La couche transport du modèle OSI

La couche transport du modèle OSI

Introduction

L'objectif ?

- Préparation des données pour le transport sur le réseau

Fonctions principales

- Permettre de mettre en relation les applications entre-elles
- Éventuellement segmenter les données provenant de la couche Applicative avant de les transmettre à la couche Réseau
- Ré-assembler les segments sur la destination finale
- Identification des applications

- 2 protocoles exemples
  - TCP – Transmission Control Protocol
  - UDP – User Datagram Protocol

Fonctions complémentaires

- Vérification d'erreurs
- Echange orienté connexion
  - Etablissement d'une connexion entre les
- Echange avec acheminement fiable
  - Possibilité de retransmettre des paquets perdus
  - 3 fonctions à assurer
    - Suivi des données
    - Acquittement des données reçues
    - Re-transmission des données non acquittées
- Control de flux
  - Régulation de la quantité d'informations transmises pour éviter des saturations

Acheminement fiable ?

- Fonction pas toujours utile

Nécessité :
- Rapidité
- Faible sur-cout
- Acquittement non nécessaire
- Inutilité de ré-expédition
- Segment analysé dans l'ordre d'arrivée

Nécessité :
- Fiabilité
- Acquittement
- Ré-expédition
- Segment délivré dans le même ordre qu'à l'expédition

C'est le développeur de l'application qui choisi le protocole dont il a besoin

La couche transport du modèle OSI

Transmission Control Protocol

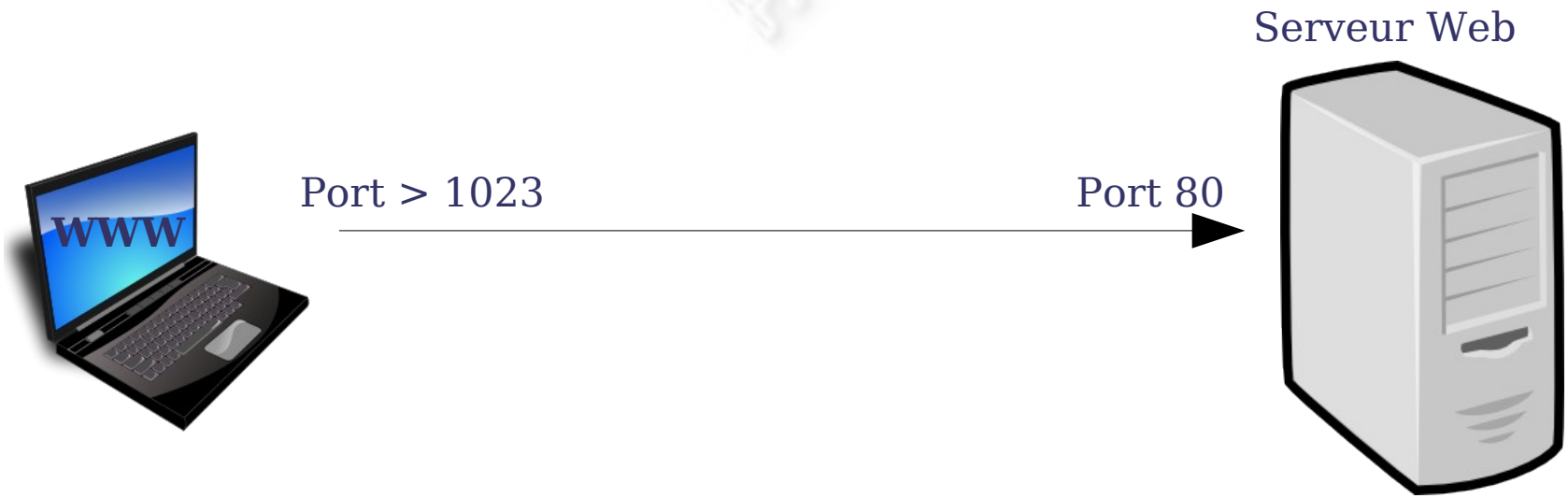
En-tête TCP

- TCP – Transmission Control Protocol

Port Source (16 bits)	Port Destination (16 bits)
Numéro de Séquence (32 bits)	
Numéro d'acquittement (32 bits)	
Long. En-tête (4), Réservé (6), Flags (6)	Taille fenêtre (16 bits)
Checksum (16 bits)	Urgent (16 bits)
Options (0 à 32 bits si présent)	
Données de la couche applicative	

Les ports de communication

- Chaque application utilise un port de communication
- Les ports <= 1023 sont des ports dit administrateur
  - Généralement utilisé par des services pour écouter les connexions entrantes
- Les ports > 1023 sont des ports dit utilisateur
  - Généralement utilisé par les applications pour communiquer avec l'extérieur



Les ports standards

- Défini par l'Internet Assigned Numbers Authority (IANA)
  - De 0 à 1023 : port dit bien connu
  - 1024 à 49151 : port enregistrée auprès de l'IANA
  - 49152 à 65535 : usage libre, port à usage privé ou dynamique

Les plus connus (TCP) :
- FTP : 21
- HTTP : 80
- Telnet : 23
- SMTP : 25
- POP3 : 110
- HTTPS : 443

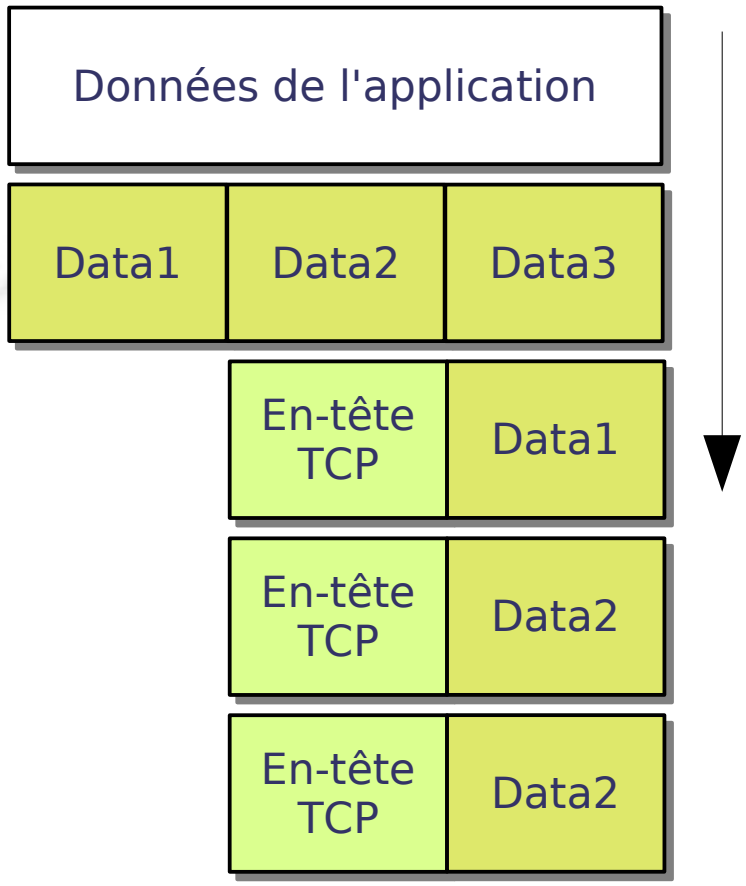
Longueur et checksum

- Codé sur 4 bits, il indique le nombre de mot de 32 bits que contient l'en-tête
  - Sans option, l'en-tête fait 5 x 32 bits, soit une longueur de 5 mots de 32 bits
- Checksum est calculé pour la totalité du segment (en-tête + données)

Port Source (16 bits)	Port Destination (16 bits)
Numéro de Séquence (32 bits)	
Numéro d'acquittement (32 bits)	
Long. En-tête (4), Réservé (6), Flags (6)	Taille fenêtre (16 bits)
Checksum (16 bits)	Urgent (16 bits)
Options (0 à 32 bits si présent)	
Données de la couche applicative	

Fonctionnement

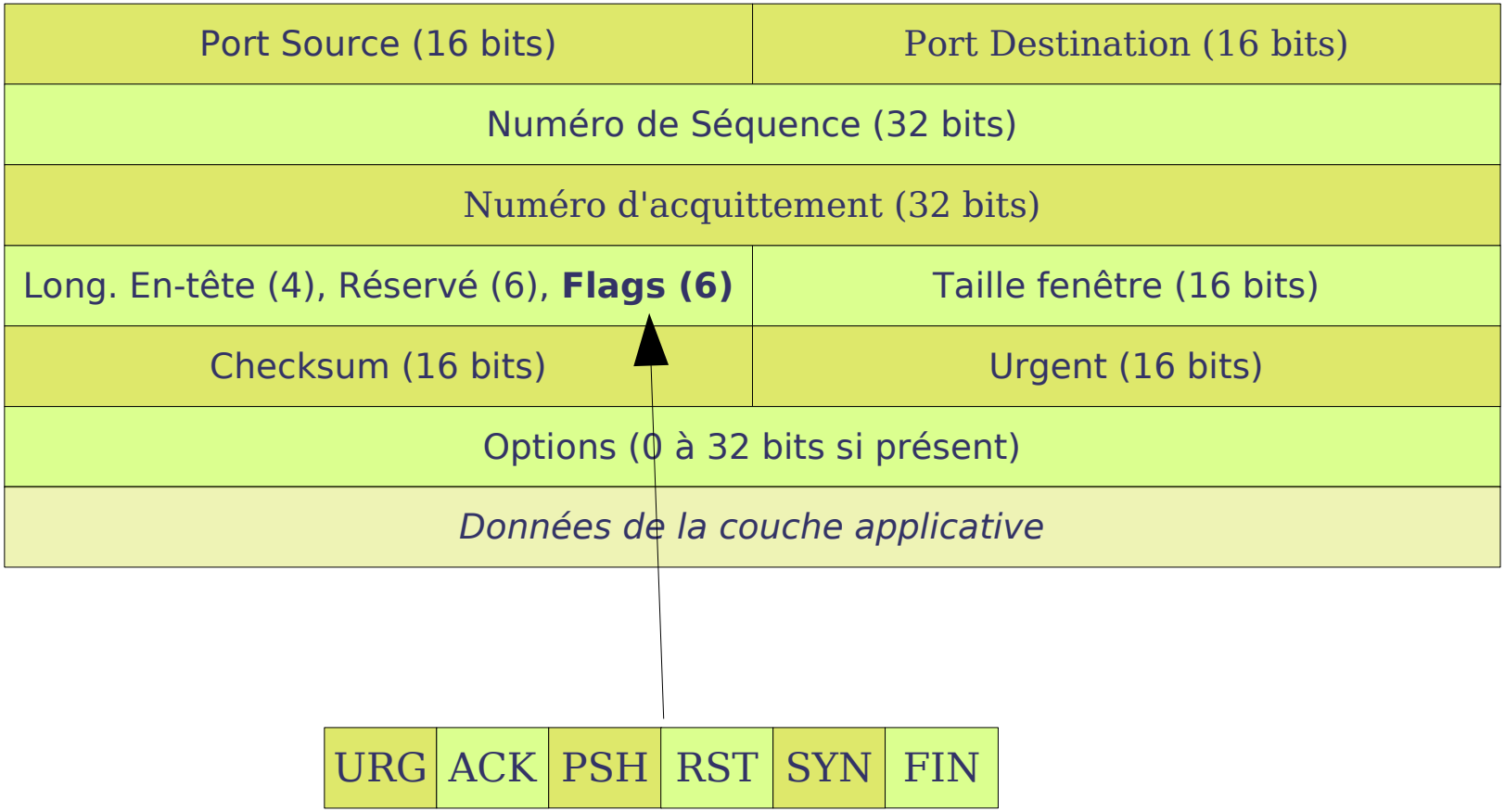
- Découpage des données applicatives lors de leurs « entrées » dans la couche transport
  - Cette taille est **définit** suivant plusieurs critères dans le **système**
  - Elle est **éventuellement envoyée** lors de l'**établissement de la connexion**
  - Utilisation dans l'en-tête TCP de l'**option MSS** (Maximum Size Segment) pour spécifier cette taille
  - **Par défaut**, elle est supposée être de **536 octets**



Les types de segment TCP

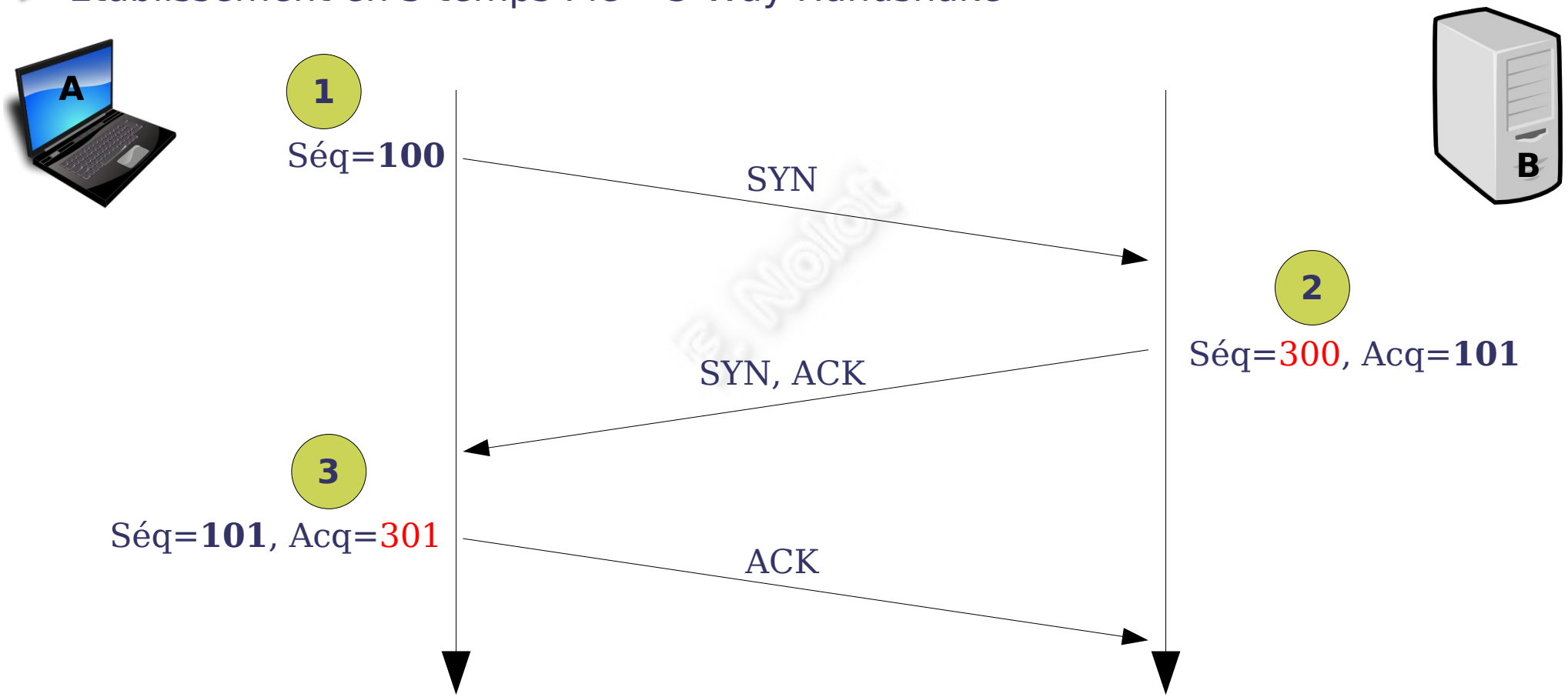
- Plusieurs segments TCP sont distingués en fonction de leur rôle
  - Segment **SYN** : pour établir une connexion
    - Rappel : TCP fonctionne en mode connecté
  - Segment **ACK** : pour acquitter un segment reçu
    - Rappel : TCP permet de faire des échanges fiables
  - Segment **FIN** : le récepteur a terminé d'envoyer des données. C'est la fin de la connexion
    - Permet de s'assurer qu'il n'y a plus de données à expédier
  - Segment **PSH** : signale au récepteur qu'il peut envoyer les données reçues à l'application dès que possible
    - Pour gagner du temps, les segments ne vont pas attendre dans la couche Transport
  - Segment **RST** : pour ré-initialiser une connexion
  - Segment **URG** : pour indiquer que le segment est urgent
- Cette identification se fait grâce aux flags de l'en-tête TCP

Les flags de l'en-tête TCP



Établissement de la connexion

- Établissement en 3 temps : le « 3-Way Handshake »



Les segments SYN « consomme » 1 numéro de séquence

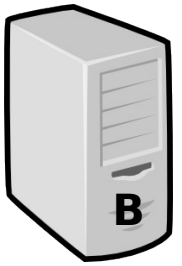
Les numéros de séquence et d'acquittement

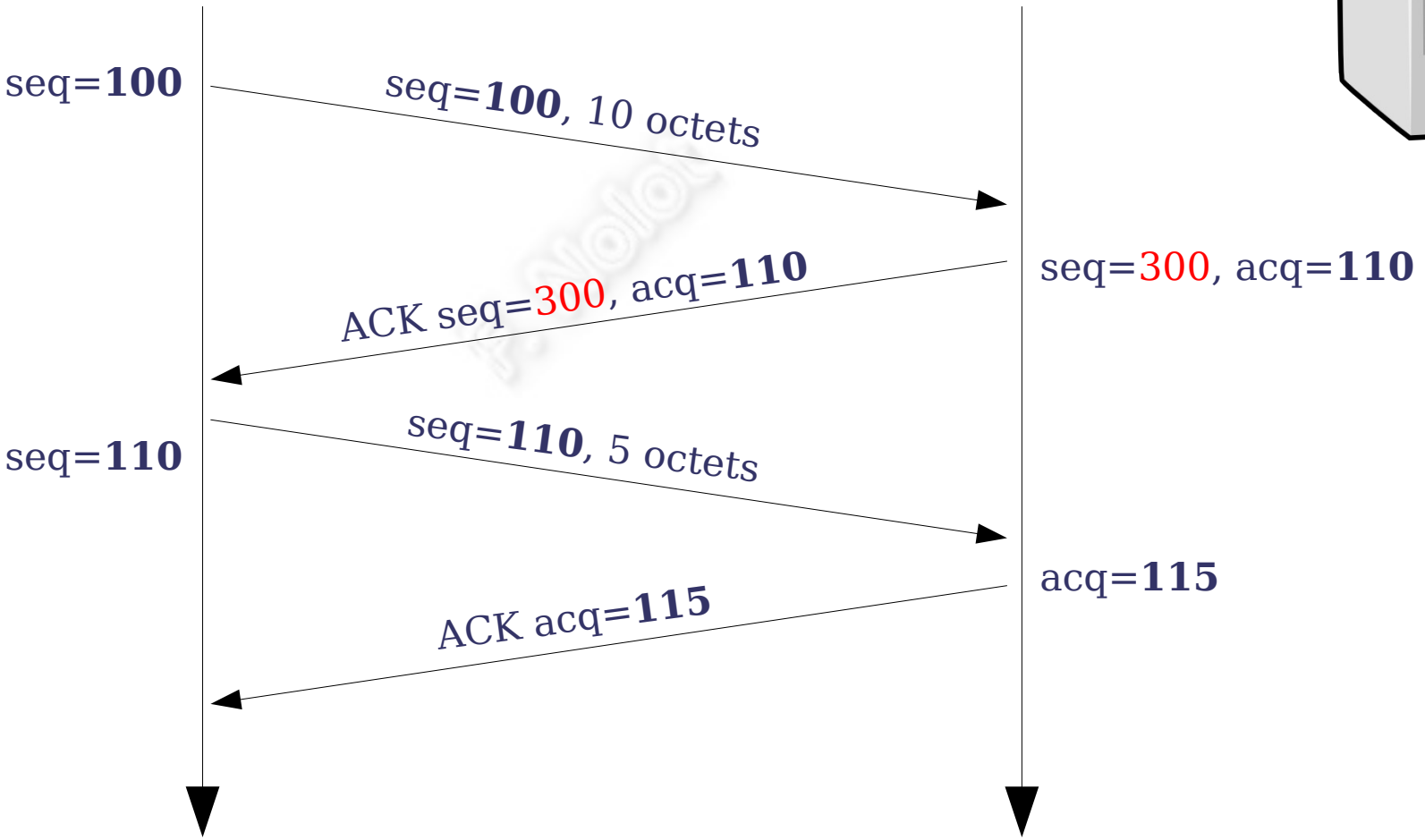
- Chaque segment TCP est identifié par un numéro de séquence
- Cette valeur est incrémentée de la taille des données envoyées (en octets)
- Pour chaque segment réceptionné, le récepteur expédie un segment ACK avec, comme valeur d'acquittement, la quantité d'octet correctement reçu +1
  - Cela indique quel octet il attend

Port Source (16 bits)	Port Destination (16 bits)
Numéro de Séquence (32 bits)	
Numéro d'acquittement (32 bits)	
Long. En-tête (4), Réservé (6), Flags (6)	Taille fenêtre (16 bits)
Checksum (16 bits)	Urgent (16 bits)
Options (0 à 32 bits si présent)	
Données de la couche applicative	

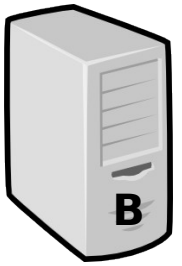
Exemple (1/3)


A


B


seq=100
seq=100, 10 octets
seq=300, acq=110
ACK seq=300, acq=110
seq=110
seq=110, 5 octets
acq=115
ACK acq=115

Exemple (2/3)



seq=**100**

seq=***100***, *10 octets*

seq=300, acq=**110**

ACK seq=300, acq=**110**, 0 octet

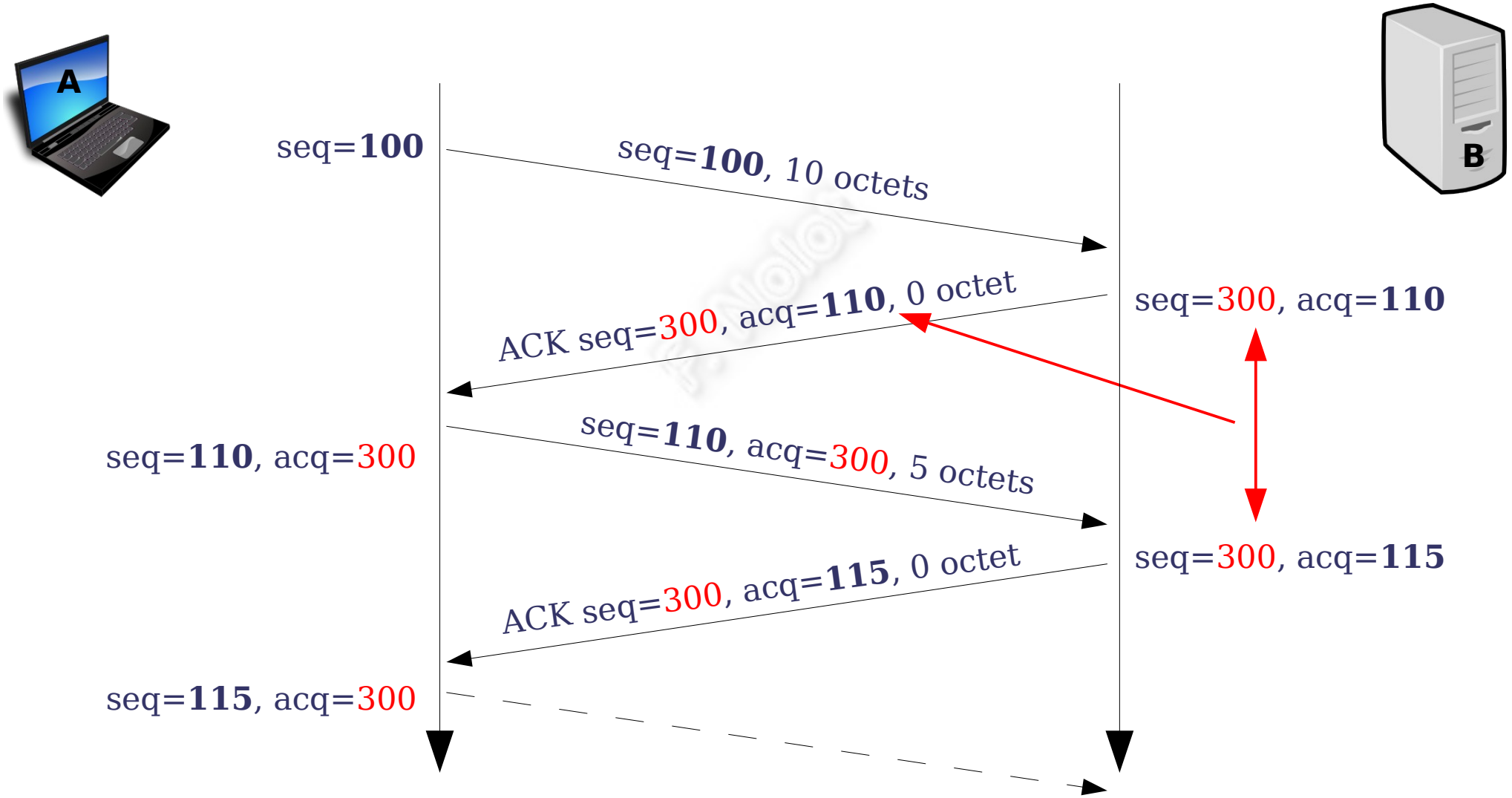
seq=**110**, acq=300

seq=***110***, acq=*300*, 5 octets

seq=300, acq=**115**

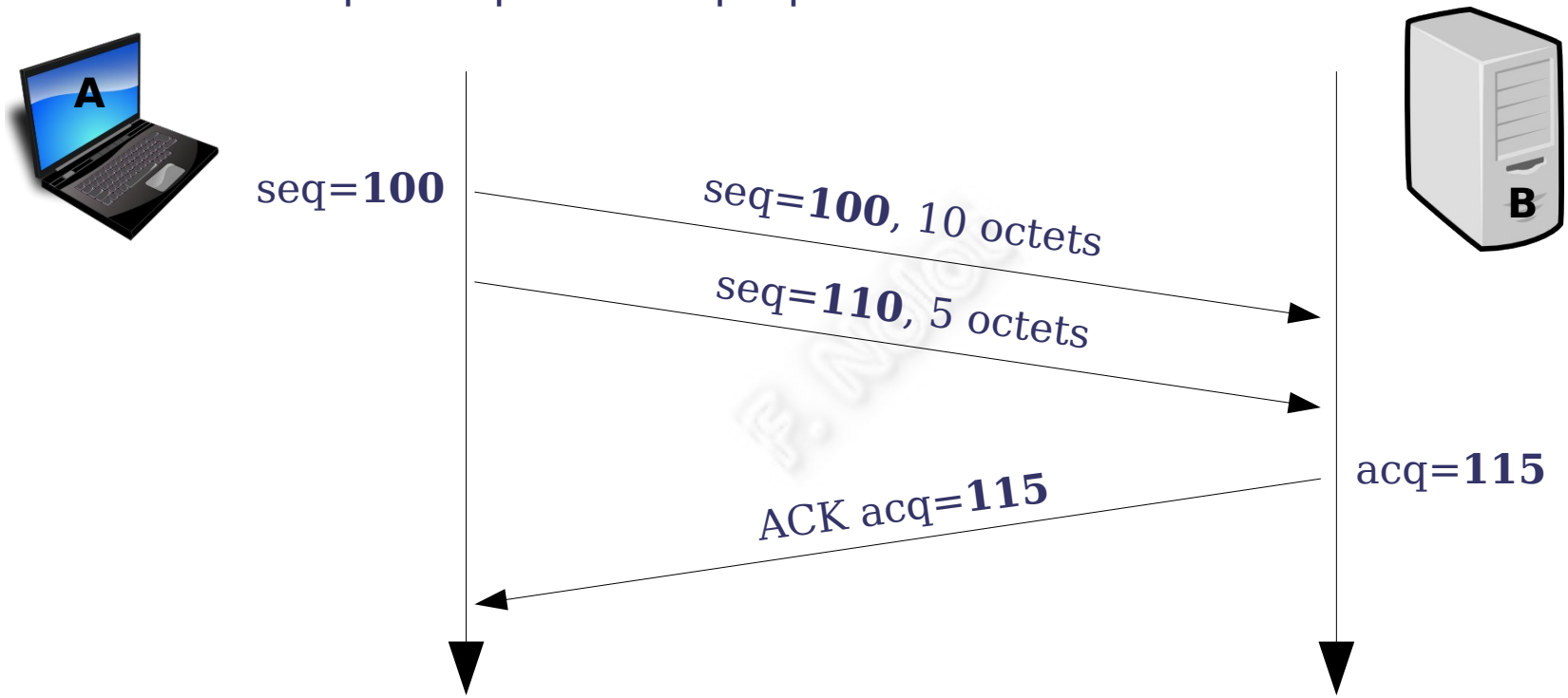
ACK seq=300, acq=**115**, 0 octet

Exemple (3/3)


A
B
seq=100
seq=100, 10 octets
seq=300, acq=110
ACK seq=300, acq=110, 0 octet
seq=110, acq=300
seq=110, acq=300, 5 octets
seq=300, acq=115
ACK seq=300, acq=115, 0 octet
seq=115, acq=300

Acquittement de multiples segments

- Possibilité d'acquitter plusieurs paquets en une fois



- Combien de segments peut-on acquitter en une seule fois ?
  - Quelle quantité de données peut-on envoyer avant d'attendre un acquittement ?

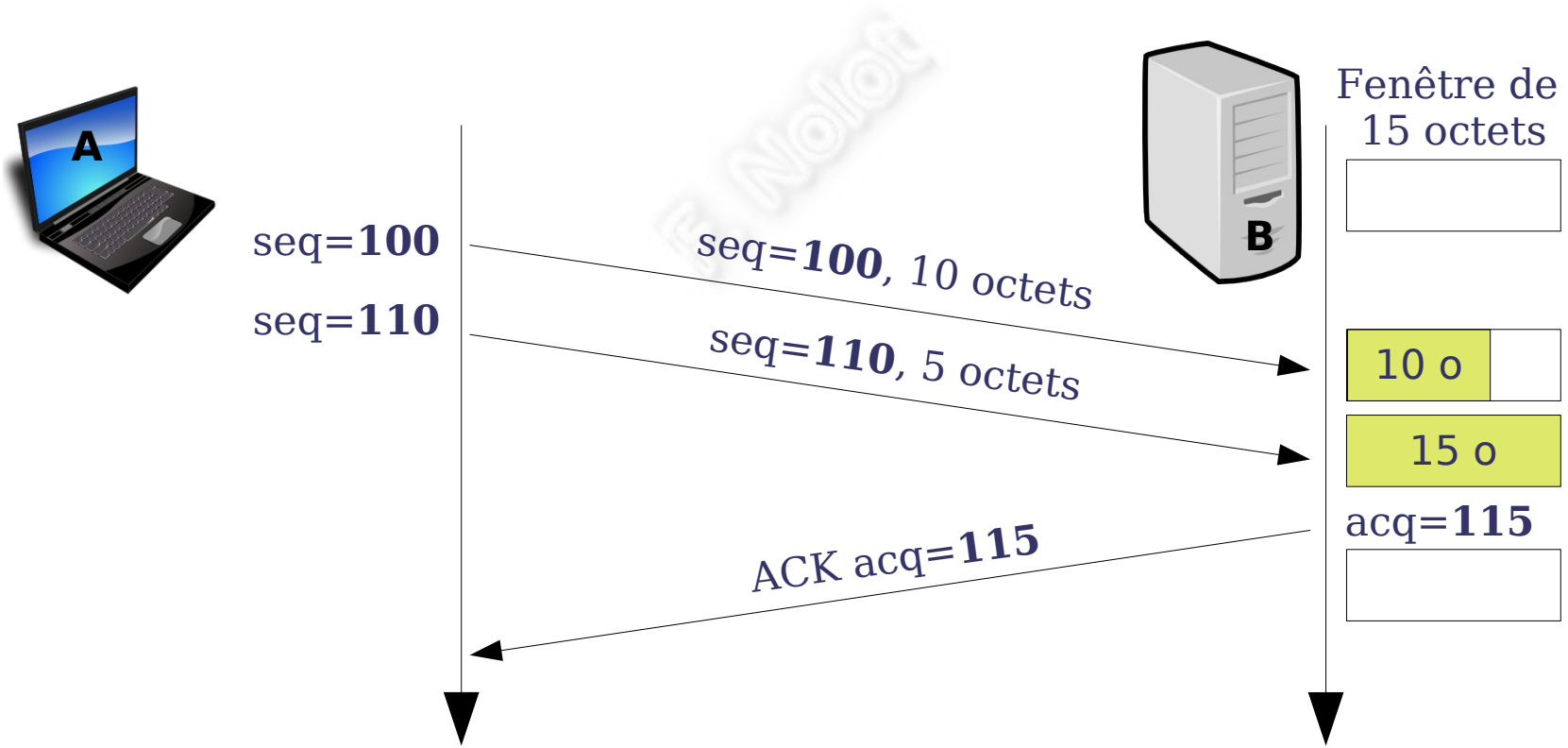
La fenêtre TCP ?

- L'expéditeur et le destinataire vont stocker une certaine quantité de données avant d'acquitter
  - Optimisation du nombre de messages échangés
- Cette quantité est appelée la taille de la fenêtre

Port Source (16 bits)	Port Destination (16 bits)
Numéro de Séquence (32 bits)	
Numéro d'acquittement (32 bits)	
Long. En-tête (4), Réservé (6), Flags (6)	**Taille fenêtre (16 bits)**
Checksum (16 bits)	Urgent (16 bits)
Options (0 à 32 bits si présent)	
Données de la couche applicative	

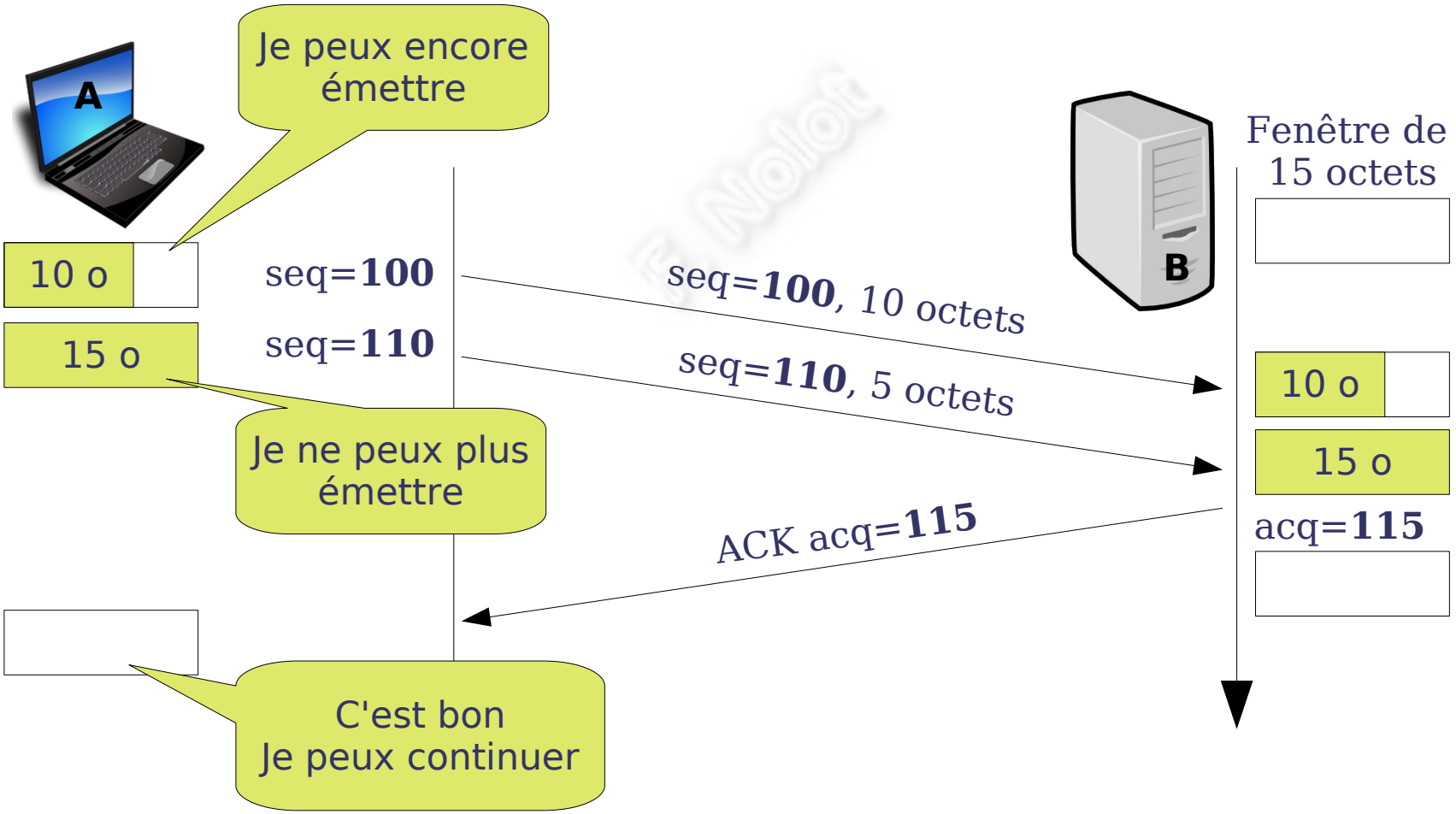
Fonctionnement de la fenêtre glissante (1/3)

- Hypothèse : fenêtre de 15 octets sur la destination
- Expéditeur va envoyer 15 octets puis attendre leur acquittement



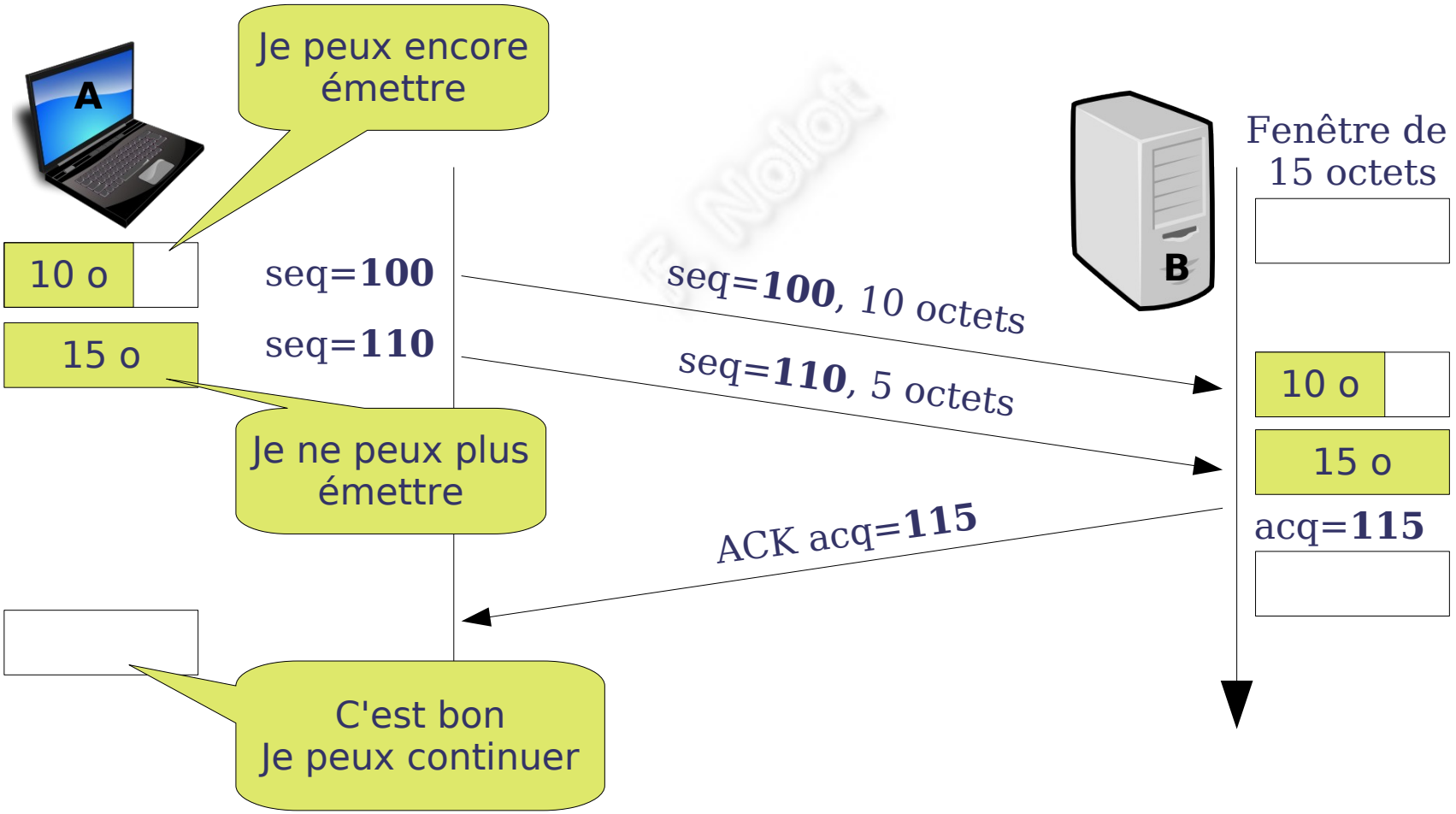
Fonctionnement de la fenêtre glissante (2/3)

- L'expéditeur et le destinataire ont connaissance des tailles de fenêtre de chacun
- L'expéditeur sait donc quand il peut arrêter d'émettre



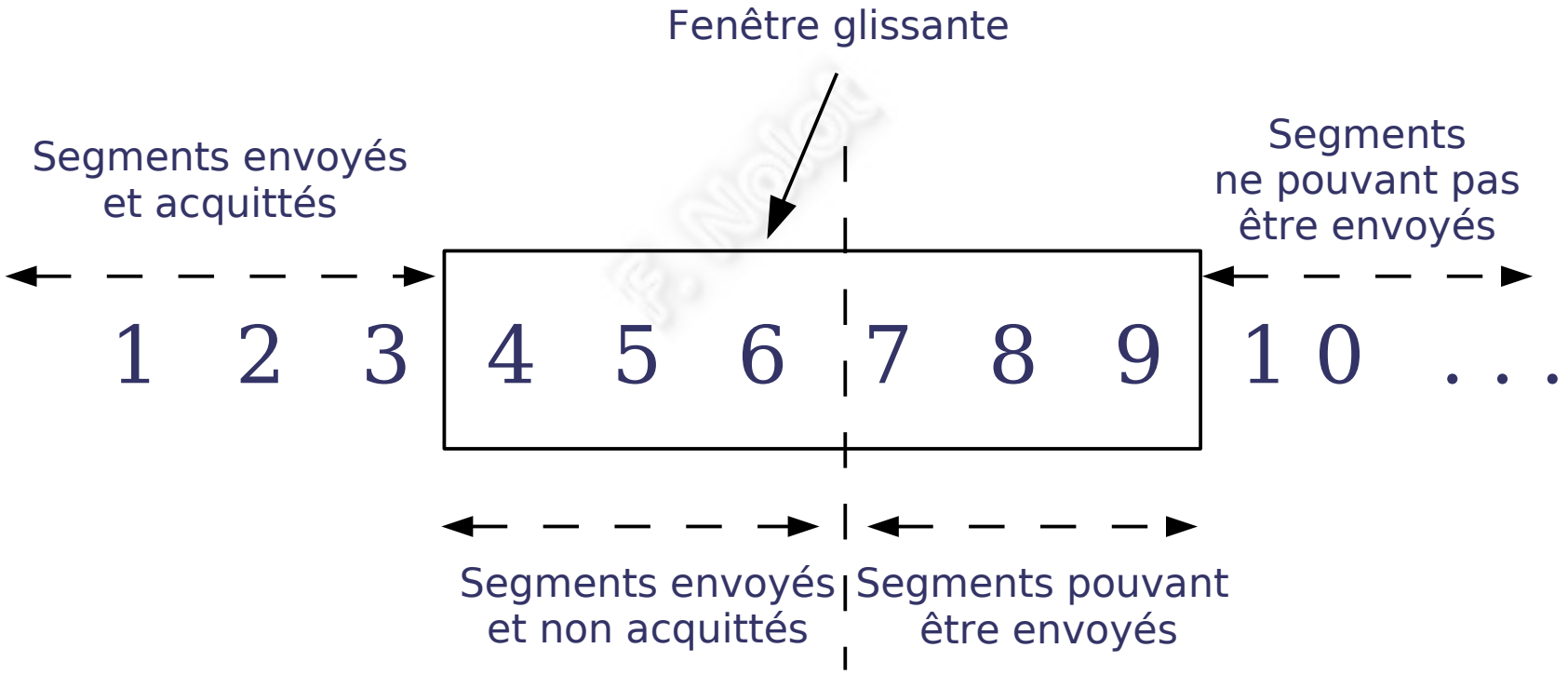
Fonctionnement de la fenêtre glissante (3/3)

- Permet au destinataire de prévenir l'expéditeur de la quantité d'octet qu'il peut envoyer sans attente systématique de l'acquittement de chaque segment



Pourquoi fenêtre glissante ?

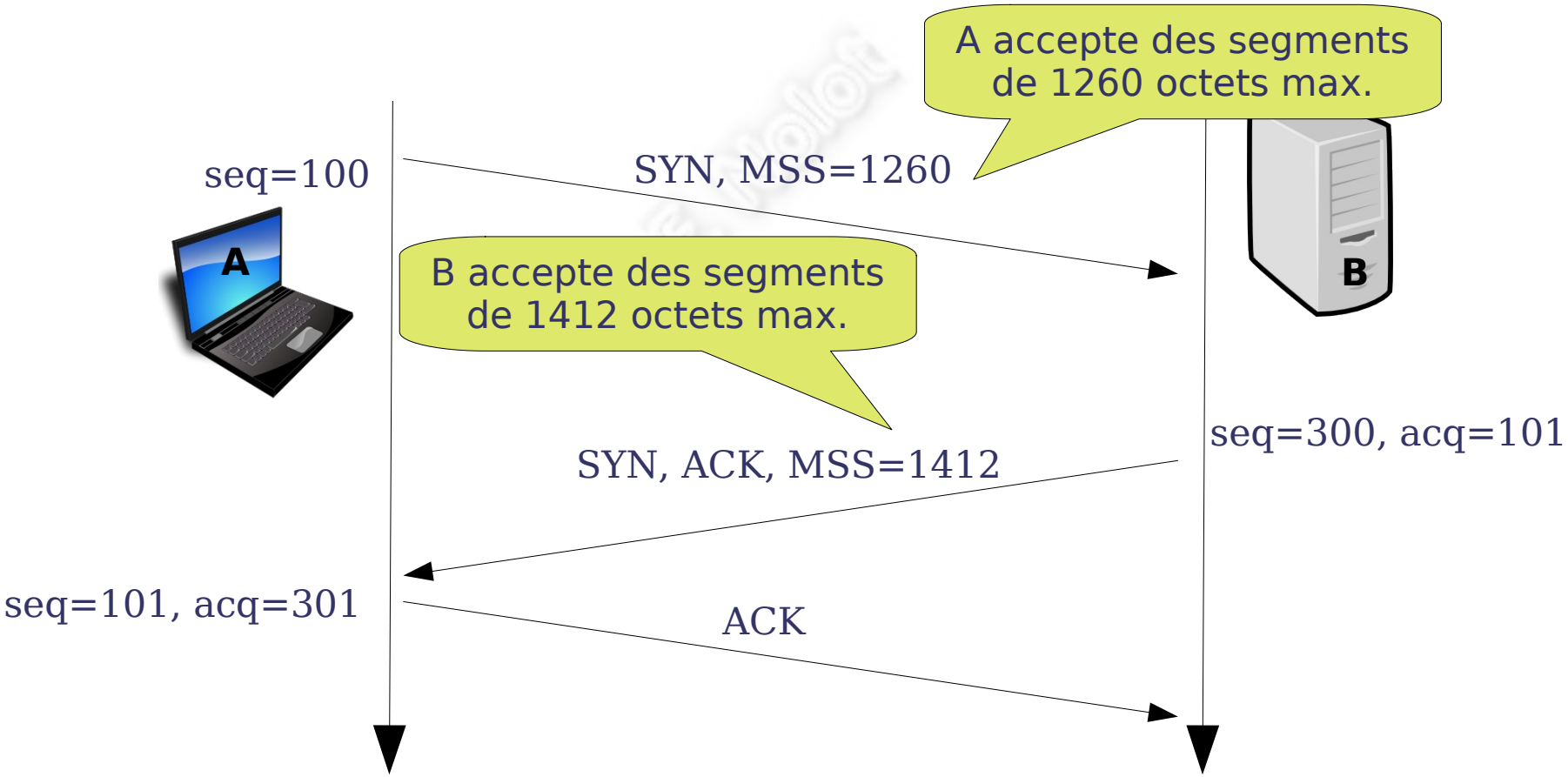
- Au fur et à mesure des acquittements, la fenêtre se décale sur la droite



- Comment et pourquoi a-t-on des tailles de fenêtres et de segment différents ?
- Pourquoi ne peut-on pas envoyer en un seul segment toutes les données ?

L'option MSS ?

- Permet de spécifier la taille maximale d'un segment qu'un équipement accepte
  - Dépend de plusieurs paramètres du système et du réseau (MTU par exemple)
- L'option MSS n'est présente QUE dans les paquets SYN

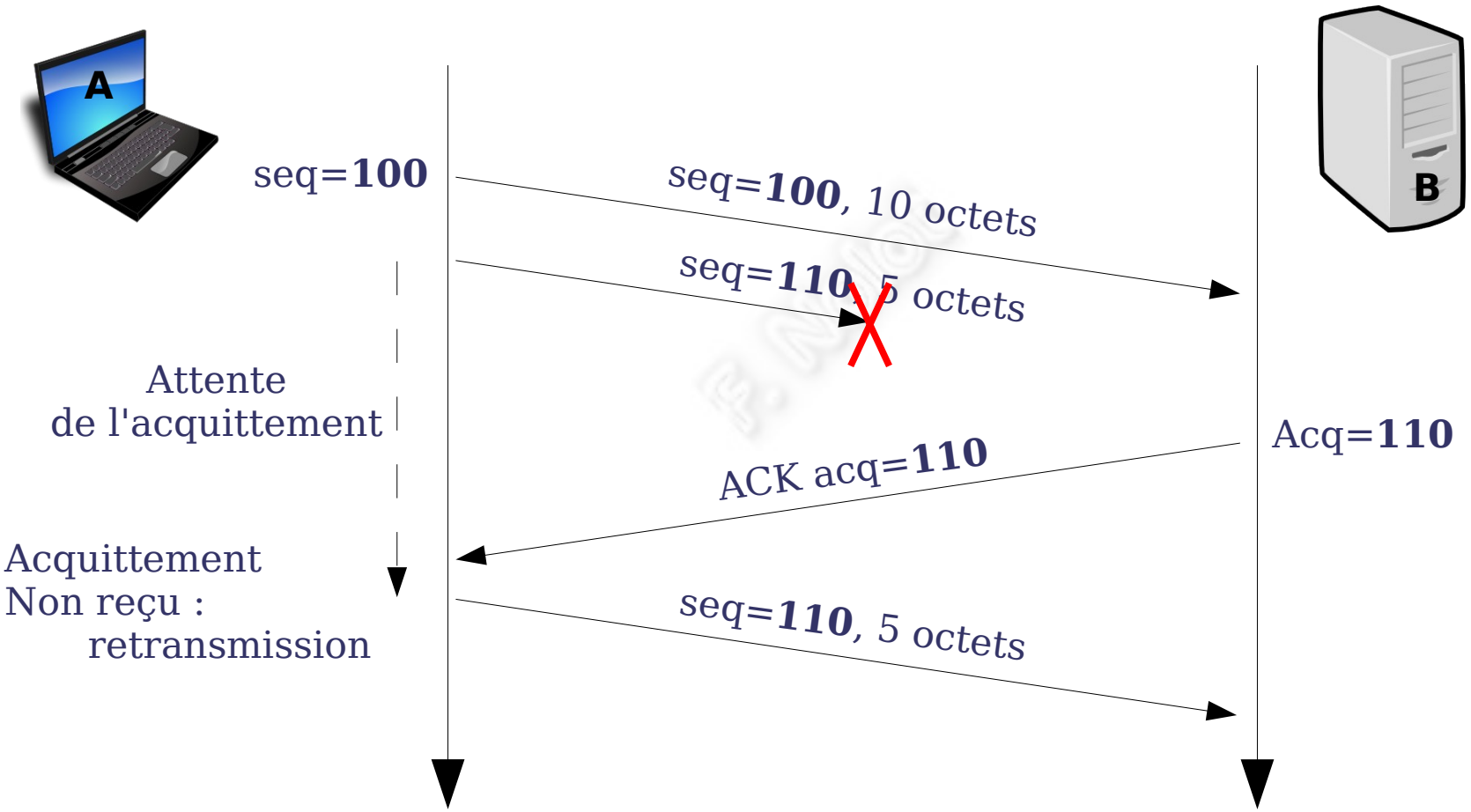


Fenêtre et MSS ?

- La taille de la fenêtre est généralement plus importante que la MSS
  - Fenêtre de taille de 8192 octets et des MSS de 1260 octets
  - Obligation d'expédier dans ce cas plusieurs segments pour pouvoir remplir la fenêtre

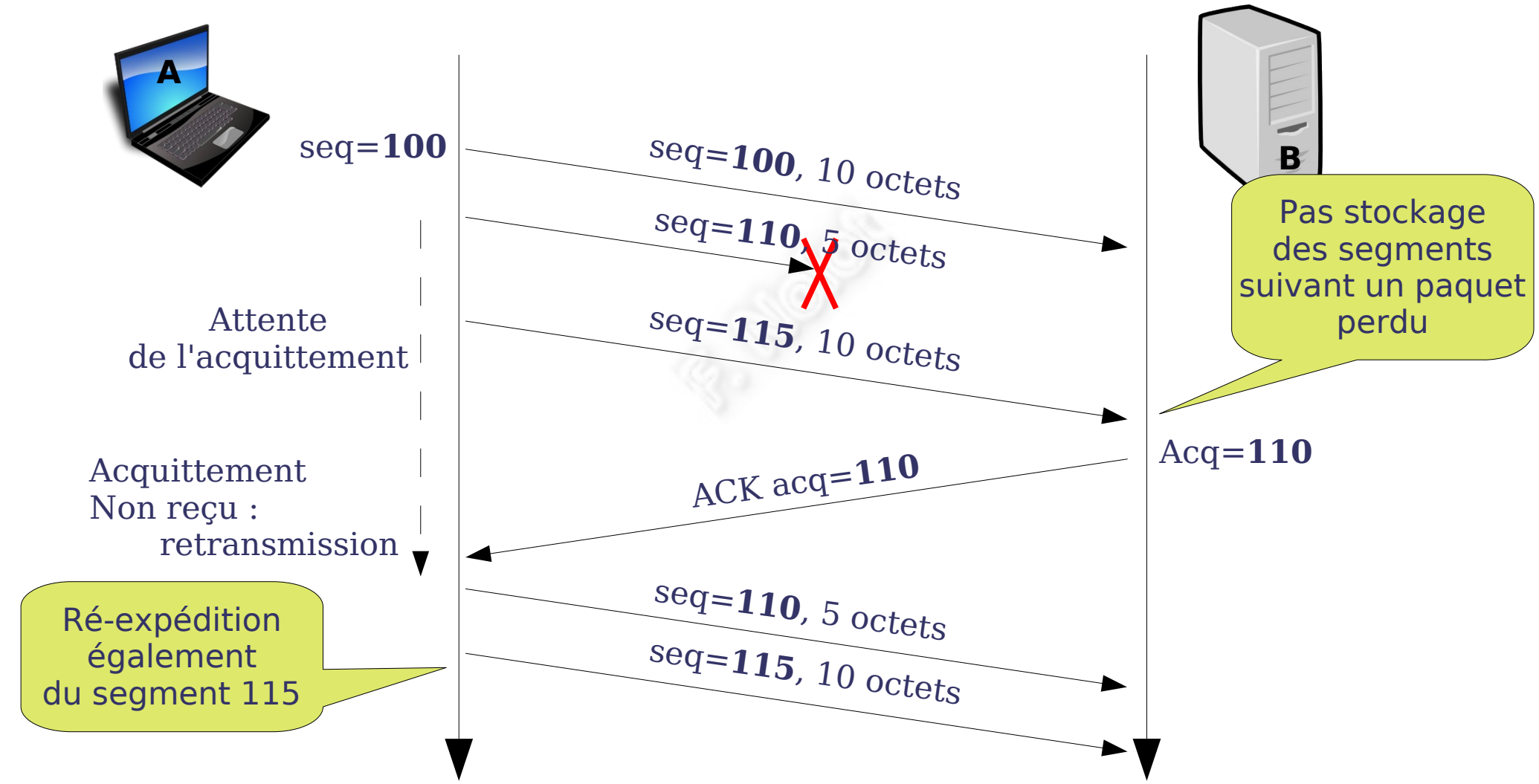
- TCP : acheminement fiable ?
  - Si un segment est perdu ?

Perte de segments ? (cas 1/3)

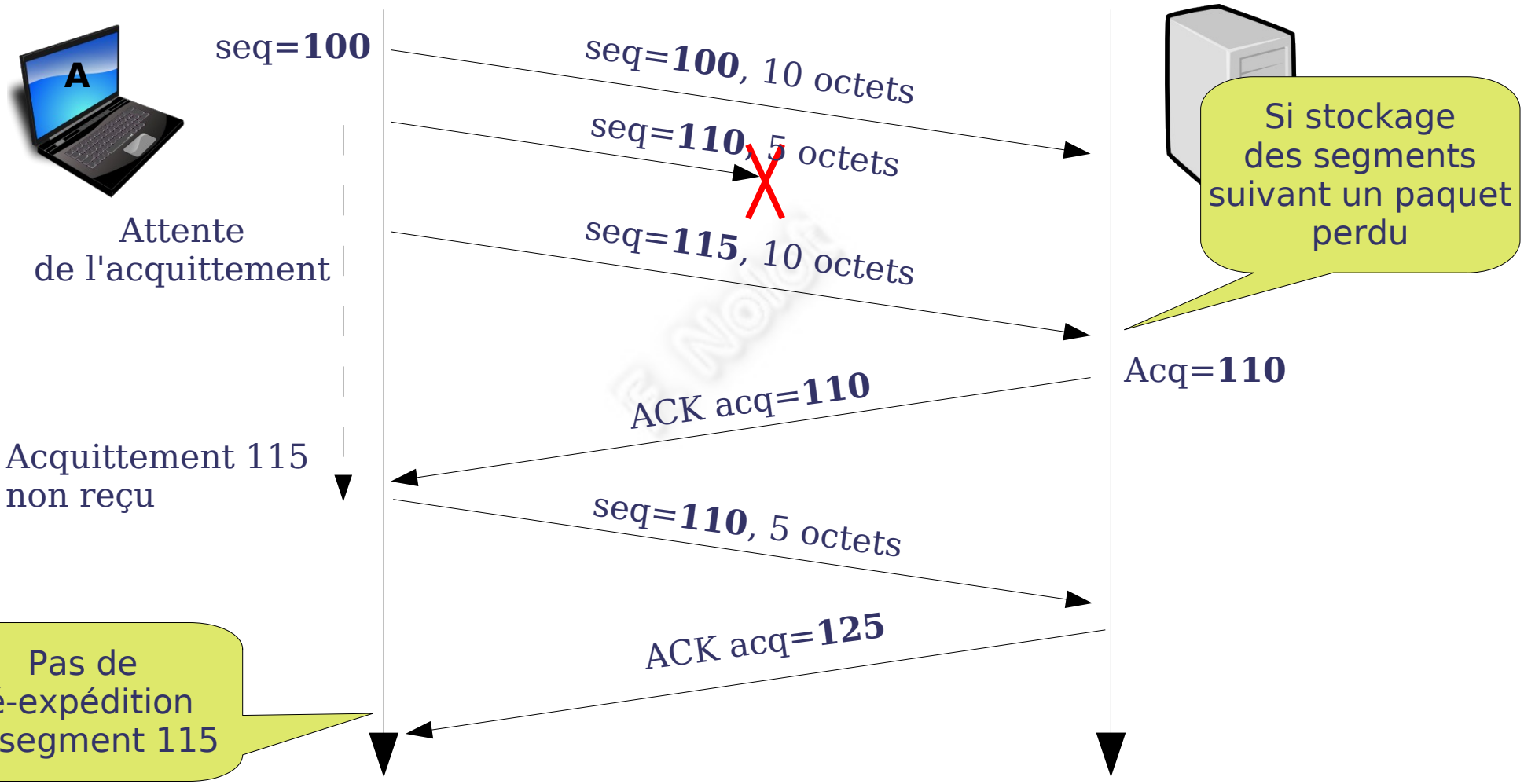


- Acquittement du dernier paquet reçu, sans aucune perte intermédiaire

Perte de segments ? (cas 2/3)



Perte de segments ? (cas 3/3)



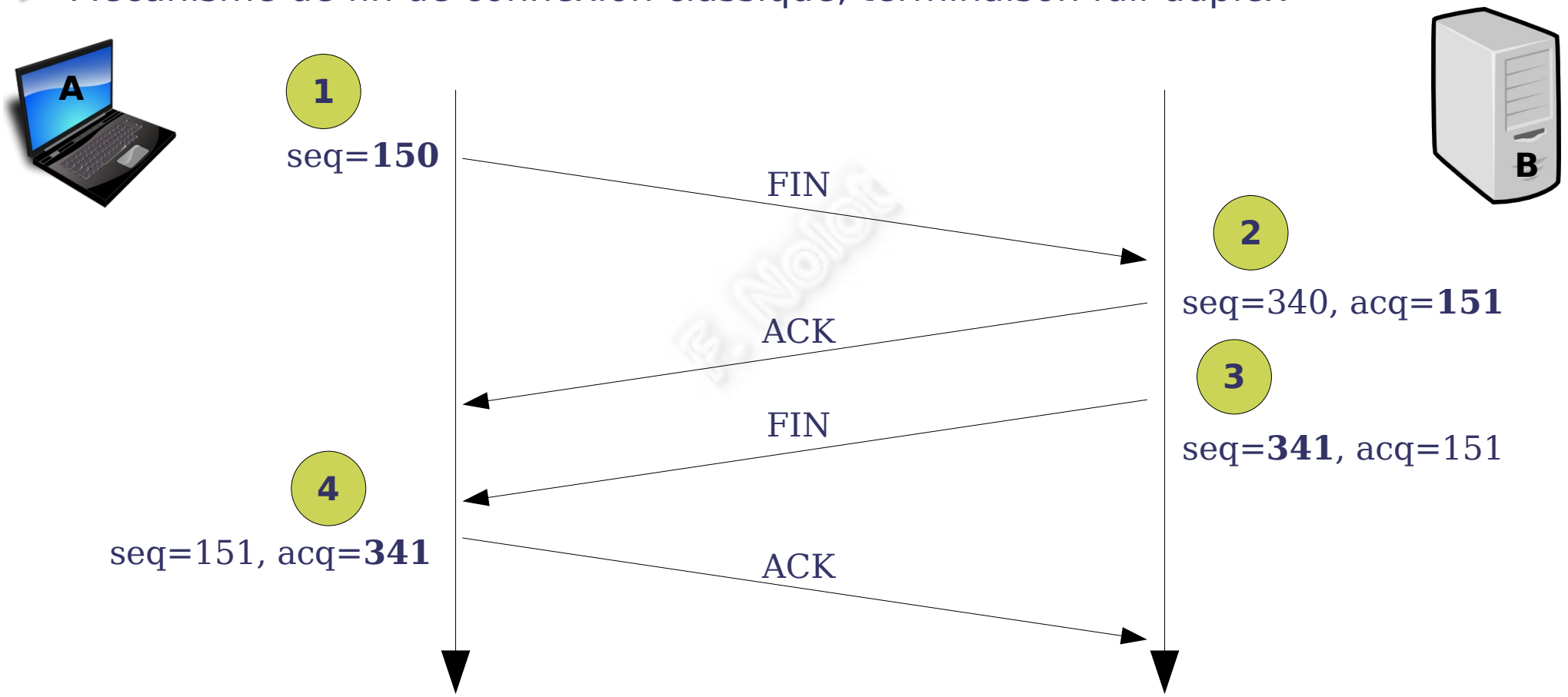
Le bit URG ?

- 2 champs sont utilisés pour les segments urgent
  - Le bit URG dans les flags
  - Le champs Urgent qui donne un pointeur vers le dernier octet urgent dans le segment
- Urgence ?
  - Quand l'utilisateur, en interactif, veut interrompre un échange (appui sur CTRL+C) par exemple

Port Source (16 bits)	Port Destination (16 bits)
Numéro de Séquence (32 bits)	
Numéro d'acquittement (32 bits)	
Long. En-tête (4), Réservé (6), Flags (6)	**Taille fenêtre (16 bits)**
Checksum (16 bits)	Urgent (16 bits)
Options (0 à 32 bits si présent)	
Données de la couche applicative	

Fin d'une connexion

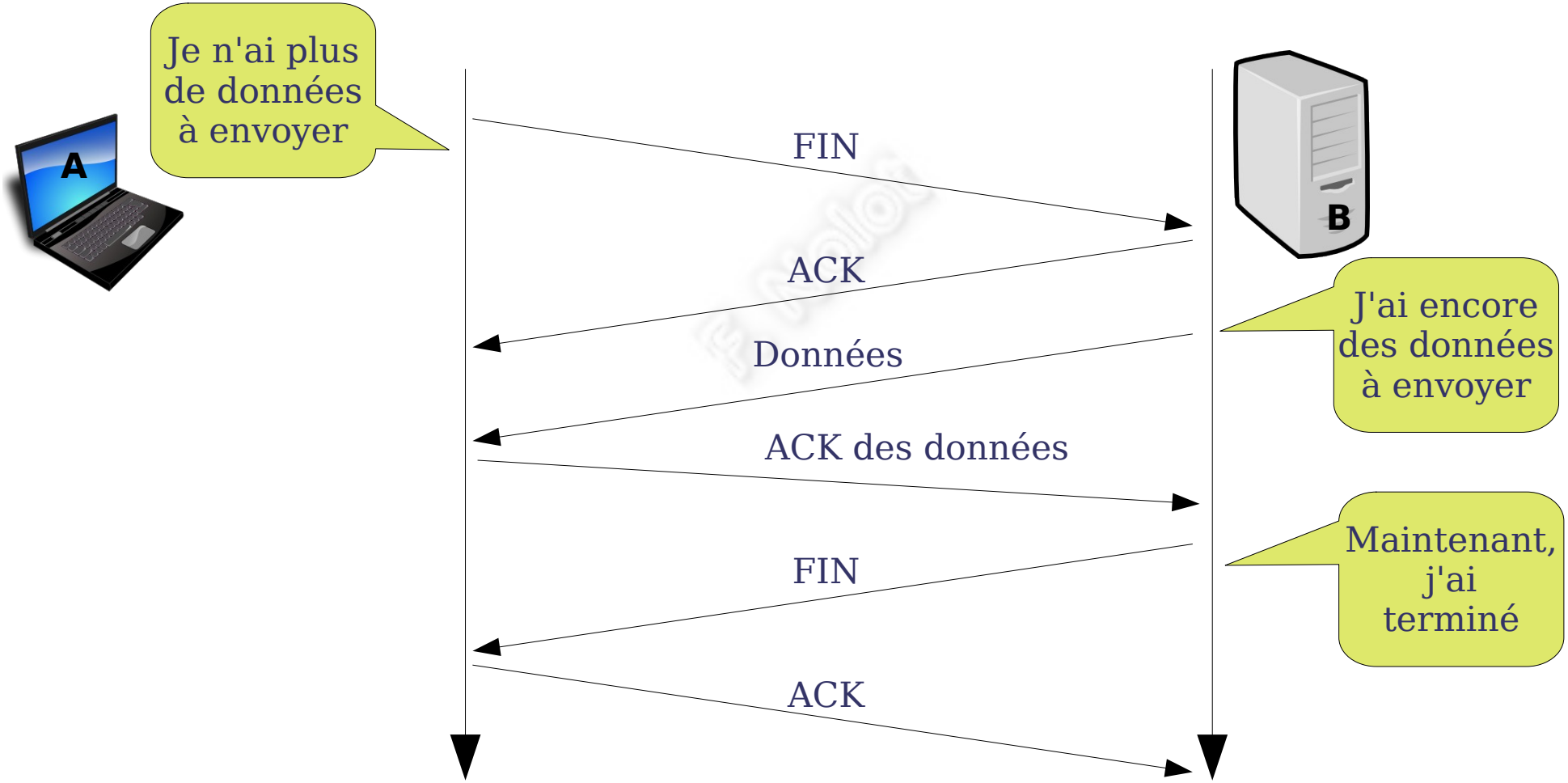
- Mécanisme de fin de connexion classique, terminaison full-duplex



Les segments FIN « consomme » 1 numéro de séquence

Fin d'une connexion half-close

- Terminaison half-close



Gestion de la congestion ?

- TCP est en mesure de gérer la congestion
  - En changeant les tailles des fenêtres afin de diminuer les envoies successif de segments
  - Utilise une autre fenêtre : la congestion window

Congestion windows (la cwnd)

- Initialisé à 1 au début de la connexion
- Envoie d'un segment
- A la réception du ACK, augmentation de la cwnd de 1 (cwnd = 2)
- Envoie de 2 segments
- A la réception des 2 ACK, augmentation de la cwnd de 2 (cwnd = 4)
- Envoie de 4 segments
- ...

- Augmentation exponentielle de la cwnd
- **Attention** : toujours respect de la fenêtre glissante. Les 2 techniques se cumulent. Nous n'allons pas transmettre 4 segments si la fenêtre est pleine !
  - La cwnd est contrôlé par l'expéditeur alors que la fenêtre glissante est contrôlée par le destinataire

La couche transport du modèle OSI

Le protocole UDP : User Datagram Protocol

Présentation

- Protocole en mode non connecté
- Aucune fiabilité
- Identification des ports sources et destination des applications qui échangent des données
- Longueur du datagramme complet (En-tête + data)
- Checksum du datagramme complet (checksum est optionnel, contrairement au checksum en TCP)

Port Source (16 bits)	Port Destination (16 bits)
Long. En-tête (16 bits)	Checksum (16 bits)
Données de la couche applicative	

Usage

- Les applications utilisant UDP :
  - DNS
  - DHCP
  - Vidéo sur IP
  - Téléphonie IP
- Application qui ont besoin
  - Rapidité
  - Mode fiable non nécessaire
    - Retransmission en Téléphonie IP est inutile, le paquet est perdu et c'est trop tard
- Les applications doivent se charger
  - des congestions
  - Du ré-ordonnancement des paquets

La couche transport du modèle OSI

Merci pour votre attention