Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI4835 Redes de Computadoras I
Sartenejas, 2022

Laboratorio N° 1

Integrantes:

Eros Cedeño	16-10216 (Ing en computación)
Valentina Tomasello	16-11158 (Ing en telecomunicaciones)
Adelina Figueira	15-10484 (Ing en computación)
Pedro Rodríguez	15-11264 (Ing en computación)
Naulymar Morillo	14-10722 (Ing en telecomunicaciones)

Reconocedor de Red

Para este laboratorio se implementaron dos utilitarios de red capaces de reconocer las interfaces de red del sistema, su estado básico, direcciones de red, direcciones físicas, conexiones activas en el equipo y el enrutador que atiende el sistema. El primer utilitario fue elaborado en lenguaje Python, mientras que el segundo fue elaborado en lenguaje C. A continuación, ahondaremos más en ambos utilitarios.

Utilitario elaborado en Python

Para este utilitario se hizo uso del paquete psutil en Python, el cual permite conseguir información muy variada sobre el sistema, incluyendo sus interfaces de red y las conexiones activas, siendo esto último de nuestro interés para el desarrollo de la herramienta.

Se eligió Python, y específicamente el paquete psutil, dado a que este es multiplataforma y es soportado en los siguientes sistemas operativos: Linux, Windows, macOS, FreeBSD, OpenBSD, NetBSD, Sun Solaris y AIX.

Para conseguir los puertos se utilizó la librería pyserial que se encuentra disponible para múltiples plataformas como Win32, OSX, Linux, BSD, Jython y IronPython.

Requisitos e instalación del utilitario

El requisito principal para poder hacer uso del utilitario, es tener instalado Python en la máquina, donde se requiere por lo menos la versión 3.9 de Python, dado a que las funcionalidades multiplataformas de psutil requieren de una versión mínima del lenguaje. Además, es necesario poder instalar los siguientes paquetes mediante el comando pip3 en Python:

- psutil (pip3 install psutil)
- pyserial (pip3 install pyserial)
- tabulate (pip3 install tabulate)

Se incluyó un archivo **requirements.txt** que puede ser utilizado de la forma **“pip3 install -r requirements.txt”** para instalar todos los paquetes automáticamente.

Uso del utilitario

Para ejecutar el utilitario, simplemente extraiga los archivos, ubíquese en el directorio donde se encuentra el archivo **network_recognizer.py** mediante su consola, y ejecútelo con el comando python de la forna

“python network_recognizer.py”. Tome en cuenta que el comando python puede variar dependiendo de su instalación, y algunas de las alternativas son: py, py3 o python3.
Una vez hecho esto, se mostrarán en la consola múltiples tablas con la información requerida en el enunciado del proyecto. A continuación como ejemplo se presenta un extracto de las tablas que deben mostrarse:

Interfaz	Estatus	Direccion	Familia	Mascara de Red	Paquetes enviados	Paquetes recibidos
Ethernet	inactiva	C8-F7-50-04-46-3B	AF_LINK		0	0
Ethernet	inactiva	169.254.4.134	AF_INET	255.255.0.0	0	0
Ethernet	inactiva	fe80::2c87:c846:a098:486	AF_INET6		0	0
vEthernet (WSL)	activa	00-15-5D-98-10-27	AF_LINK		32100	22847
vEthernet (WSL)	activa	172.29.0.1	AF_INET	255.255.240.0	32100	22847
vEthernet (WSL)	activa	fe80::56d:ebb1:e563:d5d8	AF_INET6		32100	22847
Loopback Pseudo-Interface 1	activa	127.0.0.1	AF_INET	255.0.0.0	0	0
Loopback Pseudo-Interface 1	activa	::1	AF_INET6		0	0

Direccion Local	Puerto Local	Direccion Destino	Puerto Destino	ID Proceso	Estatus
172.16.0.5	59002	35.190.56.82	443	2924	ESTABLISHED
127.0.0.1	49669	127.0.0.1	49668	2924	ESTABLISHED
172.16.0.5	60653	8.246.197.30	80	0	TIME_WAIT
172.16.0.5	60649	18.155.251.25	80	0	TIME_WAIT
127.0.0.1	49782	127.0.0.1	49781	5256	ESTABLISHED
127.0.0.1	49759	127.0.0.1	65001	5336	ESTABLISHED
172.16.0.5	60650	72.21.91.29	80	0	TIME_WAIT
172.16.0.5	60685	107.167.110.216	443	12648	ESTABLISHED

Utilitario elaborado en C

Para comparar las distintas implementaciones, teniendo en cuenta que algunos equipos más antiguos no tienen soporte para algunas características del intérprete de python (y por mera curiosidad), también se realizó un utilitario de red en lenguaje C. Este utilitario provee al usuario de toda la información que ofrece el utilitario anterior. A diferencia del utilitario en Python, este únicamente funciona para sistemas operativos basados en Unix, dado a que la API para acceder a esta clase de información en Windows, es totalmente distinta. Queda como una tarea a futuro realizar la implementación en Windows y comparar las APIs utilizadas.

Este utilitario hace uso de las librería **ifaddrs** y **netlink,** siendo la primera útil para obtener información sobre las interfaces de redes, y la segunda para establecer un diálogo con el kernel del

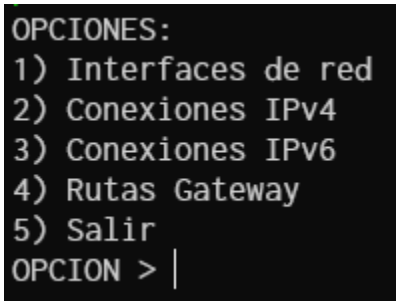
sistema operativo y así obtener información sobre las conexiones y las rutas usando los parámetros **NETLINK_SOCK_DIAG y NETLINK_ROUTE.**

Requisitos e instalación del utilitario

El único requisito para poder utilizar la herramienta, es tener un compilador de C disponible en la máquina donde se ejecutará. Se recomienda preferiblemente gcc que viene incluido en el paquete build-essentials de múltiples sistemas operativos. Una vez instalado el compilador, se debe compilar el archivo **network-suite-linux.c** y ya se podría ejecutar en la computadora.

Uso del utilitario

Primero debe ejecutar el archivo ejecutable generado al compilar el archivo **network-suite-linux.c,** y una vez hecho esto, se le presentará el siguiente menú en pantalla:



Simplemente introduzca el número de la opción que desee ver y presione la tecla Enter. Brevemente presentamos una explicación de que hace cada opción:

1. Interfaces de red: Muestra las interfaces de redes disponibles en una tabla con las siguientes columnas: Nombre de la interfaz, un estatus sencillo de la interfaz, tipo de dirección y dirección.

2. Conexiones IPv4: Muestra las conexiones IPv4 en una tabla con las siguientes columnas: Protocolo (UDP o TCP), dirección local y dirección remota.

3. Conexiones IPv6: Similar a la opción anterior pero para las conexiones IPv6.

4. Rutas Gateway: Muestra las rutas que hacen uso del gateway en una tabla con las siguientes columnas: Nombre de la interfaz de red, dirección de origen, dirección destino, gateway utilizado.

Comparación entre ambos utilitarios

A continuación, presentamos tres puntos de divergencias o diferencias entre ambos utilitarios, que nos gustaría comparar brevemente, explicando que beneficios y desventajas ofrece uno sobre el otro en cada punto.

Soporte para múltiples plataformas

- El utilitario en Python tiene una gran ventaja sobre el realizado en C, y es que este tiene soporte para múltiples plataformas (Windows, sistemas linux, macOs y otros) sin necesidad de escribir código específico para una. Esto se debe a que los paquetes en Python hacen todas estas implementaciones por debajo sin necesidad de preocuparnos

- Debido a las APIs usada por el utilitario en C, solo se provee soporte para sistemas operativos linux (probado en Debia, Ubuntu y Manjaro), dado a que las APIs usadas fueron para estos sistemas, y no se hizo uso de la API de Windows, la cual es totalmente distinta.

Soporte nativo en el lenguaje y dependencias

- El utilitario en Python está apoyado en paquetes que pueden quedarse obsoletos o ser eliminados, lo cual supone un problema dado a que el lenguaje no ofrece de forma nativa acceso a esta información, y nos hace depender de la información que nos provean estas librerías.

- C al ser un lenguaje utilizado ampliamente para sistemas operativos, dispone de librerías estándar del lenguaje que nos permiten interactuar amplia y directamente con los sistemas disponibles, sin necesidad de preocuparnos de que estas queden obsoletas o sean eliminadas.

Facilidad de implementación

- La implementación en Python no tomó más de un día de trabajo y otro de pulido, siendo bastante sencillo.

- La implementación en C, tomó aproximadamente una semana y media dado a que fue necesario investigar la documentación disponible de las distintas APIs, examinar código fuente ajeno, esto debido a la baja accesibilidad del lenguaje y la casi nula información sobre el tema en la web.

Cabe destacar que la implementación en C pudo ser simplificada sustancialmente si haciamos uso de comandos ya disponibles en los sistemas operativos como netstat o ip, pero debido al interés en saber cómo funcionan realmente estas por detrás, decidimos no hacerlo e investigar las APIs mencionadas anteriormente.

Conclusiones

El alcance de las redes de computadoras es uno de los pilares que sostiene la tecnología tal y como la conocemos. Es de esos campos que pocas veces se toman en cuenta en el desarrollo de nuevas aplicaciones, pero que son las responsables de que las comunicaciones puedan existir tal y como la conocemos y es el artífice y constructor de la interconexión en el mundo moderno.

Con este laboratorio se elaboraron dos utilitarios de reconocimiento de redes en una computadora. Cada utilitario con sus propias características y virtudes. La intención de estos es que dependiendo de las características del sistema de destino se emplee uno o el otro. Y de este modo abarcar la mayor cantidad de máquinas y así permitir que esta sea una herramienta completa para el administrador o analista de redes.

Si el analista reconoce el estado de la red a la que está conectado puede ser capaz de determinar si su red se encuentra en riesgo o si está siendo vulnerada o si es vulnerable ante ataques maliciosos de terceros y luego actuar acorde a la situación y rápidamente asegurar la red. Y de este modo cubrir en lo posible estas debilidades y establecer una red ‘segura’ dentro de lo que las tecnologías actuales permiten.

Para concluir se le recuerda al lector que el desconocimiento de las vulnerabilidades de una red por parte de su dueño es el principal recurso que explotan los delincuentes para acceder a los sistemas y quebrar la integridad de los datos que son almacenados en ellos. Por lo que conocer el estado de su red y actuar en consecuencia del diagnóstico es una prevención invaluable.