Tarea 2: Consultar Fabricantes de tarjetas de red a través de una API

Diego Retamales Castillo, diego.retamales@alumnos.uv.cl

Maximiliano Casas Echeverria, maximiliano.casas@alumnos.uv.cl

Carlos Gaete Concha, carlos.gaete@alumnos.uv.cl

1. Introducción.

La conectividad y la comunicación entre dispositivos son esenciales en las redes de computadoras. Cada dispositivo se identifica mediante una dirección MAC (Media Access Control), la cual facilita la comunicación, además de también revelar información sobre el fabricante del hardware. En este trabajo se propone desarrollar una herramienta llamada OUILookup, que permitirá a los usuarios consultar el fabricante de una tarjeta de red a partir de su dirección MAC mediante una API REST pública.

2. Descripción del problema y diseño de la solución

La necesidad de una solución sencilla y accesible que permita a los usuarios consultar el fabricante de un dispositivo de red utilizando su dirección MAC. Esto no solo facilitaría la administración de redes, sino que también ayudaría en tareas de auditoría y resolución de problemas. La solución debe ser compatible con diferentes sistemas operativos, proporcionar resultados rápidos y manejar errores de manera eficiente.

La solución propuesta es un algoritmo llamado OUILookup, desarrollado en Python. Esta herramienta permitirá a los usuarios consultar el fabricante de un dispositivo de red a partir de su dirección MAC utilizando una API REST pública. A continuación, se detallan los componentes clave del diseño

3. Implementación

La implementación de OUILookup se basa en funciones así simplificando su estructura

```python
def obtener_fabricante(mac):
    url = f"https://api.maclookup.app/v2/macs/{mac}"  # URL de la API
    tiempo_inicio = time.time()  # Medir el tiempo de respuesta
    try:
        respuesta = requests.get(url)  # Hacer la solicitud a la API
        tiempo_fin = time.time()  # Finalizar la medición de tiempo

        if respuesta.status_code == 200:  # Verificar si la solicitud fue exitosa
            datos = respuesta.json()  # Convertir la respuesta a JSON
            empresa = datos.get('company')  # Obtener el nombre de la empresa
            if not empresa or empresa.lower() == 'not found':
                empresa = "Not Found"  # Manejo de error si no se encuentra la empresa
        else:
            empresa = "Not Found"  # Manejo de error para otros códigos de estado
    except:
        empresa = "Not Found"  # Manejo de excepción
        tiempo_fin = time.time()  # Finalizar la medición de tiempo

    return empresa, int((tiempo_fin - tiempo_inicio) * 1000)  # Retornar la empresa y el tiempo de respuesta
```

Figura 1

Esta función recibe una dirección MAC como argumento y realiza una consulta a la API REST para obtener el fabricante asociado.

```python
def obtener_tabla_arp():
    if sys.platform.startswith('win'):  # Verificar si el sistema es Windows
        try:
            salida_arp = subprocess.check_output(['arp', '-a']).decode('latin-1')  # Ejecutar el comando ARP
            lineas = salida_arp.splitlines()  # Dividir la salida en líneas
            entradas = []  # Lista para almacenar las direcciones MAC
            for linea in lineas:  # Iterar sobre cada línea
                partes = linea.split()  # Dividir la línea en partes
                # Validar que la línea contenga una dirección IP y una MAC
                if len(partes) >= 3 and re.match(r'([0-9]{1,3}\.){3}[0-9]{1,3}', partes[0]) and re.match(r'([0-9A-Fa-f]{2}[:-]){5}([0-9A-Fa-f]{2})', partes[1]):
                    mac = partes[1]  # Obtener la dirección MAC
                    entradas.append(mac)  # Agregar a la lista de entradas
            return entradas  # Retornar la lista de direcciones MAC
        except FileNotFoundError:
            print("Error: Comando 'arp' no encontrado en Windows.")  # Manejo de error si el comando no se encuentra
            return []
    else:
        print("Error: Esta funcionalidad solo está disponible en Windows.")  # Mensaje de error para otros sistemas
        return []
```

Figura 2

Esta función obtiene las entradas de la tabla ARP del sistema operativo. En el caso de Windows, utiliza el comando arp -a para recuperar la información

```python
def imprimir_uso():
    print("Use: python OUILookup.py --mac <mac> | --arp | [--help]")
    print("--mac: MAC a consultar. P.e. aa:bb:cc:00:00:00.")
    print("--arp: muestra los fabricantes de los host disponibles en la tabla arp.")
    print("--help: muestra este mensaje y termina.")
```

Figura 3

Esta función muestra al usuario cómo utilizar la herramienta, proporcionando ejemplos de comandos y opciones disponibles.

```python
def normalizar_mac(mac):
    mac_limpia = re.sub(r'[.:-]', '', mac.lower())  # Eliminar caracteres no alfanuméricos
    if len(mac_limpia) == 6:  # Verificar longitud
        mac_limpia = mac_limpia * 2  # Duplicar si es necesario
    return ':'.join([mac_limpia[i:i+2] for i in range(0, 12, 2)])  # Formatear la MAC
```

Figura 4

Esta función normaliza la dirección MAC eliminando caracteres no alfanuméricos y asegurándose de que tenga el formato correcto.

```python
def imprimir_resultado_mac(mac, fabricante, tiempo_respuesta):
    print(f"MAC address : {mac}")  # Imprimir dirección MAC
    print(f"Fabricante : {fabricante}")  # Imprimir fabricante
    print(f"Tiempo de respuesta: {tiempo_respuesta}ms")  # Imprimir tiempo de respuesta
```

Figura 5

Esta función se encarga de mostrar el resultado de la consulta al usuario, incluyendo la dirección MAC, el fabricante y el tiempo de respuesta de la consulta.

```python
def main(argv):
    try:
        opts, args = getopt.getopt(argv, "hm:a", ["help", "mac=", "arp"])  # Procesar argumentos de línea de comandos
    except getopt.GetoptError:
        imprimir_uso()  # Imprimir uso si hay error
        sys.exit(2)

    for opt, arg in opts:  # Iterar sobre las opciones
        if opt in ("-h", "--help"):
            imprimir_uso()  # Mostrar el uso
            sys.exit()
        elif opt in ("-m", "--mac"):
            mac_original = arg  # Obtener la MAC original
            mac_normalizada = normalizar_mac(arg)  # Normalizar la MAC
            fabricante, tiempo_respuesta = obtener_fabricante(mac_normalizada)  # Obtener fabricante
            imprimir_resultado_mac(mac_original, fabricante, tiempo_respuesta)  # Imprimir resultado
        elif opt in ("-a", "--arp"):
            entradas_arp = obtener_tabla_arp()  # Obtener entradas ARP
            if entradas_arp:
                print("MAC/Vendor:")  # Imprimir encabezado
                for mac in entradas_arp:  # Iterar sobre entradas ARP
                    fabricante, _ = obtener_fabricante(mac)  # Obtener fabricante
                    print(f"{mac} / {fabricante}")  # Imprimir MAC y fabricante
            else:
                print("No se encontraron entradas ARP o no se pudo recuperar la tabla ARP.")  # Mensaje si no hay entradas
```

Figura 6

La función principal de la aplicación que procesa los argumentos de línea de comandos.

4. Pruebas

Las pruebas de código se realizaron de la siguiente manera.

```
PS C:\Users\noche> & C:/msys64/ucrt64/bin/python.exe c:/Users/noche/Downloads/tarea02-OUILookup.py --help
Use: python OUILookup.py --mac <mac> | --arp | [--help]
--mac: MAC a consultar. P.e. aa:bb:cc:00:00:00.
--arp: muestra los fabricantes de los host disponibles en la tabla arp.
--help: muestra este mensaje y termina.
```

Figura 7

En la figura 7 se muestran los comandos usados para la consulta de:

--mac: la consulta de la mac

--arp: la consulta de la tabla arp

--help: muestra los comandos y que hacen.

```
PS C:\Users\noche> & C:/msys64/ucrt64/bin/python.exe c:/Users/noche/Downloads/tarea02-OUILookup.py --mac  48-E7-DA
MAC address : 48-E7-DA
Fabricante : AzureWave Technology Inc.
Tiempo de respuesta: 572ms
```

Figura 8

En la figura 8 se muestra una consulta de mac, aquí se aprecia la mac del switch, quien la fabricó y el tiempo de respuesta.

```
PS C:\Users\noche> & C:/msys64/ucrt64/bin/python.exe c:/Users/noche/Downloads/tarea02-OUILookup.py --mac  9c:a5:13
MAC address : 9c:a5:13
Fabricante : Samsung Electronics Co.,Ltd
Tiempo de respuesta: 568ms
```

Figura 9

Otro ejemplo de consulta mac, mostrando la dirección mac del switch, su fabricante y el tiempo de respuesta.

```
PS C:\Users\noche> & C:/msys64/ucrt64/bin/python.exe c:/Users/noche/Downloads/tarea02-OUILookup.py --mac 98:06:3c:92:ff:c5
MAC address : 98:06:3c:92:ff:c5
Fabricante : Samsung Electronics Co.,Ltd
Tiempo de respuesta: 727ms
```

Figura 10

Ultimo ejemplo de consulta mac, en la figura 10 se muestra la dirección mac del switch, el fabricante y su tiempo de respuesta.

```
PS C:\Users\noche> & C:/msys64/ucrt64/bin/python.exe c:/Users/noche/Downloads/tarea02-OUILookup.py --arp
MAC/Vendor:
e0-00-84-e7-52-0b / HUAWEI TECHNOLOGIES CO.,LTD
84-a0-6e-19-19-70 / Sagemcom Broadband SAS
fc-d5-d9-cc-0c-d3 / Shenzhen SDMC Technology CO.,Ltd.
fc-d5-d9-cc-08-53 / Shenzhen SDMC Technology CO.,Ltd.
08-e9-f6-4b-48-e2 / AMPAK Technology,Inc.
7c-66-ef-f2-0a-d0 / Hon Hai Precision IND.CO.,LTD
08-e9-f6-4b-f2-88 / AMPAK Technology,Inc.
f4-20-15-08-71-9f / Guangzhou Shiyuan Electronic Technology Company Limited
ff-ff-ff-ff-ff-ff / Not Found
01-00-5e-00-00-16 / Not Found
01-00-5e-00-00-fb / Not Found
01-00-5e-00-00-fc / Not Found
01-00-5e-7f-ff-fa / Not Found
ff-ff-ff-ff-ff-ff / Not Found
ff-ff-ff-ff-ff-ff / Not Found
01-00-5e-00-00-16 / Not Found
01-00-5e-00-00-fb / Not Found
01-00-5e-7f-ff-fa / Not Found
```

Figura 11

En la figura 11 se aprecia la tabla ARP, en esta están contenidos los datos para cada mac en arp, en caso de no existir el programa imprime un mensaje de 'Not Found', haciendo alusión a que no es que exista la mac, sino que simplemente no se ha encontrado

4.1. Funcionamiento de las Direcciones MAC Aleatorias

Las direcciones MAC aleatorias son una técnica utilizada en dispositivos electrónicos para mejorar la privacidad del usuario al conectarse a redes Wi-Fi. Esta técnica consiste en generar una dirección MAC diferente cada vez que un dispositivo se conecta a una nueva red, lo que dificulta el rastreo de la actividad del usuario a través de diferentes redes.

1. Privacidad Mejorada: Utilizando direcciones MAC aleatorias, se reduce la capacidad de terceros para seguir la actividad de un dispositivo a lo largo del tiempo y entre diferentes redes.[3]
2. Desactivación de la Aleatorización: Los usuarios tienen la opción de desactivar la aleatorización de la dirección MAC en sus dispositivos. En entornos corporativos, esta configuración puede ser gestionada de manera remota mediante soluciones de Mobile Device Management (MDM).[1]

5. Discusión y conclusiones

En esta tarea, se desarrolló la herramienta **OUILookup**, que permite consultar el fabricante de dispositivos de red a partir de sus direcciones MAC. El objetivo principal fue crear una solución sencilla y accesible que facilite la identificación de los dispositivos en una red.

6. Resultados Obtenidos

Los resultados obtenidos fueron satisfactorios. Permiten a los usuarios realizar consultas rápidas y obtener información sobre los fabricantes. Las pruebas ejecutadas confirmaron

que **OUILookup** puede manejar tanto consultas directas de direcciones MAC como la recuperación de datos de la tabla ARP.

7. Referencias

[1] https://www.applivery.com/es/blog/actualizaciones-de-producto/direcciones-mac-aleatorias-en-dispositivos-apple/
[2] https://www.spectrum.net/es/support/internet/randomized-mac-android-devices
[3] https://telefonicatech.com/blog/mac-aleaotorias-y-privacidad-parte2#:~:text=Una%20característica%20que%20ofrece%20Windows,día%20o%20utilizar%20la%20real.