

Routage dynamique avec OSPF

Stéphane Bortzmeyer
AFNIC

Immeuble International
78181
Saint-Quentin-en-Yvelines
France
bortzmeyer@nic.fr

Gitoyen

Une grande partie de ce cours a pu être faite grâce à l'expérience acquise en concevant et en déployant le réseau de l'opérateur Internet Gitoyen.

\$Id: ospf-partial.db,v 1.2 2008/02/12 11:26:38 bortzmeyer Exp \$

Copyright © 2003-2005 AFNIC

Ce document est fourni pour vous permettre de comprendre et de configurer OSPF. Le ou les auteurs ne sont évidemment pas responsables des dégâts que vous causerez à votre réseau. Ce document est distribué sous les termes de la GNU Free Documentation License (<http://www.gnu.org/licenses/licenses.html#FDL>). Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.

Ce document explique la conception et la configuration d'un réseau TCP/IP utilisant le routage dynamique, en l'occurrence avec le protocole OSPF.

Il se veut pratique : l'accent est mis sur une configuration simple qui devrait couvrir la plupart des cas. Il ne remplace donc pas un cours complet sur OSPF.

Table des matières

| | |
|--|-----------|
| 1. Introduction | 1 |
| 2. Premier réseau et première configuration | 9 |
| 3. Déboguage..... | 15 |
| 4. Avec plusieurs zones | 23 |
| 5. Conclusion | 29 |
| A. Configuration matérielle et logicielle..... | 29 |
| Bibliographie..... | 29 |
| RFC | 29 |

1. Introduction

1.1. Dois-je vraiment faire du routage dynamique ?

D'abord, il faut rappeler que le routage dynamique n'est pas toujours nécessaire, ni même utile. Si votre réseau est simple et de petite taille, un routage statique (des routes indiquées manuellement dans les routeurs) suffit très largement.

Dans ce cas là, si votre routeur est une machine Unix, vous n'avez aucun besoin de `routed` ou de `Quagga`. Ces logiciels ne servent qu'au routage dynamique. Le routage proprement dit (*forwarding*, par opposition à *routing* qui désigne la construction des tables de routage dynamiquement) est effectué par le noyau Unix.

Même si vous avez de nombreux routeurs, si l'architecture du réseau est très simple, il peut être plus avantageux de faire du routage statique en utilisant un outil de génération des configurations comme `Rancid`, pour éviter de tout faire à la main.

D'une manière générale, le routage dynamique est :

- Nécessaire, si vous avez plusieurs chemins entre deux points et si vous voulez de la redondance automatique en cas de défaillance d'un lien.
- Utile si vous avez plus de quatre ou cinq routeurs et qu'ils sont de marque différente, rendant difficile une gestion centralisée de leur configuration.

1.2. Et quel protocole dois-je utiliser ?

Pendant un certain temps, plusieurs protocoles ont coexisté pour le routage à l'intérieur d'une organisation (entreprise, Université, etc¹). RIP et IGRP/EIGRP ont eu leur partisans. Aujourd'hui, seul OSPF reste en lice.

RIP souffrait de plusieurs limitations : convergence trop lente en cas de reconfiguration du réseau, limitation à quinze sauts, pas de routage hiérarchique.

À noter qu'il existe deux versions² de RIP : RIPv1 n'accepte pas certains adressages (par exemple il oblige tous les sous-réseaux à avoir la même longueur de préfixe) mais il est très répandu et, pendant longtemps, il avait des partisans qui mettaient en avant sa disponibilité sur toutes les plate-formes. RIPv2 supprime cet inconvénient mais il n'a pas atteint la même ubiquité.

IGRP/EIGRP est un protocole spécifique à Cisco et ne doit donc pas être utilisé.

Pour un réseau nouveau, je ne pense pas qu'il y aie d'autres choix sérieux. OSPF est désormais mis en oeuvre sur toutes les plateformes.

1.3. Petit rappel IP

Important : Ce texte n'est qu'un rappel, ce cours nécessite une bonne connaissance d'IP.

Le protocole IP (que cela soit IPv4 ou bien IPv6) repose sur la notion d'adresse. Une machine a une ou plusieurs adresses qui s'écrivent, en IPv4, avec quatre chiffres séparés par des points, comme

1. Ces protocoles se nomment les IGP *Interior Routing Protocol*. Il existe une autre famille, les EGP *Exterior Gateway Protocol* dont BGP est quasiment le seul membre. Ces EGP sont réservés aux opérateurs Internet.
2. Trois avec celle pour IPv6, RIPng.

80.67.160.1³ et en IPv6 par plusieurs groupes de chiffres séparés par des deux-points, comme 2001:910::5043:a001⁴.

Dans une adresse, les chiffres binaires les plus à gauche identifient le réseau (d'où le nom de préfixe), les chiffres les plus à droite la machine. Le nombre de chiffres pour chaque rôle dépend de la longueur du préfixe. Celle-ci s'indique par un chiffre suivant une barre oblique. Par exemple, 80.67.160.1/27 indique que l'adresse IP en question identifie une machine dans un réseau où les 27 premiers bits désignent le réseau et les 5 derniers la machine (les adresses IPv4 comptent 32 bits). De la même façon, fec0:1:0:ff00::1/64 est une adresse IPv6 sur un réseau où il y a 64 bits pour le réseau et 64 pour la machine (les adresses IPv6 comptent 128 bits).

Il est important de noter que la longueur du préfixe dépend de l'endroit où on se trouve. Du fait de l'agrégation des préfixes, le réseau 80.67.160.0/27 se trouve par exemple annoncé sur Internet comme une partie de 80.67.160.0/19.

Il existe des outils pour faciliter les calculs sur les adresses. En IPv4, netmask permet, par exemple :

```
% netmask 80.67.168.6/19❶
    80.67.160.0/19
% netmask -s 80.67.168.6/27❷
    80.67.168.0/255.255.255.224
% netmask -r 80.67.168.6/27 ❸
    80.67.168.0-80.67.168.31    (32)
```

- ❶ Quelle est l'adresse du réseau correspondant à cette adresse et cette longueur de préfixe ?
- ❷ IPv4 utilise encore beaucoup une vieille notation : au lieu de la longueur du préfixe, on affiche un masque où tous les bits "réseau" valent 1 et tous les bits "machine" valent 0.
- ❸ Quelles sont les adresses IP dans ce réseau ?

En IPv6, il n'existe malheureusement pas d'outils équivalents mais ipv6calc permet certaines choses.

1.4. Petit rappel routage

Pour transmettre un paquet, une machine IP suit sa table de routage. On peut l'afficher sur quasiment tous les Unix avec **netstat -rn**, qui est la commande la plus portable, ou bien avec **route -n** sur Linux, **route -n show** sur NetBSD, **show ip route** sur IOS, etc. La table de routage est une série d'entrées, et elle fait correspondre à un préfixe, l'adresse IP du routeur où envoyer le paquet. Par exemple, si un routeur a la table de routage suivante :

```
147.94.0.4      194.68.129.102  255.255.255.252 UG    0      0      0 eth1
213.223.128.0  194.68.129.244  255.255.255.248 UG    0      0      0 eth1
192.54.202.64❶  194.68.129.102  255.255.255.240 UG    0      0      0 eth1
194.6.149.64   194.68.129.244  255.255.255.224 UG    0      0      0 eth1
194.6.145.0    194.68.129.244  255.255.255.224 UG    0      0      0 eth1
213.56.168.160 194.68.129.224  255.255.255.224 UG    0      0      0 eth1
195.141.72.128 194.68.129.213  255.255.255.224 UG    50     0      0 eth1
```

- ❶ Les paquets à destination de 192.54.202.66 (**netmask 192.54.202.66/255.255.255.240** donne 192.54.202.64) seront envoyés à 194.68.129.102. traceroute permet de vérifier cela :

```
traceroute to 192.54.202.66 (192.54.202.66), 30 hops max, 40 byte packets
 1  194.68.129.102  1 ms  0 ms  0 ms
 2  193.51.179.158  1 ms  1 ms  1 ms
 ...
```

3. Plus de détails sur l'adressage IPv4 dans RFC 791.
4. Plus de détails sur l'adressage IPv6 dans RFC 2373.

Il y a une légère différence en IPv6 : il existe des adresses spécifiques à un lien (*link local*), leur préfixe est fe80::/10 et ce sont ces adresses qui sont utilisées dans la table de routage.

```
rachel:~ % netstat -r -n -finet6
```

```
...
Destination                                Gateway                                Flags  Refs  Use  Mtu  Int
default                                    fe80::201:96ff:fe96:dc60%epic0  UG          0     0    -    epi
```

Important : Pour une adresse IP de destination, il peut exister plusieurs entrées dans la table de routage. Si les longueurs de préfixe sont différentes, le routeur choisira la plus spécifique (le préfixe le plus long). Si les longueurs de préfixe sont égales, le résultat dépend du routeur. Avec Linux, si le noyau a été compilé avec l'option *IP: equal cost multipath (CONFIG_IP_ROUTE_MULTIPATH)*⁵, les deux chemins pourront être utilisés, répartissant ainsi la charge.

Les tables de routage peuvent être construites manuellement (l'ingénieur système ajoute des commandes dans un fichier de configuration) ou automatiquement par un système de routage dynamique. Dans ce dernier cas, les routeurs échangent de l'information entre eux ("Je suis connecté à un lien qui porte les préfixes 2001:910::/32 et FEC0:1::/32", "J'ai une liaison point-à-point avec 82.3.67.98") et, sur la base de ces informations, construisent chacun sa table de routage.

En anglais, on appelle le routage effectif des paquets *forwarding* et la construction des tables de routage *routing*. Tout routeur IP fait du *forwarding* mais le routage pouvant être statique, tous ne font pas du *routing*. Ces deux fonctions sont typiquement mises en oeuvre par des parties très distinctes du routeur⁶.

1.5. Et si ça ne marche pas ?

Si votre configuration TCP/IP ne fonctionne pas, il est utile de demander de l'aide. D'innombrables listes de diffusion⁷ existent à cette fin. Mais beaucoup de personnes ont du mal à donner sur une liste de diffusion les informations nécessaires pour résoudre le problème.

Pour toute liste, les conseils suivants sont utiles :

- Donnez les vraies informations. Il n'existe aucune raison sérieuse⁸ de dissimuler les noms et les adresses IP utilisées. Au contraire, si vous indiquez les vraies adresses, les lecteurs pourront les essayer plus facilement.
Sauf si vous soumettez un problème purement théorique, auquel cas vous devez utiliser des adresses RFC 1918, donnez les vraies adresses.
- Donnez le maximum de détails sur votre environnement : système d'exploitation utilisé, version, type de réseau (Ethernet ? FDDI ?). N'oubliez pas que vos lecteurs ne sont pas dans votre tête : pour vous, il est évident que votre machine 192.168.7.34 est un routeur Extreme mais les autres ne le savent pas.
- Faites des schémas. Un bon croquis vaut souvent mieux que bien des discours. Comme il n'existe pas de norme largement répandue pour transmettre des dessins⁹, le mieux est d'utiliser l'ASCII

5. qui nécessite l'option *IP: advanced router*

6. Sur Unix, le noyau effectue le *forwarding*, alors que le *routing* est confié à un programme extérieur comme Quagga.

7. Par exemple, en France, la liste IP (<http://www.services.cnrs.fr/www/info/ip>) ou bien en Afrique la liste d'AFNOG (<http://www.afnog.org/meeting3.html#mailinglist>), destinée aux opérateurs mais qui voit en pratique beaucoup de questions de base sur IP.

8. Et surtout pas de raisons de sécurité.

9. Je fonde beaucoup d'espérance sur SVG (<http://www.w3.org/Graphics/SVG/Overview.html>) mais qui est très peu répandu encore.

art. Avec le mode picture ([http://www.cs.cmu.edu/cgi-bin/info2www?\(emacs\)Picture](http://www.cs.cmu.edu/cgi-bin/info2www?(emacs)Picture)) de l'éditeur Emacs, c'est assez facile. En voici un exemple, avec indication des noms des routeurs (ce qui facilite beaucoup les discussions ultérieures) et des interfaces :

```

      TO THE INTERNET
+-----+
|SuperNet | Management network 192.168.173.0/28
|(upstream provider) | +-----+
+-----+
      |
      | Serial0
+-----+
| nelson  | | steve  |
|(cisco router) | |(linux router) |
+-----+
      / | | Ethernet0 (.65)
      / | +-----+
      / | | Backbone 192.168.173.64/26
PtP links
to customers
192.168.173.32/27

```

- Soyez factuel : donnez les commandes *exactes* que vous avez utilisées, les résultats *exacts* de ces commandes. Faites du copier/coller, pas de la traduction ou du résumé.

1.6. OSPF en quelques mots

Nous allons d'abord faire de l'OSPF avec une seule zone (*area*). Ce cas correspond à la grande majorité des utilisations, seules les grosses et complexes organisations ont vraiment besoin de routage en plusieurs zones (Section 4).

Vous avez donc un AS (*Autonomous System*), c'est à dire une entité administrative unique (notez bien que la définition d'un AS n'est pas technique. En pratique, un AS est un système où une seule personne ou bien une seule équipe peut décider et que cette décision soit ensuite appliquée dans tout l'AS). Il existe deux sortes de routeurs dans votre unique zone (qui porte le numéro 0, on l'appelle aussi *backbone area*). Les ASBR (*Autonomous System Border Router*) qui sont connectés à votre AS et à l'extérieur (typiquement à un ou plusieurs fournisseurs d'accès). Et les autres routeurs qui ne sont connectés qu'à des liens internes.

Chaque routeur a une ou plusieurs interfaces sur lesquelles il fait tourner OSPF. La configuration consiste donc à déclarer sur chaque routeur quelles interfaces sont activées¹⁰. Sur le ou les ASBR, il faut en outre indiquer quelle(s) routes externes sont redistribuées dans l'AS. Dans le cas le plus courant, on ne redistribue que la route par défaut.

Il existe plusieurs types de réseaux : en gros, vous risquez surtout de rencontrer des réseaux à diffusion (*broadcast networks*) comme Ethernet et des réseaux point à point (*point-to-point networks*) comme ceux réalisés avec le protocole PPP, par exemple sur une ligne téléphonique avec modem ou bien sur une ligne spécialisée numérique.

Sur les réseaux à diffusion, les routeurs OSPF du même segment sont dits voisins (*neighbor*). Sur chaque segment, les routeurs élisent un routeur (DR, *Designated Router*) qui va se charger de collecter toutes les informations et de les redistribuer.

Important : Le routeur élu n'a un rôle privilégié que pour la diffusion des informations OSPF. Pour le routage proprement dit (*forwarding*), il ne fait rien de plus que les autres.

10. Sur certains routeurs, cela suffit. Sur d'autres, il faut indiquer les réseaux IP actifs en OSPF.

Un routeur de secours (BDR, *Backup Designated Router*) est également élu et se tient prêt à le remplacer. Les autres routeurs sont dits jointifs (*adjacent*) de ces deux routeurs élus.

Bien sûr, OSPF est plus riche que cela, et permet beaucoup de réglages. Néanmoins, si vous avez déjà consulté un guide sur OSPF et que vous avez été effrayés par la quantité d'options, ne vous inquiétez pas : la plupart des configurations d'OSPF sont beaucoup plus simples.

1.7. Quel matériel et logiciel choisir ?

Comme indiqué, OSPF tourne aujourd'hui sur tous les routeurs : machines Unix, routeurs dédiés dans une petite boîte (*appliances*) et la plupart des commutateurs (*switches*) de milieu et de haut de gamme, qui ont également des fonctions de routage (les commerciaux parlent parfois de *switches de niveau 3*).

L'inclinaison personnelle de l'auteur le porte vers les logiciels libres (<http://www.april.org/>) donc la majorité des exemples concerneront le couple Unix+Zebra (<http://www.zebra.org/>) avec lequel l'auteur a le plus d'expérience. (Zebra a désormais un successeur, Quagga (<http://www.quagga.net/>), Zebra ne semblant plus maintenu.) Ce couple permet de faire du routage dynamique avec un simple PC, et un Unix libre comme Debian (<http://www.debian.org/>) ou bien FreeBSD (<http://www.freebsd.org/>).

Le système le plus utilisé pour les routeurs OSPF est sans doute IOS (Internetwork Operating System) (<http://www.cisco.com/en/US/products/sw/iosswrel/index.html>) de Cisco. Les commandes sont très proches de celles de Zebra. Beaucoup d'autres routeurs (HP Procurve, Foundry) ont un jeu de commandes très proche de celui d'IOS.

1.8. Petit rappel sur l'utilisation des routeurs IP

Important : Ce texte n'est qu'un rappel sommaire, ce cours nécessite une bonne connaissance des environnements utilisés. Par exemple, si votre routeur est une machine Unix, il faut connaître le shell, le système des journaux, etc. Par exemple, lorsque j'écris "envoyez le signal HUP au démon", je considère acquis que vous savez envoyer un signal à un démon¹¹.

1.8.1. Quagga

Quagga (<http://www.quagga.net/>) est issu du programme Zebra, qui ne semble plus maintenu mais que vous trouverez encore fréquemment en production.

Quagga est composé de plusieurs démons qui acceptent chacun des connexions TCP. Vous pouvez donc utiliser telnet pour vous connecter à chaque démon et effectuer sa configuration¹².

Si cette connexion aux démons est très pratique pour déboguer ou surveiller un routeur, je ne la trouve pas efficace pour écrire ou modifier la configuration du routeur. Je recommande plutôt de modifier les fichiers de configuration avec un éditeur, ce qui permet d'utiliser un bon outil d'édition, de gérer les configurations avec CVS, de préserver les commentaires, etc.

L'emplacement des fichiers de configuration de Quagga dépend de votre Unix et de la manière dont Quagga a été compilé¹³. J'utiliserai les emplacements du système Debian. Tous les fichiers sont alors dans `/etc/zebra`. On trouve notamment :

11. **man kill**

12. Il existe aussi un système, le `vttysh` pour se connecter de manière unifiée à tous les démons mais il n'est pas décrit ici.

13. Rappelez-vous que Quagga se nommait Zebra précédemment et beaucoup de fichiers n'ont pas encore changé de nom.

- `zebra.conf`, la configuration générale¹⁴,
- `ospfd.conf`, la configuration du protocole OSPF,
- `ospf6d.conf`, la configuration du protocole OSPFv6,
- `bgpd.conf`, la configuration du protocole BGP.

Normalement, sur Unix, une fois qu'on a modifié un fichier de configuration, on envoie le signal HUP au démon¹⁵. Dans certaines versions de Quagga, avec certains protocoles (notamment BGP), ce signal a des effets négatifs (comme de couper toutes les sessions BGP) ou nuls. Il est donc recommandé d'utiliser la démarche suivante pour changer la configuration.

1. Tapez les commandes dans la fenêtre de la console du démon.
2. Si elles sont acceptées et produisent bien le résultat attendu, copiez les commandes dans le fichier de configuration, avec les commentaires appropriés.

Une variante, surtout si on connaît bien les commandes, est de les mettre dans le fichier d'abord, puis de copier/coller les nouvelles commandes dans la console.

Venons en maintenant à la console d'administration du démon. On l'atteint en faisant un **telnet** *routeur démon*¹⁶. Le nom du démon (ou, plus exactement, le port sur lequel il écoute) est `zebra`, `ospfd`, `ospf6d` ou bien `bgp`¹⁷.

Une fois connecté à la console du démon, vous avez une invite (configurable) et vous pouvez taper des commandes :

```
monrouteur> show ip ospf route
```

Il n'est pas nécessaire de connaître toutes ces commandes par coeur. La touche `?` vous fera apparaître une aide contextuelle¹⁸. Par exemple, si vous avez déjà tapé **show ip ospf** et que vous tapez un `?`, vous avez la liste :

```
border-routers  for this area
database        Database summary
interface       Interface information
neighbor        Neighbor list
route           OSPF routing table
<cr>
```

des commandes valides à ce stade (`<cr>` signifie *Carriage Return* et désigne la touche **Entrée**).

Vous pouvez également compléter une commande en cours de saisie avec la touche **Tabulation**.

Ce mécanisme de console en ligne de commande se prête bien à l'automatisation. Par exemple, le programme Python (<http://www.python.org/>) suivant se connecte à un routeur pour afficher la liste des routes¹⁹ :

14. Ce fichier est vide la plupart du temps. Il contient des définitions d'interfaces et de routes statiques mais, sur Unix, on les fait typiquement en dehors de Quagga. Le noyau Unix informera ensuite Quagga. On ne remplit ce fichier que pour contourner des bogues du noyau (par exemple avec les adresses *multicast* sur NetBSD).

15. Debian facilite les choses en créant pour chaque démon un script `/etc/init.d/démon` qui accepte toujours les mêmes options, comme **reload** pour recharger une configuration, ce qui permet de gérer de manière uniforme tous les démons, même ceux qui ne suivent pas parfaitement la convention. D'autres systèmes d'exploitation ont un mécanisme analogue, par exemple FreeBSD dans ses toutes dernières versions (5).

16. `telnet` ne chiffrant pas les communications, il est très fortement recommandé de taper cette commande depuis le routeur lui-même, pour limiter les risques d'écoute.

17. Si aucun de ces noms ne marche, par exemple si vous obtenez `tcp/zebra: unknown service`, c'est que ces noms n'ont pas été mis dans `/etc/services`. Là encore, il est recommandé d'utiliser un système de paquets qui vous épargne ce genre d'oublis.

18. C'est-à-dire que les commandes ou options affichées sont celles qui sont réellement disponibles à ce stade. C'est d'autant plus utile que les commandes ne sont pas les mêmes pour chaque démon.

19. SNMP permet souvent la même chose.

```

from telnetlib import Telnet
tn = Telnet(router, 2601) # 2601 is the port of the Quagga console

def write_after_prompt (prompt, text):
    (index, match, read) = tn.expect([prompt], 4)
    if not match:
        raise "Text \"" + prompt + "\"" not found"
    # Eat it
    tn.read_until(prompt, 0)
    tn.write(text + '\r\n')

write_after_prompt ("Password:", password)
write_after_prompt (prompt, 'terminal length 0')
write_after_prompt (prompt, 'show ip route')

```

Certaines commandes nécessitent plus de privilège, puisqu'elles modifient l'état du routeur. La commande **enable** permet de passer dans cet état privilégié. Par exemple, pour changer la configuration du routeur ou bien pour réinitialiser une session BGP, vous devrez être en mode **enable**.

1.8.2. IOS

Important : IOS est un logiciel commercial et il est donc recommandé de s'adresser au vendeur pour toute question.

Je suis bien conscient que le vendeur n'est souvent pas très efficace pour le support mais les partisans du logiciel commercial expliquent toujours que l'avantage de ce dernier est la qualité du support : c'est donc l'occasion de tester.

Le langage de commandes d'IOS a été largement imité et ne surprendra donc pas les utilisateurs de Quagga. Comme pour Unix, je recommande d'éditer les fichiers de configuration sur une machine qui puisse les gérer avec un outil comme CVS. Si on gère plusieurs routeurs, il est recommandé d'utiliser un outil comme COSI (<http://cosi-nms.sourceforge.net/>) ou Rancid (<http://www.shrubbery.net/rancid/>). Une fois le fichier édité, on peut le charger sur le Cisco avec **configure network**²⁰.

1.8.3. SNMP

SNMP (*Simple Network Management Protocol*) est un protocole de gestion de réseau. C'est une norme (RFC 1157 et RFC 2578) donc il fonctionnera quel que soit le type de vos routeurs. Vous trouverez énormément d'informations sur SNMP sur le Simple Web (<http://www.simpleweb.org/>).

SNMP permet à un gérant (*manager*) d'interroger un agent (qui se trouve, dans notre cas, sur un routeur ou un commutateur) et d'obtenir des informations. Par exemple, ici, on utilise le gérant Net-SNMP sur Unix pour demander à un routeur combien il a d'interfaces²¹, puis combien d'octets sont passés sur la première interface depuis le démarrage du routeur :

```

% snmpget 192.134.7.246 password interfaces.ifNumber.0
interfaces.ifNumber.0 = 5

% snmpget 192.134.7.246 password interfaces.ifTable.ifEntry.ifInOctets.1
interfaces.ifTable.ifEntry.ifInOctets.1 = Counter32: 34254759

```

20. Il faudra avoir configuré un serveur TFTP avant.

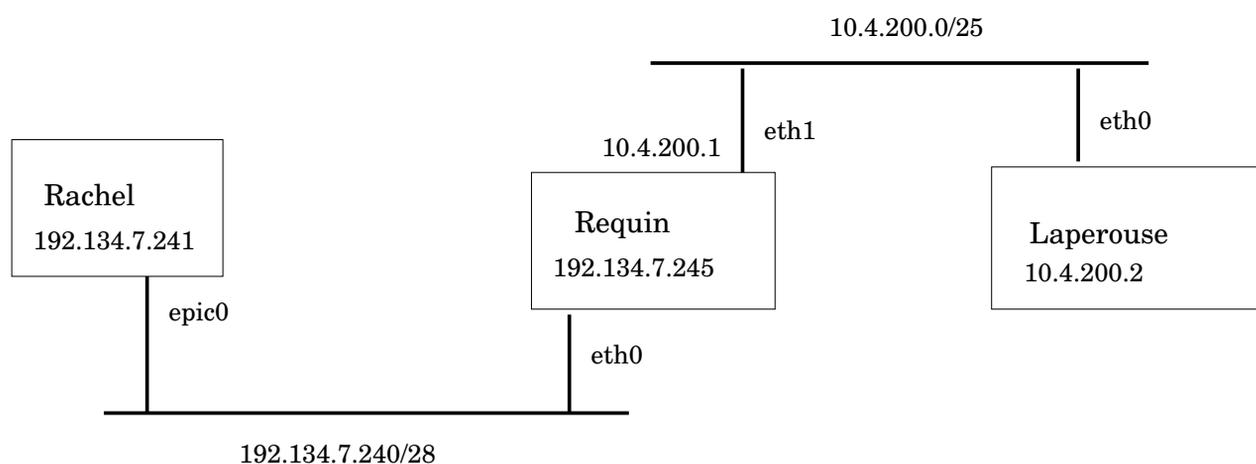
21. Cette information étant un scalaire, il faut terminer l'identificateur de l'objet SNMP par un 0. La seconde information est un vecteur, on termine l'identificateur par l'index.

2. Premier réseau et première configuration

2.1. Une zone, aucune route externe

Commençons par un réseau très simple, tellement simple qu'il n'aurait aucun besoin d'OSPF dans la réalité. Tous les réseaux sont des Ethernet, donc à diffusion, et il n'y a que trois routeurs, dont un seul a deux interfaces. Une seule zone est présente, la zéro.

Figure 1. Réseau simple



Dans chaque routeur, son *router ID*. Les liens sont tous des Ethernet.

La configuration d'un routeur à une seule interface est triviale. Ici, le routeur rachel, qui utilise Quagga :

```
hostname rachel
router ospf
ospf router-id 192.134.7.241❶
network 192.134.7.0/24 area 0❷
```

- ❶ Il n'est pas nécessaire de spécifier un *router ID*. Le routeur en choisit un automatiquement parmi les adresses IP de la machine. Mais le choisir explicitement comme étant l'adresse "principale" de la machine aidera au débogage, par exemple lors d'un **show ip ospf neighbor**.
- ❷ On note que le préfixe est plus général (plus court) que la vraie adresse. Cela n'est pas un problème, Quagga (ou IOS) n'annoncent que les routes réellement existantes. Voir aussi Section 2.5.

Et c'est tout ! rachel est désormais routeur OSPF et annonce son unique réseau. Les voisins sont trouvés automatiquement.

Avertissement

Certains systèmes (IOS) tendent à faire de l'OSPF sur toutes les interfaces. Il peut donc être prudent d'ajouter **passive-interface Serial 0** (dans la section **router ospf**) pour chaque interface où vous ne voulez *pas* faire d'OSPF.

Le routeur laperouse a une configuration quasi-identique. requin est plus intéressant car il est connecté à deux réseaux :

```
hostname requin
router ospf
  ospf router-id 192.134.7.245
  network 10.4.200.0/24 area 0
  network 192.134.7.0/24 area 0
```

requin et rachel sont voisins, ainsi que requin et laperouse. Les routes sont échangées. On peut le voir avec la console OSPF de Quagga:

```
rachel> show ip ospf neighbor
```

| Neighbor ID | Pri | State | Dead Time | Address | Interface | RXmtL | RqstL | D |
|---------------|-----|----------|-----------|---------------|---------------------|-------|-------|---|
| 192.134.7.245 | 1 | Full/DR① | 00:00:34 | 192.134.7.245 | epic0:192.134.7.241 | 0 | 0 | 0 |

① 192.134.7.245, c'est-à-dire requin est routeur élu (DR) et il est adjacent (*full*)

Les routes reçues sur rachel sont :

```
rachel> show ip ospf route
===== OSPF network routing table =====
N   10.4.200.0/25          [20] area: 0.0.0.0
    via 192.134.7.245, epic0
N   192.134.7.240/28      [10] area: 0.0.0.0
    directly attached to epic0
```

Sur requin, on voit deux voisins, un sur chaque interface. requin est routeur élu sur le réseau de son interface eth0 mais seulement routeur de secours sur le réseau de eth1 (laperouse est le routeur élu).

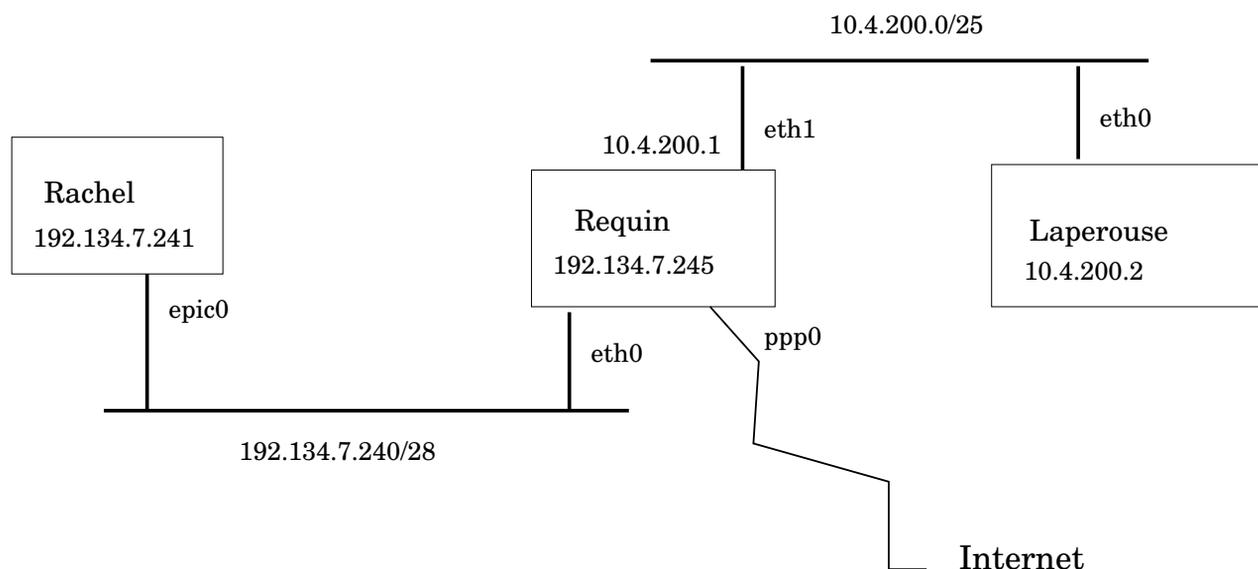
```
requin> show ip ospf neighbor
```

| Neighbor ID | Pri | State | Dead Time | Address | Interface | RXmtL | RqstL | D |
|---------------|-----|-------------|-----------|---------------|--------------------|-------|-------|---|
| 192.134.7.241 | 1 | Full/Backup | 00:00:36 | 192.134.7.241 | eth0:192.134.7.245 | 0 | 0 | |
| 10.4.100.2 | 1 | Full/DR | 00:00:40 | 10.4.200.2 | eth1:10.4.200.1 | 0 | 0 | 0 |

2.2. Une zone, et une route externe

Maintenant, ajoutons un lien vers l'extérieur en supposant que requin est connecté à l'Internet. Nous ne faisons pas d'OSPF avec notre fournisseur d'accès, la route par défaut vers celui-ci va être marquée comme externe à l'AS.

Figure 2. Réseau avec une route externe



Dans chaque routeur, son *router ID*. ppp0 est une interface PPP vers le fournisseur d'accès.

La configuration de requin devient :

```
router ospf
ospf router-id 192.134.7.245
redistribute static
network 10.4.200.0/24 area 0
network 192.134.7.0/24 area 0
default-information originate
```

Deux nouvelles commandes ont été utilisées. **default-information originate** indique à OSPF d'accepter de diffuser une route par défaut (ce qu'OSPF ne fait pas normalement) et **redistribute static** indique de redistribuer dans OSPF toutes les routes statiques (ici, il n'y en a qu'une).

Sur le routeur laperouse (qui n'a pas changé de configuration), on peut voir désormais la route par défaut :

```
laperouse> show ip ospf route
===== OSPF network routing table =====
N   10.4.200.0/25          [1] area: (0.0.0.0)
    directly attached to eth0
N   192.134.7.240/28      [11] area: (0.0.0.0)
    via 10.4.200.1, eth0

===== OSPF router routing table =====
R   192.134.7.245         [1] area: (0.0.0.0), ASBR
    via 10.4.200.1, eth0

===== OSPF external routing table =====
N E2 0.0.0.0/0           [11/10] tag: 0
    via 10.4.200.1, eth0
```

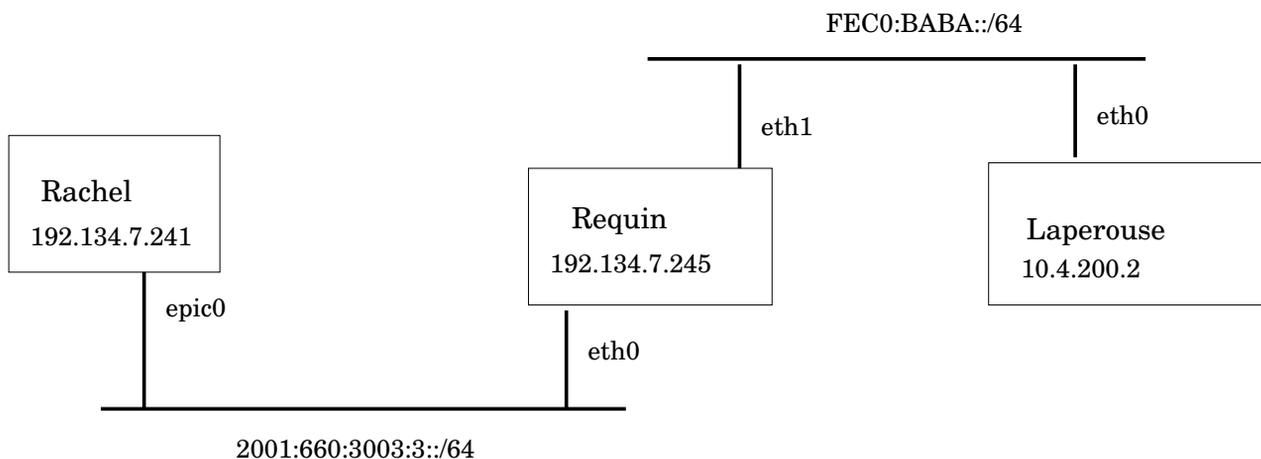
requin (192.134.7.245) y est marqué comme ASBR (*Autonomous System Boundary Router*) car il est situé aux limites de l'AS puisqu'il injecte une route externe, qui apparaît comme de type E2 (externes 2, il existe aussi des externes 1²²).

Entre crochets, vous voyez le coût de chaque route. OSPF permet d'affecter un coût à chaque interface, pour décourager l'utilisation de liens lents, qui ne sont bons qu'à servir de secours. Notre routeur a mis un coût de 10 par défaut et les liens directement connectés ont un coût de 1. C'est pourquoi 192.134.7.240/28 a un coût cumulé de 10+1 = 11.

2.3. Une zone et IPv6

En dotant les machines présentées en Section 2.1 d'adresses IPv6, nous pouvons faire tourner la version 3 du protocole OSPF.

Figure 3. Réseau simple en IPv6



Dans chaque routeur, son *router ID* qui a la forme d'une adresse IPv4. Les liens sont tous des Ethernet.

Sur Quagga, la configuration est plus simple : on peut utiliser les interfaces et pas seulement les réseaux :

```

router ospf6
  router-id 192.134.7.245❶
  interface eth0 area 0.0.0.0
  interface eth1 area 0.0.0.0
  
```

❶ Le *router ID* est sur 32 bits, et affiché comme une adresse IPv4.

On a des adjacences en IPv6 :

```

requin# show ipv6 ospf6 neighbor
RouterID      State/Duration   DR              BDR              I/F[State]
192.134.7.241❶ Full/00:01:42   192.134.7.245  192.134.7.241   eth0[DR]
10.4.200.2    Full/00:00:01   0.0.0.0         0.0.0.0         eth1[DR]
  
```

22. La différence entre les deux est subtile : les externes 2 ont toujours un coût supérieur à n'importe quelle route interne et les externes 1 ont un coût qui est comparable aux routes internes. Les externes de type 1 supposent donc qu'on aie assez d'informations sur les routes externes, ce qui est rare.

C'est l'administrateur système qui configure ce type, avec **redistribute something metric-type 2**.

- ❶ Il s'agit bien d'un *Router ID* et pas d'une adresse IP. Les *Router ID* ont la forme d'une adresse IPv4 (ils sont stockés sur 32 bits) mais ne sont pas des adresses.

De la même façon, les routeurs élus sont identifiés par leur *Router ID*.

et des routes, vers des adresses *link-local* :

```
rachel# show ipv6 ospf6 route
```

| Destination | Gateway | I/F |
|----------------------------|--------------------------|----------------|
| *N Ia 2001:660:3003:3::/64 | :: | epic0 00:02:01 |
| *N Ia fec0:baba::/64 | fe80::206:5bff:fef1:529a | epic0 00:02:01 |
| ... | | |

et ces routes se retrouvent dans la table de routage, avec une destination qui est une adresse *link local*. traceroute permet de voir leur utilisation :

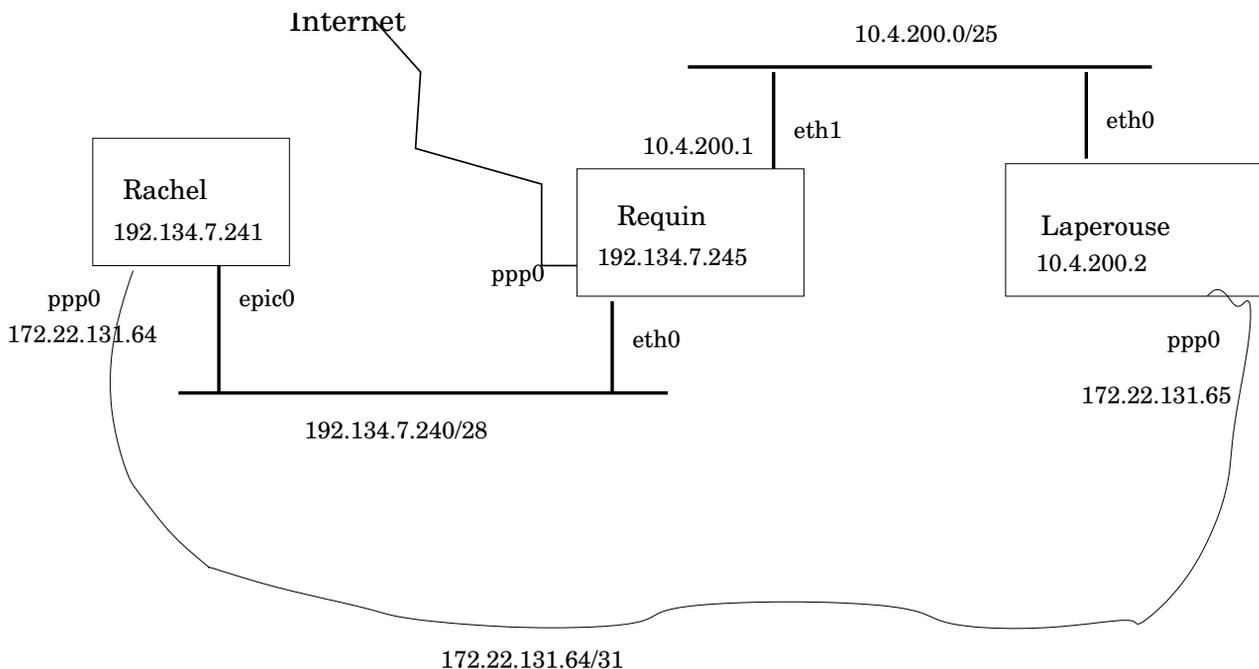
```
% traceroute6 -n fec0:baba::1
```

```
traceroute6 to fec0:baba::1 (fec0:baba::1) from 2001:660:3003:3:2e0:29ff:fe21:130, 30 hops max, 12
 1 2001:660:3003:3:206:5bff:fef1:529a 0.372 ms 0.381 ms 0.345 ms
 2 fec0:baba::1 1.517 ms 0.813 ms 0.779 ms
```

2.4. Une zone, et un lien série

Maintenant, ajoutons un lien série (point à point) entre rachel et laperouse en utilisant le protocole PPP. Bien que ce lien soit très lent (115 kb/s au lieu des 100 Mbs d'Ethernet), il a l'avantage de fournir un secours si requin tombe en panne. Cette fois, enfin, OSPF va réellement servir à quelque chose.

Figure 4. Réseau avec une ligne spécialisée



Dans chaque routeur, son *router ID*. La "ligne spécialisée" est un câble zéro-modem exploité en PPP.

La configuration de requin ne change pas. Celle de Rachel devient :

```
interface ppp0
! Ligne lente, son usage est découragé
ip ospf cost 10000
router ospf
ospf router-id 192.134.7.241
network 172.22.131.65/32 area 0
network 192.134.7.0/24 area 0
```

Vous avez noté ? Zebra exige que vous mettiez l'adresse de la machine *distante* dans une directive **network** avec un masque de /32. IOS fait cela beaucoup mieux. Un **network 172.22.131.0/24 area 0**²³ aurait suffi²⁴.

Le voisin apparaît sur le lien PPP. On notera que c'est son *router ID* qui est affiché, pas son adresse IP sur le lien PPP :

```
laperouse> show ip ospf neighbor
```

| Neighbor ID | Pri | State | Dead Time | Address | Interface | RXmtL | RqstL | D |
|---------------|-----|---------------|-----------|---------------|--------------------|-------|-------|---|
| 192.134.7.245 | 1 | Full/Backup | 00:00:37 | 10.4.200.1 | eth0:10.4.200.2 | 0 | 0 | 0 |
| 192.134.7.241 | 1 | Full/DROther❶ | 00:00:32 | 172.22.131.64 | ppp0:172.22.131.65 | 0 | 0 | 0 |

❶ Sur un réseau point-à-point, les voisins sont toujours adjacents (*full*). Le statut (ici DROther) n'a donc pas d'importance : IOS ne l'affiche d'ailleurs pas.

La table de routage de laperouse, si on débranche requin, montre un passage par le lien PPP :

```
laperouse:~ % route -n
Kernel IP routing table
Destination      Gateway          Genmask          Flags Metric Ref    Use Iface
...
192.134.7.240    172.22.131.64   255.255.255.240 UG    10010 0      0    ppp0
10.4.200.0       0.0.0.0         255.255.255.128 U      0      0      0    eth0
0.0.0.0          172.22.131.64   0.0.0.0         UG    10010 0      0    ppp0
```

Si on rebranche requin, la route vers 192.134.7.240 passe à nouveau par l'Ethernet, dont le coût OSPF est plus faible :

```
laperouse:~ % route -n
Kernel IP routing table
Destination      Gateway          Genmask          Flags Metric Ref    Use Iface
...
192.134.7.240    10.4.200.1      255.255.255.240 UG    110    0      0    eth0
10.4.200.0       0.0.0.0         255.255.255.128 U      0      0      0    eth0
0.0.0.0          10.4.200.1      0.0.0.0         UG    110    0      0    eth0
```

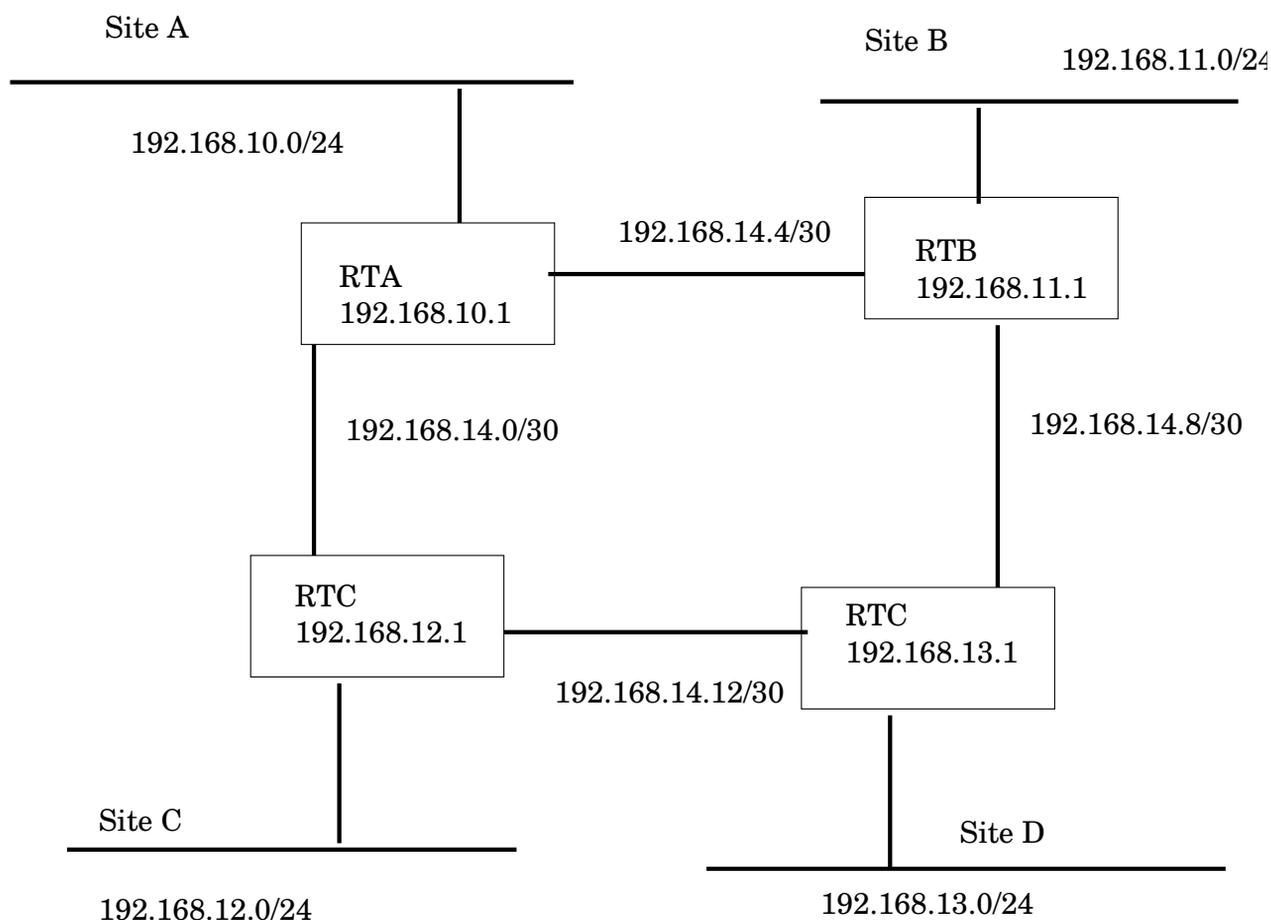
23. Quagga a été corrigée en ce sens et se configure comme IOS.

24. Les érudits noteront que ce comportement bizarre de Zebra est la transcription (trop) directe de la section 12.4.1 "Router-LSAs" du RFC 2328 qui spécifie qu'un routeur OSPF annonce, en point-à-point, l'adresse de son voisin. Le livre de Moy contient une discussion de ce problème (*Chapter 8, "Why is the representation of point-to-point links in OSPF so strange ?"*).

2.5. Une zone, et un anneau

Tous les réseaux n'ont pas forcément ces complications. Dans l'exemple suivant, nous connectons les quatre sites d'un campus (A, B, C et D) en anneau. Cela permettra à la connectivité de survivre à la coupure de n'importe lequel des quatre liens.

Figure 5. Réseau en anneau



Dans chaque routeur, son *router ID*. Les liens entre sites sont des Ethernet.

Comme nous prenons toutes les adresses dans le même préfixe, (192.168.0.0/16) et qu'il n'y a qu'une seule zone, les routeurs ont quasiment tous la même configuration.

```

hostname rta
router ospf
  network 192.168.0.0/16 area 0
  
```

3. Déboguage

Si les choses ne marchent pas comme attendu, il faut procéder avec méthode. Si vous envisagez de demander sur une liste de diffusion ²⁵, il va falloir pouvoir donner toute l'information issue de ce

25. Une très bonne idée : je suis toujours stupéfait que tant de responsables réseau passent des semaines à chercher seuls alors que la compétence de dizaines d'autres ingénieurs expérimentés est à leur disposition gratuitement. Demander de l'aide à des égaux (pas à un enseignant ou à un consultant payé pour cela) sur une liste publique ou même semi-publique est une compétence nécessaire et qui ne s'apprend pas dans les écoles, malheureusement.

processus de débogage.

3.1. Tester la connectivité de base

D'abord, il va falloir tester la connectivité de base de vos routeurs. Si deux routeurs du même réseau ne peuvent pas se pinguer, OSPF ne marchera certainement pas²⁶

Si ping échoue (naturellement, vous testez ping avec une adresse IP comme argument, pas un nom, pour ne pas dépendre du DNS), entamez la procédure de débogage classique en cas de routage statique.

Une fois que ping entre routeurs voisins fonctionne, vous pouvez regarder la table de routage. `traceroute` vous montrera le chemin pris par les paquets²⁷. Les commandes `route -n` sur Linux, `route -n show` sur NetBSD ou encore `show ip route` sur IOS vous montreront la table de routage (FIB, *Forwarding Information Base*) utilisée par le routeur.

3.2. Tester les problèmes spécifiquement OSPF

La première chose à tester est la formation des adjacences. `show ip ospf neighbor` sur Quagga et IOS vous montrera les voisins et leur état. Ils doivent tous être dans l'état FULL (adjacent) ou bien 2WAY (simples voisins, vous verrez cet état si vous avez au moins trois routeurs sur un réseau à diffusion). Voici un exemple :

```
rachel> show ip ospf neighbor
```

| Neighbor ID | Pri | State | Dead Time | Address | Interface | RXmtL | RqstL | D |
|---------------|-----|---------------|-----------|---------------|---------------------|-------|-------|---|
| 192.134.7.245 | 1 | Full/Backup | 00:00:34 | 192.134.7.245 | epic0:192.134.7.241 | 0 | 0 | |
| 10.200.3.1 | 1 | 2-Way/DROther | 00:00:33 | 192.134.7.246 | epic0:192.134.7.241 | 0 | 0 | |
| 192.134.7.248 | 1 | Full/DR | 00:00:34 | 192.134.7.248 | epic0:192.134.7.241 | 0 | 0 | |

Vous noterez que Neighbor montre le *router ID* et pas l'adresse IP.

Voici maintenant un cas où un routeur ne peut devenir adjacent :

```
ludwigV> show ip ospf neighbor
```

| Neighbor ID | Pri | State | Dead Time | Address | Interface | RXmtL | RqstL | D |
|-------------|-----|-------|-----------|---------|-----------|-------|-------|---|
|-------------|-----|-------|-----------|---------|-----------|-------|-------|---|

Rien n'apparaît alors qu'une autre machine est sur le réseau et qu'elle est pinguable. L'examen du journal de Quagga²⁸ nous donne la cause : OSPF: Packet 172.19.1.6 [Hello:RECV]:

HelloInterval mismatch. Un réglage (l'intervalle entre deux messages de bienvenue - Hello - sur le câble) est incohérent entre les deux machines.

D'une manière générale, il faut s'assurer que les réglages sont les mêmes sur tous les routeurs du réseau. C'est d'autant moins facile que tous les modèles n'ont pas forcément les mêmes valeurs par défaut²⁹. Veillez donc à :

26. Il y a une exception : en présence de filtres (comme les ACL d'IOS ou bien le Netfilter de Linux), certains protocoles marchent et d'autres pas. Compte tenu de la complexité supplémentaire que cela entraîne, je recommande de faire vos premiers essais OSPF sur une machine sans filtres.

27. Rappelez-vous toujours que `traceroute` ne montre que le chemin aller. Si vous avez plusieurs routes possibles, rien ne garantit que le chemin retour soit identique.

28. L'endroit exact où se trouve ce journal dépend de votre configuration. C'est souvent `/var/log/zebra/ospfd.log`. Je rappelle que `tail -f` est la façon la plus pratique de lire un fichier journal.

29. Vous pouvez voir les valeurs actuelles avec `show ip ospf interface nom-interface` sur Quagga et IOS.

- MTU (*Maximum Transfer Unit*, affichée par **ifconfig** sur Unix).³⁰
- Intervalles pour Hello et Dead.

Outre les mécanismes de débogage des routeurs (journal de Quagga, commandes **debug ?** d'IOS), les traditionnels outils de débogage réseau comme tcpdump et ethereal sont souvent très précieux. tcpdump est disponible partout et permet de voir facilement si des paquets OSPF circulent :

```
% sudo tcpdump -i eth1 -n proto ospf
tcpdump: listening on eth1
22:03:52.898542 172.19.1.6 > 224.0.0.5: OSPFv2-hello 48: backbone dr 172.19.1.6 bdr 172.19.1.8 [ttl 1]
22:03:57.218574 172.19.1.1 > 224.0.0.5: OSPFv2-hello 44: rtrid 172.19.1.6 backbone dr 172.19.1.1 [ttl 1]
22:03:58.205449 172.19.1.8 > 224.0.0.5: OSPFv2-hello 52: backbone dr 172.19.1.6 bdr 172.19.1.8 [ttl 1]
22:04:02.898652 172.19.1.6 > 224.0.0.5: OSPFv2-hello 48: backbone dr 172.19.1.6 bdr 172.19.1.8 [ttl 1]
22:04:03.906988 172.19.1.7 > 224.0.0.5: OSPFv2-hello 48: backbone dr 172.19.1.7 [ttl 1]
22:04:03.910303 172.19.1.6 > 224.0.0.6: OSPFv2-ls_upd 60: backbone [ttl 1]
22:04:03.910479 172.19.1.6 > 224.0.0.6: OSPFv2-ls_upd 112: backbone [ttl 1]
22:04:03.919802 172.19.1.7 > 224.0.0.5: OSPFv2-ls_upd 64: backbone [ttl 1]
22:04:03.920019 172.19.1.8 > 224.0.0.5: OSPFv2-ls_ack 64: backbone [ttl 1]
```

Ici, on a vu 172.19.1.1, 172.19.1.6 et 172.19.1.8 qui envoyaient régulièrement des messages Hello pour maintenir les adjacences. 172.19.1.7 vient de démarrer et, après les échanges d'Hello, met à jour sa base de données (ls_upd : *Link State Update*). Les routeurs OSPF, sur un réseau à diffusion, écrivent sur des adresses *multicast*, comme 224.0.0.5.

ethereal (logiciel graphique très agréable) ou tethereal (en mode texte) permettent quant à eux une analyse en profondeur du contenu des paquets OSPF. Voici un exemple d'utilisation de tethereal pour afficher pendant trente secondes les paquets OSPF sur eth 1 :

```
% sudo tethereal -n -i eth1 -a duration:30 -V -f 'proto ospf'
Internet Protocol, Src Addr: 194.68.129.103 (194.68.129.103), Dst Addr: 224.0.0.5 (224.0.0.5)
  Version: 4
...
Open Shortest Path First
  OSPF Header
    OSPF Version: 2
    Message Type: Hello Packet (1)
    Packet Length: 48
    Source OSPF Router: 193.51.206.210 (193.51.206.210)
    Area ID: 0.0.0.0 (Backbone)
    Packet Checksum: 0x55cb (correct)
    Auth Type: Null
    Auth Data (none)
  OSPF Hello Packet
    Network Mask: 255.255.255.0
    Hello Interval: 10 seconds
    Options: 0x2 (E)
    Router Priority: 1
    Router Dead Interval: 40 seconds
    Designated Router: 194.68.129.103
    Backup Designated Router: 194.68.129.102
    Active Neighbor: 193.51.206.61
```

Supposons qu'un routeur n'arrive pas à devenir adjacent. tcpdump nous montre qu'il n'envoie pas de paquets mais il est joignable par ping. Testons un **show ip ospf interface eth0** : eth0 is up, line

30. Cette question est discutée en détail (surtout pour IOS) chez Cisco (http://www.cisco.com/en/US/tech/tk648/tk365/technologies_tech)

protocol is up OSPF not enabled on this interface . Cela veut dire que l'interface a été coupée explicitement (**passive-interface eth0**) ou bien qu'aucune directive **network** ne couvre les adresses IP de cette interface.

Sur Quagga, on active le déboguage avec les commandes **debug ospf catégorie**. Par exemple, **debug ospf nsm** affichera dans le journal les changements d'état des voisins (NSM, *Neighbor State Machine*). **debug ospf packet type** est sans doute la commande de déboguage la plus utile. Par exemple, **debug ospf packet ls-update** vous affichera les changements d'état du réseau (LS *Link State*).

Sur IOS, **debug ip ospf packet** vous donnera une bonne idée du trafic OSPF³¹. **debug ip ospf adj** vous montre les arrivées et départs des voisins comme :

```
OSPF: Neighbor 192.134.7.241 is dead
OSPF: neighbor 192.134.7.241 is dead, state DOWN
OSPF: Neighbor change Event on interface Ethernet0
OSPF: DR/BDR election on Ethernet0
OSPF: Elect BDR 192.134.7.245
OSPF: Elect DR 192.134.7.248
DR: 192.134.7.248 (Id)   BDR: 192.134.7.245 (Id)
...
OSPF: 2 Way Communication to neighbor 192.134.7.241
```

Pour tout savoir sur votre réseau, il faut que vous puissiez lire la base de données d'OSPF, ce qui se fait avec la commande **show ip ospf database**. Cela n'est pas trivial et nécessite une bonne compétence OSPF.

Un peu de vocabulaire OSPF : les routeurs OSPF émettent des LSA (*Link State Advertisement*) qui sont des enregistrements donnant des informations sur un routeur et ses liens (LSA de type routeur) ou sur un réseau (LSA de type réseau) ou encore sur un préfixe externe à OSPF (LSA de type externe).

Nous allons commencer par l'exemple décrit en Section 2.2. La base³² contient :

```
laperouse> show ip ospf database ❶

OSPF Router with ID (10.4.100.2) ❷

Router Link States (Area (0.0.0.0))

Link ID          ADV Router      Age  Seq#           CkSum  Link count
10.4.100.2 ❸      10.4.100.2      458  0x8000010a    0xc586  1
192.134.7.241    192.134.7.241  258  0x800000f3    0x79d5  1
192.134.7.245    192.134.7.245  263  0x80000108    0x68fa  2❹
192.134.7.248    192.134.7.248  264  0x800000da    0x9db5  1

Net Link States (Area (0.0.0.0))

Link ID          ADV Router      Age  Seq#           CkSum
10.4.200.1❺      192.134.7.245❻  1096 0x8000000b    0x4baf
192.134.7.248    192.134.7.248  264  0x80000045    0x519a

AS External Link States ❷
```

31. N'oubliez pas un **terminal monitor** pour voir les messages d'erreur, à moins que votre Cisco n'envoie ces informations par syslog.

32. Une des conséquences du choix d'utiliser un protocole de type *Link State* pour OSPF est que la base est identique sur tous les routeurs de la zone.

| Link ID | ADV Router | Age | Seq# | CkSum | Route |
|---------|---------------|------|------------|--------|--------------------|
| 0.0.0.0 | 192.134.7.245 | 1470 | 0x800000bf | 0xbdba | E2 0.0.0.0/0 [0x0] |

- ❶ Cette commande affiche un résumé de la base de données (LS, *Link State Database*) OSPF.
- ❷ C'est le *router ID* de laperouse mais la base est la même sur tous les routeurs de la zone.
- ❸ Le *Link ID* est ici le *Router ID* puisque les LSA sont de type routeur.
- ❹ C'est le nombre de liens du routeur, ici requin qui a deux interfaces Ethernet.
- ❺ Pour un LSA de type réseau, le *Link ID* reflète le préfixe du réseau, contrairement aux LSA de type routeur, où il reflète le *Router ID* du routeur.
- ❻ Les LSA de type réseau ne sont envoyés que pour des réseaux multipoint (comme Ethernet) et le LSA n'est envoyé que par le routeur élu (DR, *Designated Router*).
- ❼ Ces LSA sont ceux qui viennent du monde extérieur à OSPF.

Avec le lien point à point de l'exemple Section 2.4, la base a les mêmes LSA mais le nombre de liens des routeurs de part et d'autre du lien PPP augmente. La base détaillée des LSA de type routeur est (tous les LSA n'ont pas été représentés, seulement ceux des routeurs les plus intéressants) :

```
ospfd> show ip ospf database router
LS age: 94
Options: 2
Flags: 0x0
LS Type: router-LSA
Link State ID: 192.134.7.241
Advertising Router: 192.134.7.241
LS Seq Number: 800000f8
Checksum: 0xc4d0
Length: 60
Number of Links: 3❶

Link connected to: a Transit Network❷
(Link ID) Designated Router address: 192.134.7.248
(Link Data) Router Interface address: 192.134.7.241
Number of TOS metrics: 0
TOS 0 Metric: 10

Link connected to: another Router (point-to-point)
(Link ID) Neighboring Router ID: 10.4.100.2
(Link Data) Router Interface address: 172.22.131.64
Number of TOS metrics: 0
TOS 0 Metric: 10

Link connected to: Stub Network❸
(Link ID) Network/subnet number: 172.22.131.65
(Link Data) Network Mask: 255.255.255.255
Number of TOS metrics: 0
TOS 0 Metric: 10

LS age: 175
Options: 2
Flags: 0x2 : ASBR❹
LS Type: router-LSA
Link State ID: 192.134.7.245
Advertising Router: 192.134.7.245
LS Seq Number: 8000010b
```

```
Checksum: 0x62fd
Length: 48
Number of Links: 2
```

```
Link connected to: a Transit Network
(Link ID) Designated Router address: 192.134.7.248
(Link Data) Router Interface address: 192.134.7.245
Number of TOS metrics: 0
```

TOS 0 Metric: 10

```
Link connected to: a Transit Network
(Link ID) Designated Router address: 10.4.200.1
(Link Data) Router Interface address: 10.4.200.1
Number of TOS metrics: 0
```

TOS 0 Metric: 10

- ❶ En fait, il n'y a que deux "vrais" liens. Voir ❸.
- ❷ Il n'y a guère de détails sur ce réseau : ils seront contenus dans un LSA de type réseau, créé par le routeur élu.
- ❸ Quagga ajoute un lien vers un *stub network*. Pour tous les détails subtils derrière ce comportement, RFC 2328, "2.1. Representation of routers and networks", "12.4.1.1. Describing point-to-point interfaces".
- ❹ requin est ASBR (*Autonomous System Boundary Router*) car il injecte une route externe (la route par défaut).

La base détaillée des LSA de type réseau contient les deux Ethernet :

```
LS age: 1335
Options: 2
LS Type: network-LSA
Link State ID: 10.4.200.1 (address of Designated Router)
Advertising Router: 192.134.7.245
LS Seq Number: 8000000d
Checksum: 0x47b1
Length: 32
Network Mask: /25
Attached Router: 192.134.7.245
Attached Router: 10.4.100.2
```

```
LS age: 937
Options: 2
LS Type: network-LSA
Link State ID: 192.134.7.248 (address of Designated Router)
Advertising Router: 192.134.7.248
LS Seq Number: 80000048
Checksum: 0x4b9d
Length: 40
Network Mask: /28
Attached Router: 192.134.7.241
Attached Router: 192.134.7.245
Attached Router: 192.134.7.248
```

On peut aussi recueillir des informations avec SNMP (Section 1.8.3). RFC 1850 décrit la MIB associée à OSPF. Malheureusement, Quagga n'en met en oeuvre qu'une petite partie. Ici, on interroge un routeur Cisco (14 est le point de départ de la MIB OSPF, voir le RFC ci-dessus) :

```
%snmpwalk -m /usr/share/mibs/OSPF-MIB.txt 192.134.7.246 password 14
...
ospf.ospfNbrTable.ospfNbrEntry.ospfNbrState.192.134.7.245.0 = full(8)
ospf.ospfNbrTable.ospfNbrEntry.ospfNbrState.192.134.7.248.0 = full(8)
```

ce qui permet de voir, entre autres, que le routeur a deux voisins, et qu'il est totalement adjacent.

3.3. Des exemples de débogage

Les exemples de cette section sont présentés sous une forme de questions à une liste de diffusion, avec les informations nécessaires puis les réponses³³Le titre de chaque sous-section fournirait un bon sujet au message.

Important : Avant de poser une question, vous pouvez aussi utiliser un moteur de recherche comme Google (<http://www.google.com/>), qui archive le Web et les News. Par exemple, un message d'erreur incompréhensible est une excellente clé de recherche (à mettre en guillemets pour être sûr que le moteur de la recherche l'utilise littéralement).

3.3.1. Pas d'adjacence établie sur un lien PPP

3.3.1.1. Question

Mes deux routeurs OSPF (un PC/NetBSD et un PC/Linux, tous les deux avec Zebra), ne peuvent établir une adjacence au dessus d'un lien PPP.

Le lien fonctionne :

```
laperouse:~ % ifconfig ppp0
ppp0      Link encap:Point-to-Point Protocol
          inet addr:172.22.131.65  P-t-P:172.22.131.64  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1006  Metric:1
          RX packets:81 errors:3 dropped:0 overruns:0 frame:3
          TX packets:82 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:10
          RX bytes:1623 (1.5 KiB)  TX bytes:1559 (1.5 KiB)
```

```
laperouse:~ % ping -c 3 172.22.131.64
PING 172.22.131.64 (172.22.131.64): 56 octets data
64 octets from 172.22.131.64: icmp_seq=0 ttl=255 time=18.0 ms
64 octets from 172.22.131.64: icmp_seq=1 ttl=255 time=10.2 ms
64 octets from 172.22.131.64: icmp_seq=2 ttl=255 time=10.2 ms
```

```
--- 172.22.131.64 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 10.2/12.8/18.0 ms
```

Mais OSPF ne marche pas, lui :

```
rachel> show ip ospf neighbor
```

```
Neighbor ID      Pri   State           Dead Time   Address          Interface          RXmtL RqstL D
...
```

33. Cela n'a rien à voir avec OSPF mais je regrette l'abondance, sur les listes de diffusion techniques, de messages mal écrits, avec insuffisamment d'informations. J'essaie donc de montrer le bon exemple.

```
10.4.100.2      1      ExStart/DROther 00:00:37      172.22.131.65      ppp0:172.22.131.64      0      0
```

Le voisin ne sort pas de cet état ExStart³⁴. Que puis-je faire ? Au dessus d’Ethernet, les deux machines établissent une adjacence sans problèmes.

3.3.1.2. Réponse

Que contiennent les journaux des deux Zebra ? (N’oubliez pas un **debug ospf packet** .) Je note que votre MTU est de 1006. N’y aurait-il pas un problème à ce sujet ? OSPF exige que les voisins soient d’accord sur la MTU du lien.

3.3.1.3. Solution

En effet, le journal me dit `2003/02/12 14:54:07 OSPF: Packet[DD]: MTU is larger than [ppp0:172.22.131.65]'s MTU. J’ai mis la MTU du lien PPP à 1500 et tout marche. Merci !`

3.3.2. Pas d’adjacence sur un lien Ethernet

3.3.2.1. Question

Sur un lien Ethernet, mes deux routeurs OSPF (un ImageStream et un PC/Linux avec Quagga) ne se voient pas. Sur le PC, **show ip ospf neighbor** ne montre pas du tout l’autre routeur (même pas dans un état autre que *Full*). **tcpdump -n -i eth0 proto ospf** ne montre aucun trafic. Pourtant, les deux machines peuvent se pinguer.

Est-ce un problème de filtrage ? Pourtant, j’autorise tout le trafic UDP.

3.3.2.2. Réponse

OSPF n’utilise pas UDP. Il a son propre protocole, qui porte le numéro 89. Vérifiez que ce protocole est autorisé.

3.3.2.3. Solution

C’était bien cela. **dmesg | grep 89** montre que je rejetais les paquets OSPF : `IN=eth0 OUT=MAC=01:00:5e:00:00:05:00:05:5d:03:86:7e:08:00 SRC=172.19.1.8 DST=224.0.0.5 LEN=68 TOS=0x00 PREC=0x00 TTL=1 ID=51540 PROTO=89. Maintenant, tout marche, merci.`

3.3.3. Incapable de configurer OSPF pour IPv6

OSPF (Zebra 0.93b sur une UltraSparc avec Linux 2.4.19) marche très bien pour IPv4. Maintenant que nous commençons le déploiement d’IPv6, je mets des directives **network** avec des adresses IPv6 dans mon `ospfd.conf` mais `ospfd` ne démarre plus :

```
There is no such command.
Error occured during reading below line.
network 2000::/3 area 0
```

Il me faut une version plus récente de Zebra ?

34. Qui signifie *Exchange Start*, début de l’échange des LSA qui constituent la base de données d’OSPF.

3.3.4. Réponse

Zebra accepte OSPF pour IPv6 depuis longtemps. Mais OSPF pour IPv6 est un protocole différent (quasi-identique mais tournant sur IPv6 et incompatible avec le premier) décrit dans le RFC 2740. Pour Zebra, il faut éditer `ospf6d.conf` et lancer le démon `ospf6d`.

En outre, la configuration est différente : vous n'indiquez pas le réseau, mais l'interface.

3.3.5. Solution

Merci beaucoup, tout marche désormais avec le `ospf6d.conf` suivant :

```
router ospf6
router-id 192.134.7.248
interface eth0 area 0.0.0.0
```

4. Avec plusieurs zones

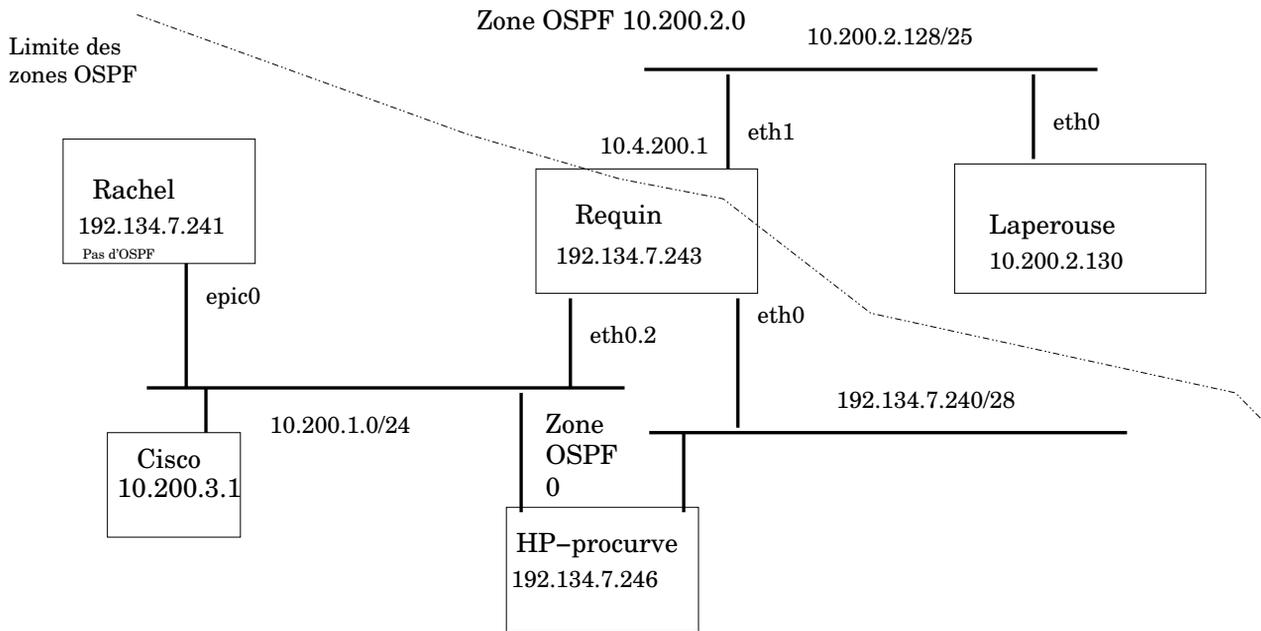
Voici un exemple d'un réseau utilisant le routage hiérarchique. Un tel découpage est utile :

- Si vous avez tellement de routeurs qu'il vient difficile de les gérer dans une même zone (par exemple, la sortie de **show ip ospf database** devient trop longue). Certains routeurs très lents ont également du mal dès que la zone (et donc les informations à traiter) devient trop grande. On dit souvent qu'il est recommandé de ne pas mettre plus de 50 ou 100 routeurs (selon leurs performances) par zone.
- S'il existe plusieurs services dans votre organisation, que ces services disposent de compétence en routage et souhaitent un minimum d'autonomie. OSPF permet, dans une certaine mesure, d'isoler l'épine dorsale (*backbone*) de ces zones.

Important : La règle d'unicité administrative dans l'AS continue à s'appliquer. Si les services en question sont totalement autonomes et ne dépendent pas de l'autorité centrale, il ne faudra pas utiliser un protocole de routage intérieur comme OSPF. Vous devrez faire du routage statique ou bien utiliser un protocole de routage extérieur comme BGP.

Les routeurs qui connectent plusieurs zones sont nommés les ABR (*Area Border Router*).

Figure 6. Réseau à plusieurs zones



requin est l'unique ABR (*Area Border Router*) OSPF. Les deux zones ont comme ID 0 (dorsale) et 10.200.2.0 (il est courant de donner aux identificateurs de zone la forme d'une adresse IPv4).

Sur le commutateur HP-procurve, on voit :

```
NIC HP ProCurve 5308XL# show ip ospf neighbor
```

```
OSPF Neighbor Information
```

| Router ID | Pri | IP Address | NbIfState | State | Rxmt | QLen | Events |
|---------------|-----|---------------|-----------|-------|------|------|--------|
| 192.134.7.243 | 1 | 10.200.1.2 | DR | FULL | 0 | | 6 |
| 10.200.3.1 | 1 | 10.200.1.4 | BDR | FULL | 0 | | 6 |
| 192.134.7.243 | 1 | 192.134.7.243 | DR | FULL | 0 | | 6 |

```
NIC HP ProCurve 5308XL# show ip route
```

```
IP Route Entries
```

| Destination | Network Mask | Gateway | Type | Sub-Type | Metric |
|---------------|-----------------|---------------|-----------|-----------|--------|
| 0.0.0.0 | 0.0.0.0 | 192.134.7.254 | static | | 1 |
| 10.200.1.0 | 255.255.255.0 | VLAN2 | connected | | 0 |
| 10.200.2.128 | 255.255.255.128 | 192.134.7.243 | ospf | InterArea | 11 |
| 127.0.0.0 | 255.0.0.0 | reject | static | | 0 |
| 127.0.0.1 | 255.255.255.255 | lo0 | connected | | 0 |
| 192.134.7.240 | 255.255.255.240 | EUREG-net | connected | | 0 |

```
NIC HP ProCurve 5308XL# show ip ospf interface
```

```
OSPF Interface Status
```

| IP Address | Status | Area ID | State | Auth-type | Cost | Priority |
|------------|--------|---------|-------|-----------|------|----------|
| | | | | | | |

```

10.200.1.1      enabled backbone      DROTHER none      1      1
192.134.7.246 enabled backbone      BDR      none      1      1

```

Sur le Cisco, on voit :

```

cisco-eureg#show ip ospf neighbor

Neighbor ID      Pri   State             Dead Time   Address      Interface
192.134.7.246    1     FULL/DROTHER      00:00:35   10.200.1.1   Ethernet0
192.134.7.243    1     FULL/DR           00:00:31   10.200.1.2   Ethernet0

cisco-eureg#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
E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
U - per-user static route

Gateway of last resort is 10.200.1.1 to network 0.0.0.0

    10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
O IA   10.200.2.128/25 [110/20] via 10.200.1.2, 00:01:46, Ethernet0
C      10.200.1.0/24 is directly connected, Ethernet0
    192.134.7.0/28 is subnetted, 1 subnets
O      192.134.7.240 [110/11] via 10.200.1.1, 00:01:46, Ethernet0
O*E2  0.0.0.0/0 [110/10] via 10.200.1.1, 00:01:46, Ethernet0

```

La base de données contient désormais un nouveau type de LSA : les LSA de type résumé, émis par les ABR dans les zones connectées.

```

cisco-eureg#show ip ospf database

OSPF Router with ID (10.200.3.1) (Process ID 1)

Router Link States (Area 0.0.0.0)

Link ID          ADV Router      Age      Seq#           Checksum Link count
10.200.3.1       10.200.3.1     1303    0x80000039    0xF3C2   1
192.134.7.243   192.134.7.243 147     0x80000045    0xDB5B   2
192.134.7.246   192.134.7.246 145     0x80000003    0xEB96   2

Net Link States (Area 0.0.0.0)

Link ID          ADV Router      Age      Seq#           Checksum
10.200.1.2       192.134.7.243 148     0x80000035    0x4269
192.134.7.243   192.134.7.243 148     0x80000001    0xA53

Summary Net Link States (Area 0.0.0.0)

Link ID          ADV Router      Age      Seq#           Checksum
10.200.2.128    192.134.7.243 1307    0x80000039    0x778B

AS External Link States

Link ID          ADV Router      Age      Seq#           Checksum Tag

```

```
0.0.0.0      192.134.7.246  155   0x80000001 0x3501  0
```

Sur la machine Unix requin :

```
requin> show ip ospf neighbor
```

| Neighbor ID | Pri | State | Dead Time | Address | Interface | RXmtL | RqsL |
|---------------|-----|--------------|-----------|---------------|--------------------|-------|------|
| 192.134.7.246 | 1 | Full/Backup | 00:00:35 | 192.134.7.246 | eth0:192.134.7.243 | 0 | 0 |
| 10.200.2.130 | 1 | Full/Backup | 00:00:31 | 10.200.2.130 | eth1:10.200.2.129 | 0 | 00 |
| 192.134.7.246 | 1 | Full/DROther | 00:00:35 | 10.200.1.1 | eth0.2:10.200.1.2 | 0 | 00 |
| 10.200.3.1 | 1 | Full/Backup | 00:00:32 | 10.200.1.4 | eth0.2:10.200.1.2 | 0 | 00 |

```
requin> show ip ospf route
```

```
===== OSPF network routing table =====
```

```
N  10.200.1.0/24      [10] area: 0.0.0.0
   directly attached to eth0.2
N  10.200.2.128/25   [10] area: 10.200.2.0
   directly attached to eth1
N  192.134.7.240/28  [10] area: 0.0.0.0
   directly attached to eth0
```

```
===== OSPF router routing table =====
```

```
R  192.134.7.246      [10] area: 0.0.0.0, ASBR
   via 192.134.7.246, eth0
   via 10.200.1.1, eth0.2
```

```
===== OSPF external routing table =====
```

```
N E2 0.0.0.0/0      [10/10] tag: 0
   via 192.134.7.254, eth0
```

```
requin> show ip ospf database
```

```
OSPF Router with ID (192.134.7.243)
```

```
Router Link States (Area 0.0.0.0)
```

| Link ID | ADV Router | Age | Seq# | CkSum | Link count |
|---------------|---------------|------|------------|--------|------------|
| 10.200.3.1 | 10.200.3.1 | 1512 | 0x80000039 | 0xf3c2 | 1 |
| 192.134.7.243 | 192.134.7.243 | 355 | 0x80000045 | 0xdb5b | 2 |
| 192.134.7.246 | 192.134.7.246 | 353 | 0x80000003 | 0xeb96 | 2 |

```
Net Link States (Area 0.0.0.0)
```

| Link ID | ADV Router | Age | Seq# | CkSum |
|---------------|---------------|-----|------------|--------|
| 10.200.1.2 | 192.134.7.243 | 355 | 0x80000035 | 0x4269 |
| 192.134.7.243 | 192.134.7.243 | 355 | 0x80000001 | 0xa53 |

```
Summary Link States (Area 0.0.0.0)
```

| Link ID | ADV Router | Age | Seq# | CkSum | Route |
|--------------|---------------|------|------------|--------|-----------------|
| 10.200.2.128 | 192.134.7.243 | 1513 | 0x80000039 | 0x778b | 10.200.2.128/25 |

```
Router Link States (Area 10.200.2.0)
```

| Link ID | ADV Router | Age | Seq# | CkSum | Link count |
|---------------|---------------|-----|------------|--------|------------|
| 10.200.2.130 | 10.200.2.130 | 614 | 0x80000042 | 0xd7d4 | 1 |
| 192.134.7.243 | 192.134.7.243 | 628 | 0x80000034 | 0xf0f3 | 1 |

```
Net Link States (Area 10.200.2.0)
```

```
Link ID      ADV Router    Age  Seq#      CkSum
10.200.2.129 192.134.7.243 628 0x80000030 0x86f1
```

Summary Link States (Area 10.200.2.0)

```
Link ID      ADV Router    Age  Seq#      CkSum  Route
10.200.1.0   192.134.7.243 352 0x80000032 0x9279 10.200.1.0/24
192.134.7.240 192.134.7.243 351 0x80000001 0xc21e 192.134.7.240/28
```

ASBR-Summary Link States (Area 10.200.2.0)

```
Link ID      ADV Router    Age  Seq#      CkSum
192.134.7.246 192.134.7.243 351 0x80000001 0xd2f7
```

AS External Link States

```
Link ID      ADV Router    Age  Seq#      CkSum  Route
0.0.0.0      192.134.7.246 362 0x80000001 0x3501 E2 0.0.0.0/0 [0x0]
```

Sur l'autre machine Unix, laperouse :

```
ospfd# show ip ospf neighbor
```

```
Neighbor ID  Pri  State           Dead Time  Address           Interface          RXmtL RqstL D
192.134.7.243  1   Full/DR         00:00:36  10.200.2.129     eth0:10.200.2.130  0      0
```

```
ospfd# show ip ospf route
```

===== OSPF network routing table =====

```
N IA 10.200.1.0/24      [20] area: (10.200.2.0)
   via 10.200.2.129, eth0
N   10.200.2.128/25    [10] area: (10.200.2.0)
   directly attached to eth0
N IA 192.134.7.240/28 [20] area: (10.200.2.0)
   via 10.200.2.129, eth0
```

===== OSPF router routing table =====

```
R   192.134.7.243      [10] area: (10.200.2.0), ABR
   via 10.200.2.129, eth0
R   192.134.7.246      IA [20] area: (10.200.2.0), ASBR
   via 10.200.2.129, eth0
```

===== OSPF external routing table =====

```
N E2 0.0.0.0/0        [20/10] tag: 0
   via 10.200.2.129, eth0
```

```
ospfd# show ip ospf database
```

OSPF Router with ID (10.200.2.130)

Router Link States (Area (10.200.2.0))

```
Link ID      ADV Router    Age  Seq#      CkSum  Link count
10.200.2.130 10.200.2.130 1394 0x80000043 0xd5d5 1
192.134.7.243 192.134.7.243 1410 0x80000035 0xeef4 1
```

Net Link States (Area (10.200.2.0))

```
Link ID      ADV Router    Age  Seq#      CkSum
10.200.2.129 192.134.7.243 1410 0x80000031 0x84f2
```

Summary Link States (Area (10.200.2.0))

```
Link ID      ADV Router    Age  Seq#      CkSum  Route
10.200.1.0   192.134.7.243 1183 0x80000033 0x907a 10.200.1.0/24
192.134.7.240 192.134.7.243 272 0x80000002 0xc01f 192.134.7.240/28
```

ASBR-Summary Link States (Area (10.200.2.0))

```
Link ID      ADV Router    Age  Seq#      CkSum
192.134.7.246 192.134.7.243 1603 0x80000002 0xd0f8
```

AS External Link States

```
Link ID      ADV Router    Age  Seq#      CkSum  Route
0.0.0.0      192.134.7.246 1142 0x80000002 0x3302 E2 0.0.0.0/0 [0x0]
```

Les configurations sont :

```
hostname requin
router ospf
 network 10.200.1.0/24 area 0
 network 10.200.2.0/24 area 10.200.2.0
 network 192.134.7.0/24 area 0
```

```
hostname laperouse
router ospf
 network 10.200.2.0/24 area 10.200.2.0
```

Commutateur/routeur HP

```
ip router-id 192.134.7.246
router ospf
 area backbone
 redistribute static
 exit
vlan 1
 ip ospf area backbone
 exit
vlan 2
 ip ospf area backbone
 exit
```

Routeur Cisco

```
router ospf 1
 network 10.200.0.0 0.0.255.255 area 0.0.0.0
```

- ❶ IOS fait écrire le masque, tantôt avec les bits réseau à un et les bits machine à zéro (la norme), tantôt l'inverse, comme ici.

5. Conclusion

Il teste beaucoup à voir : la sécurité, les *virtual links*, les zones terminales (*stubs*) et NSSA, l'agrégation des routes aux frontières des zones, la théorie derrière les protocoles à Etats des Liaisons, comme OSPF... Mais vous avez déjà de quoi pratiquer... À vos claviers !

A. Configuration matérielle et logicielle

Les machines utilisées dans les exemples sont :

- requin : un PC/Unix. Système Debian de test ("sid"), noyau Linux 2.4.20 et logiciel de routage Zebra 0.93.
- laperouse : un PC/Unix. Système Debian 3.0 ("woody"), noyau Linux 2.2.19 et logiciel de routage Zebra 0.92.
- cisco : un Cisco 25xx avec IOS 11.1
- hp : un commutateur/routeur Hewlett-Packard ProCurve.
- rachel : un PC/Unix. Système NetBSD 1.6, logiciel de routage Zebra 0.93.

Bibliographie

[quagga] Paul Jakma, *Quagga Home Page*, 2003.

<http://www.quagga.net/>

[moy] John Moy, *OSPF, anatomy of an Internet Routing Protocol*, 1998, Addison-Wesley.

RFC

[RFC 1850] F. Baker, R. Coltun, *OSPF Version 2 Management Information Base*, 1995.

[RFC 2328] J. Moy, *OSPF Version 2*, 1998.

[RFC 2740] R. Coltun, D. Ferguson, J. Moy, *OSPF for IPv6*, 1999.