

Initiation au routage, 2ème partie

Pacôme Massol

pacome.massol(at)laposte.net

Publié par :

Philippe Latu

philippe.latu(at)inetdoc.net

<http://www.inetdoc.net>

Cet article est le second d'une série rédigée par Pacôme Massol sur l'utilisation d'un système GNU/Linux comme routeur. L'article a été publié dans le numéro 43 de Linux Magazine en Octobre 2002. La version publiée ici ne contient que quelques différences mineures sur la présentation et la configuration du logiciel.

Table des matières

1. Copyright et Licence	1
1.1. Méta-information	1
2. Avant-propos	2
3. Pourquoi le routage dynamique ?	2
3.1. Le protocole RIP	2
3.2. Quelles sont les informations de routage à échanger ?	2
3.2.1. La notion de distance	2
3.2.2. Algorithme général de RIP	3
3.2.3. Améliorations de RIPv2 par rapport à RIPv1	4
4. Place à la pratique	4
5. Activation de RIP sur le premier routeur	6
6. Activation de RIP sur le deuxième routeur	7
7. Filtrer la diffusion des routes	7
8. Paramétrage de RIP	8
8.1. La distance administrative	9
8.2. Avant de continuer	9
8.3. La tolérance aux pannes	10
8.4. Un problème de sécurité	10
9. Conclusion	11
9.1. Bibliographie	11
9.2. Liens	11

1. Copyright et Licence

Copyright (c) 2002-2005 Pacôme Massol
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. A copy of the license is included in the section entitled "GNU Free Documentation License".

Copyright (c) 2002-2005 Pacôme Massol
Permission est accordée de copier, distribuer et/ou modifier ce document selon les termes de la Licence de Documentation Libre GNU (GNU Free Documentation License), version 1.1 ou toute version ultérieure publiée par la Free Software Foundation ; sans Sections Invariables ; sans Texte de Première de Couverture, et sans Texte de Quatrième de Couverture. Une copie de la présente licence est incluse dans la section intitulée « Licence de Documentation Libre GNU ».

1.1. Méta-information

Cet article est écrit avec *DocBook*² XML sur un système *Debian GNU/Linux*³. Il est disponible en version imprimable au format PDF : [zebra.rip.pdf](#)⁴.

Toutes les commandes utilisées dans ce document ne sont pas spécifiques à une version particulière des systèmes UNIX ou GNU/Linux. Comme la distribution *Debian GNU/Linux* est utilisée pour l'ensemble des supports du projet *inetdoc.LiNux*, voici une liste des paquets contenant les commandes nécessaires :

² <http://www.docbook.org>

³ <http://www.debian.org>

⁴ <http://www.inetdoc.net/pdf/zebra.rip.pdf>

- net-tools - The NET-3 networking toolkit
- quagga - Unofficial successor of the Zebra BGP/OSPF/RIP routing daemon
- quagga-doc - info files for quagga
- zebra & zebra-doc - anciens paquets non mis à jour depuis 2003.

Les copies d'écran présentées ici correspondent à la publication initiale. Depuis, les versions de Zebra puis de Quagga ont évolué et l'affichage des informations de routage a été modifié. Cependant, ces modifications ne devraient pas gêner les lecteurs.

2. Avant-propos

Le premier article (*Linux Magazine*⁵ numéro 42) vous a présenté les concepts nécessaires à la bonne compréhension du routage IP. Nous avons vu que les routeurs sont de véritables postes d'aiguillage qui acheminent de proche en proche les paquets IP dans l'inter-réseau. On peut configurer manuellement des routes statiques sur chaque routeur. Mais dans un réseau important, cette tâche devient rapidement cauchemardesque !

Heureusement, des protocoles de routage ont été développés afin que les routeurs s'échangent les informations dont ils disposent. On parle dans ce cas de routage dynamique. L'objet de cet article est de vous présenter le fonctionnement et la mise en œuvre d'un protocole de routage des plus élémentaires : RIP (*Routing Information Protocol*).

Avant d'aborder la partie pratique avec Zebra, nous évoquerons les avantages du routage dynamique relativement au routage statique. Nous détaillerons ensuite le fonctionnement du protocole RIP.

3. Pourquoi le routage dynamique ?

Comme nous l'avons défini dans le précédent article, le routage statique consiste à indiquer l'adresse IP des réseaux que l'on cherche à atteindre. On associe à chaque adresse, le nom de l'interface du routeur ou l'adresse IP du routeur voisin se situant sur la route vers ces réseaux de destination. Si le réseau global est complexe, la configuration peut être fastidieuse et source d'erreurs. De plus, lorsque un nouveau réseau est ajouté, il faut reconfigurer l'ensemble. Enfin, pour prévenir tout dysfonctionnement (panne d'un routeur, ligne coupée, etc.), il faut effectuer une surveillance permanente et reconfigurer chaque routeur le cas échéant. Si la route est rétablie, il faut recommencer la manipulation.

L'idée générale du routage dynamique est la suivante : plutôt que de centraliser la configuration du routage dans les mains d'un individu dont le temps de réaction est fatalement long et les risques d'erreurs importants, nous allons délocaliser cette tâche au niveau des routeurs. En effet, chaque appareil n'est-il pas le mieux placé pour connaître les adresses des réseaux auxquels il est directement relié puisque chacune de ses interfaces possède une adresse IP ? De plus, étant directement au contact des supports de communication, il peut établir un diagnostic sur l'état des liaisons. Fort de ces informations, il n'a plus qu'à les partager avec ses voisins. De proche en proche, les nouvelles se répandront à chaque routeur du réseau. L'intervention humaine se situera en amont dans la définition de directives et de règles à appliquer par les routeurs pour la diffusion des routes.

3.1. Le protocole RIP

Comme toujours, pour qu'une communication puisse s'établir, chaque interlocuteur doit parler la même langue. Il a donc été nécessaire de concevoir un protocole. RIP a été défini, pour sa version 1 dans la *RFC1058*⁶ et pour sa version 2 dans la *RFC2453*⁷. Par la suite, je ne traiterai que RIPv2. Toutefois, avant de passer à la partie pratique, nous évoquerons rapidement les différences entre ces deux versions.

3.2. Quelles sont les informations de routage à échanger ?

Le principe général est très simple. Un routeur RIP transmet à ses voisins les adresses réseau qu'il connaît (soit les adresses de ses interfaces, soit les adresses découvertes via les autres routeurs) ainsi que la distance pour les atteindre. Ces couples adresse/distance sont appelés *vecteurs de distance*.

3.2.1. La notion de distance

Nous touchons ici au concept de *métrie*, fondamental dans le domaine du routage. En effet, il arrive fréquemment (c'est même une situation recherchée pour des raisons de tolérance aux pannes) que le réseau ait une topologie maillée. Dans ce cas, plusieurs routes mènent à la même destination. Le routeur doit alors choisir la route qu'il considère la meilleure vers une destination donnée.

La seule métrique utilisée par RIP est la distance correspondant au nombre de routeurs à traverser (*hop count* ou nombre de sauts) avant d'atteindre un réseau. Pour chaque route, RIP calcule la distance. Ensuite, si des routes redondantes apparaissent, RIP retient celle qui traverse le moins de routeur (donc avec la distance la plus faible).

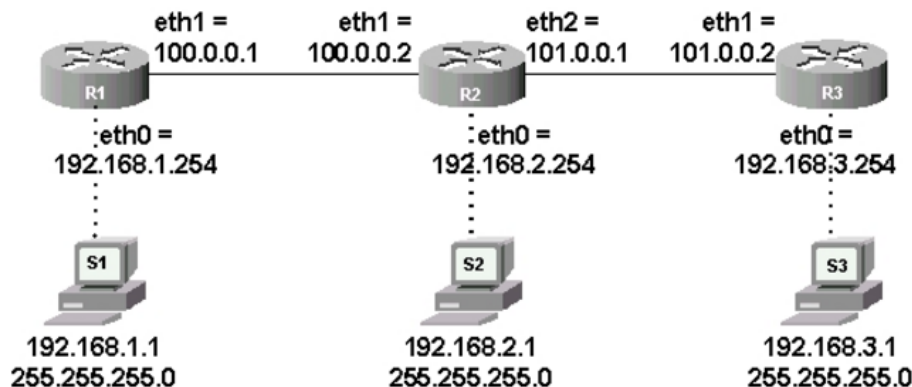
⁵ <http://www.linuxmag-france.org/>

⁶ <http://www.faqs.org/rfcs/rfc1058.html>

⁷ <http://www.faqs.org/rfcs/rfc2453.html>

Du fait de la méthode utilisée pour diffuser les routes, la longueur d'une route est limitée (et par voie de conséquence le diamètre du réseau). La norme limite la distance maximale d'une route à quinze. Cela signifie que deux réseaux ne peuvent être éloignés de plus de quinze routeurs. Nous verrons ci-après qu'une distance égale à seize (distance «infinie» pour RIP) joue un rôle particulier en indiquant qu'une route est devenue inaccessible.

Prenons l'exemple simple du réseau sur lequel nous avons travaillé dans l'article précédent :



Topologie de travail

Afin de bien comprendre le routage dynamique, supposons la situation initiale suivante : sur chaque routeur, toutes les interfaces réseau sont actives, aucune route statique n'est définie et le routage RIP est inactif.

Sur R1, lorsque l'on active le processus de routage RIP, une première table est constituée à partir des adresses IP des interfaces du routeur. Pour ces réseaux directement connectés au routeur, la distance est égale à un puisqu'il faut au moins traverser ce routeur pour les atteindre. On obtient :

Tableau 1. Table initiale constituée par R1

Adresse/Préfixe	Moyen de l'atteindre	Distance
100.0.0.0/8	eth1	1
192.168.1.0/24	eth0	1

R1 transmet à ses voisins immédiats (ici, il n'y a que R2) un seul vecteur de distance {192.168.1.0/24, 1} qui signifie : «je suis le routeur d'adresse IP 100.0.0.1 et je connais un moyen d'atteindre le réseau 192.168.1.0/24 en un saut». Aucune information sur le réseau commun aux deux routeurs (100.0.0.0/8) n'est transmise car R1 considère que R2 connaît déjà ce réseau.

Ensuite, lorsque l'on active RIP sur R2, il constitue la table ci-après à partir de ses propres informations et de celles reçues de R1 :

Tableau 2. table constituée par R2

Adresse/Préfixe	Moyen de l'atteindre	Distance
100.0.0.0/8	eth1	1
101.0.0.0/8	eth2	1
192.168.1.0/24	100.0.0.1	2
192.168.2.0/24	eth0	1

Sur R2, RIP a calculé que la distance pour atteindre 192.168.1.0/24 est égale à deux puisqu'il faut traverser R2 puis R1. R2 a déduit le «moyen de l'atteindre» à partir de l'adresse IP de l'émetteur contenue dans le paquet RIP.

Lorsque RIP sera démarré sur R3, la route vers 192.168.3.0/24 avec une distance de deux sera ajoutée dans la table ci-dessus.

Dans ce petit exemple, aucune restriction n'a été définie sur la diffusion des routes. Donc, à l'issue d'un certain délai appelé *temps de convergence*, variable selon la taille du réseau, chaque routeur connaît un moyen d'atteindre chaque réseau.

3.2.2. Algorithme général de RIP

Examinons un peu plus en détail le fonctionnement de RIP. Lors de l'initialisation du routeur, celui-ci détermine l'adresse réseau de ses interfaces puis envoie sur chacune une demande d'informations (table RIP complète) aux routeurs voisins. Lors de la réception d'une demande, un routeur envoie sa table complète ou partielle suivant la nature de cette demande. Lors de la réception d'une réponse, il met à jour sa table si besoin. Trois cas peuvent se présenter :

- pour une nouvelle route, il incrémente la distance, vérifie que celle-ci est strictement inférieure à 15 et diffuse immédiatement le vecteur de distance correspondant,
- pour une route existante mais avec une distance plus faible, la table est mise à jour. La nouvelle distance et, éventuellement, l'adresse du routeur si elle diffère sont intégrées à la table,
- pour une route existante mais avec une distance plus importante, la table est mise à jour si la nouvelle distance est émise par le même routeur voisin que précédemment.

Bien sûr, si l'appareil reçoit une route dont la distance est supérieure à celle déjà connue d'un autre voisin, RIP l'ignore. Ensuite, à intervalles réguliers (toutes les 30 secondes), la table RIP est diffusée qu'il y ait ou non des modifications.

Des routes doivent être retirées de la table gérée par RIP dans deux situations :

- En premier lieu, si un réseau immédiatement connecté devient inaccessible (panne de l'interface, de la ligne, modification de la topologie par l'administrateur, etc.), les routeurs RIP reliés à ce réseau affectent dans leur table une distance «infinie» (16 comme indiqué plus haut) à cette route. Elle est conservée pendant la durée d'un temporisateur de «maintien» (*garbage collect*) de 120 secondes puis est supprimée. Immédiatement après, le vecteur avec une distance «infinie» est diffusé. Un routeur qui reçoit un vecteur avec une distance de 16 comprend : «il faut que tu retires cette route de ta table car elle est devenue invalide !» De proche en proche, cette information se propage.
- En second lieu, un routeur du réseau tombe en panne. Cela veut peut-être dire que les réseaux situés derrière cet appareil sont devenus inaccessibles. Mais comment savoir si un routeur est en panne ? RIP considère qu'un routeur qui n'a pas donné de nouvelles depuis trois minutes est hors service. Pour gérer cette situation, il attribue à toutes les routes dynamiques un temporisateur initialisé à 180 secondes par défaut. A chaque réception d'un vecteur de distance déjà présent dans la table, le compteur est réinitialisé. Mais si jamais ce compteur atteint zéro, la route est considérée comme invalide. On se retrouve alors dans la situation précédente (distance infinie, temporisateur de maintien, diffusion de l'information puis suppression de la route). Maintenant, si un autre routeur connaît une route menant vers un des réseaux que l'on vient de retirer, c'est parfait ! Notre routeur intégrera cette nouvelle route dans sa table. De cette façon, RIP permet la **tolérance aux pannes**.

Comment justifier l'existence de ces mécanismes qui peuvent paraître un peu complexes ? Cela est dû à une faiblesse des algorithmes à vecteurs de distance que l'on appelle «problème de la convergence lente». Dans certains cas, après la panne d'un accès réseau, deux routeurs voisins risquent de se transmettre mutuellement puis, ensuite, de propager des informations contradictoires au sujet de ce réseau et créer ainsi une boucle de routage infinie. Zebra et son successeur Quagga mettent en œuvre les mécanismes suivants :

split horizon

Une information de routage reçue sur une interface n'est jamais retransmise sur celle-ci.

poison reverse

Les mises à jour de routage *poison reverse* appliquent une métrique «infinie» aux routes transmises par l'interface d'émission. Ce type de mise à jour aide à prévenir les boucles de routage.

triggered update

Une panne est immédiatement diffusée sans attendre le prochain cycle de diffusion des tables afin de réduire le délai de convergence.

3.2.3. Améliorations de RIPv2 par rapport à RIPv1

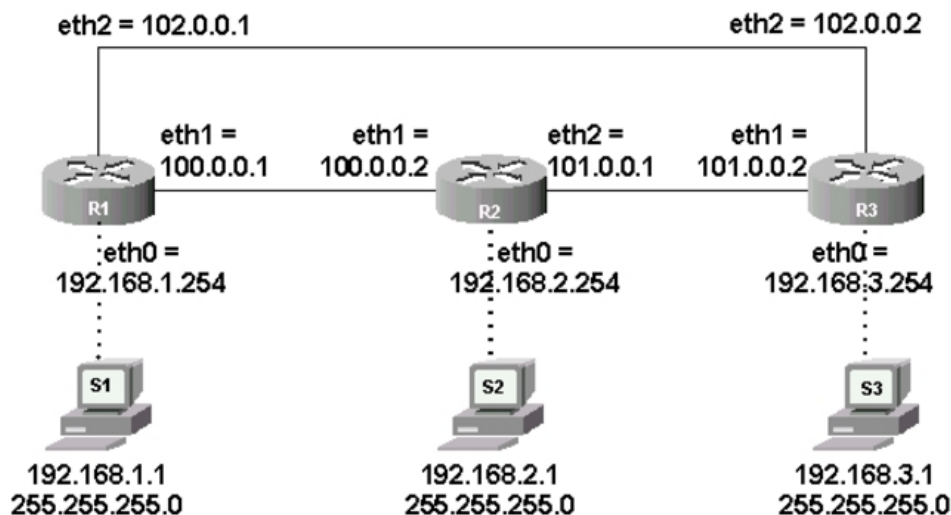
Même si les principes évoqués ci-dessus sont valables quelle que soit la version de RIP, les différences restent intéressantes à relever. Les améliorations de RIPv2 sont :

- Diffusion des masques de sous-réseaux associés aux adresses réseaux (RIPv1 n'utilisait que les masques réseau par défaut).
- Utilisation d'une adresse de *multicast* pour diffuser les vecteurs de distance au lieu de l'adresses de *broadcast* ; ce qui réduit l'encombrement sur le réseau.
- Support de l'authentification en transportant un mot de passe crypté avec MD5.
- Interopérabilité entre protocoles de routage en diffusant des routes apprises à partir d'autres protocoles.

L'ensemble de ces raisons rendent RIPv1 obsolète bien qu'il soit encore supporté par la plupart des routeurs logiciels ou matériels.

4. Place à la pratique

Afin de mieux apprécier les facilités offertes par le routage dynamique, je vous propose de travailler sur une topologie légèrement modifiée afin d'introduire un lien redondant.



Topologie de travail

Que vous ayez suivi ou non la première partie de cet article, vous devez partir sur chaque appareil avec un fichier de configuration du routeur Zebra ou Quagga (/etc/quagga/zebra.conf avec le paquet Quagga) vierge à l'exception de ces deux lignes :

```
hostname Rx(Zebra)
password foo
```

- Remplacez le x de Rx(Zebra) par le numéro du routeur
- A la place de foo, indiquez le mot de passe que vous souhaitez saisir lorsque vous vous connecterez au routeur via telnet.

Vous devez également créer un fichier de configuration pour le routeur RIP (/etc/quagga/ripd.conf avec le paquet Quagga) ayant une apparence très proche du précédent :

```
hostname Rx(RIP)
password foo
```

Lorsque ces manipulations sont faites, lancez les deux démons de routage sur les trois routeurs.

Configuration manuelle sans paquet

Exemple du routeur R1 ; respectez bien l'ordre des commandes :

```
R1 # zebra -d
R1 # ripd -d
```

Configuration avec le paquet Quagga :

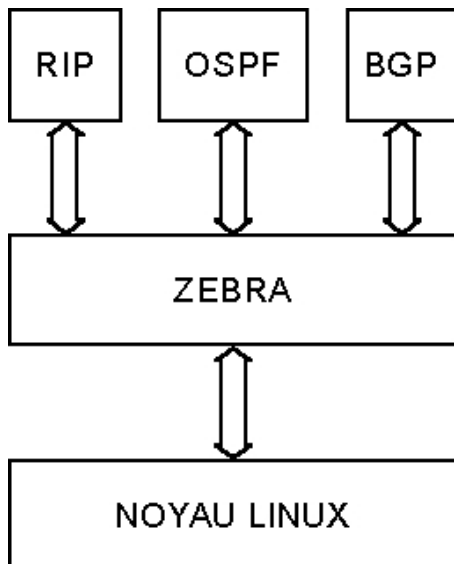
Le fichier /etc/quagga/daemons spécifie la liste des démons à utiliser :

```
# This file tells the quagga package which daemons to start.
#
# Entries are in the format: <daemon>=(yes|no|priority)
# 0, "no" = disabled
# 1, "yes" = highest priority
# 2 .. 10 = lower priorities
# Read /usr/share/doc/quagga/README.Debian for details.
#
# Sample configurations for these daemons can be found in
# /usr/share/doc/quagga/examples/.
#
# ATTENTION:
#
# When activation a daemon at the first time, a config file, even if it is
# empty, has to be present *and* be owned by the user and group "quagga", else
# the daemon will not be started by /etc/init.d/quagga. The permissions should
# be u=rw,g=r,o=.
# When using "vtysh" such a config file is also needed. It should be owned by
# group "quaggavty" and set to ug=rw,o= though.
#
zebra=yes
bgpd=no
ospfd=no
ospf6d=no
ripd=yes
ripngd=no
isisd=no
```

La commande suivante sert à relancer les démons après avoir édité le fichier ci-dessus :

```
R1 # /etc/init.d/quagga restart
```

Zebra ou Quagga nécessitent que les deux démons soient présents. L'architecture est la suivante :



Architecture de Zebra

Le démon **zebra** est un intermédiaire entre le noyau de Linux et les démons de routage dynamique. Il peut récupérer les routes statiques définies directement sous Linux afin de les diffuser via le routage dynamique. **zebra** lui-même permet de définir des routes statiques. Enfin, il peut récupérer des routes dynamiques pour les intégrer à la table de routage gérée par le noyau Linux. Les routages statique et dynamique peuvent cohabiter sans problème avec Zebra mais les concepteurs du logiciel conseillent fortement de ne définir les routes statiques que dans **zebra** (évitons de les définir dans le shell Linux ou dans les démons de routage dynamique).

5. Activation de RIP sur le premier routeur

Afin d'observer la diffusion des routes qu'opère RIP, je vous propose de saisir la commande suivante dans le shell d'un routeur immédiatement voisin de R1, R2 par exemple :

```
R2 # tcpdump -i eth1 -nt -s 0 src host 100.0.0.1
tcpdump: listening on eth1
```

Ensuite, connectons-nous au routeur RIP sur R1 avec un telnet sur le port 2602. Dans le shell de R1 :

```
R1 # telnet localhost 2602

Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

Hello, this is zebra (version 0.91a).
Copyright 1996-2001 Kunihiro Ishiguro.

User Access Verification

Password:

R1(RIP) >
```

Cette opération étant réalisée, comme pour **zebra** il faut activer le mode privilégié, passer dans le terminal de configuration et enfin, entrer dans la configuration du routeur RIP :

```
R1(RIP)> enable
R1(RIP)# conf t
R1(RIP)(config)# router rip
R1(RIP)(config-router)# version 2
```

La première tâche consiste à déterminer les types de routes en notre «possession» que nous souhaitons voir diffuser à nos voisins. Cette configuration se fait par la commande **redistribute**. Voici les différents paramètres de cette commande :

```
R1(RIP)(config-router)# redistribute ?

  bgp Border Gateway Protocol (BGP)
  connected Connected
  kernel Kernel routes
  ospf Open Shortest Path First (OSPF)
  static Static routes
```

On constate que l'on peut diffuser des routes propres à la machine comme les routes statiques et les adresses des réseaux directement connectés. Mais nous pouvons également utiliser RIP pour diffuser des routes dynamiques apprises via RIP ou d'autres protocoles de routage comme OSPF ou BGP. Dans tous les cas, les routes diffusées aux voisins seront vues par eux comme des routes étiquetées «découvertes grâce à RIP».

Nous choisissons de diffuser les adresses des réseaux directement connectés :

```
R1(RIP)(config-router)# redistribute connected
```

Pour l'instant, rien ne se produit. Il faut indiquer à RIP sur quels réseaux nous souhaitons voir la diffusion des routes s'opérer. Nous retrouvons ici une commande commune avec le routage statique. Avant de la valider, pensez à observer le résultat du `tcpdump` sur l'écran de R2 :

```
R1(RIP)(config-router)# network 100.0.0.0/8
```

À ce stade, R1 diffuse sur le réseau 100.0.0.0/8 la table RIP toutes les 30 secondes. Le résultat sur R2 doit ressembler à ceci :

```
R2 # tcpdump -i eth1 -nt -s 0 src host 100.0.0.1
tcpdump: listening on eth1

100.0.0.1.router > 224.0.0.9.router: RIPv2-req 24 (DF) [ttl 1]

100.0.0.1 > 224.0.0.9: igmp v2 report 224.0.0.9 (DF) [ttl 1]

100.0.0.1.router > 224.0.0.9.router: RIPv2-resp [items 2]:
{102.0.0.0/255.0.0.0}(1)
{192.168.1.0/255.255.255.0}(1) (DF) [ttl 1]
```

Les messages adressés par R1 se font via une adresse *multicast* convenue pour les routeurs RIP : 224.0.0.9. Les dernières lignes montrent clairement que RIP diffuse deux vecteurs de distance : un concernant le réseau 102.0.0.0/8 et un autre concernant le réseau 192.168.1.0/24. Observons sur R1 la table avec laquelle RIP travaille :

```
R1(RIP)(config-router)# end
R1(RIP)# show ip rip
Codes: R - RIP, C - connected, O - OSPF, B - BGP
   Network      Next Hop      Metric From      Time
C 100.0.0.0/8
C 102.0.0.0/8
C 192.168.1.0/24
```

RIP a été activé sur le réseau 100.0.0.0/8, donc aucune information le concernant n'est diffusée sur ce même réseau pour des raisons évidentes d'optimisation mais aussi, pour la gestion du problème de la convergence lente (Voir *poison reverse*).

6. Activation de RIP sur le deuxième routeur

Bien, nous avons fait la moitié du travail. Un routeur diffuse grâce à RIP les informations de routage qu'il possède. Mais pour l'instant, c'est inefficace car personne n'est là pour les écouter et les exploiter. Il faut donc faire les mêmes manipulations sur R2 puis à terme sur R3. Passons dans le shell de R2 :

```
R2 # telnet localhost 2602
...
R2(RIP)> enable
R2(RIP)# conf t
R2(RIP)(config)# router rip
R2(RIP)(config-router)# redistribute connected
R2(RIP)(config-router)# network 100.0.0.0/8
R2(RIP)(config-router)# end
R2(RIP)# show ip rip
Codes: R - RIP, C - connected, O - OSPF, B - BGP
   Network      Next Hop      Metric From      Time
C 100.0.0.0/8
C 101.0.0.0/8
R 102.0.0.0/8    100.0.0.1      2      100.0.0.1      02:52
R 192.168.1.0/24 100.0.0.1      2      100.0.0.1      02:52
C 192.168.2.0/24
```

La table ci-dessus a été constituée par le processus RIP exécuté sur R2. Le routeur d'adresse 100.0.0.1 (R1) l'a informé de la présence de deux routes vers deux réseaux, ce qui est conforme aux informations affichées par `tcpdump` tout à l'heure. La distance (Metric) est égale à deux puisque ces réseaux sont directement connectés à R1. Un compteur est activé pour chaque route dynamique notée R (pour RIP). C'est un compte à rebours qui périodiquement redémarre à 03:00 (soit 180 secondes ou 6 périodes de mise à jour de la table de routage) à chaque diffusion reçue de R1.

Vous pouvez lancer la commande `show ip rip` sur R1 afin de constater qu'il a effectué un travail similaire.

7. Filtrer la diffusion des routes

Lorsque l'on saisit la commande `redistribute connected` dans RIP, le routeur diffuse toutes les routes de type «directement connectées», sans distinction. Difficile de garder une certaine «intimité» dans ces conditions ! Zebra, qui est bien conçu, propose des mécanismes pour filtrer la diffusion des routes grâce aux «listes de distribution».

Supposons qu'un nouveau réseau soit connecté à R2. Pour les besoins de l'exemple, vous pouvez créer une interface fictive simulant ce réseau. Dans le shell Linux, créons cette interface :


```
R2 # ifconfig dummy0 111.0.0.1/8 up
```

Zebra détecte cette nouvelle interface et transmet l'information à RIP. Comme à ce stade, RIP doit diffuser toutes les routes connectées. Il informe immédiatement ses voisins. Vérifions ceci sur R1 :

```
R1(RIP)# show ip rip
Codes: R - RIP, C - connected, O - OSPF, B - BGP
Network      Next Hop      Metric From      Time
C 100.0.0.0/8          1
R 101.0.0.0/8          100.0.0.2      2      100.0.0.2      02:43
C 102.0.0.0/8          1
R 111.0.0.0/8          100.0.0.2      2      100.0.0.2      02:43
C 192.168.1.0/24        1
R 192.168.2.0/24        100.0.0.2      2      100.0.0.2      02:43
R1(RIP)#
```

On constate que R1 a appris l'existence de 111.0.0.0/8. Nous allons interdire à R2 de diffuser l'existence de ce réseau à ses petits camarades. Pour ce faire, il faut créer une règle indiquant que l'adresse 111.0.0.0/8 est bloquée grâce à une *liste de contrôle d'accès*. Ensuite, il faut affecter cette règle à une *liste de distribution* qui indiquera sur quelle interface l'appliquer. Retournons sur R2, dans le terminal de configuration de RIP :

```
R2(RIP)> enable
R2(RIP)# conf t
R2(RIP)(config)# access-list❶ 1 deny 111.0.0.0/8
R2(RIP)(config)# access-list 1❷ permit any
```

- ❶ Définition de la règle.
- ❷ Le chiffre '1' après la commande **access-list** identifie la liste de contrôle d'accès. Ce numéro sera utilisé pour l'associer à la liste de distribution. N'oubliez pas la deuxième ligne. Il faut dire explicitement à RIP que toutes les autres adresses ne sont pas bloquées.

Maintenant, affectons la liste de contrôle d'accès à une liste de distribution. Il faut indiquer sur quelles interfaces ces règles sont à appliquer :

```
R2(RIP)(config)# router rip
R2(RIP)(config-router)# distribute-list 1 out eth1
R2(RIP)(config-router)# distribute-list 1 out eth2
```

A partir de cet instant, plus aucune information n'est diffusée par R2 concernant 111.0.0.0/8. Sur R1, avec une commande **show ip rip**, vous constaterez que le temporisateur de la route tombe à 0. Elle se voit ensuite attribuer une métrique infinie pendant le délai du temporisateur *garbage collect* puis elle disparaît.

Dans notre exemple, le résultat de cette manipulation est que les réseaux directement connectés au routeur R2, en particulier 192.168.2.0/24 qui contient des ordinateurs, peuvent communiquer avec 111.0.0.0/8. En revanche, l'extérieur n'a pas connaissance du réseau 111.0.0.0/8 qui ne peut pas communiquer avec les réseaux situés derrière les autres routeurs.

Cet article n'a pas la prétention de présenter toutes possibilités offertes par les listes de contrôle d'accès et les listes de distribution qui sont, en fait, très nombreuses. La documentation du logiciel indique l'ensemble des paramètres de ces différentes commandes.

8. Paramétrage de RIP

Toute la configuration de RIP peut être affichée sous une forme synthétique. Par exemple, sur le routeur R1, en mode privilégié (#) :

```
R1(RIP)# show ip protocols
Routing Protocol is "rip"
Sending updates every 30 seconds with +/-50%, next due in 35
Timeout after 180 seconds, garbage collect after 120 seconds
Outgoing update filter list for all interface is not set
Incoming update filter list for all interface is not set
Default redistribution metric is 1
Redistributing: connected
Default version control: send version 2, receive version 2
Interface      Send      Recv      Key-chain
eth1            2         2
Routing for Networks:
100.0.0.0/8
Routing Information Sources:
Gateway    BadPackets BadRoutes Distance Last Update
100.0.0.2  0           0        120      00:00:34
Distance: (default is 120)
```

Examinons brièvement les principaux champs :

- Les différents temporisateurs sont fixés aux valeurs par défaut.
- Aucun filtrage des routes en entrée comme en sortie n'est défini.
- La métrique par défaut de ce routeur est égale à un (c'est cette valeur qui sera ajoutée aux distances des routes apprises dynamiquement).

- Zebra et Quagga supportent les deux versions de RIP que l'on peut faire cohabiter mais par défaut, Zebra n'autorise en réception comme en émission que la version 2.
- Le routage n'est activé pour l'instant que sur l'interface Ethernet 1. Aucun mot de passe n'est défini (nous aborderons cette notion un peu plus loin).
- La dernière ligne concerne la *distance administrative*. Comme cette notion est importante, nous la développons ci-dessous.

8.1. La distance administrative

La dernière ligne du listing précédent évoque une «distance» dont la valeur par défaut est 120. Il s'agit de la *distance administrative*. Elle n'a aucun rapport avec la distance (métrique) en nombre de sauts calculée par RIP.

Zebra et Quagga peuvent constituer une table de routage à partir de routes apprises de différentes manières (réseau directement connecté, route statique, RIP, OSPF, BGP). Si Zebra ou Quagga se trouvent avec plusieurs routes menant vers un même réseau mais rapportée par des moyens différents, il doit en choisir une. Il a été décidé d'attribuer à chaque moyen d'apprendre une route un score. La route découverte par un moyen dont le score est le plus faible sera élue. Les distances administratives standards sont les suivantes :

Tableau 3. Distances administratives par défaut

Découverte d'une route	Distance administrative
Connected	0
Static	1
BGP	20
OSPF	110
RIP	120

Ainsi, une route configurée de façon statique (donc par un administrateur) est jugée plus crédible qu'une même route rapportée par RIP (notez au passage que RIP est considéré comme le moins crédible...). On retrouve cette notion de distance administrative dans la table de routage de Zebra. Sur R1, connectez vous avec telnet au terminal de configuration de Zebra (dans le shell, faites un **telnet localhost 2601**, puis saisissez le mot de passe) :

```
R1(Zebra)> show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
      B - BGP, > - selected route, * - FIB route

C>* 100.0.0.0/8 is directly connected, eth1
R>* 101.0.0.0/8 [120/2] via 100.0.0.2, eth1, 00:08:11
C>* 102.0.0.0/8 is directly connected, eth2
C>* 127.0.0.0/8 is directly connected, lo
C>* 192.168.1.0/24 is directly connected, eth0
R>* 192.168.2.0/24 [120/2] via 100.0.0.2, eth1, 00:08:11
R1(Zebra)>
```

Les deux routes dynamiques notées R comportent deux nombres entre crochets ([120/2]). Le premier correspond à la distance administrative et le deuxième à la distance en nombre de sauts.



Remarque importante

J'en profite pour bien préciser que la table ci-dessus est la table de routage, donc utilisée par l'appareil pour router les paquets IP reçus sur ses interfaces réseau. La table que vous consultez via le démon RIP en utilisant la commande **show ip rip** n'est pas la table de routage, c'est la table qui sera diffusée aux routeurs voisins. La signification de ces deux tables est donc radicalement différente.

8.2. Avant de continuer

Je vous invite maintenant à activer RIP sur vos trois routeurs en redistribuant les adresses des réseaux immédiatement connectés sur tous les réseaux. Vous connaissez les manipulations à effectuer. A la fin du processus, chaque routeur doit connaître les adresses des six réseaux ainsi que le moyen de les atteindre. Pour information, je vous donne le contenu du fichier de configuration de R3 (fichier `/etc/quagga/ripd.conf`) :

```
hostname R3(RIP)
password foo
!
interface lo
!
interface eth0
!
interface eth1
!
```

```
interface eth2
!
router rip
redistribute connected
network 101.0.0.0/8
network 102.0.0.0/8
!
line vty
!
end
```

Vous pouvez également, si vous le souhaitez, modifier la valeur par défaut des temporisateurs utilisés par RIP afin de visionner plus rapidement le résultat des manipulations que nous allons réaliser par la suite. Ceci se fait de la façon suivante, par exemple dans R1 :

```
R1(RIP)# conf t
R1(RIP)(config)# router rip
R1(RIP)(config-router)# timers basic 10 30 20
```

Notez bien qu'en exploitation, je vous conseille vivement de conserver ces compteurs à leur valeur par défaut. Avec les durées que nous avons indiqué ici, une partie importante de votre bande passante va être consommée par les diffusions de RIP.

8.3. La tolérance aux pannes

Supposons que la liaison entre R1 et R2 va tomber en panne. Visionnons la table RIP de R1 avant ce triste événement :

```
R1(RIP)> show ip rip
Codes: R - RIP, C - connected, O - OSPF, B - BGP
Network      Next Hop      Metric From      Time
C 100.0.0.0/8          1
R 101.0.0.0/8          100.0.0.2      2      100.0.0.2      02:52
C 102.0.0.0/8          1
C 192.168.1.0/24       1
R 192.168.2.0/24       100.0.0.2      2      100.0.0.2      02:52
R 192.168.3.0/24       102.0.0.2      2      102.0.0.2      02:36
```

Le réseau 100.0.0.0/8 tombe en panne. R1 ne reçoit donc plus d'informations de routage à partir de R2. Si vous observez la table RIP sur R1, vous verrez que toutes les routes issues de R2 finissent par disparaître. Mais pendant ce temps, R3 continue à envoyer des mises à jour via le réseau 102.0.0.0/8. R3 connaît un moyen d'atteindre les réseaux que l'on pouvait joindre auparavant par R2. Aussi, au bout d'un certain délai de convergence, R1 construit la table suivante :

```
R1(RIP)> show ip rip
Codes: R - RIP, C - connected, O - OSPF, B - BGP
Network      Next Hop      Metric From      Time
R 100.0.0.0/8          102.0.0.2      3      102.0.0.2      02:34
R 101.0.0.0/8          102.0.0.2      2      102.0.0.2      02:34
C 102.0.0.0/8          1
C 192.168.1.0/24       1
R 192.168.2.0/24       102.0.0.2      3      102.0.0.2      02:34
R 192.168.3.0/24       102.0.0.2      2      102.0.0.2      02:34
```

Tous les réseaux sont à nouveau accessibles à partir de R1 ! Cela démontre que RIP a su digérer une panne de liaison. Rétablissons le lien entre R1 et R2. Progressivement, on retourne vers la première table car les métriques via R2 sont plus faibles.

8.4. Un problème de sécurité

Le routage dynamique est pratique car avec très peu de commandes de configuration on arrive à une solution qui fonctionne correctement et qui est même capable de prendre en compte automatiquement des modifications de la topologie. Seulement voilà : imaginez qu'un petit malin insère sur le réseau un routeur RIP et qu'il lui fasse diffuser des routes totalement farfelues. Cela peut créer un certain nombre de désagréments comme des dénis de service par exemple. Pour limiter ce risque, RIPv2 permet d'associer un mot de passe crypté à chaque diffusion de vecteurs de distance. Seuls les routeurs ayant connaissance de ce mot de passe traiteront les informations de routage. Mettons en place ce mécanisme entre R1 et R2 :

```
R1(RIP)# conf t
R1(RIP)(config)# key chain test
R1(RIP)(config-keychain)# key 1
R1(RIP)(config-keychain-key)# key-string motdepasse
R1(RIP)(config-keychain-key)# exit
R1(RIP)(config-keychain)# exit
R1(RIP)(config)# int eth1
R1(RIP)(config-if)# ip rip authentication mode md5
R1(RIP)(config-if)# ip rip authentication key-chain test
R1(RIP)(config-if)#
```

Nous créons le porte-clé (*keychain*) nommé test avec le mot de passe motdepasse. Ce mot de passe est associé à l'interface eth1, il sera transmis au format MD5 (sinon, il est transmis en clair !). Pour que cela fonctionne, vous devrez faire des manipulations identiques sur R2.

Examinons avec une capture de paquets le contenu des informations de routage reçues de R2 :

```
R1 # tcpdump -i eth1 -nt -s0 src host 100.0.0.2
```

```

tcpdump: listening on eth1
100.0.0.2.router > 224.0.0.9.router: RIPv2-resp [items 6]:
[auth 3: 0068 0114 3cfb 0c6f 0000 0000 0000 0000]
{101.0.0.0/255.0.0.0}(1)
{102.0.0.0/255.0.0.0}(2)
{192.168.2.0/255.255.255.0}(1)
{192.168.3.0/255.255.255.0}(2)
[auth 1: 4d71 f8e0 077c cc58 8247 6656 17c3 95f2]
(DF) [ttl 1]

```

Notez au passage que seul le mot de passe est crypté, les informations de routage continuent à circuler en clair.

9. Conclusion

RIP constitue un excellent moyen pédagogique pour aborder la problématique du routage dynamique. Mais il est peu utilisé en exploitation car il souffre de certaines limitations et défauts qui le cantonnent à des réseaux de taille moyenne. Nous avons vu que le diamètre maximum d'un réseau géré avec RIP est limité à 15 routeurs soit 16 segments de réseau. RIP est un gros consommateur de bande passante du fait de la méthode utilisée pour diffuser les informations de routage (toutes les 30 secondes, l'intégralité de la table RIP est diffusée même si elle n'a subi aucune modification). C'est fâcheux, en particulier sur des liaisons lentes ou facturées au volume de données transférées. La métrique utilisée ne garantit pas que le routage soit optimal. En effet, la distance masque les caractéristiques réelles de la voie de transmission (débit ou coût en particulier). Enfin, le temps de convergence, délai avant que tous les routeurs ne possèdent des tables cohérentes peut être long dans certaines situations. Pour toutes ces raisons, on a cherché à développer un protocole de routage beaucoup plus efficace : OSPF, objet du prochain article.

9.1. Bibliographie

TCP/IP : Architecture, protocoles et applications

Douglas COMER, DUNOD. ISBN: 2-10-005384-1 (09/2001) 830 p.

Le routage dans l'Internet

Christian HUITEMA, EYROLLES. ISBN: 2-212-08902-3 (10/1994) 418 p.

9.2. Liens

- *Quagga*⁸
- *Linux Magazine*⁹
- Version originale du document et page personnelle de *Pacôme Massol*¹⁰

⁸ <http://www.quagga.net/>

⁹ <http://www.linuxmag-france.org/>

¹⁰ <http://www.pmassol.net/>