

Configurations et commandes réseau

Table des matières

1	Configuration et commandes réseaux Unix	2
1.1	Les interfaces réseau	2
1.2	ifconfig : configuration manuelle d'une interface sous Unix	4
1.3	dhclient : configuration automatique par DHCP	7
1.4	ping : tester la connectivité avec un hôte	9
1.5	Configuration d'une table de routage	11
1.6	Consultation et modification du cache ARP	13
1.7	traceroute : connaître le chemin suivi par les datagrammes	14
1.8	Résolution de noms et DNS	16
1.8.1	Configuration de la résolution par /etc/nsswitch.conf	16
1.8.2	Interrogation du DNS avec host	18
1.8.3	Interrogation du DNS avec dig	20
1.8.4	Interrogation interactive du DNS avec nslookup	22
2	Commandes réseau dans l'environnement Windows	23
2.1	ipconfig	23
2.2	netsh	25
2.3	ping	27
2.4	arp	28
2.5	route	29
2.6	tracert	32

1 Configuration et commandes réseaux Unix

1.1 Les interfaces réseau

Une interface réseau identifie un périphérique permettant de se connecter à un réseau ainsi que les méthodes d'accès à ce réseau. Ce peut être un modem, une carte réseau, un port série, un port USB, ou autre. Une station ne possède et n'utilise généralement qu'une seule carte réseau. Les routeurs possèdent en revanche une interface par réseau auquel ils sont connectés.


Sur Unix (et Linux), une interface correspond à un point d'entrée dans le noyau (cœur du système). Envoyer des messages via les interfaces réseaux revient à passer des données à des procédures spéciales du noyau chargées d'effectuer les opérations d'entrées-sorties physiques. Une interface est généralement identifiée par un nom logique indiquant le type d'interface et le numéro d'ordre de la carte. Par exemple, sous Linux, une carte Ethernet classique sera identifiée par :

- **eth0** pour la première carte ;
- **eth1** pour la seconde ;
- etc.

et les cartes Wifi, plutôt par **wlan0**, **wlan1**, ...

Sur SunOS 7.0 (Unix de Sun Microsystems), une carte Ethernet sera identifiée par :


- **le0** pour la première ;
- **le1** pour la seconde ;
- etc.

 Une exception concerne l'interface **loopback** (boucle locale) identifiée par **lo** suivi éventuellement d'un numéro. C'est une **interface virtuelle** associée à l'adresse IP **127.0.0.1** (en fait à toutes les adresses IP commençant par **127**, qui sont inutilisables pour adresser un hôte sur Internet). Virtuelle veut dire que **loopback n'est rattachée à aucune carte réseau réelle**. Elle permet, avec son adresse IP, de tester uniquement en local des programmes utilisant TCP/IP, sans même disposer d'une liaison réseau. Elle permet donc de réaliser des tests sans pour autant provoquer de transmission sur le réseau, et d'utiliser localement des services réseaux. Généralement, les stations sont configurées pour que cette interface puisse être désignée par le nom **localhost** (cas des stations Unix et Windows).


Une interface possède (généralement) une adresse physique, communément appelée adresse MAC. C'est l'adresse qui l'identifie dans les communications au niveau trame (couche liaison OSI), au sein du réseau physique auquel elle est connectée. Pour que cette interface puisse être une destination dans l'Internet, il faut lui associer une adresse IP (une seule suffit). Ainsi, une station va posséder une seule adresse IP alors qu'un routeur va posséder une adresse IP par réseau auquel il est connecté, via une interface. Pour réaliser cette association, il faut configurer l'interface.

La configuration d'une interface comprend :

- son adresse IP ;
- son masque de sous-réseau ;

 le système déduit l'adresse de son réseau en appliquant le masque à l'adresse IP, et ajoute automatiquement la route directe correspondante dans la table de routage.


- selon le système et le contexte, l'adresse d'un routeur (*Gateway*) pour créer la route par défaut ;
- l'adresse IP de diffusion dirigée (si la diffusion est possible) dans le réseau de l'hôte. Un datagramme envoyé vers cette adresse est aussi destiné à cette interface ainsi qu'à toutes celles raccordées à ce réseau ;

 l'adresse de diffusion dirigée est aussi déduite de l'adresse IP et du masque, en mettant à 1 dans l'adresse IP, les bits qui sont à zéro dans le masque.

- un état actif (UP) ou inactif (DOWN) ;
- un certain nombre d'options :
 - ◇ le **MTU** (Maximum Transmission Unit) : charge utile maximale d'une trame émise via cette interface, dépendant du réseau. Vaut 1500 pour Ethernet ;
 - ◇ l'activation ou non du protocole ARP sur cette interface ;
 - ◇ la possibilité de diffuser ou non via l'interface (BROADCAST) ;
 - ◇ la possibilité de recevoir des messages émis en multi-diffusion (MULTICAST) ;
 - ◇ l'activation du mode ***promiscuous***, donnant la possibilité de garder toutes les trames reçues sur cette interface, même celles n'étant pas destinées à l'adresse physique de cette interface. Ce mode est utilisé pour réaliser des captures de trafic réseau.

1.2 ifconfig : configuration manuelle d'une interface sous Unix

La commande permettant de configurer une interface sous Unix est **ifconfig** (interface **configuration**), qui se trouve dans le répertoire `/sbin`. Elle admet d'assez nombreuses options car elle permet de configurer tout type de carte, pour différents besoins et protocoles réseau (pas seulement IPv4). Elle sert aussi à la consultation de la configuration courante des interfaces.

 Le répertoire `/sbin` tout comme `/usr/sbin` contiennent des commandes destinées normalement aux administrateurs. Ces répertoires ne sont donc pas contenus par défaut dans le **PATH** des utilisateurs. Un utilisateur normal voulant les utiliser doit soit saisir leur référence absolue soit modifier son **PATH** en tapant :

```
$ PATH="$PATH:/sbin:/usr/sbin"
```

Synopsis


```
ifconfig [-a | interface]
```

```
ifconfig interface [adresse-ip] [netmask masque] [mtu mtu] [hw ether adresse-mac]
```

La première forme sert à la consultation de la configuration courante. Si *interface* n'est pas indiquée, seule la configuration des interfaces actives (UP) est affichée. Les interfaces inactives (DOWN) sont aussi affichées si on utilise **-a**. Sinon, seule la configuration de *interface* est affichée.

La deuxième forme permet une configuration élémentaire de l'*interface* indiquée, où :

- **adresse-ip** est l'adresse IPv4 à lui attribuer ;
- **netmask masque** précise le masque de sous-réseau à utiliser ;
- **mtu mtu** précise le MTU (en octets). Il n'est généralement pas utile de le modifier ;
- **hw ether adresse-mac** permet d'utiliser une autre adresse MAC (Ethernet) que celle de la carte réseau. Sauf cas particuliers, il n'est pas nécessaire de préciser ce paramètre.

 En ligne de commandes sous Windows, la configuration d'une interface passe par la commande **netsh interface ip** (voir section 2.2 page 25), bien qu'**ipconfig** (voir section 2.1 p. 23) puisse servir, notamment pour afficher la configuration.

Exemple 1

Affichage de la configuration courante des interfaces actives :

```
# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:21:9b:dd:db:f9
          inet adr:139.124.187.55  Bcast:139.124.187.255  Masque:255.255.255.0
          adr inet6: fe80::221:9bff:fedf:dbf9/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:116984 errors:0 dropped:0 overruns:0 frame:0
          TX packets:48298 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:148500091 (141.6 MiB)  TX bytes:3921033 (3.7 MiB)
          Interruption:17
```

```
lo      Link encap:Boucle locale
        inet adr:127.0.0.1  Masque:255.0.0.0
        adr inet6: ::1/128 Scope:Hôte
        UP LOOPBACK RUNNING  MTU:16436  Metric:1
        RX packets:10964 errors:0 dropped:0 overruns:0 frame:0
        TX packets:10964 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 lg file transmission:0
        RX bytes:654158 (638.8 KiB)  TX bytes:654158 (638.8 KiB)
```

⇒ les interfaces `eth0` et `lo` (*loopback*) sont actives. Pour `eth0`, on peut noter la configuration mise en gras sur l'exemple :

- adresse MAC (*HWaddr*) : `00:21:9b:dd:db:f9`
- adresse IPv4 (*inet adr*) : `139.124.187.55`
- adresse de diffusion dirigée dans le réseau (*Bcast*) : `139.124.187.255`
- masque de sous-réseau : `255.255.255.0`
- MTU : `1500`

Affichage de la configuration courante de toutes les interfaces :

```
# ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:21:9b:dd:db:f9
        inet adr:139.124.187.55  Bcast:139.124.187.255  Masque:255.255.255.0
        ...

lo        Link encap:Boucle locale
        inet adr:127.0.0.1  Masque:255.0.0.0
        ...

tap426275 Link encap:Ethernet  HWaddr 46:86:d4:80:98:8b
        inet adr:172.23.0.254  Bcast:172.23.255.255  Masque:255.255.255.255
        BROADCAST MULTICAST  MTU:1500  Metric:1
        RX packets:1212 errors:0 dropped:0 overruns:0 frame:0
        TX packets:59 errors:0 dropped:34 overruns:0 carrier:0
        collisions:0 lg file transmission:500
        RX bytes:39060 (38.1 KiB)  TX bytes:8629 (8.4 KiB)

wlan0     Link encap:Ethernet  HWaddr 00:1f:3c:c1:05:e6
        BROADCAST MULTICAST  MTU:1500  Metric:1
        RX packets:33446 errors:0 dropped:0 overruns:0 frame:0
        TX packets:10817 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 lg file transmission:1000
        RX bytes:14998208 (14.3 MiB)  TX bytes:4210593 (4.0 MiB)
```

⇒ l'interface `wlan0` correspond à la première carte Wifi ; l'interface `tap426275` est une interface virtuelle créée par le simulateur Marionnet pour avoir un accès réseau à une machine virtuelle.

Configuration de l'interface **eth0** avec l'adresse 150.151.152.1/16 :

```
# ifconfig eth0 150.151.152.1 netmask 255.255.0.0
```

Vérification :

```
# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:21:9b:dd:db:f9
          inet adr:150.151.152.1  Bcast:150.151.255.255  Masque:255.255.0.0
          ...
```



1.3 dhclient : configuration automatique par DHCP

La commande Linux permettant d'utiliser un client DHCP (*Dynamic Host Configuration Protocol*) pour configurer automatiquement un hôte est **dhclient**. Dans notre cas, le synopsis suivant suffira.

Synopsis

```
dhclient [-r] interface
```

où *interface* est le nom de l'interface que l'on veut configurer par DHCP, ce qui nécessite la présence d'un serveur DHCP accessible par cette interface.

Sans option, la commande active un client DHCP qui cherchera un serveur DHCP depuis l'interface spécifiée, pour la configurer automatiquement. Outre l'adresse IP allouée à cette interface par le serveur et le masque du sous-réseau associé, **dhclient** configure aussi la table de routage et le solveur DNS, ainsi que d'éventuels autres paramètres fournis par le serveur DHCP.

A contrario, l'option **-r** demande de restituer l'adresse IP obtenue et annule les effets des paramètres qui avaient été fournis.

Exemple 2

Voici un exemple d'utilisation de cette commande sous Linux pour paramétrer l'interface **eth1** :

```
# dhclient eth1
Internet Systems Consortium DHCP Client V3.1.3
Copyright 2004-2009 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/eth1/00:21:9b:df:db:f9
Sending on   LPF/eth1/00:21:9b:df:db:f9
Sending on   Socket/fallback
DHCPDISCOVER on eth1 to 255.255.255.255 port 67 interval 5
DHCPOFFER from 139.124.187.10
DHCPREQUEST on eth1 to 255.255.255.255 port 67
DHCPACK from 139.124.187.10
bound to 139.124.187.34 -- renewal in 23348 seconds.
```

- ➡ le serveur 139.124.187.10 a répondu et a attribué l'adresse 139.124.187.34 à cette interface. Il faudra renouveler le bail dans 23348 secondes (près de 7 heures).

Vérification par **ifconfig** de l'adresse et du masque obtenus :

```
# ifconfig eth1
eth1      Link encap:Ethernet  HWaddr 00:21:9b:df:db:f9
          inet addr:139.124.187.34  Bcast:139.124.187.255  Mask:255.255.255.0
```

Par la commande précédente, d'autres paramètres réseaux ont aussi été obtenus auprès du serveur :

```
# route -n
```

Table de routage IP du noyau

Destination	Passerelle	Genmask	Indic	Metric	Ref	Use	Iface
0.0.0.0	139.124.187.1	0.0.0.0	UG	100	0	0	eth1
139.124.187.0	0.0.0.0	255.255.255.0	U	100	0	0	eth1

⇒ le routeur par défaut 139.124.187.1 a été communiqué par DHCP et la route associée a été ajoutée dans la table de routage

```
# cat /etc/resolv.conf
```

```
search aix.univ-amu.fr univ-amu.fr
nameserver 139.124.182.56
nameserver 139.124.1.2
```

⇒ les paramètres du solveur DNS ont été aussi communiqués et placés dans /etc/resolv.conf

Restitution de l'adresse obtenue et suppression des paramètres associés :

```
# dhclient -r eth1
```

```
Internet Systems Consortium DHCP Client V3.1.1
Copyright 2004-2008 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/
```

```
Listening on LPF/eth1/00:21:9b:df:db:f9
Sending on   LPF/eth1/00:21:9b:df:db:f9
Sending on   Socket/fallback
DHCPRELEASE on eth1 to 139.124.187.10 port 67
```

```
# route -n
```


```
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Iface
#
```

⇒ il n'y a plus de route suite à la suppression des paramètres obtenus précédemment.



1.4 ping : tester la connectivité avec un hôte

La commande **ping** est disponible sur les systèmes Unix et Windows (voir section 2.3, page 27). Elle permet de tester l'acheminement de datagrammes sur le réseau et, accessoirement, de vérifier qu'un hôte est bien opérationnel. Elle permet aussi de réaliser des statistiques sur les temps de réponse ainsi que sur le pourcentage de paquets perdus.

 Sous Linux, la commande **ping** se trouve dans le répertoire `/bin`, déjà présent dans le **PATH** d'un utilisateur normal.

Synopsis

ping [-c *nombre*] *hôte*

ping utilise par défaut le protocole ICMP en envoyant des messages (ICMP) de type "Demande d'ECHO" (ECHO REQUEST) demandant à l' *hôte* destinataire de répondre par un "Réponse d'ECHO" (ECHO REPLY). Sur certains systèmes, **ping** effectue plusieurs envois puis s'arrête en fournissant des statistiques sur le temps de propagation aller-retour (*Round Trip Time*). Sur d'autres systèmes, il faut arrêter **ping** en tapant `CTRL-C`, à moins d'avoir limité le nombre de requêtes envoyées avec l'option **-c** *nombre*.

Ainsi, lorsqu'une réponse arrive, on est assuré que l'ordinateur qu'on utilise est correctement configuré, de même que l'ordinateur interrogé, que les réseaux qui les séparent sont opérationnels et que les routeurs intermédiaires sont correctement configurés.

Exemple 3

Test d'accessibilité à l'hôte 172.16.203.250 (par un utilisateur quelconque) :

```
$ ping 172.16.203.250
PING 172.16.203.250 (172.16.203.250) 56(84) bytes of data.
64 bytes from 172.16.203.250: icmp_seq=1 ttl=255 time=1.28 ms
64 bytes from 172.16.203.250: icmp_seq=2 ttl=255 time=1.27 ms
64 bytes from 172.16.203.250: icmp_seq=3 ttl=255 time=1.29 ms
64 bytes from 172.16.203.250: icmp_seq=4 ttl=255 time=1.90 ms
CTRL-C
--- 172.16.203.250 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 1.271/1.437/1.902/0.269 ms
```

⇒ toutes les (4) requêtes ont reçu une réponse : l'hôte ciblé est parfaitement accessible depuis l'hôte courant.

Test d'accessibilité à l'hôte 192.168.1.1 limité à 5 requêtes :

```
$ ping -c 5 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.

--- 192.168.1.1 ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 4090ms
```

⇒ aucune requête n'a reçu de réponse (100% packet loss) : cet hôte n'est absolument pas accessible

Autre cas d'erreur (parmi d'autres) :

```
$ ping -c 5 8.8.8.8
```

```
connect: Le réseau n'est pas accessible
```

⇒ aucune interface réseau n'est configurée sur l'hôte courant ou il n'y a pas de route pour cette destination



1.5 Configuration d'une table de routage

Les informations que l'on peut (ou doit) spécifier lorsqu'on ajoute une route dans une table de routage sont :


- le type de destination (réseau ou hôte);
- son adresse IP;
- le masque de cette adresse (afin de prendre en compte des regroupements de réseaux) s'il est différent de celui de la classe du réseau;
- le routeur associé (0.0.0.0 si la route est directe);
- l'interface à utiliser pour cette route (lo, eth0,...), et permettant de communiquer avec la destination (route directe) ou le routeur.


La configuration de la table de routage se fait au moyen de la commande **route** sur Unix et Windows (voir section 2.5, page 29). Cette commande permet d'ajouter ou de supprimer des routeurs vers des réseaux ou des stations. Dans notre cas, le synopsis suivant devrait suffire.

Synopsis

```
route [-n]
route add -net adresse-destination netmask masque gw routeur [dev interface]
route del -net adresse-destination netmask masque gw routeur [dev interface]
```

La première forme permet d'afficher la table de routage. L'option **-n** désactive la résolution inverse DNS, ce qui est parfois bien pratique. La seconde permet d'ajouter une route passant par le routeur *routeur* vers la destination *adresse-destination* de masque *masque*. Si besoin, l'interface à utiliser pour cette route peut être indiquée avec **dev interface**. La dernière forme permet de supprimer une route.

 Il n'est pas nécessaire d'ajouter dans la table le réseau auquel est connectée l'interface de la station. En effet, sa configuration avec **ifconfig** a automatiquement provoqué son ajout dans la table.

 Il est possible avec **route** d'interdire une destination en utilisant l'option **reject** mais cela ne remplace pas l'utilisation d'un bon *firewall*...

Exemple 4

Supposons que nous voulons ajouter une route vers la destination 139.124.110.0/24 qui passe par le routeur 150.151.152.250 et **une route par défaut** qui passe par le routeur 150.151.152.153.

Dans un premier temps, supposons que l'hôte est déjà configuré au niveau IP, et consultons sa table de routage :

```
# route -n
Table de routage IP du noyau
Destination      Passerelle      Genmask          Indic Metric Ref       Use Iface
150.151.0.0      0.0.0.0         255.255.0.0      U         0      0         0 eth0
```

On ajoute ensuite la route vers 139.124.187.0/24 :

```
# route add -net 139.124.110.0 netmask 255.255.255.0 gw 150.151.152.250
```

puis la route par défaut :

```
# route add -net 0.0.0.0 netmask 0.0.0.0 gw 150.151.152.153
```

Puis on peut vérifier en affichant la table :

```
# route -n
Table de routage IP du noyau
Destination      Passerelle      Genmask          Indic Metric Ref       Use Iface
150.151.0.0      0.0.0.0         255.255.0.0      U        0        0        0 eth0
139.124.110.0    150.151.152.250 255.255.255.0    UG       0        0        0 eth0
0.0.0.0          150.151.152.153 0.0.0.0          UG       0        0        0 eth0
```

Puis en testant la connectivité d'un hôte (dont on sait qu'il répond en temps normal) :

```
# ping 139.124.110.8
PING 139.124.110.8 (139.124.110.8) 56(84) bytes of data.
64 bytes from 139.124.110.8: icmp_seq=1 ttl=64 time=0.276 ms
64 bytes from 139.124.110.8: icmp_seq=2 ttl=64 time=0.485 ms
64 bytes from 139.124.110.8: icmp_seq=3 ttl=64 time=0.408 ms
64 bytes from 139.124.110.8: icmp_seq=4 ttl=64 time=0.363 ms
CTRL+C
--- 139.124.110.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2998ms
rtt min/avg/max/mdev = 0.276/0.383/0.485/0.075 ms
```



❗ La colonne **Indic** (ou **flags**, selon l'installation) contient une combinaison d'indicateurs donnant quelques renseignements sur la route. Parmi les indicateurs possibles, il y a **U**, **H**, **G**, **D**, **M** et **!** :

- **U** : la route est en service (activée).
- **H** : la destination est un ordinateur (*host*). Sans cet indicateur, la destination est un réseau.
- **G** : la route n'est pas directe et la passerelle est un routeur (*gateway*). Sans cet indicateur, la destination est directement accessible.
- **D** : la route a été créée par une redirection (message ICMP).
- **M** : la route a été modifiée par une redirection (message ICMP).
- **!** : la route est rejetée (option **reject**).

Dans certaines implémentations, la table de routage peut contenir les adresses de la station elle-même. Par exemple, pour la station d'adresse 139.124.187.4, on pourrait avoir comme table :

Destination	Passerelle	Genmask	Indic	Metric	Ref	Use	Iface
139.124.187.4	0.0.0.0	255.255.255.255	UH	0	0	0	eth0
139.124.187.0	0.0.0.0	255.255.255.0	U	10	0	0	eth0
127.0.0.0	0.0.0.0	255.0.0.0	U	0	0	0	lo
0.0.0.0	139.124.187.1	0.0.0.0	UG	10	0	0	eth0




Un raccourci pour créer une route par défaut est d'utiliser la forme :

```
route add default gw routeur [dev interface]
```

1.6 Consultation et modification du cache ARP

La commande **arp** permet de consulter et de modifier le cache ARP, sous Unix (Linux) et Windows (voir section 2.4, page 28). L'option **-h** vous permettra d'obtenir une aide succincte. Comme d'habitude, plus d'informations se trouvent dans les pages de **man** de la commande.

 Sous Linux, la commande **arp** est située dans le répertoire `/usr/sbin` qui ne figure pas par défaut dans le **PATH** d'un utilisateur normal.

Exemple 5

Voici un petit exemple d'informations affichées par **arp** :

```
$ /usr/sbin/arp
Address                HWtype  HWaddress          Flags Mask          Iface
b122.iut.univ-aix.fr   ether    f0:4d:a2:24:24:5a  C                   eth0
imp-a.iut.univ-aix.fr   (incomplete)
iutextreme.iut.univ-aix ether    00:23:89:57:5e:a3  C                   eth0
...
$ /usr/sbin/arp -n
Address                HWtype  HWaddress          Flags Mask          Iface
139.124.187.122        ether    f0:4d:a2:24:24:5a  C                   eth0
139.124.187.94         (incomplete)
139.124.187.1          ether    00:23:89:57:5e:a3  C                   eth0
...
```

⇒ on voit que l'entrée 139.124.187.94 (imp-a.iut.univ-aix.fr) est incomplète car ce PC cherche à la résoudre (attend la réponse ARP de sa requête). Les entrées marquées **C** dans la colonne *Flags* sont des entrées temporaires normales.



Exemple 6

En poursuivant l'exemple précédent, voici un exemple de suppression manuelle d'une entrée du cache (en supposant que **sudo** n'a ici pas besoin de mot de passe) :


```
$ sudo arp -d 139.124.187.122
$ sudo arp
Address                HWtype  HWaddress          Flags Mask          Iface
139.124.187.94         (incomplete)
139.124.187.1          ether    00:23:89:57:5e:a3  C                   eth0
...
```

⇒ on voit que l'entrée 139.124.187.122 a bien été retirée du cache.



1.7 traceroute : connaître le chemin suivi par les datagrammes

La commande **traceroute** permet de connaître le chemin (route) que suivent les datagrammes envoyés vers un hôte donné. Son homologue sous Windows est **tracert** (voir section 2.6, page 32).

 **traceroute** affiche les adresses IP des routeurs traversés pour atteindre l'hôte. Cependant, en cas de routage dynamique, ce ne sera peut être plus le chemin utilisé pour des envois ultérieurs vers ce même hôte.

Elle permet ainsi de savoir à quel endroit bloque la transmission d'un paquet que l'on tente d'envoyer sans succès (malheureusement, ça arrive).

Pour afficher ces routeurs, elle provoque une erreur d'acheminement sur chaque routeur par lequel passe le datagramme IP en agissant sur le champ TTL de ce dernier. En effet, **traceroute** commence par envoyer un datagramme UDP véhiculé par un datagramme IP avec un TTL positionné à 1. Le premier routeur rencontré détruit le datagramme et renvoie une erreur ICMP de TTL expiré. On obtient ainsi l'adresse du premier routeur de la route. **traceroute** envoie ensuite un datagramme UDP dans un datagramme IP avec un TTL à 2 pour connaître le second routeur, et ainsi de suite, jusqu'à atteindre la destination spécifiée (mais sur un port non attribué pour recevoir un message ICMP de port inaccessible).

Notons qu'à des fins de statistiques sur les temps de réponse des routeurs, par défaut **traceroute** envoie en réalité 3 messages UDP consécutifs avec le même TTL.

Exemple 7

Liste des routeurs traversés pour atteindre l'hôte `www.kernel.org` (d'adresse `149.20.4.69`) :

```
$ traceroute www.kernel.org
traceroute to www.kernel.org (149.20.4.69), 30 hops max, 60 byte packets
 1 gateway (172.16.203.250)  2.553 ms  2.930 ms  3.351 ms
 2 192.168.110.250 (192.168.110.250)  1.645 ms  1.888 ms  2.093 ms
 3 193.51.105.85 (193.51.105.85)  1.707 ms  1.957 ms  2.156 ms
 4 192.168.151.1 (192.168.151.1)  0.546 ms  0.618 ms  0.767 ms
 5 10.251.50.2 (10.251.50.2)  1.022 ms  1.008 ms  1.002 ms
 6 10.251.60.1 (10.251.60.1)  1.291 ms  1.174 ms  1.291 ms
 7 10.100.200.9 (10.100.200.9)  1.443 ms  1.571 ms  1.688 ms
 8 * * *
 9 te1-1-marseille2[etc] (193.51.177.185)  12.268 ms te0-6-0-2-lyon1[etc] (193.51.177.223)  14.654 ms te0-0-0-0-lyon1[etc] (193.51.177.16)  16.959 ms
10 te2-5-lyon2-rtr-021.noc.renater.fr (193.51.177.217)  6.628 ms  6.591 ms  6.640 ms
11 te0-6-0-0-paris2-rtr-001.noc.renater.fr (193.51.177.144)  14.589 ms  14.453 ms  14.043 ms
12 10gigabitethernet-2-2.par2.he.net (195.42.144.104)  22.054 ms  22.088 ms  21.678 ms
13 100ge10-1.core1.nyc4.he.net (184.105.81.77)  85.619 ms  84.605 ms  84.546 ms
14 100ge14-2.core1.sjc2.he.net (184.105.81.213)  153.018 ms  148.242 ms  145.994 ms
15 10ge3-3.core1.pao1.he.net (72.52.92.69)  157.847 ms  157.801 ms  152.401 ms
16 isc.gige-g4-17.core1.pao1.he.net (72.52.94.70)  151.070 ms  151.115 ms  150.925 ms
17 pub2.kernel.org (149.20.4.69)  149.895 ms  149.725 ms  150.352 ms
```

⇒ on en déduit qu'il faut traverser 16 routeurs pour atteindre la destination. On remarque aussi que le 8^e routeur n'a pas renvoyé le message d'erreur ICMP et reste inconnu. Enfin, on observe que sur les 3 messages envoyés au 9^e routeur, on n'obtient jamais le même routeur. Cela est dû au routage dynamique.

La même commande tapée quelques instants après, en désactivant la résolution de noms avec **-n** :

```
$ traceroute -n www.kernel.org
traceroute to www.kernel.org (199.204.44.194), 30 hops max, 60 byte packets
 1 172.16.203.250  2.575 ms  3.012 ms  3.357 ms
 2 192.168.110.250  1.384 ms  1.606 ms  1.833 ms
 3 193.51.105.85   1.868 ms  2.131 ms  2.354 ms
 4 192.168.151.1   0.597 ms  0.741 ms  0.886 ms
 5 10.251.50.2     1.052 ms  1.044 ms  1.037 ms
 6 10.251.60.1     1.283 ms  1.436 ms  1.552 ms
 7 10.100.200.9    1.866 ms  1.769 ms  1.845 ms
 8 * * *
 9 149.6.155.233   1.495 ms  1.617 ms  1.654 ms
10 130.117.49.153  12.249 ms 12.566 ms 12.417 ms
11 154.54.56.129   19.914 ms 19.976 ms 19.853 ms
12 154.54.39.150   26.664 ms 26.618 ms 26.575 ms
13 154.54.44.162   95.869 ms 95.804 ms 95.773 ms
14 154.54.25.125   95.965 ms 96.084 ms 95.847 ms
15 154.54.27.142   96.136 ms 154.54.40.162 96.221 ms 96.210 ms
16 154.24.5.190    96.202 ms 96.197 ms 96.201 ms
17 38.140.46.58    95.781 ms 95.694 ms 95.676 ms
18 199.204.44.194  95.606 ms 96.021 ms 95.865 ms
```

⇒ on observe à nouveau un changement majeur de chemin au plus tard au 9^e routeur, et un "allongement" de la route mais avec de bien meilleurs temps de réponse.




❗ Il existe des versions graphiques de **traceroute**, notamment **xtraceroute** (qui n'est pas installé sur nos stations) mais la localisation géographique des routeurs est loin d'être vraiment précise...

1.8 Résolution de noms et DNS

1.8.1 Configuration de la résolution par `/etc/nsswitch.conf`

Le fichier `/etc/nsswitch.conf` liste les méthodes de résolution à utiliser et dans quel ordre.

 `/etc/nsswitch.conf` provient de Sun Microsystems et de son système Solaris 2 et veut dire "aiguilleur de service de nom" (*name service switch*).


Outre la résolution de noms, ce fichier indique comment obtenir d'autres ressources du système éventuellement décentralisées (utilisateurs et mots de passe, groupes d'utilisateurs, alias de messagerie, etc.). Pour chaque ressource, il contient une ligne qui indique comment la rechercher, par ordre de méthode de recherche.

Pour une *ressource* donnée, la ligne a la simple syntaxe suivante :

`ressource : { service }`

où *service* est le service à utiliser. Lorsqu'il y a plusieurs services, ils sont consultés de gauche à droite jusqu'à trouver une correspondance. Parmi les services possibles actuellement on peut citer :

- **files** : rechercher dans les fichiers locaux. Le(s) fichier(s) à consulter dépend(ent) de la ressource (`/etc/hosts` pour un nom d'hôte, `/etc/passwd` pour un utilisateur, etc.)
- **dns** : rechercher par le service DNS
- **nis** : rechercher par le service NIS (de Sun Microsystems), appelé aussi pages-jaunes (*yellow pages*), maintenant obsolète
- **ldap** : rechercher dans un annuaire par le protocole LDAP
- **winbind** : rechercher dans un domaine Windows (informations utilisateur/groupe, authentification) via le logiciel SAMBA
- **wins** : rechercher via le système de nommage Windows
- **mdns4_minimal** : méthode récente de recherche des noms se terminant par **.local** via une interrogation sur le réseau local appelée multicast DNS, à la manière du protocole *Apple Bonjour*. Cette méthode a été introduite pour permettre à des hôtes (sans configuration ni serveur DNS) d'un réseau local de s'auto-configurer et de collaborer pour se découvrir et se nommer
- **mdns4** : comme `mdns4_minimal` mais étendu aux noms ne se terminant pas par **.local**
- **myhostname** : méthode récente pour des noms locaux de l'hôte lui-même, tels que **localhost** ou se terminant par **.localhost** ou **.localhost.localdomain**, ou comme **gateway** pour le routeur par défaut

 La *ressource* qui concerne les noms d'hôtes est **hosts**. En principe, pour la résolution de noms d'hôtes, c'est le fichier local `/etc/hosts` qui est préféré avant tout autre service (notamment le DNS). Autrement dit, le DNS n'est utilisé que si le nom recherché ne figure pas dans `/etc/hosts`.

Exemple 8

Sur une Debian récente avec une configuration standard, le fichier `/etc/nsswitch.conf` devrait contenir :

```
passwd:      compat
group:       compat
shadow:      compat
gshadow:     files

hosts:       files myhostname mdns4_minimal [NOTFOUND=return] dns
networks:    files

protocols:   db files
services:    db files
ethers:       db files
rpc:          db files

netgroup:    nis
```

- ⇒ la ligne mise en gras indique que pour la résolution de noms d'hôtes (`hosts`), il faut d'abord utiliser, dans l'ordre : le fichier `/etc/hosts` (`files`), les noms locaux (`myhostname`), le multicast DNS pour les noms en `.local` (`mdns4_minimal`), et enfin le DNS (`dns`).
Le `[NOTFOUND=return]` empêche d'utiliser le DNS si le nom en `.local` n'a pas été trouvé avec le multicast DNS (dont c'est le rôle).



❗ Pour en savoir plus sur le fichier `nsswitch.conf`, consulter le manuel de ce fichier en tapant :

```
$ man nsswitch.conf
```

1.8.2 Interrogation du DNS avec host

host est une commande non interactive qui s'utilise en ligne de commandes.

 Sous Linux **host** se trouve dans le répertoire `/usr/bin`.


Son synopsis le plus basique est le suivant.

Synopsis

```
host [-r] [-t type] nom [serveur]
```

où :

- *nom* est le nom que l'on veut résoudre. Ce peut être une adresse IP, dans ce cas, **host** effectue une **résolution inverse**
- *serveur* est le serveur de noms qui sera utilisé pour la résolution. S'il n'est pas spécifié, le serveur utilisé est celui par défaut (le premier **nameserver** de `/etc/resolv.conf`)
- **-r** désactive la recherche récursive. La recherche récursive est activée par défaut et demande au serveur de noms, si celui-ci ne sait pas résoudre le *nom* (pour le *type* de question posée), de contacter un serveur adéquat qui saura le faire (ou qui contactera lui-même un autre serveur, etc.), et d'afficher la réponse
- **-t type** précise le type d'information (voir encadré ci-dessous) que l'on souhaite obtenir, correspondant à un ou plusieurs **enregistrements DNS**.

 Pour ces enseignements, les informations qui nous intéressent sont demandées par les *types* suivants (la casse importe peu) :

- **A** : adresse IPv4 correspondant à *nom* (il existe aussi le type **AAAA** pour les adresses IPv6)
- **NS** : (Name Server) serveur de domaines ayant autorité sur le domaine *nom*. Il faut dans ce cas que *nom* soit un domaine (pas un hôte)
- **MX** : (Mail eXchanger) serveur SMTP du domaine *nom*. Il faut dans ce cas que *nom* soit un domaine (pas un hôte)
- **PTR** : enregistrement servant à la **résolution inverse** (d'adresse IPv4 en nom d'hôte)
- **ANY** : tout type d'information disponible pour ce *nom*

Exemple 9

(a) Obtenir l'adresse IPv4 de `www.service-public.fr` :

```
$ host -t a www.service-public.fr
www.service-public.fr has address 160.92.50.135
```

 on apprend que son adresse IPv4 est `160.92.50.135`

(b) Obtenir l'adresse IPv4 de `www.debian.org` :

```
$ host -t a www.debian.org
www.debian.org has address 5.153.231.4
www.debian.org has address 130.89.148.14
```

⇒ Il y a deux ordinateurs répondant à ce nom (pour répartir la charge du serveur)

(c) Obtenir l'adresse IPv4 de `www.rfc-editor.org` :

```
$ host -t a www.rfc-editor.org
www.rfc-editor.org is an alias for rfc-editor.org.
rfc-editor.org has address 4.31.198.49
```

⇒ on apprend que ce hôte s'appelle aussi simplement `rfc-editor.org` et que son adresse IPv4 est `4.31.198.49`

(d) Savoir si le `www.rfc-editor.org` est un alias et de quel nom (la réciproque n'étant pas vraie) :

```
$ host -t cname www.rfc-editor.org
www.rfc-editor.org is an alias for rfc-editor.org.
$ host -t cname rfc-editor.org
rfc-editor.org has no CNAME record
```

(e) Demander la résolution inverse requiert de poser la question en présentant à l'envers l'adresse `4.31.198.49` dans le domaine **`in-addr.arpa`** :

```
$ host -t ptr 49.198.31.4.in-addr.arpa
49.198.31.4.in-addr.arpa domain name pointer rfc-editor.org.
```

(f) Obtenir le(s) serveur(s) de noms du domaine `service-public.fr` :

```
$ host -t ns service-public.fr
service-public.fr name server ns1.dila.fr.
service-public.fr name server ns2.dila.fr.
service-public.fr name server ns3.dila.fr.
```

(g) Obtenir le(s) serveur(s) SMTP en charge de la réception des courriels envoyés aux utilisateurs du domaine `hotmail.fr` (adresses électroniques de type `toto@hotmail.fr`) :

```
$ host -t mx hotmail.fr
hotmail.fr mail is handled by 5 mx1.hotmail.com.
hotmail.fr mail is handled by 5 mx2.hotmail.com.
hotmail.fr mail is handled by 5 mx3.hotmail.com.
hotmail.fr mail is handled by 5 mx4.hotmail.com.
```


(h) Obtenir tout type d'information disponible sur le domaine `univ-amu.fr` :

```
$ host -t any univ-amu.fr
univ-amu.fr name server ns1.univmed.fr.
univ-amu.fr name server cnudns.cines.fr.
univ-amu.fr mail is handled by 10 mx1-phr.univ-amu.fr.
univ-amu.fr mail is handled by 0 mx1-stj.univ-amu.fr.
univ-amu.fr has address 139.124.244.77
```



1.8.3 Interrogation du DNS avec dig

dig est une commande non interactive qui s'utilise en ligne de commandes et qui présente la totalité des informations obtenues.

 Sous Linux **dig** se trouve dans le répertoire `/usr/bin`.

Son synopsis le plus basique est le suivant.

Synopsis

dig [*@serveur*] *nom type*

où :

- *serveur* est le serveur de noms à contacter
- *nom* est le nom que l'on veut résoudre
- *type* est le type de question posée (comme pour **host**)

Exemple 10

Quelques interrogations de l'exemple précédent réalisées avec **dig** :

(a) IPv4 d'`www.rfc-editor.org` (CNAME et IPv4 mis en évidence en gras) :

```
$ dig www.rfc-editor.org a

; <<>> DiG 9.10.3-P4-Debian <<>> www.rfc-editor.org a
;; global options: +cmd
;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 33688
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.rfc-editor.org. IN A

;; ANSWER SECTION:
www.rfc-editor.org. 1800 IN CNAME rfc-editor.org.
rfc-editor.org. 1800 IN A 4.31.198.49

;; Query time: 276 msec
;; SERVER: 193.51.217.162#53(193.51.217.162)
;; WHEN: Thu Mar 02 13:59:24 CET 2017
;; MSG SIZE rcvd: 77
```

(b) Résolution inverse de 4.31.198.49 :

```
$ dig 49.198.31.4.in-addr.arpa ptr

; <<>> DiG 9.10.3-P4-Debian <<>> 49.198.31.4.in-addr.arpa ptr
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22932
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;49.198.31.4.in-addr.arpa. IN PTR

;; ANSWER SECTION:
49.198.31.4.in-addr.arpa. 85690 IN PTR rfc-editor.org.

;; AUTHORITY SECTION:
31.4.in-addr.arpa. 85690 IN NS dnsauth2.sys.gtei.net.
31.4.in-addr.arpa. 85690 IN NS dnsauth1.sys.gtei.net.
31.4.in-addr.arpa. 85690 IN NS dnsauth3.sys.gtei.net.

;; Query time: 0 msec
;; SERVER: 193.51.217.162#53(193.51.217.162)
;; WHEN: Thu Mar 02 13:57:43 CET 2017
;; MSG SIZE rcvd: 162
```

(c) Serveur(s) de noms du domaine service-public.fr (réponse tronquée) :

```
$ dig service-public.fr NS
[...]
;; ANSWER SECTION:
service-public.fr. 2988 IN NS ns1.dila.fr.
service-public.fr. 2988 IN NS ns2.dila.fr.
service-public.fr. 2988 IN NS ns3.dila.fr.
[...]
```

(d) Serveur(s) SMTP du domaine hotmail.fr

```
$ dig hotmail.fr mx
[...]
;; ANSWER SECTION:
hotmail.fr. 2739 IN MX 5 mx2.hotmail.com.
hotmail.fr. 2739 IN MX 5 mx3.hotmail.com.
hotmail.fr. 2739 IN MX 5 mx4.hotmail.com.
hotmail.fr. 2739 IN MX 5 mx1.hotmail.com.
[...]
```


```
;; ADDITIONAL SECTION:
mx1.hotmail.com. 2739 IN A 65.55.37.88
mx1.hotmail.com. 2739 IN A 65.55.37.104
[...]
```

```
mx4.hotmail.com. 2739 IN A 65.55.37.88
mx4.hotmail.com. 2739 IN A 65.55.37.120
mx4.hotmail.com. 2739 IN A 65.55.92.152
[...]
```



1.8.4 Interrogation interactive du DNS avec nslookup

À la différence des deux commandes précédentes, **nslookup** est une commande interactive qui offre un invite de commande (*prompt*).

 Sous Linux **nslookup** se trouve dans le répertoire `/usr/bin`.

Son synopsis le plus basique est le suivant.

Synopsis


nslookup

En l'exécutant, un prompt s'affiche et l'on peut paramétrer le solveur de nom avec les commandes suivantes :

- **set [no] rec** pour désactiver (**norec**) ou activer (**rec**) le mode récursif. Il est activé par défaut
- **server serveur** pour indiquer que l'on veut interroger le serveur de noms *serveur*. Sans cette commande, le serveur utilisé est celui par défaut de l'ordinateur (voir `/etc/resolv.conf`)
- **set q=type** pour préciser le *type* de question posée. Le *type* par défaut est **A**
- **nom** provoque l'émission d'une requête DNS au serveur choisi précédemment. Elle demande les informations correspondant au *type* choisi et concernant *nom*. Si *nom* ne se termine pas par un point, il n'est pas complètement qualifié (FQDN) et si la recherche échoue pour ce *nom*, des requêtes supplémentaires seront émises en ajoutant à *nom* les domaines de recherche de la machine (lignes **domain** et **search** dans `/etc/resolv.conf`) jusqu'à obtenir une réponse
- **exit** pour quitter

2 Commandes réseau dans l'environnement Windows

Cette section regroupe les équivalents Windows des commandes étudiées. Elle est constituée principalement d'exemples d'utilisation.

 Les exemples d'exécution des commandes sont tapés sur une fenêtre terminal de Windows XP (émulation MSDOS), lancée via le menu *Démarrer* → *Exécuter* → **cmd**.

2.1 ipconfig

ipconfig permet de consulter la configuration IP, et d'obtenir des informations plus détaillées sur les interfaces et autres éléments de configuration. Elle permet de réaliser des opérations élémentaires DHCP comme le renouvellement d'une adresse ou sa résiliation.

Exemple 11

- Obtenir de l'aide sur **ipconfig** :

```
C:>ipconfig/?
```

UTILISATION :

```
ipconfig [/? | /all | /renew [carte] | /release [carte] |
        /flushdns | /displaydns | /registerdns |
        /showclassid carte |
        /setclassid carte [ID de classe] ]
```

...

- Afficher les informations principales sur la configuration IP :

```
C:\>ipconfig
```

Configuration IP de Windows

Carte Ethernet Connexion au réseau local:

```
Suffixe DNS propre à la connexion :
Adresse IP. . . . . : 10.0.2.15
Masque de sous-réseau . . . . . : 255.255.255.0
Passerelle par défaut . . . . . : 10.0.2.2
```

- Afficher une synthèse des configurations réseau de l'hôte :

```
C:\>ipconfig/all
```


Configuration IP de Windows

```
Nom de l'hôte . . . . . : vb-xp-iut
Suffixe DNS principal . . . . . :
Type de noud . . . . . : Inconnu
```

```
Routage IP activé . . . . . : Non
Proxy WINS activé . . . . . : Non
```

Carte Ethernet Connexion au réseau local:

```
Suffixe DNS propre à la connexion :
Description . . . . . : Carte AMD PCNET Family Ethernet PCI
Adresse physique . . . . . : 08-00-27-71-76-54
DHCP activé. . . . . : Oui
Configuration automatique activée . . . . : Oui
Adresse IP. . . . . : 10.0.2.15
Masque de sous-réseau . . . . . : 255.255.255.0
Passerelle par défaut . . . . . : 10.0.2.2
Serveur DHCP. . . . . : 10.0.2.2
Serveurs DNS . . . . . : 212.27.40.240
                        212.27.40.241
Bail obtenu . . . . . : jeudi 12 avril 2012 14:33:28
Bail expirant . . . . . : vendredi 13 avril 2012 14:33:28
```

 Sous Windows, les interfaces sont nommées comme ici "**Connexion au réseau local**". On voit aussi que la représentation d'une adresse Ethernet change sur Windows (08-00-27-71-76-54).

- Utiliser **ipconfig** pour résilier une allocation DHCP (pour une interface configurée pour DHCP, voir **netsh**) :
C:>**ipconfig/release**

Configuration IP de Windows

Carte Ethernet Connexion au réseau local:

```
Suffixe DNS propre à la connexion :
Adresse IP. . . . . : 0.0.0.0
Masque de sous-réseau . . . . . : 0.0.0.0
Passerelle par défaut . . . . . :
```

- et pour renouveler sa configuration par DHCP :
C:>**ipconfig/renew**

Configuration IP de Windows

Carte Ethernet Connexion au réseau local:

```
Suffixe DNS propre à la connexion : iut.univ-aix.fr
Adresse IP. . . . . : 10.0.2.15
Masque de sous-réseau . . . . . : 255.255.255.0
Passerelle par défaut . . . . . : 10.0.2.2
```



2.2 netsh

La commande **netsh** regroupe un ensemble de modules permettant d'agir sur la configuration réseau d'un poste Windows. Pour configurer les paramètres IPv4 d'une interface réseau, il faut utiliser le module **netsh interface ip**. Il permet notamment d'affecter manuellement (statiquement) les paramètres IPv4 ou de configurer l'interface pour les obtenir par DHCP (dynamiquement).

Exemple 12

- Obtenir de l'aide sur **netsh interface ip** :

```
C:>netsh interface ip
```

Les commandes suivantes sont disponibles :

Commandes dans ce contexte :

?	- Affiche une liste de commandes.
add	- Ajoute une entrée de configuration à une table.
delete	- Supprime une entrée de configuration d'une table.
dump	- Affiche un script de configuration.
help	- Affiche une liste de commandes.
reset	- Réinitialisez TCP/IP et les composants associés.
set	- Définit l'information de configuration.
show	- Affiche les informations.

Pour consulter l'aide d'une commande, entrez la commande, suivie par un espace, et ensuite entrez ?.

- Fixer statiquement l'adresse IPv4 10.0.2.100/24 pour l'interface Connexion au réseau local. Fixer aussi les paramètres (optionnels) *Gateway* et *métrique* de la route par défaut à 10.0.2.2 et 20 (le tout sur une seule ligne) :

```
C:>netsh int ip set address "Connexion au réseau local"  
static 10.0.2.100 255.255.255.0 10.0.2.2 20
```

Ok.

- Vérifications de la configuration :

```
C:>ipconfig
```

Configuration IP de Windows

Carte Ethernet Connexion au réseau local:

```
Suffixe DNS propre à la connexion :  
Adresse IP. . . . . : 10.0.2.100  
Masque de sous-réseau . . . . . : 255.255.255.0  
Passerelle par défaut . . . . . : 10.0.2.2
```

```
C:>route print
```

```
=====
```

```
Liste d'Interfaces
```

```
0x1 ..... MS TCP Loopback interface
0x2 ...08 00 27 71 76 54 ..... Carte AMD PCNET Family Ethernet PCI - Mini-
port d'ordonnement de paquets
=====
```

```
=====
```

```
Itinéraires actifs :
```

Destination réseau	Masque réseau	Adr. passerelle	Adr. interface	Métrique
0.0.0.0	0.0.0.0	10.0.2.2	10.0.2.100	20
10.0.2.0	255.255.255.0	10.0.2.100	10.0.2.100	20
10.0.2.100	255.255.255.255	127.0.0.1	127.0.0.1	20
10.255.255.255	255.255.255.255	10.0.2.100	10.0.2.100	20
127.0.0.0	255.0.0.0	127.0.0.1	127.0.0.1	1
224.0.0.0	240.0.0.0	10.0.2.100	10.0.2.100	20
255.255.255.255	255.255.255.255	10.0.2.100	10.0.2.100	1

```
Passerelle par défaut : 10.0.2.2
=====
```

```
Itinéraires persistants :
```

```
Aucun
```

- Configurer la même interface pour obtenir les informations par DHCP (et abandonner la configuration statique) puis vérification :

```
C:>netsh int ip set address "Connexion au réseau local" dhcp
```

```
Ok.
```

```
C:>ipconfig
```

```
Configuration IP de Windows
```

```
Carte Ethernet Connexion au réseau local:
```

```
Suffixe DNS propre à la connexion : iut.univ-aix.fr
Adresse IP. . . . . : 10.0.2.15
Masque de sous-réseau . . . . . : 255.255.255.0
Passerelle par défaut . . . . . : 10.0.2.2
```



2.3 ping

La commande **ping** sous Windows a les mêmes fonctionnalités que sous Unix, bien qu'elle n'admette pas tout à fait les mêmes options. Notamment, elle s'arrête par défaut après l'émission de 4 requêtes.

Exemple 13

- Obtenir de l'aide sur **ping** :

C:>**ping**/?

Utilisation : ping [-t] [-a] [-n échos] [-l taille] [-f] [-i vie] [-v TypServ]
 [-r NbSauts] [-s NbSauts] [[-j ListeHôtes] | [-k ListeHôtes]]
 [-w Délai] NomCible

Options :

- t Envoie la requête ping sur l'hôte spécifié jusqu'à interruption.
 Entrez Ctrl-Attn pour afficher les statistiques et continuer, Ctrl-C pour arrêter.
- a Recherche les noms d'hôte à partir des adresses.
- n échos Nombre de requêtes d'écho à envoyer.

- Tester l'accissibilité d'un hôte :

C:>**ping 139.124.187.4**

Envoi d'une requête 'ping' sur 139.124.187.4 avec 32 octets de données :

Réponse de 139.124.187.4 : octets=32 temps=58 ms TTL=127
Réponse de 139.124.187.4 : octets=32 temps=48 ms TTL=127
Réponse de 139.124.187.4 : octets=32 temps=57 ms TTL=127
Réponse de 139.124.187.4 : octets=32 temps=47 ms TTL=127

Statistiques Ping pour 139.124.187.4:

Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
Minimum = 47ms, Maximum = 58ms, Moyenne = 52ms



2.4 arp

La commande **arp** sous Windows a les mêmes fonctionnalités que sous Unix.

Exemple 14

- Obtenir de l'aide sur **arp** :

```
C:\>arp/?
```

Affiche et modifie les tables de traduction d'adresses IP en adresses physiques utilisées par le protocole de résolution d'adresses ARP.

```
ARP -s inet_addr eth_addr [if_addr]
```

```
ARP -d inet_addr [if_addr]
```

```
ARP -a [inet_addr] [-N if_addr]
```

-a Affiche les entrées ARP en cours en interrogeant les données en cours du protocole. Si inet_addr est spécifié, seules les adresses IP et physiques de l'ordinateur spécifié sont affichées. Si plus d'une interface réseau utilise ARP, les entrées de chaque table ARP sont affichées.

...

Exemples :

```
> arp -s 157.55.85.212 00-aa-00-62-c6-09 .... Ajoute une entrée statique.
> arp -a .... Affiche la table ARP.
```

- Afficher le cache ARP :

```
C:\>arp -a
```

```
Interface : 10.0.2.15 --- 0x2
```

Adresse Internet	Adresse physique	Type
10.0.2.2	52-54-00-12-35-02	dynamique



2.5 route

Comme sous Unix, la commande **route** sous Windows permet d'afficher la table de routage et de la modifier. Cependant, on notera que les informations figurant dans les tables Windows sont sensiblement différentes que celles affichées sous Linux.

Exemple 15

- Obtenir de l'aide sur **route** :

```
C:>route
```

Manipule les tables de routage du réseau.

```
ROUTE [-f] [-p] [cmde [destin]
                    [MASK MasqueRés] [passerelle] [METRIC coût] [IF interface]
```

...

cmde	Spécifie une des quatre commandes suivantes :
PRINT	Affiche un itinéraire
ADD	Ajoute un itinéraire
DELETE	Supprime un itinéraire
CHANGE	Modifie un itinéraire existant
destin	Spécifie l'hôte destination.
MASK	Si le mot clé MASK est présent, le paramètre qui le suit est interprété en tant que paramètre de masque réseau.
MasqueRés	Spécifie la valeur éventuelle du sous-masque réseau à associer avec cette entrée d'itinéraire. La valeur par défaut est : 255.255.255.255.
passerelle	Spécifie la passerelle.
interface	numéro d'interface pour l'itinéraire spécifié.
METRIC	Spécifie le coût métrique pour la destination

...

Exemples :

```
> route PRINT
> route ADD 157.0.0.0 MASK 255.0.0.0 157.55.80.1 METRIC 3 IF 2

destination^          ^masque passerelle^  métrique^  ^
                                   interface^
```

Si IF n'est pas fourni, la meilleure interface pour une passerelle donnée est recherchée.

```
> route PRINT
> route PRINT 157* .... N'imprime que les adresses commençant par 157*
> route CHANGE 157.0.0.0 MASK 255.0.0.0 157.55.80.5 METRIC 2 IF 2
```

CHANGE est utilisé pour modifier la passerelle et/ou la métrique seulement

.

```
> route PRINT
> route DELETE 157.0.0.0
```

```
> route PRINT
```

- Supposons que la configuration IP actuelle d'un hôte est la suivante :

```
C:>ipconfig
```

```
Configuration IP de Windows
```

```
Carte Ethernet Connexion au réseau local:
```

```
Suffixe DNS propre à la connexion :
Adresse IP. . . . . : 10.0.2.15
Masque de sous-réseau . . . . . : 255.255.255.0
Passerelle par défaut . . . . . : 10.0.2.2
```

- Étudions l'affichage de la table de routage :

```
C:>route print
```

```
=====
```

```
Liste d'Interfaces
```

```
0x1 ..... MS TCP Loopback interface
0x2 ...08 00 27 71 76 54 ..... Carte AMD PCNET Family Ethernet PCI - Mini...
```

```
=====
```

```
Itinéraires actifs :
```

Destination réseau	Masque réseau	Adr. passerelle	Adr. interface	Métrique
0.0.0.0	0.0.0.0	10.0.2.2	10.0.2.15	20
10.0.2.0	255.255.255.0	10.0.2.15	10.0.2.15	20
10.0.2.15	255.255.255.255	127.0.0.1	127.0.0.1	20
10.0.2.255	255.255.255.255	10.0.2.15	10.0.2.15	20
127.0.0.0	255.0.0.0	127.0.0.1	127.0.0.1	1
224.0.0.0	240.0.0.0	10.0.2.15	10.0.2.15	20
255.255.255.255	255.255.255.255	10.0.2.15	10.0.2.15	1

```
Passerelle par défaut : 10.0.2.2
```

```
=====
```

```
Itinéraires persistants :
```

```
Aucun
```


- ➡ La première partie de la commande liste les interfaces réseau de l'hôte. Windows attribue un numéro à chaque interface. Ce numéro est requis pour l'ajout (**ADD**) et la modification (**CHANGE**) d'une route avec **route**. Dans cet exemple, il y a 2 interfaces :

- ◇ l'interface **1** (0x1) est l'interface *loopback* ;
- ◇ l'interface **2** (0x2) correspond à la carte réseau de l'hôte. Elle a pour adresse MAC 08-00-27-71-76-54 (selon l'écriture Windows).


On pourra faire référence à une interface en utilisant son numéro en hexadécimal (en laissant le 0x) ou en décimal.

Ensuite, **route** affiche la table de routage qui contient beaucoup plus de lignes que sous Linux (ou que sur un routeur CISCO) sans pour autant ajouter de fonctionnalité supplémentaire. Les routes importantes ont été mises en évidence en gras dans l'exemple :


- ◇ la 1^{re} ligne correspond à la route par défaut qui passe par le routeur (passerelle) 10.0.2.2 ;

 on peut remarquer que la colonne *Adr. interface* indique l'adresse IP de l'interface à utiliser pour la route.

- ◇ la 2^e ligne correspond à la route directe vers le réseau local ;

 on peut remarquer qu'au lieu de l'adresse 0.0.0.0 comme passerelle pour les destinations directement accessibles (comme sous Linux), Windows affiche l'adresse IP de l'hôte sur ce réseau.

- ◇ la 3^e ligne correspond à l'adresse IP de l'hôte. Elle indique que si l'hôte doit discuter avec lui-même, il doit utiliser l'interface *loopback* ;
- ◇ la 4^e ligne correspond à l'adresse de diffusion dirigée dans le réseau local ;
- ◇ la 5^e ligne correspond à l'interface *loopback* et regroupe toutes les adresses commençant par 127 ;
- ◇ la 6^e ligne correspond à une route pour les adresses de classe D (multidiffusion). Ce type d'adresse est aussi compris par Linux mais il ne les fait pas apparaître dans la table de routage ;
- ◇ la dernière route correspond à l'adresse de diffusion limitée.

 On peut remarquer que Linux n'afficherait que les 2 premières routes tout en gérant les adresses formant les autres routes affichées par Windows.

- Modification de la route par défaut pour passer par le routeur 10.0.2.1 via l'interface 2 :

```
C:>route CHANGE 0.0.0.0 MASK 0.0.0.0 10.0.2.1 METRIC 2 IF 2
```

- Vérification de la modification :

```
C:>route print
```

```
=====
```

```
Liste d'Interfaces
```

```
0x1 ..... MS TCP Loopback interface
0x2 ...08 00 27 71 76 54 ..... Carte AMD PCNET Family Ethernet PCI - Mini...
```

```
=====
```

```
=====
```

```
Itinéraires actifs :
```

Destination réseau	Masque réseau	Adr. passerelle	Adr. interface	Métrique
0.0.0.0	0.0.0.0	10.0.2.1	10.0.2.15	2
10.0.2.0	255.255.255.0	10.0.2.15	10.0.2.15	20
10.0.2.15	255.255.255.255	127.0.0.1	127.0.0.1	20
10.255.255.255	255.255.255.255	10.0.2.15	10.0.2.15	20
127.0.0.0	255.0.0.0	127.0.0.1	127.0.0.1	1
224.0.0.0	240.0.0.0	10.0.2.15	10.0.2.15	20
255.255.255.255	255.255.255.255	10.0.2.15	10.0.2.15	1

```
Passerelle par défaut : 10.0.2.1
```

```
=====
```

```
Itinéraires persistants :
```

```
Aucun
```



2.6 tracert

La commande **tracert** offre la même fonctionnalité principale que **traceroute** sous Linux, mais admet un nombre réduit d'options.

Exemple 16

- Obtenir de l'aide sur **tracert** :

```
C:>tracert
```

```
Utilisation : tracert [-d] [-h SautsMaxi] [-j ListeHôtes] [-w délai] NomCible
```

Options :

-d	Ne pas convertir les adresses en noms d'hôtes.
-h SautsMaxi	Nombre maximum de sauts pour rechercher la cible.
-j ListeHôtes	Itinéraire source libre parmi la liste des hôtes.
-w délai	Attente d'un délai en millisecondes pour chaque réponse.

- Tracer la route vers `www.univmed.fr` en désactivant la résolution inverse :

```
C:>tracert -d www.univmed.fr
```

Détermination de l'itinéraire vers `mozart.pg.univmed.fr` [139.124.196.119]
avec un maximum de 30 sauts :

1	<1 ms	1 ms	<1 ms	10.0.2.2
2	*	*	*	Délai d'attente de la demande dépassé.
3	<1 ms	<1 ms	<1 ms	193.50.131.177
4	5 ms	1 ms	2 ms	192.168.100.33
5	2 ms	1 ms	1 ms	193.50.131.18
6	3 ms	2 ms	2 ms	193.50.131.25
7	3 ms	2 ms	1 ms	139.124.196.119

Itinéraire déterminé.

- Tracer la route vers `www.google.fr` en limitant à 20 sauts :

```
C:>tracert -h 20 www.google.fr
```

Détermination de l'itinéraire vers `www-cctld.l.google.com` [74.125.230.216]
avec un maximum de 20 sauts :

1	<1 ms	<1 ms	<1 ms	10.0.2.2
2	*	*	*	Délai d'attente de la demande dépassé.
3	*	*	*	Délai d'attente de la demande dépassé.
4	2 ms	<1 ms	<1 ms	193.50.131.225
5	2 ms	4 ms	2 ms	192.168.100.33
6	37 ms	2 ms	2 ms	192.168.100.2

7	15 ms	43 ms	50 ms	vl10-gi8-2-marseille1-rtr-021.noc.renater.fr [193.51.181.182]
8	12 ms	9 ms	14 ms	tel-1-marseille2-rtr-021.noc.renater.fr [193.51.179.158]
9	11 ms	10 ms	11 ms	193.51.179.182
10	14 ms	10 ms	19 ms	te0-0-0-1-paris2-rtr-001.noc.renater.fr [193.51.189.2]
11	21 ms	13 ms	10 ms	tel-1-paris2-rtr-021.noc.renater.fr [193.51.189.9]
12	10 ms	13 ms	11 ms	193.51.182.197
13	23 ms	14 ms	17 ms	72.14.238.228
14	17 ms	12 ms	13 ms	209.85.242.49
15	8 ms	18 ms	12 ms	par08s09-in-f24.1e100.net [74.125.230.216]

Itinéraire déterminé.

