Título del informe

Rodrigo Guerrero Barros, rodrigo.guerrero@alumnos.uv.cl

Yerko Yuivar Concha, yerko.yuivar@alumnos.uv.cl

Bastian Puebla Gallardo, bastian.puebla@alumnos.uv.cl

1. Introducción

Este proyecto busca crear una herramienta llamada "OUILookup" en Python. La herramienta permitirá a los usuarios averiguar quién fabricó una tarjeta de red usando su dirección MAC o IP. Para esto, utilizaremos una API en línea, como la de https://maclookup.app, que nos dará información sobre los fabricantes asociados a las direcciones proporcionadas.

2. Materiales y Métodos

argparse: Utilizada para procesar argumentos de línea de comandos de una manera fácil y eficiente. requests: Utilizada para realizar solicitudes HTTP y obtener información de un servicio web. subprocess: Utilizada para ejecutar comandos del sistema operativo desde el script. getopt: Otro módulo para procesar argumentos de línea de comandos, utilizado como alternativa a argparse en algunas partes del script. Funciones definidas:

Librerías utilizadas:

argparse: Utilizada para procesar argumentos de línea de comandos de una manera fácil y eficiente.

requests: Utilizada para realizar solicitudes HTTP y obtener información de un servicio web.

subprocess: Utilizada para ejecutar comandos del sistema operativo desde el script.

getopt: Otro módulo para procesar argumentos de línea de comandos, utilizado como alternativa a argparse en algunas partes del script.

Funciones definidas:

obtener_mac_address(ip): Utiliza el comando arp para obtener la dirección MAC asociada con una dirección IP dada.

obtener_fabricante(mac): Realiza una solicitud a un servicio web que devuelve el fabricante asociado con una dirección MAC dada.

obtener_tabla_arp(url): Utiliza el comando arp -a para obtener la tabla ARP de la máquina local. También intenta hacer una solicitud HTTP con la tabla ARP a una URL específica.

parse_args_argparse(): Utiliza la librería argparse para analizar y procesar los argumentos de línea de comandos.

parse_args_getopt(argv): Utiliza la librería getopt para analizar y procesar los argumentos de línea de comandos en una forma alternativa.

main(): La función principal que coordina la ejecución del script, determina qué función llamar en función de los argumentos proporcionados.

3. los Resultados

Los resultados se imprimen en la consola, incluyendo la dirección MAC, el fabricante y la tabla ARP, según las opciones proporcionadas.

4. Discusión y conclusiones

Tras comprobar las distintas conexiones se dedujo que el fabricante encontrado y su MS varia según la persona, como Rodrigo tiene 697MS y luego Yerko tiene 423. Se nota un impacto en la latencia.

4.1. Figuras y Tablas

```
Fabricante: ('Samsung Electronics Co..Ltd', '49ms')

PS C:\Users\gogyt\onedrive\documentos\A UNIVERSIDAD> python OUILookup.py --mac  00:01:97:bb:bb:bb

MAC address: 00:01:97:bb:bb:bb
Fabricante: ('Cisco                 # CISCO SYSTEMS, INC.', '697ms')

PS C:\Users\gogyt\onedrive\documentos\A UNIVERSIDAD> python OUILookup.py --mac  00:01:97:bb:bb:bb -l

MAC address: 00:01:97:bb:bb:bb
Fabricante: ('Cisco Systems. Inc', '42ms')

PS C:\Users\gogyt\onedrive\documentos\A UNIVERSIDAD> python OUILookup.py --ip 192.168.0.1

MAC address: f8:d2:ac:fd:35:df
Fabricante: ('Not found', '418ms')

PS C:\Users\gogyt\onedrive\documentos\A UNIVERSIDAD> python OUILookup.py --ip 192.168.0.1 -l

MAC address: f8:d2:ac:fd:35:df
Fabricante: ('Vantiva USA LLC', '50ms')
```

Figura 1. En esta figura se puede observar el fabricante, la Mac y los ms de Rodrigo Guerrero

```
C:\Users\DIEGO\OneDrive\Escritorio\xd>python OUILookup.py --mac 00:01:97:bb:bb:bb

MAC address: 00:01:97:bb:bb:bb
Fabricante: ('Cisco                  # CISCO SYSTEMS, INC.', '423ms')


C:\Users\DIEGO\OneDrive\Escritorio\xd>python OUILookup.py --mac 00:01:97:bb:bb:bb -l

MAC address: 00:01:97:bb:bb:bb
Fabricante: ('Cisco Systems. Inc', '42ms')


C:\Users\DIEGO\OneDrive\Escritorio\xd>python OUILookup.py --ip 192.168.1.1

MAC address: 58:ae:f1:52:ca:30
Fabricante: ('Not found', '418ms')


C:\Users\DIEGO\OneDrive\Escritorio\xd>python OUILookup.py --ip 192.168.1.1 -l

MAC address: 58:ae:f1:52:ca:30
Fabricante: ('Fiberhome Telecommunication Technologies Co..LTD', '52ms')
```

Figura 2. En esta figura se puede observar el fabricante, la Mac y los ms de Yerko Yuivar

```
C:\Users\basti\OneDrive\Escritorio\pgroas>python OUILookup.py --mac 00:01:97:bb:bb:bb

MAC address: 00:01:97:bb:bb:bb
Fabricante: ('Cisco                  # CISCO SYSTEMS, INC.', '96ms')


C:\Users\basti\OneDrive\Escritorio\pgroas>python OUILookup.py --mac 00:01:97:bb:bb:bb -l

MAC address: 00:01:97:bb:bb:bb
Fabricante: ('Cisco Systems. Inc', '40ms')


C:\Users\basti\OneDrive\Escritorio\pgroas>python OUILookup.py --ip 192.168.100.1

MAC address: b0:ec:dd:71:a8:0f
Fabricante: ('Not found', '97ms')


C:\Users\basti\OneDrive\Escritorio\pgroas>python OUILookup.py --ip 192.168.100.1 -l

MAC address: b0:ec:dd:71:a8:0f
Fabricante: ('HUAWEI TECHNOLOGIES CO..LTD', '49ms')
```

Figura 3. En esta figura se puede observar el fabricante, la Mac y los ms de Bastian Puebla

Preguntas y respuesta

A) REST = (transferencia de estado representacional) es una interfaz destinada a conectar varios sistemas basados en el protocolo HTTP.

API = (interfaz de programa de aplicación) es la que define las reglas y protocolos a seguir para permitir la comunicación con otros sistemas de software.

B) HTTP y API REST**:** HTTP (Protocolo de Transferencia de Hipertexto) se utiliza en las API REST para llevar a cabo operaciones sobre recursos, utilizando métodos estándar como GET para obtener datos y POST para enviar datos.

C) Dirección IP**:** En el contexto de una API REST, la dirección IP identifica de manera única un dispositivo en una red y se utiliza para acceder a recursos específicos en servidores.

D) Latencia y Ancho de Banda: La latencia, que es el tiempo de ida y vuelta en la comunicación, y el ancho de banda, que influye en la cantidad de datos transferidos simultáneamente, son factores críticos para la velocidad de respuesta de una API REST.

E) Rendimiento del programa con API REST: La ejecución lenta de un programa desarrollado con API REST puede atribuirse a diversos factores, como la latencia de red, la eficiencia en la implementación y la gestión de recursos en el servidor y cliente.

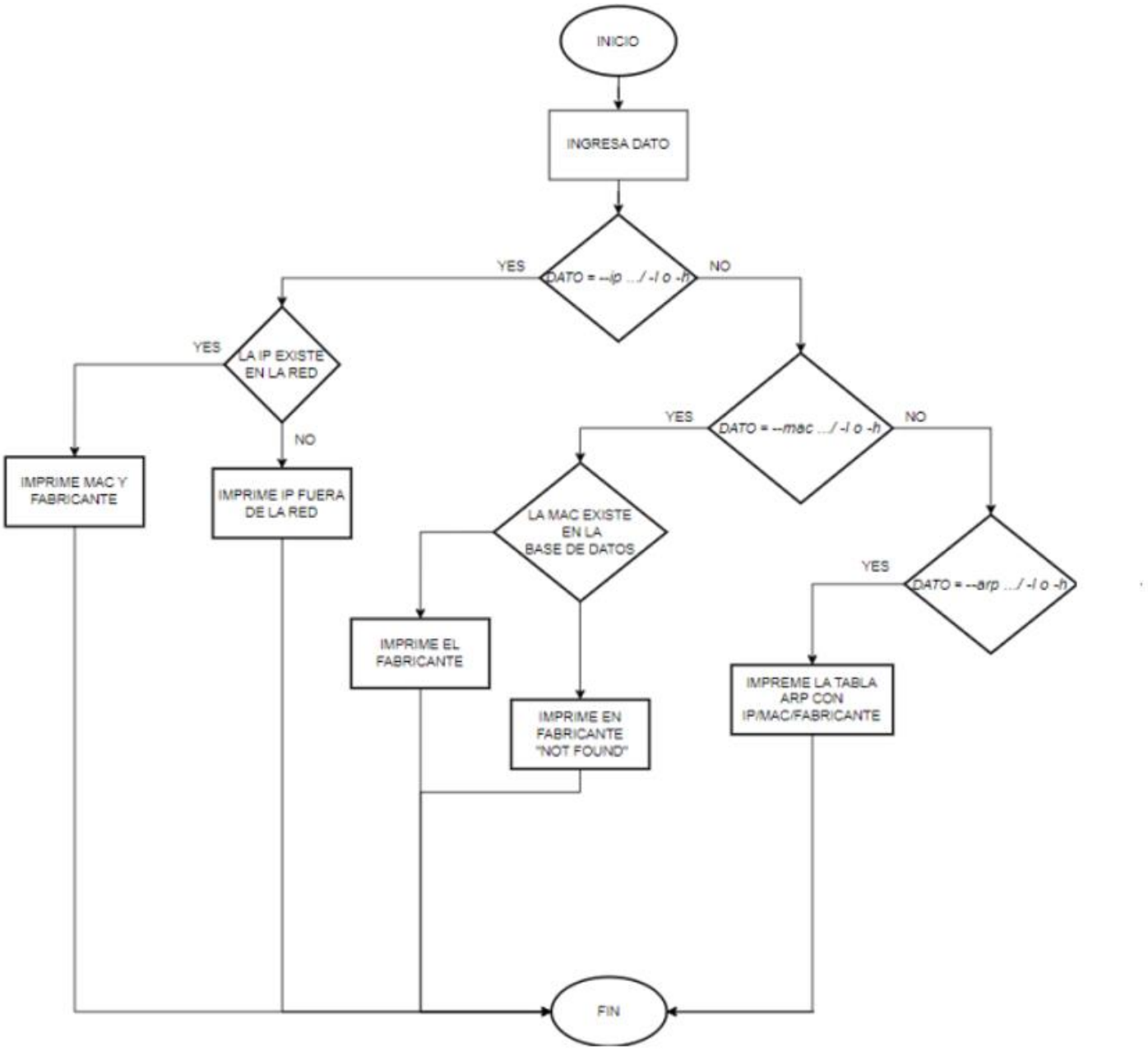
F) Diferencia entre dirección MAC y dirección IP: La dirección MAC (Media Access Control) es única para cada interfaz de red y opera en la capa de enlace de datos, mientras que la dirección IP identifica dispositivos en la red y opera en la capa de red.

G) Impacto de LAN y WAN: Las redes LAN (Local Area Networks) suelen ser más rápidas y accesibles que las WAN (Wide Area Networks) debido a la distancia y el número de intermediarios en una WAN, lo que afecta la accesibilidad y velocidad de respuesta de una API REST.

H) Enrutador y enrutamiento de solicitudes en API REST: Un enrutador dirige el tráfico de datos en una red; en el contexto de una API REST, el enrutamiento de solicitudes implica asociar rutas URL con funciones específicas del servidor, facilitando así la dirección adecuada de las solicitudes.

I)Asociación de puertos con servicios y aplicaciones: En el ámbito de las redes, los puertos se asocian con servicios y aplicaciones específicas para facilitar la comunicación entre dispositivos.

Diagrama:


INICIO
INGRESA DATO
DATO = --ip .../ -l o -h
YES
NO
LA IP EXISTE EN LA RED
YES
NO
IMPRIME MAC Y FABRICANTE
IMPRIME IP FUERA DE LA RED
DATO = --mac .../ -l o -h
YES
NO
LA MAC EXISTE EN LA BASE DE DATOS
IMPRIME EL FABRICANTE
IMPRIME EN FABRICANTE "NOT FOUND"
DATO = --arp .../ -l o -h
YES
IMPREME LA TABLA ARP CON IP/MAC/FABRICANTE
FIN