10

Protocoles de communication dans un réseau

Extrait du programme

Thème : Architectures matérielles et systèmes d'exploitation

- **Contenus :**
  Transmission de données dans un réseau
  Protocoles de communication
  Architecture d'un réseau

Capacités attendus :
Mettre en évidence l'intérêt du découpage des données en paquets et de leur encapsulation.
Dérouler le fonctionnement d'un protocole simple de récupération de perte de paquets (bit alterné).
Simuler ou mettre en œuvre un réseau.

Commentaires :
Le protocole peut être expliqué et simulé en mode débranché.
Le lien est fait avec ce qui a été vu en classe de seconde sur le protocole TCP/IP.
Le rôle des différents constituants du réseau local de l'établissement est présenté.

Ce cours est fortement inspiré du travail de G.Lassus, Lycée François Mauriac – Bordeaux.

Pour commencer

Et si nous faisions une petite lecture de l'article suivant :
`https://members.loria.fr/MDuflot/reseaux-info-a-ma-fille-intro/`...

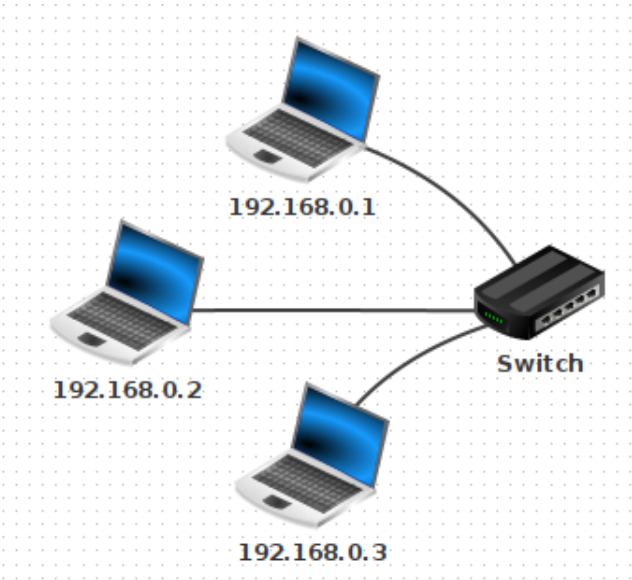
Ensuite nous pourrions télécharger et installer le logiciel **Filius** que nous allons utiliser tout au long de ce cours. Il s'agit d'un simulateur de réseau :
`https://www.lernsoftware-filius.de/Herunterladen`

I. Architecture d'un réseau

Ce cours a pour but de présenter la constitution classique d'un réseau, et les équipements associés. La partie relative aux protocoles utilisés lors des échanges entre deux machines sera détaillée un peu plus loin le cours.

a) Premier réseau local

Avec le logiciel Filius, créons le réseau local ci-dessous :



Testons le `ping` de la machine `192.168.0.1` vers la machine `192.168.0.3`.
Résultat :

```
root /> ping 192.168.0.3
PING 192.168.0.3 (192.168.0.3)
From 192.168.0.3 (192.168.0.3): icmp_seq=1 ttl=64 time=413ms
From 192.168.0.3 (192.168.0.3): icmp_seq=2 ttl=64 time=204ms
From 192.168.0.3 (192.168.0.3): icmp_seq=3 ttl=64 time=205ms
From 192.168.0.3 (192.168.0.3): icmp_seq=4 ttl=64 time=203ms
--- 192.168.0.3 Statistiques des paquets ---
4 paquets transmis, 4 paquets reçus, 0% paquets perdus
```

La carte réseau et son adresse MAC

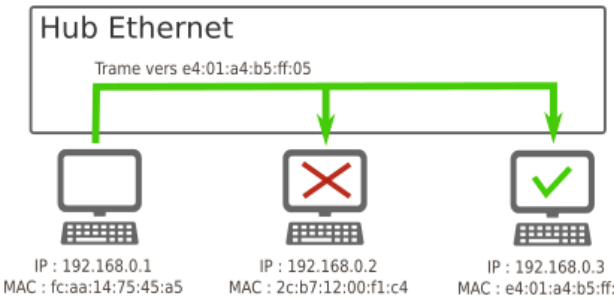
Chaque ordinateur sur le réseau dispose d'une adresse **MAC**, qui une valeur unique attribuée à sa carte réseau (Ethernet, Wifi, 4G, 5G, ...) lors de sa fabrication en usine. Cette adresse est codée sur 48 bits, présentés sous la forme de 6 octets en hexadécimal.

Exemple : `fc:aa:14:75:45:a5`

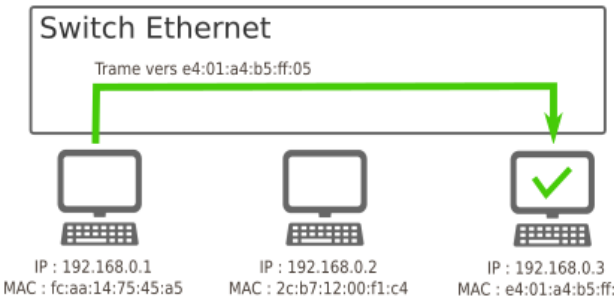
Les trois premiers octets correspondent au code du fabricant. Un site comme `https://www.macvendorlookup.com/` vous permet de retrouver le fabricant d'une adresse MAC quelconque.

Switch, hub, quelle différence

- Au sein d'un **hub Ethernet** (de moins en moins vendus), il n'y a **aucune analyse** des données qui transitent : il s'agit simplement d'un dédoublement des fils de cuivre (tout comme une multiprise électrique). L'intégralité des messages est donc envoyée à l'intégralité des ordinateurs du réseau, même s'ils ne sont pas concernés.



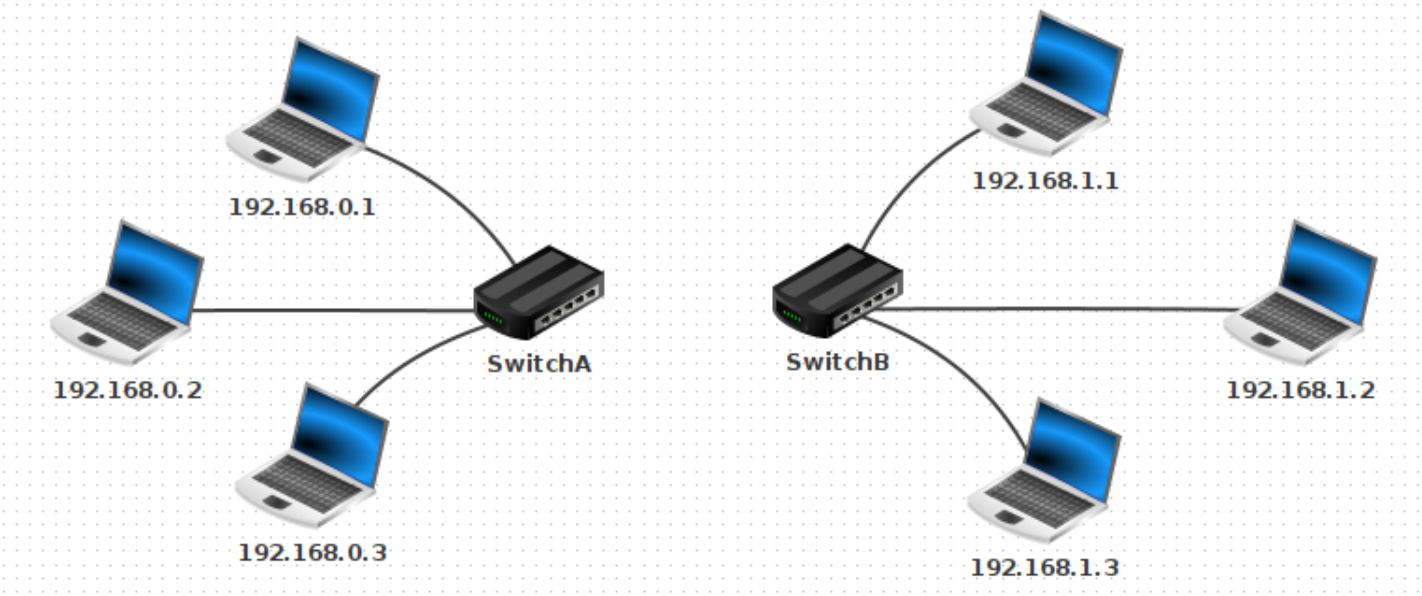
- Au sein d'un **switch Ethernet** , une analyse est effectuée sur la trame qui est à distribuer. Lors d'un branchement d'un nouvel ordinateur sur le switch, celui-ci récupère son adresse MAC, ce qui lui permet de **trier** les messages et de ne les distribuer qu'au bon destinataire.



b) Un deuxième sous-réseau

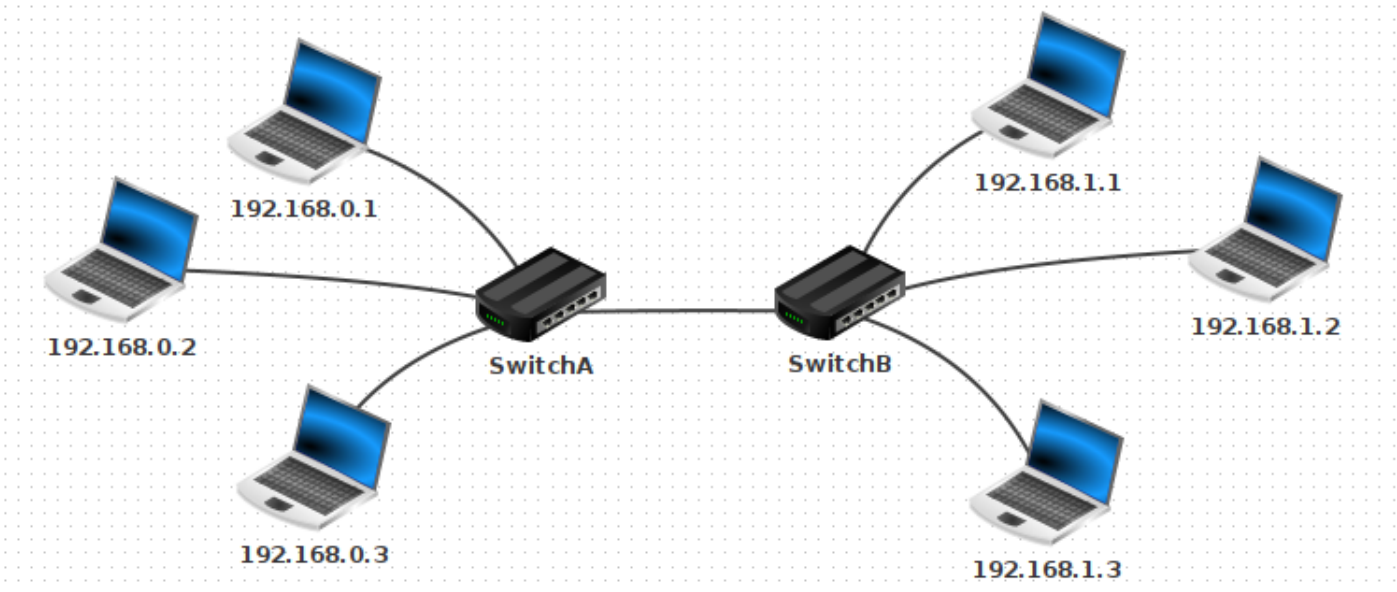
Comment relier deux sous-réseaux ?

Rajoutons un deuxième sous-réseau de la manière suivante (penser à bien renommer les switchs).



Comment allons-nous relier ces deux sous-réseaux ?

Une réponse pas si bête : avec un câble entre les deux switchs !



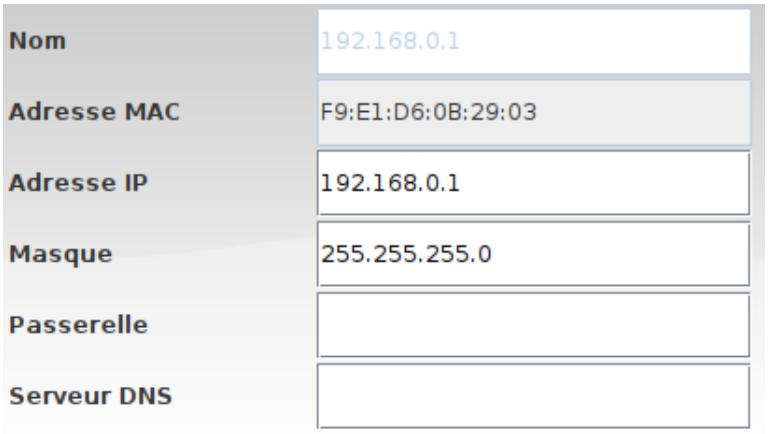
Testons cette hypothèse en essayant de pinger la machine `192.168.1.2` depuis la machine `192.168.0.1`. Résultat :

```
root /> ping 192.168.1.2
Destination inaccessible
root />
```

Cela ne marche pas. Les paquets sont perdus.
Intuition : la notion de sous-réseau n'est pas topologique, autrement dit, il ne suffit de relier les ordinateurs entre eux pour qu'ils communiquent. Un sous-réseau obéit à des règles numériques.

c) Notion de masque de sous-réseau

Dans Filius, lors de l'attribution de l'adresse IP à une machine, une ligne nous permet de spécifier le **masque de sous-réseau** (appelé simplement « Masque » dans Filius). C'est ce masque qui va permettre de déterminer si une machine appartient à un sous-réseau ou non, en fonction de son adresse IP.



Explication basique

- Si le masque est `255.255.255.0`, toutes les machines partageant les mêmes **trois** premiers nombres de leur adresse IP appartiendront au même sous-réseau. Comme ceci est le réglage par défaut de Filius, cela explique pourquoi `192.168.0.3` et `192.168.0.1` sont sur le même sous-réseau, et pourquoi `192.168.1.2` et `192.168.0.1` ne sont pas sur le même sous-réseau.
  Dans cette configuration, 256 machines peuvent donc appartenir au même sous-réseau (ce n'est pas tout à fait le cas car des adresses finissant par 0 ou par 255 sont réservées).
- Si le masque est `255.255.0.0`, toutes les machines partageant les mêmes **deux** premiers nombres de leur adresse IP appartiendront au même sous-réseau.
  Dans cette configuration, 65 536 machines peuvent être dans le même sous-réseau. (car $256^2 = 65\,536$)

Exercice 1

- Modifier aussi le masque de `192.168.0.1` en `255.255.0.0`.
- Tester le ping de `192.168.0.1` vers `192.168.1.2`. Le ping ne fonctionne toujours pas. Pourquoi ? Que faut-il corriger ?

Il faut aussi préciser à la machine `192.168.1.2` à un masque `255.255.0.0`. Sinon, `192.168.1.2` reçoit bien un message de `192.168.0.1` mais est incapable de répondre[1].

```
root /> ping 192.168.1.2
PING 192.168.1.2 (192.168.1.2)
From 192.168.1.2 (192.168.1.2): icmp_seq=1 ttl=64 time=306ms
From 192.168.1.2 (192.168.1.2): icmp_seq=2 ttl=64 time=304ms
From 192.168.1.2 (192.168.1.2): icmp_seq=3 ttl=64 time=305ms
From 192.168.1.2 (192.168.1.2): icmp_seq=4 ttl=64 time=306ms
--- 192.168.1.2 Statistiques des paquets ---
4 paquets transmis, 4 paquets reçus, 0% paquets perdus
```

Cela marche. Les deux machines appartiennent maintenant au même sous-réseau.

1. Penser que le nom de la commande `ping` vient du jeu Ping Pong. Lorsque l'on fait un `ping` on attend une réponse de l'autre machine, le `pong`.

Explication avancée

Lorsqu'une machine A veut envoyer un message à une machine B, elle doit déterminer si cette machine :

- appartient au même sous-réseau : auquel cas le message est envoyé directement via un ou plusieurs switchs ;
- n'appartient pas au même sous-réseau : auquel cas le message doit d'abord transiter par un routeur.

Quelle opération permet de distinguer cette appartenance à un même sous-réseau ?

Propriété : Appelons `IP_A` et `IP_B` les adresses IP respectives des machines A et B.
Appelons `M` le masque de sous-réseau.
Nommons `&` l'opérateur **ET** binaire.
A et B appartiennent au même sous-réseau $\iff$ `IP_A & M` = `IP_B & M`

Exemple
Considérons trois machines A, B, C d'IP respectives `192.168.129.10`, `192.168.135.200` et `192.168.145.1`, configurées avec un masque de sous-réseau égal à `255.255.248.0`.

	machine A	machine B	machine C
IP	192.168.129.10	192.168.135.200	192.168.145.1
M	255.255.248.0	255.255.248.0	255.255.248.0
IP & M	192.168.128.0	192.168.128.0	192.168.144.0

Ce tableau a été rempli avec les règles de calcul pour chaque bit suivantes : 1&1 =1, 1&0=0, 0&1=0, 0&0=0.
On a alors, `168 & 255` qui s'écrit en binaire `10101000 & 11111111` qui vaut bit par bit `10101000` soit `168`.
De même `129 & 248` qui s'écrit en binaire `10000001 & 11111000` qui vaut bit par bit `10000000` soit `128`.
Conclusion : les machines A et B sont sous le même sous-réseau, mais pas la machine C.

Cohérence entre les deux explications

Lorsqu'un masque de sous-réseau est égal à `255.255.255.0`, l'opération `&` avec chaque IP ne laissera intacts que les 3 premiers octets, le dernier sera égal à 0. Donc si deux adresses s'écrivent `A.B.C.X` et `A.B.C.Y`, elles appartiendront forcément au même sous-réseau (typiquement, c'est le cas de `192.168.0.3` et `192.168.0.1`).

d) Écriture des masques de sous-réseau : notation CIDR

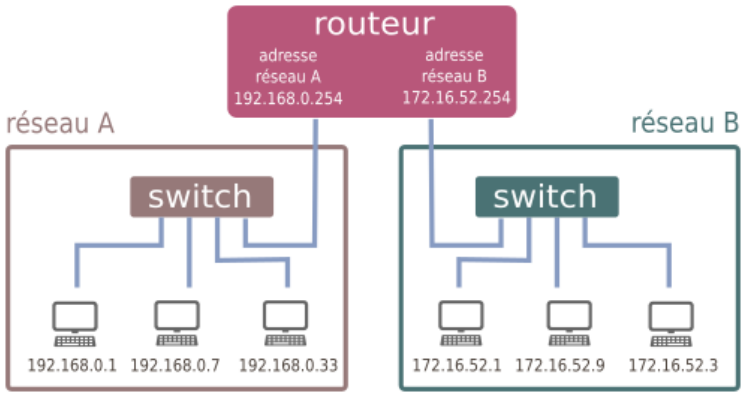
D'après ce qui précède, 2 informations sont nécessaires pour déterminer le sous-réseau auquel appartient une machine : son IP et le masque de sous-réseau. Une convention de notation permet d'écrire simplement ces deux renseignements : la notation CIDR.

Exemple
Une machine d'IP `192.168.0.3` avec un masque de sous-réseau `255.255.255.0` sera désignée par `192.168.0.33 / 24` en notation **CIDR** (pour *Classless Inter-Domain Routing*).

Le suffixe `/ 24` signifie que le masque de sous-réseau commence par 24 bits consécutifs de valeur 1 : le reste des bits (donc 8 bits) est à mis à 0. Autrement dit, ce masque vaut `11111111.11111111.11111111.00000000`, soit `255.255.255.0`.
De la même manière, le suffixe `/ 16` donnera un masque de `11111111.11111111.00000000.00000000` , soit `255.255.0.0`.
Ou encore, un suffixe `/ 21` donnera un masque de `11111111.11111111.11111000.00000000` , soit `255.255.248.0`.

e) Un vrai réseau contenant deux sous-réseaux distincts : la nécessité d'un routeur

Notre solution initiale (relier les deux switchs par un câble pour unifier les deux sous-réseaux) n'est pas viable à l'échelle d'un réseau planétaire. Pour que les machines de deux réseaux différents puissent être connectées, on va utiliser un dispositif équipé de **deux cartes réseaux**, situé à cheval entre les deux sous-réseaux. Ce équipement de réseau est appelé **routeur** ou **passerelle**.



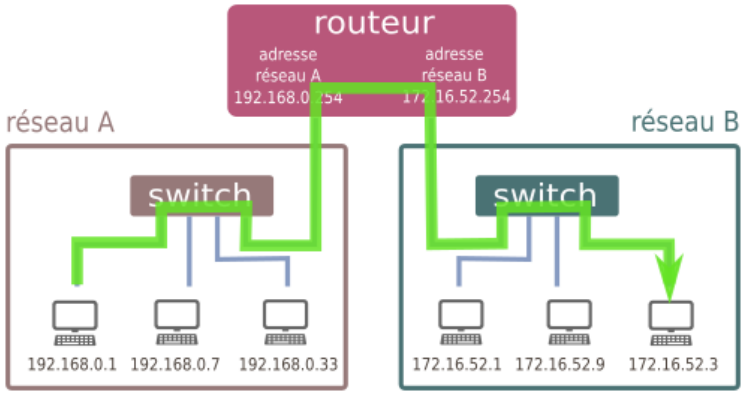
Principe de fonctionnement

Imaginons que la machine `192.168.0.1 / 24` veuille communiquer avec la machine `172.16.52.3 / 24`. L'observation du masque de sous-réseau de la machine `192.168.0.1 / 24` nous apprend qu'elle ne peut communiquer qu'avec les adresses de la forme `192.168.0.X / 24`, où `X` est un nombre entre 0 et 255.

Les 3 étapes du **routage** :

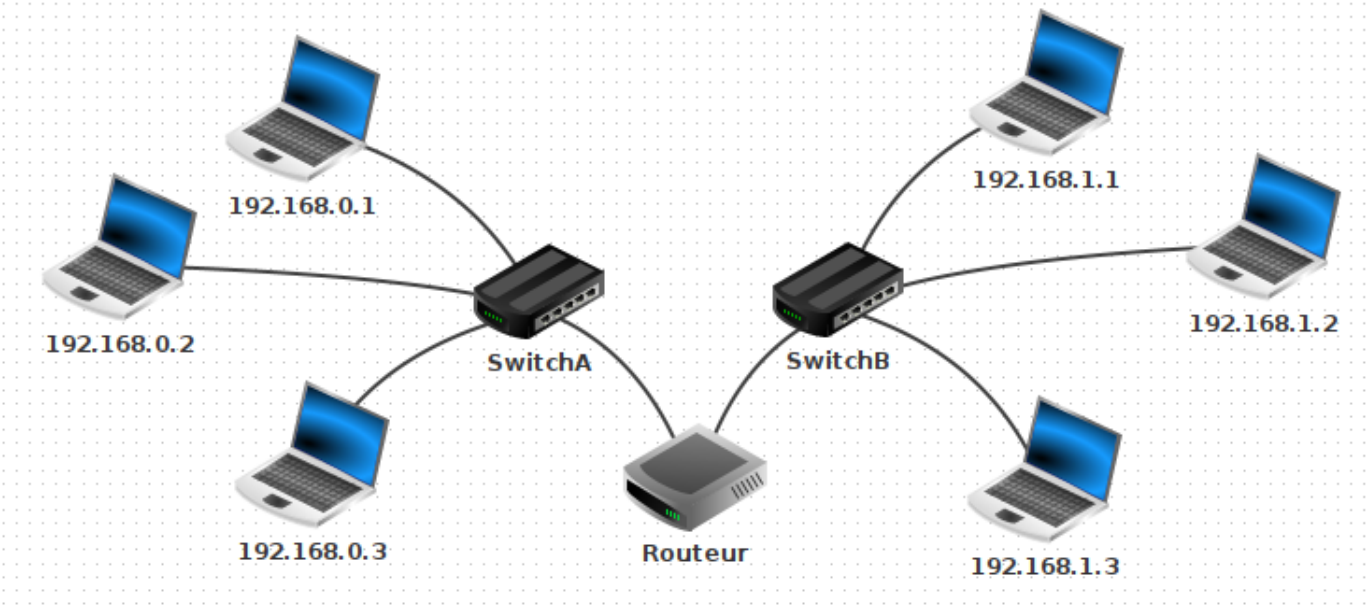
- Lorsque qu'une machine A veut envoyer un message à une machine B, elle va tout d'abord vérifier si cette machine appartient à son réseau local. si c'est le cas, le message est envoyé par l'intermédiaire du switch qui relie les deux machines.
- Si la machine B n'est pas trouvée sur le réseau local de la machine A, le message va être acheminé vers le routeur, par l'intermédiaire de son adresse de passerelle (qui est bien une adresse appartenant au sous-réseau de A).
- De là, le routeur va regarder si la machine B appartient au deuxième sous-réseau auquel il est connecté. Si c'est le cas, le message est distribué, sinon, le routeur va donner le message à un autre routeur auquel il est connecté et va le charger de distribuer ce message : c'est le procédé (complexe) de routage [2], qui sera vu en classe de Terminale.

Dans notre exemple, l'adresse `172.16.52.3` n'est pas dans le sous-réseau de `192.168.0.1`. Le message va donc transiter par le routeur.

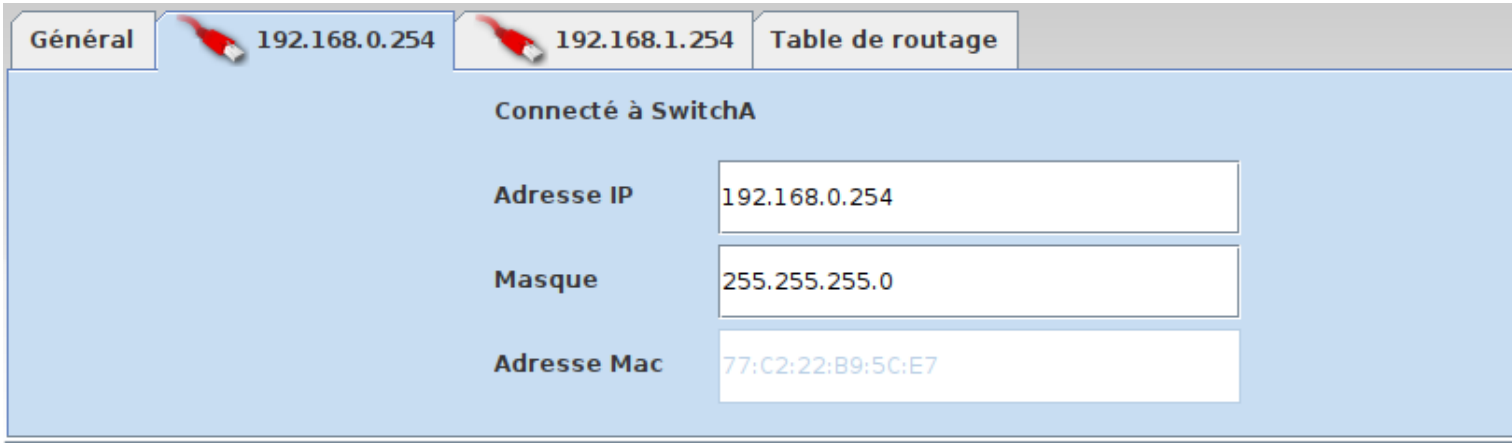


2. Pour en savoir plus, vous pouvez consulter le site openclassrooms : `https://openclassrooms.com/fr/courses/857447-apprenez-le-fonctionnement-des-reseaux-tcp-ip/854659-le-routage`.

Illustration avec Filius



- Rajoutons un routeur entre le Switch A et le Switch B.
- Configuration du routeur :
  L'interface reliée au Switch A doit avoir une adresse du sous-réseau A. On donne souvent une adresse finissant par `254`, qui est en quelque sorte la dernière adresse du réseau (en effet l'adresse en `255` est appelée adresse de **broadcast**, utilisée pour pinger en une seule fois l'intégralité d'un sous-réseau).
  On donne donc l'adresse `192.168.0.254` pour l'interface reliée au Switch A, et `192.168.1.254` pour l'interface reliée au Switch B.



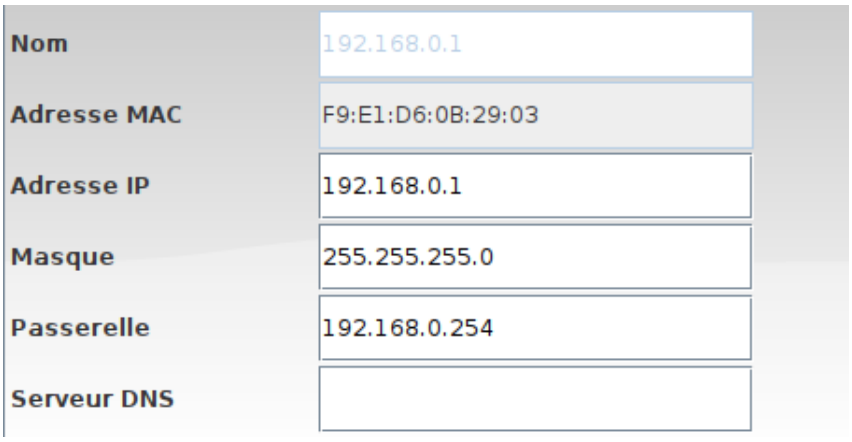
  Dans l'onglet général, sélectionner « Routage automatique ».
  Ainsi configuré notre routeur peut jouer le rôle de **passerelle** entre les deux sous-réseaux.

Test du ping entre `192.168.0.1` et `192.168.1.2` :
Résultat :

```
root /> ping 192.168.1.2
Destination inaccessible
root />
```

Cela ne marche pas. Les paquets sont toujours perdus.

Pourquoi cet échec ?
Parce que nous devons dire à chaque machine qu'une passerelle est maintenant disponible pour pouvoir sortir de son propre sous-réseau. Il faut donc aller sur la machine `192.168.0.1` et lui donner l'adresse de sa passerelle, qui est `192.168.0.254`.



Attention, il faut faire de même pour `192.168.1.2` (avec la bonne passerelle...)
Testons à nouveau le ping... Cette fois cela marche.

Plus intéressant : effectuons un `traceroute` entre `192.168.0.1` et `192.168.1.2`.

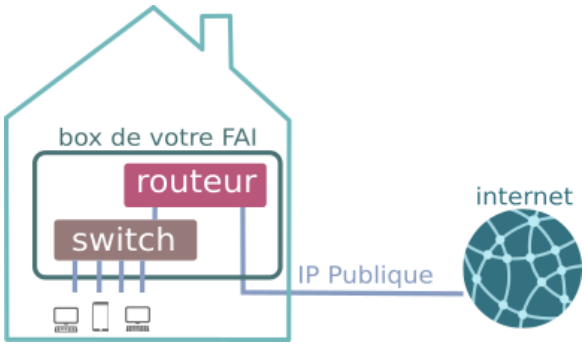
```
root /> traceroute 192.168.1.2
Établissement de la connexion avec 192.168.1.2 (en 20 sauts max.).
 1    192.168.0.254
 2    192.168.1.2

192.168.1.2 a été atteint en 2 sauts.
```

On y aperçoit que la machine `192.168.1.2` est atteignable en deux sauts depuis `192.168.0.1`, en passant par la passerelle `192.168.0.254`.

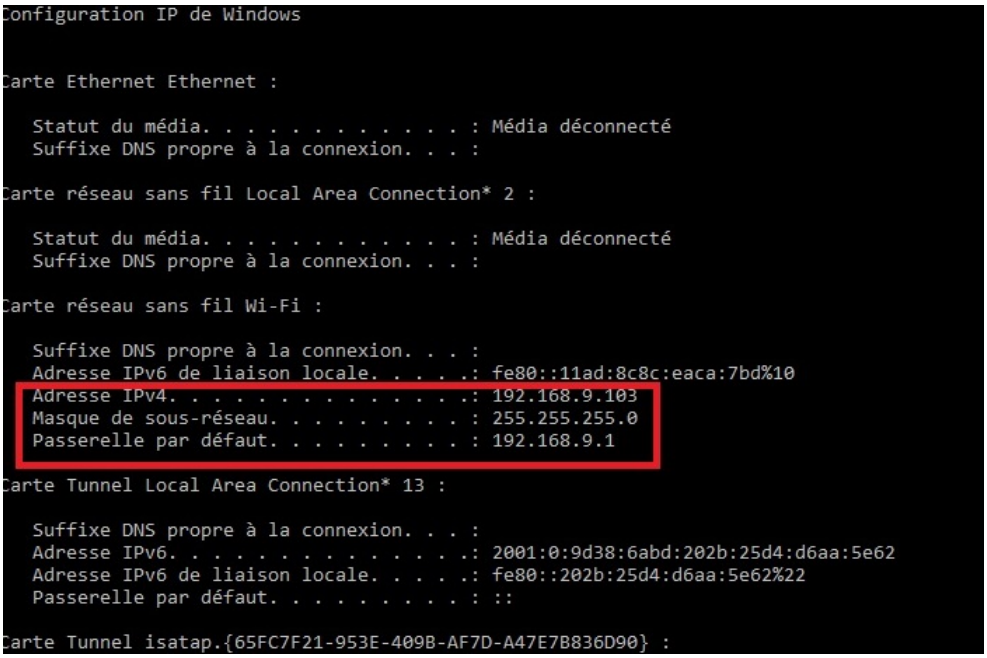
Cas d'un réseau domestique

Chez vous, la box de votre opérateur joue simultanément le rôle de switch et de routeur :

- switch car elle répartit la connexion entre les différents dispositifs (ordinateurs branchés en ethernet, smartphone en wifi, tv connectée...)
- routeur car elle fait le lien entre ce sous-réseau domestique (les appareils de votre maison) et le réseau internet.



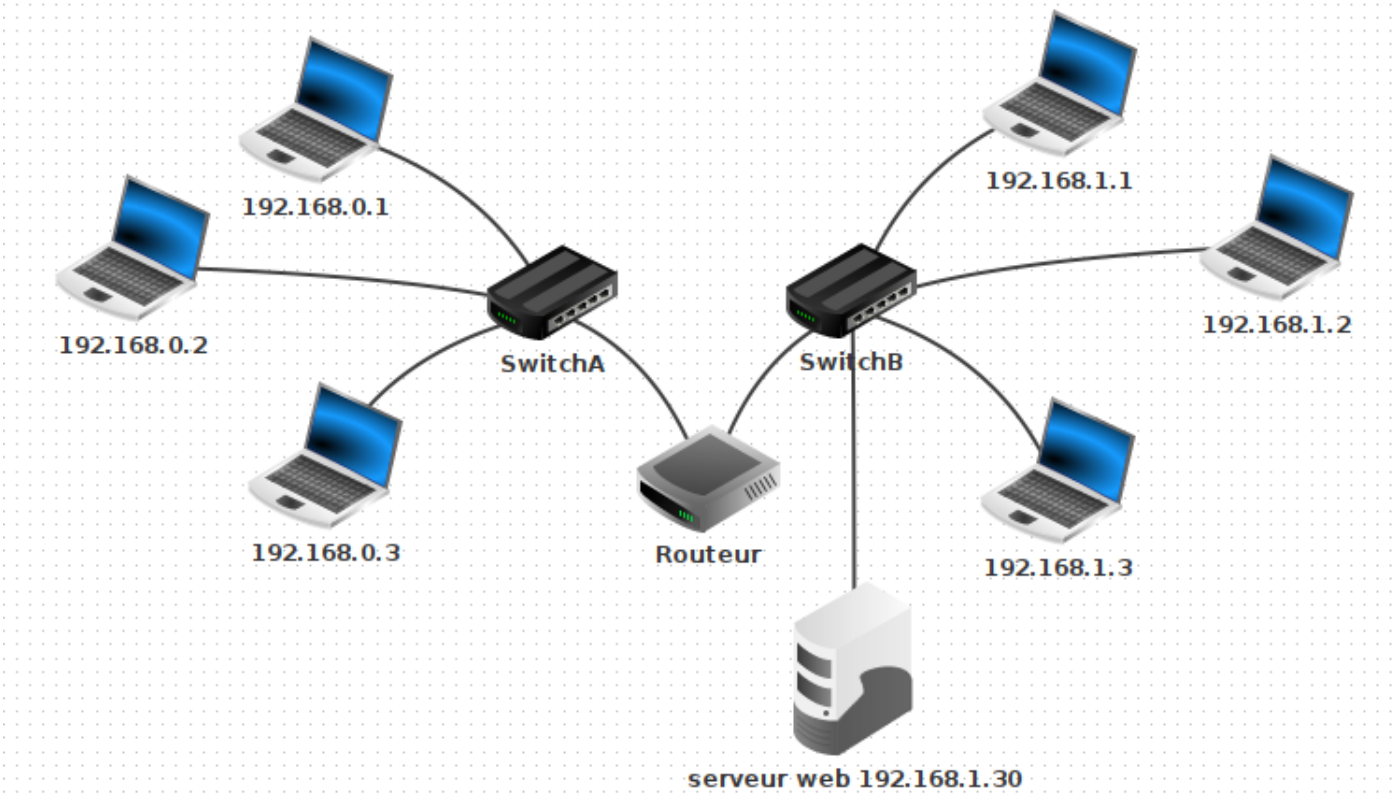
L'image ci-dessous présente le résultat de la commande `ipconfig` sous Windows. On y retrouve l'adresse IP locale `192.168.9.103` le masque de sous-réseau `255.255.255.0` et l'adresse de la passerelle `192.168.9.1`.


Configuration IP de Windows
Carte Ethernet Ethernet :
Statut du média. . . . . : Média déconnecté
Suffixe DNS propre à la connexion. . . :
Carte réseau sans fil Local Area Connection* 2 :
Statut du média. . . . . : Média déconnecté
Suffixe DNS propre à la connexion. . . :
Carte réseau sans fil Wi-Fi :
Suffixe DNS propre à la connexion. . . :
Adresse IPv6 de liaison locale. . . . .: fe80::11ad:8c8c:eaca:7bd%10
Adresse IPv4. . . . . : 192.168.9.103
Masque de sous-réseau. . . . . : 255.255.255.0
Passerelle par défaut. . . . . : 192.168.9.1
Carte Tunnel Local Area Connection* 13 :
Suffixe DNS propre à la connexion. . . :
Adresse IPv6. . . . . : 2001:0:9d38:6abd:202b:25d4:d6aa:5e62
Adresse IPv6 de liaison locale. . . . .: fe80::202b:25d4:d6aa:5e62%22
Passerelle par défaut. . . . . : ::
Carte Tunnel isatap.{65FC7F21-953E-409B-AF7D-A47E7B836D90} :

f) Annexe : rajout d'un serveur DNS

Rajout d'un serveur web

- Connectons un ordinateur au Switch B, sur l'adresse `192.168.1.30` et installons dessus un Serveur web et démarrons-le.



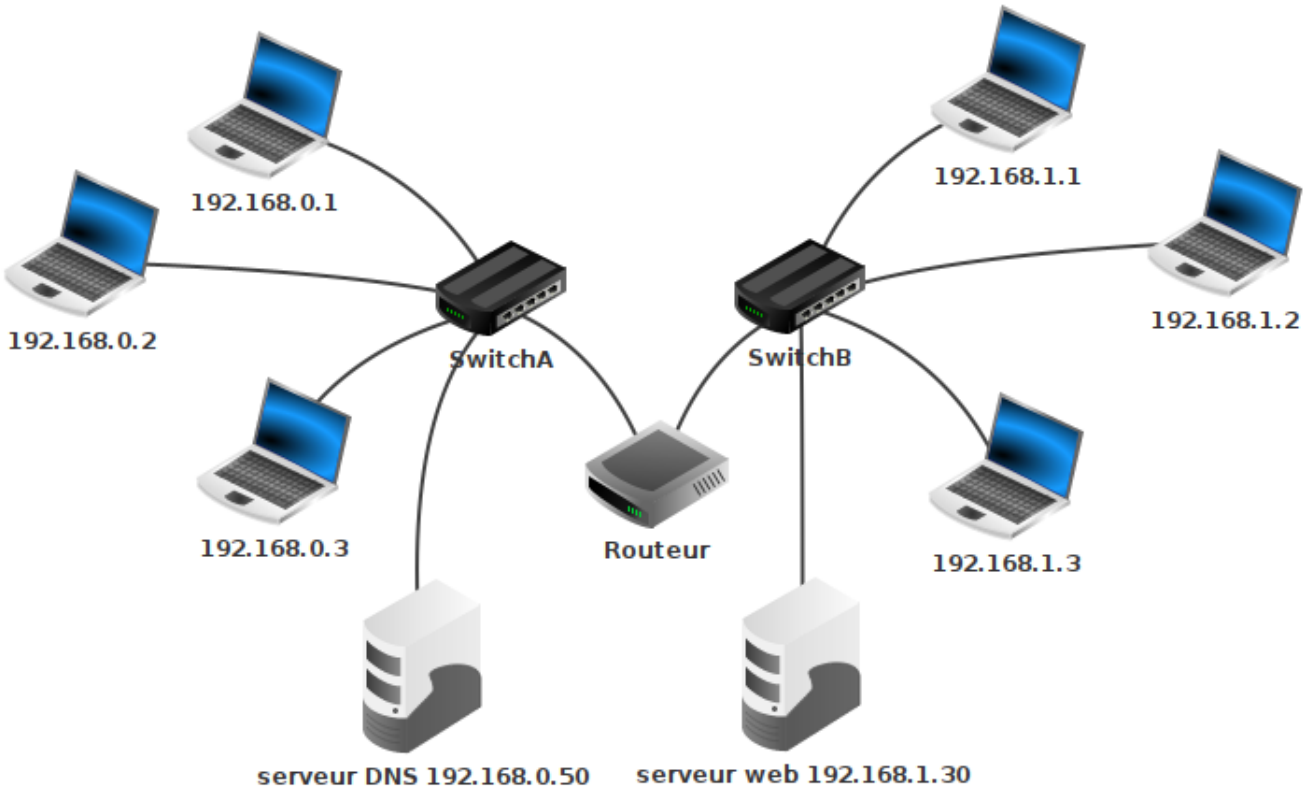
- Sur la machine `192.168.0.1`, rajoutons un Navigateur Web. En tapant dans la barre d'adresse l'adresse IP du Serveur web, la page d'accueil de Filius s'affiche.

Lors d'une utilisation classique d'un navigateur web, c'est une url mémorisable qui s'affiche, et non une adresse IP : on retient en effet plus facilement `https://www.google.com/` que `http://216.58.213.131`, qui renvoient pourtant à la même adresse. La machine qui assure ce rôle d'annuaire entre les serveurs web et leur adresse IP s'appelle un **serveur DNS** (DNS pour *Domain Name System*). Pour pouvoir indexer la totalité des sites internet, son rôle est structuré de manière hiérarchique [3].

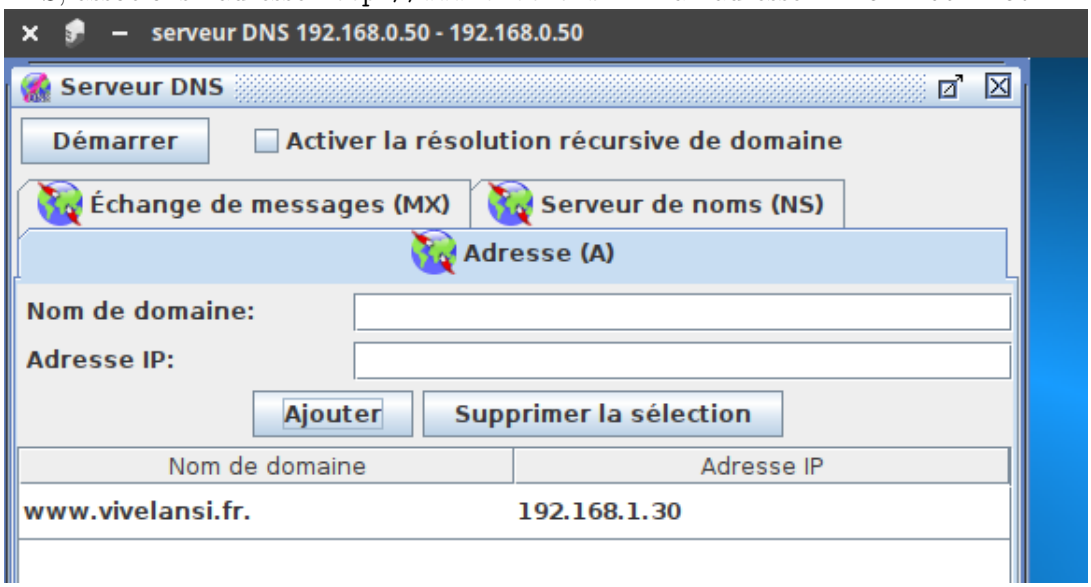
3. Vous trouverez des détails ici : `https://openclassrooms.com/fr/courses/857447-apprenez-le-fonctionnement-des-reseaux-tcp-ip/857163-le-service-dns`.

Rajout d'un serveur DNS

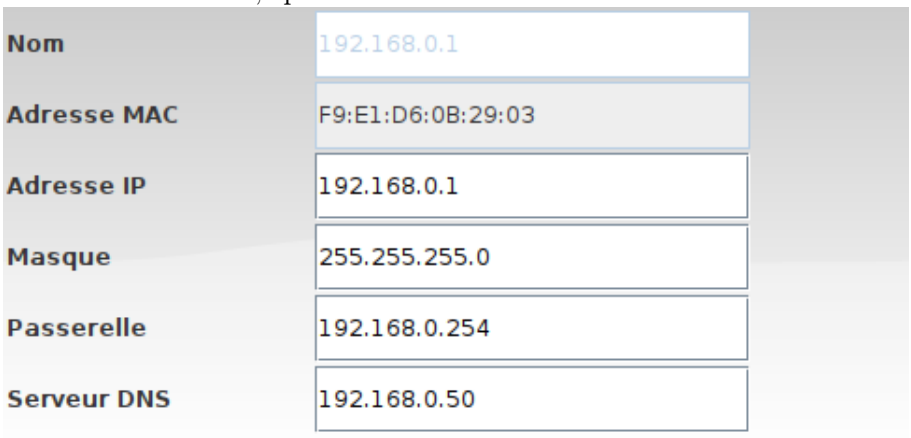
- Rajoutons un serveur DNS minimal, qui n'aura dans son annuaire d'un seul site. Il faut pour cela raccorder une nouvelle machine (mais une machine déjà sur le réseau aurait très bien pu jouer ce rôle), et installer dessus un serveur DNS.



- Sur ce serveur DNS, associons l'adresse `http://www.vivelansi.fr` à l'adresse IP `192.168.1.30`.



- De retour sur notre machine `192.168.0.1`, spécifions maintenant l'adresse du serveur DNS :



- Depuis le navigateur web de la machine `192.168.0.1`, le site `http://www.vivelansi.fr` est maintenant accessible.

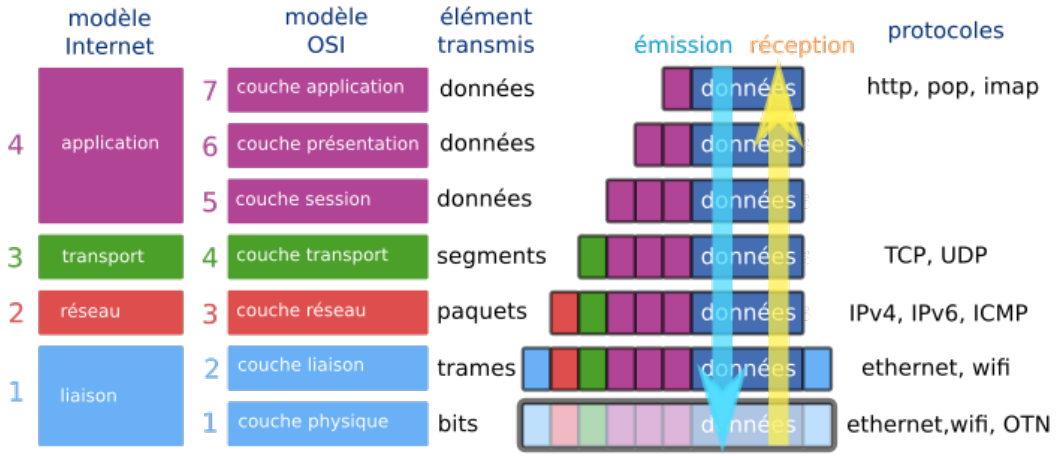
II. Protocoles de communication dans un réseau

a) Modèle OSI, modèle Internet

Les bits transmis d'un ordinateur à un autre contiennent, en plus des données utiles (le mot « bonjour » dans un courriel), une multitude de données (tout aussi utiles) qui vont aider à l'acheminement de ces bits au bon endroit, puis au bon ordinateur, puis au bon logiciel. Les différents protocoles qui régissent cette transmission sont regroupés dans ce qui est appelé un **modèle**. Deux modèles synthétisent ces protocoles :

- le **modèle Internet** (ou modèle **TCP/IP**, 1974), organisé en **4** couches : liaison, réseau, transport, application.
- le **modèle OSI** (*Open Systems Interconnection,* 1984), organisé en **7** couches : physique, liaison, réseau, transport, session, présentation,application.

Ces deux modèles coïncident suivant le schéma ci-dessus. Ce sont des modèles théoriques et d'une certaine rigidité. Leur utilisation dans la pratique est parfois plus floue, avec des protocoles à cheval sur plusieurs couches. Dans la suite de ce cours, nous évoquerons les couches par leur numéro dans le modèle OSI.



Lors de son émission, un message va subir successivement toutes les transformations effectuées par chaque couche, depuis sa création (couche 7) jusqu'à sa transmission physique (couche 1). Lorsque ce même message sera réceptionné, les transformations seront effectuées dans l'ordre inverse, jusqu'à la présentation du message au destinataire.

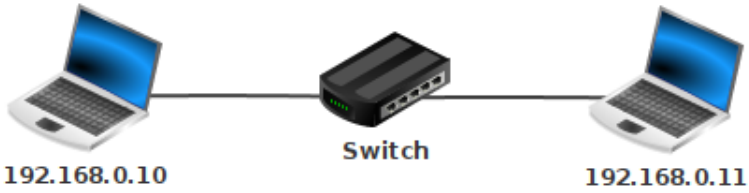
- **couches 7-6-5 — couches application-présentation-session :**
  Ces couches (réunies dans le modèle Internet en une couche unique « application ») regroupent les protocoles nécessaires à la bonne mise en forme d'un message (au sens large) avant sa transmission. Ces protocoles peuvent être de nature très différente : protocole HTTP pour la transmission de pages web, protocole FTP pour le transfert de fichiers, protocoles POP ou IMAP pour le courrier électronique...
- **couche 4 — couche transport :**
  Le protocole majeur de cette couche est le protocole TCP. Il s'assure par SYN-ACK que l'émetteur et le récepteur sont prêts à échanger des messages. Il découpe en segments numérotés le message à transmettre (côté émetteur) ou bien recompose le message total en remettant les segments dans l'ordre (côté récepteur). Les éléments échangés avec la couche inférieure sont des **segments**.
- **couche 3 — couche réseau :**
  C'est la couche où chaque segment numéroté est encapsulé dans un paquet qui, suivant le protocole IP, va contenir son adresse source et son adresse de destination. C'est à ce niveau que se décide si le message doit rester dans le réseau local ou être envoyé sur un autre réseau via la passerelle du routeur. Les éléments échangés avec la couche inférieure sont des **paquets**.
- **couche 2 — couche liaison :**
  C'est l'encapsulation finale du message. Suivant le protocole Ethernet, les informations sont transmises d'une carte réseau à une autre, grâce à leur adresse MAC (Media Access Controler). Les éléments échangés avec la couche inférieure sont des **trames**.
- **couche 1 — couche physique :**
  C'est la couche où le message est transmis physiquement d'un point à un autre. Par signal lumineux (fibre optique), par ondes (wifi), par courant électrique (Ethernet)... Les éléments transmis sont les **bits**.

Lors de son parcours, une trame peut être partiellement décapsulée et remonter à la couche 3, avant de redescendre et de continuer son chemin. C'est le cas notamment lors du passage dans un routeur. Mais jamais, lors de son acheminement, le contenu réel du message n'est ouvert : les paquets transmis sont acheminés de manière identique, qu'ils contiennent les éléments constitutifs d'une vidéo YouTube ou d'un email à votre cousin. Ce principe fondateur, actuellement menacé par certains acteurs politiques et industriels, est connu sous l'expression **« la neutralité du net »**.

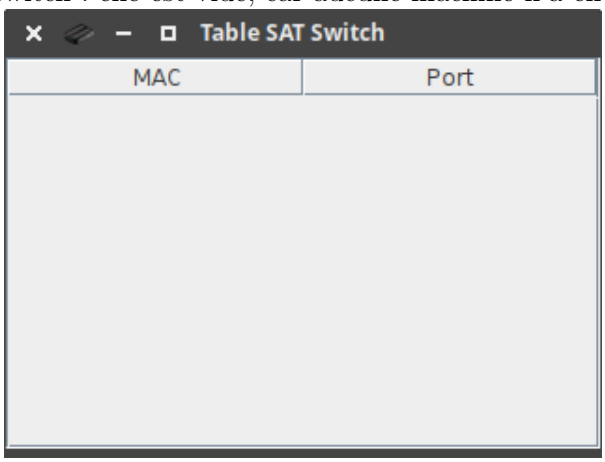
b) Observation des trames avec Filius

Ping à travers un switch

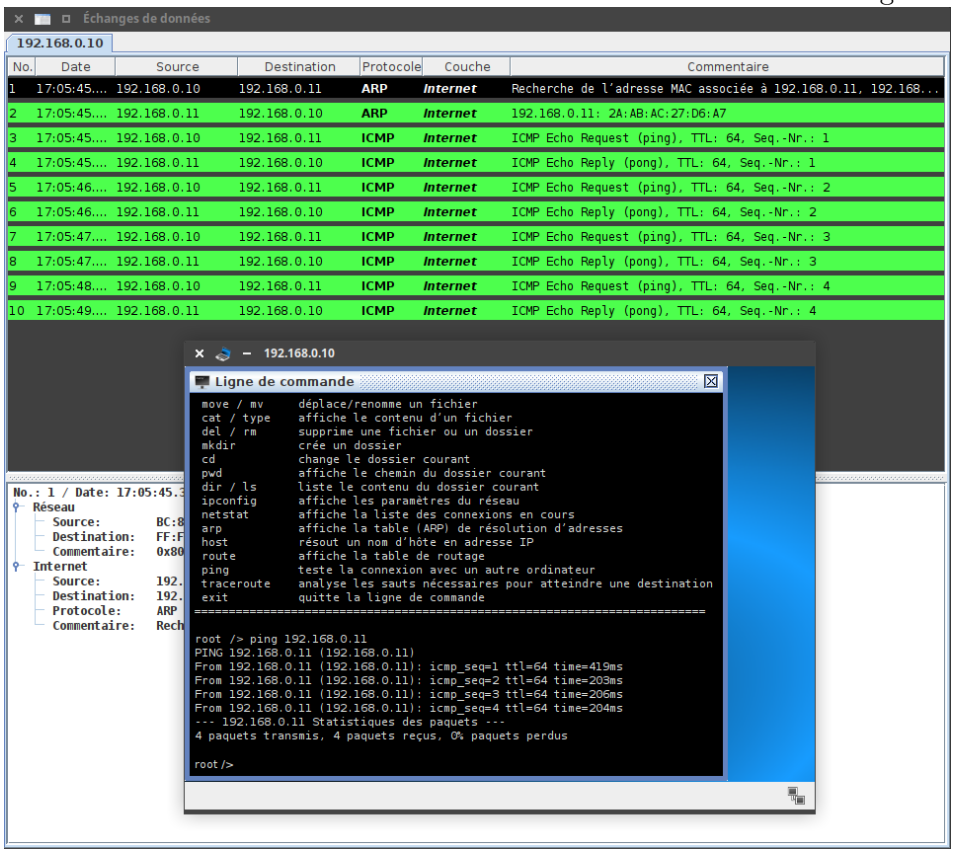
- Relions une machine `192.168.0.10` d'adresse MAC `BC:81:81:42:9C:31` à une machine `192.168.0.11` d'adresse MAC `2A:AB:AC:27:D6:A7` à travers un switch.



- Observons la table SAT de notre switch : elle est vide, car aucune machine n'a encore cherché à communiquer.



- Lançons un ping depuis `192.168.0.10` vers `192.168.0.11` et observons les données échangées :



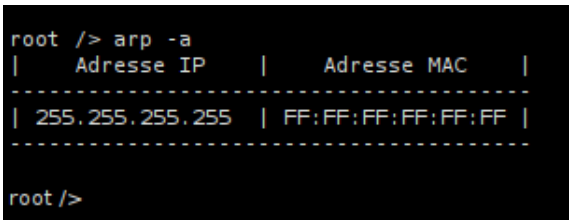
- Observons de plus près la première ligne de données échangées.

```
No.: 1 / Date: 17:05:45.331
Réseau
    Source:        BC:81:81:42:9C:31
    Destination:   FF:FF:FF:FF:FF:FF
    Commentaire:   0x806
Internet
    Source:        192.168.0.10
    Destination:   192.168.0.11
    Protocole:     ARP
    Commentaire:   Recherche de l'adresse MAC associée à 192.168.0.11, 192.168.0.10: BC:81:81:42:9C:31
```

Cette première ligne est une requête **ARP**. ARP (*Address Resolution Protocol*) est un protocole qui s'interface entre la couche 3 / réseau (appelée dans la capture d'écran Internet) et la couche 2 / liaison (appelée dans la capture d'écran Réseau). Comme indiqué dans le commentaire, elle consiste à un appel à tout le réseau :
« Est-ce que quelqu'un ici possède l'IP `192.168.0.11` ? »

Message 1 : « Qui possède l'IP `192.168.0.11` ? »

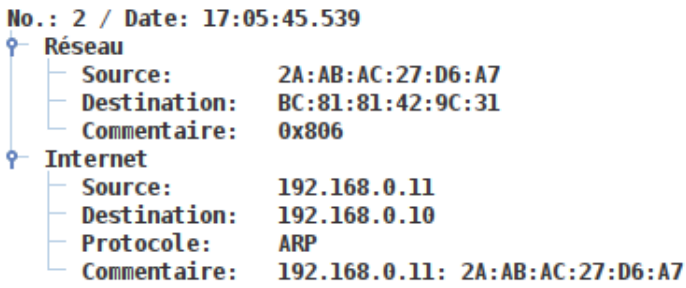
Il faut comprendre à cette étape que l'adresse IP est totalement inutile pour repérer un ordinateur dans un sous-réseau. Ce sont les adresses MAC qui permettent de se repérer dans un sous-réseau. Les adresses IP, elles, permettront éventuellement d'acheminer le message jusqu'au bon sous-réseau (elles n'intéressent donc que les routeurs).
Revenons à notre ping vers `192.168.0.11`.
La commande `arp -a` effectuée dans un terminal de la machine `192.168.0.10` nous permet de voir qu'elle ne connaît encore personne dans son sous-réseau. La table de correspondance IP ↔ MAC ne contient que l'adresse de broadcast `255.255.255.255`, qui permet d'envoyer un message à tout le réseau.



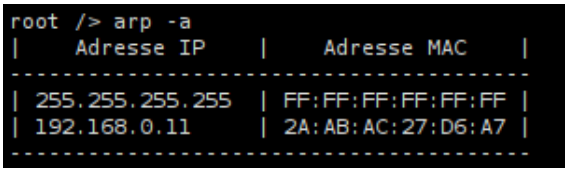
Constatant qu'elle ne sait pas quelle est l'adresse MAC de `192.168.0.11`, la machine `192.168.0.10` commence donc par envoyer un message à tout le sous-réseau, par l'adresse MAC de broadcast `FF:FF:FF:FF:FF:FF`. Le switch va lui aussi relayer ce message à tous les équipements qui lui sont connectés (dans notre cas, un seul ordinateur).

Message 2 : « Moi ! »

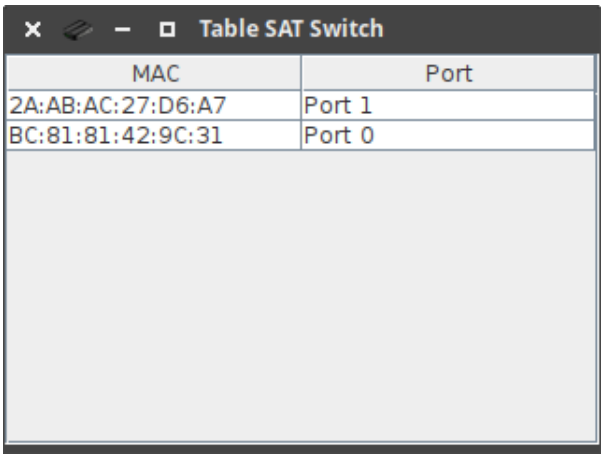
La machine `192.168.0.11` s'est reconnu dans le message de broadcast de la machine `192.168.0.10`. Elle lui répond pour lui donner son adresse MAC.



À partir de ce moment, la machine `192.168.0.10` sait comment communiquer avec `192.168.0.11`. Elle l'écrit dans sa table `ARP`, afin de ne plus avoir à émettre le message n°1 :



Le switch, qui a vu passer sur ses ports 0 et 1 des messages venant des cartes MAC `BC:81:81:42:9C:31` et `2A:AB:AC:27:D6:A7`, peut mettre à jour sa table SAT :



Par la suite, il saura sur quel port rediriger les messages destinés à ces deux adresses MAC. Un switch est un équipement de réseau de la couche 2 du modèle OSI, il ne sait pas lire les adresses IP : il ne travaille qu'avec les adresses MAC.

Message 3 : le ping est envoyé

```
3   17:05:45.... 192.168.0.10     192.168.0.11     ICMP    Internet    ICMP Echo Request (ping), TTL: 64, Seq.-Nr.: 1
```

```
No.: 3 / Date: 17:05:45.540
Réseau
    Source:         BC:81:81:42:9C:31
    Destination:    2A:AB:AC:27:D6:A7
    Commentaire:    0x800
Internet
    Source:         192.168.0.10
    Destination:    192.168.0.11
    Protocole:      ICMP
    Commentaire:    ICMP Echo Request (ping), TTL: 64, Seq.-Nr.: 1
```

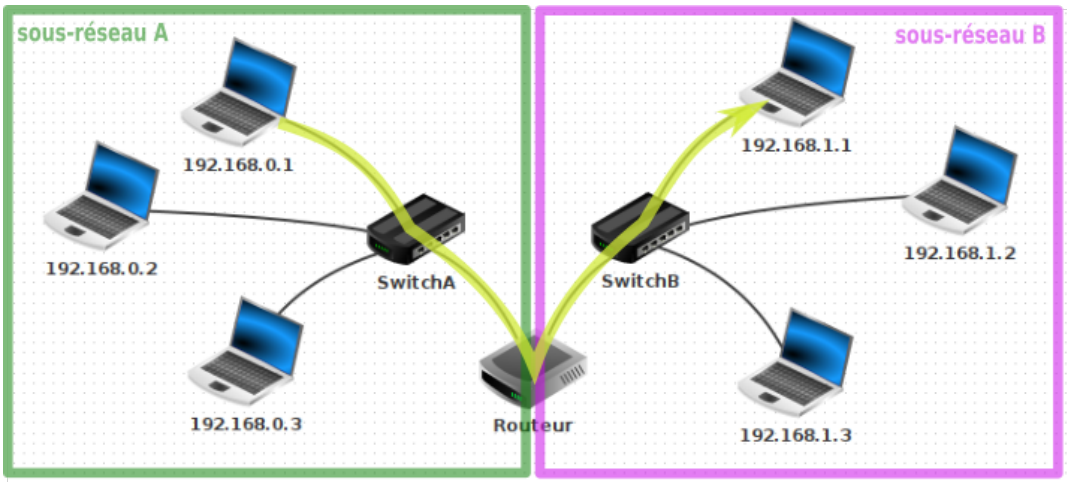
Schématisons cette trame Ethernet (couche 2 du modèle OSI) :

Message 4 : le pong est retourné

```
4   17:05:45.... 192.168.0.11     192.168.0.10     ICMP    Internet    ICMP Echo Reply (pong), TTL: 64, Seq.-Nr.: 1
```

```
No.: 4 / Date: 17:05:45.750
Réseau
    Source:         2A:AB:AC:27:D6:A7
    Destination:    BC:81:81:42:9C:31
    Commentaire:    0x800
Internet
    Source:         192.168.0.11
    Destination:    192.168.0.10
    Protocole:      ICMP
    Commentaire:    ICMP Echo Reply (pong), TTL: 64, Seq.-Nr.: 1
```

Ping à travers un routeur



L'objectif est d'observer les différentes trames lors d'un ping entre :

- la machine `192.168.0.1 / 24` (adresse MAC `F9:E1:D6:0B:29:03`) et
- la machine `192.168.1.1 / 24` (adresse MAC `D3:79:96:B8:5C:A4`)

Le routeur est configuré ainsi :

- interface sur le réseau A :
  IP : `192.168.0.254`
  MAC : `77:C2:22:C9:5C:E7`
- interface sur le réseau B :
  IP : `192.168.1.254`
  MAC : `66:E5:4E:7D:0B:B0`

Étape 0 : le routeur signale sa présence

Lors de l'observation des messages reçus ou émis par la machine `192.168.0.1`, on peut être intrigué par ce tout premier message reçu, émis par le routeur :

```
No.: 1 / Date: 18:20:45.824
Réseau
    Source:        77:C2:22:B9:5C:E7
    Destination:   FF:FF:FF:FF:FF:FF
    Commentaire:   0x800
Internet
    Source:        192.168.0.254
    Destination:   255.255.255.255
    Protocole:     IP
    Commentaire:   Protocole:17, TTL: 1
Transport
    Source:        521
    Destination:   520
    Protocole:     UDP
Application
    Commentaire:
        192.168.0.254
        192.168.1.254
        16
        75000
        192.168.1.0 255.255.255.0 0
```

On peut y distinguer les 4 couches du modèle Internet. Le routeur, par ce message distribué à tous les éléments du sous-réseau A (il envoie un message équivalent sur son sous-réseau B), déclare sa présence, et le fait qu'il possède deux interfaces, une pour chaque réseau. Il se positionne ainsi comme une passerelle : « c'est par moi qu'il faudra passer si vous voulez sortir de votre sous-réseau ». Dans cette trame envoyée figure son adresse MAC, de sorte que tous les membres de son sous-réseau pourront donc communiquer avec lui.

Étape 1 : de `192.168.0.1` vers le routeur

La machine `192.168.0.1 / 24` calcule que la machine `192.168.1.1 / 24` avec laquelle elle veut communiquer n'est pas dans son sous-réseau (avec le masque de sous-réseau). Elle va donc envoyer son message à sa passerelle, qui est l'adresse du routeur dans son sous-réseau. Cette première trame est :

Si vous regardez bien, l'adresse MAC destination n'est pas celle de la machine d'adresse IP `192.168.1.1 / 24`, mais celle du routeur qui sert de passerelle.

Étape 2 : le routeur décapsule la trame

Le routeur est un équipement de réseau de couche 3 (couche réseau). Il doit observer le contenu du paquet IP (sans remonter jusqu'au contenu du message) pour savoir, suivant le procédé de **routage** (voir cours de Terminale), où acheminer ce paquet.
Dans notre cas, l'adresse IP `192.168.1.1` de destination lui est accessible : elle fait partie de son sous-réseau B.
Le routeur va modifier la valeur du **TTL** (*Time To Live*), en la décrémentant de 1. Si, après de multiples routages, cette valeur devenait égale à 0, ce paquet serait détruit. Ceci a pour but d'éviter l'encombrement des réseaux avec des paquets ne trouvant pas leur destination.
Dans notre cas, le routeur va laisser intacte l'adresse IP Source. Ce n'est pas toujours le cas [4].

Le routeur va ré-encapsuler le paquet IP modifié, et créer une nouvelle trame Ethernet en modifiant :

- l'adresse MAC source : il va mettre l'adresse MAC de son interface dans le sous-réseau B.
- l'adresse MAC de destination : il va mettre l'adresse MAC de `192.168.1.1` (qu'il aura peut-être récupérée au préalable par le protocole ARP)

Cette deuxième trame est donc :

On peut observer dans Filius cette trame, en se positionnant sur l'interface `192.168.1.254` du routeur, ou sur **192.168.1.1** :

```
No.: 4 / Date: 18:21:03.179
Réseau
    Source:        66:E5:4E:7D:0B:B0
    Destination:   D3:79:96:B8:5C:A4
    Commentaire:   0x800
Internet
    Source:        192.168.0.1
    Destination:   192.168.1.1
    Protocole:     ICMP
    Commentaire:   ICMP Echo Request (ping), TTL: 63, Seq.-Nr.: 1
```

En suivant le même principe, la machine `192.168.1.1` pourra envoyer son pong.

4. Dans le cas classique de la box qui relie votre domicile à internet, le routeur contenu dans celle-ci va remplacer l'adresse locale de votre ordinateur ou smartphone (ex `192.168.0.26`) par son IP publique (celle apparaissant sur `http://whatsmyip.com`, par exemple). Elle effectue ce qu'on appelle une translation d'adresse (NAT).
Pourquoi ? Parce que sinon la réponse du serveur distant que vous interrogez serait envoyée sur une adresse locale (votre adresse `192.168.0.26`), qui est introuvable depuis un réseau extérieur. Il faut donc remplacer toutes les adresses locales par l'IP publique de votre box.
Pour éviter que la réponse du serveur web que vous avez interrogé ne soit affichée sur l'ordinateur de vos parents, le routeur affecte des ports différents à chaque machine de son sous-réseau. Ce port est inclus dans le message transmis au serveur, et il l'est aussi dans sa réponse : le routeur peut donc rediriger le trafic vers la bonne machine du sous-réseau.

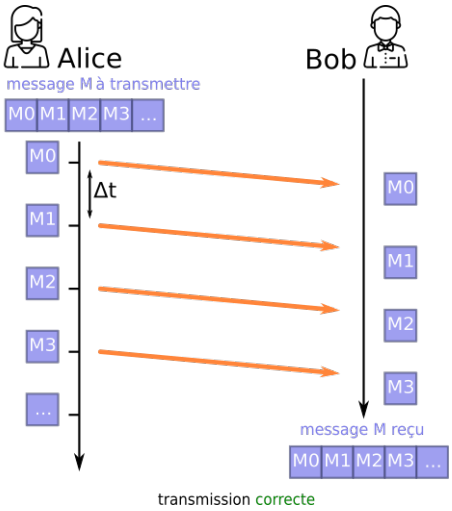
c) Protocole du bit alterné

Ce protocole est un exemple simple de fiabilisation du transfert de données.

Contexte

- Alice veut envoyer à Bob un message M, qu'elle a prédécoupé en sous-messages M0, M1, M2,...
- Alice envoie ses sous-messages à une cadence Δt fixée (en pratique, les sous-messages partent quand leur acquittement a été reçu ou qu'on a attendu celui-ci trop longtemps : on parle alors de *timeout*)

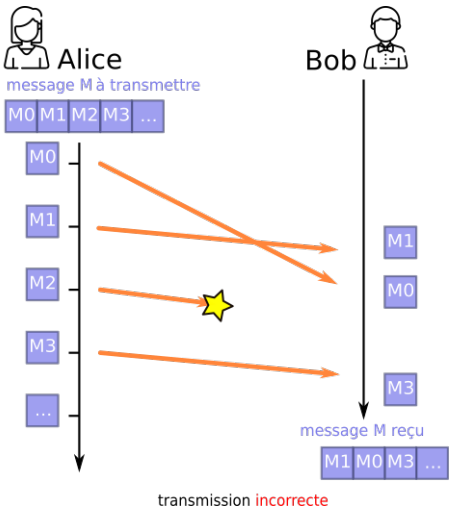
Situation idéale



Dans cette situation, les sous-messages arrivent tous à destination dans le bon ordre. La transmission est correcte.

Situation réelle

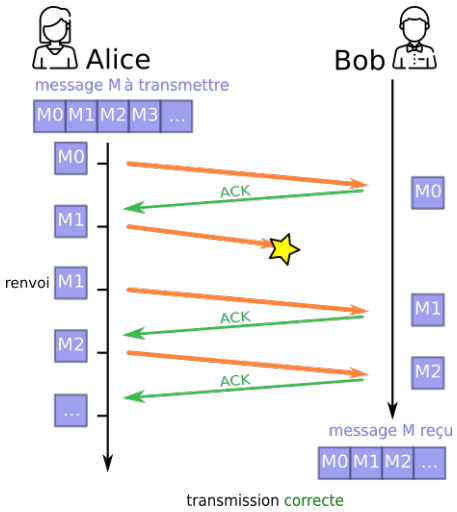
Mais parfois, les choses ne se passent pas toujours aussi bien. Car si on maîtrise parfaitement le timing de l'envoi des sous-messages d'Alice, on ne sait pas combien de temps vont mettre ces sous-messages pour arriver, ni même (attention je vais passer dans un tunnel) s'ils ne vont pas être détruits en route.



Le sous-message M0 est arrivé après le M1, le message M2 n'est jamais arrivé...
Que faire ?
Écartons l'idée de numéroter les sous-messages, afin que Bob puisse remettre dans l'ordre les messages arrivés, ou même redemander spécifiquement des sous-messages perdus. C'est ce que réalise le protocole TCP (couche 4 — transport), c'est très efficace, mais cher en ressources. Essayons de trouver une solution plus basique.

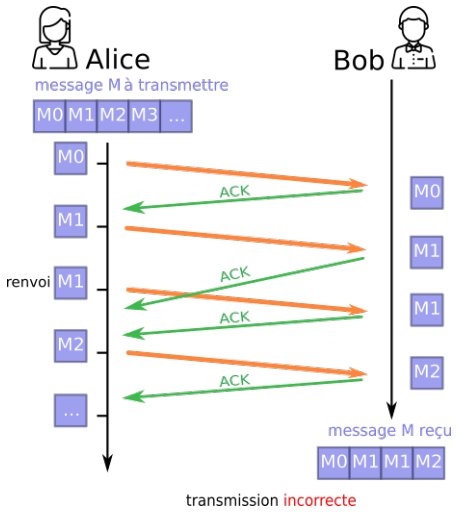
Solution naïve...

Pourquoi ne pas demander à Bob d'envoyer un signal pour dire à Alice qu'il vient bien de recevoir son sous-message ? Nous appelerons ce signal ACK (comme *acknowledgement*, traduisible par « accusé de réception »).
Ce signal ACK permettra à Alice de renvoyer un message qu'elle considérera comme perdu :



N'ayant pas reçu le ACK consécutif à son message M1, Alice suppose (avec raison) que ce message n'est pas parvenu jusqu'à Bob, et donc renvoie le message M1.

Mais peu efficace...



Le deuxième ACK de Bob a mis trop de temps pour arriver (ou s'est perdu en route) et donc Alice a supposé que son sous-message M1 n'était pas arrivé. Elle l'a donc renvoyé, et Bob se retrouve avec deux fois le sous-message M1. La transmission est incorrecte. En faisant transiter un message entre Bob et Alice, nous multiplions par 2 la probabilité que des problèmes techniques de transmission interviennent. Et pour l'instant rien ne nous permet de les détecter.

Bob prend le contrôle

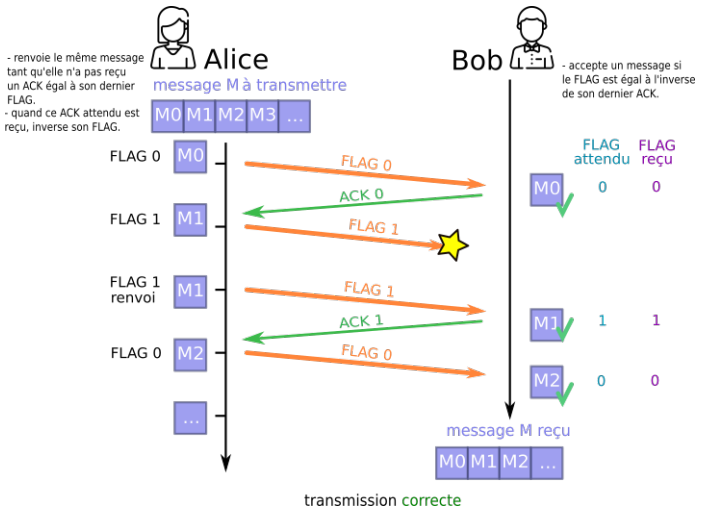
Bob va maintenant intégrer une méthode de validation du sous-message reçu. Il pourra décider de le garder ou de l'écarter. Le but est d'éviter les doublons. Pour réaliser ceci, Alice va rajouter à chacun de ses sous-messages un bit de contrôle, que nous appellerons **FLAG** (drapeau). Au départ, ce FLAG vaut 0.
Quand Bob reçoit un FLAG, il renvoie un ACK **égal au FLAG reçu**.
Alice va attendre ce ACK contenant le même bit que son dernier FLAG envoyé :

- tant qu'elle ne l'aura pas reçu, elle continuera à envoyer **le même sous-message, avec le même FLAG**.
- dès qu'elle l'a reçu, elle peut envoyer un nouveau sous-message en **inversant** (« alternant ») **le bit de son dernier FLAG** (d'où le nom de ce protocole).

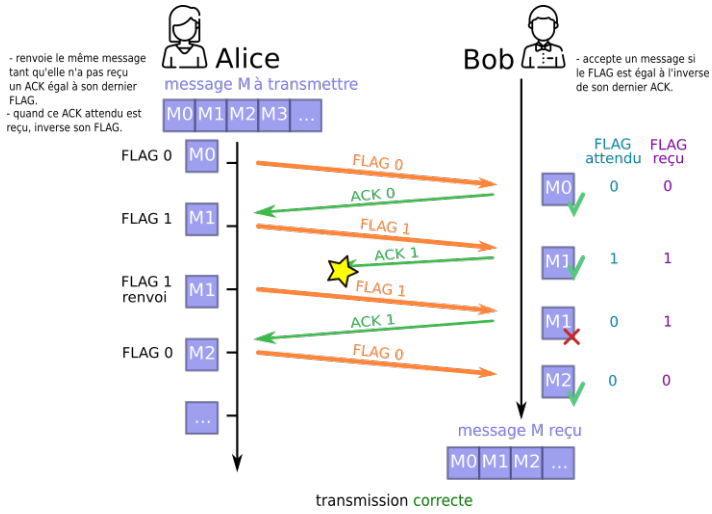
Bob, de son côté, va contrôler la validité de ce qu'il reçoit : il ne gardera que **les sous-messages dont le FLAG est égal à son dernier ACK**. C'est cette méthode qui lui permettra d'écarter les doublons.

Observons ce protocole dans plusieurs cas :

Cas où le sous-message est perdu

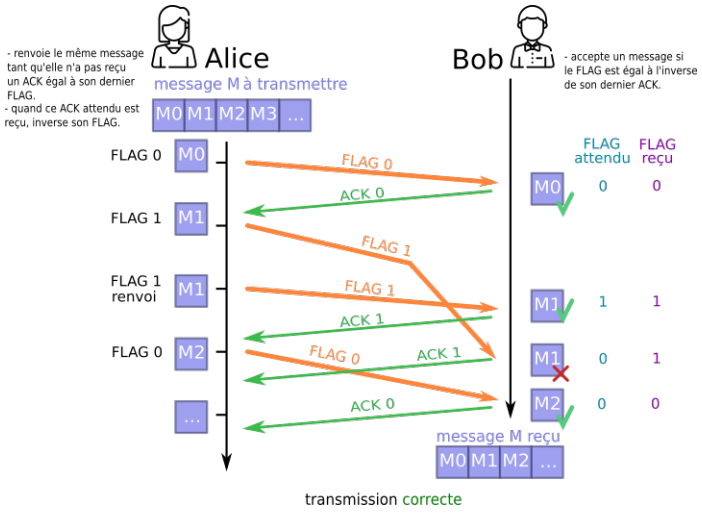


Cas où le ACK est perdu



Le protocole a bien détecté le doublon du sous-message M1.

Cas où un sous-message est en retard



Le protocole a bien détecté le doublon du sous-message M1... mais que se passerait-il si notre premier sous-message M1 était encore plus en retard ?

d) Conclusion

Le protocole du bit alterné a longtemps été utilisé au sein de la couche 2 du modèle OSI (distribution des trames Ethernet). Simple et léger, il peut toutefois être facilement mis en défaut, ce qui explique qu'il ait été remplacé par des protocoles plus performants.

III. D'autres protocoles ?

Il existe de multiples protocoles standardisés de communication en fonction du type d'information transmise, en fonction aussi de la source et de la déstination. `https://tools.ietf.org/html/rfc1149`

IV. Exercices

a) Masque de sous réseau

Exercice 2
Montrer que 145 & 248 = 144.

V. Corrections

a) Masque de sous réseau

Correction de l'exercice 2
Il faut commencer par écrire les nombres 145 et 248 en binaire.
Pour cela, rappelez-vous, une des méthodes consiste à trouver la plus grande puissance de 2 que l'on peut extraire du nombre, puis son recommence avec le reste. Pour 1 octet, les puissances de 2 sont :

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

$145 = 128 + 17$
$145 = 128 + 16 + 1$
Donc :
$145_{10} = 1001\ 0001_2$

De même :
$248 = 128 + 120$
$248 = 128 + 64 + 56$
$248 = 128 + 64 + 32 + 24$
$248 = 128 + 64 + 32 + 16 + 8$
Donc :
$248_{10} = 1111\ 1000_2$

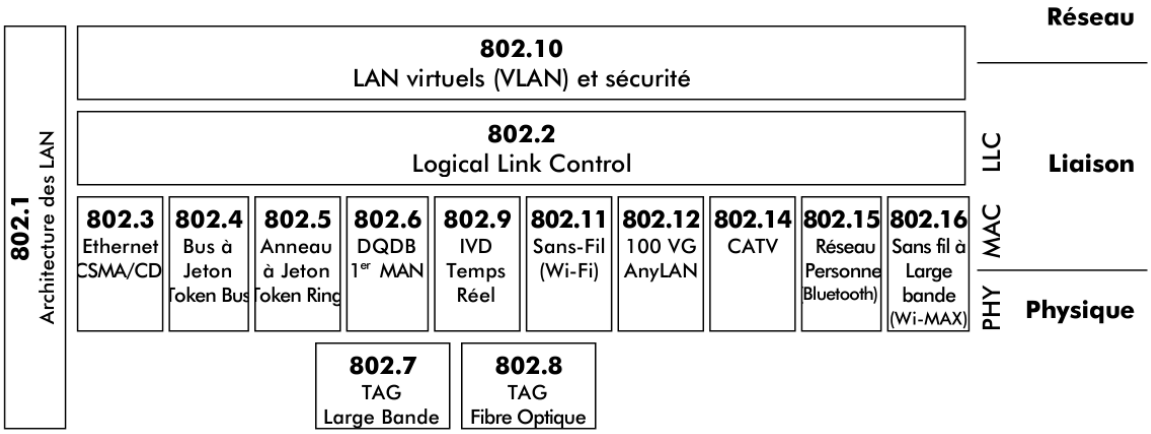
On effectue ensuite le & (ET) bit par bit :

	1	0	0	1	0	0	1	1	(145)
&	1	1	1	1	1	0	0	0	(248)
	1	0	0	1	0	0	0	0	

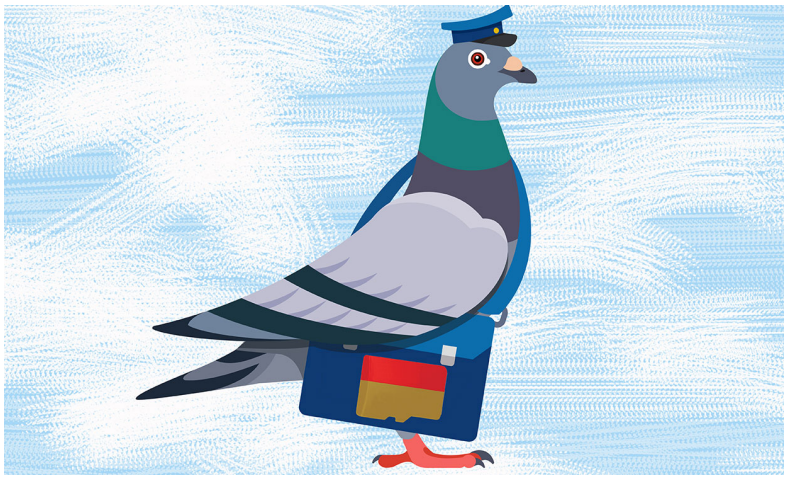
Il ne reste plus qu'à convertir 1001 0000 en décimale :
$1001\ 0000_2 = 128 + 16 = 144_{10}$

Annexe A - Les protocoles

Initiées par liIEEE (*Institute of Electrical and Electronics Engineers*), des travaux de normalisation des réseaux locaux ont débuté en 1979. L'objectif est adapter les couches 1 et 2 du modèle OSI (*Open System Interconnection*) aux particularités des réseaux LAN (locaux) et MAN (métropolitains).
En février 1980, le groupe de travail a pris le nom de groupe 802 (80 indiquant l'année et 2 indiquant le mois). Le but du comité IEEE 802 est de développer un standard permettant la transmission de trames d'information entre deux systèmes différents via un support partagé.
En 1982, le comité 802 a été réorganisé et plusieurs groupes ont vu le jour. Chaque groupe (802.x) est concerné par un sujet particulier pour lequel un standard est élaboré portant le nom du groupe. Les travaux ayant abouti au sein d'un sous-comité 802.x ont donné lieu à des normes ISO de la série 8802.x (IS 8802.x)



Le 1er avril 1990, David Waitzman proposa un protocole de communication au moyen d'un transporteur aérien ... qui n'est autre que le pigeon. On peut retrouver ce protocole à l'adresse suivante : `https://tools.ietf.org/html/rfc1149`.
Ce protocole a été plusieurs modifié, comme par exemple le 1er avril 2011 par Brian Carpenter et Robert M. Hinden pour prendre en compte le protocole **IPv6**. Voici le protocole : `https://tools.ietf.org/html/rfc6214`.
Pour plus d'information sur le sujte, vous pouvez lire l'article suivant, écrit sur le site très sérieux du IEEE, le 1er avril 2019 : `https://spectrum.ieee.org/tech-talk/computing/networks/pigeonbased-feathernet-still-wingsdown-fastest-way-`
(L'image du pigeon voyageur provient de ce site).



Annexe B - Les sigles

ACK (*acknowledgement*) l'acquittement d'une donnée ou d'une information consiste à informer son émetteur de sa bonne réception.

ARP (*Address Resolution Protocol*) 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).

CIDR (*Classless Inter-Domain Routing*) notation de l'adresse IP qui intègre le masque de sous-réseau.

FTP (*File Transfer Protocol*) protocole de communication destiné au partage de fichiers sur un réseau TCP/IP.

HTTP (*Hypertext Transfer Protocol*) protocole de communication client-serveur développé pour le World Wide Web.

IMAP (*Internet Message Access Protocol*) protocole qui permet à un client de messagerie d'accéder aux messages stockés sur un serveur de messagerie. Il est plus moderne et plus riche en fonctionnalités que le protocole POP.

IP (*Internet Protocol*) numéro d'identification attribué de façon permanente ou provisoire à chaque périphérique relié à un réseau informatique qui utilise l'Internet Protocol.

DNS (*Domain Name System*) service informatique utilisé pour traduire les noms de domaine Internet en adresse IP.

MAC (*Media Access Control*) adresse physique d'une carte réseau.

OSI (*Open Systems Interconnection*) norme de communication, en réseau, de tous les systèmes informatiques.

POP (*Post Office Protocol*) protocole qui permet de récupérer les courriers électroniques situés sur un serveur de messagerie électronique.

TCP/IP (*Transmission Control Protocol* et *Internet Protocol*) ensemble des protocoles TCP et IP utilisés pour le transfert des données sur Internet.

TTL (*Time To Live*) indique le temps pendant lequel une information doit être conservée.

WIFI (*Wireless Fidelity*) ensemble de protocoles de communication sans fil.