Architectures matérielles et systèmes d’exploitation

Suite

Transmission de données dans un réseau

Protocoles de communication

Architecture d’un réseau

Compétences attendues :

- Architecture d’un réseau
- 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 présents

1. Modèle OSI, modèle Internet

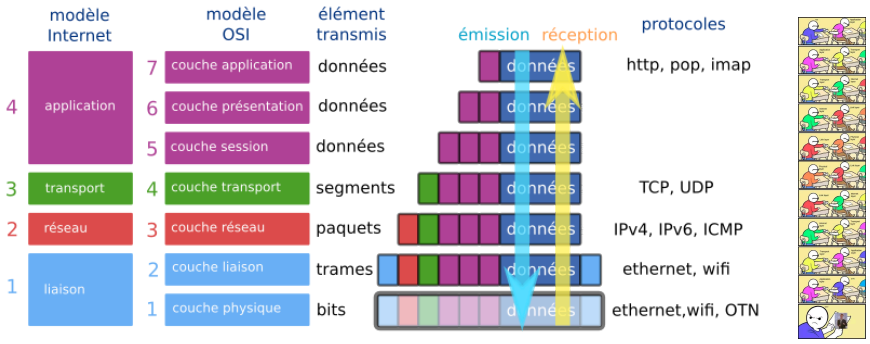
La transmission de données entre ordinateurs est un processus complexe qui nécessite divers types d'informations pour garantir que les bits atteignent la bonne destination et soient correctement interprétés. Cette complexité est gérée par des protocoles de communication, organisés en couches dans des modèles théoriques.

Deux modèles principaux définissent ces protocoles :

- Le modèle Internet (ou TCP/IP), créé en 1974, est organisé en 4 couches :
  - liaison,
  - réseau,
  - transport
  - application.
- Le modèle OSI (Open Systems Interconnection), créé en 1984, est plus détaillé avec 7 couches :
  - physique,
  - liaison,
  - réseau,
  - transport,
  - session,
  - présentation
  - application.

Les deux modèles sont théoriques et peuvent parfois se chevaucher dans la pratique. Néanmoins, ils fournissent un cadre pour comprendre la complexité des communications réseau. Dans les discussions futures, les couches seront généralement référencées par leur numéro dans le modèle OSI pour plus de clarté.

Ces modèles aident à comprendre comment les bits, contenant à la fois des données utiles et des informations d'acheminement, voyagent d'un point à un autre dans un réseau.



Un message passe par toutes les couches du modèle OSI ou TCP/IP lors de son envoi, de la création à la transmission physique.
À la réception, le message fait le chemin inverse à travers les couches pour être décodé et livré au destinataire.

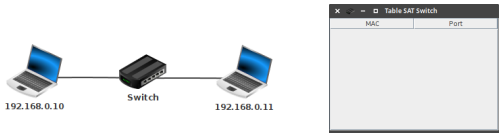
- Couches Application-Présentation-Session (7-6-5)
  - Fusionnées en une seule couche "application" dans le modèle Internet.
  - S'occupent de la mise en forme des messages pour la transmission.
  - Utilisent divers protocoles tels que HTTP, FTP, POP, et IMAP selon le type de données à transmettre.
- Couche Transport (4)
  - Utilise principalement le protocole TCP.
  - Établit une connexion sûre via SYN-ACK.
  - Découpe les messages en segments numérotés pour l'émission et les recompose à la réception.
  - Échange des "segments" avec la couche inférieure.
- Couche Réseau (3)
  - Encapsule chaque segment en un paquet avec des adresses IP source et de destination.
  - Décide si le message doit rester dans le réseau local ou être transmis à un autre réseau.
  - Échange des "paquets" avec la couche inférieure.
- Couche Liaison (2)
  - Dernière encapsulation du message en trames.
  - Transmet les trames entre les cartes réseau via des adresses MAC.
  - Échange des "trames" avec la couche inférieure.
- Couche Physique (1)
  - Transmission physique des bits par divers moyens comme la fibre optique, le wifi, ou le courant électrique.

2. Observation des tram

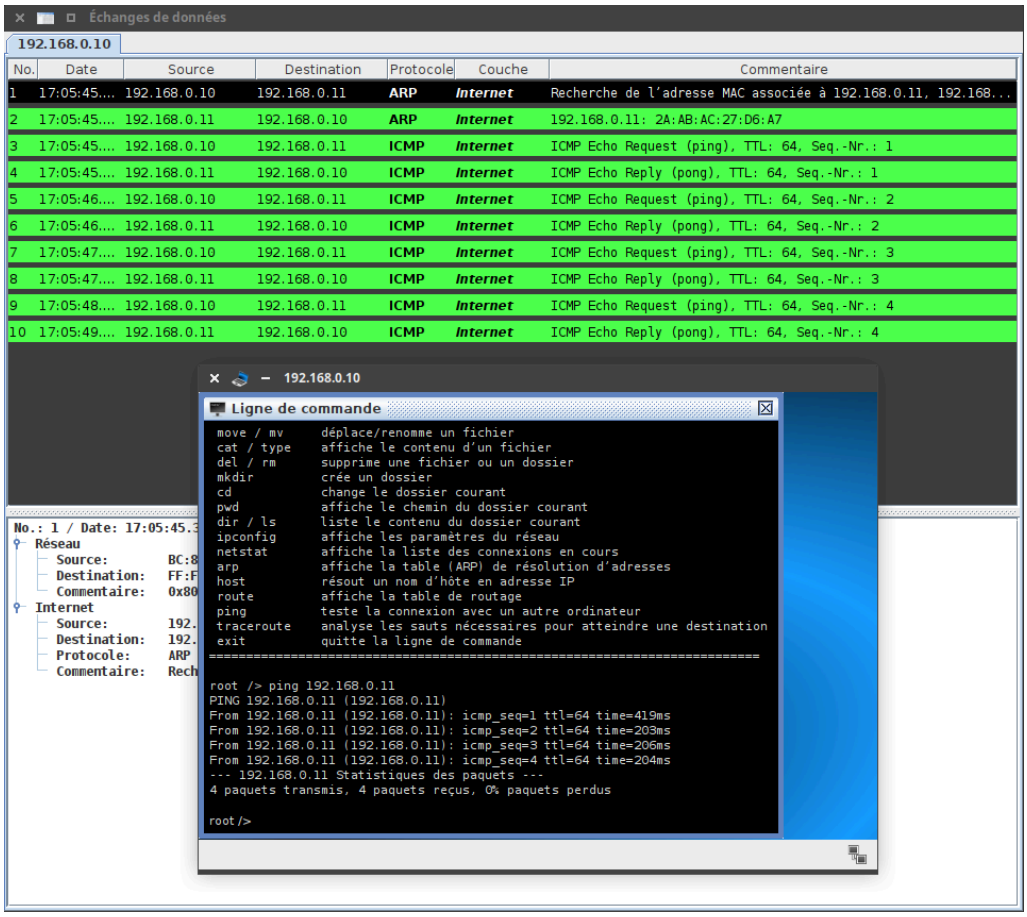
2.1. Ping à travers un switch

Télécharger le fichier ping_switch.fls.

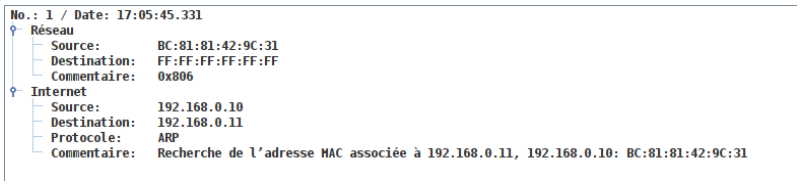
- Une machine 192.168.0.10 d'adresse MAC BC:81:81:42:9C:31 est reliée à une machine 192.168.0.11 d'adresse MAC 2A:AB:AC:27:D6:A7 à travers un switch. Observons la table SAT de notre switch : elle est vide, car aucune machine n'a encore cherché à communiquer.



- 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.



Cette première ligne est une requête ARP. ARP 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 répérer 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.

```
root /> arp -a
|    Adresse IP    |    Adresse MAC    |
---------------------------------------
| 255.255.255.255  | FF:FF:FF:FF:FF:FF |
---------------------------------------

root />
```

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 lui aussi relayer ce message à tous les équipements qui lui sont connectés (dans notre cas, un seul ordinateur)

Message 2 : « Moi ! »

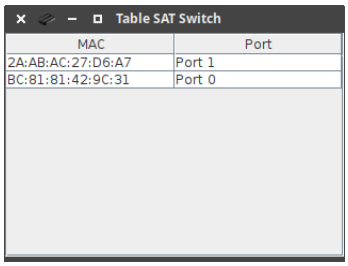
La machine 192.168.0.11 s'est reconnu dans le message de broadcast de la machine 192.168.0.10. Elle lui répond pour lui donner son adresse MAC.

```
No.: 2 / Date: 17:05:45.539
Réseau
    Source:         2A:AB:AC:27:D6:A7
    Destination:    BC:81:81:42:9C:31
    Commentaire:    0x806
Internet
    Source:         192.168.0.11
    Destination:    192.168.0.10
    Protocole:      ARP
    Commentaire:    192.168.0.11: 2A:AB:AC:27:D6:A7
```

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 :

```
root /> arp -a
|    Adresse IP    |    Adresse MAC    |
---------------------------------------
| 255.255.255.255  | FF:FF:FF:FF:FF:FF |
| 192.168.0.11     | 2A:AB:AC:27:D6:A7 |
---------------------------------------
```

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

```
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

```
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
```

2.2. Ping à travers un routeur

Télécharger le fichier ping_routeur.fls.

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

- l'ordinateur du réeau A :
  - IP : 192.168.0.1
  - MAC : F9:E1:D6:0B:29:03
- et l'ordinateur du réseau B :
  - IP : 192.168.1.1
  - 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 2 messages émis à tous par le routeur sur les reseaux A et B, on peut être intrigué :

```
No.: 1 / Date: 20:03:22.669
Réseau
    Source:        77:C2:22:B9:5C:E7
    Destination:   FF:FF:FF:FF:FF:FF
    Commentaire / Détail: 0x800
Internet
    Source:        192.168.0.254
    Destination:   255.255.255.255
    Protocole:     IP
    Commentaire / Détail: Protocole : 17, TTL: 1
Transport
    Source:        521
    Destination:   520
    Protocole:     UDP
Application
    Protocole:     RIP
    Commentaire / Détail (64 Bytes):
        192.168.0.254
        192.168.0.254
        16
        75000
        192.168.1.0 255.255.255.0 0
```

```
No.: 1 / Date: 20:03:22.670
Réseau
    Source:        66:E5:4E:7D:0B:B0
    Destination:   FF:FF:FF:FF:FF:FF
    Commentaire / Détail: 0x800
Internet
    Source:        192.168.1.254
    Destination:   255.255.255.255
    Protocole:     IP
    Commentaire / Détail: Protocole : 17, TTL: 1
Transport
    Source:        521
    Destination:   520
    Protocole:     UDP
Application
    Protocole:     RIP
    Commentaire / Détail (64 Bytes):
        192.168.1.254
        192.168.0.254
        16
        75000
        192.168.0.0 255.255.255.0 0
```

On peut y distinguer les 4 couches du modèle Internet.
Le routeur, par ces 2 pour déclare sa présence, et le fait qu'il possède deux interfaces 192.168.1.254 et 192.168.0.254, une pour chaque réseau. Il se positionne ainsi comme une passerelle entre les 2 sous-réseaux.

Dans ces trame envoyée figure ses adresses MAC 77:C2:22:B9:5C:E7 et 66:E5:4E:7D:0B:B0, 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 conprends que la machine 192.168.1.1 avec laquelle elle veut communiquer n'est pas dans son 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 :

```
No.: 4 / Date: 20:03:46.815
Réseau
    Source:        F9:E1:D6:0B:29:03
    Destination:   77:C2:22:B9:5C:E7
    Commentaire / Détail: 0x800
Internet
    Source:        192.168.0.1
    Destination:   192.168.1.1
    Protocole:     ICMP
    Commentaire / Détail: ICMP Echo Request (ping), TTL: 64, Seq.-No.: 1
```

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, 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.

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.

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: 20:03:47.162
Réseau
    Source:        66:E5:4E:7D:0B:B0
    Destination:   D3:79:96:B8:5C:A4
    Commentaire / Détail: 0x800
Internet
    Source:        192.168.0.1
    Destination:   192.168.1.1
    Protocole:     ICMP
    Commentaire / Détail: ICMP Echo Request (ping), TTL: 63, Seq.-No.: 1
```

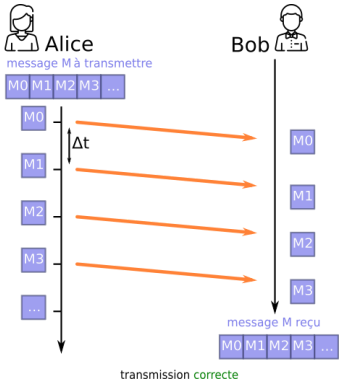
3. Protocole du bit alterné

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

1. 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)

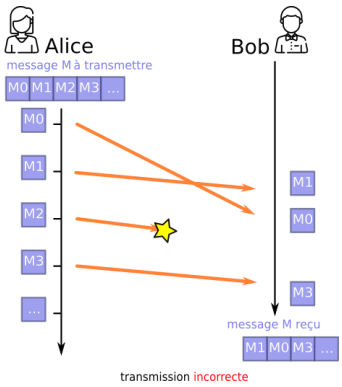
2. Situation idéale



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

3. 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.



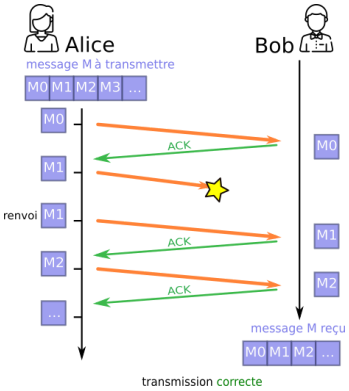
Le sous-message M0 est arrivé après le M1, le message M2 n'est jamais arrivé...

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.

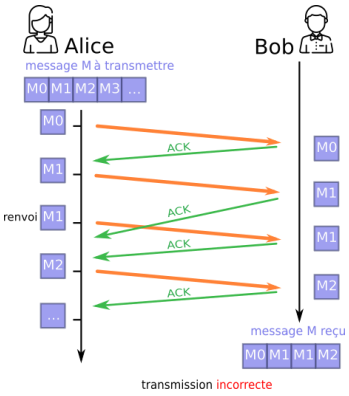
3. Solution naïve...

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.

4. 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.

5. 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 appelerons FLAG (drapeau). Au départ, ce FLAG vaut 0. Quand Bob reçoit un FLAG, il renvoie un ACK égal au FLAG reçu.

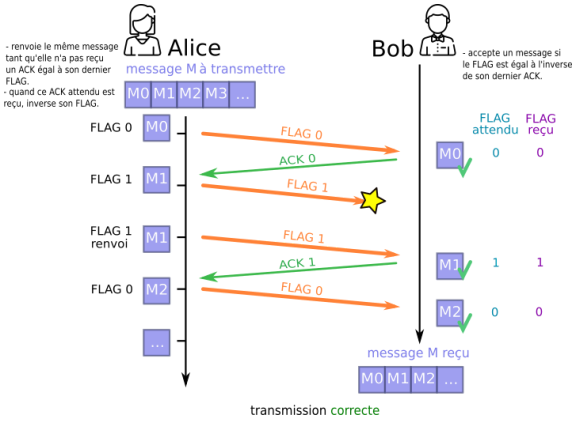
Alice va attendre ce ACK contenant le même bit que son dernier FLAG envoyé :

- 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).

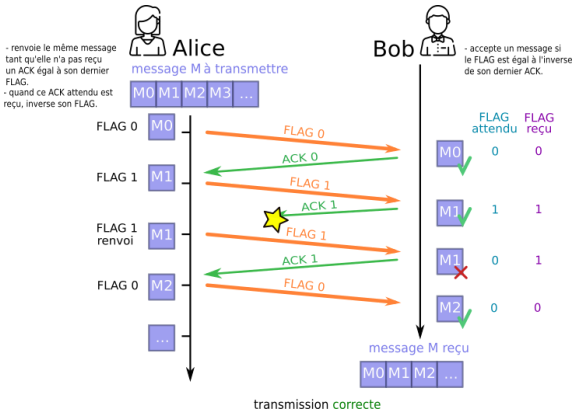
Bob, de son côté, va contrôler la validité de ce qu'il reçoit : il ne gardera que les sous-messages dont le FLAG est égal à l'inverse de son dernier ACK. C'est cette méthode qui lui permettra d'écarter les doublons.

Observons ce protocole dans plusieurs cas :

5.1 Cas où le sous-message est perdu

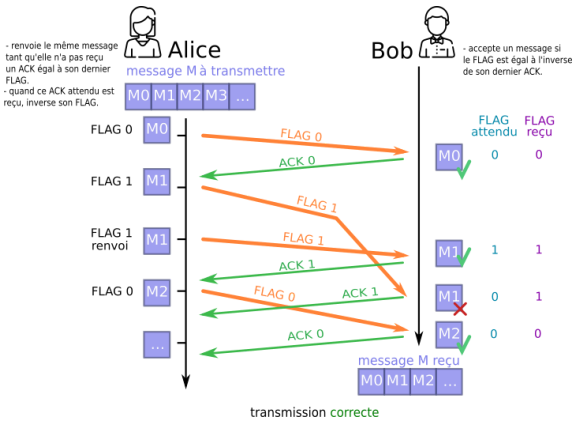


5.2 Cas où le ACK est perdu



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

5.3 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 ?

6. 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.

source : glassus