La faille DNSSEC KeyTrap

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 19 mars 2024

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

Le 16 février a été publiée la faille de sécurité DNSSEC KeyTrap. Je sais, c'est un peu tard pour en parler mais c'est quand même utile, non ?

KeyTrap <https://www.athene-center.de/en/keytrap> est une faille qui permet un déni de service contre des résolveurs DNS <https://www.bortzmeyer.org/resolveur-dns.html> validants, c'est-à-dire qui vérifient les signatures cryptographiques de DNSSEC. Avec peu de messages DNS, parfois un seul, elle permet de stopper toute résolution de noms pendant des minutes, voire des heures. Comme souvent, hélas, en sécurité informatique, les découvreurs de la faille ont sérieusement abusé des grands mots dans leur communication (que les médias, comme d'habitude, ont bien relayés sans esprit critique) mais la faille est réelle et ils ont fait un bon travail pour la cerner exactement, et la reproduire afin de tester des remèdes.

Comment fonctionne KeyTrap ? Le point de départ est d'observer qu'un résolveur validant ne cherche pas à échouer dans la résolution de noms, il cherche à trouver, si nécessaire en insistant, un enregistrement correctement signé. Par exemple, si un des serveurs faisant autorité <https://www.bortzmeyer.org/serveur-dns-faisant-autorite.html> ne renvoie pas de signatures (il devrait), le résolveur va demander à un autre. Si une signature est invalide, le résolveur va en essayer d'autres. Cela part d'une bonne intention, éviter un échec des requêtes des clients. Mais cela ouvre la possibilité, pour un client malveillant, de donner **beaucoup** de travail au résolveur.

Le résolveur va essayer, pour chaque signature trouvée, toutes les clés cryptographiques présentes. En temps normal, il n'y a qu'une signature et une clé. Lors d'opérations comme le remplacement d'une clé, il peut y en avoir deux. Mais, et c'est le point important, ces nombres sont contrôlés par l'attaquant. S'il envoie dix clés et dix signatures, le résolveur aura cent vérifications à faire. Et si l'attaquant arrive à faire tenir cent clés et cent signatures dans le message, il faudra dix mille vérifications...Elles retiendront le résolveur pendant longtemps, l'empêchant de répondre aux autres requêtes (voire le bloquant complètement si ces vérifications ne sont pas réellement parallélisées).

Voyons maintenant quelques détails pratiques. D'abord, le résolveur ne va pas tester toutes les clés mais uniquement toutes celles qui ont l'identificateur, le "*keytag*" indiqué dans la signature. Si on trouve cette signature :

```
% dig +multi +dnssec brisbane.now.weather.dyn.bortzmeyer.fr TXT
...
;; ANSWER SECTION:
brisbane.now.weather.dyn.bortzmeyer.fr. 1800 IN TXT "Brisbane" "Partly cloudy" "27.0 C" "precipitation 0.17

brisbane.now.weather.dyn.bortzmeyer.fr. 1800 IN RRSIG TXT 8 6 1800 (
20240323040000 20240318154000 63937 dyn.bortzmeyer.fr.
gUXD/SnFywfzQVRstRH9t5k6DIGXLHUeuSgM8ShNfTUx
...
```

Alors, il n'est pas nécessaire d'essayer toutes les clés mais seulement celle(s) qui ont un identificateur ("*keytag*" ou "*key id*") de 63937 (il est indiqué dans la signature, après les dates). Ce système est censé limiter le travail du résolveur. Si un attaquant envoie cent clés, une seule sera utilisée. Mais, car il y a un mais, ce "*keytag*" ne fait que deux octets (donc les collisions sont relativement fréquentes) et surtout, ce n'est pas un condensat cryptographique sûr, comme par exemple SHA-256. Il est facile pour un attaquant de générer cent clés ayant le même "*keytag*" et celui-ci ne sert alors plus à rien, le malheureux résolveur doit essayer toutes les clés.

Autre problème, pour l'attaquant (qui veut évidemment faire le plus de mal possible), la taille de la réponse. En UDP, un client DNS typique ne va pas accepter des réponses de plus de 4 096 octets, voire souvent 1 460 octets. Comme l'attaquant veut fourrer le plus grand nombre possible de clés et de signatures dans sa réponse, il a intérêt à utiliser les algorithmes à courbes elliptiques, comme ECDSA, dont les clés et les signatures sont bien plus petites qu'avec RSA.

Maintenant, quelles sont les solutions ? Le rapport sur KeyTrap est plein de phrases boursouflées comme « "*Solving these issues fundamentally requires to reconsider the basics of the design philosophy of the Internet*". » C'est évidemment faux. Il faut simplement limiter le nombre d'opérations et/ou le temps passé. C'est une nécessité général du DNS, bien antérieure à KeyTrap. Le RFC 1035[1], en 1987, disait déjà « "*The amount of work which a resolver will do in response to a client request must be limited to guard against errors in the database, such as circular CNAME references, and operational problems, such as network partition which prevents the resolver from accessing the name servers it needs. While local limits on the number of times a resolver will retransmit a particular query to a particular name server address are essential, the resolver should have a global per-request counter to limit work on a single request.*" ». L'oubli de ce paragraphe a déjà mené à des possibilités d'attaque par déni de service comme l'attaque infinie, iDNS <https://indico.dns-oarc.net/event/21/contributions/301/attachments/272/492/slides.pdf>. La solution est donc évidente, limiter le nombre de vérifications. Il y a une part d'arbitraire dans cette limite (stopper au bout de trois clés ? de quatre ?) mais il est clair qu'il n'y a aucune raison légitime de vérifier cent clés. C'est la modification qui a été faite dans tous les résolveurs.

Noter que, pour l'exploitation de cette faille, il faut pouvoir parler au résolveur, pour lui demander de résoudre un nom dans le domaine contrôlé par l'attaquant, domaine qu'il aura rempli de clés et de signatures. C'est évidemment facile pour un résolveur public, et pas trop difficile pour les gros FAI, mais plus compliqué pour des petits résolveurs locaux. Donc, tout résolveur validant était plus ou moins menacé. Ceci dit, tous les logiciels ont été patchés très vite et tous les administrateurs système sérieux ont déjà appliqué les patches.

Notez que cette attaque ne remet pas en cause l'importance de DNSSEC. Toute technique de sécurité peut être (plus ou moins facilement) détournée pour en faire une attaque par déni de service.

Quelques lectures :

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc1035.txt

— Le rapport originel <https://www.athene-center.de/fileadmin/content/PDF/Keytrap_2401.pdf>. Très bon travail, très détaillé, sa lecture est recommandée, il faut juste ignorer les nombreuses exagérations dramatiques. Idem dans ce résumé d'une des auteures <https://labs.ripe.net/author/haya-shulman/keytrap-algorithmic-complexity-attacks-exploit-func>.
— Un bon résumé de Geoff Huston <https://www.potaroo.net/ispcol/2024-03/keytrap.html>.
— L'attaque avait été documentée des années auparavant <https://essay.utwente.nl/78777/> mais sans être médiatisée, et était donc passé inaperçue.
— Coïncidence, le TLD .ru a connu au même moment une grande panne <https://www.bortzmeyer.org/ru-dnssec.html> liée à une collision de "*keytags*"