Tarea 03

Javier Rivera Pizarro, **javier.riverapi@alumnos.uv.cl**

1. Introducción

En el presente informe se explicará el código realizado el cual es una herramienta en Python llamada "OUILookup". Esta herramienta permite consultar información sobre el fabricante de una tarjeta de red utilizando su dirección MAC o dirección IP. El propósito principal es facilitar la identificación del fabricante de dispositivos de red a través de consultas a una API REST pública.

2. Materiales y Métodos

La implementación de la herramienta "OUILookup" se llevó a cabo en una infraestructura computacional estándar. A continuación, se detallan los elementos clave:

Sistema Operativo:

- El desarrollo y ejecución del software se realizó en sistemas operativos compatibles con Python, tanto en entornos Linux como en Windows.

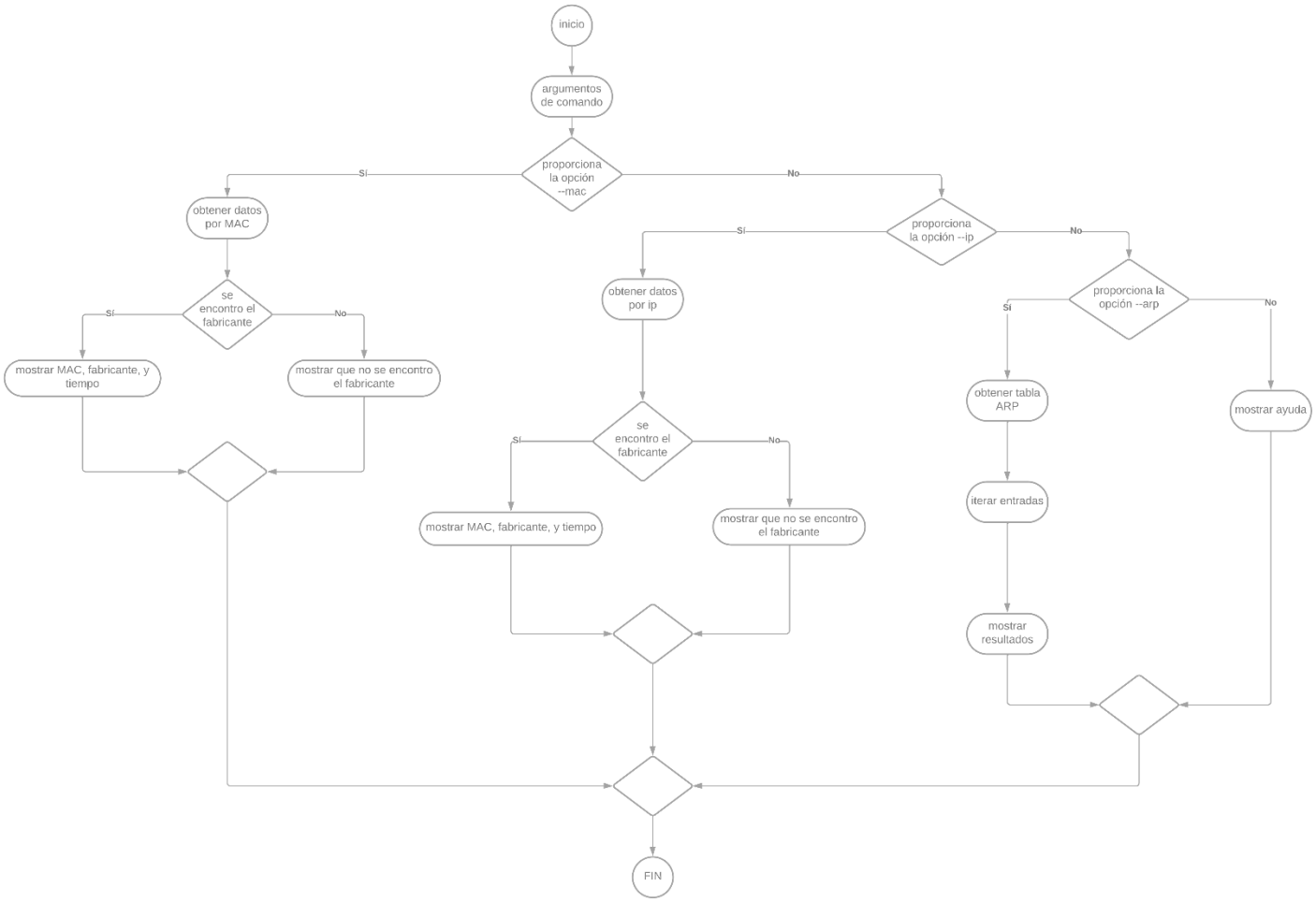
Lenguaje de Programación:

- El código de la herramienta se escribió en Python, versión 3.x.

Se utilizaron las siguientes bibliotecas y módulos de Python:

- argparse: Para gestionar los argumentos de línea de comandos.
- requests: Para realizar solicitudes HTTP a la API REST.
- re: Para realizar expresiones regulares en el procesamiento de la tabla ARP.
- os: Para ejecutar comandos del sistema operativo y obtener la salida de la tabla ARP.
- time: Para medir el tiempo de ejecución de las solicitudes.

El diseño lógico de "OUILookup" se visualiza a través de un diagrama de flujo general que traza la secuencia de operaciones desde la entrada de comandos hasta la presentación de resultados. Este diagrama proporciona una visión global del funcionamiento del programa, destacando las etapas clave del proceso.



Ahora voy a explicar cómo funciona cada parte del código:

1. Función `obtener_datos_por_ip(ip)`:

- Toma una dirección IP como parámetro.
- Construye la URI para consultar la API REST utilizando una dirección IP.
- Realiza la solicitud HTTP a la API y obtiene la respuesta y el tiempo de respuesta.
- Intenta obtener la dirección MAC desde la salida de ARP.
- Muestra la información obtenida.

2. Función `obtener_datos_por_mac(mac)`:

- Toma una dirección MAC como parámetro.
- Construye la URI para consultar la API REST utilizando una dirección MAC.
- Realiza la solicitud HTTP a la API y obtiene la respuesta y el tiempo de respuesta.
- Extrae el fabricante de la respuesta JSON.
- Retorna el fabricante y el tiempo de respuesta.

3. Función `realizar_solicitud(uri)`:

- Toma una URI como parámetro.
- Mide el tiempo de inicio.
- Realiza una solicitud GET a la URI utilizando la biblioteca `requests`.
- Mide el tiempo de finalización y calcula el tiempo transcurrido.
- Retorna la respuesta y el tiempo de respuesta.

4. Función `obtener_fabricante_desde_arp(arp_output)`:

- Toma la salida de la tabla ARP como parámetro.
- Divide la salida en líneas y busca una dirección MAC en la subred 192.168.1.
- Retorna la dirección MAC encontrada.

5. Función `main()':

- Utiliza `argparse` para gestionar los argumentos de línea de comandos.
- Según la opción proporcionada (--ip, --mac, --arp, --api), realiza la acción correspondiente.
- Muestra la información obtenida o la tabla ARP según la opción.
- Imprime la ayuda si no se proporcionan opciones válidas.

El programa en sí comienza ejecutando la función `main()` cuando se llama directamente (es decir, cuando se ejecuta como un script). Dentro de `main()`, se utiliza `argparse` para manejar los argumentos de línea de comandos y decidir qué acción realizar. Luego, se llaman a las funciones específicas según la opción proporcionada.

Por ejemplo, si ejecutas el programa con `python OUILookup.py --mac 98:06:3c:92:ff:c5`, llamará a `obtener_datos_por_mac("98:06:3c:92:ff:c5")`, que consultará la API REST y mostrará la información del fabricante.

3. Resultados

Usando el ejemplo explicado en la sección anterior si usamos el comando `python OUILookup.py --mac 98:06:3c:92:ff:c5`, obtendremos el siguiente resultado:

```
C:\Users\javie\OneDrive\Desktop\tarea3>python OUILookup.py --mac 98:06:3c:992:ff:c5
IP address: 98:06:3c:992:ff:c5
Fabricante: Samsung Electronics Co.,Ltd
Tiempo de respuesta: 530ms
```

4. Discusión y conclusiones

El programa "OUILookup" proporciona una interfaz de usuario clara y flexible, que permite consultas intuitivas tanto por dirección IP como por dirección MAC. Su integración con API REST externas demuestra la capacidad de interactuar eficazmente con recursos externos y recuperar información sobre los fabricantes de tarjetas de red.

La inclusión de métricas de latencia refleja una consideración consciente de los factores de rendimiento que contribuyen a mejorar la comunicación con la API.

La implementación aborda casos de error y sigue buenas prácticas de programación, como el uso de la biblioteca 'argparse' y el modularidad del código.

Además, la capacidad de ejecutarse en diferentes sistemas operativos e integrarse con herramientas del sistema, como el comando "arp", agrega flexibilidad al programa. "OUILookup" proporciona una herramienta funcional y bien estructurada que cumple con los requisitos especificados, pero siempre tiene espacio para mejoras y optimizaciones continuas.

5. Preguntas para responder:

a) **¿Qué es REST? ¿Qué es una API?**

REST es una interfaz para conectar varios sistemas basados en el protocolo HTTP, sirve para obtener y generar datos, devolviéndolos en formatos como XML y JSON.

Una API (“Interfaz de Programación de Aplicaciones”). Una aplicación usa un api para obtener información de otra aplicación, esta información puede ser datos o el software en sí. En otras palabras, una API es un simple enlace que conecta dos cosas, permitiendo que la información fluya de un software a otro, el cual transmite toda la comunicación de una aplicación a otra.

b) **¿Cómo se relaciona el protocolo HTTP con las API REST y cuál es su función en la comunicación entre clientes y servidores?**

En las palabras simples, HTTP proporciona un marco para la comunicación entre los clientes y servidor, y las API REST definen como se estructuran y maneja los recursos en esa comunicación utilizando los métodos y códigos de estado de http.

Ya que REST es la transferencia de representaciones de recursos entre el cliente y el servidor. Esto implica que el servidor envía representaciones de recursos al cliente en respuesta a las solicitudes HTTP (como GET, POST, PUT y DELETE). Estas representaciones pueden estar en formatos como JSON o XML.

c) **¿Qué papel juega la dirección IP en el acceso a recursos a través de una API REST?**

En el contexto de una API, la dirección IP se utiliza para identificar la ubicación de la solicitud entrante. En el proceso de autenticación y autorización, la dirección IP puede ser utilizada para aplicar políticas de seguridad y controlar el acceso a los recursos.

d) **¿Por qué es importante considerar la latencia de red y el ancho de banda? ¿Cómo afectan estos factores al rendimiento de la API?**

La latencia de red se refiere al tiempo que tarda una solicitud en viajar desde el cliente hasta el servidor y viceversa. El ancho de banda se refiere a la cantidad de datos que se pueden transmitir en un período de tiempo. Ambos son críticos para el rendimiento de una API, ya que una alta latencia o un ancho de banda limitado pueden provocar tiempos de respuesta lentos y una experiencia deficiente para los usuarios.

e) ¿Por qué el programa desarrollado utilizando API REST es más lenta su ejecución?

La ejecución de un programa que utiliza una API REST puede ser más lenta debido a factores como la latencia de red, el tiempo de procesamiento en el servidor y la cantidad de datos transferidos. Además, el rendimiento puede estar influenciado por la eficiencia de la implementación de la API y la infraestructura subyacente

f) ¿Cuál es la diferencia entre la dirección MAC (Media Access Control) y la dirección IP, y en qué capa de la red se utilizan cada una de ellas?

La dirección MAC (Media Access Control) es una identificación única asociada a la tarjeta de red de un dispositivo. Se utiliza en la capa de enlace de datos para la comunicación dentro de una red local. La dirección IP es un identificador asignado a nivel de red y se utiliza en la capa de red para la comunicación entre redes. Mientras que la dirección MAC es específica de la interfaz de red, la dirección IP identifica el dispositivo en la red.

g) ¿Cómo pueden las redes LAN (Local Area Networks) y WAN (Wide Area Networks) afectar la accesibilidad y la velocidad de respuesta de una API REST?

En una LAN (Local Area Network), la accesibilidad y velocidad suelen ser altas debido a la proximidad física de los dispositivos en la red. En una WAN (Wide Area Network), la accesibilidad y velocidad pueden verse afectadas por la distancia geográfica, la congestión de la red y la calidad de la conexión, lo que podría resultar en tiempos de respuesta más lentos.

h) ¿Qué es un enrutador y cómo se utiliza para dirigir el tráfico de datos? ¿Qué relación tiene esto con el enrutamiento de solicitudes en una API REST?

Un enrutador es un dispositivo de red que dirige el tráfico de datos entre redes. En el contexto de una API REST, el enrutamiento de solicitudes se refiere a la asignación de solicitudes HTTP a las funciones y recursos correspondientes en el servidor. El enrutador juega un papel importante en dirigir las solicitudes entrantes a los controladores o recursos adecuados en la aplicación.

i) ¿Cómo se asocian los puertos de red con servicios y aplicaciones específicas?

Los puertos de red son números de identificación asociados con servicios y aplicaciones específicas en un dispositivo. Cuando un dispositivo recibe datos, el número de puerto se utiliza para determinar a qué servicio o aplicación se deben enviar esos datos. Por ejemplo, el puerto 80 se asocia comúnmente con HTTP, mientras que el puerto 443 se asocia con HTTPS. En una API REST, el uso de puertos específicos puede ser parte de la convención o configuración del servidor.