CISCO

Protocoles et concepts de routage - Configuration avancée des routeurs

André VAUCAMPS

Résumé

Ce livre sur la **configuration avancée des routeurs Cisco** s'adresse à tous les techniciens, ingénieurs, concernés par l'administration des protocoles de routage sur des réseaux informatiques mettant en œuvre des routeurs CISCO.
Après avoir resitué le contexte du routage, les notions de **route**, de **métrique**, de **distance administrative**, l'ouvrage inventorie les solutions possibles puis débute par la mise en œuvre du **routage statique**. **RIP** (*Routing Information Protocol*) est le premier protocole de routage dynamique étudié même s'il s'agit avant tout de justifier la migration vers des protocoles de routage plus sophistiqués. Les problèmes d'adressage sont également approfondis, l'ouvrage montre comment Internet a dû se résoudre à abandonner les classes d'adresses au profit de **CIDR** (*Classless Interdomain Routing*).
Le lecteur est initié à l'usage des **masques de longueur variable VLSM** (*Variable Length Subnet Mask*) tant pour diviser que pour agréger des réseaux. Puis l'ouvrage dédie deux longs chapitres à la mise en œuvre des deux protocoles phares en matière de routage que sont **EIGRP** (*Enhanced Interior Gateway Routing Protocol*), protocole propriété de CISCO et **OSPF** (*Open Shortest Path First*), le protocole recommandé par l'IETF. L'articulation avec les procédés de commutation en couche 2 n'est pas oubliée, c'est ainsi que le **mode de commutation CEF** (*Cisco Express Forwarding*) et sa mise en œuvre sur les routeurs sont détaillés.
L'ouvrage se veut pratique, une place importante est accordée à la **réalisation d'ateliers** dans des environnements émulés que le lecteur pourra reproduire sur son PC (fichiers disponibles en téléchargement sur www.editions-eni.fr.

Les chapitres du livre :

L'auteur

Ancien Responsable de Formation en Centre AFPA, **André VAUCAMPS** enseigne aujourd'hui dans les sections de Techniciens Supérieurs en Réseaux Informatiques et Télécommunications d'Entreprise. Il est l'auteur des livres de préparation à la certification CCNA 640-802 aux Editions ENI.

Avant-propos

Cet ouvrage n'est-il qu'un livre de plus sur les réseaux ? C'est bien sûr la question qui taraude sans doute tout auteur d'ouvrage technique, partagé entre une nécessaire humilité et le désir de faire œuvre utile et différente. Remarquons d'abord que l'abondance est surtout anglo-saxonne et que les ouvrages rédigés en français et non traduits sont plutôt rares. La seconde distinction vient de mon passé (et présent) de formateur de la formation professionnelle. Dans mon activité quotidienne, je fais toujours suivre mes cours de nombreux ateliers dans lesquels mes élèves sont invités à mettre en œuvre et donc à vérifier ce qui a été affirmé pendant la partie théorique. Il est rare que l'objectif fixé soit atteint au premier jet et la démarche que doivent alors adopter mes élèves s'approche de celle qu'il faudrait adopter en vraie grandeur quand un sinistre se produit. L'apprenti administrateur devient enquêteur et la recherche de solutions aux dysfonctionnements observés mobilise toutes ses ressources de logique. Ce modèle pédagogique a fait ses preuves, l'apprentissage est ludique et la conclusion bien connue de mes élèves : « dans 99 % des cas, le problème se situe entre le clavier et le dossier de la chaise !».

Ce qui distingue donc cet ouvrage, c'est d'avoir reproduit ce modèle en permettant le même type d'apprentissage, mais cette fois délocalisé (en dehors de tout organisme de formation), en clair « à la maison ». Pour ce faire, l'ouvrage progresse sur deux fronts, les exposés théoriques sont toujours accompagnés ou suivis de près par des vérifications dans des contextes reconstitués presque réels. Observons qu'il n'aurait pas été possible de concevoir un tel ouvrage dans les années 1980 ni même dans les années 1990. Ce n'est que depuis une époque récente, que puissance de calcul, mémoire, génie logiciel se conjuguent pour rendre possible l'existence d'outils tels que VMware, Packet Tracer ou GNS3. Avec de tels outils, et sauf pour quelques phases critiques (la sortie d'un carton et sa mise en service initiale par exemple) il devient presque inutile de travailler au pied du matériel pour apprendre à s'en servir.

Le lecteur profitera au mieux de l'ouvrage s'il maîtrise le modèle de référence OSI en couches. De plus, le lecteur est supposé averti des fondamentaux de l'adressage IP et notamment des modes d'obtention d'une adresse IP. En effet, même si cet ouvrage comporte un chapitre dédié à l'adressage, il s'agit plus de présenter les concepts de l'adressage sans classe CIDR et du masque de longueur variable VLSM que d'initier à l'adressage.

Pour le reste, la progression adoptée est classique, sont abordés le routage statique puis les protocoles de routage dynamique, les protocoles type vecteur de distance puis les protocoles type états de liens. Tout administrateur réseau sait que RIP n'est pas un protocole d'avenir, impossible pourtant de le passer sous silence. RIP reste le protocole de routage historique et peut prétendre à une certaine universalité. De plus, l'étudier présente un incontestable intérêt pédagogique. Mais dès que le réseau prend une certaine importance, il faut abandonner RIP. L'administrateur en charge d'un réseau constitué de routeurs exclusivement CISCO doit faire un choix : EIGRP, protocole propriété de CISCO ou OSPF, protocole préconisé par l'IETF. La vie de l'administrateur en charge d'un réseau hétérogène est plus simple, le choix se résume à OSPF, protocole à états de lien présent de façon systématique quelle que soit l'origine du routeur. Deux gros chapitres sont donc dédiés à chacun de ces protocoles et ambitionnent de s'affronter à la complexité. Au-delà du fonctionnement de ces protocoles, sont abordés des problèmes tels que l'agrégation de routes, l'authentification des messages ou le partage de charge. EIGRP est placé avant OSPF dans l'ouvrage mais en réalité, ces deux chapitres peuvent être étudiés dans un ordre quelconque.

L'objectif d'apprentissage sera atteint par toute personne motivée et pugnace qui dispose de cet ouvrage, d'un PC suffisamment puissant et d'une connexion à Internet. Je prends cet engagement : à la fin de l'ouvrage, le lecteur qui a réalisé l'ensemble des exercices et ateliers proposés dispose sans conteste de bases solides qui aideront à faire de lui un administrateur réseau à prendre au sérieux quand il s'agit de problèmes de routage. Ce livre s'adresse donc autant à l'administrateur réseau qui doit planifier une topologie de réseau et configurer des routeurs CISCO qu'à l'étudiant engagé dans un processus de certification professionnelle. Il sera utile enfin à toute personne intéressée par une vraie pratique du routage IP.

Le livre CISCO - Protocoles, concepts de routage et sécurité dans la collection Certifications aux Éditions ENI inclut des chapitres et ateliers supplémentaires afin d'optimiser une préparation à la Certification au CCNA 640-802.

Rappels, routage statique ou dynamique

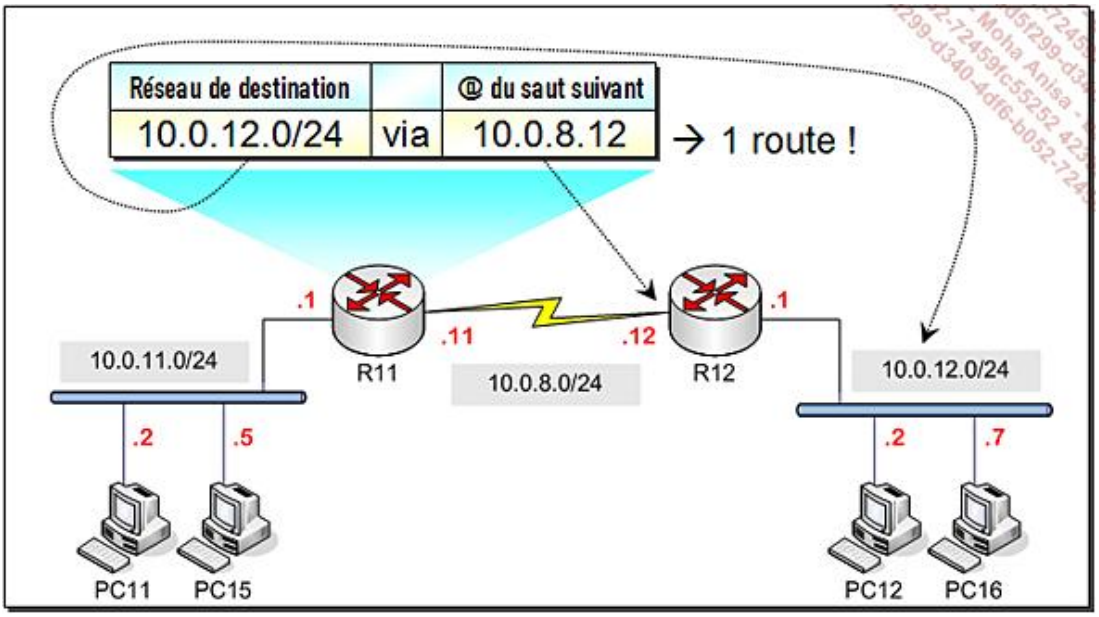
Rappelons d'abord ce qui arrive lorsqu'une trame est reçue par l'une des interfaces d'un routeur. La couche Liaison recherche l'adresse de destination. Si cette adresse identifie l'interface ou s'il s'agit d'une adresse de diffusion, le datagramme contenu dans la trame est extrait et remis au processus de couche réseau convenable (démultiplexage de protocole à l'aide du champ Type). La couche réseau examine l'adresse IP de destination. Si cette adresse est, soit l'adresse IP de l'interface, soit l'adresse de diffusion limitée (255.255.255.255), alors ce datagramme est arrivé à destination (il n'ira pas plus loin). S'il s'agit d'un datagramme IP, le champ « Protocol » est utilisé afin de remettre le contenu du datagramme au processus convenable (par exemple, 1 pour ICMP, 6 pour TCP, 17 pour UDP...).

Toute adresse de destination différente indique que le datagramme doit être routé. Il peut s'agir de l'adresse d'un hôte ou d'une adresse de diffusion dirigée. Si l'adresse est celle d'un hôte, cet hôte peut appartenir à un réseau directement connecté (ce routeur est le dernier, le datagramme doit maintenant être remis à l'hôte) ou non directement connecté, le routeur examine alors sa table de routage à la recherche d'une route convenable.

Il est indispensable de resituer les quelques notions qui suivent extraites de l'ouvrage Cisco - Notions de base sur les réseaux dans la collection Certifications aux Editions ENI. Cette courte section est donc destinée au lecteur qui aurait acquis le présent ouvrage seul.

1. Notion de route

Dans l'exemple ci-dessous, le routeur passerelle se voit confier les datagrammes dont l'adresse de destination est extérieure au réseau. Charge à lui de les faire progresser vers leur destination et pour ce faire, le routeur consulte sa table de routage à la recherche d'une route vers le réseau en question. Comment se présente une route dans cette table de routage ?



A minima, il s'agit d'une correspondance entre un réseau qu'il est possible d'atteindre et l'adresse IP du prochain routeur à qui il faut confier les datagrammes pour s'approcher de ce réseau ou l'atteindre. Ainsi dans l'exemple ci-dessus, le routeur R11 pour atteindre le réseau 10.0.12.0/24 doit confier les paquets à l'adresse 10.0.8.12. La route est donc la correspondance 10.0.12.0/24 via 10.0.8.12.

L'apprentissage de cette route et par suite, le remplissage de la table de routage peut être le fait de l'administrateur, on parle alors de routage **statique**. Il existe également des protocoles de routage qui, par des échanges réguliers entre routeurs, permettent à chacun des routeurs de découvrir des informations de route ou de topologie de réseau, le remplissage de la table de routage est alors automatisé, ce que l'on désigne par routage **dynamique**.

2. Routage statique

Une route statique est le fait de l'administrateur, il faut l'inscrire manuellement dans la table de routage.

Parmi les inconvénients :

- Toute modification de topologie requiert l'intervention de l'administrateur ce qui peut rapidement devenir

pesant.

- La panne d'un équipement ou d'une interface est une modification de topologie accidentelle, non planifiée. Le temps d'indisponibilité est fonction du délai de prise en compte du défaut par l'administrateur.

Parmi les avantages :

- Le routeur n'a pas à consacrer une partie de ses ressources à l'entretien d'un protocole de routage (CPU, mémoire).

Les domaines d'emploi du routage statique sont :

- Les petits réseaux.
- Les réseaux privés connectés à l'Internet via un seul fournisseur d'accès.

3. Routage dynamique

À l'aide d'un protocole de routage, un routeur partage des informations concernant les réseaux qu'il connaît avec d'autres routeurs qui utilisent le même protocole. Chaque correspondance @réseau distant - @prochain saut (chaque route) mentionne le mode d'apprentissage de la route (S pour statique, C pour directement connectée, R pour RIP (*Routing Information Protocol*)...).

Les correspondances sont maintenues à jour au fur et à mesure de la vie du réseau. C'est même l'une des performances attendues d'un protocole de routage que de diminuer autant que faire se peut le temps qui s'écoule entre une modification de topologie, planifiée ou accidentelle, et sa prise en compte dans les tables de routages. Ce délai est appelé « temps de convergence ».

4. La table de routage

Routage statique et dynamique peuvent être utilisés conjointement, la table de routage comporte alors :

- des routes directement connectées :
  - les premières à apparaître dans la table ;
  - leur présence est obligatoire (un routeur sans interfaces n'a pas de sens) ;
  - une route directement connectée n'apparaît que lorsque l'interface correspondante est active.
- des routes statiques ;
- des routes dynamiques.

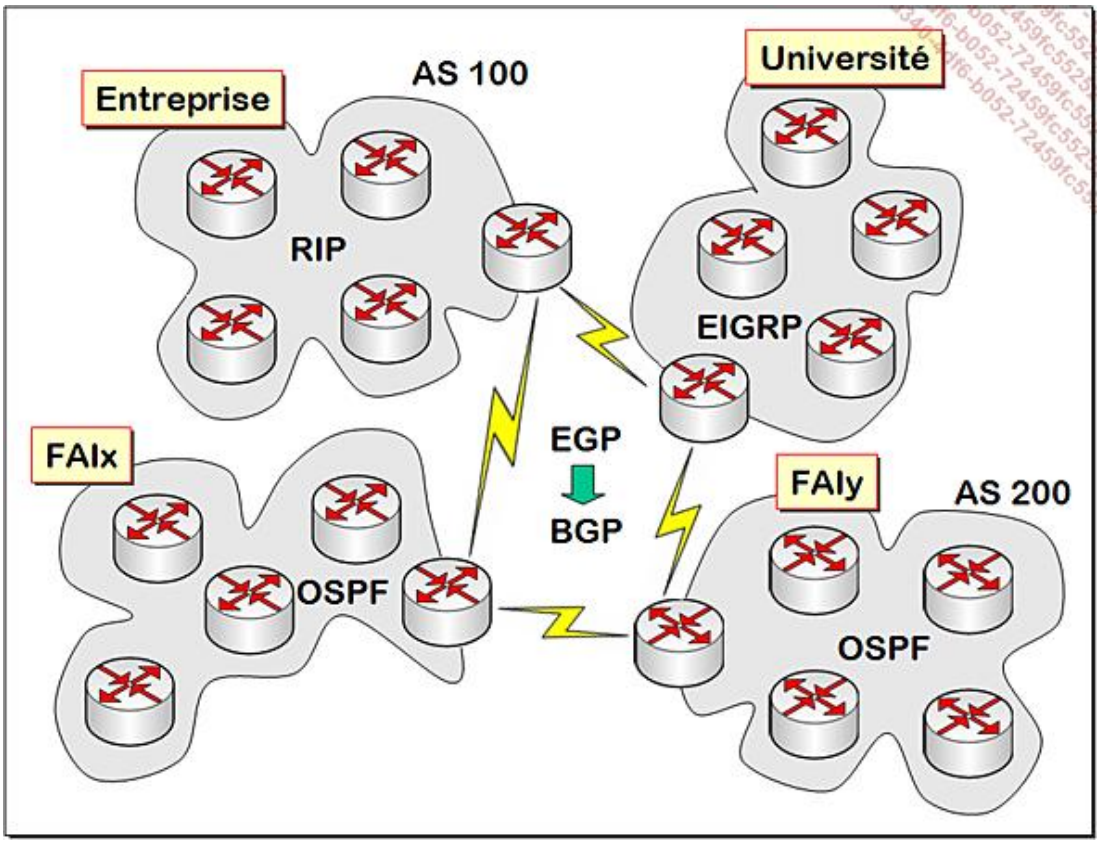
Seules les routes statiques et dynamiques concernent les réseaux distants (non directement connectés).

La table de routage est stockée en mémoire RAM et doit donc être reconstruite à chaque initialisation de l'équipement.

5. Les protocoles de routage

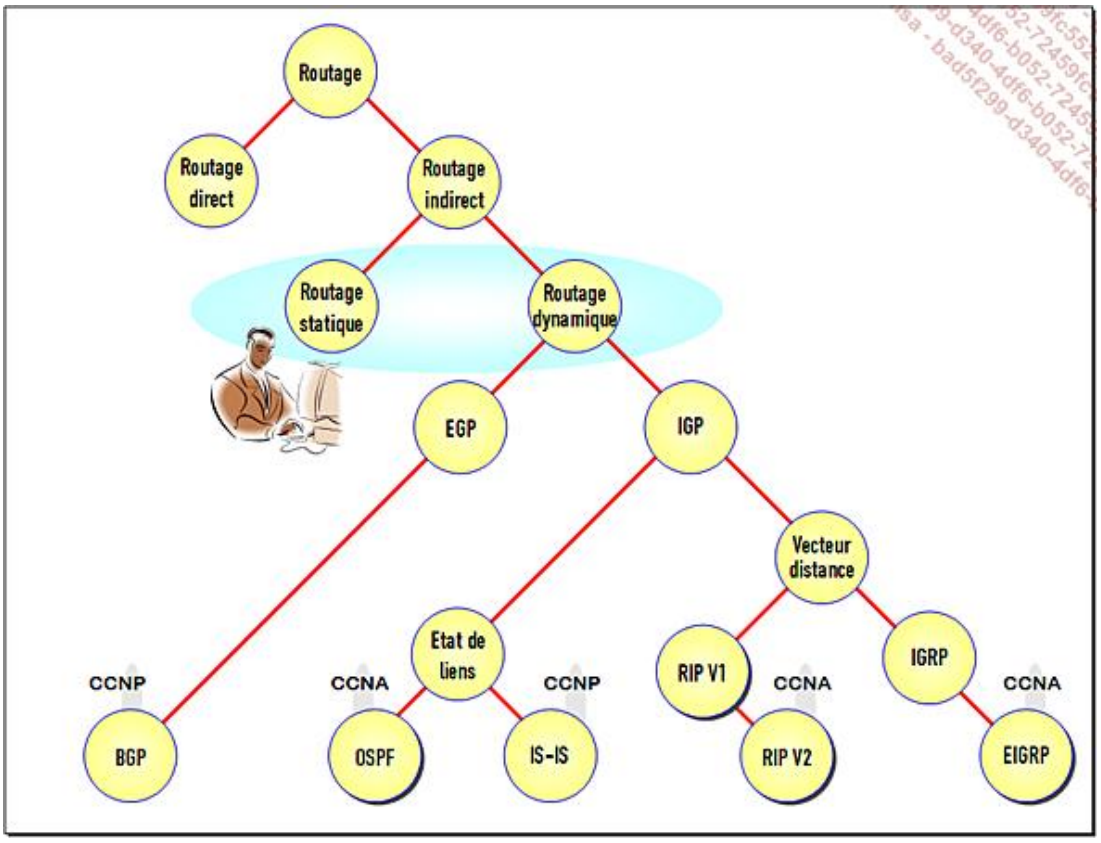
a. Notion de système autonome

Vouloir propager l'information de topologie de chaque routeur sur l'ensemble de la planète est hors de portée (consommation de bande passante, difficultés de maintenance, sécurité). Le réseau mondial résulte d'un assemblage de systèmes autonomes. Un système autonome (AS : *Autonomous System*) est un ensemble de réseaux et de routeurs partageant le même protocole de routage et géré par une même autorité administrative.



b. Protocoles de routage internes, externes

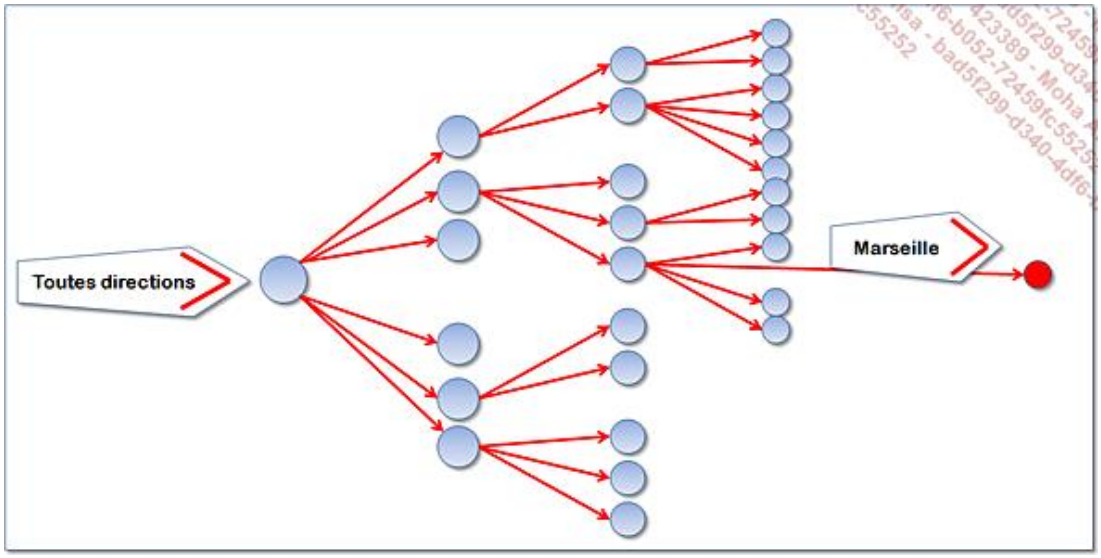
Les protocoles mis en œuvre dans un système autonome appartiennent à la famille des IGP (*Interior Gateway Protocol*). Entre systèmes autonomes interviennent les procoles EGP (*Exterior Gateway Protocol*) mais cette famille se résume au seul protocole actuellement viable BGP (*Border Gateway Protocol*).



Les protocoles IGP fondent leurs décisions sur des critères de performances, débit, fiabilité, nombre de sauts... Le protocole BGP intègre en plus des critères politiques. Imaginons que vous ayez à établir un plan de vol de Compiègne au nord de Paris à Etampes au sud de Paris, un protocole IGP trace une route directe qui vous fait survoler Paris. Le protocole BGP vous fera contourner Paris parce que le survol de la capitale est interdit.

La famille des protocoles IGP est une famille nombreuse mais essentiellement fondée sur deux technologies : le routage à vecteur de distance et le routage à état de liens.

6. Ce qui caractérise une route

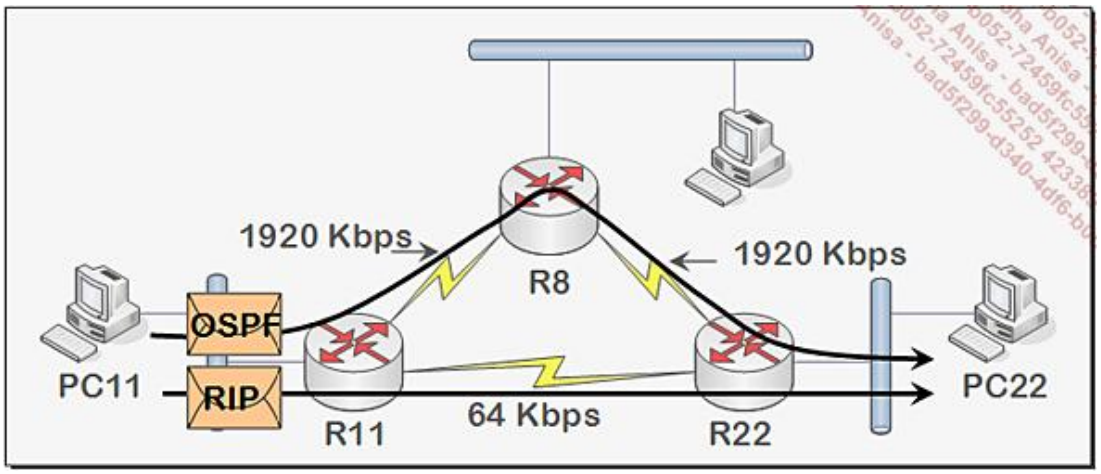


Vous habitez Paris et préparez un itinéraire afin de vous rendre à Marseille, sur le boulevard de la Canebière au numéro 10. Votre destination est donc le Sud mais aussi la région PACA, plus encore les Bouches-du-Rhône, évidemment Marseille, la Canebière et enfin le n°10. Toutes les destinations que nous venons de citer sont exactes mais plus ou moins précises. La première caractéristique d'une route est sa destination, la seconde est son degré d'acuité. Vous décidez d'utiliser un site de préparation d'itinéraire type ViaMichelin ou Mappy. Il vous faut préciser un choix parmi {Conseillé | Plus rapide | Plus court | Découverte | Economique}. Ce choix est déterminant sur le coût du voyage, cette notion existe également pour une route réseau, on parle de **métrique**. Enfin, vous vous précipitez sur une promotion en cours et achetez un GPS dernier cri. Après quelques heures passées à errer sur nos belles routes françaises, vous voilà au milieu d'une cour de ferme. Quelle confiance fallait-il accorder à la route proposée par ce GPS ? Ce degré de confiance caractérise non pas la route en elle-même mais la source d'apprentissage de la route, on l'appelle **distance administrative**.

a. Métrique associée à une route

La métrique est donc l'une des caractéristiques d'un protocole de routage. La plus simple est sans doute celle du protocole à vecteur de distance RIP, égale au nombre de sauts. L'une des plus sophistiquées est celle du protocole propriétaire EIGRP (*Enhanced Interior Gateway Routing Protocol*) puisqu'elle associe délai, bande passante, fiabilité et charge. La métrique d'OSPF (*Open Shortest Path First*) additionne les coûts des différents liens qui composent la route, le coût d'un lien est fonction de sa bande passante.

L'exemple suivant, classique, montre les absurdités auxquelles peut conduire une métrique rudimentaire :

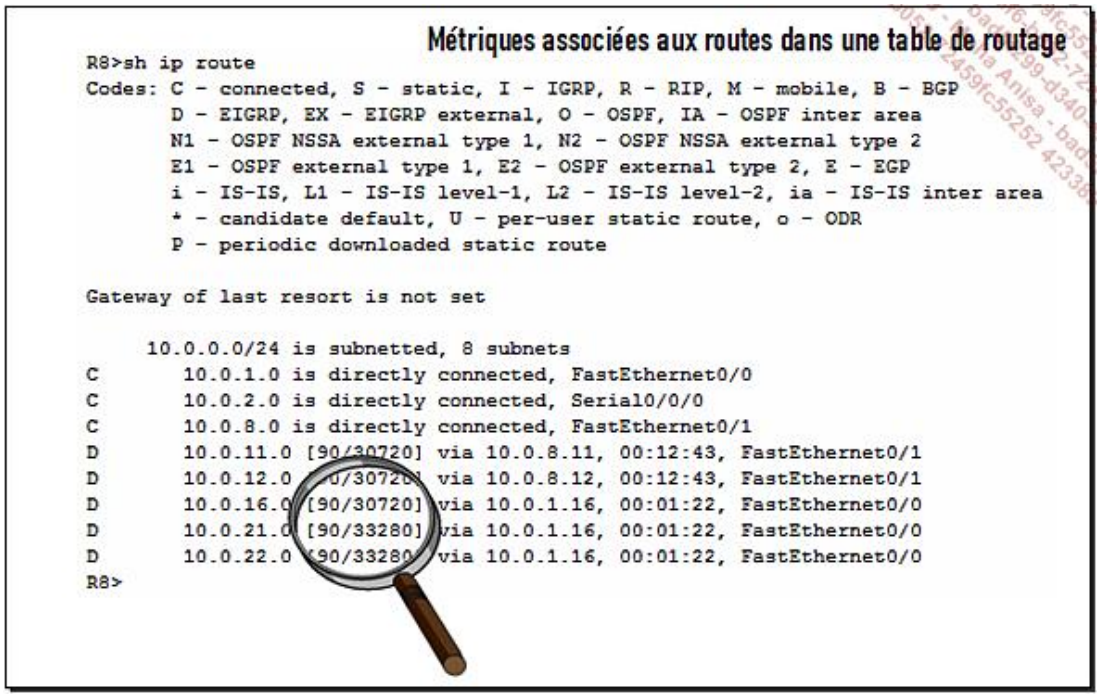


Si les trois routeurs remplissent leur table de routage avec RIP, un paquet émis par PC11 et destiné à PC22 transite par une route directe dont certes le nombre de sauts est moindre mais dont la bande passante n'est que le trentième de celle offerte par la route qui transite par R8.

Il arrive qu'un protocole de routage fournisse plusieurs routes pour une même destination. Dans ce cas, il ne place dans sa table de routage que la route la plus favorable. Pour un protocole de routage donné, la meilleure route est celle dont la métrique est **la plus faible**.

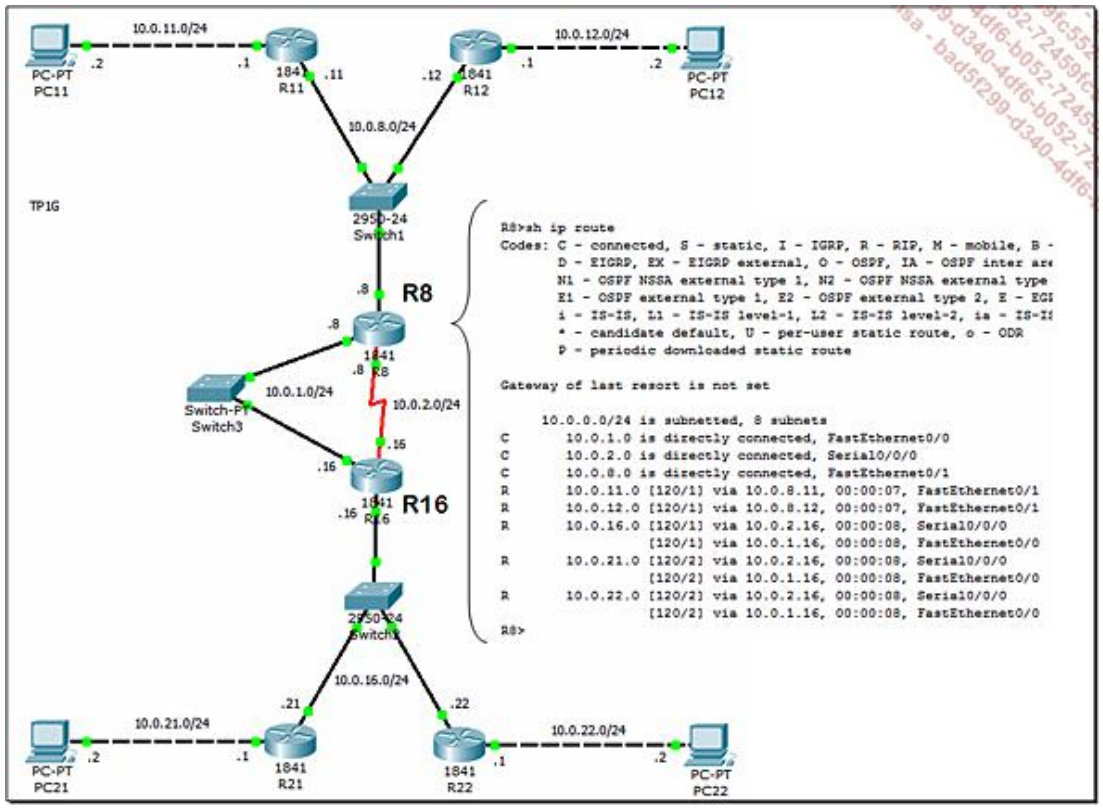
À un instant donné et pour un protocole de routage donné, chaque route contenue dans la table de routage est le meilleur chemin parmi les routes connues vers une destination. Les autres routes sont ignorées.

Attention, ignorées ne signifie pas perdues. Dans le cas où une modification de topologie entraînerait l'indisponibilité d'une route présente dans la table et dans le cas où cette route résultait d'un choix parmi des routes à métriques différentes, l'une des routes ignorées jusque-là pourrait se substituer à la route défaillante.



L'exemple ci-dessus montre la table de routage d'un routeur configuré pour mettre en oeuvre le protocole EIGRP. Chaque route découverte à l'aide de ce protocole est précédée de la lettre « D » (EIGRP est fondé sur l'algorithme DUAL : *Diffusing Update Algorithm*). Observez les champs présents immédiatement à droite du réseau de destination, deux valeurs entre crochets et séparées par un caractère « / » : la première valeur est la distance administrative (patientez...), la seconde valeur est la métrique.

Il arrive qu'un protocole de routage fournisse deux ou plusieurs routes vers une même destination et avec des métriques identiques :



Dans l'exemple ci-dessus, le protocole de routage mis en œuvre est RIP. Parce qu'il existe deux liaisons entre R8 et R16, R8 inscrit dans sa table de routage deux routes à métriques identiques vers les réseaux 10.0.16.0/24, 10.0.21.0/24 et 10.0.22.0/24. Observez la table de routage, chaque réseau de destination concerné apparaît associé aux deux sauts possibles.

On parle dans ce cas de chemins à coût égal, seul cas où le routeur ne choisit pas une route mais prend les deux routes (ou davantage) en compte pour faire progresser le trafic vers le réseau de destination en le répartissant sur les deux liens, ce que l'on désigne par « **Partage de charge à coût égal** ».

b. La distance administrative

Nous avons dit que la métrique est caractéristique du protocole de routage. Comparer deux métriques n'a de sens que si elles sont issues toutes deux du même protocole de routage. Le plus ordinairement, les routes dynamiques installées dans la table de routage sont issues d'un unique protocole de routage qui les a choisies parce que, parmi les routes connues, ces routes avaient la meilleure métrique. Quelques cas rares obligent à configurer plusieurs protocoles de routage sur un même routeur, ce qui peut se produire lorsqu'un routeur est placé sur la frontière séparant deux domaines distincts, un protocole de routage distinct étant déployé sur chacun de ces domaines.

Comment le routeur peut-il opérer un choix parmi plusieurs routes pour un même réseau de destination quand ces routes sont issues de protocoles de routage différents ?

Impossible cette fois de comparer les métriques. Le choix qui a été fait est d'associer un degré de confiance à chacun des protocoles de routage, degré de confiance appelé distance administrative. Sa valeur est comprise entre 0 et 255, le routeur privilégie la route à distance administrative la plus faible.

Il est possible d'utiliser des valeurs autres que celles attribuées par défaut, mais il est conseillé de connaître ces valeurs par défaut :

- Route directement connectée : DA = 0 (une confiance absolue).
- Route statique : DA = 1 (c'est l'administrateur qui entre la route, on considère qu'il sait ce qu'il fait).
- Route issue de EIGRP : DA = 90.
- Route issue de IGRP : DA = 100 (normalement abandonné au profit de EIGRP).
- Route issue de OSPF : DA = 110.

- Route issue de RIP : DA = 120.

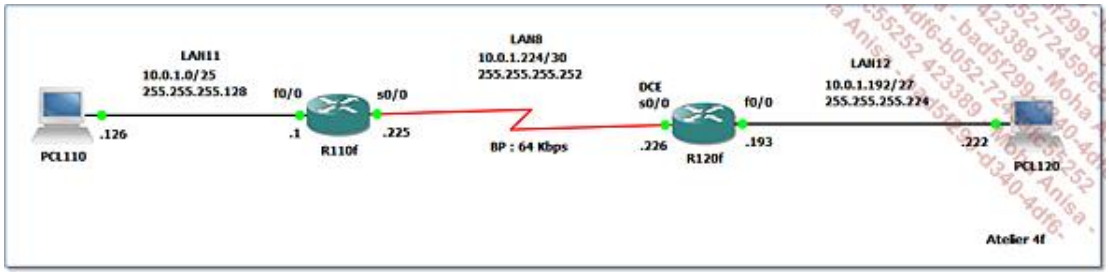
- DA = 255 → source non fiable, la route n'est pas installée dans la table de routage.

Revenez un peu plus haut à la figure illustrant les métriques associées aux routes, dans le résultat d'une commande **show ip route**. À chaque route sont associées les deux valeurs [DA/Métrique]. Puisqu'il s'agissait du protocole EIGRP, on retrouve la valeur DA = 90.

Quand la métrique permet de choisir la route la plus pertinente, la distance administrative permet d'établir le mode d'apprentissage de route préféré.

Réseaux directement connectés

Sans entrer de route statique et sans avoir mis en œuvre un quelconque protocole de routage, on pourrait penser que la table de routage reste vide. Mais chaque interface à l'état monté « up » et dotée d'une configuration IP provoque l'ajout d'une route dans la table de routage. Appuyons notre propos sur le contexte ci-dessous, reproductible dans GNS3 :



Une route directement connectée est ajoutée à la table de routage si les deux conditions suivantes sont satisfaites :

- L'interface est dans un état actif à la fois en couche 1 et en couche 2. L'état actif en couche 1 suppose qu'il y ait bien une connectivité physique entre cette interface et l'interface de l'équipement en vis-à-vis. Par exemple, une interface WAN est connectée à son boîtier CSU/DSU lui-même actif, une interface LAN est connectée à un port de commutateur actif. L'état actif en couche 2 suppose qu'il y ait compatibilité de protocole sur les couches 2 respectives des interfaces en vis-à-vis. Dans le cas d'une interface LAN, pas de problème depuis la prééminence d'Ethernet. Dans le cas d'une interface WAN, l'administrateur doit avoir configuré le même protocole de couche 2 aux deux extrémités du lien. Par défaut, l'IOS réalise une encapsulation HDLC.

- Une configuration IP a été assignée à l'interface, soit par le fait de l'administrateur (commande **ip address**), soit obtenue d'un serveur DHCP.

L'administrateur peut confirmer l'état d'une ou plusieurs interfaces à l'aide de la commande **show ip interface brief** (abrégée en **sh ip int br**). Par exemple sur R110f :

```
R110f#sh ip int br
Interface              IP-Address      OK? Method Status                Protocol
FastEthernet0/0        10.0.1.1        YES manual up                    up
Serial0/0              10.0.1.225      YES manual up                    up
FastEthernet0/1        unassigned      YES unset  administratively down down
R110f#
```

La commande **show ip route** affiche l'intégralité du contenu de la table de routage. Mais avant de fournir les différentes routes qui composent la table, l'interface rappelle quelles sont les sources d'apprentissage possible. Chaque source est associée à une lettre : **S** pour les routes statiques, **I** pour le protocole de routage IGRP, etc. Les routes directement connectées apparaissent avec la lettre **C** en regard :

```
R110f#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 3 subnets, 3 masks
C       10.0.1.0/25 is directly connected, FastEthernet0/0
S       10.0.1.192/27 [1/0] via 10.0.1.226
C       10.0.1.224/30 is directly connected, Serial0/0
R110f#
```

Il est également possible de n'afficher que les seules routes directement connectées en ajoutant le mot-clé « *connected* » à la commande **show ip route** :

```
R110f#sh ip route connected
     10.0.0.0/8 is variably subnetted, 3 subnets, 3 masks
C       10.0.1.0/25 is directly connected, FastEthernet0/0
C       10.0.1.224/30 is directly connected, Serial0/0
R110f#
```

Avec cette même topologie, imaginons le routeur R120f hors service (sous GNS3, clic droit sur le routeur R120f puis sélectionnez **Suspendre**). Sur la console du routeur R110f, un message SYSLOG avertit de la tombée de l'interface s0/0 :

```
00:58:01: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state
to down
```

Toujours sur le routeur R110f, la commande **sh ip int br** donne cette fois :

```
R110f#sh ip int br
Interface              IP-Address     OK? Method Status             Protocol
FastEthernet0/0        10.0.1.1       YES manual up                    up
Serial0/0              10.0.1.225     YES manual up                    down
FastEthernet0/1        unassigned     YES unset  administratively down down
R110f#
```

Une commande **show ip route** confirme l'évènement, la route directement connectée vers le réseau 10.0.1.224/30 a disparu :

```
R110f#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is not set

     10.0.0.0/25 is subnetted, 1 subnets
C       10.0.1.0 is directly connected, FastEthernet0/0
R110f#
```

Routes statiques

1. Route statique vers l'adresse du saut suivant

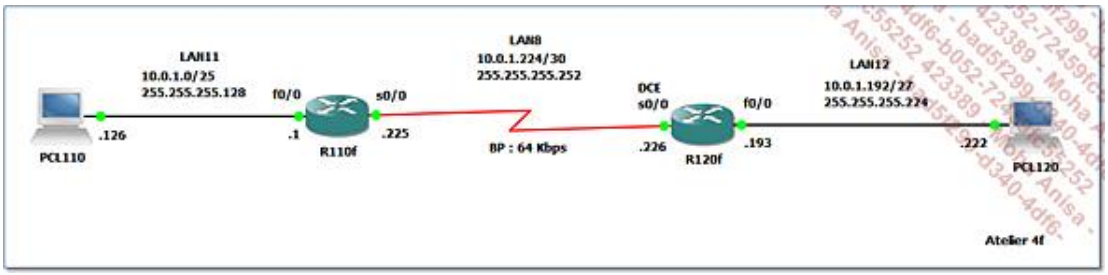
La commande **ip route**, en mode de configuration globale, doit être utilisée autant de fois qu'il y a de routes statiques à créer. Voici sa syntaxe la plus globale :

```
Router(config)#ip route @destination masque {@saut_suivant | interface_sortie}
[distance] [tag] [permanent]
```

Dont les arguments sont les suivants :

Mot-clé	Description
@destination	L'adresse IP du réseau distant.
Masque	Le masque du réseau distant. Plus le préfixe est long, plus la route est précise. Plus le préfixe est court, plus la route agrège des réseaux.
@saut_suivant	L'adresse IP du prochain saut.
Interface_sortie	Quand on fait le choix de pointer une interface de ce routeur plutôt que l'adresse IP de l'interface du routeur suivant.
Distance	Impose une distance administrative différente de la distance par défaut pour une route statique (valeur 1).
Tag	Marqueur utilisé pour la redistribution de routes.
Permanent	La route n'est jamais effacée de la table de routage même en cas de tombée de l'interface.

Appuyons à nouveau notre propos sur le contexte ci-dessous, reproductible dans GNS3 :



À cet instant, les tables de routage des deux routeurs R110f et R120f ne contiennent que les routes directement connectées. Sur PCL110, une requête **ping** émise vers PCL120 est bien remise à la passerelle de PCL110, c'est-à-dire l'interface f0/0 de R110f. Sur R110f, ajoutons une route statique vers le réseau LAN12 :

```
R110f#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R110f(config)#ip route 10.0.1.192 255.255.255.224 10.0.1.226
R110f(config)#^Z
R110f#
03:22:10: %SYS-5-CONFIG_I: Configured from console by console
R110f#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
```

Si R110f peut atteindre le saut suivant 10.0.1.226, il ajoute la route vers LAN12 à la table de routage, ce que confirme une commande **sh ip route** :

```
R110f#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 3 subnets, 3 masks
C       10.0.1.0/25 is directly connected, FastEthernet0/0
S       10.0.1.192/27 [1/0] via 10.0.1.226
C       10.0.1.224/30 is directly connected, Serial0/0
R110f#
```

R120f reçoit la requête **ping** émise par PCL110 et peut la remettre à son destinataire puisque PCL120 est placé sur un réseau directement connecté. PCL120 répond à la requête, l'adresse de destination de la réponse est 10.0.1.126, placée sur le réseau LAN11, la réponse est remise à la passerelle de PCL120, c'est-à-dire R120f. L'administrateur a pris soin de taper une commande **debug ip icmp** sur R120f, dont voici le résultat :

```
R120f#debug ip icmp
ICMP packet debugging is on
R120f#
03:19:49: ICMP: dst (10.0.1.126) host unreachable sent to 10.0.1.222
03:19:50: ICMP: dst (10.0.1.126) host unreachable sent to 10.0.1.222
03:19:51: ICMP: dst (10.0.1.126) host unreachable sent to 10.0.1.222
R120f#
```

Sur R120f, ajoutons la route manquante vers LAN11 :

```
R120f#u all
All possible debugging has been turned off
R120f#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R120f(config)#ip route 10.0.1.0 255.255.255.128 10.0.1.225
R120f(config)#^Z
R120f#
04:14:57: %SYS-5-CONFIG_I: Configured from console by console
R120f#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
```

Si R120f peut atteindre le saut suivant 10.0.1.225, il ajoute la route vers LAN11 à la table de routage, ce que confirme une commande **sh ip route** :

```
R120f#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 3 subnets, 3 masks
S       10.0.1.0/25 [1/0] via 10.0.1.225
C       10.0.1.192/27 is directly connected, FastEthernet0/0
C       10.0.1.224/30 is directly connected, Serial0/0
R120f#
```

Cette fois, la requête ICMP de PCL110 aboutit.

a. La commande ping étendue

Toujours en nous appuyant sur la topologie précédente, imaginons cette fois que l'administrateur se trouve devant une session console sur R110f et que sauf à envisager des déplacements compliqués, il souhaite régler la totalité de la configuration (tests compris) depuis cette session console. Pour rétablir le contexte tel qu'il était avant l'introduction des routes statiques, utilisons la forme **no** des commandes **ip route**.

Sur R110f :

```
R110f#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
```

```
R110f(config)#no ip route 10.0.1.192 255.255.255.224 10.0.1.226
R110f(config)#^Z
R110f#
```

Sur R120f :

```
R120f#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R120f(config)#no ip route 10.0.1.0 255.255.255.128 10.0.1.225
R120f(config)#^Z
R120f#
```

Astreignons-nous à opérer toutes les manipulations à venir depuis la session console ouverte sur R110f. Vérifions la connectivité de R110f vers PCL120 :

```
R110f#ping 10.0.1.222
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.222, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
R110f#
```

Une commande **ping** @destination provoque 5 requêtes ICMP consécutives. Dans le cas présent, l'ensemble des requêtes échoue, c'est normal. Ajoutons la route statique vers LAN12 :

```
R110f#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R110f(config)#ip route 10.0.1.192 255.255.255.224 10.0.1.226
R110f(config)#^Z
R110f#
05:02:43: %SYS-5-CONFIG_I: Configured from console by console
R110f#
```

Tentons à nouveau la requête **ping** vers PCL120 :

```
R110f#ping 10.0.1.222

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.222, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 40/70/92 ms
R110f#ping 10.0.1.222

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.222, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/57/76 ms
R110f#
```

Cette fois, au premier essai, 4 requêtes sur 5 réussissent. Aux essais suivants, toutes les requêtes aboutissent. Pourquoi diable la toute première requête échoue ? Après quelques analyses Wireshark, l'une sur le lien « serial » reliant les deux routeurs (**cap_2D_03.pcap** disponible sur le site des Editions ENI), l'autre sur LAN12 (**cap_2D_04.pcap** disponible sur le site des Editions ENI), l'explication tombe. La première requête sans réponse n'est pas le fait de R110f qui a bien fait progresser l'ensemble des 5 requêtes. Sur le lien « serial », on observe bien les 5 requêtes mais seulement 4 réponses. En revanche, la capture réalisée sur LAN12 ne fait apparaître que 4 requêtes. C'est donc R120f qui, faute de disposer de la correspondance @IP-@MAC dans le cache ARP, diffuse une requête ARP (trames 5 et 6 de la capture) mais supprime la requête ICMP. Une erreur assez répandue est de penser que la première requête échoue du fait du routeur d'origine, R110f dans le cas présent, et à cause d'une correspondance non présente dans le cache ARP. Mais R110f n'a nul besoin de cette correspondance car la liaison vers R120f est de type point à point, l'adresse physique du correspondant est connue.

Revenons à notre problème de départ qui consiste à bâtir la connectivité de PCL110 à PCL120. Le **ping** réussi précédent nous confirme la connectivité de R110f à PCL120, c'est insuffisant. Comment tester la connectivité de PCL110 à PCL120 sans être devant PCL110 ? La commande **ping** admet en fait de nombreux paramètres en dehors de l'adresse de destination dont un qui permet de spécifier la source, ce peut être une interface ou une adresse IP. N'allez pas croire que l'on peut spécifier l'adresse de PCL110 en tant qu'adresse source, ce serait une usurpation d'identité. Mais avec le mot-clé source, on peut spécifier soit l'interface f0/0 de R110f soit l'adresse IP 10.0.1.1. L'important est que cette interface ou cette adresse appartiennent au réseau LAN11. Si la réponse à une requête dotée d'une telle adresse source parvient jusqu'à R110f, alors nous sommes assurés de la connectivité de PCL110 à PCL120.

```
R110f#ping 10.0.1.222 source 10.0.1.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.222, timeout is 2 seconds:
Packet sent with a source address of 10.0.1.1
.....
Success rate is 0 percent (0/5)
R110f#
```

C'était prévisible puisque R120f ne connaît pas encore de route vers LAN11. Toujours sur la session console R110f, ouvrons une session Telnet vers R120f afin d'ajouter la route statique manquante :

```
R110f#telnet 10.0.1.226
Trying 10.0.1.226 ... Open

User Access Verification
Password:
R120f>en
Password:
R120f#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R120f(config)#ip route 10.0.1.0 255.255.255.128 10.0.1.225
R120f(config)#^Z
R120f#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
R120f#exit

[Connection to 10.0.1.226 closed by foreign host]
R110f#
```

Vérifions à nouveau la connectivité de LAN11 vers PCL120 :

```
R110f#ping 10.0.1.222 source 10.0.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.222, timeout is 2 seconds:
Packet sent with a source address of 10.0.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 28/74/156 ms
R110f#
```

Magique !

L'administrateur las de devoir mémoriser des mots-clés supplémentaires sera satisfait d'apprendre que CISCO a prévu une commande **ping** interactive. Il suffit de lancer la commande sans arguments. Ceci provoque l'exécution d'un process ping qui pose toutes les questions nécessaires en vue de composer une requête ICMP « sur mesure ». Voici l'équivalent interactif de la commande **ping** précédente :

```
R110f#ping
Protocol [ip]:
Target IP address: 10.0.1.222
Repeat count [5]:
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: 10.0.1.1
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]:
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.222, timeout is 2 seconds:
Packet sent with a source address of 10.0.1.1
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/64/136 ms
R110f#
```

Observez que, hormis pour les adresses source et destination, les questions posées proposent une réponse par défaut que l'on peut accepter en validant simplement par la touche [Entrée].

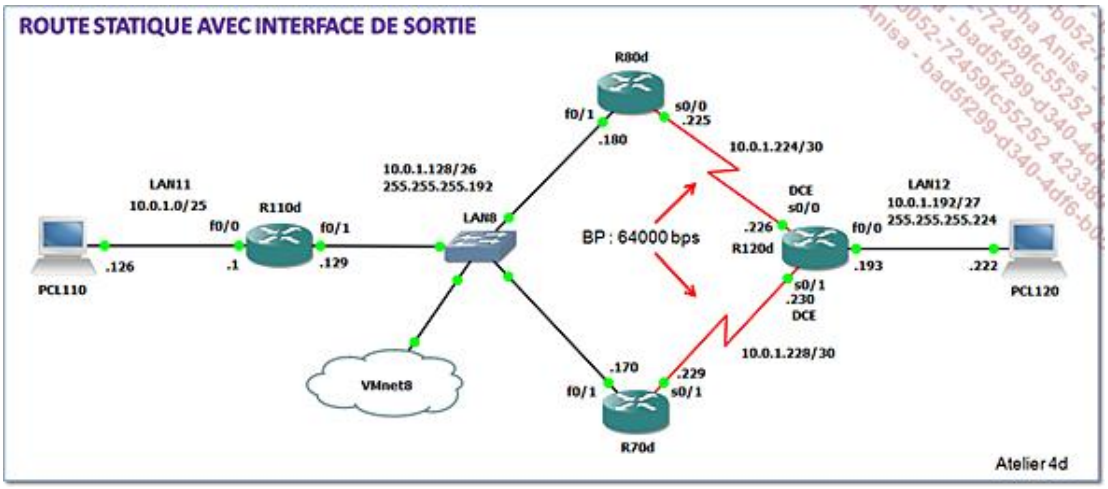
2. Route statique vers une interface de sortie

Jusqu'à présent, une route directement connectée était une route que l'IOS avait déduite de la configuration d'une interface. « SI l'adresse IP de l'interface fa0/0 est 10.0.1.1/25, ALORS le réseau 10.0.1.0/25 est directement connecté à cette interface ». L'IOS offre également cette possibilité qui consiste, dans une route statique, à indiquer une interface de sortie plutôt que l'adresse de saut suivant. L'IOS considère alors cette route comme également directement connectée. Ce qui amène à se remémorer son comportement avec une route directement connectée. Avec une telle route, l'IOS considère toute adresse de destination appartenant au réseau directement connecté comme étant directement joignable. Il lui reste à confier le datagramme au pilote de l'interface afin qu'il compose l'ultime trame qui encapsulera ce datagramme. Pour ce faire, le pilote doit disposer de l'adresse physique de destination.

À ce stade, le comportement diffère selon le type d'interface de sortie, LAN ou WAN. Dans le cas d'une interface « serial », l'interface de sortie est connectée à une liaison point à point. Pas d'ambigüité, le pilote sait qui il est et qui est l'autre, il peut composer la trame. Le datagramme est donc remis à l'autre extrémité qui est aussi l'interface de saut suivant. Les choses se compliquent quand l'interface de sortie est une interface LAN. L'adresse de destination est une adresse de couche 3, le pilote doit disposer de l'adresse de couche 2 correspondante. En IPv4, c'est à ce moment qu'intervient le mécanisme ARP. Il peut être utile de se reporter à l'ouvrage Cisco - Notions de base sur les réseaux dans la collection Certifications aux Editions ENI - chapitre Obtention d'une adresse IP - section Le protocole ARP. Si la correspondance @IP-@MAC est présente dans le cache ARP, le pilote peut composer la trame. Dans le cas contraire, le datagramme est placé en file d'attente et le process ARP diffuse une requête afin d'apprendre l'adresse physique du destinataire.

Raisonnons à nouveau sur la route statique qui utilise une interface de sortie. Dans le cas d'une interface serial, un datagramme qui emprunte cette route est remis sans formalités au pilote de l'interface qui peut composer la trame. Y-a-t-il un intérêt à procéder de la sorte ? Difficile de répondre, il faudrait avoir une conversation sérieuse avec les personnes chargées du développement chez CISCO. Tout au plus, peut-on remarquer que l'IOS balaye la table de routage une seule fois quand la route qu'il décide d'utiliser est directement connectée. Quand il choisit une route avec adresse de saut suivant, il doit parcourir la table une seconde fois à la recherche de l'interface connectée au réseau qui comprend l'adresse de saut suivant.

Le comportement avec une interface de sortie de type LAN est plus commode à expliquer en s'appuyant sur un cas concret. Examinez la topologie proposée ci-après :



Ce contexte a été testé à l'aide de GNS3. Sur le routeur R110d, la route statique créée pour rejoindre le réseau LAN12 est de type à interface de sortie :

```
R110d#show run
Building configuration...
...
...
ip route 10.0.1.192 255.255.255.224 FastEthernet0/1
ip route 10.0.1.224 255.255.255.252 FastEthernet0/1
ip route 10.0.1.228 255.255.255.252 FastEthernet0/1
ip http server
...
end
```

```
R110d#
```

Une commande **show ip route** sur le routeur R110d donne le résultat suivant :

```
R110d#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 5 subnets, 4 masks
C       10.0.1.0/25 is directly connected, FastEthernet0/0
C       10.0.1.128/26 is directly connected, FastEthernet0/1
S       10.0.1.192/27 is directly connected, FastEthernet0/1
S       10.0.1.224/30 is directly connected, FastEthernet0/1
S       10.0.1.228/30 is directly connected, FastEthernet0/1
R110d#
```

Sur R80d, une commande **show controller f0/1** permet de découvrir l'adresse MAC de cette interface :

```
R80d#sh controller f0/1
Interface FastEthernet0/1
Hardware is AMD Am79c971
...
  Promiscuous Mode Disabled, PHY Addr Enabled, Broadcast Addr Enabled
PHY Addr=C800.0764.0001, Multicast Filter=0x0000 0x0100 0x0000 0x0000
...
R80d#
```

La même commande sur R70d :

```
R70d#show controller f0/1
Interface FastEthernet0/1
Hardware is AMD Am79c971
...
  Promiscuous Mode Disabled, PHY Addr Enabled, Broadcast Addr Enabled
PHY Addr=C802.0764.0001, Multicast Filter=0x0000 0x0100 0x0000 0x0000
...

R70d#
```

Imaginons un datagramme en provenance de PCL110 destiné à PCL120. R110d considère la route statique vers LAN12 comme une route directement connectée et peut remettre ce datagramme au pilote de l'interface f0/1 en même temps qu'il sollicite le processus ARP, ce afin d'obtenir l'adresse physique correspondant à l'adresse de destination 10.0.1.222. Cette correspondance n'est pas encore présente dans le cache ARP comme le confirme le résultat d'une commande **show ip arp** :

```
R110d#sh ip arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  10.0.1.1                -   c801.0764.0000  ARPA   FastEthernet0/0
Internet  10.0.1.129              -   c801.0764.0001  ARPA   FastEthernet0/1
R110d#debug arp
ARP packet debugging is on
R110d#
00:01:02: IP ARP: creating incomplete entry for IP address: 10.0.1.222 interface
FastEthernet0/1
00:01:02: IP ARP: sent reqsrc 10.0.1.129 c801.0764.0001,dst 10.0.1.222
0000.0000.0000 FastEthernet0/1
00:01:02: IP ARP: rcvd rep src 10.0.1.222 c802.0764.0001, dst 10.0.1.129
FastEthernet0/1
00:01:02: IP ARP: rcvd rep src 10.0.1.222 c800.0764.0001, dst 10.0.1.129
FastEthernet0/1
R110d#
00:01:03: IP ARP: creating incomplete entry for IP address: 10.0.1.126 interface
FastEthernet0/0
00:01:03: IP ARP: sent reqsrc 10.0.1.1 c801.0764.0000,dst 10.0.1.126 0000.0000.0000
FastEthernet0/0
00:01:03: IP ARP: rcvd rep src 10.0.1.126 000c.2953.c7b7, dst 10.0.1.1
FastEthernet0/0
```

```
R110d#u all
All possible debugging has been turned off
R110d#sh iparp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  10.0.1.1                -   c801.0764.0000  ARPA   FastEthernet0/0
Internet  10.0.1.126              1   000c.2953.c7b7  ARPA   FastEthernet0/0
Internet  10.0.1.129              -   c801.0764.0001  ARPA   FastEthernet0/1
Internet  10.0.1.189              0   0050.56c0.0008  ARPA   FastEthernet0/1
Internet  10.0.1.222              1   c800.0764.0001  ARPA   FastEthernet0/1
Internet  10.0.1.170              0   c802.0764.0001  ARPA   FastEthernet0/1
Internet  10.0.1.180              0   c800.0764.0001  ARPA   FastEthernet0/1
R110d#
```

L'administrateur a pris soin de lancer une commande **debug arp**. Le processus debug arp informe de la diffusion d'une requête afin de découvrir l'adresse physique associée à l'adresse distante 10.0.1.222. Les deux routeurs R70d et R80d disposent chacun d'une route vers l'adresse objet de la requête. Chacun des deux routeurs répond à la requête ARP « je suis 10.0.1.222 ». Le processus ARP fait ce qu'il fait toujours quand il reçoit une correspondance ARP : il met à jour la correspondance dans le cache ARP. Dans le cas présent, la première réponse émane du routeur R70d et pendant un très court laps de temps, la correspondance placée dans le cache ARP de R110d est « 10.0.1.222 - C802.0764.0001 ». Mais une seconde réponse intervient presque immédiatement. Elle émane de R80d et provoque une nouvelle mise à jour du cache ARP qui contient désormais la correspondance « 10.0.1.222 - C800.0764.0001 ». R110d remettra les prochains datagrammes destinés au réseau LAN12 au routeur R80d. Il est dommage de constater que c'est le routeur ayant la plus grande latence qui devra se charger de faire progresser ces datagrammes.

Un routeur qui répond ainsi à une requête ARP avec sa propre adresse MAC se comporte en Proxy ARP, comportement adopté par défaut par l'IOS (il est possible de désactiver le Proxy ARP en entrant la commande **no ip proxy-arp** en configuration d'interface). Pour être complet, citons la possibilité d'effacer le contenu du cache ARP d'un routeur à l'aide de la commande **clear arp-cache**, en mode privilégié. Comment un administrateur (chevronné) reconnaît-il le comportement Proxy ARP ? En observant le contenu du cache ARP : une même adresse MAC correspond à plusieurs adresses IP.

En guise de conclusion, observons également une légère différence quant à la distance administrative entre une route statique avec adresse de saut suivant et une route statique avec interface de sortie. La distance administrative par défaut d'une route statique est 1. Mais une route statique avec interface de sortie est considérée par le routeur comme une route directement connectée, route dont la distance administrative est 0.

Une route statique avec interface de sortie vers un réseau LAN fonctionne à la condition que le comportement Proxy-ARP du ou des routeurs suivants n'ait pas été désactivé. Mais quand cette route est sollicitée, le choix du routeur suivant opéré par le mécanisme ARP n'est pas le plus pertinent. On réservera donc les routes statiques à interface de sortie aux liaisons point à point.

Résolution d’une route, la recherche récursive

Pour le processus de routage, une seule question mérite d’être posée : à quelle interface de sortie faut-il confier ce paquet ? Observez la table de routage ci-dessous :

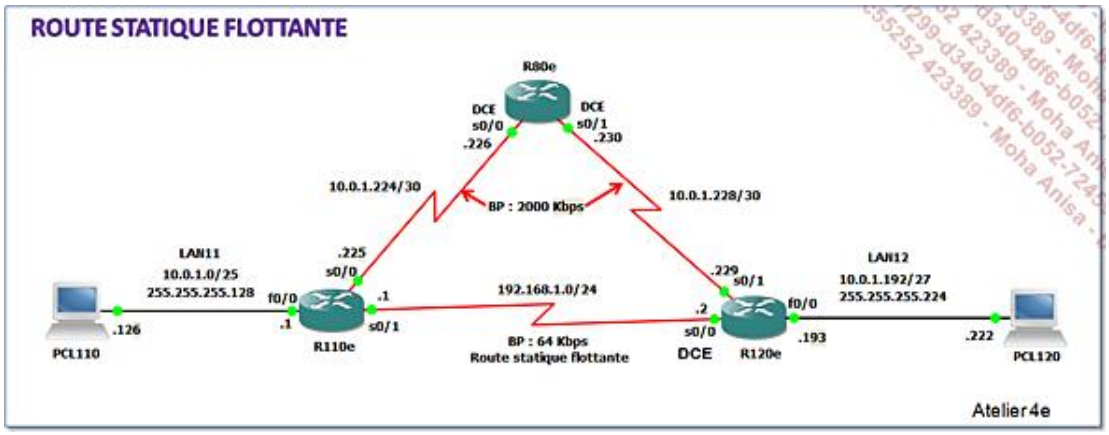
```
R110b#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 5 subnets, 5 masks
C       10.0.1.0/25 is directly connected, FastEthernet0/0
S       10.0.1.128/26 [1/0] via 10.0.1.226
S       10.0.1.192/27 [1/0] via 10.0.1.234
C       10.0.1.232/29 is directly connected, Serial0/1
C       10.0.1.224/30 is directly connected, Serial0/0
R110b#
```

Imaginons que R110b se voit confier un paquet destiné à l’hôte 10.0.1.129. Le routeur entame un premier balayage de la table de routage et trouve une solution dans la route 10.0.1.128/26 mais hélas, cette solution passe par 10.0.1.226. Et voilà notre routeur contraint d’entamer un second balayage de la table de routage à la recherche cette fois de 10.0.1.226. Le routeur trouve une route 10.0.1.224. Notez bien que cette route aurait pu mener à nouveau à une adresse de prochain saut, adresse qui nécessiterait un balayage supplémentaire mais notre routeur a plus de chance cette fois et la route fournit l’interface de sortie S0/0. Ce mécanisme de recherche qui consiste à enchaîner des balayages successifs de la table de routage jusqu’à trouver sur quelle interface acheminer le paquet est appelé recherche récursive.

Bien sûr, chaque nouvelle lecture de la table de routage dégrade le temps d’acheminement du paquet et le processus de routage est d’autant plus performant qu’il peut trouver rapidement quelle est l’interface de sortie adéquate. À ce point de vue, la route statique vers une interface de sortie trouve ici une justification supplémentaire.

Établissement d'une route statique flottante

Voici une stratégie qui intéressera certainement tous ceux pour qui la recherche de sûreté et de fiabilité frise l'obsession. Le contexte est le suivant : l'administrateur a confié le remplissage des tables de routage de l'ensemble de ses routeurs au protocole de routage EIGRP. Mais il souhaite prévoir le crash possible de ce processus. Si ce crash se produit, il doit assurer une connectivité minimale entre les deux routeurs R110e et R120e. Pour ce faire, il décide d'utiliser une route statique. Cette route doit rester inactive en temps normal mais doit se substituer à la route EIGRP en temps de crise :



Le lecteur pourra reproduire le contexte proposé sous GNS3. Les trois routeurs mettent en œuvre le protocole de routage EIGRP. Le réseau affecté au lien entre R110e et R120e n'est pas pris en compte par EIGRP. Puisque la distance administrative d'une route EIGRP est 90, la route statique a été dotée d'une distance administrative de 100. Quand les routeurs R110e et R120e installent dans la table de routage la route prévue par EIGRP, cette route passe par R80e.

Extrait de la configuration de R110e, la route statique est en gras :

```
R110e#sh run
Building configuration...

...
!
interface FastEthernet0/0
ip address 10.0.1.1 255.255.255.128
duplex auto
speed auto
!
interface Serial0/0
bandwidth 2000
ip address 10.0.1.225 255.255.255.252
!
interface Serial0/1
ip address 192.168.1.1 255.255.255.0
!
routereigrp 1
network 10.0.1.0 0.0.0.255
auto-summary
!
ip classless
ip route 10.0.1.192 255.255.255.224 192.168.1.2 100
ip http server
!
end

R110e#
```

Extrait de la configuration de R120e :

```
R120e#sh run
Building configuration...

...
!
```

```
interface FastEthernet0/0
ip address 10.0.1.193 255.255.255.224
duplex auto
speed auto
!
interface Serial0/0
ip address 192.168.1.2 255.255.255.0
clock rate 64000
!
interface Serial0/1
bandwidth 2000
ip address 10.0.1.229 255.255.255.252
!
routereigrp 1
network 10.0.1.0 0.0.0.255
auto-summary
!
ip classless
ip route 10.0.1.0 255.255.255.128 192.168.1.1 100
ip http server
!
end

R120e#
```

Extrait de la configuration de R80e :

```
R80e#sh run
Building configuration...

...
!
interface Serial0/0
bandwidth 2000
ip address 10.0.1.226 255.255.255.252
clock rate 2000000
!
interface Serial0/1
bandwidth 2000
ip address 10.0.1.230 255.255.255.252
clock rate 2000000
!
router eigrp 1
network 10.0.1.0 0.0.0.255
auto-summary
!
end

R80e#
```

La table de routage de R110e en temps normal, EIGRP opérationnel :

```
R110e#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
..........

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 4 subnets, 3 masks
C       10.0.1.0/25 is directly connected, FastEthernet0/0
D       10.0.1.192/27 [90/2306560] via 10.0.1.226, 00:16:11, Serial0/0
C       10.0.1.224/30 is directly connected, Serial0/0
D       10.0.1.228/30 [90/2304000] via 10.0.1.226, 00:16:11, Serial0/0
C    192.168.1.0/24 is directly connected, Serial0/1
R110e#
```

Reproduire le crash de la route EIGRP est simple, il suffit de suspendre le routeur R80e (sous GNS3, effectuez un clic droit sur le routeur puis sélectionnez **Suspendre**). Plusieurs messages SYSLOG sur les consoles de R110e et R120e

informent de la perte d'un voisin (patientez jusqu'à l'étude du protocole EIGRP) puis de la tombée du lien « serial » :

```
R110e#
00:19:54: %DUAL-5-NBRCHANGE: IP-EIGRP 1: Neighbor 10.0.1.226 (Serial0/0) is down:
holding time expired
R110e#
00:20:11: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state
to down
R110e#
```

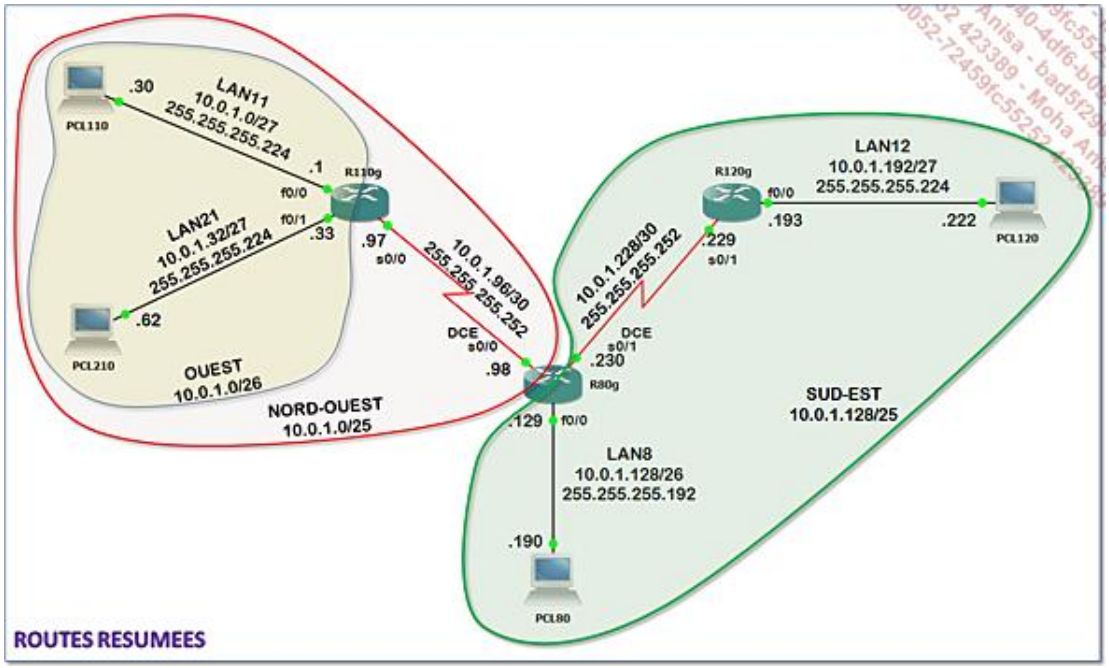
Une nouvelle commande **sh ip route** fait apparaître la route statique espérée. La route statique est revenue à la surface, c'est une route statique flottante :

```
R110e#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       10.0.1.0/25 is directly connected, FastEthernet0/0
S       10.0.1.192/27 [100/0] via 192.168.1.2
C    192.168.1.0/24 is directly connected, Serial0/1
R110e#
```

Routes résumées

Jusqu'à présent, dans les différentes mises en situation proposées, nous avons entré autant de routes statiques qu'il y avait de réseaux de destination. À l'évidence, en procédant de la sorte, les tables de routage vont rapidement s'alourdir et avec elles, la tâche des IOS. Mais puisque la commande permettant d'établir une route statique admet le masque en paramètre, il est tout indiqué chaque fois que c'est possible d'agréger plusieurs routes en une seule. Le chapitre dédié au découpage VLSM (*Variable Length Subnet Mask*) de l'ouvrage Notions de base sur les réseaux concluait qu'à la condition d'adopter une assignation des préfixes topologique, il devenait possible d'agréger les routes et par suite de réduire le volume des tables de routage.

Traitons un exemple concret :



Sans résumé de routes, inventorions les routes statiques nécessaires sur chacun des routeurs afin d'assurer une connectivité totale.

Sur R110g :

```
R110g(config)#ip route 10.0.1.128 255.255.255.192 10.0.1.98
R110g(config)#ip route 10.0.1.228 255.255.255.252 10.0.1.98
R110g(config)#ip route 10.0.1.192 255.255.255.224 10.0.1.98
```

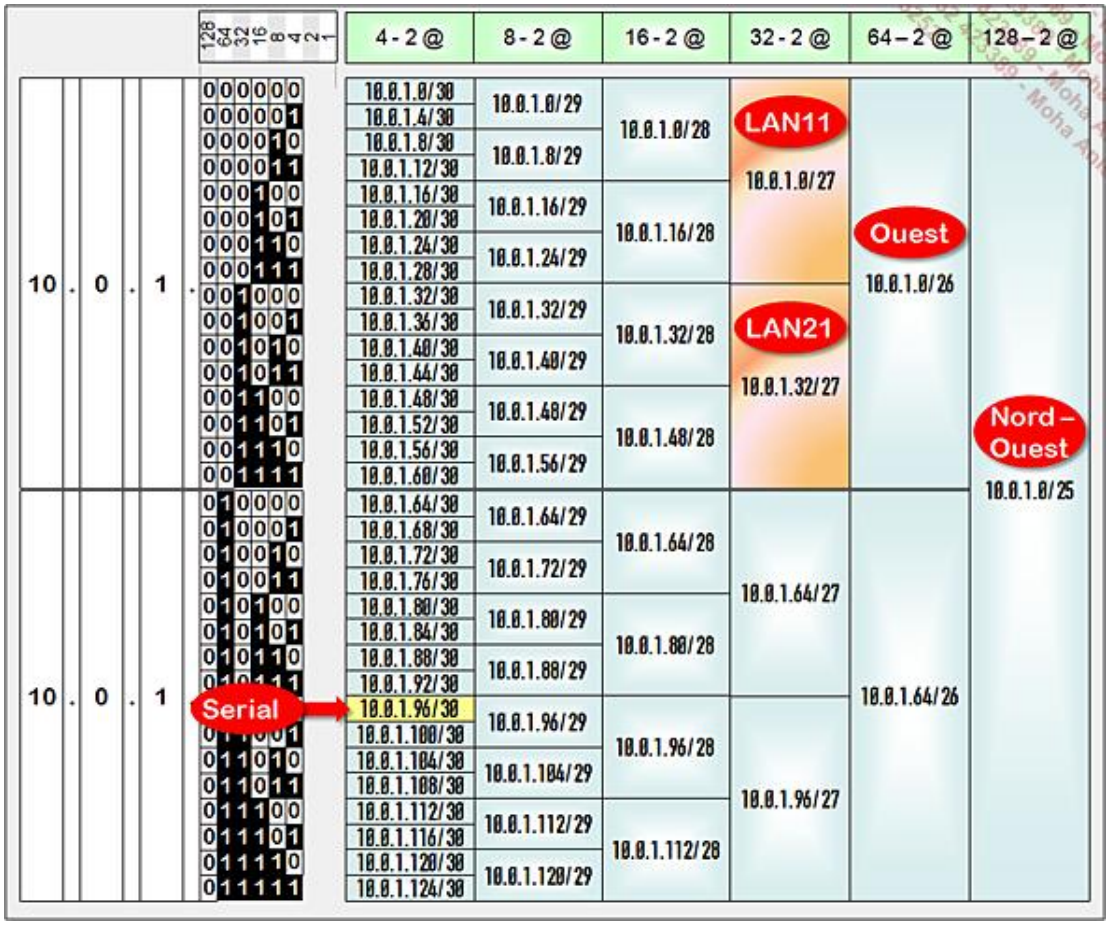
Sur R80g :

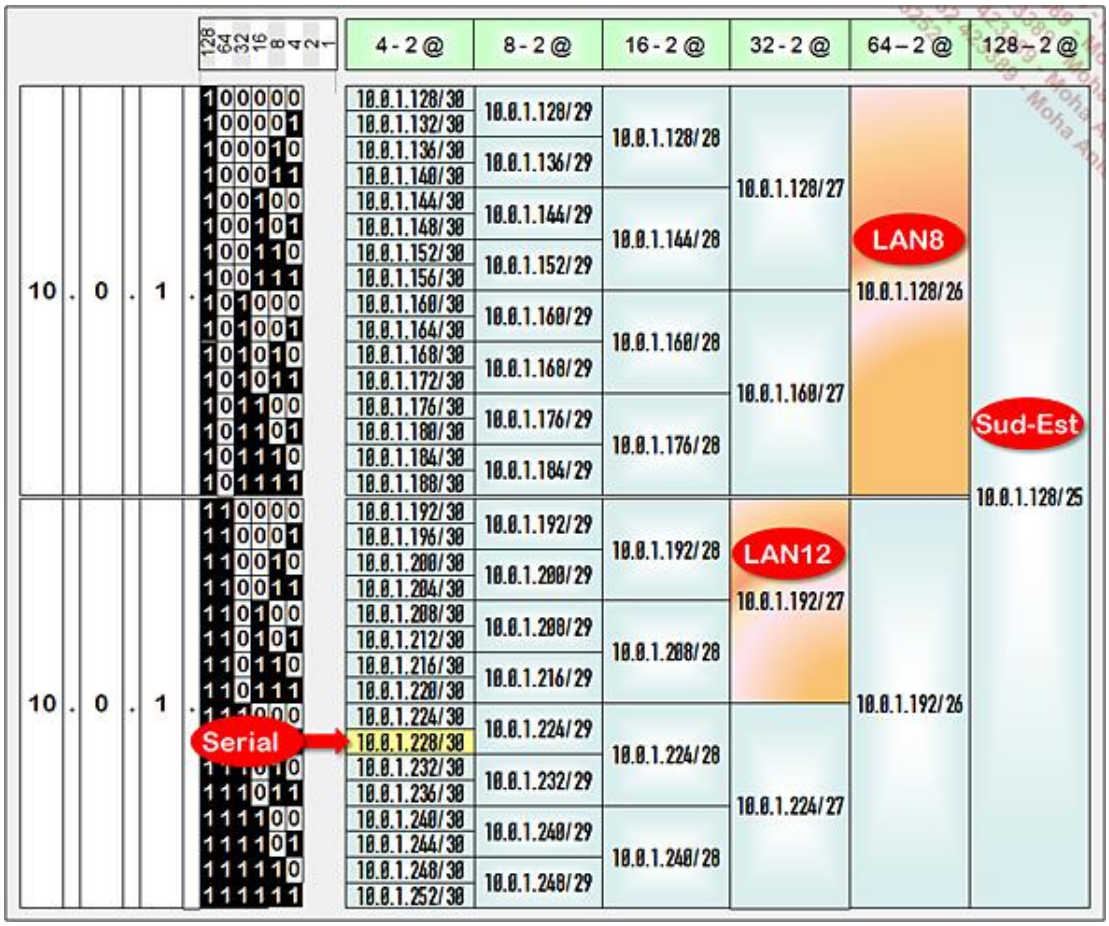
```
R80g(config)#ip route 10.0.1.0 255.255.255.224 10.0.1.97
R80g(config)#ip route 10.0.1.32 255.255.255.224 10.0.1.97
R80g(config)#ip route 10.0.1.192 255.255.255.224 10.0.1.229
```

Sur R120g :

```
R120g(config)#ip route 10.0.1.128 255.255.255.192 10.0.1.230
R120g(config)#ip route 10.0.1.224 255.255.255.252 10.0.1.230
R120g(config)#ip route 10.0.1.0 255.255.255.224 10.0.1.230
R120g(config)#ip route 10.0.1.32 255.255.255.224 10.0.1.230
```

L'administrateur en charge de ce réseau a réalisé la division de l'espace d'adressage 10.0.1.0/24 à l'aide d'un tableau VLSM dichotomotique depuis la longueur de préfixe 24 jusqu'à la longueur de préfixe maximale, soit 30 :





Avec un tel tableau, très visuel, l'agrégation devient un jeu d'enfant. R110g n'a besoin que d'une seule route vers le

réseau Sud-Est :

```
R110g(config)#ip route 10.0.1.128 255.255.255.128 10.0.1.98
```

R80g a besoin de deux routes, l'une vers le réseau Ouest, l'autre vers le réseau LAN12 :

```
R80g(config)#ip route 10.0.1.0 255.255.255.192 10.0.1.97
R80g(config)#ip route 10.0.1.192 255.255.255.224 10.0.1.229
```

R120g enfin, a besoin de deux routes, l'une vers le réseau LAN8, l'autre vers le réseau Nord-Ouest :

```
R120g(config)#ip route 10.0.1.128 255.255.255.192 10.0.1.230
R120g(config)#ip route 10.0.1.0 255.255.255.128 10.0.1.230
```

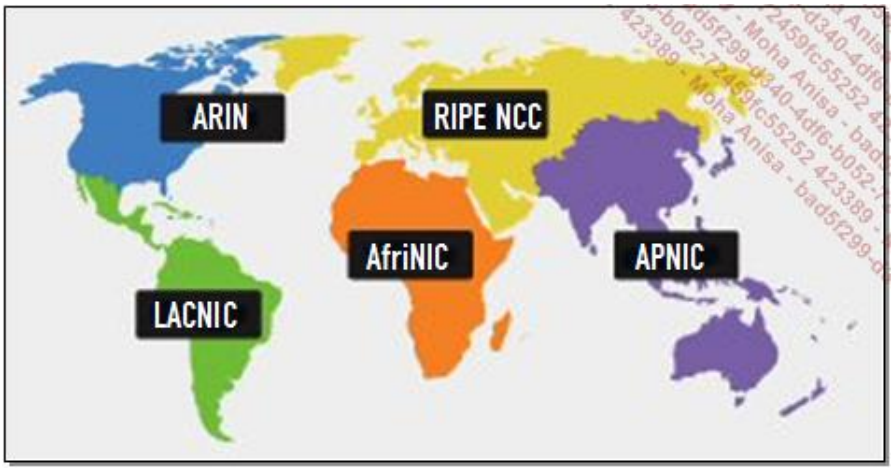
Ainsi, de 10 routes statiques, nous sommes passés à 5 routes. Attention aux conséquences de l'adressage topologique. Par exemple, nous avons décidé d'agréger le réseau Ouest 10.0.1.0/26 avec le réseau affecté au lien « serial » 10.0.1.96/30 pour aboutir au réseau Nord-Ouest 10.0.1.0/25. De fait, l'ensemble des adresses de l'espace 10.0.1.0/25 est plus vaste que l'ensemble constitué par le réseau Ouest additionné au réseau du lien série. Parmi les espaces non utilisés, l'espace 10.0.1.64/27 est disponible. Mais il n'est disponible que pour une future affectation dans l'espace Nord-Ouest. L'administrateur qui voudrait récupérer cet espace ailleurs, devrait limiter l'agrégation au réseau Ouest et modifier les routes sur R120g :

```
R120g(config)#ip route 10.0.1.128 255.255.255.192 10.0.1.230
R120g(config)#ip route 10.0.1.0 255.255.255.192 10.0.1.230
R120g(config)#ip route 10.0.1.96 255.255.255.252 10.0.1.230
```

La plate-forme Exploration de l'académie propose une méthode d'agrégation qui consiste à chercher la plus longue correspondance de préfixe pour exprimer la route agrégée. Cette méthode est très insuffisante devant le problème posé. Imaginons par exemple devoir exprimer la route agrégée qui mènerait aux réseaux 10.0.1.0/30 et 10.0.1.64/30. La plus longue correspondance de préfixe est 10.0.1.0/25. Mais 10.0.1.0/25 représente un espace de 128 adresses alors que les deux blocs 10.0.1.0/30 et 10.0.1.64/30 n'en représentent ensemble que 8. L'administrateur doit absolument mesurer l'étendue d'adresses effectivement couverte par une route agrégée sous peine de graves déconvenues.

Routes par défaut

On se souvient que c'est l'IANA (*Internet Assigned Numbers Authority*) qui est en charge de la gestion de l'espace d'adressage IP. Depuis CIDR (*Classless Inter-Domain Routing*), l'IANA a segmenté l'espace d'adressage en 256 blocs de taille /8 numérotés de 0/8 à 255/8. Chacun de ces blocs représente 16 millions d'adresses (24 bits). Puis, l'IANA délègue l'administration de ces segments à cinq RIR (*Regional Internet Registry*) selon la cartographie suivante :



Les adresses allouées pour l'Europe sont gérées par le RIPE NCC (Réseaux IP Européens - Network Coordination Centre, www.ripe.net). Les segments délégués par l'IANA au RIPE NCC sont au nombre de trente, on peut découvrir quels sont ces segments en consultant le document http://www.iana.org/assignments/ipv4-address-space/. Pour un routeur de l'Internet situé aux Etats-Unis, router un paquet vers l'Europe revient à découvrir que le premier octet de l'adresse IP de destination appartient à l'une des trente valeurs possibles pour l'Europe (62, 77 à 95, 141, 145, 151, 188, 193 à 195, 212, 213, 217).

Ce petit préambule afin de poser une question : quelle est l'agrégation maximale et quelle est sa représentation ?

Par exemple, le bloc 128/8 représente 16 millions d'adresses et est affecté à l'ARIN. Le bloc 128/7 comprend les blocs 128/8 et 129/8. Le bloc 128/6 contient les blocs 128/8, 129/8, 130/8 et 131/8. Plus la longueur de préfixe diminue et plus l'espace d'adresses correspondant est grand. Quels sont les préfixes de longueur 1 ? Ils sont au nombre de deux, l'espace 0/1 et l'espace 1/1. Chacun représente deux milliards d'adresses. Comment agréger 0/1 et 1/1 ? Par l'espace 0/0. Ainsi, 0/0 représente la totalité de l'espace d'adressage IPv4 soit 4 milliards d'adresses (2^{32}). Ce n'est pas une représentation conventionnelle comme on peut le lire dans divers documents, mais bien la représentation CIDR de l'espace total.

Le réseau 0/0 est le réseau total. Son adresse est 0.0.0.0 masque 0.0.0.0.

---

1. Route par défaut statique

Puisque le réseau 0/0 représente l'ensemble des adresses, toute adresse de destination appartient à ce réseau. Une route qui pointe vers le réseau 0/0 est une route que tous les datagrammes peuvent emprunter faute de disposer d'une route plus spécifique. Quand le routeur ajoute cette route dans la table de routage, elle prend le nom de route par défaut.

Une route par défaut est toute indiquée quand un routeur est utilisé pour connecter un réseau local au reste du monde. En effet, dans ce cas, toute requête émise par l'un des hôtes du réseau local et destinée au reste du monde doit transiter par ce routeur, le chemin est unique. Dans la mise en situation précédente, le routeur R120g répond au critère requis pour tirer avantage d'une route par défaut. L'administrateur décide de remplacer les routes statiques agrégées par une route par défaut :

```
R120g#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R120g(config)#no ip route 10.0.1.0 255.255.255.128 10.0.1.230
R120g(config)#no ip route 10.0.1.128 255.255.255.192 10.0.1.230
R120g(config)#ip route 0.0.0.0 0.0.0.0 10.0.1.230
R120g(config)#exit
R120g#
00:05:37: %SYS-5-CONFIG_I: Configured from console by console
R120g#copy run start
Destination filename [startup-config]?
```

```
Building configuration...
[OK]
R120g#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........
ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 10.0.1.230 to network 0.0.0.0

     10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       10.0.1.192/27 is directly connected, FastEthernet0/0
C       10.0.1.228/30 is directly connected, Serial0/1
S*   0.0.0.0/0 [1/0] via 10.0.1.230
R120g#
```

Dans le résultat de la commande **show ip route**, observez le choix opéré par le routeur pour élire la passerelle de dernier recours « Gateway of last resort » associée à l'adresse 10.0.1.230. Dans le cas présent, l'élection est simple puisque la seule route candidate est celle entrée de façon statique. Mais le routeur peut avoir connaissance de plusieurs routes par défaut apprises, pour partie d'une configuration statique, pour partie d'un protocole de routage. Les routes candidates sont marquées par une étoile dans la table de routage, la passerelle de dernier recours choisie est celle correspondant à la route candidate dont la distance administrative est la plus faible.

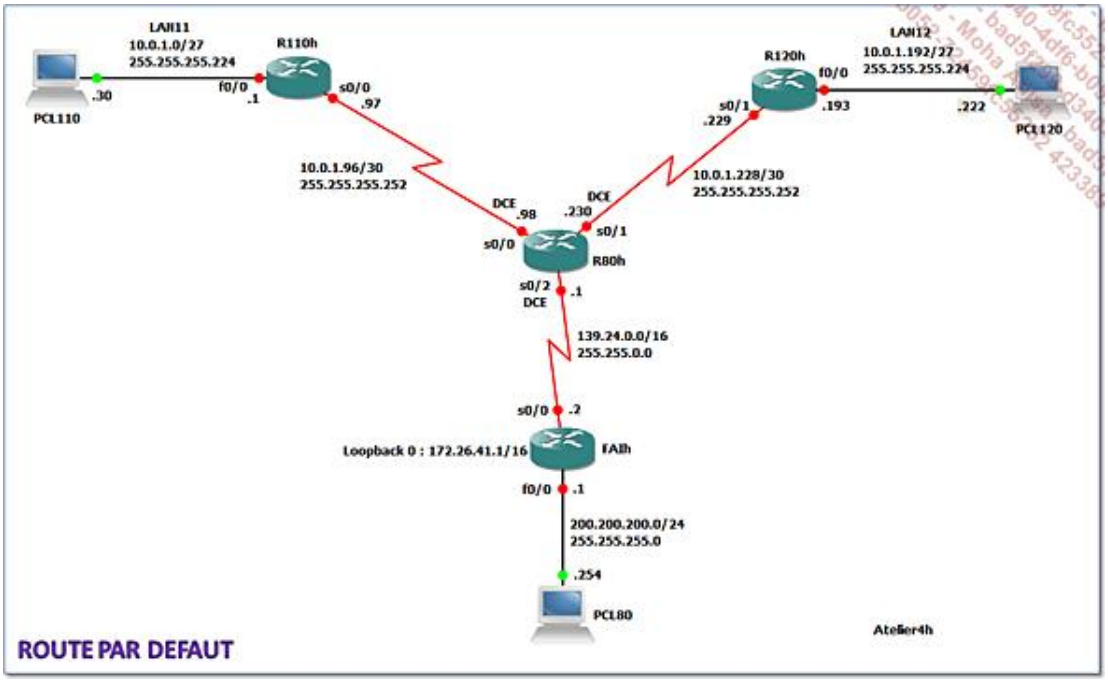
2. La commande ip default-gateway

L'IOS propose deux autres commandes en rapport avec l'établissement d'une route par défaut : **ip default-network** et **ip default-gateway**. Réglons de suite le cas de la commande **ip default-gateway**, très différente des deux commandes **ip route** et **ip default-network**. Il arrive qu'un routeur doive être considéré, temporairement, comme un simple hôte, c'est-à-dire un système d'extrémité. C'est le cas par exemple, lorsque dans le mode boot, il faut charger une image IOS depuis un serveur TFTP. Quelle est la différence entre le comportement d'un hôte et le comportement d'un routeur ? Seul le second dispose d'un processus de routage IP actif. L'hôte quant à lui, ne peut classer les adresses de destination qu'en deux catégories : interne auquel cas le paquet peut être remis directement ; externe et dans ce cas, le paquet doit être confié à la passerelle par défaut. C'est précisément le rôle de la commande **ip default-gateway** que de configurer l'adresse de cette passerelle.

La commande ip default-gateway permet de définir la passerelle par défaut pour un routeur dont le processus de routage IP n'est pas actif.

3. La commande ip default-network

Montrer la mise en œuvre de la commande **ip default-network** nécessite un contexte un peu plus élaboré. Examinez la topologie proposée ci-après :



Les trois routeurs R110h, R120h et R80h mettent en œuvre un protocole de routage. L'objectif est d'établir une route par défaut vers le routeur FAIh sur le routeur R80h. L'administrateur serait évidemment très satisfait de voir cette route par défaut se propager de façon automatique sur les routeurs R110h et R120h. Le comportement des protocoles de routage vis-à-vis des routes par défaut reste à étudier, d'autres détails seront fournis lors de l'étude de ces protocoles.

Extrait de la configuration de R80h :

```
!
interface Serial0/0
ip address 10.0.1.98 255.255.255.252
clock rate 64000
!
interface Serial0/1
ip address 10.0.1.230 255.255.255.252
clock rate 64000
!
interface Serial0/2
ip address 139.24.0.1 255.255.0.0
clock rate 64000
!
router rip
version 2
network 10.0.0.0
default-information originate
!
ip classless
ip default-network 172.26.0.0
ip route 172.26.0.0 255.255.0.0 139.24.0.2
ip http server

R80h#sh ip route
```

Extrait de la configuration de FAIh :

```
!
interface Loopback0
ip address 172.26.41.1 255.255.0.0
!
interface FastEthernet0/0
ip address 200.200.200.1 255.255.255.0
duplex auto
speed auto
!
```

```
interface Serial0/0
ip address 139.24.0.2 255.255.0.0
!
ip classless
ip route 10.0.1.0 255.255.255.0 139.24.0.1
ip http server
!
FAIh#
```

La table de routage de R80h :

```
R80h#sh ip route

Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is 139.24.0.2 to network 172.26.0.0

C    139.24.0.0/16 is directly connected, Serial0/2
S*   172.26.0.0/16 [1/0] via 139.24.0.2
     10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
R       10.0.1.0/27 [120/1] via 10.0.1.97, 00:00:25, Serial0/0
C       10.0.1.96/30 is directly connected, Serial0/0
R       10.0.1.192/27 [120/1] via 10.0.1.229, 00:00:22, Serial0/1
C       10.0.1.228/30 is directly connected, Serial0/1
R80h#
```

La table de routage de R110h :

```
R110h#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is 10.0.1.98 to network 0.0.0.0

     10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C       10.0.1.0/27 is directly connected, FastEthernet0/0
C       10.0.1.96/30 is directly connected, Serial0/0
R       10.0.1.192/27 [120/2] via 10.0.1.98, 00:00:09, Serial0/0
R       10.0.1.228/30 [120/1] via 10.0.1.98, 00:00:09, Serial0/0
R*   0.0.0.0/0 [120/1] via 10.0.1.98, 00:00:09, Serial0/0
R110h#
```

La table de routage de R120h :

```
R120h#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is 10.0.1.230 to network 0.0.0.0

     10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
R       10.0.1.0/27 [120/2] via 10.0.1.230, 00:00:05, Serial0/1
R       10.0.1.96/30 [120/1] via 10.0.1.230, 00:00:05, Serial0/1
C       10.0.1.192/27 is directly connected, FastEthernet0/0
C       10.0.1.228/30 is directly connected, Serial0/1
R*   0.0.0.0/0 [120/1] via 10.0.1.230, 00:00:05, Serial0/1
R120h#
```

La table de routage de FAIh :

```
FAIh#ship route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is not set

C    200.200.200.0/24 is directly connected, FastEthernet0/0
C    139.24.0.0/16 is directly connected, Serial0/0
```

```
C    172.26.0.0/16 is directly connected, Loopback0
     10.0.0.0/24 is subnetted, 1 subnets
S       10.0.1.0 [1/0] via 139.24.0.1
FAIh#
```

Un point intéressant est l'utilisation de l'interface virtuelle **loopback** sur le routeur FAIh. Cette faculté de créer des interfaces virtuelles rend de nombreux services dans la configuration des routeurs CISCO. Par sa nature indépendante des aspects physiques, une interface virtuelle est à l'état monté tant que l'IOS est fonctionnel. En lui affectant une adresse IP, on crée une route directement connectée. Des paquets destinés à ce réseau directement connecté seraient simplement éliminés par le routeur, voilà un moyen de faire du filtrage à peu de frais (peu de consommation de ressources machine). Il est également fréquent de mettre en place une interface de loopback pour que l'adresse IP affectée à l'interface identifie le routeur.

Dans le cas présent, l'administrateur souhaite sur R80h créer une route par défaut via le routeur FAIh. Pour ce faire, la commande **ip default-network** reçoit en paramètre l'adresse du réseau connecté à l'interface virtuelle de FAIh, soit 172.16.0.0. Puisque R80h connaît une route statique vers ce réseau, il fait de cette route une candidate pour le choix de la passerelle de dernier recours (voir l'étoile dans la table de routage de R80h). Il n'y a pas d'autres routes candidates, l'adresse 139.24.0.2 devient passerelle de dernier recours.

4. Influence du routage sans classe sur la route par défaut

Le routage avec classe est un héritage du passé. IPv4 n'aurait pas pu survivre si le système avec classe avait été maintenu de par la pénurie des adresses et l'explosion des tables de routage des routeurs de l'Internet. Ce sujet fait l'objet d'un exposé sérieux dans l'ouvrage Cisco - Notions de base sur les réseaux dans la collection Certifications aux Editions ENI - La couche réseau, l'adressage IP. L'adoption de CIDR (*Classless Inter Domain Routing*) en 1996 s'est traduite par la nécessité pour les constructeurs de routeurs de proposer des matériels et donc des IOS capables de contribuer au routage dans des réseaux avec classe (l'existant) et dans des réseaux sans classe (migration). L'IOS CISCO est capable de fonctionner suivant les deux modes, le choix s'opérant à l'aide de la commande, passée inaperçue jusqu'ici, **ip classless** (observez les extraits de configuration précédents).

Plaçons-nous sur R80h et imaginons que ce routeur doive traiter un paquet dont l'adresse de destination est 10.0.1.65. Cette adresse appartient au bloc 10.0.1.64/27. R80h dispose de quatre routes vers des fragments de l'espace 10.0.1.0/24 qui sont 10.0.1.0/27, 10.0.1.96/30, 10.0.1.192/27 et 10.0.1.228/30. R80h n'a pas de route vers 10.0.1.65, la commande **ip classless** est active, le routeur décide de faire emprunter la route par défaut, le paquet est confié à la passerelle de dernier recours.

Mais si la commande **no ip classless** est entrée, le comportement est différent et proche de l'étrange. Puisque le routeur connaît des routes vers une partie de l'adresse de classe A 10.0.0.0/8, un paquet destiné à l'adresse 10.0.1.65 est mis au rebut. Le routeur utilise la route par défaut à la condition de ne rien connaître du réseau de destination avec classe. Par exemple, un paquet destiné à l'adresse 11.0.1.65 transiterait bien par la route par défaut, cette adresse appartient au réseau de classe A 11.0.0.0 dont R80h n'a aucune connaissance.

Partage de charge en routage statique

1. Qu'est-ce que le partage de charge ?

Le partage de charge (*Load balancing*) consiste à répartir le trafic vers une même destination sur plusieurs chemins. Un routeur peut automatiquement réaliser du partage de charge à la condition de disposer de plusieurs routes vers la même destination dans sa table de routage.

Entendons-nous bien sur la notion de même destination : il s'agit d'adresses de destination ayant le même préfixe. Ainsi, la destination 10.0.1.0/26 n'est pas identique à la destination 10.0.1.0/24. La première est plus précise, la seconde plus agrégée.

Le partage de charge peut être à coût égal ou à coût inégal, le terme *coût* fait référence à la métrique associée à la route :

- À coût égal : le trafic est également réparti sur plusieurs routes de métriques identiques.
- À coût inégal : le trafic est réparti sur plusieurs routes de métriques différentes. La part de trafic transportée par chacune des routes est inversement proportionnelle à sa métrique (plus la métrique est faible, synonyme de bande passante élevée, plus la part de trafic absorbé est importante).

Il faut rappeler le comportement des protocoles de routage dans ce contexte. Par essence, un protocole de routage cherche à déterminer la meilleure route qui souvent devient la route unique, l'objectif du protocole est donc un peu en contradiction avec la recherche de répartition de charge. RIP, OSPF et IS-IS ne supportent que le partage de charge à coût égal. Autrement dit, ces protocoles n'installent plusieurs routes dans la table de routage que si ces routes ont même métrique. EIGRP et BGP admettent quant à eux le partage de charge à coût inégal.

Par principe, puisque toutes les routes statiques ont même métrique (métrique nulle), le routage statique ne supporte que le partage de charge à coût égal. Nous verrons un peu plus loin comment contourner ce principe et avec un peu d'astuce, comment réaliser un partage de charge à coût inégal sur plusieurs routes statiques. Mais attention, si le lecteur est confronté à une question portant sur ce sujet dans l'un des nombreux QCM qui émaillent la route vers la certification, la réponse est claire : routes statiques impliquent partage de charge à coût égal.

À ce stade, il faut expliciter à minima quels peuvent être les différents comportements du routeur qui doit réaliser un partage de charge. Par défaut, le routeur adopte un comportement dit « Partage par destination ». Mais l'administrateur qui le souhaite peut revenir à un comportement dit « Partage par paquet ».

a. Partage de charge par destination et « Fast Switching »

La clé de répartition est fournie par l'adresse de destination. Imaginons deux routes pour le même réseau de destination. Tous les datagrammes destinés à une première adresse de ce réseau empruntent la première route, tous les datagrammes destinés à une seconde adresse empruntent la seconde route, tous les datagrammes destinés à une troisième adresse empruntent à nouveau la première route et ainsi de suite. Ce mode de répartition est appelé « Fast Switching » par CISCO, c'est le mode de commutation par défaut.

Plus dans le détail, imaginons le premier paquet d'un flux vers une nouvelle adresse de ce réseau pour lequel il existe deux routes. Le routeur consulte sa table de routage et détermine une interface de sortie. Le datagramme est ensuite remis au pilote de l'interface réseau afin d'être encapsulé dans une nouvelle trame. Le pilote doit disposer de l'adresse physique de l'interface en vis-à-vis, pas de problème dans le cas d'une liaison « serial » (WAN, point à point), peut-être la nécessité de lancer une résolution ARP dans le cas d'une interface LAN. Une fois l'adresse physique de destination connue, le pilote compose la trame et la transmet. C'est là que le routeur se montre intelligent en plaçant dans un cache, appelons le cache de commutation rapide, la correspondance qu'il vient de trouver @destination - Interface de sortie - @physique du correspondant. Les paquets suivants vers la même destination sont identifiés immédiatement grâce à une consultation du cache et confiés directement au pilote de l'interface de sortie, c'est pourquoi on peut parler de commutation rapide. La recherche dans la table de routage ainsi que la recherche dans le cache ARP sont devenues superflues.

Bien sûr, la répartition de charge sur l'ensemble des chemins ne peut s'opérer de façon équitable que si le nombre de destinations est suffisant.

b. Partage de charge par paquet et « Process Switching »

L'algorithme est confondant de simplicité : le premier paquet destiné à ce réseau objet de plusieurs routes emprunte la première route. Le paquet suivant emprunte la route suivante et ainsi de suite pour obtenir un partage de charge à coût égal. Mais le routeur peut aussi changer cette répartition lorsqu'il faut réaliser un partage de charge à coût inégal. Par exemple, un paquet sur la première route, deux paquets sur la seconde, à nouveau un sur la première et ainsi de suite pour réaliser une répartition 1/3 - 2/3 ou toute autre répartition pour s'ajuster au

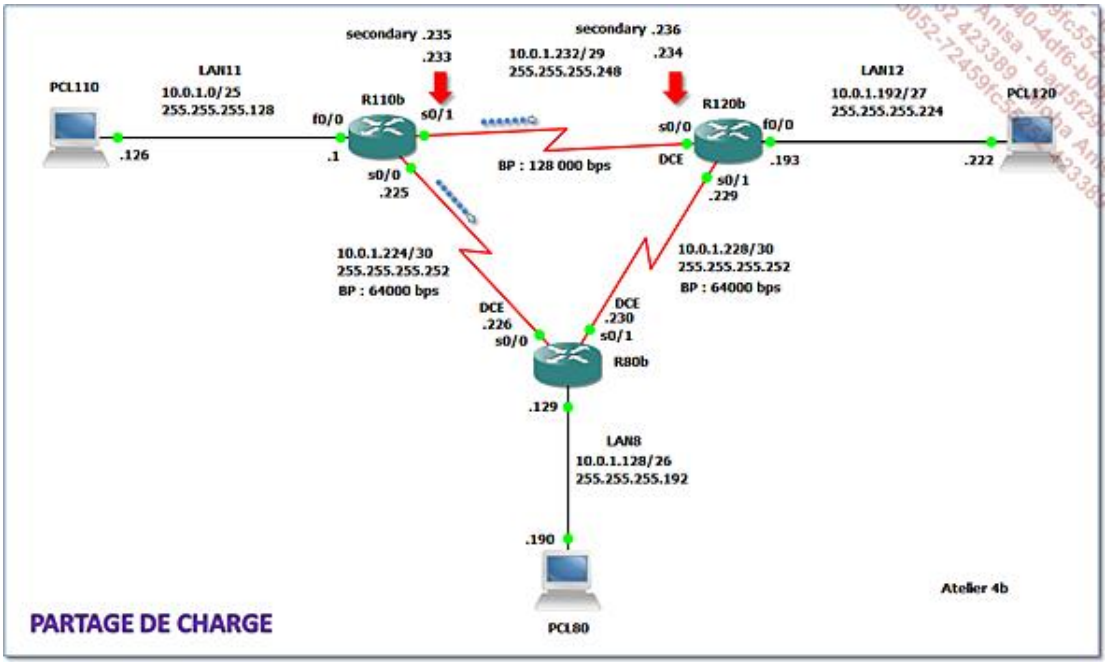
prorata des métriques de chaque route. CISCO nomme « *Process Switching* » ce mode de commutation.

Dans ce mode « *Process Switching* », pour chacun des paquets, le routeur doit consulter la table de routage, sélectionner une interface de sortie puis retrouver l'adresse physique du correspondant. Ceci entraîne qu'un flux de paquets destiné à une même adresse emprunte plusieurs interfaces de sortie.

On perçoit bien que la performance est du côté du mode de commutation « *Fast Switching* ». Mais notre souhait dans un environnement didactique est de voir la répartition de charge s'opérer, ce qui est possible à l'aide d'une commande **debug ip packet**. Il faut au préalable désactiver le mode de commutation rapide au profit du mode « *Process Switching* », ce à l'aide de la commande **no ip route-cache** en configuration d'interface. Attention à la commande **debug ip packet** qui est un peu la commande « de la mort » car elle contraint l'IOS à afficher tous les paquets. Cette commande est quasi exclue dans un environnement de production. Dans tous les cas, l'administrateur prendra garde à entrer d'abord la commande **undebug all** (**u all** de façon abrégée), ce qui la place dans l'historique de commandes. Le peu de ressources processeur encore disponible après l'exécution de la commande **debug ip packet** sera peut-être encore suffisant pour rappeler la commande **u all** depuis l'historique ([Flèche en haut] puis [Entrée]).

2. Partage de charge à coût égal

Nous appuierons notre propos sur le contexte suivant, à nouveau réalisé à l'aide de GNS3, qui décidément aura rendu bien des services tant il permet de multiplier les mises en situation :



Deux routes existent qui relient R110b et R120b, l'une passe par R80b, l'autre est directe. On décide d'établir les routes statiques correspondantes afin de répartir la charge pour moitié sur la route directe, pour l'autre moitié sur la route qui passe par R80b.

Extrait de la configuration de R110b :

```
R110b#sh run
Building configuration...

...
!
interface FastEthernet0/0
ip address 10.0.1.1 255.255.255.128
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.1.225 255.255.255.252
noip route-cache
noipmroute-cache
!
interface Serial0/1
ip address 10.0.1.233 255.255.255.248
```

```
noip route-cache
noipmroute-cache
!
ip classless
ip route 10.0.1.128 255.255.255.192 10.0.1.226
ip route 10.0.1.192 255.255.255.224 10.0.1.226
ip route 10.0.1.192 255.255.255.224 10.0.1.234
ip http server
!
...
end

R110b#
```

L'administrateur n'oublie pas de désactiver le mode de commutation rapide sur les interfaces s0/0 et s0/1 de R110b :

```
R110b#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R110b(config)#
R110b(config)#int S0/0
R110b(config-if)#no ip route-cache
R110b(config-if)#int s0/1
R110b(config-if)#no ip route-cache
R110b(config-if)#^Z
R110b#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
```

La configuration de R120b est calquée sur celle de R110b. L'administrateur lance le **debug ip packet** puis depuis PCL110, lance la commande **ping** vers PCL120 :

```
R110b#undebug all
All possible debugging has been turned off
R110b#debug ip packet
IP packet debugging is on
R110b#
00:22:10: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222
(Serial0/1),routed via RIB
00:22:10: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
g=10.0.1.234, len 84, forward
00:22:11: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222
(Serial0/0),routed via RIB
00:22:11: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/0),
g=10.0.1.226, len 84, forward
00:22:12: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
routed via RIB
00:22:12: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
g=10.0.1.234, len 84, forward
00:22:13: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222
(Serial0/0),routed via RIB
00:22:13: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/0),
g=10.0.1.226, len 84, forward
00:22:14: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
routed via RIB
00:22:14: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
g=10.0.1.234, len 84, forward
00:22:15: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222
(Serial0/0),routed via RIB
00:22:15: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/0),
g=10.0.1.226, len 84, forward
00:22:16: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
routed via RIB
00:22:16: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
g=10.0.1.234, len 84, forward
00:22:17: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/0),
routed via RIB
00:22:17: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/0),
```

```
g=10.0.1.226, len 84, forward
R110b#undebug all
All possible debugging has been turned off
R110b#
```

Les requêtes ICMP sont réparties sur les deux routes dans une alternance parfaite, CQFD.

3. Partage de charge à coût inégal

La liaison directe entre R110b et R120b a une bande passante de 128 Kbps. Celle passant par R80b n'offre que la moitié de cette bande soit 64 Kbps. Répartir le débit en tenant compte des bandes passantes effectives des deux chemins suppose que pour trois paquets, deux transitent par la route directe, un seul par la route via R80b. Puisque le routeur se fonde sur les routes pour répartir le trafic, l'idée est de créer deux routes directes et une route seulement passant par R80b. Pour le routeur, il s'agit toujours de partage de charge à coût égal, mais pour l'administrateur, le trafic est effectivement réparti au prorata des bandes passantes. La question devient : « Comment créer plusieurs routes sur un même lien physique ? ».

La solution consiste à affecter non pas une, mais deux adresses IP à la même interface. On ne change rien à la façon d'assigner la première adresse IP à l'aide de la commande **ip address** en configuration d'interface. La seconde adresse (et les suivantes si nécessaire) est affectée à l'aide de la même commande à laquelle on ajoute le mot-clé **secondary**. L'administrateur a affecté le réseau 10.0.1.232/29 au lien « serial » reliant R110b et R120b. Ce réseau comporte quatre adresses .233, .234, .235, 236 dont deux ont été affectées à l'interface s0/1 de R110b et deux à l'interface s0/0 de R120b. Ceci permet à l'administrateur de porter à deux le nombre de routes statiques pointant vers LAN12 sur R110b.

Voici les modifications apportées à la configuration de R110b :

```
R110b#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R110b(config)#int s0/1
R110b(config-if)#ip address 10.0.1.235 255.255.255.248 secondary
R110b(config-if)#exit
R110b(config)#ip route 10.0.1.192 255.255.255.224 10.0.1.236
R110b(config)#^Z
R110b#co
02:24:49: %SYS-5-CONFIG_I: Configured from console by console
R110b#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
R110b#
```

Puis celles apportées à la configuration de R120b :

```
R120b#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R120b(config)#int S0/0
R120b(config-if)#ip address 10.0.1.236 255.255.255.248 secondary
R120b(config-if)#exit
R120b(config)#ip route 10.0.1.0 255.255.255.128 10.0.1.235
R120b(config)#^Z
R120b#
02:27:21: %SYS-5-CONFIG_I: Configured from console by console
R120b#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
R120b#
```

Un coup d'œil à la table de routage de R110b permet de vérifier qu'il existe bien désormais trois routes vers LAN12, deux via le lien direct entre R110b et R120b, une via R80b :

```
R110b#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 5 subnets, 5 masks
```

```
C       10.0.1.0/25 is directly connected, FastEthernet0/0
S       10.0.1.128/26 [1/0] via 10.0.1.226
S       10.0.1.192/27 [1/0] via 10.0.1.234
                      [1/0] via 10.0.1.226
                      [1/0] via 10.0.1.236
C       10.0.1.232/29 is directly connected, Serial0/1
C       10.0.1.224/30 is directly connected, Serial0/0
R110b#
```

De même sur R120b :

```
R120b#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 5 subnets, 5 masks
S       10.0.1.0/25 [1/0] via 10.0.1.233
                    [1/0] via 10.0.1.230
                    [1/0] via 10.0.1.235
S       10.0.1.128/26 [1/0] via 10.0.1.30
C       10.0.1.192/27 is directly connected, FastEthernet0/0
C       10.0.1.232/29 is directly connected, Serial0/0
C       10.0.1.228/30 is directly connected, Serial0/1
R120b#
```

À nouveau, l'administrateur lance la commande **debug ip packet** puis depuis PCL110, lance la commande **ping** vers PCL120 :

```
R110b#undebug all
All possible debugging has been turned off
R110b#debug ip packet
IP packet debugging is on
R110b#
02:34:56: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/0),
routed via RIB
02:34:56: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/0),
g=10.0.1.226, len 84, forward
02:34:57: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
routed via RIB
02:34:57: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
g=10.0.1.236, len 84, forward
02:34:58: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
routed via RIB
02:34:58: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
g=10.0.1.234, len 84, forward
02:34:59: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/0),
routed via RIB
02:34:59: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/0),
g=10.0.1.226, len 84, forward
02:35:00: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
routed via RIB
02:35:00: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
g=10.0.1.236, len 84, forward
02:35:01: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
routed via RIB
02:35:01: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
g=10.0.1.234, len 84, forward
02:35:02: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/0),
routed via RIB
02:35:02: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/0),
g=10.0.1.226, len 84, forward
02:35:03: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
routed via RIB
02:35:03: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
g=10.0.1.236, len 84, forward
02:35:04: IP: tableid=0, s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
```

```
routed via RIB
02:35:04: IP: s=10.0.1.126 (FastEthernet0/0), d=10.0.1.222 (Serial0/1),
g=10.0.1.234, len 84, forward
R110b#
```

Et miracle, cette fois pour deux paquets vers LAN12 qui empruntent le lien direct, un seul emprunte la route via R80b. Il resterait à rétablir le mode de commutation rapide sur R110b et R120b.

La répartition obtenue est 1/3 - 2/3 mais d'autres répartitions sont possibles à la condition de ne pas dépasser six routes vers la même destination : 1/4 - 3/4 ; 1/5 - 4/5 ; 2/5 - 3/5 ; 1/6 - 5/6. En effet, les versions actuelles de l'IOS admettent jusqu'à six routes vers la même destination en routage statique (quatre par défaut pour des routes qui seraient issues d'un protocole de routage, une seule par défaut pour le protocole de routage BGP, chiffres qu'il est possible de changer à l'aide de la commande **maximum-paths** en mode de configuration de routeur, sans toutefois que le nombre de routes pour une même destination ne puisse dépasser six).

Synthèse

Les notions suivantes devraient être très sérieusement ancrées :

- Le routeur n'ajoute une route à la table de routage que s'il a connaissance de l'état « up » du saut suivant. Si le saut suivant n'est plus joignable (panne du routeur suivant, problème de liaison...), la route est retirée de la table.

- Des préfixes différents donnent des routes différentes. Par exemple, le réseau 10.0.1.0/25 est bien inclus dans 10.0.1.0/24. Pourtant, si l'occasion est donnée de découvrir une route vers chacun de ces préfixes, alors les deux routes sont ajoutées à la table quelle que soit leur distance administrative.

- S'il existe plusieurs routes vers la même destination (même préfixe), le routeur ajoute à la table la route dont la distance administrative est la plus faible, les autres routes sont ignorées mais pas perdues. Si la route retenue venait à disparaître, l'une des routes ignorées jusque-là pourrait s'y substituer.

- Pour une même distance administrative, c'est-à-dire un même mode d'apprentissage, la route préférée est celle dont la métrique est la plus faible.

- Une route statique de type à adresse de saut suivant est caractérisée par une distance administrative de 1 et une métrique égale à 0.

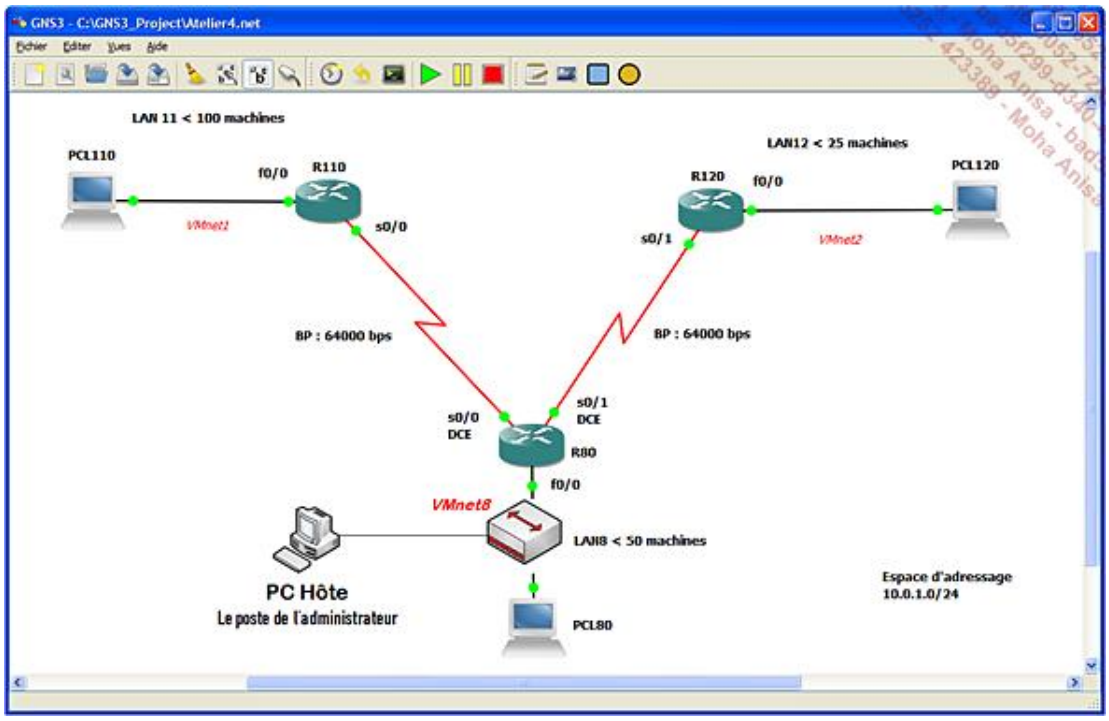
- Une route directement connectée est caractérisée par une distance administrative égale à 0.

- Quand une route fournit une adresse de saut suivant, le processus de routage est contraint de balayer à nouveau la table de routage. Cette recherche dite "récursive" dans la table de routage ne prend fin que lorsque la route fournit une interface de sortie.

- Quand deux routes ou davantage existent vers la même destination (même préfixe) et sont dotées de la même distance administrative, alors ces routes coexistent dans la table de routage. Le routeur répartit le trafic vers cette destination sur l'ensemble des routes, réalisant ainsi un partage de charge à coût égal.

TP : Mise en œuvre d'un routage statique

Proposition de topologie :



1. Tâche 1 : Conception du plan d'adressage

- Vous êtes l'administrateur du réseau ci-dessus et avez obtenu le préfixe 10.0.1.0/24. Déterminez la division en sous-réseaux convenable afin de satisfaire les besoins des réseaux LAN11, LAN8 et LAN12 dans cet ordre. Pour chaque réseau, affectez la première adresse disponible à l'interface du routeur, la dernière adresse disponible au PC de test. Dans le cas particulier du réseau LAN8, affectez l'avant-dernière adresse disponible à l'adaptateur virtuel qui équipe le PC hôte.

- Une fois l'affectation d'adresses réalisée pour les trois réseaux LAN8, LAN11 et LAN12, affectez le premier espace disponible et suffisant à la liaison WAN séparant R80 et R110. Affectez la première adresse au routeur R110.

- Affectez l'espace disponible et suffisant suivant à la liaison WAN séparant R80 et R120. Affectez la première adresse au routeur R120.

- Complétez le tableau de documentation des adresses ci-après :

Affectation	Réseau	Première adresse	Dernière adresse	Adresse de diffusion
LAN11				
LAN8				
LAN12				
WAN R80 - R110				
WAN R80 - R120				

- Complétez le tableau de documentation des interfaces ci-après :

Equipement	Interface	Adresse IP	Masque	Passerelle
R80	F0/0			
	S0/0			
	S0/1			
R110	F0/0			
	S0/0			
R120	F0/0			
	S0/1			
PCL80				
PCL110				
PCL120				
PC Hôte				

2. Tâche 2 : Réalisation de la topologie sous GNS3

- Si vous n'êtes pas encore familier de l'utilisation de GNS3, merci de vous reporter au chapitre Annexes - Définition d'un contexte d'atelier.

- Créez la topologie sous GNS3. Placez vos trois routeurs. La valeur IDLE PC des routeurs a déjà été calculée, inutile d'y revenir. Configurez chaque routeur afin qu'il dispose de ports WAN. Si la plate-forme utilisée est celle du 2600, ajoutez la carte WIC2T. Placez les trois PC « Nuage » et configurez-les afin que chacun d'eux soit connecté à l'adaptateur réseau convenable et donc au concentrateur VMnet convenable. Établissez les liens LAN et WAN. Nommez les différents équipements afin de refléter la topologie proposée. Sauvegardez la topologie sous Atelier4.net.

- Démarrez chacun des trois routeurs.

- Ouvrez une seconde instance de VMware et avec elle, créez une équipe de trois PC PC80, PC110 et PC120. PC80 est obtenu par clonage sans lien (option **create a full clone**) de la machine Ubuntu fournie sur le site des Editions ENI. PC110 et PC120 sont obtenus par clone avec lien (option **linked clone**) de PC80. Nommez cette équipe 3PCL, faites en sorte que l'adaptateur réseau virtuel de chacune des machines soit connecté au concentrateur VMnet convenable, respectivement VMnet8, VMnet1 et VMnet2. Démarrez l'équipe puis réglez chacune des trois configurations IP.

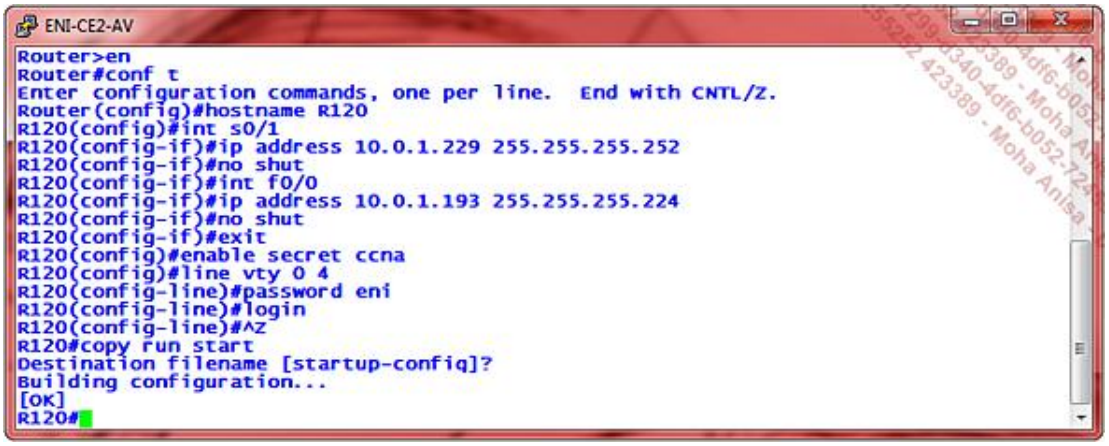
3. Tâche 3 : Configuration minimale des routeurs

- Effectuez un clic droit sur R80 et sélectionnez **Console**. Prenez garde à bien désélectionner R80 avant de sélectionner R110 et de lancer la console correspondante. Renouvelez l'opération avec R120. Au final, vous devez disposer des trois consoles opérationnelles.

- À l'aide de la session console ouverte sur R80, saisissez la configuration suivante :

- À l'aide de la session console ouverte sur R110, saisissez la configuration suivante :

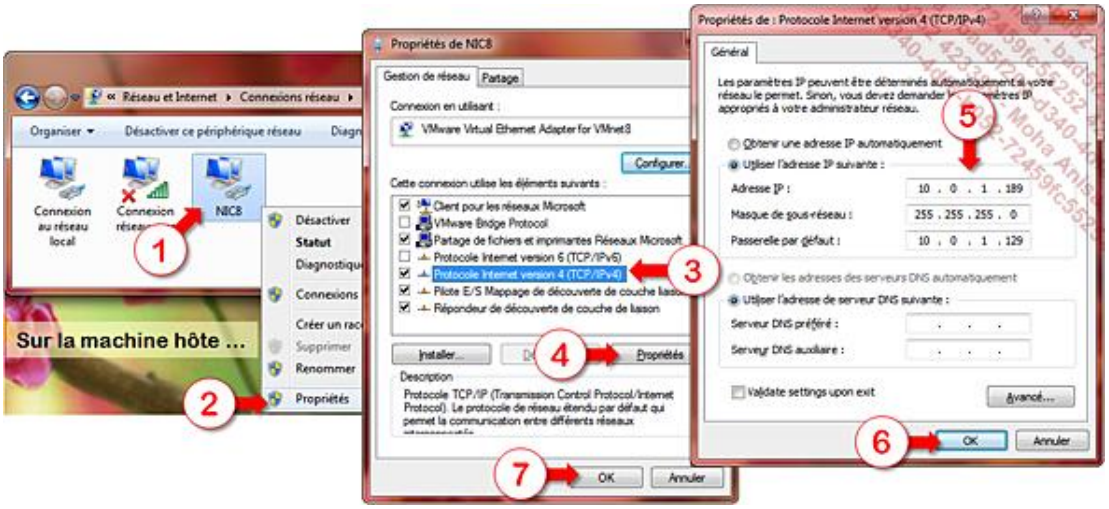
- À l'aide de la session console ouverte sur R120, saisissez la configuration suivante :



4. Tâche 4 : Gestion des connexions Telnet

La configuration entrée dans chacun des routeurs est suffisante pour pouvoir prendre la main à distance à l'aide d'une session Telnet. L'administrateur satisfait peut rejoindre le confort douillet de son bureau et entamer une configuration plus sérieuse à distance. En effet, pendant l'atelier « Prise en main de l'interface ILC », nous avons ajouté à la machine hôte un adaptateur réseau virtuel placé sur le concentrateur VMnet8. Il suffit d'attribuer à cet adaptateur une adresse IP du réseau LAN8 pour qu'une connexion Telnet depuis la machine hôte sur le routeur R80 devienne possible. En résumé, le PC hôte jouera le rôle de la station de l'administrateur.

- Configurez l'adaptateur NIC8 de la machine hôte :



Oui mais... Aucun routage n'a été configuré et la seule connexion possible depuis le poste de l'administrateur (dans le cas présent la machine hôte) l'est sur le routeur R80 via VMnet8. Dans ces conditions, est-il possible d'ouvrir une session Telnet sur R110 et R120 ? R80 connaît le réseau directement connecté correspondant au lien WAN jusqu'à R110. Donc les paquets remis par le PC hôte et destinés à R110 vont aboutir. C'est le retour qui est impossible car R110 et R120 ne disposent pas encore de route vers le réseau LAN8. Que fait l'administrateur astucieux ? Il ouvre une session Telnet sur R80, puis depuis cette session ouvre deux autres sessions Telnet vers R110 et R120 (les professionels parlent de rebond. R80 accessible a permis de rebondir sur R110 et R120) :

Le problème consiste alors à « naviguer » entre les sessions ouvertes sans se perdre. La seule information non sollicitée est l'invite de commandes qui, si la commande **hostname** a été entrée, permet de savoir quelle est la session affichée, mais rien de plus.

- Depuis la machine hôte, ouvrez une session Telnet sur R80. Depuis cette session, ouvrez une session sur R110 :

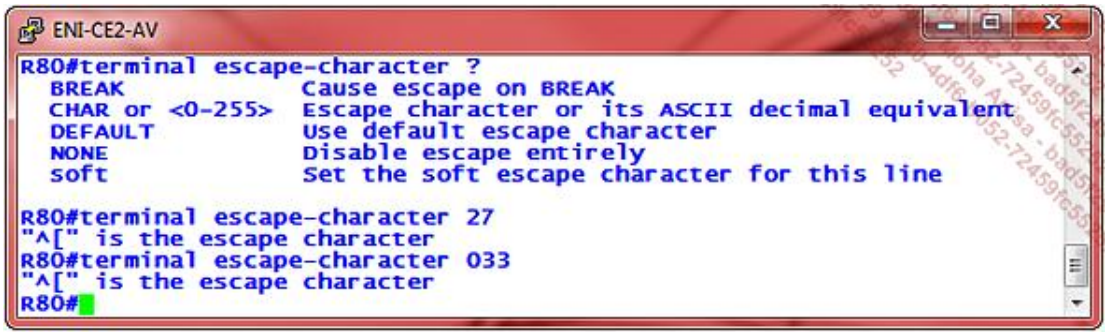
Attention, la session en cours sur R80 n'est pas fermée. C'est elle qui héberge la session sur R110, ce que l'invite de commande ne rappelle pas. Comment revenir à la session sur R80 ? Une première solution consiste simplement à mettre fin à la session sur R110 :

Mais l'administrateur préférera sans doute laisser la session ouverte et basculer d'une session à l'autre selon les besoins. Il existe une séquence d'échappement prévue pour revenir de la session hébergée à la session hôte sans mettre fin à la session hébergée. Il s'agit de la combinaison de touches [Ctrl][Flèche en haut] 6 (pressez ensemble les trois touches) suivie de la touche X. En réalité, la séquence [Ctrl][Flèche en haut] 6 provoque l'émission du caractère 0x1E de la table ASCII, appelé « Record Separator » (RS). La séquence composée du caractère RS associé au caractère x est représentée par **^^x**. Mais parce qu'il n'est jamais commode de frapper trois touches simultanément, parce qu'également il peut s'avérer difficile de faire parvenir cette séquence jusqu'au routeur selon le

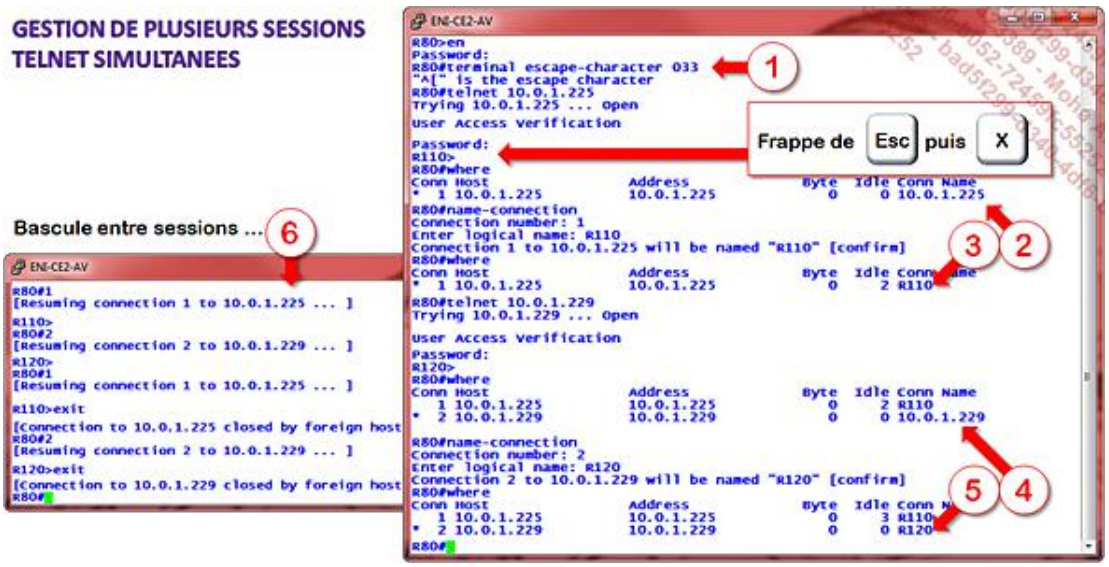
clavier (QWERTY ou AZERTY) ou le logiciel client utilisé, il peut être utile de changer la séquence d'échappement par défaut.

- Ouvrez à nouveau une session Telnet sur R80, passez en mode privilégié, entrez la commande **show terminal** afin de vérifier la séquence d'échappement en cours, entrez la commande **terminal escape-character 033** afin de remplacer la combinaison [Ctrl][Flèche en haut] 6 par la simple touche [Echap]. Provoquez à nouveau une commande **show terminal** pour vérifier la nouvelle séquence d'échappement :

La commande **Terminal** n'est pas une commande de configuration puisqu'elle est entrée en mode privilégié. Son effet disparaît avec la fin de la session en cours. Il est également possible d'entrer la commande **escape-character** en mode de configuration de ligne, son effet est alors définitif pour toute session ouverte depuis la ligne en question. C'est anecdotique mais on peut noter que la commande admet en paramètre le code ASCII du caractère choisi qu'il soit exprimé en décimal ou en octal dans la convention Intel (le nombre est précédé d'un zéro) :



- Ouvrez une session Telnet sur R80 puis à partir de cette session, ouvrez deux sessions sur R110 et R120. Utilisez la séquence d'échappement pour revenir à la session hôte :



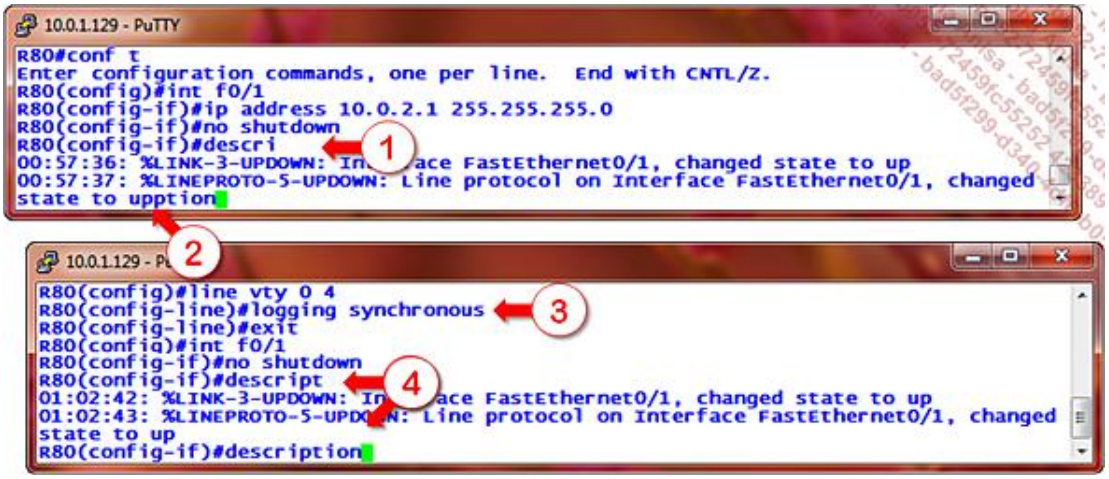
- À l'aide de la commande **where**, équivalente à une commande **show sessions**, inventoriez les sessions en cours. Observez que chaque session est associée à un numéro d'ordre. C'est ce numéro qu'il faut entrer en invite de commande de la session hôte pour afficher à nouveau la session hébergée correspondante. L'administrateur méticuleux peut aller jusqu'à nommer une session à l'aide de la commande **name-connection**, commande à taper sans argument. En effet, cette commande provoque un processus invitant l'utilisateur à entrer d'abord le numéro de la session qu'il désire nommer, puis le nom qu'il souhaite attribuer. Ainsi, dans l'exemple ci-dessus, les sessions 1 et 2 ont respectivement été nommées R110 et R120.

Parmi les multiples tâches accomplies par l'IOS, l'une d'elles intéresse particulièrement l'administrateur parce qu'elle lui permet de découvrir ou mieux comprendre les évènements qui affectent le fonctionnement du routeur et donc du réseau. Il s'agit de l'activité de journalisation des évènements. En la matière, CISCO comme une majorité de constructeurs se conforme au protocole SYSLOG normalisé dans le RFC 5424. Par défaut, la manifestation de SYSLOG sur un routeur se limite à l'émission des messages d'évènements vers le port console. Quel administrateur au cours de son travail depuis la console n'a pas été agacé par l'arrivée impromptue de ces messages qui viennent perturber la saisie en cours ? Problème facile à résoudre d'ailleurs, car il existe une commande de configuration de ligne **logging synchronous** qui peut être appliquée à la console ainsi qu'aux lignes vty et qui modifie le comportement de l'IOS quand il envoie un message : si une commande est en cours de saisie, alors l'IOS réaffiche le contenu de la ligne saisie dans l'état où elle se trouvait immédiatement avant l'envoi du message.

Par défaut, l'IOS envoie l'ensemble des messages sur le port console, ce qui correspond à la commande « **console logging** ». Depuis le mode privilégié, la commande **Terminal monitor** provoque l'affichage des messages SYSLOG, messages debug compris, sur la session en cours. Cette commande est utile pour obtenir les messages depuis une session ouverte via une ligne vty. Puisqu'il ne s'agit pas d'une commande de configuration, cette commande n'est pas mémorisée et son effet cesse lorsque la session prend fin (ceci est vrai pour toutes les commandes Terminal).

- La session Telnet est toujours ouverte sur R80. Entrez la commande **terminal monitor** puis vérifiez son effet en provoquant un message de console :

- Passez en configuration d'interface sur l'interface Ethernet f0/1 de R80 jusqu'ici non utilisée. Configurez l'adresse IP puis entrez la commande **no shutdown** et sans attendre, tentez d'entrer la commande **description**. Constatez l'effet désastreux de l'arrivée impromptue d'un message SYSLOG. Replacez l'interface dans l'état **shutdown**. En configuration de ligne vty, entrez la commande **logging synchronous**. Revenez à la configuration de l'interface f0/1, entrez à nouveau la commande **no shutdown** et à nouveau sans attendre, tentez d'entrer la commande **description**. Certes le message SYSLOG arrive encore mais cette fois sans inconvénient pour la saisie en cours :



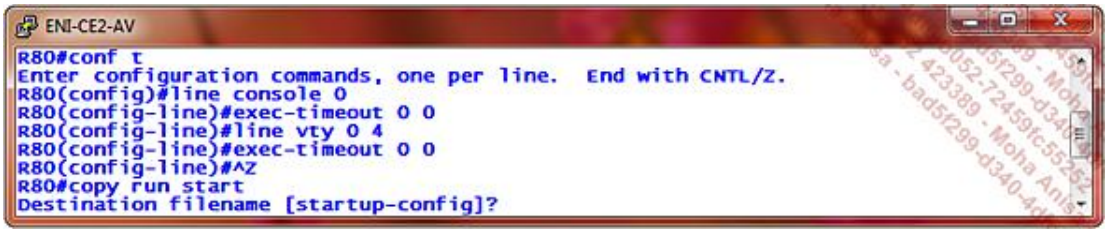
Nous sommes à présent plutôt bien outillés pour gérer nos sessions Telnet. Reste cependant un problème. Au moindre petit temps mort dans notre activité (dix minutes par défaut), voilà la session fermée par l'IOS. Dans un environnement de production, cette mesure de sécurité se justifie pleinement. Mais dans notre environnement didactique, que de temps perdu à ouvrir des sessions. Ce problème est facilement levé grâce à la commande de configuration de ligne **exec-timeout**. Cette commande définit le délai d'attente maximal de l'interpréteur de commandes EXEC jusqu'à détection d'une entrée quelconque de l'utilisateur. Parvenu à ce délai, la session est interrompue. La syntaxe est la suivante :

```
Router(config-line)#exec-timeout minutes [secondes]
```

Pour désactiver tout temporisateur d'interruption de session, il suffit de saisir la commande :

```
Router(config-line)#exec-timeout 0 0
```

- Ajoutez la ligne de commande **exec-timeout 0 0** aux lignes console et vty de chacun des routeurs R80, R110 et R120. Pour R80 :



5. Tâche 5 : Mise en œuvre de la commande debug ip routing

La commande **debug ip routing** permet de suivre en temps réel l'évolution de la table de routage. Chaque route ajoutée, modifiée ou supprimée fait l'objet d'un message SYSLOG debug. Puisque nous nous apprêtons à introduire des routes, cette commande peut fort à propos confirmer l'effet des commandes entrées.

Rappel : la commande undebug all ou no debug all désactive toutes les commandes debug qui seraient en cours de traitement. Une mesure sage consiste à entrer cette commande de façon systématique avant d'entrer une quelconque commande debug. La commande undebug all se trouve alors dans l'historique de commandes et il suffit d'une action sur la touche [Flèche en haut] puis de valider par la touche [Entrée] pour la provoquer. L'administrateur peut ainsi espérer arrêter une commande debug malheureuse même avec un processeur « noyé ».

- Otez toute commande de configuration des interfaces S0/0 et S0/1 du routeur R80. À l'exception de la commande **shutdown**, utilisez la forme **no** des commandes (exemple : **no ip address**).

- Sur R80, saisissez la commande **debug ip routing** :

```
R80#debug ip routing
```

```
IP routing debugging is on
R80#
```

- Passez en configuration d'interface et configurez l'adresse IP de l'interface S0/0 :

```
R80#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R80(config)#int s0/0
R80(config-if)#ip address 10.0.1.226 255.255.255.252
R80(config-if)#
00:53:42: is_up: 0 state: 6 sub state: 1 line: 0
00:53:42: is_up: 0 state: 6 sub state: 1 line: 0
R80(config-if)#
```

Dès validation par la touche [Entrée], SYSLOG signale la présence d'une nouvelle route mais cette route n'a pas encore été placée en table de routage.

- Introduisez la commande **clockrate 64000** (cette interface est côté DCE) puis la commande **no shutdown** :

```
R80(config-if)#clockrate 64000
R80(config-if)#no shutdown
R80(config-if)#
01:03:32: is_up: 0 state: 4 sub state: 1 line: 0
R80(config-if)#
01:03:34: %LINK-3-UPDOWN: Interface Serial0/0, changed state to up
R80(config-if)#
01:03:34: is_up: 1 state: 4 sub state: 1 line: 0
01:03:34: RT: network 10.0.0.0 is now variably masked
01:03:34: RT: add 10.0.1.224/30 via 0.0.0.0, connected metric [0/0]
01:03:34: RT: interface Serial0/0 added to routing table
01:03:34: is_up: 1 state: 4 sub state: 1 line: 0
01:03:35: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state
to up
R80(config-if)#
01:03:35: is_up: 1 state: 4 sub state: 1 line: 0
R80(config-if)#
```

La table de routage comporte désormais une route supplémentaire, ce que confirme la partie en gras des messages SYSLOG ci-dessus. Quand cela ne se produit pas, l'administrateur troque sa casquette d'administrateur pour une casquette d'enquêteur, ce qui n'est pas sans intérêt. L'auteur conseille de raisonner par couche (attention : les réflexions ci-après s'appliquent à un environnement physique, certaines sont donc sans objet dans notre environnement GNS3) :

Que faut-il pour que la liaison puisse s'établir en **couche 1** ? Une connectivité physique fonctionnelle ce dont l'administrateur enquêteur peut s'assurer en vérifiant les points suivants :

- Est-ce bien l'interface s0/0 qui a été câblée ?
- Y-a-t-il un équipement fonctionnel en vis-à-vis ? (les deux interfaces en vis-à-vis ne peuvent que « monter » ensemble. Si l'une est « down », l'autre ne peut être que « down »). Pour une liaison LAN, l'interface en vis-à-vis est certainement un port de commutateur : ce port est-il bien actif ? Pour une liaison WAN, l'interface WAN en vis-à-vis a-t-elle déjà été configurée ?

Une interface WAN convenablement câblée puis convenablement configurée n'implique pas nécessairement l'ajout d'une route à la table de routage. Ce n'est le cas que s'il existe en vis-à-vis une interface également câblée et configurée.

- Que disent les voyants de liaison ?
- Dans un environnement didactique, une liaison WAN est réalisée à l'aide de deux câbles reliés ensemble, l'un DTE, l'autre DCE. La commande **clockrate** a-t-elle bien été entrée côté DCE ?

Que faut-il pour que la liaison puisse s'établir en **couche 2** ? Un protocole de couche 2 identique de part et d'autre de

la liaison ce qui amène de nouvelles questions :

- Dans le cas d'une liaison LAN, le problème ne se pose pas (ou plus) sauf cas d'école et donc configuration exotique. Est-ce qu'il viendrait à l'idée d'un administrateur sérieux de forcer le mode full duplex alors que l'équipement en vis-à-vis est un concentrateur, c'est-à-dire un équipement fondamentalement half-duplex (CSMA/CD) ? Dans un environnement commuté, on profite aujourd'hui des avancées d'Ethernet dont l'auto négociation et l'auto MDI/MDI-X qui doivent aboutir à la montée du niveau 2 (si nécessaire, se reporter à l'ouvrage Cisco - Notions de base sur les réseaux dans la collection Certifications aux Editions ENI).

- Dans le cas d'une liaison WAN, par défaut, l'IOS fait le choix d'une encapsulation HDLC. L'important est que l'encapsulation soit réglée de façon identique à chaque extrémité de la liaison. Est-ce le cas ?

```
R80(config-if)#encapsulation ?
atm-dxi         ATM-DXI encapsulation
bstun           Block Serial tunneling (BSTUN)
frame-relay     Frame Relay networks
hdlc            Serial HDLC synchronous
lapb            LAPB (X.25 Level 2)
ppp             Point-to-Point protocol
sdlc            SDLC
sdlc-primary    SDLC (primary)
sdlc-secondary  SDLC (secondary)
smds            Switched Megabit Data Service (SMDS)
stun            Serial tunneling (STUN)
x25             X.25
R80(config-if)#
```

Vous avez vérifié l'ensemble de ces points et l'interface s0/0 n'est toujours pas montée ? Sans rire, il faut y croire un peu !

- Répétez sur l'interface S0/1 les opérations réalisées pour l'interface S0/0 :

```
R80(config-if)#int s0/1
R80(config-if)#ip address 10.0.1.230 255.255.255.252
R80(config-if)#
01:52:38: is_up: 0 state: 6 sub state: 1 line: 0
01:52:38: is_up: 0 state: 6 sub state: 1 line: 0
R80(config-if)#clockrate 64000
R80(config-if)#no shutdown
R80(config-if)#
01:52:58: is_up: 0 state: 4 sub state: 1 line: 0
01:53:00: %LINK-3-UPDOWN: Interface Serial0/1, changed state to up
R80(config-if)#
01:53:00: is_up: 1 state: 4 sub state: 1 line: 0
01:53:00: RT: add 10.0.1.228/30 via 0.0.0.0, connected metric [0/0]
01:53:00: RT: interface Serial0/1 added to routing table
01:53:00: is_up: 1 state: 4 sub state: 1 line: 0
01:53:01: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1, changed state
to up
R80(config-if)#
01:53:01: is_up: 1 state: 4 sub state: 1 line: 0
R80(config-if)#
```

Il est temps de vérifier l'état de la table de routage.

- Désactivez le mode **debug ip routing** puis utilisez la commande **show ip route** :

```
R80(config-if)#^Z
R80#
01:56:44: %SYS-5-CONFIG_I: Configured from console by vty1 (10.0.1.189)
R80#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
R80#undebug all
```

```
All possible debugging has been turned off
R80#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
C       10.0.1.128/26 is directly connected, FastEthernet0/0
C       10.0.1.224/30 is directly connected, Serial0/0
C       10.0.1.228/30 is directly connected, Serial0/1
R80#
```

Puisque R80 dispose de trois interfaces, il est normal de trouver trois routes directement connectées dans la table de routage, routes que le routeur a apprises depuis la configuration des interfaces.

- Ouvrez une session sur R110 puis visualisez sa table de routage. Répétez l'opération pour R120 :

Pour R110 :

```
User Access Verification
Password:
R80>en
Password:
R80#terminal escape-character 033
"^[" is the escape character
R80#telnet 10.0.1.225
Trying 10.0.1.225 ... Open

User Access Verification
Password:
R110>ship route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is not set

10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       10.0.1.0/25 is directly connected, FastEthernet0/0
C       10.0.1.224/30 is directly connected, Serial0/0
R110>
```

Pour R120 :

```
R110#exit
 [Connection to 10.0.1.225 closed by foreign host]
R80#telnet 10.0.1.229
Trying 10.0.1.229 ... Open

User Access Verification
Password:
R120>ship route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is not set

10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       10.0.1.192/27 is directly connected, FastEthernet0/0
C       10.0.1.228/30 is directly connected, Serial0/1
R120>
```

6. Tests de connectivité entre équipements adjacents

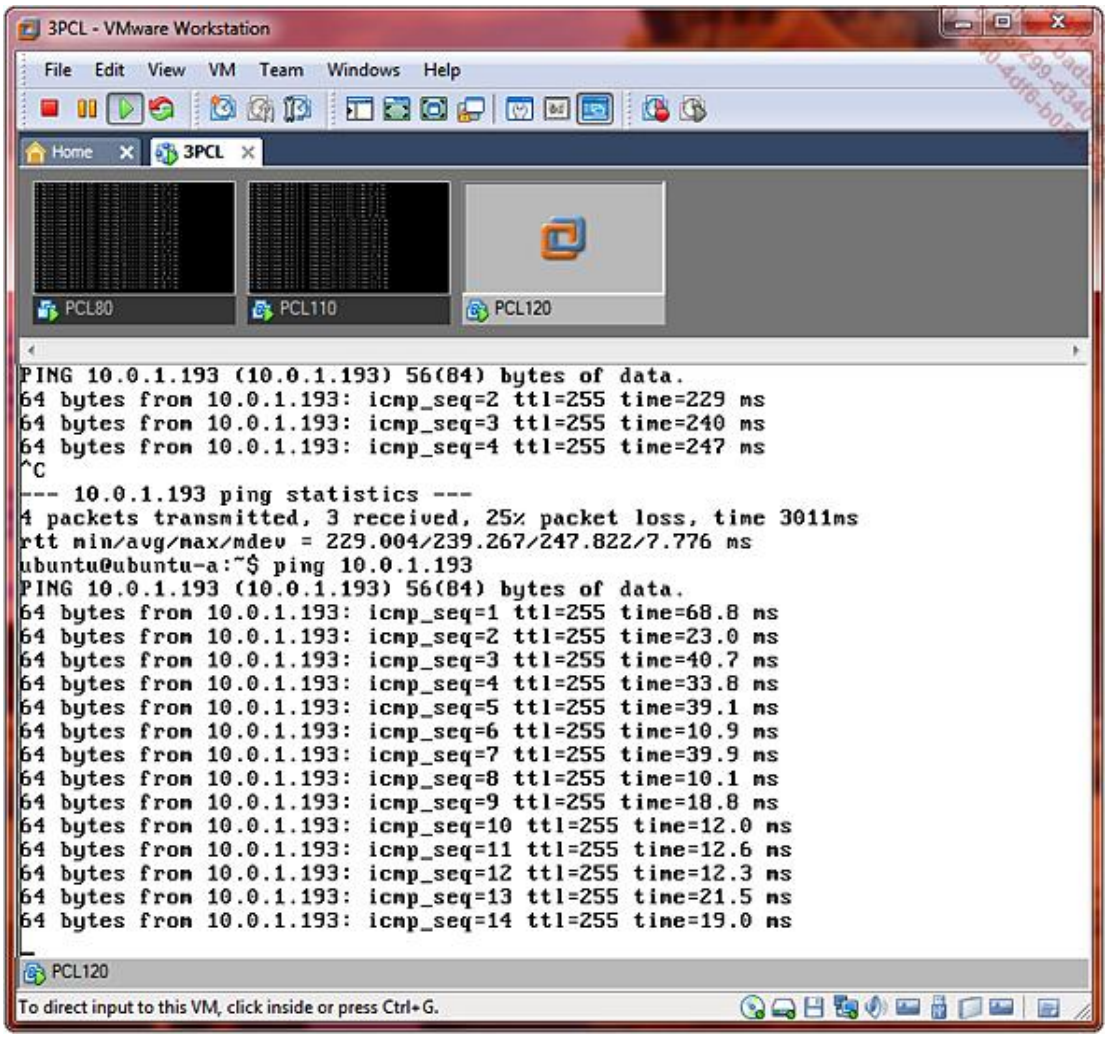
Plus rien ne s'oppose à ce que nous testions la connectivité entre équipements adjacents. Ceci comprend le test de connectivité entre chaque PC et la passerelle correspondante puis le test entre les routeurs pris deux à deux. Naturellement, si vous êtes déjà parvenu à prendre la main sur R80 à l'aide d'une session Telnet puis à prendre la main sur R110 et R120 à l'aide de sessions Telnet ouvertes depuis R80, la connectivité est déjà assurée sur un certain nombre de liens. En forme de synthèse donc :

1. Depuis PC80, tentez une requête ping vers sa passerelle par défaut 10.0.1.129.

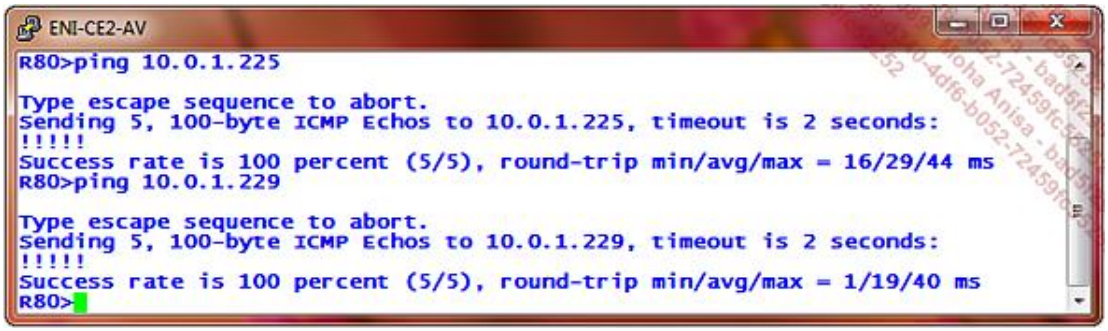
2. Depuis PC110, tentez une requête ping vers sa passerelle par défaut 10.0.1.1.

3. Depuis PC120, tentez une requête ping vers sa passerelle par défaut 10.0.1.193.

Pour mémoire, si vous avez adopté les machines Linux en ligne de commande, la commande ping provoque une succession non interrompue de requêtes ping à laquelle l'administrateur peut mettre fin à l'aide de la combinaison de touches [Ctrl] C. Autre spécificité VMware cette fois : reprendre le focus d'une fenêtre de PC Linux afin d'afficher une autre fenêtre nécessite la frappe de la combinaison [Ctrl][Alt].



- Depuis chacun la console de chacun des trois routeurs, tentez une requête ping vers chacun des deux autres routeurs. Exemple depuis R80 :



Si cela ne se passe pas aussi bien que dans les captures ci-dessus, voilà l'administrateur à nouveau engagé dans une procédure de dépannage et les questions à se poser sont toujours les mêmes.

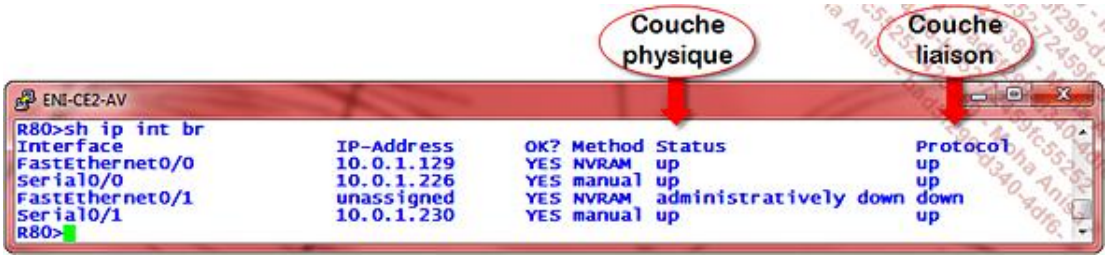
Dans une situation réelle :

- Chaque PC est-il bien physiquement raccordé au routeur correspondant ?
- Les voyants associés aux différents ports reflètent-ils un état de lien établi (selon les cas, voyants LINK ou CONN). Les voyants d'activité clignotent-ils ?
- La configuration IP des PC est-elle conforme à la configuration prévue dans le tableau de documentation des interfaces ?

Dans une situation émulée sous GNS3 :

- Chaque nuage de GNS3 est-il connecté à l'adaptateur réseau convenable et donc connecté au concentrateur VMnet convenable ?
- Chaque adaptateur réseau virtuel de chacun des PC de test est-il connecté au concentrateur VMnet convenable ?

Dans cette situation, la commande utile sur un routeur est sans doute **show ip interface brief** que l'on pourra abréger en **sh ip int br**. Pour chacune des interfaces, cette commande fournit l'état « *up* » ou « *down* » à la fois pour la couche physique et pour la couche liaison. L'état « *administratively down* » est affiché lorsqu'il existe une commande **shutdown** dans la configuration de l'interface :



7. Tests de connectivité entre équipements non adjacents

- Depuis chacun des PC de test, tentez une requête ping vers chacun des deux autres PC.

Aucune de ces requêtes n'aboutit ! Est-ce normal ? Oui, rassurez-vous. Chaque routeur ne connaît que les routes directement connectées. R80 ne connaît pas l'existence des réseaux LAN11 et LAN12, R110 ne connaît pas les réseaux LAN8 et LAN12, enfin R120 ne connaît pas les réseaux LAN8 et LAN11.

8. Introduction de routes statiques

a. Routes statiques avec adresse de saut suivant

Nous utiliserons cette forme de la commande :

```
Router(config)# ip route @destination masque @saut_suivant
```

Le choix du masque conditionne le niveau d'agrégation. Par exemple, 10.0.1.0/24 englobe l'ensemble de notre topologie dont les réseaux LAN8, LAN11 et LAN12. Alors que 10.0.1.0/25 ne représente que LAN11.

- Sur R110, introduisez une route statique vers le réseau LAN8 en utilisant l'adresse de saut suivant :

```
R110#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R110(config)#ip route 10.0.1.128 255.255.255.192 10.0.1.226
R110(config)#^Z
R110#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
```

- Sur R110, provoquez une commande **show ip route** et observez la route ajoutée. Remarquez que cette route est précédée de la lettre S qui rappelle qu'il s'agit d'une route statique :

```
R110#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is not set

10.0.0.0/8 is variably subnetted, 3 subnets, 3 masks
C       10.0.1.0/25 is directly connected, FastEthernet0/0
S       10.0.1.128/26 [1/0] via 10.0.1.226
C       10.0.1.224/30 is directly connected, Serial0/0
R110#
```

Désormais, tous les paquets qui parviennent à R110 et dont les 26 bits de poids fort de l'adresse IP de destination correspondent aux 26 bits de poids fort du réseau 10.0.1.128 seront transférés vers R80 via son interface 10.0.1.226. Pour ce faire, R110 utilise son interface s0/0.

Le tableau ci-dessous imagine quelques paquets parvenus à R110. Pour chacun des paquets, indiquez comment se comportera le routeur. Va-t-il transférer ou rejeter ? Quelle est l'interface utilisée quand il transfère ?

Paquet	Adresse IP de destination	Rejet ou Transfert ?	Interface de sortie
1	10.0.1.229		
2	10.0.1.190		
3	10.0.1.226		
4	10.0.1.222		
5	10.0.1.126		

Il ne suffit pas que le routeur R110 transfère un paquet vers une destination pour laquelle il dispose d'une route pour que ce paquet parvienne nécessairement à sa destination finale.

- Si ce n'est déjà fait, installez Wireshark sur la machine hôte puis provoquez une capture sur l'adaptateur virtuel installé sur la machine hôte, connecté à VMnet8 et que nous avions nommé dans un précédent atelier NIC8.

- Revenez à PCL110 et lancez la commande ping sur l'hôte 10.0.1.188.

- Après une succession suffisante de requêtes, arrêtez la commande ping sur PCL110 puis cessez la capture de Wireshark.

La capture est disponible en téléchargement sur le site ENI : **cap_2D_01.pcap**

L'hôte 10.0.1.188 s'il existe, appartient au réseau LAN8. PCL110 remet la requête à sa passerelle. R110 dispose d'une route vers le réseau LAN8 (la route statique que nous venons d'introduire) et transfère le paquet à R80. À son tour, R80 dispose d'une route directement connectée à LAN8 mais pour transférer le paquet, l'adresse MAC du destinataire 10.0.1.188 lui est nécessaire. R80 ne trouve pas cette adresse dans son cache ARP et diffuse une requête ARP « who has 10.0.1.188 ? Tell 10.0.1.129 ». Il n'obtient évidemment pas de réponse. Ce processus est répété pour chaque requête ICMP en provenance de PCL110. Ceci explique la répétition des diffusions ARP dans la capture (trames 1, 3, 4, 5, 6, 8...).

- Revenez à PCL110 et lancez la commande ping vers PCL80. Revenez au PC hôte et relancez une capture Wireshark sur l'adaptateur virtuel connecté à VMnet8. Après un nombre suffisant de requêtes, arrêtez la commande ping sur PCL110 et cessez la capture Wireshark.

La capture est disponible en téléchargement sur le site ENI : **cap_2D_02.pcap**

Cette fois le destinataire existe et pourtant PCL110 n'obtient pas de réponse ICMP à ses requêtes ICMP. Chaque requête est bien transférée par R110 à R80, puis par R80 à PCL80 (trames 1, 4, 8, 11, 14... de la capture). Chaque requête provoque une réponse de PCL80 (trames 2, 5, 9, 12, 15...). Mais cette réponse est destinée à 10.0.1.126 qui appartient au réseau LAN11. R80 ne dispose pas de route vers ce réseau et réagit en transmettant à PCL80 un paquet ICMP « Destination unreachable » (trames 3, 6, 10, 13, 16... de la capture).

- Sur R80, introduisez une route statique vers le réseau LAN11 en utilisant l'adresse de saut suivant :

```
R80#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R80(config)#ip route 10.0.1.0 255.255.255.128 10.0.1.225
R80(config)#^Z
R80#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
R80#
```

- Sur R80, provoquez une commande **show ip route** et observez la route ajoutée. Remarquez que cette route est précédée de la lettre S qui rappelle qu'il s'agit d'une route statique :

```
R80#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
..........

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 4 subnets, 3 masks
S       10.0.1.0/25 [1/0] via 10.0.1.225
C       10.0.1.128/26 is directly connected, FastEthernet0/0
C       10.0.1.224/30 is directly connected, Serial0/0
C       10.0.1.228/30 is directly connected, Serial0/1
R80#
```

- Revenez à PCL110 et lancez la commande ping vers PCL80. Après un nombre suffisant de requêtes, arrêtez la commande ping sur PCL110.

Les requêtes doivent réussir.

b. Route statique avec interface de sortie

Cette seconde partie d'atelier est moins guidée à dessein. Le lecteur a certainement hâte d'être autonome. Alors, jetons-nous à l'eau.

- Sur la console de R80, tentez une commande ping vers PCL120.

Cette commande doit échouer, R80 ne disposant pas de route vers le réseau LAN12.

- Sur R80, ajoutez une route statique de type à interface de sortie vers le réseau LAN12.

```
R80#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R80(config)#ip route 10.0.1.192 255.255.255.224 s0/1
R80(config)#^Z
R80#
00:09:49: %SYS-5-CONFIG_I: Configured from console by console
R80#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
R80#
```

- Sur la console de R80, tentez à nouveau une commande ping vers PCL120.

```
R80#ping 10.0.1.222
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.222, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/66/108 ms
R80#
```

L'administrateur souhaite vérifier la connectivité de PCL80 à PCL120 sans se déplacer, c'est-à-dire en restant devant la console de R80.

- Sur la console de R80, utilisez une commande ping étendue afin de vérifier la connectivité de PCL80 à PCL120.

```
R80#ping 10.0.1.222 source 10.0.1.129

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.222, timeout is 2 seconds:
Packet sent with a source address of 10.0.1.129
.....
Success rate is 0 percent (0/5)
R80#
```

La commande échoue parce que R120 ne connaît pas de route vers le réseau LAN8.

- Sans vous déplacer sur la console de R120, en restant sur R80 donc, ouvrez une session Telnet sur R120 puis ajoutez une route statique par défaut de type à adresse de saut suivant. Fermez la session Telnet afin de revenir à la session console sur R80 :

```
R80#telnet 10.0.1.229
Trying 10.0.1.229 ... Open

User Access Verification
Password:
R120>en
Password:
R120#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R120(config)#ip route 0.0.0.0 0.0.0.0 10.0.1.230
R120(config)#^Z
R120#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
R120#exit

[Connection to 10.0.1.229 closed by foreign host]
R80#
```

- Sur la console de R80, utilisez à nouveau une commande ping étendue afin de vérifier la connectivité de PCL80 à PCL120 :

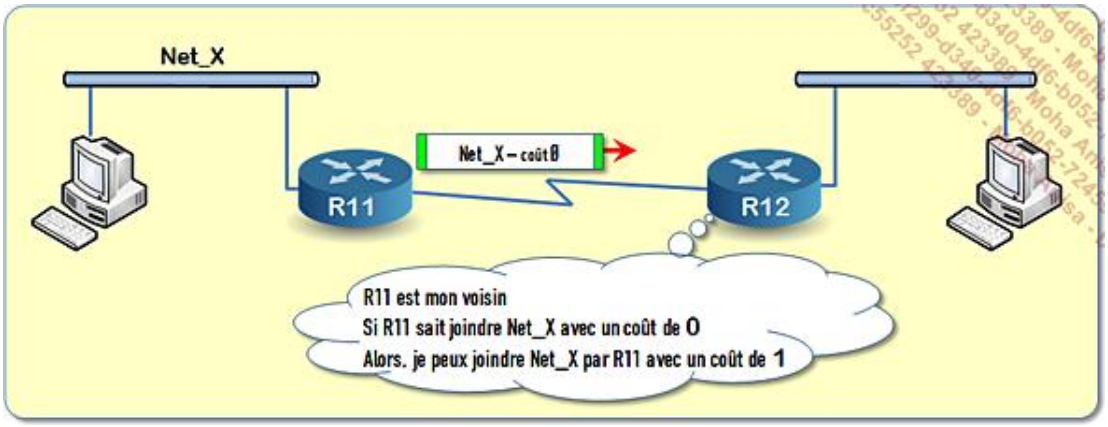
```
R80#ping 10.0.1.222 source 10.0.1.129

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.222, timeout is 2 seconds:
Packet sent with a source address of 10.0.1.129
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 48/426/1840 ms
R80#
```

Les requêtes ICMP obtiennent leurs réponses. Bravo, vous n'ignorez plus grand-chose du routage statique et avez encore progressé dans le maniement de l'interface ILC et des commandes de l'IOS.

Les protocoles de routage type vecteur de distance

La plupart des protocoles de routage appartiennent à l'une des deux familles, état de liens ou vecteur de distance. Les protocoles de routage à état de liens sont présentés au travers du protocole OSPF (*Open Shortest Path First*) dans le chapitre éponyme de cet ouvrage. Puisque RIP est la première occasion d'aborder les protocoles de routage à vecteur de distance, nous commencerons par brosser un portrait général de cette famille. RIP ainsi que tous les membres de sa famille utilisent un algorithme que l'on doit à Messieurs Richard Bellman et Lester Ford (mathématiciens américains) et dont l'objet est de trouver le plus court chemin depuis un sommet dans un graphe pondéré. Cet algorithme fut utilisé pour la première fois en 1967 en tant qu'algorithme de routage dans le réseau ARPANET (premier réseau à transfert de paquets développé aux Etats-Unis, considéré comme l'ancêtre d'Internet).

Pourquoi vecteur de distance ? Parce que dans cette famille, les routes sont annoncées comme des vecteurs. Le vecteur du mathématicien, c'est un segment orienté d'une certaine longueur. La route se définit comme une direction associée à une distance, ce qui est effectivement analogue :



Ainsi, chaque routeur du réseau dépend de ses voisins pour apprendre les routes et contribue à l'apprentissage de ses voisins en leur annonçant les routes qu'il connaît. L'information de topologie circule de proche en proche sur le réseau ce qui fait parfois appeler ce type de routage « routage par la rumeur ».

La famille des protocoles à vecteur de distance est assez nombreuse :

- RIP (*Routing Information Protocol*), le protocole objet de ce chapitre ;
- XNS RIP (*Xerox Network Services*) ;
- Novell IPX RIP ;
- IGRP (*Interior Gateway Routing Protocol*), protocole propriétaire de CISCO ;
- RTMP (*Routing Table Maintenance Protocol*), en usage dans la suite de protocoles propriétaire AppleTalk d'Apple.

1. Caractéristiques génériques

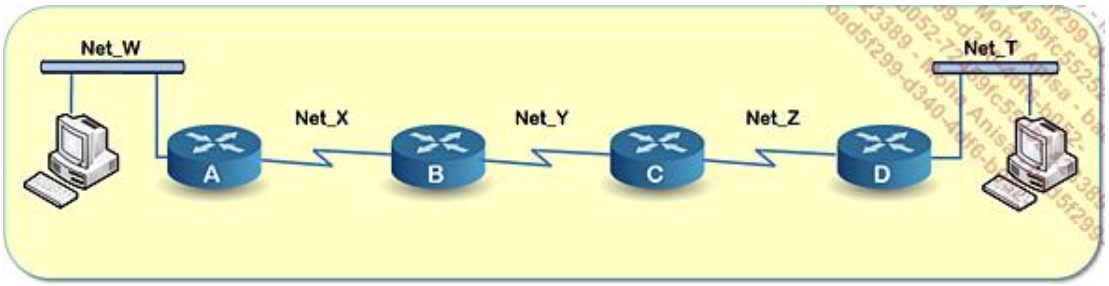
La caractéristique la plus notable partagée par l'ensemble de ces protocoles consiste pour un routeur à diffuser de façon périodique la totalité de sa table de routage sur toutes ses interfaces qui participent au protocole. Le seul protocole dérogeant à cette règle est EIGRP (*Enhanced Interior Gateway Routing Protocol*), protocole propriétaire de CISCO qui remplace IGRP. En effet, les mises à jour d'EIGRP ne sont ni diffusées, ni générées de façon périodique pas plus qu'elles ne contiennent l'entièreté de la table de routage. EIGRP fait l'objet d'un chapitre dédié dans cet ouvrage.

- Mises à jour diffusées : dans la version la plus simple, un routeur type vecteur de distance envoie ses mises à jour vers l'adresse de diffusion (255.255.255.255 soit l'adresse de diffusion limitée dans le cas d'IP). Les routeurs voisins qui mettent en œuvre le même protocole entendent et mettent à profit ces mises à jour.
- Mises à jour périodiques : la période qui sépare deux émissions de mises à jour est fixe. La valeur choisie varie selon les implémentations de 10 à 90 secondes. C'est que le choix de cette valeur conditionne la poursuite de deux objectifs contradictoires : faut-il privilégier le temps de convergence en diminuant la période des mises à jour ou limiter la consommation de bande passante en l'augmentant ?

- Mises à jour totales : à nouveau dans leur version la plus simple, les protocoles type vecteur de distance placent l'ensemble de la table de routage dans les mises à jour. C'est donc au routeur qui entend la mise à jour d'extraire l'information dont il a besoin et d'ignorer celle qui n'est pas pertinente pour lui.

2. Le routage par la rumeur

Observons l'évolution des tables de routage dans le contexte suivant :



Les quatre routeurs terminent leur démarrage au même instant t0. Chacun des routeurs ne connaît dans un premier temps que les réseaux qui lui sont directement connectés :

	Routeur A			**Routeur B**			**Routeur C**			**Routeur D**		
	Réseau	Via	Métrique	Réseau	Via	Métrique	Réseau	Via	Métrique	Réseau	Via	Métrique
t0	Net_W		0	Net_X		0	Net_Y		0	Net_Z		0
	Net_X		0	Net_Y		0	Net_Z		0	Net_T		0

À l'instant t1, chaque routeur reçoit et met à profit une première mise à jour de ses voisins. Prenons le cas du routeur A. Le routeur B lui dit pouvoir joindre les réseaux X et Y avec un coût de 0. Si le coût est 0 pour B, il doit être de 1 pour A voisin de B. A incrémente les coûts annoncés de 1 saut puis compare les routes obtenues avec celles contenues dans sa table. X est déjà connu avec un meilleur coût, cette partie de la mise à jour est donc ignorée. Le réseau Y par contre est nouveau, A ajoute une route vers Y dans sa table, route qu'il associe au coût calculé soit 1. La mise à jour émane de B, B est donc le prochain saut pour cette route, A découvre l'adresse IP du prochain saut en lisant l'adresse source du paquet IP qui contient la mise à jour :

	Réseau	**Via**	**Métrique**	**Réseau**	**Via**	**Métrique**	**Réseau**	**Via**	**Métrique**	**Réseau**	**Via**	**Métrique**
t1	Net_W		0	Net_X		0	Net_Y		0	Net_Z		0
	Net_X		0	Net_Y		0	Net_Z		0	Net_T		0
	Net_Y	B	1	Net_W	A	1	Net_X	B	1	Net_Y	C	1
				Net_Z	C	1	Net_T	D	1			

Chacun des routeurs conduit des opérations similaires. À l'instant t2, chaque routeur reçoit une nouvelle mise à jour. Plaçons-nous à nouveau sur le routeur A. Le routeur B lui dit pouvoir joindre X, Y, W et Z aux coûts respectifs de 0, 0, 1 et 1. Le routeur A incrémente ces valeurs qui deviennent 1, 1, 2 et 2 puis compare les routes obtenues aux routes contenues dans sa table. Les routes vers W et X sont écartées car déjà connues avec un meilleur coût. La route vers Y est déjà connue et le nouveau coût calculé est identique au coût connu, l'information est également ignorée. La route vers Z est nouvelle, A l'ajoute à sa table associée au coût calculé :

	Réseau	**Via**	**Métrique**	**Réseau**	**Via**	**Métrique**	**Réseau**	**Via**	**Métrique**	**Réseau**	**Via**	**Métrique**
t2	Net_W		0	Net_X		0	Net_Y		0	Net_Z		0
	Net_X		0	Net_Y		0	Net_Z		0	Net_T		0
	Net_Y	B	1	Net_W	A	1	Net_X	B	1	Net_Y	C	1

	Net_Z	B	2	Net_Z	C	1	Net_T	D	1	Net_X	C	2
				Net_T	C	2	Net_W	B	2			

C'est seulement à la réception de la mise à jour de l'instant t3 que chacun des routeurs a connaissance de l'ensemble des réseaux, on dit que le réseau a convergé :

	Réseau	Via	Métrique	Réseau	Via	Métrique	Réseau	Via	Métrique	Réseau	Via	Métrique
t3	Net_W		0	Net_X		0	Net_Y		0	Net_Z		0
	Net_X		0	Net_Y		0	Net_Z		0	Net_T		0
	Net_Y	B	1	Net_W	A	1	Net_X	B	1	Net_Y	C	1
	Net_Z	B	2	Net_Z	C	1	Net_T	D	1	Net_X	C	2
	Net_T	B	3	Net_T	C	2	Net_W	B	2	Net_W	C	3

a. Et quand la rumeur se tait ?

Maintenant que notre modeste réseau a convergé, voyons comment il appréhende un changement de topologie. Un premier cas simple est celui du réseau qui devient inactif. Imaginons que le réseau W passe à l'état inactif, il suffit au routeur A d'annoncer ce réseau comme étant injoignable dans la prochaine mise à jour. Mais les choses se compliquent quand c'est le routeur lui-même qui est défaillant. Imaginons que le routeur A devienne inactif. Les routeurs B, C et D disposent toujours d'une route vers le réseau W et plus aucun routeur pour leur dire que ce réseau est désormais injoignable.

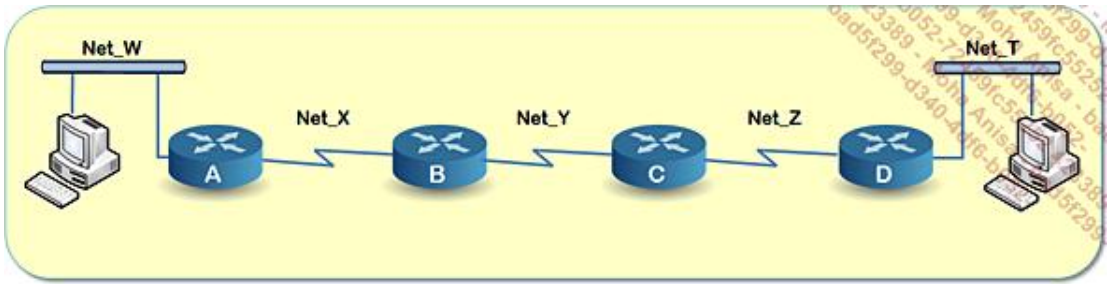
Ce problème est résolu par les protocoles de routage à vecteur de distance en associant un temporisateur à chaque route entrée dans la table. Un routeur qui reçoit une route déjà connue dans une mise à jour écarte l'information et dans le même temps rétablit le temporisateur à sa valeur initiale. Si les mises à jour cessent de parvenir au routeur pour la route concernée, le temporisateur finit par expirer et le routeur marque cette route comme étant injoignable. Il l'annoncera comme telle lors de sa prochaine émission de mise à jour.

Ce temporisateur d'espérance de vie est appelé temporisateur d'invalidation de route (*Route Invalidation Timer*) et sa valeur est typiquement égale à 3 à 6 fois la période d'émission des mises à jour. Diminuer davantage la valeur du temporisateur, c'est prendre le risque de marquer injoignable une route valide parce qu'un paquet de mise à jour a été perdu. L'augmenter au contraire dégradera le temps de reconvergence du réseau suite à un changement de topologie.

3. Prévention des boucles de routage

a. Partage de l'horizon (Split Horizon)

Raisonnons à nouveau sur ce contexte :



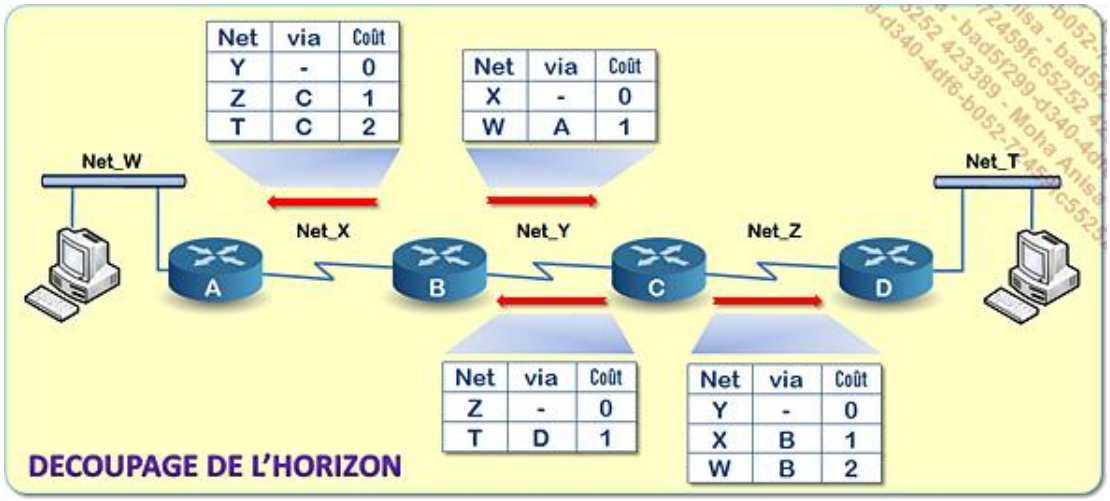
Nous avons affirmé de façon péremptoire que chaque routeur qui participe à un protocole DV (*Distance Vector*) diffuse l'ensemble de sa table de routage vers chacun de ces voisins. Mais est-ce indispensable, seulement utile, inutile ou carrément dangereux ? Vous êtes l'élève face au professeur. Vous viendrait-il à l'idée de tenter de lui apprendre ce qu'il vient de vous inculquer ? Appliqué au réseau ci-dessus, la question devient « Pourquoi un routeur diffuserait sur une interface ce qu'il a appris via cette interface ? » Ainsi, le routeur A ne peut pas apprendre à B ce

qu'il vient d'apprendre de B. La première réponse à la question précédente est donc : non, il n'est pas utile de diffuser l'ensemble de la table de routage. Dans l'exemple ci-dessus, chaque routeur ne devrait diffuser vers la droite que ce qu'il a appris par la gauche et vice-versa. Diffuser l'ensemble de la table de routage consommerait inutilement des ressources machines et de la bande passante.

Mais allons plus loin en imaginant que le réseau W devienne inactif. Pour un temps, la diffusion de la table de routage est maintenue dans sa totalité. Le routeur A détecte l'état injoignable du réseau W et se prépare à en informer le routeur B lors de la prochaine mise à jour. À ce moment précis, le routeur A reçoit la mise à jour du routeur B qui l'informe disposer d'une route vers le réseau W au coût de 1. Nous, gentils administrateurs, savons que cette route est une illusion puisqu'elle passe par A. Mais le routeur A ne dispose d'aucun moyen qui lui permettrait de remettre cette route en question. En conséquence, A incrémente le coût reçu et place W dans sa table via B au coût de 2.

Imaginons à ce stade un paquet entrant sur le routeur B, venu d'on ne sait où et destiné à un hôte du réseau W. Le routeur B dispose d'une route vers W via A et fait donc progresser le paquet vers A. Le routeur A dispose d'une route vers W via B et fait progresser le paquet vers B. Ceci peut se répéter indéfiniment jusqu'à épuisement du TTL du paquet, nous voilà installés dans une boucle de routage.

On le voit, il faut encore revoir la réponse à la question précédente qui passe de inutile à carrément dangereux. La solution imaginée pour éviter l'effet inutile et pallier l'effet dangereux se nomme Partage de l'horizon (*Split horizon*). Il en existe deux déclinaisons : le partage d'horizon dit simple et le partage d'horizon avec empoisonnement de route inverse (*Split horizon with poisoned reverse*). La figure suivante illustre le fonctionnement du partage d'horizon simple :



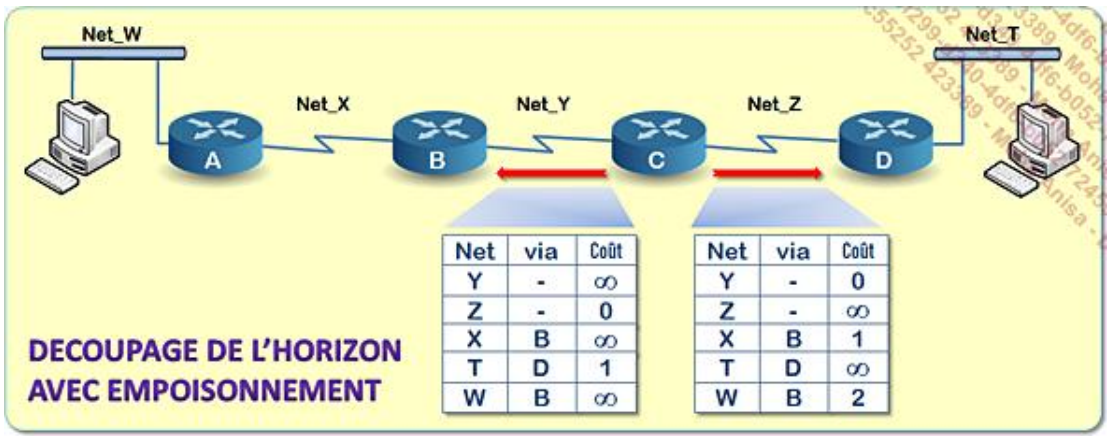
Quand le partage d'horizon simple est appliqué, la règle est de ne pas inclure dans une mise à jour émise sur une interface particulière les réseaux qui ont été appris via cette interface. Ainsi dans la figure ci-dessus, le routeur C informe le routeur B des réseaux Z et T. Les réseaux X et W sont absents de cette mise à jour parce qu'ils ont été appris par l'interface connectée à B. De la même façon, le routeur B informe le routeur C des réseaux X et W. Les deux réseaux Z et T sont absents de cette mise à jour parce qu'ils ont été appris par l'interface connectée à C.

La règle de partage d'horizon consiste à ne pas annoncer à un voisin la route apprise via ce voisin.

---

b. Partage de l'horizon avec empoisonnement

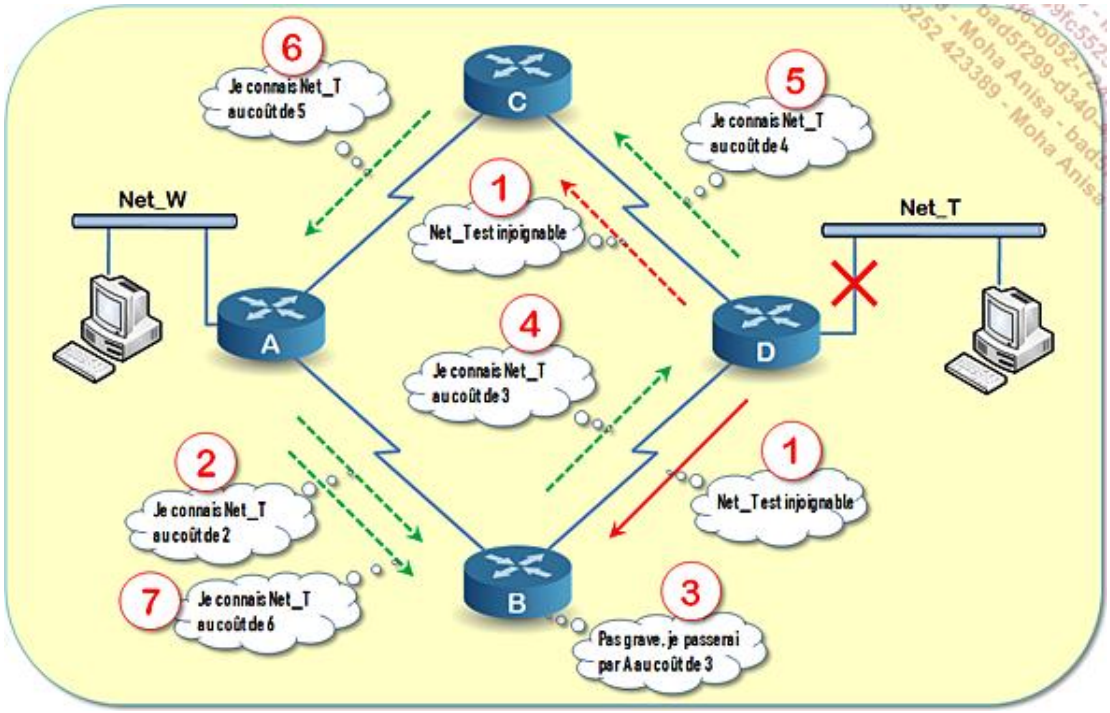
Si la règle de partage d'horizon avait consisté à occulter une partie des informations, la déclinaison avec empoisonnement procède d'une logique plus élaborée. Le dicton dit « pas de nouvelles, bonnes nouvelles ». Le partage d'horizon avec empoisonnement lui dit « mieux vaut de mauvaises nouvelles que pas de nouvelles ». La règle consiste à annoncer comme injoignable sur une interface tout réseau appris via cette interface. L'application de cette règle au contexte précédent modifie quelque peu le contenu des mises à jour :



Si, quelle que soit la source de l'erreur, les routeurs B et D installent à tort des routes via C vers Y, X et W dans le cas du routeur B, vers Z et T dans le cas du routeur D, alors très rapidement, l'annonce faite par C des routes empoisonnées rétablirait la vérité et donc supprimerait une source potentielle de boucles de routage. Ceci fait considérer le partage d'horizon avec empoisonnement comme étant généralement plus sûr que le partage d'horizon simple, cette fiabilité étant acquise au prix d'un alourdissement des mises à jour.

c. Le comptage à l'infini

Observez le contexte ci-après :



Le réseau Net_T passe à l'état inactif :

- Étiquette 1 : le routeur D annonce Net_T comme injoignable dans ses mises à jour vers C et B.
- Étiquette 2 : le routeur B reçoit successivement la mise à jour de D qui lui annonce T injoignable puis la mise à jour de A qui lui annonce connaître T au coût de 2.
- Étiquette 3 : le routeur B place la route vers T via A au coût de 3 dans sa table.
- Étiquette 4 : le routeur B annonce connaître le réseau T au coût de 3.
- Le routeur D place T dans sa table au coût de 4 via B.

- Étiquette 5 : le routeur D annonce connaître T au coût de 4.
- Le routeur C place T dans sa table au coût de 5 via D.
- Étiquette 6 : le routeur C annonce connaître T au coût de 5.
- Le routeur A place T dans sa table au coût de 6.
- Étiquette 7 : le routeur A annonce connaître T au coût de 6...
- Le routeur B se dit « le coût de la route via A vers T a augmenté mais il n'y a pas d'autre alternative, je la conserve ».

On le voit, sans précaution particulière, nous voilà engagés dans une boucle sans fin dont la règle du partage d'horizon ne protège pas (le routeur A par exemple annonce à B une route qu'il a apprise de C, la règle du partage d'horizon est bien respectée), le coût de la route vers T pourrait augmenter jusqu'à l'infini.

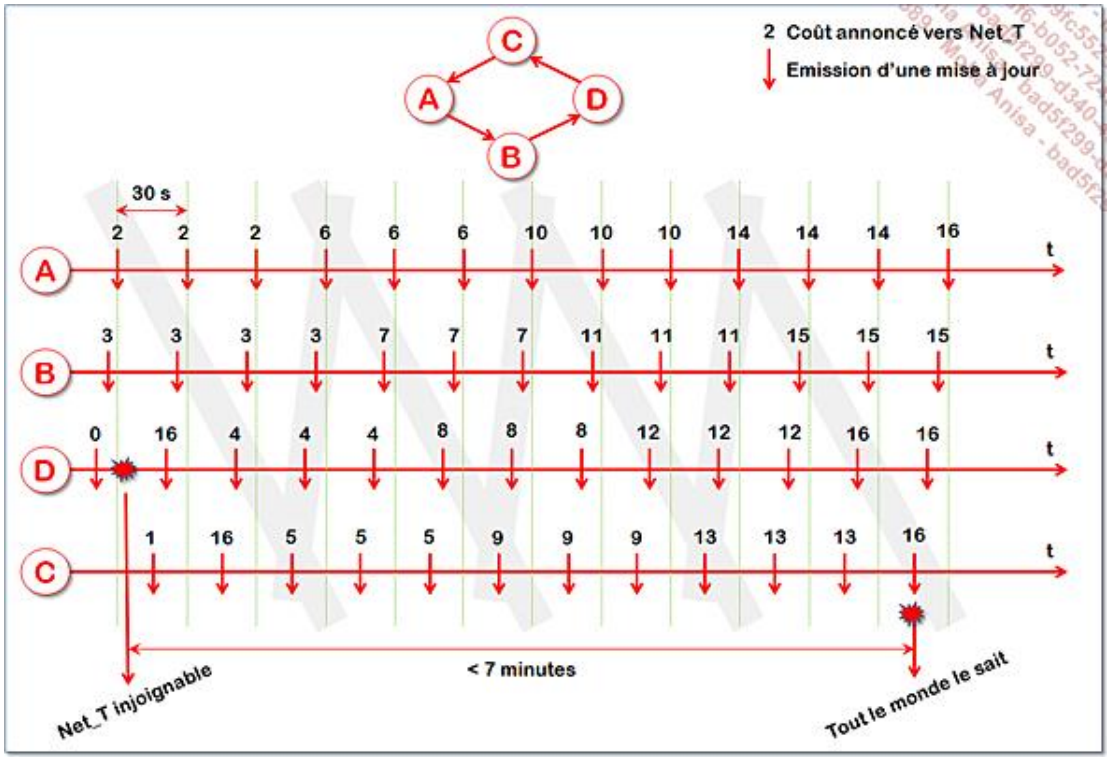
Comment résoudre ce nouveau problème de comptage à l'infini ? En définissant l'infini qui ainsi en devient fini ! La plupart des protocoles à vecteur de distance fixent l'infini à la valeur 16. Dans l'exemple ci-dessus, le coût de la route vers T continuerait à augmenter jusqu'à 16. Un routeur qui reçoit une annonce de route associée à un coût de 16 cesse d'incrémenter la valeur et considère la route comme injoignable.

Conséquence inattendue de la solution apportée au problème du comptage à l'infini : nous disposons désormais d'une méthode pour annoncer un réseau comme étant injoignable. Il suffit de l'annoncer avec un coût de 16.

Évitons pourtant de conclure hâtivement que le monde du vecteur de distance est un monde merveilleux. En fixant l'infini à 16, on limite de fait le nombre de sauts à 15 et c'est également ce nombre de sauts qui fixe l'étendue du réseau :

Le nombre de sauts dans un réseau pour lequel on a fait le choix d'un protocole de routage à vecteur de distance est limité à 15. Ceci cantonne l'utilisation de cette famille de protocoles à des réseaux faiblement étendus.

Autre conséquence de l'infini fixé à 16 : l'obtention de la convergence peut nécessiter beaucoup de temps. En imaginant par exemple des mises à jour espacées de 30 secondes, le temps de convergence peut atteindre 7 minutes ce qui est beaucoup (14 sauts espacés de 30 secondes) :



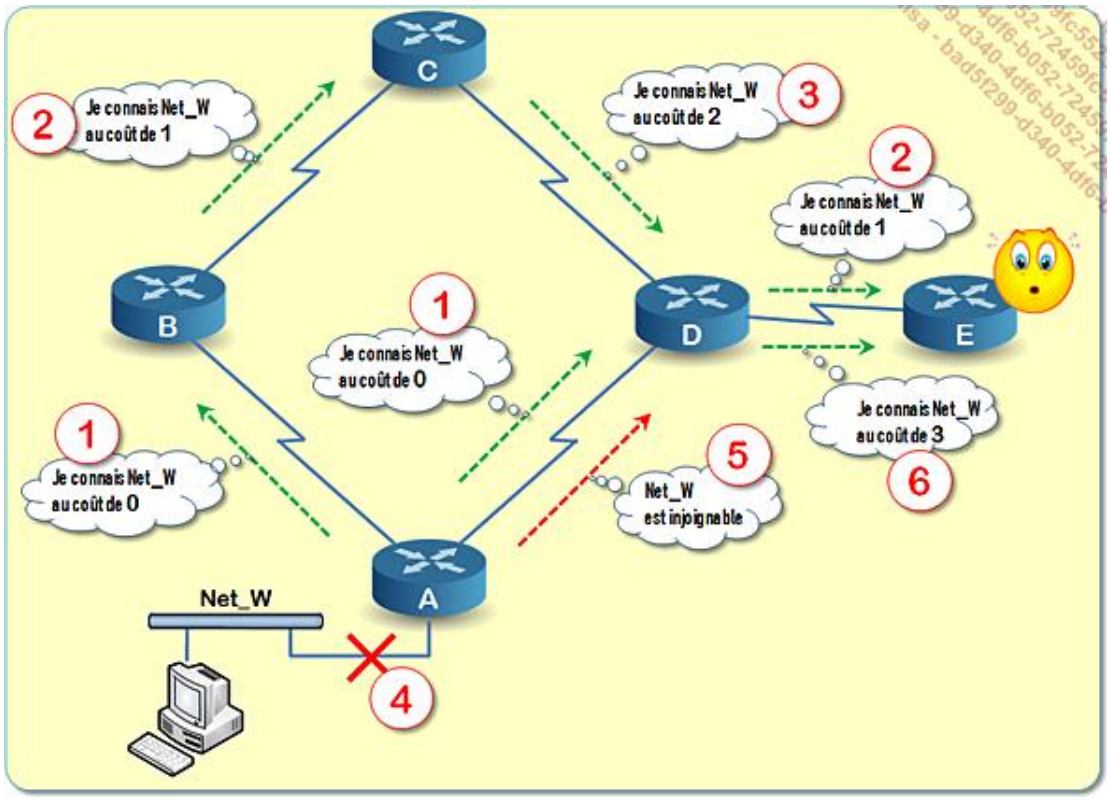
d. Mises à jour déclenchées

Jusqu'ici et quoiqu'il advienne, les mises à jour envoyées aux voisins l'étaient de façon régulière. Est-ce le comportement le plus pertinent ? Quand un réseau passe à l'état inactif, un routeur qui le perçoit doit attendre jusqu'à 30 secondes (cas de RIP) avant de transmettre cette précieuse information. Les mises à jour déclenchées (*Triggered Updates* ou *Flash Updates*) apportent une solution des plus simples à ce manque de réactivité. L'idée consiste à provoquer l'émission d'une mise à jour sans attendre l'expiration du temporisateur de mise à jour, ce dès qu'un changement de coût se produit sur une route qu'il s'agisse d'un changement à la hausse ou à la baisse. L'installation d'une nouvelle route dans la table de routage est considérée comme un changement de coût et provoque également une mise à jour déclenchée. Le temps de convergence s'en trouve évidemment très amélioré de même que le problème du comptage à l'infini devient un problème assez marginal. Le choix de provoquer des mises à jour déclenchées ne fait pas disparaître le besoin de mises à jour régulières pour deux raisons : il faut empêcher le temporisateur d'invalidation de route d'expirer et il faut également prévoir le cas où un routeur reçoit une mise à jour corrompue issue d'un routeur qui n'a pas achevé sa convergence, la mise à jour régulière rétablirait alors la situation. Quoi qu'il en soit :

Les mises à jour déclenchées contribuent à accélérer la convergence du réseau.

e. Refus de mises à jour (HolddownTimer)

Jusqu'à présent, nous n'avons placé un peu d'intelligence que dans la façon dont un routeur émettait ses mises à jour de routage : de façon régulière, plus ou moins fréquemment, sur évènement, totales, partielles, partielles avec routes inverses empoisonnées. Mais nous n'avons rien fait sur la réception des mises à jour. Toute mise à jour est également considérée. Ne conviendrait-il pas d'introduire un peu de scepticisme ? Considérez le contexte ci-dessous :



- Une fois que l'ensemble du réseau a convergé, le routeur D annonce le réseau W au routeur E avec un coût de 1.
- Étiquette 4 : le réseau W passe à l'état inactif.
- Étiquette 5 : le routeur A annonce W comme étant injoignable aux routeurs B et D.
- Le routeur C connaît annonce connaître W au coût de 2, le routeur D inscrit une route vers W via C dans sa table de routage au coût de 3.

- Étiquette 6 : Le routeur E reçoit une mise à jour du routeur D. Dans cette mise à jour, le réseau W est annoncé avec un coût de 3, ce coût était à 1 dans la mise à jour précédente.

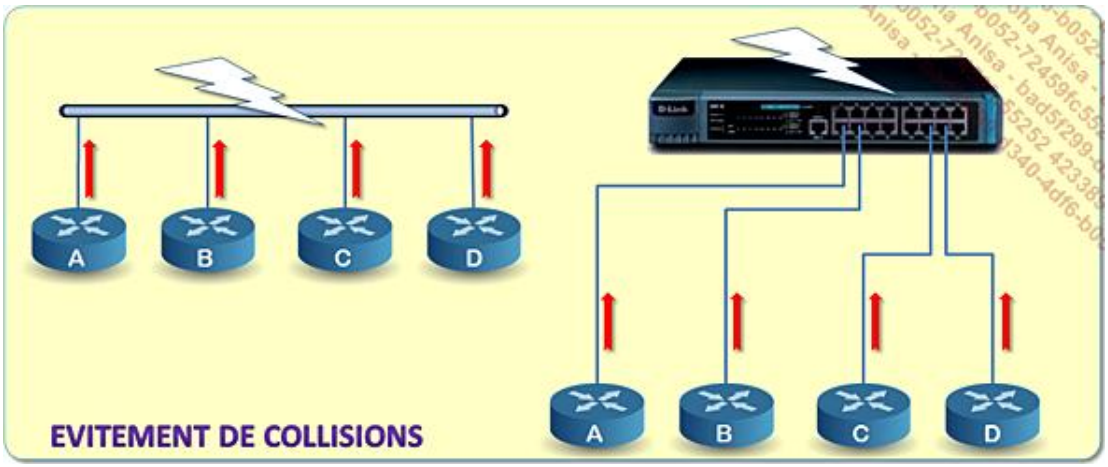
Les quatre routeurs A, B, C et D sont lancés dans un comptage à l'infini. Le coût annoncé par le routeur D va donc progressivement augmenter pour atteindre 16. Mais dans les faits, le réseau W est injoignable. Très légitimement, le routeur E devrait donc se méfier de la route annoncée par le routeur D. Mais comment éveiller la méfiance du routeur E ? En s'appuyant sur le seul signe tangible du point de vue du routeur E, à savoir l'augmentation du coût annoncé vers le réseau W.

On décide donc que chaque fois qu'un routeur reçoit une mise à jour dans laquelle le coût d'une route déjà connue augmente, le routeur arme un temporisateur dit de retenue et refuse toute mise à jour à coût égal ou augmenté pour cette route tant que ce temporisateur n'a pas expiré. En revanche, une mise à jour reçue avec un coût meilleur que le coût original provoque la désactivation du temporisateur de retenue. De la même façon, une mise à jour reçue avec la route marquée injoignable (coût 16) devrait désactiver le temporisateur de retenue afin que le routeur participe à la propagation de la mauvaise nouvelle et donc éviter que ce temporisateur ne dégrade le temps de convergence.

Ce temporisateur n'est pas prévu par le RFC et son implémentation est le fait de CISCO. Le comportement décrit est donc susceptible de varier selon la version de l'IOS.

4. Collisions de mises à jour

Considérez le contexte ci-dessous :



Les routeurs de la topologie mettent en œuvre le même protocole de routage DV. Chacun des routeurs émet donc sa mise à jour de façon régulière, les périodes qui séparent deux émissions sont identiques sur tous les routeurs. Une probabilité existe pour que deux ou davantage de ces mises à jour se produisent en même temps, provoquant ainsi une collision de mises à jour. N'oublions pas que ces mises à jour sont diffusées et rappelons-nous également le comportement d'un commutateur LAN quand il doit faire progresser une trame diffusée : il se comporte alors comme un banal concentrateur. Autrement dit, la collision est possible. En outre si la collision se produit, les émetteurs impliqués retardent leur émission ce qui la rapproche de l'émission des mises à jour à venir émanant des autres routeurs non impliqués jusque-là. On s'installe dans un phénomène cumulatif. De plus, si cette collision ne se produit pas aujourd'hui, elle se produira peut-être demain car les horloges des temporisateurs dériveront inévitablement.

La solution consiste à abandonner partiellement la période fixe des mises à jour, en établissant le temps initial du temporisateur de mise à jour avec un temps fixe additionné d'un temps aléatoire (*timing jitter*).

RIP

RFC utiles :

RFC 1058	**Routing Information Protocol**	Juin 1988

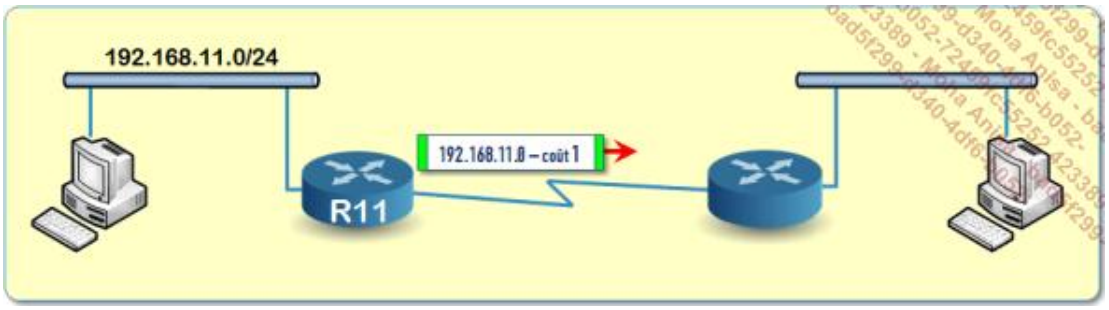
Impossible de prétendre mener une étude sérieuse des protocoles de routage dans un contexte IP sans débuter par le protocole RIP (*Routing Information Protocol*). RIP fut développé par la compagnie Xerox au début des années 1980 comme partie intégrante de son architecture réseau XNS. RIP fournit le résultat attendu à la condition de cantonner son utilisation à des réseaux de petite et moyenne (moins) envergure et de fait est très largement répandu. Les défauts de RIP sont bien connus : le nombre de sauts entre deux hôtes est limité à 16 et le protocole est long à converger. La métrique, constituée par le nombre de sauts, n'est pas intelligente en ce sens qu'elle ne tient compte ni de la bande passante des liens ni de leur charge. Il ne serait pas compréhensible qu'une entreprise ayant en charge un large réseau ne migre pas ou n'ait pas déjà migré vers un protocole de routage plus sophistiqué.

1. Le protocole, messages échangés

Les messages RIP sont encapsulés dans des datagrammes UDP. Les deux numéros de port, source et destination, d'un datagramme UDP qui transporte RIP sont établis à la même valeur 520. RIP utilise deux types de messages : la requête et la réponse :

- La requête RIP permet à un routeur d'interroger ses voisins afin d'obtenir une mise à jour.
- La réponse RIP transporte la mise à jour.

La métrique utilisée par RIP est le nombre de sauts. Un routeur annonce une route directement connectée avec une métrique égale à 1 :



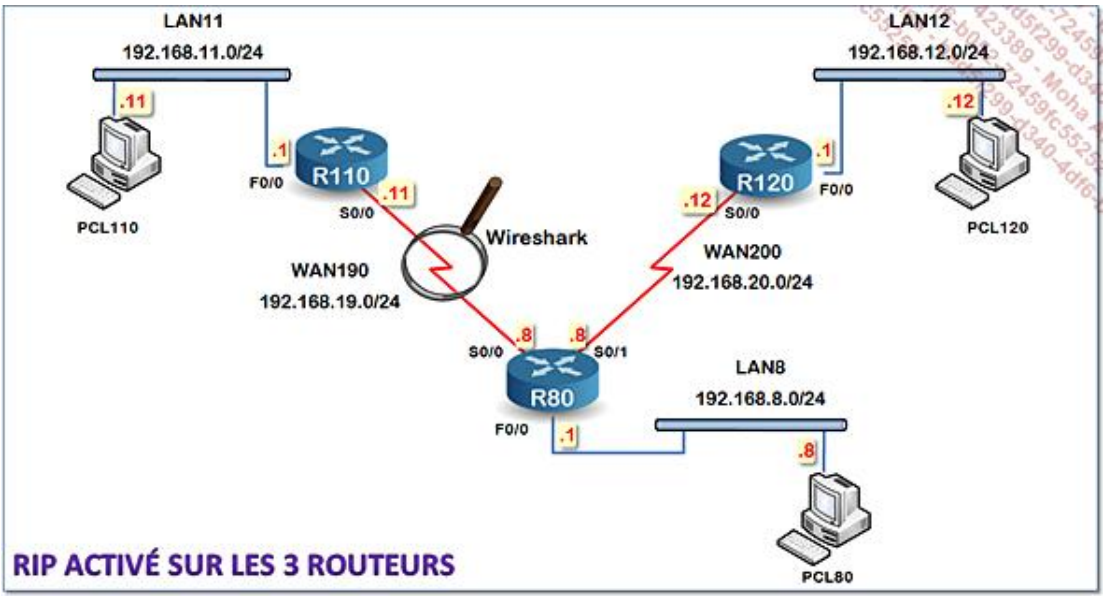
Une métrique de 16 signifie que le réseau est injoignable.

Au démarrage, un routeur RIP diffuse une requête RIP sur l'ensemble de ses interfaces participant au protocole. Puis le processus RIP se place à l'écoute des messages RIP en provenance des autres routeurs. Chaque routeur voisin qui reçoit la requête génère un message de réponse RIP et y place un extrait de sa table de routage. En régime établi, toutes les 30 secondes environ et indépendamment de toute requête, chaque routeur RIP génère de façon gratuite un message de réponse RIP et y place également un extrait de sa table de routage.

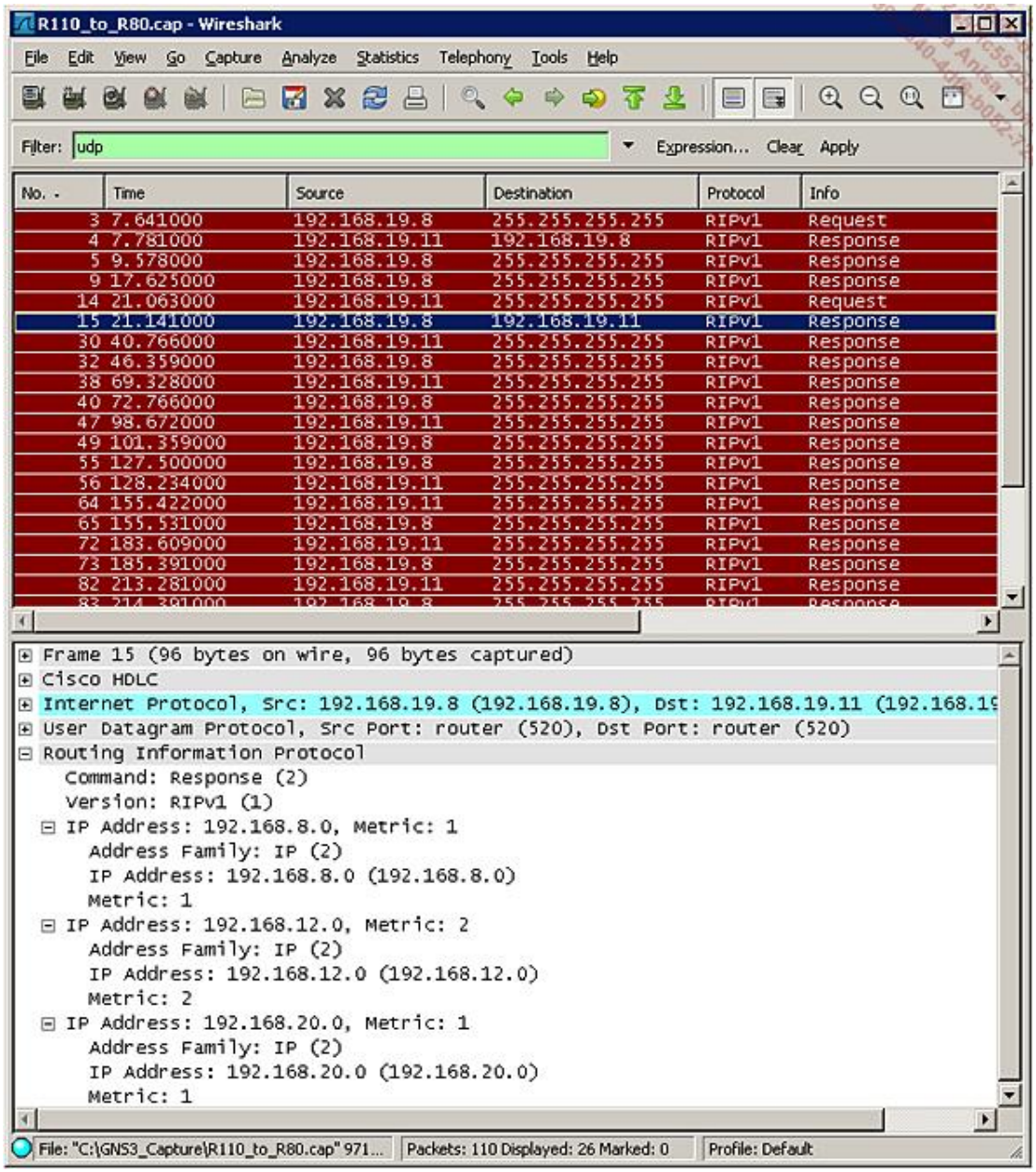
L'extrait de la table de routage contient toutes les routes à l'exception de celles supprimées par la règle de partage d'horizon. Tout message de réponse RIP est traité de manière identique, qu'il ait été obtenu suite à une requête RIP ou de façon gratuite :

- Si la réponse contient une route jusqu'ici inconnue, le routeur la place dans sa propre table de routage via le routeur qui l'a annoncé. L'adresse de prochain saut est apprise en lisant l'adresse source du message de réponse RIP.
- Si la réponse contient une route déjà connue (elle est déjà présente dans la table de routage), l'entrée existante n'est remplacée que si la route annoncée dispose d'un meilleur coût.
- Quand une route déjà connue est annoncée avec un coût plus élevé par le routeur de prochain saut, la suspicion s'installe et cette route est marquée injoignable pendant un temps d'observation appelé temps de retenue (*holddown timer*). Si à l'expiration de ce temps, le routeur de prochain saut persiste à annoncer cette route avec ce nouveau coût moins favorable, alors la nouvelle métrique est acceptée et remplace l'ancienne dans la table.

Appuyons notre propos sur le contexte suivant :



Observez la capture ci-dessous réalisée sur lien « serial » entre R110 et R80 (cap_2E_01.pcap disponible en téléchargement sur le site des Editions ENI) :

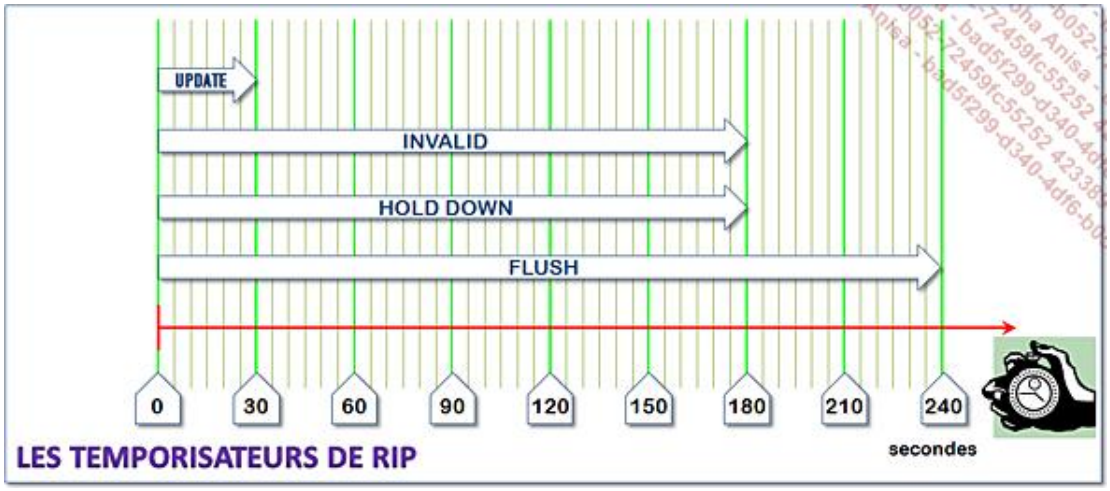


Dans Wireshark, pour obtenir l'affichage des seuls échanges RIP, il suffit d'activer un filtre, par exemple : **Analyze - Display Filters... - UDP only - OK**. R80 vient de terminer son démarrage, ce qui explique la requête RIP en trame 3. Observez que R110 répond à la requête dans un datagramme unicast. Vérifiez la période qui sépare deux émissions de mises à jour. Par exemple pour R80, les « *timestamps* » sont [46.35, 72.76, 101.36, 127.5, 155.42, 185.39, 214.39] espacés respectivement de [26.41, 28.60, 26.14, 27.92, 29.97, 29] secondes. On observe là l'effet de la variable aléatoire RIP_JITTER entretenue par l'IOS. Ce temps aléatoire est compris entre 0 et 15% de la période nominale qui sépare les mises à jour. À chaque nouvelle émission de mise à jour, l'IOS calcule ce temps puis le soustrait du temps nominal pour établir le temporisateur de la prochaine mise à jour. Puisque dans le cas présent, la période nominale est celle par défaut, soit 30 secondes, ceci explique que le temps observé entre deux mises à jour soit variable mais toujours compris dans l'intervalle [25.5 secondes - 30 secondes]. On se souvient qu'il s'agit de prévenir les collisions de mises à jour.

La partie inférieure de la fenêtre détaille le contenu de la mise à jour pour la trame 15. C'est R80 qui s'exprime et on voit que la règle de partage d'horizon est appliquée. Ne sont annoncés que les trois réseaux 192.168.8.0, 192.168.12.0 et 192.168.20.0. Observez enfin que la métrique annoncée par R80 pour un réseau directement connecté s'établit à 1.

a. Les temporisateurs de RIP

Le processus RIP entretient quatre temporisateurs :



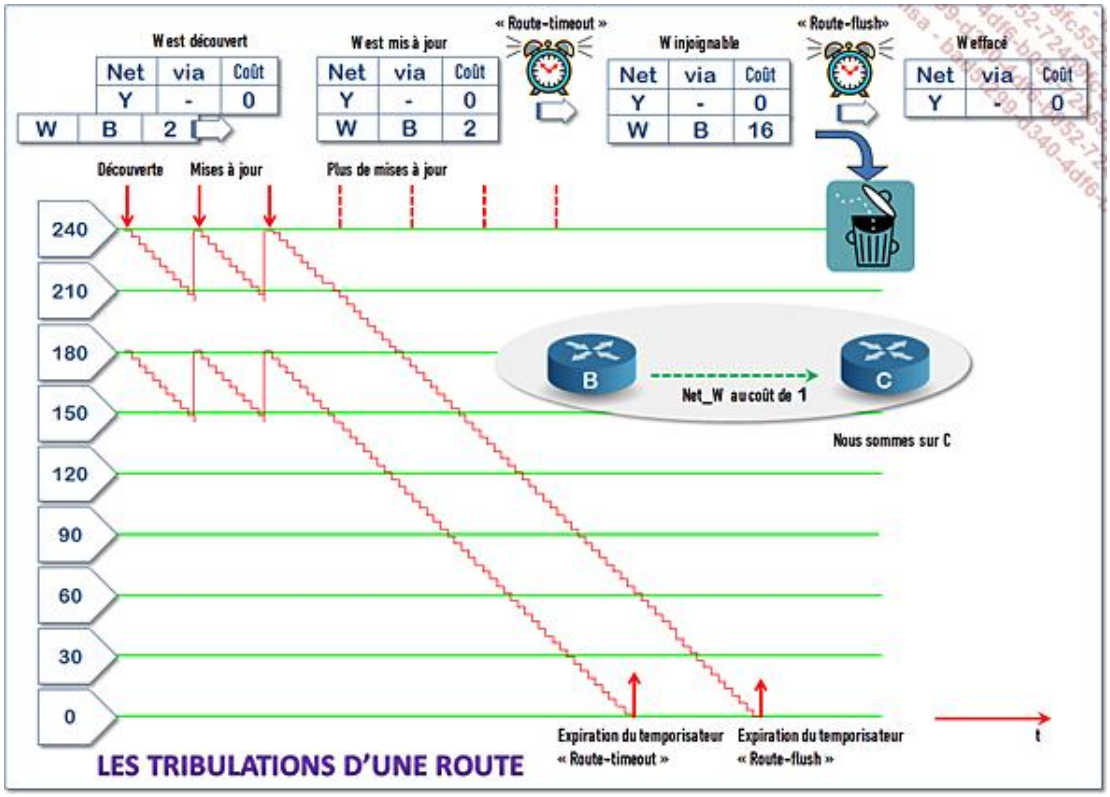
Mise à jour (*Update*) : période nominale qui sépare deux mises à jour. Le temporisateur de mise à jour est initialisé avec un temps résultant de la période nominale à laquelle l'IOS a soustrait le temps aléatoire RIP_JITTER (voir détail ci-avant).

Invalidation (*Route-timeout*) : ce temporisateur définit combien de temps une route peut rester dans la table de routage sans que le routeur reçoive de mise à jour la concernant. Également appelé « *Route-timeout timer* », il y a autant d'instances de ce temporisateur que de routes présentes dans la table de routage. Quand le temporisateur d'invalidation expire, la route est marquée injoignable (métrique 16) mais reste dans la table jusqu'à expiration du temporisateur d'effacement « *Route-flush timer* ». Le temporisateur d'invalidation est rétabli à sa valeur initiale (défaut 180 secondes) chaque fois que le routeur reçoit une mise à jour pour la route concernée.

Retenue (*Hold down*) : ce temporisateur n'est pas prévu dans le standard RFC 1058 mais l'IOS l'utilise. Chaque fois qu'un routeur reçoit une mise à jour dans laquelle le coût d'une route déjà connue augmente, le routeur arme ce temporisateur et refuse toute mise à jour à coût augmenté pour cette route tant que ce temporisateur n'a pas expiré. Pendant ce temps, le routeur considère la route comme peut-être injoignable mais continue de se comporter comme si elle l'était encore, c'est-à-dire continue d'acheminer les paquets conformément à l'information dont il disposait lors de l'entrée dans l'état « *hold down* ». Un routeur qui a armé un temporisateur de retenue peut le désactiver en plusieurs circonstances : la route est à nouveau annoncée au coût initial, la route est annoncée avec un coût de 16. Ce comportement est susceptible de varier selon les versions d'IOS.

Effacement (*Route-flush*) : ce temporisateur est appelé « *garbage collection timer* » par le RFC (littéralement temporisateur de collecte des ordures !). La valeur par défaut est établie à 240 secondes, soit 60 secondes au-delà du temporisateur d'invalidation. Pendant ces 60 secondes, la route est marquée injoignable mais reste dans la table. Elle est donc annoncée avec la métrique injoignable (16) dans les mises à jour de routage pendant ces 60 secondes. À l'expiration du temporisateur d'effacement, la route est effacée de la table de routage.

La figure suivante résume le comportement des temporisateurs du routeur RIP :



Les quatre temporisateurs peuvent être manipulés à l'aide de la commande **timers basic** en mode configuration de routeur :

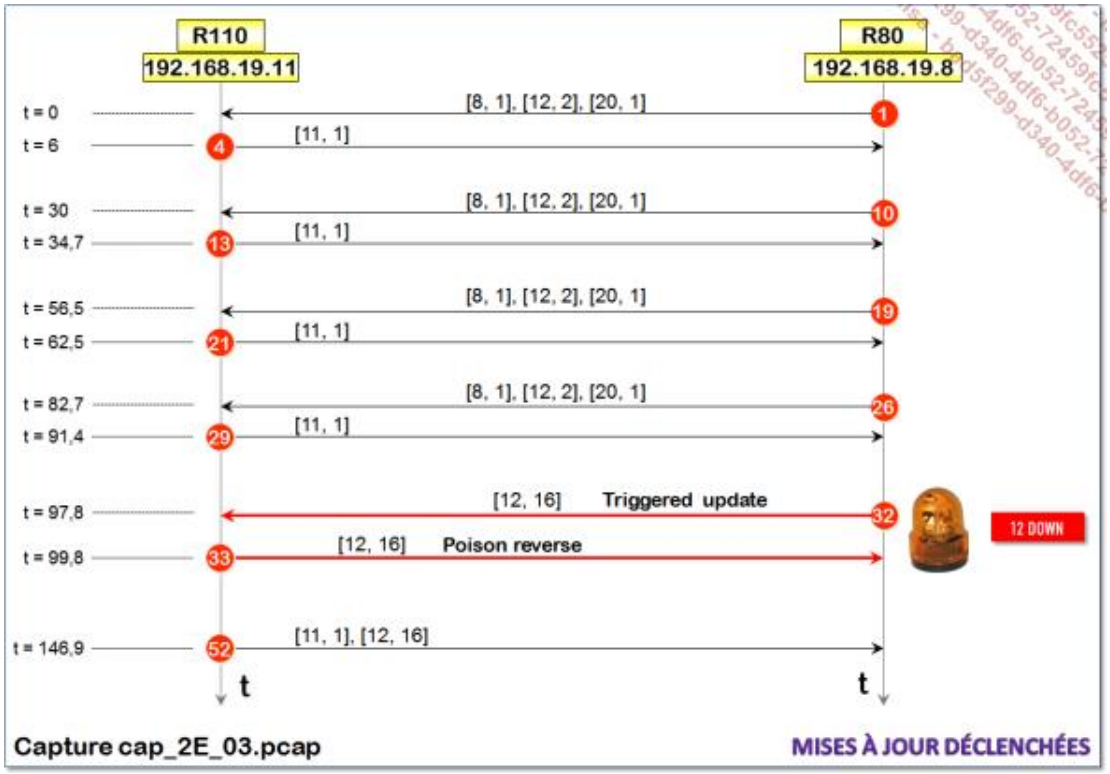
```
R110(config)#router rip
R110(config-router)#?
Router configuration commands:
.........
timers                  Adjust routing timers
.........

R110(config-router)#timers ?
basic  Basic routing protocol update timers

R110(config-router)#timers basic ?
<0-4294967295>  Interval between updates

R110(config-router)#timers basic 30 ?
<1-4294967295>  Invalid

R110(config-router)#timers basic 30 180 ?
<0-4294967295>Holddown

R110(config-router)#timers basic 30 180 180 ?
<1-4294967295>  Flush

R110(config-router)#timers basic 30 180 180 240 ?
<cr>

R110(config-router)#timers basic 30 180 180 240
R110(config-router)#^Z
R110#
```

Un routeur ne peut faire fonctionner qu'un seul processus RIP et cette commande ajuste les temporisateurs pour le processus dans son entier. Changer une ou davantage des quatre valeurs ne devrait être entrepris qu'avec beaucoup de prudence. De plus, pour que le fonctionnement de RIP reste cohérent, le changement souhaité doit être appliqué à l'ensemble des routeurs du domaine RIP. Il est possible de vérifier la valeur actuelle des temporisateurs (ainsi que beaucoup d'autres informations du protocole) à l'aide d'une commande **show ip protocols** :

```
R110#sh ip protocols
```

```
Routing Protocol is "rip"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
Sending updates every 30 seconds, next due in 18 seconds
  Invalid after 180 seconds, hold down 180, flushed after 240
  Redistributing: rip
  Default version control: send version 1, receive any version
    Interface             Send  Recv  Triggered RIP  Key-chain
    FastEthernet0/0       1     1 2
    Serial0/0             1     1 2
  Automatic network summarization is in effect
  Maximum path: 4
  Routing for Networks:
    192.168.11.0
    192.168.19.0
  Routing Information Sources:
    Gateway         Distance      Last Update
192.168.19.8         120      00:00:09
  Distance: (default is 120)

R110#
```

RIP met à profit le partage d'horizon avec empoisonnement de routes inverses ainsi que les mises à jour déclenchées. Plutôt que de l'affirmer, vérifions-le. Le contexte est inchangé et Wireshark capture toujours les mises à jour RIP sur le lien « *serial* » entre R110 et R80. Sur l'interface f0/0 du routeur R120, l'administrateur a provoqué une commande **shutdown** suivie d'une commande **no shutdown** quelques secondes plus tard. La capture cap_2E_03.pcap est disponible sur le site des Editions ENI.



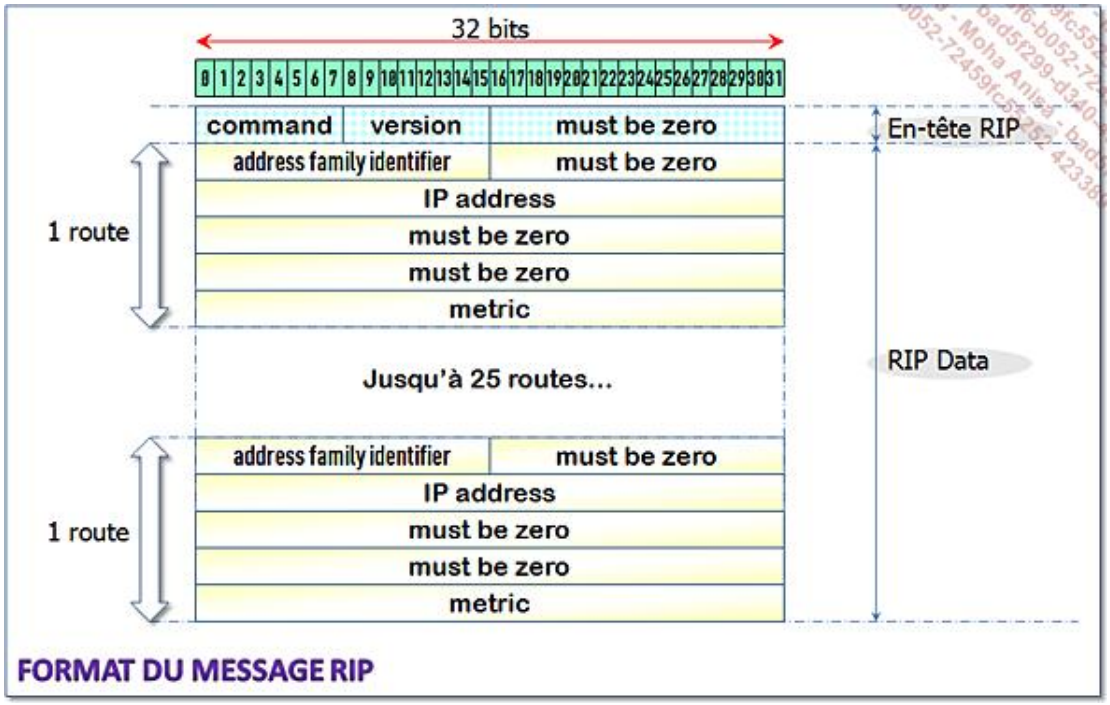
La première des deux informations entre crochets correspond à la route. Notre modeste mise en situation comporte les routes 11, 8, 12, 19, 20 (en fait 192.168.11.0/24, 192.168.8.0/24...). La seconde information correspond au coût annoncé exprimé en nombre de sauts. Observez la fonctionnalité de partage d'horizon (le routeur R80 n'envoie vers la gauche que les routes apprises par la droite). Environ 15 secondes après la mise à jour normale en trame 26, la route 12 devient injoignable. Sans attendre la mise à jour suivante, le routeur R120 génère une mise à jour déclenchée qui ne comporte que la route 12, le coût annoncé est 16 ce qui signifie route injoignable. Le routeur R80 perçoit le changement de coût et réagit immédiatement en générant une mise à jour déclenchée en trame 32. En réponse, et c'est là la première manifestation de la faculté d'empoisonnement de routes inverses, R110 génère une mise à jour déclenchée qui comporte la même route au même coût. Aucun risque donc qu'un routeur s'imagine pouvoir passer par R110 pour joindre LAN12. Observez que les mises à jour déclenchées ne remettent pas en cause l'émission des mises à jour cadencées, dont le rythme immuable est fixé par le temporisateur de mise à jour.

Impossible à observer sur la capture précédente, un routeur qui perçoit un changement de coût suite à la réception d'une mise à jour déclenchée diffère sa propre émission de mises à jour déclenchées de 1 à 5 secondes, ce afin

d'éviter que l'évènement de départ (le changement de topologie) ne se transforme en tempête de mises à jour déclenchées. Pour les curieux insatiables, la capture cap_2E_04.pcap rapporte un scénario à peine différent : c'est l'augmentation du coût de la route qui provoque la mise à jour déclenchée. Cette augmentation a été provoquée par une manipulation directe de la métrique de RIP (patientez).

b. Format des messages

Le message RIP en version 1 (la version du RFC 1058) est un message à trous. On devine que la conception de son format a été influée par le transport possible de routes dans d'autres protocoles qu'IP et par la volonté de travailler sur des mots de 4 octets :



Les arguments du message RIP sont les suivants :

- **command** → pour ce qui nous concerne, deux valeurs possibles :
  - 1 : le message est une requête ;
  - 2 : le message est une réponse ;
  - Les autres valeurs sont soit obsolètes soit réservées.
- **version** → 1 pour RIP v1.
- **address family identifier** → 2 pour IP dans les messages de réponse, 0 dans un message de requête (patientez).
- **IP address** → contient indifféremment :
  - Une adresse hôte.
  - Une adresse de sous-réseau.
  - Une adresse réseau.
  - 0 qui indique une route par défaut.

- Le comportement de RIP face à ces différents cas est décrit dans la section suivante Comportement avec classe.

- **metric** → coût de la route exprimé en nombre de sauts de 1 à 16, 1 signifiant une route directement connectée, 16 caractérisant une route injoignable.

La taille maximale du message RIP est 512 octets, sans compter les en-têtes IP et UDP (pour mémoire, un en-tête IP sans options occupe 20 octets, un en-tête UDP occupe 8 octets). En ôtant l'en-tête RIP, il reste 508 octets utiles au transport de routes. Puisqu'une route occupe 20 octets, un message RIP peut transporter jusqu'à 25 routes. Somme toute, un datagramme UDP qui transporte un message RIP ne dépassera pas 512 octets (25*x*20 + 4 + en-tête UDP).

c. Format des requêtes

Le message RIP de requête permet d'obtenir tout ou partie de la table de routage du destinataire. Ceci amène à évoquer la notion d'hôte silencieux :

Un hôte silencieux ne génère pas de mises à jour RIP mais peut profiter des mises à jour qu'il perçoit.

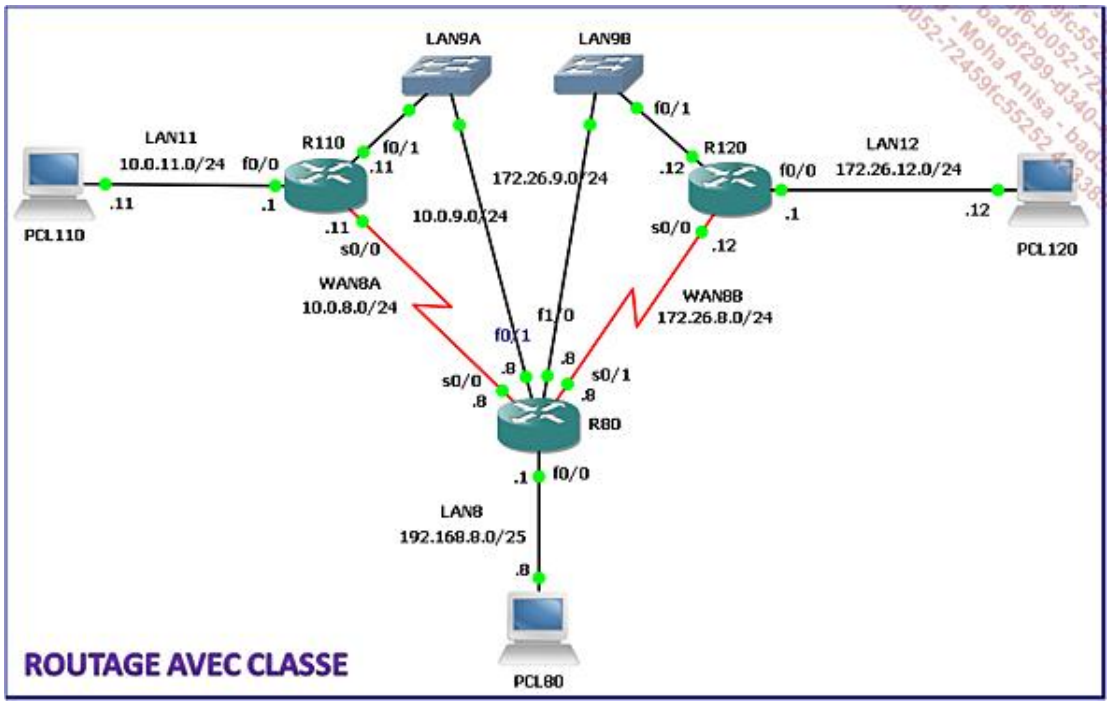
Une requête RIP est classiquement diffusée, le port source est 520. Un hôte silencieux ne répond pas à une telle requête. On peut pourtant le contraindre à le faire. Il faut pour cela générer une requête RIP dont le port source est différent de 520.

La requête peut comporter plusieurs questions, c'est-à-dire plusieurs réseaux, sous-réseaux, hôtes pour lesquels celui qui interroge souhaite découvrir si le destinataire dispose d'une route. Le destinataire prépare une réponse par question pour générer le message de réponse. À chaque question (chaque adresse demandée), le destinataire consulte sa table de routage et place la métrique trouvée dans la réponse à la question, 16 quand il n'a pas trouvé l'adresse demandée. Aucun traitement d'horizon partagé n'est effectué. Une requête de ce type est utile aux logiciels de diagnostic.

La requête peut concerner l'ensemble de la table de routage. Dans ce cas, le message de requête se distingue parce qu'il ne comporte qu'une seule question [adressfamily identifier = 0, IP address = 0.0.0.0, metric = 16]. Le destinataire génère un message de réponse RIP unicast vers l'adresse IP du demandeur. Paradoxalement, quand la question est « Tout », la réponse est partielle. En effet, il s'agit cette fois d'assurer un fonctionnement normal de RIP dans son domaine et la réponse respecte la règle du partage d'horizon ainsi que la règle de masquage des sous-réseaux (*summarization*, patientez).

d. Comportement avec classe (classfull)

Nous avons besoin d'un contexte un peu plus riche pour aborder ce qui va suivre. Comme à notre habitude maintenant, ce contexte a été préparé sous GNS3 :



Appuyons notre propos sur la table de routage de R80 :

```
R80#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     192.168.8.0/25 is subnetted, 1 subnets
C       192.168.8.0 is directly connected, FastEthernet0/0
     172.26.0.0/24 is subnetted, 3 subnets
R       172.26.12.0 [120/1] via 172.26.8.12, 00:00:27, Serial0/1
                    [120/1] via 172.26.9.12, 00:00:27, FastEthernet1/0
C       172.26.8.0 is directly connected, Serial0/1
C       172.26.9.0 is directly connected, FastEthernet1/0
     10.0.0.0/24 is subnetted, 3 subnets
R       10.0.11.0 [120/1] via 10.0.8.11, 00:00:14, Serial0/0
                  [120/1] via 10.0.9.11, 00:00:14, FastEthernet0/1
C       10.0.8.0 is directly connected, Serial0/0
C       10.0.9.0 is directly connected, FastEthernet0/1
R80#
```

Les routes marquées à l'aide de la lettre « R » sont issues du protocole de routage RIP. Pour mémoire, les deux valeurs entre crochets associées à chaque route sont la distance administrative, 120 dans le cas d'une route issue de RIP, et la métrique égale au nombre de sauts. Quand plus d'une route au même coût existent vers la même destination, RIP installe ces routes ensemble dans la table de routage et le routeur procède alors à un partage de charge à coût égal. C'est le cas de la route vers LAN11 ainsi que de la route vers LAN12 dans la capture précédente.

Le processus de routage extrait un paquet d'une file d'attente d'entrée. Imaginons qu'il s'agisse du premier paquet d'un nouveau flux vers une adresse de destination pas encore présente dans le cache de commutation rapide (relire si nécessaire la section Partage de charge par destination et « *Fast Switching* » du chapitre Le routage statique). Le routeur lit une première fois la table de routage en séquence à la recherche de la partie réseau de l'adresse de destination. Puisque RIP v1 ne sait faire qu'un routage avec classe, c'est le masque de classe A, B ou C (le masque naturel) qui est utilisé pour déterminer quelle est l'adresse réseau recherchée. Si le routeur ne trouve pas de correspondance pour un réseau majeur (un réseau de classe A, B ou C), le paquet est éliminé.

Si le routeur trouve une correspondance pour un réseau majeur, alors il recherche dans la table de routage les sous-réseaux connus de ce réseau majeur. Si une correspondance est trouvée, alors le paquet est confié à l'interface de sortie adéquate. Dans le cas contraire, le paquet est éliminé.

À chaque fois qu'un paquet est éliminé parce que le routeur n'a pas trouvé de correspondance dans la table de routage, le routeur génère un message ICMP type Destination injoignable vers la source du paquet.

Par expérience, nous touchons là un point de compréhension délicat. Ancrons ces notions à l'aide de quelques exemples :

- Un paquet destiné à l'adresse 192.168.11.5 entre sur R80. Le routeur ne trouve aucune correspondance vers le réseau majeur 192.168.11.0 et le paquet est supprimé.

- Un paquet destiné à l'adresse 172.26.6.12 entre sur R80. Le routeur trouve une correspondance vers le réseau majeur 172.26.0.0 et examine les sous-réseaux connus de ce réseau majeur. Hélas, aucune correspondance n'est trouvée vers le sous-réseau 172.26.6.0 et ce paquet est également éliminé.

- Enfin un paquet destiné à l'adresse 10.0.11.11 entre sur le routeur R80. Le routeur trouve une correspondance vers le réseau majeur 10.0.0.0 puis une correspondance vers le sous-réseau 10.0.11.0 et ce paquet est acheminé vers l'une des interfaces Serial 0/0 ou FastEthernet 0/1.

Mais la vraie question est comment un routeur exploite-t-il les mises à jour reçues pour compléter sa table de routage ? Observez à nouveau le format du message RIP : aucun champ n'est prévu pour transmettre un masque et c'est sans aucun doute la plus grave limitation de cette version 1. Comment les constructeurs de routeurs, dont Cisco, ont-ils contourné cette limitation ? Rappelons-nous comment procède le routeur pour ajouter une route directement connectée à la table de routage. Le routeur exploite pour ce faire la configuration IP de l'interface. Par exemple, dans le cas de R80, cette configuration :

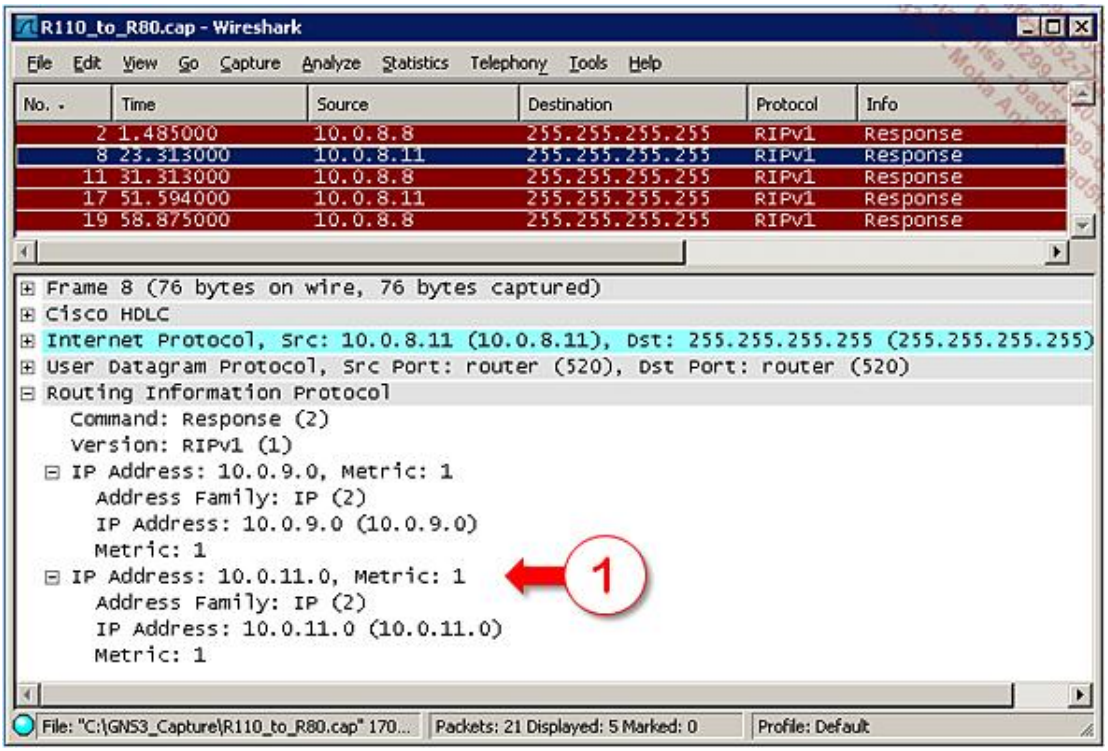
```
!
interface FastEthernet0/0
ipaddress 192.168.8.1 255.255.255.128
!
```

a provoqué l'ajout dans la table de cette route.

```
192.168.8.0/25 is subnetted, 1 subnets
C       192.168.8.0 is directly connected, FastEthernet0/0
```

Le router connaît le masque naturel /24 de l'adresse 192.168.8.1 et puisque le masque configuré est /25, le routeur déduit que le réseau auquel est connecté l'interface F0/0 est un sous-réseau de 192.168.8.0.

Cette méthode inspire celle utilisée par RIP : puisque les annonces de RIP ne comportent pas de masques, le routeur comble l'information manquante en la puisant sur ses interfaces quand c'est possible. Nous sommes toujours sur R80. Voici le message RIP reçu de R110, capturé sur le lien S0/0 de R80 :



Parmi les annonces de routes contenues dans ce message RIP, le routeur R80 reçoit [adresse IP 10.0.11.0, métrique 1] (étiquette 1). Cette information permet à R80 d'ajouter dans sa table de routage :

```
10.0.0.0/24 is subnetted, 3 subnets
R       10.0.11.0 [120/1] via 10.0.8.11, 00:00:14, Serial0/0
```

Comment R80 a-t-il procédé ? Il se trouve que l'adresse IP affectée à l'interface S0/0, interface qui reçoit le message, appartient donc au réseau majeur de classe A 10.0.0.0 :

```
!
interface Serial0/0
ipaddress 10.0.8.8 255.255.255.0
!
```

À l'adresse 10.0.11.0 reçue, R80 a appliqué le masque configuré sur l'interface qui reçoit, dans le cas présent /24. L'information orpheline [10.0.11.0/ ???] est devenue [10.0.11.0/24]. Puisque R80 est directement connecté aux deux sous-réseaux 10.0.8.0 par S0/0 et 10.0.9.0 par f0/1, ceci porte à 3 le nombre de sous-réseaux connus de R80 dans le réseau majeur 10.0.0.0, ce qui explique la mention :

```
    10.0.0.0/24 is subnetted, 3 subnets
```

dans la table de routage. Mais pourquoi afficher « *10.0.0.0/24 is subnetted* » et non pas « *10.0.0.0 is subnetted* » ? Observez à nouveau la topologie. Cette même annonce du réseau LAN11 d'adresse 10.0.11.0 est reçue à la fois par l'interface S0/0 et F0/1 de R80. La configuration de F0/1 est :

```
!
interface FastEthernet0/1
ip address 10.0.9.8 255.255.255.0
!
```

Ainsi l'adresse 10.0.11.0 se voit appliquer le même masque et ce, que l'annonce soit reçue via S0/0 ou F0/1. Imaginez le désastre si l'administrateur avait décidé d'utiliser une longueur de préfixe différente, par exemple /25, sur F0/1. Quelle route faut-il installer dans la table de routage : 10.0.11.0 /24 selon la configuration de S0/0 ou 10.0.11.0/25 selon la configuration de F0/1 ?

Autre cas de figure : l'administrateur rétablit le même masque sur S0/0 et F0/1 de R80 mais choisit une longueur de préfixe /25. La route ajoutée dans la table pointera vers le réseau 10.0.11.0/25 alors que le réseau affecté à LAN11 est effectivement 10.0.11.0/24. On aboutit ainsi à une obligation essentielle qui s'impose dans toute configuration de routage RIP avec classe :

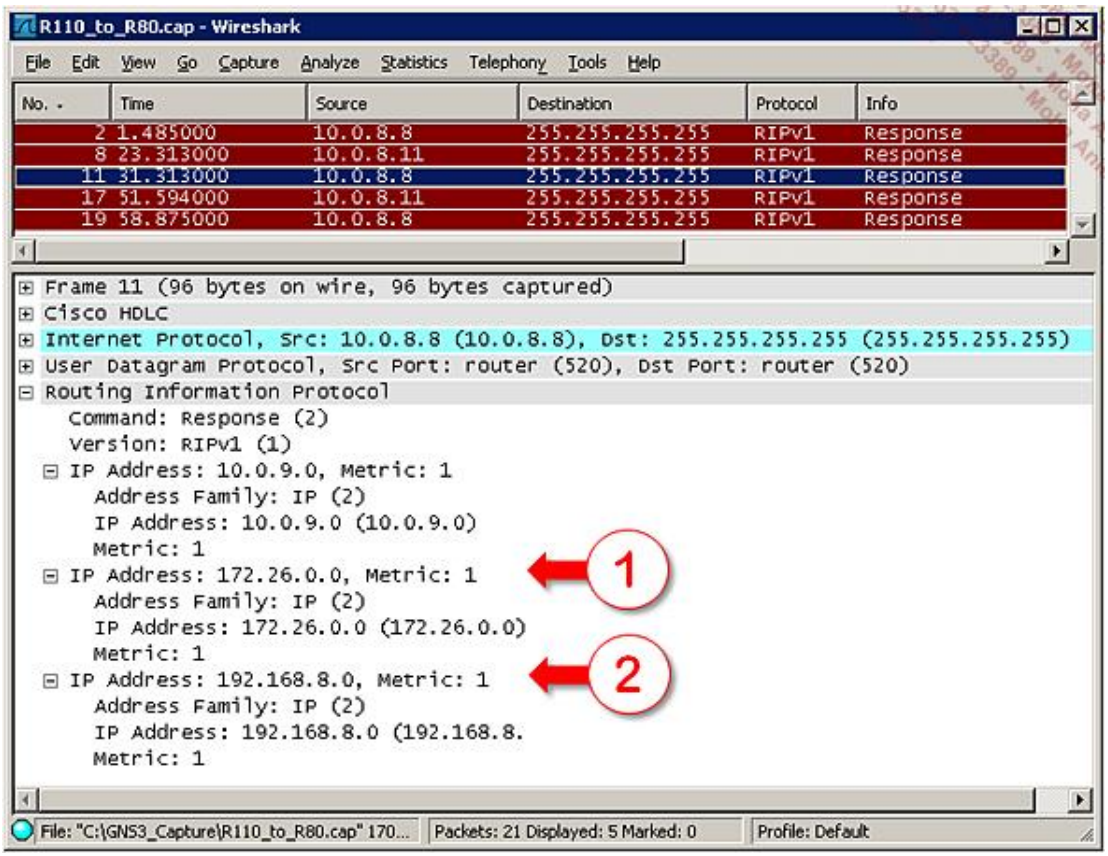
À l'intérieur d'un réseau majeur, les préfixes affectés doivent être identiques sur l'ensemble du domaine RIP.

À la question de départ « pourquoi afficher *10.0.0.0/24 is subnetted* dans la table de routage ? », la réponse est : « parce que l'ensemble des sous-réseaux du réseau majeur 10.0.0.0 découverts par R80 partagent la même longueur de préfixe ». C'est cette longueur qui est rappelée dans le résultat de la commande **show ip route**.

e. Résumé automatique de routes

Nous venons d'observer comment RIP met à profit la configuration de ses interfaces pour exploiter une annonce reçue et intégrer une route à la table de routage. Nous avions dit un peu plus haut : « le routeur comble l'information manquante en la puisant sur ses interfaces quand c'est possible ». Ce qui a rendu la chose possible dans l'explication précédente, c'est que les interfaces S0/0 et F0/1 étaient toutes deux connectées à des sous-réseaux du même réseau majeur 10.0.0.0. Qu'en est-il lorsqu'une interface reçoit une annonce RIP alors même qu'elle n'appartient pas à un sous-réseau du réseau majeur annoncé ? Exploiter le masque configuré sur l'interface n'aurait plus de sens !

L'ultime solution consiste à considérer le réseau majeur dans son entier et considérer que le routeur qui annonce ce réseau majeur est un routeur situé en bordure de ce réseau majeur. Pour nous en convaincre, observons le comportement de R80 quand il annonce vers le routeur R110 les réseaux de classe B et C auxquels il est connecté :



À nouveau, cette capture a été réalisée sur le lien S0/0 de R80. Pour mémoire, cette interface est directement

connectée au sous-réseau 10.0.8.0/24. Le message RIP comporte trois annonces :

- 10.0.9.0, métrique 1 → sous-réseau du réseau majeur 10.0.0.0, annoncé parce que l'interface S0/0 est directement connectée à un autre sous-réseau du même réseau majeur.

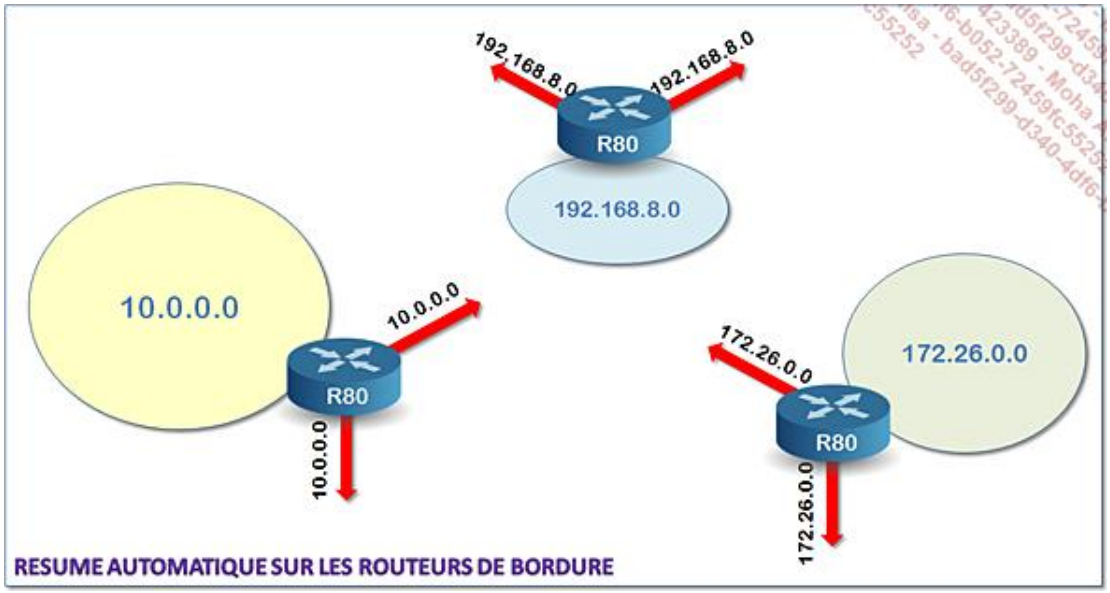
- Étiquette 1, 172.26.0.0, métrique 1 → réseau majeur de classe B 172.26.0.0. Le routeur R110 considérera que R80 est un routeur situé en bordure de ce réseau majeur et ajoutera à sa table de routage une route vers ce réseau majeur via 10.0.8.8 soit l'adresse IP source contenue dans le datagramme IP porteur du message RIP.

- Étiquette 2, 192.168.8.0, métrique 1 → réseau majeur de classe C 192.168.8.0. Le routeur R110 considérera que R80 est un routeur situé en bordure de ce réseau majeur et ajoutera à sa table de routage une route vers ce réseau majeur via 10.0.8.8.

Une capture de la table de routage de R110 confirme ce comportement :

```
R110#sh ip route
.........
Gateway of last resort is not set

R    192.168.8.0/24 [120/1] via 10.0.8.8, 00:00:22, Serial0/0
                    [120/1] via 10.0.9.8, 00:00:22, FastEthernet0/1
R    172.26.0.0/16 [120/1] via 10.0.8.8, 00:00:22, Serial0/0
                   [120/1] via 10.0.9.8, 00:00:22, FastEthernet0/1
     10.0.0.0/24 is subnetted, 3 subnets
C       10.0.11.0 is directly connected, FastEthernet0/0
C       10.0.8.0 is directly connected, Serial0/0
C       10.0.9.0 is directly connected, FastEthernet0/1
R110#
```

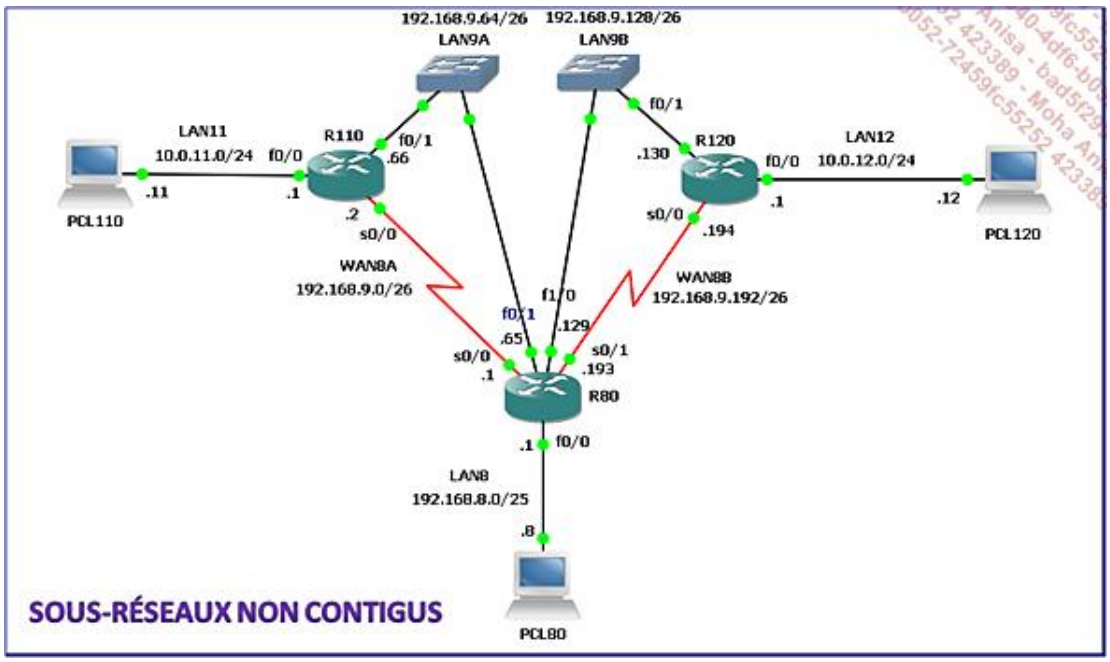
En final, R80 est situé en bordure de trois réseaux majeurs, ce qui l'amène à annoncer deux réseaux majeurs sur chacun de ses liens :



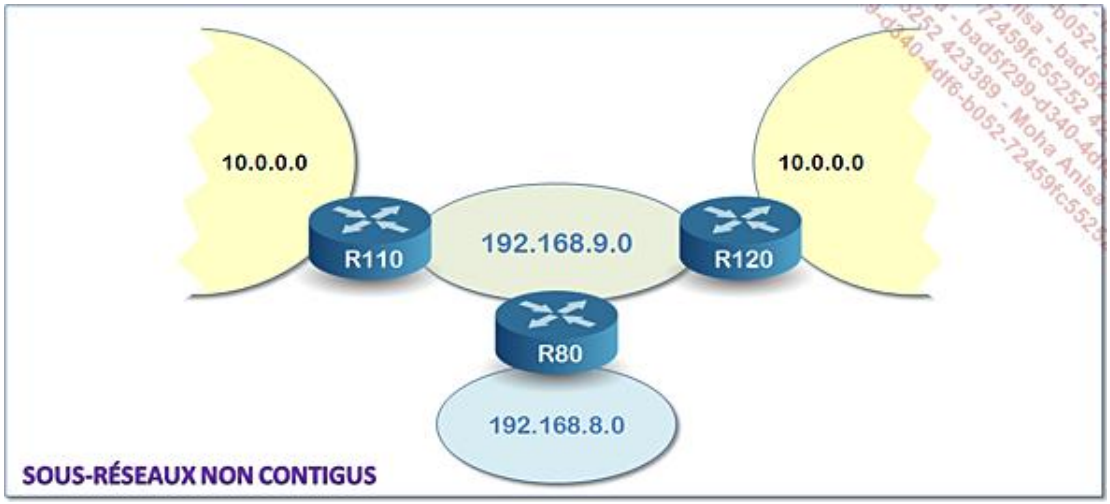
Ainsi, un routeur de bordure n'annonce pas de détails de sous-réseaux. Il ne servirait à rien de le faire puisque le routeur qui reçoit l'annonce serait incapable de les exploiter, faute de masque. On dit que le routeur de bordure procède à un résumé automatique.

f. Sous-réseaux non contigus

Considérez le contexte modifié ci-dessous :



D'un point de vue « réseaux majeurs », le contexte est le suivant :



Dans ce contexte, R110 va annoncer le réseau majeur 10.0.0.0 vers R80 ce qui en fait un routeur de bordure potentiel pour ce réseau. Hélas, R120 fait de même et l'appréhension est confirmée à la lecture de la table de routage de R80 :

```
R80#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
Gateway of last resort is not set

     192.168.8.0/25 is subnetted, 1 subnets
C       192.168.8.0 is directly connected, FastEthernet0/0
     192.168.9.0/26 is subnetted, 4 subnets
C       192.168.9.64 is directly connected, FastEthernet0/1
C       192.168.9.0 is directly connected, Serial0/0
C       192.168.9.192 is directly connected, Serial0/1
C       192.168.9.128 is directly connected, FastEthernet1/0
R    10.0.0.0/8 [120/1] via 192.168.9.66, 00:00:23, FastEthernet0/1
                [120/1] via 192.168.9.2, 00:00:23, Serial0/0
[120/1] via 192.168.9.194, 00:00:16, Serial0/1
                [120/1] via 192.168.9.130, 00:00:16, FastEthernet1/0
R80#
```

Un paquet destiné à LAN11 et qui arriverait d'on ne sait où sur R80 n'aurait qu'une chance sur deux d'être acheminé sur le réseau destinataire. On met ainsi en lumière une nouvelle contrainte qui découle du comportement « résumé

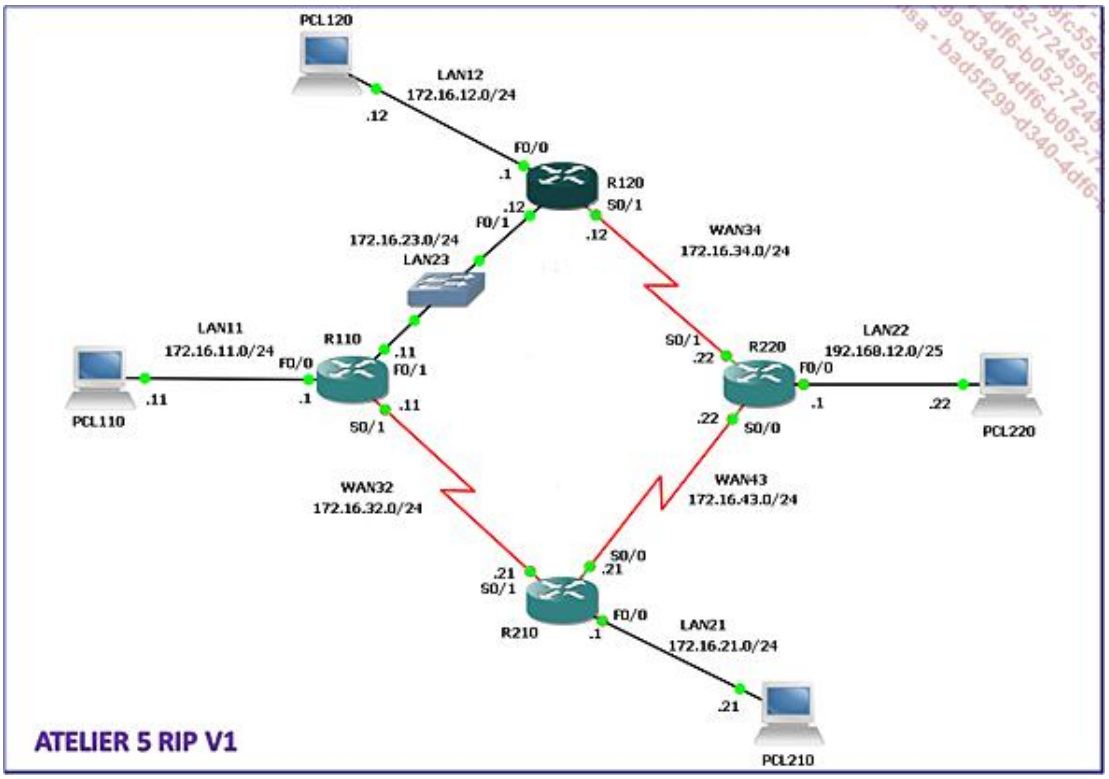
automatique de routes » décrit précédemment :

Un routeur qui annonce un réseau majeur doit être la route unique vers ce réseau. Ceci contraint à ne pas séparer plusieurs sous-réseaux d'un même réseau majeur par des sous-réseaux issus de réseaux majeurs différents.

En bref, il est nécessaire de maintenir la continuité du domaine couvert par les sous-réseaux d'un réseau majeur. Cette contrainte peut être atténuée en rétablissant la continuité du réseau majeur 10.0.0.0. Une solution consiste par exemple à attribuer aux liens qui séparent R110 et R120 de R80 des adresses IP secondaires extraites de sous-réseaux appartenant au même réseau majeur 10.0.0.0. Mais cela ne fait pas disparaître la contrainte d'une longueur de préfixe unique, ce qui aboutira dans l'exemple à affecter un sous-réseau /24 à chaque lien « *serial* » alors qu'un tel lien n'a besoin que de deux adresses.

2. Configuration de base

RIP est simple, sa configuration également. Démonstration sur cette topologie :



a. Activation du protocole

Le protocole s'active en deux temps :

1. Activer le processus RIP proprement dit à l'aide de la commande **router rip**.

2. Indiquer au processus quels réseaux majeurs doivent être pris en compte à l'aide d'une commande **network** par réseau majeur.

Toutes les interfaces ont déjà été configurées. Sur R110, R120 et R210, une seule commande **network** suffit. Sur R220, routeur de bordure des deux réseaux majeurs 172.16.0.0 et 192.168.12.0, deux commandes **network** sont nécessaires :

R110

```
R110(config)#router rip
R110(config-router)#network 172.16.0.0
R110(config-router)#^Z
R110#
```

R120

```
R120(config)#router rip
R120(config-router)#network 172.16.0.0
R120(config-router)#^Z
R120#
```

R210

```
R210(config)#router rip
R210(config-router)#network 172.16.0.0
R210(config-router)#^Z
R210#
```

R220

```
R220(config)#router rip
R220(config-router)#network 192.168.12.0
R220(config-router)#network 172.16.0.0
R220(config-router)#^Z
R220#
```

Ainsi, il est inutile de vouloir configurer la commande **network** avec un quelconque sous-réseau. La nature sans classe de RIP v1, le fait que RIP cache les détails du découpage en sous-réseaux à l'extérieur des frontières des réseaux majeurs imposent de ne configurer que des réseaux majeurs de classe A, B ou C.

Sur un routeur donné, toute interface dont l'adresse IP appartient à l'un des réseaux majeurs déclarés à l'aide d'une commande **network** participe au protocole.

La commande **debug ip rip** permet de se passer de Wireshark pour comprendre l'activité du protocole. Par exemple sur R220 :

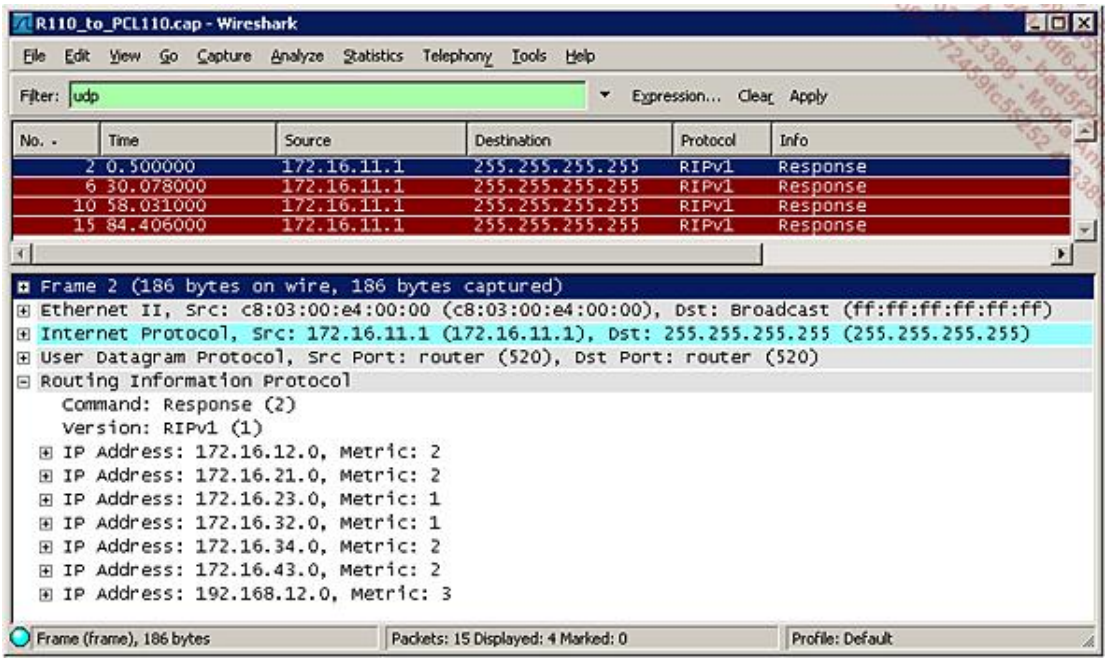
```
R220#debug ip rip
RIP protocol debugging is on
00:03:43: RIP: received v1 update from 172.16.43.21 on Serial0/0
00:03:43:      172.16.11.0 in 2 hops
00:03:43:      172.16.21.0 in 1 hops
00:03:43:      172.16.23.0 in 2 hops
00:03:43:      172.16.32.0 in 1 hops
00:03:52: RIP: received v1 update from 172.16.34.12 on Serial0/1
00:03:52:      172.16.11.0 in 2 hops
00:03:52:      172.16.12.0 in 1 hops
00:03:52:      172.16.23.0 in 1 hops
00:03:52:      172.16.32.0 in 2 hops
00:03:58: RIP: sending v1 update to 255.255.255.255 via F0/0 (192.168.12.1)
00:03:58: RIP: build update entries
00:03:58:       network 172.16.0.0 metric 1  !>>> Résumé automatique
00:03:58: RIP: sending v1 update to 255.255.255.255 via Serial0/0 (172.16.43.22)
00:03:58: RIP: build update entries
00:03:58:       subnet 172.16.12.0 metric 2
00:03:58:       subnet 172.16.23.0 metric 2
00:03:58:       subnet 172.16.34.0 metric 1
00:03:58:       network 192.168.12.0 metric 1 !>>> Résumé automatique
00:03:58: RIP: sending v1 update to 255.255.255.255 via Serial0/1 (172.16.34.22)
00:03:58: RIP: build update entries
00:03:58:       subnet 172.16.21.0 metric 2
00:03:58:       subnet 172.16.32.0 metric 2
00:03:58:       subnet 172.16.43.0 metric 1
00:03:58:       network 192.168.12.0 metric 1
R220#
```

Ce routeur a été choisi parce qu'il est routeur de bordure. Observez par exemple l'annonce faite sur l'interface F0/0 : cette interface n'appartient pas au réseau majeur 172.16.0.0. RIP annonce donc le réseau majeur 172.16.0.0, métrique 1 qui résume l'ensemble des sous-réseaux 172.16.x.0/24. De la même façon, R220 annonce le réseau majeur 192.168.12.0 sur les interfaces appartenant au réseau majeur 172.16.0.0.

Observez également les effets de la règle de partage d'horizon. Par exemple, R220 reçoit l'annonce du réseau 172.16.21.0 sur son interface S0/0 avec une métrique de 1. R220 n'annonce pas ce réseau dans la mise à jour qu'il prépare puis émet sur cette même interface. En revanche, 172.16.21.0 fait bien partie des réseaux annoncés dans la mise à jour que R220 génère sur S0/1 associé à une métrique incrémentée à 2.

b. Commande passive interface

Chacun des réseaux d'extrémité LAN11, LAN12, LAN21 et LAN22 n'est relié qu'à un seul routeur. Une capture avec Wireshark sur LAN11 confirme que les mises à jour sont bien diffusées alors même qu'aucun autre routeur n'en profitera :



On se souvient que émettre un datagramme revient pour une station à se poser une seule question : l'adresse du destinataire est-elle directement connectée ou pas ? Dans le premier cas, le datagramme est envoyé directement au destinataire. Dans le second cas, il est confié à la passerelle, charge à elle de le faire progresser vers sa destination. Les stations d'extrémité pratiquent donc un routage extrêmement simple en remettant à la passerelle tout paquet qui n'est pas destiné au réseau dont il est issu. Par conséquent, elles n'ont pas besoin de recevoir les mises à jour de routage RIP. Il est pourtant très facile de transformer une station Windows en hôte RIP silencieux. Il suffit d'activer l'écouteur RIP : **Panneau de configuration - Ajout/Suppression de programmes - Ajouter ou supprimer des composants Windows - Service de mise en réseau - Détails** et cocher la case **Ecouteur RIP**. C'est ce qui a été fait sur une station XP ajoutée sur LAN11 de notre contexte. L'adresse affectée à la station est 172.16.11.12. Une commande **route print** dans une invite de commandes confirme que la station met bien à profit les messages RIP issus de R110 :

```
C:\ >route print
===========================================================================
Liste d'Interfaces
0x1 ........................... MS TCP Loopback interface
0x4 ...00 0c 29 ce e7 00 ...... VMware Accelerated AMD PCNet Adapter -
Miniportd'ordonnancement de paquets
===========================================================================
Itinéraires actifs :
Destination réseau    Masque réseau  Adr. passerelleAdr. interface Métrique
          0.0.0.0          0.0.0.0      172.16.11.1    172.16.11.12       10
        127.0.0.0        255.0.0.0        127.0.0.1       127.0.0.1       1
      172.16.11.0    255.255.255.0     172.16.11.12    172.16.11.12       10
     172.16.11.12  255.255.255.255        127.0.0.1       127.0.0.1       10
      172.16.12.0    255.255.255.0      172.16.11.1    172.16.11.12       3
      172.16.21.0    255.255.255.0      172.16.11.1    172.16.11.12       3
      172.16.23.0    255.255.255.0      172.16.11.1    172.16.11.12       2
      172.16.32.0    255.255.255.0      172.16.11.1    172.16.11.12       2
      172.16.34.0    255.255.255.0      172.16.11.1    172.16.11.12       3
      172.16.43.0    255.255.255.0      172.16.11.1    172.16.11.12       3
   172.16.255.255  255.255.255.255     172.16.11.12    172.16.11.12       10
192.168.12.0    255.255.255.0      172.16.11.1    172.16.11.12       4
        224.0.0.0        240.0.0.0     172.16.11.12    172.16.11.12       10
  255.255.255.255  255.255.255.255     172.16.11.12    172.16.11.12       1
Passerelle par défaut :       172.16.11.1
===========================================================================
Itinéraires persistants : Aucun
C:\>
```

Faire d'une station un écouteur RIP permet de résoudre le problème d'une station connectée à plusieurs passerelles. La configuration IP de la station ne permet d'en déclarer qu'une seule. Le processus Ecouteur RIP mettra à profit les annonces des différentes passerelles pour compléter judicieusement la table de routage de la station. Ainsi, un paquet émis par la station et non destiné au réseau dont il est issu, est remis à la passerelle la plus appropriée. Mais l'administrateur peut également souhaiter que le routeur ne diffuse pas ses mises à jour sur ces réseaux d'extrémité. Une commande **passive interface** en configuration de routeur permet de réaliser cet objectif. La syntaxe de cette commande est la suivante :

```
passive-interface [default] interface-type interface-number
```

L'argument « *default* », optionnel, met l'ensemble des interfaces du routeur dans l'état passif. Cet argument est utile aux fournisseurs d'accès qui utilisent des routeurs de distribution pouvant comprendre plus de 200 interfaces, toutes ces interfaces devant être dans l'état passif. La commande **passive-interface** n'est pas particulière au protocole RIP, les autres protocoles de routage IP la proposent également (EIGRP, OSPF...).

Attention, la commande **passive-interface** ne concerne que la diffusion des messages RIP sur une interface. Les mises à jour reçues sur cette interface continuent d'être exploitées. Le réseau auquel l'interface est directement connectée continue d'être annoncé dans les messages diffusés sur les autres interfaces du routeur.

Appliquée au cas présent :

```
R110(config)#router rip
R110(config-router)#passive-interface f0/0
R110(config-router)#^Z
R110#
```

Une commande **debug ip rip** sur R110 ou une nouvelle capture Wireshark sur le lien F0/0 de R110 confirment que les mises à jour de R110 n'y sont plus diffusées.

3. Configuration avancée

a. Mise à jour unicast

Les mises à jour RIP sont diffusées. Cela ne nous dérange en rien sur les liaisons de type point à point. Mais nous sommes beaucoup plus réticents quand cette diffusion s'effectue sur un lien de type réseau local. Dans notre contexte de l'atelier 5, un lien LAN23 a été utilisé pour relier R110 et R120, ce qui nous fournit un prétexte pour explorer ce cas. La belle ouvrage consiste à désactiver les diffusions sur l'interface F0/1 de chacun des routeurs R110 et R120 puis à utiliser la commande **neighbor** afin de faire de R120 un voisin de R110 et de R110 un voisin de R120. Un routeur RIP émet ses mises à jour vers ses voisins connus en les encapsulant dans des datagrammes IP unicast.

R110

```
R110(config)#router rip
R110(config-router)#passive-interface F0/1
R110(config-router)#neighbor 172.16.23.12
R110(config-router)#^Z
```

R120

```
R120(config)#router rip
R120(config-router)#passive-interface F0/1
R120(config-router)#neighbor 172.16.23.11
R120(config-router)#^Z
R120#
```

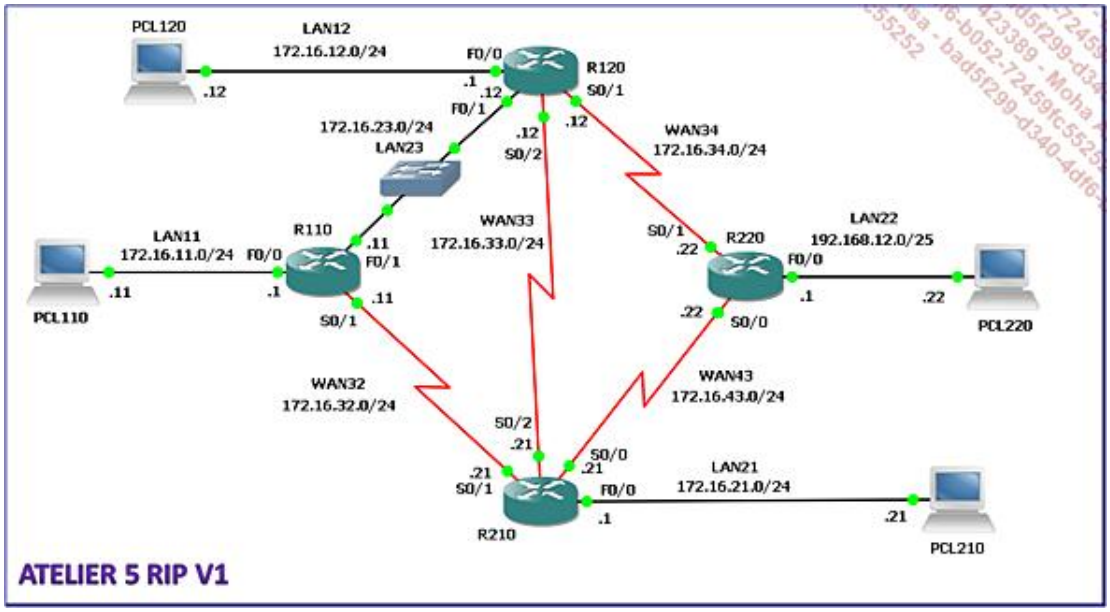
Il reste à s'assurer qu'effectivement les mises à jour sont transportées dans des datagrammes unicast. Une capture Wireshark aurait également pu convenir. Avec une commande **debug ip rip** :

```
R110#debug ip rip
RIP protocol debugging is on
00:02:26: RIP: received v1 update from 172.16.23.12 on FastEthernet0/1
00:02:26:      172.16.12.0 in 1 hops
00:02:26:      172.16.34.0 in 1 hops
00:02:26:      172.16.43.0 in 2 hops
00:02:26:      192.168.12.0 in 2 hops
00:02:38: RIP: sending v1 update to 255.255.255.255 via Serial0/1 (172.16.32.11)
00:02:38: RIP: build update entries
```

```
00:02:38:       subnet 172.16.11.0 metric 1
00:02:38:       subnet 172.16.12.0 metric 2
00:02:38:       subnet 172.16.23.0 metric 1
00:02:38:       subnet 172.16.34.0 metric 2
00:02:38: RIP: sending v1 update to 172.16.23.12 via FastEthernet0/1 (172.16.23.11)
00:02:38: RIP: build update entries
00:02:38:       subnet 172.16.11.0 metric 1
00:02:38:       subnet 172.16.21.0 metric 2
00:02:38:       subnet 172.16.32.0 metric 1
00:02:38:       subnet 172.16.43.0 metric 2
00:02:47: RIP: received v1 update from 172.16.32.21 on Serial0/1
00:02:47:      172.16.21.0 in 1 hops
00:02:47:      172.16.34.0 in 2 hops
00:02:47:      172.16.43.0 in 1 hops
00:02:47:      192.168.12.0 in 2 hops
R110#
```

b. Manipulation de la métrique de RIP

Scénario : la connectivité doit absolument être assurée entre LAN12 et LAN21. L'administrateur se propose d'ajouter un lien de secours entre R120 et R210, lien qui devra être pris en compte par RIP mais uniquement en cas de défaillance des routes préexistantes. Le contexte devient donc le suivant :



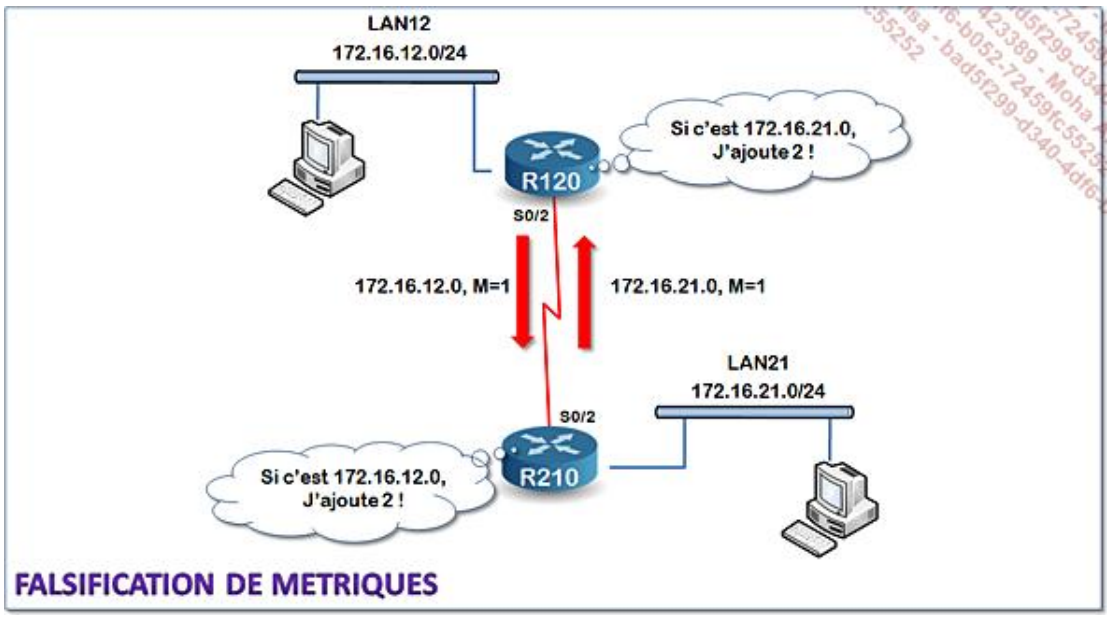
Le problème est que la rudimentaire métrique de RIP privilégiera le passage par la liaison WAN33 qui ne coûte que 1 saut contre 2 pour les alternatives par R110 et R220. La solution a été placée ici mais elle nécessite la mise en œuvre de listes d'accès. Deux solutions s'offrent au lecteur : il cesse de lire en séquence et aborde les listes d'accès ou se contente d'une lecture rapide de la solution puis la place en attente.

L'idée consiste à modifier la métrique de RIP et pour ce faire, l'IOS offre la commande **offset-list** en mode de configuration de routeur. Cette commande permet de modifier au choix une métrique annoncée (ajouter un offset à la valeur qu'aurait annoncée RIP puis annoncer la métrique falsifiée) ou une métrique reçue (ajouter un offset à la métrique reçue avant de dérouler l'algorithme RIP et par suite placer ou pas la route dans la table de routage). La commande peut être affectée à une interface ou à l'ensemble des interfaces du routeur. Sa syntaxe est la suivante :

```
offset-list {access-list-number | access-list-name} {in | out} offset [interface-
type interface-number]
```

Il faut avoir au préalable créé une liste d'accès afin de spécifier quelles routes doivent être prises en compte par la commande **offset-list**.

Dans le cas présent, sur R120, l'administrateur décide de provoquer l'ajout d'un offset égal à 2 sur les annonces issues de R210 via S0/2 quand elles concernent le réseau LAN21. Puis de façon symétrique, sur R210, provoquer l'ajout d'un offset égal à 2 sur les annonces issues de R120 via S0/2 quand elles concernent le réseau LAN12 :



Étape 1 : création d'une liste d'accès standard sur R120

- À l'aide d'une commande **access-list**. Le masque générique 0.0.0.0 spécifie que tous les bits de l'adresse spécifiée 172.16.21.0 doivent correspondre :

```
R120(config)#access-list 1 permit 172.16.21.0 0.0.0.0
```

Étape 2 : mise en œuvre de la commande offset-list sur R120

- En mode configuration de routeur :

```
R120(config)#router rip
R120(config-router)#offset-list 1 in 2 s0/2
R120(config-router)#^Z
R120#
```

Étape 3 : création d'une liste d'accès standard sur R210

- À l'aide d'une commande **access-list**. Le masque générique 0.0.0.0 spécifie que tous les bits de l'adresse spécifiée 172.16.12.0 doivent correspondre :

```
R210(config)#access-list 1 permit 172.16.12.0 0.0.0.0
```

Étape 4 : mise en œuvre de la commande offset-list sur R120

- En mode configuration de routeur :

```
R210(config)#router rip
R210(config-router)#offset-list 1 in 2 s0/2
R210(config-router)#exit
R210(config)#^Z
R210#
```

Étape 5 : recette de la solution

- Observez sur R120 les deux routes à coût égal par R110 et R220 vers LAN21. L'alternative par WAN33 n'est pas dans la table car son coût est supérieur (3).

```
R120#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
ia - IS-IS inter area, * - candidate default, U - per-user static route
.........
Gateway of last resort is 172.16.34.22 to network 0.0.0.0

     172.16.0.0/24 is subnetted, 8 subnets
R       172.16.43.0 [120/1] via 172.16.34.22, 00:00:17, Serial0/1
                    [120/1] via 172.16.33.21, 00:00:26, Serial0/2
R       172.16.32.0 [120/1] via 172.16.23.11, 00:00:01, FastEthernet0/1
                    [120/1] via 172.16.33.21, 00:00:26, Serial0/2
C       172.16.33.0 is directly connected, Serial0/2
C       172.16.34.0 is directly connected, Serial0/1
R       172.16.21.0 [120/2] via 172.16.23.11, 00:00:01, FastEthernet0/1
                    [120/2] via 172.16.34.22, 00:00:17, Serial0/1
C       172.16.23.0 is directly connected, FastEthernet0/1
C       172.16.12.0 is directly connected, FastEthernet0/0
R       172.16.11.0 [120/1] via 172.16.23.11, 00:00:01, FastEthernet0/1
R*   0.0.0.0/0 [120/1] via 172.16.34.22, 00:00:17, Serial0/1
R120#
```

- Observez sur R210 les deux routes à coût égal par R110 et R220 vers LAN12. L'alternative par WAN33 n'est pas dans la table car son coût est supérieur (3).

```
R210#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
ia - IS-IS inter area, * - candidate default, U - per-user static route
.........
Gateway of last resort is 172.16.43.22 to network 0.0.0.0

     172.16.0.0/24 is subnetted, 8 subnets
C       172.16.43.0 is directly connected, Serial0/0
C       172.16.32.0 is directly connected, Serial0/1
C       172.16.33.0 is directly connected, Serial0/2
R       172.16.34.0 [120/1] via 172.16.43.22, 00:00:05, Serial0/0
                    [120/1] via 172.16.33.12, 00:00:04, Serial0/2
C       172.16.21.0 is directly connected, FastEthernet0/0
R       172.16.23.0 [120/1] via 172.16.32.11, 00:00:13, Serial0/1
                    [120/1] via 172.16.33.12, 00:00:04, Serial0/2
R       172.16.12.0 [120/2] via 172.16.32.11, 00:00:13, Serial0/1
                    [120/2] via 172.16.43.22, 00:00:05, Serial0/0
R       172.16.11.0 [120/1] via 172.16.32.11, 00:00:13, Serial0/1
R*   0.0.0.0/0 [120/1] via 172.16.43.22, 00:00:05, Serial0/0
R210#
```

- L'administrateur veut vérifier que l'automatisme fonctionne. Pour ce faire, il fait passer les deux routeurs R110 et R220 à l'état down (sous GNS3, effectuez un clic droit sur chacun des routeurs puis **Suspendre**). La route de secours via WAN33 apparaît presque immédiatement sur R210 :

```
04:29:41: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1, changed state
to down
04:29:51: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state
to down
R210#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 6 subnets
R       172.16.32.0 [120/2] via 172.16.33.12, 00:00:23, Serial0/2
C       172.16.33.0 is directly connected, Serial0/2
C       172.16.21.0 is directly connected, FastEthernet0/0
R       172.16.23.0 [120/1] via 172.16.33.12, 00:00:23, Serial0/2
```

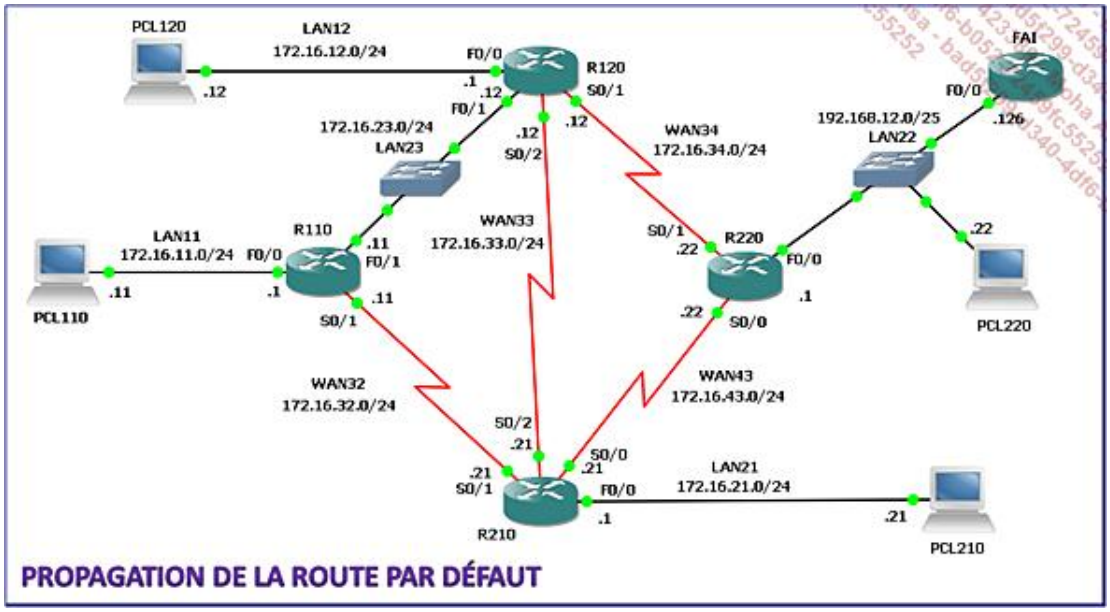
```
R       172.16.12.0 [120/3] via 172.16.33.12, 00:00:23, Serial0/2
R       172.16.11.0 [120/2] via 172.16.33.12, 00:00:23, Serial0/2
R210#
```

- Le temps de convergence est beaucoup plus long sur R120. Ceci s'explique par le fait que R120 est connecté à R110 par un lien LAN. Suspendre R110 n'est pas suffisant pour que le lien LAN passe à l'état down (le commutateur reste actif). Il faut donc attendre l'écoulement du temporisateur d'invalidation de la route vers LAN21 via 172.16.23.11 :

```
R120#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
Gateway of last resort is not set

     172.16.0.0/24 is subnetted, 4 subnets
C       172.16.33.0 is directly connected, Serial0/2
R       172.16.21.0 [120/3] via 172.16.33.21, 00:00:10, Serial0/2
C       172.16.23.0 is directly connected, FastEthernet0/1
C       172.16.12.0 is directly connected, FastEthernet0/0
R120#
```

c. Propagation de la route par défaut

Pour la circonstance, modifions un peu notre contexte :



Le scénario est le suivant : sur R220, nous installerons une route par défaut vers le routeur FAI puis nous assurerons que RIP propage effectivement cette route aux autres routeurs de la topologie.

Étape 1 : sur R220, désactiver la prise en compte par RIP du réseau 192.168.12.0

- À l'aide d'une commande **no network** :

```
R220(config)#router rip
R220(config-router)#no network 192.168.12.0
R220(config-router)#^Z
R220#
```

Étape 2 : sur R220, activer une route par défaut vers le réseau 192.168.12.0

Une route par défaut statique ne convient que si l'on souhaite établir une route par défaut sur le routeur local. Dans le cas présent, on souhaite voir se propager la route par défaut sur les autres routeurs et c'est la commande **ip default-network** déjà abordée dans le chapitre Le routage statique qui fournit une solution. La syntaxe de cette

commande est la suivante :

```
ip default-network network-number
```

- Mode de configuration globale.
- Si le routeur dispose d'une interface directement connectée au réseau spécifié, le protocole de routage dynamique activé sur le routeur annonce une route par défaut sur les autres interfaces. Dans le cas de RIP, la route par défaut est signifiée par une entrée dans le message de mise à jour RIP dont l'adresse IP est 0.0.0.0.

- Appliquée au cas présent :

```
R220(config)#ip default-network ?
A.B.C.D  IP address of default network

R220(config)#ip default-network 192.168.12.0
R220(config)#^Z
R220#
```

Étape 3 : sur R220, indiquer à RIP de propager la route par défaut

- À l'aide de la commande **default-information originate** en mode de configuration de routeur :

```
R220(config)#router rip
R220(config-router)#default-information ?
originate Distribute a default route

R220(config-router)#default-information originate
R220(config-router)#^Z
R220#
```

Étape 4 : sur FAI, activer une route statique vers le réseau 172.16.0.0

- À l'aide d'une commande **ip route** en mode de configuration globale :

```
FAI(config)#ip route 172.16.0.0 255.255.0.0 192.168.12.1
FAI(config)#^Z
FAI#
```

Étape 5 : recette de la solution

- Vérifions l'activité de RIP sur R220 afin de constater l'annonce de la route par défaut :

```
R220#debug ip rip
RIP protocol debugging is on
R220#
00:55:03: RIP: sending v1 update to 255.255.255.255 via Serial0/0 (172.16.43.22)
00:55:03: RIP: build update entries
00:55:03:       subnet 0.0.0.0 metric 1
00:55:03:       subnet 172.16.12.0 metric 2
00:55:03:       subnet 172.16.23.0 metric 2
00:55:03:       subnet 172.16.34.0 metric 1
00:55:03: RIP: sending v1 update to 255.255.255.255 via Serial0/1 (172.16.34.22)
00:55:03: RIP: build update entries
00:55:03:       subnet 0.0.0.0 metric 1
00:55:03:       subnet 172.16.21.0 metric 2
00:55:03:       subnet 172.16.32.0 metric 2
00:55:03:       subnet 172.16.43.0 metric 1
R220#
```

- Il reste à vérifier la bonne installation de la route par défaut sur les routeurs :

```
R220#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
ia - IS-IS inter area, * - candidate default, U - per-user static route
.........

Gateway of last resort is not set

*   192.168.12.0/25 is subnetted, 1 subnets
C*      192.168.12.0 is directly connected, FastEthernet0/0
172.16.0.0/24 is subnetted, 7 subnets
C       172.16.43.0 is directly connected, Serial0/0
R       172.16.32.0 [120/1] via 172.16.43.21, 00:00:12, Serial0/0
C       172.16.34.0 is directly connected, Serial0/1
R       172.16.21.0 [120/1] via 172.16.43.21, 00:00:12, Serial0/0
R       172.16.23.0 [120/1] via 172.16.34.12, 00:00:01, Serial0/1
R       172.16.12.0 [120/1] via 172.16.34.12, 00:00:01, Serial0/1
R       172.16.11.0 [120/2] via 172.16.43.21, 00:00:12, Serial0/0
[120/2] via 172.16.34.12, 00:00:01, Serial0/1
R220#
```

- Sur R110 :

```
R110#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
ia - IS-IS inter area, * - candidate default, U - per-user static route
.........
Gateway of last resort is 172.16.23.12 to network 0.0.0.0

     172.16.0.0/24 is subnetted, 7 subnets
R       172.16.43.0 [120/1] via 172.16.32.21, 00:00:07, Serial0/1
C       172.16.32.0 is directly connected, Serial0/1
R       172.16.34.0 [120/1] via 172.16.23.12, 00:00:00, FastEthernet0/1
R       172.16.21.0 [120/1] via 172.16.32.21, 00:00:07, Serial0/1
C       172.16.23.0 is directly connected, FastEthernet0/1
R       172.16.12.0 [120/1] via 172.16.23.12, 00:00:00, FastEthernet0/1
C       172.16.11.0 is directly connected, FastEthernet0/0
R*   0.0.0.0/0 [120/2] via 172.16.23.12, 00:00:00, FastEthernet0/1
               [120/2] via 172.16.32.21, 00:00:07, Serial0/1
R110#
```

4. Contrôle et dépannage

RIP et sa configuration vous ont semblé simples. La mise en service et le dépannage ne présentent pas plus de difficultés. Quand des erreurs surviennent, elles sont la plupart du temps dues au non-respect des contraintes imposées par le protocole :

À l'intérieur d'un réseau majeur, les préfixes affectés doivent être identiques sur l'ensemble du domaine RIP.

Un routeur qui annonce un réseau majeur doit être la route unique vers ce réseau. Ceci contraint à maintenir la contiguïté entre les sous-réseaux d'un même réseau majeur.

Donc, si des routes manquent ou si des routes sont incohérentes, réexaminez avec attention votre plan d'adressage et votre découpage en sous-réseaux.

5. Résumé

a. Les caractéristiques à retenir

- Les mises à jour de RIP sont totales, périodiques et diffusées.
- La métrique de RIP est le nombre de sauts. Un réseau directement connecté est annoncé avec un coût de 1. Un coût de 16 caractérise un réseau injoignable.
- En fixant l'infini à 16, RIP fixe également le nombre de sauts maximal et donc l'étendue du réseau à 15.
- Le partage d'horizon avec empoisonnement est destiné à prévenir la formation de boucles de routage. Il consiste à ne pas retransmettre (version simple) ou retransmettre avec un coût de 16 (version empoisonnement) sur une interface les routes apprises par cette interface.
- Les mises à jour déclenchées sont le dispositif prévu par le protocole pour accélérer la convergence.
- Le fonctionnement de RIP est étroitement lié à l'entretien de plusieurs temporisateurs :
    - Le temporisateur de mise à jour règle la période qui sépare deux émissions régulières de mises à jour.
    - RIP associe à chaque route une instance de temporisateur d'invalidation (180 secondes). Sans nouvelle de la route en question depuis le temps d'invalidation, la route est marquée injoignable dans la table de routage.
    - RIP associe à chaque route une instance de temporisateur d'effacement (240 secondes). Sans nouvelle de la route depuis ce temps, la route est supprimée de la table de routage. La différence entre le temporisateur d'invalidation et le temporisateur d'effacement (plus grand) s'explique par la volonté d'annoncer la route comme étant injoignable pendant 60 secondes ce qui accélère la convergence.
    - L'implémentation CISCO ajoute à ces trois temporisateurs du standard, le temporisateur de retenue. Chaque fois qu'un routeur reçoit une mise à jour dans laquelle le coût d'une route déjà connue augmente, le routeur arme ce temporisateur et refuse toute mise à jour pour cette route tant que le temporisateur n'a pas expiré.
    - Les annonces de routes ne comportent pas l'information de masque ce qui aboutit à un comportement dit avec classe.

b. Les commandes à retenir

Commande	Mode	Description
router rip	Configuration globale	Active le processus RIP.
network *@IP_réseau_majeur*	Configuration de routeur	Indique au processus RIP que ce réseau majeur doit être pris en compte.
neighbor *@IP_voisin*	Configuration de routeur	Active l'envoi de mises à jour unicast vers ce voisin.
passive-interface *interface-typeinterface-number*	Configuration de routeur	Cesse l'envoi de mises à jour sur l'interface spécifiée.
show ip route rip	Mode utilisateur	Affiche l'ensemble des routes issues de RIP.
show ip protocols	Mode utilisateur	Affiche les paramètres et l'état actuel du protocole de routage activé sur le routeur.

debug ip rip	Mode privilégié	Affiche en temps réel le trafic d'acheminement RIP.
offset-list{*access-list-number* | *access-list-name*} {**in** | **out**} *offset* [*interface-type interface-number*]	Configuration globale	Ajoute ou retranche un offset à la métrique recue ou annoncée d'une route conforme aux critères d'une liste d'accès.
timers basic *update invalid holddown flush*	Configuration de routeur	Ajuste les valeurs initiales des temporisateurs.
output-delay *delay*	Configuration de routeur	Introduit un espace de temps minimal de 8 à 50 millisecondes entre deux émissions de mises à jour RIP, utile pour qu'un routeur lent puisse s'accommoder des mises à jour issues d'un routeur puissant.

6. Atelier : Mise en œuvre d'une configuration RIP

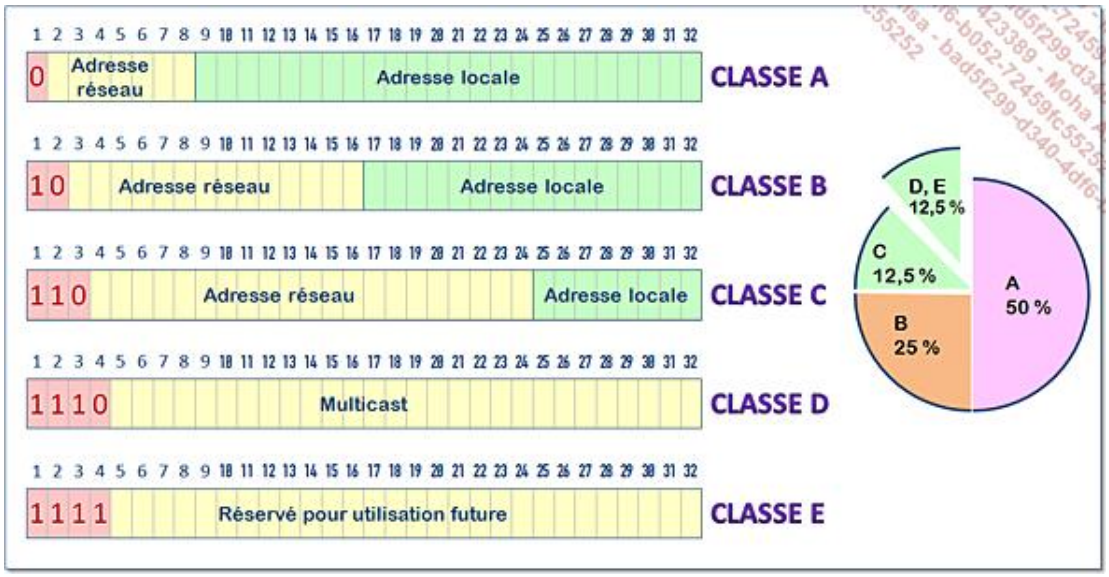
Vous voilà promu administrateur du réseau objet de la mise en situation précédente. Vous remarquez que le lien de secours WAN33 est une ressource dormante et souhaitez établir un partage de charge à coût égal sur trois routes entre LAN12 et LAN21. Les trois routes sont constituées des deux routes existantes auxquelles devrait s'additionner la route par WAN33.

Modifiez la configuration de R120 et de R210 en conséquence, sans remettre en cause le choix du protocole RIP.

Une solution est proposée au chapitre Ateliers et exercices corrigés. De plus, le lecteur qui voudrait reproduire cette topologie dans un contexte émulé GNS trouvera une aide précieuse au chapitre Annexes - Définition d"un contexte d'atelier.

Adressage avec classe, rappel

En septembre 1981, est publié le RFC790 qui entérine la création de trois classes d'adresses A, B et C. Le procédé retenu tente de satisfaire tout le monde puisqu'il permet la coexistence de découpages (*splits*) différents. La création des classes D et E n'est intervenue que plus tard. Ce système de classes a perduré jusqu'à la fin des années 1990. Pourquoi étudier un système obsolète aujourd'hui ? Parce que son importance fut telle sur les développements d'Internet qu'inévitablement, on rencontre de façon régulière des références aux classes. Les nouvelles versions du cursus CCNA nomment l'adressage avec classes : « adressage hérité ».



1. Adresses de classe A

126 réseaux possibles :

- Première adresse de classe A : **1.0.0.1**
- Dernière adresse de classe A : **126.255.255.254**
- Masque naturel : **255.0.0.0**
- Les adresses réseau « 0 » et « 127 » sont réservées.

Identificateurs hôtes particuliers :

- Une adresse **0.0.0** ne constitue pas un identificateur de station valide.
- **255.255.255** est utilisé pour le broadcast (diffusion) sur toutes les machines du réseau « @ Réseau ».

En final, un réseau de classe A peut comprendre 2^{24} - 2 = 16 777 214 adresses.

Les adresses de classe A sont définies sur 31 bits (32 bits dont 1 bit imposé) et consomment donc la moitié de l'espace d'adressage IP (environ 2 milliards d'adresses). C'était un paradoxe du mécanisme adopté alors, les adresses réseau de classe A étant les plus difficiles à attribuer (il n'y en a que 126).

2. Adresses de classe B

16 384 réseaux possibles :

- Première adresse de classe B : **128.0.0.1**

- Dernière adresse de classe B : **191.255.255.254**
- Masque naturel : **255.255.0.0**

Les réseaux de classe B étaient évidemment attribués « au compte-goutte », il fallait justifier d'un parc d'au moins 10000 machines. On peut citer l'entreprise EDF qui a obtenu 18 adresses réseau de classe B, ce qui représente 1179612 adresses possibles en théorie.

Identificateurs hôtes particuliers :

- Une adresse **0.0** ne constitue pas un identificateur de station valide.
- **255.255** est utilisé pour le broadcast (diffusion) sur toutes les machines du réseau « @ Réseau ».

En final, un réseau de classe B peut comprendre 2^{16} - 2 = 65 534 adresses.

Les adresses de classe B sont définies sur 30 bits (32 bits dont 2 imposés) et consomment donc la moitié de l'espace non déjà consommé par les adresses de classe A, soit environ 1 milliard d'adresses.

3. Adresses de classe C

2 097 152 réseaux possibles :

- Première adresse de classe C : **192.0.0.1**
- Dernière adresse de classe C : **223.255.255.254**
- Masque par défaut : **255.255.255.0**

Identificateurs hôtes particuliers :

- Une adresse **0** ne constitue pas un identificateur de station valide.
- **255** est utilisé pour le broadcast (diffusion) sur toutes les machines du réseau « @ Réseau ».

En final, un réseau de classe C peut comprendre 2^{24} - 2 = 254 adresses.

Les adresses de classe C sont définies sur 29 bits (32 bits dont 3 imposés) et consomment donc la moitié de l'espace non déjà consommé par les adresses de classe A et de classe B, soit environ 500 millions d'adresses. Ainsi, les adresses les plus faciles à attribuer étaient hélas les moins nombreuses.

4. Adresses de classe D et E

Les adresses de classe D sont utilisées pour identifier des groupes de machines et permettre des communications multicast (traduisible par multi diffusion). Un paquet multicast est un paquet destiné à **plusieurs** machines (le broadcast lui, est destiné à toutes les machines d'un réseau). Les adresses de classe D commençant par la séquence de bits 1110, le premier octet d'une adresse de classe D est compris entre 224 et 239.

Une adresse de classe D est toujours une adresse de destination

---

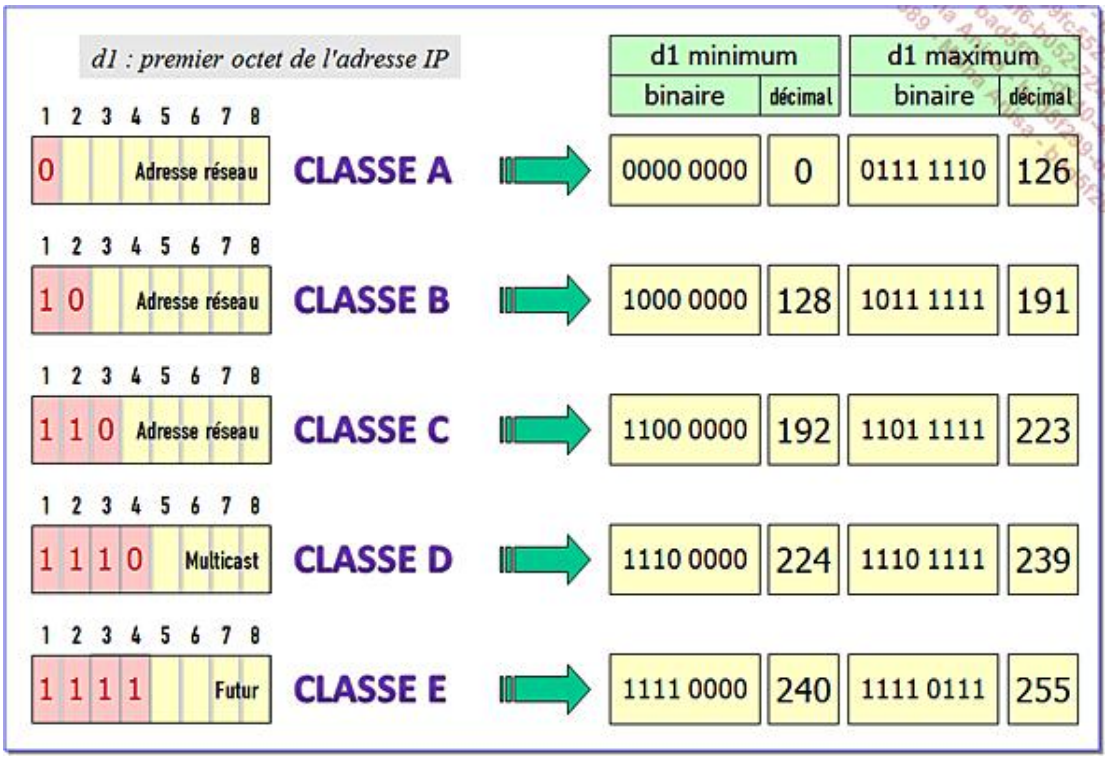
Exemple :

OSPF (*Open Shortest Path First*) est l'un des protocoles de routage étudiés dans cet ouvrage. Dans un réseau mettant en œuvre des routeurs, il est possible de joindre l'ensemble des routeurs OSPF en envoyant un paquet à l'adresse de destination **224.0.0.5**.

Toujours en approximant, sur les 4 milliards d'adresses IP possibles, les classes A, B et C représentent 3.5 milliards, les classes D et E représentent chacune 250 millions.

5. Calculer une classe d'adresses

L'administrateur réseau devrait retenir les valeurs frontières du premier octet de façon à pouvoir identifier immédiatement à quelle classe appartient une adresse IP et par suite déduire quel est le masque naturel correspondant :



6. Classes d'adresses et RFC

Les numéros assignés utilisés par la pile de protocoles TCP/IP ont fait l'objet de publications régulières à partir du RFC349 datant de mai 1972 jusqu'à l'ultime RFC1700 datant d'octobre 1994. Parmi ces RFC, le RFC790, publié en septembre 1981, définit les classes A, B et C. Les adresses au-delà des classes A, B et C, c'est-à-dire à partir de la valeur 224 (1er octet), sont simplement réservées. Le RFC1700 mentionne les 5 classes A à E.

À partir de 1994, les mises à jour trop fréquentes des numéros d'Internet obligent l'IANA à créer une base de données en ligne qui permettra la publication des numéros à jour en temps réel, base consultable sur www.iana.org. Le RFC3232 entérine ce fonctionnement en janvier 2002, rendant obsolète le RFC1700.

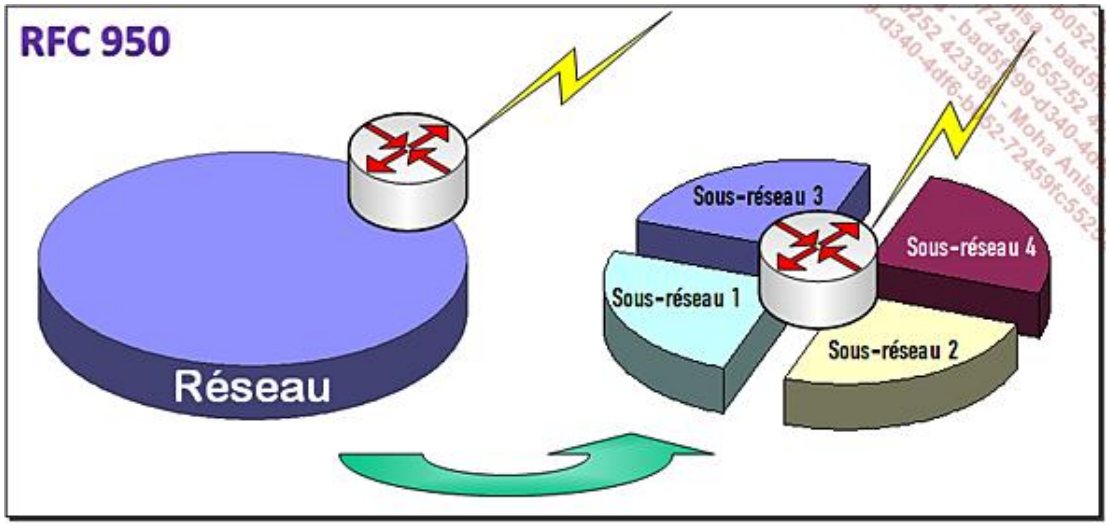
Structuration par sous-réseaux

RFC utiles :

- RFC950 - Internet standard Subnetting Procedure - Août 1985

Internet s'est trouvé un temps menacé par la pénurie d'adresses. Le réseau né dans les années 1980 était alors confidentiel, l'espace d'adressage paraissait plus que suffisant et on a sans doute à l'époque distribué les blocs d'adresses avec un peu de légèreté. Le système de classes contribuait au gaspillage, les besoins des entreprises étant souvent supérieurs à ceux pouvant être satisfaits par la classe C sans toutefois justifier l'attribution d'adresses de classe B. L'adressage par classes est devenu tout à fait inadapté mais il fallait pourtant tenter de préserver les attributions déjà réalisées.

Une partie de solution a été adoptée en 1985 en proposant la possibilité de structurer l'espace d'adressage d'un réseau de classe A, B ou C :



L'idée consiste à « emprunter » un nombre de bits à définir dans l'adresse hôte afin d'en faire une adresse de sous-réseau :

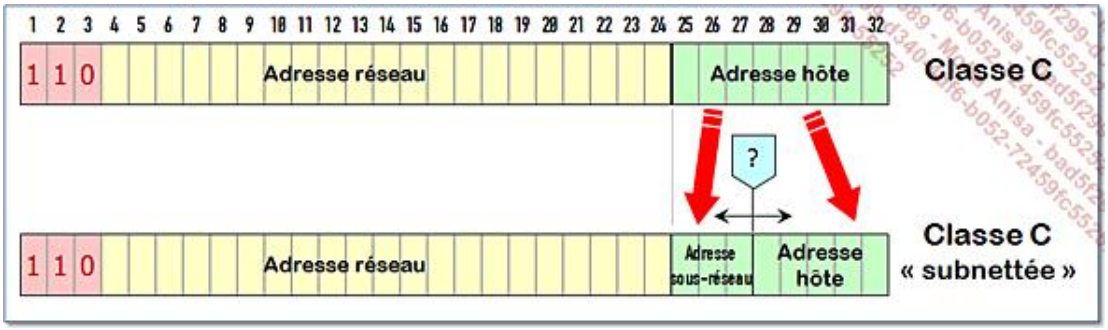
- 1 bit emprunté permet de définir deux sous-réseaux et donc de diviser l'espace de départ en deux parties égales.
- 2 bits empruntés permettent de définir 4 sous-réseaux (cas de l'illustration).
- 3 bits empruntés permettent de définir 8 sous-réseaux et ainsi de suite.

Vu de l'extérieur, l'adresse du réseau est toujours valide, un datagramme destiné à l'une des machines de l'un quelconque des sous-réseaux est toujours transporté par l'Internet en mettant à profit l'adresse réseau, il n'y a donc pas d'impact sur l'Internet mondial.

À l'interne par contre, il devient possible pour l'entreprise de mieux structurer son espace d'adressage et cela peut contribuer à éviter des demandes de blocs d'adresses supplémentaires dont l'objet ne serait pas de pourvoir un besoin d'adresses mais bien de permettre cette structuration.

C'est à l'administrateur qu'il revient de fixer la frontière entre l'adresse sous-réseau et l'adresse hôte selon les besoins de l'entreprise, chaque bit emprunté supplémentaire multiplie par 2 le nombre de sous-réseaux et divise par deux le nombre potentiel de machines à l'intérieur d'un sous-réseau. Les critères sont au choix, soit le nombre de machines qu'il lui faut atteindre dans chaque sous-réseau, soit le nombre de sous-réseaux qu'il lui faut constituer.

1. Découper une adresse de classe C



L'ouvrage Notion de base sur les réseaux avait proposé une méthode binaire pour réaliser le découpage en sous-réseaux. Faisons un peu différent cette fois en proposant une méthode de découpage rapide.

Nous disposons des 8 bits de poids faible de l'adresse IP, le tableau suivant inventorie les masques théoriques possibles :

	B8	B7	B6	B5	B4	B3	B2	B1	Masque résultant (4e octet)	Nombre d'espaces résultants	Taille de chaque espace	Adresses IP potentielles
Poids	128	64	32	16	8	4	2	1				
Exclu	1	0	0	0	0	0	0	0	128	2	128	126
Permis	1	1	0	0	0	0	0	0	192	4	64	62
Permis	1	1	1	0	0	0	0	0	224	8	32	30
Permis	1	1	1	1	0	0	0	0	240	16	16	14
Permis	1	1	1	1	1	0	0	0	248	32	8	6
Permis	1	1	1	1	1	1	0	0	252	64	4	2
Exclu	1	1	1	1	1	1	1	0	254	128	2	0

Le RFC interdit une adresse de sous-réseau dont tous les bits sont à 0 car il serait impossible de distinguer l'adresse de sous-réseau de l'adresse réseau. Exemple : l'administrateur souhaite structurer l'adresse de classe C 192.168.1.0 avec le masque 255.255.255.192. Les quatre adresses de sous-réseau potentielles sont 192.168.1.0, 192.168.1.64, 192.168.1.128 et 192.168.1.192. Mais la première de ces quatre adresses ne se distingue pas de l'adresse réseau originale.

De la même façon, le RFC interdit une adresse de sous-réseau dont tous les bits sont à 1 car cette fois, c'est l'adresse de diffusion du sous-réseau qu'il est impossible de distinguer de l'adresse de diffusion du réseau. En conservant le même exemple, l'adresse de diffusion du sous-réseau 192.168.1.192 est 192.168.1.255, soit également l'adresse de diffusion du réseau.

Il est possible de résumer de façon simple ces deux restrictions : lors d'un découpage en sous-réseaux, le premier et le dernier sous-réseau résultants du découpage sont interdits. Autre conséquence, le masque de sous-réseau 128 est de fait impossible à utiliser puisqu'il ne crée que deux sous-réseaux. Nous verrons un peu plus tard que ces deux restrictions n'ont plus cours à l'heure du découpage VLSM.

Dernier point mais non le moindre : dans l'espace créé par un sous-réseau, l'adresse hôte 0 (tous les bits de l'adresse hôte à zéro) est utilisée pour désigner le sous-réseau. L'adresse hôte dont tous les bits sont à 1 est l'adresse de diffusion du sous-réseau. La conséquence est que dans un espace de x adresses, les adresses utiles, c'est-à-dire celles qui seront affectées à des hôtes, ne sont que x-2. C'est la raison pour laquelle le masque 254 est exclu.

Traitons deux exemples pour se convaincre de la facilité de la chose :

Exemple 1 : 255.255.255.192 (/26)

Il s'agit de découper l'adresse de classe C 192.168.1.0 avec le masque de sous-réseau 255.255.255.192. Astreignons-nous à fixer le cadre en répondant systématiquement à ces quelques questions :

- Combien de sous-réseaux possibles ? Le masque de sous-réseau comporte deux bits, la réponse est 2^2 -

2 = 2.

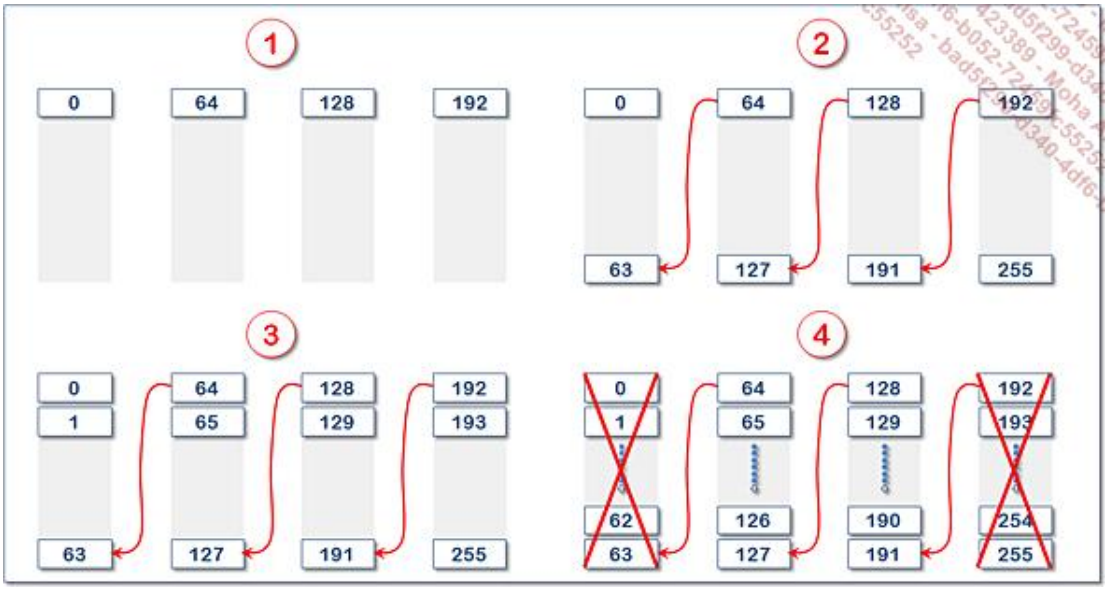
- Combien d'hôtes par sous-réseau ? L'adresse hôte comporte 6 bits, la réponse est 2^6 - 2 = 62.

- Quels sont les sous-réseaux valides ? C'est ici qu'il est possible de gagner du temps. En remarquant que la taille d'un espace résultant du découpage est donnée par 256 - *masque* = 256 - 192 = 64. Il se trouve que c'est aussi le pas d'incrémentation dans les sous-réseaux. Les espaces résultants sont 0, 64, 128 et 192 mais, pour les raisons précédemment évoquées, les seuls sous-réseaux valides sont 64 et 128.

- Quelles sont les adresses de diffusion de chaque sous-réseau ? Pour chaque sous-réseau, c'est la dernière adresse de l'espace considéré, c'est-à-dire l'adresse qui précède immédiatement l'adresse de sous-réseau suivant. Exemple : le sous-réseau 128 suit le sous-réseau 64. L'adresse de diffusion du réseau 64 est donc 127 qui précède immédiatement 128. De la même façon, l'adresse de diffusion du sous-réseau 128 est 191.

- Quelles sont les adresses hôtes valides ? Pour chaque sous-réseau, ce sont les adresses comprises entre l'adresse du sous-réseau et l'adresse de diffusion.

Un tableau devrait encore clarifier la méthode :



Exemple 2 : 255.255.255.240 (/28)

Il s'agit de découper l'adresse de classe C 192.168.1.0 avec le masque de sous-réseau 255.255.255.240. Astreignons-nous à fixer le cadre en répondant systématiquement à ces quelques questions :

- Combien de sous-réseaux possibles ? Le masque de sous-réseau comporte quatre bits, la réponse est 2^4 - 2 = 14.

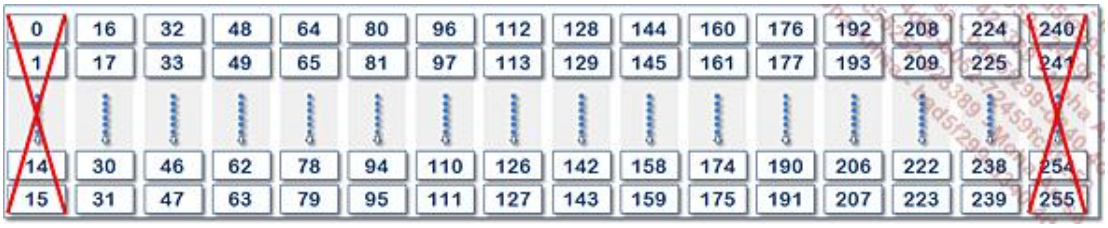
- Combien d'hôtes par sous-réseau ? L'adresse hôte comporte 4 bits, la réponse est 2^4 - 2 = 14.

- Quels sont les sous-réseaux valides ? La taille d'un espace résultant du découpage est donnée par 256 - *masque* = 256 - 240 = 16. Les espaces résultants sont 0, 16,32, 48, 64, 80, 96, 112, 128, 144, 160, 176, 192, 208, 224 et 240, pour les raisons précédemment évoquées, il faut exclure les sous-réseaux 0 et 240.

- Quelles sont les adresses de diffusion de chaque sous-réseau ? Pour chaque sous-réseau, c'est la dernière adresse de l'espace considéré, c'est-à-dire l'adresse qui précède immédiatement l'adresse de sous-réseau suivant. Exemple : le sous-réseau 128 suit le sous-réseau 112. L'adresse de diffusion du réseau 112 est donc 127 qui précède immédiatement 128. De la même façon, l'adresse de diffusion du sous-réseau 128 est 143.

- Quelles sont les adresses hôtes valides ? Pour chaque sous-réseau, ce sont les adresses comprises entre l'adresse du sous-réseau et l'adresse de diffusion.

Le tableau suivant synthétise les résultats :



a. La commande ip subnet-zero

La commande **ip subnet-zero** en mode de configuration globale permet de passer outre la règle qui interdisait les adresses de sous-réseau exclusivement composées de 0. Et cette commande est entrée par défaut depuis la version 12 de l'IOS. Puisque l'IOS tolère également que l'administrateur utilise une adresse de sous-réseau exclusivement composée de 1, on regagne ainsi les deux sous-réseaux placés aux deux extrémités de l'espace découpé. Autre conséquence, le masque 255.255.255.128 est à nouveau valide.

b. Le « sub-netting » mental

Il est admis aujourd'hui que conserver une activité cognitive simple pourrait diviser par 2 le risque de maladie d'Alzheimer. Apportons notre pierre à l'édifice en tentant d'exprimer mentalement à quel sous-réseau appartient une adresse IP quelconque.

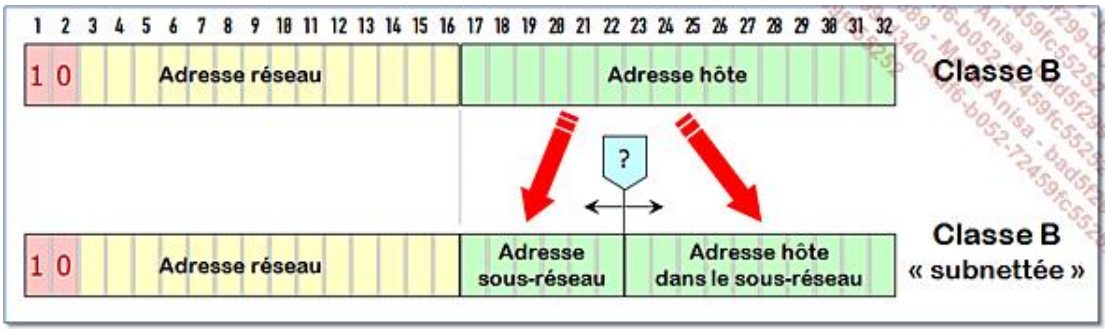
Exemple 1 : 192.168.1.71, masque 255.255.255.224

La taille d'un espace résultant du découpage est 256 - *masque* = 256 - 224 = 32. Les espaces résultants sont 0, 32, 64, 96, 128, 160, 192 et 224. L'adresse 71 est comprise entre 64 et 96. On déduit que le sous-réseau est 192.168.1.64.

Exemple 2 : 192.168.1.45, masque 255.255.255.248

La taille d'un espace résultant du découpage est 256 - *masque* = 256 - 248 = 8. Incrémentons les espaces résultants jusqu'à encadrer l'adresse IP : 0, 8, 16, 24, 32, 40, 48... L'adresse 45 est comprise entre 40 et 48. On déduit que le sous-réseau est 192.168.1.40.

2. Découper une adresse de classe B



En disposant des 16 bits de poids faible de l'adresse IP, les masques possibles sont évidemment beaucoup plus nombreux :

255.255.128.0	/17	255.255.255.0	/24
255.255.192.0	/18	255.255.255.128	/25
255.255.224.0	/19	255.255.255.192	/26
255.255.240.0	/20	255.255.255.224	/27
255.255.248.0	/21	255.255.255.240	/28

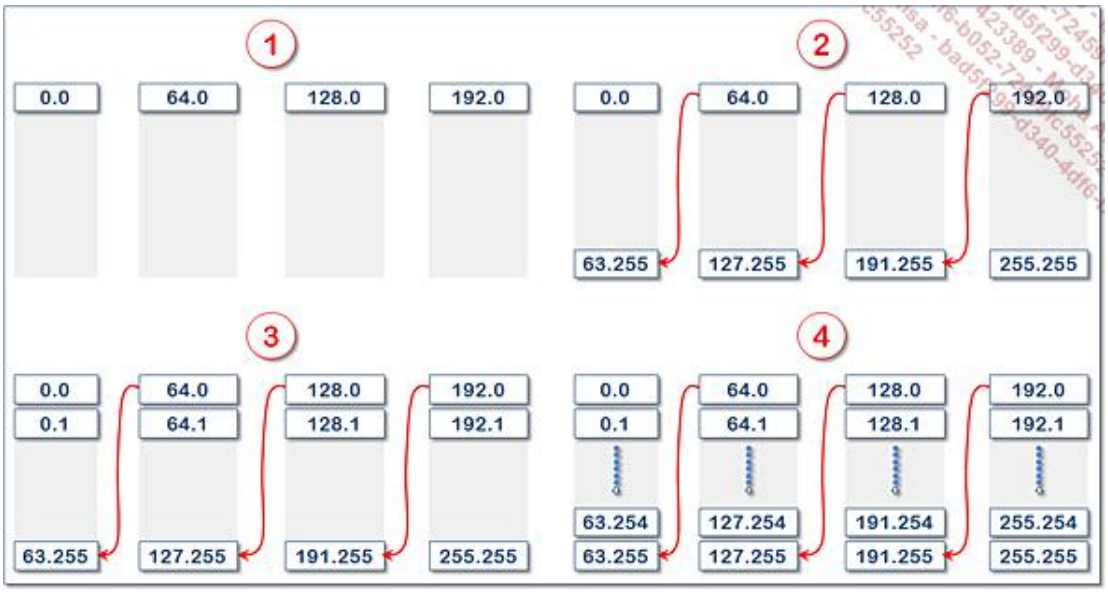
255.255.252.0	/22	255.255.255.248	/29
255.255.254.0	/23	255.255.255.252	/30

Exemple 1 : 255.255.192.0 (/18)

Il s'agit de découper l'adresse de classe B 172.16.0.0 avec le masque de sous-réseau 255.255.192.0.

- Combien de sous-réseaux possibles ? Le masque de sous-réseau comporte deux bits, la réponse est 2^2 - 2 = 2.
- Combien d'hôtes par sous-réseau ? L'adresse hôte comporte 14 bits, la réponse est 2^{14} - 2 = 16 382.
- Quels sont les sous-réseaux valides ? La taille d'un espace résultant du découpage est donnée par 256 - *masque* = 256 - 192 = 64. Attention, on parle ici du 3e octet. Il se trouve que c'est aussi le pas d'incrémentation dans les sous-réseaux. Les espaces résultants sont 0, 64, 128 et 192 dont on écartera ou pas les sous-réseaux 0 et 192.
- Quelles sont les adresses de diffusion de chaque sous-réseau ?
- Quelles sont les adresses hôtes valides ?

Un tableau devrait clarifier la méthode :



Par rapport au découpage d'un réseau de classe C, il a fallu ajouter le 4e octet, 0 quand il s'agissait d'exprimer l'adresse réseau, 255 quand il s'agissait d'exprimer l'adresse de diffusion.

Exemple 2 : 255.255.255.128 (/25)

Il s'agit de découper l'adresse de classe B 172.16.0.0 avec le masque de sous-réseau 255.255.255.128.

- Combien de sous-réseaux possibles ? Le masque de sous-réseau comporte neuf bits, la réponse est 2^9 - 2 = 510.
- Combien d'hôtes par sous-réseau ? L'adresse hôte comporte 7 bits, la réponse est 2^7 - 2 = 126.
- Quels sont les sous-réseaux valides ? La taille d'un espace résultant du découpage est donnée par 256 - *masque* = 256 - 255 = 1 dans le 3e octet et par 256 - *masque* = 256 - 128 = 128 dans le 4e octet. Le pas d'incrémentation résultant est 0.128. Les espaces résultants sont 0.0, 0.128, 1.0, 1.128, 2.0, 2.128… dont on écartera ou pas les sous-réseaux 0.0 et 255.128.

- Quelles sont les adresses de diffusion de chaque sous-réseau ?
- Quelles sont les adresses hôtes valides ?

Le tableau suivant synthétise une partie des résultats :

L’adressage sans classe

1. Masque de longueur variable VLSM

RFC utiles :

- RFC 1812 - Requirements for IP Version 4 Routers - Juin 1995

Dans le découpage d’un réseau avec classe, nous avons observé qu’il était possible de fragmenter le réseau initial de la façon la plus grossière à la façon la plus fine. Hélas, la volonté d’épouser au mieux les besoins de structuration d’une organisation se heurte à la contrainte imposée par le masque fixe.

Dans le même temps, il faut se souvenir que, vu de l’extérieur, la route qui mène vers cet ensemble de sous-réseaux n’était pas modifiée par l’opération de découpage. La route devient une « super-route » vers un ensemble de réseaux ! On comprend l’intérêt de cette route agrégée sur le volume des tables de routage.

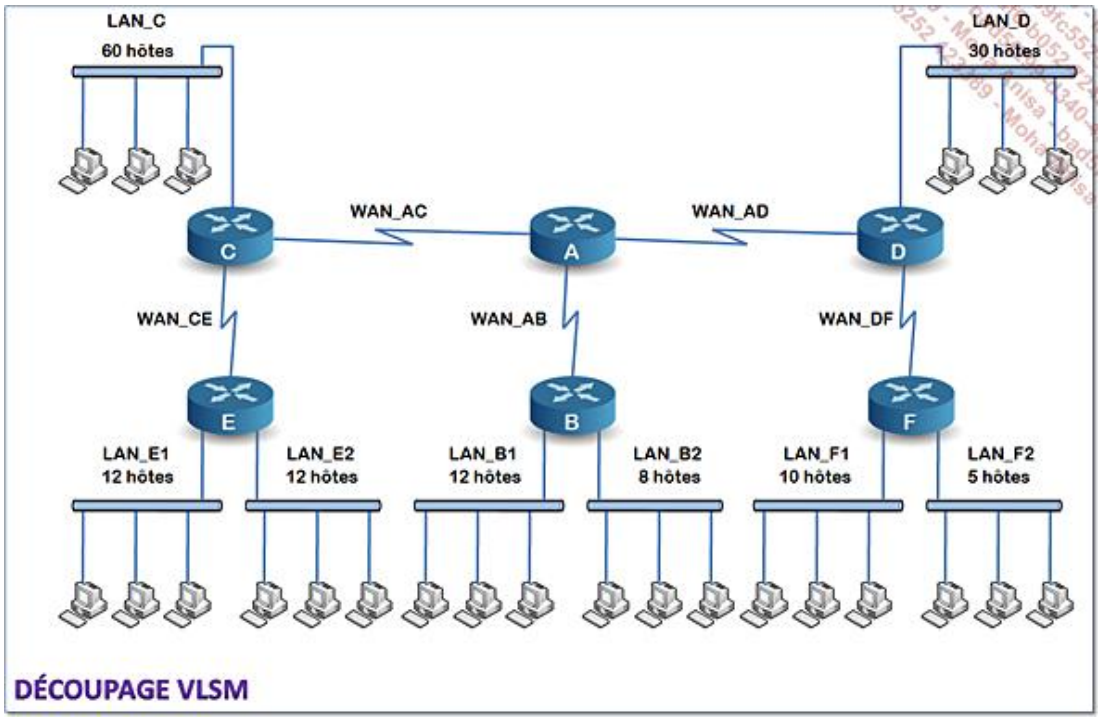
En adoptant, VLSM (*Variable Length Subnet Mask*), la structuration en sous-réseaux prend une autre dimension. Il s’agit toujours d’un découpage en sous-réseaux mais avec un masque de longueur variable. La longueur de préfixe adoptée doit se maintenir entre la longueur initiale + 1 et la longueur maximale soit /30. Les avantages sont au nombre de deux :

- Il devient possible de créer des sous-réseaux dont le nombre d’adresses hôtes « colle » au plus près du besoin d’adresses du sous-réseau considéré. Le cas le plus évident est celui des liens point à point entre routeurs qui nécessitent deux adresses et à qui l’administrateur pourra affecter un sous-réseau /30.

- En structurant le réseau de façon judicieuse, l’agrégation d’adresses est favorisée avec des avantages déterminants sur le volume des tables de routage, la consommation de ressources machine (temps passé à lire la table de routage, encombrement de la table de routage en mémoire vive) ainsi que sur la bande passante consommée par le trafic d’acheminement. Il faut entendre par trafic d’acheminement le contraire du trafic utile. Le trafic d’acheminement ne transporte pas les données utilisateur mais des informations de topologie échangées entre processus de routage.

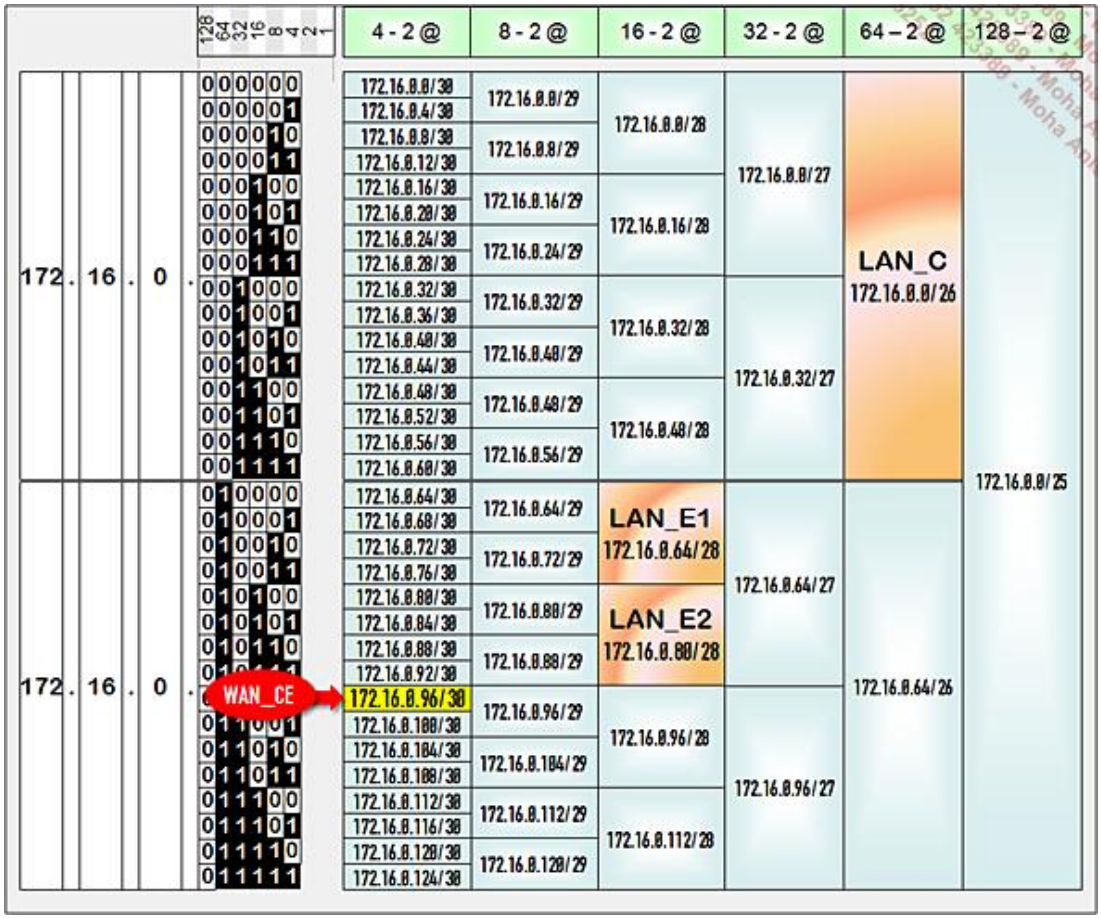
Étudions un cas concret afin de mesurer la portée d’une telle décision. L’administrateur de l’entreprise Primrose a négocié un préfixe suffisant pour attribuer des adresses à un parc d’environ 150 machines et a obtenu le préfixe 172.16.0.0/24. Cela signifie qu’il dispose de 8 bits pour l’espace d’adressage de son entreprise. 8 bits représente 2^8 - 2 = 254 adresses potentielles avant structuration.

L’entreprise Primrose avant structuration de l’espace d’adressage :



L’administrateur place dans un tableau tous les sous-réseaux possibles du préfixe 172.16.0.0/24 avec les différents

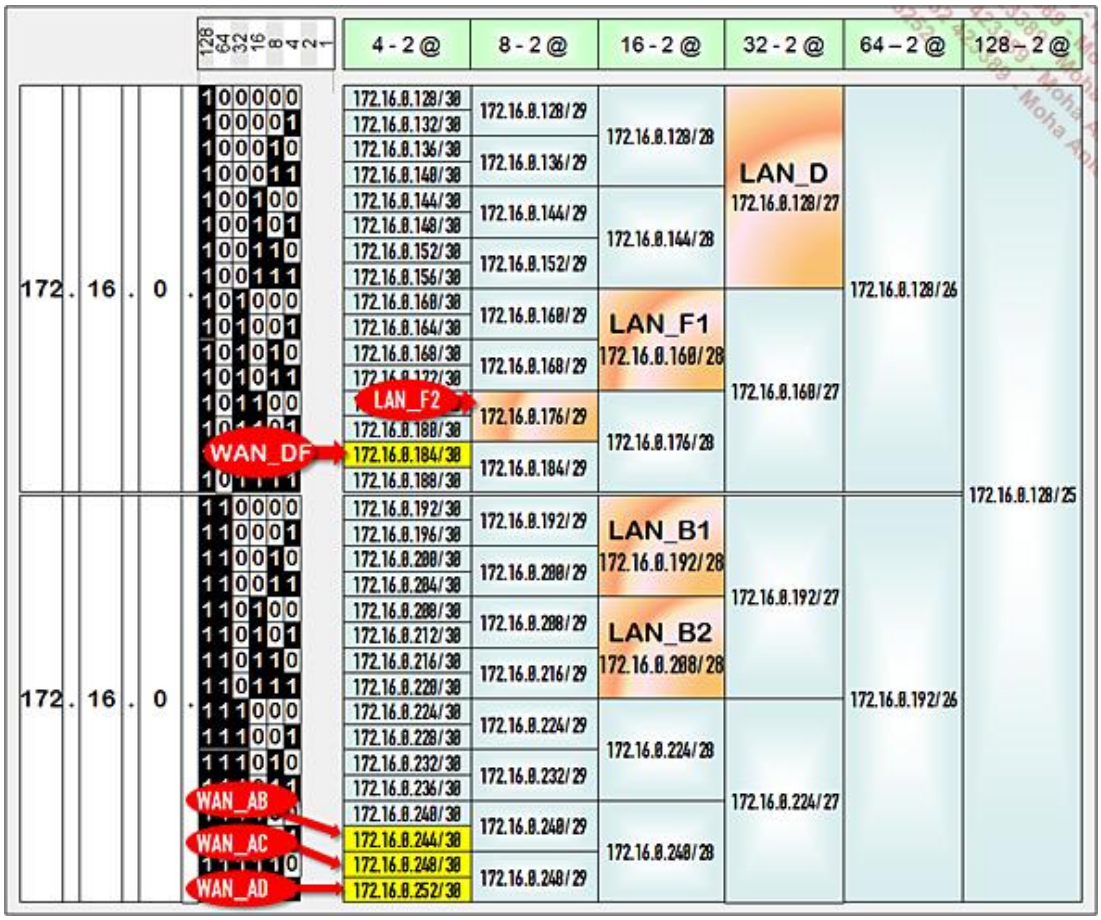
masques possibles de /25 à /30. Bien sûr, un réseau de grande taille ne pourrait pas être représenté par un seul de ces tableaux. Un peu à la façon des cartes routières qui existent à différentes échelles, l'administrateur qui aurait à gérer de grands espaces d'adresses devrait sans doute construire un tableau général puis des tableaux intermédiaires qui détailleraient des parties du préfixe initial. Dans notre cas, voici une suggestion de présentation de ce tableau, présentation que nous avons déjà utilisée à plusieurs reprises dans l'ouvrage précédent ou dans le chapitre Le routage statique de cet ouvrage. Pour mémoire, nous avions nommé tableau VLSM dichotomique ce tableau car les espaces d'adresses d'une colonne (une longueur de préfixe) sont obtenus en divisant par deux les espaces de la colonne immédiatement à droite (longueur de préfixe immédiatement inférieure).



L'administrateur observe que le routeur A est relié au reste du réseau par une structure en étoile à trois branches, Est, Ouest et Sud.

- D'un point de vue agrégation, il serait bon que toutes les routes vers les différents sous-réseaux contenus dans la zone Ouest par exemple puissent être agrégées en une seule route :
    - Le plus grand besoin à satisfaire est celui du réseau LAN_C, l'administrateur lui affecte le préfixe 172.16.0.0/26.
    - Les deux réseaux LAN_E1 et LAN_E2 se voient affecter les préfixes 172.16.0.64/28 et 172.16.0.80/28. Ainsi, il est possible d'annoncer une seule route agrégée 172.16.0.64/27 vers LAN_E1 et LAN_E2 via le routeur E.
    - L'administrateur affecte au lien « serial » WAN_CE, le préfixe 172.16.0.96/30.
- Toute la zone Ouest peut être annoncée comme une seule route 172.16.0.0/25 via le routeur C, à la condition de ne pas avoir à attribuer les espaces restants ailleurs dans le réseau. L'administrateur doit privilégier une assignation topologique car c'est elle qui rend possible l'agrégation. Mais il peut être contraint d'abandonner partiellement cet objectif s'il manque d'adresses.

Seconde partie du tableau :



- L'administrateur tente d'appliquer les mêmes principes sur la partie Est :
  - Les besoins du réseau LAN_D sont couverts par le préfixe 172.16.0.128/27.
  - LAN_F1 se voit affecter 172.16.0.160/28.
  - LAN_F2 se voit affecter 172.16.0.176/29. La situation est un peu moins favorable que pour la partie Ouest car pour agréger LAN_F1 et LAN_F2, il faudrait puiser ailleurs les adresses nécessaires au lien « serial » WAN_DF. L'administrateur renonce à cette possibilité.
  - En revanche, toute la zone Est peut être annoncée comme une seule route 172.16.0.128/26 via le routeur D, à la condition que le préfixe encore disponible 172.16.0.188/30 reste dans la partie Est.
- Il reste à affecter des adresses à la partie Sud :
  - Les besoins de LAN_B1 sont couverts par l'attribution du préfixe 172.16.0.192/28.
  - Les besoins de LAN_B2 sont couverts par l'attribution du préfixe 172.16.0.208/28.
  - Toute la zone Sud peut être annoncée comme une seule route 172.16.0.192/27 via le routeur B.
- Les liens « serial » WAN_AB, WAN_AC et WAN_AD se voient attribuer respectivement les préfixes 172.16.0.244/30, 172.16.0.248/30 et 172.16.0.252/30.

Imaginons les conséquences sur le routage :

- Un routeur extérieur à l'entreprise n'a besoin que d'une seule route vers 172.16.0.0/24 pour faire progresser des datagrammes vers des adresses de l'entreprise.

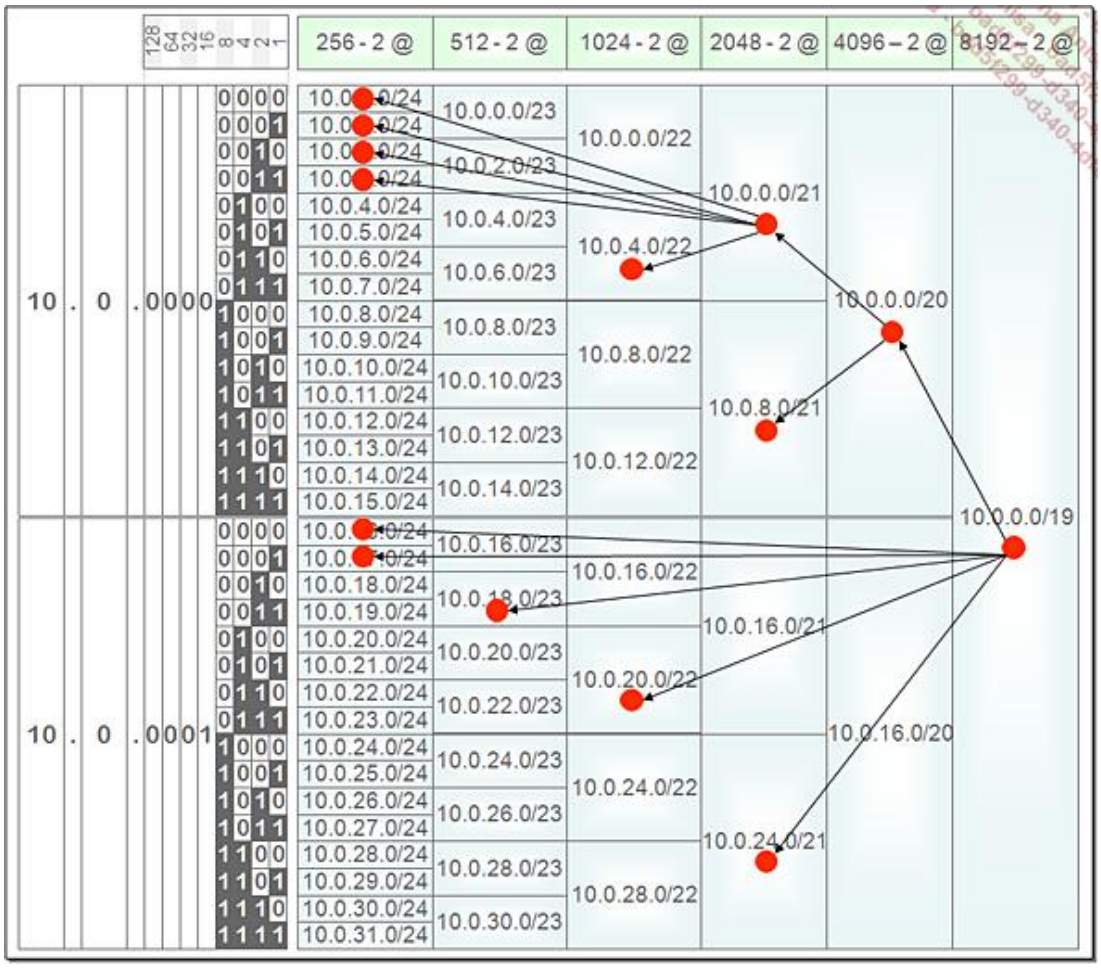
Dans l'entreprise :

- Un routeur extérieur au site Ouest n'a besoin que de la route vers 172.16.0.0/25 pour faire progresser les datagrammes vers des adresses de ce site.

- Un routeur extérieur au site Sud n'a besoin que de la route vers 172.16.0.192/27 pour faire progresser les datagrammes vers des adresses de ce site.

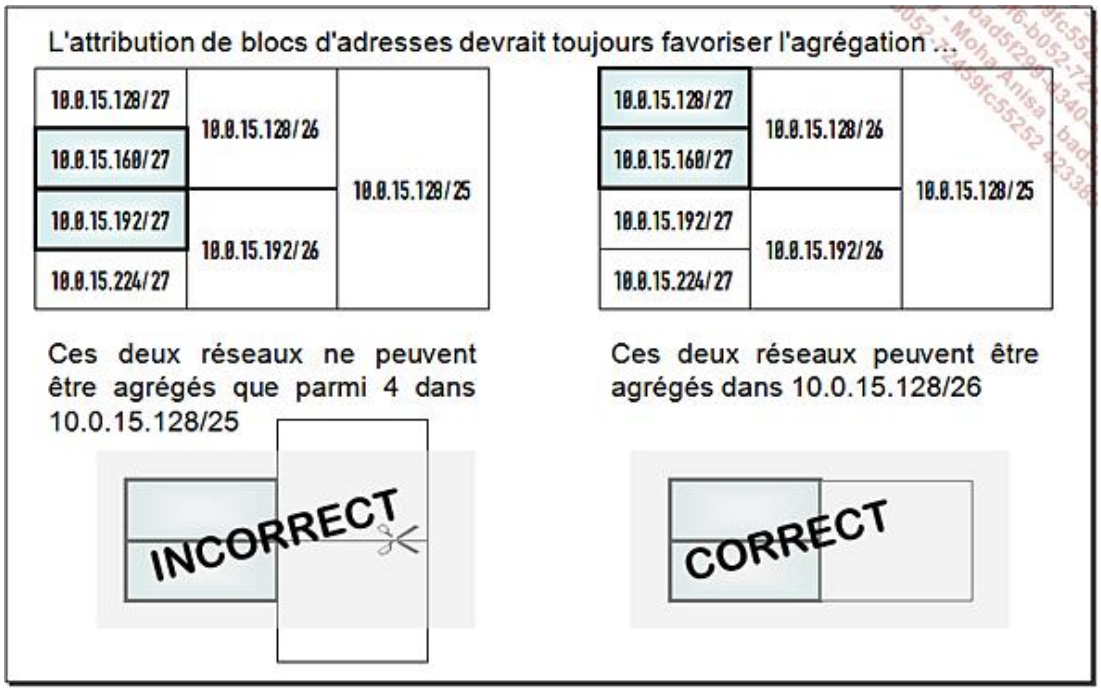
- Un routeur extérieur au site Est n'a besoin que de la route vers 172.16.0.128/26 pour faire progresser les datagrammes vers des adresses de ce site.

Le point de départ était un préfixe /24. Le découpage maximal fournit des préfixes /30. Une fois le tableau construit, la méthode a consisté à puiser les préfixes afin de satisfaire de la façon la plus précise les besoins d'adresses et de structuration. L'administrateur peut aussi décider de déléguer à son tour un préfixe. Le tableau suivant propose une méthode afin de mémoriser les attributions déjà réalisées (ce n'est pas le cas Primrose) :



Chaque pastille représente un espace délégué (l'administrateur en a confié la responsabilité à un collaborateur) ou affecté. On pourrait imaginer un code des couleurs selon qu'un espace est délégué, affecté en partie ou affecté totalement.

La représentation d'une arborescence depuis le préfixe initial n'est pas innocente. En effet, l'agrégation de routes possible avec VLSM n'est obtenue que si l'administrateur réalise une assignation topologique des blocs d'adresses issus de la subdivision du préfixe initial.



En résumé

- Le plan d'adressage permis par VLSM est hiérarchique mais devrait en outre tendre vers un adressage topologique.
- À cette condition, VLSM procure deux avantages déterminants : une utilisation efficace de l'espace d'adresses et une possibilité d'agrégation de routes.
- Souvenez-vous que le masque le plus ajusté pour satisfaire les besoins d'une liaison point à point est 255.255.255.252 (longueur de préfixe /30).

2. CIDR (Classless InterDomain Routing)

RFC utiles :

- RFC 2050 - Internet Registry IP Allocation Guidelines - Novembre 1996

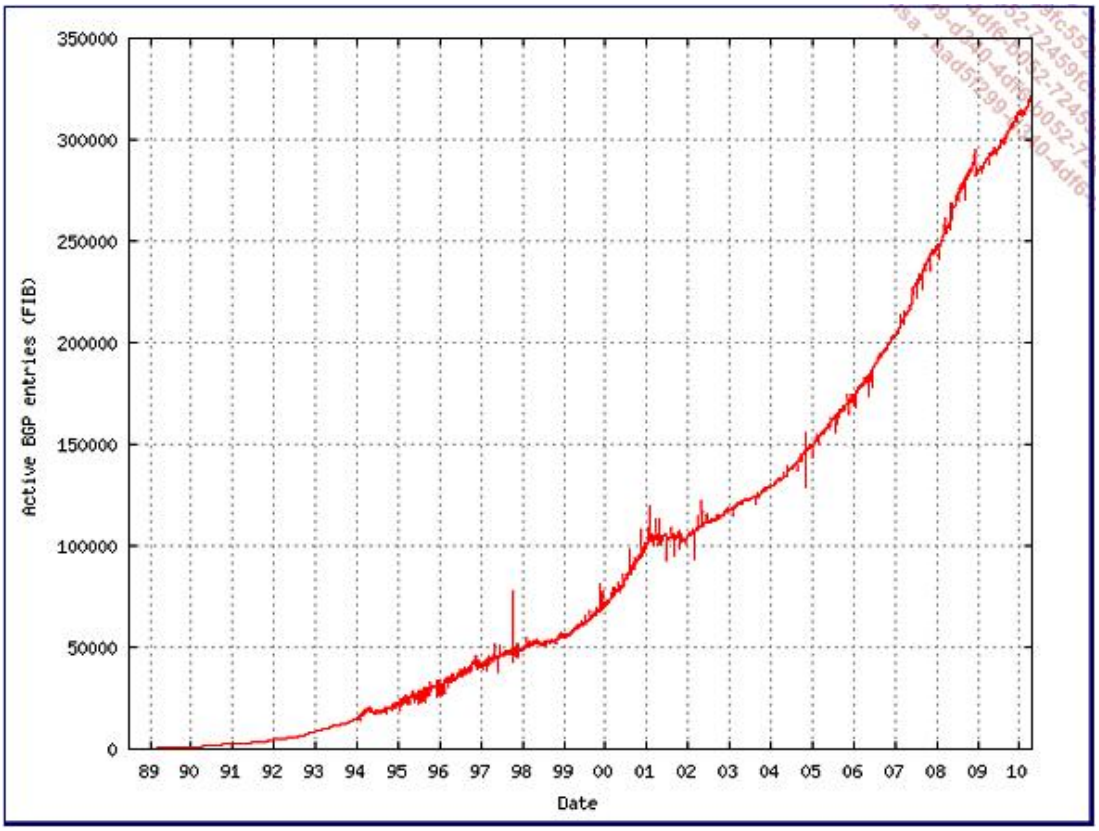
Avec le système de classes présenté au début de ce chapitre, une adresse IP porte à elle seule trois informations : le masque naturel de la classe, l'adresse du réseau et l'adresse de l'hôte dans ce réseau.

Mais dès 1990, il apparaît que parmi les freins qui pourraient nuire au développement d'Internet, il faut compter :

- La pénurie d'adresses qui pour une bonne part résulte du système de classes.
- La charge croissante des tables de routage. En effet, pour un site dont les besoins sont intermédiaires entre la classe C et la classe B, il est plus facile d'attribuer plusieurs adresses réseau de classe C qu'une seule adresse réseau de classe B mais que chaque fois que cela se produit, les tables de routage s'alourdissent de plusieurs lignes. Puisqu'il est possible d'allouer 2 millions d'adresses réseau de classe C, il est également certain que l'on va rapidement saturer la mémoire des routeurs principaux de l'Internet qui doivent maintenir de telles tables !

Visitez le site http://www.cidr-report.org/as2.0/

Depuis la page d'accueil, cliquez sur le lien « PLOT : BGP Table size ». Observez la version actualisée de l'évolution des tables de routage :



En novembre 1991, lors de la réunion IETF de Santa Fe, l'IETF crée les groupes de travail ROAD (*Routing and Addressing*) et ALE (*Address Lifetime Expectations*), chargés d'étudier les trois problèmes suivants :

- la pénurie des réseaux de classe B ;
- la croissance des tables de routage des routeurs principaux ;
- la pénurie des adresses de machines.

En mars 1992, lors de la réunion IETF de San Diego, le groupe ROAD propose d'adopter CIDR (*Classless Inter-Domain Routing*) qui apporte une solution temporaire au problème de la croissance des tables de routage, solution qui consiste à agréger les espaces d'adresses contigus (patientez). CIDR sera finalement adopté en novembre 1992 et rapidement inclus dans les protocoles de routage.

Quelles sont les conséquences si l'on décide d'abandonner le système de classes ? Essentiellement, il n'existe plus de « masque naturel ». Il devient impossible de déduire le masque d'une adresse selon la valeur du premier octet de cette adresse. La solution proposée par CIDR consiste à faire accompagner l'adresse IP de son masque et remplacer l'ancienne adresse IP « classée » par le couple « adresse IP / masque de réseau ».

Mais on conviendra que noter par exemple 192.168.1.0/255.255.255.0 n'est guère commode. En observant qu'un masque est constitué d'un nombre N de bits à 1 suivi d'un nombre (32-N) de bits à 0, il suffit de préciser la valeur N pour spécifier ce masque. Avec CIDR, la notation de l'adresse 192.168.1.2/255.255.255.0 devient **192.168.1.2/24**. Les anciens masques de classe A, B et C deviennent respectivement /8, /16 et /24.

Depuis une époque récente, sont nommés « **préfixe** » les bits de l'adresse qui représentent la partie réseau NetID. Le nombre N devient alors « longueur de préfixe ».

La notion CIDR apparaît dans les documents de l'IETF dès 1993. Ainsi, le RFC 1466 « *Guidelines for Management of IP Address Space* » de mai 1993 est encore fondé sur les classes mais évoque le travail en cours sur l'abandon des classes. Le RFC 2050 « *Internet Registry IP Allocation Guidelines* » de novembre 1996 se substitue au RFC 1466 et abandonne la notion de classes d'adresse au profit de CIDR.

C'est l'IANA qui est en charge de la gestion de l'espace d'adressage IP. Depuis CIDR, l'IANA a segmenté l'espace d'adressage en 256 blocs de taille /8 numérotés de 0/8 à 255/8. Chacun de ces blocs représente 16 millions d'adresses (24 bits). Puis, l'IANA délègue l'administration de ces segments à cinq RIR (*Regional Internet Registry*) selon la cartographie suivante :

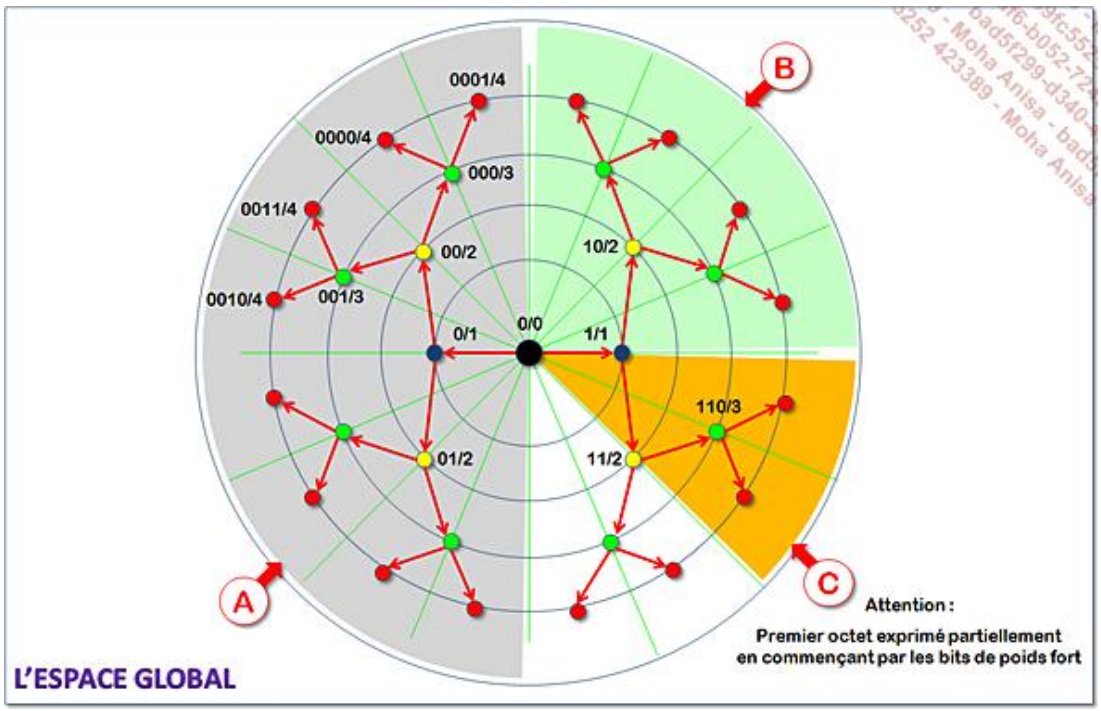
Les adresses allouées pour l'Europe sont gérées par le RIPE NCC (Réseaux IP Européens - Network Coordination Centre, www.ripe.net). Les segments délégués par l'IANA au RIPE NCC sont au nombre de 30, on peut découvrir quels sont ces segments en consultant le document http://www.iana.org/assignments/ipv4-address-space/. Pour un routeur de l'Internet situé aux Etats-Unis, router un paquet vers l'Europe revient à découvrir que le premier octet de l'adresse IP de destination appartient à l'une des 30 valeurs possibles pour l'Europe (62, 77 à 95, 141, 145, 151, 188, 193 à 195, 212, 213, 217).

Dans ce modèle CIDR, les hôtes et les routeurs ne font pas d'hypothèses sur l'utilisation de l'adressage dans l'internet. Les espaces d'adresse de Classe D (diffusion groupée IP) et de Classe E (expérimental) sont préservés, bien que ceci soit principalement une politique d'allocation.

Par définition, CIDR comprend trois éléments :

- Une allocation d'adresse topologiquement significative, le paragraphe portant sur VLSM a illustré cette notion.
- Des protocoles de routage capables d'agréger les routes de couche réseau.
- Un algorithme de routage cohérent (recherche de correspondance de préfixe la plus longue, patientez).

Nous devrions normalement adopter la terminologie « préfixe réseau » et abandonner les termes réseaux et sous-réseaux. Tout réseau est en fait un sous-réseau de l'espace global 0/0 :



Les anciens espaces de classe A, B et C ne sont représentés que pour mémoire. L'espace global est 0/0. Avec une longueur de préfixe 1, le préfixe ne peut prendre que les deux valeurs 0 ou 1 en binaire (0 ou 128 en décimal). Aux blocs 0/8 et 127/8 près, le préfixe 0/1 correspond à l'ancienne classe A (demi-cercle de gauche). Toujours exprimé en binaire, le préfixe 1/1 est à son tour divisé en deux pour donner 10/2 et 11/2. Le préfixe 10/2 correspond à l'ancienne

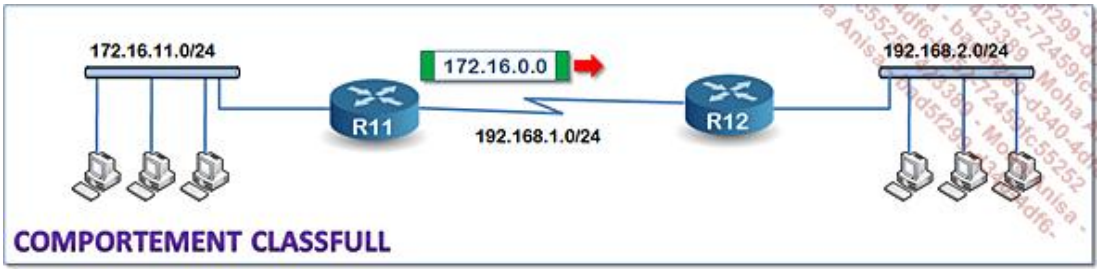
classe B. Enfin le préfixe 11/2 est à son tour divisé pour donner 110/3 et 111/3. Le préfixe 110/3 correspond à l'ancienne classe C. Cette modeste figure permet de mieux appréhender le caractère hiérarchique et topologique de ce qu'est devenu l'adressage IP. Un nœud donné du réseau (un préfixe) agrège tous les nœuds de rang inférieur qu'il a engendré. Avouez que vous n'aviez peut-être pas imaginé l'adressage IP ainsi !

Routage avec ou sans classe

1. Routage avec classe

a. Informations échangées

Un protocole de routage tel RIPv1 est dit avec classe (*class full*) parce que l'information de route est transportée sans l'information de masque. Les adresses sont considérées selon les cas avec le masque naturel (le masque de classe) ou avec le masque appliqué à l'interface qui reçoit l'annonce de route.



b. Comportement de l'algorithme de routage

Dans un routage avec classe, le processus de routage extrait un paquet d'une file d'attente d'entrée. Imaginons qu'il s'agisse du premier paquet d'un nouveau flux vers une adresse de destination pas encore présente dans le cache de commutation rapide (relire si nécessaire la section Partage de charge par destination et « *Fast Switching* » du chapitre Le routage statique). Le routeur lit une première fois la table de routage en séquence à la recherche de la partie réseau de l'adresse de destination. C'est le masque de classe A, B ou C (le masque naturel) qui est utilisé pour déterminer quelle est l'adresse réseau recherchée. Si le routeur ne trouve pas de correspondance pour un réseau majeur (un réseau de classe A, B ou C), le paquet est éliminé.

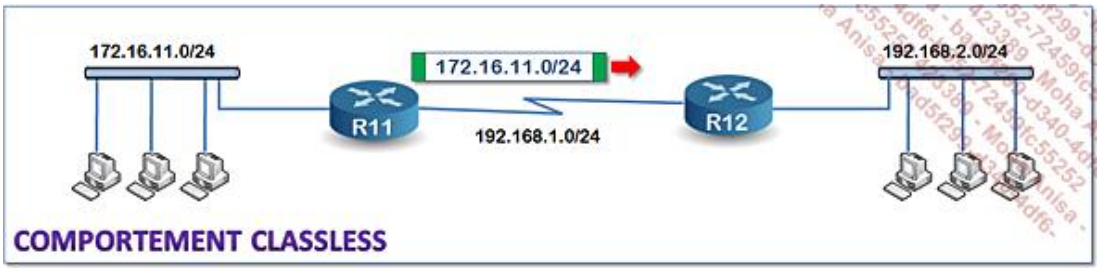
Si le routeur trouve une correspondance pour un réseau majeur, alors il recherche dans la table de routage les sous-réseaux connus de ce réseau majeur. Si une correspondance est trouvée, alors le paquet est confié à l'interface de sortie adéquate. Dans le cas contraire, le paquet est éliminé.

À chaque fois qu'un paquet est éliminé parce que le routeur n'a pas trouvé de correspondance dans la table de routage, le routeur génère un message ICMP type Destination injoignable vers la source du paquet.

2. Routage sans classe

a. Informations échangées

Déployer un adressage VLSM et basculer dans un monde CIDR nécessite d'échanger des informations de routes accompagnées de leur masque. Les protocoles de routage qui ont cette capacité sont dits sans classe (*class less*). RIP dans sa version 2 a intégré cette famille qui compte également les protocoles BGP (*Border Gateway Protocol*), EIGRP (*Enhanced Interior Gateway Routing Protocol*), IS-IS (*Intermediate System - Intermediate System*) et OSPF (*Open Shortest Path First*).

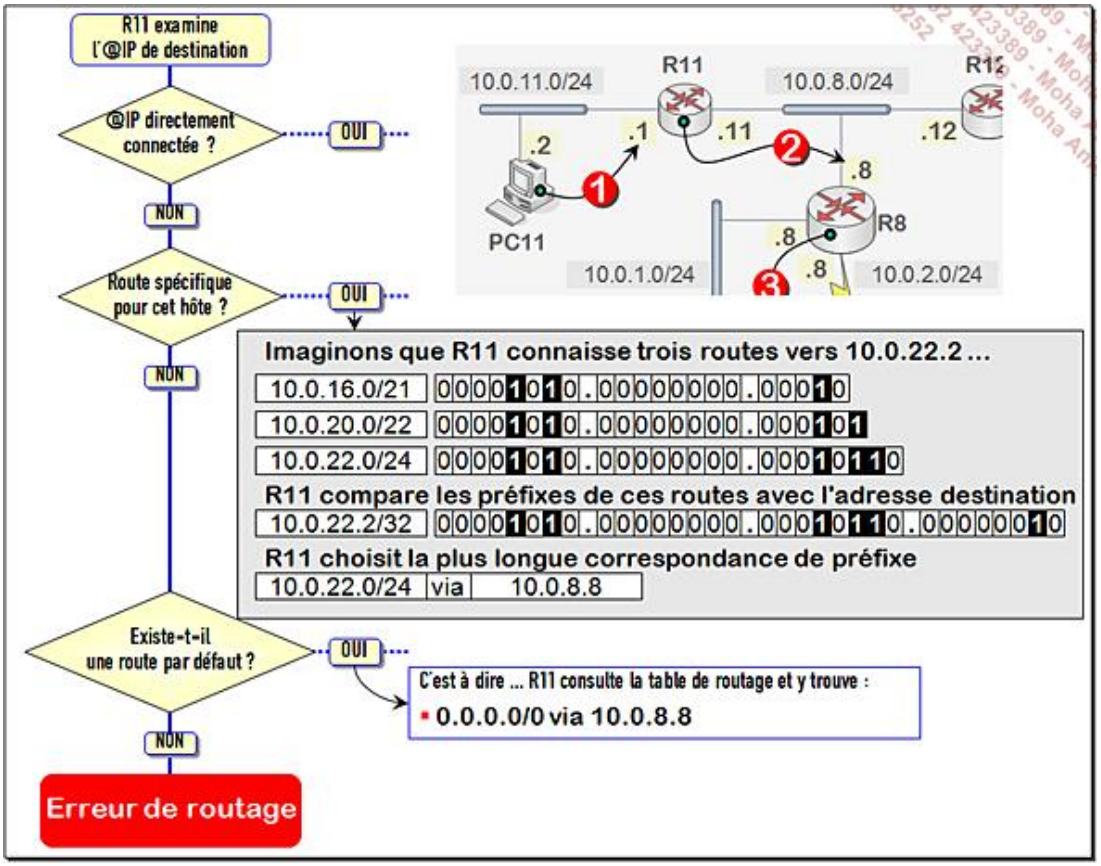


b. Comportement de l'algorithme de routage

Dans un environnement CIDR, la commande **ip classless** en mode de configuration globale permet d'abandonner le

comportement avec classe pour adopter un comportement visant à trouver la meilleure super-route (*the best supernet*) dans la table de routage. Depuis la version 11.3 de l'IOS, cette commande est activée par défaut.

Le processus de routage utilise l'algorithme de recherche de correspondance de préfixe la plus longue (*Longest Match based Forwarding Algorithm*). En quoi consiste cet algorithme ? Toutes les routes n'ont pas le même degré d'acuité. Vous êtes à Bruxelles et vous allez à Marseille. Au premier croisement, deux routes se présentent : le panneau indicateur de la première indique « Toutes directions », le panneau de la seconde indique « Marseille ». Vous êtes intrigué mais vous choisissez la seconde. Ainsi, le préfixe 10.0.22.0/24 englobe la machine 10.0.22.2 mais le préfixe 10.0.16.0/21 englobe le préfixe 10.0.22.0/24 et donc la machine 10.0.22.2. Dans sa table de routage, pour l'adresse de destination 10.0.22.2, le routeur préfèrera la route la plus spécifique 10.0.22.0/24 à la route la plus générale 10.0.16.0/21 :



Pour trouver la route la plus spécifique, le processus de routage utilise l'algorithme de recherche de correspondance de préfixe la plus longue (*Longest Match based Forwarding Algorithm*). Ce comportement est dit « sans classe ».

Limitations de RIPv1

Le message de mise à jour RIPv1 ne contient que le strict minimum d'information de topologie permettant aux routeurs RIP d'assurer leur tâche. Et paradoxalement, le message de mise à jour est un message « à trous », une bonne part de l'espace consommé restant inutilisé.

Le protocole RIPv1 a été conçu à une époque antérieure à l'apparition des systèmes autonomes et de la distinction IGP/EGP (*Interior Gateway Protocol*/*Exterior*...), à la notion de découpage en sous-réseaux ou aux préoccupations de sécurité. La carence la plus grave est constituée par l'absence de masque. Passe encore quand RIPv1 reçoit une annonce de réseau majeur car dans ce cas, il utilise le masque de la classe ou masque naturel. Mais les choses se compliquent quand RIPv1 reçoit une route dans laquelle une partie des bits hôte sont positionnés. Le routeur est alors incapable de déterminer le masque de sous-réseau ou pire de déterminer si la route annoncée l'est vers un hôte ou vers un sous-réseau. Ceci contraint les implémentations à supputer en apprenant par exemple le masque sur la configuration IP de l'interface depuis laquelle la route a été apprise. C'est le choix fait par CISCO.

RIPv2

Quelques RFC utiles :

RFC 1058	**Routing Information Protocol**	Juin 1988
RFC 1388	**RIP Version 2 Carrying Additional Information**	Janvier 1993 - remplacé par RFC1723
RFC 1723	**RIP Version 2 - Carrying Additional Information**	Novembre 1994 - remplacé par RFC2453
RFC 2453	**RIP Version 2**	Novembre 1998 - mis à jour par RFC 4822
RFC 2082	**RIP-2 MD5 Authentication**	Janvier 1997 - remplacé par RFC4822
RFC 4822	**RIPv2 Cryptographic Authentication**	Février 2007 - remplace RFC2082, met à jour RFC2453

1. Le protocole

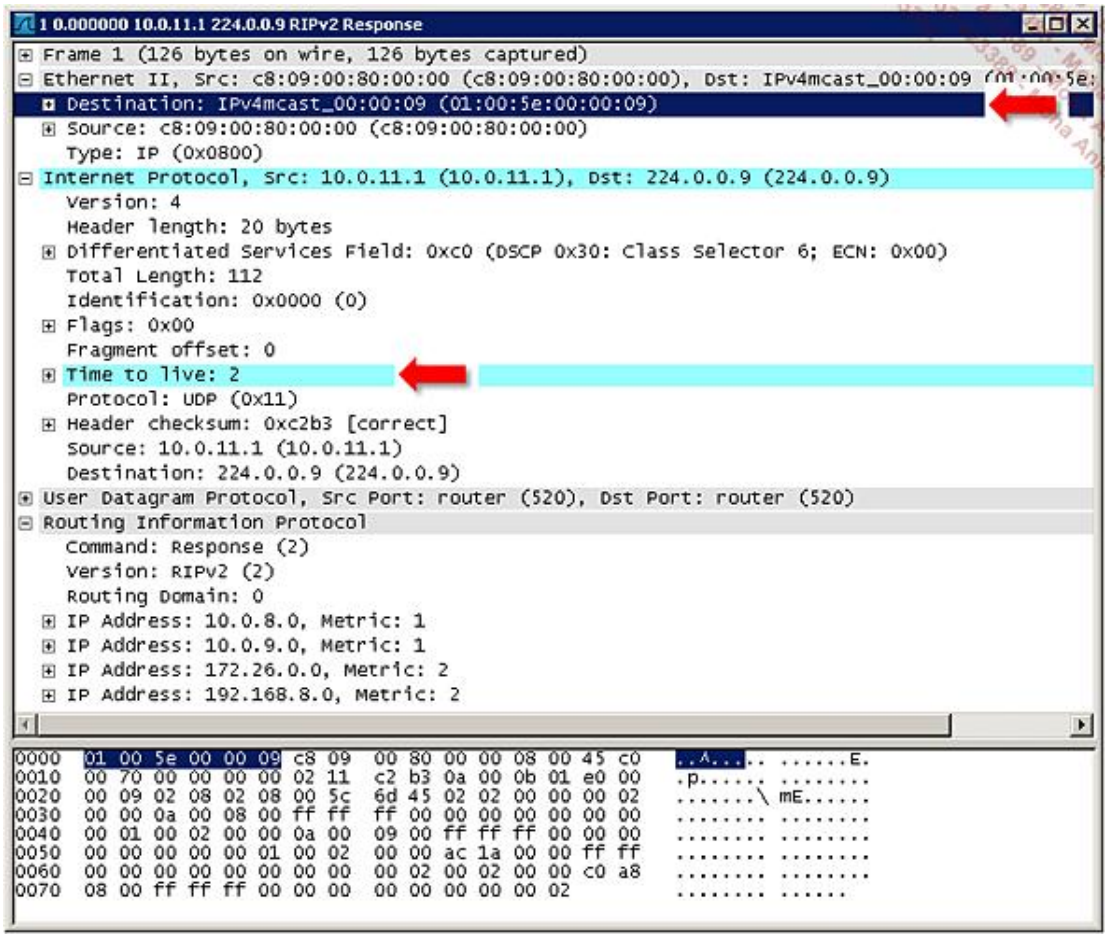
Les concepteurs de RIPv2 ne sont pas partis d'une page blanche. Il s'agissait d'apporter à RIPv1 ce qui lui manquait pour supporter l'abandon des classes d'adresses, pour l'essentiel :

- L'information de masque : chaque entrée de route dans une mise à jour RIPv2 comporte désormais l'information de masque associée à l'information d'adresse.
- Une information de prochain saut destinée à éviter des sauts inutiles dans certaines circonstances.
- Un marqueur de route destiné à distinguer une route transportée par RIP sans qu'il en soit à l'origine, c'est-à-dire une route externe.
- L'adoption de la multidiffusion : les messages RIPv2 ne sont plus diffusés mais multidiffusés.
- L'authentification optionnelle des messages.

Mais s'il ne fallait conserver qu'une de ces extensions, alors incontestablement ce serait l'ajout du masque car c'est lui qui permet à RIPv2 d'être qualifié de protocole sans classe (*classless*). L'ensemble des procédures, métriques, métrique infinie, temporisateurs, dispositifs de prévention des boucles, dispositifs visant la stabilité de RIPv1 sont conservés dans la version 2, à une exception près qui concerne la diffusion des mises à jour. RIPv1 les diffusait, RIPv2 les multi diffuse vers l'adresse réservée de classe D 224.0.0.9. Cela n'empêche pas les machines d'un réseau LAN connecté à une interface de routeur RIP de recevoir les mises à jour, que ces machines soient concernées ou non (on se souvient du comportement d'un commutateur qui doit acheminer une trame multicast : il inonde). Mais cela allège le traitement opéré par une machine non concernée : il lui suffit de décoder l'adresse de destination de couche 3 pour découvrir que le message ne l'intéresse pas.

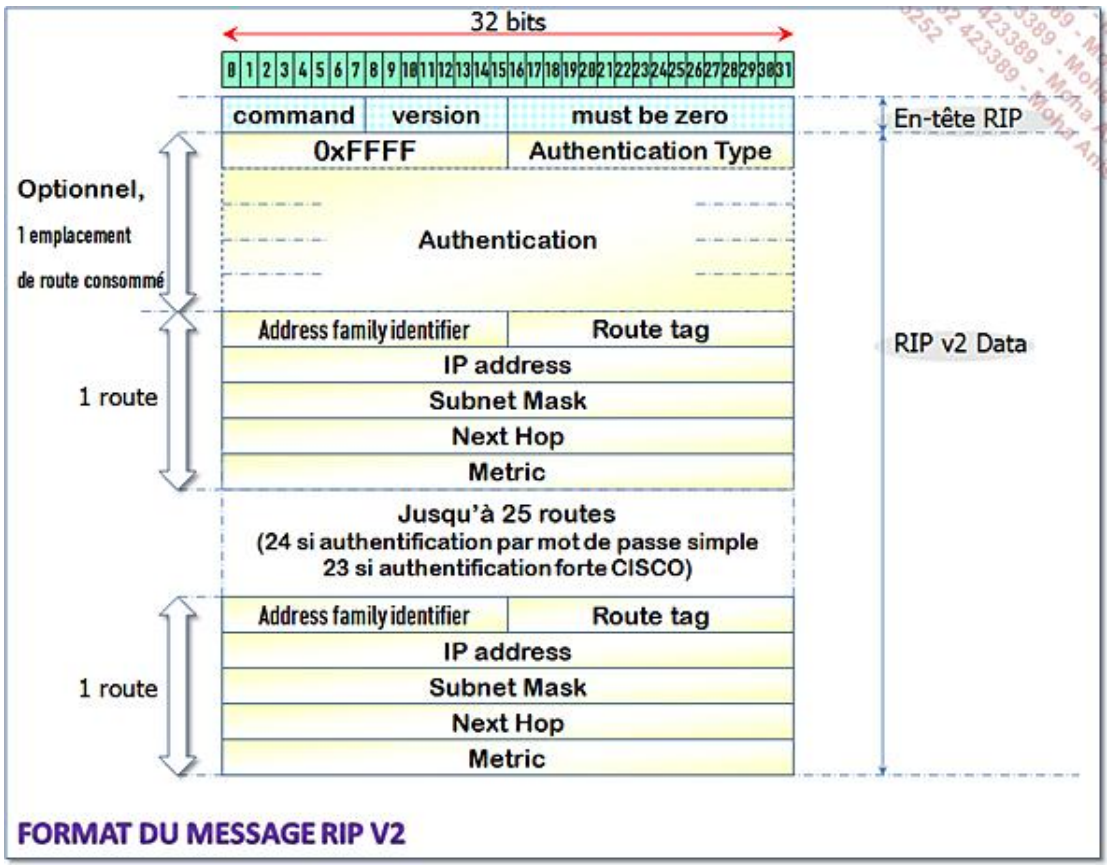
Dans le cas d'un réseau à diffusion, on se souvient que l'appartenance à un groupe de multidiffusion est répercutée en couche 2. L'adresse de couche 2 correspondant à 224.0.0.9 est 01:00:5*E*:00:00:09, le mode d'obtention de cette adresse est décrit dans la section Adresses de multidiffusion du chapitre Annexes.

La capture cap_2G_05.pcap, disponible en téléchargement sur le site ENI, a été réalisée sur un lien LAN. Observez le décodage d'un paquet de mise à jour ci-dessous. Observez notamment l'adresse de multidiffusion de couche 2. Observez également la valeur TTL attribuée qui limite à deux sauts l'espérance de vie d'un paquet de mise à jour RIP. Normal, un paquet de mise à jour RIP est destiné aux voisins, aucune raison de lui donner une espérance de vie supérieure :



a. Comparaison des messages de RIPv2 et RIPv1

Le message RIPv2 reprend le format du message RIPv1 et met à profit ses vastes champs inutilisés. Ainsi, quand l'authentification n'est pas utilisée, le nombre de routes qu'un message de mise à jour peut embarquer n'est pas modifié (25, 24 quand l'authentification par mot de passe simple est utilisée). Comme RIPv1, le message RIPv2 est transporté dans un datagramme UDP dont les ports source et destination sont 520 :

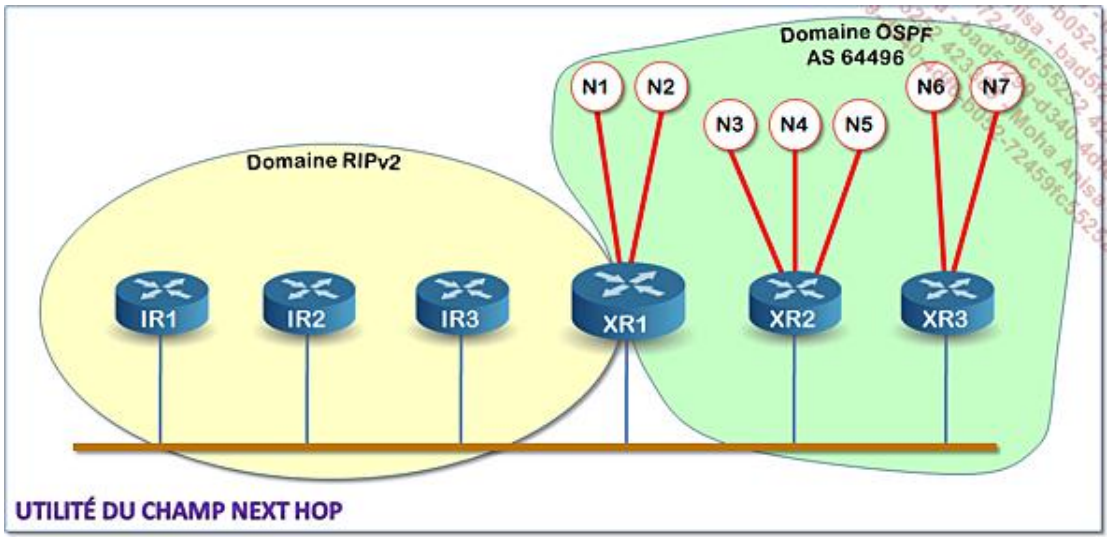


Les arguments du message RIPv2 sont les suivants :

- **command** → pour ce qui nous concerne, deux valeurs possibles :
  - 1 : le message est une requête ;
  - 2 : le message est une réponse ;
  - Les autres valeurs sont soit obsolètes, soit réservées.
- **version** → 2 pour RIP v2.
- **address family identifier** → 2 pour IP dans les messages de réponse, 0 dans un message de requête (patientez).
- **Route tag** → l'objet du champ marqueur de route est de fournir une méthode permettant de distinguer une route RIP interne au domaine RIP d'une route externe issue d'un EGP ou d'un autre IGP. Par exemple, le RFC 2453 suggère d'y placer le numéro de système autonome dans lequel la route a été apprise.
- **IP address** → contient indifféremment :
  - Une adresse hôte.
  - Une adresse de sous-réseau.
  - Une adresse réseau.
  - 0 qui distingue une route par défaut.
- **Subnet Mask** → masque associé à l'adresse. Puisque chaque route est désormais associée à un masque, plus rien ne s'oppose à l'utilisation de différentes longueurs de préfixe et donc à la réalisation d'un

adressage VLSM.

- **Next Hop** → adresse de saut suivant : adresse IP à laquelle il convient de remettre directement les paquets sans passer par le routeur à l'origine de cette annonce RIP. L'adresse de saut suivant est une adresse directement connectée au sous-réseau sur lequel est effectuée l'annonce. 0000 dans le champ Next Hop indique que le routage doit se faire via le routeur à l'origine de l'annonce RIP. L'objet de ce champ est d'éliminer des sauts inutiles dans certaines circonstances particulières, comme l'illustre l'exemple suivant proposé par le RFC et à peine modifié :



XR1 est routeur frontière entre les deux domaines RIP et OSPF. De ce fait, seul XR1 échange des routes RIP avec IR1, IR2 et IR3. Du côté RIP, XR1 annonce le réseau N3 de la façon suivante :

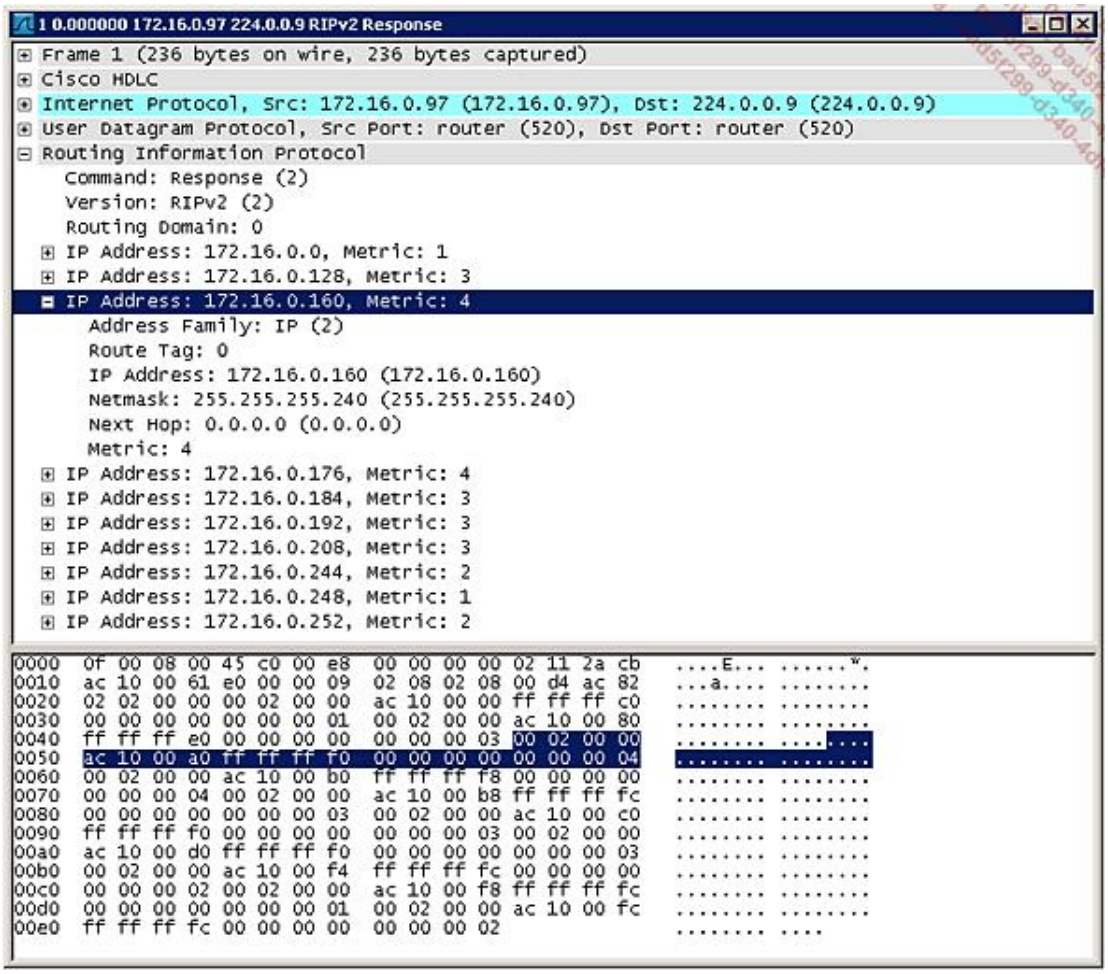
- Marqueur de route = 64496 (N° de système autonome du domaine OSPF) ;
- @réseau = N3 ;
- Next Hop = XR2.

Ainsi, un routeur IR qui aurait à faire progresser un paquet destiné à N3 le ferait via XR2. Sans le champ Next Hop, XR2 et XR3 auraient dû participer au protocole RIP pour éviter le saut inutile via XR1.

- **metric** → coût de la route exprimé en nombre de sauts de 1 à 16, 1 signifiant une route directement connectée, 16 caractérisant une route injoignable.

La taille maximale du message RIP est 512 octets, sans compter les en-têtes IP et UDP (pour mémoire, un en-tête IP sans options occupe 20 octets, un en-tête UDP occupe 8 octets). En ôtant l'en-tête RIP, il reste 508 octets utiles au transport de routes. Puisqu'une route occupe 20 octets, un message RIP peut transporter jusqu'à 25 routes (24 quand l'authentification simple est utilisée). Somme toute, un datagramme UDP qui transporte un message RIP ne dépassera pas 512 octets (25*x*20 + 4 + en-tête UDP).

En forme de synthèse du format de paquet RIPv2, voici le décodage d'un message de mise à jour à l'aide de Wireshark :



Observez les adresses source et destination du datagramme IP, les ports source et destination du datagramme UDP, les différentes entrées de route, le décodage de l'entrée de route vers 172.16.0.160/28.

b. Compatibilité avec RIPv1

Grâce à la prévoyance de ses concepteurs, RIPv1 est heureusement très tolérant quant aux mises à jour qu'il reçoit. Si le champ Version comporte 1 bien sûr, il s'attend à ce que les champs inutilisés le soient effectivement et écarte toute mise à jour non conforme. Mais si le champ Version est supérieur à 1, les champs censés être inutilisés sont ignorés et le message est exploité selon le format connu de RIPv1. C'est cette particularité de RIPv1 qui permet la rétrocompatibilité de RIPv2.

Toutefois, le RFC 2453 prévoit deux commutateurs de compatibilité dont l'objet est de permettre une adaptation facile de RIPv2 avec des implémentations exotiques de RIPv1. Ces deux commutateurs devraient être configurables au niveau interface. Le premier de ces commutateurs règle l'émission des mises à jour et prévoit quatre réglages :

- RIP-1 → seuls les messages RIPv1 sont envoyés.
- RIP-1 compatible → les messages RIPv2 sont diffusés plutôt que multi diffusés. Le RFC conseille d'utiliser ce réglage avec prudence.
- RIP-2 → les messages RIPv2 sont multi diffusés conformément au protocole RIPv2 natif.
- « none » → pas d'émission de messages RIP sur cette interface. La commande activant ce dernier mode est déjà connue puisqu'il s'agit de la commande **passive-interface**.

Le second commutateur ajuste le comportement du processus RIP lorsqu'il reçoit des messages. À nouveau, quatre réglages sont prévus :

- RIP-1 seulement ;

- RIP-2 seulement ;
- les deux ;
- aucun.

Dans le cas de l'IOS, les trois premiers réglages correspondent à des commandes examinées dans les paragraphes qui suivent, le quatrième peut être résolu selon différentes méthodes : on peut par exemple mettre en place une liste d'accès étendue afin de filtrer le trafic destiné au port UDP 520 ou établir un filtrage de routes à l'aide de la commande **distribute-list** en configuration de routeur. Dans le second cas, on est au-delà du périmètre prévu pour cet ouvrage.

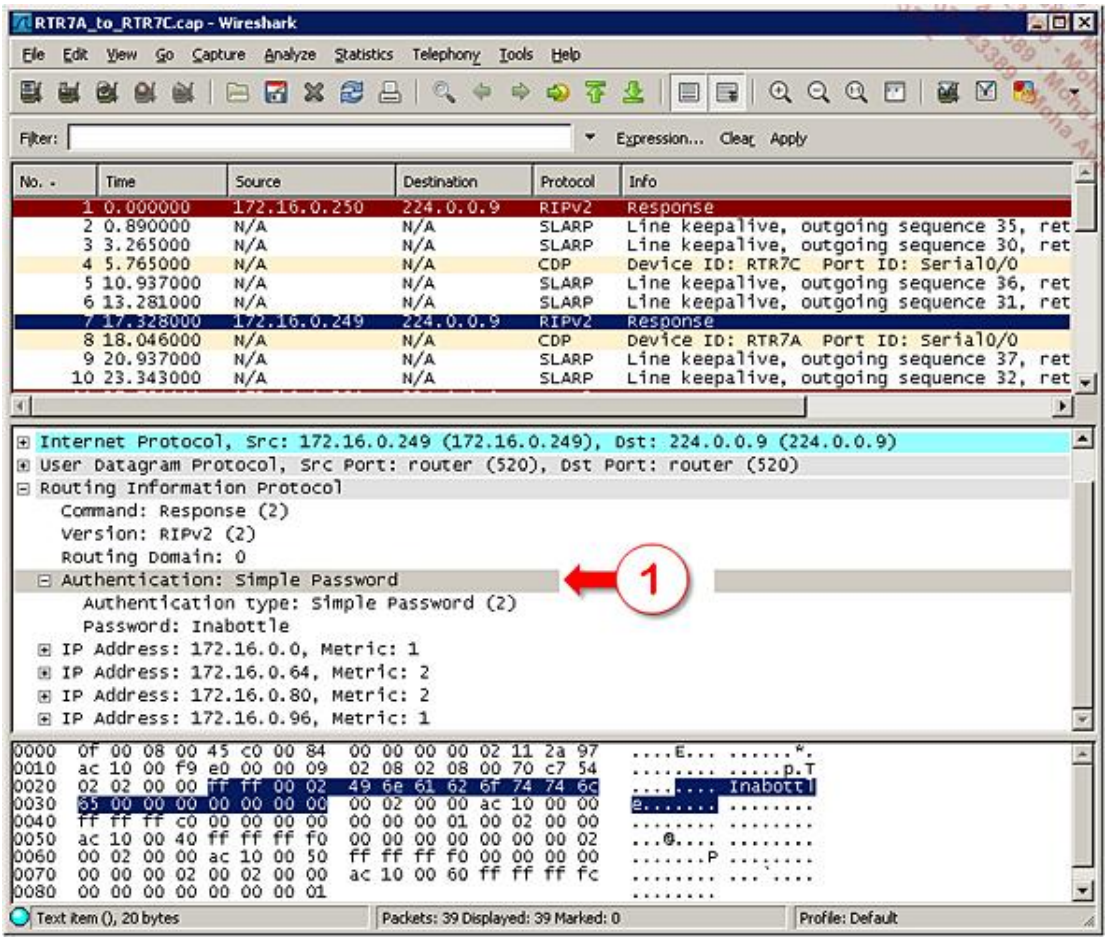
c. Authentification

Avec RIPv1, il est facile de corrompre la table de routage d'un routeur quelconque. Il suffit qu'un attaquant émette des mises à jour falsifiées en prétendant connaître un certain nombre de réseaux avec le meilleur coût pour que les paquets normalement destinés à ces réseaux transitent par lui. Les concepteurs de RIPv2 ont prévu une parade en donnant à l'émetteur d'une mise à jour le moyen de s'authentifier par un mot de passe.

Plutôt que de concevoir un nouveau format de message, ce qui n'aurait pas manqué de causer des problèmes de compatibilité avec RIPv1, les concepteurs ont préféré détourner une entrée de route de son usage normal. Le nombre maximal d'entrées de route que peut comporter un message de mise à jour passe ainsi à 24.

L'entrée de route dédiée à l'authentification, quand elle est présente, se distingue par le champ « Address family identifier » établi à 0xFFFF. Dans cette prétendue entrée de route, le RFC a prévu un champ type d'authentification ce qui permet d'envisager de nouveaux types de façon simple sans remettre en question le format du paquet. On se souvient en effet que le domaine du chiffrement est en évolution rapide. Le seul type prévu par le RFC est le type 0x02 qui correspond à un mot de passe simple. Dans ce cas, les 16 octets qui suivent portent le mot de passe en clair qui, par conséquent, ne peut dépasser 16 caractères. Le mot de passe est justifié à gauche dans ce champ, les octets inutilisés restent à 0.

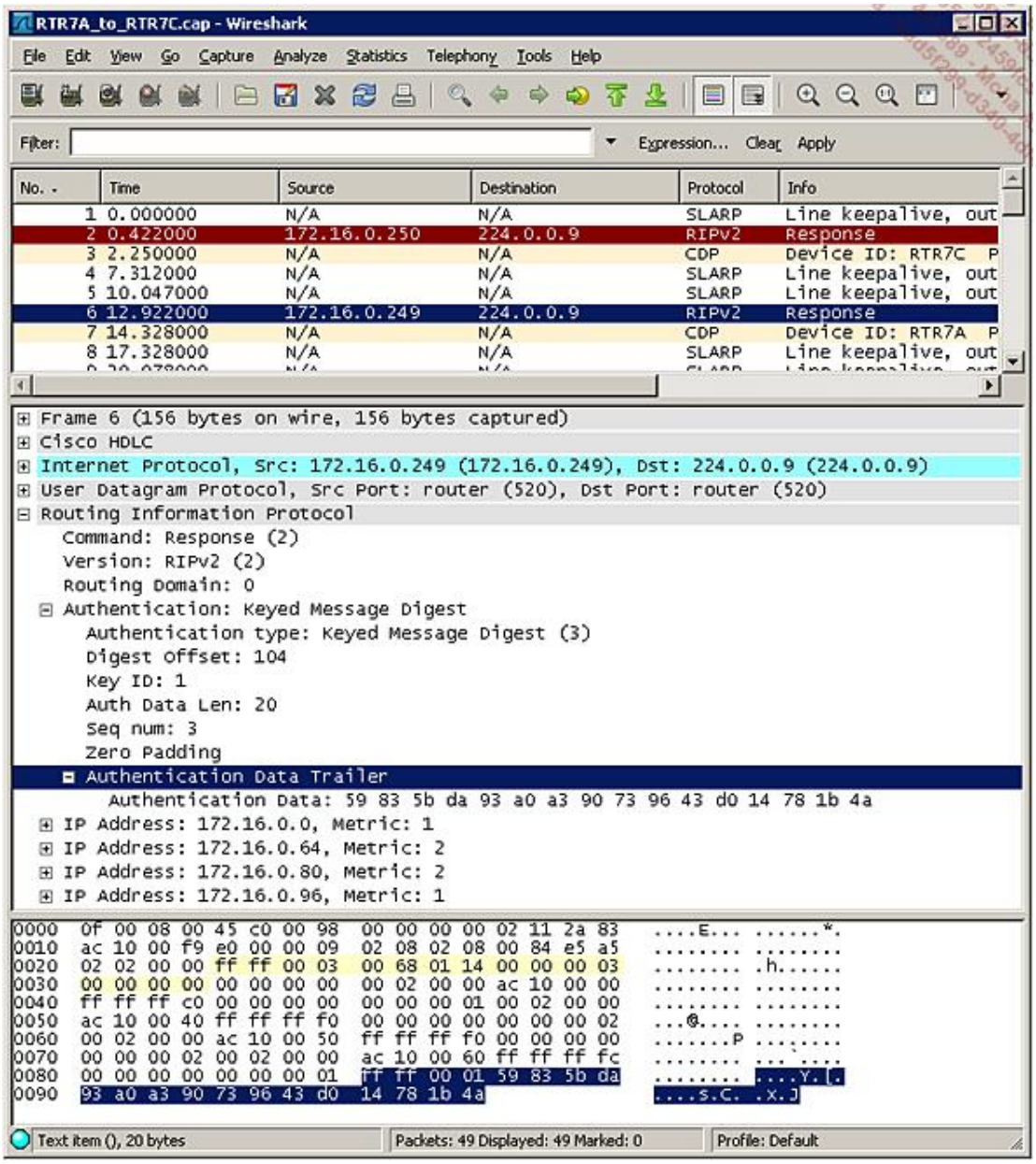
La capture suivante réalisée à l'aide de Wireshark montre un paquet de mise à jour authentifié à l'aide d'un mot de passe simple « *Inabottle* » (capture cap_2G_01.pcap disponible sur le site ENI) :



Observez l'entrée de route détournée de son usage premier pour embarquer le mot de passe. Comme le démontre

cette capture, quiconque peut écouter les mises à jour peut également apprendre le mot de passe. Il s'agit donc davantage d'être certain que les routeurs en cours de configuration RIP s'échangent bien leurs mises à jour sans être pollués par des mises à jour non désirées. Mais puisque le RFC a prévu un champ type d'authentification, il est facile de concevoir une authentification forte en créant un type supplémentaire. Ce qu'a fait CISCO en intégrant dans l'IOS la possibilité d'authentifier des mises à jour RIP à l'aide de signatures obtenues par hachage MD5 de la mise à jour avec un mot de passe faisant office de clé. Le routeur qui reçoit une telle mise à jour et qui connaît le même mot de passe calcule sa propre signature. Si les deux signatures reçues et calculées correspondent, alors le contenu de la mise à jour est exploité.

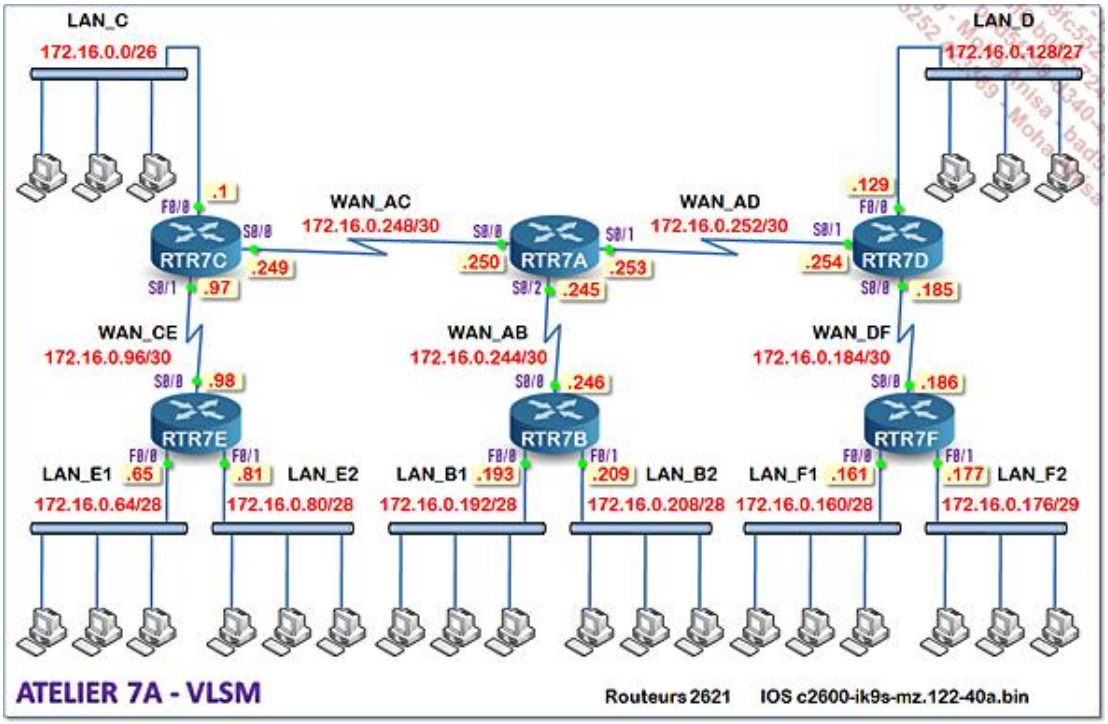
Concrètement, CISCO a étendu le procédé retenu par le RFC en dédiant non plus une mais deux entrées de route à l'authentification. La première entrée se distingue par le type d'authentification 0x03 tandis que la seconde entrée consommée, qui contient la signature, est placée en queue de mise à jour après toutes les entrées de route, ce qu'illustre la capture Wireshark suivante. Le contexte de cette capture est inchangé, seul le mode d'authentification est passé de simple à MD5 (capture cap_2G_02.pcap disponible sur le site ENI) :



Observez que Wireshark identifie parfaitement la dernière entrée de route comme devant être associée à l'information d'authentification.

2. Configuration de base

Proposons-nous de mettre en œuvre la topologie qui a servi à l'étude de cas VLSM du chapitre précédent :



Cette topologie comporte treize sous-réseaux issus du réseau majeur 172.16.0.0 et cinq masques différents de /26 à /30.

a. Activation du protocole

Toutes les interfaces ont déjà été configurées.

Le protocole s'active en trois temps :

1. Activer le processus RIP proprement dit à l'aide de la commande **router rip**.

2. Par défaut, le processus RIP une fois activé génère des mises à jour RIPv1 mais profite de mises à jour qu'elles soient v1 ou v2. Ce comportement est modifié à l'aide de la commande **version** en mode configuration de routeur.

3. Indiquer au processus quels réseaux majeurs doivent être pris en compte à l'aide d'une commande **network** par réseau majeur. Puisque l'ensemble des sous-réseaux a été obtenu par division du réseau 172.16.0.0, une seule commande **network** suffit :

RTR7A

```
RTR7A(config)#router rip
RTR7A(config-router)#version ?
<1-2>  version

RTR7A(config-router)#version 2
RTR7A(config-router)#network 172.16.0.0
RTR7A(config-router)#^Z
RTR7A#
```

RTR7B

```
RTR7B(config)#router rip
RTR7B(config-router)#version 2
RTR7B(config-router)#network 172.16.0.0
RTR7B(config-router)#^Z
RTR7B#
```

Cette configuration est à répéter sur l'ensemble des routeurs de la topologie.

La commande **debug ip rip** permet de se passer de Wireshark pour comprendre l'activité du protocole. Par exemple, sur RTR7A (1 tour complet c'est-à-dire toutes les mises à jour envoyées sur toutes les interfaces et toutes les mises à jour reçues de tous les routeurs voisins) :

```
RTR7A#debug ip rip
```

```
RIP protocol debugging is on
RTR7A#
00:19:37: RIP: sending v2 update to 224.0.0.9 via Serial0/0 (172.16.0.250)
00:19:37: RIP: build update entries
00:19:37:       172.16.0.128/27 via 0.0.0.0, metric 2, tag 0
00:19:37:       172.16.0.160/28 via 0.0.0.0, metric 3, tag 0
00:19:37:       172.16.0.176/29 via 0.0.0.0, metric 3, tag 0
00:19:37:       172.16.0.184/30 via 0.0.0.0, metric 2, tag 0
00:19:37:       172.16.0.192/28 via 0.0.0.0, metric 2, tag 0
00:19:37:       172.16.0.208/28 via 0.0.0.0, metric 2, tag 0
00:19:37:       172.16.0.244/30 via 0.0.0.0, metric 1, tag 0
00:19:37:       172.16.0.252/30 via 0.0.0.0, metric 1, tag 0
00:19:37: RIP: sending v2 update to 224.0.0.9 via Serial0/1 (172.16.0.253)
00:19:37: RIP: build update entries
00:19:37:       172.16.0.0/26 via 0.0.0.0, metric 2, tag 0
00:19:37:       172.16.0.64/28 via 0.0.0.0, metric 3, tag 0
00:19:37:       172.16.0.80/28 via 0.0.0.0, metric 3, tag 0
00:19:37:       172.16.0.96/30 via 0.0.0.0, metric 2, tag 0
00:19:37:       172.16.0.192/28 via 0.0.0.0, metric 2, tag 0
00:19:37:       172.16.0.208/28 via 0.0.0.0, metric 2, tag 0
00:19:37:       172.16.0.244/30 via 0.0.0.0, metric 1, tag 0
00:19:37:       172.16.0.248/30 via 0.0.0.0, metric 1, tag 0
00:19:37: RIP: sending v2 update to 224.0.0.9 via Serial0/2 (172.16.0.245)
00:19:37: RIP: build update entries
00:19:37:       172.16.0.0/26 via 0.0.0.0, metric 2, tag 0
00:19:37:       172.16.0.64/28 via 0.0.0.0, metric 3, tag 0
00:19:37:       172.16.0.80/28 via 0.0.0.0, metric 3, tag 0
00:19:37:       172.16.0.96/30 via 0.0.0.0, metric 2, tag 0
00:19:37:       172.16.0.128/27 via 0.0.0.0, metric 2, tag 0
00:19:37:       172.16.0.160/28 via 0.0.0.0, metric 3, tag 0
00:19:37:       172.16.0.176/29 via 0.0.0.0, metric 3, tag 0
00:19:37:       172.16.0.184/30 via 0.0.0.0, metric 2, tag 0
00:19:37:       172.16.0.248/30 via 0.0.0.0, metric 1, tag 0
00:19:37:       172.16.0.252/30 via 0.0.0.0, metric 1, tag 0
00:19:42: RIP: received v2 update from 172.16.0.254 on Serial0/1
00:19:42:      172.16.0.128/27 via 0.0.0.0 in 1 hops
00:19:42:      172.16.0.160/28 via 0.0.0.0 in 2 hops
00:19:42:      172.16.0.176/29 via 0.0.0.0 in 2 hops
00:19:42:      172.16.0.184/30 via 0.0.0.0 in 1 hops
00:19:50: RIP: received v2 update from 172.16.0.249 on Serial0/0
00:19:50:      172.16.0.0/26 via 0.0.0.0 in 1 hops
00:19:50:      172.16.0.64/28 via 0.0.0.0 in 2 hops
00:19:50:      172.16.0.80/28 via 0.0.0.0 in 2 hops
00:19:50:      172.16.0.96/30 via 0.0.0.0 in 1 hops
00:19:59: RIP: received v2 update from 172.16.0.246 on Serial0/2
00:19:59:      172.16.0.192/28 via 0.0.0.0 in 1 hops
00:19:59:      172.16.0.208/28 via 0.0.0.0 in 1 hops
RTR7A#u all
All possible debugging has been turned off
```

Par rapport à la même capture réalisée en RIPv1, on note quelques différences telle l'apparition de l'information prochain saut associée à chaque entrée de route. Puisque ce champ n'est pas exploité dans notre contexte, la valeur reste **via 0.0.0.0**. Observez également l'identification des mises à jour comme étant de la version 2 par le processus debug. Observez enfin l'adresse de destination multicast des mises à jour. À ce sujet, la commande **show ip interface** est édifiante. En voici un extrait pour l'interface S0/2 de RTR7A :

```
RTR7A#sh ip int s0/2
Serial0/2 is up, line protocol is up
  Internet address is 192.168.0.245/30
.........
Multicast reserved groups joined: 224.0.0.9
.........
RTR7A#
```

C'est parce que l'interface a été placée dans ce groupe Multicast qu'elle est capable d'exploiter les mises à jour RIPv2 reçues. Une adresse multicast ne confère pas une identité à une interface. L'adresse est celle d'un groupe et une même interface peut parfaitement être inscrite sur plusieurs groupes.

Une commande **show ip route summary** sur par exemple RTR7A rassure quant à l'efficacité du protocole RIPv2 et son caractère sans classe :

```
RTR7A#sh ip route summary
IP routing table name is Default-IP-Routing-Table(0)
Route Source    Networks    Subnets     Overhead    Memory (bytes)
connected       0           3           192         432
static          0           0           0           0
rip             0           10          640         1440
internal        2                                   2328
Total           2           13          832         4200
RTR7A#
```

Observez l'information en gras : 13 sous-réseaux. Une commande **show ip route** nous apprendrait que ces sous-réseaux sont répartis sur cinq masques différents, c'est effectivement ce que compte notre topologie, tout va bien.

Pour mémoire, voici le résultat d'une commande **show ip protocols** avant l'activation de la version 2 de RIP (Extrait) :

```
RTR7A#sh ip prot
Routing Protocol is "rip"
.........
  Default version control: send version 1, receive any version
    Interface             Send  Recv  Triggered RIP  Key-chain
    Serial0/0             1     1 2
    Serial0/1             1     1 2
    Serial0/2             1     1 2
.........
RTR7A#
```

C'est ainsi que l'on peut vérifier le comportement par défaut du processus RIP. Une fois activé, le processus génère des mises à jour RIPv1 mais profite de mises à jour qu'elles soient v1 ou v2. La même commande après passage à la version 2 (Extrait) :

```
RTR7A#sh ip prot
Routing Protocol is "rip"
.........
  Default version control: send version 2, receive version 2
    Interface             Send  Recv  Triggered RIP  Key-chain
    Serial0/0             2     2
    Serial0/1             2     2
    Serial0/2             2     2
  Automatic network summarization is in effect
  Maximum path: 4
  Routing for Networks:
    172.16.0.0
  Routing Information Sources:
    Gateway         Distance      Last Update
    172.16.0.254         120      00:00:00
    172.16.0.249         120      00:00:08
    172.16.0.246         120      00:00:21
  Distance: (default is 120)
RTR7A#
```

On le voit, en version 2 et par défaut, le processus RIP est beaucoup plus restrictif : il génère des mises à jour dans le format de la version 2 et ne prendra en compte que les mises à jour du même format. Ce comportement est modifiable (patientez).

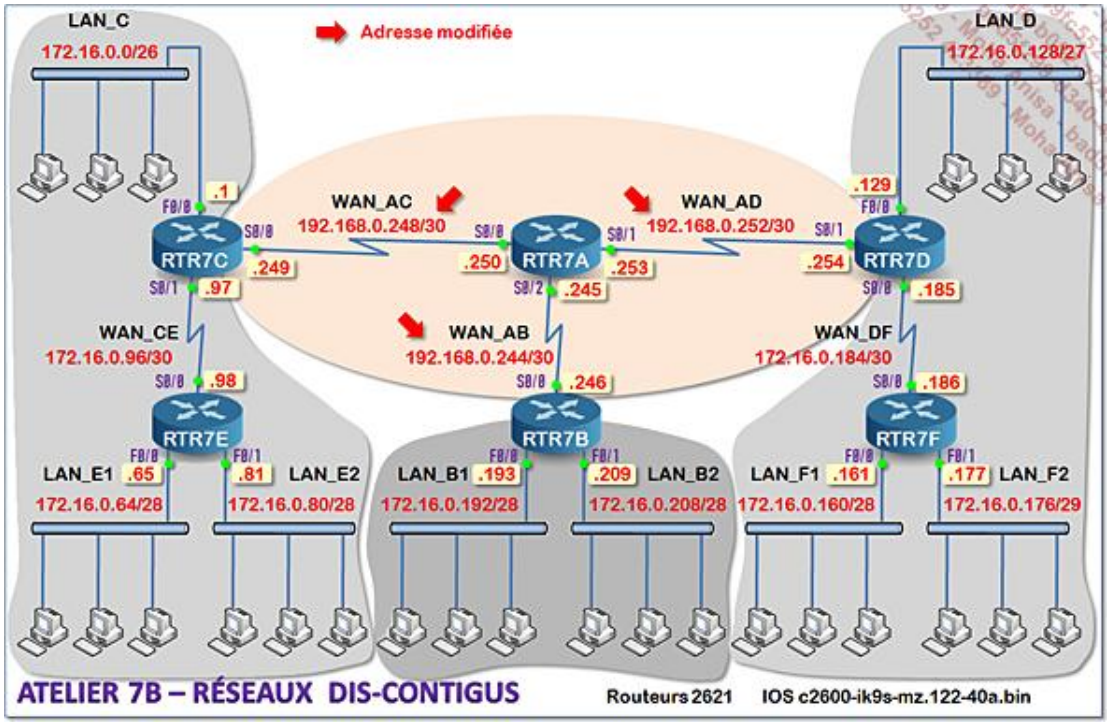
b. Commande passive-interface

La mise en œuvre de cette commande n'est pas impactée par la version du protocole RIP, merci de vous reporter à la section correspondante du chapitre Protocole de routage type DV RIPv1.

3. Configuration avancée

a. Réseaux discontigus

Si le contexte précédent constituait une bonne application d'un découpage à la mode VLSM, il n'est pas suffisant pour mettre en lumière le problème des réseaux discontigus. Affectons des adresses de l'ex-classe C aux liens WAN issus du routeur RTR7A afin de scinder le domaine couvert par 172.16.0.0 en trois parties :



Une fois les quatre routeurs concernés par le changement correctement configurés, une lecture de la table de routage du routeur RTR7A est édifiante : ce routeur connaît trois alternatives à coût égal vers 172.16.0.0 et s'apprête à faire de la répartition de charge. Tout premier paquet d'un flux destiné à 172.16.0.0/24 et traité par RTR7A a une chance sur trois d'être acheminé vers la partie qui convient du domaine couvert par 172.16.0.0 :

```
RTR7A#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
Gateway of last resort is not set

R    172.16.0.0/16 [120/1] via 192.168.0.249, 00:00:18, Serial0/0
                   [120/1] via 192.168.0.246, 00:00:15, Serial0/2
                   [120/1] via 192.168.0.254, 00:00:24, Serial0/1
     192.168.0.0/30 is subnetted, 3 subnets
C       192.168.0.248 is directly connected, Serial0/0
C       192.168.0.252 is directly connected, Serial0/1
C       192.168.0.244 is directly connected, Serial0/2
RTR7A#
```

Ceci est le fait du comportement par défaut de RIPv2 qui agrège ses annonces aux frontières des réseaux majeurs comme le faisait RIPv1. Ainsi les trois routeurs RTR7C, RTR7B et RTR7D annoncent 172.16.0.0 vers RTR7A ce, parce que leur interface vers ce routeur n'est pas dans le réseau 172.16.0.0. Rendre la vue à RTR7A implique de désactiver l'agrégation automatique (*summarization*) ce que l'administrateur doit configurer sur les trois routeurs frontières à l'aide de la commande **no auto-summary** en configuration de routeur :

RTR7C

```
RTR7C(config)#router rip
RTR7C(config-router)#no auto-summary
RTR7C(config-router)#^Z
RTR7C#wr
Building configuration...
[OK]
RTR7C#
```

RTR7B

```
RTR7B(config)#router rip
RTR7B(config-router)#no auto-summary
```

```
RTR7B(config-router)#^Z
RTR7B#wr
Building configuration...
[OK]
RTR7B#
```

RTR7D

```
RTR7D(config)#router rip
RTR7D(config-router)#no auto-summary
RTR7D(config-router)#^Z
RTR7D#wr
Building configuration...
RTR7D#
```

Bonus : il n'aura pas échappé au lecteur attentif l'utilisation de la commande **wr** dans les captures ci-dessus. C'est un « truc » de vieil administrateur las de devoir confirmer quand il utilise la commande normale **copy run start**. Il se trouve que CISCO intègre de façon régulière de nouvelles commandes à l'IOS mais répugne à supprimer les anciennes commandes, on l'espère par souci de ne pas désorienter ses utilisateurs clients. Comme d'autres, l'ancienne commande qui permettait de sauvegarder la configuration courante fonctionne toujours :

```
RTR7C#write ?
core      Write Core File
erase     Erase NV memory
memory    Write to NV memory
network   Write to network TFTP server
terminal  Write to terminal
<cr>

RTR7C#write memory
```

On peut abréger cette commande en **wr** et elle ne demande pas de confirmation. Attention, il faut être sûr de son fait mais à ce stade d'avancement, le lecteur est très certainement suffisamment aguerri pour profiter de cette commande et des précieux instants qu'elle permet d'économiser. D'autant que la commande normale **copy run start** n'est pas non plus sans danger. Imaginez que dans la précipitation, l'administrateur tape **copy run *satrt***. Ceci correspond à la création d'un nouveau fichier « ***satrt*** » et l'interface demande confirmation à l'administrateur avant d'effacer le contenu de la mémoire Flash ! Il suffit que l'administrateur confirme hâtivement sans lire réellement les messages pour que le drame arrive.

La commande **write memory** est une ancienne commande qui équivaut à la commande **copy run start** à ceci près qu'elle ne demande pas de confirmation. Cette commande peut être abrégée par **wr**.

Mais revenons à notre problématique de départ, celle de l'agrégation automatique. Le réglage du paramètre d'agrégation, automatique ou pas, peut être vérifié à l'aide d'une commande **show ip protocols**. Exemple sur RTR7C (extrait) :

```
RTR7C#sh ip protocols
Routing Protocol is "rip"
.........
     Automatic network summarization is not in effect
.........
RTR7C#
```

Une commande **show ip route** sur RTR7A ne rassure pas immédiatement l'administrateur :

```
RTR7A#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
Gateway of last resort is not set

     172.16.0.0/16 is variably subnetted, 11 subnets, 6 masks
R       172.16.0.184/30 [120/1] via 192.168.0.254, 00:00:23, Serial0/1
R       172.16.0.176/29 [120/2] via 192.168.0.254, 00:00:23, Serial0/1
R       172.16.0.160/28 [120/2] via 192.168.0.254, 00:00:23, Serial0/1
R       172.16.0.128/27 [120/1] via 192.168.0.254, 00:00:23, Serial0/1
R       172.16.0.208/28 [120/1] via 192.168.0.246, 00:00:24, Serial0/2
R       172.16.0.192/28 [120/1] via 192.168.0.246, 00:00:24, Serial0/2
R       172.16.0.0/16 [120/1] via 192.168.0.254, 00:03:08, Serial0/1
R       172.16.0.0/26 [120/1] via 192.168.0.249, 00:00:10, Serial0/0
```

```
R       172.16.0.96/30 [120/1] via 192.168.0.249, 00:00:10, Serial0/0
R       172.16.0.80/28 [120/2] via 192.168.0.249, 00:00:10, Serial0/0
R       172.16.0.64/28 [120/2] via 192.168.0.249, 00:00:10, Serial0/0
     192.168.0.0/30 is subnetted, 3 subnets
C       192.168.0.248 is directly connected, Serial0/0
C       192.168.0.252 is directly connected, Serial0/1
C       192.168.0.244 is directly connected, Serial0/2
```

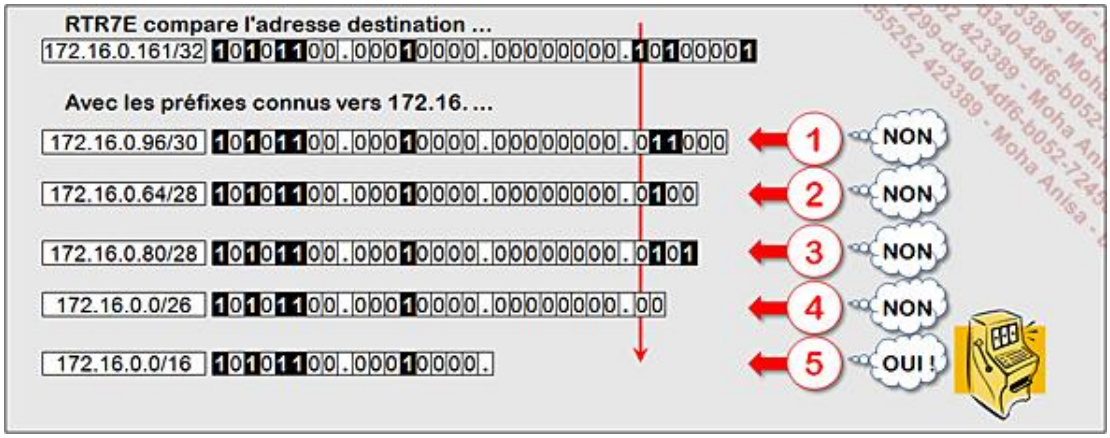
En effet, le domaine couvert par 172.16.0.0 ne devrait comporter que 10 sous-réseaux, la commande nous avertit que RTR7A en connaît 11. Cherchez l'intrus... Vous avez trouvé ? Surlignez-le. Il s'agit d'une route agrégée (la 7e route) qui n'a pas été effacée parce que son temporisateur d'effacement n'a pas encore expiré. Observez que la dernière mise à jour reçue pour cette route remonte à plus de trois minutes. La même commande provoquée quelques instants plus tard :

```
RTR7A#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
Gateway of last resort is not set

     172.16.0.0/16 is variably subnetted, 10 subnets, 5 masks
R       172.16.0.184/30 [120/1] via 192.168.0.254, 00:00:24, Serial0/1
R       172.16.0.176/29 [120/2] via 192.168.0.254, 00:00:24, Serial0/1
R       172.16.0.160/28 [120/2] via 192.168.0.254, 00:00:24, Serial0/1
R       172.16.0.128/27 [120/1] via 192.168.0.254, 00:00:24, Serial0/1
R       172.16.0.208/28 [120/1] via 192.168.0.246, 00:00:25, Serial0/2
R       172.16.0.192/28 [120/1] via 192.168.0.246, 00:00:25, Serial0/2
R       172.16.0.0/26 [120/1] via 192.168.0.249, 00:00:19, Serial0/0
R       172.16.0.96/30 [120/1] via 192.168.0.249, 00:00:19, Serial0/0
R       172.16.0.80/28 [120/2] via 192.168.0.249, 00:00:19, Serial0/0
R       172.16.0.64/28 [120/2] via 192.168.0.249, 00:00:19, Serial0/0
     192.168.0.0/30 is subnetted, 3 subnets
C       192.168.0.248 is directly connected, Serial0/0
C       192.168.0.252 is directly connected, Serial0/1
C       192.168.0.244 is directly connected, Serial0/2
RTR7A#
```

Tout est rentré dans l'ordre, l'administrateur est satisfait. Observez la table de routage de RTR7E :

```
RTR7E#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
Gateway of last resort is not set

     172.16.0.0/16 is variably subnetted, 5 subnets, 4 masks
R       172.16.0.0/16 [120/3] via 172.16.0.97, 00:00:02, Serial0/0
R       172.16.0.0/26 [120/1] via 172.16.0.97, 00:00:02, Serial0/0
C       172.16.0.96/30 is directly connected, Serial0/0
C       172.16.0.80/28 is directly connected, FastEthernet0/1
C       172.16.0.64/28 is directly connected, FastEthernet0/0
     192.168.0.0/30 is subnetted, 3 subnets
R       192.168.0.248 [120/1] via 172.16.0.97, 00:00:02, Serial0/0
R       192.168.0.252 [120/2] via 172.16.0.97, 00:00:02, Serial0/0
R       192.168.0.244 [120/2] via 172.16.0.97, 00:00:02, Serial0/0
RTR7E#
```

Ce routeur dispose encore d'une route agrégée vers 172.16.0.0 via 172.16.0.97. Ceci s'explique par le fait que la commande **no auto-summary** n'a pas été utilisée sur RTR7A. De ce fait, RTR7A agrège 172.16.0.0 et annonce le réseau majeur dans son entier sur toutes ses interfaces. Mais cela n'est absolument pas un problème car on se souvient (relire si nécessaire la section Routage sans classe du chapitre précédent) que la commande **ip classless** est activée par défaut. Cette commande fait adopter au processus de routage l'algorithme de recherche de correspondance de préfixe la plus longue et notre contexte offre le moyen de vérifier son utilité et sa pertinence. Imaginons un paquet reçu par RTR7E et destiné à 172.16.0.161 :



La seule correspondance de préfixe présente dans la table est 172.16.0.0/16 et RTR7E remet le paquet à 172.16.0.97 via S0/0 :

```
RTR7E#ping 172.16.0.161
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.0.161, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 120/142/184 ms
RTR7E#
```

Imaginons un second paquet destiné à 172.16.0.1 :

La route vers 172.16.0.0/16 donnait une correspondance mais elle n'est pas examinée puisqu'une correspondance plus longue est trouvée dès le 4e test. Le paquet est remis à 172.16.0.97 via S0/0. Le routeur RTR7C le reçoit et utilise le même algorithme. Sa table de routage est :

```
RTR7C#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
Gateway of last resort is not set

     172.16.0.0/16 is variably subnetted, 5 subnets, 4 masks
R       172.16.0.0/16 [120/2] via 192.168.0.250, 00:00:25, Serial0/0
C       172.16.0.0/26 is directly connected, FastEthernet0/0
C       172.16.0.96/30 is directly connected, Serial0/1
R       172.16.0.80/28 [120/1] via 172.16.0.98, 00:00:23, Serial0/1
R       172.16.0.64/28 [120/1] via 172.16.0.98, 00:00:23, Serial0/1
     192.168.0.0/30 is subnetted, 3 subnets
C       192.168.0.248 is directly connected, Serial0/0
R       192.168.0.252 [120/1] via 192.168.0.250, 00:00:25, Serial0/0
R       192.168.0.244 [120/1] via 192.168.0.250, 00:00:25, Serial0/0
RTR7C#
```

RTR7C connaît les mêmes préfixes que RTR7E et trouve donc la même plus longue correspondance avec 172.16.0.0/26, le paquet est remis à l'interface F0/0. Une commande ping confirme le bon acheminement :

```
RTR7E#ping 172.16.0.1
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 8/21/28 ms
RTR7E#
```

b. Activation de l'authentification

L'IOS offre le moyen d'authentifier les messages de façon très souple en permettant de définir non pas une mais des ensembles de clés. Un ensemble peut contenir une ou plusieurs clés. Chaque clé peut être périssable ou non. Ces clés peuvent être émises en tant que mots de passe en texte clair ou servir à générer une signature MD5. Procédons par étapes.

Scénario 1 - Objectif :

- Authentifier les échanges de mises à jour RIP entre les deux routeurs RTR7A et RTR7C par un mot de passe simple. Le mot de passe est invariable, l'administrateur a choisi « Inabottle ».

Étape 1 : créer un ensemble de clés

La commande **key chain** permet de créer un ensemble de clés. Sa syntaxe est la suivante :

```
Router(config)#key chain name-of-chain
```

... dont l'argument est :

name-of-chain

Nom attribué à l'ensemble de clés. L'ensemble doit comporter au moins une clé et peut comporter jusqu'à 2 147 483 647 clés.

Appliquée au cas présent :

```
RTR7C(config)#key ?
chain       Key-chain management
config-key  Set a private configuration key

RTR7C(config)#key chain ?
WORD  Key-chain name

RTR7C(config)#key chain turing
RTR7C(config-keychain)#
```

Le nom attribué à l'ensemble de clés, « Turing » dans le cas présent, n'a qu'une portée locale. Il n'est donc pas indispensable (mais pas interdit) de nommer de façon identique les ensembles de clés de deux routeurs qui doivent authentifier leurs échanges.

Étape 2 : placer au moins une clé dans l'ensemble de clés

La commande **key** crée une clé dans l'ensemble de clés et doit être utilisée autant de fois qu'il y a de clés à créer dans l'ensemble. Sa syntaxe est la suivante :

```
Router(config-keychain)#key key-id
Router(config-keychain-key)#
```

... dont l'argument est :

key-id

Numéro qui identifie la clé en cours de création, doit appartenir à l'espace [0 - 2 147 483 647]. Les identifiants de clés n'ont pas besoin d'être consécutifs.

La commande **key-string** crée la chaîne de caractères qui constitue la clé. Sa syntaxe est la suivante :

```
Router(config-keychain-key)# key-string text
```

... dont l'argument est :

text

Chaîne à utiliser dans les messages à authentifier, peut contenir de 1 à 80 caractères alphanumériques : minuscules, majuscules ou chiffres. Le premier caractère ne peut être un chiffre. L'administrateur qui le souhaite peut protéger les clés des regards indiscrets à l'aide de la commande **service password-encryption**.

Appliquées au cas présent :

```
RTR7C(config-keychain)#key ?
<0-2147483647>  Key identifier

RTR7C(config-keychain)#key 1
RTR7C(config-keychain-key)#?
Key-chain key configuration commands:
  accept-lifetime  Set accept lifetime of key
default          Set a command to its defaults
exit             Exit from key-chain key configuration mode
key-string       Set key string
no               Negate a command or set its defaults
send-lifetime    Set send lifetime of key
RTR7C(config-keychain-key)#key-string Inabottle
```

Pour ce cahier des charges, inutile d'en faire plus. D'autres contextes nécessiteront d'étoffer cette étape pour placer plusieurs clés et régler leur espérance de vie.

Étape 3 : appliquer l'authentification à une interface

La commande **ip rip authentication key-chain** en mode de configuration d'interface applique l'authentification et spécifie quel ensemble de clés doit être utilisé. Appliquée au cas présent :

```
RTR7C(config-keychain-key)#int s0/0
RTR7C(config-if)#ip rip authentication ?
  key-chain  Authentication key-chain
  mode       Authentication mode
RTR7C(config-if)#ip rip authentication key-chain ?
  LINE       name of key-chain
RTR7C(config-if)#ip rip authentication key-chain turing
```

Étape 4 : choisir le mode d'authentification

La commande **ip rip authentication mode** permet de choisir l'authentification par mot de passe simple ou l'authentification forte par signature MD5. Sa syntaxe est la suivante :

```
Router(config-if)#ip rip authentication mode {text|md5}
```

Appliquée au cas présent :

```
RTR7C(config-if)#ip rip authentication mode ?
  md5   Keyed message digest
  text  Clear text authentication
RTR7C(config-if)#ip rip authentication mode text
RTR7C(config-if)#^Z
RTR7C#
```

Étape 5 : répéter l'ensemble des opérations sur le second routeur concerné

Dans le cas présent, il s'agit de RTR7A :

```
RTR7A(config)#key chain Turing
RTR7A(config-keychain-key)#key-string Inabottle
RTR7A(config-keychain-key)#int s0/0
RTR7A(config-if)#ip rip authentication key-chain Turing
RTR7A(config-if)#ip rip authentication mode text
RTR7A(config-if)#^Z
RTR7A#
```

Étape 6 : recette

Une commande **show key chain** est utile pour vérifier la configuration effectuée. Sur RTR7C :

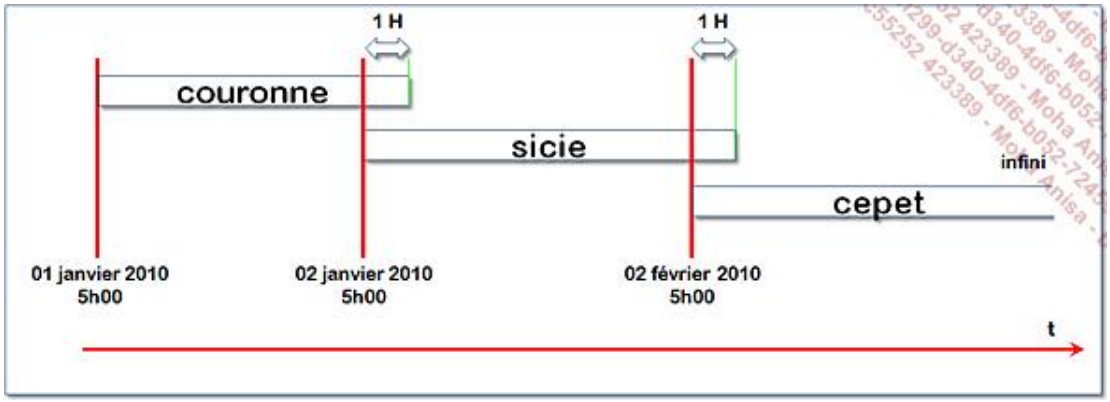
```
RTR7C#show key chain
Key-chain Turing:
  key 1 -- text "Inabottle"
      accept lifetime (always valid) - (always valid) [valid now]
      send lifetime (always valid) - (always valid) [valid now]
```

Une commande **debug ip rip** confirme que les échanges entre RTR7A et RTR7C s'effectuent de manière authentifiée :

```
RTR7C#debug ip rip
RIP protocol debugging is on
RTR7C#
00:08:04: RIP: received packet with text authentication Inabottle
00:08:04: RIP: received v2 update from 172.16.0.250 on Serial0/0
00:08:04:      172.16.0.128/27 via 0.0.0.0 in 2 hops
00:08:04:      172.16.0.160/28 via 0.0.0.0 in 3 hops
00:08:04:      172.16.0.176/29 via 0.0.0.0 in 3 hops
00:08:04:      172.16.0.184/30 via 0.0.0.0 in 2 hops
RTR7C#u all
```

Scénario 2 - Objectif :

- Authentifier les échanges de mises à jour RIP entre les deux routeurs RTR7A et RTR7C par une signature MD5. Les mots de passe utilisés doivent changer selon le calendrier suivant :



Étape 1 : modifier l'ensemble de clés Turing

Nous aurons besoin pour ce faire des deux commandes **accept-lifetime** et **send-lifetime**. La commande **accept-lifetime** spécifie la durée de validité des clés reçues. La commande **send-lifetime** spécifie la durée d'utilisation des clés du côté émission des mises à jour authentifiées. Leur syntaxe est la suivante :

```
RTR7C(config-keychain-key)#accept-lifetime start-time {infinite | end-time |
durationseconds}
RTR7C(config-keychain-key)#send-lifetime start-time {infinite | end-time |
durationseconds
```

... dont les arguments sont :

start-time

La syntaxe est indifféremment :

hh:mm:ss Month date year

hh:mm:ss date Month year

Month → les trois premières lettres du mois exprimé en anglais.

Date → de 1 à 31.

Year → année exprimée sur 4 chiffres.

infinite

La clé est valide indéfiniment à partir de l'instant défini par start-time.

end-time

La clé est valide entre les deux instants définis par start-time et end-time. Syntaxe identique à celle de start-time. La valeur **infinite** est adoptée par défaut.

duration

La clé est valide pendant **duration** secondes à partir de l'instant défini par start-time.

Appliquée au scénario présent :

```
RTR7C(config)#key chain Turing
RTR7C(config-keychain)#key 1
RTR7C(config-keychain-key)#key-string couronne
RTR7C(config-keychain-key)#accept-lifetime 05:00:00 jan 01 2010 duration 90000
RTR7C(config-keychain-key)#send-lifetime 05:00:00 jan 01 2010 duration 90000
RTR7C(config-keychain-key)#key 2
RTR7C(config-keychain-key)#key-string sicie
RTR7C(config-keychain-key)#accept-lifetime 05:00:00 jan 02 2010 06:00:00 feb 022010
RTR7C(config-keychain-key)#send-lifetime 05:00:00 jan 02 2010 06:00:00 feb 02 2010
RTR7C(config-keychain-key)#key 3
RTR7C(config-keychain-key)#key-string cepet
RTR7C(config-keychain-key)#accept-lifetime 05:00:00 feb 02 2010 infinite
RTR7C(config-keychain-key)#send-lifetime 05:00:00 feb 02 2010 infinite
RTR7C(config-keychain-key)#^Z
RTR7C#
```

Observez que l'administrateur a pris la précaution de faire coexister deux clés pendant un court moment (1 heure) afin de pallier le fait que les deux routeurs pourraient ne pas partager la même heure (90000 secondes correspondent à 25 heures).

Étape 2 : passer en MD5

```
RTR7C#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
RTR7C(config)#int s0/0
RTR7C(config-if)#ip rip authentication mode md5
RTR7C(config-if)#^Z
RTR7C#
```

Étape 2 : recette

Avant toute chose, assurons-nous que les deux routeurs sont à l'heure. En production, l'idéal serait évidemment de confier la mise à l'heure des équipements à un serveur NTP. Pour notre modeste mise en situation, une commande **clock set** fera l'affaire, à répéter sur RTR7C :

```
RTR7A#clock set 21:00:00 3 may 2010
```

Une commande **debug ip rip** confirme l'authentification par signature MD5 des mises à jour échangées :

```
RTR7A#debug ip rip
RIP protocol debugging is on
00:24:18: RIP: received packet with MD5 authentication
00:24:18: RIP: received v2 update from 192.168.0.249 on Serial0/0
00:24:18:      172.16.0.0/26 via 0.0.0.0 in 1 hops
.........
```

```
RTR7A#u all
All possible debugging has been turned off
```

Une commande **show key chain** montre les clés contenues dans l'ensemble de clés ainsi que la clé en cours d'utilisation :

```
RTR7A#sh key chain
Key-chain Turing:
key 1 -- text "couronne"
accept lifetime (05:00:00 UTC Jan 1 2010) - (90000 seconds)
send lifetime (05:00:00 UTC Jan 1 2010) - (90000 seconds)
key 2 -- text "sicie"
accept lifetime (05:00:00 UTC Jan 2 2010) - (06:00:00 UTC Feb 2 2010)
send lifetime (05:00:00 UTC Jan 2 2010) - (06:00:00 UTC Feb 2 2010)
key 3 -- text "cepet"
accept lifetime (05:00:00 UTC Feb 2 2010) - (infinite) [valid now]
send lifetime (05:00:00 UTC Feb 2 2010) - (infinite) [valid now]
RTR7A#
```

c. Réglage de la compatibilité

L'IOS prévoit deux commandes pour respecter les préconisations du RFC en matière de compatibilité avec RIPv1.

Appliquée à une interface, la commande **ip rip send version** permet de générer des mises à jour selon le format RIPv1 ou RIPv2 ou selon les deux formats. Sa syntaxe est la suivante :

```
Router(config-if)#ip rip send version [1] [2]
```

Par défaut, un processus RIPv2 activé ne génère que des mises à jour version 2. On peut outrepasser ce comportement afin d'adapter le routeur à une partie de son environnement. Exemples :

- Sur l'une de ses interfaces, un routeur RIPv2 est connecté à une machine ne comprenant exclusivement que les messages RIPv1 → entrez la commande **ip rip send version 1**.

- Sur l'une de ses interfaces, un routeur RIPv2 est connecté à plusieurs équipements dont une partie peut exploiter des mises à jour RIPv2, d'autres ne comprennent que les mises à jour RIPv1 → entrez la commande **ip rip send version 1 2**. Attention, le trafic d'acheminement sur l'interface considérée est doublé car les annonces sont répétées deux fois, une par format.

Appliquée à une interface, la commande **ip rip receive version** permet de ne prendre en compte que l'un des formats possibles de mises à jour RIP. Sa syntaxe est la suivante :

```
Router(config-if)#ip rip receive version [1] [2]
```

Par défaut, un processus RIPv2 activé n'exploite exclusivement que les mises à jour RIPv2. On peut outrepasser ce comportement afin de faire en sorte que le processus RIP accepte également des mises à jour issues de machines exclusivement RIPv1 à l'aide de la commande **ip rip receive 1 2**. Autre possibilité : on pourrait également contraindre le processus à ignorer les mises à jour RIPv2 pour n'accepter que les mises à jour issues de RIPv1 à l'aide d'une commande **ip rip receive 1**, mais cela n'a pas beaucoup de sens.

In fine, manipuler ces commandes entraîne le besoin de vérifier le comportement effectif du processus RIP vis-à-vis des mises à jour, ce que l'administrateur obtient à l'aide de la commande **show ip protocols** (recherchez si besoin des captures de cette commande dans ce chapitre).

4. Résumé

a. Les caractéristiques à retenir

RIPv2 est un protocole de routage sans classe. Il doit cette faculté au fait qu'il associe l'information de route et son masque. Mais son comportement dans l'agrégation rappelle l'époque avec classe.

Un routeur RIPv2 agrège ses annonces aux frontières des réseaux majeurs comme le faisait RIPv1. L'agrégation annoncée est le réseau majeur. C'est donc une agrégation /8, /16 ou /24.

Ainsi, la granularité de l'agrégation est pauvre. Dans le chapitre suivant dédié à EIGRP, nous verrons que l'administrateur peut régler manuellement le préfixe agrégé et donc sa longueur, ce qui lui permet d'ajuster beaucoup plus finement l'agrégation à un découpage VLSM complexe (si tant est qu'il soit souhaitable d'établir des découpages VLSM complexes, mais c'est un autre débat).

La mise en service de RIPv2 n'est guère plus difficile que celle de RIPv1. Quand des erreurs surviennent, elles sont la plupart du temps dues au non-respect des contraintes imposées par le protocole, essentiellement.

Un routeur qui annonce un réseau majeur doit être la route unique vers ce réseau. Quand ce n'est pas le cas, il faut contraindre le routeur à annoncer les sous-réseaux qu'il connaît du réseau majeur à l'aide de la commande **no auto-summary**.

Donc, si des routes manquent ou si des routes sont incohérentes, réexaminez avec attention votre plan d'adressage et votre découpage en sous-réseaux et identifiez les routeurs frontières.

b. Les commandes importantes

Commande	Mode	Description
router rip	Configuration globale	Active le processus RIP.
network *@IP_réseau*	Configuration de routeur	L'adresse indiquée doit inclure les adresses des interfaces qui participent au protocole.
version	Configuration de routeur	Utile pour passer en version 2.
neighbor *@IP_voisin*	Configuration de routeur	Active l'envoi de mises à jour unicast vers ce voisin.
passive-interface *interface-typeinterface-number*	Configuration de routeur	Cesse l'envoi de mises à jour sur l'interface spécifiée.
ip classless	Configuration globale	Active l'algorithme de recherche de correspondance de préfixe la plus longue (*Longest Match based Forwarding Algorithm*) dans le processus de recherche de route. C'est la commande par défaut.
ip subnet-zéro	Configuration globale	Autorise les adresses de sous-réseaux exclusivement composées de 0.
ip rip authentication key-chain *name_of_key-chain*	Configuration d'interface	Active l'authentification des mises à jour RIP émises depuis cette interface ou reçues sur cette interface.
ip rip authentication mode {**text**|**md5**}	Configuration d'interface	Authentification par mot de passe simple ou par signature MD5.
key *number*	Configuration d'ensemble de clés	Spécifie une clé dans l'ensemble de clés.
key-string *text*	Configuration de clé	Attribue une chaîne de caractères à la clé qui servira soit de mot de passe, soit de clé pour élaborer une signature MD5 de la mise à jour.
accept-lifetime *start-time* {**infinite** | *end-time* | **duration** *seconds*}	Configuration de clé	Spécifie un laps de temps pendant lequel la clé est valide en réception.
send-lifetime *start-time* {**infinite** | *end-time* | **duration** *seconds*}	Configuration de clé	Spécifie un laps de temps pendant lequel la clé peut être utilisée dans les mises à jour émises.

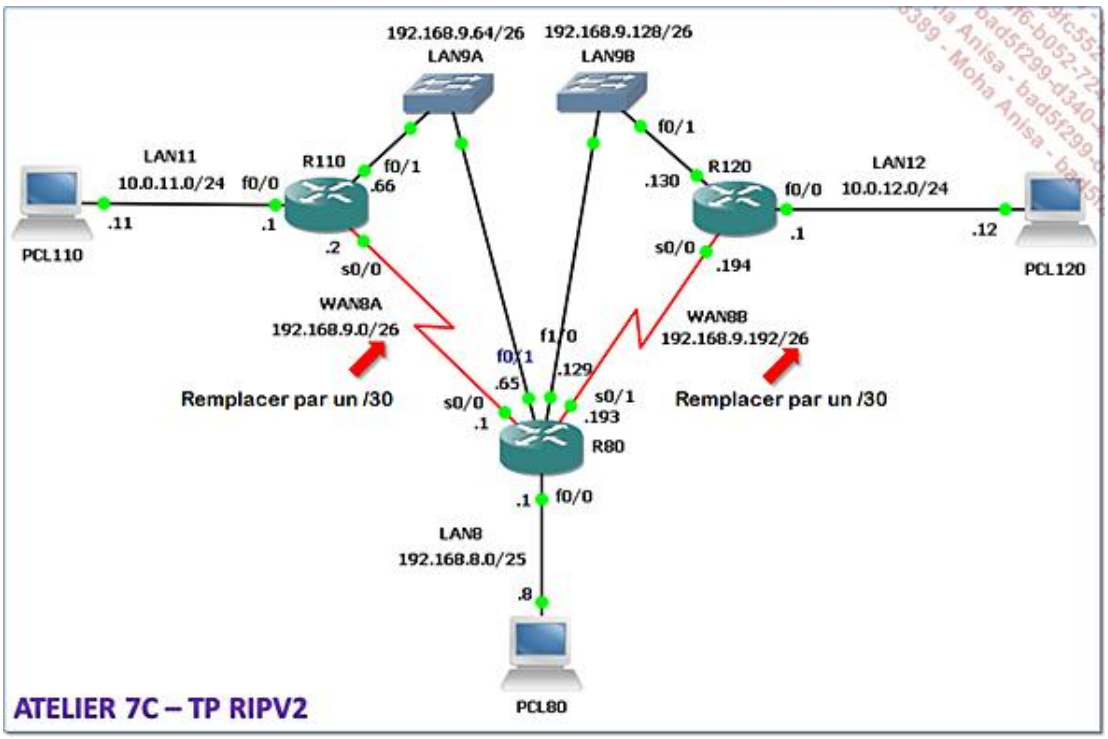
ip receive version [**1**] [**2**]	Configuration d'interface	Spécifie les formats de mises à jour acceptées sur cette interface.
ip send version [**1**] [**2**]	Configuration d'interface	Spécifie les formats de mises à jour générées sur cette interface.
show ip route rip	Mode utilisateur	Affiche l'ensemble des routes issues de RIP.
show ip protocols	Mode utilisateur	Affiche les paramètres et l'état actuel du protocole de routage activé sur le routeur.
debug ip rip	Mode privilégié	Affiche en temps réel le trafic d'acheminement RIP.
auto-summary	Configuration de routeur	Active ou désactive l'agrégation automatique à la frontière d'un réseau majeur.
timers basic *update invalid holddown flush*	Configuration de routeur	Ajuste les valeurs initiales des temporisateurs.
output-delay *delay*	Configuration de routeur	Introduit un espace de temps minimal de 8 à 50 millisecondes entre deux émissions de mises à jour RIP, utile pour qu'un routeur lent puisse s'accommoder des mises à jour issues d'un routeur puissant.

5. TP : Mise en œuvre d'une configuration RIPv2

Bien sûr, rien n'interdit au lecteur de reproduire l'ensemble des topologies de ce chapitre dans GNS3 mais il faut bien avouer que faire fonctionner les six routeurs des ateliers 7A et 7B n'est pas une mince affaire, l'ensemble a manqué cruellement de stabilité sur la machine de l'auteur, malgré ses efforts.

Plus modestement, proposons-nous d'utiliser la topologie ci-dessous qui avait servi à illustrer les limites d'un protocole de routage avec classe. Le domaine couvert par le réseau 10 est scindé en deux parties. Par ailleurs, le masque adopté pour découper le réseau 192.168.9.0 est unique ce qui conduit à un important gaspillage d'adresses sur les liens « serial ». L'objectif est donc double :

- Remplacer les sous-réseaux /26 des liens « serial » par des sous-réseaux /30 du même réseau majeur 192.168.9.0.
- Appliquer le protocole RIPv2.



Une solution est proposée au chapitre Ateliers et exercices corrigés. De plus, le lecteur qui voudrait reproduire cette topologie dans un contexte émulé GNS trouvera une aide précieuse au chapitre Annexes - Définition d"un contexte d'atelier.

Contexte

1. Les objectifs de CISCO

Cisco revendique pour son protocole de routage un temps de convergence au moins aussi bon que celui d'OSPF, tout en affirmant le surpasser en termes de consommation de ressources machines. Avant de découvrir les réelles qualités de ce protocole, constatons qu'il est inutile d'ouvrir une polémique car le problème est ailleurs : EIGRP est une technologie qui appartient à CISCO, ce qui limite son déploiement à des réseaux 100 % CISCO. La lettre « E » d'EIGRP signifie « *Enhanced* » et rappelle qu'EIGRP a pour parent le protocole IGRP. Il se trouve qu'IGRP est un protocole avec classe et que l'abandon des classes imposait à CISCO de le réviser.

Difficile de classer EIGRP dans l'une des deux familles DV ou états de liens, du fait même de la communication CISCO sur ce sujet, communication qui a manqué de constance. CISCO a tantôt présenté EIGRP comme un protocole DV amélioré, tantôt comme un protocole hybride DV - états de liens. Pour résumer le sentiment généralement accepté, EIGRP est un protocole DV qui se comporte comme un protocole à états de liens.

Rappelons qu'un routeur sous protocole DV partage tout ce qu'il sait mais uniquement avec ses voisins directement connectés. À l'opposé, un routeur sous protocole à états de liens annonce peu l'état de ses liens, mais partage cette information avec l'ensemble des routeurs du domaine.

La description du protocole RIP a permis de mesurer les affres dont souffrent les protocoles DV et leur propension à créer des boucles de routage. Ces défauts intrinsèques sont combattus à coups d'artifices type partage d'horizon, empoisonnement de routes et temporisateurs de retenue. Un routeur DV n'annonce pas les routes qu'il reçoit mais les routes contenues dans sa table. Une annonce reçue est passée au crible de l'algorithme de routage avant d'être éventuellement installée dans la table, le temps nécessaire diffère d'autant la propagation de l'information de topologie dans le réseau, ceci rend par nature un protocole DV lent à converger.

Par ailleurs, et puisqu'un protocole DV annonce des routes, le changement d'état d'un seul lien du réseau peut avoir des conséquences sur de nombreuses routes ce qui peut se traduire par un important trafic d'acheminement quand l'évènement se produit.

Inventorier tous ces défauts renforce instantanément l'intérêt pour les protocoles à états de liens. Il est vrai qu'ils sont peu susceptibles de provoquer des boucles de routage. Il est vrai également qu'une annonce de lien reçue par un routeur peut être transmise au routeur suivant sans délai, de grands réseaux peuvent ainsi converger rapidement. Enfin, un changement d'état de lien provoque l'annonce par le routeur concerné du nouvel état du lien, c'est mieux qu'une tempête de mises à jour de routes concernées par le lien.

Gardons-nous pourtant de parer les protocoles à états de liens de toutes les vertus. Car maintenir les bases de données nécessaires au protocole et dérouler régulièrement l'algorithme du plus court chemin (patientez) sont très consommateurs de ressources machine, mémoire et CPU. À moins que ce défaut ne devienne marginal du fait de l'évolution de la puissance et du coût des machines.

Tout routeur calcule, un routeur DV le fait avant d'envoyer des mises à jour à ses voisins, un routeur à états de liens le fait après avoir construit sa base de données topologique (patientez). Mais dans les deux cas, chaque routeur d'un domaine mène l'ensemble de ses calculs individuellement en utilisant l'information dont il dispose. C'est ici qu'EIGRP se distingue car avec l'algorithme utilisé, tout se passe comme si EIGRP répartissait ses calculs sur l'ensemble des routeurs, avec à la clé une consommation parcimonieuse des ressources réseau et machine, une convergence rapide (moins de cinq secondes selon CISCO dans la plupart des cas) et une topologie exempte de boucles.

2. Caractéristiques clés

EIGRP est un protocole de routage sans classe, c'est même la caractéristique essentielle qui a justifié son développement à partir d'IGRP. Tout comme RIPv2, chaque entrée de route associe à l'adresse réseau l'information de masque. VLSM peut être utilisé sans retenue, qu'il s'agisse de diviser ou d'agréger des espaces d'adresses.

Comme l'implémentation CISCO de RIPv2, EIGRP supporte l'authentification forte par signature MD5 de ses échanges. Dernière caractéristique qui fut importante par le passé, qui ne l'est plus aujourd'hui, EIGRP est capable autant de router le protocole IP que les protocoles IPX et Appletalk.

Quelques autres caractéristiques :

- Les routeurs découvrent leurs voisins puis entretiennent les relations de voisinage à l'aide de messages Hello émis de façon périodique.
- La métrique est composite et peut faire intervenir la bande passante, le délai, la fiabilité et la charge des liens.
- Les mises à jour émises par un routeur sont non périodiques, partielles, ciblées et fiables :

- Non périodiques : les mises à jour sont événementielles, c'est-à-dire qu'elles n'interviennent que lors d'un changement de métrique ou de topologie.

- Partielles : les mises à jour ne contiennent que les routes qui ont changé et non tout le contenu de la table de routage.

- Ciblées : les mises à jour envoyées à un voisin le sont parce que les changements relatés par la mise à jour affectent ce voisin.

- Fiables : les mises à jour sont transportées à l'aide du protocole RTP (*Reliable Transport Protocol*), protocole conçu par CISCO et qui garantit la remise, l'intégrité et le séquencement des paquets.

- EIGRP est capable de partage de charge à coût égal mais aussi, et c'est le seul IGP à revendiquer cette faculté, capable de partage de charge à coût inégal au prix d'une légère configuration.

Détail du protocole

Comme nous pourrons le vérifier en abordant OSPF un peu plus tard, EIGRP utilise une séquence d'opérations très analogue à celle du protocole à états de liens :

1. Un routeur EIGRP cherche à découvrir ses voisins et à faire connaître sa propre existence. Pour ce faire, EIGRP génère des messages Hello de façon régulière et exploite les messages éventuellement reçus afin de déterminer qui sont ses voisins.

2. L'établissement d'une relation de voisinage s'accompagne d'échanges de mises à jour de topologie permettant à deux voisins d'égaliser leur perception ou leur connaissance du réseau.

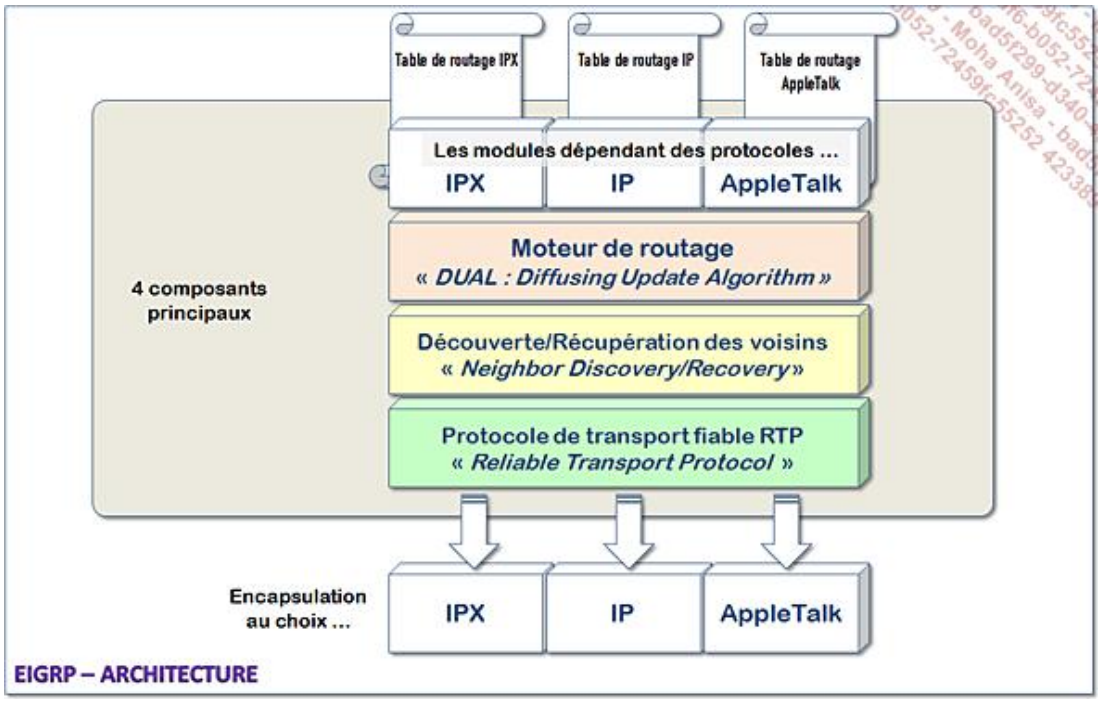
3. Chaque routeur exploite sa table de topologie EIGRP pour construire sa table de routage.

Ainsi, comme OSPF, EIGRP entretient plusieurs bases de données :

- La table des voisins consultable à l'aide d'une commande **show ip eigrp neighbor**.
- La base de données topologique consultable via **show ip eigrp topology**.
- La table de routage qui regroupe toutes les routes qu'elles soient issues d'EIGRP ou pas, consultable à l'aide de la commande **show ip route** ou mieux **show ip route eigrp**.

Le problème s'aggravera encore avec OSPF mais il faut bien reconnaître que la complexité d'EIGRP est telle qu'on ne sait pas par où commencer. Il est quasi impossible de bâtir un exposé parfaitement linéaire et séquencé, c'est-à-dire un exposé dans lequel le lecteur n'aurait pas à faire des allers et retours. L'auteur réclame donc l'indulgence du lecteur face aux probables nombreuses références-avant. Une référence-avant clairement identifiée l'est par la mention « (patientez) ».

1. Architecture



Cette illustration n'a absolument rien d'inédit, on la retrouve absolument partout. Il faudra pourtant bien un jour admettre que seul subsiste IP et dépoussiérer un peu. S'il n'y avait la crainte d'une question de certification portant sur cette architecture...

2. Notion de successeur faisable

EIGRP entretient une table intermédiaire dite table de topologie dont il déduit sa table de routage. Les mises à jour

envoyées vers un voisin comportent des informations issues non pas de la table de routage mais de la table de topologie. Chaque réseau non encore connu et annoncé par un voisin est ajouté à la table de topologie. Si le réseau annoncé est déjà connu via un ou plusieurs autres voisins, le voisin qui annonce est ajouté à la collection de voisins qui ont également annoncé ce réseau.

Un réseau (une destination, une entrée de la table de topologie) est déplacé de la table de topologie vers la table de routage quand il existe un successeur potentiel (ou successeur faisable) pour ce réseau. Dans la collection de voisins ayant annoncé le réseau, les successeurs potentiels sont ceux qui ont annoncé une métrique inférieure à la métrique en cours dans la table de routage. Le routeur considère les successeurs potentiels comme des voisins situés en aval vis-à-vis du réseau objet. Autrement dit, le routeur doit être en amont de tous les successeurs potentiels pour une route donnée.

Cette première explication n'est certainement pas suffisante, d'autres détails sont fournis dans la section dédiée à DUAL.

3. Le protocole de transport RTP

Le protocole RTP est un protocole développé par CISCO pour assurer un transport fiable du trafic d'acheminement. Pourquoi ne pas avoir utilisé TCP ? Pour au moins deux raisons. La première est que RTP devait pouvoir s'appuyer indifféremment sur IP, IPX ou AppleTalk. Il fallait donc concevoir un protocole indépendant. La seconde raison est que le fonctionnement de TCP en mode connecté qui lui interdit la diffusion. RTP au contraire peut assurer un transport fiable mais sans connexion. Si on oublie IPX et AppleTalk, RTP s'encapsule dans IP et est identifié par le numéro de protocole 88. RTP est capable de multidiffusion, l'adresse de destination est dans ce cas 224.0.0.10, adresse de classe D qui identifie le groupe des routeurs EIGRP. En final, RTP offre un service à la carte selon les exigences du transport, unicast ou multicast, avec ou sans garantie :

- Garantie de remise : un routeur qui reçoit un paquet RTP multicast doit acquitter (c'est-à-dire émettre un paquet accusé de réception), il le fait à l'aide d'un paquet unicast.

- Garantie de séquencement : chaque paquet RTP embarque deux numéros de séquence. Le routeur émetteur d'un message positionne le numéro de séquence du paquet. Ce numéro est incrémenté à chaque nouveau paquet émis. Le second numéro sert l'autre sens de flux et permet au routeur émetteur de faire savoir à son correspondant quel est le numéro de séquence du dernier paquet reçu.

- Paquets sans garantie de remise : RTP est également capable de transporter des paquets sans garantie de remise (c'est un peu comme si RTP était tantôt capable de se comporter comme TCP, tantôt comme UDP). Ces paquets ne sont pas acquittés et par conséquent, n'incluent pas de numéros de séquence.

RTP transporte cinq types de messages :

- Les messages **Hello/Acks** : les messages Hello sont diffusés et destinés à découvrir les voisins puis entretenir les relations de voisinage. Ces messages ne sont pas acquittés. Un message Hello qui ne comporte pas de données fait office de message d'acquittement. Les acquittements sont toujours envoyés à des adresses unicast.

- Les messages de mises à jour (***Updates***) : ces messages portent l'information de topologie issue des tables topologiques. Quand un routeur se découvre un nouveau voisin, il lui envoie des messages de mises à jour afin que le voisin puisse construire sa table de topologie. Les messages sont dans ce cas envoyés à l'adresse unicast du voisin. Dans tous les autres cas, par exemple lorsque le coût d'un lien change, l'information intéresse tous les voisins et le routeur utilise des messages de mises à jour multidiffusés. Quand il transporte des messages de mises à jour, RTP fonctionne dans le mode fiable (les paquets doivent être acquittés).

- Les messages de requête et de réponse (***Queries/Replies***) : ces messages sont utilisés suite au passage à l'état actif d'une route. Une route (une entrée de la base de données topologique d'EIGRP) est dans l'un des deux états passif ou actif :

    - L'état passif signifie que le système est stable, le routeur n'entreprend ni ne participe à aucune recherche ni aucun calcul à son sujet (en anglais, c'est plus court, les documentations parlent de « *Diffusing computation* »). Tant qu'EIGRP dispose, pour une route donnée, d'un successeur potentiel ou successeur faisable (FS ou *FeasibleSuccessor*), une route ne doit pas quitter l'état Passif.

    - Si le dernier successeur potentiel disparaît, la route passe à l'état actif et un calcul diffusé est entrepris par le domaine EIGRP. Le routeur qui a fait le constat qu'il ne disposait plus de successeur potentiel diffuse une requête à l'ensemble de ses voisins. Chaque voisin interrogé a deux choix : répondre s'il dispose d'un successeur potentiel pour la route objet de la requête ou répondre par une requête qui signifie qu'à son tour, il initie une recherche. C'est bien en cela qu'il faut parler de processus ou de calcul diffusé.

- Tant que la route est dans l'état actif, un routeur ne doit pas modifier l'adresse du routeur de prochain saut utilisée pour acheminer les paquets destinés à cette route. Ce n'est qu'une fois toutes les réponses reçues pour une requête donnée que la route objet de la requête peut retourner à l'état passif et qu'un nouveau successeur peut être désigné.

- À moins qu'elle ne soit envoyée en réponse, une requête est toujours multi diffusée. Quand elle est utilisée en tant que réponse, elle est envoyée à l'adresse unicast du routeur à l'origine de la requête.

- Les messages de réponse sont toujours envoyés à l'adresse unicast du routeur à l'origine de la requête.

- Quand il transporte des requêtes ou des réponses, RTP fonctionne dans le mode fiable.

- Les messages de demande (***Request***) : ces messages sont utilisés quand un routeur souhaite obtenir de l'information spécifique d'un ou plusieurs de ses voisins, dans le cadre d'applications de type « *Route server* » (patientez). RTP transporte ces messages en mode non fiable.

4. Entretien des relations de voisinage

Cela a été dit, les mises à jour d'EIGRP ne sont pas périodiques. La conséquence est qu'il est impossible à un routeur de juger de l'état de ses voisins en se fondant sur la réception des paquets correspondants. Les concepteurs d'EIGRP ont dû imaginer un mécanisme de substitution dont l'objet est de découvrir puis surveiller les voisins. La solution retenue passe par l'émission périodique de messages Hello. Un message Hello est un message très court dont le principal objet est de rendre son émetteur visible des voisins.

Un routeur EIGRP A découvre un voisin B quand il en reçoit le premier message Hello issu d'un réseau directement connecté. Quand le voisin B est découvert, le processus EIGRP A crée une nouvelle entrée dans la table de voisinage et y place entre autres l'adresse du voisin B ainsi que l'interface via laquelle il a été découvert. Puis le routeur A sollicite DUAL afin d'envoyer au nouveau voisin B la totalité de sa table de topologie. Il le fait à l'aide de messages de mises à jour unicast dans lesquels le drapeau d'initialisation est positionné. Le nouveau voisin B qui reçoit un message de mise à jour avec ce drapeau positionné envoie au routeur A sa propre table de topologie.

EIGRP émet ses messages Hello vers l'adresse de multidiffusion 224.0.0.10. Un message Hello ne nécessite pas d'être acquitté. La période qui sépare deux messages Hello est 5 secondes sur les liens large bande mais 60 secondes sur les liens lents.

Les liens pour lesquels la période des messages Hello est 5 secondes :

- Réseaux à diffusion tels qu'Ethernet, Token Ring et FDDI.

- Liens WAN point à point, il peut s'agir de liens loués avec encapsulation PPP ou HDLC, de liens point à point Frame-Relay ou ATM.

- Liens multipoints large bande (supérieure au T1 - 1544 Kbps) telles que RNIS accès primaire et Frame-Relay.

Les liens pour lesquels la période des messages Hello est 60 secondes :

- Liens multipoints dont la bande est inférieure à celle du T1.

L'administrateur peut modifier la période qui sépare deux émissions de messages Hello à l'aide de la commande **ip hello-interval eigrp** en configuration d'interface :

```
R11(config)#int s0/0
R11(config-if)#ip hello-interval eigrp ?
   <1-65535>  Autonomous system number

R11(config-if)#ip hello-interval eigrp 64501 ?
   <1-65535>  Seconds between hello transmissions

R11(config-if)#ip hello-interval eigrp 64501 5
R11(config-if)#
```

Le message Hello comporte un champ Temps de maintien (*Hold Time*). Le routeur qui reçoit un message Hello arme un temporisateur avec cette valeur. Si le temporisateur expire avant d'avoir été réarmé par le message Hello suivant, le

voisin est jugé injoignable et le processus DUAL est informé de sa perte. Par défaut, le temps de maintien est égal à trois périodes du message Hello, soit 15 secondes ou 180 secondes selon les cas. À nouveau, l'administrateur peut modifier cette valeur à l'aide de la commande **ip hold-time eigrp** en configuration d'interface :

```
R11(config)#int s0/0
R11(config-if)#ip hold-time ?
   eigrp  Enhanced Interior Gateway Routing Protocol (EIGRP)

R11(config-if)#ip hold-time eigrp ?
   <1-65535>  Autonomous system number

R11(config-if)#ip hold-time eigrp 64501 ?
   <1-65535>  Seconds before neighbor is considered down

R11(config-if)#ip hold-time eigrp 64501 15
R11(config-if)#
```

Outre son existence, un routeur EIGRP annonce son propre temps de maintien dans ses messages Hello. Ceci évite que les temps *hello_interval* et *hold_time* doivent être réglés à l'identique sur l'ensemble des routeurs d'un domaine EIGRP. L'administrateur qui modifie le temps *hello_interval* doit également modifier le temps *hold_time* afin de le maintenir égal à trois périodes *hello_interval*.

Comparez le temps nécessaire à EIGRP pour détecter la disparition d'un voisin (15 secondes dans la plupart des cas) à celui exigé par RIP (Temporisateur d'invalidation, 180 secondes), c'est un premier facteur évidemment très favorable à l'obtention d'un temps de convergence court.

Ce n'est que depuis une époque récente (IOS 12.4) que CISCO a jugé bon de doter son IOS d'une commande permettant de découvrir quel est le temps **hello_interval** configuré sur un routeur inconnu. Et sauf erreur, l'affichage du réglage du temps **hold_time** manque toujours. Quelle que soit la version de l'IOS, l'administrateur peut malgré tout déduire ces valeurs de l'observation du temporisateur associé à chaque voisin chez un voisin du routeur inconnu. La valeur de ce temporisateur apparaît dans la colonne **Hold** d'une commande **show ip eigrp neighbor** :

```
R102#sh ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface   Hold Uptime   SRTT   RTO  Q  Seq Type
                                        (sec)         (ms)       Cnt Num
1   10.0.50.1               Se0/0        178 00:12:45  261  1566  0  19
2   10.1.100.101            Fa0/1         14 00:16:01 1113  5000  0  17
0   10.1.100.100            Fa0/1         13 00:16:06  403  2418  0  10
R102#sh ip eigrp n
IP-EIGRP neighbors for process 100
H   Address                 Interface   Hold Uptime   SRTT   RTO  Q  Seq Type
                                        (sec)         (ms)       Cnt Num
1   10.0.50.1               Se0/0        177 00:12:46  261  1566  0  19
2   10.1.100.101            Fa0/1         13 00:16:03 1113  5000  0  17
0   10.1.100.100            Fa0/1         12 00:16:08  403  2418  0  10
R102#sh ip eigrp n
IP-EIGRP neighbors for process 100
H   Address                 Interface   Hold Uptime   SRTT   RTO  Q  Seq Type
                                        (sec)         (ms)       Cnt Num
1   10.0.50.1               Se0/0        176 00:12:47  261  1566  0  19
2   10.1.100.101            Fa0/1         12 00:16:04 1113  5000  0  17
0   10.1.100.100            Fa0/1         11 00:16:09  403  2418  0  10
R102#sh ip eigrp n
IP-EIGRP neighbors for process 100
H   Address                 Interface   Hold Uptime   SRTT   RTO  Q  Seq Type
                                        (sec)         (ms)       Cnt Num
1   10.0.50.1               Se0/0        175 00:12:48  261  1566  0  19
2   10.1.100.101            Fa0/1         11 00:16:05 1113  5000  0  17
0   10.1.100.100            Fa0/1         10 00:16:10  403  2418  0  10
R102#sh ip eigrp n
IP-EIGRP neighbors for process 100
H   Address                 Interface   Hold Uptime   SRTT   RTO  Q  Seq Type
                                        (sec)         (ms)       Cnt Num
1   10.0.50.1               Se0/0        174 00:12:49  261  1566  0  19
2   10.1.100.101            Fa0/1         10 00:16:06 1113  5000  0  17
0   10.1.100.100            Fa0/1         14 00:16:11  403  2418  0  10
```

Il reste à relancer consécutivement plusieurs fois la commande. Sauf perte de messages Hello, le temps observé évolue entre les bornes *hold_time* et *hold_time* - *hello_interval*. Un temps qui évolue entre 15 et 10 secondes correspond aux réglages hello-interval 5s et hold-time 15s. Un temps qui évolue entre 180 et 120 secondes

correspond aux réglages *hello_interval* 60s et *hold_time* 180s. Un temps qui évoluerait dans un espace différent pourrait signifier des réglages différents des valeurs par défaut, il faut déduire que l'administrateur a configuré manuellement ces valeurs sur le routeur inconnu.

Observez à nouveau la capture ci-dessus. Elle montre le contenu de la table de voisinage et donc les différentes informations associées à chaque entrée créée dans la table. Pour un voisin :

- La première colonne de la capture, nommée « H », renseigne sur l'ordre de découverte du voisin.
- Son adresse IP.
- L'interface via laquelle le voisin a été découvert.
- Le temporisateur de maintien « Hold ».
- Le temps « Uptime » est le temps écoulé depuis la découverte de ce voisin.
- Le temps SRTT (*Smooth Round-Trip Time*, temps lissé d'aller-retour) est le temps moyen qui s'écoule entre l'émission d'un paquet vers ce voisin et l'instant où le routeur émetteur reçoit l'acquittement du paquet émis. Le processus EIGRP entretient cette valeur SRTT afin d'en déduire une valeur RTO.
- Le temps RTO (*Retransmission Time Out*, en millisecondes) est le temps limite au-delà duquel le routeur en attente d'un acquittement du voisin provoque un nouvel envoi du message non acquitté. Tout message généré par EIGRP est également placé dans une pile de messages émis et non encore acquittés (un mécanisme semblable existe dans TCP). Si le temporisateur RTO expire avant réception d'un acquittement, le processus EIGRP émet une nouvelle instance du message. Observez que RTO est pris égal à six fois SRTT, la valeur résultante est bornée dans l'espace [200 ms - 5000 ms].
- Le nombre « Q count » est le nombre de messages émis vers ce voisin mais qui sont encore en attente d'acquittement et sont donc encore dans la pile.
- Le nombre « Seq Num » est le numéro de séquence contenu dans le dernier paquet, mise à jour ou réponse, issu de ce voisin.

La commande **show ip eigrp interfaces detail** peut également se révéler utile. En voici le résultat sur un IOS 12.4 (Extrait) :

```
R101#sh ip eigrp interfaces detail
IP-EIGRP interfaces for process 100

                      Xmit Queue   Mean   Pacing Time   Multicast    Pending
Interface        Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Fa1/0              1        0/0       604       0/1         3664           0
  Hello interval is 5 sec
  Next xmit serial <none>
  Un/reliable mcasts: 0/4  Un/reliable ucasts: 4/9
  Mcast exceptions: 3  CR packets: 3  ACKs suppressed: 0
  Retransmissions sent: 2  Out-of-sequence rcvd: 0
  Authentication mode is not set
  Use multicast
Se2/0              1        0/0       617       0/9         3017           0
  Hello interval is 60 sec
  Next xmit serial <none>

                      Xmit Queue   Mean   Pacing Time   Multicast    Pending
Interface        Peers  Un/Reliable  SRTT   Un/Reliable   Flow Timer   Routes
Un/reliable mcasts: 0/0  Un/reliable ucasts: 5/14
  Mcast exceptions: 0  CR packets: 0  ACKs suppressed: 0
  Retransmissions sent: 2  Out-of-sequence rcvd: 0
  Authentication mode is not set
  Use unicast
R101#
```

Pour être complet, observons qu'EIGRP ignore les adresses secondaires éventuellement attribuées aux interfaces. Un trafic d'acheminement EIGRP est toujours issu de l'adresse primaire attribuée à l'interface. EIGRP ne limite pas le nombre de voisins qu'il est capable d'ajouter à sa table de voisinage, la limite vient donc d'ailleurs, mémoire

embarquée, consommation CPU, trafic d'acheminement, etc.

À retenir : la table de voisinage mémorise des informations qui décrivent les voisins EIGRP en activité.

5. La métrique d'EIGRP

La métrique complexe d'EIGRP est calculée de façon identique à celle du protocole parent IGRP à ceci près que ses concepteurs ont souhaité disposer d'une granularité plus fine et ont ajouté un facteur 256 (à moins qu'il ne s'agisse du souhait de retomber sur une frontière 32 bits, la métrique d'IGRP étant exprimée sur 24 bits) :

$$Métrique\ EIGRP = Métrique\ IGRP \times 256$$

La formule complète exprimant la métrique est donc :

$$\frac{Métrique\ EIGRP}{256} = \left\{ k1 \times \frac{10^7}{BWmin(Kbps)} + \frac{k2 \times \frac{10^7}{BWmin(Kbps)}}{(256 - charge)} + k3 \times \sum_{le\ long\ du\ chemin} DLY\,(10\ \mu s) \right\}$$
$$\ldots \times \{(k5 \neq 0)/(Fiabilité + k4) \mid 1\ quand\ k5 = 0\}$$

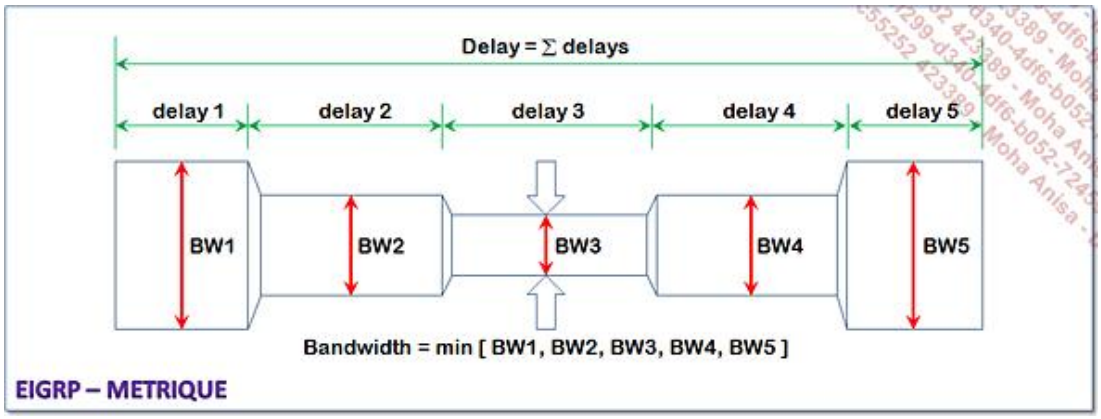
L'administrateur peut intervenir sur les coefficients à l'aide de la commande **metric weights** en configuration de routeur :

```
R11(config)#router eigrp 64501
R11(config-router)#metric weights ?
<0-8>  Type Of Service (Only TOS 0 supported)
R11(config-router)#metric weights 0 ?
<0-255>  K1
R11(config-router)#metric weights 0 1 ?
<0-255>  K2
R11(config-router)#metric weights 0 1 0 ?
<0-255>  K3
R11(config-router)#metric weights 0 1 0 1 ?
<0-255>  K4
R11(config-router)#metric weights 0 1 0 1 0 ?
<0-255>  K5
R11(config-router)#metric weights 0 1 0 1 0 0 ?
<cr>
R11(config-router)#metric weights 0 1 0 1 0 0
R11(config-router)#
```

La métrique d'EIGRP peut sembler sophistiquée mais sans précaution, cette sophistication pourrait être contre-productive car un protocole de routage a besoin de stabilité. Imaginez que parce qu'une route se charge tout à coup, un routeur joue les bisons futés, décide d'en adopter une autre et en informe tous ses petits camarades. Nous voilà peut-être lancés dans une oscillation auto-entretenue (la route se charge donc on l'évite donc elle se décharge donc on l'adopte à nouveau). Par défaut, les facteurs k2, k4 et k5 sont réglés à 0, les facteurs k1 et k3 sont réglés à 1. Si l'administrateur décide de tenir compte de la charge dans le calcul de la métrique en adoptant une valeur k2 différente de zéro, alors EIGRP tient compte de la charge mais pas n'importe quand. En fait, c'est seulement à l'occasion de l'émission d'un message de mise à jour non motivé par un changement de charge que la charge est prise en compte. Ainsi, aucun risque de créer de l'instabilité. Sans fournir d'explication, CISCO déconseille de manipuler les coefficients k1 à k5, si bien que la formule se simplifie beaucoup :

$$\frac{Métrique\ EIGRP}{256} = \left\{ \frac{10^7}{BWmin(Kbps)} + \sum_{le\ long\ du\ chemin} DLY\,(10\ \mu s) \right\}$$

Cette formule simplifiée considère le « tuyau » de bout en bout et fait intervenir le diamètre minimum de ce tuyau (la bande passante) ainsi que la longueur du tuyau (la somme des délais de chaque tronçon) :



Hélas l'IOS n'est pas toujours très cohérent dans les unités utilisées. Par exemple, la commande **clock rate** qui permet d'ajuster l'horloge côté DCE d'un câble « *back-to-back* » attend une valeur exprimée en bps alors que la commande **bandwidth** utile pour informer l'IOS de la bande passante d'un lien attend une valeur exprimée en Kbps. Soyons donc très précis sur ce point :

- **BWmin** → bande passante du lien la moins favorable parmi l'ensemble des liens qui composent la route jusqu'au réseau annoncé. Sur un routeur donné, cette valeur est statique et l'IOS la déduit automatiquement dans le cas des interfaces LAN mais l'administrateur doit la configurer dans le cas des interfaces « serial ». À défaut, la valeur de bande adoptée par l'IOS est celle d'un lien T1 (1544 Kbps), US oblige. L'administrateur ajuste la valeur associée à une interface à l'aide de la commande **bandwidth** en configuration d'interface. Cette commande attend une valeur également exprimée en Kbps.

- **DLY** → chaque routeur ajoute le délai de l'interface sur laquelle il a reçu une information de route au délai annoncé dans la mise à jour. Si bien que le délai annoncé par un routeur donné représente la somme des délais jusqu'à la route annoncée. Dans la formule utilisée pour le calcul de la métrique, les délais s'expriment en dizaines de µs. Sur un routeur donné, l'IOS associe un délai à chaque interface de façon statique et selon le type et le débit de l'interface. L'administrateur peut modifier la valeur par défaut associée à une interface à l'aide de la commande **delay** en configuration d'interface. Cette commande attend une valeur également exprimée en dizaine de µs.

La capture ci-dessous montre comment intervenir sur ces deux réglages. Observez le mot-clé « **inherit** » qui permet à une sous-interface d'hériter de la bande passante réglée sur l'interface principale. Le mot-clé « **receive** », quant à lui, est utile lorsque l'interface fonctionne selon des débits asymétriques et permet de spécifier une bande passante entrante différente de la bande passante sortante :

```
R8(config)#int s0/0
R8(config-if)#band
R8(config-if)#bandwidth ?
  <1-10000000>  Bandwidth in kilobits
  inherit       Specify that bandwidth is inherited
  receive       Specify receive-side bandwidth

R8(config-if)#bandwidth 5000
R8(config-if)#delay ?
  <1-16777215>  Throughput delay (tens of microseconds)

R8(config-if)#delay 990
R8(config-if)#
```

Si l'administrateur décide d'ignorer les conseils de CISCO et de faire intervenir charge et fiabilité dans le calcul de la métrique :

- Fiabilité (**Reliability**) → pour chaque interface, l'IOS entretient des compteurs de paquets émis et reçus ainsi des compteurs de paquets en erreur (voir capture ci-dessous). Ces comptages lui permettent d'entretenir de façon dynamique un indice de fiabilité. Cet indice est rangé dans un champ de 8 bits, il n'est donc pas exprimé en % mais en « pour 255 ». Aucun paquet en erreur implique un indice de 255, l'indice maximal.

- Charge (**Load**) → les mêmes compteurs de paquets émis et reçus permettent de calculer une bande passante effectivement occupée puis de la comparer à la bande passante disponible pour exprimer un indice de charge. Comme l'indice de fiabilité, cet indice est rangé dans un champ de 8 bits, l'indice 1 indique une charge nulle, l'indice 255 n'est jamais atteint, un indice qui dépasse 200 indique un lien très fortement chargé.

Ces deux dernières valeurs sont entretenues de façon dynamique et calculées selon une moyenne pondérée par le

temps de façon exponentielle (sic). Cela mérite quelques explications. L'IOS utilise la même méthode pour calculer l'indice de charge, l'indice de fiabilité et le débit constaté en paquets/secondes. L'ensemble des calculs est réactualisé toutes les 5 secondes. Pendant 5 secondes et pour chaque sens de flux, l'IOS compte les octets qui transitent par l'interface puis compare le contenu du compteur à ce qu'il serait si le débit avait été égal au paramètre Bandwidth.

Exemple :

BW = 1000 Kbps. Au mieux, pendant 5 secondes, l'interface peut voir transiter 5000 Kbits. Durant les 5 dernières secondes ont effectivement transité 2500 Kbits. L'échantillon de charge est 0,5.

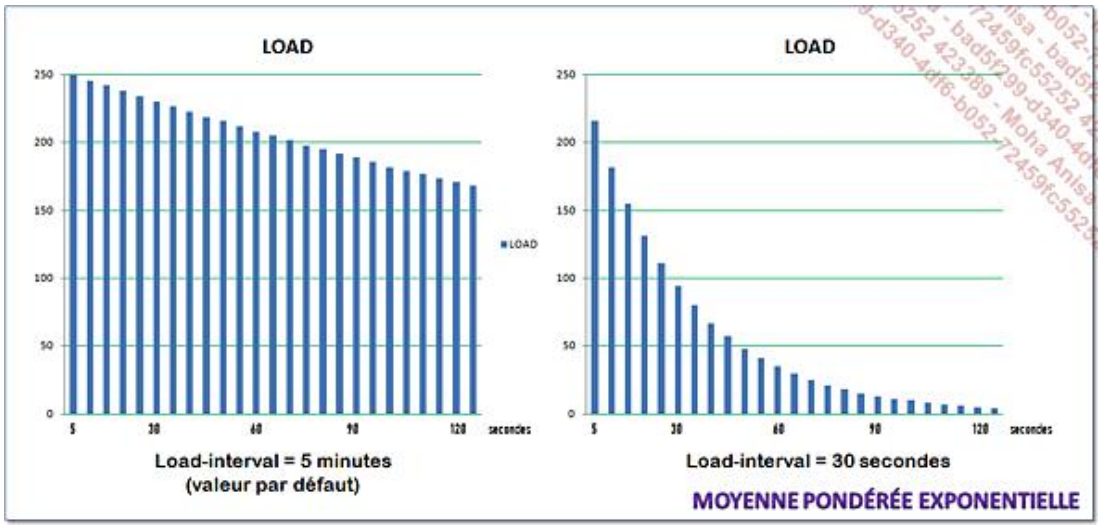
Puis l'IOS utilise une formule intégratrice dont l'objet est de pondérer les échantillons selon leur degré d'ancienneté. Le poids maximum est donné à l'échantillon le plus récent :

$$Moyenne = \left\{(Moyenne - échantillon) \times e^{-5/300}\right\} + échantillon$$

Le temps de pondération 300 secondes, soit 5 minutes est appelé **load-interval**. Il est modifiable par l'administrateur en configuration d'interface :

```
R8(config)#int s0/0
R8(config-if)#load-interval ?
<30-600>  Load interval delay in seconds

R8(config-if)#load-interval 30
R8(config-if)#^Z
R8#
```

Le tableau Excel LOAD_EIGRP.xlsx, disponible en téléchargement, permet de visualiser les effets du réglage **load-interval**. Imaginons que l'interface soit chargée à 100 % depuis une longue période et que brusquement le débit tombe à zéro. Les deux tableaux qui suivent comparent l'évolution de l'indice de charge avec pour l'un *load - interval* = 300*s* (valeur par défaut), pour l'autre *load - interval* = 30*s*, valeur minimale :



Certains administrateurs souhaitent que les indices suivent les sollicitations de l'interface de façon plus réactive et diminuent la valeur **load-interval** à 30 secondes.

Une commande **show interface** fournit la globalité des réglages en cours pouvant intervenir dans le calcul de la métrique :

```
R11#sh int s0/0
Serial0/0 is up, line protocol is up
  Hardware is PowerQUICC Serial
  Internet address is 10.0.120.5/30
  MTU 1500 bytes, BW 2500 Kbit, DLY 20000 usec,
  reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation HDLC, loopback not set
  Keepalive set (10 sec)
  Last input 00:00:00, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: weighted fair
  Output queue: 0/1000/64/0 (size/max total/threshold/drops)
```

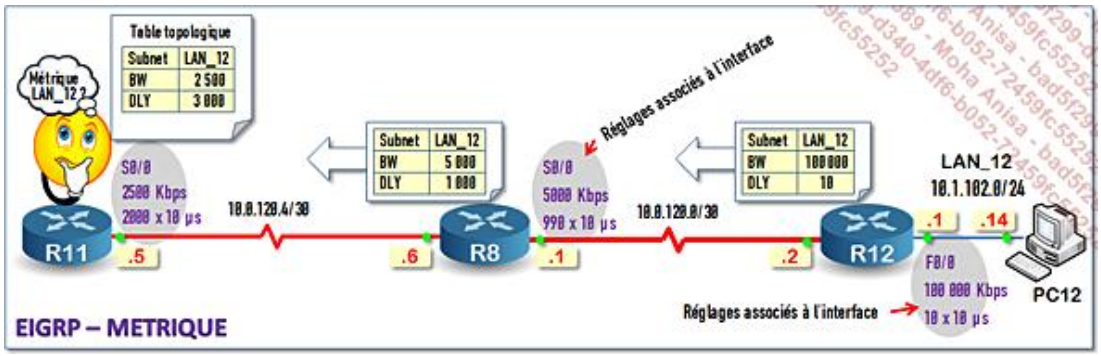
```
  Conversations  0/1/256 (active/max active/max total)
  Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 1875 kilobits/sec
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     161 packets input, 11095 bytes, 0 no buffer
     Received 55 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     234 packets output, 13590 bytes, 0 underruns
     0 output errors, 0 collisions, 18 interface resets
     0 unknown protocol drops
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up

R11#
```

La section suivante Espacement des mises à jour montre comment EIGRP procède afin de limiter la bande passante occupée par le trafic d'acheminement. Pour mémoire, le trafic d'acheminement est le trafic généré par le protocole de routage pour ses besoins propres. C'est le contraire d'un trafic utile. Il ne serait évidemment pas admissible qu'un protocole de routage consomme l'essentiel de la ressource bande passante d'un lien. Il se trouve qu'EIGRP se fonde sur l'information **bandwidth** pour calculer la bande qu'il peut occuper sur un lien. La conséquence est que la bande passante réglée à l'aide de la commande **bandwidth** doit refléter la bande passante réelle d'un lien. L'administrateur doit s'interdire d'utiliser ce paramètre pour influer la métrique et doit au contraire s'assurer que chaque interface « *serial* » dispose de sa commande **bandwidth** afin d'ajuster la bande passante effectivement disponible.

La bande passante réglée à l'aide de la commande **bandwidth** devrait toujours refléter la bande passante réelle des liens. Cette règle peut souffrir quelques exceptions dans le cas de liaisons série multipoint, ce point est détaillé dans la section dédiée aux liens WAN du dernier ouvrage de cette série.

Un administrateur qui souhaite influer sur le résultat du calcul de la métrique peut le faire à l'aide du paramètre délai ajusté à l'aide de la commande **delay**.

Il est temps d'étudier un cas concret. Considérez la topologie ci-dessous :



Les coefficients k1 à k5 sont ceux par défaut et c'est donc la formule simplifiée du calcul de métrique qui est utilisée.

- Quand R12 annonce connaître LAN_12, il le fait en annonçant la bande passante et le délai associés à son interface F0/0, soit BW = 100 000 Kbps et DLY = 10 x 10 µs.

- Quand R8 reçoit une annonce de R12 au sujet de LAN_12, R8 compare la bande passante annoncée et la bande passante de l'interface qui reçoit. R8 ne retient que la plus basse des deux valeurs, 5000 Kbps dans le cas présent. R8 ajoute le délai annoncé au délai associé à l'interface qui reçoit, ce qui porte le délai total à (990 + 10) x 10 = 1000 x 10 µs.

- Quand R8 annonce connaître LAN_12, il le fait en annonçant une bande passante de 5000 Kbps et un délai de 1000 x 10 µs.

- Quand R11 reçoit une annonce de R8 au sujet de LAN_12, R11 compare la bande passante annoncée et la bande passante de l'interface qui reçoit. R11 ne retient que la plus basse des deux valeurs, 2500 Kbps dans le cas présent. R11 ajoute le délai annoncé au délai associé à l'interface qui reçoit, ce qui porte le délai total à (1000 + 2000) x 10 = 3000 x 10 µs.

À l'aide du tableau Excel « Calcul_Métrique_EIGRP.xlsm » fourni en téléchargement, calculons la métrique de la route vers LAN_12 vue des routeurs R8 et R11. Le tableau Excel fourni mérite une explication sur la façon de le remplir. Le tableau prévoit dix liens maximum entre le réseau de destination et le routeur considéré. Chaque ligne du tableau correspond à un lien pour lequel il suffit de compléter les deux cellules BW en Kbps et DLY en µs. Pour une ligne donnée, c'est-à-dire pour un lien, ce sont les paramètres BW et DLY de l'interface qui reçoit qu'il convient de consigner dans le tableau. Pour chaque ligne, le tableau Excel retient la bande passante la moins favorable et fait la somme des délais afin d'exprimer la métrique IGRP puis d'en extrapoler la métrique EIGRP. Appliqué au cas présent :

Lien	BW (Kbps)	DLY (µs)	BW résultante	DLY résultant	Métrique IGRP	Métrique EIGRP
F0/0 de R12	100 000	100	100 000	100	110	28 160
S0/0 de R8	5 000	9 900	5 000	10 000	3 000	768 000
S0/0 de R11	2 500	20 000	2 500	30 000	7 000	1 792 000

La métrique 768 000 est donc celle calculée par R8, ce que nous pouvons vérifier à l'aide d'une commande **show ip route** sur ce routeur :

```
R8#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
D       10.1.102.0/24 [90/768000] via 10.0.120.2, 00:02:34, Serial0/1
C       10.0.120.0/30 is directly connected, Serial0/1
C       10.0.120.4/30 is directly connected, Serial0/0
R8#
```

Toujours pour la route vers LAN_12, la métrique 1 792 000 est celle calculée par R11 :

```
R11#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
D       10.1.102.0/24 [90/1792000] via 10.0.120.6, 00:03:04, Serial0/0
D       10.0.120.0/30 [90/1789440] via 10.0.120.6, 00:03:15, Serial0/0
C       10.0.120.4/30 is directly connected, Serial0/0
R11#
```

6. Construction de la table topologique

RIP et IGRP transmettaient leurs tables de routage à leurs voisins, nous avons vu que ce mode de propagation de l'information contribuait à dégrader le temps de convergence. EIGRP entretient une table intermédiaire dite table de topologie dont il déduit sa table de routage. Les mises à jour envoyées vers un voisin comportent des informations issues non pas de la table de routage mais de la table de topologie. Il est possible d'examiner un contenu partiel de cette table à l'aide d'une commande **show ip eigrp topology** :

```
R11#sh ip eigrp topology ?
  <1-65535>        AS Number
  A.B.C.D          IP prefix <network>/<length>, e.g., 192.168.0.0/16
  A.B.C.D          Network to display information about
  active           Show only active entries
  all-links        Show all links in topology table
  detail-links     Show all links in topology table
  pending          Show only entries pending transmission
  summary          Show a summary of the topology table
  zero-successors  Show only zero successor entries
  |                Output modifiers
<cr>
```

```
R11#sh ip eigrp topology
IP-EIGRP Topology Table for AS(64501)/ID(10.0.120.5)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

P 10.1.102.0/24, 1 successors, FD is 1792000
  via 10.0.120.6 (1792000/768000), Serial0/0
P 10.0.120.0/30, 1 successors, FD is 1789440
  via 10.0.120.6 (1789440/765440), Serial0/0
P 10.0.120.4/30, 1 successors, FD is 1536000
  via Connected, Serial0/0
```

L'entrée dans cette base de données est le réseau. Chaque nouveau réseau annoncé par un voisin est ajouté dans la table. Si ce réseau était déjà présent dans la table mais annoncé antérieurement par un autre voisin, le voisin qui l'annonce cette fois est ajouté si bien que l'entrée comprend l'ensemble des voisins qui ont annoncé cette destination. À chaque voisin, l'entrée enregistre la métrique annoncée, il s'agit de la distance qui sépare ce voisin de la destination tel qu'il l'a placé dans sa table de routage. Autrement dit, la route annoncée est effectivement celle que le voisin utilise pour acheminer les paquets vers cette destination (souvenons-nous que c'est là un trait de caractère des protocoles DV).

En fait, dans cette table, EIGRP collecte l'information nécessaire lui permettant la création d'ensembles [distance, vecteur] pour chaque réseau connu. Dans le détail, pour chaque sous-entrée de voisin dans une entrée de réseau, il s'agit de :

- La bande passante la plus faible rencontrée sur le chemin qui mène au réseau annoncé, telle qu'elle est annoncée par le voisin.
- Le délai total du même chemin qui mène au réseau annoncé.
- La fiabilité du même chemin.
- La charge du même chemin.
- Le MTU minimum de ce chemin, parmi les MTU des différents tronçons qui composent le chemin. On se souvient que le MTU (*Maximum Transmission Unit*) est la capacité d'emport de la trame en couche 2. Une section assez détaillée traite ce sujet dans le chapitre La couche Transport TCP/IP de l'ouvrage Cisco - Notions de base sur les réseaux dans la collection Certifications aux Editions ENI. Chaque routeur qui reçoit une annonce de réseau compare l'information MTU placée dans l'annonce au MTU de l'interface qui reçoit et ne retient que la plus faible des deux valeurs. Tout se passe comme si EIGRP avait intégré l'objectif du protocole PMTUd (*Path MTU Discovery*, décrit dans le RFC 1191) dont l'objet est de découvrir quel est le MTU idéal entre deux hôtes, c'est-à-dire celui qui évite la fragmentation.
- La distance annoncée (*Reported distance*). En réalité, cette distance n'est pas annoncée stricto sensu mais « c'est tout comme » puisque le voisin fournit l'ensemble des éléments qui ont servi à son propre calcul (BW, DLY, fiabilité et charge).
- La source d'apprentissage de la route (S'agit-il d'une route interne ou externe ?).

Retenez cette définition temporaire : la table de topologie contient la liste des routes connues.

Cette définition sera amandée après avoir défini la notion de successeur faisable.

7. Espacement des mises à jour

EIGRP est très soucieux de ne pas accaparer la bande passante des liens pour son propre usage. Par défaut, le trafic d'acheminement sur un lien n'occupe pas plus de 50 % de la bande passante disponible. L'administrateur peut intervenir sur ce pourcentage à l'aide de la commande **ip bandwidth-percent eigrp** en configuration d'interface. La syntaxe de cette commande est :

```
Router(config-if)#ip bandwidth-percent eigrp as-number percent
```

... dont les arguments sont les suivants :

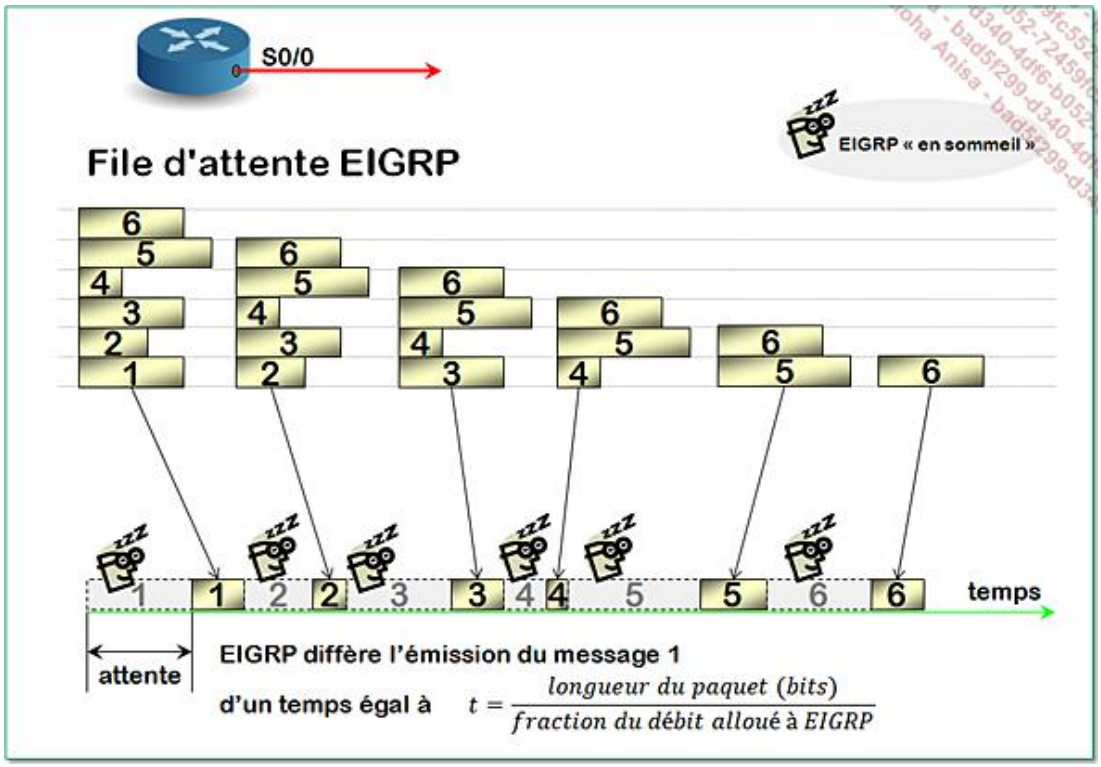
as-number

Numéro d'AS.

mask percent

Part de bande passante qu'EIGRP peut utiliser, exprimée en %.

Le très sérieux document n°16406 téléchargeable sur le site CISCO et dédié à EIGRP explique la procédure mise en place par le constructeur. La figure suivante devrait aider à en comprendre le principe :



À chaque fois qu'EIGRP doit émettre un paquet (un quelconque message encapsulé dans RTP), ce paquet est placé en file d'attente et EIGRP diffère son émission d'un temps égal au temps qu'occuperait le paquet s'il était émis sur l'interface avec un débit égal à la bande passante allouée à EIGRP (mais si !). Faisons quelques essais pour mesurer la portée d'une telle décision.

Imaginons qu'EIGRP doive émettre un paquet de 1000 bits, que le débit de l'interface réglé à l'aide de la commande **bandwidth** soit 2000 bits/s et que le pourcentage de débit autorisé à EIGRP soit laissé à la valeur par défaut soit 50 %. La fraction du débit alloué à EIGRP est donc 1000 bits/s. Le temps du paquet à ce débit est 1 seconde. EIGRP attend une seconde puis émet le paquet à 2000 bits/s, le temps d'émission est 0,5 secondes. En final, EIGRP n'a occupé le canal que pendant le tiers du temps et en fait d'occupation à 50 %, nous plafonnons en réalité à 33 % (mathématiques CISCO sans doute).

Traitons un deuxième et dernier exemple, toutes choses étant égales sauf le pourcentage alloué à EIGRP, cette fois réglé par l'administrateur à 25 %. La fraction du débit allouée à EIGRP est donc 500 bits/s. À ce rythme, le temps du paquet est 2 secondes. EIGRP attend 2 secondes puis émet le paquet au rythme de l'interface, le temps d'émission est encore 0,5 secondes. En final, EIGRP n'a occupé le canal que pendant 20 % du temps total.

Bref, l'important est de constater qu'EIGRP fait ce qu'il dit, c'est-à-dire qu'il ne dépasse pas la bande passante que lui octroie l'administrateur.

8. Algorithme DUAL

L'algorithme DUAL (*Diffusing Update ALgorithm*) remplace l'algorithme dit Bellman-Ford en usage dans RIP. On le doit aux travaux initiés par M. E. W. Dijkstra et M. C. S. Scholten puis développés par M. J.J. Garcia-Luna-Aceves. On trouve sans difficulté les publications correspondantes sur Internet et notamment :

« *A unified Approach to Loop-free Routing using Distance Vectors or Link States* » publiée dans la revue « *ACM SIGCOMM Computer Communications Review Vol 19, Issue 4* » en septembre 1989.

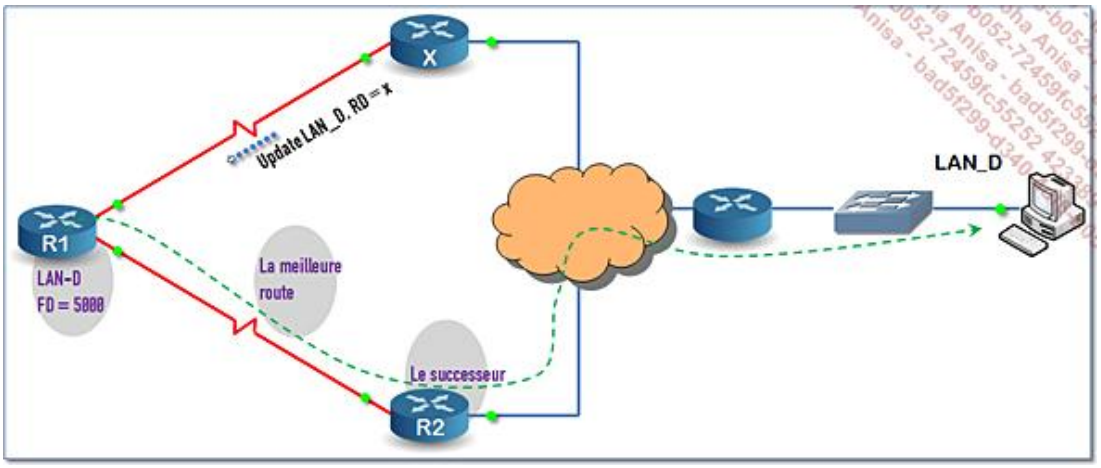
DUAL permet à EIGRP de déterminer le meilleur chemin sans boucle mais aussi les alternatives de secours sans boucle.

a. Le successeur, Notion de distance de faisabilité

De la collection de voisins associés à une entrée réseau dans la table topologique, le routeur choisit le successeur c'est-à-dire le voisin qui permet de rejoindre la destination avec la meilleure métrique. La distance en question, c'est-à-dire la métrique calculée via le successeur, prend alors le nom de distance faisable (FD, *Feasible Distance*). La distance faisable est également ajoutée à la table topologique associée à l'entrée réseau et c'est cette distance que le routeur place dans sa propre table de routage et annoncera à ses voisins pour le réseau en question.

b. Condition de faisabilité et successeurs potentiels

Considérez la topologie ci-dessous :



La métrique utilisée dans cet exemple est la métrique IGRP (divisez la métrique EIGRP par 256) afin de manipuler des nombres plus petits.

Pour le routeur R1, la meilleure route vers LAN_D passe par R2 au coût de 5000. Autrement dit, la distance faisable pour rejoindre LAN_D est → *FD* = 5000.

Le routeur X connaît également LAN_D et l'annonce au coût de x → *RD* = *x*.

R1 ajoute X à sa collection de voisins qui connaissent LAN_D. Imaginons maintenant que le lien de R1 à R2 vienne à tomber. R1 peut substituer X à R2 à la condition que X n'utilise pas R1 pour faire progresser les paquets destinés à LAN_D. Autrement dit, le chemin via X doit être exempt de boucle et R1 doit pouvoir s'en persuader.

La solution consiste pour R1 à comparer la distance rapportée par X à la distance faisable pour LAN_D :

RD < *FD* ?

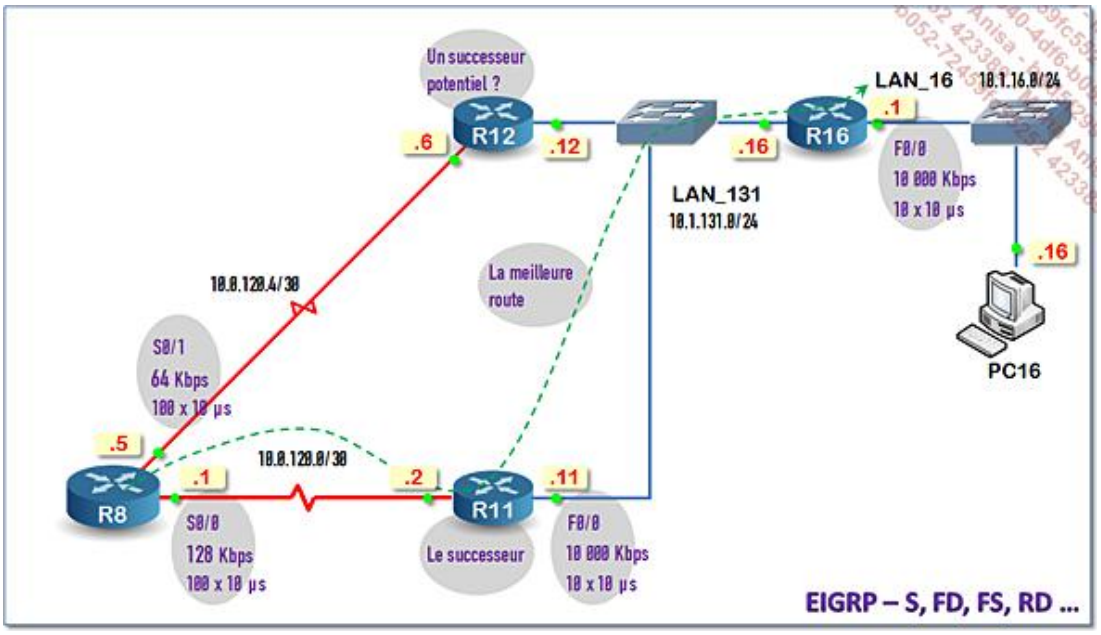
- Si la réponse est non, par exemple si RD = 7000, il est possible que 7000 résulte du coût de R1 vers LAN_D additionné au coût du lien de X à R1. Ce n'est pas certain mais il y a un risque.

- Si la réponse est oui, par exemple si RD = 2000, alors R1 peut avoir la certitude que le chemin connu par X pour rejoindre LAN_D ne passe pas par lui. Cette condition réalisée fait de X un successeur potentiel ou successeur faisable. En cas de défaillance du chemin par R2, R1 peut instantanément lui substituer le chemin par X.

Cette condition est appelée condition de faisabilité (*Feasibility condition*). C'est le « grand truc » d'EIGRP. La condition de faisabilité n'est pas une condition nécessaire, il peut exister des topologies exemptes de boucle pour lesquelles RD serait supérieur à FD. Mais c'est une condition suffisante : dès que RD est inférieur à FD, on peut être certain que la topologie est exempte de boucle.

Parmi les voisins restants ayant annoncé le réseau, les successeurs potentiels sont ceux pour lesquels la condition de faisabilité est respectée, c'est-à-dire ceux qui ont annoncé une métrique inférieure à la métrique en cours dans la table de routage. Il est possible qu'aucun des voisins ne respecte cette condition et dans ce cas, le routeur ne dispose d'aucun successeur potentiel. Il est possible que parmi ces voisins non potentiels, l'un d'eux offre une alternative sans boucle mais EIGRP ne peut en être sûr et préfère l'ignorer. En adoptant la condition de faisabilité, EIGRP passera peut-être à côté d'une route possible qu'il aurait pu substituer à la route en cours défaillante mais EIGRP ne prend pas de risque.

Ancrons ces notions avec deux exemples.



L'administrateur vérifie que R8 connaît effectivement deux voisins :

```
R8#sh ip eigrp neighbors
IP-EIGRP neighbors for process 64501
H   Address                 Interface   Hold Uptime   SRTT   RTO  Q  Seq Type
                                        (sec)         (ms)       Cnt Num
1   10.0.120.6              Se0/1         12 00:00:02  292  2280  0  10
0   10.0.120.2              Se0/0         14 00:00:10   40  1140  0  13
```

À l'aide du tableau Excel « Calcul_Métrique_EIGRP.xlsm » fourni en téléchargement, calculons la métrique de la route vers LAN_16 vue des routeurs R8 et R11 :

Lien	BW (Kbps)	DLY (µs)	BW résultante	DLY résultant	Métrique IGRP	Métrique EIGRP
F0/0 de R16	10 000	100	10 000	100	1 010	258 560
F0/0 de R11	10 000	100	10 000	200	1 020	261 120
S0/0 de R8	128	1 000	128	1 200	78 245	20 030 720

Répétons le calcul mais cette fois via R12 :

Lien	BW (Kbps)	DLY (µs)	BW résultante	DLY résultant	Métrique IGRP	Métrique EIGRP
F0/0 de R16	10 000	100	10 000	100	1 010	258 560
F0/0 de R12	10 000	100	10 000	200	1 020	261 120
S0/1 de R8	64	1 000	64	1 200	156 370	40 030 720

Il est donc établi que le routeur R8 dispose de deux alternatives pour rejoindre LAN_16 :

- La route via R11 au coût de 20 030 720 dont la distance rapportée (le coût tel qu'il est calculé par R11 pour rejoindre LAN_16) est 261 120.
- La route via R12 au coût de 40 030 720 dont la distance rapportée (le coût tel qu'il est calculé par R12 pour rejoindre LAN_16) est également 261 120.

Une commande **show ip eigrp topology** confirme nos calculs :

```
R8#sh ip eigrp topology
IP-EIGRP Topology Table for AS(64501)/ID(10.0.120.5)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

P 10.0.16.0/24, 1 successors, FD is 20030720
     via 10.0.120.2 (20030720/261120), Serial0/0
     via 10.0.120.6 (40030720/261120), Serial0/1
P 10.0.120.0/30, 1 successors, FD is 20025600
     via Connected, Serial0/0
P 10.0.120.4/30, 1 successors, FD is 40025600
     via Connected, Serial0/1
P 10.0.131.0/24, 1 successors, FD is 20028160
     via 10.0.120.2 (20028160/258560), Serial0/0
     via 10.0.120.6 (40028160/258560), Serial0/1
R8#
```

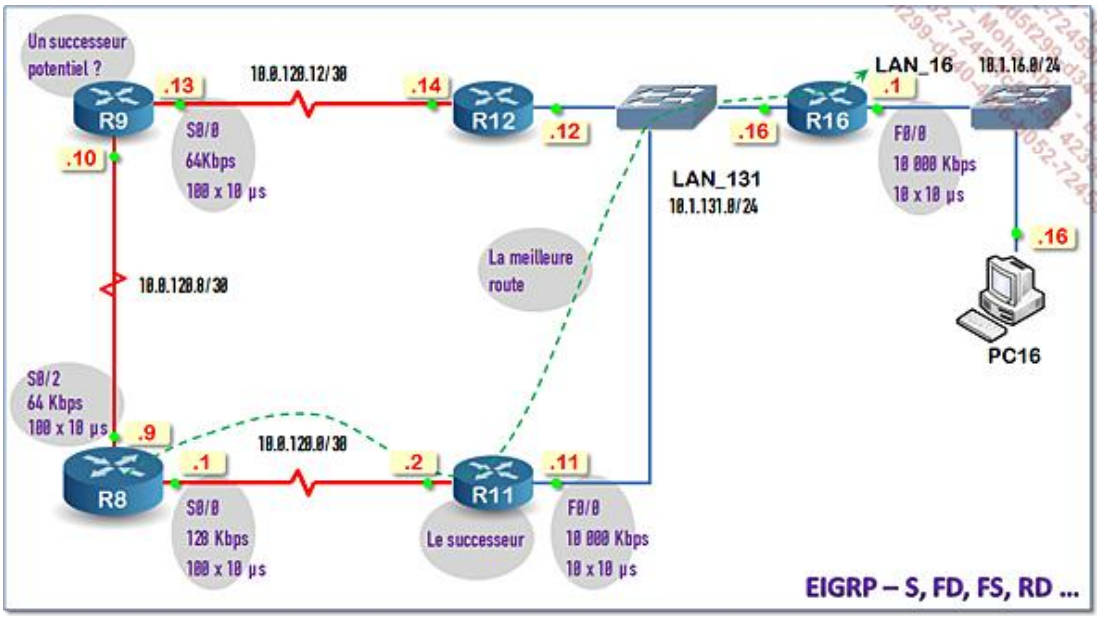
R8 choisit la route au meilleur coût, c'est donc la route via R11, la métrique de cette route devient la distance faisable, R11 est le successeur. Mais puisque la distance rapportée par R12 est inférieure à la distance faisable, R8 considère R12 comme un successeur faisable. C'est pourquoi les deux voisins apparaissent dans la table de topologie pour le réseau 10.0.16.0/24.

L'ultime confirmation vient de l'observation de la table de routage :

```
R8#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
D       10.0.16.0/24 [90/20030720] via 10.0.120.2, 00:00:13, Serial0/0
C       10.0.120.0/30 is directly connected, Serial0/0
C       10.0.120.4/30 is directly connected, Serial0/1
D       10.0.131.0/24 [90/20028160] via 10.0.120.2, 00:00:13, Serial0/0
R8#
```

L'administrateur est heureux et commence à trouver EIGRP décidément bien plaisant.

Complexifions un peu notre scénario :



L'administrateur vérifie que R8 connaît effectivement deux voisins :

```
R8#sh ip eigrp neighbors
IP-EIGRP neighbors for process 64501
H   Address                 Interface   Hold Uptime   SRTT   RTO  Q  Seq Type
```

```
                                   (sec)         (ms)       Cnt Num
1   10.0.120.2              Se0/0        12 00:01:25   45  1140  0  28
0   10.0.120.10             Se0/2        13 00:01:43   44  2280  0  17
```

À l'aide du tableau Excel « Calcul_Métrique_EIGRP.xlsm » fourni en téléchargement, calculons la métrique de la route vers LAN_16 vue des routeurs R8 et R11 :

Lien	BW (Kbps)	DLY (µs)	BW résultante	DLY résultant	Métrique IGRP	Métrique EIGRP
F0/0 de R16	10 000	100	10 000	100	1 010	258 560
F0/0 de R11	10 000	100	10 000	200	1 020	261 120
S0/0 de R8	128	1 000	128	1 200	78 245	20 030 720

Répétons le calcul mais cette fois via R9 :

Lien	BW (Kbps)	DLY (µs)	BW résultante	DLY résultant	Métrique IGRP	Métrique EIGRP
F0/0 de R16	10 000	100	10 000	100	1 010	258 560
F0/0 de R12	10 000	100	10 000	200	1 020	261 120
S0/0 de R9	64	1 000	64	1 200	156 370	40 030 720
S0/0 de R8	64	1 000	64	2 200	156 470	40 056 320

Il est donc établi que le routeur R8 dispose de deux alternatives pour rejoindre LAN_16 :

- La route via R11 au coût de 20 030 720 dont la distance rapportée (le coût tel qu'il est calculé par R11 pour rejoindre LAN_16) est 261 120.
- La route via R9 au coût de 40 056 320 dont la distance rapportée (le coût tel qu'il est calculé par R9 pour rejoindre LAN_16) est 40 030 720.

Une commande **show ip eigrp topology** confirme partiellement nos calculs :

```
R8#sh ip eigrp topology
IP-EIGRP Topology Table for AS(64501)/ID(10.0.120.5)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

P 10.0.16.0/24, 1 successors, FD is 20030720
   via 10.0.120.2 (20030720/261120), Serial0/0
P 10.0.120.0/30, 1 successors, FD is 20025600
   via Connected, Serial0/0
P 10.0.120.8/30, 1 successors, FD is 40025600
   via Connected, Serial0/2
P 10.0.120.12/30, 1 successors, FD is 40051200
   via 10.0.120.10 (40051200/40025600), Serial0/2
   via 10.0.120.2 (40053760/40028160), Serial0/0
P 10.0.131.0/24, 1 successors, FD is 20028160
   via 10.0.120.2 (20028160/258560), Serial0/0
```

R8 choisit la route au meilleur coût, c'est donc la route via R11, la métrique de cette route devient la distance faisable, R11 est le successeur. Mais puisque la distance reportée par R9 est supérieure à la distance faisable, R8 ne considère pas R9 comme un successeur faisable. C'est pourquoi seul le voisin R11 apparaît dans la table de topologie pour le réseau 10.0.16.0/24. En réalité, la table de topologie comporte bien les deux voisins c'est-à-dire le successeur et le voisin simple (ni successeur, ni successeur faisable) mais pour l'observer, il faut utiliser la commande **show ip eigrp topology** associée au mot-clé optionnel **all-links** :

```
R8#sh ip eigrp topology all-links
```

```
IP-EIGRP Topology Table for AS(64501)/ID(10.0.120.5)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

P 10.0.16.0/24, 1 successors, FD is 20030720, serno 22
         via 10.0.120.2 (20030720/261120), Serial0/0
         via 10.0.120.10 (40056320/40030720), Serial0/2
P 10.0.120.0/30, 1 successors, FD is 20025600, serno 15
         via Connected, Serial0/0
P 10.0.120.8/30, 1 successors, FD is 40025600, serno 7
         via Connected, Serial0/2
P 10.0.120.12/30, 1 successors, FD is 40051200, serno 8
         via 10.0.120.10 (40051200/40025600), Serial0/2
         via 10.0.120.2 (40053760/40028160), Serial0/0
P 10.0.131.0/24, 1 successors, FD is 20028160, serno 16
         via 10.0.120.2 (20028160/258560), Serial0/0
         via 10.0.120.10 (40053760/40028160), Serial0/2
```

Une commande **show ip route** montre que la route vers LAN_16 est inchangée :

```
R8#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 5 subnets, 2 masks
D       10.0.16.0/24 [90/20030720] via 10.0.120.2, 00:00:55, Serial0/0
C       10.0.120.0/30 is directly connected, Serial0/0
C       10.0.120.8/30 is directly connected, Serial0/2
D       10.0.120.12/30 [90/40051200] via 10.0.120.10, 00:01:33, Serial0/2
D       10.0.131.0/24 [90/20028160] via 10.0.120.2, 00:01:33, Serial0/0
R8#
```

Le vrai changement ne peut s'observer qu'en cas de défaillance de la route vers LAN_16. Dans le scénario précédent, R8 pouvait instantanément basculer sur l'alternative via le successeur potentiel R12. Dans le scénario présent, imaginons que le lien entre R8 et R11 tombe. R8 constate que la seule alternative qu'il est autorisé à utiliser vers LAN_16 vient de disparaître. Il reste à R8 à interroger tous ses voisins en espérant que l'un d'eux connaisse LAN_16. R9, seul routeur voisin restant, répond qu'il connaît LAN_16. R8 accepte la proposition et fait de R9 le nouveau successeur.

Le temps de convergence est un peu plus long mais encore très bref, le questionnement n'ayant pas dépassé le routeur voisin R9. Pour vérifier ce comportement, il suffit de lancer une commande **debug ip eigrp** puis de provoquer une défaillance en un point de la route vers LAN_16 (dans un atelier réalisé sous GNS3, un clic droit sur R11 puis Suspendre). Observez la requête de R8, la valeur 4294967295 est le délai annoncé pour LAN_16 (FF FF FF FF) et doit être interprétée comme un réseau injoignable. Observez la réponse de R9 et la métrique calculée par R8 pour rejoindre LAN_16 à partir des paramètres BW et DLY fournis par R9, soit 40 056 320. Observez enfin le coût réactualisé de la route vers LAN_16 dans la table de routage :

```
R8#debug ip eigrp
00:15:43: %DUAL-5-NBRCHANGE: IP-EIGRP 64501: Neighbor 10.0.120.2 (Serial0/0) is up:
new adjacency
R8#debug ip eigrp
IP-EIGRP Route Events debugging is on
R8#
00:16:26: %DUAL-5-NBRCHANGE: IP-EIGRP 64501: Neighbor 10.0.120.2 (Serial0/0) is
down: holding time expired
R8#
00:16:26: IP-EIGRP: 10.0.16.0/24 - do advertise out Serial0/2  !>>> La requête à R9
00:16:26: IP-EIGRP: Int 10.0.16.0/24 metric 4294967295 - 20000000 4294967295
00:16:26: IP-EIGRP: 10.0.131.0/24 - do advertise out Serial0/2
00:16:26: IP-EIGRP: Int 10.0.131.0/24 metric 4294967295 - 20000000 4294967295
00:16:26: IP-EIGRP: Processing incoming REPLY packet !>>>La réponse de R9
00:16:26: IP-EIGRP: Int 10.0.16.0/24 M 40056320 - 40000000 56320 SM 40030720 -
40000000 30720
.........
R8#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
```

```
.........
Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
D       10.0.16.0/24 [90/40056320] via 10.0.120.10, 00:00:47, Serial0/2
C       10.0.120.8/30 is directly connected, Serial0/2
D       10.0.120.12/30 [90/40051200] via 10.0.120.10, 00:01:29, Serial0/2
D       10.0.131.0/24 [90/40053760] via 10.0.120.10, 00:00:47, Serial0/2
R8#
```

L'administrateur a également pris la précaution en outre de lancer une capture Wireshark sur le lien S0/2 de R8 (capture cap_2H_01.pcap disponible en téléchargement). Ouvrez la capture et observez la séquence suivante :

- Trame 27 → Requête de R8, l'adresse de destination est l'adresse unicast de R9. Le numéro de séquence RTP est 54.
- Trame 28 → R9 acquitte la requête à l'aide d'un message Hello sans données, numéro de séquence RTP acquitté 54.
- Trame 29 → R9 répond à la requête en informant R8 de son coût vers LAN_16. Le numéro de séquence RTP est 50.
- Trame 30 → R8 acquitte la réponse.
- Trame 31 → R8 envoie un message de mise à jour à R9 pour l'informer que LAN_16 est injoignable par lui. De quoi peut-il s'agir ? Remémorez-vous les mécanismes qui avaient permis, plus ou moins bien, d'empêcher les boucles dans RIP. Mais oui, c'est une manifestation de type partage d'horizon avec empoisonnement de route inverse !

c. DUAL, synthèse partielle

À ce stade, une synthèse partielle n'est sans doute pas superflue :

- Un **successeur** est un voisin que le routeur utilise pour faire progresser les paquets. Le successeur est choisi parmi les voisins parce qu'il offre l'alternative au meilleur coût jusqu'au réseau de destination. Attention, c'est aussi celui que nous avions désigné jusqu'à présent par routeur de prochain saut, mais il faut nous adapter à la terminologie EIGRP.
- La distance qui sépare le routeur du réseau de destination via le successeur est appelée **distance faisable** ou distance de faisabilité (FD, *Feasible Distance*).
- Un **successeur potentiel** ou successeur faisable (FS, *Feasible Successor*) est un voisin qui satisfait la condition de faisabilité, c'est-à-dire dont la distance annoncée ou rapportée (RD, *Reported Distance*) est inférieure à la distance faisable.
- La **condition de faisabilité** (FC) est remplie lorsque la distance annoncée (ou rapportée) par un voisin (RD) est inférieure à la distance faisable (FD). Satisfaire la condition de faisabilité assure que le voisin offre une alternative sans boucle.
- La **table topologique EIGRP** contient l'ensemble des routes connues qui ont un successeur faisable. À la première annonce d'un réseau inconnu jusque-là, la distance faisable était infinie et le voisin qui annonce la route remplit obligatoirement la condition de faisabilité. Le voisin reste un successeur faisable mais devient en outre le successeur, la distance faisable est mise à jour.
- Chaque route peut avoir été annoncée par plusieurs voisins, la table de topologie les inventorie. À chaque voisin, la table topologique associe deux métriques : (xD / RD). xD est le coût du chemin qui sépare le routeur du réseau de destination, via le voisin. RD est le coût du chemin qui sépare le voisin du réseau de destination. Les voisins se classent en trois catégories :
  - Les successeurs, c'est-à-dire les voisins qui présentent la plus petite valeur xD, cette valeur devient FD, la distance de faisabilité.
  - Les successeurs potentiels, c'est-à-dire les voisins pour lesquels RD < FD.

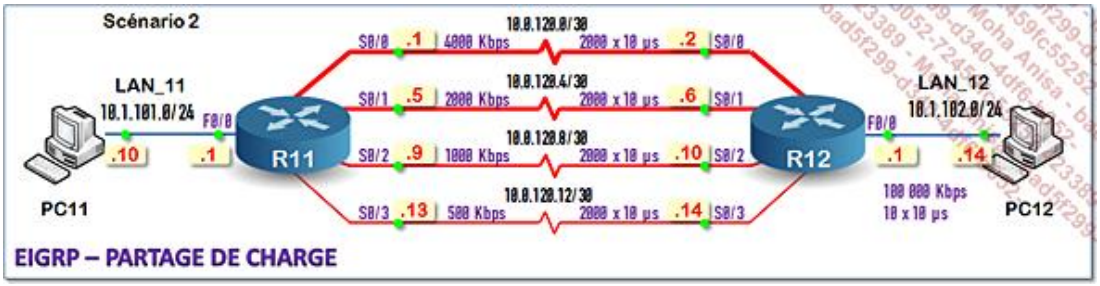
- Les voisins simples, c'est-à-dire les voisins qui ne sont ni successeurs, ni successeurs potentiels.

Une commande **show ip eigrp topology** montre le contenu partiel de la table de topologie, les voisins simples sont ignorés. Une commande **show ip eigrp topology all-links** montre tous les chemins possibles, voisins simples y compris.

Un routeur qui dispose d'au moins un successeur potentiel pour un réseau peut basculer de façon quasi instantanée vers le chemin via le successeur potentiel quand le chemin via le successeur est défaillant. Ceci explique en bonne partie les performances d'EIGRP en matière de temps de convergence. En final, EIGRP fonctionnera d'autant mieux que les routeurs disposent de successeurs potentiels.

Un voisin est toujours un routeur mais un même routeur peut être voisin plusieurs fois.

Considérez la topologie ci-dessous :



R12 est quatre fois voisin de R11 parce qu'il existe quatre liens parallèles entre R11 et R12, ce que confirme une commande **show ip eigrp topology** (Extrait) :

```
R11#sh ip eigrp topology
IP-EIGRP Topology Table for AS(64501)/ID(10.1.101.1)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

P 10.1.102.0/24, 1 successors, FD is 1154560
via 10.0.120.2 (1154560/28160), Serial0/0
via 10.0.120.6 (1794560/28160), Serial0/1
via 10.0.120.14 (5634560/28160), Serial0/3
via 10.0.120.10 (3074560/28160), Serial0/2
.........
R11#
```

d. Notion de calcul diffusé

EIGRP peut se trouver dans un état stable, c'est évidemment le souhait de l'administrateur, quand chaque route qui compose la table de topologie est à l'état passif. Observez le résultat de la commande **show ip eigrp topology** immédiatement précédente. L'état passif de la route est rappelé par la lettre **P** en première colonne associée à l'entrée de route dans la table topologique.

Un routeur est amené à remettre en question sa liste de successeurs faisables pour une route quand se produit l'un des évènements suivants :

- Un lien directement connecté change de coût (probablement suite à une intervention de l'administrateur).
- Un lien directement connecté change d'état.
- Le routeur reçoit un message parmi les messages mise à jour (*Update*), de requête (*Query*) ou de réponse (*Reply*).

La première étape de cette remise en question consiste à réévaluer localement la distance qui sépare le routeur de la destination, ce pour chaque successeur faisable. Les premiers résultats possibles de ce calcul local sont :

1. Il existe un successeur faisable qui présente désormais un meilleur coût que le successeur actuel → ce successeur faisable prend la place du successeur.

2. La nouvelle distance calculée via le successeur actuel est plus favorable que la distance faisable → EIGRP met à jour la distance faisable.

Dans ces deux premiers cas, la distance faisable a changé. Notez bien que, pendant cette phase locale de réévaluation, la route est restée à l'état passif. Mais finalement, le vrai résultat de cette réévaluation se ramène à une seule question : oui ou non, le routeur a-t-il trouvé un successeur faisable pour la route en question ?

Oui → un message de mise à jour est envoyé à tous les voisins mais la route reste dans l'état passif.

Non → le routeur est contraint d'entamer une procédure de calcul diffusé (*Diffusing computation*), la route passe à l'état actif.

Une route passe à l'état actif quand il n'existe pas de successeur faisable dans la table de topologie.

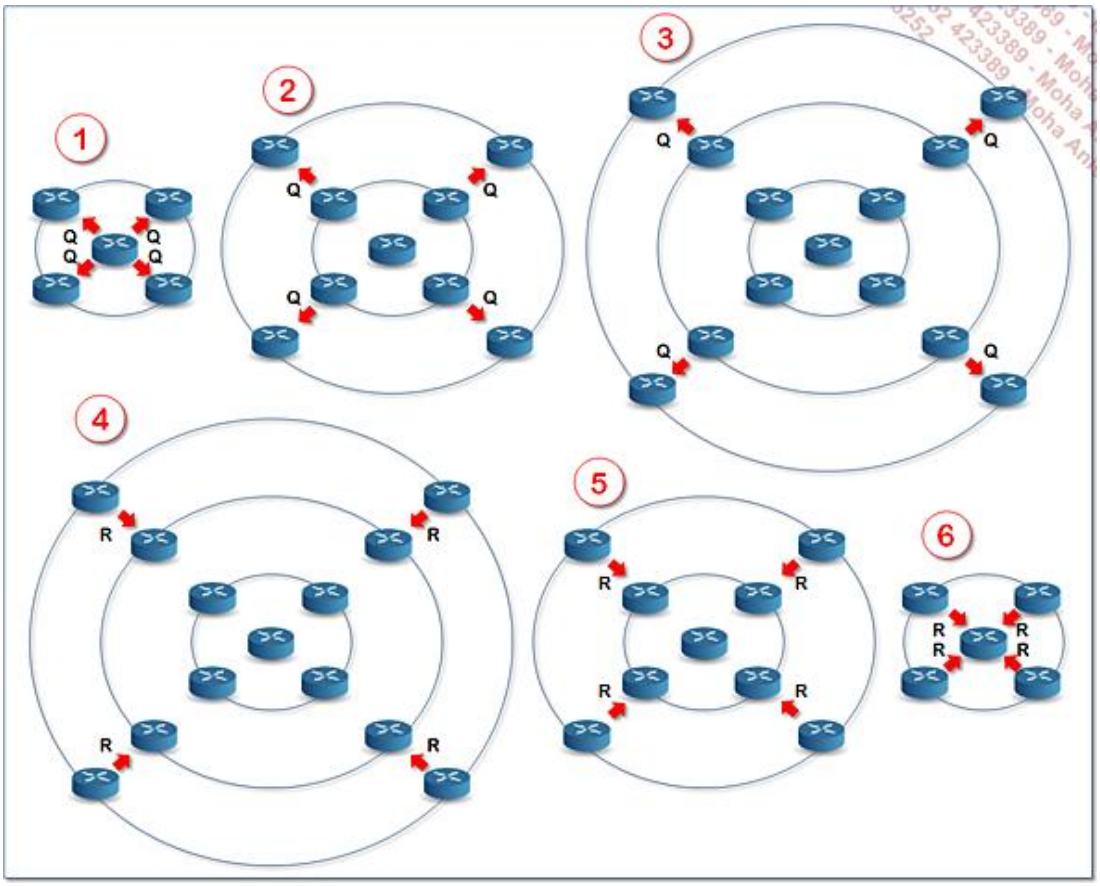
Tant que la procédure de calcul diffusé n'est pas achevée et donc tant que la route n'est pas retournée à l'état passif, les latitudes du routeur au sujet de la route incriminée sont des plus limitées. Il ne peut :

- ni changer le successeur de cette route ;
- ni changer la distance annoncée pour cette route ;
- ni changer la distance faisable de cette route ;
- ni déclencher un autre calcul diffusé pour cette route.

La phase de calcul diffusé débute par une requête envoyée à chacun des voisins. La requête porte la nouvelle distance résultat de la phase locale de réévaluation. Chaque voisin qui reçoit la requête débute à son tour une phase locale de réévaluation. Comme pour le routeur d'origine, le résultat est binaire : oui ou non, le voisin a-t-il trouvé un successeur faisable pour le réseau de destination ?

- Oui → le voisin répond à la requête qui a déclenché chez lui cette phase de réévaluation. La réponse contient le meilleur coût connu du voisin pour rejoindre la destination.
- Non → à son tour, le voisin fait passer la route dans l'état actif et entame une procédure de calcul diffusé.

La figure suivante tente d'illustrer le phénomène à l'échelle d'un réseau :



Ainsi, la diffusion du calcul prend de l'ampleur dans le réseau au fur et à mesure que des requêtes sont envoyées mais voit son ampleur diminuer au fur et à mesure que les réponses sont reçues.

Le routeur émetteur de toute requête envoyée à un voisin positionne dans le même temps un drapeau, appelons le drapeau de réponse « r » afin de se souvenir que cette requête est encore en attente de réponse. En première approche, la procédure de calcul diffusé prend fin quand le routeur n'a plus aucun drapeau de réponse positionné pour la route objet de la procédure, c'est-à-dire quand le routeur a reçu une réponse à chaque question posée.

Il peut arriver qu'une question reste sans réponse. Les raisons sont à chercher dans des réseaux physiques de qualité insuffisante ou dans une architecture inadaptée (patientez). Pour se prémunir de questions qui resteraient sans réponse, le routeur qui entame une procédure de calcul diffusé arme dans le même temps un temporisateur « Route active », au temps initial de 3 minutes. Si le temporisateur expire avant que toutes les requêtes n'aient reçu leurs réponses, la route est déclarée SIA (*Stuck-In-Active*, littéralement « coincée dans l'état actif »). Le jugement est sans appel pour le ou les voisins qui n'ont pas répondu : ils sont retirés de la table de voisinage (ils ne sont donc plus voisins) et la procédure de calcul diffusé considère la non-réponse comme équivalent à une réponse avec métrique infinie.

La perte d'un voisin qui résulte d'une non-réponse est un événement grave, on devine que les conséquences peuvent être funestes et l'objectif de l'administrateur est évidemment d'éviter coûte que coûte les routes SIA, au besoin en réexaminant l'architecture de son réseau. C'est pourquoi une section dédiée leur est consacrée.

En seconde approche, cette fois définitive, la procédure de calcul diffusé prend fin quand pour chaque question, une réponse a été reçue ou une non-réponse a été prise en compte suite à l'expiration du temporisateur « Route active ». Cet achèvement se traduit par :

- Le retour à l'état passif de la route.
- Le positionnement de FD à la valeur infinie par le routeur à l'origine de la procédure, ce afin que tout voisin ayant répondu avec une métrique finie puisse satisfaire la condition de faisabilité et devenir successeur faisable. Parmi ces successeurs faisables, le routeur choisit le successeur, c'est celui qui permet de rejoindre la destination avec la meilleure métrique. La valeur de FD est à nouveau modifiée et le statut des successeurs potentiels est réévalué, ceux qui ne satisfont plus la condition de faisabilité sont rétrogradés voisins simples.

e. L'automate à états finis de DUAL

Régir le fonctionnement de DUAL tel qu'il vient d'être décrit est le fait d'une machine à états finis (FSM, *Finite State Machine*). Une machine FSM est une machine abstraite que les informaticiens utilisent lorsqu'il faut décrire puis gérer

un fonctionnement constitué d'états et de transitions entre états, le franchissement des transitions étant provoqué par des évènements. La machine FSM est également l'automate des automaticiens. Quand l'état N est vrai, si l'évènement associé à la transition de N vers N+1 se produit, alors l'automate passe à l'état N+1.

Exemple : l'ascenseur est à l'arrêt, j'appuie sur le bouton Monter, l'ascenseur quitte l'état Arrêt et passe dans un état Montée.

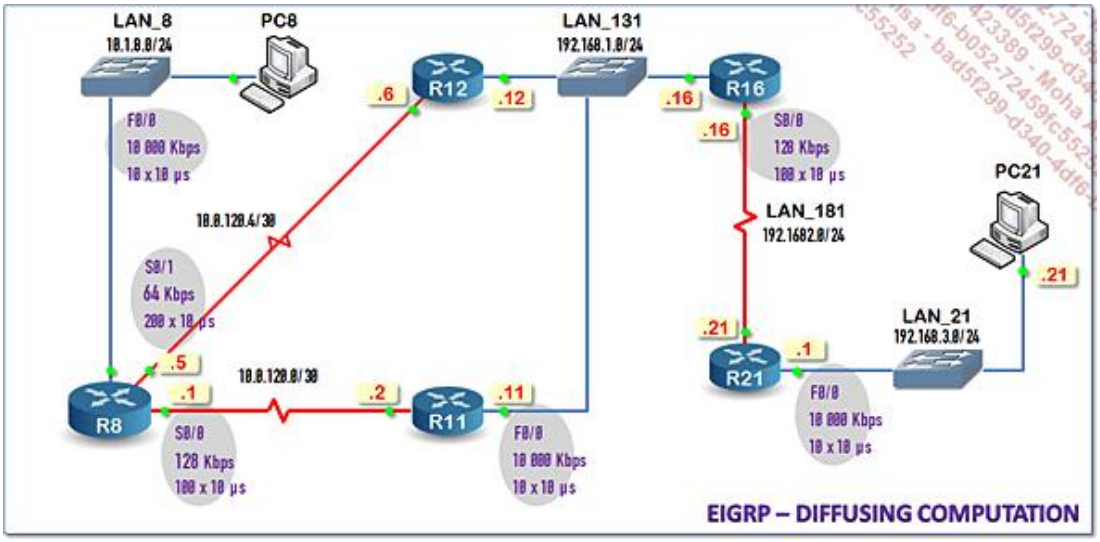
Avant le franchissement de la transition, l'état N était vrai. Après son franchissement, l'état N n'est plus vrai mais l'état N+1 l'est. Le mode de représentation de l'automate diffère selon que l'on est informaticien ou automaticien. Le premier utilise un diagramme d'états. Il s'agit d'un graphe orienté dont les états sont les sommets et les transitions les arêtes étiquetées. Le second (l'automaticien) utilise un graphe Étapes/Transitions appelé Grafcet (traité en annexe).

Il n'y a pas d'études de protocoles sans étude de diagrammes d'états. Par exemple, l'ouvrage Cisco - Notions de base sur les réseaux dans la collection Certifications aux Editions ENI avait illustré le fonctionnement du client DHCP à l'aide de l'un de ces diagrammes. Le chapitre dédié à OSPF dans cet ouvrage offre l'occasion d'étudier plusieurs machines à états finis pour lesquels l'auteur a préféré utiliser un mode de représentation inspiré du Grafcet.

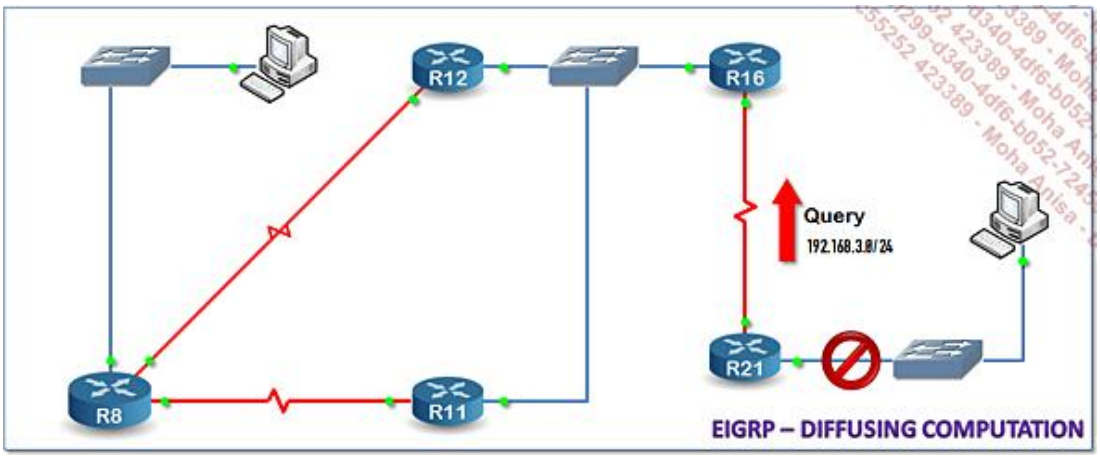
Ce qui est possible pour OSPF parce qu'il s'agit d'un standard de l'Internet publié dans un RFC ne l'est plus pour EIGRP qui est, rappelons-le, un protocole propriétaire. La très volumineuse documentation de CISCO fournie autour d'EIGRP évoque la machine DUAL FSM à de nombreuses reprises sans jamais fournir les détails d'implémentation.

Que le lecteur se rassure, nous sommes, une fois de plus, largement au-delà des attendus de la certification CCNA.

Tentons d'en bâtir une représentation en traitant l'exemple ci-dessous :



Simulons une défaillance du réseau LAN_21 à l'extrémité droite du réseau en provoquant un **shutdown** sur l'interface F0/0 de R21. R21 entame sa phase locale de réévaluation, ne trouve pas de successeur faisable pour LAN_21 et poursuit avec une procédure de calcul diffusé. La route passe à l'état actif et R21 envoie une requête à R16 :

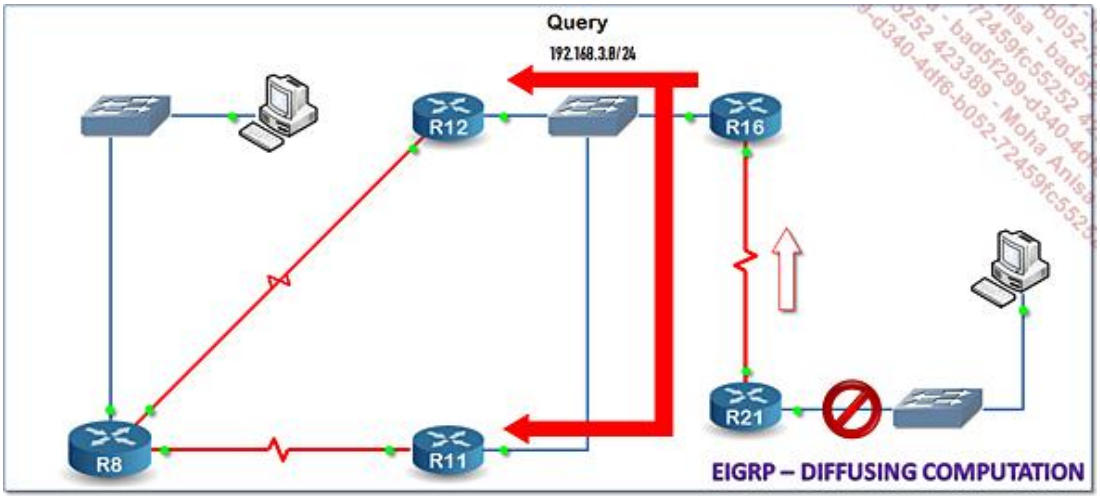


L'activité de l'automate DUAL FSM peut être visualisée à l'aide d'une commande **debug eigrp fsm** :

```
R21#debug eigrp fsm
EIGRP FSM Events/Actions debugging is on
R21#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
```

```
R21(config)#int f0/0
R21(config-if)#shutdown
R21(config-if)#
00:07:39: DUAL: rcvupdate: 192.168.3.0/24 via Connected metric
4294967295/4294967295
00:07:39: DUAL: Find FS for dest 192.168.3.0/24. FD is 258560, RD is 258560
00:07:39: DUAL:         0.0.0.0 metric 4294967295/4294967295 not found Dmin is
4294967295
00:07:39: DUAL: Dest 192.168.3.0/24 entering active state.
00:07:39: DUAL: Set reply-status table. Count is 1.
00:07:39: DUAL: Not doing split horizon
00:07:41: %LINK-5-CHANGED: Interface FastEthernet0/0, changed state to
administratively down
00:07:42: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0,
changed state to down
00:07:44: DUAL: dest(192.168.3.0/24) active
```

Le routeur R16 reçoit la requête de R21. Cette requête prend un sens particulier parce qu'elle émane du successeur connu de R16 pour la route LAN_21. Cet évènement provoque chez R16 l'exécution de la phase de réévaluation locale, R16 cherche un successeur faisable pour LAN_21. R16 n'en trouve pas, marque la route comme étant injoignable, fait passer la route à l'état actif puis envoie une requête sur ses deux voisins R11 et R12 :



L'activité correspondante observée à l'aide d'une commande **debug eigrp fsm** sur R16 :

```
R16#debug eigrp fsm
EIGRP FSM Events/Actions debugging is on
00:07:19: DUAL: rcvquery: 192.168.3.0/24 via 192.168.2.21 metric
4294967295/4294967295, RD is 20028160
00:07:19: DUAL: Find FS for dest 192.168.3.0/24. FD is 20028160, RD is 20028160
00:07:19: DUAL:         192.168.2.21 metric 4294967295/4294967295 not foundDmin
is4294967295
00:07:19: DUAL: Dest 192.168.3.0/24 entering active state.
```

De la même façon, les routeurs R11 et R12 font le constat que la seule route faisable vers LAN_21 est perdue, marquent la route comme étant injoignable puis envoient tous deux une requête au routeur R8. Une commande **debug eigrp fsm** permet de découvrir que la requête issue de R11 est vue en premier. En réalité, l'ordre importe peu, le résultat final sera le même :

```
R8#
03:01:48: DUAL: rcvquery: 192.168.3.0/24 via 10.0.120.2 metric 4294967295/4294967295,
RD is 20056320
03:01:48: DUAL: Find FS for dest 192.168.3.0/24. FD is 20056320, RD is 20056320
03:01:48: DUAL:         10.0.120.2 metric 4294967295/4294967295
03:01:48: DUAL:         10.0.120.6 metric 40056320/20030720 found Dmin is 40056320
03:01:48: DUAL: send REPLY(r1/n1) about 192.168.3.0/24 to 10.0.120.2
03:01:48: DUAL: RT installed 192.168.3.0/24 via 10.0.120.6
03:01:48: DUAL: Send update about 192.168.3.0/24.  Reason: metric chg
03:01:48: DUAL: Send update about 192.168.3.0/24.  Reason: new if
```

La capture ci-dessus nécessite un peu de concentration. Que s'est-il passé ?

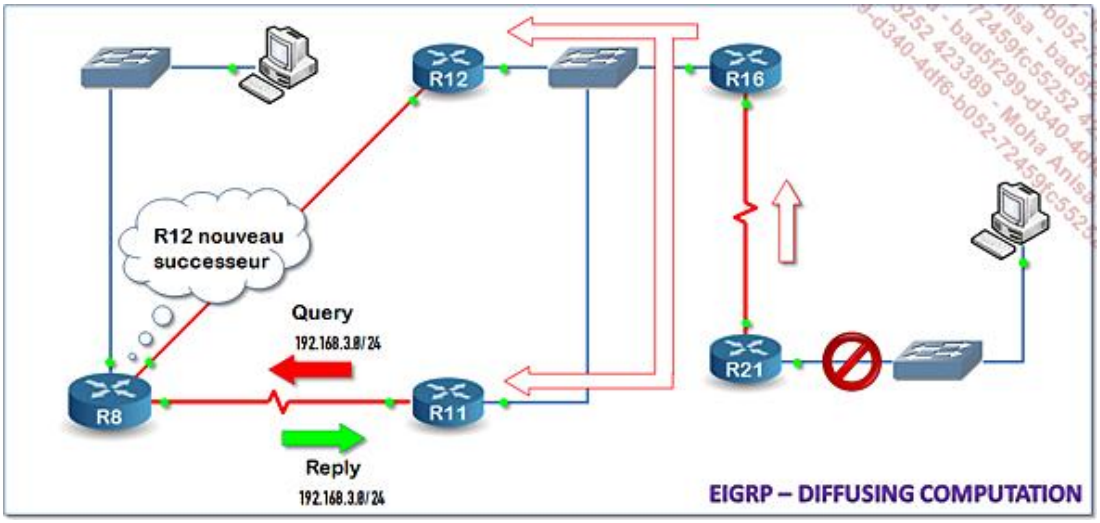
- Message 1 → R8 reçoit la requête du voisin R11 qui se trouve être également le successeur. Cet évènement provoque chez R8 l'exécution de la phase de réévaluation locale, R8 cherche un successeur faisable pour LAN_21.

- Messages 2, 3 et 4 → R8 trouve dans sa table de topologie que R12 est un successeur potentiel (sa distance rapportée est 20030720 effectivement inférieure à la distance faisable 20056320).

- Message 5 → R8 répond à la requête de R11 en annonçant la distance faisable via R12.

- Message 6 → R8 installe le nouveau successeur dans la table de routage.

- Messages 7 et 8 → R8 informe ses voisins du changement de coût à l'aide de messages de mises à jour.



Presque aussitôt parvient la requête de R12 :

```
R8#
03:01:48: DUAL: rcvquery: 192.168.3.0/24 via 10.0.120.6 metric 4294967295/4294967295,
RD is 40056320
03:01:48: DUAL: Find FS for dest 192.168.3.0/24. FD is 20056320, RD is 40056320
03:01:48: DUAL:         10.0.120.6 metric 4294967295/4294967295
03:01:48: DUAL:         10.0.120.2 metric 4294967295/4294967295 not found Dmin is
4294967295
03:01:48: DUAL: Dest 192.168.3.0/24 entering active state.
03:01:48: DUAL: Set reply-status table. Count is 2.
03:01:48: DUAL: Not doing split horizon
03:01:48: DUAL: Going from state 1 to state 3
03:01:48: DUAL: rcvreply: 192.168.3.0/24 via 10.0.120.6 metric 4294967295/4294967295
03:01:48: DUAL: reply count is 2
03:01:48: DUAL: Clearing handle 0, count now 1
03:01:48: DUAL: rcvreply: 192.168.3.0/24 via 10.0.120.2 metric 4294967295/4294967295
03:01:48: DUAL: reply count is 1
03:01:48: DUAL: Clearing handle 1, count now 0
03:01:48: DUAL: Freeing reply status table
03:01:48: DUAL: Find FS for dest 192.168.3.0/24. FD is 4294967295, Rdis4294967295
found
03:01:48: DUAL: send REPLY(r1/n1) about 192.168.3.0/24 to 10.0.120.6
03:01:48: DUAL: Removing dest 192.168.3.0/24, nexthop 10.0.120.2
03:01:48: DUAL: Going from state 3 to state 1
03:01:48: DUAL: Removing dest 192.168.3.0/24, nexthop 10.0.120.6
03:01:48: DUAL: No routes.  Flushing dest 192.168.3.0/24
R8#
```

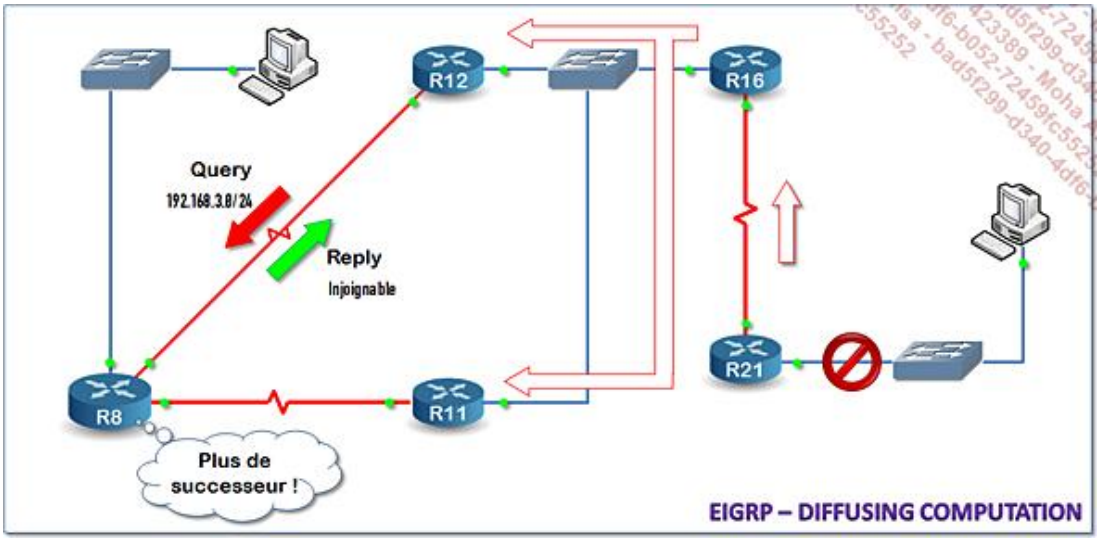
- Message 1 : R8 reçoit la requête du voisin R12. Jusqu'à présent, la route était restée dans l'état passif. Cet évènement provoque une nouvelle exécution de la phase de réévaluation locale.

- Messages 2 à 4 : R12 était le successeur, R8 cherche un nouveau successeur faisable sans succès.

- Message 5 : la route vers LAN_21 passe à l'état actif. R8 envoie une requête à ses voisins à la recherche d'une alternative pour LAN_21.

- Messages 9 et 12 : pour chaque question posée, R8 a reçu une réponse LAN_21 injoignable. R8 fait retourner la route LAN_21 à l'état passif et met à jour la table de routage.

- Message 17 : R8 répond à la question posée par R12 (message 1).

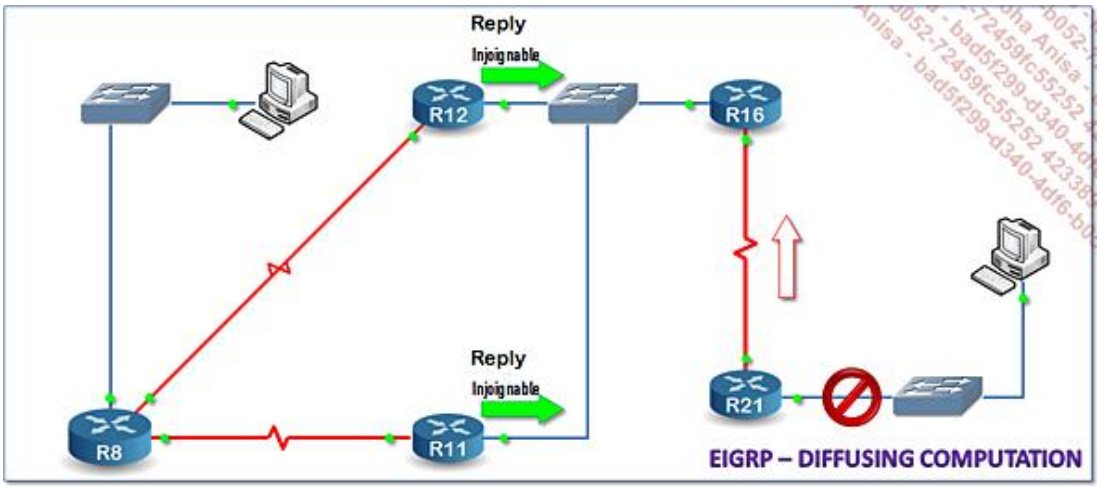


Chacun des deux routeurs R11 et R12 répond à R16 puis fait retourner la route à l'état passif. Par exemple pour R11 :

```
R11#debug eigrp fsm
EIGRP FSM Events/Actions debugging is on
00:07:25: DUAL: rcvreply: 192.168.3.0/24 via 10.0.120.1 metric 4294967295/4294967295
00:07:25: DUAL: reply count is 1
00:07:25: DUAL: Clearing handle 0, count now 0
00:07:25: DUAL: Freeing reply status table
00:07:25: DUAL: Find FS for dest 192.168.3.0/24. FD is 4294967295, RD is 4294967295
found
00:07:25: DUAL: send REPLY(r1/n1) about 192.168.3.0/24 to 192.168.1.16
00:07:25: DUAL: Removing dest 192.168.3.0/24, nexthop 192.168.1.12
00:07:25: DUAL: Removing dest 192.168.3.0/24, nexthop 10.0.120.1
00:07:25: DUAL: Going from state 3 to state 1
00:07:25: DUAL: Removing dest 192.168.3.0/24, nexthop 192.168.1.16
00:07:25: DUAL: No routes.  Flushing dest 192.168.3.0/24
R11#
```

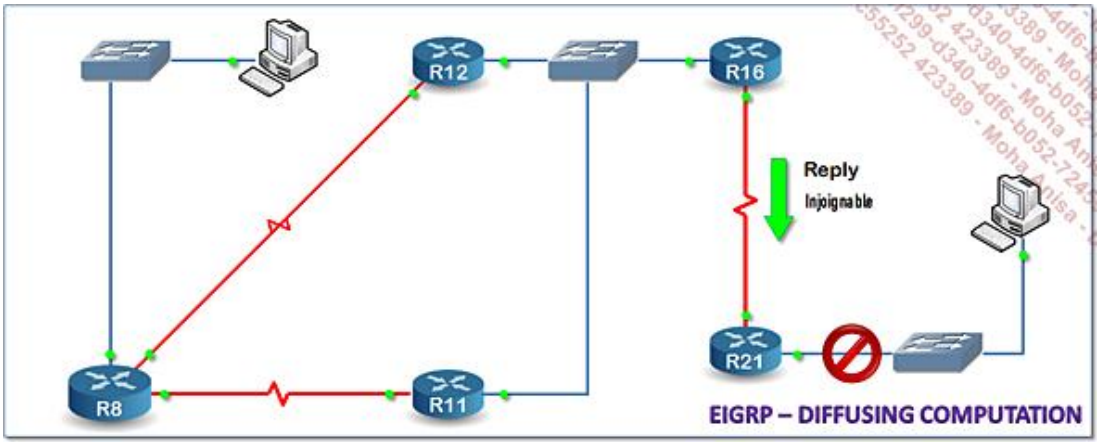
Pour R12 :

```
R12#
03:21:25: DUAL: rcvreply: 192.168.3.0/24 via 10.0.120.5 metric4294967295/4294967295
03:21:25: DUAL: reply count is 1
03:21:25: DUAL: Clearing handle 0, count now 0
03:21:25: DUAL: Freeing reply status table
03:21:25: DUAL: Find FS for dest 192.168.3.0/24. FD is 4294967295, RD is 4294967295
found
03:21:25: DUAL: send REPLY(r1/n1) about 192.168.3.0/24 to 192.168.1.16
03:21:25: DUAL: Removing dest 192.168.3.0/24, nexthop 10.0.120.5
03:21:25: DUAL: Going from state 3 to state 1
03:21:26: DUAL: Removing dest 192.168.3.0/24, nexthop 192.168.1.16
03:21:26: DUAL: No routes.  Flushing dest 192.168.3.0/24
R12#
```



On remonte progressivement vers le routeur à l'origine de la procédure de calcul diffusé :

```
R16#debug eigrp fsm
.........
00:07:19: DUAL: Set reply-status table. Count is 2.
00:07:19: DUAL: Doing split horizon on Serial0/0
00:07:19: DUAL: Going from state 1 to state 3
00:07:21: DUAL: dest(192.168.3.0/24) active
00:07:21: DUAL: rcvreply: 192.168.3.0/24 via 192.168.1.12 metric
4294967295/4294967295
00:07:21: DUAL: reply count is 2
00:07:21: DUAL: Clearing handle 2, count now 1
00:07:21: DUAL: Removing dest 192.168.3.0/24, nexthop 192.168.1.12
00:07:21: DUAL: dest(192.168.3.0/24) active
00:07:21: DUAL: rcvquery: 192.168.3.0/24 via 192.168.1.12 metric
4294967295/4294967295, RD is 4294967295
00:07:21: DUAL: send REPLY(r1/n1) about 192.168.3.0/24 to 192.168.1.12
00:07:24: DUAL: dest(192.168.3.0/24) active
00:07:24: DUAL: rcvreply: 192.168.3.0/24 via 192.168.1.11 metric
4294967295/4294967295
00:07:24: DUAL: reply count is 1
00:07:24: DUAL: Clearing handle 0, count now 0
00:07:24: DUAL: Freeing reply status table
00:07:24: DUAL: Find FS for dest 192.168.3.0/24. FD is 4294967295, RD is 4294967295
found
00:07:24: DUAL: send REPLY(r1/n1) about 192.168.3.0/24 to 192.168.2.21
00:07:24: DUAL: Removing dest 192.168.3.0/24, nexthop 192.168.1.11
00:07:24: DUAL: Removing dest 192.168.3.0/24, nexthop 192.168.1.12
00:07:24: DUAL: Going from state 3 to state 1
00:07:24: DUAL: Removing dest 192.168.3.0/24, nexthop 192.168.2.21
00:07:24: DUAL: No routes.  Flushing dest 192.168.3.0/24
R16#
```



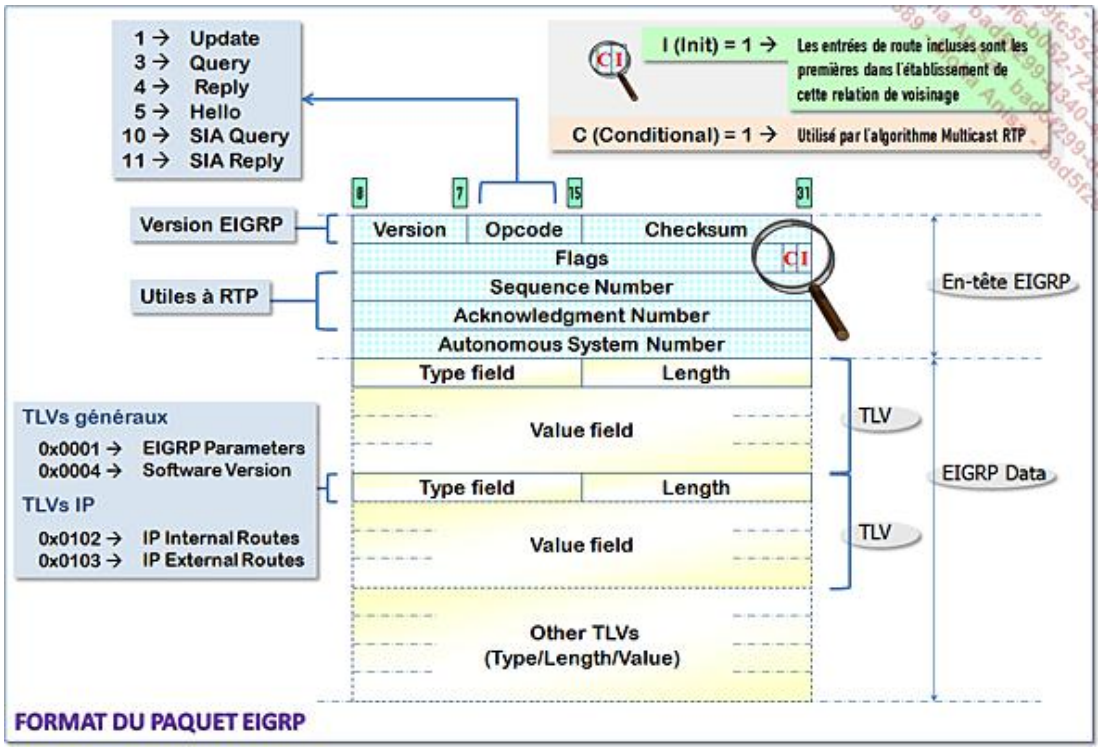
Pour enfin achever la procédure sur le routeur qui l'a initié :

```
R21#debug eigrp fsm
..........! La suite
00:07:44: DUAL: rcvreply: 192.168.3.0/24 via 192.168.2.16 metric
4294967295/4294967295
00:07:44: DUAL: reply count is 1
00:07:44: DUAL: Clearing handle 0, count now 0
00:07:44: DUAL: Freeing reply status table
00:07:44: DUAL: Find FS for dest 192.168.3.0/24. FD is 4294967295, RD is 4294967295
found
00:07:44: DUAL: Removing dest 192.168.3.0/24, nexthop 0.0.0.0
00:07:44: DUAL: Removing dest 192.168.3.0/24, nexthop 192.168.2.16
00:07:44: DUAL: No routes.  Flushing dest 192.168.3.0/24
R21(config-if)#
```

Observez que chacun des routeurs de la topologie a traité au moins une requête pour le réseau LAN_21. Reproduire plusieurs fois de suite le même évènement montre que la procédure de calcul diffusé peut emprunter des voies différentes selon qu'une requête arrive avant l'autre ou selon l'ordre de traitement par le routeur (particulièrement vrai pour R8 dans le contexte ci-dessus). Mais le résultat final, à savoir la convergence du réseau, est toujours obtenu très rapidement. Un évènement qui provoque une procédure de calcul diffusé, c'est un peu comme la mèche qui provoque la déflagration.

9. Format des paquets EIGRP

Puisque EIGRP dispose de son propre protocole de transport, il s'encapsule directement dans IP, identifié par le numéro de protocole 88. La longueur du paquet EIGRP est donc limitée à la capacité d'emport du datagramme IP, soit le MTU diminué de l'en-tête IP. L'adresse de destination IP est selon les cas (relire la section Le protocole de transport RTP) l'adresse multicast 224.0.0.10 qui identifie le groupe des routeurs EIGRP ou l'adresse unicast d'un routeur. L'adresse source est toujours l'adresse de l'interface qui a servi à émettre le paquet. L'en-tête IP est suivi d'un en-tête EIGRP lui-même suivi d'un certain nombre de triplets TLVs (*Type/Length/Value*). Les TLVs transportent les informations de routes mais aussi des paramètres utiles à la gestion du protocole et notamment de DUAL. Le format général du paquet est le suivant :



- « *Version* » : EIGRP a connu deux révisions majeures, toutes les versions d'IOS au-delà de 11.1(3) disposent de sa version la plus récente. Le numéro de version, 2 dans les captures d'EIGRP observées avec Wireshark dans ce chapitre, identifie une version particulière du processus EIGRP. L'administrateur peut obtenir en savoir à l'aide d'une commande **sh ip eigrp neighbor detail** :

```
R8#sh ip eigrp neighbors detail
IP-EIGRP neighbors for process 64501
```

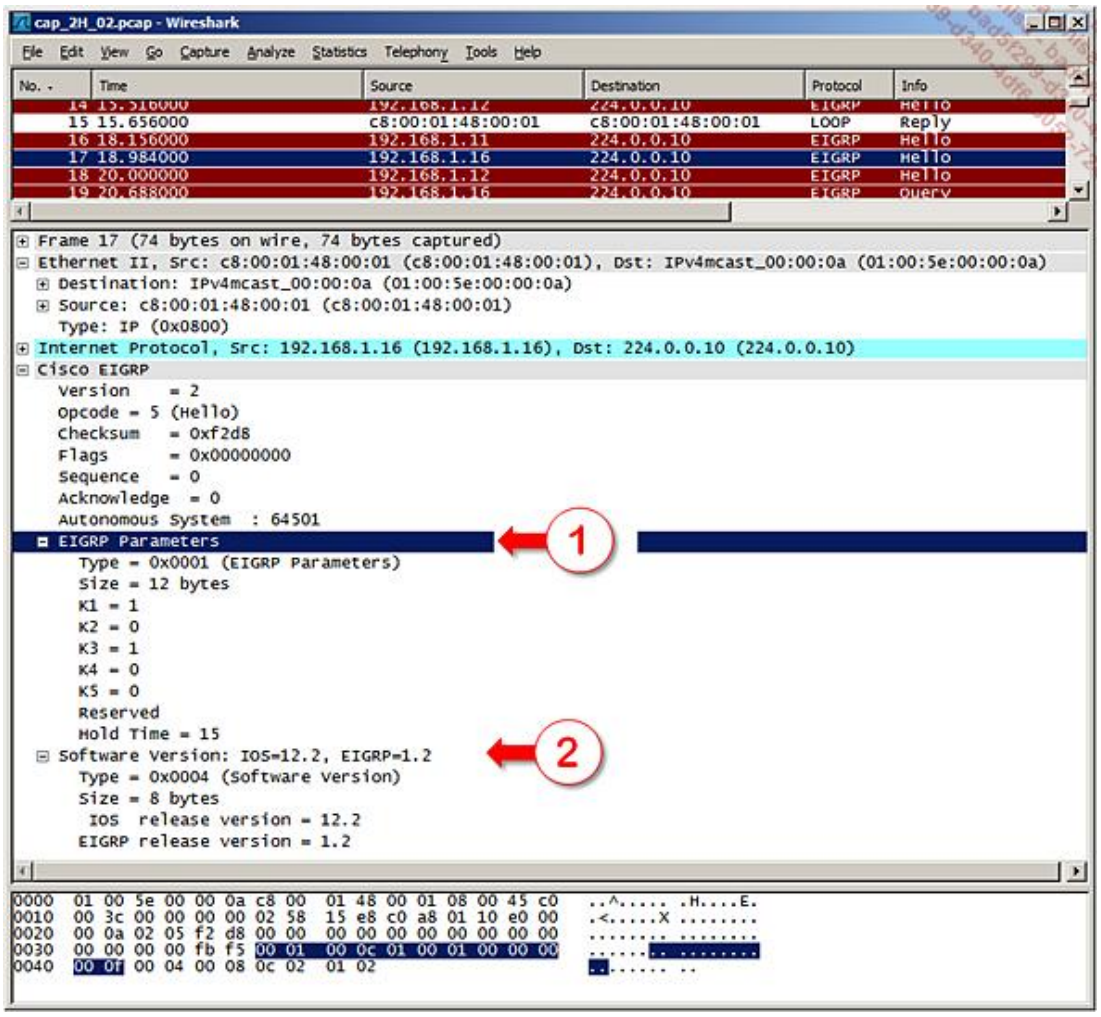
```
H   Address                 Interface   Hold Uptime   SRTT   RTO  Q  Seq Type
                                        (sec)         (ms)       Cnt Num
1   10.0.120.2              Se0/0         12 03:22:16   66  1140  0  110
Version 12.2/1.2, Retrans: 1, Retries: 0
0   10.0.120.6              Se0/1         10 03:22:57   52  2280  0  121
Version 12.2/1.2, Retrans: 0, Retries: 0
R8#
```

- « *Opcode* » : identifie le type de paquet EIGRP, « *update* », « *query* », « *reply* », « *Hello* »...
- « *Checksum* » : cette somme de contrôle s'applique à la totalité du paquet EIGRP, en-tête IP exclue.
- « *Flags* » : seuls deux bits sont utilisés. Le bit de poids faible « I » (Init) est utilisé lors de l'établissement d'une nouvelle relation de voisinage (relire la section Entretien des relations de voisinage). Le bit de poids 2 « *Conditional Receive* » est utilisé par l'algorithme de multidiffusion fiable (messages multidiffusés, acquittements unicast) de RTP.
- « *Seq Number* », « *Ack Number* » : utiles au mécanisme de fiabilité procuré par RTP. Un message fiable de RTP qui porte le numéro de séquence N doit être acquitté par un message d'acquittement dont le numéro d'acquittement est N. Pour vous en convaincre, observez si nécessaire les trames n°19, 20 et 21 de la capture Wireshark cap_2H_02.pcap disponible en téléchargement. Cette capture a été réalisée sur le LAN_131 qui connecte R11, R12 et R16 dans la topologie précédente (mise en œuvre pour illustrer le fonctionnement de DUAL). En trame 19, R16 émet un message multicast dont le numéro de séquence est 18, numéro d'acquittement 0. En trame 20, R11 acquitte le message, le numéro de séquence est 0, le numéro d'acquittement est 18. En trame 21, c'est au tour de R12 d'acquitter le message, le numéro d'acquittement est 18. Observez que les messages d'acquittement sont en réalité des messages Hello (Opcode = 5) mais que le protocole n'a aucune difficulté à distinguer car un véritable message Hello est toujours diffusé et par conséquent, son champ numéro d'acquittement est toujours 0.
- « *Autonomous System Number* » : identifie le domaine EIGRP. Un processus EIGRP identifié par un numéro d'AS n'exploite un paquet EIGRP que s'il porte le même numéro d'AS.

Le corps du message EIGRP est constitué de un ou plusieurs triplets TLVs. Chaque TLV débute par un champ type exprimé sur deux octets suivi d'un champ longueur également sur deux octets. La longueur en question est celle du TLV, champs type et longueur inclus. Le champ valeur qui suit type et longueur est fonction du type du TLV. Nous n'examinerons évidemment pas tous les types de TLVs, ne serait-ce que parce que les TLVs spécifiques à IPX ou AppleTalk ont perdu de leur intérêt avec la prédominance d'IP.

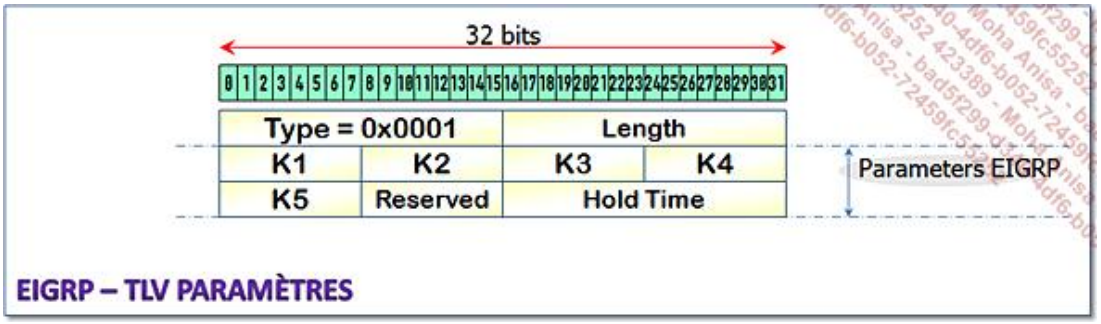
a. TLVs généraux

Observez l'extrait de capture Wireshark ci-après :



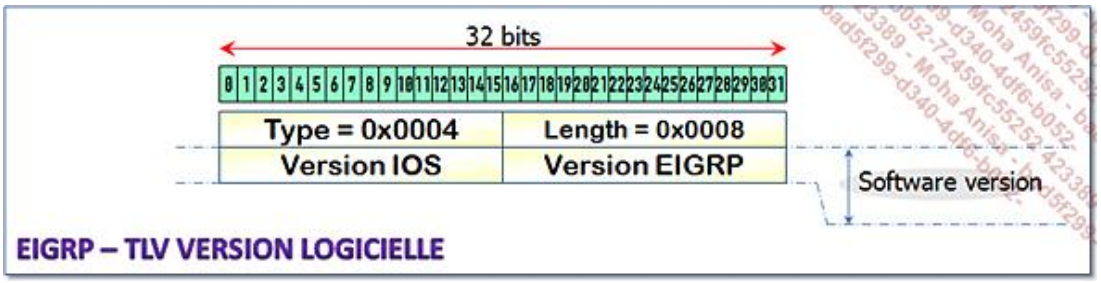
Ainsi les messages Hello d'EIGRP embarquent de façon systématique deux TLVs dits généraux c'est-à-dire utiles à la gestion du protocole EIGRP proprement dit :

- Le TLV de paramètres dont le format est le suivant :



Le lecteur reconnaîtra sans difficulté les coefficients k1 à k5 utilisés par EIGRP pour calculer les distances des routes, ainsi que la valeur « *Hold Time* » utile pour surveiller les relations de voisinage :

- Le TLV Version de logiciel dont le format est le suivant :



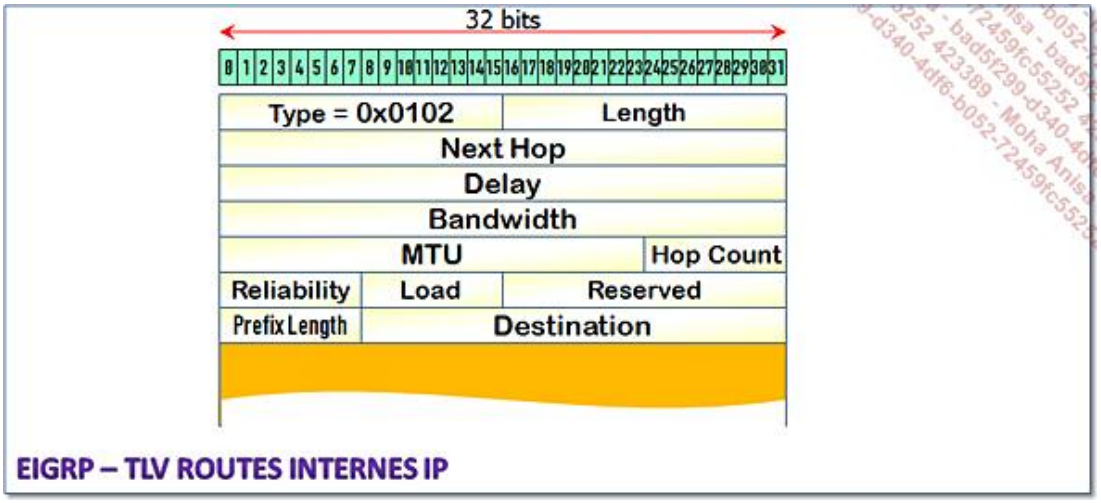
b. TLVs IP

Deux types de TLVs sont dédiés au transport d'information de routes :

- Le TLV Route interne contient une destination apprise à l'intérieur du système autonome EIGRP.
- Le TLV Route externe contient une route redistribuée (remplacez par récupérée) dans le système autonome EIGRP et issue d'un autre processus de routage.

Chaque TLV IP contient une entrée de route. Chaque message EIGRP parmi les messages de mise à jour, de requête et de réponse contient au moins un TLV IP.

Le format du TLV Route interne est le suivant :



... dont les champs sont :

- « *Next Hop* » : le plus souvent le routeur de prochain saut est le routeur dont est issue l'information de route, l'information « *Next Hop* » est dans ce cas inutile et le champ reste à 0.0.0.0. Il peut arriver dans des circonstances particulières que le routeur de prochain saut ne puisse s'annoncer lui-même. Le champ « *Next Hop* » vient alors à point nommé pour indiquer à quelle adresse IP il convient de remettre directement les paquets sans passer par le routeur à l'origine de l'annonce EIGRP. L'adresse de saut suivant est une adresse directement connectée au sous-réseau sur lequel est effectuée l'annonce. 0000 dans le champ Next Hop indique que le routage doit se faire via le routeur à l'origine de l'annonce EIGRP.
- « *Delay* » : somme des délais de toutes les interfaces entrantes placées entre le réseau de destination et le routeur qui annonce, exprimée en dizaines de microsecondes. Ce champ occupait 24 bits pour le protocole parent IGRP et en occupe 32 dans la version EIGRP suite à l'ajout du facteur 256.

Un délai 0xFFFFFFFF indique une route injoignable !

- « *Bandwidth* » : bande passante la plus faible parmi les bandes passantes de toutes les interfaces entrantes placées entre le réseau de destination et le routeur qui annonce. Ce champ occupait 24 bits pour le protocole parent IGRP et en occupe 32 dans la version EIGRP suite à l'ajout du facteur 256. EIGRP y place le résultat du calcul suivant :

$$BWannoncée = 256 \times \frac{10^7}{BWmin(Kbps)}$$

- « *MTU* » : MTU minimum du chemin, parmi les MTU des différents tronçons qui composent le chemin. On se souvient que le MTU (*Maximum Transmission Unit*) est la capacité d'emport de la trame en couche 2, ou ce qui est équivalent, la longueur maximale du datagramme IP. Bien sûr, cette information ne participe pas au calcul de la métrique.

- « *Hop Count* » : nombre de sauts pour rejoindre le réseau de destination. Le routeur auquel le réseau est directement connecté annonce 0, chaque routeur subséquent incrémente la valeur et annonce la valeur incrémentée.

- « *Reliability* » : fiabilité. Pour chaque interface, l'IOS entretient des compteurs de paquets émis et reçus ainsi que des compteurs de paquets en erreur. Ces comptages lui permettent d'entretenir de façon dynamique un indice de fiabilité. Cet indice est rangé dans un champ de 8 bits, il n'est donc pas exprimé en % mais en « pour 255 ». Aucun paquet en erreur implique un indice de 255, l'indice maximal (détails complémentaires dans la section La métrique d'EIGRP).

- « *Load* » : charge. Les mêmes compteurs de paquets émis et reçus permettent de calculer une bande passante effectivement occupée puis de la comparer à la bande passante disponible pour exprimer un indice de charge. Comme l'indice de fiabilité, cet indice est rangé dans un champ de 8 bits, l'indice 1 indique une charge nulle, l'indice 255 n'est jamais atteint, un indice qui dépasse 200 indique un lien très fortement chargé (détails complémentaires dans la section La métrique d'EIGRP).

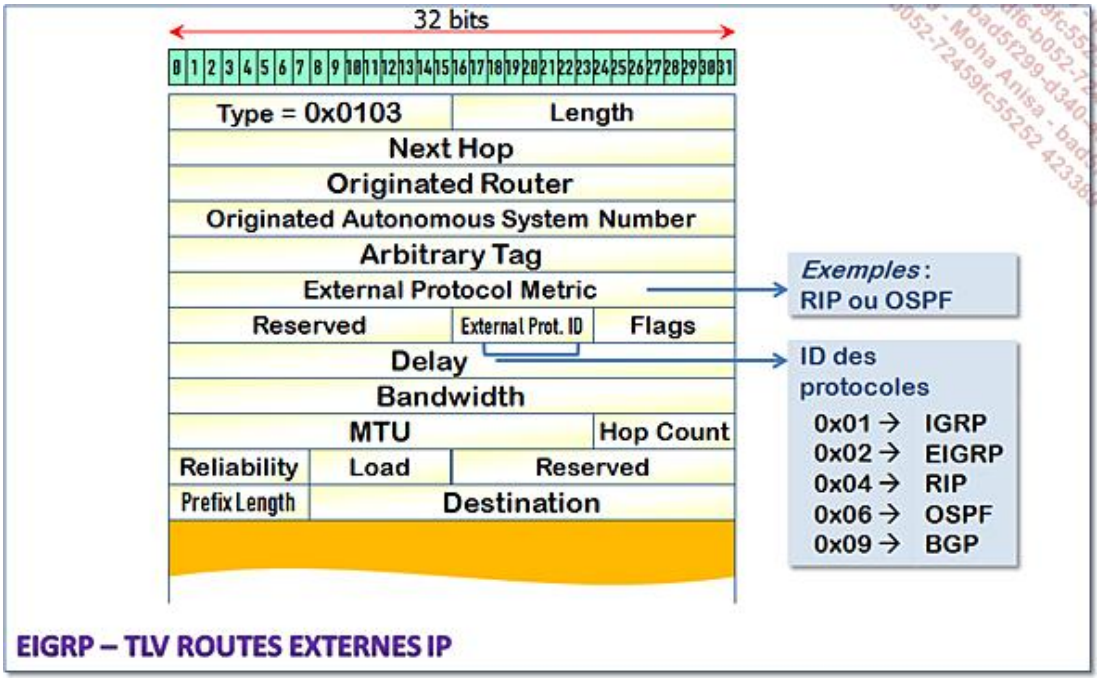
- « *Prefix Length* » : nombre de bits du masque réseau.

- « *Destination* » : adresse de destination de la route. La longueur de ce champ dépend de la valeur contenue dans le champ Longueur de préfixe. Exemples :

    - 10.8.0.0/16 : le champ **Longueur** contient 16, le champ Destination occupe 2 octets qui contiennent 10.8.

    - 172.16.0.96/27 : le champ **Longueur** contient 27, le champ Destination occupe 4 octets parce que l'adresse destination est 172.16.0.96 qui occupe plus de 3 octets. Les bits non utilisés par l'adresse de destination sont laissés à 0. En final, le champ Destination occupe de 1 à 4 octets.

Le format du TLV Route externe est le suivant :



... dont les champs sont :

- « *Next Hop* » : description identique au champ correspondant du TLV Routes internes IP.
- « *Originated Router* » : adresse IP ou identifiant du routeur qui a redistribué (récupéré) la route externe dans ce système autonome EIGRP.
- « *Originated Autonomous System Number* » : numéro de l'AS dont la route est issue.
- « *Arbitrary Tag* » : utile dans la redistribution de routes entre protocoles de routage. La commande **set tag** en mode configuration **route-map** permet de marquer une route récupérée, la commande **match tag** quant à elle, permet de ne récupérer une route que si elle porte le marquage attendu.
- « *External Protocol Metric* » : métrique telle qu'elle était connue du protocole externe pour cette route.
    - Exemple : une route a été apprise depuis le protocole RIP avec un nombre de sauts (la métrique de RIP) égal à 3 puis redistribuée dans cet AS EIGRP : le champ « *Originated Router* » porte l'adresse IP du routeur EIGRP qui a récupéré cette route, le champ « *Originated AS Number* » reste à 0 puisque RIP ne connaît pas cette information, le champ « *External Protocol Metric* » porte la valeur 3 et enfin le champ « *External Protocol ID* » identifie RIP par la valeur 0x04.
- « *Flags* » : CISCO n'utiliserait que deux bits sur les 8 que compte ce champ, le bit de poids 2 identifie une route par défaut candidate.
- Les champs qui suivent sont identiques à ceux déjà décrits pour un TVL Route interne.

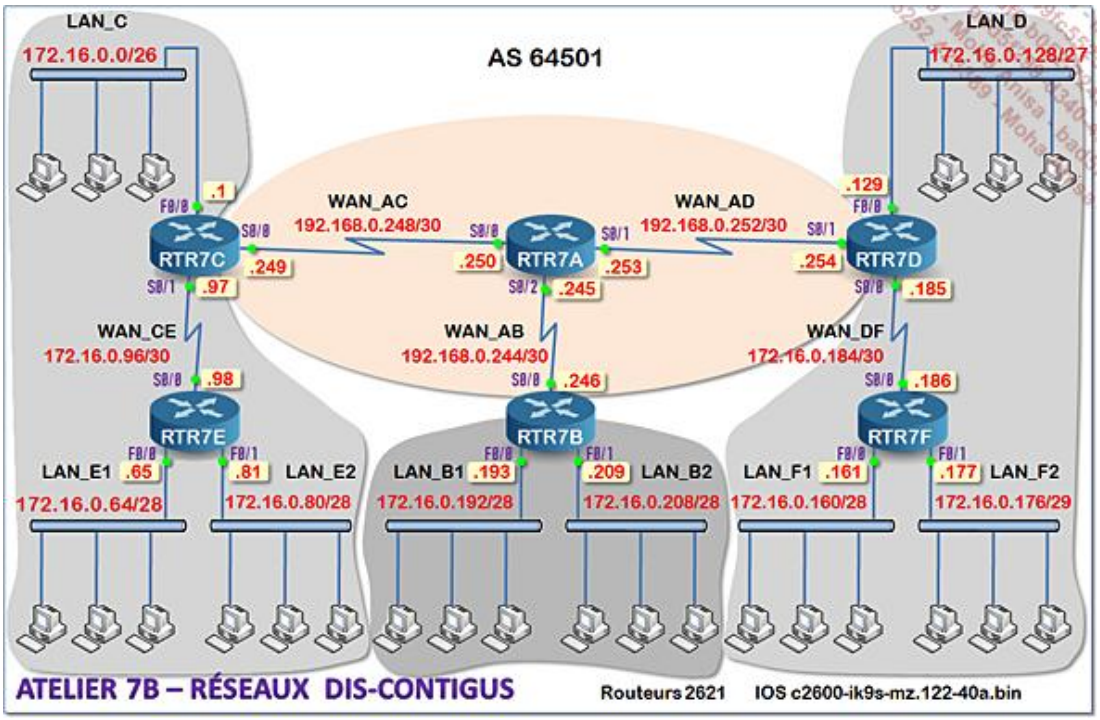
10. Agrégation de routes

Pour mémoire, un protocole de routage sans classe, tel EIGRP, inclut le masque ou la longueur de préfixe dans ses annonces de routes. C'est ce qui a permis l'abandon des classes d'adresses et l'adoption de VLSM. Pour autant, EIGRP peut continuer à se comporter « avec classe », c'est ce qu'il fait lorsque l'administrateur laisse l'agrégation automatique activée, elle l'est par défaut. Ce comportement est rassurant et prudent pour l'ensemble du réseau car il concourt à maintenir des tables de routage « raisonnables ». Pourtant, l'administrateur peut être conduit à désactiver cette agrégation automatique, c'est notamment le cas lorsque plusieurs parties d'un domaine sont fédérées par un réseau de transit, comme illustré dans l'étude de cas ci-après.

La conséquence de la désactivation de l'agrégation automatique est l'inflation du nombre d'entrées dans les tables de routage, inflation que ne pouvait contenir un protocole de routage tel RIPv2 faute d'outils. EIGRP propose fort heureusement un développement intéressant en offrant un mécanisme d'agrégation manuelle.

a. Agrégation automatique de routes

Reprenons la topologie du chapitre précédent qui avait servi à illustrer le comportement de RIP face à des réseaux discontigus. Pour mémoire, nous avions affecté des adresses de l'ex-classe C aux liens WAN issus du routeur RTR7A afin de scinder le domaine couvert par 172.16.0.0 en trois parties :



EIGRP a remplacé RIP sur chacun des routeurs de la topologie qui participent désormais au système autonome n° 64501. Une fois tous les routeurs correctement configurés, une lecture de la table de routage du routeur RTR7A est édifiante : ce routeur connaît trois alternatives à coût égal vers 172.16.0.0 et s'apprête à faire de la répartition de charge. Tout premier paquet d'un flux destiné à 172.16.0.0/24 et traité par RTR7A a une chance sur trois d'être acheminé vers la partie qui convient du domaine couvert par 172.16.0.0 :

```
RTR7A#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is not set

D    172.16.0.0/16 [90/2172416] via 192.168.0.249, 00:00:31, Serial0/0
                   [90/2172416] via 192.168.0.254, 00:00:31, Serial0/1
                   [90/2172416] via 192.168.0.246, 00:00:31, Serial0/2
     192.168.0.0/30 is subnetted, 3 subnets
C       192.168.0.248 is directly connected, Serial0/0
C       192.168.0.252 is directly connected, Serial0/1
C       192.168.0.244 is directly connected, Serial0/2
RTR7A#
```

Ainsi, comme RIP, le comportement par défaut d'EIGRP est d'agréger ses annonces aux frontières des réseaux majeurs. Les trois routeurs RTR7C, RTR7B et RTR7D annoncent 172.16.0.0 vers RTR7A ce, parce que leur interface vers ce routeur n'est pas dans le réseau 172.16.0.0. Rendre la vue à RTR7A implique de désactiver l'agrégation automatique (*summarization*) ce que l'administrateur doit configurer sur les trois routeurs frontière à l'aide de la commande **no auto-summary** en configuration de routeur :

RTR7C

```
RTR7C(config)#router eigrp 64501
RTR7C(config-router)#no auto-summary
RTR7C(config-router)#^Z
RTR7C#wr
Building configuration...
[OK]
RTR7C#
```

RTR7B

```
RTR7B(config)#router eigrp 64501
RTR7B(config-router)#no auto-summary
RTR7B(config-router)#^Z
RTR7B#wr
Building configuration...
```

```
[OK]
RTR7B#
```

RTR7D

```
RTR7B(config)#router eigrp 64501
RTR7B(config-router)#no auto-summary
RTR7B(config-router)#^Z
RTR7B#
RTR7B#wr
Building configuration...
[OK]
RTR7B#
```

Le réglage du paramètre d'agrégation, automatique ou pas, peut être vérifié à l'aide d'une commande **show ip protocols**. Exemple sur RTR7C (extrait) :

```
RTR7C#sh ip prot
Routing Protocol is "eigrp 64501"
.........
  EIGRP metric weight K1=1, K2=0, K3=1, K4=0, K5=0
  EIGRP maximum hopcount 100
  EIGRP maximum metric variance 1
  Redistributing: eigrp 64501
  Automatic network summarization is not in effect
.........
  Distance: internal 90 external 170

RTR7C#
```

Une commande **show ip route** sur RTR7A rassure immédiatement l'administrateur. En effet, le domaine couvert par 172.16.0.0 doit comporter que 10 sous-réseaux, la commande nous avertit que RTR7A en connaît 10 :

```
RTR7A#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is not set

     172.16.0.0/16 is variably subnetted, 10 subnets, 5 masks
D       172.16.0.184/30 [90/2681856] via 192.168.0.254, 00:06:11, Serial0/1
D       172.16.0.176/29 [90/2684416] via 192.168.0.254, 00:06:07, Serial0/1
D       172.16.0.160/28 [90/2684416] via 192.168.0.254, 00:06:07, Serial0/1
D       172.16.0.128/27 [90/2172416] via 192.168.0.254, 00:06:11, Serial0/1
D       172.16.0.208/28 [90/2172416] via 192.168.0.246, 00:04:09, Serial0/2
D       172.16.0.192/28 [90/2172416] via 192.168.0.246, 00:06:37, Serial0/2
D       172.16.0.0/26 [90/2172416] via 192.168.0.249, 00:00:28, Serial0/0
D       172.16.0.96/30 [90/2681856] via 192.168.0.249, 00:00:28, Serial0/0
D       172.16.0.80/28 [90/2684416] via 192.168.0.249, 00:00:28, Serial0/0
D       172.16.0.64/28 [90/2684416] via 192.168.0.249, 00:00:28, Serial0/0
     192.168.0.0/30 is subnetted, 3 subnets
C       192.168.0.248 is directly connected, Serial0/0
C       192.168.0.252 is directly connected, Serial0/1
C       192.168.0.244 is directly connected, Serial0/2
RTR7A#
```

b. Agrégation manuelle de routes

Les besoins d'agrégation amènent à préciser les notions de réseau de transit et réseau d'extrémité :

Réseau de transit

On reconnaît un réseau de transit au fait qu'aucune des adresses source et destination des paquets observés sur ce réseau n'appartient à ce réseau. Les paquets ne font que « passer » sur un réseau de transit.

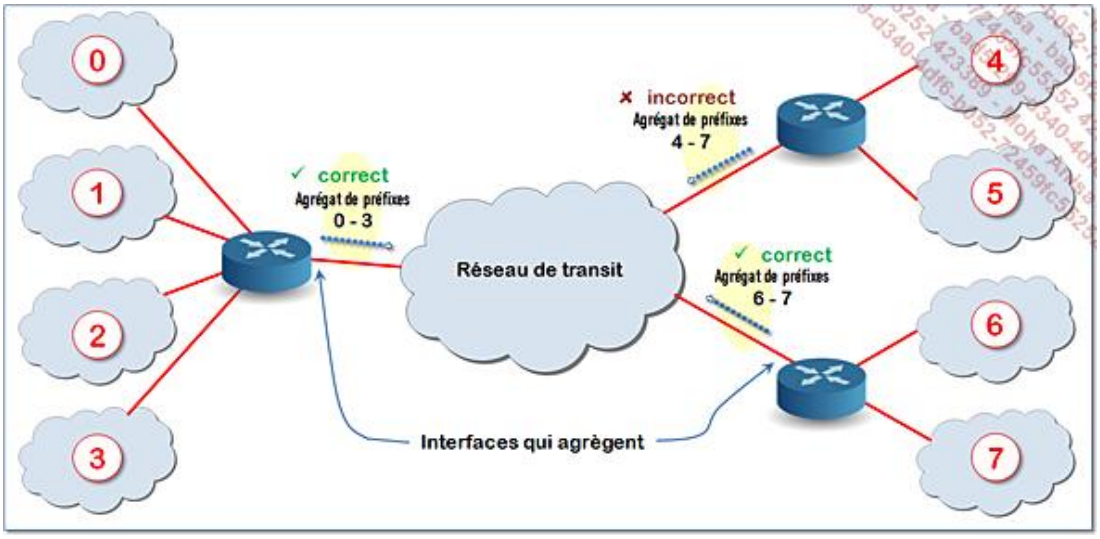
Réseau d'extrémité (Stub network)

Un réseau d'extrémité n'a qu'un seul routeur attaché. Chaque paquet observé sur un réseau d'extrémité a, soit

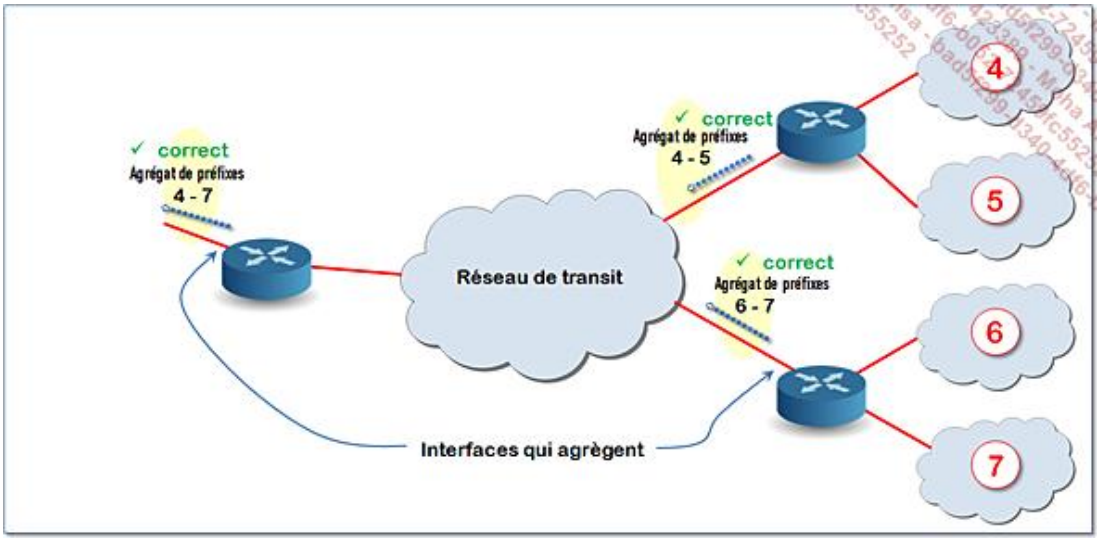
l'adresse source, soit l'adresse destination qui appartient à ce réseau. Les interfaces de loopback sont considérées comme des réseaux d'extrémité et annoncées comme tels.

L'agrégation de routes doit toujours être recherchée car elle réduit la taille des tables de routage et par suite, diminue le temps de recherche d'une destination dans la table. Une autre conséquence non évoquée jusqu'à présent est que si un réseau n'est pas annoncé explicitement mais l'est au travers d'un agrégat, ses changements d'état « up » ou « down » se trouvent masqués et ne sont pas propagés dans le reste du réseau. Voilà qui diminue d'autant le trafic d'acheminement dans les réseaux de transit.

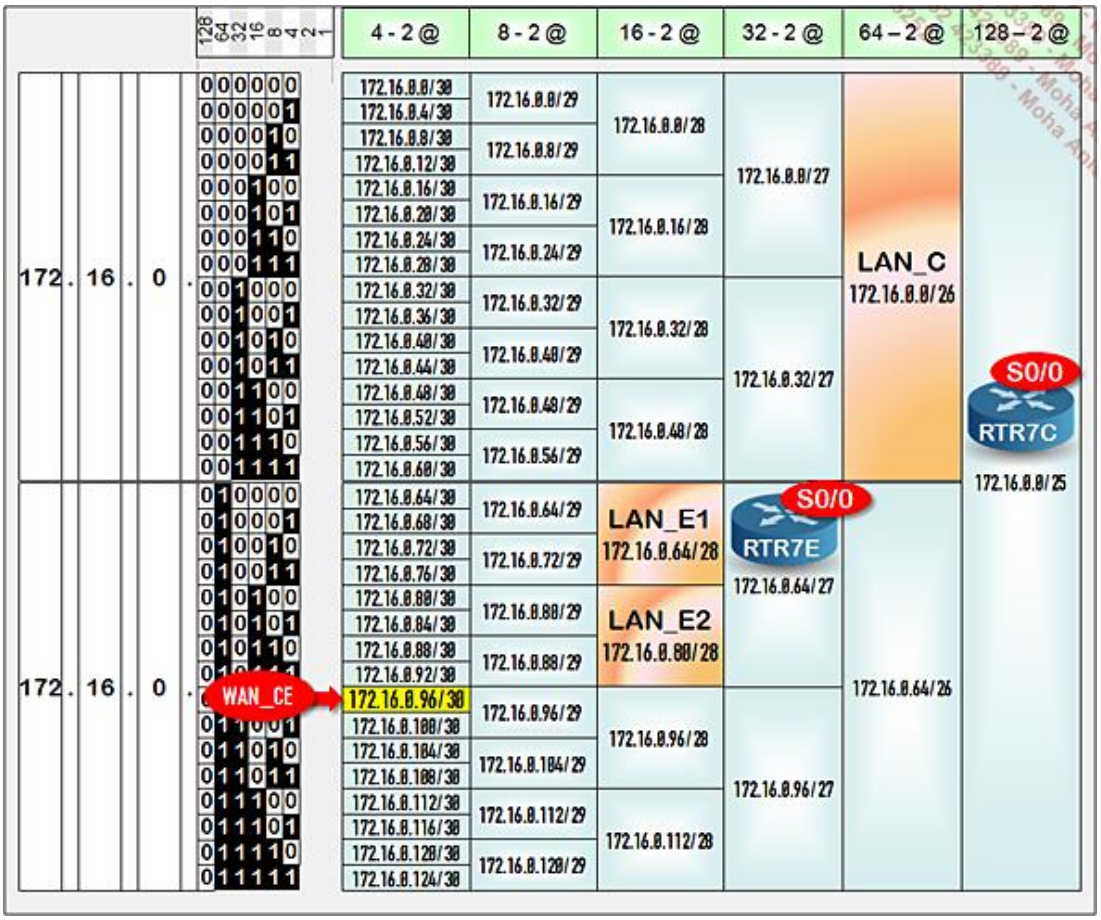
L'agrégat doit regrouper autant de préfixes que possible mais pas plus. Autrement dit, tous les réseaux annoncés par l'agrégat via une interface de routeur doivent effectivement être accessibles par cette interface. On se souvient également qu'on ne peut agréger des réseaux qu'en nombres égaux à des puissances de 2 (notre tableau VLSM dichotomique) :



Il est souhaitable d'agréger les agrégats quand c'est possible :



Revenons à notre étude de cas en construisant le tableau dichotomique de l'espace d'adressage utilisé, à savoir 172.16.0.0/24. Les routeurs concernés par la configuration d'agrégations ont été placés sur la partie de l'espace d'adressage convenable :



L'agrégation se configure sur l'interface sortante. Quatre agrégations sont possibles dans le cas présent :

- RTR7E peut annoncer 172.16.0.64/27 sur son interface S0/0.
- RTR7C peut annoncer 172.16.0.0/25 sur son interface S0/0.
- RTR7B peut annoncer 172.16.0.192/27 sur son interface S0/0.
- RTR7D peut annoncer 172.16.0.128/26 sur son interface S0/1.

La commande utile est **ip summary-address** à entrer en configuration d'interface, dont la syntaxe est la suivante :

```
ip summary-address eigrp as-number network-address subnet-mask [admin-distance]
```

... et dont les arguments sont les suivants :

as-number

Numéro d'AS.

network-address

Le préfixe de l'agrégat.

subnet-mask

Un masque ordinaire et non un masque générique.

admin-distance

Optionnel, l'agrégat peut être annoncé avec une distance administrative particulière. La valeur doit appartenir à l'espace [0 - 255]. On se souvient que la distance administrative par défaut d'EIGRP est 90 quand il s'agit d'une route EIGRP interne, 170 quand la route EIGRP est externe.

Il reste à configurer les agrégats sur les routeurs concernés :

RTR7E

```
RTR7E(config)#int s0/0
RTR7E(config-if)#ip summary-address eigrp 64501 172.16.0.64 255.255.255.224
RTR7E(config-if)#
00:16:09: %DUAL-5-NBRCHANGE: IP-EIGRP 64501: Neighbor 172.16.0.97 (Serial0/0) is
down: summary configured
00:16:10: %DUAL-5-NBRCHANGE: IP-EIGRP 64501: Neighbor 172.16.0.97 (Serial0/0) is
up: new adjacency
RTR7E(config-if)#^Z
RTR7E#
```

RTR7C

```
RTR7C(config)#int s0/0
RTR7C(config-if)#ip summary-address eigrp 64501 172.16.0.0 255.255.255.128
RTR7C(config-if)#^Z
RTR7C#
```

RTR7B

```
RTR7B(config)#int s0/0
RTR7B(config-if)#ip summary-address eigrp 64501 172.16.0.192 255.255.255.224
RTR7B(config-if)#^Z
RTR7B#
```

RTR7D

```
RTR7D(config)#int s0/1
RTR7D(config-if)#ip summary-address eigrp 64501 172.16.0.128 255.255.255.192
RTR7D(config-if)#^Z
RTR7D#
```

Notre réseau de transit se ramène au seul routeur RTR7A sur lequel l'administrateur observe avec satisfaction une diminution conséquente du volume de la table de routage :

```
RTR7A#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     172.16.0.0/16 is variably subnetted, 3 subnets, 3 masks
D       172.16.0.128/26 [90/2172416] via 192.168.0.254, 00:01:43, Serial0/1
D       172.16.0.192/27 [90/2172416] via 192.168.0.246, 00:04:06, Serial0/2
D       172.16.0.0/25 [90/2172416] via 192.168.0.249, 00:06:39, Serial0/0
     192.168.0.0/30 is subnetted, 3 subnets
C       192.168.0.248 is directly connected, Serial0/0
C       192.168.0.252 is directly connected, Serial0/1
C       192.168.0.244 is directly connected, Serial0/2
RTR7A#
```

Il est intéressant d'observer également ce qu'est devenue la table de routage de l'un des routeurs qui agrège. Par exemple, sur RTR7C :

```
RTR7C#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is not set

     172.16.0.0/16 is variably subnetted, 6 subnets, 4 masks
D       172.16.0.128/26 [90/2684416] via 192.168.0.250, 00:00:09, Serial0/0
D       172.16.0.192/27 [90/2684416] via 192.168.0.250, 00:00:09, Serial0/0
D       172.16.0.0/25 is a summary, 00:02:35, Null0
C       172.16.0.0/26 is directly connected, FastEthernet0/0
C       172.16.0.96/30 is directly connected, Serial0/1
D       172.16.0.64/27 [90/2172416] via 172.16.0.98, 00:02:30, Serial0/1
     192.168.0.0/30 is subnetted, 3 subnets
C       192.168.0.248 is directly connected, Serial0/0
D       192.168.0.252 [90/2681856] via 192.168.0.250, 00:00:09, Serial0/0
D       192.168.0.244 [90/2681856] via 192.168.0.250, 00:00:09, Serial0/0
RTR7C#
```

Ainsi, le routeur a installé une route vers l'agrégat 172.16.0.0/25, route dont le prochain saut est l'interface virtuelle Null0. Tout paquet reçu par RTR7C et destiné à l'agrégat mais pour lequel le routeur n'aurait pas de réseau enfant plus spécifique serait détruit.

EIGRP utilise cette interface afin d'éliminer les paquets qui correspondent à la route parent mais pour lesquels aucun des réseaux enfants ne correspond à l'adresse de destination des paquets. EIGRP inclut de façon automatique une route résumée vers l'interface Null0 à chaque fois que l'une de ces deux conditions existe :

- Un résumé de route est activé, qu'il soit automatique ou manuel.
- Un ou plusieurs sous-réseaux du réseau agrégé existent qui ont été appris par EIGRP.

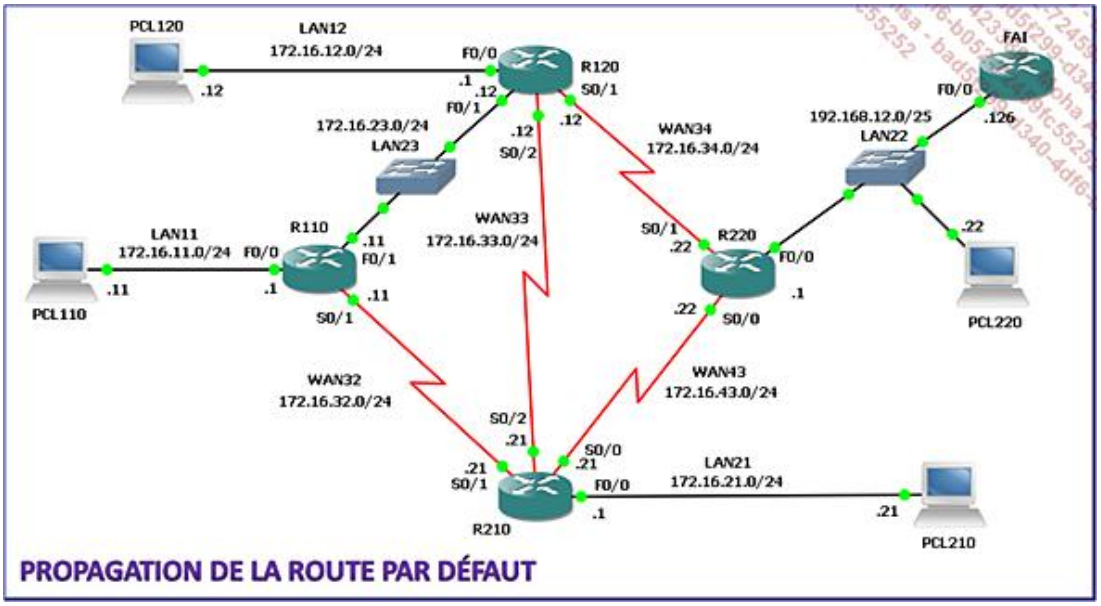
À retenir : le routeur configuré pour agréger plusieurs préfixes n'annonce l'agrégat que s'il connaît au moins l'un des préfixes de l'agrégat. Dans le même temps, il installe dans la table de routage une route pour l'agrégat vers l'interface Null0 afin d'éliminer les paquets destinés à l'agrégat et pour lesquels il ne disposerait pas de route spécifique.

---

c. Route par défaut

Pour la circonstance, nous réutiliserons un contexte préparé antérieurement, il avait servi à illustrer la propagation

de la route par défaut dans un domaine RIP :



Le scénario est le suivant : sur R220, nous installerons une route par défaut vers le routeur FAI puis nous assurerons qu'EIGRP propage effectivement cette route aux autres routeurs de la topologie.

Étape 1 : sur chaque routeur, remplacer RIP par EIGRP

- Par exemple sur R220 :

```
R220(config)#no router rip
R220(config)#router eigrp 64501
R220(config-router)#network 172.16.0.0
R220(config-router)#^Z
R220#
```

Étape 2 : sur R220, ôter la route par défaut vers le réseau 192.168.12.0

Cette route convenait associée à la commande **default-information originate** en configuration de routeur RIP. Ce n'est pas la méthode que nous utiliserons ici.

- Appliquée au cas présent :

```
R220(config)#ip default-network ?
A.B.C.D  IP address of default network

R220(config)#no ip default-network 192.168.12.0
R220(config)#^Z
R220#
```

Étape 3 : sur R220, activer une route statique vers le routeur FAI

- À l'aide d'une commande **ip route** en mode de configuration globale :

```
R220(config)#ip route 0.0.0.0 0.0.0.0 192.168.12.126
R220(config)#^Z
R220#
```

Étape 4 : sur R220, activer la redistribution de route statique dans EIGRP

- À l'aide d'une commande **redistribute** en mode configuration de routeur :

```
R220(config)#router eigrp 64501
R220(config-router)#redistribute static
R220(config-router)#
```

Une seconde méthode pouvait consister à créer une route statique par défaut vers une interface de sortie. On se souvient que l'IOS considère alors la route comme une route directement connectée (relire si nécessaire le chapitre dédié au routage statique). La commande **redistribute** aurait dans ce cas dû être associée au mot-clé **connected**.

Étape 5 : vérifier l'installation de la route statique par défaut sur le routeur qui redistribue

- À l'aide d'une commande **show ip route** sur R220 :

```
R220#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is 192.168.12.126 to network 0.0.0.0

     192.168.12.0/25 is subnetted, 1 subnets
C       192.168.12.0 is directly connected, FastEthernet0/0
     172.16.0.0/24 is subnetted, 8 subnets
C       172.16.43.0 is directly connected, Serial0/0
D       172.16.32.0 [90/41024000] via 172.16.43.21, 00:09:39, Serial0/0
D       172.16.33.0 [90/41024000] via 172.16.34.12, 00:08:57, Serial0/1
                    [90/41024000] via 172.16.43.21, 00:08:57, Serial0/0
C       172.16.34.0 is directly connected, Serial0/1
D       172.16.21.0 [90/20514560] via 172.16.43.21, 00:08:03, Serial0/0
D       172.16.23.0 [90/40514560] via 172.16.34.12, 00:07:55, Serial0/1
D       172.16.12.0 [90/40514560] via 172.16.34.12, 00:07:55, Serial0/1
D       172.16.11.0 [90/40517120] via 172.16.34.12, 00:07:55, Serial0/1
S*   0.0.0.0/0 [1/0] via 192.168.12.126
R220#
```

Étape 6 : vérifier la propagation de la route par défaut sur les autres routeurs

- À l'aide d'une commande **show ip route** sur R120 par exemple :

```
R120#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is 172.16.34.22 to network 0.0.0.0

     172.16.0.0/24 is subnetted, 8 subnets
D       172.16.43.0 [90/41024000] via 172.16.34.22, 00:09:55, Serial0/1
                    [90/41024000] via 172.16.33.21, 00:09:55, Serial0/2
D       172.16.32.0 [90/40514560] via 172.16.23.11, 00:10:54, FastEthernet0/1
C       172.16.33.0 is directly connected, Serial0/2
C       172.16.34.0 is directly connected, Serial0/1
D       172.16.21.0 [90/40514562] via 172.16.33.21, 00:09:56, Serial0/2
C       172.16.23.0 is directly connected, FastEthernet0/1
C       172.16.12.0 is directly connected, FastEthernet0/0
D       172.16.11.0 [90/30720] via 172.16.23.11, 00:09:11, FastEthernet0/1
D*EX 0.0.0.0/0 [170/40514560] via 172.16.34.22, 00:09:57, Serial0/1
R120#
```

Observez la route par défaut candidate associée à la distance administrative 170 puisque pour EIGRP, il s'agit d'une route externe. Observez également l'adresse IP affectée à la passerelle de dernier recours 172.16.34.22, soit l'adresse IP de l'interface S0/1 sur R220. CQFD.

À tout hasard, le tableau suivant aide à comprendre la distance affichée 40 514 560 de la route candidate :

Lien	BW (Kbps)	DLY (µs)	BW résultante	DLY résultant	Métrique IGRP	Métrique EIGRP
F0/0 de	10 000	100	10 000	100	1 010	258 560

R220						
S0/1 de R120	64	20 000	64	20 100	158 260	40 514 560

Nous aurions pu utiliser la commande **default-metric** en configuration de routeur pour ajuster les paramètres bandwidth, delay, charge et reliability de la route statique introduite dans EIGRP. En fait, c'est inutile parce qu'EIGRP sait redistribuer (récupérer) le coût de la route statique (il saurait également récupérer le coût d'une route issue d'un autre AS EIGRP). En revanche, s'il avait fallu redistribuer des routes issues d'un autre protocole de routage tel RIP ou OSPF, alors la commande **default-metric** ou le mot-clé **metric** à associer à la commande **redistribute** auraient été utiles.

Dernière remarque : s'inquiéter du coût d'un chemin lorsque le chemin est unique n'a pas beaucoup de sens, c'est de toute façon le chemin qu'EIGRP adoptera. Cette remarque s'applique au cas de la route par défaut.

Bonus : il est possible de filtrer la sortie d'une commande **show** à l'aide d'un « tube ». Le tube est entré à l'aide du caractère « | ». Sur un clavier AZERTY, ce caractère s'obtient avec la combinaison des deux touches [Alt Gr] 6. Trois mots-clés permettent de préciser l'action du filtre : « **begin** », « **include** » et « **exclude** ». Trois exemples devraient montrer si besoin tout l'intérêt de la chose :

- Exemple 1 : l'administrateur utilise un filtre **begin** pour afficher le contenu du fichier de configuration à partir des lignes qui concernent la configuration du protocole de routage :

```
R220#show run | begin router
router eigrp 64501
redistribute static metric 10000 10 255 1 1500
network 172.16.0.0
 auto-summary
!
ip classless
ip route 0.0.0.0 0.0.0.0 192.168.12.126
ip http server
!
!
dial-peer cor custom
!
!
!
!
!
line con 0
exec-timeout 0 0
logging synchronous
line aux 0
line vty 0 4
login
!
end
```

- Exemple 2 : l'administrateur utilise un filtre **include** afin de n'afficher que les lignes qui comportent la séquence « ip » :

```
R220#show run | include ip
ip subnet-zero
ip address 192.168.12.1 255.255.255.128
ip address 172.16.43.22 255.255.255.0
no ip address
ip address 172.16.34.22 255.255.255.0
ip classless
ip route 0.0.0.0 0.0.0.0 192.168.12.126
ip http server
R220#
```

- Exemple 3 : l'administrateur utilise un filtre **exclude** afin de supprimer les lignes de commentaires :

```
R220#show run | exclude !
```

```
Building configuration...

Current configuration : 821 bytes
version 12.2
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
hostname R220
memory-size iomem 15
ip subnet-zero
call rsvp-sync
interface FastEthernet0/0
ip address 192.168.12.1 255.255.255.128
duplex auto
speed auto
interface Serial0/0
bandwidth 128
ip address 172.16.43.22 255.255.255.0
interface FastEthernet0/1
no ip address
shutdown
duplex auto
speed auto
interface Serial0/1
bandwidth 64
ip address 172.16.34.22 255.255.255.0
router eigrp 64501
redistribute static metric 10000 100 255 1 1500
network 172.16.0.0
auto-summary
ip classless
ip route 0.0.0.0 0.0.0.0 192.168.12.126
ip http server
dial-peer cor custom
line con 0
exec-timeout 0 0
logging synchronous
line aux 0
line vty 0 4
login
end

R220#
```

Ce dernier exemple pourrait se révéler utile quand l'administrateur décide de placer une configuration dans un fichier texte pour un travail hors interface ILC. Bien sûr, il est possible d'appliquer ces filtres à toute commande **show**.

11. Partage de charge

Le chapitre Le routage statique a introduit la notion de partage de charge à coût égal ou à coût inégal et a décrit les deux modes de commutation « Fast switching » et « Process switching ». Afin de ne pas effaroucher le lecteur, cette description, sans être fausse n'était que partielle. Complétons-la en n'ignorant plus le mécanisme de commutation CEF (*Cisco Express Forwarding*).

Pour mémoire, le partage de charge consiste à répartir le trafic vers une même destination (en couche 3) sur plusieurs liens. Quand un routeur découvre plusieurs alternatives vers une destination, sa table de routage peut comporter plusieurs entrées pour cette destination. Par exemple :

```
R110b#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
I       10.0.1.192/27 [115/10] via 10.0.1.234
                      [115/10] via 10.0.1.226
                      [115/10] via 10.0.1.236
.........
R110b#
```

Le plus ordinairement, les alternatives ont même métrique mais certains protocoles de routage, dont EIGRP, savent

placer dans la table de routage des alternatives à coût inégal.

Le partage de charge est étroitement lié au mécanisme de commutation du routeur, mécanisme chargé de placer un paquet d'une file de sortie dans la trame convenable puis d'émettre cette trame sur l'interface choisie. En effet, au niveau 3, stricto sensu, la fonction de routage doit rapprocher les paquets de leur destination via un chemin et il lui importe peu de savoir que plusieurs chemins existent. Mais le bon sens recommande de faire en sorte que la bande passante disponible soit utilisée de la façon la plus efficace. D'autant que prévoir un mécanisme de partage de charge ne nécessite rien de plus que de la réflexion et de la configuration sur le routeur considéré.

Le nombre de chemins alternatifs utilisés est limité par le nombre d'entrées que peut placer un protocole de routage dans la table de routage pour une même destination. Par défaut, l'IOS limite ce nombre maximal à 4 pour la plupart des protocoles de routage à l'exception de BGP (une entrée !), l'administrateur peut porter cette valeur à 6 à l'aide de la commande **maximum-paths** en configuration de routeur :

```
R11(config)#router eigrp 64501
R11(config-router)#maximum-paths ?
<1-6>  Number of paths

R11(config-router)#maximum-paths 6
R11(config-router)#^Z
R11#
```

L'IOS CISCO supporte deux modes de partage de charge : le partage par destination et le partage par paquet :

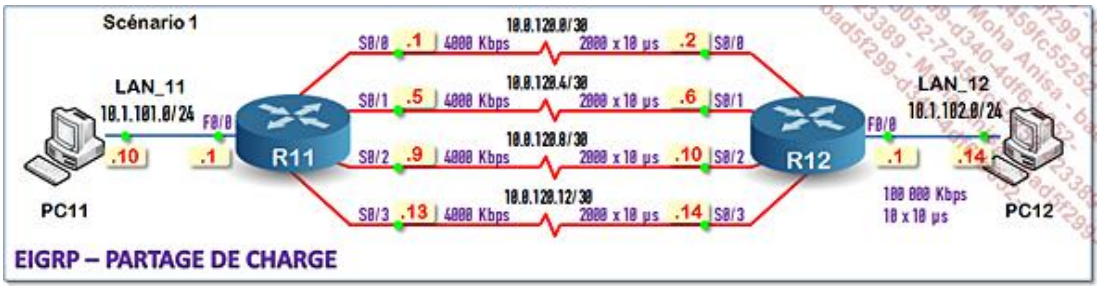
- Dans le partage par destination, tous les paquets pour une destination donnée empruntent le même chemin. Cela préserve le séquencement des paquets mais ne garantit pas une réelle répartition du trafic et donc ne garantit pas plus l'utilisation la plus efficace de tous les liens disponibles. C'est à l'administrateur qu'il revient de vérifier qu'un hôte n'est pas destinataire de la majeure partie du trafic. En final, le trafic est d'autant mieux réparti que le nombre de destinataires est important. Pour chaque nouvelle destination, l'IOS construit une entrée dans un cache de routage qui comprend la destination et l'interface de sortie. Le traitement des paquets suivants est simplifié, le processus de commutation peut remplir sa mission sans solliciter le processus de routage. Ce fonctionnement efficace à première vue n'est pas sans inconvénient, chaque nouvel hôte destinataire provoquant une entrée supplémentaire dans le cache que le réseau destinataire soit identique ou pas, y compris quand un seul chemin existe pour cette destination. L'entretien du cache peut absorber une grande quantité de ressources machine. Par ailleurs, la décision d'emprunter tel ou tel chemin est prise au premier paquet d'un flux. La charge constatée sur les différents liens peut ensuite évoluer sans que le chemin emprunté par le flux ne soit remis en question. Si bien qu'on peut aboutir à cette situation paradoxale dans laquelle le lien emprunté serait saturé alors même que les autres liens disponibles « dormiraient ».

- Dans le partage par paquet, le partage de charge est garanti. En revanche, les différents paquets d'un même flux vont emprunter des chemins de latences différentes et le séquencement des paquets n'est plus garanti. Avant l'introduction du mécanisme CEF, faire le choix du partage par paquet revenait à désactiver le cache de route et perdre l'accélération du traitement qui en découlait (on quittait le mode « *Fast Switching* » pour revenir au mode « *Process Switching* »). Chaque paquet traité nécessite de solliciter le processus de routage qui doit lire la table de routage, parfois plusieurs fois, avant de trouver l'interface de sortie adéquate, l'interface la moins sollicitée quand plusieurs alternatives existent. Certes l'utilisation des liens est optimisée mais ce résultat est obtenu au prix d'une lourde tâche qu'il faut accomplir à chaque paquet. Autant dire que ce fonctionnement est incompatible avec des interfaces à haut débit.

À partir de la version 11.1 de l'IOS, dans un premier temps sur ses routeurs haut de gamme, Cisco a introduit un nouveau mécanisme de commutation des paquets dit CEF et qui permet de réaliser un vrai partage de charge sans perdre l'intérêt d'un cache de route, y compris quand le partage par paquet est choisi. CEF est un processus, activé par la commande **ip cef** en mode de configuration globale. Optionnellement, le mot-clé **distributed**, quand il est ajouté à la commande, provoque une délocalisation du cache directement sur les cartes d'interface compatibles, c'est un fonctionnement possible sur les séries 2420, 2600, 3600, 3700 et 7200. Au-delà, sur les gammes ASR1000, le mot-clé distributed n'est plus une option mais le seul fonctionnement prévu. À chaque nouveau flux, CEF extrait une seule fois de la table de routage toute l'information dont il a besoin pour assurer sa tâche de commutation. CEF place cette information dans une table dite table FIB (*Forwarding Information Base*).

a. EIGRP et le partage de charge à coût égal

Nous aurons besoin d'un exemple concret pour asseoir notre propos. Adoptons cette topologie simple :



Les points essentiels de la configuration de R11 :

```
!
hostname R11
!
memory-size iomem 10
ip subnet-zero
ip cef
!
call rsvp-sync
!
interface FastEthernet0/0
ip address 10.1.101.1 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
bandwidth 4000
ip address 10.0.120.1 255.255.255.252
ip access-group 10 out
no ip mroute-cache
!
interface Serial0/1
bandwidth 4000
ip address 10.0.120.5 255.255.255.252
ip access-group 20 out
no ip mroute-cache
!
interface Serial0/2
bandwidth 4000
ip address 10.0.120.9 255.255.255.252
ip access-group 30 out
no ip mroute-cache
!
interface Serial0/3
bandwidth 4000
ip address 10.0.120.13 255.255.255.252
ip access-group 40 out
no ip mroute-cache
!
router eigrp 64501
network 10.0.0.0
auto-summary
!
ip classless
ip http server
!
access-list 10 permit 10.1.101.0 0.0.0.255
access-list 10 permit any
access-list 20 permit 10.1.101.0 0.0.0.255
access-list 20 permit any
access-list 30 permit 10.1.101.0 0.0.0.255
access-list 30 permit any
access-list 40 permit 10.1.101.0 0.0.0.255
access-list 40 permit any
!
dial-peer cor custom
!
line con 0
```

```
exec-timeout 0 0
logging synchronous
line aux 0
line vty 0 4
login
!
end
```

Le chapitre dédié au routage statique avait montré comment vérifier le partage de charge à l'aide de la commande **debug ip packet** tout en soulignant les dangers de cette commande. Dans le cas présent, l'administrateur a préféré utiliser des listes d'accès. L'administrateur a créé quatre listes d'accès dont l'objet n'est pas de filtrer le trafic mais de le comptabiliser. Une commande **show ip access-lists** permettra de découvrir combien de paquets ont transité sur chaque interface. La commande **clear ip access-lists counters** sera utile quand il faudra remettre les compteurs à zéro. La liste d'accès n°10 correspond à l'interface S0/0, la n°11 à S0/1 et ainsi de suite.

À l'aide du tableau Excel « Calcul_Métrique_EIGRP.xlsm » fourni en téléchargement, calculons la métrique de la route vers LAN_12 vue du routeur R11 :

Lien	BW (Kbps)	DLY (µs)	BW résultante	DLY résultant	Métrique IGRP	Métrique EIGRP
F0/0 de R12	100 000	100	100 000	100	110	28 160
S0/0 de R11	4 000	20 000	4 000	20 100	4 510	1 154 560

Les quatre routes vers LAN_12 sont à coût égal, EIGRP les place ensemble dans la table de routage (Extrait) :

```
R11#sh ip route
.........
Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
D       10.1.102.0/24 [90/1154560] via 10.0.120.6, 00:01:33, Serial0/1
                      [90/1154560] via 10.0.120.2, 00:01:33, Serial0/0
                      [90/1154560] via 10.0.120.10, 00:01:33, Serial0/2
                      [90/1154560] via 10.0.120.14, 00:01:33, Serial0/3
.........
```

CISCO n'est pas très disert sur le fonctionnement interne du processus CEF mais nous pouvons nous en faire une bonne idée à l'aide d'une commande réservée aux ingénieurs du constructeur et qui n'est pas documentée. Il faut donc la frapper « dans le noir », l'auto-complétion ne fonctionne pas :

```
R11#sh ip cef 10.1.102.14 ?
A.B.C.D  prefix mask
detail   Display full information
  |        Output modifiers
<cr>

R11#sh ip cef 10.1.102.0 internal
10.1.102.0/24, version 23, per-destination sharing
0 packets, 0 bytes
via 10.0.120.2, Serial0/0, 0 dependencies
traffic share 1
next hop 10.0.120.2, Serial0/0
valid adjacency
via 10.0.120.10, Serial0/2, 0 dependencies
traffic share 1
next hop 10.0.120.10, Serial0/2
valid adjacency
via 10.0.120.14, Serial0/3, 0 dependencies
traffic share 1
next hop 10.0.120.14, Serial0/3
valid adjacency
via 10.0.120.6, Serial0/1, 0 dependencies
traffic share 1
next hop 10.0.120.6, Serial0/1
valid adjacency

  0 packets, 0 bytes switched through the prefix
```

```
tmstats: external 0 packets, 0 bytes
internal 0 packets, 0 bytes
  Load distribution: 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 (refcount 1)

Hash  OK  Interface                 Address         Packets
  1     Y   Serial0/0                 point2point           0
  2     Y   Serial0/2                 point2point           0
  3     Y   Serial0/3                 point2point           0
  4     Y   Serial0/1                 point2point           0
  5     Y   Serial0/0                 point2point           0
  6     Y   Serial0/2                 point2point           0
  7     Y   Serial0/3                 point2point           0
  8     Y   Serial0/1                 point2point           0
  9     Y   Serial0/0                 point2point           0
  10    Y   Serial0/2                 point2point           0
  11    Y   Serial0/3                 point2point           0
  12    Y   Serial0/1                 point2point           0
  13    Y   Serial0/0                 point2point           0
  14    Y   Serial0/2                 point2point           0
  15    Y   Serial0/3                 point2point           0
  16    Y   Serial0/1                 point2point           0
R11#
```

CEF a créé une table de répartition à 16 entrées. Cette table, appelée table FIB (Forwarding Information Base) par CISCO, est destinée à hacher le flux et le répartir sur les différentes interfaces. Dans le cas présent, les 4 interfaces possibles sont à coût égal, elles apparaissent donc dans une succession parfaite 0, 2, 3, 1, ce quatre fois consécutivement.

Par ailleurs, on apprend également en début de capture que le mode de partage en cours est par destination. Une fois l'entrée de la table CEF construite, chaque nouveau flux destiné à l'un des hôtes du réseau 10.1.102.0/24 sera commuté sur l'interface pointée par l'entrée « Hash » suivante. Dans l'exemple qui suit, l'administrateur a entré une commande ping 10.1.102.14 (PC12) depuis la station PC11 (avec l'option -t si PC11 est une station Windows de façon à ce que le ping soit répétitif). Une surveillance régulière des compteurs des listes d'accès montre que ce flux emprunte l'interface S0/3.

```
R11#sh access-lists
Standard IP access list 10
permit 10.1.101.0, wildcard bits 0.0.0.255
Standard IP access list 20
permit 10.1.101.0, wildcard bits 0.0.0.255
Standard IP access list 30
permit 10.1.101.0, wildcard bits 0.0.0.255
Standard IP access list 40
permit 10.1.101.0, wildcard bits 0.0.0.255(174 matches)
R11#
```

L'administrateur fait cesser la requête ping répétitive vers PC12 puis en lance une autre vers un autre PC sur LAN_12. Attention, il ne suffit pas de changer l'adresse IP de la station PC12 pour constater le phénomène, ce qui prouve que CEF utilise les adresses de couche 2. Puisque le flux précédent a emprunté S0/3, le prochain flux devrait, selon la table de hachage, emprunter l'interface S0/1. L'administrateur se hâte de vérifier ce qu'il pressent :

```
R11#sh access-lists
Standard IP access list 10
permit 10.1.101.0, wildcard bits 0.0.0.255
Standard IP access list 20
permit 10.1.101.0, wildcard bits 0.0.0.255 (7 matches)
Standard IP access list 30
permit 10.1.101.0, wildcard bits 0.0.0.255
Standard IP access list 40
permit 10.1.101.0, wildcard bits 0.0.0.255 (174 matches)
R11#
```

À nouveau, nous vérifions que ce mode de partage par destination ne correspondra à un partage effectif que si le nombre de destinations est suffisant. Mais avec CEF, il n'y a plus de raison de se priver d'un réel partage en activant le mode par paquet. La commande utile est **ip load-sharing per-packet** en configuration d'interface.

Basculer sur le mode de partage par paquet nécessite d'entrer la commande **ip load-sharing per-packet** sur l'ensemble des interfaces concernées par le partage du trafic.

Dans le cas présent :

```
R11(config)#int s0/0
R11(config-if)#ip load-sharing ?
  per-destination  Deterministic distribution
  per-packet       Random distribution

R11(config-if)#ip load-sharing per-packet
R11(config-if)#int s0/1
R11(config-if)#ip load-sharing per-packet
R11(config-if)#int s0/2
R11(config-if)#ip load-sharing per-packet
R11(config-if)#int s0/3
R11(config-if)#ip load-sharing per-packet
R11(config-if)#^Z
R11#
```

L'administrateur peut vérifier la prise en compte effective du nouveau mode de partage par CEF (Extrait) :

```
R11#show ip cef 10.1.102.0 internal
10.1.102.0/24, version 31, per-packet sharing
0 packets, 0 bytes
  .........
```

L'administrateur relance un ping permanent depuis PC11 vers PC12, remet à zéro les compteurs des listes d'accès puis surveille régulièrement l'évolution des compteurs :

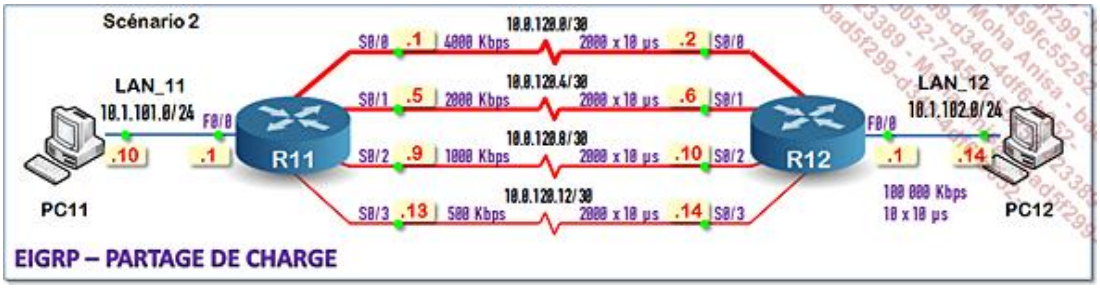
```
R11#clear ip access-list counters
R11#sh access-lists
Standard IP access list 10
  permit 10.1.101.0, wildcard bits 0.0.0.255
Standard IP access list 20
  permit 10.1.101.0, wildcard bits 0.0.0.255
Standard IP access list 30
  permit 10.1.101.0, wildcard bits 0.0.0.255
Standard IP access list 40
  permit 10.1.101.0, wildcard bits 0.0.0.255
R11#sh access-lists
Standard IP access list 10
  permit 10.1.101.0, wildcard bits 0.0.0.255 (1 match)
Standard IP access list 20
  permit 10.1.101.0, wildcard bits 0.0.0.255 (2 matches)
Standard IP access list 30
  permit 10.1.101.0, wildcard bits 0.0.0.255 (2 matches)
Standard IP access list 40
  permit 10.1.101.0, wildcard bits 0.0.0.255 (2 matches)
R11#sh access-lists
Standard IP access list 10
  permit 10.1.101.0, wildcard bits 0.0.0.255 (2 matches)
Standard IP access list 20
  permit 10.1.101.0, wildcard bits 0.0.0.255 (2 matches)
Standard IP access list 30
  permit 10.1.101.0, wildcard bits 0.0.0.255 (3 matches)
Standard IP access list 40
  permit 10.1.101.0, wildcard bits 0.0.0.255 (3 matches)
R11#sh access-lists
Standard IP access list 10
  permit 10.1.101.0, wildcard bits 0.0.0.255 (3 matches)
Standard IP access list 20
  permit 10.1.101.0, wildcard bits 0.0.0.255 (4 matches)
Standard IP access list 30
  permit 10.1.101.0, wildcard bits 0.0.0.255 (4 matches)
Standard IP access list 40
  permit 10.1.101.0, wildcard bits 0.0.0.255 (4 matches)
R11#sh access-lists
Standard IP access list 10
  permit 10.1.101.0, wildcard bits 0.0.0.255 (5 matches)
Standard IP access list 20
  permit 10.1.101.0, wildcard bits 0.0.0.255 (5 matches)
Standard IP access list 30
  permit 10.1.101.0, wildcard bits 0.0.0.255 (5 matches)
```

```
Standard IP access list 40
  permit 10.1.101.0, wildcard bits 0.0.0.255 (5 matches)
R11#sh access-lists
Standard IP access list 10
  permit 10.1.101.0, wildcard bits 0.0.0.255 (6 matches)
Standard IP access list 20
  permit 10.1.101.0, wildcard bits 0.0.0.255 (6 matches)
Standard IP access list 30
  permit 10.1.101.0, wildcard bits 0.0.0.255 (6 matches)
Standard IP access list 40
  permit 10.1.101.0, wildcard bits 0.0.0.255 (6 matches)
R11#sh access-lists
Standard IP access list 10
  permit 10.1.101.0, wildcard bits 0.0.0.255 (8 matches)
Standard IP access list 20
  permit 10.1.101.0, wildcard bits 0.0.0.255 (8 matches)
Standard IP access list 30
  permit 10.1.101.0, wildcard bits 0.0.0.255 (9 matches)
Standard IP access list 40
  permit 10.1.101.0, wildcard bits 0.0.0.255 (8 matches)
R11#
```

CQFD.

b. EIGRP et le partage de charge à coût inégal

Modifions un peu le scénario en imaginant des routes à coût différent entre R11 et R12 :



L'administrateur modifie en conséquence la configuration des interfaces :

```
R11(config)#int S0/1
R11(config-if)#bandwidth 2000
R11(config-if)#int S0/2
R11(config-if)#bandwidth 1000
R11(config-if)#int S0/3
R11(config-if)#bandwidth 500
R11(config-if)#^Z
R11#
```

Par défaut, EIGRP ne pratique que le partage à coût égal. Fort logiquement, la table de routage de R11 ne montre qu'une seule alternative vers LAN_12 :

```
R11#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
     10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
D       10.1.102.0/24 [90/1154560] via 10.0.120.2, 00:00:11, Serial0/0
.........
```

EIGRP offre une commande des plus intéressantes puisqu'elle lui permet de placer plusieurs entrées pour une même destination dans la table de routage bien que le coût de ces entrées soit différent du meilleur coût. Il s'agit de la commande **variance**. Le facteur variance doit être considéré comme un coefficient multiplicateur. Pour une destination donnée, EIGRP placera dans la table de routage toute alternative dont le coût est inférieur au meilleur coût multiplié par la variance. Posons-nous la question : quelle est la variance nécessaire pour faire apparaître l'alternative via S0/1 dans la table de routage de R11 et pour la destination 10.1.102.0/24 ? Pour répondre, il faut calculer le coût de cette alternative :

Lien	BW (Kbps)	DLY (µs)	BW résultante	DLY résultant	Métrique IGRP	Métrique EIGRP
F0/0 de R12	100 000	100	100 000	100	110	28 160
S0/1 de R11	2 000	20 000	2 000	20 100	7 010	1 794 560

L'alternative via S0/1 a donc un coût $\frac{1\,794\,560}{1\,154\,560} = 1{,}55$ fois plus élevé que le meilleur coût. Puisque la variance est un nombre entier qui doit être compris entre 1 et 128, l'administrateur adopte la valeur 2 :

```
R11#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R11(config)#router eigrp 64501
R11(config-router)#variance 2
R11(config-router)#^Z
R11#
```

L'administrateur observe satisfait l'apparition des deux alternatives :

```
R11#clear ip route *
R11#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
D       10.1.102.0/24 [90/1154560] via 10.0.120.2, 00:00:02, Serial0/0
                      [90/1794560] via 10.0.120.6, 00:00:02, Serial0/1
.........
R11#
```

Et CEF dans tout ça ? L'administrateur utilise à nouveau la commande non documentée :

```
R11#show ip cef 10.1.102.0 internal
10.1.102.0/24, version 26, per-packet sharing
0 packets, 0 bytes
via 10.0.120.2, Serial0/0, 0 dependencies
traffic share 120, current path
next hop 10.0.120.2, Serial0/0
valid adjacency
via 10.0.120.6, Serial0/1, 0 dependencies
traffic share 77
next hop 10.0.120.6, Serial0/1
valid adjacency

  0 packets, 0 bytes switched through the prefix
tmstats: external 0 packets, 0 bytes
internal 0 packets, 0 bytes
  Load distribution: 0 1 0 1 0 1 0 1 0 1 0 1 0 0 0 0 (refcount 1)

Hash  OK  Interface                 Address         Packets
  1     Y   Serial0/0                 point2point           0
  2     Y   Serial0/1                 point2point           0
  3     Y   Serial0/0                 point2point           0
  4     Y   Serial0/1                 point2point           0
  5     Y   Serial0/0                 point2point           0
  6     Y   Serial0/1                 point2point           0
  7     Y   Serial0/0                 point2point           0
  8     Y   Serial0/1                 point2point           0
  9     Y   Serial0/0                 point2point           0
  10    Y   Serial0/1                 point2point           0
  11    Y   Serial0/0                 point2point           0
  12    Y   Serial0/1                 point2point           0
  13    Y   Serial0/0                 point2point           0
  14    Y   Serial0/0                 point2point           0
```

```
  15    Y   Serial0/0                point2point          0
  16    Y   Serial0/0                point2point          0
R11#
```

Observez que la table FIB comporte deux adjacences qui sont S0/0 et S0/1. Puis le tableau de hachage fait apparaître l'occurrence S0/0 dix fois alors que l'occurrence S0/1 n'apparaît que six fois. Sur 16 paquets, CEF en commutera dix sur S0/0 pour seulement 6 sur S0/1. S0/0 absorbera par conséquent 10/6=1,67 fois plus de trafic que S0/1, ce qui est conforme au ratio des métriques des deux liens. L'administrateur provoque à nouveau du trafic et surveille les compteurs des listes d'accès :

```
R11#sh access-lists
Standard IP access list 10
  permit 10.1.101.0, wildcard bits 0.0.0.255 (18 matches)
  permit any
Standard IP access list 20
  permit 10.1.101.0, wildcard bits 0.0.0.255 (9 matches)
  permit any
Standard IP access list 30
  permit 10.1.101.0, wildcard bits 0.0.0.255
  permit any
Standard IP access list 40
  permit 10.1.101.0, wildcard bits 0.0.0.255
  permit any
R11#
```

Le trafic se répartit comme prévu. Confirmons une dernière fois le raisonnement en nous demandant quelle variance permettrait de faire apparaître les quatre alternatives dans la table de routage de R11 pour la destination 10.1.102.0/24.

Calculons le coût de l'alternative via S0/2 :

Lien	BW (Kbps)	DLY (µs)	BW résultante	DLY résultant	Métrique IGRP	Métrique EIGRP
F0/0 de R12	100 000	100	100 000	100	110	28 160
S0/2 de R11	1 000	20 000	1 000	20 100	12 010	3 074 560

L'alternative via S0/2 a donc un coût $\frac{3\,074\,560}{1\,154\,560} = 2{,}66$ fois plus élevé que le meilleur coût.

Mais en réalité, pour répondre, nous avons surtout besoin du coût de l'alternative la plus défavorable, c'est-à-dire celle de l'interface S0/3 :

Lien	BW (Kbps)	DLY (µs)	BW résultante	DLY résultant	Métrique IGRP	Métrique EIGRP
F0/0 de R12	100 000	100	100 000	100	110	28 160
S0/3 de R11	500	20 000	500	20 100	22 010	5 634 560

L'alternative via S0/3 a donc un coût $\frac{5\,634\,560}{1\,154\,560} = 4{,}88$ fois plus élevé que le meilleur coût. Puisque la variance est un nombre entier qui doit être compris entre 1 et 128, l'administrateur adopte la valeur 5 :

```
R11#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R11(config)#router eigrp 64501
R11(config-router)#variance ?
<1-128>  Metric variance multiplier
R11(config-router)#variance 5
R11(config-router)#^Z
R11#
```

L'administrateur observe satisfait l'apparition des quatre alternatives :

```
R11#clear ip route *
R11#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
```

```
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
     10.0.0.0/8 is variably subnetted, 6 subnets, 2 masks
D       10.1.102.0/24 [90/1154560] via 10.0.120.2, 00:00:02, Serial0/0
                      [90/5634560] via 10.0.120.14, 00:00:02, Serial0/3
                      [90/3074560] via 10.0.120.10, 00:00:02, Serial0/2
                      [90/1794560] via 10.0.120.6, 00:00:02, Serial0/1
.........
```

Le hachage prévu par CEF :

```
R11#sh ip cef 10.1.102.0 internal
10.1.102.0/24, version 26, per-packet sharing
0 packets, 0 bytes
via 10.0.120.2, Serial0/0, 0 dependencies
traffic share 240, current path
next hop 10.0.120.2, Serial0/0
valid adjacency
via 10.0.120.14, Serial0/3, 0 dependencies
traffic share 49
next hop 10.0.120.14, Serial0/3
valid adjacency
via 10.0.120.10, Serial0/2, 0 dependencies
traffic share 90
next hop 10.0.120.10, Serial0/2
valid adjacency
via 10.0.120.6, Serial0/1, 0 dependencies
traffic share 154
next hop 10.0.120.6, Serial0/1
valid adjacency

  0 packets, 0 bytes switched through the prefix
tmstats: external 0 packets, 0 bytes
internal 0 packets, 0 bytes
  Load distribution: 0 1 2 3 0 2 3 0 2 3 0 3 0 3 0 0 (refcount 1)

Hash  OK  Interface                 Address         Packets
  1     Y   Serial0/0                 point2point           0
  2     Y   Serial0/3                 point2point           0
  3     Y   Serial0/2                 point2point           0
  4     Y   Serial0/1                 point2point           0
  5     Y   Serial0/0                 point2point           0
  6     Y   Serial0/2                 point2point           0
  7     Y   Serial0/1                 point2point           0
  8     Y   Serial0/0                 point2point           0
  9     Y   Serial0/2                 point2point           0
  10    Y   Serial0/1                 point2point           0
  11    Y   Serial0/0                 point2point           0
  12    Y   Serial0/1                 point2point           0
  13    Y   Serial0/0                 point2point           0
  14    Y   Serial0/1                 point2point           0
  15    Y   Serial0/0                 point2point           0
  16    Y   Serial0/0                 point2point           0
R11#
```

Pour parvenir à répartir le trafic selon la métrique de chaque lien, CEF fait apparaître sept fois l'occurrence S0/0, cinq fois l'occurrence S0/1, trois fois l'occurrence S0/2 et une seule fois l'occurrence S0/3. Observez que CEF alterne les différentes adjacences jusqu'à épuisement du nombre d'occurrences prévues. C'est ainsi que la table de hachage s'achève toujours par plusieurs occurrences de la même adjacence, celle correspondant au lien ayant le meilleur coût.

Avec ces valeurs, le lien le plus rapide absorbe 1,4 fois plus de trafic que le lien S0/1, 2,33 fois plus de trafic que le lien S0/2 et 7 fois plus de trafic que le lien S0/3, ratios à comparer aux ratios de métriques 1,55, 2,66 et 4,88. Une dernière fois, l'administrateur vérifie la répartition du trafic :

```
R11#sh access-lists
Standard IP access list 10
  permit 10.1.101.0, wildcard bits 0.0.0.255 (484 matches)
Standard IP access list 20
  permit 10.1.101.0, wildcard bits 0.0.0.255 (346 matches)
```

```
Standard IP access list 30
  permit 10.1.101.0, wildcard bits 0.0.0.255 (208 matches)
Standard IP access list 40
  permit 10.1.101.0, wildcard bits 0.0.0.255 (74 matches)
R11#
```

Un peu plus tard :

```
R11#sh access-lists
Standard IP access list 10
  permit 10.1.101.0, wildcard bits 0.0.0.255 (1067 matches)
Standard IP access list 20
  permit 10.1.101.0, wildcard bits 0.0.0.255 (763 matches)
Standard IP access list 30
  permit 10.1.101.0, wildcard bits 0.0.0.255 (459 matches)
Standard IP access list 40
  permit 10.1.101.0, wildcard bits 0.0.0.255 (157 matches)
R11#
```

... et observe avec plaisir que les ratios prévus sont respectés.

c. Synthèse Partage de charge

Des conversations régulières avec des professionnels en activité ont persuadé l'auteur que ce sujet mal connu posait fréquemment des problèmes. Il est souhaitable de bien ancrer les notions suivantes :

- Dire qu'un protocole de routage est capable de partage de charge rappelle simplement sa faculté à placer plusieurs entrées pour une même destination dans sa table de routage.
- Par défaut, le nombre d'entrées maximal est fixé à 4, l'administrateur peut le porter à 6 à l'aide de la commande **maximum-paths** en configuration de routeur.
- C'est en réalité le processus de commutation dont est équipé le routeur qui assure la répartition de charge quand plusieurs alternatives existent pour une même destination.
- Quand le processus de commutation CEF est activé, l'administrateur peut au choix :
  - Laisser le partage réglé dans le mode par destination, c'est le mode par défaut. L'administrateur privilégie dans ce cas le séquencement des paquets.
  - Préférer le partage dans le mode par paquet parce qu'il souhaite une véritable répartition du trafic.
- Quand EIGRP place plusieurs alternatives à coût inégal dans la table de routage, CEF écoule le trafic au prorata des métriques des différentes alternatives.

Configuration EIGRP

1. Configuration de base

a. Choix du système autonome et identificateur de processus

La commande **router eigrp** fait entrer en mode configuration d'un processus EIGRP. Sa syntaxe est la suivante :

```
Router(config)#router eigrp {autonomous-system-number* | virtual-instance-name}
```

... dont les arguments sont les suivants :

autonomous-system-number

Ce numéro est utilisé par le routeur EIGRP pour marquer (« tag ») les routes qu'il place dans ses mises à jour. De la même façon, le routeur EIGRP n'exploitera une mise à jour que si elle est marquée avec ce numéro. En final, le numéro d'AS doit être identique sur tous les routeurs EIGRP d'un domaine EIGRP, c'est la condition pour que deux routeurs directement connectés puissent devenir voisins et par suite puissent partager l'information de routage.

Curieusement, 32 bits sont prévus dans le format des paquets EIGRP pour transporter cette information, mais la commande **router eigrp**, au moins dans la version 12.4 de l'IOS, impose que les numéros soient compris dans l'espace [1 - 65 535]. Si le domaine EIGRP est strictement privé, l'administrateur est absolument libre de choisir un numéro quelconque. Si le domaine est un véritable AS appelé à être annoncé au reste du monde via BGP, alors les numéros d'AS sont administrés par l'IANA et on peut en trouver la liste sur le site http://www.iana.org/assignments/as-numbers/as-numbers.xml.

Observez sur ce site que l'IANA a prévu un espace de numéros pour essais ou documentation :

64496-64511	Reserved for use in documentation and sample code	RFC5398	03/12/08

virtual-instance-name

L'argument fait référence à une configuration EIGRP nommée. Dans ce cas, le nom attribué n'a qu'une portée locale. Nous n'utiliserons pas cette forme de la commande.

*CISCO parle de numéro de système autonome mais beaucoup d'auteurs préfèrent parler d'identificateur de processus.

b. Participation des interfaces, le masque générique

En mode configuration de routeur, la commande **network** est utilisée pour inclure les interfaces devant participer au protocole. Sa syntaxe est la suivante :

```
Router(config-router)#network ip-address [wildcard-mask]
```

... dont les arguments sont les suivants :

ip-address

Adresse IP d'un réseau directement connecté.

wildcard-mask

Optionnel, masque générique.

Parmi ses interfaces, le routeur fait participer au protocole toute interface dont l'adresse appartient au réseau ou à l'un des réseaux déclarés à l'aide de la commande ou des commandes **network**. Le routeur utilise ensuite les interfaces qui répondent à ce critère pour établir des relations de voisinage. Il n'y a pas de limite au nombre de commandes **network** qui peuvent être configurées sur un routeur si bien qu'un administrateur peut très légitimement entrer autant de commandes **network** qu'il y a d'interfaces devant participer au protocole, en spécifiant à chaque commande l'adresse IP de l'interface considérée.

Exemples

L'adresse IP de l'interface F0/1 est 172.26.100.100/24. Cette interface doit participer au protocole.

- Choix 1 : l'administrateur peut entrer la commande **network 172.26.0.0**, soit l'adresse du réseau majeur. Il le fait parce que d'autres interfaces de ce routeur sont directement connectées à d'autres sous-réseaux de ce réseau majeur et que ces interfaces doivent également participer au protocole. Ainsi, une seule commande **network** a suffi pour inclure l'ensemble des interfaces.

- Choix 2 : l'administrateur entre la commande **network 172.26.100.100 0.0.0.0**, soit l'adresse exacte de l'interface exprimée à l'aide du masque générique. Il le fait parce que d'autres interfaces de ce routeur sont directement connectées à d'autres sous-réseaux du réseau majeur 172.26.0.0 mais que ces sous-réseaux ne doivent pas être annoncés ou doivent l'être dans un autre domaine EIGRP, voire être annoncés dans un domaine géré par un autre protocole de routage.

Observez que le choix 1 peut être satisfaisant mais peut aussi ne pas l'être, quand l'administrateur fait une erreur en incluant une interface qui ne devrait pas l'être ou en omettant une interface qui aurait dû participer au protocole. Alors que le choix 2 qui consiste à entrer une commande **network** par interface est toujours correct et présente l'avantage de la systématicité.

En première approche, le masque générique (que CISCO désigne par « *Wildcard* », littéralement joker) utilise une logique inversée. Quand un masque désigne les bits qu'il faut retenir, le masque générique dit quels sont les bits qu'il faut ignorer.

Exemples

- L'association 172.26.100.0 0.0.0.255 inclut toutes les adresses du réseau 172.26.100.0/24.

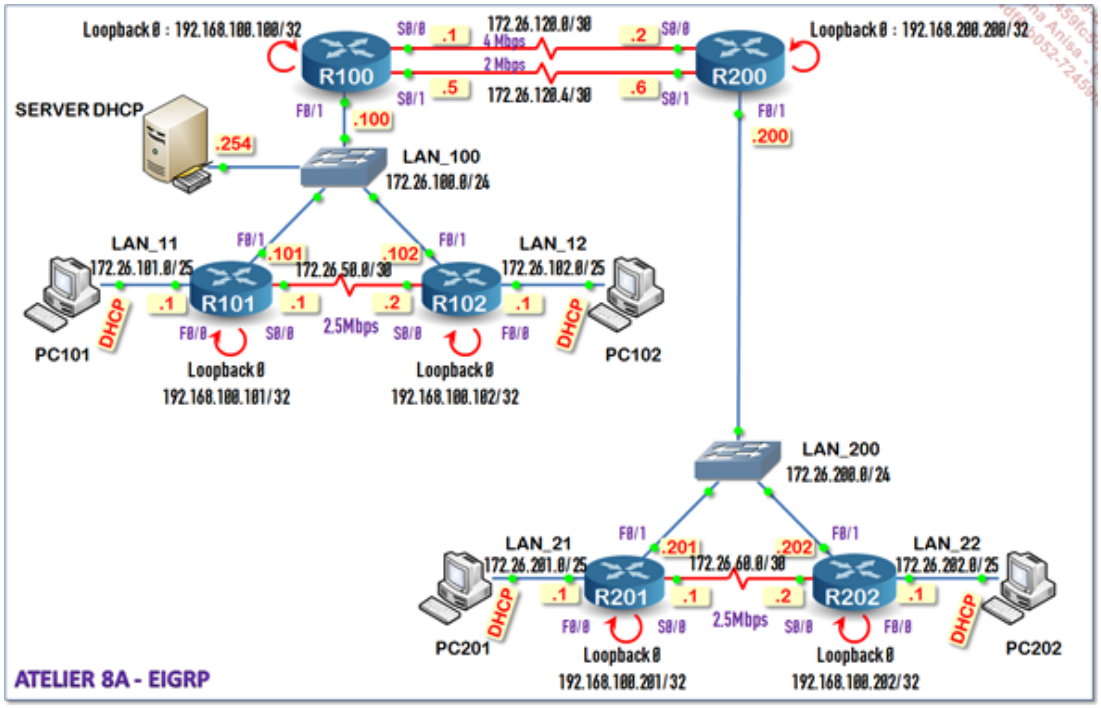
- L'association 172.26.120.0 0.0.0.3 inclut toutes les adresses du réseau 172.26.120.0/30.

- L'association 172.26.120.0 0.0.0.7 inclut toutes les adresses du réseau 172.26.120.0/29.

La réalité est un peu plus complexe mais pour ce chapitre, il suffira de considérer que le masque générique est un masque inversé.

2. Atelier : Mise en œuvre d'une configuration EIGRP

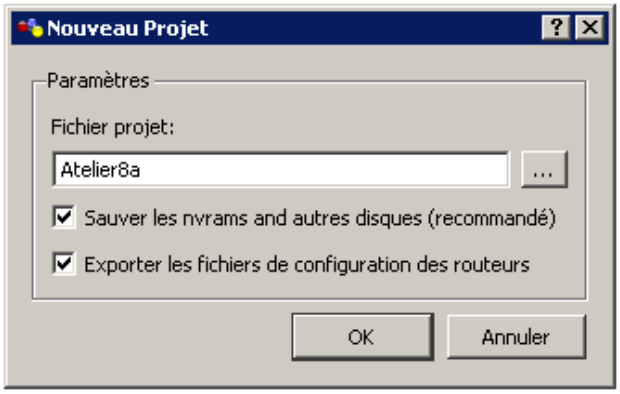
La topologie proposée est la suivante :



a. Tâche 1 : Réalisation de la topologie sous GNS3

- À ce stade de l'ouvrage, nous vous supposons déjà familier de l'utilisation de GNS3. Dans le cas contraire, reportez-vous au chapitre Annexes - Définition d'un contexte d'atelier.

- Créez la topologie sous GNS3 en prenant bien soin de cocher les cases **Sauver les nvrams and autres disques (recommandé)** et **Exporter les fichiers de configuration des routeurs** :



- Placez les six routeurs. La valeur IDLE PC des routeurs a déjà été calculée, inutile d'y revenir. Utilisez la plate-forme 2600, l'auteur a réalisé l'ensemble de la topologie avec des routeurs 2621, l'IOS était la version c2600-ik9s-mz.122-40a.bin qui présente l'avantage de se suffire de 48 Mo de mémoire vive. Ajoutez la carte WIC1T ou WIC2T selon que le routeur est connecté à une ou deux lignes série. Placez les cinq PC « Nuage » et configurez-les afin que chacun d'eux soit connecté à l'adaptateur réseau convenable et donc au concentrateur VMnet convenable. Établissez les liens LAN et WAN. Nommez les différents équipements afin de refléter la topologie proposée.

- Démarrez R100 et selon une méthode à votre convenance, injectez-y la configuration fournie en téléchargement sur le site ENI (fichier Atelier8a_R100.txt).

- Démarrez R200 et selon une méthode à votre convenance, injectez-y la configuration fournie en téléchargement sur le site ENI (fichier Atelier8a_R200.txt).

- Ouvrez une seconde instance de VMware et avec elle, créez une équipe de quatre PC PC101, PC102, PC201 et PC202. Réutilisez les PC créés lors des ateliers précédents (PC11, PC12, PC21 et PC22). Pour mémoire, PC101 est obtenu par clonage sans lien (option **create a full clone**) de la machine Ubuntu fournie sur le site ENI. PC102, PC201 et PC202 sont obtenus par clone avec lien (option **linked clone**) de PC101. Ajoutez le serveur VMSRV01 à l'équipe. Nommez cette équipe 5PC, faites en sorte que l'adaptateur réseau virtuel de chacune des machines soit connecté au concentrateur VMnet convenable, respectivement VMnet1 à VMnet4 ainsi que VMnet8. Démarrez totalement ou partiellement l'équipe puis réglez chacune des cinq configurations IP. Il est rare que l'on ait besoin de ces PC ensemble. Suspendez-les et ne sortez un PC de son sommeil que lorsque l'activité le demande.

b. Tâche 2 : Activer les interfaces dont l'interface de loopback

- Réalisez la configuration des interfaces de R101 de la façon suivante :

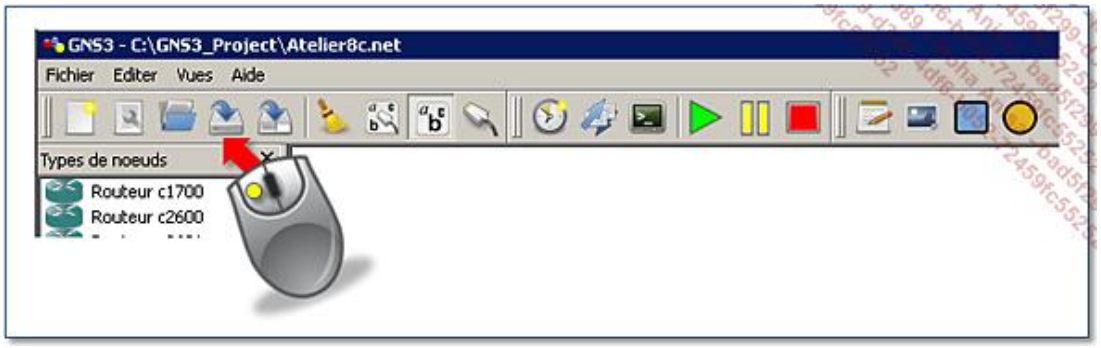
```
Router#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#hostname R101
R101(config)#line con 0
R101(config-line)#logging synchronous
R101(config-line)#exec-timeout 0
R101(config-line)#exit
R101(config)#int f0/0
R101(config-if)#ip address 172.26.101.1 255.255.255.128
R101(config-if)#no shutdown
R101(config-if)#int f0/1
R101(config-if)#ip address 172.26.100.101 255.255.255.0
R101(config-if)#no shutdown
R101(config-if)#int s0/0
```

```
R101(config-if)#ip address 172.26.50.1 255.255.255.252
R101(config-if)#no shutdown
R101(config-if)#bandwidth 2500
R101(config-if)#int loopback 0
R101(config-if)#ip address 192.168.100.101 255.255.255.255
R101(config-if)#^Z
R101#
```

- Sauvegardez de façon régulière :

```
R101#wr
Building configuration...
```

- Faite suivre chaque sauvegarde dans l'interface ILC d'une sauvegarde de la topologie sous GNS3 :



- En utilisant, les adresses appropriées, réalisez de façon identique les configurations de R102 à R104.

- Vérifiez l'état des interfaces. Par exemple, sur R101 :

```
R101#sh ip int br
Interface                  IP-Address      OK? Method Status
Protocol
FastEthernet0/0            172.26.101.1    YES NVRAM  up                    up
Serial0/0                  172.26.50.1     YES NVRAM  up                    up
FastEthernet0/1            172.26.100.101  YES NVRAM  up                    up
Loopback0                  192.168.100.101 YES manual up                    up
```

c. Tâche 3 : Activer le processus EIGRP

- Configurez sur R101 un processus de routage EIGRP en affectant au numéro d'AS la partie numérique du nom du routeur :

```
R101#conf t
R101(config)#router eigrp 101
R101(config-router)#
```

- Hormis l'interface de loopback, toutes les interfaces du routeur R101 appartiennent au réseau majeur 172.26.0.0. Faites-les participer au protocole à l'aide d'une seule commande **network** :

```
R101(config-router)#network 172.26.0.0
R101(config-router)#
```

- Vérifiez le processus de routage exécuté par R101 :

```
R101#sh ip protocols
Routing Protocol is "eigrp 101"
```

```
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
  EIGRP metric weight K1=1, K2=0, K3=1, K4=0, K5=0
  EIGRP maximum hopcount 100
  EIGRP maximum metric variance 1
  Redistributing: eigrp 101
  Automatic network summarization is in effect
  Maximum path: 4
  Routing for Networks:
    172.26.0.0
  Routing Information Sources:
    Gateway         Distance      Last Update
Distance: internal 90 external 170

R101#
```

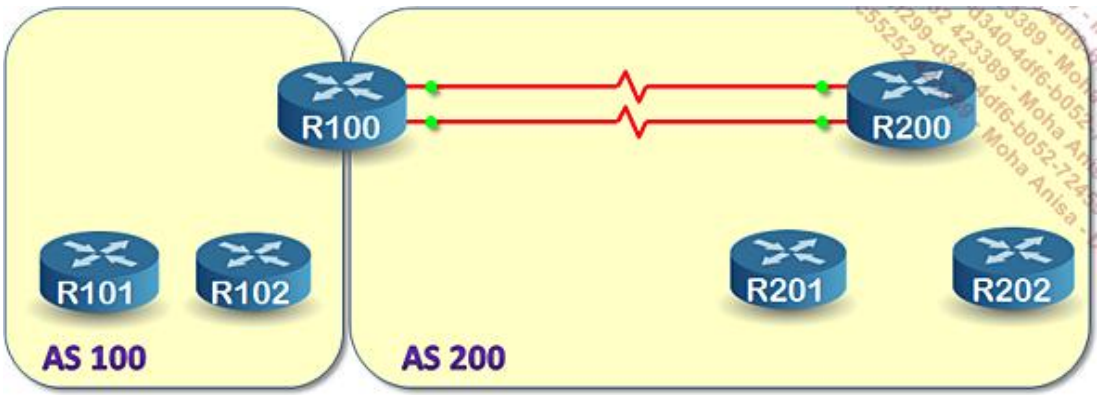
- Observez notamment la plage d'adresses couverte par le processus sous la section « *Routing for Networks* » : …… (complétez).

- Observez les distances administratives attribuées par EIGRP selon qu'une route est interne ou externe : …… (complétez).

- Attention au contexte, R100 et R200 sont fonctionnels avec la configuration proposée. Vérifiez le contenu de la table de routage de R101 :

```
R101#sh ip route
.........
Gateway of last resort is not set

     172.26.0.0/16 is variably subnetted, 3 subnets, 3 masks
C       172.26.50.0/30 is directly connected, Serial0/0
C       172.26.100.0/24 is directly connected, FastEthernet0/1
C       172.26.101.0/25 is directly connected, FastEthernet0/0
     192.168.100.0/32 is subnetted, 1 subnets
C       192.168.100.101 is directly connected, Loopback0
R101#
```

Sauf erreur, la table de routage de R101 ne devrait contenir que les réseaux directement connectés. En effet, le système autonome 101 ne comprend que le routeur R101. Par défaut, un système autonome est en quelque sorte « étanche ». Si l'administrateur souhaite lui faire exploiter des informations de route issues d'autres AS, il doit configurer une redistribution de routes. Il se trouve que notre topologie de test a été divisée en deux domaines EIGRP (deux AS) 100 et 200. R200 exécute le processus EIGRP 200. R100 exécute les deux processus EIGRP 100 et 200.



On se propose donc de placer R101 et R102 dans l'AS 100 puis de placer R201 et R202 dans l'AS 200.

- À l'aide d'une commande **no router eigrp 101**, supprimez le processus correspondant sur le routeur R101 :

```
R101#conf t
```

```
Enter configuration commands, one per line.  End with CNTL/Z.
R101(config)#no router eigrp 101
```

- Créez sur R101 un processus EIGRP avec le numéro d'AS 100 sans omettre la commande **network** adéquate :

```
R101(config)#router eigrp 100
R101(config-router)#network 172.26.0.0
R101(config-router)#^Z
R101#
```

- Répétez cette configuration sur R102.

- Répétez cette configuration sur R201 et R202 mais en leur faisant partager l'information de routage de l'AS 200. Par exemple sur R201 :

```
R201(config)#router eigrp 200
R201(config-router)#network 172.26.0.0
R201(config-router)#^Z
R201#
```

- Immédiatement après avoir créé un processus EIGRP et inclus des interfaces, observez l'établissement des relations de voisinage du protocole. Par exemple sur R101 :

```
00:27:04:%DUAL-5-NBRCHANGE: IP-EIGRP 100: Neighbor 172.26.100.102 (FastEthernet0/1) is
up: new adjacency
00:27:05: %DUAL-5-NBRCHANGE: IP-EIGRP 100: Neighbor 172.26.50.2 (Serial0/0) is up: new
adjacency
00:27:05: %DUAL-5-NBRCHANGE: IP-EIGRP 100: Neighbor 172.26.100.100 (FastEthernet0/1)
is up: new adjacency
```

- Il est temps d'observer à nouveau les tables de routage. Sur R101 :

```
R101#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is not set

     172.26.0.0/16 is variably subnetted, 9 subnets, 4 masks
D EX    172.26.202.0/25
[170/1159680] via 172.26.100.100, 00:00:50, FastEthernet0/1
D EX    172.26.200.0/24
           [170/1157120] via 172.26.100.100, 00:00:50, FastEthernet0/1
D EX    172.26.201.0/25
[170/1159680] via 172.26.100.100, 00:00:50, FastEthernet0/1
C       172.26.50.0/30 is directly connected, Serial0/0
D EX    172.26.60.0/30
           [170/2053120] via 172.26.100.100, 00:00:50, FastEthernet0/1
D       172.26.120.0/29
           [90/1154560] via 172.26.100.100, 00:00:50, FastEthernet0/1
D       172.26.102.0/25
           [90/30720] via 172.26.100.102, 00:00:50, FastEthernet0/1
C       172.26.100.0/24 is directly connected, FastEthernet0/1
C       172.26.101.0/25 is directly connected, FastEthernet0/0
     192.168.200.0/32 is subnetted, 1 subnets
D EX    192.168.200.200
           [170/1282560] via 172.26.100.100, 00:00:50, FastEthernet0/1
     192.168.100.0/32 is subnetted, 2 subnets
D       192.168.100.100
           [90/156160] via 172.26.100.100, 00:00:50, FastEthernet0/1
C       192.168.100.101 is directly connected, Loopback0
R101#
```

Observez les réseaux non directement connectés, le protocole de routage dont ils sont issus. La lettre D rappelle le protocole EIGRP (D pour l'algorithme DUAL). La lettre D accompagnée de l'acronyme EX rappelle que la route est externe à l'AS. Dans le cas présent, nous sommes placés sur un routeur de l'AS 100 et observons des routes de l'AS 200.

Observez les distances administratives associées aux routes, 170 dans le cas d'une route externe à l'AS apprise par EIGRP, 90 dans le cas d'une route interne à l'AS.

Observez la présence dans la table de l'adresse de loopback du routeur R101 mais l'absence de celles des routeurs R102, R201 et R202. En revanche, celle des routeurs fédérateurs R100 et R200 apparaît. Que diable avons-nous oublié ? La commande **network** idoine bien sûr qui ferait participer l'interface de loopback au protocole !

- Sur R101 et R102, ajoutez la commande **network 192.168.100.0** en configuration de routeur EIGRP 100 :

```
R101#conf t
R101(config)#router eigrp 100
R101(config-router)#network 192.168.100.0
R101(config-router)#
R101(config-router)#^Z
R101#
```

- Sur R201 et R202, ajoutez la commande **network 192.168.200.0** en configuration de routeur EIGRP 200 :

```
R201#conf t
R201(config)#router eigrp 200
R201(config-router)#network 192.168.200.0
R201(config-router)#
R201(config-router)#^Z
R201#
```

- Tentez un ping depuis R101 vers l'adresse de loopback de l'un des routeurs R102 ou R201. Est-ce que le ping aboutit ? Peu probable car une nouvelle observation de la table de routage de R101 montre qu'il n'existe pas de route vers 192.168.100.102. En revanche, il existe une route mais une seule vers le réseau majeur 192.168.200.0 (extrait) :

```
R101#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP

192.168.200.0/24 is variably subnetted, 2 subnets, 2 masks
D EX    192.168.200.200/32
           [170/1282560] via 172.26.100.100, 00:00:21, FastEthernet0/1
D EX    192.168.200.0/24
           [170/1285120] via 172.26.100.100, 00:00:21, FastEthernet0/1
     192.168.100.0/24 is variably subnetted, 3 subnets, 2 masks
D       192.168.100.0/24 is a summary, 00:06:26, Null0
D       192.168.100.100/32
           [90/156160] via 172.26.100.100, 00:00:22, FastEthernet0/1
C       192.168.100.101/32 is directly connected, Loopback0
R101#
```

À ce stade, il faut se souvenir du comportement par défaut d'EIGRP qui agrège automatiquement aux frontières du réseau majeur. Ainsi, chacun des deux routeurs R101 et R102 annonce le réseau majeur 192.168.100.0. De la même façon, chacun des deux routeurs R201 et R202 annonce le réseau majeur 192.168.200.0. Rendre la vue à ces routeurs et leur donner le moyen d'ajouter des routes vers les adresses de loopback implique de désactiver l'agrégation automatique.

- Désactivez l'agrégation automatique sur chacun des quatre routeurs R101, R102, R201 et R202. Par exemple, sur R101 :

```
R101#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R101(config)#router eigrp 100
R101(config-router)#no auto-summary
R101(config-router)#^Z
R101#
```

- Vérifiez la connectivité :

```
R101#conf t
R101(config)#ip host R102 192.168.100.102
R101(config)#ip host R201 192.168.200.201
R101(config)#ip host R202 192.168.200.202
R101(config)#^Z
R101#ping R102
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.100.102, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/57/188 ms
R101#ping R201
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.200.201, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 48/85/100 ms
```

- Vérifiez une dernière fois les tables de routage et observez que cette fois, les adresses de loopback sont bien visibles (extrait) :

```
R101#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
     192.168.200.0/32 is subnetted, 3 subnets
D EX    192.168.200.200
[170/1282560] via 172.26.100.100, 00:01:51, FastEthernet0/1
D EX    192.168.200.201
           [170/1285120] via 172.26.100.100, 00:00:50, FastEthernet0/1
D EX    192.168.200.202
[170/1285120] via 172.26.100.100, 00:00:22, FastEthernet0/1
     192.168.100.0/32 is subnetted, 3 subnets
D       192.168.100.100
           [90/156160] via 172.26.100.100, 00:01:51, FastEthernet0/1
C       192.168.100.101 is directly connected, Loopback0
D       192.168.100.102
           [90/156160] via 172.26.100.102, 00:01:51, FastEthernet0/1
R101#
```

d. Tâche 4 : Agir sur la bande passante des liens série

- Si cela n'avait pas déjà été fait, établissez la bande passante de l'interface S0/0 à 2500 Kbps sur chacun des quatre routeurs R101, R102, R201 et R202. Par exemple sur R101 :

```
R101#conf t
R101(config)#int s0/0
R101(config-if)#bandwidth 2500
R101(config-if)#^Z
R101#
```

- Collectez l'information utile au calcul de métrique à l'aide de commandes **show interface**. Sur R101 :

```
R101#sh int s0/0
Serial0/0 is up, line protocol is up
  Hardware is PowerQUICC Serial
  Internet address is 172.26.50.1/30
  MTU 1500 bytes, BW 2500 Kbit, DLY 20000 usec,
reliability 255/255, txload 1/255, rxload 1/255
.........
R101#sh int f0/0
FastEthernet0/0 is up, line protocol is up
  Hardware is AmdFE, address is c804.0448.0000 (bia c804.0448.0000)
```

```
  Internet address is 172.26.101.1/25
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
reliability 255/255, txload 1/255, rxload 1/255
.........
R101#
```

- Sur R101, observez le coût de la route vers LAN_12 (172.26.102.0/25) à l'aide d'une commande **show ip route** (Extrait) :

```
R101#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is not set

     172.26.0.0/16 is variably subnetted, 4 subnets, 3 masks
C       172.26.50.0/30 is directly connected, Serial0/0
D       172.26.102.0/25
           [90/30720] via 172.26.100.102, 00:00:02, FastEthernet0/1
C       172.26.100.0/24 is directly connected, FastEthernet0/1
C       172.26.101.0/25 is directly connected, FastEthernet0/0
.........
R101#
```

Comment R101 est-il parvenu à ce résultat ? Condensons ses calculs dans le tableau suivant :

	BW (Kbps)	**DLY (us)**	
Int F0/0 de R102	100 000	100	
Int F0/1 de R101	100 000	100	
Retenu par le calcul IGRP	Valeur mini de la colonne soit : 100 000	Somme des délais de la colonne soit : 200	Métrique IGRP résultante : $\frac{10^7}{10^5} + \frac{200}{10} = 120$
Métrique EIGRP			256 x 120 = 30 720

Le routeur R102 connaît le réseau 172.26.102.0/25, ce réseau lui est directement connecté via l'interface F0/0. R102 annonce par conséquent les valeurs *BW* 100 000 et *DLY* 100. R101 reçoit cette annonce sur son interface F0/1 et calcule le coût de la route en faisant intervenir la bande passante et le délai de l'interface qui reçoit.

- Provoquez le passage à l'état down de l'interface F0/1 du routeur R101 :

```
R101#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R101(config)#int f0/1
R101(config-if)#shutdown
R101(config-if)#^Z
R101#
```

- Observez le nouveau coût de la route vers LAN_12 (Extrait) :

```
R101#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is not set

     172.26.0.0/16 is variably subnetted, 4 subnets, 3 masks
C       172.26.50.0/30 is directly connected, Serial0/0
```

```
D       172.26.102.0/25 [90/1538560] via 172.26.50.2, 00:00:39, Serial0/0
D       172.26.100.0/24 [90/1538560] via 172.26.50.2, 00:00:39, Serial0/0
C       172.26.101.0/25 is directly connected, FastEthernet0/0
.........
```

Comment R101 est-il parvenu à ce résultat ? Condensons ses calculs dans le tableau suivant :

	BW (Kbps)	**DLY (µs)**	
Int F0/0 de R102	100 000	100	
Int S0/0 de R101	2 500	20 000	
Retenu par le calcul IGRP	Valeur mini de la colonne soit : 2 500	Somme des délais de la colonne soit : 20 100	Métrique IGRP résultante : $\frac{10^7}{2500} + \frac{20\,100}{10} = 6010$
Métrique EIGRP			256 x 6 010 = 1538 560

L'IOS est capable de déterminer la bande passante d'une interface LAN. En revanche, c'est à l'administrateur qu'il revient de configurer la vraie bande passante des liens WAN. S'il ne le fait pas, l'IOS adopte la bande passante par défaut qui est celle d'un lien T1 (US oblige) soit 1544 Kbps. La métrique composite d'EIGRP est l'un des points forts du protocole mais pour qu'elle fournisse le service attendu, il faut en quelque sorte « l'alimenter ».

e. Tâche 5 : Observer la base de données topologique d'EIGRP

- Mettez à profit la commande **show ip eigrp neighbors** pour afficher les relations de voisinage établies. Par exemple, sur R101 :

```
R101#sh ip eigrp neighbors
IP-EIGRP neighbors for process 100
H   Address                 Interface   Hold Uptime   SRTT   RTO  Q  Seq Type
                                        (sec)         (ms)       Cnt Num
1   172.26.50.2             Se0/0         13 00:01:43   68   408  0  19
2   172.26.100.102          Fa0/1         12 00:01:57   96   576  0  17
0   172.26.100.100          Fa0/1         14 00:02:23   91   546  0  19
R101#
```

Le résultat de cette commande doit refléter la topologie du réseau. Dans le cas présent, on apprend que R101 a établi deux relations de voisinage avec R102 et une avec R100.

- Affichez le contenu de la base de données topologique à l'aide d'une commande **show ip eigrp topology**. Par exemple, sur R101 (Extrait, les adresses loopback ne sont pas représentées) :

```
R101#sh ip eigrp topology
IP-EIGRP Topology Table for AS(100)/ID(192.168.100.101)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status
.........
P 172.26.202.0/25, 1 successors, FD is 1159680, tag is 200
  via 172.26.100.100 (1159680/1157120), FastEthernet0/1
P 172.26.200.0/24, 1 successors, FD is 1157120, tag is 200
  via 172.26.100.100 (1157120/1154560), FastEthernet0/1
P 172.26.201.0/25, 1 successors, FD is 1159680, tag is 200
  via 172.26.100.100 (1159680/1157120), FastEthernet0/1
P 172.26.50.0/30, 1 successors, FD is 1536000
  via Connected, Serial0/0
P 172.26.60.0/30, 1 successors, FD is 2053120, tag is 200
  via 172.26.100.100 (2053120/2050560), FastEthernet0/1
P 172.26.120.0/29, 1 successors, FD is 1154560
  via 172.26.100.100 (1154560/1152000), FastEthernet0/1
```

```
P 172.26.102.0/25, 1 successors, FD is 30720
  via 172.26.100.102 (30720/28160), FastEthernet0/1
  via 172.26.50.2 (1538560/28160), Serial0/0
P 172.26.100.0/24, 1 successors, FD is 28160
  via Connected, FastEthernet0/1
P 172.26.101.0/25, 1 successors, FD is 28160
  via Connected, FastEthernet0/0
R101#
```

Observez par exemple les données qui concernent la route vers LAN_12. EIGRP indique une distance faisable de 30720 (« *FD is 30720* »). C'est donc que la route choisie est la route via 172.26.100.102. Le routeur annonçant, dans le cas présent R102, a un coût de 28160. EIGRP a pu placer dans sa base une route alternative (le plan B) qui respecte le critère de faisabilité, c'est-à-dire dont le coût du routeur annonçant est inférieur à la distance faisable en cours (28160 < 30720).

f. Tâche 6 : Bonus → Activation du relais DHCP

Objectif

L'administrateur souhaite mettre en place un serveur DHCP unique qui distribuerait des adresses sur les différents réseaux LAN_11, LAN_12, LAN_21 et LAN_22.

Commentaire

Le service DHCP est décrit de façon précise dans l'ouvrage Cisco - Notions de base sur les réseaux dans la collection Certifications aux Editions ENI. Un serveur DHCP (*Dynamic Host Configuration Protocol*) peut fournir des adresses IP à des clients situés sur des sous-réseaux différents à la condition que le routeur qui sépare chaque sous-réseau du serveur puisse fonctionner en tant qu'agent relais. Cette fonctionnalité est décrite dans le RFC 1542. L'agent relais conforme au RFC relaie les messages DHCP sur le côté serveur même quand il s'agit de paquets diffusés. Avant de relayer le message d'un client DHCP, l'agent examine le champ **giaddr** (*Gateway IP Address*) du message. Si ce champ porte l'adresse 0.0.0.0, l'agent relais le complète avec l'adresse IP du routeur, en réalité l'adresse IP de l'interface du routeur qui a reçu la requête.

Lorsque le serveur DHCP reçoit le message, il examine ce champ devenu champ adresse IP du relais afin de déterminer si oui ou non il dispose d'un pool d'adresses IP (une étendue DHCP dans le vocabulaire Microsoft) pour le sous-réseau en question.

Le serveur DHCP doit disposer d'autant de pools d'adresses qu'il y a de sous-réseaux à desservir.

Lorsqu'il reçoit le message DHCPDISCOVER, le serveur DHCP envoie un DHCPOFFER directement à l'agent de relais identifié dans le champ **giaddr** puis l'agent transmet le message au client DHCP. À ce stade, l'adresse IP du client est toujours inconnue, l'agent relais doit donc diffuser le message DHCPOFFER sur le sous-réseau. La séquence se poursuit, un message DHCPREQUEST est relayé du client au serveur et un message DHCPACK est relayé du serveur au client, conformément au RFC 1542.

Étape 1 : mise en place du serveur DHCP

Si le lecteur s'en est tenu aux ateliers proposés, il dispose à cet instant d'une machine virtuelle VMSRV01 qui exécute le système d'exploitation Windows 2000 Server. On se souvient que ce système a été choisi parce qu'il offre tous les services réseau tout en restant compact (selon les critères actuels).

- Sur VMSRV01, assurez-vous que l'adaptateur réseau est connecté au VMnet convenable (VMnet8) afin d'assurer la connectivité avec le réseau LAN_100 de notre topologie. Réglez la configuration IP : Adresse 172.26.100.254/24, passerelle 172.26.100.100.

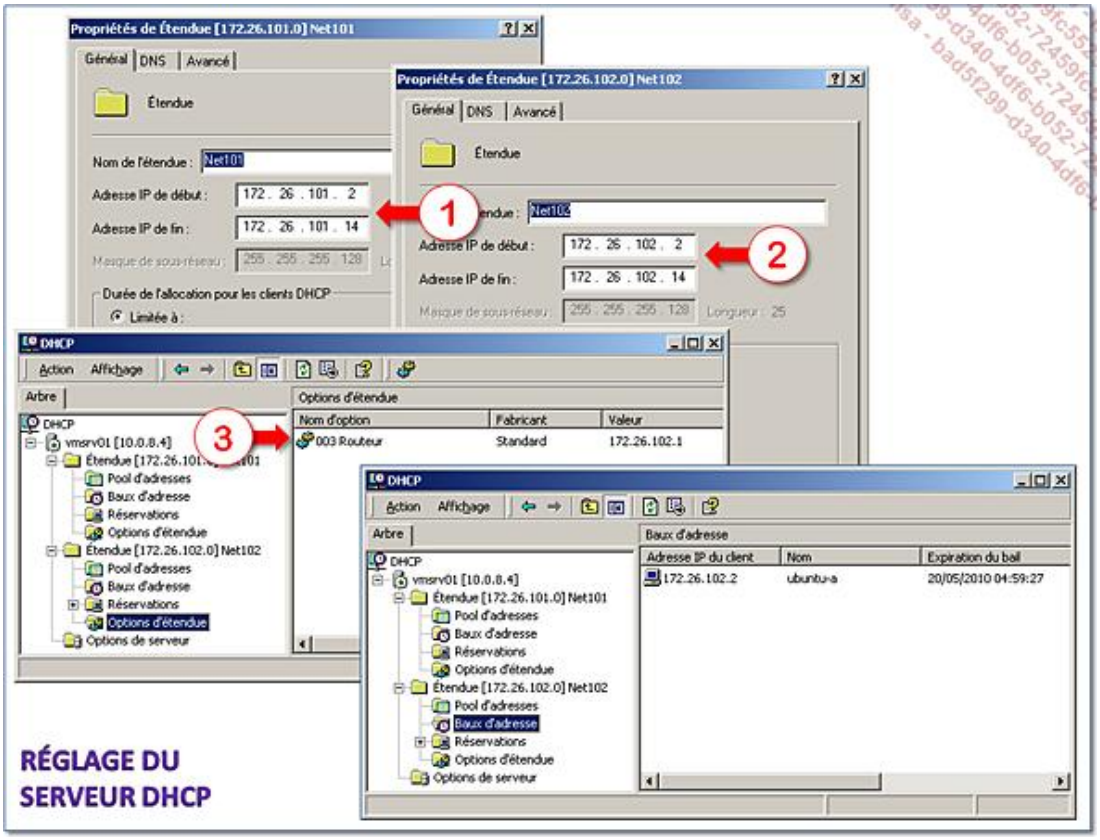
- Si ce n'est pas déjà fait, installez le service DHCP : **Panneau de configuration - Ajout/Suppression de programmes - Ajouter/Supprimer des composants Windows - Services de mise en réseau/Détails**, cochez la case **Protocole DHCP**.

- Ouvrez la console d'administration du service DHCP : **Menu Démarrer - Programmes - Outils d'administration - DHCP**.

- Créez une étendue pour le sous-réseau LAN_11, nommez cette étendue Net101, elle comprend les adresses 172.26.101.2 à 172.26.101.14, masque /25. Configurez l'option d'étendue Routeur afin que le client DHCP obtienne également son adresse de passerelle.

- Répétez ces opérations afin de créer trois autres étendues pour les réseaux LAN_12, LAN_21 et LAN_22.

La figure suivante illustre en partie ces opérations :



Étape 2 : activation du relais DHCP

- En configuration d'interface, la commande **ip helper-address** active le relais DHCP. Sur l'interface F0/0 de R101, soit l'interface connectée aux clients DHCP du réseau LAN_11, activez le relais :

```
R101(config)#int f0/0
R101(config-if)#ip help
R101(config-if)#ip helper-address 172.26.100.254
R101(config-if)#^Z
R101#
```

- Répétez l'opération pour les interfaces F0/0 des trois routeurs R102, R201 et R202.

Étape 3 : activation du client et recette de la solution

- Sur chacune des machines virtuelles clientes, remplacez la configuration IP statique par une configuration dynamique. Si les machines sont celles préconisées (Ubuntu), la procédure est décrite dans le chapitre Annexes. Pour mémoire :

- Configurez l'interface réseau (mot de passe *sunrise*) :

```
$ sudonano /etc/network/interfaces
```

- Activez le client DHCP (remplacez eth0 par ethx, votre eth) :

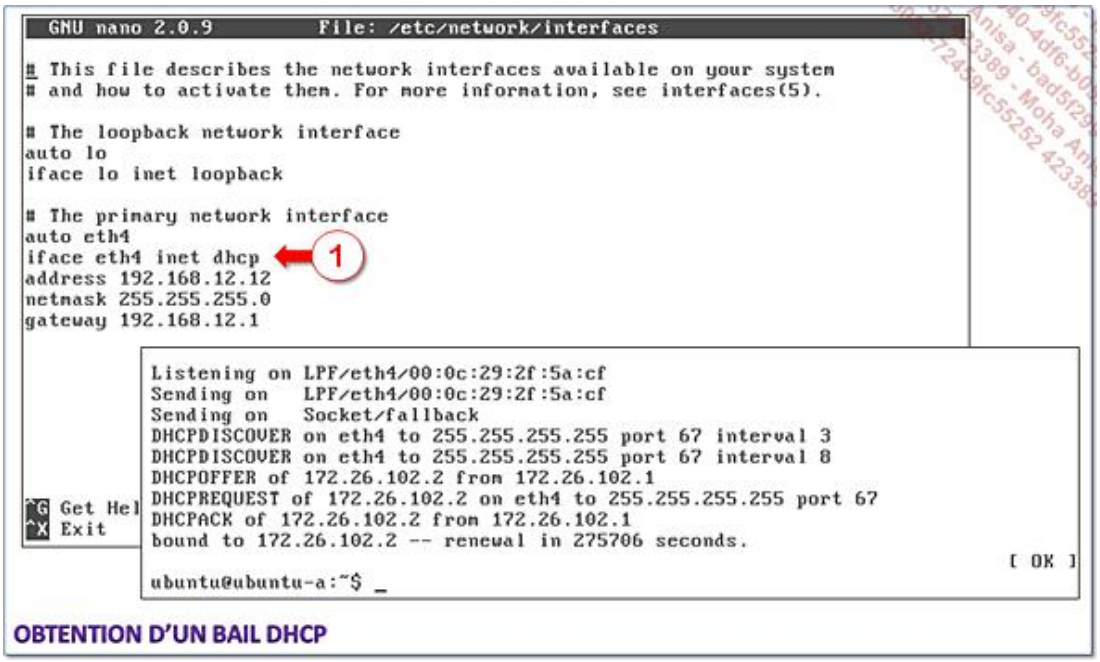
```
auto lo
iface lo inet loopback
auto eth0
```

```
iface eth0 inet dhcp
```

- Quittez par [Ctrl] X, acceptez le lieu et le nom de sauvegarde en tapant Y, puis confirmez par la touche [Entrée].
- Provoquez un redémarrage du réseau afin de prendre en compte la modification de la configuration :

```
$ sudo /etc/init.d/networking restart
```

La figure suivante illustre en partie ces opérations sur la machine PC102 de notre topologie :



Vos clients ont obtenu leur configuration IP. Bravo. Cet atelier est maintenant terminé.

3. Configuration avancée

a. Authentification

EIGRP peut authentifier ses messages et c'est le moyen pour l'administrateur de s'assurer que les routeurs n'accepteront que les messages issus de leurs pairs, c'est-à-dire les messages issus de routeurs partageant la même clé, dans le domaine dont il a la charge. Ainsi, un routeur ou toute source d'information de routage non approuvée ne risque pas de polluer les tables du domaine EIGRP avec de l'information illicite, ce quelle que soit l'intrusion, accidentelle ou malveillante.

C'est l'atelier 7B qui a servi de support pour illustrer la mise en œuvre de l'authentification. La création de clés a déjà été décrite, merci de vous reporter à la section Activation de l'authentification du chapitre Protocoles de routage type DV RIPv2. L'ensemble de clés Turing a été créé sur chacun des routeurs concernés, pour cet atelier nous nous en sommes tenus à authentifier les échanges entre RTR7C et RTR7A :

```
RTR7A#sh run
Building configuration...
.........
key chain Turing
  key 1
  key-string couronne
  accept-lifetime 05:00:00 Jan 1 2010 duration 90000
  send-lifetime 05:00:00 Jan 1 2010 duration 90000
  key 2
  key-string sicie
  accept-lifetime 05:00:00 Jan 2 2010 06:00:00 Feb 2 2010
  send-lifetime 05:00:00 Jan 2 2010 06:00:00 Feb 2 2010
  key 3
  key-string cepet
```

```
  accept-lifetime 05:00:00 Feb 2 2010 infinite
  send-lifetime 05:00:00 Feb 2 2010 infinite
call rsvp-sync
!
```

Il reste à appliquer l'authentification aux interfaces concernées de chacun des routeurs. La commande **ip authentication mode eigrp** en configuration d'interface spécifie le type d'authentification à mettre en œuvre :

```
Router(config-if)#ip authentication mode eigrp as-number md5
```

... mais au moment où ces lignes sont écrites, le seul mode prévu est MD5.

Toujours en configuration d'interface, une seconde commande est nécessaire pour activer l'authentification proprement dite :

```
Router(config-if)#ip authentication key-chain eigrp as-number key-chain
```

Appliquées au cas présent :

```
.........
interface Serial0/0
  ip address 192.168.0.250 255.255.255.252
  ip authentication mode eigrp 64501 md5
  ip authentication key-chain eigrp 64501 Turing
!
.........
RTR7A#
```

L'administrateur souhaitera probablement superviser l'authentification :

```
RTR7A#show key chain
Key-chain Turing:
  key 1 -- text "couronne"
  accept lifetime (05:00:00 UTC Jan 1 2010) - (90000 seconds)
  send lifetime (05:00:00 UTC Jan 1 2010) - (90000 seconds)
  key 2 -- text "sicie"
  accept lifetime (05:00:00 UTC Jan 2 2010) - (06:00:00 UTC Feb 2 2010)
  send lifetime (05:00:00 UTC Jan 2 2010) - (06:00:00 UTC Feb 2 2010)
  key 3 -- text "cepet"
  accept lifetime (05:00:00 UTC Feb 2 2010) - (infinite) [valid now]
  send lifetime (05:00:00 UTC Feb 2 2010) - (infinite) [valid now]
RTR7A#
```

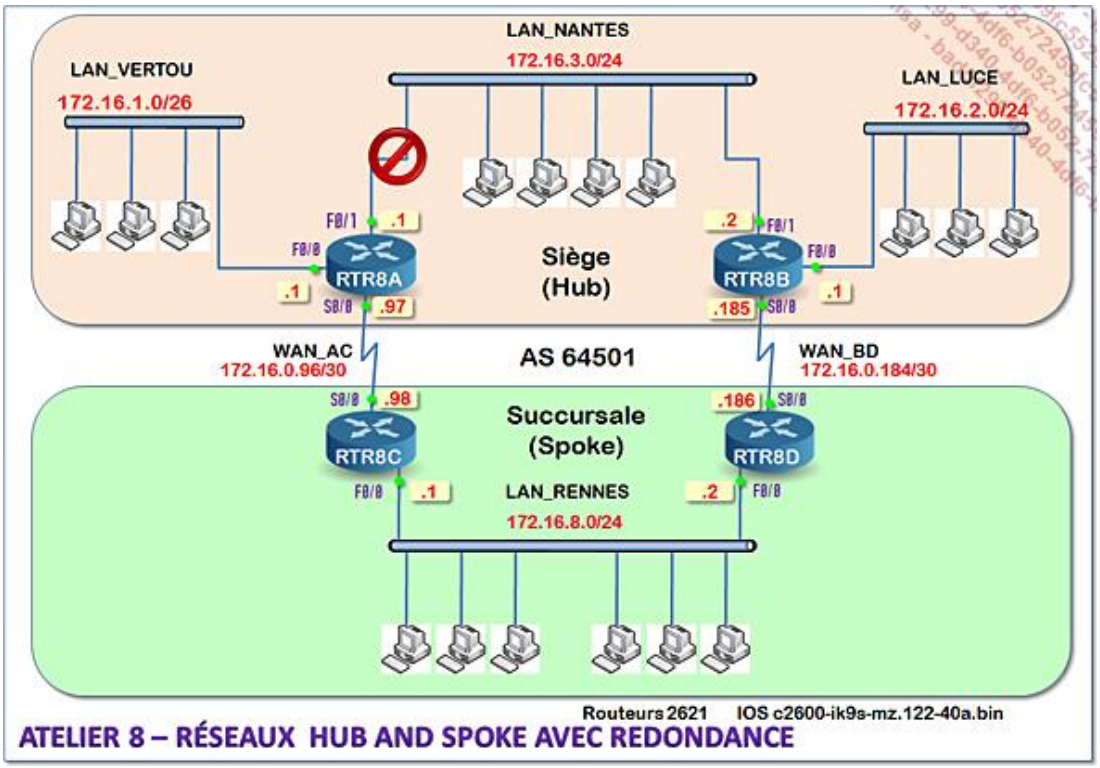
Une commande **debug** est également instructive mais évidemment plus dangereuse :

```
RTR7A#debug eigrp packets
EIGRP Packets debugging is on
    (UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK, STUB, SIAQUERY,
SIAREPLY)
00:12:55: EIGRP: received packet with MD5 authentication, key id = 3
00:12:55: EIGRP: Received HELLO on Serial0/0 nbr 192.168.0.249
00:12:55:   AS 64501, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely
0/0
RTR7A#
```

On découvre ainsi, c'est une confirmation, que parmi les clés qui composent l'ensemble de clés, la clé en service est la clé n°3.

b. Routeur EIGRP de bout

Considérez la topologie ci-après :



L'administrateur en charge de ce réseau a pris soin d'assurer la redondance des liens qui relient les deux sites. Mais le mieux étant toujours l'ennemi du bien, l'administrateur s'est ainsi créé un certain nombre de problèmes qu'il ne soupçonnait pas au début de sa réflexion. Imaginez par exemple que le lien qui relie RTR8A au commutateur de LAN_NANTES fasse défaut. C'est entendu, le trafic intersites bascule vers RTR8B. Mais qu'advient-il du trafic entre LAN_VERTOU et LAN_LUCE ? Vous dites ? Interrompu ? Ne transiterait-il pas plutôt par les quatre routeurs ? Hélas oui et il est peu probable que les liens WAN puissent accepter ce trafic et même s'ils le pouvaient, ce serait probablement une aberration économique (Un trafic local qui effectue un aller-retour par un site distant !) De plus, la bande passante consommée le sera au détriment du trafic utile intersites.

Alors bien sûr, on peut imaginer moult solutions comme par exemple ajouter de la redondance entre les deux routeurs du siège, sous forme de liens Ethernet supplémentaires. Mais puisqu'EIGRP a été choisi sur cette topologie, peut-être peut-il apporter une solution à peu de frais, juste le prix d'un peu de configuration.

Qu'est-ce qui caractérise le réseau de la succursale à Rennes ? C'est un réseau d'extrémité. Autrement dit, les paquets qui arrivent sur le réseau de la succursale lui sont destinés. Par définition, un routeur d'extrémité ne peut que remettre les paquets au réseau de destination. Il est donc inutile de l'interroger sur la façon de faire progresser des paquets comme on le ferait sur un routeur de transit.

EIGRP est capable de distinguer un routeur d'extrémité et un routeur de transit. La fonctionnalité correspondante, dite EIGRP « Stub Router », a été introduite dans la version 12.0(7)T de l'IOS. Un routeur configuré comme routeur d'extrémité ou routeur de bout utilise également des messages Hello pour établir puis entretenir ses relations de voisinage. Mais ces messages Hello sont un peu différents et intègrent un triplet TLV supplémentaire, le TLV « *Stub* », qui permet au routeur de se proclamer routeur de bout.

De plus, un routeur de bout peut n'annoncer que sa connaissance de certaines catégories de routes, selon le choix de l'administrateur. La commande permettant de faire d'un routeur un routeur de bout est **eigrp stub** en configuration de routeur. Sa syntaxe est la suivante :

```
Router(config-router)#eigrp stub [receive-only | connected | static | summary |
redistributed]
```

... dont les arguments sont les suivants :

receive-only

Optionnel, fait de ce routeur de bout EIGRP un routeur silencieux (analogue à l'hôte silencieux de RIP).

connected

Optionnel, n'annonce que les routes directement connectées.

static

Optionnel, n'annonce que les routes directement statiques.

summary

Optionnel, n'annonce que les routes agrégées.

redistributed

Optionnel, n'annonce que les routes issues d'autres protocoles de routage, par exemple OSPF.

La commande **eigrp stub** ne doit être entrée que sur les routeurs de bout (que les documentations anglaises désignent par « *Spoke* »). Le routeur « *Hub* » (un routeur du site central connecté à plusieurs routeurs « *Spoke* ») reconnaît les routeurs de bout grâce à leurs messages Hello particuliers et adapte son comportement en conséquence.

Un routeur « Hub » n'interroge pas un routeur de bout !

Autrement dit, un routeur de bout ne participe pas aux procédures de calcul diffusé.

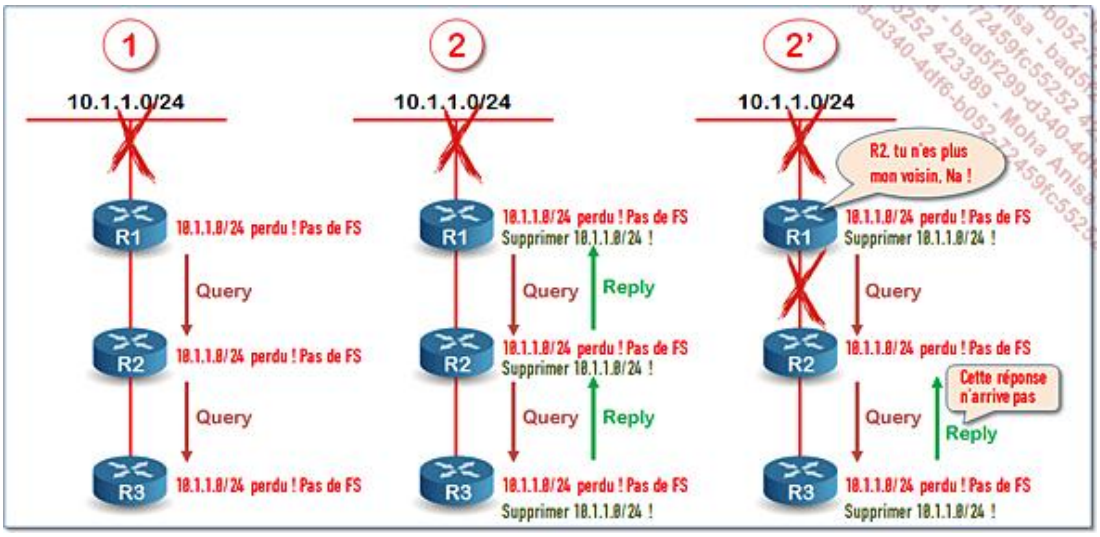
Dans notre contexte, il reste à faire de RTR8C et RTR8D deux routeurs de bout. Un dernier avertissement au lecteur : il est très facile de rendre le routage inopérant dans une topologie avec routeurs de bout. Il suffit d'ajouter le mot-clé « **receive-only** » à la commande **eigrp stub**. Le routeur n'annonce plus rien et donc le routeur « Hub » n'apprend pas le réseau de la succursale. Il semble que ce piège soit fréquent dans les configurations proposées lors des tests de certification.

c. Routes SIA

Les routes SIA (*Stuck-In-Active*, littéralement « coincée dans l'état actif ») ont été évoquées dans la section dédiée à DUAL. Pour mémoire, le problème survient quand une procédure de calcul diffusé est entreprise et donc que la route concernée est passée à l'état actif. Chaque routeur interrogé et qui ne peut répondre interroge ses voisins à son tour, la procédure s'étend progressivement sur l'ensemble du réseau.

On sent confusément que la probabilité pour qu'une question reste sans réponse augmente de la même façon que le diamètre du réseau. L'important n'est pas seulement que la réponse arrive mais qu'elle arrive dans les temps. Une réponse qui arrive hors délai n'est pas nécessairement causée par un dysfonctionnement, le chemin à parcourir « aller et retour » était simplement trop long et le nombre de routeurs rencontrés sur ce chemin trop important. Certains routeurs, peut-être congestionnés, ont tardé à répondre ou à émettre leurs propres requêtes. L'un des liens transite peut-être par une liaison satellitaire, ces liaisons peuvent offrir des débits importants mais jamais de courts délais.

Au moins jusqu'à la version 12.2 de l'IOS, la protection vis-à-vis de questions qui restent sans réponse passe par le temporisateur de route active. La figure suivante tente de résumer la chronologie des évènements :



La chronologie normale est la suivante (étiquettes 1 et 2 de l'illustration) :

- R1 perd la route 10.1.1.0/24. La phase locale de réévaluation n'a pas donné de successeur faisable, R1 marque la route active et entame une procédure de calcul diffusé. R1 interroge chacun de ses voisins (R2) et arme dans le même temps ce temporisateur « Route active », au temps initial de 3 minutes.

- R2 reçoit la requête de R1 et entame une phase locale de réévaluation. Au terme de cette phase, R2 n'a pas trouvé de successeur faisable, marque la route active, interroge tous ses voisins (R3) et arme un temporisateur de route active.

- R3 reçoit la requête de R2, examine sa table de topologie sans trouver de successeur faisable et n'a pas d'autre voisin à interroger. R3 supprime la route de ses propres tables et répond à la requête de R2.

- R2 reçoit la réponse de R3 et n'attend plus d'autre réponse. R2 supprime la route de ses propres tables et répond à la requête de R1.

- R1 reçoit la réponse de R2 et n'attend pas d'autre réponse. R1 supprime la route de ses tables.

Le tout s'est achevé en moins de 3 minutes, le temporisateur de route active n'a pas été sollicité. Le cas décrit par l'étiquette 2' de l'illustration est moins favorable : R2 ne reçoit jamais la réponse de R3 !

- Le temporisateur de route active sur R1 continue de décompter pendant que R2 et R3 tentent de répondre à la question qui leur a été posée.

- Si le temporisateur expire avant que toutes les requêtes n'aient reçu leurs réponses, la route est déclarée SIA. La relation de voisinage qui lie R1 à R2 est supprimée.

Reproduire ce genre de désagrément dans une topologie simulée sous GNS3 n'est guère difficile. Il suffit d'abaisser la bande passante des liens à une valeur ridiculement faible (on peut descendre à 1 Kbps) puis de limiter la bande passante octroyée à EIGRP à une part tout aussi ridicule, on peut l'abaisser de 50 %, la valeur par défaut, à 1 %. En procédant ainsi sur la topologie de l'atelier 7b (qui a servi à illustrer la section dédiée à l'agrégation), voici le résultat d'une requête émanant du routeur RTR7A, destinée à son voisin RTR7E et non satisfaite :

```
RTR7A#sh ip eigrp topology active
IP-EIGRP Topology Table for AS(64501)/ID(192.168.0.250)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

A 172.16.0.128/26, 0 successors, FD is 40514560, q
    1 replies, active 00:01:33, query-origin: Local origin, retries(1)
via 192.168.0.254 (40514560/28160), Serial0/1
via 192.168.0.249 (Infinity/Infinity), rs, q, Serial0/0, serno 64,anchored
```

Observez la lettre « A » à gauche de la capture qui identifie une route active, le compteur qui rappelle que 1 minute et 33 secondes se sont écoulées depuis le passage à l'état actif, le drapeau « r » qui indique qu'une question posée est encore sans réponse.

Une fois les 3 minutes du temporisateur de route active écoulées, le processus SYSLOG émet deux messages d'avertissement dont un de niveau 3 (« *Error condition* ») :

```
RTR7A#sh ip eigrp topology active
IP-EIGRP Topology Table for AS(64501)/ID(192.168.0.250)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

A 172.16.0.128/26, 0 successors, FD is 40514560
    1 replies, active 00:03:02, query-origin: Local origin, retries(1)
via 192.168.0.254 (40514560/28160), Serial0/1
via 192.168.0.249 (Infinity/Infinity), rs, Serial0/0, serno 64
RTR7A#
00:46:13: %DUAL-3-SIA: Route 172.16.0.128/26 stuck-in-active state in IP-EIGRP
64501.  Cleaning up
00:46:13: %DUAL-5-NBRCHANGE: IP-EIGRP 64501: Neighbor 192.168.0.249 (Serial0/0) is
down: stuck in active
```

L'administrateur confronté à ce problème peut tenter de le résoudre en augmentant le temps initial du temporisateur de route active, ce qui s'opère à l'aide de la commande **timers active-time** en configuration d'interface. La syntaxe de cette commande est la suivante :

```
Router(config-if)#timers active-time [time-limit | disabled]
```

... dont les arguments sont :

time-limit

Temps d'attente des réponses aux requêtes d'EIGRP, exprimé en minutes dans l'espace [1 - 4294967295].

disabled

Désactive le temporisateur, la conséquence est qu'une route qui passe à l'état actif peut le rester indéfiniment si toutes les réponses n'ont pas été obtenues.

Cette solution ne peut être qu'un pis-aller et va contribuer à dégrader le temps de convergence d'EIGRP. Le plus souvent, l'administrateur devra s'atteler à la tâche délicate qui consiste à isoler le lien ou le routeur qui pose problème. Il lui faudra alors procéder par ordre :

1. Quelles sont les routes annoncées régulièrement dans l'état SIA ?

2. Quel routeur croit bon régulièrement de ne pas répondre ?

3. Pourquoi ce routeur ne répond-il pas alors qu'on l'interroge poliment ?

Ses outils principaux sont alors :

- La commande **show ip eigrp topology**.
- La commande **show ip eigrp topology active** qui ne montre que les routes actives.
- La commande **debug eigrp packets request reply** afin de surveiller l'activité EIGRP avec les réserves habituelles quant à la dangerosité de toute commande **debug**.

CISCO ne cesse de poursuivre les développements autour d'EIGRP et ce problème de route marquée SIA a été pris en compte. La réflexion est la suivante (reprenez sous les yeux l'illustration précédente) : En quoi remettre en question la relation de voisinage entre R1 et R2 fait-il progresser l'efficacité d'EIGRP ? Quel rapport avec la disparition du réseau 10.1.1.0/24 ?

Cisco a donc modifié le comportement de SIA (IOS 12.1(4.0.3)T et 12.1(4.1)). Les modifications s'appuient sur la création de deux triplets TLV supplémentaires : *SIA-Query* et *SIA-Reply* ainsi que sur un temporisateur « *SIA-Query* » initialisé à 90 secondes, soit la moitié du temporisateur de route active.

- Lorsque R1 interroge R2, il arme non plus un mais deux temporisateurs : le temporisateur de route active initialisé à 180 secondes et le temporisateur « *SIA-Query* » initialisé à 90 secondes.
- Si R1 n'a pas reçu de réponse quand le temporisateur « *SIA-Query* » expire, R1 envoie une requête « *SIA-Query* ».
- Si R2 répond à cette requête par un message « *SIA-Reply* », alors R1 réinitialise ses deux temporisateurs et la relation de voisinage est maintenue.
- L'émission d'un nouveau « *SIA-Query* » est réitérée 90 secondes plus tard. En cas de réponse de R2 par un message « *SIA-Reply* », le réseau se voit octroyer un délai supplémentaire de 90 secondes.

R1 peut ainsi émettre jusqu'à trois requêtes « *SIA-Query*» et si on additionne les 90 secondes qui ont précédé l'émission du premier « *SIA-Query* », c'est finalement six minutes que R1 aura laissé au réseau pour collecter l'ensemble des réponses. Bien sûr, à l'intérieur de ces six minutes, la procédure peut prendre fin à tout moment dès que la vraie réponse attendue, qui concerne la route, est obtenue.

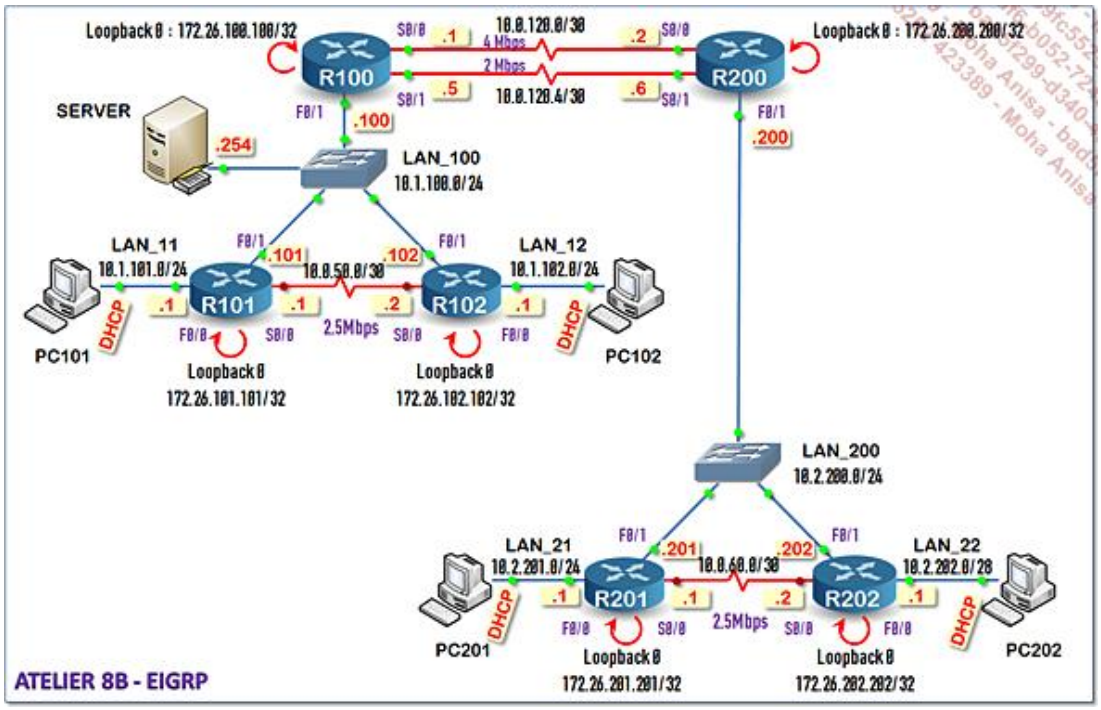
Pour l'administrateur, surveiller la fréquence d'apparition des couples « SIA-Query/SIA-Reply » donne un moyen supplémentaire de détecter et peut-être prévenir la formation de routes SIA et l'instabilité de routage qui en découle. Une commande utile pour ce faire :

```
R220#show ip eigrp trafic
IP-EIGRP Traffic Statistics for process 64501
  Hellos sent/received: 722/722
  Updates sent/received: 27/27
  Queries sent/received: 3/7
  Replies sent/received: 7/3
  Acks sent/received: 34/34
  Input queue high water mark 4, 0 drops
```

```
  SIA-Queries sent/received: 0/0
  SIA-Replies sent/received: 0/0

R220#
```

4. Atelier : Mise en œuvre d'une configuration EIGRP avancée

La topologie proposée est la même, seul le plan d'adressage est modifié :



a. Tâche 1 : Modifier le plan d'adressage

- Téléchargez depuis le site des Editions ENI les deux fichiers texte **Atelier8b_R100.txt** et **Atelier8b_R200.txt**. Selon toute méthode à votre convenance, injectez les configurations fournies sur R100 et R200.

- Sur chacun des quatre routeurs R101, R102, R201 et R202 (par exemple sur R101) configurez les nouvelles adresses sur les interfaces F0/1, S0/0 et Lo0 :

```
R101(config)#int f0/1
R101(config-if)#ip address 10.1.100.101 255.255.255.0
R101(config-if)#int s0/0
R101(config-if)#ip address 10.0.50.1 255.255.255.252
R101(config-if)#int f0/0
R101(config-if)#ip address 10.1.101.1 255.255.255.0
R101(config-if)#int loopback 0
R101(config-if)#ip address 172.26.101.101 255.255.255.255
R101(config-if)#^Z
R101#
```

- Modifiez l'association statique du nom serveur :

```
R101(config)#ip host server 10.1.100.254
R101(config)#^Z
```

b. Tâche 2 : Modifier le processus de routage EIGRP

- Sur les deux routeurs R101 et R102, modifiez la configuration EIGRP de l'AS 100. Par exemple, sur R101 :

```
R101(config)#no router eigrp 100
R101(config)#router eigrp 100
R101(config-router)#network 10.0.0.0
R101(config-router)#^Z
R101#
```

- Sur les deux routeurs R201 et R202, modifiez la configuration EIGRP de l'AS 200. Par exemple sur R201 :

```
R201(config)#no router eigrp 200
R201(config)#router eigrp 200
R201(config-router)#network 10.0.0.0
R201(config-router)#^Z
R201#
```

c. Tâche 3 : Ajouter un processus de routage RIP

- Sur chacun des quatre routeurs, créez six interfaces de loopback et attribuez-leur respectivement les adresses 172.26.xxx.111 à 172.26.xxx.116 où xxx est le numéro du routeur considéré 101, 102, 201 ou 202. Par exemple sur R101 :

```
R101(config)#int loopback 1
R101(config-if)#ip address 172.26.101.111 255.255.255.255
R101(config-if)#int loopback 2
R101(config-if)#ip address 172.26.101.112 255.255.255.255
R101(config-if)#int loopback 3
R101(config-if)#ip address 172.26.101.113 255.255.255.255
R101(config-if)#int loopback 4
R101(config-if)#ip address 172.26.101.114 255.255.255.255
R101(config-if)#int loopback 5
R101(config-if)#ip address 172.26.101.115 255.255.255.255
R101(config-if)#int loopback 6
R101(config-if)#ip address 172.26.101.116 255.255.255.255
R101(config-if)#^Z
R101#wr
```

- Sur chacun des quatre routeurs, ajoutez un processus RIP version 2 qui prenne en compte ces interfaces. Par exemple sur R101 :

```
R101(config)#router rip
R101(config-router)#version 2
R101(config-router)#network 172.26.0.0
R101(config-router)#^Z
R101#
```

d. Tâche 4 : Configurer la redistribution de routes RIP vers EIGRP

- Sur chacun des quatre routeurs, configurez la redistribution des routes apprises par RIP dans EIGRP. N'oubliez pas ce moyen mnémotechnique qui consiste à remplacer le mot-clé **redistribute** par le mot français « récupère », la compréhension de la redistribution y gagne :

```
R101(config)#router eigrp 100
R101(config-router)#redistribute rip
R101(config-router)#default-metric ?
<1-4294967295>  Bandwidth in Kbits per second

R101(config-router)#default-metric 10000 ?
<0-4294967295>  Delay metric, in 10 microsecond units

R101(config-router)#default-metric 10000 10 ?
```

```
<0-255>  Reliability metric where 255 is 100% reliable

R101(config-router)#default-metric 10000 10 255 ?
<1-255>  Effective bandwidth metric (Loading) where 255 is 100% loaded

R101(config-router)#default-metric 10000 10 255 1 ?
<1-4294967295>  Maximum Transmission Unit metric of the path

R101(config-router)#default-metric 10000 10 255 1 1500
R101(config-router)#^Z
R101#
```

- Comptabilisez les routes contenues dans chaque table de routage :

```
R101#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

172.26.0.0/32 is subnetted, 28 subnets
D EX    172.26.202.116
           [170/1159680] via 10.1.100.100, 00:01:36, FastEthernet0/1
D EX    172.26.201.116
           [170/1159680] via 10.1.100.100, 00:02:25, FastEthernet0/1
D EX    172.26.202.115
           [170/1159680] via 10.1.100.100, 00:01:37, FastEthernet0/1
D EX    172.26.201.115
           [170/1159680] via 10.1.100.100, 00:02:25, FastEthernet0/1
D EX    172.26.202.114
           [170/1159680] via 10.1.100.100, 00:01:37, FastEthernet0/1
D EX    172.26.201.114
           [170/1159680] via 10.1.100.100, 00:02:25, FastEthernet0/1
D EX    172.26.202.113
           [170/1159680] via 10.1.100.100, 00:01:36, FastEthernet0/1
D EX    172.26.201.113
           [170/1159680] via 10.1.100.100, 00:02:26, FastEthernet0/1
D EX    172.26.202.112
           [170/1159680] via 10.1.100.100, 00:01:36, FastEthernet0/1
D EX    172.26.201.112
           [170/1159680] via 10.1.100.100, 00:02:26, FastEthernet0/1
D EX    172.26.202.111
           [170/1159680] via 10.1.100.100, 00:01:37, FastEthernet0/1
D EX    172.26.201.111
           [170/1159680] via 10.1.100.100, 00:02:26, FastEthernet0/1
D EX    172.26.202.201
           [170/1159680] via 10.1.100.100, 00:01:37, FastEthernet0/1
D EX    172.26.201.201
           [170/1159680] via 10.1.100.100, 00:02:26, FastEthernet0/1
D EX    172.26.102.116
           [170/261120] via 10.1.100.102, 00:04:14, FastEthernet0/1
D EX    172.26.102.115
           [170/261120] via 10.1.100.102, 00:04:14, FastEthernet0/1
D EX    172.26.102.114
           [170/261120] via 10.1.100.102, 00:04:14, FastEthernet0/1
D EX    172.26.102.113
           [170/261120] via 10.1.100.102, 00:04:14, FastEthernet0/1
D EX    172.26.102.112
           [170/261120] via 10.1.100.102, 00:04:14, FastEthernet0/1
D EX    172.26.102.111
           [170/261120] via 10.1.100.102, 00:04:14, FastEthernet0/1
D EX    172.26.102.102
```

```
[170/261120] via 10.1.100.102, 00:04:14, FastEthernet0/1
C       172.26.101.116 is directly connected, Loopback6
C       172.26.101.115 is directly connected, Loopback5
C       172.26.101.114 is directly connected, Loopback4
C       172.26.101.113 is directly connected, Loopback3
C       172.26.101.112 is directly connected, Loopback2
C       172.26.101.111 is directly connected, Loopback1
C       172.26.101.101 is directly connected, Loopback0
     10.0.0.0/8 is variably subnetted, 9 subnets, 3 masks
D EX    10.0.60.0/30 [170/2053120] via 10.1.100.100, 00:09:34, FastEthernet0/1
C       10.0.50.0/30 is directly connected, Serial0/0
D       10.1.102.0/24 [90/30720] via 10.1.100.102, 00:25:37, FastEthernet0/1
C       10.1.101.0/24 is directly connected, FastEthernet0/0
C       10.1.100.0/24 is directly connected, FastEthernet0/1
D       10.0.120.0/29 [90/1154560] via 10.1.100.100, 00:11:52, FastEthernet0/1
D EX    10.2.202.0/24
           [170/1159680] via 10.1.100.100, 00:09:35, FastEthernet0/1
D EX    10.2.201.0/24
           [170/1159680] via 10.1.100.100, 00:09:35, FastEthernet0/1
D EX    10.2.200.0/24
           [170/1157120] via 10.1.100.100, 00:09:36, FastEthernet0/1
R101#
```

Sauf erreur, vous devriez constater une certaine inflation : 37 routes ! Les routes concernées par l'agrégation ont été mises en gras.

- Sur chacun des quatre routeurs, configurez l'agrégation afin de contenir l'inflation du nombre de routes. Par exemple sur R101 :

```
R101(config)#router eigrp 100
R101(config)#int f0/1
R101(config-if)#ip summary-address eigrp 100 172.26.101.0 255.255.255.0
R101(config-if)#^Z
R101#
```

- Vérifiez les effets de cette agrégation. Toujours sur R101 :

```
R101#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     172.26.0.0/16 is variably subnetted, 11 subnets, 2 masks
D EX    172.26.202.0/24
           [170/1159680] via 10.1.100.100, 00:03:03, FastEthernet0/1
D EX    172.26.201.0/24
           [170/1159680] via 10.1.100.100, 00:03:04, FastEthernet0/1
D       172.26.102.0/24
           [90/261120] via 10.1.100.102, 00:03:04, FastEthernet0/1
D       172.26.101.0/24 is a summary, 00:03:04, Null0
C       172.26.101.115/32 is directly connected, Loopback5
C       172.26.101.114/32 is directly connected, Loopback4
C       172.26.101.113/32 is directly connected, Loopback3
C       172.26.101.112/32 is directly connected, Loopback2
C       172.26.101.116/32 is directly connected, Loopback6
C       172.26.101.101/32 is directly connected, Loopback0
C       172.26.101.111/32 is directly connected, Loopback1
     10.0.0.0/8 is variably subnetted, 9 subnets, 3 masks
D EX    10.0.60.0/30 [170/2053120] via 10.1.100.100, 00:03:04, FastEthernet0/1
C       10.0.50.0/30 is directly connected, Serial0/0
D       10.1.102.0/24 [90/30720] via 10.1.100.102, 00:03:04, FastEthernet0/1
```

```
C       10.1.101.0/24 is directly connected, FastEthernet0/0
C       10.1.100.0/24 is directly connected, FastEthernet0/1
D       10.0.120.0/29 [90/1154560] via 10.1.100.100, 00:03:04, FastEthernet0/1
D EX    10.2.202.0/24
[170/1159680] via 10.1.100.100, 00:03:04, FastEthernet0/1
D EX    10.2.201.0/24
           [170/1159680] via 10.1.100.100, 00:03:04, FastEthernet0/1
D EX    10.2.200.0/24
           [170/1157120] via 10.1.100.100, 00:03:04, FastEthernet0/1
R101#
```

On passe de 3 x 7 = 21 à 3 x 1 + la route 172.26.101.0/24 vers l'interface **Null0**. On se souvient que par défaut, EIGRP utilise cette interface afin d'éliminer les paquets qui correspondent à la route parent mais pour lesquels aucun des réseaux enfants ne correspond à l'adresse de destination des paquets. EIGRP inclut de façon automatique une route résumée vers l'interface Null0 à chaque fois que l'une de ces deux conditions existe :

- Un résumé de route est activé, qu'il soit automatique ou manuel.
- Un ou plusieurs sous-réseaux du réseau agrégé existent qui ont été appris par EIGRP.

Imaginez par exemple que R101 reçoive un paquet destiné à 172.26.101.117. L'hôte 172.26.101.117 lui est inconnu. La seule possibilité qui reste est l'action Poubelle, c'est l'objet de la route vers **Null0**. De 37 routes, l'administrateur a réussi à ramener la table de routage à 20 routes.

Votre agrégation de routes a porté ses fruits, bravo. Cet atelier est maintenant terminé.

Résumé

1. Les caractéristiques à retenir

La relative complexité du protocole a fait préférer des synthèses réparties au long du chapitre. Merci de vous reporter notamment aux sections :

- DUAL, synthèse partielle ;
- Partage de charge, synthèse.

Il existe au moins trois ouvrages dans le commerce, tous américains, dont EIGRP est le sujet unique. Autant dire que si nous avons essayé ici de faire un tour sérieux du sujet, il est loin d'être clos. De son côté, CISCO, évidemment promoteur de son protocole, continue de le faire progresser. Un administrateur en charge d'un grand réseau EIGRP devra donc rester en veille constante.

2. Les commandes à retenir

a. Commandes de configuration

Commande	Mode	Description
ip classless	Configuration globale	Active l'algorithme de recherche de correspondance de préfixe la plus longue (*Longest Match based Forwarding Algorithm*) dans le processus de recherche de route. C'est la commande par défaut.
ip subnet-zéro	Configuration globale	Autorise les adresses de sous-réseaux exclusivement composées de 0.
router eigrp *as-id*	Configuration globale	Active le processus EIGRP et le fait participer au système autonome de numéro *as-id*.
network *@IP_réseau*	Configuration de routeur	L'adresse indiquée doit inclure les adresses des interfaces qui participent au protocole.
passive-interface *interface-typeinterface-number*	Configuration de routeur	Cesse l'envoi de mises à jour sur l'interface spécifiée.
auto-summary	Configuration de routeur	Active ou désactive l'agrégation automatique à la frontière d'un réseau majeur.
metric weights *tos k1 k2 k3 k4 k5*	Configuration de routeur	Permet un réglage fin du calcul de la métrique par EIGRP. La valeur tos doit rester à zéro.
eigrp stub [**receive-only** | **connected** | **static** | **summary** | **redestributed**]	Configuration de routeur	Configure un routeur d'extrémité (*Spoke Router*) de façon à ce qu'il ne reçoive plus les requêtes de ses voisins.
variance *multiplier*	Configuration de routeur	La valeur par défaut est 1 et entraîne un partage de charge à coût égal. La valeur acceptée doit être comprise entre 1 et 128.
timers active-time [*time-limit* | **disabled**]	Configuration de routeur	Par défaut, ce temporisateur est désactivé (cela n'a pas toujours été le cas). Dans cet état, une route peut rester à l'état actif indéfiniment.

ip hello-interval eigrp *as-id seconds*	Configuration d'interface	Configure la période qui sépare deux messages Hello. 60 s par défaut sur les interfaces lentes de type NBMA, 5 s par défaut sur les autres interfaces.
ip hold-time eigrp *as-id seconds*	Configuration d'interface	Configure le temporisateur d'attente du prochain message Hello. 180 s par défaut sur les interfaces lentes de type NBMA, 15 s par défaut sur les autres interfaces.
bandwidth *kbps*	Configuration d'interface	Affecte à une interface WAN sa vraie bande passante, exprimée en kilobits par seconde.
delay *tens-of-µsecondes*	Configuration d'interface	Affecte à une interface une estimation de son délai, exprimée en dizaines de microsecondes.
ip bandwidth-percent eigrp *as-id percent*	Configuration d'interface	Spécifie le pourcentage de bande passante qu'EIGRP est en droit d'utiliser sur une interface. La valeur par défaut est 50 %.
ip summary-address eigrp *as-number network-address subnet-mask* [*admin-distance*]	Configuration d'interface	Provoque l'annonce d'un agrégat de prefixes sur une interface.
ip authentication mode eigrp *as_id* **md5**	Configuration d'interface	Spécifie le type d'authentification des échanges EIGRP par signature MD5.
ip authentication key-chain eigrp *as_idname_of_key-chain*	Configuration d'interface	Active l'authentification des messages EIGRP émis depuis cette interface ou reçus sur cette interface.
key *number*	Configuration d'ensemble de clés	Spécifie une clé dans l'ensemble de clés.
key-string *text*	Configuration de clé	Attribue une chaîne de caractères à la clé qui servira soit de mot de passe soit de clé pour élaborer une signature MD5 de la mise à jour.
accept-lifetime *start-time* {**infinite** | *end-time* | **duration** *seconds*}	Configuration de clé	Spécifie un laps de temps pendant lequel la clé est valide en réception.
send-lifetime *start-time* {**infinite** | *end-time* | **duration** *seconds*}	Configuration de clé	Spécifie un laps de temps pendant lequel la clé peut être utilisée dans les mises à jour émises.
show ip route eigrp	Mode utilisateur	Affiche l'ensemble des routes issues d'EIGRP.
show ip protocols	Mode utilisateur	Affiche les paramètres et l'état actuel du protocole de routage activé sur le routeur.
debug ip eigrp packets	Mode privilégié	Affiche en temps réel le trafic d'acheminement EIGRP.

b. Commandes de supervision

Commande	Mode	Description
show ip protocols	Mode utilisateur	Affiche les paramètres et l'état actuel du protocole de routage activé sur le routeur.
show ip route eigrp	Mode utilisateur	Affiche l'ensemble des routes issues d'EIGRP.
show ip route *prefix*	Mode utilisateur	Affiche l'ensemble des routes menant à *prefix*.

show ip eigrp interface	Mode utilisateur	Affiche des informations sur les interfaces qui participent au protocole.
show ip eigrp neighbors	Mode utilisateur	Affiche la table de voisinage.
show ip eigrp topology	Mode utilisateur	Affiche la table de topologie.
show ip eigrp traffic	Mode utilisateur	Affiche des statistiques sur le trafic d'acheminement.
debug ip eigrp as-id | neighbor | notifications | summary	Mode privilégié	Débogage du routage IP.
debug eigrp fsm | neighbors | packets | transmit	Mode privilégié	Débogage de l'algorithme de routage.

Aperçu du protocole

1. Principes généraux

Parce qu'il fallait tourner la page des écueils de RIP, « *Open Shortest Path First* » fut développé par l'IETF (*Internet Engineering Task Force*) dans le but de devenir LE protocole IGP recommandé (par l'IETF). OSPF est un protocole à états de liens, il utilise l'algorithme de Dijkstra (ou algorithme SPF) pour construire une topologie exempte de boucles. Comme son nom l'indique, il est ouvert, autrement dit il n'appartient ni à un constructeur, ni à une organisation. Des travaux de recherche sur l'algorithme SPF avaient été entamés dès 1978 pour le réseau ARPANET, le groupe de travail de l'IETF fut créé en 1988 et aboutit à la publication du RFC1131 en 1989 :

RFC1131	OSPF Specification	John MOY	Oct. 1989
RFC1247	OSPF Version 2	John MOY	Juillet 1991
RFC1583	OSPF Version 2	John MOY	Mars 1994
RFC2178	OSPF Version 2	John MOY	Juillet 1997
RFC2328	OSPF Version 2, toujours d'actualité	John MOY	Avril 1998
RFC2740	OSPF for IPv6 Ce RFC est effectivement nommé « OSPF for IPv6 ». Cependant, un certain nombre de publications le nomme OSPF Version 3.	R. Coltun, D. Ferguson, J. Moy	Décembre 1999
RFC5340	OSPF for IPv6	R. Coltun, D. Ferguson, J. Moy, A. Lindem	Juillet 2008

Tableau 1 : Historique des RFC d'OSPF

Le RFC1583 est disponible au format PDF et présente l'avantage d'utiliser de vrais schémas qui peuvent aider à la compréhension de la complexité d'OSPF (la plupart des RFC utilisent de pseudo-schémas réalisés en mode texte car ils privilégient la légèreté et l'universalité). Le RFC2328 est disponible en français sur le site : http://abcdrfc.free.fr/ ce qui aide évidemment (merci au traducteur car la tâche doit être harassante et ingrate) même s'il faut alerter sur certains choix qui peuvent être regrettés (ex : « *net mask* » est traduit par gabarit de réseau, « *routing table* » est traduit par tableau d'acheminement).

OSPF appartient à la classe des protocoles de routage à états de liens et revendique les avantages de sa classe :

- Convergence rapide aidée en cela par des mises à jour déclenchées et de type incrémental.
- Capacité à prendre en compte de grands réseaux.
- Quelques autres caractéristiques qui font l'intérêt du protocole :
    - OSPF met à profit un concept d'aires afin de contenir ses exigences en ressources machine (mémoire et CPU), afin également de réduire autant que faire se peut le trafic d'acheminement. Pour mémoire, le trafic d'acheminement est le trafic inhérent au protocole de routage. La bande passante consommée pour ce trafic l'est au détriment de la bande passante utile.
    - Les aires d'OSPF sont construites selon une topologie hiérarchique autour de l'aire « 0 », appelée backbone, et toujours présente.
    - Le comportement du protocole est de type « *classless* ». À ce titre, il supporte VLSM ainsi que les routes résumées.
    - Métrique intelligente fondée sur la bande passante des liens.
    - Adresses multicast réservées à OSPF de façon à réduire l'impact sur les dispositifs non-OSPF.

- Les échanges OSPF peuvent être authentifiés.

- Prise en compte de routes issues d'autres protocoles à l'aide du marquage de routes (*route tagging*)...

Ces avantages indubitables d'OSPF sont évidemment obtenus au prix de quelques inconvénients :

- OSPF consomme de la mémoire. Chaque routeur doit entretenir plusieurs bases de données, dont une base de données de voisinage (*OSPF neighbors*) et une base de données des états de liens de chacun des autres routeurs appelée LSD (*Link State Database*).

- OSPF consomme de la ressource CPU, c'est spécialement vrai au démarrage du protocole lorsqu'un routeur qui ne part de rien doit construire l'ensemble des bases de données pour ensuite extraire sa table de routage.

- L'application d'OSPF à de grands réseaux est complexe car elle doit s'appuyer sur une topologie hiérarchique d'aires, une aire regroupe plusieurs routeurs. De fait, OSPF nécessite des experts bien formés.

- L'étape de configuration et de mise en œuvre est également complexe, une panne ou une erreur de conception sont plus difficiles à diagnostiquer/éradiquer qu'avec un protocole à vecteur de distance.

Ces réserves étant posées, il faut retenir deux arguments essentiels qui militent pour imposer OSPF :

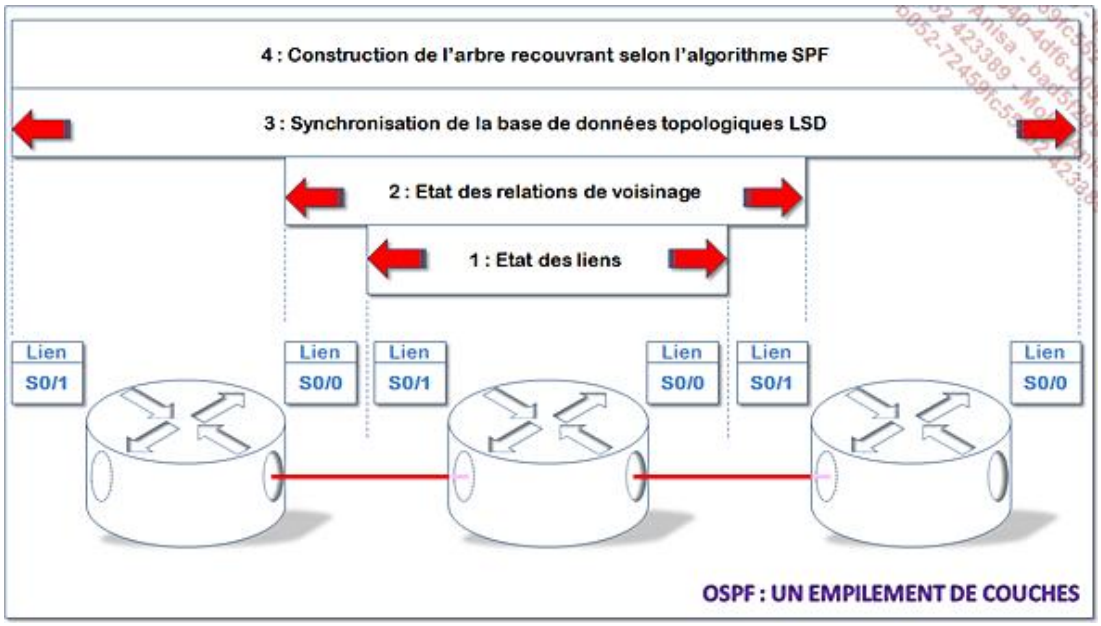
- OSPF sait prendre en compte de grands réseaux. En pareil cas (disons au-delà de cinquante routeurs), RIP est automatiquement éliminé.

- OSPF est un protocole ouvert. Si outre le fait d'être grand, le réseau est hétérogène (Comment..., avec des routeurs non CISCO... ?), alors OSPF peut même devenir le choix unique.

2. Terminologie

Le problème avec OSPF est qu'on ne sait pas par où commencer. Il est quasi impossible de bâtir un exposé parfaitement linéaire et séquencé, c'est-à-dire un exposé dans lequel le lecteur n'aurait pas à faire des allers et retours. L'auteur réclame donc l'indulgence du lecteur face aux probables nombreuses référence-avant. Une référence-avant clairement identifiée l'est par la mention « (patientez) ». Par ailleurs, le formateur doit souvent décider où placer le curseur entre le souci d'exhaustivité et le besoin d'un message clair quand il faut médiatiser la complexité. En final, la seule exhaustivité est celle du RFC par définition.

Au niveau le plus global, tentons une première description du fonctionnement d'OSPF :



OSPF peut être vu comme un empilement de couches :

- En couche 1, OSPF doit apprendre l'état de ses propres liens. La portée de la couche 1 est limitée au routeur.

- En couche 2, un routeur OSPF construit des relations avec ses voisins OSPF. La portée de la couche 2 est limitée aux interfaces des voisins physiques, ces interfaces sont directement connectées.

    - Un routeur parlant OSPF envoie des messages HELLO sur toutes ses interfaces participant au protocole.

    - Deux routeurs qui partagent un lien et conviennent de certains paramètres à l'aide de leurs messages Hello respectifs deviennent des voisins (*neighbors*).

    - Une relation d'adjacence ou de proximité, qui peut se concevoir comme l'état ultime de la relation de voisinage, peut s'établir entre voisins. Cette relation est conditionnée par le type de routeurs échangeant les messages Hello ainsi que par le type de réseau transportant ces messages.

- La couche 3 consiste à synchroniser la base de données topologiques (*Link State Database* : LSD) et s'appuie sur les relations de proximité établies par la couche 2.

    - Chaque routeur envoie ses LSAs (*Link State Advertisements*) à tous ses routeurs adjacents. Un LSA de routeur décrit l'état de tous les liens de ce routeur.

    - Parce qu'il existe une grande variété de liens, OSPF définit également de nombreux types de LSAs.

    - Chaque routeur qui reçoit un LSA de l'un de ses proches enregistre ce LSA dans sa LSD (*Link State Database*) et en envoie une copie à la totalité de ses autres proches.

    - Les LSAs étant inondés sur une aire donnée, permettent à l'ensemble des routeurs de construire une LSD identique.

- Enfin, la couche 4 pourrait être la construction de l'arbre de recouvrement en se fondant sur les données topologiques recueillies dans la LSD par la couche 3.

    - Une fois la LSD complète, chaque routeur déroule l'algorithme SPF pour construire un arbre décrivant l'ensemble des chemins à meilleur coût vers toutes les destinations connues. Chaque routeur est à la racine de l'arbre qu'il calcule.

    - Chaque routeur déduit sa table de routage en extrayant les routes de l'arbre SPF qu'il a calculé.

Une fois les LSAs inondés dans une aire donnée, une fois les LSDs synchronisées (identiques), une fois les tables de routage déduites, OSPF est un protocole que l'on peut qualifier de calme et silencieux. Les échanges se limitent aux courts messages Hello, utiles afin de maintenir les liens en vie (*Keep alive*), les échanges de LSAs n'intervenant que toutes les 30 minutes. Pas d'autre activité notable à observer en dehors de celle qu'entraînerait un changement de topologie.

N'ajoutons pas à la complexité un doute sur le sens que doivent prendre les mots. Les termes nouveaux propres au protocole OSPF (ou propres aux protocoles à états de liens) sont définis ci-dessous :

Link

Puisqu'il est question d'un protocole à états de liens, ce terme doit revêtir une importance particulière. C'est pourtant très simple puisque c'est ainsi qu'OSPF désigne une interface de routeur. Quand une interface est ajoutée au processus OSPF, elle est considérée comme un lien. À ce lien sont associés diverses informations qu'il est possible d'afficher en partie à l'aide d'une commande **show ip ospf interface**.

Router-ID ou RID

Il est très important qu'OSPF puisse identifier sans ambiguïté chaque routeur qui participe au protocole. L'IETF a choisi de le faire à l'aide d'un identifiant au format identique à celui d'une adresse IP. Ainsi, il est facile d'extrapoler un RID par défaut de la configuration du routeur. L'administrateur a également la possibilité de le définir directement comme illustré à la section Configuration OSPF - Configuration de base - Identifiant du routeur.

Un point clé du protocole réside dans la méthode d'attribution de cet identifiant. Le routeur a recours à l'une des méthodes suivantes :

1. Le routeur choisit l'adresse IP la plus élevée (l'adresse est alors considérée comme un nombre) parmi les adresses attribuées à ses interfaces de loopback.

2. Pour le cas, peu probable, où l'administrateur n'aurait configuré aucune interface de loopback, le routeur lui substitue l'adresse IP la plus élevée parmi les adresses affectées à ses interfaces physiques. L'interface physique qui a fourni l'adresse IP n'a pas nécessairement besoin d'être l'une des interfaces participant au protocole.

Configurer une interface de loopback offre à l'administrateur le moyen de maîtriser l'identifiant OSPF attribué au routeur OSPF. En effet, l'interface de loopback étant virtuelle, elle ne souffre pas des mêmes affres que les interfaces physiques. L'interface est « up » tant que le routeur est actif et ne tombe « down » que si le routeur s'interrompt. Autre avantage non négligeable, l'adresse IP attribuée à l'interface de loopback n'est pas contrainte par le plan d'adressage du réseau, l'administrateur peut donc utiliser une adresse qu'il identifiera facilement.

Pour être complet, si l'identifiant RID provient d'une interface physique, le fait que l'interface vienne ultérieurement à tomber ne remettrait pas en cause l'identifiant attribué. Dans ce cas précis, la stabilité de l'interface de loopback n'est donc qu'un avantage secondaire. L'avantage déterminant réside dans le fait de pouvoir affecter l'identifiant plutôt que de le voir déduit d'une configuration d'interface physique.

« *Originating routeur* »

Chaque routeur qui participe à un réseau OSPF vit sa propre existence. Les raisonnements qu'il faut bâtir doivent l'être en se plaçant du point de vue d'un routeur. Le routeur choisi pour jouer ce rôle est souvent désigné dans ce chapitre par « ce routeur ».

Le protocole « *Hello* »

Afin d'apprendre qui sont ses voisins puis entretenir les relations avec eux, un routeur OSPF génère des messages Hello sur toutes ses interfaces, de façon régulière. La période qui sépare deux messages Hello est « *HelloInterval* », ce délai est configurable par interface à l'aide de la commande **ip ospf hello-interval**. L'implémentation OSPF de CISCO utilise une valeur par défaut de 10 secondes.

Une fois ajouté à la table de voisinage, un voisin fait l'objet d'une surveillance régulière. Si les messages Hello que le routeur s'attend à recevoir ne lui parviennent plus pendant un temps d'observation suffisant, le voisin est considéré comme perdu (il n'est donc plus voisin). C'est le délai « *RouterDeadInterval* » qui règle la durée de ce coma pré mortem. Cisco s'en tient à la valeur suggérée par le RFC, soit 4 x *HelloInterval*, c'est-à-dire 40 secondes par défaut. À nouveau, cette valeur est modifiable à l'aide de la commande **ip ospf dead-interval**.

Base de données de voisinage (« Neighbor ship database »)

Liste de tous les routeurs OSPF pour lesquels un message Hello a été reçu récemment (depuis moins de « *RouterDeadInterval* » secondes). À chaque routeur sont associés un certain nombre de détails tels le RID, la priorité (valeur utilisée lors de l'élection du DR (*Designated Router*) et du BDR (*Backup Designated Router*) sur un réseau multi-accès), l'état de la relation de voisinage, l'interface depuis laquelle le message a été entendu, l'adresse IP de l'interface qui a émis le message... L'administrateur peut observer le contenu de cette table à l'aide de la commande **show ip ospf neighbor**. Il est possible d'obtenir un niveau de détail plus élevé en entrant la commande **show ip ospf neighbor *#RID***.

Relation de voisinage

Hélas, le RFC utilise « *neighbor* » pour désigner indifféremment le voisin et la relation de voisinage entretenue avec lui. OSPF entretient un diagramme d'état pour chaque relation de voisinage établie avec chaque voisin découvert. La relation passe par différents états, les premiers établissent une relation de voisinage, les suivants établissent une relation de proximité. En final, un voisin physique peut devenir voisin ou proche.

Proches (Adjacence)

Faut-il traduire le terme « *Adjacency* » du RFC par relation d'adjacence ou relation de proximité ? Les choix diffèrent selon les publications. Dans les deux cas, la relation d'adjacence est l'état ultime d'une relation de voisinage. Deux voisins ne deviennent pas nécessairement adjacents mais deux routeurs adjacents ont d'abord été deux voisins. À l'état adjacent, les bases de données de liens LSD sont synchronisées (identiques). Dans la suite de ce chapitre, un voisin désigne un routeur OSPF voisin, un proche désigne un routeur OSPF adjacent.

Type de réseau

OSPF distingue trois types de média :

- les réseaux point à point ;

- les réseaux à diffusion (*Broadcast Network*) ;
- les réseaux qui ne connaissent pas la diffusion (*Non-Broadcast*).

Quand le réseau est sans diffusion, OSPF peut adopter deux types de comportement :

- le mode NBMA (*Non-Broadcast Multi Access*) ;
- le mode point à multipoint.

Enfin, il faut citer également la possibilité de créer un lien virtuel :

- Les liens virtuels. Sans eux, la hiérarchie créée à l'aide des aires est limitée à deux niveaux. Le lien virtuel permet de connecter une aire à l'aire backbone de façon logique sans que l'aire soit physiquement raccordée à l'aire backbone (voir lien virtuel).

Réseau point à point

La liaison « serial » qui relie une paire de routeurs. Deux voisins connectés par un réseau point à point deviennent nécessairement adjacents. Les paquets OSPF destinés à l'autre auraient pu l'être à l'aide de son adresse IP unicast. Mais dans ce cas, le routeur aurait dû découvrir au préalable cette adresse. C'est donc l'adresse multicast 224.0.0.5 (qui signifie tous les routeurs OSPF, « *AllSPFRouters* ») qui fait office d'adresse de destination. Cette règle souffre une exception dans le cas de LSA retransmis, qui sont toujours émis vers l'adresse unicast du demandeur. C'est bien normal, seul le demandeur est intéressé par la retransmission.

Réseau à diffusion

Tous les réseaux qui supportent la diffusion (*Broadcast*), à savoir Ethernet, Token Ring, FDDI. Ces réseaux sont multi-accès puisqu'ils connectent ensemble plus de deux machines. Mais en outre toutes les machines reliées par ce réseau peuvent être destinataires d'un seul paquet lorsqu'il est émis vers l'adresse de diffusion.

Comme expliqué lors de l'exposé sur l'arbre SPF, afin de réduire le nombre de relations de proximité à entretenir et donc le trafic d'acheminement, le processus OSPF doit élire parmi les routeurs connectés via ce réseau deux routeurs appelés DR (*Designated Router*) et BDR (*Backup Designated Router*) qui se verront confier des missions particulières.

Sur un réseau à diffusion, les voisins sont découverts de façon dynamique à l'aide du protocole « *OSPF Hello* ». Quelle que soit leur origine, les messages Hello sont diffusés vers l'adresse de multicast 224.0.0.5 (qui signifie tous les routeurs OSPF, « *AllSPFRouters* »). C'est également le cas des messages OSPF quand ils sont émis par l'un des deux routeurs DR ou BDR. En revanche, les messages « *LSA update* » et « *LSA Acknowledgment* » d'OSPF quand ils sont issus des autres routeurs présents sur le réseau à diffusion (ni DR, ni BDR, → « DRother ») et destinés aux routeurs DR et BDR, sont émis vers l'adresse multicast 224.0.0.6 (qui signifie les deux routeurs DR et BDR, « *AllDRouters* »).

Un rappel sur les adresses de multidiffusion et la façon de former les adresses de couche 2 correspondantes a été placé en Annexe.

Mode NBMA

Éligibles pour fonctionner dans ce mode, les réseaux X25, Frame Relay et ATM soit un ensemble de réseaux WAN. Ces réseaux sont capables de relier plus de deux routeurs sans pour autant disposer de la capacité de diffusion.

C'est encore le protocole OSPF Hello qui est utilisé afin de maintenir les relations de voisinage mais puisqu'il faut se passer de la possibilité de diffuser un message, la découverte des voisins ne peut se faire qu'au prix d'une configuration supplémentaire.

Ces réseaux peuvent fonctionner vis-à-vis d'OSPF comme le font les réseaux à diffusion, c'est-à-dire en procédant à l'élection d'un routeur désigné ainsi qu'à l'élection d'un routeur désigné de secours. Une différence cependant au niveau des adresses de destination : le multicast n'étant plus supporté, l'ensemble des paquets OSPF est émis vers des adresses unicast.

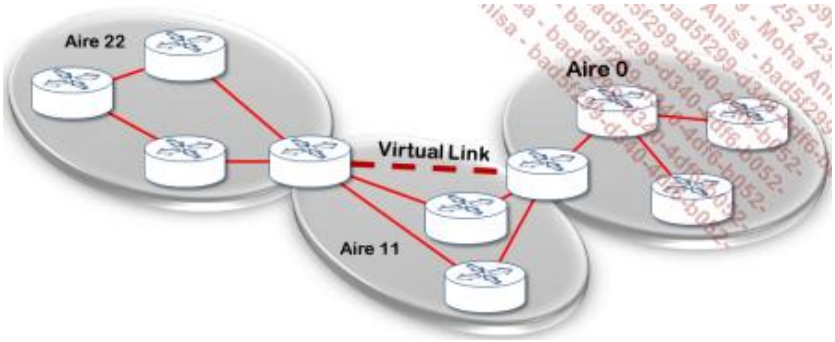
Attention, le mot-clé qui désigne ce mode dans l'interface ILC est **non-broadcast**.

Mode point à multipoint

Il s'agit d'un réseau sans diffusion ayant fait l'objet d'une configuration particulière dont il résulte un comportement identique à celui d'une collection de réseaux point à point. Les paquets OSPF peuvent être émis vers l'adresse multicast 224.0.0.5 (qui signifie tous les routeurs OSPF, « *AllSPFRouters* »).

Lien virtuel (« Virtual Link »)

Un lien vers l'aire « backbone » qui transite par une aire quelconque différente de 0. Exemple :



Réseau de transit

Au-delà de cette classification qui consiste à ranger les réseaux selon les cinq types reconnus par OSPF, il faut en outre distinguer un réseau selon qu'il est de transit ou d'extrémité. On reconnaît un réseau de transit au fait qu'aucune des adresses source et destination des paquets observés sur ce réseau n'appartient à ce réseau. Les paquets ne font que « passer » sur un réseau de transit.

Réseau d'extrémité (« Stub network »)

Un réseau d'extrémité n'a qu'un seul routeur attaché. Chaque paquet observé sur un réseau d'extrémité a, soit l'adresse source, soit l'adresse destination qui appartient à ce réseau. OSPF annonce une route vers un hôte comme étant une route vers un réseau d'extrémité. De même, les interfaces de loopback sont considérées comme des réseaux d'extrémité et annoncées comme tels.

Réseau de rattachement

Afin d'éviter les fréquentes répétitions, précisons que nous nommerons ainsi le réseau auquel l'interface dont il est question est connectée.

LSA (*Link State Advertisement*)

À la différence des protocoles à vecteur de distance qui diffusent des informations de routage (en fait les routes contenues dans la table de routage), un routeur OSPF diffuse des informations de topologie. Il le fait sous forme de LSAs traduisibles par annonces d'états de liens ou avis d'états de liens. Attention, chaque mot est pesé et chaque mot compte... Chaque routeur dans le système autonome génère un ou plusieurs LSA. Chaque routeur transmet ses LSA à tous ses proches. Chaque routeur qui reçoit un LSA de l'un de ses proches le transmet aux autres proches dans un processus d'inondation. L'ensemble des LSA générés et collectés par un routeur constitue la base de données d'états de liens, c'est-à-dire la LSD (*Link State Database*).

Le RFC 2328 connaît cinq types de LSA, chacun ayant une fonction propre. Les deux types les plus importants sont certainement les LSA de routeur (Type 1 : « *Router-LSA* ») et les LSA de réseau (Type 2 : « *Network-LSA* ») qui décrivent comment les routeurs et les réseaux d'une zone sont interconnectés. Les LSA de résumé (Type 3 et 4 : « *Summary-LSA* ») annoncent les destinations extérieures à l'aire ou les routeurs placés à la frontière du système autonome. Enfin les LSA externes au système autonome (Type 5 : « *AS-external-LSA* ») annoncent les destinations extérieures au système autonome.

3. Algorithme Dijkstra du plus court chemin

Voilà l'occasion de faire une rencontre, virtuelle bien sûr, avec un personnage des plus intéressants. Monsieur Edsger Wybe Dijkstra (1930 - 2002) était physicien, mathématicien et informaticien néerlandais. Dès 1955, il fut l'un des pionniers éclairés de l'informatique et on lui doit de nombreuses contributions dans le domaine des systèmes et celui de la programmation. C'est par exemple lui qui a formalisé le concept de sémaphore et s'en est servi pour résoudre quelques problèmes devenus classiques de l'informatique mais mis en scène de façon très ludique : le problème des lecteurs et des rédacteurs (Accès aux bases de données) et le dîner des philosophes (Partage de ressources et ordonnancement des processus en informatique système). Monsieur Dijkstra a reçu le prix Turing en 1972 (prix attribué chaque année pour une contribution majeure à la communauté informatique) et a prononcé à cette occasion un discours resté célèbre « le programmeur modeste » qu'il est facile de retrouver sur le net. Monsieur Dijkstra était paraît-il un caractère difficile et appréciait les aphorismes (sentences), alors l'auteur ne résiste pas au plaisir de vous livrer deux d'entre eux :

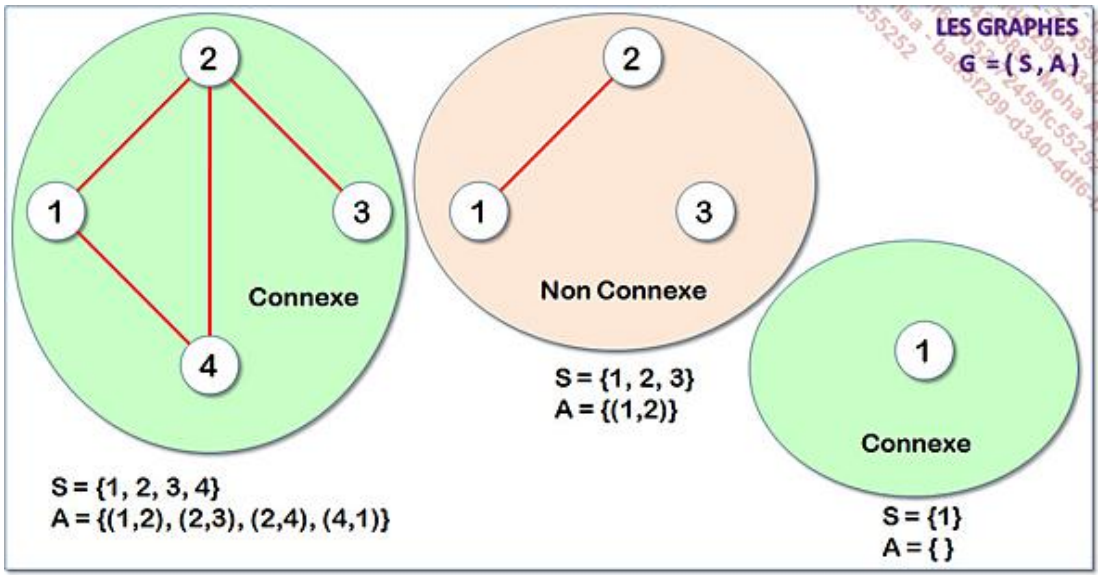
- « Tester un programme peut démontrer la présence de bugs, jamais leur absence » ;
- « Se demander si un ordinateur peut penser est aussi intéressant que de se demander si un sous-marin peut nager ».

Monsieur Dijkstra n'était pas un adepte des outils numériques et publiait ses travaux sous forme de lettres manuscrites toutes référencées EWD, la dernière fut EWD 1318.

L'algorithme du plus court chemin, devenu algorithme de Dijkstra, fut publié en 1959. L'algorithme répond à cette question :

- Quel est le plus court chemin entre deux sommets d'un graphe connexe dont le poids lié aux arêtes est positif ou nul ?

En théorie des graphes, un graphe est noté $G = (S, A)$ où :

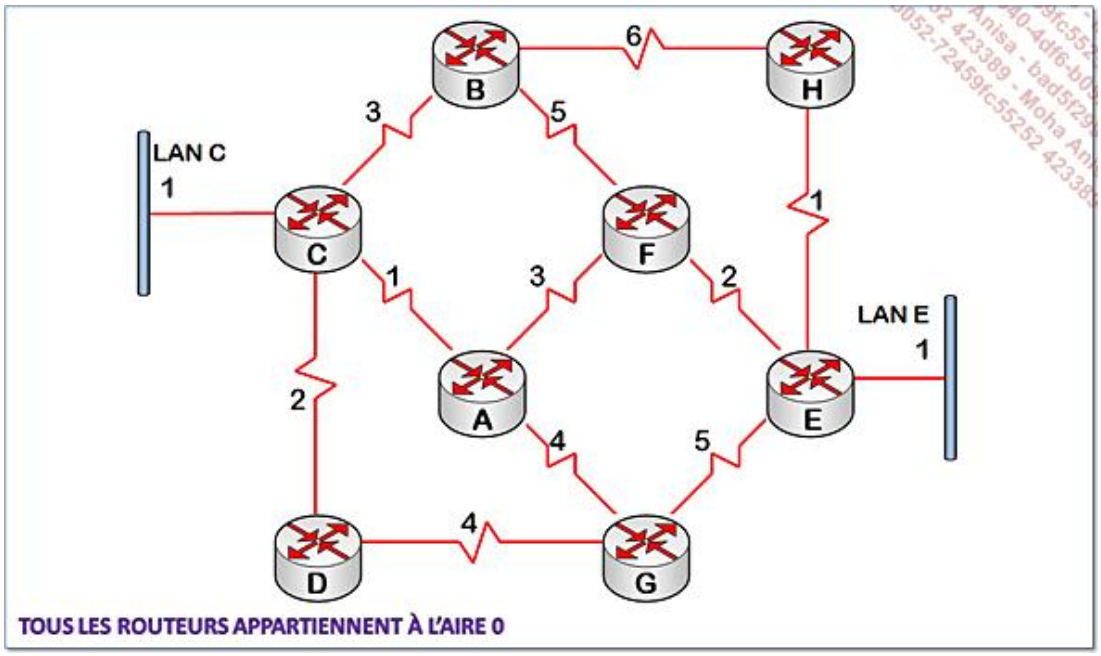


- ***S*** est l'ensemble des sommets du graphe *G*.
- ***A*** est l'ensemble des arêtes de *G* tel que : si (s_1,s_2) est dans *A*, alors il existe une arête depuis le nœud s_1 vers le nœud s_2.
- *Poids*(*s*1,*s*2) est défini sur *A* et renvoie un nombre positif caractérisant le coût de l'arête reliant s_1 à s_2 (un poids infini caractérise une paire de sommets qui ne sont pas connectés par une arête).
- Un graphe est connexe si quels que soient les sommets *u* et *v* de S, il existe un chemin du sommet *u* au sommet *v*, c'est-à-dire une suite d'arêtes permettant d'atteindre le sommet *v* en partant du sommet *u*.

L'efficacité et la relative simplicité de l'algorithme de Dijkstra n'intéressent pas que l'informatique. Imaginez un réseau routier, chaque sommet est une ville et chaque arête est une route dont le poids est la longueur de la route exprimée en kilomètres. L'algorithme de Dijkstra est utilisé par les sites qui proposent des itinéraires routiers, les notions de trajets plus rapides, plus courts, plus économiques... se traduisant par un ajustement du poids des arcs.

OSPF n'est pas le seul protocole de routage à avoir adopté l'algorithme de Dijkstra. IS-IS (*Intermediate System to Intermediate System*), protocole IGP défini par l'ISO (norme internationale ISO/IEC 10589:2002) a fait de même. L'IETF a publié les spécifications d'IS-IS dans la RFC 1142.

Mais revenons au cas qui nous préoccupe, celui d'OSPF et raisonnons sur un exemple simple mais concret. Considérez le réseau ci-après :



Ce réseau est assimilable à un graphe connexe, l'algorithme de Dijkstra peut s'appliquer.

Modifions un peu la terminologie des graphes, appelons nœud un sommet de graphe et arc une arête du graphe. Le défi consiste à trouver pour un nœud du graphe appelé nœud racine, le chemin le plus court vers tous les autres nœuds accessibles. Le chemin le plus court est celui dont le poids cumulé de tous les arcs vers le nœud de destination est minimal, appelons distance à la racine ce poids cumulé.

L'algorithme utilise deux tables de nœuds, à chaque nœud est associée la distance au nœud racine. La première table est appelée table SP (*Shortest Path*) et contiendra les nœuds pour lesquels le chemin le plus court a déjà été trouvé. La seconde table est appelée table CSP (*Candidate Shortest Path*) et contient les nœuds pour lesquels le chemin le plus court reste à trouver. Au démarrage, l'algorithme range le nœud racine dans la table SP associée à une distance de 0, la table CSP est vide. L'algorithme procède par itérations successives. À chaque itération, l'algorithme effectue les actions suivantes :

1. Calculer la distance du nœud source vers tous ses voisins. À l'itération N=1, le nœud source est le nœud racine. Aux itérations suivantes, le nœud source est le nœud découlant de l'action 4 de la présente itération.

2. Ranger les nœuds voisins du nœud source dans la table CSP associés à leur distance minimale.

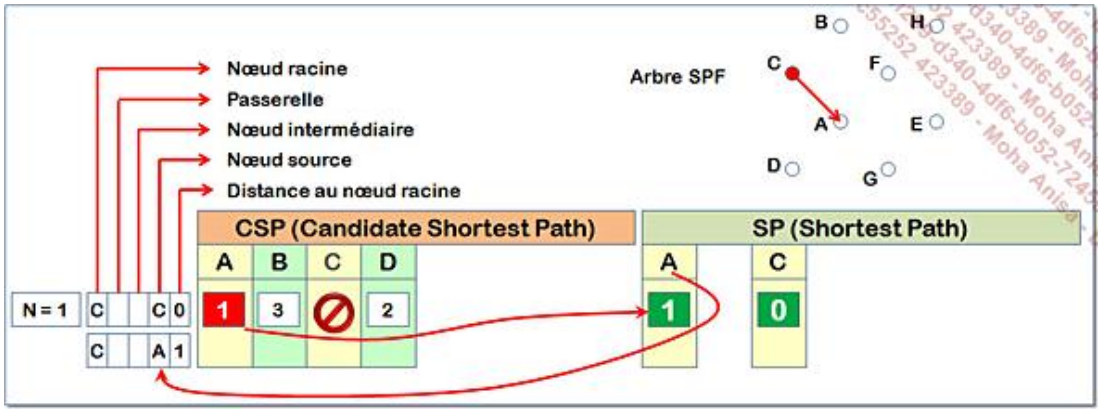
3. Choisir le nœud à distance minimale parmi tous les nœuds présents à cet instant dans la table CSP et le ranger en table SP :

a. Ce faisant, le nœud en question disparaît de la table CSP et avec lui tous les chemins alternatifs qui avaient été trouvés vers ce nœud.

b. Puisqu'il est à présent en SP, il n'y aura plus d'autre tentative de cheminement vers ce nœud.

4. Le nœud qui vient d'être rangé en SP devient également le nœud source pour l'itération à venir.

Plaçons-nous sur le routeur C du contexte précédent et testons l'algorithme (chaque routeur déroule le même algorithme mais évidemment, chaque routeur est le nœud racine de l'algorithme qu'il déroule) :



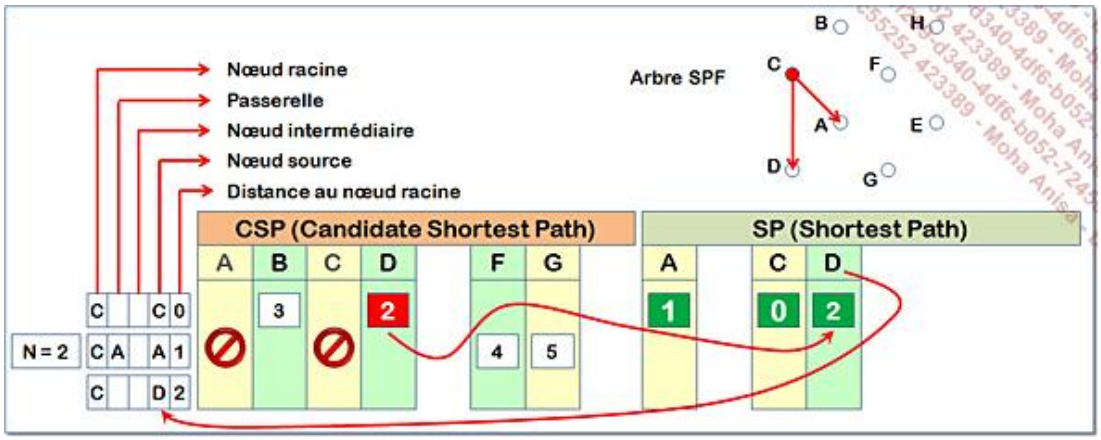
1. Nous sommes à l'itération N=1. Les colonnes suivantes renseignent pour l'itération en cours : Qui est le nœud racine ? Qui est le nœud source ? Quelle est la distance au nœud racine ? Dans le cas présent, le nœud source est

également le nœud racine soit le routeur C, la distance en cours est 0. Les voisins de C sont les routeurs A, B et D avec distances respectives de 1, 3 et 2.

2. Les trois voisins de C sont placés en CSP associés à leurs distances respectives 1, 3 et 2.

3. Le nœud à distance minimale est A, il est extrait de la CSP et placé en SP.

4. A devient le nœud source pour l'itération suivante avec une distance à la racine de 1.

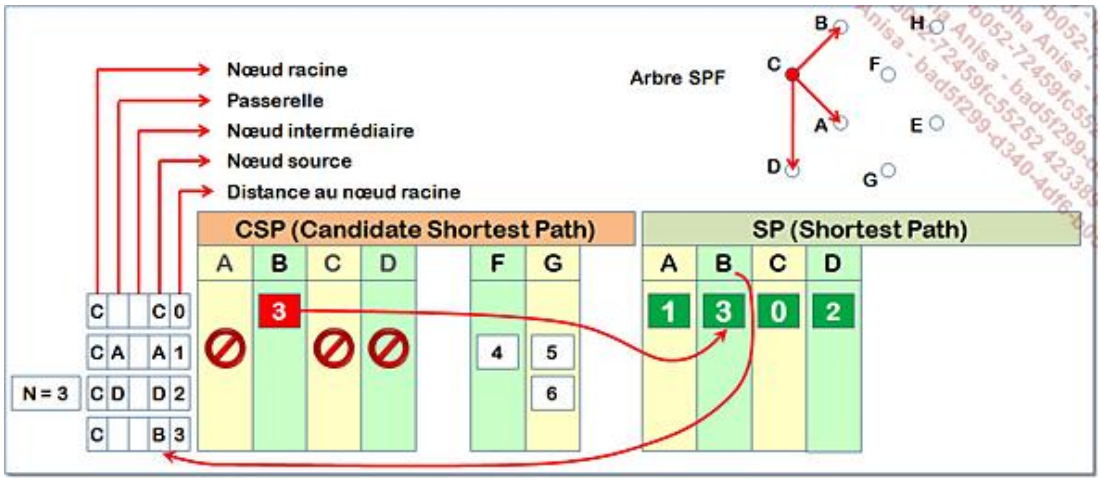


1. Les voisins de A sont les nœuds F et G. La distance vers F est égale à la distance du nœud source soit 1, additionné au coût du lien entre A et F soit 3, ce qui donne une distance cumulée de 4. La distance vers G est égale à la distance du nœud source soit 1, additionné au coût du lien entre A et G soit 4, ce qui donne une distance cumulée de 5.

2. Les deux voisins de A sont placés en CSP associés à leurs distances 4 et 5. Observez également l'apparition de A dans la colonne Passerelle. En effet, vu du routeur A, une seule chose importe : à qui remettre les paquets ? La réponse doit être choisie parmi les routeurs A, B et D tous directement connectés à C. Puisque le chemin choisi pour rejoindre les routeurs F et G passe par A, la passerelle (le prochain saut) de C pour ces destinations est le routeur A.

3. Parmi tous les nœuds présents à cet instant dans la CSP, le nœud à distance minimale est D, il est extrait de la CSP et placé en SP.

4. Le nœud D devient le nœud source pour l'itération suivante.

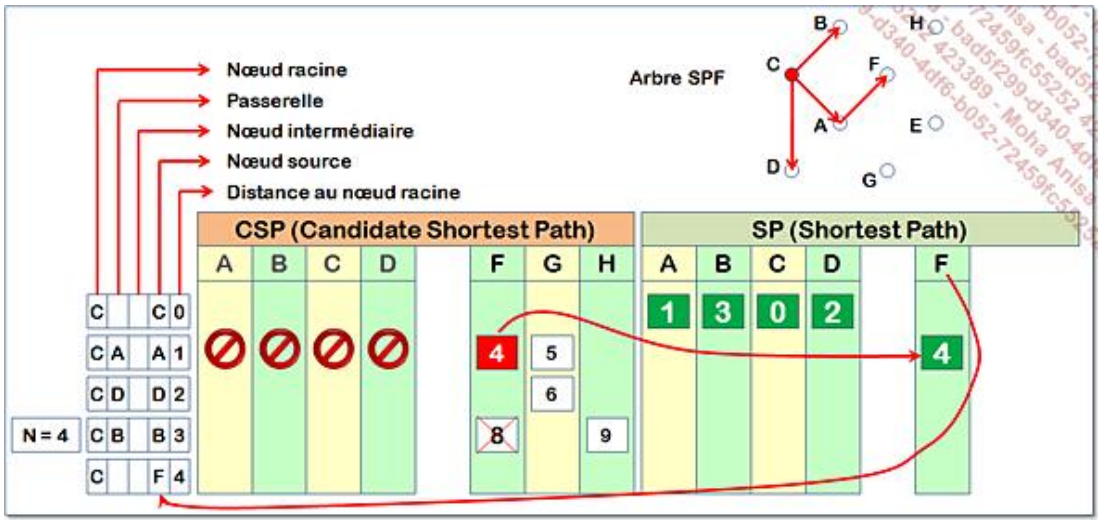


1. Le seul voisin de D est le nœud G. Par D, sa distance cumulée à la racine est égale à 6.

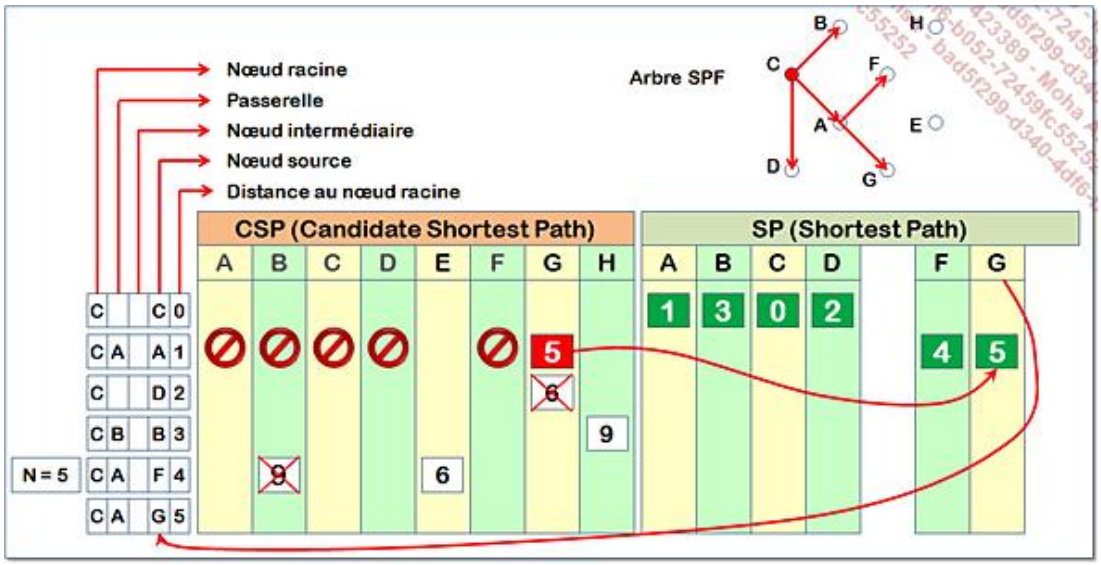
2. Une alternative vers G par A existe déjà dans la table à un meilleur coût.

3. Parmi tous les nœuds présents à cet instant dans la table, le nœud à distance minimale est B, il est extrait de la CSP et placé en SP.

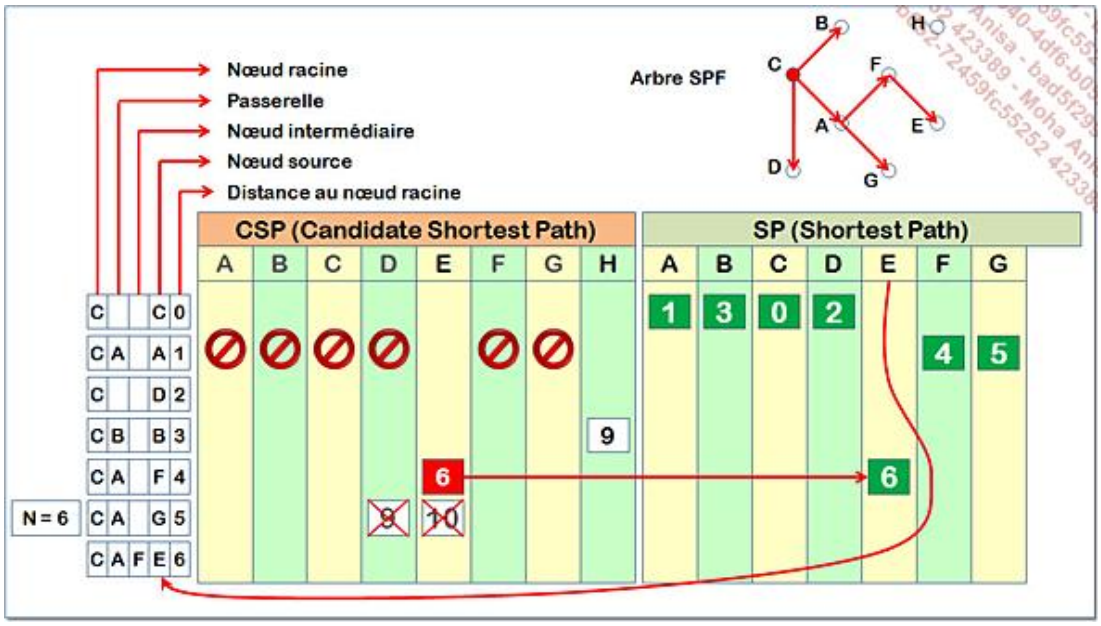
4. Le nœud B devient le nœud source pour l'itération suivante.



1. Les voisins de B sont F et H aux coûts de 8 et 9.
2. Une alternative vers F par A existe déjà dans la CSP à un meilleur coût. H est placé dans le CSP.
3. Le meilleur candidat à cet instant est F, il est extrait de la CSP et placé en SP.
4. Le nœud F devient le nœud source pour l'itération suivante.



1. Les voisins de F sont B et E aux coûts de 9 et 6.
2. Une alternative vers B par A est ignorée car B est déjà présent en SP. E est placé dans le CSP.
3. Le meilleur candidat à cet instant est G, il est extrait de la CSP et placé en SP.
4. Le nœud G devient le nœud source pour l'itération suivante.

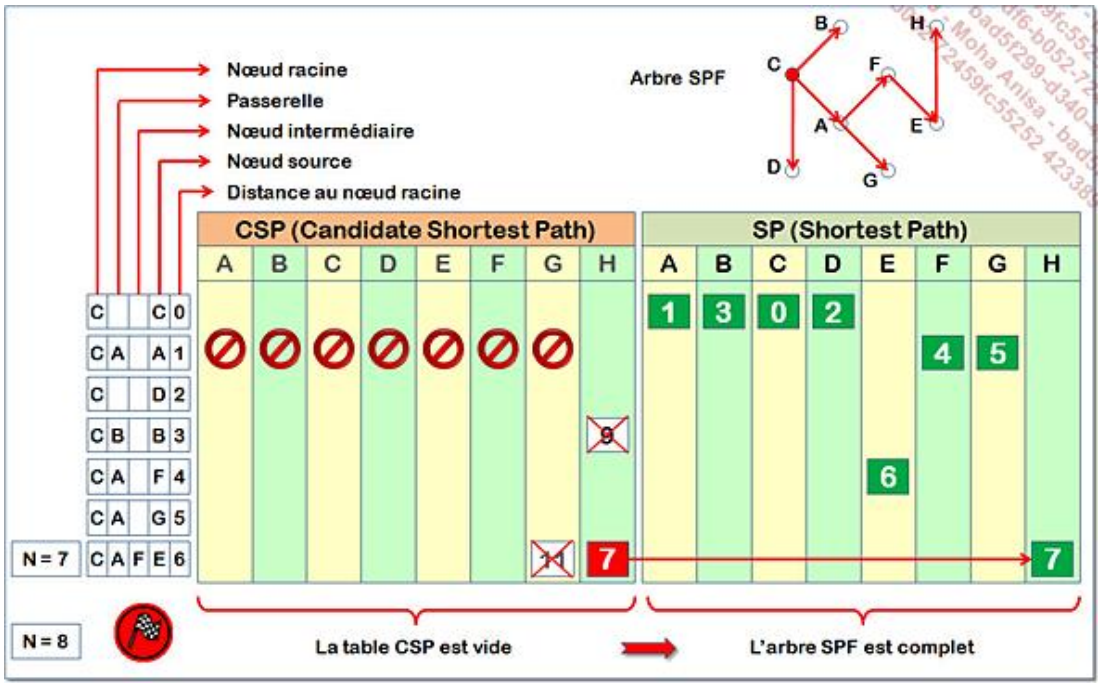


1. Les voisins de G sont D et E aux coûts de 9 et 10.

2. Une alternative vers D par A est ignorée car D est déjà présent en SP. E déjà présent en CSP à un meilleur coût.

3. Le meilleur candidat à cet instant est E, il est extrait de la CSP et placé en SP.

4. Le nœud E devient le nœud source pour l'itération suivante.



1. Les voisins de E sont G et H aux coûts de 11 et 7.

2. Une alternative vers G par A est ignorée car G est déjà présent en SP. H était déjà présent en CSP mais associé à une distance moins favorable.

3. Le dernier candidat à cet instant est H, il est extrait de la CSP et placé en SP.

4. La table CSP est désormais vide, l'itération cesse, l'arbre SPF est complet.

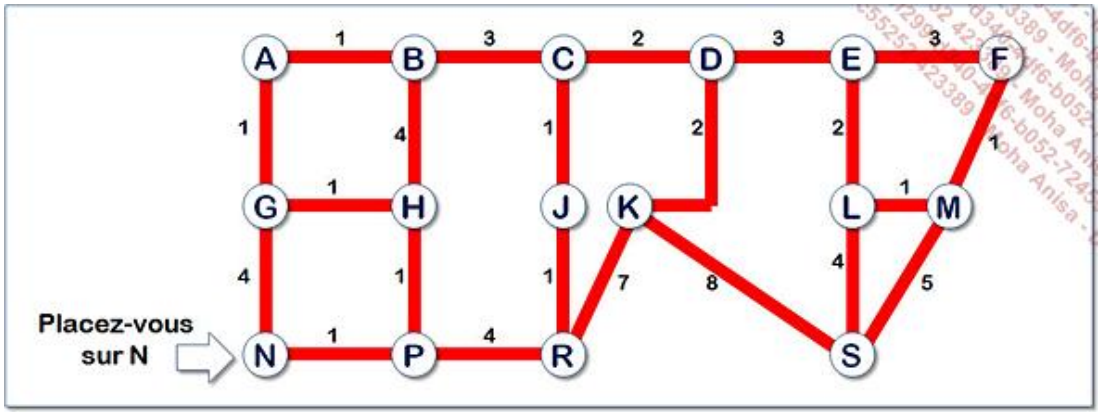
Observez l'évolution de l'arbre SPF résultant au fur et à mesure des itérations. Quand l'algorithme a terminé, la table CSP est vide et l'arbre SPF recouvre toutes les destinations, c'est pourquoi on parle souvent d'arbre couvrant.

Imaginons que chacun des routeurs soit connecté à un LAN de coût 1 à l'identique de LAN C et LAN E seuls représentés sur la figure. De la table SP qu'il vient de construire, le processus OSPF du routeur C peut maintenant déduire les routes et les placer dans la table de routage :

Pour joindre le réseau...	Passer par...	Au coût de...
LAN A	A	2
LAN B	B	4
LAN C	Directement connecté	1
LAN D	D	3
LAN E	A	7
LAN F	A	5
LAN G	A	6
LAN H	A	8

Amusez-vous :

Construisez l'arbre SPF du réseau ci-dessous :



Solution au chapitre Ateliers et exercices corrigés.

4. L'interface OSPF

a. Structure des données de l'interface

Cela a été dit, OSPF peut être vu comme un empilement de couches avec en couche 1, l'état des liens du routeur. Les autres couches ont besoins d'une absolue fiabilité de la couche 1 d'OSPF vis-à-vis des liens du routeur. Attention à la terminologie, quand OSPF parle de lien (*Link*), l'administrateur en charge d'un réseau de routeur peut traduire le terme par interface.

Pour remplir les fonctions qui sont les siennes, OSPF crée et associe une structure de données à chaque interface. On peut l'observer en partie à l'aide de la commande **show ip ospf interface** :

```
R1100b#sh ip ospf int f0/1
FastEthernet0/1 is up, line protocol is up
Internet Address 10.0.8.11/24, Area 0
Process ID 1, Router ID 1.0.0.11, Network Type BROADCAST, Cost: 1
  Transmit Delay is 1 sec, State DROTHER, Priority 1
  Designated Router (ID) 1.0.0.22, Interface address 10.0.8.22
  Backup Designated router (ID) 1.0.0.21, Interface address 10.0.8.21
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
    Hello due in 00:00:01
  Neighbor Count is 3, Adjacent neighbor count is 2
    Adjacent with neighbor 1.0.0.22  (Designated Router)
```

```
    Adjacent with neighbor 1.0.0.21  (Backup Designated Router)
  Suppress hello for 0 neighbor(s)
R1100b#
```

Dans un autre contexte, sur une interface « *serial* » connectée à un réseau NBMA :

```
R1100c#sh ip ospf int s0/0
Serial0/0 is up, line protocol is up
Internet Address 10.0.8.11/24, Area 0
Process ID 1, Router ID 1.0.0.11, Network Type NON_BROADCAST, Cost: 64
  Transmit Delay is 1 sec, State DROTHER, Priority 1
  Designated Router (ID) 1.0.0.22, Interface address 10.0.8.22
  Backup Designated router (ID) 1.0.0.21, Interface address 10.0.8.21
  Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit 5
    Hello due in 00:00:25
  Neighbor Count is 3, Adjacent neighbor count is 2
    Adjacent with neighbor 1.0.0.22  (Designated Router)
    Adjacent with neighbor 1.0.0.21  (Backup Designated Router)
  Suppress hello for 0 neighbor(s)
```

Les informations contenues dans la structure sont les suivantes :

- L'adresse IP affectée à l'interface et son masque réseau, 10.0.8.11/24 dans le cas présent. Tous les paquets OSPF issus de cette interface utiliseront cette adresse en tant qu'adresse IP source.

- L'identifiant d'aire (*Area ID*) : cette interface ainsi que le réseau auquel cette interface est connectée. L'en-tête OSPF de tous les paquets OSPF issus de cette interface porte cet identifiant. Dans la capture ci-dessus, l'identifiant est 0, c'est-à-dire l'identifiant de l'aire « *backbone* ». Un domaine OSPF peut comporter plusieurs aires mais l'aire 0 doit toujours être présente.

- L'identifiant de processus OSPF (*Process ID*) : cet identifiant n'appartient pas au protocole OSPF mais à l'IOS CISCO qui l'utilise pour distinguer les différentes instances OSPF quand l'administrateur en a configuré plusieurs. L'identifiant de processus n'a pas de signification en dehors du routeur sur lequel il est configuré. Il est établi à 1 dans la capture ci-dessus.

- L'identifiant de routeur (*Router ID* que nous avons choisi d'abréger en RID) : 1.0.0.11. Une interface de loopback a été configurée sur ce routeur, c'est donc l'adresse affectée à cette interface qu'OSPF utilise pour identifier ce routeur. Au même titre que l'identifiant d'aire, l'en-tête OSPF de tous les paquets OSPF issus de cette interface porte l'identifiant de routeur.

- Le type de réseau (*Network Type*), BROADCAST dans le premier cas présenté : l'interface peut être connectée à l'un des cinq types { BROADCAST | NON_BROASCAST | point-to-point | point-to-multipoint | *Virtual Link* }.

- Le coût de l'interface (*Cost*), 1 dans le premier cas présenté, 64 dans le second : la métrique OSPF est fondée sur cette notion de coût appliqué aux paquets sortants de l'interface. Par défaut, le coût de l'interface est le résultat du calcul 10^8/*Bandwidth*. À l'évidence, le numérateur a été choisi à une époque où le débit 100 Mbps de la technologie Fast Ethernet ou FDDI représentait le débit maximum que l'on envisageait sur une interface. Le résultat du calcul est un entier non-signé exprimé sur 16 bits de 1 à 65535. Quand le débit et donc la bande passante est 100 Mbps, le coût de l'interface est 1. Ce coût reste 1 quelle que soit l'interface dont le débit serait supérieur à celui de Fast Ethernet. C'est évidemment un inconvénient qu'il est possible de corriger.

    - Depuis la version 11.2 de l'IOS, il est possible de modifier la bande passante de référence à l'aide de la commande **auto-cost reference-bandwidth** en configuration de routeur :

```
R1100b#sh ip ospf int F0/0
FastEthernet0/0 is up, line protocol is up
Internet Address 10.0.11.1/24, Area 0
  Process ID 1, Router ID 1.0.0.11, Network Type BROADCAST, Cost: 1
.........
R1100b(config)#router ospf 1
R1100b(config-router)#auto-cost reference-bandwidth ?
<1-4294967>  The reference bandwidth in terms of Mbits per second
R1100b(config-router)#auto-cost reference-bandwidth 1000
% OSPF: Reference bandwidth is changed.
        Please ensure reference bandwidth is consistent across all routers.
```

```
R1100b(config-router)#^Z
R1100b#sh ip ospf int F0/0
FastEthernet0/0 is up, line protocol is up
Internet Address 10.0.11.1/24, Area 0
  Process ID 1, Router ID 1.0.0.11, Network Type BROADCAST, Cost: 10
.........
```

- Observez le nouveau coût de l'interface F0/0 après multiplication par 10 de la bande passante de référence. Comme le rappelle l'avertissement affiché immédiatement après l'entrée de la commande, OSPF ne peut fournir des résultats cohérents que si l'ensemble des routeurs qui participent au protocole utilisent la même bande passante de référence.

- Le délai de transmission « *InfTransDelay* », noté « *Transmit Delay* » dans la capture : estimation du temps consommé par la transmission d'un paquet « *LS Update* » de mise à jour des états de lien sur cette interface. Les LSA contenus dans le paquet voient leur âge incrémenté de cette valeur lors du passage par l'interface. La valeur doit être supérieure à 0 et devrait prendre en compte les délais de transmission et de propagation.

- L'état de l'interface (*State*), DROTHER dans les deux cas présentés : état actuel de l'interface parmi les états gérés par l'automate d'états d'interface détaillé dans la section suivante.

- La priorité de routeur (*Router Priority*) : valeur utilisée par le mécanisme d'élection du routeur désigné, ainsi que du routeur désigné de secours. Ce paramètre ne concerne pas les réseaux point à point ou point à multipoint. La priorité étant exprimée sur 8 bits, l'administrateur peut attribuer toute valeur de 0 à 255. La valeur 0 entraîne la non-éligibilité de l'interface. La valeur 1 est la valeur par défaut. La valeur configurée sur l'interface est annoncée dans le contenu du paquet Hello. Dans l'exemple ci-dessous, l'administrateur souhaite que le routeur R1100b emporte l'élection et devienne routeur désigné. Pour ce faire, il laisse la priorité par défaut sur les autres interfaces concernées et ne modifie que la priorité de l'interface sur le routeur R1100b :

```
R1100b(config)#int F0/0
R1100b(config-if)#ip ospf priority ?
<0-255>  Priority

R1100b(config-if)#ip ospf priority 10
R1100b(config-if)#^Z
R1100b#sh ip ospf int f0/0
FastEthernet0/0 is up, line protocol is up
Internet Address 10.0.11.1/24, Area 0
  Process ID 1, Router ID 1.0.0.11, Network Type BROADCAST, Cost: 10
.........
```

- Le routeur désigné (*Designated Router*) : DR du réseau auquel l'interface de ce routeur est connectée. Est affiché l'identifiant RID du DR mais également l'adresse IP de l'interface du DR connectée à ce réseau, respectivement 1.0.0.22 et 10.0.8.22 dans les deux cas présentés.

- Le routeur désigné de secours (*Backup Designated Router*) : BDR du réseau auquel l'interface de ce routeur est connectée. Le BDR est identifié de la même façon que le DR. RID 1.0.0.21, interface 10.0.8.21 dans les deux cas présentés.

- La période d'émission des messages Hello (*HelloInterval*) : exprimée en secondes, espace de temps qui sépare deux émissions de messages Hello. La valeur configurée sur l'interface est annoncée dans le contenu du paquet Hello. Deux routeurs OSPF qui sont voisins physiques mais qui ne sont pas réglés avec la même valeur « *HelloInterval* » ne peuvent devenir voisins au sens OSPF. Il paraît donc logique de configurer la même valeur sur l'ensemble des routeurs connectés à un réseau. L'administrateur peut modifier la valeur à l'aide de la commande **ip ospf hello-interval** en configuration d'interface :

```
R1100b(config)#int f0/0
R1100b(config-if)#ip ospf hello-interval ?
<1-65535>  Seconds
R1100b(config-if)#ip ospf hello-interval 15
R1100b(config-if)# ^Z
R1100b#sh ip ospf int f0/0
FastEthernet0/0 is up, line protocol is up
.........
  Timer intervals configured, Hello 15, Dead 60, Wait 60, Retransmit 5
.........
```

```
R1100b(config)#int f0/0
R1100b(config-if)#no ip ospf hello-interval 15
R1100b(config-if)#^Z
R1100b#sh ip ospf int f0/0
FastEthernet0/0 is up, line protocol is up
.........
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
.........
```

- Observez que les valeurs « RouterDeadInterval » (notée Dead dans la capture) et « WaitTimer » (notée Wait dans la capture) sont par défaut prises égales à 4 fois la valeur attribuée à « HelloInterval ».

- En final, la valeur attribuée à « HelloInterval » ne peut être qu'un compromis. En diminuant la valeur, l'administrateur augmente la réactivité d'OSPF aux changements de topologie mais dans le même temps, augmente le trafic d'acheminement. La valeur par défaut dépend du type de réseau : 10 secondes dans le cas d'un réseau à diffusion (premier cas présenté), 30 secondes dans le cas d'un réseau sans diffusion (second cas présenté).

- Le délai « *RouterDeadInterval* » : en l'absence de messages Hello reçus pendant ce temps d'observation, le voisin est considéré comme perdu (il n'est plus voisin). La valeur configurée sur l'interface est annoncée dans le contenu du paquet Hello. Deux routeurs OSPF qui sont voisins physiques mais qui ne sont pas réglés avec la même valeur « *RouterDeadInterval* » ne peuvent devenir voisins au sens OSPF. À nouveau, il faut configurer la même valeur sur l'ensemble des routeurs connectés à un réseau. L'administrateur peut modifier la valeur à l'aide de la commande **ip ospf dead-interval** en configuration d'interface :

```
R1100c(config)#int s0/0
R1100c(config-if)#ip ospf dead-interval ?
<1-65535>  Seconds

R1100c(config-if)#ip ospf dead-interval 53
R1100c(config-if)#^Z
R1100c#sh ip ospf int s0/0
.........
  Timer intervals configured, Hello 10, Dead 53, Wait 53, Retransmit 5
.........
R1100c(config)#int s0/0
R1100c(config-if)#no ip ospf dead-interval 53
R1100c(config-if)#^Z
R1100c#sh ip ospf int s0/0
Serial0/0 is up, line protocol is up
.........
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 5
.........
```

- Le temporisateur d'attente « *WaitTimer* » : délai pendant lequel le routeur écoute dans l'attente de messages Hello qui lui permettraient de déterminer qui sont les routeurs DR et BDR. Ce délai est sollicité une seule fois après l'activation de l'interface. Observez sur la capture ci-dessus que ce délai est ajusté par l'IOS à une valeur identique à celle de « *RouterDeadInterval* ».

- Le délai « *RxmtInterval* », noté « *Retransmit* » dans les captures précédentes : exprimé en secondes, c'est le laps de temps qui sépare l'émission d'un paquet OSPF et sa retransmission en l'absence d'acquittement. L'IOS attribue la valeur 5 secondes par défaut mais permet à l'administrateur de modifier cette valeur à l'aide de la commande **ip ospf retransmit-interval** en configuration d'interface :

```
R1100c(config)#int S0/0
R1100c(config-if)#ip ospf retransmit-interval ?
<1-65535>  Seconds

R1100c(config-if)#ip ospf retransmit-interval 7
R1100c(config-if)#^Z
R1100c#sh ip ospf int s0/0
Serial0/0 is up, line protocol is up
  Internet Address 10.0.8.11/24, Area 0
  Process ID 1, Router ID 1.0.0.11, Network Type BROADCAST, Cost: 64
  Transmit Delay is 1 sec, State DROTHER, Priority 1
```

```
  Designated Router (ID) 1.0.0.22, Interface address 10.0.8.22
  Backup Designated router (ID) 1.0.0.21, Interface address 10.0.8.21
  Flush timer for old DR LSA due in 00:01:09
  Timer intervals configured, Hello 10, Dead 40, Wait 40, Retransmit 7
    Hello due in 00:00:02
  Neighbor Count is 3, Adjacent neighbor count is 2
    Adjacent with neighbor 1.0.0.21  (Backup Designated Router)
    Adjacent with neighbor 1.0.0.22  (Designated Router)
  Suppress hello for 0 neighbor(s)
Message digest authentication enabled
Youngest key id is 5
```

- « *Hello due in...* » : état actuel du temporisateur de messages Hello. Ce temporisateur est armé à la valeur « *HelloInterval* » puis décompte. L'arrivée à échéance provoque l'émission d'un nouveau message Hello ainsi que le réarmement du temporisateur. Dans la capture précédente, la prochaine émission aura lieu dans deux secondes.

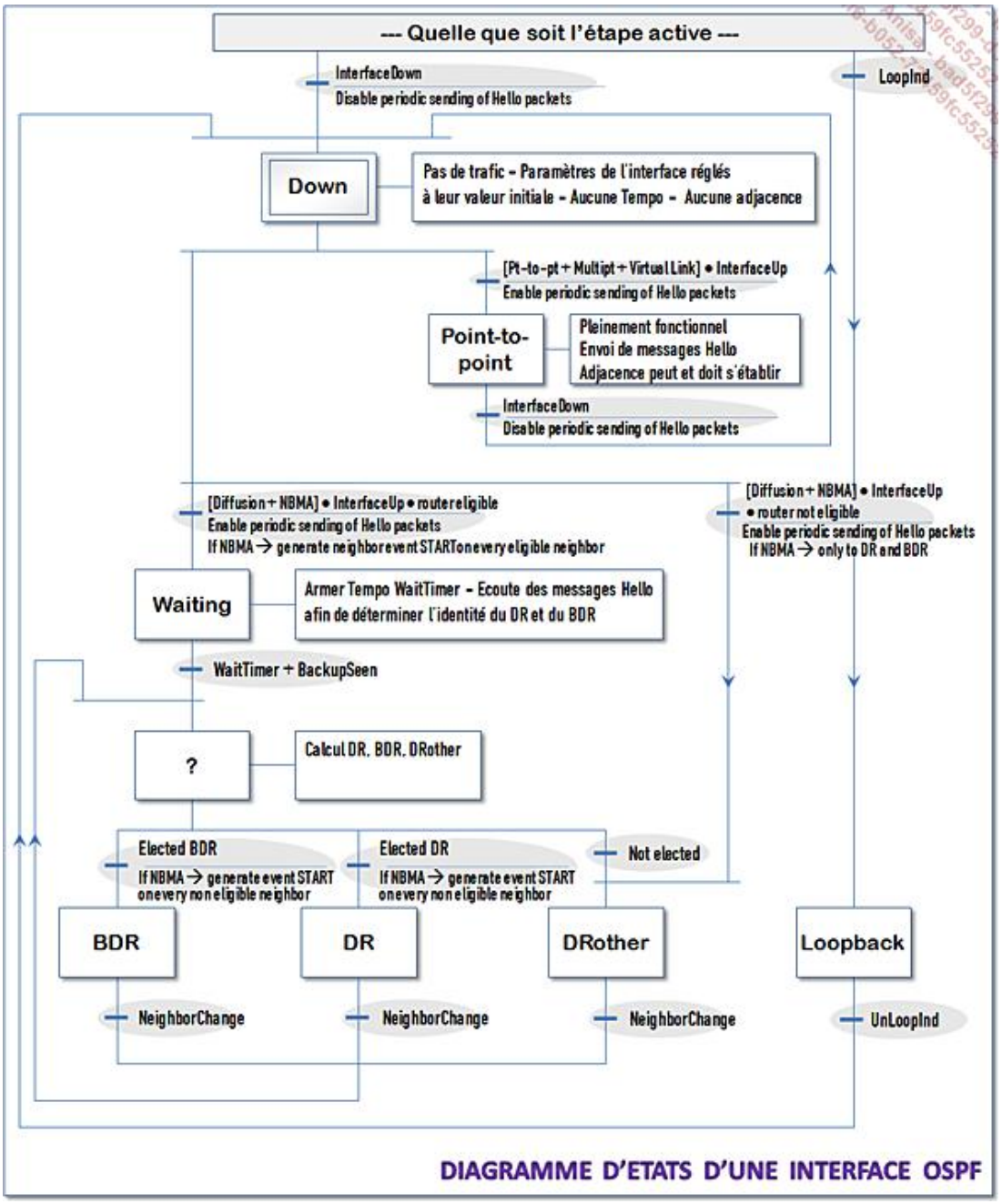
- La liste des routeurs voisins (*List of neighboringrouters*) : liste de l'ensemble des voisins sur le réseau de rattachement pour lesquels un message Hello a été reçu au cours du dernier laps de temps « *RouterDeadInterval* ». Dans la capture immédiatement précédente, cette liste comporte trois routeurs mais ce routeur (R1100c) n'a établi une relation de proximité qu'avec deux d'entre eux, le DR et le BDR. En effet, quand le réseau justifie l'élection d'un DR et d'un BDR, chacun des autres routeurs n'établit une relation de proximité qu'avec les deux routeurs désignés. Dans le cas présenté, l'information est cohérente avec le statut actuel du routeur sur ce réseau (*State DROTHER*).

- Le type d'authentification « *AuType* » : précise l'authentification en cours sur le réseau de rattachement. Les choix possibles sont {pas d'authentification | mot de passe simple | mot de passe crypté}. L'objet n'est pas d'empêcher la lecture des paquets OSPF qui transitent sur le réseau mais bien d'authentifier les échanges entre routeurs. Observez la capture immédiatement précédente, l'administrateur a configuré une authentification par mot de passe crypté. Au moment où ces lignes sont écrites, c'est MD5 qui est utilisé mais le RFC laisse la porte ouverte aux futures et probables évolutions dans le domaine mouvant de la cryptographie.

- La clé d'authentification (*Authentication Key*) : ce peut être le mot de passe simple exprimé dans ce cas sur 64 bits ou la clé la plus récente, comme le suggère la capture ci-dessus, quand le choix s'est porté sur un mot de passe crypté. Ainsi, il est possible de créer une seconde clé avant de détruire la première, ce qui permet un changement de clé en douceur sans perdre ni l'authentification des messages, ni la connectivité OSPF.

b. Automate d'états d'interface

Le GRAFCET suivant tente de rendre moins austères les diagrammes d'états du RFC 2328. À ce sujet, l'auteur a placé en ligne un document écrit dans une vie antérieure qui explique les rudiments de cet outil magique (grafcet.pdf disponible sur le site eni). L'outil appliqué de façon stricte prévoit d'associer à chaque transition une condition logique appelée réceptivité. L'étape (N-1) quand elle est active, valide la transition de l'étape (N-1) vers l'étape N. Quand la réceptivité associée à cette transition (une condition logique qui fournit un résultat vrai ou faux) devient vraie, la transition est franchie, l'étape (N-1) est désactivée, l'étape N devient active, la transition de l'étape N vers l'étape (N+1) est validée. Pour les besoins de la cause, nous nous sommes permis une application peu orthodoxe de l'outil en associant à la transition, outre une réceptivité, une action. Ceci explique les deux éléments associés à certaines transitions et séparés par un trait horizontal : l'élément du haut est la réceptivité, c'est-à-dire la condition logique qui doit devenir vraie pour franchir la transition, l'élément placé au-dessous est l'action provoquée par le franchissement de la transition.



Seule l'interface OSPF connectée à un réseau de type point à point, point à multipoint et « *virtuallink* » permet de rejoindre directement un état pleinement fonctionnel, appelé « Point-to-Point » lors de l'activation de l'interface. Pour les interfaces connectées à un réseau à diffusion ou un réseau sans diffusion en mode NBMA, aboutir à l'un des trois états pleinement fonctionnels DR, BDR ou DRother nécessite le passage par un état intermédiaire d'attente « *Waiting* ».

Détaillons les états et évènements de ce diagramme d'états :

- L'état « *Down* » : état initial de l'interface. Les protocoles de couche inférieure indiquent que l'interface n'est pas utilisable. On ne peut ni recevoir ni envoyer du trafic OSPF sur une interface à l'état « *Down* ». Les paramètres de l'interface sont ajustés à leurs valeurs initiales, les temporisateurs associés sont désactivés, l'interface ne peut bâtir de relation d'adjacence.

- L'évènement « *InterfaceUp* » : les protocoles de couche inférieure (ou le résultat d'un calcul dans le cas du réseau « *Virtual Link* ») indiquent que l'interface est désormais opérationnelle. L'interface peut sortir de l'état « Down » :

    - Si le réseau de rattachement est de type point à point, point à multipoint ou « Virtual Link », l'état passe à « Point-to-Point ».

- Dans le cas contraire et si le routeur n'est pas éligible (l'administrateur a réglé la valeur « Router Priority » à 0), l'état passe à « DRother ».

- Si le routeur est éligible, l'état passe à « Waiting ».

- Dans tous les cas, le franchissement de la transition qui permet de quitter l'état « Down » provoque l'activation de l'envoi périodique de messages Hello avec les restrictions suivantes qui concernent les réseaux sans diffusion en mode NBMA :

    - Si le routeur est éligible à devenir DR ou BDR, il envoie les messages Hello à tous les autres routeurs éligibles (préalable nécessaire au mécanisme de l'élection). En mode NBMA, c'est l'administrateur qui crée la liste des voisins. L'envoi périodique de messages Hello vers ces voisins est obtenu en générant l'évènement « ***Start*** » pour chacun des voisins de la liste. L'événement « ***Start*** » intervient dans le diagramme d'états associé à chaque relation de voisinage (patientez).

    - Si le routeur n'est pas éligible, il n'envoie des messages Hello qu'aux seuls DR et BDR, s'ils existent. À nouveau, ceci est obtenu en générant l'évènement « ***Start*** » dans les diagrammes d'états de la relation de voisinage correspondants.

- L'état « *Point-to-Point* » : l'interface tente d'établir une relation de proximité avec le voisin.

- L'évènement « *InterfaceDown* » : les protocoles de couche inférieure indiquent que l'interface n'est plus opérationnelle. Quel que soit l'état actuel de l'interface, le diagramme d'états retourne à l'état « *Down* ».

- L'état « *Waiting* » : dans cet état, le routeur écoute les messages Hello afin de tenter de déterminer qui sont les DR et BDR. Le routeur n'est pas autorisé à choisir ni DR, ni BDR. Les valeurs DR et BDR sont initialisées à 0.0.0.0.

- L'évènement « *WaitTimer* » : le temporisateur « *WaitTimer* » est arrivé à échéance, le routeur a assez attendu, il est temps de désigner quels seront DR et BDR, le diagramme d'états quitte l'état « *Waiting* » et entreprend un calcul (il s'agit plus de dérouler un algorithme, détaillé ci-après) afin de déterminer qui devraient être DR et BDR.

- L'évènement « *BackupSeen* » : cet évènement survient quand le routeur a obtenu la certitude qu'il existe un BDR sur le réseau de rattachement ou au contraire a obtenu la certitude qu'il n'existait pas de BDR sur ce réseau. Comment est-il parvenu à cette conclusion ? Dans le premier cas, il a reçu un message Hello dont l'émetteur affirme qu'il est le BDR. Dans le second cas, il a reçu un message Hello dont l'émetteur affirme qu'il est DR et qu'il n'existe pas de BDR. Dans les deux cas, une communication bidirectionnelle (*2-Way*) existait au préalable avec ce voisin (le voisin physique est également un voisin au sens OSPF). Inutile donc de rester plus longtemps dans l'état « *Waiting* ».

- L'état « ? » : ce n'est pas un état prévu par le RFC mais un arrangement de l'auteur pour tenter de modéliser la complexité du RFC dans le GRAFCET. Dans ce pseudo-état, le processus OSPF déroule un algorithme, détaillé ci-après, afin de déterminer qui sont DR et DBR et d'en déduire si ce routeur (cette interface) doit passer à l'état DR, BDR ou DRother.

- Les pseudo-évènements « *Elected DR* », « *Elected BDR* » et « *Not elected* » : générés par le pseudo-état « ? ». En mode NBMA, jusque-là, chacun des routeurs éligibles n'envoyait de messages Hello qu'aux seuls autres routeurs éligibles. Le franchissement des transitions « *Elected DR* » et « *Elected BDR* » s'accompagne de l'activation de l'envoi périodique de messages Hello vers les voisins qui ne sont pas éligibles (priorité nulle). Autrement dit, DR et BDR sur un réseau en mode NBMA envoient des messages Hello à tous leurs voisins éligibles ou non. À nouveau, ceci est obtenu en générant l'évènement « ***Start*** » dans le diagramme d'état de chacune des relations de voisinage correspondantes.

- L'état « DR » : ce routeur est le routeur désigné sur le réseau de rattachement. Il doit assumer les responsabilités correspondantes, à savoir établir une relation de proximité avec chacun des autres routeurs présents et générer un LSA de type réseau pour représenter le nœud réseau (patientez). Ce LSA contient la liste de tous les routeurs connectés au réseau de rattachement, y compris le DR lui-même. Exemple de LSA de réseau :

```
R1100c#sh ip ospf database network 10.0.8.22

OSPF Router with ID (1.0.0.11) (Process ID 1)
```

```
               Net Link States (Area 0)

 Routing Bit Set on this LSA
 LS age: 111
 Options: (No TOS-capability, DC)
 LS Type: Network Links
 Link State ID: 10.0.8.22 (address of Designated Router)
 Advertising Router: 1.0.0.22
 LS Seq Number: 80000001
 Checksum: 0x6E3E
 Length: 40
 Network Mask: /24
       Attached Router: 1.0.0.22
       Attached Router: 1.0.0.21
       Attached Router: 1.0.0.12
       Attached Router: 1.0.0.11
```

- L'état « BDR » : ce routeur est le routeur désigné de secours sur le réseau de rattachement. Il est appelé à devenir DR en cas de défaillance du DR actuel. Comme le DR, le BDR forme des relations de proximité avec l'ensemble des autres routeurs présents. Comme le DR, le BDR profite des « *LS Update* » qu'il reçoit pour maintenir à jour sa LSD. À la différence du DR, le BDR n'inonde pas c'est-à-dire ne fait pas progresser les « *LS Update* » reçus sur le reste du domaine OSPF et ne génère pas de « *LS Update* ».

- L'état « DRother » : état d'un routeur qui n'a été choisi ni comme DR ni comme BDR ou d'un routeur qui n'est pas éligible. Les seules relations de proximité entretenues le sont avec le DR et le BDR s'ils existent.

- L'évènement « *NeighborChange* » : un changement affecte l'un des voisins qui justifie la remise en question des choix opérés à l'état « ? ». La liste suivante recense les changements qui provoquent l'évènement :

    - Une nouvelle communication bidirectionnelle (*2-Way*, patientez) a été établie avec un voisin physique. L'état de la relation de voisinage est passé à « *2-Way* » ou supérieur.

    - Au contraire, une relation bidirectionnelle jusqu'ici est repassée à l'état « *Init* » ou inférieur.

    - Un routeur se déclare et c'est nouveau, DR ou BDR.

    - Un routeur et c'est nouveau, ne se déclare plus DR ou BDR.

    - La priorité de routeur annoncée par un voisin a changé.

- L'état de bouclage « *Loopback* » et les évènements associés « *LoopInd* » et « *UnLoopInd* » : quel que soit l'état actif du diagramme d'états, l'évènement « *LoopInd* », généré par les protocoles de couche inférieure ou par la gestion de réseau, fait passer dans l'état « *Loopback* ». Seul l'évènement « *UnLoopInd* », généré de la même façon que l'évènement « *LoopInd* », peut faire sortir de cet état pour aller dans l'état « *Down* ».

c. Représentation des réseaux LAN et NBMA

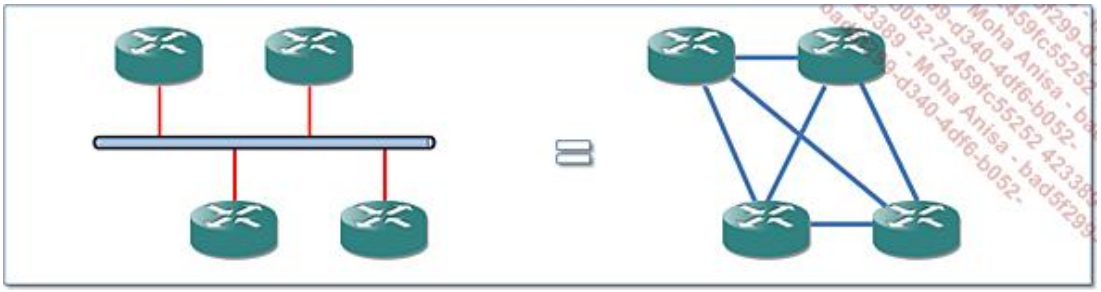
L'algorithme de Dijkstra s'applique à des graphes composés de nœuds et d'arcs. Pour l'appliquer à la réalité de nos réseaux informatiques, il a fallu procéder à certains aménagements :

- Pas de problème avec les routeurs qui sont évidemment les nœuds du graphe.

- Pas plus de problème avec les liaisons point à point qui sont naturellement les arcs du graphe. Deux voisins connectés par un réseau point à point deviennent toujours adjacents. Les paquets OSPF sont émis vers l'adresse multicast 224.0.0.5 (qui signifie tous les routeurs OSPF, « *AllSPFRouters* »).

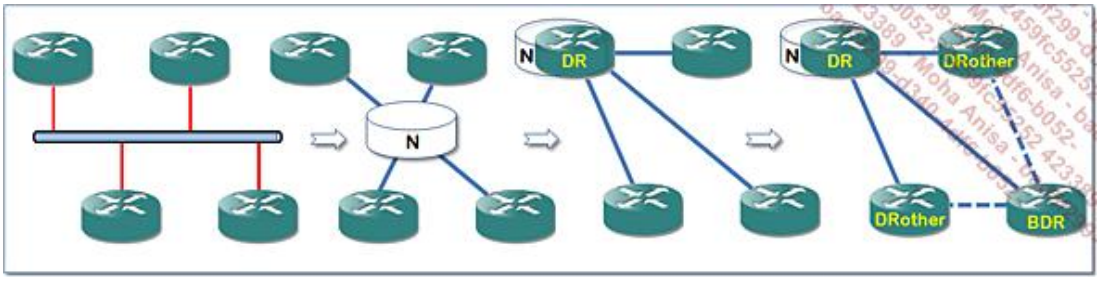
- En revanche, les réseaux à accès multiple de type LAN ou NBMA posent problème. Ces deux types de réseau permettent de relier plusieurs routeurs via un seul support physique, topologie qui n'est pas transposable en l'état dans un graphe.

Une première solution, qui n'a pas été retenue, aurait pu consister à considérer les routeurs connectés à un LAN comme étant interconnectés par un réseau intégralement maillé :



Les inconvénients d'un tel choix deviennent évidents quand le nombre de routeurs s'accroît. Si N est le nombre de routeurs interconnectés, alors chaque routeur doit entretenir *N* - 1 relations de voisinage OSPF (patientez) et le réseau doit supporter autant de trafic lié aux relations de voisinage qu'il y a de relations bilatérales à entretenir soit $\frac{N \times (N-1)}{2}$ relations (1 relation pour 2 routeurs, 3 pour 3, 6 pour 4, 10 pour 5, 15 pour 6, 21 pour 7...).

La solution retenue par l'IETF consiste à représenter le support physique comme étant lui-même un nœud auquel chaque routeur (un nœud également) est relié par son interface réseau qui devient un arc du graphe :

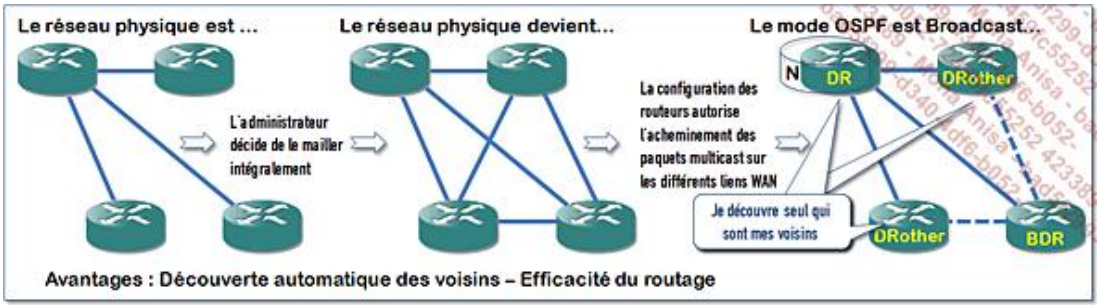


Mais puisque le nœud réseau (N pour « *Network* ») représentant le support physique ne peut participer activement au protocole OSPF, l'un des routeurs se substitue à lui et devient ainsi le routeur désigné ou DR (*Designated Router*). Pour limiter autant que possible les conséquences d'une possible défaillance du routeur désigné, la norme prévoit également un routeur désigné de secours appelé BDR (*Backup Designated Router*).

Sur un réseau à diffusion, utilisé en mode OSPF Broadcast, les messages Hello sont toujours émis vers l'adresse « *AllSPFRouters* », 224.0.0.5. De même, tous les paquets OSPF sans distinction sont émis vers l'adresse « *AllSPFRouters* », 224.0.0.5, quand ils sont issus du DR ou du BDR. Les routeurs autres que le DR ou le BDR (appelés « *DRother* ») envoient leurs paquets LSupdate (*Link State Update*) et LSAck (*Link State Acknowledgement*) vers l'adresse multicast 224.0.0.6 qui signifie les routeurs OSPF désignés (*AllDRouters*).

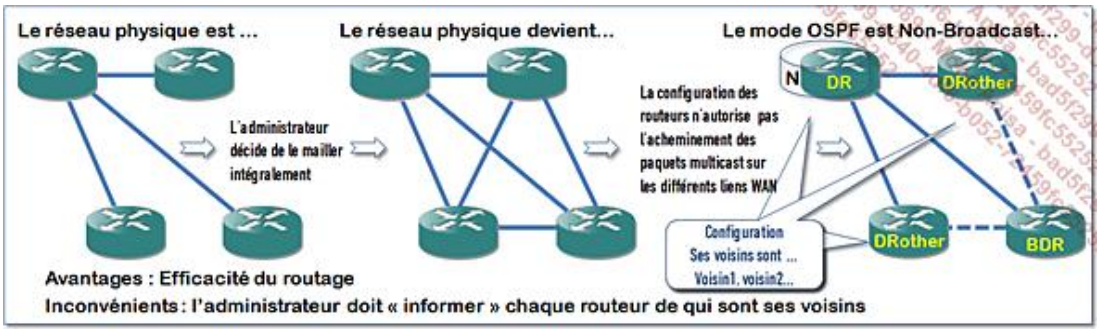
L'administrateur chargé de mettre en place OSPF sur un réseau sans diffusion devra bien maîtriser son sujet car les alternatives sont au nombre de trois.

Réseau sans diffusion, alternative 1



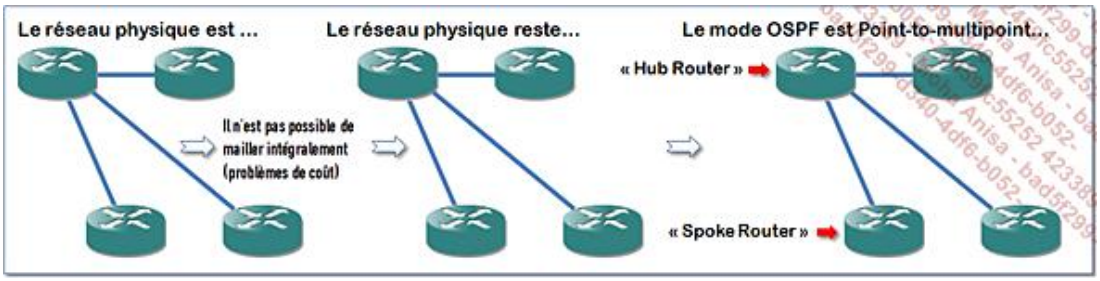
À la condition de relier tous les routeurs deux à deux, c'est-à-dire de réaliser un réseau intégralement maillé (*Fullymeshed*), il est possible de reproduire le fonctionnement d'OSPF sur un réseau à diffusion. C'est certainement la solution la plus efficace, le seul reproche que l'on puisse faire à cette solution est son coût (monétaire). En effet, s'il suffit d'établir 6 liens pour mailler 4 routeurs, il en faut 120 pour mailler intégralement 16 routeurs.

Réseau sans diffusion, alternative 2



Le réseau est encore intégralement maillé mais il n'est pas possible de faire progresser les paquets multicast nécessaires au mécanisme de découverte automatique des voisins. Au prix d'une configuration supplémentaire de chaque routeur (déclaration des voisins), OSPF procède à nouveau à l'élection d'un DR et d'un BDR. L'ensemble des échanges se fait à l'aide de paquets unicast.

Réseau sans diffusion, alternative 3



Lorsqu'il n'est pas possible de mailler intégralement les routeurs présents sur le réseau sans diffusion, il reste la possibilité de configurer les routeurs dans le mode dit « point-to-multipoint ». Puisque le comportement est identique à celui d'une collection de réseaux point à point, il n'y a pas d'élection de DR, ni de BDR. Les paquets OSPF peuvent être émis vers l'adresse multicast 224.0.0.5 (qui signifie tous les routeurs OSPF, « *AllSPFRouters* »).

d. Mécanisme d'élection d'un routeur désigné et d'un routeur désigné de secours

L'algorithme dont il est question ici est sollicité par le diagramme d'états de l'interface lors de la pseudo-étape « ? ». Remémorons-nous le contexte général qui encadre ce mécanisme d'élection :

- L'interface de chaque routeur concerné sur le réseau d'attachement est dotée d'une priorité de routeur exprimée sur 8 bits et dont la valeur par défaut s'établit à 1. L'administrateur peut à loisir modifier cette valeur à l'aide de la commande **ip ospf priority** entrée en configuration d'interface. Une valeur 0 entraîne la non-éligibilité de l'interface qui n'est donc pas concernée par l'élection.

- Observez le format du message Hello fourni un peu plus avant pour considérer trois champs intervenant dans le mécanisme d'élection : le champ « *Priority Router* » qui annonce la priorité de ce routeur ainsi que les champs DR et BDR qui annoncent les adresses IP des interfaces que le routeur auteur du message considère comme étant celles des DR et BDR actuels.

- Le passage à l'état « *Waiting* » du diagramme d'états de l'interface provoque l'initialisation des variables DR et BDR à 0.0.0.0.

- Une communication bidirectionnelle (*2-Way*) est établie avec un ou davantage de voisins, la matière première du mécanisme consiste en ces trois valeurs priorité, DR et BDR annoncées par chacun des voisins dans ses messages Hello.

Décrivons dans un premier temps ce mécanisme de façon littérale, il sera commenté ensuite. Nous sommes placés sur le routeur X :

- Mémoriser les valeurs actuelles des variables DR et BDR afin de pouvoir les comparer aux valeurs obtenues après exécution de l'algorithme.

- Composer les listes de routeurs suivantes :

- « E » est la liste des routeurs éligibles c'est-à-dire ceux avec qui une relation bidirectionnelle au moins est établie et dont la priorité est différente de 0. Le routeur X s'inclut dans cette liste.

- Parmi les routeurs de la liste « E », extraire les routeurs se déclarant BDR pour composer la liste « B ».

- Parmi les routeurs de la liste « E », extraire les routeurs se déclarant DR pour composer la liste « D ».

- « *I* » est la liste des routeurs inscrits, c'est-à-dire éligibles à l'exclusion de ceux se déclarant DR → $I = E - D$.

- « *IB* » est la liste des BDR inscrits, c'est-à-dire les routeurs de la liste *I* qui se déclarent BDR → $IB = I \cap B$ (∩ est le symbole de l'intersection).

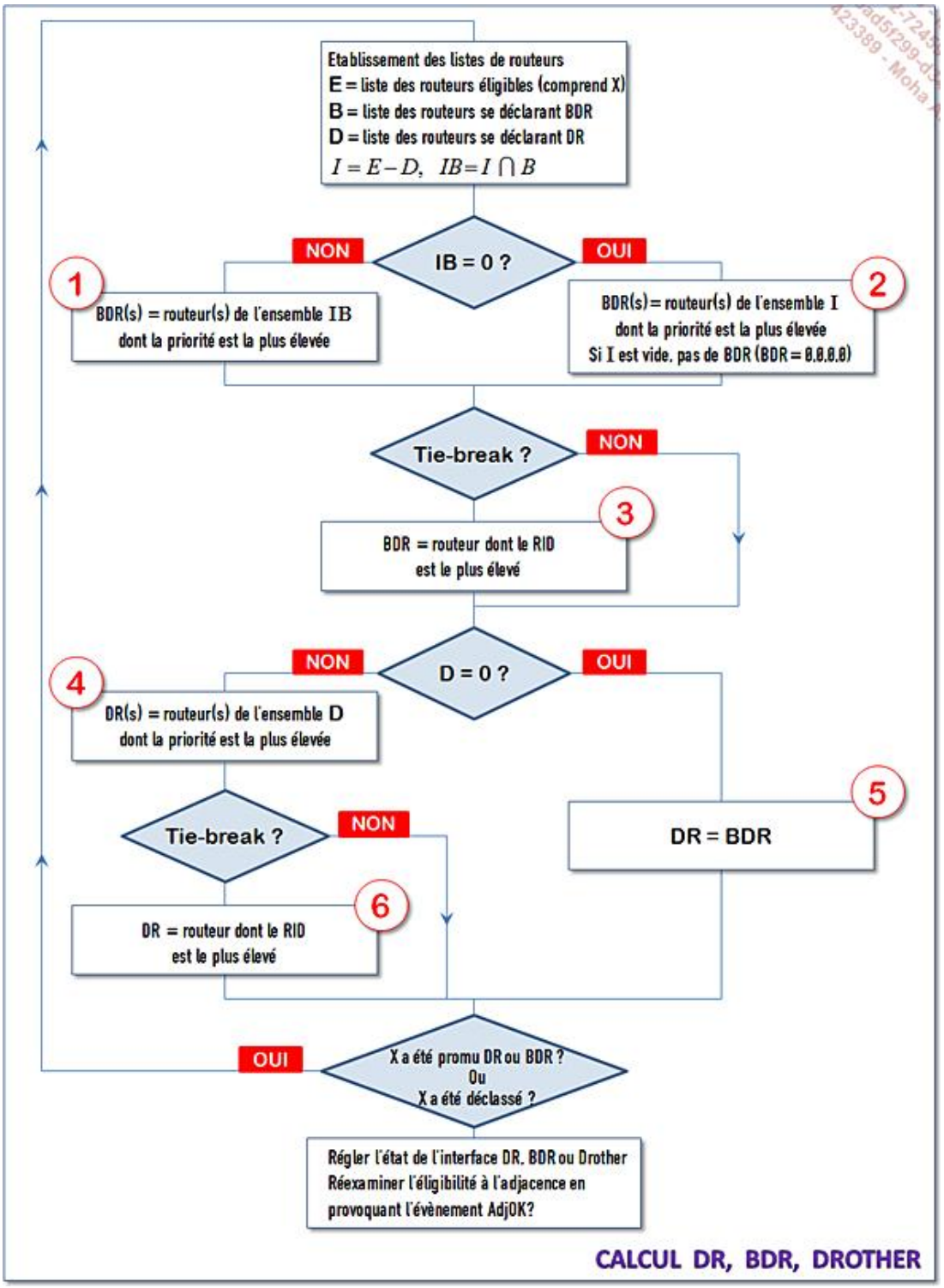
- L'algorithme désigne BDR le routeur de l'ensemble *IB* (routeurs éligibles se déclarant BDR sans se déclarer DR) dont la priorité est la plus élevée. Si cet ensemble est vide, l'algorithme désigne BDR le routeur de l'ensemble *I* (routeurs éligibles qui ne se déclarent pas DR) dont la priorité est la plus élevée. Si la priorité ne suffit pas à départager plusieurs routeurs, le routeur choisi est celui dont le RID est le plus élevé.

- L'algorithme désigne ensuite DR le routeur de l'ensemble D (routeurs éligibles se déclarant DR) dont la priorité est la plus élevée. Si la priorité ne suffit pas à départager plusieurs routeurs, le routeur choisi est celui dont le RID est le plus élevé. Si l'ensemble D est vide, l'algorithme désigne DR le routeur choisi en tant que BDR à l'étape précédente et à cet instant, pour l'algorithme, les rôles DR et BDR sont donc portés par un même routeur.

- Si le routeur X a été désigné DR ou BDR alors que X n'était pas porteur de ce rôle avant exécution de l'algorithme ou si au contraire, X a été déchu et n'est plus porteur de ce dont il disposait avant exécution de l'algorithme, alors l'algorithme est exécuté une nouvelle fois. Par exemple, si le routeur X s'est désigné DR au premier passage dans l'algorithme, il n'est plus éligible BDR au second passage, on obtient ainsi la garantie qu'aucun routeur ne puisse se proclamer DR et BDR.

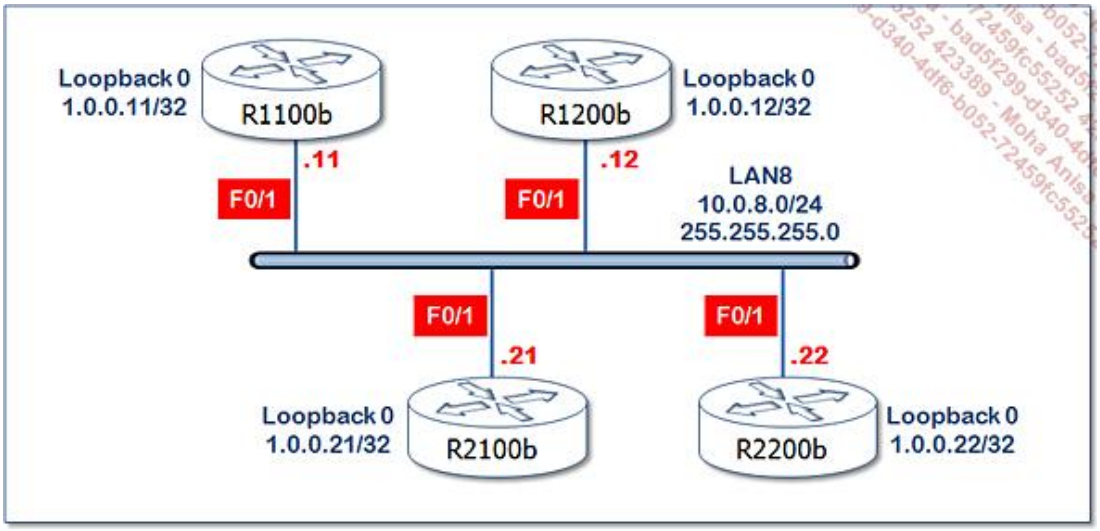
- Au terme du calcul, le diagramme d'états de l'interface doit rejoindre l'un des trois états DR, BDR ou DRother. Pour ce faire, l'algorithme génère l'un des trois pseudo-évènements « *Elected DR* », « *Elected BDR* » ou « *Not elected* ».

- Puisque dans un réseau à diffusion ou dans un réseau en mode NBMA, chacun des DRother ne bâtit une relation de proximité qu'avec le DR et le BDR, tout changement d'identité de l'un des routeurs désignés doit s'accompagner d'une remise en question de l'ensemble des relations de voisinage (Mon voisin déménage, j'étais proche de lui, je ne le serai plus. Je serai peut-être proche du nouveau locataire). Ceci est obtenu en générant l'évènement « *AdjOK* ? » dans le diagramme d'états associé à chaque relation de voisinage dont l'état était au moins « *2-Way* » (patientez).

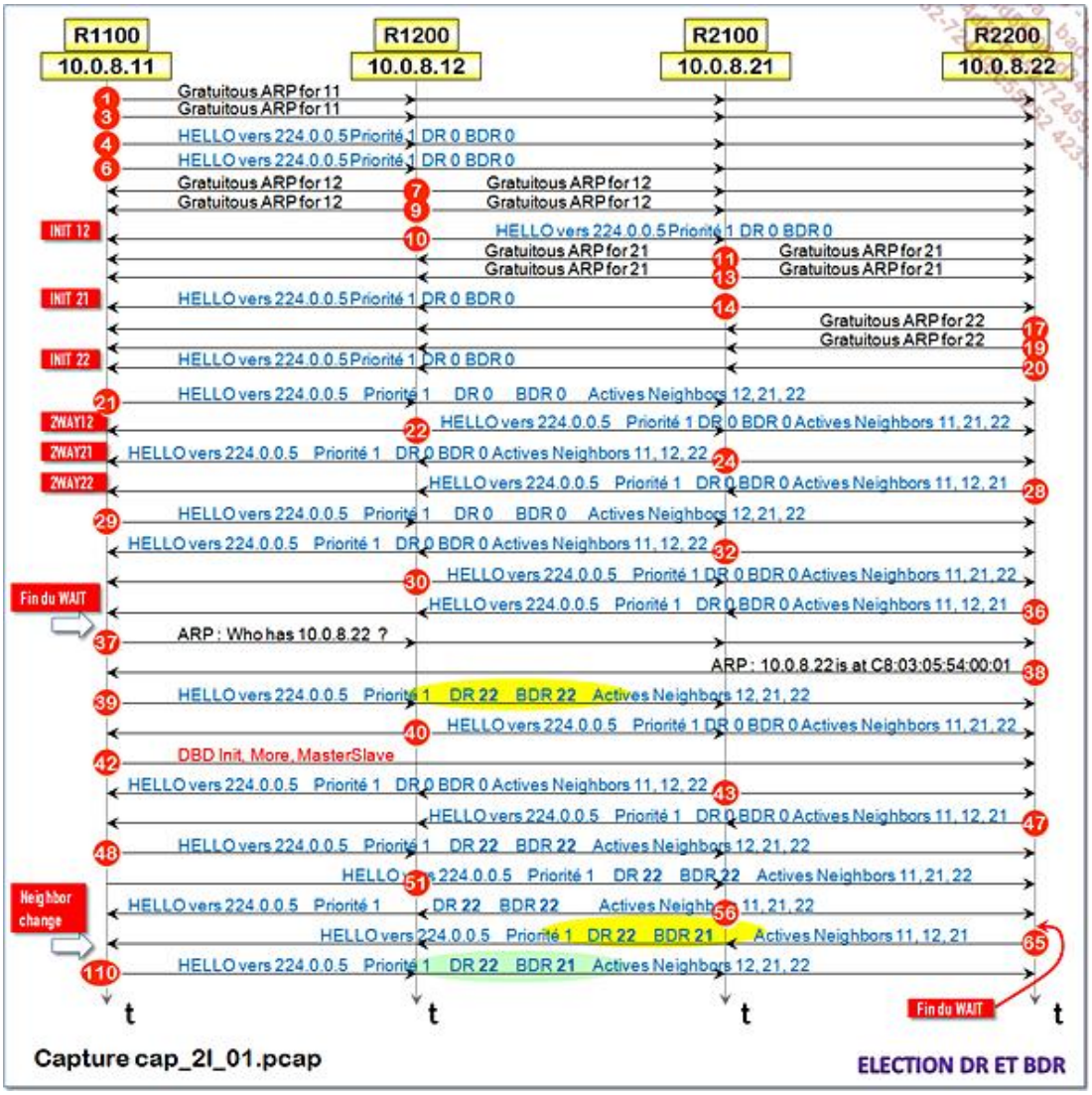


Forts de cet algorithme, imaginons quelques scénarios. Quand un routeur devient actif sur un réseau à diffusion ou NBMA, il se place à l'écoute à la recherche de routeurs désignés existants. S'il découvre DR et BDR, il les accepte. S'il est seul sur le réseau, il se déclare BDR puis devient DR et il n'y a pas de BDR. Si, sur un réseau constitué où DR et BDR existaient, le DR vient à disparaître, l'évènement « *NeighborChange* » se produit qui provoque l'exécution de l'algorithme par chacun des autres routeurs. Chaque routeur désigne le BDR comme nouveau DR mais sans remplacer le BDR. Seul le routeur BDR, qui exécute également l'algorithme, accepte d'être le nouveau DR et s'annonce comme tel. Quand les autres routeurs perçoivent un message Hello de l'ex-BDR qui s'annonce désormais DR, ceci provoque un nouvel évènement « *NeighborChange* » et par suite, une nouvelle exécution de l'algorithme. L'un des routeurs devient le nouveau BDR et s'annonce en tant que tel ce qui provoque une dernière fois l'évènement « *NeighborChange* » et l'exécution de l'algorithme par les autres routeurs qui acceptent alors le nouveau BDR.

On le voit, la priorité influence le résultat d'une élection mais ne peut remettre en question les rôles déjà attribués. Un DR reste DR même si un routeur à priorité plus élevée arrive sur le réseau. Seule la défaillance du DR peut provoquer l'élection d'un nouveau DR. Une fois les DR et BDR élus, chacun des autres routeurs établit une relation de proximité avec les deux routeurs désignés et seulement avec eux. Deux routeurs « DRother » ne deviennent jamais adjacents. Mettons l'algorithme à l'épreuve avec cette mise en situation :



Les routeurs ont été démarrés dans l'ordre R1100, R1200, R2100, R2200. La capture correspondante **cap_2i_01.pcap** est disponible en téléchargement sur le site des Editions ENI.



11, 12, 21 et 22 compactent indifféremment l'adresse IP de l'interface ou le RID du routeur dans l'illustration ci-dessus. Plaçons-nous sur le routeur 11 (R1100).

- Toute interface qui s'active sur un réseau à diffusion commence par émettre deux paquets ARP gratuits afin de vérifier que l'adresse IP attribuée à l'interface n'est pas dupliquée sur le réseau de rattachement. Le diagramme d'états de l'interface passe à l'état « *Waiting* », ce qui active l'émission de messages Hello

périodiques. Les deux premiers messages Hello des trames 4 et 6 disent DR = 0.0.0.0, BDR = 0.0.0.0 et pas de voisins actifs.

- Le voisin physique 12 s'active, émet ses deux ARP gratuits, trames 7 et 9, puis son premier message Hello trame 10.

- La réception de ce message Hello par R1100 provoque la création d'une structure de données pour une nouvelle relation de voisinage, le diagramme d'états de cette relation passe à l'état « *Init* » (patientez).

- De même, la réception des premiers messages Hello émis par les voisins 21 et 22, trames 14 et 20, crée à nouveau deux structures de donnée. Dans chaque cas, le diagramme d'états de la relation de voisinage passe à l'état « *Init* ».

- Au moment d'émettre le message Hello suivant en trame 21, R1100 a accumulé une connaissance plus fine de son entourage, il peut annoncer les trois voisins qu'il a découvert.

- De même, les messages Hello émis par 12, 21 et 22 dans les trames 22, 24 et 28 nous apprennent que chacun de ces routeurs a découvert ses trois voisins.

- En trame 22, la réception d'un message Hello dans lequel 12 dit « 11 est un de mes voisins actif » (11 se voit dans le message) provoque le passage du diagramme d'états de la relation de voisinage à l'état « *2-Way* », la communication bidirectionnelle est établie. Pour 11, 12 est désormais candidat potentiel à l'élection des routeurs désignés.

- De même, la réception du Hello en trame 24 fait passer la relation de voisinage avec 21 à l'état « *2-Way* ».

- Enfin, la réception du Hello en trame 28 fait passer la relation de voisinage avec 22 à l'état « *2-Way* ».

- Il ne se passe rien de nouveau entre la trame 28 et la trame 36 parce que chacun des quatre protagonistes est encore dans l'état « *Waiting* ». R1100, premier démarré, est donc le premier à en sortir environ 40 secondes après son démarrage. R1100 déroule une première fois l'algorithme de calcul DR, BDR. Faisons-le avec lui :

    - *E = {11,12,21,22}, B = { }, D = { }, I = {11,12,21,22}, IB = { }*

    - *L'ensemble IB est vide, l'algorithme passe par la case étiquetée 2, les quatre candidats ont même priorité.*

    - *En case 3 de l'algorithme, c'est le RID qui permet de départager les candidats, 22 est désigné BDR.*

    - *L'ensemble D est vide, l'algorithme passe par la case 5, 22 est désigné DR.*

    - *Au sortir de l'algorithme, R1100 n'a été ni promu, ni déclassé. Il n'y a donc pas bouclage de l'algorithme. Le diagramme d'états de l'interface passe à l'état « DRother ».*

- Dans la capture, nous n'apprenons le résultat de ce calcul qu'en trame 39, c'est-à-dire bien après que le calcul ait été effectué. Pourtant, on peut déjà deviner ce résultat au vu de l'émission d'une requête ARP en trame 37 : « *Who has* 10.0.8.22 ? ». Ainsi, R1100 qui sait que R2200 sera DR, s'apprête à lui décrire sa LSD à l'aide de paquets OSPF unicast et pour ce faire, doit découvrir l'adresse physique du futur DR.

- Le message Hello en trame 39 confirme ce que nous pressentions, R1100 annonce 22 à la fois BDR et DR.

- En trame 42, R1100 dont la relation de voisinage avec le futur DR est passée à l'état « *ExStart* », tente de régler les paramètres de l'échange des paquets DBD de description de base de données, mais il n'obtient pas de réponse car R2200 est encore dans l'état « *Waiting* ».

- De la trame 42 à la trame 61, rien ne change car le routeur R2200 est encore dans l'état « *Waiting* ». À son tour, R1200 est passé à l'état « *ExStart* » et émet le premier paquet DBD en trame 59. Pendant ce temps, R1100 qui n'obtient pas de réponse, répète son premier paquet DBD tous les « *RxmtInterval* = 5 secondes » (trames 42, 49, 54).

- Dans les trames 62, 63 et 64, R2200 répond aux paquets DBD reçus des routeurs 11, 12 et 21. On peut en

déduire que R2200 est sorti de l'état « *Waiting* » et a déroulé l'algorithme de calcul DR, BDR. Il le fait avec la même matière première que R1100 et arrive au même résultat, mais cela correspond pour lui à une promotion. L'algorithme est donc exécuté à nouveau. Faisons ce deuxième tour avec lui :

- *E = {11,12,21,22}, B = {22}, D = {22}, I = {11,12,21}, IB = { }*
- *L'ensemble IB est vide, l'algorithme passe par la case étiquetée 2, les 3 candidats de l'ensemble I ont même priorité.*
- *En case 3 de l'algorithme, c'est le RID qui permet de départager les candidats, 21 est désigné BDR.*
- *L'élection du DR au premier tour n'est pas remise en question par ce second tour.*

- Le message Hello émis par R2200 en trame 65 confirme ce résultat, R2200 s'annonce DR et annonce 21 en tant que BDR.
- Vu des autres routeurs, R2200 se déclare DR et c'est nouveau, ce qui se traduit par la génération de l'évènement « *NeighborChange* » dans chacun des diagrammes d'états associés aux interfaces. Chacun de ces diagrammes repasse au pseudo-état « ? », il s'en suit une nouvelle exécution de l'algorithme de calcul DR, BDR. Plaçons-nous sur R1100 et faisons-le avec lui :
  - *E = {11,12,21,22}, B = { }, D = {22}, I = {11,12,21}, IB = { }*
  - *L'ensemble IB est vide, l'algorithme passe par la case étiquetée 2, les 3 candidats de I'ensemble I ont même priorité.*
  - *En case 3 de l'algorithme, c'est le RID qui permet de départager les candidats, 21 est désigné BDR.*
  - *Le passage par la case 4 de l'algorithme ne remet pas en question le choix du DR.*
- Le message Hello émis par R1100 en trame 110 confirme ce résultat.

En forme de synthèse, une commande **show** montre l'état du voisinage sur R1100 :

```
R1100b#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address       Interface
1.0.0.22          1   FULL/DR         00:00:39    10.0.8.22     FastEthernet0/1
1.0.0.21          1   FULL/BDR        00:00:37    10.0.8.21     FastEthernet0/1
1.0.0.12          1   2WAY/DROTHER    00:00:32    10.0.8.12     FastEthernet0/1
```

5. Voisinage et proximité

a. Vivons en bon voisinage

Deux routeurs qui se découvrent construisent une relation de voisinage. Cette relation passe par différents états. Une bonne analogie peut être faite avec les relations qui nous lient, nous les hommes, avec nos voisins. Imaginez l'appartement voisin du vôtre inoccupé. Voilà qu'arrive un nouveau locataire.

La première étape consiste à se découvrir mutuellement : « Tiens, j'ai un nouveau voisin ! » et du point de vue du voisin « Tiens, j'ai un voisin ! ». La seconde étape devrait être, à l'occasion d'une rencontre fortuite ou provoquée, de se saluer : « Hello, je me présente Brett Sinclair ! » … « Hello, enchanté, Danny Wilde ! ».

Les quelques propos échangés pendant les premières rencontres permettent rapidement de savoir si la relation avec ce voisin va en rester là, c'est-à-dire s'en tenir à un échange régulier de salutations lors des rencontres inévitables dans la cage d'escalier ou sur le parking, la relation reste alors « de voisinage », ou va évoluer vers une relation plus construite qui peut aller jusqu'à se recevoir parce que l'on a des choses à partager, parce que passer un moment ensemble nous fait plaisir. De voisinage, la relation devient de proximité.

De la même façon, deux routeurs voisins physiquement vont tenter de construire une relation de voisinage dont l'état ultime peut être une relation de proximité (*adjacency*). Afin de se découvrir mutuellement puis afin de maintenir l'état de leurs liens par une surveillance réciproque, les routeurs OSPF utilisent un protocole de communication appelé « *OSPF Hello* ».

b. Le protocole Hello

Quoi de plus naturel qu'un protocole Hello pour faire connaissance. Inventorions les usages qu'en fait OSPF :

- Cela a été dit, il permet à deux voisins de se découvrir.
- Les messages Hello comportent un certain nombre de paramètres au sujet desquels les deux routeurs concernés par une relation en cours de construction doivent se mettre d'accord avant de pouvoir effectivement devenir voisins ou proches.
- Les messages Hello puisqu'ils sont échangés de façon régulière permettent de maintenir la relation « en vie » (*Keep alive*).
- L'élection des routeurs désignés DR et BDR s'opère parmi les interfaces connectées au réseau concerné, à diffusion ou sans diffusion dans le mode NBMA, lorsque ces interfaces sont dans l'état « voisin » (*2-Way*, patientez).

Un routeur OSPF génère des messages Hello sur toutes ses interfaces participant au protocole, ce de façon régulière. Pour tous les types de réseau prévus par OSPF en dehors du mode NBMA, la période qui sépare deux messages Hello est « *HelloInterval* », ce délai est configurable par interface à l'aide de la commande **ip ospf hello-interval** en configuration d'interface. L'implémentation OSPF de CISCO utilise une valeur par défaut de 10 secondes. Sur un réseau qui fonctionne selon le mode NBMA, les messages Hello sont espacés du délai « *PollInterval* » dont la valeur par défaut sur les routeurs CISCO s'établit à 120 secondes.

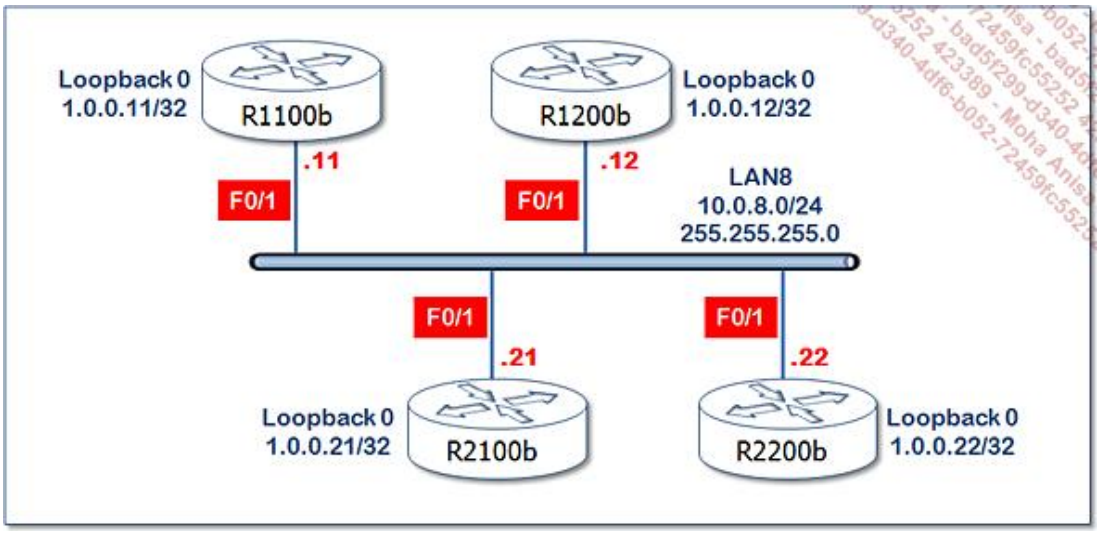
Une fois ajouté à la table de voisinage, un voisin fait l'objet d'une surveillance régulière. Si les messages Hello que le routeur s'attend à recevoir ne lui parviennent plus pendant un temps d'observation suffisant, le voisin est considéré comme perdu (il n'est donc plus voisin). C'est le délai « *RouterDeadInterval* » qui règle la durée de ce coma pré mortem. Cisco s'en tient à la valeur suggérée par le RFC, soit 4 x *HelloInterval*, c'est-à-dire 40 secondes par défaut. À nouveau, cette valeur est modifiable à l'aide de la commande **ip ospf dead-interval** en configuration d'interface :

```
R800(config-if)#ip ospf ?
authentication       Enable authentication
authentication-key   Authentication password (key)
cost                 Interface cost
database-filter      Filter OSPF LSA during synchronization and flooding
dead-interval        Interval after which a neighbor is declared dead
demand-circuit       OSPF demand circuit
flood-reduction      OSPF Flood Reduction
hello-interval       Time between HELLO packets
message-digest-key   Message digest authentication password (key)
mtu-ignore           Ignores the MTU in DBD packets
network              Network type
priority             Router priority
  retransmit-interval  Time between retransmitting lost link state
advertisements
transmit-delay       Link state transmit delay

R800(config-if)#ip ospf dead-interval ?
<1-65535>  Seconds

R800(config-if)#ip ospf hello-interval ?
<1-65535>  Seconds
```

c. Base de données de voisinage

Un routeur qui doit construire un message Hello vers un réseau le fait en puisant l'information dans la structure de données associée à l'interface connectée à ce réseau. Puisque cette information est particulière à chaque interface, un même routeur génère autant de messages Hello différents qu'il a d'interfaces participant au protocole. Ce faisant, le routeur informe ses voisins de ce qu'il est (il se présente). De la même façon, à chaque fois qu'il découvre un nouveau voisin, le routeur crée une nouvelle entrée (un enregistrement) dans sa base de données de voisinage. À cette entrée, il associe un certain nombre d'informations extraites des messages Hello qu'il reçoit de ce voisin.

Il est possible de se faire une idée des informations associées à une entrée de la base de données de voisinage à l'aide d'une commande **show ip ospf neighbor**. Précisons d'abord le contexte :



Dans ce contexte, le résultat de la commande entrée sur le routeur R1100b :

```
R1100b#sh ip ospf neighbor 1.0.0.12
Neighbor 1.0.0.12, interface address 10.0.8.12
    In the area 0 via interface FastEthernet0/1
Neighbor priority is 1, State is 2WAY, 2 state changes
DR is 10.0.8.22 BDR is 10.0.8.21
    Options is 0x2
Dead timer due in 00:00:38
    Neighbor is up for 00:01:11
    Index 0/0, retransmission queue length 0, number of retransmission 0
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 0, maximum is 0
    Last retransmission scan time is 0 msec, maximum is 0 msec
```

La section suivante fournit un premier détail du format du message OSPF Hello. Au prix d'une petite navigation entre cette page et la section Format des messages OSPF - Le message Hello, le lecteur pourra repérer la plupart des éléments suivants dans le message Hello. Les éléments importants associés à chaque entrée sont :

- L'identifiant RID du voisin (*Neighbor ID*), dans le cas présent 1.0.0.12.
- L'adresse IP de l'interface du routeur voisin connectée à ce réseau, dans le cas présent 10.0.8.12. Cette adresse a été apprise à l'aide du champ adresse source de l'en-tête IP. Si le processus OSPF de ce routeur a besoin d'envoyer un message vers l'adresse unicast de ce voisin, alors il peut utiliser cette adresse en tant qu'adresse de destination.
- L'identifiant d'aire (*Area ID*), dans le cas présent 0. Pour que deux routeurs voisins physiquement puissent devenir voisins au sens OSPF, l'identifiant d'aire contenu dans le message Hello reçu doit correspondre (être identique à : « *matches* ») à l'identifiant d'aire configuré sur l'interface de réception. Dans le cas présent, l'interface F0/1 du routeur R1100b participe au processus OSPF pour l'aire 0. Le message Hello reçu contenait un identifiant d'aire 0. Tout va bien, ce critère n'empêche pas les deux interfaces de devenir voisines.
- L'interface par laquelle ce routeur est connecté au réseau sur lequel se trouve le voisin. Dans le cas présent, l'interface FastEthernet0/1.
- La priorité du voisin (*Neighborpriority*) telle qu'elle est reçue dans les messages Hello. Dans le cas présent, la priorité est de 1, soit la valeur par défaut sur les routeurs CISCO. Cette priorité est utilisée par le mécanisme d'élection des routeurs désignés DR et BDR sur les réseaux à diffusion (notre contexte) ainsi que sur les réseaux sans diffusion fonctionnant en mode NBMA.
- L'état de la relation de voisinage (*State*) qui a atteint dans le cas présent la valeur « *2-Way* ». Sur un réseau à diffusion, les seules relations qui vont jusqu'à l'état de pleine adjacence « *FULL* » sont les relations établies avec le routeur désigné ainsi qu'avec le routeur désigné de secours. 1.0.0.12 n'étant ni l'un, ni l'autre, la relation ne dépasse pas l'état « *2-Way* ».
- Le temporisateur d'inactivité (*InactivityTimer*). À chaque message Hello reçu, le routeur arme ce temporisateur. Si aucun nouveau message Hello n'est reçu avant « *RouterDeadInterval* » secondes, la relation

de voisinage repasse à l'état initial soit l'état « *Down* ». La commande **show** nous informe qu'il reste 31 secondes avant l'échéance.

- Le routeur désigné DR tel qu'il est inclus dans le champ DR du message Hello reçu.

- Le routeur désigné de backup BDR tel qu'il est inclus dans le champ BDR du message Hello reçu.

Dans le contexte d'un réseau fonctionnant en mode NBMA, la même commande **show ip ospf neighbor** fait apparaître un élément supplémentaire :

```
R1100c#sh ip ospf neighbor 1.0.0.12
 Neighbor 1.0.0.12, interface address 10.0.8.12
    In the area 0 via interface Serial0/0
    Neighbor priority is 1, State is 2WAY, 7 state changes
    DR is 10.0.8.22 BDR is 10.0.8.21
    Poll interval 120
    Options is 0x2
    Dead timer due in 00:01:49
    Neighbor is up for 00:16:12
    Index 0/0, retransmission queue length 0, number of retransmission 0
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 0, maximum is 0
    Last retransmission scan time is 0 msec, maximum is 0 msec
```

- Le temporisateur « *PollInterval* ». Cette valeur est associée à l'entrée (enregistrement) uniquement quand cette interface et l'interface voisine sont connectées à un réseau sans diffusion qui fonctionne en mode NBMA. Si la relation de voisinage est à l'état « Down » (patientez), un message Hello est envoyé au voisin tous les « *PollInterval* » secondes, période plus longue que celle fixée par la valeur « *HelloInterval* » en usage sur les autres types de réseau. L'implémentation OSPF de CISCO utilise un délai « *PollInterval* » fixé à 120 secondes.

Ces éléments existent dans chaque enregistrement de la base de données de voisinage sans toutefois être affichés lors de la commande **show ip ospf neighbor** :

- Le booléen « *Master/Slave* ». Dans l'état « *ExStart* » de la relation de voisinage (patientez), les deux voisins négocient afin de déterminer qui sera le maître de la relation de proximité en cours de construction.

- Le numéro de séquence en cours (*DD SequenceNumber*) pour les envois de paquets de description de base de données LSD. Le prochain paquet DBD envoyé au voisin portera ce numéro.

- Les bits « *Initial* », « *More* » et « *Master* » (patientez), les options ainsi que le numéro de séquence du dernier paquet de description de base de données LSD reçu. Cette information permet au processus OSPF de ce routeur de distinguer un paquet DBD reçu qui serait dupliqué.

- La liste « *Link State Retransmission List* ». C'est l'une des trois listes importantes dans la synchronisation des LSD de deux routeurs adjacents. Cette liste contient les LSAs qui ont été inondés sur le proche sans avoir été acquittés. À chaque LSA inondé, le processus OSPF arme un temporisateur dont l'échéance est réglée à « *RxmtInterval* » secondes. À moins qu'entre temps, l'adjacence ne soit détruite (absence de message Hello reçu pendant « *RouterDeadInterval* » et retour de la relation à l'état « Down ») le processus OSPF retransmet à nouveau un LSA qui n'aurait pas été acquitté quand le temporisateur associé arrive à échéance.

- La liste « *Link State Summary List* ». Cette liste contient l'ensemble des LSAs qu'il faudra décrire au voisin lorsque la relation de voisinage parviendra à l'état « *Exchange* ». Cette liste est normalement constituée lors de l'évènement « *NegotiationDone* » dans le diagramme d'états de la relation de voisinage (patientez).

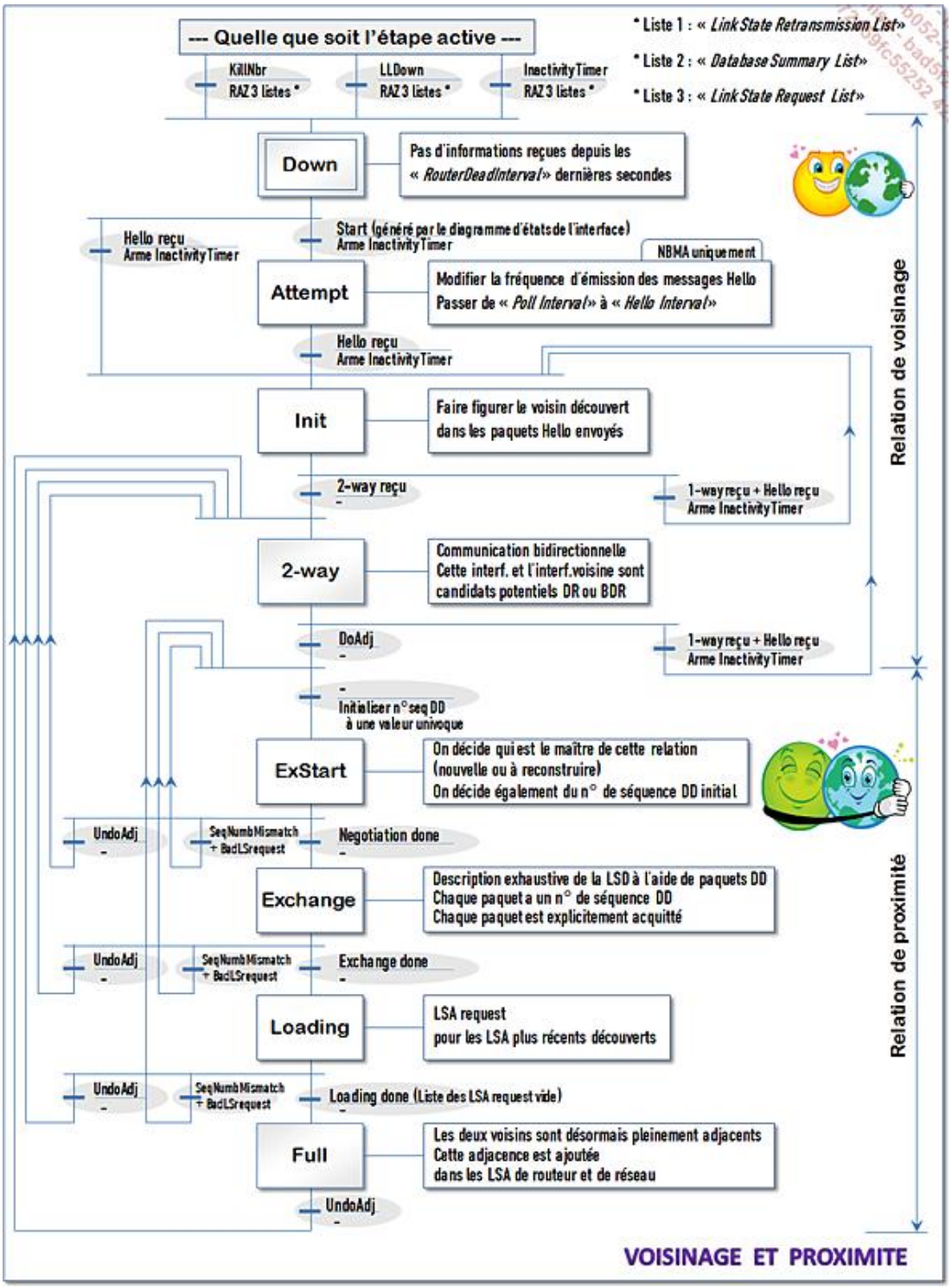
- La liste « *Link State Request List* ». Cette liste contient l'ensemble des LSAs du voisin qui ont été découverts plus récents que les versions contenues dans la LSD de ce routeur. Ces LSAs qu'il faut obtenir feront l'objet de paquets « *Link State Request* ». Au fur et à mesure que les réponses parviennent via des paquets OSPF « *Link State Update* », la liste « *Link State Request List* » s'appauvrit. L'évènement « *LoadingDone* », dans le diagramme d'états de la relation de voisinage, ne peut se produire qu'une fois cette liste vide (patientez).

Les trois listes « *Link State Retransmission List* », « *Data Summary List* » et « *Link State Request List* » ne sont que des listes, c'est-à-dire qu'elles ne contiennent que les identifiants de LSA, qui jouent le rôle de pointeurs vers les enregistrements LSAs de la base de données LSD.

Reportez-vous à la section Format des messages OSPF - Le message Hello afin d'observer les différents champs qui permettent au routeur qui reçoit ce message d'alimenter les différents éléments associés à l'entrée (l'enregistrement) de la base de données de voisinage. Les champs marqués par la petite flèche sont ceux qui doivent correspondre entre ce routeur et le routeur voisin pour qu'ils puissent prétendre devenir voisins au sens OSPF.

d. États de la relation de voisinage

À nouveau, mettons le GRAFCET à profit pour tenter d'y voir clair dans les états de la relation de voisinage. Le RFC évoque souvent la notion d'état supérieur à ... et nous ferons de même. Expliquons-nous : tous les états sont supérieurs à l'état « *Down* » mais il n'y a pas d'état supérieur à l'état « *Full* » ; les états supérieurs à l'état « *Exstart* » sont les états « *Exchange* », « *Loading* » et « *Full* ».



Un routeur OSPF entretient autant de diagrammes d'états de la relation de voisinage qu'il a découvert de voisins.

Gardons-nous bien de traduire les termes employés par le RFC, ce sont la plupart du temps ces mêmes termes qu'utilise CISCO dans son implémentation d'OSPF et donc dans les messages fournis par l'interface ILC. Souvenons-nous également que l'évolution du diagramme d'états de chaque relation de voisinage est étroitement liée à l'évolution du diagramme d'états de l'interface. La compréhension est facilitée en conservant les deux diagrammes sous les yeux. Commentons ce diagramme d'états :

- L'état « *Down* » : c'est l'état initial d'une relation de voisinage. Il indique qu'aucune information récente n'a été reçue de ce voisin depuis « *DeadRouterInterval* » secondes (40 secondes par défaut). L'activation de l'envoi périodique de messages Hello est le fait du diagramme d'états de l'interface. Alors que sur un réseau à diffusion, les messages Hello sont diffusés et sont espacés de « *HelloInterval* » (10 secondes par défaut), dans le cas d'un réseau en mode NBMA, les messages sont envoyés vers des adresses unicast et le choix de la période qui sépare deux émissions est le fait de ce diagramme d'états à l'état « Down », la fréquence d'envoi est moindre, le délai est « *PollInterval* » (120 secondes).

- L'évènement « ***Start*** » : généré par le diagramme d'états de l'interface en plusieurs circonstances mais uniquement sur les réseaux en mode NBMA. Il s'agit en quelque sorte de « réveiller » un voisin et pour ce faire, le diagramme passe à l'état « *Attempt* ».

- L'état « *Attempt* » : cet état ne s'applique qu'aux relations de voisinage construites sur un réseau fonctionnant en mode NBMA. Il s'agit de « réveiller » le voisin avec qui il est temps d'entretenir une relation de voisinage en augmentant la fréquence d'envoi des messages Hello. De « *PollInterval* », la période qui sépare deux messages repasse à « *HelloInterval* ».

- L'évènement « *HelloReceived* » : un message Hello a été reçu du voisin mais le RID de ce (notre) routeur n'apparaît pas encore dans ce message, autrement dit le voisin n'a pas encore découvert notre existence. Que l'état actuel soit « *Down* » ou « *Attempt* », le diagramme passe à l'état « *Init* ». De plus, le temporisateur d'inactivité est armé afin de pouvoir vérifier que ce voisin est encore en vie. L'arrivée à expiration de ce temporisateur indiquerait que ce voisin est mort.

- L'état « *Init* » : à partir de l'état « *Init* » et donc dans les états supérieurs, ce routeur annonce les RID de tous les voisins qu'il connaît dans les messages Hello qu'il envoie à ce voisin. De plus chaque nouvel évènement « *HelloReceived* » réarme le temporisateur d'inactivité.

- L'évènement « *2-WayReceived* » : ce routeur voit son propre RID dans le champ « *Neighbors* » du message Hello reçu. Si l'état était « *Init* », le diagramme passe à l'état « *2-Way* ».

- L'état « *2-Way* » : ce routeur voit son propre RID dans le champ « *Neighbors* » du message Hello reçu et en conclut que la communication avec ce voisin est désormais bidirectionnelle. C'est l'état ultime de la relation de voisinage avant d'entamer un éventuel rapprochement. Sur un réseau à diffusion ou NBMA, un DR ou un BDR ne peuvent être choisis que parmi les interfaces dont les relations de voisinage ont atteint cet état ou un état supérieur.

- L'évènement « *AdjOK ?* » : cet évènement n'apparaît pas directement dans le diagramme d'états de la relation de voisinage. Il est provoqué par le diagramme d'états de l'interface quand le déroulement de l'algorithme de désignation du DR et du BDR a conclu au changement d'identité de l'un des routeurs désignés. Deux cas sont alors envisageables :

    - La relation de proximité en cours avec ce voisin n'a plus lieu d'être, il faut rompre l'adjacence et revenir à l'état « *2-Way* », ce qui est opéré à l'aide du pseudo-évènement « *UndoAdj* » (à nouveau une liberté prise avec le RFC).

    - Jusqu'à présent, la relation avec ce voisin n'avait pas dépassé l'état « *2-Way* » mais l'un des deux routeurs (lui ou moi) est devenu DR ou BDR ce qui justifie l'établissement d'une relation de proximité avec lui opérée cette fois à l'aide du pseudo-évènement « *DoAdj* ».

- L'état « *ExStart* » : première étape dans la construction d'une relation de proximité. Dans cet état, ce routeur et son voisin décident de qui sera le maître, qui sera l'esclave dans la relation de proximité en cours de construction puis déterminent le numéro de séquence DBD initial en prévision du prochain échange de paquets de description de base de données. Des deux voisins de cette relation, celui qui dispose de la plus grande adresse d'interface devient le maître.

- L'évènement « *NegotiationDone* » : la négociation maître/esclave a été achevée, les numéros de séquence DBD ont été échangés (patientez). C'est lors du franchissement de cette transition que le processus constitue la liste « *Data Summary List* ». Cette liste contient l'ensemble des LSAs qu'il faudra envoyer au voisin. L'échange des paquets de description de base de données DBD peut commencer. Le diagramme passe à l'état « *Exchange* ».

- L'état « *Exchange* » : dans cet état, le routeur décrit exhaustivement sa LSD à l'aide de paquets de description de base de données (notés DD dans le RFC mais DBD dans l'interface ILC de CISCO). Dans le même temps, le routeur reçoit des paquets DBD de son voisin et peut comparer ses LSAs avec les LSAs détenus par son voisin. Chaque fois qu'il découvre que son voisin détient un LSA plus récent ou un LSA qui lui est encore inconnu, il l'ajoute à la liste « *Link State Request List* ». Dès l'état « *Exchange* », il lui est déjà possible de demander les LSAs plus récents découverts chez ce voisin en lui envoyant des paquets « *LS Request* » qui contiennent les LSAs à obtenir extraits de cette liste.

- L'évènement « *ExchangeDone* » : ce routeur et son voisin ont terminé la description exhaustive de leur LSD. Le diagramme passe à l'état « *Loading* ».

- L'état « *Loading* » : ce routeur demande à son voisin les LSAs qu'il n'aurait pas encore obtenus à l'aide des paquets « *LS Request* ».

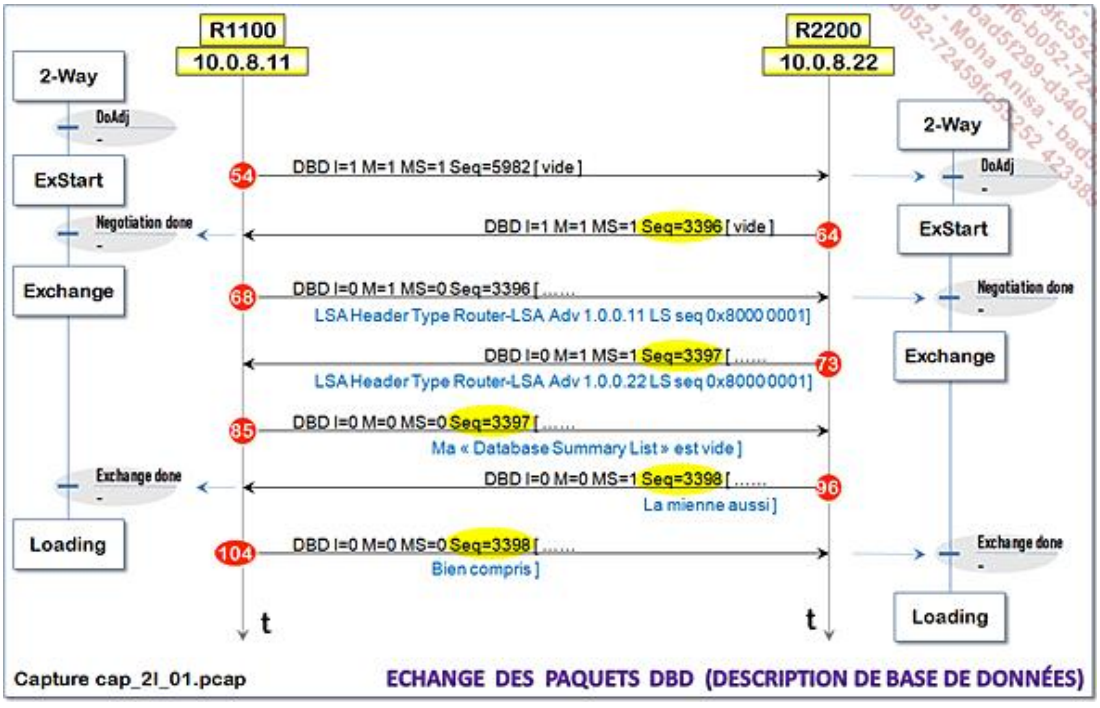
- L'évènement « *LoadingDone* » : la liste « *Link State Request List* » est vide.

- L'état « *Full* » : les deux voisins sont maintenant pleinement adjacents, en bref des proches, et cet état d'adjacence ou de proximité est digne de figurer dans les LSAs de routeur et de réseau (patientez).

e. Construction d'une relation de voisinage

Merci de bien vouloir consulter cette section en ayant également sous les yeux la section Format des messages OSPF - Le paquet DBD.

Le paquet DBD est utilisé pendant la construction d'une relation de voisinage afin de décrire les LSAs contenus dans la LSD. La description exhaustive peut nécessiter de nombreux paquets DBD et c'est pour cette raison qu'une procédure de type « polling » a été choisie. Une telle procédure consiste à désigner un maître qui prend l'initiative des échanges. L'esclave ne peut émettre un paquet DBD qu'en réponse à un paquet DBD reçu du maître. Dans le cas d'OSPF, un message DBD émis par le maître est à la fois une invitation à recevoir et une invitation à émettre. La procédure fait correspondre un message d'invitation à émettre avec sa réponse à l'aide des numéros de séquence DBD.



- Puisque R1100 a été le premier routeur démarré, il est arrivé plus vite à la conclusion qu'il lui fallait établir une relation d'adjacence avec le futur DR R2200. R1100 passe à l'état « *ExStart* » et dès la trame 42, commence l'émission de paquets de description de base de données vides, avec les bits « *Initial* », « *More* » et « *MasterSlave* » à 1. R1100 a choisi un numéro de séquence DBD égal à 5982. Ces paquets DBD vides sont répétés tous les « *RxmtInterval* » secondes, trames 49 et 54.

- R2200 sort de l'état « *Waiting* » et déroule l'algorithme de calcul DR, BDR. Il le fait avec la même matière première que R1100 et arrive au même résultat. Il est devenu DR ce qui justifie l'établissement d'une relation

de proximité avec R1100. R2200 passe à l'état « *ExStart* » et envoie à son tour un paquet DBD vide en trame 64 dont les bits I, M et MS sont positionnés. R2200 a choisi un numéro de séquence égal à **3396**.

- R1100 reçoit en trame 64 un paquet DBD vide dont les bits I, M et MS sont positionnés. R1100 observe que l'identifiant de R2200 est plus élevé. Ces conditions prises ensemble provoquent l'évènement « *Negotiationdone* ». R1100 est désormais esclave. Le franchissement de la transition provoque entre autres le remplissage de la liste « *Link State Summary List* » ainsi que l'activation de l'état « Exchange ». Le numéro de séquence reçu et accepté du maître **3396** est nouveau, il est donc interprété comme une invitation à émettre. R1100 peut émettre son premier paquet DBD non vide en trame 68. Dans ce paquet, le bit Initial est à 0 car le diagramme d'états a quitté l'état « *ExStart* », le bit « More » est à 1 car il y a encore des paquets DBD à suivre, le bit « *MasterSlave* » est à 0 car R1100 est esclave. Le numéro de séquence renvoyé fait écho au numéro de séquence reçu, soit **3396**. Le paquet contient un en-tête LSA de routeur pour le routeur dont le RID est 1.0.0.11.

- R2200 reçoit la trame 68 et y voit un paquet DBD dont les bits I et MS sont à 0 et dont le numéro de séquence est égal au numéro de séquence émis **3396**. Ces conditions prises ensemble provoquent l'évènement « *Negotiation Done* », R2200 est désormais le maître, le diagramme d'états de la relation de voisinage passe à l'état « *Exchange* ». L'esclave a fait écho au numéro de séquence émis, ceci est interprété comme un acquittement par le maître, ce qui lui permet d'émettre un nouveau paquet DBD en trame 73. Dans ce paquet, le bit Initial est à 0 car le diagramme d'états a quitté l'état « *ExStart* », le bit « More » est à 1 car il y a encore des paquets DBD à suivre, le bit « *MasterSlave* » est à 1 car R2200 est maître, le numéro de séquence est incrémenté et vaut **3397**. Le paquet contient un en-tête LSA de routeur pour le routeur dont le RID est 1.0.0.22. Si l'esclave n'avait pas renvoyé de paquet DBD faisant écho au numéro de séquence 3396, dans le cas d'une erreur de transmission par exemple, le maître aurait émis à nouveau le paquet DBD contenu dans la trame 64 après « *RxmtInterval* » secondes.

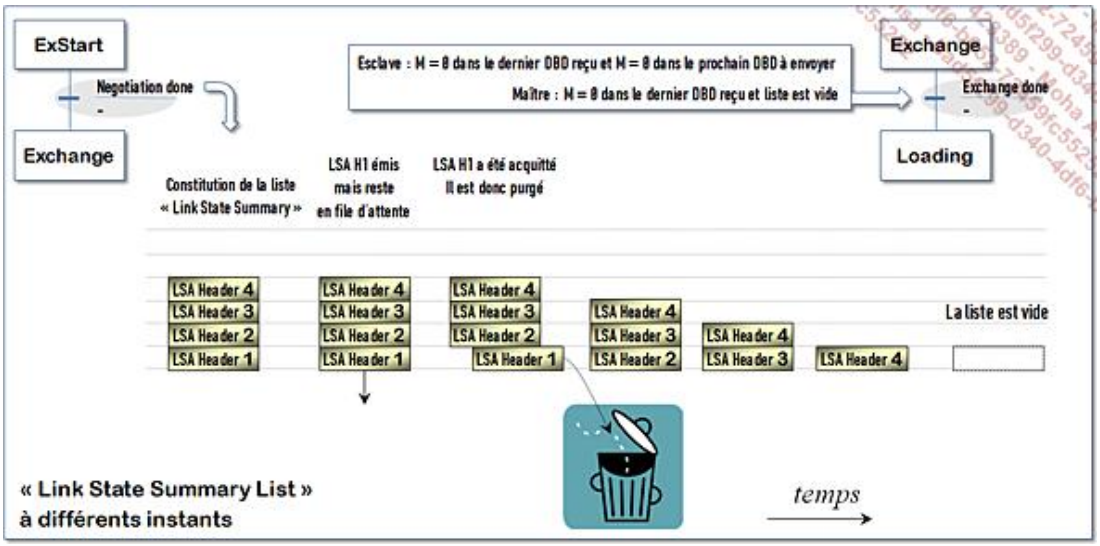
- R1100 reçoit la trame 73 et y voit un paquet DBD nouveau puisque porteur d'un nouveau numéro de séquence **3397**. Ceci est interprété comme un acquittement du paquet DBD émis en trame 68 et comme une invitation à émettre. Mais puisque la « *Link State Summary List* » de cette relation de voisinage ne comportait qu'un seul en-tête de LSA, la liste est déjà vide et le paquet DBD émis en trame 85 est un paquet vide dont le bit « More » est à 0. Le numéro de séquence renvoyé fait écho au numéro de séquence reçu, soit **3397**. Si le paquet DBD reçu avait été porteur du même numéro de séquence 3396, R1100 se serait contenté d'émettre à nouveau le paquet DBD précédent contenu dans la trame 68.

- R2200 reçoit la trame 85 et y voit un paquet DBD dont le numéro de séquence **3397** fait écho au numéro de séquence émis. Ceci est interprété comme un acquittement par le maître, ce qui lui permet d'émettre un nouveau paquet DBD en trame 96. Mais puisque la « *Link State Summary List* » de cette relation de voisinage ne comportait qu'un seul en-tête de LSA, la liste est déjà vide et le paquet DBD émis en trame 96 est un paquet vide dont le bit « *More* » est à 0, le numéro de séquence est incrémenté et vaut **3398**.

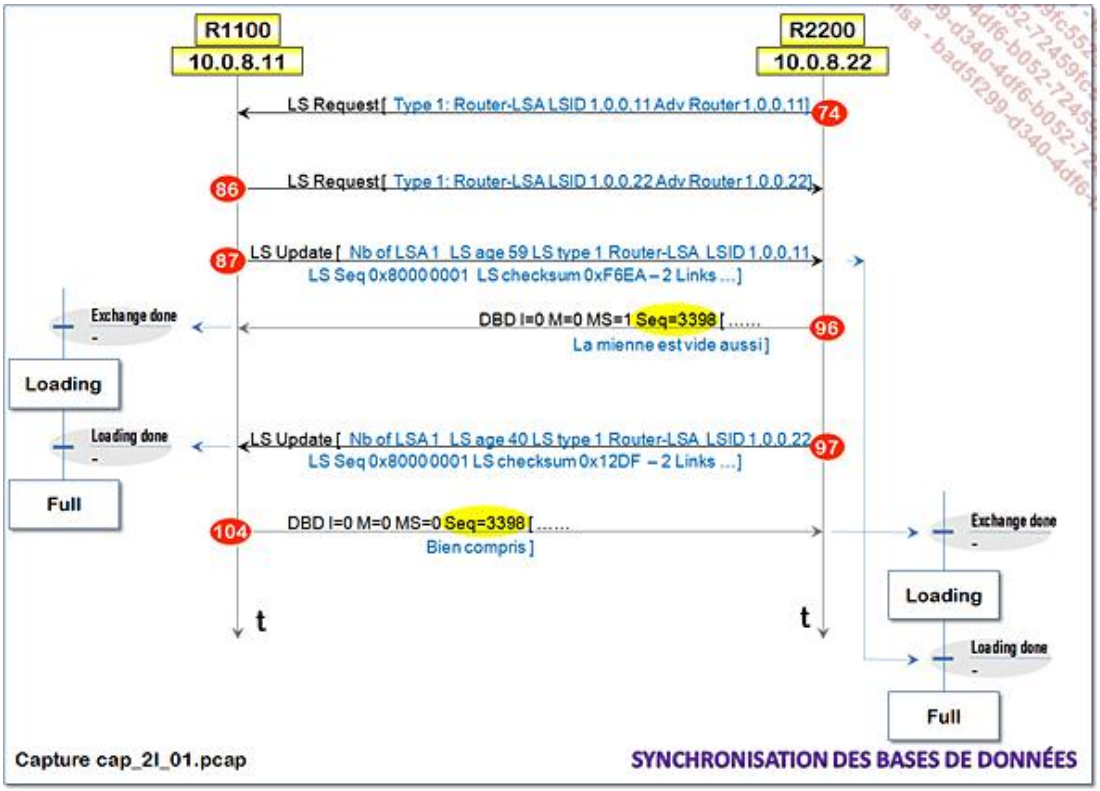
- R1100 reçoit la trame 96 et y voit un paquet DBD porteur d'un nouveau numéro de séquence **3398**. L'esclave doit renvoyer un paquet en réponse. Le paquet DBD reçu a le bit « *More* » à 0. Le paquet DBD que s'apprête à émettre l'esclave a également le bit « *More* » à 0. Ces deux conditions prises ensemble provoquent l'évènement « *Exchange done* » et par suite, l'activation de l'état « *Loading* » du diagramme d'états de la relation de voisinage. Il faut observer que l'esclave génère toujours cet évènement avant le maître. R1100 envoie un paquet DBD en trame 104 avec le bit « More » à 0 et un numéro de séquence faisant écho au numéro de séquence reçu, soit **3398**.

- R2200 reçoit la trame 104 et y voit un paquet DBD dont le numéro de séquence **3398** fait écho au numéro de séquence émis. Ceci est interprété comme un acquittement par le maître. R2200 a déjà envoyé sa séquence complète de paquets DBD et le paquet DBD reçu a le bit « *More* » à 0. Ces deux conditions prises ensemble provoquent l'évènement « *Exchange done* » et par suite, l'activation de l'état « *Loading* » du diagramme d'états de la relation de voisinage.

La figure suivante tente d'illustrer l'évolution de la « *Link State Summary List* » :



Les protagonistes sont maintenant tous deux dans l'état « *Loading* », cette phase de description de la LSD est terminée. Chaque routeur a pu comparer ses LSAs avec les LSAs détenus par son voisin. Chaque LSA découvert ou observé plus récent chez le voisin a été ajouté à la liste « *Link State Request List* ». Dès l'état « Exchange », le routeur peut déjà demander les LSAs plus récents découverts chez le voisin en lui envoyant des paquets « *LS Request* » porteurs de tout ou partie de cette liste. Dans le cas présent, la liste est des plus courtes. Pour R1100, elle ne comporte que le seul « *Link State ID* » 1.0.0.22. Pour R2200, elle ne comporte que le seul « *LS ID* » 1.0.0.11 :



La figure illustre la suite de l'exemple déjà utilisé pour expliciter la phase d'élection DR et BDR puis la phase de description des bases de données :

- R2200 a découvert en trame 68 que R1100 détenait le LSA 1.0.0.11. Puisque ce LSA est encore inconnu de R2200, il est demandé à l'aide d'un paquet « *LS Request* » en trame 74.

- De même, R1100 a découvert en trame 73 que R2200 détenait le LSA 1.0.0.22. Puisque ce LSA est encore inconnu de R1100, il est demandé à l'aide d'un paquet « *LS Request* » en trame 86.

- R2200 reçoit la réponse à sa requête en trame 87. La liste de demandes « *Link State Request List* » ne comportait qu'un seul élément, elle est maintenant vide, R2200 n'a plus d'autre requête à formuler. Ceci provoque l'évènement « *Loading done* » dans le diagramme d'états de la relation de voisinage. Pourtant, le

diagramme ne pourra passer à l'état « *Full* » qu'après être passé par l'état « *Loading* », ce qui ne se produira qu'à la trame 104. Notez que le paquet « *LS Request* » de la trame 74 pouvait embarquer bien plus qu'une seule demande. Quand c'est le cas, chaque LSA demandé l'est explicitement (voir Format des messages OSPF - Le paquet Demande d'état de lien).

- R1100 reçoit la réponse à sa requête en trame 97. La liste de demandes « *Link State Request List* » ne comportait qu'un seul item, elle est maintenant vide, R1100 n'a plus d'autre requête à formuler. Ceci provoque l'évènement « *Loading done* » dans le diagramme d'états de la relation de voisinage et par suite, le passage à l'état « *Full* ».

Le diagramme d'états de la relation de voisinage entretenu par R1100 pour sa relation avec R2200 est maintenant dans l'état « *Full* ». De même, le diagramme entretenu par R2200 pour sa relation avec R1100 est dans l'état « *Full* ». Les deux voisins R1100 et R2200 sont donc pleinement adjacents (des proches), leurs bases de données sont synchronisées. Il reste maintenant à OSPF la charge de maintenir les LSD synchronisées, c'est l'objet du processus d'inondation. Il lui faut également maintenir le degré de pertinence ou de confiance dans chacun des LSAs contenus dans la LSD, c'est l'objet du processus de rafraîchissement des LSAs. Ces deux processus étroitement imbriqués sont décrits à la section Processus d'inondation dans ce chapitre.

f. Maintenance

Les quelques commandes **show** ou **debug** suivantes peuvent aider à comprendre l'activité d'OSPF quand elle est liée aux problèmes de voisinage. Le contexte est le suivant : les trois routeurs R1200, R2100 et R2200 sont opérationnels depuis un certain temps, R2200 est DR, R2100 est BDR. La commande **debug ip ospf adj** a été entrée sur R2200. Puis on active R1100.

Repérez dans la capture précédente le passage de la relation de voisinage R1100 - R2200 par les différents états : 2WAY, EXSTART, EXCHANGE et FULL. Observez également l'échange de paquets DBD associés aux numéros de séquence. R1100 est esclave dans cet échange, l'adresse IP de l'interface concernée est 10.0.8.11, mais puisque cette adresse est aussi la plus petite des quatre adresses présentes sur LAN8, on peut déduire que R1100 aurait été esclave quelle que soit la relation de voisinage à construire sur ce réseau. Repérez les différents évènements « *Neighbor change* » qui interviennent dans le diagramme d'états de l'interface puis les évènements « *Negotiation done* », « *Exchange done* », « *Loading done* » qui interviennent dans le diagramme d'états de la relation de voisinage. Au final, **debug** fournit un résultat presque aussi détaillé que celui d'un analyseur de protocole.

```
R2200b#debug ip ospf adj
OSPF adjacency events debugging is on
01:34:11: OSPF: 2 Way Communication to 1.0.0.11 on FastEthernet0/1, state 2WAY
01:34:11: OSPF: Neighbor change Event on interface FastEthernet0/1
01:34:11: OSPF: DR/BDR election on FastEthernet0/1
01:34:11: OSPF: Elect BDR 1.0.0.21
01:34:11: OSPF: Elect DR 1.0.0.22
01:34:11:  DR: 1.0.0.22 (Id)   BDR: 1.0.0.21 (Id)
01:34:11: OSPF: Send DBD to 1.0.0.11 on FastEthernet0/1 seq 0x2499 opt 0x42 flag 0x7
len 32
01:34:11: OSPF: Rcv DBD from 1.0.0.11 on FastEthernet0/1 seq 0x2499 opt 0x42 flag 0x2
len 52  mtu 1500 state EXSTART
01:34:11: OSPF: NBR Negotiation Done. We are the MASTER
01:34:11: OSPF: Send DBD to 1.0.0.11 on FastEthernet0/1 seq 0x249A opt 0x42 flag 0x3
len 132
01:34:11: OSPF: Rcv DBD from 1.0.0.11 on FastEthernet0/1 seq 0x249A opt 0x42 flag 0x0
len 32  mtu 1500 state EXCHANGE
01:34:11: OSPF: Send DBD to 1.0.0.11 on FastEthernet0/1 seq 0x249B opt 0x42 flag 0x1
len 32
01:34:11: OSPF: Rcv DBD from 1.0.0.11 on FastEthernet0/1 seq 0x249B opt 0x42 flag 0x0
len 32 mtu 1500 state EXCHANGE
01:34:11: OSPF: Exchange Done with 1.0.0.11 on FastEthernet0/1
01:34:11: OSPF: Synchronized with 1.0.0.11 on FastEthernet0/1, state FULL
01:34:11: %OSPF-5-ADJCHG: Process 1, Nbr 1.0.0.11 on FastEthernet0/1 from LOADING
to FULL, Loading Done
```

En dehors du résultat de la commande **debug**, SYSLOG informe également de l'activité OSPF liée au voisinage. Le passage de la relation de R2200 avec son voisin R1100 à l'état FULL est salué par un message SYSLOG classé au niveau 5 « Notifications » (Evènement normal mais important) :

```
01:34:11: %OSPF-5-ADJCHG: Process 1, Nbr 1.0.0.11 on FastEthernet0/1 from LOADING
to FULL, Loading Done
```

Pour R2200, un des voisins (R1100) est arrivé à l'état « Full » et cet évènement le contraint à produire une nouvelle instance du LSA de réseau :

```
01:34:12: OSPF: Build network LSA for FastEthernet0/1, router ID 1.0.0.22
```

Après avoir fait cesser la commande **debug**, l'administrateur peut vérifier quelles sont les relations de voisinage entretenues par ce routeur à l'aide de la commande **show ip ospf neighbor**, commande des plus utiles car elle fournit une information concise. Par exemple sur R1100 :

```
R1100b#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address      Interface
1.0.0.12          1   2WAY/DROTHER    00:00:39    10.0.8.12    FastEthernet0/1
1.0.0.22          1   FULL/DR         00:00:39    10.0.8.22    FastEthernet0/1
1.0.0.21          1   FULL/BDR        00:00:35    10.0.8.21    FastEthernet0/1
```

6. Partage de l'AS en aires

a. Notion d'aire

En préambule, il a été dit qu'OSPF met à profit un concept d'aires afin de contenir ses exigences en ressources machine (mémoire et CPU), afin également de réduire autant que faire se peut le trafic d'acheminement. Pour mémoire, le trafic d'acheminement est le trafic inhérent au protocole de routage. La bande passante consommée pour ce trafic l'est au détriment de la bande passante utile.

Du point de vue OSPF, l'aire est un regroupement logique de routeurs OSPF et de liens entre routeurs. À l'aide des aires d'OSPF, il devient possible de subdiviser le domaine en sous-domaines. Un routeur dans une aire donnée peut ignorer les détails de topologie des aires voisines. Toutes proportions gardées, l'aire est à OSPF ce que la zone est à DNS, un fractionnement de l'information globale :

- Dans DNS, l'existence de la zone est réelle, physique puisqu'on peut lui associer une base de données, l'ensemble des zones juxtaposées constituant la base de données globale. À l'intérieur d'une zone, l'ensemble des serveurs de noms entretiennent la même copie de la base de données associée à la zone.

- Il en va de même dans une aire OSPF à laquelle il faut associer la base de données d'états de liens (LSD) pour l'aire. À l'intérieur de l'aire, chaque routeur entretient la même copie de la LSD.

L'analogie s'arrête là car la base de données d'une zone contient les informations que l'administrateur a bien voulu y placer alors que la base de données LSD est auto-construite par les routeurs à l'aide d'un processus d'inondation, chaque routeur apportant sa pierre (ses LSAs) à l'édifice (la LSD de l'aire).

Un routeur OSPF ne partage une LSD qu'avec les autres routeurs de la même aire. La taille de la LSD et donc l'impact sur la mémoire des routeurs sont évidemment plus réduits que si la LSD avait dû couvrir le système autonome dans son entier.

Une LSD plus réduite signifie également moins de LSAs à entretenir et des LSAs qu'il faut propager moins loin, avec à la clé, une consommation de ressources CPU revue à la baisse ainsi qu'un trafic d'acheminement moindre.

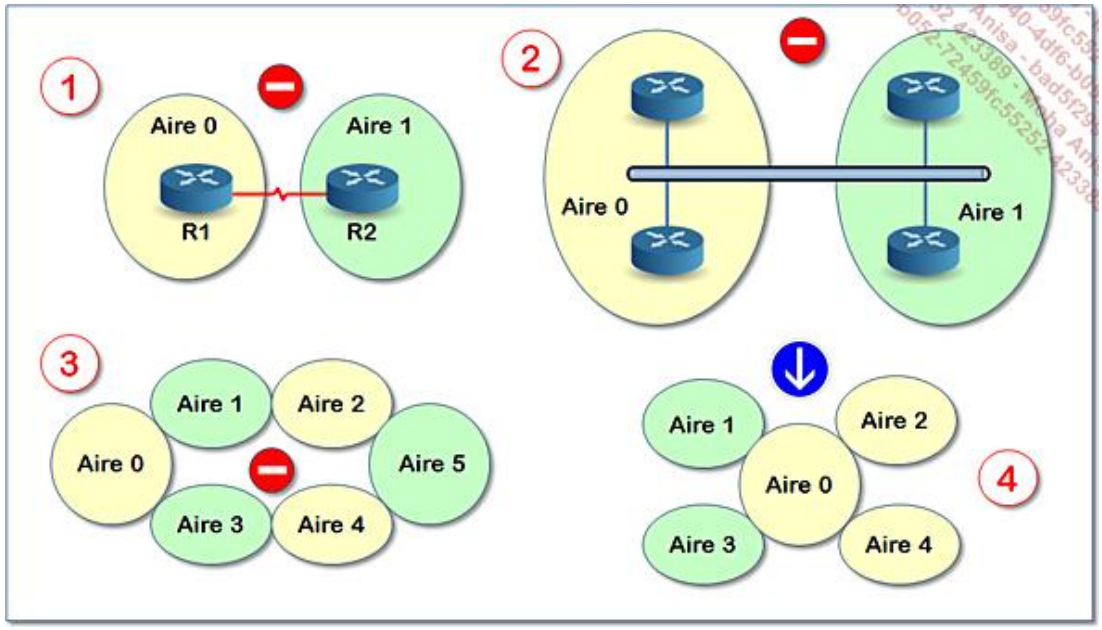
Définissons formellement l'aire :

- Un lien ou un réseau appartient à une aire et une seule. Les frontières d'aires sont par conséquent placées sur les routeurs et non sur les liens.

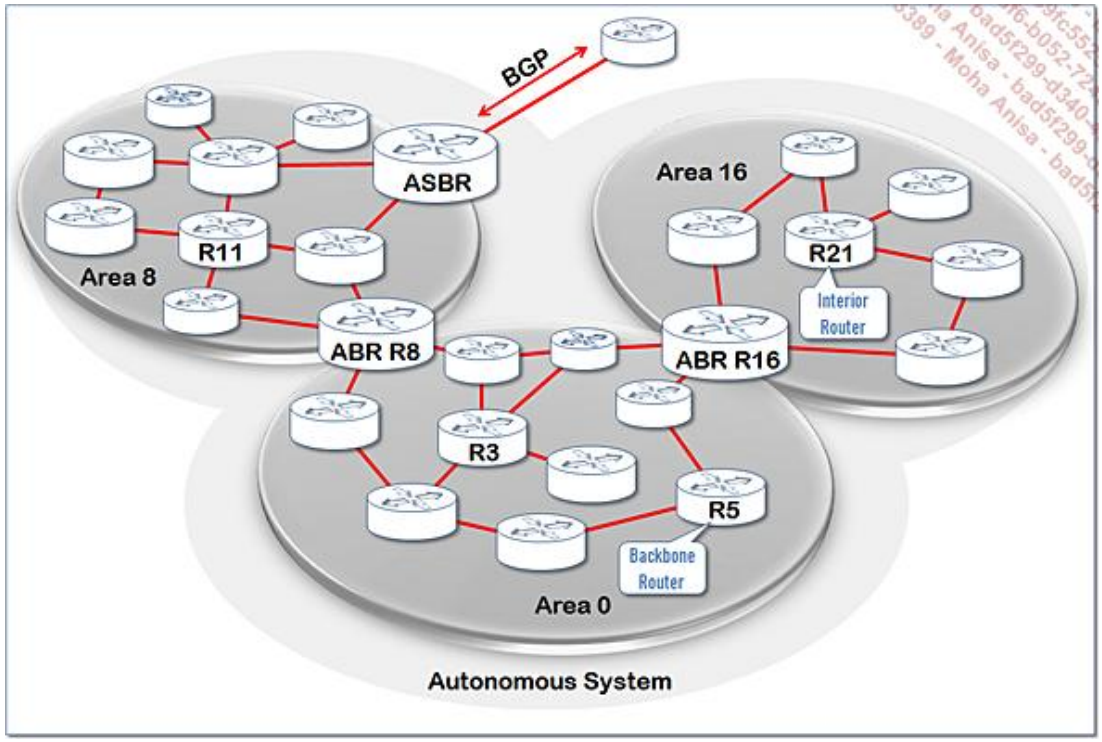
- Une aire est identifiée par un numéro exprimé sur 32 bits et totalement indépendant du plan d'adressage IP du réseau :

    - Les identifiants d'aire sont formatés comme des adresses IP. L'aire 0 est également l'aire 0.0.0.0. Tant que l'identifiant est en deçà de 256, les choses sont simples, « x » ou « 0.0.0.x » identifient une seule et même aire. Au-delà de 256 (mais est-ce utile ?), on préférera sans doute s'en tenir au format décimal pointé.

- Tous les routeurs frontières appartiennent au moins à une aire particulière, dite backbone ou aire 0. La figure suivante illustre ce qui est possible et ce qui ne l'est pas :

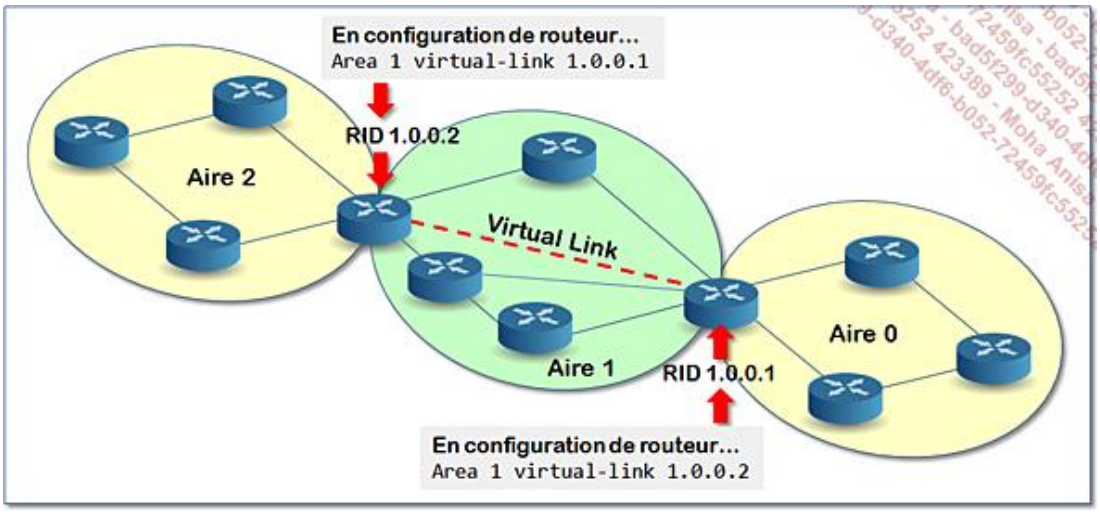


- Les aires d'OSPF doivent être construites selon une topologie hiérarchique autour de l'aire backbone, toujours présente. L'identifiant 0 (ou 0.0.0.0) est réservé à l'aire backbone (littéralement épine dorsale). Illustration :



Topologie hiérarchique d'un domaine OSPF

- Il est possible d'atténuer la contrainte précédente en établissant un lien virtuel vers l'aire backbone au travers d'une aire non backbone. Illustration :



En relation avec le découpage du système autonome en aires, il devient possible de classer le trafic en trois catégories :

- Le trafic « intra-aire » constitué par les paquets quand ils transitent entre routeurs sans quitter l'aire dont ils sont issus.
- Le trafic « inter-aire » constitué par les paquets dont les adresses source et destination n'appartiennent pas à la même aire.
- Le trafic externe quand des paquets doivent transiter entre le domaine OSPF et un autre système autonome.

L'aire backbone a la charge de résumer les destinations accessibles d'une aire et de publier ce résumé auprès des autres aires. En effet, le trafic « inter-aire » doit nécessairement transiter par l'aire backbone. Deux aires non-backbone ne peuvent échanger directement.

Comme nous aurons l'occasion de le détailler un peu plus tard en examinant les différents types de LSAs, l'échange d'informations de routage entre aires est essentiellement de type vecteur de distance. Vérifions-le en raisonnant sur la figure précédente légendée Topologie hiérarchique d'un domaine OSPF :

- Le protocole OSPF permet au routeur de bordure R8 de connaître l'ensemble des destinations dans l'aire 8 qui lui est directement connectée. R8 annonce ces destinations dans l'aire 0 à l'aide de LSAs de type résumés de réseau.
- Le routeur R16 reçoit ces LSAs résumés de réseau grâce au processus d'inondation.
- Chaque LSA résumé de réseau annoncé par R8 contient une destination et un coût. R16 place la destination dans sa table de routage via R8 en ayant soin de mettre le coût à jour. Le coût calculé par R16 est la somme du coût reçu et du coût pour atteindre R8. Puis R16 génère ses propres LSAs résumés de réseau et les inonde vers l'aire 16.

C'est bien la description d'un algorithme type vecteur de distance. Ceci explique également pourquoi les aires OSPF doivent être construites en étoile autour de l'aire backbone, on évite ainsi l'écueil des boucles de routage que pourrait amener un tel algorithme.

Classiquement, les architectes réseau en charge de la planification et du déploiement d'OSPF mémorisent des ordres de grandeur quant à la taille maximale d'une aire et le font en s'imposant un nombre de routeurs maximal, par exemple 50. Il n'est pas certain que cette démarche soit la plus pertinente car beaucoup d'autres facteurs influent sur la consommation de ressources mémoire et CPU du processus OSPF et notamment le nombre de liens de l'aire ou le nombre de LSA résumé issus des autres aires, liste non exhaustive. Ceci explique que des aires à 30 routeurs puissent être parfois plus chargées que des aires à 300 routeurs. L'architecte réseau ne peut s'absoudre d'une surveillance des ressources consommées après le déploiement. Par ailleurs, rien n'interdit de déployer OSPF dans un domaine unique quand l'inter-réseau est de taille raisonnable. Diviser un domaine en aires est une possibilité, pas une obligation.

b. Classifions les routeurs

Observez la figure légendée Topologie hiérarchique d'un domaine OSPF et distinguez les routeurs suivants :

- Le routeur interne : toutes ses interfaces appartiennent à une seule et même aire. Un routeur interne n'entretient qu'une seule LSD.

- Le routeur de bordure ABR (*Area Border Router*) : connecte une aire ou davantage à l'aire backbone, c'est en quelque sorte la passerelle pour le trafic inter-aire. Le routeur ABR a au moins une interface appartenant à l'aire backbone. La vie d'un ABR est évidemment plus compliquée que celle d'un routeur interne car il doit entretenir autant de LSD que d'aires connectées. Ces routeurs justifient probablement un effort quant à la quantité de mémoire embarquée ou la puissance CPU. C'est le routeur ABR qui résume la topologie de l'aire ou des aires connectées puis publie ces résumés vers l'aire backbone à l'aide de LSAs résumé (patientez).

- Le routeur backbone : il a au moins une interface dans l'aire backbone. Un routeur ABR est nécessairement un routeur backbone mais l'assertion inverse n'est pas vraie. De même, un routeur interne dont toutes les interfaces appartiennent à l'aire backbone est un routeur backbone.

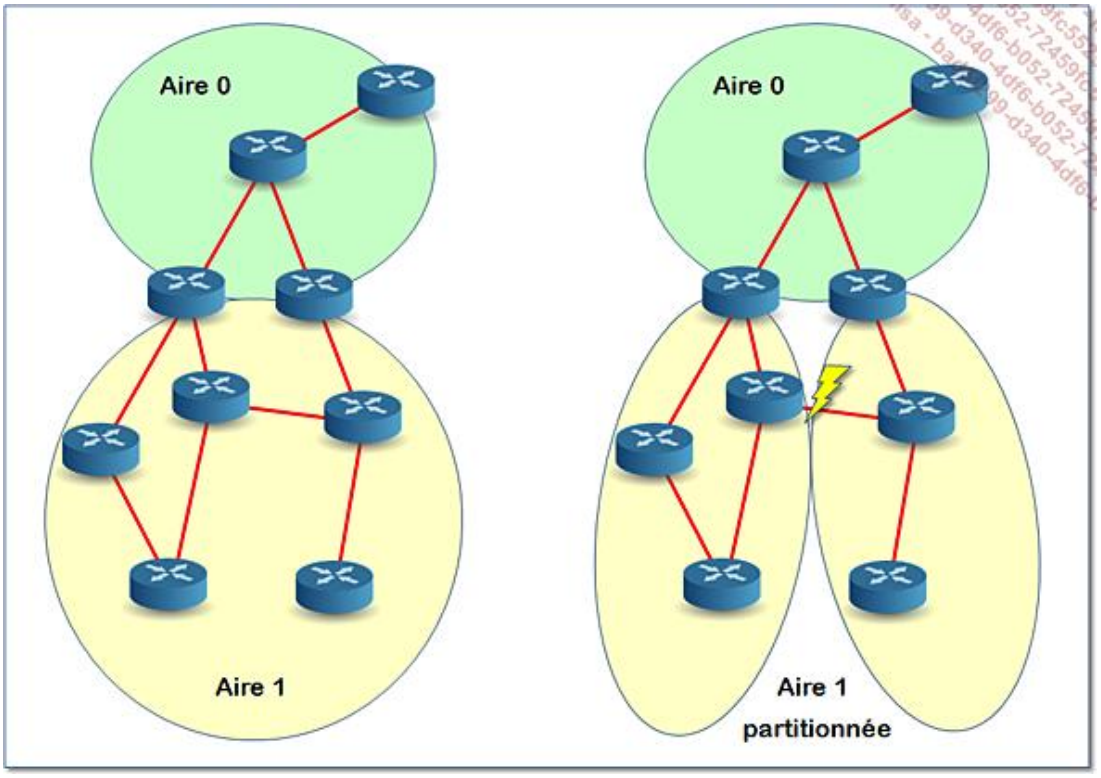
- Le routeur frontière au système autonome ASBR (*Autonomous System Boundary Router*) : l'ASBR est au système autonome ce que l'ABR est à l'aire, une passerelle vers les destinations externes au système autonome. L'ASBR apprend des routes externes à l'aide de protocoles de routage tels qu'EIGRP ou BGP puis les injecte dans le domaine OSPF. L'ASBR peut être indifféremment un routeur interne, un routeur ABR ou un routeur backbone.

c. Continuité de service

Nous sommes depuis un moment hors cadre CCNA, le lecteur intéressé exclusivement par la certification peut aller directement à la description du processus d'inondation.

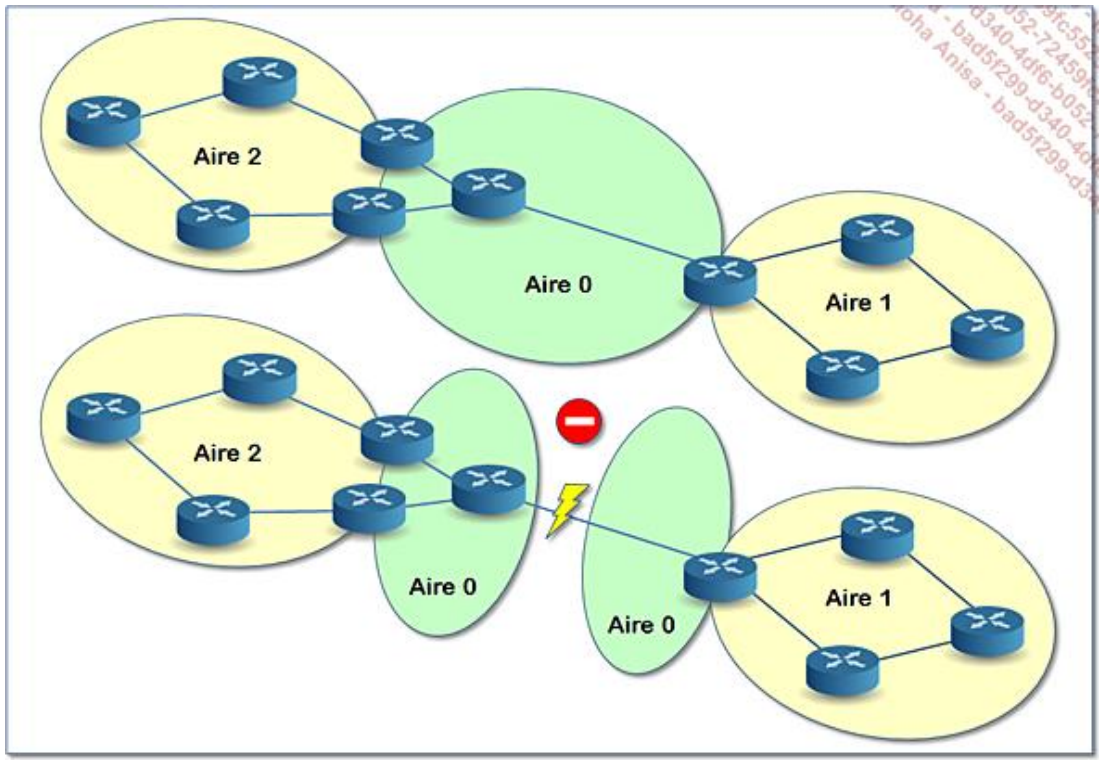
L'architecte réseau doit garder à l'esprit des règles de bon sens et assurer un maillage suffisant à même de contourner un lien défaillant.

Exemple 1 :

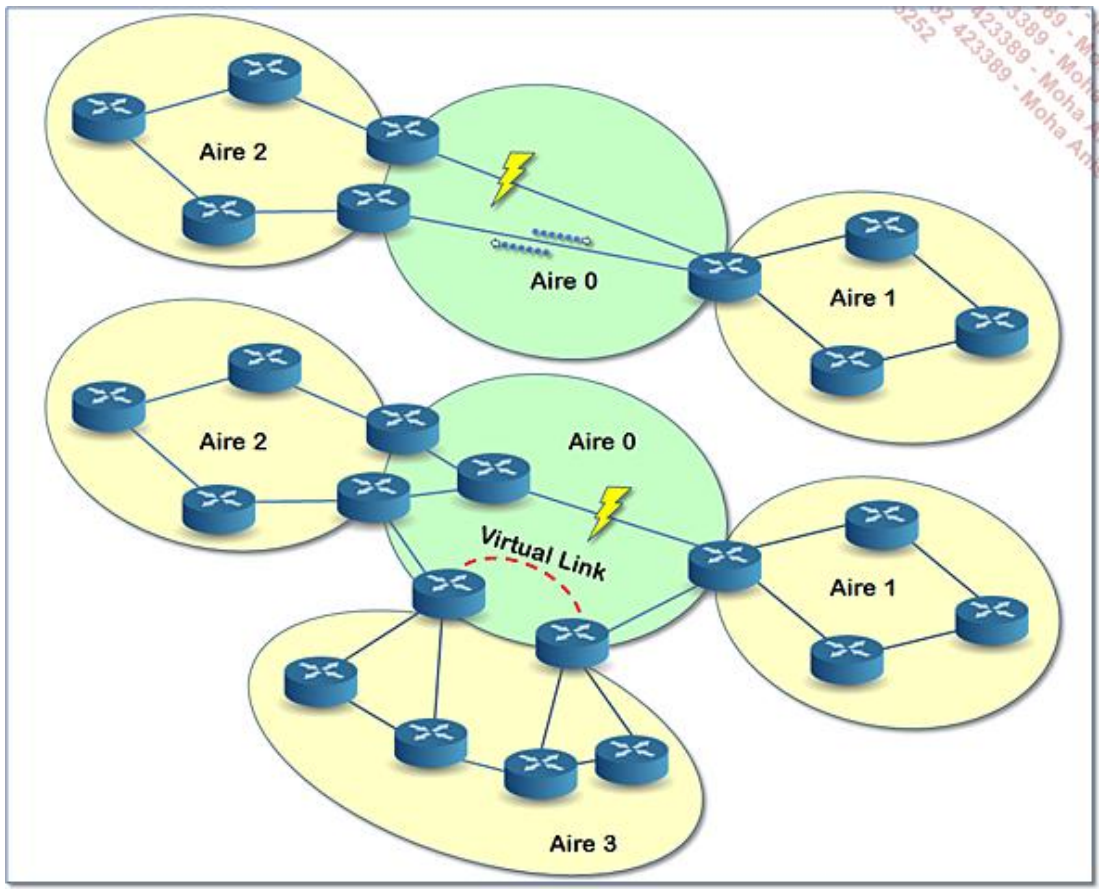


L'aire 1 est sans doute insuffisamment maillée et une défaillance du lien telle qu'illustrée peut conduire à un partitionnement de l'aire. Fort heureusement dans ce cas précis, chaque partition de l'aire 1 reste connectée à un ABR. L'aire backbone considère alors les deux partitions comme deux aires distinctes. Le trafic intra-aire d'une partition à l'autre devient un trafic inter-aire qui transite par l'aire backbone, le service est maintenu.

Exemple 2



Les routeurs de l'aire backbone sont insuffisamment maillés et un lien défaillant provoque le partitionnement de l'aire backbone. C'est évidemment beaucoup plus grave, tout se passe comme s'il existait désormais deux domaines OSPF. Prévenir cet accident est à nouveau un travail d'architecte, voici deux solutions possibles :



La première solution consiste à mailler davantage les routeurs de l'aire backbone. Dans l'illustration précédente, la défaillance d'un lien de l'aire 0 ne peut causer une interruption de service. La seconde solution consiste à rétablir la connectivité en établissant un lien virtuel au travers d'une aire non backbone.

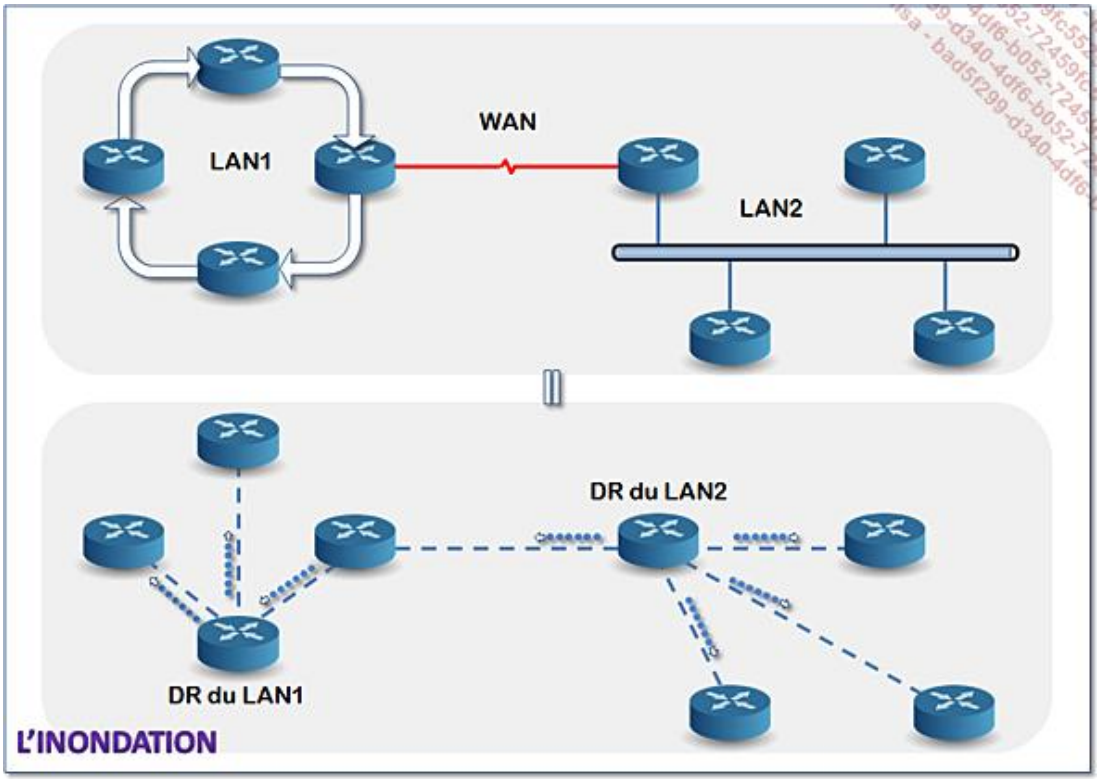
7. Processus d'inondation

a. Concepts généraux

Il faut distinguer le trafic IP utile du trafic d'acheminement. Chaque routeur OSPF effectue son calcul de l'arbre SPF et en déduit des routes qu'il place en table de routage. Un trafic utile emprunte l'une de ces routes pour progresser vers sa destination. Le trafic d'acheminement consiste en tous les messages OSPF utilisés pour entretenir le protocole. À ce stade du chapitre, nous avons mis à profit les types de paquets OSPF suivants :

- Les messages Hello pour découvrir les voisins puis maintenir les relations de voisinage en vie.
- Les paquets DBD pour décrire le contenu des LSD.
- Les paquets « *LS Request* », « *LS Update* » pour synchroniser les bases de données LSDs.

Ces aspects du protocole ont permis de tisser un ensemble de relations de proximité qui en final construisent un réseau logique très différent du réseau physique et qui sert de support au trafic d'acheminement destiné à maintenir l'ensemble des LSDs d'une aire identiques :



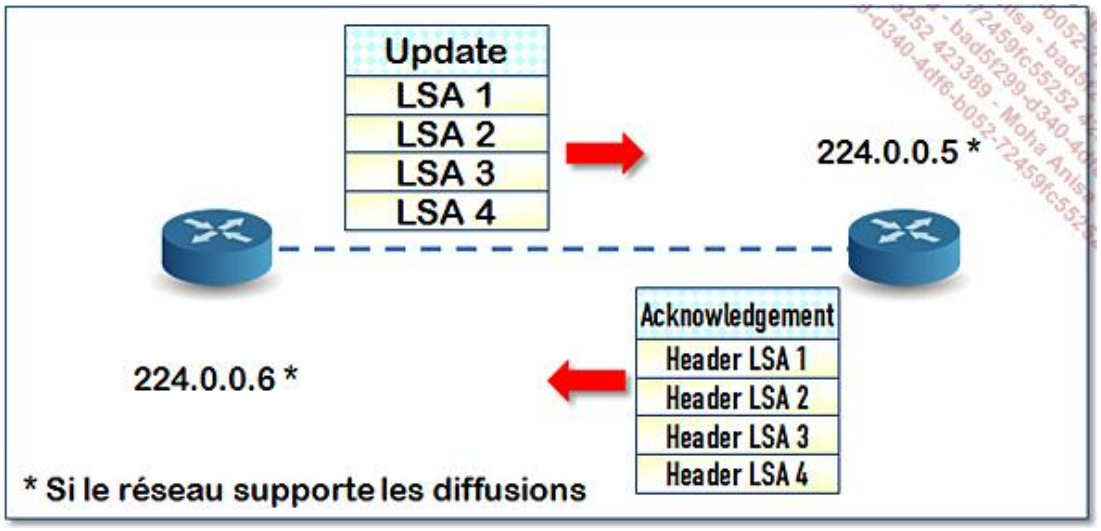
Dans cet exemple, deux réseaux physiques LAN, un Token Ring et un Ethernet, sont interconnectés par une liaison point à point WAN. OSPF a procédé à l'élection d'un DR sur chacun des réseaux à diffusion puis a établi une relation de proximité entre les deux routeurs de part et d'autre du lien WAN, ainsi qu'entre le DR de chaque réseau LAN avec les autres routeurs du réseau. La vue logique est faite d'un ensemble de relations de proximités, c'est-à-dire un ensemble de relations logiques point à point.

Immédiatement après l'établissement d'une relation de proximité, les deux LSD de part et d'autre de la relation sont identiques, mais elles ne le resteraient pas longtemps sans un processus destiné à maintenir en temps réel la cohérence de l'ensemble des LSD sur l'aire. La LSD contient l'ensemble des LSAs générés par les routeurs de l'aire, c'est donc une base de données topologique. Tout changement dans la topologie de l'aire (un lien qui monte ou qui tombe, un routeur qui s'active ou qui disparaît...) se traduit par un changement dans au moins un LSA.

Le processus d'inondation est le processus prévu par OSPF pour maintenir des bases de données d'états de lien identiques sur l'ensemble des routeurs d'une aire.

Le processus d'inondation consiste à faire en sorte que tout LSA nouveau ou modifié soit diffusé à l'ensemble des routeurs d'une aire. Il utilise pour ce faire les paquets OSPF de type :

- « *Link State Update* » (type 4) ;
- « *Link State Acknowledgement* » (type 5).



Un paquet « *Link State Update* » peut porter plusieurs LSAs distincts. Attention à bien comprendre ce point essentiel : les LSA sont un peu aux paquets « *LS Update* » ce que sont les datagrammes IP pour les trames Ethernet :

- Pour mémoire, la trame reste locale c'est-à-dire échangée entre deux interfaces directement connectées. Seul le datagramme transite dans le réseau.
- On peut faire l'analogie avec les LSAs encapsulés dans le « *LS update* ». Le paquet « *LS update* » comme tout paquet OSPF est généré avec une valeur TTL fixée à 1. On est ainsi assuré que le paquet ne puisse pas effectuer plus d'un saut. Seul le LSA transite dans le réseau.

OSPF utilise les paquets « *LS Update* » dans deux circonstances : lorsqu'il faut répondre à un paquet « *LS Request* » ainsi que dans le processus d'inondation des LSAs. Dans ce cas précis, souvenons-nous qu'un paquet OSPF n'a jamais besoin d'être acheminé au-delà du réseau dont il est issu. C'est ce constat qui a permis d'ajuster la valeur TTL des paquets OSPF à 1. La conséquence est que parmi les LSAs reçus, un routeur voisin doit ré-encapsuler les LSAs appropriés dans de nouveaux paquets « *LS Update* » pour que le processus d'inondation puisse progresser.

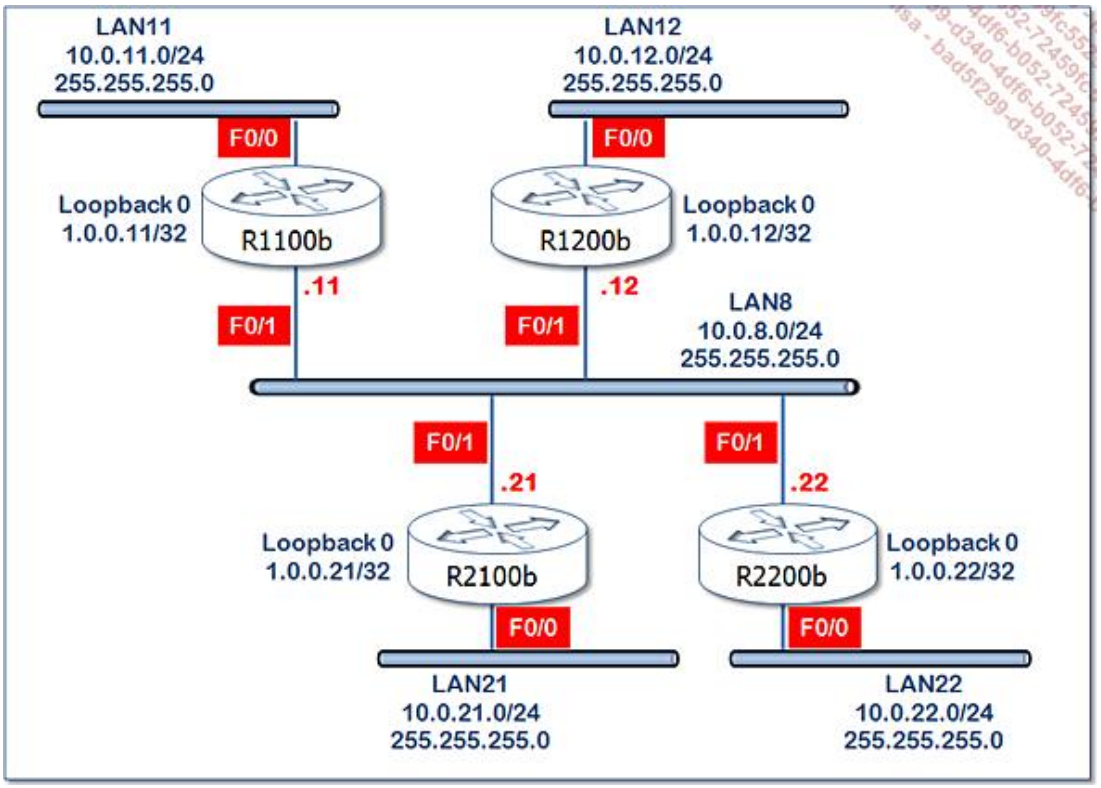
Pour faire une analogie encore plus osée, les LSAs sont des auto-stoppeurs et le « *LS Update* » une vaste voiture break conduite par un chauffeur compréhensif. La voiture a ramassé des auto-stoppeurs de toutes origines et de tous âges dont le seul but est d'aller partout. Mais la voiture ne dépasse pas le nœud routier suivant. Arrivé à ce nœud, tout le monde débarque et les auto-stoppeurs cherchent un autre chauffeur compréhensif.

L'adresse de destination d'un paquet « *LS update* » diffère selon le type de réseau :

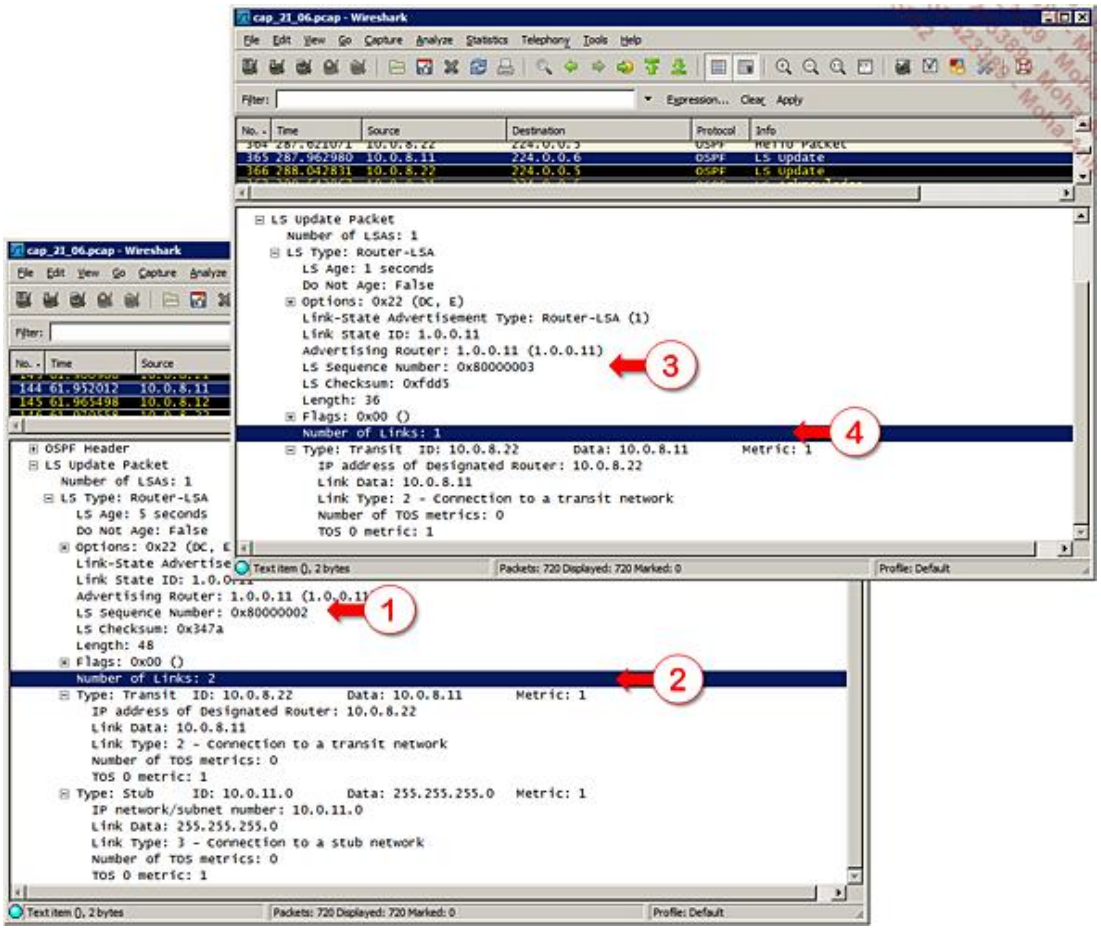
- Réseau point à point : les paquets « *LS Update* » sont envoyés vers l'adresse de multicast 224.0.0.5 (qui signifie tous les routeurs OSPF, « *AllSPFRouters* »).
- Réseau point à multipoint : les paquets « *LS Update* » sont envoyés vers l'adresse unicast du proche.
- Réseau à diffusion et réseau en mode NBMA : détaillés ci-après.

b. Inondation sur un réseau à diffusion

Sur les réseaux à diffusion, les « *DRothers* » forment une relation de proximité avec le DR et le BDR. Un paquet « *LS update* » issu d'un « *DRother* » est donc envoyé vers l'adresse multicast 224.0.0.6 (qui signifie les deux routeurs DR et BDR, « *AllDRouters* »). En retour, le DR émet le ou les LSAs nouveaux ou mis à jour dans un paquet « *LS Update* » destiné à tous les routeurs adjacents sur ce réseau, il le fait en utilisant l'adresse multicast 224.0.0.5. Le BDR ne participe pas activement au processus d'inondation mais il se tient prêt en profitant des « *LS Update* » reçus pour mettre à jour sa LSD. Illustrons à l'aide de cet exemple :

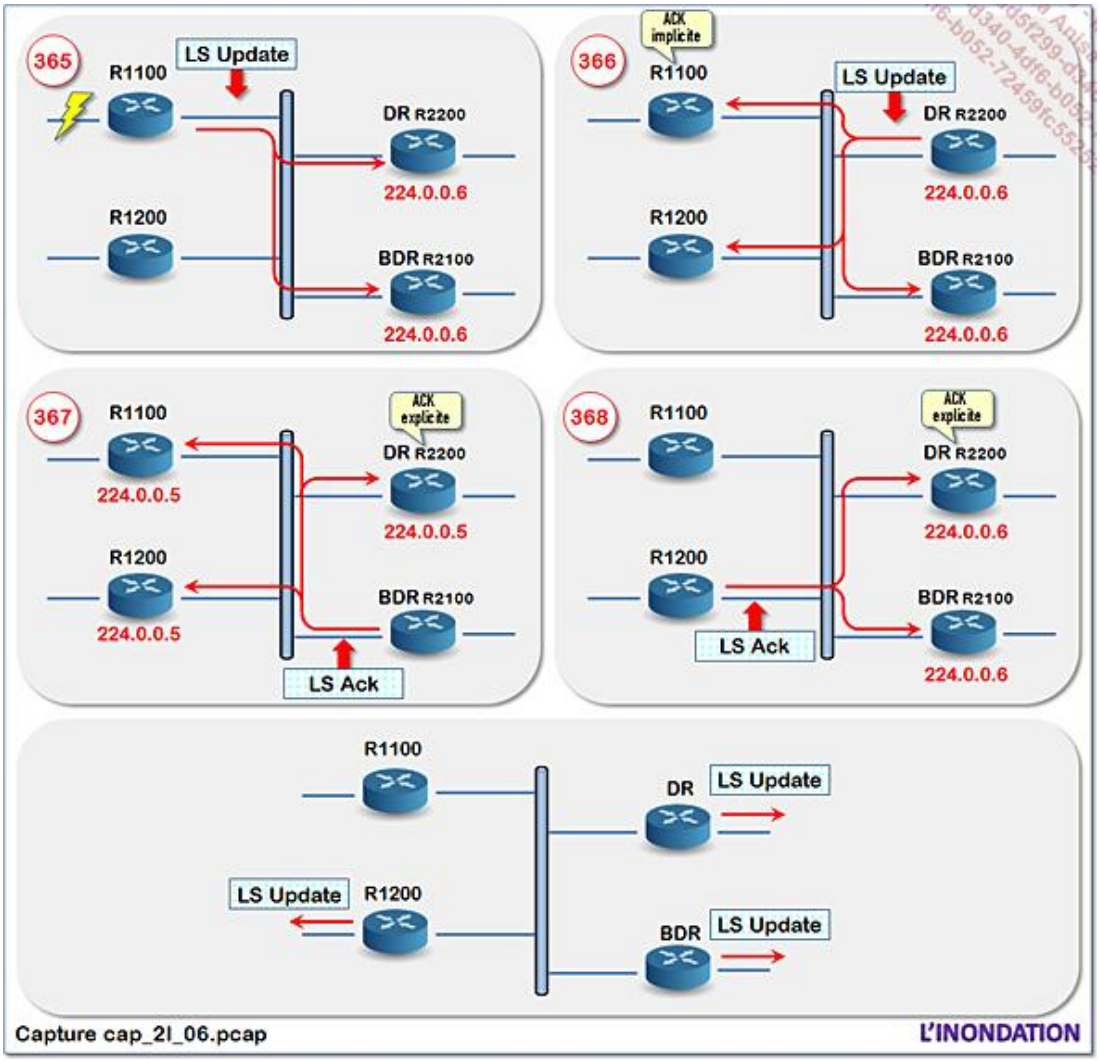


La capture correspondante cap_2I_06.pcap est disponible en téléchargement sur le site ENI. Le contexte est le suivant : le réseau est opérationnel depuis un temps suffisant. Sur LAN8, R2200 est DR, R2100 est BDR. L'interface F0/0 de R1100 tombe ce qui contraint R1100 à générer un nouveau LSA. Comparez le LSA avant l'incident et le LSA généré après que l'interface soit devenue inactive :



L'instance du LSA de routeur identifiée par « *Link State ID* » = 1.0.0.11, « *LS seq Number* » = 0x80000002, « *LS checksum* » = 0x347a, générée par R1100 portait deux liens avant l'incident. Le RFC 2328 inventorie dix

circonstances qui provoquent la génération d'une nouvelle instance de LSA. Le changement d'état d'une interface fait partie de ces dix évènements. R1100 génère donc une nouvelle instance de son LSA de routeur identifiée par « *Link State ID* » = 1.0.0.11, « *LS seq Number* » = 0x80000003, « *LS checksum* » = 0xfdd5 et diffuse cette instance dans un paquet « *LS Update* » vers l'adresse multicast 224.0.0.6 en trame 365. Cette instance ne porte plus qu'un seul lien vers le réseau LAN8.



c. Les acquittements

OSPF ne peut fournir des résultats de routage cohérents que si les LSDs des différents routeurs d'une aire sont parfaitement identiques. Puisque les mises à jour des LSAs qui composent la LSD sont propagées à l'aide du processus d'inondation, il est nécessaire que ce processus soit rendu parfaitement fiable. Un routeur qui transmet un ou plusieurs LSAs doit obtenir l'assurance que ces LSAs ont été bien reçus et un routeur qui reçoit des LSAs doit avoir la certitude que ces LSAs sont intègres. L'ouvrage Cisco - Notions de base sur les réseaux dans la collection Certifications aux Editions ENI a donné l'occasion d'étudier les protocoles de couche Transport et la fiabilité apportée par TCP. Mais OSPF s'encapsule directement dans IP, il faut se passer des garanties RIS (Remise, Intégrité, Séquencement) de TCP. Si OSPF a besoin de fiabilité, il doit se fabriquer cette fiabilité « sur mesures ». Pour ce faire, les mécanismes mis en œuvre sont :

1) Une somme de contrôle est ajoutée dans l'en-tête du LSA et porte sur l'ensemble du LSA à l'exclusion du champ âge (patientez).

2) Chaque LSA fait l'objet d'un acquittement individuel. Un routeur qui doit acquitter un LSA peut le faire de façon explicite ou implicite (patientez).

3) Un LSA est dotée dès sa naissance d'une espérance de vie. Si aucun évènement ne cause sa régénération, alors il meurt de mort naturelle à l'expiration de son espérance de vie et une nouvelle instance est générée (patientez).

- Acquittement implicite : imaginons deux routeurs A et B. Le routeur A émet un ou plusieurs LSAs nouveaux ou mis à jour vers le routeur B. Il se trouve que le routeur B qui reçoit ces LSAs doit à son tour émettre un ou plusieurs LSAs vers A. Pourquoi ne pas profiter de ce train en partance pour y glisser les acquittements : « Je veux te dire blablabla et j'en profite pour te dire que j'ai bien reçu tes LSAs X, Y et Z ». Oui mais voilà, un

paquet « LS Update » ne peut que porter des LSAs. Comment y insérer un ou plusieurs acquittements ?... En répétant en écho les LSAs reçus !

- Acquittement explicite : les acquittements sont transmis dans des paquets « *LS Acknowledgement* ». Un paquet « *LS Ack* » peut regrouper plusieurs acquittements de LSAs. Le paquet « *LS Ack* » n'embarque que les en-têtes des LSAs qu'il faut acquitter.

Revenons au contexte précédent qui nous offre l'opportunité de vérifier la coexistence des deux types d'acquittements :

- Le DR a reçu le LSA de routeur 1.0.0.11 et doit à son tour inonder ce LSA sur tous ses proches. Il le fait en encapsulant le LSA dans un paquet « *LS Update* » émis cette fois vers l'adresse multicast 224.0.0.5, trame 366.

- R1100 reçoit en écho l'instance de LSA qu'il vient d'émettre. C'est un acquittement implicite.

- Sans participer au processus d'inondation, le BDR doit au moins acquitter le LSA reçu du DR, il le fait à l'aide d'un « *LS Ack* » en trame 367, c'est un acquittement explicite.

- R1200 doit acquitter le LSA reçu du DR, il le fait à l'aide d'un « *LS Ack* » en trame 368, c'est un acquittement explicite.

- Le cas échéant, chacun des routeurs parmi R1200, R2100 et R2200 peut faire progresser la nouvelle instance de LSA en l'encapsulant dans un paquet « *LS Update* » puis en émettant ce paquet sur les interfaces appropriées.

Par ailleurs, toute nouvelle instance de LSA envoyée par un routeur vers un proche est également copiée dans la « Link State Retransmission List », liste qui appartient à la structure de données de l'interface. Dans le même temps, le processus OSPF arme un temporisateur dont l'échéance est réglée à « *RxmtInterval* » secondes. À moins qu'entre temps, l'adjacence ne soit détruite (absence de message Hello reçu pendant « *RouterDeadInterval* » et retour de la relation à l'état « Down ») le processus OSPF retransmet à nouveau un LSA qui n'aurait pas été acquitté quand le temporisateur associé arrive à échéance. Le LSA est purgé de la liste une fois l'acquittement reçu. Que la première émission de l'instance de LSA ait été diffusée vers une adresse multicast ou pas, un LSA retransmis l'est toujours dans un paquet « *LS Update* » émis vers l'adresse unicast du proche.

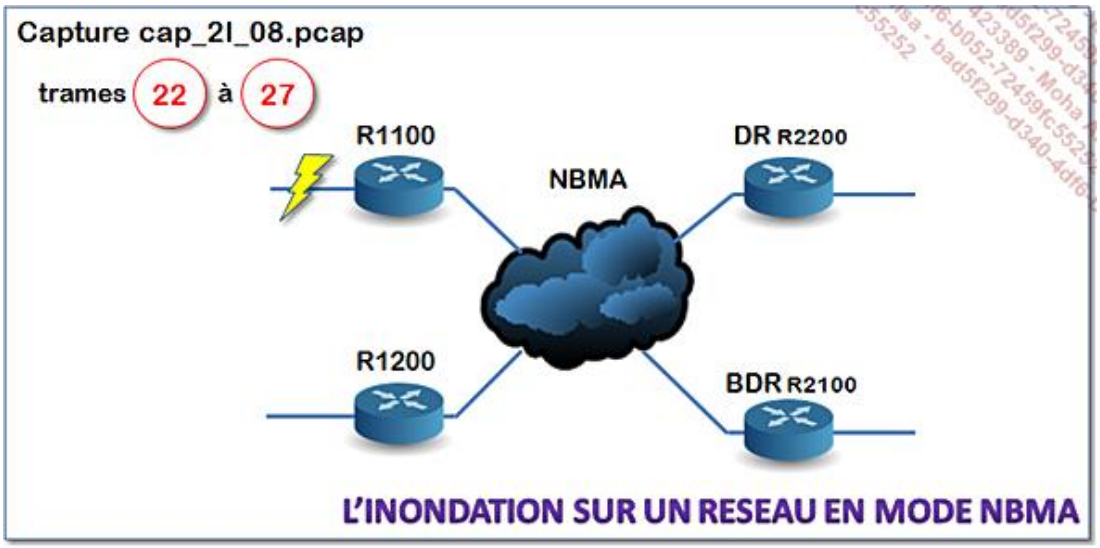
Pour être complet, signalons cette faculté qui consiste à différer un acquittement de façon à augmenter le nombre de LSAs acquittés à l'aide d'un seul paquet « *LS Ack* ». Le délai d'attente ne peut excéder « *RxmtInterval* » sous peine d'entraîner d'inutiles retransmissions. Un routeur ne peut recourir systématiquement à l'acquittement différé, quelques cas requièrent un acquittement immédiat :

- Le routeur reçoit à nouveau une même instance de LSA issue du même proche. On parle alors de LSA dupliqué et il laisse penser que probablement, l'acquittement n'a pas été reçu. Un LSA dupliqué justifie un acquittement immédiat.

- L'âge du LSA est « *MaxAge* » et la LSD du routeur qui reçoit le LSA ne contient pas d'instance de ce LSA.

d. Inondation sur un réseau en mode NBMA

Modifions le contexte en remplaçant LAN8 de l'exemple précédent par un réseau sans diffusion, la simulation a été réalisée sous GNS3 à l'aide d'un commutateur Frame-Relay, les routeurs sont configurés en mode « non-broadcast » (patientez) :



La capture cap_2I_08.pcap, disponible en téléchargement, a été réalisée sur l'interface « *serial* » du DR. On provoque à nouveau la tombée de l'interface F0/0 sur R1100, ce qui contraint R1100 à générer une nouvelle instance de son LSA de routeur puis à l'émettre vers le DR en trame 22. Mais cette fois, tous les échanges s'opèrent à l'aide de paquets unicast. Le DR réagit de la même façon que sur un réseau à diffusion, c'est-à-dire en inondant le LSA mis à jour à l'aide d'un paquet « LS Update ». Mais ce paquet est répliqué autant de fois qu'il y a de proches à informer, dans les trames 23, 24 et 25. Rien ne change pour les acquittements, le LSA issu de R1100 fait toujours l'objet d'un acquittement implicite en trame 25. Le LSA issu du DR est acquitté explicitement par le BDR et R1200.

e. Choix de l'instance convenable

Tout LSA débute par un en-tête de 20 octets qui contient notamment les informations suffisantes pour identifier le LSA sans équivoque, ce sont les trois champs « *LS type* », « *Link State ID* » et « *Advertising Router* ». De plus, puisque plusieurs instances d'un même LSA peuvent coexister dans le domaine d'acheminement, il est nécessaire de pouvoir déterminer l'instance la plus récente. OSPF utilise pour ce faire les trois champs « *LS age* », « *LS sequence number* » et « *LS checksum* » :

- L'âge d'un LSA « *LS age* » est exprimé en secondes sur 16 bits sans toutefois pouvoir dépasser la valeur 3600 (1 heure) appelée valeur « *MaxAge* ». Quand un routeur génère un nouveau LSA, il lui attribue l'âge 0. Au fur et à mesure que le LSA progresse dans l'aire pendant le processus d'inondation, chaque routeur traversé (en réalité chaque interface de sortie, c'est-à-dire l'interface qui émet le LSA ré encapsulé dans un nouveau paquet « *LS Update* ») incrémente l'âge de « *InfTransDelay* » secondes. L'IOS autorise la modification de cette valeur à l'aide de la commande **ip ospf transmit-delay** en configuration d'interface, la valeur par défaut est une seconde. Une instance de LSA qui a atteint la limite d'âge est inondée puis supprimée de la LSD.

- Le numéro de séquence du LSA est exprimé en binaire signé sur 32 bits. Quand un routeur génère un LSA, il lui attribué le premier numéro 0x80000001 (en décimal -2147483647). Puis à chaque fois que le routeur produit une nouvelle instance de ce LSA, il incrémente le numéro de séquence de 1. La valeur maximale est 0x7FFFFFFF (soit en décimal +2147483647). Que doit faire un routeur quand il doit générer une nouvelle instance alors que le numéro de séquence en cours est 0x7FFFFFFF ? Remarquons d'abord que c'est un cas très improbable puisqu'en imaginant par exemple qu'il faille générer une nouvelle instance par seconde (tout aussi improbable), plus de 136 années seraient nécessaires pour atteindre la valeur maximale. La procédure prévoit d'effacer dans un premier temps toutes les occurrences du LSA à numéro de séquence maximal en inondant le LSA dans lequel le routeur a pris soin de régler la valeur Age à « *MaxAge* ». Une fois l'acquittement de ce LSA (prématurément vieilli) reçu, le routeur peut alors inonder un LSA nouveau avec le numéro de séquence réglé au numéro de séquence initial, soit 0x80000001.

- La somme de contrôle « *LS checksum* » est calculée selon l'algorithme de Fletcher décrit dans le RFC 1146. Pour mémoire, l'algorithme classique consiste à additionner simplement tous les mots de 16 bits contenus dans le message dont il faut vérifier l'intégrité. Un tel algorithme est incapable de détecter par exemple l'insertion ou la suppression d'un octet à 0, ce que sait faire en revanche l'algorithme de Fletcher. La somme de contrôle porte sur l'ensemble du LSA à l'exclusion du champ âge. En effet, inclure ce champ aurait contraint à recalculer la somme à chaque incrémentation de l'âge. Bien sûr, un LSA vieillit également quand il reste dans la LSD, l'IOS maintient son âge à jour.

Quand un routeur reçoit plusieurs instances d'un même LSA, il détermine la plus récente en déroulant l'algorithme suivant :

1. Le processus compare les numéros de séquence, le numéro de séquence le plus élevé est le plus récent.

2. Si les numéros de séquence sont identiques, le processus compare les sommes de contrôle. Le LSA dont la somme de contrôle est la plus élevée est le LSA le plus récent (c'est arbitraire mais il fallait faire un choix).

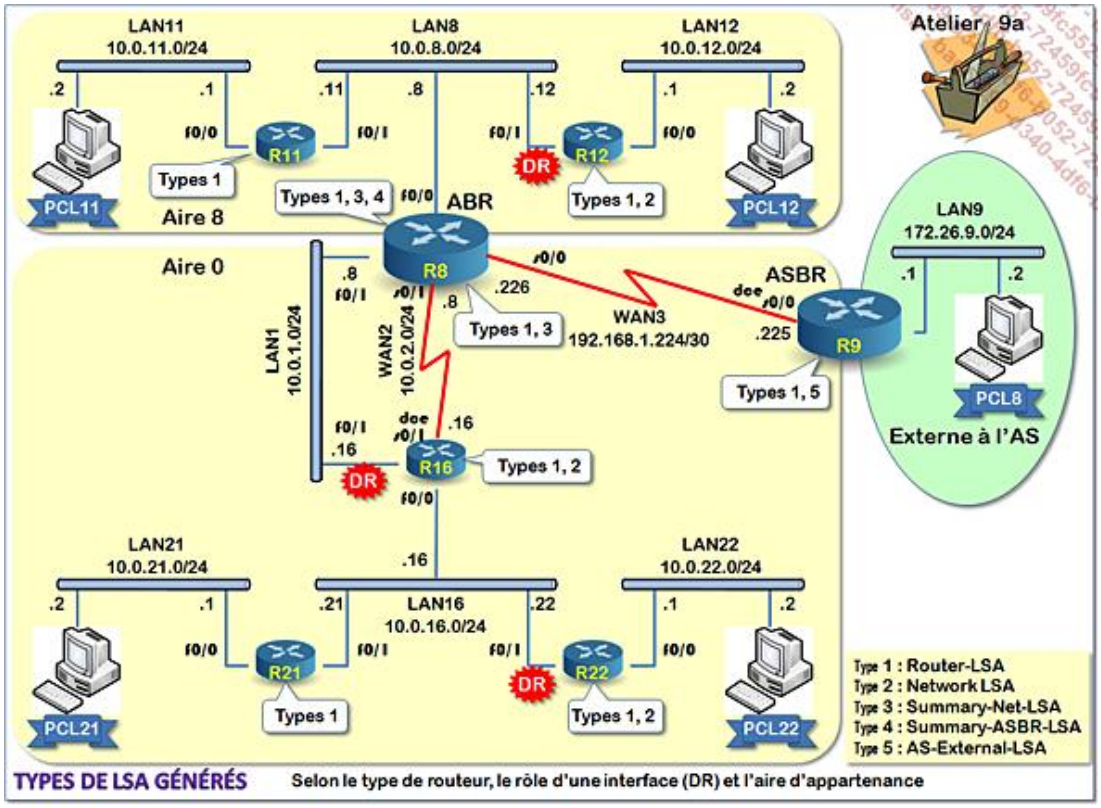
3. En cas d'égalité des sommes de contrôle, le processus compare l'âge des différentes instances. Quand l'une des instances est dotée d'un âge égal à « *MaxAge* », elle est considérée comme la plus récente. Dans le cas contraire, si les âges diffèrent de plus de 15 minutes (« MaxAgeDiff »), le LSA qui est doté de l'âge le moins élevé est le plus récent.

4. Si aucune des conditions précédentes n'est réalisée, alors l'algorithme considère les instances de LSA comme étant identiques.

8. La base de données d'états de liens ou LSD

Encore une fois, c'est bien d'une base de données topologiques qu'il s'agit. La LSD (*Link State Database*) contient l'ensemble des LSAs générés ou collectés par un routeur. De la fiabilité de sa réplication sur l'ensemble des routeurs d'une aire dépend la fiabilité des routes ajoutées en table de routage.

Comme à notre habitude maintenant, nous fonderons les explications qui suivent sur une mise en situation que le lecteur sera invité à reproduire dans l'atelier de ce chapitre, toujours sous GNS3, les routeurs choisis sont des 2621, l'image IOS est « ik9s-mz.122-40a.bin » choisie pour sa compacité, 48 Mo de RAM lui suffisent :



Le domaine a été partagé en deux aires 0 et 8, ce de façon à avoir au moins un routeur ABR dans la topologie car seul ce routeur génère des LSAs de type résumé (types 3 et 4, patientez). De même, le domaine OSPF n'englobe pas LAN9 de façon à disposer d'un routeur ASBR, le seul à générer des LSAs externes à l'AS (Type 5, patientez), de façon également à offrir un cas où il est nécessaire de redistribuer des routes vers le protocole OSPF. Sur chacun des routeurs a été configurée l'interface de loopback 0, l'adresse IP attribuée est 1.0.0.X quand le routeur est nommé RX. C'est une façon pour l'administrateur de maîtriser le RID attribué à chaque routeur (le RID de R11 est 1.0.0.11...) et par suite de prévoir le résultat de l'élection des routeurs désignés.

Les réseaux LAN1, LAN8 et LAN16 sont tous trois à diffusion et ont été le théâtre d'une élection DR, BDR. Les interfaces élues DR sont f0/1 de R12 pour LAN8, f0/1 de R16 pour LAN1 et f0/1 de R22 pour LAN16. Seuls les routeurs dont au moins une interface est élue DR génèrent des LSAs de type réseau (type 2, patientez). La figure précédente inventorie les types de LSAs générés par chacun des routeurs.

Il est possible d'observer le contenu de la LSD à l'aide d'une commande **show ip ospf database**. L'information proposée est condensée, seuls les en-têtes de LSAs sont affichés. Observez la capture suivante. Sans avoir la topologie sous les yeux, il est possible de déduire que le domaine OSPF a été divisé en deux aires ou davantage et que le routeur R8 est un ABR. En effet, une partie des LSAs appartient à l'aire 0, l'autre partie appartient à l'aire 8. Inventoriez les différents types de LSAs contenus dans chaque aire :

```
R8#sh ip ospf database
```

```
           OSPF Router with ID (1.0.0.8) (Process ID 1)

               Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
1.0.0.8         1.0.0.8         1160        0x80000003 0x00BFA2 5
1.0.0.9         1.0.0.9         1167        0x80000002 0x00161A 2
1.0.0.16        1.0.0.16        1161        0x80000006 0x0007D4 4
1.0.0.21        1.0.0.21        1184        0x80000002 0x00A3D2 2
1.0.0.22        1.0.0.22        1183        0x80000002 0x00B6BB 2

               Net Link States (Area 0)

Link ID         ADV Router      Age          Seq#       Checksum
10.0.1.16       1.0.0.16        1171         0x80000001 0x002BC8
10.0.16.22      1.0.0.22        1184         0x80000001 0x00644C

               Summary Net Link States (Area 0)

Link ID         ADV Router      Age          Seq#       Checksum
10.0.8.0        1.0.0.8         1135         0x80000003 0x007BA3
10.0.11.0       1.0.0.8         1141         0x80000001 0x0068B4
10.0.12.0       1.0.0.8         1141         0x80000001 0x005DBE

               Router Link States (Area 8)

Link ID         ADV Router      Age          Seq#       Checksum Link count
1.0.0.8         1.0.0.8         1147         0x80000002 0x00A442 1
1.0.0.11        1.0.0.11        1147         0x80000002 0x00CFE8 2
1.0.0.12        1.0.0.12        1147         0x80000002 0x00E2D1 2

               Net Link States (Area 8)

Link ID         ADV Router      Age          Seq#       Checksum
10.0.8.12       1.0.0.12        1148         0x80000001 0x00D315

               Summary Net Link States (Area 8)

Link ID         ADV Router      Age          Seq#       Checksum
10.0.1.0        1.0.0.8         1156         0x80000003 0x00C85D
10.0.2.0        1.0.0.8         1172         0x80000001 0x00F215
10.0.16.0       1.0.0.8         1162         0x80000001 0x0031E6
10.0.21.0       1.0.0.8         1162         0x80000001 0x00040E
10.0.22.0       1.0.0.8         1162         0x80000001 0x00F818
192.168.1.224   1.0.0.8         1172         0x80000001 0x00F3D7

               Summary ASB Link States (Area 8)

Link ID         ADV Router      Age          Seq#       Checksum
1.0.0.9         1.0.0.8         1162         0x80000001 0x0016F2

               Type-5 AS External Link States

Link ID         ADV Router      Age          Seq#       Checksum Tag
1.0.0.0         1.0.0.9         1169         0x80000001 0x003768 0
172.26.9.0      1.0.0.9         1169         0x80000001 0x00E2ED 0
R8#
```

Un véritable routeur en production aurait sûrement une liste de LSAs bien plus étendue que celle de notre modeste mise en situation. Il peut être utile dans ce cas d'entrer la commande **show ip ospf database database-summary**, commande qui fournit un résumé du condensé précédent :

```
R8#show ip ospf database database-summary

           OSPF Router with ID (1.0.0.8) (Process ID 1)

Area 0 database summary
  LSA Type      Count    Delete   Maxage
  Router        5        0        0
```

```
  Network       2        0        0
  Summary Net   3        0        0
  Summary ASBR  0        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Subtotal      10       0        0

Area 8 database summary
LSA Type      Count    Delete   Maxage
  Router        3        0        0
  Network       1        0        0
  Summary Net   6        0        0
  Summary ASBR  1        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Subtotal      11       0        0

Process 1 database summary
  LSA Type      Count    Delete   Maxage
  Router        8        0        0
  Network       3        0        0
  Summary Net   9        0        0
  Summary ASBR  1        0        0
  Type-7 Ext    0        0        0
Opaque Link   0        0        0
  Opaque Area   0        0        0
  Type-5 Ext    2        0        0
  Opaque AS     0        0        0
  Total         23       0        0
R8#
```

a. Rafraîchissement des LSAs

Comme précisé à la section Choix de l'instance convenable, un LSA naît avec l'âge 0 puis vieillit, soit quand il transite par une interface de sortie, soit au fur et à mesure qu'il réside dans la LSD. Un LSA qui atteint la limite d'âge n'est pas utilisé dans le calcul de la table de routage. Un routeur qui porte un LSA dont l'âge atteint la limite, tente de le supprimer de l'aire OSPF en l'inondant comme s'il s'agissait d'une nouvelle instance de LSA. Comme pour toute nouvelle instance, le LSA est également copié dans la « *Link State Retransmission List* » de chaque interface concernée. Le LSA est purgé d'une liste une fois l'acquittement reçu sur l'interface concernée. Si aucun des voisins n'est dans l'état « *Exchange* » ou « *Loading* », un LSA de « *MaxAge* » est supprimé de la LSD quand il n'est plus contenu dans aucune des listes « *Link State Retransmission List* ».

En pratique, sauf accident (disparition du routeur qui a généré ce LSA), l'âge d'un LSA ne devrait pas atteindre l'âge limite. Le RFC 2328 inventorie dix évènements susceptibles de provoquer la génération d'une nouvelle instance de LSA. Rappelons que quand un routeur génère une nouvelle instance de son LSA ou de l'un de ses LSAs, le numéro de séquence du LSA est incrémenté de 1 et l'âge du LSA remis à 0. La place manque pour passer en revue ces dix évènements, citons quand même :

- Le changement d'état d'une interface qui nécessite de produire une nouvelle instance de LSA de routeur.
- Un changement de routeur désigné DR sur le réseau de rattachement qui nécessite également de produire une nouvelle instance de LSA de routeur. De plus, si ce routeur est élu DR, il doit produire un nouveau LSA de réseau. Si ce routeur était DR et ne l'est plus, le LSA de réseau qu'il a généré par le passé doit être supprimé des LSDs de l'aire.
- Le champ « *LS age* » de l'un des LSAs atteint la valeur « *LSRefreshTime* » (30 minutes). Dans ce cas, quel que soit le contenu du LSA modifié ou pas (c'est le seul évènement parmi les dix inventoriés à imposer une nouvelle génération que le contenu du LSA ait changé ou pas), le routeur à l'origine de la création de ce LSA doit générer une nouvelle instance de son LSA.

Ce dernier élément est capital car il signifie que les LSAs sont rafraîchies de façon périodique et systématique, ce qui ajoute à la robustesse du protocole. En effet, les LSAs doivent être maintenus en vie pour rester dans les LSDs. Un LSA oublié finirait par disparaître, atteint par la limite d'âge. Un LSA corrompu est finalement remplacé par le rafraîchissement suivant.

L'autre intérêt d'associer de façon individuelle un âge à chaque LSA est que précisément, chaque LSA vit sa vie. Imaginez un instant que tous les LSA soient rafraîchis en même temps, il s'en suivrait un processus d'inondation

global qui nécessiterait ponctuellement un trafic d'acheminement peut-être très consommateur de bande passante et donc à même de compromettre le trafic utile. Au lieu de cela, un LSA est régénéré puis inondé puis un autre ou deux ou aucun et ainsi de suite, si bien que de façon globale, le trafic d'acheminement est réparti (lissé) de façon suffisamment aléatoire dans le temps.

Mais le mieux étant l'ennemi du bien, on en vient à regretter que ce trafic soit si émietté car chaque nouvelle instance de LSA inondée peut nécessiter d'être transportée par un paquet « *LS Update* » (un paquet par saut, donc autant de paquets que de sauts à franchir dans le réseau logique constitué par les relations de proximité). La bande passante est plus efficacement consommée quand un paquet « *LS Update* » (la voiture Break) embarque plusieurs LSAs. Comment faire pour réaliser un compromis entre un temporisateur unique (les LSAs sont tous rafraîchis en même temps) et un temporisateur individuel par LSA ?

Réfléchissons à nouveau avec notre parabole de la voiture break et des auto-stoppeurs. Un premier auto-stoppeur est monté dans la voiture. Pourtant, le chauffeur décidément très compréhensif, décide de différer un peu son départ car il reste des places libres dans le véhicule dont pourraient peut-être profiter d'autres auto-stoppeurs, à la condition qu'ils ne tardent pas trop à se manifester.

C'est la solution proposée par l'IOS CISCO à partir de la version 11.3AA et nommée « *pacing lsa-group* ». Avec ce mécanisme, les temporisateurs associés à chaque LSA sont bien individuels mais ils ne sont rafraîchis que tous les « *pacing lsa-group* » secondes, 240 secondes par défaut. Si bien que quand ce temporisateur expire, tous les LSAs qui doivent faire l'objet d'une nouvelle instance (c'est-à-dire tous ceux dont l'âge + 4 minutes dépasse « *LSRefreshTime* ») peuvent être groupés avant d'être inondés. La commande **timers pacing lsa-group *x*** en mode de configuration globale établit le temporisateur « *pacing lsa-group* » à *x* secondes, *x* doit être compris entre 10 et 1800 secondes.

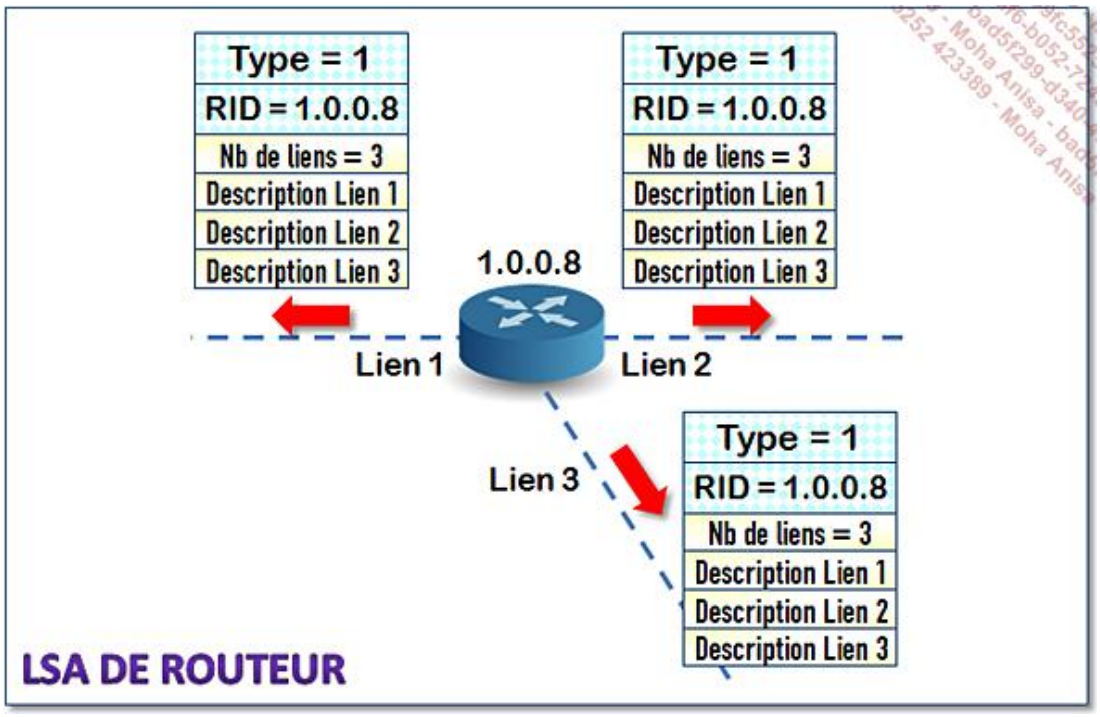
b. Les différents types de LSA

Type	Description	Inondé dans...
1	« *Router-LSA* », LSA de routeur	Aire
2	« *Network-LSA* », LSA de réseau	Aire
3	« *Summary-Net-LSA* », LSA résumé de réseau	Aire
4	« *Summary-ASBR-LSA* », LSA résumé ASBR	Aire
5	« *AS-external-LSA* », LSA externe à l'AS	Domaine OSPF dans son entier

D'autres types existent hors RFC2328 et qui dépassent le cadre de cette présentation, ils ne seront qu'évoqués.

LSA de routeur (Router-LSA)

C'est certainement le LSA le plus important puisqu'il est généré par chacun des routeurs comme l'atteste le résultat de la commande **show ip ospf database**. Le LSA de routeur est inondé sur l'ensemble des routeurs de l'aire dont il est issu. Il est possible de demander à ne visualiser que les LSA de routeur à l'aide de la commande **show ip ospf database router**, voire de demander le détail d'un LSA à l'aide de la commande **show ip ospf database router #LSID**. Dans le cas d'un LSA de routeur, LSID est en fait le RID du routeur qui a généré le LSA.



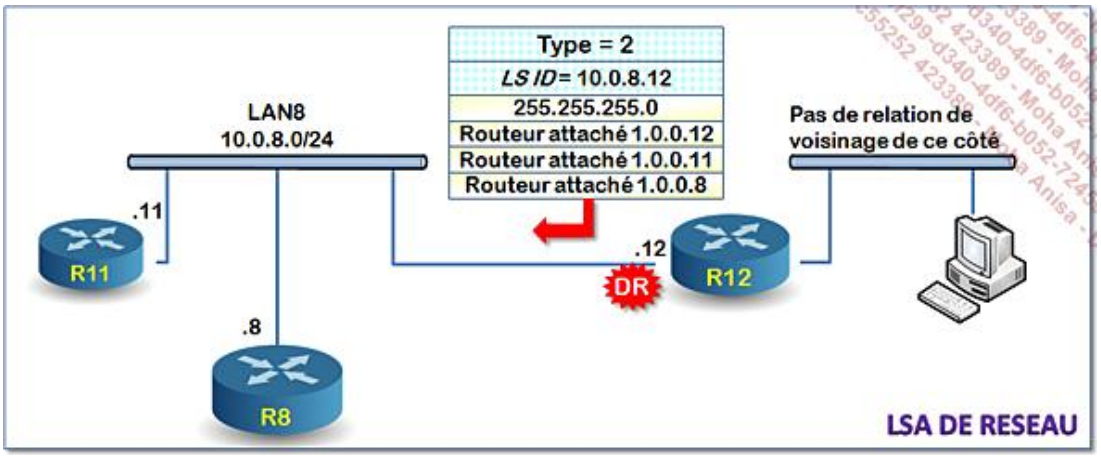
Observez les deux LSAs de routeur générés par l'ABR R8, l'un est inondé dans l'aire 0 et comporte cinq liens, l'autre est inondé dans l'aire 8 et ne comporte qu'un lien. Dans l'aire 0, le LSA a déjà fait l'objet de six instances, la dernière remonte à un peu plus de 31 minutes :

```
R8#sh ip ospf database router 1.0.0.8

           OSPF Router with ID (1.0.0.8) (Process ID 1)

               Router Link States (Area 0)

LS age: 1890
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 1.0.0.8
  Advertising Router: 1.0.0.8
  LS Seq Number: 80000006
  Checksum: 0xB9A5
  Length: 84
  Area Border Router
  Number of Links: 5

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 1.0.0.9
     (Link Data) Router Interface address: 192.168.1.226
      Number of TOS metrics: 0
       TOS 0 Metrics: 1562

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 192.168.1.224
     (Link Data) Network Mask: 255.255.255.252
      Number of TOS metrics: 0
       TOS 0 Metrics: 1562

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 1.0.0.16
     (Link Data) Router Interface address: 10.0.2.8
      Number of TOS metrics: 0
       TOS 0 Metrics: 1562

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 10.0.2.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
```

```
      TOS 0 Metrics: 1562

   Link connected to: a Transit Network
    (Link ID) Designated Router address: 10.0.1.16
    (Link Data) Router Interface address: 10.0.1.8
     Number of TOS metrics: 0
      TOS 0 Metrics: 1

               Router Link States (Area 8)

  LS age: 1891
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 1.0.0.8
  Advertising Router: 1.0.0.8
  LS Seq Number: 80000005
  Checksum: 0x9E45
  Length: 36
  Area Border Router
  Number of Links: 1

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.8.12
     (Link Data) Router Interface address: 10.0.8.8
      Number of TOS metrics: 0
       TOS 0 Metrics: 1
```

LSA de réseau (Network-LSA)

Souvenons-nous que le support physique d'un réseau à diffusion est assimilé à un nœud auquel chaque routeur (un nœud également) est relié par son interface réseau qui devient un arc du graphe. Mais puisque le nœud réseau représentant le support physique ne peut participer activement au protocole OSPF, le DR se substitue à lui. C'est donc le DR qui doit générer le LSA de réseau, LSA qui représente le nœud du support. Le LSA de réseau énumère l'ensemble des routeurs connectés au réseau de rattachement, y compris le DR lui-même. Le LSA de réseau est inondé sur l'ensemble des routeurs de l'aire dont il est issu.



L'administrateur peut visualiser l'ensemble des LSAs de réseau à l'aide de la commande **show ip ospf database network** :

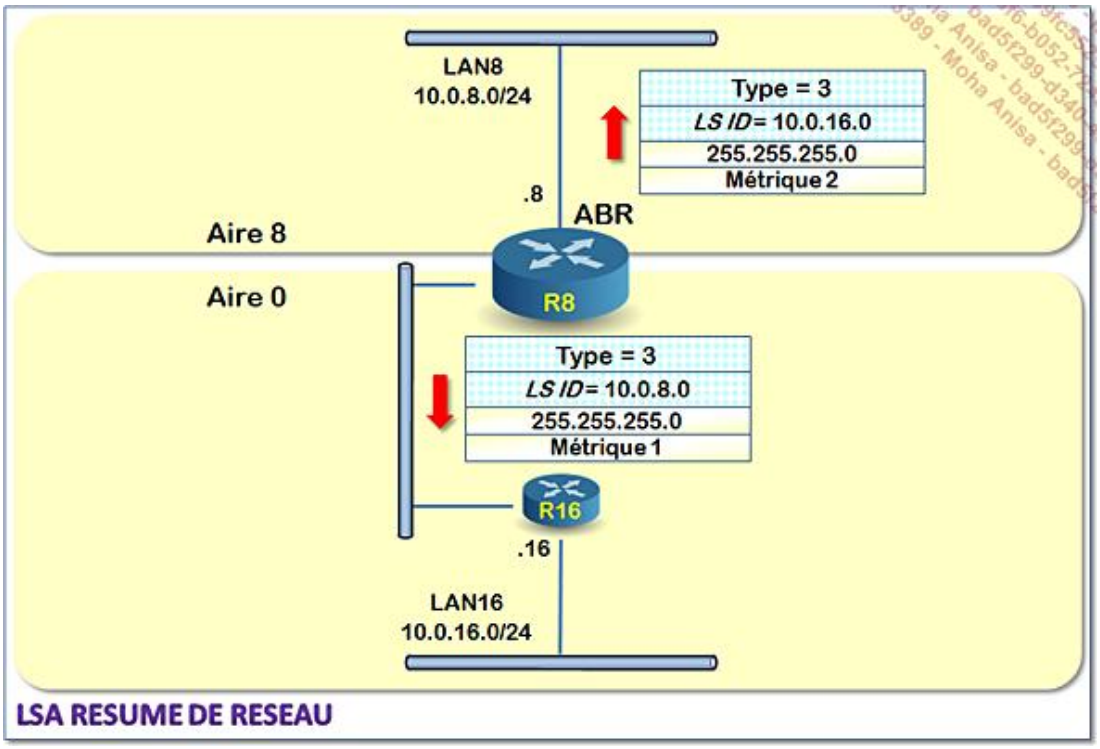
```
R8#show ip ospf database network

            OSPF Router with ID (1.0.0.8) (Process ID 1)

                Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 40
  Options: (No TOS-capability, DC)
  LS Type: Network Links
  Link State ID: 10.0.1.16 (address of Designated Router)
  Advertising Router: 1.0.0.16
  LS Seq Number: 80000001
  Checksum: 0x2BC8
```

```
Length: 32
Network Mask: /24
      Attached Router: 1.0.0.16
      Attached Router: 1.0.0.8

Routing Bit Set on this LSA
LS age: 64
Options: (No TOS-capability, DC)
LS Type: Network Links
Link State ID: 10.0.16.22 (address of Designated Router)
Advertising Router: 1.0.0.22
LS Seq Number: 80000002
Checksum: 0x624D
Length: 36
Network Mask: /24
      Attached Router: 1.0.0.22
      Attached Router: 1.0.0.16
      Attached Router: 1.0.0.21

          Net Link States (Area 8)

Routing Bit Set on this LSA
LS age: 41
Options: (No TOS-capability, DC)
LS Type: Network Links
Link State ID: 10.0.8.12 (address of Designated Router)
Advertising Router: 1.0.0.12
LS Seq Number: 80000002
Checksum: 0xD116
Length: 36
Network Mask: /24
      Attached Router: 1.0.0.12
      Attached Router: 1.0.0.8
      Attached Router: 1.0.0.11
```

À nouveau, puisque R8 est un ABR, il contient les LSAs de réseau de l'aire 0 et de l'aire 8. Observez que l'identifiant « Link State ID » d'un LSA de réseau n'est autre que l'adresse IP de l'interface du DR sur le réseau de rattachement.

LSA résumé de réseau (Summary-Net-LSA)

Seul un ABR peut générer un LSA résumé de réseau. Il génère un tel LSA pour chaque destination accessible par lui en dehors de l'aire. Ainsi, R8 dans notre atelier OSPF génère trois LSA résumés de réseau pour annoncer dans l'aire 0 les trois destinations qu'il connaît dans l'aire 8, à savoir les réseaux 10.0.8.0/24, 10.0.11.0/24 et 10.0.12.0/24. De la même façon, il génère six LSA résumés de réseau pour annoncer dans l'aire 8 les six destinations qu'il connaît dans l'aire 0, à savoir les réseaux 10.0.1.0/24, 10.0.2.0/24, 10.0.16.0/24, 10.0.21.0/24, 10.0.22.0/24 et 192.168.1.224/30.



L'administrateur peut visualiser l'ensemble des LSAs résumés de réseau à l'aide de la commande **show ip ospf database summary**. Dans la capture ci-dessous, la LSD de l'aire 0 comporte trois LSA résumés de réseau tous trois générés par l'ABR R8.

```
R16#show ip ospf database summary

            OSPF Router with ID (1.0.0.16) (Process ID 1)

                Summary Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 738
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 10.0.8.0 (summary Network Number)
  Advertising Router: 1.0.0.8
  LS Seq Number: 80000001
  Checksum: 0x7FA1
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 1

  Routing Bit Set on this LSA
  LS age: 738
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 10.0.11.0 (summary Network Number)
  Advertising Router: 1.0.0.8
  LS Seq Number: 80000001
  Checksum: 0x68B4
  Length: 28
  Network Mask: /24
        TOS: 0  Metric: 2

  Routing Bit Set on this LSA
  LS age: 739
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 10.0.12.0 (summary Network Number)
  Advertising Router: 1.0.0.8
  LS Seq Number: 80000001
  Checksum: 0x5DBE
```

```
  Length: 28
  Network Mask: /24
TOS: 0  Metric: 2
```

Dans la capture ci-dessous, la LSD comporte parmi d'autres un LSA résumé de réseau qui annonce une route par défaut via l'ABR. Ce LSA doit son existence au fait que la configuration de l'aire 8 a été modifiée pour en faire une zone de bout (**area 8 stub** en configuration de routeur OSPF sur chacun des trois routeurs R8, R11 et R12) :

```
R11#sh ip ospf database summary

            OSPF Router with ID (1.0.0.11) (Process ID 1)

                Summary Net Link States (Area 8)

  Routing Bit Set on this LSA
  LS age: 119
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(Network)
  Link State ID: 0.0.0.0 (summary Network Number)
  Advertising Router: 1.0.0.8
  LS Seq Number: 80000001
  Checksum: 0x78BC
  Length: 28
  Network Mask: /0
TOS: 0  Metric: 1

......... 6 autres LSA suivent .........
Link State ID: 10.0.1.0 (summary Network Number)
  Link State ID: 10.0.2.0 (summary Network Number)
  Link State ID: 10.0.16.0 (summary Network Number)
  Link State ID: 10.0.21.0 (summary Network Number)
  Link State ID: 10.0.22.0 (summary Network Number)
  Link State ID: 192.168.1.224 (summary Network Number)
```

Observez que chaque destination annoncée est associée à un coût. Si l'ABR connaît une même destination via deux routes différentes, il n'annonce la destination une seule fois associée au meilleur coût.

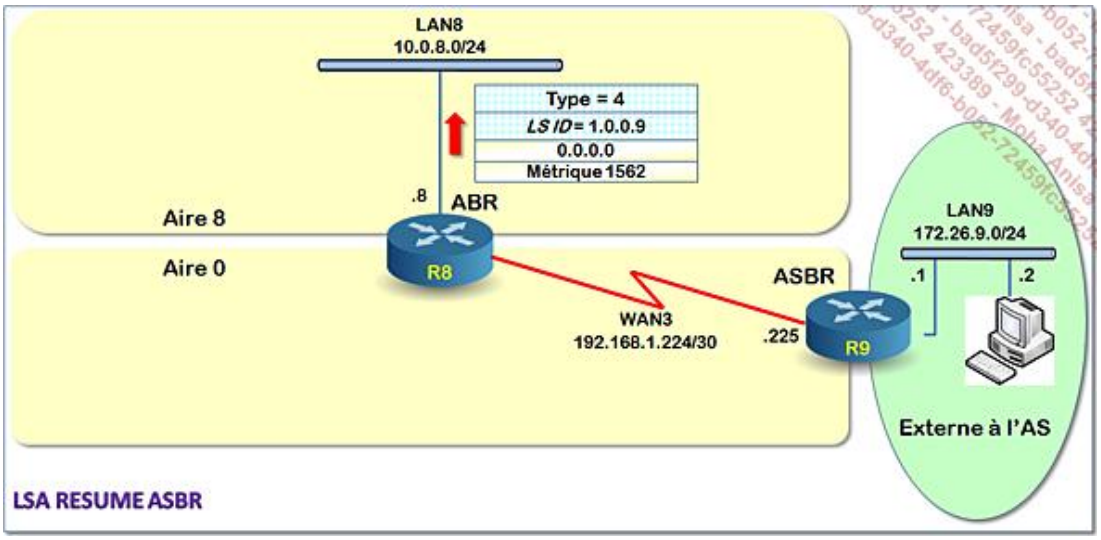
Quand un routeur (R11 par exemple) reçoit un LSA résumé de réseau d'un ABR (R8), il inclut la destination annoncée dans sa table de routage via l'ABR. Le coût de la route ajoutée est la somme du coût annoncé dans le LSA résumé et du coût pour atteindre l'ABR. En poursuivant l'exemple, le LSA résumé identifié 10.0.16.0 a été annoncé par l'ABR avec un coût de 2. R11 sait joindre R8 avec un coût de 1, R11 ajoute une route vers 10.0.16.0 via l'ABR au coût de 3 :

```
R11#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........
Gateway of last resort is 10.0.8.8 to network 0.0.0.0
.........
O IA    10.0.16.0 [110/3] via 10.0.8.8, 01:24:40, FastEthernet0/1
.........
O*IA 0.0.0.0/0 [110/2] via 10.0.8.8, 01:24:41, FastEthernet0/1
```

Observez la mention « IA » ajoutée à la route qui rappelle qu'il s'agit d'un routeur inter-aires. Il est amusant de constater que ce faisant, OSPF adopte un comportement de type vecteur de distance. Le comportement à états de liens du protocole OSPF est confiné aux aires.

LSA résumé ASBR (Summary-ASBR-LSA)

Le LSA résumé ASBR est identique au LSA résumé de réseau à ceci près qu'il annonce un routeur ASBR et non une destination. Notre contexte d'atelier ne connaît qu'un seul routeur ASBR annoncé par l'ABR dans l'aire 8 à l'aide du LSA identifié 1.0.0.9 :



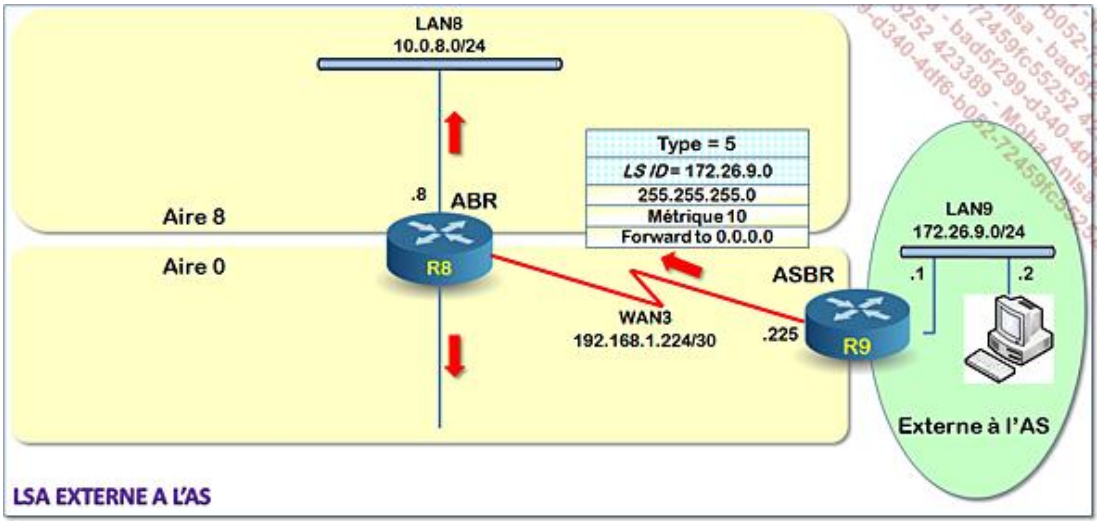
L'administrateur peut visualiser les LSAs résumés ASBR à l'aide d'une commande **show ip ospf database asbr-summary** :

```
R8#show ip ospf database asbr-summary

            OSPF Router with ID (1.0.0.8) (Process ID 1)

                Summary ASB Link States (Area 8)

  LS age: 171
  Options: (No TOS-capability, DC, Upward)
  LS Type: Summary Links(AS Boundary Router)
  Link State ID: 1.0.0.9 (AS Boundary Router address)
  Advertising Router: 1.0.0.8
  LS Seq Number: 80000001
  Checksum: 0x16F2
  Length: 28
  Network Mask: /0
  TOS: 0  Metric: 1562
```

Observez que le champ « Link State ID » identifie cette fois le routeur ASBR. Observez également que le masque de réseau reste à 0.

LSA externe à l'AS (AS-external-LSA)

Seul un routeur ASBR peut générer un LSA externe à l'AS. Il le fait pour annoncer soit une destination externe à l'AS, soit une route par défaut externe à l'AS. Un LSA externe à l'AS est inondé dans l'intégralité du système autonome, c'est le seul LSA parmi les cinq décrits à être dans ce cas.



La commande **show ip ospf database external** affiche les LSAs externes à l'AS :

```
R8#sh ip ospf database external

          OSPF Router with ID (1.0.0.8) (Process ID 1)

              Type-5 AS External Link States

.........

  Routing Bit Set on this LSA
  LS age: 259
  Options: (No TOS-capability, DC)
  LS Type: AS External Link
  Link State ID: 172.26.9.0 (External Network Number)
  Advertising Router: 1.0.0.9
  LS Seq Number: 80000007
  Checksum: 0xD6F3
  Length: 36
  Network Mask: /24
        Metric Type: 2 (Larger than any link state path)
        TOS: 0
        Metric: 10
        Forward Address: 0.0.0.0
        External Route Tag: 0
```

L'adresse IP et le masque de la destination annoncée sont contenus dans les champs « Link State ID » et « Network Mask » du LSA. Observez le champ « Forward Address ». Quand le champ est réglé à 0.0.0.0, les paquets doivent être transmis à l'ASBR lui-même.

Autres types de LSA

Une commande **show ip ospf database ?** est édifiante sur ce qui resterait à découvrir :

```
R16#show ip ospf database ?
adv-router        Advertising Router link states → LSAs générés par 1routspécifié
asbr-summary      ASBR summary link states → Type 4
database-summary  Summary of database → information condensée
external          External link states → Type 5
network           Network link states → Type 2
nssa-external     NSSA External link states → Type 7
opaque-area       Opaque Area link states
opaque-as         Opaque AS link states
opaque-link       Opaque Link-Local link states
router            Router link states → Type 1
self-originate    Self-originated link states → Les LSAs générés par ce routeur
summary           Network summary link states → Type 3
|                 Output modifiers
<cr>
```

Deux mots-clés de la commande ne sont pas des types de LSAs mais des facilités offertes à l'administrateur :

- **adv-router** : permet de n'afficher que les LSAs générés par un routeur spécifié.
- **self-originate** : permet de n'afficher que les LSAs générés par ce routeur.

Les mots-clés **nssa** (Not-so-stubby) et opaque en revanche font référence à des types de LSAs. Certains types ne sont pas supportés par CISCO et manquent donc à l'appel. Au moment où ces lignes sont écrites, il faut ajouter à la liste des cinq types de LSAs déjà décrits, les types suivants :

« Group Membership LSA » (Type 6)

Extension d'OSPF connue sous le nom de MOSPF (Multicast OSPF), objet du RFC 1584. Parce que MOSPF nécessite un algorithme SPF par arbre ou groupe multicast, il y a un risque pour qu'un routeur qui supporterait cette extension doive faire fonctionner des centaines ou milliers de processus OSPF. Au mieux, MOSPF fonctionnerait avec quelques groupes multicast. C'est pourquoi CISCO a choisi de ne pas l'implémenter et a préféré poursuivre la voie des protocoles PIM (*Protocol Independent Multicast*). PIM présente l'avantage de ne pas induire sa propre topologie mais de se servir des routes apprises par les protocoles de routage connus. Inutile d'aller plus loin donc...

« NSSA External LSA » (Type 7)

Le LSA de type 7 est identique au type 5, il ne peut être généré que par un routeur ASBR, à ceci près qu'il n'est inondé que dans l'aire NSSA. D'autres détails sont fournis dans la section L'aire NSSA. L'aire OSPF NSSA (Not-So-Stubby) est décrite dans le RFC 3101.

« ExternalAttribute LSA » (Type 8)

L'objet du LSA de type 8 est de transporter de l'information BGP (protocole de routage utilisé entre systèmes autonomes) au travers d'un domaine OSPF. Au moment où ces lignes sont écrites, le type 8 n'a pas été décrit dans un RFC et n'a donc pas été implémenté.

« Opaque LSA » (Types 9, 10 et 11)

Ces LSAs sont décrits dans le RFC 5250. Il s'agit de transporter dans des LSAs de l'information spécifique à certaines applications, par exemple MPLS (*Multi Protocol Label Switching*). Les trois types ne diffèrent que par leur portée :

- Type 9 : portée limitée au lien local (*Link Local Scope*), c'est-à-dire à l'ensemble des interfaces directement connectées ou comme on voudra au réseau ou sous-réseau.
- Type 10 : portée étendue à l'aire.
- Type 11 : portée étendue à tout le système autonome.

Cas particulier des aires de bout

Raisonnons à nouveau sur l'atelier 9a. On observe que toute route externe apprise par l'ASBR est inondée dans l'aire 8 alors même que cette aire ne présente qu'une seule route de sortie via l'ABR. Dans la vraie vie, la quantité de LSAs externes à l'AS peut représenter une part importante des LSAs présents en LSD.

Faire de l'aire 8 une aire de bout (*stub area*) offre le moyen de remplacer l'ensemble de ces LSAs externes par une simple route par défaut annoncée par le routeur ABR à l'aide d'un LSA résumé de réseau (type 3). Par ailleurs, un routeur qui veut exploiter une route externe à l'AS a besoin du LSA externe à l'AS (type 5) mais a également besoin du LSA résumé ASBR (type 4) pour trouver la route vers l'ASBR. En supprimant l'inondation des LSAs de type 5, on peut également supprimer les LSAs de type 4, ils ne sont plus nécessaires. Dans l'aire de bout, tout paquet émis vers une destination pour laquelle un routeur de l'aire ne trouve pas de route ni parmi les routes intra-aire ni parmi les routes inter-aires est acheminé sur la route par défaut.

La LSD d'une aire de bout ne contient ni LSA externe à l'AS, ni LSA résumé ASBR. Ceci préserve la mémoire des routeurs de l'aire. Ce peut être une solution pour retarder l'achat de mémoire supplémentaire voire différer le remplacement des équipements.

Cet avantage a un prix, une aire de bout impose les contraintes suivantes :

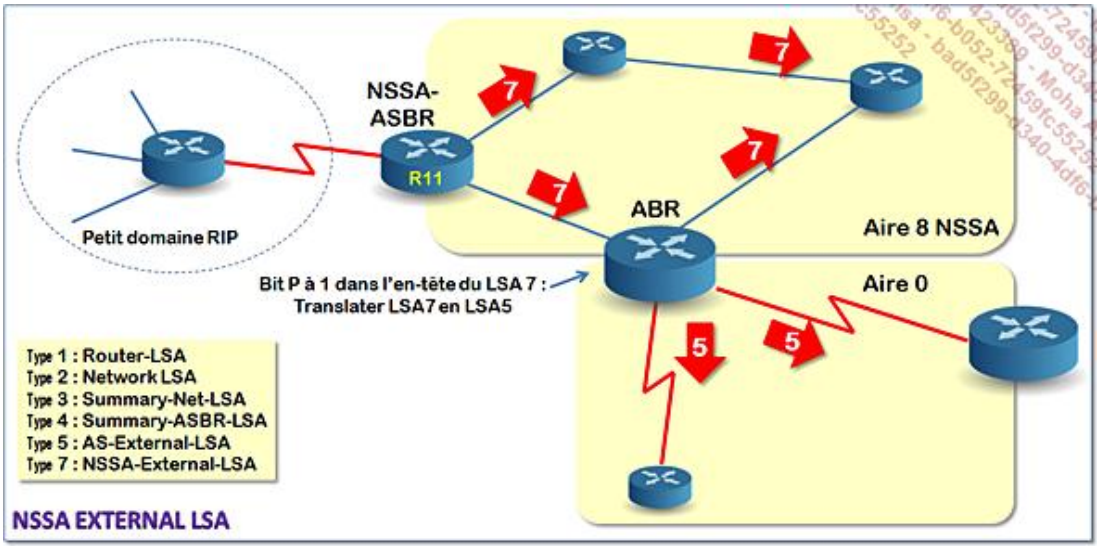
- Tous les routeurs d'une aire de bout doivent être configurés en tant que tel. Ainsi pour l'aire 8, il faudrait en configuration de routeur OSPF, ajouter la commande **area 8 stub** sur chacun des trois routeurs R8, R11 et R12. Ceci a pour effet de mettre à 0 le bit E (*External*) dans le champ Options du message Hello (voir Format des messages OSPF - Le message Hello plus loin dans ce chapitre). Or ce bit E doit être identique dans les messages Hello issus de deux routeurs pour que la relation de proximité puisse se construire. En final, l'effet de cette configuration est d'interdire l'établissement d'une relation de proximité entre un routeur « stub » et un routeur qui ne l'est pas.
- Il est impossible d'établir un lien virtuel sur une aire de bout (logique, elle ne serait plus au bout !).
- Un routeur « stub » ne peut être ASBR. En effet, un ASBR annonce des LSAs de type 5 et ce type est interdit dans une aire de bout.
- Rien n'interdit à l'architecte de doter l'aire de bout de plusieurs ABRs mais dans ce cas, rien ne dit qu'un routeur de l'aire fera le choix optimal quand il faudra acheminer un paquet vers l'extérieur.

Cas particulier des aires totalement de bout

Il s'agit d'étendre encore le concept de l'aire de bout en supprimant également les LSAs résumés de réseau. La conséquence est que les routeurs d'une aire totalement de bout ne peuvent plus apprendre de routes inter-aires. Tout paquet dont la destination est externe à l'aire est acheminé sur la route par défaut donc transmis à l'ABR. Le seul LSA de type 3 encore autorisé sur une aire totalement de bout est celui annonçant la route par défaut.

L'aire NSSA

L'aire NSSA, décrite dans le RFC 3101, est une extension de l'aire de bout. Si la plupart des restrictions imposées par une aire configurée en aire de bout sont acceptables, il arrive qu'un architecte souhaite importer une petite quantité d'informations de routage externes dans l'aire pour ensuite distribuer cette information au reste du domaine OSPF. Appuyons notre réflexion sur la figure suivante :



Dans cet exemple, l'architecte souhaite importer les routes d'un petit domaine RIP qui jouxte l'aire de bout 8. Mais la taille du domaine RIP ne justifie pas de réactiver le support des LSAs externes à l'AS dans l'aire 8 et de plus, les routeurs de cette aire ne disposent pas de capacités suffisantes pour le faire (manque de mémoire, trop de LSAs externes à l'AS issus de l'aire backbone).

L'aire NSSA offre la solution en permettant aux routes issues du domaine RIP d'être annoncées à l'aide de « *NSSA External LSA* ». Ce LSA de type 7 est identique au LSA de type 5, il a également pour rôle d'annoncer des routes externes à l'AS mais il diffère en un point : sa portée est limitée à l'aire NSSA, il est bloqué par l'ABR et n'est donc pas inondé dans le reste du domaine OSPF.

Dans la topologie ci-dessus, l'aire 8 devenue aire NSSA fait de R11 un routeur « NSSA ASBR », ce routeur génère les LSAs de type 7. Il peut en outre décider de mettre à 1 ou pas le bit « P » (*Propagate*) contenu dans le champ Options de l'en-tête LSA. Quand il le fait, le routeur ABR sait qu'il doit convertir les LSAs de type 7 en LSAs de type 5 afin de les inonder dans le reste du domaine OSPF.

9. Remplissage de la table de routage

Souvenons-nous que la métrique d'OSPF est fondée sur une notion de coût. La structure de données associée à chaque interface comporte entre autres un coût résultat du calcul 10^8/*Bandwidth*. Le coût d'une route est le coût cumulé de chaque interface de sortie qui fait progresser le trafic sur cette route.

L'administrateur peut influer sur le coût d'une interface de deux façons différentes :

- En ajustant la bande passante de l'interface à une valeur réelle ou déclarée à l'aide de la commande **bandwidth** en configuration d'interface.

- En réglant directement le coût à l'aide de la commande **ip ospf cost**, également en configuration d'interface.

```
R11(config)#int f0/1
R11(config-if)#ip ospf cost ?
   <1-65535>  Cost
R11(config-if)#bandwidth ?
   <1-10000000>  Bandwidth in kilobits
```

Le champ « metric » du LSA de routeur est exprimé sur 16 bits, le coût de l'interface doit donc être compris entre 1 et 65535. En revanche, le RFC 2328 ne limite pas le coût d'une route (compréhensible car ce coût n'est pas transporté par les paquets OSPF).

Le tableau suivant énumère les débits les plus fréquemment rencontrés et le coût OSPF correspondant :

Interface	Débit	Coût (10^8/Bandwidth)
Ethernet Gigabit	1 Gbps	1
FDDI, Fast Ethernet	100 Mbps	1
HSSI (*High Speed Serial Interface*)	45 Mbps	2
Token Ring 16	16 Mbps	6
Ethernet historique	10 Mbps	10
T1 (24 canaux à 8 bits = 192 bits + 1 bit de frame, le tout 8000 fois par seconde)	1,544 Mbps	64
DS0 (*Digital Signal 0*) ou E0	64 Kbps	1562
	56 Kbps	1785

Hélas, seules les interfaces LAN des routeurs CISCO sont automatiquement configurées avec la bande passante convenable. Les interfaces WAN ne peuvent pas l'être car le débit est en réalité cadencé par l'horloge fournie par le boîtier ETCD ou CSU/DSU. L'IOS n'établit aucune corrélation entre le débit physique, cadencé par l'équipement de terminaison de circuit de données et le paramètre **bandwidth** de la configuration d'interface. Par défaut, l'IOS considère que l'interface « *serial* » est connectée à un canal T1 de débit 1,544 Mbps (décidément, CISCO est américain !). Par voie de conséquence, OSPF associe à l'interface un coût de 1562. C'est donc à l'administrateur qu'il revient de rétablir la vraie bande passante des liens WAN s'il souhaite qu'OSPF déroule l'algorithme SPF avec des coûts réels.

a. Classification des destinations

Nous sommes bien entraînés à l'utilisation de la table de routage mais l'administrateur sait moins qu'en fait OSPF alimente deux tables de routage, l'une regroupe les destinations vers les réseaux, c'est elle que nous observons à l'aide de la commande **show ip route**, l'autre regroupe les routeurs ABR et ASBR et peut être observée à l'aide de la commande **show ip ospf border-routers**.

Toujours dans le contexte de l'atelier 9a, les destinations vers les réseaux :

```
R11#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     1.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
O E2    1.0.0.0/24 [110/10] via 10.0.8.8, 00:00:49, FastEthernet0/1
C       1.0.0.11/32 is directly connected, Loopback0
     172.26.0.0/24 is subnetted, 1 subnets
O E2    172.26.9.0 [110/10] via 10.0.8.8, 00:00:49, FastEthernet0/1
     10.0.0.0/24 is subnetted, 8 subnets
C       10.0.11.0 is directly connected, FastEthernet0/0
C       10.0.8.0 is directly connected, FastEthernet0/1
O       10.0.12.0 [110/2] via 10.0.8.12, 00:00:49, FastEthernet0/1
O IA    10.0.2.0 [110/1563] via 10.0.8.8, 00:00:49, FastEthernet0/1
O IA    10.0.1.0 [110/2] via 10.0.8.8, 00:00:49, FastEthernet0/1
O IA    10.0.16.0 [110/3] via 10.0.8.8, 00:00:49, FastEthernet0/1
O IA    10.0.22.0 [110/4] via 10.0.8.8, 00:00:49, FastEthernet0/1
O IA    10.0.21.0 [110/4] via 10.0.8.8, 00:00:50, FastEthernet0/1
     192.168.1.0/30 is subnetted, 1 subnets
O IA    192.168.1.224 [110/1563] via 10.0.8.8, 00:00:50, FastEthernet0/1
R11#
```

Les routeurs ABR et ASBR :

```
R11#sh ip ospf border-routers

OSPF Process 1 internal Routing Table

Codes: i - Intra-area route, I - Inter-area route

I 1.0.0.9 [1563] via 10.0.8.8, FastEthernet0/1, ASBR, Area 8, SPF 3
i 1.0.0.8 [1] via 10.0.8.8, FastEthernet0/1, ABR, Area 8, SPF 3
R11#
```

Dans cette seconde table, observez que les destinations ne pointent pas vers des réseaux mais vers des routeurs identifiés par leur RID. Hormis cette différence, on a bien affaire à des routes dotées d'une métrique, d'une adresse de prochain saut et d'une interface de sortie. Les routeurs ABR sont tagués à l'aide de la lettre minuscule « i », les routeurs ASBR le sont à l'aide de la lettre majuscule « I ».

b. Classification des routes

Il est possible de classer les routes en quatre familles :

- Une route intra-aire mène à un réseau situé dans l'aire et donc connectée à l'un des routeurs de l'aire. Exemple dans la table de routage de R11 objet de la capture précédente :

```
O       10.0.12.0 [110/2] via 10.0.8.12, 00:00:49, FastEthernet0/1
```

- Une route inter-aire mène à un réseau situé en dehors de l'aire mais à l'intérieur du système autonome. Une telle route transite nécessairement par un ABR. La route est taguée IA dans la capture, exemple :

```
O IA    10.0.22.0 [110/4] via 10.0.8.8, 00:00:49, FastEthernet0/1
```

- Une route externe de type 1 mène à une destination située en dehors du système autonome. Quand l'ASBR annonce cette destination, il doit en outre lui associer un coût cohérent pour le domaine OSPF. Quand l'ASBR annonce une destination de type 1, un routeur qui reçoit le LSA ajoute la destination à sa table de routage en lui associant un coût égal à la somme du coût annoncé dans le LSA et du coût pour atteindre l'ASBR.

- Une route externe de type 2 mène également à une destination en dehors du SA mais le routeur qui reçoit le LSA de l'ASBR ajoute la destination à sa table de routage en lui associant le coût annoncé et donc sans tenir compte du coût pour atteindre l'ASBR.

Plaçons-nous encore dans le contexte de l'atelier 9a. L'administrateur a configuré sur R9 la redistribution des routes directement connectées :

```
R9(config)#router ospf 1
R9(config-router)#redistribute connected metric 10
```

L'IOS CISCO annonce par défaut les routes externes en type 2. Puisque l'administrateur a appliqué un coût de 10 à toute route connectée redistribuée, on lit dans la table de routage de R11, l'entrée suivante :

```
O E2    172.26.9.0 [110/10] via 10.0.8.8, 00:00:49, FastEthernet0/1
```

Observez que le coût de la route est resté fixé à 10, la route est taguée « E2 ». L'administrateur souhaite basculer vers le type 1, il modifie pour ce faire la configuration de l'ASBR :

```
R9(config)#router ospf 1
R9(config-router)#redistribute connected metric 10 ?
metric       Metric for redistributed routes
metric-type  OSPF/IS-IS exterior metric type for redistributed routes
route-map    Route map reference
subnets      Consider subnets for redistribution into OSPF
tag          Set tag for routes redistributed into OSPF
<cr>

R9(config-router)#redistribute connected metric 10 metric-type ?
```

```
1  Set OSPF External Type 1 metrics
2  Set OSPF External Type 2 metrics

R9(config-router)#redistribute connected metric 10 metric-type 1 subnets
R9(config-router)#^Z
R9#
```

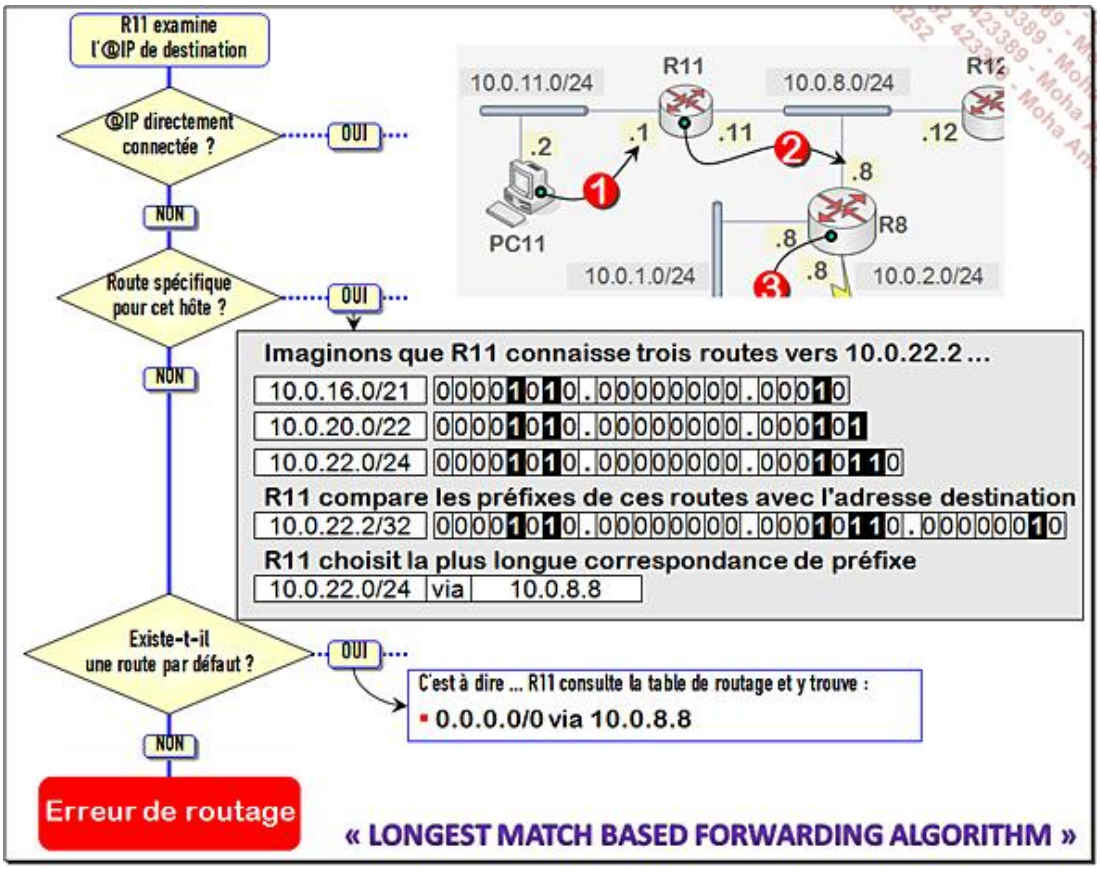
Immédiatement après la modification apportée à R9, l'administrateur revient à R11 :

```
R11#sh ip route
.........
O E1    172.26.9.0 [110/1573] via 10.0.8.8, 00:05:57, FastEthernet0/1
.........
```

L'administrateur constate satisfait que la route est cette fois taguée « E1 » et que le coût associé est bien celui pour rejoindre l'ASBR, soit 1563 (vérifier dans le résultat de la commande **show ip ospf border-routers**) additionné du coût 10 affecté à la route redistribuée.

c. À la recherche de la meilleure route

Toutes les routes n'ont pas le même degré d'acuité. Vous êtes à Bruxelles et vous allez à Marseille. Au premier croisement, deux routes se présentent : le panneau indicateur de la première indique « Toutes directions », le panneau de la seconde indique « Marseille ». Vous êtes intrigué mais vous choisissez la seconde. Ainsi, le réseau 10.0.22.0/24 englobe la machine 10.0.22.2 mais le réseau 10.0.16.0/21 englobe le réseau 10.0.22.0/24 et donc la machine 10.0.22.2. Dans sa table de routage, pour l'adresse de destination 10.0.22.2, le routeur préférera la route la plus spécifique 10.0.22.0/24 à la route la plus générale 10.0.16.0/21 :



Pour trouver la route la plus spécifique, le processus de routage utilise l'algorithme de recherche de correspondance de préfixe la plus longue (*Longest Match based Forwarding Algorithm*).

Si toutes les recherches précédentes ont échoué, le processus de routage utilise la route par défaut si elle existe. Enfin, pour le cas ultime où le processus de routage n'a pas trouvé de route et ne dispose pas d'une route par défaut, le datagramme est supprimé (« *dropped* », c'est l'action Poubelle) et le routeur génère un message ICMP « *Destination Host Unreachable* » destiné à alerter l'émetteur du datagramme supprimé.

d. Partage de charge (Load balancing)

Chaque fois que l'algorithme SPF est exécuté, il place les routes au meilleur coût dans la table de routage. Si une destination préexistait et que l'algorithme découvre une route à un meilleur coût, cette route se substitue à la route moins favorable. Quand l'algorithme découvre plusieurs routes de coûts identiques, alors il peut les placer ensemble dans la table de routage, ce jusqu'à concurrence de 16 routes, quoique l'IOS limite à 4 le nombre de routes à coût égal par défaut. Quant au nombre de 16 maximal, il est à prendre avec précaution et dépend de la plate-forme ainsi que de la version d'IOS, il est préférable de vérifier ce nombre avec l'aide en ligne de commande :

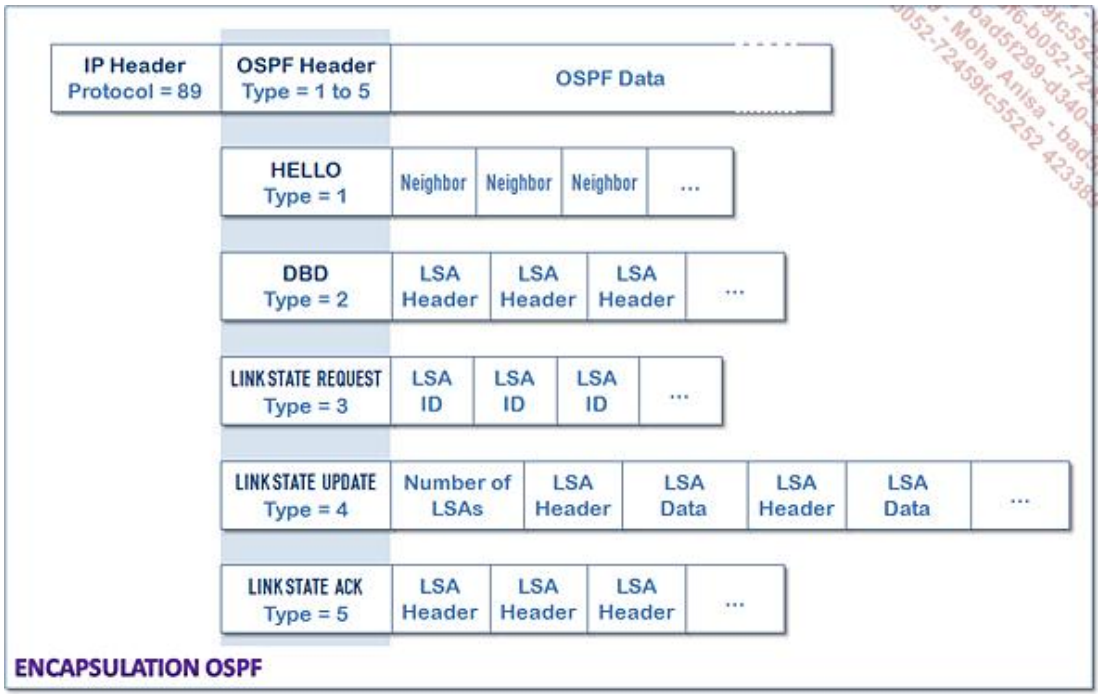
```
R8(config)#router ospf 1
R8(config-router)#maximum-paths ?
<1-6>  Number of paths
```

Ainsi, sur la version d'IOS en cours dans notre atelier 9a, le nombre de routes à coût égal ne peut dépasser 6. La façon dont l'IOS exploite ensuite ces routes est indépendante du protocole OSPF et a déjà été explicitée, le lecteur pourra se reporter aux sections Partage de charge par paquet et Partage de charge par destination dans le chapitre Le routage statique. À nouveau dans notre contexte favori, l'administrateur a configuré le coût de l'interface S0/1 sur R8 à 1 afin que les deux routes qui séparent R8 et R16 aient coût égal. Observez la présence de deux routes pour les destinations 10.0.16.0, 10.0.22.0 et 10.0.21.0 :

```
R8(config)#int s0/1
R8(config-if)#ip ospf cost 1
R8(config-if)#^Z
R8#sh ip route
.........
     1.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
O E1    1.0.0.0/24 [110/1572] via 192.168.1.225, 00:00:39, Serial0/0
C       1.0.0.8/32 is directly connected, Loopback0
     172.26.0.0/24 is subnetted, 1 subnets
O E1    172.26.9.0 [110/1572] via 192.168.1.225, 00:00:39, Serial0/0
     10.0.0.0/24 is subnetted, 8 subnets
O       10.0.11.0 [110/2] via 10.0.8.11, 01:15:08, FastEthernet0/0
C       10.0.8.0 is directly connected, FastEthernet0/0
O       10.0.12.0 [110/2] via 10.0.8.12, 01:15:08, FastEthernet0/0
C       10.0.2.0 is directly connected, Serial0/1
C       10.0.1.0 is directly connected, FastEthernet0/1
O       10.0.16.0 [110/2] via 10.0.1.16, 00:00:39, FastEthernet0/1
                  [110/2] via 10.0.2.16, 00:00:39, Serial0/1
O       10.0.22.0 [110/3] via 10.0.1.16, 00:00:58, FastEthernet0/1
                  [110/3] via 10.0.2.16, 00:00:58, Serial0/1
O       10.0.21.0 [110/3] via 10.0.1.16, 00:00:58, FastEthernet0/1
                  [110/3] via 10.0.2.16, 00:00:58, Serial0/1
     192.168.1.0/30 is subnetted, 1 subnets
C       192.168.1.224 is directly connected, Serial0/0
R8#
```

10. Format des messages OSPF

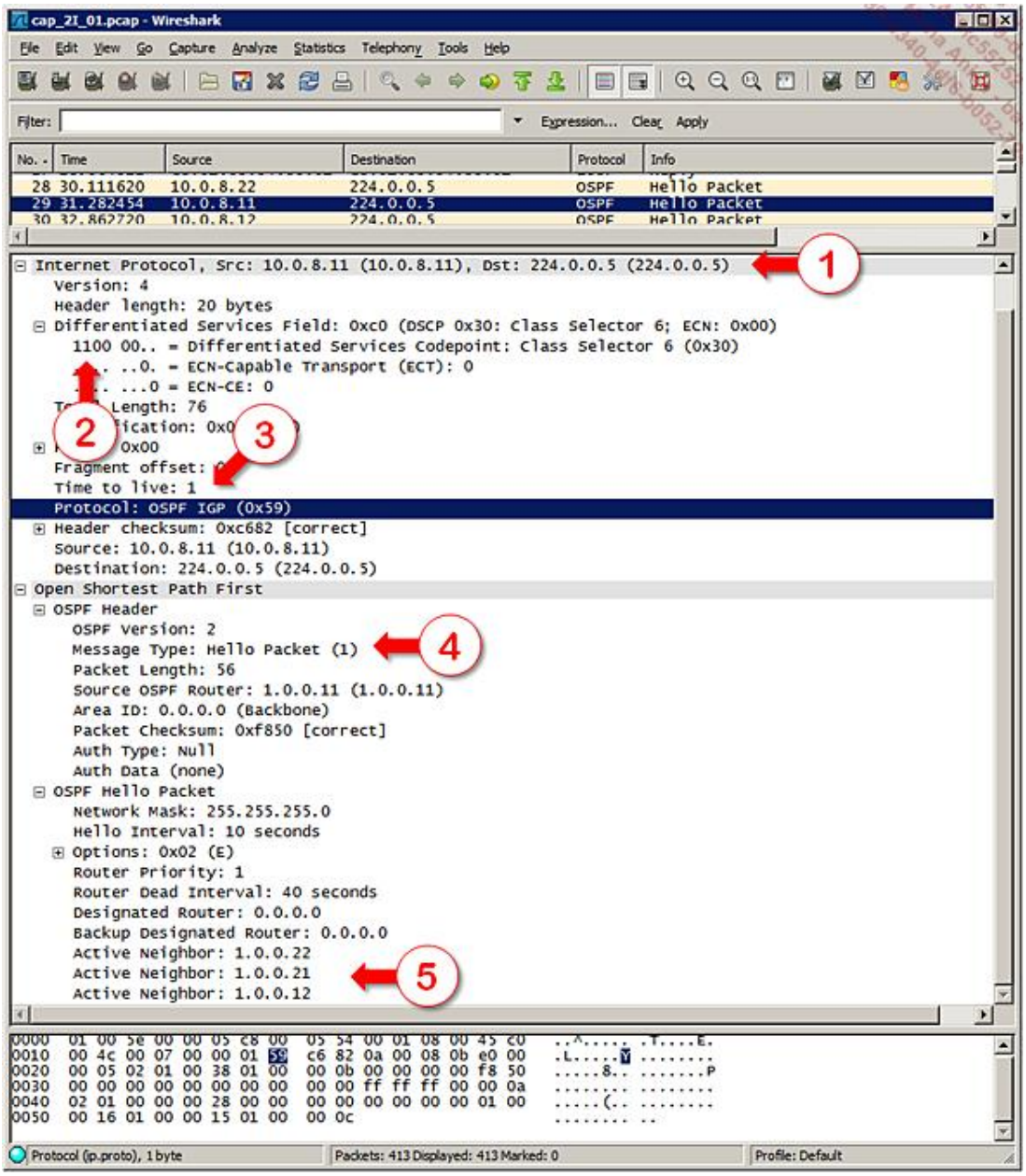
OSPF est directement encapsulé dans IP, il est alors identifié par le numéro de protocole 89. OSPF utilise cinq types de paquets identifiés par un champ Type de 1 à 5 dans l'en-tête OSPF :



Le format de l'en-tête est le même quel que soit le type de paquet. Le message Hello porte la liste des voisins connus. Le paquet DBD porte tout ou partie des LSAs, identifiés par leurs en-têtes, qui composent la LSD du routeur. Le paquet « *LS Request* » contient la liste des identifiants de LSA réclamés. Le paquet « *LS Update* » contient une liste de LSAs et c'est le seul à porter une information complète (non abrégée). Enfin, le paquet « *LS Acknowledgment* » contient une liste de LSAs reçus, identifiés par leurs en-têtes comme dans le cas du paquet DBD.

Les paquets OSPF sont toujours échangés entre voisins. Puisque deux voisins sont toujours directement connectés, un paquet OSPF n'est jamais « routé » au-delà du réseau dont il est issu.

Observez l'extrait de capture ci-dessous :



1. OSPF est transporté par IP (0x59 = 89).

2. Wireshark interprète le champ Type de service de l'en-tête IP de façon à y lire une valeur DSCP (*Differentiated Services Code point*). Pour mémoire, cette valeur définit le comportement du routeur dans l'acheminement des paquets. Mais les valeurs attribuées au DSCP n'interfèrent pas avec les anciennes de priorité **11x** contenues dans les trois premiers bits du champ et qui correspondent à la gestion du réseau. Ainsi, la valeur **110** lue dans la capture (étiquette 2) correspond en réalité à la priorité « *Internetwork Control* », ce qui confère aux paquets OSPF une priorité élevée.

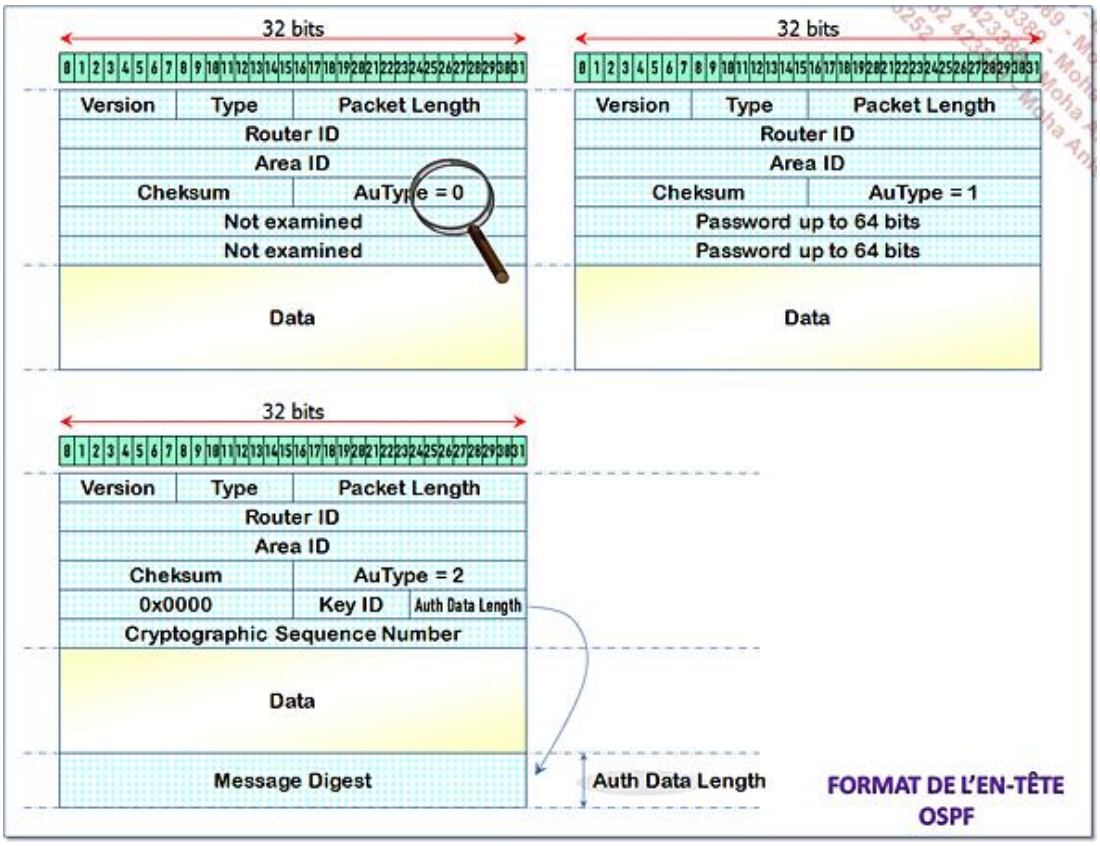
3. Observez également la valeur 1 attribuée au champ TTL : puisqu'un paquet OSPF n'a jamais besoin d'être acheminé au-delà du réseau dont il est issu, on est ainsi assuré que jamais le paquet n'effectuera un saut supplémentaire (« ceinture et bretelles ! »).

4. Le type 1 identifie un paquet OSPF Hello.

5. Parmi les informations portées par le message Hello, la liste des voisins connus est l'information essentielle.

a. L'en-tête OSPF

Tout paquet OSPF débute par un en-tête commun de 24 octets :



- Version : 2 pour la version OSPF version 2 définie dans le RFC 2328, 3 pour OSPF version 3 définie dans le RFC 5340.
- Type : les cinq types de paquets sont :
    - Type 1 : Hello ;
    - Type 2 : DBD - « *Database Description* » ;
    - Type 3 : « *Link State Request* » ;
    - Type 4 : « *Link State Update* » ;
    - Type 5 : « *Link State Acknowledgment* ».
- « *Packet Length* » : longueur du paquet OSPF en incluant l'en-tête, exprimée en octets.
- « *Router ID* » : RID du routeur émetteur de ce message.
- « *Area ID* » : identifiant de l'aire dont est issu ce message. Quand le paquet est émis sur un « *virtuallink* », l'identifiant d'aire est 0.0.0.0 car les liens virtuels sont considérés appartenir à l'aire backbone.
- « *Checksum* » : somme de contrôle qui porte sur l'ensemble du paquet OSPF en incluant l'en-tête mais en excluant les 64 bits du champ Authentification.
- « *AuType* » : définit le mode d'authentification parmi les trois modes possibles :
    - AuType = 0 → Pas d'authentification ;
    - AuType = 1 → Authentification par mot de passe en clair ;

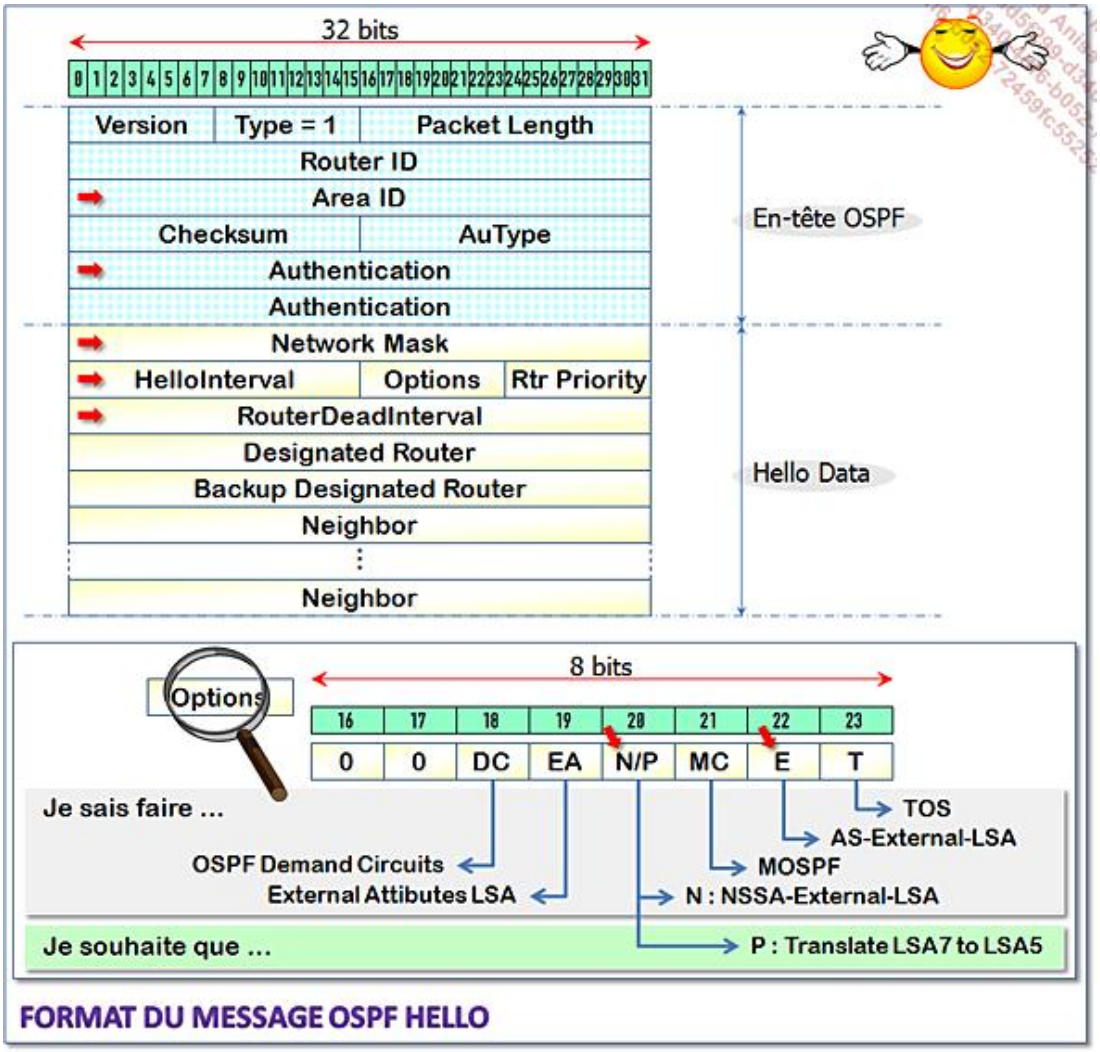
- AuType = 2 → Authentification cryptographique fondée sur MD5.

Le type d'authentification ainsi que les données utiles à l'authentification choisie sont configurables par interface. Le type 0 entraîne que les échanges OSPF ne sont pas authentifiés. Le champ « *Authentication* » sur 64 bits n'est pas examiné et peut contenir n'importe quelle valeur. Le type 1 entraîne une authentification par mot de passe simple partagé sur l'ensemble des routeurs concernés. Il ne fournit pas de protection contre les attaques mais simplement l'assurance que des routeurs étrangers ne se joignent pas au domaine par inadvertance. Le type 2 utilise une clé secrète partagée sur l'ensemble des routeurs concernés. L'algorithme, de type MD5, utilise la clé pour produire un « *digest* » de chaque paquet OSPF, digest ajouté à la fin du paquet. La protection est bonne car la clé secrète n'est jamais envoyée sur le réseau.

Quand « *AuType* » = 2...

- o « *KeyID* » : identifie l'algorithme ainsi que la clé secrète utilisée.
- o « *Auth Data Length* » : spécifie la longueur du « digest » placé à la fin du paquet.
- o « *Cryptographic Sequence Number* » : numéro de séquence non décroissant est inclus dans chaque paquet afin d'éviter les attaques par répétition.

b. Le message Hello



Le message Hello est utilisé afin de découvrir dynamiquement le voisinage, établir des relations de voisinage et de proximité. Observez les différents champs qui ont permis au routeur qui reçoit ce message d'alimenter les éléments associés à l'entrée (l'enregistrement) de la base de données de voisinage. Les champs marqués par la petite flèche sont ceux qui doivent correspondre entre ce routeur et le routeur voisin pour qu'ils puissent prétendre devenir voisins au sens OSPF.

- « *Network Mask* » : masque associé à l'adresse dans la configuration IP de l'interface qui envoie le paquet. Si ce masque est différent du masque configuré sur l'interface qui reçoit, le message est supprimé, ce afin d'être certain que deux routeurs ne deviennent voisins que s'ils appartiennent au même réseau.

- « *HelloInterval* » : période, exprimée en secondes, qui sépare deux émissions du message Hello sur une interface.

- « *Options* » : présent dans les messages Hello et DBD ainsi que dans chaque LSA. Un routeur utilise le champ Options pour annoncer ses capacités parmi un ensemble de capacités facultatives. La seule capacité décrite dans le RFC 2328 est celle qui concerne les aires de bout (« stub »). Pour mémoire, une aire de bout est une aire reliée à, au plus, une autre aire. En temps normal, les LSAs externes à l'AS sont inondées dans tout le système autonome. Mais cela alourdit inutilement les LSD des routeurs appartenant à une aire de bout. Le routeur placé à la frontière de l'aire de bout peut avantageusement leur substituer une route par défaut et l'annoncer aux routeurs de l'aire de bout à l'aide de LSA de résumés (« *Summary-LSAs* »).

    - Bit N/P : ce bit a un double usage selon qu'il est observé dans un message Hello, il s'agit alors du bit N, ou observé dans l'en-tête d'un « *NSSA-External-LSA* », auquel cas il est le bit P (« *Propagate* »).

        - Bit N : à 1 indique que le routeur participe au processus d'inondation des « *NSSA-External-LSA* », à 0 indique que le routeur les refuse. Quand le bit N est à 1, le bit E doit être à 0. Deux interfaces de capacités N différentes ne peuvent devenir proches.

        - Bit P : à 1 demande à l'ABR de convertir le LSA de type 7 de l'aire NSSA en LSA de type 5 dans l'aire 0.

    - Bit « E » : à 1 quand le routeur est en mesure d'accepter des LSAs externes à l'AS, devrait être à 0 dans tous les LSAs issus d'une aire de bout. Dans les messages Hello, ce bit indique la capacité de l'interface à envoyer et recevoir des LSAs de type 5 (AS-external-LSA). Deux interfaces de capacités E différentes ne peuvent devenir proches.

- « *Router priority* » : utilisée par le mécanisme d'élection des routeurs désignés DR et BDR sur les réseaux à diffusion ainsi que sur les réseaux sans diffusion fonctionnant en mode NBMA.

- « *RouterDeadInterval* » : si les messages Hello que le routeur s'attend à recevoir ne lui parviennent plus pendant ce délai, le voisin est considéré comme perdu (il n'est donc plus voisin). Le RFC suggère d'utiliser un délai « *RouterDeadInterval* » égal à 4 x « *HelloInterval* », c'est-à-dire 40 secondes par défaut.

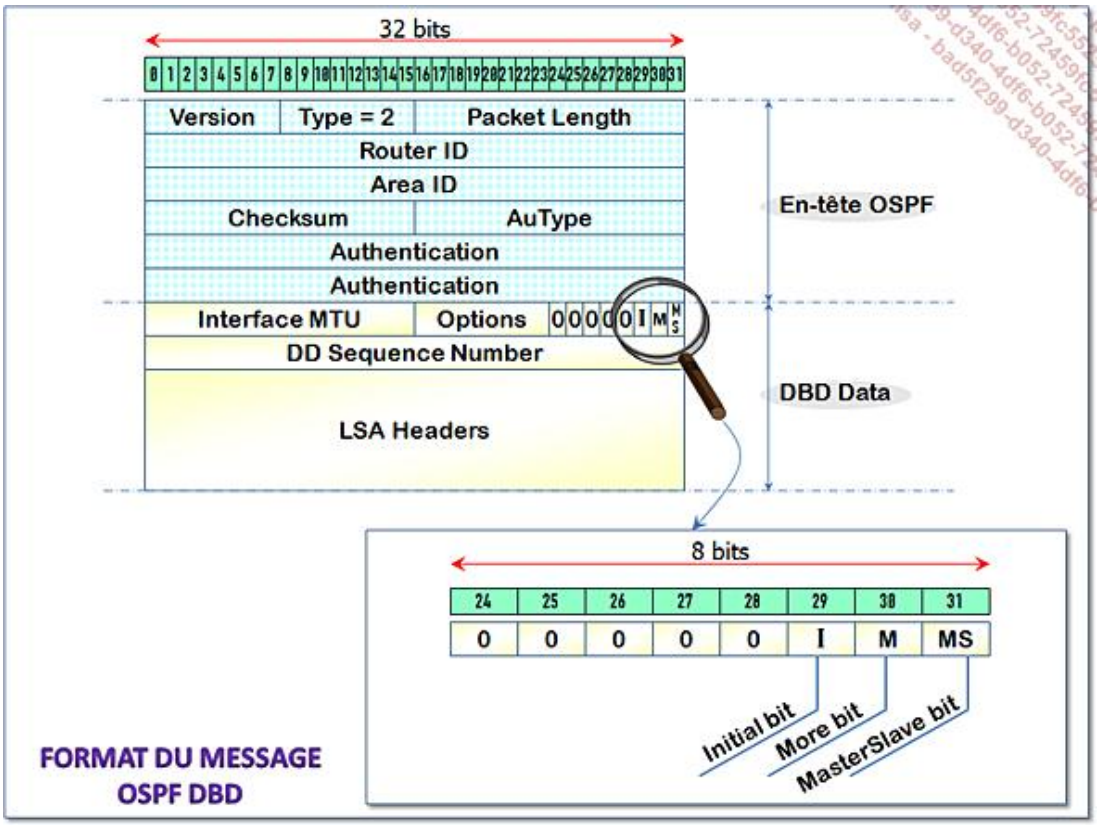
- « *Designated Router* » : adresse IP (et non RID) de l'interface du DR sur le réseau de rattachement. À défaut de DR, la valeur reste à 0.0.0.0.

- « *Backup Designated Router* » : adresse IP (et non RID) de l'interface du BDR sur le réseau de rattachement. À défaut de BDR, la valeur reste à 0.0.0.0.

- « *Neighbor* » : autant de champs que nécessaire afin de préciser de façon exhaustive l'ensemble des voisins connus de ce routeur sur le réseau de rattachement.

c. Le paquet DBD

Le paquet DBD est utilisé pendant la construction d'une relation de voisinage (voir la section éponyme) afin de décrire les LSAs contenus dans la LSD. La description exhaustive peut nécessiter de nombreux paquets DBD et c'est pour cette raison qu'une procédure de type « polling » a été choisie. Dans une procédure de polling, un maître interroge l'esclave et lui dit : « Je souhaite te dire blablabla et toi, as-tu quelque chose à me dire ? ». Les messages DBD envoyés par le maître sont donc des messages d'invitation à émettre acquittés en retour par les messages de réponse DBD envoyés par l'esclave. La procédure fait correspondre un message d'invitation à émettre avec sa réponse à l'aide des numéros de séquence DBD.



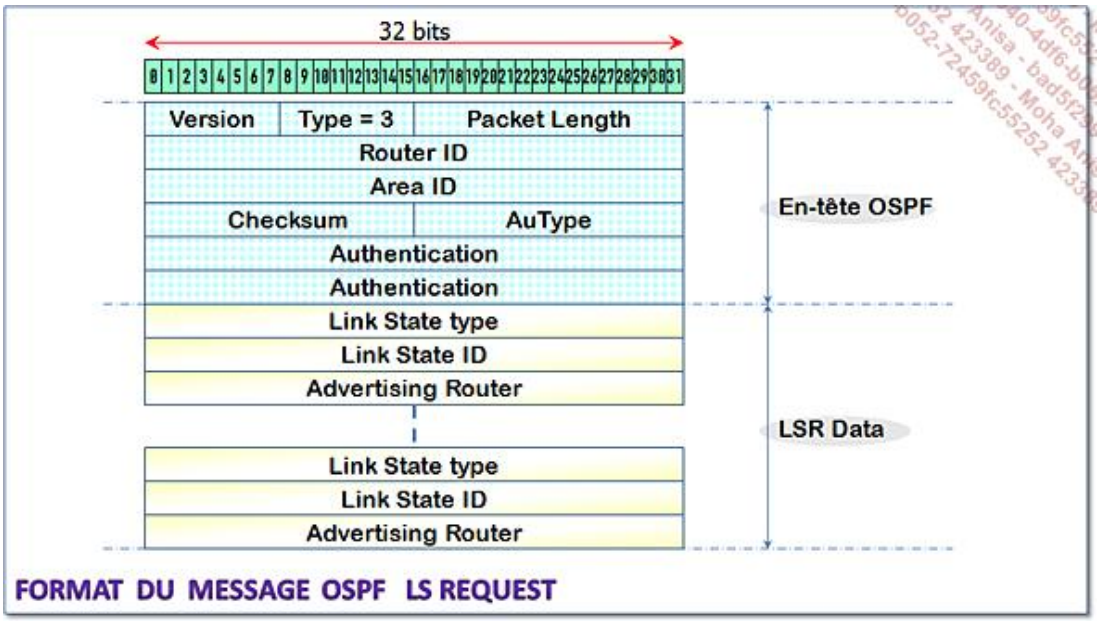
- « *Interface MTU* » : exprimée en octets, taille du plus grand datagramme IP qu'il est possible d'envoyer depuis cette interface sans fragmentation. Devrait être réglé à 0x0000 dans les paquets DBD lorsqu'ils sont émis sur un « *virtual link* ».
- « Options » : également présent dans le message Hello et donc déjà décrit.
- Les 5 bits suivants du prochain octet sont à 0 et suivis des trois bits :
  - « *Initial bit* » : n'est à 1 que dans le premier paquet DBD d'une séquence, qui correspond à l'état « *ExStart* » du diagramme d'états de la relation de voisinage.
  - « *More bit* » : d'autres paquets DBD suivent.
  - « *MasterSlave bit* » : désigne le maître dans la procédure de polling.
- « *DD Sequence Number* » : utilisé pour régler les échanges de DBD. Le maître génère un nouveau numéro de séquence quand il procède à une nouvelle invitation à émettre. Ce faisant, il acquitte réception du message DBD précédemment reçu. L'esclave acquitte réception du message du maître en faisant écho du numéro de séquence reçu dans le message DBD qu'il émet.

Suit une liste partielle ou complète de LSAs décrits à l'aide de leurs en-têtes. L'en-tête d'un LSA contient les informations suffisantes pour identifier sans équivoque à la fois le LSA et l'instance actuelle du LSA (telle qu'il est connu par le routeur émetteur de ce paquet DBD).

d. Le paquet Demande d'état de lien

Au vu des messages DBD reçus de son voisin pendant la phase de description de la LSD, le routeur a pu distinguer parmi ses LSAs, ceux pour lesquels le voisin connaît une version plus récente et qui nécessitent d'être mis à jour. Le routeur utilise pour ce faire un ou plusieurs paquets « Link State Request ».

Une instance de LSA est définie à l'aide des trois paramètres « *LS Sequence Number* », « *LS age* » et « *LS checksum* ». Pourtant, le paquet « *LS Request* » ne porte pas ces informations. C'est inutile car demander un LSA au voisin, c'est implicitement lui demander le LSA le plus récent connu par ce voisin. C'est donc le LSA qui est identifié dans la requête, et non une instance particulière de LSA, ce à l'aide des trois champs « *Link State type* », « *Link State ID* » et « *Advertising Router* ».

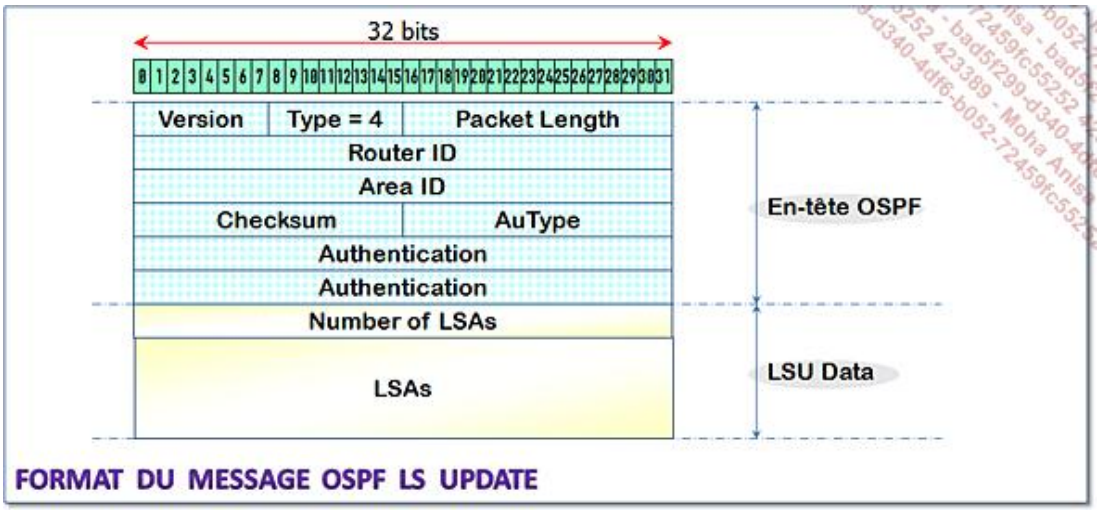


- « *Link State type* » : type de LSA. La liste ci-dessous reprend les types du RFC 2328, elle n'est donc pas exhaustive. Ils existent d'autres types qui sortent par contre trop du cadre de cet ouvrage :
    - Type 1 : « *router-LSA* ». Décrit les états collectés des interfaces du routeur.
    - Type 2 : « *network-LSA* ». Décrit l'ensemble des routeurs rattachés au réseau.
    - Type 3 ou 4 : « *summary-LSA* »
        - Type 3 : « *network summary-LSA* » : décrit des routes vers les réseaux.
        - Type 4 : « *ASBR summary-LSA* » : décrit des routes vers les routeurs de bordure de système autonome (ABR : « *AS boundary router* »).
    - Type 5 : « *AS-external-LSA* ». Générés par les routeurs situés à la frontière du système autonome, ils décrivent les chemins pour les destinations externes à l'AS. Un chemin par défaut pour le système autonome peut aussi être décrit.
- « *Link State ID* » : identifie le LSA, information extraite de l'en-tête du LSA.
- « *Advertising Router* » : identifiant du routeur qui a généré le LSA. Par exemple, dans le cas d'un LSA de réseau, ce champ est le RID du DR.

e. Le paquet Mise à jour d'état de lien

OSPF utilise les paquets « *LS Update* » dans deux circonstances : lorsqu'il faut répondre à un paquet « *LS Request* » ainsi que dans le processus d'inondation des LSAs. Dans ce cas précis, souvenons-nous qu'un paquet OSPF n'a jamais besoin d'être acheminé au-delà du réseau dont il est issu. C'est ce constat qui a permis d'ajuster la valeur TTL des paquets OSPF à 1. La conséquence est que parmi les LSAs reçus, un routeur voisin doit ré-encapsuler les LSAs appropriés dans de nouveaux paquets « *LS Update* » pour que le processus d'inondation puisse progresser.

Selon les cas, les paquets « LS update » sont diffusés vers l'adresse multicast 224.0.0.5 (qui signifie tous les routeurs OSPF, « *AllSPFRouters* ») ou vers l'adresse multicast 224.0.0.6 (qui signifie les deux routeurs DR et BDR, « *AllDRouters* »). Tout LSA diffusé doit être acquitté et peut l'être à l'aide de paquets « *LS Acknowledgment* ». Quand la retransmission d'un LSA est nécessaire, le LSA retransmis est envoyé vers l'adresse unicast du voisin nécessiteux.



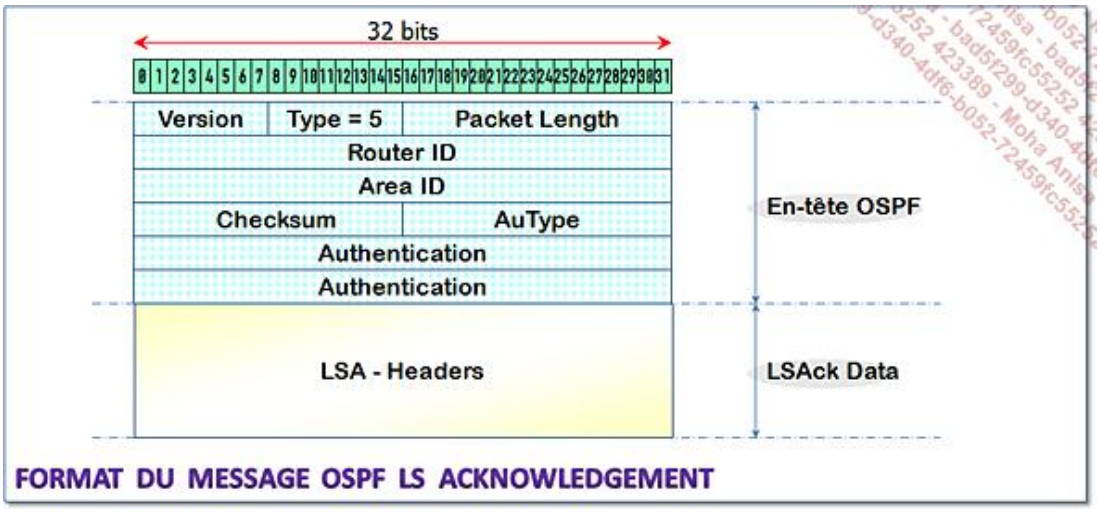
- « *Number of LSAs* » : nombre de LSAs inclus dans ce paquet.

- « *LSAs* » : enfin une information complète, puisqu'il s'agit des LSAs entiers dans le format décrit ci-après. Un paquet « *LS Update* » peut porter plusieurs LSAs complets à concurrence de la taille maximale de paquet autorisée sur le lien.

f. Le paquet d'acquittement d'état de lien

Dans le but de rendre fiable le mécanisme d'inondation des LSAs, OSPF exige que tout LSA reçu soit acquitté. Un paquet « LS Ack » peut acquitter de nombreux LSAs, chaque LSA acquitté est identifié par son en-tête. En final, le paquet « LS Ack » n'est rien de plus que l'assemblage d'un en-tête OSPF et des en-têtes de LSAs qu'il faut acquitter explicitement.

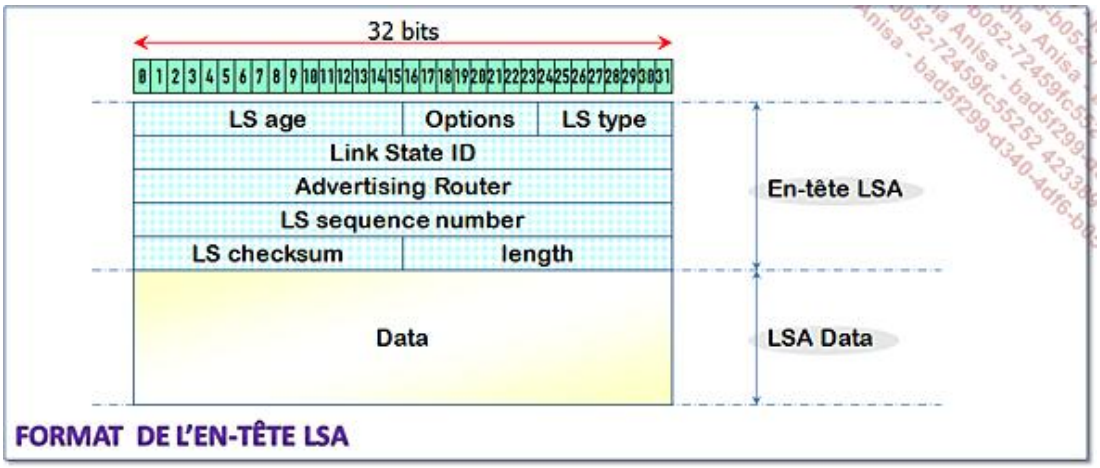
Un routeur n'a pas nécessairement besoin d'un paquet « *LS Ack* » pour acquitter un LSA reçu. OSPF améliore l'efficacité globale en permettant à un routeur d'acquitter implicitement un LSA en répétant ce même LSA dans un message « LS Update » en partance vers le voisin à l'origine du LSA, dans un mode que l'on pourrait appeler « Echoplex ». Le paquet « *LS Acknowledgement* » est donc utilisé pour réaliser un acquittement explicite quand l'acquittement implicite n'est pas possible.



g. Format des LSAs

Format de l'en-tête LSA

Tout LSA débute par cet en-tête de 20 octets qui contient notamment les informations suffisantes pour identifier le LSA sans équivoque, ce sont les trois champs « *LS type* », « *Link State ID* » et « *Advertising Router* ». De plus, puisque plusieurs instances d'un même LSA peuvent coexister dans le domaine d'acheminement, il est nécessaire de pouvoir déterminer l'instance la plus récente. OSPF utilise pour ce faire les trois champs « *LS age* », « *LS sequence number* » et « *LS checksum* ».



- « *LS age* » : exprimé en secondes, temps écoulé depuis la création du LSA. Au fur et à mesure que le LSA progresse de routeur en routeur dans le processus d'inondation, l'âge est incrémenté de la valeur « *InfTransDelay* » (noté « *Transmit Delay* » par l'ILC) associée à l'interface de chaque interface qu'il traverse. Bien sûr, l'âge s'incrémente également du temps passé dans la LSD.

- « Options » : champ déjà décrit, voir le message Hello. Dans le cas de l'en-tête LSA, le champ « Options » décrit les capacités supportées par le fragment de domaine OSPF objet du LSA.

- « *LS type* » : type de LSA, déjà décrit, voir le paquet « *LS Request* ». En résumé :

    - Type 1 : « *router-LSA* » ;

    - Type 2 : « *network-LSA* » ;

    - Type 3: « *summary-LSA* » (réseau IP) ;

    - Type 4 : « *summary-LSA* » (ASBR) ;

    - Type 5 : « *AS external LSA* ».

- « *LS ID* » : identifie le LSA et donc le fragment du domaine OSPF décrit par le LSA. Fonction du « *LS type* ». Exemple : dans le cas d'un LSA de réseau, « *LS ID* » est réglé à l'adresse IP de l'interface du DR.

- « *Advertising Router* » : identifiant du routeur qui a généré le LSA. Par exemple, dans le cas d'un LSA de réseau, ce champ est le RID du DR.

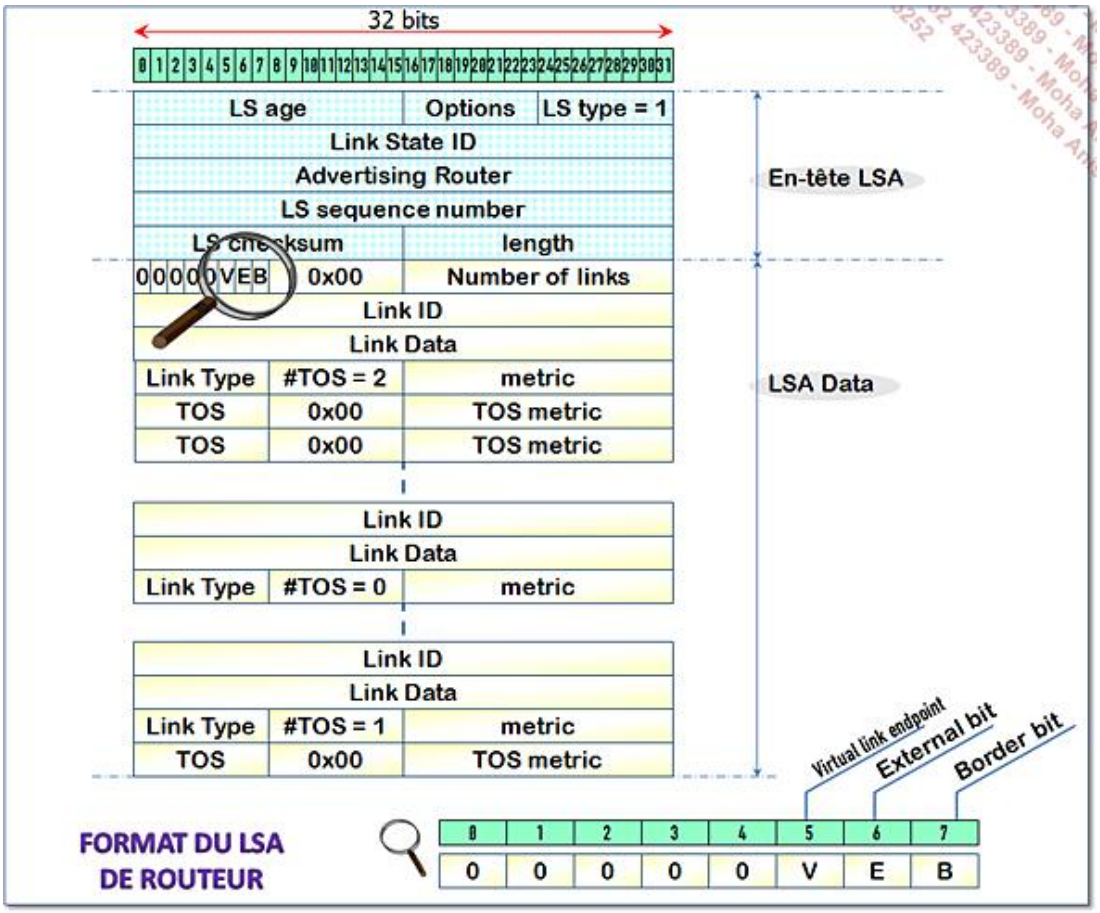
- « *LS sequence number* » : incrémentée chaque fois qu'une nouvelle instance du LSA est générée. Ce nombre participe à l'identification de l'instance la plus récente.

- « *LS checksum* » : checksum calculé selon l'algorithme de Fletcher décrit dans le RFC 1146. Pour mémoire, l'algorithme classique consiste à additionner simplement tous les mots de 16 bits contenus dans le message dont il faut vérifier l'intégrité. Un tel algorithme est incapable de détecter par exemple l'insertion ou la suppression d'un octet à 0, ce que sait faire en revanche l'algorithme de Fletcher. La somme de contrôle porte sur l'ensemble du LSA à l'exclusion du champ âge. En effet, inclure ce champ aurait contraint à recalculer la somme à chaque incrémentation de l'âge.

- « *length* » : exprimée en octets, longueur du LSA.

Le LSA de routeur

Chaque routeur d'une aire génère un LSA de routeur qui décrit l'état ainsi que le coût des liens du routeur (c'est-à-dire de ses interfaces) avec l'aire. Tous les liens du routeur doivent être décrits par un seul LSA. Pour mémoire, les LSAs sont inondés à l'intérieur d'une seule aire, celle dont ils sont issus. Une commande **show ip database router** permet de visualiser l'ensemble des LSAs de routeur dans une LSD.



Dans l'en-tête du LSA :

- « *Link State ID* » : RID du routeur qui a généré ce LSA.

Dans les données du LSA :

- « V » : « *Virtual linkendpoint* », ce bit est à 1 lorsque le routeur est une extrémité de liaison virtuelle.
- « E » : « *External bit* », à 1 quand le routeur qui a généré le LSA est un ASBR (*Autonomous System Boundary Router*) c'est-à-dire un routeur de bordure du système autonome.
- « B » : « *Border bit* », à 1 quand le routeur qui a généré le LSA est un ABR (*Area Border Router*).
- « *Number of links* » : nombre de liens décrits dans le LSA. Tous les liens du routeur avec l'aire doivent être présents.

Les champs qui suivent décrivent chaque lien et apparaissent donc autant de fois qu'il y a de liens à décrire (trois fois dans l'exemple ci-dessus). Le champ « *Link Type* » détermine la façon dont il faut interpréter les champs « *Link ID* » et « *Link Data* », c'est pourquoi il faut débuter la description par ce champ :

- « Link Type » : fournit une brève description du lien. Les valeurs possibles sont les suivantes :
  - Type 1 : connexion point à point avec un autre routeur ;
  - Type 2 : connexion à un réseau de transit ;
  - Type 3 : connexion à un réseau d'extrémité ;
  - Type 4 : liaison virtuelle.

- « *Link ID* » : identifiant du lien. Identifie l'objet auquel le lien connecte. Quand le lien connecte à un routeur ou à un réseau de transit, deux objets qui génèrent un LSA (dans le cas du réseau de transit, c'est le rôle du DR que de générer ce LSA), « *Link ID* » est égal au champ « Link State ID » du LSA généré par ce voisin. Ce chaînage est mis à profit par le processus OSPF pour construire la table de routage, « *Link ID* » fournissant la clé afin d'effectuer la recherche du LSA du voisin dans la LSD. Les valeurs possibles sont fonction de « Link Type » et résumées dans le tableau ci-dessous :

- « *Link Data* » : valeur également dépendante de « Link Type » selon le tableau ci-après :

« Link Type »	**Description**	**Interprétation de « Link ID »**	**Interprétation de « Link Data »**
1	Connexion point à point avec un autre routeur	RID du voisin	Adresse IP sur le réseau point à point de l'interface du routeur qui a généré le LSA.
2	Connexion à un réseau de transit	Adresse IP de l'interface du DR	Adresse IP sur le réseau de transit de l'interface du routeur qui a généré le LSA.
3	Connexion à un réseau d'extrémité	Adresse IP du réseau	Masque de réseau.
4	Liaison virtuelle	RID du voisin	Adresse IP de l'interface du routeur qui a généré le LSA.

Tableau 2 - Interprétation de "Link ID " et "Link Data"

- « #TOS » : spécifie le nombre de métriques TOS (*Type of Service*) listées dans ce lien en excluant la métrique de lien requise, spécifiée dans le champ « metric ». Le RFC 2328 ne prévoit pas le support de ces fonctionnalités TOS. Par conséquent, ce champ ainsi que les champs « *TOS* » et « *TOS metric* » ne sont présents que pour permettre une rétrocompatibilité avec des implémentations OSPF plus anciennes. Quand aucune valeur « *TOS metric* » n'est associée au lien, ce champ reste à 0x00.

- « *metric* » : coût du lien et donc de l'interface (relire si nécessaire la rubrique coût de l'interface présentée à la section Structure des données de l'interface dans ce chapitre).

À l'intérieur de la description d'un lien, les champs suivants n'apparaissent que si #TOS est différent de 0. L'exemple fourni dans l'illustration ci-dessus montre un LSA à trois liens dont le premier comporte deux métriques TOS, le second n'en comporte pas et le troisième en comporte une.

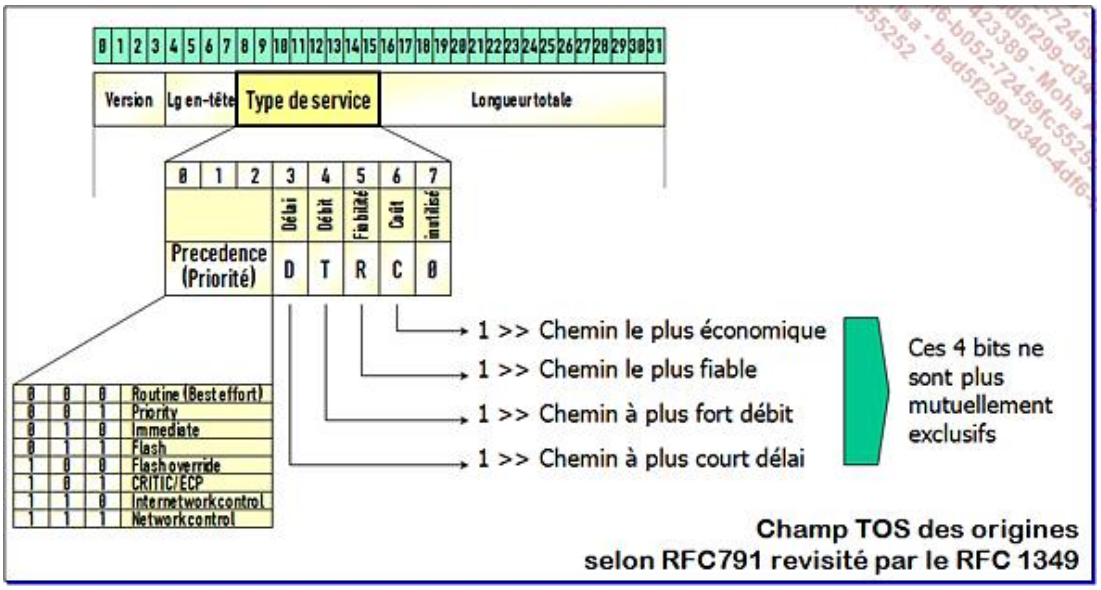
- « *TOS* » : spécifie le type de service auquel la métrique fournie dans le champ « *TOS metric* » s'applique selon le tableau ci-dessous :

Valeur de TOS définie dans le RFC 1349				**Description**	**Codage OSPF**
$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$		
0	0	0	0	Service normal.	0
0	0	0	1	Minimise le coût monétaire.	2
0	0	1	0	Maximise la fiabilité.	4
0	0	1	1		6
0	1	0	0	Maximise le débit.	8
0	1	0	1		10
0	1	1	0		12
0	1	1	1		14

1	0	0	0	Minimise le délai.	16
1	0	0	1		18
1	0	1	0		20
1	0	1	1		22
1	1	0	0		24
1	1	0	1		26
1	1	1	0		28
1	1	1	1		30

Tableau 3 - Valeurs de TOS du RFC 1349

Pour aider à resituer le contexte, voici un extrait de l'en-tête du datagramme IP, tel qu'il avait cours quand le RFC 1349 était d'actualité :

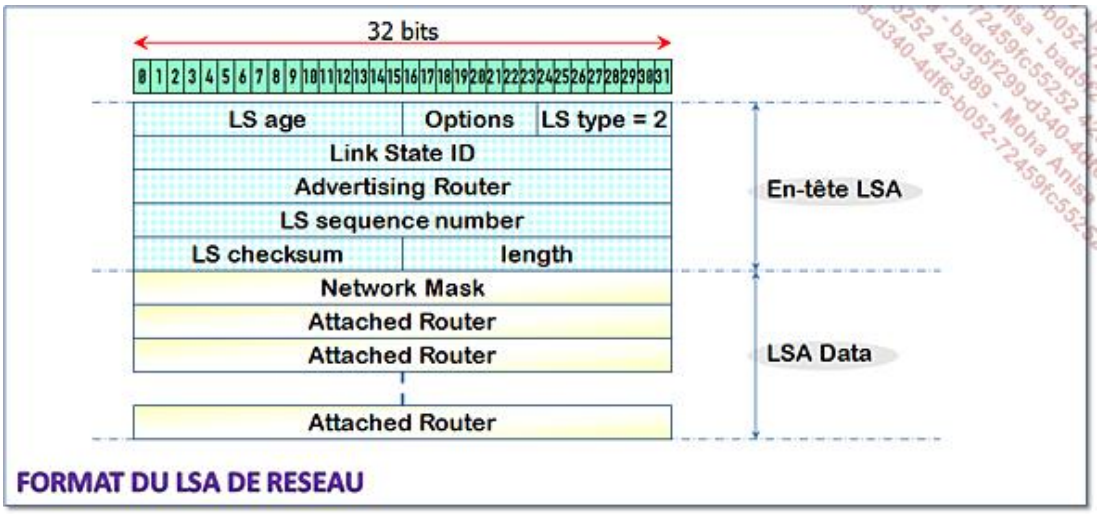


Le RFC 791 considérait les 3 bits du champ TOS comme mutuellement exclusifs, le bit « *Minimizemonetarycost* » n'existait pas encore. Le RFC 1349 ajoute le bit « *Minimizemonetarycost* » et ne considère plus ces bits comme une collection de bits mais comme un entier exprimé sur 4 bits. La conséquence est que ces bits ne sont plus mutuellement exclusifs. La valeur codée dans le champ TOS résulte de la réaffectation des poids binaires 2, 4, 8, 16 aux 4 bits, ce qui peut s'expliquer par le fait que le bit de poids le plus faible n'était pas utilisé et restait à 0.

- « *TOS metric* » : métrique que le processus OSPF doit associer à la valeur « *TOS* ».

Le LSA de réseau

Chaque réseau à diffusion ou en mode NBMA est représenté par un LSA de réseau généré par le DR. Ce LSA annonce le réseau ainsi que tous les routeurs connectés à ce réseau, DR inclus. De la même façon que les LSAs de routeur, les LSAs de réseau sont inondés à l'intérieur d'une seule aire, celle dont ils sont issus.



Dans l'en-tête du LSA :

- « *Link State ID* » : adresse IP de l'interface du DR sur le réseau décrit par ce LSA.

Dans les données du LSA :

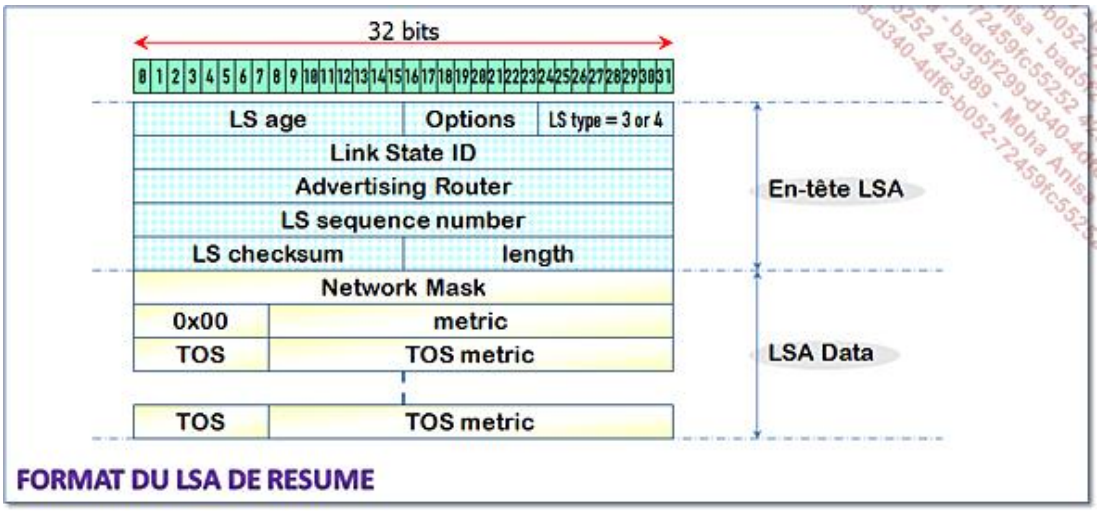
- « *Network Mask* » : masque de réseau sur le réseau décrit par ce LSA.
- « *Attached Router* » : liste des RID de routeurs pleinement adjacents au DR sur le réseau décrit par ce LSA. Le DR est inclus dans la liste.

Les LSA de résumé

Il en existe deux types, tous deux générés exclusivement par les ABR et inondés dans une seule aire :

- Type 3 : « *Network Summary-LSA* », décrit les routes vers les destinations en dehors de l'aire (en incluant les routes par défaut), mais à l'intérieur du système autonome.
- Type 4 : « *ASBR Summary-LSA* », décrit les routes vers les ASBR situés en dehors de l'aire.

Les deux types ont le même format :



En dehors du champ type, la seule différence provient de l'interprétation qu'il faut faire du champ « Link State ID » :

- « *Link State ID* » : pour le type 3, le champ contient l'adresse IP du réseau annoncé. Quand le type est 4, le champ contient le RID de l'ASBR annoncé.

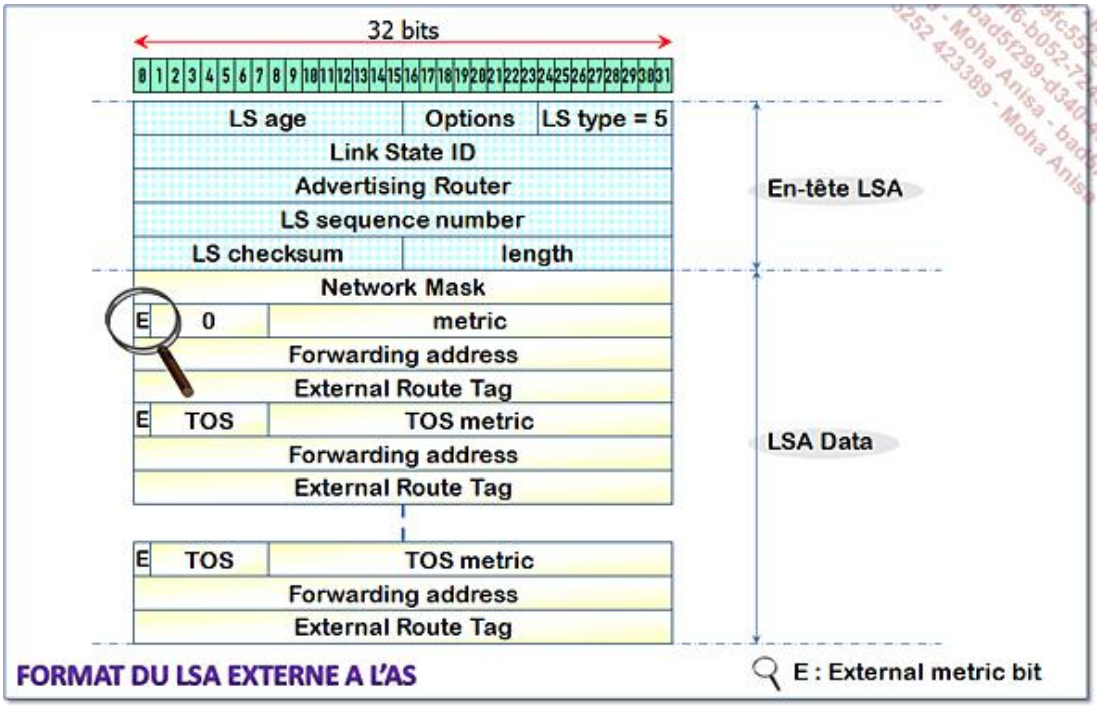
- « *Network Mask* » : pour le type 3, le champ contient le masque du réseau annoncé. Quand le type est 4, le champ n'a pas de signification et contient 0.0.0.0.

Quand le type 3 est utilisé pour annoncer une route par défaut, les champs « *Link State ID* » et « *Network Mask* » sont tous deux complétés avec 0.0.0.0.

- « *metric* » : coût de la route annoncée.
- « *TOS* », « *TOS metric* » : déjà décrit, voir LSA de routeur.

Les LSA externes à l'AS

Les « *AS-external-LSA* » sont générés exclusivement par les ASBR et décrivent les destinations externes au système autonome ou des routes externes qui peuvent être utilisées comme routes par défaut. Ces LSAs sont inondés dans tout le système autonome à l'exclusion des aires de bout. Ce sont donc les seuls LSAs dont l'inondation ne concerne pas une aire et une seule.



- « *Link State ID* » : pour le type 5, le champ contient l'adresse IP du réseau annoncé.
- « *Network Mask* » : pour le type 5, le champ contient le masque du réseau annoncé.

Quand le type 5 est utilisé pour annoncer une route par défaut, les champs « *Link State ID* » et « *Network Mask* » sont tous deux complétés avec 0.0.0.0.

- « E » ou « *External Metric bit* » : spécifie le type de métrique associée à la destination annoncée. Quand une route externe est redistribuée dans le système autonome, c'est le routeur frontière ASBR qui « fait rentrer la route » dans le système autonome et qui est en charge de lui attribuer un coût :
  - « E » = 0, la métrique qu'il faut utiliser pour cette route est de type 1 ce qui signifie que le coût pour atteindre la destination externe doit être calculé en sommant le coût annoncé par l'ASBR dans le LSA et le coût pour atteindre l'ASBR.
  - « E » = 1, la métrique qu'il faut utiliser pour cette route est de type 2 ce qui signifie que le coût pour atteindre la destination externe est calculé en ignorant le coût pour atteindre l'ASBR et en ne prenant en compte que le coût annoncé par l'ASBR dans le LSA.

- C’est le mot-clé **metric-type** ajouté à la commande **redistribute** qui permet à l’administrateur de configurer le type choisi. Ce réglage prend de l’importance quand plusieurs routeurs frontières constituent des alternatives vers la destination en question.

- « *metric* » : coût vers la destination annoncée tel qu’il est réglé sur le routeur ASBR. Puisque la destination est probablement connue de l’ASBR à l’aide d’un protocole de routage différent, c’est l’administrateur qui configure la redistribution des routes apprises par ce protocole de routage vers OSPF, la métrique qu’il convient d’appliquer est réglée à ce moment de la configuration.

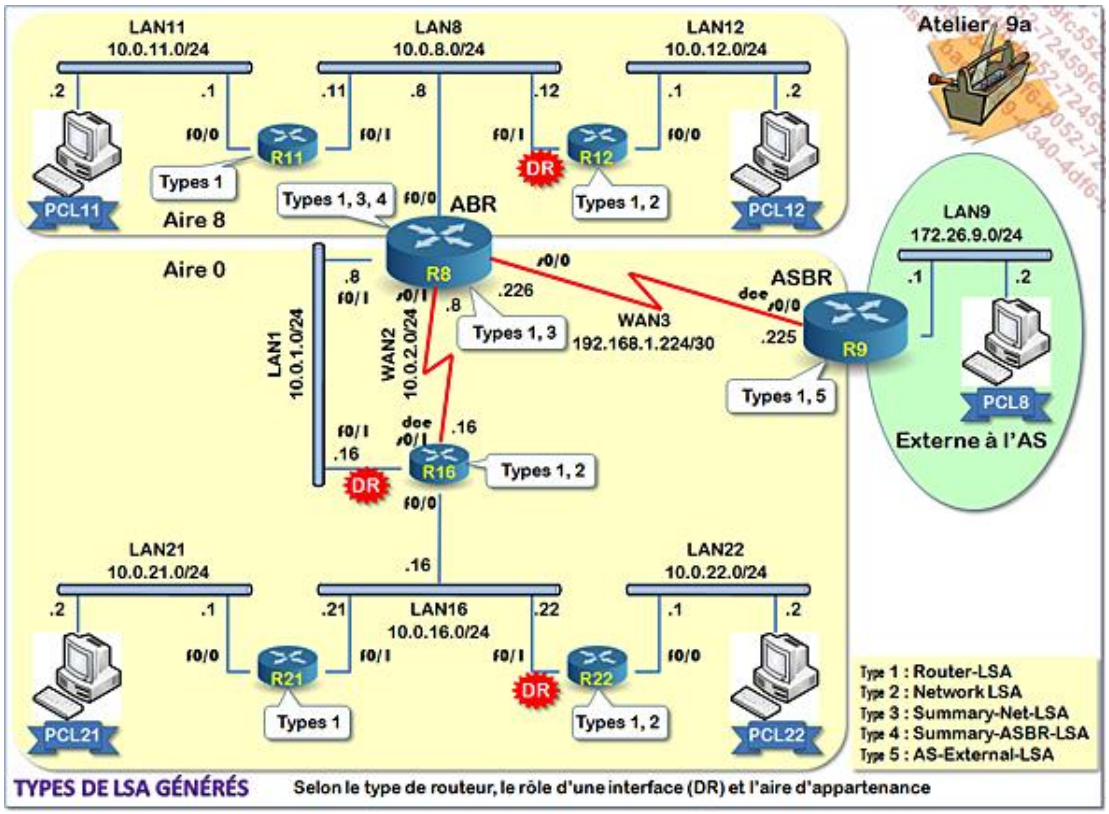
- « *Forwarding address* » : le trafic vers la destination annoncée doit être transmis à cette adresse. Quand le champ est réglé à 0.0.0.0, les paquets doivent être transmis à l’ASBR lui-même.

- « *External Route Tag* » : étiquette de 32 bits associée à la route externe. Ce champ n’est pas normalisé par le RFC 2328 et n’est pas utilisé par OSPF. Il permet l’échange d’informations entre routeurs frontière du système autonome.

- Les champs qui suivent sont à nouveau des champs TOS optionnels. Ces champs sont identiques à ceux déjà décrits (voir LSA de routeur) à l’exception de la métrique TOS qui cette fois est associée à son propre bit de type de métrique, sa propre adresse de transmission et sa propre étiquette.

Configuration OSPF

L'illustration de la topologie a déjà été fournie mais la voici à nouveau afin d'éviter d'avoir à feuilleter trop fréquemment l'ouvrage :



1. Configuration de base

Hors OSPF, la configuration des sept routeurs est la suivante :

R8

```
hostname R8
enable secret 5 ccna
interface FastEthernet0/0
 ip address 10.0.8.8 255.255.255.0
interface Serial0/0
ip address 192.168.1.226
255.255.255.252
interface FastEthernet0/1
 ip address 10.0.1.8 255.255.255.0
interface Serial0/1
ip address 10.0.2.8 255.255.255.0
line con 0
 exec-timeout 0 0
 logging synchronous
end
```

R16

```
hostname R16
enable secret 5 ccna
interface FastEthernet0/0
 ip address 10.0.16.16
255.255.255.0
interface FastEthernet0/1
 ip address 10.0.1.16 255.255.255.0
```

```
interface Serial0/1
ip address 10.0.2.16255.255.255.0
 clock rate 64000
line con 0
 exec-timeout 0 0
 logging synchronous
end
```

R11

```
hostname R11
enable secret 5 ccna
interface FastEthernet0/0
 ip address 10.0.11.1 255.255.255.0
interface FastEthernet0/1
 ip address 10.0.8.11 255.255.255.0
line con 0
 exec-timeout 0 0
 logging synchronous
end
```

R12

```
hostname R12
enable secret 5 ccna
interface FastEthernet0/0
 ip address 10.0.12.1 255.255.255.0
interface FastEthernet0/1
 ip address 10.0.8.12 255.255.255.0
line con 0
 exec-timeout 0 0
 logging synchronous
end
```

R21

```
hostname R21
enable secret 5 ccna
interface FastEthernet0/0
 ip address 10.0.21.1 255.255.255.0
interface FastEthernet0/1
 ip address 10.0.16.21 255.255.255.0
line con 0
 exec-timeout 0 0
 logging synchronous
end
```

R22

```
hostname R22
enable secret 5 ccna
interface FastEthernet0/0
 ip address 10.0.22.1 255.255.255.0
interface FastEthernet0/1
 ip address 10.0.16.22 255.255.255.0
line con 0
 exec-timeout 0 0
 logging synchronous
end
```

R9

```
hostname R9
enable secret 5 ccna
interface FastEthernet0/0
 ip address 172.26.9.1 255.255.255.0
interface Serial0/0
ip address 192.168.1.225 255.255.255.252
 clock rate 64000
```

```
line con 0
 exec-timeout 0 0
 logging synchronous
end
```

a. Identifiant du routeur

Relire si nécessaire la terminologie RID en début de ce chapitre. L'IOS offre une première possibilité qui consiste à fixer directement le RID d'un routeur à l'aide de la commande **router-id** en configuration de routeur :

```
R8(config)#router ospf 1
R8(config-router)#router-id 1.0.0.8
```

Si cette possibilité n'a pas été mise à profit, OSPF qui a vraiment besoin d'identifier tout routeur dérive un identifiant d'une adresse de loopback, à défaut d'une adresse d'interface. Une seconde solution consiste donc, sur chacun des routeurs OSPF, à créer une interface de loopback puis à lui affecter une adresse IP, adresse qui deviendra le RID du routeur :

R8

```
interface Loopback0
 ip address 1.0.0.8 255.255.255.255
```

R16

```
interface Loopback0
 ip address 1.0.0.16 255.255.255.255
```

R11

```
interface Loopback0
 ip address 1.0.0.11 255.255.255.255
```

R12

```
interface Loopback0
 ip address 1.0.0.12 255.255.255.0
```

R21

```
interface Loopback0
 ip address 1.0.0.21 255.255.255.0
```

R22

```
interface Loopback0
 ip address 1.0.0.22 255.255.255.255
```

R9

```
interface Loopback0
 ip address 1.0.0.9 255.255.255.0
```

b. Création du processus OSPF

Les premières étapes d'une configuration OSPF ressemblent à celles déjà expérimentées avec les protocoles de routage précédents. Il faut créer un processus OSPF puis l'avertir des réseaux que le protocole devra prendre en compte. Une différence cependant : OSPF est un protocole qui permet de hiérarchiser la topologie en deux niveaux à l'aide des aires. Ceci s'opère à l'aide de l'argument **area**. Le processus OSPF est démarré de la même façon que le processus EIGRP et nécessite un numéro de processus. Pour mémoire, EIGRP demandait lui un numéro de système autonome encore qu'en réalité ce ne soit jamais qu'un numéro de processus déguisé. Simplement, le processus EIGRP d'un routeur communique avec les autres routeurs EIGRP à la condition d'être configuré avec un même numéro de système autonome. Rien de tout cela avec OSPF, le numéro de processus n'a qu'une signification locale et n'a pas besoin d'être identique sur l'ensemble des routeurs du domaine OSPF. Mais dans un souci de clarté ou de cohérence, l'administrateur qui n'a rien d'un farfelu choisira le même numéro de processus pour tous les routeurs :

```
R8(config)#router ospf ?
```

```
<1-65535>  Process ID
R8(config)#router ospf 1
R8(config-router)#
```

Le numéro attribué au processus OSPF doit appartenir à l'espace [1-65535]. Rien n'interdit de configurer plusieurs processus OSPF sur un même routeur mais pourquoi faudrait-il le faire ? D'autant que chaque processus activé va entraîner la création puis l'entretien de l'ensemble des structures de données dont a besoin OSPF.

c. La commande network

La seconde étape consiste à activer le traitement des mises à jour sur les interfaces adéquates à l'aide de la commande **network** en configuration de routeur. Ce faisant, on crée aussi les aires de la topologie :

```
Router(config-router)# network @IP masque_générique area area_id
```

L'ensemble @IP/masque générique définit au choix, une adresse unique, une plage d'adresses contiguës ou enfin un ensemble d'adresses composé de plusieurs plages non contiguës. En effet, à l'inverse du masque réseau, la disposition des bits du masque générique n'a pas à être continue. Chaque bit à 1 dans le masque signifie que le bit correspondant de l'adresse doit être ignoré. Le tableau ci-dessous reprend les deux premières étapes, création de processus OSPF puis renseignement des interfaces et création des aires pour les sept routeurs de la topologie :

R8

```
router ospf 1
 log-adjacency-changes
 network 10.0.0.0 0.0.7.255 area 0
network 10.0.8.0 0.0.7.255 area 8
 network 192.168.1.224 0.0.0.3 area 0
```

R16

```
router ospf 1
 log-adjacency-changes
 network 10.0.0.0 0.0.7.255 area 0
 network 10.0.16.0 0.0.7.255 area 0
```

R11

```
router ospf 1
 log-adjacency-changes
 network 10.0.8.0 0.0.7.255 area 8
```

R12

```
router ospf 1
 log-adjacency-changes
 network 10.0.8.0 0.0.7.255 area 8
```

R21

```
router ospf 1
 log-adjacency-changes
 network 10.0.16.0 0.0.7.255 area 0
```

R22

```
router ospf 1
 log-adjacency-changes
 network 10.0.16.0 0.0.7.255 area 0
```

R9

```
router ospf 1
 log-adjacency-changes
 redistribute connected metric 10 subnets
 network 192.168.1.224 0.0.0.3 area 0
```

La commande **log-adjacency-changes** a été ajoutée de façon automatique par l'IOS. Ainsi, par défaut, des

messages SYSLOG avertissent de l'établissement ou de la disparition de relations de voisinage ou de proximité :

```
04:30:51: %OSPF-5-ADJCHG: Process 1, Nbr 1.0.0.11 on FastEthernet0/0 from LOADING
to FULL, Loading Done
```

Cette configuration n'est compréhensible qu'en disposant du plan d'adressage. Le lecteur attentif aura remarqué que ce plan a déjà servi de support dans l'ouvrage Cisco - Notions de base sur les réseaux dans la collection Certifications aux Editions ENI pour expliquer les fondements de l'adressage VLSM :

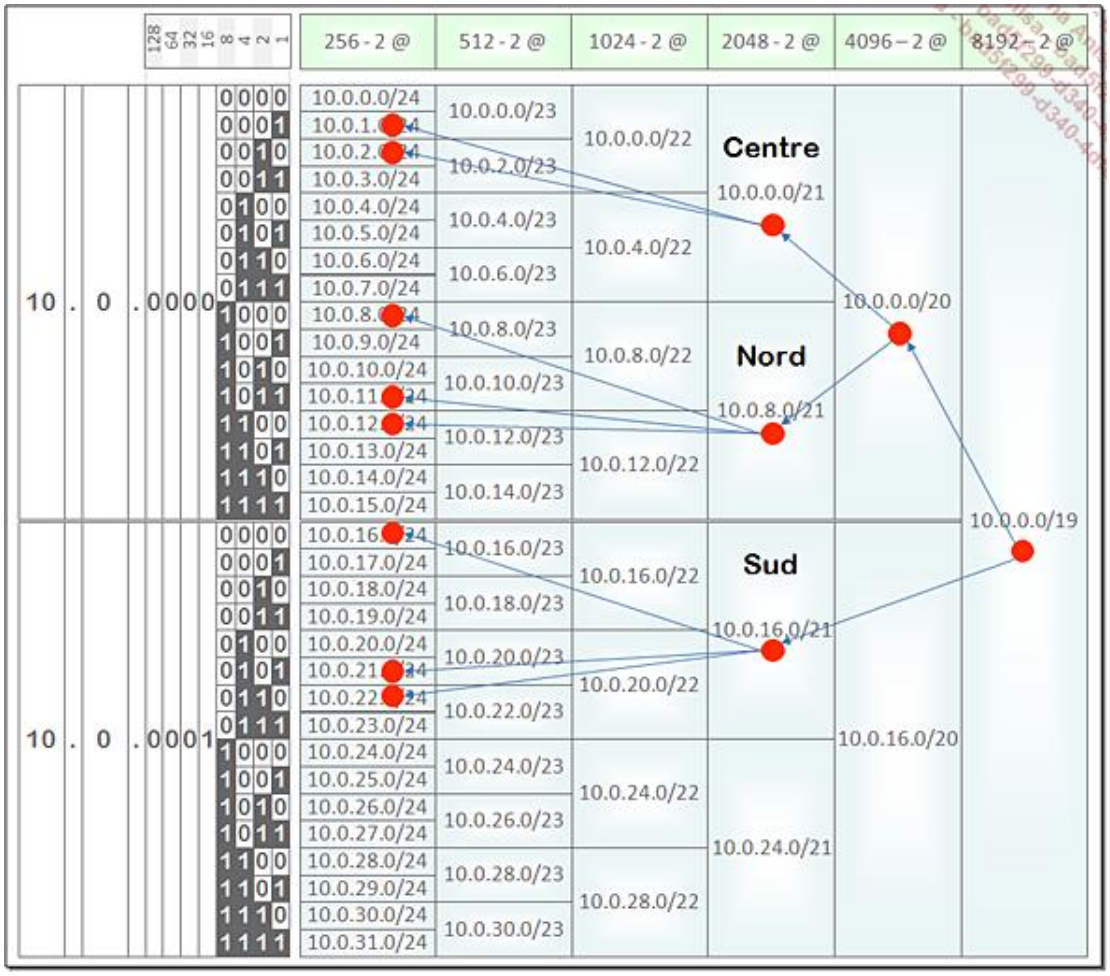


Tableau VLSM dichotomique 1

- R11, R12 appartiennent à l'aire 8 qui est au Nord de la topologie, aire qui se voit affectée l'étendue 10.0.8.0/21. La commande network correspondante est **network 10.0.8.0 0.0.7.255 area 8**.
- R21 et 22 appartiennent à la zone Sud, partie de l'aire 0, qui se voit affectée l'étendue 10.0.16.0/21. La commande network correspondante est **network 10.0.16.0 0.0.7.255 area 0**.
- R8 est un ABR placé à la frontière des aires 8 et 0. De plus, il est connecté à l'ASBR. Ceci justifie les trois commandes network.
- Une seule commande network suffit pour R9 car LAN9 n'appartient pas au domaine OSPF.

Pour l'anecdote, on se souvient que l'identifiant d'aire est formaté comme une adresse IP :

```
R8(config)#router ospf 1
R8(config-router)#network 10.0.8.0 0.0.7.255 area ?
<0-4294967295>  OSPF area ID as a decimal value
  A.B.C.D   OSPF area ID in IP address format
R8(config-router)#network 10.0.8.0 0.0.7.255 area 0.0.0.8
R8(config-router)#^Z
R8#sh run
```

```
.........
router ospf 1
log-adjacency-changes
network 10.0.0.0 0.0.7.255 area 0
network 10.0.8.0 0.0.7.255 area 8
network 192.168.1.224 0.0.0.3 area 0
.........
R8#
```

Astuce : pour plus de commodité, il est possible de renseigner la commande network avec l'adresse IP affectée à l'interface et donc à répéter la commande network autant de fois qu'il y a d'interfaces participant au protocole OSPF. Le masque générique est dans ce cas 0.0.0.0 qui spécifie une adresse d'hôte.

d. Contrôle et dépannage

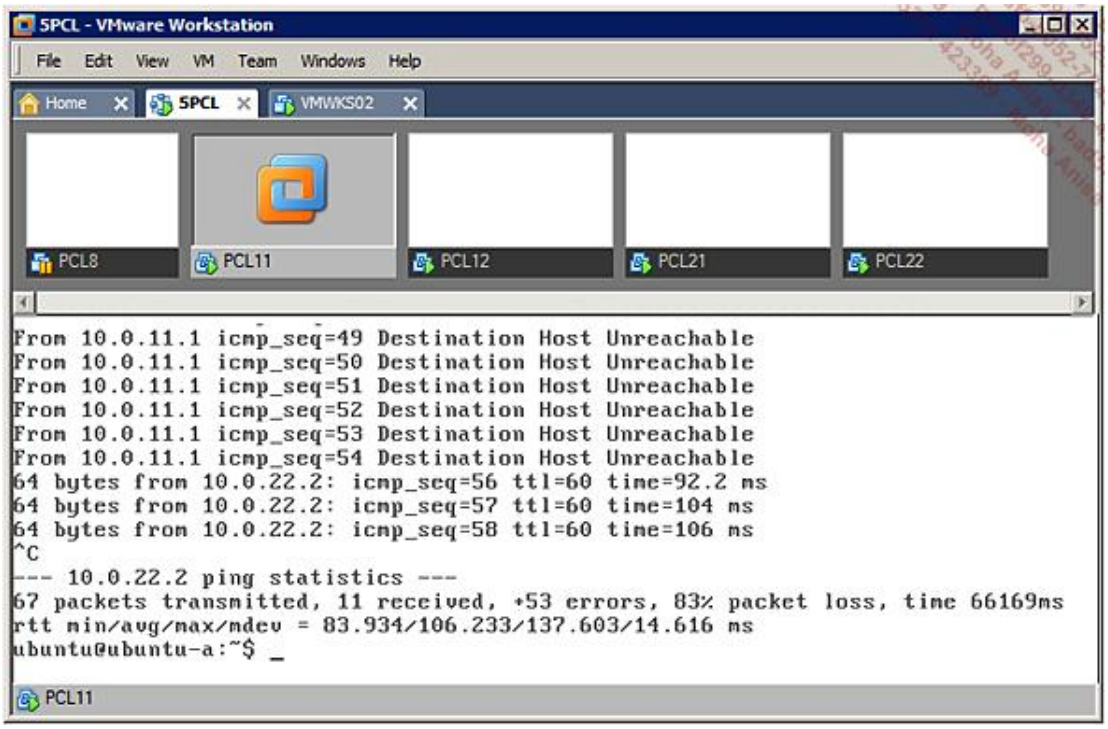
Sur chacun des routeurs, un certain nombre de commandes **show** renseignent sur le bon fonctionnement d'OSPF :

Commande	Commande abrégée	Usage
show ip interface brief	sh ip int br	État des interfaces, couche 1 et 2.
show ip ospf neighbor	sh ip o n	Liste des voisins avec pour chacun, son RID, l'état du diagramme d'états de l'interface, l'état du diagramme d'états de la relation de voisinage, l'adresse IP de l'interface du voisin sur le réseau de rattachement.
show ip ospf interface	sh ip oi	Pour chaque interface, une bonne part des informations OSPF : RID, aire d'appartenance, type de réseau, coût, réglage des temporisateurs...
show ip protocols	Sh ip pro	RID, type de routeurs, aires, types d'aires...
show ip route ospf	Sh ip ro os	Les routes apprises par OSPF.
Show ip route 10.0.16.1	Sh ip ro 10.0.16.1	Y-a-t-il une route dans la table de routage vers cette destination ?

Pour des cas plus difficiles, l'administrateur peut s'adjoindre les services de la commande **debug**. Ne pas oublier de commencer par une commande **undebug all** afin de pouvoir rappeler cette commande de façon simple depuis l'historique de commandes :

Commande	Commande abrégée	Usage
undebug all	u all	Faire cesser tout processus debug.
debug ip ospf adj	deb ip o adj	Le détail de l'établissement des relations de voisinage.
debug ip ospf events	deb ip o e	Informations sur chaque paquet du trafic d'acheminement OSPF.
debug ip ospf packet	deb ip o p	Contenu de chaque paquet du trafic d'acheminement OSPF.
debug ip ospf hello	deb ip o h	Informations sur le trafic de messages Hello.

Le test ultime : la commande **ping** avait été lancée depuis PCL11 vers PCL22 avant de démarrer les sept routeurs de l'atelier 9a, ne boudons pas notre plaisir :



2. Configuration avancée

a. Modification de la bande passante d'un lien

LAN1 qui relie R8 et R16 est hors service. WAN2 maintient la connectivité dans l'aire 0 mais quand l'administrateur entre une commande **show ip route** sur R8, il constate un problème :

```
R8#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
.........
O       10.0.16.0 [110/65] via 10.0.2.16, 00:01:40, Serial0/1
O       10.0.22.0 [110/66] via 10.0.2.16, 00:01:40, Serial0/1
O       10.0.21.0 [110/66] via 10.0.2.16, 00:04:05, Serial0/1
```

Le coût de la route vers 10.0.16.0 est 65. Puisque LAN16 est de type Fast Ethernet, l'interface f0/0 de R16 a un coût de 1. L'interface s0/0 de R8 a donc un coût de 65 - 1 = 64 ce qu'il vérifie à l'aide d'une commande **show int s0/0** :

```
R8#sh int s0/0
Serial0/0 is up, line protocol is up
  Hardware is PowerQUICC Serial
  Internet address is 192.168.1.226/30
  MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
```

La raison en est que la bande passante sur l'interface s0/0 est restée à la valeur par défaut, soit le débit du T1 américain. Il se trouve que les liens WAN entre R8 et R16 ainsi qu'entre R8 et R9 sont des liens à 64 Kbps. L'administrateur décide de mettre à jour l'ensemble des bandes passantes des liens « *serial* » :

R8

```
R8(config)#int s0/1
R8(config-if)#bandwidth 64
R8(config-if)#int s0/0
R8(config-if)#bandwidth 64
R8(config-if)#^Z
```

R16

```
R16(config)#int s0/1
R16(config-if)#bandwidth 64
```

```
R16(config-if)#^Z
```

R9

```
R9(config)#int s0/0
R9(config-if)#bandwidth 64
R9(config-if)#^Z
```

Une vérification de la table de routage confirme la prise en compte des nouvelles bandes passantes :

```
R8#sh ip route
.........
O       10.0.16.0 [110/1563] via 10.0.2.16, 00:00:00, Serial0/1
O       10.0.22.0 [110/1564] via 10.0.2.16, 00:00:00, Serial0/1
O       10.0.21.0 [110/1564] via 10.0.2.16, 00:00:00, Serial0/1
.........
```

b. Trucage des élections

L'administrateur souhaite que R16 soit DR, R21 BDR et R22 non éligible :

R16

```
R16(config)#int f0/0
R16(config-if)#ip ospf priority 10
R16(config-if)#^Z
R16#
```

R22

```
R22(config)#intf0/1
R22(config-if)#ip ospf priority 0
R22(config-if)#^Z
R22#
```

La configuration de R21 n'a pas été modifiée, sa priorité reste à 1, soit la priorité par défaut.

Le résultat n'est pas conforme à celui attendu parce que R16 a été démarré après que l'élection de R21 se soit déjà produite :

```
R21#sh ip ospf n

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.16          10   FULL/BDR        00:00:31    10.0.16.16      FastEthernet0/1
1.0.0.22           0   FULL/DROTHER    00:00:32    10.0.16.22      FastEthernet0/1
```

Un redémarrage de R21 rétablit le résultat souhaité :

```
R22#sh ip ospf n

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.16          10   FULL/DR         00:00:35    10.0.16.16      FastEthernet0/1
1.0.0.21           1   FULL/BDR        00:00:35    10.0.16.21      FastEthernet0/1
```

c. Agrégation de routes

Seul un ABR peut générer un LSA résumé de réseau. Il génère un tel LSA pour chaque destination accessible par lui en dehors de l'aire. Ainsi, R8 dans notre atelier OSPF génère six LSA résumés de réseau pour annoncer dans l'aire 8 les six destinations qu'il connaît dans l'aire 0, à savoir les réseaux 10.0.1.0/24, 10.0.2.0/24, 10.0.16.0/24, 10.0.21.0/24, 10.0.22.0/24 et 192.168.1.224/30 :

```
R11#sh ip ospf database database-summary

            OSPF Router with ID (1.0.0.11) (Process ID 1)

Area 8 database summary
  LSA Type      Count    Delete   Maxage
```

```
  Router        3        0        0
  Network       1        0        0
  Summary Net   6        0        0
  Summary ASBR  1        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Subtotal      11       0        0
.........
```

La commande **show ip ospf database summary** confirme les six destinations (extraits) :

```
R11#sh ip ospf database summary

            OSPF Router with ID (1.0.0.11) (Process ID 1)

                Summary Net Link States (Area 8)

...
  Link State ID: 10.0.1.0 (summary Network Number)
  Network Mask: /24
...
  Link State ID: 10.0.2.0 (summary Network Number)
  Network Mask: /24
...
  Link State ID: 10.0.16.0 (summary Network Number)
  Network Mask: /24
...
  Link State ID: 10.0.21.0 (summary Network Number)
  Network Mask: /24
...
  Link State ID: 10.0.22.0 (summary Network Number)
  Network Mask: /24
...
  Link State ID: 192.168.1.224 (summary Network Number)
Network Mask: /30
```

Il est possible de réduire le nombre de LSAs résumés de réseau annoncés par R8 dans l'aire 8 :

```
R8(config)#router ospf 1
R8(config-router)#area 0 range 10.0.0.0 255.255.248.0 ! Agréger Réseau Centre
R8(config-router)#area 0 range 10.0.16.0 255.255.248.0 ! Agréger Réseau Sud
R8(config-router)#^Z
```

Vérifions les effets sur la LSD de R11 :

```
R11#sh ip ospf database database-summary

            OSPF Router with ID (1.0.0.11) (Process ID 1)

Area 8 database summary
  LSA Type      Count    Delete   Maxage
  Router        3        0        0
  Network       1        0        0
  Summary Net   3        0        0
  Summary ASBR  1        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Subtotal      8        0        0
```

Les LSAs résumés de réseau sont passés de six à trois :

```
R11#sh ip ospf database summary

            OSPF Router with ID (1.0.0.11) (Process ID 1)

                Summary Net Link States (Area 8)
```

```
...
  Link State ID: 10.0.0.0 (summary Network Number)
  Network Mask: /21
...
  Link State ID: 10.0.16.0 (summary Network Number)
  Network Mask: /21
...
  Link State ID: 192.168.1.224 (summary Network Number)
Network Mask: /30
```

De la même façon, il est possible d'agréger les LSAs résumés de réseau correspondant aux trois destinations du réseau Nord, afin que R8 n'annonce qu'un seul LSA résumé de réseau dans l'aire 0 :

Avant l'intervention, l'ABR R8 annonce trois LSAs résumés de réseau dans l'aire 0 :

```
R16#sh ip ospf database database-summary

            OSPF Router with ID (1.0.0.16) (Process ID 1)

Area 0 database summary
  LSA Type      Count    Delete   Maxage
  Router        5        0        0
  Network       2        0        0
  Summary Net   3        0        0
  Summary ASBR  0        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Subtotal      10       0        0
```

Ces trois LSAs sont :

```
R16#sh ip ospf database summary

            OSPF Router with ID (1.0.0.16) (Process ID 1)

                Summary Net Link States (Area 0)
...
  Link State ID: 10.0.8.0 (summary Network Number)
  Network Mask: /24
...
  Link State ID: 10.0.11.0 (summary Network Number)
  Network Mask: /24
...
  Link State ID: 10.0.12.0 (summary Network Number)
  Network Mask: /24
```

Agrégons...

```
R8(config)#router ospf 1
R8(config-router)#area 8 range 10.0.8.0 255.255.248.0
R8(config-router)#^Z
```

Vérifions les effets sur la LSD de R16 :

```
R16#sh ip ospf database database-summary

            OSPF Router with ID (1.0.0.16) (Process ID 1)

Area 0 database summary
  LSA Type      Count    Delete   Maxage
  Router        5        0        0
  Network       2        0        0
  Summary Net   1        0        0
  Summary ASBR  0        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Subtotal      8        0        0
```

Les LSAs résumés de réseau sont passés de trois à un :

```
R16#sh ip ospf database summary

            OSPF Router with ID (1.0.0.16) (Process ID 1)

                Summary Net Link States (Area 0)
...
  Link State ID: 10.0.8.0 (summary Network Number)
  Network Mask: /21
```

CQFD.

d. Aire stub

R8

```
R8(config)#router ospf 1
R8(config-router)#area 8 stub
R8(config-router)#
04:14:06: %OSPF-5-ADJCHG:
Process 1, Nbr 1.0.0.11 on
FastEthernet0/0 from FULL to
DOWN, Neighbor Down: Adjacency
forced to reset
```

R11

```
R11(config)#router ospf 1
R11(config-router)#area 8 stub
R11(config-router)#
04:15:06: %OSPF-5-ADJCHG: Process 1, Nbr
1.0.0.12 on FastEthernet0/1 from FULL to
DOWN, Neighbor Down: Adjacency forced to
reset
```

R12

```
R12(config)#router ospf 1
R12(config-router)#area 8 stub
R12(config-router)#
```

Puisque l'aire 8 est désormais une aire de bout, l'ABR R8 génère un LSA résumé de réseau afin d'annoncer une route par défaut qui passe par lui, ce que confirme une commande **show ip route** :

```
R11#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 10.0.8.8 to network 0.0.0.0

     1.0.0.0/32 is subnetted, 1 subnets
C       1.0.0.11 is directly connected, Loopback0
     10.0.0.0/8 is variably subnetted, 5 subnets, 2 masks
C       10.0.11.0/24 is directly connected, FastEthernet0/0
C       10.0.8.0/24 is directly connected, FastEthernet0/1
O       10.0.12.0/24 [110/2] via 10.0.8.12, 00:00:49, FastEthernet0/1
O IA    10.0.0.0/21 [110/2] via 10.0.8.8, 00:00:49, FastEthernet0/1
O IA    10.0.16.0/21 [110/3] via 10.0.8.8, 00:00:49, FastEthernet0/1
     192.168.1.0/30 is subnetted, 1 subnets
O IA    192.168.1.224 [110/1563] via 10.0.8.8, 00:00:49, FastEthernet0/1
O*IA 0.0.0.0/0 [110/2] via 10.0.8.8, 00:00:49, FastEthernet0/1
```

e. Aire « totally-stubby »

On peut vouloir considérer que le seul LSA résumé de réseau utile est celui annonçant la route par défaut. Une intervention sur l'ABR R8 suffit pour transformer l'aire de bout en aire totalement de bout :

```
R8(config)#router ospf 1
R8(config-router)#no area 8 stub
R8(config-router)#area 8 stub ?
  no-summary  Do not send summary LSA into stub area
<cr>
R8(config-router)#area 8 stub no-summary
R8(config-router)#
```

Constatons les effets sur R12 cette fois :

```
R12#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 10.0.8.8 to network 0.0.0.0

     1.0.0.0/24 is subnetted, 1 subnets
C      1.0.0.0 is directly connected, Loopback0
     10.0.0.0/24 is subnetted, 3 subnets
O      10.0.11.0 [110/2] via 10.0.8.11, 00:00:05, FastEthernet0/1
C      10.0.8.0 is directly connected, FastEthernet0/1
C      10.0.12.0 is directly connected, FastEthernet0/0
O*IA 0.0.0.0/0 [110/2] via 10.0.8.8, 00:00:05, FastEthernet0/1
```

Ainsi, les routes vers 10.0.1.0/24, 10.0.2.0/24, 10.0.16.0/24, 10.0.21.0/24 et 10.0.22.0/24 ont disparu. La seule route inter-aires qui subsiste est la route par défaut. Ceci est cohérent avec le nombre de LSAs résumés de réseau présents en LSD :

```
R12#sh ip ospf database database-summary

            OSPF Router with ID (1.0.0.12) (Process ID 1)

Area 8 database summary
  LSA Type      Count    Delete   Maxage
  Router        3        0        0
  Network       1        0        0
  Summary Net   1        0        0
  Summary ASBR  0        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Subtotal      5        0        0
```

Le seul LSA résumé de réseau qui subsiste est :

```
R12#sh ip ospf database summary

            OSPF Router with ID (1.0.0.12) (Process ID 1)

                Summary Net Link States (Area 8)

...
  Link State ID: 0.0.0.0 (summary Network Number)
  Network Mask: /0
```

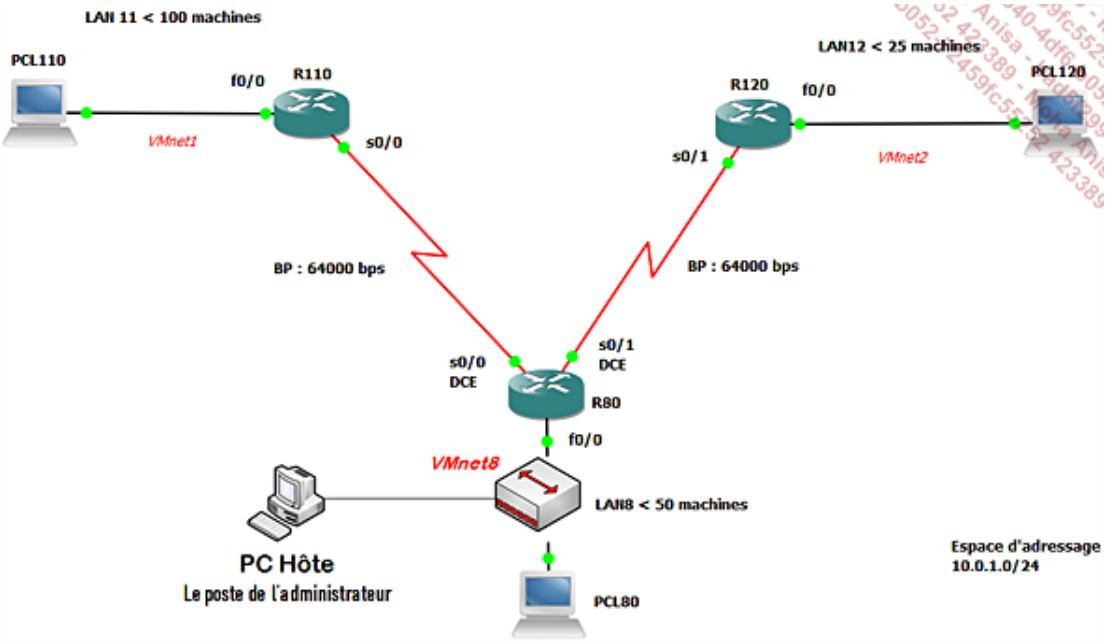
3. Conclusion

OSPF est plus difficile à mettre en œuvre que RIP mais il est beaucoup plus efficace dans les phases transitoires et ne crée pas de boucles de routage. De plus, cette caractéristique lui est naturelle alors qu'elle n'est obtenue qu'à coups d'artifices pour RIP. OSPF est fondé sur plusieurs sous-protocoles, dont un qui permet une inondation fiable du réseau. Les routeurs possèdent alors chacun une copie de la base de données topologiques du réseau, et peuvent calculer individuellement le plus court chemin pour aller vers l'ensemble des destinations.

Pour réduire la durée des calculs et surtout pour éviter un nouveau calcul des routes à chaque changement de configuration, OSPF offre la possibilité de découper le réseau en aires. Une aire principale appelée « ***backbone*** » relie toutes les autres aires. Les réseaux trouvés dans une aire donnée sont annoncés aux autres aires par les routeurs qui sont placés aux frontières d'aire.

Chapitre 1 - Le routage statique

Proposition de topologie



1. Tâche 1 : Conception du plan d'adressage

Tableau de documentation des adresses :

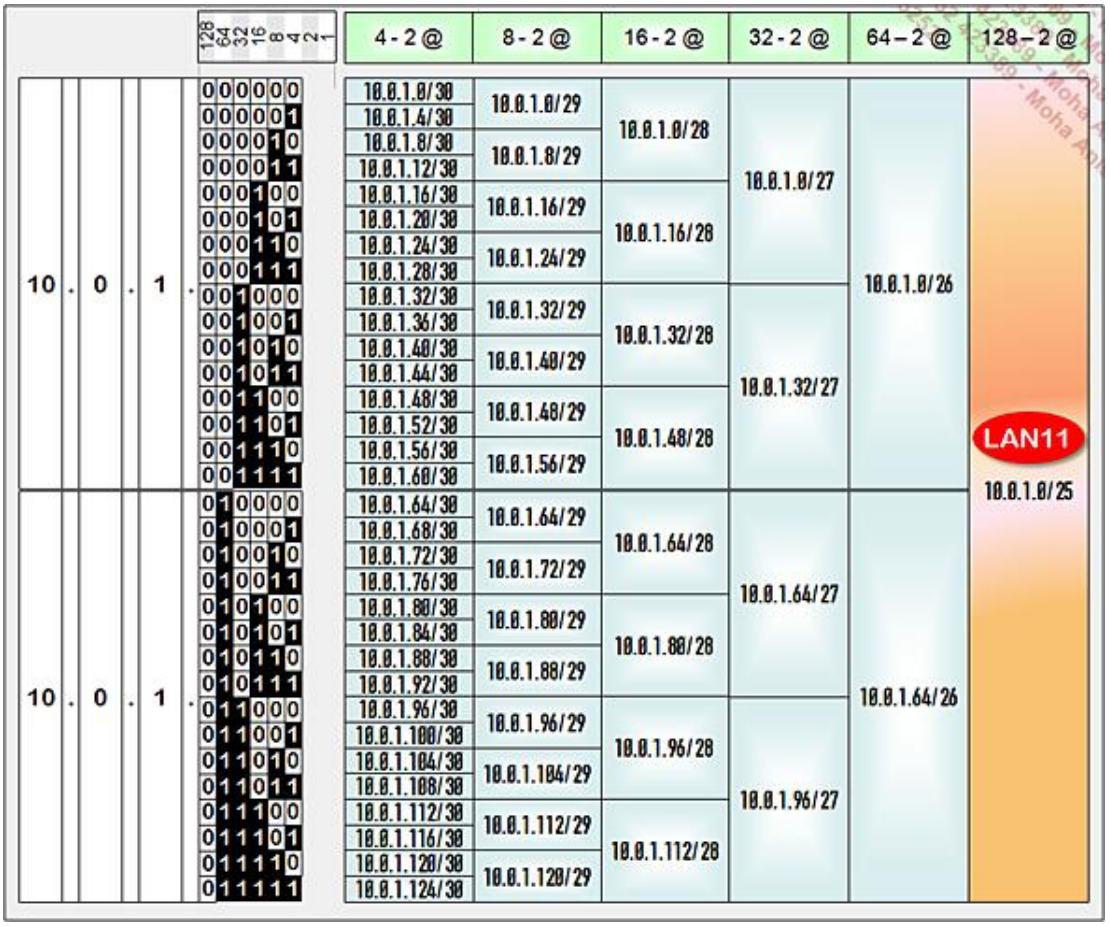
Affectation	Réseau	Première adresse	Dernière adresse	Adresse de diffusion
LAN11	10.0.1.0/25	10.0.1.1	10.0.1.126	10.0.1.127
LAN8	10.0.1.128/26	10.0.1.129	10.0.1.190	10.0.1.191
LAN12	10.0.1.192/27	10.0.1.193	10.0.1.222	10.0.1.223
WAN R80 - R110	10.0.1.224/30	10.0.1.225	10.0.1.226	10.0.1.227
WAN R80 - R120	10.0.1.228/30	10.0.1.229	10.0.1.230	10.0.1.231

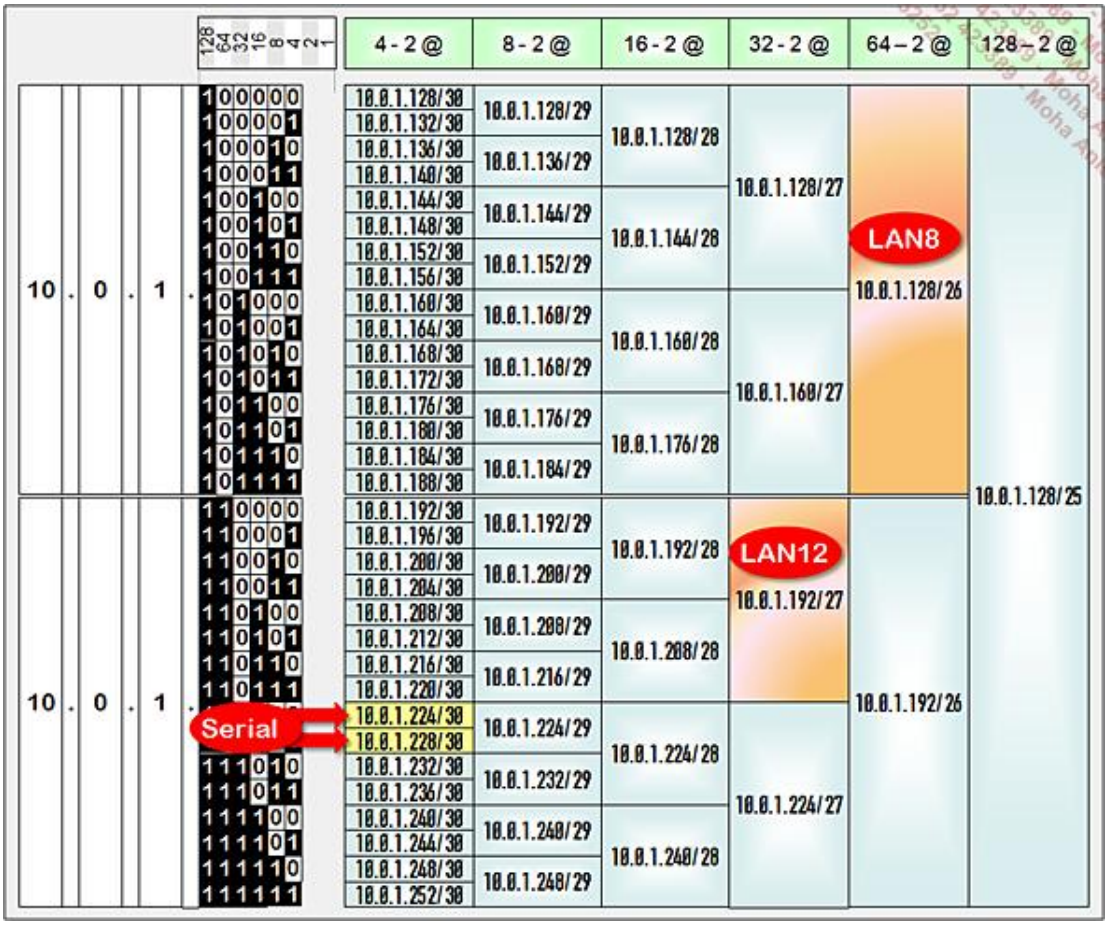
Tableau de documentation des interfaces :

Équipement	Interface	Adresse IP	Masque	Passerelle
R80	F0/0	10.0.1.129	255.255.255.192	Sans objet
	S0/0	10.0.1.226	255.255.255.252	Sans objet
	S0/1	10.0.1.230	255.255.255.252	Sans objet
R110	F0/0	10.0.1.1	255.255.255.128	Sans objet
	S0/0	10.0.1.225	255.255.255.252	Sans objet
R120	F0/0	10.0.1.193	255.255.255.224	Sans objet

	S0/1	10.0.1.229	255.255.255.252	Sans objet
PCL80	VMnet8	10.0.1.190	255.255.255.192	10.0.1.129
PCL110	VMnet1	10.0.1.126	255.255.255.128	10.0.1.1
PCL120	VMnet2	10.0.1.222	255.255.255.224	10.0.1.193
PC Hôte	VMnet8	10.0.1.189	255.255.255.192	10.0.1.129

L'administrateur a rempli ces tableaux sans difficulté grâce au précieux concours du tableau VLSM dichotomique qu'il a construit au préalable pour segmenter l'espace d'adressage 10.0.1.0/24 :





2. Introduction de routes statiques

a. Routes statiques avec adresse de saut suivant

Le tableau ci-dessous imagine quelques paquets parvenus à R110. Pour chacun des paquets, indiquez comment se comportera le routeur, va-t-il transférer ou rejeter ? Quelle est l'interface utilisée quand il transfère ?

Paquet	Adresse IP de destination	Rejet ou Transfert ?	Interface de sortie
1	10.0.1.229	Rejet	Sans objet
2	10.0.1.190	Transfert	S0/0
3	10.0.1.226	Transfert	S0/0
4	10.0.1.222	Rejet	Sans objet
5	10.0.1.126	Transfert	F0/0

b. Route statique avec interface de sortie

Cette seconde partie d'atelier est moins guidée à dessein. Le lecteur a certainement hâte d'être autonome. Alors, jetons-nous à l'eau...

- Sur la console de R80, tentez une commande ping vers PCL120.

Cette commande doit échouer, R80 ne disposant pas de route vers le réseau LAN12.

- Sur R80, ajoutez une route statique de type à interface de sortie vers le réseau LAN12.

```
R80#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R80(config)#ip route 10.0.1.192 255.255.255.224 s0/1
R80(config)#^Z
R80#
00:09:49: %SYS-5-CONFIG_I: Configured from console by console
R80#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
R80#
```

- Sur la console de R80, tentez à nouveau une commande ping vers PCL120.

```
R80#ping 10.0.1.222
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.222, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 32/66/108 ms
R80#
```

L'administrateur souhaite vérifier la connectivité de PCL80 à PCL120 sans se déplacer, c'est-à-dire en restant devant la console de R80.

- Sur la console de R80, utilisez une commande ping étendue afin de vérifier la connectivité de PCL80 à PCL120.

```
R80#ping 10.0.1.222 source 10.0.1.129

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.222, timeout is 2 seconds:
Packet sent with a source address of 10.0.1.129
.....
Success rate is 0 percent (0/5)
R80#
```

La commande échoue parce que R120 ne connaît pas de route vers le réseau LAN8.

- Sans vous déplacer sur la console de R120, en restant sur R80 donc, ouvrez une session Telnet sur R120 puis ajoutez une route statique par défaut de type à adresse de saut suivant. Fermez la session Telnet afin de revenir à la session console sur R80.

```
R80#telnet 10.0.1.229
Trying 10.0.1.229 ... Open

User Access Verification
Password:
R120>en
Password:
R120#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R120(config)#ip route 0.0.0.0 0.0.0.0 10.0.1.230
R120(config)#^Z
R120#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
R120#exit

[Connection to 10.0.1.229 closed by foreign host]
R80#
```

- Sur la console de R80, utilisez à nouveau une commande ping étendue afin de vérifier la connectivité de PCL80 à PCL120.

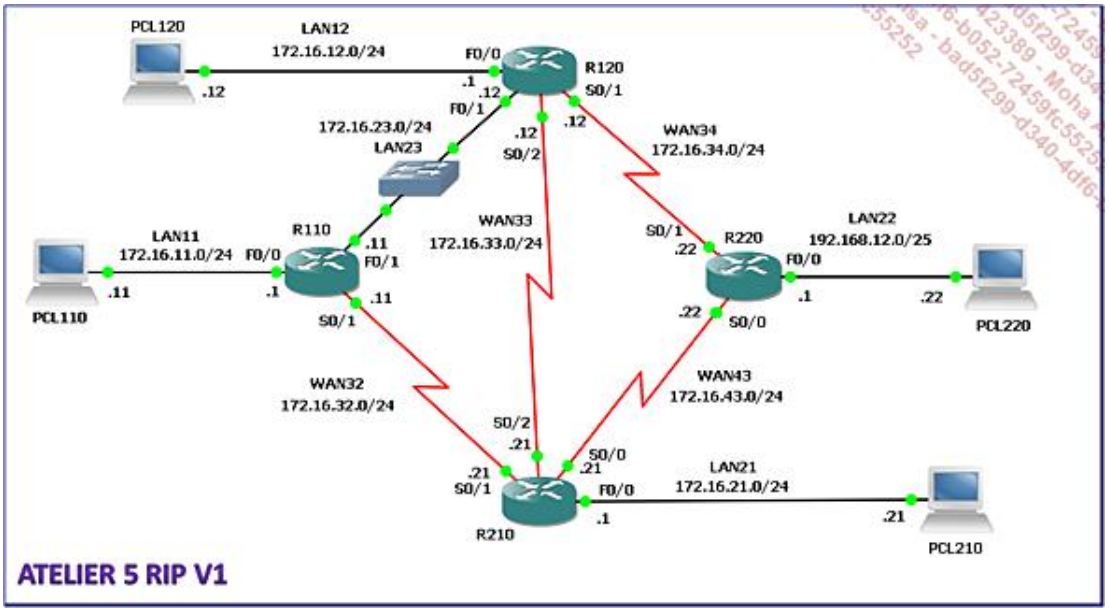
```
R80#ping 10.0.1.222 source 10.0.1.129

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.1.222, timeout is 2 seconds:
Packet sent with a source address of 10.0.1.129
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 48/426/1840 ms
R80#
```

Les requêtes ICMP obtiennent leurs réponses. Bravo, vous n'ignorez plus grand-chose du routage statique et avez encore progressé dans le maniement de l'interface ILC et des commandes de l'IOS.

Chapitre 2 - Le protocole de routage type DV RIPv1

1. Atelier : Mise en œuvre d'une configuration RIP

Vous voilà promu administrateur du réseau suivant :



Vous remarquez que le lien de secours WAN33 est une ressource dormante et souhaitez établir un partage de charge à coût égal sur trois routes entre LAN12 et LAN21. Les trois routes sont constituées des deux routes existantes auxquelles devrait s'additionner la route par WAN33.

Modifiez la configuration de R120 et de R210 en conséquence, sans remettre en cause le choix du protocole RIP.

L'idée consiste à mettre à profit la commande offset-list déjà en place sur chacun des deux routeurs R120 et R210. En ajoutant un offset de valeur 2 à la métrique annoncée des réseaux LAN12 et LAN21, le coût de la route par WAN33 atteignait 3, soit une valeur supérieure à celle des routes alternatives par R110 et R220. Mais en se contentant d'ajouter un offset de valeur 1, la route résultante par WAN33 voit son coût ramené à une valeur identique à celle des deux routes alternatives.

- Sur R120 :

```
R120(config)#router rip
R120(config-router)#offset-list 1 in 1 s0/2
R120(config-router)#^Z
R120#
```

- Sur R210 :

```
R210(config)#router rip
R210(config-router)#offset-list 1 in 1 s0/2
R210(config-router)#^Z
R210#
```

- Recette sur R120 :

```
R120#sh ip route
.........
Gateway of last resort is 172.16.34.22 to network 0.0.0.0

     172.16.0.0/24 is subnetted, 8 subnets
R       172.16.43.0 [120/1] via 172.16.34.22, 00:00:04, Serial0/1
```

```
                    [120/1] via 172.16.33.21, 00:00:22, Serial0/2
R       172.16.32.0 [120/1] via 172.16.23.11, 00:00:17, FastEthernet0/1
                    [120/1] via 172.16.33.21, 00:00:22, Serial0/2
C       172.16.33.0 is directly connected, Serial0/2
C       172.16.34.0 is directly connected, Serial0/1
R       172.16.21.0 [120/2] via 172.16.34.22, 00:00:04, Serial0/1
                    [120/2] via 172.16.23.11, 00:00:17, FastEthernet0/1
                    [120/2] via 172.16.33.21, 00:00:22, Serial0/2
C       172.16.23.0 is directly connected, FastEthernet0/1
C       172.16.12.0 is directly connected, FastEthernet0/0
R       172.16.11.0 [120/1] via 172.16.23.11, 00:00:17, FastEthernet0/1
R*   0.0.0.0/0 [120/1] via 172.16.34.22, 00:00:06, Serial0/1
R120#
```

- Recette sur R210 :

```
R210#sh ip route
.........
Gateway of last resort is 172.16.43.22 to network 0.0.0.0

     172.16.0.0/24 is subnetted, 8 subnets
C       172.16.43.0 is directly connected, Serial0/0
C       172.16.32.0 is directly connected, Serial0/1
C       172.16.33.0 is directly connected, Serial0/2
R       172.16.34.0 [120/1] via 172.16.43.22, 00:00:21, Serial0/0
                    [120/1] via 172.16.33.12, 00:00:01, Serial0/2
C       172.16.21.0 is directly connected, FastEthernet0/0
R       172.16.23.0 [120/1] via 172.16.33.12, 00:00:01, Serial0/2
                    [120/1] via 172.16.32.11, 00:00:03, Serial0/1
R       172.16.12.0 [120/2] via 172.16.43.22, 00:00:21, Serial0/0
                    [120/2] via 172.16.32.11, 00:00:03, Serial0/1
                    [120/2] via 172.16.33.12, 00:00:01, Serial0/2
R       172.16.11.0 [120/1] via 172.16.32.11, 00:00:03, Serial0/1
R*   0.0.0.0/0 [120/1] via 172.16.43.22, 00:00:19, Serial0/0
R210#
```

Vous aussi pouvez observer trois routes RIP à coût égal entre LAN12 et LAN21, peut-être avec une solution différente. Quoi qu'il en soit, bravo, votre maîtrise de RIP ne fait aucun doute.

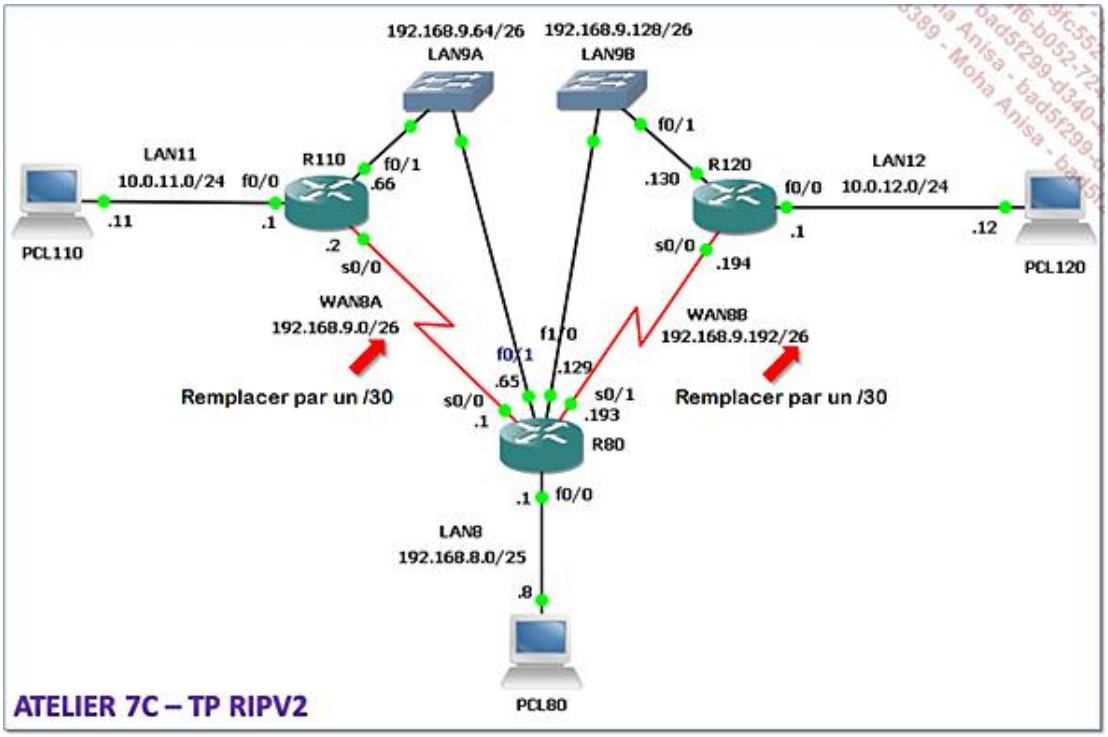
Chapitre 4 - Le protocole de routage type DV RIPv2

1. TP : Mise en œuvre d'une configuration RIPv2

Bien sûr, rien n'interdit au lecteur de reproduire l'ensemble des topologies du chapitre 4 dans GNS3 mais il faut bien avouer que faire fonctionner les six routeurs des ateliers 7A et 7B n'est pas une mince affaire, l'ensemble a manqué cruellement de stabilité sur la machine de l'auteur, malgré ses efforts.

Plus modestement, proposons-nous d'utiliser la topologie ci-dessous qui avait servi à illustrer les limites d'un protocole de routage avec classe. Le domaine couvert par le réseau 10 est scindé en deux parties. Par ailleurs, le masque adopté pour découper le réseau 192.168.9.0 est unique ce qui conduit à un important gaspillage d'adresses sur les liens « serial ». L'objectif est donc double :

- Remplacer les sous-réseaux /26 des liens « serial » par des sous-réseaux /30 du même réseau majeur 192.168.9.0 ;
- Appliquer le protocole RIPv2.



Configuration de R110 :

```
R110#sh run
Building configuration...
.........
!
hostname R110
!
enable secret 5 $1$7VWa$54CP2goqJ7nQwn152lU4b.
!
Memory-sizeiomem 15
ip subnet-zero
!
call rsvp-sync
!
interface FastEthernet0/0
ip address 10.0.11.1 255.255.255.0
duplex auto
speed auto
!
```

```
interface Serial0/0
bandwidth 64
ip address 192.168.9.2 255.255.255.252
!
interface FastEthernet0/1
ip address 192.168.9.66 255.255.255.192
duplex auto
speed auto
!
router rip
  version 2
  network 10.0.0.0
  network 192.168.9.0
  no auto-summary
!
Ipclassless
.........
```

Configuration de R80 :

```
R80#sh run
Building configuration...
.........
hostname R80
!
enable secret 5 $1$l652$eM7qlHN9OeZK87Tf9GzUT.
!
Memory-sizeiomem 15
ip subnet-zero
!
call rsvp-sync
!
interface FastEthernet0/0
ip address 192.168.8.1 255.255.255.128
duplex auto
speed auto
!
interface Serial0/0
bandwidth 64
ip address 192.168.9.1 255.255.255.252
clock rate 64000
!
interface FastEthernet0/1
ip address 192.168.9.65 255.255.255.192
duplex auto
speed auto
!
interface Serial0/1
bandwidth 64
ip address 192.168.9.193 255.255.255.252
clock rate 64000
!
interface FastEthernet1/0
ip address 192.168.9.129 255.255.255.192
duplex auto
speed auto
!
router rip
version 2
network 192.168.8.0
network 192.168.9.0
no auto-summary
!
Ipclassless
.........
```

Configuration de R120 :

```
R120#sh run
```

```
Building configuration...
.........
hostname R120
!
enable secret 5 $1$joNT$9qqIcFuwkyBAlJ8Oh891D0
!
Memory-sizeiomem 15
ip subnet-zero
!
call rsvp-sync
!
interface FastEthernet0/0
ip address 10.0.12.1 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
bandwidth 64
ip address 192.168.9.194 255.255.255.252
!
interface FastEthernet0/1
ip address 192.168.9.130 255.255.255.192
duplex auto
speed auto
!
Router rip
version 2
network 10.0.0.0
network 192.168.9.0
no auto-summary
!
ip classless
……
```

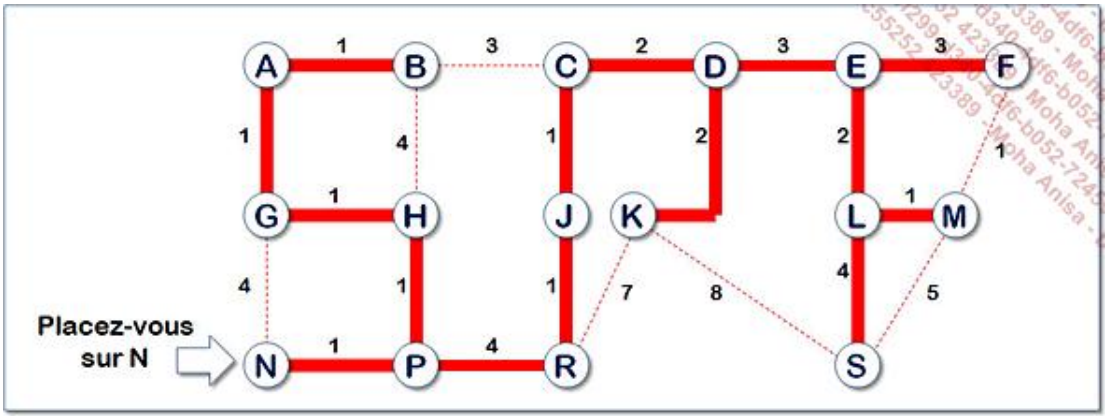
Extrait limité aux routes apprises par RIP de la table de routage de R110 :

```
R110#sh iproute  rip
192.168.8.0/25 issubnetted, 1 subnets
R       192.168.8.0 [120/1] via 192.168.9.1, 00:00:24, Serial0/0
                    [120/1] via 192.168.9.65, 00:00:24, FastEthernet0/1
     192.168.9.0/24 isvariablysubnetted, 4 subnets, 2 masks
R       192.168.9.192/30 [120/1] via 192.168.9.65, 00:00:24, FastEthernet0/1
                         [120/1] via 192.168.9.1, 00:00:24, Serial0/0
R       192.168.9.128/26 [120/1] via 192.168.9.65, 00:00:24, FastEthernet0/1
                         [120/1] via 192.168.9.1, 00:00:24, Serial0/0
     10.0.0.0/24 issubnetted, 2 subnets
R       10.0.12.0 [120/2] via 192.168.9.1, 00:00:24, Serial0/0
                  [120/2] via 192.168.9.65, 00:00:24, FastEthernet0/1
R110#
```

Cet atelier est maintenant terminé.

Chapitre 6 - Le protocole de routage type états de liens OSPF

1. Jeu - Construire un arbre SPF



Vous avez certainement trouvé un arbre qui, avec un peu d'imagination, doit rappeler les trois lettres SPF.

Les ports d'administration

En dehors des interfaces réseau, le routeur est pourvu de deux interfaces de type série asynchrone, nommées port console et port auxiliaire et dédiées à l'administration du système. Le port console autorise un accès local et l'administrateur l'utilisera plutôt pour réaliser la configuration initiale du routeur. En effet, une fois configuré et en exploitation, le routeur est accessible par le réseau et l'administrateur peut en assurer la gestion à l'aide d'une session TELNET ou SSH (Secure Shell, version sécurisée destinée à remplacer TELNET). La seconde interface série, nommée port auxiliaire permet un accès de l'administrateur à distance. Un scénario possible est de reprendre la main sur l'équipement distant quand l'administrateur n'y parvient plus par le réseau. Pour profiter de cette possibilité, il faut avoir été prévoyant, c'est-à-dire avoir installé un modem au voisinage du routeur et avoir amené une ligne du réseau téléphonique commuté sur ce modem, ce qui revient à attribuer un numéro de téléphone au routeur. Une exception cependant : les routeurs de la gamme 800 ne disposent que d'un seul port série asynchrone destiné à l'administration. Appelé « console aux port », ce port est, selon le choix de l'administrateur, tantôt un port console, tantôt un port auxiliaire.

En fait, il n'y a pas de différence de nature entre le port console et le port aux, les deux ports sont des ports série asynchrones. La différence vient du câblage qu'en fait CISCO sur la face avant de ses routeurs. S'il avait fallu se conformer strictement à la norme RS-232, le port console aurait dû se présenter sous la forme d'un port DB-25 femelle et l'extrémité être de type ETCD (en anglais DCE) afin de faire face au terminal nécessairement ETTD (DTE). Le port auxiliaire lui, aurait dû adopter un port DB-25 mâle et adopter le type ETTD pour faire face au modem nécessairement ETCD. C'est ce que CISCO a fait sur certains de ses routeurs, par exemple quelques routeurs de la série 7200 (7200-I/O-FE, 7200-I/O, 7200-I/O-FE-MII).

Mais CISCO, sur la plupart de ses routeurs, a préféré privilégier la recherche de compacité et les deux ports ont été câblés sur des sockets RJ-45. Sur toute la gamme, il est facile de distinguer les deux sockets grâce à un code de couleurs : le socket attribué au port console est repérable à sa couleur bleu ciel, le socket attribué au port auxiliaire est repérable à sa couleur noire.

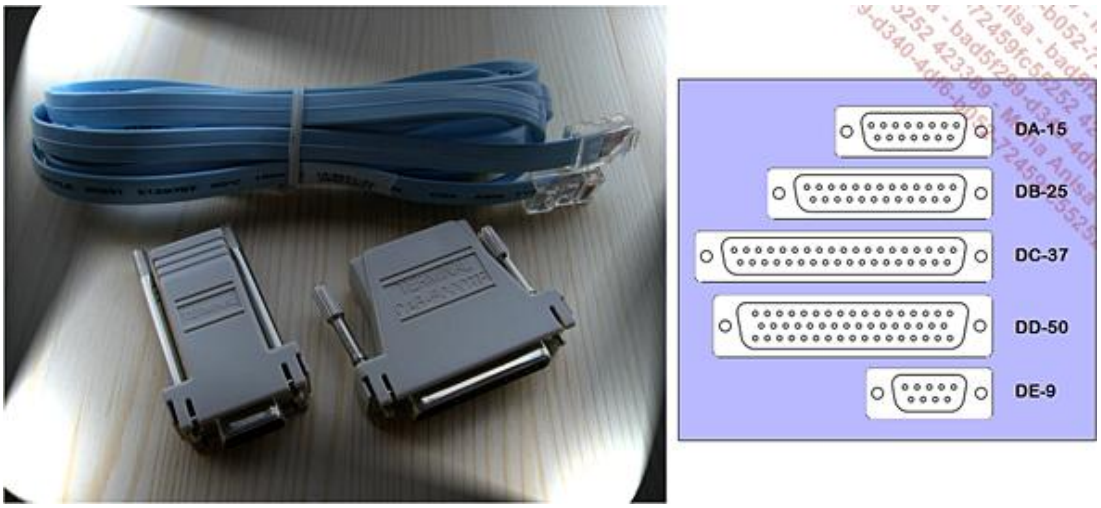
En dehors de la couleur, les ports console et auxiliaire se distinguent également par les choix qu'a fait le constructeur quant aux circuits de la norme RS-232 présents sur les 8 fils des sockets et quant à l'état de ces circuits, opérationnel ou forcé. En effet, les contraintes imposées par la liaison avec un terminal ne sont pas les mêmes que celles imposées par une liaison via modems. À moins d'avoir affaire à un administrateur fou, aucun risque pour que les caractères frappés au clavier du terminal et donc reçus par le port console ne saturent la capacité de réception du routeur sur ce port. De la même façon, il est peu probable que les caractères générés par le routeur ne dépassent les capacités d'affichage à l'écran du terminal (qui dépassent largement les capacités de lecture de l'administrateur et plus encore ses capacités à interpréter les messages). Le contrôle de flux est donc inutile sur ce port et CISCO n'en prévoit aucun, ni logiciel (à l'aide des caractères XON/XOFF), ni matériel (fondé sur l'état de l'un des circuits de la jonction). En revanche, le port auxiliaire dispose des circuits nécessaires à la réalisation d'un contrôle de flux matériel.

CISCO fournit également les cordons et adaptateurs nécessaires à la mise en œuvre de ces ports. En parcourant les guides d'installation des différents modèles de la gamme (tous téléchargeables sur le site CISCO), on peut dégager trois types de packagings :

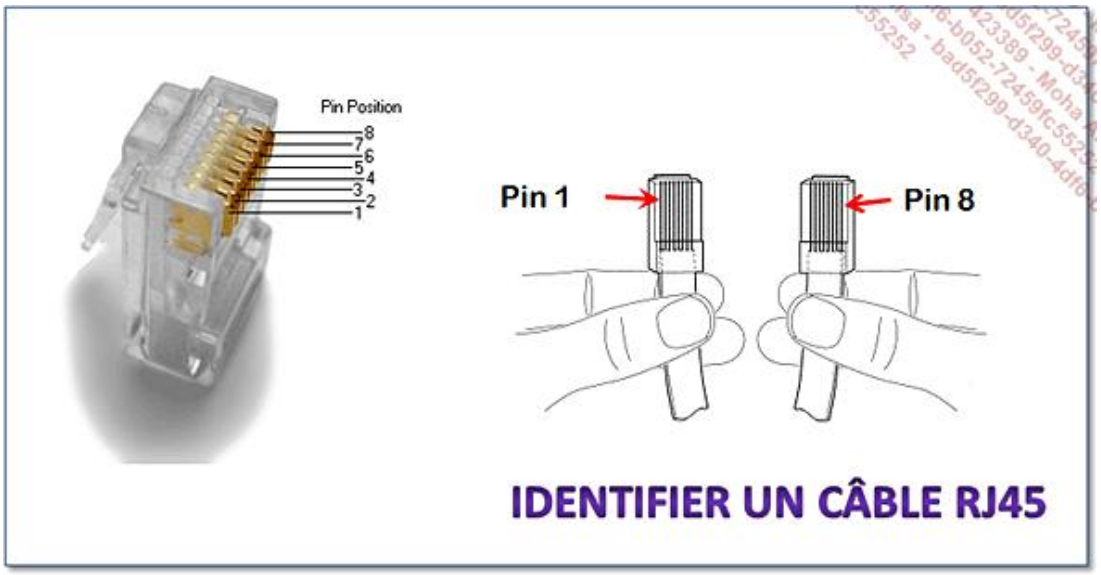
- Packaging 1 > Concerne les séries 800, 1800 et le routeur 2801. Le routeur est fourni avec :
    - Un câble console (RJ-45-to-DE-9, couleur bleu ciel) également appelé « management cable ».
    - Un adaptateur DE-9-to-DB-25.
- Packaging 2 > Concerne les séries 2800 à l'exception du routeur 2801, 3800, 7200, etc. Le routeur est fourni avec :
    - Un câble console (RJ-45-to-DE-9, couleur bleu ciel) également appelé « management cable » ou « console adapter câble ».
    - Un câble modem (RJ-45-to-DB-25, couleur noir) également appelé « modem adapter cable ».
- Packaging 3 > Concerne les séries 2600, 3600, 3700 et 3800. Le routeur est fourni avec un kit comprenant :
    - Un câble inversé (RJ-45-to-RJ-45) appelé « rollovercable » dans la documentation CISCO.
    - Un adaptateur RJ-45-to-DE-9 femelle permettant la connexion du port console au port série d'un PC émulant le terminal. Cet adaptateur porte la mention « TERMINAL ».
    - Un adaptateur RJ-45-to-DB-25 femelle permettant la connexion du port console au port série d'un terminal. Cet adaptateur porte également la mention « TERMINAL ».

- Un adaptateur RJ-45-to-DB-25 mâle permettant la connexion du port auxiliaire au port série d'un modem. Cet adaptateur porte la mention « MODEM ».

La photo ci-dessous rassemble les éléments du kit fourni avec les séries 2600, 3600, 3700 et 3800 à l'exception de l'adaptateur modem :

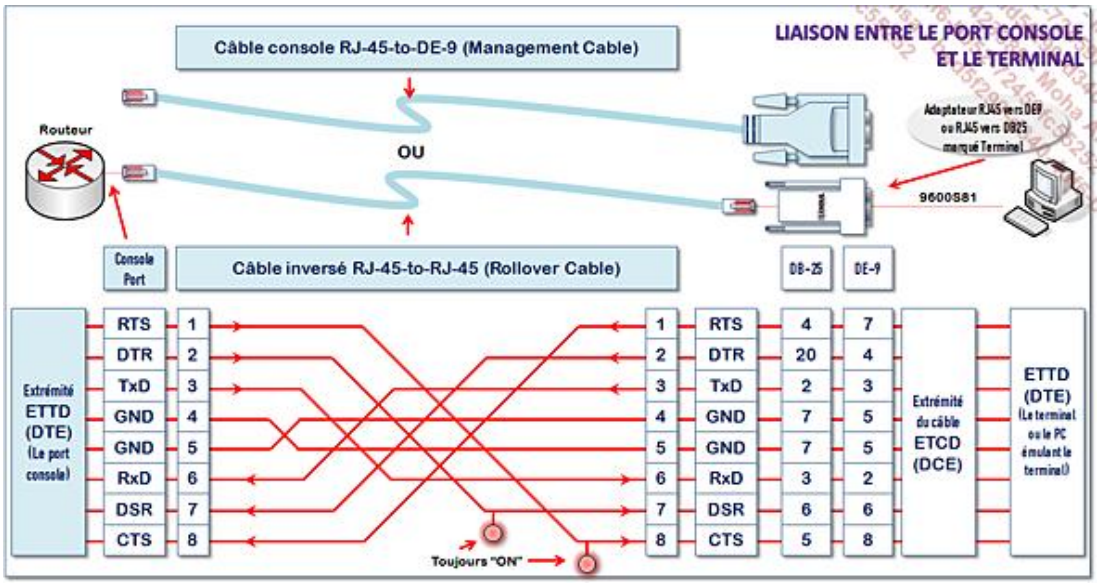


Réglons le cas du câble inversé « rollover » qui fait décidément couler beaucoup d'encre. Ce câble résulte simplement du fait que CISCO utilise du câble en nappe pour réaliser ses cordons destinés aux ports d'administration. Dans ce câble en nappe, les 8 fils cheminent de façon parallèle et ne sont pas organisés en paires. Les cordons réalisés avec une telle nappe ne pourraient supporter les débits des réseaux locaux. Fort heureusement, les débits à supporter sont ceux d'une liaison série asynchrone RS-232 et ne devraient donc jamais dépasser 19200 bits/s (il est possible de régler la jonction à 115200 bits/s mais c'est hors standard). Il faut observer que le sertissage d'un connecteur RJ-45 à chaque extrémité d'une nappe à 8 fils produit naturellement un câble inversé. Il ne faut pas y voir une volonté quelconque du constructeur mais simplement une conséquence du choix de la nappe en lieu et place d'un classique câble à paires torsadées. Si le lecteur dispose d'un câble inversé à proximité, c'est le moment de le vérifier en plaçant les deux connecteurs RJ-45 en vis-à-vis :



Les fils reliés à Pin 1 d'un connecteur et Pin 8 de l'autre connecteur ont la même couleur, puis pin 2 et pin 7 et ainsi de suite. Même s'il ne s'agit que d'un avatar de cordon RJ-45-to-RJ-45, l'administrateur ajoutera le câble inversé (*rollover*) à la panoplie de types qu'il doit connaître, qui comprenait déjà le câble droit (*straight-through*) et le câble croisé (*crossover*).

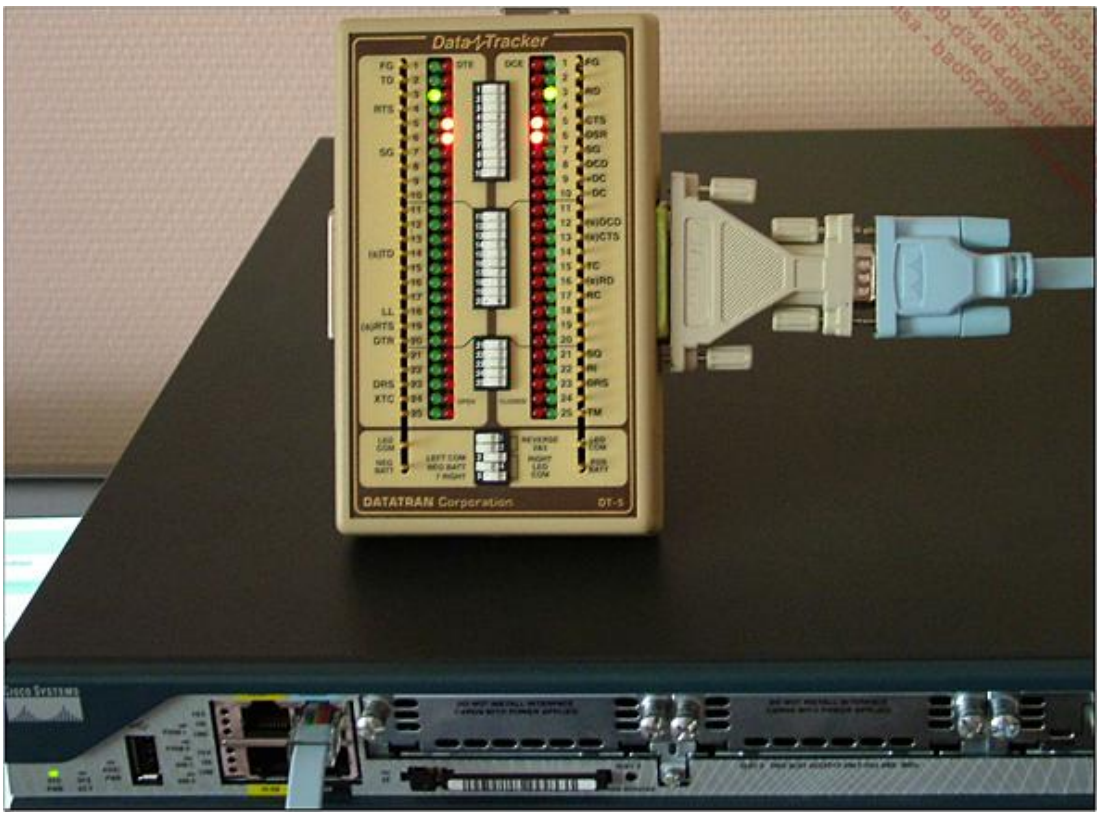
Le rétablissement des connexions attendues de façon à relier convenablement deux ETTD (Port console - Terminal) ou un ETTD et un ETCD (Port auxiliaire - Modem) incombe à l'adaptateur RJ-45-to-DB-xx. Ainsi, l'illustration suivante inventorie les circuits utilisés dans le cas d'une liaison port console - terminal :



Observez que les circuits DTR (*Data Terminal Ready*) et RTS (*Request To Send*) du port console restent en permanence montés. On est ainsi assuré que le terminal, qui reçoit ces états sur ses circuits DSR (*Data Set Ready*) et CTS (*Clear To Send*) est « satisfait » et consentira à afficher les caractères reçus du routeur ainsi qu'à envoyer au routeur les caractères frappés au clavier.

En final, il ne devrait subsister aucune ambigüité quand il faut relier le PC émulant un terminal au port console, c'est le câble console RJ-45-to-DE-9 qu'il faut utiliser (ou son équivalent reconstitué à l'aide du câble inversé bleu ciel associé à l'adaptateur RJ-45-to-DE-9), ce câble est de couleur bleu ciel, le socket du port console est également de couleur bleu ciel (le fil bleu sur le bouton bleu...). L'extrémité DE-9 du câble console est femelle, le connecteur DE-9 du port série du PC est mâle.

Le placement d'une jonction éclatée à l'extrémité d'un câble console confirme que cette extrémité est bien de type ETCD. En effet, le voyant associé au circuit RD est allumé et confirme que ce circuit est générateur (au sens électrique), ce qui est bien le fait d'une jonction ETCD.



Définition d'un contexte d'atelier

Dès le premier ouvrage de cette série, le souci de l'auteur a été de permettre au lecteur de progresser chez lui avec ses propres moyens. Les ateliers proposés faisaient un abondant usage de VMware Workstation. Pour mémoire, VMware installé sur une machine hôte, permet d'y créer autant de machines virtuelles que nécessaire et que peut en supporter l'hôte. Chaque machine virtuelle est un espace clos dans lequel il est possible d'installer la plupart des systèmes d'exploitation que connaît le PC (l'éditeur en revendique deux cent). Si la machine hôte dispose de plusieurs processeurs (cas des machines « dual core » et « quad core », alors il est possible de configurer la machine virtuelle pour qu'elle profite d'un, de deux ou de quatre processeurs, cela n'a évidemment d'intérêt que si le système d'exploitation installé sur la machine virtuelle est conçu pour tirer parti de plusieurs processeurs (Windows 2000 Professionnel ou Windows XP par exemple peuvent mettre à profit deux processeurs, mais il faut se tourner vers des versions Serveur pour en supporter davantage). La quantité de mémoire disponible sur la machine hôte est déterminante mais si le système d'exploitation, comme c'est encore le plus souvent le cas au moment où ces lignes sont écrites, est une version 32 bits, alors le PC hôte embarque au maximum 2^{32}=4 Go de RAM, limite qui ne sera franchie qu'avec l'adoption des systèmes d'exploitation 64 bits.

L'essentiel est encore à venir : chaque machine virtuelle peut être dotée d'un ou plusieurs adaptateurs réseaux virtuels. L'administrateur décide ensuite de connecter chacun de ces adaptateurs à l'un des concentrateurs virtuels, VMware Workstation en fournit dix notés VMnet0 à VMnet9. Chacun de ces concentrateurs peut être à son tour relié à un adaptateur réseau virtuel installé cette fois dans la machine hôte (noté « *VMware virtual ethernet adapter for VMnet**x*** ». Parmi les nombreuses autres possibilités, notons celle qui consiste à établir un lien de type pont (« bridge ») entre l'adaptateur réseau physique de la machine hôte et l'un des concentrateurs virtuels VMnetx. En final, toute configuration de réseau local mêlant machines virtuelles et la machine physique est donc facile à réaliser.

Quant aux routeurs, CISCO propose bien un excellent outil nommé Packet Tracer. Mais il ne s'agit que de simulations. Cet ouvrage propose de passer de la simulation à l'émulation à l'aide de l'outil GNS3 (*Graphical Network Simulator*) qu'il est possible de télécharger sur le site : http://www.gns3.net/

GNS3 se définit comme un simulateur de réseau graphique, mais en réalité, il s'agit plutôt d'une interface qui facilite la mise en œuvre de Dynamips, logiciel qui permet d'émuler un routeur physique. Dynamips est au routeur ce que VMware est au PC. VMware permet de créer une machine virtuelle dans laquelle l'administrateur installe un système d'exploitation comme il le ferait sur un PC réel. De la même façon, Dynamips permet de créer un routeur virtuel sur lequel l'administrateur charge l'image IOS convenable comme il le ferait sur un routeur réel. Et c'est là tout l'intérêt pédagogique. Apprendre sur un PC virtuel émulé dans VMware crée les mêmes savoir-faire qu'apprendre sur un PC physique. De façon analogue, un routeur émulé à l'aide de Dynamips se comporte strictement comme le routeur physique porteur de la même image IOS, les apprentissages sont donc les mêmes mais il devient possible de les délocaliser. C'est un peu comme si on permettait à l'étudiant d'emmener le bundle (ensemble de matériels CISCO que doit acquérir tout organisme de formation qui adhère à l'académie CISCO) sous le bras ! Merci donc à son créateur M. Christophe FILLOT de l'Université de Technologie de Compiègne.

Dynamips est associé à Dynagen, une interface écrite dans le langage de programmation Python et qui facilite l'interconnexion de plusieurs machines émulées d'une même topologie. GNS3, également écrit en langage Python, fournit une interface utilisateur graphique facilitant l'exploitation de Dynamips/Dynagen. Dynamips est capable d'émuler actuellement les plates-formes 1700, 2600, 3600, 3700 et 7200.

L'ensemble des ateliers de cet ouvrage a été réalisé sur un portable équipé d'un processeur Intel T9600 Dual Core cadencé à 2,8 GHz et qui embarquait 4 Go de RAM. Le système d'exploitation hôte a été Windows 7 en version d'évaluation 7100. VMware Workstation était en version 6.5 (une version 7 est disponible). GNS3 pour Windows était en version 0.6.1.

Il reste à évoquer le problème des licences logicielles. En effet, le routeur émulé résulte de l'association de Dynamips et d'une image IOS valide. Si cela ne pose aucune difficulté pour un client CISCO, il en va en principe autrement pour l'étudiant administrateur en devenir. Le problème n'est pas différent de celui posé par le choix d'un système d'exploitation pour toute machine virtuelle VMware.

1. Préparation des machines virtuelles VMware

Les machines virtuelles préparées par l'auteur afin de servir de cadre aux ateliers de cet ouvrage sont inventoriées dans le tableau ci-dessous :

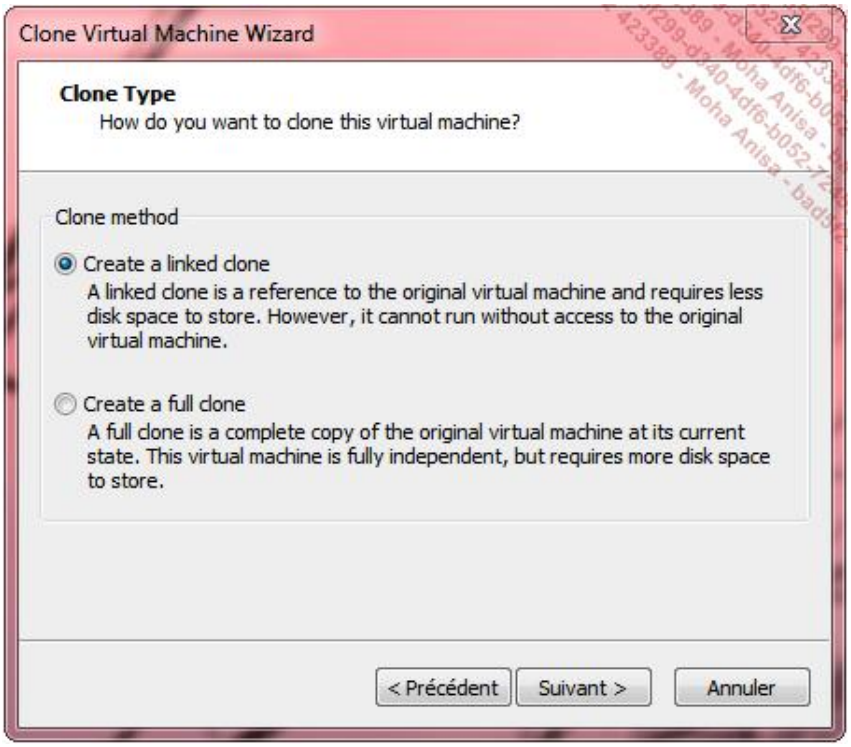
	VMSRV01	VMWKS02	VMWKS03	PC8, 11, 12, 21, 22 si Windows PCL8, 11, 12, 21, 22 si Linux
Nombre processeurs	1	2	1	1
Mémoire vive	384 Mo	1024 Mo	128 Mo	64 Mo si W2000 48 Mo si Linux
Disque	8 Go	6 Go	6 Go	6 Go
Nombre Adaptateurs réseau	1	5	1	1
Système d'exploitation	W2000 SRV	XP Prof SP3	W2000 Prof SP4	W2000 Prof SP4 Ou Ubuntu
Logiciels notables	Services réseau	GNS3	Serveur SYSLOG (Kiwi) Serveur RADIUS (RADL)	PuTTY Wireshark

Chacun des ateliers proposés ensuite ne nécessitera pas d'activer ensemble toutes ces machines fort heureusement. À chaque instant, le souci doit être d'économiser la quantité de mémoire affectée aux différentes machines. C'est ainsi que les machines PC8, PC11, PC12, PC21, PC22 qui ne servent qu'à tester la connectivité doivent être des machines « Weight Watchers ». N'hésitez pas à désactiver des services inutiles (dans ce contexte) tels que :

- Client de suivi de lien distribué.
- Mises à jour automatiques.
- Planificateur de tâches.
- Agent de stratégie Ipsec.

À ce sujet, l'auteur utilise depuis peu les services de TuneUp 2010 sur la machine hôte. Il a ainsi été très facile de désactiver tout ce qui ne sert que l'apparence au profit d'une machine devenue très fluide, il faut savoir ce que l'on veut.

PC11, PC12, PC21 et PC22 peuvent être obtenus très simplement par clonage de la machine PC8. De plus, parmi les options proposées lors du clonage, l'une d'elles permet, en conservant un lien avec la machine d'origine, d'obtenir une nouvelle machine avec très peu d'espace disque consommé :



Les férus de Linux gagneront sans doute à reproduire ces mises en situation sous leur système d'exploitation préféré. Pour sa part, l'auteur avoue une certaine réticence, toujours gêné par le militantisme, quelquefois assez peu professionnel, des porteurs de la bonne nouvelle du manchot. Mais la recherche d'efficacité et de compacité prime. C'est pourquoi nous avons placé en téléchargement sur le site ENI une machine virtuelle prête à l'emploi (merci à Gaëtan d'avoir préparé cette machine) dans l'archive wm_ubuntu.zip. Une documentation est incluse au format pdf. Sous VMware, ouvrez cette machine Ubuntu puis clonez-la afin de produire les machines PCL8, PCL11, PCL12, PCL21 et PCL22. Pour chacune des machines, voici la séquence de commandes nécessaires afin d'adapter la machine au contexte :

```
Login : ubuntu
Mot de passe : sunrise
$ ifconfig ▲ a
```

L'invite de commandes (le « prompt ») est matérialisé par le symbole $. Le système renvoie la liste des interfaces réseau qu'il connaît, notez le numéro d'ordre affecté à l'interface Ethernet, par exemple **eth4**.

```
$ sudo ▲ nano ▲ /etc/network/interfaces
```

Nano est un éditeur de texte. **Sudo** informe le système d'exploitation qu'il doit exécuter la commande avec le niveau de privilège root (administrateur). Le mot de passe root est également sunrise. Le fichier ouvert contient la configuration courante des interfaces. Sous le label **# The primary network interface**, remplacez les deux occurrences **ethx** par **eth4** puis adaptez la configuration IP. Sortez par [Ctrl] X, Y pour yes et validez par la touche [Entrée]. Redémarrez la partie réseau afin de rendre effective la nouvelle configuration à l'aide de la commande :

```
$ sudo ▲ /etc/init.d/networking ▲ restart
```

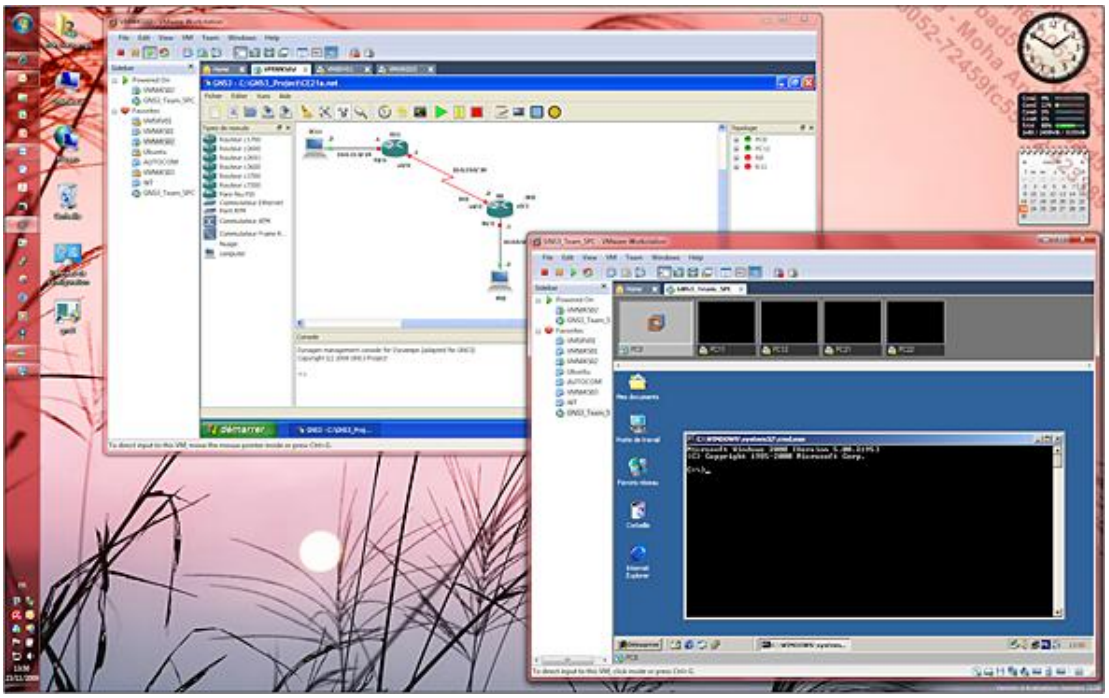
Le système répond :

```
* Reconfiguring network interfaces -     [OK]
```

Voilà votre machine prête à l'emploi. Attention à la commande **ping** qui, à la différence des systèmes Windows, génère des requêtes de façon continue (une commande **ping -t** provoquerait le même effet sous Windows) et dont on sort à l'aide de la combinaison de touches [Ctrl] C. Attention également au fait que les outils de VMware (« VM Tools ») ne sont pas installés sur cette machine. Par conséquent, une fois que la fenêtre correspondante à cette machine a le focus, la seule façon d'en sortir est la combinaison de touches [Ctrl][Alt]. La barre d'état de VMware rappelle cette particularité.

À l'inverse des machines PC8 à PC22 (ou PCL8 à PCL22), la machine virtuelle nommée VMWKS02 accueille GNS3/Dynamips et a l'ambition de parvenir à faire fonctionner des topologies comprenant jusqu'à 6 routeurs. Ceci justifie une configuration plus musclée : 1 Go de RAM et deux processeurs. La machine VMWKS03 ne sera activée que pendant les ateliers organisés autour de SYSLOG (journalisation d'évènements) et RADIUS (authentification des utilisateurs). Enfin, la machine VMSRV01 héberge un Windows 2000 Server, ce qui peut s'avérer utile s'il fallait mettre en place un quelconque service réseau (DHCP, DNS...).

Nous ne détaillerons pas toutes les étapes nécessaires à la préparation d'un tel ensemble de machines, l'aide fournie par VMware est très complète. Il peut être agréable d'ouvrir plusieurs instances de VMware, c'est le cas quand on a la chance de disposer de plusieurs écrans sur la même machine hôte. Ceci s'opère à l'aide de la commande de menu VMware **File - New - Window**. Ainsi, dans l'exemple ci-dessous, une instance est ouverte sur une topologie de routeurs créée dans GNS3 hébergé par la machine virtuelle VMWKS02, l'autre instance de VMware est ouverte sur une équipe (« team ») composée des machines PC8, PC11, PC12, PC21 et PC22 :



Rassembler plusieurs machines virtuelles dans une équipe procure un certain nombre d'avantages :

- L'administrateur peut provoquer le démarrage de l'équipe par une seule action. L'ordre de démarrage des PC dans l'équipe est prédéterminé. De plus, l'administrateur peut ajuster le temps qui s'écoule entre le démarrage d'un PC et le démarrage du PC suivant (10 secondes par défaut).

- Les machines virtuelles d'une équipe peuvent être connectées à un segment LAN dont l'administrateur peut à la fois régler la bande passante et le taux d'erreur !

- Le focus est porté sur une machine de l'équipe mais les autres machines apparaissant sous forme de vignettes qui représentent l'activité réelle de l'écran.

2. Préparation des réseaux virtuels VMware

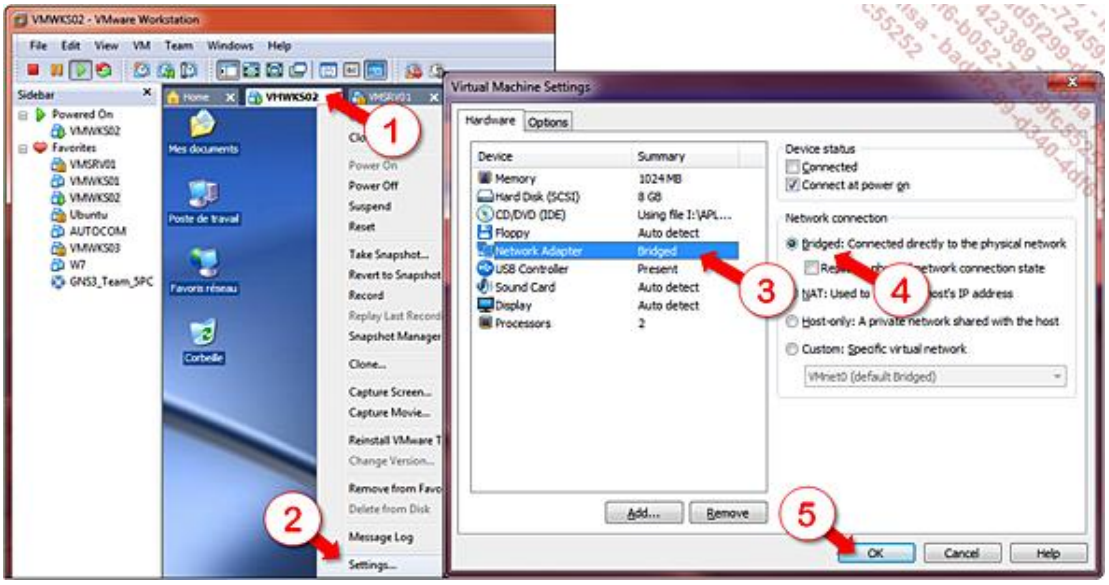
Il s'agit d'assurer la connectivité convenable des machines virtuelles entre elles, des machines virtuelles avec la machine hôte voire des machines virtuelles avec le ou les adaptateurs réseau physique qui équipent la machine hôte. Le tableau ci-dessous tente d'inventorier les connexions établies, il semblera probablement nébuleux au lecteur, mais il prendra du sens à mesure de l'avancée dans l'atelier :

	Hôte	VMSRV01	VMWKS02	VMWKS03	PC8	PC11	PC12	PC21	PC22
VMnet0			BRIDGE						
VMnet1			NIC11			NIC11			
VMnet2			NIC12				NIC12		
VMnet3			NIC21					NIC21	
VMnet4			NIC22						NIC22

VMnet5									
VMnet6									
VMnet7									
VMnet8	NIC8		NIC8		NIC8				
VMnet9									

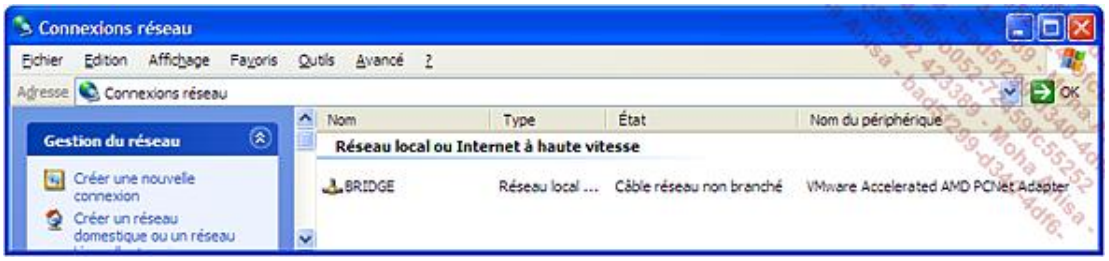
Chacune des cinq machines **PC(L)x** est reliée à la machine qui héberge GNS3. Les noms donnés aux adaptateurs réseau (NIC : *Network Interface Card*) le sont au sein du système d'exploitation qui équipe la machine correspondante. Dans la machine VMWKS02, il est conseillé d'ajouter un seul adaptateur réseau à la fois. Détaillons la procédure d'ajout d'un adaptateur réseau dans la machine VMWKS02 :

- La machine virtuelle **VMWKS02** est active. Commençons par un état des lieux. Effectuez un clic droit sur l'onglet de la machine virtuelle puis sélectionnez **Settings** :



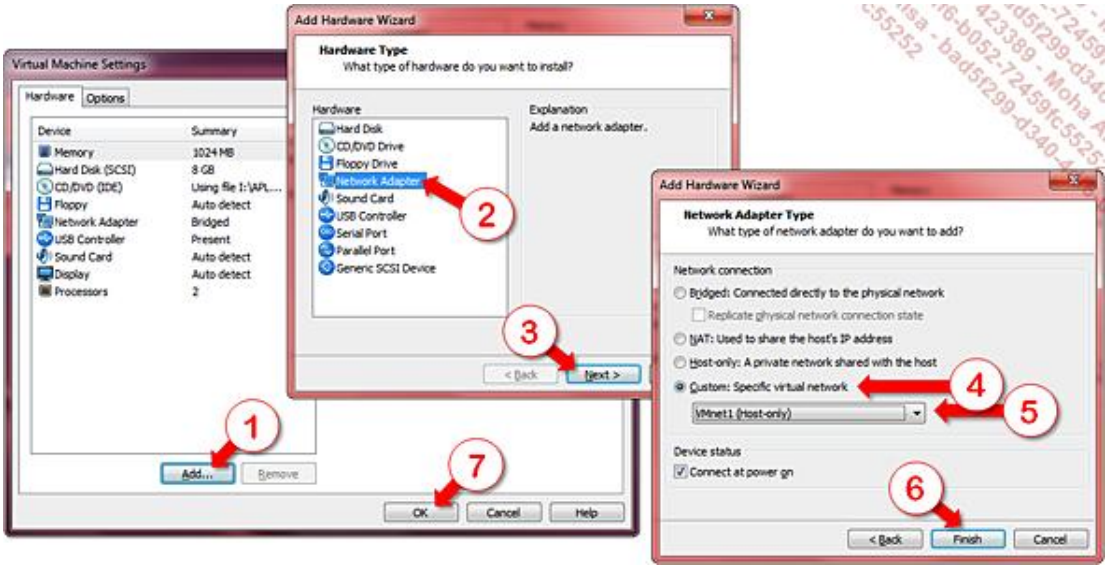
- Dans la fenêtre **Virtual Machine Settings**, sélectionnez l'élément **Network Adapter**. Faisons le choix de connecter cette carte virtuelle au réseau physique de la machine hôte en sélectionnant le bouton radio **Bridged**.

- Sur le bureau de la machine virtuelle, effectuez un clic droit sur l'icône **Favoris réseau** et sélectionnez **Propriétés** :

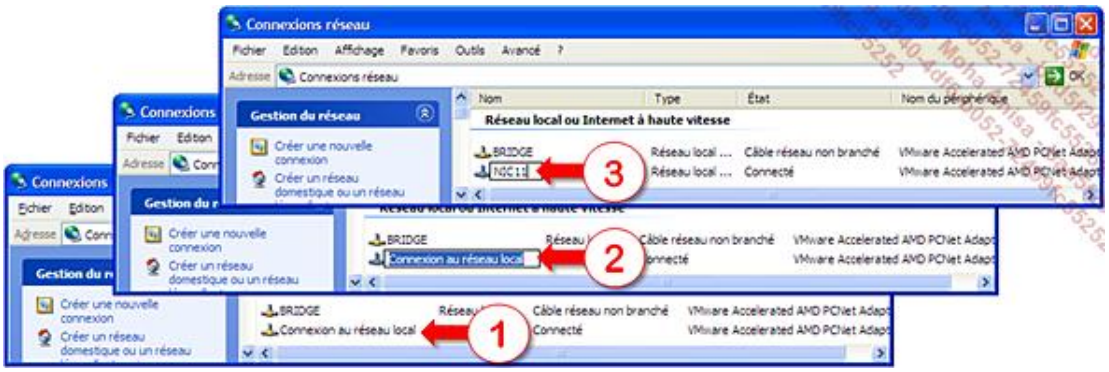


- Renommez l'adaptateur réseau BRIDGE (effectuez un clic droit sur l'adaptateur puis sélectionnez **Renommer**) afin d'éviter de confondre cet adaptateur existant avec les adaptateurs à venir.

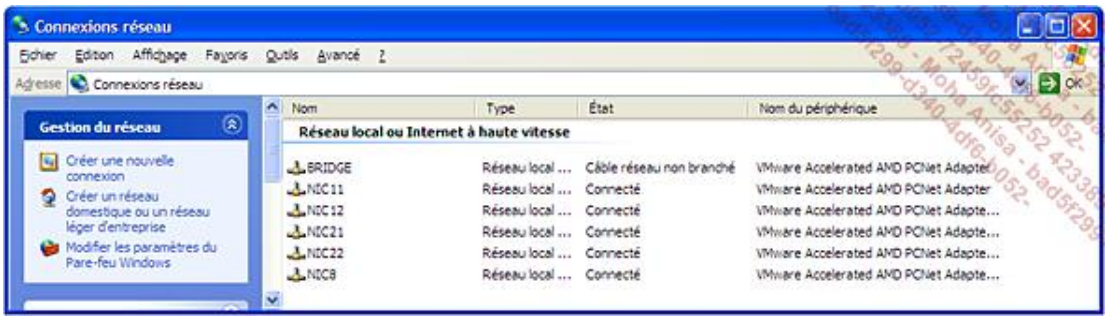
- Revenez aux réglages de la machine virtuelle et cliquez sur **Add**. Ajoutez un adaptateur réseau virtuel et connectez-le au concentrateur VMnet1 :



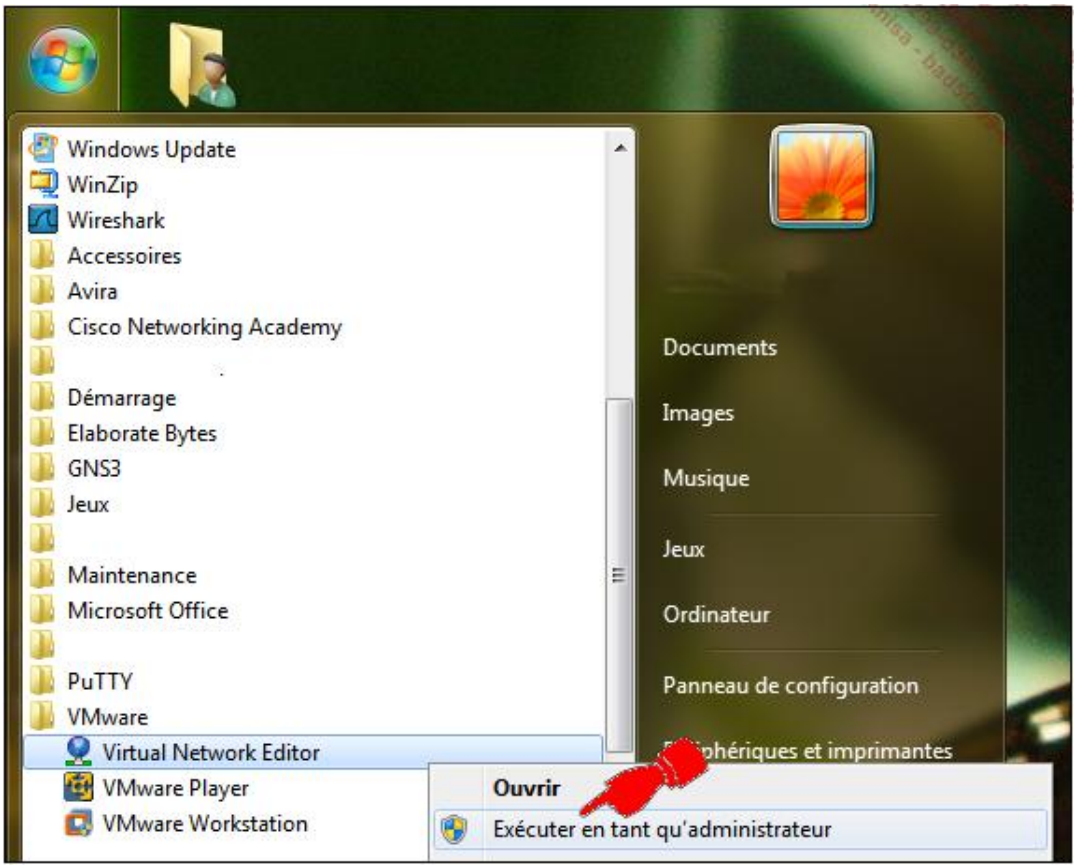
- Revenez au bureau de la machine virtuelle, effectuez un clic droit sur l'icône **Favoris réseau** et sélectionnez **Propriétés**. Renommez l'adaptateur ajouté en NIC11 :



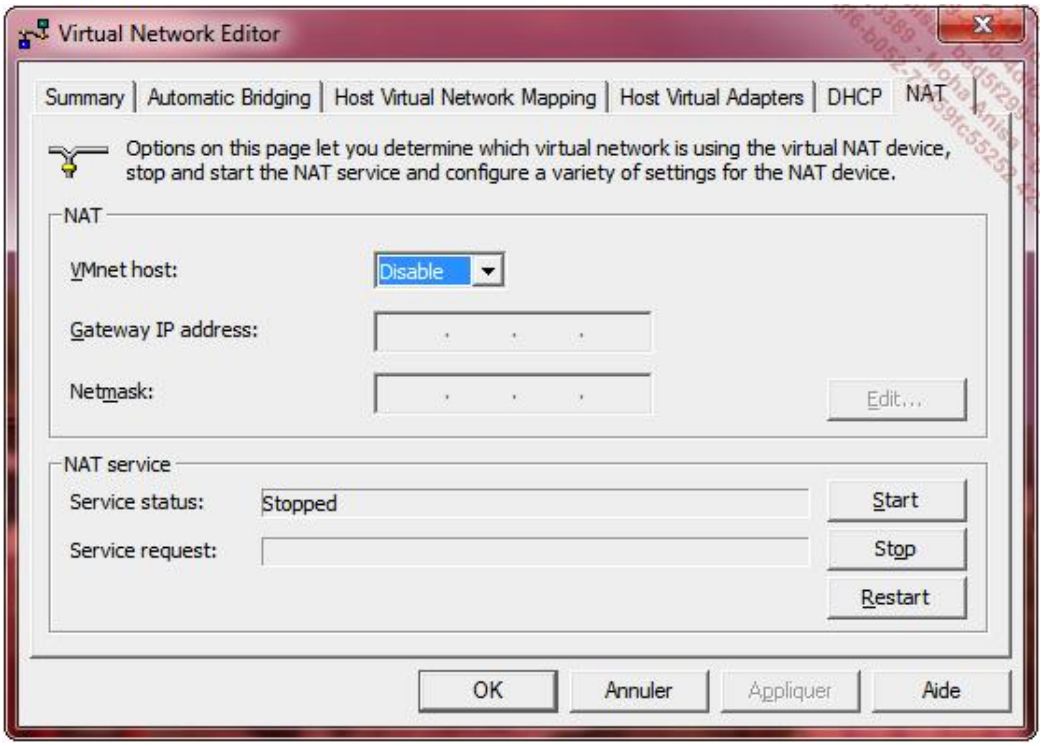
- Renouvelez l'ensemble de la séquence jusqu'à ce que la machine **VMWKS02** dispose de ses cinq adaptateurs NIC11, NIC12, NIC21, NIC22 et NIC8, chaque adaptateur étant connecté au concentrateur **VMnet** convenable, selon le tableau d'affectation des VMnet fourni en début de paragraphe :



- Depuis le menu **Démarrer** de la station hôte, lancez l'application **Virtual Network Editor** de VMware. Attention, si la station hôte est sous Windows Vista ou Windows 7, ce lancement doit s'opérer en mode administrateur ce qui est obtenu en effectuant un clic droit sur le raccourci de l'application :

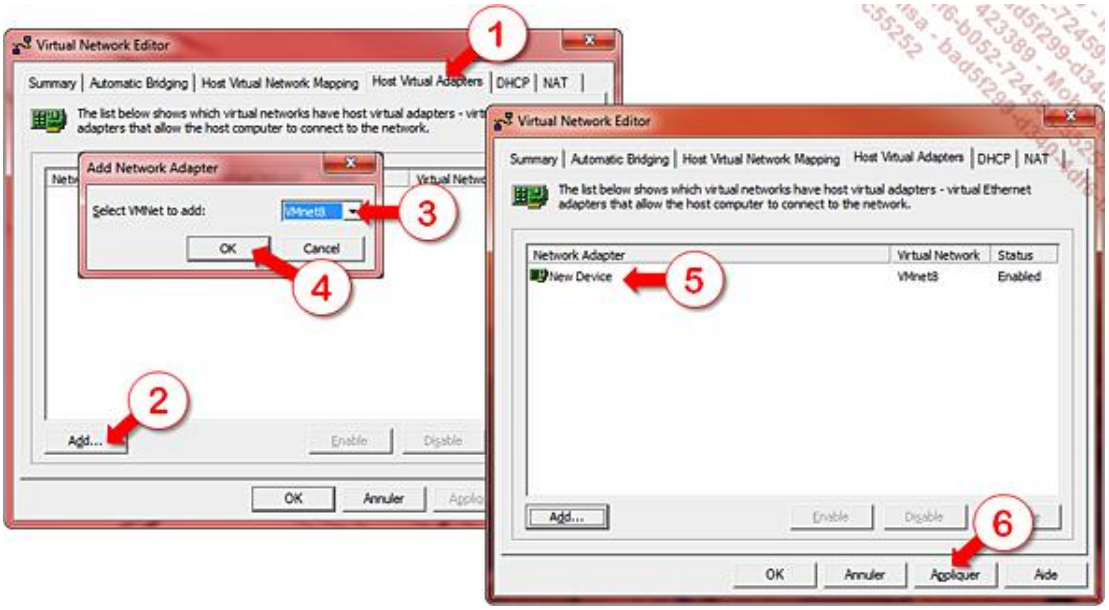


- De la fenêtre **Virtual Network Editor**, sélectionnez l'onglet **NAT** et désactivez toute translation d'adresses NAT :



Il est possible d'établir une connexion réseau entre la machine hôte et un ou plusieurs des concentrateurs virtuels VMnet. Par exemple, imaginons que l'on souhaite établir un lien avec VMnet8 :

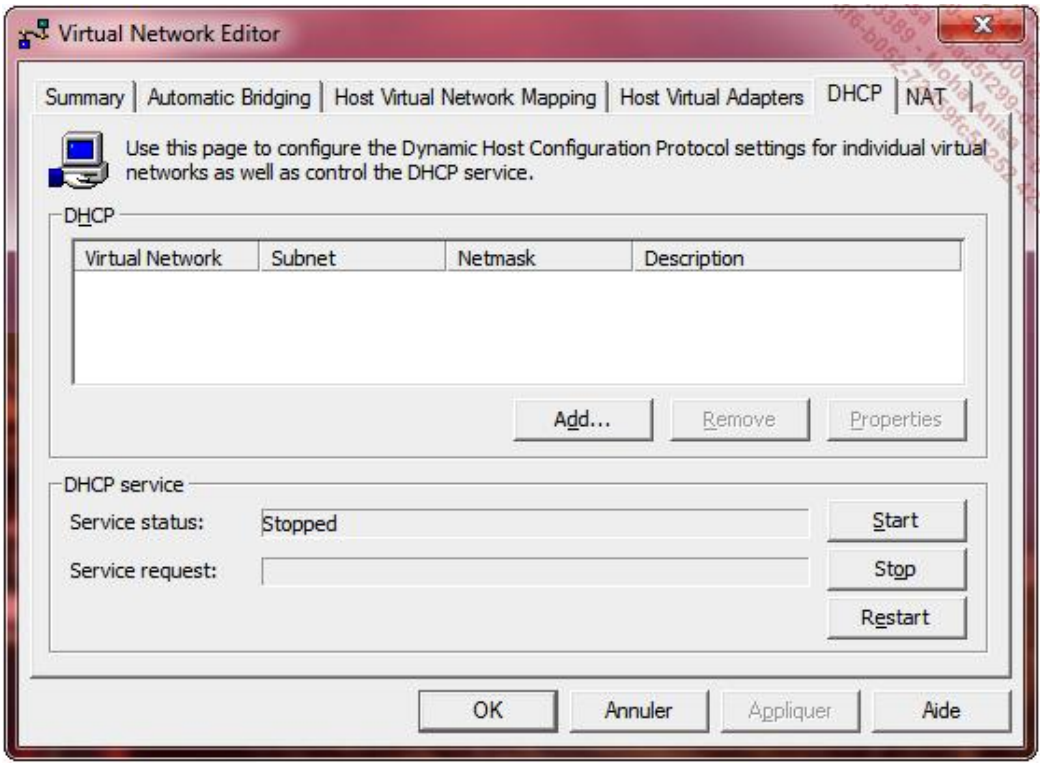
- De la fenêtre **Virtual Network Editor**, sélectionnez l'onglet **Host Virtual Adapters**. Cliquez sur **Add**. Sélectionnez **VMnet8** dans la liste déroulante puis confirmez et appliquez :



- De retour à la machine hôte, effectuez un clic droit sur l'icône **Réseau** ou **Favoris réseau** et sélectionnez **Propriétés**. Constatez la création du nouvel adaptateur et renommez-le en **NIC8** afin de ne pas le confondre avec d'autres adaptateurs à venir :

- Adaptez la configuration IP de **NIC8** aux tests en cours. Imaginons par exemple qu'il faille ouvrir une session Telnet sur un routeur émulé dans GNS3 et dont le port f0/0 d'adresse IP 10.0.8.1/24 soit connecté à VMnet8. Dans ce cas, il faut affecter l'une des adresses du réseau 10.0.8.0/24 à l'adaptateur **NIC8** de la machine hôte.

- Ouvrez à nouveau **Virtual Network Editor**, sélectionnez l'onglet **Host Virtual Adapters** et constatez que l'adaptateur virtuel **New device** est devenu l'adaptateur **NIC8**. Sélectionnez l'onglet **DHCP**. Otez tout réseau virtuel, cliquez sur **Stop** pour arrêter le service puis confirmez en cliquant sur **Appliquer** :



- Fermez l'application **Virtual Network Editor**.

3. Préparation du contexte GNS3/Dynamips

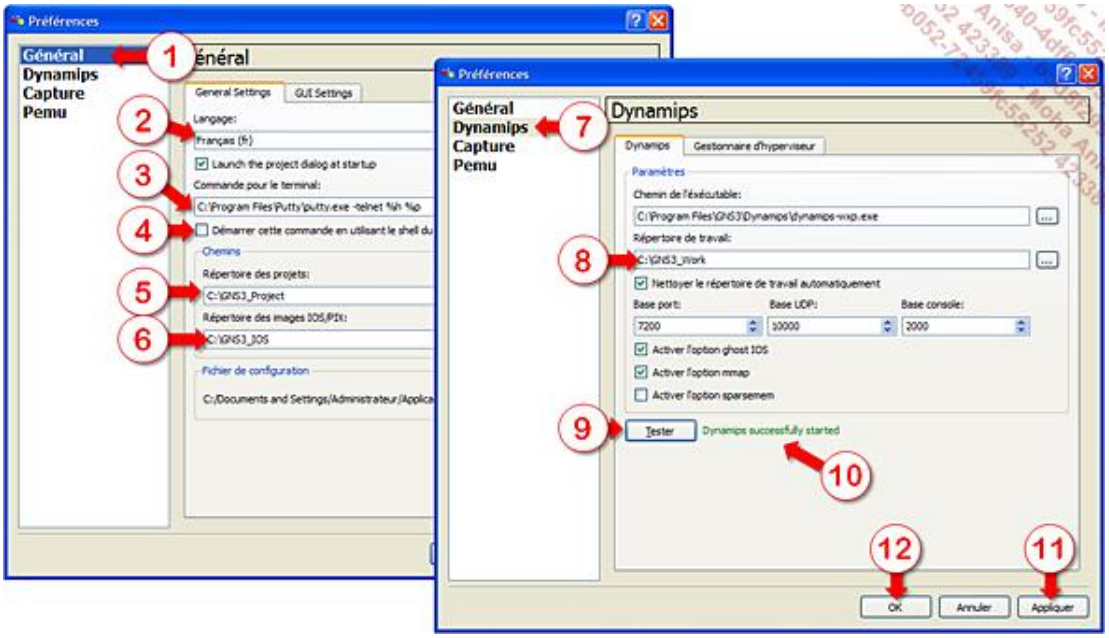
- Si ce n'est déjà fait, téléchargez PuTTY sur le site : http://www.chiark.greenend.org.uk/~sgtatham/putty/

- Installez PuTTY sur la machine virtuelle VMWKS02. VMware offre deux possibilités quand il faut copier des fichiers depuis la machine hôte vers une machine virtuelle :

  1. Partager un répertoire de la machine hôte que la machine virtuelle voit comme un lecteur réseau. Cette fonctionnalité, appelée **Shared folders** dans VMware, s'active depuis l'onglet **Options** de la fenêtre **Virtual Machine Settings**.

  2. Si les outils VMware (« **VM Tools** ») ont été installés sur la machine virtuelle, alors l'opération Glisser-Déposer fonctionne entre la machine hôte et les machines virtuelles.

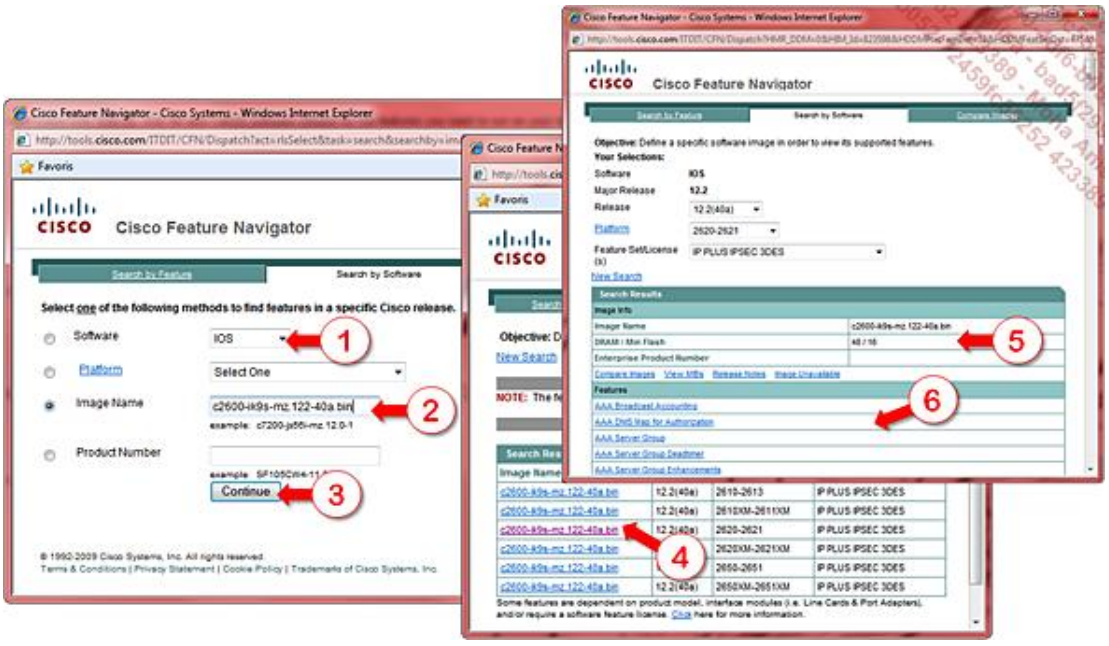
- Téléchargez GNS3 sur le site : www.gns3.net (GNS3-0.6.1-win32-all-in-one.exe - 11309 Ko au moment où ces lignes sont écrites). Installez GNS3 sur la machine virtuelle VMWKS02. Une fois l'installation terminée, GNS3 vous propose de régler immédiatement un certain nombre de paramètres qu'il est possible de retrouver ensuite via la commande de menu **Editer - Préférences** :



- La commande pour le terminal (point 3) permet de substituer PuTTY au Telnet intégré du système d'exploitation. Les différents répertoires GNS3_Project, GNS3_IOS, GNS3_Work ont été créés au préalable. Respectez bien les noms proposés afin de conserver la cohérence avec la suite de l'ouvrage (points 5, 6 et 8). En final, cliquez sur le bouton **Tester** pour vérifier le lancement de Dynamips. GNS3 doit répondre **Dynamips successfully started**.

- L'installation de GNS3 propose ensuite de renseigner l'image IOS à utiliser pour chaque plate-forme qu'il lui est possible d'émuler. Différez ce paramétrage, nous y reviendrons ultérieurement car il reste accessible via la commande de menu **Editer - Images IOS et hyperviseurs**.

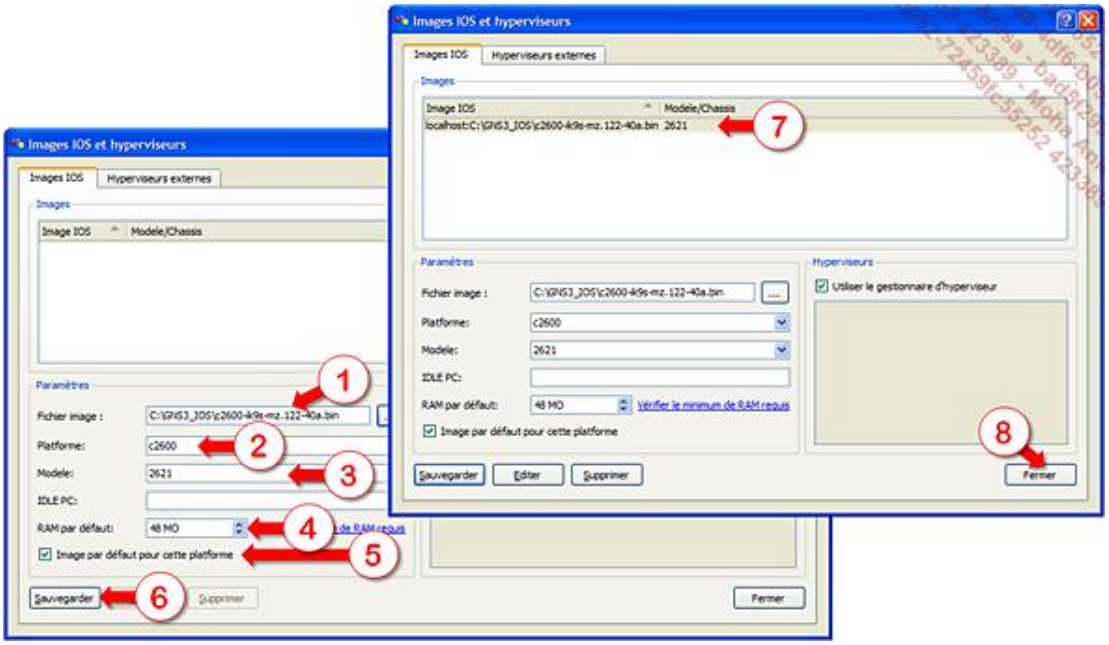
- Sur la machine hôte, vous êtes parvenu à récupérer une image CISCO IOS valide. Il est utile d'en vérifier les fonctionnalités à l'aide de l'outil **Cisco Feature Navigator** (à entrer dans un moteur de recherche pour trouver le lien) :



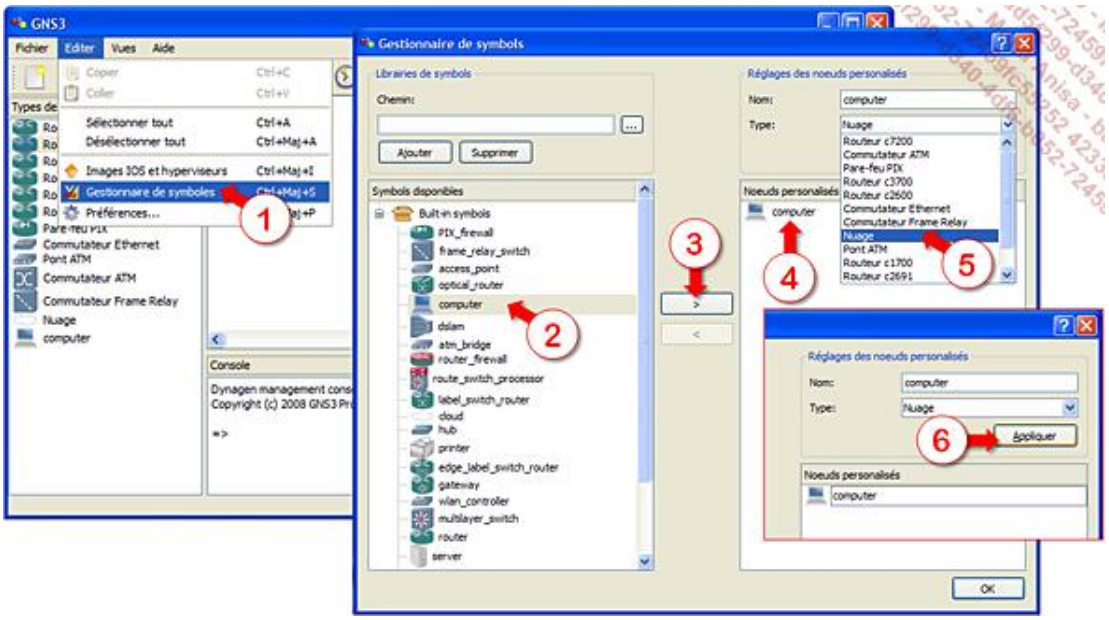
- Parmi les caractéristiques fournies, l'une intéresse le fonctionnement de Dynamips : il s'agit de la quantité de mémoire RAM nécessaire pour assurer le bon fonctionnement de l'IOS en question. Dans l'exemple ci-dessus, l'image IOS est c2600-ik9s-mz.122-40a.bin destinée à faire fonctionner un routeur émulé de type 2621. L'outil **CISCO Feature Navigator** informe que la plate-forme doit disposer de 48 Mo de RAM.

- Placez la précieuse image dans le répertoire GNS3_IOS de la machine virtuelle VMWKS02. Revenez à GNS3 et

lancez la commande de menu **Editer - Images IOS et hyperviseurs**. Paramétrez GNS3 afin qu'il utilise par défaut cette image pour la plate-forme 2600 et en lui affectant 48 Mo de mémoire vive :

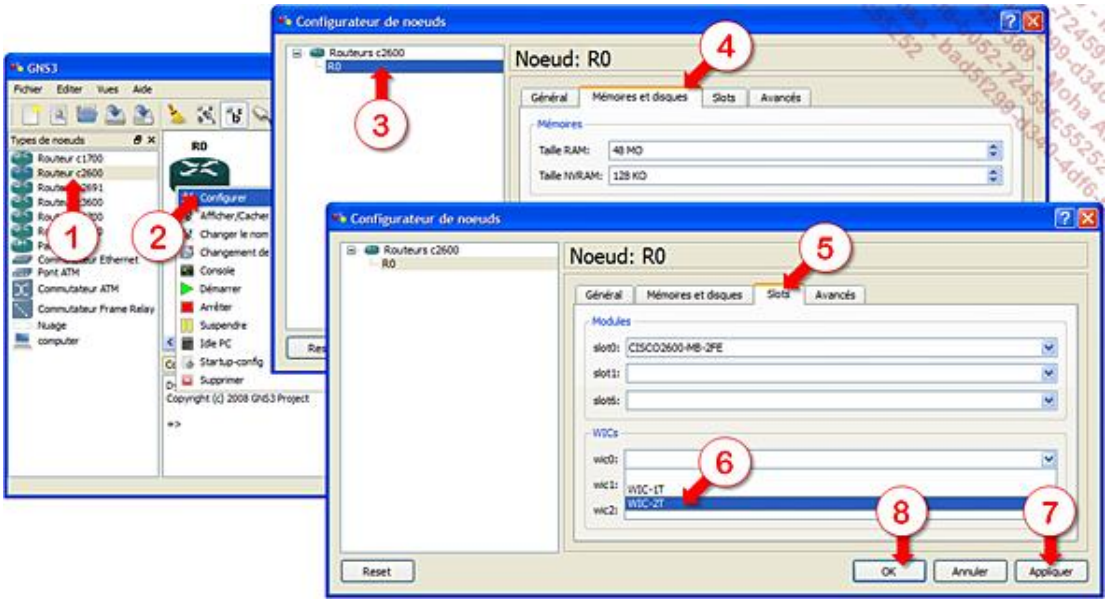


- Ouvrez le gestionnaire de symboles via la commande de menu **Editer - Gestionnaire de symboles**. Dans les symboles disponibles, sélectionnez **Computer** et transférez ce symbole du côté **Nœuds personnalisés**. Double cliquez sur **Computer** du côté Nœuds personnalisés (au point 4) puis remplacez le type **Nœud décoratif** par le type **Nuage** de la liste déroulante. Appliquez et fermez :

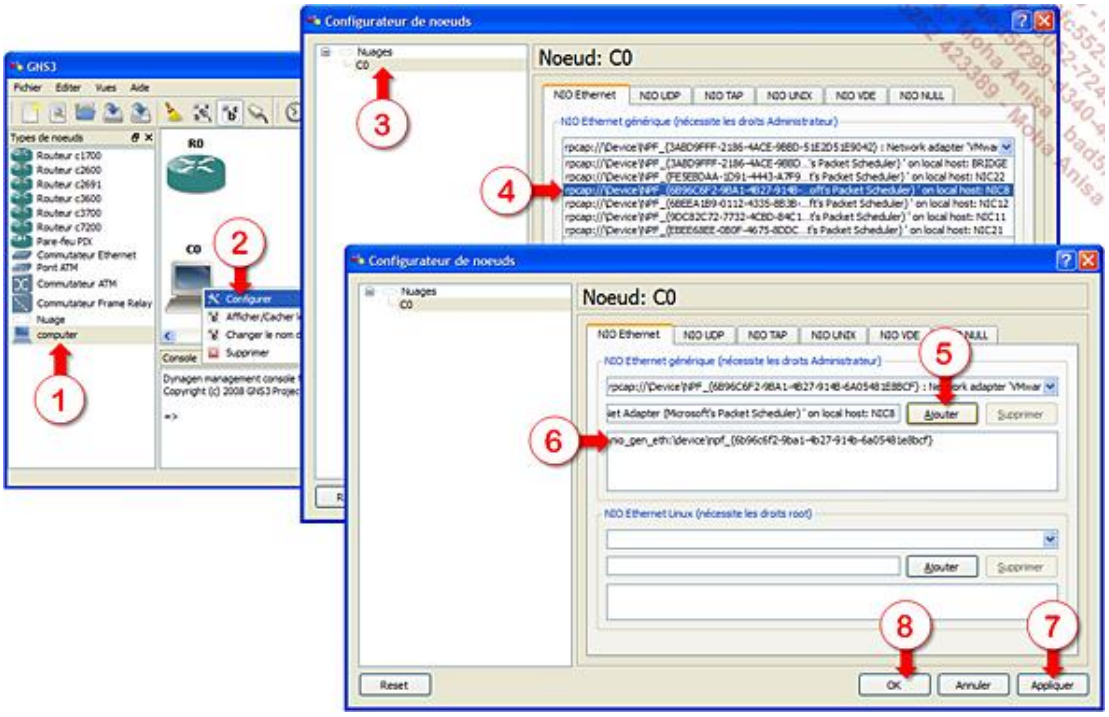


- Créez un nouveau projet via la commande de menu **Fichier - Nouveau Projet** de GNS3. Nommez votre premier projet **Atelier1a**, veillez à bien cocher les cases **Sauver les nvrams et autres disques (recommandé)** ainsi que **Exporter les fichiers de configuration des routeurs**. Confirmez. Quand le projet est ouvert et à chaque fois que la topologie ou la configuration d'un routeur ont fait l'objet de modifications, il est possible de sauvegarder l'ensemble en un seul clic sur le bouton étiqueté 2 :

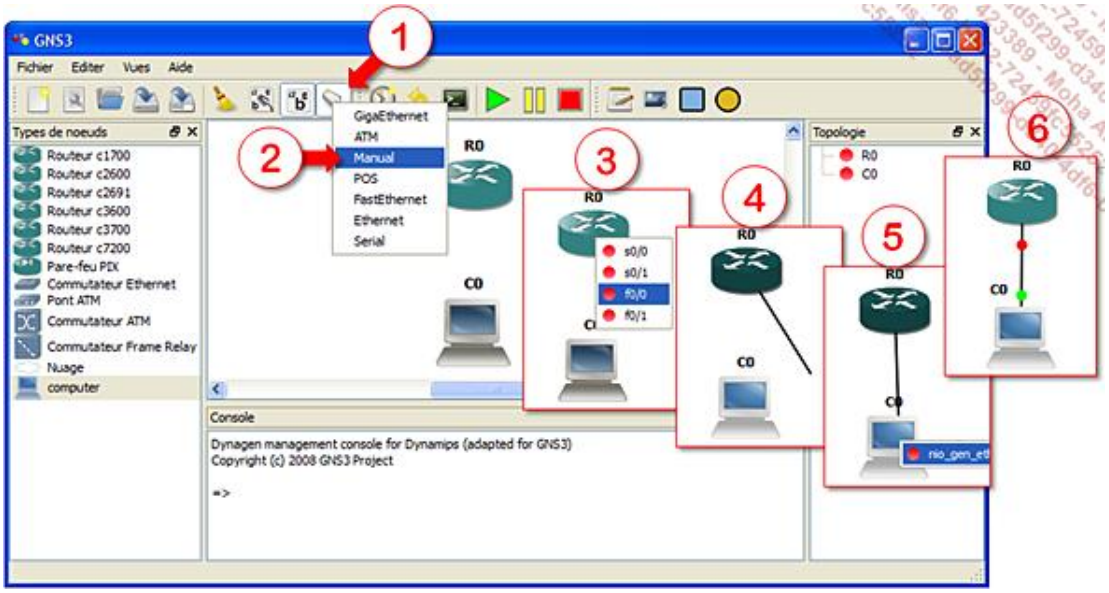
- Glissez/déposez un routeur C2600 sur la zone centrale de GNS3, zone destinée à recevoir votre topologie. Le routeur est numéroté automatiquement **R0**. Effectuez un clic droit sur R0 et sélectionnez **Configurer**. Dans la fenêtre **Configurateur de nœuds**, observez sans modifier l'onglet **Mémoires et disques** et notamment le dimensionnement des deux partitions RAM et NVRAM du routeur. Dans l'onglet **Slots**, observez que par défaut, le seul slot occupé l'est par une carte à deux ports Fast Ethernet. Nous aurons besoin également de ports WAN. Dans la sous-fenêtre **WICs**, ajoutez une carte **WIC-2T** (deux ports de type « serial »). Appliquez et fermez :



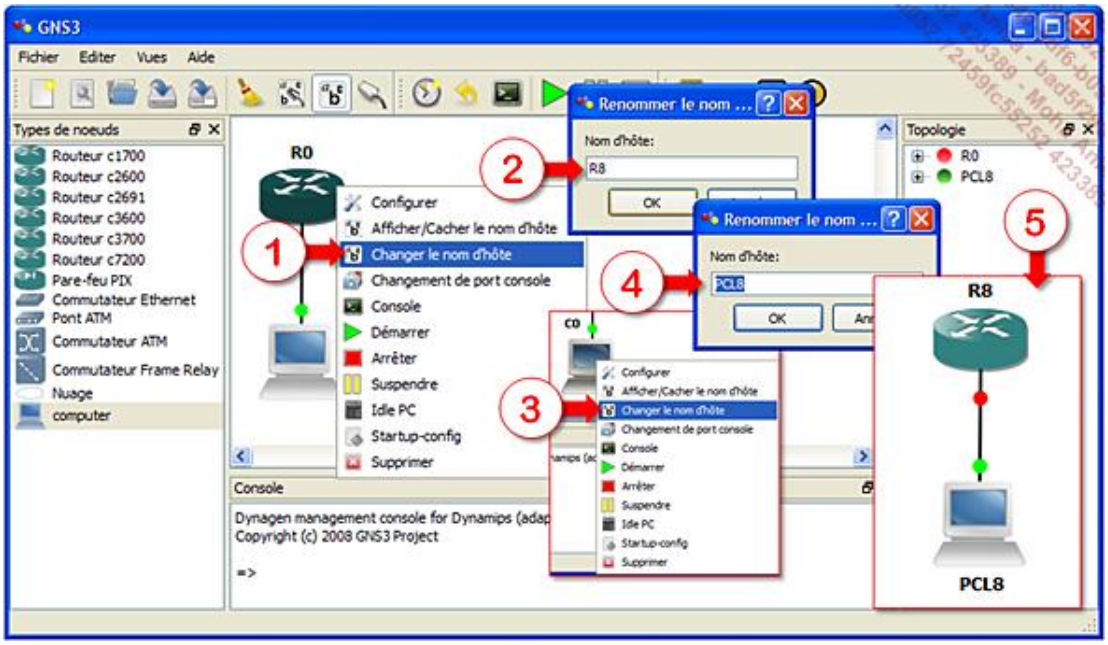
- Glissez/déposez un PC sur la topologie. Pour GNS3, il s'agit d'un nuage (*Cloud*) ce qui explique qu'il soit numéroté **C0**. Effectuez un clic droit sur C0 et sélectionnez **Configurer**. Dans la fenêtre **Configurateur de nœuds**, sélectionnez l'onglet **NIO Ethernet**. Déroulez la liste des adaptateurs réseau portés par la machine virtuelle VMWKS02. Vous devez y retrouver les adaptateurs précédemment créés, c'est-à-dire NIC8, NIC11, NIC12, NIC21, NIC22. Sélectionnez **NIC8** puis cliquez sur **Ajouter**. Ainsi, ce nuage sera désormais connecté à l'adaptateur virtuel NIC8 et donc au concentrateur VMnet8. Appliquez et fermez :



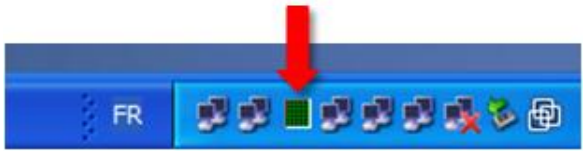
- Cliquez sur le bouton de barre d'outils **Ajouter un lien**. Dans le menu contextuel qui s'affiche, sélectionnez **Manual**. Le curseur devient une croix et le bouton de barre d'outils reste rouge pour indiquer que GNS3 est en mode d'édition de liens. Cliquez sur **R0** et sélectionnez le port Ethernet **f0/0**. Une connexion apparaît depuis R0. Emmenez cette connexion jusqu'à C0. Parvenu à C0, GNS3 propose l'unique port affecté à ce nuage. Sélectionnez le port proposé. Une connexion Ethernet est maintenant établie entre VMnet8 et f0/0 de R0. Cliquez à nouveau sur le bouton de la barre d'outils **Ajouter un lien** afin de sortir du mode édition de liens :



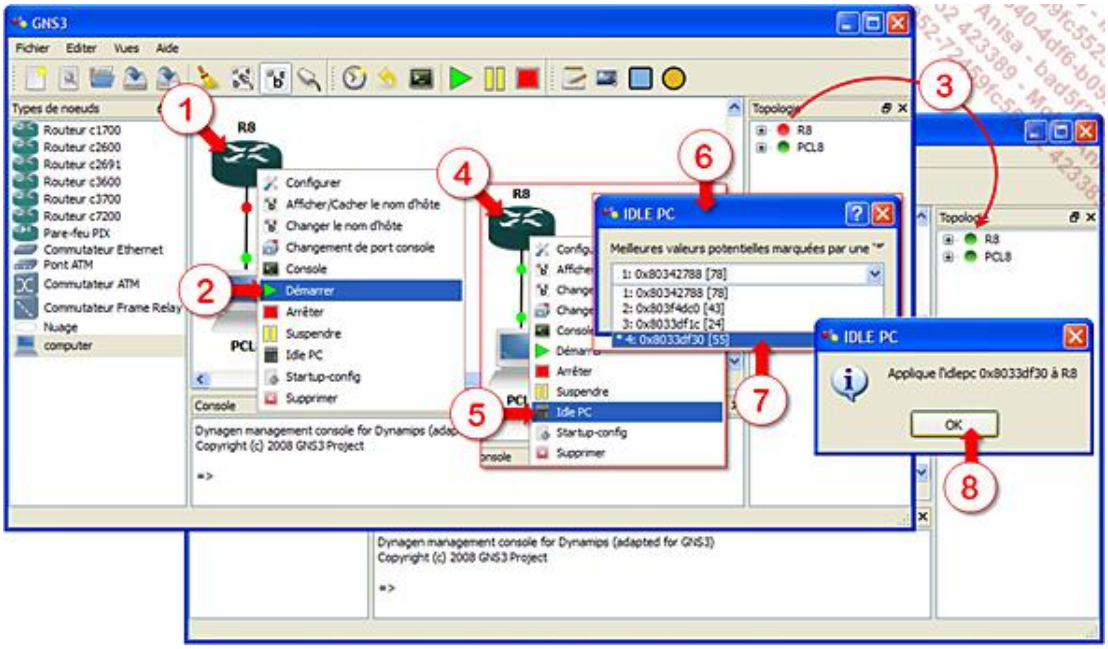
- Effectuez un clic droit sur R0 et sélectionnez **Changer le nom d'hôte**. Renommez R0 en **R8**. Faites de même afin de renommer C0 en **PCL8**. Observez également qu'il est possible de modifier la position des étiquettes afin d'éviter le chevauchement avec des liens. En final, vous devriez parvenir à une topologie comparable à celle proposée au point 5 de la figure ci-après :



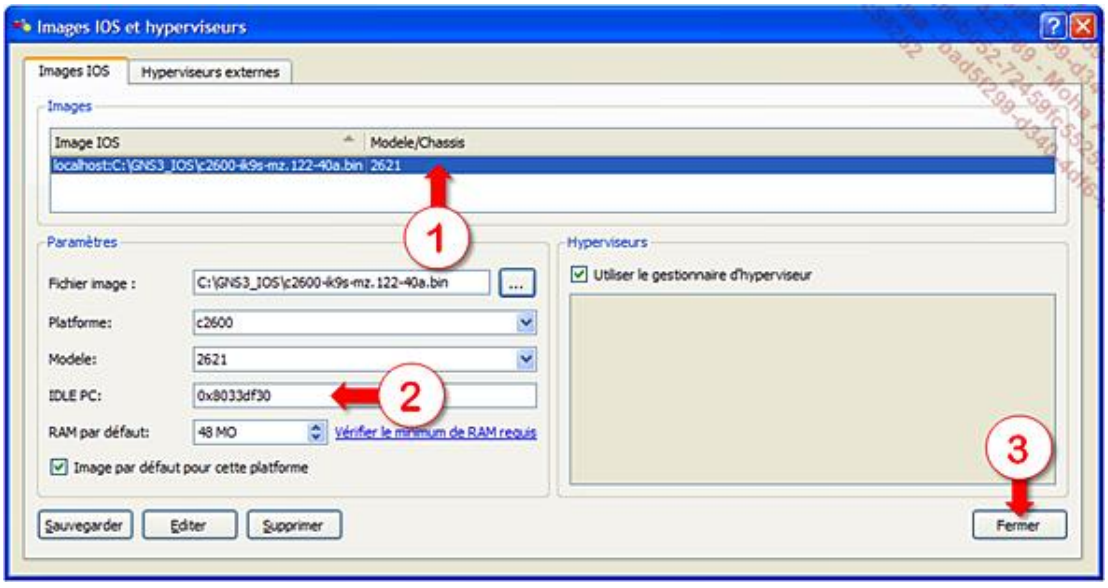
- Mettez en service puis réduisez en barre des tâches le Gestionnaire de tâches de Windows afin de surveiller l'activité du processeur de la machine virtuelle VMWKS02 :



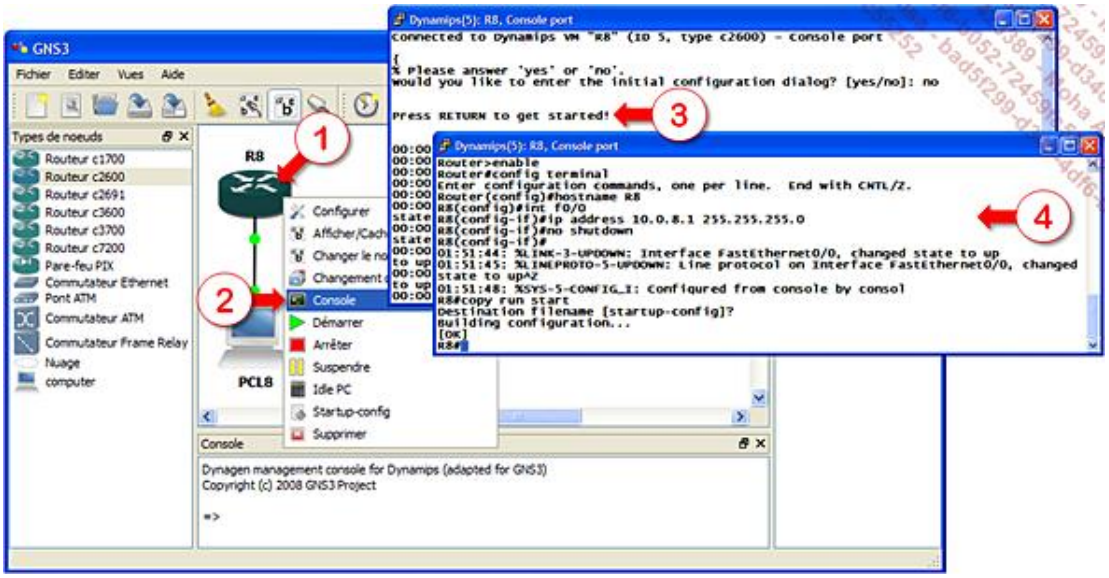
- Il est temps de provoquer le démarrage de notre nouveau routeur sorti du carton. Effectuez un clic droit sur R8 et sélectionnez **Démarrer**. Ne faites rien pendant quelques instants. Il est probable que ce démarrage consomme la plus grande partie des ressources de votre machine. Une bizarrerie de programmation fait que Dynamips réclame l'exclusivité du processeur tant qu'aucune valeur « temps mort » n'a été calculée. Effectuez un clic droit à nouveau sur **R8** et sélectionnez **IdlePC**. Dynamips se lance dans une surveillance de l'activité du processeur jusqu'à détecter des boucles de programmation où le processeur est consommé « à vide ». De cette surveillance, Dynamips déduit un certain nombre de valeurs temporelles (point 6). Les valeurs marquées d'une étoile sont celles qui présentent une probabilité potentielle de relâchement adéquat de la ressource processeur par Dynamips. Si vous n'observez aucune étoile, relancez le calcul et ce, autant de fois que nécessaire. Si c'est votre jour de chance, une, voire plusieurs valeurs IdlePC « avec étoile » apparaissent dans la liste. Sélectionnez une de ces valeurs et confirmez (point 8).



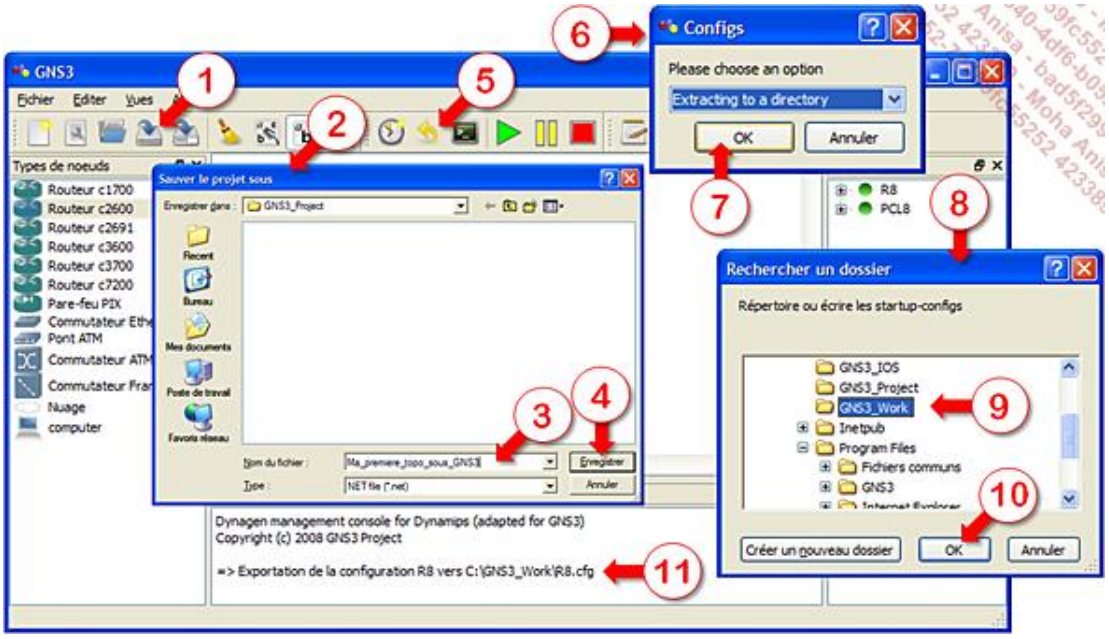
- Vous devriez observer une baisse significative de l'activité processeur à la fois pour la machine virtuelle VMWKS02 et pour la machine hôte. Lancez la commande de menu **Editer - Images IOS et hyperviseurs**. Dans la fenêtre **Images IOS et hyperviseurs**, double cliquez sur l'image associée à la plate-forme en cours d'émulation pour **R8**. Observez que la valeur IDLE PC est dorénavant renseignée (comparez avec cette même fenêtre au début de l'atelier). Ainsi, toute nouvelle instance de routeur issue de cette plate-forme et déposée sur la topologie dispose d'emblée d'une valeur IdePC, l'administrateur n'a pas à provoquer un nouveau calcul :



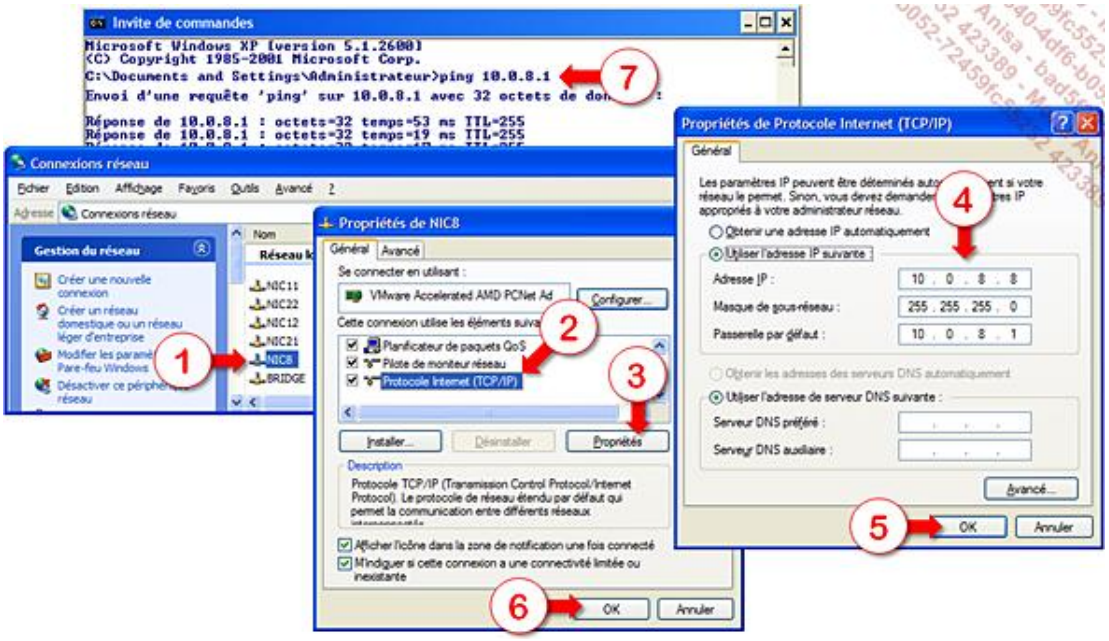
- À ce stade, nous avons bien mérité de pouvoir lancer la console et ainsi taper nos premières commandes. Effectuez un clic droit sur le routeur **R8** et sélectionnez **Console**. Si tout va bien, c'est magique, nous voilà aux commandes d'un routeur 2621. Répondez **no** à la proposition de setup, il sera toujours temps d'y revenir ensuite. Pressez la touche [Entrée] et après quelques instants interminables, R8 consent à afficher l'invite de commandes. L'interface ILC est en mode utilisateur. Passez en mode privilégié puis en mode de configuration afin de configurer l'interface f0/0, c'est-à-dire l'interface connectée à VMnet8 donc à PCL8 :



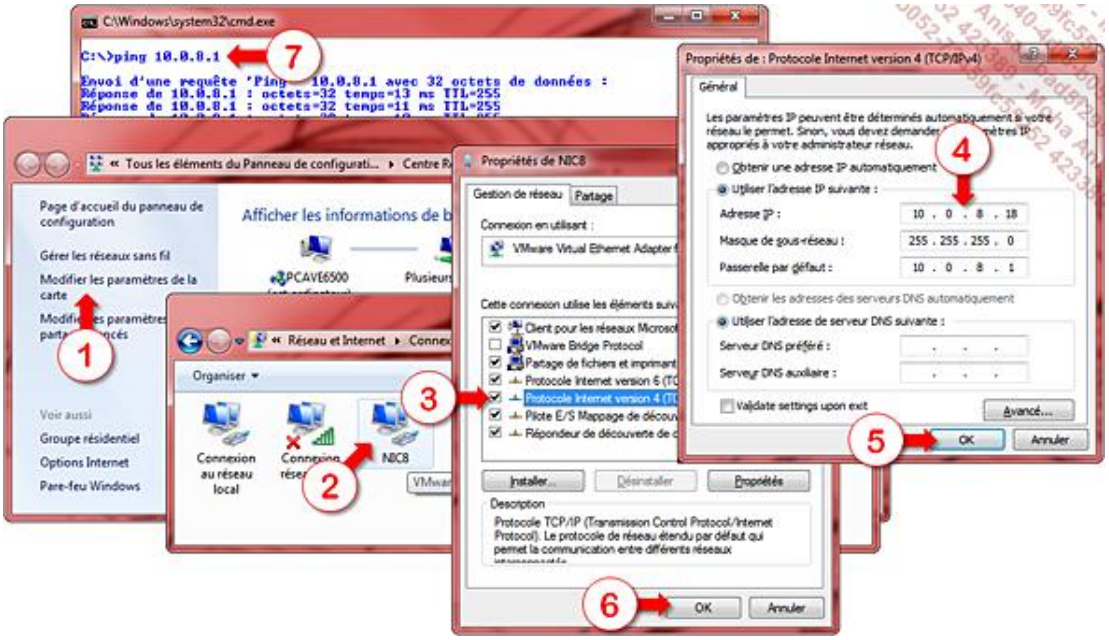
- Il est temps de sauvegarder notre projet. Cliquez sur le bouton **Sauver la topologie** de la barre d'outils. Saisissez un nom qui vous convienne et confirmez. Cliquez sur le bouton **Extract/Import all startup-configs** de la barre d'outils (point 5). Dans la fenêtre **Configs**, sélectionnez **Extracting to a directory** et confirmez. Dans la fenêtre **Rechercher un dossier**, choisissez par exemple **GNS3_Work** et confirmez. Observez le panneau **Console** situé au bas de la fenêtre GNS3 : un message confirme que l'exportation s'est bien réalisée. Désormais, chaque nouvelle sauvegarde de la topologie entraînera une extraction des fichiers de configuration **startup-configs** vers le répertoire **GNS3_Work** sans intervention de l'administrateur :



- À la condition de régler la configuration IP de l'adaptateur virtuel NIC8 dans la machine virtuelle VMWKS02, il est possible de pinguer le routeur R8 depuis VMWKS02 :

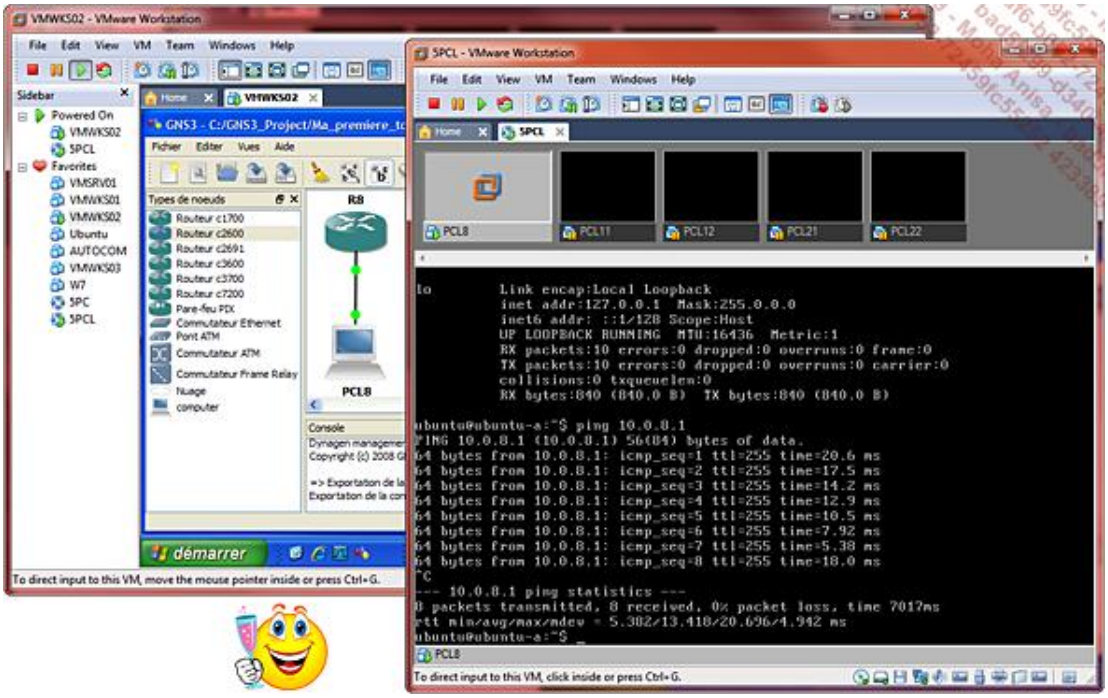


- Il se trouve que sur le concentrateur **VMnet8**, nous avons également placé un adaptateur réseau virtuel hébergé par la machine hôte. Cet adaptateur a été nommé **NIC8**. À la condition de régler la configuration IP de l'adaptateur virtuel NIC8 dans la machine hôte, il est possible de pinguer le routeur R8 depuis la machine hôte :



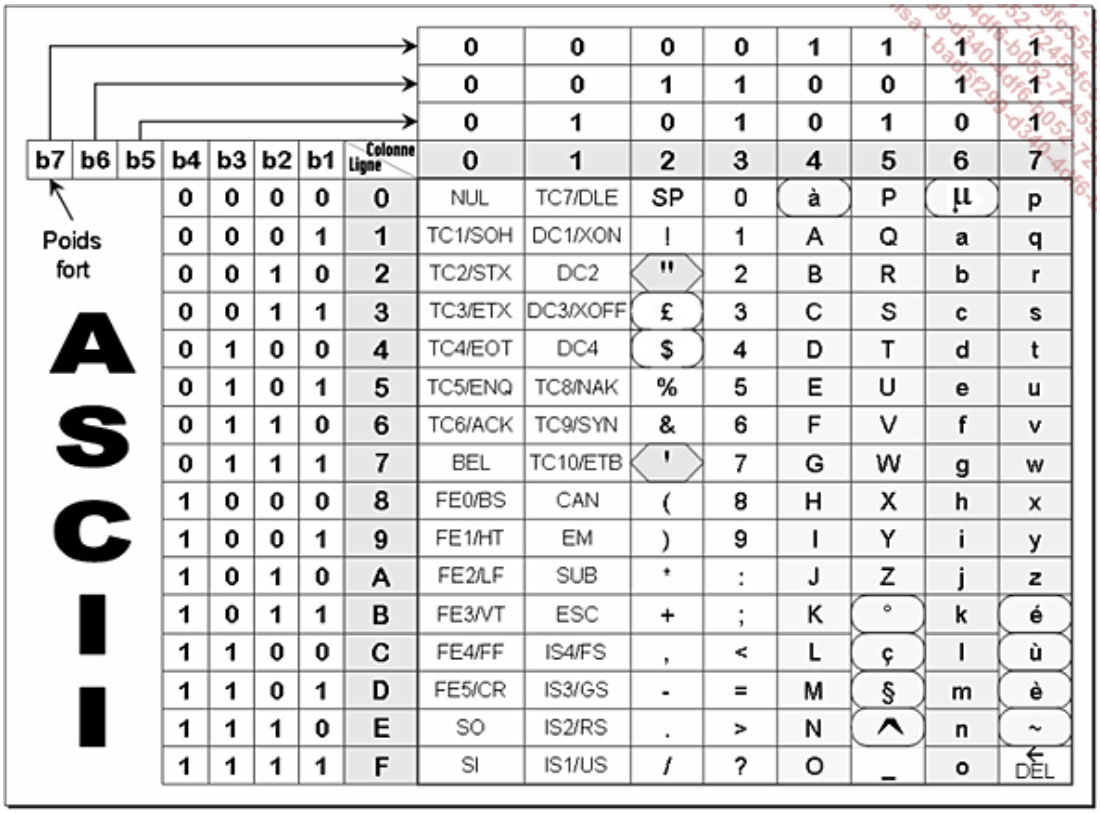
- N'activez l'adaptateur **NIC8** de la machine hôte qu'au moment de vous en servir dans l'une des mises en situation et désactivez-le ensuite. En effet, quand cet adaptateur est activé, le système d'exploitation hôte dispose de deux passerelles (celle de l'adaptateur réseau physique + celle de l'adaptateur virtuel) ce qui n'est une situation acceptable que si les passerelles sont dans le même réseau. Dans le cas contraire, il est fort probable que la machine hôte ne puisse plus par exemple sortir sur Internet.

- Mais le plus intéressant est encore à venir. Ouvrez une seconde instance de VMware à l'aide de la commande de menu **File - New - Window**. À l'aide de cette instance, ouvrez l'équipe de PC destinée aux tests. Pour le moment, seul PCL8 est utile. Réglez la configuration IP de l'adaptateur réseau virtuel NIC8 qui équipe PCL8 : {@IP → 10.0.8.2/24 ; Passerelle → 10.0.8.1}. Tentez un ping vers R8 :



- Vous êtes parvenu à ce stade. Sincèrement, félicitations et bravo de vos efforts. L'auteur a conscience que la mise en place de ce contexte était lourde. Mais avouez que les perspectives ouvertes avec cette plate-forme méritent que l'on s'acharne !

La table ASCII



La table débute par 32 caractères de contrôle. Ces caractères sont dits non visualisables car lorsqu'ils sont reçus par un équipement, celui-ci ne les affiche pas mais les interprète comme des commandes.

Sur les terminaux compatibles télétype (TTY), il est possible d'émettre un caractère des colonnes 0 et 1 par appui simultané de la touche [Ctrl] et de la touche correspondant au caractère de même rang dans les colonnes 4 et 5. Ce que rappelle la colonne key du tableau ci-dessous qui fait référence à la combinaison de touches adéquate.

La combinaison « 000 0000 » n'est pas associée à un caractère ni à une commande et n'a donc pas de signification particulière. Ainsi, un système en réception connectée à une ligne au repos ne risque pas d'interpréter les « 0 » non significatifs.

Les caractères TC1 à TC10 (*TC* = *Transmission Control*) ont été pendant un temps utilisés afin de définir des protocoles de communication. Certains caractères, tels STX et ETX, servaient de délimiteurs aux blocs de données, d'autres, tels ENQ ou ACK, servaient dans la procédure de dialogue. On peut citer le protocole BSC d'IBM (*Binary Synchronous Communications*) annoncé en 1967. L'utilisation de caractères de commande pour créer un protocole génère deux types de problèmes : 1> le protocole est lié au code, on ne peut faire évoluer l'un sans faire évoluer l'autre ; 2> si les données comprises entre deux caractères d'encadrement comportent des séquences binaires susceptibles d'être confondues avec des caractères de commande, il faudra faire précéder ces séquences par des caractères d'échappement. C'est ce qu'on appelle le problème de la transparence au code.

Les caractères FE1 à FE5 (*FE* = *Format Effector*) sont dédiés à la mise en page de l'information, qu'il s'agisse d'une imprimante ou d'un écran de terminal et comprennent le retour chariot, le changement de ligne, les tabulations horizontale et verticale, le saut de page.

Les caractères DC1 à DC4 (*DC* = *Device Control*) sont dédiés à la gestion de périphériques connectés. Deux de ces caractères sont célèbres puisqu'ils sont utilisés dans le contrôle de flux logiciel, ce sont les caractères XON (= Reprendre le flux) et XOFF (= Suspendre le flux).

Les caractères IS1 à IS4 (*IS* = *Information Separators*) permettent de hiérarchiser l'information en fichiers, articles, sous-articles…

Enfin, il reste les caractères inclassables, tel le caractère BEL qui entraîne l'émission d'un signal audible par l'équipement qui le reçoit.

Hex	Dec	Key	Name	Description
00	0	^@	NUL	Null
01	1	^A	TC1/SOH	Start of Header

02	2	^B	TC2/STX	Start of Text
03	3	^C	TC3/ETX	End of Text
04	4	^D	TC4/EOT	End of Transmission
05	5	^E	TC5/ENQ	Enquiry
06	6	^F	TC6/ACK	Acknowledge
07	7	^G	BEL	Bell
08	8	^H	FE0/BS	Backspace
09	9	^I	FE1/HT	Horizontal Tab
0A	10	^J	FE2/LF ou NL	Line Feed ou New Line*
0B	11	^K	FE3/VT	Vertical Tab
0C	12	^L	FE4/FF	Form Feed
0D	13	^M	FE5/CR	Carriage Return
0E	14	^N	SO	Shift Out
0F	15	^O	SI	Shift In
10	16	^P	TC7/DLE	Data Link Escape
11	17	^Q	DC1/XON	Device Control 1
12	18	^R	DC2	Device Control 2
13	19	^S	DC3/XOFF	Device Control 3
14	20	^T	DC4	Device Control 4
15	21	^U	TC8/NAK	Negative Acknowledge
16	22	^V	TC9/SYN	SynchronousIdle
17	23	^W	TC10/ETB	End Transmission Block
18	24	^X	CAN	Cancel
19	25	^Y	EM	End of Medium
1A	26	^Z	SUB	Substitute
1B	27	^[	ESC	Escape
1C	28	^\	IS4/FS	File Separator
1D	29	^]	IS3/GS	Group Separator
1E	30	^^	IS2/RS	Record Separator
1F	31	^_	IS1/US	Unit Separator

*Certains appareils ne comportent qu'une seule commande pour l'opération combinée de retour chariot et de saut de ligne, la fonction FE2 prend alors la signification NL (*New Line*).

Numérotation des interfaces des routeurs de la série 2800

Dans le cas du routeur 2801, la numérotation s'établit ainsi :

Numéro de slot	Type de slot	Étendue de numérotation
Ports embarqués	Fast Ethernet	0/0 et 0/1.
0	VIC/VWIC (voix seulement)	De 0/0/0 à 0/0/3.
1	HWIC / WIC / VIC / VWIC (*)	De 0/1/0 à 0/1/3 dans le cas d'un module HWIC simple largeur. De 0/1/0 à 0/1/7 dans le cas d'un module HWIC double largeur.
2	WIC / VIC / VWIC (*)	De 0/2/0 à 0/2/3.
3	HWIC / WIC / VIC / VWIC (*)	De 0/3/0 à 0/3/3 dans le cas d'un module HWIC simple largeur. De 0/3/0 à 0/3/7 dans le cas d'un module HWIC double largeur.

(*) Un module VWIC placé dans l'un des slots 1, 2 ou 3 peut fonctionner indifféremment dans les modes donnée et voix. Ce même module placé dans le slot 0 ne peut fonctionner que dans le mode voix.

Dans le cas des routeurs 2811, 2821 et 2851, le nommage s'établit ainsi :

Emplacement du port	Format de la numérotation	Exemples
Embarqué, face avant.	Interface-type ▲ port	usb ▲ 0 usb ▲ 1
Embarqué, face arrière.	Interface-type ▲ 0/port	Interface ▲ fa ▲ 0/x Interface ▲ gi ▲ 0/x
Sur une carte d'interface (HWIC, HWIC-D, WIC, VWIC, VIC) installée directement dans un slot HWIC du châssis.	Interface-type ▲ 0/slot/port	interface ▲ serial ▲ 0/x/y interface ▲ async ▲ 0/x/y line ▲ 0/x/y interface ▲ fa ▲ 0/x/y voice-port ▲ 0/x/y
Sur une carte d'interface (WIC, VWIC, VIC) installée dans un slot appartenant à un module NM.	Interface-type ▲ 1/slot/port « 1 » identifie le module NM sur tous les routeurs de la série 2800.	controller ▲ t1 ▲ 1/x/y voice-port ▲ 1/x/y interface ▲ serial ▲ 1/x/y interface ▲ async ▲ 1/x/y line ▲ 1/x/y
Embarqué directement sur le module NM (NME, NME-X, NMD, NMD-XD).	Interface-type ▲ 1/port « 1 » identifie le module NM sur tous les routeurs de la série 2800.	interface ▲ gi ▲ 1/x interface ▲ serial ▲ 1/x interface ▲ async ▲ 1/x line ▲ 1/x
Port FXS ou FXO installé	Interface-type ▲ 2/0/port	voice-port ▲ 2/0/x

sur un module EVM (voix).	« 2 » identifie le module EVM sur tous les routeurs 2821 et 2851 (autres routeurs non concernés). FXS/DID, les ports 0 à 7 sont embarqués directement sur le module EVM. FXS/FX0, les ports 8 à 15 appartiennent au module d'extension 0. FXS/FX0, les ports 16 à 23 appartiennent au module d'extension 1. Le chiffre « 0 » au deuxième rang du nommage est requis par la syntaxe imposée pour un module EVM mais n'identifie pas un slot sur le module.	
Ports voix dans une extension BRI (*Basic Rate Interface*, l'une des interfaces du réseau RNIS) placée dans un module EVM.	Interface-type ▲ 2/0/port « 2 » identifie le module EVM sur tous les routeurs 2821 et 2851 (autres routeurs non concernés). Les ports 8 à 11 appartiennent au module d'extension 0. Les ports 16 à 19 appartiennent au module d'extension 1. Le chiffre « 0 » au deuxième rang du nommage est requis par la syntaxe imposée pour un module EVM mais n'identifie pas un slot sur le module.	interface ▲ bri ▲ 2/x

Une partie des acronymes contenus dans ce tableau est certainement déjà connue à ce stade. À tout hasard :

WIC	WAN Interface Card.
HWIC	High-Speed WIC.
VWIC	Voice WIC.
VIC	Voice Interface Card.
NM	Network Module.
NME	Network Module Enhanced.
BRI	Basic Rate Interface.
RNIS	Réseau Numérique à Intégration de Services.
FXS	Foreign eXchange Subscriber. Port qui amène la ligne téléphonique de l'abonné. Cette interface fournit notamment la tonalité, le courant du mode décroché, la tension du mode raccroché, la tension de sonnerie. Un téléphone analogique classique, branché sur cette interface, reçoit le service téléphonique.
FXO	Foreign eXchange Office. Extrémité du câble permettant de relier un appareil, tel un téléphone ou un télécopieur, au port FXS. On parle souvent de périphérique FXO. FXO et FXS vont toujours de pair.

Quelques notions sur la représentation binaire signée

1. Code complément à 1 ou complément restreint

Ce code n'a d'autre utilité que celle d'introduire le code complément à 2.

- Le complément à 1 de 0 est 1.
- Le complément à 1 de 1 est 0.

Pour exprimer le complément à 1 d'un mot binaire, il suffit donc de complémenter chaque bit du mot.

Exemple : le complément à 1 de 10010 est 01101.

La nécessité d'introduire les codes complément à 1 et complément à 2 provient de la solution choisie pour représenter les nombres négatifs ou de la solution choisie pour réaliser une soustraction binaire. Si la plus petite entité manipulable par une machine informatique est le bit, la seule opération arithmétique réalisable est l'addition. Pour faire une soustraction, on additionne le nombre négatif correspondant au nombre que l'on voulait soustraire. La question à se poser est donc « comment représenter les nombres négatifs en binaire ? »

Supposons un nombre binaire x = 0101

Notons son complément à 1 $\bar{x}$. $\bar{x}=1010$

Réalisons l'addition de x et de son complément à 1 :

$x=$	0	1	0	1	
$\bar{x}=$	1	0	1	0	
$x+\bar{x}=$	1	1	1	1	$=15_{(10)}$

On constate que $x+\bar{x}=15_{(10)} \Leftrightarrow \bar{x}=15_{(10)}-x$.

Ce faisant, on a introduit le signe moins ce qui est satisfaisant mais on a également introduit une constante ($15_{(10)}$) dont il faut se débarrasser, ce que permet le code complément à 2.

2. Code complément à 2 ou complément vrai

Le complément à 2 est égal au complément à 1 + 1.

Ajoutons 1 au complément à 1 de x et notons $\bar{\bar{x}}$ le résultat :

$\bar{x}=$		1	0	1	0	
+					1	
$\bar{\bar{x}}=\bar{x}+1=$		1	0	1	1	

Puis, additionnons x et son complément à 2 $\bar{\bar{x}}$:

$x=$		0	1	0	1	
$\overline{=}$ $x=$		1	0	1	1	
$x+\overline{=}x=$	1	0	0	0	0	$=16_{(10)}$

On constate que $x+\overline{=}x=16_{(10)} \Leftrightarrow \overline{=}x=16_{(10)}-x$.

La constante est toujours là mais si l'on admet que le domaine de définition des mots binaires est celui des mots à 4 bits (pour cet exemple), il faut ignorer le cinquième bit ce qui revient à ignorer la constante. Ce faisant, on obtient : $\overline{=}x=-x$.

Exemple

Supposons que l'on désire réaliser l'opération binaire correspondant à l'opération décimale suivante :

7
-5
=2

Sur 4 bits, le mot binaire représentant 5 est 0101.

Son complément à 1 est 1010. Le nombre négatif représentant -5 est donc 1011 (1010 + 1).

Il reste à additionner le mot binaire représentant 7 et le mot binaire représentant -5 :

	0	1	1	1
+	1	0	1	1
= 1	0	0	1	0

En ignorant le cinquième bit, le résultat est 0010 ce qui est bien le mot binaire représentant 2.

Lorsque le complément à 2 est utilisé, il faut impérativement connaître le format (nombre de bits) utilisé. En effet, le complément à 2 d'un mot binaire ne sera pas le même selon que le format est 4, 8, 16 ou 32 bits.

Toujours sur 4 bits, essayons d'envisager ce que deviennent les 16 combinaisons possibles en complément à 2 :

Pour ce faire, construisons un tableau à quatre colonnes. Les deux premières colonnes contiendront les nombres positifs en représentation décimale et binaire. Pour chaque mot binaire représentant un nombre positif, on place dans les deux dernières colonnes, son complément à 2 ainsi que le nombre négatif correspondant :

Nombre positif	Mot binaire positif	Complément à 2	Nombre négatif
0	0000	0000	0
1	0001	1111	-1
2	0010	1110	-2
3	0011	1101	-3
4	0100	1100	-4

5	0101	1011	-5
6	0110	1010	-6
7	0111	1001	-7
		1000	-8

On constate ainsi que sur 4 bits et en complément à 2, il est possible de représenter les nombres décimaux de -8 à +7. On pourrait penser que les nombres négatifs sont privilégiés mais si l'on veut bien admettre que 0 est positif, alors il y a autant de nombres positifs que de nombres négatifs.

Sans que cela prête à conséquence, le complément à 2 des mots binaires représentant les nombres négatifs donne à nouveau les nombres positifs (le complément à 2 de 1110→-2 est 0010→+2) sauf dans le cas du nombre négatif -8 car le complément à 2 de 1000 est 1000.

Enfin, il faut constater que le code complément à 2 est toujours un code pondéré. Pour trouver les poids correspondant à chaque bit, il suffit d'observer les quatre cas où un seul bit est vrai : 0001, 0010, 0100 et 1000. On remarque ainsi que si les poids binaires des trois bits de poids faible sont toujours 1, 2, 4, le poids binaire du quatrième bit est -8.

Nécessairement, puisque 0111 → 7 et 1000 → -8, lorsque le quatrième bit est à 1, on peut conclure que le nombre représenté est négatif. Ceci apparaît clairement dans le tableau ordonné suivant :

POIDS				**NOMBRE DÉCIMAL**
- 8	4	2	1	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	-8
1	0	0	1	-7
1	0	1	0	-6
1	0	1	1	-5
1	1	0	0	-4
1	1	0	1	-3
1	1	1	0	-2
1	1	1	1	-1

Ceci fait parfois appeler le bit de poids fort bit de signe.

Sur quatre bits, en binaire naturel, on pouvait représenter les nombres décimaux de 0 à 15. En complément à 2, les nombres décimaux s'étendent de -8 à +7. Essayons d'imaginer l'étendue de représentation lorsqu'on passe à 8, 16

ou 32 bits :

Format	Binaire naturel	Complément à 2
4 bits	0 à 15	-8 à +7
8 bits	0 à 255	-128 à +127
12 bits	0 à 4095	-2048 à +2047
16 bits	0 à 65535	-32768 à +32767
32 bits	0 à 4.294.967.295	-2.147.483.648 à +2.147.483.647
n bits	0 à $2^n - 1$	-2^{n-1} à $+ 2^{n-1} -1$

Adresses de multidiffusion multicast

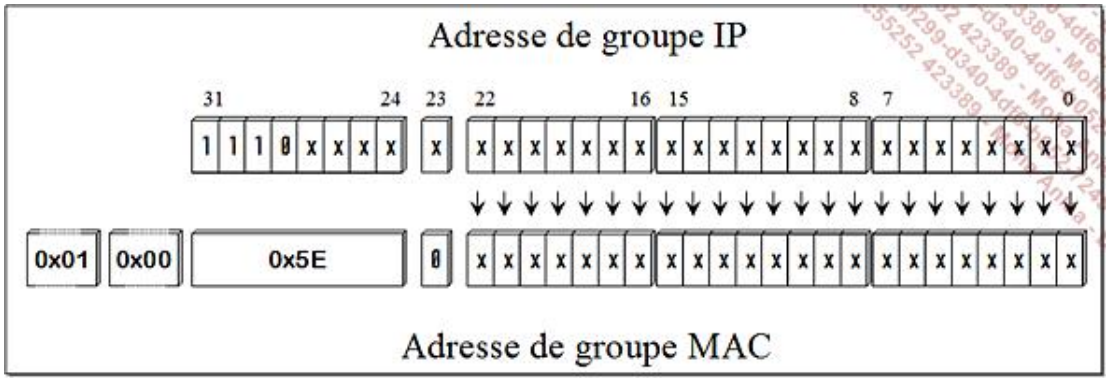
Le bloc **224.0.0.0/4**, ex-adresses de classe D de 224 à 239, est dédié aux adresses de multi diffusion. À l'intérieur de ce bloc, la subdivision en partie réseau et partie hôte de l'adresse est sans objet : les 4 premiers bits sont « 1110 » et les 28 bits restants désignent le groupe. Sans fournir le niveau de détail du RFC3171, il est utile de connaître le bloc « Local Network Control Block » :

224.0.0.0/24 → La traduction est dangereuse, le bloc est réservé par l'IANA sous le nom « Local Network Control Block ». Ces adresses sont allouées aux protocoles chargés de la gestion du réseau, tels les protocoles de routage, des protocoles de découverte de topologies, des protocoles de maintenance. Tout comme dans le cas des adresses « locales-liens », les paquets émis vers des adresses du bloc 224.0.0.0/24 le sont avec une durée de vie de 1 et restent donc locaux. Exemples d'adresse dans ce bloc : **224.0.0.0** est réservée ; **224.0.0.1** désigne toutes les interfaces présentes sur le réseau local (*All systems on this subnet*) ; **224.0.0.2** désigne toutes les interfaces de routeurs présentes sur le réseau local (*All routers on this subnet*). Les autres affectations peuvent être consultées sur le site de l'IANA : http://www.iana.org/assignments/multicast-addresses/multicast-addresses.xml

Comment reconnaître la trame qui encapsule un paquet envoyé vers une adresse IP multicast ? Il faut se souvenir que la carte Ethernet en réception, ne se saisit que des trames qui lui sont destinées (dont l'adresse de destination correspond à l'adresse MAC de la carte) et des trames qui sont émises en diffusion (dont l'adresse de destination correspond à FF-FF-FF-FF-FF-FF.

Donc pour envoyer un paquet vers une adresse multi diffusion, la première possibilité consistait à l'encapsuler dans une trame dont l'adresse de destination est l'adresse de diffusion de couche 2. L'inconvénient est qu'une interface n'appartenant pas au groupe ne le découvrira qu'en couche 3, imposant une surcharge inutile du processeur de la machine.

L'idéal serait de pouvoir répercuter l'appartenance à un groupe de multi diffusion sur la couche 2. Il se trouve que l'IANA disposait d'une plage d'adresses IEEE (un OUI) : 00-00-5E-XX-XX-XX. Il fut décidé d'en réserver la moitié (23 bits sur 24) pour le multicast, les valeurs retenues sont 01-00-5E-00-00-00 à 01-00-5E-7F-FF-FF. Le premier octet peut sembler différent, il s'agit simplement du bit I/G qui est positionné à « 1 » pour rappeler que l'on a affaire à une adresse de groupe. Le passage de l'adresse de groupe IP à l'adresse de groupe MAC s'effectue de la façon suivante :

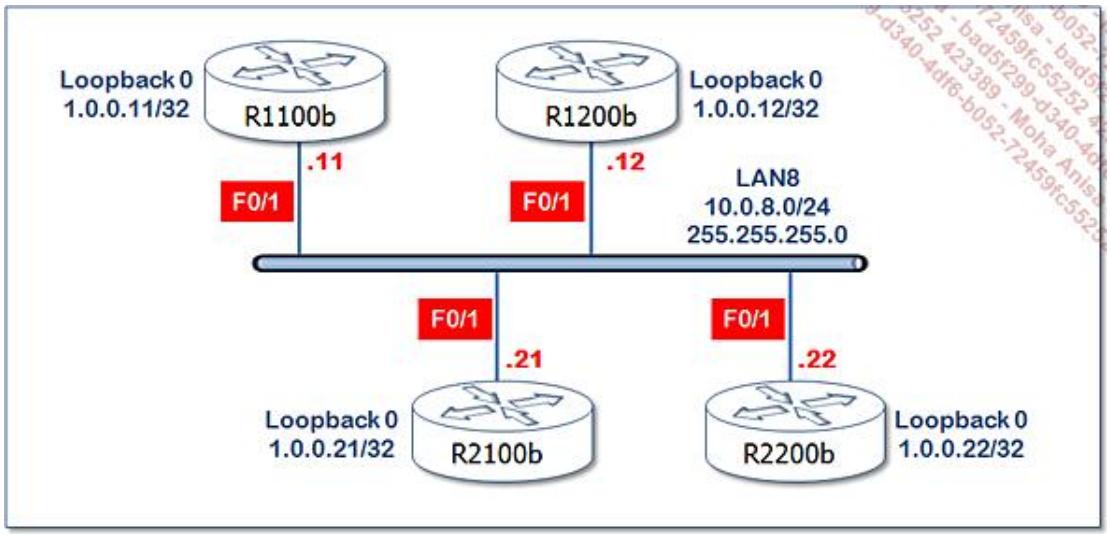


Les 23 bits de poids faible de l'adresse de groupe IP sont placés dans les 23 bits de l'adresse de groupe MAC. On objectera que sur les 32 bits de l'adresse IP d'un groupe, 28 désignent le groupe et que par conséquent plusieurs groupes IP pourraient avoir la même adresse de groupe MAC. En effet, la conséquence est que même si la couche 2 fait remonter le paquet parce qu'elle a reconnu l'adresse de groupe, la couche 3 doit quand même vérifier qu'elle est effectivement concernée par le paquet en question.

Configuration OSPF des modes NBMA et Point à multipoint

1. Réseau physique à diffusion, mode broadcast

a. Contexte



b. Captures dans l'interface ILC

```
R1100b#
00:01:07: %OSPF-5-ADJCHG: Process 1, Nbr 1.0.0.21 on FastEthernet0/1 from
LOADING to FULL, Loading Done
R1100b#
00:01:09: OSPF: Neighbor change Event on interface FastEthernet0/1
00:01:09: OSPF: DR/BDR election on FastEthernet0/1
00:01:09: OSPF: Elect BDR 1.0.0.21
00:01:09: OSPF: Elect DR 1.0.0.22
00:01:09:        DR: 1.0.0.22 (Id)   BDR: 1.0.0.21 (Id)
```

```
R1100b#sh ip ospf database router 1.0.0.11

            OSPF Router with ID (1.0.0.11) (Process ID 1)

                Router Link States (Area 0)

  LS age: 166
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 1.0.0.11
  Advertising Router: 1.0.0.11
  LS Seq Number: 80000002
  Checksum: 0x347A
  Length: 48
  Number of Links: 2

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.8.22
     (Link Data) Router Interface address: 10.0.8.11
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 10.0.11.0
```

```
    (Link Data) Network Mask: 255.255.255.0
     Number of TOS metrics: 0
      TOS 0 Metrics: 1
```

```
R1100b#sh ip ospf database router 1.0.0.12

            OSPF Router with ID (1.0.0.11) (Process ID 1)

                Router Link States (Area 0)

  LS age: 220
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 1.0.0.12
  Advertising Router: 1.0.0.12
  LS Seq Number: 80000002
  Checksum: 0x4763
  Length: 48
  Number of Links: 2

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.8.22
     (Link Data) Router Interface address: 10.0.8.12
      Number of TOS metrics: 0
       TOS 0 Metrics: 1

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 10.0.12.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1
```

```
R1100b#sh ip ospf database

            OSPF Router with ID (1.0.0.11) (Process ID 1)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
1.0.0.11        1.0.0.11        540         0x80000002 0x00347A 2
1.0.0.12        1.0.0.12        541         0x80000002 0x004763 2
1.0.0.21        1.0.0.21        541         0x80000002 0x00EBCA 1
1.0.0.22        1.0.0.22        541         0x80000002 0x00E9C9 1

                Net Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum
10.0.8.22       1.0.0.22        541         0x80000001 0x006E3E
```

```
R1100b#sh ip ospf database network 10.0.8.22

            OSPF Router with ID (1.0.0.11) (Process ID 1)

                Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 597
  Options: (No TOS-capability, DC)
  LS Type: Network Links
  Link State ID: 10.0.8.22 (address of Designated Router)
  Advertising Router: 1.0.0.22
  LS Seq Number: 80000001
  Checksum: 0x6E3E
  Length: 40
  Network Mask: /24
        Attached Router: 1.0.0.22
        Attached Router: 1.0.0.21
        Attached Router: 1.0.0.12
```

```
Attached Router: 1.0.0.11
```

```
R1100b#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.22          1   FULL/DR         00:00:39    10.0.8.22  FastEthernet0/1
1.0.0.21          1   FULL/BDR        00:00:37    10.0.8.21  FastEthernet0/1
1.0.0.12          1   2WAY/DROTHER    00:00:32    10.0.8.12  FastEthernet0/1
```

```
R1200b#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.11          1   2WAY/DROTHER    00:00:36    10.0.8.11  FastEthernet0/1
1.0.0.22          1   FULL/DR         00:00:34    10.0.8.22  FastEthernet0/1
1.0.0.21          1   FULL/BDR        00:00:31    10.0.8.21  FastEthernet0/1
```

```
R2100b#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.12          1   FULL/DROTHER    00:00:36    10.0.8.12  FastEthernet0/1
1.0.0.11          1   FULL/DROTHER    00:00:35    10.0.8.11  FastEthernet0/1
1.0.0.22          1   FULL/DR         00:00:34    10.0.8.22  FastEthernet0/1
```

```
R2200b#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.21          1   FULL/BDR        00:00:34    10.0.8.21  FastEthernet0/1
1.0.0.12          1   FULL/DROTHER    00:00:38    10.0.8.12  FastEthernet0/1
1.0.0.11          1   FULL/DROTHER    00:00:37    10.0.8.11  FastEthernet0/1
```

c. Captures Wireshark

Les quatre routeurs montés quasi simultanément dans l'ordre R1100, R1200, R2100, R2200.

- Cap_2I_01.pcap

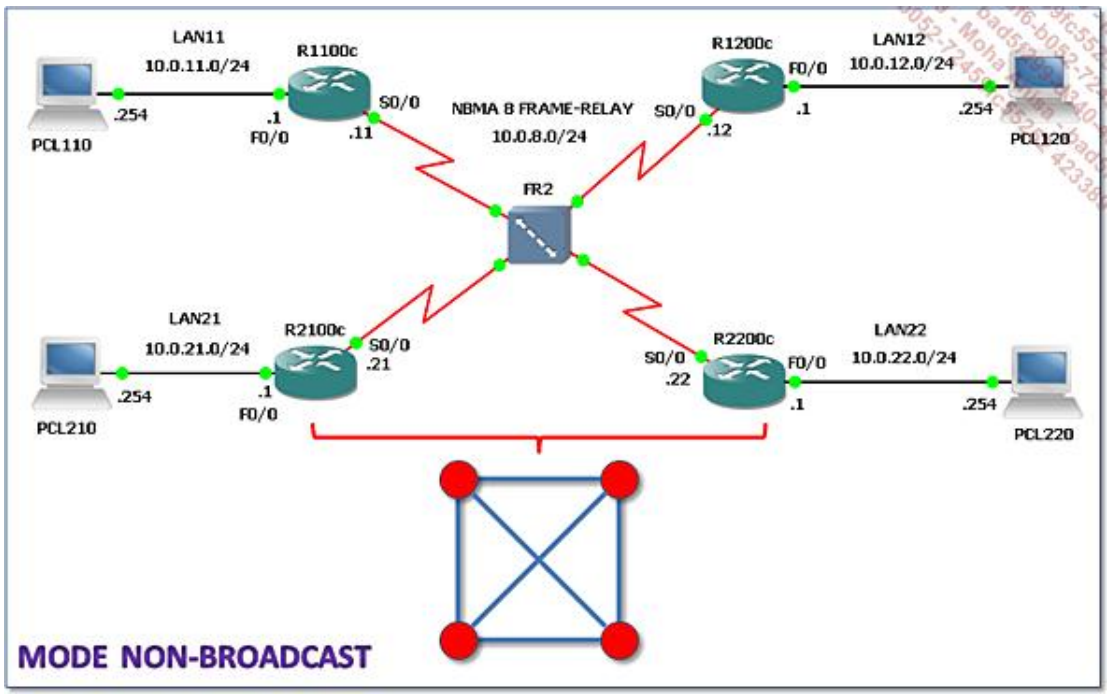
Capture sur plus de 30 minutes pour vérifier l'échange de LSD toutes les 30 minutes :

- Cap_2I_02.pcap

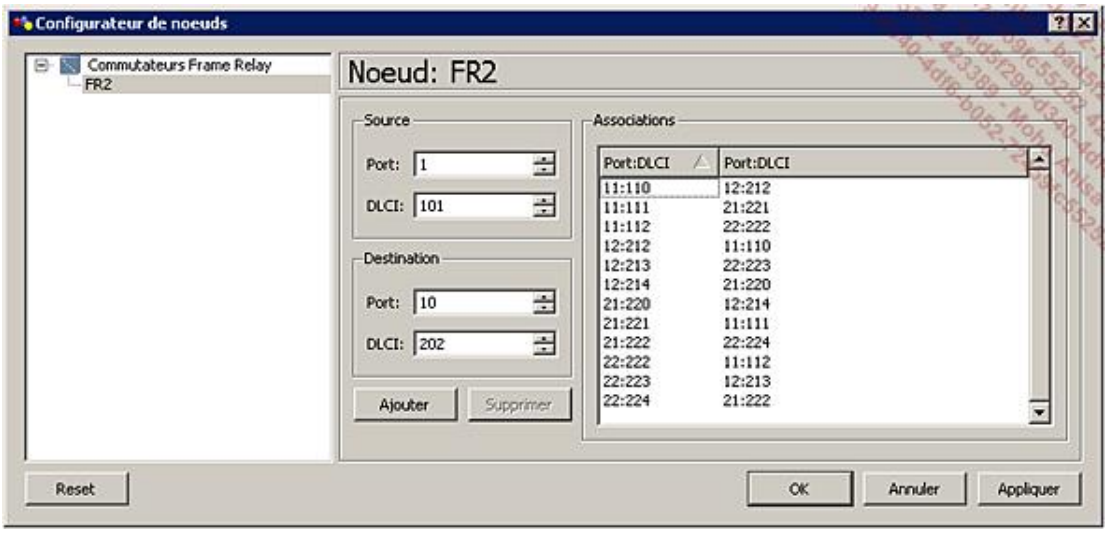
2. Mode NBMA complètement maillé

Mais configuré en non-broadcast.

a. Contexte



La configuration du commutateur FrameRelay sous GNS3 :



b. Les configurations de routeurs

R1100c :

```
R1100c#sh run
.........
interface Loopback0
ip address 1.0.0.11 255.255.255.255
!
interface FastEthernet0/0
ip address 10.0.11.1 255.255.255.0
ip ospf network non-broadcast
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.8.11 255.255.255.0
encapsulation frame-relay
ip ospf network non-broadcast
frame-relay map ip 10.0.8.12 110
```

```
frame-relay map ip 10.0.8.21 111
frame-relay map ip 10.0.8.22 112
!
router ospf 1
log-adjacency-changes
network 10.0.8.0 0.0.31.255 area 0
neighbor 10.0.8.22 priority 1
neighbor 10.0.8.21 priority 1
neighbor 10.0.8.12 priority 1
!
.........
R1100c#
```

R12OOc :

```
R1200c#sh run
.........
interface Loopback0
ip address 1.0.0.12 255.255.255.255
!
interface FastEthernet0/0
ip address 10.0.12.1 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.8.12 255.255.255.0
encapsulation frame-relay
ip ospf network non-broadcast
frame-relay map ip 10.0.8.11 212
frame-relay map ip 10.0.8.21 214
frame-relay map ip 10.0.8.22 213
!
router ospf 1
log-adjacency-changes
network 10.0.8.0 0.0.31.255 area 0
neighbor 10.0.8.22 priority 1
neighbor 10.0.8.21 priority 1
neighbor 10.0.8.11 priority 1
!
.........

R1200c#
```

R21OOc :

```
R2100c#sh run
.........
interface Loopback0
ip address 1.0.0.21 255.255.255.255
!
interface FastEthernet0/0
ip address 10.0.21.1 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.8.21 255.255.255.0
encapsulation frame-relay
ip ospf network non-broadcast
frame-relay map ip 10.0.8.11 221
frame-relay map ip 10.0.8.12 220
frame-relay map ip 10.0.8.22 222
!
router ospf 1
log-adjacency-changes
network 10.0.8.0 0.0.31.255 area 0
neighbor 10.0.8.22 priority 1
neighbor 10.0.8.12 priority 1
```

```
neighbor 10.0.8.11 priority 1
!
R2100c#
```

R2200c :

```
R2200c#sh run
.........
!
interface Loopback0
ip address 1.0.0.22 255.255.255.255
!
interface FastEthernet0/0
ip address 10.0.22.1 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.8.22 255.255.255.0
encapsulation frame-relay
ip ospf network non-broadcast
frame-relay map ip 10.0.8.11 222
frame-relay map ip 10.0.8.12 223
frame-relay map ip 10.0.8.21 224
!
router ospf 1
log-adjacency-changes
network 10.0.8.0 0.0.31.255 area 0
neighbor 10.0.8.21 priority 1
neighbor 10.0.8.12 priority 1
neighbor 10.0.8.11 priority 1
!
.........
```

c. Captures dans l'interface ILC

L'activité montrée à l'aide d'une commande **debug ip ospf adj** sur R1100c :

```
00:01:13: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state
to up
00:01:13: OSPF: Interface Serial0/0 going Up
00:01:13: OSPF: Starting 0.0.0.0 address 10.0.8.22 on Serial0/0
00:01:13: OSPF: Starting 0.0.0.0 address 10.0.8.21 on Serial0/0
00:01:13: OSPF: Starting 0.0.0.0 address 10.0.8.12 on Serial0/0
00:01:47: OSPF: 2 Way Communication to 1.0.0.12 on Serial0/0, state 2WAY
00:01:52: OSPF: 2 Way Communication to 1.0.0.21 on Serial0/0, state 2WAY
00:01:56: OSPF: 2 Way Communication to 1.0.0.22 on Serial0/0, state 2WAY
00:02:04: OSPF: end of Wait on interface FastEthernet0/0
00:02:04: OSPF: DR/BDR election on FastEthernet0/0
00:02:04: OSPF: Elect BDR 1.0.0.11
00:02:04: OSPF: Elect DR 1.0.0.11
00:02:04: OSPF: Elect BDR 0.0.0.0
00:02:04: OSPF: Elect DR 1.0.0.11
00:02:04:        DR: 1.0.0.11 (Id)   BDR: none
00:02:05: OSPF: No full nbrs to build Net Lsa for interface FastEthernet0/0
00:03:13: OSPF: end of Wait on interface Serial0/0
00:03:13: OSPF: DR/BDR election on Serial0/0
00:03:13: OSPF: Elect BDR 1.0.0.22
00:03:13: OSPF: Elect DR 1.0.0.22
00:03:13:        DR: 1.0.0.22 (Id)   BDR: 1.0.0.22 (Id)
00:03:13: OSPF: Send DBD to 1.0.0.22 on Serial0/0 seq 0xDA0 opt 0x4en
00:03:18: OSPF: Send DBD to 1.0.0.22 on Serial0/0 seq 0xDA0 opt 0x42 flag 0x7 len 32
00:03:18: OSPF: Retransmitting DBD to 1.0.0.22 on Serial0/0 [1]
00:03:23: OSPF: Send DBD to 1.0.0.22 on Serial0/0 seq 0xDA0 opt 0x42 flag 0x7 len 32
00:03:23: OSPF: Retransmitting DBD to 1.0.0.22 on Serial0/0 [2]
00:03:26: OSPF: Send DBD to 1.0.0.21 on Serial0/0 seq 0x2EE opt 0x42 flag 0x7 len 32
00:03:26: OSPF: Rcv DBD from 1.0.0.22 on Serial0/0 seq 0x174E opt 0x42 flag 0x7
```

```
len 32  mtu 1500 state EXSTART
00:03:26: OSPF: NBR Negotiation Done. We are the SLAVE
00:03:26: OSPF: Send DBD to 1.0.0.22 on Serial0/0 seq 0x174E opt 0x42 flag 0x2 len 52
00:03:26: OSPF: Rcv DBD from 1.0.0.21 on Serial0/0 seq 0x69B opt 0x42 flag 0x7
len 32  mtu 1500 state EXSTART
00:03:26: OSPF: NBR Negotiation Done. We are the SLAVE
00:03:26: OSPF: Send DBD to 1.0.0.21 on Serial0/0 seq 0x69B opt 0x42 flag 0x2 len 52
00:03:26: OSPF: Rcv DBD from 1.0.0.21 on Serial0/0 seq 0x69C opt 0x42 flag 0x3
len 52  mtu 1500 state EXCHANGE
00:03:26: OSPF: Send DBD to 1.0.0.21 on Serial0/0 seq 0x69C opt 0x42 flag 0x0 len 32
00:03:26: OSPF: Database request to 1.0.0.21
00:03:26: OSPF: sent LS REQ packet to 10.0.8.21, length 12
00:03:26: OSPF: Rcv DBD from 1.0.0.21 on Serial0/0 seq 0x69D opt 0x42 flag 0x1
len 32  mtu 1500 state EXCHANGE
00:03:26: OSPF: Exchange Done with 1.0.0.21 on Serial0/0
00:03:26: OSPF: Send DBD to 1.0.0.21 on Serial0/0 seq 0x69D opt 0x42 flag 0x0 len 32
00:03:26: OSPF: Synchronized with 1.0.0.21 on Serial0/0, state FULL
00:03:26: %OSPF-5-ADJCHG: Process 1, Nbr 1.0.0.21 on Serial0/0 from LOADING to FULL,
Loading Done
00:03:26: %OSPF: Rcv DBD from 1.0.0.22 on Serial0/0 seq 0x174F opt 0x42 flag 0x3
len 52  mtu 1500 state EXCHANGE
00:03:26: OSPF: Send DBD to 1.0.0.22 on Serial0/0 seq 0x174F opt 0x42 flag 0x0 len 32
00:03:26: OSPF: Database request to 1.0.0.22
00:03:26: OSPF: sent LS REQ packet to 10.0.8.22, length 12
00:03:26: OSPF: Rcv DBD from 1.0.0.22 on Serial0/0 seq 0x1750 opt 0x42 flag 0x1
len 32  mtu 1500 state EXCHANGE
00:03:26: OSPF: Exchange Done with 1.0.0.22 on Serial0/0
00:03:26: OSPF: Send DBD to 1.0.0.22 on Serial0/0 seq 0x1750 opt 0x42 flag 0x0 len 32
00:03:26: OSPF: Synchronized with 1.0.0.22 on Serial0/0, state FULL
00:03:26: %OSPF-5-ADJCHG: Process 1, Nbr 1.0.0.22 on Serial0/0 from LOADING to FULL,
Loading Done
00:03:27: OSPF: Build router LSA for area 0,router ID 1.0.0.11,seq0x80000004
00:03:47: OSPF: Neighbor change Event on interface Serial0/0
00:03:47: OSPF: DR/BDR election on Serial0/0
00:03:47: OSPF: Elect BDR 1.0.0.21
00:03:47: OSPF: Elect DR 1.0.0.22
00:03:47:        DR: 1.0.0.22 (Id)   BDR: 1.0.0.21 (Id)
```

```
R1100c#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.22          1   FULL/DR         00:01:36    10.0.8.22       Serial0/0
1.0.0.21          1   FULL/BDR        00:01:57    10.0.8.21       Serial0/0
1.0.0.12          1   2WAY/DROTHER    00:01:52    10.0.8.12       Serial0/0
```

```
R1200c#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.22          1   FULL/DR         00:01:56    10.0.8.22       Serial0/0
1.0.0.21          1   FULL/BDR        00:01:32    10.0.8.21       Serial0/0
1.0.0.11          1   2WAY/DROTHER    00:01:52    10.0.8.11       Serial0/0
```

```
R2100c#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.22          1   FULL/DR         00:01:30    10.0.8.22       Serial0/0
1.0.0.12          1   FULL/DROTHER    00:01:39    10.0.8.12       Serial0/0
1.0.0.11          1   FULL/DROTHER    00:01:57    10.0.8.11       Serial0/0
```

```
R2200c#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.21          1   FULL/BDR        00:01:40    10.0.8.21       Serial0/0
1.0.0.12          1   FULL/DROTHER    00:01:43    10.0.8.12       Serial0/0
1.0.0.11          1   FULL/DROTHER    00:01:31    10.0.8.11       Serial0/0
```

```
R1100c#sh ip ospf int s0/0
```

```
Serial0/0 is up, line protocol is up
  Internet Address 10.0.8.11/24, Area 0
  Process ID 1, Router ID 1.0.0.11, Network Type NON_BROADCAST, Cost: 64
  Transmit Delay is 1 sec, State DROTHER, Priority 1
  Designated Router (ID) 1.0.0.22, Interface address 10.0.8.22
  Backup Designated router (ID) 1.0.0.21, Interface address 10.0.8.21
  Timer intervals configured, Hello 30, Dead 120, Wait 120, Retransmit 5
    Hello due in 00:00:25
  Index 2/2, flood queue length 0
  Next 0x0(0)/0x0(0)
  Last flood scan length is 0, maximum is 1
  Last flood scan time is 0 msec, maximum is 4 msec
  Neighbor Count is 3, Adjacent neighbor count is 2
    Adjacent with neighbor 1.0.0.22  (Designated Router)
    Adjacent with neighbor 1.0.0.21  (Backup Designated Router)
  Suppress hello for 0 neighbor(s)
```

```
R1100c#sh ip ospf database database-summary

            OSPF Router with ID (1.0.0.11) (Process ID 1)

Area 0 database summary
  LSA Type      Count    Delete   Maxage
  Router        4        0        0
  Network       1        0        0
  Summary Net   0        0        0
  Summary ASBR  0        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Subtotal      5        0        0

Process 1 database summary
  LSA Type      Count    Delete   Maxage
  Router        4        0        0
  Network       1        0        0
  Summary Net   0        0        0
  Summary ASBR  0        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Type-5 Ext    0        0        0
  Opaque AS     0        0        0
  Total         5        0        0
```

```
R1100c#sh ip ospf database

            OSPF Router with ID (1.0.0.11) (Process ID 1)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
1.0.0.11        1.0.0.11        72          0x80000004 0x00A2CA 2
1.0.0.12        1.0.0.12        73          0x80000004 0x00B5B3 2
1.0.0.21        1.0.0.21        72          0x80000004 0x005A1B 1
1.0.0.22        1.0.0.22        72          0x80000004 0x00581A 1

                Net Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum
10.0.8.22       1.0.0.22        73          0x80000001 0x006E3E
```

```
R1100c#sh ip ospf database router 1.0.0.11

            OSPF Router with ID (1.0.0.11) (Process ID 1)

                Router Link States (Area 0)
  LS age: 85
```

```
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 1.0.0.11
  Advertising Router: 1.0.0.11
  LS Seq Number: 80000004
  Checksum: 0xA2CA
  Length: 48
  Number of Links: 2

    Link connected to: a Transit Network
     (Link ID) Designated Router address: 10.0.8.22
     (Link Data) Router Interface address: 10.0.8.11
      Number of TOS metrics: 0
       TOS 0 Metrics: 64

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 10.0.11.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
       TOS 0 Metrics: 1
```

```
R1100c#sh ip ospf database network 10.0.8.22

            OSPF Router with ID (1.0.0.11) (Process ID 1)

                Net Link States (Area 0)

  Routing Bit Set on this LSA
  LS age: 111
  Options: (No TOS-capability, DC)
  LS Type: Network Links
  Link State ID: 10.0.8.22 (address of Designated Router)
  Advertising Router: 1.0.0.22
  LS Seq Number: 80000001
  Checksum: 0x6E3E
  Length: 40
  Network Mask: /24
        Attached Router: 1.0.0.22
        Attached Router: 1.0.0.21
        Attached Router: 1.0.0.12
Attached Router: 1.0.0.11
```

d. Captures Wireshark

- Cap_2I_03.pcap

3. Réseau physique Frame-Relay complètement maillé

a. Contexte

Contexte inchangé mais configuration établie de façon à fonctionner en mode broadcast.

b. Configuration de routeurs

R1100c

```
R1100c#sh run
.........
!
interface Loopback0
ip address 1.0.0.11 255.255.255.255
!
```

```
interface FastEthernet0/0
ip address 10.0.11.1 255.255.255.0
ip ospf network non-broadcast
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.8.11 255.255.255.0
encapsulation frame-relay
ip ospf network broadcast
frame-relay map ip 10.0.8.12 110 broadcast
frame-relay map ip 10.0.8.21 111 broadcast
frame-relay map ip 10.0.8.22 112 broadcast
!
Routerospf 1
log-adjacency-changes
network 10.0.0.0 0.0.31.255 area 0
.........
!
end
```

R1200c

```
R1200c#sh run
.........
!
interface Loopback0
ip address 1.0.0.12 255.255.255.255
!
interface FastEthernet0/0
ip address 10.0.12.1 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.8.12 255.255.255.0
encapsulation frame-relay
ip ospf network broadcast
frame-relay map ip 10.0.8.11 212 broadcast
frame-relay map ip 10.0.8.21 214 broadcast
frame-relay map ip 10.0.8.22 213 broadcast
!
router ospf 1
log-adjacency-changes
network 10.0.0.0 0.0.31.255 area 0
.........
end
```

R2100c

```
R2100c#sh run
..........
!
interface Loopback0
ip address 1.0.0.21 255.255.255.255
!
interface FastEthernet0/0
ip address 10.0.21.1 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.8.21 255.255.255.0
encapsulation frame-relay
ip ospf network broadcast
frame-relay map ip 10.0.8.11 221 broadcast
frame-relay map ip 10.0.8.12 220 broadcast
frame-relay map ip 10.0.8.22 222 broadcast
!
```

```
router ospf 1
log-adjacency-changes
network 10.0.0.0 0.0.31.255 area 0
.........
end
```

R2200c

```
R2200c#sh run
.........
!
interface Loopback0
ip address 1.0.0.22 255.255.255.255
!
interface FastEthernet0/0
ip address 10.0.22.1 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.8.22 255.255.255.0
encapsulation frame-relay
ip ospf network broadcast
frame-relay map ip 10.0.8.11 222 broadcast
frame-relay map ip 10.0.8.12 223 broadcast
frame-relay map ip 10.0.8.21 224 broadcast
!
router ospf 1
log-adjacency-changes
network 10.0.0.0 0.0.31.255 area 0
.........
```

c. Captures dans l'interface ILC

Table de voisinage :

```
R1100c#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.22          1   FULL/DR         00:00:30    10.0.8.22       Serial0/0
1.0.0.21          1   FULL/BDR        00:00:37    10.0.8.21       Serial0/0
1.0.0.12          1   2WAY/DROTHER    00:00:33    10.0.8.12       Serial0/0
R1100c#
```

Table de routage (1 sur 4) :

```
R1100c#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
.........

Gateway of last resort is not set

     1.0.0.0/32 is subnetted, 1 subnets
C       1.0.0.11 is directly connected, Loopback0
     10.0.0.0/24 is subnetted, 5 subnets
C       10.0.11.0 is directly connected, FastEthernet0/0
C       10.0.8.0 is directly connected, Serial0/0
O       10.0.12.0 [110/65] via 10.0.8.12, 00:00:23, Serial0/0
O       10.0.22.0 [110/65] via 10.0.8.22, 00:00:23, Serial0/0
O       10.0.21.0 [110/65] via 10.0.8.21, 00:00:23, Serial0/0
R1100c#
```

d. Captures Wireshark

Capture réalisée sur S0/0 de R1100c :

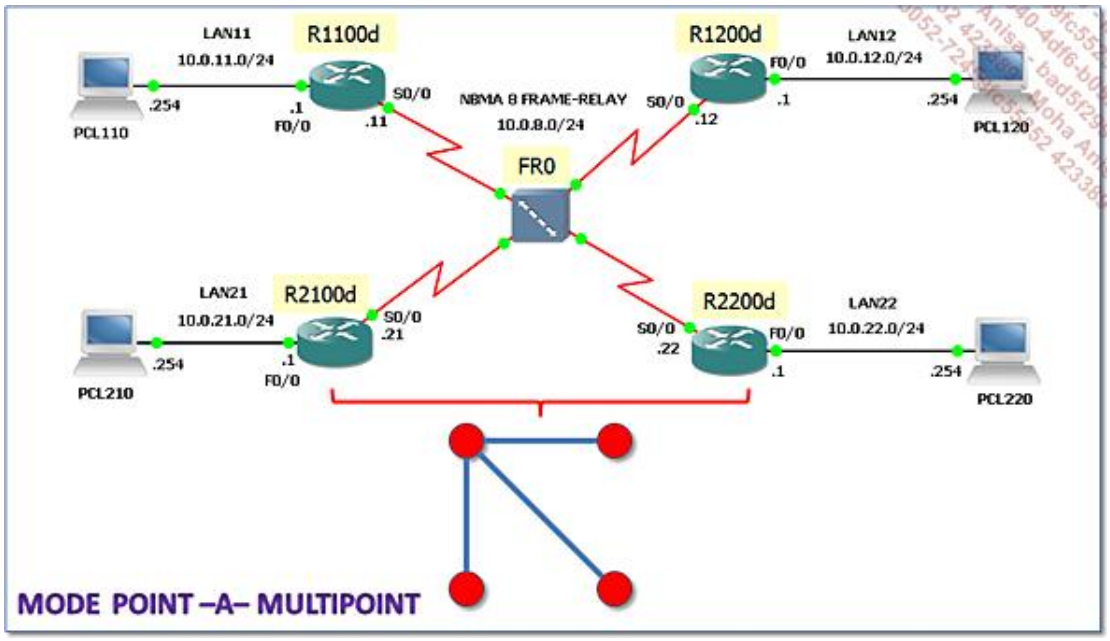
- Cap_2I_04.pcap

4. Réseau physique Frame-Relay partiellement maillé

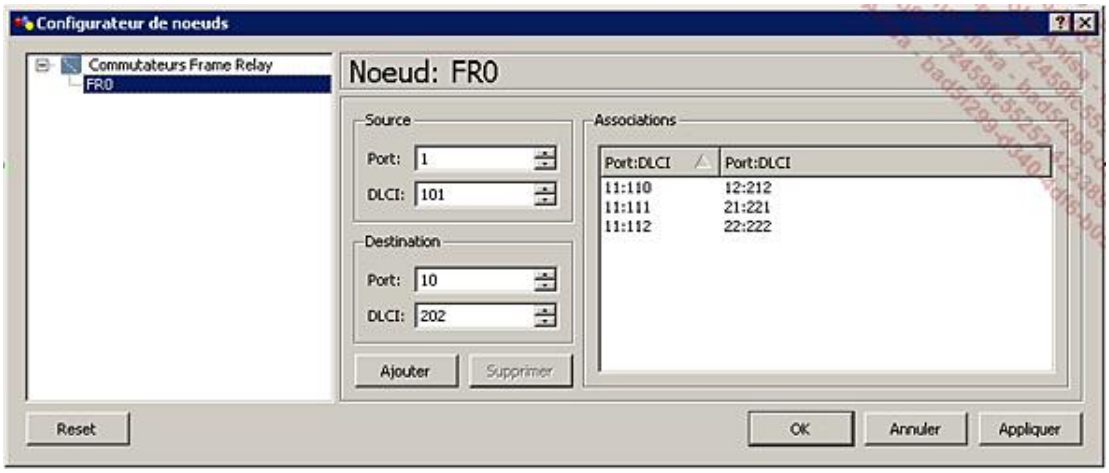
a. Contexte

Le maillage partiel impose de fonctionner dans le mode Point à Multipoint.

Contexte modifié :



Configuration du commutateur Frame Relay sous GNS3 :



b. Configuration de routeurs

R1100d

```
R1100d#sh run
.........
!
interface Loopback0
ip address 1.0.0.11 255.255.255.255
!
```

```
interface FastEthernet0/0
ip address 10.0.11.1 255.255.255.0
ip ospf network broadcast
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.8.11 255.255.255.0
encapsulation frame-relay
ip ospf network point-to-multipoint
!
router ospf 1
log-adjacency-changes
network 10.0.0.0 0.0.31.255 area 0
!
end
```

R1200d

```
R1200d#sh run
.........
!
interface Loopback0
ip address 1.0.0.12 255.255.255.255
!
interface FastEthernet0/0
ip address 10.0.12.1 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.8.12 255.255.255.0
encapsulation frame-relay
ip ospf network point-to-multipoint
!
router ospf 1
log-adjacency-changes
network 10.0.0.0 0.0.31.255 area 0
.........
end
```

R2100d

```
R2100d#sh run
.........
!
interface Loopback0
ip address 1.0.0.21 255.255.255.255
!
interface FastEthernet0/0
ip address 10.0.21.1 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.8.21 255.255.255.0
encapsulation frame-relay
ip ospf network point-to-multipoint
!
router ospf 1
log-adjacency-changes
network 10.0.0.0 0.0.31.255 area 0
.........
end
```

R2200d

```
R2200d#sh run
.........
```

```
!
interface Loopback0
ip address 1.0.0.22 255.255.255.255
!
interface FastEthernet0/0
ip address 10.0.22.1 255.255.255.0
duplex auto
speed auto
!
interface Serial0/0
ip address 10.0.8.22 255.255.255.0
encapsulation frame-relay
ip ospf network point-to-multipoint
!
router ospf 1
log-adjacency-changes
network 10.0.0.0 0.0.31.255 area 0
end
```

c. Captures dans l'interface ILC

Quelques commandes **show** sur le « *hub router* » :

```
R1100d#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.12          1   FULL/  -        00:01:39    10.0.8.12       Serial0/0
1.0.0.22          1   FULL/  -        00:01:39    10.0.8.22       Serial0/0
1.0.0.21          1   FULL/  -        00:01:39    10.0.8.21       Serial0/0
```

```
R1100d#sh ip ospf neighbor 1.0.0.12
 Neighbor 1.0.0.12, interface address 10.0.8.12
    In the area 0 via interface Serial0/0
    Neighbor priority is 1, State is FULL, 6 state changes
    DR is 0.0.0.0 BDR is 0.0.0.0
    Options is 0x42
    Dead timer due in 00:01:46
    Neighbor is up for 00:01:39
    Index 2/2, retransmission queue length 0, number of retransmission 1
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 1, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec
```

```
R1100d#sh ip ospf neighbor 1.0.0.21
 Neighbor 1.0.0.21, interface address 10.0.8.21
    In the area 0 via interface Serial0/0
    Neighbor priority is 1, State is FULL, 6 state changes
    DR is 0.0.0.0 BDR is 0.0.0.0
    Options is 0x42
    Dead timer due in 00:01:41
    Neighbor is up for 00:02:13
    Index 3/3, retransmission queue length 0, number of retransmission 1
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 1, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec
```

```
R1100d#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set
```

```
     1.0.0.0/32 is subnetted, 1 subnets
C       1.0.0.11 is directly connected, Loopback0
     10.0.0.0/8 is variably subnetted, 8 subnets, 2 masks
C       10.0.11.0/24 is directly connected, FastEthernet0/0
C       10.0.8.0/24 is directly connected, Serial0/0
O       10.0.12.0/24 [110/65] via 10.0.8.12, 00:02:12, Serial0/0
O       10.0.8.12/32 [110/64] via 10.0.8.12, 00:02:12, Serial0/0
O       10.0.8.22/32 [110/64] via 10.0.8.22, 00:02:12, Serial0/0
O       10.0.8.21/32 [110/64] via 10.0.8.21, 00:02:12, Serial0/0
O       10.0.22.0/24 [110/65] via 10.0.8.22, 00:02:12, Serial0/0
O       10.0.21.0/24 [110/65] via 10.0.8.21, 00:02:12, Serial0/0
R1100d#
```

Quelques commandes **show** sur un « *Spoke Router* », R1200d :

```
R1200d#sh ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.0.0.11          1   FULL/  -        00:01:50    10.0.8.11       Serial0/0
```

```
R1200d#sh ip ospf neighbor 1.0.0.11
 Neighbor 1.0.0.11, interface address 10.0.8.11
    In the area 0 via interface Serial0/0
    Neighbor priority is 1, State is FULL, 6 state changes
    DR is 0.0.0.0 BDR is 0.0.0.0
    Options is 0x42
    Dead timer due in 00:01:33
    Neighbor is up for 00:09:27
    Index 1/1, retransmission queue length 0, number of retransmission 1
    First 0x0(0)/0x0(0) Next 0x0(0)/0x0(0)
    Last retransmission scan length is 1, maximum is 1
    Last retransmission scan time is 0 msec, maximum is 0 msec
```

```
R1200d#sh ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     1.0.0.0/32 is subnetted, 1 subnets
C       1.0.0.12 is directly connected, Loopback0
     10.0.0.0/8 is variably subnetted, 8 subnets, 2 masks
O       10.0.11.0/24 [110/65] via 10.0.8.11, 00:09:20, Serial0/0
C       10.0.8.0/24 is directly connected, Serial0/0
C       10.0.12.0/24 is directly connected, FastEthernet0/0
O       10.0.8.11/32 [110/64] via 10.0.8.11, 00:09:20, Serial0/0
O       10.0.8.22/32 [110/128] via 10.0.8.11, 00:09:20, Serial0/0
O       10.0.8.21/32 [110/128] via 10.0.8.11, 00:09:20, Serial0/0
O       10.0.22.0/24 [110/129] via 10.0.8.11, 00:09:20, Serial0/0
O       10.0.21.0/24 [110/129] via 10.0.8.11, 00:09:20, Serial0/0
```

```
R1200d#sh ip ospf database

            OSPF Router with ID (1.0.0.12) (Process ID 1)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
1.0.0.11        1.0.0.11        584         0x80000004 0x006CEB 5
1.0.0.12        1.0.0.12        584         0x80000004 0x00C98E 3
1.0.0.21        1.0.0.21        585         0x80000002 0x00AC80 3
1.0.0.22        1.0.0.22        585         0x80000002 0x00E146 3
```

```
R1200d#sh ip ospf database router 1.0.0.11

            OSPF Router with ID (1.0.0.12) (Process ID 1)

                Router Link States (Area 0)

  LS age: 593
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 1.0.0.11
  Advertising Router: 1.0.0.11
  LS Seq Number: 80000004
  Checksum: 0x6CEB
  Length: 84
  Number of Links: 5

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 1.0.0.12
     (Link Data) Router Interface address: 10.0.8.11
      Number of TOS metrics: 0
       TOS 0 Metrics: 64

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 1.0.0.22
     (Link Data) Router Interface address: 10.0.8.11
      Number of TOS metrics: 0
       TOS 0 Metrics: 64

    Link connected to: another Router (point-to-point)
     (Link ID) Neighboring Router ID: 1.0.0.21
     (Link Data) Router Interface address: 10.0.8.11
      Number of TOS metrics: 0
       TOS 0 Metrics: 64

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 10.0.8.11
     (Link Data) Network Mask: 255.255.255.255
      Number of TOS metrics: 0
       TOS 0 Metrics: 0

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 10.0.11.0
     (Link Data) Network Mask: 255.255.255.0
      Number of TOS metrics: 0
TOS 0 Metrics: 1
```

d. Captures Wireshark

Capture sur S0/0 de R1100d, démarrage dans l'ordre des routeurs R1100, R1200, R2100, R2200 :

- Cap_2I_05.pcap

Fin de l'annexe Configuration OSPF des modes NBMA et Point à multipoint.