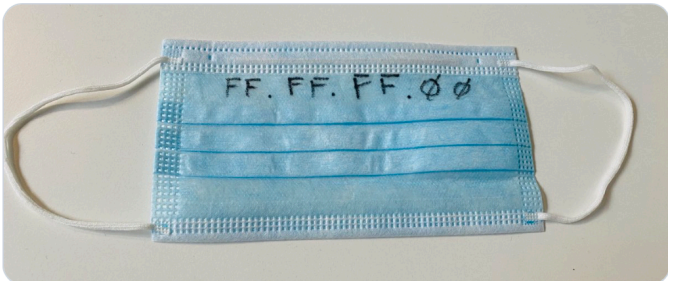
Masque de sous-réseau.



Mif05
Réseaux

Isabelle Guérin Lassous
isabelle.guerin-lassous@univ-lyon1.fr
http://perso.univ-lyon1.fr/isabelle.guerin-lassous

Équipe enseignante de l'UE

- Responsable : Isabelle Guérin Lassous

- Intervenants : Thomas Begin, Loïc Desgeorges, Florent Dupont, Élise Jeanneau, Augustin Laouar

Organisation de l'UE

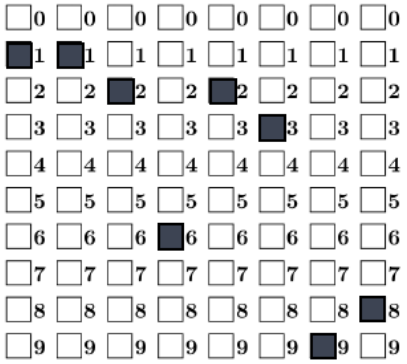
- 12h CM ; 12h TD ; 6h TP + 1 projet en autonomie
  - complémentarité CM/TD/TP/projet
- Principalement dans les semaines de présence des alternants
  - sauf pour certains TPs pour des groupes de non alternants
- Très important de régulièrement consulter
  - vos emails
  - la page web du cours
    - http://perso.ens-lyon.fr/isabelle.guerin-lassous/index-M1if05.htm

Évaluation de l'UE

- 3 ECTS
- Contrôle Continu Intégral
- Note finale est la moyenne de
  - 1 épreuve commune anonyme (ECA)
    - 90 minutes
    - 40%
    - session 2 (60 min)
  - 4 contrôles lors des TDs
    - QCM
    - portant sur les CM et sur le projet en autonomie
    - 14% pour chaque contrôle
  - 1 note de TP : 4%

QCM

- Barème
  - 1 seule réponse juste
  - réponse juste : 1 pt
  - réponse fausse : 0 pt ou -1 ou -0,5 pt (dépendra du niveau du contrôle)
  - pas de réponse : 0 pt
- Réponses
  - noircir les cases intégralement avec un stylo noir
  - exemple ■
  - contre-exemple ☒ ▣
- Numéro étudiant
  - exemple : 1126 2398



Planning de l'UE

Date	Jour	Horaire	Semaine	Durée	Activités	Groupes
02-sept	Lundi	9h45	Avec Alt	3	CM1	
09-sept	Lundi	8h	Avec Alt	3	CM2	
30-sept	Lundi	8h	Avec Alt	1,5	CM3	
30-sept	Lundi	9h45	Avec Alt	3	TD1	A,B,C,D
07-oct	Lundi	9h45	Avec Alt	3	CM4	
28-oct	Lundi	9h45	Avec Alt	3	TD2	A,B,C,D
04-nov	Lundi	8h	Avec Alt	1,5	CM5	
04-nov	Lundi	9h45	Avec Alt	3	TD3	A,B,C,D
25-nov	Lundi	9h45	Avec Alt	3	TD4	A,B,C,D
02-déc	Lundi	9h45	Avec Alt	3	TP1	A1,A2,B1,B2
09-déc	Lundi	9h45	Sans Alt	3	TP1	C1,C2,D1,D2
06-janv	Lundi	9h45	Avec Alt	3	TP2	A1,A2,B1,B2
13-janv	Lundi	9h45	Sans Alt	3	TP2	C1,C2,D1,D2
?/01/2025	?	?	Avec Alt	1,5	Examen	

Planning des CM

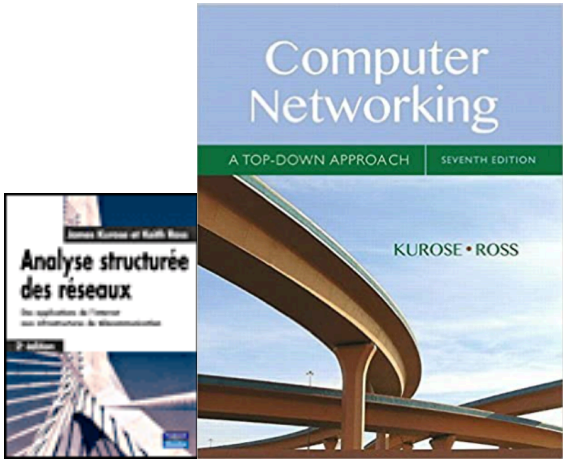
Id	Durée	Contenu	Enseignant
CM1	3	Introduction & Application, puis Transport	IGL
CM2	3	Transport & Réseau & Liaison de données	IGL
CM3	1,5	Wi-Fi	IGL
CM4 & CM5	4,5	Physique	FD
	12		

Pré-requis & Positionnement de l'UE

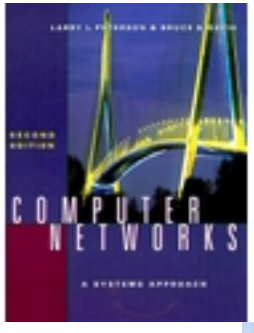
- Pré-requis
  - Connaissances en réseaux de niveau L3
  - Objectifs & principes généraux des réseaux
    - principaux éléments de l'Internet
    - ce qu'est un protocole
    - architecture en couches
    - objectifs des couches transport, réseau et liaison de données & principaux protocoles associés
- Positionnement de l'UE par rapport à LIFRES (UE Réseaux du L3 Informatique de l'UCBL)
  - Rappels des principes indispensables à connaître pour tout étudiant de Master Informatique
  - Nouvelles notions sur
    - couche Physique
    - Wi-Fi
  - Vision centrée performances et résolument focalisée sur ce qui existe dans l'Internet
  - Meilleure maîtrise de l'articulation des différents protocoles

Pour se mettre à niveau ou en savoir plus…

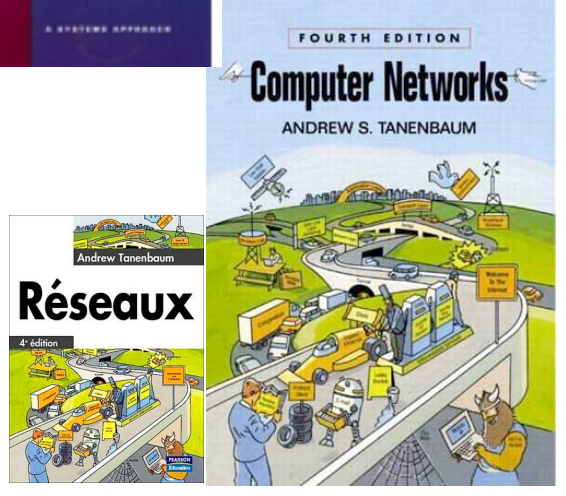
- ***Computer Networking - A Top-Down Approach Featuring the Internet***, 7th Edition, J. Kurose and K. Ross, Pearson, 2020.
  - *Analyse structurée des réseaux , Des applications de l'Internet aux infrastructures de télécommunication*, 2nde Edition, 2003.
  - Versions anglaise et française disponibles à la bibliothèque



- ***Computer Networks: A Systems approach*** , 2nd Edition, L. Peterson and B. Davie, Morgan Kaufmann, 1999.



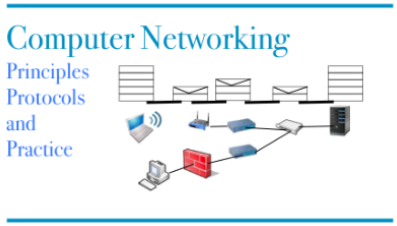
- ***Computer Networks***, 4th Edition, A. Tanenbaum, Prentice Hall, 2003.
  - *Réseaux*, Pearson Education, 2003.



- The Good Warriors of the Net - IP for Peace https://www.youtube.com/watch?time_continue=673&v=x9XWxD6cJuY

Pour se mettre à niveau ou en savoir plus…

- **Computer Networking : Principles, Protocols and**
  - open-source ebook
  - livre, ressources pédagogiques, énoncés exercices / TP
  - https://www.computer-networking.info



- Page web du cours LIFRES (L3)
  - https://perso.univ-lyon1.fr/olivier.gluck/supports_enseig.html

Pastilles de couleur

- Notions indispensables à maîtriser
- Notions à savoir en M1
- Notions avancées ou culturelles dont la connaissance est un plus

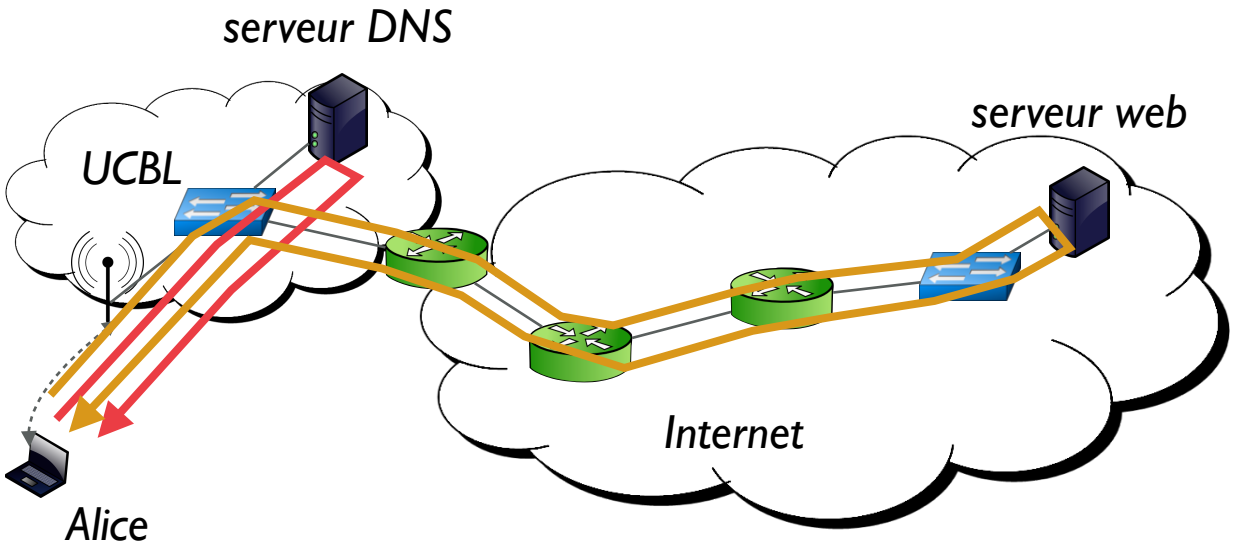
Introduction

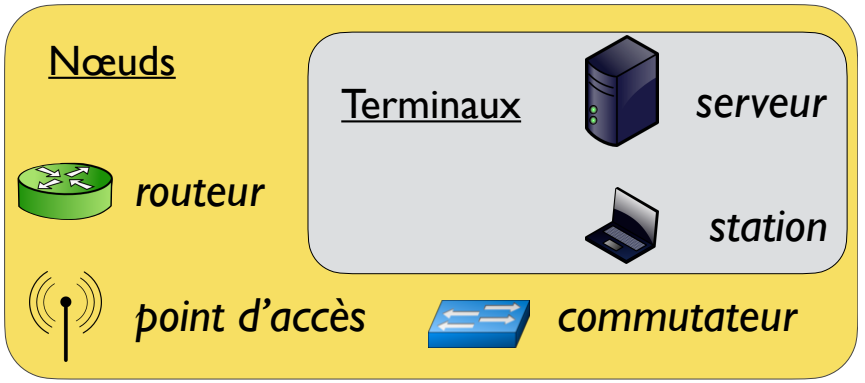
Rappels & Vocabulaire

Plan

1. Première présentation par l'exemple
2. Internet, paramètres de performance
3. Retards et pertes dans les réseaux à commutation de paquets
4. Architecture en couches

Exemple 1





Alice https://www.youtube.com/watch?v=9Y29TXdrBM4

Résolution DNS : 172.217.171.238

Requête / réponse http :

Premières conclusions :

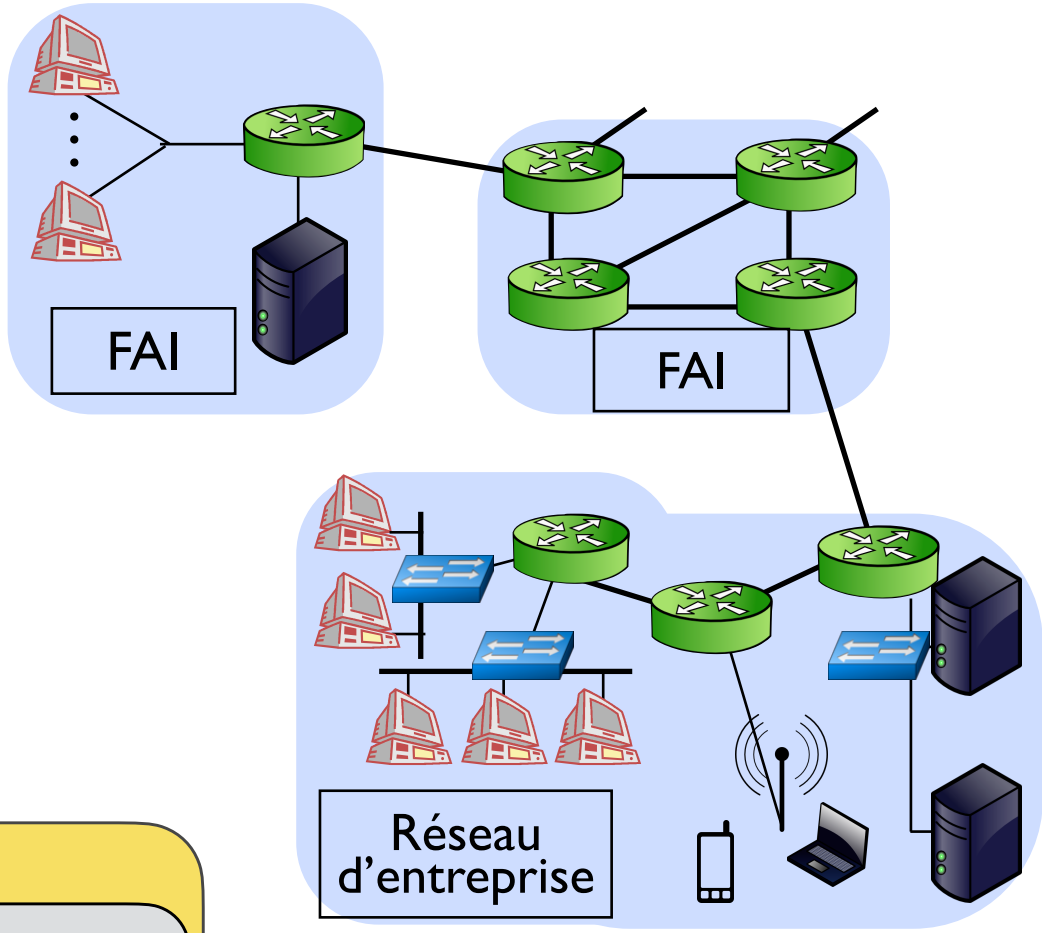
- Nombreuses applications reposent sur des réseaux
- Approche Client / Serveur
- Nœuds ont différents rôles
- Nœuds doivent être identifiables : adresses IP
- Système pour acheminer les données

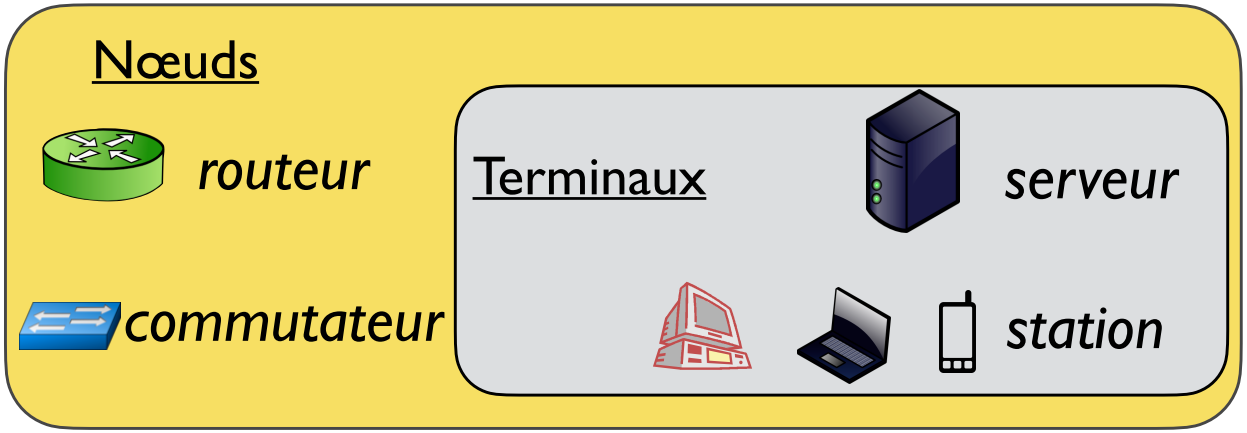
Plan

1. Première présentation par l'exemple
2. Internet, paramètres de performance
3. Retards et pertes dans les réseaux à commutation de paquets
4. Architecture en couches

Internet

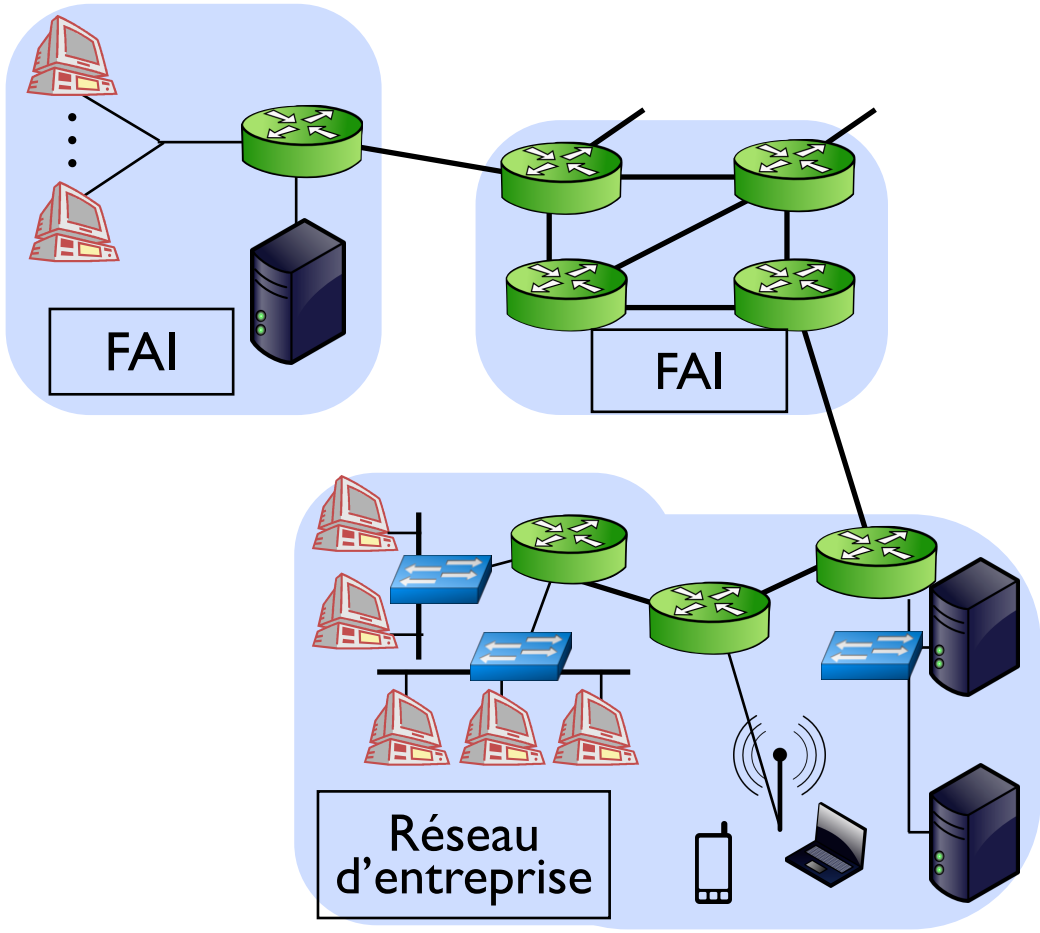
- Millions de nœuds (équipements) inter-connectés
  - exécutant des applications
  - générant et acheminant des paquets
  - formant une infrastructure de communication
- Terminaux : générer ou recevoir les paquets
- Routeurs / commutateurs : acheminer les paquets





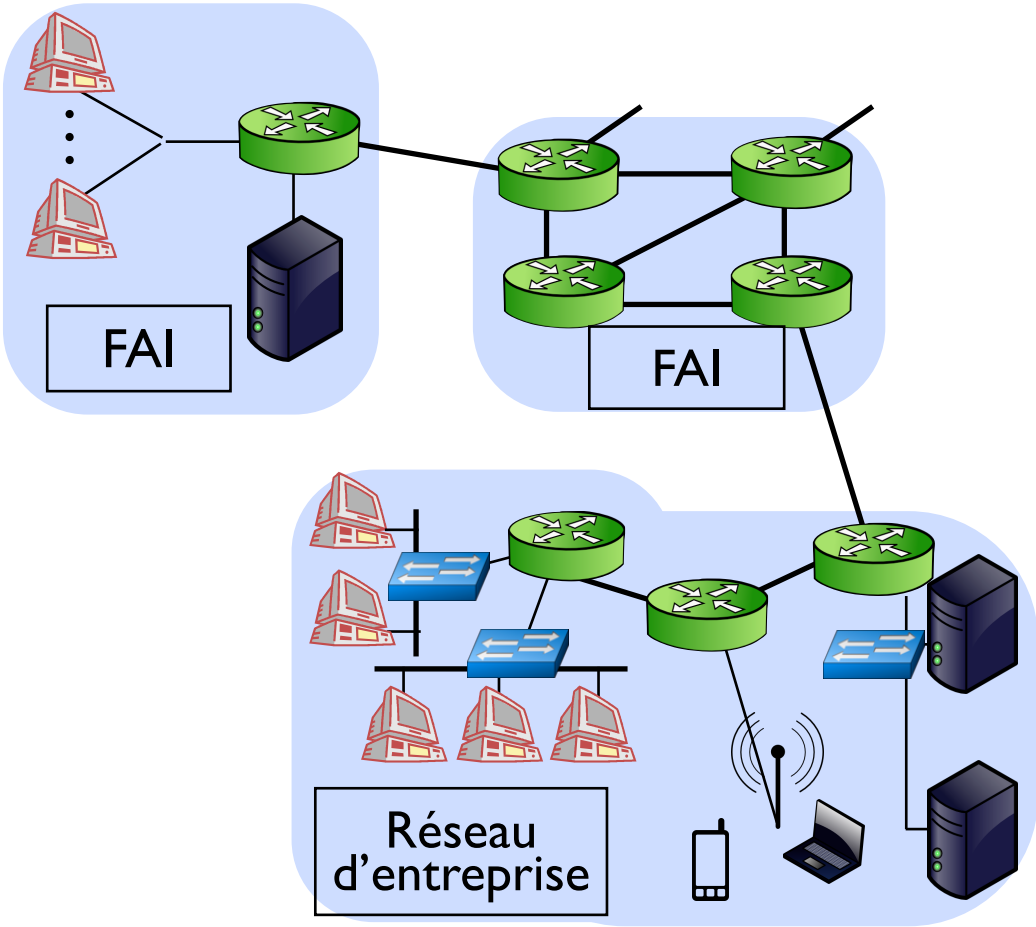
Liens de communication & performances associées

- Liens de communication
  - très hétérogènes
    - fibre, cuivre, radio, satellite…
  - 3 caractéristiques majeures
  - capacité d'émission, débit d'émission ou débit physique
    - unité ? Mb/s ou Gb/s
  - délai de propagation ou latence
    - unité ? ms ou µs
  - taux d'erreur binaire
    - unité ? en %



Protocoles

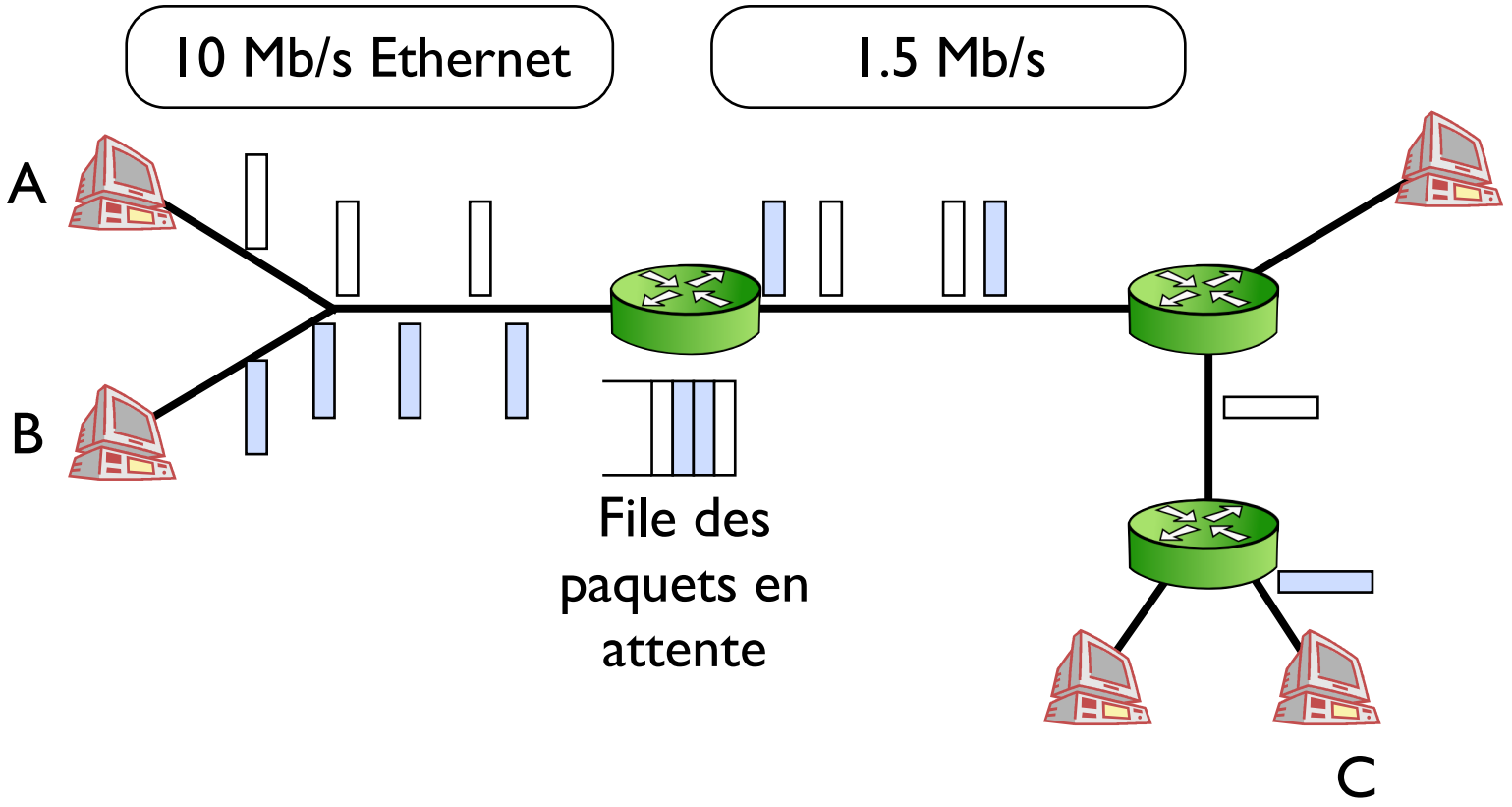
- Les protocoles contrôlent l'émission et la réception des messages délivrés par les applications
  - Exemples : IP, TCP, HTTP, FTP
- Internet : le réseau des réseaux
  - Interconnexion de multiples réseaux
  - Hiérarchie souple
  - Internet public vs intranet privé



Commutation de paquets (1)

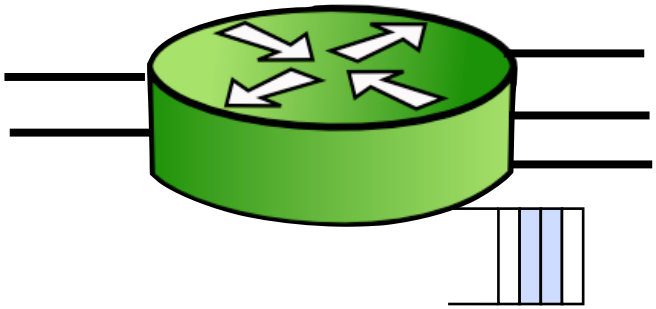
- Commutation de paquets
  - Divise les messages échangés entre les applications en paquets
  - Les paquets se partagent les ressources à tour de rôle
    - Lors de son émission, un paquet prend toute la capacité du lien
    - Aucune réservation
  - Lorsqu'un lien emprunté est congestionné → attente dans un buffer → délais supplémentaires
  - Internet fait de son mieux mais aucune garantie de service
    - "Best effort" 

Commutation de paquets (2)



- Séquence des paquets de A et B sur le lien 1.5 Mb/s n'a pas un motif régulier
- Ressources du lien 1.5 Mb/s sont partagées à la demande
  - Multiplexage statistique

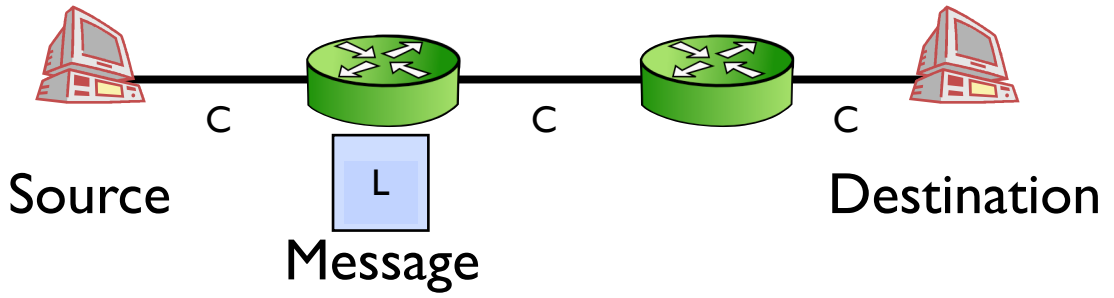
Routeurs



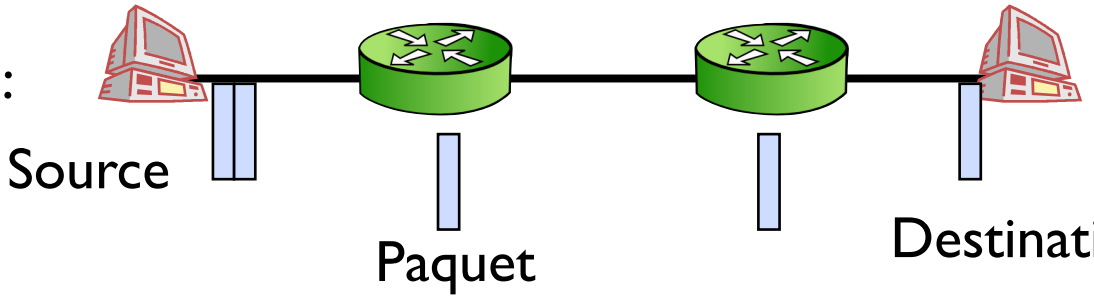
- Routeurs = Commutateurs de paquets
- Plusieurs ports d'entrée et de sortie
- En réception (entrée)
  - Mode "store-and-forward"
    - Un paquet doit être entièrement reçu avant d'être ré-émis au noeud suivant
    - ≠ "cut-through"
- En émission (sortie)
  - Les paquets attendent dans un buffer si le port est occupé par un autre paquet

Fragmentation en paquets

Sans fragmentation :



Avec fragmentation :



- Hypothèses
  - Taille du message, L = 7.5 Mbits
  - Capacité d'émission des liens, C = 1.5 Mb/s
  - Sans congestion
- Temps de transfert
  - sans fragmentation ? $T_{transfert} = 3L/C = 15$ s
  - avec 5 fragments ? $T_{transfert} = 7L/(5C) = 7$ s
- La fragmentation peut accélérer le temps de transfert des messages (effet "Pipeline")
  - Toujours mieux ?

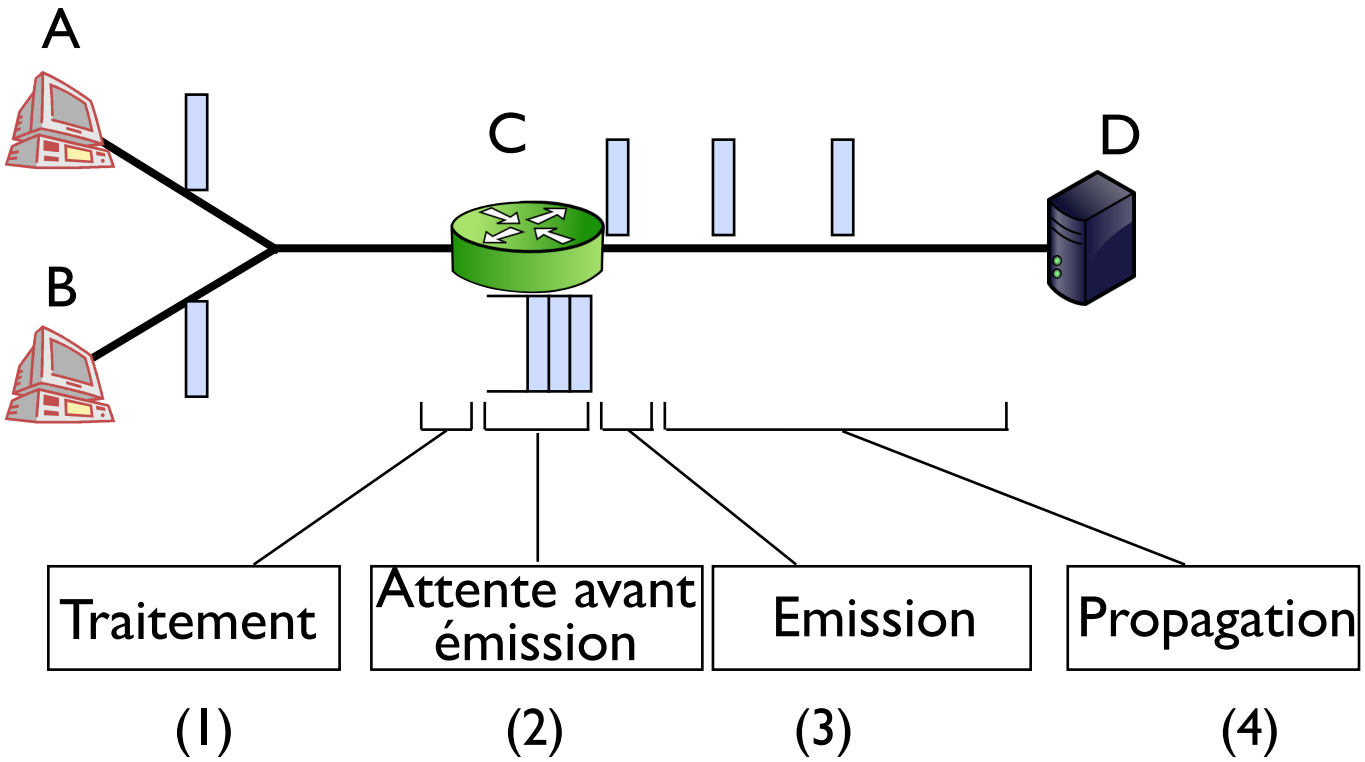
Plan

1. Première présentation par l'exemple
2. Internet, paramètres de performance
3. Retards et pertes dans les réseaux à commutation de paquets
4. Architecture en couches

Délais et pertes dans les réseaux à commutation de paquet

- Un paquet part d'une source, traverse des liens et des routeurs et atteint sa destination
- Il subit plusieurs types de délais…
  - Lorsque les buffers se remplissent … les délais d'attente s'allongent
- ... et peut également être perdu
  - Dépassement de capacité des buffers
    - Lorsque les buffers sont pleins ... des pertes surviennent
  - Erreur lors de la transmission sur un lien
    - Surtout sur les liens radio / sans-fil
- ... ou arriver trop tard

Types de délais (1)

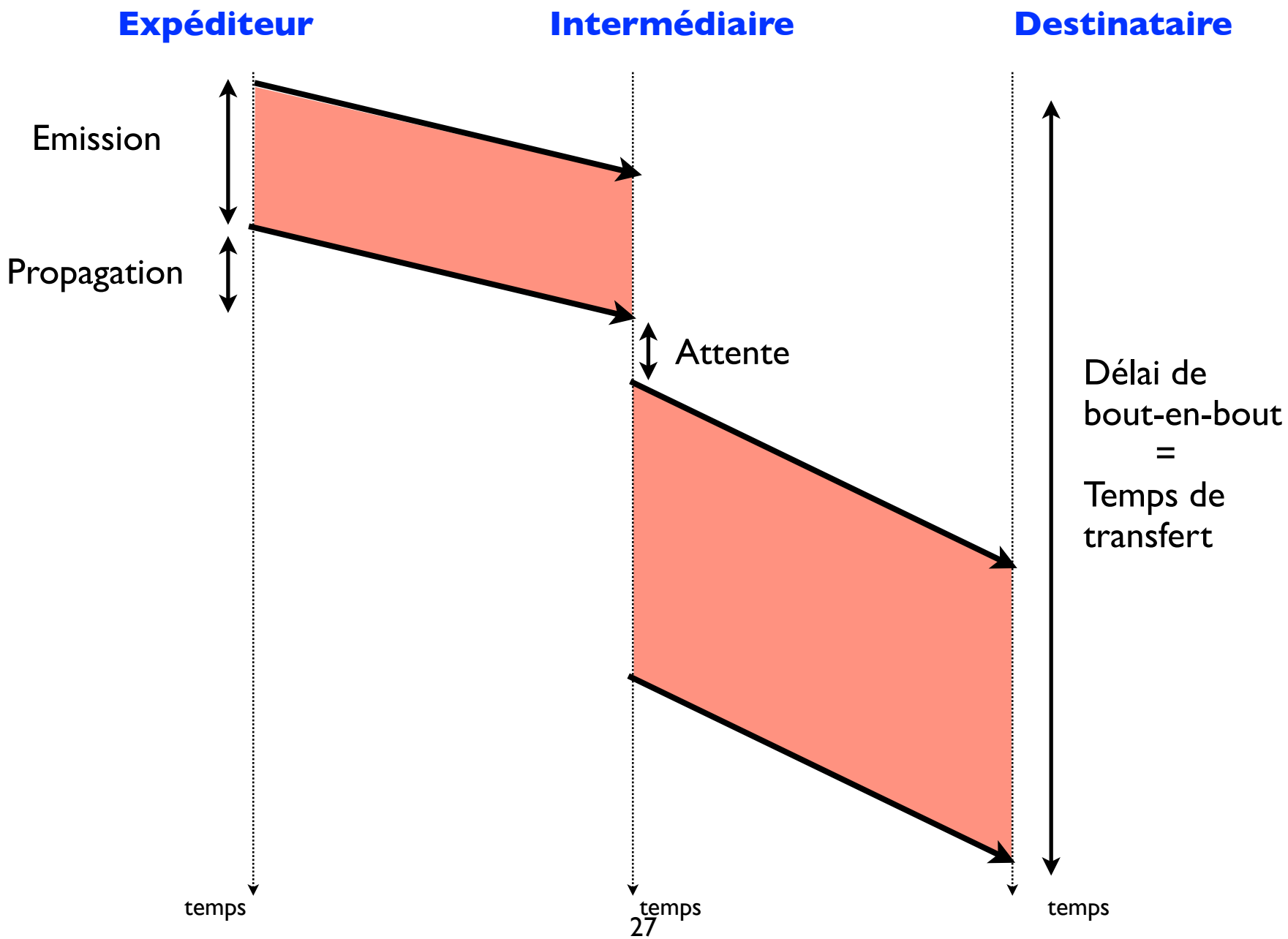


Types de délais (2)

- Traitement
  - lecture en-tête & contrôle erreur-bits
  - détermine l'interface de sortie
  - ~ns
- Attente
  - si le lien est occupé
  - dépend du nombre de paquets déjà dans la file
  - très variable, [0, x] ms
- Émission
  - temps pour envoyer le paquet
  - L/C avec taille du paquet L et un lien de capacité C (hypothèse couche Liaison parfaite)
  -  : si lien partagé, penser aux surcoûts de la couche Liaison
  - ~[µs, ms]
- Propagation
  - temps pour qu'un bit se propage sur le lien
  - d/v avec longueur du lien d et vitesse de propagation v
  - ~ [µs, ms]

C et v sont des quantités différentes !!!!

Types de délais (3)



RTT

- RTT - Round Trip Time (Temps Aller-Retour)
  - délai pour qu'un paquet atteigne sa destination et revienne à la source
  - comprend les temps de traitement, d'attente, d'émission et de propagation
  - mesure simple avec la commande *ping*

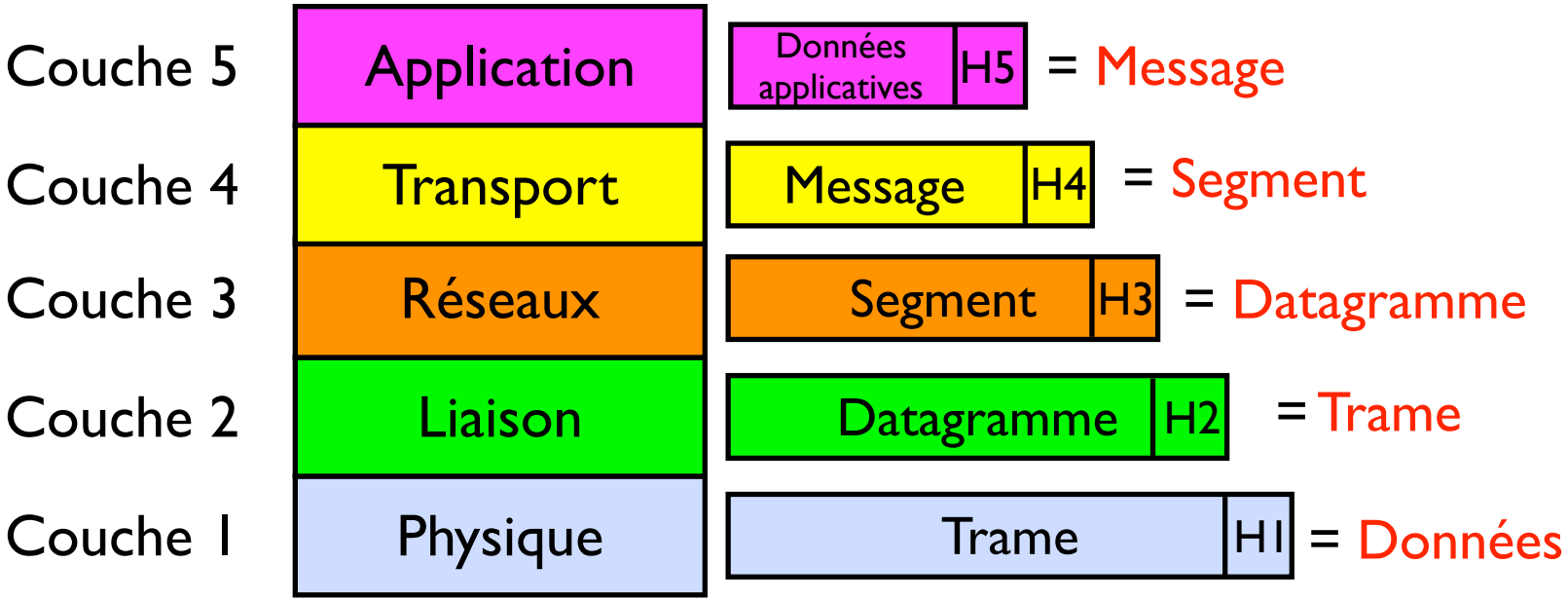
Plan

1. Première présentation par l'exemple
2. Internet, paramètres de performance
3. Retards et pertes dans les réseaux à commutation de paquets
4. Architecture en couches

La couche protocolaire Internet

- 5 couches (7 dans le modèle OSI)
- Implantées en "Software" ou "Hardware" ou les deux
- Paquet de couche n est encapsulé dans un paquet de couche n-1

➡ [Paquet couche n-1] = [En-tête couche n-1] + [Paquet couche n]

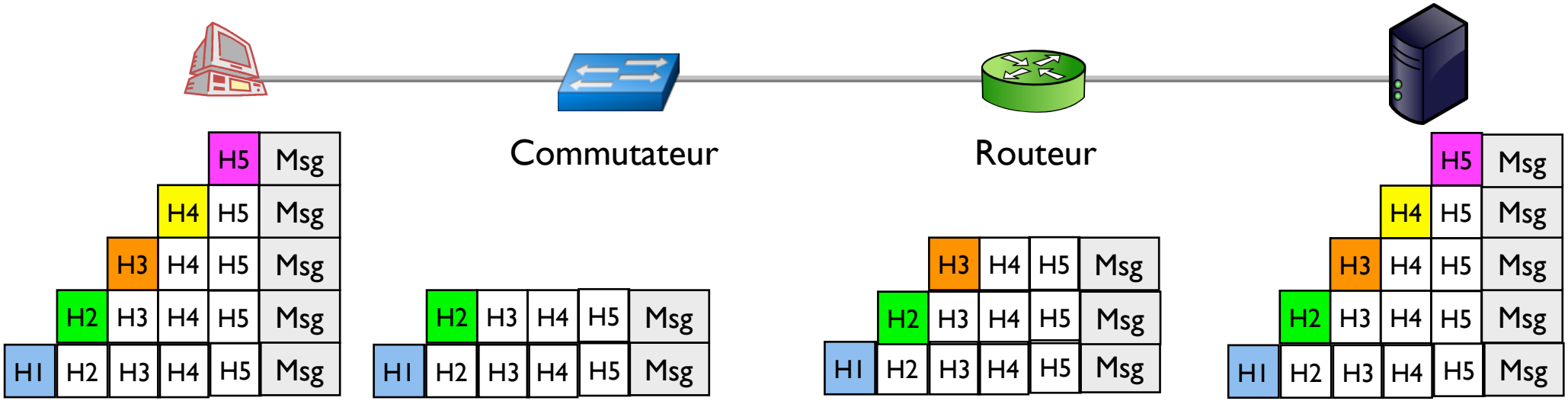


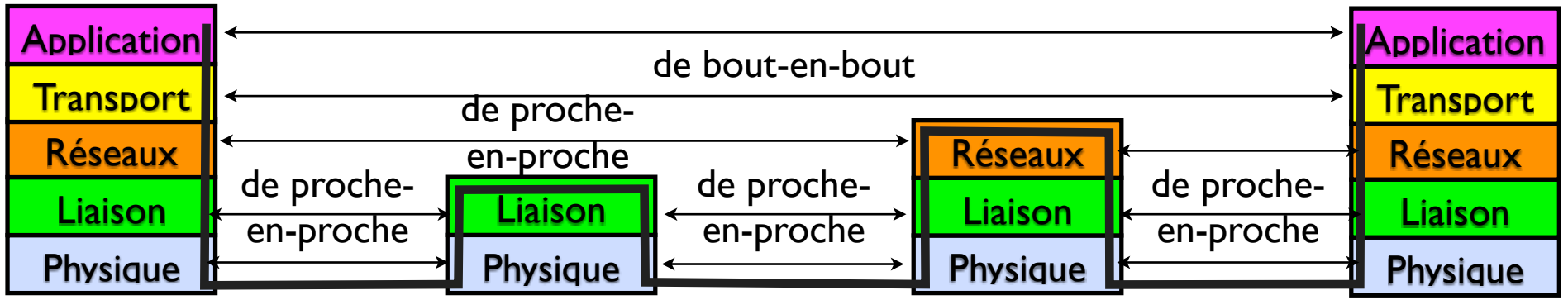
- Exemples : D3 = H4 + D4 ou D3 = H4 + H5 + D5
  (D3 = données couche 3)

Architecture en couches

- Chaque couche a une fonction
- Chaque protocole appartient à une couche
- ∑ couches = pile protocolaire
- La couche n-1 assure un service à la couche n
  - Ex : contrôle d'erreur, acheminement des données, accès au médium, fragmentation/reconstitution …
- Avantages
  - Identification précise d'un protocole
  - Modularité dans l'implémentation d'une couche
    - il suffit de conserver les services rendus à la couche n + 1
- Inconvénients
  - Redondance de fonctions (détection d'erreur)
  - Impossibilité d'échanger avec couches n±2 ("cross-layer")
  - Nécessité de traverser toutes les couches

Equipements et couches





Couches de proche-en-proche ? 1, 2 et 3

Couches de bout-en-bout ? 4 et 5

Couche Application

Pas de réseaux sans applications

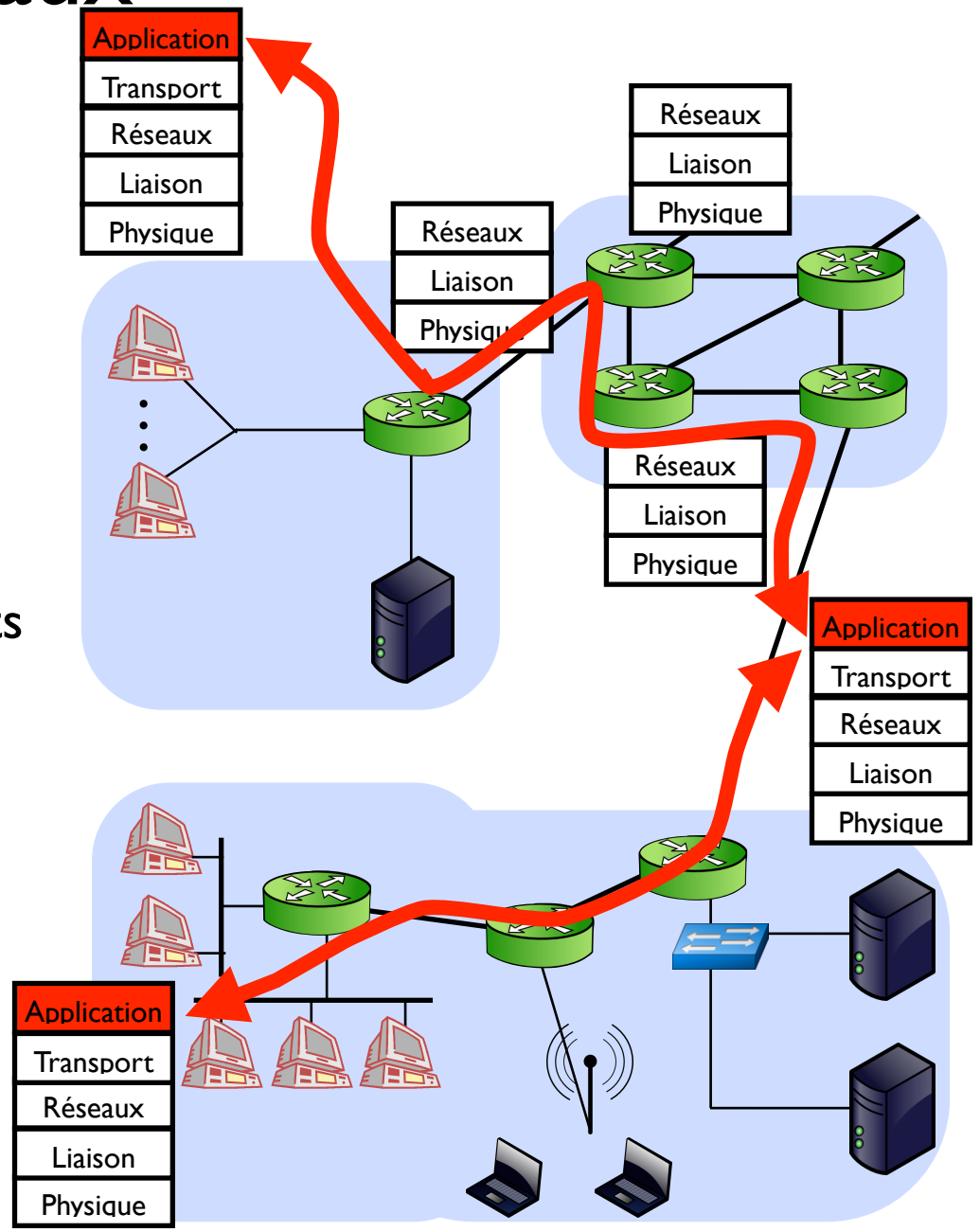
- Applications nombreuses et variées
- Applications « classiques »
  - e-mails
  - accès à une machine distante
  - transfer de fichiers
  - Web
  - commerce électronique
  - messagerie instantanée
  - réseaux sociaux
  - vidéo-conférence
  - streaming vidéo
  - jeux en réseau
  - …
- Applications « réseaux »
  - applications qui permettent aux réseaux et aux applications reposant sur ces réseaux de fonctionner

Plan

1. Principes des applications utilisant les réseaux
2. Web et HTTP
3. Applications pour faire fonctionner les réseaux

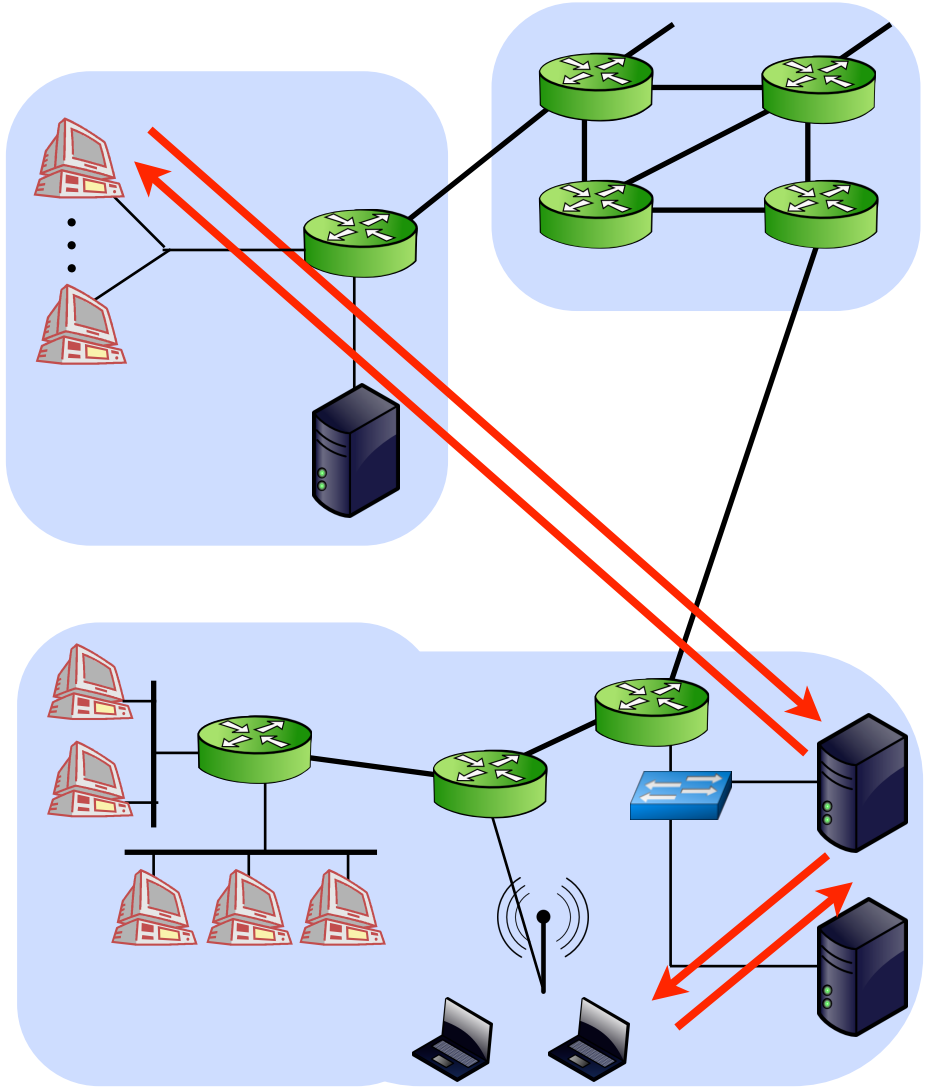
Créer une application utilisant les réseaux

- Une application repose sur des programmes
- Les programmes sont
  - exécutés sur des terminaux distants
  - et communiquent par le réseau
- Peu de programmes sur les équipements du cœur de réseau
  - Pas d'utilisateurs
  - *"Conserver le cœur simple"*
- Différentes architectures possibles
  - Client-serveur, P2P, hybride



Architecture client-serveur

- Un Serveur
  - Toujours actif
  - Adresse IP permanente
  - Cluster (grappe) de serveurs / Data center pour passage à l'échelle / disponibilité
- Des Clients
  - Communiquent avec le serveur
  - Activité intermittente
  - Adresse IP peut être dynamique
  - Ne communiquent pas directement entre eux

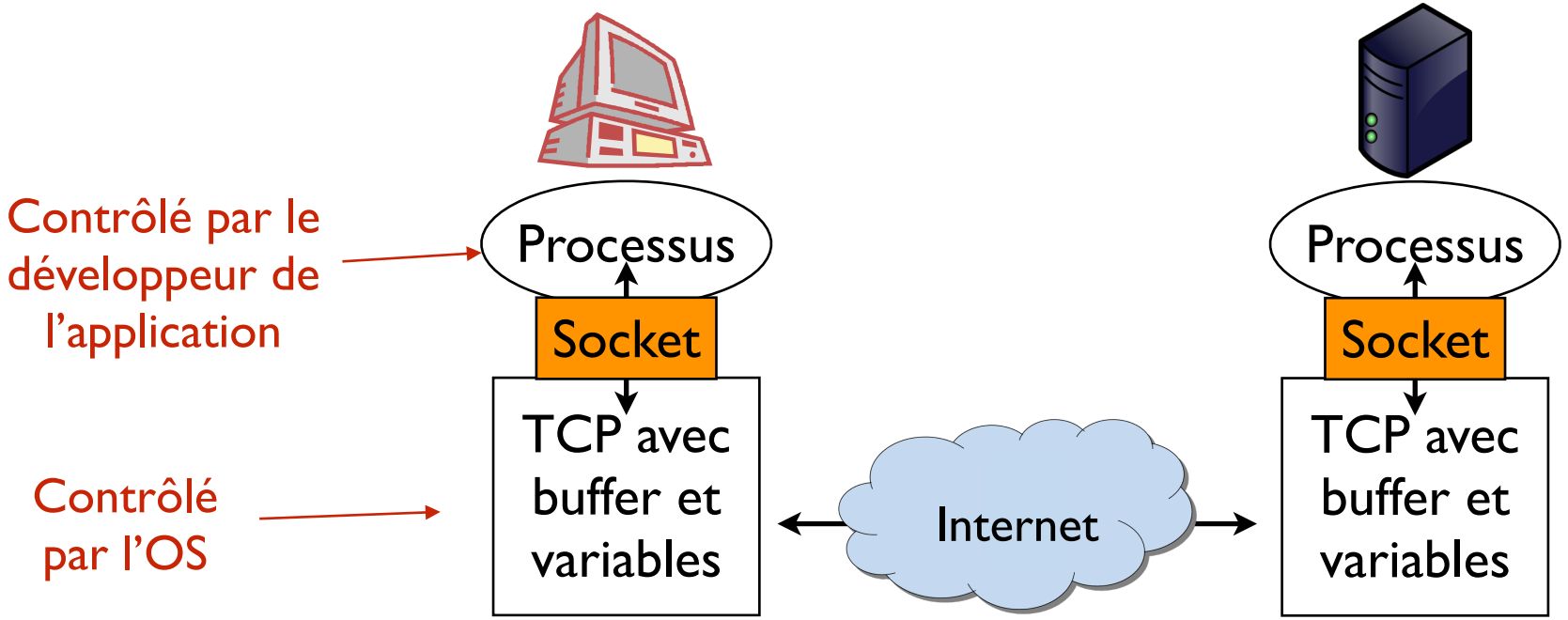


Comment les applications communiquent entre elles ?

- Processus : une application (programme) qui s'exécute sur un nœud
- Sur des nœuds différents
  - deux processus communiquent par des échanges de messages
  - message = entité transportant les informations d'une application
- Processus client : initie la communication et sollicite le serveur
- Processus serveur : attend d'être sollicité

Comment les processus communiquent entre eux ?

- Chaque processus ouvre une socket
- Une socket
  - Interface de communication pour les applications distribuées - API
  - "Zone de départ et d'arrivées" des messages
  - Paramètres côté Application : paramètres de l'application, choix du protocole de Transport, taille du buffer, …
    - à la discrétion du programmeur
- Les deux sockets sont ensuite raccordées grâce au protocole de transport choisi



Comment identifier un processus sur un nœud distant ?

- Pour recevoir des messages, un processus doit être identifiable
- Est-ce que l'adresse IP du nœud sur lequel le processus est exécuté suffit à identifier le processus ?
  - Non, car plusieurs processus peuvent s'exécuter sur un même noeud
  - L'identifiant comprend l'adresse IP ET un numéro de port associé au processus
- Exemples de numéros de port
  - Serveur HTTP : 80
  - Serveur E-mail : 25
- Serveur HTTP de www.lequipe.fr
  - adresse IP : 160.92.167.203
  - numéro de port : 80

```
nc -z www.lequipe.fr 1-65535
• Connection to www.lequipe.fr port 80
  [tcp/http] succeeded!
• Connection to www.lequipe.fr port 443
  [tcp/https] succeeded!
```

Protocoles de la couche Application

- Protocole applicatif définit
  - types de messages échangés
  - syntaxe des messages
  - signification des champs
  - règles d'envoi et de réception des messages
- De nombreux protocoles applicatifs

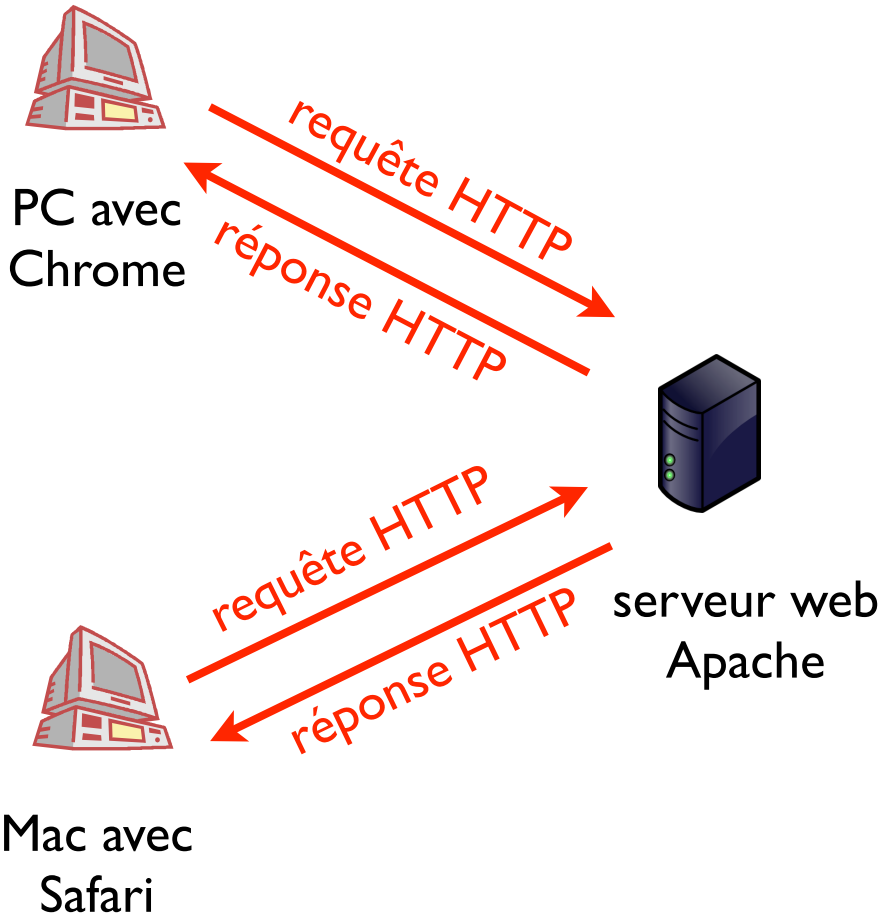
Application	Protocole Applicatif
e-mail	SMTP [RFC 2821]
Accès terminal à distance	Telnet [RFC 854]
Page web	HTTP [RFC 2616]
Transfert de fichier	FTP [RFC 959]
Streaming Audio/Vidéo	HTTP (ex Youtube), RTP [RFC 1889]
Téléphonie Internet	SIP, RTP, privé (ex Skype)

Plan

1. Principes des applications utilisant les réseaux
2. Web et HTTP
3. Applications pour faire fonctionner les réseaux

Généralités sur HTTP (1/2)

- HTTP : HyperText Transfer Protocol
  - Protocole Application pour le web
- Modèle client / serveur
  - Client : navigateur qui demande, reçoit et affiche des objets web
    - Chrome, Firefox, Edge, …
  - Serveur : serveur web qui envoie des objets en réponse à des requêtes
    - Apache, IIS, Nginx…
  - 2 types de message HTTP : requêtes et réponses au format ASCII donc "lisibles"
- HTTP repose (majoritairement) sur TCP
- Port 80
- Protocole sans état
  - Serveur ne maintient aucune information sur les requêtes passées



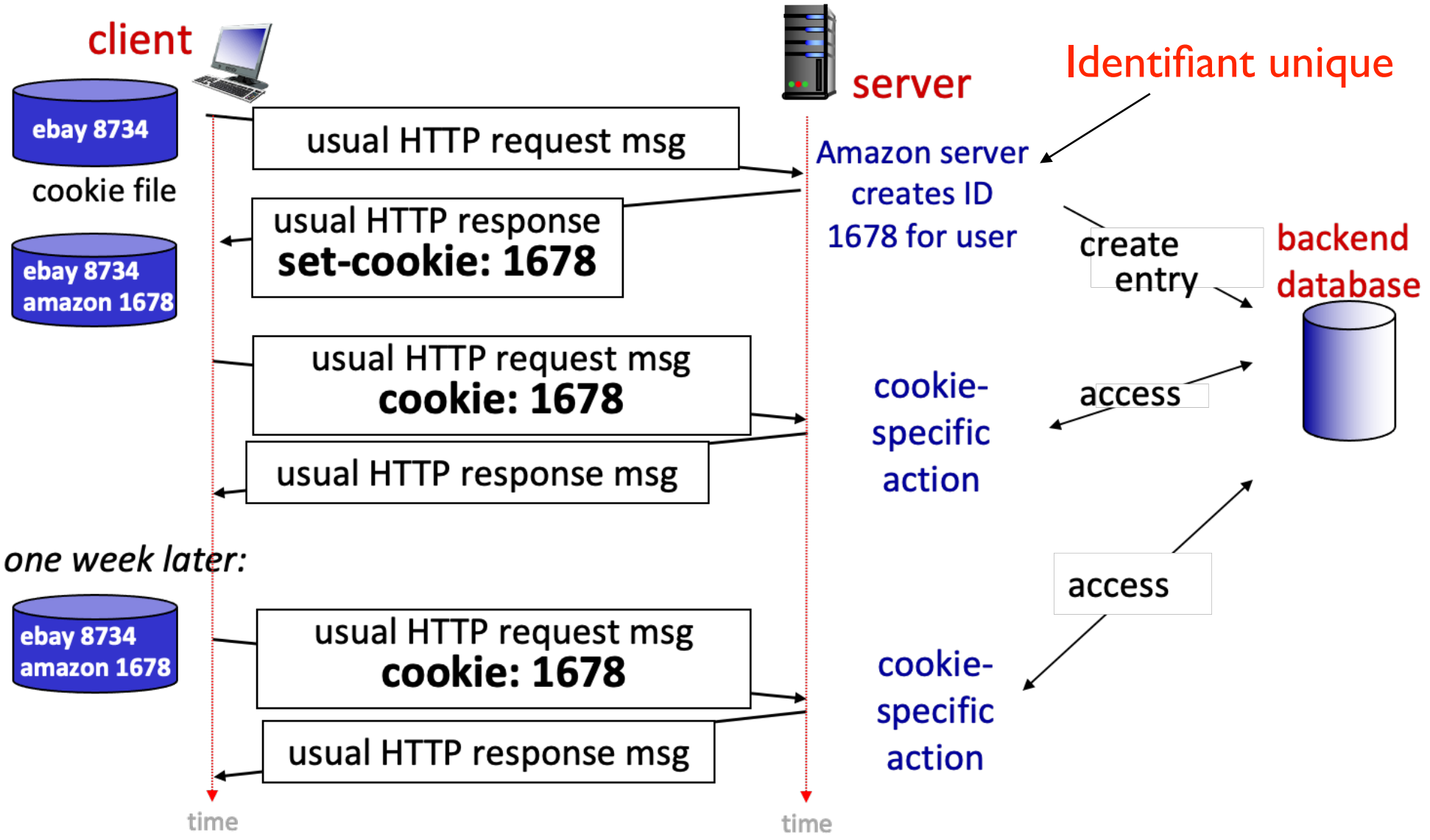
Généralités sur HTTP (2/2)

- HTTP/1.0 et HTTP/1.1
  - RFC 7231
  - Introduit en 1991 et publié en 1996
  - Connexion TCP non-persistante (1.0)
  - Connexion TCP persistante (1.1)
- HTTP/2.0
  - Publié en 2015
  - Réduction du délai de bout-en-bout avec des connexions multiples
- HTTP/3.0
  - Publié en 2018
  - Repose sur QUIC / UDP

Utilisation des cookies (1/2)

- HTTP sans état sur le serveur
  - permet de gérer beaucoup de connexions en parallèle
- Cookies : permet, aux serveurs, de garder une trace des utilisateurs
  - très utiles pour les transactions
- 4 composants dans l'utilisation d'un cookie
  - champ dans l'en-tête de réponses HTTP
  - champ dans l'en-tête de requêtes HTTP
  - fichier stocké sur le client et géré par le navigateur
  - base de données associée au serveur Web
- Utilisés pour
  - autorisations / paniers d'achat / recommandations / …
- Soulèvent des problèmes de vie privée

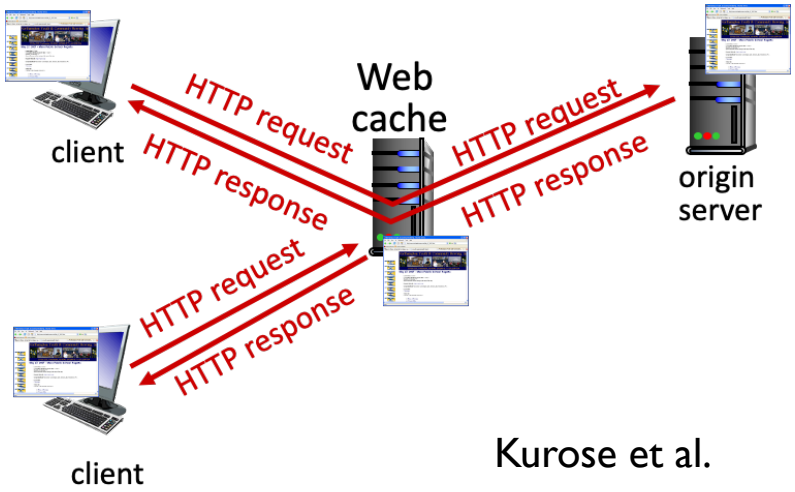
Utilisation des cookies (2/2)



Kurose et al.

Caches Web

- Cache Web : peut traiter les requêtes HTTP à la place du serveur Web d'origine
  - Proxy Web
- Comprend
  - son propre espace de stockage
  - copies d'objets déjà demandés
- Client et serveur à la fois
- Avantages
  - Réduire le temps de réponse
  - Réduire le trafic sur les liens d'accès
- Navigateur peut être configuré de telle sorte que ses requêtes soient d'abord envoyées au Cache Web



Kurose et al.

Plan

1. Principes des applications utilisant les réseaux
2. Web et HTTP
3. Applications pour faire fonctionner les réseaux

Applications réseaux

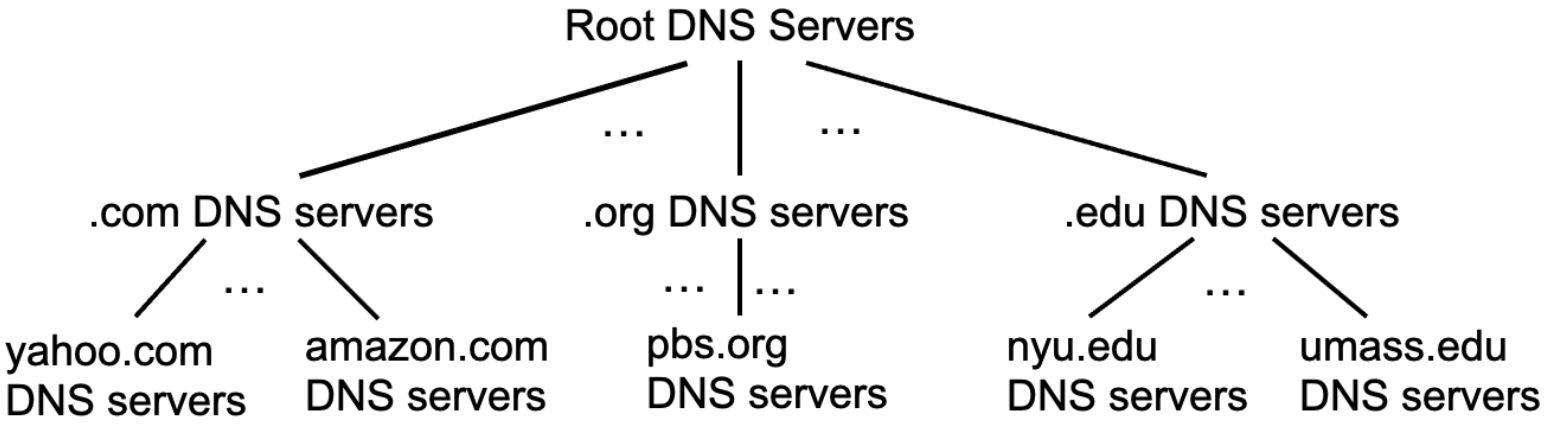
- Plusieurs applications nécessaires ou très utiles au fonctionnement des réseaux
  - DHCP : Dynamic Host Configuration Protocol
  - DNS : Domain Name System
  - NTP : Network Time Protocol
  - SNMP : Simple Network Management Protocol
  - Etc.

DNS : Domain Name System

- Protocole applicatif clé pour le fonctionnement des réseaux
- Comment identifier les nœuds ?
  - adresses IP : utilisées dans l'Internet
  - noms d'hôtes : utilisés par les humains
    - hostnames
    - ex. : www.univ-lyon1.fr
  - Comment faire le lien entre les adresses IP et les noms d'hôtes ?
- Services DNS
  - Traduction entre noms d'hôtes et adresses IP
  - Distribution de charge : pour un nom d'hôte qui a plusieurs serveurs et donc plusieurs adresses IP
- Architecture Client / Serveur
- Repose sur UDP (principalement) ou TCP

DNS : fonctionnement (1/2)

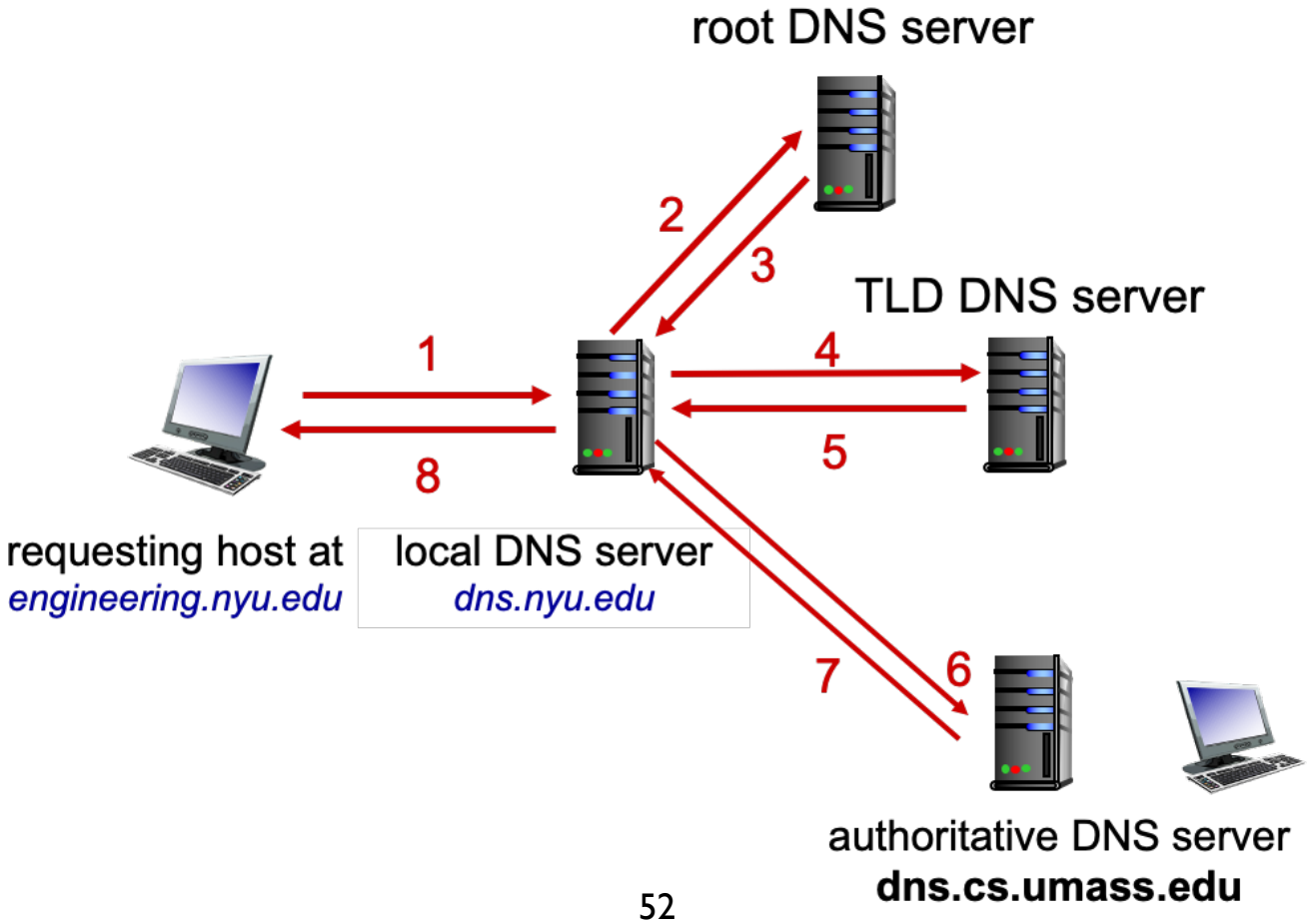
- Un seul serveur DNS pas adapté
  - pas résilient aux pannes
  - ne peut pas traiter un très grand volume de requêtes
  - ne peut pas être « près » de tous les clients
  - ne peut pas gérer une base de données gigantesque
- Ensemble de serveurs organisé de manière distribuée et hiérarchique
  + serveurs DNS locaux



Kurose et al.

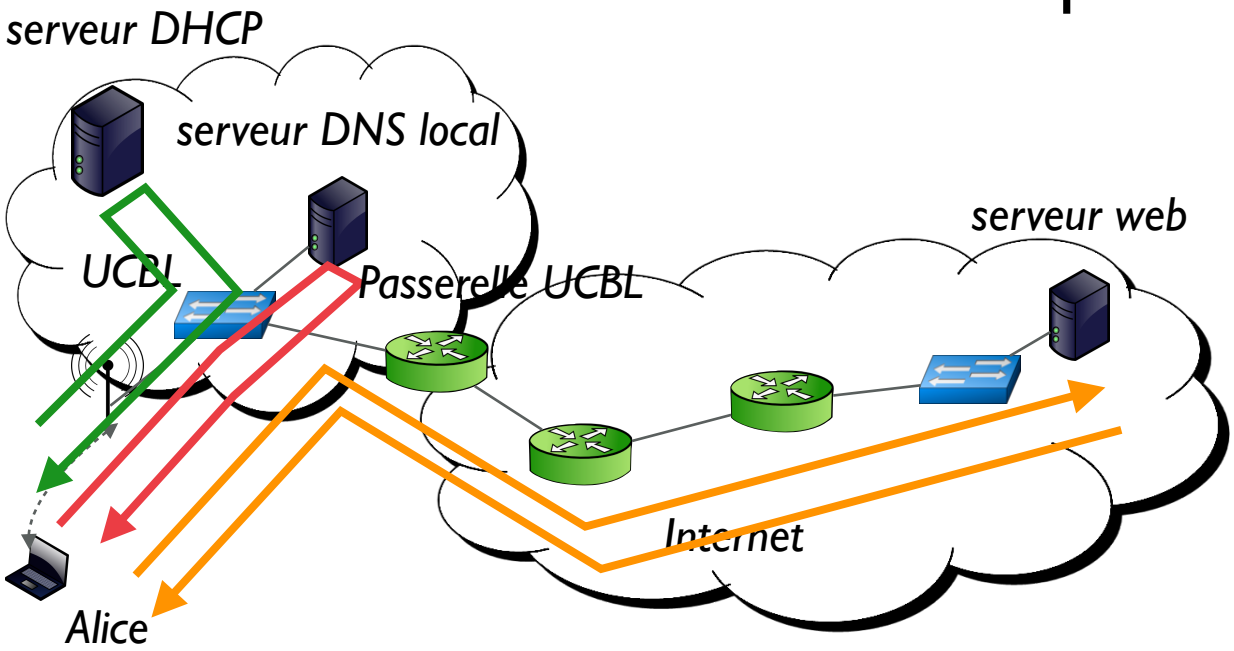
DNS : fonctionnement (2/2)

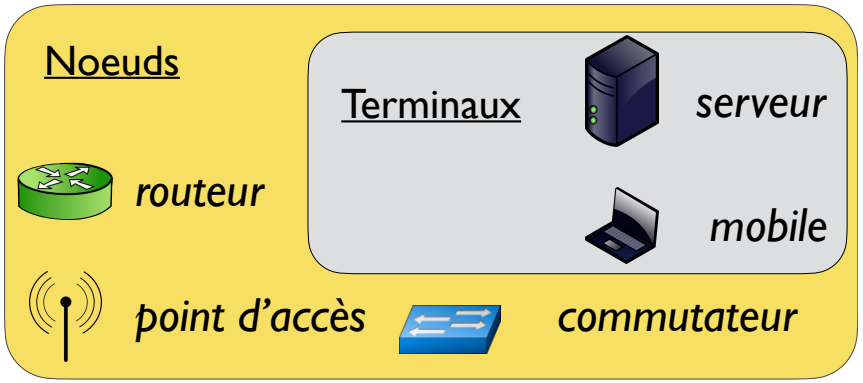
- Ex. : le client engineering.nyu.edu cherche l'adresse IP de gaia.cs.umass.edu
- Cache DNS
  - possibilité de stocker la correspondance apprise
  - améliore le temps de service



Kurose et al.

Exemple 2





Alice : Connexion au réseau Wi-Fi eduroam

Serveur DHCP : Adresse IP ; adresse IP passerelle ; adresse IP serveur DNS local

Alice : https://www.youtube.com/watch?v=9Y29TXdrBM4

Résolution DNS : 172.217.171.238

Alice : requête HTTP

Serveur Web : réponse HTTP

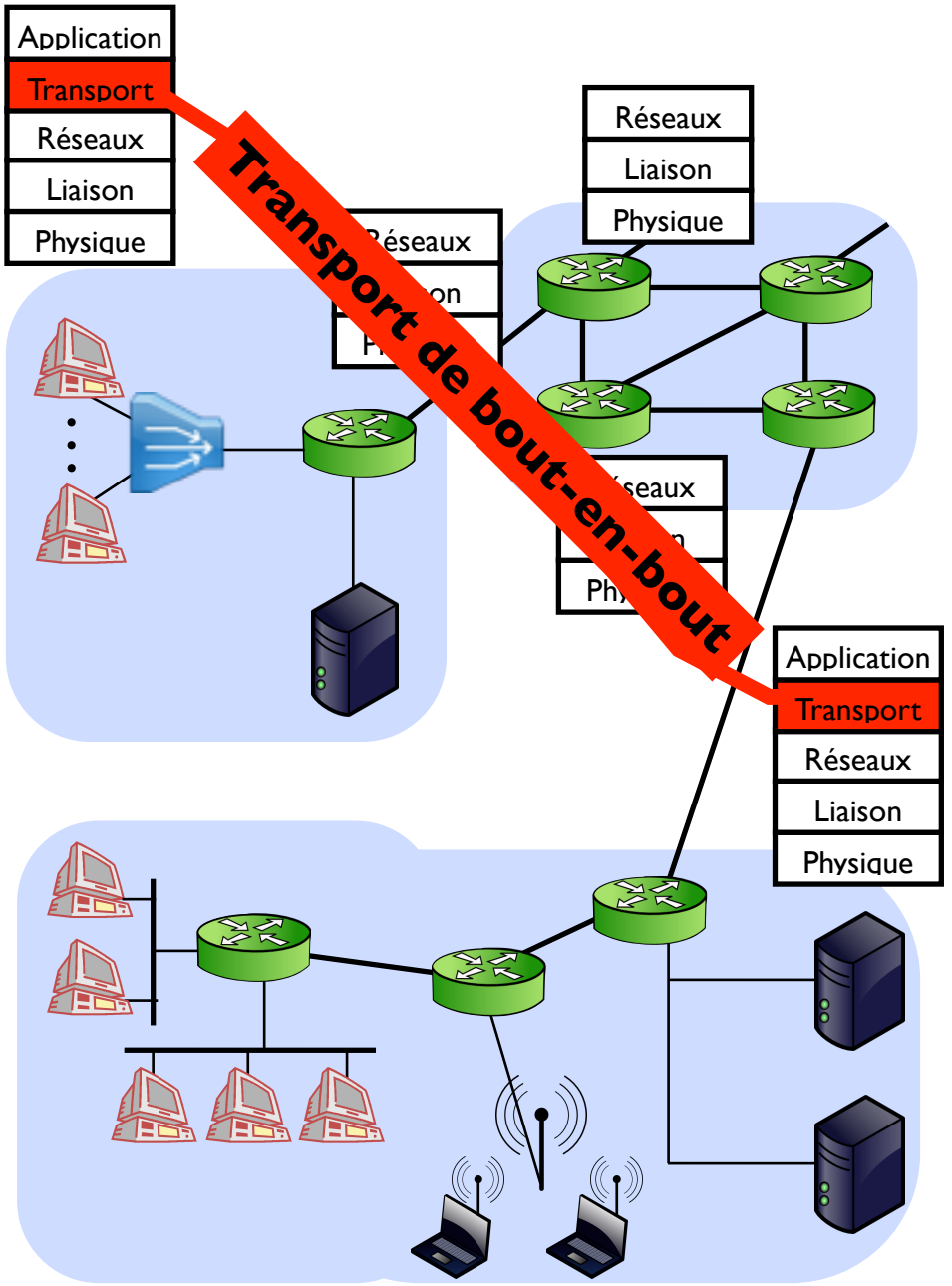
Couche Transport

Plan

1. Services de la couche Transport
2. Multiplexage et démultiplexage
3. Transport sans connexion : UDP
4. Principes du transfert de données fiable
5. Contrôle de congestion TCP

Services et protocoles de transport

- Protocoles de Transport
  - une communication logique entre des applications (processus) distantes
  - uniquement sur les terminaux
- Émetteur
  - divise les messages des applications en segments ...
  - et les transmet à la couche Réseau
- Récepteur
  - ré-assemble les segments en messages...
  - et les transmet à la couche Application
- Plusieurs protocoles possibles
  - UDP et TCP : principaux protocoles
  - nouveaux protocoles : QUIC par ex.



Protocoles de Transport sur Internet

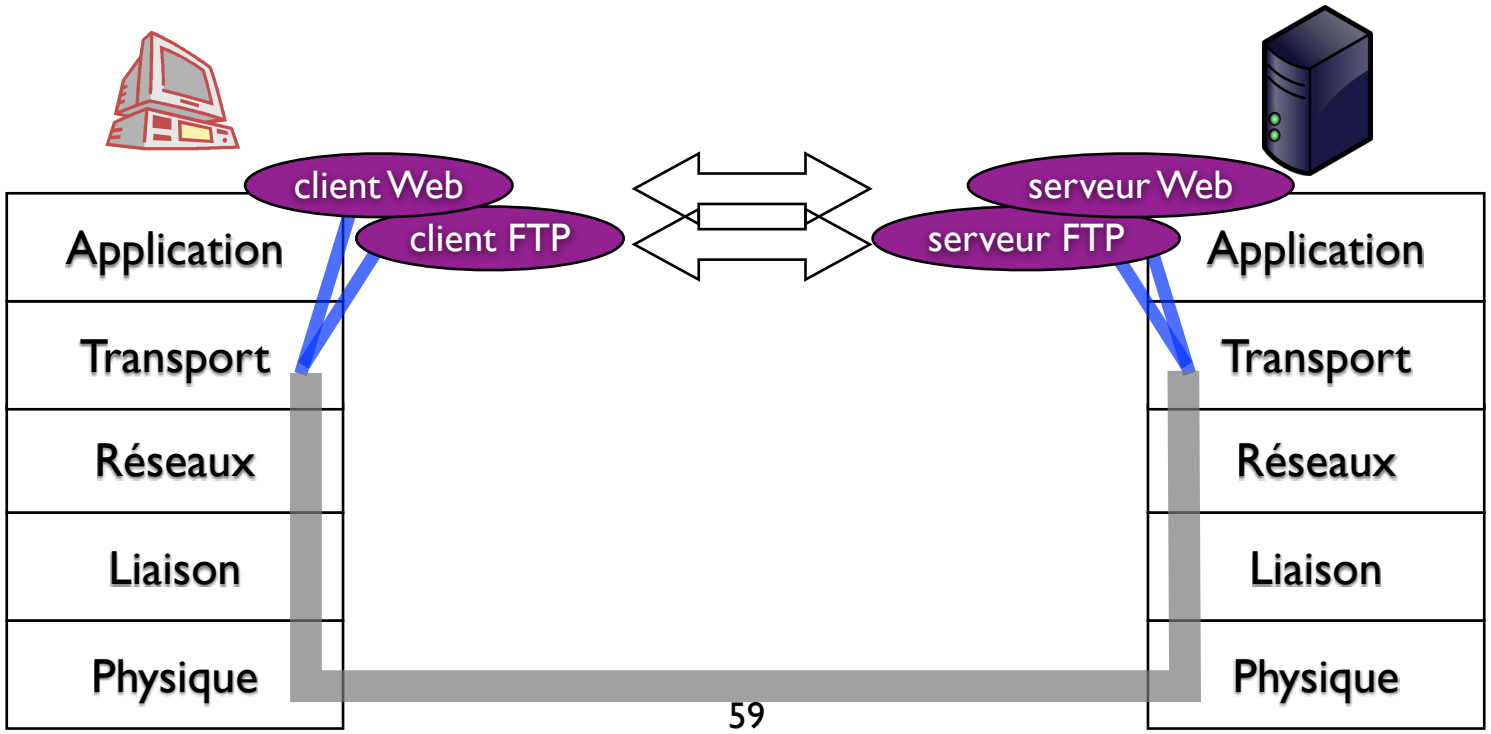
- Services communs à UDP et TCP
  - Multiplexage
  - Détection d'erreur
- Services exclusifs à TCP
  - Livraison fiable et ordonnée
  - Contrôle de congestion
  - Contrôle de flux
- Et pour UDP ?
  - Pas de services supplémentaires
- Aucun ne garantit une QoS sur les délais ou sur les débits
- TCP représente toujours entre 90 et 95% du trafic sur Internet

Plan

1. Services de la couche Transport
2. Multiplexage et démultiplexage
3. Transport sans connexion : UDP
4. Principes du transfert de données fiable
5. Contrôle de congestion TCP

Multiplexage/Démultiplexage (1/2)

- Multiplexage
  - Couche 4 : plusieurs flux d'Applications sur un même chemin IP
- Démultiplexage
  - À quelle application est destiné ce segment ?



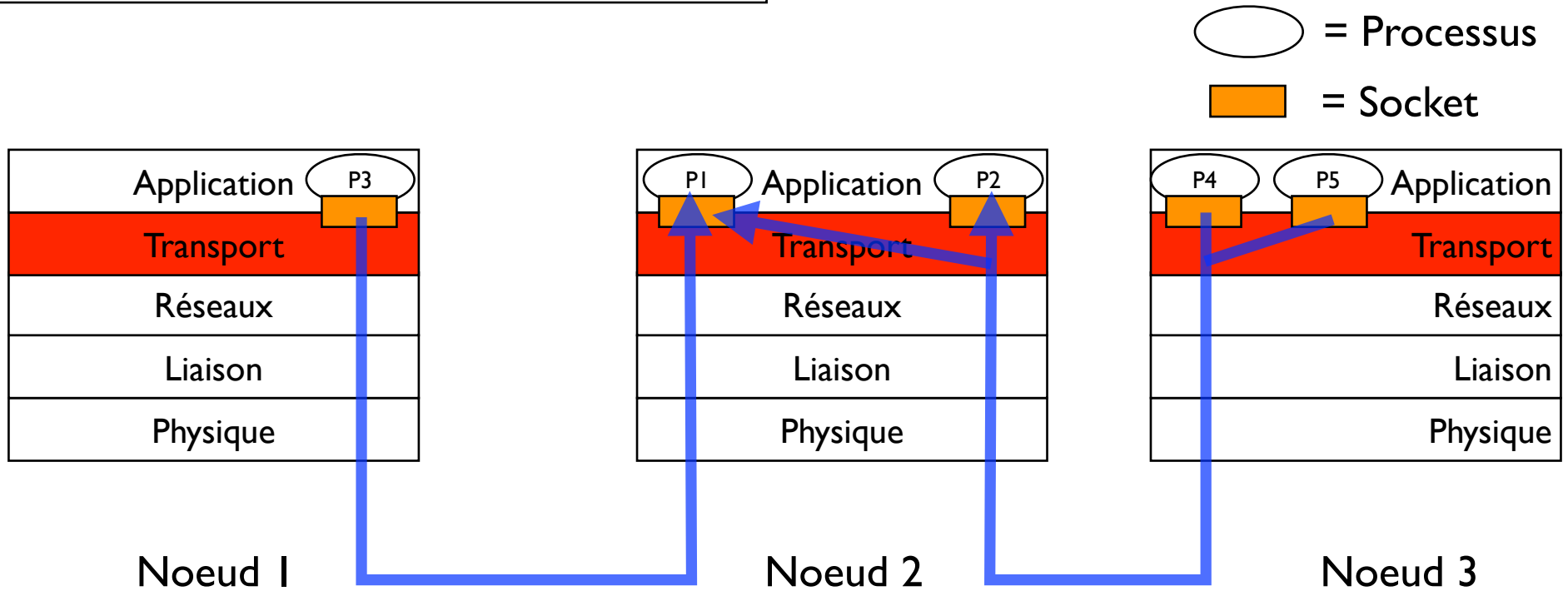
Multiplexage/Démultiplexage (2/2)

Multiplexage à l'émetteur

- rassemble les données des différentes sockets
- les encapsule en segments
- et les remet à la couche Réseau

Démultiplexage au récepteur

- délivre les segments reçus à la bonne socket

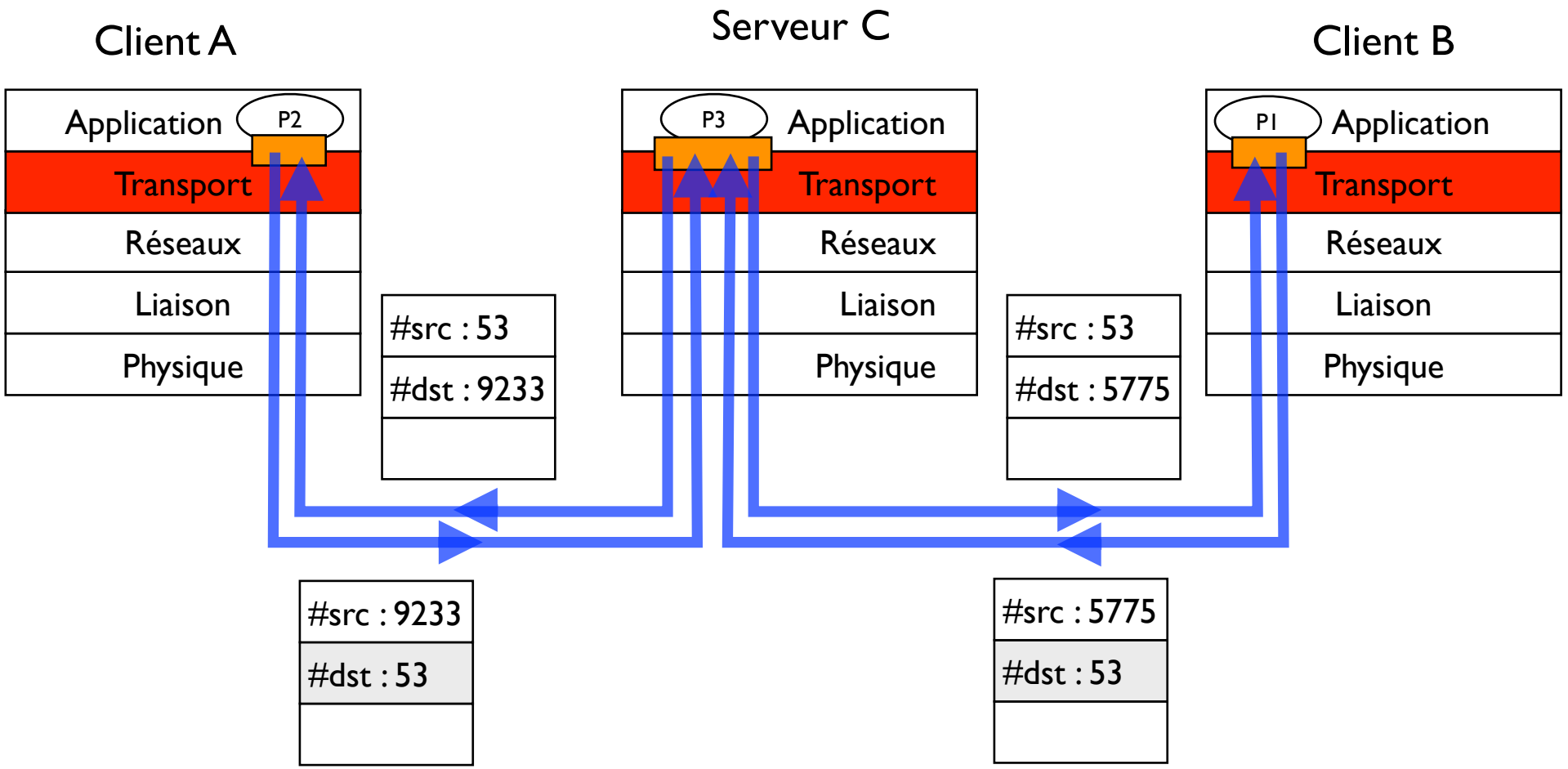


Multiplexage sans connexion (1)

- Socket UDP est identifiée par un doublet
  - @IP Dst
  - #port Dst
- À l'arrivée d'un segment UDP
  - la socket associée est choisie selon le #port Dst
- Plusieurs nœuds peuvent-ils émettre vers une même socket ?
  - oui !

Multiplexage sans connexion (2)

- #port Dst : identifie une socket/processus sur la machine @IP Dst
- #port Src présent dans en-tête UDP : peut servir pour le chemin retour

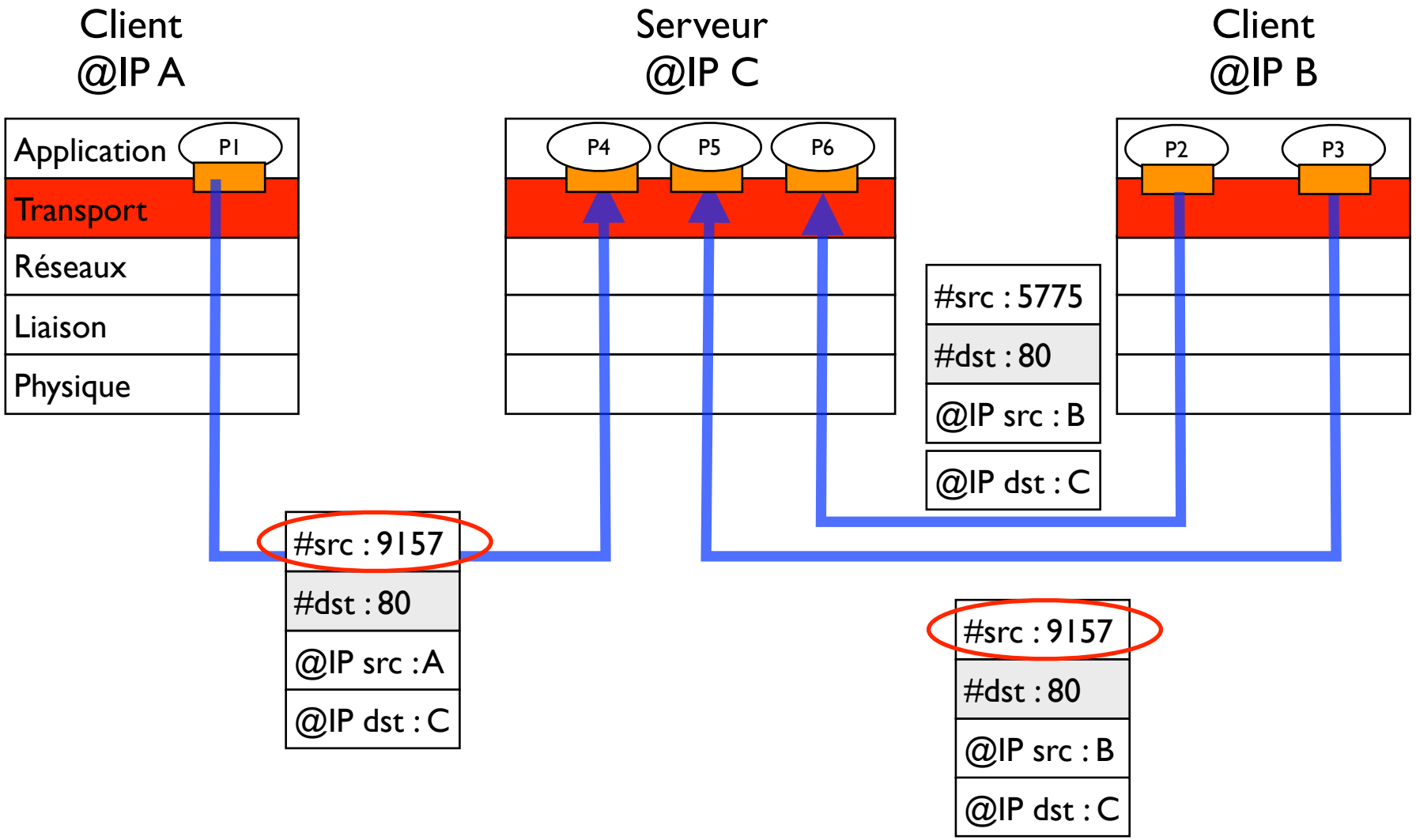


- *Si C est un serveur DNS (basé sur UDP), les paquets des clients A et B interrogent la même socket*

Multiplexage orienté connexion (1)

- Socket TCP est identifiée par un quadruplet
  - @IP Src
  - #port Src
  - @IP Dst
  - #port Dst
- Le noeud utilise ce quadruplet pour diriger le segment vers la bonne socket ou la créer
- Plusieurs nœuds peuvent-ils émettre vers une même socket ?
  - Non ! Chaque noeud envoie ses segments vers une socket différente du serveur.

Multiplexage orienté connexion (2)



- *Si C est un serveur Web, les client A et client B ouvrent des connexions HTTP séparées sur C et le client B ouvre deux connexions HTTP*

Choix du numéro de port

- Le processus sur le serveur écoute et est prêt à recevoir (socket ouverte) sur un (ou plusieurs) port(s)
- Le processus sur le client doit envoyer ses segments (associés à une même application) avec le même numéro de port du serveur comme numéro de port destination
- Le processus sur le client choisit (souvent aléatoirement) son numéro de port source
- Quels ports choisir pour un serveur ? Convention
  - FTP : 20 et 21 (TCP)
  - SSH : 22 (TCP)
  - Telnet : 23 (TCP)
  - SMTP : 25 (TCP)
  - DNS : 53 (TCP ou UDP)
  - DHCP : 67 et 68 (UDP)
  - TFTP : 69 (UDP)
  - HTTP : 80 (TCP)
  - POP : 110 (TCP)
  - NTP : 123 (UDP)
  - NetBIOS : 137,138 et 139 (TCP et UDP)
  - IMAP : 143 (TCP)
  - SNMP : 161 et 162 (TCP et UDP)
  - LDAP : 389 (TCP et UDP)
  - HTTPS : 443 (TCP)
  - Skype : 443 (TCP), 3478-3481 (UDP), 49152-65535 (TCP et UDP)
  - …

Plan

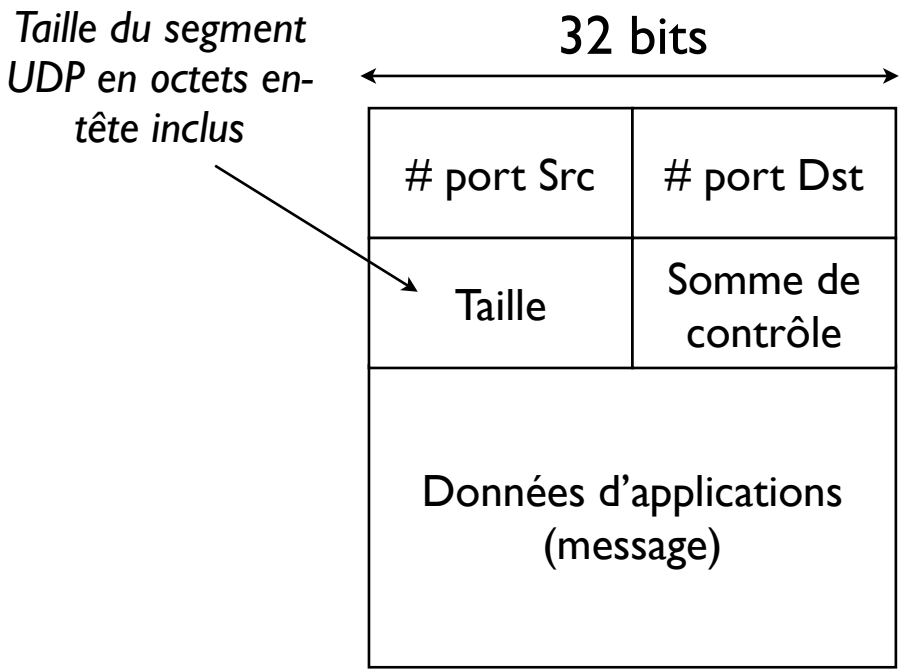
1. Services de la couche Transport
2. Multiplexage et démultiplexage
3. Transport sans connexion : UDP
4. Principes du transfert de données fiable
5. Contrôle de congestion TCP

UDP : User Datagram Protocol [RFC 768]

- Service minimum
  - Multiplexage et démultiplexage (cf slides 61&62)
  - Détection d'erreurs (pas obligatoire, fixé à 0 si pas utilisé)
- Sans connexion
  - Pas d'échange initial entre les couches Transport de l'émetteur & du récepteur
- Alors pourquoi utilise-t-on UDP ?
  - + rapide
    - Pas d'établissement de connexion
  - + simple
    - Pas d'état de connexion à maintenir, ni paquets à mémoriser
  - + léger
    - Faible volume des en-têtes (8 octets vs 20 pour TCP) et moins de sockets
  - + contrôle de l'appli
    - L'appli décide de son débit (taille des segments & instants d'émission)

UDP (2)

- Très répandu dans les applications
  - nécessitant une très grande réactivité
    - DNS, SNMP, RIP,…
  - multimédias tolérantes aux pertes
    - téléphonie par Internet, visioconférence, ...
- Mais risque de congestion du réseau par le trafic UDP
- # port Src non utilisé pour atteindre la socket du destinataire
  - utile pour les messages retours éventuels



Format des segments UDP

Somme de contrôle UDP (1)

- But : Détecter les erreurs (bits altérés) dans les segments transmis

Émetteur

1. Considère le contenu d'un segment comme des mots de 16 bits
2. Calcule le complément à 1 de la somme de tous ces mots
3. Inscrit la valeur trouvée dans le champ réservé du segment UDP

Récepteur

1. Calcule la somme de contrôle pour le segment reçu
2. Vérifie si la valeur calculée égale celle inscrite dans le champ :
   - Si NON, erreur détectée
   - Si OUI, pas d'erreur détectée...
   - .. absolument certain ?

Somme de contrôle UDP (2)

- Exemple :
  - En additionnant des nombres, la retenue du bit le plus significatif sera ajoutée au résultat
  - 2 mots de 16-bits

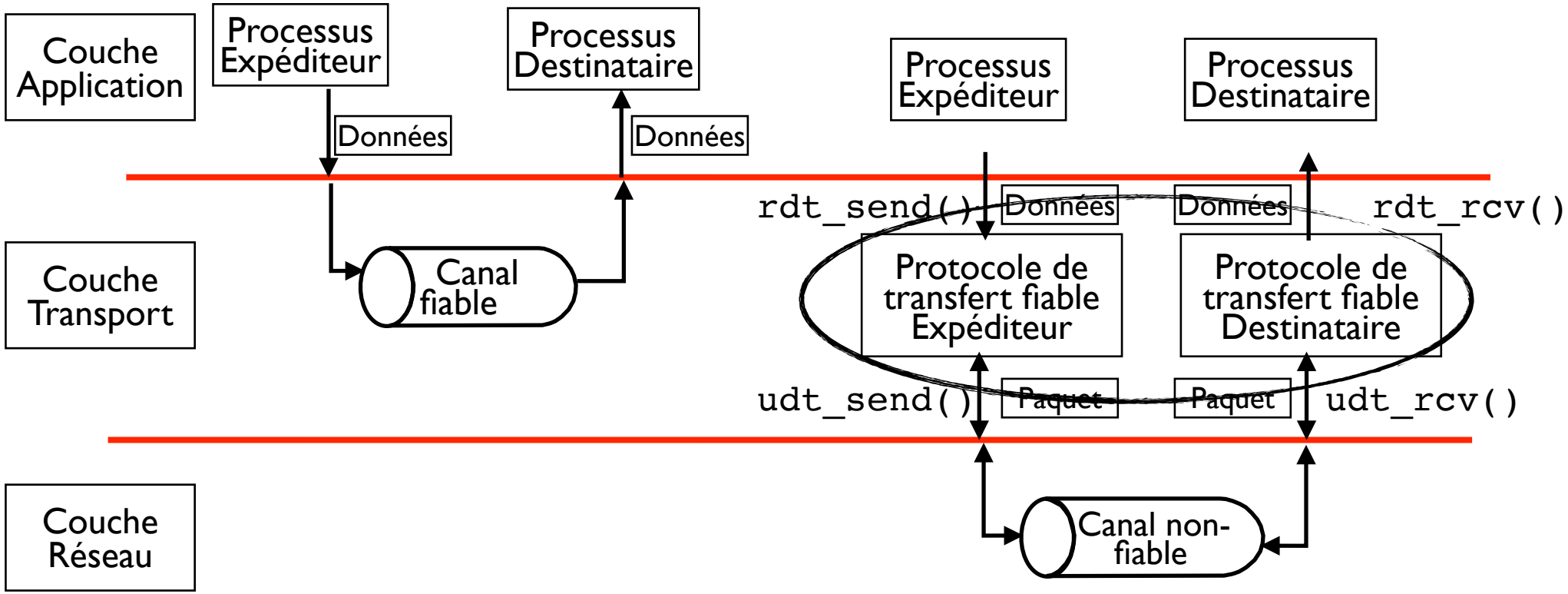
mot1	1 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0
mot2	1 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
∑ retenue	1 1 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1
∑	1 0 1 1 1 0 1 1 1 0 1 1 1 1 0 0
∑ contrôle	0 1 0 0 0 1 0 0 0 1 0 0 0 0 1 1

Plan

1. Services de la couche Transport
2. Multiplexage et démultiplexage
3. Transport sans connexion : UDP
4. Principes du transfert de données fiable
5. Contrôle de congestion TCP

Transfert fiable sur un canal non-fiable ?

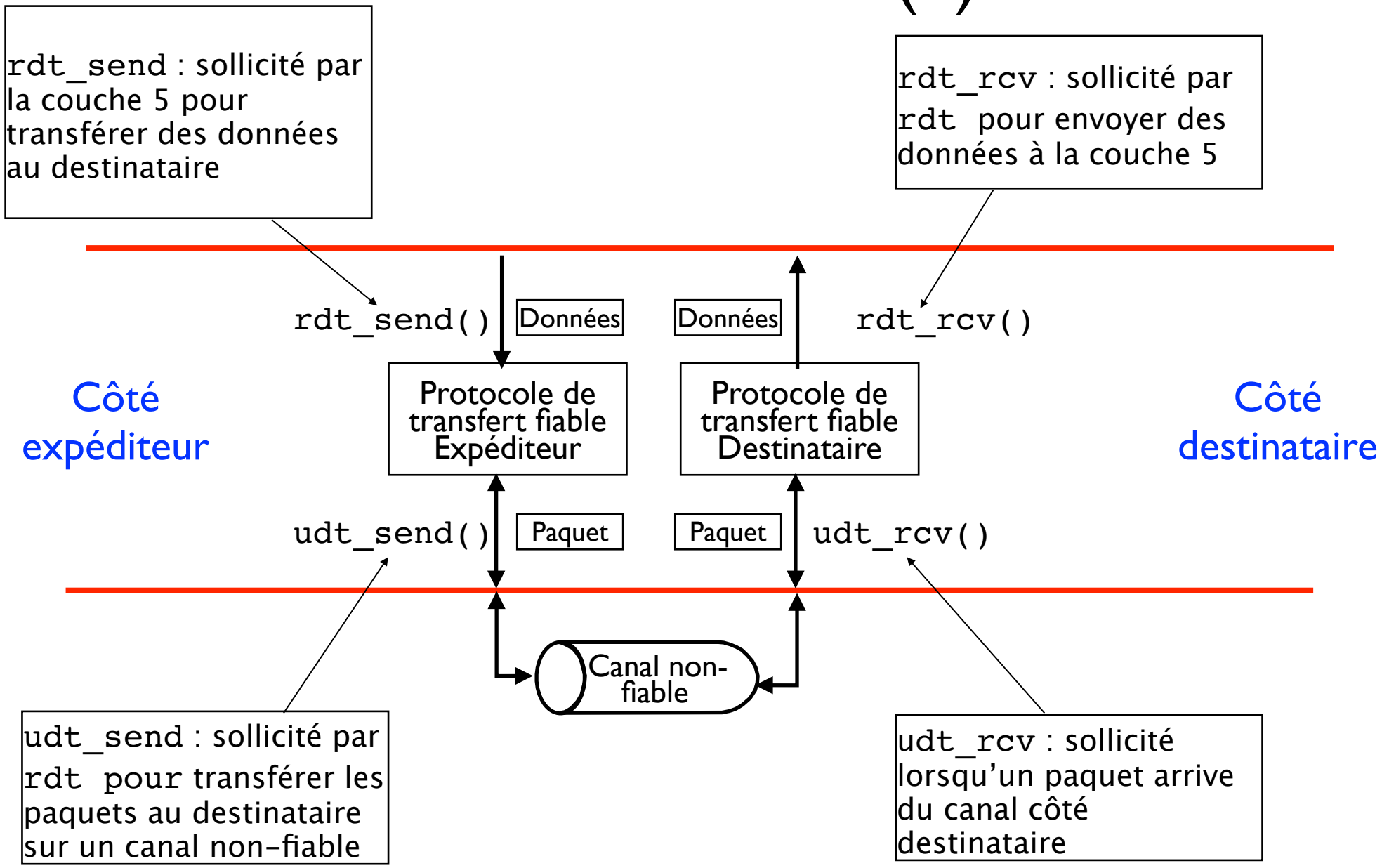
- rdt = “reliable data transfer” (fiable)
- udt = “unreliable data transfer” (non-fiable)
- Comment construire `rdt` à partir de `udt` ?



a) Service voulu

b) Service existant

Principes d'un transfert de données fiable (1)

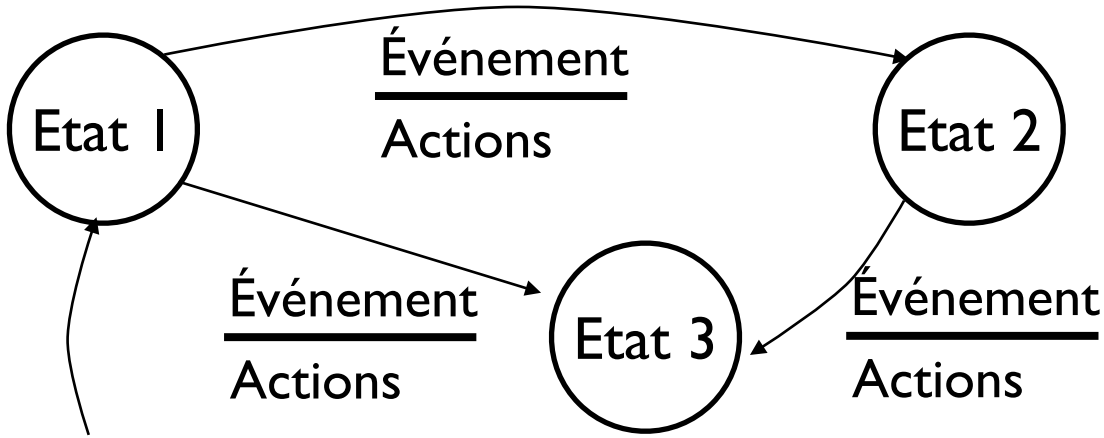


- La complexité du protocole rdt dépend des caractéristiques du canal

Principes d'un transfert de données fiable (2)

- Construction d'un protocole de transfert de données fiable
  - *"reliable data transfer"*
  - appellation : rdt x.x
- Considère uniquement le transfert de données unidirectionnel
  - Les paquets de contrôle circulent eux dans les deux sens
- Description par des automates à nombre d'états finis

L'état suivant est déterminé uniquement par le prochain événement

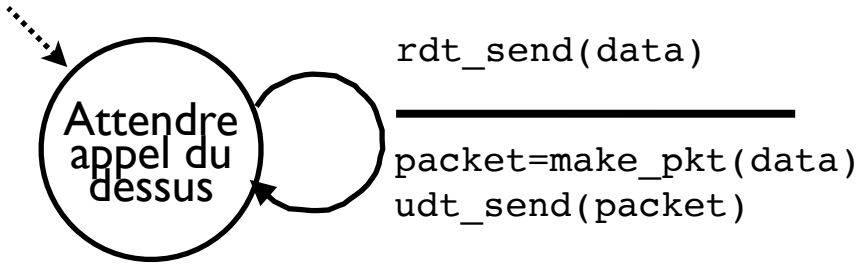


Les événements causent les transitions d'état

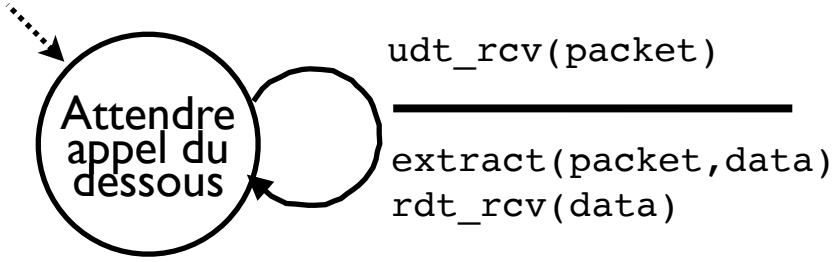
Les actions sont prises lors des transitions d'état

rdt1.0 : transfert fiable sur canal fiable

- Hypothèse : canal totalement fiable
  - pas d'erreur binaire sur le canal
  - pas de perte de paquets
- Automates différents pour l'expéditeur et le destinataire
  - l'émetteur envoie les données dans le canal inférieur
  - le destinataire reçoit les données du canal inférieur



Expéditeur



Destinataire

rdt2.0 : canal avec erreurs binaires

- Hypothèse : canal introduit des erreurs binaires dans les paquets
- Comment se rétablir de ces erreurs ?

  (a) Détection d'erreurs

  - Somme de contrôle

  (b) Accusé de réception

  - ACK : acquittement - le destinataire informe l'expéditeur que le paquet a été bien reçu
  - NAK : acquittement négatif - le destinataire informe l'expéditeur que le paquet comporte une erreur
  - Boucle de retour

  (c) Retransmission des paquets erronés

  - À la réception d'un NAK

- Protocole "Send and wait"
  - l'expéditeur émet un paquet ... puis se met en attente d'un ACK/NAK