Réseau
By VINCENT Steeve

I. Expédition de donnée

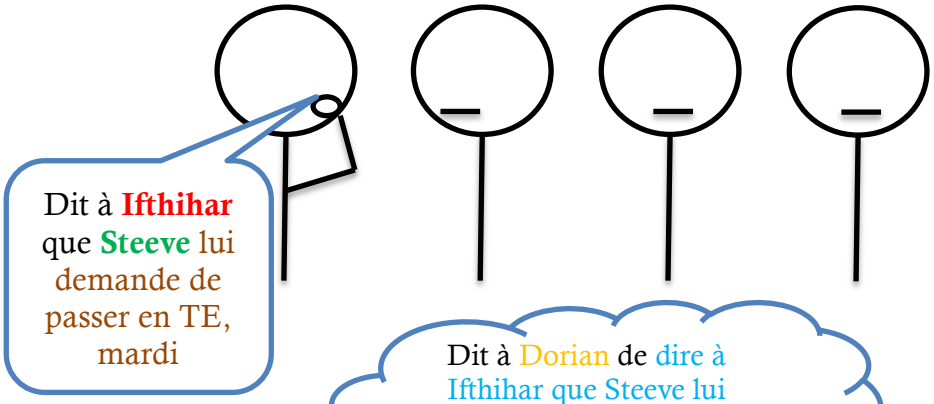
Pour envoyer des données vers un autre ordinateur dont on connaît l'adresse les (oui les) ordinateurs vont encapsuler les informations sur plusieurs couche.

L'expéditeur encapsule d'abord les données dans un **« Paquet »**, ce paquet est une suite d'information qui contient essentiellement l'adresse de l'expéditeur et celui du destinataire. Rien de bien difficile.

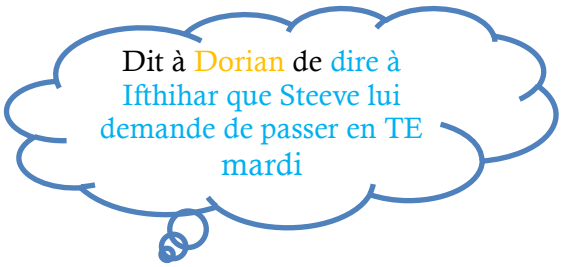
Le paquet ne sait pas encore quel chemin il va prendre. Il va être transmis à un ordinateur local (directement relié par un câble croisé) puis cet ordinateur va retransmettre ce paquet à un autre ordinateur qui lui est lié qui va lui aussi le retransmettre à un autre, ainsi de suite jusqu'à ce que le paquet arrive à destination.

Pour ce faire chaque ordinateur va encapsuler le paquet dans une **« Trame »** qui contient l'adresse de l'ordinateur qui possède le paquet et celui à qui va être transmis le paquet. Celui qui reçoit le paquet va décapsuler la Trame (on supprime la trame actuelle) et en recrée une autre (donc ainsi de suite jusqu'à ce que le destinataire reçoive le paquet).

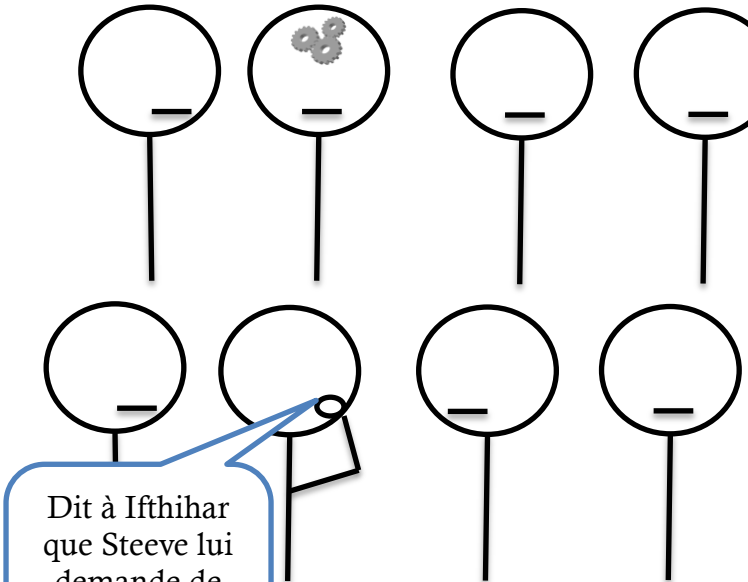
La couche **« bit »** concerne l'ordinateur en lui-même, il s'agit d'une couche qui contient l'instruction qu'il doit lui-même suivre. Il n'est pas très important.



L'expéditeur crée le paquet et le transmet à son voisin via une trame pour qu'il retransmette le paquet. Le paquet contient le nom du destinataire, celui de l'expéditeur et les données.



La personne qui reçoit le paquet (en dehors du destinataire) va créer une couche bit qui va lui lister les instructions. Les instructions lui demandent de crée une trame destinée à son voisin et qui contiendra le paquet.



Chaque personne va donc répéter l'action jusqu'à ce que l'information arrive à son destinataire.

Il existe d'autres couches, mais elles sont traitées que par certains protocoles. On choisit le protocole qui nous convient le plus, par exemple certains vont gérer aussi la correction des informations reçus et d'autre vont gérer la synchronisation des réceptions (On reçoit les données dans un ordre précis).

II. Routage

Pour pouvoir faire communiquer plusieurs ordinateurs entre eux, il faut gérer les adresses des ordinateurs.

Les adresses MAC sont les adresses physiques des ordinateurs, elles ne dépendent pas du réseau mais des câbles Ethernet branchés à l'ordinateur (Il s'agit en fait de l'adresse de chaque port connecté). C'est qu'utilise les ordinateurs pour communiquer entre eux mais nous n'avons pas à les connaitre pour envoyer des données à un autre ordinateur. Nous, on utilise **les adresses IP**, donc qui dépende du réseau elles sont définis par défauts.

On peut toutefois changer l'adresse MAC :

ifconfig eth0 hw ether *Adresse-MAC*

ou

ip link set eth0 adress *Adresse-MAC*

Un ordinateur qui est connecté à un réseau ne connait pas forcement tous les autres ordinateurs. S'il veut envoyer des données à un autre ordinateur dont il connait l'adresse IP, il lui faut définir **une route**. Pour cela on lui définit l'adresse de l'ordinateur, dont on connaît le chemin pour lui communiquer des informations, qui retransmettra les informations de telle sorte à ce que les informations aille jusqu'au réseau où se trouve l'ordinateur.

Pour cela :

ip route add *adresse du réseau/masque du réseau sous forme réduite* **via** *adresse de l'ordinateur qui va retransmettre les données* **dev** *port qui est connecté à cette ordinateur*

On peut aussi mettre une route par défaut, où on définis l'ordinateur qui va retransmettre tous les données dont on connais pas le chemin (par exemple on définis internet pour retrouver le destinataire)

Pour cela :

ip route add default via *adresse de l'ordinateur par défaut*

Ces commandes permettent de configurer un réseau mais de **façon volatile**, c'est-à-dire qu'il faudra les reconfigurer lorsqu'il vous redémarrerez l'ordinateur.

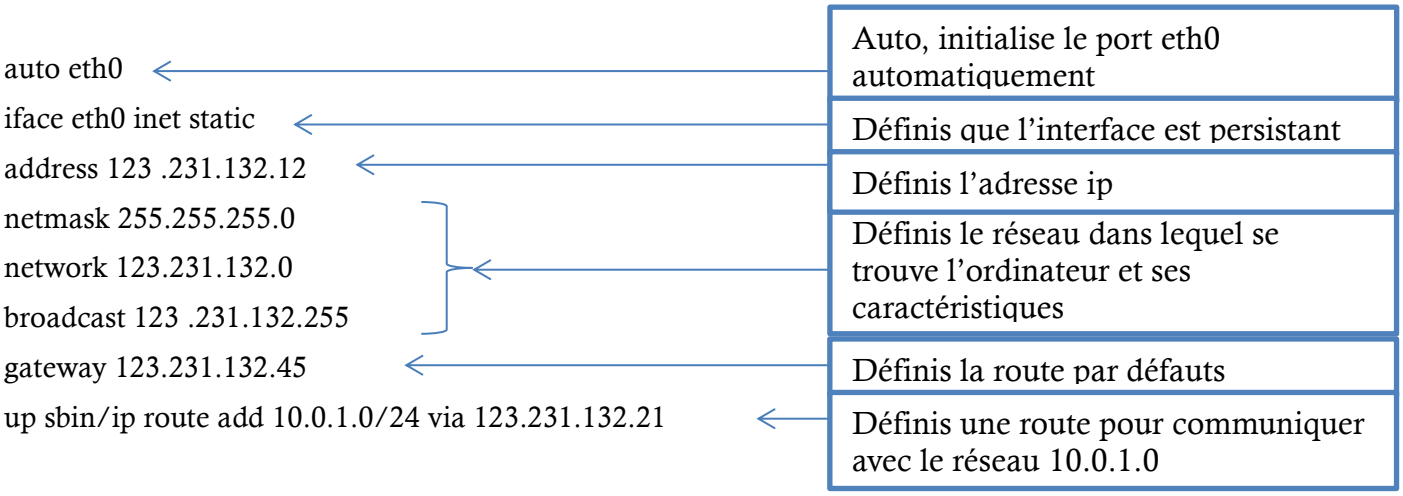
Pour configurer l'ordinateur de **façon persistante**, il faut lui dire de se configurer automatiquement à chaque fois qu'il démarre.

Pour cela il faut configurer le fichier suivant : **/etc/network/interface**

Donc nano /etc/network/interface ->

auto lo

iface lo inet loopback



III. Les adresses

Les réseaux utilise l'adressage ip, il y a les **adresses IPv4**, codées sur 32 bit en binaire, et les **adresses IPv6**, codées sur 128 bits en hexadécimal.

On utilise plus couramment l'IPv4 mais l'IPv6 commence à se généraliser donc regardons un peu ce type d'adressage.

Elles sont codées sous formes de 8 blocs de 16 bits, un bloc contient 4 caractères hexadécimaux codés respectivement sur 4 bits.

Exemples d'adresses :

2001 :0ac8 :00c0 :0000 :0000 :0000 :2301 :a201

Qu'on peut écrire aussi :

2001 :ac8 :c0 ::2301 :a201

Il s'agit d'une forme simplifié qui consiste à ne pas mettre les 0 qui sont à gauches dans un bloc (0ac8 devient ac8) et de remplacer une suite de 0 par :: (0000 :0000 :0000 devient ::)

Mais revenons à nos adresses IPv4, donc codées sous formes de 4 blocs de 8 bits.

Exemple :

128.192.16.4 ⇔ (en binaire) 1000 0000.1100 0000.0001 0000.0000 0001

127.0.0.1 (adresse de **loopback**, vue précédemment)

⇔ 0111 1111.0000 0000.0000 0000.0000 0001

Une adresse est composée de la partie réseau et de la partie machine.

La partie machine, qui est à droite, permet de distinguer tous les ordinateurs d'un même réseau local. Deux machines n'auront donc pas la même identification dans un même réseau.

La partie réseau, qui est à gauche, permet de déterminée dans quelle réseau locale se trouve l'ordinateur, les ordinateurs d'un même réseau possède donc la même adresse réseau. De même que gens d'une même rue possède des adresse dans une partie est commune à tous.

Cette partie peut être distinguée de celle de la machine à l'aide du **masque**. Ce masque est composé de la même façon que l'adresse (donc soit IPv4, soit IPv6). La partie gauche du masque est composé de 1 et montre la longueur de la partie réseau.

Par exemple si mon adresse est 1000 0000. 1100 0000. 0001 0000. 0000 0001 et mon masque est
1111 1111. 1111 1111. 0000 0000. 0000 0000

Alors la partie réseau de mon adresse est 1000 0000. 1100 0000 ⇔ 128.192 et l'autre partie représente l'identifiant de la machine dans le réseau soit 16.4

Il faut donc convertir les adresses en binaires si on veut déterminer les différentes parties d'une adresses.

Le masque peut s'écrire sous forme développé : 255.255.0.0 et sous forme réduite : /16,

Le nombre après le « / » représente le nombre de 1 que possède le masque, cela représente donc sur combien de bits la partie réseau s'entend.

Dans un réseau, il y a deux adresses qui sont réservé et qui ne peuvent être attribuées à une machine. L'adresse du réseau et l'adresse de broadcast (qui permet de communiquer avec tous les ordinateurs du réseau en même temps).

L'adresse du réseau est composée de la partie réseau et de 0.

Exemple 1000 0000. 1100 0000. 0000 0000. 0000 0000 ⇔128.192.0 .0

L'adresse de broadcast à l'inverse, est composée de la partie réseau et de 1.

Exemple 1000 0000. 1100 0000. 1111 1111. 1111 1111 ⇔128.192.255.255

Petit exercice :

Trouver l'adresse réseau de 145.189.21.180 sachant que son masque est /26

On va donc convertir d'abord en binaire cet adresse.

145 = 128 +16 + 1 = 1001 0001

189 = 128 + 32 + 16 + 8 + 4 +1 = 1011 1101

21 = 16 + 4 + 1 = 0001 0101

180 = 128 +32 +16 +4 = 1011 0100

145.189.21.180 = 1001 0001. 1011 1101.0001.0101.1011 0100

Sachant que les 26 premiers bits représentent l'adresse du réseau

L'adresse réseau est 1001 0001. 1011 1101.0001.0101.1000 0000 = 145.189.21.128

Si on veut savoir combien d'ordinateur ce réseau peut contenir il suffit de voir combien de bits on peut modifier (donc, pour l'exemple en haut, les bits en noirs (pas de sous-entendu merci)). Pour chacun de ces bits on peut attribuer soit 1 soit 0 donc le nombre de combinaisons possibles est $2^6 = 64$. De ce nombre on enlève les adresses réservée (adresse de broadcast et du réseau) et on obtient 62 machines disponible.

IV. Protocole

Le protocole ARP

Le protocole ARP est utilisé pour toutes les communications. Ce protocole consiste à trouver l'adresse MAC d'un ordinateur dont on connaît l'adresse IP.

La commande **ping** utilise le protocole ARP, elle envoie à l'adresse de broadcast de son réseau une requête pour trouver l'ordinateur qui possède l'adresse IP qu'on a adressé. L'ordinateur qui

possède cette adresse se désigne et envoie une réponse à l'expéditeur pour lui donner son adresse MAC.

A la fin de cette requête, si toute fois il y a bien une réponse, l'expéditeur mémorise l'adresse de l'ordinateur en question dans un cache pour éviter de refaire une requête ARP.

Le Protocole DHCP

Le protocole DHCP est utilisé entre un serveur et ses clients, le client envoie une requête dhcp pour quelle lui donne une adresse IP qui lui permettra d'accéder au réseau. Pour ce faire le protocole se divise en plusieurs étapes. D'abord le client envoie un discovery à l'adresse 0 (ou à l'adresse du serveur s'il est connu) pour trouver un serveur DHCP. Pourquoi 0 ? Je ne sais pas mais on s'en fout. Le serveur lui répond en lui proposant une adresse IP avec un offer, et le client fait un request pour lui demander d'allouer cette adresse. Enfin le serveur peut lui envoyer un ack (ah c'est ok) pour lui donner l'adresse IP (avec toute les conditions) ou un nack (non(ah c'est ok)) pour l'envoyer voir ailleurs.

Mais assez de blabla. Du coté serveur, si on veut démarrer un serveur dhcp, il faut faire appel aux services DHCP.

Par défauts, ce service s'appelle **dhcp** ou **dhcp3**. On en fait appel avec la commande :

service dhcp3-server start ou dhcp-server start. Et on peut les redémarrer tout simplement en replaçant **start** par **restart** et les arrêter avec stop.

Mais dans le TP5, ils ont installé un service qui s'appelle isc-dhcp-server. Rien de bien méchant, il suffit de remplacer dhcp-server par isc-dhcp-server au moment de l'appel.

Pour configurer un serveur, il faut modifier le fichier suivant : **/etc/dhcp/dhcpd.conf** ou celui-ci **/etc/dhcp3/dhcpd.conf**

Exemple de configuration de ce fichier :

Option domain-name « on_s'emmerde.fr » ;

Option domain-name-servers « steeve-serveur.fr » ;

Subnet 192.168.0.0 netmask 255.255.255 :

```
{
        range 192.168.0.20  192.168.0.50 ;
        option routeurs 192.168.0.2 ;
        option broadcast-adress 192.168.0.255 ;
        option domain-name-serveurs 62.4.16.70 62.4.17.109 ;
        default-lease-time 86400 ;
        max-lease-time 604800 ;

        host machine2 {
                hardware ethernet 02 :04 :06 :8d :5d :fe ;
                fixed-adress 192.168.0.200 ;
                default-lease-time-604800 ;
        }
}
```

Ici, le service DHCP est configuré uniquement pour le serveur 192.168.0.0, mais on peut configurer plusieurs réseaux spécifiquement, il suffit de rajouter des blocs **Subnet** *adresse du réseau* **netmask** *masque du réseau*.

Pour cette configuration-là, le serveur peut donner des adresses allant de 192.168.0.20 192.168.0.50, cela correspond au **range**.

Les clients reçoivent en plus de l'adresse, des informations sur la route par défaut, **option routeur**, l'adresse de diffusion**, option broadcast-adress,** les adresses du serveur, et la durée du bail en seconde(la durée durant laquelle vous possédez l'adresse IP), **default-lease-time et max-lease-time**.

Vous pouvez aussi configurer des machines spécifiques, ici par exemple l'ordinateur qui possède l'adresse MAC 02 :04 :06 :8d :5d :fe se verra confier l'adresse 192.168.0.200 pendant 604800 secondes.

Si vous définissez une adresse unique pour un ordinateur spécifique, il faut qu'elle soit différente de celles définis par défauts.

Du côté client, pour faire une demande au serveur il faut faire la commande :

dhclient eth0

Et pour que la demande soit faite automatiquement au moment de l'allumage, il faut modifier le fichier **/etc/network/interface** :

auto eth0

iface eth0 inet dhcp

Protocole UDP/TCP

Les protocoles UDP et TCP sont des protocoles qui permettent d'envoyer des données. Ces deux protocoles contrôlent aussi le transport des données, il s'agit d'une autre couche, en plus des 3 autres vu en haut. Pour l'UDP, ce contrôle consiste à vérifier essentiellement si les paquets arrivent dans l'ordre, il est utilisé pour tous ce qui est streaming. Alors que le TCP fais plus de contrôle, il vérifie notamment s'il y a des erreurs dans la transmissions des données et si certains paquets se perdent, il arrête la transmission. Il est donc plus lent que l'UDP.

Pour mettre en place une connexion UPD ou TCP, on doit utiliser la commande nc (netcat).

Celui qui reçoit l'information doit faire la commande **nc –l** *numéro du port*.

Par défauts cela ouvre un port TCP avec le numéro que vous avez choisi, pour ouvrir un port UDP il faut rajouter l'option –**u**. On dit à ce moment-là que le port est à l'écoute.

Et de l'autre côté, si vous voulez envoyer des données à cette machine vous devez utiliser **nc** *ip-écouteur numéro du port* **<** *donnée*.

Maintenant si vous voulez envoyer un fichier et que de l'autre côté vous voulez l'enregistrer, vous devez redirigez le résultat de la commande dans un fichier.

Exemple

Coté écouteur : **nc –l 5000> merci.txt**

Coté expéditeur : **nc 168.212.0.1 5000 < merci.txt**

Vous pouvez vérifier si un port est ouvert avec la commande **lsof –i –P**

lsof sert à la base à trouver tous les fichiers ouverts, l'option –i permet de sélectionner les fichiers en rapport avec le réseau.

Protocole ICMPv4

Comme je vous l'ai dit tout à l'heure, lorsqu'une machine envoie des donnée à un autre, les données se balade de machine en machine pour arriver jusqu'au destinataire.

Les protocoles énoncés précédemment ne nous permet pas de les identifiés. Le protocole ICMPv4 permet d'identifier des informations liées à la transmission d'un message, par exemple le temps d'envoie et les destinataires.

Les paquets ICMP peuvent aussi renvoyer des erreurs dans le cas où le paquet n'atteint pas sa destination.

Je vous invite donc à voir le tableau des erreurs possible à la page 29 de votre excellent poly.

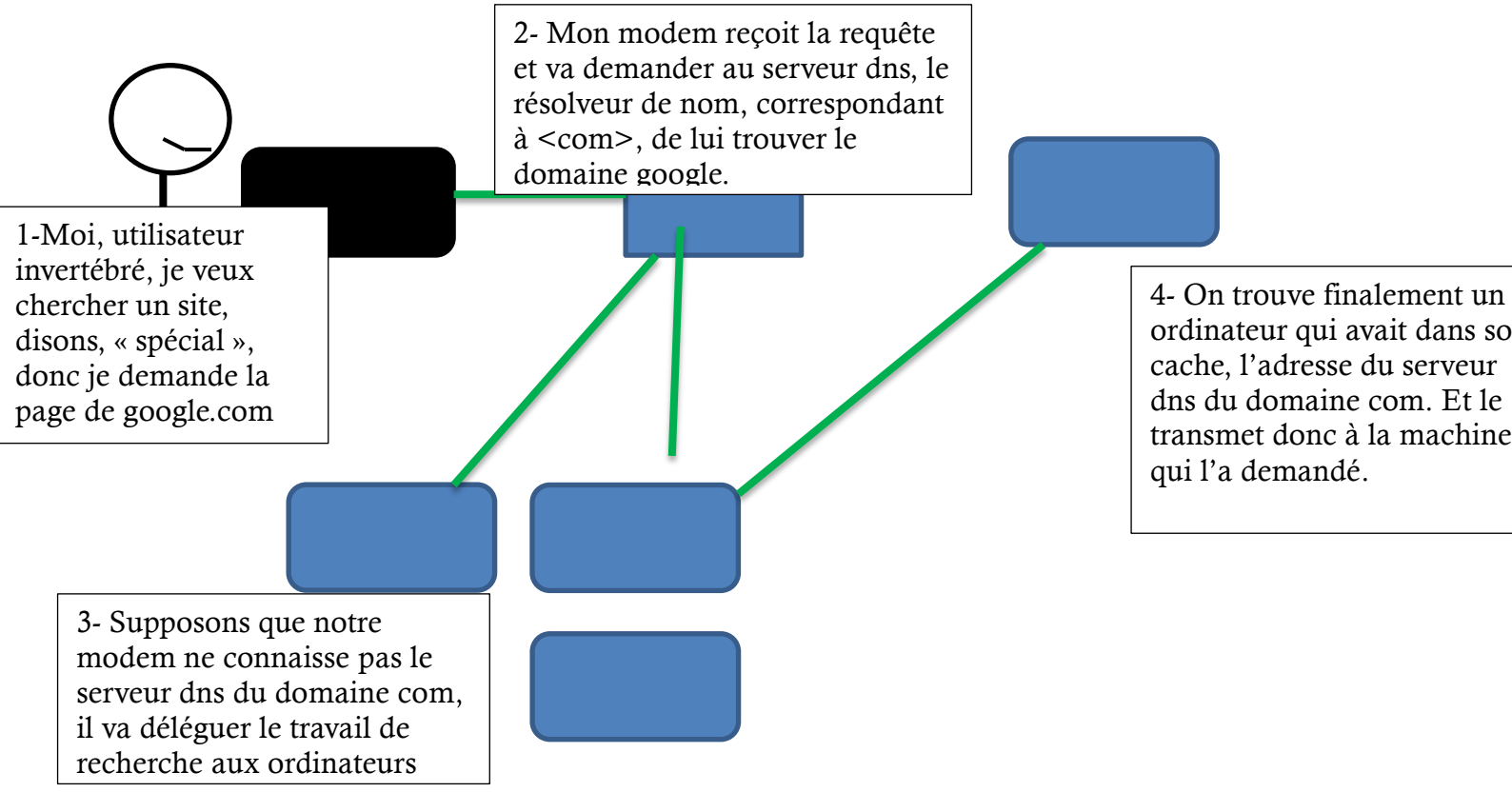
La commande **ping** utilise l'ICMP.

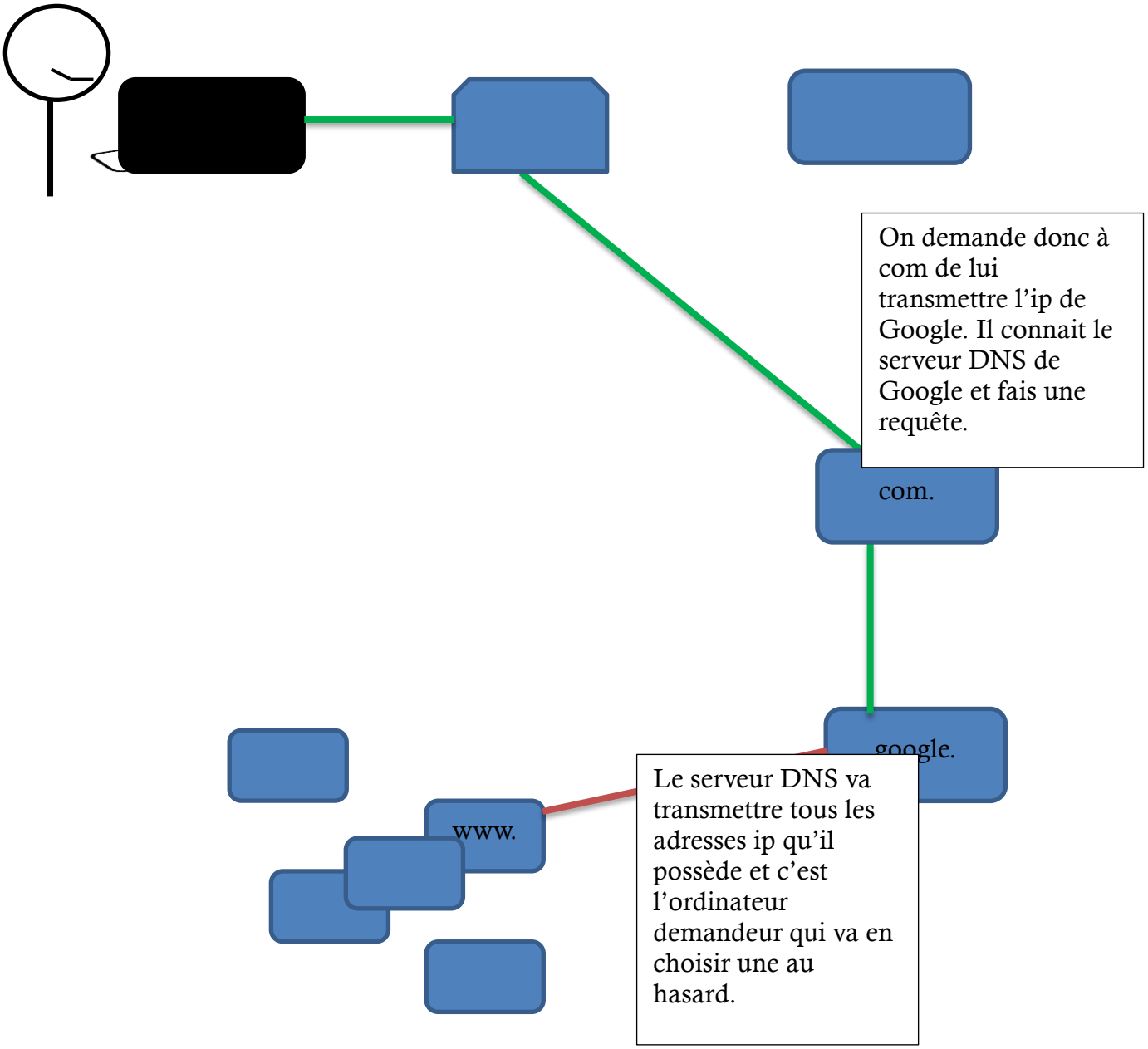
La commande **traceroute** est une commande qui ping toutes les machines qui retransmettent les données envoyées.

Protocole DNS

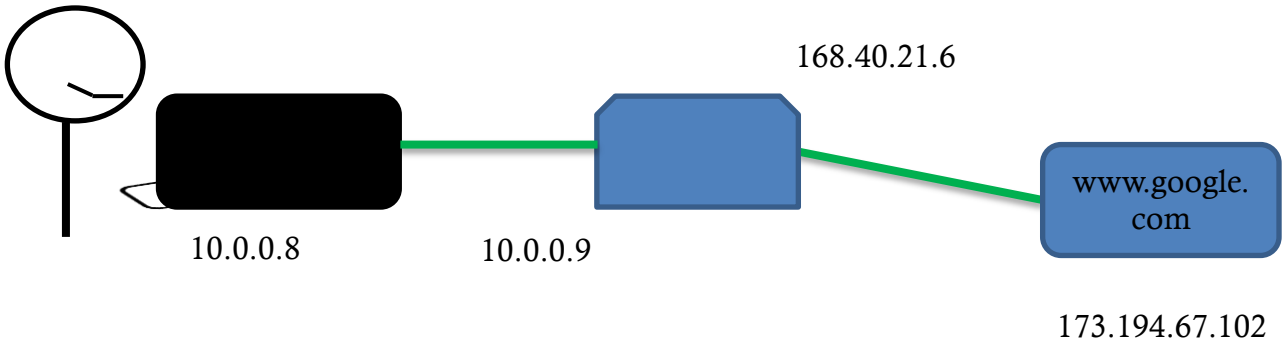
A quoi sert le protocole DNS ? Il s'agit d'un convertisseur de nom en adresse IP. Il s'agit surtout de récupérer l'adresse IP d'un ordinateur dont vous donner l'adresse sur internet.

Pour la question, « puis-je récupérer l'adresse MAC de google.com », la réponse est « non, casse toi ». Parce que l'adresse MAC des serveurs de Google n'est valables que dans le réseau dans lequel ils se trouvent. Pour atteindre les serveurs de Google, il faut trouver l'adresse IP du réseau dans lequel se trouve la cible à atteindre. Et c'est là qu'interviennent les domaines. Petit exemple :

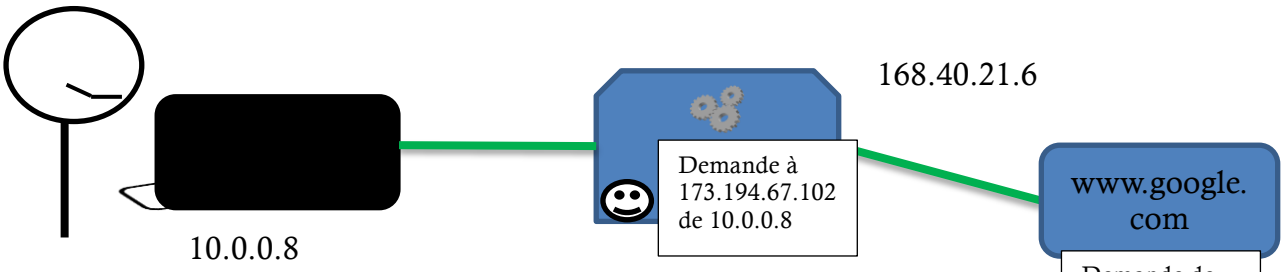




Petite parenthèse, l'adressage IPv4 permet l'accès à plus de 4 milliards de réseaux et pas de machines. Le problème d'aujourd'hui c'est que plusieurs ordinateurs peuvent avoir la même adresse IP, ce problème est gérer par le principe de translation d'adresse (NAT). Reprenons notre exemple :



Je souhaite avoir la page de Google, Google va donc m'envoyer la page html correspondant, cependant il doit connaitre l'adresse IP, du demandeur. Mon adresse qui est 10.0.0.8 est sans aucun doute utilisée par un autre utilisateur.



Le modem va enregistrer la demande avant de faire la demande, il va transmettre alors à Google non pas l'adresse 10.0.0.8 mais 168.40.21.6 (adresse unique), et quand il va recevoir un paquet de google, il saura que celui qui a fait une demande à google c'est 10.0.0.8, donc il retransmet le paquet. C'est le principe de NAT ou translation d'adresse.

Recherche DNS :

Lorsqu'un ordinateur veut faire une recherche DNS, il va d'abord aller dans le fichier **/etc/nsswitch** pour suivre les différentes instructions définis pour une résolution de nom. Ce fichier va d'abord le rediriger vers le fichier **/etc/hosts**. Ce fichier host possède tous les noms de domaines dont on connait l'adresse.

Si le nom de domaine dont on veut l'adresse ne se trouve pas dans le fichier hosts alors d'après le fichier **nsswitch**, l'ordinateur va regarder le fichier **/ect/resolv.conf** pour y trouver d'autres instructions pour la résolution de nom.

Exemple de configuration hosts :

127.0.0.1 localhost moi

[configuration pour IPv6 prédéfinis]

192.168.19.21 thefiregreenhost

Si vous pinguez « moi » vous vous pinguerez vous-même

Si vous pinguez « thefiregreenhost » vous pinguerez la machine ayant l'adresse 192.168.19.21

Exemple de configuration resolv.conf :

domaine thefiregreen.fr

search thefiregreen.fr beaujeu.thefiregreen.fr

nameserver 172.16.35.1

Cette ligne spécifie que lors d'une recherche de nom vous rechercherez par défaut ce nom dans le domaine thefiregreen.fr

Cette ligne spécifie que lors d'une recherche dans le domaine thefiregreen.fr vous rechercherez ce nom dans le sous-domaine beaujeu.thefiregreen.fr

Cette ligne indique quelle machine va pouvoir résoudre votre nom

Pour effectuer une recherche DNS, vous pouvez utiliser la commande **dig** ou la commande **host**.

dig @192.168.0.2 google.com -> cherche l'adresse de google.com on spécifiant que la demande de résolution est faite à la machine d'adresse 192.168.0.2.

dig google.com -> cherche l'adresse de google.com en suivant les instructions définis par nsswitch.

dig google.com AAAA -> recherche l'adresse IPv6 de google.com

Les requêtes DNS renvoie plusieurs champs dans lesquels se trouve des informations spécifiques au résultat de la recherche, ici je lui ai demandé de me renvoyer le champ AAAA

qui correspond à l'adresse IPv6, pour connaitre tous les champs, veuillez-vous référer au poly du cher Mr.Motchane, p43.

dig 101.10.55.193.firegreen.fr PTR -> recherche le nom de domaine associé à l'adresse 101.10.55.193 du domaine firegreen.fr