Protocole IP

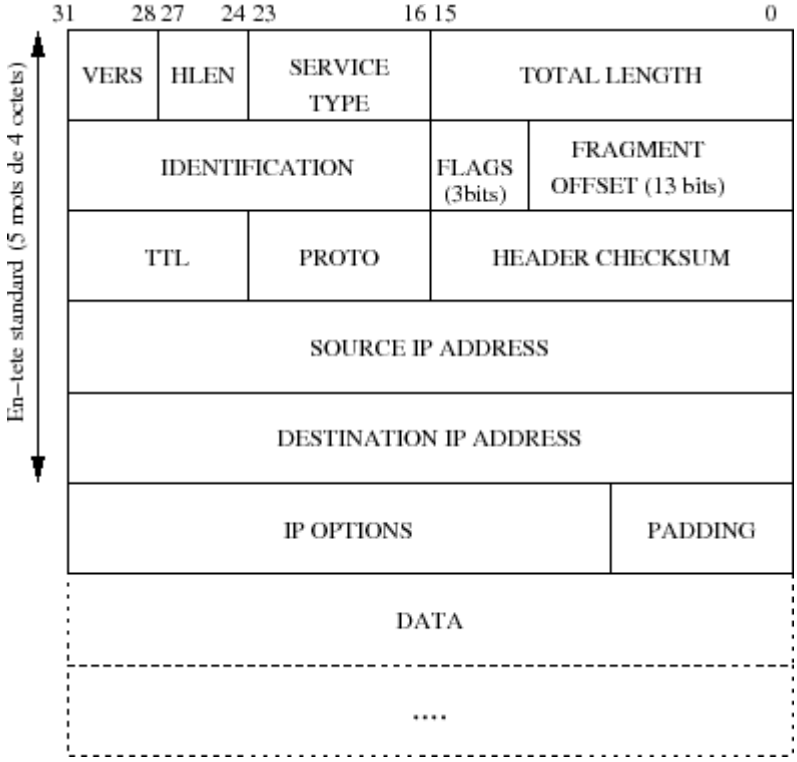
Table des matières

IP est l'acronyme de **I**nternet **P**rotocol, il est défini dans la RFC 791 et a été conçu en 1980 pour remplacer NCP (Network Control Protocol), le protocole de l'Arpanet. Presque trente ans après sa première implémentation, ses limitations se font de plus en plus pénalisantes pour les nouveaux usages sur les réseaux.

1. Datagramme IP

1.1. Structure de l'en-tête

Les octets issus de la couche de transport et encapsulés à l'aide d'un en-tête IP avant d'être propagés vers la couche réseau (Ethernet par exemple), sont collectivement nommés datagramme IP, datagramme Internet ou datagramme tout court. Ces datagrammes ont une taille maximale liée aux caractéristiques de propagation du support physique, c'est le Maximum Transfer Unit ou MTU.



Quelques caractéristiques du protocole IP :

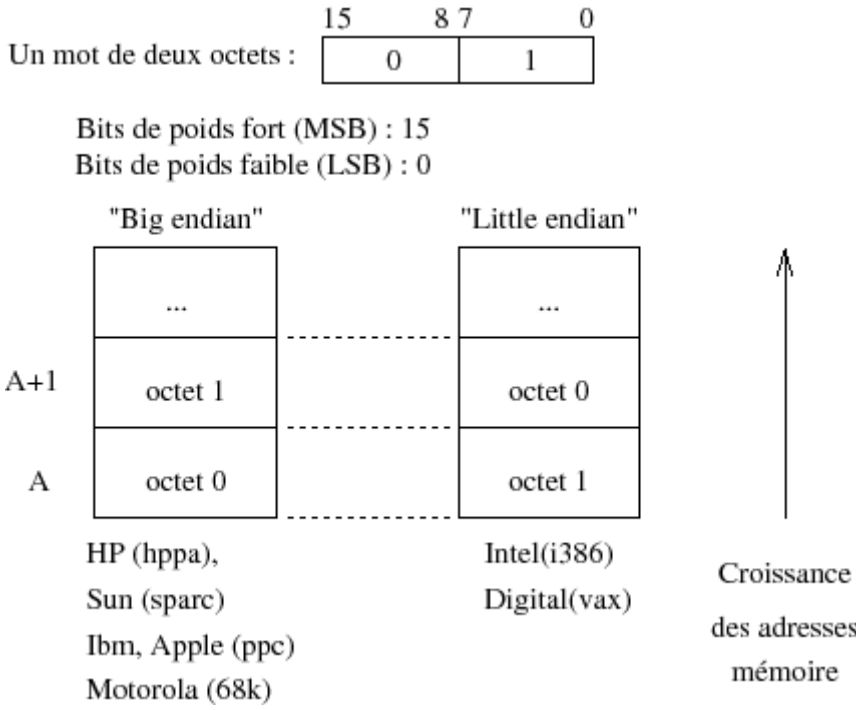
- IP est le support de travail des protocoles de la couche de transport, UDP, TCP et SCTP.
- IP ne donne aucune garantie quant au bon acheminement des données qu'il envoie. Il n'entretient aucun dialogue avec une autre couche IP distante, on dit aussi qu'il délivre les datagramme « au mieux ».
- Chaque datagramme est géré indépendamment des autres datagrammes même au sein du transfert des octets d'un même fichier. Cela signifie que les datagrammes peuvent être mélangés, dupliqués, perdus ou altérés !

  Ces problèmes ne sont pas détectés par IP et donc il ne peut en informer la couche de transport.
- Les octets sont lus et transmis au réseau en respectant le Network Byte Order ou NBO quelle que soit l'architecture cpu de l'hôte.
- L'en-tête IP minimale fait 5 mots de 4 octets, soit 20 octets. S'il y a des options la taille maximale peut atteindre 60 octets.

1.2. Network Byte Order

Sur la *figure ci-dessus* les bits les plus significatifs de chaque mot de quatre octets sont à gauche (31...). Ils sont d'ailleurs transmis sur le réseau dans cet ordre, c'est un standard, c'est le Network Byte Order.

Toutes les architectures de CPU ne sont pas bâties sur le même modèle :



Les termes « Big endian » et « Little endian » indiquent quelle est la terminaison (end) de deux octets que l'on écrit en premier le poids fort (big), c'est aussi le sens de l'écriture humaine, ou le poids faible (little).

1.3. Description de l'en-tête

VERS

4 bits qui spécifient la version du protocol IP. L'objet de ce champ est la vérification que l'émetteur et le destinataire des datagrammes sont bien en phases avec la même version. Actuellement c'est la version 4 qui est principalement utilisé sur l'Internet, bien que quelques implémentations de la version 6 existent et soient déjà en expérimentation.

HLEN

4bits qui donnent la longueur de l'en-tête en mots de 4 octets. La taille standard de cette en-tête fait 5 mots, la taille maximale fait : $(2^3 + 2^2 + 2^1 + 2^0)$ x 4 = 60 octets

TOTAL LENGTH

Donne la taille du datagramme, en-tête plus données. S'il y fragmentation (voir plus loin) il s'agit également de la taille du fragment (chaque datagramme est indépendant des autres).

La taille des données est donc à calculer par soustraction de la taille de l'en-tête.

16 bits autorisent la valeur 65535... La limitation vient le plus souvent du support physique (MTU) qui impose une taille plus petite, sauf sur les liaisons de type « hyperchannel ».

TYPE OF SERVICE

Ce champ joue potentiellement deux rôles selon les bits examinés (préséance et type de service). Pratiquement, la préséance ne sert plus et la RFC 1349 définit 4 bits utiles sur les huit (3 à 6). Ceux-ci indiquent au routeur l'attitude à avoir vis à vis du datagramme.

Par exemple, des datagrammes d'un transfert de fichier (`ftp`) peuvent avoir à laisser passer un datagramme repéré comme contenant des caractères frappés au clavier (session telnet).

`0x00`	-	Service normal	Transfert banal
`0x10`	bit 3,D	Minimiser le délai	Session telnet
`0x08`	bit 4,T	Maximiser le débit	Transfert ftp
`0x04`	bit 5,R	Maximiser la qualité	ICMP
`0x02`	bit 6,C	Minimiser le coût	News (nntp)

L'usage de ces bits est mutuellement exclusif.

Les nouveaux besoins de routage on conduit l'IETF a revoir la définition de ce champ. Celle ci partage les huit bits en deux parties, les premiers bits définissent le DSCP (Differentiated Services CodePoints) qui est une version beaucoup plus fine des quatre bits ci-dessus. Les deux derniers bits définissent l'ECN (Explicit Congestion Notification) qui est un mécanisme permettant de prévenir les congestions, contrairement au mécanisme plus ancien basé sur les messages ICMP qui tente de régler le flux en cas de congestion.

Il faut noter que les protocoles de routage qui tiennent compte de l'état des liaisons (OSPF,IS-IS...) sont susceptibles d'utiliser ce champ.

Les deux écritures du champ ne sont pas compatibles entre elles...

IDENTIFICATION, FLAGS et FRAGMENT OFFSET

Ces mots sont prévus pour contrôler la fragmentation des datagrammes. Les données sont fragmentées car les datagrammes peuvent avoir à traverser des réseaux avec des MTU plus petits que celui du premier support physique employé.

TTL

« Time To Live » 8 bits, 255 secondes maximum de temps de vie pour un datagramme sur le net.

Prévu à l'origine pour décompter un temps, ce champ n'est qu'un compteur décrémenté d'une unité à chaque passage dans un routeur.

Couramment la valeur de départ est 32 ou même 64. Son objet est d'éviter la présence de paquets fantômes circulant indéfiniment...

Si un routeur passe le compteur à zéro avant délivrance du datagramme, un message d'erreur ICMP est renvoyé à l'émetteur avec l'indication du routeur. Le paquet en lui-même est perdu.

PROTOCOL
8 bits pour identifier le format et le contenu des données, un peu comme le champ type d'une trame Ethernet. Il permet à IP d'adresser les données extraites à l'une ou l'autre des couches de transport.

HEADER CHECKSUM
16 bits pour s'assurer de l'intégrité de l'en-tête. Lors du calcul de ce checksum ce champ est à 0.

A la réception de chaque paquet, la couche calcule cette valeur, si elle ne correspond pas à celle trouvée dans l'en-tête le datagramme est oublié sans message d'erreur.

SOURCE ADDRESS
Adresse IP de l'émetteur, à l'origine du datagramme.

DESTINATION ADDRESS
Adresse IP du destinataire du datagramme.

IP OPTIONS
24 bits pour préciser des options de comportement des couches IP traversées et destinatrices. Les options les plus courantes concernent :

- Des problèmes de sécurité
- Des enregistrements de routes
- Des enregistrements d'heure
- Des spécifications de route à suivre
- ...

Historiquement ces options ont été prévues dès le début mais leur implémentation n'a pas été terminée et la plupart des routeurs filtrants bloquent les datagrammes IP comportant des options.

PADDING
Remplissage pour aligner sur 32 bits...

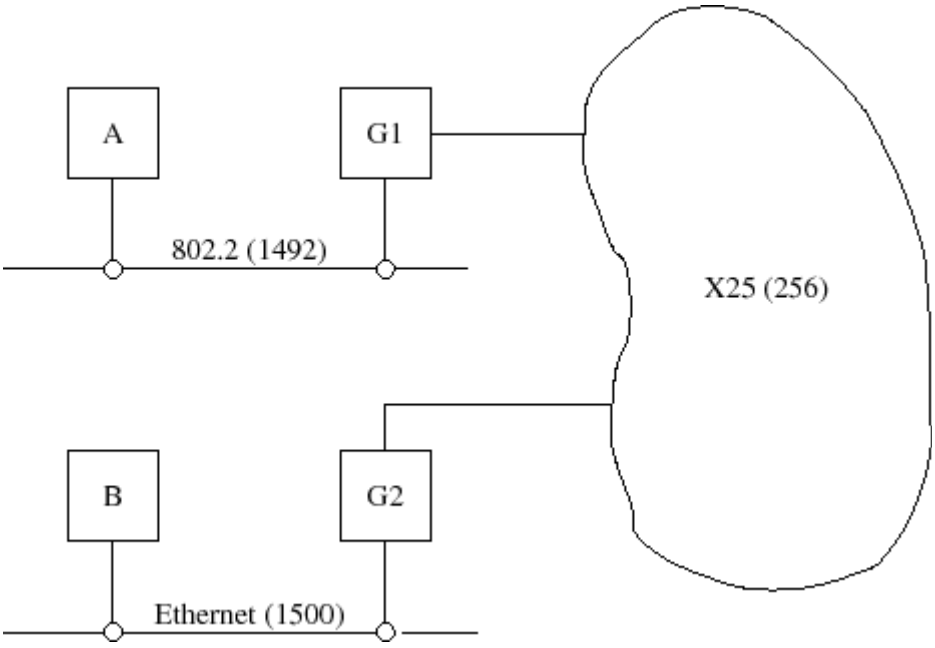
1.4. Fragmentation IP - MTU

La couche de liaison (Couche 2) impose une taille limite, le Maximum Transfer Unit. Par exemple cette valeur est de 1500 pour une trame Ethernet, elle peut être de 256 avec SLIP (Serial Line IP) sur liaison série (RS232...).

Dans ces conditions, si la couche IP doit transmettre un bloc de données de taille supérieure au MTU à employer, il y a fragmentation !

Par exemple, un bloc de 1481 octets sur Ethernet sera décomposé en un datagramme de 1480 (1480 + 20 = 1500) et un datagramme de 1 octet !

Il existe une exception à cette opération, due à la présence active du bit « Don't Fragment bit » du champ FLAGS de l'en-tête IP. La présence à 1 de ce bit interdit la fragmentation dudit datagramme par la couche IP qui en aurait besoin. C'est une situation de blocage, la couche émettrice est tenue au courant par un message ICMP « *Fragmentation needed but don't fragment bit set* » et bien sûr le datagramme n'est pas transmis plus loin.



Fragmentation IP

1.4.1. Fragmentation

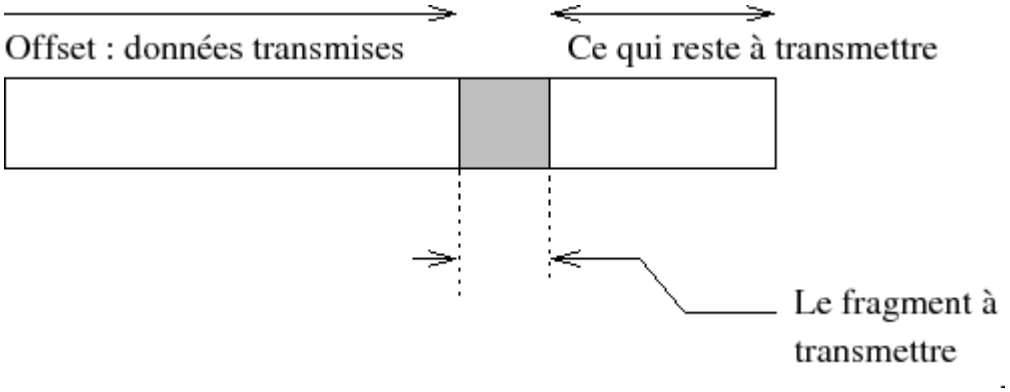
- Quand un datagramme est fragmenté, il n'est réassemblé que par la couche IP destinatrice finale. Cela implique trois remarques :
    1. La taille des datagrammes reçus par le destinataire final est directement dépendante du plus petit MTU rencontré.
    2. Les fragments deviennent des datagrammes à part entière.
    3. Rien ne s'oppose à ce qu'un fragment soit à nouveau fragmenté.
- Cette opération est absolument transparente pour les couches de transport qui utilisent IP.
- Quand un datagramme est fragmenté, chaque fragment comporte la même valeur de champ `IDENTIFICATION` que le datagramme initial.

  S'il y a encore des fragments, un des bits du champ `FLAGS` est positionné à 1 pour indiquer « More fragment » !

  Ce champ a une longueur de 3 bits.

  `FRAGMENT OFFSET` contient l'offset du fragment, relativement au datagramme initial.

  Cet offset est codé sur 13 bits.



Fragment à transmettre

Pour tous les fragments :

1. Les données doivent faire un multiple de 8 octets, sauf pour le dernier fragment, évidement.
2. Le champ `TOTAL LENGTH` change.
3. Chaque fragment est un datagramme indépendant, susceptible d'être à son tour fragmenté.

Pour le dernier fragment :

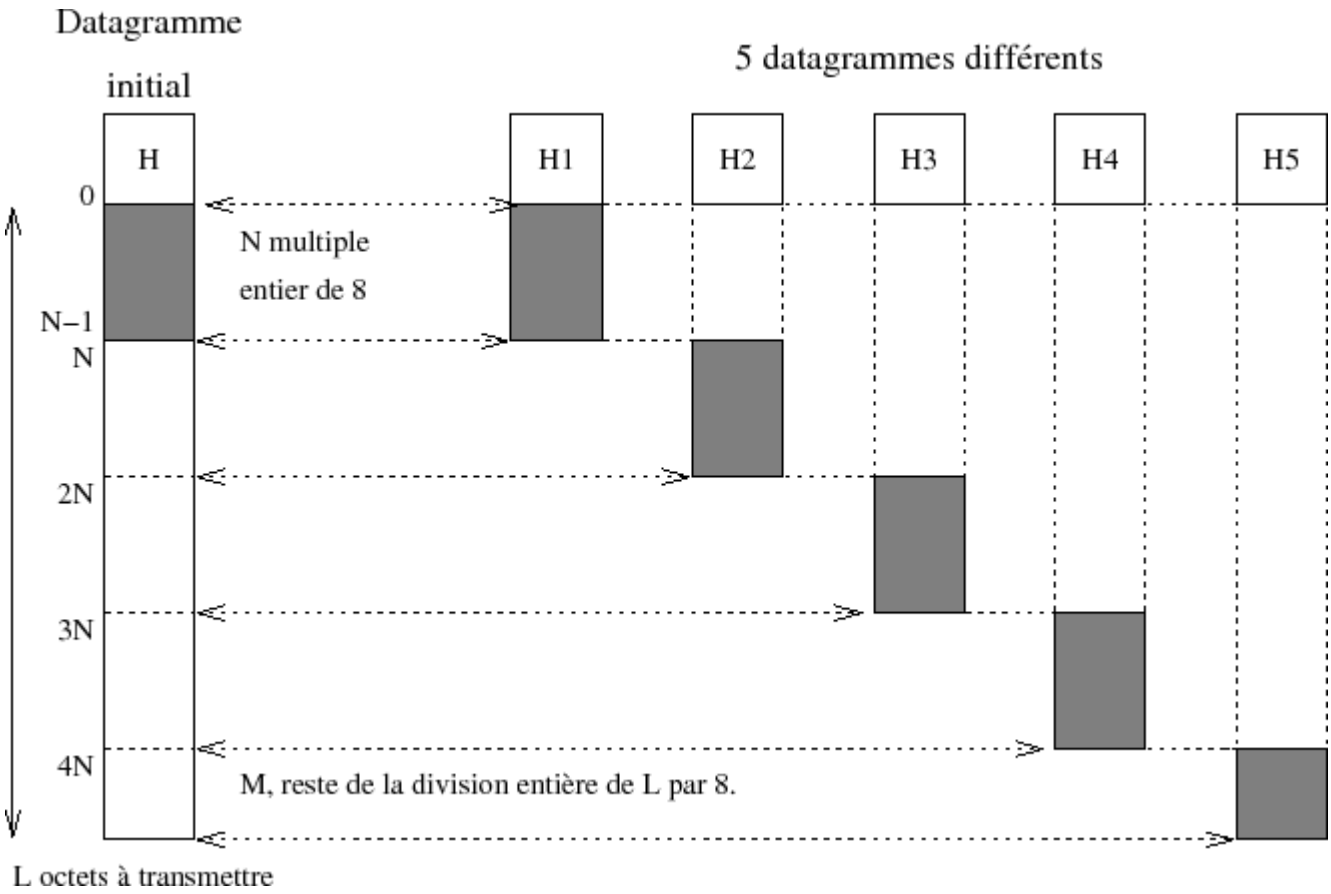
4. `FLAGS` est remis à zéro.
5. Les données ont une taille quelconque.

1.4.2. Réassemblage

- Tous les datagrammes issus d'une fragmentation deviennent des datagrammes IP comme (presque) les autres.
- Ils arrivent à destination, peut être dans le désordre, dupliqués. IP doit faire le tri.
- il y a suffisamment d'information dans l'en-tête pour réassembler les fragments épars.
- **Mais** si un fragment manque, la totalité du datagramme est perdu car aucun mécanisme de contrôle n'est implémenté pour cela dans IP.

  C'est la raison principale pour laquelle il faut absolument éviter de fragmenter un datagramme IP !

La *figure ci-dessous* résume l'opération de fragmentation d'un datagramme IP.



Résumé de la fragmentation

	H1	H2	H3	H4	H5
`IDENTIFICATION`	I	I	I	I	I
`FLAG`	`MF`	`MF`	`MF`	`MF`	0
`OFFSET`	0	N	2 x *N*	3 x *N*	4 x *N*
`TOTAL LENGTH`	*H+N*	*H+N*	*H+N*	*H+N*	*H+M*
`HEADER CHECKSUM`	C_1	C_2	C_3	C_4	C_5

Notez les variations de certains champs de l'en-tête :

1. `IDENTIFICATION` est le même pour tous
2. `FLAG` est 0 pour le dernier datagramme
3. `OFFSET` croît de la taille du fragment, ici N.
4. `TOTAL LENGTH` est généralement différent pour le dernier fragment, sauf cas particulier.
5. `HEADER CHECKSUM` change à chaque fois car l'`OFFSET` change (rappel : il ne tient pas compte des données).

2. Protocole ARP

ARP est l'acronyme de Address Resolution Protocol.

- Le problème à résoudre est issu de la constatation qu'une adresse IP n'a de sens que pour la

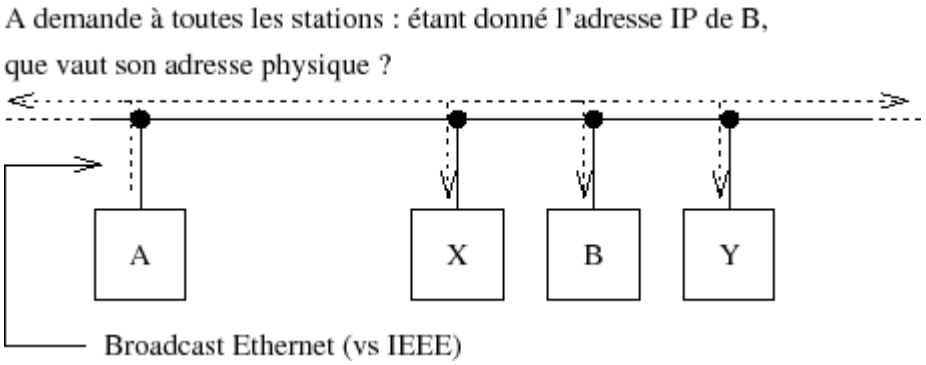
suite de protocole TCP/IP ; celle-ci étant indépendante de la partie matérielle il faut avoir un moyen d'établir un lien entre ces deux constituants.

- La norme Ethernet (vs IEEE) suppose l'identification unique de chaque carte construite et vendue.
- Sur une même liaison physique, Ethernet par exemple, deux machines peuvent communiquer si elles connaissent leurs adresses physiques respectives.

  On suppose qu'une machine connaît sa propre adresse physique par un moyen qui n'est pas décrit ici (ne fait pas partie du protocole).

  Remarque importante : Cette information n'a pas de sens dans le cadre d'une liaison de type « point à point » avec un protocole tel que `ppp`.
- Lors du premier échange entre 2 machines d'un même LAN, si les adresses physiques ne sont pas déjà connues, la solution à ce problème passe par l'usage du protocole `ARP`.
- L'usage de `ARP` est complètement transparent pour l'utilisateur.

2.1. Fonctionnement

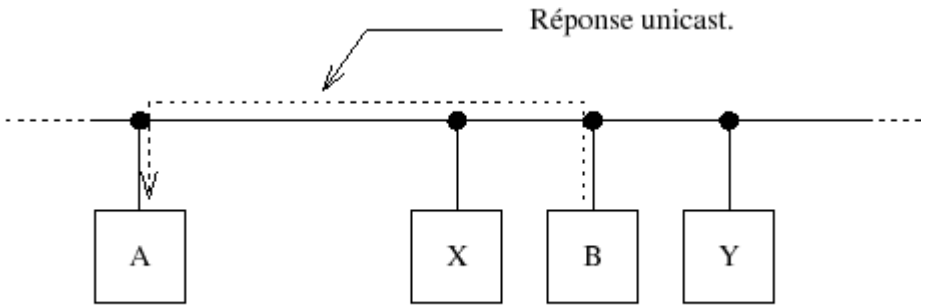


Sur la *figure ci-dessus* la station Ethernet A (I_A, P_A) a besoin de connaître l'adresse physique de la station Ethernet B (I_B, P_B), pour ce faire elle envoie un datagramme de format spécial, dédié à `ARP`, qui lui permet de poser la question à l'ensemble des machines actives. L'adresse de la machine qui doit répondre étant l'objet de la question, son adresse (champ destinataire) est donc remplacée par une adresse de broadcast (48 bits à 1).

Toutes les machines du LAN écoutent cet échange et peuvent mettre à jour leur table de conversion (adresse IP adresse Ethernet) pour la machine A.

Le broadcast, coûteux en bande passante, est ainsi utilisé au maximum de ses possibilités. Sur la *figure ci-dessous* la réponse de B est du type unicast.

Remarque : quand une station Ethernet ne répond plus (cf `ICMP`) il y a suppression de l'association adresse IP - adresse MAC.



B répond directement à A en lui communiquant son adresse physique.

Si la station B ne répond pas, la station continuera à poser la question à intervalles réguliers pendant un temps infini...

Il n'est pas besoin d'utiliser ARP préalablement à chaque échange, car heureusement le résultat est mémorisé.

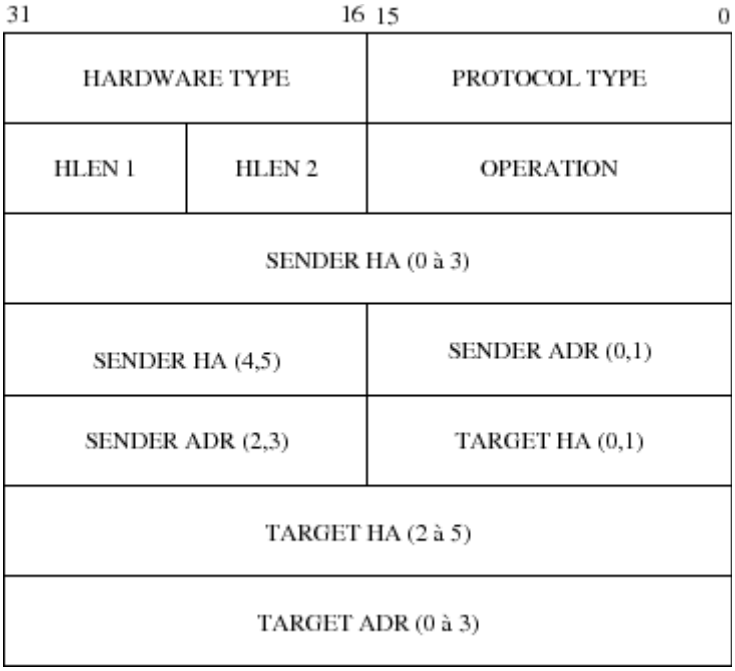
En règle générale la durée de vie d'une adresse en mémoire est de l'ordre de 20 minutes et chaque utilisation remet à jour ce compteur.

La commande `arp -a` sous Unix permet d'avoir le contenu de la table de la machine sur laquelle on se trouve, par exemple :

```
$ arp -a
soupirs.chezmoi.fr  (192.168.192.10) at 8:0:9:85:76:9c
espoirs.chezmoi.fr  (192.168.192.11) at 8:0:9:85:76:bd
plethore.chezmoi.fr (192.168.192.12) at 8:0:9:a:f9:aa
byzance.chezmoi.fr  (192.168.192.13) at 8:0:9:a:f9:bc
ramidus.chezmoi.fr  (192.168.192.14) at 0:4f:49:1:28:22 permanent
desiree.chezmoi.fr  (192.168.192.33) at 8:0:9:70:44:52
pythie.chezmoi.fr   (192.168.192.34) at 0:20:af:2f:8f:f1
ramidus.chezmoi.fr  (192.168.192.35) at 0:4f:49:1:36:50 permanent
gateway.chezmoi.fr  (192.168.192.36) at 0:60:8c:81:d5:1b
```

Enfin, et c'est un point très important, du fait de l'utilisation de broadcast physiques, les messages ARP ne franchissent pas les routeurs.

2.2. Format du datagramme



Datagramme ARP

Le datagramme ci-dessus est encapsulé dans une trame physique du type `0x0806`.

`HARDWARE TYPE`
pour spécifier le type d'adresse physique dans les champs `SENDER HA` et `TARGET HA`, c'est 1 pour Ethernet.

`PROTOCOL TYPE`
pour spécifier le type d'adresse logique dans les champs `SENDER ADR` et `TARGET ADR`, c'est `0x0800` (même valeur que dans la trame Ethernet) pour des adresses IP.

`HLEN 1`
pour spécifier la longueur de l'adresse physique (6 octets pour Ethernet).

`HLEN 2`
pour spécifier la longueur de l'adresse logique (4 octets pour IP).

`OPERATION`
ce champ précise le type de l'opération, il est nécessaire car la trame est la même pour toutes les opérations des deux protocoles qui l'utilisent.

	Question	Réponse
ARP	1	2
RARP	3	4

`SENDER HA`
adresse physique de l'émetteur

SENDER ADR
adresse logique de l'émetteur
TARGET HA
adresse physique du destinataire
TARGET ADR
adresse logique du destinataire

3. Protocole RARP

RARP est l'acronyme de Reverse Address Resolution Protocol (BOOTP et DHCP en sont des alternatives avec plus de possibilités).

- Normalement une machine qui démarre obtient son adresse IP par lecture d'un fichier sur son disque dur (ou depuis sa configuration figée dans une mémoire non volatile).
- Pour certains équipements cette opération n'est pas possible voire même non souhaitée par l'administrateur du réseau :
  - Terminaux X Windows
  - Stations de travail « diskless »
  - Imprimante en réseau
  - PC en réseau
  - ...
- Pour communiquer en TCP/IP une machine a besoin d'au moins une adresse IP, l'idée de ce protocole est de la demander au réseau.
- Le protocole RARP est adapté de ARP : l'émetteur envoie une requête RARP spécifiant son adresse physique dans un datagramme de même format que celui de ARP et avec une adresse de broadcast physique.
- Toutes les stations en activité reçoivent la requête, celles qui sont habilités à répondre (serveurs RARP) complètent le datagramme et le renvoient directement (unicast) à l'émetteur de la requête puisqu'elle connaissent son adresse physique.

Sur une machine Unix configurée en serveur RARP les correspondances entres adresses IP et adresses physiques sont enregistrées dans un fichier nommé généralement `/etc/bootptab`.

4. Protocole ICMP

ICMP est l'acronyme de Internet Control Message Protocol.

Les paquets circulent d'une passerelle vers un autre jusqu'à en trouver une qui puisse les délivrer directement à un hôte. Si une passerelle ne peut router ou délivrer directement un paquet ou si un événement anormal arrive sur le réseau comme un trafic trop important ou une machine indisponible, il faut pouvoir en informer l'hôte qui a émis le paquet. Celui-ci pourra alors réagir en fonction du type de problème rencontré.

ICMP est un mécanisme de contrôle des erreurs au niveau IP, mais le niveau *Application* peut également avoir un accès direct à ce protocole.

4.1. Le système de messages d'erreur

Dans le système que nous avons décrit, chaque passerelle et chaque hôte opère de manière

autonome, route et délivre les datagrammes qui arrivent sans coordination avec l'émetteur.

Le système fonctionne parfaitement si toutes les machines sont en ordre de marche et si toutes les tables de routage sont à jour. Malheureusement c'est une situation idéale...

Il peut y avoir des rupture de lignes de communication, des machines peuvent être à l'arrêt, en pannes, déconnectées du réseau ou incapables de router les paquets parce qu'en surcharge.

Des paquets IP peuvent alors ne pas être délivrés à leur destinataire et le protocol IP lui-même ne contient rien qui puisse permettre de détecter cet échec de transmission.

C'est pourquoi est ajouté systématiquement un mécanisme de gestion des erreurs connu sous le doux nom de `ICMP`. Il fait partie de la couche IP et porte le numéro de protocole **1**.

Ainsi, quand un message d'erreur arrive pour un paquet émis, c'est la couche IP elle-même qui gère le problème, la plupart des cas sans en informer les couches supérieures (certaines applications utilisent `ICMP`).

Initialement prévu pour permettre aux passerelles d'informer les hôtes sur des erreurs de transmission, `ICMP` n'est pas restreint aux échanges passerelles-hôtes, des échanges entres hôtes sont tout à fait possibles.

Le même mécanisme est valable pour les deux types d'échanges.

4.2. Format des messages ICMP

Chaque message ICMP traverse le réseau dans la partie DATA d'un datagramme IP :

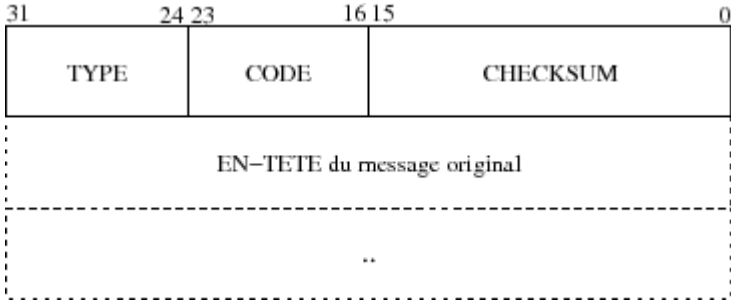
Message ICMP

La conséquence directe est que les messages `ICMP` sont routés comme les autres paquets IP au travers le réseau. Il y a toutefois une exception : il peut arriver qu'un paquet d'erreur rencontre lui-même un problème de transmission, dans ce cas on ne génère pas d'erreur sur l'erreur !

Il est important de bien voir que puisque les messages `ICMP` sont encapsulés dans un datagramme `IP`, `ICMP` n'est pas considéré comme un protocole de niveau plus élevé.

La raison de l'utilisation d'`IP` pour délivrer de telles informations, est que les messages peuvent avoir à traverser plusieurs réseaux avant d'arriver à leur destination finale. Il n'était donc pas possible de rester au niveau physique du réseau (à l'inverse de ARP ou RARP).

La *figure V.10* décrit le format du message `ICMP` :



Format d'un message ICMP

Chaque message ICMP a un type particulier qui caractérise le problème qu'il signale. Un en-tête de 32 bits est composé comme suit :

TYPE
contient le code d'erreur.

CODE
complète l'information du champ précédent.

CHECKSUM
est utilisé avec le même mécanisme de vérification que pour les datagrammes IP mais ici il ne porte que sur le message ICMP (rappel : le checksum de l'en-tête IP ne porte que sur son en-tête et non sur les données véhiculées).

En addition, les messages ICMP donnent toujours l'en-tête IP et les 64 premiers bits (les deux premiers mots de quatre octets) du datagramme qui est à l'origine du problème, pour permettre au destinataire du message d'identifier quel paquet est à l'origine du problème.

5. Routage IP

Sur l'Internet, ou au sein de toute entité qui utilise IP, les datagrammes ne sont pas routés par des machines Unix, mais par des routeurs dont c'est la fonction par définition. Ils sont plus efficaces et plus perfectionnés pour cette tâche par construction, et surtout autorisent l'application d'une politique de routage (routing policy) ce que la pile IP standard d'une machine Unix ne sait pas faire.

Le routage des datagrammes se fait au niveau de la couche IP, et c'est son travail le plus important. Toutes les machines multiprocessus sont théoriquement capables d'effectuer cette opération.

La différence entre un routeur et un hôte est que le premier est capable de transmettre un datagramme d'un interface à un autre et pas le deuxième.

Cette opération est délicate si les machines qui doivent dialoguer sont connectées à de multiples réseaux physiques.

D'un point de vue idéal établir une route pour des datagrammes devrait tenir compte d'éléments comme la charge du réseau, la taille des datagrammes, le type de service demandé, les délais de propagation, l'état des liaisons, le trajet le plus court... La pratique est plus rudimentaire !

Il s'agit de transporter des datagrammes aux travers de multiples réseaux physiques, donc aux travers de multiples passerelles.

On divise le routage en deux grandes familles :

Le routage direct

Il s'agit de délivrer un datagramme à une machine raccordée au même LAN.

L'émetteur trouve l'adresse physique du correspondant (ARP), encapsule le datagramme dans une trame et l'envoie.

Le routage indirect

Le destinataire n'est pas sur le même LAN comme précédemment. Il est absolument nécessaire de franchir une passerelle connue d'avance ou d'employer un chemin par défaut.

En effet, toutes les machines à atteindre ne sont pas forcément sur le même réseau physique. C'est le cas le plus courant, par exemple sur l'Internet qui regroupe des centaines de milliers de réseaux différents.

Cette opération est beaucoup plus délicate que la précédente car il faut sélectionner une passerelle.

Parce que le routage est une opération fondamentalement orientée « réseau », le routage s'appuie sur cette partie de l'adresse `IP` du destinataire. La couche `IP` détermine celle-ci en examinant les bits de poids fort qui conditionnent la classe d'adresse et donc la segmentation « network.host ».

5.1. Table de routage

Sous Unix toutes les opérations de routage se font grâce à une table, dite « table de routage », qui se trouve dans le noyau lui-même. Cette table est très fréquemment utilisée par `IP` : sur un serveur plusieurs centaines de fois par secondes.

Elle est créée

- Au démarrage avec la commande `route`, invoquée dans les scripts de lancement du système.
- Au coup par coup avec la commande `route`, à partir du shell (administrateur système uniquement).
- Dynamiquement avec les démons de routage `routed` ou `gated` (la fréquence de mise à jour est typiquement de l'ordre de 30 sec.).
- Par des messages « `ICMP` redirect ».

La commande `netstat -rn` permet de la visualiser au niveau de l'interface utilisateur :

```
$ netstat -rn
Routing tables

Internet:
Destination          Gateway            Flags
default              192.168.192.36     UGS
127.0.0.1            127.0.0.1          UH
192.168.192/27       link#1             UC
192.168.192.10       8:0:9:85:76:9c     UHLW
192.168.192.11       8:0:9:85:76:bd     UHLW
192.168.192.12       8:0:9:88:8e:31     UHLW
192.168.192.13       8:0:9:a:f9:bc      UHLW
192.168.192.14       0:4f:49:1:28:22    UHLW
192.168.192.15       link#1             UHLW
192.168.192.32/27    link#2             UC
192.168.192.33       8:0:9:70:44:52     UHLW
192.168.192.34       0:20:af:2f:8f:f1   UHLW
192.168.192.35       0:4f:49:1:36:50    UHLW
192.168.192.36       link#2             UHLW
```

Cette table comme est essentiellement composée d'une colonne origine, d'une colonne destination.

De plus, chaque route qui désigne une passerelle (ici la route par défaut) doit s'accompagner d'un nombre de sauts (hop), ou encore métrique, qui permet le choix d'une route plutôt qu'une autre en fonction de cette valeur. Chaque franchissement d'un routeur compte pour un saut. Dans la table ci-

dessus, la métrique de la route par défaut est 1.

Les drapeaux (flags) les plus courants :

- C La route est générée par la machine, à l'usage.
- D La route a été crée dynamiquement (démons de routage).
- G La route désigne une passerelle, sinon c'est une route directe.
- H La route est vers une machine, sinon elle est vers un réseau.
- L Désigne la conversion vers une adresse physique (cf ARP).
- M La route a été modifiée par un `` redirect ".
- S La route a été ajoutée manuellement.
- U La route est active.
- W La route est le résultat d'un clônage.

5.2. Routage statique

Les routes statiques sont celles crées au démarrage de la machine ou ajoutées manuellement par l'administrateur système, en cours de fonctionnement.

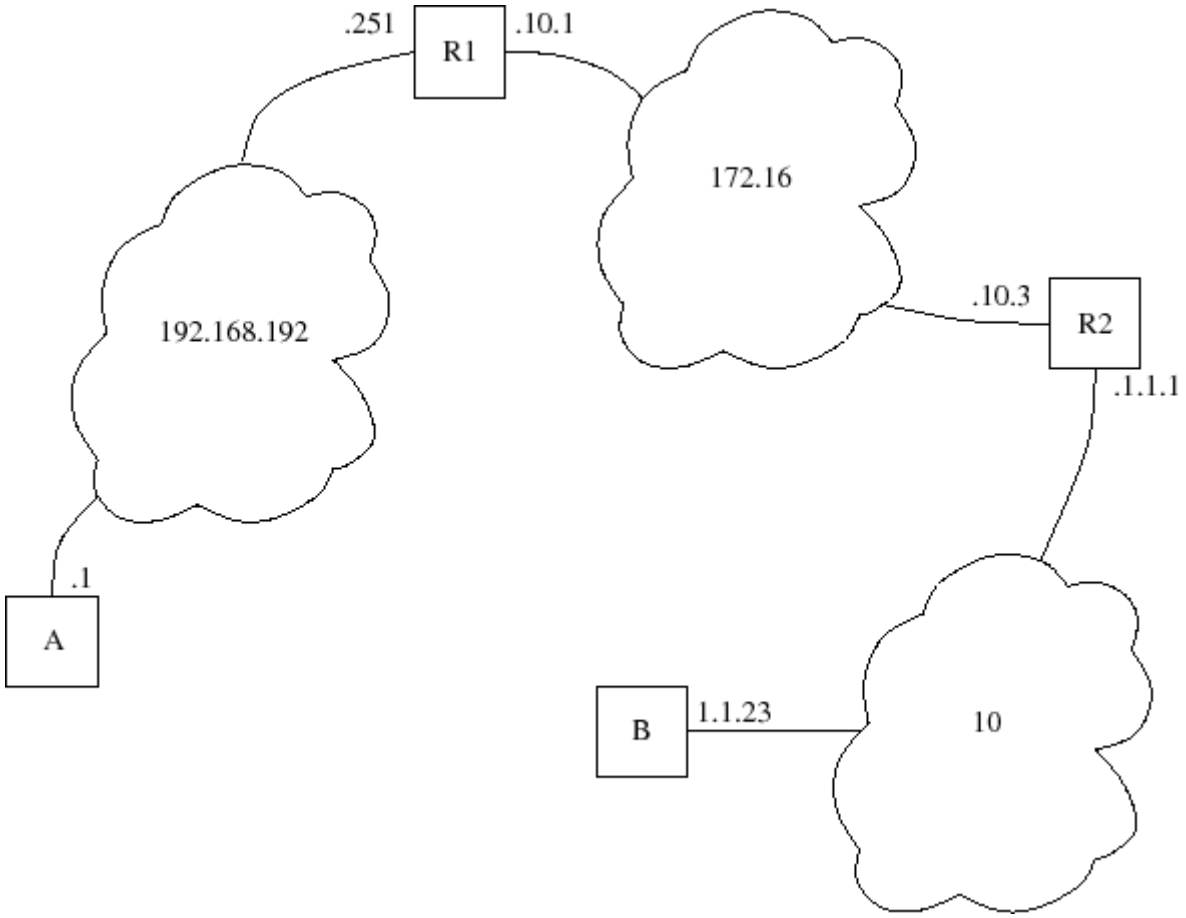
Le nombre de machines possibles à atteindre potentiellement sur l'Internet est beaucoup trop élevé pour que chaque machine puisse espérer en conserver l'adresse, qui plus est, même si cela était concevable, cette information ne serait jamais à jour donc inutilisable.

Plutôt que d'envisager la situation précédente on préfère restreindre l'étendue du « monde connu » et utiliser la stratégie de proche en proche précédement citée.

Si une machine ne peut pas router un datagramme, elle connaît (ou est supposée connaître) l'adresse d'une passerelle supposée être mieux informée pour transmettre ce datagramme.

Exemple simplifié des tables de routage statiques présentes sont les machines A, B, R1 et R2 :

Machine A
default : 192.168.192.251
Machine B
default : 10.1.1.1
Routeur R1
10 : 172.16.10.3
Routeur R2
192.168.192 : 172.16.10.1



Exemple de nuage avec routage statique

5.3. Routage dynamique

Si la topologie d'un réseau offre la possibilité de plusieurs routes pour atteindre une même destination, s'il est vaste et complexe, sujet à des changements fréquents de configuration...Le routage dynamique est alors un bon moyen d'entretenir les tables de routages et de manière automatique.

Il existe de nombreux protocoles de routage dynamique dont certains sont aussi anciens que l'Internet. Néanmoins tous ne conviennent pas à tous les types de problème, il en existe une hiérarchie.

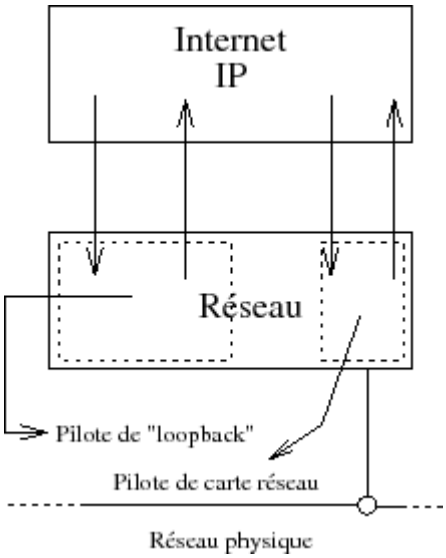
Schématiquement on peut imaginer l'Internet comme une hiérarchie de routeurs. Les routeurs principaux (core gateways) de cette architecture utilisent entres-eux des protocoles comme GGP (Gateway to Gateway Protocol), l'ensemble de ces routeurs forment ce que l'on nomme l'« Internet Core ».

En bordure de ces routeurs principaux se situent les routeurs qui marquent la frontière avec ce que l'on nomme les Autonomous systems, c'est à dire des systèmes de routeurs et de réseaux qui possèdent leurs mécanismes propres de propagation des routes. Le protocole utilisé par ces routeurs limitrophes est souvent EGP (Exterior Gateway Protocol) ou BGP (Border Gateway Protocol).

5.4. Interface de `loopback`

Toutes les implémentations d'`IP` supportent une interface de type `loopback`. L'objet de cette interface est de pouvoir utiliser les outils du réseau en local, sans passer par un interface réseau réel (associé à une carte physique).

La *figure* ci-contre, montre que la couche `IP` peut utiliser, selon le routage, l'interface standard du réseau, où l'interface de `loopback`.



Le routage est ici bien sûr basé sur l'adresse `IP` associée à chacune des interfaces. Cette association est effectuée sur une machine Unix à l'aide de la commande `ifconfig`, qui établit une correspondance entre un pilote de périphérique (repéré par son fichier spécial) et une adresse `IP`.

Dans le cas du pilote de `loopback`, l'adresse est standardisée à n'importe quelle adresse valide du réseau 127.

La valeur courante est 127.0.0.1

Dans toutes les machines Unix modernes cette configuration est déjà prévue d'emblée dans les scripts de démarrage.

6. Comment ça marche ?

Soit deux réseaux privés : 192.168.10.0 et 192.168.20.0 et faisons l'hypothèse que la passerelle fonctionne comme une machine Unix qui ferait du routage entre deux de ses interfaces !

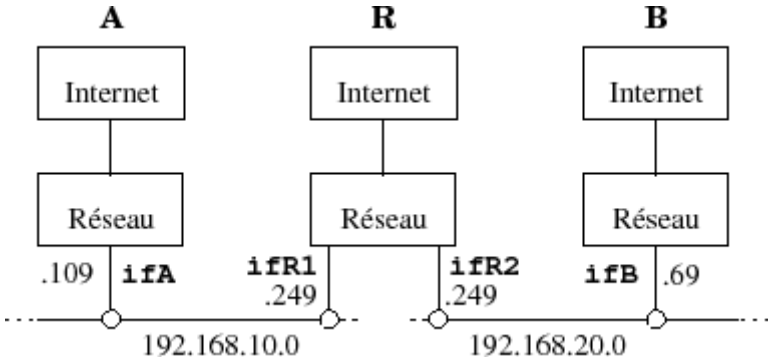


Illustration du routage direct et indirect

Ce tableau résume l'adressage physique et logique de la situation :

Interface	Adresse MAC	Adresse IP
`ifA`	`08:00:20:20:cf:af`	192.168.10.109
`ifB`	`00:01:e6:a1:07:64`	192.168.20.69
`ifR1`	`00:06:5b:0f:5a:1f`	192.168.10.249
`ifR2`	`00:06:5b:0f:5a:20`	192.168.20.249

Nous faisons en outre les hypothèses suivantes :

1. Les caches ARP des machines **A**, **B** et **R** sont vides

2. La machine **A** a connaissance d'une route vers le réseau 192.168.20 passant par 192.168.10.249 et réciproquement la machine **B** voit le réseau 192.168.10.0 via le 192.168.20.249
3. La machine **A** a connaissance de l'adresse IP de la machine **B**

La machine A envoie un datagramme à la machine B, que se passe t-il sur le réseau ?

Étape 1

La machine **A** applique l'algorithme de routage et s'aperçoit que la partie réseau de l'adresse de **B** n'est pas dans le même LAN (192.168.10/24 et 192.168.20/20 diffèrent).

L'hypothèse 2 entraîne qu'une route route existe pour atteindre ce réseau, passant par **R**. L'adresse IP de **R** est dans le même LAN, **A** peut donc atteindre **R** par un routage direct. La conséquence de l'hypothèse 1 implique que pour atteindre **R** directement il nous faut d'abord déterminer son adresse physique. Le protocole ARP doit être utilisé.

A envoie en conséquence une trame ARP comportant les éléments suivants :

`SENDER HA`	`08:00:20:20:cf:af`
`SENDER ADR`	192.168.10.109
`TARGET HA`	`ff:ff:ff:ff:ff:ff`
`TARGET ADR`	192.168.10.249

Avec un champ `OPERATION` qui contient la valeur 1, comme « question ARP ».

Remarque : ici l'adresse IP destination est celle de **R** !

Étape 2

R répond à la « question ARP » par une « réponse ARP » (`OPERATION` contient 2) et un champ complété :

`SENDER HA`	`00:06:5b:0f:5a:1f`
`SENDER ADR`	192.168.10.249
`TARGET HA`	`08:00:20:20:cf:af`
`TARGET ADR`	192.168.10.109

Étape 3

A est en mesure d'envoyer son datagramme à **B** en passant par **R**. Il s'agit de routage indirect puisque l'adresse de **B** n'est pas sur le même LAN. Les adresses physiques et logiques se répartissent maintenant comme ceci :

`IP SOURCE`	192.168.10.109
`IP TARGET`	192.168.20.69
`MAC SOURCE`	`08:00:20:20:cf:af`
`MAC TARGET`	`00:06:5b:0f:5a:1f`

Remarque : ici l'adresse IP destination est celle de **B** !

Étape 4

R a reçu le datagramme depuis **A** et à destination de **B**. Celle-ci est sur un LAN dans lequel **R** se trouve également, un routage direct est donc le moyen de transférer le datagramme. Pour la même

raison qu'à l'étape 1 **R** n'a pas l'adresse MAC de **B** et doit utiliser ARP pour obtenir cette adresse. Voici les éléments de cette « question ARP » :

SENDER HA	00:06:5b:0f:5a:20
SENDER ADR	192.168.20.249
TARGET HA	ff:ff:ff:ff:ff:ff
TARGET ADR	192.168.20.69

Étape 5

Et la « réponse ARP » :

SENDER HA	00:01:e6:a1:07:64
SENDER ADR	192.168.20.69
TARGET HA	00:06:5b:0f:5a:20
TARGET ADR	192.168.20.249

Étape 6

Enfin, dans cette dernière étape, **R** envoie le datagramme en provenance de **A**, à **B** :

IP SOURCE	192.168.10.109
IP TARGET	192.168.20.69
MAC SOURCE	00:06:5b:0f:5a:20
MAC TARGET	00:01:e6:a1:07:64

Remarques :

- si les adresses IP n'ont pas changé, les adresses MAC, diffèrent complètement !
- si **A** envoie un deuxième datagramme, les caches ARP ont les adresses MAC utiles et donc les étape 1, 2, 4 et 5 deviennent inutiles...

7. Conclusion

Après ce tour d'horizon sur nous pouvons conclure que l'espace d'adressage trop limité d' IPv4 pas la seule raison qui a motivé le développement d'IPv6 :

1. Son en-tête comporte deux problèmes, la somme de contrôle (checksum) doit être calculée à chaque traitement de datagramme, chaque routeur doit analyser le contenu du champ option.
2. Sa configuration nécessite au moins trois informations que sont l'adresse, le masque de sous réseau et la route par défaut.
3. Issu d'un monde fermé où la sécurité n'était pas un problème, le datagramme de base n'offre aucun service de confidentialité, d'intégrité et d'authentification.
4. Son absence de qualité de service ne répond pas aux exigences des protocoles applicatifs modernes (téléphonie, vidéo, jeux interactifs en réseau, ...).