

Configuration d'une interface réseau

Philippe Latu

philippe.latu(at)inetdoc.net

<http://www.inetdoc.net>

Ce support de travaux pratiques est destiné aux débutants. Il traite la configuration d'une interface réseau sur un système GNU/Linux. Les manipulations présentées décrivent les couches de la modélisation réseau en partant du niveau physique. Les questions cherchent à illustrer les relations entre les différents formats d'adressage utilisés à chaque niveau.

Table des matières

1. Copyright et Licence	1
1.1. Méta-information	1
1.2. Conventions typographiques	2
1.3. Méthodologie et modélisation	2
2. Identification des interfaces disponibles	2
2.1. Comment identifier le périphérique réseau ?	2
2.2. Comment identifier le pilote logiciel ?	3
2.3. Comment charger et valider le pilote logiciel ?	4
2.4. Comment visualiser l'état du lien ?	4
3. Configuration d'une interface	5
3.1. Commande ifconfig	5
3.1.1. Etat de l'interface	5
3.1.2. Configurer l'interface	5
3.2. Rendre la configuration permanente	6
4. Tests de communication ICMP	6
4.1. Etat de la pile TCP/IP	6
4.2. Tests vers l'extérieur	7
4.3. Tests de la résolution des noms	7
5. Localisation des hôtes du réseau local	8
6. Résolution des noms de domaines et d'hôtes	8
7. Travaux pratiques	10
7.1. Relevé des paramètres existants	10
7.2. Reconfiguration de l'interface	11
8. Table de routage locale	12
8.1. Commande route	12
8.2. Commande traceroute	12
8.3. Travaux pratiques	14
9. Fonctions réseau d'une interface	14
9.1. Comment visualiser les paramètres du noyau ?	14
9.2. Comment changer les valeurs des paramètres ?	14
9.3. Travaux pratiques	15
9.4. Pour aller plus loin !	16

1. Copyright et Licence

Copyright (c) 2000,2012 Philippe Latu.
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Copyright (c) 2000,2012 Philippe Latu.
Permission est accordée de copier, distribuer et/ou modifier ce document selon les termes de la Licence de Documentation Libre GNU (GNU Free Documentation License), version 1.3 ou toute version ultérieure publiée par la Free Software Foundation ; sans Sections Invariables ; sans Texte de Première de Couverture, et sans Texte de Quatrième de Couverture. Une copie de la présente Licence est incluse dans la section intitulée « Licence de Documentation Libre GNU ».

1.1. Méta-information

Cet article est écrit avec *DocBook*¹ XML sur un système *Debian GNU/Linux*². Il est disponible en version imprimable au format PDF : [config.interface.lan.pdf](http://www.inetdoc.net/pdf/config.interface.lan.pdf)³.

¹ <http://www.docbook.org>

² <http://www.debian.org>

³ <http://www.inetdoc.net/pdf/config.interface.lan.pdf>

Toutes les commandes utilisées dans ce document ne sont pas spécifiques à une version particulière des systèmes UNIX ou GNU/Linux. C'est la distribution *Debian GNU/Linux* qui est utilisée pour les tests présentés. Voici une liste des paquets contenant les commandes :

- procs - The /proc file system utilities
- pciutils - Linux PCI Utilities
- net-tools - The NET-3 networking toolkit
- ifupdown - High level tools to configure network interfaces
- iputils-ping - Tools to test the reachability of network hosts
- bind9-host - Version of 'host' bundled with BIND 9.X
- dnstools - Clients provided with BIND
- traceroute - Traces the route taken by packets over a TCP/IP network
- hping2 - Active Network Smashing Tool

1.2. Conventions typographiques

Tous les exemples d'exécution des commandes sont précédés d'une invite utilisateur ou *prompt* spécifique au niveau des droits utilisateurs nécessaires sur le système.

- Toute commande précédée de l'invite \$ ne nécessite aucun privilège particulier et peut être utilisée au niveau utilisateur simple.
- Toute commande précédée de l'invite # nécessite les privilèges du super-utilisateur.

1.3. Méthodologie et modélisation

Ce support a pour but de fournir une méthodologie de dépannage simple d'une connexion réseau pour chaque couche de la modélisation contemporaine.

- Couche Physique (1) : identification de l'interface dans les messages systèmes.
- Couche Liaison (2) : manipulation des adresses MAC avec la commande **arp**.
- Couche Réseau (3) : manipulation des adresses IP avec la commande **ifconfig**, de l'adresse de la passerelle par défaut avec la commande **route** et tests de communication ICMP avec la commande **ping**.
- Couche Transport (4) : configuration de la résolution des noms et tests avec les commandes **host** et **dig**.
- Synthèse : validation des communications en utilisant la commande **traceroute** avec et sans résolution des noms d'hôtes.

2. Identification des interfaces disponibles

Avant de pouvoir configurer une interface, il faut que le pilote de périphérique correspondant ait été chargé en mémoire. Comme une interface réseau est un dispositif matériel, c'est au niveau du noyau Linux que l'opération doit s'effectuer. Soit le pilote d'interface a été inclus dans la partie monolithique du noyau soit il est chargé en mémoire sous forme de module. C'est cette dernière solution qui est le plus souvent retenue. Un module peut être chargé ou déchargé à volonté sans avoir à redémarrer la machine. De plus, les fonctions de reconnaissance automatique des composants périphériques permettent de ne charger que les modules correspondant aux composants effectivement présents sur le système.

2.1. Comment identifier le périphérique réseau ?

Il existe une grande variété de contrôleurs réseau Ethernet. À chaque composant correspond un pilote logiciel spécifique. Qu'il s'agisse d'une carte additionnelle ou d'un composant intégré sur la carte mère, le contrôleur est toujours un périphérique connecté au bus PCI. En mode console et à partir de la connexion super-utilisateur, la commande **lspci** du paquet **pciutils** donne la liste des périphériques reliés au bus PCI. Voici quelques exemples caractéristiques.

Identification de deux contrôleurs de marque Broadcom™ intégrés sur la carte mère d'un serveur rackable Dell™. On remarque, au niveau des deux dernières lignes d'informations sur les interfaces, que le module correspondant au contrôleur BCM5704 est baptisé **tg3**.

```
# lspci -v
<snip/>
15:02.0 Ethernet controller: Broadcom Corporation NetXtreme BCM5704 Gigabit Ethernet
Subsystem: Dell Unknown device 0170
Flags: bus master, 66MHz, medium devsel, latency 64, IRQ 64
Memory at df2f0000 (64-bit, non-prefetchable) [size=64K]
Expansion ROM at <ignored> [disabled]
Capabilities: [40] PCI-X non-bridge device
Capabilities: [48] Power Management version 2
```

```

Capabilities: [50] Vital Product Data <?>
Capabilities: [58] Message Signalled Interrupts: Mask- 64bit+ Queue=0/3 Enable-
Kernel driver in use: tg3
Kernel modules: tg3

15:02.1 Ethernet controller: Broadcom Corporation NetXtreme BCM5704 Gigabit Ethernet
Subsystem: Dell Unknown device 0170
Flags: bus master, 66MHz, medium devsel, latency 64, IRQ 65
Memory at df2e0000 (64-bit, non-prefetchable) [size=64K]
Expansion ROM at <ignored> [disabled]
Capabilities: [40] PCI-X non-bridge device
Capabilities: [48] Power Management version 2
Capabilities: [50] Vital Product Data <?>
Capabilities: [58] Message Signalled Interrupts: Mask- 64bit+ Queue=0/3 Enable-
Kernel driver in use: tg3
Kernel modules: tg3
<snip/>

```

Identification d'un contrôleur de marque Intel™. Comme dans l'exemple précédent, on remarque que les deux dernières lignes d'informations indiquent le nom du module correspondant au contrôleur 82572EI : e1000.

```

# lspci -v
<snip/>
0c:00.0 Ethernet controller: Intel Corporation 82572EI Gigabit Ethernet Controller (Copper)
Subsystem: Intel Corporation PRO/1000 PT Server Adapter
Flags: bus master, fast devsel, latency 0, IRQ 16
Memory at fc4e0000 (32-bit, non-prefetchable) [size=128K]
Memory at fc4c0000 (32-bit, non-prefetchable) [size=128K]
I/O ports at ece0 [size=32]
Capabilities: [c8] Power Management version 2
Capabilities: [d0] Message Signalled Interrupts: Mask- 64bit+ Queue=0/0 Enable-
Capabilities: [e0] Express Endpoint, MSI 00
Capabilities: [100] Advanced Error Reporting <?>
Capabilities: [140] Device Serial Number a9-9b-19-ff-ff-17-15-00
Kernel driver in use: e1000
Kernel modules: e1000
<snip/>

```

2.2. Comment identifier le pilote logiciel ?

Comme précisé plus haut, les contrôleurs réseau utilisent des composants matériel. On retrouve donc les informations sur l'initialisation des interfaces réseau dans les messages relatifs au lancement du système d'exploitation. La commande historique **dmesg** (*dump messages*) permet d'afficher tous les messages systèmes à la console. Pour isoler les parties relatives aux interfaces réseau, on filtre cet affichage avec la commande **grep** en recherchant des mots clés particuliers : noms de modules, noms d'interfaces, noms de marques, etc.

Exemple de recherche basée sur le nom du module logiciel de pilotage de l'interface :

```

# dmesg |grep e1000
e1000: 0000:0c:00.0: e1000_probe: (PCI Express:2.5Gb/s:Width x1) 00:15:17:19:9b:a9
e1000: eth1: e1000_probe: Intel(R) PRO/1000 Network Connection

```

Exemple de recherche basée sur la marque du composant contrôleur réseau :

```

# dmesg |grep -i broadcom
Broadcom NetXtreme II Gigabit Ethernet Driver bnx2 v1.6.9 (December 8, 2007)
eth0: Broadcom NetXtreme II BCM5708 1000Base-T (B2) PCI-X 64-bit 133MHz found at \
mem f4000000, IRQ 16, node addr 00:1a:a0:01:f0:32
eth2: Broadcom NetXtreme II BCM5708 1000Base-T (B2) PCI-X 64-bit 133MHz found at \
mem f8000000, IRQ 16, node addr 00:1a:a0:01:f0:30

```

Avec un peu plus d'expérience, on peut naviguer dans les menus de configuration du noyau Linux et consulter le catalogue des interfaces réseau disponibles par catégories de liaisons. Pour les interfaces gigabit Ethernet, on obtient la liste suivante :

```

--- Ethernet (1000 Mbit)
< > Alteon AceNIC/3Com 3C985/NetGear GA620 Gigabit support
< > DL2000/TC902x-based Gigabit Ethernet support
< > Intel(R) PRO/1000 Gigabit Ethernet support
<M> Intel(R) PRO/1000 PCI-Express Gigabit Ethernet support
< > IP1000 Gigabit Ethernet support (NEW)
< > National Semiconductor DP83820 support
< > Packet Engines Hamachi GNIC-II support
< > Packet Engines Yellowfin Gigabit-NIC support (EXPERIMENTAL)
< > Realtek 8169 gigabit ethernet support
< > SiS190/SiS191 gigabit ethernet support
< > New SysKonnnect GigaEthernet support
< > SysKonnnect Yukon2 support
< > Marvell Yukon Chipset / SysKonnnect SK-98xx Support (DEPRECATED)
< > VIA Velocity support
<M> Broadcom Tigon3 support
< > Broadcom NetXtremeII support
< > QLogic QLA3XXX Network Driver Support
< > Attansic L1 Gigabit Ethernet support (EXPERIMENTAL)

```

Liste des interfaces gigabit Ethernet du noyau Linux - vue complète⁴

2.3. Comment charger et valider le pilote logiciel ?

Avec une distribution GNU/Linux moderne, le chargement des modules de pilotage des interfaces réseau se fait automatiquement. Il s'agit justement d'un point fort du noyau Linux ; il ne charge en mémoire que les pilotes correspondant aux composants effectivement présents sur le système. Cette section présente les opérations manuelles de chargement et déchargement de pilote de contrôleur réseau.

Le chargement manuel d'un module s'effectue avec la commande **modprobe** et la validation du résultat avec les commandes **dmesg** et **ifconfig**. De plus, on peut vérifier la présence du pilote dans la liste des modules chargé avec la commande **lsmod**.

**Avertissement**

Pour tester les manipulations ci-dessous à partir d'une configuration déjà établie, il faut :

- désactiver la configuration de l'interface : `/etc/init.d/networking stop`,
- retrouver le pilote de l'interface dans la liste des modules : `lsmod`,
- décharger le module pilote de cette même interface : `rmmod <module_carte>`,
- Attention ! Lors du (re)chargement du module pilote de l'interface, les scripts de configuration de l'interface sont automatiquement relancés. Le système effectue une opération identique à la commande : `/etc/init.d/networking start`.

Enfin, si le pilote de carte est chargé dans la partie monolithique du noyau, toute manipulation de module est impossible.

En reprenant l'exemple du contrôleur de marque Intel™, on peut effectuer les manipulations suivantes :

- Identification du module et relevé du nom de l'interface : eth1

```
# dmesg |grep e1000
e1000: 0000:0c:00.0: e1000_probe: (PCI Express:2.5Gb/s:Width x1) 00:15:17:19:9b:a9
e1000: eth2: e1000_probe: Intel(R) PRO/1000 Network Connection
```

- Interface non configurée : sans indicateur UP et sans adresse IP.

```
# ifconfig eth1
eth2      Link encap:Ethernet  HWaddr 00:15:17:19:9b:a9
          BROADCAST MULTICAST  MTU:1500  Metric:1
<snip/>
```

- Déchargement du module de pilotage de l'interface.

```
# modprobe -rv e1000
rmmod /lib/modules/2.6.24.3/kernel/drivers/net/e1000/e1000.ko
```

- Chargement du module de pilotage de l'interface.

```
# modprobe -v e1000
insmod /lib/modules/2.6.24.3/kernel/drivers/net/e1000/e1000.ko
```

- Consultation des messages systèmes avec la commande **dmesg**.

```
ACPI: PCI Interrupt 0000:0c:00.0[A] -> GSI 16 (level, low) -> IRQ 16
PCI: Setting latency timer of device 0000:0c:00.0 to 64
e1000: 0000:0c:00.0: e1000_probe: (PCI Express:2.5Gb/s:Width x1) 00:15:17:19:9b:a9
e1000: eth2: e1000_probe: Intel(R) PRO/1000 Network Connection
```

2.4. Comment visualiser l'état du lien ?

Même avec une configuration correcte de l'interface, il est possible que les communications soient bloquées si le lien physique entre l'hôte et l'équipement réseau n'est pas actif. Sur les liaisons utilisant des câbles en paires torsadées cuivre, on peut visualiser l'état du lien à l'aide de la commande **mii-tool**.

```
# mii-tool -v
eth0: negotiated 1000baseT-FD flow-control, link ok❶
  product info: vendor 00:08:18, model 54 rev 6
  basic mode:   autonegotiation enabled
  basic status: autonegotiation complete, link ok
  capabilities: 1000baseT-FD 100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD❷
  advertising:  1000baseT-FD 100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD flow-control
  link partner: 1000baseT-FD 100baseTx-FD 100baseTx-HD 10baseT-FD 10baseT-HD❸
```

- ❶ Le lien entre l'interface eth0 et l'équipement réseau est actif et le débit négocié est de 1000Mbps en mode Full-Duplex avec contrôle de flux.

⁴ <http://www.linux-france.org/prj/inetdoc/cours/config.interface.lan/images/kernelgbelist.png>

- ② Les débits possibles sur cette interface sont : 1000Mbps en mode Full-Duplex, 100Mbps en mode Full-Duplex, 100Mbps en mode Half-Duplex, 10Mbps en mode Full-Duplex et 10Mbps en mode Half-Duplex.
- ③ Les débits offerts par l'équipement réseau sont identiques à ceux disponibles sur l'interface de l'hôte.

Pour aller plus loin dans l'étude des caractéristiques techniques des réseaux locaux, il est conseillé de lire l'article *Technologie Ethernet*⁵.

3. Configuration d'une interface

Pour configurer une interface réseau, il faut utiliser les commandes de base disponibles sur n'importe quel système Unix. Voici une présentation succincte des commandes classiques de configuration et de test d'une connexion réseau : **ifconfig**, **ping**, **arp**, **host** et **dig**.

3.1. Commande ifconfig

Pour toute information sur le format des adresses IP utilisées ci-après, se référer à l'article *Adressage IPv4*⁶. **ifconfig** sert à fixer les paramètres d'une interface ; eth0 dans notre exemple.

3.1.1. Etat de l'interface

```
$ /sbin/ifconfig -a
eth0      Lien encap:Ethernet  HWaddr 00:50:04:4C:28:27 ①
          inet adr:192.168.1.1  Bcast:192.168.1.255  Masque:255.255.255.0 ②
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1 ③
          Paquets Reçus:134 erreurs:0 jetés:0 débordements:0 trames:0 ④
          Paquets transmis:17 erreurs:0 jetés:0 débordements:0 carrier:0
          collisions:0 lg file transmission:100
          Interruption:10 Adresse de base:0xe000 ⑤

lo        Lien encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          UP LOOPBACK RUNNING  MTU:3924  Metric:1
          Paquets Reçus:13599 erreurs:0 jetés:0 débordements:0 trames:0
          Paquets transmis:13599 erreurs:0 jetés:0 débordements:0 carrier:0
          collisions:0 lg file transmission:0
```

① Informations sur la couche liaison (2) :

- encap:Ethernet : format de trame Ethernet II.
- HWaddr ... : Adresse MAC de la carte réseau.

② Informations sur la couche réseau (3) :

- inet adr : adresse IP de l'interface.
- Bcast : adresse de diffusion du réseau.
- Masque : masque de sous-réseau.

③ Informations sur l'état de l'interface :

- UP BROADCAST RUNNING MULTICAST : interface de diffusion active.
- MTU:1500 : *Maximum Transmission Unit*. La taille maximum des trames Ethernet transmises sur Internet est fixée par le document *RFC1191 Path MTU discovery*⁷.

- Metric:1 : nombre de sauts autorisés pour obtenir un routage vers n'importe quelle destination.

④ Statistiques de l'interface. Ces informations sont essentielles pour déterminer la *qualité* du réseau.

⑤ Paramètres d'entrées/sorties de l'interface. Ces informations indiquent si la carte réseau est correctement reconnue par le système.

3.1.2. Configurer l'interface

Typiquement, on configure une interface Ethernet avec une commande du type :

```
# ifconfig eth0 192.168.1.1 netmask 255.255.255.0 broadcast 192.168.1.255 up
```

La commande **ifconfig** possède de nombreuses options. Les principales sont :

- up : activation de l'interface,
- down : désactivation de l'interface,
- [-]arp : activation/désactivation du protocole ARP sur l'interface,
- netmask <addr> : valeur du masque de réseau,

⁵ <http://www.inetdoc.net/articles/ethernet/>

⁶ <http://www.inetdoc.net/articles/adressage.ipv4/>

⁷ <http://www.faqs.org/rfcs/rfc1191.html>

- broadcast <addr> : valeur de l'adresse de diffusion.

Pour obtenir la syntaxe de toutes les options disponibles, il faut utiliser la commande **man ifconfig** ou **kdehelp** : System man page contents → Section 8 Administration système → ifconfig.

3.2. Rendre la configuration permanente

Avec la distribution *Debian GNU/Linux* les paramètres de configuration des interfaces réseau sont stockés dans le répertoire `/etc/network`. Le fichier `interfaces` de ce répertoire rassemble la configuration des interfaces réseau.

Voici l'exemple d'une interface ethernet configurée à l'aide du protocole DHCP :

```
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)

# The loopback interface
auto lo
iface lo inet loopback

# The first network card - this entry was created during the Debian installation
# (network, broadcast and gateway are optional)
auto eth0
iface eth0 inet dhcp
```

Pour une configuration statique de l'interface, il faut utiliser les pages de manuels : **man interfaces**. Voici un exemple :

```
<snip/>
auto eth0
iface eth0 inet static
    address 192.168.1.1
    netmask 255.255.255.0
    network 192.168.1.0
    broadcast 192.168.1.255
```

4. Tests de communication ICMP

Commande ping

Le protocole *Internet Control Message Protocol* ou ICMP est décrit dans le document [RFC792 Internet Control Message Protocol](#)⁸. Comme le protocole IP de la couche réseau fonctionne en mode non connecté, il ne fournit aucun service de contrôle lors de la transmission des paquets sur le réseau. Le rôle du protocole ICMP est de notifier l'émetteur lorsqu'il y a eu un problème.

La commande **ping** utilise principalement deux types de messages du protocole ICMP pour informer l'utilisateur sur les conditions de transmissions :

- L'hôte distant est-il actif ou inactif.
- Le temps de propagation en boucle (*round-trip delay*) lors de la communication avec l'hôte distant.
- Les pertes de paquets pendant la communication.

Il existe 18 types de messages ICMP. Les deux types de messages employés par la commande **ping** sont :

- Le type 8 (echo request) est émis vers l'hôte distant.
- Le type 0 (echo reply) est émis par l'hôte distant en réponse.

Quelques autres types sont abordés dans la partie [Section 9, « Fonctions réseau d'une interface »](#).

Pour valider le bon fonctionnement d'une communication sur un réseau IP, on suit une séquence précise de tests :

1. adresse IP de l'interface de boucle locale : `lo`,
2. adresse IP de l'interface du poste de travail : `eth0` ou `ppp0`,
3. adresse IP du destinataire de la passerelle par défaut,
4. adresse IP extérieure au réseau local.

4.1. Etat de la pile TCP/IP

La commande suivante permet de valider le fonctionnement du protocole réseau IP pour les processus appartenant au même système. On parle alors de validation *inter-processus*.

```
$ ping -c 2 127.0.0.1
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=255 time=0.1 ms
```

⁸ <http://www.faqs.org/rfcs/rfc792.html>

```
64 bytes from 127.0.0.1: icmp_seq=1 ttl=255 time=0.1 ms
--- 127.0.0.1 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.1/0.1/0.1 ms
```

Il s'agit ici de contrôler que les processus pairs à l'intérieur du même système sont capables de dialoguer entre eux.

On teste ensuite le fonctionnement de l'interface seule :

```
$ ping -c 2 192.168.1.1
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=255 time=0.1 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=255 time=0.1 ms
--- 192.168.1.1 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.1/0.1/0.1 ms
```

Il s'agit ici de contrôler que l'interface réseau est bien configurée et active.

Une fois ces deux étapes franchies, on peut tester les communications avec les autres systèmes.

4.2. Tests vers l'extérieur

Exemple d'échec :

```
$ ping -c 5 192.168.1.14
PING 192.168.1.14 (192.168.1.14): 56 data bytes
--- 192.168.1.14 ping statistics ---
5 packets transmitted, 0 packets received, 100% packet loss
```

Exemple de succès :

```
$ ping -c 2 192.168.1.13
PING 192.168.1.13 (192.168.1.13): 56 data bytes
64 bytes from 192.168.1.13: ❶ icmp_seq=0❷ ttl=255❸ time=1.1 ms
64 bytes from 192.168.1.13: icmp_seq=1 ttl=255 time=0.8 ms
--- 192.168.1.13 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.8/0.9/1.1 ms
```

- ❶ Adresse de réponse du message ICMP : destinataire du test.
- ❷ Numéro de séquence du message.
- ❸ La valeur du champ TTL d'un paquet IP correspond au nombre d'interfaces de routage traversées pour arriver à l'interface.

Pour obtenir la syntaxe de toutes les options disponibles, il faut accéder aux pages de manuels Unix :

- via la console avec la commande `man ping`.
- via l'interface graphique avec le centre d'aide de KDE : Pages de manuels Unix → Section 8 Administration système → ping.

4.3. Tests de la résolution des noms

Cette commande est aussi très utile pour savoir si la résolution des noms d'hôtes fonctionne correctement. Dans ce cas on fait appel à un service Internet appelé *Domain Name Service* (DNS). Cet appel au service DNS nécessite un minimum de configuration.

```
$ ping -c 5 www.nic.fr ❶
PING rigolo.nic.fr (192.134.4.20)❷: 56 data bytes
64 bytes from 192.134.4.20: icmp_seq=0 ttl=54 time=57.6 ms
64 bytes from 192.134.4.20: icmp_seq=1 ttl=54 time=51.0 ms
64 bytes from 192.134.4.20: icmp_seq=2 ttl=54 time=57.0 ms
64 bytes from 192.134.4.20: icmp_seq=3 ttl=54 time=109.8 ms
64 bytes from 192.134.4.20: icmp_seq=4 ttl=54 time=165.3 ms
--- rigolo.nic.fr ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 51.0/88.1/165.3 ms
```

- ❶ Utilisation de la commande `ping` avec un nom d'hôte au lieu d'une adresse IP.
- ❷ Affichage de la correspondance entre le nom de l'hôte et son adresse IP.

En cas d'échec sur la résolution des noms, il faut contrôler la validité des informations dans les deux fichiers suivants :

- `/etc/resolv.conf`

```
search <domaine-fai>.fr ❶
nameserver <addr dns-fai>❷
```

- ❶ Nom du domaine auquel l'interface de l'hôte est connectée.
- ❷ Adresse IP du serveur de noms qui doit résoudre toutes les requêtes au service DNS.

- /etc/host.conf

```
order hosts, bind❶
multi on
```

- ❶ Ordre de recherche des noms d'hôtes. Dans le cas présenté, la recherche est d'abord effectuée localement puis à l'aide du service DNS.

5. Localisation des hôtes du réseau local

Commande **arp**

La commande **arp** utilise le protocole du même nom : *Address Resolution Protocol* décrit dans le document [RFC826 Ethernet Address Resolution Protocol](#)⁹.

Elle sert à localiser un hôte du réseau local en faisant la correspondance entre l'adresse IP et l'adresse MAC de cet hôte.

Entre deux hôtes d'un même réseau, il n'existe pas de service de «détermination du chemin à suivre» (c'est le travail des routeurs entre réseaux différents). La tâche du protocole ARP est donc indispensable pour la communication entre les hôtes d'un réseau local.

Dans l'exemple suivant, on visualise la table des adresses MAC connues avec la commande **arp**.

```
# arp
Adresse      TypeMap      AdresseMat    Indicateurs   Iface
router       ether        00:60:3E:10:48:20  C              eth0
dns          ether        00:A0:24:A0:A4:11  C              eth0
```

On effectue une *localisation* sur le réseau local avec la commande **ping**.

```
$ ping -c 2 server
PING server (192.168.10.10) from 192.168.10.34 : 56(84) bytes of data.
64 bytes from server (192.168.10.10): icmp_seq=0 ttl=128 time=0.9 ms
64 bytes from server (192.168.10.10): icmp_seq=1 ttl=128 time=0.4 ms

--- server ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.4/0.6/0.9 ms
```

Le résultat de la *localisation* apparaît lorsque l'on visualise à nouveau la table des adresses MAC.

```
# arp
Adresse      TypeMap      AdresseMat    Indicateurs   Iface
router       ether        00:60:3E:10:48:20  C              eth0
dns          ether        00:A0:24:A0:A4:11  C              eth0
server       ether        00:60:97:91:60:0A  C              eth0
```

La table des adresses MAC est maintenue dynamiquement en fonction du trafic reçu par les interfaces. Les entrées valides sont contrôlées toutes les 30 secondes sans qu'il y ait émission de trafic. Ensuite, les entrées passent phase de «gel» (*stale*) pendant 60 secondes avant d'être effacées.

Pour obtenir la syntaxe de toutes les options disponibles, il faut accéder aux pages de manuels Unix :

- via la console avec la commande **man arp**.
- via l'interface graphique avec le centre d'aide KDE :Pages de manuels Unix → Section 8 Administration système → arp.

6. Résolution des noms de domaines et d'hôtes

Commandes **host** & **dig**

L'opération de *résolution* consiste à faire la correspondance entre un nom de domaine et son adresse IP. Pour effectuer ces opérations de résolution on utilise un service Internet particulier appelé *Domain Name System* ou DNS. Ce service fonctionne sur le même mode qu'un annuaire téléphonique dans lequel les chiffres du numéro de téléphone sont remplacés par les chiffres de l'adresse IP et le nom d'abonné est remplacé par le nom de domaine.

La commande historique **host** effectue l'opération de recherche de l'adresse IP correspondant à un nom d'hôte réseau enregistré sur le service DNS et vice versa :

- Résolution d'un nom d'hôte.

```
$ host www.nic.fr
www.nic.fr is an alias for rigolo.nic.fr.
rigolo.nic.fr has address 192.134.4.20
```

- Résolution d'une adresse IP.

⁹ <http://www.faqs.org/rfcs/rfc826.html>

```
$ host 192.134.4.20
20.4.134.192.in-addr.arpa domain name pointer rigolo.nic.fr.
```

Il existe une autre commande historique qui donne davantage d'informations sur les jeux de questions/réponses DNS : **nslookup**. Cette commande est disponible sur les systèmes Microsoft ; elle est accessible à partir d'un *Shell*. Dans l'univers Unix/Linux, cette commande est abandonnée au profit de la commande **dig** qui offre davantage d'options et des messages d'erreurs plus rigoureux. Voici un exemple d'exécution à partir du même exemple que ci-dessus.

Exemple de requête DNS de type A qui renvoie l'adresse IP correspondant à un nom d'hôte.

```
$ dig www.nic.fr

; <<> DiG 9.4.2 <<> www.nic.fr
;; global options: printcmd
;; Got answer:
;; ->HEADER<<- opcode: QUERY, status: NOERROR, id: 355
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 7, ADDITIONAL: 9

;; QUESTION SECTION:①
;www.nic.fr.                IN      A

;; ANSWER SECTION:②
www.nic.fr.                163437 IN      CNAME  rigolo.nic.fr.
rigolo.nic.fr.            163437 IN      A      192.134.4.20

;; AUTHORITY SECTION:③
nic.fr.                    70358  IN      NS      ns2.nic.fr.
nic.fr.                    70358  IN      NS      ns1.nic.fr.
nic.fr.                    70358  IN      NS      ns1.oleane.net.
nic.fr.                    70358  IN      NS      dns.inria.fr.
nic.fr.                    70358  IN      NS      ns-sec.ripe.net.
nic.fr.                    70358  IN      NS      ns0.oleane.net.
nic.fr.                    70358  IN      NS      ns3.nic.fr.

;; ADDITIONAL SECTION:④
dns.inria.fr.              23315  IN      A      193.51.208.13
ns0.oleane.net.           133065 IN      A      194.2.0.30
ns1.nic.fr.                153582 IN      A      192.93.0.1
ns1.nic.fr.                153582 IN      AAAA   2001:660:3005:1::1:1
ns2.nic.fr.                153582 IN      A      192.93.0.4
ns2.nic.fr.                153582 IN      AAAA   2001:660:3005:1::1:2
ns3.nic.fr.                70358  IN      A      192.134.0.49
ns3.nic.fr.                70358  IN      AAAA   2001:660:3006:1::1:1
ns-sec.ripe.net.          169587 IN      A      193.0.0.196

;; Query time: 22 msec⑤
;; SERVER: 127.0.0.1#53(127.0.0.1)⑥
;; WHEN: Mon Mar  3 17:27:29 2008
;; MSG SIZE rcvd: 395
```

Exemple de requête DNS de type PTR qui renvoie le nom d'hôte correspondant à une adresse IP.

```
$ dig -x 192.134.4.20

; <<> DiG 9.4.2 <<> -x 192.134.4.20
;; global options: printcmd
;; Got answer:
;; ->HEADER<<- opcode: QUERY, status: NOERROR, id: 22357
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 6

;; QUESTION SECTION:①
;20.4.134.192.in-addr.arpa.  IN      PTR

;; ANSWER SECTION:②
20.4.134.192.in-addr.arpa. 163116 IN      PTR      rigolo.nic.fr.

;; AUTHORITY SECTION:③
4.134.192.in-addr.arpa.    163116 IN      NS      ns2.nic.fr.
4.134.192.in-addr.arpa.    163116 IN      NS      ns3.nic.fr.
4.134.192.in-addr.arpa.    163116 IN      NS      ns1.nic.fr.

;; ADDITIONAL SECTION:④
ns3.nic.fr.                70037  IN      A      192.134.0.49
ns3.nic.fr.                70037  IN      AAAA   2001:660:3006:1::1:1
ns1.nic.fr.                153261 IN      A      192.93.0.1
ns1.nic.fr.                153261 IN      AAAA   2001:660:3005:1::1:1
ns2.nic.fr.                153261 IN      A      192.93.0.4
ns2.nic.fr.                153261 IN      AAAA   2001:660:3005:1::1:2

;; Query time: 2 msec⑤
;; SERVER: 127.0.0.1#53(127.0.0.1)⑥
;; WHEN: Mon Mar  3 17:32:50 2008
;; MSG SIZE rcvd: 256
```

Ces exemples montrent que la commande **dig** donne des informations très complètes sur l'état des requêtes.

- ❶ Le champ QUESTION reprend le terme de la requête émise.
- ❷ Le champ ANSWER donne la réponse à la requête.
- ❸ Le champ AUTHORITY donne la liste des serveurs de noms qui ont autorité sur les enregistrements DNS. Ce sont les seuls serveurs aptes à fournir une réponse aux requêtes sur le domaine concerné.
- ❹ Le champ ADDITIONAL donne les adresses IP des serveurs DNS de référence du domaine.
- ❺ Le champ Query time donne le temps de traitement de la requête. La valeur obtenue permet de déduire si le serveur interrogé a déjà la réponse en mémoire cache ou non.
- ❻ Le champ SERVER identifie le serveur qui a pris la requête DNS en charge.

Pour obtenir la syntaxe de toutes les options disponibles, il faut accéder aux pages de manuels Unix :

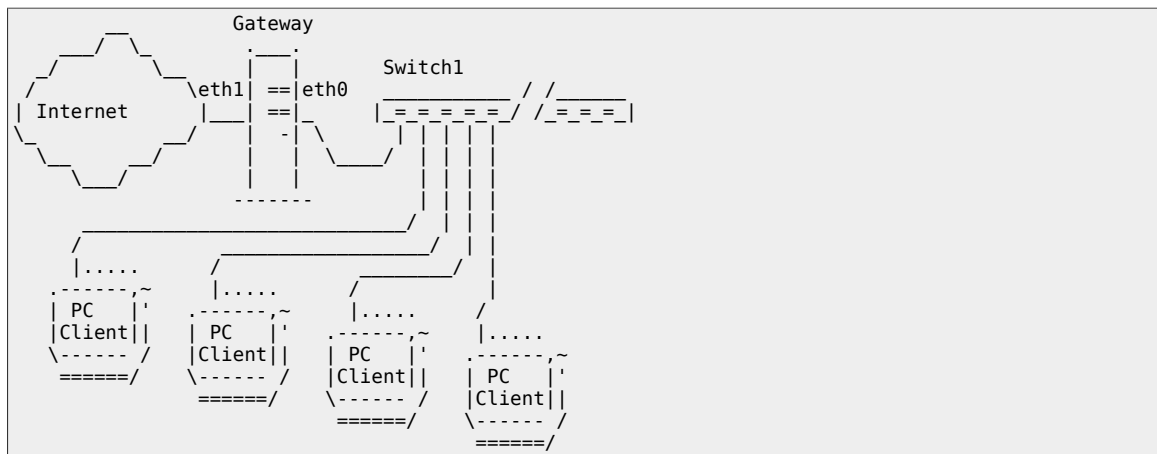
- via la console avec les commandes `man host` et `man dig`.
- via l'interface graphique avec le centre d'aide KDE : Pages de manuels Unix → Section 1 Commandes utilisateur → dig.

Pour aller plus loin dans l'étude du fonctionnement du service de noms de domaines, il est conseillé de lire le support *Introduction au service DNS*¹⁰.

7. Travaux pratiques

7.1. Relevé des paramètres existants

On suppose que chaque client dispose d'une interface déjà configurée avec un accès à un réseau local Ethernet et à d'autres réseaux ; l'Internet par exemple.



1. Quelle est la commande qui permet de visualiser la configuration d'une ou plusieurs interfaces réseau ?
Voir [Section 3, « Configuration d'une interface »](#).
2. Quelles sont les adresses de niveau liaison (MAC) et de niveau réseau (IP) de l'interface Ethernet ?
Voir [Section 3, « Configuration d'une interface »](#).
3. Quelles sont les indications données par la configuration qui montrent que cette interface est active ?
Voir [Section 3, « Configuration d'une interface »](#).
4. Quel est le rôle des adresses MAC par rapport aux adresses IP ?
Retrouver les fonctions de localisation des niveaux liaison et réseau.
5. Quelles indications sont fournies par le masque de réseau et l'adresse de diffusion ?
Retrouver les «limites» de l'adressage logique et de l'adressage matériel.
6. Comment déduire l'adresse IP de l'interface de la passerelle par défaut à partir des éléments précédents ?
Généralement, il s'agit de la première adresse utile du réseau ou sous-réseau auquel l'hôte est connecté. Cette question sera à nouveau traitée dans la partie [routage](#).
7. Comment visualiser la table de correspondance entre les adresses MAC et IP connues de votre station ?
Retrouver la commande à utiliser et observer l'évolution des entrées de la table.

¹⁰ <http://www.inetdoc.net/cours/admin.reseau.dns/>

8. Comment forcer l'ajout d'une entrée dans la table ARP de votre station ?

Retrouver la commande à utiliser et visualiser le(s) résultat(s).

9. Pourquoi des entrées apparaissent dans la table ARP sans action particulière ?

Essayer de repérer les stations du réseau local qui ont contacté votre station.

10. Pourquoi l'adresse MAC de la passerelle par défaut est-elle «presque» toujours présente dans la table d'un poste client ?

Retrouver la fréquence à laquelle les entrées de la table ARP sont rafraîchies. Cette fréquence est-elle fixe ?

11. Pourquoi l'adresse MAC de la passerelle par défaut est-elle «rafraîchie» après un test de la commande **ping** utilisant un nom d'hôte ?

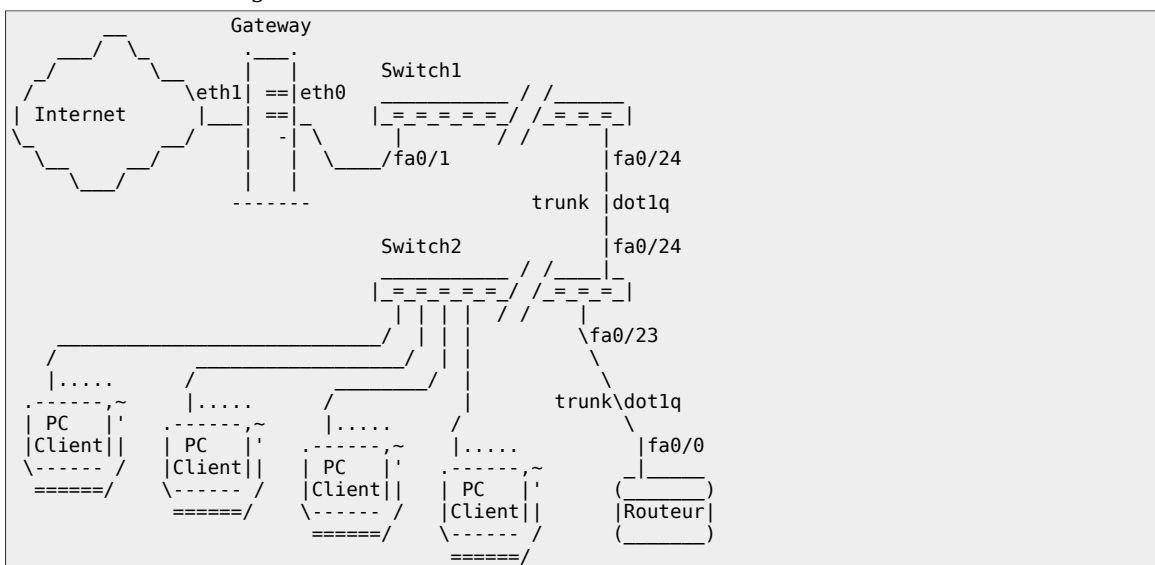
Revoir la configuration de la résolution des noms d'hôtes et localiser le serveur de noms vers lequel toutes les requêtes DNS sont dirigées.

12. Comment vérifier que le serveur DNS indiqué dans les fichiers de configuration prend bien en charge les requêtes émises par votre station ?

Retrouver la commande à utiliser, consulter les pages de manuels correspondantes et repérer l'option qui permet de visualiser le détail des échanges entre client et serveur DNS.

7.2. Reconfiguration de l'interface

Avec la configuration ci-dessous, il est possible de changer l'adressage réseau des postes clients par groupes en utilisant le routage inter VLAN.



Voici un exemple de plan d'adressage pour 4 groupes de postes :

Interface Routeur	association	Ports Commutateur	Switch2
fa0/0.1 - 172.16.80.15/20	VLAN 2	fa0/21 à fa0/24 (trunks)	
fa0/0.2 - 192.168.3.1/24	VLAN 3	fa0/1 à fa0/4 + fa0/5	
fa0/0.3 - 192.168.4.1/24	VLAN 4	fa0/6 à fa0/9 + fa0/10	
fa0/0.4 - 192.168.5.1/24	VLAN 5	fa0/11 à fa0/14 + fa0/15	
fa0/0.5 - 192.168.6.1/24	VLAN 6	fa0/16 à fa0/19 + fa0/20	

Les adresses IP des sous-interfaces du routeur fa0/0.2 à fa0/0.5 sont données à titre indicatif. Elles peuvent être modifiées à volonté. Une fois les questions précédentes traitées, il faut rebrancher la connexion du poste sur un des groupes de prises du commutateur Switch2.

1. A partir de l'adresse de l'interface du routeur utilisée, quels sont les paramètres de l'interface Ethernet du poste ?

Partant du rôle de l'interface du routeur pour le client, déterminer les adresses : réseau, masque de réseau, passerelle par défaut et diffusion. Il ne reste qu'à choisir une adresse pour le poste.

Utiliser la commande **ifconfig** pour appliquer ces paramètres.

2. Comment valider la connectivité vers l'Internet ?

Reprendre la séquence des tests ICMP.

Arrivé à cette étape, les communications entre postes du même réseau local sont possibles mais il manque un élément important pour gagner la connectivité vers les autres réseaux : la passerelle par défaut qui doit renseigner toutes les routes vers les autres réseaux IP.

Cette passerelle par défaut apparaît dans la *table de routage*. Même sur un poste client, une table de routage est nécessaire ! C'est l'objet du point suivant.

8. Table de routage locale

Le routage est un sujet à part entière auquel il faut consacrer beaucoup de temps pour avoir une bonne compréhension des échanges entre plusieurs réseaux. L'objectif de cette section est limité à l'observation des routes connues de l'interface de l'hôte et à la détection de pannes.

8.1. Commande route

La commande **route**, tout comme **ifconfig** sert à la fois à connaître l'état de la table de routage de l'hôte et à configurer de nouvelles routes au besoin.

Cette commande n'a rien à voir avec le routage dynamique qui fonctionne sur un routeur. Elle ne sert qu'à poser des routes statiques entre interfaces.

```
# route -n
Kernel IP routing table
Destination      Gateway         Genmask         Flags① Metric Ref     Use Iface②
192.168.1.0      0.0.0.0        255.255.255.0   U        0      0       0 eth0
0.0.0.0          192.168.1.1    0.0.0.0         UG       0      0       0 eth0
```

① Indicateurs d'état :

- U : Up ; l'interface est active.
- H : Host ; désigne un hôte.
- G : Gateway ; C'est l'interface à partir de laquelle on atteint les autres hôtes/réseaux.

② Interface Ethernet baptisée eth0 sur les systèmes GNU/Linux.

Pour obtenir la syntaxe de toutes les options disponibles, il faut accéder aux pages de manuels Unix :

- via la console avec la commande **man route**.
- via l'interface graphique avec le centre d'aide KDE : Pages de manuels Unix → Section 8 Administration système → route.

8.2. Commande traceroute

traceroute renvoie les informations sur la route suivie pour atteindre un hôte. Le résultat obtenu donne la liste des routeurs traversés.

```
# traceroute www.nic.fr
traceroute to rigolo.nic.fr (192.134.4.20), 30 hops max, 38 byte packets
 1 toulouse-50-254-gw.dial.proxad.net (212.27.50.254) 24.806 ms 21.489 ms 21.530 ms
 2 paris11-2-p1.routers.proxad.net (212.27.32.225) 43.597 ms 33.325 ms 33.270 ms
 3 paris11-1-p1.routers.proxad.net (212.27.32.226) 149.188 ms 129.723 ms 147.430 ms
 4 sfinx.routers.proxad.net (212.27.32.167) 126.530 ms 138.881 ms 126.858 ms
 5 ri-renater.gix-paris.ft.net (194.68.129.34) 107.966 ms 132.974 ms 135.544 ms
 6 nio-i.cssi.renater.fr (193.51.206.57) 144.283 ms 122.517 ms 127.308 ms
 7 193.51.206.146 (193.51.206.146) 132.595 ms 145.998 ms 148.399 ms
 8 stlambert1.rerif.ft.net (193.48.53.102) 124.040 ms 260.685 ms 108.853 ms
 9 inria-rocquencourt-atm.rerif.ft.net (193.48.53.226) 38.604 ms 167.956 ms 143.657 ms
10 rocq-gw.inria.fr (192.93.122.2) 151.084 ms 96.052 ms 100.700 ms
11 nic-gw.inria.fr (192.93.1.112) 126.699 ms 153.840 ms *
12 rigolo.nic.fr (192.134.4.20) 155.644 ms 150.290 ms 191.674 ms
```

Dans l'exemple ci-dessus, l'hôte recherché a été trouvé. En cas de défaut, cette commande est très utile pour repérer le routeur sur lequel se situe le problème d'interconnexion.

Les tests ICMP effectués avec la commande **ping** ne permettent pas de localiser le point de rupture de la communication entre deux hôtes distants. La commande **traceroute** identifie tous les équipements d'interconnexion réseau traversés.

Le principe de ce tracé de route est le suivant :

- Émettre un premier message avec la valeur 1 dans le champ TTL de l'en-tête IP.
- L'équipement d'interconnexion qui reçoit ce message décrémente la valeur du champ TTL de l'en-tête IP et obtient 0. Il jette donc le message et émet un message ICMP à destination de l'émetteur indiquant qu'il est impossible d'atteindre la destination.

- Émettre un second message avec la valeur 2 dans le champ TTL de l'en-tête IP.
- Cette fois-ci, c'est le second équipement d'interconnexion qui décrémentera la valeur pour obtenir 0. Ce sera donc à ce second équipement d'émettre un message ICMP à destination de l'émetteur.
- Ainsi de suite avec les valeurs 3, 4, etc. Pour le champ TTL de l'en-tête IP.

L'utilisation de la commande **traceroute** est de plus en plus limitée par les divers systèmes de filtrage réseau et pour contrer le travail des outils automatisés dont l'objectif est de relever la topologie d'un réseau à distance.

La méthode la plus immédiate pour bloquer la commande **traceroute** consiste à bloquer en entrée d'un périmètre les ports UDP de la plage 33434 à 33600.

Pour autant, la fonction **traceroute** est très utile pour qualifier la validité d'une communication. Pour essayer de contourner les systèmes de filtrage, la commande **traceroute** offre de nombreuses options telles que la possibilité de fixer les numéros des ports source et destination ou la possibilité de choisir le protocole de couche transport.

Voici un exemple élémentaire permettant de caractériser la différence de fonctionnement entre les deux protocoles de couche transport.

- Utilisation classique du protocole UDP :

```
# traceroute www.neuf.fr
traceroute to www.neuf.fr (212.30.118.74), 30 hops max, 60 byte packets
<snipped>
 3 ge-2-1-0-0.nctou102.Toulouse.francetelecom.net (193.249.214.14) 58.498 ms 59.478 ms 60.458 ms
 4 xe-3-1-3-0.nrpoi202.Poitiers.francetelecom.net (81.253.131.178) 68.446 ms 70.425 ms 73.405 ms
 5 xe-0-1-0-0.ntaub102.Aubervilliers.francetelecom.net (193.251.126.202) 84.394 ms 87.372 ms 89.350 ms
 6 81.253.181.158 (81.253.181.158) 91.335 ms 55.605 ms 55.921 ms
 7 41-197-118-80.kaptech.net (80.118.197.41) 56.919 ms 55.929 ms 57.934 ms
 8 242-193-118-80.kaptech.net (80.118.193.242) 198.938 ms 191.932 ms 190.931 ms
 9 212.94.163.13 (212.94.163.13) 55.931 ms 55.923 ms 58.930 ms
10 10.5.39-62.rev.gaoland.net (62.39.5.10) 57.936 ms 57.922 ms 55.933 ms
11 Vlan4053.9velizy1-0-ro-t-3.9tel.net (213.203.124.181) 56.930 ms 56.934 ms 57.938 ms
12 * * *
13 * * *
14 * * *
15 * * *
16 * * *
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * * *
23 * * *
24 * * *
25 * * *
26 * * *
27 * * *
28 * * *
29 * * *
30 * * *
```

- Utilisation du protocole TCP :

```
# # traceroute -T www.neuf.fr
traceroute to www.neuf.fr (212.30.118.74), 30 hops max, 60 byte packets
<snipped>
 3 ge-2-1-0-0.nctou102.Toulouse.francetelecom.net (193.249.214.14) 55.474 ms 58.456 ms 60.440 ms
 4 xe-3-1-2-0.nrpoi202.Poitiers.francetelecom.net (81.253.130.122) 70.428 ms 72.403 ms 74.391 ms
 5 xe-0-1-0-0.ntaub102.Aubervilliers.francetelecom.net (193.251.126.202) 85.373 ms 87.352 ms 90.335 ms
 6 81.253.181.130 (81.253.181.130) 93.317 ms 55.676 ms 55.927 ms
 7 41-197-118-80.kaptech.net (80.118.197.41) 148.936 ms 139.944 ms 137.946 ms
 8 242-193-118-80.kaptech.net (80.118.193.242) 58.945 ms 164.900 ms 163.940 ms
 9 212.94.163.13 (212.94.163.13) 56.941 ms 55.938 ms 57.951 ms
10 10.5.39-62.rev.gaoland.net (62.39.5.10) 55.949 ms 56.950 ms 55.942 ms
11 Vlan4053.9velizy1-0-ro-t-3.9tel.net (213.203.124.181) 56.949 ms 56.969 ms 56.937 ms
12 62.62.153.54 (62.62.153.54) 57.966 ms 56.963 ms 57.958 ms
13 ilma.finnis.isp.9tel.net (212.30.118.74) 57.924 ms 56.989 ms 56.956 ms
```

Pour obtenir la syntaxe de toutes les options disponibles, il faut accéder aux pages de manuels Unix :

- via la console avec la commande **man traceroute**.
- via l'interface graphique avec le centre d'aide KDE :Pages de manuels Unix → Section 8 Administration système → traceroute.

Il existe quantité d'outils qui permettent d'avancer plus loin dans une utilisation plus subtile des valeurs du champ TTL de l'en-tête IP. Une technique appelée *firewalking* a connu son heure de gloire au début des

années 2000. De nos jours, les pare-feux et les serveurs mandataires (*proxy*) manipulent ces valeurs de façon à masquer le nombre réel de sauts pour atteindre les hôtes d'une infrastructure.

8.3. Travaux pratiques

1. Comment repérer les paramètres de la passerelle par défaut dans la table de routage de votre station ?

Relever l'indicateur de passerelle puis l'adresse IP et le masque réseau.

2. La passerelle par défaut peut-elle appartenir à un autre réseau que celui de la station ?

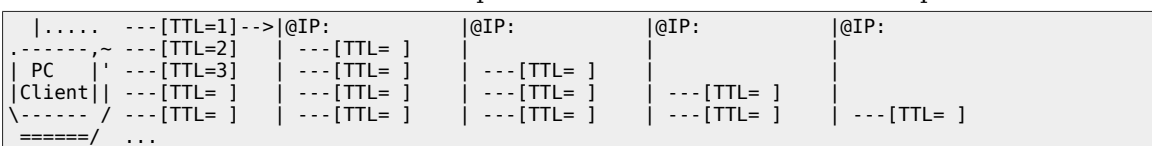
La fonction d'une passerelle par défaut est de fournir une voie de communication vers tous les autres réseaux. Compléter le raisonnement à partir du cas où cette voie de communication n'appartient pas au réseau local.

3. Quel est le rôle de la première entrée de la table de routage ?

Normalement, la détermination du chemin de communication vers les hôtes du réseau local ne doit pas passer par la passerelle par défaut. Compléter le raisonnement sur le mode de communication avec les hôtes du réseau local.

4. Reconstituer les étapes décrites lors de l'exécution de la commande **traceroute** ?

Identifier la station, la passerelle par défaut et les éventuels routeurs en notant les équipements d'interconnexion traversés. Compléter un schéma sous la forme indiquée ci-dessous :



5. Relativement à la question précédente, quelle est la signification des différents champs de chaque ligne affichée par la commande **traceroute** ?

Utiliser les pages de manuels de la commande pour identifier les champs.

6. Dans quelles conditions peut-on obtenir des caractères '*' à la place des champs usuels ?

Toujours à partir des pages de manuels de la commande, identifier les limites de l'utilisation de cette commande.

9. Fonctions réseau d'une interface

Sur tous les systèmes, un certain nombre de paramètres sont actifs par défaut sur les interfaces réseau. Avec le noyau Linux, ces paramètres sont placés dans le système de fichiers virtuel `/proc`.

9.1. Comment visualiser les paramètres du noyau ?

Dans le noyau Linux, la granularité du paramétrage de la pile de protocoles TCP/IP est très fine. Aussi le nombre de paramètres est important. Il suffit de visualiser le résultat des commandes `ls /proc/sys/net/ipv4/` ou `sysctl -A |grep net` pour le constater.

Voici un petit script appelé `show_proc.sh` qui permet de visualiser les paramètres par protocole ou catégorie et leurs valeurs :

```
#!/bin/bash

for param in `find /proc/sys -type f -name "*$1*"`; do
  echo $param = `cat $param`
done
```

Dans le cas des réglages ICMP on obtient le résultat suivant avec un noyau de distribution standard :

```
# ./bin/show_proc.sh icmp
/proc/sys/net/ipv4/netfilter/ip_conntrack_icmp_timeout = 30
/proc/sys/net/ipv4/icmp_ratemask = 6168
/proc/sys/net/ipv4/icmp_ratelimit = 1000
/proc/sys/net/ipv4/icmp_ignore_bogus_error_responses = 0
/proc/sys/net/ipv4/icmp_echo_ignore_broadcasts = 0
/proc/sys/net/ipv4/icmp_echo_ignore_all = 0
```

9.2. Comment changer les valeurs des paramètres ?

Pour changer les valeurs par défaut attribuées dans le noyau, il existe au moins trois solutions :

- La première solution consiste à affecter les valeurs individuellement. Prenons l'exemple célèbre de la fonction de routage des paquets IP du noyau :

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

- La seconde solution utilise le fichier de configuration `/etc/sysctl.conf` de la commande **sysctl** appartenant au paquet `procps`. Ce fichier de configuration n'est pas limité aux fonctions réseau du noyau Linux comme le montre le résultat de la commande **sysctl -A**. Voici un exemple très simple de fichier `/etc/sysctl.conf` :

```
# Activation de protection contre les mauvais messages d'erreurs ICMP
net.ipv4.icmp_ignore_bogus_error_responses=1
```

La commande **sysctl -p** active l'ensemble des valeurs indiquées dans le fichier de configuration. On obtient alors :

```
# ./bin/show_proc.sh icmp_ignore_bogus_error_responses
/proc/sys/net/ipv4/icmp_ignore_bogus_error_responses = 1
```

Comme le fichier `/etc/sysctl.conf` est lu à chaque démarrage du système, les valeurs des paramètres ajustés seront reprises. Ce fichier de configuration est un moyen pratique de conserver les paramètres personnels des fonctions réseau d'une interface.

- La troisième solution est présentée dans la section «*Sécurisations des accès réseau*» du *Manuel de sécurisation de Debian*¹¹.

9.3. Travaux pratiques

Voici un exemple de fichier de configuration `/etc/sysctl.conf` type :

```
# Refuser la prise en charge des requêtes ARP pour d'autres hôtes
net.ipv4.conf.all.proxy_arp = 0

# Ignorer les mauvais messages d'erreurs ICMP
net.ipv4.icmp_ignore_bogus_error_responses = 1

# Ignorer les messages de diffusion ICMP
net.ipv4.icmp_echo_ignore_broadcasts = 1

# Journaliser les adresses sources falsifiées ou non routables
net.ipv4.conf.all.log_martians = 1

# Refuser les adresses sources falsifiées ou non routables
net.ipv4.conf.all.rp_filter = 1

# Refuser les messages ICMP redirect
net.ipv4.conf.all.accept_redirects = 0

net.ipv4.conf.all.send_redirects = 0

# Refuser le routage source
net.ipv4.conf.all.accept_source_route = 0
```

1. Quels sont les tests de communication ICMP à effectuer pour mettre en évidence le résultat du paramètre `net.ipv4.icmp_echo_ignore_broadcasts = 1` ?

Retrouver l'adresse de diffusion du réseau local à utiliser avec la commande **ping**.

2. Quels sont les tests de communication ICMP à effectuer pour mettre en évidence la journalisation des adresses falsifiées ?

La commande **ping** ne permet pas de modifier l'adresse source d'un message ICMP de type 8 (echo request). Il est donc nécessaire d'utiliser un autre outil tel que **hping2**. Voici 2 exemples d'utilisation de cette commande :

- Syntaxe équivalente à celle de la commande **ping** :

```
# hping2 -n -c 4 -1 192.168.1.1
HPING 192.168.1.1 (eth1 192.168.1.1): icmp mode set, 28 headers + 0 data bytes
len=46 ip=192.168.1.1 ttl=64 xml:id=60429 icmp_seq=0 rtt=1.5 ms
len=46 ip=192.168.1.1 ttl=64 xml:id=60430 icmp_seq=1 rtt=1.4 ms
len=46 ip=192.168.1.1 ttl=64 xml:id=60431 icmp_seq=2 rtt=1.4 ms
len=46 ip=192.168.1.1 ttl=64 xml:id=60432 icmp_seq=3 rtt=1.4 ms

--- 192.168.1.1 hping statistic ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.4/1.5/1.5 ms
```

- Syntaxe utilisant l'adresse source falsifiée 192.168.2.2 :

```
# hping2 -n -c 4 -1 -a 192.168.2.2 192.168.1.1
HPING 192.168.1.1 (eth1 192.168.1.1): icmp mode set, 28 headers + 0 data bytes

--- 192.168.1.1 hping statistic ---
4 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Résultat produit dans le journal système de l'hôte 192.168.1.1 :

```
kernel: martian source 192.168.1.1 from 192.168.2.2, on dev eth0
```

¹¹ <http://www.debian.org/doc/manuals/securing-debian-howto/ch4.fr.html#s-network-secure>

```
kernel: ll header: 00:04:75:fd:13:cd:00:0d:bc:ef:a6:8e:08:00
kernel: martian source 192.168.1.1 from 192.168.2.2, on dev eth0
kernel: ll header: 00:04:75:fd:13:cd:00:0d:bc:ef:a6:8e:08:00
kernel: martian source 192.168.1.1 from 192.168.2.2, on dev eth0
kernel: ll header: 00:04:75:fd:13:cd:00:0d:bc:ef:a6:8e:08:00
kernel: martian source 192.168.1.1 from 192.168.2.2, on dev eth0
kernel: ll header: 00:04:75:fd:13:cd:00:0d:bc:ef:a6:8e:08:00
```

9.4. Pour aller plus loin !

Les quelques paramètres des fonctions réseau du noyau Linux présentés ci-avant ne constituent qu'une infime partie. Le document *Ipsysctl tutorial*¹² présente l'ensemble des paramètres utilisables pour ajuster le fonctionnement de la pile de protocoles TCP/IP.

¹² <http://ipsysctl-tutorial.frozentux.net/ipsysctl-tutorial.html>