Chapitre 3

Le réseau Éthernet

Ce chapitre décrit la façon dont se passe la connexion de deux ordinateurs en utilisant le réseau Éthernet, le plus fréquemment utilisé à l'heure actuelle pour relier entre eux les ordinateurs à travers un réseau local.

Le contenu du chapitre correspond aux couches 1 et 2 (physique et liaison) du modèle OSI.

Il est important de garder à l'esprit que, si ce chapitre s'appuie sur le réseau Éthernet, il existe d'autres techniques pour relier les ordinateurs dont les modes de fonctionnements sont différents, par exemple les réseaux Token Ring (issu de la société IBM, ce fut un concurrent sérieux pour Éthernet) ou FDDI (qui utilise des anneaux de fibre optique). Nous n'aborderons pas les détails de ces techniques ici mais nous nous souviendrons qu'ils existent et que les problèmes résolus par Éthernet peuvent être résolus d'autres manières par d'autres techniques.

Éthernet s'appuie sur un principe appelé le CSMA/CD comme *Carrier Sense Multiple Access / Collision Detection*.

3.1 Le principe du CSMA/CD

Le principal problème pour partager un médium et connecter des ordinateurs indépendants, c'est de s'assurer qu'il n'y a qu'un seul ordinateur qui utilise le réseau à un moment donné ; si plusieurs utilisent le medium de communication en même temps, le message est incompréhensible (figure 3.1).

Une solution consiste à donner à l'un des ordinateurs le rôle d'un président de séance : quand un ordinateur veut émettre, il doit d'abord demander l'autorisation à cette machine (figure 3.2). L'inconvénient évident, c'est qu'il faut des communications supplémentaires pour échanger des données (demande d'autorisation, autorisation, envoi des données) et que si le coordinateur tombe en panne plus rien de marche.

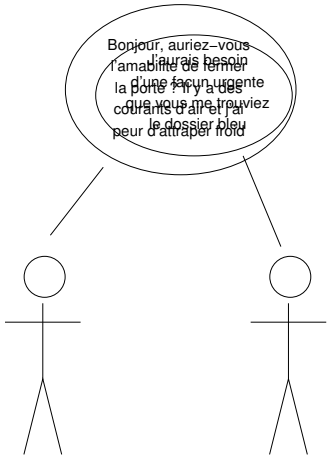

FIGURE 3.1 – Quand on a plusieurs émetteurs simultanés, les messages sont incompréhensibles. Ceci est vrai dans la communication orale entre personnes humaines aussi bien que dans la communication entre ordinateurs dans un réseau informatique.

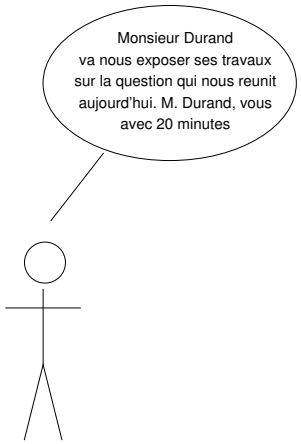

FIGURE 3.2 – Pour contrôler l'usage du médium de communication, une solution est un président de séance qui distribue la parole.

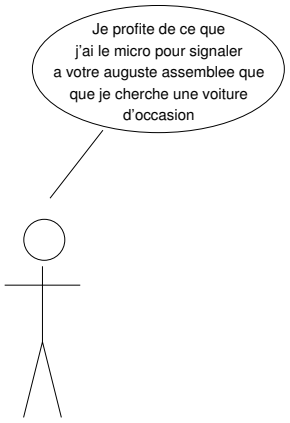

FIGURE 3.3 – Dans une réunion, le micro peut servir de *jeton* : celui qui l'a est le seul qui puisse s'exprimer.

Une autre manière de faire consiste à utiliser un *jeton* qui circule entre les ordinateurs : un ordinateur qui souhaite émettre attend de recevoir le jeton ; quand il tient le jeton, il émet puis il passe le jeton à son voisin. C'était le principe de base du réseau Token Ring. Un problème se produit quand l'ordinateur qui détient le jeton tombe en panne : il faut alors détecter qu'il n'y a plus de jeton en circulation et coordonner toutes les machines pour en fabriquer un autre.

Le réseau Éthernet résout ce problème de coordination d'une façon originale et intéressante : quand un ordinateur souhaite émettre, il s'assure que le médium n'est pas utilisé puis commence à émettre : c'est le CSMA (*Carrier Sense Multiple Access*. Malheureusement, rien ne garantit qu'il n'y a pas un autre ordinateur qui souhaitait émettre lui aussi, qui a écouté sur le médium, qui a constaté qu'il n'y avait pas d'émission au même moment et qui a commencé à émettre (presque) exactement en même temps que le premier ; il se produit dans ce cas une *collision* et aucun des deux messages n'est transmis correctement.

Pour résoudre le problème des collisions l'ordinateur écoute sur le médium le message transporté en même temps qu'il l'émet : s'il y a une collision, alors le message émis ne sera pas celui qu'il reçoit ; l'ordinateur identifie alors une collision. C'est le CD du CSMA/CD, comme *Collision Detection*.

Dans le cas où il détecte une collision, les choses se passent d'une façon qui rappelle ce qui se passe quand deux personnes bien élevées commencent à parler en même temps (figure 3.4). L'ordinateur attend pendant un intervalle de temps tiré au hasard puis re-tente d'émettre ; s'il se produit une nouvelle collision, il recommence en tirant l'intervalle de temps dans un intervalle de temps deux fois plus grand, jusqu'à ce que la transmission réussisse ou que le nombre de tentatives devienne trop grand : dans ce cas, il considère qu'il y a une erreur d'écriture sur le réseau.

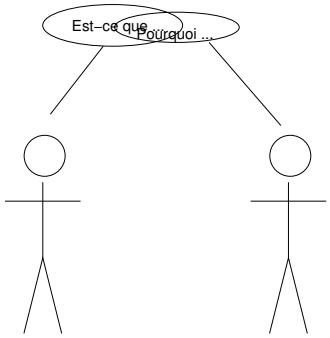

FIGURE 3.4 – Il arrive que deux personnes qui discutent commencent une phrase (presque) en même temps. Normalement, les deux s'arrêtent, puis l'une d'elles reprend sa phrase au début. C'est à peu près sur ce mode que fonctionne le CSMA/CD du réseau Ethernet.

3.1.1 Les implications du CSMA/CD

Pour que la détection de collision soit possible dans tous les cas, il faut d'une part que les messages envoyés soient suffisamment longs et d'autre part que la distance maximale ne soit pas trop grande, pour que la collision, si elle se produit, soit détectée avant la fin de l'émission du message.

Imaginons un réseau élémentaire avec seulement deux ordinateurs A et B connectés sur câble à une distance d. Il n'y a pas de signal et A commence à émettre. Le signal se propage sur le câble à à environ 300 000 kilomètres par seconde. Juste avant que le signal arrive à la machine B, celle-ci commence elle aussi à envoyer une trame. Pour que A détecte la collision, il faut que le début de la trame émise par B arrive à la machine A avant qu'elle n'aie terminé d'émettre sa propre trame. Le temps pour que la trame envoyée par A atteigne B puis celle émise par B atteigne A est de $2 \times d/300\,000$ secondes. Soit v la vitesse d'émission (en bits par seconde) et l la longueur minimum d'un message en bits. La durée d'émission d'un message est $v \times l$. Il faut donc que $v \times l > 2 \times d/300\,000$

Plus le débit du réseau est grand, plus les messages sont courts et plus la distance entre les deux ordinateurs est limitée pour s'assurer que toutes les collisions sont détectées.

En fonctionnement normal, il peut parfois se produire des collisions sans que cela nuise significativement au débit du réseau ; il arrive cependant un moment (entre 66 et 80 pour cent d'utilisation de la capacité nominale du réseau) où les tentatives d'émission débouchent pour la plupart sur des collisions qui empêchent presque complètement la transmission des données. (C'est l'équivalent ce ces réunions houleuses dans lesquelles tout le monde essaye résolument de parler en même temps.)

3.2 Le médium Éthernet

Dans l'ancêtre du réseau Éthernet appelé *Alohanet*, les ordinateurs communiquaient par radio. Dans le réseau Éthernet original, le médium était un gros câble coaxial (comme un câble d'antenne de télévision) d'une impédance de 50 ohms, assez coûteux (de l'ordre de 30 euros le mètre) dont la longueur maximum était de l'ordre du kilomètre et dont le débit était de 10 Mégabits par seconde.

A l'heure actuelle, le câble est *simulé* par des boîtiers qu'on appelle des *hubs* (en français on dit des concentrateurs) sur lequel les ordinateurs se branchent le plus souvent avec des câbles comprenant 8 fils, munis de connecteurs dits *RJ45*. (La dénomination *rj45* employée pour ce câble est incorrecte ; voir la page http ://en.wikipedia.org/wiki/Registered_jack, qui contient la dénomination correcte, un bref historique et une photographie d'un connecteur.)

Les concentrateurs ont un certain nombre de *ports* (généralement entre 4 et 32) qui permettent chacun de brancher un ordinateur ; quand le concentrateur détecte un signal sur un des ports, il le recopie sur tous les autres ports, comme le gros câble jaune d'autrefois transportait le signal.

Il existe des boîtiers plus sophistiqués qu'on appelle des *switch* (en français, des commutateurs) qui déterminent le port sur lequel est branché l'ordinateur destinataire du message et n'émettent le message que vers ce port.

La plupart des concentrateurs et des commutateurs qu'on achète couramment à l'heure actuelle fonctionnent en 100 Mégabits par seconde. La plupart permettent encore de connecter les anciennes machines qui fonctionnent en 10 Mégabits par seconde. On le note comme 10/100 Mb/s.

3.2.1 Les câbles RJ45 pour Éthernet

Les câbles RJ45 utilisés pour connecter l'ordinateur aux concentrateurs ou aux commutateurs sont composés de quatre paires de fils enroulés l'un autour de l'autre (on les appelle des *paires torsadées*), avec dans chaque paire un fil d'une couleur unie (marron, jaune, bleu et vert) et l'autre soit uni blanc, soit n'ayant qu'un filet de la même couleur.

Pour utiliser le câble pour du 100 Mégas-bits par seconde, il doit répondre à des contraintes électriques bien définies ; on l'appelle du câble de *catégorie 5*, sur lequel on peut faire passer du 100 Mégabits par seconde, ou du *catégorie 5e* qu'on peut utiliser pour du Gigabit par seconde. Voir la page `http://en.wikipedia.org/wiki/Category\_5\_cÃćble` pour des détails.

On peut acheter les câbles RJ45 tout faits, ou bien les fabriquer soi-même avec une pince spéciale qu'on utilise pour sertir les huit fils dans le connecteur. Attention, l'ordre des couleurs est un peu important ; il permet de minimiser les interférences générées par le câble en associant les signaux qui vont bien dans chaque paire. La longueur maximum spécifiée par la norme est d'une centaine de mètres.

3.3 Les détails d'une trame Éthernet

Quand des données sont à transférer entre deux ordinateurs reliés par Éthernet, elles sont placées à l'intérieur d'une unité qu'on appelle une *trame* (*frame* en anglais) composé d'un en-tête et de données.

3.3.1 En-tête et données

L'en-tête de la trame joue un peu le rôle d'une enveloppe et les données celui de la lettre transportée par l'enveloppe.

Avant l'en-tête et les données, la carte Éthernet envoie un paquet de huit octets (64 bits) avec toujours les mêmes valeurs pour permettre aux autres ordinateurs de détecter le début d'un message (et permettre de détecter une collision, le cas échéant).

Les trames peuvent faire n'importe quelle taille entre 64 et 1518 octets (c'est à dire transporter au maximum un peu moins de 1,5 Kilo-octets de données).

Les données sont suivies d'une somme de contrôle qui permet de détecter les erreurs de transmission ; on appelle cette somme de contrôle le *FCS* comme *Frame Check Sum*.

3.3.2 La structure de l'en-tête

L'en-tête contient trois champs : l'adresse de la machine à qui la trame est destinée, l'adresse de la machine qui a envoyé la trame et finalement deux octets qui décrivent le type de données transportées par la trame.

La MAC Address

L'adresse qui permet de reconnaître les ordinateurs sur le médium porte un nom générique qui s'applique non seulement à Éthernet mais aussi aux autres technologies de réseau : on les appelle des *MAC address* (comme c'est de l'anglais, *address* s'écrit avec deux *d* et pas de *e* à la fin). *MAC* est un acronyme pour *Medium Access Control*. On parle couramment de MAC address plutôt que d'adresse Éthernet quand le contexte permet de comprendre que le médium est Éthernet (d'autant qu'il est parfois difficile de distinguer si quelqu'un parle d'adresse Éthernet ou d'adresse Internet ; les deux se prononcent presque pareil).

Les adresses Éthernet

Les adresses Éthernet sont des adresses uniques, sur 6 octets. Chaque périphérique dispose d'une adresse Éthernet qui lui a été attribuée lors de sa construction et qui ne changera jamais (l'adresse n'est fixe qu'en principe : en réalité, il y a presque toujours moyen de changer l'adresse Éthernet d'un périphérique).

L'usage est de noter les adresses Éthernet sous la forme de 6 paires de chiffres hexadécimaux séparés par des deux points. Par exemple `01:23:45:67:89` ou `ff:ee:dd:cc:bb:aa` ou `de:ad:c0:ff:ee:77` sont des adresses Éthernet correctes.

Au moment du boot, le noyau (Unix ou Linux) a recensé les périphériques présents sur la machine ; en trouvant la carte réseau, il a activé un morceau de code spécifique pour la carte (le *driver* ou *pilote*) qui a extrait l'adresse Éthernet de la carte ; pour information, cette adresse a été placée dans le journal d'activité du système et on peut probablement l'y récupérer peu après le boot, sans doute avec la commande `dmesg | grep eth0` sous Linux.

L'adresse de *broadcast*

Les trames Éthernet peuvent aussi avoir une adresse destination spéciale, dite *broadcast address*, dont les bits valent tous 1 (l'adresse est donc `ff:ff:ff:ff:ff:ff` dans la notation usuelle). Quand une trame sera envoyée à cette adresse, toutes les cartes Éthernet qui la recevront considéreront que la trame leur est adressée.

Quand une trame est broadcastée, les switchs (comme les hubs) doivent la propager vers toutes les sorties.

3.3.3 Éthernet dans les fichiers d'en-tête du noyau

La structure d'une trame Éthernet est définie dans les fichiers que le noyau inclue. Dans le fichier `/usr/include/linux/if_ether.h` (où *if* est une abbréviation pour *interface*), on trouve entre autres (légèrement édité) :

```
#define ETH_ALEN        6       /* Octets in one ethernet addr */
#define ETH_HLEN        14      /* Total octets in header.     */
#define ETH_DATA_LEN    1500    /* Max. octets in payload      */
...
#define ETH_P_PUP       0x0200  /* Xerox PUP packet            */
#define ETH_P_IP        0x0800  /* Internet Protocol packet    */
#define ETH_P_ARP       0x0806  /* Address Resolution packet   */
...
struct ethhdr {
  unsigned char   h_dest[ETH_ALEN];   /* destination eth addr */
  unsigned char   h_source[ETH_ALEN]; /* source ether addr    */
  short           h_proto;            /* packet type ID field */
};
```

On trouve d'abord des constantes (je n'ai recopié ici que les trois premières), qui définissent, comme indiqué dans les commentaires, la longueur d'une adresse Éthernet en octet, la longueur totale de l'en-tête, le nombre maximum d'octets transportés dans une trame.

Ensuite vient un autre paquet de constantes, dont je n'ai reproduit ici qu'une petite partie ; leur nom commence par `ETH_P` ; les valeurs sont celles qu'on peut trouver dans le champs de l'en-tête qui indique comment interpréter les données. J'ai conservé les constantes qui indiquent que le message transporte des messages de IP ou de ARP (que nous verrons au chapitre prochain) et celle pour PUP parce que je ne sais presque pas ce que c'est.

Finalement vient le prototype d'une structure `ethhdr` (comme *Éthernet header*, c'est à dire *en-tête Éthernet*), avec les deux adresses de destination et d'émission. et le champs `h_proto` qui indique le type de données transportées.

(Un truc mnémotechnique pour se souvenir de l'ordre des adresses : l'adresse destination est en premier de manière à permettre à une carte Éthernet de savoir, dès le sixième octet, si la trame lui est destinée ou pas ; si les adresses étaient dans l'autre sens, avec l'émetteur en tête, il serait nécessaire de stocker l'adresse émetteur puis de décider de la conserver ou pas, selon que l'adresse du destinataire correspond à l'adresse locale ou pas.)

3.3.4 D'autres normes Éthernet

Comme souvent dans les normes, la norme Éthernet possède des variantes, elles aussi normalisées. La plus répandue est la norme IEEE 802.3 qui est presque identique à celle que nous avons décrite ; cependant, dans celle-ci, l'en-tête ne contient plus les deux octets qui indiquent le type des données transportées mais un champs avec la longueur des données.

Notez que l'IEEE a une famille de normes pour la liaison physique entre deux réseaux, dont le numéro commence toujours par 802. Pour le réseau Éthernet, c'est donc le 802.3 ; le 802.5 est une norme pour Token Ring, le concurrent malheureux d'Éthernet promu par IBM ; le 802.11 normalise le réseau Wifi.

3.4 La répartition du travail entre noyau, pilote et carte réseau

Supposons qu'un ordinateur a des données à envoyer à un autre ordinateur dont il connaît déjà l'adresse Éthernet. Supposons aussi pour simplifier que les données en question font moins de 1500 octets. La séquence d'opérations sera la suivante :

Il va y avoir trois entités en jeu : le noyau du système d'exploitation à proprement parler, le pilote (driver) de la carte et finalement le logiciel câblé dans la carte. Dans les documentations techniques, la carte elle-même est souvent nommé NIC comme *Network Interface Card* (*carte d'interface réseau* en français).

3.4.1 Émission

Le noyau du système d'exploitation va transmettre au pilote les données, l'adresse destination et le type de données (qui vont dans l'en-tête) et les données à émettre.

Le pilote va ajouter l'adresse d'émission à l'en-tête et le placer avec les données à un endroit où la carte peut y accéder ; il peut s'agir d'une mémoire disponible sur la carte, ou bien d'une mémoire générale dans laquelle la carte va pouvoir repêcher des données sans intervention du processus à travers un dispositif qu'on appelle un *canal DMA* (comme *Direct Memory Access*, *accès direct à la mémoire* en français).

La carte réseau va ensuite calculer et ajouter le FCS derrière les données, puis tenter d'envoyer les données, en détectant les collisions ; sur les cartes modernes, la carte détecte les collisions et tente de retransmettre la trame en multipliant l'intervalle dans lequel le délai d'attente est tiré au hasard ; sur une vieille carte, il peut être nécessaire que le pilote demande une nouvelle émission à la carte (qui va néanmoins gérer seule l'attente après la collision).

A la fin de la transmission, la carte envoie une interruption au processeur ; le noyau reçoit l'interruption, en identifie la source et active la routine d'interruption du pilote. La routine d'interruption consulte l'état de la carte (à travers un *registre d'état*) et s'il constate que la transmission s'est bien passée, revient simplement de la routine d'interruption.

Si un nombre excessif de collisions se produit, le pilote le verra dans le registre d'état de la carte. Il le signale au noyau, qui signale au processus qui a pris l'initiative de la transmission qu'il y a eu une erreur d'écriture.

3.4.2 Réception

Quand une trame arrive, la carte compare l'adresse destination avec sa propre adresse et avec l'adresse de broadcast ; si elle est égale à l'une ou l'autre, la carte stocke la trame dans sa mémoire ou, à travers un canal de DMA à un endroit dans la mémoire centrale que le pilote a fixé, puis envoie une interruption au processeur.

Le noyau attrape l'interruption, en identifie la source et active la routine d'interruption du pilote. Le pilote découvre dans le registre d'état de la carte qu'une trame est arrivée, la passe au noyau et termine le traitement d'interruption. Le noyau fait alors rentrer la trame dans la pile de traitement des paquets reçus sur le réseau.

Il est possible de placer la carte en mode *promiscuous* ; dans ce cas, quelle que soit l'adresse de destination, chaque trame est transmise au noyau. Nous utiliserons ce mécanisme dans la suite du cours pour faire de la surveillance réseau.

3.5 A lire

J'ai récupéré sur Internet la documentation technique de deux cartes réseaux de l'entreprise 3COM (fondée par un des détenteurs du brevet initial d'Éthernet) ; ces documentations techniques s'appellent des *datasheets* (prononcer *datachiite*). Elle contiennent tout ce que les concepteurs de la carte ont pensé que les auteurs de logiciel pourraient avoir besoin de savoir pour l'utiliser.

3.5.1 La carte 3C501

La documentation de la carte 3C501 date de 1988 et contient toutes les fonctionnalités décrites dans la section précédente. Elle présente deux intérêts particuliers : d'une part la carte est relativement simple et donc la documentation n'est pas trop longue. D'autre part elle contient un programme exemple qui fait le travail d'un driver.

Après des généralités, on voit la liste des registres de la carte, au nombre de 15, sur la page 16. Le pilote peut lire ou écrire dans ces registres presque comme dans de la mémoire. (En fait la lecture et l'écriture utilisent des instructions spéciales, qu'on peut appeler depuis le C avec les fonctions `inb` et `outb` pour lire et écrire des octets, `inw` et `outw` pour les mots de 16 bits et `inl` et `outl` pour les mots de 32 bits.) Les pages suivantes sont consacrées à la description du contenu de chacun de ces registres.

La page 13 contient la description de la suite d'opérations à effectuer pour lire ou écrire des données. Les pages 14 à 18 contiennent un exemple de programme pour DOS qui permet d'utiliser la carte ; Le programme est en C ancien, avant la normalisation ANSI ; la principale différence est que le type des arguments d'une fonction est déclaré entre le nom de la fonction(arguments) et l'accolade ouvrante qui commence le corps de la fonction.

Regardons en guise d'exemple la fonction `ie4reset` ; elle possède un argument, `base`, du type short : il s'agit de l'endroit où les registres de la carte apparaissent dans la mémoire du processeur. La fonction commence par modifier l'état du processeur : bloquer le canal de DMA numéro 1 (que la carte pourrait utiliser), bloquer les interruptions que la carte pourrait envoyer. Ensuite elle écrit dans le registre CSR de la carte la valeur `IE4_RESET` qui lui demande de faire un reset, puis un 0 pour faire cesser le reset et vérifie que le reset s'est bien passé en consultant le contenu des registres d'état en émission et en réception `EDLC_XMT` et `EDLC_RCV`.

3.5.2 La carte 3C90x

La documentation de la carte 3C90x est dix fois plus longue que la précédente ; elle date pourtant de 1999. Tout y est beaucoup plus compliqué. La table d'index des registres, à elle seule, occupe trois pages (pp. 205–207).

Pour permettre de minimiser les interactions entre carte, pilote et noyau,

les trames sont placées dans des listes chaînées de la mémoire centrale ; cela permet par exemple au pilote de préparer une liste de trames à émettre puis de demander en une seule fois à la carte d'émettre ces trames.

3.6 Le WIFI

L'autre médium couramment utilisé dans les réseaux, ce sont les ondes radio à travers le Wifi. Attention, le terme *Wifi* désigne en fait un consortium industriel et le nom exact de la norme est en fait le 802.11 (c'est une norme IEEE).

Ce qui fait la particularité des connexions Wifi, c'est qu'elles utilisent les ondes radio : le médium est donc partagé entre tous et il est susceptible d'être bruité, alors que les erreurs de transmission à travers Éthernet sont exceptionnelles.

Le Wifi est normalisé sous le nom générique de IEEE 802.11, avec des variantes suivant les fréquences utilisées et le type de modulation du signal ; les variantes sont indiqués par une lettre qui suit le nom de la norme : 802.11a, 802.11b etc.

3.6.1 Le CSMA/CA

Comme Éthernet, chaque carte Wifi possède une adresse unique, qui est d'ailleurs dans le même format qu'une adresse Éthernet.

Comme pour Éthernet, l'émission d'une trame se fait en écoutant le réseau jusqu'à ce qu'il soit libre, puis en émettant une trame. Si la transmission échoue, alors il y a une collision et là aussi la carte attend pendant un intervalle de temps tiré au hasard dans un intervalle de temps qui grandit avec chaque collision : là aussi on a du CSMA.

La plupart des circuits utilisés pour la transmission des ondes radio dans les cartes Wifi ne permettent pas à la fois d'émettre et de recevoir. Pour cette raison, une carte qui émet ne peut pas détecter les collisions. Le Wifi utilise donc des accusés de réception et une méthode appelée *Collision Avoidance* (en français : *évitement de collision*).

L'accusé de réception

La transmission élémentaire d'une donnée sur le Wifi comprend deux trames : la première est celle qui contient les données ; la seconde est la réponse du récepteur dans laquelle il accuse réception de la trame pour l'émetteur (en anglais un accusé de réception se dit *acknowledge*, souvent abrégé en ACK).

Si l'émetteur ne reçoit pas l'accusé de réception juste après son émission, il considère que l'émission à échoué et le cas est traité comme une collision.

La *collision avoidance*

Pour diminuer le nombre de collisions, le protocole d'échange de données sur le Wifi est conçu pour permettre de les éviter. C'est le *Collision Avoidance*, le CA de CSMA/CA.

Chaque trame contient dans son en-tête un champs qui indique la durée (supposée) de l'échange. Les cartes qui reçoivent les trames consultent ce champs dans chaque trame reçue (même si elle ne leur est pas destinée) pour tenter de déterminer si le média est libre.

Ainsi, lors de l'émission d'une donnée, la machine émettrice placera dans l'en-tête une durée correspondant au temps nécessaire pour l'envoi de la trame + le petit délai qui sépare deux trames + la réponse du récepteur avec le ACK. Une troisième machine qui recevra cette trame attendra donc que le ACK soit transmis avant de tenter de prendre la main sur les ondes.

3.6.2 Le problème du nœud caché, les RTS et CTS

Il est facile d'imaginer une situation où on a trois machines dont deux sont incapables de communiquer directement entre elles. On place la machine A au centre. On place la machine B à coté ; la distance est courte, donc la réception est bonne ; ensuite on éloigne la machine B vers l'ouest (par exemple) : la communication entre A et B devient de plus en plus difficile ; on s'arrête juste à la distance limite à laquelle A et B peuvent encore communiquer. Maintenant on place une troisième machine C à coté de A et on l'éloigne dans l'autre sens (vers l'est dans notre exemple) de la même manière. On a maintenant une situation où A peut communiquer avec B et C mais où B et C ne reçoivent pas les trames que l'autre émet. On appelle cela un *nœud caché* (*hidden node* en anglais).

Il va de soi que dans cette situation, si B et C tentent tous les deux d'émettre vers A, ils ne seront pas en mesure de prévenir une collision (puisqu'ils ne reçoivent pas les trames de l'autre, ils ne savent pas que l'autre occupe le médium) ; au nœud A, les messages de B et C entrent en interférence et aucune trame n'est transmise correctement.

Pour résoudre ce problème, après un certain nombre d'échecs (le nombre n'est pas spécifié dans la norme ...), un nœud peut choisir de faire précéder l'émission d'une trame de donnée par une demande de permission d'émettre. Par exemple, B va envoyer une trame spécifique RTS comme *Request To Send* (*demande d'émettre* en français) qui va contenir dans le champs *durée* le temps nécessaire pour la réception d'une réponse RTS et dans un autre champs la durée d'émission de la trame de donnée et de son ACK. Si le nœud A reçoit la trame, il va répondre avec une trame CTS comme *Clear To Send* (*ok pour émettre* en franglais) qui va indiquer la durée d'émission du RTS plus celle des données et de leur ACK. Le nœud C, puisqu'il peut communiquer avec A, recevra cette trame et laissera le canal libre le temps de la transmission des données et de leur ACK par B et A

3.6.3 Les bornes, les beacons, les réseaux ad'hoc

Une borne Wifi est un ordinateur qui permet de transporter des données entre d'un coté un réseau type Éthernet et un réseau Wifi de l'autre coté.

A intervalle régulier, les bornes Wifi émettent une trame spécifique, dite *beacon* (*balise* en français) pour permettre aux ordinateurs de tenter de se connecter à la borne à travers le Wifi. Cette trame contient les caractéristiques importantes du réseau géré par la borne. Une commande comme `iwlist scanning` permet de lister les informations des trames *beacons* reçues par une carte wifi.

On peut aussi avoir des ordinateurs ordinaires qui communiquent entre eux par le Wifi, sans intervention d'une borne. On parle alors de réseau *ad'hoc*. Les ordinateurs récents d'Apple sont particulièrement pratiques pour cet usage.

3.6.4 Le BSSID, l'association, le roaming

Puisque chaque trame est transportée par les ondes et atteint tous les récepteurs qui se trouvent à portée, la trame contient un champs (appelé BSSID) qui permet de différencier les différents réseaux qui se partagent les ondes.

Pour permettre à un ordinateur d'échanger des données avec une borne, il doit d'abord s'*associer* avec la borne, en s'authentifiant et en échangeant les paramètres qui vont permettre de crypter les échanges. Cette étape s'appelle l'association.

Quand on souhaite couvrir avec un réseau Wifi une zone plus grande que celle couverte par une seule borne, il est possible de déployer plusieurs bornes pour couvrir le même BSSID ; les ordinateurs mobiles vont alors pouvoir se dissocier d'une des bornes du réseau pour s'associer avec une autre : ces opérations sont elles aussi spécifiées par des trames spécifiques et en compliquent sérieusement le format. (En revanche, les bornes sont toujours censées communiquer entre elles par un *réseau d'infrastructure*, type réseau Éthernet, qui leur permet des communications rapides et relativement fiables.)

3.6.5 La borne en guise de relai

Une des complications de ce type de communication vient d'un choix des concepteurs du protocole : la borne agit comme un *relai* entre un réseau Éthernet et le réseau sans fil. La trame transportée sur les ondes contient *trois* MAC address : celle des deux interfaces aux deux extrémités et celle de la borne intermédiaire. (On peut même avoir des trames avec quatre MAC address, quand les ondes sont utilisées pour faire le relai entre deux réseaux Éthernet.) La borne effectue la traduction entre le format de données transporté coté Éthernet et les trames Wifi.

3.6.6 Le reste

Il y a plein d'autres caractéristiques dans les trames Wifi, dont la norme traite des situations variées au niveau de la trame. Elles ne sont pas développées ici.

La norme 802.11 permet de diviser une grosse trame en plusieurs *fragments* qui sont ré-assemblés dès la réception, pour qu'une interférence éventuelle n'affecte qu'une partie de la transmission.

Elle permet d'encrypter le contenu des trames pour éviter les interceptions de messages.

Elle permet de regrouper des trames de manière à n'avoir qu'un seul accusé de réception pour un ensemble de trames.

3.7 Résumé

Les machines disposent chacune d'une adresse unique sur Éthernet. Elles transmettent sur le support quand il est libre, sans coordination. En cas de collision, elles recommencent l'émission après une petite attente. Le Wifi fonctionne à peu près pareil.

3.8 Exercices

Exercice 3.1 — (facile) Combien y a-t-il d'adresses Éthernet différentes ?

Exercice 3.2 — (facile) Quelle est la ligne qui décrit la MAC address de la carte Éthernet de votre ordinateur au moment où il boote ?

Exercice 3.3 — La carte 3c501 permettait-elle de positionner la MAC address ? Si oui de quelle façon ?

Exercice 3.4 — Comment les cartes 3c90x permettent-elles de positionner la MAC address ?

Exercice 3.5 — Trouver la datasheet d'une carte réseau récente. Combien compte-t-elle de pages ?

Exercice 3.6 — (vague, pas de corrigé) Une carte Éthernet tient-elle compte des collisions qui se sont produites lors de l'émission de la dernière trame ou de celles qui se sont produites lors de toutes les émissions récentes ? Trouver la réponse à travers le web. Donner une opinion.