****

**Ingeniería en Sistemas Computacionales**

**Centro de Ciencias Básicas**

**5º C**

**Proyecto redes I**

**Alumnos:**

353079 Tadeo Andrade Sustaita

350114 Cindy Fabiola Hernández Muñoz

350944 Reyli Uvaldo Martínez Hernández

281547 Juan Carlos Uriarte Padilla

**Profesor:** Javier Santiago Cortes López

**Fecha de entrega:** 17 de diciembre de 2024

**Contenido**

[Introducción 4](#_Toc185455555)

[Objetivos 4](#_Toc185455556)

[Marco teórico 4](#_Toc185455557)

[Tecnologías utilizadas 4](#_Toc185455558)

[Temas del Curso Aplicados en el Proyecto 4](#_Toc185455559)

[Desarrollo y funcionalidades 5](#_Toc185455560)

[Interfaz de usuario 5](#_Toc185455561)

[Captura de Paquetes 5](#_Toc185455562)

[Filtros de captura 5](#_Toc185455563)

[Exportación de Datos 5](#_Toc185455564)

[Especificaciones generales 5](#_Toc185455565)

[Explicación detallada de lo realizado por cada punto solicitado 6](#_Toc185455566)

[Interfaz de Usuario 6](#_Toc185455567)

[Captura de Paquetes 6](#_Toc185455568)

[Implementación de Filtros 6](#_Toc185455569)

[Funcionalidad de Exportación 7](#_Toc185455570)

[Principales problemas 7](#_Toc185455571)

[Problema de Compatibilidad con Librerías de Captura 7](#_Toc185455572)

[Dificultades en la Interfaz Gráfica 7](#_Toc185455573)

[Fallas en la Exportación a CSV/Excel 7](#_Toc185455574)

[Problemas al Aplicar los Filtros 7](#_Toc185455575)

[Mala Comunicación en el Equipo 8](#_Toc185455576)

[Importancia de funcionalidades implementadas como Outputs 8](#_Toc185455577)

[Captura en Tiempo Real 8](#_Toc185455578)

[Aplicación de Filtros 8](#_Toc185455579)

[Exportación a CSV 8](#_Toc185455580)

[Enlace de descarga del programa 8](#_Toc185455581)

[Manual del usuario 9](#_Toc185455582)

[Futuras mejoras 9](#_Toc185455583)

[Conclusiones 9](#_Toc185455584)

[Conclusión general 9](#_Toc185455585)

[Conclusiones personales 10](#_Toc185455586)

[Bibliografías 11](#_Toc185455587)

**REPORTE DE PROYECTO: PACKET SNIFFER**

Introducción

El objetivo principal de este proyecto es el desarrollo de un software que implemente las funciones esenciales de un Packet Sniffer, permitiendo la captura y el análisis detallado de paquetes de red. Para ello, se utilizará el lenguaje de programación C y la biblioteca libpcap, que proporciona acceso a la información del tráfico de red. La herramienta está diseñada para ser intuitiva, eficiente y fácil de usar, ofreciendo a los usuarios la capacidad de visualizar y exportar el tráfico de red de manera efectiva, facilitando así tareas de monitoreo y diagnóstico de redes.

## Objetivos

El proyecto busca crear un software que:

* Capture paquetes de red en tiempo real.
* Permita aplicar filtros para analizar el tráfico.
* Tenga una interfaz fácil de usar.
* Exporte los resultados a un archivo CSV.

Marco teórico

## Tecnologías utilizadas

Para el desarrollo de este proyecto, decidimos utilizar el lenguaje de programación C, ya que todos los integrantes del equipo tenemos conocimientos previos en este lenguaje, lo que nos permitió facilitar su implementación y desarrollar el proyecto de manera más eficiente. Además, optamos por trabajar en Linux, ya que a cada uno de nosotros nos resultó más sencillo realizar la primera parte del proyecto en este sistema operativo, aprovechando su compatibilidad y las herramientas disponibles, como libpcap, que fue esencial para la captura de paquetes de red. También decidimos implementar una interfaz de texto, ya que su desarrollo fue sencillo, eficiente y adecuado para cumplir con los objetivos del proyecto.

## Temas del Curso Aplicados en el Proyecto

Durante el proyecto, aplicamos varios conceptos aprendidos en clase. Comenzamos por entender la estructura de los paquetes de red y la información que transportan. También trabajamos con protocolos clave como TCP, UDP e ICMP. Para analizar el tráfico de red, utilizamos filtros basados en criterios como direcciones IP, puertos y protocolos. Todo esto fue fundamental para cumplir los objetivos del proyecto y poner en práctica los conocimientos adquiridos en clase.

# Desarrollo y funcionalidades

Interfaz de usuario:

La interfaz de usuario, basada en texto, fue diseñada con el objetivo de ser intuitiva y fácil de usar. Está estructurada en tres áreas claramente diferenciadas para facilitar la navegación y el análisis del tráfico de red. La primera área, Tráfico Capturado, muestra los paquetes en tiempo real, permitiendo a los usuarios ver de manera continua los datos que están siendo transmitidos. La segunda área, Contenido Raw, presenta el contenido crudo del paquete seleccionado, lo que permite un análisis más detallado y profundo de cada paquete. Además, se incluye un menú de usuario que ofrece opciones para cambiar entre diferentes funcionalidades, como consultar los paquetes, ver detalles específicos de alguno, aplicar filtros personalizados o exportar los resultados a un archivo csv.

Captura de Paquetes:

Se implementó la funcionalidad básica para la captura en tiempo real de paquetes, lo que permite a los usuarios observar el tráfico de red a medida que ocurre. Esta herramienta facilita la visualización detallada de los paquetes transmitidos, mostrando información clave sobre cada uno, como el número de paquete, el tiempo de captura, el protocolo utilizado, la dirección IP de origen, la dirección IP de destino y la longitud del paquete.

Filtros de captura:

El sistema permite aplicar diferentes filtros para ajustar la captura de paquetes según lo que el usuario necesite. Los filtros disponibles incluyen la opción de filtrar por IP de destino, IP de origen, puerto de origen, puerto de destino y protocolo. Estos filtros ayudan a que el usuario pueda ver solo los paquetes que le interesan, haciendo más fácil el análisis del tráfico de red.

Exportación de Datos:

El sistema permite exportar los resultados de la captura de tráfico a un archivo CSV para su posterior análisis, facilitando el almacenamiento y la manipulación de los datos fuera del programa.

# Especificaciones generales

El proyecto está desarrollado en Lenguaje de Programación C y se ejecuta sobre el Sistema Operativo Linux. Se utilizó la biblioteca libpcap para capturar y analizar el tráfico de red. Es importante destacar que el programa es estático, lo que significa que no se desarrolló un instalador para su instalación. Además, no cuenta con la opción de ver el tráfico de diferentes tarjetas de red, limitándose a la tarjeta configurada en el sistema.

# Explicación detallada de lo realizado por cada punto solicitado

Interfaz de Usuario:

La interfaz fue diseñada para ser sencilla y basada en texto, lo que facilita tanto su implementación como su uso. Está organizada en tres secciones principales para ofrecer al usuario una experiencia intuitiva y accesible. La primera sección, Tráfico Capturado, muestra los paquetes de red que se están capturando en tiempo real. La segunda sección, Contenido Raw, presenta los datos crudos del paquete seleccionado. Además, se incluye un menú de usuario que permite acceder a varias funcionalidades, como iniciar la captura de paquetes, ver los paquetes capturados, obtener detalles de algún paquete en específico, aplicar filtros personalizados o exportar los datos a un archivo CSV. Sin embargo, cabe destacar que la interfaz no incluye un área en la que se muestre la información de los paquetes de manera estructurada.

Captura de Paquetes:

La funcionalidad de captura de paquetes se implementó utilizando la biblioteca libpcap, que proporciona las herramientas necesarias para acceder al tráfico de red y capturar los paquetes transmitidos. La función call\_me es la encargada de procesar cada paquete capturado. Cuando se captura un paquete, la función obtiene información relevante, como la dirección IP de origen y destino, el protocolo utilizado y la longitud del paquete. Además, se calcula el tiempo de captura en milisegundos a partir de la marca de tiempo del paquete, y esta información se almacena en una estructura de datos llamada packets. Posteriormente, se imprime la información del paquete en tiempo real utilizando la función print\_packet\_info. De esta manera, el usuario puede observar el tráfico de red conforme se captura, con detalles como el número de paquete, la dirección IP de origen y destino, el protocolo, la longitud del paquete y el tiempo de captura.

Implementación de Filtros:

El sistema permite aplicar filtros para personalizar la captura de paquetes y facilitar el análisis. Utilizando libpcap, se implementó un sistema de filtrado que permite al usuario aplicar criterios como la IP de origen, la IP de destino, el puerto de origen, el puerto de destino y el protocolo. La función apply\_filter se encarga de compilar y aplicar los filtros utilizando el formato BPF. Dependiendo del tipo de filtro seleccionado por el usuario, se configura el filtro correspondiente mediante funciones como set\_filter\_src\_ip para filtrar por IP de origen, set\_filter\_dst\_ip para filtrar por IP de destino, set\_filter\_src\_port y set\_filter\_dst\_port para filtrar por puertos de origen y destino, respectivamente, y set\_filter\_protocol para filtrar por protocolo. Cada uno de estos filtros se aplica al tráfico de red, permitiendo al usuario ver solo los paquetes que cumplen con los criterios especificados.

Funcionalidad de Exportación:

La funcionalidad de exportación permite al usuario guardar los datos capturados en un archivo CSV. La función export\_data\_csv abre un archivo con el nombre proporcionado y escribe los datos de cada paquete capturado en él. El archivo CSV contiene un encabezado con las columnas correspondientes, como el número de paquete, el tiempo de captura, el protocolo, las direcciones IP de origen y destino, y la longitud del paquete. Los datos se escriben en el archivo y el tiempo de captura se convierte a segundos y milisegundos para facilitar su interpretación. Al finalizar, el archivo se cierra y el usuario recibe una notificación de que los datos se han exportado correctamente.

# Principales problemas

Problema de Compatibilidad con Librerías de Captura:

Inicialmente, tuvimos dificultades para lograr que la librería libpcap funcionara correctamente en diferentes distribuciones de Linux. Esto se debía a la falta de dependencias necesarias y a la configuración incorrecta de los permisos de acceso para realizar capturas a nivel de red.

Investigamos las dependencias requeridas y ajustamos los permisos del sistema, asegurando así que la captura de paquetes pudiera ejecutarse sin errores.

Dificultades en la Interfaz Gráfica:

Aunque teníamos conocimientos necesarios para implementar una interfaz gráfica, la complejidad de integrarla con las demás funcionalidades del proyecto generó algunas dudas sobre la mejor forma de llevarla a cabo. Por esta razón, decidimos optar por una interfaz basada en texto, que resultó más sencilla de implementar y proporcionó un funcionamiento más estable. De esta forma, pudimos enfocar nuestros esfuerzos en el desarrollo y optimización de las demás funcionalidades, asegurando un mejor rendimiento global del proyecto.

Fallas en la Exportación a CSV/Excel:

En un inicio, la exportación de datos a un archivo CSV no incluía toda la información requerida o presentaba un formato incorrecto, dificultando su uso en otras herramientas.  
Resolvimos el problema mejorando el código de exportación. Aseguramos que los datos estuvieran correctamente formateados y completos, además de implementar un sistema de validación para garantizar su integridad y precisión.

Problemas al Aplicar los Filtros:

Tuvimos dificultades al implementar la funcionalidad de filtros, ya que al principio no sabíamos cómo aplicarlos correctamente dentro del programa.

Logramos solucionarlo investigando más a fondo el uso de filtros con la librería libpcap. Finalmente, conseguimos integrarlos de manera efectiva.

Mala Comunicación en el Equipo:

Un desafío constante fue la falta de comunicación efectiva entre los integrantes del equipo. Frecuentemente surgían desacuerdos sobre cómo abordar ciertas tareas, ya que cada miembro tenía ideas diferentes sobre la implementación.

Lo superamos dialogando de manera abierta y clara. Establecimos acuerdos concretos antes de continuar con las tareas, lo que mejoró la colaboración y nos permitió avanzar de forma organizada.

# Importancia de funcionalidades implementadas como Outputs

Captura en Tiempo Real:

Permite monitorear el tráfico de red de forma instantánea y continua, lo que resulta fundamental para identificar y analizar problemas de conectividad o posibles amenazas de seguridad en la red en tiempo real.

Aplicación de Filtros:

Los filtros permiten a los usuarios seleccionar paquetes específicos, optimizando el análisis y ofreciendo una forma más precisa y enfocada de examinar el tráfico de red, lo cual facilita la identificación de patrones o anomalías.

Exportación a CSV:

La capacidad de exportar los datos capturados a un archivo CSV proporciona una forma práctica de almacenar la información para su análisis posterior, permitiendo manipular y examinar los resultados con herramientas externas, lo que mejora la interpretación y el uso de los datos.

# Enlace de descarga del programa

Para acceder y descargar el programa, se debe utilizar el siguiente enlace al repositorio en GitHub:

<https://github.com/JuanCarlos-UriartePadilla/Proyecto-Redes/tree/main>

Manual del usuario

Se elaboró un manual de usuario en formato electrónico, que explica detalladamente cómo utilizar el software, aplicar filtros y exportar los resultados.

El manual incluye los siguientes apartados:

* Introducción al uso del software.
* Cómo iniciar y detener la captura de paquetes.
* Aplicación de filtros.
* Exportación de datos.

Enlace al manual de usuario:

[manual.docx](https://eduuaa-my.sharepoint.com/:w:/g/personal/al350114_edu_uaa_mx/EZW-MRDkFDJGp-kpRlvaW5cBvPQ1B-CU42wtHCuYjBRcnQ?e=byqoZg)

Futuras mejoras

Para seguir mejorando el software, se sugerimos las siguientes acciones:

* Interfaz gráfica más completa: Desarrollar una interfaz gráfica más avanzada que facilite una experiencia de usuario más intuitiva y visual, permitiendo a los usuarios interactuar de manera más eficiente con las diferentes funcionalidades del programa.
* Opciones de filtrado más avanzadas: Añadir más opciones de filtrado para un análisis más avanzado.
* Captura de tráfico en redes más complejas: Incluir la capacidad de capturar tráfico en redes más complejas.

# Conclusiones

Conclusión general:

El proyecto nos permitió desarrollar una herramienta básica para capturar y analizar el tráfico de red. A lo largo del proceso, enfrentamos varios desafíos que, aunque complicados, nos dejaron aprendizajes valiosos y nos ayudaron a mejorar nuestras habilidades.

Si bien logramos cumplir algunos objetivos, reconocemos que hay áreas importantes para mejorar, como el desarrollo de una interfaz gráfica más completa y opciones de filtrado más avanzadas. La verdad es que no quedamos del todo conformes con el resultado, ya que no logró cumplir nuestras expectativas. Sin embargo, estamos conscientes del esfuerzo que pusimos como equipo. Este proyecto nos enseñó la importancia de investigar con calma, planificar con claridad y trabajar con suficiente tiempo de anticipación para evitar tanto estrés.

Conclusiones personales:

Tadeo: Este proyecto fue un éxito en muchos aspectos, ya que logramos cumplir con los objetivos propuestos. Desarrollamos una herramienta funcional para la captura y análisis de paquetes de red, que facilita el acceso eficiente a los datos de tráfico utilizando la librería libpcap. Las opciones de exportación añadidas también mejoraron la utilidad del software, permitiendo un análisis más profundo y organizado del tráfico capturado. Me siento satisfecho con los resultados obtenidos y con las lecciones aprendidas durante el desarrollo de este proyecto. Mi contribución principal fue en la implementación de los filtros, lo cual permitió a los usuarios personalizar y enfocar el análisis del tráfico capturado según sus necesidades.

Fabiola: Aunque no me siento completamente satisfecha con el resultado final, considero que logramos cumplir con algunos de los objetivos planteados al inicio del proyecto. Se desarrolló una herramienta funcional para la captura y análisis de paquetes de red, con una interfaz amigable y opciones útiles para los usuarios. A pesar de los desafíos, me quedo con valiosos aprendizajes y lecciones que me ayudarán en futuros proyectos. Mi contribución principal abarcó el diseño de la interfaz de texto, la implementación de la funcionalidad para exportar los paquetes capturados a un archivo, la documentación del proyecto y la elaboración del video explicativo. Confío en que, con futuras mejoras, esta herramienta podrá perfeccionarse y cumplir con todas las expectativas planteadas.

Reyli: A lo largo de este proyecto, logramos desarrollar una herramienta funcional para la captura y análisis de paquetes de red, cumpliendo con varios de los objetivos planteados al inicio. La librería libpcap fue clave para acceder de manera eficiente a los datos de tráfico, y la capacidad de exportar los resultados a archivos añadió un valor importante al software. Mi contribución se centró en la implementación de la función para consultar los paquetes capturados y mostrar sus detalles. Aunque existen áreas de mejora, estoy satisfecho con el progreso alcanzado y con los aprendizajes obtenidos durante el desarrollo del proyecto.

Juan Carlos: El proyecto representó una gran oportunidad para poner en práctica nuestros conocimientos y adquirir nuevas habilidades. Conseguimos desarrollar una herramienta funcional capaz de capturar y analizar paquetes de red mediante libpcap. Mi aporte principal consistió en implementar la funcionalidad para iniciar la captura de paquetes, editar el video explicativo y subir el proyecto al repositorio de GitHub. A pesar de los desafíos que enfrentamos, estoy satisfecho con los resultados obtenidos y con todo lo aprendido a lo largo del proceso.

# Bibliografías

*Wireshark · Go Deep*. (s. f.). Wireshark. <https://www.wireshark.org/>

Tanwar, P. (2021, 16 julio). Capturing packets in C program using LIBPCaP | Open Source. *Open Source For You*. <https://www.opensourceforu.com/2011/02/capturing-packets-c-program-libpcap/>

Talal. (2023b, agosto 2). Building a packet sniffer: the basics | Medium. *Medium*. <https://talalio.medium.com/building-a-packet-sniffer-9460f394041>

Talal. (2024, 26 abril). Building a packet sniffer Part 2 : Layers headers | Medium. *Medium*. <https://talalio.medium.com/building-a-packet-sniffer-part-2-6fb33bd68d53>

Talal. (2023, 2 agosto). Building a packet sniffer Part 3:  Filters | Medium. *Medium*. <https://talalio.medium.com/building-a-packet-sniffer-part-3-a404e60d91c5>