UDP, TCP et la couche applicative (en 2 heures)

FIPA24 - Année scolaire 22/23

Pascal Cotret, ENSTA Bretagne

14 octobre 2022

UDP

▲ *User Datagram Protocol* - UDP I

Caractéristiques d'UDP

- **Non connecté** : on envoie en espérant que quelqu'un écoute…
- **Non fiable** : pas de garantie de remise, protocole en *best effort* ne détectant pas les pertes sur le canal ;
- **Sans garantie d'ordonnancement** : des datagrammes routés différemments peuvent arriver dans le désordre ;
- **Sans contrôle de flux ni contrôle de congestion** : l'expéditeur émet à son débit maximum, reçoive qui peut !

▲ *User Datagram Protocol* - UDP II

Socket

En l'absence de mode connecté, un *socket* UDP est uniquement défini par l'**adresse IP** et le **numéro de port** locaux.

▲ *User Datagram Protocol* - UDP II

Socket

En l'absence de mode connecté, un *socket* UDP est uniquement défini par l'**adresse IP** et le **numéro de port** locaux.

Problème

Mais alors, quel intérêt ? Ce protocole n'a que des défauts !

User Datagram Protocol - UDP III

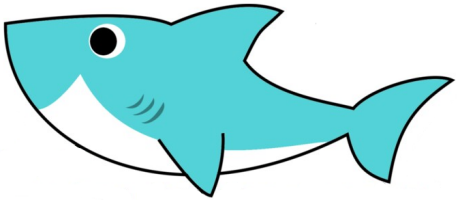
UDP est un protocole **simple**, induisant peu de surcharge protocolaire (*overhead*) et peu de délais supplémentaires.

Ses caractéristiques le rendent très adapté pour le **multimédia**, et notamment le ***streaming*** (en particulier en multidiffusion) et d'une manière générale les flux continus, rapides et résilients aux pertes.

Certains protocoles très légers (DNS, SNMP…) en profitent également.

Il est également adapté lorsque la notion de fiabilité est plus efficacement définie au niveau application qu'au niveau transport (contrôles spécifiques à une application).

Format d'en-tête des datagrammes UDP



TCP

1. UDP

2. TCP
   - Propriétés de TCP
   - Format des segments TCP
   - Connexion et déconnexion

▲ *Transmission Control Protocol* – TCP

Caractéristiques de TCP

- **Connecté** : établissement, maintien, fermeture de connexions point-à-point ;
- **Fiable** : réémission en cas de pertes ;
- **Ordre garanti** : les segments sont numérotés et réordonnés à la réception ;
- **Contrôle de flux** : l'émetteur adapte son débit aux capacités de traitement du récepteur ;
- **Contrôle de congestion** : l'émetteur adapte son débit à l'encombrement du réseau ;

Socket TCP

Un *socket* TCP est *full-duplex* et défini par un quadruplet **(IP locale, port local, IP distante, port distant)**.

▲ *Transmission Control Protocol – TCP*

Définition

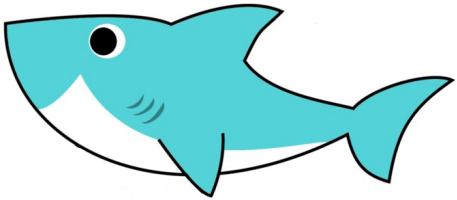
TCP est conçu pour **pallier les défaillances d'un canal de communication non sûr**, ce qui est l'une des problématiques majeures de l'ingénierie du réseau et des télécommunications.

Le canal peut avoir des pertes, générer des erreurs, avoir une bande passante variable : **TCP s'adapte et garantit que les données seront transmises correctement** (sauf rupture réelle de connectivité).

Et donc ?

TCP constitue une abstraction fiable des communications réseau, utilisée pour le développement d'applications connectées.

Format d'en-tête des segments TCP I



Format d'en-tête des segments TCP II

Paramètres de l'en-tête

- ***Source port* :** port TCP utilisé par l'émetteur ;
- ***Destination port* :** port TCP utilisé par le destinataire ;
- ***Sequence number* :** numéro de séquence des données émises, en octets, permettant le réordonnancement et la détection de pertes ;
- ***Acknowledgement number* :** prochain numéro de séquence attendu pour les données en réception ;
- ***Data offset* :** taille des en-têtes en mots de 32 bits (variable à cause des options) ;

Format d'en-tête des segments TCP III

Paramètres de l'en-tête

- *Reserved* : non utilisé, réservé pour un usage ultérieur depuis 1981…
- **Drapeaux / *flags* :** voir transparent suivant ;
- ***Window size* :** taille utilisable de la fenêtre de réception, en octets (contrôle de flux) ;
- ***Checksum* :** code de détection d'erreur sur les en-têtes, les données et un pseudo-en-tête IP ;
- *Urgent pointer* : désigne le dernier octet de données urgentes du segment.

Format d'en-tête des segments TCP IV

Les drapeaux (*flags*) TCP

- *ECN-nonce (NS)* : expérimental, protocole ECN de gestion de congestion ;
- *Congestion Window Reduced (CWR)* : utilisé par le protocole ECN de gestion de congestion ;
- *ECN-Echo* : protocole ECN de gestion de congestion ;
- *Urgent (URG)* : indique l'existence de données urgentes ;
- ***Acknowledge (ACK)* :** indique que le champ *Acknowledgement number* est pertinent. Toujours à 1 après la connexion ;

Format d'en-tête des segments TCP V

Les drapeaux (*flags*) TCP

- *Push (PSH)* : demande de pousser les données du buffer de réception vers l'application destinataire ;
- ***Reset (RST)* :** réinitialise (termine brutalement) la connexion ;
- ***Synchronize (SYN)* :** synchronisation des numéros de séquence, utilisé pour l'établissement de la connexion ;
- ***Finalize (FIN)* :** utilisé pour clore la connexion.

Établissement d'une connexion TCP I

OK, je vois à peu près comment ça se passe une fois qu'on est connecté, mais… Ca se passe comment justement au démarrage ?

Établissement d'une connexion TCP I

OK, je vois à peu près comment ça se passe une fois qu'on est connecté, mais… Ca se passe comment justement au démarrage ?

Le *three-way handshake*

En TCP, il y a toujours, au début, un **serveur**, qui attend des connexions sur un port donné, et un **client**, qui souhaite se connecter.

Il faut arriver à un état où **les deux parties sont d'accord sur le fait d'être connectées** : il ne faut pas en arriver à des connexions à moitié ouvertes où l'une des deux parties pense être connectée et l'autre non.

▲ Établissement d'une connexion TCP II

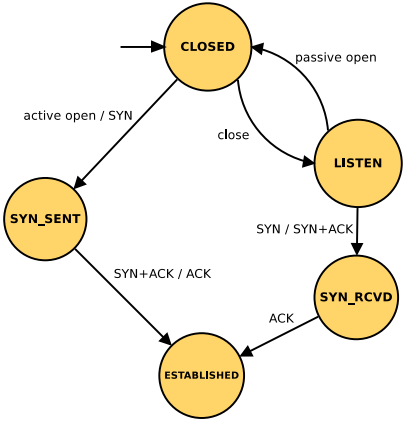
OK, je vois à peu près comment ça se passe une fois qu'on est connecté, mais… ça se passe comment justement au démarrage ?

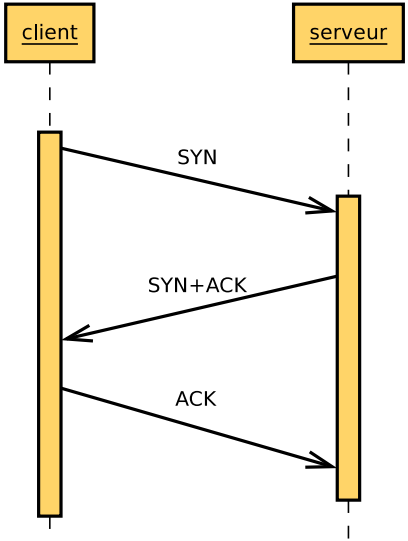
Le *three-way handshake*

Si l'on considère que le canal est non sûr et avec délais, un simple échange ne suffit pas, il faut **trois paquets** :

1. Le client envoie un paquet `SYN` ;
2. Le serveur répond avec un paquet `SYN+ACK` ;
3. Le client confirme avec un `ACK`.

Établissement d'une connexion TCP III





Établissement d'une connexion TCP IV

Question

Que signifie `SYN` ? On se "synchronise" ? Sur quoi ?

Établissement d'une connexion TCP IV

Question

Que signifie `SYN` ? On se "synchronise" ? Sur quoi ?

Réponse

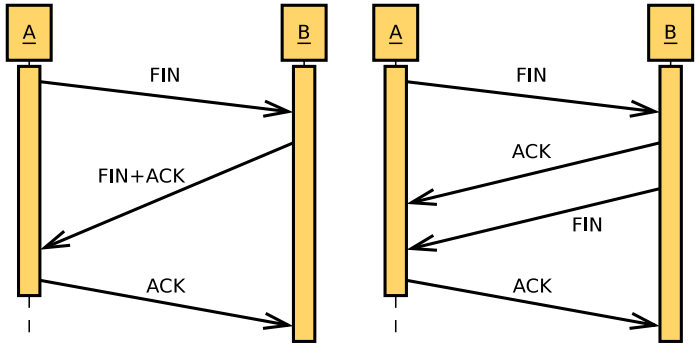
Client et serveur profitent de cette phase pour s'accorder sur plusieurs choses :

- Un **numéro de séquence initial**, aléatoire, dans chacun des sens de communication, pour éviter les collisions entre deux connexions proches dans le temps et les attaques en prédiction de numéro de séquence *(on ne commence jamais à seq=0)* ;
- Un ***Maximum Segment Size*** (MSS), correspondant au minimum des MTU des deux parties (utilisé pour éviter la fragmentation IPv4) ;
- Une valeur initiale pour chacune des **fenêtres de réception**.

▲ Déconnexion

Une déconnexion TCP “normale”

Chaque partie envoie un `FIN`, qui chacun reçoit un `ACK` (croisements et combinaisons possibles).



La connexion peut également prendre fin sur *timeout* ou sur réception d’un `RST`.

Références

[1] Cookie Connecté. *Comprendre le DHCP en 3 minutes*. https://www.youtube.com/watch?v=yH9UvkeAz-I. 2020.

[2] Cookie Connecté. *Comprendre les modèles OSI et TCP/IP*. https://www.youtube.com/watch?v=26jazyc7VNk. 2020.

[3] Wikipedia. *IPsec*. https://fr.wikipedia.org/wiki/IPsec. 2021.

[4] Wireshark. *Sample captures*. https://gitlab.com/wireshark/wireshark/-/wikis/SampleCaptures. 2020.