# Accommodating IP Version 6 Address Resource Records for the Root of the Domain Name System


A Joint Report from the ICANN
Security and Stability Advisory and
Root Server System Advisory Committees

SAC018 2007

Version 1.0

## About the Security and Stability Advisory Committee

The Security and Stability Advisory Committee (SSAC) is an advisory committee to the Internet Corporation for Assigned Names and Numbers (ICANN). The Committee's purpose is to offer independent advice to the ICANN board, the ICANN staff and the various ICANN supporting organizations, councils and committees as well as to the technical community at large on matters relating to the security and integrity of the Internet's naming and address allocation systems. The Committee has no official authority to regulate, enforce or adjudicate. Those functions belong to others. The advice offered by the Committee should be evaluated on its merits, not on the status of the Committee or its members.

## About the Root Server System Advisory Committee

The Root Server System Advisory Committee (RSSAC) is an advisory committee to ICANN. The Committee's purpose to advise the Board about the operation of the root name servers of the domain name system. Specifically, the committee provides advice on the operational requirements of root name servers, including host hardware capacities, operating systems and name server software versions, network connectivity and physical environment. The Committee also examines and advises on the security aspects of the root name server system, and reviews the number, location, and distribution of root name servers considering the total system performance, robustness, and reliability.

### About this Report

This report was prepared by the SSAC Fellow, Dave Piscitello, under the direction of the joint committees and represents output from the committees as a whole. The Appendix contains the current list of members and contributors to this report.

## Executive Summary

This Report considers the issues related to the inclusion of the IPv6 addresses for the root level of the DNS. IPv6 addresses are already included for Top Level Domain Name Servers in the root zone file, and the operators of a number of root name servers have assigned IPv6 addresses to their servers. These addresses are not included in the root hints file and the root zone at this time. Thus IPv6 addresses of root name servers are not returned in responses to DNS queries sent by recursive name servers.

To enable name resolution, resolvers are pre-configured with the addresses of at least one root name server. Commonly called "hints", recursive name servers initially rely on these addresses to provide recursive name service. Many recursive name servers also perform a bootstrap process called *priming*. Priming ensures that a recursive name server always starts operation with the most up-to-date list of root name servers.

The User Datagram Protocol (UDP) serves as the transport for priming messages. RFC 1035, Domain Names Implementation and Specification, specifies a 512 byte maximum UDP-encapsulated DNS message size. Adding the IPv6 address information for more than two root name servers to the root hints file and to the root zone will increase the size of the DNS priming response so that it exceeds this maximum. Ultimately, when all 13 root name servers assign IPv6 addresses, the priming response will increase in size to 811 bytes. This imposes additional conditions for the successful completion of a priming exchange that do not exist today:

- Intermediate systems that are situated between recursive name servers and root name servers must be able to process DNS messages containing IPv6 addresses.
- Resolvers must use DNS Extensions to notify root name servers that they are able to process DNS response messages larger than the 512 byte maximum UDP-encapsulated DNS message size specified in RFC 1035.
- Intermediate systems must be configured to forward UDP-encapsulated DNS responses that exceed the 512 byte maximum DNS message size specified in RFC 1035.

In this report, the ICANN Root Server System Advisory and Security and Stability Advisory Committees examine the problems that might arise if IP Version 6 (IPv6) host address resource records of root name servers were added to the root hints and root zone file for the DNS. We describe and report the results of testing performed by committee members and the community at large, including recursive name server operators as well as commercial vendors of security systems and DNS name server products, to determine the extent to which these problems are likely to be encountered. The test results figure prominently in the recommendations we propose to ICANN and IANA.

We conclude the Report with a roadmap the community can follow to assure that the inclusion of AAAA records in the root hints file and DNS priming responses from root name servers has minimum impact and maximum benefit.

# 1. Introduction

Many TLD name servers have IP version 6 (IPv6) addresses and provide domain name service for IPv6 today. A number of root name server operators have assigned IPv6 addresses to their systems as well. To date, however, the IPv6 addresses of root name servers are not included in the IANA-maintained root hints and root zone files. A lack of a clear understanding of how the inclusion of these addresses might affect name service has to date prevented IANA from including these addresses in two critical root-level resources: the root hints file and the root zone. As a result, root name servers do not return IPv6 addresses of root name servers in response to DNS queries they receive from recursive name servers.

In this report, the ICANN Root Server System Advisory and Security and Stability Advisory Committees examine the problems that might arise if IPv6 host address resource records of root name servers were added to the root hints and root zone files for the DNS. We report the results of testing performed by committee members and the community at large to determine the extent to which these problems are likely to be encountered. The test results figure prominently in the recommendations we propose to ICANN and IANA. We conclude the report with a recommended course of action for ICANN and IANA to include IPv6 addresses of root name servers in the root level of the DNS.

The report is organized as follows:

**Section 2** describes how adding IPv6 addresses at the root of the DNS affects the root hints file and the priming exchange

**Section 3** considers the strengths, weaknesses and issues of the alternatives proposed in Section 2.

In **Section 4**, SSAC and RSSAC present their findings.

In **Section 5**, SSAC and RSSAC provide a roadmap the community can follow to assure that the inclusion of IPv6 address records in the root hints file and DNS priming responses from root name servers has minimum impact and maximum benefit.

This report discusses the operation of the DNS in considerable technical detail. Appendix A provides background material covering the terminology, nomenclature, and operation of the Domain Name System. In particular, this appendix provides detailed descriptions of the composition, use and administration of the root hints and root zone files, and of DNS protocol exchanges between root name servers and recursive name servers that are essential to assuring accurate name resolution. Readers who are unfamiliar with these concepts are strongly encouraged to read Appendix A and complementing Appendices before proceeding to Section 2.

# 2. Inclusion of IPv6 addresses at the Root of the DNS

In this section, we describe how adding IPv6 addresses at the root of the DNS affects the root hints file and the priming exchange.

## *Adding AAAA Records to Root Hints*

A recursive name server's iterative resolver must know the IP address of at least one root name server to function properly. Commonly, name server software provides sufficient configuration information during installation to assure that a host connected to the Internet can query a root name server by including a *hints* file. The IANA maintains the authoritative root hints file.

The existing procedures for publishing root hints need not be changed to add AAAA addresses of root name servers in the files published at `ftp://ftp.internic.net/domain/`.

When the root hints file is changed, it is expected that all resolvers and name servers will use one of the update methods identified in Appendix A in the section entitled *Updating and Maintaining Root Hints Files.*

## *Adding AAAA Records to Priming Exchange*

Before adding AAAA records to the priming exchange, we consider ways to avoid or minimize the impact or adverse affects such changes may have on deployed systems:

- For performance and resiliency purposes, it is desirable that root name servers continue to include the A records for all thirteen root name servers.

- Root name servers should return the same DNS priming response irrespective of which IP transport is used (v4 or v6).

- Situations where a large DNS response message forces root name servers to mark the message as truncated and thereby cause a resolver to resend the priming query using TCP should be avoided. Root name servers should not be burdened with the additional processing associated with establishing TCP connections for priming exchanges.

Thus, the committees considered the following options:

1) Include as many AAAA records of root name server addresses as will fit into the Additional Section of a UDP-encapsulated DNS message of 512 bytes in priming responses. Each AAAA record will occupy 28 bytes in the Additional section. Thus a DNS Priming Response would be composed in the following manner:

| DNS Priming Response Message (IPv4 and IPv6) | # Bytes |
|---|---|
| Required Headers:<br>• Transaction ID, Flags, Questions, Answer RR count, Authority RR count, Additional RR count | 12 |
| Query<br>• Name ".", Type NS, Class INET | 5 |
| Answers:<br>• First answer contains name, type, class, TTL and Data length (value 20), plus the Fully Qualified Domain Name (FQDN) of a root name server (e.g., H.ROOT-SERVERS.NET)<br>• Second through 13th answers contain name, type, class, TTL and Data length (value 4), plus the label of a root name server (e.g., G, F, E…) | 31<br>180 |
| Additional Records<br>• Each of the 13 A records in the Additional section contains name, type, class, TTL and Data length (value 4) and an 4-byte IPv4 address and occupies 16 bytes<br>(13 records times 16 bytes per record equals 208 bytes) | 208 |
| Additional Records<br>• Two AAAA records in the Additional section contain name, type, class, TTL and Data length (value 16) and a 16-byte IPv6 address and occupies 28 bytes<br>(2 records times 28 bytes per record equals 56 bytes) | 56 |
| Total length | 492 |

2) Plan for the eventual inclusion of AAAA records of all thirteen root name servers in the Additional Section of priming response messages. Again, each AAAA record is 28 bytes. An options (type OPT) section of 11 bytes must be present to indicate that EDNS0 has been offered by the querying name server. The DNS Priming Response is thus composed in the following manner:

| DNS Priming Response Message (IPv4 and IPv6) | # Bytes |
|---|---|
| Required Headers:<br>• Transaction ID, Flags, Questions, Answer RR count, Authority RR count, Additional RR count | 12 |
| Query<br>• Name ".", Type NS, Class INET | 5 |
| Answers:<br>• First answer contains name, type, class, TTL and Data length (value 20), plus the Fully Qualified Domain Name (FQDN)of a root name server (e.g., H.ROOT-SERVERS.NET)<br>• Second through 13th answers contain name, type, class, TTL and Data length (value 4), plus the label of a root name server (e.g., G, F, E…) | 31<br>180 |
| Additional Records<br>• Each of the 13 A records in the Additional section contains name, type, class, TTL and Data length (value 4) and an 4-byte IPv4 address and occupies 16 bytes (13 records x 16 bytes/record) | 208 |
| Additional Records<br>• 13 AAAA records in the Additional section contain name, type, class, TTL and Data length (value 16) and a 16-byte IPv6 address and occupies 28 bytes (13 records x 28 bytes/record) | 364 |
| EDNSO Option (OPT) | 11 |
| Total length | 811 |

# 3. Discussion

In this section, SSAC and RSSAC consider the strengths, weaknesses and issues of each alternative proposed in Section 2, *Inclusion of IPv6 addresses at the Root of the DNS.*

## *Root Name Server Considerations*

Under alternative (1), root name servers return sufficient AAAA information in a DNS priming response message to bootstrap IPv6 name service. The advantage to this alternative is that implementations that have not yet implemented EDNS0 will continue to operate without the possibility of DNS response message truncation, providing they are able to process DNS response messages containing AAAA records correctly.

Alternative (1) has certain disadvantages:

- The priming response only identifies two of thirteen root name servers and thus provides minimal resiliency for all users who need to prime name servers with IPv6 addresses.

- Two of the thirteen root name servers to be included in the DNS priming response would need to be chosen.

Alternative (2) has no such disadvantages. Root name servers can eventually include the A and AAAA records of all root name servers that are currently assigned IPv6 addresses. Since this is the desired end state, this Report will focus on the issues in achieving this objective.

Currently, root name servers use the BIND 8, BIND 9, and NSD name server software packages. Root name servers currently running BIND 9 and NSD can be configured to build a DNS priming response message as illustrated for alternative (2). BIND version 8 composes the Additional section in a slightly different manner. Specifically, BIND 8 will return an A record of a root name server, followed by an AAAA record of that same name server. Simply put, the DNS priming response returned by a BIND 8 implementation would return more AAAA records than a BIND 9 or NSD implementation and fewer A records but a sufficient number of both to allow the bootstrapping of IPv4 and IPv6 name service to complete.

## *Resolver Considerations*

In this section, we consider several issues related to choosing alternative (2).

*Is EDNS0 support among resolvers in production networks prevalent enough to choose a priming response alternative that cannot fit within the maximum DNS message size specified in RFC 1035?*

The priming response exceeds the maximum DNS message size recommended in RFC 1035 when more than two type AAAA resource records are added to the Additional section. To achieve the desired end condition of having all root name servers return the A and AAAA records of all root name servers in the priming response message,

1) Resolvers must be able to process DNS priming message responses containing AAAA records and must be able to reassemble IP packets.

2) Resolvers that do not support EDNS0 and resolvers that support EDNS0 but advertise a receive buffer of less than 811 bytes should use whatever AAAA information root name servers return to bootstrap IPv6 name service. See Appendix A, *DNS Message Composition and Size Considerations*.

3) Resolvers that support EDNS0 should advertise a receive buffer of at least 800 bytes. (Note: data collected by RIPE-NCC suggest that 99% of EDNS0-capable resolver installations advertise 1024 or larger receive buffers, See Table 2 and Figure 2 of [1]).

4) Resolvers should retry the priming response without advertising EDNS0 if they do not receive a DNS response message within a timeout period.

5) If resolvers do not receive a priming response message, they use whatever "hints" they have.

To approximate the potential impact, members of the committee informally tested several resolver implementations by composing and issuing Type NS queries to Top Level Domains that currently return A and AAAA records. In this case, the queries used the EDNS0 option to advertise a buffer size of 4096 bytes. The sizes of the responses ranged from 521 bytes to 730 bytes. We observed that resolvers provided with popular operating systems (Windows Server 2000/2003, Mac OSX, various Linux builds including Fedora and Red Hat) are able to process UDP-encapsulated DNS response messages that are longer than 512 bytes.

*Will the presence of AAAA records in the DNS priming response adversely affect resolver implementations used today in IPv4-only production networks?*

For resolvers, three adverse conditions may result from this action:

1. A resolver that is not IPv6-aware may not operate correctly when it receives a priming response that contains AAAA records from a root name server.

2. A resolver that is not IPv6-aware may ignore AAAA records in a priming response but otherwise behave properly.

3. A resolver that is IPv6-aware but has not been configured to use IPv6 will ignore priming messages containing AAAA records but otherwise process a priming response correctly.

To approximate the potential impact, members of the committee informally tested several resolver implementations by composing and issuing type NS queries that currently return A and AAAA records of TLD name servers (UA, FR, JP). The size of the response messages ranged from 208 to 439 bytes. From the results, we observe that resolvers provided with commonly used operating systems (Windows Server 2000/2003, Mac OSX, various Linux builds including Fedora and Red Hat) are able to process DNS priming responses, and use and cache the AAAA records. [Note: we assume that the same logic used to process a type NS response is used to process a priming response.]

*Is the sequencing of records in the Additional data in the DNS priming response important? Specifically, is it necessary to put all Type A records before any Type AAAA records in the Additional section of the priming response?*

Members of the joint committees speculate that some DNS implementations may be sensitive to the order that Type A and AAAA records are encoded in the Additional Section; specifically, some implementations may expect Type A resource records to be encoded immediately following the Answers Section (as illustrated in Section 2, *Inclusion of IPv6 addresses at the Root of the DNS*). It seems appropriate to accommodate for this possibility by specifying that all Additional records containing Type A resource records precede Additional records containing Type AAAA resource records.

The informal tests of resolver implementations imitate part of the resolver bootstrap process. These informal tests were valuable, but the committees sought broader and more formal testing from DNS server vendors, developers and the user community at large. These are described in the following section.

## Testing Iterative Resolvers for AAAA and EDNS0 Support

The complete name server bootstrap process must be tested to verify that changes at the root level of DNS service do not adversely affect production name service. Tests must verify that an implementation:

- Use the root name server information in the DNS response message without failing when it is configured with a hints file containing type AAAA resource records.
- Perform the priming exchange over UDP, which involves sending a DNS query for type NS for the root (".") to one or more of the root name servers identified in the local copy of the hints file.
- Process the UDP-encapsulated DNS response message from a root name server.
- Use the information in DNS response message to perform iterative name resolution.

Ideally, the test response contains type A and AAAA resource records of the authoritative root name servers and is larger than the 512-byte maximum UDP DNS message size specified in RFC 1035. Several root name server operators have volunteered to operate test name servers for this exercise. These servers have been configured to be authoritative for "test" root and root-servers.net zones that contain both type A and AAAA resource records for the authoritative root name servers.

RSSAC and SSAC have solicited Internet community participation to test whether iterative resolvers can be configured with a hints file containing both type A and AAAA resource records and also whether iterative resolvers are able to process priming responses containing IPv6 (AAAA) resource records and priming responses greater than 512 bytes (See SAC017, [12]). The results reported to the ICANN SSAC Fellow when this report was published are reproduced in Appendix D.

The results indicate that "modern day" (post 2000) DNS products used as recursive name servers are able to bootstrap when AAAA resource records are present in the root hints or equivalent configuration data and that these name servers will function properly if they receive a priming response greater than 512 bytes containing AAAA resource records. We conclude that very few recursive name servers used in production today will be adversely affected by the inclusion of IPv6 addresses for root name servers in the root hints and root zone files.

## *Intermediate System Considerations*

Anecdotal reports suggest that certain intermediate devices used in production networks (e.g., security systems such as an Internet firewall) inspect DNS messages for security purposes may be adversely affected by the inclusion of AAAA records in the DNS priming response messages. Again, three adverse conditions may result from this action:

1. The security system is not IPv6-aware and by default blocks DNS messages that contain resource records that do not conform to RFC 1034/1035.

2. The security system is IPv6-aware but the default configuration setting of the system is to block DNS messages that contain resource records that do not strictly conform to RFC 1034/1035.

3. The security policy enforced by an organization currently blocks DNS messages that contain resource records that do not conform to RFC 1034/1035.

To better understand these situations, first consider the behavior of a security system, e.g., an Internet firewall or software firewall executing on a host that has not implemented IPv6. When this security system receives an IPv6 datagram used to transport a priming message over an Ethernet segment, it will inspect the EtherType field of Ethernet header, extract the value encoded (for IPv6, 0x86DD), and compare this value against the set of "allowed EtherTypes" in its security policy database. Since IPv6 is not implemented, it is classified as unwanted traffic, so the security system will discard this packet.

Now consider an application layer gateway that is implemented or configured to enforce a policy that only allows RFC 1035 compliant DNS protocol elements. The application layer gateway will inspect the Additional Section in the expanded DNS priming request, parse and process type A resource records as "allowed" but it will reject a DNS priming response if it encounters AAAA records because these are "not defined" in RFC 1035 and thus treated as potentially malicious (hostile).

We thus consider the following issues with respect to choosing alternative (2).

*Will the presence of AAAA records in the DNS priming response influence the way intermediate devices enforce security policy on DNS messages?*

Using the same tests performed against TLD name servers that return AAAA records, members of the committee were able to demonstrate that DNS response messages containing AAAA records will pass through a number of commercial firewalls that are commonly used by large organizations and commonly interposed between an organization's internal name servers and external name servers (e.g., TLD and root name servers).

*Is EDNS0 support among intermediate systems in production networks prevalent enough to choose a priming response alternative that cannot fit within the maximum DNS message size specified in RFC 1035?*

Some intermediate systems and application layer gateways may not support EDNS0 extension mechanisms or may be configured to reject DNS messages containing the OPT parameter resolvers use to indicate they are capable of receiving UDP-encapsulated messages larger than 512 bytes. Other intermediate systems may be capable of processing EDNS0 extension mechanisms but may have been configured to block them. For some systems, this may be the default behavior, as in the case of the Cisco PIX version 6.2.5 and earlier. In some cases, organizations may have configured a security policy at a firewall to protect against attacks that use large DNS responses as a means to exploit vulnerabilities in certain name server implementations [3].

Members of the committee informally tested intermediate (security) systems by composing and issuing Type NS queries to Top Level Domains that currently return A and AAAA records from hosts behind the security system. In this case, the queries used the EDNS0 option to advertise a buffer size of 4096 bytes. The sizes of the responses ranged from 521 bytes to 730 bytes. Members of the committee were able to demonstrate that a number of commercial firewalls will allow UDP-encapsulated DNS responses larger than 512 bytes to pass unless a security policy is specifically configured to block such traffic. These informal tests were again valuable, but the committees sought broader and more formal testing from DNS server vendors, developers and the user community at large. These are described in the following section.

### Testing Firewalls for IPv6 and EDNS0 Support

Any party, vendor or user, can test the action an intermediate system takes when it encounters type AAAA resource records by composing and issuing Type NS queries that currently return A and AAAA records of certain TLD name servers (e.g., UA, FR, JP, and HK). By advertising a receive buffer of at least 811 bytes, any party can also test the action an intermediate system takes when it receives a UDP-encapsulated DNS response message larger than 512 bytes by composing from TLD name servers such as FR and HK. These tests are sufficient to verify that an intermediate system implementation and policy configuration will allow priming response messages to pass without modification or interference.

RRSAC and SSAC have solicited Internet community participation to test how intermediate systems react when DNS response messages contain AAAA RRs and when UDP-encapsulated DNS response messages are greater than 512 bytes (See SAC016, [4]). The results reported to the ICANN SSAC Fellow when this report was published are reproduced in Appendix E

## *IP Reassembly and Security Policy Issues*

The issue we consider here is related to EDNS0 support and the use of DNS messages larger than 512 bytes. All implementations and intermediate systems ought to be capable of reassembling IP packets that have been fragmented in transit [5]; however, security administrators may configure security systems to intentionally block DNS messages that exceed 512 octets to thwart forms of DDoS attacks that make use of IP fragmentation.

SSAC Advisory SAC008 does in fact recommend that TLD name servers block IP packets carrying UDP messages exceeding the standard 512 bytes, with the caveat that "TLD name server operators should recognize that future protocol extensions and enhancements may result in changes to this filtering rule" [6]. One possible change is for TLD operators to allow UDP-encapsulated DNS response messages exceeding 512 bytes from root name servers only (e.g., a list of trusted IP addresses). While these addresses could be used in spoofing attacks, the amplification factor is not quite the same as it would be if TLD operators removed the filter entirely.

# 4. Findings

The SSAC and RSSAC offer the following findings for consideration:

1. Adding IPv6 addresses at the root of the DNS affects the root hints file and the priming exchange.

2. The existing procedures for publishing root hints need not be changed to add AAAA addresses of root name servers in the files made available at ftp://ftp.internic.net/domain/, however making a version of root hints that includes AAAA records for the root name servers configured with IPv6 addresses may be appropriate.

3. DNS implementations used by all thirteen root name server operators are capable of including IPv6 records.

4. Changes to include IPv6 addresses affect the DNS priming response in two respects:

    a. Adding IPv6 addresses adds a resource record type (AAAA) that many implementations have never seen returned in a DNS priming response.

    b. No more than two (2) AAAA resource records can be included in the response if the overall message size is to fit within the 512 byte maximum UDP-encapsulated DNS message size specified by RFC 1035.

    c. A DNS priming response containing the names, type A records and type AAAA records for all thirteen root name servers will result in a response message of 811 bytes. Resolvers that use EDNS0 and advertise a receive buffer of at least 811 bytes will receive the entirety of the message. Resolvers that use EDNS0 but advertise a receive buffer less than 800 bytes and resolvers that do not use EDNS0 will receive DNS response message containing an *abbreviated* Additional section which will contain at least two type AAAA records (see *Root Name Server Considerations* in Section 3).

5. Testing conducted by members of the committee and the community at large indicate that:

    a. Resolvers commonly used in production networks today are able to process IPv6 address records returned in response to type NS queries by TLD name servers without incident.

    b. Intermediate systems commonly used in production networks today allow DNS messages containing IPv6 addresses to pass without incident (either as a default

policy or by user configuration).

c. Resolvers commonly used in production networks today are EDNS0 capable. Statistics from RIPE suggest that the majority of these resolver installations advertise receive buffers greater than the 811 bytes that root name servers would require to return a DNS priming response message containing the IPv4 and IPv6 address records for all 13 root name servers.

d. Many intermediate systems commonly used in production networks today allow UDP-encapsulated DNS messages that exceed 512 bytes to pass without incident. Some systems block longer messages by default. Other systems are intentionally configured to block such messages to protect against IP-level fragmentation attacks. ICANN and IANA should give the community ample time to test security policy configuration at intermediate systems before making changes to the root hints and root zone file that would increases the size of UDP-encapsulated DNS response messages beyond 512 bytes.

On the basis of the above findings, the committees conclude that changing the DNS priming response to include IPv6 address records will have minimal impact on name server implementations and intermediate systems used in production networks.

Additional study and testing is encouraged to continue to assess the impact of including AAAA records in the DNS priming response. Testing should be part of an overall strategy or "road map" for deployment that would ultimately result in the inclusion of the names, type A records and type AAAA records for all thirteen root name servers in the priming response. Root name server operators should continue to offer a public test facility for a reasonable time frame that can be used by product implementers as well as DNS, network, and security administrators to verify that their name service will not be interrupted on the cutover date.

Providing advanced notice of this change in a variety of venues – ICANN and supporting organization web sites, trade publications, and other technology news venues and forums – is an important element of the overall strategy. Advanced notice will give sufficient time to test well in advance of the date when root name servers will begin returning a "full" priming response.

# 5. Recommendations

ICANN SSAC and RRSAC recommend that type AAAA records for all root name servers so addressed should be included in the root hints and root zone files and that they be returned in priming responses from root name servers as soon as practically possible, The committees jointly conclude that the most expedient way to proceed is for ICANN, IANA and the root name server operators to coordinate a phased deployment.

1. ICANN and IANA should provide advanced public notice and identify a date on which DNS priming responses from root name servers will include names, type A records and type AAAA.

2. ICANN should continue to solicit testing and report how recursive name server and intermediate system implementations behave when they receive the larger priming response to the community at large. Currently SAC 016 [4] and SAC 017 [2] serve this purpose. These documents should continue to identify software, versions, and (where appropriate) special configuration settings that will permit systems to behave correctly when root hints and DNS priming responses contain AAAA addresses and when the priming response exceeds the RFC 1035 maximum message size.

3. After the specified date, IANA should publish a root hints file containing all thirteen A resource records of root name servers plus the AAAA resource records of all root name servers so addressed at ftp://ftp.internic.net/domain/. On the specified date, IANA should add the AAAA records for the root name servers so addressed to the root and root-servers.net zones. Once all root name servers load these updated zones, DNS priming responses will return names, type A records for all root name servers, and type AAAA records for root name servers that are assigned IPv6 addresses.

4. IANA should add AAAA resource records for other root name servers *as they are assigned* and in accordance with existing update policy and practice so that ultimately, the priming response will return both A and AAAA resource records for all thirteen root name servers.

# Appendix A. Background Information

## *The Domain Name System*

The Internet Domain Name Service ([7, 8] is modeled as a distributed database, organized as a tree structure. In the structure, each node in the name space and all its descendents are called a **domain**. A domain is thus a subtree of the Internet name space. Domains have names. Each domain is named after its topmost node, and each descendent (node) of a domain has a **label** assigned or registered within the domain. A node's **domain name** is the list of the labels on the path from the node to the root of the tree. The labels of sibling nodes must be unique.

There is a single, authoritative **root** for the DNS and it is commonly referred to as "dot" or "." Labels assigned to nodes directly subordinate to the root identify Top Level Domains (TLDs). The registration of labels within TLDs is delegated to Registry operators. Organizations and individuals who register labels within TLDs are called domain name **registrants.**

The label relationships between the root, TLD operators, and domain name registrants who register second level labels within TLDs is depicted in Figure A-1:

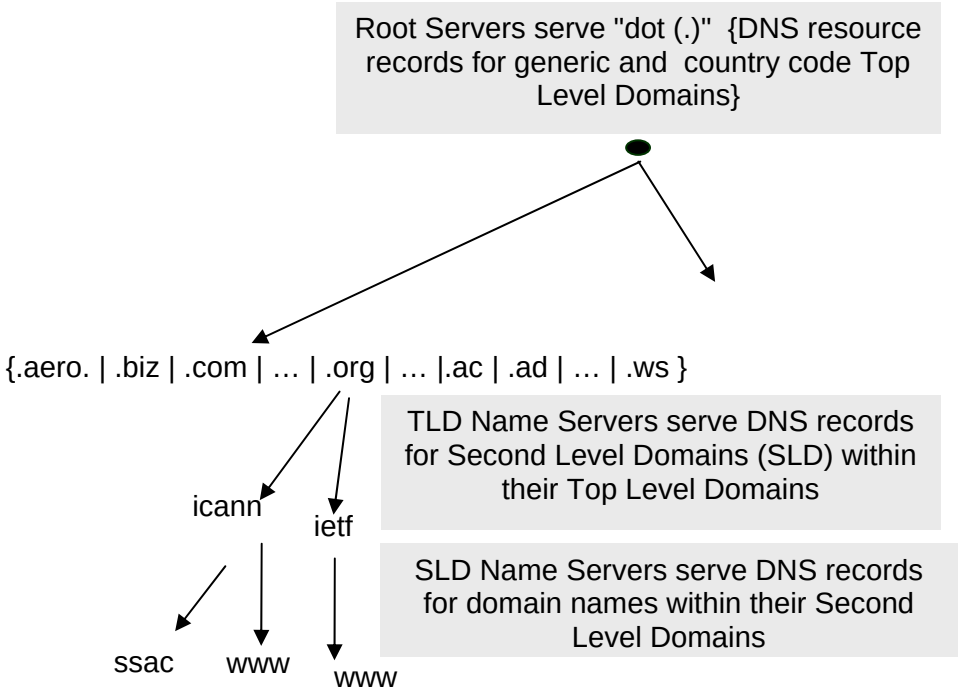


**Figure A-1. Label Relationships in the Domain Name System**

Domain name records are commonly stored in master files distributed throughout the Internet. Master files are hosted on **name servers**. Name servers are key components of the DNS. They store complete information for some part of the domain tree over which they have administrative control. In particular, name servers that host the complete

database or **zone** for a particular sub-tree of the domain space are said to be **authoritative** for that sub-tree.

## *Root Name Servers*

The **root name servers** host a critically important master file. The **root zone file** contains authoritative data for the top most level of the DNS. The root zone file contains several classes of resource records, as illustrated in Table 1-1. (Note: the symbol ✂ is used to indicate that some data have been trimmed from the example.)

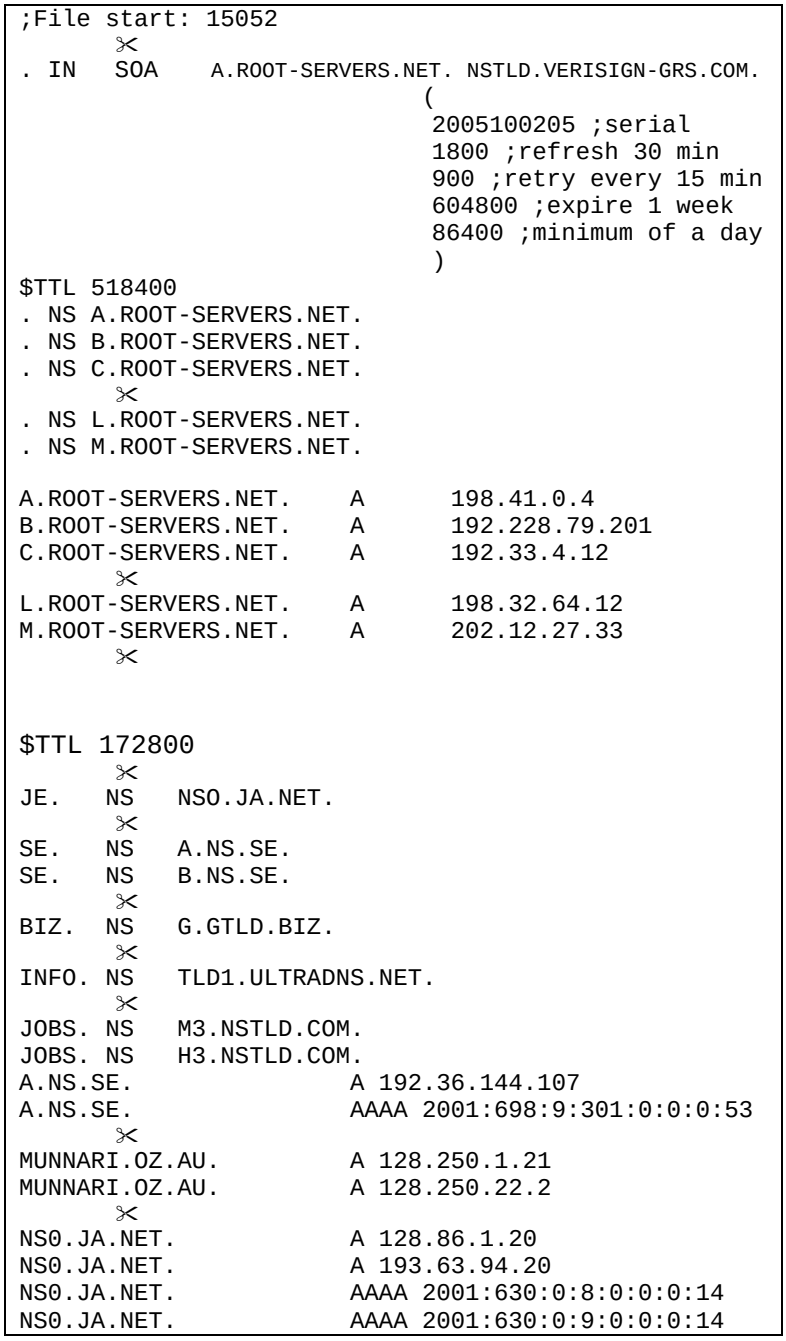


**Figure A-2. Label Relationships in the Domain Name System**

## *Resolver and Name Servers*

A **resolver** asks questions about domain names, e.g., it queries the DNS. In the client-server model used by many Internet applications, the resolver is the DNS client. Typically, a user application determines the IP address associated with a domain name by issuing a (remote) procedure call to a name resolution process called a **stub resolver**. A second type of DNS client, the **iterative resolver**, is typically an element of a recursive name server. Both stub and iterative resolvers direct queries to **name servers**, which provide the server element of DNS.

Authoritative name servers answer queries using the zone data over which they exercise authority. A **recursive name server** performs name server and iterative resolver functions, as follows. When a recursive name server receives a DNS query from a user application that it cannot answer using DNS information at hand, the iterative resolver composes a DNS query message requesting the address record associated with the domain name and forwards the request to a root name server. If the root name server knows the answer, it returns the requested information in a DNS response message. If the root name server does not know the answer, it provides the resolver with the names and addresses of the top level domain name servers in which the queried domain name is registered. This is called a **referral**. The recursive name server will then query one of the TLD name servers serving the top level domain of the name being resolved. If the TLD name server knows the answer, it returns the requested information. If it does not, the TLD name server provides the recursive name server with the names and addresses of the second level domain name servers. The process continues (iterates) until the name is resolved or determined not to exist. Figure A-3 illustrates the role of a recursive name server.

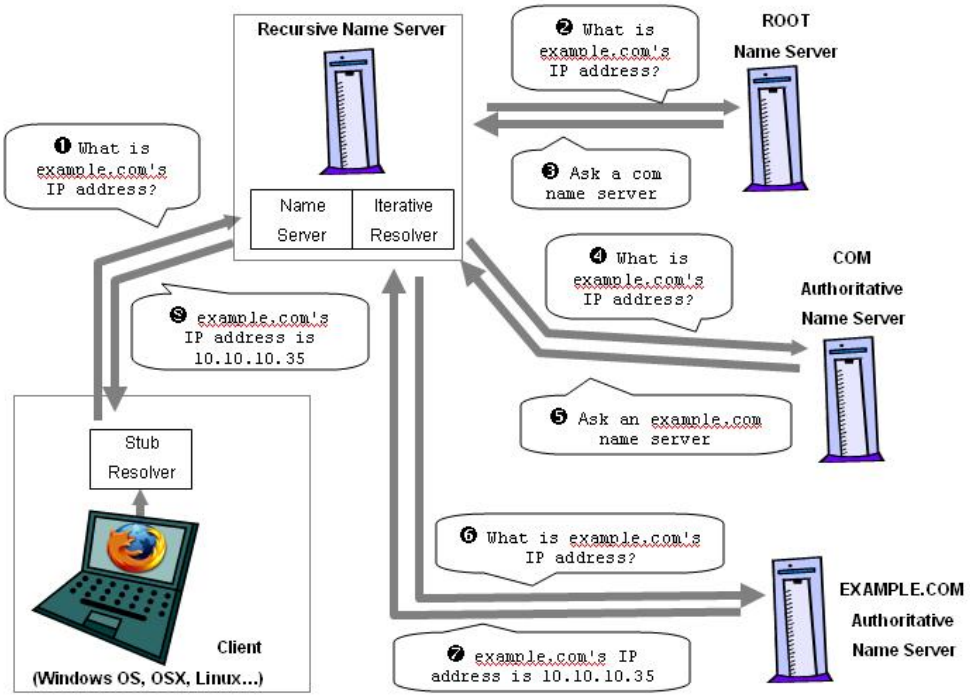


**Figure A-3. Name Resolution via a Recursive Name Server**

In practice, a resolver on a client host is configured to query (local) recursive name servers that *cache* DNS response information for frequently queried domain names. When caching is used and a recursive name server receives a domain name resolution request from a resolver, the recursive name server examines its cache to determine if the requested name information has already been stored locally *before* it iterates the request as described earlier. If the information is locally available, the recursive name server immediately returns a response to the requesting resolver (and does not query the root name servers).

Caching implies that not every query is referred to a root name server, but caching depends on referrals from the root. Caching is important, however, because it reduces DNS traffic and message processing loads on root as well as TLD name servers.

Cached information is not authoritative, but the DNS uses timeouts to purge potentially stale information. As DNS Security (DNSSEC, [9]) becomes more widely deployed, a resolver will be able to verify the integrity of DNS data returned in a DNS response message irrespective of the name server it has queried.

### *DNS Traffic and Intermediate Systems*

In practice, the communication paths between client hosts, name servers, and root name servers comprise many types of intermediate systems. While many of these perform network level routing and switching operations, others may inspect or process application traffic for a variety of (security) policy enforcement purposes. Such systems include network and application firewalls, in-line intrusion prevention systems, and application layer gateways, also known as security proxies. Many such intermediate devices process and inspect DNS messages for security purposes, e.g., to ensure proper protocol behavior and to detect and block:

- malformed or maliciously composed messages that can be used to probe for and exploit vulnerabilities in specific DNS implementations
- traffic flooding attacks (e.g., DNS DDoS amplification attacks [6])
- traffic that violates a security policy; for example, an organization may wish to control DNS traffic by
    - Destination and source IP address,
    - Query type (e.g., to prevent zone transfers), and
    - Protocol operation type.
    - Protocol composition (e.g., to block DNS messages exceeding the maximum message size specified in RFC 1035)

It is also worthwhile to note that host intrusion detection software may be installed on name servers. Such security software may process and inspect DNS messages for security purposes as well, and may detect and block traffic in the same manner as intermediate devices.

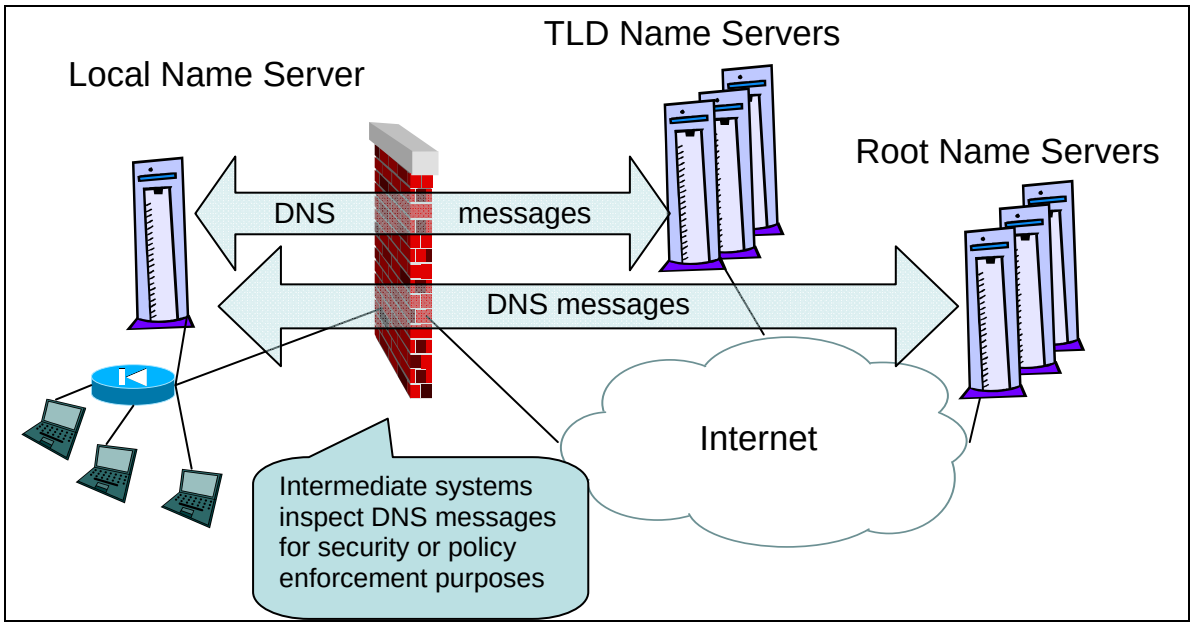


**Figure A-5: Communications Paths between Name Servers (conceptual)**

## *The Root Hints File*

A recursive name server's iterative resolver must know the IP address of at least one root name server to function properly. Commonly, name server software provides sufficient configuration information during installation to assure that a host connected to the Internet can query a root name server by including a *hints* file. (Note: Some implementations, including BIND version 9, include root hints in the binary distribution. Such implementations may use a hints file if one is present.)

The hints file contains the name of one or more root name servers and the IP address(es) assigned to the root name server(s). For example, the `cache.dns` file in the folder `C:\winnt\System32\DNS` contains the root hints information for the DNS service of Microsoft Windows Server 2003 [10]. For the BIND DNS server, LINUX and BSD distributions include root hints information in a file typically in the directory `/var/named`. The file name varies across distributions but is commonly one of `named.cache`, `named.root`, or `db.cache`.

### Creation and Maintenance of the Root Hints File

By convention, root name server domain names are assigned single letter labels within the domain `ROOT-SERVERS.NET`; specifically, the root name servers are assigned third-level labels `A` through `M`. Root name server operators [11] are responsible for assigning IP addresses to root name servers. Only thirteen root named server names can serve the root zone. The number thirteen was imposed as an upper limit to allow a specific DNS message response called the *priming response* to fit within the maximum DNS message size specified in RFC 1035. Note that the number thirteen relates to the number of domain names assigned to root name servers. In several cases, a single root server name represents multiple actual name servers using a technique called anycast addressing, where one IP address can be bound to many geographically diverse network endpoints.

All the root name servers have IPv4 addresses. Some root name server operators have assigned IPv6 addresses as well. These addresses do not yet appear in the root hints file.

Root name server operators are responsible for notifying IANA when they add or change the addresses of the name servers they administer. The IANA maintains the authoritative root hints file. Changes to root hints information are made at the explicit request of root name server operators and are reflected in root hints by mutual agreement between ICANN and the U.S. Department of Commerce.

The root hints are published at `ftp://ftp.internic.net/domain/` [12] under the popular names `named.cache`, `named.root`, and `db.cache` to facilitate this method. VeriSign, the company that hosts the ftp.internic.net server, hashes and signs these files for integrity protection and authentication purposes using PGP encryption software (the signature files can be found at `ftp://ftp.internic.net/domain/`, as well), thus automated methods can be used with some confidence by programming to verify both the hash and digital signature prior to replacing the local file. The root hints file is reproduced in Appendix C.

## *Maintaining Accurate Root Hints Information*

Iterative resolvers must have accurate information about root name servers to operate properly. Maintaining the accuracy of root hints information on a resolver or a recursive name server has two dimensions. The first – maintaining the accuracy of any pre-configured information regarding the names and IP addresses of root name servers – is a configuration matter. The second – verifying the accuracy of pre- or statically configured root hints information – is a bootstrap procedure performed by many resolvers when name service is initialized (or according to a pre-defined time interval) and involves a DNS protocol exchange called **priming**. Strictly speaking, recursive name servers are not required to perform a priming exchange, but the practice is very common and is thus worth discussing.

### Updating and Maintaining Root Hints Files

Historically, name server administrators were responsible for updating root hints information on their respective servers. Today, administrators continue to perform this in several ways:

1) **Manual process**. An administrator can manually replace the local copy of the root hints file with one he downloads from ftp://ftp.internic.net/domain/.

2) **Scripted process**. An administrator can schedule a program to periodically check the accuracy of the local copy of the root hints file [13]. If the local copy is incorrect, the program can automatically replace it with one it can download from ftp://ftp.internic.net/domain/.

3) **Commercial OS vendor updates**. Administrators can rely on software updates by commercial vendors to update root hints files. Microsoft, for example, updates

the `cache.dns` file in a service pack distribution [14].

4) **DNS software updates**. A new installation or an upgrade of existing DNS software obtained from the vendor will often include the root.hints file current when the distribution was packaged [15].

## *DNS Priming Exchange*

Name server administrators perform the actions described in the previous section to keep static configuration current. Since there are margins for error in all the common practices described above, many resolver implementations attempt to verify the root hints information at hand. This verification process is called a priming exchange.

The priming exchange makes use of standard DNS query and response messages. A DNS query may be represented as a 3-tuple of {QNAME, QTYPE, QCLASS}. QNAME is the domain name about which we are interested in obtaining information: for the priming query, this is ".", meaning the root. QTYPE specifies the type of resource record we seek, e.g., a name server resource record (NS). QCLASS specifies the class of resource record, typically IN.

The priming query is for (QNAME=".", QTYPE="NS", QCLASS="IN"). The answer contains NS records in the authority section and the corresponding A records in the additional section. All DNS messages share a common format, as follows:

| | |
|---|---|
| Header Section | Protocol parameters |
| Question Section | The question or query from the client (what is being asked) |
| Answer Section | Resource records that answer the question |
| Authority Section | Resource records identifying the domain authority |
| Additional Section | Resource records containing additional information that complement the answer, these are answer-dependent |

A name server begins the priming exchange by sending a **DNS query message for a resource record of type NS** to one or more of the root name servers identified in the root hints file.

## DNS Priming Query

In the case of the priming exchange, the name queried is "." and the class is "IN". Figure A-6 provides a screen snapshot of how a packet capture utility would decode and display the priming exchange, and thus illustrates the exact composition of the priming query as hosts transmit it today:

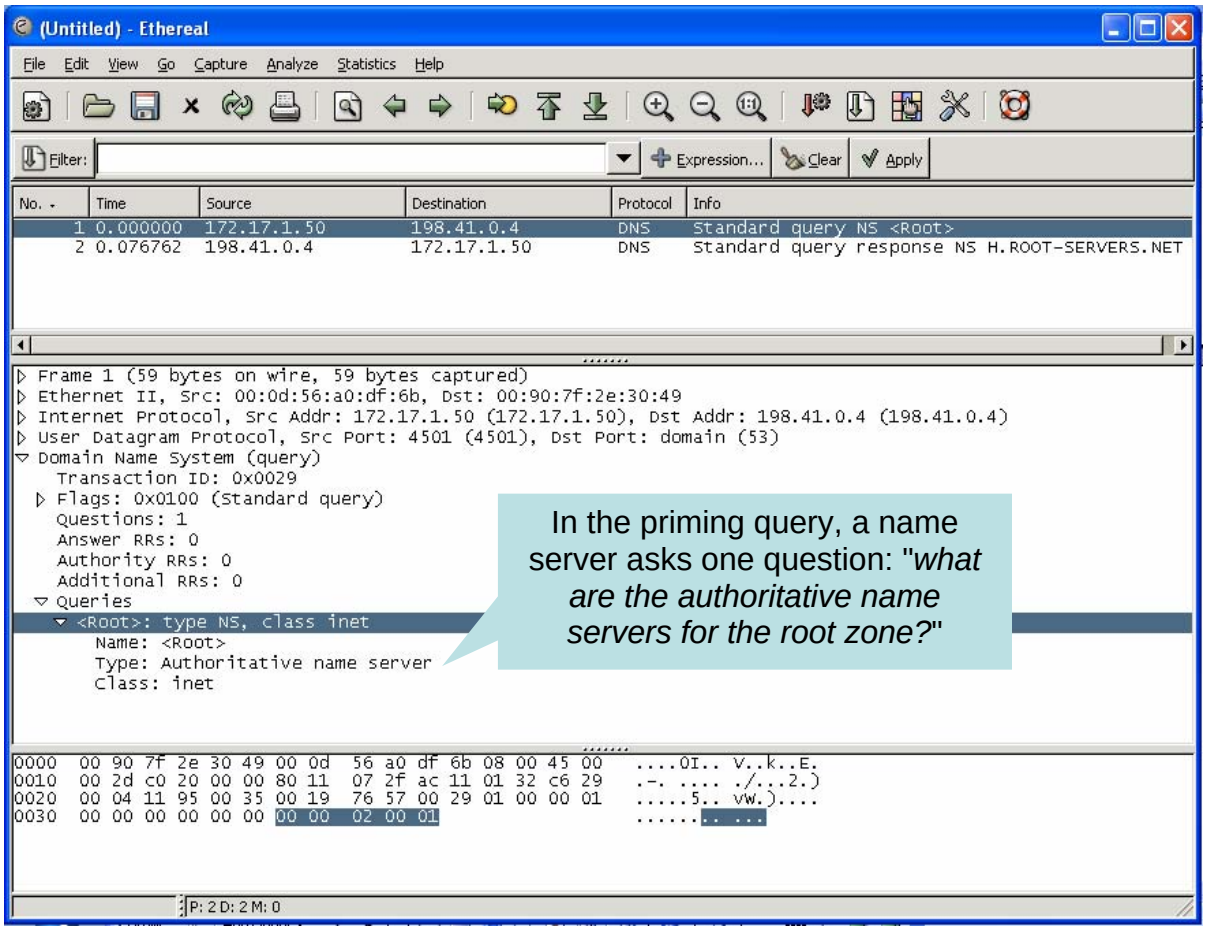


**Figure A-6. DNS Priming Query**

The priming query is sent to at least one root name server. Commercial and open source operating systems and name server resolver implementations behave differently with respect to which and how many root name servers they will query during this bootstrap process [13, 15]. A name server administrator can also influence this behavior using scripts or by modifying the default configuration of name service on a host he administers.

If the DNS priming exchange fails to complete, name servers will use locally available "hints" information.

## DNS Priming Response

A root name server responds to the DNS priming query message (type NS) with a response message listing the **NS resource records for the root**. The priming response message conveys important information in the Answers and Additional Sections.

### The Answer Section

The **Answer** Section contains the name, type, class, and TTL (time to live) of all the root name servers. Figure A-7 illustrates the DNS priming response message with the Answer section expanded for closer examination:

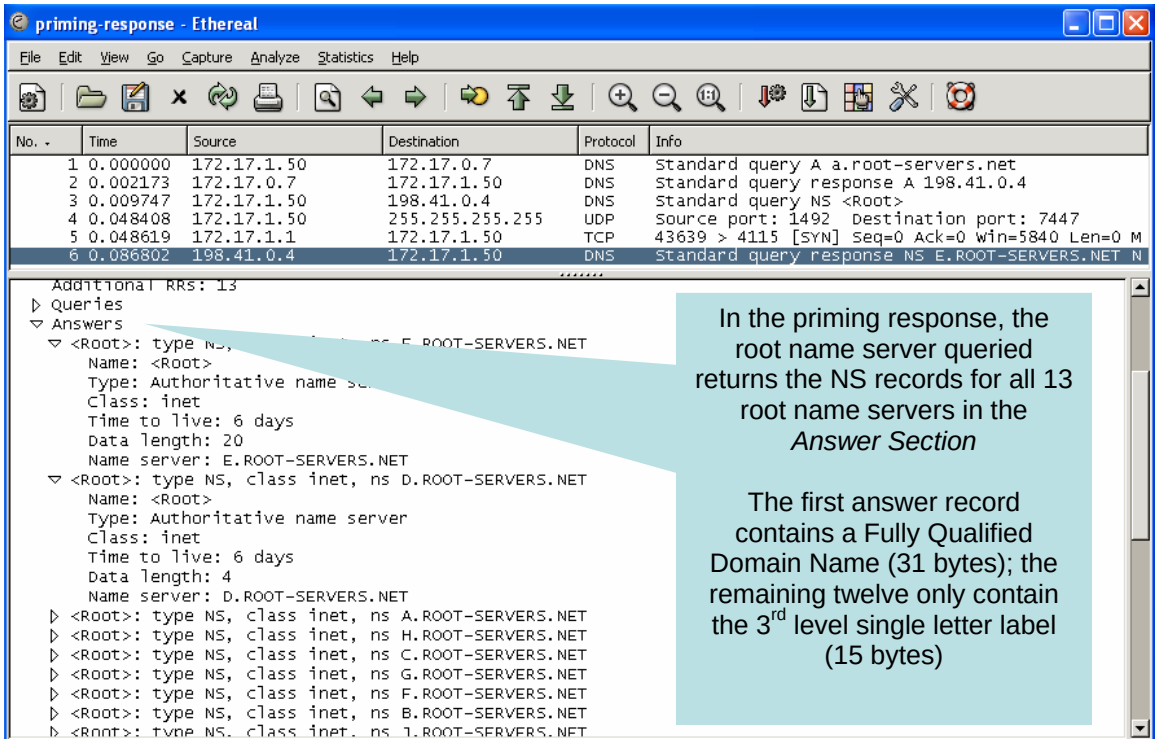


**Figure A-7. DNS Priming Response (Answers expanded)**

A root name server returns a fully qualified domain name in the first NS resource record, which occupies 31 bytes of the message. To conserve space, the root name server only returns the third level label in the second through thirteenth NS resource records in the Answer Section of the priming responses (using name compression, only four bytes are required instead of the twenty required for the fully qualified domain name). Each compressed NS resource record occupies 15 byes of the message.

### Additional Section

In addition to the Answer section, the DNS priming response message will contain data in the **Additional Section**. Each record in the Additional Section provides the name, type, class, TTL, and IPv4 address (resource record type A) of a root name server identified in the Answer Section:

Figure A-8 illustrates the DNS priming response message with the Additional Section expanded for closer examination:

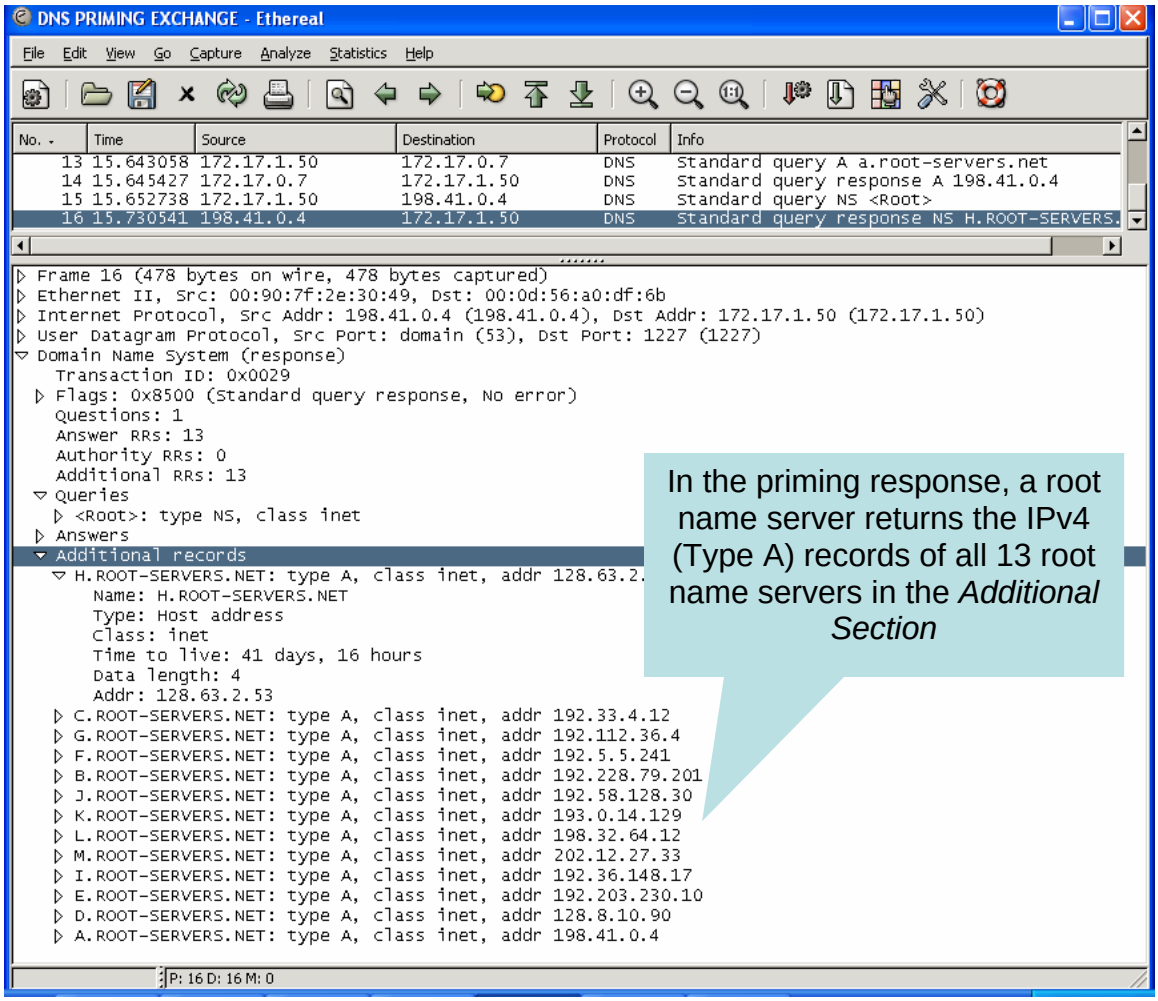


**Figure A-8. DNS Priming Response (Additional Records expanded)**

The DNS priming response message illustrated in both Figures A-7 and A-8 only returns IPv4 addresses of root name servers.

**DNS Priming Response Message Size**

A DNS priming response message is encapsulated in a UDP datagram that is transmitted in an IP datagram having a total length of 464 bytes. Subtracting the IPv4 and UDP headers (20 bytes and 8 bytes, respectively), the length of the DNS message (e.g., the UDP payload) is 436 bytes, allocated as illustrated in Table A-1:

| DNS Priming Response Message (IPv4 only) | # Bytes |
|---|---|
| Required Headers:<br>• Transaction ID, Flags, Questions, Answer RR count, Authority RR count, Additional RR count | 12 |
| Query<br>• Name "." Type NS, Class INET | 5 |
| Answers:<br>• First answer contains name, type, class, time-to-live (TTL) and Data length (value 20), plus the Fully Qualified Domain Name (FQDN) of a root name server (e.g., H.ROOT-SERVERS.NET)<br>• Second through 13th answers contain only name, type, class, TTL and Data length (value 4) plus the Relative Domain Name (RDN) of a root name server (e.g., the single letter G, F, E…) and occupy 15 bytes<br>(Thus, we have 12 answers and each is 15 bytes long). | 31<br><br>180 |
| Additional:<br>• Each of the 13 A records in the Additional contains name, type, class, TTL and Data length (value 4) and an 4-byte IPv4 address and occupies 16 bytes<br>(13 records times 16 bytes per record equals 208 bytes) | 208 |
| Total length | 436 bytes |

**Table A-1 DNS Priming Response Message (IPv4 only)**

Note that root name servers use *name compression* in the DNS protocol to reduce the number of bytes required to return the domain names of all 13 root name servers. This allows the overall length of the DNS priming response message to fit within the 512 byte maximum UDP-encapsulated DNS message size specified in RFC 1035, and assures that a UDP-encapsulated response will not be fragmented over any link that supports the default IP maximum datagram size of 576 bytes (see RFC 879, [16]).

## *IPv6 Addressing*

IPv6 addresses are 128 bits long and, like IPv4 addresses, are assigned to network interfaces of Internet hosts [17, 18]. IPv6 addresses are represented as eight groups of sixteen bits. Each group of sixteen bits is represented as four hexadecimal digits, separated by colons, e.g., `FEDC:BA98:7654:3210:FEDC:BA98:7654:3210`. For readability, leading zeroes in any subfield may be omitted, thus, writing `1080:0:0:0:8:800:200C:417A` is equivalent to writing the IPv6 address as `1080:0000:0000:0000:0008:0800:200C:417A`. One can further compress IPv6 addresses when writing them by using "::" to indicate multiple groups of 16-bits of zeros (Note: this convention may only be used once in an address).

The introduction of IPv6 into the Internet affects the DNS and several extensions to DNS standards are defined [19] to accommodate IPv6. A new resource record type for IPv6, the `AAAA RR`, maps domain names to IPv6 addresses, and a new domain, `IP6.ARPA`, is defined for reverse lookups using IPv6 addresses. Modern DNS servers can now process Additional Sections containing both IPv4 and IPv6 addresses record types (A and AAAA, respectively).

## *DNS Message Composition and Size Considerations*

RFC 2181, Clarifications to the DNS Specification [20], describes how name servers should compose UDP-encapsulated DNS messages in the event that a response will not fit within the maximum message size of 512 bytes specified in RFC 1035:

- If a name server cannot fit a complete resource record set (RRset) that is *required* in the Answer or Authority Section without exceeding the maximum UDP payload, the name server marks the response as *truncated* by setting the Truncation bit (TC) in the header of the DNS response message. This would apply, for example, to a name server record in the Answer section of a type NS response message.

- Upon receipt of a DNS message response that is marked as truncated, the resolver ignores the contents of this response. The resolver can retry the DNS query using TCP to accommodate the larger sized message.

- In the event that all the RRsets required for the response will fit but the entirety of the additional data a name server could return will not fit within the 512 byte maximum DNS message size specified in RFC 1035, the name server may return *abbreviated* additional data. In this case, the truncation bit is *not* set.

- Upon receipt of abbreviated data, and if the resolver needs missing data, the querying resolver can issue an additional DNS query using UDP to explicitly request the additional data that the name server was unable to include in the original query.

These guidelines clarify existing DNS protocol requirements. In addition, to accommodate longer DNS messages for both IP version 6 and DNS Security extensions, the DNS protocol was augmented by Extension Mechanisms for DNS (EDNS0, [21]). EDNS0 defines a method a host may use when it composes a DNS query message to indicate that the querying host is capable of receiving and processing UDP-encapsulated DNS messages greater than the maximum message size of 512 bytes specified in RFC 1035.

The extensions allow the host to indicate exactly how large a DNS response message it is prepared to handle. Hosts that have indicated they are able to use EDNS0 in a DNS query message but do not receive a DNS response message within a timeout period often retry the query without advertising EDNS0. This is useful in topologies where intermediate systems block DNS messages that exceed 512 bytes to thwart forms of DDoS attacks that make use of IP fragmentation. Iterative resolvers also retry without EDNS0 when the queried name server doesn't support EDNS0.

## Appendix B. References


[1] Measuring the Resource Requirements of DNSSEC
[2] Testing Recursive Name Servers for IPv6 and EDNS0 Support
http://www.icann.org/committees/security/sac017.htm
[3] US-CERT Vulnerability #738331, Domain Name System (DNS) resolver libraries vulnerable to read buffer overflow
http://www.kb.cert.org/vuls/id/738331
[4] Testing Firewalls for IPv6 and EDNS0 Support
http://www.icann.org/committees/security/sac016.htm
[5] Requirements for Internet Hosts – Communications Layers
http://www.ietf.org/rfc/rfc1122.txt
[6] DNS Distributed Denial of Service (DDoS) Attacks
http://www.icann.org/committees/security/dns-ddos-advisory-31mar06.pdf
[7] RFC 1034, Domain Names – Concepts and Facilities
http://www.ietf.org/rfc/rfc1034.txt
[8] RFC 1035, Domain Names – Implementation and Specification
http://www.ietf.org/rfc/rfc1035.txt
[9] DNS Security – Introduction and Requirements
http://www.ietf.org/rfc/rfc4033.txt
[10] Updating Root Hints (Microsoft Windows Server 2003 Technical Library)
http://technet2.microsoft.com/WindowsServer/en/library/7b69b6f9-f25e-4594-a04b-f08f3effa2031033.mspx?mfr=true
[11] Root Servers Operators web site, http://www.root-servers.org/
[12] Official Root Hints File, ftp://ftp.internic.net/domain/
[13] Configuring a BIND DNS Server
http://www.digitalpeer.com/id/configuringa
[14] List of Directory Services Fixes in Windows 2000 Service Pack 4
http://support.microsoft.com/default.aspx?scid=kb;en-us;815024
[15] Windows 2000 DNS: New Features of Windows 2000 DNS
http://www.microsoft.com/technet/prodtechnol/windows2000serv/plan/w2kdns2.msx#ENJAC
[16] TCP Maximum Segment Size and Related Topics
http://www.ietf.org/rfc879.txt
[17] Internet Protocol Version 6 (IPv6) Addressing Architecture
http://www.ietf.org/rfc/rfc3513.txt
[18] IPv6 Global Unicast Address Format
http://www.ietf.org/rfc/rfc3587.txt
[19] DNS Extensions to Support IPv6 Address Aggregation and Renumbering
http://www.ietf.org/rfc/rfc2874.txt
[20] Clarifications to the DNS Specification
http://www.ietf.org/rfc2181.txt
[21] Extension Mechanisms for DNS (EDNS0)
http://www.ietf.org/rfc/rfc2671.txt

# Appendix C. Root Name Server Hints File

```
;
;       This file is made available by InterNIC under anonymous FTP as
;           file                /domain/db.cache
;           on server           FTP.INTERNIC.NET
;       -OR-                    RS.INTERNIC.NET
;
;       last update:    Jan 29, 2004
;       related version of root zone:   2004012900
;
;
; formerly NS.INTERNIC.NET
;
.                        3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.      3600000      A     198.41.0.4
;
; formerly NS1.ISI.EDU
;
.                        3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000      A     192.228.79.201
;
; formerly C.PSI.NET
;
.                        3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000      A     192.33.4.12
;
; formerly TERP.UMD.EDU
;
.                        3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.      3600000      A     128.8.10.90
;
; formerly NS.NASA.GOV
;
.                        3600000      NS    E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.      3600000      A     192.203.230.10
;
; formerly NS.ISC.ORG
;
.                        3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.      3600000      A     192.5.5.241
;
; formerly NS.NIC.DDN.MIL
;
.                        3600000      NS    G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.      3600000      A     192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
;
.                        3600000      NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.      3600000      A     128.63.2.53
;
; formerly NIC.NORDU.NET
;
.                        3600000      NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.      3600000      A     192.36.148.17
;
; operated by VeriSign, Inc.
;
.                        3600000      NS    J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.      3600000      A     192.58.128.30
;
; operated by RIPE NCC
;
.                        3600000      NS    K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.      3600000      A     193.0.14.129
;
; operated by ICANN
;
.                        3600000      NS    L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.      3600000      A     198.32.64.12
;
; operated by WIDE
;
.                        3600000      NS    M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.      3600000      A     202.12.27.33
; End of File
```

# Appendix D. Emulating a DNS Priming Exchange Using the `dig` program

```
Microsoft Windows XP [Version 5.1.2600]
C:\dig>dig @a.root-servers.net ns

; <<>> DiG 9.2.3 <<>> @a.root-servers.net ns
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41
;; flags: qr aa rd; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 13
;; QUESTION SECTION:
;.                              IN      NS
;; ANSWER SECTION:
.                       518400  IN      NS      B.ROOT-SERVERS.NET.
.                       518400  IN      NS      J.ROOT-SERVERS.NET.
.                       518400  IN      NS      K.ROOT-SERVERS.NET.
.                       518400  IN      NS      L.ROOT-SERVERS.NET.
.                       518400  IN      NS      M.ROOT-SERVERS.NET.
.                       518400  IN      NS      I.ROOT-SERVERS.NET.
.                       518400  IN      NS      E.ROOT-SERVERS.NET.
.                       518400  IN      NS      D.ROOT-SERVERS.NET.
.                       518400  IN      NS      A.ROOT-SERVERS.NET.
.                       518400  IN      NS      H.ROOT-SERVERS.NET.
.                       518400  IN      NS      C.ROOT-SERVERS.NET.
.                       518400  IN      NS      G.ROOT-SERVERS.NET.
.                       518400  IN      NS      F.ROOT-SERVERS.NET.
;; ADDITIONAL SECTION:
B.ROOT-SERVERS.NET.     3600000 IN      A       192.228.79.201
J.ROOT-SERVERS.NET.     3600000 IN      A       192.58.128.30
K.ROOT-SERVERS.NET.     3600000 IN      A       193.0.14.129
L.ROOT-SERVERS.NET.     3600000 IN      A       198.32.64.12
M.ROOT-SERVERS.NET.     3600000 IN      A       202.12.27.33
I.ROOT-SERVERS.NET.     3600000 IN      A       192.36.148.17
E.ROOT-SERVERS.NET.     3600000 IN      A       192.203.230.10
D.ROOT-SERVERS.NET.     3600000 IN      A       128.8.10.90
A.ROOT-SERVERS.NET.     3600000 IN      A       198.41.0.4
H.ROOT-SERVERS.NET.     3600000 IN      A       128.63.2.53
C.ROOT-SERVERS.NET.     3600000 IN      A       192.33.4.12
G.ROOT-SERVERS.NET.     3600000 IN      A       192.112.36.4
F.ROOT-SERVERS.NET.     3600000 IN      A       192.5.5.241
;; Query time: 125 msec
;; SERVER: 198.41.0.4#53(a.root-servers.net)
;; WHEN: Tue Aug 29 09:06:25 2006
;; MSG SIZE  rcvd: 436
```

# Appendix E. Results Reported: Testing Recursive Name Servers for IPv6 and EDNS0 Support

The following results have been reported to the SSAC fellow:

| DNS Software | Operating System | Bootstraps when AAAA RRs present in hints file | Primes using IPv4 transport | Supports EDNS0 | Parses AAAA RRs | Functions properly following a priming exchange with a test root name server | Source |
|---|---|---|---|---|---|---|---|
| BIND 4.9.3-REL [5] | Redhat Fedora Core 6 Linux | YES | YES | NO | NO | YES | User |
| BIND 4.9.11-REL | Redhat Fedora Core 6 Linux | YES | YES | NO | YES | YES | User |
| BIND 8.2.2-P5 | SunOS Blakey 5.8 | YES | YES | NO | NO | YES | User |
| BIND 9.2.4 | Debian GNU/Linux | YES | YES | YES | YES | YES | User |
| BIND 9.3.2 | Mac OS X version 10.4.8 | YES | YES | YES | YES | YES | User |
| BIND 9.3.4 | FreeBSD 6.2 | YES | YES | YES | YES | YES | User |
| BIND 9.4.0 rc2 | FreeBSD 6.2, Suse Linux 10.1 | YES | YES | YES | YES | YES | User |
| djbdns dnscache 1.05 | Redhat Fedora Core 6 Linux | YES | YES | YES | NO | YES | User |
| DNS Commander [4] | Windows NT/XP, Linux, Solaris | YES | N/A | YES | YES | N/A | Vendor |
| DNSJava | Java (any OS with Java support) | N/A | N/A | YES | YES | N/A | Developer |
| JDNSS [1] | Java (any OS with Java support) | N/A | N/A | NO | | N/A | Developer |
| MaraDNS 1.2.12.04 [2] | BSD, Linux, Windows | NO | NO | NO | YES | N/A | Developer |
| Men & Mice Suite 5.x with current BIND 8 or BIND 9 | Windows 2000/Windows 2003/Linux/FreeBSD/ MacOSX/Solaris | YES | YES | YES | YES | YES | Vendor |
| Mice & Men QuickDNS v1.0 - 3.0 | Apple MacOS Classic (System 7 to MacOS 9) | NO | YES | NO | NO | NO | Vendor |
| Microsoft DNS Server | Windows 2000 5.00.2195 SP4 | YES | YES | NO | NO | YES | User |
| Microsoft DNS | Windows 2003 | YES | YES | YES | YES | YES | User |

| Server | | | | | | | |
|---|---|---|---|---|---|---|---|
| Nominum CNS 1.6.5.0 | Solaris 10 | YES | YES | YES | YES | YES | Vendor |
| Posadis DNS version 6 | Windows XP SP2 | YES | NO | NO | YES | YES | User |
| PowerDNS Recursor 3.1.4 | Debian GNU/Linux | YES | YES | YES | YES | YES | User |
| QuickDNS 3.5 to 4.6 with current BIND 8 or BIND 9 | Windows 2000/Windows 2003/Linux/FreeBSD/ MacOSX/ Solaris | YES | YES | YES | YES | YES | Vendor |
| SimpleDNS version 4.00.06 [3] | Windows XP SP2 | YES | YES | NO | YES | YES | User, Vendor |

[1] Used as a leaf or stub resolver. Does not perform recursive lookups and does not prime.
[2] Recursive resolver does not have IPv6 support; recursion must be disabled to bind to IPv6 address.
[3] Priming is performed according to a preconfigured time interval (default once every 7 days).
[4] This product does not perform a priming query and relies on root hints configured for the name server.
[5] Server operates correctly despite error messages recorded in syslog.

# Appendix F. Results Reported: Testing Firewalls for IPv6 and EDNS0 Support

The following results have been reported to the SSAC fellow:

| Product | Version | Action when AAAA RR encountered | Action when large DNS message received | Source |
|---|---|---|---|---|
| ARKOON Fast360 | 3.0/1 to 3.0/22 | Allow | Deny | vendor |
| ARKOON Fast360 | 3.0/23 and above, 4.x | Allow | Allow | vendor |
| Checkpoint Firewall-1 | NG, R55 | Allow | Allow | user |
| Check Point FW-1 NGX R61 HFA 1 on Nokia | IPSO 4.1-BUILD013 | Allow | Allow | user |
| Cisco C2600 | IOS 12.2(37) | Allow | Allow | user |
| Cisco FWSM | 2.3(4) | Allow | Allow | user |
| Cisco PIX | Version 6.2.5 | Allow | Deny | vendor |
| Cisco PIX | Version 6.3.5 | Allow | Allow[1] | vendor |
| Cisco PIX | Version 7.2.1 | Allow | Allow | vendor |
| Clavister | Security Gateway (All models) | Allow | Allow | vendor |
| Eland Systems SYS-2, SYS-2 SOHO | 3.x, 4.x | Allow | Allow | vendor |
| Fortinet Fortigate 60 | Version 3.0.x | Allow | Allow | user |
| FreeBSD OpenBSD pf | 6.2-PRERELEASE | Allow | Allow | user |
| GajShield Infotech | Securegate version 5.4 | Allow | Allow | vendor |
| Juniper/Netscreen | ScreenOS Versions 5.4r2, 5.30r3, 4.0.3r4.0 | Allow | Allow | user |
| Kobelt Development NetSentron | 3.1.0p11-Pro | Allow | Allow | vendor |
| Linux 2.6 kernel Shoreline Shorewall Firewall | 2.4.1-3 | Allow | Allow | user |
| Linux kernel - Debian iptables 2.6.17.1 Firewall | 2.6.17.1 | Allow | Allow | user |
| Lucidata Lucigate Firewall | 3.14 | Allow | Allow | vendor |
| Mandriva Linux 2006 OpenBSD | 4.0 pf | Allow | Allow | user |
| NetStealth Firewall | StealthOS | Not supported | Not supported | vendor |
| Secure Computing Sidewinder | Versions 5.2.1, 6.1.2.00 | Allow | Allow | user |
| Shiva/Eicon 3105 | v 8.42 | Allow | Allow | user |

| | | | | |
|---|---|---|---|---|
| Sonicwall | SonicOS Standard 3.1.0.7-77s | Allow | Allow | user |
| Sepehr 3400 | GOS 3.0 | Allow | Allow | vendor |
| Sepehr 4100 | GOS 3.0 | Allow | Allow | vendor |
| Watchguard Firebox X 1000 | Fireware v8.2 | Allow | Allow | user |
| Watchguard Firebox X Edge | 8.0 | Allow | Allow | user |
| XNet Solutions SN330 | Version 1.2.1 | Allow | Allow | vendor |
| XNet Solutions EN400 | Version 1.0.0 | Allow | Allow | vendor |

# Appendix G. Members of the SSAC and RSSAC committees

| SSAC | RSSAC |
|---|---|
| Alain Aina, Consultant | Rob Austein, IAB |
| Jaap Akkerhuis, NLnet Labs | Piet Barber, VeriSign |
| KC Claffy, CAIDA | Brett Carr, RIPE |
| Steve Crocker, Shinkuro (Chairman) | K. C. Claffy, CAIDA |
| Daniel Karrenberg, RIPE/NCC | Kenjiro Cho, WIDE Project |
| Johan Ihrén, Autonomica | David Conrad, ICANN |
| Rodney Joffe, Centergate | Steve Conte, ICANN |
| Mark Kosters, Verisign | Brian Coppola, VeriSign |
| Ram Mohan, Afilias | John Crain, ICANN |
| Russ Mundy, SPARTA, Inc | Joao Damas, ISC |
| Frederico Neves, Registro Brazil | Thomas de Haan, GAC Liaison |
| Jon Peterson, NeuStar | Cathy Handley, US Dept of Commerce |
| David Piscitello, ICANN SSAC Fellow | Geoff Huston, Telstra |
| Ray Plzak, ARIN, Vice Chairman | Johan Ihrén, Autonomica |
| Mike St. Johns, Nominum | Daniel Karrenberg, RIPE |
| Doron Shikmoni, ForeScout, ISOC-IL | Akira Kato, WIDE Project |
| Bruce Tonkin, Melbourne IT; | Mark Kosters, VeriSign Labs |
| Paul Vixie, ISC | Matt Larson, VeriSign |
| Suzanne Woolf, ISC | Bill Manning, EP.NET |
| | George Michaelson, APNIC |
| | Jun Murai, Keio University, WIDE Project |
| | Catherine Murphy, ARIN |
| | Evi Nemeth, CAIDA |
| | Frederico A C Neves, LACNIC |
| | Axel Pawlik, RIPE |
| | Ray Plzak, ARIN |
| | Karl Reuss, University of Maryland |
| | Andrei Robachevsky, RIPE |
| | Yuji Sekiya, WIDE Project |
| | Gerry Sneeringer, University of Maryland |
| | Dave Swager, NASA |
| | Paul Twomey , ICANN |
| | Paul Vixie, ISC |
| | Paul Wilson, APNIC |
| | Suzanne Woolf, ISC |
| | Chris Yarnell, ISC |