Workgroup: Network Working Group
Internet-Draft: draft-klh-dnsop-rfc8109bis-05
Obsoletes: 8109 (if approved)
Published: 16 November 2022
Intended Status: Best Current Practice
Expires: 20 May 2023
Authors: P. Koch DENIC eG
M. Larson ICANN
P. Hoffman ICANN

# Initializing a DNS Resolver with Priming Queries

## Abstract

This document describes the queries that a DNS resolver should emit to initialize its cache. The result is that the resolver gets both a current NS Resource Record Set (RRset) for the root zone and the necessary address information for reaching the root servers.

This document, when published, obsoletes RFC 8109.

## 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 20 May 2023.

**Table of Contents**



## 1. Introduction

Recursive DNS resolvers need a starting point to resolve queries. [RFC1034] describes a common scenario for recursive resolvers: they begin with an empty cache and some configuration for finding the names and addresses of the DNS root servers. [RFC1034] describes that configuration as a list of servers that will give authoritative answers to queries about the root. This has become a common implementation choice for recursive resolvers, and is the topic of this document.

This document describes the steps needed for this common implementation choice. Note that this is not the only way to start a recursive name server with an empty cache, but it is the only one described in [RFC1034]. Some implementers have chosen other directions, some of which work well and others of which fail (sometimes disastrously) under different conditions. For example, an implementation that only gets the addresses of the root name servers from configuration, not from the DNS as described in this document, will have stale data that could cause slower resolution.

This document only deals with recursive name servers (recursive resolvers, resolvers) for the IN class.

### 1.1. Changes from RFC 8109

This document obsoletes [RFC8109]. The significant changes from RFC 8109 are:

- Added section on the content of priming information.
- Added paragraph about no expectation that the TC bit in responses will be set.
- Changed "man-in-the-middle" to "machine-in-the-middle" to be both less sexist and more technically accurate.
- Clarified that there are other effects of machine-in-the-middle attacks.
- Clarified language for root server domain names as "root server identifiers".
- Added informative references to RSSAC documents.
- Added short discussion about this document and private DNS.
- Future changes noted in the current text with [[ text like this ]].

### 1.2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

See [RSSAC026v2] for terminology that relates to the root server system.

## 2. Description of Priming

Priming is the act of finding the list of root servers from a configuration that lists some or all of the purported IP addresses of some or all of those root servers. In priming, a recursive resolver starts with no cached information about the root servers, and finishes with a full list of their names and their addresses in its cache.

Priming is described in Sections 5.3.2 and 5.3.3 of [RFC1034]. The scenario used in that description, that of a recursive server that is also authoritative, is no longer as common.

The configured list of IP addresses for the root servers usually comes from the vendor or distributor of the recursive server software. This list is usually correct and complete when shipped, but may become out of date over time.

The domain names for the root servers are called the "root server identifiers". This list has been stable since 1997, but the IPv4 and IPv6 addresses for the root server identifiers sometimes change, Research shows that after those addresses change, some resolvers never get the new addresses. Therefore, it is important that resolvers be able to cope with change, even without relying upon configuration updates to be applied by their operator. Root server identifier and address changes are the main reasons that resolvers need to do priming instead of just going from a configured list to get a full and accurate list of root servers.

See [RSSAC023v2]for a history of the root server system.

Although this document is targeted at the global DNS, it also could apply to a private DNS as well. These terms are defined in [RFC8499].

### 2.1. Content of Priming Information

As described above, the configuration for priming is a list of IP addresses. The priming information in software may be in any format that gives he software the addresses associated with at least some of the root server identifiers.

Some software has configuration that also contains the root server identifiers, sometimes as comments and sometimes as data consumed by the software. For example, IANA's "Root Hints File" at <https://www.internic.net/domain/named.root> is derived directly from the root zone and contains all of the addresses of the root server identifiers found in the root zone, It is in DNS master file format, and includes the root server identifiers. Although there is no harm to adding such information, it is not useful in the root priming process.

## 3. Priming Queries

A priming query is a DNS query used to get the root server information in a resolver. It has a QNAME of ".", a QTYPE of NS, and a QCLASS of IN; it is sent to one of the addresses in the configuration for the recursive resolver. The priming query can be sent over either UDP or TCP. If the query is sent over UDP, the source port SHOULD be randomly selected (see [RFC5452]). The Recursion Desired (RD) bit MAY be set to 0 or 1, although the meaning of it being set to 1 is undefined for priming queries.

The recursive resolver SHOULD use EDNS0 [RFC6891] for priming queries and SHOULD announce and handle a reassembly size of at least 1024 octets [RFC3226]. Doing so allows responses that cover the size of a full priming response (see Section 4.2) for the current set of root servers. See Section 3.3 for discussion of setting the DNSSEC OK (DO) bit (defined in [RFC4033]).

### 3.1. Repeating Priming Queries

The recursive resolver SHOULD send a priming query only when it is needed, such as when the resolver starts with an empty cache and when the NS RRset for the root zone has expired. Because the NS records for the root are not special, the recursive resolver expires those NS records according to their TTL values. (Note that a recursive resolver MAY pre-fetch the NS RRset before it expires.)

[[ Need to discuss: when pre-fetching, does the resolver send the queries to the addresses associated with the . / NS RRset in the cache, or does the resolver go back to the addresses in the configuration? ]]

If a priming query does not get a response, the recursive resolver needs to retry the query with a different target address from the configuration.

### 3.2. Target Selection

In order to spread the load across all the root server identifiers, the recursive resolver SHOULD select the target for a priming query randomly from the list of addresses. The recursive resolver might choose either IPv4 or IPv6 addresses based on its knowledge of whether the system on which it is running has adequate connectivity on either type of address.

Note that this recommended method is not the only way to choose from the list in a recursive resolver's configuration. Two other common methods include picking the first from the list, and remembering which address in the list gave the fastest response earlier and using that one. There are probably other methods in use today. However, the random method listed above SHOULD be used for priming.

### 3.3. DNSSEC with Priming Queries

[[ This section talks about sending the DO bit, but does not actually talk about validating the response to the priming query. This became important after the root KSK rollover in 2018 because some resolvers apparently were validating and only had the old KSK, but were still sending RFC 8145 telemetry even after failing to validate their priming response. ]]

The resolver MAY set the DNSSEC OK (DO) bit. At the time of publication, there is little use to performing DNSSEC validation on the priming query. Currently, all root name server names end in "root-servers.net" and the AAAA and A RRsets for the root server names reside in the "root-servers.net" zone. All root servers are also authoritative for this zone, allowing priming responses to include the appropriate root name server A and AAAA RRsets. But, because the "root-servers.net" zone is not currently signed, these RRsets cannot be validated.

A machine-in-the-middle attack on the priming query could direct a resolver to a rogue root name server. Note, however, that a validating resolver will not accept responses from rogue root name servers if they are different from the real responses because the resolver has a trust anchor for the root and the answers from the root are signed. Thus, if there is a machine-in-the-middle attack on the priming query, the results for a validating resolver could be a denial of service, or the attacker seeing queries while returning good answers, but not the resolver's accepting the bad responses.

If the "root-servers.net" zone is later signed, or if the root servers are named in a different zone and that zone is signed, having DNSSEC validation for the priming queries might be valuable.

## 4. Priming Responses

A priming query is a normal DNS query. Thus, a root name server cannot distinguish a priming query from any other query for the root NS RRset. Thus, the root server's response will also be a normal DNS response.

### 4.1. Expected Properties of the Priming Response

The priming response is expected to have an RCODE of NOERROR, and to have the Authoritative Answer (AA) bit set. Also, it is expected to have an NS RRset in the Answer section (because the NS RRset originates from the root zone), and an empty Authority section (because the NS RRset already appears in the Answer section). There will also be an Additional section with A and/or AAAA RRsets for the root name servers pointed at by the NS RRset.

Resolver software SHOULD treat the response to the priming query as a normal DNS response, just as it would use any other data fed to its cache. Resolver software SHOULD NOT expect exactly 13 NS RRs because, historically, some root servers have returned fewer.

### 4.2. Completeness of the Response

There are currently 13 root servers. All have one IPv4 address and one IPv6 address. Not even counting the NS RRset, the combined size

of all the A and AAAA RRsets exceeds the original 512-octet payload limit from [RFC1035].

In the event of a response where the Additional section omits certain root server address information, re-issuing of the priming query does not help with those root name servers that respond with a fixed order of addresses in the Additional section. Instead, the recursive resolver needs to issue direct queries for A and AAAA RRsets for the remaining names. Currently, these RRsets would be authoritatively available from the root name servers.

If the Additional section is truncated, there is no expectation that the TC bit in the response will be set to 1. At the time that this document is written, many of the root servers are not setting the TC bit on responses with a truncated Additional section.

## 5. Post-Priming Strategies

[[ Describe some common post-priming strategies for picking which RSI to use for queries sent to the root, such as "always use the fastest", "create buckets of fastness and pick randomly in the buckets", and others. ]]

## 6. Security Considerations

Spoofing a response to a priming query can be used to redirect all of the queries originating from a victim recursive resolver to one or more servers for the attacker. Until the responses to priming queries are protected with DNSSEC, there is no definitive way to prevent such redirection.

An on-path attacker who sees a priming query coming from a resolver can inject false answers before a root server can give correct answers. If the attacker's answers are accepted, this can set up the ability to give further false answers for future queries to the resolver. False answers for root servers are more dangerous than, say, false answers for Top-Level Domains (TLDs), because the root is the highest node of the DNS. See Section 3.3 for more discussion.

In both of the scenarios above, a validating resolver will be able to detect the attack if its chain of queries comes to a zone that is signed, but not for those that are unsigned.

## 7. IANA Considerations

This document does not require any IANA actions.

## 8. References

### 8.1. Normative References

**[RFC1034]** Mockapetris, P. and RFC Publisher, "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. and RFC Publisher, "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>.

**[RFC3226]** Gudmundsson, O. and RFC Publisher, "DNSSEC and IPv6 A6 aware server/resolver message size requirements", RFC 3226, DOI 10.17487/RFC3226, December 2001, <https://www.rfc-editor.org/info/rfc3226>.

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

**[RFC5452]** Hubert, A., van Mook, R., and RFC Publisher, "Measures for Making DNS More Resilient against Forged Answers", RFC 5452, DOI 10.17487/RFC5452, January 2009, <https://www.rfc-editor.org/info/rfc5452>.

**[RFC6891]** Damas, J., Graff, M., Vixie, P., and RFC Publisher, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <https://www.rfc-editor.org/info/rfc6891>.

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

**[RFC8174]** Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

**[RFC8499]** Hoffman, P., Sullivan, A., Fujiwara, K., and RFC Publisher, "DNS Terminology", BCP 219, RFC 8499, DOI 10.17487/RFC8499, January 2019, <https://www.rfc-editor.org/info/rfc8499>.

### 8.2. Informative References

**[RSSAC023v2]** "History of the Root Server System", 2016, <https://www.icann.org/en/system/files/files/rssac-023-17jun20-en.pdf>.

**[RSSAC026v2]** "RSSAC Lexicon", 2020, <https://www.icann.org/en/system/files/files/rssac-026-lexicon-12mar20-en.pdf>.

## Appendix A. Acknowledgements

RFC 8109 was the product of the DNSOP WG and benefitted from the reviews done there.

## Authors' Addresses

Peter Koch
DENIC eG
Kaiserstrasse 75-77
60329 Frankfurt
Germany

Phone: +49 69 27235 0
Email: pk@DENIC.DE

Matt Larson
ICANN

Email: matt.larson@icann.org

Paul Hoffman
ICANN

Email: paul.hoffman@icann.org