

Couche "Transport" : TCP et UDP

Introduction

La couche "réseau" s'occupe de router les paquets :

- ▶ Il n'y a aucune garantie qu'un paquet arrive...
 - ▶ Même si des messages de contrôle sont prévu (ICMP), il n'est pas garanti qu'on reçoive une erreur si un paquet n'arrive pas
- La couche "transport" :
- ▶ s'occupe de rajouter de la fiabilité
 - ▶ contrôle que tous les paquets arrivent
 - ▶ si un paquet n'arrive pas, elle le renvoie automatiquement
 - ▶ contrôle que les paquets arrivent dans l'ordre
 - ▶ sinon, elle les remet dans l'ordre avant de passer à la couche suivante ("Application")
 - ▶ est l'interface qu'on va utiliser dans les applications.
 - ▶ c'est la dernière couche dans le "système"
 - ▶ c'est celle qu'on va appeler dans nos programmes

Protocoles : sur Internet, principalement TCP et UDP

Interface avec les applications

Une application (et vous) :

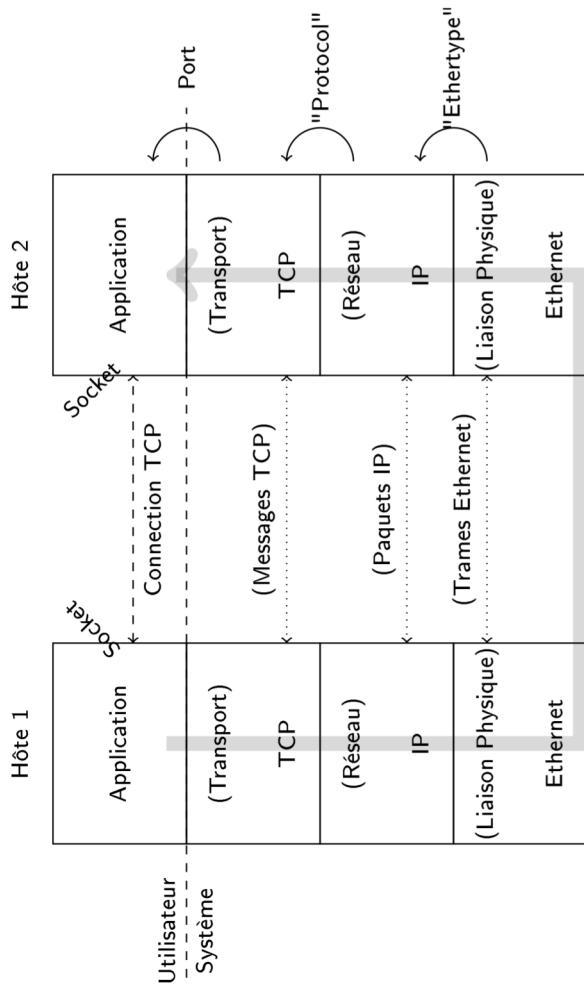
- ▶ veut avoir une interface standard pour accéder au réseau
- ▶ ne veut pas (en général) avoir à gérer les vérifications (contrôle de somme, qu'un paquet n'a pas été perdu...)

La couche transport s'occupe de cela.

Le système propose des sockets pour les connexions réseaux :

- ▶ identifiées par des descripteurs de fichiers
- ▶ elles fonctionnent comme des tubes
- ▶ bi-directionnelles

Côté réseau, les sockets sont identifiées par leur numéro de port



Ports

TCP et UDP rajoutent des numéros de port (un entier entre 1 et 65535).

Ces numéro servent à identifier les applications de la couche supérieure ("Application")

- ▶ L'application demande à la couche TCP/UDP d'ouvrir un port (via une socket)
- ▶ Quand un segment TCP arrive sur la machine (par la couche "réseau"), la couche TCP/UDP regarde le numéro de port
- ▶ Si c'est un numéro associé, le segment sera transmis à l'application correspondante (via la socket)
- ▶ Sinon, la couche renvoie un "message" d'erreur

Chaque type de service à un port normalement assigné. HTTP : 80, SSH : 22... (voir /etc/services)

Différence TCP/UDP

Sur Internet : TCP et UDP

UDP (User Datagram Protocol)

- ▶ non connecté
- ▶ somme de contrôle
- ▶ pas de garantie :
- ▶ le paquet peut ne pas arriver, ou arriver en double
- ▶ les paquets peuvent être intervertis
- ▶ (Proche des garanties de la couche réseau)

PDU : "Datagrammes UDP"

Différence TCP/UDP

TCP (Transmission Control Protocol)

- ▶ connecté
- ▶ somme de contrôle
- ▶ la couche TCP s'occupe que les paquets arrivent, et arrivent dans l'ordre :
 - ▶ si un paquet n'arrive pas, ou arrive erroné (mauvaise somme de contrôle), elle se charge elle même de renvoyer le paquet
 - ▶ si des paquets sont intervertis, elle les remet dans le bon ordre avant de les délivrer à la couche supérieure

PDU : "Segments TCP"

UDP (User Datagram Protocol)

| Entête : | | | | | | | | | | | | | | | | |
|----------|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|-------------|
| bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| bytes | ↑ | | | | | | | | | | | | | | | source port |
| | | | | | | | | | | | | | | | | length |
| ↓ | | | | | | | | | | | | | | | | checksum |
| bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

Notes (idem pour TCP) :

- ▶ Il y a un port destination, mais également un port source (sert à retourner une réponse)
- ▶ la somme de contrôle se fait sur une "pseudo-entête IP"
(adresse source, adresse destination, protocole et longueur), plus le datagramme UDP (avec le champ checksum à 0)

Rappel : Un datagramme perdu ou erroné ne sera pas automatiquement renvoyé (contrairement à TCP)

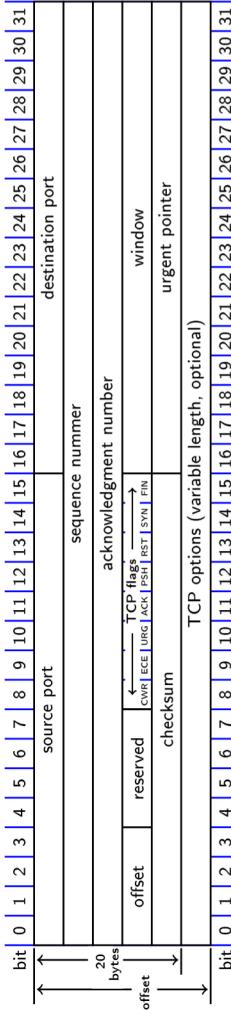
TCP (Transmission Control Protocol)

Mode connecté :

- ▶ établissement d'une connexion entre les 2 parties (initiateur / receveur)
- ▶ transmission des informations, avec accusés de réceptions du destinataire
- ▶ fermeture de la connexion

Note : agrément uniquement entre la source et la destination. Les routeurs/routes n'ont rien à voir avec la connexion TCP !

Entête TCP



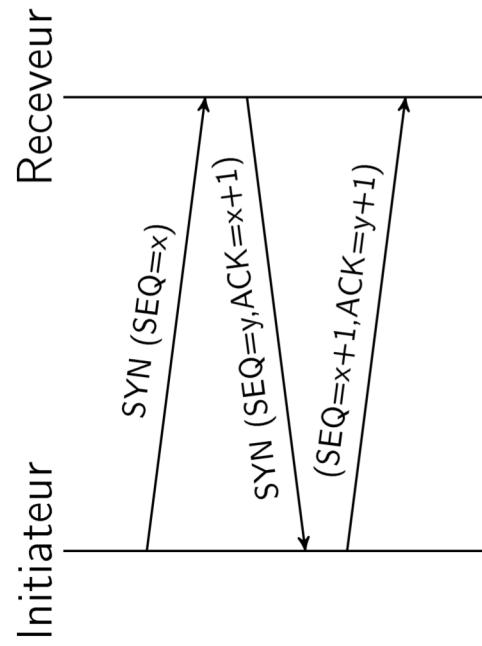
- ▶ SYN = synchronize : établissement d'une connexion
- ▶ ACK = acknowledge (accusé de réception présent)
- ▶ FIN : fermeture d'une connexion
- ▶ CWR/ECE : signalisation de congestion

TCP : Établissement de la connexion

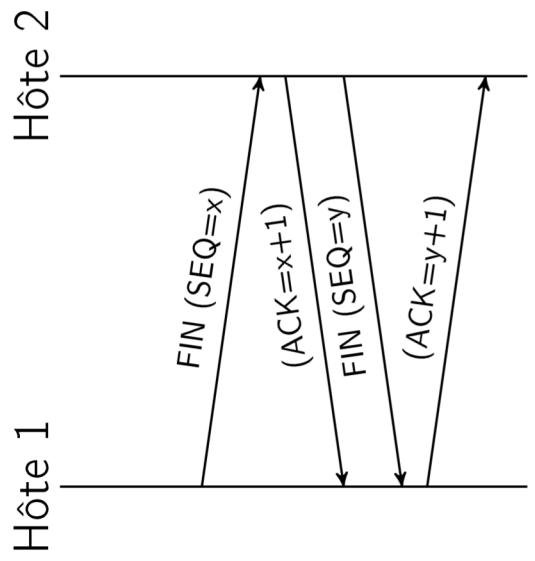
En 3 temps ("three-way-handshake")

- ▶ L'initiateur envoie au receveur un paquet SYN avec un numéro de séquence x (un nombre aléatoire)
- ▶ Le receveur envoie à l'initiateur un paquet SYN+ACK avec un numéro de séquence y (un nombre aléatoire), et l'accusé de réception = $x + 1$
- ▶ L'initiateur envoie au receveur un paquet ACK avec l'accusé de réception = $y + 1$

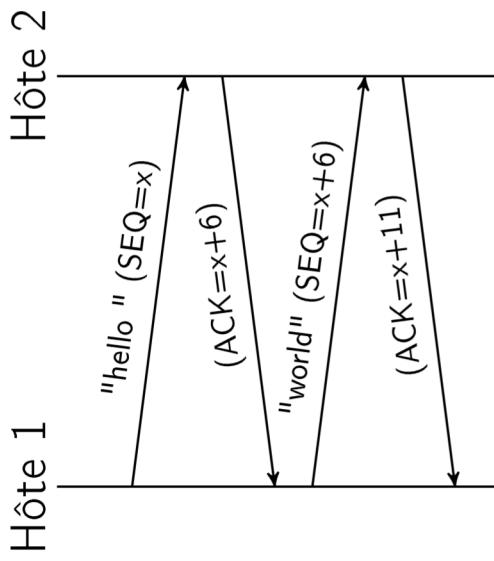
TCP : Établissement de la connexion



TCP : Fermeture de la connexion

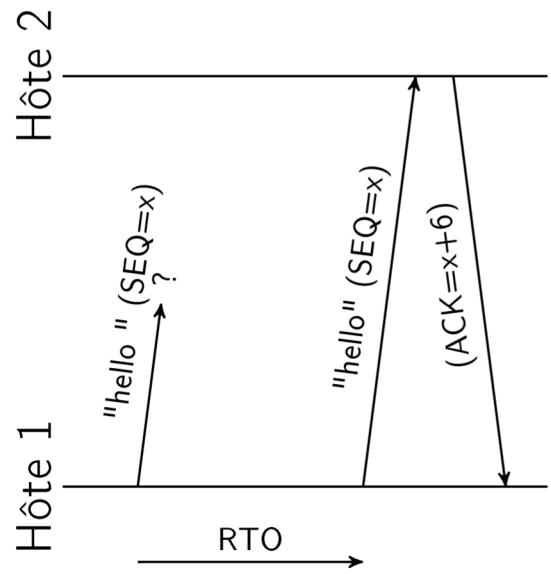


TCP : Transmission des informations



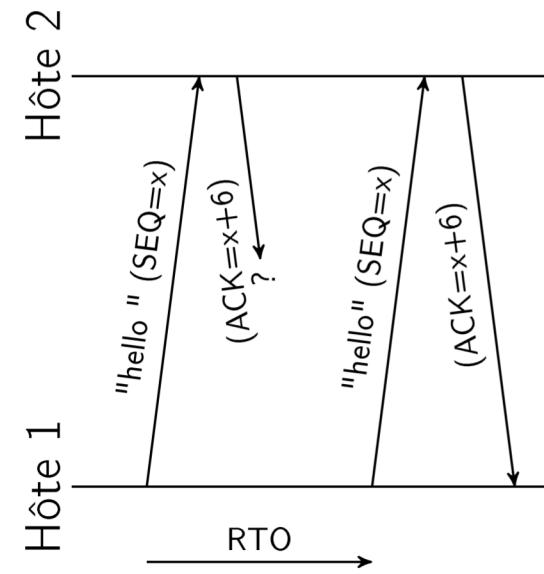
TCP : Transmission des informations

Perte de paquet ?



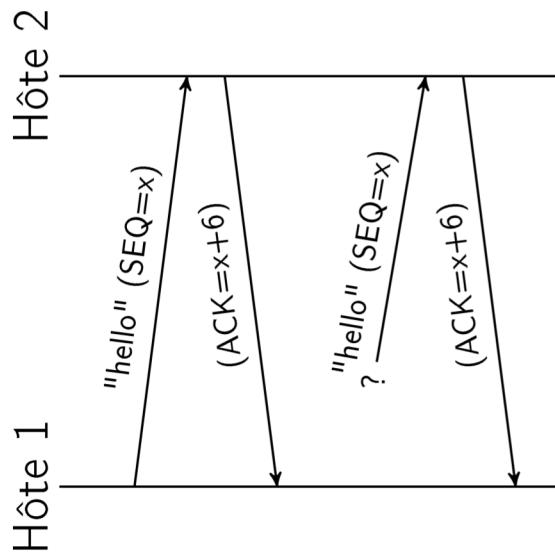
TCP : Transmission des informations

Perte de paquet ?



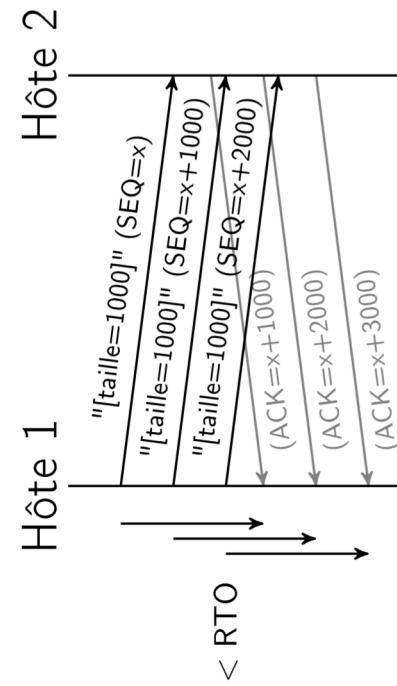
TCP : Transmission des informations

Duplication de paquet ?

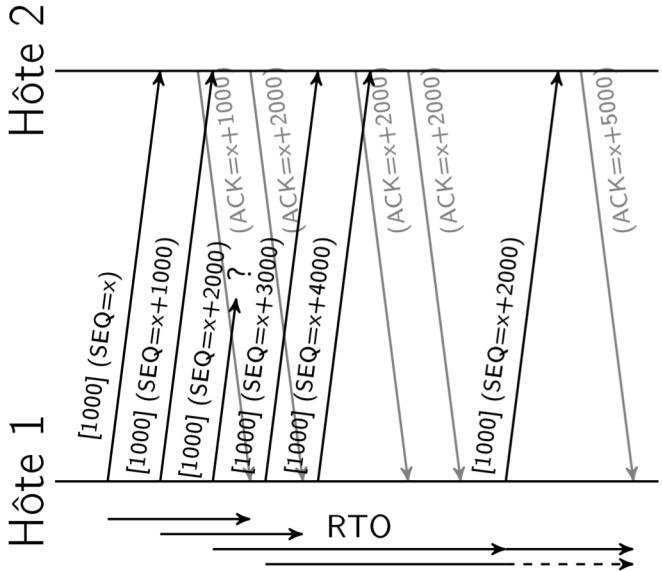


TCP : Transmission des informations

Problème : lent !



TCP : Transmission des informations



TCP : Transmission des informations

Pour limiter les ACK inutiles, il est possible :

- ▶ d'envoyer des ACK et des données en même temps
- ▶ d'attendre avant d'envoyer les ACK (et les cumuler)

Combien de temps attendre avant de retransmettre un segment ?
(RTO = Retransmission TimeOut)

- ▶ se baser sur le temps "moyen" des aller-retours
(RTT = Round-Trip Time)
- ▶ se baser sur l'écart type des aller-retours

Combien de temps attendre entre l'émission de deux segments ?

- ▶ l'émetteur essaye des temps de plus en plus courts, jusqu'à ce qu'il y ait signalé une congestion, ou perte de segments.

Fenêtre TCP

La couche transport renvoie les acquittements :

- ▶ à la réception des données,
- ▶ et non pas quand l'application lit les données dans la socket

Problème : il se peut que l'application ne lise pas assez vite les données.

Solution : la couche transport dispose d'un "tampon" (ou "fenêtre")

Le récepteur envoie avec l'acquittement la taille maximum d'octets à envoyer après l'octet acquitté (la place restante dans le tampon)

- ▶ L'émetteur se mettra en attente si la fenêtre est vide (ou trop petite)
- ▶ l'écriture sera bloquante (ou échouera, si la socket est non bloquante) du côté de l'application de l'émetteur

TCP ou UDP ?

Quand utiliser TCP :

- ▶ Quand on privilégie la simplicité
- ▶ Quand on privilégie la fiabilité

Quand utiliser UDP :

- ▶ Quand on privilégie la vitesse
- ▶ Si la perte de paquet pas très importante (streaming, voix)
- ▶ Si on gère d'un autre moyen les problèmes de transfert

Généralement pour vos applications : TCP

Couche "application"

Couche "application"

Généralement en espace utilisateur

- ▶ Multitude d'applications utilisant Internet
- ▶ vos programmes

DNS

DNS = Domain Name System

Traduit des noms en adresse IP. Exemple :
www.ens-lyon.fr → 140.77.167.5

- ▶ Organisation hiérarchique des noms et serveurs
- ▶ Architecture client/serveur
- ▶ Ports : 53 (TCP et UDP)
- ▶ Serveur : bind (Berkeley Internet Name Daemon),...
- ▶ Client : dans l'OS, nslookup
- ▶ Sous Linux les IP des serveurs de nom à consulter : dans /etc/resolv.conf

DNS

Organisation hiérarchique :

- ▶ un serveur "racine" traduit ".fr", et redirige vers un serveur qui traduit les ".fr"
- ▶ le serveur pour ".fr" traduit "ens-lyon.fr", et redirige vers un serveur qui traduit les ".ens-lyon.fr"
- ▶ le serveur pour ".ens-lyon.fr" traduit "www.ens-lyon.fr" vers un numéro IP
- ▶ 13 serveur racines (A-M), gérés par l'ICANN (Internet Corporation for Assigned Names and Numbers)
- ▶ .fr géré par AFNIC (Association française pour le nommage Internet en coopération)
- ▶ ens-lyon.fr géré par l'ENS de Lyon

DNS

exemple de fichier de configuration :

```
$TTL      86400 ; 24 hours could have been written as 24h or 1d
; $TTL used for all RRs without explicit TTL value
$ORIGIN example.com.
@ 1D IN SOA ns1.example.com. hostmaster.example.com. (
                                2002022401 ; serial
                                3H ; refresh
                                15 ; retry
                                1w ; expire
                                3h ; nxdomain ttl
)
IN NS      ns1.example.com. ; in the domain
IN NS      ns2.smokyjoe.com. ; external to domain
IN MX     10 mail.another.com. ; external mail provider
; server host definitions
ns1   IN A      192.168.0.1 ; name server definition
www    IN A      192.168.0.2 ; web server definition
ftp    IN CNAME www.example.com. ; ftp server definition
; non server domain hosts
bill   IN A      192.168.0.3
fred   IN A      192.168.0.4
```

DNS

Notes :

- ▶ Les serveurs font "cache" (les données ont une durée de vie).
- ▶ Il y a généralement plusieurs serveurs de nom pour un domaine. Il y a généralement un serveur primaire (maître) et des secondaires (esclaves).
- ▶ Un nom peut avoir plusieurs IP associées ("round robin", pour répartir la charge)

Web et HTTP

WWW (Word Wide Web) : système hypertexte (contenus reliés par des hyperliens) sur internet, utilisant généralement le protocole HTTP.

HTTP (Hypertext transfert protocol)

- ▶ Client/serveur. Protocole : TCP, Port : 80
- ▶ Serveur HTTP : Apache...
- ▶ Client (Navigateur) : Firefox, Chrome...

Web et HTTP

Le serveur HTTP permet de récupérer des "pages" (avec une organisation hiérarchique).

Les pages sont identifiées par une URL (Uniform Resource Locator)
<http://serveur.org/chemin/page>

Généralement, ces pages sont au format HTML (Hypertext Markup Language).

Une page est soit :

- ▶ statique (càd provient d'un fichier sur le disque du serveur)
- ▶ dynamique : résultat de l'exécution d'un programme/script :
 - ▶ PHP
 - ▶ CGI...

Web et HTTP

- Le client :
 - récupère les pages identifiées par leur URL
 - affiche les pages HTML, en utilisant les informations de mise en page
 - exécute les scripts de la page (javascript, AJAX)

Web et HTTP

Protocole HTTP (au travers d'une connexion TCP) :

Requête HTTP :

```
GET /chermin/page.html HTTP/1.1  
Host: www.serveur.com
```

Réponse HTTP :

```
HTTP/1.1 200 OK  
Date: Wed, 27 Apr 2016 14:33:32 GMT  
Server: Apache/2.4.10 (Debian)  
Vary: Accept-Encoding  
Content-Length: 1535  
Content-Type: text/html; charset=UTF-8  
<!DOCTYPE html>  
<html><head><title>Titre</title>  
...  
</head>  
<body>  
...  
</body>
```

Autres

- ▶ HTTPS : HTTP sur SSL (Secure Sockets Layer) ou TLS (Transport Layer Security) Port : 443
- ▶ FTP (File Transfert Protocol)
- ▶ SMTP (Simple Mail Transfert Protocol)
- ▶ Récupération des courriers sur un serveur :
 - ▶ IMAP (Internet Message Access Protocol)
 - ▶ POP (Post Office Protocol)
- ▶ SSH (Secure Shell) :
 - ▶ remplace telnet (non sécurisé)

