

Réseau

Encapsulation

Prenons un exemple pour montrer comment les données circulent de manière encapsulée.

Remarque: les termes en gras sont définis plus bas.

Un navigateur s'exécutant dans un ordinateur veut envoyer une requête **HTTP** (demande d'une page Web) à un serveur.

Il écrit donc sa requête sur la couche 7 (**Application**); par exemple:

GET / HTTP/1.1

Host: www.eduge.ch

et la transfère aux couches inférieures pour faire voyager l'information.

Les deux couches en-dessous encapsulent la requête en rajoutant chacune un en-tête:

- couche 6 (**Présentation**): pour l'encodage, chiffrement et compression
- couche 5 (**Session**): pour la gestion des connexions avec le serveur

Ensuite, la couche 4 (**Transport**) ajoute l'en-tête de transport pour attribuer un **port** de départ et un **port** d'arrivée. Ce sera ce qui permettra à l'application qui communique de savoir que l'information lui est destinée. En effet, il pourrait y avoir d'autres onglets du navigateur ou d'autres navigateurs sur le même ordinateur qui demandent aussi des ressources, comment savoir, quand les ressources demandées arrivent, à qui elles sont destinées?

Les deux protocoles les plus utilisés dans cette couche sont **UDP** et **TCP**.

Le premier (UDP) ajoute un en-tête simple (permet d'aller très vite mais sans certifier la réception des données).

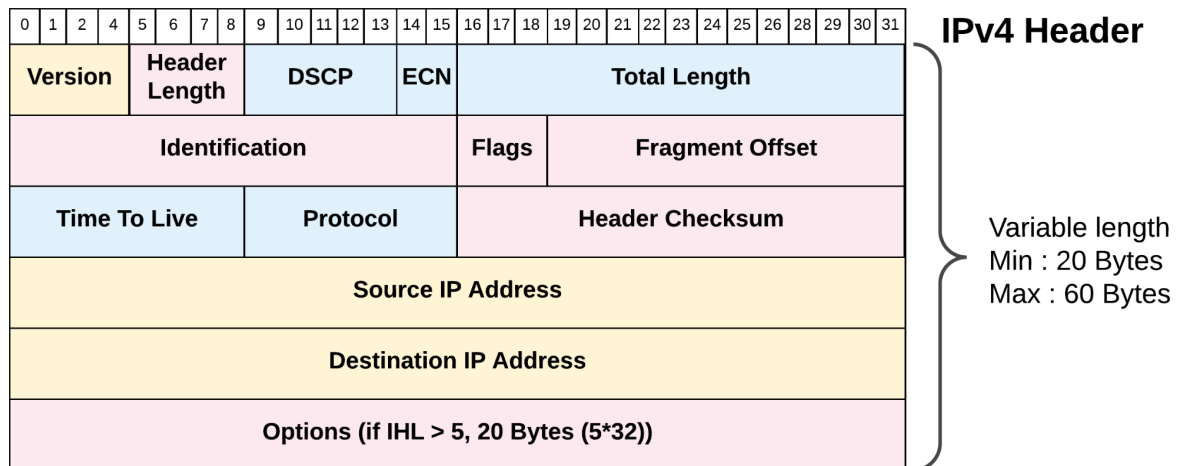
Le deuxième (TCP) ajoute un en-tête plus long (un peu plus gourmand en données mais permet notamment de réaliser des connexions entre ordinateurs pour s'assurer que les données envoyées ont bien été reçues ou les renvoyer dans le cas contraire)

Voici à quoi ressemblent ces en-têtes:

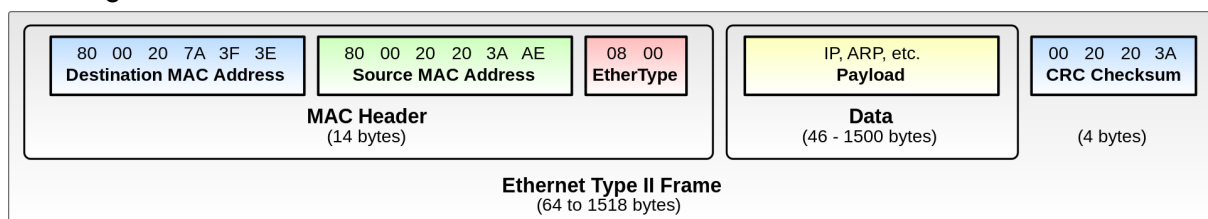
TCP Segment Header Format								
Bit #	0	7	8	15	16	23	24	31
0	Source Port				Destination Port			
32	Sequence Number							
64	Acknowledgment Number							
96	Data Offset	Res	Flags		Window Size			
128	Header and Data Checksum				Urgent Pointer			
160...	Options							

UDP Datagram Header Format								
Bit #	0	7	8	15	16	23	24	31
0	Source Port				Destination Port			
32	Length				Header and Data Checksum			

Tout ceci (données d'origine encapsulées grâce aux divers en-têtes) descend encore vers les couches basses. On arrive donc dans la couche 3 (**Réseau** ou **Internet**) qui utilise le protocole **IPv4** (ou **IPv6**). Elle va donc rajouter l'en-tête IP. Celle-ci correspond notamment aux adresses IP de l'ordinateur et du serveur:

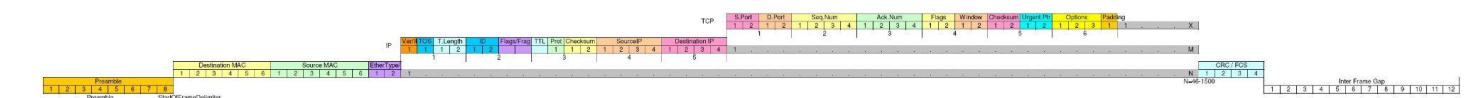


Pour finir, tout ceci va encore être encapsulé dans la couche 2 (**Liaison**) qui va rajouter l'en-tête **Ethernet** (ou **Mac**) ainsi qu'un pied de frame (à la fin) pour assurer une vérification de l'intégrité des données lors du transfert:



C'est le résultat de toutes ces encapsulations qui va voyager sur le réseau, à travers les ondes Wi-Fi, les câbles réseau ou les fibres optiques. C'est ce qu'on appelle la couche 1 (**Physique**).

Voici donc en détail, ce que contient la trame qui va sortir de l'ordinateur:



Une fois que tout ceci sera arrivé, en passant par d'innombrables commutateurs et routeurs, au serveur Web, celui-ci va décapsuler les données, couche après couche, pour en extraire les données:

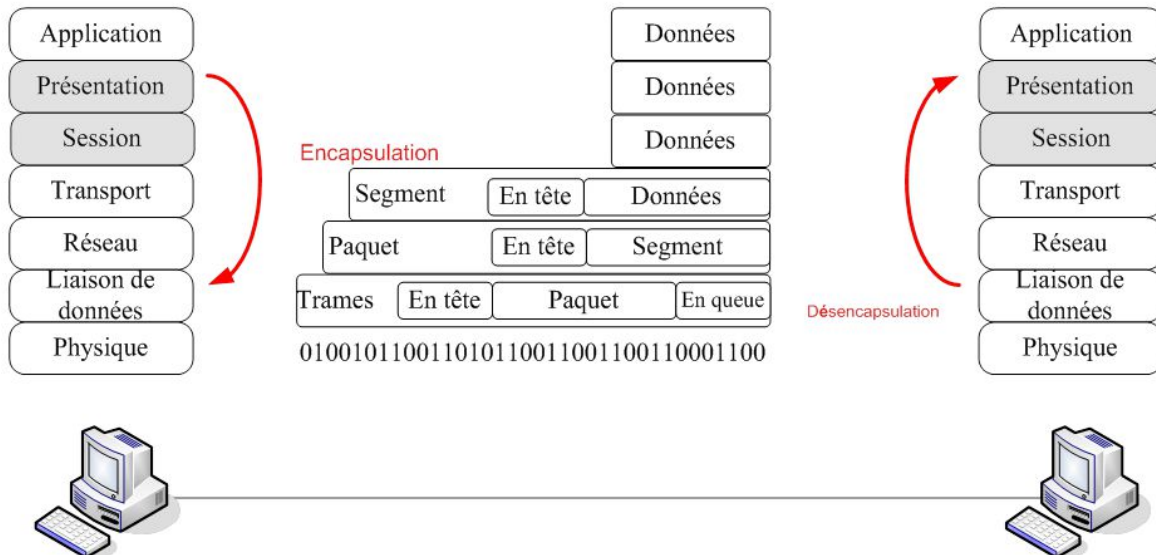
GET / HTTP/1.1
Host: www.eduge.ch

Il va donc envoyer à son tour la page demandée et les données qu'il enverra vont de nouveau être encapsulées pour rejoindre l'ordinateur qui avait demandé la page.

Voici un résumé de la transmission avec l'encapsulation de l'émetteur (à gauche) et la décapsulation du récepteur (à droite) avec les couches du modèle OSI:

Le modèle OSI

■ Principe de l'encapsulation :



Glossaire:

- Couche 1 (**Physique**)
- Couche 2 (**Liaison**)
 - Adresse **MAC**: C'est l'adresse unique d'un appareil connecté au réseau (attribuée dans l'usine où il a été fabriqué). Par exemple, une carte réseau, un routeur, etc.
 - Elle est composée de 6 octets, en notation hexadécimale (chaque octet s'écrit en 2 chiffres de 0 à F), séparés par ":".
 - Exemple: CC:F9:E4:9E:DD:06
 - Protocole **Ethernet**: C'est le protocole qui gère l'envoi des trames à travers la couche Physique, en gérant notamment les collisions et en indiquant les adresses MAC de l'émetteur et du destinataire de chaque trame. Ceci se fait à l'aide d'un en-tête de trame et d'une queue de trame (où on trouve un checksum qui est le résultat d'un calcul pour tester l'intégrité des données de la trame).

- Couche 3 (**Réseau**)
 - Protocole **IP**: C'est le protocole qui permet d'encapsuler un paquet, c'est-à-dire une unité d'information qui comporte un en-tête mentionnant l'**adresse IP** de l'émetteur et du destinataire ainsi que d'autres informations.
 - **Adresse IP**: C'est l'adresse logique d'un ordinateur. Elle est composée de 4 nombres normalement donnés en notation décimale et séparés par un point.
 - Exemple: 192.168.1.202
 - Souvent, dans notre réseau LAN à la maison, les trois premiers nombres correspondent au réseau géré par le routeur de la box alors que le dernier nombre correspond au sous-réseau, c'est-à-dire au nombre attribué à chaque appareil connecté par la box.
- Couche 4 (**Transport**)
 - **Port**: C'est le nombre attribué à un processus (programme) tournant sur un ordinateur. Ceci permet de déterminer, quand un message est reçu par un ordinateur, à quel processus il est destiné.
 - Certains ports sont standards comme le numéro 80 pour un serveur Web.
 - **UDP**: C'est le protocole le plus simple pour un segment, c'est-à-dire l'unité d'information sur la couche transport. Il ne gère pas de vérification de l'arrivée des segments à bon port.
 - **TCP**: C'est l'autre protocole le plus utilisé sur la couche 4. Il est plus complexe (donc gourmand en données) mais il permet de gérer la vérification d'arrivée des données et donc d'assurer la fiabilité du transport!
- Couche 5 (Session)
- Couche 6 (Présentation)
- Couche 7 (Application)
 - Ces trois dernières couches sont les couches les plus hautes. Dans le modèle TCP/IP (alternative au modèle OSI), on les réduit à une seule couche.
 - La couche application transporte l'information qu'utilise concrètement l'utilisateur. On y trouve de très nombreux protocoles:
 - **HTTP**: C'est le protocole qui permet d'interroger un serveur Web pour lui demander des ressources comme telle page, telle image.
 - Beaucoup d'autres protocoles comme
 - **FTP** (transfert de fichiers)
 - **POP3, IMAP, SMTP** (transport d'emails)
 - **TLS** (transport chiffré de données)
 - **DNS** (Domain Name Server pour transformer une adresse URL d'un site Web en l'adresse IP du serveur qui l'héberge)
 - **ARP** (protocole pour associer les adresses IP des appareils connectés à un LAN à leurs adresses MAC)
 - etc.