Delegation Revalidation

https://tools.ietf.org/html/draft-huque-dnsop-ns-revalidation-01

IETF DNS Operations WG Virtual Interim
April 23rd 2020

Paul Vixie, **Shumon Huque**, Ralph Dolmans

Motivation

A range of behaviors in DNS resolvers today.

Aim to get more commonality and predictability in behavior.

And solve a set of operational issues.

Recommendations in the draft:

1. Deterministically obtain Child NS RRset and cache it over Parent
2. Re-validate the delegation at the Parent at the expiration of the Parent NS RRset TTL

Prefer Child NS (Upgrading NS RRset Credibility)

- Child NS RRset is “authoritative”. Parent is “non-authoritative” glue.
- From RFC 2181 “Clarifications to the DNS Specification”, Section 5.4.1, “Ranking Data”:
  - Authoritative data included in the answer section of an authoritative reply
  - Data from the authority section of an authoritative reply
  - [...]
  - Data from the authority section of a non-authoritative answer
- Child NS set can be signed with DNSSEC. Parent NS set can’t.
  - Again from RFC2181: “When DNS security [RFC2065] is in use, and an authenticated reply has been received and verified, the data thus authenticated shall be considered more trustworthy than unauthenticated data of the same type.

How to obtain the Child NS set?

DNS protocol does not require resolvers to explicitly fetch the authoritative NS set.

They typically see it when authoritative responses from the child zone include the NS RRset in the Authority section of their DNS responses. (By data ranking rules, they should then place this data preferentially in their cache). But this is becoming less common with “minimal responses”.

Otherwise, they have to wait until some downstream client of the resolver explicitly makes an NS query - not something that normal end-user applications do.

The draft recommends: when following referral, issue parallel query for child NS.

Who dictates the NS RRset TTL

DNS protocol says the authoritative content of the NS set is located at the apex of the zone in question (i.e. child zone).

This allows the zone owner to dictate the TTL of the NS set (if resolvers see and preferentially place it in their caches)

Allows the child zone operator to more rapidly make changes to their nameserver infrastructure by temporarily deploying shorter TTLs (e.g. moving to a new DNS operator/provider).

Implementer’s note

NS query need not be in the fast path

Could be sent in parallel and processed opportunistically, doesn’t have to delay name resolution for the triggering query.

Not much risk, little complexity, no speed difference.

Opportunistic nature can deal with broken authoritative servers that don’t respond to explicit NS queries (yes, there are some still out there), without penalty.

Delegation Revalidation

Resolvers need to re-check the parent delegation at the expiration of the parent NS set TTL (at the latest).

Prevents child zones living on beyond their authorized lifetime (in case the parent has removed the delegation, or re-delegated the zone elsewhere).

This could happen because the child zone operator maliciously set a very long TTL in an attempt to prolong its life in resolver caches, or perhaps by accident, or because resolvers that do pre-fetching of the NS RRset continue to extend the lifetime of the child zone. (see papers on “Ghost Domains”)

Delegation Revalidation

The simple scheme to do this: cap the NS TTL to the lower of the child and parent NS TTL.

A more complex algorithm is also presented in the draft that deals more effectively with some more involved, corner-case configurations.

From dnsop list discussion: If all child name servers lame, this should also trigger a revalidation action, in conjunction with a hold-down timer, to avoid DoS.

Past and current work

Vixie *et al*'s "resimprove" draft from 2010

Wijngaard's "Resolver Mitigations" draft from 2009

The Unbound resolver, from NLnet Labs roughly implements this, with a configuration knob, "harden-referral-path".

We are not re-designing zone delegation (yet)

Proposing a minimal set of changes to improve things; no wire protocol change & unilateral resolver action.

However, there are undoubtedly deficiencies in the zone delegation mechanism

- Why can delegation data be spoofed without detection, and only be discovered at a much later stage in the iterative resolution process?
- Shouldn’t we have had signed delegation records in the parent from the beginning (with a new record type, at a possibly distinct node)?
- Maybe, but if so, that is a longer term project that requires more significant protocol changes.