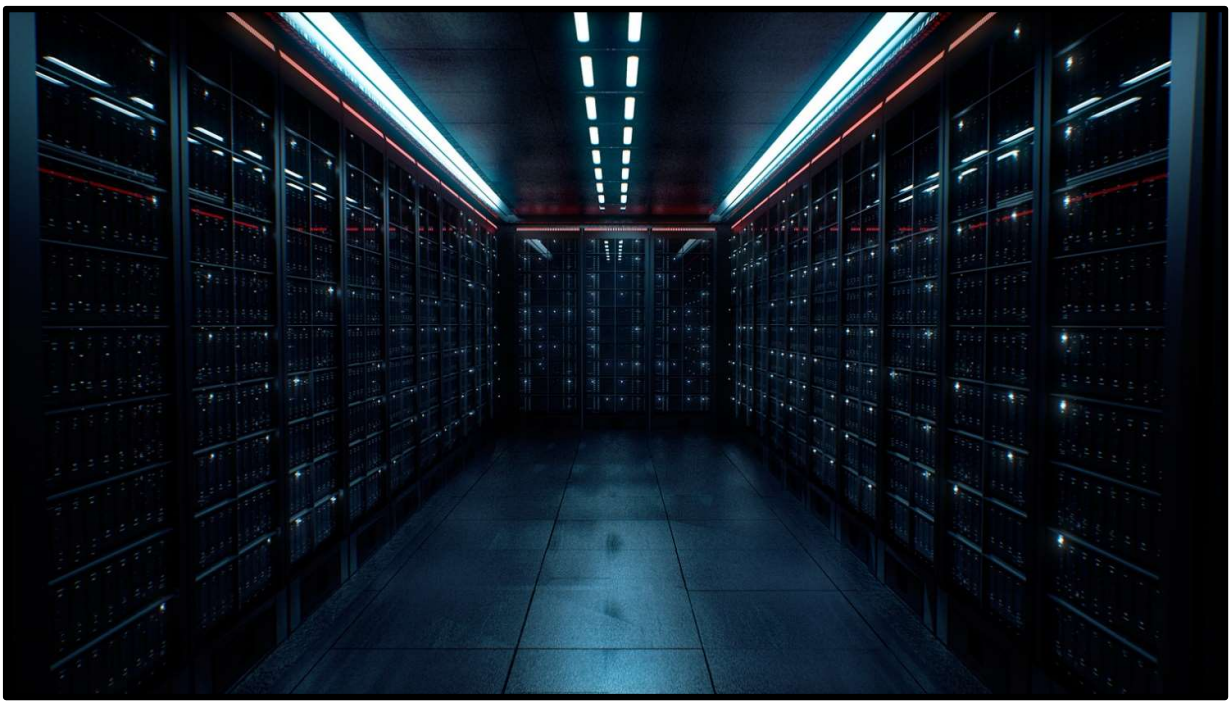
INTRODUCTION AUX RESEAUX INFORMATIQUES
LES NOTIONS FONDAMENTALES

PAR FRANCK FERMAN



*S*ommaire

Table des matières

Qu'est-ce qu'un ordinateur ?

Avant d'approfondir vos compétences dans le domaine des réseaux informatiques, vous devriez (si ce n'est pas déjà le cas) commencer par apprendre les bases de l'informatique en elle-même — comprendre le fonctionnement d'un ordinateur, ses différents composants, les prémices de l'ordinateur (**Turing** et ses machines et le modèle de **Von NEUMANN** par exemple), etc.

Ce document n'est véritablement pas un cours d'informatique général mais un cours spécifiquement dédié au monde des réseaux informatiques.

Nous n'allons donc pas nous attarder avec ces notions, je vous invite très fortement à approfondir ces notions élémentaires de votre côté.

Tout comme un gastro-entérologue ou un gynécologue a dû passer par des études de médecine générale avant de se spécialiser, un bon « **expert** » devra approfondir certaines notions d'informatique générale avec de se spécialiser.

Certaines personnes brûlent les étapes en voulant aller trop vite, minimisant l'importance de certaines notions et compétences pourtant élémentaires.

Selon moi, les prérequis qui vous permettront de suivre un cursus d'apprentissage le plus linéaire possible sont les suivants (il s'agit d'une liste non exhaustive et purement personnelle bien que créée en essayant d'être le plus objectif possible) :

- Développer vos softs-skills (apprendre à apprendre, trouver les meilleures méthodes de travail, établir un diagramme de **GANTT**…)

- Découvrir les différents métiers de l'informatique (technicien, administrateur, expert forensic, analyste SOC, développeur full stack…)

- Comprendre ce qu'est un ordinateur et ses prémices (cf. **TURING** ou **NEUMMAN** par exemple), les bases de l'électronique, monter son ordinateur, connaitre les différents périphériques internes et externes…

- Connaitre les différents systèmes d'exploitation tels que Windows (ses différentes éditions et versions), Linux (**Debian**, **RHEL**, **Arch**, **FreeBSD**), savoir installer, configurer et déployer ces systèmes, les bases de l'algorithmie et le Scripting (**Shell**, **PowerShell**, **Batch**, **Ansible**…)

- Apprendre la virtualisation et le principe de conteneurisation (**Hyper-V**, **VMware/ESXi**, **Proxmox**, **Docker**, **Kubernetes**…)

Pour ne pas s'égarer, revenons-en à notre définition, qu'est-ce qu'un ordinateur ?

Si l'on tente rapidement de définir ce qu'est un ordinateur au sens large du terme (tablettes, smartphones, IoTs), on peut y lister les éléments communs suivants :

- Un ordinateur reçoit des informations par l'intermédiaire d'un utilisateur ou d'un réseau.

- Un ordinateur émet des informations via le réseau ou un de ses périphériques.

- Un ordinateur a besoin d'une source d'énergie pour fonctionner.

Néanmoins cette première tentative de définition s'avère quelque peu infructueuse puisque l'on peut penser à plusieurs contre-exemples qui satisfont ces trois critères et qui ne sont pas pour autant des ordinateurs :

- Un réfrigérateur nécessite une source d'énergie, il reçoit des informations de la part de capteurs (la température) et il en émet sous la forme de signaux lumineux électriques.

- Un interrupteur fonctionnant avec la luminosité ambiante reçoit de l'information par le biais de son capteur et transmet de l'information, de plus le capteur peut nécessiter une source d'énergie pour fonctionner.

- Le système ABS (Antiblockiersystem) d'aide au freinage d'urgence d'une voiture reçoit également de l'information (la vitesse) et en émet sous forme de pression hydraulique sur le système de freinage.

Plus généralement, tout système électronique embarqué (système électronique et informatique autonome, ayant une tâche précise) vérifie ces trois critères.

Tous ces exemples montrent simplement que la définition de ce qu'est un ordinateur n'est pas une chose si triviale que cela.

Qu’est-ce qu’un réseau informatique ?

Un réseau informatique est un ensemble d'équipements reliés entre eux dont le but principal est d’aboutir à un échange d’informations.

Un ensemble ? Plus précisément, un réseau est composé d’au moins deux machines. En effet, comme dit précédemment, le but principal d’un réseau étant d'aboutir à un échange d’informations, cet échange ne peut donc logiquement pas s’effectuer avec une seule machine.

D’équipements, de machines ? Un équipement ou une machine désigne simplement un appareil ayant la capacité d'envoyer ou de recevoir des données informatiques. Parmi ces équipements, vous y trouverez par exemple les ordinateurs, les imprimantes, les serveurs, les téléphones portables (smartphones), les tablettes, etc.

L’interconnexion et l’échange d’informations

Il est nécessaire d’avoir certains éléments pour permettre une interconnexion et un échange d’informations entre des machines.

En effet, pour que des machines puissent se comprendre et échanger des données, il faut qu’ils passent par un langage commun (ce que l'on appelle un protocole).

En ce qui concerne l’interconnexion, il faut passer par des équipements réseaux (un commutateur par exemple) et un lien (que l’on appelle support) pour que les données puissent transiter d’un équipement à l’autre.

Les composants physiques du réseau

Les réseaux possèdent trois types de composants physiques.

- **Les périphériques de terminaison** : il s'agit des équipements qui se trouvent en bout de chaine, tels que les ordinateurs, les imprimantes, les serveurs, les téléphones portables (smartphones), les tablettes, etc.

- **Les équipements réseau** : ce sont les équipements intermédiaires ayant pour rôle d'interconnecter les périphériques de terminaisons entre eux, tels que le commutateur (switch), le routeur (router), le concentrateur (hub), le répéteur, etc.

- **Le support** : comme vu (brièvement) précédemment, c'est ce qui transporte les données sur le réseau, il s'agit par exemple des câbles (dans lesquels circulent des signaux électriques), la fibre optique (qui propagent des ondes lumineuses dans des fils très fins, en verre ou en plastique), les ondes radio (qui propagent des ondes électromagnétiques dans le vide), etc.

Les différents types de réseaux

Il existe plusieurs types de réseaux que l'on distingue selon leur taille (en terme de nombre de machines), leur vitesse de transfert des données ainsi que leur étendue.

- Le *PAN* pour *Personal Area Network* ou *réseau personnel*.
  Le *PAN* dessert généralement *un seul individu* et couvre *quelques mètres* de distance et permet la *communication* des *appareils* dans *une même pièce*. Il peut s'agir *d'équipements* tels qu'une *souris*, un *casque*, une *enceinte*, une *montre connectée*, etc.

  - Le *LAN* pour *Local Area Network* ou *réseau local*.
  Le *LAN* couvre *généralement* une zone *entre 10m à 1km* et permet la *communication* des *appareils dans un même bâtiment*, une *maison*, une *école*, etc.

  - Le *MAN* pour *Metropolitan Area Network* ou *réseau métropolitain*.
  Le *MAN* relie plusieurs LAN entre eux à proximité grâce à des *commutateurs* (*switchs*) et des *routeurs interconnectés* par des *liens hauts débits* (en général de la *fibre optique*) et couvre généralement *une zone maximale d'environ 10km, à l'échelle d'une ville*.

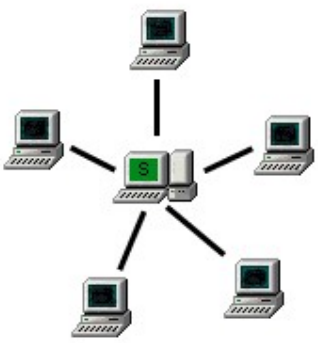
  - Le *WAN* pour *Wide Area Network* ou *réseau étendu*.
  Le *WAN* est une interconnexion de plusieurs LAN ou MAN, à l'échelle d'un pays ou d'un continent. Le *réseau Internet* est le plus grand *WAN publique*, mais il existe également des WAN privés pour les *organisations internationales* tels que des *banques*.
  Les *supports* utilisés peuvent par exemple être des *câbles sous-marins* ou la *transmission par satellite*.
  Les fournisseurs d'accès *Internet* utilisent *le réseau WAN*.

  - Le *GAN* pour *Global Area Network* ou *réseau global*.
  Le *GAN* est une interconnexion de *plusieurs WAN, à l'échelle mondiale*. Les *GAN* utilisent les infrastructures de *fibre optique* des réseaux étendus (*WAN*) et combinent ces derniers avec des *câbles sous-marins internationaux* ou la *transmission par satellite*.

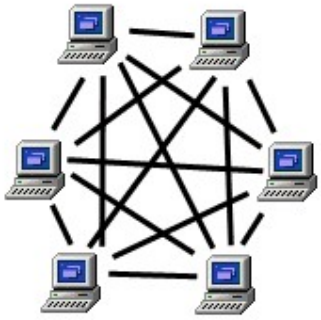
Le réseau Pair-à-pair ou poste à poste (Peer-to-peer ou P2P)

La particularité des architectures *pair-à-pair* réside dans le fait que les *échanges* peuvent se faire directement entre deux ordinateurs connectés au système*, sans transiter par un serveur central*. Il permet ainsi à tous les ordinateurs de jouer directement le rôle de *client* et de *serveur*.

On appelle souvent *nœud* les *postes connectés par un protocole réseau pair à pair*. Un *réseau pair à pair* est alors *l'interconnexion de nœuds* créés à un moment donné selon le protocole *pair-à-pair* choisi, tel le *Bluetooth*.



Client-serveur



Poste à poste

Le réseau Client-Serveur

Dans un réseau *client-serveur*, un *ordinateur central* est utilisé pour *délivrer des informations* et des *ressources* auprès des *autres ordinateurs*.

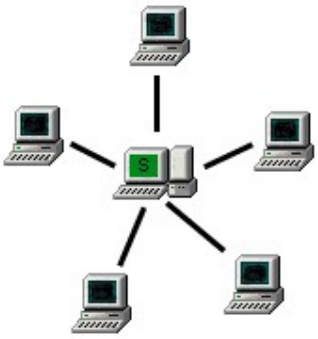
Cet *ordinateur central* est appelé *serveur* et les *autres ordinateurs* (*demandeurs de ressources)* sont appelés les clients.

Le serveur peut mettre à disposition des ressources telles que des *imprimantes*, des *services* tels qu'un *service d'authentification*, de *partage de fichiers*, etc.

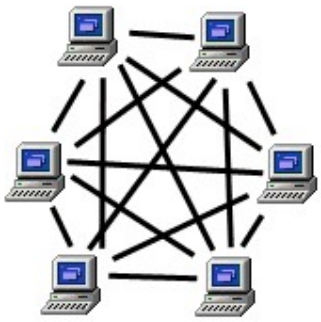
Quant au client, il s'agit simplement d'un ordinateur qui va se *contenter* d'utiliser ces *ressources* et *informations* stockées sur ce *serveur*.

Quand le client *récupère* des *ressources* vers le *serveur*, on parle de *téléchargement montant*, à l'inverse (*serveur vers le client*) de *téléchargement descendant*.

Le réseau *client-serveur* est souvent la solution la plus adaptée pour les *grandes et moyennes entreprises*.



Client-serveur



Poste à poste

Comprendre le fonctionnement des composants physiques du réseau

Comme vu précédemment, le réseau possède plusieurs (*trois*) types de *composants physiques* : les *périphériques de terminaisons*, les *équipements réseau*, ainsi que le *support*.

Nous allons dès à présent ajouter quelques précisions importantes en ce qui concerne les composants physiques du réseau.

- *La carte réseau :* cet *équipement* peut s'intégrer dans un *périphérique de terminaison* pour y fournir une *connectique réseau*.

  Il existe différents types de *cartes réseaux* : *internes (PCE, adapté pour les ordinateurs fixes)*, *USB* (pour éviter d'avoir à démonter son ordinateur), de type *PCMCIA* (plus adapté pour les ordinateurs portables).

  La *carte réseau* transforme les *données numériques* en *signaux électriques* vers le *support*, qui peut être filaire (la *carte réseau* est, dans ce cas, munie d'un *connecteur* sur lequel on branche le *câble réseau RJ45*), ou par transmission sans fil (*Wi-Fi*).

  Chaque carte dispose d'un *identifiant unique* permettant son *identification*, on parle alors *d'adresse matérielle* ou adresse *Mac* (pour *Media Access Control*).

- Le *répéteur* (*repeater*) : sa fonction principale est de *répéter le signal* tout en le *régénérant* pour *pallier l'affaiblissement du signal sur de très longues distances*.

- Le *concentrateur (hub)* permet de *concentrer* les données de *plusieurs équipements (périphériques de terminaisons)* sur un même *support*.

  En utilisant un *concentrateur (hub)*, chaque équipement attaché à celui-ci partage le même domaine de diffusion.

- Le *pont (bridge)* : pour vulgariser, il s'agit d'un *répéteur* en plus *intelligent*.

  Plus précisément, son objectif est *d'interconnecter* deux *segments de réseaux distincts physiquement* séparés à la conception pour diverses raisons (*géographique*, *extension* de site etc.), typiquement, lorsqu'on veut raccorder deux bâtiments entre eux, alors on utilise un *bridge,* et il permet de créer deux *domaines de collision distincts*.

- Le *commutateur* (*switch*) : il s'agit d'un équipement qui relie plusieurs *segments* (*câbles ou fibres*) dans un *réseau*.

  Contrairement à un *concentrateur (hub)*, un *commutateur* (*switch*) ne reproduit pas sur tous les ports chaque *trame* qu'il reçoit, il sait déterminer sur quel port il doit envoyer une *trame*, en fonction de l'adresse de destination de cette *trame*.

  Pour information, une *trame* est un *bloc d'information* véhiculée au travers d'un *support physique* (*cuivre, fibre optique*), elle se situe au *niveau 2* du *modèle OSI* (*nous verrons plus tard ce qu'est le modèle OSI*).

  *S*a caractéristique est qu'il est possible d'en reconnaître le début et la fin (*grâce à une série de bits particulière dénommée fanion*).

  Une trame est composée d'un *en-tête* (*header*), des *informations que l'on veut transmettre,* et d'un *postambule* (*trailer*).

  Un paquet (dans le cas *d'IP* par exemple) *ne peut transiter directement sur un réseau* : il est *encapsulé* à l'intérieur d'une *trame*.

- La *passerelle* (*gateway*) : Il s'agit d'un dispositif permettant de relier deux réseaux informatiques de *types différents*, par exemple un *réseau local* et *le réseau Internet*.

  Lorsque l'utilisateur d'un réseau souhaite accéder à un réseau utilisant un *protocole* différent, la *passerelle* (*gateway)* examine la légitimité de sa demande.

Si celle-ci respecte les conditions fixées par l'administrateur du réseau visé, alors la *passerelle (gateway)* établit une liaison entre les deux réseaux.

La *passerelle* joue ainsi un rôle de pare-feu et participe à la sécurisation des échanges via des protocoles réseau différents.

La plupart du temps, la *passerelle applicative* a pour mission de relier un *réseau local* à *Internet*.

Sur le plan technique, il existe diverses formes de *passerelles* : un *répéteur* est considéré comme une *passerelle* de *niveau 1*, un *pont* comme une *passerelle* de *niveau 2* et un *routeur* comme une *passerelle* de *niveau 3*.

- Le *routeur (router)* : Il a la capacité *d'examiner* les *données* transmises pour en déterminer les meilleurs chemins possibles pour la *transmission*.

  Un *routeur* permet la *communication* des *machines* sur un *réseau différent*. Prenons l'exemple d'une machine *PC1* en *192.168.1.0/24*, et d'une machine *PC2* en *192.168.2.0/24*, grâce à l'aide du routeur, le *PC 1* pourra communiquer avec le *PC 2*.

Le support

Il existe plusieurs types de *support* permettant la *transmission* de *données*.

- Le *câble coaxia*l : ce type de câble n'est quasiment plus utilisé mais il faut le connaitre.

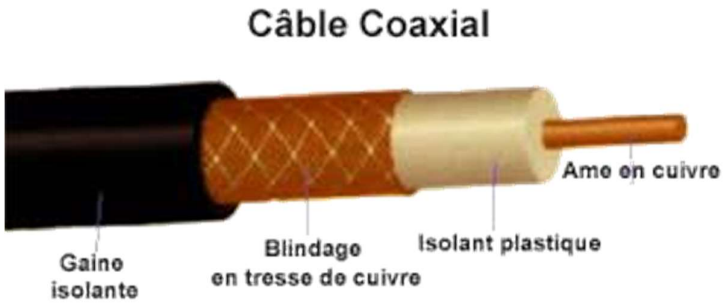
  Le *câble coaxial* est composé *d'un fil de cuivre rigide en son centre, enveloppé d'une couche plastique, elle-même entourer par une feuille ou une tresse métallique*, l'ensemble du *câble* est *recouvert d'une gaine souple en plastique*.

  La *feuille* ou la *tresse métallique* protège la *transmission* des *informations* (contre les *interférences*) mais les *signaux électriques voisins* comme les *imprimantes* peuvent *perturber* la *transmission* des *signaux électriques*.

  Au niveau des *réseaux*, il permet de *relier les ordinateurs* pour *échanger des données*. Pour connecter les ordinateurs entre eux, il faut utiliser des *connecteurs BNC en T*.

  Pour les *réseaux en bus*, il faut y ajouter un *bouchon de terminaison* aux extrémités du *câble* pour *absorber* les *signaux*.

  Le *câble coaxial* propose *un faible taux de transmission* (*50 Mbps*) et une *longueur maximale* de *50 mètres*.



- Le *câble à paire torsadée* : un câble en *paire torsadée* est constitué de plusieurs *paires de fils torsadées* entre elles, afin d'éviter les phénomènes de *parasites électriques* provenant des autres *paires torsadées* à l'intérieur du même câble.

  Ces paires torsadées sont ensuite protégées par une gaine isolante extérieure pour former le câble en lui-même.

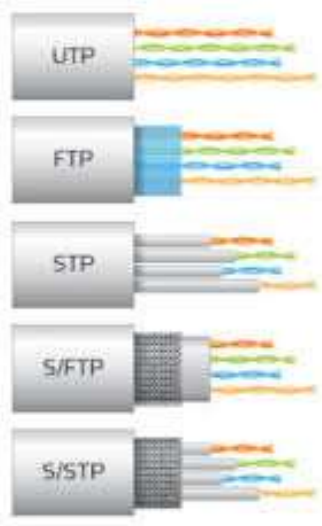
  Chaque câble contient quatre paires torsadées, il existe plusieurs types de câble permis les plus utilisés :

- La *paire torsadée non blindée* (*UTP*)
- La *paire torsadée écrantée*, ayant un écran de *protection* en *aluminium* (*FTP*)
- La *paire torsadée blindée* (*STP*)
- La *paire torsadée écrantée et blindée* (*SFTP*)

Ces différents types de câbles sont normalisés et catégorisés.

Parmi ces catégories, vous avez :

- La *catégorie 3* (avec un débit de *10 Mbps*)
- La Catégorie 4 (avec un débit de *16 Mbps*)
- La Catégorie 5 (avec un débit de *100 Mbps*)
- La Catégorie 5e (avec un débit de *1000 Mbps*)
- La Catégorie 6 (avec un débit de *10 Gbps*)
- La Catégorie 7 (avec un débit de *10 Gbps*)



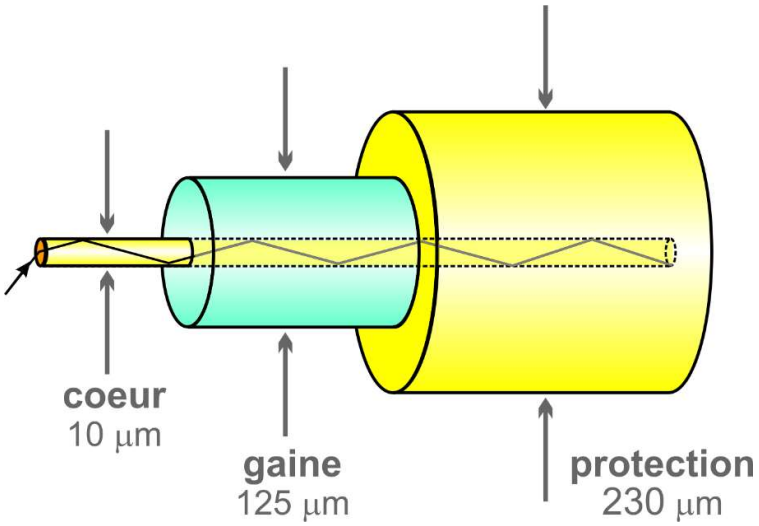
- La *fibre optique* : ces câbles transmettent un *signal lumineux modulée*.

  Il contient un *cœur* en *verre* ou en *plastique* utilisé pour *transporter* les *impulsions lumineuses*, et supporte des *débits bien plus élevés* que la *paire torsadée* et ses *signaux électriques*, en revanche, il est également beaucoup plus *fragile* que ceux-ci.

  Il existe deux *types* de *fibre* :

- La *fibre monomode* : c'est un *laser* qui est utilisé comme *source lumineuse*, le *cœur* aura un diamètre de *8 à 10 microns* (*1000 microns* sont égaux à *1 millimètre, soi en l'occurrence 0,001 millimètre pour 10 microns)*, contenant *un seul chemin direct* pour *transporter la lumière*, et ce, *jusqu'à 10 km*.

- La *fibre multimode :* c'est une *LED qui est utilisée comme source lumineuse*, le *cœur* aura un diamètre de 50 *microns* (*0,5 millimètres)*, contenant *cette fois-ci plusieurs chemins pour transporter la lumière*, ce *jusqu'à 2 km*.



µm – micromètre

- Le *sans fil* : les *signaux sans fil* sont des *ondes électromagnétiques* qui *transitent* dans l'air.

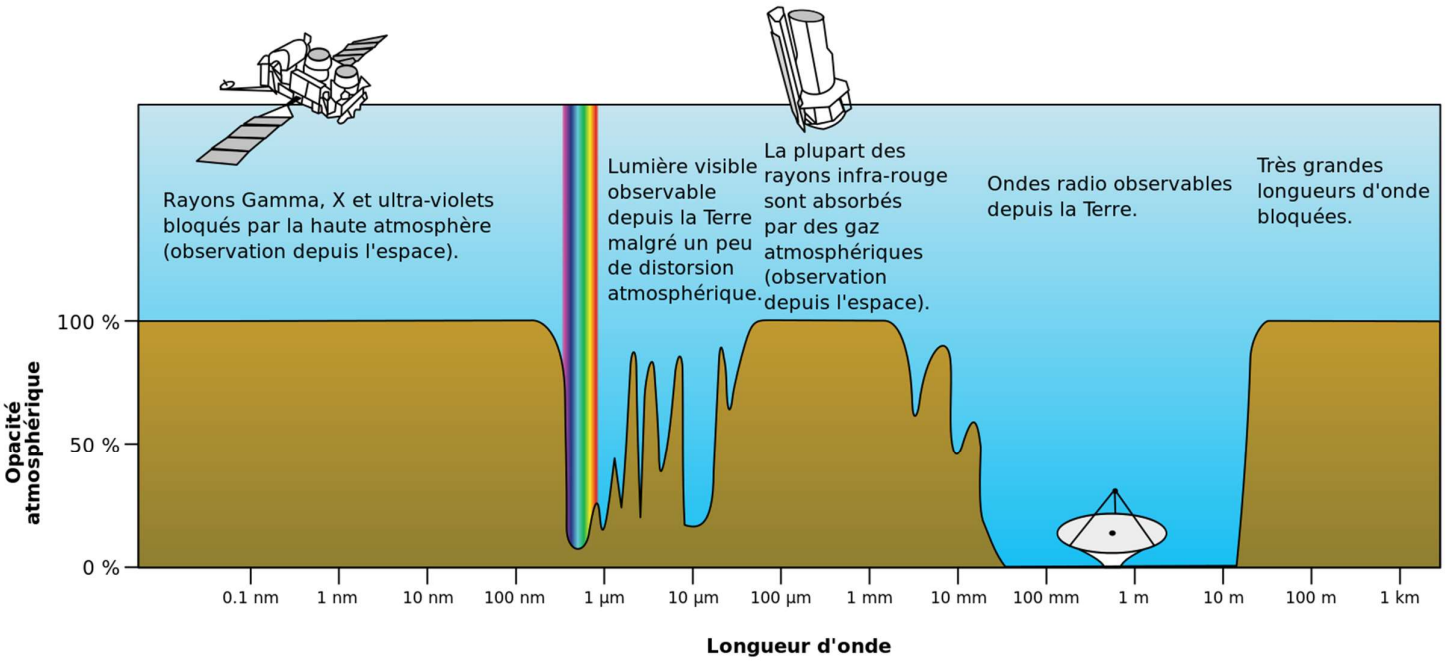
Les *réseaux sans fil* utilisent les *ondes radio* (ou *RF, pour radiofréquence*) ou des *infrarouges* (*IR*) pour *transporter* les *données*.

Une *onde radioélectrique*, abrégée en *onde radio*, est une *onde électromagnétique* dont la fréquence est *inférieure à 300 GHz* (*gigahertz*).

Comme toutes les *ondes électromagnétiques*, les *ondes radio* se *propagent* dans *l'espace vide* à la *vitesse de la lumière* et avec une *atténuation* de la puissance transportée par unité de surface *proportionnelle au carré de la distance parcourue* selon *l'équation des télécommunications*.

Dans *l'atmosphère*, elles subissent des *atténuations* liées aux *précipitations*, et peuvent être *réfléchies* ou *guidées* par la partie de la *haute atmosphère* appelée *ionosphère*.

Elles sont *atténuées* ou *déviées* par les *obstacles*, selon leur *longueur d'onde*, la nature du matériau, sa forme et sa dimension. Pour simplifier, un matériau *conducteur* aura un effet de réflexion, alors qu'un matériau *diélectrique* produira une *déviation*, et l'effet est lié au rapport entre la dimension de l'objet et la longueur d'onde.



- L'infrarouge (IR) : il s'agit d'un moyen de transmission des données sans fil, qui exploite la lumière.

  Avant l'arrivée des *technologies radio* comme le *Wi-Fi* et le *Bluetooth*, il était malgré tout possible de *transférer des données sans fil* entre deux appareils, grâce à *l'infrarouge* mais il reste cependant beaucoup moins pratique que ceux-ci car il faut que les périphériques qui communiquent entre eux soient proches l'un de l'autre (avec une portée *de 1 à 5 mètres*).

  De plus, *aucun obstacle* ne doit *séparer* les deux appareils et ils doivent être *alignés*, en effet la lumière ne se *propage* pas de la même manière que les *ondes radio*.

- Le *Wi-Fi* : un *réseau Wi-Fi* permet de relier par *ondes radio* plusieurs *appareils informatiques* (o*rdinateur, routeur, smartphone, etc.*) au sein d'un *réseau informatique* afin de permettre la *transmission de données entre eux.*

  Le *Wi-Fi* peut utiliser deux *bandes de fréquences* différentes :

- Les *ondes 2.4 gHz* : étant plus longues, elles conviendront plus à couvrir de longues distances ainsi que pour traverser les murs et les obstacles.

- Les *ondes 5 gHz* : Le *Wi-Fi 5 gHz* est capable d'aller jusqu'à 1300 Mbps (contre 600 Mbps en 2.4 gHz), en revanche, étant plus courtes, la *5 gHz* la rendent moins apte à *voyager* sur de *longues distances* et *traverser* les objets.

- Le *Bluetooth* : il s'agit d'une norme de communication permettant l'échange *bidirectionnel* (*qui peut assurer dans les deux sens la liaison entre deux éléments, comme avec le Wi-Fi*) de données à *courte distance* en utilisant des *ondes radio UHF (ultra haute fréquence)* sur la bande de fréquence de *2,4 gHz*.

Full-duplex, half-duplex et simplex

La communication entre deux systèmes électroniques peut se faire simultanément dans les deux sens, alternativement dans un sens ou l'autre, ou uniquement dans un sens, ces trois modes de communication qui portent respectivement les noms de full-duplex, half-duplex et simplex.

- *Le full-duplex*

Dans la communication *full-duplex*, deux systèmes interconnectés sont capables *d'émettre* et de *recevoir simultanément*. Outre l'existence d'un canal de transmission dédié à chaque sens de communication, ce mode de communication exige aussi que chacun des deux systèmes soit capable de traiter à la fois des données entrantes et sortantes.

Un exemple simple est le téléphone : en effet, lors d'un appel, il est tout à fait possible aux deux correspondants de parler simultanément et de s'entendre l'un l'autre. De la même manière, certains disques durs permettent de simultanément lire un fichier et en écrire un autre. Cette fonctionnalité requiert un bus de communication full-duplex comme SAS (*Serial attached SCSI*).

- Le *half-duplex*

Dans la communication *half-duplex*, deux systèmes interconnectés sont capables d'émettre et de recevoir *chacun leur tour*.

L'avantage de ce système de communication par rapport au mode *full-duplex* est qu'il réduit par deux le nombre de canaux de communication nécessaires. Par contre, il impose que les deux systèmes communicants soient en mesure de déterminer qui à le droit de parler.

Dans le cas contraire, on risque d'avoir une collision (*quand les deux systèmes tentent de parler simultanément*) ou un *blocage* (*quand les deux systèmes se mettent à l'écoute simultanément*). De plus, un délai supplémentaire peut être induit lors du basculement du sens de communication d'une direction à l'autre.

Plusieurs stratégies sont possibles pour permettre aux deux systèmes de se coordonner. Par exemple, on peut envisager un multiplexage temporel dans lequel un timing précis indique à chacun le sens de la communication.

Il peut également exister un canal de commande supplémentaire chargé d'indiquer à chaque périphérique s'il doit être en réception ou en transmission.

Dans le même ordre d'esprit, un des deux systèmes peut être par défaut en réception et l'autre en émission, l'inversion ne se faisant qu'à la demande explicite du système en émission. Cette dernière solution s'approche des notions de maître esclave ou encore de jeton.

Un exemple de ce style de communication est le télégraphe Morse : celui-ci est constitué par une ligne électrique munie à chacune de ses extrémités d'un émetteur-récepteur. Quand un opérateur tape un message sur son manipulateur, les impulsions électriques sont transmises sur la ligne et produisent un son et/ou une transcription sur papier à l'autre extrémité. Comme c'est la même ligne qui achemine les signaux dans les deux sens, il s'agit bien d'un système half-duplex dans lequel les opérateurs doivent se coordonner pour ne pas transmettre simultanément.

- Le *Simplex*

La communication *simplex* est un mode de communication *unidirectionnel*, dans lequel chaque appareil est soit toujours émetteur soit toujours récepteur.

Ce mode de communication est notamment utilisé quand il n'est pas nécessaire pour l'émetteur d'obtenir une réponse de la part du récepteur. Un circuit électronique comme un capteur qui envoie régulièrement et de manière autonome des données pourra utiliser une liaison *simplex*.

C'est aussi un mode de communication utilisé pour la diffusion, c'est à dire lorsqu'un même émetteur transmet simultanément à de nombreux récepteurs. Ainsi, la liaison entre un émetteur de télévision et les postes récepteurs est une liaison simplex.

Pour terminer sur un exemple encore plus simple, la télécommande de votre téléviseur communique avec ce dernier par une liaison *simplex* : quand vous pressez sur un bouton pour changer de chaîne, un train de signaux infrarouge est émis par la télécommande. Mais celle-ci est incapable de savoir si l'ordre a bien été reçu par le téléviseur ou pas. Cette solution est retenue car elle simplifie la conception du système et en réduit les coûts.

Par ailleurs, aucun retour de type *accusé de réception* n'est nécessaire pour cette application : en effet, l'utilisateur est parfaitement en mesure de déterminer si la transmission s'est bien passée ou pas, et le cas échéant de réappuyer sur le bouton.

Les *unités de mesure*

- Les *unités de taille de mémoire*

Les *unités de mesure* suivantes sont utilisées en informatique pour quantifier la taille de la mémoire d'un dispositif numérique, l'espace utilisable sur un disque dur, une clé USB, la taille d'un fichier, d'un répertoire, etc.

- *Bit* : il s'agit de l'unité de mesure de base (à ne pas confondre avec byte). Son symbole est *b* ou *bit*. Le *bit* (pour *binary digit*, ou *chiffre binaire* en français) représente la *plus petite unité de mémoire* utilisable sur un ordinateur.

  Cette *mémoire* ne peut prendre que *deux valeurs*, la plupart du temps interprétées comme *0* ou *1*.

  Les autres *unités de mesure* ne correspondent qu'à des *regroupements de bits*.

- *Octet* : un *octet* est composé de *huit bits*. Son symbole est *o*.

  Un *octet* permet de *coder des valeurs numériques* ou *jusqu'à 256 caractères différents*.

- *Quartet* ou *nibble* : la moitié d'un *octet*, soit quatre *bits*.

- *Septet* : sept *bits*.

- *Seizet* ou *doublet* : *seize bits*.

- *Trente-deuzet* ou *quadlet* : *trente-deux bits*.

- *Octlet* : *soixante-quatre bits*

- Les *unités de débit*

Ces *unités de mesure* servent à *mesurer un débit* (*maximal*) d'information *entre deux points d'un réseau informatique*. Le *débit binaire* est une mesure de la quantité de *données numériques* transmises par *unité de temps*.

Selon ses définitions normatives, il s'exprime en *bits par seconde* (*bit/s*, *b/s* ou *bps*) ou un de ses multiples en employant les préfixes : *kb/s* (*kilobits par seconde*), *Mb/s* (*mégabits par seconde*) et ainsi de suite.

- *Bit (par seconde)* : ses *notations* sont *bit/s, b/s ou bps* et sa *valeur* de *1 bit/s.*

- *Kilobit (par seconde)* : ses *notations* sont *Kbit/s* ou *Kb/s* et sa *valeur* de *10 puissance 3 bit/s* (*1 Kb/s équivaut à 1000 bits*).

- *Mégabit (par seconde)* : ses *notations* sont *Mbits/s* ou *Mb/s* et sa *valeur* de *10 puissance 6 bit/s* (*1 Mb/s* équivaut à *1000 Kb/s*).

- *Gigabit (par seconde)* : ses notations sont *Gbit/s* ou *Gb/s* et sa valeur de *10 puissance 9 bit/s* (*1 Gb/s* équivaut à *1000 Mb/s*).

- *Térabit* (*par seconde*) : ses notations sont *Tbit/s* ou *Tb/s* et sa valeur de *10 puissance 12 bit/s* (*1 Tb/s* équivaut à *1000 Gb/s*).

Le binaire

Il est indispensable, à minima de *connaitre* et *comprendre* le *système numérique binaire*, qui est en complète adéquation avec notre ordinateur, et être capable de réaliser par exemple une *conversion* d'un *nombre binaire* en *décimal*.

En *informatique* et en *électronique*, vous n'avez que deux états possibles ; *je n'ai pas d'électricité* ; *0*, *j'en ai* ; *1*.

Je vais vous dévoiler la méthode que j'utilise personnellement pour convertir un nombre *binaire* en *décimal*.

Admettons que l'on souhaite convertir « *1000000* » en décimal, tout d'abord, il faut savoir que le calcul s'effectuera de droite, à gauche. Nous allons (au-dessus du nombre à convertir) y implémenter un tableau avec différentes valeurs.

128,64,32,16,8,4,2,1

(Le tableau, avec les valeurs en question)

1, 1, 0, 0, 0, 0, 0, 0

(Le nombre binaire)

Comme vu précédemment*, le 1 signifie un état actif, le 0 un état inactif, en partant de ce postulat, quels sont les états actifs et inactifs dans ce nombre (en partant de la droite) ?*

***1** (actif)**, 1** (actif)**, 0** (inactif)**, 0** (inactif)**, 0** (inactif)**, 0** (inactif)**,* ***0**(inactif)**, 0**(inactif).*

Actif, actif, inactif, inactif, inactif, inactif, inactif, inactif.

Dès à présent, *transmettons* ces résultats *dans notre tableau.*

Actif, actif, inactif, inactif, inactif, inactif, inactif, inactif.

128 *(actif),* ***64*** *(actif),* ***32*** *(inactif),* ***16*** *(inactif),* ***8*** *(inactif),* ***4*** *(inactif),* ***2*** *(inactif),* ***1*** *(inactif).*

Nous n'avons plus qu'à *additionner nos actifs*, ce qui va nous donner un *total* de *: 192 (64+128), les reste étant inactif (0), nous ne les avons pas ajoutés dans notre calcul.*

1000 en *binaire* donnera ***8*** en *décimal.*

Actif (1), inactif (0), inactif (0), inactif (0)

8 0 0 0

0011 en *binaire* donnera ***3*** en *décimal.*

Inactif (0), inactif (0), actif (1), actif (1)

0 0 1 1

Pourquoi spécifiquement « *1, 2, 4, 8, 16, 32, 64, 128* » ? En partant de 1, il faut tout simplement additionner son propre chiffre.

1+1=2, 2+2=4, 4+4=8, 8+8=16, 16+16=32.

Nous avons pu effectuer le calcul car jusqu'ici, le *nombre binaire* à convertir était *inférieur* ou *égal* aux *chiffres* de notre *tableau* (*huit bits*, *un octet*), mais nous pouvons élargir notre tableau autant que nous en avons le besoin. *Comment convertir ce nombre binaire de 16 bits (2 octets, ou 1 seizet ou doublet) :*

1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0

32 768, 16 384, 8192, 4096, 2048, 1024, 512, 256, 128, 64, 32, 16, 8, 4, 2, 1
Actif (1), actif (1), actif (1), actif (1), actif (1), actif (1), actif (1), actif (1), actif (1), inactif (0), actif (1), inactif (0), actif (1), inactif (0), actif (1), inactif (0).

Ce qui nous donne donc : *65450* en *décimal.*

- *Le système hexadécimal* : celui-ci se compose de *16 symboles* représentant les *16 premiers entiers naturels*, appelés *chiffres hexadécimaux* : *0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E et F.*

Cette fois-ci, même si le début du calcul s'effectuera toujours *de droite à gauche*, *la méthode de calcul* et le *tableau* seront *légèrement différents.*

Prenons l'exemple du nombre *binaire* « ***10000010*** », pour commencer, il faudra diviser les nombres par *groupes de 2 octets ou quartet* (*quatre par quatre*) et utiliser un *tableau* allant jusqu'à *8* par groupes.

8, 4, 2, 1 / 8, 4, 2, 1

(À droite, le premier tableau pour le premier groupe de quartet, à gauche le deuxième tableau pour le deuxième groupe).

1 (actif), 0 (inactif), 0 (inactif) , 0 (inactif) / 0 (inactif), 0 (inactif), 1 (actif), 0 (inactif)

À chaque *1* (*actif*), il s'agit d'une *activation* en quelque sorte, et si vous *activer* le chiffre 2 et 4 (*0110*) par exemple, vous devrez *calculer la somme de ces deux nombres, 6*, puis faire pareil, pour le deuxième *quartet*.

En l'occurrence, vous avez donc 8 à gauche (1000), puis 2 à droite (0010), il ne vous reste plus qu'à écrire le *premier chiffre (8)*, *coller au deuxième (2)*, ce qui vous donne donc, *82*. ***10000010*** *signifie 82 en hexa*décimal.

Petite particularité, comme vu précédemment l'hexa va de 1 à 9, puis de A à F, alors pour 101010111000 par exemple, déjà puisqu'il y a 3 octets il faudra y faire trois tableaux (*d'un quartet*), et ensuite, *à partir de 10* en *hexa*, il faudra y mettre un *A*. À partir de 16, nous repartons à 10. Pour 32, 20.

101010111000 signifie donc *AB8* (*car 10+11+8)*.

Le modèle OSI

Pour *qu'un ensemble d'équipements reliés entre eux* puissent *s'échanger des informations*, il faut qu'ils puissent *se comprendre*.

Qu'ils puissent *se comprendre* ?
Un *Japonais* ou un *Arménien* ne pourra pas comprendre un *Français* puisqu'ils ne parlent pas la même langue (*qui ont tous deux leurs propres règles*). Les règles de la *communication* en *France* se basent sur les *règles* de *la langue Française, définies et gérés* par des *organismes* tels que *l'Académie Française*.

En l'occurrence, les règles du langage commun de la communication en réseau sont définies par le *modèle OSI (*Open Systems Interconnection ou Interconnexion de systèmes ouverts).

Il s'agit tout simplement de *la norme de communication en réseau* de tous les systèmes informatiques, *définie et géré par l'ISO (International Organization for Standardization ou Organisation internationale de normalisation* qui est un *organisme de* normalisation international*)*. Pour reprendre mon analogie, les *Français* parlent *Français*, les *Japonais* en *Japonais*, les *systèmes informatiques en réseau* le *modèle OSI*.

Repartons sur une nouvelle analogie, de quoi avez-vous besoin, *physiquement*, pour communiquer avec une autre personne ? D'une *bouche* (un *émetteur*), des *oreilles* (un *récepteur*), *l'air* (un *support de transmission*), mais ce qui est intéressant, c'est que vous pouvez également utiliser *différents intermédiaires* par exemple pour *communiquer* avec une autre personne *à distance*, *l'intermédiaire* étant par exemple le *livreur*, qui *transportera* votre message dans une *enveloppe* ou un *colis* (un *contenant*).

Le *modèle OSI* est lui décomposé en sept couches, en partant de la *couche 1* avec la *couche Physique, Liaison, Réseau, Transport, Session, Présentation,* jusqu'à la *couche 7* étant la *couche Application.*

En plus de vous expliquer le *principe* de ces *différentes couches*, je vais également vous donner le *PDU* (*Protocol Data Unit* ou *Unité de données de protocole*) de chacune de ces couches.

Il s'agit de *l'unité de mesure* des informations échangées dans un réseau informatique. Il est également bon de rappeler que le *modèle OSI* est un modèle *théorique* utilisé principalement pour la *compréhension* du *réseau*, en pratique c'est le modèle *TCP/IP* qui est implémenté.

- La *couche 7* où couche *Application* est là pour représenter les applications pour lesquelles nous allons mettre en œuvre des *communications*. Cette couche est le *point d'accès* pour l'utilisateur aux *services applicatifs* comme *Firefox, Thunderbird, FileZilla*, etc.

- Le rôle de la *couche 6* où couche *Présentation encode, compresse, convertit* et *reformate* les données. Il s'agit en quelque sorte de *l'affichage et la mise en forme des données*.

- Le rôle de la *couche 5* où *couche Session* est de *gérer l'organisation des échanges,* elle gère *l'ouverture et la fermeture des sessions d'échanges*.

- La *couche 4* où couche *Transport* choisit la meilleure façon d'envoyer une information. Il existe deux *protocoles* dans cette couche, *TCP* et *UDP*. UDP envoie des informations *sans garantie de réception*, à l'inverse *TCP* permet de *fiabiliser* les *communications*. Le *PDU* de la couche transport est le segment pour *TCP*, *datagramme* pour *UDP,* ou encore *paquet*.

- Le rôle de la *couche 3* où couche *Réseau* est d'assurer le *routage* des paquets d'un point *A* à un point *B,* et le *protocole* le plus utilisé et le *protocole IP,* c'est à ce niveau-là qu'il y a la *gestion des adressages IP*.

- Le rôle de la *couche 2* où couche Liaison (*de données*) est de *définir la transmission de données* entre *deux machines* du *même réseau* (le *matériel associé* étant : le *switch*). Le *PDU* de la couche *liaison* est la *trame*.

- *La couche 1* où *couche Physique* est en charge de la *connexion physique* sur le *réseau* pour *l'émission* et la *réception* de *bits* sur le *réseau* (*câbles, fibre optique, wi-fi*). Le *PDU* de la couche *physique* est le *bit*.

Le *modèle TCP/IP*

Le problème du *modèle OSI* est qu'il est trop segmenté car en réalité, certaines couches peuvent faire le travail d'autres couches, et donc comme vu précédemment, *en pratique* c'est le modèle *TCP/IP* qui est implémenté au sein des machines, mais alors concrètement, comment se passe une *communication* dans le *modèle TCP/IP* ?

Le *modèle TCP/IP* simplifie le *modèle OSI* en *4 couches*.

- La *couche 4* du *modèle TCP/IP* est la couche *Application*.

Il s'agit de *la couche de remplacement* de *la couche 7, 6 et 5* du *modèle OSI (Application, Présentation et Session)*.

Comme son nom l'indique, il s'agit des applications, l'utilisation de *protocoles* tels que *DHCP, DNS, FTP, HTTP, SSH, SMTP, etc*.

- La *couche 3* du *modèle TCP/IP* est la couche *Transport*.
  Elle utilise principalement le *protocole TCP*, cette couche Transport fonctionne à peu près de la même façon que celle du *modèle OSI*.
  Il *fragmente* le message à transmettre de manière à pouvoir le faire passer sur la couche *Internet*, a l'inverse, *TCP* replace dans l'ordre les *fragments transmis* sur la couche *internet* pour *retransmettre le message initial.*

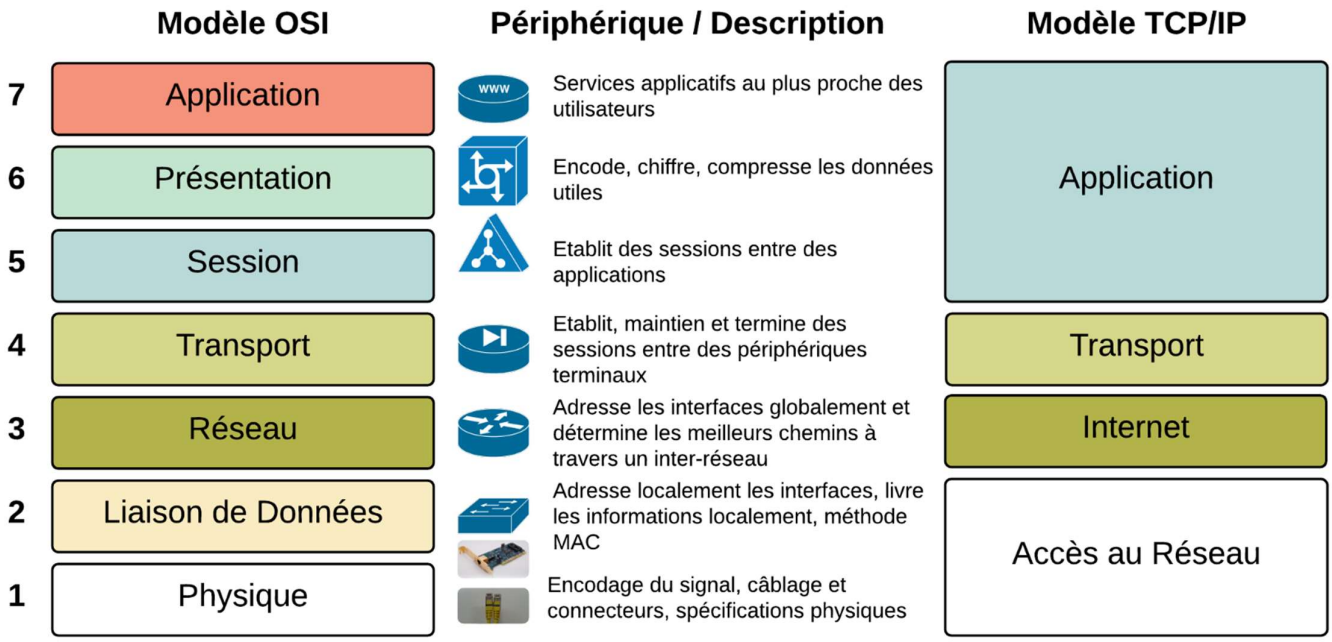
- La *couche 2* du *modèle TCP/IP* est la *couche Internet* et il s'agit du remplaçant de la couche 3 du *modèle OSI* (*Réseau*). Le protocole le plus utilisé de la couche *Internet* est appelé *IP* (*Internet Protocol*). Cette couche gère le *routage* et *IP* assure *l'acheminement des paquets depuis une source vers une destination*, cette couche *ajoute des informations* comme *l'adresse IP de l'expéditeur*, du *destinataire,* et *contrôle l'acheminement du message.*

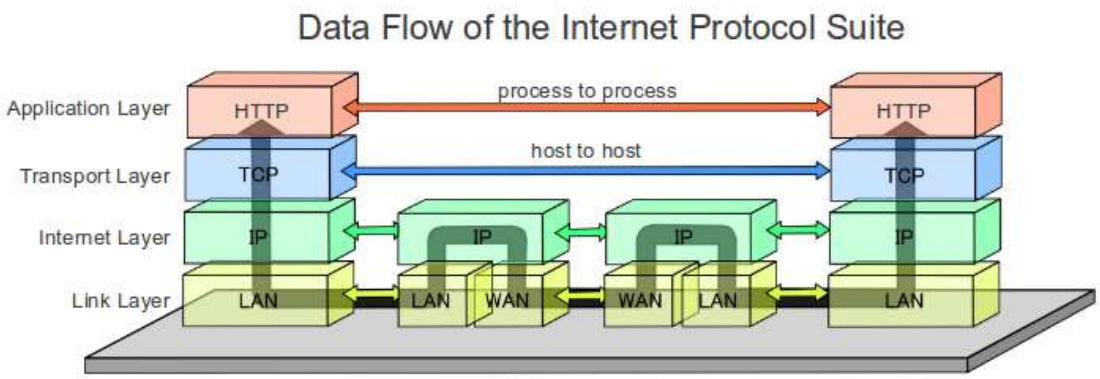
- La *couche 1* du *modèle TCP/IP* est la couche *Accès au réseau*.
  Il s'agit de la couche de *remplacement* de la *couche 1 et 2* du *modèle OSI* (*Physique* et *Liaison*). Cette couche s'occupe de la connexion entre deux machines.

En *émission*, les données traversent chacune des couches, et à chaque couche une information de données est ajoutée au paquet de données, il s'agit d'un entête.

L'entête définit le *protocole* utilisé dans chaque couche, on parle alors *d'encapsulation*. Tout cela pour ensuite être transformé en *bits* pour être transmis sur l'autre machine *via le réseau*.

Au niveau de la *machine réceptrice*, lors du passage dans chacune des couches, *l'entête est lu, interprété puis supprimé*, puis *arrivé en bout de chaine à la couche application* la *donnée* est dans son *état de départ*, on parle alors de désencapsulation.





La *topologie réseau*

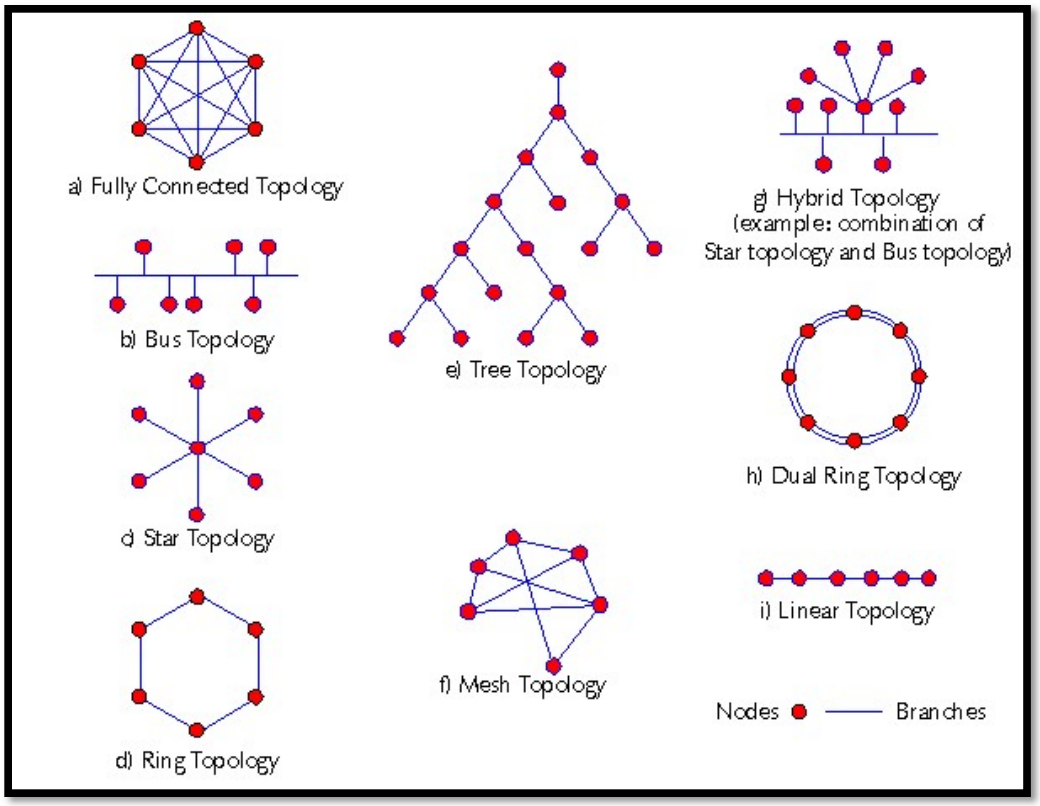
La manière dont les machines sont *interconnectées* et *la méthode utilisée* pour *transmettre les données* est appelée *topologie.*

Il existe *deux types* de *topologies* :

- La *topologie physique :* elle décrit la disposition *physique* d'un *réseau* (des *équipements* et des *supports* de *transmission* de *données*).

Les *<u>principales</u> topologies physiques* sont (il en existe d'autres, mais *il ne s'agit que d'une introduction,* comprenant les *notions fondamentales)* :

- Le *réseau en bus.*
- Le *réseau en étoile.*
- Le *réseau en anneau.*
- Le *réseau maillé.*

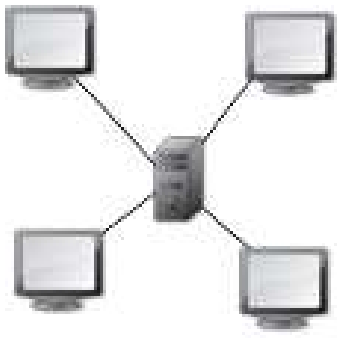


- *Le réseau en bus : les réseaux en bus permettent de relier simplement de multiples clients, mais posent des problèmes de collision quand deux clients veulent transmettre des données au même moment sur le bus.*

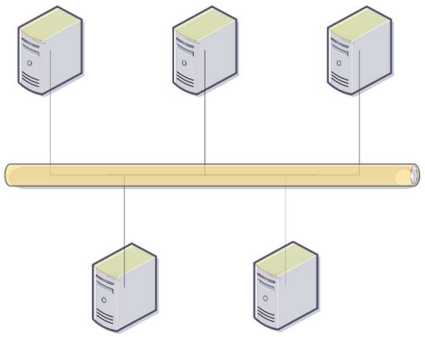
L'information est *diffusée sur tout le réseau*, ce qui lui apporte un *faible niveau de sécurité* des données transitant sur le réseau (*toutes les stations connectées au bus peuvent lire toutes les données transmises sur le bus*). De plus elle est extrêmement vulnérable car si l'une des connexions est défectueuse, c'est l'ensemble du réseau qui en est affecté. Un autre problème (en mettant déjà de côté tous ces problèmes), est qu'un câble coupé peut tout simplement interrompre le réseau.

Mais *rassurez-vous*, ce type de *réseau (topologie)* n'est quasiment plus utilisé(e) de nos jours. Son *seul véritable avantage* est probablement qu'il présente l'un des coûts de mise en œuvre *le plus bas.*

- La *topologie en étoile* : aussi appelé « *Hub and spoke* », dans cette *topologie*, les *équipements du réseau* sont *reliés à un système matériel central* (le *nœud*) qui a pour *rôle d'assurer la communication* entre les différents *équipements du réseau*. En pratique, *l'équipement central* peut être un *concentrateur* (*hub*) un *commutateur* (*switch*) ou un *routeur*.



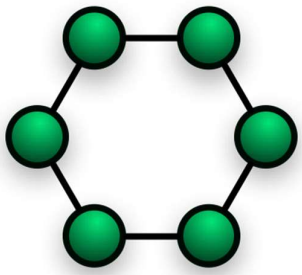
Topologie en étoile.



Topologie en bus.

- Le *réseau en anneau* : dans cette *topologie*, *chaque machine* est *connectée à la suivante*, l'information circule *dans un seul sens* et les machines reçoivent les données *tour par tour*. Toutes les entités sont *reliées entre elles* dans une *boucle fermée*.

Une entité n'accepte une donnée en circulation sur l'anneau *que si elle correspond bien à son adresse*. Dans le cas contraire, l'entité en question fait passer la donnée à l'entité suivante. Ce type de topologie utilise le plus souvent le principe de l'anneau à jeton (cf. *Wikipédia*).



Topologie en anneau

- Topologie maillée (*mesh*) : il s'agit d'une topologie (*filaire et sans fil*) où *tous les hôtes sont connectés pair à pair sans hiérarchie centrale*, formant ainsi une structure en forme de *filet*, l'avantage ? Aucun risque de panne générale si une machine tombe en panne. *Cependant*, imaginez une *entreprise* possédant *500 machines*, *124750* câbles seront nécessaires pour tous les relier ensemble.

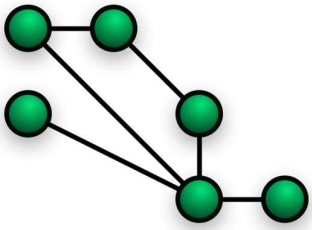
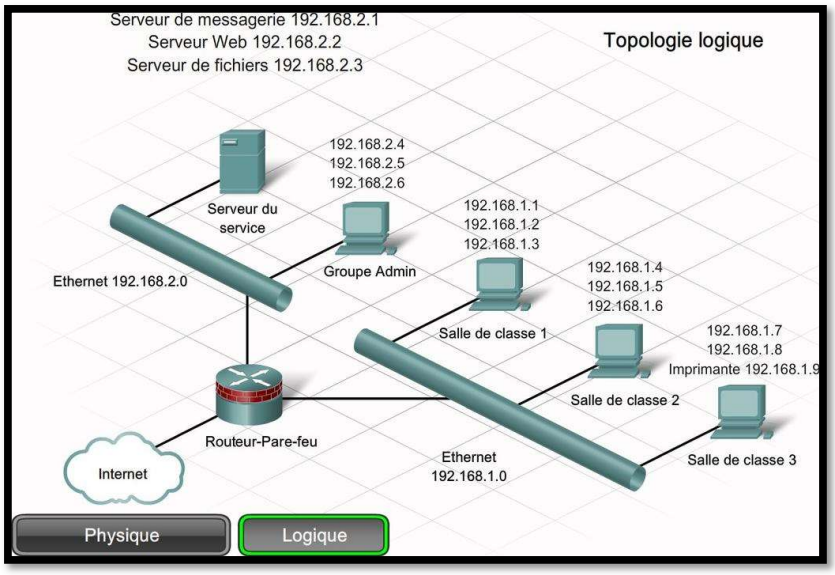


Illustration d'un réseau maillé partiellement connecté. Dans un réseau maillé entièrement connecté, chaque nœud est connecté à tous les autres nœuds du réseau.

La topologie logique : outre le type de *topologie physique*, il existe également la *topologie logique* qui définit la manière dont les données sont véhiculées sur le réseau.

Pour rappel, une *topologie physique* peut définir la façon dont les *équipements* sont *interconnectés* et la *représentation spatiale du réseau*. En ce qui concerne la *topologie logique*, elle peut définir la façon dont les données transitent dans les lignes de communication, comment se passe la communication dans la topologie physique.

Pour résumer (et pour être sûr que vous avez bien compris la différence entre ces deux types), l'une (*topologie physique*) définit la *structure physique* (*l'apparence physique, la forme*) de votre *réseau*, l'autre (*topologie logique*) définit comment la communication se passe dans cette *forme physique*.



Le *protocole IPv4*

Une *adresse IPv4* est représentée sous la forme de quatre nombres entiers séparés par des points comme *192.168.0.1*, chacun des nombres représente un *octet*.

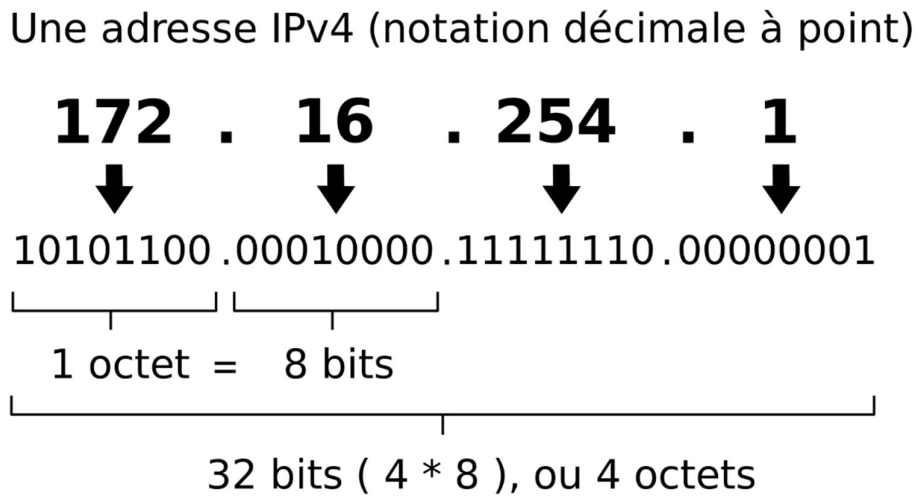
La plage d'attribution s'étend de *0.0.0.0* à *255.255.255.255*, sachant qu'il existe des contraintes empêchant l'utilisation de certaines adresses (*réservée, masque, broadcast, etc.*).

IPv4 (*Internet Protocol version 4*) est la *première version d'Internet Protocol* (*IP*) à avoir été largement déployée, et qui forme encore en *2020* la base de *la majorité* des communications sur Internet, *par rapport à l'IPv6*.

Une adresse *IPv4* est codée sur *32 bits (trente-deuzet* ou *quadlet)*, et est capable de fournir jusqu'à *4 294 967 296* (soit *2^32*) adresses.

À ce jour, l'ensemble des adresses IPv4 disponibles (*sur Internet*) est *épuisé*, cet épuisement a conduit au développement d'une nouvelle version *d'IP*, *IPv6*, et à la transition *d'IPv4* vers *IPv6* afin d'adopter *cette nouvelle version*.

Une adresse *IPv6* est *codée* sur 128 bits (au lieu de *32 bits* avec *IPv4*), avec une telle marge, on pourrait attribuer *plus de 660 millions d'adresses IP à chaque millimètre carré de la planète Terre.*



Une adresse IPv4 permet d'identifier une machine sur le réseau.

Pour faire une analogie, il s'agit de son "*adresse complète*" (il faut cependant connaitre le masque de sous-réseau pour différencier la partie hôte de la partie réseau, nous en reparlerons plus tard, cf. le masque de sous-réseau), je m'explique :

J'ai rendez-vous avec une femme, qui m'a donné son adresse pour que je puisse la rejoindre, il s'agit du "*141 Boulevard Mortier*".

Je vais commencer par me rendre au *Boulevard Mortier*, puis chercher le numéro *141*.

Tout comme une adresse physique est composée d'une partie "*générale*" *(nom de la rue)* et d'une partie "*personnelle*", une adresse *IPv4* est également composé d'une partie "générale" pour le réseau, et d'une autre partie pour l'hôte.

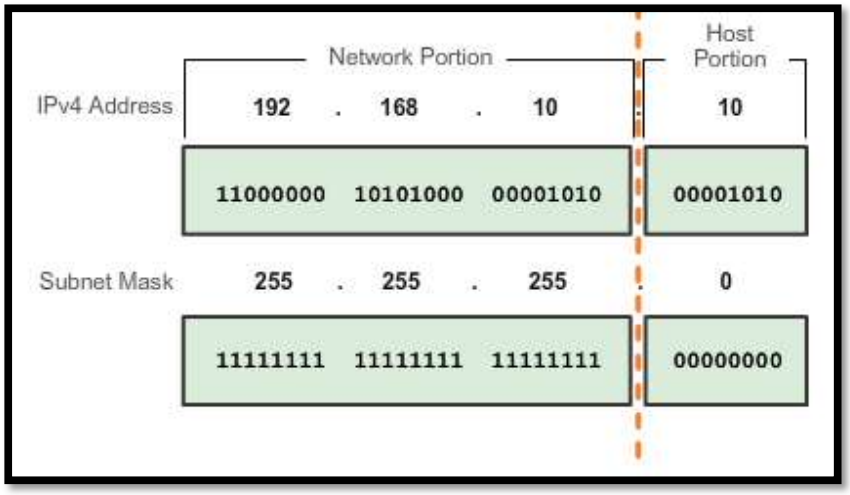
Exemple concret :

192.168.0.1/24 (si vous ne comprenez pas le /24, ce n'est pas un problème nous en parlerons plus tard, il s'agit du principe de CIDR).

192.168.0 = la partie réseau, par analogie avec l'adresse physique, il s'agit du *Boulevard Mortier*.

.1 = la partie hôte, par analogie, il s'agit du *141*.

192.168.0 [*Boulevard Mortier*] *.1* [*141*].



- Les classes d'adresses

À l'origine *d'IPv4*, on distingue une *organisation* en *classes d'adresses* dont *les quatre premiers bits indiquent la classe*.

- Les adresses de *Classe A* commencent par *oxxx* en *binaire*, ou *0 à 127 en décimal.*
- Les adresses de *Classe B* commencent par *10xx* en *binaire*, ou *128 à 191 en décimal.*
- Les adresses de *Classe C* commencent par *110x* en *binaire*, ou *192 à 223 en décimal.*
- Les adresses de *Classe D* commencent par *1110* en *binaire*, ou *224 à 239 en décimal.*
- Les adresses de *Classe E* commencent par *1111* en *binaire*, ou *240 à 255 en décimal.*

Seules les adresses de Classes *A, B et C* sont assignables à des interfaces (*adresse d'Unicast,* on entend par *unicast* le fait de *communiquer* entre *deux ordinateurs identifiés* chacun par une *adresse réseau unique*).

La *classe D* est utilisée pour des adresses de *Multicast* (*adresse unique identifiant de nombreuses destinations*)

La *classe E* est utilisée pour des besoins futurs ou des *objectifs scientifiques.*

Adresses spécifiques : les adresses commençant par *127.0.0.0* à *127.255.255.255* sont *réservées* pour le *bouclage* (*loopback*).

Une *loopback* est une interface virtuelle d'un matériel réseau (*ou d'un ordinateur*), et par extension *une adresse associée à cette interface*.

Ainsi, quand il la contacte il "*boucle*" sur lui-même. Le but étant de renvoyer un signal reçu vers son envoyeur sans modification ni traitement, et qui peut par exemple être utilisé à des fins de tests.

Adresses privées non routables vers *l'Internet* sont (*RFC1918*) :

Pour la *classe A* : de 10.0.0.0 à 10.255.255.255

Pour la *classe B* : de *172.16.0.0 à 172.31.255.255*

Pour la *classe C* : de *192.168.0.0 à 192.168.255.255*

Le protocole IPv6

IPv6 (Internet Protocol version 6) a été principalement développé en réponse à la demande d'adresses *qu'IPv4* ne permettait plus de contenter. Le développement rapide *d'Internet* a conduit à la pénurie du nombre d'adresses *IPv4* disponibles.

Pour rappel, une adresse IPv4 contient 32 bits, en l'occurrence, une adresse IPv6 contient 128 bits, pour vous donner une idée de la différence, avec une telle marge, on pourrait attribuer *plus de 660 millions d'adresses IP* à chaque *millimètre carré* de la *planète Terre*.

Plus précisément, on dispose ainsi de *2 ^ 128 ≈ 3,4 × 10 ^ 38 = 340 sextillions* d'adresses *IPv6*, contre *2 ^ 32 ≈ 4 milliards d'adresses IPv4*.

340 282 366 920 938 463 463 374 607 431 768 211 456 adresses *IPv6* possibles, contre *4 294 967 296 IPv4* possibles (*environ 4 milliards*).

La notation décimale pointée employée pour les adresses *IPv4* (par exemple *172.31.128.1*) est abandonnée au profit d'une écriture *hexadécimale*, où les 8 groupes de *2 octets* (soit *16 bits* par groupe) sont séparés par un signe deux-points, « : ».

2001:0db8:0000:85a3:0000:0000:ac1f:8001

La notation complète ci-dessus comprend exactement *39 caractères*.

Il peut exister plusieurs façons différentes de représenter une adresse IPv6. La *RFC59522* définit une *représentation canonique*.

L’adresse MAC

Une *adresse MAC* (*Media Access Control*), parfois nommée *adresse physique*, est un *identifiant physique* stocké dans une *carte réseau* ou une *interface réseau* similaire.

À moins qu'elle n'ait été modifiée par l'utilisateur, elle est *unique* au monde. *Toutes les cartes réseau* ont une *adresse MAC*, qui ne se trouvent pas uniquement sur un *ordinateur*, mais sur tous types d'appareils *connectés* tels que les *tablettes tactiles*, *smartphones*, *consoles (PS3, Wii, etc.)*, *réfrigérateurs* et montres connectés, etc.

Une *adresse MAC* est constituée de *48 bits* (*6 octets*) et est généralement représentée sous la forme *hexadécimale* en séparant les *octets* par un *double point*. Par exemple *5E:FF:56:A2:AF:15*.

Ces *48* bits sont répartis de la façon suivante :

- 1 bit I/G : indique si l'adresse est individuelle, auquel cas le bit sera à 0 (pour une machine unique, unicast) ou de groupe (multicast ou broadcast), en passant le bit à 1.
- 1 bit U/L : indique 0 si l'adresse est universelle (conforme au format de l'IEEE) ou locale, 1 pour une adresse administrée localement.
- 22 bits réservés : tous les bits sont à zéro pour une adresse locale, sinon ils contiennent l'adresse du constructeur.
- 24 bits : adresse unique (pour différencier les différentes cartes réseaux d'un même constructeur).

MAC constitue la partie *inférieure* de la couche de *liaison* (*OSI*). Elle *insère* et *traite* ces adresses au sein des *trames transmises*. Elle est parfois appelée adresse *ethernet*, *UAA* (*Universally Administered Address*), *BIA* (*Burned-In Address*), *MAC-48 ou EUI-48.*

- Adresses particulières :
FF:FF:FF:FF:FF:FF - Adresse broadcast
01:00:0C:CC:CC:CC - Cisco Discovery Protocol (CDP)
01:80:C2:00:00:00 - Spanning Tree Protocol
33:33:xx:xx:xx:xx - Adresses multicast IPv6
01:00:5E:xx:xx:xx - Adresses multicast IPv4
00:00:0c:07:ac:xx - Adresses HSRP
00:00:5E:00:01:XX - Adresses VRRP

Le *masque de sous-réseau*

Le masque de sous-réseau est un séparateur entre la partie réseau et la partie machine d'une adresse IP.

Un *masque de sous-réseau* utilise la même représentation que celles des *adresses IPv4*.

Comme vu précédemment, une adresse *IPv4* est codée sur *4 octets*, soit *32 bits* (*représentés en notation décimale à point*). Un *masque de sous-réseau* possède lui aussi *4 octets*.

Une adresse IPv4 comporte une partie identifiant le réseau et une autre partie l'hôte, et bien le masque de sous-réseau est un masque distinguant les bits d'une adresse IPv4 utilisés pour identifier le sous-réseau de ceux utilisés pour identifier l'hôte.

L'adresse du réseau est obtenue en appliquant l'opérateur ET logique entre l'adresse (binaire) IPv4 et le masque de sous-réseau.

Prenons l'exemple d'une adresse IPv4 en 192.168.0.1, avec un masque de sous-réseau en 255.255.255.0.

La séparation se fait au *quatrième octet*, les *trois premiers octets* sont la *partie réseau*, le *dernier octet,* celle de la *machine* (*l'hôte*), en l'occurrence on peut dire que sur le *réseau 192.168.0*, ma machine à l'adresse *1*.

Avant de partir dans de nouvelles explications, je vais commencer par vous montrer une méthode pour retrouver *l'adresse du réseau* sur laquelle je me trouve, à partir de *l'adresse IPv4* d'une *machine quelconque* et de son *masque de sous-réseau*.

Commençons par convertir notre adresse *IPv4* ainsi que notre *masque de sous-réseau* en *binaire*, et plaçons les l'un en dessous de l'autre (*192.168.0.1 ET 255.255.255.0*).

11000000.10101000.00000000.00000001 ET

11111111.11111111.11111111.00000000

Effectuons un *ET logique* (cf. *logique combinatoire*) entre *l'adresse IPv4* et le *masque de sous-réseau*, ce qui nous donne donc :

1100000000.10101000.00000000.00000000

Cela nous donne donc l'adresse du réseau (en décimal) :

192.168.0.0

Autre exemple, pour une adresse 58.39.104.225/24 :

00111010.00100111.01101000.11100001 ET

11111111.11111111.11111111.00000000

Effectuons un *ET logique* (cf. *logique combinatoire*) entre *l'adresse IPv4* et le *masque de sous-réseau*, ce qui nous donne donc :

00111010.00100111.01101000.00000000

Cela nous donne donc l'adresse du réseau (en décimal) :

58.39.104.0

- *Les classes*

Même si en pratique la notion de classe n'est plus utilisée, il faut les connaitre. Comme vu précédemment, il existe différentes classes d'adresses (*nous n'allons pas toutes les citer*) mais nous allons y ajouter un point de détail pour les masques.

- La *classe A* : qui va de l'adresse *IPv4 0.0.0.0* à *127.255.255.255*, avec un *masque de sous réseau* en *255.0.0.0*.
- La *classe B* : qui va de l'adresse *IPv4* 128.0.0.0 à 191.255.255.255 avec un *masque de sous réseau* en *255.255.0.0*.
- La *classe C* : qui va de l'adresse *IPv4 192.0.0.0* à *223.255.255.255*, avec un *masque de sous réseau* en *255.0.0.0*.

Classe	Masque réseau	Adresses réseau	Nombre de réseaux	Nombre d'hôtes par réseau
A	255.0.0.0	1.0.0.0 – 126.255.255.255	126	16777214
B	255.255.0.0	128.0.0.0 – 191.255.255.255	16384	65534
C	255.255.255.0	192.0.0.0 – 223.255.255.255	2097152	254
D	240.0.0.0	224.0.0.0 – 239.255.255.255	adresses uniques	adresses uniques
E	non défini	240.0.0.0 – 255.255.255.255	adresses uniques	adresses uniques

Je tiens à *notifier* une *précision importante*, il faut toujours qu'il y ait des *1 à gauche* et des *0 à droite* (en *binaire*) de votre *masque de sous-réseau*, par exemple le masque *11111111.11111111.00000000.11111111* (*255.255.0.255*) est *incorrect*, cela n'existe pas. Je précise cela car j'ai vu des étudiants arrivés à des calculs de ce genre (je ne sais comment) mais cela est encore une fois simplement impossible. Un masque de sous-réseau en 255.255.0.255 n'existe pas. Tout comme, encore une fois, il ne peut y avoir de 0.

- Comment calculer l'adresse du réseau, du broadcast, etc.

Par exemple, avec *l'adresse IPv4 172.128.10.5* et *le masque de sous réseau 255.255.192.0,* comment calculer *l'adresse du réseau*, le *nombre d'adresses utilisables*, *l'adresse de broadcast* et *la plage adressable du réseau.*

Pour commencer, effectuons une *conversion en binaire* puis le *ET logique* :

10101100.10000000.00001010.00000101 ET

11111111.11111111.11000000.00000000

L'adresse du *réseau* est donc : *172.128.0.0*
(10101100.10000000.00000000.00000000).

Pour déterminer le *nombre d'adresses attribuables*, *depuis le masque de sous réseau en binaire*, vous allez compter le nombre de *bits* à partir des *zéros*, en l'occurrence, *14 bits* (*11111111.11111111.11/000000.00000000*).

Vous allez les transformer en *1, 11111111111111 (en binaire) équivaut donc à 16 383 (en décimal), il faut rajouter +1 (car nous comptons de 0 à 9), puis -2* (car *l'adresse de broadcast* et *l'adresse rése*au *ne peuvent pas être utilisés*),
16 383, + 1 = 16 384, – 2 = 16 382

16 382 est donc le nombre *d'adresses attribuables.*

Pour déterminer le nombre *d'adresses attribuables,* nous avons transformé *les 14 derniers bits du masque de sous réseau*.

Pour déterminer *l'adresse du broadcast,* nous allons effectuer la même chose, *mais cette fois-ci avec des 1 (*et il faut bien évidemment s'adapter *en fonction du nombre de bits*, s'il y avait eu *16 bits*, il aurait fallu y transformer les *16 derniers bits)*. Après avoir transformé les *14 derniers bits* en *1*, nous allons *poser l'adresse du réseau* et la convertir (*en décimal*).

10101100.10000000.00111111.11111111 (précédemment *10101100.10000000.00000000.00000000)* soit 172.128.63.255.

La plage *adressable* va de *l'adresse du réseau +1 (172.128.0.1)*, jusqu'à *l'adresse de broadcast -1* (172.128.63.254).

LE SUBNETTING

Le *subnetting* est une technique qui consiste à *diviser* un *réseau* en plusieurs *sous-réseaux*, ou l'action de *créer des sous-réseaux*, et ce en *personnalisant les masques*.

Admettons que vous ayez un *réseau* de *200 ordinateurs*. La gestion d'un tel *réseau* ne sera pas forcément évidente. Grâce au *subnetting*, nous pourrons *diviser* ce *grand réseau,* par exemple en *8 (sous) réseaux* de *30 ordinateurs* chacun.

Le *subnetting* peut vous apporter de *nombreux avantages* :

- *Une délégation de l'administration* : le *subnetting, puisqu'il* permet de *diviser un grand réseau en plusieurs réseaux plus petits*, permet une certaine de forme de délégation de la *gestion* de chaque *sous-réseau,* à *une personne différente*. Dans une entreprise possédant un *réseau* de *plusieurs centaines de machines*, sa *gestion* sera *simplifiée*.

- *La réduction du trafic* : *2 ordinateurs* dans un même *sous-réseau* qui communiquent n'exploiteront *que* la *bande passante* allouée à leur *sous-réseau*, et non celle du *réseau entier*. Considérons une entreprise possédant *un réseau de 500 machines*, divisé en *8 sous-réseaux* de *x machines*. Les *machines* appartenant à un même *sous-réseau* communiquant entre elles n'utilisent *que* la bande passante qui est allouée à leur sous-réseau, ce qui permet de *ne pas réduire le débit des autres*. Cela se remarque notamment lors des *broadcast*s (elles ne sont transmises *qu'aux ordinateurs du sous-réseau*).

- *La facilité du diagnostic* : par exemple, si un ordinateur consomme une quantité de bande passante *inhabituelle*, il sera beaucoup plus simple d'analyser son comportement et régler le problème s'il se trouve dans un *petit sous-réseau plutôt* que s'il se trouve dans *le même réseau* que *1000 autres machines*.

Les *administrateurs réseaux* passent plus de temps à *analyser* qu'à *implémenter*. C'est d'ailleurs le rôle *principal* d'un *administrateur réseau* : appo*rter son expertise dans l'analyse et le design d'une infrastructure réseau*.

Quant à *l'implémentation*, ce qui prend le plus de temps n'est pas nécessairement la mise en place en elle-même mais *l'analyse*.

Comme en *programmation* (en entreprise) : le plus souvent, le *chef de projet analyse les contraintes* et les *demandes* des *clients*, écrit éventuellement un *cahier des charges*, et remet ces informations aux *développeurs*.

En *réseau*, c'est plus ou moins le *même principe* : vous (*l'administrateur)* allez réfléchir, en fonction des *contraintes et d'une demande*, proposer une *solution,* en tenant compte de plusieurs *critères* (tels que le *prix*, la *facilité de mise en place*, *l'évolution de l'infrastructure*, etc.).

- *Analyse des contraintes*

Avant de *subnetter* un *réseau*, il faut réaliser une *minutieuse analyse*, je vais vous apporter quelques *pistes de réflexion (les principaux critères à prendre en compte pour votre analyse)*.

- Le *prix : subnetter* un *réseau*, c'est le *subdiviser* en *plusieurs sous-réseaux*. Il en résulte *explicitement* que l'achat de matériel additionnel sera obligatoire. En effet il faudra du matériel tel qu'un routeur (ou un commutateur de niveau 3) pour que les *sous-réseaux* obtenus puissent communiquer.

  Qui dit *nouveau matériel* dit forcément, *câblage.* Et il faudra prendre en compte les *contraintes financières imposé par votre client* (*ou votre patron*), il faudra nécessairement trouver des *compromis* pour allier le meilleur rapport qualité-prix.

- *L'évolution du réseau* : une vision *courtermiste* est une vision *inadéquate* pour être un *bon administrateur*. Le domaine de *l'informatique* et les *réseaux évoluent très vite*. Il ne faut jamais penser à une solution (*infrastructure*) *courtermiste* et toujours préparer une éventuelle *évolution* de *l'infrastructure* que vous mettez en en place.

- *Le nombre d'adresses IP* : il faut *déterminer au préalable* le nombre *d'adresses IP* dont vous aurez besoin sur le réseau. Les *administrateurs débutants* ont tendance à choisir un *sous-réseau* qui leur offre *exactement,* ou *quasiment le même* nombre *d'adresses IP* dont ils auront besoin. Or, cette pratique est *fortement déconseillée pour une raison très simple*. Par exemple, si nous nous restreignons à un *sous-réseau* qui nous permet d'avoir *17 adresses IP*, et qu'il ne nous reste que 5 adresses de libres par sous réseau, mais que l'entreprise se met à *agrandir son infrastructure* en y ajoutant plus de 5 postes par sous réseau. Il faudra *subnetter* correctement et redéfinir les plages, ce qui va demander *un travail conséquent*.

  Il est donc plutôt *recommandé* de choisir un *masque* en se basant sur le *maximum d'adresses IP* qu'un réseau donné *pourrait* avoir, et non le *minimum,* ou *l'actuel*. Si vous avez besoin d'un *sous-réseau* de *10 adresses IP* mais *qu'il pourrait dans le futur* y avoir un agrandissement du réseau, mais que vous êtes certain qu'il ne dépassera pas un maximum de *40 postes*, il serait alors judicieux de commencer par choisir un masque qui vous donne plus de *40 adresses IP, et ne pas vous contenter du nombre de postes actuels*.

- *L'organisation* : l'une des choses les plus importantes *(hormis les contraintes évoquées ci-dessus)* est de dresser une *parfaite organisation* du *plan d'adressage*.

  Le *plan d'adressage* est un plan résultant d'une *analyse de contrainte* qui servira de *modèle* pour gérer *l'adressage* et *l'assignation* des *adresses* dans un réseau donné.

  Comment allez-vous organiser vos réseaux et sous-réseaux ? Cette question est fondamentale. Il faut savoir que plusieurs méthodes d'organisation existent sont courantes sur lesquels vous pourrez prendre exemple.

L'organisation par bâtiment : certaines entreprises ont une organisation par *architecture* (*physique*). Par exemple, elles peuvent avoir un *bâtiment A*, qui regroupera les services administratifs de l'entreprise, et un bâtiment *B* qui regroupera les services informatiques (développement, sécurité, administration), etc.

Par conséquent, vous pouvez être amené à *subnetter* et *organiser les sous-réseaux* par *bâtiment* en créant un sous-réseau pour le *bâtiment A*, un autre pour le *bâtiment B*, etc. Cette organisation consiste tout simplement à créer autant de *sous-réseaux* qu'il y a de bâtiments.

Elle offre une certaine facilité de diagnostic *grâce au repérage physique*. Les ordinateurs du bâtiment A pourraient éventuellement être nommés *A1, A2, A3*, etc. et les ordinateurs du bâtiments *B, B1, B2, B3,* etc.

- *L'organisation par fonction :* cette organisation est différente de la précédente mais pas moins intéressante, partons du postulat qu'il y ait un bâtiment A qui regroupent les employés du service du support informatique. Dans ce bâtiment, il y a un chef de projet, des techniciens informatiques et des réceptionnistes, mais il se peut que l'entreprise ait aussi des *techniciens* de proximité dans le *bâtiment B* ou un autre *chef de projet* dans le *bâtiment C,* qui s'occupe d'un autre service. Dans cette situation, vous pourrez *subnetter* par *fonction*.

C'est-à-dire, créer un *sous-réseau* qui n'hébergera que les ordinateurs des *chefs de projets* (*tous services confondus*), un autre *sous-réseau* qui n'hébergera cette fois-ci *que les secrétaires* (*tous services confondus*), etc.

Le plus gros avantage de cette *répartition* est de pouvoir par exemple, attribuer un *haut débit* pour un groupe *plutôt qu'un autre*. *En effet*, par exemple les secrétaires n'ont pas forcément le même besoin en matière de performance (débit) que les techniciens.

- Calcul du subnetting

Dans cette partie, je vais vous expliquer comment calculer, *en fonction du masque de sous réseau*, le nombre de *sous réseaux* pouvant être mis en place sur le réseau.

Contextualisons, un grand lycée a besoin de vos services d'administrateur, s'agissant d'un grand lycée, vous devrez compter environ 50 machines par sous réseau, pour 4 sous-réseaux (un sous-réseau pour le CDI, pour l'administration, la salle des profs, et une salle dédiée au cours d'informatique).

L'adresse IP du réseau sera 192.168.0.0/24, quel masque de sous réseau utiliser pour avoir la possibilité de créer 4 sous-réseaux ?

Pour commencer, cela ne pourra pas être un masque dit *classful* (*255.0.0.0, 255.255.0.0, 255.255.255.0*), qui ne possède qu'un seul réseau.

11111111.11111111.11111111.11000000 possèdera *64 IP attribuables* (*dont 4 sous-réseaux*), le masque de sous réseau sera *255.255.255.192*.

L'adresse du premier réseau est 192.168.0.0, les adresses attribuables vont de *192.168.0.1* à *192.168.0.62 (-1 pour le broadcast et -1 pour la prochaine adresse réseau)*, le broadcast est 192.168.0.63.

L'adresse du deuxième réseau est de *192.168.0.64, qui va de l'adresse 192.168.0.65 à 192.168.0.126*, le broadcast étant 192.168.0.127.

Le troisième *192.168.0.128, allant de 192.168.0.129 à 192.168.0.190, le broadcast étant 192.168.0.191.*

Et enfin, le dernier réseau est 192.168.0.192, allant de 192.168.0.193 à 192.168.0.254, le broadcast étant 192.168.0.255.

Le calcul pour les sous-réseaux :

À chaque bit ajouté au masque (*classful par défaut*), nous allons additionner le chiffre par lui-même en commençant par deux, qui sera le *résultat* du *nombre de sous réseaux*.

- *11111111.11111111.11111111.00000000 = 0 sous réseaux, masque classful.*
- *11111111.11111111.11111111.10000000 = 2 sous réseaux.*
- *11111111.11111111.11111111.11000000* = 4 sous réseaux.
- *11111111.11111111.11111111.11100000* = 8 sous réseaux.
- *11111111.11111111.11111111.11110000* = 16 sous réseaux.
- *11111111.11111111.11111111.11111000* = 32 sous réseaux.
- *11111111.11111111.11111111.11111100* = 64 sous réseaux.

Par la suite, pour calculer le nombre d'IP attribuables, il ne reste plus qu'à faire la même chose avec les zéros. Puis retirer 2 au résultat final (*pour l'adresse de broadcast ainsi que l'adresse du réseau*).

- *11111111.11111111.11111111.00000000 = 256 - 2 = 254 adresses attribuables.*
- *11111111.11111111.11111111.10000000 = 128 - 2 = 126 adresses attribuables.*
- *11111111.11111111.11111111.11000000 = 64 – 2 =* 62 *adresses attribuables*.
- *11111111.11111111.11111111.11100000 = 32 – 2 =* 30 *adresses attribuables*.
- *11111111.11111111.11111111.11110000 = 16 - 2 =* 14 *adresses attribuables*.
- *11111111.11111111.11111111.11111000 = 8 - 2 = 6 adresses attribuables*
- *11111111.11111111.11111111.11111100 = 4 – 2 = 2 adresses attribuables*.

Par la suite, vous devrez effectuer le calcul des adresses en fonction du nombre de bits ajoutés, par exemple avec *2 bits* ajoutés (*11111111.11111111.11111111.11000000),* vous devrez compter de *64 en 64 (car 64 dans la partie host).*

Pour un réseau 192.168.0.0 et un masque de sous réseau en 255.255.255.192 (11111111.11111111.11111111.11000000) :

Le premier réseau est de 192.168.0.0, ses adresses attribuables vont de 192.168.0.1 à 192.168.0.62 (64-2), le broadcast étant 192.168.0.63 (0 + 64 – 1 pour le broadcast et -1 pour l'adresse du prochain réseau).

Le deuxième réseau est de *192.168.0.64, allant de* 192.168.0.65 à 192.168.0.126 (64 + 64 – 2 car -1 pour le broadcast et -1 pour le prochain réseau), le broadcast étant 192.168.0.127.

Le troisième réseau *192.168.0.128… Le quatrième (dernier) 192.168.0.192 (128 + 64 = 192), avec ses adresses attribuables allant de 192.168.0.193 à 192.168.0.254, le broadcast étant 192.168.0.255 (192 + 64 = 256).*

bit ajouté au masque classfull	adresses reseau
1 bit = /2 reseaux	de 128 en 128 (x2)
2 bits = /4 reseaux	de 64 en 64 (x4)
3 bits = /8 reseaux	de 32 en 32 (x8)
4 bits = /16 reseaux	de 16 en 16 (x16)
5 bits = /32 reseaux	de 8 en 8 (x32)
6 bits = /64 reseaux	de 4 en 4 (x64)
7 bits = /128 reseaux	de 2 en 2 (x128)
8 bits = /256 reseaux	de 1 en 1 (x256)

La notation CIDR

La *notation CIDR* (*Classless Inter-Domain Routing*) donne le numéro du réseau suivi d'une barre oblique (*slash*), et à la suite de cette barre oblique se trouve le nombre de bits à 1 dans la notation binaire du masque de sous-réseau.

Je m'explique par un exemple concret.
Le *CIDR /25* signifie un masque de *sous réseau* en 255.255.255.128.

Pourquoi ?
Comme décrit dans l'énoncé, « à la suite de cette barre oblique se trouve le nombre de bits à 1 dans la notation binaire du masque de sous-réseau », en l'occurrence :
11111111.11111111.11111111.10000000, soit *25 bits*, donc un *CIDR* en */25*.

Prenons un autre exemple, le masque *255.255.224.0*, équivalent en binaire à 11111111.11111111.11100000.00000000, et sera donc représenté par */19*, car il y a *19 « 1 », s'il y en avait eu 27, le CIDR* aurait été */27*.

La notation *91.198.174.2*/19 désigne donc l'adresse IP *91.198.174.2* avec le masque *255.255.224.0*, et signifie que les *19 premiers bits* de l'adresse sont dédiés au réseau / sous-réseau, et le reste à l'*hôte / l'intérieur du sous-réseau*.

/32 désigne un réseau qui ne comporte *qu'une seule adresse IP*, c'est-à-dire une *adresse IP individuelle, et n'autorise ni les adresses de diffusion ni les adresses réseau.*

Pour numéroter des adresses de liens point à point (*liaison entre deux hôtes uniquemen*t), on utilisait des */30*, soit quatre adresses, dont deux utilisables pour adresser des interfaces.

Le RFC 30211 permet cependant d'utiliser plus efficacement l'espace d'adressage en permettant le /31 (il n'y a dans ce cas pas d'adresse de broadcast et l'adresse du sous-réseau est utilisée pour numéroter une interface).

Le protocole DHCP

DHCP (Dynamic Host Protocol, protocole de configuration dynamique des hôtes en français*)* est un *protocole réseau* dont le rôle est d'assurer la *configuration automatique des paramètres IP d'une machine*, notamment e*n lui attribuant automatiquement une adresse IP et un masque de sous-réseau*.

DHCP peut aussi configurer l'adresse de la *passerelle par défaut*, des serveurs *DNS* et des serveurs de noms *NBNS* (connus sous le nom de serveurs *WINS* sur les réseaux de la société *Microsoft*). Il faut également savoir que les serveurs DHCP doivent être pourvus d'une adresse IP statique.

Pour faire une analogie, il s'agit en quelque sorte d'un distributeur automatique de bonbons, à la différence près qu'il distribue des adresses.

Plus précisément, on peut faire une analogie du protocole DHCP avec un théâtre, je m'explique :

Avant d'aller voir une pièce de théâtre et de rentrer dans la salle, il vous faudra passer par le guichet pour commander un ticket. Le caissier vous attribuera un numéro de salle et un numéro de place, salle 2, place 24 par exemple.

La conception initiale *d'IP* supposait la pré-configuration de chaque ordinateur connecté au réseau avec les *paramètres TCP/IP* adéquats : c'est *l'adressage statique* (nommée également *IP fixe*).

Cependant, sur des réseaux de *grandes dimensions ou étendues*, où des modifications interviennent souvent, *l'adressage statique* engendre *une lourde charge de maintenance* et *des risques d'erreurs*.

Ce protocole peut fonctionner avec *IPv4*, il fonctionne aussi avec *IPv6* et il est alors appelé *DHCPv6*. Toutefois, en IPv6, les adresses peuvent être *autoconfigurées* sans *DHCP*.

- *Comment fonctionne DHCP ?*

L'ordinateur équipé d'une *carte réseau*, mais *dépourvu d'adresse IP*, envoie en diffusion (*Broadcast)* un *datagramme* (*DHCP DISCOVER*) qui s'adresse au port *67 (*c'est le port sur lequel un serveur *DHCP* écoute*)* de n'importe quel serveur à l'écoute sur ce port.

Ce datagramme comporte entre autres l'adresse *physique* (*MAC*) du client.

Pour rappel, un datagramme est un *paquet de données* transmis avec ses *adresses de source* et de *destination*.

Tout serveur DHCP ayant reçu ce datagramme, s'il est en mesure de proposer une adresse sur le réseau auquel appartient le client, envoie une offre DHCP (*DHCP OFFER*) à l'attention du client (*sur son port 68*), identifié par son *adresse physique*.

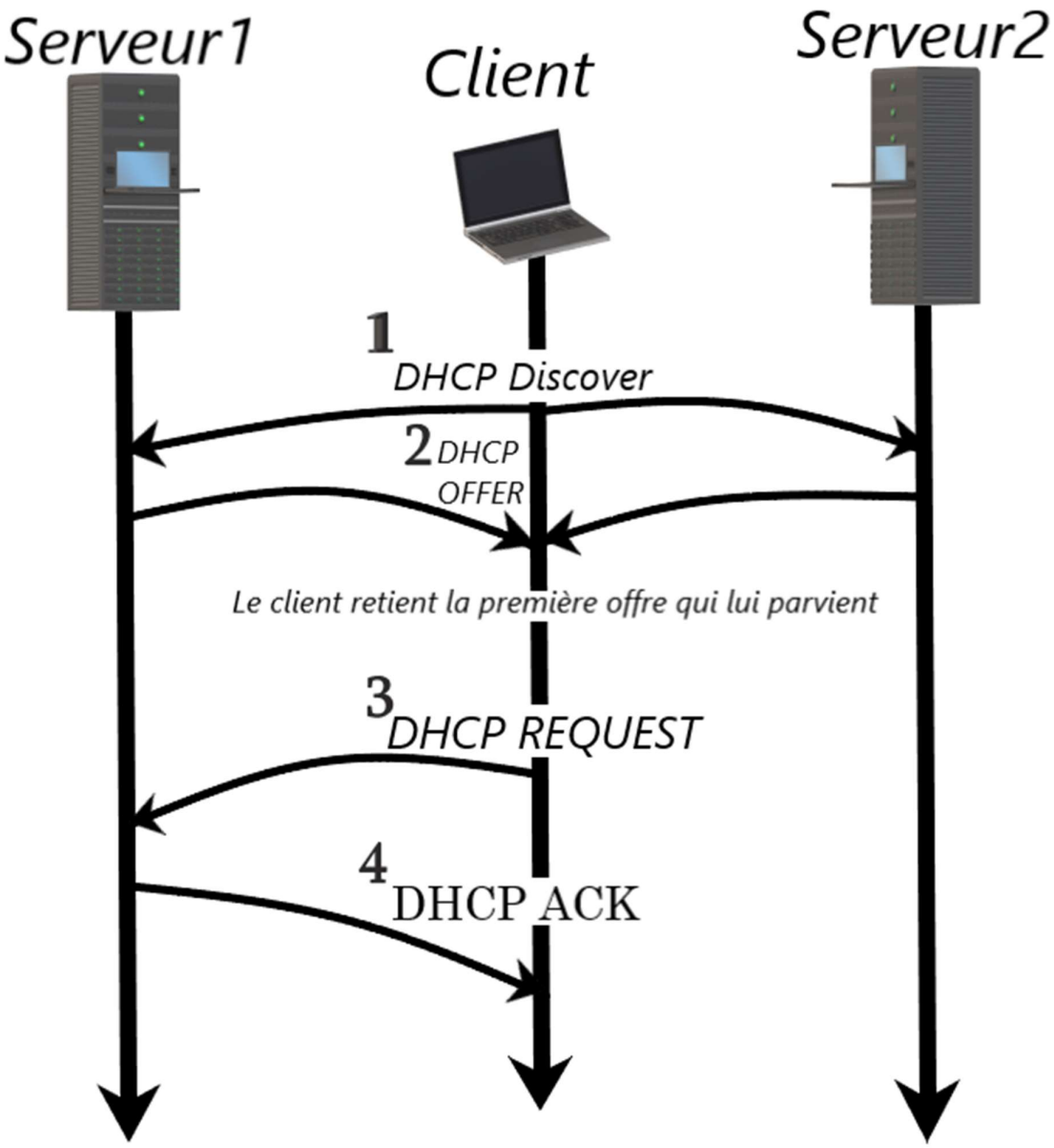
Cette offre comporte *l'adresse IP du serveur,* ainsi que *l'adresse IP* et le *masque de sous-réseau* qu'il propose au client.

Il se peut que *plusieurs offres* soient adressées au client. Le client retient une des offres reçues (*la première qui lui parvient*), et diffuse sur le réseau un *datagramme* de requête *DHCP* (*DHCP REQUEST*).

Ce datagramme comporte l'adresse IP du serveur et celle qui vient d'être proposée au client. Elle a pour effet de demander au serveur choisi, l'assignation de cette adresse, l'envoi éventuel des valeurs des paramètres, et d'informer les autres serveurs qui ont fait une offre qui n'a pas été retenue.

Le serveur *DHCP* élabore un *datagramme* d'accusé de réception (*DHCP ACK* pour *acknowledgement*) qui assigne au client *l'adresse IP* et son *masque de sous-réseau*, la durée du bail de cette adresse (dont découlent deux valeurs *T1* et *T2* qui déterminent le comportement du client en fin de bail), et éventuellement d'autres paramètres tels que *l'adresse IP* de la passerelle par défaut, des serveurs *DNS*, des serveurs *NBNS* (*WINS*), etc.

- *Schéma (synthétisation) :*



1 - DHCP Discover : Le client envoie un datagramme en broadcast.

2 - DHCP Offer : Le serveur DHCP envoie une offre DHCP.

3 - DHCP Request : Le client diffuse sur le réseau un *datagramme*.

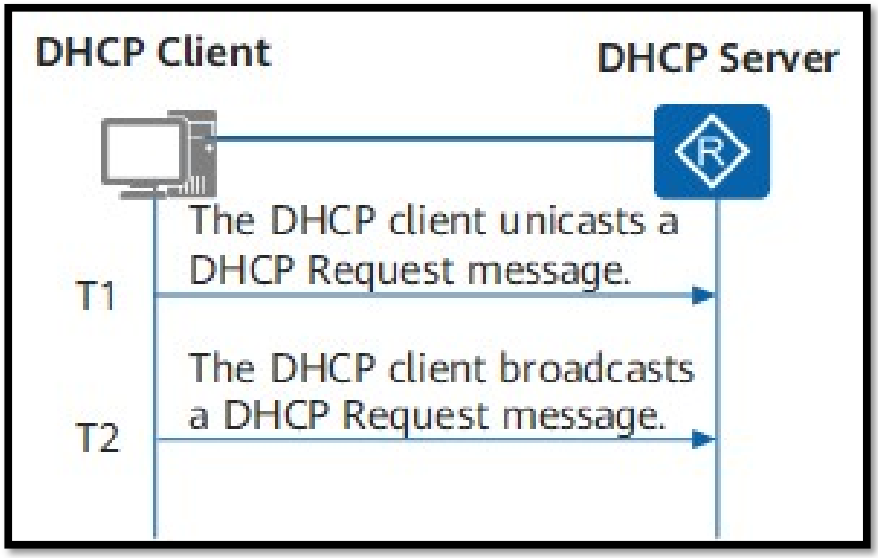
4 – DHCP Ack : Le serveur élabore un *datagramme* d'accusé de réception.

Il faut également savoir que les *adresses IP dynamiques* sont octroyées pour une *durée limitée* (*durée du bail*, ou *lease time*), qui est transmise au client dans l'accusé de réception qui clôture la transaction *DHCP*.

La valeur *T1* qui l'accompagne détermine la durée après laquelle le client commence à demander périodiquement le renouvellement de son bail auprès du serveur qui lui a accordé son adresse. Cette fois, la transaction est effectuée par transmission IP classique, d'adresse à adresse.

Si, lorsque le délai fixé par la deuxième valeur, *T2*, est écoulé et que le bail n'a pas pu être renouvelé (par exemple, si le serveur *DHCP* d'origine est *hors service*), le client demande une nouvelle allocation d'adresse par diffusion.

Si, au terme du bail le client n'a pu ni en obtenir le renouvellement, ni obtenir une nouvelle allocation, l'adresse est désactivée et il perd la faculté d'utiliser le réseau TCP/IP de façon normale.



L'adresse APIPA

APIPA (*Automatic Private Internet Protocol Addressing*) est un processus qui permet à un système d'exploitation de s'attribuer automatiquement une adresse IP, lorsque le serveur *DHCP* est hors service ou injoignable.

APIPA utilise la plage d'adresses IP *169.254.0.0/16* (que l'on peut également noter *169.254.0.0/255.255.0.0*), c'est-à-dire la plage dont les adresses vont de *169.254.0.0 à 169.254.255.255*.

Le *protocole ARP*

L'ARP ou Address Resolution Protocol (*en français, protocole de résolution d'adresse*) est un protocole utilisé pour traduire une adresse de protocole de couche *réseau* (*typiquement une adresse IPv4*) en une adresse de protocole de couche de liaison (*typiquement une adresse MAC*).

Il se situe à l'interface entre la *couche réseau* (*couche 3 du modèle OSI*) et la *couche de liaison* (*couche 2 du modèle OSI*). Le *protocole ARP* est nécessaire au bon fonctionnement *d'IPv4* utilisé au-dessus d'un réseau de type *Ethernet*. En *IPv6*, les fonctions de *ARP* sont reprises par le *Neighbor Discovery Protocol* (*NDP*).

Pour résumer plus simplement, le *protocole ARP* joue un rôle phare parmi les protocoles de la couche Internet de la suite *TCP/IP*, car il permet de connaître l'adresse *physique d'une carte réseau* correspondant à une *adresse IP*, c'est pour cela qu'il s'appelle *Protocole de résolution d'adresse* (*Address Resolution Protocol*). À l'inverse, le protocole *RARP* (*Reverse Address Resolution Protocol*) permet *à partir d'une adresse matérielle (adresse MAC)* de déterminer *l'adresse IP* d'une machine.

Pour rappel, chaque machine connectée au réseau possède un numéro d'identification de 48 *bits*. Ce numéro unique est fixé dès la fabrication de la carte en usine, il s'agit de *l'adresse MAC* (ou *physique*). Toutefois la communication sur *Internet* ne se fait pas directement à partir de ce numéro mais à partir d'une adresse *logique (IP)*.

Ainsi, pour faire correspondre les adresses *physiques* aux adresses logiques, le protocole *ARP* interroge les machines du réseau pour connaître leur *adresse physique (Mac)*, puis crée une *table de correspondance* entre les *adresses logiques (IP)* et les *adresses physiques* (*Mac*) dans une *mémoire cache*.

Lorsqu'une machine doit communiquer avec une autre, elle consulte la *table de correspondance*. Si jamais l'adresse demandée *ne se trouve pas* dans la table, le protocole *ARP* émet une requête sur le réseau.

L'ensemble des machines du réseau vont comparer cette *adresse logique* (*IP*) à la leur. Si l'une d'entre-elles s'identifie à cette *adresse*, la machine va répondre à *ARP* qui va stocker le couple d'adresses dans la *table de correspondance* et la communication va alors pouvoir avoir lieu.

- Reformulons une dernière fois

Un ordinateur connecté à un réseau informatique souhaite émettre une trame *ethernet* à destination d'un autre ordinateur dont il connaît *l'adresse IP,* placé dans le même sous-réseau.

Dans ce cas, cet ordinateur va effectuer une *requête ARP* en *broadcast*.

Cette requête *crie* (*en quelque sorte*) « *S'il vous plaît, écoutez-moi ! Quelle est l'adresse MAC correspondant à l'adresse IP AdresseIp ? Répondez à MonAdresseIp.* ».

Puisqu'il s'agit d'un *broadcast*, *tous les ordinateurs* du *segment* vont recevoir la *requête*.

En observant son contenu, ils pourront déterminer quelle est *l'adresse IP* sur laquelle porte la recherche.

La machine qui possède cette *adresse IP* sera la seule à répondre en envoyant à la machine émettrice une *réponse ARP* du type « *Hey, c'est moi, je suis adresseIP, mon adresse MAC est adresseMAC* ».

Pour émettre cette réponse au bon ordinateur, il crée une entrée dans son *cache ARP* à partir des données contenues dans la *requête ARP* qu'il vient de recevoir.

La machine à l'origine de la requête *ARP* reçoit la réponse, met à jour son *cache ARP* et peut donc envoyer à l'ordinateur concerné le message qu'elle avait mis en attente.

L’ARP poisoning

L'ARP *spoofing* ou ARP *poisoning* est une technique utilisée en informatique pour attaquer les machines d’un réseau local utilisant le protocole de résolution d'adresse *ARP*.

Cette technique permet à l'attaquant de *détourner des flux de communications* transitant *entre une machine cible* et une *passerelle* : *routeur, box, etc*. L'attaquant peut ensuite *écouter*, *modifier* ou encore *bloquer les paquets réseaux*.

Comme vu précédemment, le *protocole ARP* a pour but de *traduire* une *adresse IP* en *adresse Mac* (*pour résumer brièvement*).

Pour qu'une *machine A* puisse communiquer avec une *machine B* sur le réseau, la *machine A* doit connaitre *l'adresse Mac* de la *machine B*, la machine va donc envoyer une *requête* (*broadcast*) contenant le message suivant : "*Quelle est l'adresse Mac de l'ordinateur possédant l'adresse IP adresseIP, je suis monAdresseIP, répondez-moi sur cette adresse*", seul l'ordinateur *concerné par cette demande* est censé répondre "*Bonjour toi, c'est moi adresseIP, voici mon adresse Mac*", par la suite, un *cache ARP* est créé *à partir de ces données*.

Le problème avec ce protocole est qu'il n'a pas été conçu en prenant en compte le facteur *sécurité* (la possibilité de tomber sur des *individus malveillants)*.

Je peux donc (*par exemple*) me faire passer pour le *routeur*, sachant qu'absolument toutes les données passent par celui-ci, et c'est par moi que toutes ces données vont passer.

Je peux potentiellement *recevoir* des données, en *modifier*, en *supprimer*, *pousser l'utilisateur à se rendre sur un site* (*frauduleux*), plutôt qu'un autre, etc.

Ce type d'attaque peut être appelé une *attaque de l'homme du milieu* (*MITM* ou *Man in the middle*).

Ce type d'attaque a pour but *d'intercepter toutes les communications entre deux périphériques*, sans qu'aucun des deux ne se doute de quoi que ce soit.

Pour *lutter* contre ce type d'attaques, il est possible :

- De mettre en place des *entrées statiques* dans le *cache ARP* de chaque machine du *réseau*.

  Cette méthode n'est applicable qu'à un faible nombre de machines (*privilégier les plus critiques*, comme les *serveurs* et les *passerelles*), auquel cas cela vous prendra énormément de temps.

- De limiter les *adresses MAC* sur *chaque port* (*renseignement statique*) des commutateurs s'ils le permettent grâce à la *fonction Port Security*.

  Le *switch* utilise une *table CAM* qui mémorise les adresses *Mac* ainsi que les adresses IP des équipements connectés.

  Si un équipement ne communique plus pendant 300 secondes, son entrée dans la table est effacée (cette table a une taille limitée de quelques milliers d'adresses en règle générale).

  Le *port-security* permet de *contrôler* les *adresses Mac* des équipements connectés sur une ou plusieurs interfaces d'un *switch*.

  Il permet de limiter le nombre d'adresses Mac apprises sur une interface donnée.

  Il peut même ne laisser passer *que* certaines adresses *connues*, et bloquer les autres.

  Les commutateurs de *niveau 3* (un *switch* de *niveau 3* est à la fois un *commutateur* et un *routeur*) par exemple offrent la possibilité de paramétrer des associations *port/MAC/IP statiques*.

- De surveiller les *messages ARP* circulant sur *réseau informatique*, à l'aide d'outils de surveillance tels *qu'ARPwatch*, *d'arpalert* ou, *d'un IDS* (*Systèmes de Détection d'Intrusion*).

Chaque entrée a une *durée de vie* (*cela oblige d'ailleurs l'attaquant à corrompre régulièrement le cache de la victime*). Certains Systèmes d'exploitation comme *Solaris* permettent de modifier la valeur de ce temps d'expiration.

La *table CAM*

La table *CAM* (*Content-Addressable-Memory*) fait le lien *entre les ports physiques du commutateur et les adresses Mac qui arrivent sur ces ports*.

Cette table va permettre au *commutateur* de savoir *ou envoyer les informations qu'il reçoit*. Il ne *faut pas* confondre (malheureusement très souvent confondu) la table *ARP* qui établit la relation *entre une adresse IP et une adresse MAC*, et la table *CAM* qui établit la relation *entre une adresse MAC et un numéro de port*.

Partons du principe qu'il y ait un *PC1* et un *PC2* branchés sur le même *commutateur* et *que le commutateur vient de démarrer* (ces tables sont stockées en *RAM*, ce qui fait qu'à chaque redémarrage, toutes les informations sont effacées).

PC1 souhaite *communiquer* avec *PC2*, il a besoin d'une *IPsource* et d'une *MACsource*, une *IPdest* et une *MACdest* (*destination*). Connaissant l'adresse *IPdest* de *PC2* mais *pas* la *MACdest*, il va effectuer une *requête ARP* (*cf. le protocole ARP*) que le *commutateur* va recevoir et va transmettre en *broadcast* sur *tous les ports, toutes les machines connectées au commutateur vont donc recevoir la requête*, puis *seul le PC concerné* (en l'occurrence *PC2*) va répondre à la requête.

Que se passe-t'il au niveau du *commutateur* ? Après avoir renvoyé à tous ses ports le *broadcast*, il regarde la *MACsource* qu'il reçoit sur le *port correspondant*. *Admettons* que le *PC1* soit branché sur le *port 1 du commutateur*, il va associer le *port 1,* à l'adresse *MACdest,* du *PC1*.

Après la réponse au *broadcast* de *PC1* par *PC2* (en *unicast,* cette fois), le commutateur va donc également associer l'adresse Mac du PC2 sur le port 2 (en admettant qu'il soit bien branché sur le *port 2*).

Désormais, le commutateur sait (*et a ajouté les informations dans sa table CAM*) que *PC 1=port 1* et *PC 2=port 2* du *commutateur*. Si aucun échange n'a été effectué par une machine pendant *300 secondes ou plus,* ces informations se verront *supprimées* de la table (*s'il ne communique plus avec le commutateur*), *sur un router,* le délai est fixé à *14400 secondes* (4 heures).

IDS (Système de détection d'intrusion)

Un système de *détection d'intrusion* (ou *IDS* pour I*ntrusion detection System*) est un mécanisme destiné à repérer des *activités anormales* ou *suspectes* sur la cible analysée (un *réseau* ou un *hôte*). Il permet ainsi d'avoir une connaissance sur les tentatives réussites comme échouées des *intrusions*.

Il existe de nombreuses *catégories* de *systèmes de détection d'intrusion*, les plus communs sont les *systèmes de détection d'intrusion réseau* et les *systèmes de détection d'intrusion hôte*.

Certains *systèmes* de détection ont la possibilité de *répondre aux menaces* qu'ils ont détectées, il s'agit *d'IPS* (*systèmes de prévention d'intrusion* ou *intrusion prevention system*), qu'il ne faut pas confondre avec un *IDS*.

Un *IPS* est un outil (similaire aux *IDS)* permettant de *prendre des mesures* afin de *diminuer* les *impacts* d'une *attaque*. C'est un *IDS actif,* il *détecte* un *balayage automatisé*, *l'IPS* peut *bloquer les ports automatiquement*. Les *IPS* peuvent donc *parer* les attaques connues et inconnues. Comme les *IDS*, ils ne sont pas fiables à *100%* et risquent même *en cas de faux positif* de bloquer du *trafic légitime*. Pour faire une analogie, voyez *l'IDS* comme un *chihuahua* qui ne se contente que d'aboyer en cas de menaces, et *l'IPS* comme un *berger allemand,* capable de contre-attaquer en cas de menaces.

En ce qui concerne *l'IDS*, son objectif premier est de *détecter des activités malicieuses* sur la cible qu'il surveille. Une alerte sera déclenchée dès lors qu'un comportement malicieux est détecté. Les systèmes de détection d'intrusion sont utilisés en plus des solutions traditionnelles telles que les pare-feux, pour détecter différents types d'utilisation malicieuse de leur cible qui ne peuvent être détectée par ces dernières.

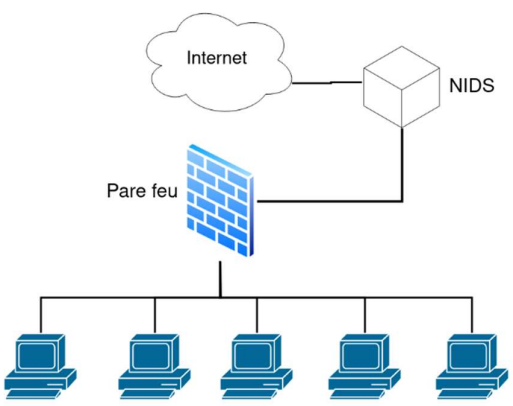
Pour cela, de nombreux paramètres doivent être pris en compte selon ce que l'on cherche à surveiller. En effet, le système de détection d'intrusion ne se placera pas au même endroit dans l'architecture réseau. Celui-ci peut être placé en coupure du réseau, ou sur un hôte. De plus, la temporalité de l'analyse est un paramètre important, celui-ci peut produire son analyse en temps réel ou à posteriori.

- *Systèmes de détection d'intrusion réseau (NIDS)*

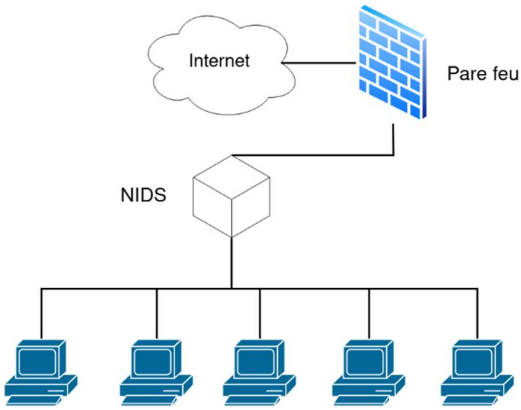
Les *systèmes de détection d'intrusion réseau* (ou *NIDS* pour *Network Intrusion Detection System*) sont les *IDS* les plus répandus. Ce sont des outils très utiles pour *l'administrateur réseaux* qui va pouvoir, *en temps réel*, comprendre ce qui se passe sur son réseau et prendre des décisions en ayant toutes les informations.

Ils peuvent être placés à divers endroits sur le réseau, en amont ou en aval d'un pare feu ou encore sur chaque hôte, comme un anti-virus. Ces *IDS* vont analyser tout le trafic entrant et sortant du réseau afin d'y déceler des attaques. Cependant, un NIDS placé sur chaque hôte ne saura pas détecter toutes les attaques possibles comme les attaques par *déni de service (DDoS)* car il ne verra pas tout le trafic réseau, uniquement celui qui arrive à l'hôte final.

Quand un NIDS est positionné en amont d'un pare feu, il pourra alors générer des alertes pour le pare feu qui va pouvoir filtrer le réseau. Placé en aval du pare feu, le NIDS produira moins de faux positifs, car le trafic réseau qu'il analysera aura déjà été filtré par le pare feu.



NIDS placé en amont.



NIDS placé en aval.

Attaque par déni de service

Une attaque par *déni de service* (*DoS* ou *Denial of Service attack*) est une attaque ayant pour but de rendre *indisponible* un service, d'empêcher les utilisateurs légitimes d'un service de l'utiliser. À l'heure actuelle la grande majorité de ces attaques se font à partir de plusieurs sources, on parle alors d'attaque par *déni de service distribuée* (DDoS pour *Distributed Denial of Service attack*).

Parlons du *DDOS*, cette attaque est le plus souvent effectuée par un utilisateur malveillant ayant pris le contrôle d'un nombre conséquent de machines grâce à un (*ou des*) *virus*, il peut ordonner à ces machines d'envoyer de multiples *requêtes* en même temps sur un même *service* (*serveur, application, infra, etc*.), ce qui aura pour effet de *saturer* l'équipement se faisant attaquer, en conséquence de quoi les utilisateurs légitimes ne pourront plus s'y connecter.

En effet, imaginez *10 000 équipements* envoyant de *multiples requêtes* en même temps à un autre *équipement*, pour faire une *analogie*, imaginez une partie de *ping-pong,* ou un joueur déciderait (grâce à la complicité de ses collègues) d'envoyer une dizaine de balles *d'un seul coup*, forcément, l'adversaire se verrait surcharger *d'informations* (*de balles*) et pourra difficilement se défendre contre cette attaque sans préparation au préalable.

Ce réseau de multiples machines contrôlé par un utilisateur malveillant, s'appelle un *botnet*.

Le retour aux conditions normales après une attaque peut exiger une intervention humaine, car certains logiciels ne retrouvent dans certains cas, pas un fonctionnement normal après une attaque de ce type.

Le pare-feu

Un *pare-feu* (*firewall*) est un logiciel *et/ou* un matériel permettant de faire respecter la *politique de sécurité* du réseau, celle-ci définissant quels sont les types de communications autorisés sur ce réseau informatique. Il surveille et contrôle les applications et les *flux de données* (*paquets*) *entrants et sortants*.

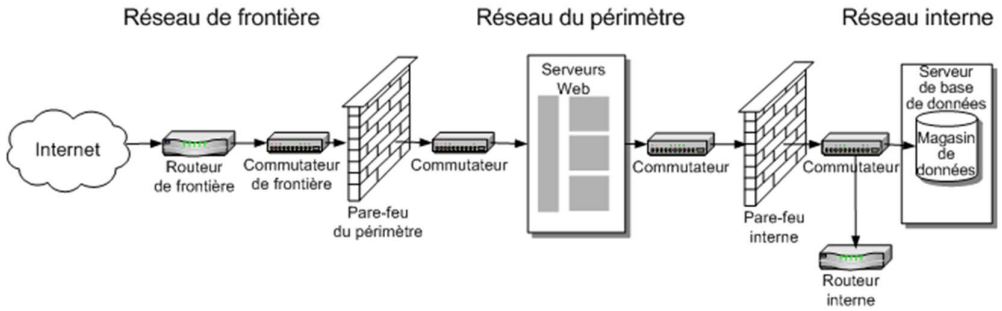
Il a pour *principale tâche* de contrôler le trafic entre différentes zones de confiance, en filtrant les flux de données qui y transitent. Généralement, les zones de confiance incluent *Internet* (*une zone dont la confiance est nulle*) et au moins un *réseau interne* (*une zone dont la confiance est plus importante*).

Le but est de fournir une connectivité *contrôlée* et *maîtrisée* entre des zones de *différents niveaux de confiance*, grâce à l'application de la *politique de sécurité* et d'un *modèle de connexion* basé sur le principe du moindre privilège.

Le *filtrage* se fait selon divers critères. Les plus courants sont :

- *L'origine* ou la *destination* des *paquets* (*adresse IP, ports TCP ou UDP, interface réseau, etc.*).
- Les options contenues dans les données (*fragmentation, validité, etc.*).
- Les données elles-mêmes (*taille, correspondance, etc.*).
- Les utilisateurs (*pour les firewalls les plus récents*).

Un *pare-feu* fait souvent office de routeur et permet ainsi *d'isoler le réseau en plusieurs zones de sécurité* appelées *zones démilitarisées* ou *DMZ*. Ces zones sont séparées suivant le niveau de confiance qu'on leur porte.



Les ports

Dans le *modèle TCP/IP* (et correspondant à la couche *transport* du *modèle OSI)*, la notion de *port* permet, sur un ordinateur donné, de distinguer différents *interlocuteurs*. Ces *interlocuteurs* sont des *programmes informatiques* qui, selon les cas, écoutent ou émettent des informations sur ces *ports*. Un *port* est distingué par son numéro.

L'adresse IP sert à identifier de façon unique une machine sur le réseau, le numéro de port quant à lui identifie l'application à laquelle les données sont destinées.

Pour faire une analogie, on peut considérer les *ports* comme des *portes*. Lorsqu'une application qui a besoin de communiquer en réseau se lance, elle ouvre sa propre porte (possédant un numéro de porte unique, elle ne change en général pas de bureau) pour que les données puissent entrer et sortir. Lorsqu'elle enverra des informations, elle spécifiera qu'elle les envoie sur la porte numéro *22* (*par exemple*) de l'ordinateur distant.

Un numéro de port est codé sur *16 bits*, ce qui fait qu'il existe un maximum de *2^16*, soit *65 536 ports* distincts par machine. Ces ports sont classés en *3 catégories* :

- Les numéros de port de *0 à 1 023* correspondent aux ports *well-known ports*, utilisés pour les services *réseaux* les plus courants.
- Les numéros de ports de *1 024 à 49 151* correspondent aux *registered ports*), assignés par *l'IANA*.
- Les numéros de ports de *49 152 à 65 535* correspondent aux *ports dynamiques*, utilisables pour tout type de *requêtes TCP* ou *UDP* autres que celle citées précédemment.

Lorsqu'un logiciel client veut dialoguer avec un logiciel serveur, aussi appelé service, il a besoin de connaître le port écouté par ce dernier. Les ports utilisés par les services devant être connus par les clients, les principaux types de services utilisent des ports qui sont dits réservés. Par convention, ce sont tous ceux compris entre 0 et 1 0232 inclus et leur utilisation par un logiciel serveur nécessite souvent que celui-ci s'exécute avec des droits d'accès particuliers. Les services utilisant ces ports sont appelés les *Well-Known Services*.

- Quelques exemples de ports à connaitre

- *9*, pour le *WoL* (*Wake-on-LAN)*.
- *20/21*, pour l'échange de fichiers via *FTP*.
- *22*, pour l'accès à un shell sécurisé *Secure Shell* (*SSH*), également utilisé pour l'échange de fichiers sécurisés *SFTP*.
- *23*, pour le port *telnet*.
- *25*, pour l'envoi d'un courrier électronique via un serveur dédié *SMTP*.
- *53*, pour la résolution de noms de domaine en adresses *IP* : *DNS*.
- *67/68*, pour *DHCP* et *BOOTP*.
- 69, pour le *TFTP*.
- *80*, pour la consultation d'un serveur *HTTP* par le biais d'un navigateur web.
- *110*, pour la récupération de son courrier électronique via *POP*.
- *123*, pour la synchronisation de l'horloge : *Network Time Protocol* (*NTP*).
- *143*, pour la récupération de son courrier électronique via *IMAP*.
- *389*, pour la connexion à un *LDAP*.
- *443*, pour les connexions *HTTP* utilisant une surcouche de sécurité de type SSL : *HTTPS*.
- *465*, pour l'envoi d'un courrier électronique via un serveur dédié utilisant une surcouche de sécurité de type *SSL* : *SMTPS*.
- *500*, port utilisé pour le canal d'échange de clés *IPsec*.
- *554*, port utilisé pour accepter les connexions client *RTSP* entrantes et pour fournir des paquets de données aux clients qui diffusent en utilisant *RTSPT*.
- *636*, pour l'utilisation d'une connexion à un *LDAP* sécurisé par une couche *SSL/TLS*.
- *1352*, pour le protocole *Lotus Notes Domino*.
- *1433*, serveur de base de données *MS SQL*.
- *1521*, serveur de base de données *Oracle Database*.
- *1723*, pour l'utilisation du protocole de *VPN PPTP*.
- *3306*, serveur de base de données *MySQL*.
- *3389*, pour la prise de contrôle à distance *RDP*.
- *5432*, serveur de base de données *PostgreSQL*.
- *6667*, pour la connexion aux serveurs *IRC*.
- *7777*, port souvent utilisé pour des serveurs de jeux-vidéo telle que *Terraria* ou *Unreal Tournament*.
- 25565, port par défaut des *serveurs Minecraft*.

Les protocoles

Jusqu'ici, nous avons vu quelques protocoles tels que *DHCP* et *ARP*, mais concrètement, qu'est-ce qu'un *protocole* ?

Reprenons *l'analogie* utilisé pour l'explication du *modèle OSI* (vu précédemment) :

Un *Japonais* (ou un *Arménien*) ne pourra pas comprendre un *Français,* puisqu'ils ne parlent pas la même langue (langues, qui ont tous deux *leurs propres règles*).

Pour parler une langue (pour parler en *Français* par exemple), il faut s'appuyer sur *un ensemble de règles*, de *syntaxe* à respecter, etc.

Un *protocole* est un ensemble de règles qui régissent les échanges de données ou le comportement collectif de processus ou d'ordinateurs en réseaux ou d'objets connectés. Il définit les *règles de communication* avec une application, *le type d'échanges*, la *syntaxe*, le type d'envoie et la réponse d'une *donnée*, etc.

Le but d'un *protocole* est de faire coopérer des agents (processus, ordinateurs, composants électroniques, objets connectés) hétérogènes à une tâche commune ou de leur faire s'échanger des *données*, les *protocoles* sont donc souvent rigoureusement définis.

Les *protocoles* sont souvent normalisés par des comités de normalisation, formel comme *l'ISO (Organisation internationale de normalisation)* ou informel comme *l'IETF* (*Internet Engineering Task Force*) pour *Internet*, on parle parfois aussi de standards. Les *protocoles* dits *propriétaires*, sont spécifiques à un constructeur ou fabricant, ne sont pas publics et sont, en général, rigoureusement définis. En revanche, un protocole normalisé et *public* permet à plusieurs constructeurs ou fabricants de faire *coopérer des agents*.

Un *protocole* applicatif est attribué à chaque *port* réservé.
Par exemple, le *port 80* est associé au *protocole HTTP*, le *port 21* est associé au *protocole FTP*. Les *protocoles applicatifs* s'appuient sur les *services* du *protocole* de la *couche transport* (soit la *4ème* couche du *modèle OSI*).

Le protocole DNS

Le *DNS* (*Domain Name System* ou *système de noms de domaine*), est le service chargé de faire la correspondance entre les noms de domaine et les adresses IP. *Résoudre un nom de domaine* consiste à trouver *l'adresse IP* qui lui est associée.

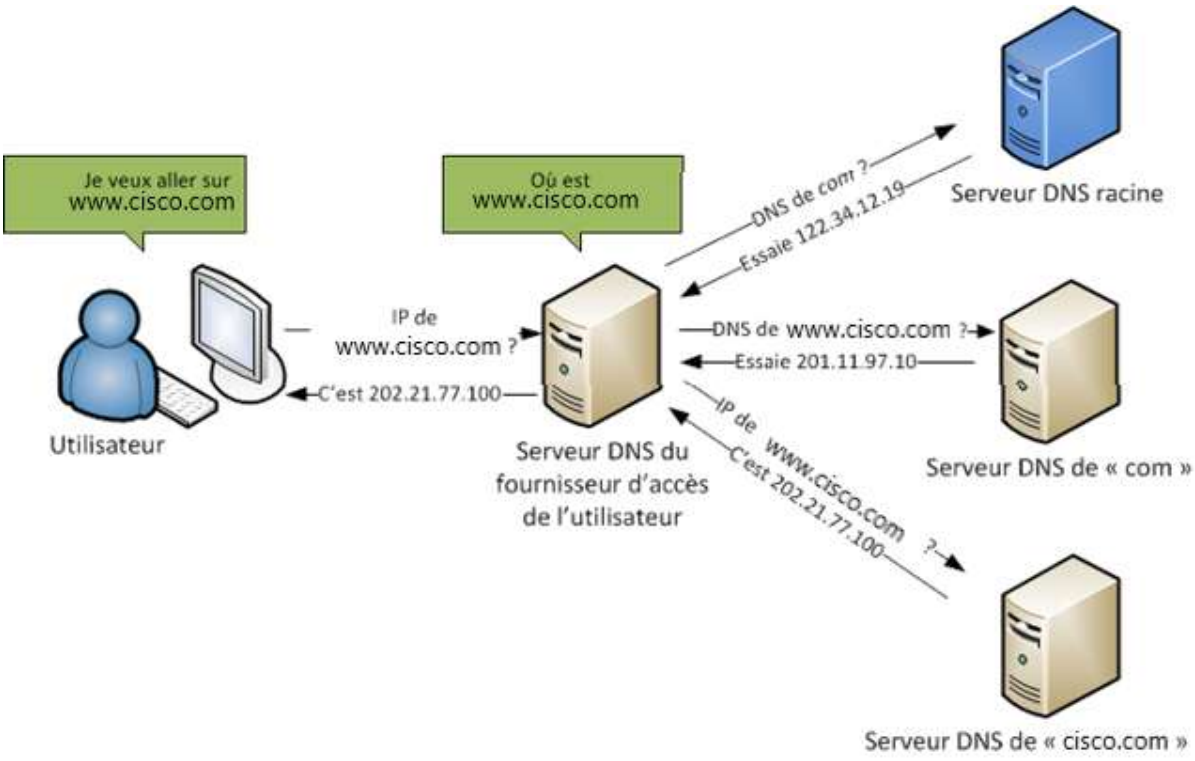
Lorsque vous vous utilisez un site web, par exemple que vous souhaitez aller sur le site de *Cisco*, vous allez utiliser un nom de domaine, en l'occurrence *cisco.com*, dans la pratique, votre machine n'a besoin que de l'adresse IP du serveur pour s'y connecter, cependant, cela risque d'être compliqué de devoir retenir une centaine d'adresses IP telles que *72.163.4.185* pour *cisco.com*, *91.198.174.192* pour *wikipédia.org*, etc.

Pour faire une analogie, c'est en quelque sorte comme une liste de contacts sur un téléphone portable. Il serait difficile pour un humain de retenir des centaines de numéros de téléphone à dix chiffres, c'est là qu'entre en jeu votre liste de contacts, *Franck FERMAN* est attribué au numéro *+33XXXXXXXXX* (et le contact Camille ARNAL au *+33XXXXXXXXX*, *Samy KAMKAR +1XXXXXXXXXX, etc.), le DNS, quant à lui, permet donc de trouver une adresse à partir d'un nom de domaine.*

Le poste client possède un à deux serveurs DNS fournis par l'entreprise (*ou votre FAI*), quand vous accéder à un site, la machine envoie une requête DNS au serveur (*DNS* qui se trouve dans la configuration de la carte réseau de la machine), qui va dans un premier temps vérifier s'il a l'information dans son cache, autrement, qui va contacter un des serveurs DNS racine au niveau mondiale qui lui va lui indiquer quel est le serveur DNS qui fait autorité pour ce nom de domaine, qui va par la suite contacter le DNS qui fait autorité pour ce domaine-là, l'information va revenir jusqu'au serveur DNS que vous utilisez sur votre PC (*recherche DNS récursive*). Le *DNS* a pour objectif final de vous renvoyer vers l'adresse IP du site, à partir du nom de domaine (*qui lui est associée*).

Il est également possible de faire l'inverse, c'est-à-dire retrouver un nom de domaine à partir d'une adresse IP, il s'agit d'une *résolution inverse*.

En plus des adresses IP, des informations complémentaires peuvent être associées aux noms de domaines comme des enregistrements dans le contexte de la lutte contre le spam (*SPF*), *RRSIG* pour la sécurité des informations du DNS (*DNSSEC*) ou *NAPTR* pour associer des numéros de téléphone à des adresses e-mail (*ENUM*).



- Il existe plusieurs types d'enregistrements DNS

- *A* : Permet de faire la correspondance entre un nom DNS et une adresse IPv4 (Exemple : fr.wikipedia.org. IN A 91.198.174.192).

- *AAAA* : Permet de faire la correspondance entre un nom DNS et une adresse IPv6 (Exemple : fr.wikipedia.org. IN AAAA 2620:0:862:ed1a::1).

- *CNAME* : Il s'agit d'un alias.

- *MX* : Permet de configurer une zone DNS donnée, les serveurs de mails sous la forme d'un nom DNS.

- *PTR* : Résolution inverse.

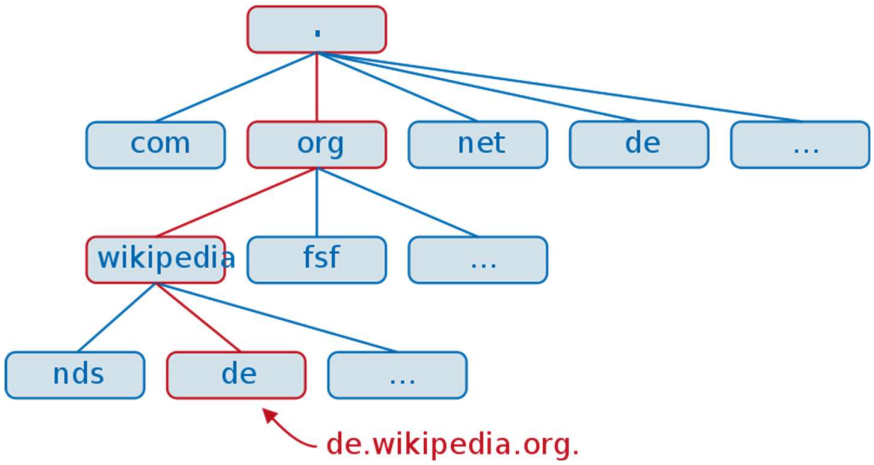
- *SN* : Serveur DNS de domaine.

- *SOA* : Infos générales sur la zone.

- *SRV* : utilisé par des services spécifiques.

- *TXT* : Cet enregistrement permet de stocker une chaîne de 1024 caractères au maximum, souvent utilisé pour le *SPF (*lutte contre le spam).

Le système des noms de domaine consiste en une *hiérarchie* dont le sommet est appelé la *racine*. On représente cette dernière par un point. Dans un domaine, on peut créer un ou plusieurs sous-domaines ainsi qu'une délégation pour ceux-ci, c'est-à-dire une indication que les informations relatives à ce sous-domaine sont enregistrées sur un autre serveur. Ces sous-domaines peuvent à leur tour déléguer des sous-domaines vers d'autres serveurs. Les domaines se trouvant immédiatement sous la racine sont appelés domaine de premier niveau (*TLD : Top Level Domain*). Les noms de domaines ne correspondant pas à une extension de pays sont appelés des domaines génériques (*gTLD*), par exemple *.org* ou *.com*. S'ils correspondent à des codes de pays (*fr, be, ch...*), ce sont des domaines de *premier niveau national*, aussi appelés *ccTLD* de l'anglais *country code TLD*.



On représente un nom de domaine en indiquant les domaines successifs séparés par un point, les noms de domaines supérieurs se trouvant à droite. Par exemple, le domaine *org.* est un *TLD*, *sous-domaine de la racine*. Le domaine *wikipedia.org.* est un *sous-domaine* de *.org.* Cette délégation est accomplie en indiquant la liste des serveurs DNS associée au sous-domaine dans le domaine de niveau supérieur.

Les noms de domaines sont donc résolus en parcourant la hiérarchie depuis le sommet et en suivant les délégations successives, c'est-à-dire en parcourant le nom de domaine de droite à gauche. Pour qu'il fonctionne normalement, un nom de domaine doit avoir fait l'objet d'une délégation correcte dans le domaine de niveau supérieur.

Qu'est-ce qu'un VLAN ?

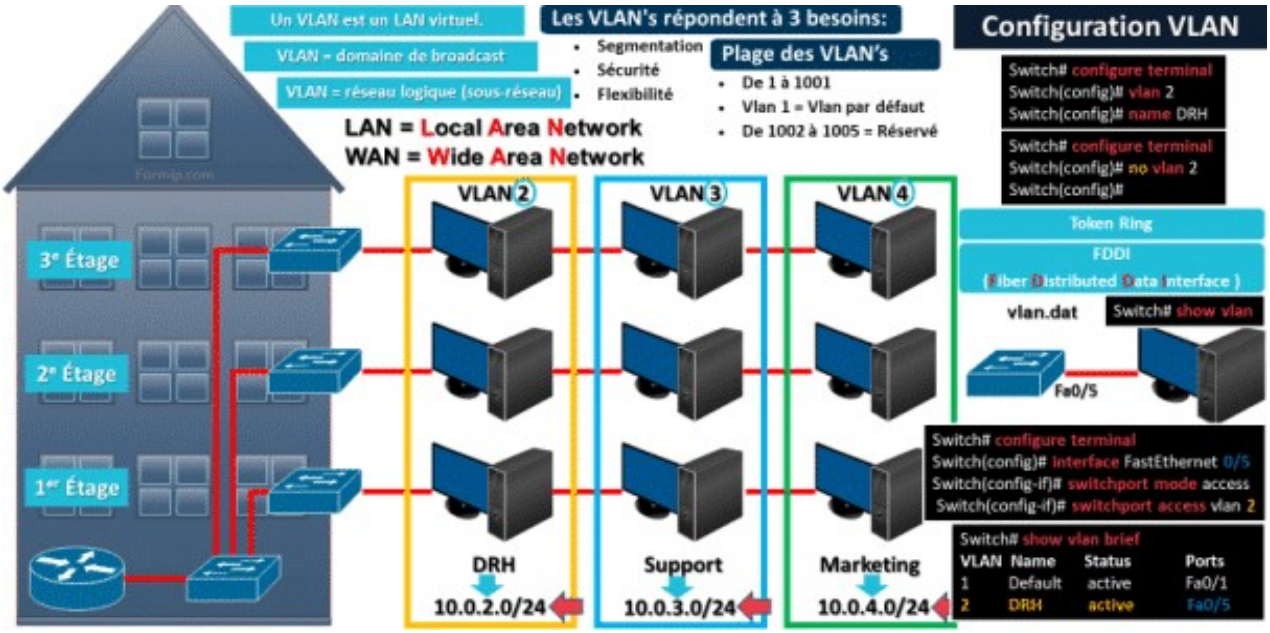
Un *réseau local virtuel*, communément appelé *VLAN* (*Virtual LAN*), est un réseau informatique logique indépendant. De nombreux VLAN peuvent coexister sur un même commutateur réseau.

La différence entre le *VLAN* et un *sous réseau* est qu'un *sous-réseau* est déterminé par *l'adresse IP* que vous utilisez et *l'adresse IP* peut être choisie par l'administrateur d'un *périphérique*. Par conséquent, cela se fait *côté client*.

Un *VLAN* est configuré *côté serveur* / *routeur*. Celui qui contrôle le *routeur* / serveur décide quel *ordinateur* / *port* est affecté à quel *VLAN*.

Un (*ou plusieurs*) *routeurs* / *serveurs centraux* peuvent être protégés logiquement (*mot de passe de connexion*) et *physiquement* (*accès à la salle des serveurs*).

La raison principale pour la mise en place d'un *VLAN* est la réduction de la taille d'un *domaine de broadcast*, de plus cela permet de créer un ensemble logique isolé pour améliorer la *sécurité*.



Comprendre le NAT

Le NAT, pour *Network Address Translation* est *pour résumer* le plus souvent utilisé par un routeur pour permettre à des machines disposant d'adresses privées (par exemple, 192.168.0.1) qui font partie d'un intranet et ne sont ni uniques ni routables à l'échelle d'Internet, de communiquer avec le reste d'Internet en utilisant vers l'extérieur des adresses externes publiques, uniques et routables.

Partons sur une analogie, prenons l'exemple d'un *réseau téléphonique d'une entreprise*, chacun des employés a un numéro de téléphone *court* pour qu'il puisse être retenu et appelé plus facilement au téléphone.

M. *DEVEDJIAN* possède le *poste* (*numéro*) *1000*, *M. BONAPARTE* le 1001, Mme. *CURIE* le *1002*, etc.

Ce sont des numéros accessibles *uniquement* en interne, pour faciliter les échanges, mais il leur faut bien un véritable numéro (*public*) pour leur permettre d'appeler à l'extérieur, cependant l'entreprise n'a pas nécessairement le budget, ou n'a tout simplement aucun intérêt à attribuer un numéro de téléphone public pour tous ses employés, sachant que certains n'auront pas besoin d'appeler vers l'extérieur par exemple.

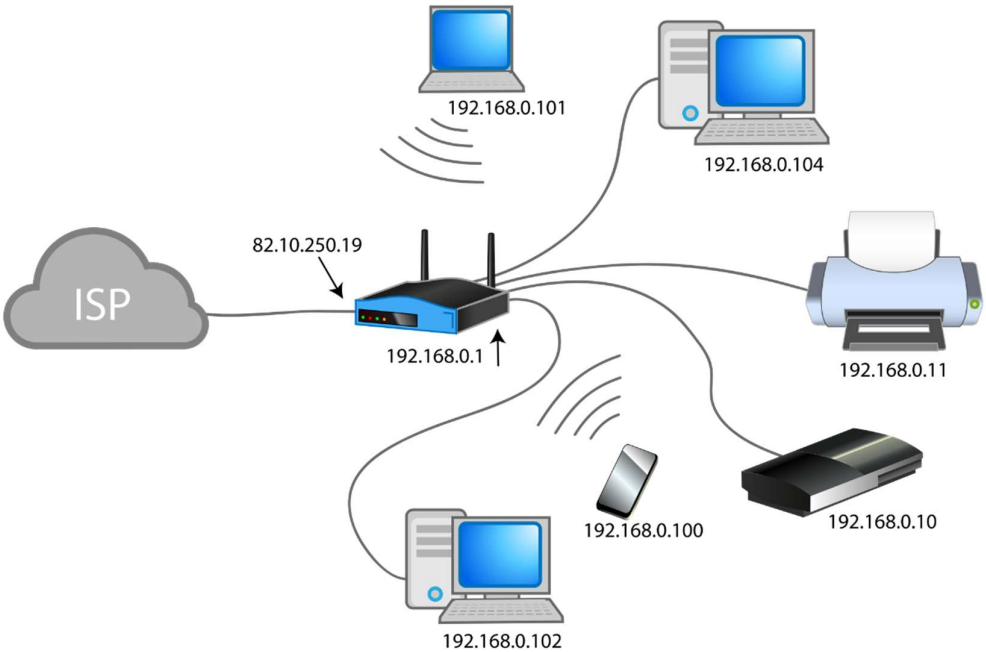
Quand un employé souhaite appeler quelqu'un à l'extérieur, il doit passer par la standardiste de la boîte qui va être contactée en interne, et qui va le mettre en relation avec l'extérieur, et le numéro affiché depuis l'extérieur sera le numéro *public* de l'entreprise mais ne verra jamais le numéro interne des employés.

C'est *en quelque sorte* le principe du *NAT*, dans le réseau interne d'une entreprise, on utilise des *adresses IP privés* qui ne peuvent pas communiquer sur *Internet*, il leur faut donc une adresse *IP publique*, c'est le rôle du *NAT*, de permettre aux *machines* qui n'ont qu'une *adresse IP privé* (*interne*), de se servir de *l'adresse IP publique* pour communiquer sur Internet.

- Comment implémenter le NAT

C'est une *passerelle* qui s'occupe de la traduction pour communiquer avec Internet, plus précisément cet équipement peut être un routeur ou un firewall. Passerelle, qui possède deux interfaces, une interface connectée au réseau interne avec une IP privée, et une interface réseau connectée à Internet avec une adresse IP publique utilisable sur Internet pour communiquer avec Internet.

La passerelle fait alors le lien entre le réseau interne et Internet en traduisant les adresses internes en adresses publiques, et vice-versa lors du retour, ainsi chaque machine du réseau interne et configurée pour transiter par la passerelle Nat pour les communications externes, lorsqu'une machine interne effectue une requête vers internet, la passerelle effectue la requête à sa place, reçois la réponse puis la transmet à la passerelle initiale.

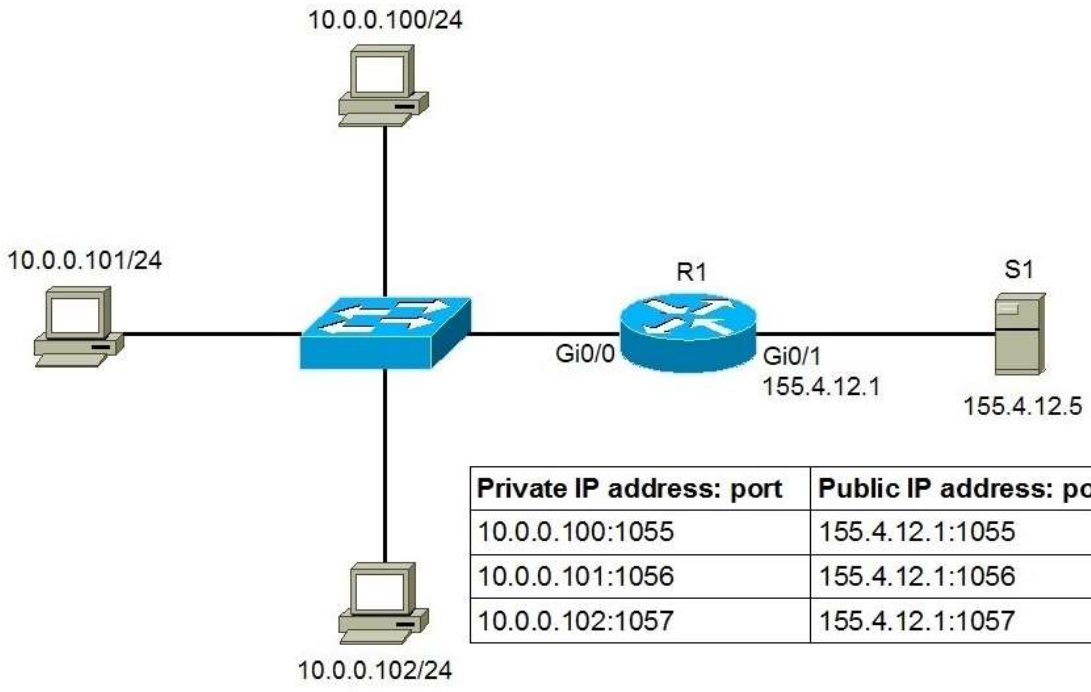


- *La suite du NAT, le PAT*

Que se passe-t-il s'il y a deux demandes de communication en même temps, et comment la passerelle sait vers qui renvoyer la communication lors du retour ?

La passerelle va faire un *PAT* (*Port Address Translation*) lorsqu'elle reçoit la communication, pour résumer la passerelle va associer dynamiquement à chaque demande de communication l'adresse IP privée du demandeur et un numéro de port.

Prenons l'exemple d'un *PC 1* (*10.0.0.100*) et d'un *PC 2* (*10.0.0.101*), le *PC 1* souhaite communiquer, la passerelle va attribuer dynamiquement un port source à cette communication, par exemple le port *1055*, de ce fait, au retour les réponses renvoyées vers l'adresse IP publique de NAT, sur le port *1055* seront renvoyées au PC1, pour le second client, la passerelle va lui attribuer un port différent, par exemple *1056* et au retour, les retours sur le port *1056* seront donc renvoyés sur le second client (*PC2*).



Private IP address: port	Public IP address: port
10.0.0.100:1055	155.4.12.1:1055
10.0.0.101:1056	155.4.12.1:1056
10.0.0.102:1057	155.4.12.1:1057

La passerelle

Comme vu précédemment, deux hôtes ne se situant pas dans le même *sous-réseau* ne peuvent pas communiquer *directement*.

Il faut que quelque chose intervienne entre les deux pour transmettre à l'un, les données au nom de l'autre.

Il s'agit de la *passerelle* (*gateway*) : *un service qui connecte plusieurs réseaux / sous-réseaux*. Cette position fait donc que la *passerelle* se situe à califourchon entre *plusieurs réseaux / sous-réseaux*.

Plus conventionnellement, une *passerelle* est le nom générique d'un dispositif permettant de relier deux réseaux informatiques de types différents, par exemple un *réseau local* et le *réseau Internet*.

Dans un réseau comprenant plusieurs *routeurs*, la *passerelle* par défaut (*default gateway*) est l'interface du routeur vers laquelle sont dirigés tous les paquets dont on ne connait pas la *route* à emprunter pour atteindre le réseau dans lequel se trouve le destinataire.

Chaque *routeur* a une table de routage. Pour faire simple, les *paquets* représentent des parties de vos données. En fait, lorsque vous envoyez des données sur un réseau, celles-ci sont découpées en plusieurs portions et chaque portion est appelée un *paquet*.

Quant à la table de routage, il s'agit d'une liste des différentes "*routes*" (*chemins*) vers d'autres réseaux / *sous-réseaux*. Ces définitions sont très simplifiées pour vous permettre de comprendre le principe.

Prenons un exemple concret pour illustrer le mode de fonctionnement d'une *passerelle*.

- PC 1 : *192.168.1.5/24*
- PC 2 : *72.40.2.1/16*

Ils n'appartiennent pas au même sous-réseau et ne peuvent donc pas communiquer : il leur faut une *passerelle*.

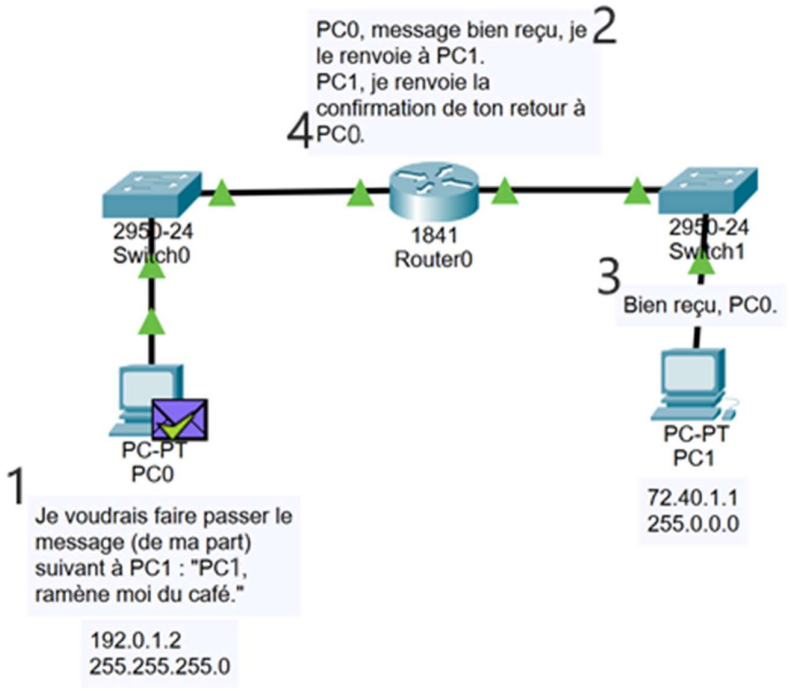
- Le fonctionnement

Le *PC1* a recours à un processus nommé *ANDing* (*ET logique*, cf. *Comment calculer l'adresse du réseau, du broadcast, etc.),* pour déterminer si *PC2*, avec qui il veut communiquer, est dans le même sous-réseau que lui. Il réalise que ce n'est pas le cas, il va donc transférer son message à la passerelle en lui indiquant l'adresse du destinataire.

Supposons que ce soit un routeur qui offre ce service. Il a 2 interfaces. Pour que la communication puisse avoir lieu, une de ses interfaces doit être dans le même sous-réseau que *PC1* et l'autre dans le même que *PC2*. Voici une configuration possible pour ce routeur :

- *Interface Routeur 1* : 192.0.1.1/24 *et*
- *Interface Routeur 2* : 72.40.1.2/8
- *PC1* : 192.0.1.6/24, *et*
- *PC2* : 72.40.1.1/8

Avec une telle configuration, *PC1* et *PC2* peuvent à présent communiquer. Quand *PC1* voudra parler à *PC2*, il vérifiera grâce au *ET logique* si le destinataire est dans le même sous-réseau. Si oui, il enverra son message directement à son adresse IP, sinon, il l'envoie à la passerelle en lui demandant de transmettre à bon port. La passerelle étant entre les 2, cela ne pose pas de problème.



Qu'est-ce qu'un *Proxy* ?

Un *proxy* joue le rôle d'intermédiaire en se plaçant entre deux hôtes.

Admettons que *PC1* souhaite se connecter au site *fr.wikipedia.com* (*91.198.174.194*), en passant par un proxy, *PC1* va *d'abord* envoyer une requête sur le *serveur proxy*, qui va *à son tour* effectuer la demande pour *PC1* à *Wikipedia* (*91.198.174.194*), qui recevra donc une requête de la part du serveur proxy.

En entreprise ou dans les établissements scolaires, il est presque systématique que l'accès *Internet* se fasse à travers un *serveur proxy*.

L'utilisateur ne voit le plus souvent pas la différence, sauf quand il tente de naviguer sur un site interdit, auquel cas il pourra recevoir un message d'erreur, un tel *proxy* est appelé *proxy filtrant*.

À l'inverse, un *proxy* peut aussi servir à *contourner* les filtrages. Supposons le cas d'un pays qui bloque l'accès à certains sites *considérés comme subversifs* (comme la Turquie, le Pakistan, etc.) et qu'ils effectuent ce filtrage uniquement en se basant sur l'adresse du site, en utilisant un proxy comme intermédiaire *situé dans un autre pays*, le filtrage peut être contourné.

Le principe fonctionne également dans l'autre sens. Si un site n'accepte *que* les utilisateurs d'un certain pays, en passant par un *proxy* situé dans ce même pays, un utilisateur français pourra tout de même consulter ce site.

Un des rôles du *proxy* peut être de compliquer la remontée vers l'internaute (*anonymisation*). Cependant, certaines techniques permettent de remonter à travers un *proxy,* de plus*,* la *plupart* des *proxys* qui offrent des services vous permettant d'être *anonyme* en ligne collectent en réalité une grande quantité de *données personnelles* et *d'identification*, qu'ils peuvent garder pour vous dénoncer aux autorités à la moindre demande (de la police, par exemple), ou pire encore les revendre.

Sachez que c'est également le cas avec de nombreux VPN, ne vous croyez pas invulnérable sous prétexte que vous utilisez un VPN. Sachez que la plupart des VPN *même payants*, conservent *certaines* données permettant de vous identifier en cas de poursuite judiciaire.

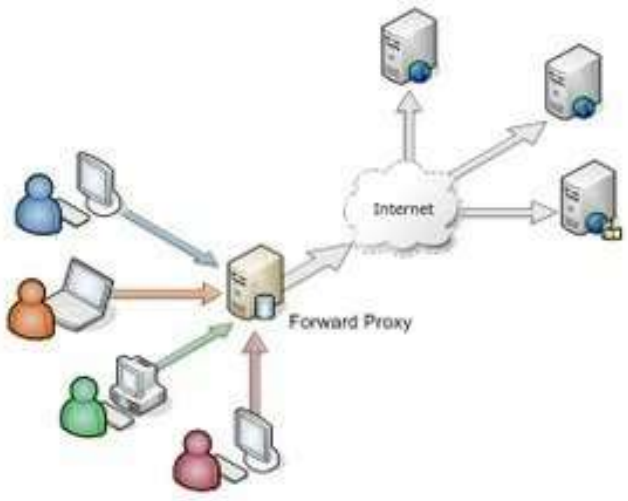
- Le *proxy inversé* (*reverse proxy*)

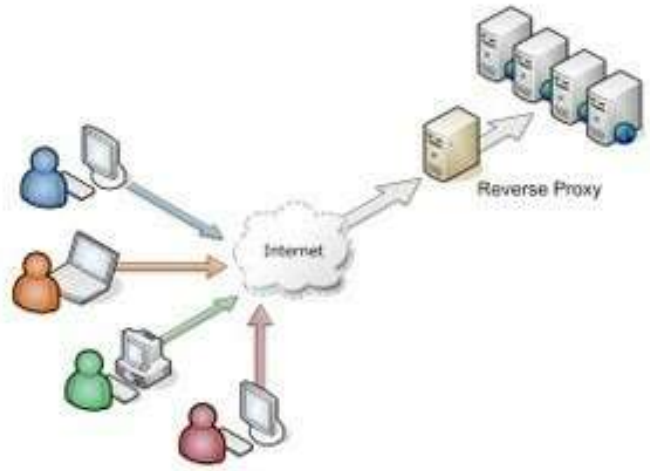
Comme nous avons pu le voir précédemment, un serveur proxy agit comme un intermédiaire entre vous et le serveur de votre choix.

Un proxy transmet les connexions que vous souhaitez établir : il transmet vos demandes, reçoit les réponses des sites et services que vous utilisez, puis vous les envoie.

En revanche, un *proxy inverse* reçoit les requêtes d'un client sur un autre réseau, les transmet à un serveur interne, reçoit le résultat, puis le relaie au client.

Autrement dit, si vous avez des difficultés à cerner la différence entre un *serveur proxy* et un *proxy inverse*, les deux font exactement (*globalement*) la même chose, mais le *proxy inverse* agit à l'envers : un proxy direct agit au nom d'un client, tandis qu'un proxy inverse agit au nom du serveur.

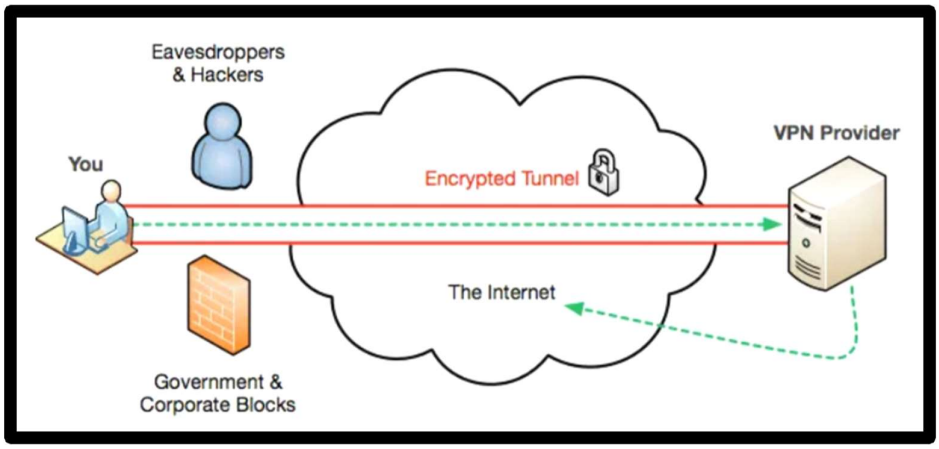




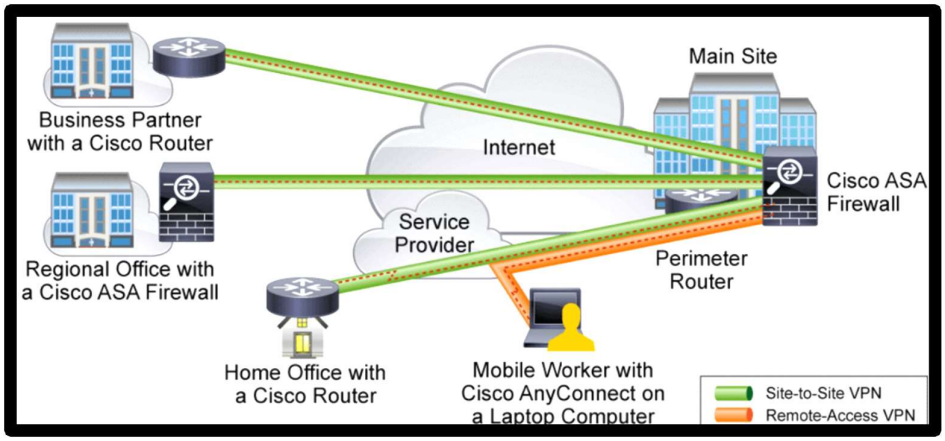
Qu'est-ce qu'un VPN

Un réseau privé virtuel (ou virtual private network), est un système permettant (pour résumer grossièrement) de connecter deux "choses" de manière sécurisée.

En *théorie*, hormis le *gérant* derrière le *VPN*, personne n'est censé avoir la possibilité de *comprendre* les données échangées, que ce soit votre *FAI* (*fournisseur d'accès à Internet*) ou un pirate *sniffant* le réseau.



Au-delà de la tendance "NordVPN" vous expliquant qu'un VPN n'est globalement utile qu'à masquer votre adresse IP en ligne, un VPN permet d'interconnecter divers réseaux avec ce même tunnel chiffré de manière sécurisé pour échanger des informations, et par échanger des informations je parle *également* d'accéder, *par exemple* aux ressources d'un réseau d'entreprise.



Le *protocole SMTP*

Simple Mail Transfer Protocol (*SMTP, protocole simple de transfert de courrier*) est un protocole de communication utilisé pour transférer le courrier électronique vers les serveurs de messagerie électronique.

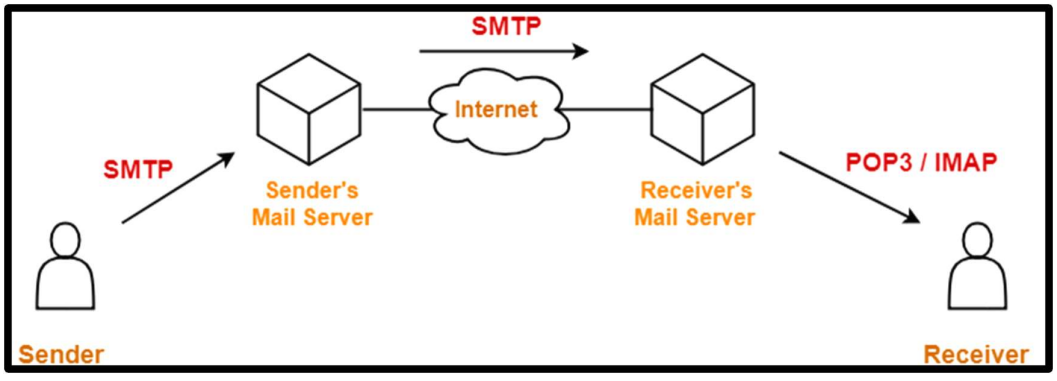
On commence par spécifier l'expéditeur du message, puis le ou les destinataires d'un message, puis, en général après avoir vérifié leur existence, le corps du message est transféré. Il est possible de tester un serveur *SMTP* en utilisant la commande *telnet* sur le *port 25* d'un serveur distant.

SMTP utilise *TCP* pour le transfert des données. *SMTP* ne permet pas de récupérer à distance des courriels arrivés dans une boîte aux lettres sur un serveur. Les standards *Post Office Protocol* (*POP*) et *IMAP* ont été créés dans ce but.

Le transfert de messages entre serveurs de messagerie électronique se fait généralement sur le port 25.

Les serveurs utilisent les enregistrements *MX* des serveurs *DNS* pour acheminer le courrier.

Les administrateurs de serveur peuvent choisir si les clients utilisent le port *TCP 25* (*SMTP*) ou le *port 587* (*submission*), pour relayer le courrier sortant vers un serveur de messagerie.



- Le *protocole POP*

Les protocoles *POP* et *IMAP* servent *globalement* à faire la même chose, simplement, chacun présente des avantages que l'autre n'a pas.

POP (*Post Office Protocole* : *protocole de bureau de poste*) a l'avantage d'être simple et efficace et surtout, il est supporté par tous les clients de messagerie.

Pour comprendre le rôle assuré par ce protocole, nous allons examiner le rôle d'un bureau de poste dans la vie courante.

Quels sont les services offerts par un bureau de poste ?

Le bureau de poste a pour fonction principale de traiter les courriers : il les reçoit et les distribue à leurs destinataires respectifs. Il est également en contact avec les autres bureaux de poste distants avec qui il peut communiquer.

En réseau, en ce qui concerne la messagerie électronique, le protocole *POP* fait plus ou moins la même chose. La différence avec un véritable service postal est que deux bureaux de poste de villes différentes peuvent échanger du courrier, alors que le protocole *POP* ne peut pas en envoyer.

POP n'est qu'un protocole de retrait : il permet d'aller chercher un mail se situant sur un serveur de messagerie, mais pas d'en envoyer. L'envoi est assuré par le protocole *SMTP*.

Il existe trois versions de ce protocole :

- *POP1 (cf. RFC 918)*
- *POP2 (cf. RFC 937)*
- *POP3 (cf. RFC 1939)*

Le protocole *POP* permet bien sûr de récupérer son courrier, mais aussi d'en laisser une copie sur le serveur. Cela est particulièrement utile si l'on ne peut plus accéder pour une raison quelconque (*une panne par exemple*) aux e-mails déjà téléchargés : on peut toujours les télécharger de nouveau.

- Le *protocole IMAP*

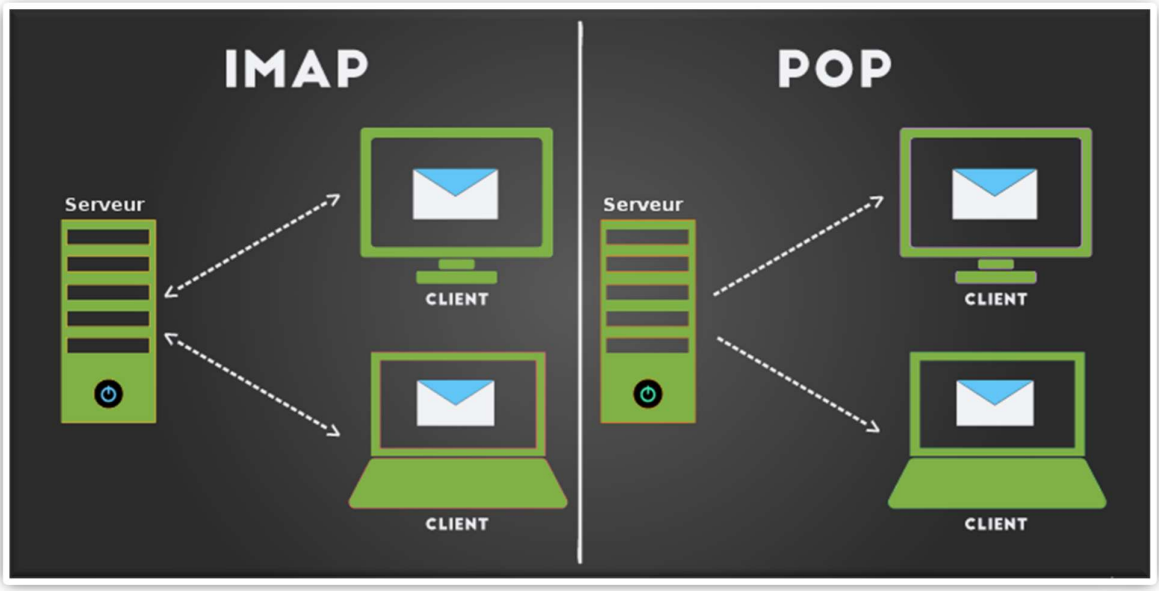
IMAP (*Internet Message Access Protocol*) est un protocole de lecture d'e-mails. Contrairement à POP, il n'a pas été conçu pour recevoir des messages mais pour les *consulter directement* depuis un serveur.

Cette consultation s'apparente à du *clouding*, c'est-à-dire l'accès par *Internet* à des données qui ne se trouvent pas sur notre disque dur.

IMAP est assez avancé puisqu'il permet de gérer ses messages directement sur un serveur distant pour organiser nos messages en dossiers, par exemple. Il supporte également *TLS*.

Dans le cas *d'IMAP*, le *clouding* est à la fois un *avantage* et un *inconvénient* : on peut accéder à ses messages depuis n'importe quel ordinateur - à condition d'être connecté à son compte de messagerie.

Quelques clients permettent néanmoins de télécharger les messages pour pallier ce problème, mais certains clients de messagerie ne gèrent pas très bien le protocole IMAP.



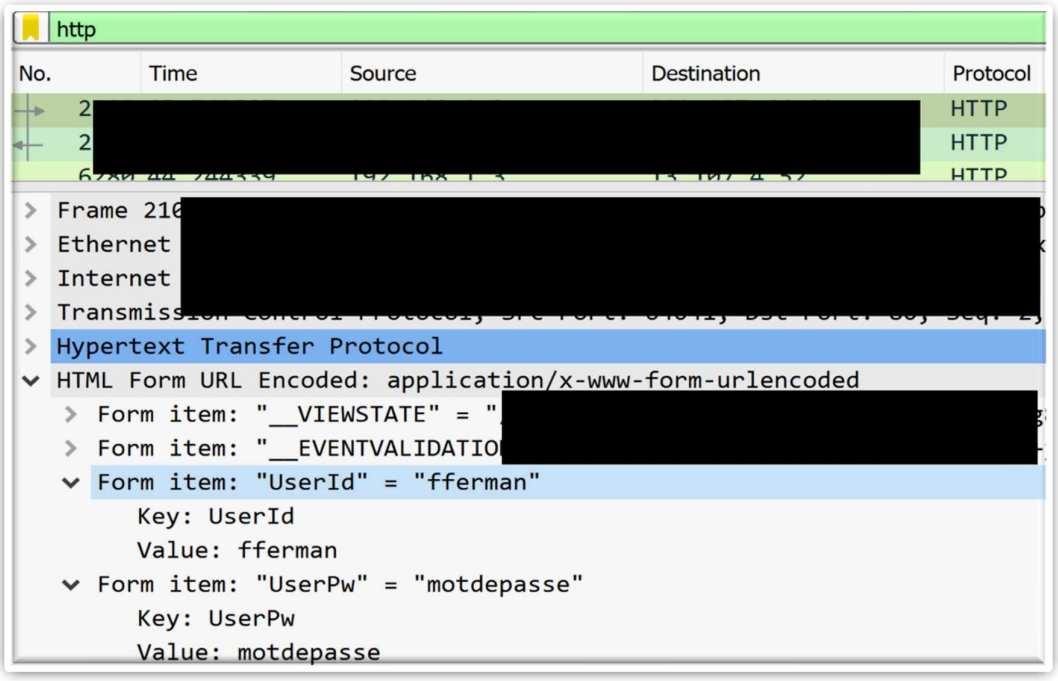
Le *protocole TLS*

Le *protocole TLS* pour *Transport Layer Security* permet entre autres de sécuriser d'autres protocoles de *l'Application Layer*.

En effet, *HTTPS* est par exemple *en quelque sorte* un *alias* pour *HTTP + TLS*. Tout comme *SMTPS* (*SMTP + TLS*) ou *FTPS* (*FTP + TLS*).

Certaines personnes ont tendance à confondre *TLS* et *SSL*, étant une ancienne version de *TLS banni définitivement* car *comportant trop de failles de sécurité*.

Le protocole *TLS* est utilisé pour empêcher un tiers d'intercepter des données sensibles transitant par le réseau tel que des mots de passe, numéro de carte lors d'un paiement, etc.



Voici un exemple de communication non chiffrée avec TLS (HTTPS), une personne mal intentionnée aura la possibilité de voir vos échanges de cette façon.

TLS fonctionne suivant un mode *client-serveur*. Il permet de couvrir les principaux objectifs de sécurité suivants :

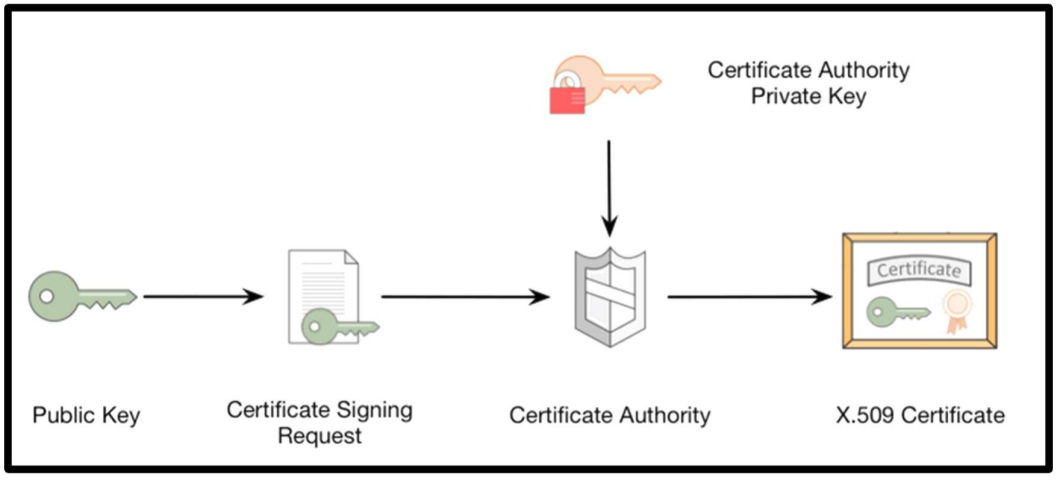
- *La confidentialité des données échangées*, personne ne doit avoir la possibilité de *lire les données* (*car chiffrées, contrairement à notre exemple ci-dessus*)

- *L'authenticité* ou *l'intégrité* des données échangées, un utilisateur malveillant ne doit pas avoir la possibilité de modifier les données en cours de route par exemple.

- *L'authentification*, qui est tout aussi importante que les deux autres objectifs ci-dessus, en effet vous devez être sûr d'envoyer les informations au véritable site et non pas une copie frauduleuse du site en question par exemple.

Dans la majorité des cas, l'utilisateur authentifie le serveur *TLS* sur lequel il se connecte. Cette authentification est réalisée par l'utilisation d'un certificat numérique *X.509* (*X.509* est une norme spécifiant les formats pour les certificats à clé publique, les listes de révocation de certificat, les attributs de certificat, et un algorithme de validation du chemin de certification), délivré par une *autorité de certification* (*CA/AC*).

Des applications web peuvent utiliser l'authentification du poste client en exploitant TLS. Il est alors possible d'offrir une authentification mutuelle entre le client et le serveur. Le certificat client peut être stocké au format logiciel sur le poste client ou fourni par un dispositif matériel (*carte à puce, token USB*). Cette solution permet d'offrir des mécanismes d'authentification forte.



Lorsqu'un utilisateur se connecte à un site web qui utilise *TLS*, les étapes suivantes ont lieu :

- Le *navigateur du client* envoie au *serveur* une demande de mise en place de connexion sécurisée par *TLS*.

- Le *serveur* envoie au client son *certificat* : celui-ci contient sa *clé publique*, *ses informations* (*nom de la société, adresse, pays*...) ainsi qu'une *signature numérique*.

- Le *navigateur* du client tente de *vérifier* la *signature numérique* du *certificat* du serveur en utilisant les clés publiques contenues dans les certificats des *autorités de certifications* (CA/*AC*) intégrées par défaut dans le navigateur.

- Si aucune d'entre elles ne fonctionne, le navigateur web tente de vérifier la signature numérique du certificat du serveur à l'aide de la clé publique contenue dans celui-ci, en cas de réussite, cela signifie que le serveur web a lui-même signé son certificat. Un message d'avertissement s'affiche alors sur le navigateur web, prévenant l'utilisateur que l'identité du serveur n'a pas été vérifiée par une autorité de certification et qu'il peut donc s'agir potentiellement d'un site frauduleux, en cas d'échec, le certificat est invalide, la connexion ne peut pas aboutir.

- Le navigateur du client génère une clé de *chiffrement symétrique*, appelée *clé de session*, qu'il chiffre *à l'aide de la clé publique* contenue dans le certificat du serveur puis transmet cette clé de session au serveur.

- Le serveur *déchiffre* la clé de session envoyée par le client grâce à sa clé privée.

- Le client et le serveur commencent à s'échanger des données en chiffrant celles-ci avec la clé de session qu'ils ont en commun. On considère à partir de ce moment que la connexion *TLS* est alors établie entre le client et le serveur.

- Une fois la connexion terminée (*déconnexion volontaire* de l'utilisateur ou *si durée d'inactivité trop élevée*), le serveur révoque la clé de session.

La *simulation d'architectures réseaux*

La *simulation d'architectures réseaux* a une grande importance, aussi bien pour *l'apprentissage* (des réseaux) que pour la mise en production.

Je m'explique, en *développement Web* par exemple, si vous avez une modification à effectuer, vous allez réaliser des tests en amont, *en local*, *avant de mettre à jour le code sur vos serveurs*.

L'architecture de systèmes et réseaux, même simple, nécessite également la réalisation de tests avant le lancement en production.

Cette bonne pratique est *indispensable* pour vous éviter de nombreuses erreurs en production.

Autrement, en ce qui concerne *le cadre de l'apprentissage*, imaginez-vous devoir acheter des dizaines de routeurs, commutateurs, câbles, ordinateurs, cela mènera inévitablement à un cout assez conséquent.

Il existe de nombreux outils tels que G*NS3, Packet Tracer, Virl, Marionnet, Eve, etc.* qui vont vous permettre de créer vos réseaux, les tester, les installer, et paramétrer des switchs, des routeurs, et même des serveurs.

Le simulateur *GNS3* vous permet même de connecter votre hyperviseur de machines virtuelles (*Vmware, Virtualbox*).

Pour résumer, vous pourrez architecturer vos réseaux simples et complexes, et les simuler virtuellement, comme si vous y étiez.

- La simulation, l'émulation et la virtualisation

Il est important que vous puissiez comprendre certains concepts, dont trois notions, qu'il faut savoir différencier :

- *La simulation*

La simulation est une représentation fictive de la réalité. Il s'agit d'imiter une situation. Par exemple comme avec les simulateurs de vol (*qui permettent aux pilotes de s'entraîner sans risque*).

La simulation réseau revient à reproduire l'architecture d'un réseau et cela sans utiliser de machine physique. En d'autres termes, comme le pilote d'avion apprend à voler sans avion, vous allez apprendre à gérer un réseau sans machine physique.

La simulation passe par un logiciel qui calcule (*modélise*) et donc prédit les événements qui seraient amenés à se produire en prenant en compte leurs caractéristiques.

- *L'émulation*

L'émulation permet non pas de modéliser, mais bel et bien de reproduire à l'identique le comportement d'un logiciel et son architecture matérielle.

Ce terme n'apparaît qu'en informatique. Comme par exemple pour les émulateurs de veilles consoles (*Game Boy, PSP…),* les machines s'exécuteront exactement comme ils le feraient dans le monde réel.

- *La virtualisation*

La virtualisation *reprend* les concepts de *l'émulation*, à quelques différences notables près. Elle utilise *l'architecture* du *système hôte,* alors que *l'émulation* la reproduisait de manière *logicielle*.

En d'autres termes, l'émulation d'une *PSP* reproduit le processeur d'une *PSP,* alors que la virtualisation utilise directement le processeur de votre ordinateur menant à un *gain de performances*.

L'Active Directory

L'active directory (*AD*) étant créé par *Microsoft,* est un service d'annuaire (*LDAP*) destiné aux environnements *Windows*.

Outre les fonctions *d'authentification* et *d'autorisation* dont il est doté, il s'agit d'une *base de données distribuée* et *hiérarchisée* qui *partage des informations relatives à l'infrastructure* permettant de *localiser*, de *sécuriser*, de *gérer* et *d'organiser des ressources ordinateur et réseau*, dont des *fichiers*, *utilisateurs*, *groupes*, *périphériques* et *appareils réseau*.

Pour résumer, l'active directory comporte trois principaux avantages :

- *Une administration centralisée et simplifiée* : la gestion des objets, notamment des comptes utilisateurs et ordinateurs est simplifiée, car tout est centralisé dans l'annuaire Active Directory. De plus, on peut s'appuyer sur cet annuaire pour de nombreuses tâches annexes comme le déploiement de stratégies de groupe sur ces objets.

- *Unifier l'authentification* : un utilisateur authentifié sur une machine, elle-même authentifiée, pourra accéder aux ressources stockées sur d'autres serveurs ou ordinateurs enregistrés dans l'annuaire (*à condition d'avoir les autorisations nécessaires*). Ainsi, une authentification permettra d'accéder à tout un système d'information par la suite, surtout que de nombreuses applications sont capables de s'appuyer sur l'Active Directory pour l'authentification. Un seul compte peut permettre un accès à tout le système d'information, ce qui est fortement intéressant pour les collaborateurs.

- *Identifier les objets sur le réseau* : chaque objet enregistré dans l'annuaire est unique, ce qui permet d'identifier facilement un objet sur le réseau et de le retrouver ensuite dans l'annuaire.

- *Référencer les utilisateurs et les ordinateurs* : l'annuaire s'apparente à une énorme base de données qui référence les utilisateurs, les groupes et les ordinateurs d'une entreprise. On s'appuie sur cette base de données pour réaliser de nombreuses opérations : authentification, identification, stratégie de groupe, déploiement de logiciels, etc.

Mentions légales

La plupart des informations fournies dans ce document viennent de :

- « *Moi-même* », plus précisément mes connaissances acquises : *en autodidacte, en cours, ou au travail*.
- *Wikipédia* (*https://www.wikipedia.org*).
- Des sites spécialisés tels que : *IT-Connect*, *FormIP*...
- Des vidéos : sur *YouTube* (*avec la chaine d'Adrien Linuxtricks, par exemple*), LinkedIn Learning, *Udemy* ou encore *OpenClassrooms*.