DNS Zone Transfer Protocol (AXFR)

Introduction

A mechanism for prompt notification of zone changes (NOTIFY) is defined in "A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)" [1]. This document defines a new definition of the AXFR, new in the sense that it is recording an accurate definition of an interoperable AXFR mechanism.

Scope

The Domain Name System standard facilities for maintaining coherent servers for a zone consist of three elements. The Authoritative Transfer (AXFR) is defined in RFC 1034 and RFC 1035. The Incremental Zone Transfer (IXFR) is defined in RFC 1995. A mechanism for prompt notification of zone changes (NOTIFY) is defined in RFC 1996. The base definition of these facilities, that of the AXFR, has proven insufficient in detail, resulting in no implementation complying with it.

Context

Yet today we have a satisfactory set of implementations that do interoperate. This document is a new definition of the AXFR, new in the sense that it is recording an accurate definition of an interoperable AXFR mechanism.

Abstract

The Domain Name System standard facilities for maintaining coherent servers for a zone consist of three elements. Authoritative Transfer (AXFR) is defined in "Domain Names - Concepts and Facilities" [RFC1034] (referred to in this document as RFC 1034) and "Domain Names - Implementation and Specification" [RFC1035] (aka RFC 1035). Incremental Zone Transfer (IXFR) is 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.

Updates:

This Internet-Draft will expire on December 1, 2008.

Copyright Notice

Copyright (C) The IETF Trust (2008).

Key words for use in requirements documents and the meaning of these key words are defined in RFC 2119 [BCP14].

Abstract

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 "Key words for use in requirements documents" [RFC2119].

"Newer"/"New" DNS and "older"/"old" DNS refers to implementations written after and prior to the publication of this document.

1.1 Definition of Terms

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 "Key words for use in requirements documents" [RFC2119].

"Newer"/"New" DNS and "older"/"old" DNS refers to implementations written after and prior to the publication of this document.

1.2 Scope

The Domain Name System standard facilities for maintaining coherent servers for a zone consist of three elements. Authoritative Transfer (AXFR) is defined in "Domain Names - Concepts and Facilities" [RFC1034]. 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. As far as it is known, AXFR, IXFR and NOTIFY are the only mechanisms that provide an interoperable solution to the desire for coherency within the definition of DNS, they certainly are the only mechanisms documented 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 those configurations, a coherency mechanism as described here would be unsuitable.

"General purpose DNS implementation" refers to DNS software developed for wide-spread use. This includes resolvers and servers freely accessible as libraries and standalone processes. This also includes proprietary implementations used only in support of DNS service offerings.

"Turnkey DNS implementation" refers to custom made, single use implementations of DNS. Such implementations consist of software that employs the DNS protocol message format yet do not conform to the entire range of DNS functionality.

A DNS implementation is not required to support AXFR, IXFR and NOTIFY. A DNS implementation SHOULD have some means for maintaining name server coherency. A general purpose DNS implementation SHOULD include AXFR, IXFR and NOTIFY, but turnkey DNS implementations MAY operate without it.

1.3 Context

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."
Besides describing the mechanisms themselves, there is the context in which they operate to consider. When AXFR, IXFR and NOTIFY were defined, there was little consideration given to security and privacy issues. Since the original definition of AXFR, new opinions have appeared on the access to an entire zone’s contents. In this document, the basic mechanisms will be discussed separately from the permission to use these mechanisms.

1.4 Coverage

This document concentrates on just the definition of AXFR. Any effort to update the IXFR or NOTIFY mechanisms would be done in different documents. This is not strictly a clarification of the definition in RFC 1034 and RFC 1035. This document will update those sections, and invalidate at least one part of that definition. The goal of this document is to define AXFR as it exists, or is supposed to exist, currently.

2 AXFR Messages

An AXFR message exchange (or session) consists of an AXFR query message and a set of AXFR response messages. In this document, AXFR client is the sender of the AXFR query and the AXFR server is the responder.

(Use of terms such as master, slave, primary, secondary are not important to defining the AXFR exchange.) The reason for the imbalance in number of messages derives from large zones whose contents cannot be fit into the limited permissible size of a DNS message.

An important aspect to keep in mind is that the definition of AXFR is restricted to TCP [RFC0793]. The design of the AXFR process has certain inherent features that are not easily ported to UDP [RFC2764].

The basic format of an AXFR message is the DNS message as defined in RFC 1035, Section 4.4.3, updated by the following:

- "A Mechanism for Prompt Notification of Zone Changes (AXFR)" [RFC1996]
- "Domain Name System (DNS) IANA Considerations" [RFC2595]
- "Dynamic Updates in the Domain Name System (DNS UPDATE)" [RFC2136]
- "Extension Mechanisms for DNS (EDNS0)" [RFC2671]
- "Secret Key Transaction Authentication for DNS (TSIG)" [RFC2845]
- "Secret Key Establishment for DNS (TKEY RR)" [RFC2847]
- "Obsoleting IQEVRX" [RFC3425]
- "Handling of Unknown DNS Resource Record (RR) Types" [RFC3597]
- "Protocol Modifications for the DNS Security Extensions" [RFC4035]
- "HMAC SHA TSIG Algorithm Identifiers" [RFC4036]

The upper limit on the permissible size of a DNS message over TCP is defined in RFC 1035, section 4.2.2. Unlike DNS messages over UDP, this limit is not changed by EDNS0.

Field names used in this document will correspond to the names as they appear in the IANA registry for DNS Header Flags [DNSFLGS].

2.1 AXFR query

An AXFR query is sent by a client whenever there is a reason to ask. This might be because of zone maintenance activities or as a result of a command line request, say for debugging.

2.1.1 Header Values

These are the DNS message header values for an AXFR query.

- ID  See note 2.1.1.a
- QR  MUST be 0 (Query)
- Opcode  MUST be 0 (Standard Query)
- AA  See note 2.1.1.b
- TC  See note 2.1.1.b
- RD  See note 2.1.1.b
- RA  See note 2.1.1.b
- Z  See note 2.1.1.c
- AD  See note 2.1.1.b
- CD  See note 2.1.1.b
- RCODE  MUST be 0 (No Error)
- QDCOUNT  MUST be 1
- ANCOUNT  MUST be 0
- NSCOUNT  MUST be 0
- ARCOUNT  See note 2.1.1.d

Note 2.1.1.a Set to any value that the client desires. There is no specific means for selecting the value in this field. (Recall that AXFR is done only via TCP connections.)

Note 2.1.1.b The client must set to 0, the server must ignore.

Note 2.1.1.c The value in this field has no meaning in the context of AXFR query messages. For the client, it is RECOMMENDED that the value be zero. The server MUST ignore this value.

Note 2.1.1.d The value MAY be 0, 1 or 2. If it is 1, the additional section MUST contain both an EDNS0 [RFC2671] OPT resource record and a record carrying transaction integrity and authentication data, currently a choice of TSIG [RFC2845] and SIG(0) [RFC2931]. If the value is 2, then the additional section MUST contain either only an EDNS0 OPT resource record or a record carrying transaction integrity and authentication data. If the value is 0, the additional section MUST be empty.

A note on "future proofing" this document. It is possible that in the future more records might be introduced that share the property of being placed in the additional section. Such records might be other options to, say, TSIG and SIG(0) for message authentication or may be completely unrelated to that service. In any case, each new record that might appear in the additional section might expand the range of values that this field can take on. As predicting the future is still an unproven field, further details are not available. Check back later for updates.

2.1.2 Query Section

The Query section of the AXFR query MUST conform to section 4.1.2 of RFC 1035, and contain the following values:
QNAME the name of the zone requested
QTYPE AXFR (= 252), the pseudo-RR type for zone transfer [DNSVALS]
QCLASS the class of the zone requested

2.1.3 Answer Section
MUST be empty.

2.1.4 Authority Section
MUST be empty.

2.1.5 Additional Section
The client MAY include an EDNS0 OPT resource record. If the server has indicated that it does not support EDNS0, the client MUST send this section without an EDNS0 OPT resource record if there is a retry. Indication that a server does not support EDNS0 is not an explicit element in the protocol, it is up to the client to interpret. Most likely, the server will return a FORMERR which might be related to the OPT resource record.

The client MAY include a transaction integrity and authentication resource record, currently a choice of TSIG or SIG(0). If the server has indicated that it does not recognize the resource record, and that the error is indeed caused by the resource record, the client probably ought not try again. Removing the security data in the face of an obstacle ought to only be done with full awareness of the implication of doing so.

In general, if an AXFR client is aware that an AXFR server does not support a particular mechanism, the client SHOULD NOT attempt to engage the server using the mechanism (or at all). A client MAY become aware of a server’s abilities via a configuration setting.

2.2 AXFR response
The AXFR response will consist of 0 or more messages.

A 0 message response is very exceptional. It is unhealthy for there to be 0 responses in a protocol that is designed around a query - response paradigm. A 0 message response is reserved for situations in which the server has a reason to suspect that the query is sent for the purpose of abuse. Therefore any earnest query has the expectation of some response.

An AXFR response that is transferring the zone’s contents will consist of a series of DNS messages bounded in size by the limited permissible size. In such a series, the first message MUST begin with the SOA resource record of the zone, the last message MUST conclude with the same SOA resource record. Intermediate message MUST NOT contain the SOA resource record. The first message MUST copy the Query Section from the corresponding AXFR query message in to the first response message’s query section. Subsequent messages MAY do the same.

Editorial note “MAY” or SHOULD/are RECOMMENDED TO
An AXFR response that is indicating an error MUST consist of a single DNS message with the return code set to the appropriate value for the condition encountered - once the error condition is detected. Such a message MUST copy the AXFR query Query Section into its Query Section.

An AXFR client might receive a number of AXFR response messages free of an error condition before the message indicating the error is received. But once an error is reported, the AXFR client can assume this the reporting message is the last.

An AXFR client MUST be able to react to no AXFR response messages from the server. An AXFR server MAY elect to silently discard the AXFR query but this is only RECOMMENDED if the server has reasons to deduce that the query was sent maliciously.

An AXFR server MAY elect to close the underlying TCP connection in response to an AXFR query. Because this action could impact other DNS queries and responses, it is RECOMMENDED that this tactic only be employed when there are strong indications of malicious activity. Still, an AXFR client MUST be able to adequately react to this situation.

2.2.1 Header Values
ID See note 2.2.1.a
QR MUST be 1 (Response)
opcode MUST be 0 (Standard Query)
AA See note 2.2.1.b
TC MUST be 0 (Not truncated)
RD RECOMMENDED copy request’s value, MAY be set to 0
RA See note 2.2.1.c
Z See note 2.2.1.d
AD See note 2.2.1.e
CD See note 2.2.1.f
Z CODE See note 2.2.1.g
QDCOUNT MUST be 1 in the first message; MUST be 0 or 1 in all following
ANCOUNT See note 2.2.1.h
NSCOUNT MUST be 0
ARCOUNT See note 2.2.1.i

Note 2.2.1.a Because of old implementations, the requirement on this section is stated in detail. New DNS servers MUST set this field to the value of the AXFR query ID in each AXFR response message for the session. New AXFR clients MUST be able to accept sessions in which the responses do not have the same ID field.

If a client detects or is aware that the server is new, that is, all of the responses have the same ID value as the query, the client MAY issue other DNS queries (of any type) to the server using the same transport. Unless the client is sure that the server will consistently set the ID field to the server’s ID, the client is NOT RECOMMENDED to issue any other queries until the end of the zone transfer. A client MAY become aware of a server’s abilities via a configuration setting.
Note 2.2.1.b If the RCODE is 0 (no error), then the AA bit MUST be 1. For any other value of RCODE, the AA bit MUST be set according to rules for that error code. If in doubt, it is RECOMMENDED that is be set to 1. It is RECOMMENDED that the value be ignored by the AXFR client.

Note 2.2.1.c It is RECOMMENDED that the server set the value to 0, the client MUST ignore this value.

The server MAY set this value according to the local policy regarding recursive service, but doing so might confuse the interpretation of the response as AXFR cannot be retrieved recursively. A client MAY note the server’s policy regarding recursive from this value, but SHOULD NOT conclude that the AXFR response was obtained recursively even if the RD bit was 1 in the query.

Note 2.2.1.d The server MUST set to 0, and the client MUST ignore.

Note 2.2.1.e If the implementation supports the DNS Security Extensions (see below) then this value MUST be set according to the rules in RFC 4035, section 3.1.6, “The AD and CD Bits in an Authoritative Response”. If the implementation does not support the DNS Security Extensions, then this value MUST be set to 0 and MUST be ignored upon receipt.

The DNS Security Extensions (DNSSEC) is defined in these base documents:
- "DNS Security Introduction and Requirements" [RFC4033]
- "Resource Records for the DNS Security Extensions" [RFC4034]
- "Protocol Modifications for the DNS Security Extensions" [RFC4035]

Note 2.2.1.f In the absence of an error, the server MUST set the value of this field to NoError. If a server is not authoritative for the queried zone, the server SHOULD set the value to NotAuth. (Reminder, consult the appropriate IANA registry [DNSVALS]:) If a client receives any other value in response, it MUST act according to the error. For example, a malformed AXFR query or the presence of an EDNS0 OPT resource record sent to an old server will garner a FormErr value. This value is not set as part of the AXFR response processing. The same is true for other error-indicating values.

Note 2.2.1.g The count of answer records MUST equal the number of resource records in the AXFR Answer Section. When a server is aware that a client will only accept one resource record per response message, then the value MUST be 1. A server MAY be made aware of a client’s limitations via configuration data.

Note 2.2.1.h The value MAY be 0, 1 or 2. If it is 2, the additional section MUST contain both an EDNS0 [RFC2671] OPT resource record and a record carrying transaction integrity and authentication data, currently a choice of TSIG [RFC2845] and SIG(0) [RFC2893]. If the value is 1, then the additional section MUST contain either only an EDNS0 OPT resource record or a record carrying transaction integrity and authentication data. If the value is 0, the additional section MUST be empty.

A note on “future proofing” this document. It is possible that in the future more records might be introduced that share the property of being placed in the additional section. Such records might be other options to, say, TSIG and SIG(0), for message authentication or may be completely unrelated to that service. In any case, each new record that might appear in the additional section might expand the range of values of this field that can take on. As predicting the future is still an unproven field, further details are not available. Check back later for updates.

2.2.2 Query Section

In the first response message, this section MUST be copied from the query. In subsequent messages, this section MAY be copied from the query, MAY be empty. The content of this section MAY be used to determine the context of the message, that is, the name of the zone being transferred.

2.2.3 Authority Section

MUST be populated with the zone contents. See later section on encoding zone contents.

2.2.4 Authority Section

MUST be empty.

2.2.5 Additional Section

The contents of this section MUST follow the guidelines for EDNS0, TSIG, SIG(0), or what ever other future record is possible here. See the appropriate specifications for instructions and restrictions.

3 Zone Contents

The objective of the AXFR session is to request and transfer the contents of a zone. The objective is to permit the client to reconstruct the zone as it exists at the server for the given zone serial number. Over time the definition of a zone has evolved from a static set of records to a dynamically updated set of records to a continually regenerated set of records.

3.1 Records to Include

In the answer section of AXFR response messages the resource records within a zone for the given serial number MUST appear. The definition of what belongs in a zone is described in RFC 1034, Section 4.2.1, "How the database is divided into zones", and in particular, section 4.2.1, "Technical considerations".

The first resource record of the final AXFR response message sent by the AXFR server MUST be the zone’s SOA resource record. The last resource record of the final AXFR response message sent by the AXFR server MUST be the zone’s SOA resource record. The order and grouping of all other records in the AXFR is arbitrary, but the AXFR server SHOULD group resource record sets together.

Unless the AXFR server knows that the AXFR client expects just one resource record per AXFR response message, an AXFR server SHOULD
Delegation Records, provides, "Name space specifications and BCP14, this text appears (keep in mind that the Name Compression], cf.].) "Message compression". The issue highlighted here relates to a Glue Records] "The RRs should preserve its case." ("Should" in the quote predates [terminology" which says "When you receive a domain name or label, you it relates. A nudge for updated text on this.) (Editorial note, the 4th reason was suggested, but I don't see how exist. add a record that already exists or to delete a record that does not another IXFR protocol error state where an attempt is made to incrementally another IXFR transfer from the other server, the client can encounter inconsistent zone contents for a given zone serial number, if a client 4) Beginning with an error state of two servers for a zone having authoritative set, concealing the error.) (A server authoritative for both halves would otherwise always answer from the more authoritative set, concealing the error.) 3) The inconsistent NS resource record set might indicate a problem for both halves would otherwise always answer from the more authoritative set, concealing the error.) The question that arises is, when facing a situation in which a cut point's NS resource records do not match the authoritative set, whether an AXFR server responds with the NS resource record set that is in the zone or is at the authoritative location. The AXFR response MUST contain the cut point NS resource record set registered with the zone whether it agrees with the authoritative set or not. "Registered with" can be widely interpreted to include data residing in the zone file of the zone for the particular serial number (in zone file environments) or as any data configured to be in the zone (database), statically or dynamically. The reasons for this requirement are: 1) The AXFR server might not be able to determine that there is an inconsistency given local data, hence requiring consistency would mean a lot more needed work and even network retrieval of data. An authoritative server ought not be required to perform any queries. 2) By transferring the inconsistent NS resource records from a server that is authoritative for both the cut point and the apex to a client that is not authoritative for both, the error is exposed. For example, an authorized administrator can manually request the AXFR and inspect the results to see the inconsistent records. (A server authoritative for both halves would otherwise always answer from the more authoritative set, concealing the error.) 3) The inconsistent NS resource record set might indicate a problem in a registration database. 4) Beginning with an error state of two servers for a zone having inconsistent zone contents for a given zone serial number, if a client requests and receives an IXFR transfer from one server followed by another IXFR transfer from the other server, the client can encounter an IXFR protocol error state where an attempt is made to incrementally add a record that already exists or to delete a record that does not exist. (Editorial note, the 4th reason was suggested, but I don’t see how it relates. A nudge for updated text on this.) 3.3 Glue Records As quoted in the previous section, RFC 1034, section 4.2.1, provides guidance and rationale for the inclusion of glue records as part of an AXFR transfer. And, as also argued in the previous section of this document, even when there is an inconsistency between the address in a glue record and the authoritative copy of the name server’s address, the glue resource record that is registered as part of the zone for that serial number is to be included. This applies for glue records for any address family. The AXFR response MUST contain the appropriate glue records as registered with the zone. The interpretation of "registered with" in the previous section applies here. Inconsistent glue records are an operational matter. 3.4 Name Compression Compression of names in DNS messages is described in RFC 1035, section 4.1.4, "Message compression". The issue highlighted here relates to a comment made in RFC 1034, section 3.1, "Name space specifications and terminology" which says "When you receive a domain name or label, you should preserve its case." ("Should" in the quote predates [BCP14].) Name compression in an AXFR message MUST preserve the case of the original domain name. That is, although when comparing a domain name, "A" equals "A", when comparing for the purposes of message compression, "a" is not equal to "A". Note that this is not the usual definition of name comparison in the DNS protocol and represents a new
requirement on AXFR servers.

Rules governing name compression of RDATA in an AXFR message MUST abide by the specification in "Handling of Unknown DNS Resource Record (RR) Types" [RFC3597], specifically, section 4 on "Domain Name Compression."

3.5 Occluded Names

Dynamic Update [RFC2136] (and including DNAME [2672]) operations can have a side effect of occluding names in a zone. The addition of a delegation point via dynamic update will render all subordinate domain names to be in a limbo, still part of the zone but not available for use in the lookup process. The addition of a DNAME resource record set has the same impact. The subordinate names are said to be "occluded."

Occluded names MUST be included in AXFR responses. An AXFR client MUST be able to identify and handle occluded names. The rationale for this action is based on a speedy recovery if the dynamic update operation was in error and is to be undone.

4 Transport

AXFR sessions are currently restricted to TCP by section 4.3.5 of RFC 1034 that states: "Because accuracy is essential, TCP or some other reliable protocol must be used for AXFR requests." The most common scenario is for an AXFR client to open a TCP connection to the AXFR server, send an AXFR query, receive the AXFR response, and then close the connection. There are variations on this, such as a query for the zone's SOA resource record first, and so on.

Two issues have emerged since the original specification of AXFR. One is that lack of specificity has yielded some implementations that assume the TCP connection is dedicated to the single AXFR session, which has led to implementation choices that prevent either multiple concurrent zone transfers or the use of the open connection for other queries. The other issue is the prospect of using UDP as a transport has come to look promising because of trends in the past two decades.

Being able to have multiple concurrent zone transfers is considered desirable by operators who have sets of name servers that are authoritative for a common set of zones. It would be desirable if the name server implementations did not have to wait for one zone to transfer before the next could begin. The desire here is to tighten the specification, not a change, but adding words to the unclear areas, to define what is needed to permit two servers to share a TCP connection among concurrent AXFR sessions. The challenge is to design this in a way that can fallback to the old behavior if either the AXFR client or AXFR server is incapable of performing multiple concurrent AXFR sessions.

With the addition of EDNS0 and applications which require many small zones such as in web hosting and some ENUM scenarios, AXFR sessions on UDP are now possible and desirable. However, there are still some aspects of the AXFR session that are not easily translated to UDP. This document leaves AXFR over UDP undefined, with the issue to be discussed and possibly appear in a separate definition.

4.1 TCP

In the original definition there is an implicit assumption (probably unintentional) that a TCP connection is used for one and only one AXFR session. This is evidenced in no requirement to copy either the Query Section nor the message ID in responses, no explicit ordering information within the AXFR response messages and the lack of an explicit notice indicating that a zone transfer continues in the next message.

The guidance given here is intended to enable better performance of the AXFR exchange as well as guidelines on interactions with older software. Better performance includes being able to multiplex DNS message exchanges including zone transfer sessions. Guidelines for interacting with older software are generally applicable to AXFR clients as reversing the situation, older AXFR client and newer AXFR server ought to induce the server to operate within the specification for an older server.

4.1.1 AXFR client TCP

An AXFR client MAY request an connection to an AXFR server for any reason. An AXFR client SHOULD close the connection when there is no apparent need to use the connection for some time period. The AXFR server ought not to maintain idle connections, the burden of connection closure ought to be on the client. Apparent need for the connection is a judgement for the AXFR client and the DNS client. If the connection is used for multiple sessions, or if it is known sessions will be coming or is there is other query/response traffic on the open connection, that is "apparent need."

An AXFR client MAY cancel delivery of a zone only by closing the connection. However, this action will also cancel all other outstanding activity using the connection. There is no other mechanism by which an AXFR response can be cancelled.

When a TCP connection is closed remotely (relative to the client), whether by the AXFR server or due to a network event, the AXFR client MUST cancel all outstanding sessions. Recovery from this situation is not straightforward. If the disruption was a spurious event, attempting to restart the connection would be proper. If the disruption was caused by a medium or long term disruption, the AXFR client would be wise to not spend too many resources trying to rebuild the connection. Finally, if the connection was dropped because of a policy at the AXFR server (as can be the case with older AXFR servers), the AXFR client would be wise to not retry the connection.

Unfortunately, knowing which of the three cases above applies is not clear (momentary disruption, failure, policy).

An AXFR client MAY use an already opened TCP connection to start an AXFR session. Using an existing open connection is RECOMMENDED over opening a new connection. (Non AXFR session traffic can also use an open connection.) If in doing so the AXFR client realizes that
the responses cannot be properly differentiated (lack of matching query IDs for example) or the connection is terminated for a remote reason, then the AXFR client SHOULD not attempt to reuse an open connection with the specific AXFR server until the AXFR server is updated (which is of course, not an event captured in the DNS protocol).

4.1.2 AXFR server TCP

An AXFR server MUST be able to handle multiple AXFR sessions on a single TCP connection, as well as handle other query/response sessions.

If a TCP connection is closed remotely, the AXFR server MUST cancel all AXFR sessions in place. No retry activity is necessary, that is initiated by the AXFR client.

Local policy MAY dictate that a TCP connection is to be closed. Such an action SHOULD be in reaction to limits such as those placed on the number of outstanding open connections. Closing a connection in response to a suspected security event SHOULD be done only in extreme cases, when the server is certain the action is warranted.

An isolated request for a zone not on the AXFR server SHOULD receive a response with the appropriate return code and not see the connection broken.

4.2 UDP

AXFR sessions over UDP transport are not defined.

5 Authorization

A zone administrator has the option to restrict AXFR access to a zone. This was not envisioned in the original design of the DNS but has emerged as a requirement as the DNS has evolved. Restrictions on AXFR could be for various reasons including a desire for in some instances having a legal requirement to keep the bulk version of the zone concealed or to prevent the servers from handling the load incurred in serving AXFR. All reasons are arguable, but the fact remains that there is a requirement to provide mechanisms to restrict AXFR.

A DNS implementation SHOULD provide means to restrict AXFR sessions to specific clients. By default, a DNS implementation SHOULD only allow the designated authoritative servers to have access to the zone.

An implementation SHOULD allow access to be granted to Internet Protocol addresses and ranges, regardless of whether a source address could be spoofed. Combining this with techniques such as Virtual Private Networks (VPN) [RFC2764] or Virtual LANs has proven to be effective.

A general purpose implementation is RECOMMENDED to implement access control based upon "Secret Key Transaction Authentication for DNS" [RFC2845] and/or "DNS Request and Transaction Signatures ( SIG(0)s )" [RFC2931].

A general purpose implementation SHOULD allow access to be open to all AXFR requests. I.e., an operator ought to be able to allow any AXFR query to be granted.

A general purpose implementation SHOULD NOT have a default policy for AXFR requests to be "open to all."

6 Zone Integrity

Ensuring that an AXFR client does not accept a forged copy of a zone is important to the security of a zone. If a zone operator has the opportunity, protection can be afforded via dedicated links, physical or virtual via a VPN among the authoritative servers. But there are instances in which zone operators have no choice but to run AXFR sessions over the global public Internet.

Besides best attempts at securing TCP sessions, DNS implementations SHOULD provide means to make use of "Secret Key Transaction Authentication for DNS" [RFC2845] and/or "DNS Request and Transaction Signatures ( SIG(0)s )" [RFC2931] to allow AXFR clients to verify the contents. These techniques MAY also be used for authorization.

7 Backwards Compatibility

Describing backwards compatibility is difficult because of the lack of specifics in the original definition. In this section some hints at building in backwards compatibility are given, mostly repeated from the earlier sections.

Backwards compatibility is not necessary, but the greater extent of an implementation’s compatibility increases it’s interoperability. For turnkey implementations this is not usually a concern. For general purpose implementations this takes on varying levels of importance depending on the implementer’s desire to maintain interoperability.

It is unfortunate that a need to fall back to older behavior cannot be discovered, hence needs to be noted in a configuration file. An implementation SHOULD, in it’s documentation, encourage operators to periodically review AXFR clients and servers it has made notes about as old software periodically gets updated.

7.1 Server

An AXFR server has the luxury of being able to react to an AXFR client’s abilities with the exception of knowing if the client can accept multiple resource records per AXFR response message. The knowledge that a client is so restricted apparently cannot be discovered, hence it has to be set by configuration.

An implementation of an AXFR server SHOULD permit configuring, on a per AXFR client basis, a need to revert to single resource record per message. The default SHOULD be to use multiple records per message.

7.2 Client

An AXFR client has the opportunity to try extensions when querying an AXFR server.
12.1 Normative


13 Editor’s Address

Edward Lewis
46000 Center Oak Plaza
Sterling, VA, 22032, US
+1-703-436-5468
ed.lewis@freespire.biz