**LE PROTOCOLE TCP**

**Tcp : transport control Protocol**

But : transport fiable de la technologie tcp/ip.

* fiabilité = illusion assurée par le service
* transfert tamponnés : découpage en segment
* connexions bidirectionnelles et simultanées
* service en mode connecté
* garantie de non perte de messages ainsi que de l'ordonnancement

**La connexion**

* une connexion de type circuit virtuel est établie avant que les données ne soient échangées : appel + négociation + transferts
* une connexion = une paire d'extrémités de connexion
* une extrémité de connexion = couple (adresse IP, port)
* exemple de connexion : ((124.32.12.1 , 1034),(19.24.67.2,21))
* une extrémité de connexion peut être partagée par plusieurs autres extrémités de connexions ( multi-instanciation)
* la mise en œuvre de la connexion se fait en deux étapes :
  + une application (extrémité) effectue une ouverture passive en indiquant qu'elle accepte une connexion entrante
  + une autre application (extrémité) effectue une ouverture active pour demander l'établissement de la connexion

**La segmentation**

* segmentation, contrôle de flux
  + les données transmises à TCP (tout paquet doit être acquitté) constituent un flot d'octets de longueur variable
  + TCP divise ce flot de données en segments en utilisant un mécanisme de fenêtrage
  + Un segment est émis dans un datagramme IP
* Acquittement de message
  + Contrairement u UDP, TCP garantit l'arrivée des messages, c’est-à-dire qu'en cas de perte, les deux extrémités sont prévues.
  + Ce concept repose sur les techniques d'acquittement de messages : lorsqu'une source S émet un message Mi vers une destination D, S attend un acquittement Ai de D avant d'émettre le message suivant Mi+1.
  + Si l'acquittement Ai ne parvient pas à S, S considère au bout d'un certain temps que le message est perdu et réémet Mi :

**Acquittement**

(Schéma 1)

**Le fenêtrage**

* La technique acquittement simple pénalise les performances puisqu'il faut attendre un acquittement avant d'émettre un nouveau message. Le fenêtrage améliore le rendement des réseaux
* La technique du fenêtrage : une fenêtre de taille T, permet l'émission d'au plus T messages "non acquittés" avant de ne plus pouvoir émettre :

(Schema2)

**Technique de fenêtrage**

* Fenêtrage glissante permettant d'optimiser la bande passante
* Permet également au destinataire de faire diminuer le débit de l'émetteur donc de gérer le contrôle de flux
* Le mécanisme de fenêtrage mis en œuvre dans TCP opère au niveau de l'octet et non pas au niveau du segment ; il repose sur :
  + La numérotation séquentielle des octets de données,
  + La gestion de 3 pointeurs par fenêtrage :

(Schéma 3)

**Segments**

* Segment : unité de transfert du protocole TCP
  + Echangés pour établir les connexions
  + Transférer les données
  + Emettre des acquittements
  + Fermer les connexions

(Schéma 4)

**Format du segment**

* Numéro de séquence : le numéro de séquence du premier octet (NSP) de ce segment. Généralement à la suite d'octets 01,02,…, 0n (données du message) est associé la suite de numéro de séquence NSP, NSP+1,…, NCP+n

il existe 2 exceptions à cette regle :

* + Lorsque le bit SYN(voir CODE BITS) est positionné, le NSP représente cette donnée de contrôle et par conséquent la suite NSP, NSP+1,…, NCP+n, NSP+n+1, NSP+n+2, associe la suite de données SYN 01,02,…,0n.
  + Lorsque le bit FIN (voir CODE BITS) est positionné, le NSP+n représente cette donnée de contrôle et par conséquent la suite NSP , NSP+1, NSP+2,…, NSP+n, associe la suite de données 01,02,…,0n,FIN.
* Numéro d'acquittement : le prochain numéro de séquence NS attendu par l'émetteur de cet acquittement. Acquitte implicitement les octets NS-1, NS-2, etc.
* Fenêtre : la quantité de données que l'émetteur de ce segment est capable de recevoir;(…)

**Format du segment**

* CODE BITS (URG,ACK, PSH, RST, SYN, FIN): indique la nature du segment
  + URG: le pointeur de données urgente est valide(exemple : interrupt en remote login), les données sont émises sans délai, les données reçues sont remises sans délai.
  + SYN: utilité à l'initialisation de la connexion pour indiquer où la numérotation séquentielle commence. SYN occupe lui-même un numéro de séquence bien que ne figurant pas dans le champ de données. Le Numéro de séquence inscrit dans le datagramme (correspondant à SYN) est alors un *Initial Sequence Number* (ISN) produit par un générateur garantissant l'unicité de l'ISN sur le réseau (indispensable pour identifier les duplications).
  + FIN : utilisé lors de la libération de la connexion ;
  + PSH: fonction push. Normalement, en émission, TCP reçoit les données depuis l'applicatif, les transforme en segments a sa guise puis transfère les segments sur le réseau ; un récepteur TCP décodant le bit PSH, transmet à l'application réceptrice, les données correspondantes sans attendre plus de données de l'émetteur. Exemple : émulation terminale, pour envoyer chaque caractère entré au clavier (mode caractère asynchrone)
  + RST : utilisé par une extrémité pour indiquer à l'autre extrémité qu'elle doit réinitialiser la connexion. Ceci est utilisé lorsque les extrémités sont désynchronisées. Exemple : (Schéma 5)

**Tcp format du segment**

* CHECKSUM : calcul du champ de contrôle : utilise un pseudo-en-tete et s'applique a la totalité du segment obtenu (PROTO -6) :

Schéma 6

**Format du header**

OPTION :

* Permet de négocier la taille maximale des segments échangés. Cette option n'est présente que dans les segments d'initialisation de connexion (avec bit SYN)
* TCP calcule une taille maximale de segment de manière a ce que le datagramme IP résultant corresponde au MTU du réseau. La recommandation est de 536 octets.
* La taille optimale du segment correspond au cas où le datagramme IP n'est pas fragmenté mais :
  + Il n'existe pas de mécanisme pour connaitre le MTU
  + Le routage peut entrainer des variations de MTU
  + La taille optimale dépend de la taille des en-têtes (options)

**Acquittements**

Acquittements et retransmissions :

* Le mécanisme de TCP est cumulatif :
  + Il indique le numéro de séquence du prochain octet attendu : tous les octets précédents cumulés sont implicitement acquittés
  + Si un segment a un numéro de séquence supérieur au numéro de séquence attendu (bien que dans la fenêtre), le segment est conservé mais l'acquittement référence toujours le numéro de séquence attendu (🡪)
* Pour tout segment émis, TCP s'attend à recevoir un acquittement
  + Si le segment n'est pas acquitté, le segment est considéré comme perdu et TCP le transmet
  + Or un réseau d'interconnexion offre des temps de transit variable nécessitant le réglage des temporisations
  + TCP gère des temporisations variables pour chaque connexion en utilisant un algorithme de retransmission adaptative

Schema 7

**CM 6**