COMPRENDRE LE FONCTIONNEMENT DE LA COMMUTATION ETHERNET

Guillaume LEHMANN (lehmann@free.fr)

Le fonctionnement d'un hub est très simple. Lorsqu'il reçoit un signal (paquet de données ou collision), il propage le signal à toutes ses interfaces hormis celle dont provient le signal. Il ne délimite donc pas ni les domaines de collisions, ni les domaines de broadcast.
Mise à jour le : 10/04/2004, version 1.0

---

1 Pourquoi ajouter des switchs et non se satisfaire des hubs ?

Lorsqu'un réseau atteint une taille conséquente, il devient difficile d'obtenir des performances intéressantes en ne laissant à un moment donné qu'une seule station communiquer. Nous rencontrons les problèmes suivants :

Disponibilité : Dans un réseau à base de hubs, la bande-passante est partagée entre tous. Cela pose un problème de performance lorsque le réseau devient important. Aujourd'hui, les applications demandent toujours plus de bande-passante. Le réseau doit d'ailleurs de temps en temps être réarchitecturé pour pouvoir monter en puissance et apporter une bonne qualité de service aux nouvelles applications. Malheureusement, cette bande-passante n'est pas partagée équitablement entres toutes les machines, et une machine peut occuper toute la bande-passante à cause d'une défaillance ou tout simplement à cause du transfert d'un gros fichier.

Latence : C'est le temps qu'il faut à un paquet pour parvenir à sa destination. Sur un réseau à base de hubs, il est nécessaire d'attendre une opportunité de transmettre sans causer de collisions. Cela augmente fortement la latence, et d'autant plus qu'il y a de machines connectées sur le réseau.

Défaillance réseau : Dans un réseau typique, un périphérique sur un hub peut causer des problèmes à d'autres périphériques connectés à ce hub à cause d'une mauvaise configuration de la vitesse de transmission, ou à cause d'un trop grand nombre de broadcasts. Un réseau à base de hub est plus sensible à la panne qu'un réseau à base de switch.

Le switch permet de résoudre ces problèmes. Tout d'abord améliore la disponibilité et la latence en délimitant les domaines de collisions. Ainsi, chaque équipement connecté au switch, dispose de toute la bande-passante entre lui et le switch, et n'a pas à se soucier de l'occupation du média entre lui et le destinataire du message avant de transmettre un paquet. Cela permet d'utiliser le mode full-duplex entre 2 équipements (directement connectés). Enfin, le switch contient de nombreux mécanismes tels que le spanning tree ou encore l'autonégociation qui permettent d'obtenir une meilleure fiabilité. Nous aborderons plus en détail le spanning tree dans la suite de l'article.

2 Comment le switch peut-il séparer les domaines de collisions ?

Lorsqu'un paquet arrive sur un de ses ports, il ne le retransmettra que sur le port auquel est connecté (directement ou indirectement) le destinataire. Il n'ira donc pas polluer les autres segments avec ce paquet.

En ce qui concerne les signaux de collisions, le switch ne les propage pas. Pour des raisons de coûts, beaucoup de réseaux utilisent à la fois des switchs pour le cœur de réseau et des hubs pour interconnecter un groupe de stations de travail au switch.

Le fonctionnement interne d'un switch est le suivant : un paquet, fragment d'une trame, arrive dans le switch. Il est mis en mémoire tampon. L'adresse MAC du paquet est lue et comparée à la liste des adresses MAC connues par le switch, et gardée dans la table lookup. Pour l'envoi du paquet par le switch, plusieurs méthodes :

Cut through : Le switch lit l'adresse MAC dès que le paquet est détecté par le switch. Après avoir reçu les 6 octets qui permettent de remonter les informations concernant les adresses, le switch commence à renvoyer le paquet vers le segment destinataire, et cela avant que le reste de la trame ne soit entièrement arrivé dans le switch.

Store and forward : Ici le switch sauvegarde la totalité du paquet dans un buffer, vérifie les erreurs CRC ou autres problèmes, puis l'envoie sur le segment destinataire après avoir regardé les adresses MAC émettrices et destinataires. Si le paquet présente des erreurs, il est rejeté. Beaucoup de switchs combinent les méthodes cut through et store and forward. La première est alors utilisée jusqu'à ce qu'on atteigne un certain niveau d'erreur. C'est alors que le switch va utiliser la deuxième méthode. Peu de switchs utilisent seulement la méthode cut through car cette dernière ne fournit aucune correction d'erreurs.

Fragment free : Cette méthode est moins utilisée que les précédentes. Elle fonctionne comme cut through si ce n'est qu'elle stocke les 64 premiers octets du paquet avant de l'envoyer. La raison est que la plupart des erreurs et des collisions interviennent lors du temps de transmission des 64 premiers octets du paquet.

3 Fonctionnement interne du switch :

Mémoire partagée : Le switch stocke tous les paquets entrants dans une même mémoire, quel que soit le port d'arrivée ou de départ des paquets. Le paquet est ensuite envoyé par le port correspondant au nœud de destination.

Matrice : Le switch possède ici une "grille" interne avec d'un côté les ports d'entrée et de l'autre les ports de sortie. Lorsqu'un paquet est détecté dans un port d'entrée, l'adresse MAC est compare à la liste des adresses MAC connues pour ensuite trouver le port de sortie approprié. Le switch crée alors une connexion dans la grille à l'intersection des deux ports.

Architecture bus : Ici, le chemin interne de transmission (*common bus*) est partagé par tous les ports grâce à l'utilisation de TDMA. Un switch basé sur cette architecture a une mémoire dédiée pour chaque port. Un ASIC contrôle l'accès au bus interne partagé.

4 Pont transparent

La plupart des switchs Ethernet utilisent un intéressant système appelé *transparent bridging* pour créer leur table lookup d'adresses. Transparent bridging est une technologie qui permet à un switch d'apprendre la localisation des nœuds au fur et à mesure des communications, sans intervention de l'administrateur. Cela se déroule en cinq étapes :

- Apprentissage
- Inondation
- Filtrage
- Forwarding
- Vieillissement (Aging)

Le fonctionnement est le suivant :

L'architecture est montée avec au centre un switch et les segments A, B, C lui sont connectés. L'ordinateur A sur le segment A, l'ordinateur B dans le segment B et l'ordinateur C sur le segment C.

Un ordinateur A (Noeud A) dans le premier segment (Segment A) envoie des données à l'ordinateur présent sur un autre segment (Segment B).

Le switch prends ce premier paquet du nœud A. Il lit l'adresse MAC et comme il ne la connaît pas, il la sauve dans sa table lookup. Le switch sait maintenant que le nœud A se trouve sur le segment A. Cette étape est l'*apprentissage*.

Vu que le switch ne sait pas encore où se trouve le nœud B, il réagit comme un hub et envoie le paquet sur toutes ses interfaces (Segment B et C) exceptée l'interface d'entrée (Segment A). C'est l'étape d'*inondation*.

Le nœud B, comme le nœud C, reçoit le paquet et répond au nœud A. Le nœud C, voyant que le paquet ne lui est pas destiné, l'ignore/le détruit.

Le paquet du nœud B arrive au switch par le segment B. Le switch sait maintenant que le nœud B se trouve sur le segment B. Il ajoute cette information dans sa table lookup du segment B. Etant donné que le switch connaît l'adresse du nœud A, il lui envoie directement le paquet. Comme les nœuds A et B se trouvent sur des segments différents, le switch doit faire passer les paquets d'un segment à l'autre. C'est l'étape de *forwarding*.

Pour la suite des communications entre les nœuds A et B, le switch sachant sur quel segment se trouvent ces 2 nœuds, il forwarde les données sans utiliser l'étape d'inondation.

Rajoutons maintenant un nœud D présent sur le segment A, donc sur le même segment que le nœud A. Par une précédente transmission ayant pour émetteur le nœud D, le switch sait que le nœud D est sur le segment A. A envoie un paquet à D. Lorsque le paquet arrive sur le switch, ce dernier ignore le paquet et ne retransmit pas sur un autre segment. C'est l'étape de *filtrage*.

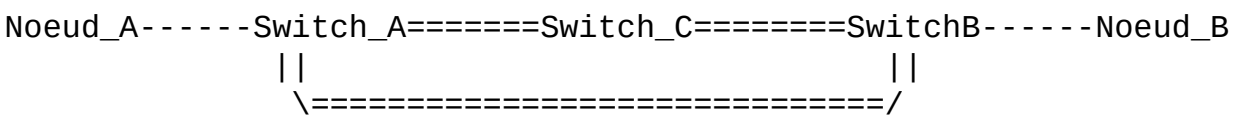
L'apprentissage et le flooding permettent de compléter la table lookup du switch. Mais la mémoire des switchs, bien que conséquente, n'est pas infinie. Pour optimiser l'utilisation de cette mémoire, il est possible de supprimer des vieilles entrées (pas forcément obsolète !). Cela s'appelle le *vieillissement (aging)*. Lorsqu'une entrée est ajouté dans la table lookup, un timer, que l'utilisateur aura configuré, est déclenché. L'entrée sera supprimée lorsque le timer de l'entrée dans la table lookup arrivera à son terme sans avoir détecté d'activité pour le nœud enregistré.

Dans un réseau entièrement switché, il ne devrait pas y avoir 2 nœuds sur un même segment. Cela évite les collisions et de devoir faire du filtrage.

5 Redondance et tempête de broadcasts

2 topologies se présentent à nous. La topologie en étoile avec un switch au centre, la topologie en anneau. Le problème de la première est que si le switch au centre tombe, les segments sont isolés les uns des autres : nous avons un point unique de défaillance. Dans une topologie en anneau, la chute d'un élément actif n'entraîne pas l'isolement d'une partie du réseau : il faut 2 points de défaillances pour isolé une partie du réseau. Pour s'assurer une meilleure résistance encore à ce problème, on peut utiliser un maillage plus complexe. Mais le fonctionnement standard du switch ne permet d'avoir des architectures avec des boucles pour la redondance.

En effet, reprenons les différentes étapes pour localiser un nœud et transmettre un paquet dans l'exemple suivant :



Si le nœud A veut transmettre un paquet à B, il envoie le paquet vers le switch A qui ne sait pas où se trouve le nœud B. Il inonde donc le réseau en envoyant le paquet vers les switchs B et C. Le switch C fais suivre ce paquet à B qui le reçoit donc une fois par le switch C, et une fois par le switch A. Comme le switch

ne sais pas où se trouve le nœud B, il envoie le paquet reçu par le switch C vers le segment où se trouve le nœud B, et vers le switch A. Et pour le paquet envoyé par le switch A, il est envoyé au nœud B et vers le switch C. Les switchs A et C renvoient alors à leur tour le paquet en broadcasts, etc. Le paquet est bien reçu par le nœud B, mais il résulte de cet envoi une tempête de broadcasts qui peut rapidement congestionner le réseau.

Pour résoudre ce problème, il est nécessaire d'utiliser le *spanning trees*.

6 Spanning Trees

Pour prévenir les tempêtes de broadcast et autres effets de bords de boucles sur le réseau, Digital Equipment Coporation créa le *spanning-tree protocol (STP)* qui fut ensuite standardisé par l'IEEE (Institute of Electrical and Electronic Engineers) comme norme *802.1d*. Le STP utilise le *spanning-tree algorithm (STA)* qui prend en compte le fait qu'un switch puisse avoir plus d'un seul chemin pour atteindre un destinataire. Il détermine alors le chemin optimal et bloque l'utilisation des autres chemins. Cependant, il garde en mémoire les autres chemins au cas où le premier serait défaillant.

Nous allons maintenant voir plus en détail le fonctionnement du STP. Cela rappelle le RIP sur quelques points . . .

A chaque switch est assigné un ensemble d'ID :

- Un ID pour le switch lui-même : le *bridge ID (BID)*, d'une longueur de 8 octets, est composé de la priorité du switch (2 octets) et d'une des adresse MAC du switch (sur 6 octets).
- Un ID pour chaque port du switch : le *port ID*, d'une longueur de 16 bits, est composé de 6 bits de priorité et du numéro de port sur 10 bits.
- Un coût de chemin (*path cost*) est donné à chaque port. En accord avec la spécification originale, la référence est 1Gbit/s. On divise cette valeur par la bande-passante pour obtenir le coût du chemin. Ci-après quelques exemples de valeurs. Au vu de l'augmentation des bandes passantes disponible sur les réseaux locaux, au-delà du gigabit maintenant, le standard a légèrement été modifié pour 1Gbits et 10Gbits.

  Bande passante => Coût STP
  4Mbits => 250
  10Mbits => 100
  100Mbits => 19
  1Gbits => 4
  10Gbits => 2

  L'administrateur peut aussi fixer les coûts de chemin comme bon lui semble.

Chaque switch lance une découverte du réseau pour choisir le meilleur à utiliser pour contacter les différents segments. Cette information est partagée entre tous les switchs grâce à des trames réseaux spécifiques. Ce sont les *bridge protocol data units (BPDU)*, constituées de la façon suivante :

Root BID : C'est le BID de l'actuel switch root.

Coût du chemin vers le switch root : Il détermine à quelle distance (au sens réseau) se trouve le switch root par rapport au switch qui a envoyé la trame. Si la trame passe par 2 segments de 100Mbits puis par un segment de 10Mbits avant d'arriver au segment connecté au switch root, le coût sera de 19+19+100=138. Le segment rattaché au switch root a un coût nul.

BID de l'émetteur : BID du switch ayant émis initialement la BPDU.

ID du port : c'est le port utilisé par lequel le switch qui a envoyé initialement la BPDU.

Tous ces switchs envoient régulièrement des BPDU entre eux, pour recalculer les meilleurs chemins. Lorsqu'un switch reçoit une BPDU (depuis un autre switch) qui propose un meilleur chemin que celle qu'il

est en train d'envoyer pour le même chemin, il arrête son broadcast. A la place, il stocke la BPDU de l'autre switch comme référence et la renvoie en broadcast aux autres sous-segments, plus éloignés encore du switch root.

7 Qu'est-ce que le switch root ?

Le switch root est le switch qui sert de référence dans le calcul des coûts des chemins. Il ne doit donc y avoir qu'un seul, mais en cas de défaillance de celui-ci, un autre prendre sa place.

Le switch root est choisi après échange de BPDU entre les switchs. Initialement, chaque switch se considère comme un switch root. D'ailleurs, même si un switch root est défini sur un réseau, si un nouveau switch est connecté, il se présentera comme un switch root : il envoie dans la BPDU son propre BID dans le champ BID root. Comment s'effectue alors le choix du switch root puisqu'il ne peut y en avoir qu'un seul ? Lorsqu'un switch reçoit la BPDU avec ce nouveau BID root, il compare cette valeur avec celle qu'il avait déjà en mémoire. Si le nouveau BID root a une valeur moins élevée, il prendra ce nouveau BID root. Ensuite, le switch qui se considère comme root et qui ne l'est pas en prendra connaissance en recevant des BPDU où le BID root n'est pas le sien (et est moins élevé). Il met donc à jour sa table en conséquence. Ainsi, le switch ayant le BID le moins élevé sera reconnu par les autres switchs comme switch root.

Les autres switchs calculent alors par quel port ils peuvent contacter le switch root en passant par le chemin le plus court. Ce port là est alors appelé *root port*, et chaque switch en définit un (sauf le switch root biensûr).

Ensuite les switchs déterminent un unique port, le *port désigné*, par lequel les données seront reçues et émises sur un segment, et donc quel switch possèdera ce port pour un segment donné. Ainsi, chaque segment à un seul et unique port désigné, et donc une seule et unique porte de communication avec le reste du réseau, ce qui empêche la présence de boucle logique. Comme l'on peut s'en douter, le port désigné est choisi pour son faible coût de chemin par rapport aux autres chemins possibles pour y accéder. Si plusieurs chemins ont le même coût le plus bas, alors c'est le switch avec le BID le plus faible qui est choisi. Etant donné que tous les segments qui sont connectés au switch root ont un chemin nul, tous les ports de ce switch sont des ports désignés. En ce qui concerne les autres switchs, le coût des chemins sont comparés pour un segment donné.

Une fois choisi le port désigné pour un segment donné, tous les autres ports connectés à ce même segment deviennent des *ports non-désignés*. Ils bloqueront tout trafic voulant passer par ces ports.

Chaque switch possède une table de BPDU qui est constamment mise à jour. Le réseau est maintenant configuré avec un unique spanning tree. Le switch root sert de tronc, et tous les autres switchs de branches. Tous les switchs communiquent avec le switch root par leurs ports root, et au travers de chaque port désigné pour chaque segment traversé. Donc l'arbre ne présente pas de boucle.

Dans le cas où le switch root soit défaillant, STP permet aux autres switchs de reconfigurer le réseau pour définir une nouvelle architecture logique autour d'un nouveau switch root élu. Ce processus permet à une entreprise d'obtenir un réseau complexe mais tolérant aux pannes et avec une maintenance aisée.

8 Routage et commutation de niveau 3

La plupart des switchs opèrent la commutation au niveau 2 du modèle OSI. Cependant, certains switchs embarquent aussi des fonctionnalités de niveau 3. Comme nous allons le voir, la distinction entre la *commutation de niveau 3* (switchs de niveau 3) et le *routage* (routeurs) est très fine.

Lorsqu'un routeur reçoit un paquet, il regarde à la couche 3 les adresses source et destination afin de déterminer le meilleur chemin entre ces 2 points. Un switch standard, donc de niveau 2, s'appuie sur les adresses MAC émettrices et destinataires.

La différence fondamentatle entre un routeur et un switch de niveau 3 est que ce dernier embarque une couche matérielle optimisée pour passer les données aussi vite qu'au niveau 2. Mais la décision vers où transmettre le paquet se prend au niveau 3. La commutation étant plus rapide que le routage (car c'est matériel et non logiciel), les switchs de niveau 3 sont plus rapides que les routeurs. Cependant, les switchs, qu'ils soient de niveau 2 ou 3, ne peuvent être utilisés que sur des LAN.

La conception interne des switchs de niveau 3 est similaire à celle des routeurs. Les 2 utilisent un protocole de routage et une table de routage pour déterminer le meilleur chemin. Cependant, un switch de niveau 3 a la capacité de *reprogrammer* la couche matérielle dynamiquement avec les informations de routage de niveau 3 courrantes. C'est cela qui lui permet de traiter plus rapidement les paquets.

Actuellement dans les switchs de niveau 3, les informations reçues des protocoles de routage sont utilisées pour mettre à jour les tables de cache de la couche matérielle.

9 VLANs

Au vu de la complexité et de la taille grandissantes des LANs, de nombreuses entreprises se sont tournées vers les réseaux locaux virtuels (*virtual local area networks* : VLANs). Ainsi, il est possible de rationaliser l'architecture réseaux locale, tout en restant au niveau 2 du modèle OSI. Grosso modo, le VLAN est un ensemble logique de n ?ud qui peuvent communiquer ensembles, dans un même domaine de broadcast. 2 switchs, même physiquement connectés, ne pourront pas communiquer entre eux s'ils font parti de VLANs distincts, comme s'ils n'étaient pas physiquement connectés. Idem pour les PC se trouvant sur chacun des VLANs.

Bien que l'on ne fasse pas du routage, le switch sur lequel est implémenté un VLAN, sépare comme un routeur, les domaines de broadcast. Si le switch fait parti de plusieurs VLAN, alors il est une frontière entre ces différents domaines de broadcast.

Cependant, il est impossible pour un switch de faire transiter des paquets d'un VLAN à l'autre. Pour cela, il faut faire appel à un routeur, ou a un switch disposant de fonctionnalités de niveau 3.

Voici quelques cas de figure où il est intéressant d'utiliser les VLANs :

Sécurité : Pour séparer des systèmes sensibles ou hébergeants de données sensibles, du reste du réseau. Ainsi, ces systèmes seront protégés des écoutes passives sur le réseau.

Projets/Applications spécifiques : Un projet ou une application peut nécessiter de travailler sur un réseau spécialisé, où certains n ?uds doivent communiquer entre eux, et d'autres non. Avec les VLANs, il est possible de réarchitecturer le réseau au niveau logique, sans toucher au réseau physique.

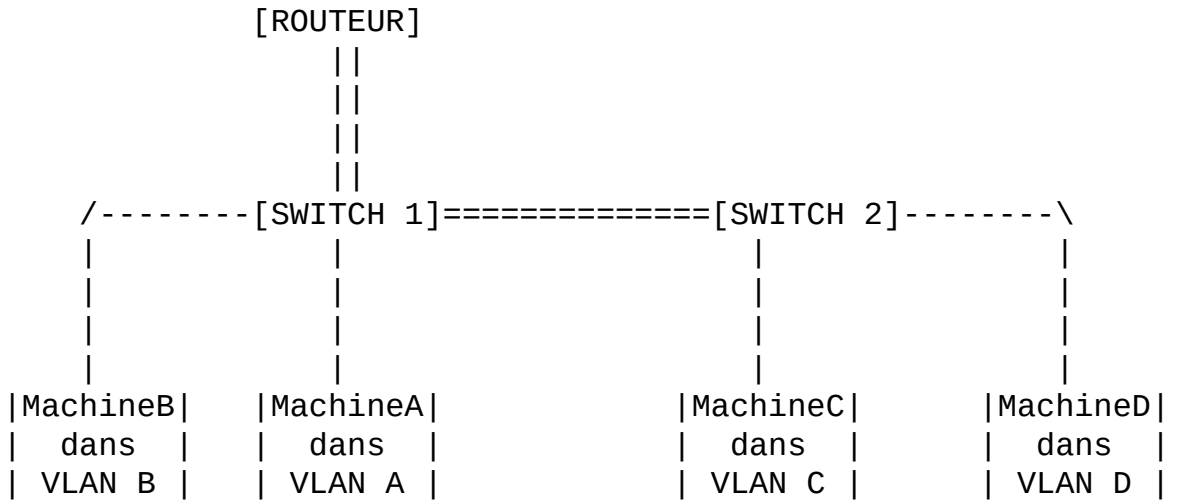
Performance/Bande-passante : En délimitant les domaines de broadcasts, et en réarchitecturant le réseau logique, on peut gérer de façon plus fine la bande-passante allouée aux utilisateurs, et donc améliorer les performances.

Profil des utilisateurs différents : Dans une entreprise où cohabitent des utilisateurs gourmands en bande-passante (ingénierie, multimédia) et des utilisateurs ayant une consommation plus modeste (managers, commerciaux), la mise en place de VLAN va permettre de répartir les ressources aux différents utilisateurs/services, et les séparer pour que les services consommant beaucoup de bande-passante ne viennent pas empiéter sur le réseau des autres services de l'entreprise, et inversement.

La création de VLAN est aisée. Il suffit de se connecter par web ou telnet au switch, et d'entrer les paramètres (le plus souvent un numéro de VLAN et ensuite il faut préciser à quels VLANs du switch sont alloués les différents ports. On peut aussi préciser un nom de domaine ou de VLAN). Une fois le VLAN créée, tous les segments réseaux connectés aux ports inclus dans un VLAN, font eux aussi parti du VLAN. De même pour les PC connectés à ces segments.

Un VLAN peut inclure plusieurs switchs. Un switch peut faire parti de plusieurs VLANs. Un segment (et donc les ports des switchs à chaque extrémité) peut faire transiter les données de plusieurs VLANs ; il faut alors utiliser le *trunking*.

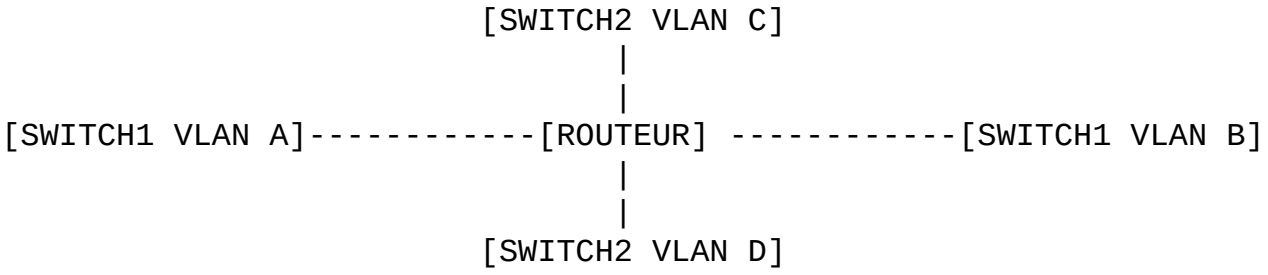
Le *VLAN trunking protocol (VTP)* est un protocole dédié à la communication inter-switchs, à propos de la configuration des VLANs



Dans l'exemple précédent, chaque switch définit 2 VLANs. Pour le premier switch, les VLAN A et B. Pour le second switch, les VLAN C et D.

A et B sont envoyés vers le routeur ou vers l'autre switch au travers de trunks. Le premier switch a d'ailleurs 2 trunks, alors que le switch connecté aux VLAN C et D n'en a qu'un seul. C et D sont envoyés vers le switch par ce trunk, mais passent aussi par le même trunk que A et B pour aller vers le routeur. Ainsi, on peut voir que les 2 trunks supportent le trafic des 4 VLANs.

Dans cette architecture, le routeur apparaît connecté aux 4 VLANS, comme s'il avait 4 liens vers les switchs :



Pour qu'un PC du VLAN A communique avec un PC du VLAN, son paquet passe par le premier switch, utilise le trunk entre le switch et le routeur. Il arrive ensuite au routeur qui le renvoie vers le switch (toujours par le trunk) en le mettant sur le VLAN B. Arrivé au switch, le paquet passe sur le segment où se trouve le PC destinataire.

Pour aller du VLAN A vers le VLAN C ou D, c'est la même procédure : passage par le premier switch, arrivée au routeur qui renvoie vers le premier switch, qui ensuite fais suivre vers le deuxième switch.

Grâce à l'utilisation d'un algorithme de trunking et de bridging transparent, les 2 PCs des 2 VLANs pensent être sur le même segment physique, alors qu'en réalité les paquets font un très grand « détour ».

10 Conclusion :

Nous avons vu ici qu’un réseau switché apporte de meilleures performances et une meilleure qualité pour les réseaux locaux. De plus, de nombreuses fonctionnalités sont apparues qui, telles que les VLANs et le spanning-tree, apportent une robustesse et une flexibilité très intéressantes aux LANs.