TCP/IP

Du paquet brut au navigateur Web : comment les protocoles s'empilent

Remarque 1 (Objectifs). À l'issue de ce chapitre, vous saurez :

- expliquer pourquoi le modèle TCP/IP est organisé en couches ;
- décrire les 4 couches du modèle TCP/IP et nommer un protocole pour chacune ;
- définir l'encapsulation et la désencapsulation ;
- expliquer le rôle du DNS et la structure d'une URL ;
- décrire le fonctionnement requête/réponse du protocole HTTP.

① Pourquoi un modèle en couches ?

Faire communiquer deux machines, c'est résoudre plein de problèmes : comment transmettre les bits sur le câble ? Comment trouver le destinataire à travers plusieurs réseaux ? Comment vérifier que rien n'a été perdu en route ? Comment comprendre le message ?

Propriété 1 **(Principe des couches)** On découpe ces problèmes en **couches** : chaque couche résout un problème bien délimité et s'appuie sur les services de la couche du dessous.

Avantage. Chaque couche peut évoluer indépendamment des autres. On peut changer de Wi-Fi à Ethernet sans rien changer à la façon dont une page web est demandée.

② Les quatre couches du modèle TCP/IP

Le modèle TCP/IP organise la communication en **quatre couches**, du plus proche de l'utilisateur au plus proche du matériel.

4. Application — `HTTP · DNS · SMTP`
protocoles utilisés par les logiciels

3. Transport — `TCP · UDP`
fiabilité de bout en bout

2. Internet — `IP (IPv4, IPv6)`
adressage et routage entre réseaux

1. Accès réseau — `Ethernet · Wi-Fi`
transmission physique sur le support

2.1 Ce que fait chaque couche

— **Accès réseau** : comment envoyer des bits sur le câble Ethernet ou les ondes Wi-Fi (carte réseau, adresses MAC).

— **Internet** : comment trouver la machine destinataire à travers plusieurs réseaux (adresses IP, routeurs).

— **Transport** : comment découper le message en petits morceaux, les numéroter, vérifier qu'ils arrivent (**TCP**) ou non (**UDP**).

— **Application** : ce que veut faire le logiciel (afficher une page web, envoyer un mail, regarder une vidéo...).

Remarque 2. Les protocoles « parlent » horizontalement (entre couches de même niveau sur les deux machines) mais passent verticalement (chaque couche rend service à celle du dessus).

③ Le protocole du bit alterné

Comment un émetteur peut-il s'assurer qu'un paquet est bien arrivé alors que le réseau peut en perdre ? Le **protocole du bit alterné** est la réponse la plus simple ; TCP en utilise une version élaborée.

1. L'émetteur envoie un paquet accompagné d'un bit de contrôle, 0 pour le premier paquet.
2. Le récepteur renvoie un **accusé de réception** (*acknowledgement*, ACK) portant le même bit.
3. À la réception de cet accusé, l'émetteur envoie le paquet suivant avec le bit inversé (1, puis 0, puis 1...).
4. Si l'accusé n'arrive pas avant la fin d'un délai (le **temporisateur**), l'émetteur **renvoie le même paquet**, avec le même bit.

Le bit permet de distinguer un nouveau paquet d'une répétition : si le récepteur reçoit deux fois de suite un paquet portant le bit 0, il sait que c'est un doublon (l'accusé de réception s'était perdu), il l'ignore et renvoie l'accusé.

Exemple 1 (Perte d'un paquet).

Émetteur	Réseau	Récepteur
envoie A, bit 0	→	reçoit A, bit 0 ; renvoie ACK 0
reçoit ACK 0 ; envoie B, bit 1	→ perdu	
délai écoulé ; renvoie B, bit 1	→	reçoit B, bit 1 ; renvoie ACK 1
reçoit ACK 1 ; envoie C, bit 0	→	reçoit C, bit 0 ; renvoie ACK 0

Exemple 2 (Perte d'un accusé de réception).

Émetteur	Réseau	Récepteur
envoie A, bit 0	→	reçoit A, bit 0 ; renvoie ACK 0
	← perdu	
délai écoulé ; renvoie A, bit 0	→	reçoit A, bit 0 : doublon, ignoré ; renvoie ACK 0
reçoit ACK 0 ; envoie B, bit 1	→	reçoit B, bit 1 ; renvoie ACK 1

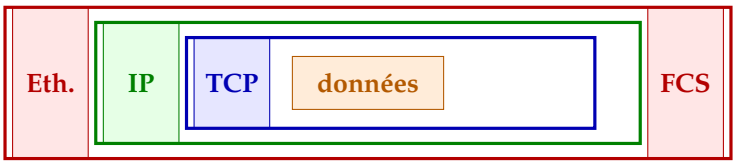
Dans les deux cas, chaque paquet finit par arriver **une seule fois et dans l'ordre**. Le prix à payer : l'émetteur attend l'accusé avant d'envoyer le paquet suivant, ce qui limite le débit. TCP généralise l'idée avec des numéros de séquence sur 32 bits et l'envoi de plusieurs paquets avant le premier accusé.

④ L'encapsulation

Définition 1 **(Encapsulation)** Lorsqu'un logiciel envoie un message, chaque couche ajoute son propre en-tête autour du contenu reçu de la couche supérieure. C'est l'**encapsulation**. À l'arrivée, la machine destinataire fait l'opération inverse (la **désencapsulation**), couche par couche.

Remarque 3 (Image à retenir). On enferme une lettre dans une enveloppe, qu'on glisse dans un carton, qu'on met dans un conteneur. À l'arrivée, on ouvre le conteneur, puis le carton, puis l'enveloppe.

chaque couche ajoute son en-tête autour du contenu



trame Ethernet envoyée sur le câble

La couche Accès réseau ajoute aussi un bloc en fin de trame, noté FCS (séquence de contrôle de trame) : c'est un code calculé à partir du contenu de la trame, qui permet au destinataire de détecter les erreurs de transmission.

⑤ DNS, URL, HTTP

5.1 DNS : l'annuaire d'Internet

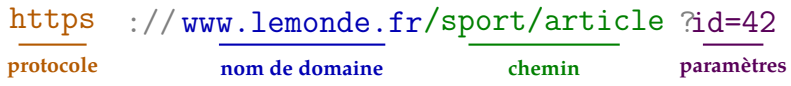
Vous tapez `google.com` dans votre navigateur. Mais votre ordinateur a besoin d'une **adresse IP** pour contacter le serveur. C'est le rôle du DNS de faire la conversion.

Définition 2 **(DNS : Domain Name System)** Un service distribué qui traduit les **noms de domaine** (`google.com`) en **adresses IP** (`142.250.179.110`), et inversement.



5.2 Une URL décomposée

L'**URL** (*Uniform Resource Locator*) est l'adresse complète d'une ressource sur le web.



— **Protocole** : `HTTP` ou `HTTPS` (la version sécurisée).

— **Nom de domaine** : résolu en IP par le DNS.

— **Chemin** : la ressource demandée sur le serveur.

— **Paramètres** (facultatif) : des informations supplémentaires.

5.3 HTTP : le protocole du Web

Définition 3 **(HTTP : HyperText Transfer Protocol)** HTTP est le protocole d'application utilisé par les navigateurs pour demander des pages web aux serveurs.

Un échange HTTP suit le schéma **requête** → **réponse** :

— le navigateur envoie une **requête** au serveur (verbe `GET` pour demander une page, `POST` pour envoyer des données...) ;

— le serveur renvoie une **réponse** contenant un **code de statut** (`200` = OK, `404` = page introuvable) et le contenu de la page.

Propriété 2 **(HTTPS)** **HTTPS** est la version **sécurisée** de HTTP : la communication est chiffrée de bout en bout. C'est aujourd'hui la norme.

6 Synthèse

Propriété 3 **(À retenir)**

— Le modèle **TCP/IP** organise la communication en **4 couches** : accès réseau, internet, transport, application.

— Chaque couche résout un problème bien délimité et s'appuie sur les services de la couche du dessous.

— L'**encapsulation** consiste à ajouter, à chaque couche, un en-tête autour du contenu reçu. La désencapsulation fait l'opération inverse à l'arrivée.

— Le **DNS** traduit les noms de domaine en adresses IP. Une **URL** se compose d'un protocole, d'un nom de domaine, d'un chemin et de paramètres.

— **HTTP(S)** est le protocole du Web : requête (`GET`, `POST`) → réponse (code de statut + contenu). HTTPS chiffre la communication de bout en bout.