

Systèmes et réseaux



Chapitre 8 Couche transport



Pablo Rauzy <pr@up8.edu>
pablo.rauzy.name/teaching/sr

Couche transport

- ▶ La couche transport est la quatrième couche du modèle OSI.
- ▶ Elle est en charge des communications de bout en bout entre *services*.
- ▶ C'est généralement la plus haute couche où on se préoccupe des erreurs de transmissions.
- ▶ Le service offert par cette couche est donc une abstraction complète des services réseaux fournis par les couches du dessous : au dessus, on transfère simplement des octets, potentiellement garantis sans corruptions ou pertes.

Principaux protocoles

- ▶ Les deux principaux protocoles de niveau 4 sont incontestablement UDP et TCP.
 - On a déjà vu un cas d'utilisation de chacun de ces protocoles lors de la séance d'introduction sur le chargement d'une page web.

- ▶ UDP signifie *User Datagram Protocol*.
- ▶ Son rôle est de permettre simplement et à moindre frais la transmission de données entre deux services définis par une adresse IP et un *numéro de port*.
- ▶ Il ne fournit aucune garantie sur la livraison des datagrammes, ni sur leur ordre d'arrivée.

Cas d'utilisations

- ▶ UDP est utilisé essentiellement dans deux cas :
 - pour transmettre des petites quantités de données,
 - pour transmettre rapidement beaucoup de données, dans les cas où perdre un datagramme de temps en temps n'est pas grave.
- ▶ Le premier cas d'utilisation typique est le DNS.
- ▶ Le second cas d'utilisation sont le streaming, la VoIP, ou les jeux en ligne.

Entêtes UDP

► Les entêtes d'UDP sont réduits au strict minimum :

- le port source,
- le port destination,
- longueur totale du datagramme (entêtes+données),
- somme de contrôle.

Port Source (16 bits)	Port Destination (16 bits)
Longueur (16 bits)	Somme de contrôle (16 bits)
Données (longueur variable)	

La somme de contrôle

- ▶ Elle est calculée non seulement sur les entêtes UDP, mais aussi sur :
 - les données,
 - certains des entêtes IP.
- ▶ Il existe une variante UDP-Lite qui permet de choisir sur combien d'octets à partir du début du paquet est calculée la somme de contrôle.

+	Bits 0 - 7	8 - 15	16 - 23	24 - 31
-96	Adresse Source			
-64	Adresse Destination			
-32	Zéros	Protocole	Taille UDP	
0	Port Source		Port Destination	
32	Longueur		Somme de contrôle	
64	Données			

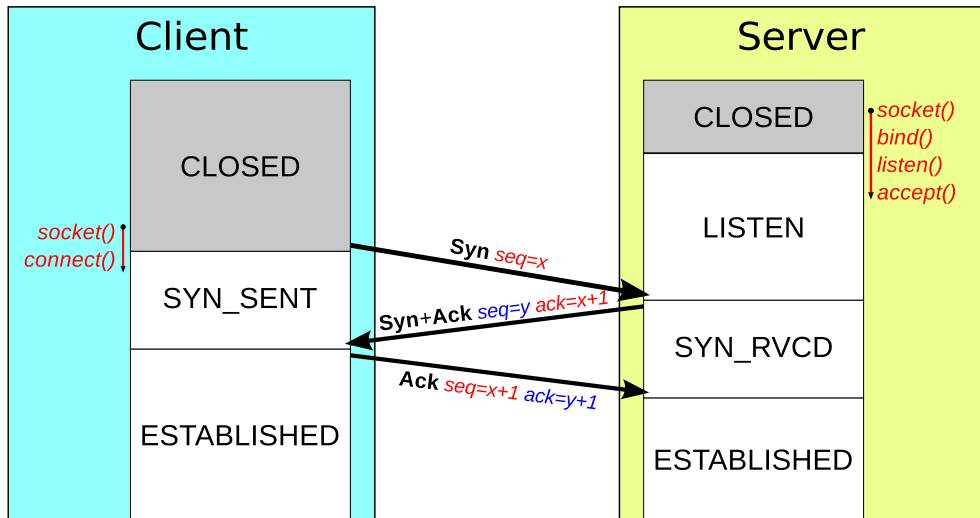
- ▶ Pour ouvrir une socket UDP, on utilise le type `SOCK_DGRAM`.
- ▶ Rappel :
 - `#include <sys/socket.h>`
`int socket(int domain, int type, int protocol);`
- ▶ Avec zéro comme protocole il sera choisit automatiquement en fonction du type.
 - Pour faire les choses parfaitement et proprement, il faudrait utiliser `getprotobyname` avec `"udp"` pour obtenir une `struct protoent` dont le membre `p_proto` contient la bonne valeur pour le paramètre `protocol` (en l'occurrence, 17).

- ▶ TCP signifie *Transmission Control Protocol*.
- ▶ C'est un protocole de transport *fiable* en *mode connecté*.
- ▶ Fiable, cela signifie que TCP garanti aux couches d'au dessus que :
 - les *segments* arrivent tous à destination,
 - les segments sont dans l'ordre à destination,
 - les données échangées sont intègres.
- ▶ Une *session* TCP se passe en 3 étapes :
 - l'ouverture de la connexion,
 - l'échange de données,
 - la fermeture de la connexion.

L'ouverture de la connexion

- ▶ Dans TCP, la connexion est ouverte par un *handshake* en trois temps entre le client et le serveur.
- ▶ Le but de cet échange est d'établir des numéros de séquence qui permettront de garantir l'ordre et la complétude de l'ensemble des segments transmit.
- ▶ L'ouverture se passe en 3 étapes :
 - Le client envoie un segment SYN au serveur avec un numéro de séquence aléatoire A.
 - Le serveur répond au client avec un segment SYN-ACK.
Il contient un numéro d'acquittement qui vaut A+1.
Il contient aussi un numéro de séquence aléatoire B.
 - Le client répond au serveur avec un segment ACK.
Le numéro de séquence du segment vaut A+1.
Le numéro d'acquittement vaut B+1.

Schéma



L'échange de données

- ▶ Pendant l'échange de données :
 - les sommes de contrôle sont utilisées pour vérifier l'intégrité des segments,
 - les numéros de séquence et acquittement pour vérifier que l'ensemble des données est bien arrivé et est ordonnable.

L'échange de données

- ▶ Pendant l'échange de données :
 - les sommes de contrôle sont utilisées pour vérifier l'intégrité des segments,
 - les numéros de séquence et acquittement pour vérifier que l'ensemble des données est bien arrivé et est ordonnable.
- ▶ Comme pour UDP la somme de contrôle est calculée sur les entêtes et les données.

L'échange de données

- ▶ Pendant l'échange de données :
 - les sommes de contrôle sont utilisées pour vérifier l'intégrité des segments,
 - les numéros de séquence et acquittement pour vérifier que l'ensemble des données est bien arrivé et est ordonnable.
- ▶ Comme pour UDP la somme de contrôle est calculée sur les entêtes et les données.
- ▶ À chaque transfert de segment :

L'échange de données

- ▶ Pendant l'échange de données :
 - les sommes de contrôle sont utilisées pour vérifier l'intégrité des segments,
 - les numéros de séquence et acquittement pour vérifier que l'ensemble des données est bien arrivé et est ordonnable.
- ▶ Comme pour UDP la somme de contrôle est calculée sur les entêtes et les données.
- ▶ À chaque transfert de segment :
 - l'émetteur envoie :
 - numéro de séquence : initial + nombre d'octets envoyés,
 - numéro d'acquittement : prochain numéro de séquence attendu du destinataire ;

L'échange de données

- ▶ Pendant l'échange de données :
 - les sommes de contrôle sont utilisées pour vérifier l'intégrité des segments,
 - les numéros de séquence et acquittement pour vérifier que l'ensemble des données est bien arrivé et est ordonnable.
- ▶ Comme pour UDP la somme de contrôle est calculée sur les entêtes et les données.
- ▶ À chaque transfert de segment :
 - l'émetteur envoie :
 - numéro de séquence : initial + nombre d'octets envoyés,
 - numéro d'acquittement : prochain numéro de séquence attendu du destinataire ;
 - le destinataire répond avec un segment ACK :
 - numéro de séquence : numéro d'acquittement du segment reçu,
 - numéro d'acquittement : numéro de séquence reçu augmenté du nombre d'octets reçus ;

L'échange de données

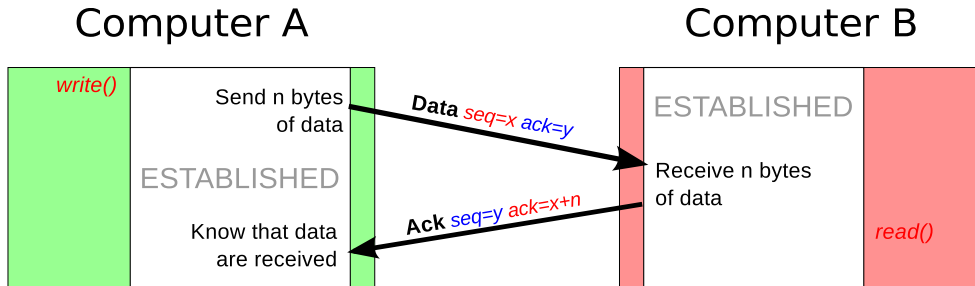
- ▶ Pendant l'échange de données :
 - les sommes de contrôle sont utilisées pour vérifier l'intégrité des segments,
 - les numéros de séquence et acquittement pour vérifier que l'ensemble des données est bien arrivé et est ordonnable.
- ▶ Comme pour UDP la somme de contrôle est calculée sur les entêtes et les données.
- ▶ À chaque transfert de segment :
 - l'émetteur envoie :
 - numéro de séquence : initial + nombre d'octets envoyés,
 - numéro d'acquittement : prochain numéro de séquence attendu du destinataire ;
 - le destinataire répond avec un segment ACK :
 - numéro de séquence : numéro d'acquittement du segment reçu,
 - numéro d'acquittement : numéro de séquence reçu augmenté du nombre d'octets reçus ;
 - si au bout d'un temps limite (qui est adapté au fur et à mesure des échanges) l'émetteur ne reçoit pas d'ACK, il renvoie le segment,

L'échange de données

- ▶ Pendant l'échange de données :
 - les sommes de contrôle sont utilisées pour vérifier l'intégrité des segments,
 - les numéros de séquence et acquittement pour vérifier que l'ensemble des données est bien arrivé et est ordonnable.
- ▶ Comme pour UDP la somme de contrôle est calculée sur les entêtes et les données.
- ▶ À chaque transfert de segment :
 - l'émetteur envoie :
 - numéro de séquence : initial + nombre d'octets envoyés,
 - numéro d'acquittement : prochain numéro de séquence attendu du destinataire ;
 - le destinataire répond avec un segment ACK :
 - numéro de séquence : numéro d'acquittement du segment reçu,
 - numéro d'acquittement : numéro de séquence reçu augmenté du nombre d'octets reçus ;
 - si au bout d'un temps limite (qui est adapté au fur et à mesure des échanges) l'émetteur ne reçoit pas d'ACK, il renvoie le segment,
 - si le destinataire reçoit deux fois le même segment il s'en rend compte grâce au numéro de séquence et ignore le second,

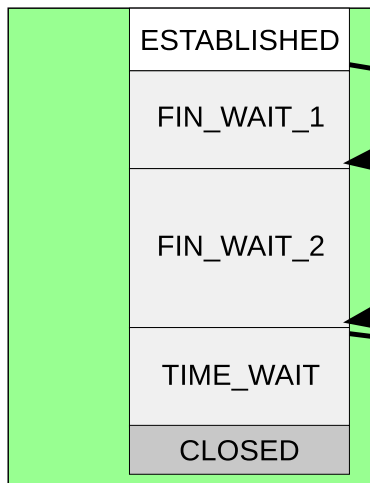
L'échange de données

- ▶ Pendant l'échange de données :
 - les sommes de contrôle sont utilisées pour vérifier l'intégrité des segments,
 - les numéros de séquence et acquittement pour vérifier que l'ensemble des données est bien arrivé et est ordonnable.
- ▶ Comme pour UDP la somme de contrôle est calculée sur les entêtes et les données.
- ▶ À chaque transfert de segment :
 - l'émetteur envoie :
 - numéro de séquence : initial + nombre d'octets envoyés,
 - numéro d'acquittement : prochain numéro de séquence attendu du destinataire ;
 - le destinataire répond avec un segment ACK :
 - numéro de séquence : numéro d'acquittement du segment reçu,
 - numéro d'acquittement : numéro de séquence reçu augmenté du nombre d'octets reçus ;
 - si au bout d'un temps limite (qui est adapté au fur et à mesure des échanges) l'émetteur ne reçoit pas d'ACK, il renvoie le segment,
 - si le destinataire reçoit deux fois le même segment il s'en rend compte grâce au numéro de séquence et ignore le second,
 - dans chaque segment est indiqué une taille de buffer disponible chez son émetteur, qui signale combien il peut recevoir d'octet avant d'envoyer un ACK.

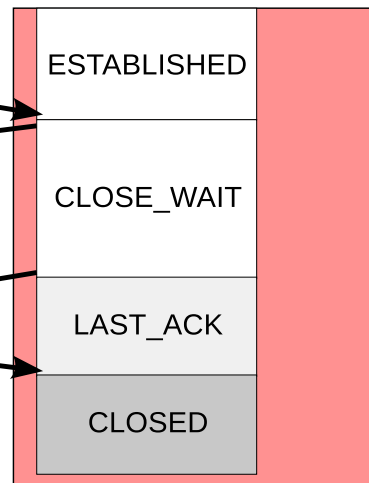


- ▶ La fermeture de connexion se fait indépendamment de chaque côté mais avec confirmation, donc en quatre temps en tout.
- ▶ D'abord un des deux participants A ferme sa connexion vers l'autre B :
 - A envoie un segment FIN avec son numéro de séquence courant X,
 - B répond un segment ACK avec comme numéro d'acquittement X+1.
- ▶ Ensuite, B ferme sa connexion vers A :
 - B envoie un segment FIN avec son numéro de séquence courant Y.
 - A répond un segment ACK avec comme numéro d'acquittement Y+1.

Computer A



Computer B



Fin $seq=x$

Ack $ack=x+1$

*Connection is
half-closed
Computer B can still
send data to A*

Fin $seq=y$

Ack $ack=y+1$

Entêtes TCP

Offsets	Octet	0								1								2								3																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																					
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
0	0	Source port																Destination port																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
4	32	Sequence number																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
8	64	Acknowledgment number (if ACK set)																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													
12	96	Data offset				Reserved 0 0 0			N S	C	E	U	A	P	R	S	F	Window Size																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																													

- ▶ Pour ouvrir une socket TCP, on utilise le type `SOCK_STREAM`.
- ▶ Rappel :
 - `#include <sys/socket.h>`
`int socket(int domain, int type, int protocol);`
- ▶ Avec zéro comme protocole il sera choisit automatiquement en fonction du type.
 - Pour faire les choses parfaitement et proprement, il faudrait utiliser `getprotobyname` avec `"tcp"` pour obtenir une `struct protoent` dont le membre `p_proto` contient la bonne valeur pour le paramètre `protocol` (en l'occurrence, 6).

► Un mot rapide sur QUIC.

- QUIC a été introduit il y a une dizaine d'année, son déploiement n'est pas généralisé.
- Il s'agit d'un protocole de niveau transport aussi mais qui pour des raisons de compatibilité avec le matériel et les systèmes existants est encapsulé dans UDP.
- Il a pour objectif de remplacer TCP notamment pour les connexions HTTP (en v2 et v3) pour lesquels il permet le *multiplexing*.
- Il inclut aussi directement TLS (*transport layer security*) plutôt que d'en faire une couche supplémentaire comme le fait TCP.