RFC 9079 : Source-Specific Routing in Babel

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 29 août 2021

Date de publication du RFC : Août 2021

https://www.bortzmeyer.org/9079.html

Traditionnellement, le routage dans l'Internet se fait sur la base de l'adresse IP de **destination**. Mais on peut aussi envisager un routage où l'adresse IP source est prise en compte. Ce nouveau RFC étend le protocole de routage Babel en lui ajoutant cette possibilité.

Babel, normalisé dans le RFC 8966 [1] est un protocole de routage à vecteur de distance. Il est prévu par défaut pour du routage IP traditionnel, où le protocole de routage construit une table de transmission qui associe aux préfixes IP de destination le routeur suivant ("*next hop*"). Un paquet arrive dans un routeur ? Le routeur regarde dans sa table le préfixe le plus spécifique qui correspond à l'adresse IP de destination (utilisant pour cela une structure de données comme, par exemple, le "*Patricia trie*") et, lorsqu'il le trouve, la table contient les coordonnées du routeur suivant. Par exemple, si une entrée de la table dit que `2001:db8:1:fada::/64` a pour routeur suivant `fe80::f816:3eff:fef2:47db%eth0`, un paquet destiné à `2001:db8:1:fada::bad:1234` sera transmis à ce `fe80::f816:3eff:fef2:47db`, sur l'interface réseau `eth0`. Cette transmission classique a de nombreux avantages : elle est simple et rapide, et marche dans la plupart des cas.

Mais ce mécanisme a des limites. On peut vouloir prendre en compte d'autres critères que l'adresse de destination pour la transmission d'un paquet. Un exemple où un tel critère serait utile est le "*multihoming*". Soit un réseau IP qui est connecté à plusieurs opérateurs, afin d'assurer son indépendance et pour faire face à un éventuel problème (technique ou commercial) chez l'un des opérateurs. La méthode officielle pour cela est d'acquérir des adresses indépendantes du fournisseur et de faire router ces adresses par les opérateurs auxquels on est connectés (par exemple en leur parlant en BGP). Mais avoir de ces adresses PI ("*Provider-Independent*") est souvent difficile. Quand on est passionnée, on y arrive <https://blog.ataxya.net/un-as-chez-soi-cest-possible/> mais ce n'est pas vraiment accessible à tout le monde. On se retrouve donc en général avec des adresses dépendant du fournisseur. C'est par exemple le cas de loin le plus courant pour le particulier, la petite association ou la PME.

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc8966.txt`

Dans ce cas, le "*multihoming*" est compliqué. Mettons que les deux fournisseurs d'accès se nomment A et B et que chacun nous fournisse un préfixe d'adresses IP. Si une machine interne a deux adresses IP, chacune tirée d'un des préfixes, et qu'elle envoie vers l'extérieur un paquet ayant comme adresse source une adresse de B, que se passe-t-il ? Si le paquet est envoyé via le fournisseur B, tout va bien. Mais s'il est envoyé via le fournisseur A, il sera probablement jeté par le premier routeur de A, puisque son adresse IP source ne sera pas une adresse du fournisseur (RFC 3704 et RFC 8704). Pour éviter cela, il faudrait pouvoir router selon l'adresse IP source, envoyant les paquets d'adresse source A vers le fournisseur A et ceux ayant une adresse source B vers B. En gros, soit l'information de routage influe sur le choix de l'adresse source, soit c'est le choix de l'adresse source qui influe sur le routage. On ne peut pas faire les deux, ou alors on risque une boucle de rétroaction. Ce RFC choisit la deuxième solution.

La première solution serait que la machine émettrice choisisse l'adresse IP source selon que les adresses de destination soient routées vers A ou vers B. Le RFC 3484 explique comment le faire en séquentiel et le RFC 8305 le fait en parallèle (algorithme dit des « globes oculaires heureux »). Mais les systèmes actuels ne savent typiquement pas le faire et, de toute façon, cela ne réglerait pas le cas où le routage change pendant la vie d'une connexion TCP. TCP, contrairement à QUIC <https://www.bortzmeyer.org/quic.html> ou SCTP, ne supporte pas qu'une adresse IP change en cours de route, sauf à utiliser les extensions du RFC 8684. Une dernière solution serait de laisser les applications gérer cela (cf. RFC 8445) mais on ne peut pas compter que toutes les applications le fassent. Bref, le routage tenant compte de la source reste souvent la meilleure solution.

À part le cas du "*multihoming*", d'autres usages peuvent tirer profit d'un routage selon la source. Ce routage peut simplifier la configuration des tunnels. Il peut aussi être utile dans le cas de l'"*anycast*". Lorsqu'on utilise du routage classique, fondé uniquement sur la destination, il est difficile de prédire quels clients du groupe "*anycast*" seront servis par quelle instance du groupe. Cela rend la répartition de charge compliquée (il faut jouer avec les paramètres des annonces BGP, par exemple) et cela pose des problèmes aux protocoles avec état, comme TCP, où un même client doit rester sur la même instance du groupe "*anycast*". (Des utilisations de l'"*anycast*" pour des services sans état, comme le DNS sur UDP, n'ont pas ce problème.) Au contraire, avec du routage fait selon la source, chaque instance "*anycast*" peut annoncer sa route avec le ou les préfixes IP qu'on veut que cette instance serve.

D'ailleurs, si vous voulez en savoir plus sur le routage selon la source (ou SADR "*Source-Address Dependent Routing*", également appelé SSR pour "*Source-Specific Routing*"), vous pouvez lire l'article détaillé des auteurs du RFC, « "*Source-Specific Routing*" <http://arxiv.org/pdf/1403.0445> ». On peut aussi noter que ce routage dépendant de la source n'est pas la même chose qu'un routage décidé par la source ("*source routing*"), qui indiquerait une liste de routeurs par où passer. Ici, chaque routeur reste maitre du saut suivant (sur le routage décidé par la source, voir RFC 8354, et un exemple dans le RFC 6554).

Bon, le routage tenant compte de la source est intéressant, OK. Mais, avant de s'y lancer, il faut faire attention à un petit problème, la spécificité des routes. Lorsqu'on route uniquement selon la destination, et que plusieurs routes sont possibles, la règle classique d'IP est celle de la spécificité. La route avec le préfixe le plus spécifique (le plus long) gagne. Mais que se passe t-il si on ajoute l'adresse source comme critère ? Imaginons qu'on ait deux routes, une pour la destination `2001:db8:0:1::/64` et s'appliquant à toutes les sources et une autre route pour toutes les destinations (route par défaut), ne s'appliquant qu'à la source `2001:db8:0:2::/64`. On va noter ces deux routes sous forme d'un tuple (destination, source) donc, ici, (`2001:db8:0:1::/64`, `::/0`) et (`::/0`, `2001:db8:0:2::/64`). Pour un paquet venant de `2001:db8:0:2::1` et allant en `2001:db8:0:1::1`, quelle est la route la plus spécifique ? Si on considère la spécificité de la destination d'abord, la route choisie sera la (`2001:db8:0:1::/64`, `::/0`). Mais si on regarde la spécificité de la source en premier, ce sera l'autre route, la (`::/0`, `2001:db8:0:2::/64`) qui sera adoptée. Doit-on laisser chaque machine libre de faire du destination d'abord ou du source d'abord ? Non, car tous les routeurs du domaine doivent suivre la même règle, autrement des boucles pourraient se former, un paquet faisant du ping-pong éternel entre un routeur « destination d'abord » et

un routeur « source d'abord ». Babel impose donc une règle : pour déterminer la spécificité d'une route, on regarde la destination d'abord. Le choix n'est pas arbitraire, il correspond à des topologies réseau typique, avec des réseaux locaux pour lesquelles on a une route spécifique, et une route par défaut pour tout le reste.

Un autre point important est celui du système de transmission des paquets. Il faut en effet rappeler que le terme « routage » est souvent utilisé pour désigner deux choses très différentes. La première est le routage à proprement parler ("*routing*" en anglais), c'est-à-dire la construction des tables de transmission, soit statiquement, soit avec des protocoles de routage comme Babel. La seconde est la transmission effective des paquets qui passent par le routeur ("*forwarding*" en anglais). Si on prend un routeur qui tourne sur Unix avec le logiciel babeld <https://github.com/jech/babeld>, ce dernier assure bien le routage au sens strict (construire les tables de routage grâce aux messages échangés avec les autres routeurs) mais c'est le noyau Unix qui s'occupe de la transmission. Donc, si on veut ajouter des fonctions de routage tenant compte de la source, il ne suffit pas de le faire dans le programme qui fait du Babel, il faut aussi s'assurer que le système de transmission en est capable, et avec la bonne sémantique (notamment l'utilisation de la destination d'abord pour trouver la route la plus spécifique). Si Babel, avec les extensions de ce RFC, tourne sur un système qui ne permet pas cela, il faut se résigner à ignorer les annonces spécifiques à une source. (Pour des systèmes comme Linux, qui permet le routage selon la source mais avec une mauvaise sémantique, puisqu'il utilise la source d'abord pour trouver la route la plus spécifique, la section V-B de l'article des auteurs cité plus haut fournit un algorithme permettant de créer des tables de transmission correctes.)

Le passage du routage classique au routage tenant compte de la source nécessite de modifier les structures de données du protocole de routage, ici Babel (section 3 du RFC). Les différentes structures de données utilisées par une mise en œuvre de Babel doivent être modifiées pour ajouter le préfixe de la source, et plusieurs des TLV dans les messages que s'échangent les routeurs Babel doivent être étendus avec un sous-TLV (RFC 8966, section 4.4) qui contient un préfixe source. La section 5 du RFC détaille les modifications nécessaires du protocole Babel. Les trois types de TLV qui indiquent un préfixe de destination, "*Updates*", "*Route Requests*" et "*Seqno*", doivent désormais transporter en plus un sous-TLV indiquant le préfixe source, marqué comme obligatoire (RFC 8966, section 4.4), de façon à ce qu'une version de Babel ne connaissant pas le routage selon la source n'accepte pas des annonces incomplètes. Si une annonce ne dépend pas de l'adresse IP source, il ne faut pas envoyer un sous-TLV avec le préfixe `::/0` mais au contraire ne pas mettre de sous-TLV (les autres versions de Babel accepteront alors l'annonce).

Ainsi, des versions de Babel gérant les extensions de notre RFC 9079 et d'autres qui ne les gèrent pas (fidèles au RFC 8966) pourront coexister sur le même réseau, sans que des boucles de routage ne se forment. En revanche, comme la notion de sous-TLV obligatoire n'est apparue qu'avec le RFC 8966, les vieilles versions de Babel, qui suivent l'ancien RFC 6126, ne doivent pas être sur le même réseau que les routeurs à routage dépendant de la source.

Le nouveau sous-TLV "*Source Prefix*" a le type 128 <https://www.iana.org/assignments/babel/babel.xml#sub-tlv-types> et sa valeur comporte un préfixe d'adresses IP, encodé en {longueur, préfixe}.

Notez que si les routeurs qui font du routage selon la source n'annoncent que des routes ayant le sous-TLV indiquant un préfixe source, les routeurs qui ne gèrent pas ce routage selon la source, et qui ignorent donc cette annonce, souffriront de famine : il n'y aura pas de routes vers certaines destinations, conduisant le routeur à jeter les paquets. Il peut donc être prudent, par exemple, d'avoir une route par défaut sans condition liée à la source, pour ces routeurs qui ignorent cette extension.

Question sécurité, il faut signaler (section 9 du RFC), que cette extension au routage apporte davantage de souplesse, et que cela peut nécessiter une révision des règles de sécurité, si celles-ci supposaient

que tous les paquets vers une destination donnée suivaient le même chemin. Ainsi, le filtrage des routes (RFC 8966, annexe C) pourra être inutile si des routes spécifiques à une source sont présentes. Autre supposition qui peut être désormais fausse : une route spécifique à une machine de destination ("*host route*", c'est-à-dire un préfixe /128) n'est plus forcément obligatoire, une autre route pour certaines sources peut prendre le dessus.

Au moins deux mises en œuvre de cette extension existent, dans babeld <https://github.com/jech/babeld/> depuis la version 1.6 (en IPv6 seulement), et dans BIRD depuis la 2.0.2.

Merci à Juliusz Chroboczek pour sa relecture, et pour ses bonnes remarques sur le choix entre le routage selon la source et les autres solutions.