RFC 6742 : DNS Resource Records for ILNP

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 10 novembre 2012

Date de publication du RFC : Novembre 2012

http://www.bortzmeyer.org/6742.html

Le système ILNP, qui vise à fournir une nouvelle (et meilleure) architecture à l'Internet, dépend du DNS pour plusieurs fonctions critiques. Plusieurs nouveaux types d'enregistrement DNS sont nécessaires et ce RFC les décrit.

Avant de lire ce RFC, il faut connaître ILNP. Le mieux est de commencer par le RFC 6740 [1], qui décrit tout le système. Le RFC 6741 fournit ensuite quelques détails pratiques. Il faut aussi naturellement connaître les bases du DNS (RFC 1034 et RFC 1035).

C'est fait ? Vous avez lu tout cela ? Alors, place à ce nouveau RFC, le RFC 6742. ILNP utilise le DNS essentiellement pour permettre un **rendez-vous** entre deux machines. Celle qui veut commencer connaît le nom de l'autre mais aucune information pratique sur comment la rencontrer. Quatre types d'enregistrements sont utilisés pour cela :

— `NID` qui permet de trouver l'**identificateur** d'une machine à partir de son nom,
— `L64` et `L32` qui permettent de trouver le **localisateur** d'une machine à partir de son nom,
— Et `LP`, une optimisation facultative qui sera présentée plus loin.

Commençons par le `NID`. Un de ses rôles est d'indiquer à la machine qui veut en contacter une autre si sa future correspondante connait ILNP (ou bien si elle est uniquement en IP Classic). Sa présence dans le DNS est donc un signal : « tu peux y aller, elle parle ILNP ». Un enregistrement erroné (disant qu'une machine a un `NID` alors qu'en fait elle ne parle pas ILNP) n'est pas fatal mais ralentirait sérieusement l'établissement de la connexion.

Le `NID` a reçu le numéro 104 (cf. le registre IANA <https://www.iana.org/assignments/dns-parameters>) et il comporte deux champs, une préférence et l'identificateur. Attention, ce sont les préférences les plus faibles qui sont celles à choisir en premier. Par exemple, voici trois `NID` pour `host1.example.com` :

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc6740.txt`

```
host1.example.com. IN NID 10 0014:4fff:ff20:ee64
                   IN NID 20 0015:5fff:ff21:ee65
                   IN NID 50 0016:6fff:ff22:ee66
```

L'identificateur `0014:4fff:ff20:ee64` est le préféré. (La syntaxe des identificateurs et des localisateurs est décrite dans le RFC 6740.)

Si un serveur DNS « connaît ILNP », c'est-à-dire qu'il a été programmé avec du code spécifique pour ILNP, au lieu d'attribuer aux enregistrements ILNP le traitement par défaut, il peut optimiser, par exemple en envoyant des enregistrements non demandées mais qui seront probablement utiles. Cette optimisation, **qui n'a rien d'obligatoire**, est analogue à ce que font les serveurs actuels pour une requête MX : si la réponse est `mail.example.com`, ils incluent également les adresses IP de `mail.example.com` dans la réponse, conscients qu'elles seront probablement nécessaires bientôt (si on demande le MX de `example.com`, c'est en général pour lui envoyer du courrier juste après). Ici, un serveur « connaissant ILNP » à qui on demande le `NID` de `host1.example.com` peut aussi mettre dans le réponse des `L64` et des `L32`, non demandés mais qui seront probablement utiles juste après.

Le type `L64` stocke des localisateurs pour ILNPv6, c'est_à-dire ILNP tournant sur IPv6. La présence de `L64` dans le DNS indique que la machine peut faire de l'ILNP au-dessus d'IPv6. Pour ce protocole, les localisateurs font 64 bits, d'où le nom de ce type. Son numéro est 106. Lui aussi a une préférence, suivie par le localisateur proprement dit.

Voici un exemple où la machine `host1.example.com` a deux localisateurs, avec une préférence pour `2001:0db8:1140:1000` :

```
host1.example.com. IN L64 10 2001:0db8:1140:1000
                   IN L64 20 2001:0db8:2140:2000
```

Rappelez-vous que les localisateurs d'ILNPv6 ont une syntaxe qui ressemble à celle des adresses IPv6 mais il s'agit bien d'un type différent.

Quant au type `L32`, il stocke des localisateurs pour ILNPv4, c'est-à-dire l'incarnation d'ILNP pour IPv4. Leur présence dans le DNS indique que la machine peut faire de l'ILNP au-dessus d'IPv4. Pour ce protocole, les localisateurs font 32 bits, d'où le nom de ce type. Son numéro est 105. Il comprend une préférence, puis le localisateur proprement dit.

Voici un exemple où la machine `host1.example.com` a trois localisateurs, `10.1.2.0` étant le préféré (la machine est sans doute "*multi-homée*") :

```
host1.example.com. IN L32 10 10.1.2.0
                   IN L32 20 10.1.4.0
                   IN L32 30 10.1.8.0
```

Le dernier type d'enregistrement DNS, `LP`, est un peu différent. Il s'agit de fournir un niveau d'indirection. En effet, si une machine change de localisateurs (ce qui est normal, en ILNP), et veut pouvoir être jointe, la méthode est de mettre à jour le DNS. Or, la zone où est déclarée la machine peut ne pas être facilement accessible en écriture (RFC 2136 ou méthode équivalente). L'idée (voir RFC 6740) est donc de pointer, via un enregistrement `LP`, vers une zone prévue pour les mises à jour rapides, où se trouveront les localisateurs.

L'enregistrement LP a le numéro 107 et comprend une préférence et un FQDN. Ce FQDN aura à son tour des `L32` ou des `L64`. Voici un exemple :

```
host1.example.com. IN LP 10 subnet1.dynamic.example.net.
                   IN LP 20 subnet3.backup.example.com.

; Puis, dans la zone dynamic.example.net :

subnet1.dynamic.example.net. IN L64 50 2001:0db8:1140:1000
                             IN L64 100 2001:0db8:2140:2000
```

Avec cet exemple, si `host1.example.com` change de localisateur, on ne touche pas à la zone `example.com`, on ne modifie que `dynamic.example.net`. Cette indirection via les `LP` n'est pas indispensable mais elle aidera dans bien des cas, lorsque la zone « principale » ne permettra pas les mises à jour rapides. Autre avantage, toutes les machines d'un même sous-réseau ayant le même localisateur, leurs noms pourront pointer vers le même `LP` et une seule mise à jour suffira pour tout le monde.

La section 3 synthétise ensuite l'usage de ces types d'enregistrement. Le minimum pour une machine ILNP est un `NID` + un `L64` (ou `L32` en ILNPv4) :

```
host1.example.com. IN NID 10 0014:4fff:ff20:ee64
                   IN L64 10 2001:0db8:1140:1000
```

Si la machine est "*multi-homée*", on aura plusieurs `L64` :

```
host1.example.com. IN NID 10 0014:4fff:ff20:ee64
                   IN L64 10 2001:0db8:1140:1000
                   IN L64 20 2001:0db8:2140:2000
```

Si la machine est mobile, elle aura souvent intérêt à utiliser les `LP` et cela donnera quelque chose du genre :

```
host1.example.com. IN NID 10 0014:4fff:ff20:ee64
                   IN LP  10 mobile-1.example.net.

; ...

mobile-1.example.net. IN L64 2001:0db8:8140:8000
```

Rappelons que ces enregistrements ne doivent apparaître que si la machine parle ILNP. Un de leurs rôles est de signaler qu'ILNP est disponible.

Un client ILNP qui veut récupérer ces informations ne doit pas compter qu'elles soient toutes envoyées d'un coup (même si c'est une optimisation recommandée). Au passage, rappelez-vous que la requête `ANY`, envoyée à un résolveur, n'a pas forcément la sémantique attendue (elle renvoie ce que le résolveur a dans son cache, pas forcément toutes les données disponibles sur les serveurs faisant autorité). D'autre part, même si la réponse du serveur faisant autorité incluait tous les enregistrements, certains résolveurs peuvent, pour des raisons de sécurité, en jeter certains. Enfin, même si les données sont gardées dans le cache, elles expirent au bout d'un moment et les TTL ne sont pas forcément les mêmes pour tous les types d'enregistrement. Le client ILNP doit donc se préparer à faire plusieurs requêtes, par exemple `NID` (s'il n'y a pas de résultat, la machine distante ne parle pas ILNP) puis `LP` (qui est facultatif), puis `L64` puis enfin, s'il n'y a pas de `L64`, se résigner à `L32`.

À propos des TTL, notez qu'il est logique que les TTL soient assez longs pour les `NID` (une information normalement très stable, je dirais que plusieurs jours ne sont pas un problème). En revanche, les `L32` et `L64` peuvent avoir des TTL bien plus courts (le RFC ne donne pas de chiffre mais, disons, moins d'une minute), surtout si la machine ou le réseau est mobile. On peut alors s'inquiéter « est-ce que ces courts TTL, empêchant une mise en cache sérieuse, ne vont pas augmenter excessivement la charge sur le système DNS, et ralentir les requêtes ? ». Mais plusieurs articles (« "*Reducing DNS Caching*" <http://www.dcs.gla.ac.uk/conferences/gi2011/p803-bhatti.pdf> », « "*DNS performance and the effectiveness of caching*" <http://nms.lcs.mit.edu/papers/dns-ton2002.pdf> ») ont montré que les caches du DNS étaient surtout rentables pour l'information d'infrastructure (enregistrements `NS` et colle). Pour les enregistrements « terminaux » comme les adresses IP ou les localisateurs, les caches sont moins nécessaires.

Un petit mot sur la sécurité, maintenant. Comme les informations nécessaires sont vraiment critiques (un `L64` erroné et tout le trafic va à une autre machine), la section 4 recommande d'utiliser DNSSEC (RFC 4033) pour protéger ses informations. (À l'été 2012, le déploiement de DNSSEC est très avancé, alors que celui d'ILNP est inexistant. Il est donc raisonnable de compter que DNSSEC sera disponible.) Lors des mises à jour dynamiques, il faut utiliser les sécurités décrites dans le RFC 3007. Par exemple, avec le serveur de noms Yadifa, on peut metttre à jour dynamiquement une zone signée. On copie la clé privée (par exemple, pour une clé générée avec `dnssec-keygen` pour la zone `secure.example`, le fichier de la clé privée peut être `Ksecure.example.+005+05461.private`) dans le répertoire indiqué par la directive `keyspath` de Yadifa et c'est tout. Les mises à jour dynamiques seront faites et automatiquement signées. Le journal de Yadifa dira :

```
2012-09-25 20:54:12.754705 | server   | I | update (831e) secure.example. SOA (127.0.0.1:44480)
2012-09-25 20:54:12.754749 | server   | D | database: update: checking DNSKEY availability
2012-09-25 20:54:12.754778 | server   | D | database: update: processing 0 prerequisites
2012-09-25 20:54:12.754791 | server   | D | database: update: dryrun of 2 updates
2012-09-25 20:54:12.754828 | server   | D | database: update: opening journal page
2012-09-25 20:54:12.755008 | server   | D | database: update: run of 2 updates
2012-09-25 20:54:13.000526 | server   | D | database: update: closed journal page
```

À la date de publication de ce RFC, il n'existe pas encore de logiciel DNS qui gère ILNP.

Pour finir, quelques articles sur l'idée d'utiliser le DNS pour localiser une machine mobile : « "*An End-To-End Approach To Host Mobility*" <http://nms.lcs.mit.edu/papers/e2emobility.pdf> », « "*Reconsidering Internet Mobility*" <http://nms.csail.mit.edu/papers/migrate-hotOS.html> » et « "*Mobile Host Location Tracking through DNS*" <http://www.ee.ucl.ac.uk/lcs/previous/LCS2002/LCS072.pdf> ».