Network Working Group
Internet-Draft
Updates: 4034, 4035 (if approved)
Intended status: Standards Track
Expires: September 3, 2018

P. Hoffman
M. Larson
ICANN

March 02, 2018

Additional Method for Filling DNS Caches
draft-hl-dnsop-cache-filling-00

Abstract

DNS recursive resolvers do not always have access to the authoritative resolvers from which they need to get information. For example, when the DNS root servers are under DDoS attack, a recursive resolver may not be able to get an answer from any of the root servers. This document describes how resolvers can populate their caches with zone information, and keep their cache populated, using out-of-band mechanisms that do not rely on the DNS protocol. The protocol is primarily designed for the root zone, but can apply to any zone that wants to participate by publishing values to be cached.

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 September 3, 2018.

1. Introduction

As decscribed in [RFC1035], recursive resolvers keep a cache of resource records that they have already seen. Entries in that cache are removed when the TTL of the records reach zero. Filling the cache for the root zone begins with priming, as decribed in [RFC8109].

Some useful DNS protocol extensions rely on entries being in a resolver's cache. In specific, "aggressive NSEC" ([RFC8198]) reduces the need for a resolver to be in contact with an authoritative server because it uses information in its cache to prevent asking about zones that do not exist. The more entries that the cache has in it, the more effective a resolver can use aggressive NSEC.

This document describes a new method for resolvers to fill their caches with zone information, and keep their cache populated, using out-of-band mechanisms that do not rely on the DNS protocol. It introduces the following:

- The use cases for which this protocol might be useful
- A method for a resolver to get batched DNS entries that it will include in its cache
- A message format for carrying those batched DNS entries that is a trivial extension to the DNS wire format from [RFC1035]
- An optional method to find servers that deliver the batched entries for various zones

A primary design goal for this protocol is that if the protocol fails for any reason, a resolver still accesses authoritative servers just as it does today. That is, this protocol helps fill the resolvers cache so that it avoids talking to the authoritative server over DNS, but it isn't intended to preclude the resolver from doing so.

Note that this document operates differently than the method described in [RFC7706]. In that document, the resolver operator changes the source of its root information; in this document, the resolver can still get its information from the regular DNS

authoritative servers (such as the root servers), but can also fill its cache from sources other than the authoritative servers.

This document assumes that the resolver is validating responses with DNSSEC; it does not change anything about a resolver's processing of DNSSEC data.

1.1. Use Cases

The primary use case it to give resolver operators more consistent access to DNS data from authoritative servers, particularly for the root zone. The DNS root server system has experienced distributed denial of service (DDoS) attacks that have, from certain points on the network, made a majority of root servers unresponsive to some resolvers while under attack. If a resolver can extend the intervals between when it needs to reach an authoritative server, the resolver can provide more reliable access to its customers.

The second use case is to speed up answers from the recursive servers to their customers by having more data in the servers' caches. This is achieved by refilling the entries in a resolver's cache before they expire. This is primarily of value to resolvers using aggressive NSEC caching [RFC8198].

Although both of these use cases focus on the root zone of the DNS, they can apply to other zones as well. For example, some TLDs allow complete access to their zone data, and they might use the publication methodology described in this document. However, the method described here might not work well for zones where some entries have TTLs that are much shorter than the 1- and 2-day TTLs in the root zone.

1.2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119, BCP 14 [RFC2119] and indicate requirement levels for compliant CBOR implementations.

Throughout this document, the term "DNSSEC" means the protocols defined in [RFC4033], [RFC4034], [RFC4035] and all RFCs that update those three.

A great deal of DNS-related terminology can be found in [I-D.ietf-dnsop-terminology-bis].

2. Protocol Overview

A resolver that wants to fill its cache for a zone refers to one or more URLs of the service that provides the cache-filling information. The URLs might come out of band (for example, in the resolver's configuration could specify URLs for a zone), or can be found with the new CACHE-FILL RRtype (described in Section 3).

Once it has the URL, the resolver fetches a set of entries for its cache. The format described in Section 4. Basically, the format is a series of DNS queries and responses encoded in DNS wire format. The resolver processes the request-response pairs in the retrived data just as it would any request-response pairs it sees from clients (such as checking for and discarding records that should not be in the response and performing DNSSEC validation). The new data is interpreted as queries to and replies from the specified zone's authoritative name server. It then adds the responses to its cache, acting as if each response is the last response in a chain of queries to the authoritative server.

3. Discovery of Servers That Provide Cache-Filling Information

A resolver contacts the server that has the cache-filling material by resolving a URL. That URL might come out of band, or might be discovered by sending a query for the zone with the CACHE-FILL RRtype. A zone might have many records in the CACHE-FILL RRset for alternate URLs.

Using out-of-band URLs is acceptable if the resolver operator believes that the source would give the same answers as the authoritative server would have given. If a resolver wants to use this protocol for initial root zone priming, the URL would inherently need to be out-of-band (because the resolver doesn't yet have any ability to resolve names) and the URL MUST have an IP address as the host portion of the URL.

3.1. The CACHE-FILL RRtype

The CACHE-FILL resource record is used to store a list of URLs.

The Type value for the CACHE-FILL RR is TBD1. The CACHE-FILL RR is defined only for the IN class. The CACHE-FILL RR has no special Time-to-Live (TTL) requirements.

The RDATA for an CACHE-FILL RR is a text string that holds a URL. The presentation format has the URL in a quoted string.

The CACHE-FILL RR for a zone is located at the zone apex. For example, the record below specifies a CACHE-FILL RR for the root zone because it is located at the root zone's apex (the root node of the DNS):

```
. 86400 IN CACHE-FILL "http://cdn.example.com/root-cache-fill"
```

See Section 7 for the registration of RRtype TBD1.

4. Message Format

The format of the cache-filling data is a series of DNS request-response message pairs in wire format as defined in [RFC1035]. Each request and response message in the data MUST be formatted as a TCP DNS message with the two-octet header.

HTTP servers that are providing this data SHOULD use the media type of application/TBD2. See Section 7 for the registration of this media type.

5. Filling the Cache Using the Data Received

*** Warning that the origin must be specified for out-of-band URLs to know the context to interpret the data retrieved.

*** Determining how often to fetch the cache file file. There are several options and more text is needed here. One option would be to fetch the file when first RRset from that zone expires, i.e., the RRset with the lowest TTL. Another option is to use the zone's SOA refresh timer to determine how often to refetch the cache fill file.

5.1. Parsing the Received Data

The resolver parses the data received as follows:

1. Take the first two octets as the message length, then read that many octets of data. Verify that this is a properly-formatted DNS request.

2. Take the next two octets as the message length, then read that many octets of data. Verify that this is a properly-formatted DNS response, and that the Question section of this response matches the Question section of the immediately-preceding message.

3. Process this pair of messages; see Section 5.2.

4. Repeat if there is more octets in the data.

If there is an error in parsing, the resolver must stop processing and not use any data starting at the point where the error occurred.

5.2. Processing Parsed Messages

After a request-response pair is parsed, the data MUST be treated as if the request had come from a client and the response had come from the authoritative server for the zone. For example, if the resolver is normally performing DNSSEC validation on responses to queries to authoritative servers, the resolver MUST validate the processed data using the same rules.

The data MUST be interpreted as queries to and replies from the specified zone's authoritative name server. The resolver MUST use the normal rules (such as those from [RFC2181] for data in the Authoritative and Additional sections in responses) before putting that data into the cache. For responses from the root zone, all data is in-bailiwick, but for lower zones, the resolver MUST only fill its cache with data it would have accepted from those zones, i.e., data that is in-bailwick for that zone.

The resolver SHOULD process the data immediately after it is received so that the TTLs are treated as if they had just been received from an authoritative server. The exception to this rule would be if a resolver is using a stored version of data as described in Section 5.4

5.3. Using the Cache-Filling to Initially Prime Root Server Information

*** This will require that resolvers come with a second "root hints" file that has URLs, and those URLs will have to have IP addresses. The servers at those addresses need to only use TLS extensions that do not require the client to access the DNS. This is possible, but tricky. Maybe punt this to a separate draft after this one completes?

5.4. Using the Cache-Filling to Re-Prime Root Server Information

A resolver following this protocol for the root zone could keep the latest copy of the root zone data it received on disk and use that data to fully prime its cache after a restart. This can be useful if the root server system is not available when the resolver starts up. Note, however, that it is possible that some of the data would be out of date and incorrect because the root zone has changed, so the resolver SHOULD immediatly pull a new version of the cache-filling data after re-priming.

6. Security Considerations

If a malicious actor can hijack the source of the cache-filling data, they can cause that data to be used by everyone who accesses that source. This will not affect signed records for resolvers who are validating with DNSSEC, but will affect unsigned records, and all resolvers that are not validating with DNSSEC.

HTTPS URLs for the cache-filling data give privacy and prevent modification of unsigned data such as glue records. Resolvers SHOULD prefer HTTPS URLs to HTTP URLs. It is possible (but unlikely) that URLs with different schemes (such as for FTP) will be used, and these URLs could have very different security properties than HTTPs URLs.

*** There are certainly going to be some more.

7. IANA Considerations

*** Registration for CACHE-FILL RRtype (TBD1)

*** New media type for responses (TBD2)

8. References

8.1. Normative References

[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>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[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>.

8.2. Informative References

[I-D.ietf-dnsop-terminology-bis] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", draft-ietf-dnsop-terminology-bis-08 (work in progress), November 2017.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <https://www.rfc-editor.org/info/rfc4033>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <https://www.rfc-editor.org/info/rfc4034>.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005, <https://www.rfc-editor.org/info/rfc4035>.

[RFC7706] Kumari, W. and P. Hoffman, "Decreasing Access Time to Root Servers by Running One on Loopback", RFC 7706, DOI 10.17487/RFC7706, November 2015, <https://www.rfc-editor.org/info/rfc7706>.

[RFC8109] Koch, P., Larson, M., and P. Hoffman, "Initializing a DNS Resolver with Priming Queries", BCP 209, RFC 8109, DOI 10.17487/RFC8109, March 2017, <https://www.rfc-editor.org/info/rfc8109>.

[RFC8198] Fujiwara, K., Kato, A., and W. Kumari, "Aggressive Use of DNSSEC-Validated Cache", RFC 8198, DOI 10.17487/RFC8198, July 2017, <https://www.rfc-editor.org/info/rfc8198>.

Appendix A. Design Rationale

A.1. Design Rationale for the Message Format

The paired message format was used instead of master file format because resolvers already know how to process request-response pairs, but don't currently stuff sets of records into their cache.

A.2. Design Rationale for using URLs of Data Instead of AXFR of the Zone

*** CDNs have already optimized for content delivery, so building a scalable AXFR service for the root zone would not be required.

*** Resolvers adding HTTPS fetching is probably less work than adding full AXFR logic.

Authors' Addresses

Paul Hoffman
ICANN

Email: paul.hoffman@icann.org

Matt Larson
ICANN

Email: matt.larson@icann.org