Group no: - 8

Group Member:- Shrut Shah – 19BCP125

Shubham Kathiriya – 19BCP127

Vedant Patel – 19BCP138

Subject: - Cyber Security Lab

Division:-2

Lab 5:- MAC changer Algorithm

Aim:-

Designing algorithm of MAC changer using Python.

Introduction:-

❖ What is MAC Address:-

- MAC address is the physical address, which uniquely identifies each device on a given network. To make communication between two networked devices, we need two addresses: IP address and MAC address. It is assigned to the NIC (Network Interface card) of each device that can be connected to the internet.
- It stands for Media Access Control, and also known as Physical address, hardware address, or BIA (Burned In Address).
- It is globally unique; it means two devices cannot have the same MAC address. It is represented in a hexadecimal format on each device, such as 00:0a:95:9d:67:16.
- It is 12-digit, and 48 bits long, out of which the first *24 bits are used for OUI(Organization Unique Identifier),* and *24 bits are for NIC/vendor-specific.*

❖ Characteristic of mac address:-

- TCP/IP networks can use MAC addresses in the communication
- It helps you to Identify a specific NIC in a computer on a network

- Network devices cannot efficiently route traffic using MAC addresses
- Not provide information about physical or logical network configuration.

❖ What need of changing MAC address:-

- Your ISP uses MAC address to identify or authenticate your Internet connection. So in case your network card goes boom, the new card you replace it with will have different MAC address and so the Internet wont work. So changing the MAC address to old network adapter is the quickest fix instead of telling your ISP to register your new MAC address which may take lot of time.
- If you want to access a network, which limits access based on MAC address, from another machine then you can change MAC address to the one for which you have access. Note that only one computer would be able to access the same network (no two computers can have same MAC address on same network to access it without any problem)
- A very important reason is **privacy**. Your MAC address can be seen by everyone on the local Ethernet network using many simple tools. A hacker on local network thus can track machines (and thus you) on the network. This is especially a threat when you are on a wireless network and are using a public WiFi network like in coffee shops, hotels or airports.
- If your original MAC address is revealed, an hacker can use it to impersonate you! On many networks (wired or wireless) access is restricted based on MAC address to avoid access to unauthorized devices on the network. So, when you go offline, someone can use your machine's MAC address and access the network as 'you'.
- You can get a new IP address lease from DHCP server by changing MAC address. On many networks, DHCP lease is set to last many days or is associated directly with a MAC address such that you get the same IP address all the time.

❖ Technique – MAC spoofing:-

- MAC spoofing is a technique for changing a factory-assigned Media Access Control (MAC) address of a network interface on a networked device. The MAC address that is hard-coded on a network interface controller (NIC) cannot be changed. However, many drivers allow the MAC address to be changed. Additionally, there are tools which can make an operating system believe that the NIC has the MAC address of a user's choosing. The process of masking a MAC address

is known as MAC spoofing. Essentially, MAC spoofing entails changing a computer's identity, for any reason

Algorithm:-

On Windows, we will be using three main commands, which are:

- **getmac**: This command returns a list of network interfaces and their MAC addresses and transport name; the latter is not shown when an interface is not connected.
- **reg**: This is the command used to interact with the Windows registry. We can use the winreg module for the same purpose. However, I preferred using the reg command.
- **wmic**: We'll use this command to disable and enable the network adapter, so the MAC address change is reflected

Code:-

- **Method-1 :- Using uuid.getnode()**

✓ **Module:-**

getnode() can be used to extract the MAC address of the computer. This function is defined in the uuid module. The illustrated code given below shows how to generate a UUID for a given host, identified by its MAC address, using the uuid1() function.

✓ **Code:-**

```
#Python Program to compute
# MAC address of host
# using UUID module
import uuid
# printing the value of unique MAC
# address using uuid and getnode() function
print (hex(uuid.getnode()))
```

✓ **Output Photo:-**

```
C:\Users\Subham\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/Subham/PycharmProjects/pythonProject/main.py
0x9828a64726b9

Process finished with exit code 0
```

✓ **Disadvantages:-**

The visible drawback is that the output is not in the formatted form

○ Method-2 :- Using getnode() + format()

✓ **Code:-**

```
# Python 3 code to print MAC
# in formatted way.
import uuid
# joins elements of getnode() after each 2 digits.
print ("The MAC address in formatted way is : ", end="")
print (':'.join(['{:02x}'.format((uuid.getnode() >> ele) & 0xff)
for ele in range(0,8*6,8)][::-1]))
```

✓ **Output Photo:-**

```
C:\Users\Subham\PycharmProjects\pythonProject\venv\Scripts\python.exe C:/Users/Subham/PycharmProjects/pythonProject/main.py
The MAC address in formatted way is : 98:28:a6:47:26:b9

Process finished with exit code 0
```

✓ **Disadvantages:-**

This code appears to be complex.

○ Method-3:- Using Subprocess and regex Library

```
import subprocess
import regex as re
import string
import random

network_interface_reg_path =
r"HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Control\\Class\\{4d36e972-
e325-11ce-bfc1-08002be10318}"
transport_name_regex = re.compile("{.+}")
mac_address_regex = re.compile(r"([A-Z0-9]{2}[:-]){5}([A-Z0-9]{2})")


def get_random_mac_address():
    uppercased_hexdigits = ''.join(set(string.hexdigits.upper()))
    return random.choice(uppercased_hexdigits) + random.choice("24AE") +
"".join(
        random.sample(uppercased_hexdigits, k=10))


def clean_mac(mac):
    return "".join(c for c in mac if c in string.hexdigits).upper()
```

```
def get_connected_adapters_mac_address():
    connected_adapters_mac = []
    for potential_mac in
subprocess.check_output("getmac").decode().splitlines():
        mac_address = mac_address_regex.search(potential_mac)
        transport_name = transport_name_regex.search(potential_mac)
        if mac_address and transport_name:
            connected_adapters_mac.append((mac_address.group(),
transport_name.group()))
    return connected_adapters_mac

def get_user_adapter_choice(connected_adapters_mac):
    for i, option in enumerate(connected_adapters_mac):
        print(f"#{i}: {option[0]}, {option[1]}")
    if len(connected_adapters_mac) <= 1:
        return connected_adapters_mac[0]
    try:
        choice = int(input("Please choose the interface you want to change the
MAC address:"))
        return connected_adapters_mac[choice]
    except:
        print("Not a valid choice, quitting...")
        exit()

def change_mac_address(adapter_transport_name, new_mac_address):
    output = subprocess.check_output(f"reg QUERY " +
network_interface_reg_path.replace("\\\\", "\\")).decode()
    for interface in re.findall(rf"{network_interface_reg_path}\\\d+",
output):
        adapter_index = int(interface.split("\\")[-1])
        interface_content = subprocess.check_output(f"reg QUERY
{interface.strip()}").decode()
        if adapter_transport_name in interface_content:
            changing_mac_output = subprocess.check_output(f"reg add
{interface} /v NetworkAddress /d {new_mac_address} /f").decode()
            print(changing_mac_output)
            break
    return adapter_index

def disable_adapter(adapter_index):
    disable_output = subprocess.check_output(f"wmic path win32_networkadapter
where index={adapter_index} call disable").decode()
    return disable_output


def enable_adapter(adapter_index):
    enable_output = subprocess.check_output(f"wmic path win32_networkadapter
where index={adapter_index} call enable").decode()
    return enable_output


if __name__ == "__main__":
    import argparse

    parser = argparse.ArgumentParser(description="Python Windows MAC changer")
    parser.add_argument("-r", "--random", action="store_true", help="Whether
to generate a random MAC address")
    parser.add_argument("-m", "--mac", help="The new MAC you want to change
to")
    args = parser.parse_args()
    if args.random:
        new_mac_address = get_random_mac_address()
    elif args.mac:
        new_mac_address = clean_mac(args.mac)

    connected_adapters_mac = get_connected_adapters_mac_address()
```

```
    old_mac_address, target_transport_name =
get_user_adapter_choice(connected_adapters_mac)
    print("[*] Old MAC address:", old_mac_address)
    adapter_index = change_mac_address(target_transport_name, new_mac_address)
    print("[+] Changed to:", new_mac_address)
    disable_adapter(adapter_index)
    print("[+] Adapter is disabled")
    enable_adapter(adapter_index)
    print("[+] Adapter is enabled again")
```

✓ **Output in Console :-**

```
C:/Users/Subham/PycharmProjects/pythonProject/CS_Mac_changer.py
#0: 98-28-A6-47-26-B9, {16ADEBB8-4C6C-4174-84ED-0C2988593CA5}
#1: 00-50-56-C0-00-01, {37602871-0F9F-494E-A241-1B6FE07F3364}
#2: 02-00-4C-4F-4F-50, {DD1B45DA-B5D4-46D0-B4EA-3E07FA35BF0F}

Please choose the interface you want to change the MAC address:2

[*] Old MAC address: 00-50-56-C0-00-08
The operation completed successfully.

[+] Changed to: 02-00-4C-4F-4F-50

[+] Adapter is disabled

[+] Adapter is enabled again
```

- **Method-4 :- Using getnode() + findall() + re()**

  [for reducing complexity]

✓ **Code:-**

```
# Python 3 code to print MAC
# in formatted way and easier
# to understand
import re, uuid
# joins elements of getnode() after each 2 digits.
# using regex expression
print ("The MAC address in formatted and less complex way is : ", end="")
print (':'.join(re.findall('..', '%012x' % uuid.getnode())))
```

✓ **Output Photo:-**

✓ **Advantages:-**

This code appears to be complex.

Benefits:-

- Changing the assigned MAC address may allow the user to bypass access control lists on servers or routers, either hiding a computer on a network or allowing it to impersonate another network device.
- However, MAC spoofing does not work when trying to bypass parental controls if automatic MAC filtering is turned on. MAC spoofing is done for legitimate and illicit purposes alike.

MAC Address Randomization in WiFi

- To prevent third parties from using MAC addresses to track devices, Android, Linux, iOS, and Windows have implemented MAC address randomization.
- In June 2014, Apple announced that future versions of iOS would randomize MAC addresses for all Wi-Fi connections.
- The Linux kernel has supported MAC address randomization during network scans since March 2015, but drivers need to be updated to use this feature.
- Windows has supported it since the release of Windows 10 in July 2015.