Stream: Internet Engineering Task Force (IETF)
RFC: 9664
Category: Standards Track
Published: June 2025
ISSN: 2070-1721
Authors: S. Cheshire *Apple Inc.* T. Lemon *Apple Inc.*

RFC 9664 An EDNS(0) Option to Negotiate Leases on DNS Updates

Abstract

This document describes an EDNS(0) option that can be used between DNS Update Requesters and authoritative DNS servers to include a lifetime (lease duration) in a DNS Update or DNS Update Response, allowing a server to garbage collect stale Resource Records that have been added by DNS Updates if they are not renewed.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Dynamic Update in the Domain Name System (DNS Update) [RFC2136] allows for a mapping from a persistent hostname to an IP address that changes over time. This capability is particularly beneficial to mobile hosts, whose IP addresses may frequently change with location.

However, the mobile nature of such hosts often means that Resource Records (RRs) added using DNS Update are not properly deleted. For instance, consider a mobile user who publishes address RRs via DNS Update. If this user moves their laptop out of range of the Wi-Fi access point, the address RR containing stale information may remain on the authoritative DNS server indefinitely. Thus, an extension to DNS Update is required to tell the server to automatically delete RRs after a period of time if they are not refreshed.

2. Conventions and Terminology Used in This Document

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.1. Terminology

DNS-SD: DNS-based Service Discovery [RFC6763]

EDNS(0): Extension Mechanisms for DNS [RFC6891]

Update Lease option: Update Lease EDNS(0) option

Lease: An agreement by an authoritative DNS server to continue to publish a record from the time of registration until the lease duration has elapsed and then stop publishing it

Lease Duration: The time between the start and end of a lease

Lease Update Request: DNS Update Request containing an Update Lease option

Lease Update Response: DNS Update Response containing an Update Lease option

RR: Resource Record

Registration Request: A Lease Update Request that is constructed with the purpose of adding new information that is not thought to already be present on the authoritative DNS server

Registration: The result of a successful Registration Request

Refresh Request: A Lease Update Request that extends the lease duration on an existing Registration

Refresh: The result of a successful Refresh Request

3. Mechanisms

The Update Lease option is included in a standard DNS Update Request [RFC2136] within an EDNS(0) OPT pseudo-RR [RFC6891].

4. Lease Update Request and Response Format

Lease Update Requests and Responses are formatted as standard DNS Update messages [RFC2136]. Such messages **MUST** include the EDNS(0) OPT RR [RFC6891]. This OPT RR **MUST** include an Update Lease EDNS(0) Option as shown below:

Field Name	Field Type	Description
OPTION-CODE	u_int16_t	UPDATE-LEASE (2)
OPTION-LENGTH	u_int16_t	4 (LEASE) or 8 (LEASE + KEY-LEASE)
LEASE	u_int32_t	desired lease duration (Lease Update Request) or granted lease duration (Lease Update response), in seconds
KEY-LEASE	u_int32_t	optional desired (or granted) lease duration for KEY RRs, in seconds

Table 1

Lease Update Requests contain, in the LEASE field of the OPT RDATA, an unsigned 32-bit integer indicating the lease duration in seconds, desired by the Requester, represented in network (big-endian) byte order. In Lease Update Responses, this field contains the actual lease duration granted by the authoritative DNS server. The lease durations granted by the server may be less than, greater than, or equal to the value requested by the Requester.

There are two variants of the Update Lease option: The 4-byte variant and the 8-byte variant.

In the 4-byte variant, the LEASE indicated in the Update Lease option applies to all RRs in the Update section.

In the 8-byte variant, the Update Lease communicates two lease durations. The LEASE indicated in the Update Lease option applies to all RRs in the Update section *except* for KEY RRs. The KEY-LEASE indicated in the Update Lease option applies to KEY RRs in the Update section.

More information about how the two variants are used is given in Section 4.3.

KEY RRs are given a special lease duration because these RRs are used in the DNS-SD Service Registration Protocol [RFC9665] to reserve a name (or names) when the service is not present.

In the case of a KEY RR and some other RR, obviously the KEY lease duration applies to the KEY RR, and the lease duration applies to the other RR. If more than one RR that is not a KEY RR is added by the Lease Update Request, the lease duration (not the KEY lease duration) is applied to all such RRs. RRs that are removed are permanently removed.

4.1. Types of Lease Update Requests

This document describes two types of Lease Update Requests: Registrations and Refreshes. A Registration Request is a Lease Update Request that is intended to add information not already present on the authoritative DNS server. A Refresh Request is intended simply to renew the lease on a previous Registration without changing anything. Registrations and Refreshes are both Lease Update Requests, so the term "Lease Update Request" is used to specify behavior that is the same for both types of DNS Updates.

In some cases, it may be necessary to add new information without removing old information. For the purpose of this document, such Lease Update Requests are Registrations, although in effect, they may also refresh whatever information is unchanged from a previous registration.

4.2. Requester Behavior

DNS Update Requesters **MUST** send an Update Lease option with any DNS Update that updates RRs that are not intended to be present indefinitely. The Update Lease option **SHOULD** specify a lease duration that is no shorter than 1800 seconds (30 minutes). Requesters **MAY** specify a shorter lease duration if they anticipate that the RRs being updated will change more frequently than every 30 minutes. Requesters that expect the updated RRs to be relatively static **SHOULD** request appropriately longer lease durations.

If the DNS Response received by the Requester does not include an Update Lease option, this is an indication that the authoritative DNS server does not support the Update Lease option. In this case, the Requester **SHOULD** continue sending Refresh Requests (see below) as if the server had returned an identical Update Lease option in its Response.

If the DNS Response does include an Update Lease option, the Requester **MUST** use the durations returned in this option when determining when to send Refresh Requests. This is true both if the durations returned by the server are shorter and if they are longer.

When sending a Registration Request, the Requester **MUST** delay the initial transmission by a random amount of time across the range of 0-3000 milliseconds, with a granularity of no more than 10 milliseconds. This prevents synchronization of multiple devices of the same type at a site upon recovery from a power failure. This requirement applies only to the initial Registration Request on startup; since Refresh Requests include a random factor as well, any synchronization that occurs after such an event should quickly randomize.

> The 10 ms granularity is a scheduling requirement intended to result in an even spread of Requests so that every Request doesn't come an exact number of seconds after startup. This requirement should not be construed as requiring anything of the link layer on which the packet is transmitted: the link layer may well impose its own constraints on the timing at which a message is sent, and this document does not claim to override such constraints.

> The use of a 3000 ms (3-second) random delay as opposed to some other random delay is to allow for enough time to meaningfully spread the load when many devices renew at once, without delaying so long that the delay in discovery of devices becomes obvious to an end user. A 3-second random delay means that if there are, for example, 100 devices, and the random number generator spread is even, we would have one renewal every 30 ms. In practice, on relatively constrained devices acting as Service Registration Protocol (SRP) servers, we are seeing the processing time for an SRP registration taking on the order of 7 ms, so this seems reasonable.

4.3. Server Behavior

Authoritative DNS servers implementing the Update Lease option **MUST** include an Update Lease option in response to any successful DNS Update (RCODE=0) that includes an Update Lease option. Servers **MAY** return lease durations different from those specified by the Requester, granting longer leases to reduce network traffic due to Refreshes or shorter leases to reduce the lifetime of stale data.

Although both the 4-byte and 8-byte variants are valid on both requesters and servers, older (pre-standard) requesters and servers may exist that support only the 4-byte variant. Therefore, requesters and servers that (as required by this specification) support both variants must account for the possibility that the peer with which they are communicating may be an older implementation that supports only the 4-byte variant.

A server that receives an 8-byte variant from a requester **MUST** respond with an 8-byte variant giving the granted lease times.

A server that receives a 4-byte variant from a requester **MUST** treat the 4-byte variant as specifying both the lease duration and the KEY lease duration and **MUST** respond with a 4-byte variant. In this case, the key and the other RRs expire at the same time.

A requester that receives a 4-byte variant from a server when it sent an 8-byte variant in its request **MUST** treat the 4-byte variant as specifying both the lease duration and the KEY lease duration.

5. Refresh Requests

A Refresh Request is a DNS Update Request that is sent to the server after an initial DNS Update has been sent in order to prevent the update's RRs from being garbage collected.

5.1. Refresh Request Format

Refresh Requests and their corresponding responses are formatted like Update Lease Requests and Update Lease Responses (see Section 4). The Refresh Request is constructed with the assumption that the previous Registration or Refresh is still in effect. In the case that the RRs added in a previous update were for some reason garbage collected (e.g., because of a server reboot that resulted in loss of state), the Refresh Request will result in those RRs being added again.

The Refresh Request **SHOULD NOT** include any DNS Update prerequisites that will fail if the Requester's previous Registration or Refresh is still in effect. It also **SHOULD NOT** include prerequisites that would fail if the RRs affected by the previous Registration or Refresh are no longer present; that is, the Refresh Request should also work as a Registration Request. There may be cases where this is not possible; in which case, the response from the server can be used to determine how to proceed when the Refresh Request fails.

A Lease Update Request that changes the authoritative DNS server state resulting from a previous Refresh or Registration is a Registration Request, not a Refresh Request.

In a Refresh Request, the Update Lease option contains the desired new lease duration, and in the corresponding response, the Update Lease option contains the actual granted lease. The lease duration provided in LEASE in the Update Lease option applies to all RRs in the Update section of the Refresh Request, except that when the 8-byte Update Lease variant is sent, the duration specified in KEY-LEASE applies to any KEY RRs included in the Update section.

5.2. Requester Behavior

A Requester that intends for its RRs from a previous Registration or Refresh to remain active **MUST** send a Refresh Request before the lease expires; otherwise, the RRs will be removed by the server.

In order to prevent Registrations expiring, Requesters **MUST** refresh them. When a Lease Update Request succeeds, the requester computes a time limit that is 80% of the lease duration plus a random offset between 0% and 5% of the lease duration. The random offset is to prevent refreshes from being synchronized. When this time limit has expired, the requester **MUST** send a Refresh Request if the data in the initial Registration should continue to be advertised.

For Refresh Requests, the server is expected to return an Update Lease option, if supported, just as with the initial Registration Request. As with the Registration Request, the Requester **MUST** use the durations returned by the server in the Lease Update Response when determining when to send the next Refresh Request.

When sending Refresh Requests, the Requester **MUST** include an Update Lease option, as it did in the initial Registration Request. The Update Lease option **MAY** either specify the same durations as in the initial Registration Request or use the values returned by the server in the previous Lease Update Response or other desired values as appropriate. As with responses to Registration Requests, the Requester **MUST** use the lease durations returned by the server in the response when determining when to send the next Refresh Request.

If the Requester sends a Refresh Request message and does not receive a response from the authoritative DNS server, then the Requester should implement a reasonable retry strategy to attempt to refresh the record registrations before they expire. Given that 15% - 20% of the lease lifetime still remains, these retransmissions do not need to be overly aggressive. For example, the Requester could retry nine more times, spaced uniformly at equal intervals from the time of the first failed Refresh attempt until the expiration time of the records. After the expiration time of the records, the Refresh Request effectively turns into a new Registration Request, and further retransmissions after this proceed as described in Section 6.

5.2.1. Coalescing Refresh Requests

If the Requester has performed multiple Registrations with a single server for different RRs, the Requester **MAY** send a Refresh Request containing RRs from all such Registrations to that server in a single Refresh Request. This effectively places all RRs for a Requester on the same expiration schedule, reducing network traffic due to Refreshes.

In doing so, the Requester includes in the Refresh Request all existing RRs previously successfully registered on the server, including those not yet close to expiration, so long as at least one RR updated in the Refresh Request has elapsed at least 75% of its original lease duration. If the Requester uses UDP, the Requester **MUST NOT** coalesce Refresh Requests if doing so would cause truncation of the Request; in this case, the Requester either sends multiple Requests or uses TCP to send the complete Refresh Request at once.

Requesters **SHOULD NOT** send a Refresh Request when all of the RRs in the Refresh Request would have more than 50% of their lease duration remaining before expiry. However, there may be cases where the Requester needs to send an early Refresh Request, and it **MAY** do so. For example, a power-constrained (sleepy) device may need to send a Refresh Request when the radio is powered so as to avoid having to power it up later.

Another case where this may be needed is when the lease duration registered with the server is no longer appropriate and the Requester wishes to negotiate a different lease duration. However, in this case, if the server does not honor the requested lease duration in its response, the Requester **MUST NOT** retry this negotiation.

5.3. Server Behavior

Upon receiving a valid Refresh Request, the server **MUST** send an acknowledgment. This acknowledgment is a Lease Update Response as described in Section 4 and contains the new lease duration of the Registration being Refreshed. The server **MUST NOT** increment the serial number of a zone as the result of a Refresh Request if the operation does not result in any change to the zone contents.

However, the server's state may not match what the requester expects. In this case, a Refresh Request may actually appear to be a Registration Request, from the server's perspective. If the Refresh Request changes the contents of the zone, the server **MUST** update the zone serial number.

6. Retransmission Strategy

The DNS protocol, including DNS updates, can operate over UDP or TCP. For communication using UDP, reliable transmission must be guaranteed by retransmitting when a DNS UDP message is not acknowledged in a reasonable amount of time. Section 4.2.1 of the DNS specification [RFC1035] provides some guidance on this topic, as does Section 1 of the IETF's guide to common DNS implementation errors [RFC1536]. Section 3.1.3 of the UDP Usage Guidelines [RFC8085] also provides useful guidance that is particularly relevant to DNS.

7. Garbage Collection

If the lease duration of an RR elapses without being refreshed, the authoritative DNS server **MUST NOT** return that RR in answers to queries. The server **MAY** delete that RR from its database. The lease durations returned by the server to the Requester are used in determining when the lease on an RR has expired.

For all RRs other than a KEY RR included in a Lease Update Request, the lease duration is the LEASE value in the Update Lease option. For KEY RRs, if the optional KEY-LEASE value was included, this duration is used rather than the duration specified in the LEASE. If the KEY-LEASE was not specified, the duration specified in the LEASE is used for all RRs in the Lease Update Request.

8. Security Considerations

Section 8 of the DNS Update specification [RFC2136] describes problems that can occur around DNS updates. Servers implementing this specification should follow these recommendations.

Several additional issues can arise when relying on the Update Lease option.

First, a too-long lease duration is not much different from no lease duration: the RRs associated with such a Registration will effectively never be cleaned up. Servers implementing Update Lease should have a default upper bound on the maximum acceptable value both for the LEASE and KEY-LEASE values sent by the requester. Default values for these limits of 24 hours and 7 days, respectively, are **RECOMMENDED**. Servers **MAY** provide a way for the operator to change this upper limit.

The second issue is that a too-short lease can result in increased server load as Requesters rapidly renew such Registrations. A delay in renewing could result in the registered RRs being removed prematurely. Servers implementing Update Lease **MUST** have a default minimum lease duration that avoids this issue. A minimum of 30 seconds for both the LEASE and KEY-LEASE durations is **RECOMMENDED**. However, in most cases, much longer lease durations (for example, an hour) **SHOULD** be used. Servers **MAY** provide a way for the operator to change this lower limit.

There may be some cost associated with renewing leases. A malicious (or buggy) requester could renew at a high rate in order to overload the server more than it would be overloaded by query traffic. This risk is present for an authoritative server handling normal (no-lease) DNS Updates as well. Servers should follow established industry best practices to guard against flooding attacks, both for malicious flooding of DNS messages over UDP and for similar flooding attacks using TCP [SYN] [RFC4953].

Some authentication strategy should be used when accepting DNS updates. Shared secret [RFC8945] or public key signing (e.g., SIG(0) [RFC2931]) should be required. Keys should have limited authority: compromise of a key should not result in compromise of the entire contents of one or more zones managed by the server. Key management strategy is out of scope for this document. Service Registration Protocol [RFC9665] uses DNS Update Leases with "First Come, First Served Naming" rather than an explicit trust establishment process to confer update permission to a set of RRs.

9. IANA Considerations

IANA has updated the "DNS EDNS0 Option Codes (OPT)" registry [EDNS0Reg] as regards value 2 as follows:

Value:	2
Name:	Update Lease
Status:	Standard
Reference:	RFC 9664

10. References

10.1. Normative References

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2136] Vixie, P., Ed., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, DOI 10.17487/RFC2136, April 1997, <https://www.rfc-editor.org/info/rfc2136>.

[RFC6891] Damas, J., Graff, M., and P. Vixie, "Extension Mechanisms for DNS (EDNS(0))", STD 75, RFC 6891, DOI 10.17487/RFC6891, April 2013, <https://www.rfc-editor.org/info/rfc6891>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

10.2. Informative References

[RFC1536] Kumar, A., Postel, J., Neuman, C., Danzig, P., and S. Miller, "Common DNS Implementation Errors and Suggested Fixes", RFC 1536, DOI 10.17487/RFC1536, October 1993, <https://www.rfc-editor.org/info/rfc1536>.

[RFC2931] Eastlake 3rd, D., "DNS Request and Transaction Signatures (SIG(0)s)", RFC 2931, DOI 10.17487/RFC2931, September 2000, <https://www.rfc-editor.org/info/rfc2931>.

[RFC4953] Touch, J., "Defending TCP Against Spoofing Attacks", RFC 4953, DOI 10.17487/RFC4953, July 2007, <https://www.rfc-editor.org/info/rfc4953>.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, DOI 10.17487/RFC6763, February 2013, <https://www.rfc-editor.org/info/rfc6763>.

[RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", BCP 145, RFC 8085, DOI 10.17487/RFC8085, March 2017, <https://www.rfc-editor.org/info/rfc8085>.

[RFC8945] Dupont, F., Morris, S., Vixie, P., Eastlake 3rd, D., Gudmundsson, O., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", STD 93, RFC 8945, DOI 10.17487/RFC8945, November 2020, <https://www.rfc-editor.org/info/rfc8945>.

[RFC9665] Lemon, T. and S. Cheshire, "Service Registration Protocol for DNS-Based Service Discovery", RFC 9665, DOI 10.17487/RFC9665, June 2025, <https://www.rfc-editor.org/info/rfc9665>.

[EDNS0Reg] IANA, "DNS EDNS0 Option Codes (OPT)", <https://www.iana.org/assignments/dns-parameters>.

[SYN] Eddy, W., "Defenses Against TCP SYN Flooding Attacks", The Internet Protocol Journal, Cisco Systems, Volume 9, Number 4, December 2006, <https://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_9-4/ipj_9-4.pdf>.

Acknowledgments

Thanks to Marc Krochmal and Kiren Sekar for their work in 2006 on the precursor to this document. Thanks also to Roger Pantos and Chris Sharp for their contributions. Thanks to Chris Box, Esko Dijk, Jonathan Hui, Peter van Dijk, Abtin Keshvarzian, Nathan Dyck, Steve Hanna, Gabriel Montenegro, Kangping Dong, and Tim Wicinski for their working group reviews of this document. Thanks to David Dong, Olafur Gudmundsson, Brian Trammel, and Shivan Sahib for their directorate reviews and IANA reviews.

Authors' Addresses

Stuart Cheshire
Apple Inc.
One Apple Park Way
Cupertino, CA 95014
United States of America
Phone: +1 408 974 3207
Email: cheshire@apple.com

Ted Lemon
Apple Inc.
P.O. Box 958
Brattleboro, VT 05302
United States of America
Email: mellon@fugue.com