Les connexions TCP

Rôle de la couche transport TCP

La couche 4 du modèle OSI est la couche transport avec deux protocoles :

- **TCP** : Transmission Control Protocol)
- **UDP** : User Datagram Protocol

La couche de transport TCP du réseau doit fournir à l 'utilisateur un service de transport d 'information **efficace**, **fiable** et **économique**, c'est à dire une **qualité de service**.

TCP est conçu pour traiter de **bout-en-bout** des données :

- en apportant la **fiabilité** que ne donne pas IP :
  - **détecter** les pertes éventuelles et les **corriger** en retransmettant les données perdues,
  - **ordonner** les datagrammes qui peuvent arriver dé-séquencés.
- en s'**adaptant dynamiquement** aux changements dans le réseau

TCP fonctionne en **mode connecté** :

- création de **2 points de connexion** appelés sockets (couple socket1, socket2) :
  - un à l 'émetteur identifié par l'**adresse IP** + le **numéro de port** (16 bits),
  - un au récepteur identifié par l'**adresse IP** + le **numéro de port** (16 bits)
- communication **bidirectionnelle** ; les données peuvent circuler dans les 2 sens simultanément
- **point-à-point** : 2 points d 'extrémité uniquement ; pas de multicast ou de broadcast.

Les ports TCP

Les numéros de ports source (application source) et port destination (application destinatrice) sont :

- des numéros sur 16 bits,
- Référencés par l'IANA

Les ports vont de 0 à 65535

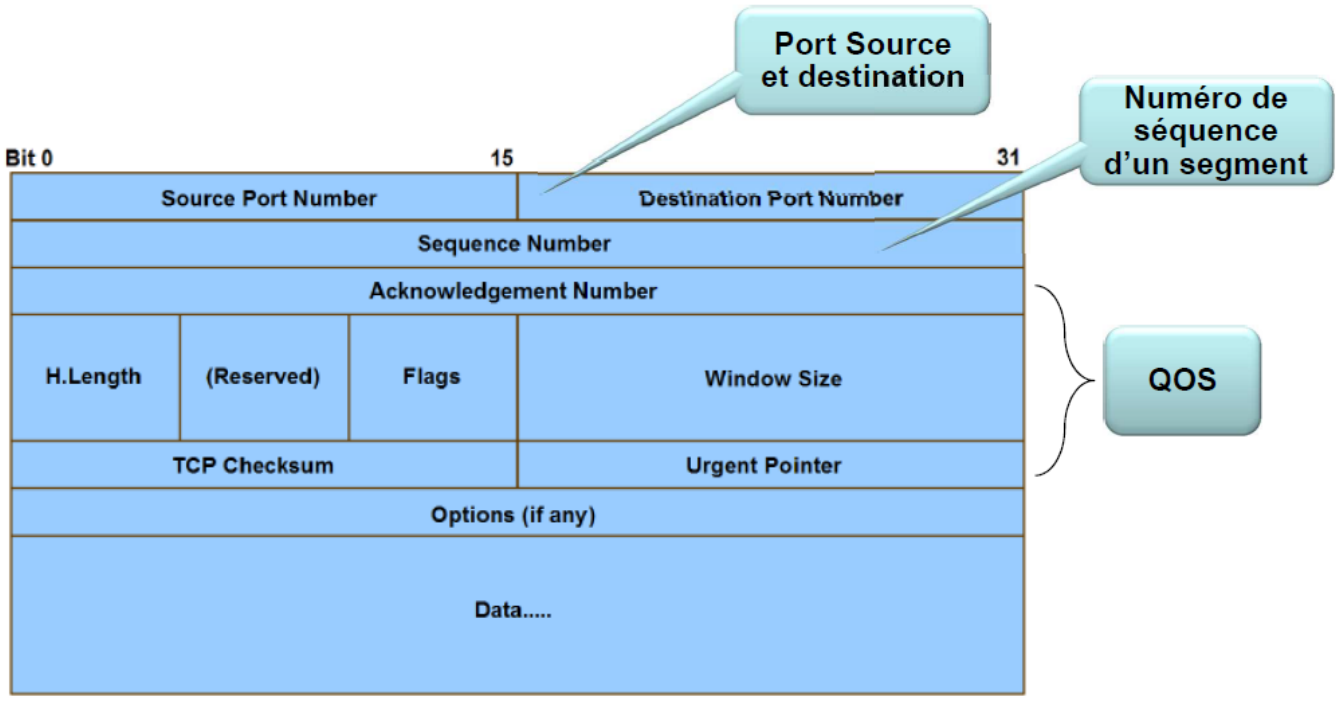
- 0-1023 : **Well-known ports**
- 1024-49151 : **Registered ports**
- 49152-65535 : **Dynamic and/or Private ports**

http://www.iana.org/assignments/service-names-port-numbers/servicenames-port-numbers.xml

L'entête TCP

TCP transmet sur le réseau des segments constitué :

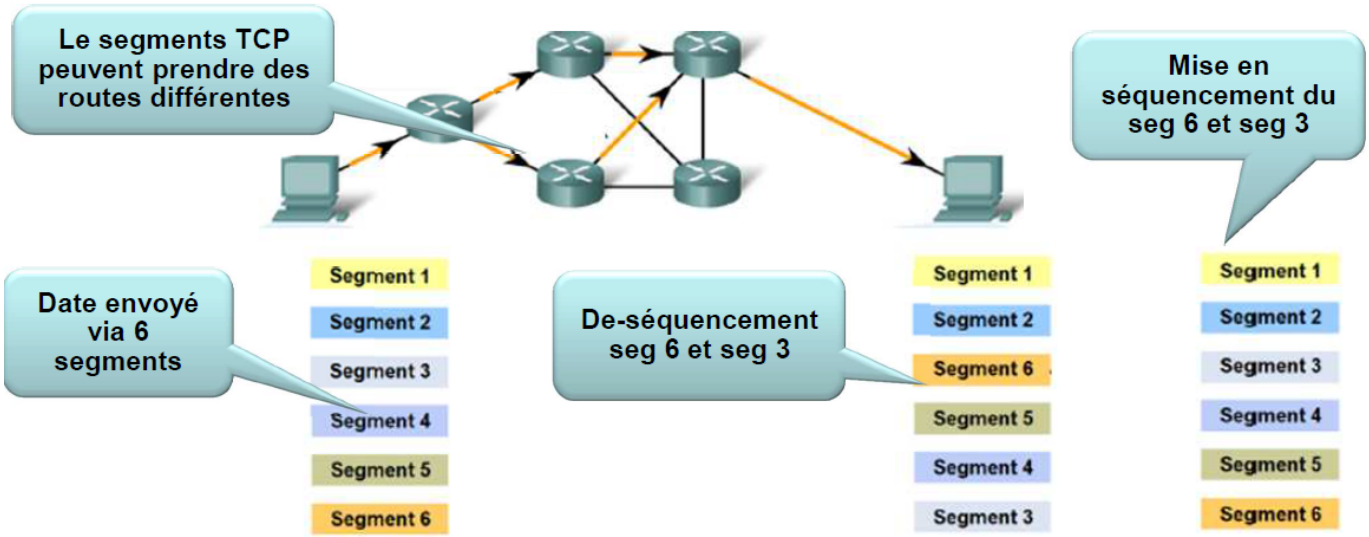
- de 20 octets d'en-tête fixes (plus une partie optionnelle) ;
- 0 ou plusieurs octets de données
- Chaque segment porte un numéro de séquence.



Le numéro de séquence

Troisième champ : Numéro de séquence sur 32 bits

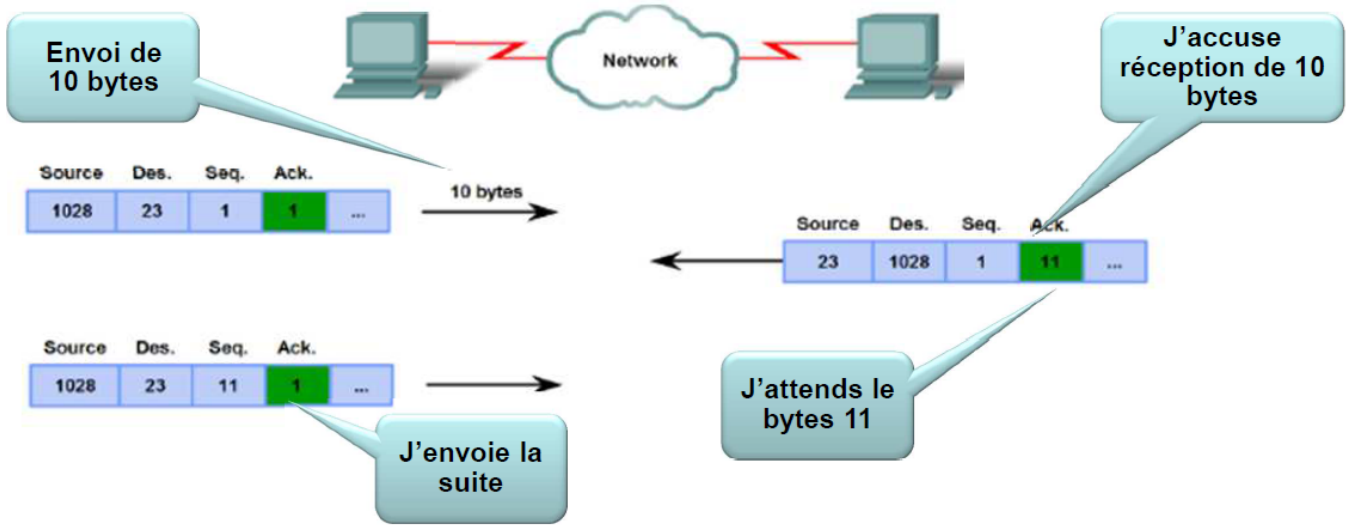
- indique le **numéro** affecté au **premier octet** du segment TCP envoyé par la connexion,
- utilisés pour **reconstruire** le flux de données en replaçant les segments **dans l'ordre**.



Le numéro d'acquitement

Quatrième champ : Numéro d 'acquittement 32 bits

- Numéro du prochain octet attendu par le récepteur, dans le flux TCP
- n'indique pas le numéro du dernier octet correctement reçu.



Le champ Flags

Septième champ : 6 drapeaux 6 foius 1 bit

- 3 premiers bits sont utilisés lors de l'**établisement de la connexion TCP** :
  - **SYN** : **demande** d 'établissement de connexion
  - **ACK** : **validité** du numéro d 'acquittement en indiquant si le segment contient un acquittement ou pas
  - **FIN** : **libération** de la connexion
- **RST** : **réinitialisation** de connexion (connexion devenue incohérente)
- **URG** : pointeur d'urgence → le segment contient des données urgentes
- **PSH** : poussée → le récepteur doit immédiatement remonter les données à l 'application

La fenêtre TCP

Huitième champ : Taille de fenêtre 16 bits :

- ce champ indique le nombre d'octets que l'expéditeur est prêt à recevoir.

La gestion de la communication TCP

MTU

Le MTU, ou **unité de transmission maximale (MTU)** représente le plus gros paquet de données qu'un appareil connecté au réseau acceptera.

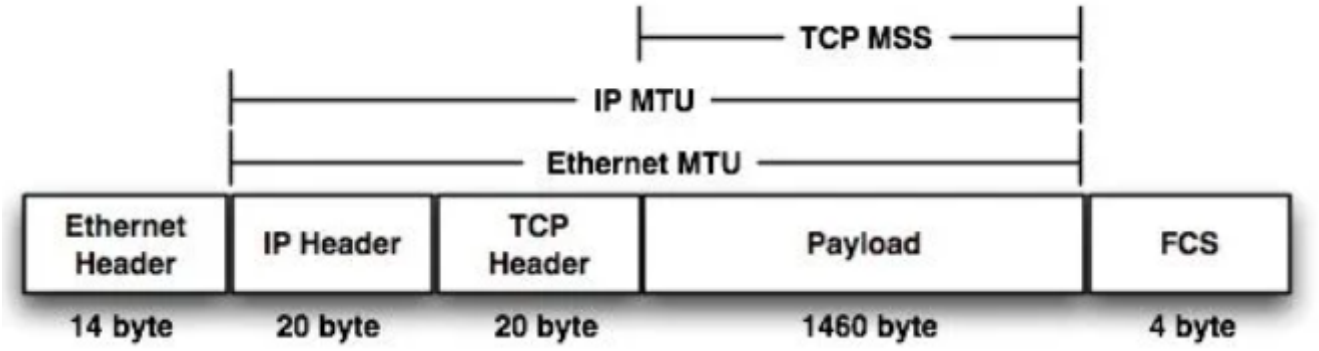
Au dela de cette taile, le paquet est divisé en plusieurs morceaux : **fragmentation du paquet**.

Cela nécessite plus de temps car plus de paquets à transmettre et le destinataire devra rassembler les paquets.

Une valeur optimale de MTU correspond à la valeur maximale que le réseau accepte. S'il est trop bas, cela ajoute de la latence.

MSS

Le **MSS (taille de segment maximale)** limite la taille des segments TCP, hors entête, appelé **charge utile** qu'un périphérique connecté au réseau peut recevoir.



Par défaut, sur un réseau Ethernet chaque trame peut faire jusqu'à 1 500 octets (MTU de 1520 octets), dont :

- au moins 20 octets doivent être réservés à l'en-tête TCP
- et 20 autres à l'en-tête IP.

Cela laisse **1 460 octets** pour les données utiles (MSS).

On utilise la formule suivante pour définir le MSS qui consiste à retirer les en-tête au MTU :

```
MTU - (TCP header + IP header) = MSS
```

Un **tunnel VPN** établi avec **IPsec (Internet Protocol security)** permet de chiffrer les paquets IP mais ajoute un entête spécifique.

La formule devient :

```
MTU - (TCP header + IP header + IPsec) = MSS
```

> Dans un réseau Ethernet dont le **MTU est de 1500**, il n'y aura que des trames d'une longueur maximale de 1 500 octets. Au delà, les paquets plus longs seront **fragmentés**.
>
> Le MSS du routeur doit être défini sur 1 460 octets et les paquets avec une taille de charge utile supérieure à 1 460 octets seront **supprimés**.

MTU discovery (PMTUD)

Lien : https://www.malekal.com/quest-ce-que-le-mtu-et-mss-et-optimiser/

MTU discovery (PMTUD) est une méthode qui permet d'obtenir le MTU de tous les équipements que traversent un paquet :

- on envoie des paquets jusqu'à ce que ce dernier ne soit plus droppé par un des routeurs le long du chemin.
  - Lorsqu'un périphérique le long du chemin abandonne le paquet, il renvoie un message ICMP avec son MTU.
  - Le périphérique source abaisse son MTU et envoie un autre paquet de test.
  - Ce processus est répété jusqu'à ce que les paquets de test soient suffisamment petits pour traverser tout le chemin du réseau sans être abandonnés.

Le site speedguide permet d'obtenir les réglages MTU et MSS d'un PC ainsi que d'autres paramètres TCP/IP.

> Pour définir un volume individuel de MSS, il suffit de le préciser dans le champ **Options** avant la transmission des premières données : l'expéditeur et le destinataire conviennent généralement du volume maximum des segments TCP à expédier : **Maximum Segment Size – MSS**. Cela est fait lors de l'établissment de la connexion TCP (3 way handshake).

Des ajustements supplémentaires doivent alors être faits pour la partie **Données utiles**.

Optimiser le MTU

La commande **ping** permet de déterminer le meilleur **MTU**, en utilisant le même principe que le **path MTU discovery** : envoyer des paquets **fragmentés de différentes tailles** pour voir si un équipement **drop** ou **fragmente** le paquet.

Paramètres de la commande ping :

- envoyer des paquets **non fragmentés** avec l'option -f
- envoyer des paquets avec une taille de MTU différente avec -l

La gestion de la connexion TCP

Une extrémité attend de manière passive l'arrivée d'une connexion :

- **LISTEN** : désigne une source précise
- **ACCEPT** : accepte l'appel d'où qu'il vienne

L'autre extrémité indique la demande de connexion dans un premier segment TCP :

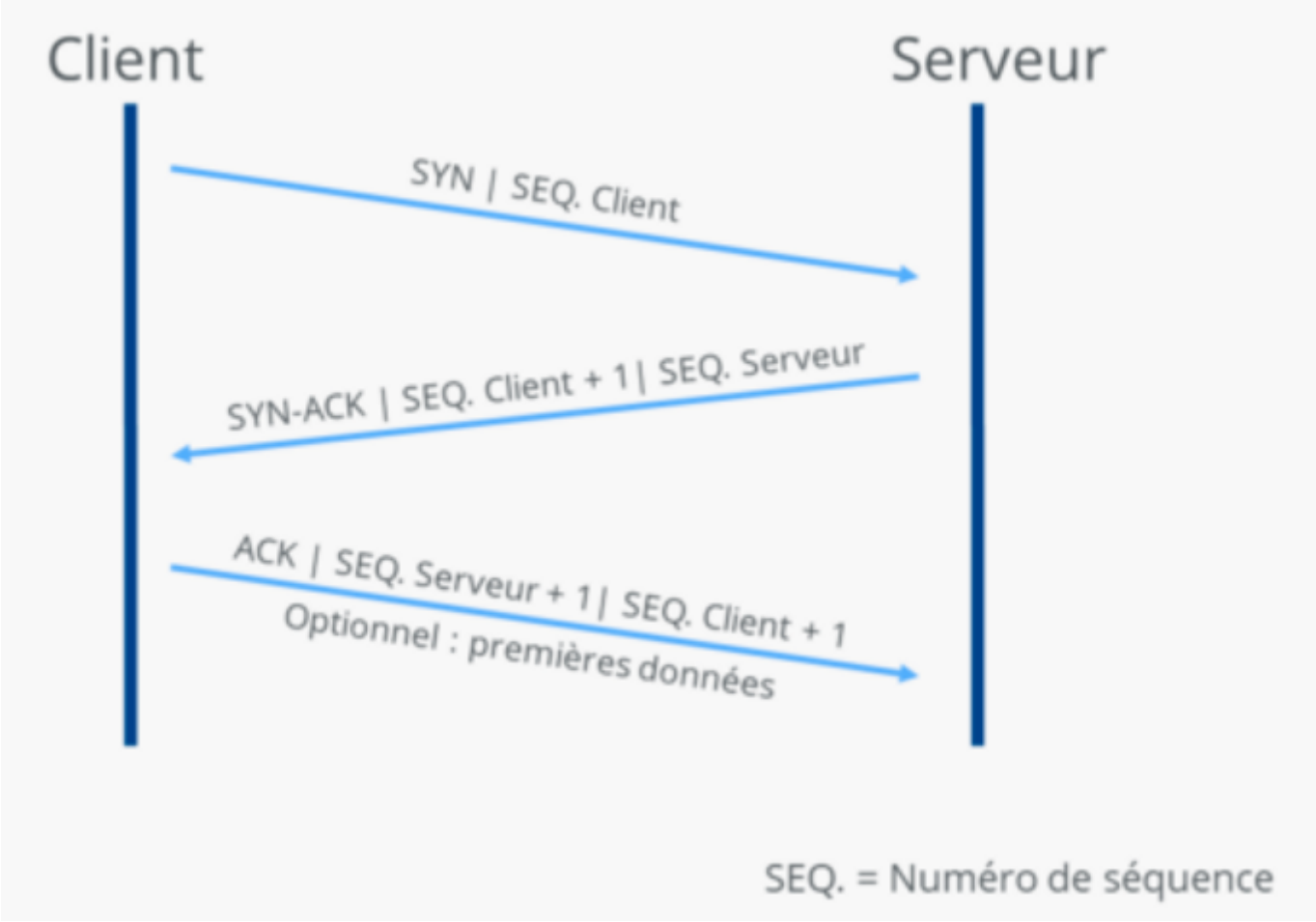
- **CONNECT** : signifie sa volonté de connexion (**SYN=1, ACK=0**)
- Fournis :
  - l'**adresse IP** et le **port** auxquels il désire se connecter ;
  - la **taille maximale des segments (MSS)** qu'il admet ;
  - éventuellement des données (ex. : mot de passe)

Arrivé à destination, recherche d'une application à l 'écoute (LISTEN) sur le Port destination indiqué :

- si **oui**, l'application envoie un acquittement (**SYN=1, ACK=1**) ;
- si non, elle renvoie un rejet (**RST=1**).

Etablissement d'une session

Etablissement en **trois étapes** : le **3-Way Handshake**.



1. le client **envoie** un paquet (segment **SYN** ou synchronize) avec un numéro séquentiel aléatoire individuel.
2. le serveur reçoit le segment et **approuve** la connexion en renvoyant un paquet **SYN-ACK** (acknowledgement = « confirmation »), ainsi que le numéro séquentiel du client augmenté de 1. Il transmet également au client son propre numéro séquentiel.
3. le client **confirme** la réception du segment SYN-ACK en envoyant son propre paquet **ACK** qui contient le numéro **séquentiel du serveur augmenté de 1**. Simultanément, il peut transmettre ses premières données au serveur.