Workgroup: Internet Engineering Task Force
Internet-Draft:
draft-ietf-dnsop-ns-revalidation-02
Published: 7 March 2022
Intended Status: Standards Track
Expires: 8 September 2022
Authors: S. Huque (Salesforce)
P. Vixie (Farsight Security)
R. Dolmans (NLnet Labs)

Delegation Revalidation by DNS Resolvers

Abstract

This document recommends improved DNS [RFC1034] [RFC1035] resolver behavior with respect to the processing of Name Server (NS) resource record sets (RRset) during iterative resolution. When following a referral response from an authoritative server to a child zone, DNS resolvers should explicitly query the authoritative NS RRset at the apex of the child zone and cache this in preference to the NS RRset on the parent side of the zone cut. Resolvers should also periodically revalidate the child delegation by re-quering the parent zone at the expiration of the TTL of the parent side NS RRset.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 8 September 2022.

Copyright Notice

Table of Contents

1. Introduction

RFC EDITOR: PLEASE REMOVE THIS PARAGRAPH BEFORE PUBLISHING: The source for this draft is maintained in GitHub at: https://github.com/shuque/ns-revalidation

This document recommends improved DNS resolver behavior with respect to the processing of NS record sets during iterative resolution. The first recommendation is that resolvers, when following a referral response from an authoritative server to a child zone, should explicitly query the authoritative NS RRset at the apex of the child zone and cache this in preference to the NS RRset on the parent side of the zone cut. The second recommendation is to revalidate the delegation by re-quering the parent zone at the expiration of the TTL of the parent side NS RRset.

2. Motivation

There is wide variability in the behavior of deployed DNS resolvers today with respect to how they process delegation records. Some of them prefer the parent NS set, some prefer the child, and for others, what they preferentially cache depends on the dynamic state of queries and responses they have processed. This document aims to bring more commonality and predictability by standardizing the behavior in a way that comports with the DNS protocol.

The delegation NS RRset at the bottom of the parent zone and the apex NS RRset in the child zone are unsynchronized in the DNS protocol. [RFC1034] Section 4.2.2 says "The administrators of both zones should insure that the NS and glue RRs which mark both sides

of the cut are consistent and remain so.". But for a variety of reasons they could not be. Officially, a child zone's apex NS RRset is authoritative and thus has a higher cache credibility than the parent's delegation NS RRset, which is non-authoritative glue ([RFC2181], Section 5.4.1. "Ranking data", and Section 6.1. "Zone authority"). Hence the NS RRset "below the zone cut" should immediately replace the parent's delegating NS RRset in cache when an iterative caching DNS resolver crosses a zone boundary. However, this can only happen if (1) the resolver receives the authoritative NS RRset in the Authority section of a response from the child zone, which is not mandatory, or (2) if the resolver explicitly issues an NS RRset query to the child zone as part of its iterative resolution algorithm. In the absence of this, it is possible for an iterative caching resolver to never learn the authoritative NS RRset for a zone, unless a downstream client of the resolver explicitly issues such an NS query, which is not something that normal enduser applications do, and thus cannot be relied upon to occur with any regularity.

Increasingly, there is a trend towards minimizing unnecessary data in DNS responses. Several popular DNS implementations default to such a configuration (see "minimal-responses" in BIND and NSD). So, they may never include the authoritative NS RRset in the Authority section of their responses.

A common reason that zone owners want to ensure that resolvers place the authoritative NS RRset preferentially in their cache is that the TTLs may differ between the parent and child side of the zone cut. Some DNS Top Level Domains (TLDs) only support long fixed TTLs in their delegation NS sets. In fact, the Extensible Provisioning Protocol (EPP) [RFC5731], that is often used by TLDs to configure delegation parameters has no provision to set the TTL. This inhibits a child zone owner's ability to make more rapid changes to their nameserver configuration using a shorter TTL, if resolvers have no systematic mechanism to observe and cache the child NS RRset.

A child zone's delegation still needs to be periodically revalidated at the parent to make sure that the parent zone has not legitimately re-delegated the zone to a different set of nameservers, or even removed the delegation. Otherwise, resolvers that refresh the TTL of a child NS RRset on subsequent queries or due to pre-fetching, may cling to those nameservers long after they have been re-delegated elsewhere. This leads to the second recommendation in this document, "Delegation Revalidation" - Resolvers should record the TTL of the parent's delegating NS RRset, and use it to trigger a revalidation action.

3. Upgrading NS RRset Credibility

* When a delegation response is received during iteration, a validation query should be sent in parallel with the resolution of the triggering query, to the delegated nameservers for the newly discovered zone cut. Note that validating resolvers today, when following a secure referral, already need to dispatch a query to the delegated nameservers for the DNSKEY RRset, so this validation query could be sent in parallel with that DNSKEY query.

* A validation query consists of a query for the child's apex NS RRset, sent to the newly discovered delegation's nameservers. Normal iterative logic applies to the processing of responses to validation queries, including storing the results in cache, trying the next server on SERVFAIL or timeout, and so on. Positive answers to this validation query will be cached with an authoritative data ranking. Successive queries directed to the same zone will be directed to the nameservers listed in the child's apex, due to the ranking of this answer. If the validation query fails, the parent NS RRset will remain the one with the highest ranking and will be used for successive queries.

* Some resolvers may choose to delay the response to the triggering query until both the triggering query and the validation query have been answered. In practice, we expect many implementations may answer the triggering query in advance of the validation query for performance reasons. An additional reason is that there are number of nameservers in the field that (incorrectly) fail to answer explicit queries for NS records, and thus the revalidation logic may need to be applied lazily and opportunistically to deal with them.

* If the resolver chooses to delay the response, and there are no nameserver names in common between the child's apex NS RRset and the parent's delegation NS RRset, then the responses received from forwarding the triggering query to the parent's delegated nameservers should be discarded after validation, and this query should be forwarded again to the child's apex nameservers.

4. Delegation Revalidation

The essence of this mechanism is re-validation of all delegation metadata that directly or indirectly supports an owner name in cache. This requires a cache to remember the delegated name server names for each zone cut as received from the parent (delegating) zone's name servers, and also the TTL of that NS RRset, and the TTL of the associated DS RRset (if seen).

A delegation under re-validation is called a "re-validation point" and is "still valid" if its parent zone's servers still respond to an in-zone question with a referral to the re-validation point, and if that referral overlaps with the previously cached referral by at least one name server name, and the DS RRset (if seen) overlaps the previously cached DS RRset (if also seen) by at least one delegated signer.

If the response is not a referral or refers to a different zone than before, then the shape of the delegation hierarchy has changed. If the response is a referral to the re-validation point but to a wholly novel NS RRset or a wholly novel DS RRset, then the authority for that zone has changed. For clarity, this includes transitions between empty and non-empty DS RRsets.

If the shape of the delegation hierarchy or the authority for a zone has been found to change, then no currently cached data whose owner names are at or below that re-validation point can be used. Such non-use can be by directed garbage collection or lazy generational garbage collection or some other method befitting the architecture of the cache. What matters is that the cache behave as though this data was removed.

Since re-validation can discover changes in the shape of the delegation hierarchy it is more efficient to re-validate from the top (root) downward (to the owner name) since an upper level re-validation may obviate lower level re-validations. What matters is that the supporting chain of delegations from the root to the owner name be demonstrably valid; further specifics are implementation details.

Re-validation is triggered when delegation meta-data has been cached for a period at most exceeding the delegating NS or DS (if seen) RRset TTL. If the corresponding child zone's apex NS RRset TTL is smaller than the delegating NS RRset TTL, revalidation should happen at that interval instead. However, resolvers should impose a sensitive minimum TTL floor they are willing to endure to avoid potential computational DoS attacks inflicted by zones with very short TTLs.

In normal operations this meta-data can be quickly re-validated with no further work. However, when re-delegation or take-down occurs, a re-validating cache will discover this within one delegation TTL period, allowing the rapid expulsion of old data from the cache.

5. IANA Considerations

This document includes no request to IANA.

6. Security Considerations

Upgrading NS RRset Credibility (Section 3) allows resolvers to cache and utilize the authoritative child apex NS RRset in preference to the non-authoriative parent NS RRset. However, it is important to implement the steps described in Delegation Revalidation (Section 4) at the expiration of the parent's delegating TTL. Otherwise, the operator of a malicious child zone, originally delegated to, but subsequently delegated away from, can cause resolvers that refresh TTLs on subsequent NS set queries, or that pre-fetch NS queries, to never learn of the redelegated zone. This problem has been seen in the wild [include reference to Ghost Domains paper here].

7. References

7.1. Normative References

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <https://www.rfc-editor.org/info/rfc1034>.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, DOI 10.17487/RFC2181, July 1997, <https://www.rfc-editor.org/info/rfc2181>.

7.2. Informative References

[I-D.vixie-dnsext-resimprove] Vixie, P., Joffe, R., and F. Neves, "Improvements to DNS Resolvers for Resiliency, Robustness, and Responsiveness", Work in Progress, Internet-Draft, draft-vixie-dnsext-resimprove-00, 23 June 2010, <https://datatracker.ietf.org/doc/html/draft-vixie-dnsext-resimprove-00>.

[I-D.wijngaards-dnsext-resolver-side-mitigation] Wijngaards, W., "Resolver side mitigations", Work in Progress, Internet-Draft, draft-wijngaards-dnsext-resolver-side-mitigation-01, 24 February 2009, <https://datatracker.ietf.org/doc/html/draft-wijngaards-dnsext-resolver-side-mitigation-01>.

[RFC5731] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Domain Name Mapping", STD 69, RFC 5731, DOI 10.17487/RFC5731, August 2009, <https://www.rfc-editor.org/info/rfc5731>.

Acknowledgements

Wouter Wijngaards proposed explicitly obtaining authoritative child NS data in [I-D.wijngaards-dnsext-resolver-side-mitigation]. This behavior has been implemented in the Unbound DNS resolver via the "harden-referral-path" option. The combination of child NS fetch and revalidating the child delegation was originally proposed in [I-D.vixie-dnsext-resimprove], by Vixie, Joffe, and Neves.

Authors' Addresses

Shumon Huque
Salesforce

Email: shuque@gmail.com

Paul Vixie
Farsight Security

Email: paul@redbarn.org

Ralph Dolmans
NLnet Labs

Email: ralph@nlnetlabs.nl