RFC 8109 : Initializing a DNS Resolver with Priming Queries

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 16 mars 2017

Date de publication du RFC : Mars 2017

http://www.bortzmeyer.org/8109.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*" (initialisation ?), opération que décrit ce RFC.

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
...
```

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
...
```

Cette configuration initiale du résolveur est décrite dans la section 2.3 du RFC 1034[1], mais ce dernier ne décrit pas réellement le "*priming*" (quoi que dise notre nouveau RFC), "*priming*" que tous les résolveurs actuels mettent en œuvre. 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 <http://root-servers.org/>, on voit :

— 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
— 2011-06-10 IPv6 service address for d.root-servers.net (2001 :500 :2D : :D)

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/2007-October/002083.html>, là <https://lists.dns-oarc.net/pipermail/dns-operations/2015-September/013635.html> ou encore ici <https://lists.dns-oarc.net/pipermail/dns-operations/2012-December/009429.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 !

Notons aussi que l'administrateur système d'un résolveur peut changer la liste des serveurs de noms de la racine pour une autre liste. C'est ainsi que fonctionnent les racines alternatives comme Yeti <https://yeti-dns.org/>. Si on veut utiliser cette racine expérimentale et pas la racine « officielle », on édite la configuration de son résolveur :

```
server:
    root-hints: "yeti-hints"
```

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc1034.txt

Et le fichier, téléchargé chez Yeti <https://yeti-dns.org/rootzone.html>, contient :

```
.                           3600000    IN   NS       bii.dns-lab.net
bii.dns-lab.net             3600000    IN   AAAA     240c:f:1:22::6
.                           3600000    IN   NS       yeti-ns.tisf.net
yeti-ns.tisf.net            3600000    IN   AAAA     2001:559:8000::6
.                           3600000    IN   NS       yeti-ns.wide.ad.jp
yeti-ns.wide.ad.jp          3600000    IN   AAAA     2001:200:1d9::35
.                           3600000    IN   NS       yeti-ns.as59715.net
...
```

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 @k.root-servers.net NS .

; <<>> DiG 9.10.3-P4-Debian <<>> +norecurse +nodnssec @k.root-servers.net NS .
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 42123
;; flags: qr aa; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 27

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;. IN NS

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

;; ADDITIONAL SECTION:
a.root-servers.net. 518400 IN A 198.41.0.4
a.root-servers.net. 518400 IN AAAA 2001:503:ba3e::2:30
b.root-servers.net. 518400 IN A 192.228.79.201
b.root-servers.net. 518400 IN AAAA 2001:500:84::b
c.root-servers.net. 518400 IN A 192.33.4.12
c.root-servers.net. 518400 IN AAAA 2001:500:2::c
d.root-servers.net. 518400 IN A 199.7.91.13
d.root-servers.net. 518400 IN AAAA 2001:500:2d::d
e.root-servers.net. 518400 IN A 192.203.230.10
e.root-servers.net. 518400 IN AAAA 2001:500:a8::e
f.root-servers.net. 518400 IN A 192.5.5.241
f.root-servers.net. 518400 IN AAAA 2001:500:2f::f
g.root-servers.net. 518400 IN A 192.112.36.4
g.root-servers.net. 518400 IN AAAA 2001:500:12::d0d
h.root-servers.net. 518400 IN A 198.97.190.53
h.root-servers.net. 518400 IN AAAA 2001:500:1::53
i.root-servers.net. 518400 IN A 192.36.148.17
```

```
i.root-servers.net. 518400 IN AAAA 2001:7fe::53
j.root-servers.net. 518400 IN A 192.58.128.30
j.root-servers.net. 518400 IN AAAA 2001:503:c27::2:30
k.root-servers.net. 518400 IN A 193.0.14.129
k.root-servers.net. 518400 IN AAAA 2001:7fd::1
l.root-servers.net. 518400 IN A 199.7.83.42
l.root-servers.net. 518400 IN AAAA 2001:500:9f::42
m.root-servers.net. 518400 IN A 202.12.27.33
m.root-servers.net. 518400 IN AAAA 2001:dc3::35

;; Query time: 3 msec
;; SERVER: 2001:7fd::1#53(2001:7fd::1)
;; WHEN: Fri Mar 03 17:29:05 CET 2017
;; MSG SIZE  rcvd: 811
```

(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 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 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 :

```
% dig @k.root-servers.net A g.root-servers.net

; <<>> DiG 9.10.3-P4-Debian <<>> @k.root-servers.net A g.root-servers.net
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 49091
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 13, ADDITIONAL: 26
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;g.root-servers.net. IN A

;; ANSWER SECTION:
g.root-servers.net. 3600000 IN A 192.112.36.4
...
```

Vous pouvez voir ici les requêtes et réponses de "*priming*" d'un Unbound utilisant Yeti <https://yeti-dns.org/>. D'abord, décodées par tcpdump :

```
20:31:36.226325 IP6 2001:4b98:dc2:43:216:3eff:fea9:41a.7300 > 2a02:cdc5:9715:0:185:5:203:53.53: 50959% [1au] NS?
20:31:36.264584 IP6 2a02:cdc5:9715:0:185:5:203:53.53 > 2001:4b98:dc2:43:216:3eff:fea9:41a.7300: 50959*- 26/0/7 N
```

Et ici par tshark :

```
1   0.000000 2001:4b98:dc2:43:216:3eff:fea9:41a &#8594; 2a02:cdc5:9715:0:185:5:203:53 DNS 90 Standard query 0xc7
2   0.038259 2a02:cdc5:9715:0:185:5:203:53 &#8594; 2001:4b98:dc2:43:216:3eff:fea9:41a DNS 1287 Standard query re
```

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

Enfin, la section 5 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, 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. 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.

Ce RFC a connu une très longue gestation puisque le premier brouillon date de février 2007 (vous pouvez admirer la chronologie <https://datatracker.ietf.org/doc/draft-ietf-dnsop-resolver-priming>).