RFC 8901 : Multi-Signer DNSSEC Models

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 25 septembre 2020

Date de publication du RFC : Septembre 2020

https://www.bortzmeyer.org/8901.html

Aujourd'hui, il est courant de confier l'hébergement de ses serveurs DNS faisant autorité <https://www.bortzmeyer.org/serveur-dns-faisant-autorite.html> à un sous-traitant. Mais si vous avez, comme c'est recommandé, plusieurs sous-traitants et qu'en prime votre zone, comme c'est recommandé, est signée avec DNSSEC ? Comment s'assurer que tous les sous-traitants ont bien l'information nécessaire ? S'ils utilisent le protocole standard du DNS pour transférer la zone, tout va bien. Mais hélas beaucoup d'hébergeurs ne permettent pas l'utilisation de cette norme. Que faire dans ce cas ? Ce nouveau RFC explique les pistes menant à des solutions possibles.

Pourquoi est-ce que des hébergeurs DNS ne permettent pas d'utiliser la solution normalisée et correcte, les transferts de zone du RFC 5936 [1] ? Il peut y avoir de mauvaises raisons (la volonté d'enfermer l'utilisateur dans un silo, en lui rendant plus difficile d'aller voir la concurrence) mais aussi des (plus ou moins) bonnes raisons :

— Si l'hébergeur signe dynamiquement les données DNS,
— Si l'hébergeur produit dynamiquement des données DNS (ce qui implique de les signer en ligne),
— Si l'hébergeur utilise des trucs non-standards, par exemple pour mettre un CNAME à l'apex d'une zone <https://www.bortzmeyer.org/cname-apex.html>.

Dans ces cas, le transfert de zones classique n'est pas une solution, puisqu'il ne permet pas de transmettre, par exemple, les instructions pour la génération dynamique de données.

Résultat, bien des titulaires de noms de domaine se limitent donc à un seul hébergeur, ce qui réduit la robustesse de leur zone face aux pannes…ou face aux conflits commerciaux avec leur hébergeur.

La section 2 de notre RFC décrit les modèles possibles pour avoir à la fois DNSSEC **et** plusieurs hébergeurs. Si aucune des trois raisons citées plus haut ne s'applique, le cas est simple : un hébergeur

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc5936.txt

(qui peut être le titulaire lui-même, par exemple avec un serveur maître caché) signe la zone, et elle est transférée ensuite, avec clés et signatures, vers les autres. C'est tout simple. (Pour information, c'est par exemple ainsi que fonctionne `.fr`, qui a plusieurs hébergeurs en plus de l'AFNIC : les serveurs dont le nom comprend un « ext » <https://dns.bortzmeyer.org/fr/NS> sont sous-traités.)

Mais si une ou davantage des trois (plus ou moins bonnes) raisons citées plus haut s'applique ? Là, pas de AXFR (RFC 5936), il faut trouver d'autres modèles. (Et je vous préviens tout de suite : aucun de ces modèles n'est géré par les logiciels existants, il va falloir faire du devops.) Celui présenté dans le RFC est celui des signeurs multiples. Chaque hébergeur reçoit les données non-signées par un mécanime quelconque (par exemple son API) et les signe lui-même avec ses clés, plus exactement avec sa ZSK ("*Zone Signing Key*"). Pour que tous les résolveurs <https://www.bortzmeyer.org/resolveur-dns.html> validants puissent valider ces signatures, il faut que l'ensemble des clés, le DNSKEY de la zone, inclus les ZSK de tous les hébergeurs de la zone. (Un résolveur peut obtenir les signatures d'un hébergeur et les clés d'un autre.) Comment faire en sorte que tous les hébergeurs reçoivent les clés de tous les autres, pour les inclure dans le DNSKEY qu'ils servent ? Il n'existe pas de protocole pour cela, et le problème n'est pas purement technique, il est également que ces hébergeurs n'ont pas de relation entre eux. C'est donc au titulaire de la zone d'assurer cette synchronisation. (Un peu de Python ou de Perl avec les API des hébergeurs…)

Il y a deux variantes de ce modèle : KSK unique ("*Key Signing Key*") et partagée, ou bien KSK différente par hébergeur. Voyons d'abord la première variante, avec une seule KSK. Elle est typiquement gérée par le titulaire de la zone, qui est responsable de la gestion de la clé privée, et d'envoyer l'enregistrement DS à la zone parente. Par contre, chaque hébergeur a sa ZSK et signe les données. Le titulaire récupère ces ZSK (par exemple via une API de l'hébergeur) et crée l'ensemble DNSKEY, qu'il signe. (Le RFC note qu'il y a un cas particulier intéressant de ce modèle, celui où il n'y a qu'un seul hébergeur, par exemple parce que le titulaire veut garder le contrôle de la KSK mais pas gérer les serveurs de noms.)

Deuxième variante du modèle : chaque hébergeur a sa KSK (et sa ZSK). Il ouvre son API aux autres hébergeurs pour que chacun connaisse les ZSK des autres et puisse les inclure dans l'ensemble DNSKEY. Le titulaire doit récolter toutes les KSK, pour envoyer les DS correspondants à la zone parente. Le remplacement d'une KSK ("*rollover*") nécessite une bonne coordination.

Dans ces deux modèles, chaque serveur faisant autorité connait toutes les ZSK utilisées, et un résolveur validant récupérera forcément la clé permettant la validation, quel que soit le serveur faisant autorité interrogé, et quelle que soit la signature que le résolveur avait obtenue (section 3 du RFC). À noter qu'il faut que tous les hébergeurs utilisent le même algorithme de signature (section 4 du RFC) puisque la section 2.2 du RFC 4035 impose de signer avec tous les algorithmes présents dans l'ensemble DNSKEY.

En revanche, les hébergeurs ne sont pas forcés d'utiliser le même mécanisme de déni d'existence (NSEC ou NSEC3, cf. RFC 7129). Chacun peut faire comme il veut (section 5 de notre RFC) puisqu'une réponse négative est toujours accompagnée de ses NSEC (ou NSEC3). Évidemment, si un hébergeur utilise NSEC et un autre NSEC3, le gain en sécurité de NSEC3 sera inutile.

Comme signalé plus haut, le remplacement ("*rollover*") des clés, déjà une opération compliquée en temps normal, va être délicat. La section 6 du RFC détaille les procédures que l'on peut suivre dans les cas « une seule KSK » et « une KSK par hébergeur ».

La discussion plus haut supposait qu'il y avait deux clés, la KSK et la ZSK, ce qui est le mode le plus fréquent d'organisation des clés. Mais ce n'est pas obligatoire : on peut avec n'avoir qu'une seule clé (baptisée alors CSK, pour "*Common Signing Key*"). Dans ce cas, il n'y a qu'un seul modèle possible,

chaque hébergeur a sa CSK. Cela ressemble au modèle « une KSK par hébergeur » : le titulaire de la zone doit récolter toutes les CSK et envoyer les DS correspondants à la zone parente (section 7 du RFC).

Cette configuration DNSSEC à plusieurs signeurs peut fonctionner avec les enregistrements CDS et CDNSKEY des RFC 7344 et RFC 8078 (section 8 de notre RFC). Dans le premier modèle « une seule KSK », le titulaire de la zone crée CDS et/ou CDNSKEY qu'il envoie à tous les hébergeurs avec les autres données. Dans le deuxième modèle « une KSK par hébergeur », il faut échanger les CDS et/ou CDNSKEY pour que chaque hébergeur ait les enregistrements des autres.

On voit que dans cette configuration, il est nécessaire de communiquer, au moins entre le titulaire de la zone et ses hébergeurs. Ces hébergeurs doivent fournir une API, typiquement de type REST. Notre RFC ne normalise pas une telle API mais sa section 9 indique les fonctions minimum qu'elle doit fournir :

— Permettre de récupérer les enregistrements DNSKEY (pour obtenir la ZSK),
— Pour le premier modèle (une seule KSK), permettre d'indiquer l'ensemble DNSKEY complet et signé,
— Pour le deuxième modèle, permettre d'indiquer les ZSK des autres hébergeurs,
— Et idem pour les CDS et CDNSKEY.

Autre problème pratique avec la co-existence de plusieurs hébergeurs DNS, qui signent eux-mêmes la zone, la taille des réponses. Comme il faut inclure toutes les ZSK dans l'ensemble DNSKEY, les réponses aux requêtes DNSKEY vont forcément être d'assez grande taille (section 10). Le RFC fait remarquer que les algorithmes cryptographiques utilisant les courbes elliptiques ont des clés plus courtes et que cela peut aider.

Enfin, dans la section 12, dédiée aux questions de sécurité restantes, le RFC note que ces mécanismes à plusieurs signeurs nécessitent d'avoir confiance dans chacun des signeurs. Au contraire, le système où le maître signe tout et transfère ensuite aux serveurs esclaves ne demande aucune confiance dans les hébergeurs.

Allez, un joli dessin pour terminer (trouvé ici <https://twitter.com/NS1/status/1276179892749643777>) :