

Université Claude Bernard  Lyon 1

**Licence Math-Informatique 1<sup>ère</sup> année**

**Partie 8**

Olivier Glück

Université LYON 1 / Département Informatique

Olivier.Gluck@univ-lyon1.fr

<http://perso.univ-lyon1.fr/olivier.gluck>

# Copyright

- Copyright © 2017 Olivier Glück; all rights reserved
- Ce support de cours est soumis aux droits d'auteur et n'est donc pas dans le domaine public. Sa reproduction est cependant autorisée à condition de respecter les conditions suivantes :
  - Si ce document est reproduit pour les besoins personnels du reproducteur, toute forme de reproduction (totale ou partielle) est autorisée à la condition de citer l'auteur.
  - Si ce document est reproduit dans le but d'être distribué à des tierces personnes, il devra être reproduit dans son intégralité sans aucune modification. Cette notice de copyright devra donc être présente. De plus, il ne devra pas être vendu.
  - Cependant, dans le seul cas d'un enseignement gratuit, une participation aux frais de reproduction pourra être demandée, mais elle ne pourra être supérieure au prix du papier et de l'encre composant le document.
  - Toute reproduction sortant du cadre précisé ci-dessus est interdite sans accord préalable écrit de l'auteur.

# Plan du cours

- CM1 : Internet, les réseaux et le web
- CM2 : Pages HTML et feuilles de styles CSS
- CM3 : Web interactif, formulaires, pages dynamiques et PHP
- CM4 : Protocole HTTP, méthodes GET et POST
- CM5 : Les applications d'Internet
- CM6 : La couche transport : les protocoles TCP et UDP
- CM7 : Le protocole IP
- CM8 : Les protocoles Ethernet, ARP et ICMP. Synthèse des échanges entre un client et serveur Web

# CM8 – Les protocoles Ethernet, ARP et ICMP

Format de la trame Ethernet

Le protocole ARP

L'architecture TCP/IP

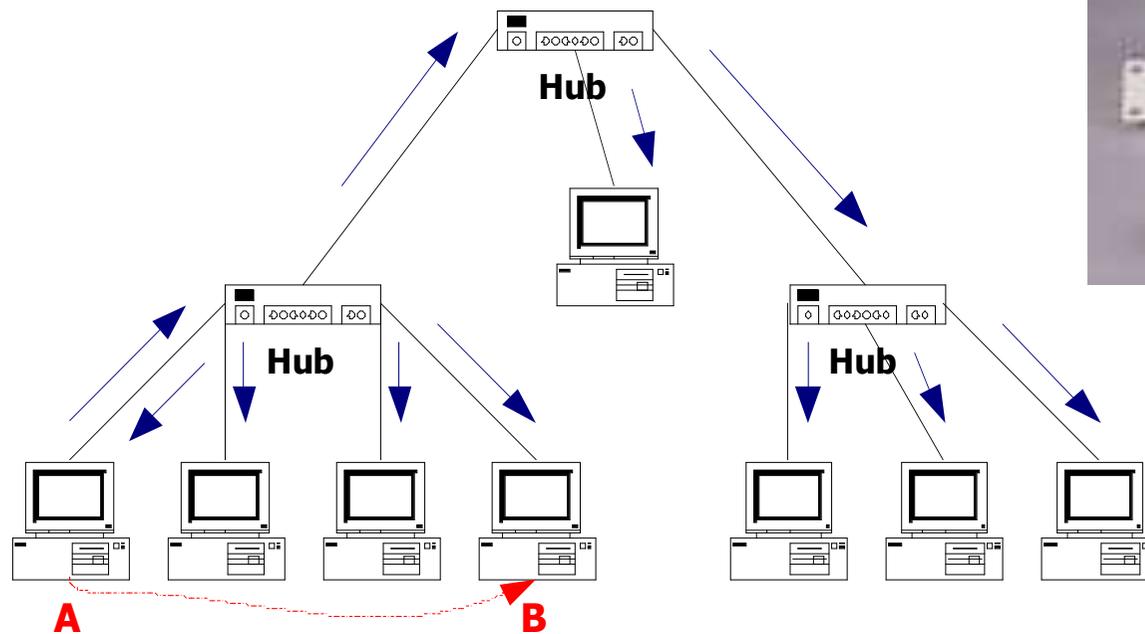
Synthèse des échanges entre un navigateur Web et un  
serveur Web

# Plan du CM8

- Format de la trame Ethernet
- Le protocole ARP
- L'architecture TCP/IP
- Synthèse des échanges entre un navigateur Web et un serveur Web

# Ethernet, le protocole des cartes réseaux

- Une carte réseau Ethernet fabrique un signal qui est envoyée à une autre carte réseau Ethernet
- Les répéteurs (Hub) ou commutateurs (Switch) interconnectent les ordinateurs entre eux
- Exemple avec des Hub



# Format de la trame Ethernet

## Trame Ethernet



- Préambule : 7 fois **10101010** pour la synchronisation bit
- SFD (*Start Frame Delimitator*) : **10101011** pour la synchronisation octet
- Bourrage si  $L_g < 46$  octets pour détection collision
- FCS sur 4 octets pour la détection d'erreur
- Différence IEEE 802.3/Ethernet : champ  $L_g$ /Type

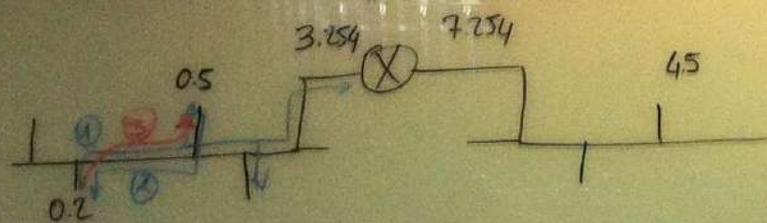
# ARP – Rôle et fonctionnement

- ARP (Address Resolution Protocol) sert à trouver l'adresse MAC d'une carte réseau alors que l'on connaît son adresse IP
- Si la machine source et destinataire sont sur le même réseau
  - 1 - requête ARP (broadcast MAC)
  - 2 - réponse ARP (le destinataire a reçu le broadcast et s'est reconnu, il envoie son @MAC)
  - 3 - la source peut envoyer ses données vers le destinataire (adresse MAC destination connue)
- Si elles ne sont pas sur le même réseau
  - la diffusion ne passe pas le routeur
  - résolution de proche en proche : le client envoie ses données au prochain saut dont l'adresse MAC est trouvée par ARP, puis le prochain envoie au suivant en utilisant sa table de routage et ARP...

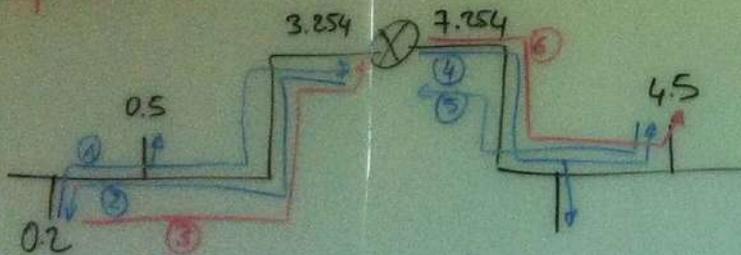
# ARP - Format du paquet

0	7	15	23	31
espace d'adressage physique		espace d'adressage logique		
lg @ physique	lg @ protocole	code		
adresse physique de l'émetteur de la trame...				
adresse physique (suite)		adresse du protocole de ...		
... l'émetteur de la trame		adresse physique du récepteur...		
... de la trame (inconnue)				
adresse du protocole récepteur du paquet				

# ARP - ICMP



- ① Requête ARP pour trouver @mac de 0.5
- ② Réponse ARP
- ③ Requête ICMP



- ① Requête ARP pour trouver @mac de 3.254
- ② Réponse ARP de 3.254
- ③ Requête ICMP de 0.2 vers 4.5
- ④ Requête ARP pour trouver @mac de 4.5
- ⑤ Réponse ARP de 4.5
- ⑥ Requête ICMP de 0.2 vers 4.5

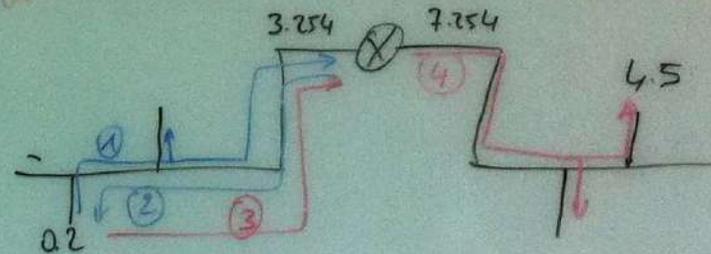
Sur 0.2 : ping 192.168.7.255 à 60

- ① Req ARP pour trouver @mac 3.254
- ② Réponse ARP de 3.254
- ③ Requête ICMP de 0.2 vers 7.255

Le routeur décapsule  
 @IP dot = 192.168.7.255  $\xrightarrow{\text{taille de routage}}$  eth1  
 diffusion IP  
 ↳ diffusion ethernet sur eth1

- ④ Requête ICMP part de eth1

@mac src = eth1(7.254)  
 @mac dest = FF:FF:FF:FF:FF:FF  
 type = 0800 (IP)



Sur le routeur: décapsulation IP. Comment aller vers 4.5?  
entête IP → @IP dst = 192.168.4.5 → table de routage du routeur

ifconfig eth0 192.168.3.254/22 192.168.0.0/22 → eth0  
eth1 192.168.7.254/22 192.168.4.0/22 → eth1

Pour aller vers 4.5, on passe par eth1

→ 14 20 8 60 14

↳ @mac src = eth1 (7.254)

type = 0800 (IP)

@mac dst = ? celle de 4.5 que le routeur ne connaît pas

#### ④ Requête ARP

ARP [ @mac src = eth1 (7.254)  
@IP src = 192.168.7.254  
@mac dst = 0 (?)  
@IP dst = 192.168.4.5

Frame 4

@mac dst = FF:FF:FF:FF:FF:FF  
@mac src = eth1 (7.254)  
type = 0806 (ARP) ] Etk

#### ⑤ Réponse ARP

ARP [ @mac src = eth1 (7.254)  
@IP src = 192.168.7.254  
@mac dst = eth0 (4.5)  
@IP dst = 192.168.4.5

Frame

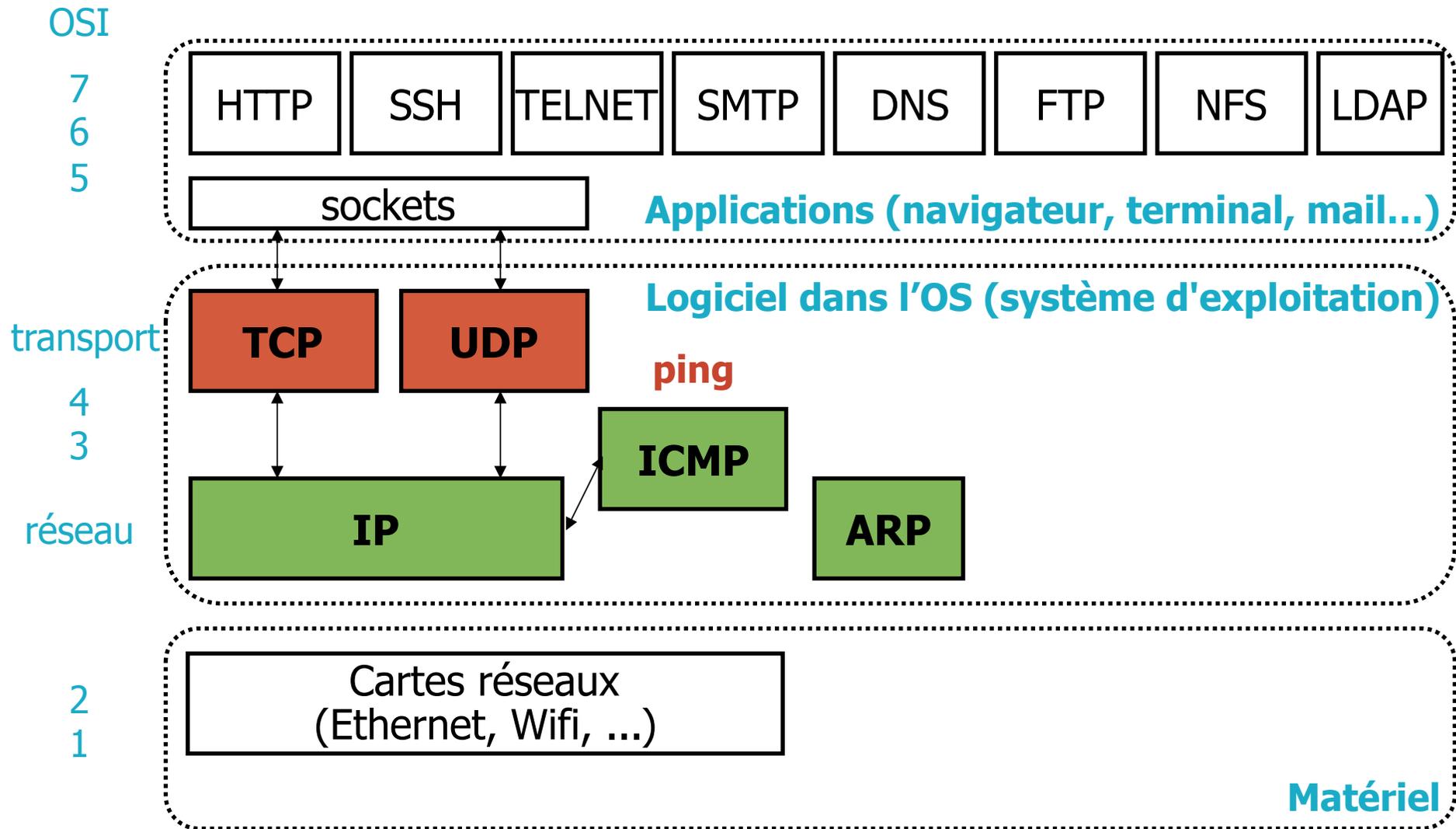
@mac dst = eth1 (7.254)  
@mac src = eth0 (4.5)  
type = 0806 (ARP) ] Etk

#### ⑥ Requête ICMP

→ @mac dst = eth0 (4.5)

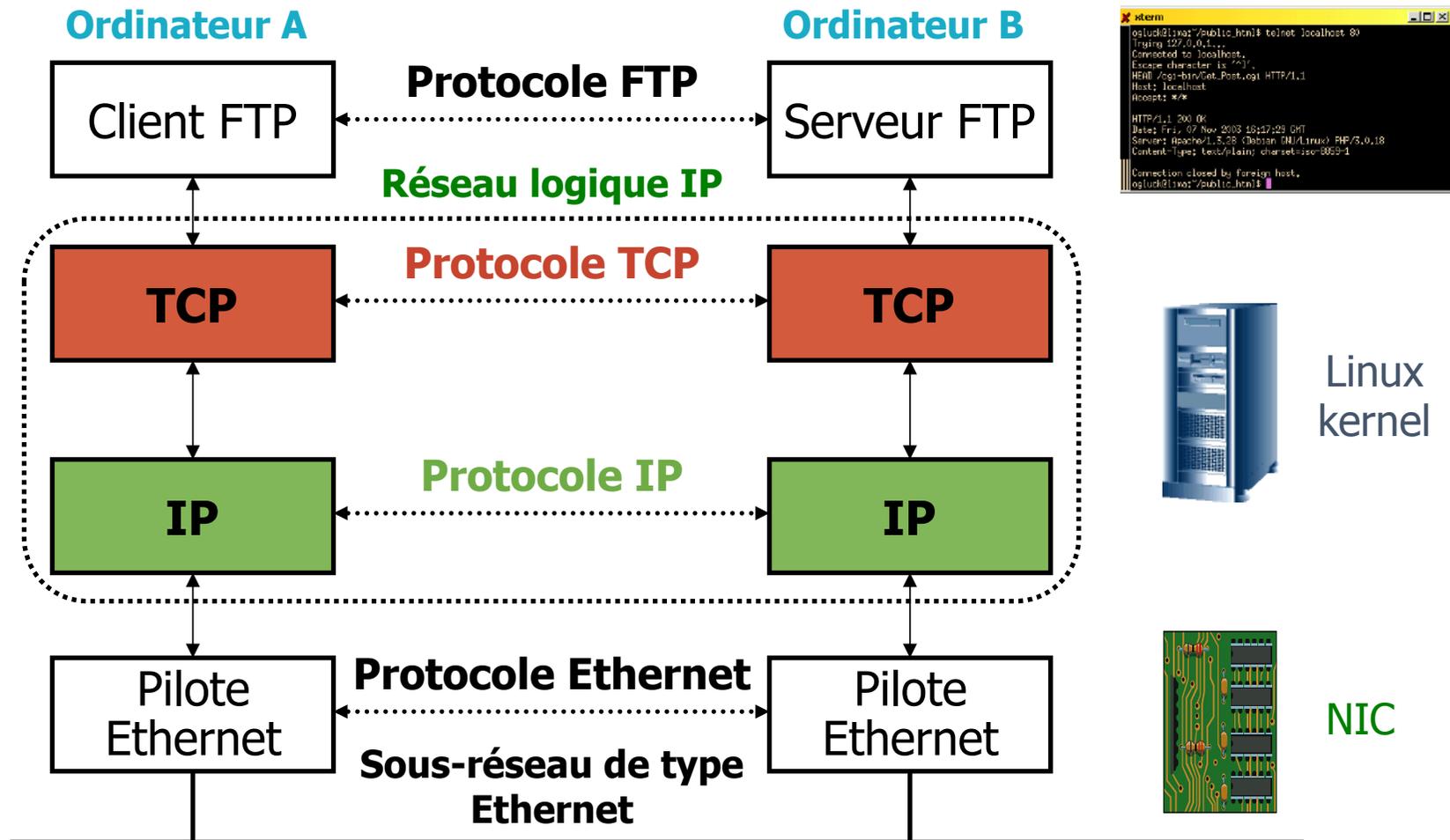
20 entête IP [ @IP dst = 192.168.4.5  
@IP src = 192.168.0.2  
proto = 1 (ICMP)

# L'architecture de TCP/IP



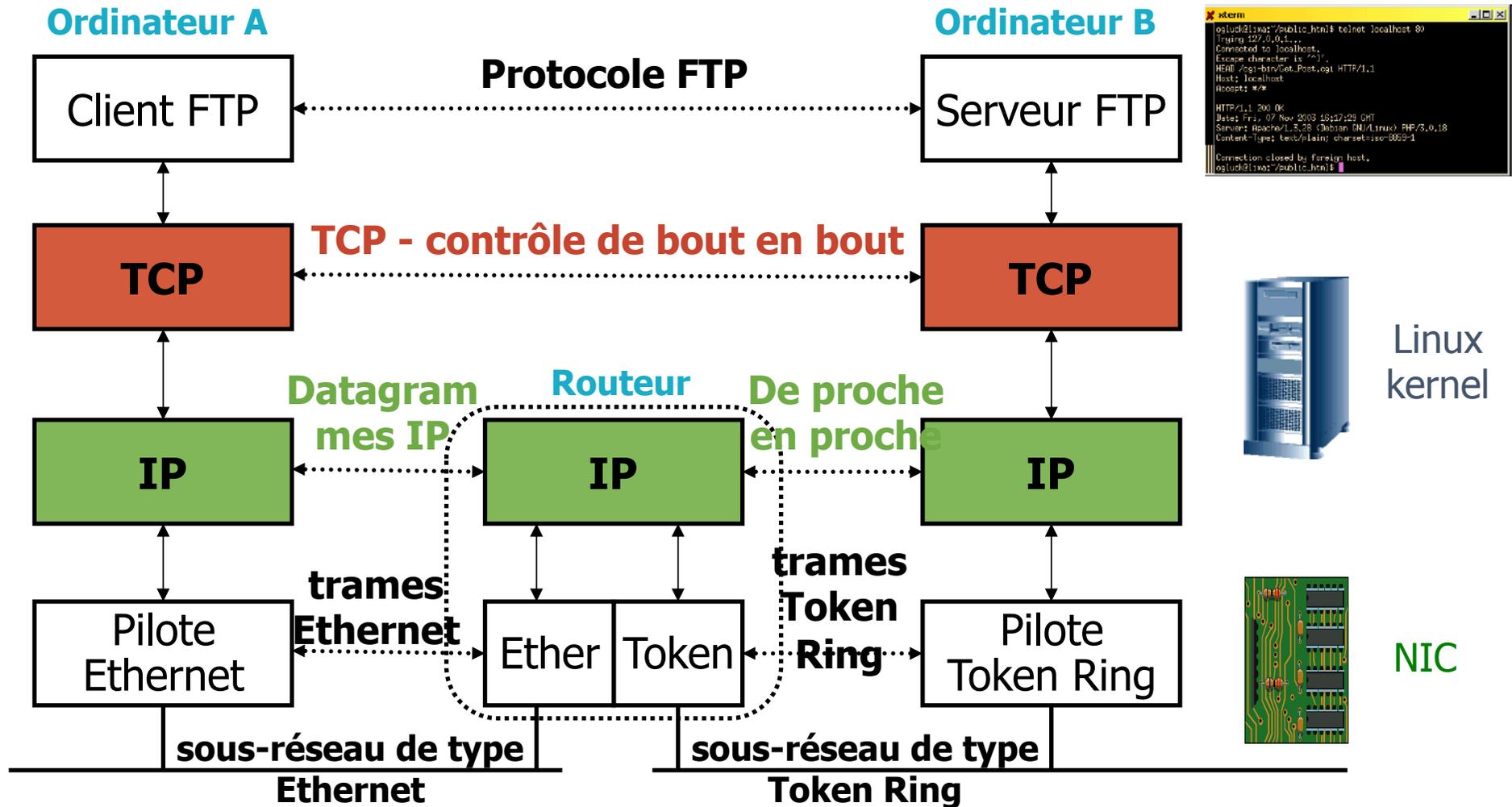
# Communications sans routeur

- Deux machines sur un même sous réseau

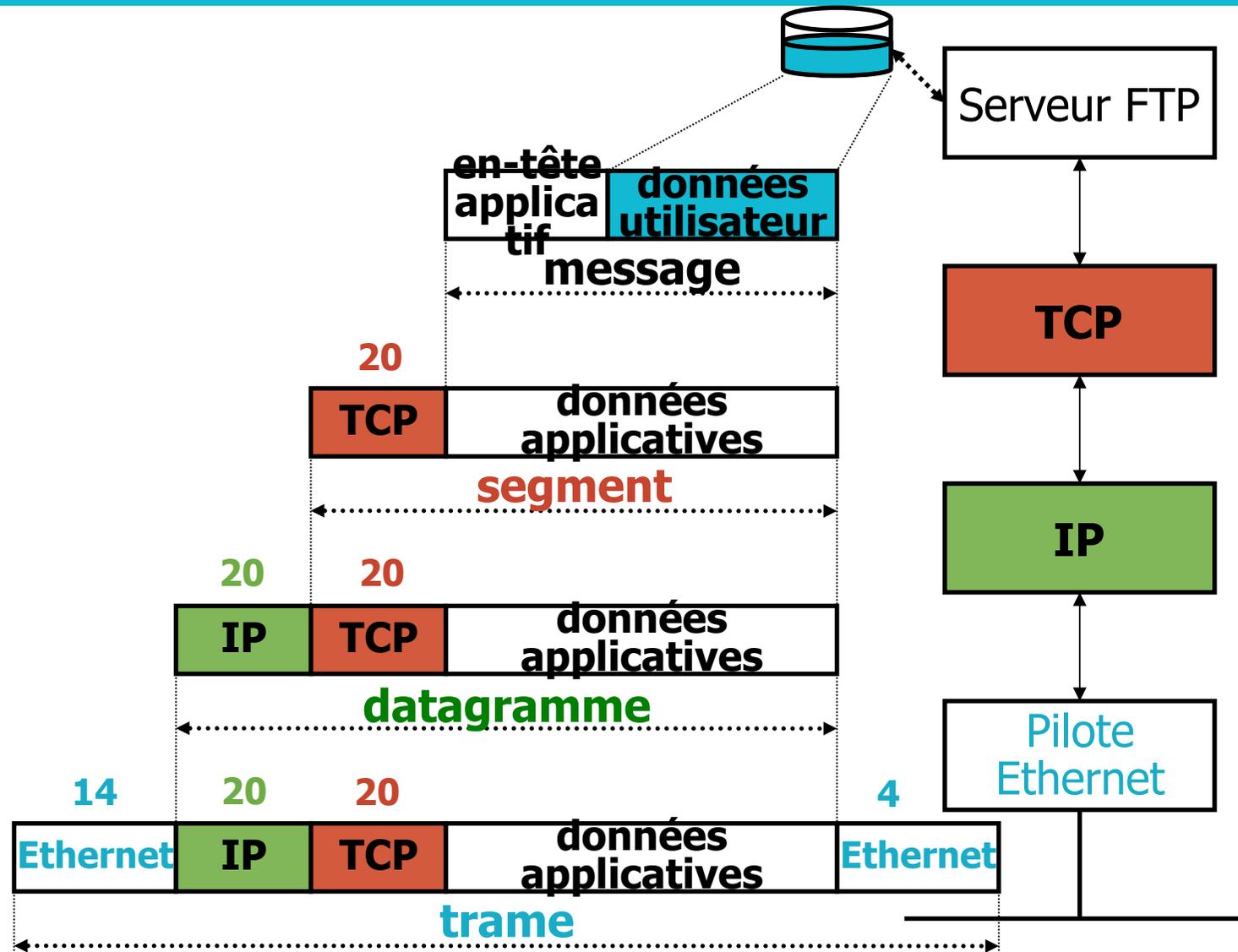


# Communications avec routeur(s)

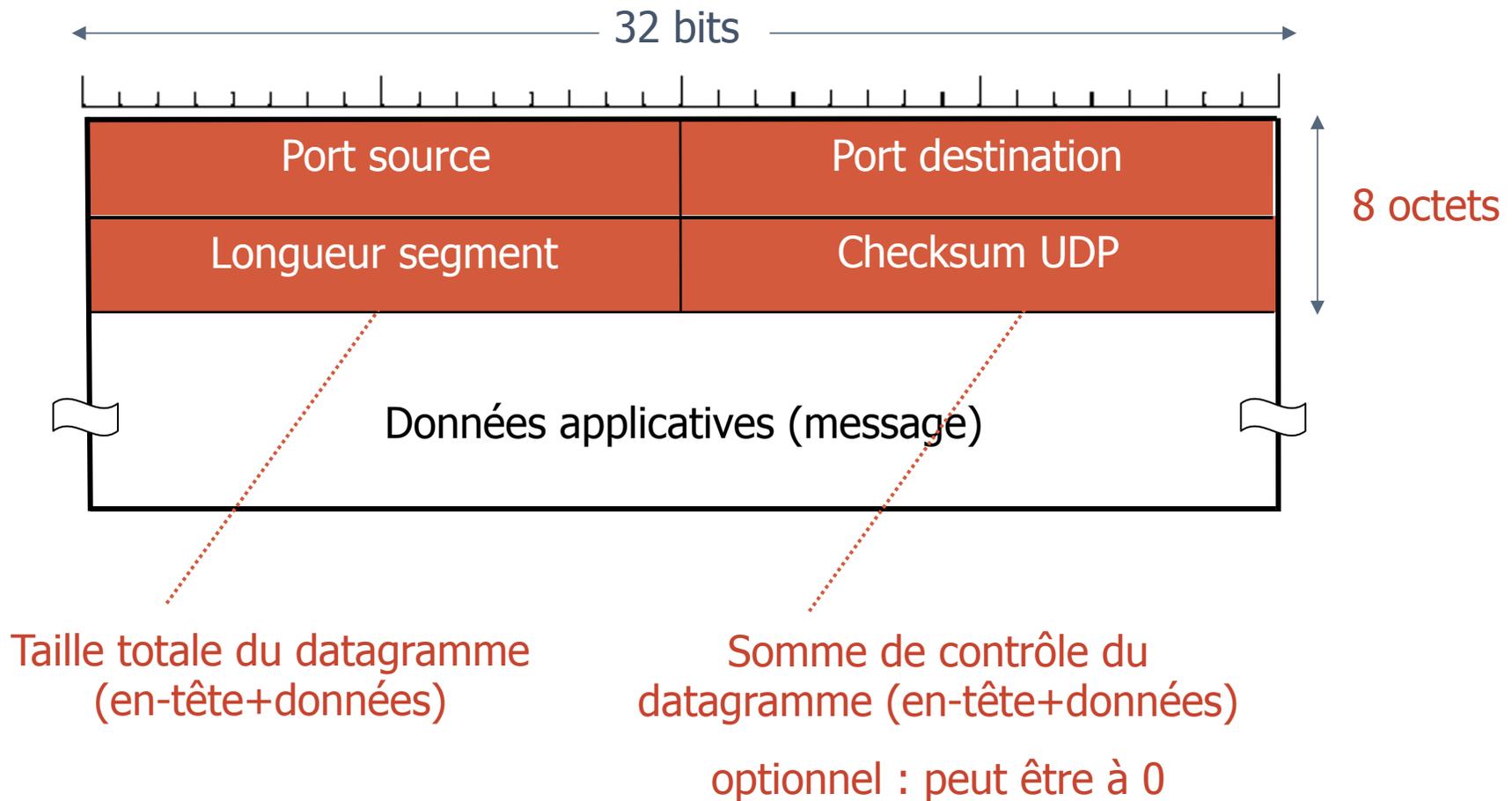
- Prise en compte de l'hétérogénéité



# L'encapsulation : ajout des en-têtes



# L'en-tête UDP : 8 octets



**UDP = IP + numéros de port !!**

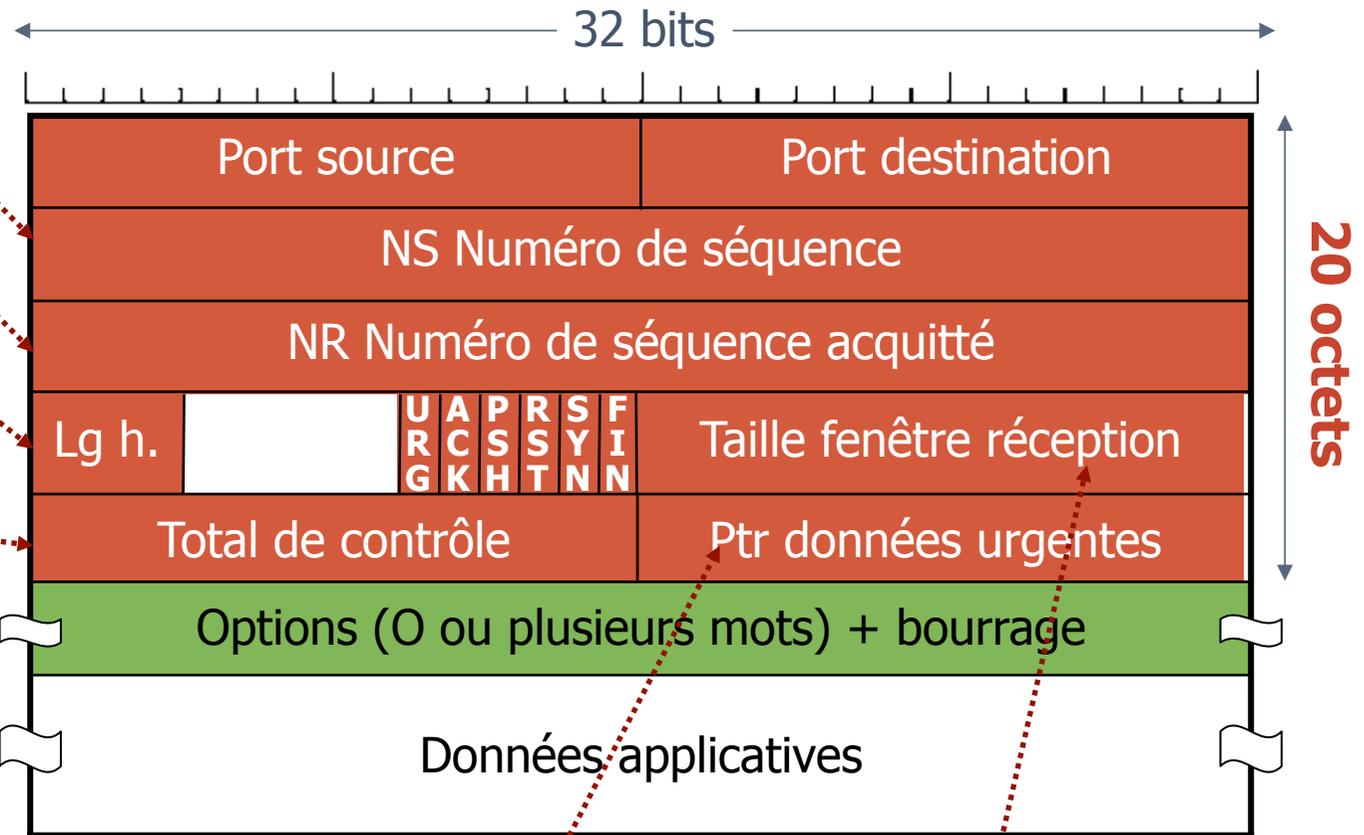
# L'en-tête TCP : 20 octets

Numéro du premier octet du segment

Numéro du prochain octet attendu

Longueur en-tête en multiple de 4 octets

Checksum sur tout le segment (cf. UDP)

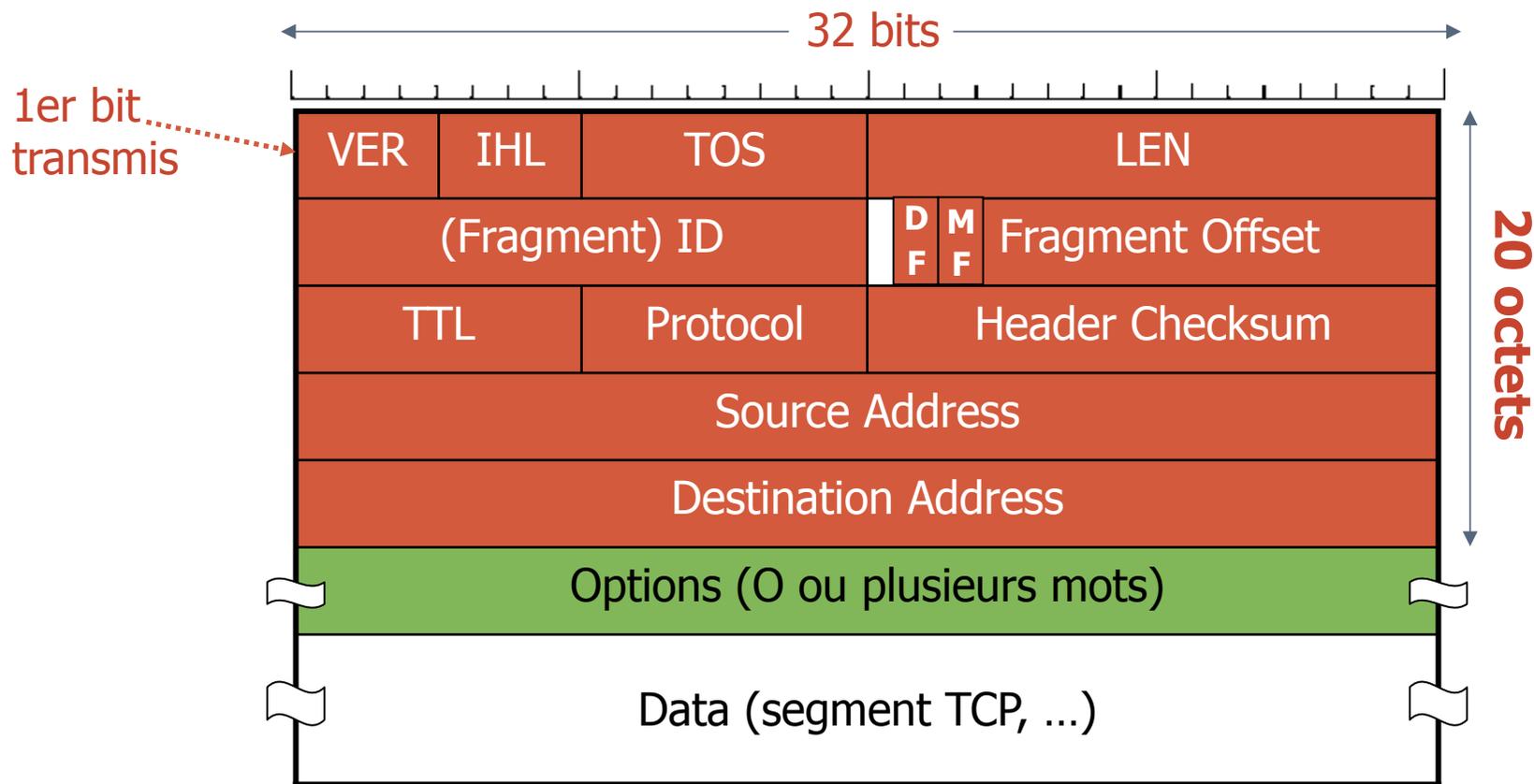


Les données comprises entre le premier octet DATA et la valeur du Ptr sont urgentes : TCP interrompt l'application pour forcer la lecture

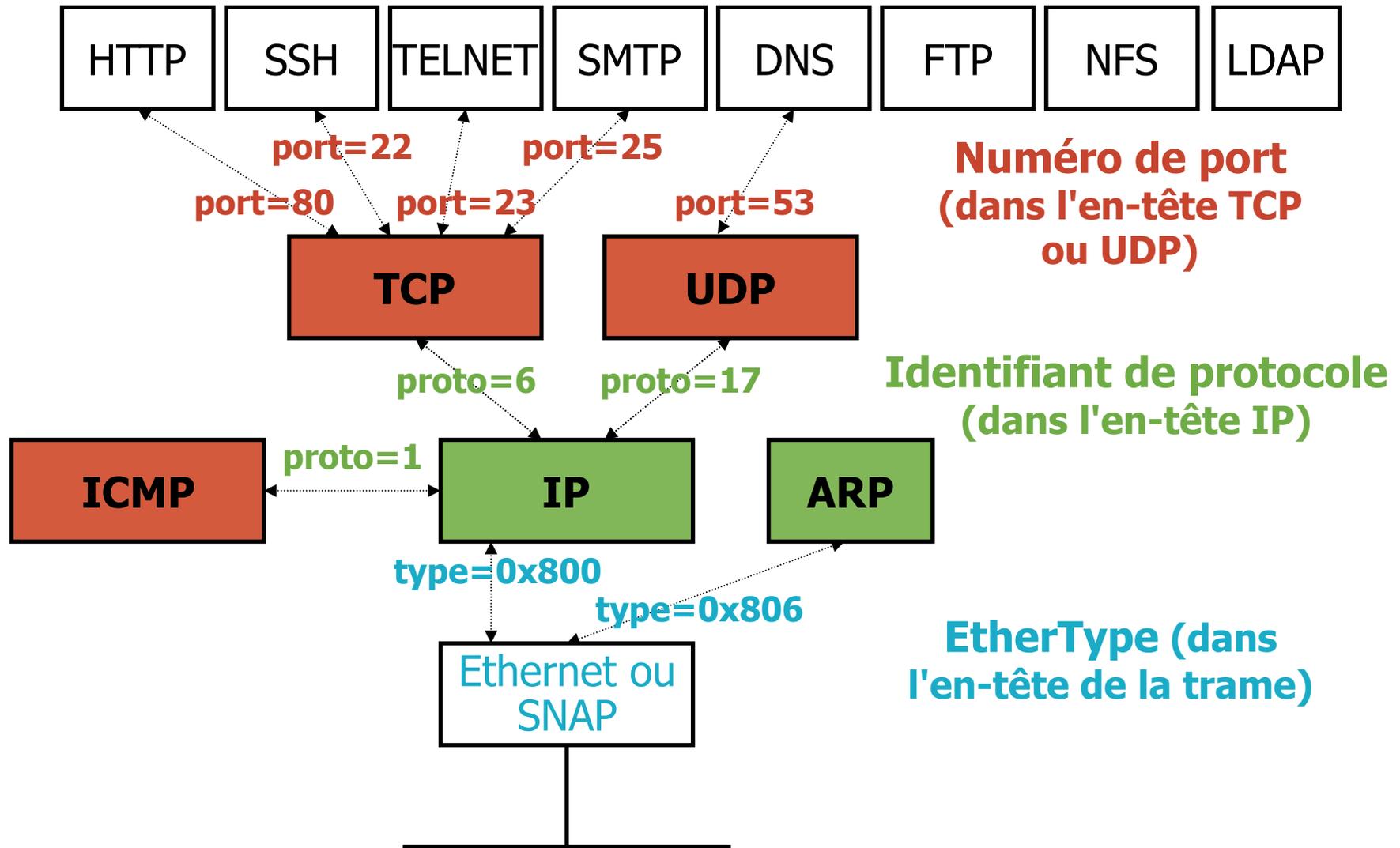
Nb d'octets que le récepteur peut recevoir

# Format de l'en-tête IPv4

- Un en-tête de 20 octets + une partie facultative de longueur variable (options)



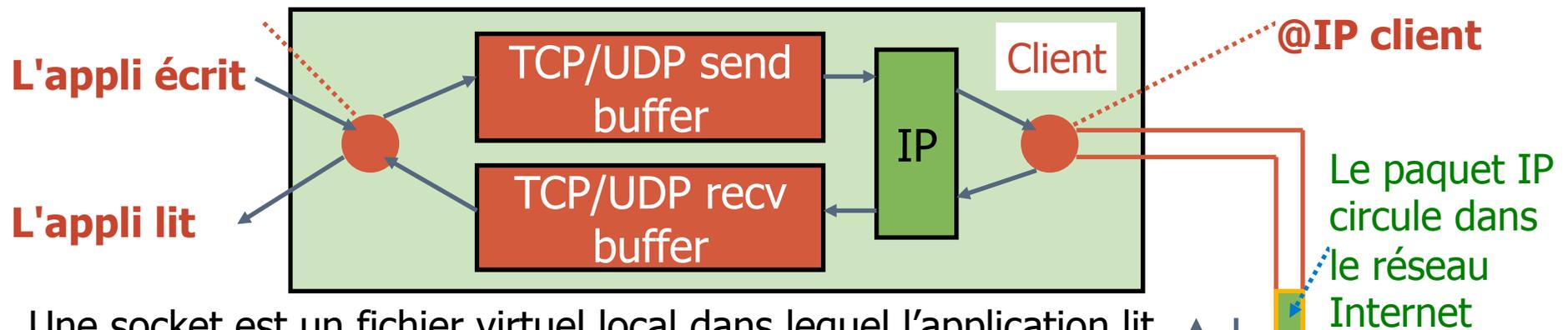
# La désencapsulation : identifier la couche >



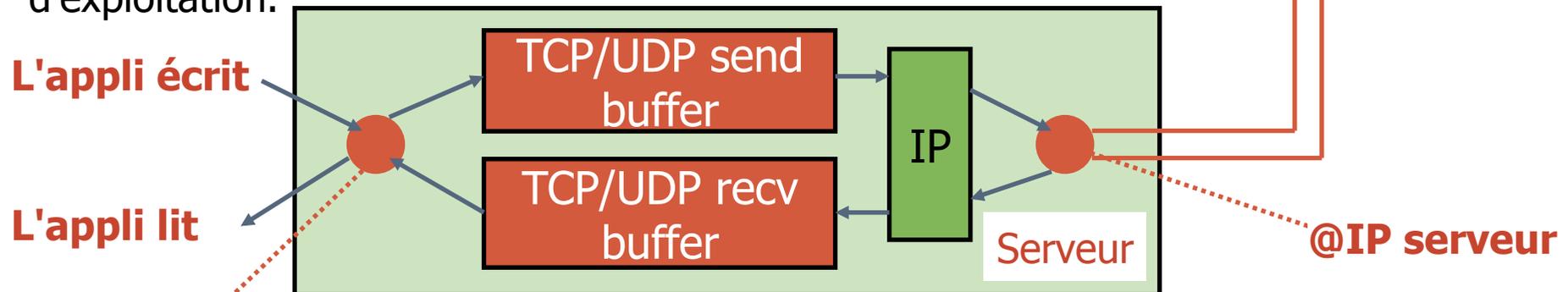
# Les sockets et les numéros de port

- Un échange Client/Serveur = (@IP\_src,port\_src,@IP\_dest,port\_dest)

## Port 5004 utilisé par le navigateur web



Une socket est un fichier virtuel local dans lequel l'application lit ou écrit. Elle est associée à un numéro de port local et des zones d'émission/réception de TCP ou UDP attribués par le système d'exploitation.



## Port 80 utilisé par le serveur web

# A la fin du module !

