

Introduction aux services réseaux - R2.05

Architecture des réseaux

Cours 2

Michel Salomon

IUT Nord Franche-Comté  
Département d'informatique

basé sur un cours de Frédéric Suter

Ouvrage de référence

*Computer Networking : A Top-Down Approach*

Jim Kurose and Keith Ross. Addison-Wesley.

# Plan du cours

## ① Couche Application

Principe des applications; *World Wide Web*; Transfert de fichiers; Courrier électronique; Nom de domaine

## ② Couche Transport

- Rôle de la couche transport
- Description des protocoles de la couche transport

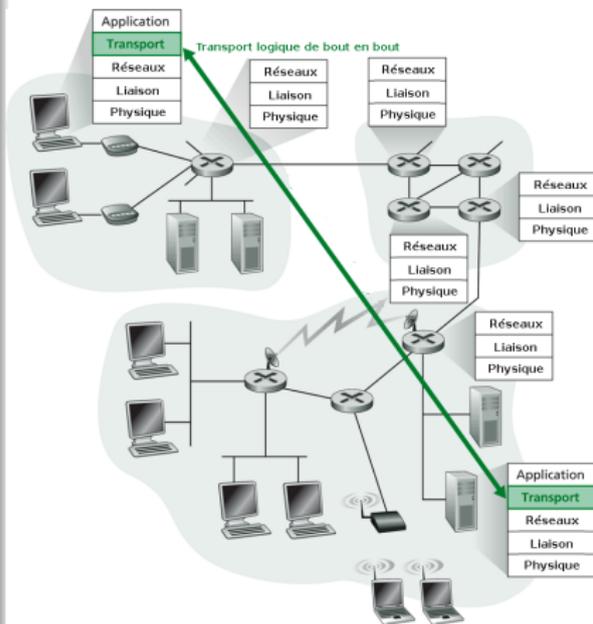
## ③ Internet Protocol v6 (IPv6)

# Couche Transport

- Quel est le rôle de la couche transport ?
- Description de protocoles de niveau transport
  - Caractéristiques
  - Transport avec connexion → TCP
  - Fiabilisation du transfert de données
  - Transport sans connexion → UDP

# Quel est le rôle des protocoles de transport ?

- Fournir une **communication logique** entre processus applicatifs
- Protocoles exécutés aux extrémités
  - Émetteur
    - Découpe les messages appli. en **segments**, puis
    - transmission à la couche réseau
  - Récepteur
    - Réassemble les segments en messages applicatifs, puis
    - transmission à la couche application
- Protocoles disponibles
  - *Transmission Control Protocol*
  - *User Datagram Protocol*



# Comparaison couches transport et réseau

## Couche réseau

Communication logique entre hôtes distants

## Couche transport

Communication logique entre processus distants  
(utilise et augmente les services de la couche réseau)

## Analogie ⇒ correspondance entre des enfants

des enfants envoient des lettres à d'autres enfants

- Messages applicatifs → lettres dans enveloppes
- Processus → enfants
- Hôtes → maisons
- Protocole de la couche transport → parents
- Protocole de la couche réseau → service postal

# Caractéristiques des protocoles de transport d'Internet

## Transmission Control Protocol

### Transfert fiable et ordonné

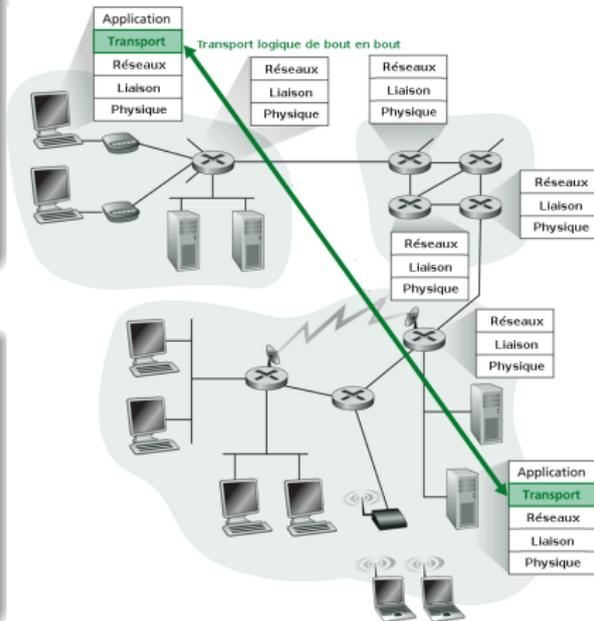
- Gestion de connexion
- Contrôle d'erreur (*Checksum*)
- Contrôle de perte (numérotation)
- Contrôle de flux

## User Datagram Protocol

### Transfert non fiable et éventuel. désordonné

- Extension du "best effort" d'IP
- Contrôle d'erreur (*Checksum*)

Services non fournis → délai et débit



# Découpage en segments

- Messages applicatifs traités comme un flux (suite) d'octets
- Découpage du flux d'octets en segments → **fragmentation**
  - Taille maximale de 65536 octets
  - Taille variable suivant le type de réseau (couche Liaison)
    - 1 segment TCP dans 1 datagramme IP
    - Taille du datagramme IP variable suivant le type de trame

## En-tête d'un segment TCP (20 octets sans options)

- Port Source (16 bits) et port destination (16 bits)
- Num. de séquence (32 bits) et d'accusé de réception (32 bits)
- Drapeaux (6 bits) déterminant le type de segment
  - Ouvrir une connexion → SYN
  - Acquiescement → ACK
  - Fermeture d'une connexion → FIN ou RST
  - etc.
- etc.

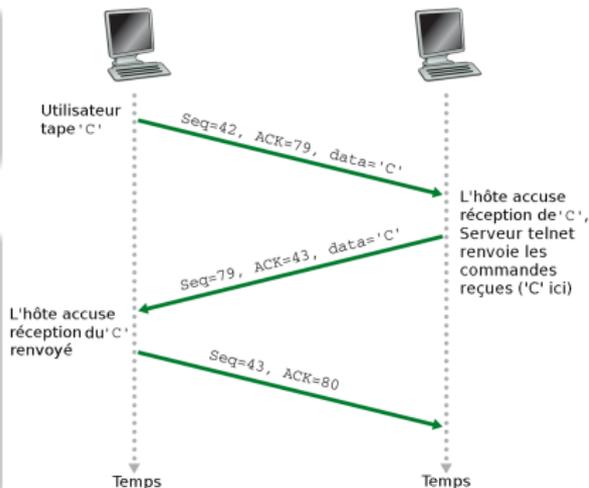
# Numéro de séquence et d'acquittement dans TCP

## Numéro de séquence (SEQ)

Numéro du premier octet du segment dans le flux d'octets envoyé

## Numéro d'acquittement (ACK)

- Numéro du prochain octet attendu dans le flux reçu
- Acquitte tous les octets dont le numéro est inférieur



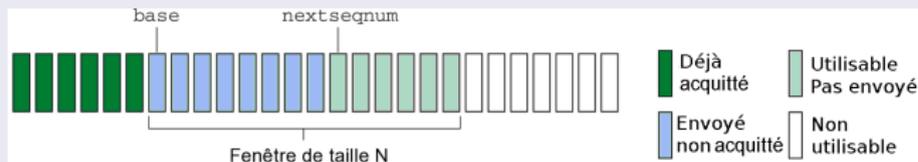
# Fiabilisation - Mécanisme de fenêtre glissante

## Mécanisme naïf

- Transmission d'un segment, puis pour une autre transmission → **attendre l'acquittement**
- Faible efficacité

## Mécanisme de fenêtre glissante (protocole pipeliné)

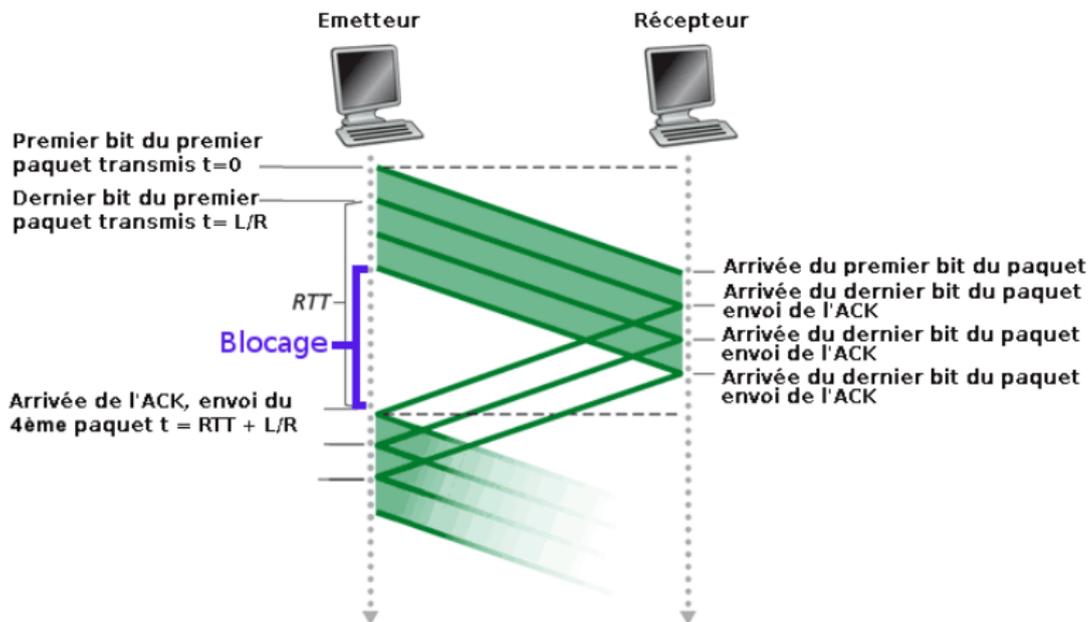
- **Fenêtre** = suite de segments pouvant être transmis  
→ **sans attendre d'acquittement** → transfert bufferisé
  - 1 fenêtre du côté émetteur, 1 fenêtre du côté récepteur



- Performances fonction de la taille de la fenêtre et du réseau
  - Adaptation de la taille en fonction du réseau (pertes observées)
  - Récepteur indique la quantité de données qu'il peut traiter

# Fiabilisation - Mécanisme de fenêtre glissante

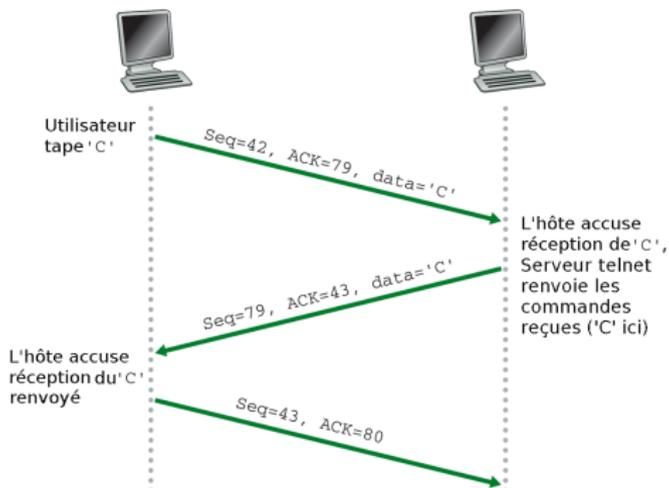
Une fenêtre avec une taille suffisante ne produit pas de blocage



# Fiabilisation - Contrôle des échanges

## Acquittement

- Technique de superposition ou *piggybacking*  
→ 1 segment transporte des données + l'ACK d'un segment reçu
- Acquittement cumulatif  
→ 1 ACK acquitte les segments de SEQ inférieur



# Établissement de la connexion en 3 étapes (3-way handshake)

- 1 Client → Serveur
  - Envoie un segment SYN
  - Initialise son num. de seq.
  - Pas de données dans le segment
- 2 Serveur → Client
  - Reçoit le segment SYN
  - Répond via un segment SYN/ACK
  - Initialise son num. de seq.
- 3 Client → Serveur
  - Reçoit le segment SYN/ACK
  - Répond avec un segment ACK
  - Peut contenir des données

