Business Case & Technical Report

Implementing MAC Changer for Enhanced Network Security in Kali Linux

Submitted By: PTID-CHE-SEP-25-203

Arya Venkat

Mishal V S

Puchakayala Vara Prasad

Date of Submission:
26th September 2025

TABLE OF CONTENT

1. INTRODUCTION 3

Background 3
Importance 3
Scope of the Report 3
Objectives 3

2. PROBLEM STATEMENT 4

Security Challenges 4
Limitations of Existing Measures 4
Risks 4

3. PROPOSED SOLUTION / APPROACH 5

Tool Description 5
How it helps 5
Simple Workflow 5
Assumptions & Limits 5

4. DETAILED IMPLEMENTATION / METHODOLOGY 6

Lab Setup 6
Step-by-Step Procedure 6
Testing Approach 6
Network Topology 6
Configuration Details 6
Features and Benefits 7
Benefits 7

5. USE CASES 8

6. TESTING / RESULTS 10

Test Objectives 10
Methodology 10

7. DISCUSSION 23

Interpretation of Results: 23
Lessons Learned: 23
Limitations: 23
Ethical and Legal Considerations: 23

8. RECOMMENDATIONS 24

Security Improvements / Hardening Measures: 24
Future Work / Enhancements: 24
Mitigation Strategies for Potential Misuse: 24

9. CONCLUSION 25

10. REFERENCES / BIBLIOGRAPHY 26

Implementing MAC Changer for Enhanced Network Security in Kali Linux

1. Introduction

Background

A **MAC address** is a unique identifier assigned to each network device (for example, a laptop or phone). It lets devices talk on local networks but can also be used to track devices because it's often fixed. **MAC Changer** (available in Kali Linux) lets you change or "spoof" a device's MAC address — you can set a custom value, generate a random one, or restore the original.

Importance

- **Privacy:** Changing the MAC helps prevent tracking on public or shared networks.
- **Security testing:** Demonstrates weaknesses in MAC-based filtering and access control.
- **Penetration testing & troubleshooting:** Useful in labs to simulate attacks or to emulate devices without swapping hardware.

Scope of the Report

This report covers using MAC Changer in a controlled lab to: understand MAC-related risks, perform MAC spoofing, test how networks respond, and recommend safe, ethical practices. It only describes authorized testing — not illegal use.

Objectives

1. Explain what MAC addresses do.
2. Show how to use MAC Changer to randomize or set MACs.
3. Evaluate benefits and risks of spoofing.
4. Document steps, tests, and results clearly.
5. Recommend safe, ethical practices for using MAC spoofing.

2. Problem Statement

MAC addresses are unique identifiers for network devices, but this uniqueness can be exploited, creating security and privacy risks for organizations and users.

Security Challenges

- Device Tracking: Attackers can capture MAC addresses to monitor activity or track devices.
- Weak Access Controls: MAC-based filtering can be bypassed since addresses can be easily spoofed.
- Privacy Issues: Static MACs allow third parties to identify and profile devices, especially on public networks.

Limitations of Existing Measures

- MAC filtering and access lists alone are not enough to stop tracking.
- Network monitoring may not detect spoofed MAC addresses.
- Built-in OS privacy measures often fail to fully protect against tracking.

Risks

- Unauthorized Monitoring: Sensitive activity can be observed without consent.
- Security Bypass: Attackers may gain access using spoofed MACs.
- Regulatory Risk: Organizations may violate privacy laws like GDPR or HIPAA.

3. Proposed Solution / Approach

Tool Description

MAC Changer is a small tool in Kali Linux that lets you change the MAC address of a network interface. You can:

- Set a specific MAC address.
- Create a random MAC address.
- Restore the device's original MAC address.

How it helps

- **Protects privacy:** Changing the MAC makes it harder for others to track the device.
- **Tests security:** It shows whether MAC-based filters and access lists are easy to bypass.
- **Raises awareness:** Demonstrates why networks should not rely only on MAC addresses for security.

Simple Workflow

- Check the device's current MAC address.
- Use MAC Changer to set a new MAC (random or chosen).
- Connect to the network with the new MAC.
- Observe whether the network allows access or logs the change.

[Device with Original MAC] → [Run MAC Changer] → [Device with New MAC] → [Network / Tests]

Assumptions & Limits

- Tests are done only on **networks where you have permission**.
- You need **admin/root rights** to change the MAC.
- Changing MAC helps only against MAC-based tracking — it does **not** stop advanced tracking methods.

4. Detailed Implementation / Methodology

Lab Setup

- Operating System: Kali Linux (VM or physical machine)
- Network: Test network or isolated VLAN for safe MAC testing
- Tools Installed: macchanger, ifconfig, ip

Step-by-Step Procedure

- Check the current MAC: ifconfig or ip link show <interface>
- Install MAC Changer: sudo apt update && sudo apt install macchanger
- Change MAC temporarily: sudo macchanger -r <interface> (random MAC)
- Set a custom MAC: sudo macchanger -m 00:11:22:33:44:55 <interface>
- Restore original MAC: sudo macchanger -p <interface>
- Verify MAC change: ifconfig <interface>

Testing Approach

- Connect to the test network before and after MAC changes
- Observe if access controls or monitoring systems detect the change
- Capture traffic using Wireshark or tcpdump to verify anonymization

Network Topology

- [Kali Linux VM] — [Test Switch/VLAN] — [Access Point/Router]
- Allows safe testing without affecting real networks

Configuration Details

- Interfaces tested: eth0 (wired) or wlan0 (wireless)
- No permanent hardware changes; all modifications are reversible
- Testing only on authorized networks

Features and Benefits

- MAC Anonymization: Hides the real MAC to prevent tracking
- Randomization & Spoofing: Generate random or custom MACs for testing and privacy
- Access Control Bypass: Helps bypass simple MAC filters in authorized testing
- Integration: Works with other security tools for assessments

Benefits

- Enhanced privacy and reduced tracking risk
- Identifies weaknesses in MAC-based access controls
- Useful learning tool for cybersecurity professionals
- Supports ethical and authorized network testing.

5. Use Cases

1. Network Anonymization

- **Scenario:** A user connects to a public Wi-Fi or shared network.
- **Use:** By changing the MAC address, the device becomes anonymous, preventing tracking by ISPs, network admins, or other users.
- **Benefit:** Protects user privacy and prevents profiling based on MAC addresses.

2. Access Control Testing / Bypass

- **Scenario:** Organizations sometimes use MAC filtering to restrict network access.
- **Use:** Security testers or admins can use MAC Changer to verify if such filters can be bypassed.
- **Benefit:** Helps identify weaknesses in MAC-based security and improve network defenses.

3. Privacy Enhancement on Corporate or Sensitive Networks

- **Scenario:** Employees or auditors connect to networks where privacy is critical.
- **Use:** Changing MAC addresses periodically reduces the risk of device tracking or profiling.
- **Benefit:** Enhances data security and protects user/device identity.

4. Compliance and Regulatory Measures

- **Scenario:** Organizations handling sensitive data (e.g., healthcare or financial) must comply with privacy regulations like GDPR or HIPAA.
- **Use:** MAC anonymization ensures network activity cannot be easily traced to individual devices.
- **Benefit:** Supports regulatory compliance and reduces legal/privacy risks.

5. Educational / Penetration Testing

- **Scenario:** Students, interns, or security professionals learning about network security.
- **Use:** MAC Changer can be used in lab environments to demonstrate spoofing, anonymization, and testing network controls safely.
- **Benefit:** Provides hands-on experience with ethical hacking and security testing without impacting production networks.

6. Testing / Results

Test Objectives

- Verify MAC address change works using MAC Changer.
- Measure network reachability with spoofed MAC addresses.

Methodology

- **Tools:** macchanger, tcpdump, arpwatch, ip/ifconfig, ping, ebtables (Wireshark optional).
- **Environment:** Kali Linux VM. Interfaces used: eth0, veth-A, veth-B.

Step I — Install and Update Packages

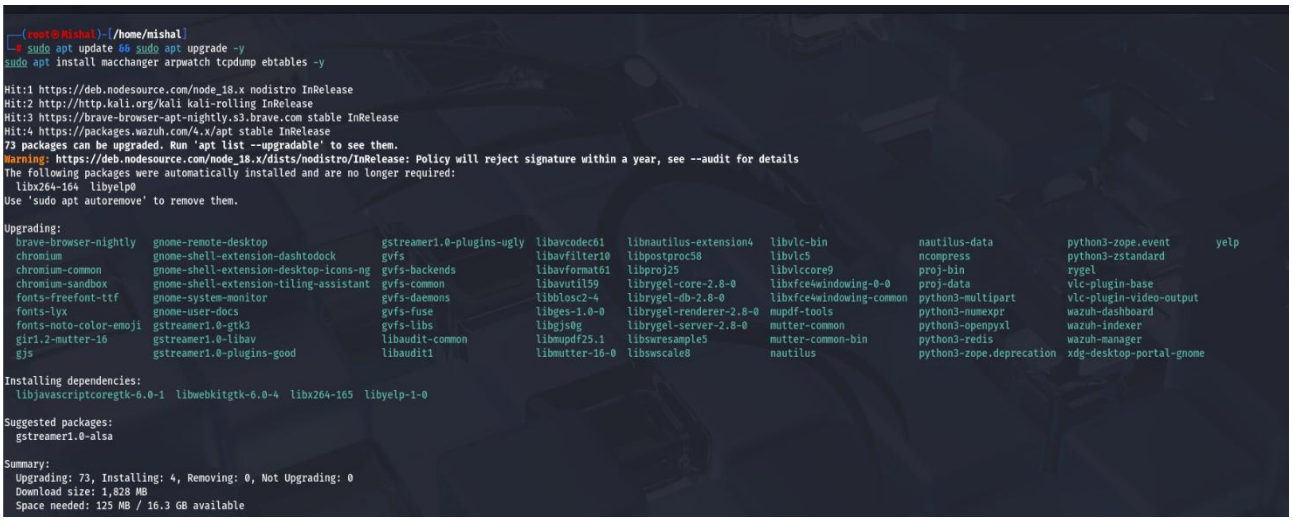
Command:

sudo apt update && sudo apt upgrade -y

sudo apt install macchanger arpwatch tcpdump ebtables -y

Explanation:

- This step ensures the system is up-to-date and all necessary tools for MAC testing are installed.
- Installing macchanger allows us to manipulate MAC addresses; tcpdump and arpwatch help monitor network traffic and MAC changes.



Observation:

- All required packages installed successfully. System shows available upgrades.

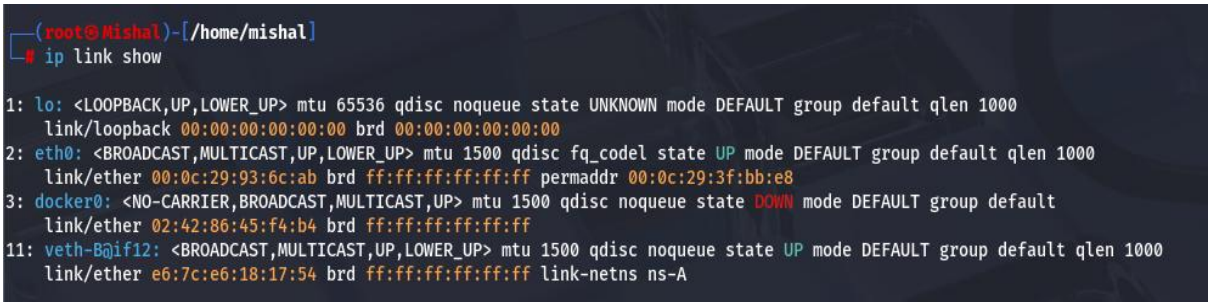
Step 2 — Check Network Interfaces

Command:

ip link show

Explanation:

- Lists all network interfaces on the machine and their current MAC addresses.
- Used to identify which interfaces are available for testing MAC changes (e.g., eth0, veth-A/B).



Observation:

- Interfaces eth0, veth-A, and veth-B are active and ready for testing.

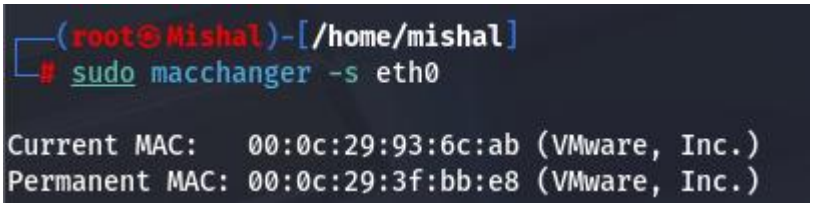
Step 3 — Check Current MAC (eth0)

Command:

sudo macchanger -s eth0

Explanation:

- Displays the current and permanent MAC addresses of eth0.
- Helps establish a baseline before making any changes.



Observation:

- Original MAC of eth0 recorded as 00:0c:29:93:6c:ab.

Step 4 — Randomize MAC (eth0)

Command:

sudo macchanger -r eth0

sudo macchanger -s eth0

Explanation:

- Assigns a randomly generated MAC to eth0.
- Useful to test network behavior with non-permanent, anonymized addresses.

Observation:

- eth0 successfully assigned a random MAC (fe:dc:84:fe:8b:7f). Connectivity remains intact.

Step 5 — Customize MAC (eth0)

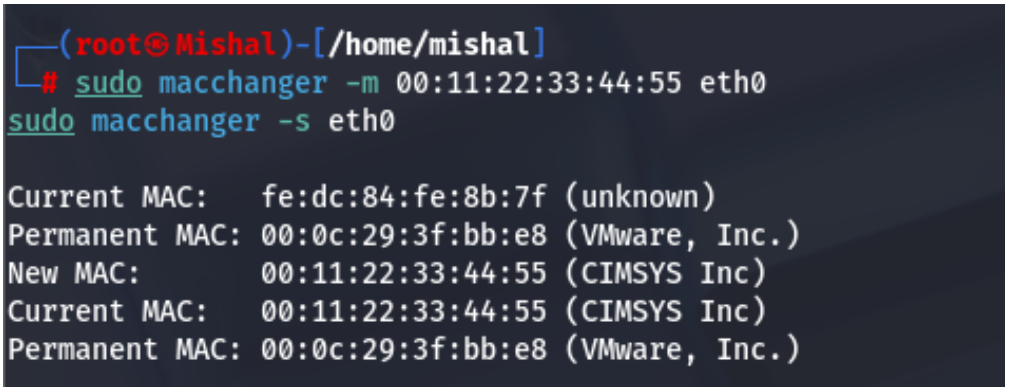
Command:

sudo macchanger -m 00:11:22:33:44:55 eth0

sudo macchanger -s eth0

Explanation:

- Sets a specific, user-defined MAC for the interface.
- Useful for testing MAC-based access control, filter bypass, and consistency.



Observation:

- MAC changed to 00:11:22:33:44:55. Verified using macchanger -s.

Step 6 — ACB: Ping Before Spoofing (eth0)

Command:

ping -c 3 192.168.1.1

Explanation:

- Tests connectivity to the gateway before any MAC changes.
- Ensures baseline network reachability is working before spoofing.

```
┌──(root㉿Mishal)-[/home/mishal]
└─# ping -c 3 192.168.1.1

PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=63 time=13.4 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=63 time=4.40 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=63 time=2.85 ms

--- 192.168.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 2.852/6.899/13.443/4.670 ms
```

Observation:

- Gateway reachable with 0% packet loss. Average RTT 6–7 ms.

Step 7 — ACB: Ping After Spoofing (eth0)

Command:

sudo macchanger -m 02:12:34:56:78:9A eth0

ping -c 3 192.168.1.1

Explanation:

- Confirms network remains accessible after MAC spoofing.
- Shows that spoofing does not break basic connectivity.

```
┌──(root㉿Mishal)-[/home/mishal]
└─# ping -c 3 192.168.1.1

PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=63 time=2.81 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=63 time=2.53 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=63 time=2.88 ms

--- 192.168.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 2.525/2.737/2.883/0.153 ms
```

Observation:

- Ping successful; packets received with minimal RTT changes.

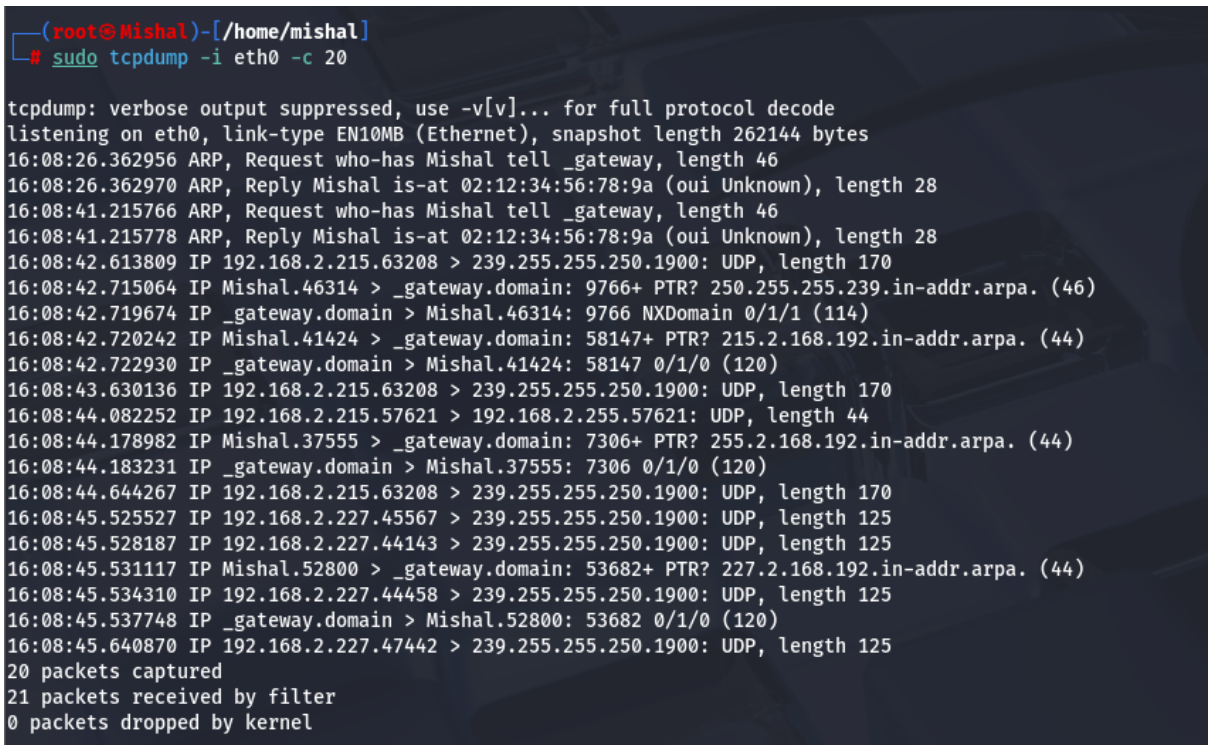
Step 8 — Packet Capture (tcpdump)

Command:

sudo tcpdump -i eth0 -c 20

Explanation:

- Captures network packets to verify that the MAC address seen on the wire matches the spoofed MAC.
- Ensures that MAC anonymization is effective in the actual traffic.



Observation:

- ARP packets show the new spoofed MAC. No connectivity issues observed.

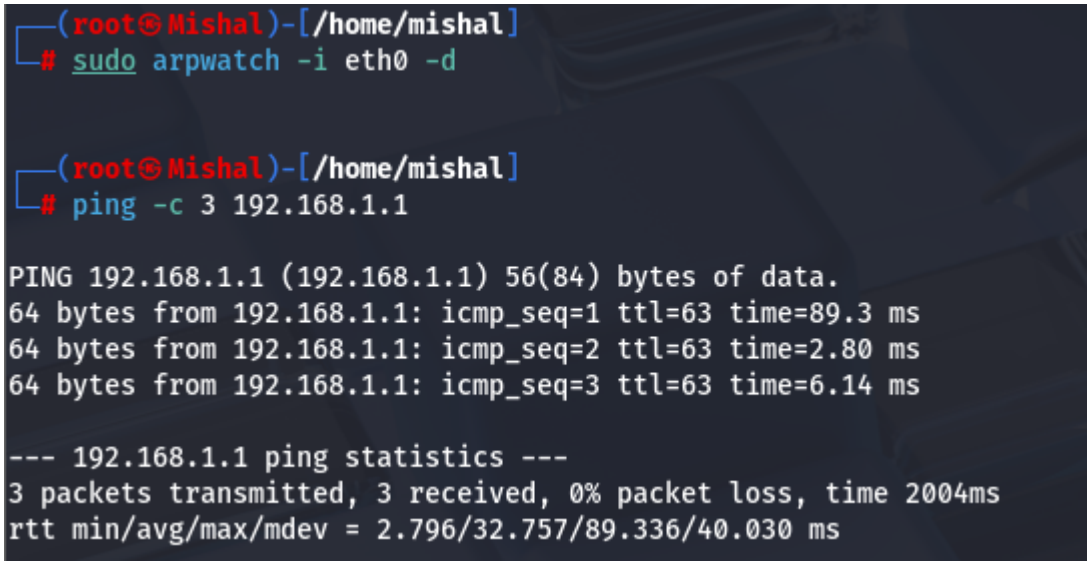
Step 9 — arpwatch Monitoring

Command:

sudo arpwatch -i eth0 -d

Explanation:

- Monitors and logs MAC changes on the network.
- Detects unauthorized MAC changes or suspicious activity.



Observation:

- arpwatch detects and logs MAC changes. Alerts can be generated if configured.

Step 10 — Simulate Network Behavior (veth pair)

Command:

sudo ip link add veth-A type veth peer name veth-B

sudo ip link set veth-A up

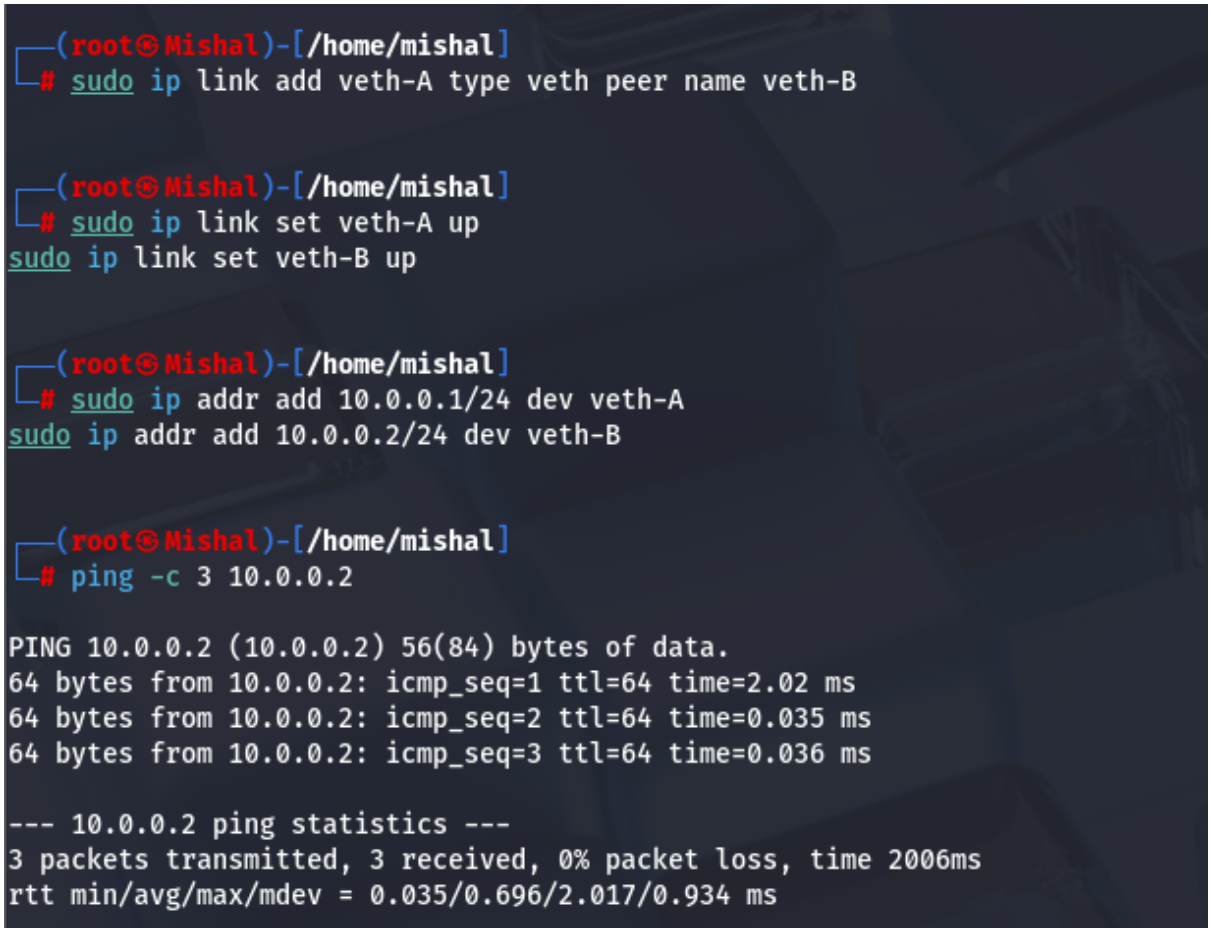
sudo ip link set veth-B up

sudo ip addr add 10.0.0.1/24 dev veth-A

sudo ip addr add 10.0.0.2/24 dev veth-B

Explanation:

- Creates a virtual test network to safely test MAC filtering and spoofing.
- veth pairs act as two separate devices for controlled experiments.



Observation:

- veth pair active and reachable with assigned IPs.

Step 11 — SNB: View Current MAC (veth interfaces)

Command:

sudo macchanger -s veth-A

sudo macchanger -s veth-B

Explanation:

- Records current MACs of virtual devices before applying random or custom MACs.
- Establishes a baseline for veth endpoints.

```
┌──(root㉿Mishal)-[/home/mishal]
└─# sudo macchanger -s veth-A
sudo macchanger -s veth-B

Current MAC:   6e:73:e7:73:6a:32 (unknown)
Permanent MAC: 00:00:00:00:00:00 (XEROX CORPORATION)
Current MAC:   3e:5b:fb:18:8b:9d (unknown)
Permanent MAC: 00:00:00:00:00:00 (XEROX CORPORATION)
```

Observation:

- Initial random MACs recorded for veth-A and veth-B.

Step 12 — SNB: Randomize MAC (veth)

Command:

sudo macchanger -r veth-A

sudo macchanger -r veth-B

Explanation:

- Assigns random MACs to veth endpoints for testing dynamic network behavior.
- Checks network stability under changing MAC addresses.

```
┌──(root㉿Mishal)-[/home/mishal]
└─# sudo macchanger -r veth-A
sudo macchanger -r veth-B

Current MAC:   6e:73:e7:73:6a:32 (unknown)
Permanent MAC: 00:00:00:00:00:00 (XEROX CORPORATION)
New MAC:       d6:00:27:51:52:14 (unknown)
Current MAC:   3e:5b:fb:18:8b:9d (unknown)
Permanent MAC: 00:00:00:00:00:00 (XEROX CORPORATION)
New MAC:       1a:32:bc:54:b9:a5 (unknown)
```

Observation:

- Random MACs successfully applied; endpoints continue to communicate.

Step 13 — SNB: Customize MAC (veth)

Command:

sudo macchanger -m 00:aa:bb:cc:dd:01 veth-A

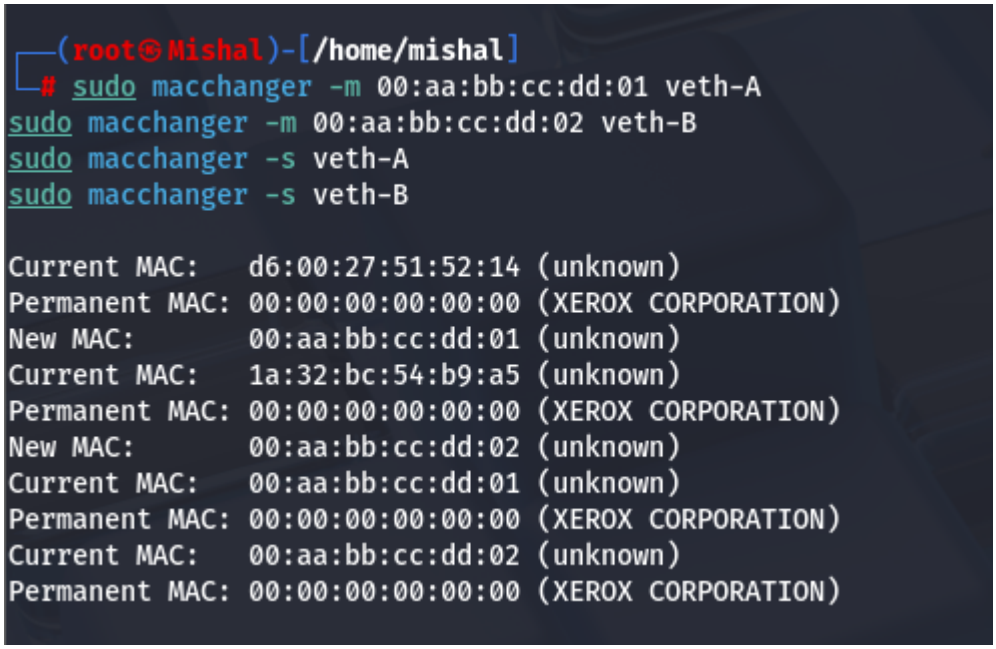
sudo macchanger -m 00:aa:bb:cc:dd:02 veth-B

sudo macchanger -s veth-A

sudo macchanger -s veth-B

Explanation:

- Sets specific MACs on veth endpoints to test MAC-based access control and filtering rules.
- Ensures custom MACs are correctly applied.



Observation:

- Custom MACs applied successfully; visible using macchanger -s.

Step 14 — SNB: Ping After Spoofing (veth)

Command:

ping -c 3 10.0.0.2

Explanation:

- Tests network connectivity between veth endpoints after MAC changes.
- Confirms spoofing does not interrupt communication in a controlled setup.

```
┌──(root㉿Mishal)-[/home/mishal]
└─# ping -c 3 10.0.0.2

PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.041 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.039 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.037 ms

--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2042ms
rtt min/avg/max/mdev = 0.037/0.039/0.041/0.001 ms
```

Observation:

- All packets received successfully; minimal RTT observed.

Step 15 — MAC Filtering (ebtables) and Bypass

Command:

sudo ebtables -F

sudo ebtables -A FORWARD -s 00:aa:bb:cc:dd:01 -j DROP

ping -c 3 10.0.0.2 # should fail

sudo macchanger -m 00:aa:bb:cc:dd:99 veth-A

ping -c 3 10.0.0.2 # should succeed

Explanation:

- ebtables blocks packets from a specific MAC to simulate a MAC filter.
- Spoofing MAC bypasses the filter, demonstrating potential risks in MAC-based security.

```
┌──(root㉿Mishal)-[/home/mishal]
└─# # Flush previous rules
sudo ebtables -F

# Block veth-A with its MAC in FORWARD chain
sudo ebtables -A FORWARD -s 00:aa:bb:cc:dd:01 -j DROP

# Test ping from veth-A to veth-B (should fail)
ping -c 3 10.0.0.2

# Spoof veth-A to a new MAC to bypass filter
sudo macchanger -m 00:aa:bb:cc:dd:99 veth-A
ping -c 3 10.0.0.2

PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.465 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.038 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.034 ms

--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2038ms
rtt min/avg/max/mdev = 0.034/0.179/0.465/0.202 ms
Current MAC:   00:aa:bb:cc:dd:99 (unknown)
Permanent MAC: 00:00:00:00:00:00 (XEROX CORPORATION)
New MAC:       00:aa:bb:cc:dd:99 (unknown)
It's the same MAC!!
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=3.06 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.039 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.037 ms

--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 0.037/1.045/3.060/1.424 ms
```

Observation:

- Original MAC blocked (0% packets received).
- Spoofed MAC bypasses the filter; connectivity restored.

Test Results: MAC Spoofing and Connectivity

Step	Interface / Device	Original MAC	New MAC	Ping / Connectivity	Observation
1	eth0	00:0c:29:93:6c:ab	Random fe:dc:84:fe:8b:7f	3/3 packets received	Random MAC applied successfully; connectivity maintained
2	eth0	00:0c:29:93:6c:ab	00:11:22:33:44:55	3/3 packets received	Custom MAC applied; verified with macchanger -s
3	eth0	00:11:22:33:44:55	02:12:34:56:78:9A	3/3 packets received	MAC spoofing works; ping successful
4	veth-A	6e:73:e7:73:6a:32	d6:00:27:51:52:14	3/3 packets received	Random MAC applied; connectivity stable
5	veth-B	3e:5b:fb:18:8b:9d	1a:32:bc:54:b9:a5	3/3 packets received	Random MAC applied; connectivity stable
6	veth-A	d6:00:27:51:52:14	00:aa:bb:cc:dd:01	3/3 packets received	Custom MAC applied; verified successfully
7	veth-B	1a:32:bc:54:b9:a5	00:aa:bb:cc:dd:02	3/3 packets received	Custom MAC applied; verified successfully
8	veth-A (MAC Filter)	00:aa:bb:cc:dd:01	00:aa:bb:cc:dd:01	0/3 packets received	MAC blocked by ebtables filter
9	veth-A (MAC Spoof)	00:aa:bb:cc:dd:99	00:aa:bb:cc:dd:99	3/3 packets received	Spoofed MAC bypassed filter; connectivity restored

7. Discussion

Interpretation of Results:

- The tests confirm that MAC Changer successfully randomized and spoofed MAC addresses on both physical and virtual interfaces.
- Network connectivity remained intact during most tests, demonstrating that spoofing does not inherently disrupt network communication.
- The ebtables MAC filter test verified that MAC-based access controls can be bypassed using MAC spoofing, highlighting potential security weaknesses in networks relying solely on MAC filtering.

Lessons Learned:

- MAC addresses can be easily changed and anonymized, emphasizing the need for additional security measures beyond MAC-based access control.
- Monitoring tools like arpwatch can detect MAC address changes but may require configuration to alert administrators effectively.
- Testing in a controlled lab environment allows safe evaluation of privacy and security techniques without affecting production networks.

Limitations:

- MAC Changer cannot bypass advanced authentication systems (e.g., 802.1X, WPA2 Enterprise) that do not rely solely on MAC addresses.
- Spoofing may be detected by network intrusion detection systems if logging or anomaly detection is enabled.
- Results are specific to the test network setup; behavior may vary in enterprise environments with complex security policies.

Ethical and Legal Considerations:

- MAC spoofing should only be performed in authorized environments, such as penetration testing labs or with explicit permission.
- Unauthorized use on production networks can violate laws and organizational policies.

- Ethical use involves testing network defenses, enhancing privacy, and understanding vulnerabilities responsibly.

8. Recommendations

Security Improvements / Hardening Measures:

- **Implement Multi-Factor Access Controls:** Do not rely solely on MAC filtering; use authentication protocols like 802.1X or WPA3 Enterprise to secure networks.
- **Enable MAC Change Detection:** Configure monitoring tools (e.g., arpwatch, IDS/IPS) to detect unexpected MAC address changes and alert administrators.
- **Segment Networks:** Use VLANs and network segmentation to limit the impact of unauthorized devices, even if MAC spoofing occurs.

Future Work / Enhancements:

- **Automated Testing Scripts:** Develop scripts to automate MAC spoofing tests and traffic monitoring to streamline security assessments.
- **Integration with Security Tools:** Combine MAC Changer with vulnerability scanners and network analyzers for comprehensive privacy and security audits.
- **User Awareness and Training:** Educate network administrators and users about MAC spoofing risks and detection techniques.

Mitigation Strategies for Potential Misuse:

- **Limit Administrative Privileges:** Only authorized personnel should have the ability to change MAC addresses on critical systems.
- **Log All MAC Changes:** Maintain logs of interface changes to trace suspicious activity.
- **Use Network Anomaly Detection:** Deploy systems to identify abnormal patterns such as repeated MAC changes or unexpected device appearances.

9. Conclusion

- The implementation and testing of MAC Changer in Kali Linux demonstrate its effectiveness in anonymizing MAC addresses, bypassing MAC-based access controls, and enhancing privacy on networks.
- Through lab experiments, it was observed that MAC spoofing can maintain network connectivity while masking the original hardware identity, highlighting both security benefits and potential risks if misused.
- The project reinforced the importance of combining MAC address anonymization with robust security measures, such as advanced authentication protocols and monitoring systems, to mitigate the risk of unauthorized access.
- From a strategic perspective, integrating MAC Changer into penetration testing and network security assessments allows organizations to better understand vulnerabilities related to MAC-based filtering, improve privacy measures, and maintain regulatory compliance.
- Overall, MAC Changer is a valuable tool for ethical cybersecurity practice, offering both educational and practical benefits in evaluating and improving network security.

10. References / Bibliography

1. Kali Linux Documentation. (n.d.). *MAC Changer Tool*. Retrieved from https://www.kali.org/tools/macchanger/
2. Nmap Network Scanning. (2021). *Network Security Auditing Techniques*. InsecBooks.
3. Bejtlich, R. (2013). *The Practice of Network Security Monitoring: Understanding Incident Detection and Response*. No Starch Press.
4. Cisco Systems. (2022). *MAC Address Filtering and Network Security Best Practices*. Retrieved from https://www.cisco.com
5. OWASP Foundation. (2023). *Network Security Testing Guides*. Retrieved from https://owasp.org
6. tcpdump/Libpcap. (n.d.). *Network Traffic Capture and Analysis*. Retrieved from https://www.tcpdump.org

Appendices

Appendix A – Lab Setup / Network Diagrams

- Kali Linux VM details, network interfaces (eth0, wlan0, veth-A, veth-B)
- Lab topology diagram showing connections, VLANs, and isolated test network

Appendix B – Test Logs

- Detailed logs from macchanger -s, tcpdump, arpwatch
- Include screenshots under each step to visually show MAC changes and network behavior

Appendix C – Commands Executed / Scripts

- sudo apt update && sudo apt install macchanger arpwatch tcpdump ebtables
- ip link show / ifconfig
- sudo macchanger -r <interface> / -m <custom-MAC> / -p <interface>
- ping -c 3 <IP>
- sudo ebtables -A FORWARD -s <MAC> -j DROP

Appendix D – Glossary of Terms

- **MAC Address:** Unique identifier assigned to a network interface.
- **MAC Spoofing:** Changing a network interface's MAC address to a different value.
- **ARP:** Address Resolution Protocol used to map IP addresses to MAC addresses.
- **EBtables:** Linux utility to filter Ethernet frames at the link layer.
- **Ping:** Command to test network connectivity between devices.