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

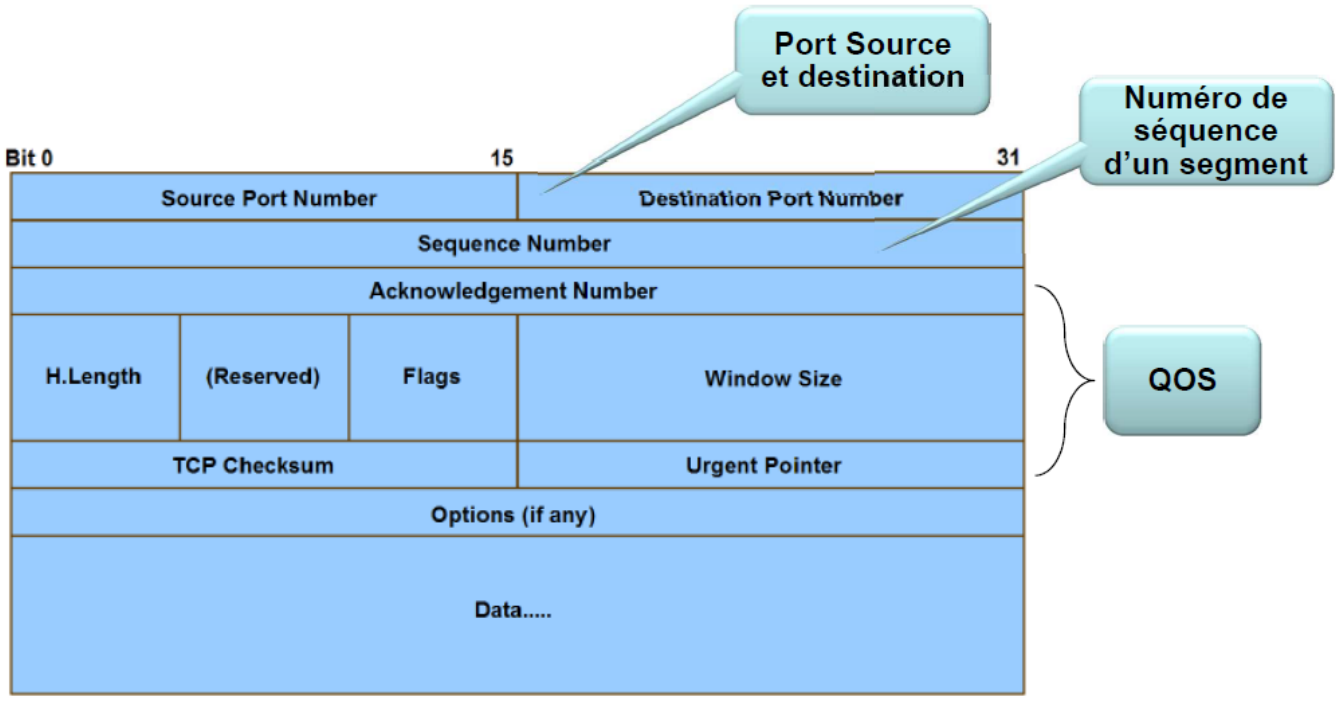
- 0-1023 : **Well-known ports**
- 1024-49151 : **Registered ports**
- 49152-65535 : **Dynamic and/or Private ports**

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

L'entête TCP

TCP transmet sur le réseau des segments constitué :

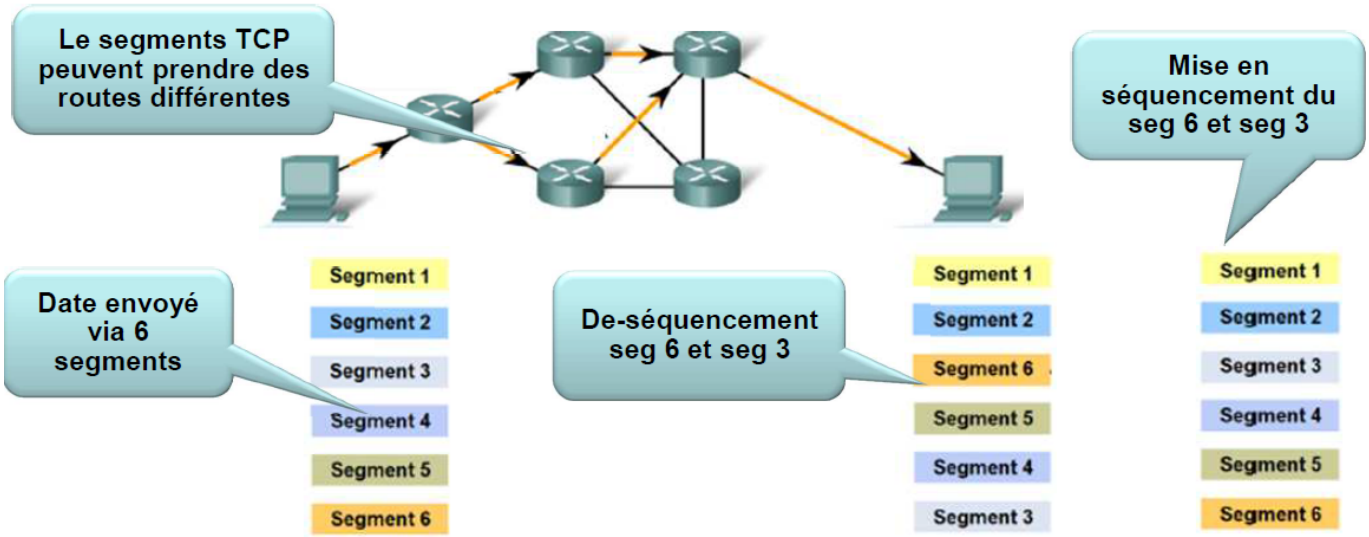
- de 20 octets d'en-tête fixes (plus une partie optionnelle) ;
- 0 ou plusieurs octets de données
- Chaque segment porte un numéro de séquence.



Le numéro de séquence

Troisième champ : Numéro de séquence sur 32 bits

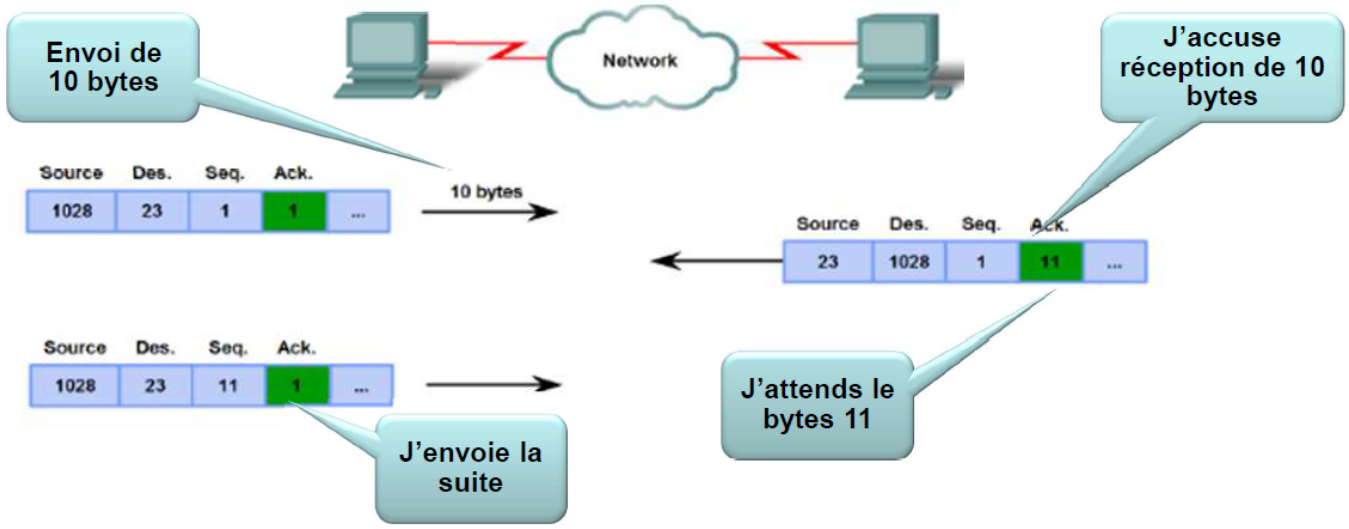
- indique le **numéro** affecté au **premier octet** du segment TCP envoyé par la connexion,
- utilisés pour **reconstruire** le flux de données en replaçant les segments **dans l'ordre**.



Le numéro d'acquitement

Quatrième champ : Numéro d 'acquittement 32 bits

- Numéro du prochain octet attendu par le récepteur (numéro de séquence), dans le flux TCP
- L'annonce de ce numéro indique implicitement que tous les octets précédents (et donc les numéros de séquences précédents ont été bien reçus.



Le champ Flags

Septième champ : 6 drapeaux 6 fois 1 bit

- 3 premiers bits sont utilisés lors de l'**établissement 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 :

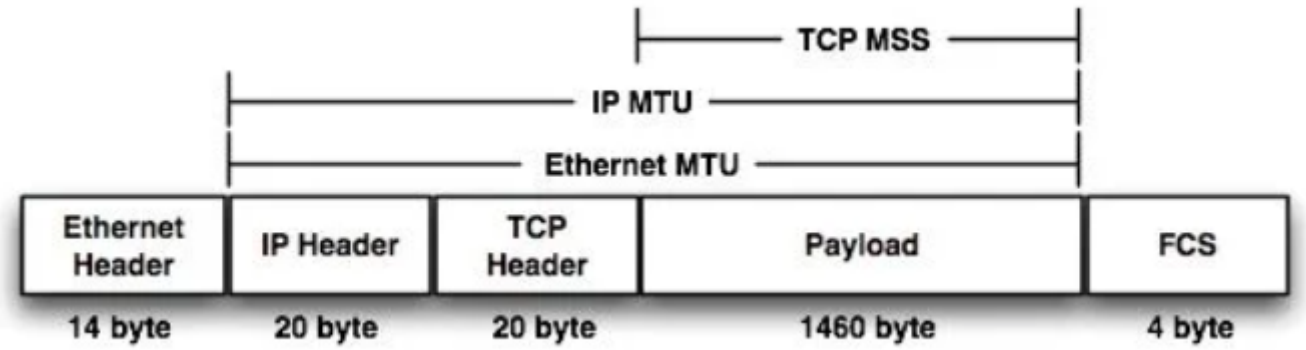
- ce champ indique le nombre d'octets que l'expéditeur est prêt à recevoir.

La gestion de la communication TCP

Lien : https://www.ionos.fr/digitalguide/serveur/know-how/presentation-de-tcp/

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 1500 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**.

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).

Attaque sur l'option MMS

Annoncer des valeurs MMS très petites (**attaque par MMS faible**) pour limiter artificiellement les charges TCP échangées dans les deux sens de la connexion. Le principe est d'obtenir que la taille des paquets IP résultants soit inférieure à une valeur choisie.

On force alors une taille de segments TCP très petite ce qui va multiplier de manière conséquente (jusqu'à supérieur à 14), le nombre de segments TCP à envoyer pour transporter la même quantité de données.

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 :

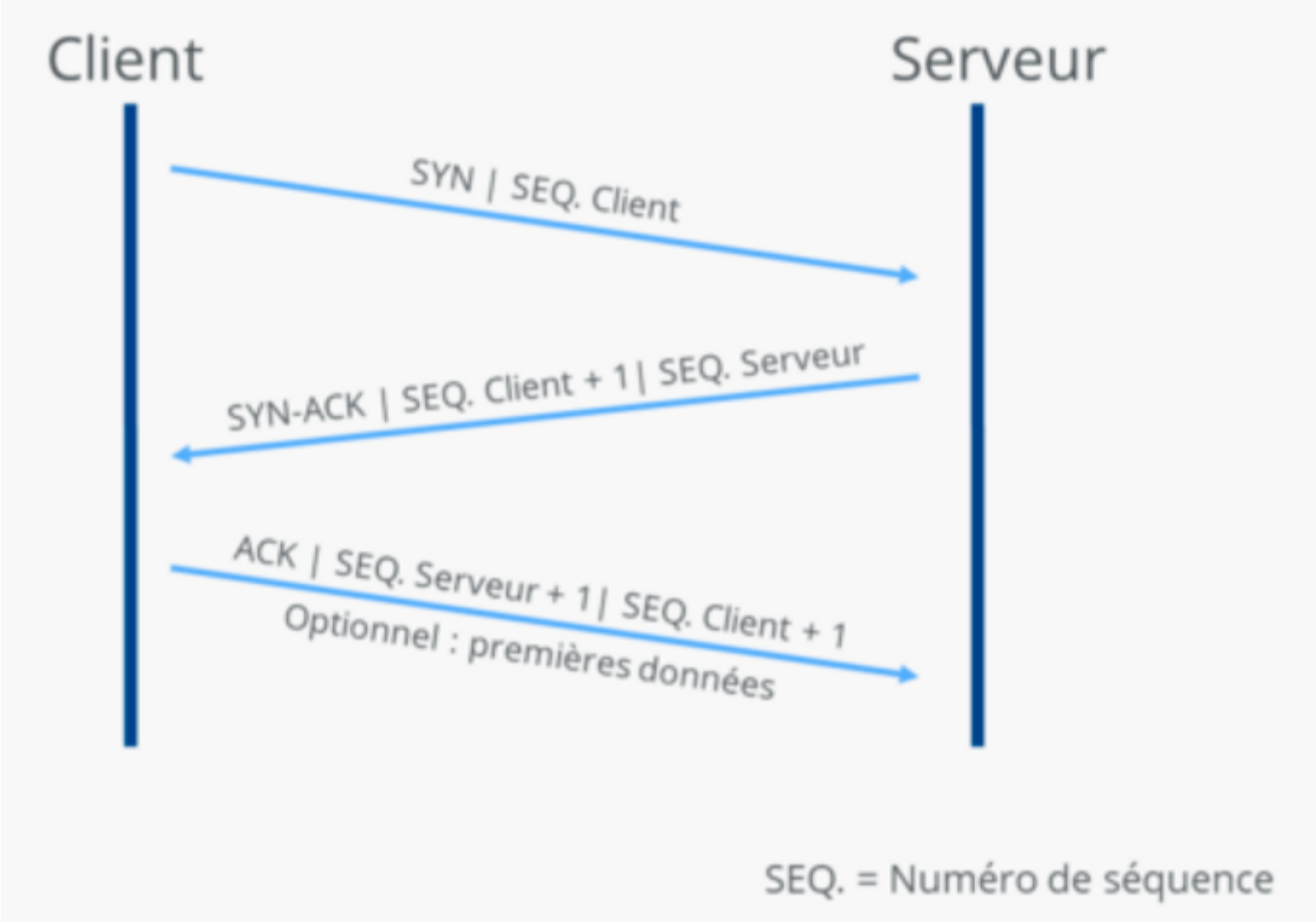
- **CONNECT** : signifie sa volonté de connexion (**SYN=1, ACK=0**)
- Fournis :
  - l'**adresse IP** et le **port** auxquels il désire se connecter ;
  - la **taille maximale des segments (MSS)** qu'il admet ;
  - éventuellement des données (ex. : mot de passe)

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

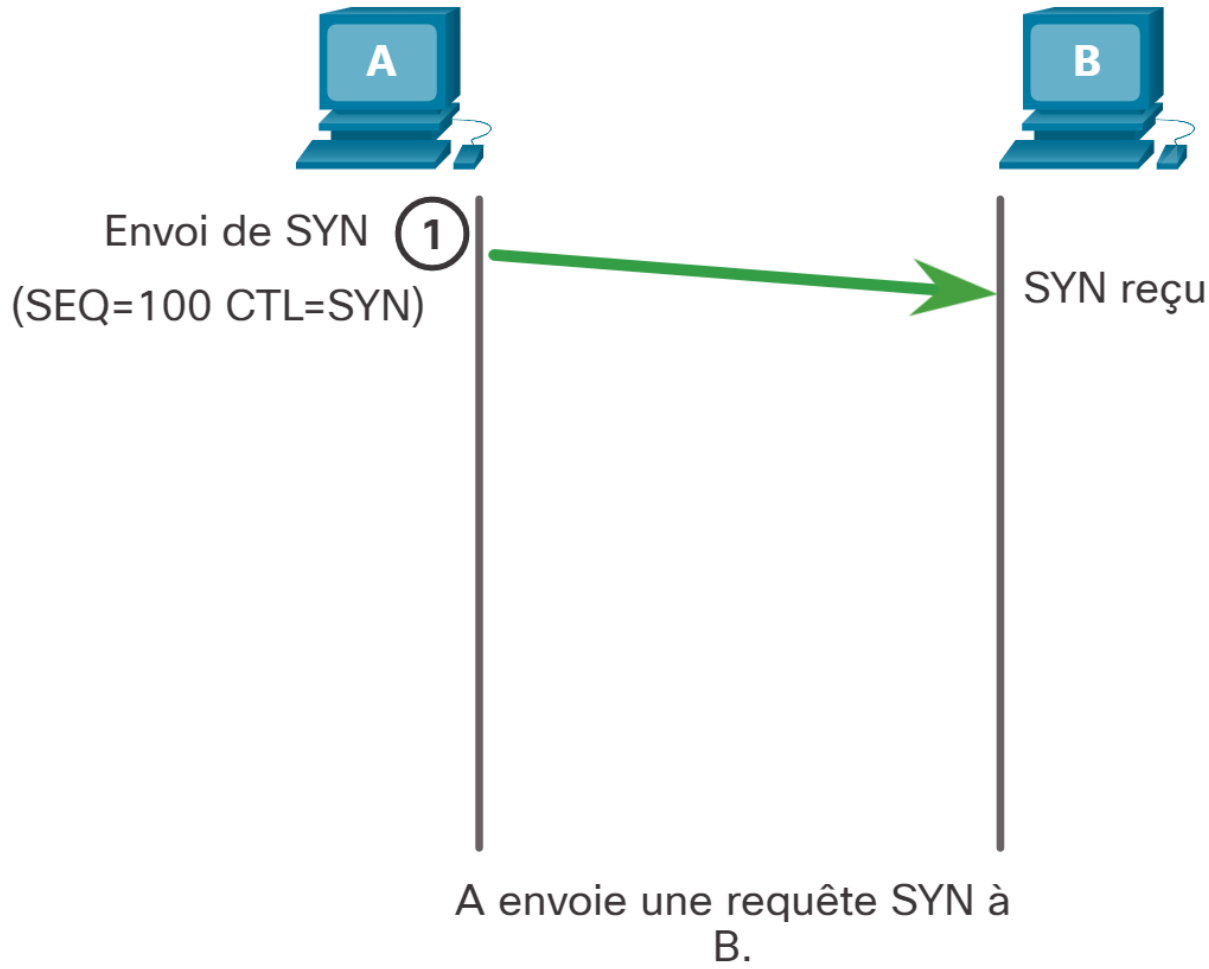
- si **oui**, l'application envoie un acquittement (**SYN=1, ACK=1**) ;
- si non, elle renvoie un rejet (**RST=1**).

Etablissement d'une session

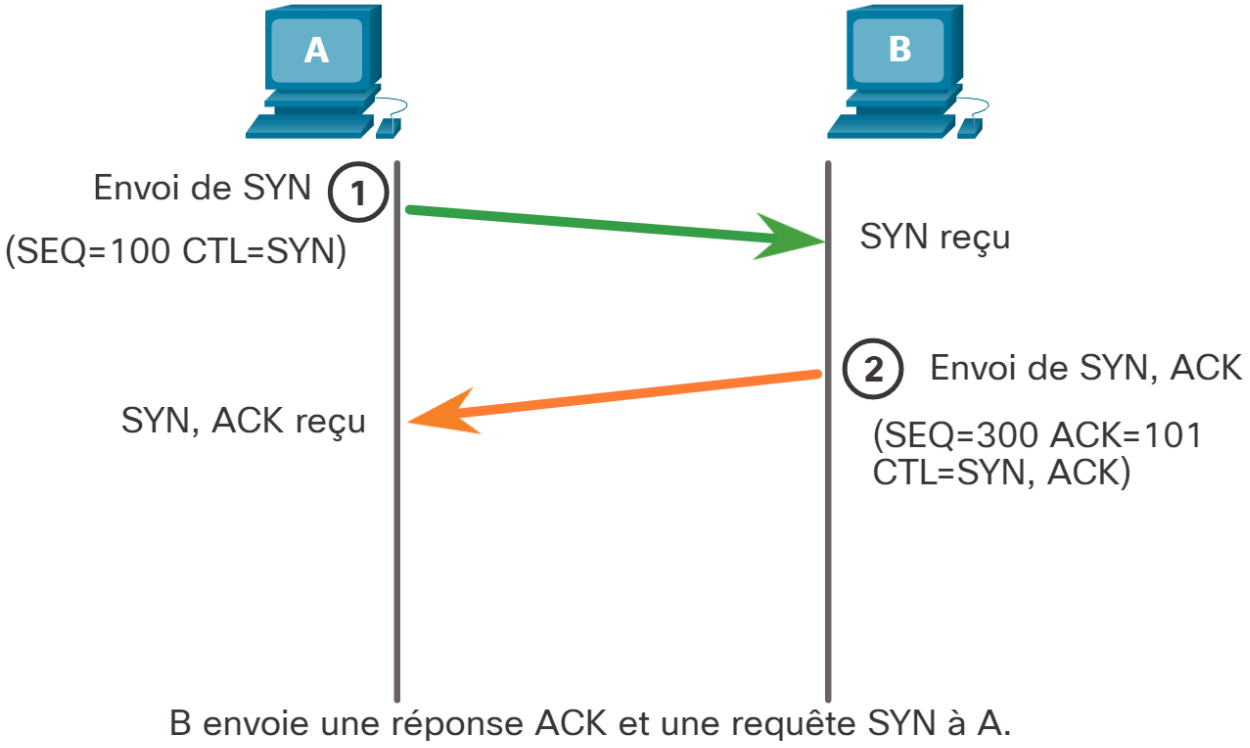
Etablissement en **trois étapes** : le **3-Way Handshake**.



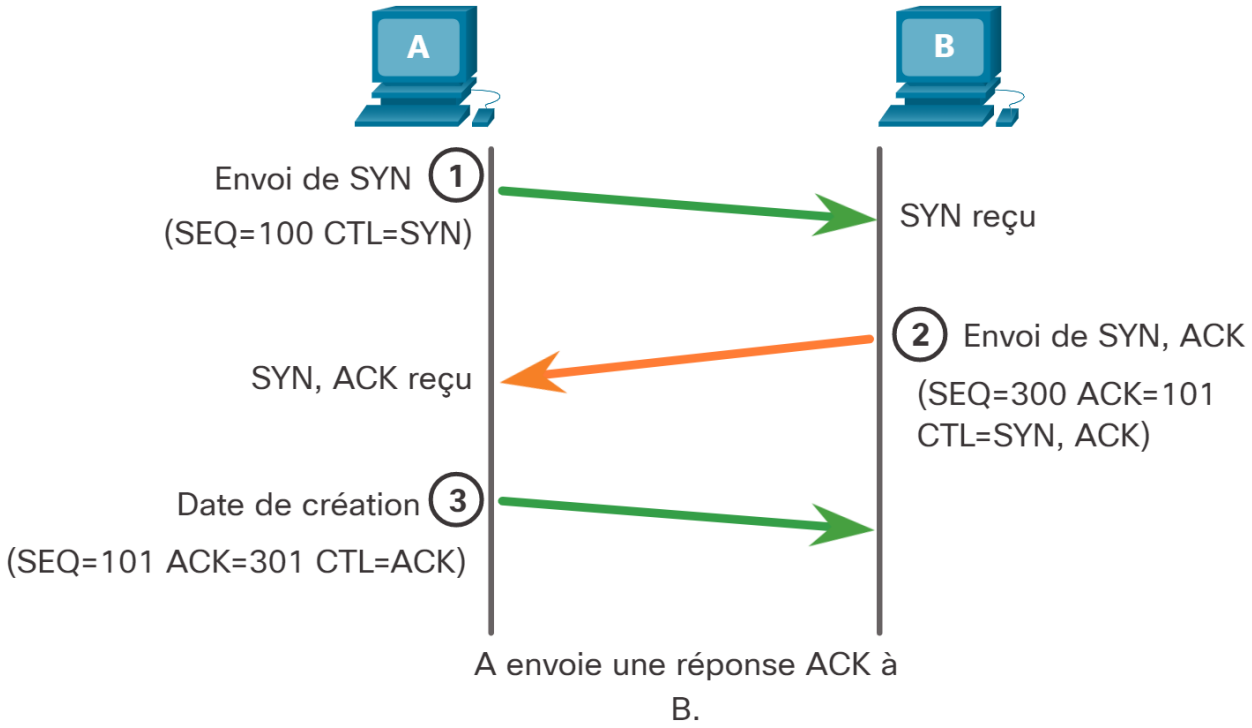
- Etape 1 : Le client demande l'établissement d'une session de communication avec le serveur en **envoyant** un paquet (segment **SYN** ou synchronize) avec un numéro séquentiel aléatoire individuel.



- Etape 2 : le serveur reçoit le segment et accuse réception de la session de communication en **approuvant** la connexion grâce à l'envoi d'une réponse **SYN-ACK** (acknowledgement = « confirmation »), ainsi que le numéro séquentiel du client augmenté de 1. Il transmet également au client son propre numéro séquentiel.



- Etape 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.

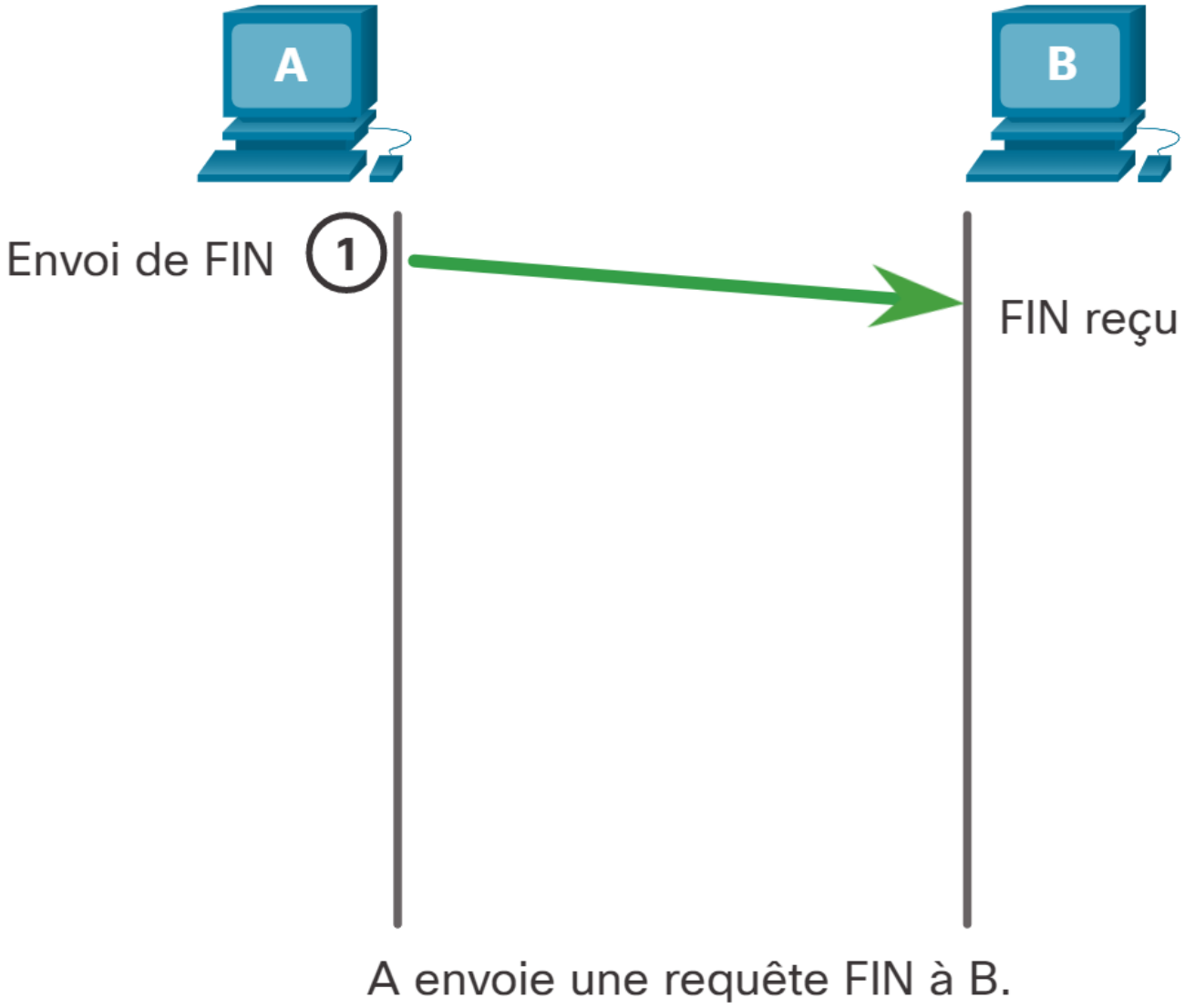


Interruption de session

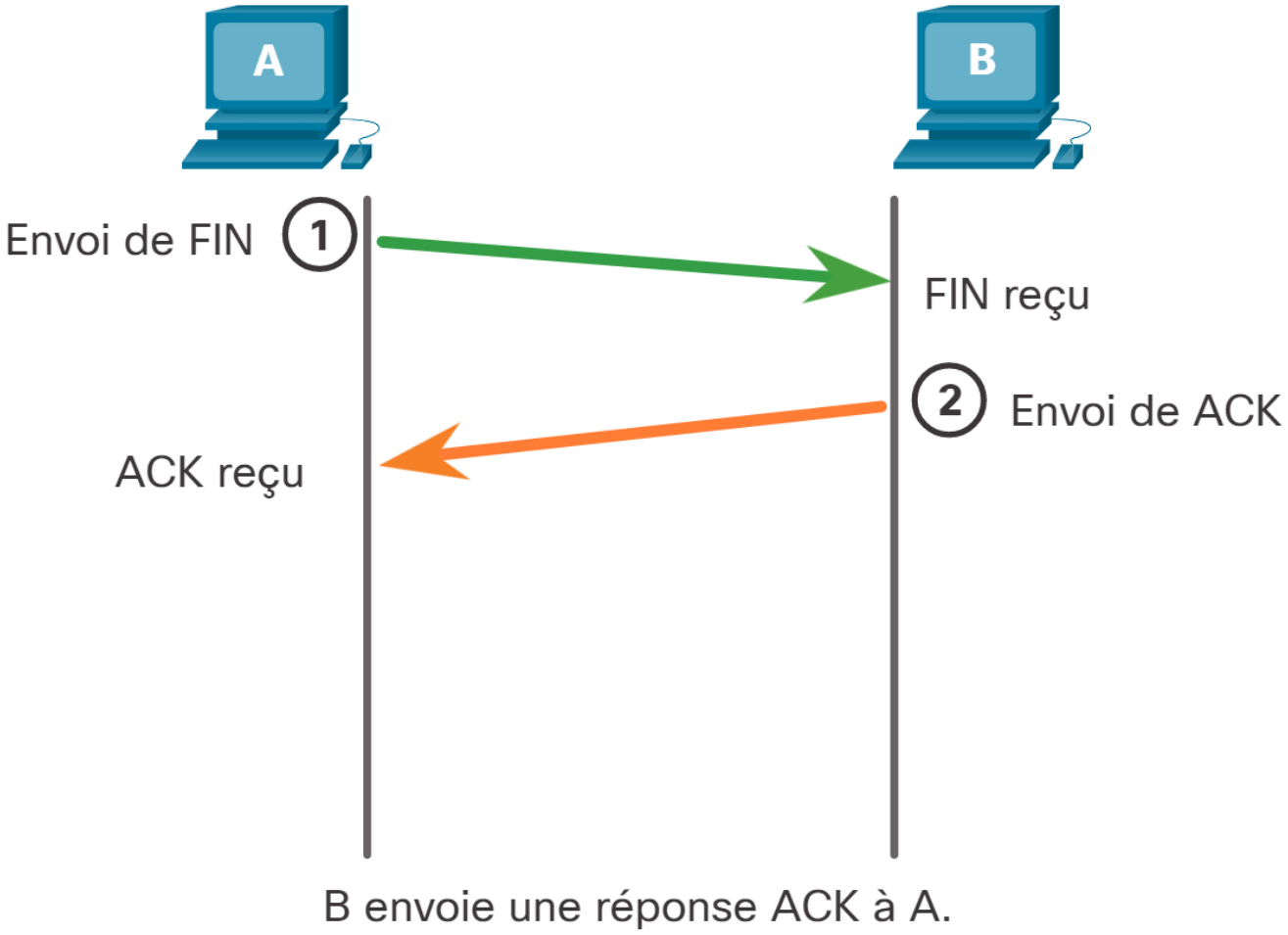
Pour mettre fin à une connexion, l'indicateur de contrôle FIN (Finish) doit être défini dans l'en-tête de segment.

Pour mettre fin à chaque session TCP unidirectionnelle, on utilise un échange en deux étapes constitué d'un segment FIN et d'un segment ACK. Pour mettre fin à une seule conversation TCP, quatre échanges sont nécessaires pour mettre fin aux deux sessions. La terminaison peut être initiée par le client ou le serveur. Dans l'exemple, les termes client et serveur sont utilisés comme référence pour plus de simplicité, mais deux hôtes quelconques qui ont une session ouverte peuvent lancer le processus de terminaison.

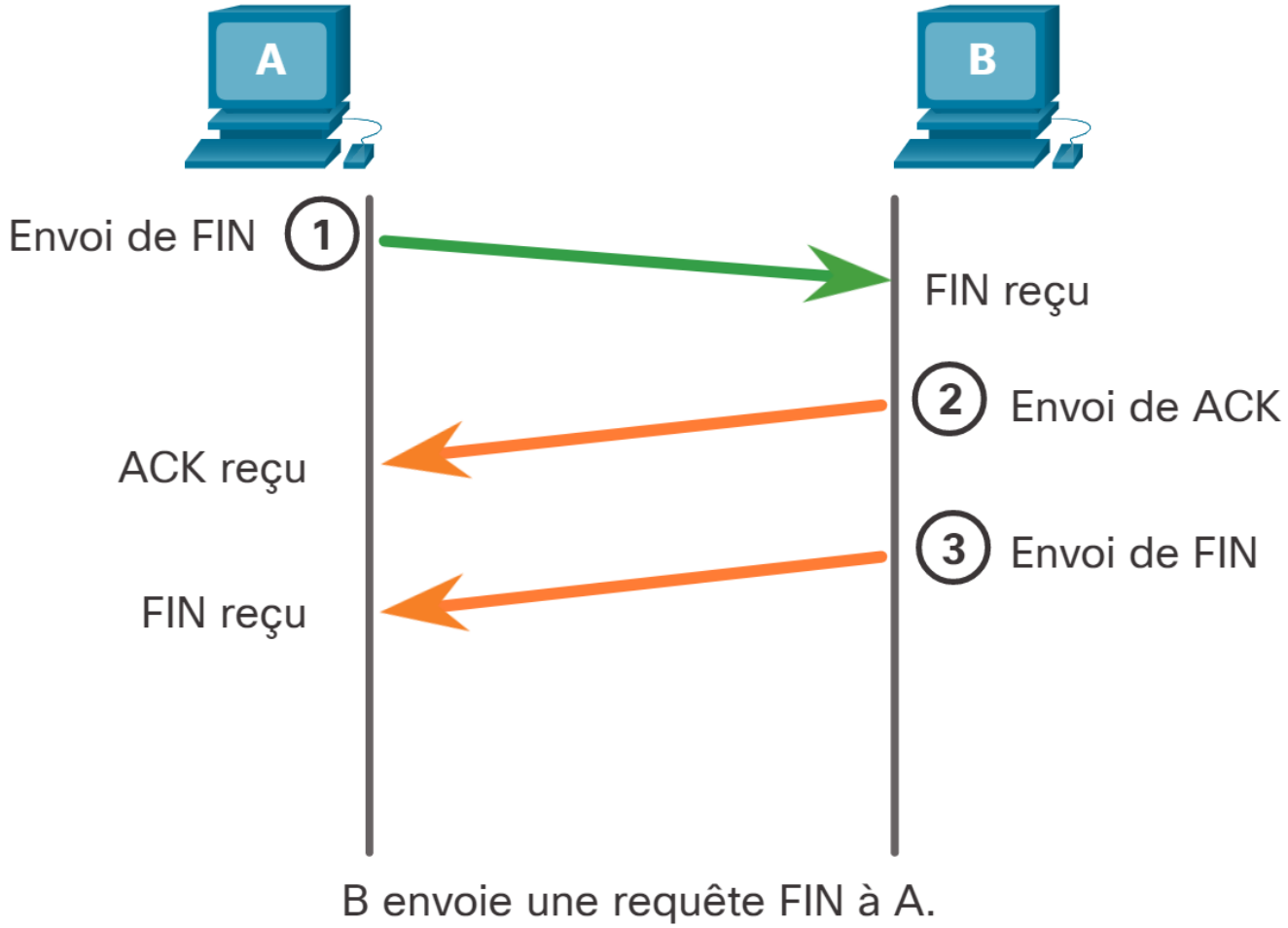
- Étape 1. FIN : quand le client n'a plus de données à envoyer dans le flux, il envoie un segment dont l'indicateur FIN est défini.



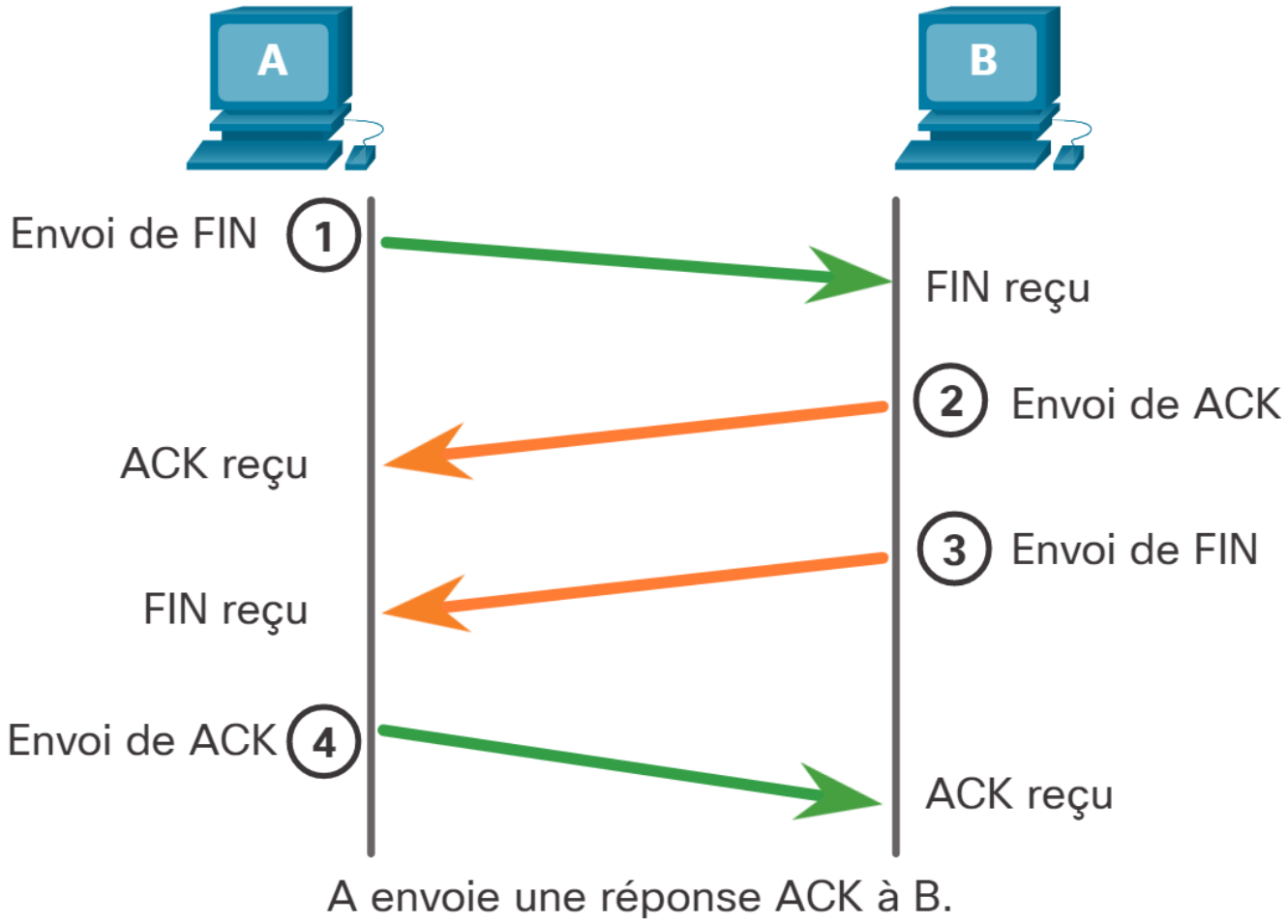
- Étape 2. ACK : Le serveur envoie un segment ACK pour informer de la bonne réception du segment FIN afin de fermer la session du client au serveur.



- Étape 3. FIN : Le serveur envoie un segment FIN au client pour mettre fin à la session du serveur au client.



- Étape 4. ACK : Le client répond à l'aide d'un segment ACK pour accuser réception du segment FIN envoyé par le serveur.



Quand la réception de tous les segments a été confirmée, la session est fermée.

Le contrôle de la connexion TCP

Les hôtes maintiennent l'état, suivent chaque segment de données au cours d'une session et échangent des informations sur les données reçues en utilisant les informations de l'en-tête TCP.

TCP est un protocole full-duplex et chaque connexion représente deux sessions de communication à sens unique.

Les six bits de contrôle du champ des bits de contrôle dans l'en-tête TCP indiquent la progression et l'état de la connexion. Ce sont également des indicateurs (bit qui est actif ou inactif). Les six indicateurs de bits de contrôle sont les suivants:

- **URG** - Champ de pointeur urgent significatif (Urgent pointer field significant)
- **ACK** - Indicateur d'accusé de réception utilisé dans l'établissement de la connexion et la fin de la session
- **PSH** - Fonction push (Push function)
- **RST** - Réinitialisation de la connexion en cas d'erreur ou de dépassement de délai
- **SYN** - Synchroniser les numéros de séquence utilisés dans l'établissement de connexion
- **FIN** - Plus de données de l'expéditeur et utilisées dans la fin de session

Attaques possibles sur le séquencement

- Prédiction du système d'exploitation pour des OS qui utilisent toujours la même valeur initiale lors de l'établissement d'une session
- Vol de session TCP (session hijacking) : en connaissant de numéro de séquence initial, l'attaquant peut prédire les numéros de séquence des prochaines connexions, information utile pour forger des paquets destinés à voler des données, clôturer une connexion ou placer des données illégitimes.