RFC 5507 : Design Choices When Expanding DNS

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 23 avril 2009

Date de publication du RFC : Avril 2009

https://www.bortzmeyer.org/5507.html

Le DNS est au cœur de la plupart des transactions sur l'Internet. Bien connu, très fiable et largement déployé, il est un service très tentant, lorsqu'on veut déployer un nouveau service d'information sur l'Internet. Bien des protocoles ont ainsi délégué la récupérations de données au DNS. Parfois, ils ne l'ont pas fait de manière optimum et ce RFC de l'IAB veut « redresser la barre » en précisant les bonnes (et les mauvaises) façons de stocker et de récupérer de l'information dans le DNS.

Dès le résumé, notre RFC pointe le problème le plus fréquent : abuser des enregistrements de type TXT (texte), le type le plus tentant pour distribuer de l'information. Par exemple, SPF à l'origine utilisait exclusivement ce type pour distribuer des politiques d'autorisation de MTA, exprimées en un mini-langage (la version finalement normalisée, dans le RFC 4408 [1], a créé un type spécifique, nommé SPF mais, en pratique, le TXT reste le plus utilisé, le SPF ayant même été abandonné dans le RFC 7208). La principale conclusion du RFC est donc qu'il vaut mieux, en général, créer un nouveau type d'enregistrement <https://www.iana.org/assignments/dns-parameters>.

À l'origine, le DNS stockait surtout deux sortes de données : celles liées à l'infrastructure DNS elle-même comme les enregistrements SOA ou NS et celles liées à la résolution de noms en adresses IP (enregistrement A) ou inversement (PTR). Le succès du DNS, sa disponibilité universelle, ont mené à bien d'autres utilisations, soit spécifiques à une application comme les enregistrement MX, soit génériques comme les enregistrements SRV du RFC 2782. Beaucoup d'autres, plus récents, s'y sont ajoutés comme les SSHFP du RFC 4255 ou les HIP du RFC 8005. (Les sections 1 et 2 du RFC rappellent les bases du DNS, nécessaires pour comprendre la question.)

Quels sont les moyens disponibles aujourd'hui pour « étendre » le DNS, pour stocker de nouveaux types de données dans cette base ? La section 3 les liste successivement.

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc4408.txt

Il y a la possibilité (section 3.1) de stocker les données dans un enregistrement générique comme TXT ou le défunt SINK ou NAPTR et d'ajouter dans ses données un **sélecteur** qui indique à l'application si ce sont bien ces données. SPF fonctionne ainsi, la présence de la chaîne `v=spf1` au début de l'enregistrement TXT indiquant qu'il s'agit bien de SPF. Les utilisateurs des NAPTR (RFC 6116) font souvent pareil. Cette méthode (nommée "*subtyping*") a un gros inconvénient. Comme les requêtes DNS se font par **clé** (la question posée, le QNAME) et pas par valeur, il n'y a aucun moyen de ne récupérer que les enregistrements intéressants. On les obtient tous et on trie après. Ainsi, si je veux les enregistrements SPF de `sources.org`, je dois récupérer deux enregistrements :

```
% dig TXT sources.org
...
sources.org. 86400 IN TXT "v=spf1 mx a:uucp.bortzmeyer.org a:central.sources.org ?all"
sources.org. 86400 IN TXT "Sources"
...
```

et trier ensuite. Cela peut poser des problèmes si le nombre d'enregistrements de ce type (le "*RRset*") est gros.

La section 3.2 propose une autre méthode : ajouter un préfixe au nom de domaine (le QNAME). C'est, par exemple, ce que fait DKIM (RFC 6376) qui utilise des enregistrements TXT mais des noms de domaine qui lui sont propres. Si je reçois un message signé par DKIM qui indique le domaine `example.net` et le sélecteur `beta`, je fais une requête DNS pour le type TXT et le nom `beta._domainkey.example.net`. XMPP a un mécanisme analogue.

La principale limite de cette méthode vient des **jokers** ("*wildcards*"). Les jokers ne peuvent apparaitre que tout à fait à gauche d'un nom de domaine. Donc, si on a un joker dans `example.net`, il ne sera pas facile d'en faire un pour `_prefixe.*.example.net` (cf. RFC 4592). Cela concerne particulièrement le courrier électronique car c'est la principale utilisation des jokers.

Continuons les méthodes possibles. Puisque le préfixe a des limites, pourquoi pas un suffixe ? (Section 3.3.) Le principal problème est alors de s'assurer que `example.net` et `example.net._suffixe` sont synchrones, alors qu'ils sont désormais dans deux sous-arbres du DNS complètement distincts.

Plus astucieuse est l'idée de jouer sur la **classe**. La question DNS ne comporte pas uniquement un nom de domaine (QNAME) mais aussi un type (QTYPE) et une classe (QCLASS). On a tendance à oublier cette dernière car elle vaut quasiment toujours IN (pour INternet) mais d'autres classes sont possibles et certaines sont déjà enregistrées <https://www.iana.org/assignments/dns-parameters>. Mais, si certaines applications comme Hesiod (j'avais déployé Hesiod dans les salles de classe du CNAM il y a longtemps...) utilisaient ces autres classes, il parait difficile aujourd'hui de s'en servir, beaucoup de logiciels, de serveurs intermédiaires et de pare-feux n'acceptant pas les classes autres que IN. Et, du point de vue administratif, cela représente une nouvelle arborescence à mettre en place, puisque l'arbre du DNS dans une autre classe ne correspond pas forcément à celui de IN.

Alors, un nouveau type ? Je trahis le suspense mais c'est en effet la solution privilégiée par l'IAB. La section 3.5 explique les avantages de ce mécanisme mais aussi ses limites. Pour insérer des données du nouveau type, et pour pouvoir les utiliser, il faut que beaucoup d'acteurs mettent à jour quelque chose :

— les auteurs de serveurs faisant autorité (serveur primaire, qui charge les données et qui doit donc pouvoir les lire, la section 6 revient sur ce problème de l'avitaillement), et serveurs secondaires, qui les copient ; historiquement, on a déjà vu des secondaires qui ne pouvaient pas recharger une zone qu'ils avaient transféré car elle contenait des types inconnus ; normalement, c'est devenu un problème très rare, cf. RFC 3597).

— les auteurs de résolveurs (normalement, ceux-ci doivent être transparents aux divers types, voir encore le RFC 3597).
— les auteurs d'applications, qui doivent pouvoir lire ces données (toutes les bibliothèques d'accès au DNS ne le permettent pas forcément, notamment dans le monde Microsoft) et les comprendre.

En plus de ces difficultés techniques, l'enregistrement d'un nouveau type utilisable dans le DNS a toujours été un processus très long et très complexe. L'IAB le reconnait dans ce RFC (après l'avoir nié pendant longtemps) mais ce n'est que depuis la sortie du RFC 5395 en novembre 2008 que le processus est devenu raisonnable. (Obtenir un nouveau type demeure un certain travail car l'espace disponible est sur 16 bits seulement, donc pas infini et il faut justifier sa demande.)

Maintenant que les différentes solutions ont été présentées, il faut fournir les éléments permettant de choisir. Avant cela, la section 4 fait un petit détour sur un problème sur lequel les débutants en DNS se cassent souvent le nez, le fait que les frontières des zones DNS soient invisibles aux applications. En effet, le découpage d'un nom de domaine en sous-domaines qui est, lui, visible (les éléments sont séparées par des points), n'indique pas où sont les frontières de zones, encore moins quelles sont les frontières administratives et politiques. Si je vois le nom `www.durand.free.fr`, est-ce un nom contrôlé par Free ou bien Free a t-il délégué à M. Durand, son client ? Rien dans le DNS ne permet de le savoir et une application qui se baserait sur cette connaissance aurait de grandes chances de se tromper. Ainsi, on a vu des applications, lorsqu'elles ne trouvaient pas un nom dans un domaine, chercher dans le domaine parent. C'est très dangereux, car rien n'indique si le domaine parent est sous le contrôle des mêmes personnes (ce mécanisme, dit de "*tree climbing*", fut la base de la faille de sécurité WPAD). La seule solution serait de connaitre la politique de **tous** les registres, y compris ceux qui ne sont pas des TLD comme `eu.org`, ce qui est évidemment irréaliste. Voir aussi le RFC 1535 sur ce point.

La section 5 synthétise tous ces points en recommandant fortement la création d'un nouveau type d'enregistrement DNS, lorsqu'on a besoin de stocker des données dans le DNS. Cette section exprime une certaine compréhension pour le développeur qui n'utilise pas le DNS pour le plaisir, mais parce qu'il a besoin de stocker ses données et d'y accéder, et qui n'est donc pas un expert DNS, et qui file donc droit vers la solution la plus évidente, les enregistrements TXT.

Mais cela ne change rien aux problèmes posés par les TXT notamment le manque de structure interne. Chaque application qui utilise les TXT doit donc définir son mini-langage, rendant ainsi difficile la cohabitation d'enregistrements TXT d'applications différentes. En pratique, le TXT est donc plutôt un enregistrement privé, dont la signification est laissé à chacun.

La cohabitation est rendue encore plus difficile par l'absence d'un sélecteur standard (le RFC 1464 avait essayé d'en définir un mais avait été un échec). Chaque application doit donc avoir son mécanisme pour « reconnaître ses petits » (comme le `v=spf1` de SPF).

Les enregistrements TXT avec un préfixe spécifique dans le nom de domaine, comme ceux de DKIM, sont meilleurs mais souffrent d'autres problèmes comme la difficile cohabitation avec les jokers.

Il reste donc, et la section 6 conclut le RFC sur ce point, à lire le RFC 5395 et à remplir un dossier pour avoir un nouveau type d'enregistrement. C'était très difficile autrefois, aussi bien à obtenir qu'à déployer ensuite, mais c'est désormais plus facile.