RFC 9609 : Initializing a DNS Resolver with Priming Queries

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 12 février 2025

Date de publication du RFC : Février 2025

https://www.bortzmeyer.org/9609.html

Un résolveur DNS ne connait au début, rien du contenu du DNS. Rien ? Pas tout à fait, il connait une liste des serveurs de noms faisant autorité pour la racine, car c'est par eux qu'il va commencer le processus de résolution de noms. Cette liste est typiquement en dur dans le code du serveur, ou bien dans un de ses fichiers de configuration. Mais peu d'administrateurs système la maintiennent à jour. Il est donc prudent, au démarrage du résolveur, de chercher une liste vraiment à jour, et c'est le "*priming*" (amorçage, comme lorsqu'on amorce une pompe pour qu'elle fonctionne seule ensuite), opération que décrit ce RFC, qui remplace l'ancienne version, le RFC 8109 [1] (les changements sont nombreux mais sont surtout des points de détail).

Le problème de départ d'un résolveur est un problème d'œuf et de poule. Le résolveur doit interroger le DNS pour avoir des informations mais comment trouve-t-il les serveurs DNS à interroger ? La solution est de traiter la racine du DNS de manière spéciale : la liste de ses serveurs est connue du résolveur au démarrage. Elle peut être dans le code du serveur lui-même, ici un Unbound qui contient les adresses IP des serveurs de la racine (je ne montre que les trois premiers, `A.root-servers.net`, `B.root-servers.net` et `C.root-servers.net`) :

```
% strings /usr/sbin/unbound | grep -i 2001:
2001:503:ba3e::2:30
2001:500:84::b
2001:500:2::c
...
```

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc8109.txt`

Ou bien elle est dans un fichier de configuration (ici, sur un Unbound) :

```
server:
  directory: "/etc/unbound"
  root-hints: "root-hints"
```

Ce fichier peut être téléchargé via l'IANA <https://www.internic.net/domain/named.root>, il peut être spécifique au logiciel résolveur, ou bien fourni par le système d'exploitation (cas du paquetage dns-root-data chez Debian). Il contient la liste des serveurs de la racine et leurs adresses :

```
.                       3600000      NS    A.ROOT-SERVERS.NET.
.                       3600000      NS    B.ROOT-SERVERS.NET.
...
A.ROOT-SERVERS.NET.     3600000      A     198.41.0.4
A.ROOT-SERVERS.NET.     3600000      AAAA  2001:503:ba3e::2:30
B.ROOT-SERVERS.NET.     3600000      A     192.228.79.201
B.ROOT-SERVERS.NET.     3600000      AAAA  2001:500:84::b
...
```

Tous les résolveurs actuels mettent en œuvre le "*priming*". En effet, les configurations locales tendent à ne plus être à jour au bout d'un moment. (Sauf dans le cas où elles sont dans un paquetage du système d'exploitation, mis à jour avec ce dernier, comme dans le bon exemple Debian ci-dessus.)

Les changements des serveurs racines sont rares. Si on regarde sur le site des opérateurs des serveurs racine <https://root-servers.org/>, on voit :

— 2017-08-10 B-Root's IPv4 address to be renumbered on 2017-10-24
— 2016-12-02 Announcement of IPv6 addresses
— 2015-11-05 L-Root IPv6 Renumbering
— 2015-08-31 H-Root to be renumbered
— 2014-03-26 IPv6 service address for c.root-servers.net (2001 :500 :2 : :C)
— 2012-12-14 D-Root IPv4 Address to be Renumbered

Bref, peu de changements. Ils sont en général annoncés sur les listes de diffusion opérationnelles (comme ici <https://lists.dns-oarc.net/pipermail/dns-operations/2015-September/013635.html>). Mais les fichiers de configuration ayant une fâcheuse tendance à ne pas être mis à jour et à prendre de l'âge, les anciennes adresses des serveurs racine continuent à recevoir du trafic des années après (comme le montre cette étude de J-root <https://indico.dns-oarc.net/event/24/session/10/contribution/10/material/slides/0.pdf>). Notez que la stabilité de la liste des serveurs racine n'est pas due qu'au désir de ne pas perturber les administrateurs système : il y a aussi des raisons politiques (aucun mécanisme en place pour choisir de nouveaux serveurs, ou pour retirer les « maillons faibles »). C'est pour cela que la liste des serveurs (mais pas leurs adresses) n'a pas changé depuis 1997 ! (Une histoire - biaisée - des serveurs racine a été publiée par l'ICANN <https://www.icann.org/en/system/files/files/rssac-023-17jun20-en.pdf>.)

Notons aussi que l'administrateur système d'un résolveur peut changer la liste des serveurs de noms de la racine pour une autre liste, par exemple s'ielle veut utiliser une racine alternative.

Le "*priming*", maintenant. Le principe du "*priming*" est, au démarrage, de faire une requête à un des serveurs listés dans la configuration et de garder sa réponse (certainement plus à jour que la configuration) :

```
%  dig +bufsize=4096 +norecurse +nodnssec @i.root-servers.net  . NS

; <<>> DiG 9.18.28-0ubuntu0.24.04.1-Ubuntu <<>> +bufsize +norecurse +nodnssec @i.root-servers.net . NS
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 54419
;; flags: qr aa; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 27

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: c0a47c1200bf6bfc0100000067377b40981cc05cdb33b736 (good)
;; QUESTION SECTION:
;.   IN NS

;; ANSWER SECTION:
.   518400 IN NS k.root-servers.net.
.   518400 IN NS h.root-servers.net.
.   518400 IN NS m.root-servers.net.
.   518400 IN NS l.root-servers.net.
.   518400 IN NS b.root-servers.net.
.   518400 IN NS e.root-servers.net.
.   518400 IN NS d.root-servers.net.
.   518400 IN NS f.root-servers.net.
.   518400 IN NS g.root-servers.net.
.   518400 IN NS i.root-servers.net.
.   518400 IN NS a.root-servers.net.
.   518400 IN NS j.root-servers.net.
.   518400 IN NS c.root-servers.net.

;; ADDITIONAL SECTION:
m.root-servers.net. 518400 IN A 202.12.27.33
l.root-servers.net. 518400 IN A 199.7.83.42
k.root-servers.net. 518400 IN A 193.0.14.129
j.root-servers.net. 518400 IN A 192.58.128.30
i.root-servers.net. 518400 IN A 192.36.148.17
h.root-servers.net. 518400 IN A 198.97.190.53
g.root-servers.net. 518400 IN A 192.112.36.4
f.root-servers.net. 518400 IN A 192.5.5.241
e.root-servers.net. 518400 IN A 192.203.230.10
d.root-servers.net. 518400 IN A 199.7.91.13
c.root-servers.net. 518400 IN A 192.33.4.12
b.root-servers.net. 518400 IN A 170.247.170.2
a.root-servers.net. 518400 IN A 198.41.0.4
m.root-servers.net. 518400 IN AAAA 2001:dc3::35
l.root-servers.net. 518400 IN AAAA 2001:500:9f::42
k.root-servers.net. 518400 IN AAAA 2001:7fd::1
j.root-servers.net. 518400 IN AAAA 2001:503:c27::2:30
i.root-servers.net. 518400 IN AAAA 2001:7fe::53
h.root-servers.net. 518400 IN AAAA 2001:500:1::53
g.root-servers.net. 518400 IN AAAA 2001:500:12::d0d
f.root-servers.net. 518400 IN AAAA 2001:500:2f::f
e.root-servers.net. 518400 IN AAAA 2001:500:a8::e
d.root-servers.net. 518400 IN AAAA 2001:500:2d::d
c.root-servers.net. 518400 IN AAAA 2001:500:2::c
b.root-servers.net. 518400 IN AAAA 2801:1b8:10::b
a.root-servers.net. 518400 IN AAAA 2001:503:ba3e::2:30

;; Query time: 6 msec
;; SERVER: 192.36.148.17#53(i.root-servers.net) (UDP)
;; WHEN: Fri Nov 15 17:48:00 CET 2024
;; MSG SIZE  rcvd: 851
```

(Les raisons du choix des trois options données à dig sont indiquées plus loin.)

La section 3 de notre RFC décrit en détail à quoi ressemblent les requêtes de "*priming*". Le type de données demandé ("*QTYPE*") est NS ("*Name Servers*", type 2) et le nom demandé ("*QNAME*") est « . » (oui, juste la racine). D'où le `dig . NS` ci-dessus. Le bit RD ("*Recursion Desired*") est typiquement mis à zéro (d'où le `+norecurse` dans l'exemple avec dig). La taille de la réponse dépassant les 512 octets (limite très ancienne du DNS), il faut utiliser EDNS (cause du `+bufsize=4096` dans l'exemple). On peut utiliser le bit DO ("*DNSSEC OK*") qui indique qu'on demande les signatures DNSSEC mais ce n'est pas habituel (d'où le `+nodnssec` dans l'exemple). En effet, si la racine est signée, permettant d'authentifier l'ensemble d'enregistrements NS, la zone `root-servers.net`, où se trouvent actuellement tous les serveurs de la racine, ne l'est pas, et les enregistrements A et AAAA ne peuvent donc pas être validés avec DNSSEC.

Cette requête de "*priming*" est envoyée lorsque le résolveur démarre, et aussi lorsque la réponse précédente a expiré (regardez le TTL dans l'exemple : six jours). Si le premier serveur testé ne répond pas, on essaie avec un autre. Ainsi, même si le fichier de configuration n'est pas parfaitement à jour (des vieilles adresses y trainent), le résolveur finira par avoir la liste correcte.

Et comment choisit-on le premier serveur qu'on interroge ? Notre RFC recommande (section 3.2) un tirage au sort, pour éviter que toutes les requêtes de "*priming*" ne se concentrent sur un seul serveur (par exemple le premier de la liste). Une fois que le résolveur a démarré, il peut aussi se souvenir du serveur le plus rapide, et n'interroger que celui-ci, ce qui est fait par la plupart des résolveurs, pour les requêtes ordinaires (mais n'est pas conseillé pour le "*priming*").

Et les réponses au "*priming*" ? Il faut bien noter que, pour le serveur racine, les requêtes "*priming*" sont des requêtes comme les autres, et ne font pas l'objet d'un traitement particulier. Normalement, la réponse doit avoir le code de retour `NOERROR` (c'est bien le cas dans mon exemple). Parmi les "*flags*", il doit y avoir AA ("*Authoritative Answer*"). La section de réponse doit évidemment contenir les NS de la racine, et la section additionnelle les adresses IP. Le résolveur garde alors cette réponse dans son cache, comme il le ferait pour n'importe quelle autre réponse. Notez que là aussi, il ne faut pas de traitement particulier. Par exemple, le résolveur ne doit pas compter sur le fait qu'il y aura exactement 13 serveurs, même si c'est le cas depuis longtemps (ça peut changer).

Normalement, le serveur racine envoie la totalité des adresses IP (deux par serveur, une en IPv4 et une en IPv6). S'il ne le fait pas (par exemple par manque de place parce qu'on a bêtement oublié EDNS), le résolveur va devoir envoyer des requêtes A et AAAA explicites pour obtenir les adresses IP (ici, on interroge `k.root-servers.net`) :

```
%  dig @2001:7fd::1  g.root-servers.net  AAAA

; <<>> DiG 9.18.33-1~deb12u2-Debian <<>> @2001:7fd::1 g.root-servers.net AAAA
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10640
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 1232
;; QUESTION SECTION:
;g.root-servers.net. IN AAAA

;; ANSWER SECTION:
g.root-servers.net. 3600000 IN AAAA 2001:500:12::d0d

;; Query time: 3 msec
```

```
;; SERVER: 2001:7fd::1#53(2001:7fd::1) (UDP)
;; WHEN: Thu Feb 13 17:26:51 CET 2025
;; MSG SIZE  rcvd: 75
```

Vous pouvez voir ici les requêtes et réponses de "*priming*" d'un Unbound. D'abord, décodées par tcpdump :

```
17:57:14.439382 IP (tos 0x0, ttl 64, id 26437, offset 0, flags [none], proto UDP (17), length 56)
    10.10.156.82.51893 > 198.41.0.4.53: [bad udp cksum 0x6cbf -> 0x49dc!] 63049% [1au] NS? . ar: . OPT UDPsize=1
17:57:14.451715 IP (tos 0x40, ttl 52, id 45305, offset 0, flags [none], proto UDP (17), length 1125)
    198.41.0.4.53 > 10.10.156.82.51893: [udp sum ok] 63049*- q: NS? . 14/0/27 . [6d] NS l.root-servers.net., . [
```

Et ici par tshark :

```
1   0.000000 10.10.156.82 → 198.41.0.4   DNS 70  Standard query 0xf649 NS <Root> OPT
2   0.012333   198.41.0.4 → 10.10.156.82 DNS 1139  Standard query response 0xf649 NS <Root> NS l.root-servers.net
```

Et un décodage plus détaillé de tshark dans ce fichier (en ligne sur `https://www.bortzmeyer.org/files/dns-priming-regular.txt`).

Enfin, la section 6 de notre RFC traite des problèmes de sécurité du "*priming*". Évidemment, si un attaquant injecte une fausse réponse aux requêtes de "*priming*", il pourra détourner toutes les requêtes ultérieures vers des machines de son choix. À part le RFC 5452, et l'utilisation de cookies (RFC 7873, mais seuls les serveurs racine C, F, G et I les gèrent aujourd'hui), la seule protection est DNSSEC : si le résolveur valide (et a donc la clé publique de la racine), il pourra détecter que les réponses sont mensongères (voir aussi la section 3.3). Cela a l'avantage de protéger également contre d'autres attaques, ne touchant pas au "*priming*", comme les attaques sur le routage.

Notez que DNSSEC est recommandé pour valider les réponses ultérieures mais, comme on l'a vu, n'est pas important pour valider la réponse de "*priming*" elle-même, puisque `root-servers.net` n'est pas signé. Si un attaquant détournait, d'une manière ou d'une autre, vers un faux serveur racine, servant de fausses données, ce ne serait qu'une attaque par déni de service, puisque le résolveur validant pourrait détecter que les réponses sont fausses.

L'annexe A du RFC liste les changements depuis le RFC 8109. Rien de vraiment crucial, juste une longue liste de clarifications et de précisions.