Título del informe

Nombre Apellido1 Apellido 2, [email@tld.com](mailto:email@tld.com) (Un integrante por línea)

# Introducción

Se ha desarrollado la segunda tarea, cuyo objetivo principal es implementar una herramienta de línea de comandos en Python, llamada **OUILookup**, que permite consultar el fabricante de una tarjeta de red a partir de su dirección MAC. Para ello, se utiliza una API REST pública que proporciona información sobre los fabricantes asociados a direcciones MAC específicas.

El proyecto busca reforzar conceptos relacionados con las redes de computadores, tales como las direcciones MAC, la tabla ARP y la programación de scripts para automatizar procesos de consulta. Asimismo, permite al estudiante aplicar prácticas de programación funcional en Python y gestionar argumentos de línea de comandos, lo cual es fundamental para el desarrollo de herramientas de red.

A lo largo de este informe, se describen los objetivos de la tarea, el trabajo realizado para implementar la herramienta, ejemplos de uso, los casos de prueba, y una explicación sobre el funcionamiento de las direcciones MAC aleatorias. Además, se incluye un diagrama de flujo que ilustra el funcionamiento del programa y se discuten las conclusiones obtenidas durante el desarrollo de la tarea.

# Descripción del problema y diseño de la solución

El problema planteado en la tarea consiste en desarrollar una herramienta de línea de comandos en Python, llamada OUILookup, que permita consultar el fabricante de una tarjeta de red a partir de su dirección MAC. La herramienta debe ser capaz de utilizar una API REST pública para obtener la información del fabricante correspondiente. Además, debe proporcionar la opción de consultar los fabricantes de los dispositivos conectados a la red local a través de la tabla ARP.

El programa debe cumplir con los siguientes requisitos:

Implementar la funcionalidad para manejar los parámetros de línea de comandos: --mac, --arp y --help.

Utilizar una API REST para consultar el fabricante de una dirección MAC.

Poder mostrar la lista de dispositivos conectados con sus fabricantes a partir de la información de la tabla ARP.

Implementar el programa siguiendo el paradigma de la programación funcional y manejar posibles errores.

Diseño de la solución

Para resolver el problema, se ha diseñado la herramienta OUILookup, la cual se estructura en las siguientes partes:

Procesamiento de parámetros de línea de comandos: Se utiliza la biblioteca getopt para gestionar los argumentos que se pasan al programa. El usuario puede proporcionar tres opciones:

--mac: Consulta el fabricante de una dirección MAC específica.

--arp: Muestra los fabricantes de los dispositivos detectados en la tabla ARP de la red local.

--help: Muestra un mensaje de ayuda que indica cómo usar la herramienta.

Consulta de la API REST: Para obtener la información del fabricante, se utiliza la API pública https://api.maclookup.app/v2/macs/. La función buscar\_mac() envía una solicitud HTTP a la API, y si la dirección MAC se encuentra en la base de datos, devuelve el nombre del fabricante asociado. En caso contrario, informa que el fabricante no se ha encontrado.

Lectura de la tabla ARP: La función leer\_tabla\_arp() utiliza el comando del sistema operativo para consultar la tabla ARP (arp -a en macOS) y extraer las direcciones MAC de los dispositivos conectados a la red. Posteriormente, para cada dirección MAC, se consulta la API para identificar el fabricante.

Gestión de errores y manejo de excepciones: El programa está diseñado para manejar errores comunes, como la falta de conectividad a la API o la ausencia de permisos para ejecutar el comando ARP. Se muestran mensajes informativos en caso de error, lo cual mejora la experiencia del usuario.

Programación funcional: La solución se ha implementado siguiendo el paradigma de programación funcional, dividiendo el código en funciones específicas para cada tarea (procesar argumentos, consultar la API, leer la tabla ARP, etc.), lo que mejora la modularidad y la mantenibilidad del código.

# Implementación

La implementación de la herramienta OUILookup se ha realizado en Python, siguiendo un enfoque modular y funcional para mejorar la claridad y mantenibilidad del código. A continuación, se detallan los aspectos más importantes de la implementación:

3.1. Estructura del código

El código fuente de OUILookup se organiza en tres funciones principales, además de la función de entrada main:

buscar\_mac(mac\_address): Consulta la API pública para obtener el fabricante asociado a la dirección MAC proporcionada.

leer\_tabla\_arp(): Ejecuta el comando del sistema operativo para leer la tabla ARP y extraer las direcciones MAC de los dispositivos conectados a la red.

main(argv): Procesa los argumentos de línea de comandos y dirige la ejecución del programa según las opciones proporcionadas.

3.2. Gestión de argumentos de línea de comandos

Para manejar los parámetros de entrada, se utiliza la biblioteca getopt, que permite capturar y procesar las opciones --mac, --arp, y --help. Los casos son gestionados de la siguiente manera:

Opción --help: Se muestra un mensaje de ayuda con la sintaxis de uso del programa y se finaliza la ejecución.

Opción --mac: Si se especifica una dirección MAC, se llama a la función buscar\_mac(mac\_address) para realizar la consulta a la API y obtener el fabricante.

Opción --arp: Si se pasa el parámetro --arp, se ejecuta la función leer\_tabla\_arp() para listar los dispositivos en la red local junto con sus fabricantes.

3.3. Consulta a la API REST

Para realizar la consulta a la API, se utiliza la biblioteca requests, que permite enviar una solicitud HTTP a la URL https://api.maclookup.app/v2/macs/. La función buscar\_mac(mac\_address) compone la URL de la solicitud con la dirección MAC proporcionada y realiza la consulta. Si la respuesta contiene el nombre del fabricante, se muestra en pantalla; de lo contrario, se informa que el fabricante no se encontró. En caso de error, se captura la excepción y se muestra un mensaje indicando que hubo un problema al consultar la API.

3.4. Lectura de la tabla ARP

La función leer\_tabla\_arp() ejecuta el comando arp -a en macOS para obtener la tabla ARP. La salida del comando se procesa para extraer las direcciones IP y MAC, que luego son utilizadas para consultar el fabricante mediante la función buscar\_mac(). La herramienta muestra el fabricante de cada dispositivo conectado a la red local.

3.5. Manejo de errores y excepciones

El código implementa una gestión de errores para situaciones comunes, como:

Fallos en la conexión a la API o respuestas inválidas.

Errores al ejecutar el comando ARP debido a la falta de permisos.

Errores de sintaxis en los argumentos proporcionados por el usuario.

En todos estos casos, el programa informa al usuario con un mensaje de error específico, lo que mejora la robustez y la usabilidad de la herramienta.

3.6. Programación funcional

La implementación sigue el paradigma de programación funcional, dividiendo las tareas en funciones específicas y evitando efectos secundarios en la medida de lo posible. Esto permite que cada función sea responsable de una tarea específica, facilitando el mantenimiento del código y futuras modificaciones.

3.7. Consideraciones para la ejecución en diferentes sistemas operativos

Aunque el código está diseñado para ejecutarse en macOS, se puede adaptar fácilmente para otros sistemas operativos cambiando el comando utilizado para leer la tabla ARP. Por ejemplo, en Linux podría utilizarse arp -n y en Windows arp -a, ajustando el procesamiento de la salida en consecuencia.

# Pruebas

Explicar cómo se realizaron las pruebas del código. Se deben incluir ejemplo de los casos de pruebas, sus resultados y cómo se aseguró el funcionamiento correcto del código.

# Discusión y conclusiones

Debe presentar un resumen de los resultados obtenidos. Hacer una reflexión sobre lo que se aprendió y lo que no se logró. Finalmente establecer posibles mejoras a la implementación realizada.

# Referencias

[1] Author 1, A.B.; Author 2, C.D. Title of the article. Abbreviated Journal Name Year, Volume, page range.

[2] Author 1, A.; Author 2, B. Title of the chapter. In *Book Title*, 2nd ed.; Editor 1, A., Editor 2, B., Eds.; Publisher: Publisher Location, Country, 2007; Volume 3, pp. 154–196.

[3] Author 1, A.; Author 2, B. *Book Title*, 3rd ed.; Publisher: Publisher Location, Country, 2008; pp. 154–196.

# Ejemplo de sección

Esta sección muestra ejemplos de uso de secciones, subsecciones, figuras, tablas y ecuaciones.

## Subsección

Esta es una subsección.

## Subsección

Esta es otra subsección.

### Subsubsección

Esta es una subsubsección. Listas con viñetas deben ser de la siguiente manera:

* Primer elemento;
* Segundo elemento;
* Tercer elemento.

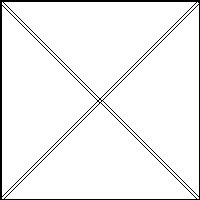
En forma similar, las lista numeradas, tiene el siguiente formato:

1. Primer elemento;
2. Segundo elemento;
3. Tercer elemento.

El texto continúa como párrafo aparte.

## Figuras y Tablas

Todas las figuras y tablas deben estar referenciadas en el texto principal como **Figura 1**, **Tabla 1**, etc. Toda figura o tabla debe tener un título explicativo. En las figuras, debe ir abajo (ver **Figura 1**). En las tablas, arriba (ver **Tabla 1**). Para evitar colocar numeración manual, utilice “insertar título” del menú Word e “insertar referencia cruzada”.



**Figura 1.** Esta es una figura. El título debe estar centrado

**Tabla 1.** Esta es una tabla. Las tablas deben colocarse cerca de la primera vez que se citan en el texto principal**.** El título debe estar centrado

|  |  |  |
| --- | --- | --- |
| Title 1 | Title 2 | Title 3 |
| entry 1 | data | data |
| entry 2 | data | data 1 |

1 Tablas pueden tener un pie de tabla para observaciones, aclaraciones, etc.

El texto continúa como párrafo aparte. En la **Figura 2** se muestra una figura que está dividida en dos paneles.

|  |  |
| --- | --- |
| C:\Users\martin\Downloads\testFigure.tif | C:\Users\martin\Downloads\testFigure.tif |
| (**a**) | (**b**) |

**Figura 2.** Esta es una figura dividida en dos sub-figuras o paneles. Cada sub-figura debe enumerarse como: (**a**) Descripción de lo que está contenido en el primer panel; (**b**) Descripción de lo contenido en el segundo panel. Las figuras deben colocarse cerca de la primera vez que se citan en el texto principal. El título debe estar centrado.

## Ecuaciones

La Ec. 1 muestra un ejemplo de ecuación:

|  |  |
| --- | --- |
| a = 1, | (1) |

el texto que sigue a una ecuación no necesariamente debe ser un nuevo párrafo cuando se sigue explicando la ecuación y sus componentes.

El texto del documento sigue en párrafo aparte.