Paramétrage réseau

- Obtenir des informations sur le réseau de la machine
- Configuration IP et interfaces sur Ubuntu (⩾20.04)
- DNS
- Pare-Feu
- Configuration réseau sous Debian 13

Obtenir des informations sur le réseau de la machine

Pour afficher les adresses IP et MAC des cartes réseaux :

```
ip a
```

Pour afficher les IP plus clairement et avec de la couleur :

```
ip -br -c 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 (⩾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"    # defaut 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 simplifié. Par exemple, UFW admet par défaut qu'on créer une règle pour le trafique 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

Configuration réseau sous Debian 13

Configuration générale

Debian 13 (Trixie) utilise généralement ifupdown pour gérer la configuration réseau sur les installations serveur traditionnelles.

La configuration se fait principalement dans le fichier `/etc/network/interfaces`. Il est également possible de répartir les configurations dans plusieurs fichiers situés dans `/etc/network/interfaces.d/`.

Pour vérifier le nom des interfaces réseau :

```
ip link show
```

Pour afficher les adresses IP et les routes :

```
ip addr show
ip route show
ip -6 route show
```

Pour vérifier que le service réseau est actif :

```
systemctl status networking
```

Application des modifications

Les interfaces peuvent être activées et désactivées avec `ifup` et `ifdown`.

Par exemple, pour appliquer une modification à l'interface `enp3s0` :

```
sudo ifdown enp3s0
sudo ifup enp3s0
```

Pour redémarrer l'ensemble du service réseau :

```
sudo systemctl restart networking
```

Attention : ces commandes peuvent interrompre une connexion SSH. Il est préférable de disposer d'un accès console ou hors bande avant de modifier la configuration réseau à distance.

Usage courant

La configuration de base se trouve dans `/etc/network/interfaces`.

Voici un exemple de configuration classique :

```
# Interface loopback
auto lo
iface lo inet loopback

# Configuration réseau
allow-hotplug enp3s0
iface enp3s0 inet dhcp
```

La directive `allow-hotplug` permet d'activer l'interface lorsqu'elle est détectée par le système.

Pour une interface qui doit impérativement être activée au démarrage, on peut utiliser `auto` à la place.

Configuration IPv4 statique

Exemple de configuration avec une adresse IPv4 statique :

```
auto enp3s0
iface enp3s0 inet static
    address 10.10.10.2/24
    gateway 10.10.10.1
    dns-nameservers 10.10.10.1 1.1.1.1
    dns-search mydomain otherdomain
```

Les directives principales :

- `address` : adresse IP et masque de sous-réseau (notation CIDR).

- `gateway` : passerelle par défaut.
- `dns-nameservers` : serveurs DNS.
- `dns-search` : domaines de recherche DNS.

Attention concernant le DNS : les directives `dns-nameservers` et `dns-search` nécessitent généralement le paquet `resolvconf` pour être prises en compte par ifupdown. Sans celui-ci, il faut gérer `/etc/resolv.conf` avec le gestionnaire DNS utilisé sur le système.

Installation éventuelle :

```
sudo apt install resolvconf
```

Configuration IPv4 via DHCP et IPv6 statique

Exemple avec une adresse IPv4 obtenue par DHCP et une adresse IPv6 statique :

```
auto enp3s0
iface enp3s0 inet dhcp

iface enp3s0 inet6 static
    address 2a01:9093:fd2e:5a0e:7714:59ce:0f8a:4/64
    gateway fe80::7e2b:1302:814a:ea68
```

La configuration IPv4 et IPv6 peut être définie indépendamment sur une même interface.

Dans cet exemple, la passerelle IPv6 utilise une adresse link-local (`fe80::`). Ce type d'adresse est limité au lien réseau local et nécessite que la passerelle soit accessible sur l'interface concernée.

Pour consulter les routes IPv6 :

```
ip -6 route show
```

Usage avancé

LACP (Link Aggregation)

Il est possible de regrouper plusieurs interfaces physiques dans une interface logique grâce au mécanisme de bonding du noyau Linux.

Pour utiliser le bonding avec ifupdown, installer le paquet `ifenslave` :

```
sudo apt install ifenslave
```

Exemple de configuration pour agréger deux interfaces physiques (`eno1` et `eno2`) en utilisant LACP :

```
auto eno1
iface eno1 inet manual

auto eno2
iface eno2 inet manual

auto bond0
iface bond0 inet static
    address 192.168.2.2/24
    gateway 192.168.2.1

    bond-slaves eno1 eno2
    bond-mode 802.3ad
    bond-miimon 100
    bond-lacp-rate 1
    bond-xmit-hash-policy layer3+4

    dns-nameservers 8.8.8.8 9.9.9.9
```

Paramètres du bond

- `bond-slaves eno1 eno2` : définit les interfaces physiques qui constituent le bond.
- `bond-mode 802.3ad` : active le mode d'agrégation LACP, conforme à la norme IEEE 802.3ad.
- `bond-miimon 100` : vérifie l'état des interfaces membres toutes les 100 ms. Une interface défaillante est retirée du bond.
- `bond-lacp-rate 1` : demande une émission rapide des paquets LACP (intervalle nominal d'une seconde).
- `bond-xmit-hash-policy layer3+4` : utilise les adresses IP et les ports TCP/UDP pour déterminer la répartition du trafic entre les interfaces.

Attention : le commutateur réseau doit également être configuré pour utiliser LACP sur les ports concernés (généralement dans un même agrégat ou port-channel).

Pour vérifier l'état du bond :

```
cat /proc/net/bonding/bond0
```

On y retrouve notamment le mode d'agrégation, l'état des interfaces membres et l'état de la négociation LACP.

VLAN

Prérequis

Pour gérer les VLAN 802.1Q avec ifupdown, installer le paquet `vlan` :

```
sudo apt install vlan
```

Le module noyau `8021q` est généralement chargé automatiquement lorsque nécessaire.

Pour vérifier s'il est chargé :

```
lsmod | grep 8021q
```

Pour le charger manuellement :

```
sudo modprobe 8021q
```

Pour le charger automatiquement au démarrage :

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

Exemples de configurations

Exemple de configuration d'une interface physique et d'une interface VLAN :

```
auto enp3s0
iface enp3s0 inet manual

auto enp3s0.10
iface enp3s0.10 inet static
    address 192.168.10.2/24
    vlan-raw-device enp3s0
```

Dans cet exemple :

- `enp3s0` est l'interface physique.
- `enp3s0.10` est l'interface virtuelle associée au VLAN 10.
- `vlan-raw-device` indique l'interface physique sur laquelle les trames VLAN sont transportées.

Il est également possible de configurer une interface VLAN avec DHCP :

```
auto enp3s0.20
iface enp3s0.20 inet dhcp
    vlan-raw-device enp3s0
```

Remarque : le nommage `interface.VLAN_ID` est une convention courante sous Linux. Il est également possible d'utiliser d'autres noms d'interfaces.

VLAN sur un bond LACP

Il est possible de combiner le bonding et les VLAN. Dans ce cas, les VLAN sont créés sur l'interface logique `bond0`, et non directement sur les interfaces physiques.

Exemple avec un bond LACP et un VLAN 10 :

```
auto eno1
iface eno1 inet manual

auto eno2
iface eno2 inet manual

auto bond0
iface bond0 inet manual
    bond-slaves eno1 eno2
    bond-mode 802.3ad
    bond-miimon 100
    bond-lacp-rate 1
    bond-xmit-hash-policy layer3+4

auto bond0.10
iface bond0.10 inet static
    address 192.168.10.2/24
    gateway 192.168.10.1
    vlan-raw-device bond0
```

Cette configuration permet de transporter plusieurs VLAN sur un même agrégat réseau.

Le commutateur doit être configuré en conséquence :

- Les ports physiques doivent appartenir au même agrégat LACP.
- Le port-channel doit être configuré en trunk.
- Les VLAN nécessaires doivent être autorisés sur ce trunk.

Outils de diagnostic

Quelques commandes utiles pour diagnostiquer la configuration réseau :

Commande	Description
`ip link show`	Liste les interfaces réseau
`ip addr show`	Affiche les adresses IP
`ip route show`	Affiche les routes IPv4
`ip -6 route show`	Affiche les routes IPv6
`ip -d link show`	Affiche les détails des interfaces VLAN et bonds
`bridge vlan show`	Affiche les VLAN configurés sur les bridges
`cat /proc/net/bonding/bond0`	Affiche l'état du bond
`ethtool enp3s0`	Affiche les informations de liaison Ethernet
`journalctl -u networking`	Consulte les journaux du service réseau

Pour vérifier les interfaces configurées par ifupdown :

```
ifquery --list
```

Pour afficher la configuration d'une interface :

```
ifquery enp3s0
```

Remarques

Debian propose plusieurs gestionnaires réseau. Il est important de ne pas faire gérer une même interface simultanément par plusieurs gestionnaires.

Gestionnaire	Configuration
ifupdown	`/etc/network/interfaces`
systemd-networkd	`/etc/systemd/network/`
NetworkManager	`/etc/NetworkManager/`
Netplan	`/etc/netplan/`

Sur Debian 13, Netplan peut notamment être présent sur certaines images cloud. NetworkManager est également courant sur les installations Desktop. Les exemples de cette page concernent exclusivement ifupdown.

Pour davantage d'informations :

- Documentation Debian – Configuration réseau.
- Manuel interfaces(5) – ifupdown.
- Documentation Debian – Bonding.
- Documentation du noyau Linux – Ethernet Bonding.

Note technique : contrairement à Netplan, ifupdown utilise une syntaxe déclarative proche de celle d'un fichier INI, avec des blocs `iface` et des directives indentées. Il n'y a donc pas de commande équivalente à `netplan apply` : les interfaces doivent être activées ou désactivées pour prendre en compte les changements.