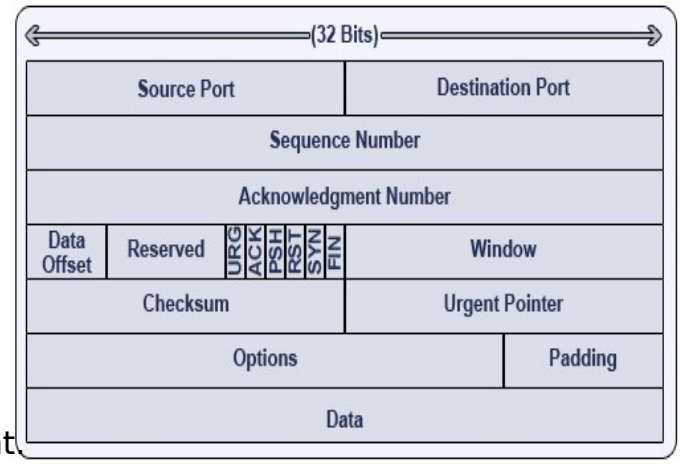
Protocole TCP

TCP (*Transmission Control Protocol*) est un protocole de transport **fiable**, en mode connecté (RFC 793) qui assure la transmission des données de bout en bout (d'un processus à un autre processus).

Segment TCP

Une communication TCP est **bidirectionnelle *full duplex*** (flux d'octets).



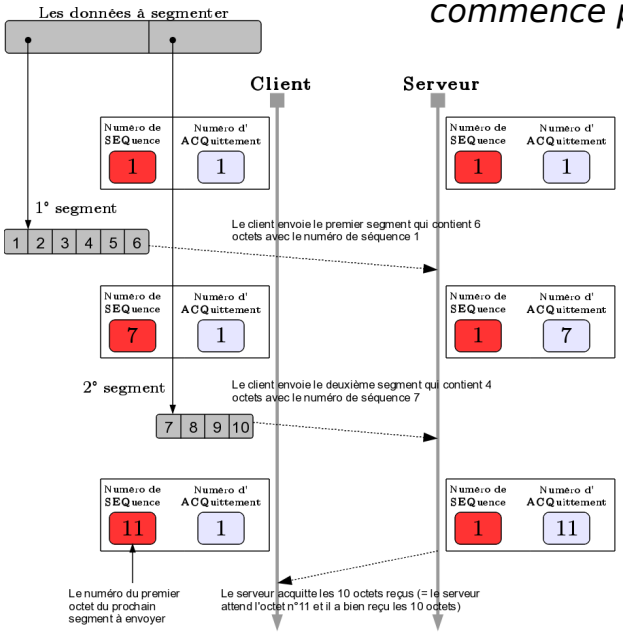
Les données en octets sont préalablement découpées en **segments**. Un segment TCP sera transporté au travers du réseau par un **paquet IP**.

TCP comptabilise tous les octets transmis.

Le champ ***Sequence Number*** indique la place du premier octet de données du segment.

Le champ ***Acknowledgment Number*** indique le prochain octet attendu par l'émetteur du segment.

Remarque : Il y a une numérotation indépendante pour chaque sens de la communication et elle ne commence pas forcément à 0.



Fiabilité

La fiabilité de la transmission est assurée par un mécanisme baptisé ***Positive Acknowledgement with Re-transmission*** **(PAR)**.

L'émetteur démarre une alarme (*timeout*) à chaque envoi de segment : SI l'alarme expire avant l'arrivée d'un acquittement ALORS retransmission des données du segment.

Une communication TCP sera considérée **fiable** car elle est basée sur :

- la numérotation des octets
- la détection des erreurs (*checksum*)
- la détection des pertes (*timeout* et acquittements successifs)
- la récupération des pertes (par retransmission)

*Pour assurer le bon fonctionnement du séquencement des données, le client et le serveur doivent d'abord **synchroniser leurs numéros de séquence initiaux**. Cette synchronisation est réalisée lors de l'établissement de la communication (la phase de connexion TCP).*

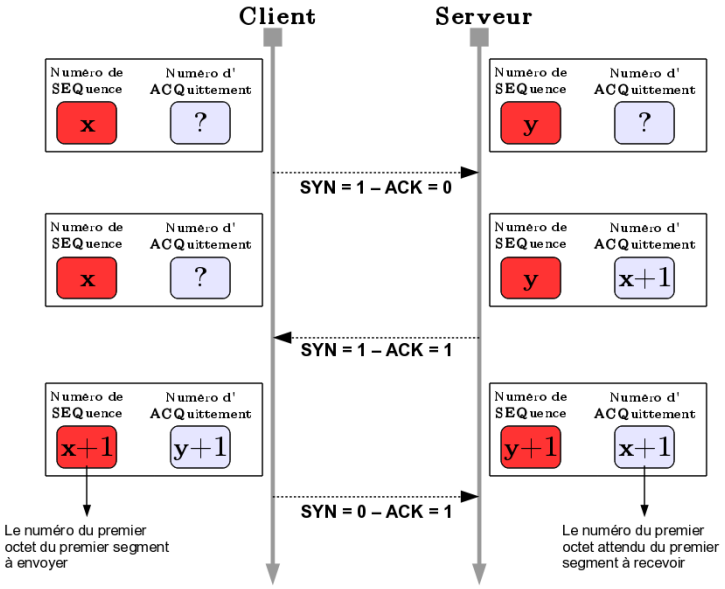
*Les **6 drapeaux (flags)** sont essentiels dans la gestion d'une communication TCP :*

- *0x20 URG : valide le champ Pointeur Urgent*
- ***0x10 ACK : valide le champ Acknowledgment Number***
- *0x08 PSH : indique au récepteur de délivrer immédiatement les données en attente*
- *0x04 RST : demande au récepteur une réinitialisation de la connexion ou met fin à une demande*
- ***0x02 SYN : demande une synchronisation du Sequence Number (connexion TCP)***
- ***0x01 FIN : l'émetteur demande une déconnexion TCP***

Connexion TCP

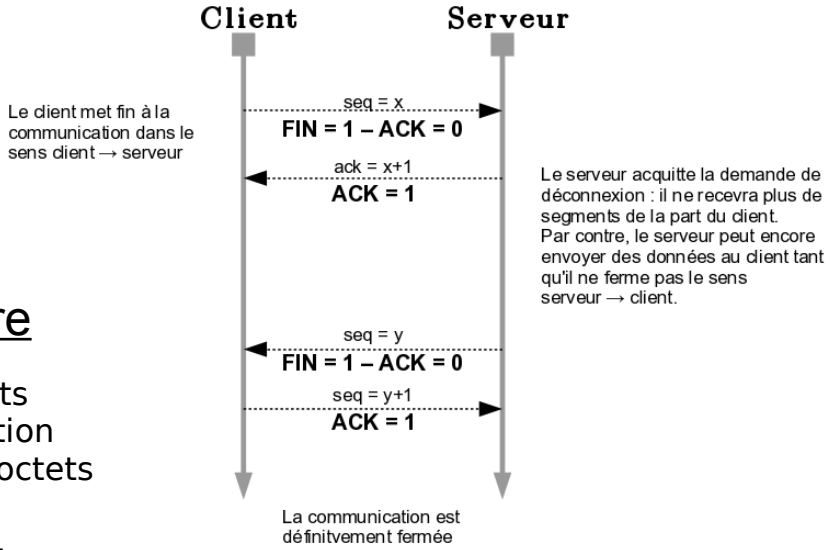
Une ouverture active de connexion TCP est établie “en trois temps“ (***Three Way Handshake***). Il faut synchroniser les deux côtés.

1) Le client TCP initialise la connexion en envoyant un segment incluant un **SYN** (***SYN****chronize sequence numbers*) et un numéro de séquence **x**.
2) Le serveur TCP lui répond par un segment avec les drapeaux **SYN** et **ACK** (***ACK****nowledgement*) avec un numéro d'acquittement **x+1** et son numéro de séquence **y**.
3) Le client TCP termine la connexion avec le ***flag*** **ACK** et le numéro d'acquittement **y+1**. Il peut déjà envoyer des données en même temps.



Déconnexion TCP

Une déconnexion TCP se fera “en quatre temps“. La raison est qu'une connexion TCP est *full-duplex*, ce qui implique que les deux directions doivent pouvoir être fermées indépendamment l'une de l'autre.



Le mécanisme de la fenêtre

La fenêtre définit le nombre d'octets pouvant être envoyés par anticipation (sans attendre l'acquittement des octets précedemment transmis).

Cela dépend de la capacité du buffer (tampon) du récepteur.

La fenêtre est un mécanisme de **contrôle de flux**.

TCP utilise le mécanisme de la fenêtre coulissante ou glissante (***sliding window***) : le nombre d'octets maximum pouvant être émis sans attendre d'acquittement (champ ***Window***).

Chaque machine gère localement ses fenêtres pour chaque sens de transmission (fenêtre d'émission et fenêtre de réception)