Network Working Group
Request for Comments: 1996
Updates: 1035
Category: Standards Track

P. Vixie
ISC

August 1996

A Mechanism for Prompt Notification of Zone Changes (DNS NOTIFY)

Status of this Memo

Abstract

This memo describes the NOTIFY opcode for DNS, by which a master server advises a set of slave servers that the master's data has been changed and that a query should be initiated to discover the new data.

1. Rationale and Scope

1.1. Slow propagation of new and changed data in a DNS zone can be due to a zone's relatively long refresh times. Longer refresh times are beneficial in that they reduce load on the master servers, but that benefit comes at the cost of long intervals of incoherence among authority servers whenever the zone is updated.

1.2. The DNS NOTIFY transaction allows master servers to inform slave servers when the zone has changed -- an interrupt as opposed to poll model -- which it is hoped will reduce propagation delay while not unduly increasing the masters' load. This specification only allows slaves to be notified of SOA RR changes, but the architechture of NOTIFY is intended to be extensible to other RR types.

1.3. This document intentionally gives more definition to the roles of "Master," "Slave" and "Stealth" servers, their enumeration in NS RRs, and the SOA MNAME field. In that sense, this document can be considered an addendum to [RFC1035].

2. Definitions and Invariants

2.1. The following definitions are used in this document:

Slave	an authoritative server which uses zone transfer to retrieve the zone. All slave servers are named in the NS RRs for the zone.
Master	any authoritative server configured to be the source of zone transfer for one or more slave servers.
Primary Master	master server at the root of the zone transfer dependency graph. The primary master is named in the zone's SOA MNAME field and optionally by an NS RR. There is by definition only one primary master server per zone.
Stealth	like a slave server except not listed in an NS RR for the zone. A stealth server, unless explicitly configured to do otherwise, will set the AA bit in responses and be capable of acting as a master. A stealth server will only be known by other servers if they are given static configuration data indicating its existence.
Notify Set	set of servers to be notified of changes to some zone. Default is all servers named in the NS RRset, except for any server also named in the SOA MNAME. Some implementations will permit the name server administrator to override this set or add elements to it (such as, for example, stealth servers).

2.2. The zone's servers must be organized into a dependency graph such that there is a primary master, and all other servers must use AXFR or IXFR either from the primary master or from some slave which is also a master. No loops are permitted in the AXFR dependency graph.

3. NOTIFY Message

3.1. When a master has updated one or more RRs in which slave servers may be interested, the master may send the changed RR's name, class, type, and optionally, new RDATA(s), to each known slave server using a best efforts protocol based on the NOTIFY opcode.

3.2. NOTIFY uses the DNS Message Format, although it uses only a subset of the available fields. Fields not otherwise described herein are to be filled with binary zero (0), and implementations

must ignore all messages for which this is not the case.

3.3. NOTIFY is similar to QUERY in that it has a request message with the header QR flag "clear" and a response message with QR "set". The response message contains no useful information, but its reception by the master is an indication that the slave has received the NOTIFY and that the master can remove the slave from any retry queue for this NOTIFY event.

3.4. The transport protocol used for a NOTIFY transaction will be UDP unless the master has reason to believe that TCP is necessary; for example, if a firewall has been installed between master and slave, and only TCP has been allowed; or, if the changed RR is too large to fit in a UDP/DNS datagram.

3.5. If TCP is used, both master and slave must continue to offer name service during the transaction, even when the TCP transaction is not making progress. The NOTIFY request is sent once, and a "timeout" is said to have occurred if no NOTIFY response is received within a reasonable interval.

3.6. If UDP is used, a master periodically sends a NOTIFY request to a slave until either too many copies have been sent (a "timeout"), an ICMP message indicating that the port is unreachable, or until a NOTIFY response is received from the slave with a matching query ID, QNAME, IP source address, and UDP source port number.

Note:
The interval between transmissions, and the total number of retransmissions, should be operational parameters specifiable by the name server administrator, perhaps on a per-zone basis. Reasonable defaults are a 60 second interval (or timeout if using TCP), and a maximum of 5 retransmissions (for UDP). It is considered reasonable to use additive or exponential backoff for the retry interval.

3.7. A NOTIFY request has QDCOUNT>0, ANCOUNT>=0, AUCOUNT>=0, ADCOUNT>=0. If ANCOUNT>0, then the answer section represents an unsecure hint at the new RRset for this <QNAME,QCLASS,QTYPE>. A slave receiving such a hint is free to treat equivilence of this answer section with its local data as a "no further work needs to be done" indication. If ANCOUNT=0, or ANCOUNT>0 and the answer section differs from the slave's local data, then the slave should query its known masters to retrieve the new data.

3.8. In no case shall the answer section of a NOTIFY request be used to update a slave's local data, or to indicate that a zone transfer needs to be undertaken, or to change the slave's zone refresh timers.

Only a "data present; data same" condition can lead a slave to act differently if ANCOUNT>0 than it would if ANCOUNT=0.

3.9. This version of the NOTIFY specification makes no use of the authority or additional data sections, and so conforming implementations should set AUCOUNT=0 and ADCOUNT=0 when transmitting requests. Since a future revision of this specification may define a backwards compatible use for either or both of these sections, current implementations must ignore these sections, but not the entire message, if AUCOUNT>0 and/or ADCOUNT>0.

3.10. If a slave receives a NOTIFY request from a host that is not a known master for the zone containing the QNAME, it should ignore the request and produce an error message in its operations log.

Note:
This implies that slaves of a multihomed master must either know their master by the "closest" of the master's interface addresses, or must know all of the master's interface addresses. Otherwise, a valid NOTIFY request might come from an address that is not on the slave's state list of masters for the zone, which would be an error.

3.11. The only defined NOTIFY event at this time is that the SOA RR has changed. Upon completion of a NOTIFY transaction for QTYPE=SOA, the slave should behave as though the zone given in the QNAME had reached its REFRESH interval (see [RFC1035]), i.e., it should query its masters for the SOA of the zone given in the NOTIFY QNAME, and check the answer to see if the SOA SERIAL has been incremented since the last time the zone was fetched. If so, a zone transfer (either AXFR or IXFR) should be initiated.

Note:
Because a deep server dependency graph may have multiple paths from the primary master to any given slave, it is possible that a slave will receive a NOTIFY from one of its known masters even though the rest of its known masters have not yet updated their copies of the zone. Therefore, when issuing a QUERY for the zone's SOA, the query should be directed at the known master who was the source of the NOTIFY event, and not at any of the other known masters. This represents a departure from [RFC1035], which specifies that upon expiry of the SOA REFRESH interval, all known masters should be queried in turn.

3.12. If a NOTIFY request is received by a slave who does not implement the NOTIFY opcode, it will respond with a NOTIMP (unimplemented feature error) message. A master server who receives such a NOTIMP should consider the NOTIFY transaction complete for

that slave.

4. Details and Examples

4.1. Retaining query state information across host reboots is optional, but it is reasonable to simply execute an SOA NOTIFY transaction on each authority zone when a server first starts.

4.2. Each slave is likely to receive several copies of the same NOTIFY request: One from the primary master, and one from each other slave as that slave transfers the new zone and notifies its potential peers. The NOTIFY protocol supports this multiplicity by requiring that NOTIFY be sent by a slave/master only AFTER it has updated the SOA RR or has determined that no update is necessary, which in practice means after a successful zone transfer. Thus, barring delivery reordering, the last NOTIFY any slave receives will be the one indicating the latest change. Since a slave always requests SOAs and AXFR/IXFRs only from its known masters, it will have an opportunity to retry its QUERY for the SOA after each of its masters have completed each zone update.

4.3. If a master server seeks to avoid causing a large number of simultaneous outbound zone transfers, it may delay for an arbitrary length of time before sending a NOTIFY message to any given slave. It is expected that the time will be chosen at random, so that each slave will begin its transfer at a unique time. The delay shall not in any case be longer than the SOA REFRESH time.

Note:
This delay should be a parameter that each primary master name server can specify, perhaps on a per-zone basis. Random delays of between 30 and 60 seconds would seem adequate if the servers share a LAN and the zones are of moderate size.

4.4. A slave which receives a valid NOTIFY should defer action on any subsequent NOTIFY with the same <QNAME,QCLASS,QTYPE> until it has completed the transaction begun by the first NOTIFY. This duplicate rejection is necessary to avoid having multiple notifications lead to pummeling the master server.

4.5 Zone has Updated on Primary Master

Primary master sends a NOTIFY request to all servers named in Notify Set. The NOTIFY request has the following characteristics:

```
query ID:   (new)
op:         NOTIFY (4)
resp:       NOERROR
flags:      AA
qcount:     1
qname:      (zone name)
qclass:     (zone class)
qtype:      T_SOA
```

4.6 Zone has Updated on a Slave that is also a Master

As above in 4.5, except that this server's Notify Set may be different from the Primary Master's due to optional static specification of local stealth servers.

4.7 Slave Receives a NOTIFY Request from a Master

When a slave server receives a NOTIFY request from one of its locally designated masters for the zone enclosing the given QNAME, with QTYPE=SOA and QR=0, it should enter the state it would if the zone's refresh timer had expired. It will also send a NOTIFY response back to the NOTIFY request's source, with the following characteristics:

```
query ID:   (same)
op:         NOTIFY (4)
resp:       NOERROR
flags:      QR AA
qcount:     1
qname:      (zone name)
qclass:     (zone class)
qtype:      T_SOA
```

This is intended to be identical to the NOTIFY request, except that the QR bit is also set. The query ID of the response must be the same as was received in the request.

4.8 Master Receives a NOTIFY Response from Slave

When a master server receives a NOTIFY response, it deletes this query from the retry queue, thus completing the "notification process" of "this" RRset change to "that" server.

5. Security Considerations

We believe that the NOTIFY operation's only security considerations are:

1. That a NOTIFY request with a forged IP/UDP source address can cause a slave to send spurious SOA queries to its masters, leading to a benign denial of service attack if the forged requests are sent very often.

2. That TCP spoofing could be used against a slave server given NOTIFY as a means of synchronizing an SOA query and UDP/DNS spoofing as a means of forcing a zone transfer.

6. References

[RFC1035]
Mockapetris, P., "Domain Names - Implementation and Specification", STD 13, RFC 1035, November 1987.

[IXFR]
Ohta, M., "Incremental Zone Transfer", RFC 1995, August 1996.

7. Author's Address

Paul Vixie
Internet Software Consortium
Star Route Box 159A
Woodside, CA 94062

Phone: +1 415 747 0204
EMail: paul@vix.com