RFC 7050 : Discovery of the IPv6 Prefix Used for IPv6 Address Synthesis

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 5 novembre 2013

Date de publication du RFC : Novembre 2013

https://www.bortzmeyer.org/7050.html

Lorsqu'on utilise le mécanisme NAT64 du RFC 6146 [1] pour donner accès à l'Internet historique IPv4 depuis des machines IPv6, seul le traducteur NAT64 connait le préfixe IPv6 utilisé pour la traduction. Les machines ordinaires du réseau local ne le savent pas. Ce nouveau RFC fournit un moyen standard de découvrir ce préfixe.

Un tout petit rappel sur NAT64 (RFC 6146) et son copain DNS64 (RFC 6147) : leur but est de fournir une connectivité IPv4 (par exemple pour accéder à des machines qui n'ont toujours pas IPv6) aux réseaux modernes qui seront entièrement en IPv6 (RFC 6144). Pour cela, le serveur DNS64 « ment » en répondant aux requêtes de type AAAA (demande d'une adresse IPv6) par une réponse synthétisée, lorsque le nom demandé n'a qu'un A (adresse IPv4). Cette réponse synthétique utilise un préfixe configuré à la fois dans le serveur DNS64 et dans le traducteur NAT64 qui, voyant une adresse portant ce préfixe dans le champ Destination, va traduire l'IPv6 en IPv4 (et réciproquement au retour).

La plupart du temps, les machines IPv6 situées sur le réseau local n'auront aucun besoin de savoir ce qui se passe. Pour elles, tous les services externes sont accessible en IPv6 et elles ne connaissent pas la magie sous-jacente. Bien sûr, en regardant les adresses IPv6 obtenues, elles pourront s'étonner de voir que tant d'entre elles commencent par le même préfixe, mais qu'elles le fassent ou pas ne change rien. NAT64 est prévu pour fonctionner entièrement dans le routeur d'accès, sans participation (et donc sans mise à jour du logiciel) des machines terminales.

Sauf qu'il y a des cas où il serait utile que ces machines terminales bossent un peu. Par exemple, DNS64 ne sert à rien si l'application n'utilise pas le DNS. Si on a un URL `http://192.168.0.33/`,

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc6146.txt`

qui est légal (quoique déconseillé) et devrait marcher, DNS64 ne sera jamais appelé et NAT64 échouera donc. Pourtant, cet URL pourrait fonctionner à travers NAT64 si seulement la machine terminale faisait le travail de DNS64 en synthétisant l'adresse IPv6 correspondant à `192.168.0.33`. Un problème analogue avec DNS64 se pose si la machine terminale fait la validation DNSSEC elle-même (ce qui est souvent une bonne idée <https://www.bortzmeyer.org/ou-valider-dnsssec.html>). Dans ce cas, les réponses « mensongères » du serveur DNS64 seront refusées. Dans ces deux cas, on souhaite que la machine terminale synthétise une adresse IPv6 elle-même et, pour cela, elle doit connaître le préfixe qui permettra au routeur NAT64 de savoir ce qu'il doit faire.

Ce préfixe « magique » (les adresses utilisant tout autre préfixe seront traitées par le routeur comme de l'IPv6 ordinaire) peut être de deux types :

— Un préfixe bien connu, réservé à cet usage, le WKP ("*Well-Known Prefix*") qui vaut `64:ff9b::/96`. Il est décrit dans le RFC 6052.
— Ou bien un préfixe décidé localement, un NSP ("*Network-Specific Prefix*"), `2001:db8:1:64::/96` dans les exemples suivants.

Dans les deux cas, ce préfixe doit être configuré à l'identique dans le routeur NAT64 et dans le serveur DNS64. Et, si on utilise la synthèse locale (locale à la machine terminale) des adresses IPv6, il ,doit aussi être connu des machines terminales, ce qui est le but de ce RFC. Attention, il peut y avoir non pas un, mais plusieurs préfixes utilisés simultanément.

La technique utilisée dépend entre autres d'un nom bien connu, `ipv4only.arpa`, qui n'aura jamais **que** des adresses IPv4, et d'adresses IPv4 bien connues, les WKA ("*Well-Known Addresses*"), `192.0.0.170` et `192.0.0.171`.

```
% dig +short A ipv4only.arpa
192.0.0.170
192.0.0.171

% dig +short AAAA ipv4only.arpa
```

Le principe (section 3 de notre RFC) est de faire une requête DNS de type AAAA (adresse IPv6) pour ce nom `ipv4only.arpa`. Comme ce nom n'a que des enregistremments A (adresse IPv4), si on obtient des enregistrements AAAA, c'est qu'il y a du DNS64 sur le trajet, qui synthétise ces enregistrements. Sinon, il n'y a pas de service DNS64. (La requête doit se faire avec le bit CD - "*Checking Disabled*" - à 0, autrement le serveur DNS64 ne fait pas la synthèse.) À la place d'un serveur DNS64, il peut aussi y avoir un serveur menteur qui répond même en l'absence de données (cela est fréquent sur les portails captifs). Dans ce cas, le client doit aussi faire une requête pour un nom qui n'existe pas (il n'est pas si facile que cela <https://www.bortzmeyer.org/noms-inexistants.html> d'en trouver un) et vérifier qu'il récupère bien `NXDOMAIN`.

Une fois reçue sa réponse, la machine doit examiner tous ces AAAA et en déduire le (ou les) préfixe(s) utilisé(s) pour la synthèse. Si le préfixe est le WKP, c'est facile. Si c'est un NSP, c'est un peu plus dur. C'est là que les WKA sont utilisées : comme la machine connait les adresses IPv4 originales, elle peut les retrouver dans les adresses IPv6. Avec les examples plus haut, la machine fait une requête AAAA pour `ipv4only.arpa`, obtient comme réponse `2001:db8:1:64::192.0.0.170` (qu'on peut également écrire `2001:db8:1:64::c000:aa`) et en déduit que le préfixe utilisé est `2001:db8:1:64::/96`. Par exemple, avec BIND, et ce fichier de configuration :

```
options {
        ...
        dns64 2001:db8:1:64::/96 { // Network-Specific Prefix
              clients { me; };
        };
```

On obtiendra :

```
% dig +nodnssec AAAA ipv4only.arpa
...
;; ANSWER SECTION:
ipv4only.arpa. 3485 IN AAAA 2001:db8:1:64::c000:ab
ipv4only.arpa. 3485 IN AAAA 2001:db8:1:64::c000:aa
```

Si cela ne marche pas (par exemple si on ne trouve pas les WKA comme `192.0.0.170` dans la réponse), alors la recherche du préfixe a échoué (format d'adresses inhabituel ou un truc comme ça) et on doit laisser tomber et donc ne pas faire de synthèse d'adresses IPv6 sur la machine cliente. La procédure de ce RFC ne prétend pas marcher dans tous les cas.

Au fait, pourquoi deux adresses WKA, `192.0.0.170` et `192.0.0.171` ? L'annexe B du RFC discute ce choix, dû au désir de limiter les faux positifs (par exemple si la chaîne de bits qui compose une des deux adresses apparait également dans le préfixe NAT64.)

Notons que, si le canal entre le client et le serveur DNS64 n'est pas protégé, un attaquant peut facilement informer le client avec un mauvais préfixe. On peut (sauf pour le WKP) valider l'information avec DNSSEC (je ne détaille pas ici, voir la section 3.1 du RFC).

C'est bien joli d'avoir appris le préfixe mais rappelez-vous que ce RFC propose essentiellement une **heuristique** : il ne donne pas de garanties. Il faut donc tester le préfixe qu'on vient d'obtenir. Après tout, des tas de choses peuvent déconner. La machine cliente peut faire les tests qu'elle veut (viser des amers publics <`https://www.bortzmeyer.org/que-pinguer.html`>). Mais le RFC suggère une procédure que le FAI qui a déployé NAT64 peut mettre en place. Le FAI doit configurer une machine de test (qui répond aux paquets ICMP `Echo` et n'a pas de limitation de trafic) et mettre deux informations dans le DNS. La machine finale fait une requête DNS de type PTR pour une adresse WKA (`192.0.0.170` ou `192.0.0.171`) représentée en IPv6 et traduite au format `ip6.arpa`. Puis elle fait une requête de type A sur le nom obtenu et cela donne l'adresse de la machine de test du FAI, si celui-ci a suivi les recommandations de notre RFC. Avec les exemples de préfixes plus haut, on utilisera l'adresse `2001:db8:1:64::192.0.0.170`, la requête PTR portera sur `a.a.0.0.0.0.0.c.0.0.0.0.0.0.0.0.4.6.0.0.1.0.` et, si elle renvoie le nom `ping.example.net`, la requête A portera sur ce nom. Mettons qu'on obtienne `192.0.2.33`, on synthétisera `2001:db8:1:64::192.0.2.33` et on verra bien si ça marche. (L'annexe A du RFC contient un exemple complet de fichier de zone DNS standard utilisant cette technique.)

Par contre, les clients ne doivent pas faire de tests de connectivité avec les adresses obtenues en interrogeant `ipv4only.arpa`. (Elles ne sont pas censées répondre.)

Notre RFC rappelle aussi que cette technique ne change rien au fait que NAT64 est fondamentalement un bricolage provisoire, que le résultat est « mieux que rien » mais que la bonne solution est évidemment le passage généralisé à IPv6.

Ah, et, pendant qu'on parle de ce que configure localement le FAI, le RFC n'interdit pas des déploiements de NAT64 où les clients utilisent un autre nom que `ipv4only.arpa`, par exemple si le FAI veut ne dépendre que de ses propres noms (`ipv4only.example.net`).

Les sections 4 et 5 donnent quelques conseils pratiques pour le déploiement de l'infrastructure nécessaire. Ainsi, le domaine `ipv4only.arpa` devra avoir un long TTL, au moins une heure, pour bénéficier des caches. Il doit être signé avec DNSSEC.

Comme pour toutes les techniques de transition (ici, d'IPv4 vers IPv6), l'IETF impose une description d'une stratégie de sortie. Comment fera t-on lorsque NAT64 et DNS64 ne seront plus nécessaires ? La section 6 demande que les machines terminales qui ont la possibilité de découvrir le préfixe NAT64, et de synthétiser elles-mêmes les adresses IPv6, aient un mécanisme pour couper cette possibilité, le jour où elle sera abandonnée.

Enfin, un peu de bureaucratie IANA en section 8. Le domaine « spécial » `ipv4only.arpa` a été enregistré selon les règles des RFC 3172, et RFC 6761, règles qui n'avaient pas vraiment été respectées, ce qui a nécessité une correction dans le RFC 8880. Le domaine a été placé dans le registre des noms spéciaux <https://www.iana.org/assignments/special-use-domain-names/special-use-domain-r
xml#special-use-domain>. Les adresses WKA, elles, ont été enregistrées selon les règles des RFC 5736 (qui gère `192.0.0.0/24`) et RFC 6890. Elles sont donc désormais dans le registre des adresses spéciales <https://www.iana.org/assignments/iana-ipv4-special-registry/iana-ipv4-special-
xhtml#iana-ipv4-special-registry-1>.

Il existe au moins une mise en œuvre de NAT64 <https://sites.google.com/site/tmoipv6/464xlat#TOC-Android-CLAT-on-a-UMTS-IPv6-only-network-with-DNS64-NAT64> qui inclus la technique de découverte de préfixe de ce RFC.