TP Redes E.T. 36 5TO 1ra

ESCÁNER DE RED

Materia: Redes
Modalidad: Individual
Tecnología: C# o Java
Autora: Carolina Jackeline Lobo

1. Descripción del Proyecto

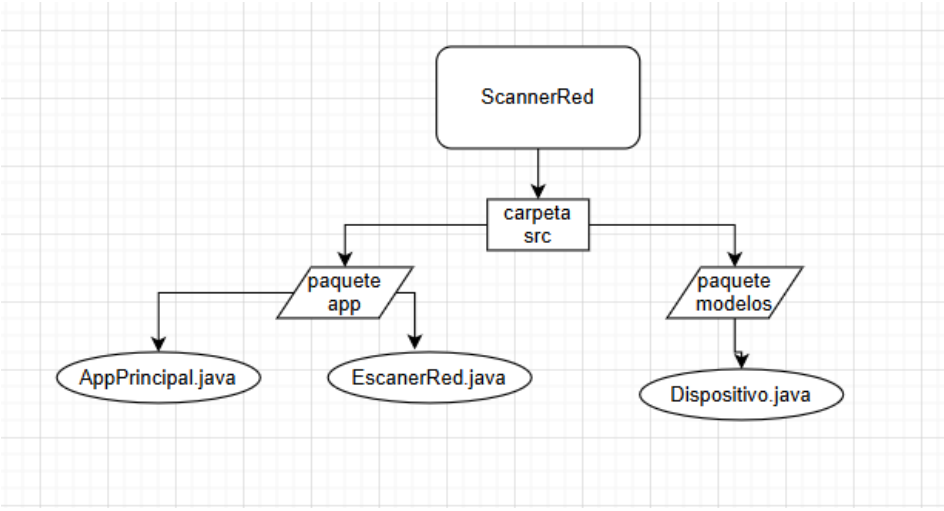
La herramienta **Red Scanner** está diseñada para identificar y listar todos los dispositivos conectados dentro de una red local. Su función principal es facilitar el diagnóstico y monitoreo de la infraestructura de red, permitiendo a los usuarios obtener información detallada sobre cada dispositivo, como su dirección IP, nombre, estado y otros datos relevantes.

2. Documentación del desarrollo

Para qué sirve el programa

- Detectar dispositivos activos en la red local.
- Mostrar información básica como IP, nombre y estado.
- Ayudar a administrar y controlar el uso de la red.

Cómo está armado el sistema (con diagramas)



Esta compuesto por una carpeta principal del proyecto y SCR que contiene:

- Tres clases principales:
    - **EscanerRed:** Responsable de realizar el escaneo de la red, enviando pings a rangos de direcciones IP para detectar dispositivos activos.
    - **Dispositivo:** Representa cada equipo o nodo encontrado en la red, almacenando información relevante como IP, nombre y estado.
    - **AppPrincipal:** Controla la interfaz gráfica y la interacción con el usuario, mostrando los resultados del escaneo.

Se incluye además un diagrama de flujo general del funcionamiento del programa, que consiste en:

- Iniciar escaneo de todos los dispositivos conectados entre 2 ips
- Detectar dispositivos en estas con sus características
- Mostrar resultados al usuario con todos sus respectivas caracteristicas

Qué métodos se usaron y por qué

- scanNetwork() - Escaneo de red (Ping a rangos de IP)

    Métodos involucrados:

    - escanearRangoAvanzado(): Punto de entrada principal para escanear un rango de IPs usando hilos.
    - escanearRango(): Gestiona el escaneo concurrente de IPs y actualiza el progreso.
    - pingSistema(): Realiza ping a una IP usando comandos del sistema operativo (Windows/Unix).
    - ipToLong() y longToIp(): Convierten IPs entre formato string y numérico para manejar rangos.
    - validarIP() y validarRangoIPs(): Validan que las IPs y rangos sean correctos.

    Propósito:
    Escanear un rango de IPs de forma eficiente (con concurrencia) para identificar dispositivos activos en la red.
    Optimizar el tiempo de escaneo mediante timeouts configurables y multihilos.

- getDeviceInfo() - Obtención de datos del dispositivo
  Métodos involucrados:

  - escanearDispositivo(): Coordina el ping, resolución de nombre y escaneo de puertos para un dispositivo.
  - obtenerNombreHost(): Obtiene el nombre del dispositivo usando nslookup (sistema) o Java (fallback).
  - ejecutarComandoNSLookup() y obtenerNombreHostJava(): Implementan la resolución de nombres.
  - escanearPuertos() y puertoEstaAbierto(): Escanean puertos predeterminados (21, 22, 80, etc.) en un dispositivo activo.

  Propósito:
  Recolectar información detallada de cada dispositivo (estado, nombre, puertos abiertos).
  Proporcionar datos útiles para diagnóstico de red o inventario.

- Métodos para actualizar la interfaz gráfica Métodos involucrados:
  - actualizarProgreso(): Actualiza una JProgressBar (barra de progreso) en tiempo real.
  - inicializarEscaneo(), detenerEscaneo(), estaEscaneando(): Controlan el estado del escaneo desde la UI.

  Propósito:
  Mostrar el progreso del escaneo al usuario (feedback visual).
  Permitir la interacción (inicio/detención) desde una interfaz gráfica (Swing/JavaFX).

- Métodos auxiliares Gestión de hilos:
  - esperarFinalizacion(), finalizarEscaneo(), manejarError().

  Propósito:
  Garantizar estabilidad al manejar concurrencia y errores.

Por qué se eligieron ciertas tecnologías

- **Java:** Es un lenguaje que me enseñaron en la institucion en la que estudio, por lo que ya tengo la mano más suelta en este tipo de lenguajes

- **Librerías:** Mas que nada utilice mucho java.util.. Para listas y objets
- **Eclipse:** Se utilizó como entorno de desarrollo integrado por su facilidad para gestionar proyectos y depurar código.. Aunque podría utilizarse otra como Visual Studio también sin ningún problema

Qué problemas aparecieron y cómo se solucionaron

- Restricciones de permisos para ejecutar pings, solucionado ejecutando el programa con privilegios adecuados.
- Errores con la gráfica visual, arreglado con muchos intentos de cambio de métodos de `UIManager`
- Detección incompleta de algunas IPs, manejado mediante el uso de excepciones para evitar que el programa se caiga y aplicando validaciones en los rangos de IP.
- Algunos errores al escanear IPs inválidas, por lo que se recomienda validar cuidadosamente las direcciones antes de iniciar el escaneo.
- Fallos con el botón “LIMPIAR”, arreglado con correcciones en el método

Qué se podría mejorar en el futuro

- Implementar escaneo multihilo para acelerar el proceso.
- Mejorar la interfaz gráfica para una experiencia más intuitiva y visual.
- Añadir la opción de exportar resultados en formatos como PDF o EXCEL para facilitar su uso externo.
- Ampliar soporte para redes más grandes o remotas, optimizando tiempos de espera y precisión.