TCP-IP

Interface réseau

Circuit permettant à une machine quelconque de communiquer sur un réseau. Une machine peut avoir plusieurs interfaces lui permettant d'atteindre plusieurs réseaux distincts (*cas typique de la passerelle*) mais rien n'empêche de relier plusieurs interfaces d'une même machine à un même réseau.

Adresse MAC – Media Access Control

Adresse **physique** fixée à la fabrication d'une interface réseau. Ne devrait pas être changée.
6 octets généralement écrits en hexadécimal comme `A0:45:3f:1E:89:61`.

Les échanges sont tous empaquetés dans une **trame MAC** (*un bloc de données, succession de bits sur le câble réseau*)

MAC-cible	MAC-source	contenu

Quand une interface voit passer une trame sur le réseau, elle n'ouvre la trame que si `MAC-cible` la concerne.

L'adresse `FF:FF:FF:FF:FF:FF` sert au *broadcast*, c'est à dire qu'elle concerne tout le monde.

Problème : L'adresse MAC seule ne permet pas de guider une requête dans un réseau complexe car l'adresse MAC de la cible ne dit rien sur l'endroit du réseau où trouver cette cible.

Adresse IP – Internet Protocol

Adresse **logique** (logique → logiciel) choisie par la l'administrateur de la machine (*dans la mesure où il est autorisé à le faire sur un réseau dont il n'est pas seul administrateur*).

IPv4 = IP version 4, utilise une adresse de 4 octets.
16 octets pour IPv6 en cours de déploiement mais pas forcément utile dans un réseau local.

L'usage est de donner l'adresse sous forme de 4 octets en décimal, par exemple `172.16.45.8`.

L'adresse IP permet aux machines de savoir quelle itinéraire donner à une requête notamment grâce à l'utilisation des **masques**.

Le masque ne fait pas partie de l'adresse IP et n'est jamais transmis. Une machine M1 ne connaît pas le masque de M2 et ne peut donc pas savoir si elles ont le même masque.

Cas d'une machine simple

Parlons d'une machine simple, c'est à dire simple du point de vue du réseau, par exemple un PC, une imprimante... pas un routeur.

Une telle machine n'a qu'une interface réseau qui peut être configurée de façon simplifiée par les 3 paramètres :

- `IP` : adresse IP de l'interface, ex : `172.16.45.8`
- `MASK` : masque, même format que IP, ex : `255.255.0.0`
- `GATEWAY` : adresse IP de la passerelle, ex : `172.16.0.1`

À noter : dans ce cas, on peut confondre machine et interface car la machine n'a qu'une interface réseau et il n'y a donc pas de confusion possible.

L'idée générale est qu'une machine ayant une interface avec `IP` et `MASK` suppose qu'elle pourra communiquer **en utilisant cette interface** avec toute adresse `IP2` telle que
`IP2 & MASK = IP & MASK`

Cas avec table de routage

Un routeur ou un PC muni de plusieurs interfaces réseau aura besoin d'une **table de routage** (*en général, un PC disposant d'une interface wifi et une interface câblée n'active qu'une des 2 à la fois*).

Les paramètres de configuration sont :

- `IP` et `MASK` pour chaque interface
- `GATEWAY` pour l'ensemble de la machine
- une table de routage qui pour chaque interface, reçoit automatiquement les lignes :

cible	masque	passerelle	interface
`IP & MASK`	`MASK`	IP	IP

et pour la passerelle, recherche d'une paire `IP`, `MASK` d'une des interfaces telle que : `GATEWAY & MASK == IP & MASK` et ajout dans la table de :

cible	masque	passerelle	interface
`0.0.0.0`	`0.0.0.0`	`GATEWAY`	`IP`

En l'absense d'`IP`, `MASK` compatible avec `GATEWAY` une interface sera choisie mais cela ne fonctionnera pas.

- possibilité d'ajouter autant de lignes que l'on veut dans la table. Dans la colonne interface on trouvera uniquement des IP des interfaces de la machine. Les lignes seront lu dans l'ordre des masques, du plus grand au plus petit (chaque masque peut-être vu comme un gros nombre entier de 4 octets).

Machine simple a une table de routage

Nous avons évoqué le cas d'une machine simple mais il ne s'agit en fait que d'une présentation simplifiée de la configuration. La machine simple a elle aussi une table de routage.

Si la machine est configurée avec `IP`, `MASK`, `GATEWAY` alors la table sera :

cible	masque	passerelle	interface
`IP & MASK`	`MASK`	`IP`	`IP`
`0.0.0.0`	`0.0.0.0`	`GATEWAY`	`IP`

Soit une machine simple avec `IP = 172.16.1.15`, `MASK = 255.255.0.0` et `GATEWAY = 172.16.0.1`. On a ajouté une route spéciale à sa table.

cible	masque	passerelle	interface
`173.20.5.0`	`255.255.255.0`	`172.16.0.2`	`172.16.1.15`
`170.16.0.0`	`255.255.0.0`	`172.16.1.15`	`172.16.1.15`
`0.0.0.0`	`0.0.0.0`	`172.16.0.1`	`172.16.1.15`

Requête pour `172.20.5.40` valide la première, transmise vers `172.16.0.2`.
Requête pour `172.16.8.32` valide la deuxième, transmise directement sur le sous-réseau.
Requête pour `238.14.5.9` valide la troisième, transmise vers `172.16.0.1`.

Switch

À ce stade de compréhension on peut le considérer comme un simple répéteur. Il sait sur quel port (prise) un câble est branché. Il ignore les IP, il travaille au niveau MAC.

Au démarrage, il ne sait pas quelles sont les adresses MAC des machines branchées à ses ports (*prises*). Chaque trame qu'il reçoit sur un une prise précise `MAC-source`, il en déduit au fur et à mesure l'adresse MAC au bout de chaque port. Il peut construire une table de correspondance `port : MAC`.

Quand arrive une requête à destination d'une adresse `MAC-cible`, Switch cherche dans sa table. S'il ne connaît pas cette adresse, il transmet sur tous les ports de MAC inconnue. S'il connaît cette adresse, il ne transmet que sur le port correspondant.

Autrement il est neutre. Il ne modifie pas le paquet de données qu'il transmet.

Comment un paquet est aiguillé

Ce qui est dit ici est valable pour tous : machine, serveur, routeur...

Deux cas se présentent :

- Nous sommes à l'origine de l'envoi. La machine veut émettre une requête IP quelconque : `ping`, `http`, etc. Le paquet IP est fabriqué et précise les adresses IP de la cible et de la source.

  La machine doit alors créer une trame MAC pour contenir ce paquet.

- Nous sommes sur une machine intermédiaire (typiquement un routeur, une passerelle). La machine a reçu une trame MAC qui lui était adressée (c'était son adresse MAC). Elle l'a donc ouvert et a constaté que la trame contenait un paquet IP. Mais l'adresse IP-cible n'est pas la sienne. Elle doit donc relancer ce paquet pour qu'il poursuive son chemin.

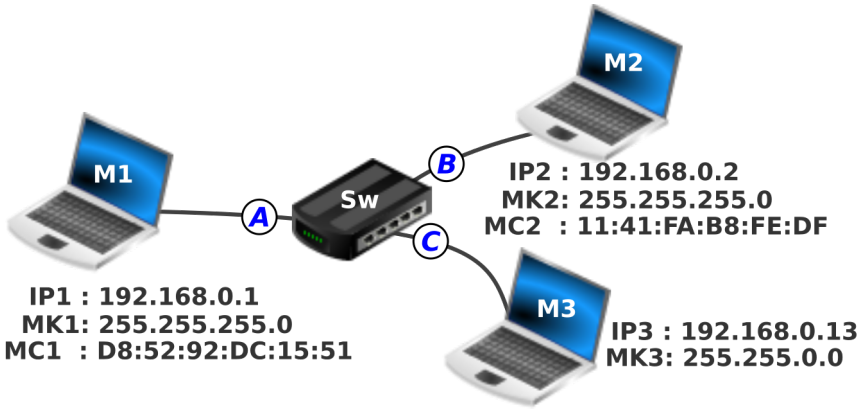
  La machine doit créer une nouvelle trame MAC pour contenir le paquet.

Voyons comment dans chaque cas la trame va être paramétrée.

```
# Je suis Machine M, Config : Interfaces, table de routage
# table routage triée par masques décroissants
# Le paquet IP est destiné à IP-cible

for cible, masque, passerelle, interface in table de routage:
  if cible == IP-cible & masque:
    sélection de cette interface et passerelle
    break

if aucune interface sélectionnée:
  # on ne va pas plus loin.
  échec : adresse introuvable.

MAC-source = MAC de interface sélectionnée

# recherche MAC-dest :
if interface sélectionnée == passerelle sélectionnée:
  # dernière étape de transmission
  IP-cible-étape = IP-cible
else:
  # il faut encore passer par une passerelle
  IP-cible-étape = passerelle

if MAC de IP-cible-étape est inconnu:
  broadcast ARP IP-cible-étape # qui a l'adresse IP-cible-étape ?

mise en attente...

if délai attente dépassé:
  # on ne va pas plus loin
  échec : timeout

enregistrement de MAC-cible-étape dans table ARP # assos MAC : IP

Création trame avec MAC-source, MAC-cible-étape et paquet IP tel quel

Envoi de la trame sur interface sélectionnée

mise en attente de la réponse...
```

On pourra noter `IP1`, `MK1`, `MC1`, `G1` pour `IP`, `MASK`, `MAC`, `GATEWAY` de la machine 1.

Cas réseau simple



Depuis machine M1 on émet `ping 192.168.0.2`. Je note `IP-cible = 192.168.0.2`.

a) `IP1 & MK1 == IP-cible & MK1` donc l'interface `IP1` est sélectionnée sans besoin de passerelle : l'émission se fera directement vers `IP-cible`.

b) Recherche de l'adresse MAC de `192.168.0.2`. Émission d'une trame ARP.

broadcast	MC1	ARP : qui est IP-cible ?

signal émis sur câble A.

c) Sw lit `MAC-cible = broadcast`, il transmet sur B et C.
De plus, Sw note : `port A : D8:52:92:DC:15:51 (MC1)`

d) M3 reçoit la trame. Il lit `MAC-cible = broadcast` donc il ouvre la trame et lit le paquet ARP. `IP3 != IP-cible`, donc M3 reste silencieux.

M2 reçoit la trame et l'ouvre aussi.
`IP2 == IP-cible`, donc M2 va pouvoir répondre par la trame :

MC1	MC2	ARP : je suis IP-cible

e) Sw reçoit sur la trame sur le port B.
Sw note : `port B : 11:41:FA:B8:FE:DF (MC2)`
Sw connaît `MAC-cible = MC1` de la trame, il sait qu'elle se trouve sur le port A (notée précédemment) il répète donc la trame uniquement sur A.

f) `MAC-cible == MC1`, M1 ouvre la trame, note la correspondance `192.168.0.2 : 11:41:FA:B8:FE:DF (MC2)`, et émet la trame :

MC2	MC1	IP1 -> ping IP-cible

g) Sw reçoit, reconnaît `MC2`, répète uniquement sur B.

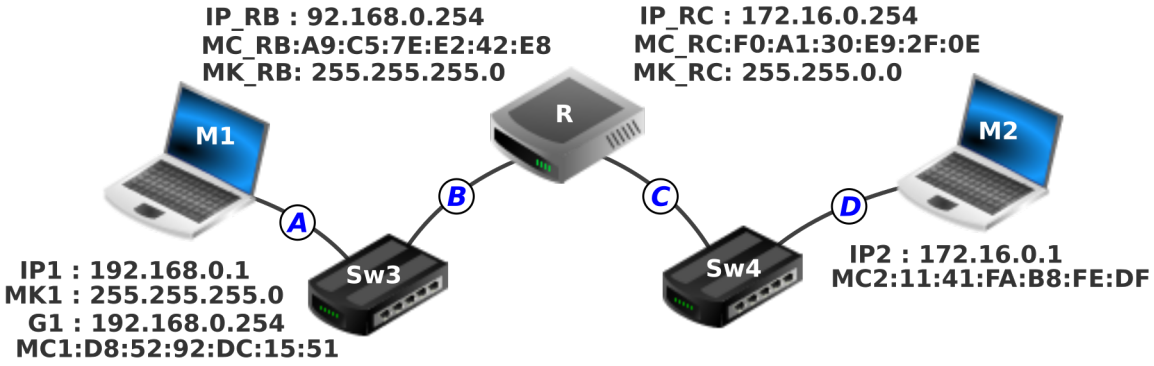
h) `MAC-cible = MC2`, M2 ouvre, reconnaît un `ping`, constate que `IP-cible == IP2`, il répond : | MC1 | MC2 | IP2 -> pong IP1 | que Sw répète de B à A.

Bien qu'ayant un masque différent, M1 et M3 peuvent parfaitement communiquer. En effet, ils sont **physiquement** sur le même sous-réseau et...

- Depuis M1, si on demande `192.168.0.13` on a bien
  `192.168.0.1 & 255.255.255.0 == 192.168.0.13 & 255.255.255.0`
  (*Ne pas oublier que M1 utilise son masque, il n'en connaît pas d'autre.*)
- Depuis M2, si on demande `192.168.0.1` on a bien
  `192.168.0.12 & 255.255.0.0 == 192.168.0.1 & 255.255.0.0`

Chacun des 2 conclut que l'autre est sur le même sous-réseau, qu'il peut être atteint directement sans passerelle, et c'est bien le cas.

Exemple avec routeur et TTL : Time To Live



Depuis machine M1 on émet `ping 172.16.0.1`. Je note `IPcible = 172.16.0.1`

a) `IP1 & MK1 != IPcible & MK1` ⇒ interface `IP1` avec passerelle `G1`.

b) Recherche de l'adresse MAC de `G1`. Émission d'une trame ARP sur A.

broadcast	MC1	ARP : qui est G1 ?

c) Sw3 note `port A : MC1`.

Répétition sur B (et tous les autres s'il y en avait)

R se reconnaît, répond

MC1	MC_RB	ARP : je suis G1

Sw3 note `port B : MC_RB` et répète sur A.

M1 reçoit la réponse et note : `G1 : MAC_RB`.

d) M1 émet sur A

MC_RB	MC1	IP1 -> ping IPcible, TTL = 64

Sw3 répète sur B

e) R reçoit la trame, ouvre la trame qui lui est destinée car `MAC-cible = MC_RB`.

R lit le paquet IP et lit `ping IPcible, TTL = 64` avec `IP-cible != IPs de R`

f) R calcule que `IPcible = 172.16.0.1` sera atteint via `IP_RC` sans passerelle supplémentaire.

R émet sur C

broadcast	MC_RC	ARP : qui est IPcible ?

J'abrège un peu... M2 répond

MC_RC	MC2	ARP : je suis IPcible

g) R peut reprendre le paquet `IP1 -> ping IPcible, TTL = 64` et le remettre dans une trame **en réduisant TTL de 1**

MC2	MC_RC	IP1 -> ping IPcible, TTL = 63

et émet sur C.

h) Sw4 répète sur D et M2 reçoit la trame.

M2 reconnaît `MAC-cible == MC2` donc ouvre la trame.

M2 lit `IP1 -> ping IPcible, TTL = 63` et constate `IPcible == IP2`, il doit répondre.

M2 émet sur D

MC_RC	MC2	IP2 -> pong IP1, TTL = 64

i) Sw4 répète sur C, R se reconnaît comme cible de la trame (toujours en se basant sur l'adresse MAC cible)

R ouvre la trame, lit le paquet l'IP de destination du paquet.

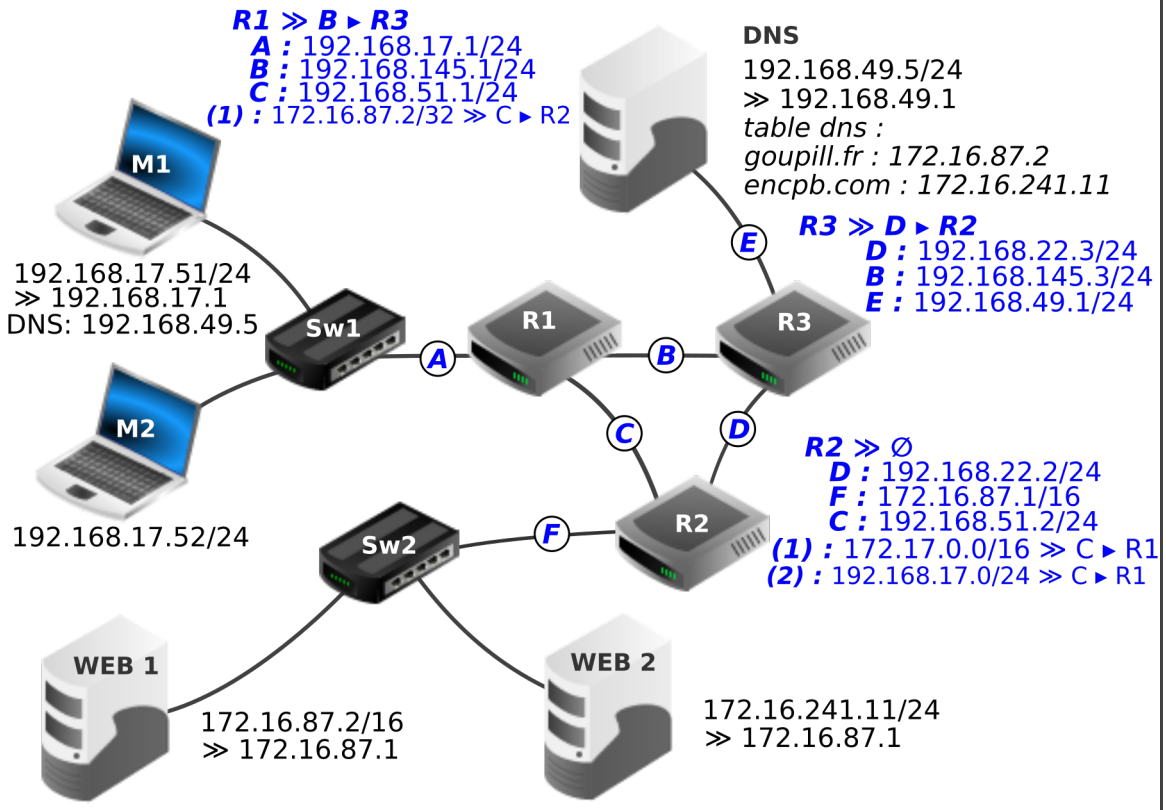
R reforme une trame côté B. Il n'a pas besoin de faire un échange ARP, toutes les adresses MAC utiles sont maintenant connues.

MC1	MC_RB	IP2 -> pong IP1, TTL = 63

émis sur B.

j) M1 reçoit sa réponse.

Quelques exemples de routes



Ce schéma indique toutes les IPs et table de routages utiles avec des abréviations évidentes (≫ pour passerelle). Nous avons détaillé la gestion des MAC, concentrons nous sur les IPs.

- M1 : `ping 172.16.87.2`.
  Comme `192.168.17.51 & 255.255.255.0 != 172.16.87.2 & 255.255.255.0`, transmis à passerelle R1-A. Dans R1, `172.16.87.2` valide la route (1), transmis vers R2. Dans R2, `172.16.87.2` valide F. Transmis direct sur F.
- M1 : `ping 172.16.241.11`.
  Comme `192.168.17.51 & 255.255.255.0 != 172.16.241.11 & 255.255.255.0`, transmis à passerelle R1-A. Dans R1, `172.16.241.1` valide la route par défaut, transmis vers R3. Dans R3, idem, transmis vers R2-D. Dans R2, `172.16.241.11` valide F. Transmis direct sur F.
- M1 : `traceroute 172.16.87.2` indique donc `192.168.17.1`, `192.168.51.2`, `172.16.87.2`, 3 sauts.

  M1 : `traceroute 172.16.241.11` indique donc `192.168.17.1`, `192.168.145.3`, `192.168.22.2`, `172.16.241.11`, 4 sauts.
- WEB2 : `ping 172.16.87.2`. WEB2 pense que la cible est en dehors de son sous-réseau, envoie donc vers R2 qui lui renvoie vers WEB1. La requête aboutit donc en dépit de l'apparente erreur de config.
- M1 : `ping goupill.fr`. Détectant un nom plutôt qu'une ip, M1 envoie une demande vers serveur DNS : M1 → R1 → R3 → DNS. Qui est `goupill.fr` ?
  DNS répond à M1 : `172.16.87.2`, DNS → R3 → R2 → M1.
  M1 reprend `ping 172.16.87.2`.
- M1 : `ping 172.17.0.1`. M1 envoie vers sa passerelle R1.
  L'adresse ne valide que la route par défaut de R1, → R3-B.
  L'adresse ne valide que la route par défaut de R3, → R2-D.
  L'adresse valide la route (1) de R2, → R1-C.
  La trame va ainsi tourner sur les 3 serveurs et pourrait tourner éternellement. C'est le but de TTL = Time To Live. À chaque passage de routeur, TTL diminue de 1 de sorte qu'au bout d'un moment, TTL = 0 et la trame cesse d'être transmise.