

# Paramétrage réseau

- [Obtenir des informations sur le réseau de la machine](#)
- [Configuration IP et interfaces sur Ubuntu \(≥20.04\)](#)
- [DNS](#)
- [Pare-Feu](#)

# Obtenir des informations sur le réseau de la machine

Pour afficher les adresses IP et MAC des cartes réseaux :

```
ip a
```

Pour afficher les serveurs DNS utilisé :

```
cat /run/systemd/resolve/resolv.conf
```

Pour obtenir des informations avancé sur les cartes réseaux :

```
sudo lshw -class network
```

# Configuration IP et interfaces sur Ubuntu ( $\geq 20.04$ )

Ubuntu à partir de Ubuntu 20.04 (environ) utilise Netplan pour la configuration IP. Netplan s'appuie sur des fichiers YAML situé dans `/etc/netplan/`.

Pour appliquer les modification du fichier, exécuter :

```
$ sudo netplan apply
```

## Usage courant

Exemple de fichier pour une configuration via DHCPv4 :

```
network:
  ethernets:
    enp3s0:
      dhcp4: true
  version: 2
```

Exemple de fichier pour une configuration IPv4 statique :

```
network:
  version: 2
  ethernets:
    eth0:
      addresses:
        - 10.10.10.2/24
      routes:
        - to: default
          via: 10.10.10.1
      nameservers:
        search: [mydomain, otherdomain]
        addresses: [10.10.10.1, 1.1.1.1]
```

Exemple de fichier pour une configuration IPv4 via DHCP et IPv6 statique :

```
network:
  ethernets:
    ens18:
      dhcp4: true
      dhcp6: false
      addresses:
        - 2a01:9093:fd2e:5a0e:7714:59ce:0f8a:4/64
      #gateway6: fe80::7e2b:1302:814a:ea68 # Obsolete
      routes:
        - to: "::/0" # default ipv6
          via: "fe80::7e2b:1302:814a:ea68"
          on-link: true

version: 2
```

**on-link: true** indique que la passerelle spécifiée (**via**) est directement accessible sur le lien local (le réseau local). Cela signifie que l'interface réseau n'a pas besoin de savoir exactement où se trouve la passerelle par l'intermédiaire d'une autre route.

# Usage avancé

## LACP

Exemple de fichier pour lier deux interfaces via LACP :

```
network:
  version: 2
  renderer: networkd
  ethernets:
    eno1:
      dhcp4: no
    eno2:
      dhcp4: no
  bonds:
    bond0:
      interfaces: [eno1, eno2]
```

```
addresses: [192.168.2.2/24]
# gateway4: 192.186.2.1 # Obsolete
routes:
- to: "0.0.0.0/0" # Route par défaut pour tout le trafic IPv4
  via: 192.168.2.1
parameters:
  mode: 802.3ad
  transmit-hash-policy: layer3+4
  mii-monitor-interval: 1
nameservers:
  addresses:
    - "8.8.8.8"
    - "9.9.9.9"
```

## Paramètres du bond :

- **mode: 802.3ad** : Il s'agit du mode d'agrégation **LACP (Link Aggregation Control Protocol)**, conforme à la norme IEEE 802.3ad. Ce mode est utilisé pour équilibrer la charge et offrir la tolérance aux pannes si le commutateur réseau supporte LACP.
- **transmit-hash-policy: layer3+4** : Cette politique d'agrégation hash sur la base des informations des couches 3 (IP) et 4 (port TCP/UDP), ce qui permet un équilibrage de charge plus efficace.
- **mii-monitor-interval: 1** : Cela configure la fréquence de surveillance de l'état des interfaces membres du bond, ici toutes les 1 ms. Si une interface tombe, elle sera retirée du bond et le trafic sera redirigé via les autres interfaces.

# VLAN

## Prérequis

Pour pouvoir utiliser les tags vlan et configurer nos cartes réseau pour les utiliser avec Netplan, il faut d'abord s'assurer que le module VLAN (**8021q**) est chargé dans le noyau Linux.

Pour vérifier si le module VLAN (8021q) est chargé sur un système Linux, utilisez la commande ``lsmod`` pour lister les modules du noyau chargés. Ouvrez un terminal et exécutez :

```
$ lsmod | grep 8021q
```

Si le module VLAN est chargé, vous verrez "8021q" dans le résultat. Sinon, cela signifie que le module n'est pas chargé.

Si le module n'est pas chargé, vous pouvez le faire manuellement avec la commande ``modprobe`` :

```
$ sudo modprobe 8021q
```

Après cela, vérifiez à nouveau son statut avec ``lsmod | grep 8021q``. Si le module est chargé, il apparaîtra dans la liste.

Pour que le module VLAN (8021q) soit chargé automatiquement au démarrage, ajoutez-le au fichier ``/etc/modules`` :

```
$ echo '8021q' | sudo tee -a /etc/modules
```

Cela garantira que le module est chargé à chaque démarrage du système.

## Exemples de configurations

Exemple de fichier pour configurer une interface physique et une interface VLAN :

```
network:
  version: 2
  renderer: networkd
  ethernets:
    ens18:
      dhcp4: no
      addresses:
        - 192.168.1.40/24
      routes:
        - to: default
          via: 192.168.1.101
      nameservers:
        addresses: [8.8.8.8, 8.8.4.4]

  vlans:
    vlan10:
      id: 10
      link: ens18
      addresses: [192.168.10.2/24]
```

# DNS

## Sous Linux Mint

Pour vider le cache DNS

```
resolvectl flush-caches
```

Pour consulter l'état du cache :

```
resolvectl statistics
```

# Pare-Feu

Configuration d'un pare-feu sous Linux.

Configurer un Pare Feu à distance (via SSH par exemple) peut empêcher la connexion. Il est important de configurer les règles afin de ne pas se bloquer soi même.

## UFW

UFW pour Uncomplicated Firewall est préinstallé sur Ubuntu mais peut être installé avec APT si besoin et sur d'autres distros.

## Commandes utiles

Activer ou désactiver UFW :

```
sudo ufw enable
```

Vérifier l'état et les règles de UFW :

```
sudo ufw status  
sudo ufw status verbose
```

Vérifier les règles existantes quand le pare feu n'est pas actif (pratique pour vérifier si on pourra encore accéder au serveur en SSH après activation) :

```
sudo ufw show added
```

## Ajouter des règles

UFW permet d'ajouter des règles efficacement via une syntaxe simplifiée. Par exemple, UFW admet par défaut qu'on crée une règle pour le trafic entrant.

### Règles simples



Ajouter une règle simple :

```
sudo ufw allow [port/protocole]
sudo ufw allow 22/tcp

sudo ufw deny 56/tcp
```

Sans précision du protocole, la règle s'appliquera à TCP et UDP.

## Règles plus complexes

Ajouter une règle sur une plage de ports :

```
sudo ufw allow/deny [Port_début:Port_fin]/protocole
```

Ajouter une règle associé à une adresse IP de machine ou de réseau :

```
sudo ufw deny from 192.168.1.2
sudo ufw deny from 192.168.1.0/24
```

Ajouter une règle liée à une adresse IP et un port :

```
sudo ufw allow from [ADRESSE_IP] to any port [PORT]
sudo ufw allow from [ADRESSE_IP] to any port [PORT] proto tcp
```

Ici, "any" fait référence à "n'importe quelle adresse IP de CETTE machine"

Ajouter une règle à une position précise :

```
sudo ufw insert 2 deny proto udp to any port 514 from 1.2.3.4
```

## Supprimer des règles

Pour supprimer des règles il faut d'abord les afficher avec une numérotation puis supprimer la règle via son numéro :

```
sudo ufw status numbered

sudo ufw delete 4
```

# Sources

- <https://www.hostinger.com/fr/tutoriels/comment-configurer-pare-feu-ufw>
- <https://doc.ubuntu-fr.org/ufw>