DNSext Working Group
Internet-Draft
Obsoletes: 1995 (if approved)
Intended status: Standards Track
Expires: October 25, 2012

A. Hoenes
TR-Sys
O. Sury
CZ.NIC
S. Kerr
ISC

April 23, 2012

DNS Incremental Zone Transfer Protocol (IXFR)
draft-ietf-dnsext-rfc1995bis-ixfr-01

Abstract

The standard means within the Domain Name System protocol for maintaining coherence among a zone's authoritative name servers consists of three mechanisms. Incremental Zone Transfer (IXFR) is one of the mechanisms and originally was defined in RFC 1995.

This document aims to provide a more detailed and up-to-date specification of the IXFR mechanism and to align it with the current specification of the primary zone transfer mechanism, AXFR, given in RFC 5936. Further, based on operational experience, this document juxtaposes to the original IXFR query a new query type, IXFR-ONLY, that will likely be preferred over IXFR in specific deployments.

This document obsoletes and replaces RFC 1995.

Discussion

Comments should be sent to the authors and/or the dnsext mailing list.

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 http://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 October 25, 2012.

Copyright Notice

Table of Contents

1. Introduction

1.1. Overview of DNS Zone Synchronization

The Domain Name System (DNS) standard facilities for maintaining coherent servers for a zone consist of three elements. Authoritative Transfer (AXFR) originally was defined in STD 13: "Domain Names - Concepts and Facilities" [RFC1034] (referred to in this document as RFC 1034) and "Domain Names - Implementation and Specification" [RFC1035] (henceforth RFC 1035), and is now precisely specified in "DNS Zone Transfer Protocol (AXFR)" [RFC5936] (henceforth RFC 5936). Incremental Transfer (IXFR) was originally defined in "Incremental Zone Transfer in DNS" [RFC1995]. A mechanism for prompt notification of zone changes (NOTIFY) is defined in "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)" [RFC1996]. The goal of these mechanisms is to enable a set of DNS name servers to remain coherently authoritative for a given zone.

For large domains that incur frequent changes that need to be available quickly to prospective DNS clients, AXFR has proven less suitable because it always transfers the whole zone content. The latency incurred in the propagation of changes to the DNS database ([RFC1034], [RFC1035]) can be substantially reduced in such scenarios by actively notifying secondary servers of the availability of a new version of the authoritative zone data at the primary server for a zone; this is accomplished by the DNS NOTIFY mechanism [RFC1996]. The time and resources needed to accomplish the transfer of the new zone content to the secondary servers in many cases can be reduced substantially by only carrying forward the changes from a previous version of the zone data. This is accomplished by the IXFR mechanism originally specified in RFC 1995 [RFC1995] and, more precisely, in this document.

1.2. Incremental Zone Transfer (IXFR) - Conclusions from Experience

The original specification for IXFR [RFC1995] has widely been perceived as not precise and detailed enough to ensure interoperable implementations, and multiple reports have been made to DNS working groups of observed IXFR implementation behavior that was not regarded as conformant with the spirit of the specification, as seen by the rough consensus of the community. Therefore, this document aims at giving a much more detailed and precise specification for the IXFR protocol, incorporating experience from more than a decade and evolved points of view -- in particular regarding security aspects of DNS operations. Implementations of this RFC are fully backward compatible with "proper" implementations of RFC 1995, but conformant implementations follow some more specific requirements included in this RFC to improve the performance and resilience of IXFR sessions.

The original IXFR automatically falls back to AXFR behavior whenever the IXFR server cannot fulfill the given delta-update request. In some deployments, in particular where multiple IXFR servers are available to the IXFR client, this can lead to premature fallback to AXFR-like behavior whenever the chosen IXFR server does not have the wanted delta-update information available, but another possible IXFR server would, which incurs the additional overhead that the client wanted to avoid whenever possible by his initial choice to use IXFR. This gap is closed by a variant of the IXFR mechanism, dubbed "IXFR-ONLY", which originally has been proposed in "IXFR-ONLY to Prevent IXFR Fallback to AXFR" [I-D.kerr-ixfr-only] and which is fully specified below as well.

Thus, this document re-specifies the IXFR mechanism as it is deployed in the Internet at large, giving more details than in the original specification, and using RFC 5936 as its foundation. Additionally, it newly specifies a versatile variant of IXFR, IXFR-ONLY.

This document is organized as follows: After presenting the terminology used and elaborations on the scope of this protocol and its specification in the next subsections, Section 2 gives an overview on the principles of operation of the IXFR protocol. Section 3 normatively specifies the IXFR query and response message format and the basic rules governing their generation and processing. Subsequent sections detail mandatory and optional server behavior, and they supply management, security, and IANA considerations.

1.3. Requirements Language

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 BCP 14, "Key words for use in RFCs to Indicate Requirement Levels" [RFC2119].

1.4. Terminology

This document freely makes use of basic DNS terminology as introduced in RFCs 1034 and 1035 ([RFC1034], [RFC1035]) and clarified and expanded upon in RFCs 2181 and 4033 ([RFC2181], [RFC4033]).

The terms "AXFR server", "AXFR client", "AXFR session", "General-purpose DNS implementation" and "Turnkey DNS implementation" are used as defined in Section 1.1 of RFC 5936 [RFC5936]. This document also adopts the typographical (letter case) conventions agreed upon there for denoting fields/parts of DNS messages -- cf. Section 3.

The bare term "IXFR" is intended to refer to the generic case of an IXFR or IXFR-ONLY query/response, unless it is clear from the context that the original IXFR Q-Type is dealt with specifically.

An "IXFR client" is a (secondary) name server for a given DNS zone that, based on a trigger event, for instance a DNS NOTIFY message, issues an IXFR query to a "superordinate" authoritative server (e.g., the primary server of the zone) and receives the IXFR response from it.

An "IXFR server" is an authoritative server that is presumed to become aware of changes to a zone earlier than other authoritiative servers, for instance the primary server for a zone as specified in STD 13 or a secondary server that already has synchronized with the primary server, and that is configured to respond to IXFR queries.

The interaction and protocol exchange(s) performed by an IXFR client and an IXFR server to issue an IXFR query and accomplish its processing are collectively denoted as an "IXFR session".

The behavior of an IXFR server falling back to full zone transfer when incremental updates are unavailable or unpractical is denoted (by common colloquial shorthand) as "Fallback to AXFR", although technically, no AXFR pseudo-RRs are involved in this protocol variant. (This is sketched in Section 2 and detailed in Section 4.)

1.5. Scope

In general terms, authoritative name servers for a given zone can use various means to achieve coherency of the zone contents they serve. For example, there are DNS implementations that assemble answers from data stored in relational databases (as opposed to master files), relying on the database's non-DNS means to synchronize the database instances. Some of these non-DNS solutions interoperate in some fashion. However, AXFR, IXFR, and NOTIFY are the only protocol-defined in-band mechanisms to provide coherence of a set of name servers, and they are the only mechanisms specified by the IETF.

This document does not cover incoherent DNS situations. There are applications of the DNS in which servers for a zone are designed to be incoherent. For these configurations, a coherency mechanism as described here would be unsuitable.

IXFR is an optional protocol component for authoritiative DNS servers; it is not needed on DNS resolver software that does not support the functions of an authoritative DNS server. Thus, all usage of normative BCP 14 [RFC2119] language is applicable only to DNS server implementations that claim support of this specification.

Whereas the original IXFR protocol already is widely implemented, IXFR-ONLY offers an operational choice for administrators of zones with a non-trivial propagation graph for the authoritative zone data, who are looking for more options to fine-tune the synchronization efficiency of their authoritative servers. It could be implemented without bare IXFR, but for the sake of backwards compatibility and flexibility, and for simplicity in documentation, it is strongly RECOMMENDED that IXFR-ONLY be always implemented in concert with bare IXFR.

2. Principles of IXFR Protocol Operation

Each version of the authoritative data of a DNS zone is identified by the SOA serial number (cf. Section 4.3.5 of [RFC1034] and Section 3.3.13 of [RFC1035]); successive zone versions are tagged with monotonically increasing serial numbers. Note that, as spelled out at the former citation, "Serial number advances and comparisons use sequence space arithmetic", and IXFR implementations need to pay particular attention to possible wraps. Below, serial numbers are symbolically referred to by "sn", mostly with some distinguishing postfix.

When an IXFR client currently serving, say, sn_o of a particular zone receives a trigger that it should incrementally update the zone data, it sends one of the two flavors of an IXFR request to an IXFR server, with the expectation to obtain in the IXFR response the change information necessary to transform the sn_o zone data into the zone data of the current zone version, say, sn_n.

The details of which triggers can and will start such IXFR session are implementation dependent. Possible triggers are some time schedule or a management request, but most likely the IXFR query will be triggered by a DNS NOTIFY message received from an authoritative server of higher precedence in the propagation graph for the zone.

Possible IXFR servers are usually configured (per zone) on an IXFR client, amended with some indication of precedence. Similarly, IXFR servers are configured (per zone) with the identities of the secondary servers they should accept as IXFR clients. This way, some authoritative servers for a given zone may act both as an IXFR client and an IXFR server. Among all authoritative servers for a zone, at least one server (the primary server of the zone) is not acting as an IXFR client. This way, the {IXFR server, IXFR client} pairs form a binary relation on the set of these servers that defines a directed graph rooted at the primary server(s); this is the IXFR propagation graph for the zone.

> Note that, for the purpose of IXFR, it is possible that more than one server can be acting as a primary server; this requires that zone synchronization between these servers is accomplished by other mechanisms, e.g., AXFR, or non-DNS means like distributed database technology.

The most simple propagation graph is a star (hub and spokes) configuration, with the primary server as the central hub. For redundancy, important zones with many authoritative servers are likely to be configured with a more dense propagation graph that, for the sake of resilience and/or load sharing, gives IXFR clients a choice of multiple IXFR servers to contact. All these configuration details are a strictly local matter and do not affect interoperability; hence, these details are out of scope for this specification. The only property of the propagation graph that needs to be ensured by the zone administration is that each secondary (i.e., non-primary) server must be reachable by at least one path in this graph that originates in a primary server.

In order to be able to act as a useful IXFR server, a DNS server needs to keep track of the zone history, to a certain extent (as directed by local policy, as well). To this end, the server must maintain knowledge of the changes that have been applied successively to the zone content from one SOA serial up to the current version. This does not necessarily mean that each change needs to be recorded, however; if some parts of the zone content change frequently, it might be preferable to coalesce subsequent chunks of change information -- both for local storage and/or for transmission --, for instance instead of the change information from sn_1 to sn_2 and the change information from sn_2 to sn_3 (where sn_1 < sn_2 < sn_3), the change information from sn_1 to sn_3 can be provided. This condensation of data has some downsides, however: if an IXFR client serves sn_2 and asks an IXFR server for the delta information to the current version of the zone, but the server has performed the above condensation, it has no way to tell the necessary information to the IXFR client, and the IXFR request necessarily is doomed to fail. Therefore, it becomes apparent that an IXFR server must maintain seemless chains of change information chunks from all past SOA serial number values it wants/needs to support (because potential IXFR clients currently serve these zone versions) to the current zone version. See Section 6.3 for more details on Condensation.

Upon receipt of any IXFR query, the IXFR server conceptionally constructs a chain of change information chunks from the SOA serial number indicated by the client (sn_o) to the current zone version (sn_n).

If this is not possible, in the case of bare IXFR, the server falls back to AXFR, i.e. it provides the full zone content. In the case of an IXFR-ONLY query, however, an error response SHOULD be returned immediately to the IXFR client, thus giving it a chance to try with an alternate IXFR server that might (still) serve the client's sn_o and not to immediately incur the potential overhead of a full zone transfer. However, if the full zone content would fit into a single response packet over UDP, an IXFR server MAY refrain from signalling an error in response to an IXFR-ONLY query and behave as if the query had been IXFR. This is allowed because, in this case, the full IXFR transaction can be executed in a single packet exchange and an error return would necessitate more messages and hence cause additional overhead and delay, contrary to the performance optimization goal of IXFR-ONLY.

In case it turns out that the IXFR client already has the current zone version (sn_o = sn_n) or even a more recent one (sn_o > sn_n), the IXFR server does not reply with an empty chain of chunks, but with only the (current) SOA record of the zone.

If the IXFR server determines that it would be inefficient to transfer the series of chunks, it also may fall back to full zone transfer. Note that this is recommended behavior for the IXFR server, but the correct protocol operation does not depend on this (useful) optimization.

Ordinarily, in the generic case, the IXFR server transmits the change information chunks in their "natural" order (by ascending sn) to the IXFR client.

Each such change information chunk -- say from sn_a to sn_b -- is represented (conceptionally and on the wire) by a sequence of RR deletions and a sequence of subsequent RR additions, all of which need to be applied in order to transform the zone content at sn_a to the zone content at sn_b. For transfer in the IXFR response, each sequence starts with the corresponding SOA RR as its delimiter, and the other RRs within it can be given in arbitrary order.

The whole chain of change information chunks is embedded in a pair of copies of the new SOA RR (at sn_n), which serve as "sentinels". It is important to point out that the SOA RR is used only as a marker in this context and it can appear multiple times, as opposed to an RRSIG(SOA) RR, which is treated as a common record and needs to appear only once in the zone. That also means that an RRSIG(SOA) RR for sn_o has to be deleted and an RRSIG(SOA) RR for sn_n has to be added. In other words, any RRSIG(SOA) doesn't get any special treatment in the context of IXFR, and SOA RRs are used as "sentinels".

For example, if the IXFR server wants to transmit the changes from sn_o to sn_n in three chunks, based on two intermediary zone versions at sn_1 and sn_2 (where sn_o < sn_1 < sn_2 < sn_n), i.e., the chunk with the change information from sn_o to sn_1, the chunk from sn_1 to sn_2, and the chunk from sn_2 to sn_n, it would deliver in the incremental IXFR response packet(s) the following resource records (RRs), in order:

```
*  SOA for sn_n    # outer bracket
*  SOA for sn_o    # start of first chunk
*  {0 or more other RRs to be deleted from the zone at sn_o}
*  SOA for sn_1
*  {0 or more other RRs to be added for getting the zone at sn_1}
*  SOA for sn_1    # start of second chunk
*  {0 or more other RRs to be deleted from the zone at sn_1}
*  SOA for sn_2
*  {0 or more other RRs to be added for getting the zone at sn_2}
*  SOA for sn_2    # start of third chunk
*  {0 or more other RRs to be deleted from the zone at sn_2}
*  SOA for sn_n
*  {0 or more other RRs to be added for getting the zone at sn_n}
*  SOA for sn_n    # outer bracket
```

In contrast, in the case of fallback to AXFR, the IXFR response would convey, in order:

```
*  SOA for sn_n    # first instance
*  {one or more other RRs of the zone at sn_n, in arbitrary order}
*  SOA for sn_n    # repeated as trailing delimiter
```

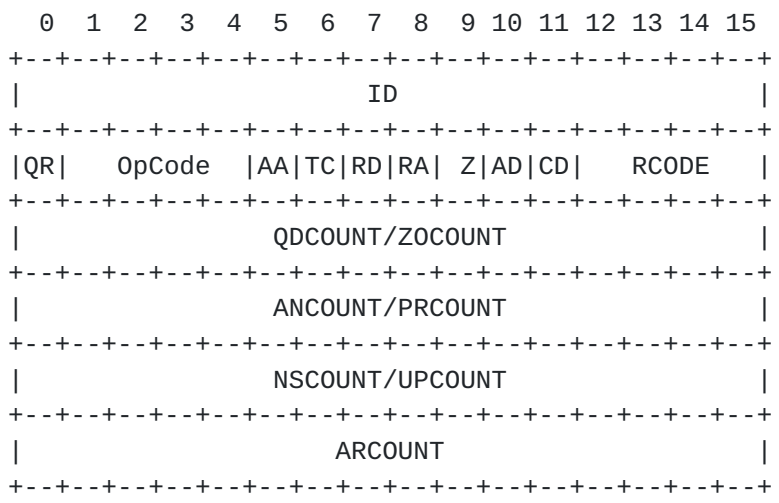
3. IXFR Messages

This section specifies the format of IXFR messages and the basic message generation and processing rules.

An IXFR session is started with an IXFR query message sent from an IXFR client to an IXFR server, which solicits one or more response messages in return.

All these messages adhere to the basic DNS message format as specified in RFC 1035 and later amended in various ways, for which Section 2 of RFC 5936 gives an expanded bibliography. Implementers should be aware of the considerations in RFC 5452, "Measures for Making DNS More Resilient against Forged Answers" [RFC5452], and follow the recommendations given there.

For convenience of the reader, the synopsis of the DNS message header from RFC 6195 [RFC6195] (and the IANA registry for DNS Parameters [DNSVALS]) is reproduced here informally:



This document makes use of the field names as they appear in this diagram. The names of sections in the body of DNS messages are capitalized in this document for clarity, e.g., "Additional section".

An IXFR session can be carried out over UDP (with tight restrictions -- see below) and over TCP. Thus, the DNS message size limit from RFC 1035 for DNS over UDP (and its extension specified in RFC 2671bis, "Extension Mechanisms for DNS (EDNS0)" [I-D.ietf-dnsext-rfc2671bis-edns0]) apply in the former case. BCP 145, "Unicast UDP Usage Guidelines for Application Designers" [RFC5405] contains valuable considerations regarding IP level fragmentation of UDP messages, and RFC 6274, "Security Assessment of the Internet Protocol Version 4" [RFC6274] contains a detailed security assessment of IPv4 segmentation and reassembly; both documents should be considered by implementers when deciding on the maximum size of DNS response messages over UPD supported by an IXFR server implementation. The upper limit on the permissible size of a DNS message over TCP is only restricted by the TCP framing defined in Section 4.2.2 of RFC 1035, which specifies a two-octet message length field, understood to be unsigned, and thus causing a limit of 65535 octets. This limit is not changed by EDNS0, and it applies to IXFR over TCP.

> Note that, independent of transport, the standard DNS message (name) compression facility (Section 4.1.4 of RFC 1035 [RFC1035]) severely limits the utility of DNS message sizes above 16k octets. The additional header overhead resulting from limiting IXFR response messages to 16k is negligible in comparison to the overhead resulting from the loss of ability to apply message compression in larger records.

3.1. IXFR Query

An IXFR query is sent by a client whenever it wants to obtain the delta information needed to update the content of a zone it is aware of (as identified by its SOA serial number) to the most recent version. The predominant trigger for this is the receipt of a DNS NOTIFY message [RFC1996], but it also can be triggered by other mechanisms or events, for instance as a result of a command line request, say for debugging. The details for these triggers are implemenation dependent and out of scope for this specification.

3.1.1. Header Values

The specification for AXFR query messages in Section 2.1.1 of RFC 5936 applies likewise for IXFR queries, with a single exception:

NSCOUNT Number of entries in Authority section; MUST be 1

3.1.2. Question Section

The Question section of an IXFR query MUST conform to Section 4.1.2 of RFC 1035, and it MUST contain (matching QDCOUNT=1 in the DNS message header) a single resource record with the following values:

QNAME the name of the zone requested

QTYPE one of the two pseudo-RR types for incremental zone transfer: IXFR (= 251) or IXFR-ONLY (= {TBD1}) [DNSVALS]

QCLASS the class of the zone requested [DNSVALS]

The choice of QTYPE depends on the intended IXFR server behavior in case the request cannot be fulfilled due to lack of stored information on the server, as detailed below in Section 3.2.

3.1.3. Answer Section

The Answer section MUST be empty, as indicated by ANCOUNT=0 in the DNS message header.

3.1.4. Authority Section

Corresponding to NSCOUNT=1 in the DNS message header, the Authority section MUST contain a single DNS resource record, the SOA record of the client's version of the zone content, identified by its serial number (denoted as sn_o in this document).

3.1.5. Additional Section

Currently, two kinds of resource records are defined that can appear in the Additional section of IXFR queries and responses: EDNS and DNS transaction security. Future specifications defining RRs that can be carried in the Additional section of normal DNS transactions need to explicitly describe their use with IXFR, should that be desired.

All considerations from Section 2.1.5 of RFC 5936 apply in the same manner for IXFR as they do for AXFR.

In order to support the extended RCODE assigned for CannotIXFR, EDNS0 support (RFC 2671bis [I-D.ietf-dnsext-rfc2671bis-edns0]) is REQUIRED for IXFR-ONLY, and an IXFR-ONLY query MUST be sent with an EDNS OPT RR in the Additional section of the query. Receipt of such query without an OPT RR SHOULD result in an error response with FormErr RCODE.

3.2. IXFR Response

An IXFR server that has received an IXFR query responds to it with an IXFR response addressed to the transport source identifier from which the query has been received, in particular using the same transport protocol.

An IXFR response consists of one or more response messages. If the IXFR query has been received over a connectionless transport (currently: UDP), the IXFR response MUST consist of a single message. If it is not possible to send the complete response in a single DNS message, a response MUST be sent that only contains the currrent SOA RR at the server; whose serial sn_n being larger than sn_o (in sequence space arithmetics) is the signal to the IXFR client to retry over connection-oriented transport (currently: TCP).

The conceptional "answer" carried in a multi-message response is the concatenation of the content of the Answer sections in these response messages, in the order they are sent; this is unambiguous from the point of view of the IXFR client as well, since the applicable connection-oriented transport preserves the order of messages sent.

If the server detects an error condition that makes it impossible to fulfill the request, it immediately sends an error response, that is a response message with non-zero RCODE. In case of connectionless transport (UDP), this is the single response message sent. In case of connection-oriented transport (TCP), the error condition might occur after one or more response messages already have been sent; in this case, the error message is sent as soon as need arises, and it will abort the ongoing IXFR session; i.e., the error message is the

last response message sent by the server. The special case of a server closing the TCP connection without sending an IXFR response message is covered in Section 3.3.

If an IXFR server is not able or willing to send the incremental zone change information to transform the client's version (sn_o) to its newer version (sn_n), the behavior is specified as follows: In the case of QTYPE=IXFR, the server SHOULD fall back to AXFR (see below). In the case of QTYPE=IXFR-ONLY, it SHOULD respond with an appropriate error, e.g., CannotIXFR (see below); however, in the (rather unlikely) corner case where a full zone transfer can be sent in a single response packet over connectionless transport (UDP), the IXFR server MAY instead proceed to send this even in response to an IXFR-ONLY query; doing so helps to achieve the overall performance optimization goals of IXFR-ONLY.

Any IXFR response that is neither an immediate error response nor a redirection to connection-oriented transport starts with the server's version of the SOA resource record, sn_n, and its kind can be determined from the second answer RR, if any. (See Sections 3.2.3 and 4 for a detailed discussion.)
If the server detects that the client's version is current (sn_o = sn_n) or already is more recent than at the server (sn_o > sn_n), or if the server detects that the entire response to an IXFR query received over connectionless transport (UDP) cannot be placed into a single response message, this SOA record is the only answer RR sent to the client. Otherwise, the subsequent answer RRs sent form a sequence of one or more change information chunks as described below in Section 4, and the final "sentinel" RR sent is another copy of the current SOA RR (sn_n).
In case of fallback to AXFR, the answer contains the same RRs (and is subject to the same ordering rules) as specified in Sections 2.2 through 3 of RFC 5936, with the single difference being the echoed QCODE (i.e., IXFR, or exceptionally -- as noted above -- IXFR-ONLY) in the Question section.

3.2.1. Header Values

The specification for AXFR in Section 2.2.1 of RFC 5936 applies likewise for IXFR queries, where in the case of IXFR the "new" behavior from RFC 5936 is always followed, i.e. the query ID from the IXFR query MUST be echoed in all IXFR response messages.

> Note that unlike with all common DNS responses, but like AXFR, the IXFR protocol makes no use of the TC (truncation) bit. To signal that an IXFR session needs to be retried over TCP, an IXFR server sends a response that in the Answer section solely contains its current version of the SOA RR for the zone.

The only additonal rule to be followed applies to the deliberations on the RCODE field in Note e) of Section 2.2.1 in RFC 5936: If the IXFR server is not able to fulfill an IXFR-ONLY request, it SHOULD respond with the extended RCODE CannotIXFR assigned on behalf of this document (see Section 10); see above for the exceptional corner case that allows to waive this requirement.

> Note that only the lower 4 bits of the conceptual RCODE can be carried in the RCODE message header field; the upper bits need to be placed into the EXTENDED-RCODE subfield of the "TTL" field in the OPT RR that, in this case, is REQUIRED in the Additional section of the response -- see Section 6.1.3 of RFC 2671bis [I-D.ietf-dnsext-rfc2671bis-edns0].

3.2.2. Question Section

In the first response message, the IXFR server MUST copy this section literally from the corresponding IXFR query message. For subsequent response messages, it MAY do the same or leave the Question section empty. However, if the last response message sent is an error message (RCODE unequal to 0), the query MUST also be copied. Accordingly, QDCOUNT in the DNS message header is set to either 1 (query copied) or 0 (otherwise).

When it is present, the content of this section MAY be used to determine the context of the message, that is, the name of the zone being transferred. The recipent (IXFR client) SHOULD apply the response verification rules from RFC 5452 [RFC5452].

Subsequent response messages with empty Question section are demultiplexed (among all open DNS transactions being carried out on a given transport connection, which already identifies the peer) to the proper IXFR session by means of the ID message header field only.

3.2.3. Answer Section

The Answer section MUST be populated with the zone change information or, in the case of fallback to AXFR, the full zone contents.

For multi-message IXFR responses, the conceptional answer is split into segments that are sent in order. Each segment is comprised of an integer number of full RRs, and for transport efficiency, the response messages SHOULD be filled up with answer RRs as much as possible for the response message size chosen by the IXFR server, taking into account the space needed for the other sections in the messages.

If an IXFR server intends to send a conceptual IXFR response that is comprised of at least two answer RRs, the first two answe RRs of the response MUST be sent in the first response packet to allow the IXFR client to immediately distinguish the kind of response coming in. An IXFR server MUST NOT send over connection-oriented transport the kind of (single-RR) IXFR response that indicates that the server has to send a multi-packet response and hence the IXFR request needs to be retried over connection-orientend transport (currently: TCP); this kind of response is only allowed in response to an IXFR query received over connectionless transport (currently: UDP).

See Section 4 below for the normative details of the various kinds of conceptual IXFR responses and the respective resource record ordering requirements, as well as how an IXFR client distinguishes these response kinds.

3.2.4. Authority Section

Corresponding to NSCOUNT=0 in the DNS message header, the Authority section in IXFR response messages MUST be empty.

3.2.5. Additional Section

All considerations from Section 2.2.5 (and hence, by reference, also Section 2.1.5) of RFC 5936 apply in the same manner for IXFR as they do for AXFR. See also Section 3.1.5 of this document.

Note that hereby the rules for any DNS transactional security meta-RRs (SIG(0)/TSIG) applicable for AXFR are implicitly extended to apply to multi-message IXFR responses as well; in the absence of explicit specification saying otherwise, this also holds for future DNS transactional security methods (if any).

3.3. Connection Aborts

In case of an IXFR session carried over connection-oriented transport (TCP), the considerations in Section 2.3 of RFC 5936 [RFC5936] apply similarly.

In a nutshell: Servers SHOULD avoid dropping active connections whenever possible, and clients nevertheless must be prepared to gracefully deal with such aborts.

4. Response Contents

This section describes the structure of the sequence of resource records (RRs) sent in IXFR reponses and how the IXFR client can immediately distinguish the various cases covered.

The possible IXFR responses can be classified into various kinds of responses by the number of answer RRs in the conceptual response.

(1) immediate-error: empty answer, non-zero RCODE;

(2) you-are-current: single SOA RR for sn_o = sn_n, RCODE = NoError;

(3) you-are-more-recent: single SOA RR for sn_o > sn_n, RCODE = NoError;

(4) redirect-to-conn: single SOA RR for sn_o < sn_n, RCODE = NoError;

(5) incremental: multiple RRs, starting and ending with the SOA RR for sn_n (where sn_o < sn_n), which enclose a sequence of one or more change information chunks, the first of which starts with the SOA RR for sn_o (which consequentially is different from sn_n);
this is detailed below in Section 4.1
(if need arises, a packetized incremental response can be aborted in the second or later response message by sending an empty Answer section and non-zero RCODE);

(6) full-xfr: multiple RRs, starting and ending with the SOA RR for sn_n (where sn_o < sn_n), enclosing a serialization of all other RRs in the zone for sn_n; this enclosed sequence of RRs is non-empty -- since any zone needs to contain at least one non-SOA RR, namely an NS RR --, does not contain any other SOA RR, and thus cannot start with another SOA RR;
this is detailed in [RFC5936]
(if need arises, a packetized full-xfr response can be aborted in the second or later response message by sending an empty Answer section and non-zero RCODE).

If the IXFR server discovers an error condition before it sends the first (or only) response message, the response content in the Answer section is empty, and consequentially, ANCOUNT is set to 0 in that message ("immediate-error" response, kind (1) above).

Otherwise, the response content starts with a copy of the current SOA RR at the IXFR server (sn_n). There are several cases:

a. The server serial (sn_n) is the same as the client serial (sn_o) sent in the Authority section of the IXFR query. In this case, this SOA RR is the only RR in the response message, indicating to the IXFR client that it already has the current zone content ("you-are-current" response, kind (2) above).

b. The server serial (sn_n) is older than the client serial (sn_o) sent in the Authority section of the IXFR query, and this SOA RR is the only RR in the response message. This indicates that the zone content held at the IXFR server actually lags behind the IXFR client, and so the IXFR server cannot provide an update to the zone data at the client ("you-are-more-recent" response, kind (3) above).

c. The server serial (sn_n) is newer than the client serial (sn_o) sent in the Authority section of the IXFR query, and this SOA RR is the only RR in the response message. This indicates to the IXFR client that its zone content is outdated and the IXFR server is willing to send the incremental zone change information, but is unable to do so in a single response message due to message size limitations.

   An IXFR server MUST NOT send this type of IXFR response over connection-oriented transport (TCP), but it MAY use this type of response over connectionless transport (UDP) to indicate to the IXFR client that it should retry the IXFR query over connection-oriented transport ("redirect-to-conn" response, kind (4) above).

   An IXFR client that receives over connection-oriented transport an IXFR response message (as the first response message related to its IXFR query) that contains only a single SOA RR with sn_n higher than sn_o MUST discard the response message (see below).

   Note again that the "truncated response message" mechanism specified in RFC 1035, which is signalled by setting the TC bit in a response message, MUST NOT be used in an IXFR response. An IXFR client that receives an IXFR response message with the TC bit set MUST discard the message (see below for details).

d. The server serial (sn_n) is newer than the client serial (sn_o) sent in the Authority section of the IXFR query, and this SOA RR is followed by another SOA RR in the same response message. In this case, the IXFR response is an incremental response (kind (5) above), as detailed in Section 4.1 below.

   If this second SOA RR also is for sn_n, a robust IXFR client will assume that the server exposes behavior arguably not explicitly forbidden in RFC 1995 to signal case (a) above. Hence, an IXFR client MAY accept for resiliency an IXFR response with exactly these two copies of the same SOA RR sent in a single response message as an "empty incremental response" indicating that the client's version of the zone is current; otherwise, the client MUST discard a response starting with two copies of the sn_n SOA RR.

Otherwise, if the second SOA RR is for sn_o, this indicates the start of an ordinary incremental response as detailed below.

Otherwise (if the second SOA RR is for sn_x that differs from both sn_o (as sent by the client) and sn_n (in the first SOA RR), the client MUST discard the response message as bogus.

e. The server serial (sn_n) is newer than the client serial (sn_o) sent in the Authority section of the IXFR query, and this SOA RR is followed by another, non-SOA RR in the same response message.

This indicates a non-incremental response ("full-xfr" response, kind (6) above, colloquially "fallback to AXFR"). In this case, the response content is structured like an AXFR response, as described in RFC 5936 ("new" behavior, no backward compatibility kludges admitted); following the initial SOA RR it contains the entire zone content besides the SOA RR and ends with a second copy of the sn_n SOA RR as an end-of-response marker.

Such non-incremental IXFR response MUST NOT be sent in response to an IXFR-ONLY query unless the entire intended response -- up to and including the trailing sentinel sn_n SOA RR -- fits into a single response message with a size that allows it to be sent over connectionless transport (UDP), or would have allowed that if it actually is carried over connection-oriented transport (TCP). An IXFR client that receives an incomplete initial IXFR response message that indicates such non-incremental response to an IXFR-ONLY query MUST discard the message as bogus.

An IXFR client MUST discard any IXFR response that does not match one of the forms described as kinds (1) through (6) above and is not recognized by the above decision tree.

Whenever in the above cases the text says that the IXFR client "MUST discard the message", the following behavior is implied: The IXFR client MUST regard the IXFR session as terminated; this results in subsequent silent discarding of further response messages that still pretend to belong to the same IXFR session by means of the query ID and the echoed Question (if present), because the IXFR client does not maintain corresponding IXFR query/session state any more. The IXFR client MAY log the event and SHOULD regard the IXFR server as broken; hence, it SHOULD refrain from using the same IXFR server again -- at least for considerable time, or until the usage has been reinstated by an implementation-dependent management interaction.

From the above decision tree for the client it also follows that, to allow unambiguous client behavior, if an IXFR server has to send a response comprised of two or more RRs, it MUST send at least the

first two RRs in the first response message, as specified in Section 3.2.3 above.

If the IXFR server discovers an error condition lately, after having sent one or more response messages (all with RCODE set to 0), it can abort the IXFR session by sending another response message with empty Answer section and a non-zero RCODE. This MUST be the last message sent in response to the IXFR query, that is, this error message aborts the ongoing IXFR session.

The above rules ensure that an IXFR client can unambiguously determine the kind of IXFR response from the first response message alone. This encludes the ability to immediately detect the end of any legitimate single-message IXFR response. An IXFR session with a multi-message IXFR response ends when either

- a late error message arrives, i.e., a response message with non-zero RCODE -- such message has an empty Answer section so that there's no ambiguity as to which answer RRs might be regarded as somehow valid anyhow (see Section 7.1 for the possible treatment of RRs received previously in such IXFR session); or
- in the case of an incremental response, the third instance of the SOA RR for sn_n is found in the answer (see Section 4.1 for the rationale); or
- in the case of a non-incremental response, the second instance of the SOA RR for sn_n is found in the answer.

Recall that in the second and third case, the SOA RR for sn_n was the first RR sent in the first response message. If in these two cases, the received response message contains more, extraneous RRs past that sentinel SOA RR, an IXFR client MAY regard the entire response (or the not yet committed part of the response, according to Section 7.1) as bogus (see above); if the client accepts an otherwise consistent response under these circumstances, it MUST discard the extraneous RRs, and it MAY log the error and take 'discard' actions as above.

4.1. Incremental Responses

In an incremental response, the leading sn_n SOA RR is followed by one or more change information chunks and concluded by a final copy of the sn_n SOA RR -- in total, from the details given below, it turns out that this is always exactly the third instance of that SOA RR in the IXFR response.

Each change information chunk describes the changes to be performed on a given "origin" version of the zone to obtain a "target" version of the zone (i.e., one SOA serial change of the zone). It consists of (1) a set of old RRs to be deleted from the "origin" zone version and (2) a set of new RRs to be added after these deletions to obtain the "target" version of the zone. (In this model, a change in a

single RR is represented by an RR deletion followed by an RR addition.) These two sets are sent in this order, with each set serialized as a sequence of the related SOA RR followed by other, non-SOA RRs in a arbitrary order. This way, each SOA RR indicates the end of the sequence of (zero or more) non-SOA RRs that precedes it, and at the same time it either starts the next set of RRs or is the trailing sn_n SOA of the response.

The "origin" sn of each change information chunk MUST precede its "target" sn in the sense of sequence number arithmentics.

The change information chunks in an incremental response MUST be ordered oldest first, newest last; in more detail: The first change information chunk in an incremental response must have the client's version (sn_o) as its origin sn; the origin sn of each subsequent change information chunk MUST be the same as the target sn of the preceding chunk, and the last change information chunk in an incremental response MUST have the server's version (sn_n) as its target sn. This "chaining" of chunks ensures that the client can correctly construct the sn_n version from the sn_o version it holds by conceptionally applying single-RR deletions and additions in the order the RRs appear in the IXFR response.

Note that, as a consequence of the aforementioned rules, a valid incremental IXFR response MUST contain exactly one copy of the sn_o SOA RR (sent as the second RR in the response) and exactly three copies of the sn_n SOA RR -- one as the first RR in the response, one as the leading RR of the second sequence (set of RRs to be added) in the last change information chunk, and one as the final "sentinel" RR that indicates the end of the response contents. Likewise, each "intermediate" SOA RR (with sn_o < sn < sn_n) will appear exactly twice, once in the second part (new RRs) of a particular change information chunk, and once in the first part of the immediately following change information chunk.

Any IXFR response classified as a (non-empty) incremental response by the decision tree presented above in Section 4 that violates any of the above rules MUST cause the IXFR client to regard the response as bogus; it MUST discard a response message in case its content allows the client to detect such violation, with the caveats for "discard" given in Section 4.

5. Transport

IXFR servers and IXFR clients MUST support transport over UDP and TCP (see RFC 5966 [RFC5966]). As with all DNS transactions, IXFR responses MUST be sent on the same transport association over which the query arrives at the server.

If an IXFR server cannot send a full IXFR response for an IXFR query received over UDP within a single response message over UDP due to message size limitations, it MUST return the special form of response that indicates to the client to retry over TCP (single-RR response with the server SOA only, as described in Sections 3.2 and 4).

Given the limitation of the basic DNS message size over UDP to 512 octets, it is strongly RECOMMENDED that implementations of IXFR servers and IXFR clients support RFC 2671bis, "Extension Mechanisms for DNS (EDNS0)" [I-D.ietf-dnsext-rfc2671bis-edns0] and choose extended DNS message size limits appropriate for their environment. The default behavior of IXFR clients regarding the EDNS message size, and the maximum accepted by servers, SHOULD both be configurable.

The considerations for AXFR transport over TCP in Section 4 of RFC 5936 [RFC5936] apply similarly for IXFR. However, IXFR is commonly being used much more frequently than AXFR between a given pair of authoritiative servers, and often not authorized for use by servers outside the set of authorities for a zone, which are all under the control of a single administrative domain or a small number of cooperating administrative domains. In this environment, it might make sense for the sake of efficiency to maintain (and reuse) persistent TCP connections between the configured IXFR peers. Therefore, implementations of IXFR should allow to configure relatively high TCP User Timeout values and support the TCP UTO mechanism ([RFC5482]) that allows the peers to exchange their view of the TCP User Timeout and adapt the behavior of their TCP accordingly. To minimize unnecessary delays, IXFR server implementations SHOULD use available API functions to cause their TCP stacks to immediately dispatch for sending the first and last packet of any IXFR response and cause these to be delivered immediately to the recipient; for instance, immediate sending and setting of the 'PSH' TCP header flag in outgoing packets can often be achieved using the TCP_NODELAY socket option.

6. Server Behavior

6.1. General

General considerations on IXFR server behavior, in particular response message generation and packet processing, are spread all over this document; in particular, see Sections 3.2 and 4.

In addition to the current zone content, IXFR servers need to maintain history information on the zone content that enables them to respond with incremental responses for a sufficient range of versions. What is considered "sufficient" and how this history information is maintained, is a local matter. It may be appropriate

to maintain the history information on stable storage as well to make it available spanning restarts of an IXFR server, as it is already required for the current zone content.

6.2. Purging Strategy

An IXFR server cannot be required to hold all previous versions forever and may delete them anytime. In general, there is a trade-off between the size of storage space and the possibility and need of using IXFR.

Information about older versions should be purged as soon as the total length of an IXFR response would otherwise become larger than that of an AXFR response. Given that the purpose of IXFR is to reduce AXFR overhead, this strategy is quite reasonable. The strategy assures that the amount of storage required is at most twice that of the current zone information.

Information older than the SOA expire period may also be purged.

Once the current serial number of a zone advances so far that older serial numbers can no more be recognized immediately as older versions (under the rules of sequence space arithmetics), such previous versions MUST NOT be held available any more for IXFR purposes. In order to account for the propagation delay along the IXFR distribution graph, use of a reasonable safety margin against this hard limit has proven to be a good strategy for defensive implementation practice -- for instance implementations could limit the maximum serial number span made available for IXFR to a specific fraction of the 32-bit serial number range, e.g., one fourth (2^30).

The Condensation techniques explored below in Section 6.3 might pose an opportunity to get rid of more recent, yet less relevant history information and as such might allow to cover a larger span of SOA versions than otherwise possible within the same amount of storage.

6.3. Optional Condensation of Zone Changes

An IXFR server MAY optionally condense a number of immediately succeeding change information chunks into a single chunk, thus dropping information on intermediate zone versions.

This may be beneficial if a lot of versions, not all of which are useful, are generated. For example, if multiple ftp servers share a single DNS name and the IP address associated with the name is changed once a minute to balance load between the ftp servers, it is not so important to keep track of all the history of changes.

Another example is where statefully managed client systems get IP addresses assigned dynamically by DHCP servers, and where the DHCP server(s) and/or the clients register their current contact information via DNS UPDATE whenever leases are given out or renewed. These transactions could be comprised of several independent update steps, for forward and reverse address resolution, for service discovery, etc., where multiple parts of the related information are maintained in the same zone. Intelligent condensation strategies might be able to identify subsequent incremental changes related to a single end-user system and collapse this information in a single change information chunk.

But this feature may not be so useful if an IXFR client has access to two IXFR servers, A and B, with inconsistent condensation results. The current version of the IXFR client, received from server A, may be unknown to server B. In such a case, server B cannot provide incremental data from the unknown version and a full zone transfer is necessary. Therefore, it is highly desirable that alternative IXFR servers for a given set of IXFR clients expose similar (or at best, the same) condensation behavior.

Condensation can be performed in two stages, perhaps in a complementary manner: Firstly, the history information stored on an IXFR server can be condensed to reduce storage requirements *and* IXFR response sizes to some degree. Additionally, IXFR servers can perform condensation "on the fly" in preparing IXFR responses; this might provide additional savings in IXFR response size while reducing the likelihood that IXFR queries cannot be responded with incremental responses due to the requested sn being "condensed out" of the stored history information.

Condensation is completely optional. Clients cannot detect from the response whether or not the server has condensed the reply.

For interoperability, IXFR servers, including those without the condensation feature, SHOULD NOT send an error response in case they receive a client's IXFR request with an unknown version number and SHOULD, instead, attempt to perform a full zone transfer. Of course, this in general does not apply if the client indicates its desire to try its luck in such case at another candidate IXFR server, by initiating the request with IXFR-ONLY (the single exception to this general case is the corner case discussed in Section 3.2).

6.4. Authorization

The considerations for AXFR presented in Section 5 of RFC 5936 [RFC5936] apply in a similar fashion for IXFR.

Given the basic desire for frequent use and the resulting processing load, operational considerations will, even more likely than for AXFR, dictate the need to closely restrict the usage of IXFR to the set of authoritiative servers for a given zone, and to precisely configure the IXFR distribution graph within the set of servers, by means of access lists on the server side and by configuring a prioritized IXFR server search list on the client side.

Since IP addresses can be spoofed rather trivially in large parts of the open Internet, better authentication methods are needed as a base for authorization decisions unless the IXFR distribution graph can be restricted to protected networks under control of the same administration as the participating DNS servers.

In particular, as detailed in the Section of RFC 5936 quoted above, implementations of IXFR SHOULD also support at least one flavor of DNS transaction security. Virtual private networks, virtual LANs, IPsec ([RFC4301]), and TCP-AO ([RFC5925] might also be applicable solutions to ensure proper authentication to base authorization decisions on. See Section 9 for more information.

7. Client Behavior

It is RECOMMENDED that IXFR client implementations supporting IXFR-ONLY allow to configure its usage per IXFR server, as part of the IXFR distribution graph configuration.

An IXFR client SHOULD set an appropriate guard timeout whenever the content of a response message indicates that this is not the final message of an IXFR response. In case this timeout period elapses without another response message arriving, it SHOULD regard the IXFR session as failed and apply the caveats for the "discard" case presented in Section 4.

If it is known to the IXFR client that an IXFR server conforms to the refined IXFR specification in this RFC, the guard timeout can be chosen rather large because the kind of IXFR response is unambiguously indicated in the first response message and the timing of the subsequent packet flow should be left to the connection-oriented transport in use, and the timeout only serves as a "last defense" in case of fatal failures not detected by the transport.

7.1. Zone Integrity

The elaborations on Zone Integrity for AXFR in Section 6 of RFC 5936 [RFC5936] apply in a similar fashion for IXFR.

However, during the receipt of an incremental IXFR response, and upon successful processing of an SOA RR that serves as a sentinel for the end of any change information chunk, an IXFR client MAY immediately apply and commit to stable storage the SOA serial number change described by that chunk (and previous chunks, if not already done). This operation MUST externally appear as an atomar operation.

Before this operation can be attempted, the IXFR client SHOULD apply all feasable sanity checks for the change information chunk under consideration. In particular, it SHOULD verify that the RRs contained in the first part of the chunk (those to be deleted) are indeed literally contained in the data set for the most recent zone version the client has constructed so far. If a DNSSEC-aware IXFR client receives an IXFR response for a zone secured with DNSSEC [RFC4033], it MAY try to verify any RRSIG RR for the new zone SOA (received in the second part of the change information chunk), as another means to detect forged responses, and in case of failure forcibly abort the IXFR session. However, in order to avoid DoS attacks targeted at processing resources and amplification attacks, this SHOULD NOT be done if the IXFR session is secured by other means (in-band by TKEY/TSIG, in lower layers, e.g. by IPsec or other VPN technology) or if the necessary keys are unavailable (not already cached) and/or not already verified. Similarly, like in the case of AXFR, it is generally NOT RECOMMENDED to perform a full cryptographic verification of the new zone version -- which would consume very substantial computing resources, hence clearing the way for another type of DoS attack.

8. Backwards Compatibility

Despite a few potentially misleading statements in the previous specification, only a single detail has been identified so far that could give rise to backward compatibility concerns. This is addressed by the compatibility rules in Section 4 that allow an IXFR client to process an "empty incremental response" consisting of only a pair of instances of the server's SOA RR.

The clarified and slightly tightened packetization requirements contained in Section 3.2.3 might cause an IXFR client to reject a poorly packetized multi-packet response previously sometimes regarded as acceptable under RFC 1995. An IXFR client implementation MAY provide a configuration option (globally, or per IXFR server) to admit such inefficient behavior on the server side, in which case it needs to wait for a second response message before it can distinguish unambiguously all response kinds (including protocol violations). In deployment scenarios with multiple candidate IXFR servers where expeditious switching to an alternate IXFR server (if needed) is intended, activation of such option would be detrimental.

The introduction of IXFR-ONLY creates further interoperability considerations. An IXFR server utilizing IXFR-ONLY may receive an error response different from CannotIXFR persistently. (The actual RCODE reveived may depend on whether or not the server is aware of the allocation of the range of RR types set aside for Q-Types in [RFC6195] (and its predecessors), from which the IXFR-ONLY code point has been assigned.) This event likely indicates that the IXFR server chosen does not support IXFR-ONLY. In such case, the client will mark the server as "unusable of IXFR-ONLY" in his server list and try another potential IXFR server, or, if all candidates fail, retry the scan with bare IXFR, or alternatively try to immediately start an AXFR session. The latter should always be the method of last resort in case of persistent IXFR failures.

9. Security Considerations

This document presents a more detailed specification for the mechanism previously specified in RFC 1995, which has similar protocol behavior and security properties as the AXFR mechanism described in RFC 5936. Hence, beyond the general security considerations for the DNS laid down in RFC 3833 [RFC3833], similar considerations apply.

Thus, the sections on Transport, Authorization and Zone Integrity that all include by reference the respective sections of RFC 5936 [RFC5936] largely address the relevant concerns. Deployments of IXFR might be interested in using large values for the EDNS message size and thereby become more exposed to the various security threats against IP fragmentation; these and suitable mitigations are discussed in [RFC6274].

Since IXFR is likely to be used in a more frequent and continuous manner and hence a possible candidate for making use of long-lived, persistent TCP connections for its transport, besides IPsec (RFC 4301 [RFC4301]), the more lightweight TCP Authentication mechanism described in RFC 5925 (TCP-AO, [RFC5925]) might, once deployed, be a suitable candidate for peer authentication and integrity protection of IXFR sessions.

10. IANA Considerations

The IANA Registry "Domain Name System (DNS) Parameters" [DNSVALS] contains a sub-registry "Resource Record (RR) TYPEs", in which [RFC6195] has reserved the range 128 through 255 for pseudo-RRs only being used in DNS queries, for short "Q-Types". This partial namespace is managed under the "DNS RRTYPE Allocation Policy" specified in RFC 6195 [RFC6195].

[[This paragraph to be deleted on publication as an RFC]]
IANA and IESG: based on the provisions of RFC 6195, we ask for RFC 4020 early allocation of the two code points needed for this memo, as described below.

Since RFC 1995, the Q-Type 251 has been assigned to IXFR. Upon publication of this memo as an RFC, IANA will update / has updated the description for that entry to say "incremental zone transfer" and the Reference for that entry to point to this RFC.

Upon publication of this memo as an RFC, IANA also will assign / has assigned the Q-Type {TBD1} to the TYPE mnemonic IXFR-ONLY, with description "incremental zone transfer w/o fallback", and pointing to this memo.

The IANA Registry "Domain Name System (DNS) Parameters" [DNSVALS] contains a sub-registry "DNS RCODEs", which is managed under "IETF Review" assignment policy, as specified in RFC 6195 [RFC6195]. IANA is requested to allocate / has allocated from that sub-registry a new Extended RCODE value (above 16, only usable with EDNS) for CannotIXFR:

RCODE Decimal	Name	Description	Reference
{TBD2}	CannotIXFR	IXFR not possible, would fall back	[RFCthis]

11. Acknowledgements

Masataka Ohta is acknowledged for his original work as the author of RFC 1995 [RFC1995], as are the contributors listed in the Acknowledgements section of that RFC. Andreas Gustafsson has initiated the first attempt to clarify RFC 1995 in November 1999 and contributed text to the DNSIND WG for a proposed document revision. Masataka Ohta's subsequent draft [OLD-IXFRbis] has not been pursued until RFC publication; thanks to Andreas for eventually bringing this earlier work to the attention of the authors of this memo.

The specification of IXFR-ONLY in this document is based on the original proposal [I-D.kerr-ixfr-only], in which the operational need for this behavior has been identified and carried into the IETF.

The DNSEXT working group and its predecessor (DNSIND) are acknowledged for their discussion on the above documents. Substantial text has been borrowed from there and from [RFC5936].

Discussions of the draft on the dnsext list have directed the evolution of this document; in particular, we acknowledge (in

alphabetical order) Mark Andrews, Ray Bellis, Brian Dickson, Andreas Gustafsson, Shane Kerr, Warren Kumari, Edward Lewis, Josh Littlefield, Masataka Ohta, Paul Vixie, Florian Weimer, and Wouter Wijngaards for their comments and reviews.

12. References

12.1. Normative References

[I-D.ietf-dnsext-rfc2671bis-edns0]
Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS0)", draft-ietf-dnsext-rfc2671bis-edns0-08 (work in progress), February 2012.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5405] Eggert, L. and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers", BCP 145, RFC 5405, November 2008.

[RFC5452] Hubert, A. and R. van Mook, "Measures for Making DNS More Resilient against Forged Answers", RFC 5452, January 2009.

[RFC5936] Lewis, E. and A. Hoenes, "DNS Zone Transfer Protocol (AXFR)", RFC 5936, June 2010.

[RFC6195] Eastlake, D., "Domain Name System (DNS) IANA Considerations", BCP 42, RFC 6195, March 2011.

12.2. Informative References

[DNSVALS] IANA Registry, "Domain Name System (DNS) Parameters", protocol parameter registry available at:, <http://www.iana.org/assignments/dns-parameters>.

[I-D.kerr-ixfr-only]
Sury, O. and S. Kerr, "IXFR-ONLY to Prevent IXFR Fallback to AXFR", draft-kerr-ixfr-only-01 (work in progress), February 2010.

[OLD-IXFRbis] Ohta, M., "Incremental Zone Transfer in DNS", draft-ietf-dnsext-ixfr-01 (work in progress), June 2000.

[RFC1995] Ohta, M., "Incremental Zone Transfer in DNS", RFC 1995, August 1996.

[RFC1996] Vixie, P., "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)", RFC 1996, August 1996.

[RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS Specification", RFC 2181, July 1997.

[RFC3833] Atkins, D. and R. Austein, "Threat Analysis of the Domain Name System (DNS)", RFC 3833, August 2004.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC5482] Eggert, L. and F. Gont, "TCP User Timeout Option", RFC 5482, March 2009.

[RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, June 2010.

[RFC5966] Bellis, R., "DNS Transport over TCP - Implementation Requirements", RFC 5966, August 2010.

[RFC6274] Gont, F., "Security Assessment of the Internet Protocol Version 4", RFC 6274, July 2011.

Appendix A. Motivation for IXFR-ONLY

IXFR is an efficient means to transfer changes in zones from IXFR servers to IXFR clients. However, when an IXFR client has multiple IXFR servers for a single zone, it is possible that not all IXFR servers hold the zone content with the same serial number(s). In this case, if an IXFR client attempts an IXFR from an IXFR server that does not have the zone content with the serial number used by the IXFR client, the IXFR server will fall back to a full zone transfer (like in AXFR) when it has a version of the zone with serial number greater than the serial requested by the IXFR client.

For example, IXFR server NS1 may have serial numbers 1, 2, and 3 for a zone, and IXFR server NS2 may have serial numbers 1 and 3 for the same zone. An IXFR client that has the zone content with serial number 2 and sends an IXFR request to IXFR server NS2 will get a full zone transfer (AXFR style) of the zone at serial number 3. This is because NS2 does not know the zone with serial number 2, and therefore is not able to report the differences between the zone with serial number 2 and 3.

If the IXFR client in this example had known to send the query to IXFR server NS1, then it could have gotten an incremental transfer. But an IXFR clients can only know what the *latest* version of the zone is at an IXFR server -- this information is available via an SOA query.

The IXFR-ONLY query type provides a way for the IXFR client to ask each IXFR server to return an error instead of sending the current version of the zone via full zone transfer. By using this, an IXFR client can check each IXFR server until it finds one able to actually provide an incremental transfer. If it doesn't succeed, it can fall back and try with bare IXFR instead of IXFR-ONLY, or it can immediately start an AXFR session with an AXFR server of its choice (the preferred AXFR server might be distinct from the most prefered IXFR server).

By providing IXFR-ONLY support, the policy control over the zone synchronization operation switches to the client side, which is preferable under various operational settings.

Appendix B. Substantial Changes Since RFC 1995

This is a summary of the substantial changes since RFC 1995 [RFC1995].

- Corrected a few technical flaws: incorrect comparison with AXFR; improper implied requirement of performing SOA query over UDP; improper reference to transfer of partial RRs in a response message corrected (to be read as transfer of partial RRsets in a response message -- as it has always been understood by implementers, since STD 13 requires only entire RRs to be present in DNS messages).

- New specification based on the revised AXFR specification, RFC 5936 [RFC5936].

- Slightly tightened packetization requirements for multi-packet responses to ensure more timely client-side processing (immediate determination of kind of response based upon first response packet, faster determination of protocol violations).

- Many clarifications and details supplied, text vastly reorganized and expanded, but no other (intentional) technical deviation from the previous specification, as understood by implementers.

- Addition of new IXFR-ONLY protocol variant, based on operational experience and perceived need.

- Major additions to Security Considerations.

- Historical example dropped (incompatible with IESG policy on examples). Instead, abstract examples have been added to Section 2.

Appendix C. Change Log of WG Draft

[[Entire section to be deleted before publication as an RFC.]]

The changes of the individual draft (draft-ah-dnsext-rfc1995bis-ixfr) up to adoption as a WG draft (draft-ietf-dnsext-rfc1995bis-ixfr-00) are detailed in Appendix C of the latter, but not reproduced below any more.

C.1. Draft Version 00 -> 01

- Removed text pertaining to individual draft stage.
- Several text adoptions based on evaluation of previous work in DNSIND & DNSEXT (in 1999 & 2000) made available by Andreas Gustafsson.
- Added more text to s1.2 regarding motivation for this memo.
- Added to s1.4 hint to spelling conventions for DNS message parts (borrowed from RFC 5936); clarified there that IXFR is a Q-Type RR.
- s2: Clarified preconditions on SOA serial numbers with quotes from STD 13; s6.2: IXFR server now MUST NOT hold available via IXFR zone versions that cannot clearly be recognized as older than the current version; with regard to propagation delays, text recommends safety margin: limit SOA serial span covered to e.g. 1/4 of the serial number space (i.e. 2^30).
- s2, s3.2, s4: Draft now accommodates the (hopefully rare) case of the IXFR server's zone being older than the version that the client has.
- s2, s4: Draft now strictly follows RFC 5936 RR ordering requirements for non-incremental responses (i.e. no more recommendation for grouping RRs into RRsets). Example in s2 now also clearly indicates possibility of having zero "other" RRs to be deleted or to be added in a change information chunk.
- Clarified which kinds of responses are covered by last para in s3.2.

- s3.2, s3.2.1: Resolved text that has become inconsistent by the introduction (per the last individual draft version) of the (optional) exceptional single-packet non-incremental IXFR-ONLY response.
- s3.2.2: Clarified that subsequent packets of multi-packet IXFR responses with empty Question section are demultiplexed into the appropriate IXFR session by only the DNS message header ID.
- s3.2.3: Based on feedback emphasizing the importance of rules that ensure efficient protocol operation and thereby making the requirements at least as strong as for AXFR, the requirements on packetization of multi-packet IXFR responses have been clearly spelled out and clarified, making the requirements comparable of what is specified for AXFR in RFC 5936; previously, the slightly tightened requirements were arguably hidden in other parts of the text.
- s3.2.5: Added explicit note regarding DNS transactional security.
- s4: Added classification of IXFR response kinds. Updated decision tree to refer to these kinds. Added discussion of end-of-session detection for multi-packet IXFR responses.
- s4.1: Dropped last para -- new packetization requirements now are limited to what is made explicit in s3.2.3.
- s5: Added note on TCP_NODELY socket option / PSH flag for TCP.
- s7: Added discussion of precautionary receive timeout.
- s8: Added elaborations on backward compatibility regarding the slightly strenghtened packetization requirements from s3.2.3.
- Updated Acknowledgements; added Informative Ref. to expired '2000 IXFR-bis draft.
- Addressed review comments from Lubos Slovak.
- Updated Appendix B with regard to above changes.
- Removed old Appendix C, added new Appendix C.
- Numerous editorial corrections and improvements.

Authors' Addresses

Alfred Hoenes
TR-Sys
Gerlinger Str. 12
Ditzingen D-71254
Germany

EMail: ah@TR-Sys.de

Ondrej Sury
CZ.NIC
Americka 23
120 00 Praha 2
CZ

Phone: +420 222 745 110
EMail: ondrej.sury@nic.cz

Shane Kerr
ISC
Bennebrokestraat 17-I
Amsterdam NL-1015 PR
Netherlands

Phone: +31 64 6336297
EMail: shane@isc.org