Déboguer les zones DNS signées avec DNSSEC

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 26 novembre 2008. Dernière mise à jour le 10 juin 2009

https://www.bortzmeyer.org/deboguer-dnssec.html

La sécurité est un aller et retour permanent entre trop et pas assez. Trop de sécurité et on ne peut plus rien faire, pas assez et les ennuis nous tombent dessus. Le DNS n'échappe pas à cette dialectique fondamentale. Pour se protéger de l'empoisonnement des résolveurs DNS, on pense à déployer DNSSEC. Oui, mais qui va déboguer les innombrables problèmes que cela va causer et comment ?

La vulnérabilité du DNS aux empoisonnements avec des données fausses est bien connue depuis longtemps (voir par exemple le RFC 3833 [1]). Elle est particulièrement prise en compte depuis la révélation de la faille Kaminsky <https://www.bortzmeyer.org/comment-fonctionne-la-faille-kaminsky.html>. Pour faire face à cette vulnérabilité, un certain nombre d'experts prônent le déploiement d'une technique de signature cryptographique, DNSSEC, normalisée dans les RFC 4033, RFC 4034 et RFC 4035.

Il existe une longue expérience de déploiement de la cryptographie sur l'Internet et l'une des leçons apprises est que des ennuis se produisent inévitablement. Bogues dans les logiciels et erreurs de procédures de la part des humains font que la cryptographie fonctionne souvent comme une serrure : elle gêne les méchants mais elle peut aussi bloquer les gentils. Normalement, le DNS est très résistant aux erreurs de configuration : il faut vraiment le faire exprès pour rendre une zone complètement inatteignable. Avec DNSSEC, on perd cette robustesse : une erreur et la zone, quoique atteignable, ne sera pas validée et les résolveurs refuseront de transmettre les données. Il est donc crucial d'apprendre à déboguer DNSSEC.

Commençons par un cas banal à l'heure actuelle. On a un résolveur DNS qui valide avec DNSSEC (dans BIND, cela se fait avec l'option `dnssec-validatation yes;`) et, un matin, une zone DNS n'est plus accessible. Le résolveur, interrogé avec dig dit juste "*Server failure*" :

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc3833.txt

```
% dig +dnssec SOA example.net

; <<>> DiG 9.5.0-P2 <<>> +dnssec @dnssec SOA example.net
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 31180
                                       ^^^^^^^^
                                       Ici
...
```

Si on a accès aux journaux de ce résolveur et s'il est configuré en mode suffisamment bavard, on pourra y trouver des détails sur la cause du problème. Mais il faut noter que les résolveurs Unbound et (surtout) BIND ont en commun que les messages ainsi journalisés sont peu compréhensibles, et nécessitent souvent la lecture des RFC pour être décodés.

À l'autre extrémité du spectre de la complexité, une solution entièrement cliquodromesque est le vérificateur du registre de `.se`, <http://dnscheck.iis.se/>. Si la zone est disponible publiquement, il peut l'analyser en détail et produire des diagnostics compréhensibles par exemple :

```
DNSSEC signature expired: RRSIG(example.net/IN/DNSKEY/8850)
Expired signatures will be ignored by validating resolvers.
```

Cet outil est également distribué, sous une licence libre, pour ceux qui veulent le faire tourner chez eux. Un autre excellent outil de vérification du DNS, Zonecheck <http://www.zonecheck.fr/>, peut faire des tests DNSSEC depuis sa version 3 <https://www.bortzmeyer.org/zonecheck-3-0.html>.

Et entre les deux ? Si on est prêt à utiliser quelques outils Unix mais qu'on n'a pas accès au journal du résolveur, ou bien qu'on ne comprend rien à ses messages ? Mark Andrews, responsable de la version actuelle de BIND, aime dire qu'on peut « déboguer DNSSEC uniquement avec `dig` et `date` ». Est-ce vrai ?

Voyons ce qu'on peut obtenir avec dig. Par défaut, la plupart des résolveurs validateurs ne distribuent aucune donnée lorsque la signature est invalide. Cela rend difficile le déboguage aussi faut-il demander gentiment au résolveur d'envoyer quand même les données invalides, avec l'option `+cd` :

```
% dig +dnssec +cd SOA example.net
...
;; ANSWER SECTION:
example.net.            86400   IN      SOA     ns1.example.net. hostmaster.example.net. 2008102101 3600 900
example.net.            86400   IN      RRSIG   SOA 5 2 86400 20081120064157 20081021064157 8850 example.net
```

Le premier enregistrement, le `SOA` est ce que nous avions demandé, le second, `RRSIG` est la signature cryptographique. La majorité est du binaire, que je ne chercherai pas à lire mais certaines données sont en clair, notamment le "*key ID*" (ici 8850) et les dates de validité de la signature.

En effet, pour empêcher un attaquant de « rejouer » de vieux enregistrements, les `RRSIG` ont une date de début et une date de fin de validité. (La période de validité est de un mois, par défaut, si on signe

avec l'outil `dnssec-signzone` de BIND.) dig affiche cette date dans un format quasiment lisible, décrit par les section 3.2 et 3.1.5 du RFC 4034 : YYYYMMDDHHmmSS en UTC. Ici, la date de fin de validité est 20081120064157, donc le 20 novembre 2008 à 06h41 UTC. Comme nous sommes le 26 novembre, pas besoin de chercher plus loin : la signature a expiré. (Pour afficher la date du jour au format identique à celui de dig, sur un Unix qui a GNU date comme Debian, on peut faire `date -u +%Y%m%d%H%M%S`.)

En effet, puisque les signatures ont une durée de vie limitée, il faut re-signer la zone périodiquement. Les futures version de BIND le feront automatiquement. Mais, en attendant, il faut mettre dans son `crontab` quelque chose comme :

```
dnssec-signzone -N increment example.net
```

À noter qu'il existe aussi des "*appliances*" DNSSEC comme celle de Secure64 qui, normalement, vous dispensent de cette tâche.

Mais, pour l'instant, il faut bien dire que la signature expirée est la cause la plus fréquente de problèmes DNSSEC. Tellement qu'elle vaut la peine d'une surveillance spécifique. Par exemple, le script check-sig (en ligne sur `https://www.bortzmeyer.org/files/check-sig.sh`) vérifie un nom de domaine et affiche un message d'erreur si sa signature est expirée ou bien si elle le sera bientôt (sept jours par défaut) :

```
% check-sig example.net
%

% check-sig SOA example.net
Name example.net has an expired signature (20081120064157)
%

% check-sig example.org
Name example.org has a signature that will expire in less than 7 days (20081201224442)
%
```

Il peut donc être mis dans une `crontab` pour donner l'alarme lorsqu'une signature risque d'expirer. (Merci à Erwan David, Thomas Parmelan, Pierre Beyssac, Phil Regnauld et tlhackque pour sa mise au point.)

Et la cause suivante, en nombre d'erreurs ? Probablement les incohérences dans la délégation. DNSSEC repose, comme le DNS, sur un modèle **hiérarchique**. Une racine délègue à des zones qui délèguent à des sous-zones et ainsi de suite. Chacune de ces délégations, matérialisées par un enregistrement `DS` (RFC 4034, section 5), est évidemment signée. La principale différence avec le DNS est que la racine de signature peut être différente de la racine tout court, grâce à DLV (RFC 5074, technique désormais abandonnée). Ainsi, à l'heure actuelle, la plupart des zones signées sont délégués depuis le registre DLV de l'ISC <`http://dlv.isc.org/`>, ce qui n'est plus le cas depuis.

Or, des problèmes peuvent survenir lors des délégations puisque elles impliquent deux organisations différentes. Par exemple, un administrateur de zone décide de signer car c'est à la mode puis arrête de le faire mais oublie qu'un enregistrement DLV pointe vers lui (et donc garantit que la zone devrait être signée). Ou bien l'administrateur modifie sa clé et oublie de prévenir la zone au-dessus. De telles contradictions entre la zone parente et la zone fille sont fréquentes dans le DNS d'aujourd'hui mais, avec DNSSEC, elles ne pardonnent pas.

Supposons donc que la zone `example.net` ne fonctionne pas : nous ne récupérons que le "*Server Failure*". Regardons sa clé (l'option `+multi` rend l'affichage des clés plus agréable) :

```
% dig +dnssec +cd +multi DNSKEY example.net
...
;; ANSWER SECTION:
example.net.              366 IN DNSKEY 257 3 5 (
                                 AwEAAc4x/KbNECb+dpDDBSvyxfTlvUxXyC3EAqCnXDp4
                                 +IxjfmwCm1QfB/VIMfqQ2bSsEu51BPk/38dBG01COvE5
                                 tYit/Ux8gIuDgZiJx+ldZ9OAJ3Lnm7v/q5+gy2LSzW46
                                 p6U45WHmGnDZ4c3uhzcfooXmQsW4UmIw+zDc2ePADy3M
                                 bkr3Srll3XDny1OHoW6Ch4o8qC+ezzRDSEnhrtpon+r9
                                 4sqXF50k6OLaqCRB3q9iaGUgviTVfZWJIlvZOwvxxpbH
                                 SDd6QThM/CZBzcx/8JHAwP7MJcUQYS8XvBwRdaAfVDuE
                                 FjUj6IF+vgn8PI1ipQUrF8L0OAHf1dHBou1XjuE=
                                 ) ; key id = 17398
```

Sa clé est la 17398. Est-elle bien chez le parent ? On demande à celui-ci :

```
% dig @addr-server-parent +dnssec +cd +multi DS example.net.
...
;; ANSWER SECTION:
example.net.              1800 IN DS 6732 5 1 (
                                 BBDDDD272C4D81EF941C722CEF79A848B543502D )
```

Oui, il y a bien un enregistrement DS mais pour une autre clé, la 6732. La chaîne de confiance est cassée là. Sans doute un changement de clé effectué sans prévenir le parent.

Attention, il est courant d'avoir plusieurs clés et il faut donc les regarder toutes.

Avec DLV, même principe. Ma zone `sources.org` est signée et enregistrée dans DLV à l'ISC, ce qu'on peut voir avec :

```
% dig @ns-ext.isc.org +multi DLV sources.org.dlv.isc.org
...
;; ANSWER SECTION:
sources.org.dlv.isc.org. 3600 IN DLV 22107 5 2 (
                                 AF12A23DFBCDB5609DCEC2C2FBD3CD65AEEFE49CBE07
                                 51C65C71C983986B7DE5 )
sources.org.dlv.isc.org. 3600 IN DLV 14347 3 1 (
                                 31FF6986A07DAC3642C18606FC992F6ED403A873 )
sources.org.dlv.isc.org. 3600 IN DLV 14347 3 2 (
                                 0D5D5B209264BBA5EDAEC9B95843901073BF27F01702
                                 B144FFC1389D747DAB7F )
sources.org.dlv.isc.org. 3600 IN DLV 22107 5 1 (
                                 EA78D532118A6C2B3C95447A7E520FF3B16FE775 )
```

Il y a deux clés, les 14347 et 22107, utilisant deux algorithmes différents, d'où les quatre DLV. Ici, tout va donc bien.

Et si la recherche d'une clé dans la zone ne donnait rien ? C'est que la zone n'est pas signée. Actuellement, c'est l'écrasante majorité. Sauf s'il existe une délégation DNSSEC depuis le parent (la zone serait alors invalide), ces zones ne doivent pas être refusées, sauf extrême paranoïa. Si elles déclenchent un "*Server Failure*", c'est qu'il y a une raison non-DNSSEC à cela.

On peut tester ces techniques avec la zone `test.dnssec-tools.org` où se trouvent de nombreux enregistrements DNSSEC cassés volontairement. Par exemple, `futuredate-A.newzsk-ns.test.dnssec-tools`

a une date des signatures valide seulement dans le futur... Même chose avec un autre domaine cassé exprès, `dnssec-failed.org`.

Bien sûr, un million d'autres choses peuvent tomber en panne. Par exemple, DNSSEC dépend d'EDNS0, puisque les réponses DNSSEC sont typiquement plus grosses. L'expérience prouve qu'un certain nombre de pare-feux bogués interceptent les paquets avec EDNS0 ou bien qu'un certain nombre d'administrateurs incompétents configurent leur pare-feu pour rejeter les paquets DNS de taille supérieur à 512 octets. Il faut donc également être prêt à déboguer la configuration réseau.

On trouve de nombreuses indications pratiques sur le déboguage de DNSSEC dans le "*DNSSEC HOWTO, a tutorial in disguise*" <`http://www.nlnetlabs.nl/dnssec_howto/`> ou bien dans les transparents "*DNSSEC in 6 minutes*" <`https://kb.isc.org/getAttach/38/AA-00820/DNSSEC_in_6_minutes.pdf`>. Merci à Mohsen Souissi pour le déboguage du texte et à Mark Andrews pour ses exemples de déboguage sur la liste bind-users.