Network Working Group
Request for Comments: 2136
Updates: 1035
Category: Standards Track

P. Vixie, Editor
ISC
S. Thomson
Bellcore
Y. Rekhter
Cisco
J. Bound
DEC

April 1997

Dynamic Updates in the Domain Name System (DNS UPDATE)

Status of this Memo

Abstract

The Domain Name System was originally designed to support queries of a statically configured database. While the data was expected to change, the frequency of those changes was expected to be fairly low, and all updates were made as external edits to a zone's Master File.

Using this specification of the UPDATE opcode, it is possible to add or delete RRs or RRsets from a specified zone. Prerequisites are specified separately from update operations, and can specify a dependency upon either the previous existence or nonexistence of an RRset, or the existence of a single RR.

UPDATE is atomic, i.e., all prerequisites must be satisfied or else no update operations will take place. There are no data dependent error conditions defined after the prerequisites have been met.

1 - Definitions

This document intentionally gives more definition to the roles of "Master," "Slave," and "Primary Master" servers, and their enumeration in NS RRs, and the SOA MNAME field. In that sense, the following server type definitions can be considered an addendum to [RFC1035], and are intended to be consistent with [RFC1996]:

Slave	an authoritative server that uses AXFR or IXFR to retrieve the zone and is named in the zone's NS RRset.

Master	an authoritative server configured to be the source of AXFR or IXFR data for one or more slave servers.
Primary Master	master server at the root of the AXFR/IXFR 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.

A domain name identifies a node within the domain name space tree structure. Each node has a set (possibly empty) of Resource Records (RRs). All RRs having the same NAME, CLASS and TYPE are called a Resource Record Set (RRset).

The pseudocode used in this document is for example purposes only. If it is found to disagree with the text, the text shall be considered authoritative. If the text is found to be ambiguous, the pseudocode can be used to help resolve the ambiguity.

1.1 - Comparison Rules

1.1.1. Two RRs are considered equal if their NAME, CLASS, TYPE, RDLENGTH and RDATA fields are equal. Note that the time-to-live (TTL) field is explicitly excluded from the comparison.

1.1.2. The rules for comparison of character strings in names are specified in [RFC1035 2.3.3].

1.1.3. Wildcarding is disabled. That is, a wildcard ("*") in an update only matches a wildcard ("*") in the zone, and vice versa.

1.1.4. Aliasing is disabled: A CNAME in the zone matches a CNAME in the update, and will not otherwise be followed. All UPDATE operations are done on the basis of canonical names.

1.1.5. The following RR types cannot be appended to an RRset. If the following comparison rules are met, then an attempt to add the new RR will result in the replacement of the previous RR:

SOA	compare only NAME, CLASS and TYPE -- it is not possible to have more than one SOA per zone, even if any of the data fields differ.
WKS	compare only NAME, CLASS, TYPE, ADDRESS, and PROTOCOL -- only one WKS RR is possible for this tuple, even if the services masks differ.

CNAME compare only NAME, CLASS, and TYPE -- it is not possible to have more than one CNAME RR, even if their data fields differ.

1.2 - Glue RRs

For the purpose of determining whether a domain name used in the UPDATE protocol is contained within a specified zone, a domain name is "in" a zone if it is owned by that zone's domain name. See section 7.18 for details.

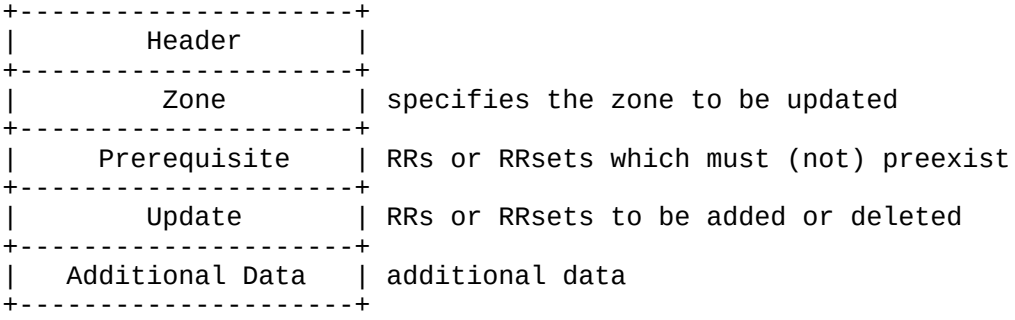
1.3 - New Assigned Numbers

```
CLASS = NONE (254)
RCODE = YXDOMAIN (6)
RCODE = YXRRSET (7)
RCODE = NXRRSET (8)
RCODE = NOTAUTH (9)
RCODE = NOTZONE (10)
Opcode = UPDATE (5)
```

2 - Update Message Format

The DNS Message Format is defined by [RFC1035 4.1]. Some extensions are necessary (for example, more error codes are possible under UPDATE than under QUERY) and some fields must be overloaded (see description of CLASS fields below).

The overall format of an UPDATE message is, following [ibid]:



The Header Section specifies that this message is an UPDATE, and describes the size of the other sections. The Zone Section names the zone that is to be updated by this message. The Prerequisite Section specifies the starting invariants (in terms of zone content) required for this update. The Update Section contains the edits to be made, and the Additional Data Section contains data which may be necessary to complete, but is not part of, this update.

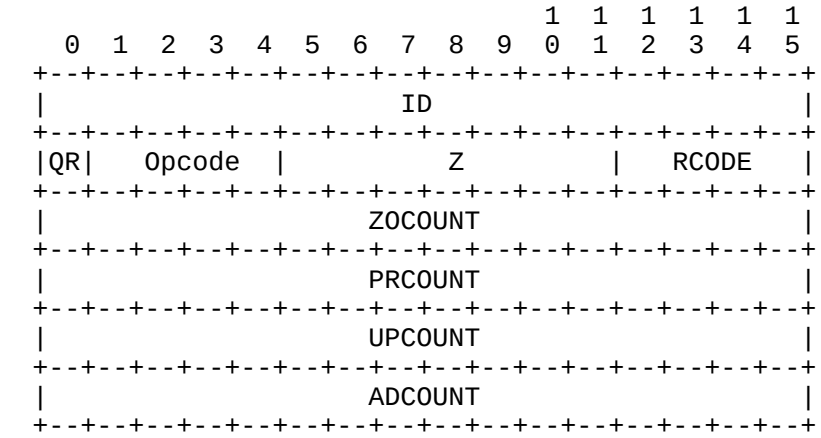
2.1 - Transport Issues

An update transaction may be carried in a UDP datagram, if the request fits, or in a TCP connection (at the discretion of the requestor). When TCP is used, the message is in the format described in [RFC1035 4.2.2].

2.2 - Message Header

The header of the DNS Message Format is defined by [RFC 1035 4.1]. Not all opcodes define the same set of flag bits, though as a practical matter most of the bits defined for QUERY (in [ibid]) are identically defined by the other opcodes. UPDATE uses only one flag bit (QR).

The DNS Message Format specifies record counts for its four sections (Question, Answer, Authority, and Additional). UPDATE uses the same fields, and the same section formats, but the naming and use of these sections differs as shown in the following modified header, after [RFC1035 4.1.1]:



These fields are used as follows:

ID A 16-bit identifier assigned by the entity that generates any kind of request. This identifier is copied in the corresponding reply and can be used by the requestor to match replies to outstanding requests, or by the server to detect duplicated requests from some requestor.

QR A one bit field that specifies whether this message is a request (0), or a response (1).

Opcode A four bit field that specifies the kind of request in this message. This value is set by the originator of a request and copied into the response. The Opcode value that identifies an UPDATE message is five (5).

Z Reserved for future use. Should be zero (0) in all requests and responses. A non-zero Z field should be ignored by implementations of this specification.

RCODE Response code - this four bit field is undefined in requests and set in responses. The values and meanings of this field within responses are as follows:

Mneumonic	Value	Description
NOERROR	0	No error condition.
FORMERR	1	The name server was unable to interpret the request due to a format error.
SERVFAIL	2	The name server encountered an internal failure while processing this request, for example an operating system error or a forwarding timeout.
NXDOMAIN	3	Some name that ought to exist, does not exist.
NOTIMP	4	The name server does not support the specified Opcode.
REFUSED	5	The name server refuses to perform the specified operation for policy or security reasons.
YXDOMAIN	6	Some name that ought not to exist, does exist.
YXRRSET	7	Some RRset that ought not to exist, does exist.
NXRRSET	8	Some RRset that ought to exist, does not exist.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.