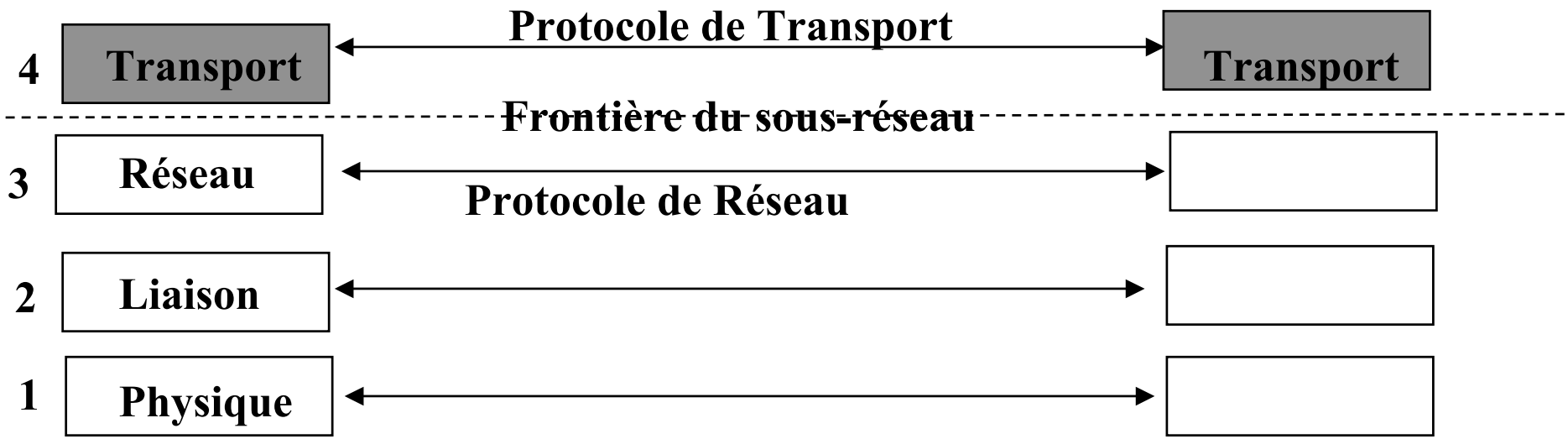
(Chapitre - 7)

La couche transport

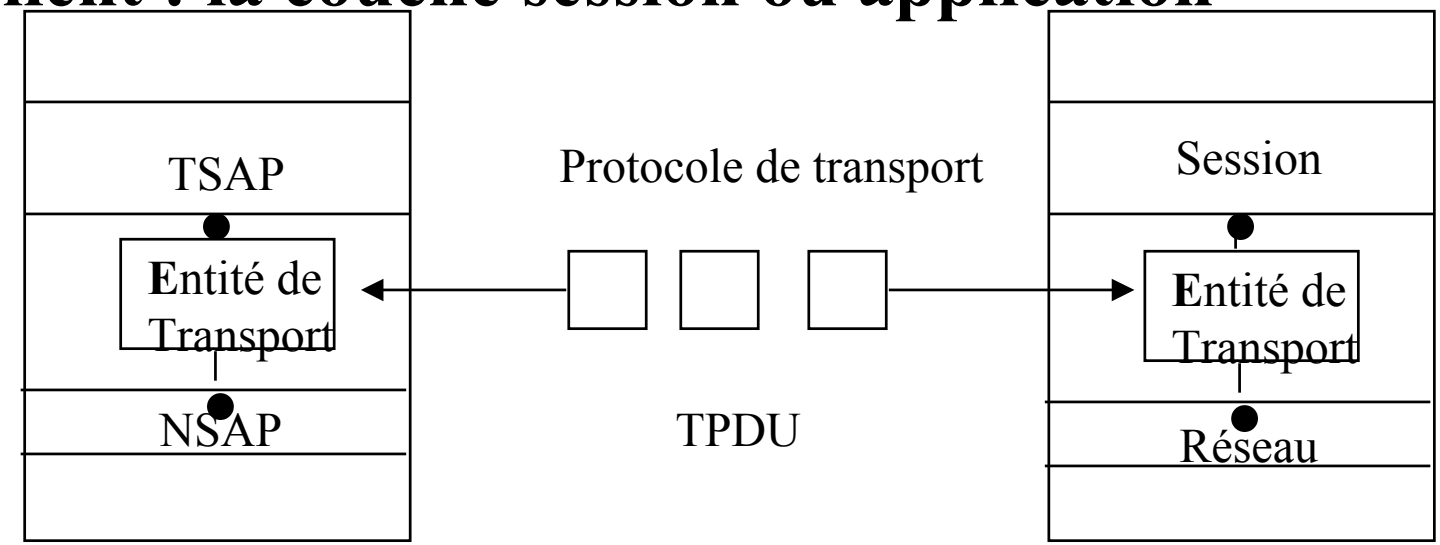


Plan du chapitre 7

- **Introduction**
- **Qualité de service**
- **Les protocoles de transport**
- **Gestion de la connexion :**
  - **L'adressage**
  - **Garantie de délivrance des messages**
  - **L'établissement d'une connexion**
  - **Libération d'une connexion**
  - **Contrôle de flux et mémorisation**
  - **Multiplexage**
- **Un exemple de transport : TCP (Transport Control Protocol)**

Introduction

- **Transfert fiable efficace, sûr et économique d'informations de bout en bout (pas de vision des sites intermédiaires)**
- **Client : la couche session ou application**



Les services de la couche transport d'Internet

- **Rappel: IP sans connexion non fiable, déséquencement possible, délai de transfert très variable**
- **Protocoles pensés au départ pour le transfert de fichier et l'utilisation de machine à distance**
- **Deux types de services suivant les besoins de l'application à développer**
  - **Sans connexion, aucune QoS: User Datagram Protocol (UDP)**
  - **Avec connexion: Transport Control Protocol (TCP)**
    - **Ouverture et fermeture de connexion**
    - **Fragmentation et reassemblage**
    - **Contrôle de flux et récupération des erreurs**
    - **Données urgentes**
- **TCP et UDP se sont imposés naturellement. Les normes OSI ont été définies en parallèle mais elles sont arrivées trop tard**
- **Dans les norme OSI il existe différents niveaux de QoS pouvant être utilisés suivant le réseau sous-jacent**

Fonctionnalités des protocoles transport d'Internet

.Protocoles client/serveur:

- Le serveur se met en attente de demandes

- Le client initie le dialogue par une demande

•Problème de pouvoir faire communiquer 2 processus sur des hôtes différents

•Interface des "sockets"

•1 serveur : 1 numéro de porte fixé (ou port) = un point d'entrée prédéfini sur une machine => 1 processus peut être associé à une porte

- Côté serveur: réservés pour applications standards

- Fichier /etc/services

- Exemple: HTTP port 80 en TCP

- Côté client: alloués dynamiquement

•Multiplexage vers les applications:

- Adresses Internet source et destination, numéros de ports source et destination

- Entêtes réseau (IP) et transport (TCP ou UDP)

Qualité de service : Paramètres (1) (Norme OSI)

- **Des besoins de qualité variables**
  - **Ex1: transfert de fichier : taux d'erreur nul, débit non primordial, temps de transit non plus**
  - **Ex2: téléphone: taux d'erreur peut être non nul, débit et temps de transit minimum (<0,25s)**
- **Temps d'établissement d'une connexion**
  - **Durée s'écoulant entre l'instant où la demande est émise par l'utilisateur et celui à partir duquel où la confirmation est reçue.**
- **Probabilité d'échec d'établissement**
  - **Probabilité que la connexion ne puisse s'établir dans un délai maximum défini.**
- **Débit de la connexion**
  - **Mesure du nombre d'octets utiles pouvant être transférés en une seconde sur la liaison. Le débit est mesuré séparément dans les deux sens.**
- **Temps de transit**
  - **Temps s'écoulant entre l'instant où un message est émis et l'instant où il est reçu par l'entité de transport du destinataire.**
- **Taux d'erreur résiduel**
  - **Quotient entre le nombre de messages perdus ou mal transmis et le nombre total de messages émis au cours d'une période de temps.**

Qualité de service : Paramètres (2)

- **Probabilité d'incident de transfert**
  - **Fraction de temps durant laquelle la qualité de service n'a pas été respectée.**
- **Temps de déconnexion**
  - **Durée s'écoulant entre la demande de déconnexion émise par un utilisateur et la déconnexion effective de l'usager distant.**
- **Probabilité d'erreur de déconnexion**
  - **Taux de demandes de déconnexion non exécutées pendant le temps maximum qui était défini.**
- **Protection**
  - **Possibilité de résister aux intrusions d'un tiers sur la connexion.**
- **Priorité**
  - **Permet aux utilisateurs de privilégier certaines connexions.**
- **Résiliation**
  - **Liberté laissée à la couche transport de fermer une connexion suite à un engorgement ou à des problèmes internes.**

Qualité de service : mécanisme

- **À l'ouverture de la connexion deux valeurs de chacun des paramètres sont transmises à l'entité transport :**
  - **La valeur souhaitée**
  - **La valeur minimum acceptée**
- **L'entité de transport locale regarde si elle peut au moins satisfaire les valeurs minimales demandées.**
  - **si oui, elle transmet à l'entité distante les valeurs définissant la QOS qu'il peut satisfaire.**
  - **si non, elle retourne une erreur à la couche application.**
- **L'entité de transport distante opère de la même façon sur les valeurs reçues.**
  - **Si une ou plusieurs valeurs ne peuvent être garanties, la demande est rejetée.**
  - **Si non la demande est acceptée et l'utilisateur reçoit les valeurs accordées.**
- **Les paramètres définissant la QOS ne sont pas normalisés**

QoS dans Internet

- **Application temps-réel : la QoS de TCP est insuffisant**
- **Niveau Réseau**

– RSVP: Ressource Reservation Protocol

- Permet de réserver dans les routeurs des ressources permettant de garantir un débit, délai de traversée...

- C'est le destinataire qui est à l'origine des réservations en fonction de la QoS qu'il espère

- Nécessite des échanges de paquets de signalisation

–Différentiation de service (DiffServ):

- Définition de classes de services de qualités différentes

- Marquage des paquets et traitement prioritaire dans les routeurs

–Ces techniques sont peu utilisées à grande échelle car difficile à mettre en oeuvre

Garanties de QoS

- **Niveau Tansport TCP inutile pour beaucoup d'application "temps-réel"**
  - **Exemple: le téléphone**
  - **Les paquets perdus et réemis arrivent trop tard**
  - **Un taux d'erreur non nul est possible**
  - **Utilisation de UDP**

- **Une couche supplémentaire entre UDP et l'application: RTP (Real Time Protocol) normalisé**
  - **Numérotation des paquets, estampillage temporel, rapports du récepteur à l'émetteur pour signaler: la QoS courante (délai de transit, taux d 'erreur, débit…)**
  - **Adaptation par l'application (changement de taux de compression, correction à l 'arrivée, tampon d 'amortissement à l 'arrivée …)**

Les protocoles de transport

- **Comparaison transport et liaison de données**
  - **Communication via un sous-réseau / communication via un canal**
    - » **adressage**
    - » **importance du protocole d'ouverture**
    - » **délai introduit par les sous- réseaux opérant en mode datagramme**
    - » **déséquencement des paquets**
    - » **Transport dépend des services offerts par la couche réseau (fiable ou non, orienté connexion ou pas)**
  - **Différence de techniques pour le contrôle de flux**
    - » **allocation statique de tampon (fenêtre)**
    - » **allocation dynamique (crédits)**
- **Norme OSI: Il existe différentes classes de protocole (0 ..4) suivant fiabilité de la couche réseau utilisée**

Établissement d'une connexion (1)

- **Hypothèse : connexion bi directionnelle entre deux entités**
- **Problème :**

  Si le niveau transport est mis en œuvre au dessus d'une couche réseau non fiable, on peut avoir plusieurs exemplaires de la même unité de protocole en cours de transmission. On doit donc impérativement reconnaître et éliminer les doublons.

  -> Perte de paquet

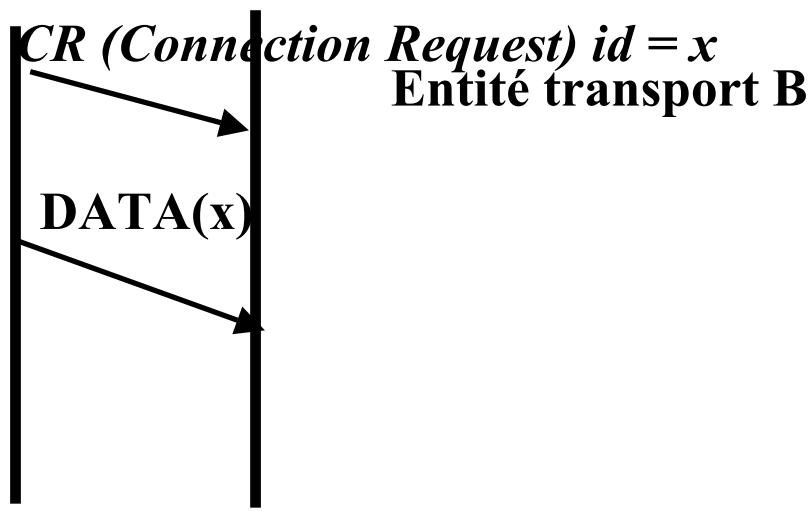
  -> Retardement de paquets

  Comment définir une connexion sans ambiguïté ?

  Comment associer une donnée à une connexion ?

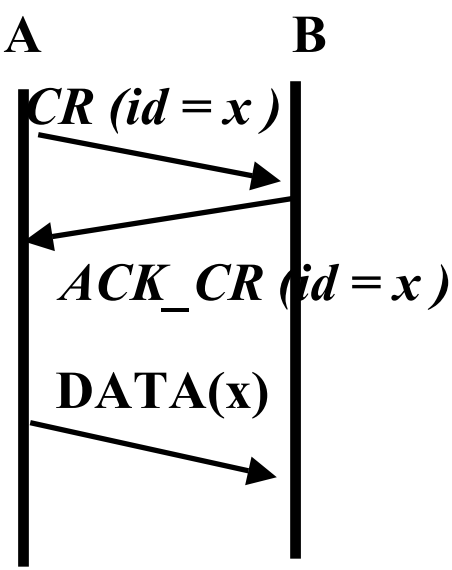
- **Echange de paquets spéciaux pour l'ouverture de la connexion**
- **Identificateur de connexion dans l'entête des paquets**
- **Numéro de séquence pour les données pour différencier les paquets d'une même connexion**
- **Chaque TPDU (Transport Protocol Data Unit) circulant est identifiée de façon unique par un couple :**

<id connexion, n°TPDU>

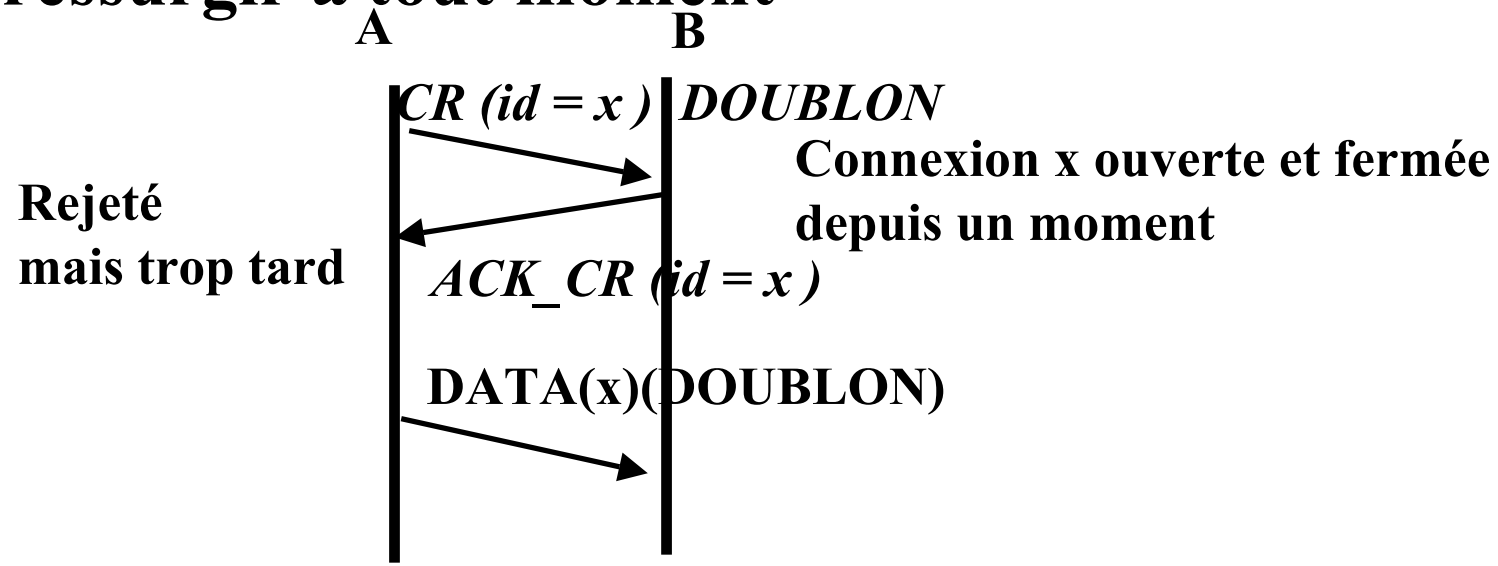


- **Problème 1 : perte de paquets**
  - **CR se perd**
  - **Les données sont envoyées mais rejetées en B (pour lui pas de connexion)**
  - **Donc on rajoute ACK_CR + armement de timer de réemission en A**

Armement d'un timer



- **Si CR se perd ou ACK_CR se perd:**
  - **A renvoie un CR avec même x**
  - **Les données sont envoyées seulement après ACK_CR**
- **Introduction de doublon potentiel dans le réseau qui peuvent ressurgir à tout moment**



- **Solutions**
  - **Se rappeler quels ids de connexion ont déjà été déjà utilisés.**
  - **Pour reconnaître les doublons de demandes d'ouverture de connexion, on conserve les références des connexions fermées.**
  - **Si on veut résister aux pannes: sauvegarde sur disque dur des ids déjà utilisés**

Problème : le nombre d'id de connexion n'est pas infini

Il faudra reprendre à un moment les ids déjà utilisés.

-> On se débrouille pour avoir une durée de vie limitée des paquets

Exemple dans TCP

- **Ouverture à trois étapes avec couple de références**
- **L'identificateur de connexion et le numéro de séquence sont confondus:**
  - **l'id sert de numéro initial**
  - **Les paquets suivants n'ont plus d'id de connexion mais seulement un numéro de séquence**
- **Id initiaux calculés sur une horloge de période = 4 microsec.**
  - **Id unique pendant 2^{32}* 4 microSec = 4 heures**
  - **Problème: l'évolution des NoSeq dépend du débit de la connexion et on peut arriver à des cas d'ambiguité par rapport aux id initiaux**
  - **Solution on attend la durée de vie max des TP avant de ré-ouvrir une connexion sur la même paire de ports (cela pourrait être affiné en fonction du débit de la connexion)**
- **Quand un ordinateur plante : On attend la durée de vie maximum d'un paquet. (Pour TCP: 120 s)**

Libération d'une connexion(1)

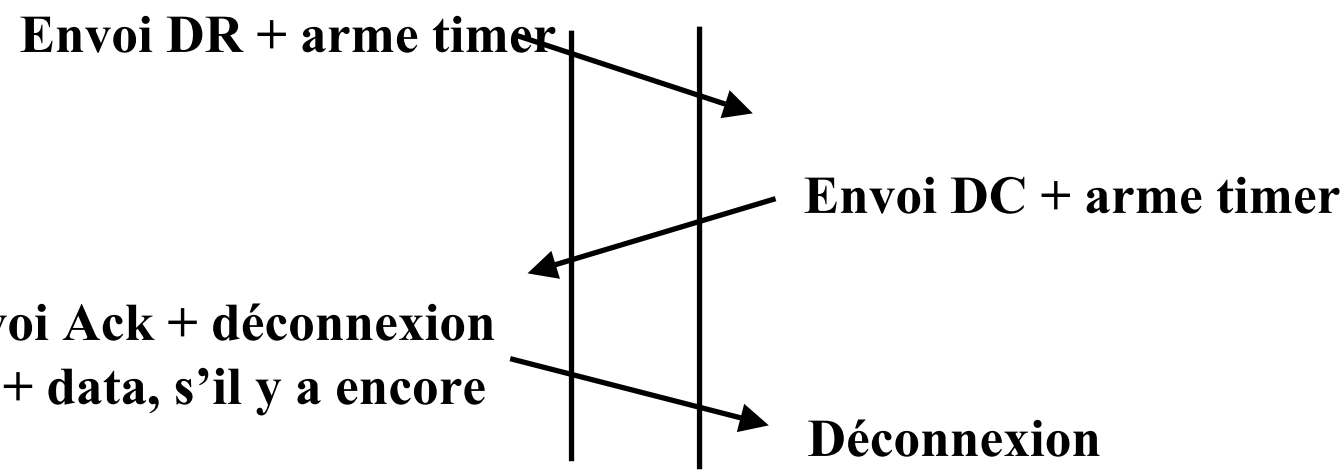
- **1ère solution:**
  - **Fermeture brutale, quand on veut fermer la connexion d'un coté , on ne se soucie pas de l'état de l'autre . Comme au téléphone, on raccroche.**
- **2ème solution**
  - **Problème**

  Il faut éviter que la rupture de la connexion provoque des pertes de messages. Pour cela, il faut essayer que les deux entités de transport aient la même vision de la connexion.

  ->la connexion est gérée comme deux demi connexions unidirectionnelles
  - **On peut fermer en émission, et continuer à recevoir, si l'autre n'a pas fini d'émettre.**

Libération d'une connexion(2)

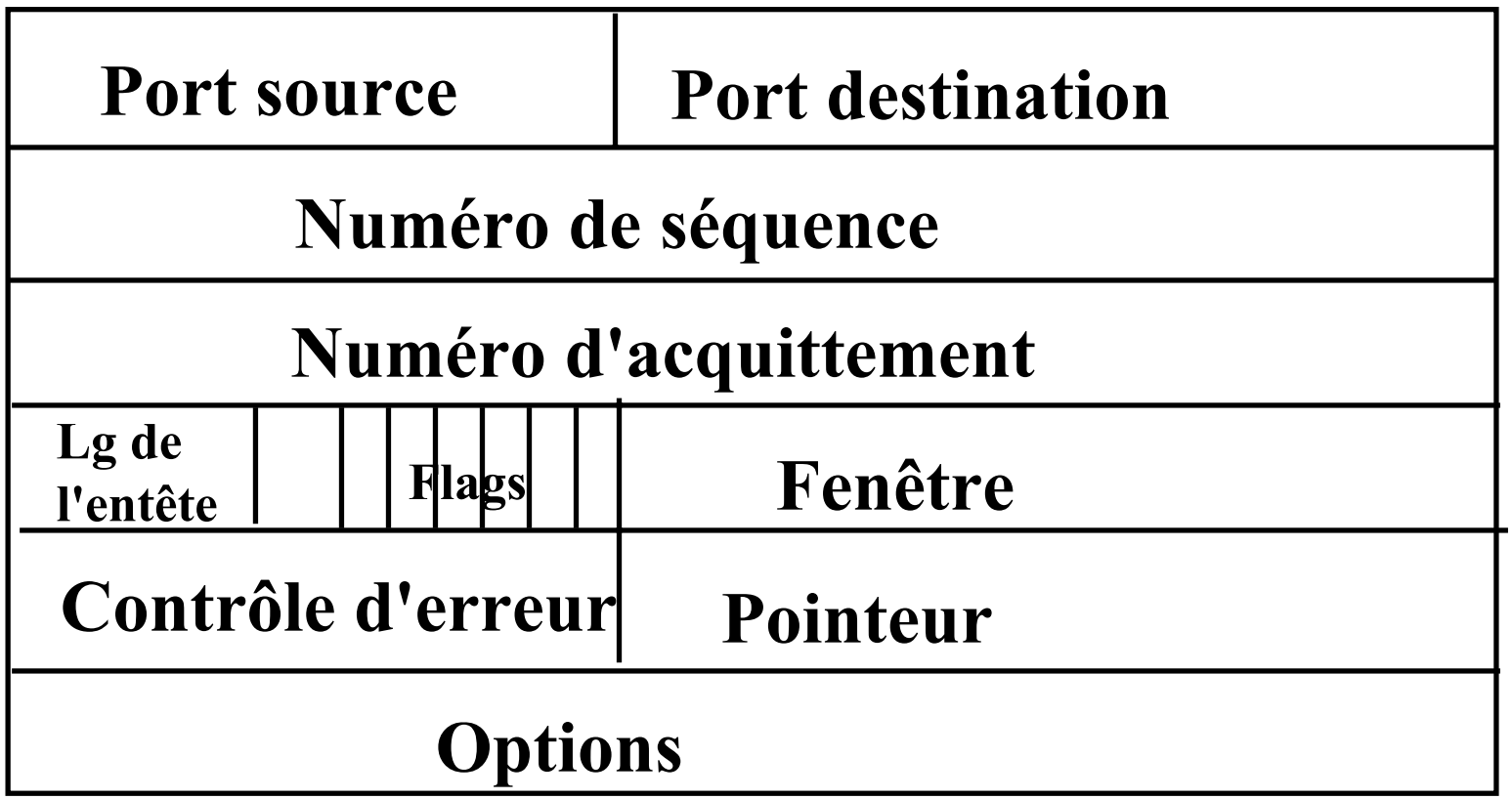
Ce problème n'a pas de solution exacte. Cependant pour diminuer la probabilité d'erreur sur la perception de l'état de la connexion qu'ont les deux entités de transport, on exécute un protocole à trois phases :



Un exemple de couche transport : TCP
les services du protocole TCP

- **Segmentation re-assemblage des messages**
  - **–Concaténation des paquets : flux d'octets bi-directionnel**
- **Rétablissement de l'ordre des paquets pour préserver l'ordre d'émision**
  - **– Numérotation des octets de données**
- **Multiplexage vers plusieurs applications (numéro de port)**
- **Service orienté connexion pour assurer la fiabilité des transmissions**
  - **–Ouverture et libération de la connexion**
- **Contrôle de flux : le récepteur demande à l'émetteur de réduire son débit d'émission**
- **Contrôle de congestion : les éléments actifs de réseau limitent le débit d'émission**
- **Détection des paquets erronés et perdus et récupération des erreurs par réémission**
- **Détection d'inactivité**

- **L'entête TCP:**



- **Flags: Urgent, Ack, Eom, Rst, Syn, Fin**

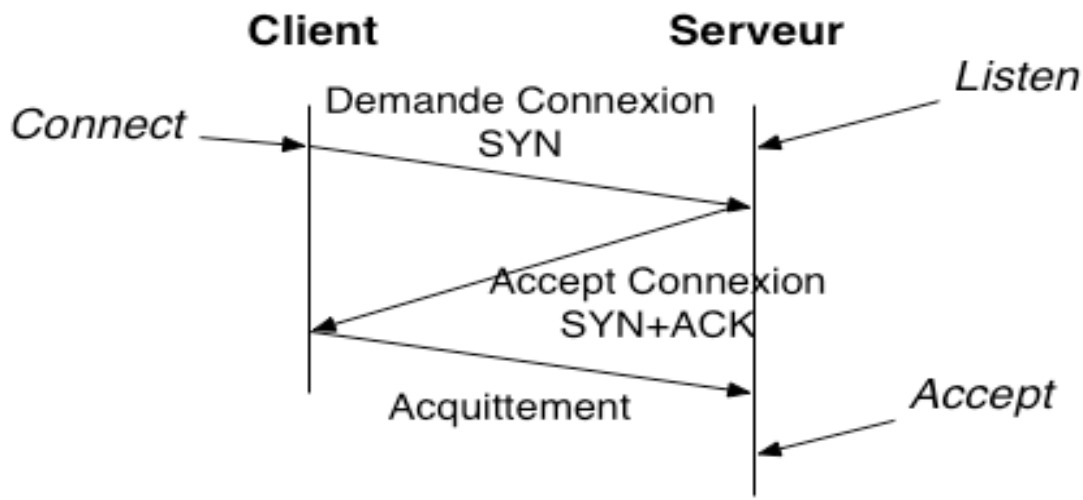
Etablissement d'une connexion TCP (1)

- **Des paquets particuliers pour ouvrir la connexion**
  - **Demande de connexion (Connection Request) : Flags SYN=1 et ACK =0**
  - **Acceptation de connexion (Connection Confirm) : Flags SYN= 1 et ACK=1**

Des options sont possibles :

- **Taille maximale des paquets**
- **Convention pour le contrôle de flux sur réseau haut débit ...**

- **Ouverture**
  - **Le protocole est un protocole à trois phases:**



Etablissement d'une connexion (2)

- **Trois paquets sont nécessaires pour garantir qu'il n'y ait pas d'ambiguité sur des demandes de connexions (doublon, timers de réémission)**
- **Le numéro de séquence initiaux permettent de différencier des demandes de connexions dupliquées**
- **Flag Reset en cas de duplication (ou autre incohérence)**
- **Les numéros de séquence initiaux sont calculés à partir d'une horloge système de période 4 microsecondes. Cela garantit l'unicité d'un paquet d'ouverture de connexion pendant 4 heures**
- **Les No Séquences servent à numéroter aussi les octets de données**
- **Problème: l'évolution des NoSeq des données dépend du débit de la connexion.**

–On peut arriver à des cas d'ambiguité par rapport aux id initiaux (zone interdite)

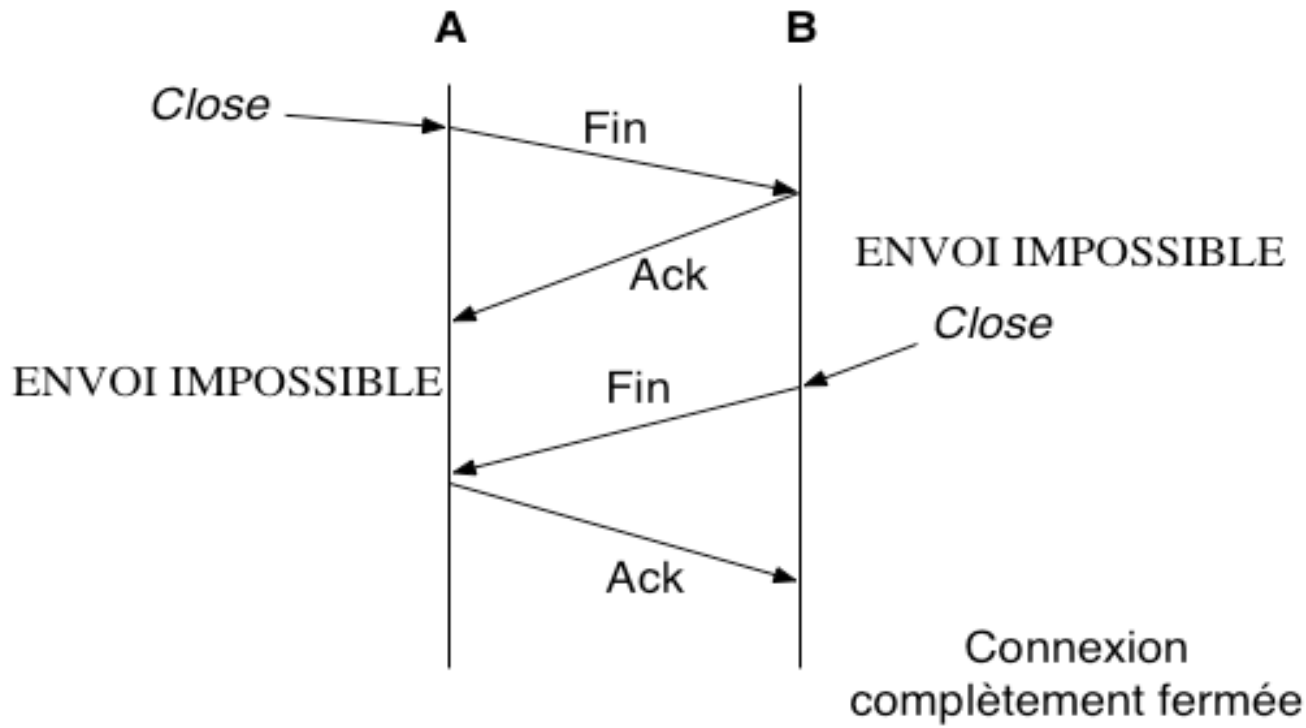
- **Solution : on attend la durée de vie maximale sur Internet avant de ré-ouvrir une connexion sur la même paire de ports (2 minutes)**

Le Transport TCP : Envoi de messages

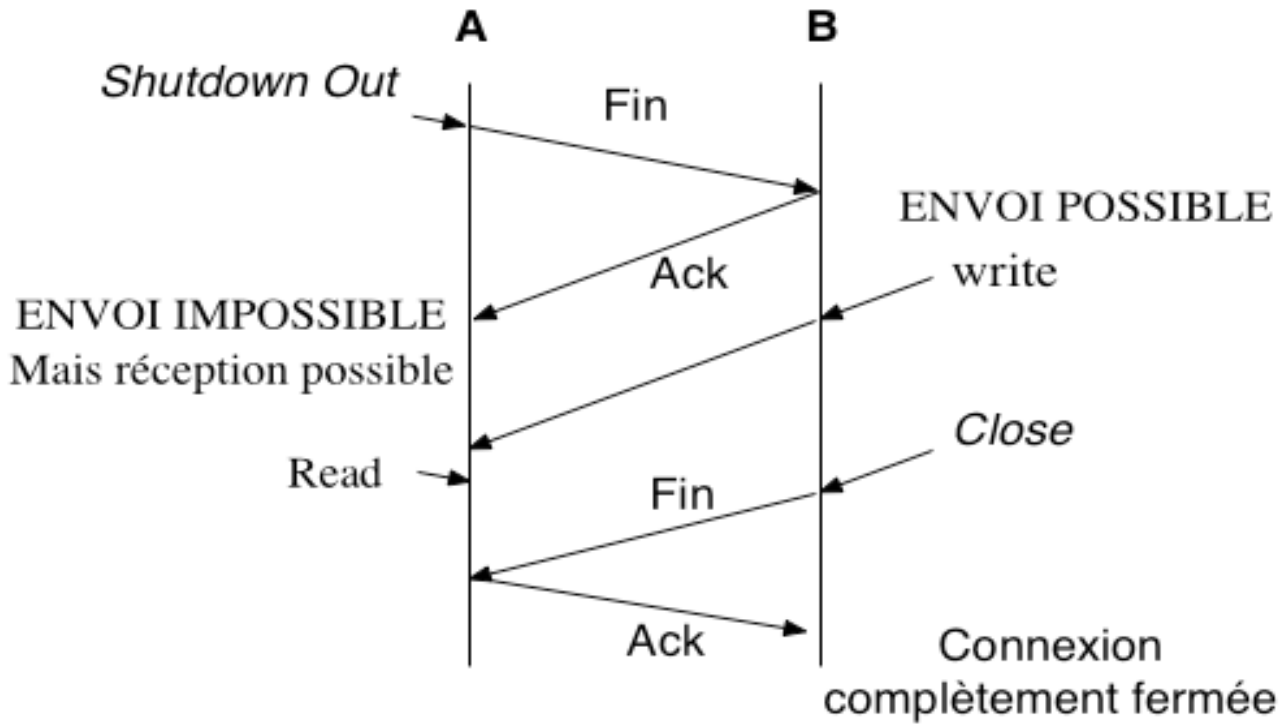
- **Segmentation:**
  - **Flag EOM pour dernier paquet d'un message segmenté (option pour délimiter des messages dans le flux d'octets)**
- **Contrôle de flux et d'erreur:**
  - **Flux d'octets et non de message**
    - » **Champs Numéro de Séquence : Numéro du premier octet**
    - » **Bit Ack= 1: Champs Numéro d'acquittement significatif**
    - » **Champs Numéro d'Acquittement : Numéro du dernier octet acquitté**
  - **Fenêtre à anticipation variable**
    - » **Champs Fenêtre : nombre d'octets pouvant être expédiés après le numéro d'acquittement**
- **Données Urgentes**
  - **Flag URG=1, Champs pointeur indique le début des octets à traiter en priorité**

Fermeture de connexion primitive close=>fermeture brutale

- **Flag FIN pour signifier la demande de fermeture**
- **L'entité distante acquitte pour qu'il n'y ai pas d'ambiguité sur l'état de**



Fermeture de connexion primitive shutdown=>fermeture non brutale



Les services du protocole UDP

Complémentaire à TCP : simplicité

- **Mode sans connexion (datagramme) : les messages sont indépendants les uns des autres**
- **Garanties minimales : aucun contrôle de flux ni récupération d'erreur : pas plus de garanties que IP**
- **En-tête UDP :**

Port source	**Port destination**
Longueur	**Détection d'erreur**
Données	

- **Exercice : Donnez les numéros de séquence et d'acquittement des paquets dans l'échange TCP suivant:**