

FICHE SYNTHÈSE : ADRESSES MAC

Définition

MAC (Media Access Control) = Adresse physique unique attribuée à chaque interface réseau (carte réseau, Wi-Fi, Bluetooth, etc.)

- **Couche OSI** : Couche 2 (Liaison de données)
- **Portée** : Réseau local (LAN) uniquement
- **Taille** : 48 bits (6 octets)
- **Attribution** : Gravée par le fabricant (théoriquement unique au monde)

Structure de l'Adresse MAC

Format standard : 6 octets = 48 bits

AA:BB:CC:DD:EE:FF

NIC (Network Interface Controller)
Partie attribuée par le fabricant
16 777 216 adresses possibles (2^24)

OUI (Organizationally Unique Identifier)
Identifiant du fabricant (IEEE)
16 777 216 fabricants possibles (2^24)

Notations possibles

Format	Exemple	Utilisation
Deux-points	AA:BB:CC:DD:EE:FF	Linux, Cisco, BSD
Tirets	AA-BB-CC-DD-EE-FF	Windows
Points (par paire)	AABB.CCDD.EEFF	Cisco (certains équipements)
Sans séparateur	AABBCCDDEEFF	Programmation

Bits spéciaux (premier octet)

7

6

5

4

3

2

1

0

(bits)

I/G (Individual/Group)
U/L (Universal/Local)

- **Bit 0 (I/G)** :

- 0 = Unicast (adresse individuelle)
- 1 = Multicast/Broadcast
- **Bit 1 (U/L) :**
 - 0 = Universally administered (fabricant)
 - 1 = Locally administered (modifiée localement)

Types d'Adresses MAC

1. Unicast (communication point-à-point)

```
00:1A:2B:3C:4D:5E (bit 0 = 0)
```

Destinée à une seule interface réseau.

2. Multicast (communication un-vers-plusieurs)

```
01:00:5E:XX:XX:XX (bit 0 = 1)
```

Exemples :

- 01:00:5E:00:00:01 → Tous les hôtes sur le sous-réseau
- 33:33:XX:XX:XX:XX → Multicast IPv6

3. Broadcast (communication un-vers-tous)

```
FF:FF:FF:FF:FF:FF
```

Tous les bits à 1. Envoyé à **toutes** les machines du segment réseau.

OUI : Identifier le Fabricant

Les 3 premiers octets identifient le fabricant (OUI).

Exemples courants

OUI	Fabricant
00:50:56	VMware
08:00:27	VirtualBox
00:0C:29	VMware
D8:9E:F3	Apple

OUI	Fabricant
B8:27:EB	Raspberry Pi Foundation
00:1B:44	Cisco Systems
00:15:5D	Microsoft Hyper-V

Rechercher un fabricant

En ligne :

- <https://www.wireshark.org/tools/oui-lookup.html>
- <https://maclookup.app/>
- <https://macvendors.com/>

En ligne de commande :

```
# Avec curl
curl -s "https://api.macvendors.com/AA:BB:CC:DD:EE:FF"

# Avec arp-scan (Linux)
arp-scan --localnet
```

Commandes pour Afficher l'Adresse MAC

Linux

```
# Méthode 1 : ip
ip link show
ip addr show
ip link show eth0

# Méthode 2 : ifconfig (obsolète mais encore utilisé)
ifconfig
ifconfig eth0

# Méthode 3 : Fichier système
cat /sys/class/net/eth0/address

# Toutes les interfaces
for i in $(ls /sys/class/net); do
    echo "$(basename $i): $(cat $i/address)";
done
```

Windows

```
# Commande ipconfig
ipconfig /all

# Commande getmac
getmac
getmac /v

# PowerShell
Get-NetAdapter | Select-Object Name, MacAddress
```

macOS

```
# ifconfig
ifconfig

# networksetup
networksetup -listallhardwareports

# system_profiler
system_profiler SPNetworkDataType
```

Aspects Sécurité

1. MAC Spoofing (Usurpation d'adresse MAC)

Définition : Modifier l'adresse MAC d'une interface réseau pour se faire passer pour une autre machine.

Motivations :

- Contourner le filtrage MAC
- Cacher son identité
- Attaques Man-in-the-Middle
- Contourner des restrictions d'accès (ex: Wi-Fi gratuit limité)

Changer son adresse MAC

Linux :

```
# Méthode 1 : ifconfig
sudo ifconfig eth0 down
sudo ifconfig eth0 hw ether AA:BB:CC:DD:EE:FF
sudo ifconfig eth0 up

# Méthode 2 : ip
sudo ip link set dev eth0 down
sudo ip link set dev eth0 address AA:BB:CC:DD:EE:FF
```

```
sudo ip link set dev eth0 up

# Méthode 3 : macchanger
sudo macchanger -m AA:BB:CC:DD:EE:FF eth0 # Adresse spécifique
sudo macchanger -r eth0                    # Aléatoire
sudo macchanger -p eth0                    # Restaurer l'originale
```

Windows :

```
# Gestionnaire de périphériques → Propriétés carte réseau
# Onglet "Avancé" → "Network Address" ou "Locally Administered Address"

# Ou via le registre (nécessite redémarrage)
```

macOS :

```
sudo ifconfig en0 ether AA:BB:CC:DD:EE:FF
```

2. Filtrage MAC (MAC Filtering)

Principe : Les switches/routeurs/points d'accès autorisent uniquement les adresses MAC configurées.

Limites de sécurité :

-  Facilement contournable par MAC spoofing
-  Gestion complexe dans les grands réseaux
-  Adresses MAC visibles en clair (sniffing)
-  Utilisable comme **couche supplémentaire** (pas unique)

3. Détection de MAC Spoofing

```
# Wireshark : détecter plusieurs IP avec la même MAC
eth.addr == AA:BB:CC:DD:EE:FF

# arp-scan : scanner le réseau
sudo arp-scan --localnet

# arping : vérifier une adresse spécifique
sudo arping -c 3 192.168.1.1

# Vérifier les duplicatas dans le cache ARP
arp -a | sort -k 4
```

4. Randomisation MAC (Privacy)

Certains systèmes modernes (smartphones, Windows 10+) randomisent l'adresse MAC pour protéger la vie privée lors des connexions Wi-Fi.

Android/iOS : Adresse MAC aléatoire par défaut pour chaque réseau Wi-Fi.

Windows 10/11 :

```
# Activer/désactiver
Paramètres → Réseau → Wi-Fi → Propriétés réseau →
"Utiliser des adresses matérielles aléatoires"
```

Différences MAC vs IP

Caractéristique	Adresse MAC	Adresse IP
Couche OSI	2 (Liaison)	3 (Réseau)
Portée	Réseau local	Internet (global)
Taille	48 bits	32 bits (IPv4) / 128 bits (IPv6)
Format	Hexadécimal	Décimal (IPv4) / Hexa (IPv6)
Attribution	Fabricant (physique)	Administrateur réseau (logique)
Modifiable	Oui (via logiciel)	Oui (configuration)
Routage	Non routable	Routable
Broadcast	FF:FF:FF:FF:FF:FF	255.255.255.255 (IPv4)

Script Python : Analyser les MAC

```
import re
from scapy.all import *

def get_mac(ip):
    """Obtenir l'adresse MAC d'une IP via ARP"""
    arp_request = ARP(pdst=ip)
    broadcast = Ether(dst="ff:ff:ff:ff:ff:ff")
    packet = broadcast / arp_request
    answered = srp(packet, timeout=2, verbose=False)[0]

    if answered:
        return answered[0][1].hwsrc
    return None

def is_local_mac(mac):
    """Vérifier si MAC est locally administered"""
    first_octet = int(mac.split(':')[0], 16)
```

```
    return bool(first_octet & 0x02) # Bit 1 = 1

def is_multicast(mac):
    """Vérifier si MAC est multicast"""
    first_octet = int(mac.split(':')[0], 16)
    return bool(first_octet & 0x01) # Bit 0 = 1

def get_oui(mac):
    """Extraire l'OUI (3 premiers octets)"""
    return ':'.join(mac.split(':')[3]).upper()

# Exemple d'utilisation
mac = "AA:BB:CC:DD:EE:FF"
print(f"OUI: {get_oui(mac)}")
print(f"Local: {is_local_mac(mac)}")
print(f"Multicast: {is_multicast(mac)}")
```

Points Clés à Retenir

- ✓ **48 bits** = 6 octets en hexadécimal
- ✓ **3 premiers octets** = OUI (fabricant)
- ✓ **3 derniers octets** = identifiant unique de l'interface
- ✓ **Unicast** : bit 0 = 0
- ✓ **Multicast** : bit 0 = 1
- ✓ **Broadcast** : FF:FF:FF:FF:FF:FF
- ✓ **Couche 2** : ne traverse pas les routeurs
- ✓ **MAC Spoofing** : facilement modifiable (faible sécurité)
- ✓ Utilisé par **ARP** pour lier IP ↔ MAC

Cas d'Usage Pratiques

Scénario	Utilisation MAC
Switch	Table CAM : associe ports ↔ MAC
ARP	Résolution IP → MAC
DHCP	Réservation IP basée sur MAC
Wake-on-LAN	Réveil machine via paquet magique
Filtrage	Autorisation/blocage par MAC
Forensics	Traçabilité des équipements

Ressources

- **IEEE OUI Database** : <https://standards.ieee.org/products-services/regauth/>
- **RFC 7042** : IANA Considerations for Ethernet Addresses

- **Wireshark OUI Lookup** : <https://www.wireshark.org/tools/oui-lookup.html>
- **Man pages** : `ip(8)`, `ifconfig(8)`, `arp(8)`