

Table des matières

1	Fondamentaux – réseaux diffusion et point-à-point	4
	Le réseau TCP/IP	14
	Adressage des matériels : Adresse IPv4 et Adresse MAC	15
	Routage direct & indirect, encapsulation	22
	Le DNS, « <i>Domain Name Server</i> » : Principe de délégation	35
2	Abstraction du réseau : les «couches»	40
	La notion de protocole	41
	La pile TCP/IP	44
3	La programmation Socket ou la programmation de la couche 4	53
	Notion de port: multiplexage et identification d'un processus	54
	Le protocole TCP : connexion et échanges	56
	Les «Sockets» Berkeley utilisées dans TCP/IP	58
	Les «Sockets» Berkeley utilisées dans TCP/IP : version Python	60



Éléments importants

- ▷ Deux topologies théoriques : diffusion et «point-à-point» ;
- ▷ Les réseaux utilisés et le matériel d'interconnexion ;
- ▷ La gouvernance d'Internet: les organisations et les RFCs ;
- ▷ Le réseau TCP/IP : adressage, encapsulation, routage direct & indirect ;
- ▷ Le DNS : global et local ;
- ▷ La configuration du poste de travail.

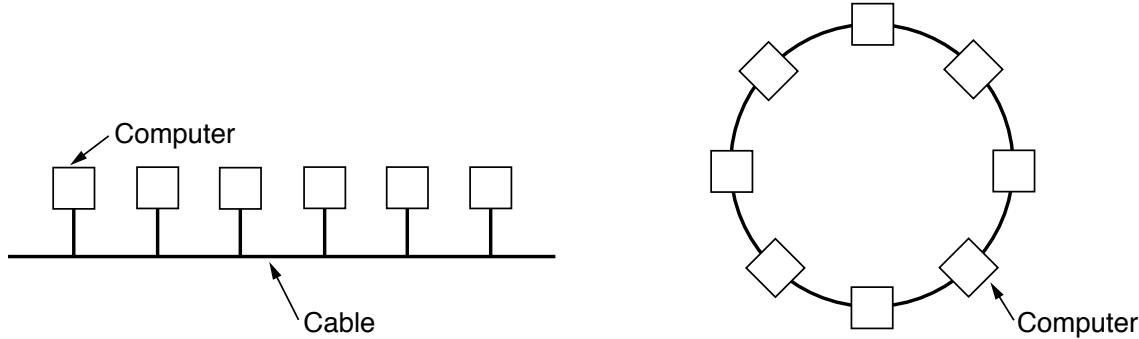


1 Fondamentaux – réseaux diffusion et point-à-point

4

Le réseau en mode «diffusion»

Les réseaux à diffusion, *broadcast network*, n'ont qu'un **seul canal de communication** que toutes les machines partagent.



Une machine envoie de **petits messages** qui sont reçus par toutes les autres machines :

- * dans le message un **champ d'adresse** permet d'identifier le destinataire
- * à la réception du message, une machine teste ce champ :
 - ◊ si le message est pour elle, elle le traite ;
 - ◊ sinon, elle l'ignore.

Exemple :

un couloir sur lequel débouche un certain nombre de portes de bureau.
quelqu'un sort dans le couloir et appelle une personne,
tout le monde entend l'appel mais une seule personne répond à l'appel
cas des annonces dans les gares ou les aéroports

Dans le cas d'Ethernet, mais aussi de WiFi, de Bluetooth...

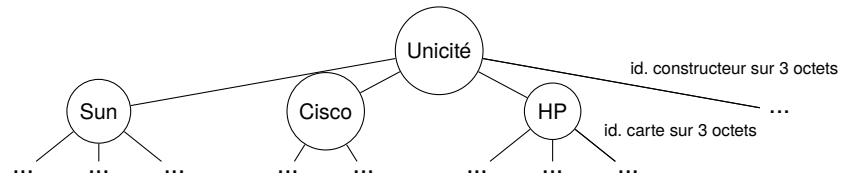
Chaque carte réseau possède une adresse matérielle appelée adresse MAC (Medium Access Control) :

- ▷ **unique** par rapport à toutes les cartes réseaux existantes !
- ▷ **exprimée sur 48 bits** ou 6 octets.
 - ◊ **Syntaxe** : 08:22:EF:E3:D0:FF
 - ◊ **Adresse de Broadcast** : FF:FF:FF:FF:FF:FF (en IPv4).

Pour garantir l'**unicité** :

- a. des «tranches d'adresses» sont affectées aux différents constructeurs :

00:00:0C:XX:XX:XX	Cisco
08:00:20:XX:XX:XX	Sun
08:00:09:XX:XX:XX	HP
00:09:BF:XX:XX:XX	Nintendo
00:D0:F1:XX:XX:XX	Sega



Ce préfixe est appelé OUI, «Organization Unique Identifier».

La liste est consultable à <http://standards.ieee.org/regauth/oui/index.shtml>.

- b. **chaque constructeur** numérote différemment chaque carte réseau qu'il construit.

Avantage

impossible de trouver deux fois la même adresse dans un même réseau

Inconvénient

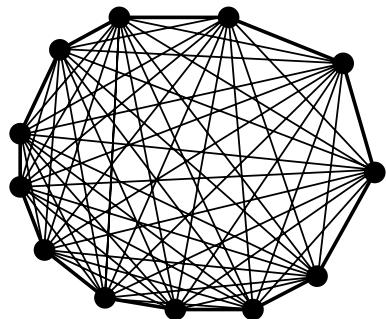
elle ne donne **aucune information sur la localisation** d'une machine
«dans quel réseau est la machine avec qui je veux parler ?»

Solution

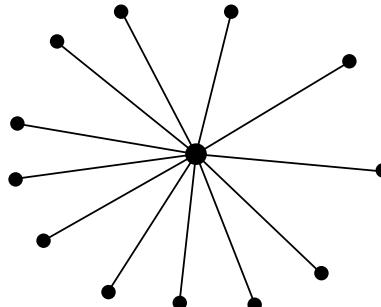
utilisation de l'adresse IP !



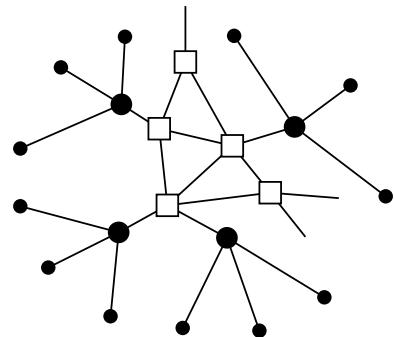
Réseaux en mode «point à point»



(a)



(b)



(c)

Ces réseaux sont formés d'un **grand nombre de connexions** entre les machines prises **deux à deux**.

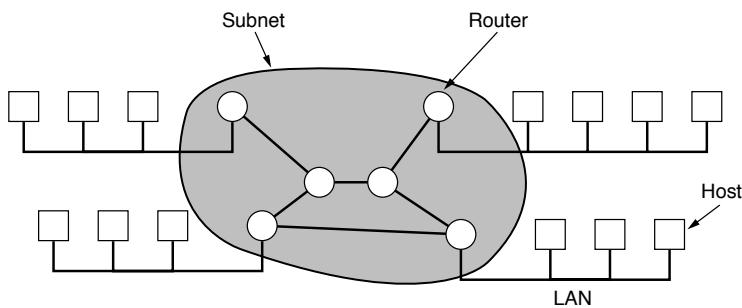
Le trajet des communications est rendu plus complexe :

- ▷ pour aller de la source au destinataire, un message doit alors **passer par un plusieurs intermédiaires**.
- ▷ il existe plusieurs routes de **longueurs différentes** pour joindre ces deux machines, il est nécessaire d'utiliser de **bons algorithmes d'acheminement des messages**;

Inconvénient

- ▷ le **temps de transfert** d'un message devient presque impossible à prévoir.
- ▷ il est nécessaire de faire du **routage** des messages dans le réseau.

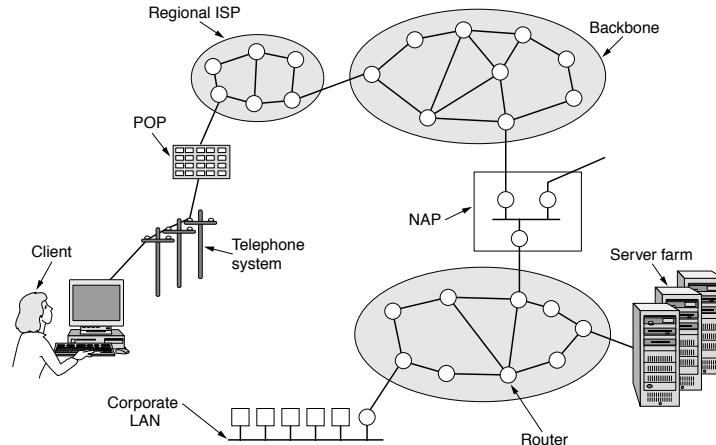




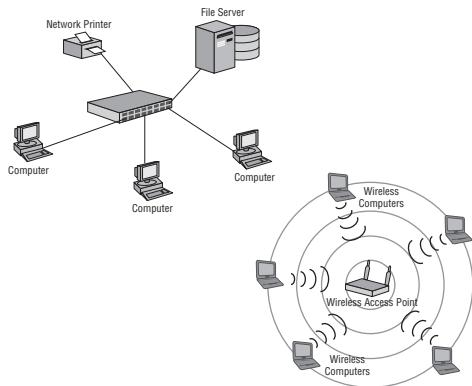
- ▷ **diffusion** : réseau de petite taille, LAN, *Local Area Network* ;
Exemple : Ethernet
- ▷ **point-à-point** : réseau d'interconnexion, constitué uniquement de routeur et de ligne de transmission
Exemple : liaison satellite.
- ▷ la **combinaison des deux** : WAN, *Wide Area Network*.

Inter(connexion)Net(work) :

- * du client à la maison ;
- * de la ligne téléphonique au POP, «Point of Presence» vers ATM ;
- * ou en passant par de la fibre optique pour une liaison IP directe vers l'ISP ;
- * en passant par l'ISP, *Internet Service Provider* ;
- * au réseau national : *backbone* ;
- * par une connexion à un réseau, *Network Access Point* ;
- * vers le LAN de l'entreprise...

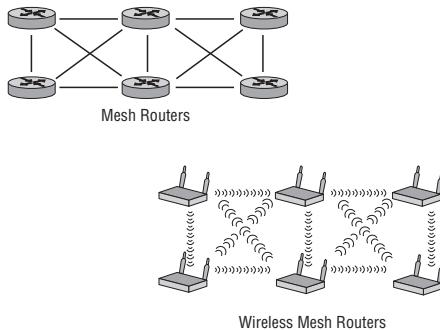


Étoile ou «star»



- * la topologie la plus courante pour définir un LAN : plusieurs matériels connectés à un nœud de connexion central :
 - ◊ un «hub» : un seul **domaine de diffusion** ;
 - ◊ un «switch» ou un point d'accès sans fil : le nœud a la capacité de mettre en relation deux matériels voulant communiquer entre eux (commutation ou «switching») ;
- * avantage : si un matériel est en panne, le réseau continue à fonctionner.

Réseau maillé ou «Mesh»

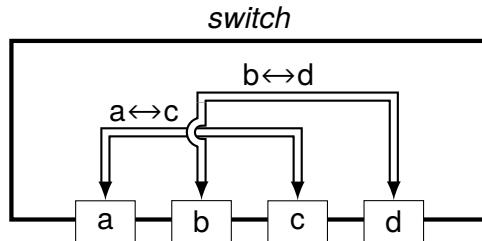


- * chaque matériel possède une ou plusieurs connexions avec les autres matériels ;
- * cette topologie offre une meilleur :
 - ◊ une meilleure résistance, «resilience», en cas de rupture d'un lien de connexion ou de panne d'un matériel ;
 - ◊ une économie de moyens par rapport à une redondance totale (maillage complet) ;
- * les liens peuvent être filaire ou sans fil ou un mélange des deux. Dans le cas du WDS, «*Wireless Distribution System*», un AP, «access point», du maillage sert d'intermédiaire de connexion vers des APs connectés au réseau filaire «*backbone*».



Réalise de la commutation

- ▷ **supprime les collisions de paquets** lors de la **compétition** pour l'accès au réseau ;
- ▷ **réalise plusieurs communications simultanées** ce qui améliore le débit global :

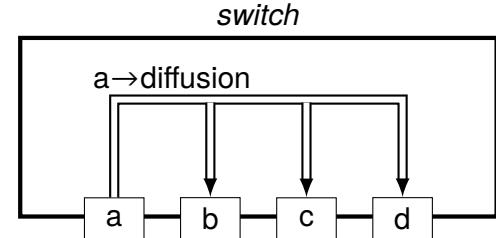


Les machines connectées aux ports «a» et «c» peuvent communiquer en même temps que les machines connectées sur les ports «b» et «d».

⇒ *deux communications simultanées au lieu d'une seule !*

Réalise de la diffusion

Lorsque la machine connectée au port «a» envoie une trame en **diffusion**, le switch diffuse cette trame à tous les ports.



- ▷ le switch représente **un seul domaine de diffusion** : c'est le réseau local !
- ▷ le switch **découpe les domaines de collision** : seules les machines en communication sont concernées, pas de risque de collisions.



Le lien point à point, «point to point»

- * un seul émetteur d'un côté du lien ;
- * un seul récepteur de l'autre côté du lien ;
- * contrôle des échanges MAC «*Medium Access Control*» : «flow control» ou contrôle de flux ;
- * exemples : PPP, tunnels, Ethernet Gigabit, switch full duplex...

Le lien à diffusion, «broadcast link»

- * de multiples matériels accèdent en émission et en réception au support de communication, «*Multiple Access*» ;
- * chaque matériel reçoit une copie du message émis ;
- * contrôle des échanges MAC «*Medium Access Control*» : CSMA/CD «*Carrier sense Multiple access with Collision Detection*», CSMA/CA «...with Collision avoidance» ;
- * exemples : Ethernet 100Mbits/1Gbps, switch, WiFi...

Les différents types de liens sous Linux

```
□ └── xterm └──
root@starfox:~# ip link
2: eth0: <NOARP,BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode
DEFAULT qlen 1000
      link/ether 00:0c:29:89:01:44 brd ff:ff:ff:ff:ff:ff

8: mon_pt_a_pt: <POINTOPOINT,NOARP> mtu 1480 qdisc noop state DOWN mode DEFAULT
      link/sit 10.1.1.1 peer 192.168.1.1
```

- ▷ BROADCAST : lien à diffusion avec support du multicast ;
- ▷ POINTOPOINT : lien point à point sans support du protocol ARP.

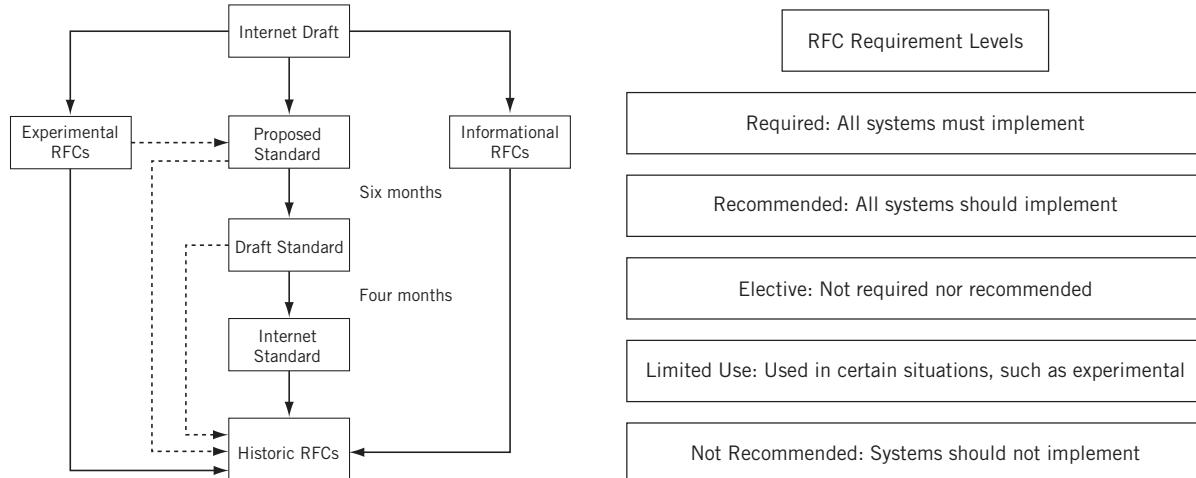


- * **IEEE**, «*Institute of Electrical and Electronics Engineers*» : organisation internationale chargée de superviser le développement et l'adaptation de standards internationaux.
Par exemple dans le cadre des communications sans fil avec l'IEEE 802.11.
- * **ANSI**, «*American National Standards Institute*» : organisation non gouvernementale à but non lucratif contribuant à l'élaboration de standards pour l'industrie en protégeant les intérêts du public.
Par exemple, le code ASCII, American Standard Code for Information Interchange.
- * **EIA**, «*Electronic Industries Association*» : organisation à but non lucratif proche de l'ANSI dédiée à la résolution des problèmes de fabrication de composants électroniques.
La prise du «twisted-pair» appelé RJ-45.
- * **ISO**, «*International Standards Organization*» : le nom vient du grec «isos» qui veut dire «égaux», organisation de standardisation internationale dont les membres appartiennent à des comités de standardisation de différents pays. Elle est basée sur le volontariat (pour les Etats-Unis, c'est l'ANSI qui participe).
*Dans le cadre des réseaux, sa contribution principale est le modèle OSI : «Open Systems Interconnection Reference Model» qui sert de base à l'analyse et la conception des **protocoles** de communication.*
- * **ITU-T**, «*International Telecommunications Union-Telecommunication Standards Sector*» : permettre une infrastructure mondiale non seulement dans les réseaux de données mais également dans la téléphonie, PSTN, «public switched telephone network». Les Nations Unies ont formées un comité le CCITT, «Consultive Committee for International Telegraphy and Telephony» compris dans l'ITU, «International Telecommunications Union».
Tout communication qui traverse les frontières d'un pays doit se conformer aux recommandations de l'ITU-T.
Une technologie comme ATM, «Asynchronous Transfer Mode» est un standard ITU-T.
- * Des **forums** : promouvoir une technologie et sa standardisation.
Exemple le MEF, «Metro Ethernet Forum».
- * Obligations de se conformer aux régulateurs nationaux, comme le **FCC**, «*Federal Communications Commission*» qui surveille, par exemple, l'utilisation des différents fréquences dans le cas de transmission sans fil.



Les «Request for Comment» & l'IETF, «Internet Engineering Task Force»

- IETF : organisme responsable du développement des standards de l'Internet.
 - ◊ son propre système de standardisation des **protocoles** utilisés par les matériels connectés à Internet.
Les protocoles concernés sont plus proches de l'utilisateur (applications) que du matériel.
- Les standards Internet : des spécifications testées et qui doivent être suivies.
 - ◊ la procédure d'élaboration est stricte :
 - * «Internet draft» : document de travail, souvent modifié sans statut particulier et de durée de vie d'au plus 6 mois ;
 - * les développeurs travaillent à partir de ces «drafts» ;
 - * un draft peut être publié sous la forme d'un RFC (juste une appellation, il n'y a plus besoin de retour ou *feedback*).
 - ◊ RFC identifiée par un numéro, librement consultable et, suivant le niveau de *requirements*, **obligatoire**, ou pas.
 - ◊ Tous les RFCs ne sont pas des standards, même ceux définissant des protocoles entiers.
 - ◊ Après un certain temps, la RFC peut arriver à maturité et finir dans les «RFC historiques».



- **InterNIC**, «*Internet Network Information Center*» entre 1992-1998 :
 - ◊ organisme public américain chargé de la gestion centrale des adresses et des noms de domaines Internet et de l'accréditation d'un organisme homologue dans chaque pays, les organismes délégués :
 - * AfriNIC (Afrique),
 - * APNIC (Asie, Pacifique),
 - * ARIN (Amérique du Nord),
 - * LACNIC (Amérique du Sud, îles Caraïbes),
 - * RIPE NCC (Europe, Moyen-Orient)
 - * NIC France ou afnic, NIC Angleterre, etc.
- **ICANN**, «*Internet Corporation for Assigned Names and Numbers*» :
 - ◊ Organisation créée en octobre 1998, pour s'ouvrir à la concurrence
 - ◊ traite les noms de domaine et leur délégation (par exemple VERISIGN Inc. : zone « .com ») ;
 - ◊ exploitation des serveurs de la racine du DNS (ceux qui font autorité) ;
 - ◊ allocation de blocs de numéro IP ;
 - ◊ en France, les prestataires (fournisseurs d'accès) font l'intermédiaire avec l'afNIC
- **IANA**, «*Internet Assigned Numbers Authority*» :
 - ◊ tient l'annuaire : adresses IP & numéros de protocoles ;
 - ◊ adresses IP et numéros d'AS : déléguées aux RIR régionaux, «Regional Internet Registries» ;
 - ◊ numéros de protocoles et de ports (entre 1 et 1023) ;
 - ◊ déléguées aux LIRs, «Local Internet Registry» (eg. FAI).
- **Les «Registrar» :**
 - ◊ Un registrar (bureau d'enregistrement) est une société ou une association permettant le dépôt de noms de domaine internet, dans les TLD, «Top Level Domain», où il n'y a pas de vente directe.
 - ◊ Il faut payer un certain montant pour acquérir et protéger un nom de domaine.
- **GIP Renater** (Groupement d'Intérêt Public) :
 - ◊ Réseau de la recherche en France, «Réseau National de télécommunications pour la Technologie l'Enseignement et la Recherche»



La conception

Contraintes du protocole IP «Internet Protocol» RFC 791 :

- * utiliser la topologie réseau point à point (pour permettre des grandes distances c'est obligatoire) ;
- * la panne d'un équipement du sous-réseau ne doit pas entraîner une rupture du réseau ;
- * privilégier la **disponibilité** du réseau : il doit servir au maximum.

Le but est que les transactions continuent :

- ▷ du moment que l'ordinateur source et l'ordinateur destination fonctionnent ;
- ▷ même si certains routeurs ou certaines lignes de transmission tombent en panne (origine militaire de la création d'Internet par le DoD états-unisens).

Les moyens

- définir une **architecture très souple** pour pouvoir mettre en œuvre des applications très diverses comme le transfert de fichiers ou la transmission de la parole en temps réel (TCP et UDP) ;
- faciliter le **routage** : construire une méthode simple et rapide (opérations binaires par exemple) ;
- permettre le **regroupement de machines** pour les gérer ensemble (regroupement en réseau) ;
- faciliter le travail de l'administrateur (*sisi...*).



Adressage pour le protocole IP (IPv4)

Chaque ordinateur et chaque routeur du réseau Internet possède une adresse IP.

L'adresse IP est une **adresse binaire** composée de deux parties <id. réseau><id. machine> :

- * un **identifiant de réseau** ;
- * un **identifiant machine** pour la distinguer dans le réseau.

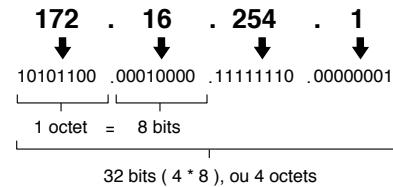
Chaque adresse IP doit être unique pour permettre de la localiser sur la planète.

- * Il existe **différentes répartitions** des 32 bits entre identifiant réseau et identifiant machine :
 - ◊ ces **différentes répartitions** définissent un ensemble de **classes de réseaux** ;
 - ◊ ces classes **ne sont plus utilisées** en CIDR, où on indique uniquement le nombre de bits de la partie réseau ;

Propriétés

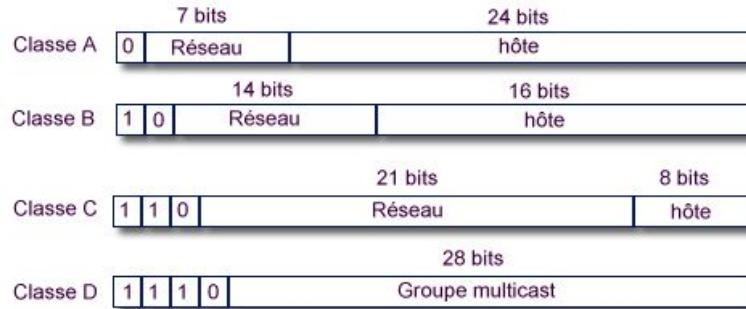
- ▷ Codée sur **32 bits**.
- ▷ Représentée par **commodité** en «décimale pointée» : 4 entiers variant entre 0 et 255 séparés par des points
exemple : 164.81.1.4

Une adresse IPv4 (notation décimale à point)



- ▷ un **organisme officiel**, le NIC, «Network Information Center», est seul habilité à délivrer des numéros d'identification des réseaux.
- ▷ il y a, **en général**, une **seule adresse IP** par interface réseau.
Dans le cas d'un routeur interconnectant 2 réseaux différents, il possède une adresse IP pour chacune de ses interfaces connectées à un réseau.



Les classes de réseaux définies par un « préfixe »

Les adresses de classe A sont peu utilisées. Exemple : 3.0.0.0/8, AS80, GE-CRD - General Electric Company.

Aux niveaux des adresses IP

	32-bit Address Starts with:	Number of Addresses:	% of Address Space
Class A	0 (0-127)	$2^{31} = 2,147,483,648$	50
Class B	10 (128-191)	$2^{30} = 1,073,741,824$	25
Class C	110 (192-223)	$2^{29} = 536,870,912$	12.5
Class D	1110 (224-239)	$2^{28} = 268,435,456$	6.25
Class E	1111 (240-255)	$2^{28} = 268,435,456$	6.25
	First byte	Second byte	Third byte
			Fourth byte



Ces **adresses** permettent :

- * des envois de messages **multi-destinataires** ;
- * désigner la **machine courante** ;
- * désigner le **réseau courant**.

Tout à zéro	L'ordinateur lui-même
Tout à zéro	Id. de machine Un ordinateur sur le réseau lui-même
Tout à 1	Diffusion limitée au réseau lui-même
Id. de réseau	Tout à 1 Diffusion dirigée vers ce réseau
127	Nombre quelconque Boucle

L'**adresse de «boucle»** (127.X.Y.Z) permet d'effectuer :

- ◊ des communications inter-programme sur la même machine
- ◊ des tests de logiciels réseaux. *Dans ces cas là, les paquets ne sont pas réellement émis sur le réseau.*

D'autres **adresses particulières** :

- ◊ 0.0.0.0 est utilisé par une machine pour connaître sa propre adresse IP lors d'un processus d'amorçage (BOOTP).
Elle devra se procurer une adresse IP par l'intermédiaire d'une autre machine.
- ◊ 255.255.255.255 est une adresse de diffusion locale car elle désigne toutes les machines du réseau auquel appartient l'ordinateur qui utilise cette adresse \Rightarrow **pas besoin de connaissance du réseau**.

Réseaux privés, RFC1918

Les adresses pour **réseau privé** ou **intranet** (sans **accès direct** à l'extérieur) :

- * 10.0.0.0/8 : de 10.0.0.0 à 10.255.255.255 \Rightarrow **classe A**
- * 172.16.0.0/12 : 172.16.0.0 à 172.31.255.255 \Rightarrow **pas de classe**
- * 192.168.0.0/16 : 192.168.0.0 à 192.168.255.255 \Rightarrow **classe B**

Ces réseaux ne sont pas «routables» !



Transmission physique des datagrammes IP

La **couche liaison de données** est chargée de :

- ▷ la mise en correspondance des @IP avec les @MAC des interfaces physiques.
- ▷ l'**encapsulation** des datagrammes IP afin qu'ils puissent être transmis sur un support physique particulier.

Lorsque le protocole IP doit envoyer un datagramme à un équipement relié à un réseau à diffusion, la couche liaison de donnée doit construire une trame ethernet avec l'@MAC du destinataire.

Correspondance entre adresses physiques, @MAC, et adresses IP, @IP

Le **protocole ARP**, «*Address Resolution Protocol*» fournit une **correspondance dynamique** entre une adresse IP connue et l'adresse matérielle correspondante.

Fonctionnement :

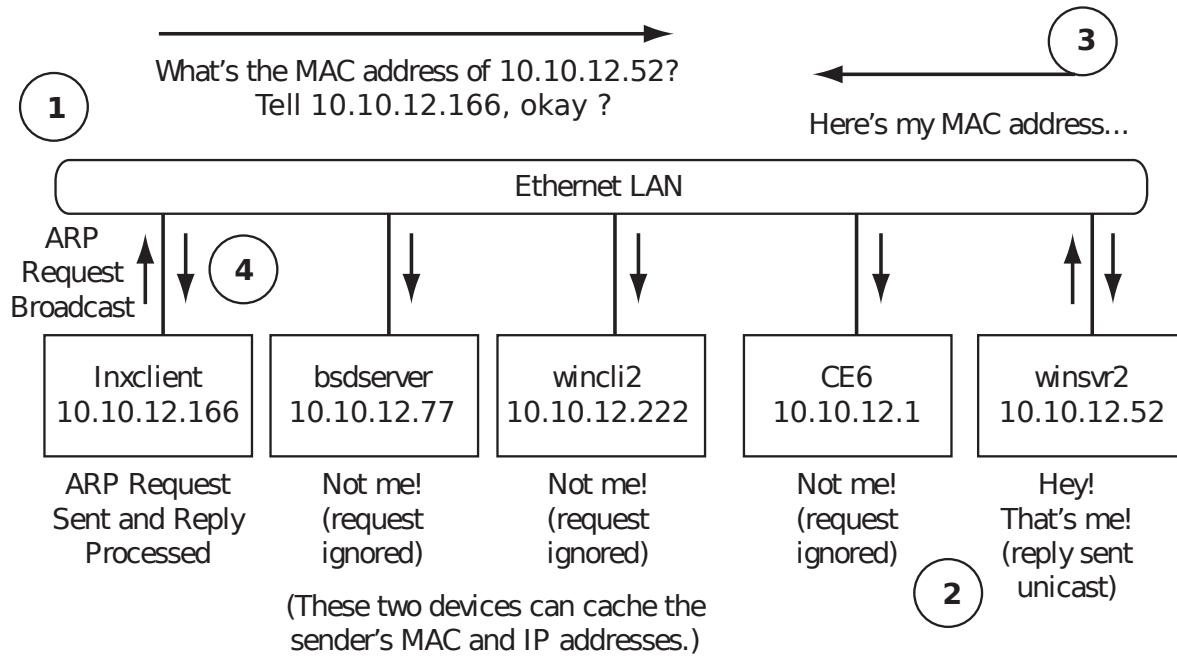
- ▷ ARP dispose d'une **mémoire cache** : lors de la demande de l'@MAC associée à une @IP, il consulte sa **mémoire cache ARP** pour voir si l'@IP distante y est mise en correspondance avec @MAC.
 - ◊ Si c'est le cas le datagramme IP est émis immédiatement, enveloppé dans une **trame Ethernet** envoyée à l'adresse physique destination (@MAC).
 - ◊ Sinon la couche liaison de données construit une **requête ARP**.
- ▷ ARP utilise le principe de «*diffusion*» du réseau local : la requête ARP est transmise en «*broadcast*».
- ▷ Lorsqu'un **message ARP** est reçu, la couche liaison de donnée fait :
 - ◊ une **première vérification** pour voir si c'est une **requête ARP** et que l'@IP demandée correspond à l'@IP locale alors une **réponse ARP** est renvoyée à destination de l'@MAC de l'expéditeur.
La machine répond parce qu'elle est concernée : c'est son adresse qui est demandée.
 - ◊ une **seconde vérification** pour vérifier si l'adresse IP de l'émetteur se trouve déjà dans la **mémoire cache ARP locale** sinon il y a **mise à jour** de la mémoire cache avec cette nouvelle association.
Elle apprend l'association, comme dans le cas d'un «gratuitous ARP», c-à-d une réponse ARP non sollicitée envoyée en broadcast.



Illustration d'ARP

Comment échanger réellement sur un réseau local à diffusion ?

- * Les machines ont chacune une carte réseau ;
- * Chaque carte a une **adresse MAC unique** donnée par le constructeur ;
- * Chaque machine dispose d'une **adresse IP** donnée par l'administrateur du réseau.



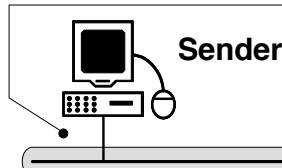
La machine 10.10.10.166 demande l'@MAC de la machine 10.10.12.52.



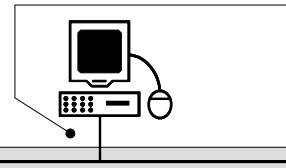
Illustration d'ARP : les échanges

20

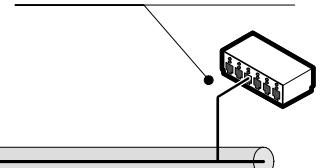
IP: 192.200.96.21
MAC: 01-23-45-67-89-AB



IP: 192.200.96.22
MAC: 00-01-03-1D-CC-F7



IP: 192.200.96.20
MAC: 49-BD-2F-54-1A-0F



ARP Request: to FF-FF-FF-FF-FF-FF

Sender MAC: 01-23-45-67-89-AB
Sender IP: 192.200.96.21
Target IP: **192.200.96.23**

ARP Reply

Sender MAC: **A3-B0-21-A1-60-35**
Sender IP: 192.200.96.23
Target MAC: 01-23-45-67-89-AB
Target IP: 192.200.96.21

IP: 192.200.96.23
MAC: A3-B0-21-A1-60-35

- ◊ La requête de la machine «192.200.96.21» demande l'@MAC de la machine 192.200.96.23;
- ◊ La réponse de la machine 192.200.96.23 donne la réponse @MAC A3:B0:21:A1:60:35.



Le protocole RARP «Reverse Address Resolution Protocol»

Il réalise l'opération inverse :

- une machine sans adresse IP connue peut envoyer une requête RARP pour demander son adresse IP.
- une machine particulière (un serveur gérant le réseau) lui répond et lui affecte son adresse IP.
- cette machine dispose d'une table de correspondance : (adresse physique, adresse IP).

Le protocole RARP est utile pour amorcer une station sans disque, un Terminal X-Window, ou une imprimante.

Pour consulter la table ARP**Sous Linux**

```
pef@solaris:~$ ip neighbor
192.168.42.254 dev eth0 lladdr 00:07:cb:cc:d4:e5 STALE
192.168.42.122 dev eth0 lladdr 64:b9:e8:d2:23:ba REACHABLE
```

Sous Windows

```
C:\> C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\PeF>arp -a
Interface : 192.168.144.128 --- 0x20002
    Adresse Internet      Adresse physique      Type
    192.168.144.254        00-50-56-e5-45-36    dynamique
C:\Documents and Settings\PeF>
```



Pour envoyer un datagramme d'une source vers une destination, il faut savoir **localiser** la machine destination.

Deux possibilités :

- ▷ les deux machines **font partie** du même réseau local : on parle de **routage direct** (sur Ethernet, on utilisera le protocole ARP et l'envoi direct sur le réseau à diffusion) ;
- ▷ les deux machines **ne font pas partie** du même réseau local : on parle de **routage indirect**.

On doit passer par un intermédiaire qui permet de sortir du réseau local pour aller vers l'extérieur : le **routeur** (ou appelé «passerelle» ou *gateway*).

Pour faire du routage direct ou indirect pour un datagramme

- ▷ savoir si les deux machines font **partie du même réseau** local ;
- ▷ connaître l'@IP d'un **routeur de sortie** ;
- ▷ remettre **directement** le datagramme à la machine destination si elles sont dans le même réseau locale ;
- ▷ remettre le datagramme **au routeur sinon**.

Comment savoir si Source et Destination sont dans le même réseau ?

Il faut **comparer** l'<id. réseau> des deux adresses : si c'est **la même** \Rightarrow les deux sont dans le **même réseau local**.

Comment remettre le datagramme au routeur

Il faut utiliser le mécanisme **d'encapsulation** d'un datagramme dans une trame :

- ▷ la **trame** sert à remettre des données d'une machine connectée à un réseau local à une autre machine connectée au même réseau local ;
- ▷ la **trame** possède une @MAC de destination **indépendante** de l'@IP : il est possible d'envoyer la trame à une machine dont l'@IP **ne correspond pas** à son @IP !

Par exemple : on peut envoyer une datagramme à destination de l'extérieur du réseau local à l'@MAC du routeur.

Attention : les attaques MiTM, «Man-in-the-Middle», opèrent sur l'association @MAC \Leftrightarrow @IP du routeur !



En utilisant la notion de classe

Il suffisait de comparer les <id. réseau> des deux @IP suivant la **répartition donnée par la classe**, mais...

La notion de sous-réseau ou *Subnetting*

But

Partition d'un réseau en **differents sous-réseaux**.

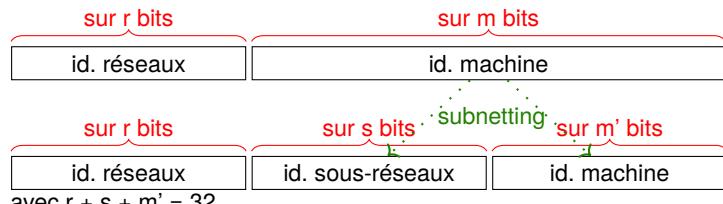
Avantages

Utiliser le réseau global fourni par le NIC à l'entité pour disposer de **réseaux autonomes** au sein de cette même entité

Exemple : l'université de Limoges avec les machines du site de Condorcet, du campus de Vanteaux, de La Borie...

Mise en œuvre

Définition de sous-réseaux en découplant <id. machine> en deux parties :



Le découpage par multiple de 8bits facilite le travail des routeurs, mais n'est pas obligatoire.

Une machine connectée à un sous-réseau doit connaître :

- son @IP ;
- le nombre de bits attribués à l'**<id. réseau + sous-réseau>** ;

Remarque : l'ensemble des sous-réseaux d'un même réseau est vu de l'extérieur comme un **unique réseau** (routage, courrier...).

Masque de sous-réseau (subnet mask)

c'est un mot de 32 bits contenant :

- * des bits à 1 à la place de l'identifiant réseau/sous-réseau,
- * des bits à zéro en lieu et place de l'identifiant machine.

Ainsi, 255.255.255.0 indique que les premiers 24bits désignent le sous-réseau.

Nouvelle notation CIDR: @IP / nombre de bits pour identifiant réseau

Exemple : 164.81.55.0/24 pour désigner un réseau ou bien 164.81.55.12/24 pour une machine.



Localisation de la machine destinataire

- Chaque ordinateur connecté au réseau Internet dispose :
- ▷ d'une **@IP** sur 32 bits ;
 - ▷ d'un **masque de sous-réseau** : répartition des 32 bits d'**@IP** entre <id. réseau><id. machine>.

Algorithme de choix routage direct/indirect

Envoi d'un datagramme :

- ▷ de S: 164.81.128.34 et masque 255.255.192.0 ;
- ▷ à destination de D: 193.50.185.12.

Le masque de réseau de la machine D n'est pas connu de S.

- * combinaison avec l'opérateur binaire «ET», $\&$, du **masque de sous-réseau de S** et des **@IP** de S et D ;

ET	0	1
0	0	0
1	0	1



- * comparaison des **@IP réseau d'appartenance** des adresses de S et D :

164.81.128.0 [10100100|01010001|10000000|00000000] 193.50.128.0 [11000001|00110010|10000000|00000000]

◊ Si égalité alors la destination D est **dans le même réseau local** : \Rightarrow **routage direct**

Si ce n'est pas vrai alors il y a un problème de choix de l'adresse réseau : l'administrateur s'est planté !

◊ Si différence alors D n'est pas dans le réseau local \Rightarrow **routage indirect**:

\Rightarrow envoi par l'**intermédiaire d'un routeur de sortie** du réseau local.

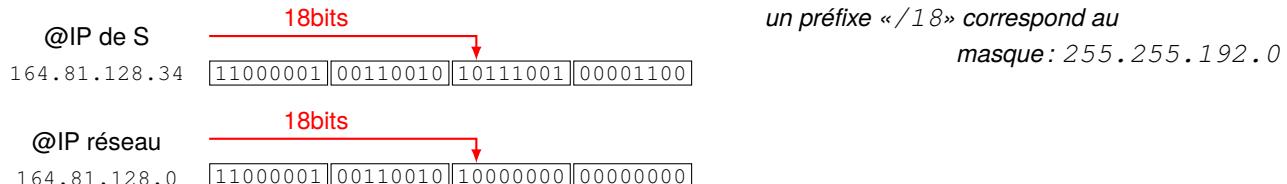
Sur l'exemple, 164.81.128.0 \neq 193.50.128.0 \Rightarrow routage indirect.



Nouvelle version : la notion de préfixe CIDR : *Classless Inter-Domain Routing*)

Plus de notion de *classe* : on indique la répartition <id. réseau><id. machine> directement à l'aide d'un «/n».

Exemple : 164.81.128.34/18 où /18 indique l'affectation des 18 premiers bits de l'adresse pour indiquer le réseau.



Si :

- ▷ $(@\text{IP réseau de S}) = (@\text{IP réseau de D suivant le préfixe de S}) \Rightarrow \text{routage direct.}$
- ▷ $(@\text{IP réseau de S}) \neq (@\text{IP réseau de D suivant le préfixe de S}) \Rightarrow \text{routage indirect.}$

Mise en œuvre du routage direct

Utilisation du réseau à diffusion et donc de la fameuse @MAC et du protocole ARP...

Mise en œuvre du routage indirect

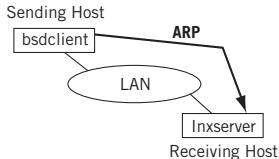
- ▷ il faut connaître l'adresse d'un **routeur de sortie** ;
- ▷ il faut remettre le datagramme à ce **routeur** en **routage direct**.



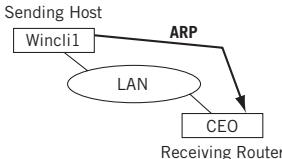
Routage & ARP : Quatre cas de figure

26

Case 1: Find the address of a host on the same subnet as the source.



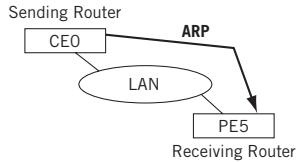
Case 2: Find the address of a router on the same subnet as the source.



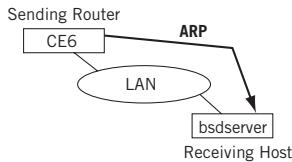
1. **hôte vers hôte**: l'émetteur est un hôte qui veut envoyer un paquet à un autre hôte dans le même réseau local.

Dans ce cas, l'@IP de destination est connue et l'@MAC de destination doit être trouvée.

Case 3: Find the address of a router on the same subnet as the source router.



Case 4: Find the address of a host on the same subnet as the source router.



2. **hôte vers routeur**: l'émetteur est un hôte et veut envoyer un paquet à un autre hôte **n'appartenant pas** au réseau local.

Il doit consulter la table de routage, «*forwarding table*» ou «*Forwarding Information Base*», pour trouver l'@IP du routeur.

L'@IP du routeur est connue et l'@MAC du routeur doit être trouvée.

3. **routeur vers routeur**: l'émetteur est un routeur et veut «faire suivre» un paquet à un autre routeur connecté dans le même réseau local.

La table de routage est utilisée pour trouver l'@IP du routeur.

L'@IP du routeur est connue et l'@MAC du routeur destination doit être trouvée.

4. **routeur vers hôte**: l'émetteur est un routeur et veut «faire suivre», «*forward*», un paquet vers un hôte dans le même réseau local.

L'@IP de l'hôte est connue, elle est contenue dans le paquet et l'@MAC de l'hôte doit être trouvée.

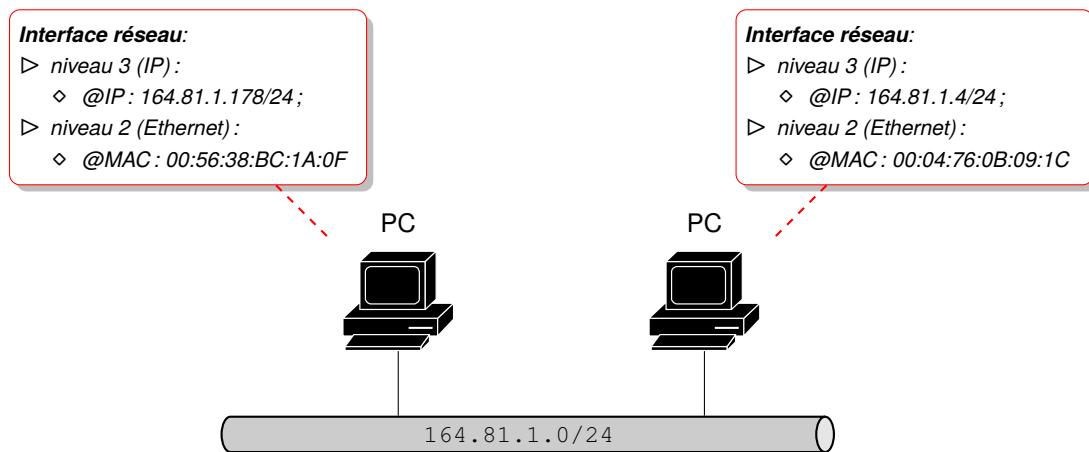
Attention

«...dans le même réseau local» ! \Rightarrow ARP ne sert quand dans un réseau local.



Chaque machine est identifiée par :

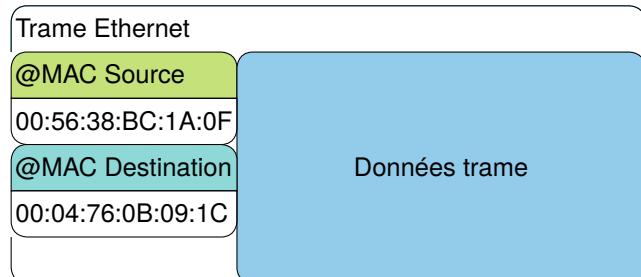
- * une adresse de niveau 2 (@MAC) ;
- * une adresse de niveau 3 (@IP) ;
- * un réseau d'appartenance connu :
 - ◊ à l'aide du **préfixe** «/n» indiquant n le nombre de bits de l'identifiant réseau»
 - ◊ ou à l'aide du **masque réseau**, *netmask*, adresse où chaque bit de l'identifiant réseau est à 1, les autres sont à 0.



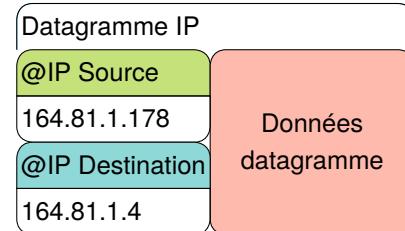
Routage direct illustré

Pour être échangé, les datagrammes IP sont encapsulés dans des trames Ethernet

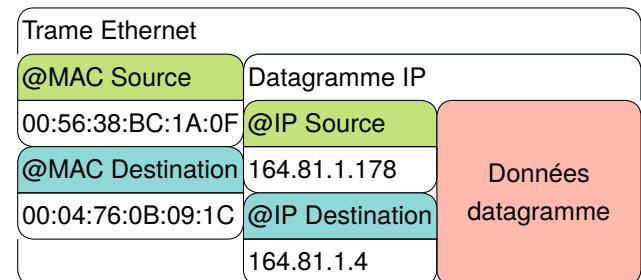
- * la **trame** contient :
 - ◊ une @MAC source ;
 - ◊ une @MAC de destination ;



- * le **datagramme** contient :
 - ◊ une @IP source ;
 - ◊ une @IP destination et des **données** ;

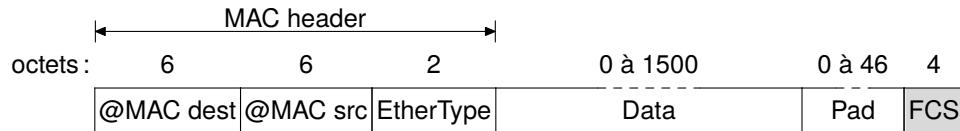


- * la **trame encapsule** le datagramme IP :



Le réseau à diffusion utilise une technologie différente du réseau IP :

- * il dispose de **ses propres adresses** : @MAC ;
- * il utilise des messages sous un **format particulier** : la trame Ethernet ou IEEE 802.3 :



Où :

- ◊ **@MAC destination puis @MAC source** : ⇒ le récepteur détermine immédiatement s'il est destinataire ;
- ◊ **EtherType** (valeur > 1536) :
 - * 0x0800 pour un datagramme IPv4, 0x08DD pour un datagramme IPv6, 0x806 pour un message ARP, 0x888E pour du 802.1x, 0x8100 pour du VLAN...;
 - * **mais, si la valeur < 1500 ⇒ trame 802.3 avec LLC**, «Logical Link Control» ;
- ◊ **Data** : les données de la trame, complétées éventuellement par du bourrage (la longueur de ce champ est comprise entre 0 et 1500 octets) ;
- ◊ **PAD** ou bourrage : octets de bourrage sans signification, insérés **si la longueur du champ Data est insuffisante** (inférieure à 46 octets) ;
- ◊ **FCS**, *frame check sequence* ou Checksum : champ pour la **détection d'erreurs** (rarement transmis aux programmes).

Attention

- * Si le FCS est incorrect alors la trame n'est **pas transmise** par la carte réseau au système d'exploitation.
- * La trame a une taille de 64 à 1518 octets ($6 + 6 + 2 + 46 + 4 = 64$ à $6 + 6 + 2 + 1500 + 4 = 1518$).



Encapsulation en technologie Ethernet vu dans Wireshark

30

Un datagramme IP est contenu dans une trame, par exemple une trame Ethernet :

The screenshot shows a Wireshark capture of an Ethernet frame (Frame 7) with the following details:

- Ethernet II:** Src: AppleCom_d6:d3:b9 (00:19:e3:d6:d3:b9), Dst: HonHaiPr_d7:a9:d2 (00:16:ce:d7:a9:d2)
- Type: IP (0x0800)
- Frame check sequence: 0xa2275cc2 [correct]
- Internet Protocol:** Src: 192.168.1.51 (192.168.1.51), Dst: 88.87.11.119 (88.87.11.119)
- User Datagram Protocol:** Src Port: 64617 (64617), Dst Port: 51534 (51534)
- Data (71 bytes)

The data bytes are displayed in three panes:

- Hex View:** Shows the raw bytes of the frame, including the Ethernet header (00:16:ce:d7:a9:d2, 00:19:e3:d6:d3:b9), the IP header (40 bytes), and the UDP header (12 bytes). The data payload starts at byte 74.
- ASCII View:** Displays the ASCII representation of the captured data, showing characters and control codes.
- Text View:** Provides a detailed breakdown of the captured data, identifying the Ethernet header, IP header, and UDP header.

Le datagramme IP encapsule lui-même un datagramme UDP...



Routage direct illustré

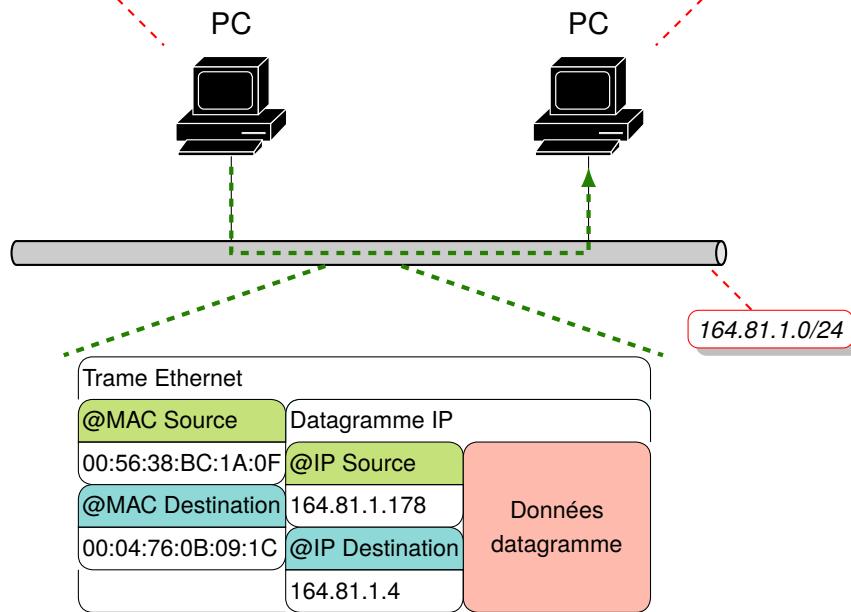
31

Interface réseau:

- ▷ niveau 3 (IP):
 - ◊ @IP: 164.81.1.178/24;
- ▷ niveau 2 (Ethernet):
 - ◊ @MAC : 00:56:38:BC:1A:0F

Interface réseau:

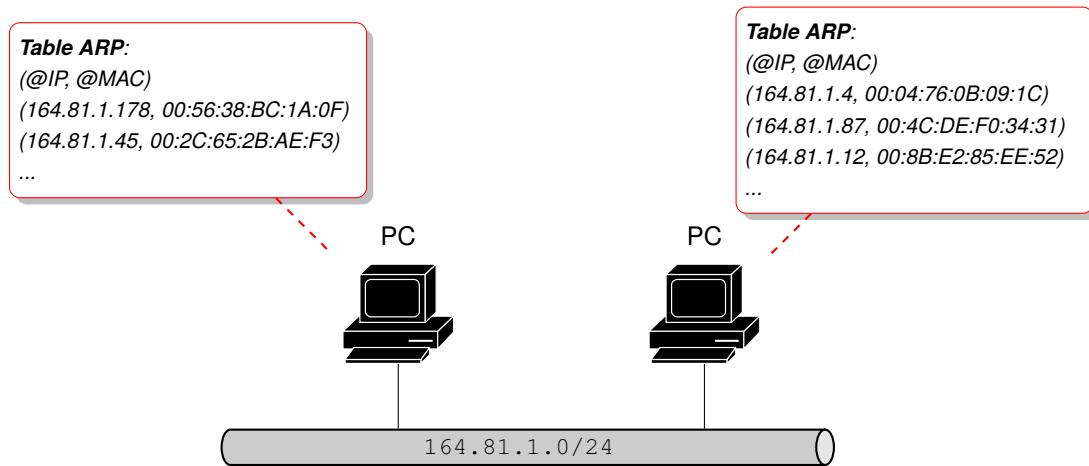
- ▷ niveau 3 (IP):
 - ◊ @IP: 164.81.1.4/24;
- ▷ niveau 2 (Ethernet):
 - ◊ @MAC : 00:04:76:0B:09:1C



Routage direct illustré & ARP

Pour connaître la correspondance entre adresse IP et adresse MAC :

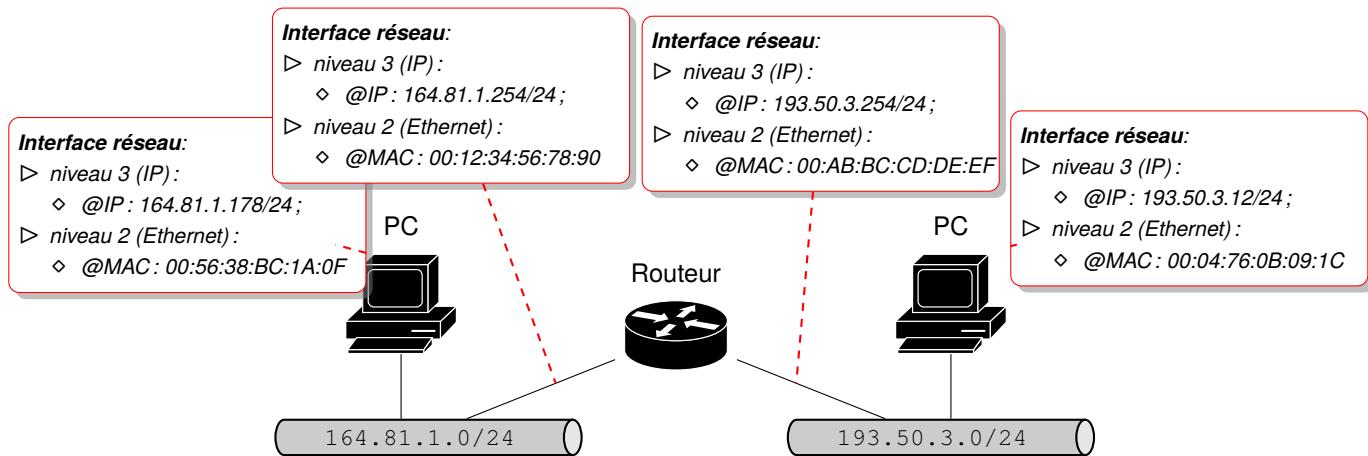
- ▷ mise en oeuvre du protocole ARP (Address Resolution Protocol) ;
- ▷ construction d'une table de correspondance entre @ IP et MAC sur chaque machine (cache ARP).



La **modification malveillante** de cette table est possible : «ARP Spoofing» (usurpation d'identité), «ARP Cache Poisoning» (insertion d'association erronée).



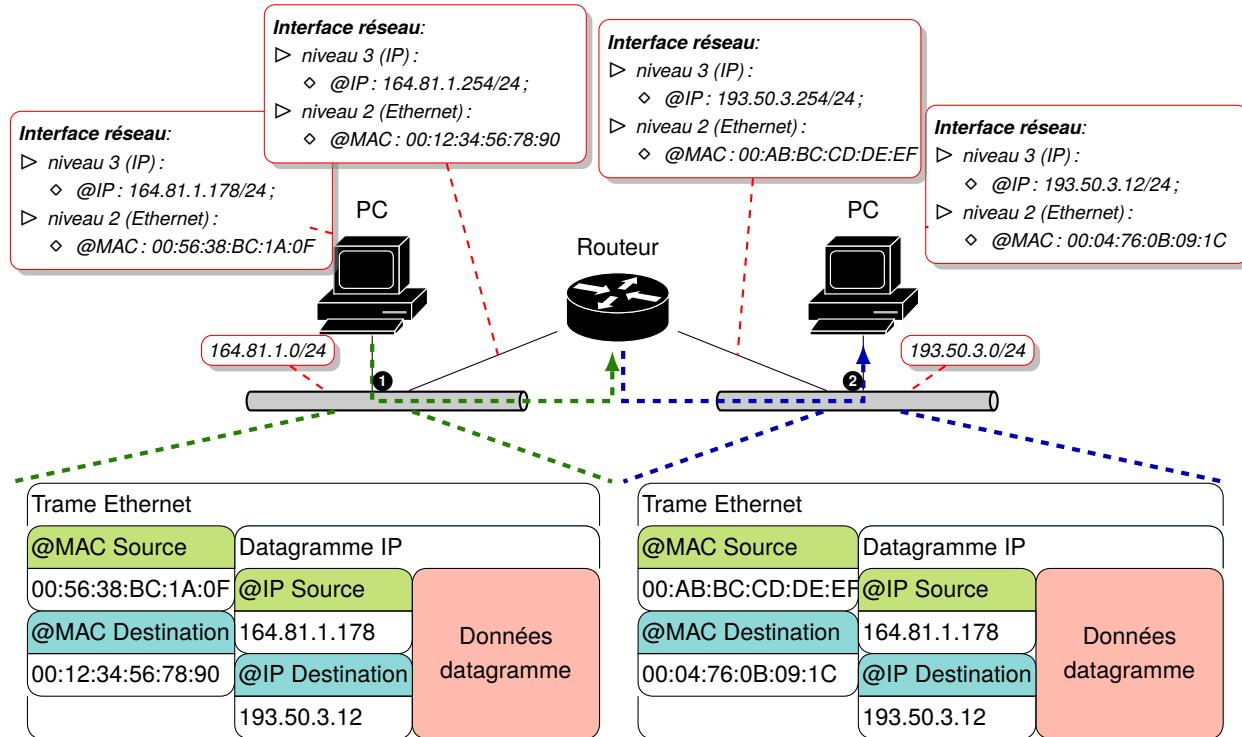
Le paquet de la machine 164.81.1.178 est routé par l'intermédiaire du routeur vers la machine 193.50.3.12.



Routage indirect illustré

Le datagramme IP est **encapsulé**:

- ▷ par la machine 164.81.1.178, dans une trame à **destination du routeur**;
- ▷ puis, par le routeur, dans une nouvelle trame à destination de la machine 193.50.3.12.



L'encapsulation permet la redirection vers le routeur sans modifier les @IP du datagramme.

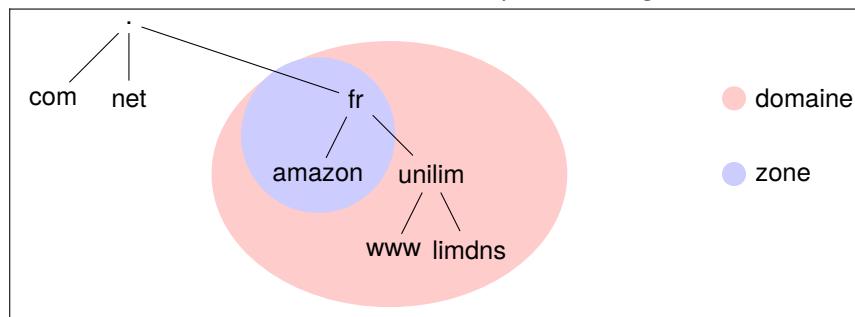


Le système DNS est entièrement distribué au niveau planétaire en utilisant la **délégation de domaine**.

À tout domaine est associé une **responsabilité administrative**.

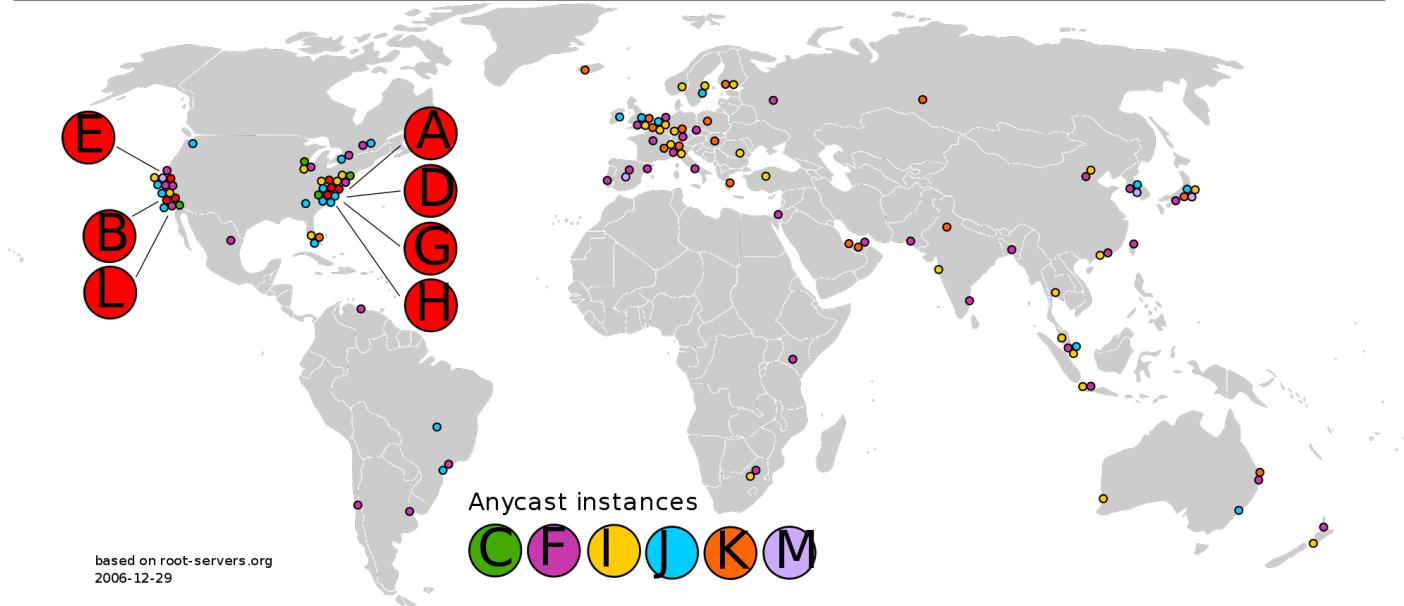
Une organisation responsable d'un domaine peut :

- ▷ **découper** le domaine en sous-domaines ;
- ▷ **déléguer** les sous-domaines à d'autres organisations :
 - ◊ qui deviennent responsables du (des) sous-domaine(s) qui leurs sont délégué(s) peuvent, à leur tour, déléguer des sous-domaines des sous-domaines qu'elles gèrent.
 - ◊ *Le domaine parent contient alors seulement un pointeur vers le sous-domaine délégué;*
- * Les serveurs de nom enregistrent les données propres à une partie de l'espace nom de domaine dans une **zone**.
- * le serveur de nom à **autorité administrative** sur cette zone ;
- * un serveur de nom peut avoir **autorité** sur plusieurs zones ;
- * une **zone** contient les informations d'un domaine **sauf** celles qui sont **déléguées** :



Les 13 serveurs racines

36



Le terme «anycast» permet d'offrir des services DNS de proximité à l'aide de cette capacité offerte par IPv6.

On peut remarquer que ce service de proximité permet même de délocaliser géographiquement les serveurs et améliore la disponibilité et la sécurité du système DNS.



Exemple : une requête DNS vers «google.com»

```

xterm
bonnefoi@msi:~$ dig +trace @164.81.1.5 www.google.com
; <>> DIG 9.7.3 <>> +trace @164.81.1.5 www.google.com
; (1 server found)
;; global options: +cmd
.          IN NS j.root-servers.net.
.          IN NS g.root-servers.net.
.          IN NS l.root-servers.net.
.          IN NS k.root-servers.net.
.          IN NS c.root-servers.net.
.          IN NS f.root-servers.net.
.          IN NS e.root-servers.net.
.          IN NS m.root-servers.net.
.          IN NS b.root-servers.net.
.          IN NS d.root-servers.net.
.          IN NS h.root-servers.net.
.          IN NS a.root-servers.net.
.          IN NS i.root-servers.net.
;; Received 272 bytes from 164.81.1.5#53(164.81.1.5) in 1 ms
com.        172800  IN NS a.gtld-servers.net.
com.        172800  IN NS b.gtld-servers.net.
com.        172800  IN NS c.gtld-servers.net.
com.        172800  IN NS d.gtld-servers.net.
com.        172800  IN NS e.gtld-servers.net.
com.        172800  IN NS f.gtld-servers.net.
com.        172800  IN NS g.gtld-servers.net.
com.        172800  IN NS h.gtld-servers.net.
com.        172800  IN NS i.gtld-servers.net.
com.        172800  IN NS j.gtld-servers.net.
com.        172800  IN NS k.gtld-servers.net.
com.        172800  IN NS l.gtld-servers.net.
com.        172800  IN NS m.gtld-servers.net.
;; Received 492 bytes from 193.0.14.129#53(k.root-servers.net) in 31 ms
google.com. 172800  IN NS ns2.google.com.
google.com. 172800  IN NS ns1.google.com.
google.com. 172800  IN NS ns3.google.com.
google.com. 172800  IN NS ns4.google.com.
;; Received 168 bytes from 192.12.94.30#53(e.gtld-servers.net) in 34 ms
www.google.com. 604800  IN CNAME www.l.google.com.
www.l.google.com. 300  IN A 209.85.227.147
www.l.google.com. 300  IN A 209.85.227.106
www.l.google.com. 300  IN A 209.85.227.103
www.l.google.com. 300  IN A 209.85.227.99
www.l.google.com. 300  IN A 209.85.227.105
www.l.google.com. 300  IN A 209.85.227.104
;; Received 148 bytes from 216.239.32.10#53(ns1.google.com) in 26 ms

```

- * la configuration de la machine msi.unilim.fr:

```

xterm
bonnefoi@msi:~$ more /etc/resolv.conf
search msi.unilim.fr unilim.fr
nameserver 164.81.60.1
nameserver 164.81.1.4
nameserver 164.81.1.5

```

- * la requête fait appel au serveur DNS 164.81.1.5 connu de la machine msi.unilim.fr, mais demandé **explicitement** dans l'utilisation de l'outil dig.
- * le serveur, *resolver*, 164.81.1.5 réalise une requête à différents «root servers»
- * il obtient une réponse de la part du serveur racine «k.root-servers.net» qui lui donne la liste des serveurs «GTLD», «Generic Top Level Domain», chargés du domaine «.com»;
- * le serveur «e.gtld-servers.net» lui renvoie la liste des serveurs DNS en charge du domaine google.com;
- * le serveur DNS «ns1.google.com» renvoie une liste d'adresse correspondant à la machine google.com.

Les serveurs racines présents dans le système d'exploitation

```

; Cache file:
. IN NS A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET. IN A 198.41.0.4
. IN NS B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET. IN A 128.9.0.107
. IN NS C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET. IN A 192.33.4.12
. IN NS D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET. IN A 128.8.10.90
. IN NS E.ROOT-SERVERS.NET.

```



Le serveur de courrier

MX = Mail eXchanger Permet l'adressage email sur la base du nom de domaine plutôt que sur l'adresse du (des) serveur(s) de mail :

- ◊ bonnefoi@unilim.fr plutot que bonnefoi@msi.unilim.fr ;
- ◊ permet à l'émetteur d'ignorer quelle est la machine serveur de mail ;
- ◊ permet le déplacement du gestionnaire de mail vers une autre machine ;
- ◊ permet la gestion de plusieurs serveurs de mail avec priorité dans l'ordre de consultation des serveurs

L'enregistrement MX est utilisé par les MTA, «*Mail Transfer Agent*», en tenant compte des priorités :

Exemple pour l'Université de Limoges :

```
□ — xterm —
bonnefoi@msi:~$ dig +short mx unilim.fr
50 mail.unilim.fr.
```

le serveur d'envoi de courrier de l'Université

Exemple pour Google :

```
□ — xterm —
bonnefoi@msi:~$ dig +short mx google.com
30 alt2.aspmx.l.google.com.
10 aspmx.l.google.com.
20 alt1.aspmx.l.google.com.
50 alt4.aspmx.l.google.com.
40 alt3.aspmx.l.google.com.
```

Le MTA utilise le protocole **SMTP**, «*Simple Mail Transfer Protocol*», en tant que client.



Une requête plus concise

```
└── xterm ━━
bonnefoi@msi:~$ dig +noall +answer web.unilim.fr
web.unilim.fr.      83023 IN A 164.81.1.61
```

Pour IPv6 avec des champs adresses 4 fois plus grand (AAAA)

```
└── xterm ━━
bonnefoi@msi:~$ dig +short AAAA www.renater.fr
2001:660:3001:4002::10
```

Interrogation du SOA

```
└── xterm ━━
bonnefoi@msi:~$ dig soa unilim.fr
; <>> DiG 9.7.3 <>> soa unilim.fr
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 36568
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 6
;; QUESTION SECTION:
unilim.fr.      IN  SOA
;; ANSWER SECTION:
unilim.fr.    86400   IN  SOA   limdns.unilim.fr. postmaster.unilim.fr. 2011091501 28800 7200 3600000 86400
;; AUTHORITY SECTION:
unilim.fr.    78813   IN  NS    cnudns.cines.fr.
unilim.fr.    78813   IN  NS    limdns2.unilim.fr.
unilim.fr.    78813   IN  NS    limdns.unilim.fr.
;; ADDITIONAL SECTION:
cnudns.cines.fr. 2583   IN  A    193.48.169.40
cnudns.cines.fr. 2583   IN  AAAA  2001:660:6301:301::2:1
limdns.unilim.fr. 73317   IN  A    164.81.1.4
limdns.unilim.fr. 85383   IN  AAAA  2001:660:6201:1::4
limdns2.unilim.fr. 73317   IN  A    164.81.1.5
limdns2.unilim.fr. 85383   IN  AAAA  2001:660:6201:1::5

;; Query time: 18 msec
;; SERVER: 164.81.60.1#53(164.81.60.1)
;; WHEN: Thu Sep 15 13:25:13 2011
;; MSG SIZE rcvd: 276
```



Organisation en série de couches

But réduire la complexité de conception.

Les réseaux sont organisés en série de couches ou niveaux, chacune étant construite sur la précédente.

Rôle d'une couche offrir certains services aux couches plus hautes, en leur masquant l'implémentation de ces services.

Relation entre couches sur différentes machines

La couche n d'une machine gère **la conversation** avec la couche n d'une autre machine.

Notion de «protocole»

Les **règles** et **conventions** utilisées pour cette conversation sont connues sous le nom de «protocole de la couche n » :

- ▷ un **protocole** est un accord entre les parties sur la **façon** de communiquer ;
- ▷ toute **Violation** du protocole rend la communication extrêmement difficile voire **impossible**.

Notion de «processus pairs»

Les couches correspondantes sur différentes machines sont appelés **processus pairs**, *peer*.

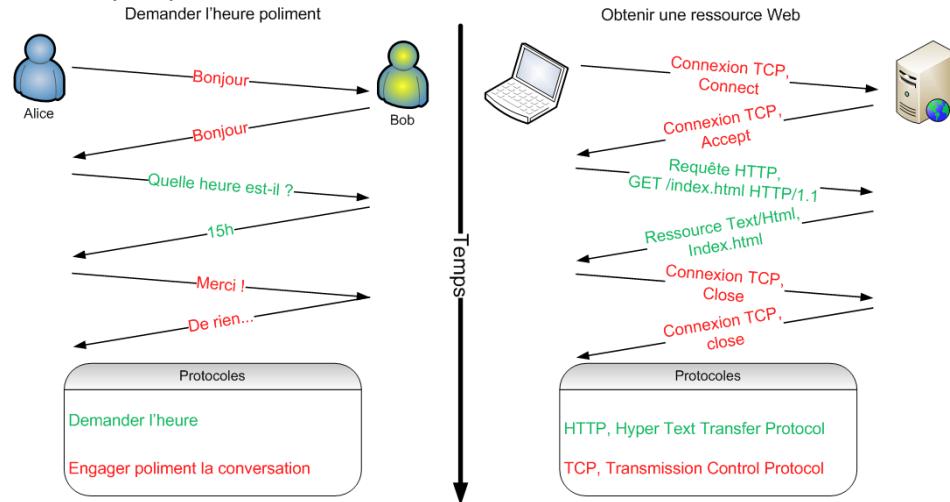
Ce sont les processus pairs qui **communiquent** à l'aide du protocole.

En réalité, aucune donnée ne passe directement de la couche n d'une machine à la couche n d'une autre machine, mais chaque couche passe par les données et les contrôle à la couche qui lui est immédiatement inférieure.



Un protocole humain et un protocole machine

« demander l'heure à quelqu'un » et « demander une ressource sur un serveur Web ».



Les protocoles définissent :

- * le **format** des données échangées ;
- * l'**ordre** des messages émis et reçus entre les entités réseaux;
- * ainsi que les **réactions** à ces messages.

Un protocole correspond à un **comportement** qui **évolue** en fonction des données échangées.

Exemple de protocole

42

Le protocole SMTP, «Simple Mail Transfer Protocol»

```
xterm
bonnefoi@msi:~$ socat - tcp:smtp.unilim.fr:25
220 smtp.unilim.fr ESMTP Sendmail 8.13.1/8.13.1; Thu, 15 Sep 2011 15:28:24 +0200
HELO msi.unilim.fr
250 smtp.unilim.fr Hello www.msi.unilim.fr [164.81.60.6], pleased to meet you
MAIL FROM: <bonnefoi@unilim.fr>
250 2.1.0 <bonnefoi@unilim.fr>... Sender ok
RCPT TO: <bonnefoi@unilim.fr>
250 2.1.5 <bonnefoi@unilim.fr>... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
Subject: Message

Message de test envoyé directement !

.
250 2.0.0 p8FDSOJe031646 Message accepted for delivery
QUIT
221 2.0.0 smtp.unilim.fr closing connection
bonnefoi@msi:~$
```

La commande «socat» permet de simplement établir une connexion TCP avec le serveur que l'on a désigné sur le port indiqué.



From: Pierre-François Bonnefoi
Subject: Message
Date: 15 septembre 2011 15:28:24 HAEC
To: undisclosed-recipients;;

Message de test envoyé directement !



Exemple de protocole

Le protocole HTTP, «*Hyper Text Transfer Protocol*»

```
xterm
pef@darkstar-8:/Users/pef socat - tcp:www.unilim.fr:80
HEAD / HTTP/1.0

HTTP/1.1 301 Moved Permanently
Server: nginx
Date: Mon, 11 Sep 2017 10:53:33 GMT
Content-Type: text/html
Content-Length: 178
Connection: close
Location: http://www.cryptis.fr/
```

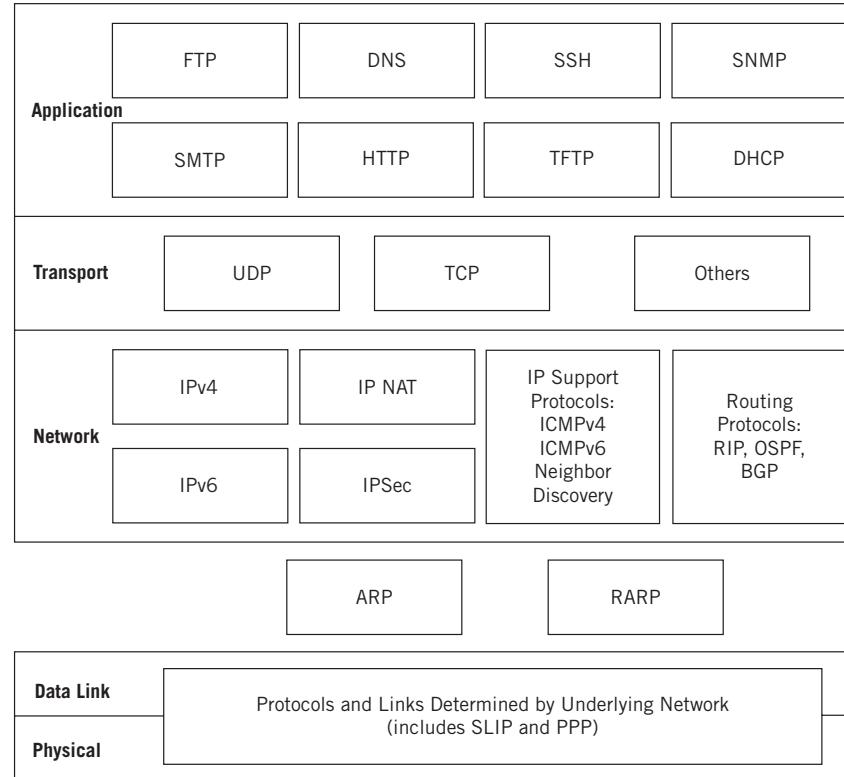
Le protocole POP, «*Post Office Protocol*»

```
xterm
bonnefoi@msi:~$ socat - tcp:pop.unilim.fr:110
+OK courriel Cyrus POP3 v2.2.13-Debian-2.2.13-14.xm.1 server ready <299345444.1316380363@courriel>
USER bonnefoi
+OK Name is a valid mailbox
PASS bob
-ERR [AUTH] Invalid login
^C
bonnefoi@msi:~$
```

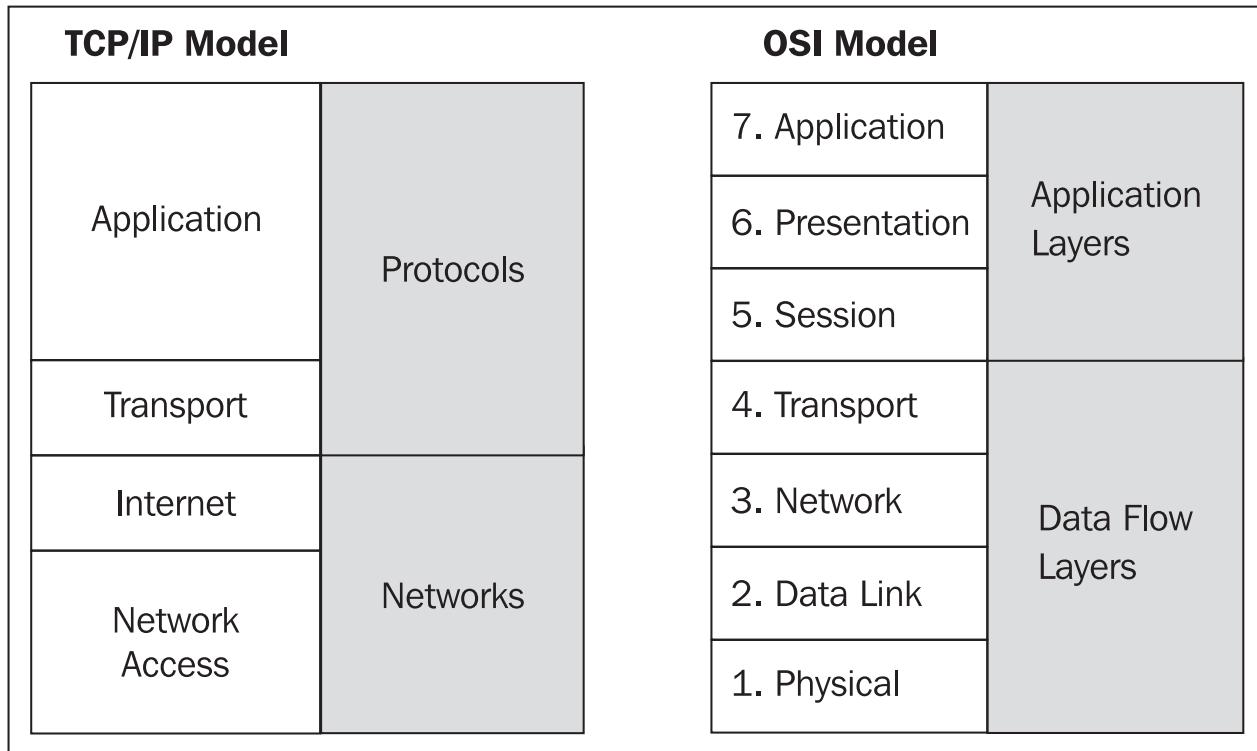
Ce protocole est utilisé pour consulter son courrier et le récupérer dans son logiciel de messagerie.

On lui préfère le protocole IMAP, port 143 en version non sécurisée, qui permet de consulter son courrier tout en le laissant sur le serveur.



**Attention :**

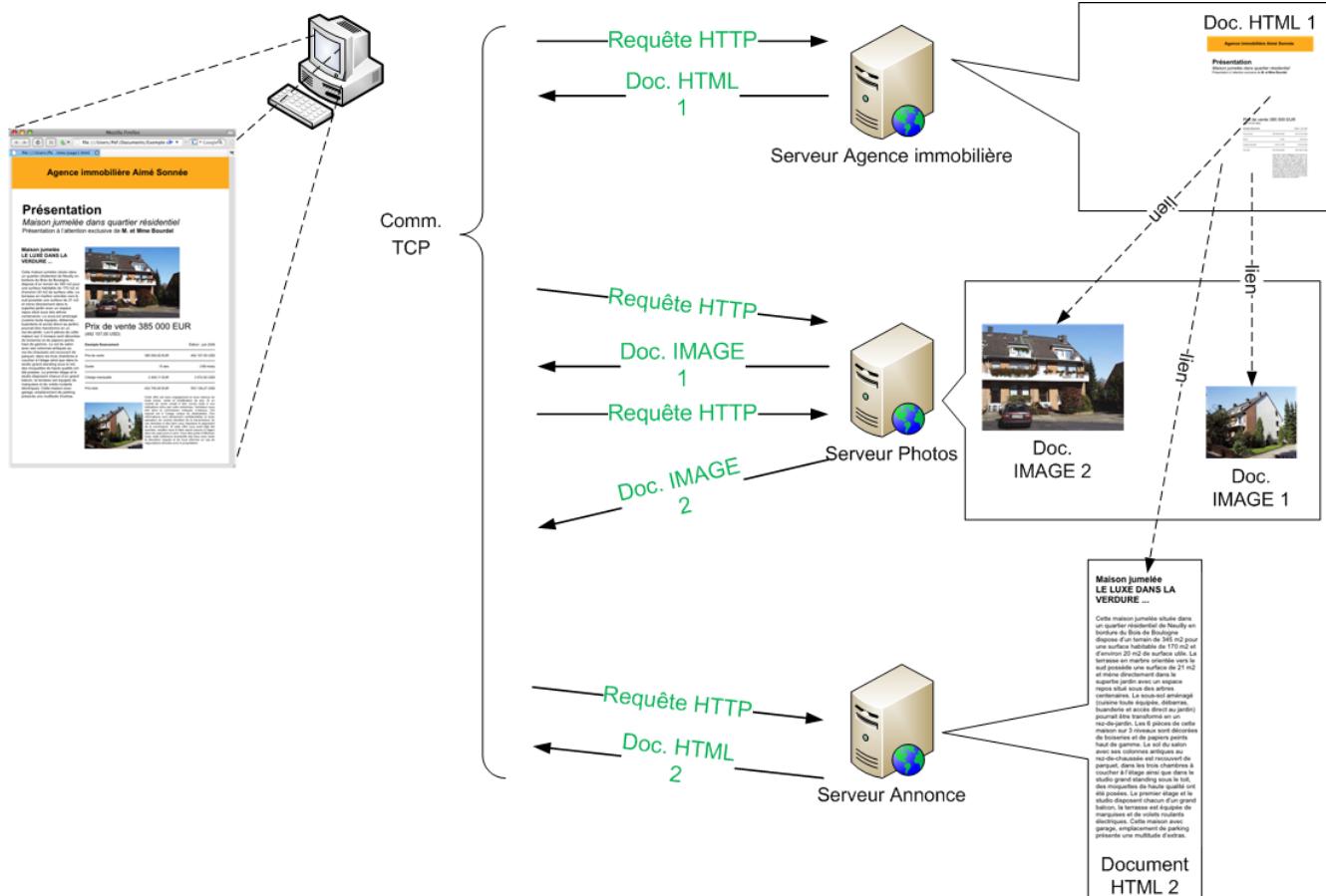
- * certains protocoles comme RIP, OSPF, BGP utilisent des protocoles de niveau 4, «Transport».
- * le protocole ARP est entre les couches 2 et 3.



Pas de couche «Présentation» et de couche «session» dans TCP/IP par rapport à OSI.

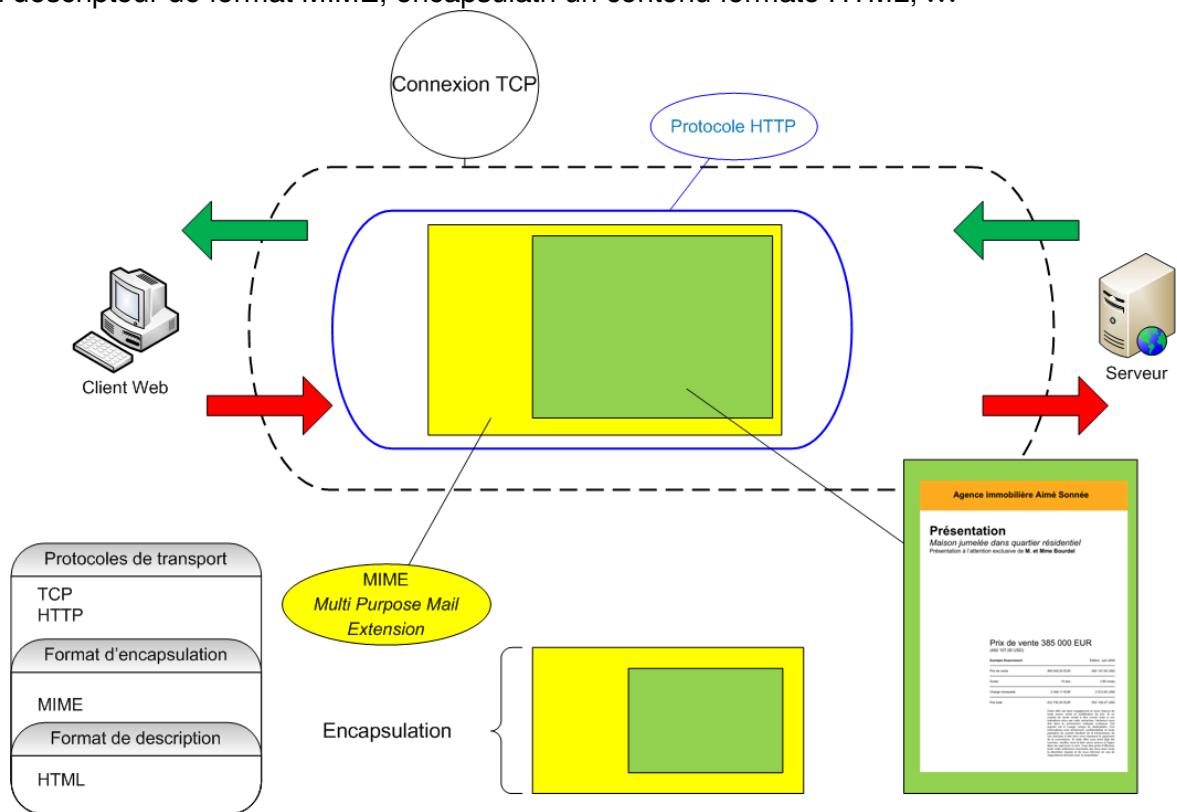
Le protocole HTTP, «HyperText Transfer Protocol», RFC 1945

46



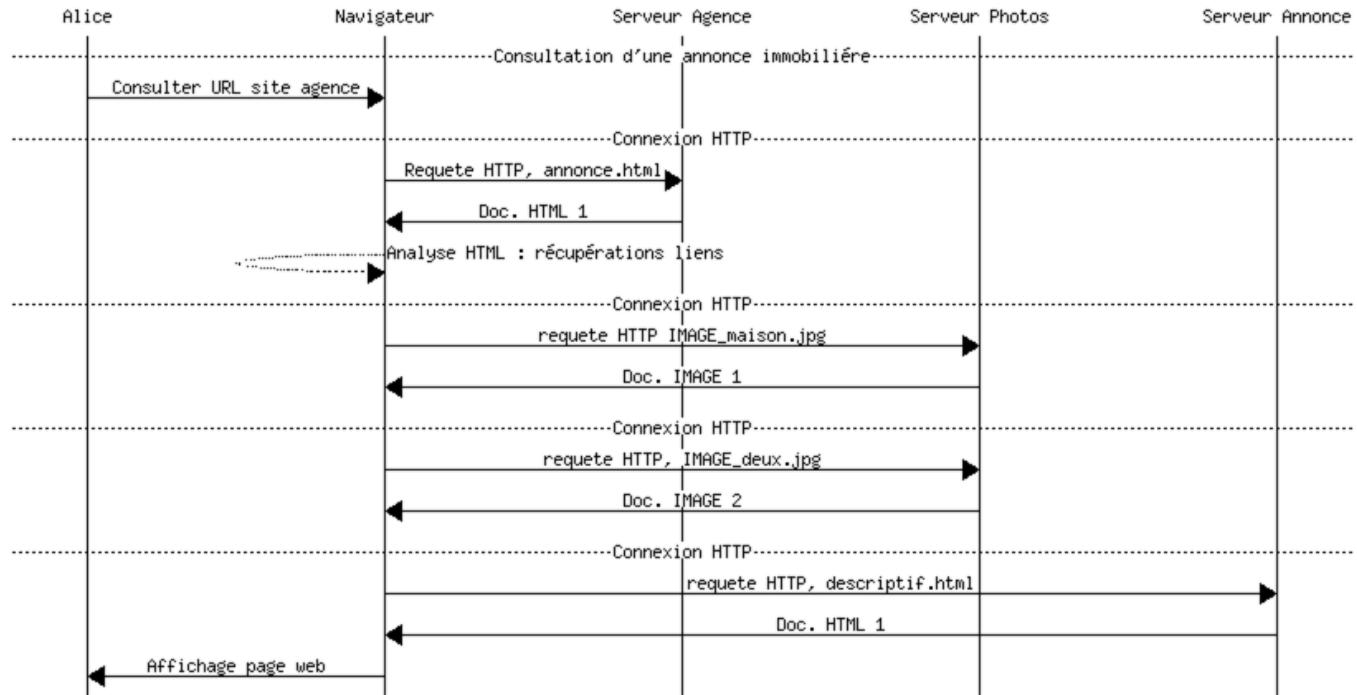
Le protocole HTTP

Utilisation du «mode connecté» : protocole de transport TCP, encapsulant le protocole HTTP pour échanger un descripteur de format MIME, encapsulatn un contenu formaté HTML, ...



Une pile de protocole

- * protocole « utilisateur » ou abstrait: consultation d'une page web ;
- * protocole de transport: TCP ;
- * protocole d'échange : HTTP ;
- * format d'échange : MIME.



Rapport entre les différents protocoles

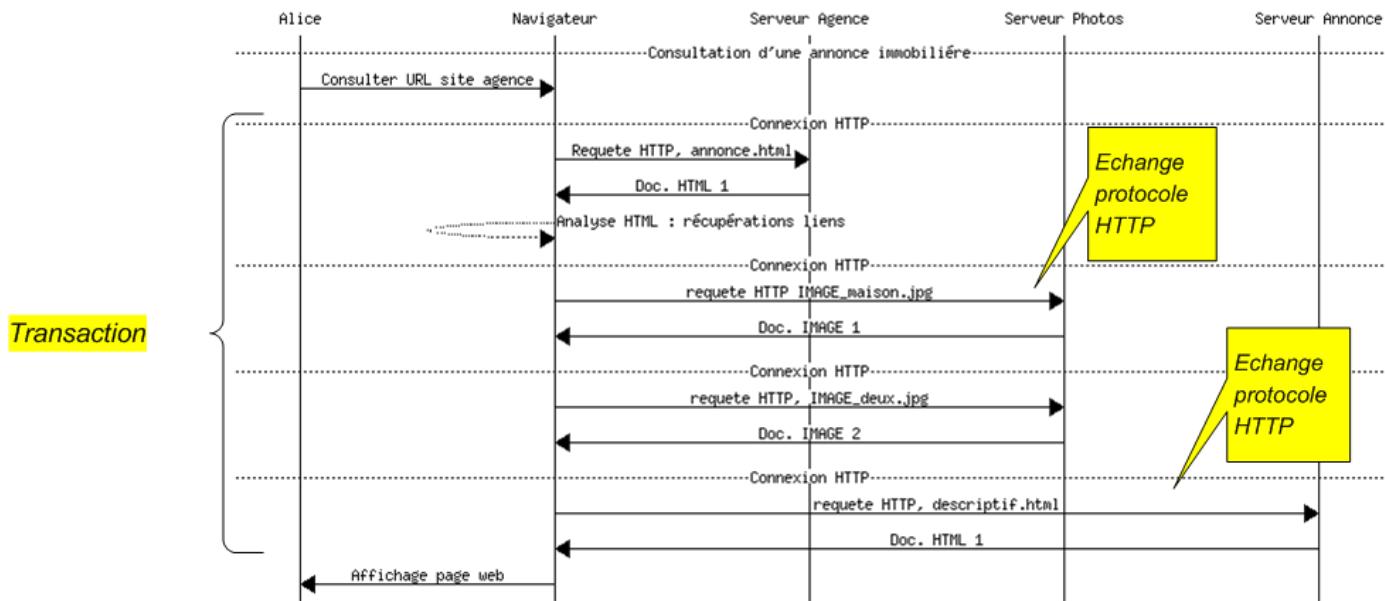
Le protocole abstrait est celui qui intéresse l'utilisateur (ici, Alice), c-à-d. « naviguer sur le Web ».

L'unité élémentaire de ce protocole est la transaction (Alice charge une page Web).

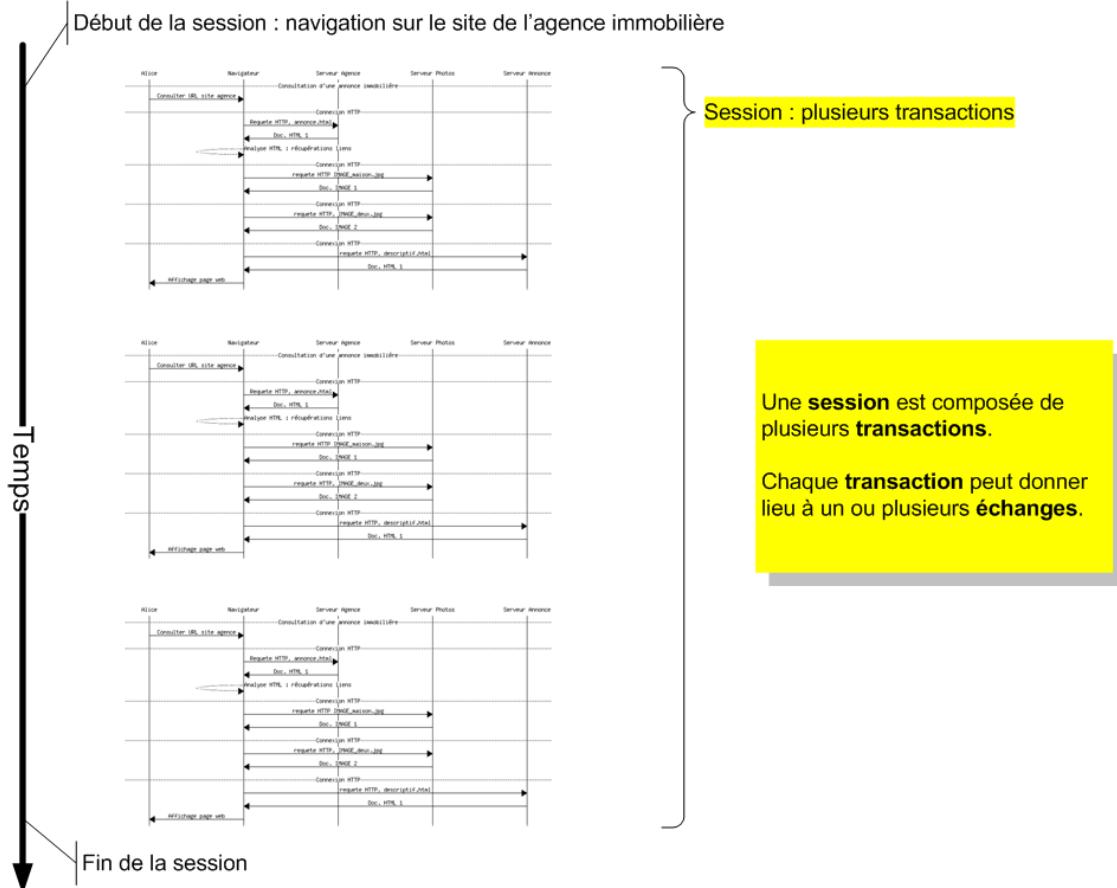
Le navigateur d'Alice réalise plusieurs échanges au format HTTP pour récupérer les contenus multimédia.

La notion de session décrit l'ensemble des transactions qui ont un certain lien entre elles.

Par exemple : entrer dans le magasin virtuel, s'identifier, remplir son caddie, payer et quitter le site.

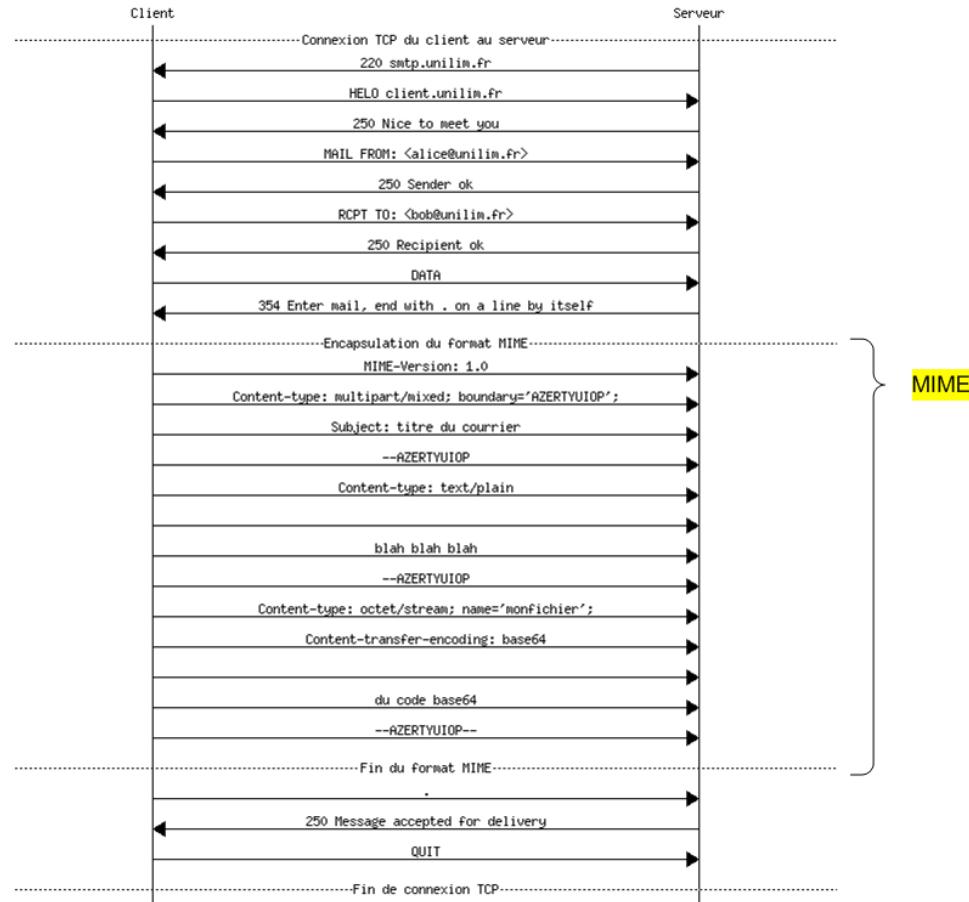


Utilisation de cookies, de données de formulaires, de contenus JSON, d'URL particulière (REST), etc.



Le protocole SMTP

51



MIME

Éléments importants

- ▷ TSAP, «*Transport Service Access Point*» ;
- ▷ Mode «orienté connexion» vs mode datagramme ;
- ▷ Protocoles TCP et UDP ;
- ▷ Mode «client/serveur» ;



Une **interface de programmation** définie pour mettre en place **simplement** des communications :

- * chaque communication a lieu avec :
 - ◊ **un interlocuteur** : communication «point à point», ou «unicast» ;
 - ◊ **plusieurs interlocuteurs** : communication par «diffusion» ou «multicast» ;
- * la communication correspond à l'échange de données entre les interlocuteurs :
 - ◊ des **données en continu** : flux d'octets de taille indéfinie, non connue à l'avance ;
 - ◊ des **paquets** : données de taille fixe et réduite connue à l'avance.

Deux types de communication uniquement en TCP/IP

1. mode «connecté»

- ◊ elle ne concerne que **deux interlocuteurs** : un de chaque côté (point à point) ;
- ◊ les données arrivent les unes après les autres dans «l'ordre d'émission» ;
- ◊ la communication est **bi-directionnelle** (dans les deux sens) ;
- ◊ elle est «full-duplex», les deux interlocuteurs peuvent échanger **simultanément** ;
- ◊ il y a une **garantie contre la perte de données**.

*C'est le mode offert par le protocole **TCP**, «Transmission Control Protocol».*

2. mode «datagramme»

- ◊ elle peut concerner un ou plusieurs interlocuteurs (unicast ou multicast) ;
- ◊ les données sont groupées dans des **paquets de taille limitée** ;
- ◊ il peut y avoir des **pertes de paquets**.

*C'est le mode offert par le protocole **UDP**, «User Datagram Protocol».*

Attention

Le mode «connecté» est simulé par TCP sur un réseau en mode «datagramme».

Modèle Client/Serveur

Un logiciel «serveur» **attend** la communication en provenance d'un logiciel «client».

Localisation du logiciel serveur

- un ordinateur est localisable sur Internet grâce à son adresse IP ;
- un ordinateur ne possède habituellement qu'une adresse IP joignable ;
- un ordinateur peut exécuter plusieurs programmes qui peuvent vouloir communiquer simultanément ;
- il faut **multiplexer ces communications** en «sachant» avec quel programme communiquer : notion de «port» !

À chaque processus communiquant est associé un port

Pour une communication en «mode connecté» :

- * un Serveur qui **attend** la connexion du client ;
- * un Client qui **effectue** la connexion au serveur.

Pour localiser le Serveur ? Connaître le numéro de port où attend la communication !

Comment connaître le numéro de port ?

Le point sur les communications sur un ordinateur :

- * chaque communication est associée à **un seul programme donné** (logiciel de messagerie, navigateur web, client de chat, etc) ;
- * chaque communication se fait suivant un **protocole donné** (SMTP, POP pour récupérer le courrier, HTTP, etc) ;
- * chaque protocole est associé à un «serveur» particulier : serveur SMTP pour l'envoi de courrier, serveur Web, serveur FTP, etc.
- * un **numéro de port identifie un serveur donné** : il faut rendre **standard** les numéros de port !

Exemple : http : 80, ftp : 21, smtp : 25, DNS : 53 etc, la liste dans le fichier /etc/services.

Le client veut communiquer avec un serveur donné ? il utilise le port standard associé !



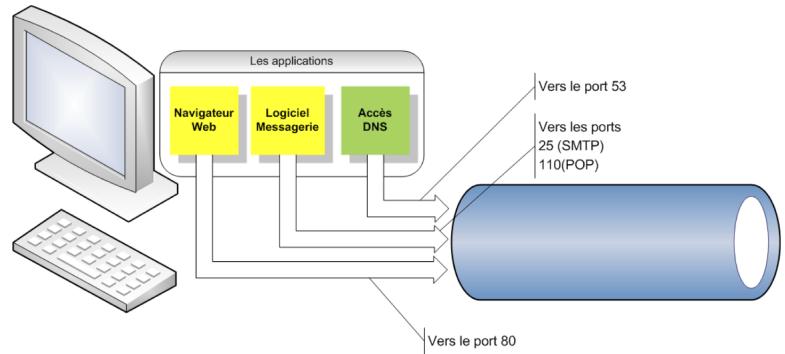
Notion de numéro de port

- ◊ différentes communications peuvent avoir lieu pour des protocoles différents, donc des programmes différents, donc des numéros de port différents ;
- ◊ chaque communication sur une machine est identifiée par un **TSAP**, «*Transport Service Access Point*», c-à-d un couple (@IP, numéro de port).

Comment un ordinateur peut-il voir plusieurs communications simultanément ?

On ajoute également la notion de **numéro de port** :

- * il varie de 1 à 65535 (sur 16 bits) ;
- * il est associé à un seul programme ;
- * du côté de la machine *cliente*, il peut prendre n'importe quelle valeur ;
- * du côté de la machine *serveur*, il permet à la machine cliente de désigner le programme que l'on veut contacter ;



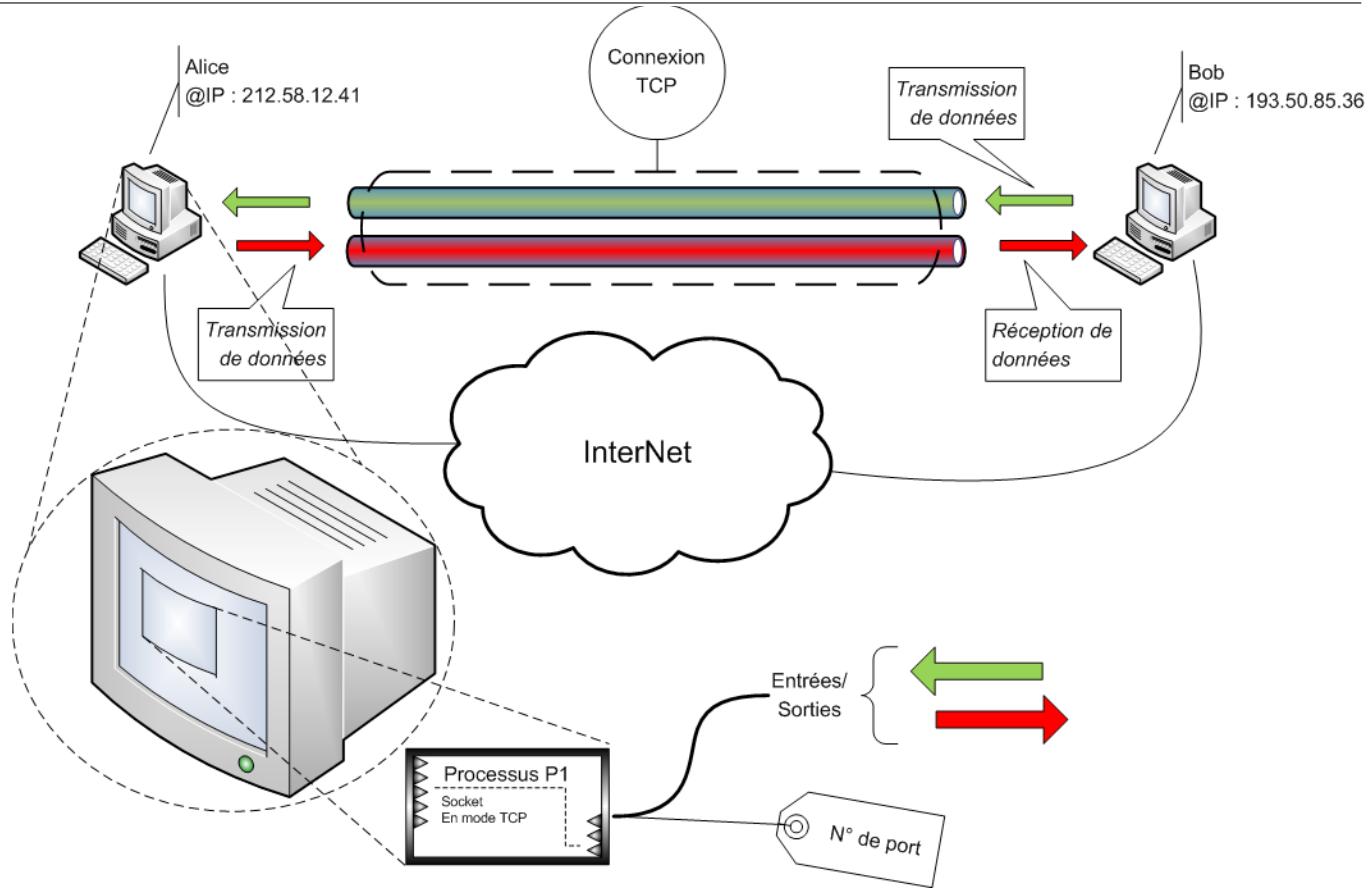
Le port permet de **multiplexer** les communications :

- chaque datagramme sera identifié par le $TSAP_{source}$ duquel il transporte les données ;
- tous les datagrammes utilisent le même lien de communication ;
- lors de leur arrivée sur la machine destination, ils sont identifiés par leur $TSAP_{destination}$ et remis au bon processus.



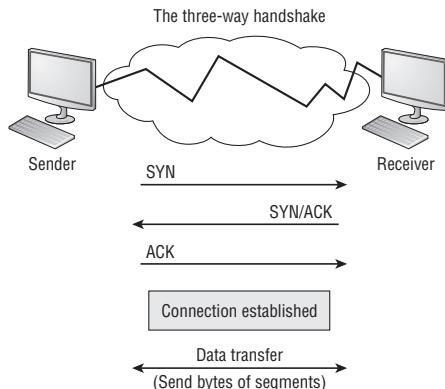
Le protocole TCP : connexion et échanges

56



Une communication «orientée connexion» correspond à :

- ▷ une **demande d'accord** de la part de l'interlocuteur **avant de lui envoyer des données**, c-à-d une 1/2 connexion;
- ▷ un envoi de données **sans perte et sans erreur**;
- ▷ une communication **bi-directionnelle** (d'un interlocuteur vers l'autre et vice-versa) et **«full duplex»** (chaque interlocuteur peut communiquer simultanément avec l'autre);



Celui qui **initie** la communication est le client.

Celui qui **attend** la communication est le serveur.

Firewall & Filtrage

Communication autorisée (non filtrée) :

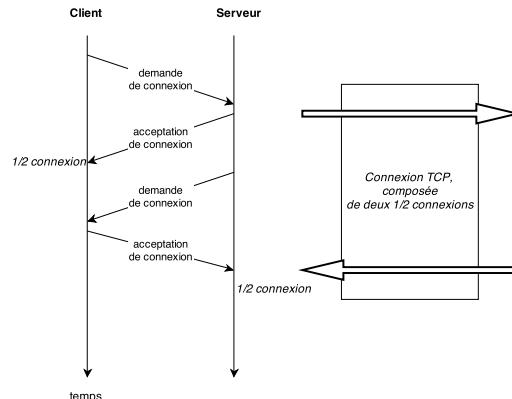
Si le client est dans le LAN et le serveur sur Internet \Rightarrow

Communication Intérieur \rightarrow Extérieur

L'appartenance du Client au LAN est déduite grâce à la présence du «SYN».

«The three-way handshake», correspond à l'établissement d'une communication depuis le client vers le serveur :

- une demi-connexion du client vers le serveur ;
- une demi-connexion du serveur vers le client ;

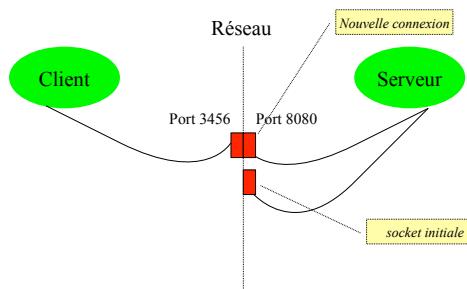
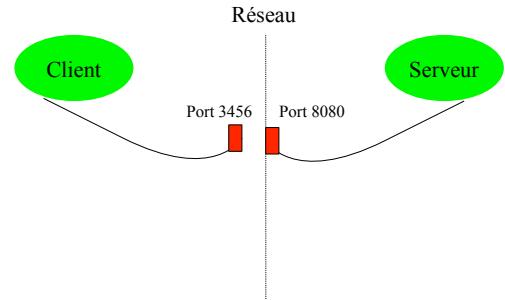


À noter : la demande de 1/2 connexion du serveur (**SYN**), ainsi que son acceptation de la demande de 1/2 connexion du client (**ACK**), sont transmises dans le même message, d'où la simplification en 3 échanges seulement.

Les différentes étapes pour l'établissement de la connexion :

Les instructions réseaux à utiliser sont indiquées dans un cadre en fin de ligne.

1. Le serveur attend sur le SAP [`@IP serveur, numéro de port`]
`socket, bind, listen, accept`
2. Le client obtient automatiquement un numéro de port libre (par ex. 3456)
`socket`



3. Le client se connecte au serveur

Le système d'exploitation du client et du serveur, mémorise la connexion par un couple :

$$(TSAP_{client} \Leftrightarrow TSAP_{serveur}) \\ [@IP client, 3456] \Leftrightarrow [@IP serveur, 8080]$$

Cette connexion peut être affichée avec la commande Unix « `netstat` » ou « `ss` » sous Linux.

Remarques :

- ◊ La primitive de programmation `accept` retourne au serveur une nouvelle socket associée à la connexion avec le client.
C'est par cette socket que l'on communique avec le client.
- ◊ Le serveur peut recevoir la connexion de nouveaux clients sur la socket initiale.
Un serveur peut avoir plusieurs communications simultanées avec différents clients.
Chacune de ces communications correspond à un couple différent de TSAP (le même du côté du serveur associé à un TSAP côté client différent pour chaque communication).



Schéma de fonctionnement en TCP

Serveur

```
1 socket  
2 bind  
3 listen  
4 accept  
5   recv, send  
6 close
```

Client

```
1 socket  
2 connect  
3   recv, send  
4 close
```

Le protocole UDP

- ◊ C'est un protocole de transport **non fiable, sans connexion**.
- ◊ L'échange de données se fait par datagrammes : un «datagramme UDP» = un «datagramme IP».
- ◊ L'ordre dans lequel les paquets sont envoyés peut ne pas être respecté lors de leur réception.

Schéma de fonctionnement en UDP

Serveur

```
1 socket  
2 bind  
3   recvfrom, sendto  
4 close
```

Client

```
1 socket  
2   recvfrom, sendto  
3 close
```



Le client

```
1 import socket, sys
2
3 adresse_symbolique_serveur = 'localhost' # la machine elle-même
4 numero_port_serveur = 6688 # le numéro de port sur le serveur
5
6 # réalisation de la requête DNS pour obtenir l'adresse IP
7 adresse_serveur = socket.gethostbyname(adresse_symbolique_serveur)
8
9 ma_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10
11 try:
12     ma_socket.connect((adresse_serveur, numero_port_serveur)) #TSAP désignant le serveur
13
14 except Exception as e:
15     print (e.args)
16     sys.exit(1)
17
18 while 1:
19     entrée_clavier = input(':>')
20     if not entrée_clavier:
21         break
22     ma_socket.sendall(bytes(entrée_clavier, encoding='UTF-8')+b'\n')
23
24 ma_socket.close()
```



Le serveur

```
1 import socket
2
3 masque_acces = '' # filtre les clients, ici aucun n'est filtré
4 numero_port_serveur = 6688 # identique à celui du client
5
6 # création de la socket d'attente de connexion
7 ma_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM, socket.IPPROTO_TCP)
8
9 # Permet de ne pas attendre pour réutiliser le numéro de port
10 ma_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
11
12 # Accroche le numéro de port à la socket
13 ma_socket.bind((masque_acces, numero_port_serveur))
14
15 # Configure la file d'attente
16 ma_socket.listen(socket.SOMAXCONN)
17
18 # L'accept renvoie une nouvelle socket
19 (nouvelle_connexion, tsap_depuis) = ma_socket.accept()
20 print ("Nouvelle connexion depuis ", tsap_depuis)
21 while 1:
22     ligne = nouvelle_connexion.recv(1000) # au plus 1000
23     if not ligne :
24         break
25     print (ligne)
26 nouvelle_connexion.close() # fermeture de la connexion vers le client
27
28 ma_socket.close() # fermeture de la socket d'attente de connexion
```

