Commandes réseau sous Linux

Réseau

1 Introduction

Les systèmes d'exploitation de type UNIX (Linux, macOS, BSD) fournissent un ensemble de commandes en ligne permettant de configurer, tester et diagnostiquer le réseau. Ces outils sont indispensables à tout administrateur système, mais aussi à quiconque cherche à comprendre concrètement le fonctionnement des réseaux étudiés en cours.

Toutes les commandes présentées ici s'exécutent dans un **terminal** (aussi appelé *shell*). Sous Linux, on ouvre un terminal avec le raccourci `Ctrl + Alt + T` ou via le menu des applications.

Remarque. Certaines commandes historiques (`ifconfig`, `netstat`, `route`) sont progressivement remplacées par des alternatives plus modernes (`ip`, `ss`). Nous présenterons les commandes modernes en priorité, avec une mention des commandes historiques encore très répandues.

2 Afficher la configuration réseau : `ip a`

2.1 Principe

La commande `ip a` (abréviation de `ip addr show`) affiche la liste de toutes les interfaces réseau de la machine ainsi que leurs adresses IP, masques de sous-réseau et adresses MAC.

2.2 Exemple d'utilisation

```
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536
    link/loopback 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500
    link/ether 4c:ed:fb:a2:33:7e
    inet 192.168.1.42/24 brd 192.168.1.255 scope global enp3s0
    inet6 fe80::4eed:fbff:fea2:337e/64 scope link
3: wlp2s0: <BROADCAST,MULTICAST> mtu 1500
    link/ether 60:f2:62:c4:11:a0
```

2.3 Lecture du résultat

Chaque bloc correspond à une interface réseau :

— `lo` est l'interface de *loopback* (boucle locale). Son adresse est toujours `127.0.0.1`. Elle permet à la machine de communiquer avec elle-même ;

— `enp3s0` est une interface Ethernet filaire. On y lit l'adresse MAC (`4c:ed:fb:a2:33:7e`), l'adresse IPv4 (`192.168.1.42/24`) et l'adresse de diffusion (`192.168.1.255`) ;

— `wlp2s0` est une interface Wi-Fi. Ici, elle n'a pas d'adresse IP : elle n'est pas connectée.

L'**adresse de diffusion** est la dernière adresse du réseau : tous les bits de la partie hôte y valent 1. Un paquet envoyé à cette adresse est reçu par toutes les machines du réseau local. Comme l'adresse réseau (tous les bits de la partie hôte à 0), elle ne peut être attribuée à aucune machine. Un réseau dont la partie hôte compte h bits accueille donc au plus $2^h - 2$ machines, soit 254 pour un masque `/24`.

Remarque. `ifconfig` affiche des informations similaires. Elle reste disponible sur la plupart des systèmes mais est considérée comme obsolète sur les distributions Linux récentes.

Exercice 1 *Configuration réseau* Exécutez `ip a` sur votre machine (ou sur le simulateur de terminal). Identifiez le nom de chaque interface, son adresse IP et son masque de sous-réseau. Quelle est l'adresse réseau associée à chaque interface ?

③ Tester la connectivité : `ping`

3.1 Principe

La commande `ping` envoie des paquets ICMP (*Internet Control Message Protocol*) à une machine distante et attend une réponse. Elle permet de vérifier que la communication avec une machine distante est possible et de mesurer le temps d'aller-retour (*round-trip time*, ou RTT).

`ping` fonctionne au niveau de la **couche internet** du modèle TCP/IP.

3.2 Exemple d'utilisation

```
$ ping -c 4 192.168.1.1
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=64 time=1.23 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=64 time=0.98 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=64 time=1.05 ms
64 bytes from 192.168.1.1: icmp_seq=4 ttl=64 time=1.12 ms

--- 192.168.1.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 0.980/1.095/1.230/0.091 ms
```

3.3 Lecture du résultat

Pour chaque paquet envoyé, on obtient :

— `icmp_seq` : le numéro de séquence du paquet ;

— `ttl` (*time to live*) : le nombre maximal de routeurs que le paquet peut encore traverser avant d'être détruit ;

— `time` : le temps d'aller-retour en millisecondes.

L'option `-c 4` limite l'envoi à 4 paquets. Sans cette option, `ping` continue indéfiniment sous Linux (on l'arrête avec `Ctrl + C`).

3.4 Diagnostic

Situation	Interprétation possible
Réponse normale	La machine distante est joignable
`Destination Host Unreachable`	La machine visée ne répond pas sur son réseau local : elle est éteinte, débranchée, ou l'adresse n'existe pas
`Network is unreachable`	Pas de route vers la destination (par exemple, aucune passerelle configurée)
Aucune réponse, puis `100% packet loss`	Les paquets partent mais aucune réponse ne revient : un pare-feu bloque le ping, ou la machine, sur un réseau distant, est injoignable
Perte partielle de paquets	Liaison instable ou saturée

Exercice 2 *Ping et diagnostic*

1. Exécutez `ping -c 4 127.0.0.1`. Que testez-vous exactement ? Quel TTL obtenez-vous et pourquoi ?
2. Exécutez `ping -c 4` suivi de l'adresse de votre passerelle par défaut (souvent `192.168.1.1`). Interprétez le résultat.

④ Tracer le chemin d'un paquet : `traceroute`

4.1 Principe

La commande `traceroute` affiche la liste de tous les routeurs traversés par un paquet entre la machine locale et une destination.

Pour comprendre son fonctionnement, il faut d'abord rappeler le rôle du champ **TTL** (*time to live*) des paquets IP. Ce champ existe pour empêcher un paquet de circuler indéfiniment dans le réseau en cas d'erreur de routage : chaque routeur traversé décrémente le TTL de 1, et si celui-ci atteint 0, le routeur détruit le paquet et renvoie un message d'erreur ICMP *Time Exceeded* à l'expéditeur.

`traceroute` détourne ce mécanisme de protection : elle envoie un premier paquet avec TTL = 1, qui est donc détruit par le premier routeur, lequel révèle son adresse IP dans le message d'erreur. Puis elle envoie un paquet avec TTL = 2, détruit par le deuxième routeur, et ainsi de suite jusqu'à atteindre la destination.

Pour chaque routeur, `traceroute` tente ensuite une résolution DNS inverse (de l'adresse IP vers un nom de domaine). Si cette résolution aboutit, le nom du routeur est affiché à côté de son adresse IP (par exemple `lag-10.nantes.francetelecom.net`) ; sinon, seule l'adresse IP apparaît.

4.2 Exemple d'utilisation

Sortie illustrative. Le site du ministère est aujourd'hui servi par un réseau de diffusion de contenu (Cloudflare) : l'adresse IP obtenue et le chemin dépendent du lieu d'où l'on interroge, et certains routeurs ne répondent pas (lignes d'astérisques).

```
$ traceroute www.education.gouv.fr
traceroute to www.education.gouv.fr (185.75.143.36), 30 hops max
 1  box (192.168.1.1)  1.234 ms  1.102 ms  1.089 ms
 2  10.45.128.1  5.678 ms  5.432 ms  5.501 ms
 3  lag-10.nantes.francetelecom.net  12.34 ms  12.12 ms  12.45 ms
 4  ae41.paris1.francetelecom.net  18.56 ms  18.23 ms  18.67 ms
 5  185.75.143.36  19.12 ms  18.98 ms  19.23 ms
```

4.3 Lecture du résultat

Chaque ligne correspond à un routeur intermédiaire (un *hop*). On y trouve le numéro du saut, le nom et l'adresse IP du routeur (quand disponible) et trois mesures de temps d'aller-retour.

Des étoiles `* * *` indiquent qu'un routeur n'a pas répondu (pare-feu ou configuration ICMP restrictive).

Remarque. Sous Windows, la commande équivalente est `tracert`. Elle utilise des paquets ICMP au lieu d'UDP.

Exercice 3 *Analyse de traceroute* En observant la sortie ci-dessus, combien de routeurs le paquet traverse-t-il ? Identifiez le routeur qui introduit le plus grand saut de latence et proposez une explication.

⑤ Interroger le DNS : `nslookup` et `dig`

5.1 Principe

Le **DNS** (*Domain Name System*) est le service qui traduit les noms de domaine (comme `www.education.gouv.fr`) en adresses IP. Les commandes `nslookup` et `dig` permettent d'interroger manuellement un serveur DNS.

5.2 Exemple avec `nslookup`

```
$ nslookup www.education.gouv.fr
Server:     192.168.1.1
Address:    192.168.1.1#53

Non-authoritative answer:
Name:   www.education.gouv.fr
Address: 185.75.143.36
```

Le résultat indique le serveur DNS interrogé (`192.168.1.1`, ici la box internet, sur le port 53), l'adresse IP associée au nom de domaine et la mention *non-authoritative answer* (la réponse provient du cache du serveur DNS).

5.3 Exemple avec `dig`

```
$ dig www.education.gouv.fr +short
185.75.143.36
```

`dig` est plus complet que `nslookup` et donne davantage de détails techniques. L'option `+short` permet d'obtenir uniquement l'adresse IP.

Remarque. La résolution DNS intervient au niveau de la **couche application** du modèle TCP/IP. Le protocole DNS utilise le port UDP 53 par défaut.

Exercice 4 *Résolution DNS*

1. Utilisez `nslookup` pour trouver l'adresse IP de `www.python.org`. Quel serveur DNS a répondu ?
2. Utilisez `dig www.python.org` (sans `+short`). Identifiez le type d'enregistrement DNS retourné (champ `A` pour une adresse IPv4).

6 Examiner les connexions réseau : `ss`

6.1 Principe

La commande `ss` (*socket statistics*) affiche les connexions réseau actives, les ports en écoute et les protocoles utilisés. Elle remplace l'ancienne commande `netstat`.

6.2 Options courantes

Option	Signification
`-t`	Connexions TCP uniquement
`-u`	Connexions UDP uniquement
`-l`	Ports en écoute (*listening*) uniquement
`-n`	Afficher les numéros de port au lieu des noms de service
`-p`	Afficher le processus associé à chaque connexion

6.3 Exemple d'utilisation

```
$ ss -tln
State    Recv-Q   Send-Q   Local Address:Port   Peer Address:Port
LISTEN   0        128      0.0.0.0:22            0.0.0.0:*
LISTEN   0        5        127.0.0.1:631         0.0.0.0:*
LISTEN   0        128      0.0.0.0:80            0.0.0.0:*
```

Cette sortie montre trois services en écoute : le port 22 (serveur SSH), le port 631 (service CUPS, uniquement en local) et le port 80 (serveur HTTP).

Remarque. `netstat -tln` produit un résultat équivalent. On la retrouve encore dans de nombreux tutoriels et sujets d'examen.

Exercice 5 *Services en écoute* Exécutez `ss -tln` sur votre machine. Quels services sont en écoute ? Pour chaque port, recherchez le service associé (les ports standards sont documentés dans le fichier `/etc/services`).

7 Consulter la table ARP : `arp` et `ip neigh`

7.1 Principe

Le protocole **ARP** (*Address Resolution Protocol*) permet de trouver l'adresse MAC associée à une adresse IP dans un réseau local. Chaque machine maintient une **table ARP** (ou cache ARP) qui mémorise les correspondances récemment résolues.

ARP intervient au niveau de la **couche accès réseau** (couche la plus basse du modèle TCP/IP).

7.2 Exemple d'utilisation

```
$ ip neigh
192.168.1.1 dev enp3s0 lladdr e4:5d:51:a3:22:f0 REACHABLE
192.168.1.15 dev enp3s0 lladdr 3c:a6:f6:12:bb:cc STALE
```

Ou avec la commande historique :

```
$ arp -a
box (192.168.1.1) at e4:5d:51:a3:22:f0 [ether] on enp3s0
portable (192.168.1.15) at 3c:a6:f6:12:bb:cc [ether] on enp3s0
```

7.3 Lecture du résultat

Chaque ligne associe une adresse IP à une adresse MAC sur une interface donnée. L'état indique la fraîcheur de l'entrée :

— **REACHABLE** : la correspondance est récente et confirmée ;

— **STALE** : la correspondance est ancienne et pourrait ne plus être valide ;

— **INCOMPLETE** : une requête ARP a été envoyée mais aucune réponse n'a été reçue.

Exercice 6 *Table ARP*

1. Exécutez `ip neigh` (ou `arp -a`) sur votre machine. Identifiez l'adresse MAC de votre passerelle par défaut.
2. Exécutez `ping -c 1 192.168.1.X` (une adresse du réseau local), puis consultez à nouveau la table ARP. Qu'observez-vous ?

⑧ Récapitulatif des commandes

Commande	Couche TCP/IP	Rôle
`ip a` / `ifconfig`	Accès réseau	Afficher la configuration des interfaces
`ping`	Internet (ICMP)	Tester la connectivité
`traceroute` / `tracert`	Internet (ICMP/UDP)	Tracer le chemin d'un paquet
`nslookup` / `dig`	Application (DNS)	Résoudre un nom de domaine
`ss` / `netstat`	Transport (TCP/UDP)	Examiner les connexions actives
`ip neigh` / `arp`	Accès réseau (ARP)	Consulter la table ARP