Abstract

Some DNS recursive resolvers have long round trip times to the nearest DSN root server, which has been an obstacle to DNS query performance. In order to decrease root record fetch time without introducing a new source of errors, this document proposes a root-specific modification to the caching rules.
1. Introduction

Some DNS recursive resolvers suffer from long round trip times to the nearest DNS root server, which has been an obstacle to DNS query performance.

A particular characteristic of the root zone is that when cached, its data is usable for very different queries: An MTA that wishes to send mail to Google needs the NS records for .com, and so does a web browser that wishes to open the Bing home page. Other public zones (such as .co.uk and .gen.nz, and perhaps tumblr.com) are shared among some queries, the root zone is used for all.

This suggests that caching rules that are appropriate to the rest of the DNS tree may not be ideal for the root zone.

We propose to refresh root zone data probabilistically when it expires, instead of when needed.

2. Terminology

The basic key words such as "MUST", "MUST NOT", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "MAY", and "MAY NOT" are to be interpreted as described in [RFC2119].
The basic DNS terms used in this specification are defined in the documents [RFC1034] and [RFC1035].

3. Design Considerations

- The RRs in the root zone do not change frequently.

- The root zone is not large, compared to the RAM of even smallish resolvers.

- DNSSEC ([RFC4033],[RFC4034],[RFC4035]) protects the data origin authentication and data integrity.

4. Changes

When an RR in a resolver’s cache expires and is in the root zone, then the resolver immediately refreshes it. There are no protocol changes or extensions.

Assuming that the lookup frequency for a root-zone RR drops by half for every additional week, (ie. half of all RRs that looked up repeatedly are looked up every week, a quarter every second week, an eighth every third week, etc), this eliminates root-zone delay as a timing factor for more than 99.999% of queries through this resolver.

In practice, this should mean that unintentional clearing of the resolver’s cache (e.g. as a side effect of restarting the resolver) is the next biggest contributor to slow queries.

OPEN ISSUE: Or perhaps better, only with 95% likelihood? If the resolver refreshes it with 100% certainty, then the resolver necessarily grows to storing all of the root-zone RRs it has needed forever. If the resolver refreshes it 95% of the time and root-zone RRs have a TTL of around a week, then an unused root-zone RR has around 50% chance of being discarded after three months. The resolver will perform around 12 DNS queries that turn out, in hindsight, not to be necessary. The text below assumes 95% likelihood.

4.1. Impact on the resolver

The resolver is able to answer DNS queries quickly for all root RRs that have been used in the past several months, instead of the past week. The cost in additional processing and RAM is negligible; there are no additional tasks that can go wrong.
4.2. Impact on the root servers

The root servers one additional query per TTL (usually week) per resolver and RR, for the RRs that have been needed by that resolver in the past, but will not be needed in the coming week. The queries arrive evenly. They do not peak around a particular time, but are distributed as the normal traffic.

4.3. Impact on the network

There is no additional network traffic related to ongoing use of the network (or DNS). There are also no savings. However, some packets are sent earlier than they would be without this document.

Around 25 additional packets are transmitted (two per week over a period of some months) when a the users of a particular resolver stop using a particular root-zone RR.

5. System Requirements

In order to implement the mechanism described in this document:

- The system MUST be able to validate DNSSEC resource records.
- The system MUST have an up-to-date copy of the DNS root key.

6. Difference between this mechanism and RFC7706 based mechanism

The following features are considered to be different compared to RFC7706 based mechanism:

- This document retrieves single RRs (or probably sets, as required by DNSSEC validation). RFC7706 retrieves the entire zone.
- This document requires no actions by human administrators.
- This document provides only a probabilistic performance improvement; RFC 7706 provides a guarantee.

7. Security Considerations

None.

8. Change History

RFC Editor: Please remove this section.
8.1. draft-arnt-yao-dnsop-root-data-caching: Version 00

- Decreasing fetch time of root data by additional caching rules

9. References

9.1. Normative References


9.2. Informative References

[Root-loopback]
Authors’ Addresses

Arnt Gulbrandsen

Email: arnt@gulbrandsen.priv.no

Jiankang Yao
CNNIC
4 South 4th Street, Zhongguancun, Haidian District
Beijing, Beijing 100190
China

Phone: +86 10 5881 3007
Email: yaojk@cnnic.cn