RFC 8145 : Signaling Trust Anchor Knowledge in DNS Security Extensions (DNSSEC)

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 19 avril 2017

Date de publication du RFC : Avril 2017

https://www.bortzmeyer.org/8145.html

L'utilisation de DNSSEC implique que le résolveur DNS ait une ou plusieurs clés de départ de la validation ("*trust anchors*"). Typiquement, le résolveur aura une clé pour la racine, les autres domaines étant validés en suivant l'arborescence du DNS (cela se configure, même si la plupart des résolveurs viennent avec une pré-configuration pour la clé ICANN de la racine). Seulement, parfois, les clés changent et le gérant d'un domaine aimerait bien savoir, avant de supprimer l'ancienne clé, si les résolveurs ont bien tous reçu la nouvelle. D'où cette nouvelle option EDNS où le résolveur signale au serveur faisant autorité la liste des clés qu'il utilise comme point de départ de la validation. (Le RFC décrit également une autre méthode, non fondée sur EDNS.)

En toute rigueur, il faut dire que le résolveur ne transmet pas les clés mais les identificateurs courts ("*key tags*" ou "*key IDs*"), qui sont un condensat de 16 bits des clés (section 3.1.6 du RFC 4034 [1], et notez dans l'annexe B du même RFC que ce ne sont pas des condensats cryptographiques). On trouve cet identificateur de clé si on utilise l'option `+multi` de dig :

```
% dig +multi DNSKEY tf
...
;; ANSWER SECTION:
tf. 172800 IN DNSKEY 257 3 8 (
                                ...
) ; KSK; alg = RSASHA256; key id = 12520
tf. 172574 IN DNSKEY 256 3 8 (
                                ...
) ; ZSK; alg = RSASHA256; key id = 51793
...
tf. 172574 IN RRSIG DNSKEY 8 1 172800 (
20170524190422 20170325180422 12520 tf.
...
```

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc4034.txt`

Il est utilisé pour la communication entre humains mais on le trouve aussi dans les enregistrements DS chez le parent :

```
% dig DS tf
...
;; ANSWER SECTION:
tf. 86400 IN DS 12520 8 2 (
2EC74274DD9AA7FFEA33E695EFF98F17F7C78ABD2D76
EDBBDE4EDD4630D68FA2 )
...
```

Ainsi que dans les signatures :

```
% dig +dnssec SOA tf
...
;; ANSWER SECTION:
tf. 172800 IN SOA nsmaster.nic.fr. hostmaster.nic.fr. (
2222242731 ; serial
                                ...
tf. 172800 IN RRSIG SOA 8 1 172800 (
20170531124004 20170401114004 51793 tf.
                                ...
```

On voit ici que la clé de `.tf` dans la racine est la 12520, qui signe la clé 51793, qui elle-même signe les enregistrements.

Si vous n'êtes pas parfaitement au point sur la terminologie DNSSEC, lisez la section 3 du RFC. Et, à titre d'exemple, voici la configuration d'un résolveur Unbound pour utiliser comme clés de départ de la validation celles de Yeti <`https://yeti-dns.org`> :

```
% cat /etc/unbound/unbound.conf
...
server:
    ...
    auto-trust-anchor-file: "/var/lib/unbound/yeti.key"
    ...

% cat /var/lib/unbound/yeti.key
. 86400 IN DNSKEY 257 3 8 AwE...8uk= ;{id = 59302 (ksk), size = 2048b} ;;state=1 [ ADDPEND ] ;;count=67 ;;la
. 86400 IN DNSKEY 257 3 8 AwE...y0U= ;{id = 19444 (ksk), size = 2048b} ;;state=2 [  VALID  ] ;;count=0 ;;las
```

On voit deux clés, d'identificateurs 59302 et 19444. Tout contenu signé avec une de ces deux clés sera accepté. (Le fait qu'il y ait deux clés alors qu'une suffirait est dû au fait qu'un changement est en cours, suivant le RFC 5011.)

Voyons maintenant la première façon de signaler ses clés dont dispose un résolveur, la méthode EDNS (section 4 de notre RFC, et voir le RFC 6891, pour les détails sur ce qu'est EDNS). On utilise une nouvelle option EDNS, `edns-key-tag` (code 14 dans le registre IANA <`https://www.iana.org/assignments/dns-parameters/dns-parameters.xml#dns-parameters-11`>). Comme toutes les options EDNS, elle comprend le code (14), la longueur, puis une suite d'identificateurs de clés. Par exemple, le résolveur Unbound montré plus haut enverrait une option {14, 4, 59302, 19444} (longueur quatre car il y a deux identificateurs, de deux octets chacun). Il est recommandé d'utiliser cette option pour toutes les requêtes de type DNSKEY (et jamais pour les autres).

Notez que le serveur qui reçoit une requête avec cette option n'a rien à faire : elle est juste là pour l'informer, la réponse n'est pas modifiée. S'il le souhaite, le serveur peut enregistrer les valeurs, permettant à son administrateur de voir, par exemple, si une nouvelle clé est largement distribuée (avant de supprimer l'ancienne).

La deuxième méthode de signalisation, celle utilisant le QNAME ("*Query Name*", le nom indiqué dans la requête DNS) figure en section 5. La requête de signalisation utilise le type NULL (valeur numérique 10 <https://www.iana.org/assignments/dns-parameters/dns-parameters.xml#dns-parameters-4>), et un nom de domaine qui commence par « _ta- », suivi de la liste des identificateurs en hexadécimal (dans cet article, ils étaient toujours montré en décimal) séparés par des traits. Le nom de la zone pour laquelle s'applique ces clés est ajouté à la fin (la plupart du temps, ce sera la racine, donc il n'y aura rien à ajouter). En reprenant l'exemple du résolveur Unbound plus haut, la requête sera `_ta-4bf4-e7a6.`. Comme ce nom n'existe pas, la réponse sera certainement NXDOMAIN.

Le serveur utilise cette requête comme il utilise l'option EDNS : ne rien changer à la réponse qui est faite, éventuellement enregistrer les valeurs indiquées, pour pouvoir informer l'administrateur du serveur.

Voilà, comme vous voyez, c'est tout simple. Reste quelques petites questions de sécurité (section 7) et de vie privée (section 8). Pour la sécurité, comme, par défaut, les requêtes DNS passent en clair (RFC 7626), un écoutant indiscret pourra savoir quelles clés utilise un résolveur. Outre que cela peut permettre, par exemple, de trouver un résolveur ayant gardé les vieilles clés, la liste peut révéler d'autres informations, par exemple sur le logiciel utilisé (selon la façon dont il met en œuvre le RFC 5011). C'est donc un problème de vie privée également.

Notez aussi que le client peut mentir, en mettant de fausses valeurs. Par exemple, il pourrait envoyer de faux messages, avec une adresse IP source usurpée, pour faire croire que beaucoup de clients ont encore l'ancienne clé, de façon à retarder un remplacement.

(Au passage, si vous voulez des informations sur le remplacement des clés DNSSEC de la racine, voyez la page de l'ICANN <https://www.icann.org/resources/pages/ksk-rollover>, et la première expérimentation Yeti <https://github.com/BII-Lab/Yeti-Project/blob/master/doc/Experiment-KROLL.md> ainsi que la deuxième <https://github.com/BII-Lab/Yeti-Project/blob/master/doc/Experiment-KROLL2.md>.)

Notez que le mécanisme utilisé a beaucoup varié au cours du développement de ce RFC (section 1.1, sur l'histoire). Au début, il n'y avait que l'option EDNS, en copiant sur le mécanisme du RFC 6975. Mais EDNS a quelques limites :

— Il n'est pas de bout en bout : si une requête passe par plusieurs résolveurs, les options EDNS ne sont pas forcément transmises,
— Il y a toujours le problème des stupides et bogués boitiers intermédiaires, qui bloquent parfois les paquets ayant une option EDNS qu'ils ne connaissent pas,
— Comme l'option n'est pas forcément envoyée à chaque requête DNS, un résolveur pourrait avoir besoin de mémoriser les valeurs envoyées par ses clients, afin de les transmettre, ce qui l'obligerait à garder davantage d'état.

L'approche concurrente, avec le QNAME, a aussi ses inconvénients :

— Elle ne permet pas de distinguer les clés connues du client, de celles connues par le client du client (si plusieurs résolveurs sont chaînés, via le mécanisme "*forwarder*"),
— Elle nécessite deux requêtes, une avec la demande normale, une avec le QNAME spécial : en cas de répartition de charge entre serveurs, par exemple avec l'"*anycast*", ces deux requêtes peuvent même aboutir sur des serveurs différents,

— Enfin la requête avec le QNAME spécial peut ne pas être transmise du tout, en cas de mise en mémoire énergique des réponses négatives par un résolveur intermédiaire (cf. RFC 8020 and RFC 8198).

D'où le choix (chaudement discuté) de fournir les deux méthodes.

À l'heure actuelle, je ne connais qu'une seule mise en œuvre de ce RFC, introduite dans BIND 9.10.5 rc1 <https://kb.isc.org/article/AA-01460/0/BIND-9.10.5rc1-Release-Notes.html> (« *"named now provides feedback to the owners of zones which have trust anchors configured by sending a daily query which encodes the keyids of the configured trust anchors for the zone. This is controlled by trust-anchor-telemetry"* »).