Réseaux locaux et équipements actifs
Module M2101

IUT de Villetaneuse — R&T 1ère année

Laure Petrucci

19 janvier 2021

Table des matières

Chapitre 1

Réseaux locaux

Un réseau local permet de relier un ensemble de machines réparties sur une zone géographique limitée.

1.1 Couches LLC et MAC

L'IEEE[1] a établi une norme pour les couches basses (1 et 2) des réseaux. Il a en particulier divisé la couche liaison de données en *deux sous-couches* :

— *LLC* : Logical Link Control

— *MAC* : Medium Access Control

Plusieurs protocoles existent pour la couche MAC, pour des méthodes d'accès et des supports physiques différents. La couche MAC offre des fonctions à la couche LLC, et fournit les méthodes d'accès au matériel. Les normes MAC les plus classiques sont :

802.3 Carrier Sense Multiple Access/Collision Detection method CSMA/CD (Ethernet)

802.4 Token-passing bus access method

802.5 Token Ring Access Method

802.6 DQDB Access Method

802.7 Broadband LAN

802.11 Wireless LAN

La couche LLC est décrite par la norme IEEE 802.2. Elle offre à la couche réseau un service de transfert de paquets tout en masquant l'hétérogénéité des méthodes d'accès.

1.2 Accès au support

Il se peut, lorsque l'on transmet une trame sur le support physique, qu'une autre trame soit en train de se propager. D'où une risque de *collision* entre trames. Les méthodes d'accès au support visent à gérer le problème de la collision, selon deux approches principales :

— l'*approche optimiste* consiste à envoyer la trame, détecter si une collision a eu lieu, et si c'est le cas, appliquer une méthode de résolution de conflits. Cette méthode met en œuvre un accès au support par *compétition*. Il est impossible de prédire l'arrivée d'une trame : l'accès est dit *non déterministe* ou aléatoire ;

— l'*approche pessimiste* a pour principe de ne donner à chaque machine le droit exclusif d'émettre que pendant un temps limité. Une telle technique nécessite un mécanisme de négociation du droit d'émission. À un instant donné, seule une machine peut accéder au support, d'où un accès *déterministe*.

1. Institute of Electrical and Electronics Engineers

1.2.1 Token Ring (norme IEEE 802.5)

Le réseau *Token Ring* a été introduit par IBM en 1982, avec un débit de $4Mb/s$, puis a été normalisé par l'IEEE en 1985. La topologie du réseau est en anneau. Chaque station est reliée à la suivante par une liaison unidirectionnelle.

Le principe d'accès au support est régi par un système de possession de *jeton*. Le Token Ring utilise donc une approche pessimiste. Le jeton circule en permanence sur l'anneau. Lorsqu'une station souhaite émettre une trame, elle *capture le jeton* au passage, puis émet sa trame. Lorsque la trame revient à l'émetteur, soit le jeton, soit une autre trame est mis en circulation. Outre ce principe général de fonctionnement, il y a un temps limite de détention du jeton par une même machine ainsi qu'un mécanisme de priorités.

Le champ de contrôle de chaque trame à le format indiqué figure 1.1, où :

- P est le *niveau de priorité* ;
- T indique si la trame correspond à la circulation d'un *jeton* ($T = 0$) ou à des données ($T = 1$) ;
- M est un *bit de surveillance* ;
- R est un *niveau de réservation*. Il permet de gérer les priorités.

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
R			M	T	P		

FIGURE 1.1 – Champ de contrôle d'une trame Token Ring

Chaque machine M_i gère 2 variables initialisées à -1 (“pas de valeur”) :

JetonCréé mémorise la priorité du jeton créé par cette machine (s'il existe) ;

JetonCapturé mémorise la valeur du jeton capturé par cette machine (s'il existe).

Chaque trame est représentée par son triplet (P, T, R). Initialement, le jeton $(0, 0, 0)$ circule sur le réseau. L'algorithme 1.1 est utilisé lors de la réception d'une trame jeton, et l'algorithme 1.2 lors de la réception d'une trame de données.

Algorithme 1.1 : Réception d'un jeton $(P, 0, R)$ par M_i

```
début
    P_i ← priorité maximale de trame à émettre par M_i
    si P_i ≥ P alors
        /* le jeton a une priorité inférieure, donc M_i peut émettre */
        envoyer (P, 1, 0)
    sinon
        si P = JetonCréé alors
            /* M_i a créé le jeton et constate que ce n'est plus le niveau de
               priorité requis. M_i retire son jeton de la circulation */
            si R > JetonCapturé alors
                /* M_i crée un nouveau jeton de priorité R */
                envoyer (R, 0, 0)
                JetonCréé ← R
            sinon
                /* M_i remet en circulation le jeton capturé */
                envoyer (JetonCapturé, 0, R)
                JetonCapturé ← −1
                JetonCréé ← −1
            fin si
        sinon
            /* M_i retransmet le jeton tout en réservant */
            envoyer (P, 0, max(R, P_i))
        fin si
    fin si
fin
```

Algorithme 1.2 : Réception d'une trame $(P, 1, R)$ par M_i

```
début
    si M_i est l'émetteur de la trame alors
        /* la trame est retirée de la circulation */

        si R > P alors
            /* un niveau de priorité plus grand a été demandé */

            si P = JetonCréé alors
                /* M_i crée un nouveau jeton */

                envoyer (max(R, P_i), 0, 0)
                JetonCréé ← max(R, P_i)
            sinon
                /* M_i capture le jeton de niveau P */

                JetonCapturé ← P
                envoyer (max(R, P_i), 0, 0)
                JetonCréé ← max(R, P_i)
            fin si
        sinon
            /* M_i essaie d'émettre */

            si P_i ≥ P alors
                /* M_i émet sa trame */

                envoyer (P, 1, 0)
            sinon
                /* M_i transmet le jeton */

                envoyer (P, 0, max(R, P_i))
            fin si
        fin si
    sinon
        envoyer (P, 1, max(R, P_i))
    fin si
fin
```

Exemple : Soient 3 machines M_1, M_2 et M_3 souhaitant chacune émettre une trame, de priorités respectives 0, 4 et 6. Le déroulement des algorithmes est décrit dans le tableau 1.1 où dans chaque case figurent sur la première ligne le triplet (P, T, R) reçu et sur la seconde le couple (JetonCapturé, JetonCréé) après émission.

1.2.2 CSMA/CD (norme IEEE 802.3)

La méthode *CSMA/CD* (Carrier Sense Multiple Access/Collision Detection) suit une approche optimiste :

— on écoute la porteuse ;

— s'il n'y a pas de trafic sur le réseau, la trame est envoyée. Si elle entre en collision, la trame est renvoyée ultérieurement.

M_1 $P_1 = 0$	M_2 $P_2 = 4$	M_3 $P_3 = 6$
$(0,0,0)$ $(-1,-1)$	$(0,1,0)$ $(-1,-1)$	$(0,1,4)$ $(-1,-1)$
$(0,1,6)$ $(0,6)$	$(6,0,0)$ $(-1,-1)$	$(6,0,4)$ $(-1,-1)$
$(6,1,0)$ $(0,6)$	$(6,1,0)$ $(-1,-1)$	$(6,1,4)$ $(-1,-1)$
$(6,0,4)$ $(0,4)$	$(4,0,0)$ $(-1,-1)$	$(4,1,0)$ $(-1,-1)$
$(4,1,0)$ $(0,4)$	$(4,1,0)$ $(-1,-1)$	$(4,0,0)$ $(-1,-1)$
$(4,0,0)$ $(-1,-1)$	$(0,0,0)$ $(-1,-1)$	$(0,0,0)$ $(-1,-1)$

TABLE 1.1 – Exemple de fonctionnement du Token Ring

La technique doit garantir qu'une machine puisse d'une part détecter les collisions des trames qu'elle a envoyées, et d'autre part éviter qu'une collision se produise à nouveau lors d'une retransmission.

Soient les temps suivants, représentés sur la figure 1.2 :

— t_1 : début de l'émission d'une trame par la machine A ;
— $t_p = t_2 - t_1$: délai de propagation de la trame émise par A ;
— t_3 : fin de l'émission de la trame par A ;
— $t_t = t_3 - t_1$: temps de transmission de la trame envoyée par A ;
— t_4 : début de l'émission d'une trame par la machine B, plus son temps de propagation.

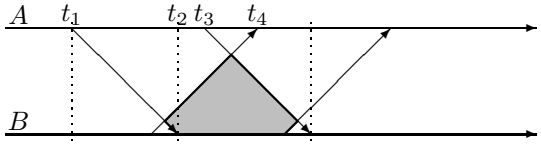


FIGURE 1.2 – Collision entre deux trames

Lorsque la machine B décide d'émettre, elle écoute la porteuse. Ne détectant aucune trame, elle envoie la sienne. Il y a ensuite collision entre les trames envoyées par les machines A et B (zone grisée dans la figure 1.2).

Propriété 1 *Pour détecter les collisions, le temps de transmission d'une trame doit être supérieur au double du temps de propagation de la trame :*

$$t_t > 2t_p$$

Preuve : Pour pouvoir détecter une collision, il suffit de garantir que $t_3 > t_4$. Pour que B puisse envoyer sa trame, il faut qu'il commence à l'émettre au plus tard en t_2. Par conséquent, $t_4 \leq t_2 + t_p$, d'où $t_3 > t_2 + t_p$. Or $t_2 = t_1 + t_p$. Donc $t_3 > t_1 + 2t_p$, c'est-à-dire $t_3 - t_1 = t_t > 2t_p$. ◊

La propriété 1 doit être satisfaite par les deux stations les plus éloignées sur le réseau.

Définition 1 (DAR) *Le* délai d'aller-retour *(DAR) est le double du temps de propagation entre les deux stations les plus éloignées du réseau. C'est une caractéristique du réseau.*

Pour que la propriété 1 de détection de collisions soit satisfaite, il faut imposer une *taille minimale* F_{min} aux trames à envoyer.

Définition 2 (bourrage) *Si la taille d'une trame à envoyer est inférieure à la taille minimale F_{min} nécessaire pour assurer la détection de collision, la trame est complétée par des données dites de* bourrage.

Il faut alors utiliser des mécanismes permettant de distinguer les données utiles des données de bourrage.

En cas de collision, la trame est retransmise ultérieurement. Le *délai de retransmission* ne doit pas toujours être le même. En effet, dans ce cas, les trames entrées en collision seraient retransmises avec un décalage identique à celui que l'on avait initialement et seraient de nouveau en collision. Pour éviter cela, un nombre *aléatoire* est utilisé dans le calcul du délai de retransmission t_r. L'*algorithme BEB* (Binary Exponential Backoff) est utilisé pour ce calcul (algorithme 1.3).

Algorithme 1.3 : Binary Exponential Backoff

```
début
    k ← nombre de collisions successives de la trame
    si k > 16 alors
        réseau déclaré hors service
    sinon
        si k ≥ 10 alors
            k ← 10
        fin si
    fin si
    N ∈ [0, 2^k − 1] (nombre aléatoire)
    t_r ← N × DAR ;
fin
```

Lorsque l'on utilise l'algorithme BEB, le délai de retransmission t_r calculé dépend à la fois du nombre de collisions de la trame et du DAR.

1.3 Ethernet (couche MAC 802.3)

Ethernet a été introduit par Xerox en 1973. En 1975, Xerox, DEC et Intel entament une normalisation. Enfin, Ethernet devient en 1985 la *norme IEEE 802.3*. Ethernet utilise la méthode d'accès CSMA/CD. L'*Ethernet II* encapsule directement le NPDU (au lieu du LLC).

1.3.1 Structure des trames Ethernet

Une trame Ethernet 802.3 encapsule une trame LLC. La structure d'une trame Ethernet est décrite dans la figure 1.3.

Préambule	SFD	DA	SA	DL/EType	données	Bourrage	FCS
7*o*	1*o*	6*o*	6*o*	2*o*	LLC/NPDU	0–46*o*	4*o*

FIGURE 1.3 – Structure d'une trame Ethernet

Le *préambule* est composé de 7 octets valant chacun 1010 1010.

Le *SFD (Start Frame Delimiter)* est l'octet 1010 1011. Il ne diffère des octets du préambule que par son dernier bit. Le SFD annonce le début de la trame Ethernet.

L'*adresse de destination DA* est l'adresse MAC de l'entité destinataire. De même, l'*adresse source SA* est l'adresse MAC de l'entité émettrice. Les adresses sont codées sur 6 octets habituellement représentés par 6 nombres hexadécimaux sur 2 chiffres, séparés par des :. Par exemple

`00:04:76:23:A1:D9` est une adresse MAC. Ces nombres sont attachés à la carte réseau de la machine : les trois premiers dépendent du constructeur de la carte et les trois derniers identifient la carte elle-même. Ces numéros sont *a priori* uniques. L'adresse `FF:FF:FF:FF:FF:FF`, composée uniquement de bits à 1 est utilisée pour envoyer une trame en *diffusion*, c'est-à-dire la transmettre à toutes les machines connectées (*broadcast*).

Le champ *DL (Data Length)* indique, dans Ethernet 802.3, la longueur, en octets, des données encapsulées, limitée à 1.500*o*. Les données étant une trame LLC, elles ont une longueur variable. Le champ DL permet donc de savoir où les données se trouvent. Dans Ethernet II, ce champ se nomme *EType* et contient le code SAP du destinataire de la trame. Pour éviter la confusion entre les trames des deux protocoles Ethernet, le numéro de SAP est toujours supérieur à 1.500. Par expemple le code hexadécimal 0800 représente IPv4.

Si la trame Ethernet, hors préambule et SFD, a une taille inférieure à 64 octets, on *rajoute des octets à* 00 pour que la taille de la trame soit 64 octets. Ces octets sont dits *de bourrage*, car ils ne servent qu'à faire du remplissage.

Enfin, la trame se termine par un *FCS (Frame Control Sequence)* permettant de détecter les erreurs éventuelles.

1.3.2 Algorithme d'émission de trame Ethernet

Chaque nœud du réseau utilise l'algorithme 1.4 pour émettre une trame. Le *délai d'aller-retour (DAR)* est fixé à $51,2\mu s$.

Algorithme 1.4 : Émission d'une trame Ethernet

```
début
    Formation de la trame
    Écoute de la porteuse pendant 9,6µs
    répéter
        si le canal est libre alors
            /* on peut émettre */

            début de l'émission
            si un conflit est détecté pendant l'émission alors
                attendre que le support se libère en brouillant la ligne
                si le nombre maximum de retransmissions est atteint alors
                    échec de la transmission
                sinon
                    attendre un délai aléatoire
                fin si
            sinon
                succès de la transmission
            fin si
        fin si
    jusqu'à fin de transmission
fin
```

Chapitre 2

Équipements d'interconnexion

2.1 Supports physiques

Ethernet est implanté en utilisant toute une panoplie de supports et d'architectures physiques. La *codification* des supports est sous la forme `D-TT-S` où :

- `D` donne le débit en Mb/s ;
- `TT` indique le type de transmission ;
- `S` renseigne sur l'architecture physique ou sur la longueur maximale d'un segment pour l'architecture bus.

Exemple : `10Base5` indique un débit de $10Mb/s$, en bande de base, sur une architecture bus dont les segments ne dépassent pas 500m.

Les *paires torsadées* sont utilisées pour `10BaseT`, `100BaseT`, `1000BaseT`. Elles assurent une qualité similaire à celle de la téléphonie. Le codage physique utilisé est le codage Manchester avec des niveaux de $2,5V$ et $-2,5V$. Le câble est relié à la machine par des *connecteurs* de type *RJ45*, qui comportent 8 pins. Seules 2 des 4 paires torsadées du câble sont utilisées : l'une pour la transmission et l'autre pour la réception. Pour les réseaux `100BaseT` et `1000BaseT`, il faut modifier le délai d'aller-retour (DAR) pour que la taille minimale d'une trame soit toujours de 64 octets. `100BaseT` utilise un codage 4b/5b, et `1000BaseT` un codage 4D-PAM5. Les 4 paires torsadées sont utilisées par `1000BaseT`.

La *fibre optique* est utilisée pour `10BaseF`, `100BaseFX` et `1000BaseSX`. Les deux derniers types de supports correspondent à une architecture en étoile.

2.2 Équipements d'interconnexion

Les équipements d'interconnexion permettent de constituer des *sous-réseaux locaux*, pour pouvoir s'affranchir des limitations de distance ou de nombre de stations connectées au réseau, répartir les stations sur des sous-réseaux différents pour diverses raisons telles que la topologie, les performances ou la sécurité, et relier des réseaux de types différents.

Les équipements d'interconnexion sont variés et se situent à différents niveaux du modèle en couches OSI, comme indiqué dans la figure 2.1.

L'équipement de base est le *répéteur* qui connecte deux segments de réseaux pour obtenir des distances plus importantes. Le répéteur amplifie le signal reçu, le reconstitue et le retransmet. Dans un réseau `10BaseT`, les stations sont connectées à un *concentrateur* (*hub*) dont la fonction

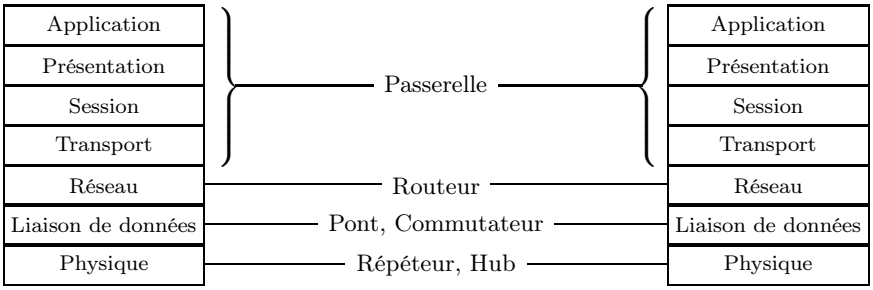


FIGURE 2.1 – Équipements d'interconnexion

est de répéter en diffusant. La distance maximale entre une station et le hub est de $100m$. On peut connecter jusqu'à 8 hubs en cascade.

2.3 Stratégies de commutation

On distingue deux types de commutateurs :

— les *commutateurs par port* : une seule station est connectée à chaque port ;

— les *commutateurs par segment* : un sous-réseau local est connecté à un port du commutateur.

Ces deux types de commutateurs permettent d'adresser respectivement une seule station ou un groupe de stations sur un port du commutateur.

De plus, les commutateurs disposent de différentes méthodes de commutation :

— la *commutation à la volée* : dès que l'adresse de destination (DA) a été analysée, la trame est envoyée à sa destination. La retransmission s'effectue pendant que l'on reçoit le reste de la trame. Le retard induit par le commutateur est alors assez faible ;

— la méthode *store and forward* : la trame est envoyée au destinataire une fois qu'elle a été complètement reçue et que le FCS a été vérifié ;

— une méthode dite *mixte* qui travaille tout d'abord à la volée, mais vérifie le FCS au passage. Si l'on détecte ainsi trop d'erreurs, la méthode à la volée est abandonnée au profit du store and forward.

2.4 Les ponts

Un *pont* permet de limiter le nombre de stations sur un segment du réseau et de dépasser les contraintes physiques telles que la longueur maximale en connectant plusieurs sous-réseaux. L'utilisation d'un pont augmente les performances locales de chaque sous-réseau en l'isolant du trafic circulant dans les sous-réseaux voisins. De plus, la sécurité est améliorée.

2.4.1 Fonctionnement d'un pont

Le pont écoute les trames circulant sur les sous-réseaux qu'il relie. Pour chaque trame il regarde l'adresse de destination. Le pont possède une *table de commutation* qui lui permet de savoir sur quel sous-réseau se trouve la station destinataire. S'il n'a pas encore l'information nécessaire, il diffuse la trame vers tous les sous-réseaux auxquels il est connecté, sauf celui dont provient la trame. Le pont dispose d'une mémoire (limitée) qui lui permet de stocker une trame jusqu'à l'avoir reçue dans son intégralité. Ensuite, il la retransmet sur un ou plusieurs sous-réseaux en fonction de l'adresse du destinataire. Cette méthode s'appelle *store and forward*.

L'utilisation d'un pont limite les risques de collisions. Les stations connectées au sous-réseau n'ont pas besoin de connaître les adresses des ports du pont. Le pont est donc qualifié de *transparent*.

2.4.2 Table de commutation

La *table de commutation* contient une ligne pour chaque station qu'elle connait. Chaque ligne contient :

— l'adresse physique (adresse MAC) de la station ;
— le numéro du port du pont auquel est connecté le sous-réseau où la station se trouve ;
— la date de création de cette ligne d'informations.

Lorsque le pont écoute une trame, il crée une nouvelle ligne dans la table de commutation, contenant :

— l'adresse MAC source ;
— le numéro du port sur lequel l'écoute a lieu.

Les lignes de la table de commutation ont une durée de vie limitée, car l'information peut devenir obsolète, par exemple si l'on déplace des machines d'un sous-réseau à un autre. Lorsque l'on crée une ligne avec une adresse MAC déjà référencée dans la table de commutation, l'ancienne ligne est remplacée par la nouvelle.

Lorsque la topologie du réseau comporte des boucles, comme dans la figure 2.2, il faut assurer le bon fonctionnement des ponts. En effet, supposons qu'une trame soit émise de A vers B sur le sous-réseau local $LAN2$. Le pont 1 comprend que la machine A se trouve sur $LAN2$. Ne sachant pas où est B, il transmet la trame sur $LAN1$. Le pont 2 croit alors que A est sur $LAN1$. La trame va être transmise de nouveau sur $LAN2$, et ainsi de suite. Pour éviter ce problème, on choisit les chemins à emprunter de manière unique, en utilisant l'*algorithme de l'arbre couvrant*.

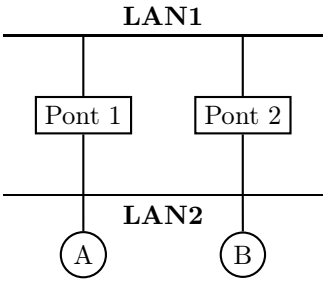


FIGURE 2.2 – Réseau comportant une boucle

2.4.3 Algorithme de l'arbre couvrant

L'*algorithme de l'arbre couvrant* (*spanning tree*) vise à choisir de manière unique un des plus courts chemins pour aller d'un sous-réseau local à un autre. Il transforme un graphe pouvant comporter des boucles en un arbre (donc sans boucle).

L'algorithme procède à l'élection d'un pont particulier, appelé *pont racine*. Pour cela, les ponts se transmettent des trames MAC spéciales, appelées *BPDU* (*Bridge Protocol Data Unit*), permettant de configurer les ponts. Une BPDU est émise par un pont sur un de ses ports. Elle est reçue et traitée par tous les autres ponts reliés au même sous-réseau local.

Définition 3 *Les termes suivants sont utilisés dans l'algorithme :*
— *Le* pont racine *est le pont ayant l'identité la plus petite.*
— *Le* port racine *est le port par lequel il faut passer pour atteindre le pont racine, éventuellement en traversant plusieurs autres ponts.*
— *Les* ports désignés *sont les autres ports utilisés dans l'arbre couvrant, permettant d'atteindre des fils. Les autres ports, n'étant donc pas utilisés, sont appelés* ports bloqués.

Propriété 2 *Un graphe connexe admet un arbre couvrant pas nécessairement unique. L'arbre couvrant contient tous les nœuds du graphe. Le chemin entre deux nœuds dans l'arbre ouvrant est unique.*

Trames BPDU

Définition 4 (BPDU) *Une BPDU comporte les informations suivantes :*
- *adresse MAC du pont racine (sur* 6 *octets) ;*
- *distance entre l'émetteur de la BPDU et le pont racine. Cette distance est exprimée en nombre de* sauts *;*
- *adresse MAC du pont émetteur de la BPDU ;*
- *numéro du port sur lequel la BPDU a été émise.*

Une trame BPDU sera notée comme un quadruplet (R, D, E, P).

Les ponts échangent des BPDUs. Ils doivent donc comparer la BPDU qu'ils ont envoyée avec celles qu'ils reçoivent pour garder la meilleure configuration. Pour cela, une comparaison des BPDU est nécessaire.

Définition 5 (comparaison de BPDUs) *Soient deux BPDUs,* $BPDU_1 = (R_1, D_1, E_1, P_1)$ *et* $BPDU_2 = (R_2, D_2, E_2, P_2)$.

$$\begin{aligned} BPDU_1 < BPDU_2 \Leftrightarrow\ & (R_1 < R_2) \vee \\ & (R_1 = R_2 \wedge D_1 < D_2) \vee \\ & (R_1 = R_2 \wedge D_1 = D_2 \wedge E_1 < E_2) \vee \\ & (R_1 = R_2 \wedge D_1 = D_2 \wedge E_1 = E_2 \wedge P_1 < P_2) \end{aligned}$$

L'ordre ainsi défini sur les BPDUs n'est autre que l'ordre classique sur les quadruplets. La meilleure configuration est celle qui correspond à la BPDU la plus faible.

Fonctionnement de l'algorithme

Au départ, chaque pont ne connait pas les autres et suppose donc qu'il est le pont racine. Il construit donc une configuration locale et la diffuse sur chacun de ses ports. Lorsqu'un pont reçoit une BPDU, il la compare à la configuration locale courante. Il en déduit la nouvelle configuration ainsi que des informations sur le pont racine et sur ses ports.

Algorithme 2.1 : arbre couvrant pour le pont i

début
  $config_i \leftarrow (i, 0, i)$ `/* Initialisation de la configuration locale */`

  répéter
    pour chaque *port désigné* p_j **faire**
      `envoyer` $(config_i, p_j)$ sur le port p_j
    fin pour chaque
    `attendre` la réception d'une $BPDU$
    `/* Choix de la meilleure BPDU */`

    soit (r, d, e, p) la meilleure BPDU reçue
    soit p_r le port sur lequel (r, d, e, p) a été reçue
    si $(r, d + 1, e) < config_i$ **alors**
      $config_i \leftarrow (r, d + 1, i)$ `/* Nouvelle configuration locale */`

      p_r devient le port racine
    fin si
    pour chaque *port* $p_j \neq p_r$ *sur lequel on a reçu une* $BPDU$ (r_j, d_j, e_j, p'_j) **faire**
      si $(r_j, d_j, e_j) < config_i$ **alors**
        p_j est bloqué
      fin si
    fin pour chaque
  jusqu'à *fin* `/* Les configurations locales ne changent plus */`

fin

Exemple : Soit la topologie de réseaux décrite dans la figure 2.3. Les numéros des ports sont indiqués à côté des sorties des ponts. Appliquons l'algorithme de l'arbre couvrant.

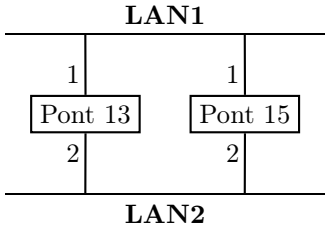


FIGURE 2.3 – Réseau comportant une boucle

Le pont 13 envoie $(13, 0, 13, 1)$ sur son port 1 et $(13, 0, 13, 2)$ sur le 2. De même, le pont 15 envoie $(15, 0, 15, 1)$ sur son port 1 et $(15, 0, 15, 2)$ sur le 2.

Pour le pont 13, la meilleure BPDU est reçue sur le port 1 : $(15, 0, 15, 1)$. La configuration locale est donc toujours $(13, 0, 13)$. Sur le port 1, la BPDU reçue est $(15, 0, 15, 1)$. Comme $(15, 0, 15) > (13, 0, 13)$, le port 1 est désigné. Sur le port 2, la BPDU reçue est $(15, 0, 15, 2)$. Comme $(15, 0, 15) > (13, 0, 13)$, le port 2 est désigné.

Pour le pont 15, la meilleure BPDU est reçue sur le port 1 : $(13, 0, 13, 1)$. La nouvelle configuration locale est donc $(13, 1, 15)$. Le port racine est le port 1. Sur le port 2, la BPDU reçue est $(13, 0, 13, 2)$. Comme $(13, 0, 13) < (13, 1, 15)$, le port 2 est bloqué.

L'arbre couvrant est présenté dans la figure 2.4.

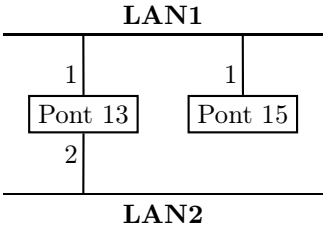


FIGURE 2.4 – Arbre couvrant du réseau de la figure 2.3

Chapitre 3

Réseaux Locaux Virtuels (VLAN)

3.1 Concepts

Les réseaux locaux imposent une structure statique. Toutefois, il est parfois préférable de regrouper des machines ayant des caractéristiques communes, même si celles-ci sont physiquement connectées à des réseaux locaux différents. Pour ce faire, on utilise le concept de *réseau local virtuel* (VLAN, Virtual Local Area Network).

Définition 6 (VLAN) *Un* Réseau Local Virtuel *(VLAN) est un* regroupement logique *de stations appartenant à des réseaux locaux existants.*

La figure 3.1 illustre un réseau comportant deux sous-réseaux locaux et trois réseaux locaux virtuels. Le réseau local LAN1 comprend les stations A, B et C, et le réseau local LAN2 les stations D, E et F. Le réseau local virtuel VLAN1 contient les stations A et B du LAN1, VLAN3 la station F du LAN2. Le réseau local virtuel VLAN2 regroupe des stations des deux réseaux locaux LAN1 et LAN2 : C, D et E.

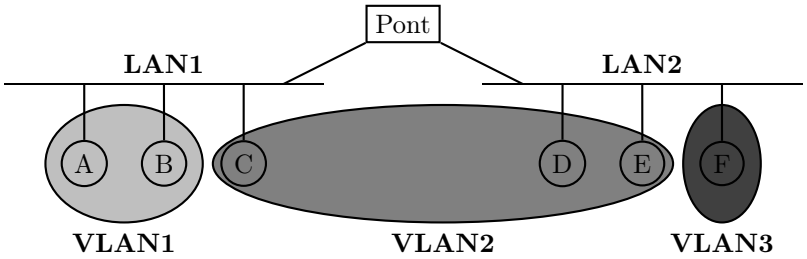


FIGURE 3.1 – LANs et VLANs

Propriété 3 *Les stations d'un VLAN communiquent comme si elles appartenaient au même segment.*

Propriété 4 *Un VLAN définit un* domaine de diffusion.

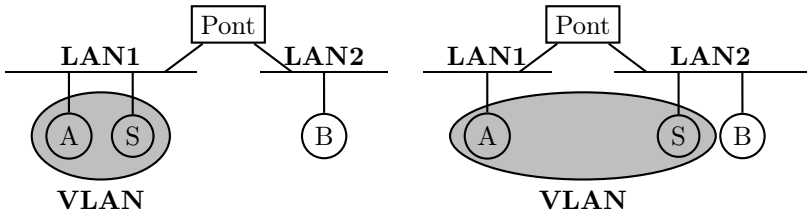
Par conséquent, les messages émis en diffusion par une station d'un VLAN ne sont reçus que par les stations de ce même VLAN.

Les VLANs offrent des avantages similaires à ceux fournis par les équipements d'interconnexion, à savoir :

— l'amélioration de la bande passante ;
— une administration du réseau plus facile ;

— une plus grande sécurité.

De plus, les VLANs permettent dans une certaine mesure de gérer aisément la mobilité. Soit le réseau configuré comme dans la figure 3.2(a). La station S peut également être connectée comme indiqué dans la figure 3.2(b) sans changement majeur.



(a) Configuration 1 (b) Configuration 2

FIGURE 3.2 – VLANs et mobilité

3.2 Mise en œuvre de VLANs

3.2.1 Équipements VLAN

Plusieurs équipements d'interconnexion permettent de créer et gérer des VLANs :

— les *HUBs intelligents* sont programmables. Ils permettent d'associer un VLAN à un sous-ensemble des ports du HUB ;

— les *commutateurs LAN* ;

— les *routeurs*.

La définition des VLANs se fait par la *programmation* des équipements d'interconnexion.

3.2.2 Construction de VLANs

Pour construire un VLAN, il faut commencer par définir un critère de regroupement des stations. Cela permet de savoir quelles stations doivent appartenir à un même VLAN.

Ensuite, la mise en œuvre de ces regroupements peut s'effectuer à différents niveaux, selon l'équipement utilisé :

— niveau 1 : on utilise le numéro de port du HUB auquel la machine est connectée ;

— niveau 2 : l'adresse MAC de la carte réseau de la station est utilisée ;

— niveau 3 : l'adresse réseau (IP) de la station permet de spécifier à quel VLAN elle appartient.

Des mécanismes plus sophistiqués tels que des *règles de filtrage* permettent une définition plus souple des VLANs.

Exemple : Soit un commutateur de niveau 1 possédant 6 ports. Une assignation des ports à 3 VLANs différents peut se programmer de la manière suivante :

```
assign port 1,3 to VLAN1
assign port 4..6 to VLAN2
assign port 2 to VLAN3
```

Le VLAN1 est alors composé des ports 1 et 3, le VLAN2 des ports 4 à 6 et le VLAN3 ne contient que le port 2.

La programmation aux niveaux 2 et 3 s'effectue de manière similaire.

De nombreux équipements disposent d'applications embarquées permettant de configurer les VLANs à partir d'une interface web.