Téléinformatique de base

Chapitre 7

TCP et UDP

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Objectifs d'apprentissage

- Savoir expliquer l'utilisation du protocole UDP
- Savoir décrire l'utilisation des numéros de port
- Comprendre les bases des mécanismes de TCP
  - Etablissement et terminaison de connexions
  - Numéros de séquence
  - Acquittement
  - Retransmission
  - Contrôle de congestion

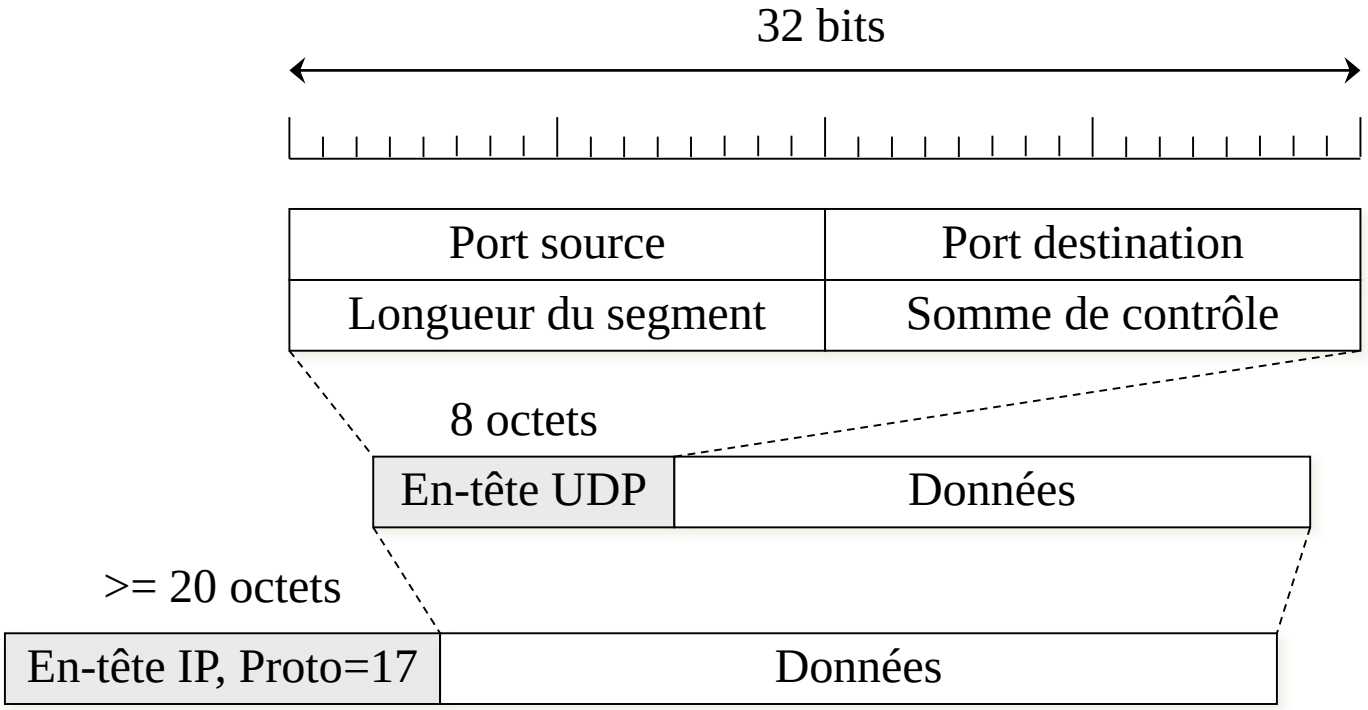
La couche Transport

- Communication de bout en bout
- Services offerts à la couche supérieure
  1. Service sans connexion
     - Protocole UDP
  2. Service avec connexion
     - Protocole TCP
     - Fiable (garantie de délivrance)
     - Délivrance dans l'ordre de l'émission
     - Régulation de la vitesse (contrôle de congestion)

Le protocole UDP

- Fonctionnalités
  - Adressage d'applications à l'aide de numéros de port
  - Contrôle d'erreur optionnel (obligatoire avec IPv6)
  - Transmission non fiable
    - Sans connexion
    - Sans acquittement ou retransmission
    - Sans contrôle du débit
- Utilisé pour
  - Les transmissions multimédia (tolèrent quelques pertes)
  - Les transmissions multicast (multicast toujours sans connexion)
  - Les échanges courts, comme DNS (pas de connexion)

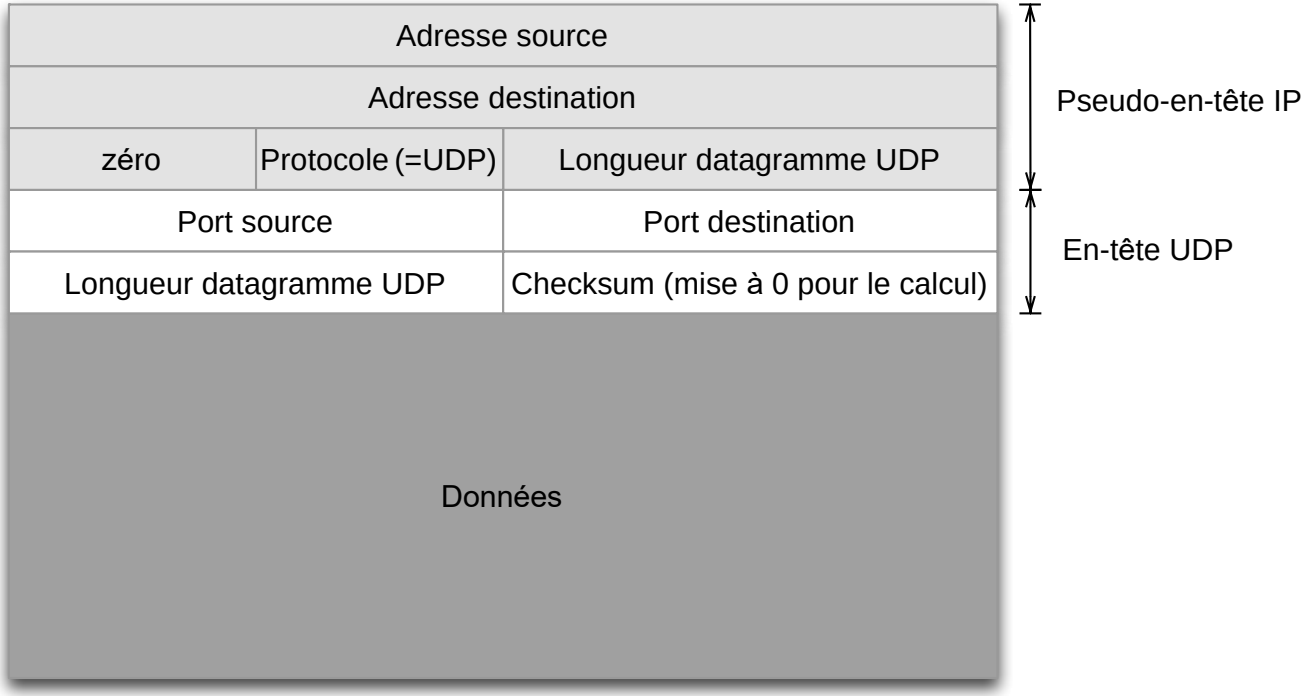
Format des datagrammes UDP



- Seulement 4 champs dans l'en-tête UDP, 8 octets en total
- Longueur maximum d'un datagramme: 65'535 octets

Somme de contrôle

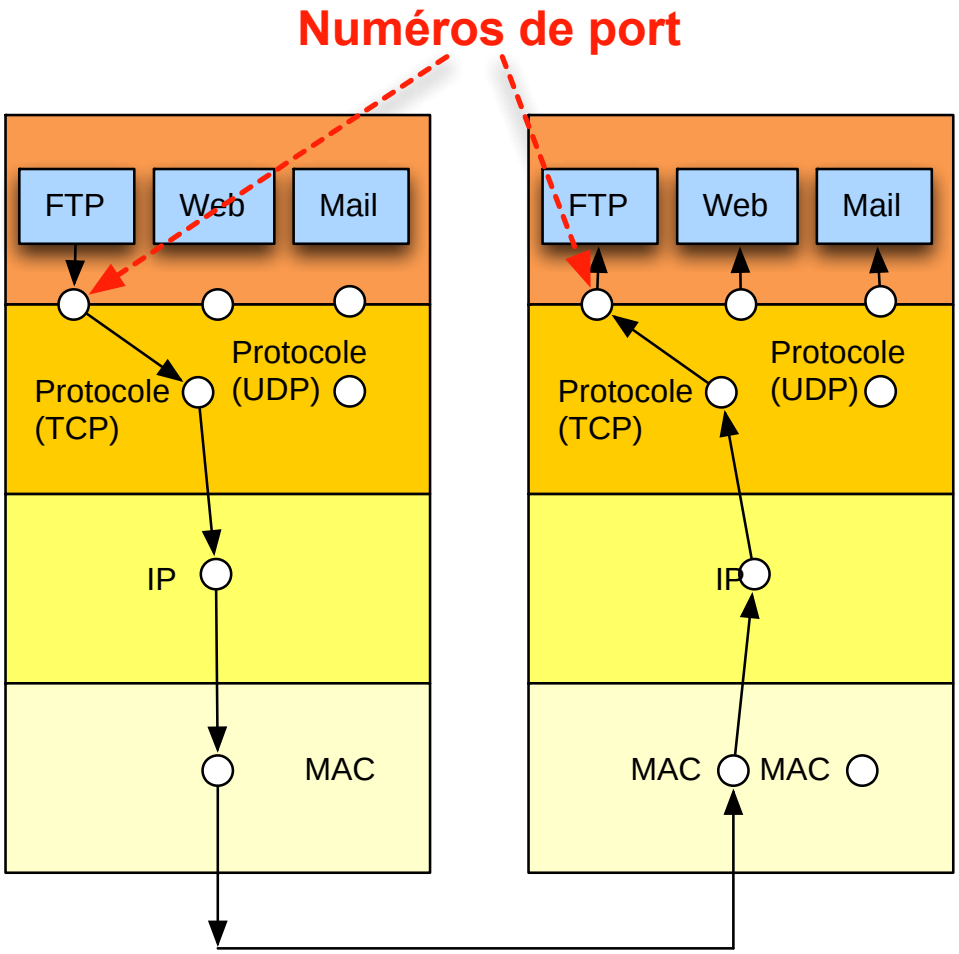
- La somme de contrôle est calculée sur
  - Le ‘pseudo-header’ IP (quelques champs de l’en-tête IP)
  - L’en-tête UDP
  - Les données encapsulées



- Protection contre
  - Des erreurs bits dans les données
  - Des adresses/ports erronées (paquet arrive au mauvais destinataire)

Numéro de ports

- Les numéros de ports sont les adresses de la couche Transport
- Permettent d'adresser plusieurs applications sur la même machine
- Entiers sur 16 bits
- Utilisés par TCP et UDP
- TCP et UDP peuvent réutiliser les mêmes ports



Les ports TCP / UDP

- Deux types de numéros de ports
  - Ports bien connus (1-1023)
    - Assignés à des applications standard
    - Typiquement utilisés par les applications serveur

Service	Port	Protocole
SSH	22	TCP
SMTP	25	TCP
DNS	53	UDP
HTTP	80	TCP

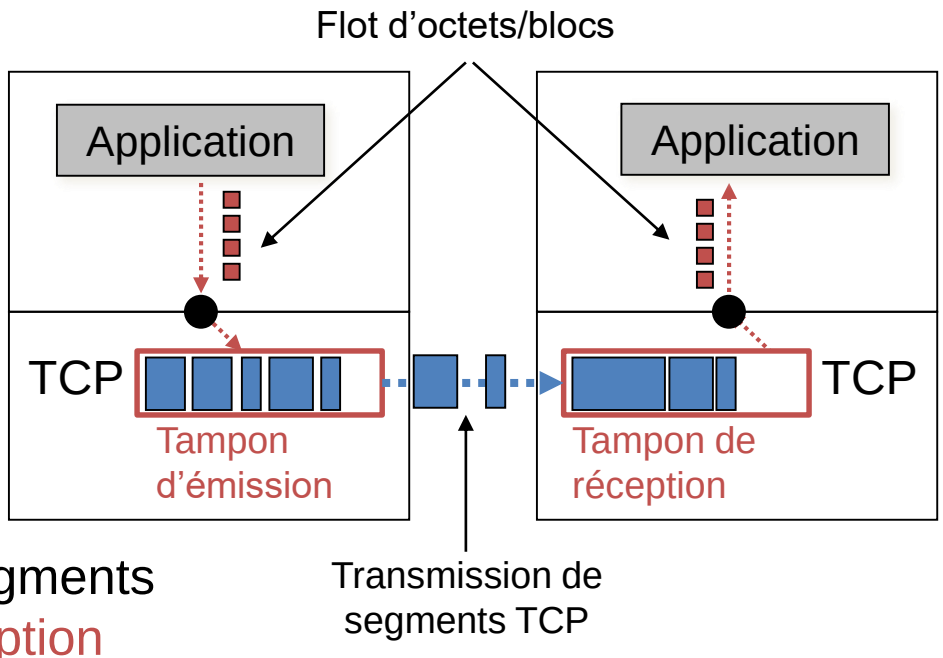
  - Ports éphémères (1024 – 65535)
    - Typiquement utilisés par les applications client
    - Assignés dynamiquement par TCP/UDP

Le protocole TCP

- TCP: *Transmission Control Protocol*
- Fonctionnalités principales
  - Transmission fiable de bout en bout
    - Numéros de séquence
    - Acquittements
    - Retransmission
  - Etablissement et terminaison de connexions
    - Connexions bidirectionnelles et point-à-point
  - Régulation du débit
    - Contrôle de flux : adaptation de la vitesse au récepteur
    - Contrôle de congestion : adaptation de la vitesse au réseau

Transfert tamponné à flot d'octets

- TCP offre à la couche supérieure un service à flot d'octets
  1. L'application passe des blocs de données à TCP
  2. TCP les met dans un tampon d'émission
  3. TCP regroupe les données en segments à transmettre
  4. Le récepteur TCP place les segments reçus dans un tampon de réception
  5. L'application lit des blocs de données, sans tenir compte de segments



- La délimitation des messages de l'application n'est pas respectée

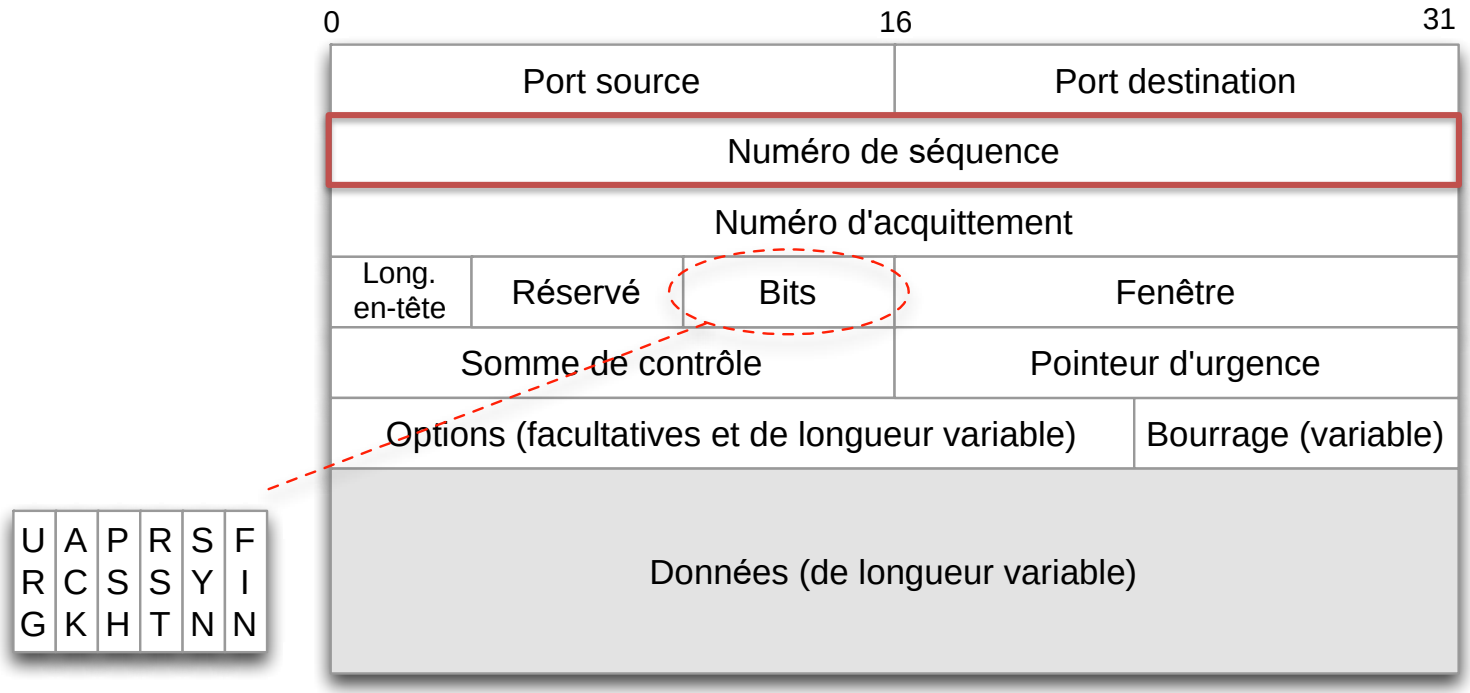
Transmission fiable

- Garantie de délivrance des données, dans l'ordre de l'émission

- Mécanismes nécessaires
  1. Numéros de séquence
  2. Acquittement
  3. Retransmission

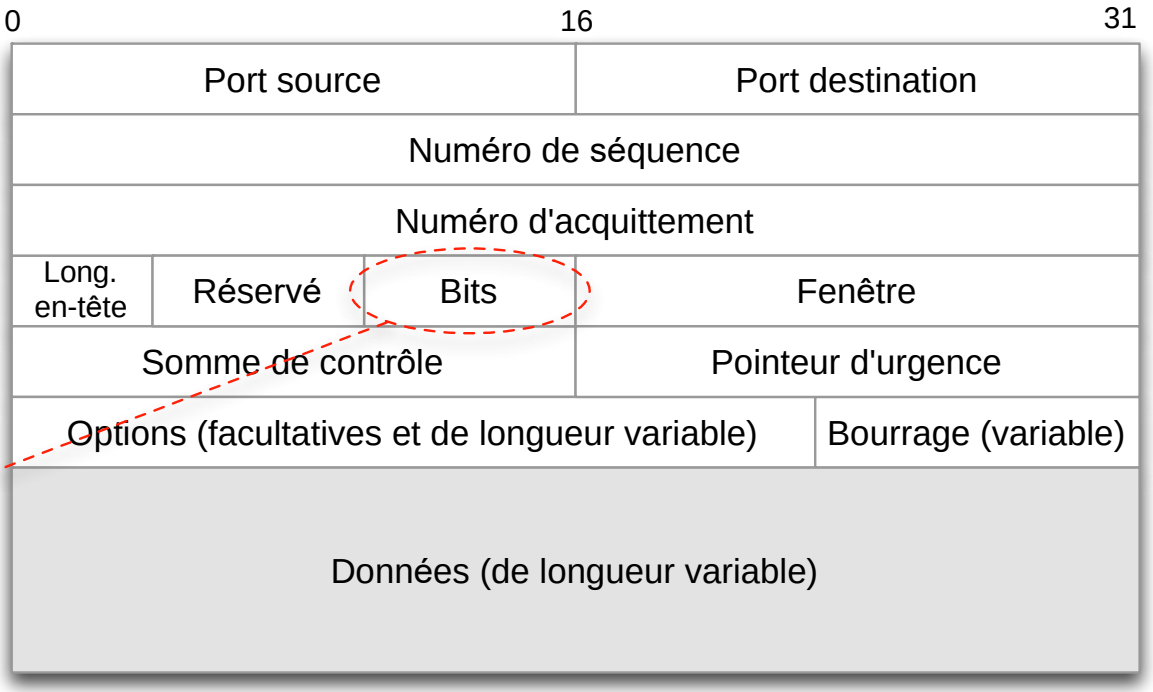
Numéros de séquence

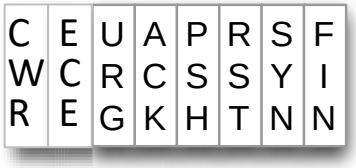
- Les octets sont numérotés, non pas les segments
- L'émetteur indique le numéro du premier octet des données du segment



Acquittement

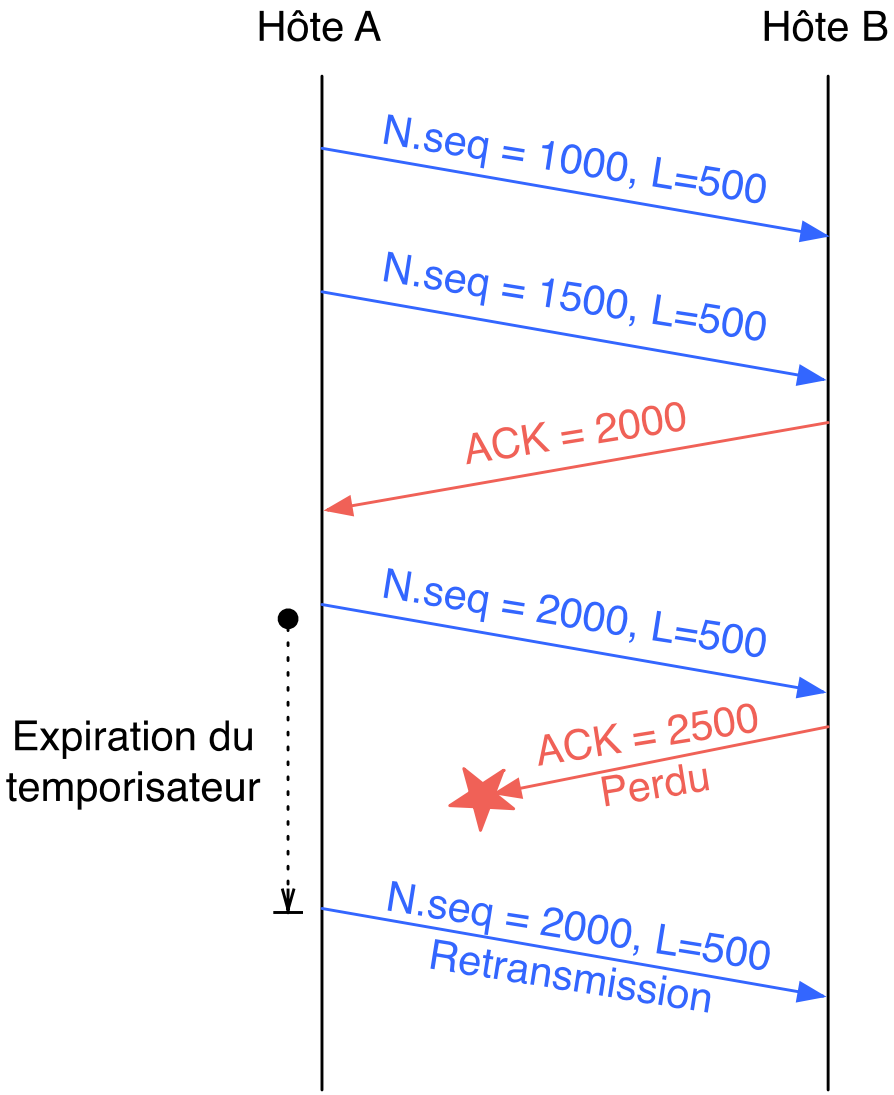
- Le numéro d'acquittement indique le prochain octet attendu par le récepteur
- Un acquittement peut être envoyé séparément ou combiné avec des données
- Les acquittements sont cumulatifs :
  - Un acquittement confirme la réception de toutes les données jusqu'à ce point





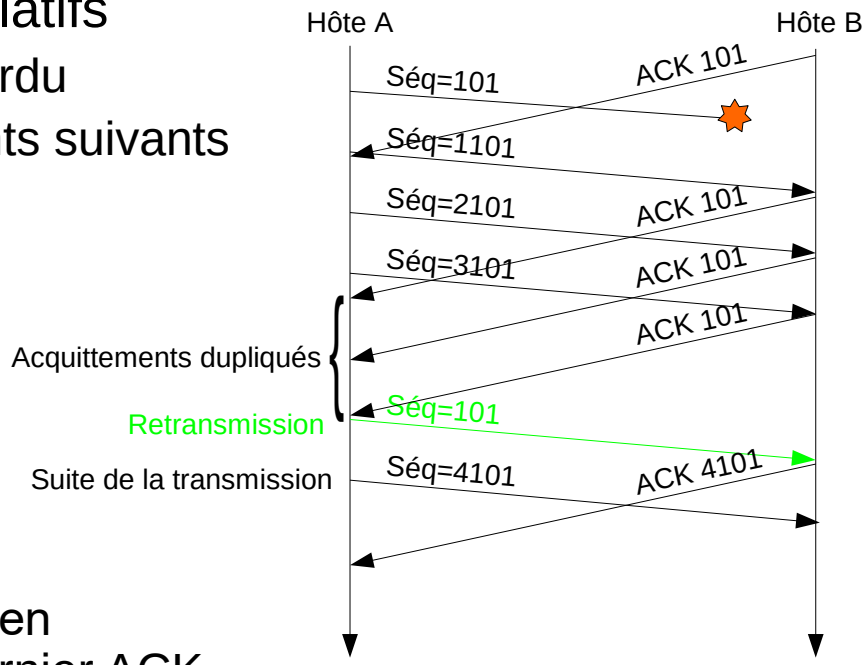
Acquittements et retransmission

- L'émetteur gère un temporisateur de retransmission pour chaque segment envoyé
- Si le temporisateur expire avant la réception d'un acquittement, le segment est retransmis
- TCP a un mécanisme pour adapter la durée du temporisateur



Retransmission rapide

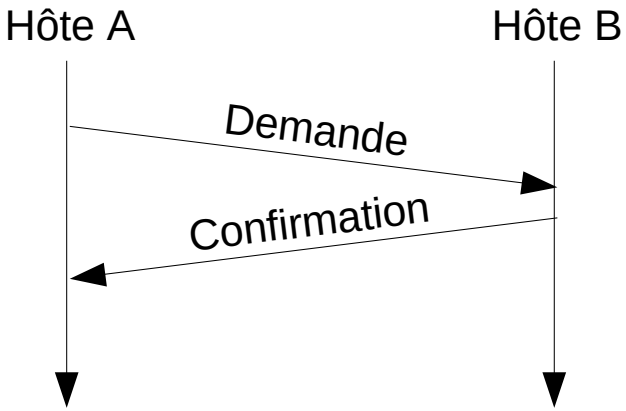
- Problème des acquittements cumulatifs
  - Un segment intermédiaire a été perdu
  - Comment signaler que les segments suivants sont arrivés ?



- Solution :
  - Lors de la réception d'un segment en désordre, le récepteur répète le dernier ACK ('*ACK dupliqué*')
  - Si l'émetteur reçoit 3 ACK dupliqués, il retransmet le segment sans attendre un timeout

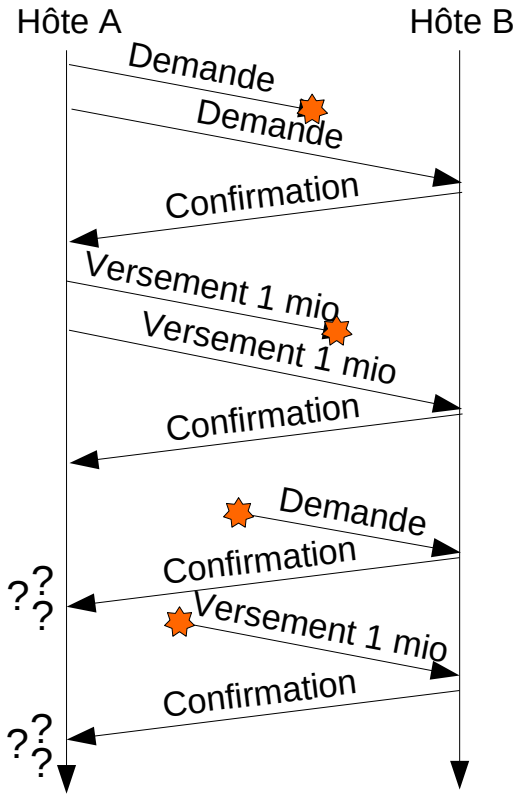
Etablissement de connexions

- C'est simple ? Non !
- Protocole simpliste :
  - Etablissement de connexion en deux temps
    1. Demande d'établissement
    2. Confirmation



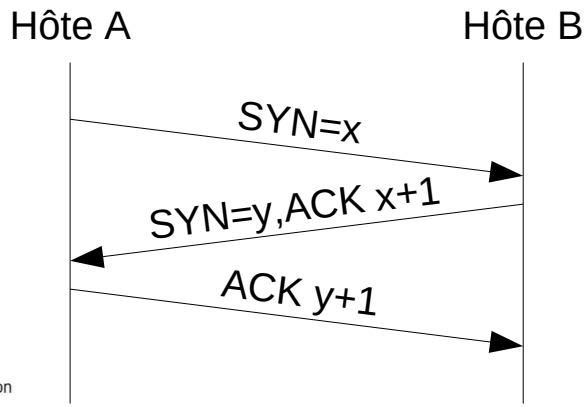
Scénario

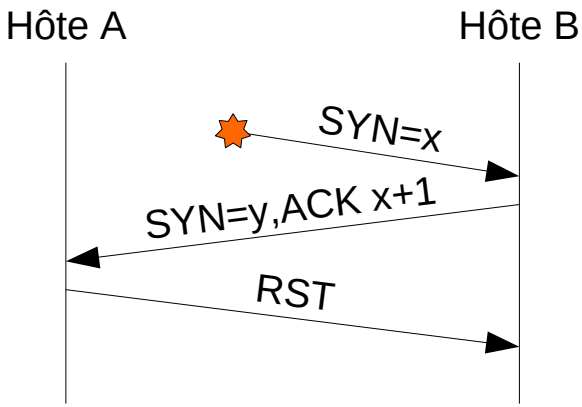
- Ordre de versement à une banque
- Certains paquets du client sont retardés et retransmis



Etablissement de connexions dans TCP

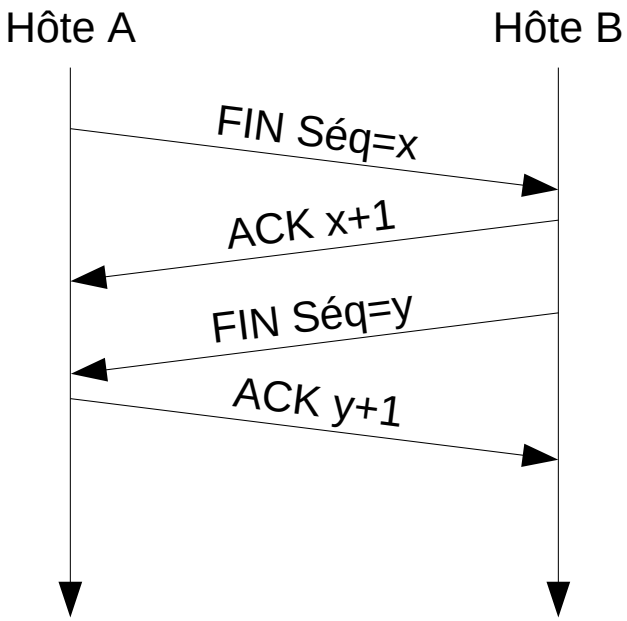
- Le protocole doit tenir compte de la possibilité de doublons
- Mécanisme de TCP : *Three way handshake*
  (établissement de connexion en trois temps)
  - Les hôtes se mettent d'accord sur les numéros de séquence initiaux
  - Lors d'une erreur, un message RST (RESET) permet d'interrompre l'établissement de connexion





Libération d'une connexion dans TCP

- Les connexions TCP sont bidirectionnelles
  - Libération séparément dans les deux sens
    - Un hôte qui n'a plus de données à transmettre peut fermer la connexion dans une direction
    - L'autre hôte peut continuer à transmettre
  - Deux segments pour libérer un sens de la connexion: FIN + ACK



Contrôle de flux et de congestion

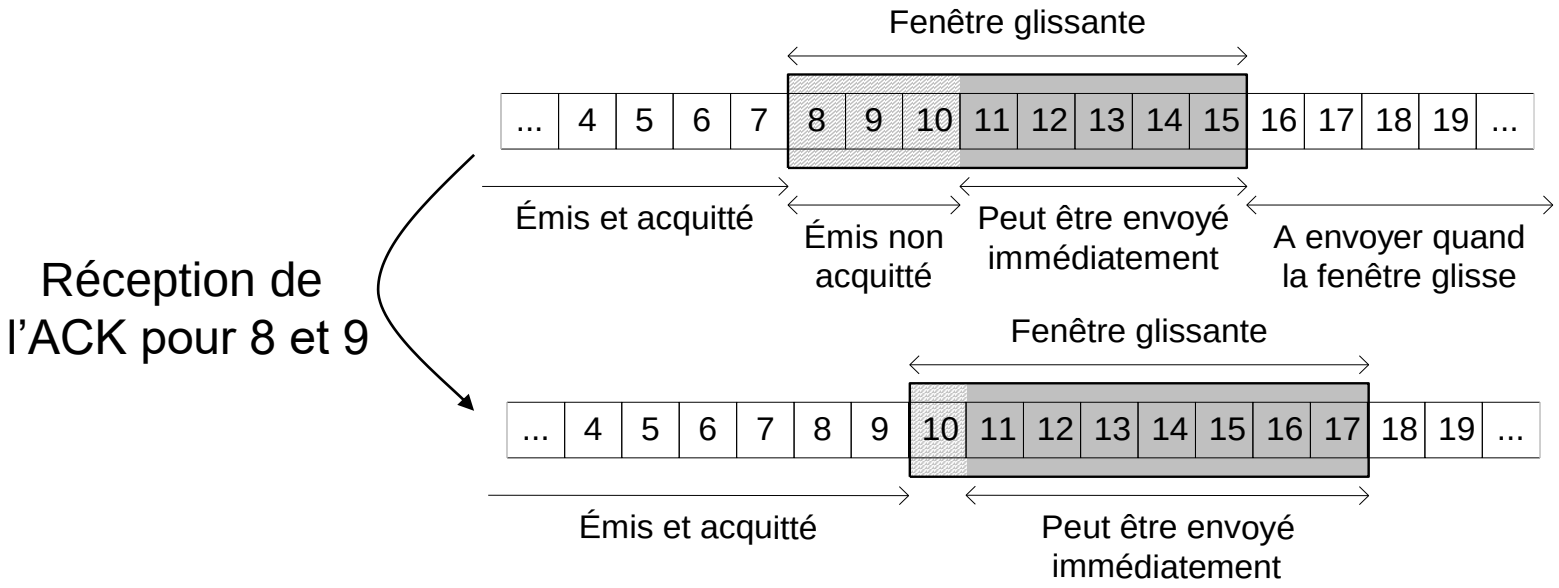
- Objectif :

  Régulation de la vitesse de transmission

- Contrôle de flux
  - Adaptation à la vitesse du récepteur
  - Évite qu'un émetteur rapide surcharge un récepteur lent
- Contrôle de congestion
  - Adaptation à la vitesse du réseau
  - Évite des pertes excessives de paquets à cause d'une surcharge du réseau

Protocole de la fenêtre glissante

- Améliore le taux d'utilisation
  - Permettant à l'émetteur d'envoyer plusieurs paquets avant de devoir attendre un accusé de réception
- Principe
  - Les données dans la fenêtre peuvent être envoyées sans attendre d'acquittement
  - La réception d'un acquittement permet de glisser la fenêtre à droite

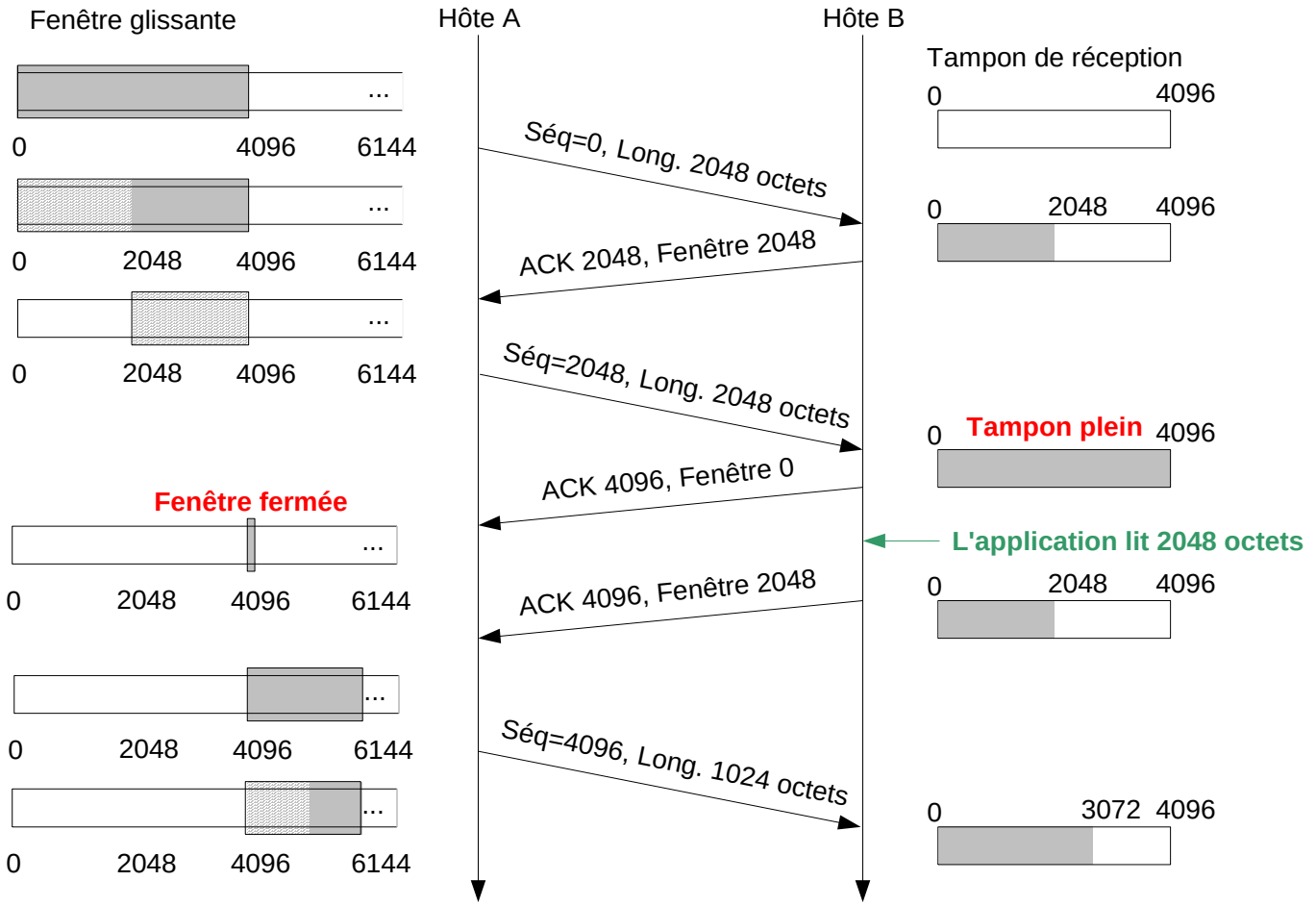


Contrôle de flux dans TCP

- Basé sur un protocole à fenêtre glissante mais avec une taille de fenêtre variable
- La fenêtre utilisable correspond à la place libre dans le tampon du récepteur
- Chaque accusé de réception indique en plus la taille de la fenêtre (window advertisement)

➢ En variant la taille de la fenêtre, le récepteur peut contrôler la vitesse de transmission de l'émetteur

Exemple du contrôle de flux dans TCP



Contrôle de congestion

- État de congestion
  - Le réseau n'est plus en mesure de transporter tout le trafic injecté et perd des paquets

- Danger de l'amplification d'une congestion par TCP
  - TCP réagit à une perte avec une retransmission
  - La retransmission peut augmenter la charge du réseau

➢ Le contrôle de congestion de TCP doit optimiser le débit de transmission sans mettre en danger la stabilité du réseau

Bases du contrôle de congestion de TCP

- La fenêtre de congestion (cwnd, *congestion window*)
  - Gérée par l'émetteur pour limiter le débit d'émission
  - TCP adapte la taille de cwnd au niveau de congestion détecté
- Combinaison du contrôle de flux et de congestion

$$\text{Fenêtre effective} = \min\left(\text{Annonce de fenêtre}, cwnd\right)$$

Principe du contrôle de congestion

- Démarrage lent : Pour détecter le débit optimal, TCP commence avec une petite fenêtre de congestion et augmente rapidement le débit
- Évitement de congestion : Dès que TCP s'approche de la zone de congestion, TCP augment le débit plus lentement
- Accroissement additif - décroissance multiplicative : Dès qu'une perte est détectée TCP ralentit rapidement et augmente à nouveau lentement

Démarrage lent (*Slow Start*)

- Motivation
  - TCP doit fonctionner correctement pour n'importe quelle capacité du réseau (entre bits/s et Gb/s)
  - Il faut éviter de surcharger un lien lent au départ
    - ➢ Petite fenêtre cwnd au début
  - Il faut rapidement arriver à exploiter la capacité de liens importants
    - ➢ Augmentation rapide de cwnd

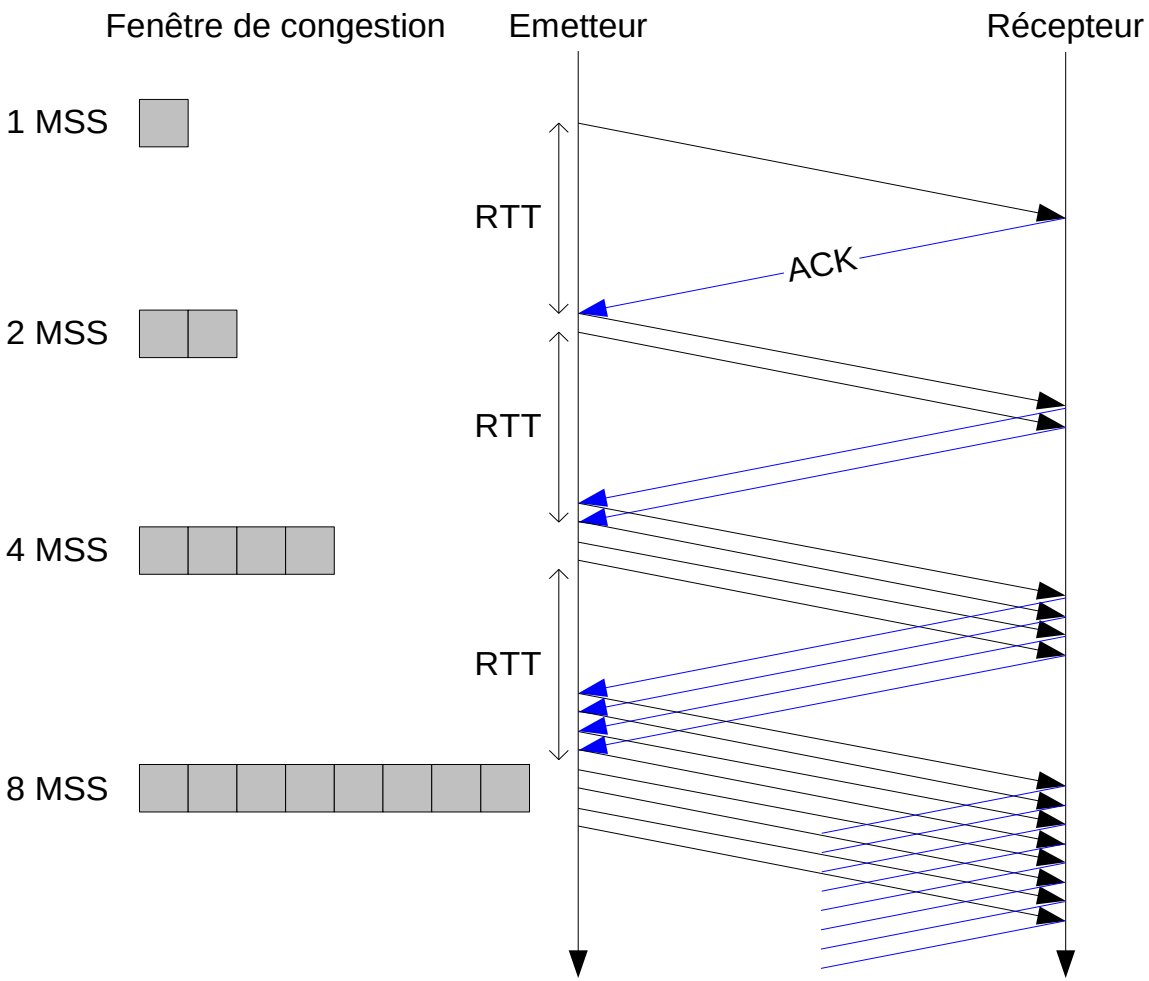
Algorithme Slow Start

- Initialement cwnd = 1 MSS
- Ensuite cwnd est incrémenté de 1 MSS par acquittement reçu

(MSS: *Maximum Segment Size*: taille maximum d'un paquet)

➢ Cwnd double chaque RTT (délai aller-retour)

Exemple de Slow Start



Évitement de congestion (*Congestion Avoidance*)

- Dans Slow Start, la taille de cwnd augmente exponentiellement
- Augmentation doit ralentir quand TCP s'approche du 'débit optimal'
- Un seuil d'évitement de congestion indique quand on s'approche d'une congestion
  - Paramètre ssthresh de TCP

Algorithme Congestion Avoidance

- Dès que cwnd > ssthresh, cwnd est agrandie linéairement
- Pour chaque acquittement reçu :

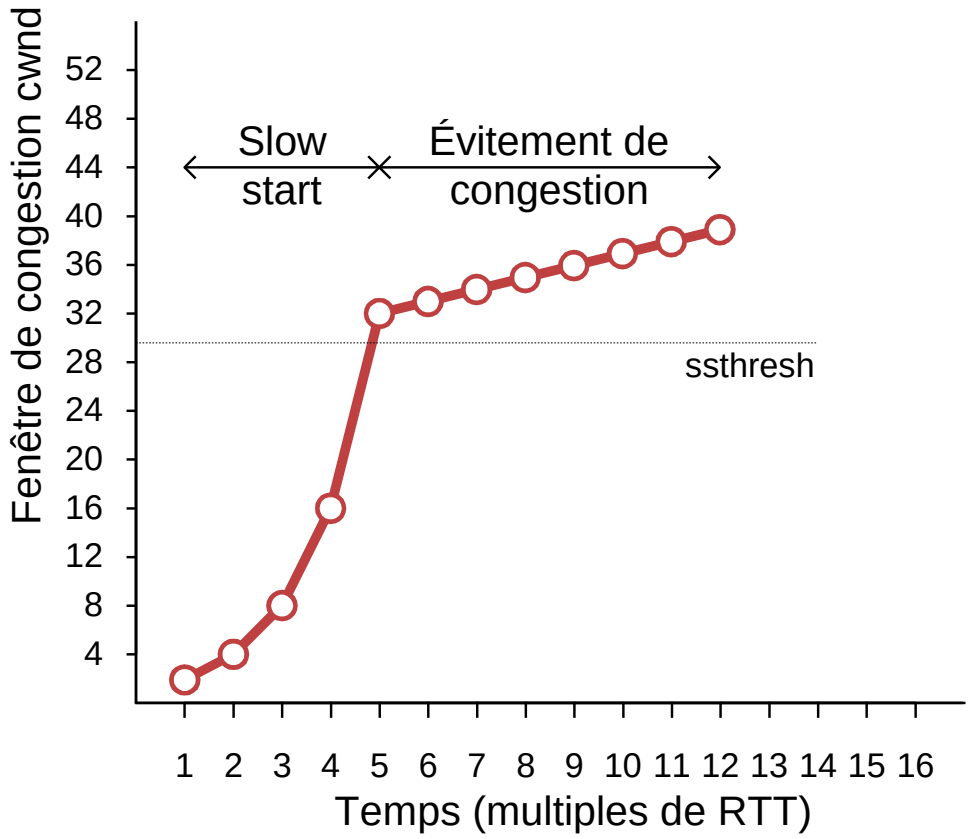
$$cwnd \leftarrow cwnd + \frac{MSS \cdot MSS}{cwnd}$$

➢ Augmentation d'un MSS par RTT

Exemple
Slow Start et Congestion Avoidance

- Au-dessous de ssthresh:
  Slow Start
  - Cwnd double chaque RTT

- Au-dessus de ssthresh:
  Congestion Avoidance
  - Cwnd croît d'un MSS chaque RTT



Variation du seuil d'évitement de congestion

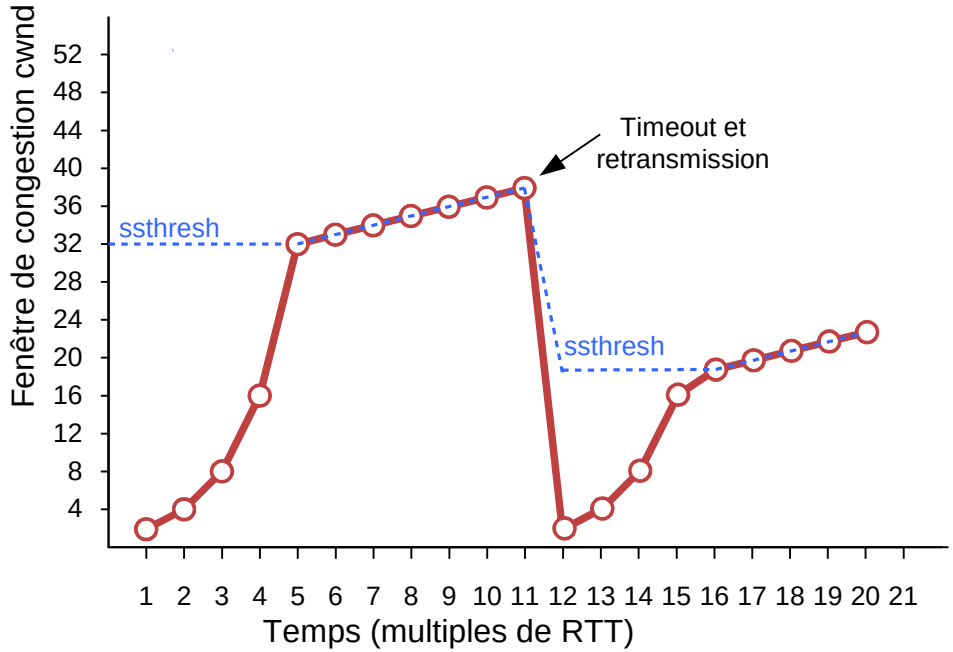
Comment déterminer la valeur du seuil de congestion ssthresh ?

- Ssthresh doit représenter la taille 'optimale' de la fenêtre de congestion

1. Lorsqu'il n'y a pas de congestion, ssthresh croît linéairement avec cwnd → accroissement additif
   - Les signal que le débit optimal a été dépassé est la perte d'un paquet
     - Théorie des files d'attente : Pour garantir la stabilité du réseau, le débit doit diminuer de manière exponentielle
2. Lorsqu'une perte a été détectée, ssthresh est diminué à la moitié → décroissance multiplicative
3. TCP (Reno) recommence avec Slow Start (après un timeout)
   - Pour éviter des rafales de retransmissions

Comportement de cwnd et ssthresh

- Trois phases
  - Slow Start avec une croissance exponentielle de cwnd
  - Congestion avoidance (accroissement additif de ssthresh)
  - Diminution de ssthresh lors d'une perte (décroissance multiplicative de ssthresh)



Résumé du contrôle de congestion dans TCP

- La taille de la fenêtre de congestion est variée en fonction de la congestion du réseau
  - Une congestion est détectée à cause de pertes de paquets
- **Slow Start** : croissance exponentielle du débit
  - Au début de la connexion et après un timeout de retransmission (→ congestion sévère)
  - Permet d'augmenter rapidement le débit
- **Congestion avoidance**: croissance linéaire du débit
  - Dès que le débit s'approche à la capacité disponible
- **Seuil d'évitement de congestion** ssthresh
  - Indique la zone critique où une congestion est possible
  - Accroissement additif et décroissance multiplicative de ssthresh
  - Ssthresh oscille entre le débit maximum et la moitié de ce débit