# Client / serveur

Un **serveur** **offre un service**. Par exemple, un serveur Web permet de mettre à disposition des utilisateurs des pages Web. On dit alors qu’une machine est un serveur dès qu’elle fournit un service.

Comme le but du serveur est de fournir un service à toute personne qui le demande en tout temps, on dit qu’il est en **écoute**. Il va écouter sur le réseau et être prêt à répondre aux requêtes qu’ils lui sont adressés.

On peut donc en déduire qu'**un service est un programme qui est en écoute sur une machine**. On peut alors appeler cette machine un serveur.  
Mais ce service ne servirait à rien s'il n'était pas utilisé, et pour cela, il faut que **des clients** viennent se connecter dessus et l'utilisent !

Le **client** est simplement un programme qui se connecte à un service pour l'utiliser. En ce moment même, vous utilisez un client Web qui est votre navigateur et qui se connecte au serveur du site Adel. C'est bien **une connexion client/serveur** qui est établie entre votre navigateur et le serveur Web d’Adel.

Vous en connaissez d’autres, comme Outlook, qui est votre client de messagerie, ou Filezilla si vous utilisez le protocole FTP.

Ma machine est un serveur, mais c'est aussi avec cette machine que je me connecte sur des sites Web en tant que client. Elle joue donc à la fois le rôle de client et celui de serveur.

La plupart du temps, les machines serveur ne jouent pas le rôle de client, ou très peu. Elles sont spécialisées en tant que serveur et pour des raisons de sécurité, on limite les services disponibles à ceux qui sont strictement nécessaires, et on évite que cette machine ait une activité de client qui pourrait engendrer des failles.  
De la même façon, on considère que les machines des utilisateurs jouent en majeure partie le rôle d'un client et sont donc vues comme des clients et non des serveurs.

# Couche 4 - Transport

Nous savons à présent faire dialoguer des machines ensemble, proches ou à distance, d’un bout à l’autre d’Internet. Mais le but est de faire dialoguer une application cliente avec une application serveur. La couche 4 rentre en jeu pour cette partie.

Elle permet de faire dialoguer ensemble des applications qui sont sur différentes machines.

Rôle couche 4 : Gérer les connexions applicatives

Sur la couche 2 et la couche 3, on a appris qu’il fallait une adresse pour identifier les éléments nécessaires à la communication => l’adresse Mac identifie la carte réseau en couche 2, et l’adresse IP identifie l’adresse de notre machine dans un réseau en couche 3.

## Port …

Pour la couche 4, le moyen d’identification c’est le **port**. Le port est l’adresse d’une application sur une machine. Ainsi, on peut identifier toute application qui tourne sur notre machine et qui a besoin de dialoguer sur le réseau.

Par exemple, le port 80 est le port utilisé pour les serveurs Webs, et le 22 est le port pour le protocole SSH.

Les ports sont codés en décimale sur deux octets. Ils peuvent donc prendre 2^16 valeurs, soit 65 536 valeurs. Étant donné qu’on commence à compter à **0**, on peut aller jusqu’au port **65 535**.

Comment pour le port 80 ou le port 22, certains d’entre eux sont réservés pour des protocoles ou applications associés, en voici une très petite partie :

| **Application** | **Port réservé** | **Protocole** |
| --- | --- | --- |
| Web | 80 | HTTP |
| Envoi mail | 25 | SMTP |
| Réception mail | 143 | IMAP |
| Connexion à distance sécurisée | 22 | SSH |
| Web sécurisé | 443 | HTTPS |
| Transfert de document | 20/21 | FTP |
| Résolution de noms | 53 | DNS |

Avant, seuls les ports inférieurs à 1024 étaient réservés. Mais ce n’est plus le cas aujourd’hui. Les nouvelles applications utilisent n’importe quel port. Cependant, cela reste une convention, de ne pas utiliser les ports en dessous de 1024 pour personnaliser nous-mêmes le port d’un protocole, afin d’éviter tout conflit.

Les **applications clientes ont elles aussi un port**, mais ils ne sont pas réservés. Ils sont **attribués** **aléatoirement au-dessus de 1024** par le système d’exploitation. Comme le serveur est en écoute permanente, il est important que l’on connaisse le port auquel il doit s’adresser. Pour un client, l’application ne sera en écoute que le temps de son fonctionnement. Comme cela, il peut choisir au hasard tant que le système d’exploitation sait quelle application se trouve derrière quel port.

## … et Protocole

En couche 2, il y avait le protocole Ethernet, en couche 3, le protocole IP.

Pour la couche 4, nous avons 2 protocoles : TCP et UDP.

Les personnes qui ont créé les réseaux se sont rendu compte qu'il pouvait y avoir deux besoins différents pour le transport des données des applications :

🡪 un protocole fiable, mais sans nécessité de rapidité ;

🡪 un protocole rapide, mais sans nécessité de fiabilité.

La première catégorie regroupe une très grande majorité des applications d'Internet, car bon nombre d'entre elles ont besoin que chaque paquet émis soit reçu coûte que coûte !

Ce sont notamment les applications comme le Web, la messagerie, le SSH, beaucoup de jeux en ligne, etc. Si un paquet est perdu, une page Web ne pourra pas s'afficher correctement, ce sera pareil pour un mail, etc.

La seconde catégorie regroupe moins d'applications, mais vous comprendrez vite pourquoi ces applications ont besoin d'être instantanées et peuvent se permettre qu'un paquet ne soit pas reçu. Il s'agit notamment des applications de streaming, comme la radio ou la télé sur Internet.

Pour une radio en ligne, il est essentiel que les informations soient envoyées en temps réel, le plus rapidement possible. Par contre, si un ou plusieurs paquets sont perdus, on ne va pas arrêter la radio pour autant. L'utilisateur aura des coupures de connexion, mais la radio continuera d'émettre.

C'est pour cela que nous avons deux protocoles pour la couche 4 : le protocole TCP et le protocole UDP.

**TCP** est de la première catégorie, c'est un protocole extrêmement fiable.

Chaque paquet envoyé doit être identifié par le receveur, qui en réémettra un autre s'il ne reçoit pas d'accusé de réception. On dit alors que c'est un **protocole connecté**.

On peut le comparer au téléphone. Quand on appelle quelqu'un, on dit "Allo ?" pour savoir si la personne est là et, s'il y a des silences dans la communication, on essaye de voir si la personne est encore à l'écoute, etc. En TCP ce sera pareil, pour chaque information envoyée, on vérifiera que la machine en face l'a bien reçue.

**UDP**, lui, est un protocole rapide, mais peu fiable. Les paquets sont envoyés dès que possible, mais on se fiche de savoir s'ils ont été reçus ou pas. On dit qu'UDP est un **protocole non connecté**.

C'est un peu comme le courrier (sauf que le courrier n'est pas rapide...) On envoie notre lettre, et ensuite, on prie très fort pour que celle-ci arrive, mais on n'en sait rien.

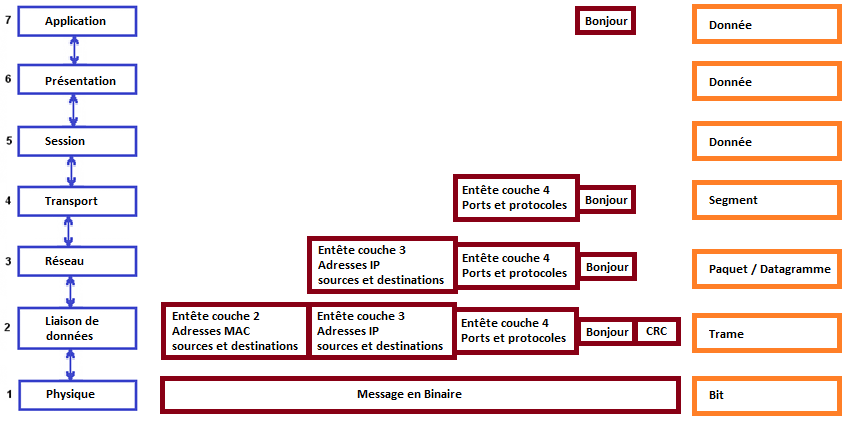
#### Récap

* Un serveur est une machine qui offre un service
* Un service est un programme qui est en écoute sur une machine (le serveur).
* Le client est un programme qui se connecter à un service pour l’utiliser
* La couche 4 du Modèle OSI est la couche Transport
* Elle gère la communication entre les applications présentes sur des machines distantes
* Elle fonctionne avec un port, qui est l’adresse d’une application et va de 0 à 65 535
* Et deux protocoles : TCP et UDP
* TCP est un protocole connecté : qui vérifie que la connexion est bien établie avant d’envoyer un message, il est donc fiable, mais peut être lent
* UDP est un protocole non connecté : il ne vérifie rien, il envoie directement, donc il n’est pas fiable, car des morceaux de paquets peuvent se perdre, mais est rapide

## Que devient notre message d'origine ?

Comment circule alors notre message sur le réseau ? Que devient notre message d’origine ?

Deux machines veulent discuter entre elles. La 1ère machine envoie un message « Bonjour » à destination de la seconde, cela se passe sur la couche 7 : Application.

Le message va redescendre toutes les couches du modèle OSI, de la couche 7 où le message est écrit, à la couche 1 où le message arrive sur le physique et va être envoyé sur le réseau en ayant récupéré des informations (en tête) au fur et à mesure du passage des couches pour envoyer ce message à la bonne personne. Le fait de rajouter des informations à notre message d’origine au fur et à mesure de la traversée des couches s’appelle **l’encapsulation**.

→ Nos données (le message) partent de la couche 7 Application, il arrive à la couche 4 Transport et forme un segment en rajoutant devant nos données l’entête de couche 4 qui contient le port et le protocole utilisé par ce message.

→ Le segment passe ensuite à la couche 3 Réseau et devient un paquet ou un datagramme (les deux termes peuvent être utilisés). Aux données et à l’entête de couche 4 est alors rajouté l’entête de couche 3, c’est-à-dire l’adresse IP de la machine source (machine émettrice du message) et l’adresse IP de la machine destinatrice.

→ Puis le message est envoyé à la couche 2, Liaison de donnée, il devient une trame, en rajoutant au début de notre paquet l’entête de couche 2, qui permet de préciser les adresses MAC source et destination de ce paquet. Et à la fin du paquet on rajoute le CRC.

→ Enfin, cette trame est transmise à la couche 1, Physique, et est transformée en bit, afin de circuler sur le réseau et être transmise à l’autre machine.

On constate qu'au final, c’est une trame de couche 2 qui va circuler sur le réseau, traduite en binaire, cette trame contient le datagramme de couche 3, qui lui-même contiendra le segment de couche 4, qui contiendra à son tour le message d’origine.

Ainsi, la trame Ethernet vue précédemment était très simplifiée, car dans les « données à envoyer » vues sur la Trame de couche 2 (en vert ci-dessous) se trouvent les informations des couches précédentes.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Entête de couche 2 | | | Données à envoyer | | | | CRC |
| Adresse MAC Destination | Adresse MAC Source | Ether Type |
| Entête de couche 3 | | Entête de couche 4 | Message initial |
| Adresse IP source | Adresse IP destination | Port et protocole utilisé | Données initiales |

Donc, quand un message est envoyé depuis une machine, il descend les couches de la 7 à la 1 et s’encapsule, puis il passe sur le réseau en binaire, transite par les matériels réseau pour arriver à son destinataire et là, il remonte les couches du modèle OSI de la couche 1 à la couche 7, et subit la désencapsulation.

