Ecole Centrale de Lyon – 3A – MOD 5.7
René CHALON

Réseaux informatiques
~
Couche Transport - TCP

1- Couche Transport

2- Protocole TCP

3- Protocole UDP

4- Autres protocoles de transport

Version du 9/10/2012

1 - Couche Transport

- Fonctions de la couche transport:
  - communication entre applications
  - transporte des messages (TPDU) de manière transparente:
    - segmentation en paquets à l'émission
    - réassemblage à la réception
  - fournit une qualité de service donnée:
    - négociée à la connexion; si elle ne peut être assurée alors la connexion est refusée !
    - Exemple: détection et correction des erreurs
  - contrôle de bout en bout (les équipements intermédiaires ne participent pas au service transport et aux couches supérieures)
  - optimisation des ressources du réseau:
    - choix du "meilleur" service réseau possible
- Exemples: TCP, UDP, TP/ISO (5 classes de 0 à 4), SCTP, etc.

7	*Application*
6	*Présentation*
5	*Session*
4	**Transport**
3	*Réseau*
2	*Liaison de données*
1	*Physique*

Couche « Transport » : TCP, UDP

- 2 protocole principaux :

	TCP *Transmission Control Protocol*	**UDP** *User Datagram Protocol*
Type de connexion	Mode connecté	Mode non-connecté
Circuit virtuel	Oui (remise dans l'ordre des donnés garantie)	Non (messages pouvant arriver en désordre)
Nature du flux	Flot d'octets (découpé en paquets IP réassemblés à l'arrivée)	Messages (placés dans des paquets IP indépendants)
Fiabilité du transport	Fiable (accusés de réception par le destinataire et retrans--mission en cas de perte)	Non-fiable (en pratique la fiabilité d'IP)
Multiplexage (notion de ports) : partage de l'interface réseau par plusieurs applications sans que les données se mélangent	Oui	Oui
Contrôle de flux : régulation de la vitesse de transmission et gestion des congestions	Oui	Non

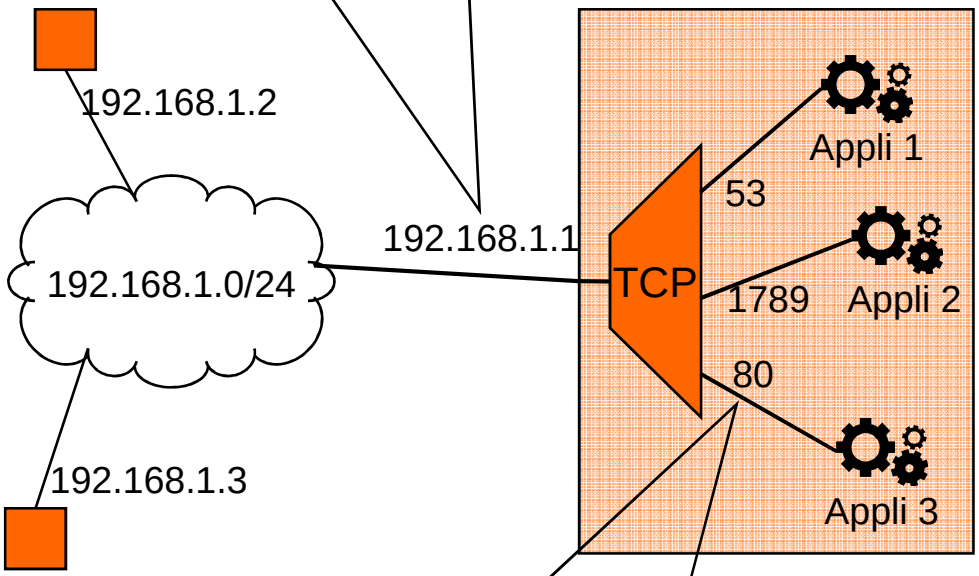
2

2 - Protocole TCP

- TCP [Transmission Control Protocol] fournit un service de transport fiable de bout en bout entre 2 applications
- Caractéristiques:
  - **mode connecté**
  - établissement d'un **circuit virtuel bidirectionnel**
  - transport fiable: correction d'erreur, re-séquencement, …
  - transferts tamponnés: grouper les petits messages
  - connexions non structurées : transport de flots d'octets
  - contrôle de flux
  - multiplexage : plusieurs connexions simultanées
  - possibilité de données prioritaires (données urgentes + "push")
- Taille des messages non-limitée:
  - ➔ les messages sont décomposés en **segments** de 64 Ko max pour être passés à la couche IP
- Défini par la RFC 793 (STD 7)

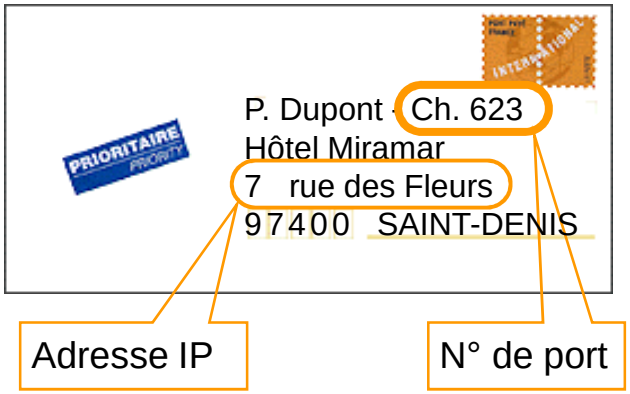
Notion de port

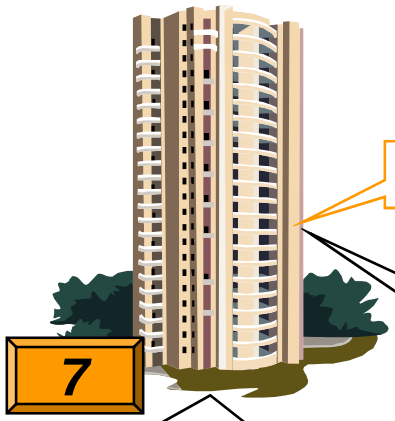
L'adresse IP permet d'identifier une **machine** particulière dans un **réseau**



Le numéro de port permet d'identifier une **application** particulière dans une **machine**

Métaphore postale





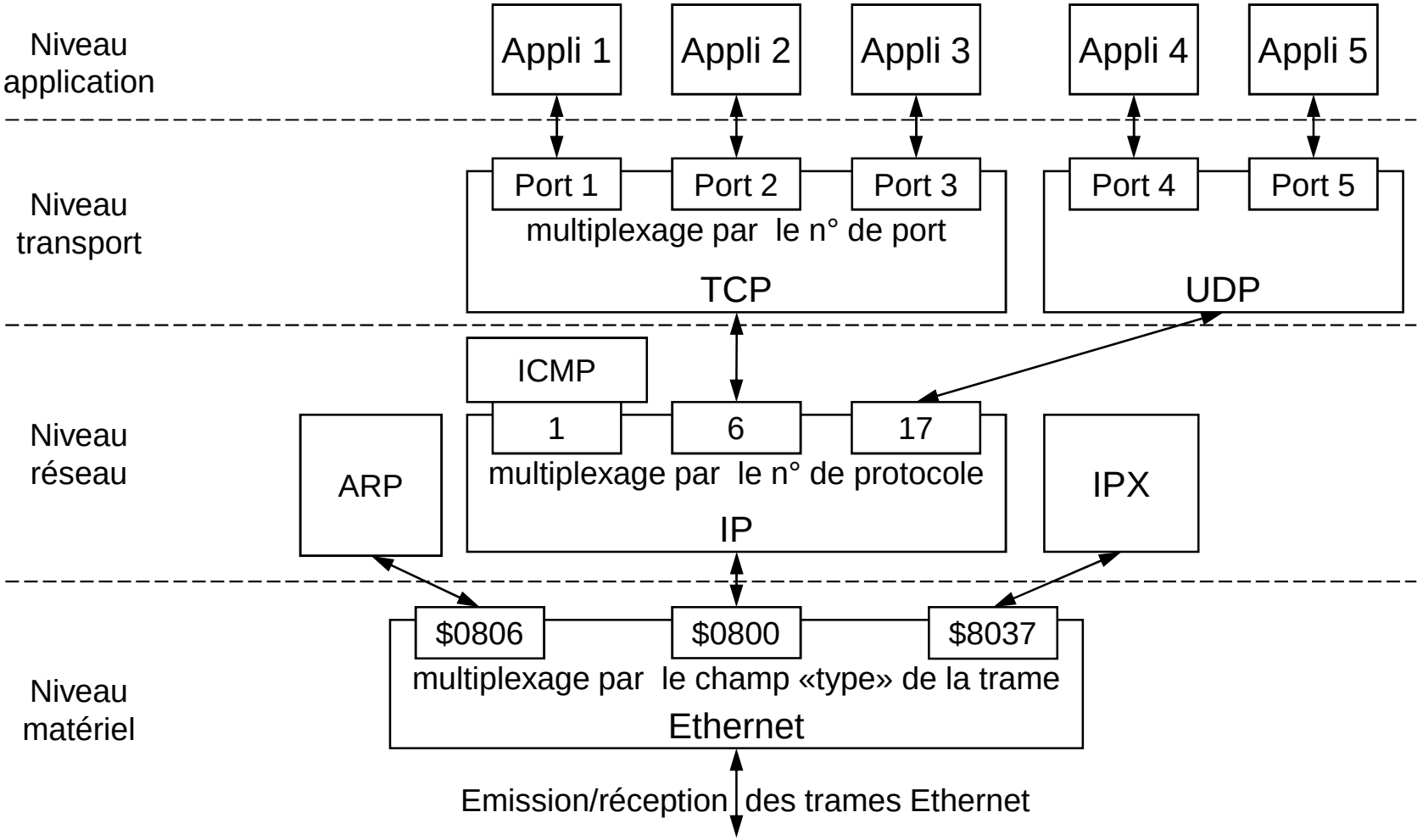
Ch. 623

Le concierge distribue le courrier dans l'hôtel

Le postier distribue le courrier au *7 rue des Fleurs*

Ports

- Assimilable à des adresses de couche 4



Utilisation des ports

- Affectation des ports aux applications
  - Chaque application peut utiliser tout port **libre** sur une machine

- Port réservé ou *Well-Known Port*
  - Pour les principaux services, un n° de port est réservé
  - Les serveurs "écoutent" sur ce port les demandes de connexion en provenance des clients

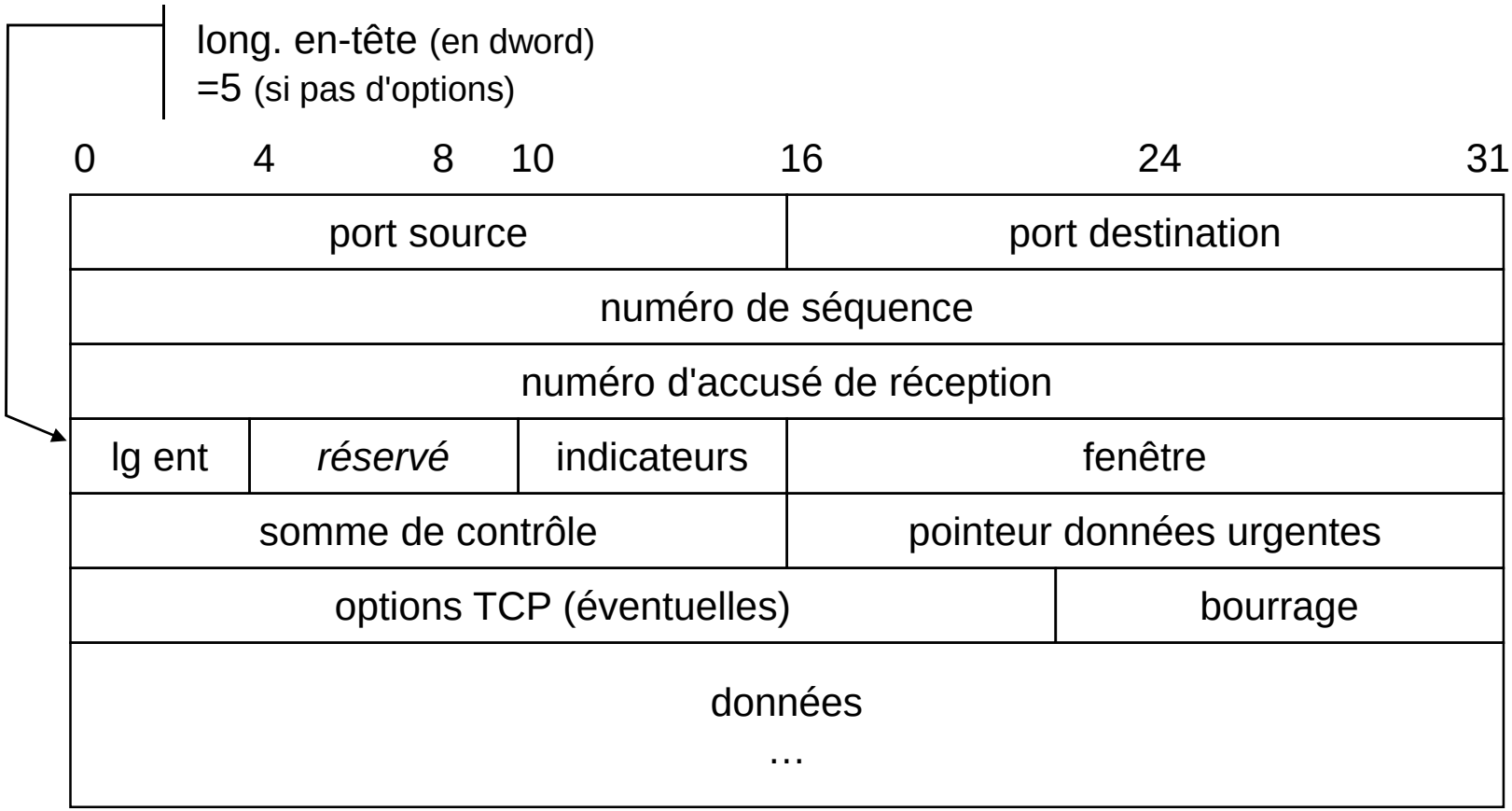
Quelques exemples de ports réservés

Protocole (service)	N° port réservé	Protocole transport
FTP données	20	TCP
FTP contrôle	21	TCP
E-mail SMTP	25	TCP
E-mail POP	110	TCP
DNS	53	UDP (TCP)
HTTP (Web)	80	TCP
NTP (horloge)	123	UDP

- Identification d'une connexion TCP :
  - Quadruplet: <adr IP source, n° port source, adr IP dest., n° port dest>
  - ➔ possibilité de partager le même port par plusieurs connexions !

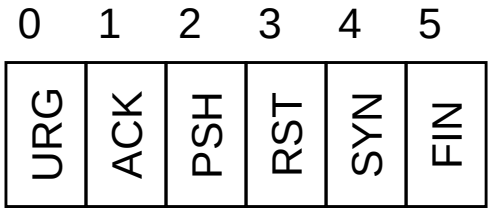
Segment TCP (1/3)

- Format du segment:



Segment TCP (2/3)

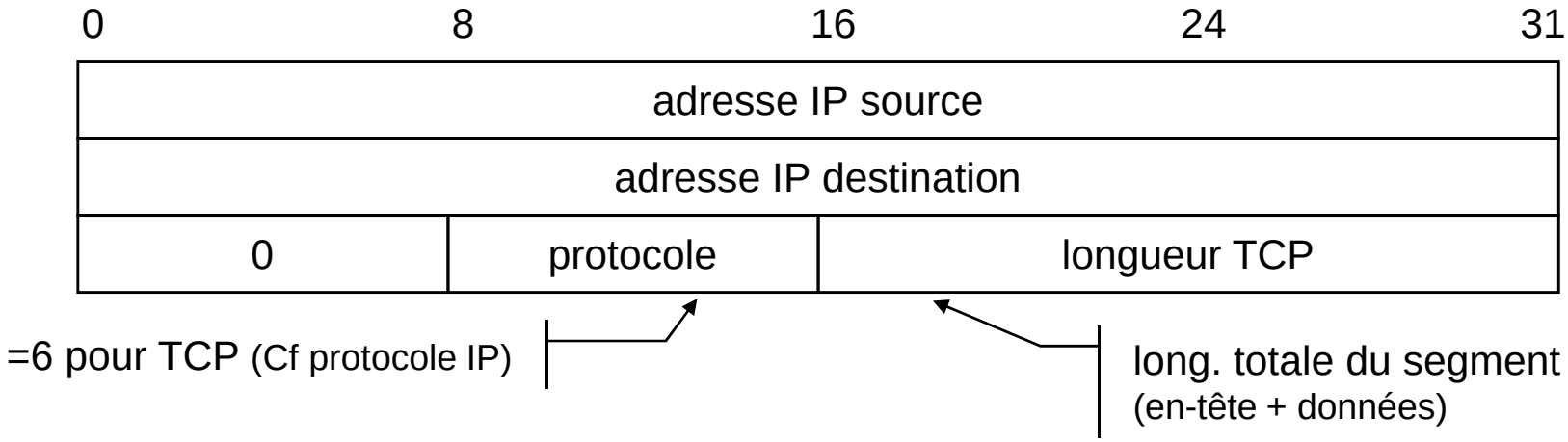
- Champ indicateurs:



  - URG=1 : données urgente
  - ACK=1 : le champs accusé de réception est valide
  - PSH=1 : "push" requit = envoyer immédiatement les données
  - RST=1 : ré-initialiser la connexion
  - SYN=1 : synchroniser les n° de séquence
  - FIN=1 : l'émetteur a atteint la fin de son flot de données
- En pratique:
  - SYN=1, ACK=0 : demande de connexion
  - SYN=1, ACK=1 : connexion acceptée

Segment TCP (3/3)

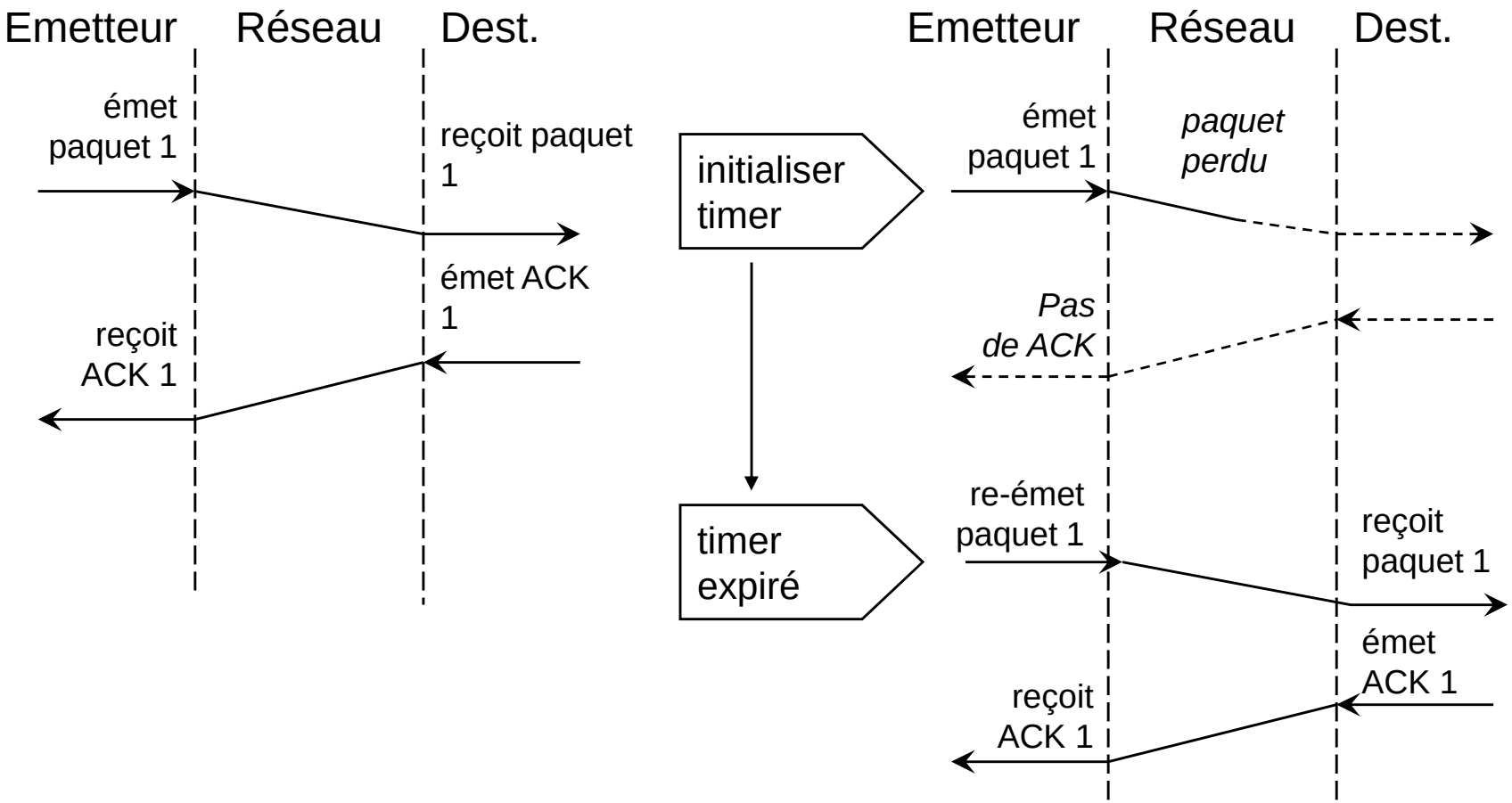
- Calcul de la somme de contrôle:
  - Avant le calcul un pseudo-en-tête est ajouté:



  - + un octet à 0 est évent^{ent} ajouté pour obtenir un multiple de 16 bits
  - La somme de contrôle est calculée sur l'ensemble des mots (arithmétique en complément à 1) puis son complément à 1 est rangé dans le champ "somme de contrôle"
  - le pseudo-en-tête et l'octet de bourrage ne sont pas transmis !

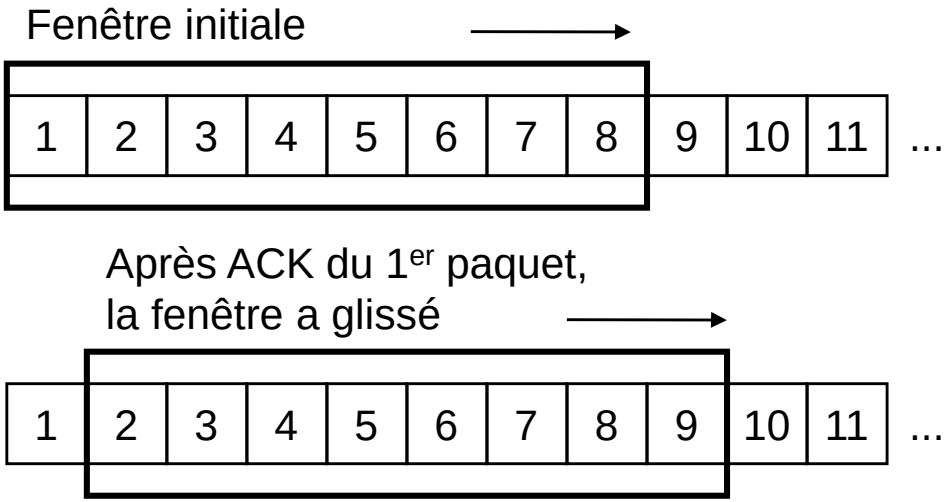
Fiabilité du service de transport

- La fiabilité est assurée par des accusés de réception avec retransmission en cas d'erreur:



Fenêtre glissante (1/2)

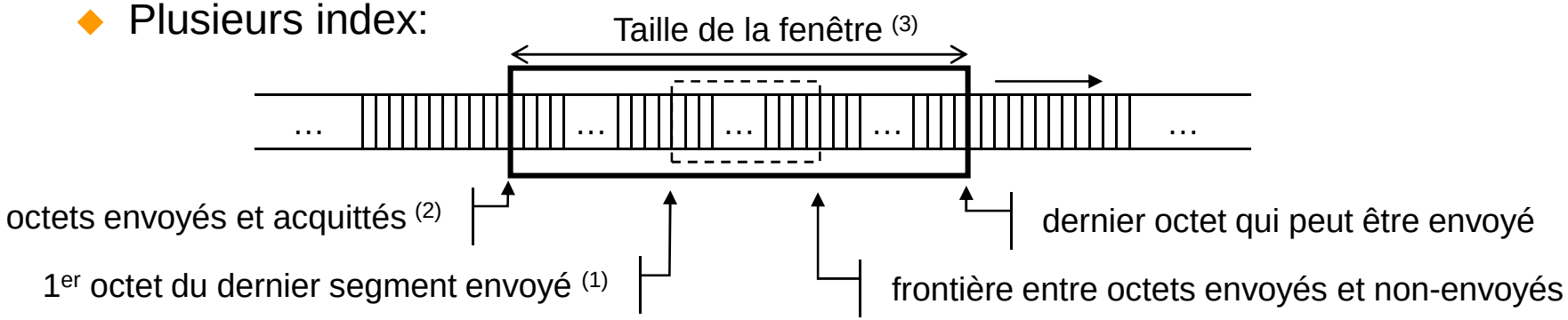
- Problème d'efficacité de l'acquittement paquet par paquet:
  - il faut attendre l'accusé de réception du paquet précédent pour pouvoir transmettre le suivant ==> perte de temps
- Solution: fenêtre glissante [Sliding window]
  - émettre plusieurs paquets d'avance avant d'avoir à attendre un accusé de réception
  - *exemple*: fenêtre de huit paquets



NOTE:
Il faut gérer une mémoire tampon à chaque extrémité de la connexion !

Fenêtre glissante (2/2)

- TCP utilise une fenêtre glissante au niveau de l'octet:
  - les octets sont numérotés séquentiellement (sur 4 octets)
  - Plusieurs index:



- Champs utilisés dans l'en-tête:
  - Sens « aller » :
    - Champ « N° de séquence »: donne le numéro du 1er octet transmis dans le segment = (1)
  - Sens « retour » (acquittements) :
    - Champ « N° d'accusé de réception »: donne le numéro du prochain octet attendu, donc tous les octets précédents sont acquittés = (2) +1
    - Champ « Fenêtre »: taille de la fenêtre de réception =(3)
- TCP est bidirectionnel ==> 2 fenêtres dans **chaque** sens

Contrôle de flux (1/3)

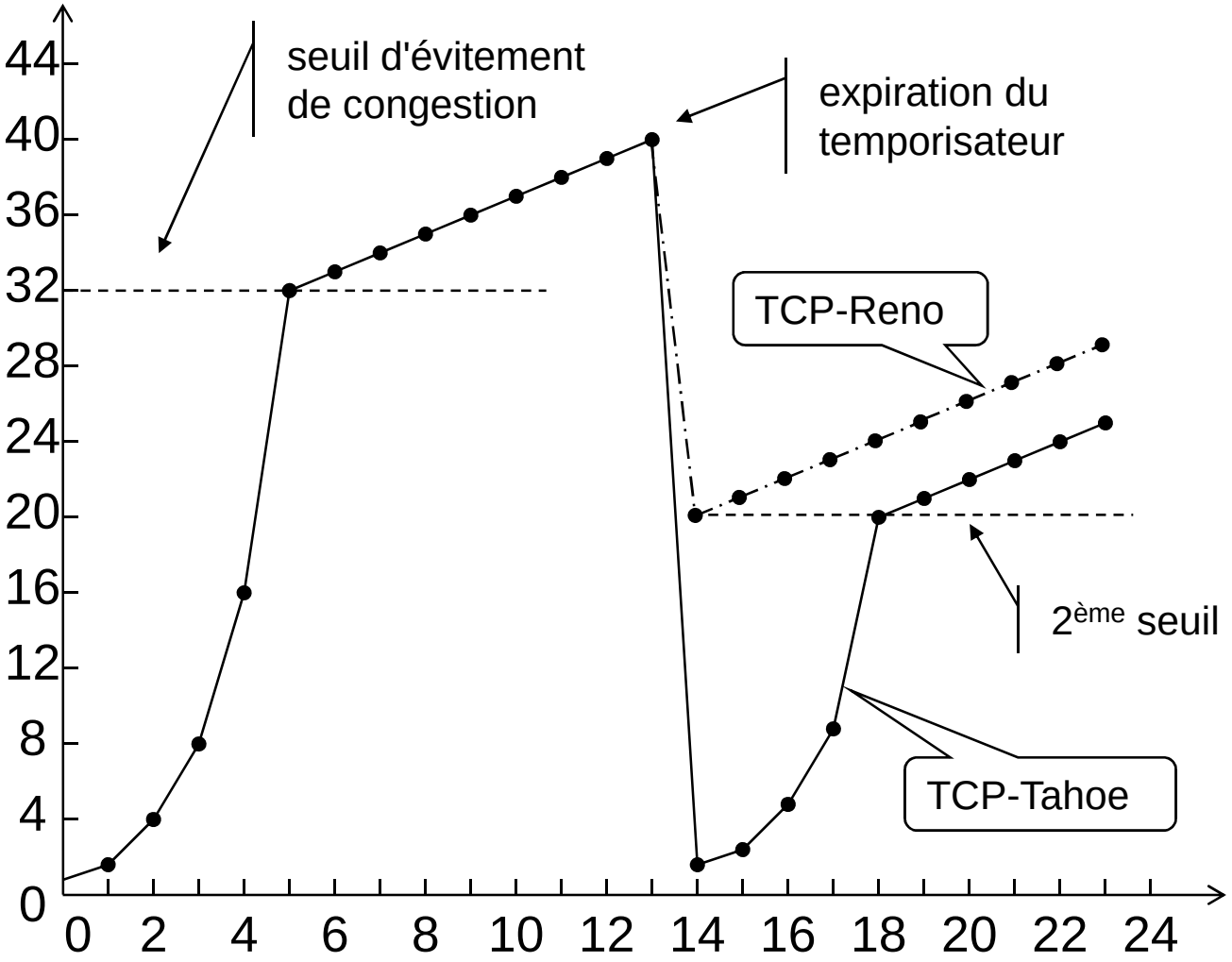
- La taille de la fenêtre peut être modifiée:
  - le destinataire envoie une nouvelle taille de fenêtre avec l'ACK du paquet précédent
  - diminuer la taille de la fenêtre en cas de surcharge (voire taille nulle pour stopper l'émission)
- Champ utilisé dans l'en-tête:
  - Champ "Fenêtre":
    - ➔ donne la taille maximale de la fenêtre en octet que la machine peut accepter
- Un mécanisme de contrôle de flux est indispensable:
  - hétérogénéité des machines:
    - ➔ TCP résoud ce problème
  - réseaux et routeurs de différentes capacités:
    - ➔ TCP se reposait sur ICMP (source quench) mais obsolète depuis RFC 6633, mai 2012

Contrôle de flux (2/3)

- TCP utilise un algorithme adaptatif pour éviter la congestion:
  - prendre en compte le temps de transmission aller/retour
    RTT [Round Trip Time] = temps de boucle moyen
  - prendre en compte le taux de perte
- Principe de l'algorithme:
  - démarrage lent:
    - commencer par fenêtre = 1 segment
    - si ACK OK alors doubler la taille de la fenêtre
  - Phase d ’évitement de la congestion:
    - lorsque la taille de la fenêtre > seuil d'évitement de congestion (32 Ko au démarrage) alors augmenter la fenêtre de 1 segment à chaque fois
  - diminution dichotomique:
    - si paquet perdu alors seuil d'évitement de congestion = taille fenêtre /2
    - recommencer le démarrage lent (TCP-Tahoe) ou la phase d’évitement de la congestion (TCP-Reno)

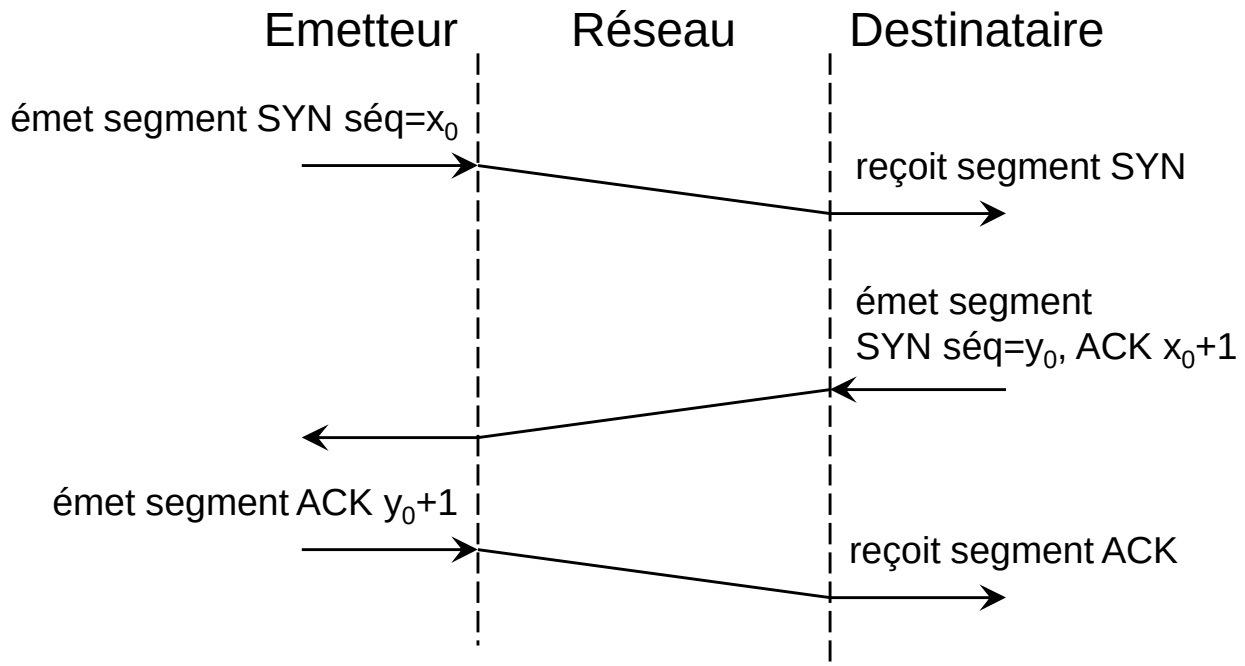
Contrôle de flux (3/3)

- Exemple:



Etablissement d'une connexion TCP

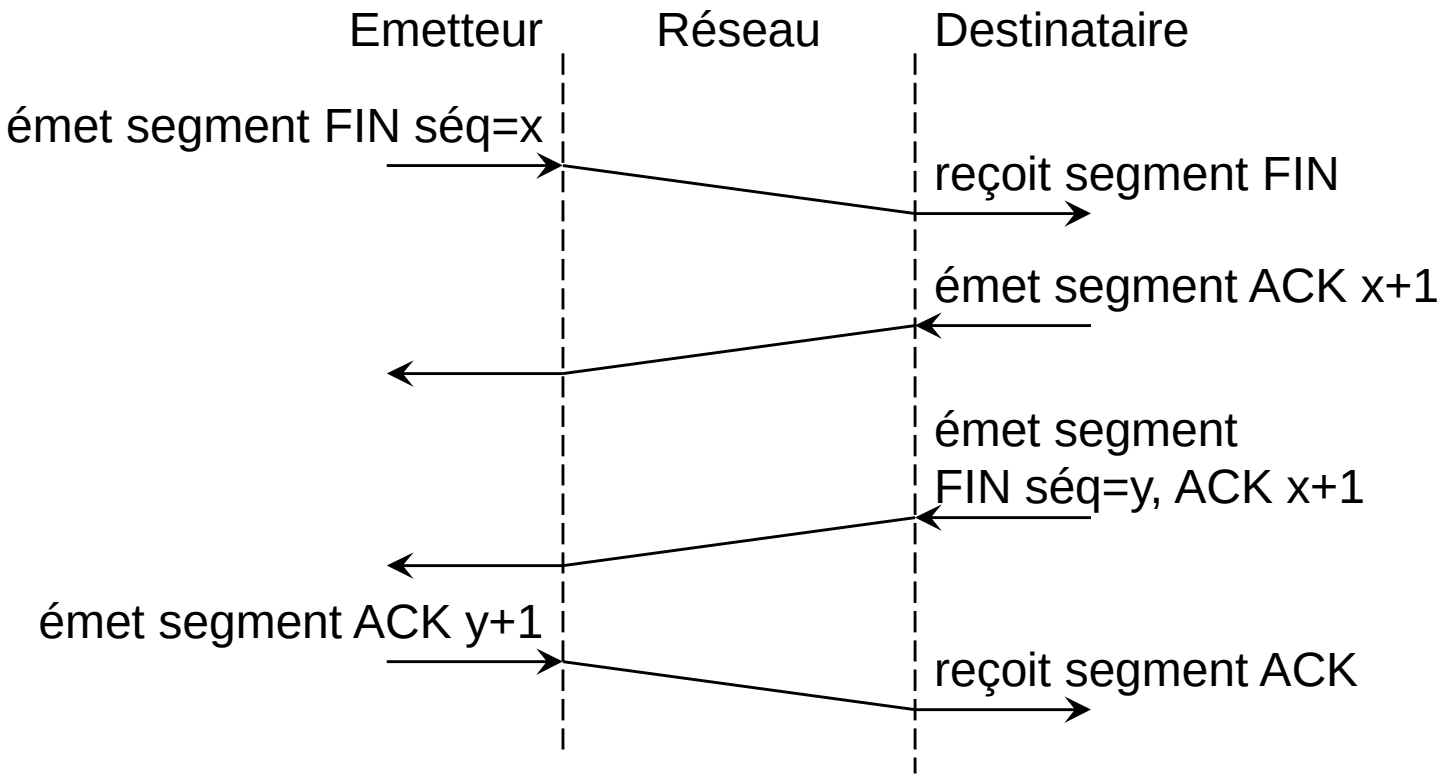
- Connexion en 3 temps:



- Toute demande de connexion supplémentaire qui arrive après que la connexion est établie doit être ignorée:
  - permet de résoudre les problèmes posés par la non-fiabilité de IP
  - résout aussi les cas de connexions simultanées

Libération d'une connexion TCP

- Libération en 3 temps modifiés:

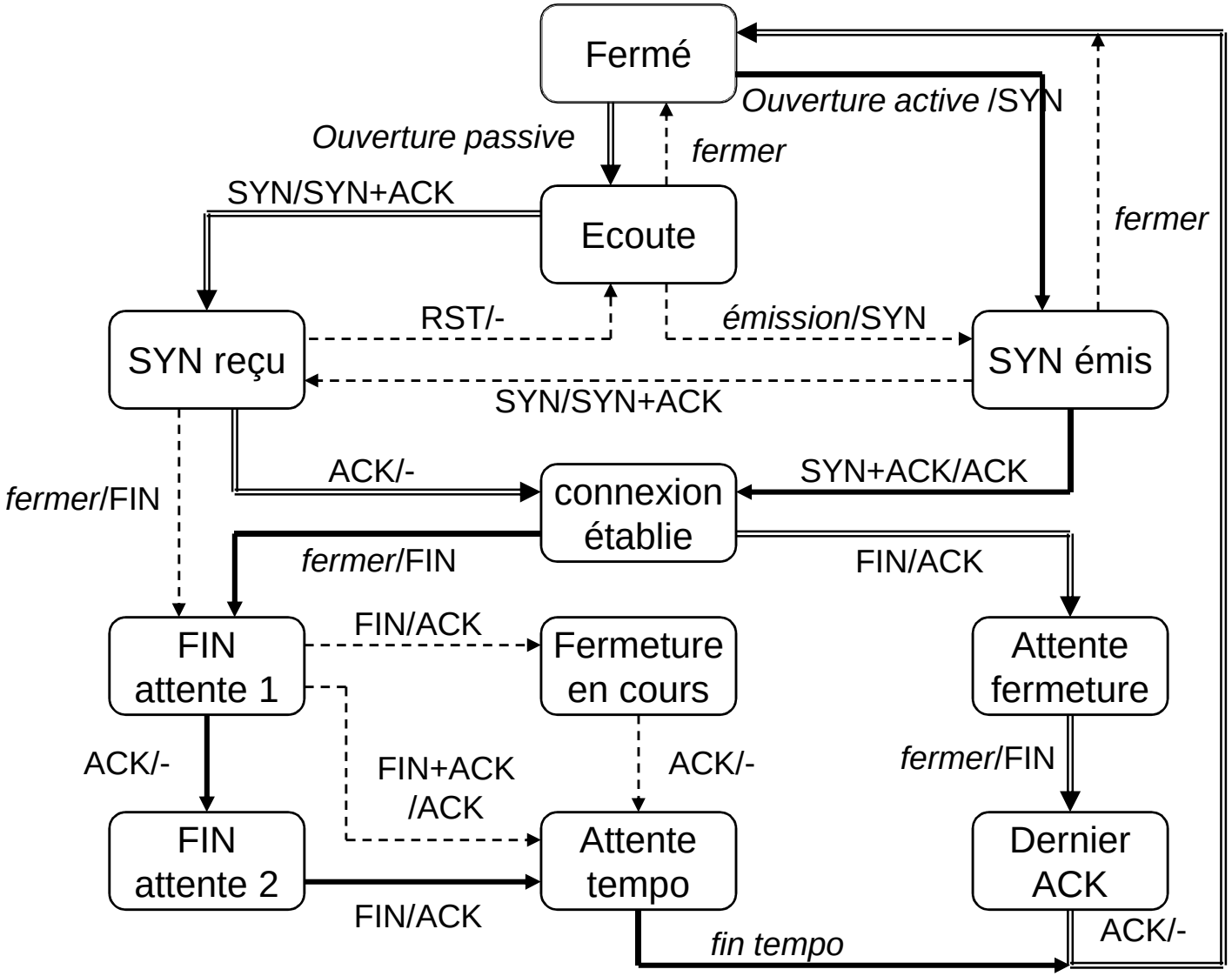


- Les 2 flux étant indépendants, la libération doit fermer chaque sens

Options et données urgentes

- Négociation du plus grand segment:
  - Utilisation du champ "options" pour négocier le MSS [Maximum Segment Size] = taille maximale de segment
  - Idée: choisir la plus grande valeur mais éviter la fragmentation
  - En général:
    - si sur le même réseau, prendre le MTU du réseau diminué de 40 (exemple sur Ethernet = 1460)
    - sinon on conseille 536 (576-40)
- Données urgentes:
  - Permettre d'envoyer des données prioritaires sur le flot normal ==> utiles pour "interrompre" un programme qui fonctionne mal
  - Champs utilisés:
    - indicateur URG=1
    - "pointeur données urgentes": indique le dernier octet de données urgentes (avant les données normales)

Automate à nombre d'états finis de TCP



Légende:

chemin normal serveur

chemin normal client

autres

évén^{ent}/réponse

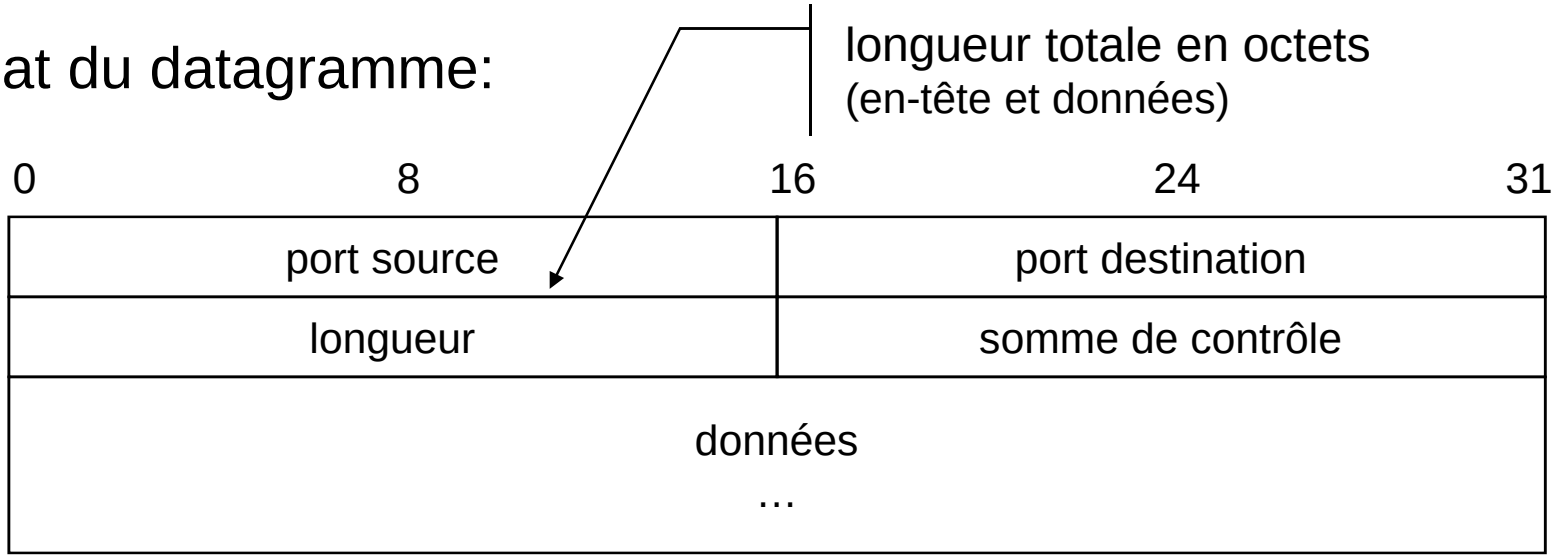
3

3 - Protocole UDP

- UDP [User Datagram Protocol] fournit un service de transport de datagrammes, basiquement le même que IP
- Caractéristiques :
  - mode **non-connecté**
  - transfert **non fiable** des données (la fiabilité des réseaux traversés)
  - taille maximale d'un datagramme= 64 Ko
- Il ajoute seulement à IP le multiplexage grâce à l'utilisation des ports (Cf. TCP):
  - ➔ le port source est facultatif (=0 si non utilisé): il identifie un port pour la réponse
- Il est défini par la RFC 768 (STD 6)

datagramme UDP

- Format du datagramme:



- Somme de contrôle:
  - facultative (=0 si non utilisé)
  - calculée avec un pseudo-en-tête qui contient les adresses IP source et destination (Cf. TCP)
- Quelques exemples d'applications utilisant UDP:
  - NFS: partage de fichiers
  - DNS: service de nommage
  - RTP [Real-Time Protocol]: support d'applications temps-réel

4

4 - Autres protocoles transport

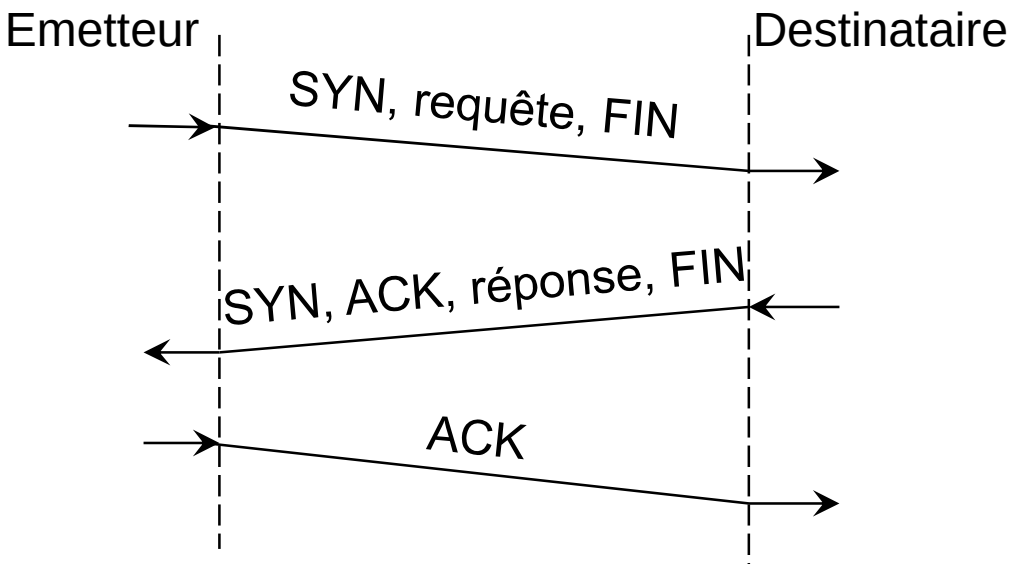
- ISO :
  - TP [Transport Protocol] : transport du modèle OSI
- IETF :
  - T/TCP, Transactional TCP
  - SCTP, Stream Control Transmission Protocol
  - DCCP, Datagram Congestion Control Protocol
  - RSVP, Resource reservation protocol

Transport OSI

- A l'ISO, dans le Transport OSI, 5 classes sont définies :
  - classe 0: classe de base, le minimum pour connexion et échange de données entre machines
  - classe 1: idem classe 0 + reprise sur erreur (pour les erreurs détectées par la couche 3)
  - classe 2: idem classe 0 + multiplexage et contrôle de flux (opt.^{el})
  - classe 3: classe 1 + classe 2
  - classe 4: idem classe 3 + détection des erreurs (non détectées par la couche 3) et reprise sur ces erreurs
    --> similaire au protocole TCP
- En pratique:
  - la classe 0 (appelée TP0) est utilisée sur des réseaux fiables pour des transferts simples
  - la classe 4 (appelée TP4) est utilisée sinon

T/TCP

- Problème des petits échanges de données :
  - UDP : OK pour taille mais pas de fiabilité
  - TCP : fiable mais lourdeur de connexion/déconnexion pour un seul segment de données
- T/TCP [Transactional TCP], RFC 1379, RFC 1644 :
  - Autoriser le transfert de données dans le paquet de demande de connexion
  - Echange complet en 3 paquets
  - Bien adapté aux appels de procédure à distance (RPC)

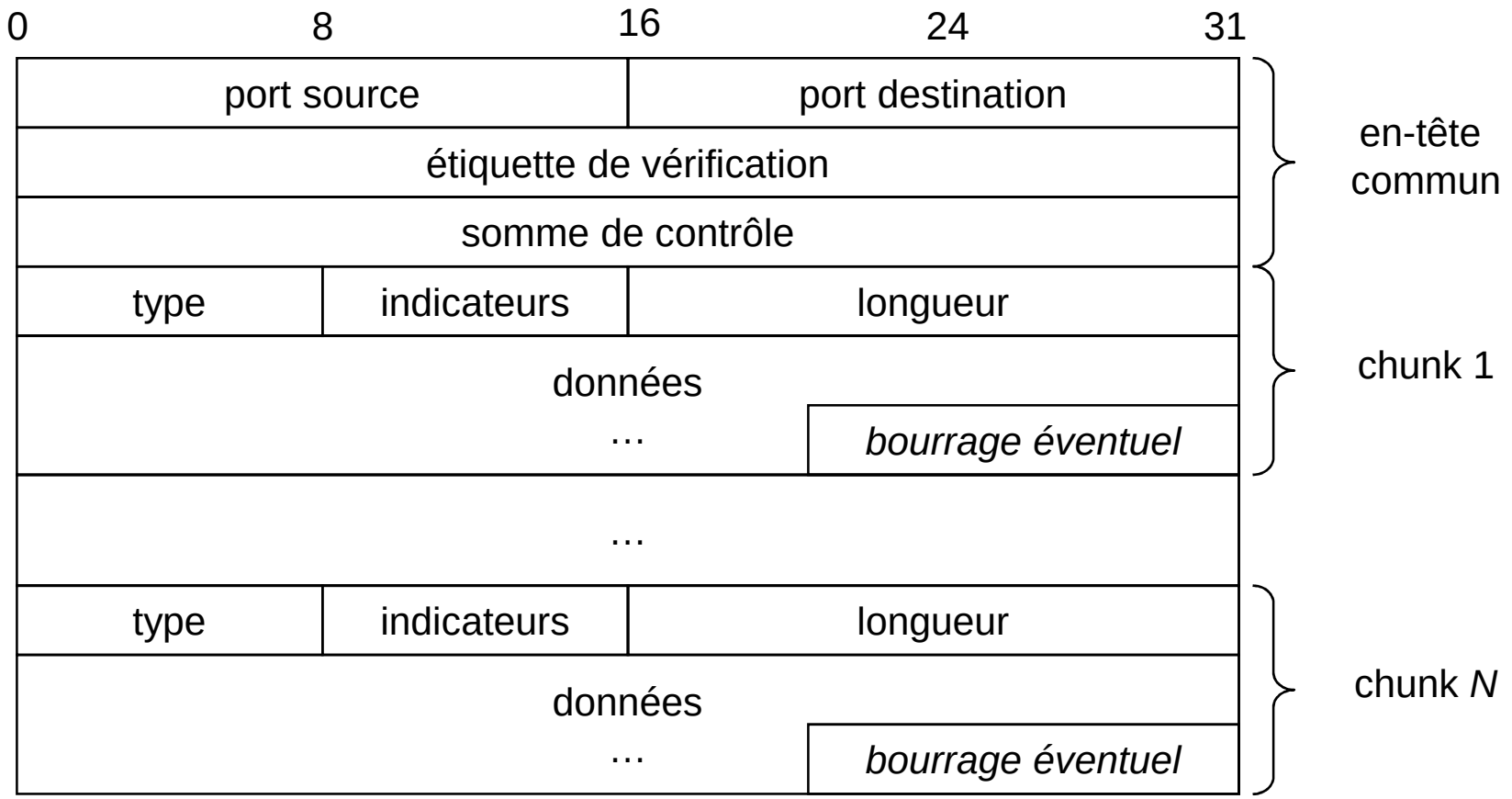


SCTP [1/5]

- SCTP [Stream Control Transmission Protocol], RFC 4960 (sept. 2007) :
  - Groupe de travail depuis 2000 : conçu à l'origine pour transporter la signalisation téléphonique sur Internet
  - Applications plus larges : peut être vu comme un remplaçant de TCP et d'UDP, mais difficile de changer les habitudes !
- Caractéristiques principales :
  - Mode-connecté : une même association SCTP peut utiliser plusieurs adresses IP (cas des machines multi-connectées)
  - Orienté datagramme
  - Transport fiable des datagrammes mais préservation de l'ordre des datagrammes facultative
  - Multi-flots : plusieurs flux de données en parallèles possibles
  - Fragmentation/réassemblage des messages de l'application
  - Contrôle de flux

SCTP [2/5]

- Format du paquet :



SCTP [3/5]

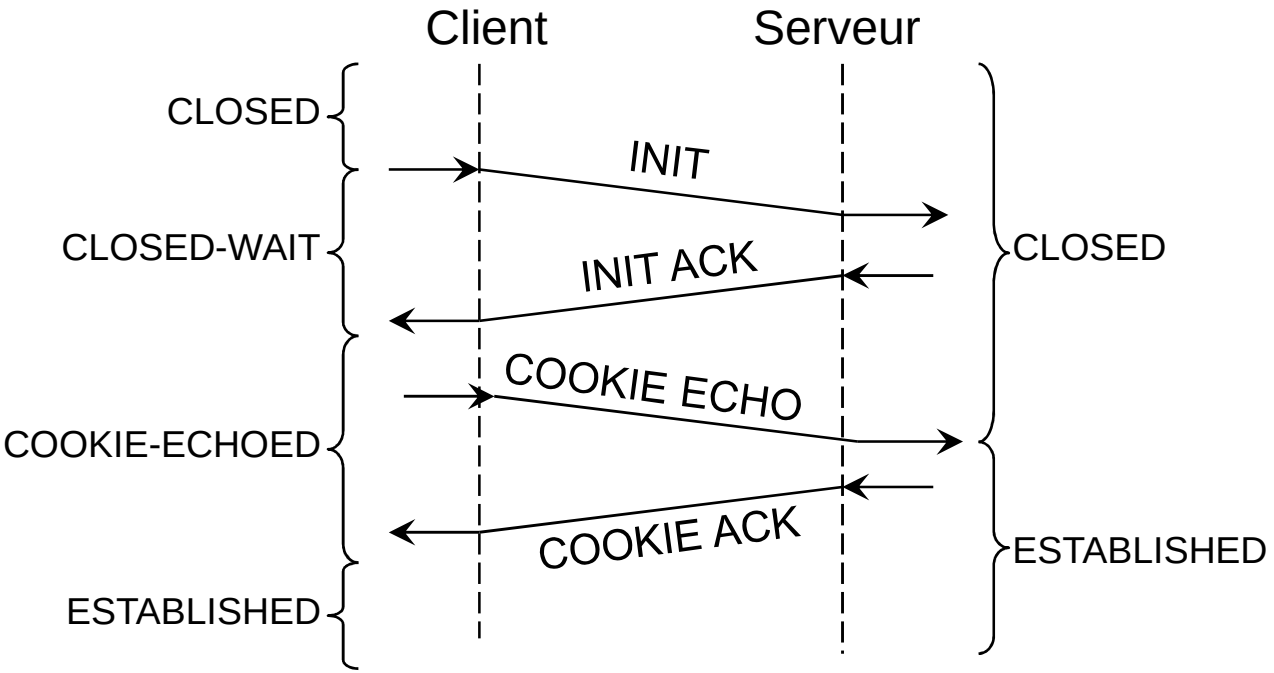
- Chunks :
  - 12 types définis
  - Extensions possibles (256 valeurs) avec plusieurs valeurs réservées

0	DATA : données
1	INIT : initialisation
2	INIT ACK
3	SACK : acquittement sélectif
4	HEARTBEAT
5	HEARBEAT ACK
6	ABORT : déconnexion brutale
7	SHUTDOWN : déconnexion négociée
8	SHUTDOWN ACK
9	ERROR
10	COOKIE ECHO
11	COOKIE ACK
14	SHUTDOWN COMPLETE

- Identification de la terminaison:
  - 1 seule interface réseau :
    - → [Adr IP : port SCTP]
  - Plusieurs interfaces :
    - → [IP1, IP2, IP3 : port SCTP]
  - Association identifiée par:
    - → { [IP1, IP2, IP3 : port 1], [IP4, IP5 : port 2] }

SCTP [4/5]

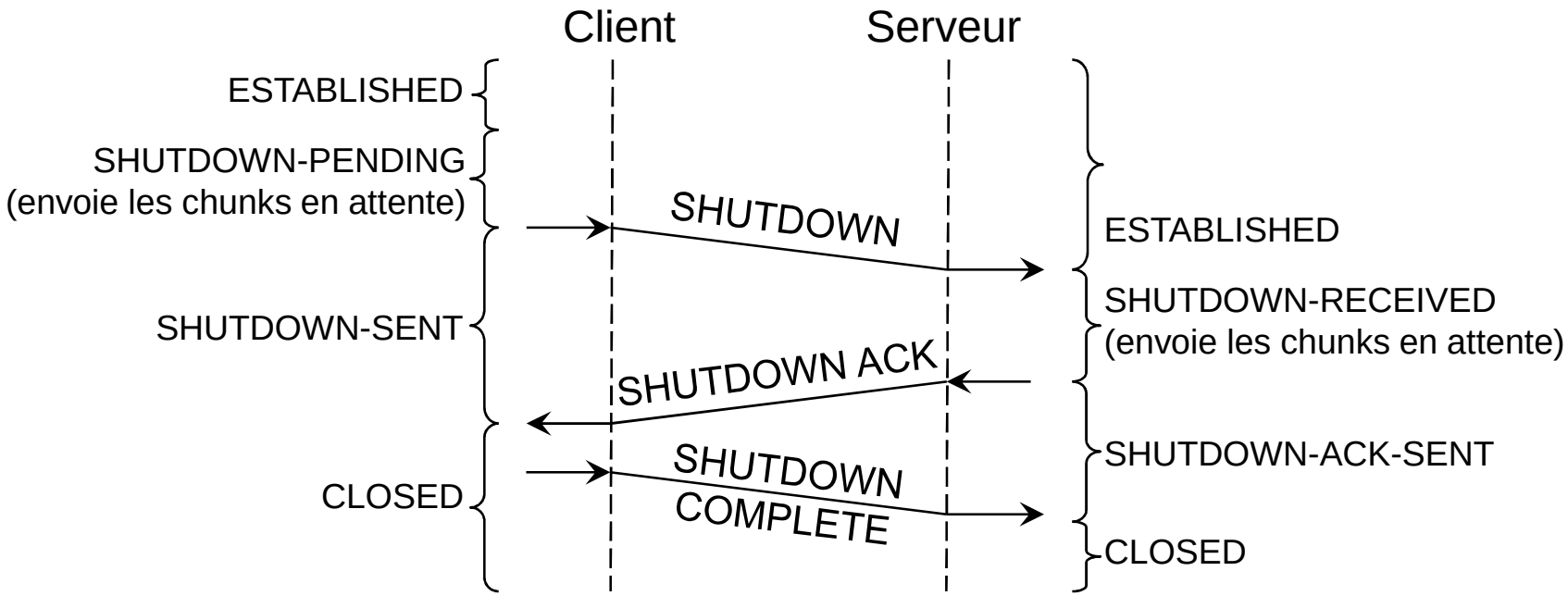
- Etablissement d'une association :
  - Etablissement en 4 temps :



  - INIT-ACK renvoie un cookie qui est retransmis par le client :
    - Eviter les « SYN attacks » comme dans TCP
    - Vérifier qu'il n'est pas modifié : signature du cookie
  - Plus de ½ connexions, le serveur passe directement de l'état CLOSED à l'état ESTABLISHED

SCTP [5/5]

- Fermeture d'une association :
  - Fermeture en 3 temps initiée par la couche supérieure :



  - Pas d'état ½ fermé comme dans TCP
  - Il existe aussi une fermeture brutale par ABORT initiée par la couche transport

Comparaison protocoles transports

Spécifications	*UDP*	*TCP*	*SCTP*	*DCCP*
Type de TPDU	datagramme	segment	datagramme	datagramme
Nature du flux	message	octets	message	message
Multiplexage (ports)	oui	oui	oui	oui
Détection des erreurs	optionnel	oui	oui	oui
Fiabilité par retransmission auto.	non	oui	oui	non
Circuits virtuels (ordonnancement)	non	oui	optionnel	oui
Contrôle de flux et gestion de la congestion	non	oui	oui	oui
Flux multiples de données	non	2 (donnée urgentes)	plusieurs	non
Connexion	non	en 3 temps	en 4 temps	oui
Déconnexion	non	en 4 temps	en 3 temps	oui