IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

TP - Commutation Ethernet

F2B407B

2018-2019

1 Principes généraux de la commutation Ethernet

Ethernet est à la base un *bus* de données, sur câble coaxial. Si l'on ramasse tout le bus dans une boîte sur laquelle tout le monde vient se brancher, on passe à une structure en étoile. C'est la structure la plus courante aujourd'hui lorsque l'on déploie de l'Ethernet. Les boîtes en questions sont de deux types : des *concentrateurs* (hub), et des *commutateurs* (switch).

1.1 Reconnaître un switch d'un hub

Visuellement, les deux types de boîtes sont extérieurement la même allure : des prises Ethernet RJ45 [1] (de plus en plus rarement une prise coaxiale BNC) et un câble d'alimentation (plus quelques diodes qui clignotent). Les différences se trouvent au niveau fonctionnel.

Un hub travaille au niveau physique (*Level 1*) OSI, c'est à dire au niveau électrique. Pour résumer les choses grossièrement, c'est essentiellement du câblage, quelques diodes anti-retour, un peu d'électronique pour filtrer le signal et décoder les bits, et les réémettre. Notons qu'un hub sait tout de même ce qu'est un bit, il fait donc un peu plus que le simple traitement analogique du signal. La fonction d'un hub est donc de répéter sur tous ses ports tout ce qu'il reçoit. Il a donc bien un comportement basique de bus Ethernet : tous les équipements branchés dessus partagent le même *domaine de diffusion* Ethernet. Mais puisqu'il se contente de répéter les bits reçus, les équipements branchés dessus vont également partager le même *domaine de collision* de paquets Ethernet.

En revanche, le switch pallie ce problème. Un switch travail au niveau accès (*Level 2*) OSI : il sait reconnaître une trame Ethernet parmi une série de bit, et comprend ce qu'est une adresse MAC dans une trame. Sa fonctionnalité nouvelle par rapport à un hub : recopier les trames sur les *bon* ports (cela nécessite une phase d'apprentissage préalable pour connaître les *bon* ports). Reportez-vous à l'annexe A page 9 pour plus de détails.

1.2 Mise en évidence du fonctionnement du hub : répéteur

La salle d'expérimentation est organisée en trois tablées (ou banc de test). Sur chaque tablée vous trouverez divers équipements : trois PCs (équipés d'au moins une carte Ethernet), deux hub, deux switchs managables, quelques câbles Ethernet, et des routeurs que nous n'utiliserons pas aujourd'hui.

Réalisez le montage suivant les indications de la figure 1. Vous brancherez les trois PC (appelés A B et C) directement sur le hub, et rien d'autre. Assurez-vous que les PC sont parfaitement silencieux sur le réseau (lancez `wireshark`) ; si ce n'est pas le cas, demandez à un encadrant.

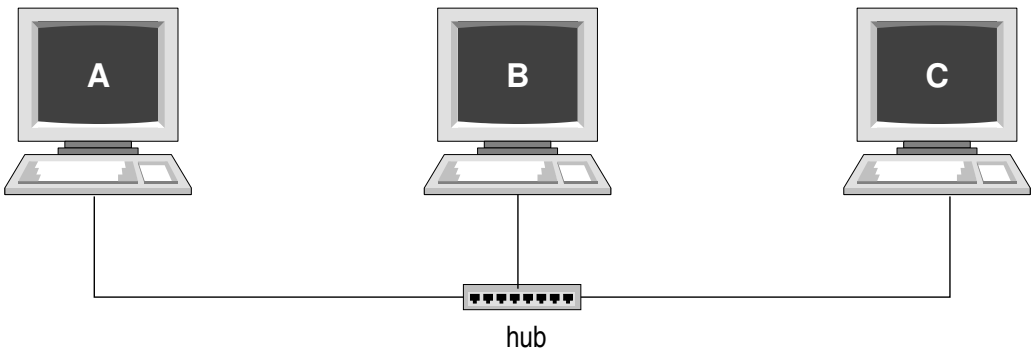

FIGURE 1 – Montage autour d'un hub

1. À quoi ressemble une prise RJ45 : `http://packetlife.net/static/cheatsheets/physical-terminations.pdf`

Note : l'outil `wireshark` [2] permet de capturer toutes les trames qui passent sur le réseau (ou du moins, celles que voient passer la carte Ethernet), et de les afficher de manière conviviale (il sait reconnaître le format de paquet d'un grand nombre de protocoles : c'est un analyseur de protocoles).

Sélectionnez `eth0`.

Dans le menu *Editer / Préférences* allez dans *Name Resolution* et retirez toutes les options.

Manipulations

- Sur chacun des 3 PC, ouvrez un terminal shell et tapez la commande `/sbin/ifconfig eth0` de manière à repérer l'adresse IP de chaque PC. (Ou bien `ip address show dev eth0`)
- Vérifiez que vous avez branché le câble ethernet sur la bonne interface : tapez `/sbin/ethtool eth0`, la dernière ligne doit dire `Link detected: yes`.
  Note : une et une seule carte Ethernet doit être branchée pour ces manipulations.
- Lancez wireshark sur les 3 PC de manière à bien voir les paquets qui transitent sur le port Ethernet du PC.
- Envoyez **un** paquet de A vers B : placez-vous sur le PC A et dans le terminal shell tapez `ping -n -c 1 adresse_de_B`
- Qu'observez-vous avec wireshark sur A B et surtout C ?
- Refaites-le une seconde fois (et une troisième, une quatrième, etc.). Conclusion ? (Éventuellement, videz le cache ARP des PC avec : `ip neigh flush all`)
- Regardez également l'état de votre cache ARP. (Commande `/usr/sbin/arp -n` ou `ip neigh show`)

1.3 Mise en évidence du fonctionnement du switch : commutateur

Nous allons utiliser un switch *manageable*, c'est à dire un peu plus sophistiqué que la boîte décrite plus haut (il est plutôt gros et possède une prise RS232). Cet équipement peut être paramétré par un administrateur. Différentes interfaces utilisateur sont possibles : en ligne de commande (CLI pour *Command Line Interface*), web, ou SNMP, et cela via une ligne série RS232 (CLI) ou via le réseau (CLI dans telnet, web, SNMP), le switch ayant également sa propre adresse IP et un véritable petit système d'exploitation avec des applications. L'interface web est bien évidement la plus conviviale et presque intuitive (quoi que...) mais souffre d'un terrible défaut commun à toutes les interfaces via le réseau : il faut que le réseau soit déjà opérationnel pour que cela marche, une hypothèse pas forcement vérifiée lorsque l'on veut justement configurer le réseau... Nous allons donc utiliser l'interface CLI via la liaison RS232. [3] À l'autre bout de la liaison RS232 nous utiliserons un terminal VT100 émulé par le PC grâce au logiciel `gtkterm` [4]. (Un véritable terminal comme le *WYSE* serait possible, et bougrement kitch, malheureusement il ne nous en reste plus qu'un et il ne marche plus très bien. D'autres logiciels émulant un terminal sont disponibles : `minicom picocom screen cu`...)

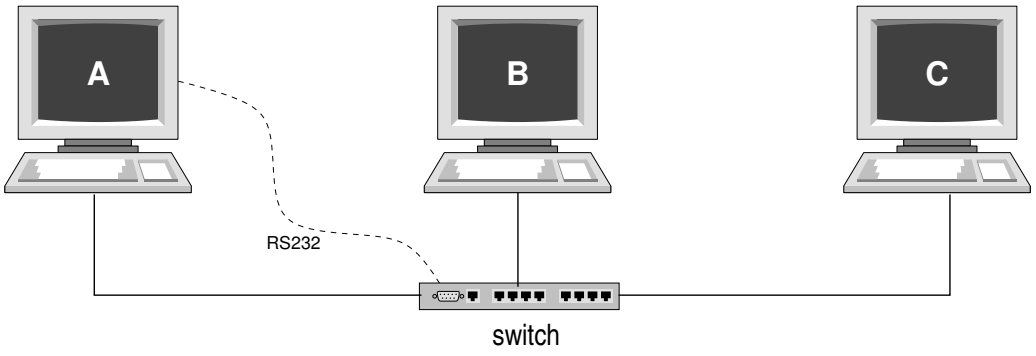

FIGURE 2 – Montage autour d'un switch

Reprenez le montage précédent en remplaçant simplement le hub par le switch (voir figure 2). La liaison RS232 du switch sera raccordée au PC A.

Note : Avec chaque switch D-Link, sont fournis un *Manuel de référence CLI*, ainsi qu'un *Manuel utilisateur* (orienté interface web). N'hésitez pas à vous y reporter.

Manipulations

- Lancez `gtkterm` sur le PC A, et tappez sur la touche `Entréee`.

2. Anciennement appelé `ethereal`.
3. Paramétrage : 9600 bauds, 8 bits de donnée, pas de bit de parité, 1 bit de stop, pas de contrôle de flux, terminal VT100
4. déjà configuré... en principe

— Le `UserName` et le `PassWord` sont vides.
— Dans la fenêtre gtkterm, tapez `show switch`. Le switch nous retourne un résumé de sa configuration de base.
— Tapez `show ports` pour connaître l'état des ports. Êtes-vous bien branché ?
— Tapez `show fdb` (*forwarding database*) pour voir les adresses MAC que le switch a déjà apprises.
— Tapez `clear fdb all` pour effacer cette table.
— Lancez wireshark sur les 3 PC.
— Éventuellement, videz la table ARP du PC A : d'abord `arp -n`, puis pour chaque adresse IP dans la table, faites `arp -d adresse_IP` (ou en plus expéditif : `ip neigh flush all`)
— Envoyez **un** paquet de A vers B (`ping -n -c 1 adresse`).
— Qu'observez-vous avec wireshark sur A B et surtout C ?
Note : pensez notamment aux requêtes ARP.
— Tapez `show fdb`. Conclusion ?
— Refaites-le une seconde fois (et une troisième, une quatrième, etc.). Conclusion ?

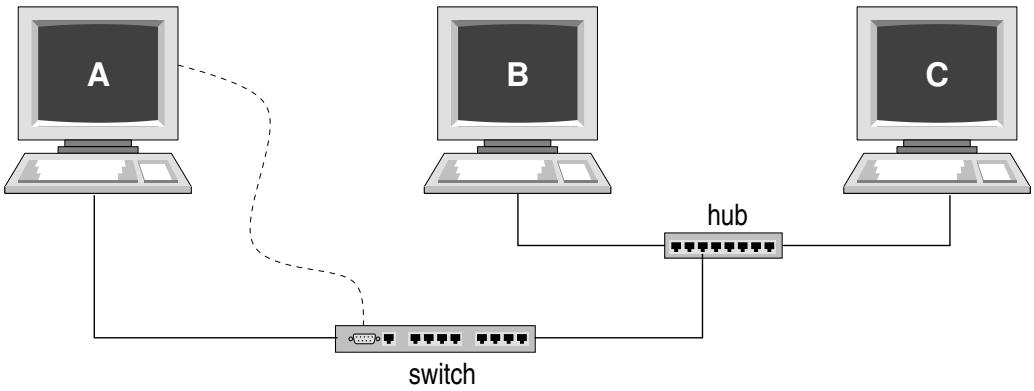

FIGURE 3 – Montage avec un hub en cascade derrière un switch

Modifiez votre montage selon les indications de la figure 3 : B et C sont sur un hub, en cascade sur le switch avec A.

Manipulations
— Tapez `clear fdb all` dans la CLI du switch.
— Envoyez **un** paquet de A vers B.
— Qu'observez-vous avec wireshark sur A B et C ?
— Envoyez **un** paquet de C vers B.
— Qu'observez-vous avec wireshark sur A B et C ?
— Tapez `show fdb`, et repérez quelles adresses MAC sont mémorisées pour chaque port.
— Est-ce que vos conclusions sont confortées ?
Note : tapez `show fdb aging_time` : les adresses sont mémorisées 5mn (300s).

2 VLAN

Le principe de base des VLANs (*Virtual Local Area Network*) consiste à couper un switch *virtuellement* en deux (ou plus) ; et ce, de manière purement interne au switch, on parle alors de *VLAN par ports*. Le second principe (un peu plus évolué que le principe de base) consiste à transporter ce découpage d'un switch à l'autre au moyen de *tags* sur des liaisons appelées *trunk* (à ne pas confondre avec le port trunking, une autre fonctionnalité des switchs).

2.1 VLAN par ports

Dans un premier temps nous allons mettre en place des VLANs simples, purement internes au switch. [5] Dans ce mode de fonctionnement, on configure le switch pour déclarer à quel VLAN appartient chacun des ports. Notons qu'initialement tous les ports appartiennent au VLAN appelé *default* (et qui porte le numéro 1). En principe, on garde ce VLAN, sauf si l'on sait exactement ce que l'on fait. Par contre, un port ne peut appartenir

5. On parle alors de VLAN *non tagués*, vous comprendrez pourquoi dans la section suivante.

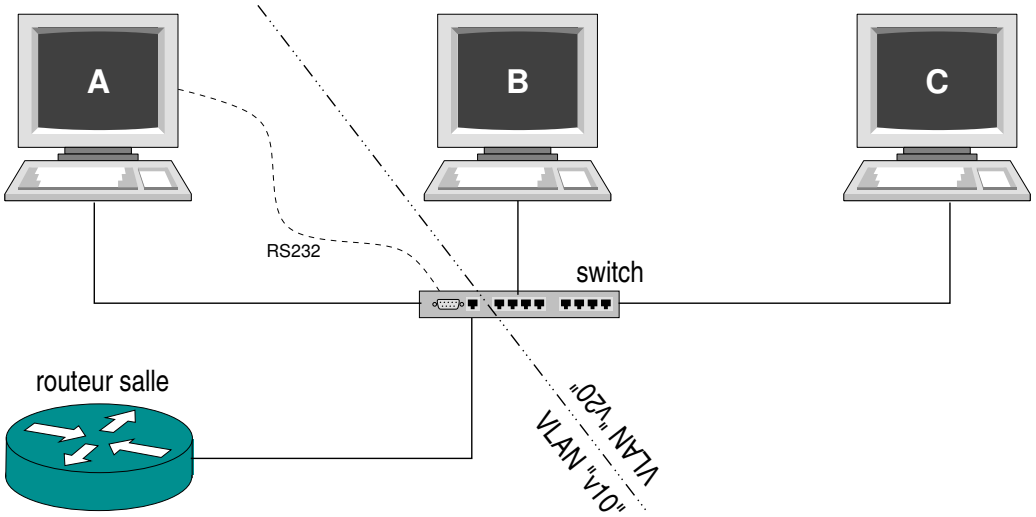

FIGURE 4 – Montage autour d'un switch - VLAN

qu'à un seul VLAN à la fois.[6] Si l'on veut placer un port dans un nouveau VLAN, il faut d'abord le retirer de l'ancien, au risque d'avoir un message *Untagged ports overlapped*. Notez également que toutes les machines branchées sur un port appartiennent de facto au VLAN de ce port.[7]

Mettez en place le montage proposé par la figure 4 : sur le switch branchez les PC A B et C, ainsi que le câble venant du routeur de la salle (un câble gris portant un numéro). Cette quatrième machine nous permettra de faire quelques tests supplémentaires.

Manipulations

- Tests avant de mettre en place les VLANs : faites des ping croisés de/vers les machines A B C et Routeur (vous trouverez son IP dans la table de routage d'un PC avec `route -n` dans un shell).
- Regardez la configuration existante : `show vlan`
- Configurez des VLAN de manière à avoir :
  - VLAN "`v10`" : A et Routeur[8]
  - VLAN "`v20`" : B et C
- Principe de configuration :
  - Création d'un nouveau VLAN non-tagué (voir manuel utilisateur) : `create vlan` *nom*
  - Ajout d'un (ou plusieurs) port(s) dans un VLAN :
    `config vlan` *nom* `add untagged` *ports*
    Pour retirer un port d'un VLAN : `config vlan` *nom* `delete` *port*
  - Afficher la configuration d'un VLAN : `show vlan` *nom*
  - Au besoin, supprimer un VLAN : `delete vlan` *nom*
- Refaites les ping croisés A B C Routeur. Conclusions ?

2.2 VLAN tagués

La norme IEEE 802.1Q[9] décrit un format de tag qui s'intercale dans l'entête des trames Ethernet et qui permet de transporter en même temps que la trame Ethernet le numéro de VLAN auquel elle appartient (ainsi qu'un niveau de priorité 802.1p utilisé parfois pour faire de la QoS dans les switchs, mais nous ne l'utiliserons pas). Ce tag permet donc de transporter plusieurs VLANs sur une même liaison Ethernet (appelée alors *trunk*). Cela permet d'avoir des VLANs répartis sur plusieurs switchs.

Dans le *Manuel utilisateur* du D-Link, page 77 vous trouverez une description du tag 802.1Q/p.

Mettez en place le montage proposé par la figure 5 : branchez A et Routeur sur un premier switch, B et C sur un second, et reliez les deux. Le second switch sera configuré depuis le PC C (via la liaison RS232 et `gtkterm`).

6. ...sauf dans le cas d'un port tagué, voir la section suivante ; dans cette première manipulation nous n'aurons que des ports non-tagués...
7. ...sauf cas des ports tagués...
8. Le câble du routeur est un câble numéroté et qui provient du double plancher.
9. `http://standards.ieee.org/getieee802/download/802.1Q-2005.pdf`

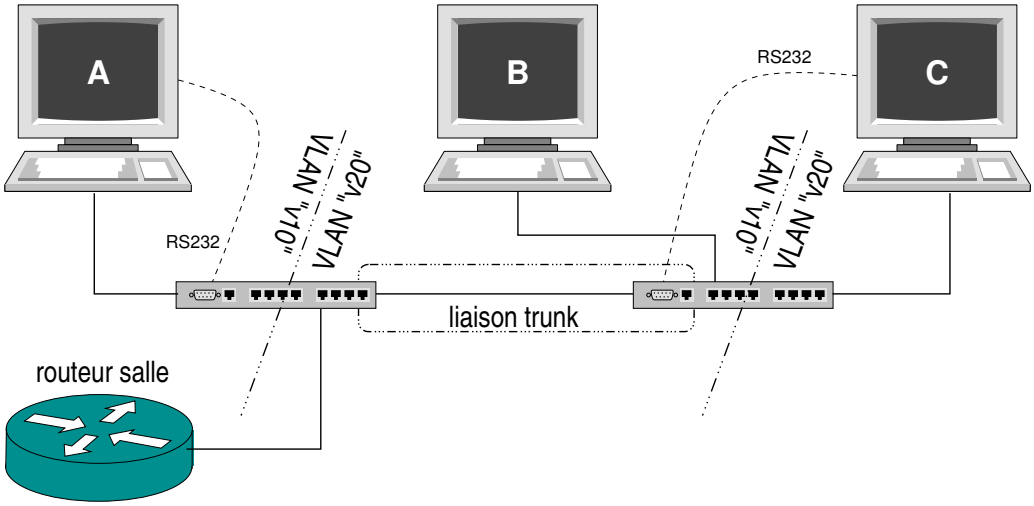

FIGURE 5 – Liaison trunk entre deux switchs

Manipulations

- Configurez les VLAN sur les deux switchs de manière à avoir :
  - VLAN "`v10`" : A et B
  - VLAN "`v20`" : Routeur et C
- Principe de configuration :
  - Créer les deux VLANs tagués "v10" et "v20" sur les deux switchs : `create vlan` *`nom`* `tag` *`numero`*
    Attention : pour chaque vlan choisissez le même numéro de tag sur les deux switchs.
  - Brancher Routeur, A, B et C sur des ports *non-tagués* de leurs VLANs respectifs selon la manipulation précédente.
  - Configurer les ports qui interconnectent (trunk) les deux switchs en *port tagués* : `config vlan` *`nom`* `add tagged` *`port`*
    Sur chaque switch, le port de la liaison trunk doit appartenir aux deux VLANs qu'il doit relayer.
- Refaites les ping croisés. Conclusions ?

Notes :

- Un port tagué peut appartenir à plusieurs VLANs tagués à la fois. Il peut également appartenir en plus à un VLAN non tagué (mais à un seul à la fois) : cela permet de préciser comment traiter les trames Ethernet qui se promènent sur ce bus sans porter de tag.
- Généralement, derrière un port tagué on branche des switchs ou parfois des routeurs, mais très rarement des machines terminales pour des utilisateurs (les administrateurs réseau n'aiment pas ça du tout).
- Les équipements branchés derrière un port tagué (switch, routeur, etc.) doivent savoir reconnaître une trame Ethernet qui comporte ce tag. Il y a un souci de compatibilité vis à vis du standard 802.1Q. Par exemple un switch bas de gamme peut ne pas savoir faire des vlan sur ses ports mais quand même *comprendre* (et relayer) une trame Ethernet tagué, ou non... On n'a bien évidemment pas cette préoccupation là avec des hub.

2.3 Contenu des tags 802.1Q

On modifie l'architecture de test précédente pour insérer un hub sur la liaison trunk et brancher le PC B sur ce hub (voir figure 6).

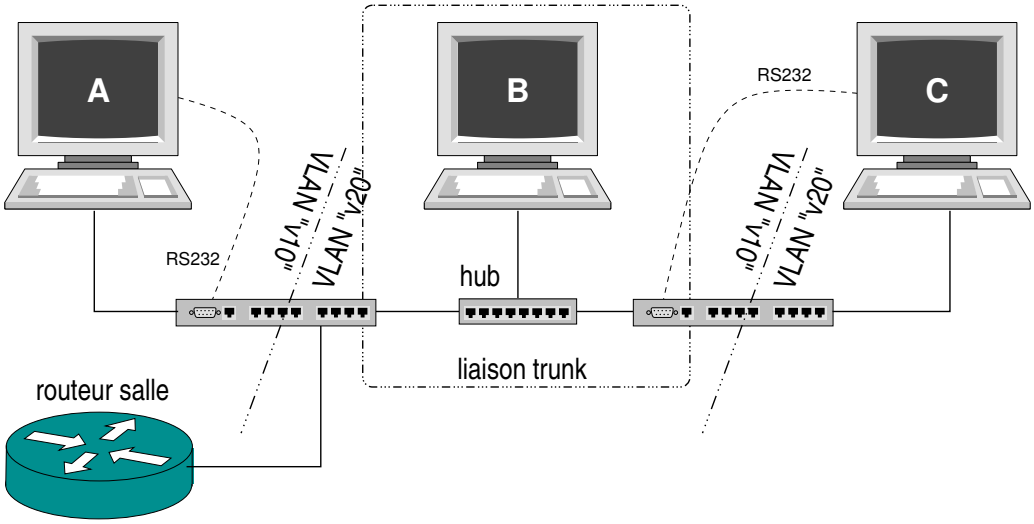

FIGURE 6 – Visualisation des tags 802.1Q

Manipulations

— Faites un ping de C vers Routeur, avec wireshark sur B. Retrouver le numéro du VLAN dans le tag avec wireshark.

Note : Le standard IEEE 802.1ad [10] *"Provider Bridges"* décrit un procédé de "double tag 802.1Q", permettant de faire des VLANs de VLANs (ou stacked VLAN, ou QinQ). (Le principe est à priori récursif mais on se limite à une profondeur de 2... pour l'instant). Par exemple, pour une offre ADSL triple play on peut avoir à la sortie du modem de l'utilisateur des VLAN séparés (pour internet, la téléphonie, la télévision). Ensuite, l'opérateur du réseau de collecte qui fournit la liaison peut encapsuler cela dans ses propres VLAN pour l'acheminer vers ses différents clients FAI.

3 Spanning Tree

L'objectif du Spanning Tree (arbre de recouvrement) est de résoudre les boucles lorsque l'on met en place des interconnections et des switchs en redondance. L'annexe A.2 page 10 illustre le type de problème introduit par des boucles sur Ethernet.

Le principe du Spanning Tree est de construire un arbre logique par dessus l'architecture physique. Cela veut dire qu'il va se choisir une *racine* et des *feuilles*. L'annexe B page 11 décrit le fonctionnement de cet algorithme, ainsi que les deux protocoles associés, le *Spanning Tree Protocol* et maintenant le *Rapid Spanning Tree Protocol*. Lisez cette annexe.

L'algorithme du Spanning Tree est un algorithme répartit, et donc par nature assez difficile à suivre. Pour l'analyser nous allons d'une part observer les messages échangés (une démarche *boite noire*), et d'autre part observer partiellement le fonctionnement interne des switchs (une démarche *boite grise*). Concrètement, autour de chaque switch nous aurons un PC avec l'analyseur de protocole wireshark et un PC avec un navigateur web connecté sur l'interface d'administration web du switch.

Nous allons mettre en place une topologie réseau qui interconnecte les trois tablées, comme indiqué sur la figure 7. Organisez vous tous ensemble, et sans chahut s'il vous plaît...

Manipulations

— Mettez en place le réseau : la figure 7, rien que la figure 7 (p.ex. attention à ne pas garder la liaison «Routeur»).

— Sur chacun des PC A, dans l'interface CLI du switch, supprimez les VLANs créés précédemment, et replacez tous les ports dans le VLAN *default*

— Sur chacun des PC B, préparez l'analyseur wirshark, mais sans lancez de capture.

— Lorsque tout est branché, si une machine a le malheur d'envoyer un paquet sur le réseau (au besoin faites un tout petit ping depuis B), la trame va tourner en boucle, être dupliquée dans chaque switch, etc. Concrètement, ça clignote de partout.

— Démarrez une capture dans wirshark (et **arrêtez la capture rapidement** avant de saturer la mémoire du PC). Vous êtes convaincu ?

10. `http://standards.ieee.org/getieee802/download/802.1ad-2005.pdf`

— Débranchez l'un des câbles de la boucle. Ça se calme rapidement. (Bravo, vous venez de faire un Spanning Tree manuel !) Ne rebranchez pas tout de suite. Au contraire, débranchez chacune des liaisons entre switch (un seul côté de la liaison suffit), le temps de paramétrer chaque switch.
— Sur chacun des PC A, dans l'interface CLI du switch, tapez `show switch`, et notez l'adresse IP et MAC du switch.
— Ensuite, lancez un navigateur web, et comme URL saisissez l'adresse IP du switch. Ciquez sur le logo `Login` qui tourne en haut. Allez dans le menu `L2 Feature` puis `Spanning Tree`.
— Dans le menu `STP Bridge Global Settings` nous retrouvons tous les paramètres présentés dans l'annexe (relisez-la une seconde fois au passage). Il y a également quelques éléments supplémentaires :
  — STP Version : nous utiliserons le RSTP. Il y a également un mode *Compatible STP* dans lequel le switch conserve le fonctionnement interne du RSTP mais utilise des messages BPDU STP. Pour des raisons pédagogiques, nous éviterons soigneusement ce fonctionnement hybride.
  — Forwarding BPDU : lorsque le Spanning Tree est désactivé sur le switch, le switch peut malgré tout forwarder les BPDU de manière quasi transparente (il se contente d'incrémenter l'âge). Mais de toutes manières nous activerons le Spanning Tree.
  — LBD : la fonction de *loopback detection* est une fonctionnalité supplémentaire apportée par D-Link, non standardisée. Nous ne l'utiliserons pas.
— Sur chacun des PC B, démarrez la capture des paquets.
— Activez le Spanning Tree sur le switch.
— Dans le menu `STP Port Settings` nous pouvons modifier le paramétrage par défaut de chaque port (reportez-vous à la page 90 du *Manuel utilisateur* du D-Link). À priori, le paramétrage par défaut convient. Repérez et notez les numéro des ports que vous utilisez.
— Maintenant rebranchez vos liaisons. Regardez défiler les messages jusqu'à une certaine stabilisation.
— Grâce aux interfaces web des switchs, repérez et notez pour chaque switch le rôle de chacun de ses ports.
— Épluchez les messages capturés, et retrouvez ces informations dans le BPDU (faites attention aux adresses MAC des switchs et de leur numéro de port). Pour vous aider, dans la barre de filtrage de wireshark tapez `stp` pour n'afficher que ce protocole.
  Qui est la racine ? Tout le monde est d'accord ?
— Perturbons le réseau : sur l'un des switchs, intervertissez deux ports. Qu'est-ce que cela change ?
— Repérez un port *designated* sur un switch. Rajoutez une liaison en parallèle de celle là (et allant sur le même hub). Qu'est-ce que cela change ? Et en parallèle d'un port *root* ? et d'un port *alternate* ?

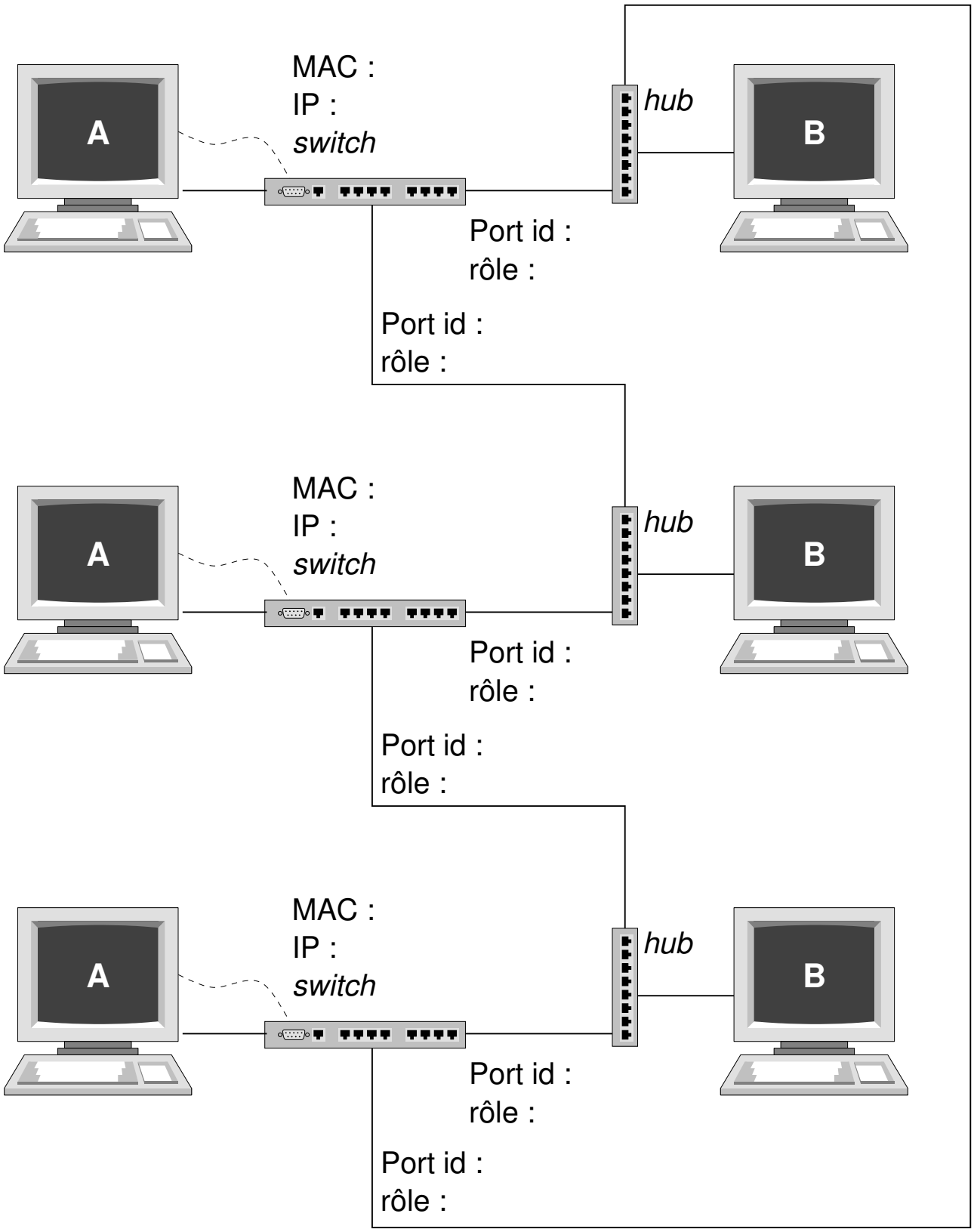

FIGURE 7 – Architecture de test du RSTP

Annexes

A Interconnexion de réseaux par ponts

Les switchs Ethernet sont des équipements maintenant courants. On les définit souvent comme des *ponts multi-ports*. Commençons donc par voir ce qu'est un *pont* (d'ailleurs, en toute rigueur le titre du standard IEEE 802.1D [11] parle bien de bridge et non de switch).

A.1 Principe des ponts

Les ponts permettent d'interconnecter des segments de réseaux locaux (LAN) de manière à ce que la communication entre les réseaux soit possible (c'est évidemment le rôle des organes d'interconnexion) mais en faisant en sorte que le trafic interne à un réseau ne vienne pas polluer celui d'un autre. Par ailleurs les collisions entre trames sur un réseau ne sont pas propagées sur un autre. On dit que les ponts permettent la segmentation du trafic d'une part et des domaines de collision d'autre part.

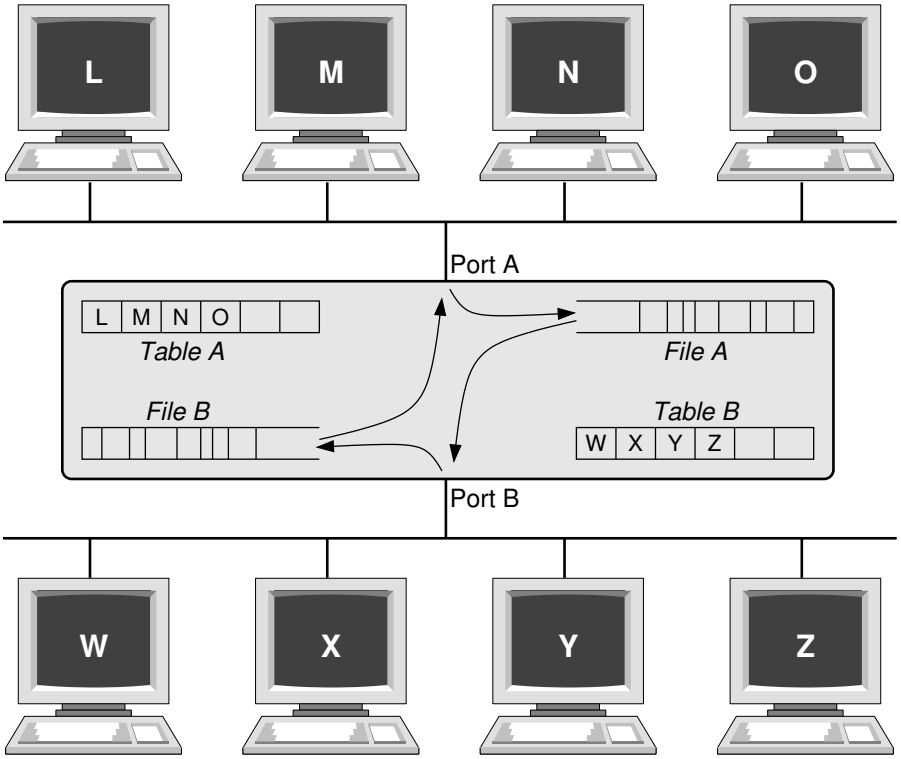

FIGURE 8 – Le principe de fonctionnement des ponts

Les ponts n'ont à priori pas besoin d'administration (on verra qu'il est néanmoins préférable de les administrer), ils apprennent automatiquement la présence des machines sur un réseau dès que celles-ci émettent une trame. Les machines sont identifiées par leur adresse MAC. Ainsi, le pont représenté figure 8 apprend, coté Port A, la présence des machines L, M, N et O sitôt que celles-ci émettent une trame. De la même manière, coté B il apprend la présence des machines W, X, Y et Z. On dit que le pont est à auto-apprentissage (*auto-learning bridge*). Une trame issue de L et à destination de O est ignorée par le pont. Une trame issue de M et à destination de W est recopiée par le pont dans une file d'attente puis émise sur le port B lorsque cela est possible. Si une machine de destination n'est pas connue du pont (elle n'a pas encore émis de trame elle-même) alors le pont relaie les trames qui lui sont destinées. Dans l'exemple ci-dessus, si la machine Z n'est pas connue du pont, une trame issue de N à destination de Z sera néanmoins relayée par le port B du pont. Du point de vue OSI, les ponts sont des organes de niveau 2 (si on accepte de placer la couche IEEE-MAC au niveau 2 OSI). Ils travaillent grâce aux adresses MAC.

Plus précisément, on peut distinguer plusieurs types de ponts en fonction de leur mode de travail :

— *Store and Forward* : comme illustré précédemment (c'est le plus courant), à chaque port est associé non seulement une table des adresses MAC joignables, mais également une file d'attente de trames Ethernet. Le fait de gérer des trames Ethernet entières permet de vérifier les codes d'erreur (RUNT). Ces files

11. `http://standards.ieee.org/getieee802/download/802.1D-2004.pdf`

introduisent malheureusement un délais, que certains équipements peuvent gérer en priorisant certaines trames selon les principes habituels de la QoS (et précisés par le standard IEEE 802.1p)

— *Cut Through* : à l'opposé n'attend pas la trame entière et la ré-émet dès que les adresses MAC sont connues. Diminue les délais de ré-émission, mais ne vérifie pas les codes d'erreur, et risque de propager des collisions.

— *Early Cut Through* : encore plus fort, n'attend même pas la réception de l'adresse MAC de la source pour vérification, et ré-émet dès que l'adresse destination est connue. On gagne ainsi le temps de 48 bits !

— *Cut Through Runt Free* : à l'inverse, attend tout de même les 64 premiers octets de la trame Ethernet (taille minimum d'une trame) pour s'assurer un minimum de l'absence de collision.

— *Adaptive Cut Through* : décompte les erreurs pour éventuellement décider de passer en *Store and Forward*.

En terme de sécurité, l'attaque naïve sur un pont/switch (qui vient immédiatement à l'esprit, n'est-ce pas ?) consiste à l'inonder d'adresses MAC différentes pour saturer ses tables et espérer qu'il passe en mode répéteur comme un hub sur tous ses ports. C'est en effet le comportement des équipements d'entrée de gamme vendu sous des appellations comme *hub commuté* ou *micro switch*. Mais en principe, face à ce genre d'attaque, les ponts/switchs se mettent en panne et envoient une alerte SNMP (par exemple) à l'attention de l'administrateur.

A.2 Redondance de ponts

Pour des raisons de fiabilité du réseau on peut être amené à placer des ponts en parallèle avec d'autres ponts pour assurer un relais en cas de panne. Malheureusement, du fait même du fonctionnement des ponts il va se produire des boucles de relayage des trames ainsi que des multiplications de trames identiques. L'exemple de la figure 9 le montre.

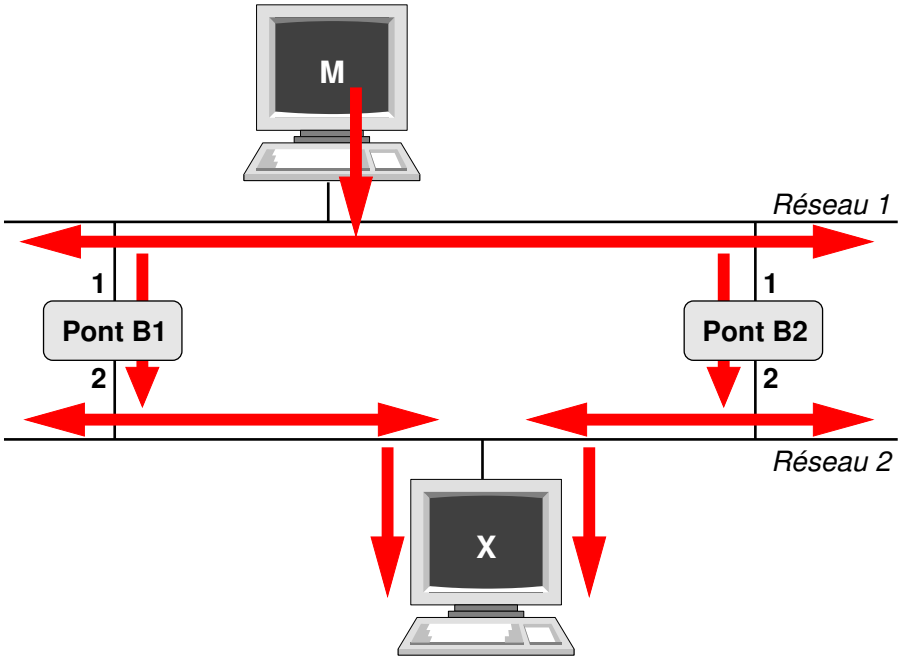

FIGURE 9 – Des ponts en parallèle forment des boucles

Si M émet une trame vers la machine X, cette dernière recevra au moins deux copies. Mais pire encore : considérons par exemple la trame relayée par B1 : elle arrive en X mais aussi en B2 port 2. Or B2, qui a détecté M du coté de son port 1 va interpréter cette trame comme nouvelle (il n'y a pas de numéros de trame, pas de moyen à ce niveau de détecter des doublons) et *penser* que M est passé du coté de son port 2. Si X a été jusque là silencieuse, les ponts ne la connaissent pas. La trame arrivant en B2, port 2, va donc être relayée port 1 et repartir sur le réseau 1. La boucle est réalisée et se poursuit de par le même raisonnement. Bien entendu, la trame d'origine est aussi relayée une fois par B2 et recopiée par B1, il existe donc une seconde boucle dans l'autre sens. Le réseau est saturé avec une seule trame. Pour éviter ce phénomène et conserver la redondance de ponts on met en oeuvre un mécanisme d'évitement de boucle appelé *Spanning Tree* (arbre de recouvrement).

Le principe des ponts, les avantages et les inconvénients qui en découlent sont entièrement applicables aux commutateurs (switchs) qui sont en fait des ponts multiports (possédant plus de deux ports).

B Algorithme du Spanning Tree

B.1 Mécanisme

Le but du mécanisme est de bâtir un réseau en arbre de manière abstraite par dessus un réseau physique affligé de vilaines boucles : il va falloir couper des liens pour résorber les cycles. Les noeuds de l'arbre sont les switchs. L'un d'entre eux sera élu comme étant la racine. Le Spanning Tree est un algorithme distribué, les switchs vont donc s'échanger des messages de configuration qui vont permettre :

- l'élection de l'un d'entre eux au statut de *switch racine*,
- de déterminer pour les autres par lequel de leurs ports atteindre au mieux le switch racine, ce port sera appelé le *port racine* (*root port*),
- de déterminer, pour chaque réseau, quel est le *meilleur* port pour desservir ce réseau, ce port est appelé *port désigné* (*designated port*),
- les ports n'étant ni *port racine* ni *port désigné* sont placés dans un état *bloqué* et ne participent plus au relayage des trames du trafic utile.

Les messages contiennent des identificateurs (dont certains sont paramétrables) permettant de décider si un switch envoie des messages *meilleurs* que d'autres. Un message est *meilleur* si les identificateurs contenus sont plus petits (philosophiquement, les identificateurs sont assimilés à des *coûts*). Au départ, chaque switch envoie son identificateur ainsi que sa propre idée sur l'identidé du switch qui pourrait être la racine. Et cette idée est que, lui-même est le switch racine. Mais chaque switch reçoit aussi les messages des autres et peut s'apercevoir qu'éventuellement certains switchs sont effectivement *meilleurs* que lui. Le switch qui ne reçoit pas de messages *meilleurs* est donc le switch racine. Les autres switch vont alors déterminer leur port racine comme étant celui qui reçoit les *meilleurs* messages. Ils vont également déterminer leurs ports désignés, c'est à dire les autres ports, ceux qui ne vont pas en direction de la racine, mais en direction des feuilles. Pour éliminer toute boucle, lorsqu'un switch reçoit sur un port désigné (un port non racine donc) un message *meilleur* venant d'ailleurs alors, ce port (qui est donc le *plus mauvais* de tous) sera mis dans un état bloquant.

B.2 Structure des messages

Les messages de configuration, appelées BPDU pour *Bridge Protocol Data Unit*, ont les caractéristiques générales suivantes :

- ils sont au format 802.3, ils contiennent donc une couche LLC (DSAP=SSAP=0x42, type de contrôle=0x3)
- ils sont émis vers l'adresse de destination multicast Ethernet `01:80:C2:00:00:00`
- la partie utile (le BPDU) contient les quatre champs suivants qui sont utilisés dans l'algorithme :
  - l'identificateur du switch racine, composé de huit octets : deux de priorité suivis par l'adresse MAC du switch
  - le coût pour atteindre la racine (2 octets de priorité)
  - l'identificateur du switch émetteur du message (adresse MAC)
  - l'identificateur du port d'émission du message (numéro)
- le BPDU contient en outre les informations suivantes :
  - deux bits pour changer / acquitter un changement d'état de la topologie ;
  - dans la version 2 du STP, deux autres bits pour proposer / acquitter une proposition de changements d'états de la topologie émanant des noeuds adjacents ;
  - dans la version 2 du STP, deux bits pour coder l'*état* du port émetteur (Learning & Forwarding), et deux bits pour coder le *rôle* (00 : Unknown, 01 : Alternate/Backup, 10 : Root, 11 : Designated) ;
  - l'âge du message (*message age*) : champ mis à 0 dans les messages émis par le switch racine. Les autres switchs rajoutent 1 par segment. Un switch relié directement à la racine (par un seul segment) émettra sur son port désigné un âge de 1. Un switch relié à la racine via deux segments, émettra un âge de 2 sur son port désigné (Rappel : les ports désignés sont ceux qui ne sont pas port racine, donc ceux qui éloignent de la racine).
  - l'âge maximal (*max age*) des informations : une valeur fixée généralement à 20s (et annoncé par la racine). Lorsque le dernier BPDU dépasse cet âge, le switch recommence un cycle où il s'annonce lui-même racine, etc. ;
  - le *hello time* : l'intervalle entre deux émissions de BPDU (par la racine), généralement fixé à 2s ;
  - le délai de relayage (*forward delay*) : le temps passé par un port dans les états *listening* et *learning* utilisés dans la version 1 du STP, état dans lesquels le switch ne peut pas relayer de trames. Ce délai est généralement de 15s. Ce champ est conservé dans la version 2 su STP pour une question de compatibilité.

Notes Les temps sont indiqués en 256ème de seconde sur deux octets (donc en pratique on se contente de la valeur sur l'octet de poids fort...).

Lorsque l'on configure le Spanning Tree sur un switch, il faut préciser (ou laisser les valeurs par défaut) le *max age*, le *hello time* et le *forward delay*, au cas où ce switch serait la racine, puisque c'est à la racine d'annoncer cela.

B.3 Exemple simple

Soit le réseau représente figure 10, composé de trois ponts (ou switchs) reliant trois réseaux.

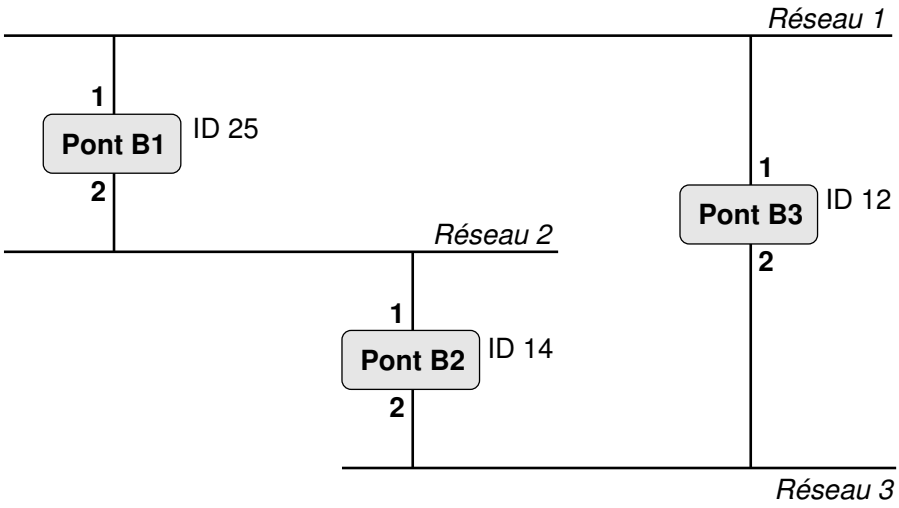

FIGURE 10 – Exemple simple de réseau switché avec redondance

Nous allons seulement considérer quelques messages :

Réseau-1

— B1 port 1 émet : 25-0-25-1 (identité racine, coût, identité switch, port), donc *je suis la racine*
— B3 port 1 émet : 12-0-12-1 donc *je suis la racine*
B1 et B3 se départagent, le message émis par B3 est *meilleur* sur le premier champ, donc meilleur ! B3 est la racine (vu de B3 et B1, ports 1).

Réseau-2

— B1 port 2 émet : 25-0-25-2
— B2 port 1 émet : 14-0-14-1
B1 et B2 se départagent, le message émis par B2 est *meilleur* ($14 < 25$). B2 peut penser qu'il est la racine. B1 sait définitivement qu'il n'est pas racine.

Réseau-3

— B3 port 2 émet : 12-0-12-2
— B2 port 2 émet : 14-0-14-2
B2 et B3 se départagent, le message émis par B3 est *meilleur*.

B3 ne reçoit pas de message *meilleur*, il continue donc à se considérer comme racine. B2 et B1 ne peuvent continuer à se considérer racine. Pour B1 : port 1 est *port racine*, pour B2 : port 2 est *port racine*.

B2 port 1 et B1 port 2 se considèrent *ports désignés* mais qu'en est-il réellement ? Voyons maintenant les messages émis par ces deux ports :

— B1 port 2 émet maintenant : 12-1-25-2 (root=12, coût=1, mon-id=25, port=2)
— B2 port 1 émet : 12-1-14-1 (root=12, coût=1, mon-id=14, port=1)

Le message émis par B2 est égal à celui de B1 sur les deux premiers champs, mais inférieur donc *meilleur* sur le troisième champ. Le message reçu par B1 port 2 est meilleur que ce qu'il peut envoyer par ce même port. Ce port ne peut donc pas rester le port désigné sur ce réseau 2, il va passer en état bloqué. Il se peut que B1 n'ait pas le temps d'émettre avant de recevoir le message de B2 qui est *meilleur*. Dans ce cas B1 ne cherche même pas à émettre.

Le port désigné sur le réseau 2 est le port 1 de B2, alors B2 est appelé *switch désigné* pour le réseau 2.

Le résultat peut être représenté par le schéma de la figure 11.

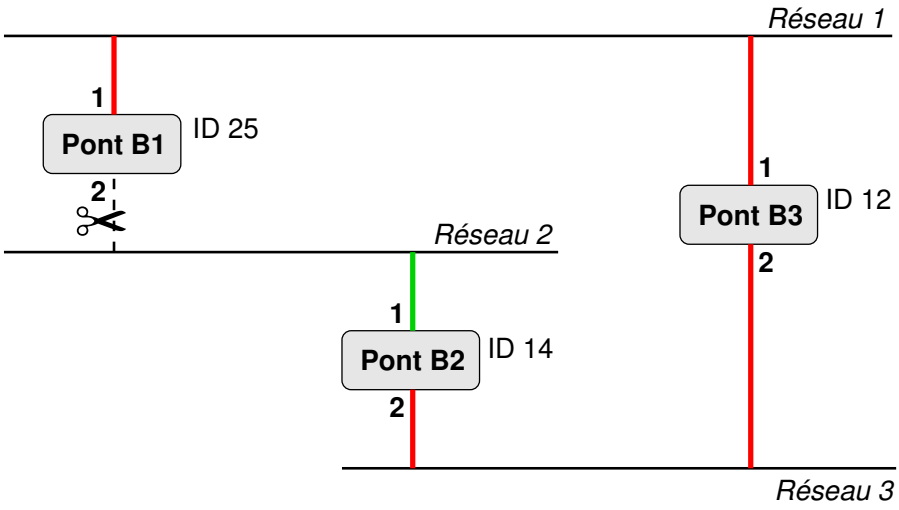

FIGURE 11 – Résultat de l'algorithme Spanning Tree

Une fois que l'algorithme a convergé les switchs continuent à émettre sur leurs ports désignés mais pas sur leurs ports racine. Le switch racine considère tous ses ports comme étant *désignés*, il continue d'émettre dessus. Ainsi dans la configuration ci dessus on verra ce qui suit :

— B3 port 1 : 12-0-12-1
— B3 port 2 : 12-0-12-2
— B2 port 1 : 12-1-14-1
— et c'est tout.

Les configurations des ports seront alors les suivantes :

— B3 : port 1 désignés, port 2 désigné
— B2 : port 1 *root port*, port 2 désigné
— B1 : port 1 *root port*, port 2 bloqué

La configuration trouvée ne semble pas optimale, pour passer de Réseau-2 à Réseau-1 il faut passer par B2, Réseau-3 puis B3, alors qu'il aurait été plus judicieux de passer par B1.

Donc, comme il faut couper les boucles quelque part, l'arbre de notre exemple est celui représenté figure 12.

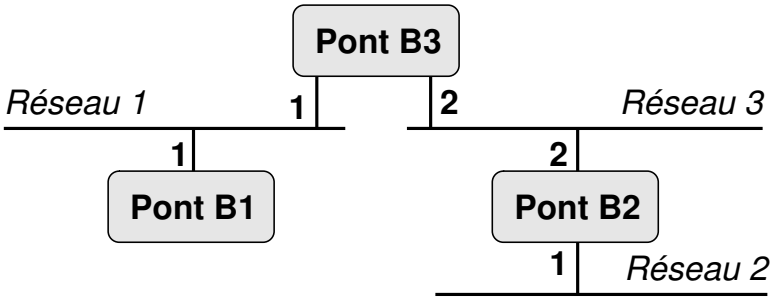

FIGURE 12 – L'arborescence de switchs et de réseaux

L'automatisation de l'algorithme avec les valeurs des identificateurs à conduit à cette arborescence.

Les paramètres d'identification sont munis d'une partie *priorité*, modifiable par administration. Il est donc possible d'agir sur la configuration souhaitée. Cependant, ici, il serait impossible d'obtenir une topologie parfaite en jouant sur la valeur des identificateurs, il y aurait toujours un chemin via deux switchs. Faites l'essai ! Retournez le problème dans tous les sens : il n'y a pas de solution optimale pour tout le monde. (Il faudrait pour cela faire autant d'arbre de recouvrement de que couple ou domaines source-destination, pas trivial...)

C Le Rapid Spanning Tree (RSTP)

L'édition de 1998 du standard IEEE 802.1d définit la première version du protocole de Spanning Tree (STP). Le mécanisme Spanning Tree (STP) d'origine présente un défaut majeur, son délai de réaction à un changement de topologie, dû essentiellement au paramètre *forward delay*. Un changement de topologie peut bloquer des chemins pendant des durées multiples de 15s par défaut.

Une amélioration est apporté par le document IEEE-802.1w-2001 (un *amendement* au sens IEEE, maintenant intégré à la version 2004 du 802.1D), définissant un nouveau Spanning Tree dit *rapide* ou *Rapid Spanning Tree Protocol* (RSTP), autrement dit, la version 2 du Spanning Tree.

Notons l'amendement 802.1s qui définit le *Multiple Spanning Tree Protocol* (MSTP), qui permet à plusieurs VLAN de participer à la même instance de STP, réduisant d'autant la charge CPU dans les switchs et permettant un équilibrage de charge. Mais oublions cela... et concentrons-nous sur le RSTP !

C.1 Version 1 - Le STP

Parlons tout d'abord de la version historique du Spanning Tree qui est souvent la mieux documentée. Elle sert également de base pour expliquer les variantes qui ont suivies. Il convient donc d'en dire quelques mots.

Dans cette version du STP, il existe cinq états possibles pour un port. Ces états sont les suivants :

désactivé (disable) : un port dans cet état est hors service, il ne participe à rien, ni à l'algorithme Spanning Tree, ni au relayage des trames. Il peut être mis dans cet état par administration ou bien parce qu'il n'est pas raccordé à un réseau (mais ce dernier point n'est pas vérifié sur tous les matériels) ;

bloqué (blocking) : le port participe au Spanning Tree mais il n'assure pas le relayage de trames (il ne participe pas à l'interconnexion) ;

écoute (listening) : état en sortie de l'état inhibé. C'est un état transitoire, avant le passage en mode presque opérationnel (état apprentissage). Le switch a reçu des messages de configuration permettant d'envisager la participation effective du port mais il est nécessaire d'attendre encore ;

apprentissage (learning) : second état transitoire pendant lequel le switch peut commencer à *apprendre* les adresses des stations accessibles via ce port ;

relayage (forwarding) : état opérationnel complet, participation au Spanning Tree, relayage des trames si besoin, apprentissage d'adresses de stations...

C.2 Version 2 - Le Rapid-STP

Cette version 2 introduit la notion de *rôle*, en plus de la notion d'*état* pour un port. Ces deux notions ne sont pas complètement orthogonales (avoir un certain rôle implique généralement d'être dans un certain état). Cependant, ces notions sont suffisamment distinctes pour êtres différenciées.

C.2.1 Les états des ports

Comme présenté dans la sous-section B.2, il y a deux bits pour décrire les états d'un port. En fait, il a trois états possibles. Nous retrouvons également ces états dans le menu de configuration du switch.

Learning Lorsque ce bit est positionné, le port
- reçoit, gère et émet des BPDU ;
- étudie et mémorise les adresses MAC sur cette liaison ;
- reçoit et répond aux requêtes de changement de topologie.

Cet état est équivalent au *learning* du STP.

Forwarding Ce bit est positionné, **en plus du bit learning**, et signifie que le port, en plus :
- retransmettent toutes les trames de données.

C'est un port complètement fonctionnel.

Discarding Lorsque les deux bits précédents sont à 0, cela signifie que le port ne participe en rien à la topologie du réseau (il y a donc peu de chances que l'on voit passer un message réseau avec ces bits à 0).
- Ce port est désactivé suite au Spanning Tree ;
- ou bien il n'y a rien de branché sur le port ;
- ou bien il est en panne ;
- ou bien le port a été désactivé par l'administrateur ;
- mais il peut éventuellement recevoir des requêtes SNMP (et y répondre) ;
- les BPDU éventuellement reçus sont analysés, mais non forwardés sur les autres ports, et ce port ne forwarde pas de BPDU.

Cet état combine les états *disable blocking* et *listening* du STP.

C.2.2 Les rôles des ports

Les ports peuvent être :

root port : le port recevant le meilleur BPDU est *root port*, en pratique c'est le port qui est le plus proche de la racine ;

designated port : c'est le *meilleur* port sur un segment de réseau donné, celui qui, pour ce segment, conduit vers la racine ;

alternate port : c'est l'inverse du port *désigné*, un port dans ce rôle (notez bien *rôle* et non *état*) reçoit des BPDU meilleurs d'un autre commutateur (notez bien le mot *autre*) ;

backup port : c'est une sorte de *alternate* mais sur le même commutateur. Un port dans ce rôle reçoit des BPDU meilleurs émis par un port du même commutateur.

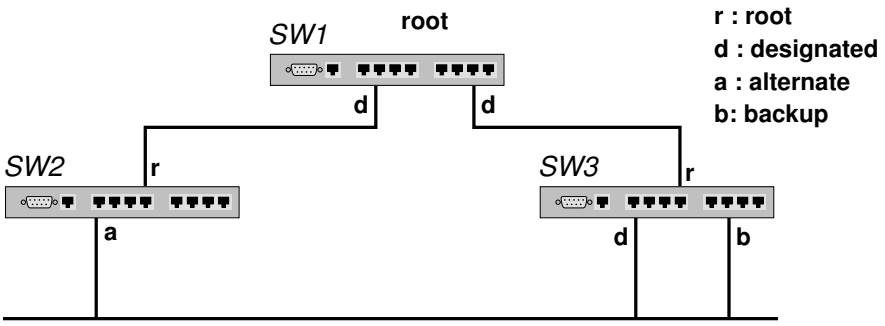

FIGURE 13 – Le rôle des ports du RSTP

La figure 13 montre un exemple de topologie, les ports du commutateur racine SW1 sont tous *designated*, un port de SW2 et un port de SW3 sont *root port*, ce sont ceux qui sont les plus proches de la racine. Un port desservant le segment inférieur est *designated*, c'est celui qui fournit le *meilleur* BPDU. Dans la figure il s'agit d'un des ports de SW3. Les autres ports sont au moins *alternate*, sauf ceux de SW3 raccordés sur le même segment, qui sont, eux, *backup*.

Les ports *alternate* et *backup* sont dans l'état *discarding*, ils sont bloqués (un célèbre constructeur utilise même le terme *blocking* pour l'état *discarding* dans ses implémentations matérielles).

Ce découpage en *état* et *rôle* permet de gérer les situations transitoires de manière plus propre. Par exemple, un port peut se retrouver dans un état *discarding* tout en étant gratifié du rôle *designated* si l'on vient juste de brancher un câble dessus, le temps que le Spanning Tree se stabilise avec cette nouvelle topologie et que les switchs de part et d'autre de cette nouvelle liaison se mettent d'accord sur le rôle des ports impliqués sur la liaison. [12]

C.2.3 Mécanismes accélérant les transitions

Différents mécanismes rendent le RSTP plus *rapide* que le STP.

Nature des ports. En fonction de ce qui est branché sur un port, celui-ci peut passer immédiatement en état *forwarding* (relayage).

— *Edge ports* : ce sont les *ports de bordure*, ceux qui sont directement reliés à des stations et non plus à des commutateurs ou des hubs.

— *Link ports* (ou *P2P ports*) : les ports reliant d'autres commutateurs peuvent passer immédiatement en mode *forwarding* s'ils sont en full-duplex, ce mode ne pouvant fonctionner qu'en point à point, on est certain qu'il n'y a que deux ports en relation. De même, les ports des liaisons en point à point (utilisant par exemple une encapsulation ppp ou HDLC) sont immédiatement mis en état *forwarding*.

Les BPDU comme principe de *keep-alive*. En STP, les ponts non-racine n'émettent un message de configuration (BPDU) que lorsqu'ils en reçoivent un sur leur port racine (principe de propagation). À contrario, en RSTP, chaque pont génère son propre BPDU tous les *hello-time* même s'il n'a rien reçu sur son port racine (en fonction des dernières informations qu'il a reçues de la racine, tout de même).

Ainsi, en RSTP, si un pont ne reçoit aucun BPDU d'un voisin pendant un laps de temps qui correspond à trois [13] fois le *hello-time*, on suppose qu'il a perdu la connexion avec ce voisin. En fait, les BPDU fournissent un mécanisme de *keep-alive* permettant la détection rapide d'un changement de topologie, et la convergence de l'algorithme. (Avec les valeurs par défaut, cela prend $3 \times 2s = 6s$.)

En STP classique, les ports passent en état *forwarding* seulement quand l'algorithme a convergé (*forwarding delay*, 15s par défaut).

12. Pour les détails, voir le mécanisme de *Proposal/Agreement* page 16.
13. C'est le `TX Hold Count (1-10)` de nos DLink

Gestion des changements de topologie. Dans les messages du STP (voir sous-section B.2), les switchs disposent de deux bits pour gérer les changements de topologie. Lorsqu'un switch détecte qu'une liaison est tombée (pas de message pendant le *forwarding delay*), il remonte cette information vers la racine en positionnant le bit *Topology Change* (TC) dans le BPDU qu'il émet sur son port racine. Lorsqu'un switch reçoit un BPDU avec le bit TC sur un port, il répond sur ce port avec un BPDU avec le bit *Topology Change Acknowledgment* (TCA) et positionne son *aging time* à la valeur du *forward delay* (pour accélérer la purge des informations), et remonte à son tour le TC vers la racine en le réémettant sur son port racine. Lorsque la racine reçoit le TC, il le redescend à tout le monde. Ainsi, en principe au bout des 15s (du aging time qui est égale au forward delay), chacun redémarre un cycle de Spanning Tree, qui prendra à nouveau 15s (du forward delay) pour se stabiliser et converger.

Pour accélérer les choses, le RSTP utilise les bits TC et TCA autrement. Lorsqu'un switch détecte qu'une liaison est tombée (pas de nouvelles au bout de 3 hello time), il l'annonce sur *tous ses ports* en positionnant TC. Lorsqu'un switch reçoit un TC, il l'acquitte avec un TCA, le réémet sur tous ses ports et purge ses informations. Ainsi, le TC inonde le réseau beaucoup plus rapidement et on redémarre un cycle de Spanning Tree plus rapidement.

Mécanisme de *Proposal/Agreement*. Dans les messages du Rapid-STP, les switchs disposent de deux bits supplémentaires pour notifier/accepter un changement de topologie.

Lorsque l'on branche une nouvelle liaison sur un port d'un swith, celui-ci le détecte et place ce port en état *discarding* mais en lui affectant le rôle *designated*, le temps de recevoir un BPDU, et envoie un BPDU sur ce port avec ces informations et en positionnant le bit *Proposal*. Il attend ensuite de recevoir un *Agreement* avant de passer en état *forwarding*.

Lorsqu'un switch reçoit sur un port un BPDU avec le bit *Proposal*, il va l'évaluer pour déterminer si le port qu'il l'a émit est meilleurs que le port sur lequel il l'a reçu, et éventuellement reconsidérer le rôle à affecter à ce port. Ensuite, il va commencer une phase de re-synchronisation de ses autres ports au regard de cette nouvelle topologie. Les ports *edge* et *discarding* (avec un rôle *alternate* ou *backup*) n'ont à priori pas besoin d'être re-synchronisés. Par contre, les autres ports (*root* ou *designates*) en ont besoin. Le switch les passe alors en état *discarding* mais en leur affectant le rôle *designated*, le temps de recevoir un BPDU dessus, et envoie un BPDU sur ces port avec ces information et en positionnant le bit *Proposal* (et attend un *Agreement* sur ces ports). Le switch peut alors notifier un *Agreement* au switch qui a émis le *Proposal* initial en lui retournant son BPDU tel quel (mais en positionnant *Agreement* à la place de *Proposal*).

Ainsi, un changement de topologie n'est considéré que sur la *branche* de l'arbre où elle impacte, des feuilles jusqu'à la racine, sans nécessairement redémarrer un cycle complet de Spanning Tree.

C.3 Valeurs standards des coûts et des délais

C.3.1 Coût standards des ports

Les coûts standards dépendent du débit nominal des ports.
Les valeurs usuelles qui étaient utilisées dans le STP (IEEE-802.1d-1998 - Table 8-5) :
— 100 pour 10 Mbps (min 50, max 600)
— 19 pour 100 Mbps (min 10, max 60)
— 4 pour 1 Gigabit (min 3, max 10)
— 2 pour 10 Gigabit (min 1, max 5)
L'édition de 2004 du 802.1D inclue l'amendement 802.1t-2001 et recommande de nouvelles valeurs (extrait) :
— 2 000 000 pour 10 Mbps
— 200 000 pour 100 Mbps
— 20 000 pour 1 Gbps
— 2 000 pour 10 Gbps

C.3.2 Délais et priorité standards et bornes recommandées

Ces recommandations sont employées dans le STP comme dans le RSTP (IEEE-802.1D-2004 - Table 17-1)
— Forward delay : 15s, min 4, max 30
— Maximum age : 20s, min 6, max 40
— Hello Time : 2s, min 1, max 10
— Priorité : 32768, min 0, max 65535
Autre recommandation du 802.1D (section 17-14) :

$$\begin{cases} 2 \times (forward\_delay - 1.0seconds) \geqslant max\_age \\ max\_age \geqslant 2 \times (hello\_time + 1.0seconds) \end{cases}$$

D Et pour aller plus loin...

Le mécanisme du Spanning Tree agit sur la topologie en ne permettant qu'un seul chemin vers le commutateur racine tout en bloquant les chemins redondants qui pourraient introduire une boucle au niveau Ethernet. C'est une approche finalement assez simple qui convient bien pour des infrastructures de la taille d'une entreprise.

Cependant, les opérateurs de réseau ont également cette préoccupation d'interconnecter de multiples liens parfois redondants, mais sur un très grand nombre de liens. Pour cela il existe des solutions comme TRILL (*Transparent Interconnection of Lots of Links*, IETF RFC-6325) ou SPB (*Shortest Path Bridging*, IEEE 802.1aq). On retrouve le même genre de mécanisme de découverte des noeuds voisins que dans le Spanning Tree, par contre au lieu de simplement désactiver une liaison (on agit sur la topologie), les noeuds vont mettre en place un genre de mécanisme de routage de trames Ethernet (on agit sur chaque trame). Pour cela les trames Ethernet sont encapsulées dans des trames d'un nouveau genre avec une entête spécifique comportant un adressage d'un nouveau genre permettant de réaliser ce nouveau type de routage...

Les solutions TRILL et SPB sont plus ou moins concurrentes, et malheureusement souffrent chacune de nombreuses "innovations/améliorations" propriétaires par les différents équipementiers du domaine, rendant ces équipements non compatibles entre eux. Le buzzword marketing à la mode dans le domaine est "Ethernet fabric"...