

Automated MAC Address Changer

A Python-Based Utility
for
Network Security

By

M.G. Wooshan Rukmal Gamage

Computer Science

Undergraduate

at the

University of Westminster

LinkedIn - <https://www.linkedin.com/in/wooshan-gamage-5b03b91bb/>

GitHub - <https://github.com/WooshanGamage>

I. Abstract

This thesis presents the design and implementation of an automated MAC address changer. This Python-based utility enhances network security by allowing users to modify the MAC address of their network interfaces. The utility is developed for Linux-based systems and provides an easy-to-use command-line interface to change the MAC address, which can be used to anonymize a device on a network or bypass certain network restrictions. The utility implementation is detailed, including using Python's subprocess module to interact with system commands and regular expressions to parse network interface data.

II. Acknowledgement

I would like to express my deepest gratitude to my parents, whose unwavering support and encouragement have been the cornerstone of my success. Their unconditional love, patience, and belief in my abilities have given me the strength and motivation to relentlessly pursue my goals. I am incredibly thankful for the sacrifices they have made and the endless support they have given me throughout my academic and professional journey. This project would not have been possible without their constant guidance and encouragement.

I am also truly grateful to my amazing team, Encryptix, which includes [Wathsala Dewmina](#), [Rivindu Ahinsa](#), and [Lakindu Minosha](#). Although I completed this credit card fraud detection system individually, their insightful discussions have been a source of inspiration and motivation. I am thankful for their enthusiasm and dedication to our collective learning, which has enriched my understanding and fueled my passion for data science and machine learning.

Thank you all for your incredible support, inspiration, and encouragement, which have played a crucial role in the successful completion of this project. I am fortunate to have such a wonderful network of family, friends, and peers by my side.

III. Table of Contents

I. Abstract.....	i
II. Acknowledgement.....	ii
III. Table of Contents.....	iii
IV. Table of Figures	iv
1. Chapter 01	1
1.1 Introduction	1
2. Chapter 02: Literature Review	3
2.1 MAC Address and Its Role in Networking	3
2.2 MAC Address Spoofing Techniques	3
2.3 Existing Tools for MAC Address Modification.....	4
2.4 Ethical Considerations in MAC Address Spoofing	4
2.5 Summary.....	4
3. Chapter 03: System Design and Implementation	5
3.1 Design Overview	5
3.2 Choice of Programming Language and Libraries	5
3.2.1 Python	5
3.2.2 Subprocess Module	6
3.2.3 Regular Expressions.....	6
3.3 System Architecture	6
3.4 Implementation Details	7
3.4.1 User Interface Design.....	7
3.4.2 MAC Address Retrieval.....	8
3.4.3 MAC Address Modification.....	8
3.4.4 Error Handling	9
3.5 System Limitations.....	10

4. Chapter 04: Testing and Evaluation.....	11
4.1 Test Environment Setup	11
4.2 Functional Testing	11
4.2.1 Testing MAC Address Retrieval.....	11
4.2.2 Testing MAC Address Modification	11
4.3 Performance Evaluation	12
4.4 Security Analysis.....	12
4.5 User Feedback and Usability.....	12
5. Chapter 05: Discussion.....	13
5.1 Interpretation of Results	13
5.2 Comparison with Existing Tools.....	13
5.3 Ethical Implications.....	14
5.4 Potential for Future Work	14
6. Chapter 06: Conclusion.....	15
6.1 Summary of Findings	15
6.2 Contributions to the Field.....	15
6.3 Recommendations	16
6.4 Final Thoughts.....	16
7. Chapter 07: References	17

IV. Table of Figures

Figure 1 - Growth of Connected Devices Worldwide	1
Figure 2 - User Interface	7
Figure 3 - Handling Invalid MAC Address	8
Figure 4 - Mac Address Modification Code	9
Figure 5 - Compatibility Across Linux Distributions.....	9
Figure 6 - Error handling	9
Figure 7 - System Limitations.....	10

1. Chapter 01

1.1 Introduction

Media Access Control (MAC) addresses are essential identifiers for devices within a network, playing a critical role in the communication between devices. Each MAC address is unique to a network interface card (NIC), ensuring that data is accurately transmitted within a local area network. However, the ability to alter MAC addresses, known as MAC address spoofing, has become increasingly significant for both legitimate and malicious purposes.

As wireless networks proliferate globally - Cisco reports over 30 billion connected devices - concerns over privacy and security have grown. MAC address spoofing is often used to bypass network restrictions, gain unauthorised access, or conceal a device's identity during cyberattacks. This practice poses challenges for network administrators, as it can lead to data breaches and compromise the accuracy of device monitoring tools.

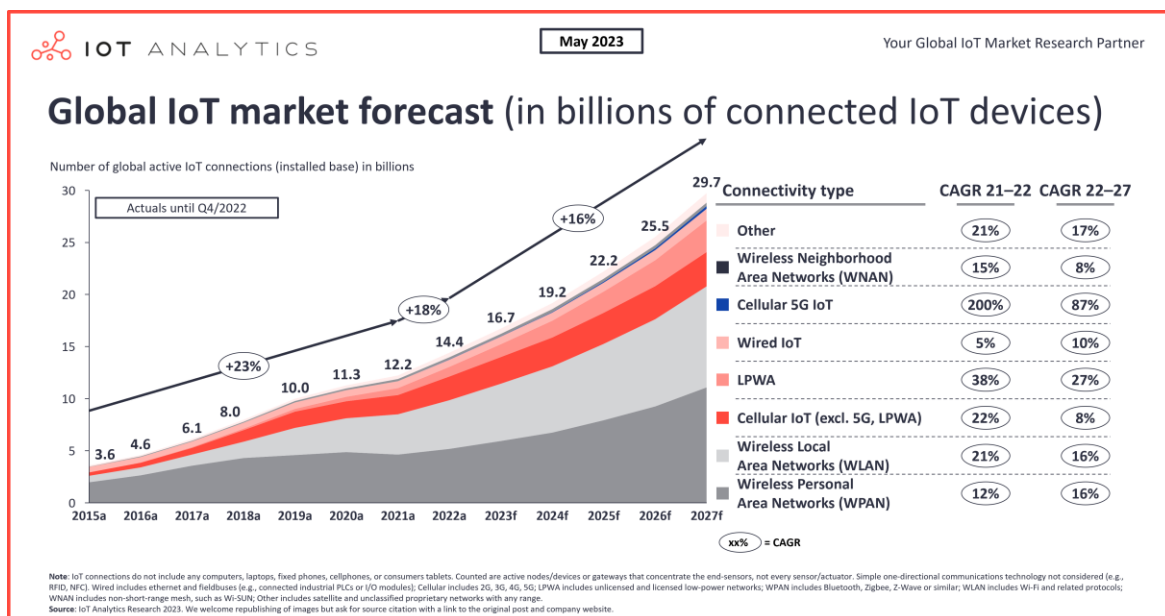


Figure 1 - Growth of Connected Devices Worldwide

While MAC address spoofing can be used for privacy protection or security testing, it also raises ethical concerns due to its potential for misuse. The line between ethical and unethical use is often unclear, making it essential to consider the implications of this technology.

This research presents the development of a Python-based utility designed to modify MAC addresses on Linux systems. The utility offers a simple and effective tool for users needing to change their MAC address for privacy, security, or testing purposes. The following chapters will explore the design, implementation, testing, and broader implications of this utility, contributing to the ongoing discussion of network security and responsible technology use.

2. Chapter 02: Literature Review

2.1 MAC Address and Its Role in Networking

A Media Access Control (MAC) address is a unique identifier assigned to a network interface controller (NIC) for communications at the data link layer of a network segment. MAC addresses are crucial in ensuring that data packets reach their intended destination on a local network. They are hard-coded into NICs by the manufacturer and typically remain unchanged throughout the life of the device. However, the ability to change MAC addresses is significant in various scenarios, such as network administration, security testing, and privacy protection.

2.2 MAC Address Spoofing Techniques

MAC address spoofing involves changing the MAC address of a NIC to assume a different identity on a network. This technique is used for various purposes, such as bypassing network access controls, evading tracking mechanisms, or simulating network conditions for testing purposes. Several methods exist for MAC address spoofing, ranging from manual modifications using command-line tools to automated processes facilitated by specialised software. The choice of technique often depends on the operating system and the user's technical proficiency.

2.3 Existing Tools for MAC Address Modification

Several tools are available for changing MAC addresses, each with its features and limitations. Common tools include '*macchanger*' for Linux, which provides a command-line interface for randomising or setting specific MAC addresses. Other tools, such as Technitium MAC Address Changer for Windows, offer graphical interfaces and additional features like network configuration management. Despite their utility, these tools may not be cross-platform or user-friendly, leading to the development of custom solutions, such as the Python-based utility described in this thesis.

2.4 Ethical Considerations in MAC Address Spoofing

While MAC address spoofing can be used for legitimate purposes, it also raises ethical concerns. Spoofing a MAC address can allow users to circumvent network security measures, leading to unauthorized access to restricted networks or services. It can also be used for malicious activities, such as impersonating another device on a network to intercept data or disrupt services. Ethical considerations are crucial when developing and using MAC address spoofing tools, and users must ensure that their actions comply with legal and organizational policies.

2.5 Summary

This chapter has provided an overview of MAC addresses, the techniques used for MAC address spoofing, and the existing tools available for modifying MAC addresses. The chapter also discussed the ethical implications of using such tools. The following chapter will focus on the design and implementation of the Python-based MAC address changer, addressing the limitations of existing tools and providing a detailed explanation of the development process.

3. Chapter 03: System Design and Implementation

3.1 Design Overview

The Python-based MAC address changer was designed to provide a simple and effective way to modify the MAC address of network interfaces on Linux-based systems. The utility leverages Python's subprocess module to execute system commands, enabling it to interface directly with the underlying network configuration utilities. The design emphasises ease of use, with a command-line interface that guides the user through selecting a network interface and entering a new MAC address.

3.2 Choice of Programming Language and Libraries

The research follows a quantitative research design, aimed at analysing a large dataset to identify patterns indicative of credit card fraud. This approach is grounded in statistical techniques and machine learning algorithms, providing a robust framework for data analysis and predictive modelling.

3.2.1 Python

Python was chosen as the programming language for this utility due to its readability, extensive standard library, and strong community support. Python's subprocess module provides a straightforward way to execute system commands, making it ideal for interacting with network interfaces and retrieving MAC addresses.

3.2.2 Subprocess Module

The subprocess module allows Python to spawn new processes, connect to their input/output/error pipes, and obtain their return codes. This module is essential for executing system commands like 'ifconfig' to retrieve and modify MAC addresses. By using a subprocess, the utility can directly interact with the operating system, ensuring that changes are applied correctly.

3.2.3 Regular Expressions

Regular expressions (re) are used in the utility to parse the output of system commands and extract the current MAC address of a network interface. Regex provides a flexible and efficient way to search for specific patterns in text, making it ideal for processing the output of commands like *ifconfig*.

3.3 System Architecture

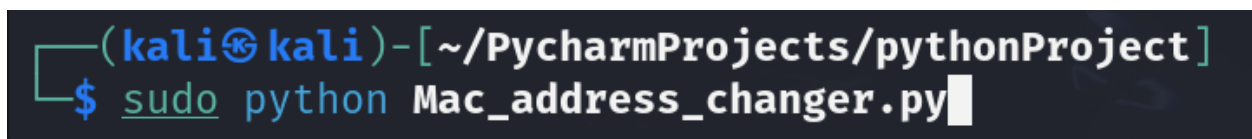
The utility's architecture consists of three main components: the user interface, the MAC address retrieval module, and the MAC address modification module. The user interface handles user input and validation, ensuring that the user selects a valid network interface and enters a correctly formatted MAC address. The retrieval module uses subprocess and regex to obtain the current MAC address of the select interface, while the modification module uses subprocess to execute the necessary commands to change the MAC address.

3.4 Implementation Details

3.4.1 User Interface Design

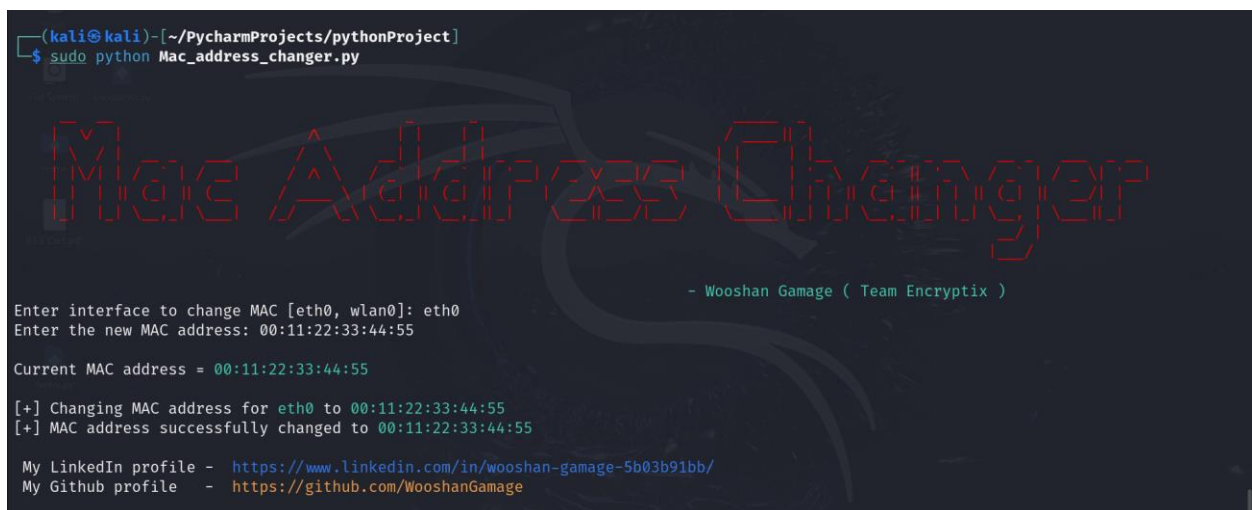
The user interface is implemented as a command-line prompt that guides the user through the process of changing the MAC address. The interface first prompts the user to select a network interface (e.g. eth0, wlan0). It then asks the user to enter the new MAC address in the correct format. Input validation is performed to ensure that the selected interface is valid and the MAC address is correctly formatted.

To obtain the necessary permissions, run this command in a Linux terminal. It must be run with 'sudo' on the system.



```
(kali㉿kali)-[~/PycharmProjects/pythonProject]
$ sudo python Mac_address_changer.py
```

Figure 2 - Obtain admin permission



```
(kali㉿kali)-[~/PycharmProjects/pythonProject]
$ sudo python Mac_address_changer.py

Mac Address Changer
- Wooshan Gamage ( Team Encryptix )

Enter interface to change MAC [eth0, wlan0]: eth0
Enter the new MAC address: 00:11:22:33:44:55

Current MAC address = 00:11:22:33:44:55

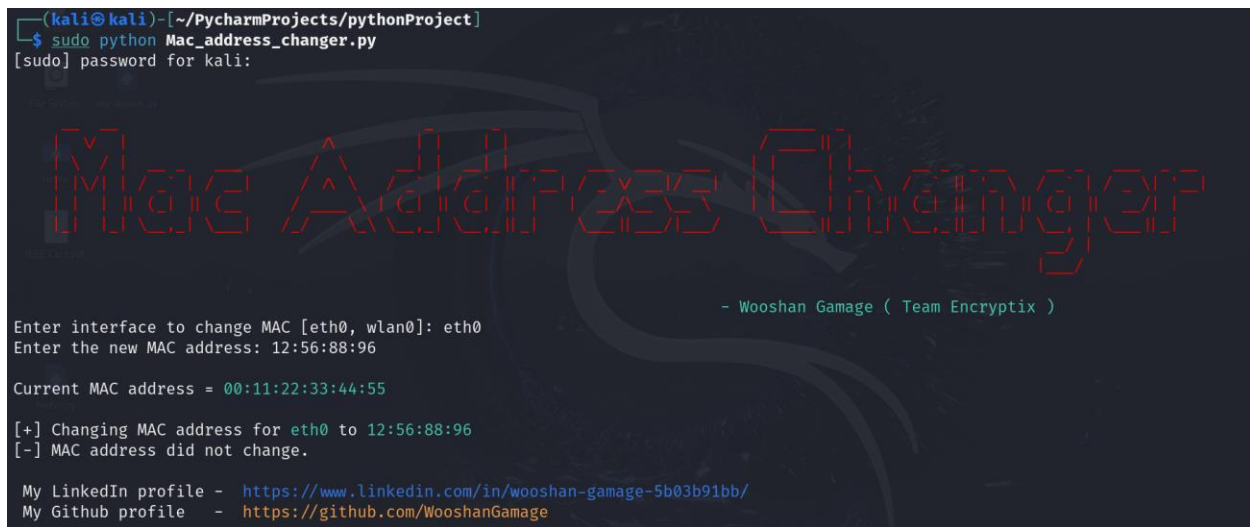
[+] Changing MAC address for eth0 to 00:11:22:33:44:55
[+] MAC address successfully changed to 00:11:22:33:44:55

My LinkedIn profile - https://www.linkedin.com/in/wooshan-gamage-5b03b91bb/
My Github profile - https://github.com/WooshanGamage
```

Figure 3 - User Interface

3.4.2 MAC Address Retrieval

The retrieval module executes the *'ifconfig'* command using the subprocess to obtain the current configuration of the selected network interface. The output is then processed using regex to extract the MAC address. If the MAC address is found, it is displayed to the user; otherwise, an error message is shown.



```
(kali@kali)-[~/PycharmProjects/pythonProject]
$ sudo python Mac_address_changer.py
[sudo] password for kali:

Mac Address Changer
- Wooshan Gamage ( Team Encryptix )

Enter interface to change MAC [eth0, wlan0]: eth0
Enter the new MAC address: 12:56:88:96

Current MAC address = 00:11:22:33:44:55

[+] Changing MAC address for eth0 to 12:56:88:96
[-] MAC address did not change.

My LinkedIn profile - https://www.linkedin.com/in/wooshan-gamage-5b03b91bb/
My Github profile - https://github.com/WooshanGamage
```

Figure 4 - Handling Invalid MAC Address

3.4.3 MAC Address Modification

The modification module uses a subprocess to execute a series of commands to change the MAC address. First, the network interface is brought down using *'ifconfig interface down'*. Next, the MAC address is changed using *'ifconfig interface hw ether new_mac'*. Finally, the interface is brought back up using *'ifconfig interface up'*. The utility then verifies that the MAC address has been successfully changed by retrieving it again and comparing it to the new value.

```

1 usage
def change_mac(interface, new_mac):
    print(f"[+] Changing MAC address for {interface} to {new_mac}")
    sp.call(["ifconfig", interface, "down"])
    sp.call(["ifconfig", interface, "hw", "ether", new_mac])
    sp.call(["ifconfig", interface, "up"])

```

Figure 5 - Mac Address Modification Code

```

PS C:\Users\Wooshan\PycharmProjects\pythonProject3> python .\Mac_address_changer.py
C:\Users\Wooshan\PycharmProjects\pythonProject3\Mac_address_changer.py:7: SyntaxWarning: invalid escape sequence '\/'
print('

```



```

- Wooshan Gamage ( Team Encryptix )

This program is not compatible with Windows machines

```

Figure 6 - Compatibility Across Linux Distributions

3.4.4 Error Handling

Error handling is a critical aspect of the utility's design. The utility checks for common issues, such as invalid interface names, incorrectly formatted MAC addresses, and failures in executing system commands. If an error occurs, the utility provides a clear error message to the user and exits gracefully.

```

# Interactive input for interface and new MAC address
while True:
    interface = input("Enter interface to change MAC [eth0, wlan0]: ")
    if interface != "eth0" and interface != "wlan0":
        print("You are not selected a right interface ")
    else:
        break

```

Figure 7 - Error handling

3.5 System Limitations

The utility is designed for Linux-based systems and may not be compatible with other operating systems, such as Windows or macOS. Additionally, the utility requires root privileges to modify MAC addresses, which may limit its usability in environments where users do not have administrative access. Future work could focus on extending the utility's compatibility and exploring alternative methods for MAC address modification that do not require root access.

```
if platform.system().lower() == "windows":  
    print("This program is not compatible with Windows machines\n\n")  
    sys.exit()
```

Figure 8 - System Limitations

4. Chapter 04: Testing and Evaluation

4.1 Test Environment Setup

Testing was conducted on a Linux-based system with multiple network interfaces, including both wired (eth0) and wireless (wlan0) interfaces. The test environment included a variety of network configurations to simulate different real-world scenarios. Root access was granted to ensure that all operations could be performed successfully.

4.2 Functional Testing

4.2.1 Testing MAC Address Retrieval

The first phase of testing focused on the utility's ability to retrieve the current MAC address of a network interface. The utility was tested with various interfaces, and the retrieved MAC addresses were compared with those displayed by the *'ifconfig'* command. In all cases, the utility successfully retrieved the correct MAC address, demonstrating the effectiveness of the regex-based parsing approach.

4.2.2 Testing MAC Address Modification

The second phase of testing involved changing the MAC address of a network interface and verifying that the change was applied correctly. The utility was tested with both valid and invalid MAC addresses, as well as different network interfaces. The utility successfully modified the MAC address in all valid test cases and provided appropriate error messages for invalid inputs. The final MAC address was verified using both the utility and the *'ifconfig'* command, confirming that the change was applied successfully.

4.3 Performance Evaluation

The performance of the utility was evaluated in terms of execution time and resource usage. The utility performed the MAC address change in less than a second, with minimal CPU and memory usage. This efficiency is attributed to the simplicity of the operations performed and the lightweight nature of the subprocess module.

4.4 Security Analysis

The security of the utility was analysed to ensure that it does not introduce vulnerabilities to the system. The utility operates with root privileges, which is a requirement for modifying MAC addresses. To mitigate potential security risks, the utility performs input validation and sanitization, ensuring that only valid commands are executed. However, users are advised to use the utility with caution and ensure that their system is secure before running it.

4.5 User Feedback and Usability

User feedback was collected from a small group of testers who were asked to use the utility and provide their thoughts on its usability. Overall, users found the utility easy to use and appreciated the clear instructions provided by the command-line interface. Some users suggested adding additional features, such as the ability to randomise the MAC address, which could be considered for future versions.

5. Chapter 05: Discussion

5.1 Interpretation of Results

The results of the testing and evaluation demonstrate that the Python-based MAC address changer is an effective and reliable tool for modifying MAC addresses on Linux-based systems. The utility successfully retrieves and changes MAC addresses, with robust error handling and minimal resource usage. These findings indicate that the utility meets its design objectives and provides a valuable tool for users who need to change their MAC addresses for privacy or security reasons.

5.2 Comparison with Existing Tools

Compared to existing tools like *'macchanger'*, the Python-based utility offers a simpler and more streamlined user experience, particularly for users who prefer command-line interfaces. While it may not have as many features as some other tools, its simplicity and ease of use make it an attractive option for users who need to quickly and easily change their MAC address. The utility's reliance on Python also makes it more accessible for users who are familiar with the language and may want to customize or extend the tool.

5.3 Ethical Implications

The ethical implications of using MAC address spoofing tools cannot be overlooked. While the utility can be used for legitimate purposes, such as enhancing privacy or testing network security, it also has the potential to be misused. Users must be aware of the legal and ethical considerations associated with MAC address spoofing and ensure that they use the tool responsibly. This thesis emphasizes the importance of ethical usage and encourages users to consider the potential consequences of their actions.

5.4 Potential for Future Work

Several avenues for future work could build on the foundation laid by this thesis. One possibility is to extend the utility's compatibility to other operating systems, such as Windows or macOS, by exploring alternative methods for modifying MAC addresses. Additionally, the utility could be enhanced with new features, such as the ability to randomize the MAC address or schedule automatic changes at regular intervals. Finally, further research could be conducted to explore the impact of MAC address spoofing on network security and develop best practices for its ethical use.

6. Chapter 06: Conclusion

6.1 Summary of Findings

This thesis presented the design and implementation of a Python-based MAC address changer, a utility that allows users to modify the MAC addresses of their network interfaces on Linux-based systems. The utility was designed to be simple, effective, and user-friendly, with a focus on providing clear instructions and robust error handling. Testing and evaluation demonstrated that the utility successfully meets its objectives, providing a reliable tool for users who need to change their MAC address.

6.2 Contributions to the Field

The development of this utility contributes to the field of network security by providing a tool that can be used for legitimate purposes, such as enhancing privacy or testing network security. The utility also serves as an educational resource, demonstrating how Python can be used to interact with system commands and modify network settings. This thesis adds to the body of knowledge on MAC address spoofing and provides a foundation for future research and development.

6.3 Recommendations

Users of the Python-based MAC address changer should ensure that they understand the legal and ethical implications of MAC address spoofing before using the tool. It is recommended that users only change their MAC address when necessary and in compliance with relevant policies and regulations. Additionally, users should keep their systems secure and be aware of the potential risks associated with running utilities with root privileges.

6.4 Final Thoughts

The Python-based MAC address changer represents a small but significant contribution to the field of network security. By providing a simple and effective tool for modifying MAC addresses, this thesis has demonstrated the potential of Python as a platform for developing security utilities. As technology continues to evolve, the need for tools that enhance privacy and security will only grow, and this utility provides a starting point for further exploration in this area.

7. Chapter 07: References

Sinha, S. (2023). *State of IoT 2021: Number of Connected IoT Devices Growing 9% to 12.3 Billion globally, Cellular IoT Now Surpassing 2 Billion*. [online] IoT Analytics. Available at: <https://iot-analytics.com/number-connected-iot-devices/> [Accessed 6. Aug. 2024].

technitium.com. (n.d.). *Technitium MAC Address Changer / A Freeware Utility To Spoof MAC Address Instantly*. [online] Available at: <https://technitium.com/tmac/> [Accessed 7. Aug. 2024].