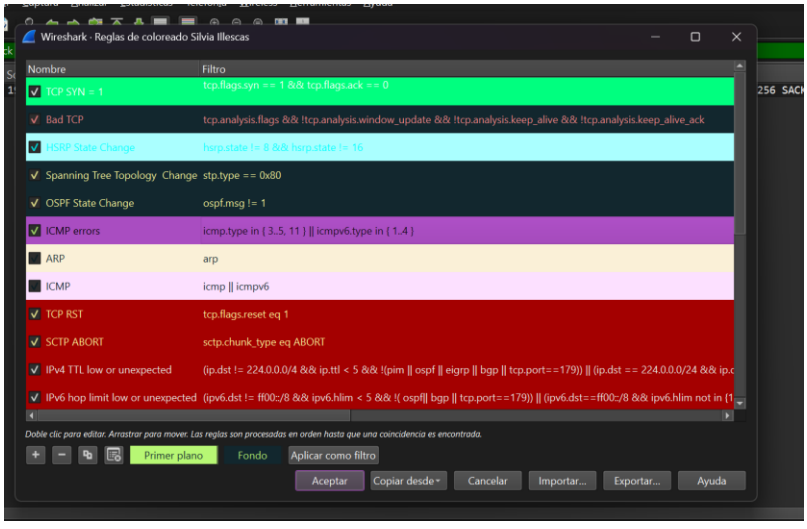
Perfil creado



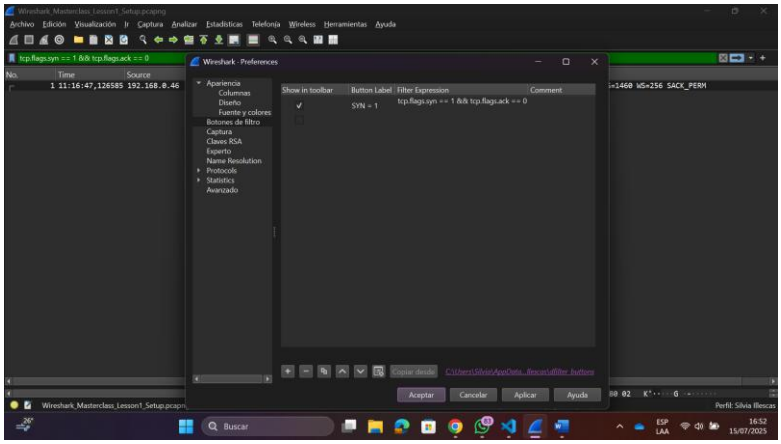
Time of day, TCP y eliminación de columna longitud



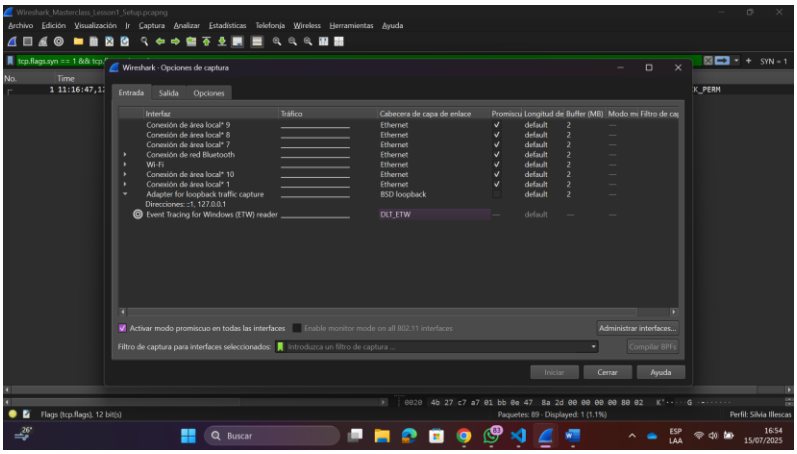
Esquema de paneles de mi preferencia



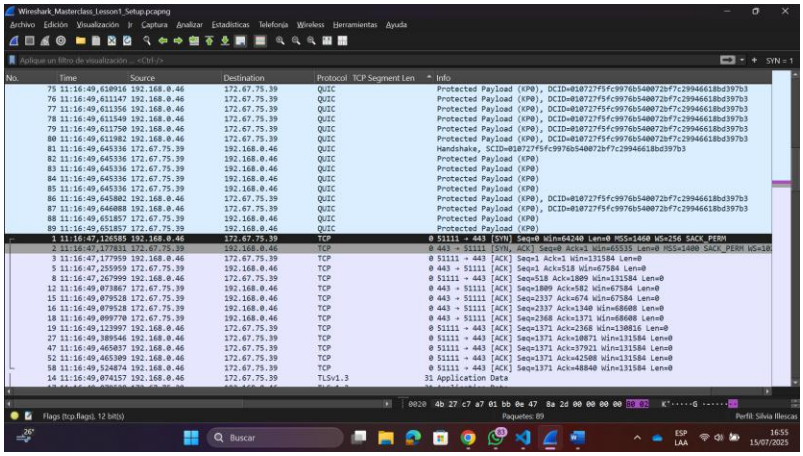
Regla de color para TCP



Botón que aplique filtro para paquetes TCP



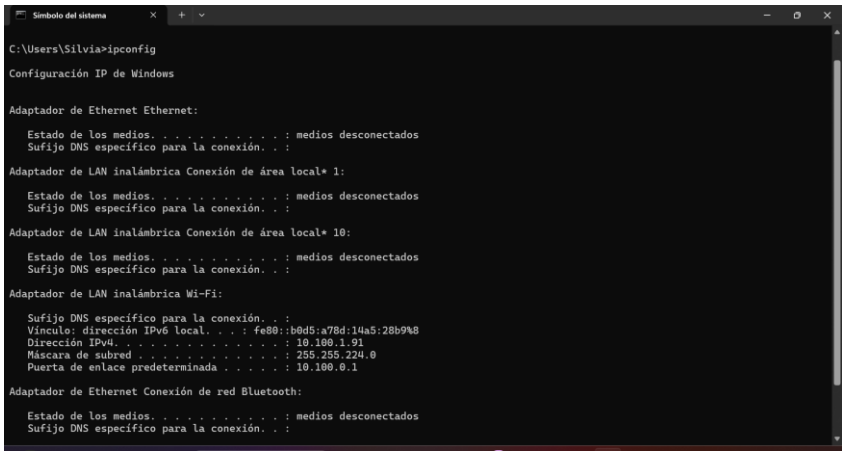
Ocultar interfaces virtuales



Entorno final luego de configuración

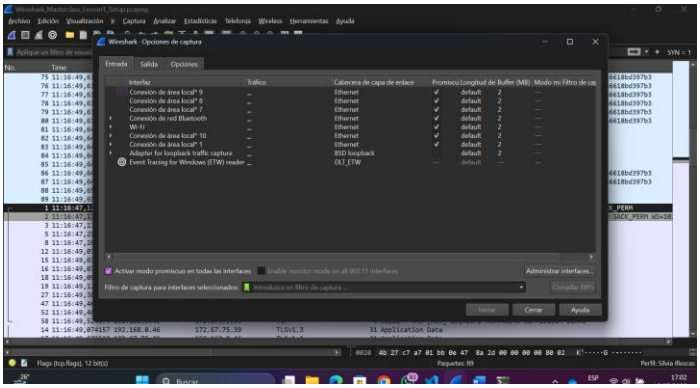
1.2 Segunda parte: configuración de la captura de paquetes

1. Ipconfig

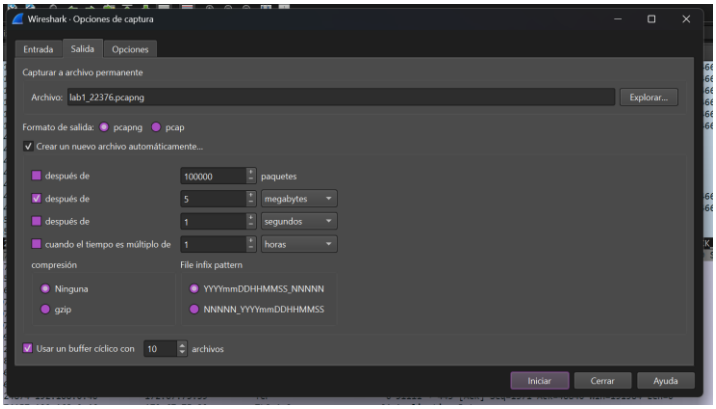


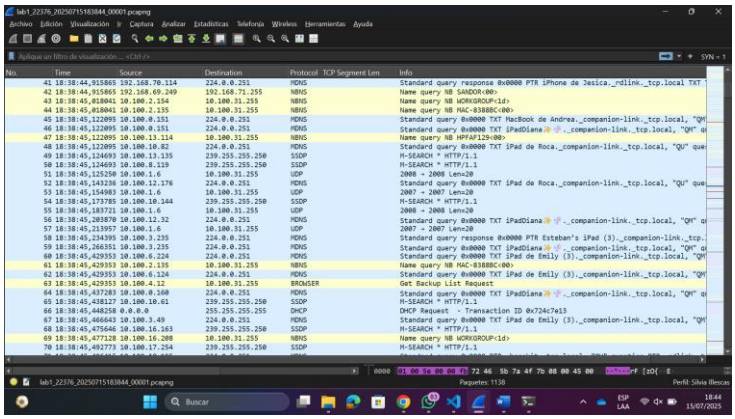
Según el comando ipconfig, la interfaz activa es la de LAN inalámbrica Wi-Fi con IP 10.100.1.91, por lo que se usará esta para la captura.

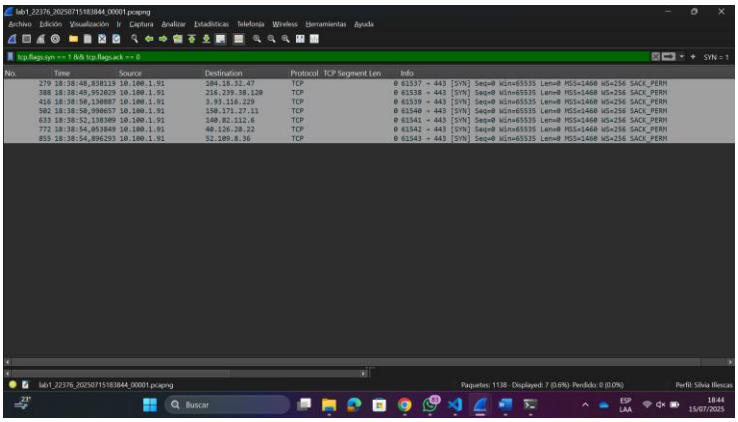
2. Desactivar interfaces virtuales



3. Captura de paquetes con los paquetes con la interfaz de wifi

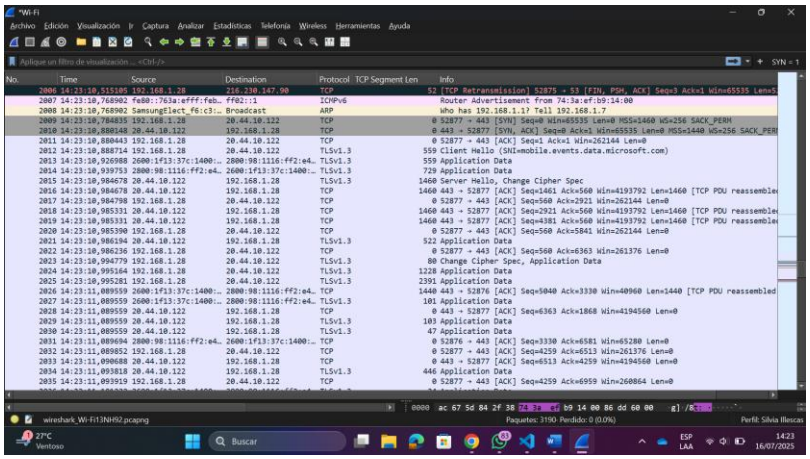


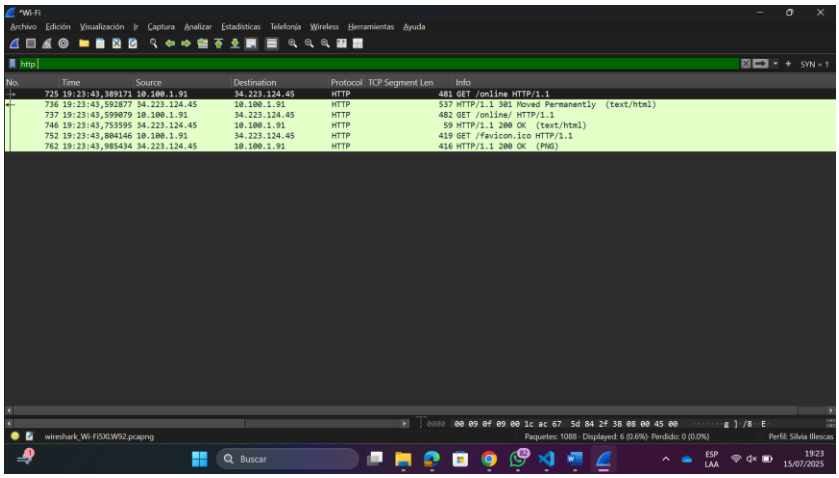




1.3 Tercera parte: análisis de paquetes

1. inicie una captura de paquetes en Wireshark (sin filtro) para lograr la captura del protocolo HTTP





a. ¿Qué versión de HTTP está ejecutando su navegador?
La versión de HTTP que está ejecutando el navegador es HTTP/1.1, como se observa en la línea de solicitud GET /online HTTP/1.1.

b. ¿Qué versión de HTTP está ejecutando el servidor?
El servidor también está ejecutando **HTTP/1.1**, lo cual se evidencia en las respuestas como HTTP/1.1 301 Moved Permanently y HTTP/1.1 200 OK.

c. ¿Qué lenguajes (si aplica) indica el navegador que acepta al servidor?
El navegador indica los lenguajes que acepta al servidor en el campo Accept-Language del encabezado HTTP, acepta contenido en español y en inglés.

d. ¿Cuántos bytes de contenido fueron devueltos por el servidor?
En la captura, el servidor devolvió 419 bytes para el archivo favicon.ico y 416 bytes para el archivo .png. En total, fueron devueltos 835 bytes de contenido.

e. En el caso que haya un problema de rendimiento mientras se descarga la página, ¿en qué dispositivos de la red convendría "escuchar" los paquetes? ¿Es conveniente instalar Wireshark en el servidor? Justifique.
Conviene escuchar los paquetes en el cliente (navegador) y en dispositivos intermedios como un **router** con port mirroring, ya que ahí se puede observar el tráfico completo. No es conveniente instalar Wireshark en el servidor porque puede consumir recursos del sistema, y no siempre se tiene acceso administrativo. Además, observar desde otro punto de la red permite detectar mejor los cuellos de botella o pérdidas de paquetes.

Discusión

La actividad me permitió experimentar de forma práctica cómo se codifican y transmiten mensajes en una red. Al comparar Morse y Baudot, observé que la complejidad del código influye en la velocidad y precisión de la comunicación. El uso de notas de voz introdujo errores auditivos, mostrando cómo el medio afecta la interpretación del mensaje.

La simulación del conmutador ayudó a entender el rol de los nodos intermediarios y los desafíos del tráfico en redes centralizadas, como los retrasos y la gestión por turnos.

Con Wireshark, analicé paquetes reales, identificando protocolos, direcciones IP y puertos. Esta herramienta evidenció la complejidad de una simple conexión a internet y destacó la importancia de usarla en entornos de prueba por razones de seguridad y rendimiento.

Comentarios

La práctica fue dinámica y útil para comprender conceptos teóricos de redes desde la experiencia. La comparación entre distintos métodos de codificación y transmisión resaltó la importancia de la claridad en la comunicación. La actividad del conmutador mostró de forma sencilla cómo se gestiona el tráfico de información en redes reales. Además, el uso de Wireshark permitió ver en detalle cómo se estructura la comunicación digital, lo que fortaleció el entendimiento de protocolos y herramientas de análisis. Fue una experiencia enriquecedora tanto a nivel técnico como colaborativo.

Conclusiones

- La codificación y transmisión de mensajes mediante métodos como Morse y Baudot permitieron comprender la importancia de la precisión y la sincronización en la comunicación entre dispositivos.

- El ejercicio del conmutador evidenció cómo funciona la intermediación y la gestión del tráfico en una red, destacando los posibles cuellos de botella y la necesidad de coordinación.

- El uso de Wireshark facilitó la visualización y análisis del tráfico de red, reforzando conocimientos sobre protocolos y el comportamiento real de las redes de comunicación.

Referencias

Mozilla Developer Network. (2024). *HTTP overview*. https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

Museum of Telephony. (n.d.). *Baudot code specification*. https://cs.stanford.edu/people/eroberts/courses/soco/projects/2008-09/colossus/baudot.html

Wireshark Foundation. (2024). *Wireshark User's Guide*. https://www.wireshark.org/docs/wsug_html_chunked/