Chapitre 5

TCP/IP

5.1 Introduction

Le nom **TCP/IP** fait référence à une série complète de protocoles de transmission de données. Son nom provient de deux de ces protocoles: Transmission Control Protocol et Internet Protocol. Bien que cette série contienne d'autres protocoles, ceux-ci sont parmi les plus importants.

Le protocole **IP** est supporté par plusieurs supports physiques dont le plus répandu est ethernet. On peut trouver aussi:

- **IP/Hyperchannel** sur cable coaxial
- **IP/X25.3** sur ligne serie synchrone
- **SLIP** (Serial Line Internet Protocole) pour ligne série asynchrone
- **PPP** (Point to Point Protocol) pour ligne série
- **IP/FDDI** sur fibre optique

5.2 TCP/IP et le modèle OSI

Un modèle d'architecture developpé par l'organisation internationale de normalisation (ISO) est utilisé pour décrire la structure et les fonctions des protocoles de transmission de données.

Ce modèle, appelé modèle OSI (Open Systems Interconnect Reference Model) comporte sept couches définissant les différentes fonctions des protocoles de transmissions de données. La figure 5.1 décrit ces couches. Chaque couche correspond à une fonction exécutée lorsque les données sont transférées entre deux applications. Une couche ne correspond pas exactement à un protocole. Plusieurs protocoles peuvent être partie intégrante de la même couche. Chaque protocole communique avec son homologue. La communication se fait de la manière suivante: Les données descendent la pile en passant d'une couche à l'autre jusqu'à ce qu'elles soient transmises sur le réseau grâce au protocole de la couche matériel. A l'autre extrémité, elle remonte la pile des couches jusqu'à l'application réceptrice. Chaque couche a juste besoin de savoir transférer des données aux couches qui lui sont adjacentes.

Bien que le modèle **OSI** s'avère utile, les protocoles TCP/IP ne correspondent pas exactement à sa structure. Nous décrivons ici les différentes couches et leur utilisation dans TCP/IP.

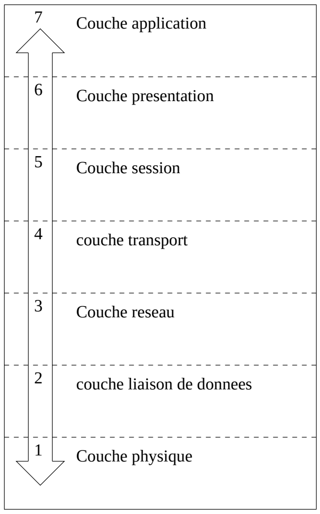

FIG. 5.1 - *Les couches du modèle OSI*

Couche application	:	Opérations du réseau accessible à l'utilisateur.
Couche présentation	:	Pour que les applications puissent échanger des données, elles doivent utiliser la même représentation de données. Cette fonction est intégrée dans les applications.
Couche session	:	Cette couche n'existe pas sous la forme d'une couche séparée dans TCP/IP. La couche session OSI gère les connections entre applications. Dans TCP/IP cette fonction est exécutée au niveau transport.
Couche transport	:	Cette couche (OSI) garantit que le recepteur reçoit les données telles qu'elles ont été transmises. Dans TCP/IP, le protocole TCP exécute cette fonction. Il existe cependant un autre protocole (UDP) qui n'offre pas de test de fiabilité.
Couche réseau	:	Cette couche gère les connections sur le réseau et isole les protocoles des couches supérieures des détails du réseau sous-jacent. Le protocole IP gère l'adressage et transmet les données.
Couche liaison de données	:	Cette couche garantit la fiabilité de la transmission des données sur le réseau physique. TCP/IP crée rarement de protocoles dans cette couche.
Couche physique	:	Cette couche définit les caractéristiques du matériel nécéssaire à l'acheminement du signal de transmission de données. TCP/IP n'intervient pas à ce niveau.

7 **6** **5**	smtp dns telnet ...	rlogin rsh rcp ...
4	TCP	UDP
3	ARP RARP	IP ICMP

TAB. 5.1 - *TCP/IP dans le modèle OSI (approximatif)*

Nous donnons ici une table pour situer l'ensemble des protocoles de TCP/IP dans le modèle OSI.

5.3 adressage

Une adresse IP est composée de quatre octets découpés en deux champs. Le premier champ est le numéro réseau. Le deuxième est l'adresse locale. Ces champs sont de taille variable. La notation utilisée pour déclarer une adresse est la notation décimale pointée. Chaque octet est écrit sous forme décimale séparé par un point. Le nombre de bits de l'adresse du réseau varie en fonction de la classe de l'adresse.

Les trois principales classes sont les classes A,B et C. elles sont déterminées par l'examen des premiers bits du premier octet.

- Si le premier bit est 0 (premier octet$<$128), il s'agit alors d'une adresse réseau de classe A. Les sept bits suivants déterminent l'adresse réseau et les trois octets suivants l'adresse de l'hôte. Il ne peut exister que 128 réseaux de classe A.
- Si les deux premiers bits sont 1 0 (128-191), il s'agit alors d'une adresse de classe B. Les 14 bits suivants référencent le réseau et les deux octets d'après, l'hôte.
- Si les trois premiers bits sont 1 1 0 (192-223) il s'agit alors d'une adresse réseau de classe C. Les 21 bits suivants représentent l'adresse du réseau. Le dernier octet représente l'adresse de l'hôte.
- Si les trois premiers bits sont à 1 ($\geq$224) alors l'adresse est réservée.

5.3.1 Le sous adressage

Chaque site connecté à d'autres réseaux IP doit obtenir une adresse officielle auprès du NIC[1]. Ces adresses IP étant limitées, cela a conduit à la notion de sous réseau. Cette notion permet de relier plusieurs réseaux par l'intermédiaire de routeurs avec une seule adresse réseau. On appelle cette opération le subnetting.

5.3.2 Le masque de sous réseau

Chaque bit du masque de sous réseau (netmask) positionné à 1 indique que la position correspondante de l'adresse Internet doit être interprétée comme partie de l'adresse réseau.

Le masque de sous réseau doit être identique pour tous les sous réseaux dérivés d'un même réseau. Les sous réseaux doivent être physiquement séparés entre eux par un routeur.

1. Network Information Center.

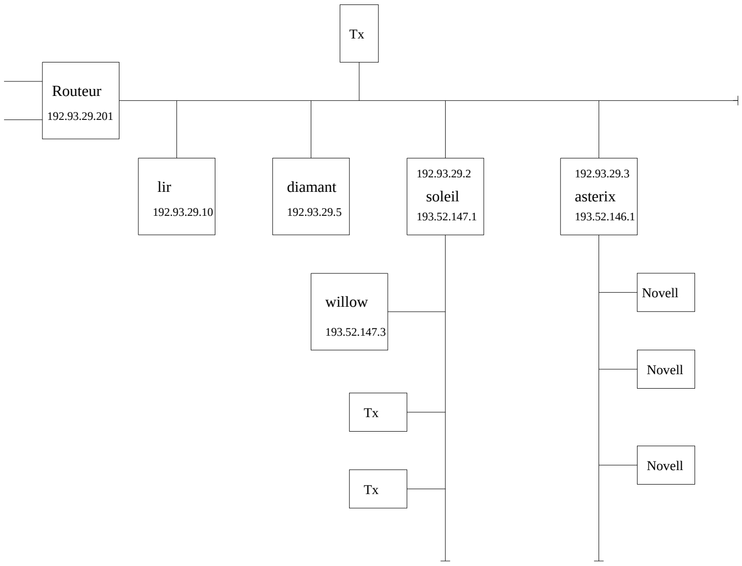

FIG. 5.2 - *Le réseau du D.I.R*

5.3.3 Les adresses particulières

Certaines valeurs des octets ont une signification particulière. On trouve en particulier 0.0.0.0, qui désigne la route par défaut (sous Linux) et 127.0.0.0, l'adresse de bouclage.

Les octets désignant la machine peuvent prendre toutes les valeurs exepté 0 et 255.

L'adresse 192.93.29.255 correspond à l'adresse de diffusion du réseau de classe C 192.93.29.0. L'adresse 127.0.0.1 permet aux processus de communiquer sur la machine locale. Un paquet émis à cette adresse n'est jamais envoyé sur le réseau.

Le masque de valeur 192.93.29.192 découpe le réseau de classe C précédant en quatre sous réseaux de 62 machines. La machine 192.93.29.66 est la machine 2 du réseau 192.93.29.64. L'adresse 192.93.29.127 est l'adresse de diffusion du précédent réseau.

5.4 Les protocoles

Les différents protocoles sont parfaitement décrits par les RFC[2]. Les RFC décrivent aussi bien les protocoles de base comme IP que les protocoles applicatifs comme FTP. L'organisation en couches des protocoles est représentée sur la figure 5.1. Les sockets de type RAW, DGRAM et STREAM sont l'interface de programmation donnant accès aux couches respectivement IP, UDP, TCP.

5.4.1 Les protocoles de bas niveau

Le protocole internet IP

Il est positionné au niveau de la couche réseau. Ce protocole est responsable du routage de l'adressage et de la fragmentation des paquets. C'est un protocole orienté sans connection.

- **adressage et routage**: Il faut déterminer vers quelle machine le paquet doit être expédié, soit pour le mettre sur la bonne route, soit pour le faire arriver à destination.

2. Request For Comment

- **fragmentation**: Les segments des protocoles de niveau supérieur (TCP,UDP) sont découpés en fragments qui sont fonction du support physique. IP doit être capable de réassembler ces fragments pour en refaire des paquets.

Les protocoles ARP et RARP

Ces protocoles permettent de mettre en correspondance une adresse ethernet sur 48 bits et une adresse Internet. Une machine A voulant communiquer avec une machine B du même réseau mais ne connaissant pas son adresse ethernet envoie une demande d'identification en broadcast. La station B répond alors en envoyant son adresse ethernet et son adresse Internet.

Ce protocole n'admet que deux types d'opérations: REQUEST et REPLY. Tous les paquets ont la même structure.

Le protocole ICMP

Le protocole ICMP(Internet Control Message Protocol) est utilisé pour rendre compte des erreurs ou envoyer des messages de contrôle aux logiciels gérant le protocole IP. Ce dernier doit faire remonter l'information vers le programme application. Il permet aussi d'effectuer des statistiques et de tester la communication avec d'autres machines (commande ping).

Les protocoles UDP et TCP

Ce sont les deux protocoles majeurs s'appuyant sur IP. Le premier assure une remise non fiable de datagrammes. Le second assure une remise fiable de datagrammes en mode connecté. Pour les deux protocoles, les données sont livŕees sur un port destination. Le processus detinataire surveille le port en attendant l'arrivée des données. Un port est identifié par son numéro et le protocole associé (telnet 23 TCP).

Le protocole UDP

UDP (User Datagramm Protocol) est un protocole orienté datagramme qui achemine les messages sans garanti ni de bon port, ni d'arrivée dans un ordre identique à celui de l'émission. Il n'y a pas non plus d'accusé de réception. Ce sera donc au récepteur de gerer les problèmes de séquencement, de duplication et d'expiration de délai. Ce protocole permet une communication en mode point à point (talkd) ou en mode diffusion (rwhod).

Le protocole TCP

TCP (Transport Control Protocol) est un protocole orienté connection. La connection est composée de trois phases:

- établissement de la connection,
- transfert des données,
- libération de la connection.

TCP est un protocole capable de gérer la perte, l'endommagement ou la duplication des données. Il établit une liaison full duplex entre émetteur et destinataire et il garantit l'émission convenable des données et l'ordre d'émission. Il s'agit d'un protocole lourd qui doit compenser le manque de fiabilité de la couche inférieure.

5.4.2 Les protocoles applicatifs

On peut citer les différents protocoles suivants :

FTP	:	File Transfert Protocol
TFTP	:	Trivial File Transfert Protocol
TELNET	:	Terminal Network Protocol
SMTP	:	Simple Mail Transfert Protocol
LPD	:	Line Printer Daemon
NFS	:	Network File System
NIS	:	Network Information System

Les applications mettant en œuvre ces protocoles sont organisées en client/serveur. Le serveur est en général un démon. Voici quelques exemples de démons et commandes :

- ftp/ftpd: transfert de fichiers
- telnet/telnetd: login sur une machine distante
- mail/sendmail: échange de courrier entre utilisateurs
- rlogin/rlogind: login sur une machine distante
- rcp/rsh-rcp: transfert d'arborescence de fichiers
- lpr,lpq,lprm/lpd: système d'impression

5.5 Les fichiers de configuration du réseau

Ces fichiers sont utilisés pour paramètrer les différents services du réseau IP. Ils sont stockés dans le répertoire `/etc` et sont présentés dans la table 5.2.

Les fichiers du répertoire utilisateurs décrits dans la table 5.3 sont aussi utilisés.

Les fichiers de `/etc` sont utilisés de manière différente suivant que le service NIS a été mis en place ou non et suivant le rôle de la machine dans NIS.

- Si NIS n'est pas mis en place, les fichiers sont directement utilisés.
- Si la machine est un serveur maître, elle utilise ces fichiers pour fabriquer la base de données.
- Dans les autres cas, ces fichiers ne sont pas consultés. Seule la base de données d'un serveur est consultée.

Les fichiers `hosts.equiv`, `inetd.conf` et `shells` ne font pas partie de la base de données NIS.

5.5.1 Le fichier /etc/ethers

Ce fichier contient des informations nécessaires au protocole RARP. Il contient l'adresse ethernet et le nom d'une machine.

```
00:00:a7:12:03:3b     piano
00:00:a7:11:b2:a9     harpe
00:00:a7:11:b2:5e     clavecin
```

Fichiers de `/etc`	Rôle
ethers	contient les numéros ethernet et les noms
gateways	décrit les passerelles
hosts	contient les numéros IP, les noms et les alias
ftpusers	interdit le ftp
hosts.equiv	donne la liste des machines équivalentes pour les r-commandes
inetd.conf	configuration du super serveur
netgroup	groupe d'utilisateurs ou de machines
netmasks	contient les masques des sous réseaux
networks	contient les numéros de réseaux et leur nom
protocols	numéros des protocoles
services	liste des services IP
shells	liste des shells admissibles

TAB. 5.2 - *Les fichiers de configuration*

Fichiers	Rôle
.rhosts	Utilisateurs équivalents par compte
.netrc	Description permettant d'automatiser les connections ftp

TAB. 5.3 - *Les fichiers de configuration du répertoire personnel*

```
00:00:a7:11:93:0c     saxo
00:00:a7:11:83:e1     flute
00:00:a7:12:fb:01     violon
00:00:a7:11:45:47     hautbois
00:00:a7:00:99:65     guitare
```

Le protocole RARP associe une adresse IP à une adresse ethernet. Il utilise aussi le fichier `/etc/hosts`

5.5.2 le fichier /etc/hosts

Ce fichier contient les noms symboliques des machines connues et le numéro Internet de celles-ci. Le fichier est de la forme suivante:

```
#
# Sun Host Database
#
# If the NIS is running, this file is only consulted when booting
```

```
#
127.0.0.1           localhost
#
192.93.29.10        lir loghost mailhost

192.93.29.2         soleil
192.93.29.201       superstar
192.93.133.35       isis
```

5.5.3 Le fichier /etc/hosts.equiv

Ce fichier contient les noms de machines considérées comme étant équivalentes à la machine locale. Ce fichier permet aux utilisateurs des r-commandes de ne pas s'authentifier sur les machines distantes listées, celles-ci étant équivalentes à la machine locale. Les utilisateurs peuvent alors passer d'une machine à l'autre sans contrainte.

5.5.4 Le fichier .rhosts

Ce fichier se situe dans le répertoire personnel de chaque utilisateur. Il permet à un autre utilisateur provenant d'une machine distante de se connecter sans authentification sur ce compte.

5.5.5 Le fichier /etc/netgroup

Ce fichier permet une représentation de groupe. Les groupes décrivent le plus souvent des machines ou des utilisateurs. Ils sont définis de la manière suivante:

groupe groupe1 groupe2
groupe(nom_machine,nom_utilisateur,nom_domaine)

Dans le deuxième type de syntaxe, un champ vide signifie que toutes les valeurs peuvent être acceptées. Un champ commençant par - signifie qu'aucune valeur n'est acceptée. On peut ajouter ces groupes dans les fichiers `hosts.equiv` et `.rhosts` de la manière suivante:
+@groupe1 décrit un groupe équivalent
-@groupe2 interdit l'accès

On peut grâce à ce mecanisme réduire l'accès à certaines machines pour un groupe d'utilisateurs dans un domaine NIS.

5.5.6 Le fichier /etc/netmasks

Ce fichier sert à définir des masques non standard pour des sous réseaux (subnetting). Il est constitué de deux colonnes qui sont le numéro de réseau et le masque qui y correspond.

5.5.7 Le fichier /etc/networks

Ce fichier décrit les réseaux connus. Il est de la forme suivante:

```
#
```

```
# Sun customer networks
# This file is never consulted when the NIS are running
#
loopback             127
sun-ether            192.9.200          sunether ethernet localnet
sun-oldether         125                sunoldether
#
# Internet networks
#
arpanet              10                 arpa
ucb-ether            46                 ucbether
```

5.5.8 Le fichier /etc/protocols

Ce fichier contient les noms officiels des protocoles et leurs numéros. Il ne doit pas être modifié. Il sert aux programmes accédant aux couches réseaux. Ses lignes sont constituées de trois champs:

- nom officiel du protocole
- numéro du protocole
- autre nom pour le protocole

```
# Internet (IP) protocols
# This file is never consulted when the NIS are running
#
ip      0       IP          # internet protocol, pseudo protocol number
icmp    1       ICMP        # internet control message protocol
igmp    2       IGMP        # internet group multicast protocol
ggp     3       GGP         # gateway-gateway protocol
tcp     6       TCP         # transmission control protocol
pup     12      PUP         # PARC universal packet protocol
udp     17      UDP         # user datagram protocol
```

5.5.9 Le fichier /etc/services

Le fichier contient les noms officiels des services réseaux, leurs numéros et le protocole utilisé. Chaque ligne est composée de trois champs:

- nom officiel
- numéro du port/nom du protocole
- autres noms pour le service

Voici un extrait de ce fichier:

```
# Network services, Internet style
# This file is never consulted when the NIS are running
```

```
#
tcpmux       1/tcp                  # rfc-1078
echo         7/tcp
echo         7/udp
discard      9/tcp    sink null
discard      9/udp    sink null
systat       11/tcp   users
daytime      13/tcp
daytime      13/udp
netstat      15/tcp
chargen      19/tcp   ttytst source
chargen      19/udp   ttytst source
```

5.5.10 Le fichier /etc/shells

Ce fichier donne la liste des shells valides pour les utilisateurs.

5.6 Le super démon inetd

Dans les anciennes versions d'Unix, il existait un démon pour chacun des services réseau. Ce démon écoutait sur un port attendant une demande de connection puis créait un fils pour gérer la communication. Il y avait autant de démons que de services. Le super démon `inetd` simplifie grandement les choses. Ce démon se met à l'écoute sur un ensemble de ports et lance le démon nécessaire lors d'une demande de communication. Le fonctionnement de `inetd` est simple. Il utilise une socket par service. Il attend l'arrivée d'une connection grâce à la primitive `select()`. Il crée un fils qu'il recouvre avec le code du service demandé (`fork()`, `exec()`) et il se remet en attente sur le port. Pour le mode datagramme le superserveur peut ou non attendre la fin du service. `inetd` prend en compte les services déclarés dans le fichier `/etc/inetd.conf`. Pour qu'une actualisation du fichier `/etc/inetd.conf` soit prise en compte, il faut envoyer le signal SIGHUP au serveur.

5.6.1 Le fichier /etc/inetd.conf

Chaque ligne de ce fichier contient les informations nécessaires à l'utilisation d'un processus serveur. Chaque ligne est composée de sept champs séparés par des blancs. Ces champs sont les suivants :

- Nom du service: tiré du fichier `/etc/services`
- type de socket: stream, dgram, ...
- protocole: tiré du fichier `/etc/protocols`
- wait/nowait: pour savoir si le serveur doit avoir terminé la requête avant d'en prendre une nouvelle (datagramme uniquement)
- utilisateur: le nom de l'utilisateur qui sera attaché au serveur
- le programme serveur: chemin absolu vers le serveur que `inetd` doit démarrer

– les arguments: arguments passés au serveur (avec le nom du serveur lui même).

Voici un extrait du fichier `/etc/inetd.conf`:

```
# Internet services syntax:
# <service_name> <socket_type> <proto>
                <flags> <user> <server_pathname> <args>
#
# Ftp and telnet are standard Internet services.
#
ftp    stream   tcp    nowait  root  /usr/etc/tcpd       in.ftpd
telnet stream   tcp    nowait  root  /usr/etc/tcpd       in.telnetd
#
# Tnamed serves the obolete IEN-116 name server protocol.
#
name   dgram    udp    wait    root  /usr/etc/in.tnamed  in.tnamed
#
# Shell, login, exec, comsat and talk are BSD protocols.
#
shell  stream   tcp    nowait  root  /usr/etc/tcpd       in.rshd
#login stream   tcp    nowait  root  /usr/etc/in.rlogind in.rlogind
exec   stream   tcp    nowait  root  /usr/etc/tcpd       in.rexecd
comsat dgram    udp    wait    root  /usr/etc/tcpd       in.comsat
talk   dgram    udp    wait    root  /usr/etc/tcpd       in.talkd
```

5.7 Le routage

5.7.1 Introduction

Le routage ou acheminement est la partie permettant au trafic internet de joindre sa destination où qu'elle soit dans le monde (après avoir probablement traversé de nombreux réseaux intermédiaires).

5.7.2 Les différentes configurations de routage

Il faut faire la distinction entre routage et protocole de routage. Les protocoles de routage sont des programmes échangeant des données afin de construire les tables de routage. Les tables de routage peuvent être créées à la main, lorsqu'il n'existe qu'une seule route par exemple. Nous allons distinguer trois cas de configuration de routage:

Routage minimal

Un réseau complétement isolé des autres réseaux requiert ce type de configuration. La commande `ifconfig` configure l'interface et crée la table de routage minimal.

Routage statique

Lorsque le nombre de passerelles vers l'extérieur est limité, l'administrateur système crée une table de routage statique grâce à la commande `route`. Le routage statique ne s'adapte pas à la modification du réseau. Si l'ensemble du trafic ne transite que par une route, un routage statique est alors adapté (D.I.R).

Routage dynamique

Un réseau pouvant utiliser plusieurs routes vers la même destination doit utiliser le routage dynamique. La table de routage dynamique est créée à partir des informations échangées par les protocoles de routage.

Les protocoles doivent non seulement tenir compte des aléas du réseau (route «HS») mais doivent de plus proposer la meilleure route vers une destination.

5.7.3 Table de routage minimale

La commande ifconfig nous a permis de créer une table de routage minimal. Analysons celle ci:

```
# netstat -rn
Routing tables
Destination  Gateway       Flags      Refcnt      Use       Interface
127.0.0.1    127.0.0.1     UH         2           18049     lo0
193.52.146.0 193.52.146.2  U          66          36838821  le0
```

La première entrée correspond à l'adresse de bouclage de la machine, la seconde correspond à l'adresse vers le réseau 192.93.29.0. Le numéro IP de la passerelle correspond à l'adresse de l'interface le0 (interface utilisée pour atteindre ce réseau). Les options du drapeau sont :

- U: Up, l'interface est active.
- H: Host, cette adresse ne permet d'atteindre qu'un hôte

5.7.4 Routage statique et routes indirectes

Considérons l'exemple suivant:

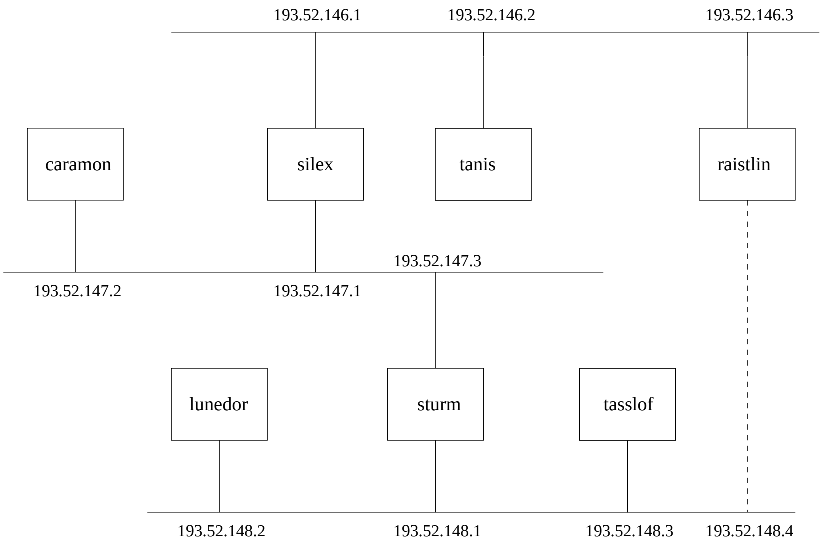

FIG. 5.3 - *Exemple de réseau*

Dans ce cas, il s'agit pour la machine tanis d'atteindre la machine caramon en utilisant la machine silex comme passerelle. Il faut dans ce cas ajouter une nouvelle route après configuration minimal:

`route add caramon silex 1` ou
`route add 193.52.147.2 193.52.146.1 1`

```
# netstat -rn
Routing tables
Destination  Gateway      Flags     Refcnt     Use       Interface
127.0.0.1    127.0.0.1    UH        2          18049     lo0
193.52.146.0 193.52.146.2 U         66         36838821  le0
193.52.147.2 193.52.146.1 UGH       66         36838821  le0
```

On peut remarquer que le drapeau G est positionné, ce qui signifie que la machine silex sert de passerelle.

Le routage est un routage machine. On peut étendre ce routage au routage réseau. On va pouvoir atteindre l'ensemble des machines du réseau 193.52.147.0 depuis tanis par la commande :

`route add net 193.52.147.0 193.52.146.1`

Cette nouvelle commande permet de diminuer le nombre de routes à ajouter. Chaque solution a ses avantages. La première permet de restreindre l'accès à telle ou telle machine. Le rôle d'un routeur est donc la retransmission (forwarding) des paquets.

Examinons maintenant le cas où il y a deux routeurs à traverser. On déclare les routage suivants:
`route add lunedor silex 2` # fait sur tanis
`route add tanis sturm 2` # fait par lunedor

De plus les routeurs devront déclarer les routes suivantes :
`route add net 193.52.148.0 193.52.147.3 1` # fait sur silex
`route add net 193.52.146.0 193.52.147.1 1` # fait sur sturm

On voit ici apparaitre le principal inconvénient du routage statique. Il demande une maintenance rigoureuse des tables de routage.

5.7.5 Le routage par redirection ICMP

Dans cette technique de routage, c'est le routeur qui centralise la décision de routage. Un tel routeur est alors déclaré par les autres routeurs comme la route par défaut. Posons comme routeur par défaut la machine silex. La machine tanis déclare alors:
`route add default silex`

Tous les messages partant vers un réseau extérieur sont alors envoyés sur silex. Si nous considérons maintenant raistlin comme connecté au réseau 193.52.148.0, la meilleure route depuis tanis pour atteindre lunedor n'est plus par silex. Le routeur émet maintenant un paquet appelé «ICMP redirect» qui signifie: pour atteindre lunedor, passer par le routeur raistlin, et qui se traduit de la manière suivante:

```
# netstat -rn
Routing tables
Destination  Gateway      Flags     Refcnt     Use       Interface
127.0.0.1    127.0.0.1    UH        2          18049     lo0
193.52.146.0 193.52.146.2 U         66         36838821  le0
default      193.52.146.1 UG        66         36838821  le0
193.52.148.2 193.52.146.3 UGHD      66         36838821  le0
```

Le drapeau D indique que cette route a été créée par un message ICMP redirect. Les deux principaux handicaps de cette méthode sont

- Chaque machine doit déclarer un routeur par défaut. Si celui-ci change, il faut changer l'ensemble des tables de routage.
- Si le routeur par défaut est en panne, le routage ne fonctionne plus même s'il existe une autre route pour atteindre la destination.

5.7.6 Le routage dynamique

Ce type de routage met en oeuvre des protocoles de routage placés au dessus d'UDP. Il s'agit sous Unix de :

- RIP Routing Information Protocol
- EGP Exterior Gateway Protocol
- HELLO Hello routing protocol

Ces protocoles sont mis en oeuvre par les démons Unix. Nous nous intéresserons principalement au protocole RIP.

Le démon routed

C'est ce démon qui met en oeuvre le protocole RIP sous Unix. `routed` vérifie d'abord quelles sont les interfaces «UP» puis dans un second temps envoie un paquet en broadcast et se met à l'écoute des requêtes et réponses des autres machines. Le démon **routed** utilise le fichier `/etc/gateways` pour connaitre l'ensemble des routes statiques ainsi que les routes définies par la commande `route`. Ces dernières ne sont cependant pas diffusées. Le fichier `/etc/gateways` est composé de la manire suivante:
<net|host> adr1 <gateway> adr2 <metric> nombre <passive|active>

- net|host: indique un routage réseau ou machine
- adr1: adresse à atteindre (réseau ou machine)
- adr2: adresse de la passerelle
- nombre: nombre de sauts
- passive|active: le routeur n'échange pas d'information ou en échange

Les routeurs déclarés passifs sont maintenus dans la table de routage alors que ceux déclarés actifs sont supprimés s'ils n'envoient plus d'information pendant un certain temps.

5.8 Les commandes de configuration

5.8.1 La commande ifconfig

Cette commande vérifie ou spécifie les valeurs de configuration. Elle est indépendante de la version Unix.

Spécification

La commande **ifconfig** est utilisée pour définir l'adresse IP, le masque de sous réseau et l'adresse de diffusion. Illustrons ceci par un exemple
`ifconfig le0 193.52.147.1 netmask 255.255.255.0 broadcast 193.52.147.255`

Décrivons ici l'ensemble des paramètres:

interface	:	Il s'agit du nom de l'interface réseau à configurer. Ici le0.
adresse	:	On donne ici une adresse IP à l'interface. Elle est entrée sous forme décimale pointée.
netmask	:	masque de sous réseau attribué au réseau sur lequel se trouve cette interface. Cet argument peut être ignoré si le réseau n'est pas divisé.
broadcast	:	Adresse de diffusion associée au réseau. Chaque système d'un sous réseau doit utiliser la même adresse de diffusion.

Vérification

La vérification d'une interface se fait en tapant la commande ifconfig sans autre argument que le nom de l'interface.

```
# ifconfig le0
le0: flags=63<UP,BROADCAST,NOTRAILERS,RUNNING>
          inet 193.52.147.1 netmask ffffff00 broadcast 193.52.147.255
```

La sortie se fait en deux lignes. La première donne l'état de l'interface

UP	:	L'interface est activée et elle peut être utilisée.
BROADCAST	:	L'interface supporte la diffusion (ethernet)
NOTRAILERS	:	L'interface ne supporte pas l'encapsulation de fin de trame (ethernet)
RUNNING	:	L'interface est opérationnelle

La deuxième ligne contient les informations TCP/IP comme nous les avons vues précédemment.

5.8.2 La commande netstat

Il est aussi possible de déterminer l'ensemble des interfaces d'un système grâce à la commande:
`# netstat -ain`

Name	Mtu	Net/Dest	Address	Ipkts	Ierrs	Opkts	Oerrs	Collis	Queue
le0	1500	193.52.147.0	193.52.147.1	130672	0	119124	0	294	0
le1	1500	192.93.29.0	192.93.29.2	20577	0	20801	0	162	0
lo0	1536	127.0.0.0	127.0.0.1	18752	0	18752	0	0	0

Les différents champs sont les suivants:

Name	:	Le nom de l'interface réseau défini par ifconfig.
Mtu	:	Le champ unité de transmission indique la trame la plus longue que cette interface peut transmettre.
Net|Dest	:	Ce champ indique l'hôte ou la destination que cette interface permet d'atteindre.
Address	:	Ce champ contient l'adresse attribuée à cette interface.
Ipkts	:	Ce champ indique le nombre de paquets que l'interface a reçus.
Ierrs	:	Ce champ indique le nombre de paquets endommagés reçus.
Opkts	:	Ce champ contient le nombre de paquets envoyés par l'interface.
Oerrs	:	Le champ erreurs de sortie indique le nombre de paquets générant une erreur.
Collis	:	Le champ collision indique le nombre de collisions détectées par cette interface (ethernet).
Queue	:	Ce champ indique le nombre de paquets dans la file d'attente attendant leur transmission via cette interface.

Cette commande permet donc en particulier de faire un suivi des différentes interfaces réseau.