U.E. ARES
Architecture des Réseaux

Cours 4/6 : Couche réseau

Olivier Fourmaux
(olivier.fourmaux@upmc.fr)

Version 5.4

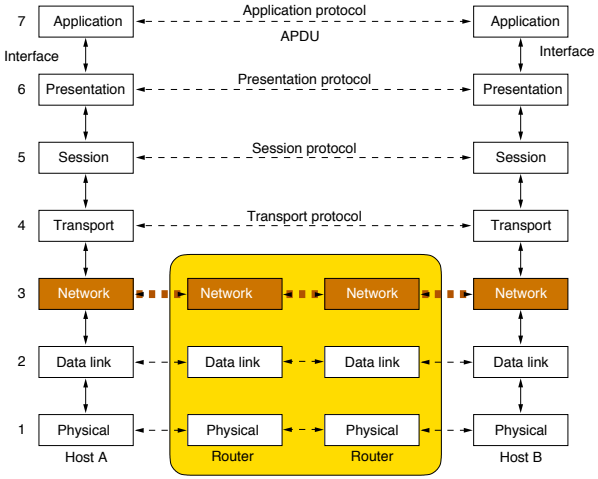
Plan

Couche Réseau

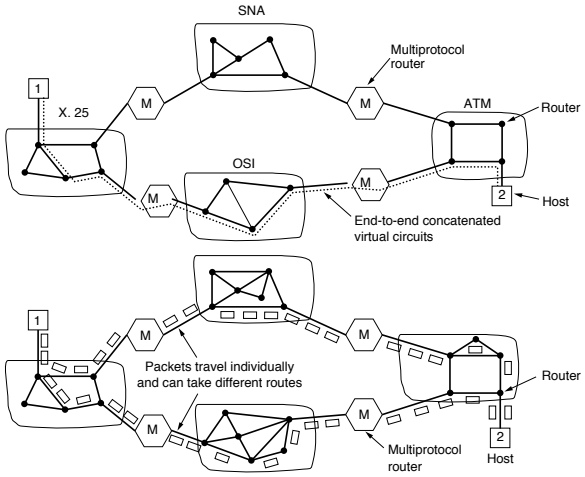
La **Couche Réseau** achemine les paquets de la source vers les destinataires en effectuant des sauts entre les différents **nœuds intermédaires**

- de bout-en-bout (*end-to-end*)
- connaissance de la topologie
- calcul du chemin (**routage**)
- adressage virtuel
- abstraction des technologies sous-jacentes
  - ✓ encapsulation sur chaque technologie
  - ✓ fragmentation
  - ✓ conversion d'adresses

Couche Réseaux : OSI



Couche Réseau : approche circuit virtuel ou datagramme

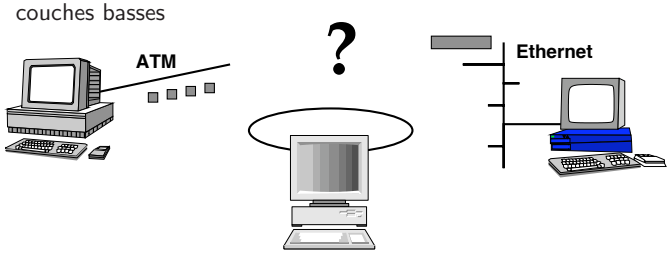


pictures from TANENBAUM A. S. *Computer Networks 3rd edition*

Couche Réseau : Encapsulation

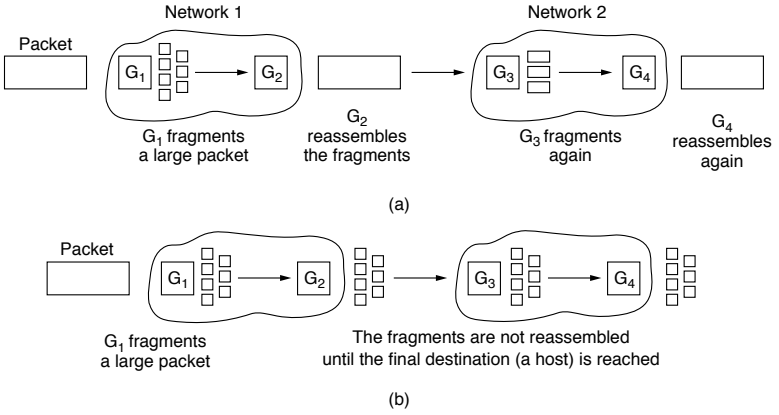
La couche réseau fait abstraction des technologies sous-jacentes

- les données doivent pouvoir circuler de réseaux en réseaux
- les couches supérieures ne doivent faire aucune hypothèse sur les couches basses



➠ sera approfondie dans les cours sur les **Architectures supports**

Couche Réseau : Fragmentation

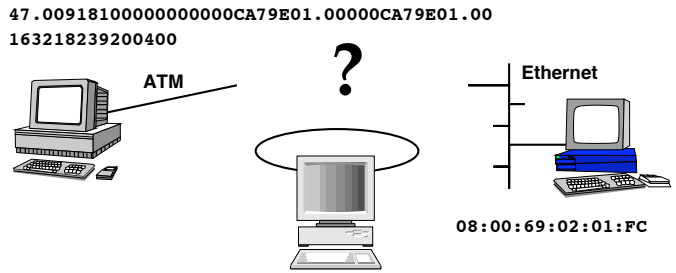


pictures from TANENBAUM A. S. *Computer Networks 3rd edition*

Couche Réseau : Adressage

La couche réseau définit un **adressage virtuel** valide sur tous les réseaux

- identification unique d'un équipement
- masquage des mécanismes d'adressages spécifiques à une technologie
  - ✓ nécessite la mise en correspondance des adresses



➠ sera aussi approfondi dans les cours sur les **Architectures supports**

Couche Réseau : Routage

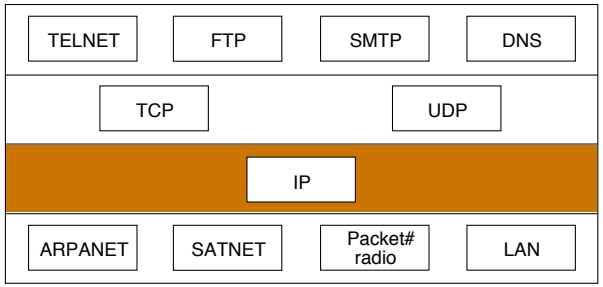
Calcul du chemin
- initial (circuits virtuels)
- à chaque paquet (sans mémoire)

Décisions de routage basée :
- table de routage
  - ✓ statique
  - ✓ dynamique
    - ☞ algorithmes de routage
    - ☞ protocoles de routage...

➠ sera approfondi dans le cours sur le **Routage**

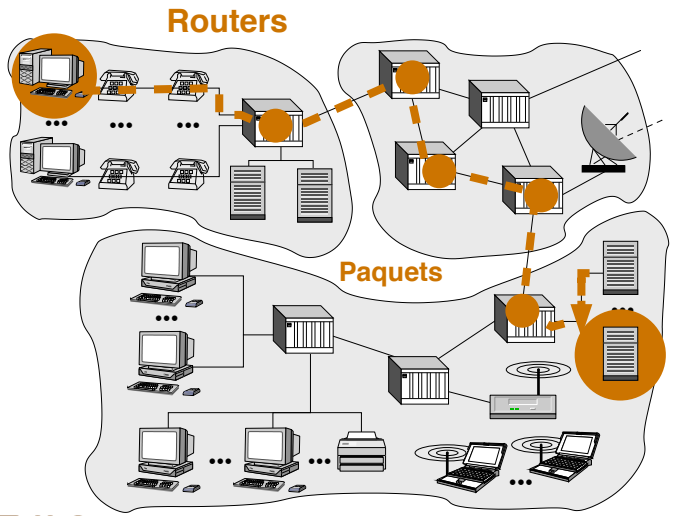
Plan

Couche Réseaux : TCP/IP



➠ IP est l'interface universelle

IPv4



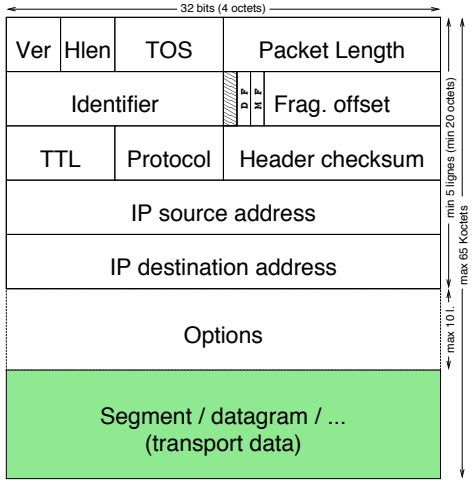
IPv4 : Service

Paquets en mode non connecté

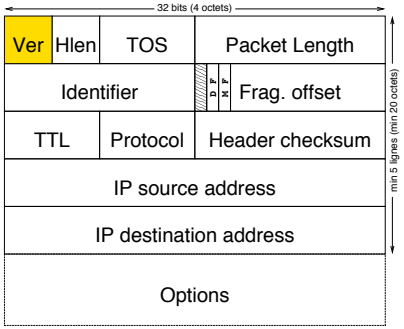
Service à remise non garantie *(best effort)*

Plan

IPv4 : Structure

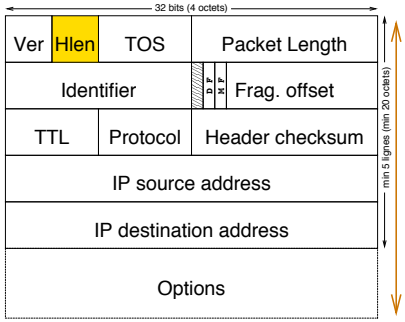


IPv4 : Version



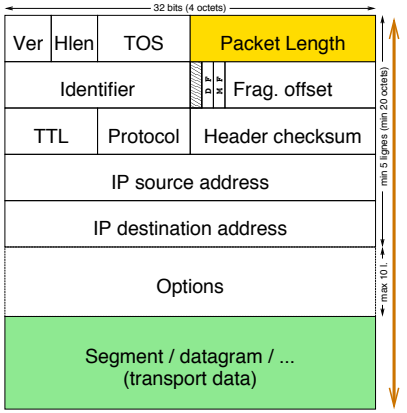
- 4 bits
- IP actuel : version 4
- IP *next génération* : version 6
  ➠ voir l'U.E. **ING**

IPv4 : Longueur de l'entête



- 4 bits (valeur 15 max)
  - ✓ indique le nombre de lignes de 32 bits dans l'entête IP
    - ☞ nécessaire car le champ option est de longueur variable (20 à 60 octets)
    - ☞ valeur de 5 (pas d'options) à 15 (10 lignes d'options, soit 40 octets)

IPv4 : Taille du paquet

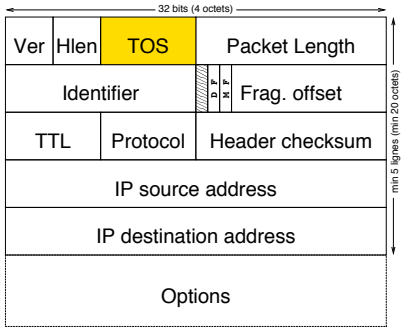


- 16 bits (64 Koctets maximum)
  - ✓ taille totale du paquet **avec entête**
  - ✓ exprimé en octets
    - ☞ le réseau support doit accepter un MTU[1] $\geq$ **576 octets**[2]

[1] MTU : Maximum Transmission Unit

[2] 576 octets = 512 de données applicative + 64 de surcoût protocolaires (entêtes IP et transport)

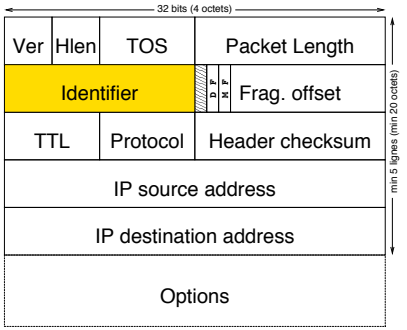
IPv4 : TOS



Type Of Service

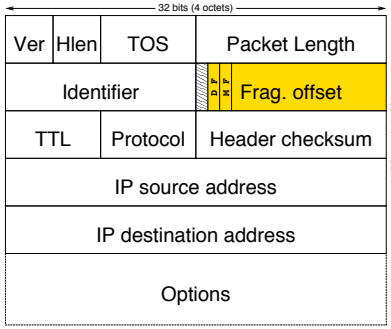
- 8 bits :
  - ✓ 3 bits de **priorité** (*precedence*)
    - ☞ 000 : *Routine*
    - ☞ 001 : *Priority*
    - ☞ 010 : *Immediate*
    - ☞ 011 : *Flash*
    - ☞ 100 : *Flash override*
    - ☞ 110 : *Internetwork control*
    - ☞ 111 : *Network control*
  - ✓ 3 bits de **service**
    - ☞ *Delay*
    - ☞ *Throughput*
    - ☞ *Reliability*
    - ☞ *(Cost)*
- non utilisé...
  ... pour le moment
  ➠ voir l'U.E. **ING**
  (*DiffServ Byte*)

IPv4 : Identificateur



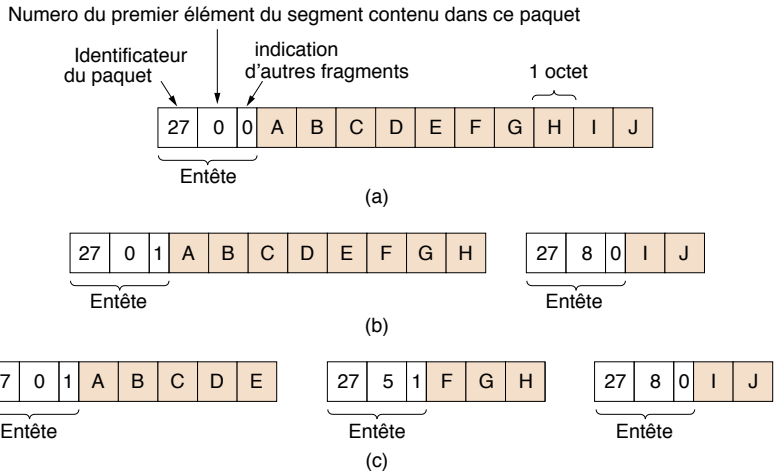
- 16 bits (boucle tous les 64 Kpaquets)
- défini de manière **unique** pour chaque paquet
- pour réassembler les fragments d'un **même** paquet
- habituellement, **incrément** d'un compteur pour chaque paquet successif

IPv4 : Fragmentation

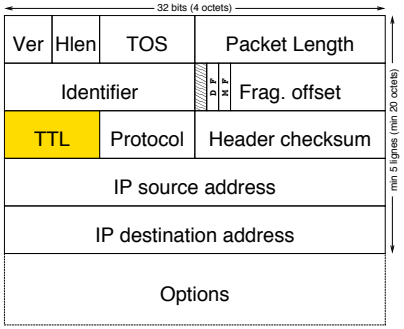


- 1 bit réservé
- 1 bit `DF` :*Don't fragment* (=1 interdit la fragmentation)
- 1 bit `MF` :*More fragment* (=0 pour le dernier fragment)
- 13 bits *fragment offset* en octets/8 (`shift 3`)
  - ✓ exemples :
    - ☞ 0x0000 paquet entier (*offset*=0)
    - ☞ 0x2000 premier fragment (*offset*=0)
    - ☞ 0x20A0 fragment central (*offset*=1280)
    - ☞ 0x00B0 dernier fragment (*offset*=1408)

IPv4 : Fragmentation



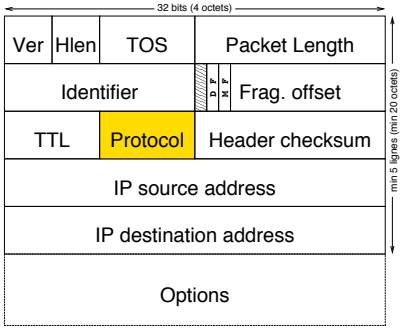
IPv4 : Temps de vie



Time To Live

- 8 bits
  - ✓ unité initiale : **seconde**
  - ✓ valeur maximum fixé par l'émetteur (255, 128, 64...)
  - ✓ décrément dans chaque routeur
    - ☞ minimum 1 par routeur
      - ➠ nombre de **sauts**
  - ✓ max 255 secondes ou sauts
    - ☞ **évite les boucles**

IPv4 : Protocole transporté

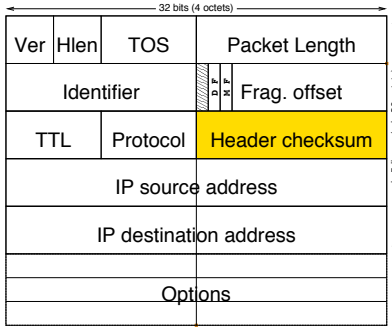


- démultiplexage vers les protocoles de la couche supérieure :

```
Unix> cat /etc/protocols
icmp    1   # internet control message protocol
ggp     3   # gateway-gateway protocol
ipencap 4   # IP encapsulated in IP
st      5   # ST datagram mode
tcp     6   # transmission control protocol
egp     8   # exterior gateway protocol
udp     17  # user datagram protocol
rdp     27  # "reliable datagram" protocol
iso-tp4 29  # ISO Transport Protocol class 4
xtp     36  # Xpress Tranfer Protocol
idrp    45  # Inter-Domain Routing Protocol
rsvp    46  # Reservation Protocol
gre     47  # General Routing Encapsulation
ospf    89  # Open Shortest Path First IGP
...
```

- 8 bits

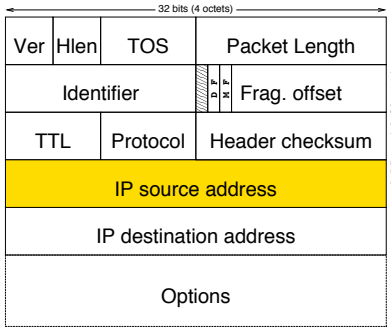
IPv4 : Contrôle d'erreur sur l'entête



- 16 bits
- contrôle d'erreur
  - ✓ idem UDP/TCP mais que sur l'entête
- émetteur :
  - ✓ entête IP $=$ suite $mot_{16\text{bits}}$
  - ✓ *checksum*[3] $= \overline{\sum mot_{16\text{bits}}}$
- récepteur :
  - ✓ recalcul de $\sum mot_{16\text{bits}}$
    - ☞ $= 0$: pas d'erreur détectée toujours possible...
    - ☞ $\neq 0$: erreur (destruction silencieuse)

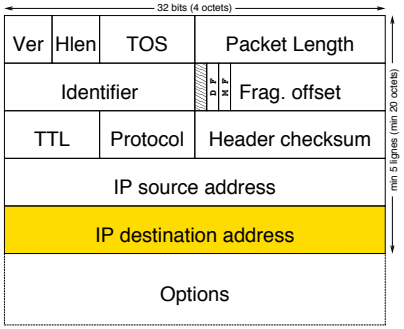
[3] Somme binaire sur 16 bits avec report de la retenue débordante ajoutée au bit de poid faible

IPv4 : Adresse source



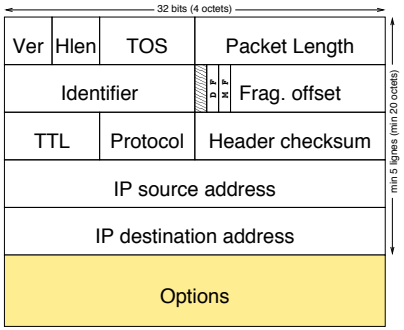
- adresse IP 32 bits
- identifie l'émetteur du paquet
- permet de retourner un message à l'émetteur (ICMP, UDP...)

IPv4 : Adresse destination



- adresse IP 32 bits
- utilisée pour le routage
  - ✓ indique le réseau (ou l'agrégation de réseau) du destinataire
  - ✓ identifie l'**interface** du destinataire dans son réseau

IPv4 : Options

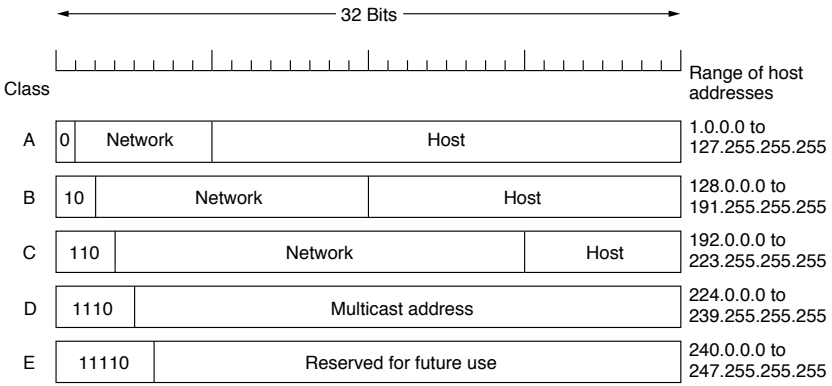


- système TLV identique à TCP
- analysées dans **chaque routeur**
- exemple :
  - ✓ enregistrement de la route
  - ✓ routage à la source strict
  - ✓ routage à la source relâché
  - ✓ estampilles temporelles
  - ✓ sécurité
  - ✓ ...
- 0 à 40 octets (alignés sur 32 bits)

➠ A éviter !

Plan

Adressage : Classes



pictures from Tanenbaum A. S. *Computer Networks 3rd edition*

Adressage : Masques

Application de masques binaires

classe	masque binaire	netmask	prefixe
A	11111111000000000000000000000000	255.0.0.0	/8
B	11111111111111110000000000000000	255.255.0.0	/16
C	11111111111111111111111100000000	255.255.255.0	/24

Extraction du `netId`

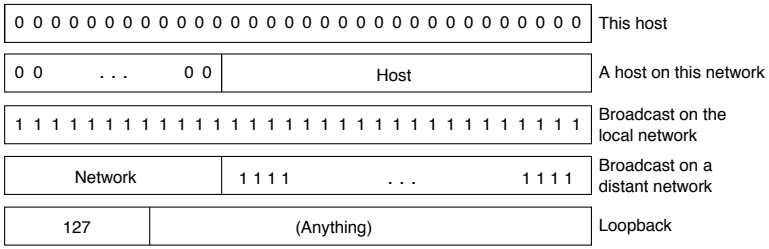
```
     132.227. 60.135        netId.hostId
&&   255.255.  0.  0   &&   netmask
     132.227.  0.  0        netId.  0. 0
```

Extraction du `hostId`

```
     132.227. 60.135        netId.hostId
&&     0.  0.255.255  &&   !netmask
              60.135              hostId
```


Adressage : Adresses particulières

Adresses particulières :



pictures from Tanenbaum A. S. *Computer Networks 3rd edition*

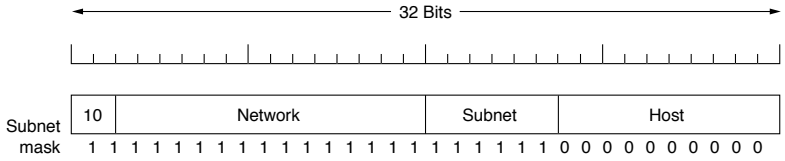
Adressage : Subneting (1)

Taille de l'identifiant de réseau (`netId`) par défaut :

- classe A : /8 (`255.0.0.0`)
- classe B : /16 (`255.255.0.0`)
- classe C : /24 (`255.255.255.0`)

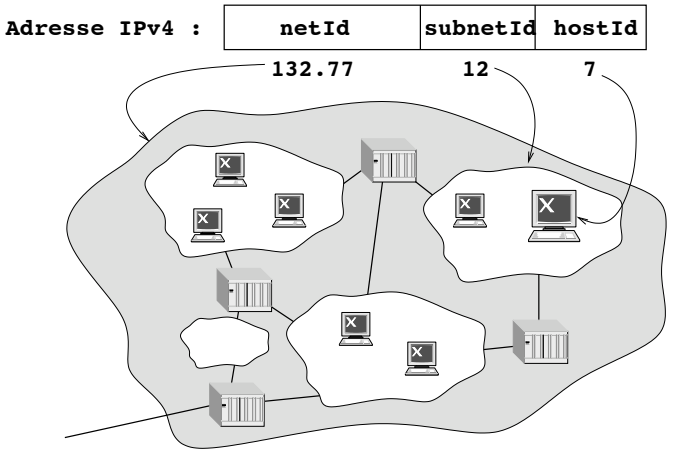
Subdivision possible :

- `132.77.12.0/22` (notation par **préfixe**)
- `132.77.12.0 netmask 255.255.252.0` (notation par **masque**)

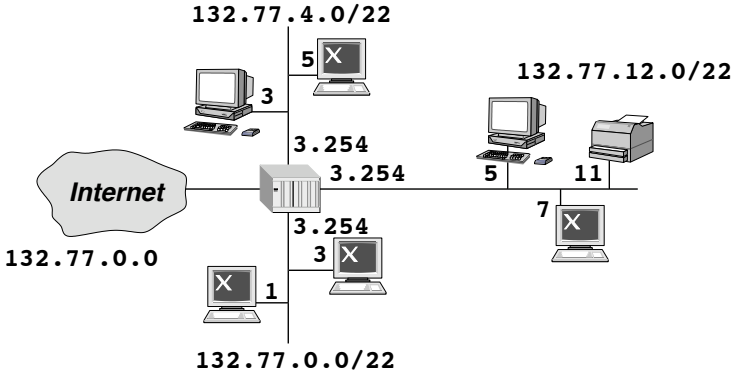


pictures from TANENBAUM A. S. *Computer Networks 3rd edition*

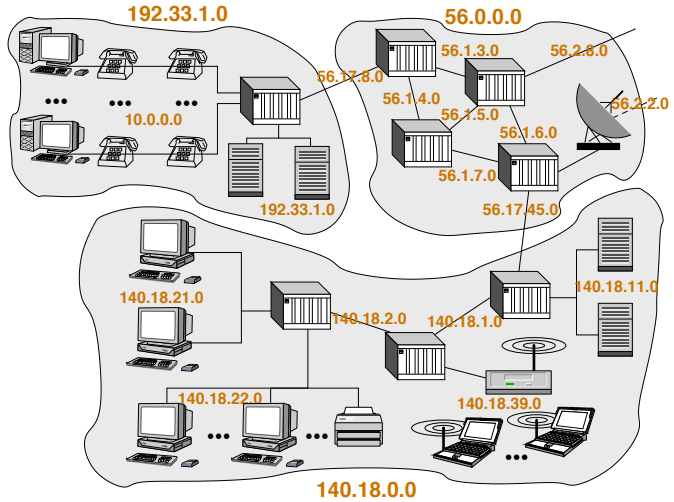
Adressage : Subneting (2)



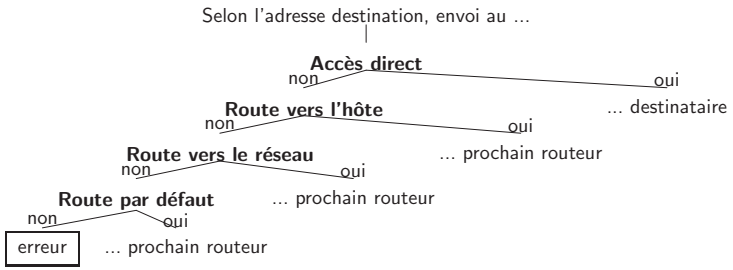
Adressage : Subneting (3)



Adressage : affectation

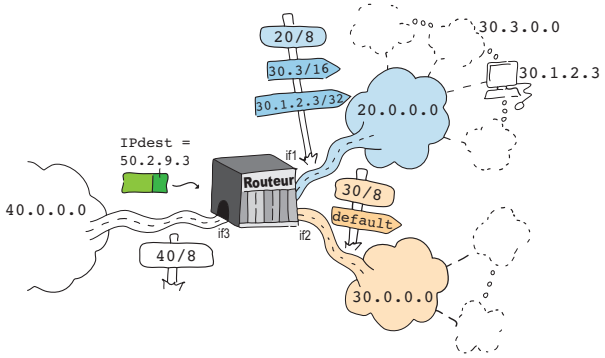


IPv4 : Logique de routage




```
Unix> route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.33.182.0    0.0.0.0         255.255.255.0   U     0      0        0 eth0
10.0.0.0        0.0.0.0         255.0.0.0       U     0      0        0 atm0
154.18.2.0      0.0.0.0         255.255.255.0   U     0      0        0 eth1
132.77.0.0      154.18.2.254    255.255.0.0     UG    0      0        0 eth1
default         192.33.182.254  0.0.0.0         UG    0      0        0 eth0
```


Routage : *Longest Préfix Match*



Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
20.0.0.0	0.0.0.0	255.0.0.0	U	0	0	0	if1
30.0.0.0	0.0.0.0	255.0.0.0	U	0	0	0	if2
40.0.0.0	0.0.0.0	255.0.0.0	U	0	0	0	if3
30.3.0.0	20.1.2.3	255.255.0.0	UG	0	0	0	if1
30.1.2.3	20.1.0.1	255.255.255.255	UGH	0	0	0	if1
60.126.6.0	30.0.0.1	255.255.255.0	UG	0	0	0	if2
default	30.0.0.1	0.0.0.0	UG	0	0	0	if2

Plan

Adressage sans classe

L'attribution des adresses IP avec classe est **inefficace**

- adresses allouées par blocs de 256, 65K ou 16M
  - ✓ les sous-réseaux permettent une meilleure gestion
- un adressage **sans classe** augmente la souplesse dans l'attribution des adresses :
  - ✓ les adresses :
    - ☞ 192.77.16.0/24
    - ☞ 192.77.17.0/24
    - ☞ 192.77.18.0/24
    - ☞ 192.77.19.0/24
  - ✓ peuvent être regroupées en :
    - ☞ notation par **préfixe** : `192.77.16.0/22`
    - ☞ notation par **masque** : `192.77.16.0 netmask 255.255.252.0`

Adressage : Superneting

CIDR (*Classless InterDomain Routing*)

- utilisé pour agréger des blocs d'adresses contigües
- permet aux routeurs de maintenir une seule entrée de table de routage
- utilisé initialement par les ISP pour grouper des adresses de classe C
- initialement décrit en réduisant la taille du préfixe réseau
  - ✓ le préfixe réseau par défaut pour la classe C est /24
  - ✓ les valeurs de préfixes réseau /23, /22, /21, etc. décrivent des agrégations d'adresses de classe C
    - ☞ exemples :

```
197.88.0.0/16    agrège 256 adresses de classe C

81.152.12.0/22   agrège ??
```

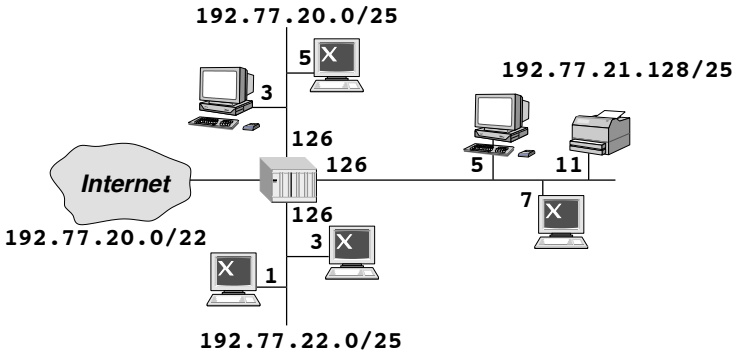

Adressage : Calcul CIDR

Un bloc CIDR est donc l'agrégation d'un ensemble d'adresses

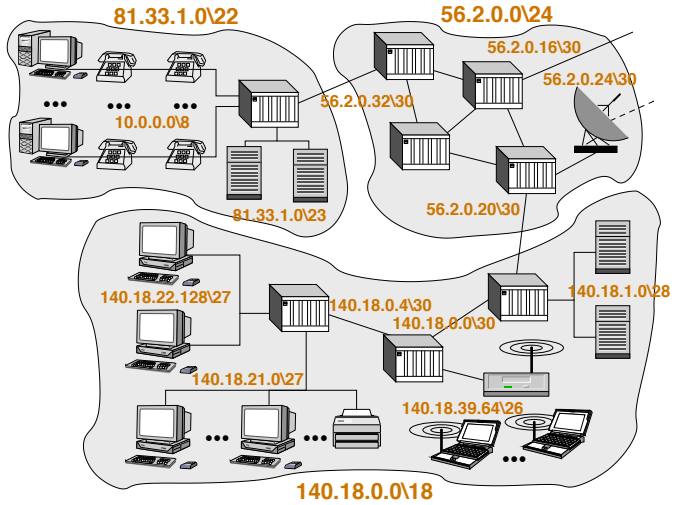
- **bits réseau** (`netId`) d'un bloc CIDR correspondent aux N bits les plus à gauche ($/N$ définit le masque réseau du bloc CIDR)
- **bits hôte** (`hostId`) du bloc CIDR correspondent aux $32 - N$ bits restants
- ensemble des adresses attribuables dans un bloc CIDR :
  - ✓ premier hôte : `hostId` = 000...0001
  - ✓ dernier hôte : `hostId` = 111...1110
  - ✓ adresse de diffusion : `hostId` = 111...1111
    - ☞ exemple :

```
Bloc CIDR -> 192.77.20.0/22
@ premier hôte : 192.77.20.1
...
@ dernier hôte : 192.77.23.254
@ de diffusion : 192.77.23.255
```


Adressage : Subneting des agrégats



Adressage : Affectation



Adressage : Synthèse

Observation sur le découpage des plages d'adresses en sous-réseaux (*subnetting*) ou en agrégats (*supernetting*) :

- **Attention** aux analyses simplistes...
  - ✓ N = nombre de bits réseau
  - ✓ H = nombre de bits hôte
  - ✓ D = préfixe réseau par défaut
    (8 pour la classe A, 16 pour la classe B, 24 pour la classe C)
    - ☞ si $N = D$, pas de *subnetting* ni de *supernetting*
    - ☞ si $N > D$, *subnetting* (sous-réseau)
    - ☞ si $N < D$, *supernetting* (CIDR)

  ... **FAUX**, car on peut combiner *subnetting* et *supernetting* !

Plan

IPv4 : Adresses privées

Deux types d'adressage :

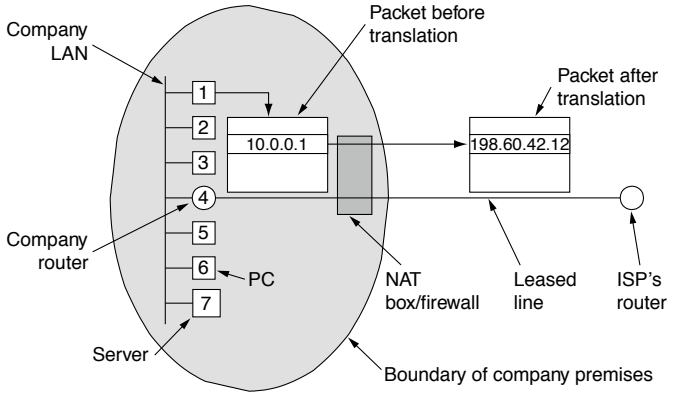
Public : tout hôte connecté à l'Internet doit avoir une adresse unique valide

Privé : pour un usage de TCP/IP non connecté à l'Internet

- gestion autonome d'un plan d'adressage (avec adresses uniques)
- utilisation de plages d'adresses spécifiques **recommandée** :
  - ✓ **adresses non routées** (adresses privées) :
    `10.0.0.0/8` (1 ex-classe A)
    `172.16.0.0/12` (16 ex-classe B)
    `192.168.0.0/16` (256 ex-classes C)
    `169.254.0.0/16` (*link local block* pour l'auto-configuration)
    - ☞ utilisable dans chaque *internet* privé
    - ☞ même en cas de connexion à l'Internet, ce trafic n'est pas relayé
    - ☞ possibilité de "sortir" du réseau privé à l'aide de :
      - ➠ serveurs proxys
      - ➠ conversion d'adresses **NAT**

IPv4 : Translation des adresses

Network Address Translation (NAT)



pictures from TANENBAUM A. S. *Computer Networks 4rd edition*

IPv4 : NAT, DNAT et NAPT

Plusieurs approches de la conversion d'adresses :

NAT statique : correspondance fixe d'adresses

NAT dynamique : correspondance dynamique d'adresses

☞ table d'adresses dynamique :

Adresse entrante	adresse sortante
10.0.0.3	192.33.182.117
10.0.0.4	192.33.182.118
...	...

NAPT (*NAT overload*) : correspondance dynamique vers une adresse (ou plusieurs adresses) avec surcharge

☞ utilisation des ports

☞ table dynamique (pour chaques protocoles) :

Proto	Adresse entrante	Port entrant	adresse sortante	Port sortant
TCP	10.0.0.3	1027	192.33.182.117	1027
TCP	10.0.0.4	1027	192.33.182.117	1028
UDP	10.0.0.4	31765	192.33.182.117	31765
...	...	...	...	...

IPv4 : Mécanismes NAPT

Où sont modifiée les adresses ?

➠ au niveau de la carte d'interface :

NAT en entrée ➠ **processus de routage** ➠ NAT en sortie

Modifications annexes :

- le *checksum* des entêtes doit être recalculé
  - ✓ **NAT** IP, TCP et UDP (adresse + *pseudo-header*)
  - ✓ **NAPT** IP, TCP et UDP (adresse + *pseudo-header* + port)
- les adresses et ports paramètres de protocoles applicatifs doivent être aussi modifiées (commande PORT de FTP)
- les messages ICMP sont analysés

IPv4 : NAT et IETF

Un standard publié : RFC 1631

- NAPT **fortement utilisé** actuellement
  - ✓ entreprises (flexibilité)
  - ✓ fournisseurs de services (manque d'adresses)
  - ✓ particuliers (n'ont qu'une adresse)
- pose qqs **problèmes**
  - ✓ architecturaux :
    - ☞ les ports doivent identifier des processus et non des machines
    - ☞ les routeurs modifient les paramètres de la couche transport
    - ☞ **principe de bout-en-bou**t : deux hôtes doivent communiquer directement
  - ✓ sécuritaires : incompatible avec les mécanismes d'**authentification**
  - ✓ techniques : comment "entrer" dans le réseau translaté
- **solutions**
  - ✓ court terme ➠ conversions statiques, serveurs intermédaires (UDP)
  - ✓ long terme ➠ IPv6

Plan

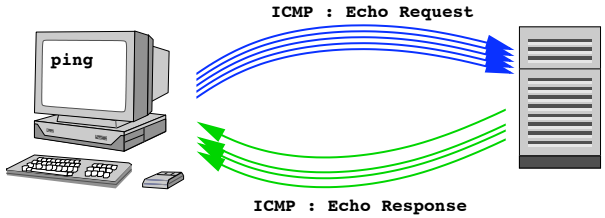
IPv4 : ICMP

Internet Control Message Protocol (RFC792)

- encapsulé dans un paquet IP (mais appartient à la couche 3)
- test et diagnostique du réseau :

ICMP Type	Code	Description
0	0	⟷*echo reply*
3	0	*destination network unreachable*
3	1	*destination host unreachable*
3	2	*destination protocol unreachable*
3	3	*destination port unreachable*
3	6	*destination network unknown*
3	7	*destination host unknown*
4	0	*source quench*
8	0	⟼*echo request*
9	0	*router advertisement*
10	0	*router discovery*
11	0	*TTL expired*
11	1	*reassembly time exeeded*
12	0	*IP header bad*

ICMP : ECHO

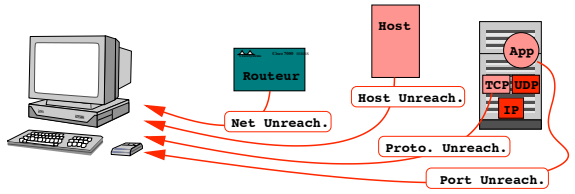


Type **8** (Echo Request) **0** (Echo Response)	Code 0 0	Checksum	Identifier	Seq. Num.	Data
1 octet	1	2	2	2	...

Teste l'accessibilité d'un équipement

- utilisé par la commande `ping` :
  - ✓ indique la connectivité et la disponibilité d'IP chez le destinataire
  - ✓ plusieurs messages permettent d'estimer le RTT et le taux de perte

ICMP : Destination inaccessible

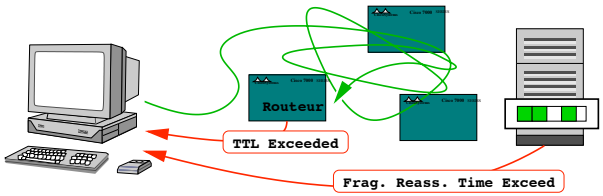


Type **3**	Code **0** (Net Unreachable) **1** (Host Unreachable) **2** (Protocol Unreachable) **3** (Port Unreachable)	Checksum	Unused	Data IP Header + 64 bits
1 octet	1	4	2	(IHL * 4) + 8

Messages émis lorsque la destination n'est pas accessible.

- l'entête IP et une partie de la couche transport sont retournés
  - ✓ @ source = créateur du message ICMP
  - ✓ @ destination = @ source de l'émetteur du paquet en cause

ICMP : *Timeout*



Type **11**	Code **0** (Time To Live Exceeded) **1** (Frag. Reass. Time Exceeded)	Checksum	Unused	Data IP Header + 64 bits
1 octet	1	4	2	(IHL * 4) + 8

Messages émis lorsque le temps de vie ou de réassemblage est dépassé.

- l'entête IP et une partie de la couche transport sont retournés
  - ✓ @ source = créateur du message ICMP
  - ✓ @ destination = @ source de l'émetteur du paquet en cause
- utilisé par la commande `traceroute`

ICMP : Autres messages

- **Source Quench** (Type **4**)
  - ✓ indique une congestion à la source
    - ☞ pas de signalisation de fin de congestion
- **Redirection** (Type **5**)
  - ✓ indique si une meilleure route est disponible
    - ☞ configuration minimale des hôtes
- autres messages principalement pour l'**autoconfiguration**

Plan

IPv4 : RARP

Reverse Adresse Résolution Protocol (RFC 903)

inverse du protocole ARP (réseaux à diffusion)

- obtention d'une @ IP à partir de @ MAC au démarrage
  - ✓ hôtes sans disques (terminaux X, imprimantes...)
  - ✓ hôtes mobiles (portable changé de réseau...)
- utilisation d'un **serveur** (`rarpd`)
  - ✓ mise en correspondance de `/etc/ethers` et de `/etc/hosts`
- format des trames identique à ARP
  - ✓ type Ethernet : `0x8035`
    - ☞ `code 3` pour une requête RARP
    - ☞ `code 4` pour une réponse RARP
- exemple d'autoconfiguration :
  - ✓ la nouvelle station déclanche un échange **RARP**
  - ✓ la station demande le *netmask* par un echange **ICMP**
  - ✓ la station demande au serveur RARP son programme de démarrage par TFTP

IPv4 : BOOTP

BOOT Protocol (RFC 951 et RTF 1542)

- Protocole **portable**, sur UDP
  - ✓ requête sur le port **68**, réponse sur le port **67**
  - ✓ *quelles addresses IP utiliser lorqu'on n'en connait aucunes ?*
    - ☞ @ IP de diffusion (`255.255.255.255`)
    - ☞ @ IP par défaut (`0.0.0.0`)
  - ✓ permet d'atteindre un serveur sur un autre réseau
    - ☞ à travers des agents BOOTP relais
  - ✓ nombreuses extensions (RFC 1533)
    - ☞ *netmask*
    - ☞ liste des **routeurs** du sous-réseau
    - ☞ liste de **serveurs NTP**
    - ☞ liste des **serveurs de noms** (DNS)
    - ☞ liste des serveurs d'impression (LPD et autres)
    - ☞ *hostname* et *domainname*
    - ☞ TTL par défaut ...

IPv4 : DHCP(1)

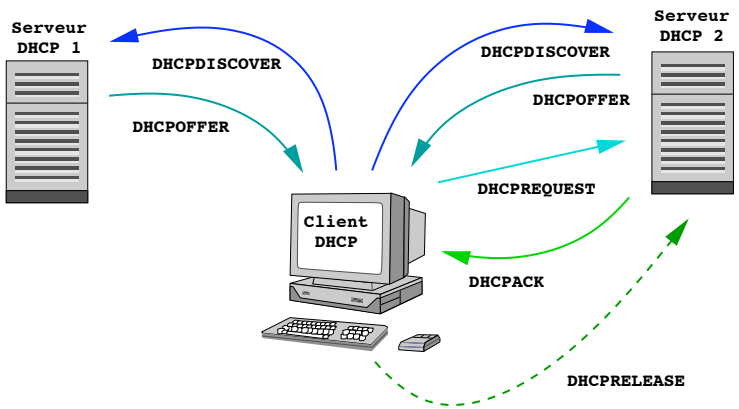
Dynamic Host Configuration Protocol (RFC 2131)

Extension compatible de BOOTP avec **gestion dynamique des @ IP**

- attribution dynamique par **bail** (*lease*) limité dans le temps
  - ✓ bail renouvelé périodiquement si nécessaire
- nouvelles **options DHCP** (extensions BOOTP)

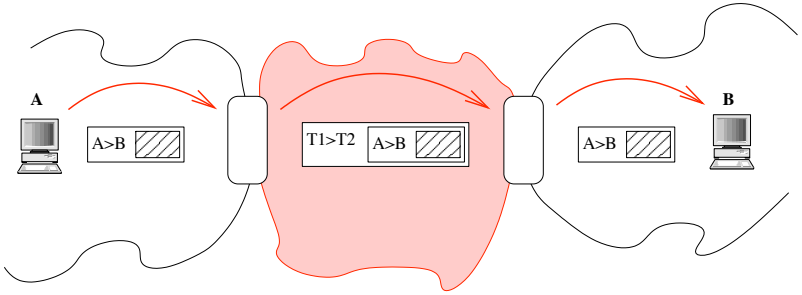
DHCPDISCOVER	C➠ S	localisation du serveur
DHCPOFFER	S➠ C	proposition au client
DHCPREQUEST	C➠ S	confirmation d'une propositon
DHCPACK	S➠ C	validation d'une configuration
DHCPNACK	S➠ C	invalidation d'une configuration
DHCPDECLINE	C➠ S	refus d'une configuration invalide
DHCPRELEASE	C➠ S	libération d'une configuration
DHCPINFORM	C➠ S	demande d'information autre que @ IP
DHCPFORCERENEW	S➠ C	demande de reconfiguration

IPv4 : DHCP(2)



Plan

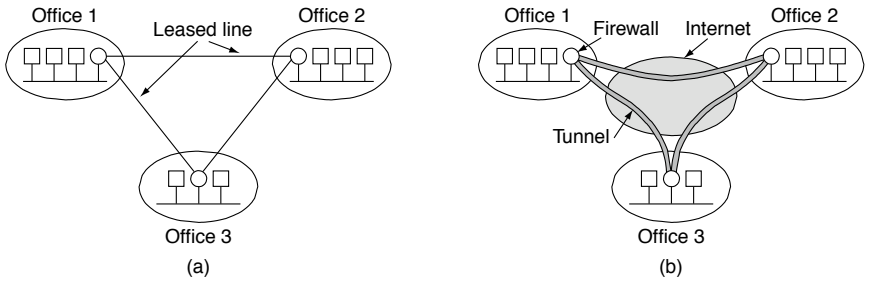
Tunneling



- **encapsulation** alternative à la traduction *(translation)*
- traversées de zones avec des protocoles différents
  - ✓ ex : relier des ilots avec des protocoles non généralisés (IPmulticast, IPv6...)
- contrôle du flux de T1 à T2 (IPv4 dans IPv4, VPN...)
  - ✓ VPN...

Couche IPv4 : VPN

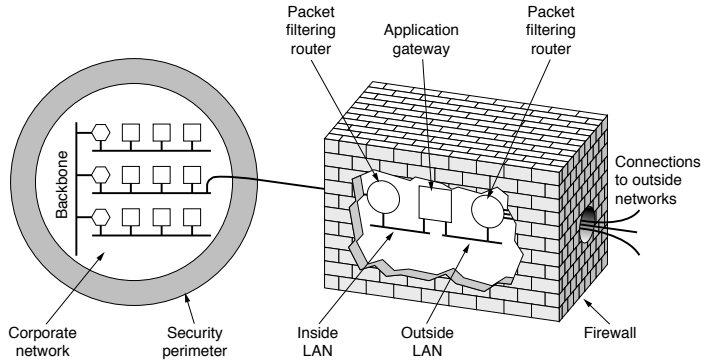
- intégration avec des mécanismes de sécurité, automatisation
  - ✓ IPSEC : confidentialité et integrité (RFC 4301 à 4309)
  - ✓ AAA *(Authentification, Autorisation, Accounting)*
- autres approches VPN au niveau de la couche 2 (PPP)...



pictures from Tanenbaum A. S. *Computer Networks 4rd edition*

IPv4 : Filtrage d'adresses

Firewall...



pictures from Tanenbaum A. S. *Computer Networks 3rd edition*

Fin

Document réalisé avec LaTeX.
Classe de document `foils`.
Dessins réalisés avec `xfig`.

Olivier Fourmaux, `olivier.fourmaux@upmc.fr`
`http://www-rp.lip6.fr/~fourmaux`

Ce document est disponible en format PDF sur le site :
`http://www-master.ufr-info-p6.jussieu.fr/`