

DNSOP Working Group
Internet-Draft
Intended status: Experimental
Expires: September 25, 2019

T. Pusateri
Unaffiliated
March 24, 2019

Private DNS Subdomains
draft-pusateri-dnsop-private-subdomains-01

Abstract

This document describes a method of providing private DNS subdomains such that each subdomain can be shared among multiple devices of a single owner or group. A private subdomain can be used for sharing personal services while increasing privacy and limiting knowledge of scarce resources.

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 <https://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 September 25, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Requirements Language	3
3. Subdomain Operations	3
3.1. Zone Creation	3
3.2. Adding / Removing Resource Records	4
3.3. Zone Destruction	5
4. Querying Resource Records	5
4.1. Signed Requests	5
5. Responses	5
6. Security Considerations	5
7. References	7
7.1. Normative References	7
7.2. Informative References	8
Author's Address	8

1. Introduction

Section 6.6 of [RFC7558] highlights the privacy risks of DNS service announcements in clear text. While there has been a long focus on access control to private services including the use of encryption and authentication through TLS [RFC8446] for connections, the DNS-SD announcements [RFC6763] themselves may leak private information including but not limited to device types and versions, enabled services on the device subject to attack, personal names and identifiers used for tracking, etc.

Some services are meant to be advertised and used freely by devices on the link but other services are restricted to an owner or group and these services are announced publicly as a side effect of the current service discovery deployment.

This document defines a method for collaborating devices to share private services with one another but without revealing the existence of these services in a public way. This provides an additional layer of privacy protection for an individual's devices or those of a defined group sharing a common purpose.

The additional privacy is achieved by creating private subdomains that require a private key for bidirectional access to DNS queries and responses for the zone.

This document defines a subdomain hierarchy for providers to enable this feature as well as a mechanism for interoperable transfers to and from the subdomain. This includes creating and destroying the subdomains, adding and removing records through DNS Update, and private authenticated queries.

2. Requirements Language

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. These words may also appear in this document in lower case as plain English words, absent their normative meanings.

3. Subdomain Operations

An administrative domain provider will enable private subdomains by creating a base subdomain at "_pvt.<domain>." In addition to the SOA and NS records, a public KEY record [RFC2535], [RFC3445] MUST be created at the apex which is openly available for queries. This public key will be used for message encryption relating to the maintenance of private subdomains.

The Zone bit for all KEY records used in this specification MUST be set to 0. The protocol value for the KEY record is set to DNSSEC (3) and the algorithm value is taken from the available algorithms defined in the IANA DNS Security Algorithm Numbers.

3.1. Zone Creation

A subdomain is created by the owner in an administrative domain for which the owner has a trusted relationship. For instance, if a user has services provided by an administrative domain and that user has been assigned a user name or account at that domain, the user could then uniquely claim ownership of the subdomain "<user>._pvt.<domain>."

The administrative domain can use any means it finds sufficient to verify the trusted relationship with the user including RADIUS, AAA, email verification, or some other method not described here which may include enabling the service on a per-user basis.

The user can then create the subdomain by sending an UPDATE [RFC2136] to the MNAME of the SOA record for "_pvt.<domain>." containing a new KEY record for the zone "<user>._pvt.<domain>." The KEY record contains a public key that will later be used by the administrative domain to verify signed requests. If this zone already exists, the UPDATE fails with RCODE YXRRSet. If the zone does not exist, the user has successfully claimed the zone and all subsequent operations by the user will require knowledge of the private key associated with the registered public key in the KEY record. The user can distribute

this private key to any or all of its devices for access to the private subdomain.

In response, appropriate NS records for "<user>" will be created in the "_pvt.<domain>." and a new SOA record "<user>._pvt.<domain>." with appropriate record fields will also be created along side the new KEY record submitted by the user. The authoritative servers for the new "<user>._pvt.<domain>." will be managed by the administrative domain provider.

At this point, the owner has knowledge of the public key of "_pvt.<domain>." and the administrative domain has knowledge of the public key of "<user>._pvt.<domain>." All subsequent operations for the subdomain "<user>._pvt.<domain>." are signed with the private key of the sender. The administrative domain provider can verify the sender should have access to the records in the private zone and the owner can verify the received records are authentic and haven't been tampered with.

3.2. Adding / Removing Resource Records

Once the zone is created, the user can begin adding or removing records to the "<user>._pvt.<domain>" subdomain through additional UPDATE messages for the zone.

To ensure zone additions and deletions are from the subdomain owner, the UPDATE message must be signed with the owner's private key and the signature included in the additional data section in the form of a SIG(0) record [RFC2931]. More information about authenticating UPDATE messages can be found in [RFC3007].

If the administrative domain provider can verify the signature with the subdomain owner's public key, the records are added to, or removed from, the "<user>._pvt.<domain>." zone. See section 2.5.2 of [RFC2136] for the specifics of adding or removing records in the Update section.

In order to change the KEY record at the apex of the zone, the old KEY record should be added to the Prerequisite section and the new KEY record in the Update section. Then the message is signed with the subdomain owner's private key.

This is useful for changing the algorithm or key length for the public/private key pair.

If instead the private key associated with the public key in the KEY record has been compromised, the user should contact the

administrative domain out of band to verify authenticity and replace the KEY record through a process established by the provider.

3.3. Zone Destruction

When the owner is ready to destroy the subdomain "`<user>._pvt.<domain>.`", it can do so by deleting the KEY record at the apex through an UPDATE message. As above, the UPDATE message additional data section MUST contain a SIG(0) signature over the entire message. Once validated, the zone will be removed along with the NS records for "`<user>.`" in the "`_pvt.<domain>.`" zone.

The owner is free to destroy and re-create the subdomain as needed for as long as the relationship continues with the administrative domain.

4. Querying Resource Records

Only the owner and the administrative domain provider know the true contents of the records. Queries must be made through direct connections to an authoritative server for the subdomain over a TLS connection. If the authoritative server also provides connections over UDP or TCP without TLS, queries for records in private zones over non-TLS connections MUST return an RCODE containing REFUSED.

4.1. Signed Requests

All queries MUST be signed with the private key of the owner. The signature MUST be in the Additional records section in the form of a SIG(0) record. If a query is received that is not signed or cannot be verified with the public key in the KEY record at the apex of the "`<user>._pvt.<domain>.`", a response containing the question as received with an RCODE of REFUSED with no answers MUST be returned. The Authority section of the response MUST contain the SOA record for the "`<user>._pvt.<domain>.`".

Authoritative servers MUST verify the signature in the query before returning the results.

5. Responses

6. Security Considerations

The Strict Privacy Usage Profile for DNS over TLS is REQUIRED for connections to the authoritative name servers [RFC8310]. Since this is a new protocol, transition mechanisms from the Opportunistic Privacy profile are unnecessary.

See Section 9 of [RFC8310] for additional recommendations for various versions of TLS usage.

DNSSEC is RECOMMENDED for the authentication of authoritative DNS servers. TLS alone does not provide complete security. TLS certificate verification can provide reasonable assurance that the client is really communicating with the server associated with the desired host name, but since the desired host name is learned via a DNS query, if the DNS response is subverted then the client may have a secure connection to a rogue server. DNSSEC can provide added confidence that the response has not been subverted.

Deployment recommendations on the appropriate key lengths and cypher suites are beyond the scope of this document. Please refer to TLS Recommendations [RFC7525] for the best current practices. Consider that best practices only exist for a snapshot in time and recommendations will continue to change. Updated versions or errata may exist for these recommendations.

The authoritative server connection endpoint SHOULD be authenticated using DANE TLSA records for the associated DNS record used to determine the name (SOA, SRV, etc). This associates the target's name with a trusted TLS certificate [RFC7673]. This procedure uses the TLS Server Name Indication (SNI) extension [RFC6066] to inform the server of the name the client has authenticated through the use of TLSA records. Therefore, if the DNS record used to obtain the name passes DNSSEC validation and a TLSA record matching the target name is useable, an SNI extension must be used for the target name to ensure the client is connecting to the server it has authenticated. If the target name does not have a usable TLSA record, then the use of the SNI extension is optional. See Usage Profiles for DNS over TLS and DNS over DTLS [RFC8310] for more information on authenticating domain names.

7. References

7.1. Normative References

- [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>>.
- [RFC2535] Eastlake 3rd, D., "Domain Name System Security Extensions", RFC 2535, DOI 10.17487/RFC2535, March 1999, <<https://www.rfc-editor.org/info/rfc2535>>.
- [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>>.
- [RFC3007] Wellington, B., "Secure Domain Name System (DNS) Dynamic Update", RFC 3007, DOI 10.17487/RFC3007, November 2000, <<https://www.rfc-editor.org/info/rfc3007>>.
- [RFC3445] Massey, D. and S. Rose, "Limiting the Scope of the KEY Resource Record (RR)", RFC 3445, DOI 10.17487/RFC3445, December 2002, <<https://www.rfc-editor.org/info/rfc3445>>.
- [RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <<https://www.rfc-editor.org/info/rfc6066>>.
- [RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <<https://www.rfc-editor.org/info/rfc7525>>.
- [RFC7673] Finch, T., Miller, M., and P. Saint-Andre, "Using DNS-Based Authentication of Named Entities (DANE) TLSA Records with SRV Records", RFC 7673, DOI 10.17487/RFC7673, October 2015, <<https://www.rfc-editor.org/info/rfc7673>>.

- [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>>.
- [RFC8310] Dickinson, S., Gillmor, D., and T. Reddy, "Usage Profiles for DNS over TLS and DNS over DTLS", RFC 8310, DOI 10.17487/RFC8310, March 2018, <<https://www.rfc-editor.org/info/rfc8310>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

7.2. Informative References

- [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>>.
- [RFC7558] Lynn, K., Cheshire, S., Blanchet, M., and D. Migault, "Requirements for Scalable DNS-Based Service Discovery (DNS-SD) / Multicast DNS (mDNS) Extensions", RFC 7558, DOI 10.17487/RFC7558, July 2015, <<https://www.rfc-editor.org/info/rfc7558>>.

Author's Address

Tom Pusateri
Unaffiliated
Raleigh NC 27608
USA

Phone: +1 (919) 867-1330
Email: pusateri@bangj.com

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: September 5, 2019

T. Pusateri
T. Wattenberg
Unaffiliated
March 4, 2019

DNS TIMEOUT Resource Record
draft-pusateri-dnsop-update-timeout-02

Abstract

This specification defines a new DNS TIMEOUT resource record (RR) that associates a lifetime with one or more zone resource records with the same owner name, type, and class. It is intended to be used to transfer resource record lifetime state between a zone's primary and secondary servers and to store lifetime state during server software restarts.

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 <https://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 September 5, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Requirements Language	3
3. Sources of TIMEOUT Expiry Time	3
4. Resource Record Composition	3
4.1. Represented Record Type	4
4.2. Represented Record Count	4
4.3. Method Identifiers	4
4.3.1. Method Identifier 0: NO METHOD	5
4.3.2. Method Identifier 1: RDATA	5
4.4. Expiry Time	5
5. TIMEOUT RDATA Wire Format	5
6. Primary Server Behavior	7
7. Secondary Server Behavior	7
8. TIMEOUT RDATA Presentation Format	7
9. IANA Considerations	9
10. Security Considerations	9
11. Acknowledgments	9
12. References	10
12.1. Normative References	10
12.2. Informative References	10
Appendix A. Example TIMEOUT resource records	11
Authors' Addresses	13

1. Introduction

DNS Update [RFC2136] provides a mechanism to dynamically add/remove DNS resource records to/from a zone. When a resource record is dynamically added, it remains in the zone until it is removed manually or via a subsequent DNS Update. A zone administrator may want to enforce a default lifetime for dynamic updates (such as the DHCP lease lifetime) or the DNS Update may contain a lifetime using an EDNS(0) Update Lease option [I-D.sekar-dns-ul]. However, this lease lifetime is not communicated to secondary servers and will not endure through server software restarts. Therefore, this specification defines a new DNS TIMEOUT resource record that associates a lifetime with one or more resource records with the same owner name, type, and class that can be transferred to secondary servers through normal AXFR [RFC5936], IXFR [RFC1995] transfer mechanisms.

An UPDATE lifetime could be stored in a proprietary database on an authoritative primary server but there is an advantage to saving it as a resource record: redundant master servers and secondary servers

capable of taking over as the primary server for a zone automatically can benefit from the existing database synchronization of resource records. In addition, primary and secondary servers from multiple vendors can synchronize the lifetimes through the open format provided by a resource record.

2. Requirements Language

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. These words may also appear in this document in lower case as plain English words, absent their normative meanings.

3. Sources of TIMEOUT Expiry Time

The expire time may come from many different sources. A few are listed here however, this list is not considered complete.

1. Via DHCP Dynamic Lease Lifetime communicated out of band.
2. Via EDNS(0) Update Lease option [I-D.sekar-dns-ul] communicated in DNS Update.
3. Via an administrative default server value such as one day (86400 seconds).

4. Resource Record Composition

TIMEOUT resource records provide expiry times for a mixed variety of resource record types with the same owner name, type, and class. Since there could exist multiple records of the same record type with the same owner name and class, the TIMEOUT resource record must be able to identify each of these records individually with only different RDATA. As an example, PTR records for service discovery [RFC6763] provide a level of indirection to SRV and TXT records by instance name. The instance name is stored in the PTR RDATA and multiple PTR records with the same owner name but only differing RDATA often exist.

In order to distinguish each individual record with potentially different expiry times, the TIMEOUT resource record contains an expiry time, the record type, a method to identify the actual records for which the expiry time applies, and a count of the number of records represented. Multiple TIMEOUT records with the same owner name and class are created for each expiry time, record type, and

resource record representation. If the expiry time is the same, multiple records can be combined into a single TIMEOUT record with the same owner name, class, and record type but this is NOT REQUIRED.

The fields and their values in a TIMEOUT record are defined as:

4.1. Represented Record Type

A 16-bit field containing the resource record type to which the TIMEOUT record applies. Multiple TIMEOUT records for the same owner name, class, and represented type can exist.

4.2. Represented Record Count

The Represented Record Count is a 8-bit value that specifies the number of records of the specified record type with this expiry time.

An RR Count of zero indicates that it is not necessary to represent any records in the list. This is a shortcut notation meaning all resource records with the same owner name, class, and record type use the same Expiry Time. There MUST be only one TIMEOUT record for the same owner name, class, and record type if the Represented Record Count is zero. If an additional TIMEOUT record exists with the same owner name, class, and record type, it MUST be ignored and SHOULD be removed. When the Represented Record Count is 0, the Method Identifier is set to NO METHOD (0) on transmission and ignored on reception.

In the unlikely event that the Represented Record Count exceeds 255 which is the largest number representable in 8 bits, multiple instances of the same Expiry Time can exist.

4.3. Method Identifiers

The Method Identifier is a 8-bit value that specifies an identifier for the algorithm used to distinguish between resource records. The identifiers are declared in a registry maintained by IANA for the purpose of listing acceptable methods for this purpose. In addition to the method and the index, the registry MAY contain a fixed output length in bits of the method to be used or the term "variable" to denote a variable length output per record. It is conceivable, though not likely, that the same method could be used with different fixed output lengths. In this case, each fixed output length would require a different identifier in the registry. Additions to this registry will be approved with additional documentation under expert review. At the time that the registry is created by IANA, a group of expert reviewers will be established.

Additional methods of representing records such as hashes or other algorithms may be defined in the future. If such methods are defined, a primary server could create TIMEOUT record using a new method that is not understood by a secondary server that could take over as the primary in the event of an outage or administrative change. In this case, the new primary would not be able to identify the records it is supposed to TIMEOUT. This is a misconfiguration and it is the responsibility of the administrator to ensure that secondary servers in a position to become primary understand the TIMEOUT record methods of the primary server.

4.3.1. Method Identifier 0: NO METHOD

The method identifier of 0 is defined as "NO METHOD" and MUST NOT be used if the represented record count is greater than 0. The value of 0 is to be included in the IANA registry of method identifier values.

4.3.2. Method Identifier 1: RDATA

The method identifier of 1 is defined as "RDATA". It begins with the RDATA length as a 16-bit value containing the length of the RDATA in bytes followed by the number of bytes of RDATA as appears in the record being represented. The record MUST be in canonical DNSSEC form as described in Section 6 of [RFC4034].

4.4. Expiry Time

The expiry time is a 64-bit number expressed as the number of seconds since the UNIX epoch (00:00:00 UTC on January 1, 1970). This value is an absolute time at which the record will expire. Lease times must be converted to an absolute expiry time when received.

5. TIMEOUT RDATA Wire Format

The TIMEOUT resource record follows the same pattern as other DNS resource records as defined in Section 3.2.1 of [RFC1035] including owner name, type, class, TTL, RDATA length, and RDATA.

The RDATA section of the resource record with method identifier RDATA (1) is illustrated in Figure 1:

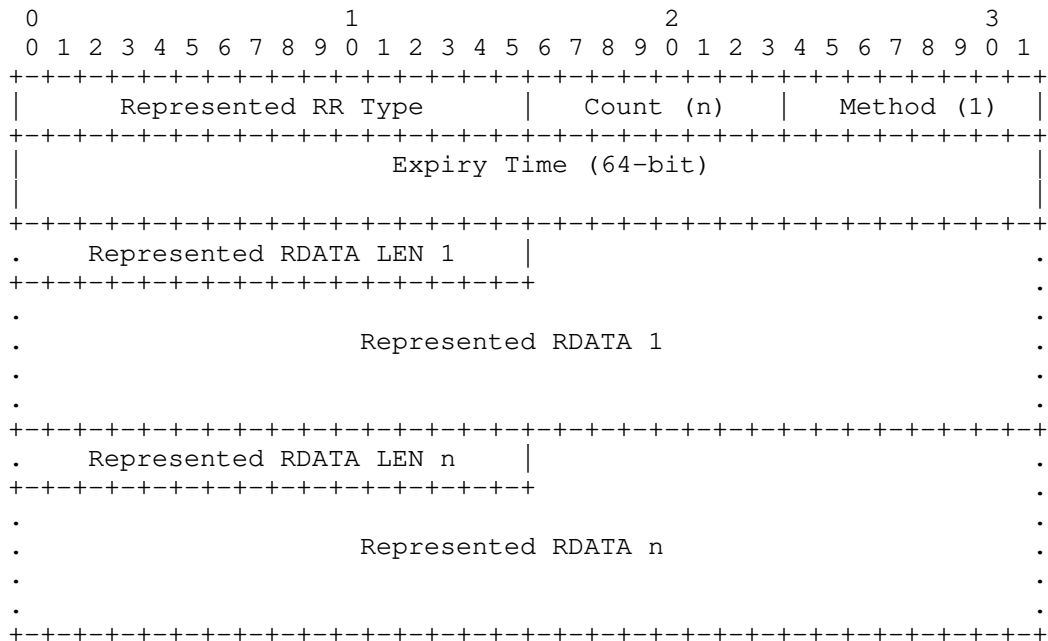


Figure 1: Method (1) RDATA Wire Format

Figure 1 represents an arbitrary number of represented records with the same owner name, class, and represented type. For each expiry time, a list of RDATA length and RDATA pairs are attached. The overall RDATA length of the TIMEOUT record indicates when the last represented record is contained in the record.

The RDATA section of the resource record with method identifier NO METHOD (0) is illustrated in Figure 2:

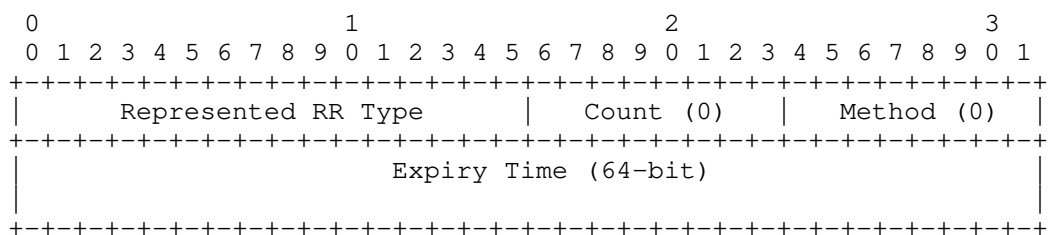


Figure 2: Method (0) RDATA Wire Format

Figure 2 represents the TIMEOUT RDATA field of all matching records of the represented type for the same owner name and class.

6. Primary Server Behavior

A TIMEOUT resource record MUST be removed when the last resource record it covers has been removed. This may be due to the record expiring (reaching the expiry time) or due to a subsequent DNS Update or administrative action.

Upon receiving any DNS UPDATE deleting resource records that might have been covered by a TIMEOUT RR, a primary server MUST remove all represented records in all of the TIMEOUT records with the same owner name, class, and represented type.

As a reminder from Section 3.3.13 of [RFC1035], the MINIMUM field of the SOA for the zone is used as a lower bound of the TTL for all records in the zone. Therefore, even if the TIMEOUT record will expire in less time than the MINIMUM, the TTL is still set to the MINIMUM for records covered by the TIMEOUT record and the TIMEOUT record itself when a response is returned by an authoritative server. The TIMEOUT RR is mostly for the benefit of the authoritative server to know when to remove the records. The fact that some records might live longer in the cache of a resolver is no different than other records that might get removed while still in a remote resolver cache.

7. Secondary Server Behavior

A secondary server may or may not understand TIMEOUT resource records. If a secondary server does not understand them, they are treated like any other resource record that the server may not understand [RFC3597].

A secondary server MUST NOT expire the records in a zone it maintains covered by the TIMEOUT resource record and it MUST NOT expire the TIMEOUT resource record itself when the last record it covers has expired. The secondary server MUST always wait for the records to be removed or updated by the primary server.

8. TIMEOUT RDATA Presentation Format

Record Type:

resource record type mnemonics. When the mnemonic is unknown, the TYPE representation described in Section 5 of [RFC3597]

Represented Record Count:

unsigned decimal integer (0-255)

Method Identifier:

unsigned decimal integer (0-255)

Expiry Time:

The Expiry Time is displayed as a compact numeric-only representation of ISO 8601. All punctuation is removed. This form is slightly different than the recommendation in [RFC3339] but is common for DNS protocols. It is defined in Section 3.2 of [RFC4034] as YYYYMMDDHHmmSS in UTC. This form will always be exactly 14 digits since no component is optional.

YYYY is the year;

MM is the month number (01-12);

DD is the day of the month (01-31);

HH is the hour, in 24 hour notation (00-23);

mm is the minute (00-59); and

SS is the second (00-59).

RDATA Length:

unsigned decimal integer

RDATA:

record type specific

9. IANA Considerations

This document defines a new DNS Resource Record Type named TIMEOUT to be exchanged between authoritative primary and secondary DNS servers. It is assigned out of the DNS Parameters Resource Record (RR) Type registry. The value for the TIMEOUT resource record type is TBA.

This document establishes a new registry of DNS TIMEOUT Resource Record Method Identifier values. The registry shall include a numeric identifier, a method name, a description of the method, and the length of the output function in bits or the keyword "variable". The identifier is to be used in the RDATA section of the TIMEOUT resource record.

ID	Method Name	Description	Length (bits)	Definition
0	NO METHOD	All records match	0	Section 4.3.1
1	RDATA	Actual RDATA of represented records	variable	Section 4.3.2

Table 1: TIMEOUT RR Method Identifier values

10. Security Considerations

There is no secure relationship between a TIMEOUT resource record and the represented resource records it applies to. TIMEOUT records should typically only apply to resource records created through the UPDATE mechanism. Protection for permanent resource records in a zone is advisable.

Authenticated UPDATE operations MUST be REQUIRED at authoritative name servers supporting TIMEOUT resource records.

11. Acknowledgments

This idea was motivated through conversations with Mark Andrews. Thanks to Mark as well as Paul Vixie, Joe Abley, Ted Lemon, Tony Finch, Robert Story, Paul Wouters, and Dick Franks for their suggestions, review, and comments.

12. References

12.1. 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>>.
- [RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <<https://www.rfc-editor.org/info/rfc3339>>.
- [RFC3597] Gustafsson, A., "Handling of Unknown DNS Resource Record (RR) Types", RFC 3597, DOI 10.17487/RFC3597, September 2003, <<https://www.rfc-editor.org/info/rfc3597>>.
- [RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <<https://www.rfc-editor.org/info/rfc4034>>.
- [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>>.

12.2. Informative References

- [I-D.sekar-dns-ul] Cheshire, S. and T. Lemon, "Dynamic DNS Update Leases", draft-sekar-dns-ul-02 (work in progress), August 2018.
- [RFC1995] Ohta, M., "Incremental Zone Transfer in DNS", RFC 1995, DOI 10.17487/RFC1995, August 1996, <<https://www.rfc-editor.org/info/rfc1995>>.
- [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>>.
- [RFC5936] Lewis, E. and A. Hoenes, Ed., "DNS Zone Transfer Protocol (AXFR)", RFC 5936, DOI 10.17487/RFC5936, June 2010, <<https://www.rfc-editor.org/info/rfc5936>>.

[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>>.

Appendix A. Example TIMEOUT resource records

The following example shows sample TIMEOUT resource records based on DNS UPDATES containing A and AAAA address records plus the corresponding PTR records.

A host sending a name registration at time T_n for "A" and "AAAA" records with lease lifetime L_n would have a series of UPDATES (one for each zone) that contain:

Name	RR Type	Value
s.example.com.	A	192.0.2.5
s.example.com.	AAAA	12001:db8::5
5.2.0.192.in-addr.arpa.	PTR	s.example.com.
5.0.0.0.0.0.0.0.0.0.0.0.b8.0d.01.20.ip6.arpa. (bytes)	PTR	s.example.com.

Table 2: Example Address Records Update

Next, consider the TIMEOUT resource records that would be generated for the records in Table 2. Notice that none of the 4 TIMEOUT records on the server would require a hash:

Owner Name	For Type	Count	Method	Expiration
s.example.com.	A	0	0	$T_n + L_n$
s.example.com.	AAAA	0	0	$T_n + L_n$
5.2.0.192.in-addr.arpa.	PTR	0	0	$T_n + L_n$
5.0.0.0.0.0.0.0.0.0.0.0.b8.0d.01.20.ip6.arpa. (bytes)	PTR	0	0	$T_n + L_n$

Table 3: Address TIMEOUT records

Next, assume there are two hosts advertising the same service type (different service types will have different owner names). We will use `_ipp._tcp.example.com` as an example.

Host A sends an UPDATE at time T_a with lease life L_a for PTR, SRV, A, AAAA, and TXT records. Host B sends an UPDATE at time T_b with lease life L_b for PTR, SRV, A, and TXT records.

Owner name	RR Type	Value
_ipp._tcp.example.com.	PTR	p1._ipp._tcp.example.com.
p1._ipp._tcp.example.com.	SRV	0 0 631 p1.example.com.
p1._ipp._tcp.example.com.	TXT	paper=A4
p1.example.com.	A	192.0.2.1
p1.example.com.	AAAA	2001:db8::1

Table 4: DNS UPDATE from Host A

Owner name	RR Type	Value
_ipp._tcp.example.com.	PTR	p2._ipp._tcp.example.com.
p2._ipp._tcp.example.com.	SRV	0 0 631 p2.example.com.
p2._ipp._tcp.example.com.	TXT	paper=B4
p2.example.com.	A	192.0.2.2

Table 5: DNS UPDATE from Host B

For these printer registrations, the TIMEOUT records on the server would look like the following:

Owner Name	For Type	C o u n t	Met hod	Expire / RDLEN RDATA
_ipp.tcp.example.com.	PTR	1	1	Ta + La 25 p1._ipp._tcp.example.com.
_ipp.tcp.example.com.	PTR	1	1	Tb + Lb 25 p2._ipp._tcp.example.com.
p1._ipp._tcp.example.com.	SRV	0	0	Ta + La
p1._ipp._tcp.example.com.	TXT	0	0	Ta + La
p2._ipp._tcp.example.com.	SRV	0	0	Tb + Lb
p2._ipp._tcp.example.com.	TXT	0	0	Tb + Lb
p1.example.com.	A	0	0	Ta + La
p1.example.com.	AAAA	0	0	Ta + La
p2.example.com.	A	0	0	Tb + Lb

Table 6: Service TIMEOUT records

Authors' Addresses

Tom Pusateri
Unaffiliated
Raleigh, NC
USA

Email: pusateri@bangj.com

Tim Wattenberg
Unaffiliated
Cologne
Germany

Email: mail@timwattenberg.de

DNSSD Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 7, 2019

T. Pusateri
Unaffiliated
March 6, 2019

DNS Update Proxy for Service Discovery
draft-pusateri-dnssd-update-proxy-01

Abstract

This document describes a method to dynamically map multicast DNS announcements into the unicast DNS namespace for use by service discovery clients. It does not define any new protocols but uses existing DNS protocols in new ways. This solves existing problems with service discovery across multiple IP subnets in a simple, yet efficient, manner.

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 <https://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 September 7, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Requirements Language	3
3. DNS Subdomain Model	3
3.1. Subdomain Naming	4
3.2. Domain Name Discovery	5
3.3. Client Service Discovery	5
4. Update Proxy Behavior	6
4.1. mDNS Service Announcements	6
4.2. Service Caching and Refresh	6
4.3. mDNS Probing	7
4.4. Link-local Addressing	8
4.5. IPv6 and IPv4 on Same Link	8
4.6. Multiple Logical IP Subnets	8
4.7. Proxy Redundancy	9
4.8. Service Filtering and Translation	9
5. DNS UPDATE	9
5.1. Selection of Authoritative Unicast DNS Server	10
5.2. DNS UPDATE Sections	10
5.2.1. Zone Section	10
5.2.2. Prerequisite Section	11
5.2.3. Update Section	11
5.2.4. Additional Data Section	11
6. DNS Authoritative Server Behavior	12
6.1. DNS Push Notifications	12
6.2. DNSSEC Compatibility	12
6.3. DNS Update Record Lifetimes	12
7. Transitioning to Unicast	13
8. Security Considerations	14
9. References	16
9.1. Normative References	16
9.2. Informative References	17
Appendix A. Comparison to Discovery Proxy	18
Author's Address	18

1. Introduction

Multicast DNS is used today for link-local service discovery. While this has worked reasonably well on the local link, current deployment reveals two problems. First, mDNS wasn't designed to traverse across multi-subnet campus networks. Second, IP multicast doesn't work across all link types and can be problematic on 802.11 Wifi networks. Therefore, a solution is desired to contain legacy multicast DNS service discovery and transition to a unicast DNS service discovery

model. By mapping the current mDNS discovered services into regular authoritative unicast DNS servers, clients from any IP subnet can make unicast queries through normal unicast DNS resolvers.

There are many ways to map services discovered using multicast DNS into the unicast namespace. This document describes a way to do the mapping using a proxy that sends DNS UPDATE messages [RFC2136] directly to an authoritative unicast DNS server. While it is possible for each host providing a service to send it's own DNS UPDATE, key management has prevented widespread deployment of DNS UPDATES across a domain. By having a limited number of proxies sitting on one or more IP subnets, it is possible to provide secure DNS updates at a manageable scale. Future work to automate secure DNS UPDATES on a larger scale is needed.

This document will explain how services on each .local domain will be mapped into the unicast DNS namespace and how unicast clients will discover these services. It is important to note that no changes are required in either the clients, DNS authoritative servers, or DNS resolver infrastructure. In addition, while the Update proxy is a new logical concept, it requires no new protocols to be defined and can be built using existing DNS libraries.

An Update proxy is an ideal service to run on routers and/or switches to map local services into a larger network infrastructure.

2. Requirements Language

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. These words may also appear in this document in lower case as plain English words, absent their normative meanings.

3. DNS Subdomain Model

Each .local domain which logically maps to an IP subnet is modeled as a separate subdomain in the unicast DNS hierarchy. Each of these subdomains must be browsable (respond to PTR queries for "b._dns-sd._udp.<subdomain>."). See Section 11 of [RFC6763] for more details about browsing. In the context of the Update proxy, these subdomains are typically special use subdomains for mDNS mappings.

3.1. Subdomain Naming

The browseable subdomain label is prepended to the domain name and separated by a period. See [RFC7719] for more information on subdomains and labels. It is not important that the label be human readable or have organizational significance. End users will not be interacting with these labels. The main requirement is that they be unique within the domain for each IP subnet. Subdomain labels can be obtained by the proxy in several ways. The following methods should be attempted in order to assure consistency among redundant proxies:

1. address-derived domain enumeration through local resolver

The proxy issues a PTR query for the registration or browse domains based on the IP subnet. Separate queries are performed for IPv4 and IPv6 on the same link since they are different IP subnets. Since the Update proxy will be registering services with DNS UPDATE, it should begin querying for registration domains and fallback to browse domains if no registration domains are configured.

As an example, suppose a proxy was connected to IPv4 subnet "203.0.113.0/24". In order to determine if there was a subdomain name for this subnet, the base domain name to query would be derived as "0.113.0.203.in-addr.arpa." The proxy would issue a PTR query for the following names in order to find the subdomain for the IP subnet:

```
"dr._dns-sd._udp.0.113.0.203.in-addr.arpa."  
"r._dns-sd._udp.0.113.0.203.in-addr.arpa."  
"db._dns-sd._udp.0.113.0.203.in-addr.arpa."  
"b._dns-sd._udp.0.113.0.203.in-addr.arpa."  
"lb._dns-sd._udp.0.113.0.203.in-addr.arpa."
```

The first response with an answer should be the subdomain name including the domain name for the network and further queries through this list are not needed. If multiple answers are returned in the same response, any one of the answers can be used but the proxy should only use a single subdomain name for the IP subnet.

The Update proxy should periodically rediscover the subdomain name at approximately 5 minute intervals for each IP subnet adding appropriate random jitter across IP subnets so as to prevent synchronization.

2. proxy local configuration override

If no answer is returned, the proxy may have local configuration containing a subdomain name for the network. If so, this subdomain should be used.

3. algorithmic subdomain label generation

If no local configuration is present for the IP subnet, the proxy may generate a unique label and use that for the subdomain by appending a common domain name. One such algorithm is to take the network form of an IPv4 subnet without a prefix length (host portion all zeros) and convert it to a hexadecimal string. This will give a 8 character unique string to use as a subdomain label. For the example above, this label would be "cb007100".

In the cases where through either local configuration or algorithmic generation of subdomain names, a subdomain name is known by the proxy but cannot be address derived through a PTR query, the Update proxy SHOULD register the appropriate address-derived enumeration records through an UPDATE to the zone master. If the IP subnet is removed or becomes inactive on the Update proxy, the proxy MUST attempt to remove these records.

3.2. Domain Name Discovery

The base domain name to use for each subdomain also has to be discovered on a per IP subnet basis. In most cases, the domain name will be the same for all IP subnets because they are all contained in a single administrative domain. However, this is not required and a proxy administrator may need to span multiple administrative boundaries requiring different domain names on different IP subnets (and therefore, subdomains).

There is not a direct query to discover a separate domain name but the domain name is included with the subdomain in the response to the PTR query above in Section 3.1. If the PTR query returns an empty response, then the domain name can be obtained from local proxy configuration and if no domain name is specified there, the default domain for the host should be used.

3.3. Client Service Discovery

Fortunately, clients performing service discovery require no changes in order to work with the Update proxy. Existing clients already support wide-area bonjour which specifies how to query search domains and subdomains for services. See section 11 of [RFC6763].

However, in order for clients to discover the subdomain for each IP subnet, the subdomain MUST be browseable and a browse record for the

domain must enumerate all of the subdomains. If the domain records do not exist, the Update proxy MUST create them in the domain and MUST ensure each subdomain is browseable.

In the future, authoritative unicast DNS servers may add support for DNS Push Notifications [I-D.ietf-dnssd-push] which would allow clients to maintain long lived subscriptions to services. Clients may also wish to add support for this feature to provide an efficient alternative to polling.

4. Update Proxy Behavior

Since no new protocols are defined, this document mostly describes the expected behavior of the Update proxy and how it uses existing protocols to achieve multi IP subnet service discovery. The behavior is mostly intuitive but is described to ensure compatibility and completeness.

4.1. mDNS Service Announcements

The Update proxy should listen to mDNS service announcements (responses) on all interfaces it is proxying for. Multiple Update proxies can be active on the same IP subnet at the same time. See [RFC6762] for more information on multicast DNS.

4.2. Service Caching and Refresh

As specified in Section 8.3 of [RFC6762], service announcements are sent multiple times for redundancy. However, there is no need to send duplicate UPDATE messages to the authoritative unicast DNS server. Therefore, the Update proxy should cache service announcements and only send DNS UPDATE messages when needed.

As described in Section 8.4 of [RFC6762], a host may send "goodbye" announcements by setting the TTL to 0. In this case, the record MUST be removed from the cache or otherwise marked as expired and a DNS UPDATE should be sent to the authoritative unicast DNS server removing the record.

The Update proxy MUST also remove/expire old cache entries and remove the records from the authoritative unicast DNS server when the cache-flush bit is set on new announcements as described in Section 10.2 of [RFC6762].

A host providing a service may automatically refresh the TTL in the announcement from time to time keeping the service valid based on subsequent multicast queries it receives. However, if no mDNS clients are requesting the particular service for the length of the

TTL value, the service announcement could timeout naturally. In order to keep accurate information regarding all of the services on the IP subnet, the Update proxy SHOULD send a unicast PTR query for the service name directly to the host announcing the service. This query should be sent at a random time between 5 and 10 seconds before the TTL value indicates the announcement will expire.

As described in Section 11 of [RFC6762], the Update proxy should use an IP source address of the IP subnet of the interface it is transmitting over and that is on the same IP subnet as the service provider. It is also permissible to use a link-local IP address in the IPv6 case as long as the service itself is available on an IPv6 address that is reachable from outside the local link.

In order for the Update proxy to discover as many services available on each IP subnet as possible, it should periodically send a PTR multicast query for "_services._dns-sd._udp.local." on each subnet. The unicast response bit SHOULD be set in the query in order to force unicast responses to the Update proxy. As PTR responses are received, The Update proxy can then send Service Instance Enumeration PTR queries (also with the unicast response bit set) for each service.

This was not the intended behavior of mDNS since local clients would just ask dynamically when they needed to know all of the providers of a service name but keeping this information up to date in the authoritative server provides benefits to remote clients such as faster response times and ability to use DNSSEC validation that were not previously possible with multicast DNS. These benefits are provided at the additional cost of a slight increase in network activity and processing time by the hosts announcing services. However, if the Update proxy uses unicast to query the service providers directly, other clients are not affected by these refresh queries and do not have to turn their radios on for queries/responses that they have no interest in.

4.3. mDNS Probing

While Section 8.2 of [RFC6762] recommends all potential answers be included in mDNS probe queries, because these records haven't gone through conflict resolution, they should not be regarded as announcements of services. Therefore, an Update proxy MUST NOT rely on information in any section of DNS query messages.

4.4. Link-local Addressing

In the IPv6 case, the source address of the announcements is a link-local IPv6 address that will probably be different than the IP subnet that the service is being provided on. However, it is certainly possible that link-local addressing is used with IPv4 as well. This is not as common but exists in a zero-conf environment where no IPv4 addresses are assigned via DHCP or statically and the hosts revert to link-local IPv4 addresses ("169.254/16"), see [RFC3927].

If the service SRV target resolves to only a link-local address, then the service is not eligible to be advertised outside of the link and shouldn't be sent to the authoritative unicast DNS server by the Update proxy.

In general, the Update proxy needs to ensure that the service is reachable outside of the link it is announced on before sending an UPDATE to the authoritative server for the subdomain.

4.5. IPv6 and IPv4 on Same Link

Announced services may be available on IPv4, IPv6, or both on the same link. If both IPv4 A records [RFC1035] and IPv6 AAAA records [RFC3596] are published for an SRV target [RFC2782] name, the administrator should provide the service over both protocols.

In some cases, this won't be possible. This will not incur any extra delays if clients attempt connections over both IPv4 and IPv6 protocols simultaneously but if one protocol is preferred over another, delays may occur.

4.6. Multiple Logical IP Subnets

Multiple IP subnets on the same link is just a more general case of IPv4 and IPv6 on the same link. When multiple IP subnets exist for the same protocol on the same link, they appear as separate interfaces to the Update proxy and require a separate subdomain name just as IPv4 and IPv6 do.

This is required for a client on one logical IP subnet of an interface to communicate with a service provided by a host on a different IP subnet of the same link.

If a SRV target resolves to addresses on multiple logical IP subnets of the same interface, the service can be included in multiple subdomains on the appropriate server(s) for those subdomains.

4.7. Proxy Redundancy

Providing redundant Update proxies for the same IP subnet can be easily achieved by virtue of the DNS UPDATE protocol. None of the redundant proxies needs to be aware of any of the other redundant proxies on an IP subnet.

Alternatives for ways to format DNS UPDATE messages are defined below in Section 5.2.2 as to possible uses of the Prerequisite section for use with redundant Update proxies.

Alternatively, a proxy MAY choose to hibernate when it discovers another active Update proxy as described below in Section 7.

4.8. Service Filtering and Translation

In the process of registering services with an authoritative unicast DNS server, the proxy can perform filtering and translation on the dynamically discovered services.

As an example, suppose legacy printers are discovered that do not support the current AirPrint feature set. The proxy can alter the TXT record associated with the printer to add the necessary keys as well as any additional service records to allow AirPrint clients to discover and use the legacy printer.

As another example, suppose there is a printer that is behind a locked door where students do not have access. In this case, the printer's resource records MAY be filtered by the proxy so it does not show up during a browse operation on the subnet.

An Update proxy could have rulesets that define the translations it performs on the fly as it learns about matching services.

5. DNS UPDATE

While DNS UPDATE is well supported in authoritative DNS servers, it typically requires some form of authentication for the server to accept the update. The most common form is TSIG [RFC2845], [RFC4635] which is based on a shared secret and a one way hash over the contents of the record.

The Update proxy doesn't dictate a method of privacy or authentication for communication to an authoritative DNS UPDATE server. However, implementations SHOULD ensure some form of authentication exists and even refuse to operate in an environment without authentication.

5.1. Selection of Authoritative Unicast DNS Server

The Update proxy should attempt to locate the authoritative DNS UPDATE server for each subdomain in the following manner:

1. An Update proxy should first send an SRV query for "_dns-update-tls._tcp.<subdomain>." If an answer is received, the target and port number will provide the parameters needed for where to send updates. The proxy can also try the TCP and UDP variants of this service name "_dns-update._tcp" and "_dns-update._udp" if the TLS variant does not exist. If no TLS variant is found, the proxy can still attempt a TLS connection on the SRV port of the TCP or UDP variant. The proxy can also attempt to connect to the target on the reserved port (853) for DNS over TLS as defined in Section 3.1 of [RFC7858].
2. The Update proxy can make a similar query for the same service in the domain if a subdomain specific answer isn't returned: "_dns-update-tls._tcp.<domain>." as well as the TCP and UDP variants.
3. If no matching SRV records are returned, the Update proxy SHOULD consult local configuration policy to see if an DNS UPDATE server has been configured.
4. If no local configuration exists for a DNS UPDATE server, the Update proxy can query the SOA records for the subdomain and try sending updates to the MNAME master server configured for the subdomain. Again, using TLS/TCP is encouraged if available.
5. If DNS UPDATES are not accepted by the server(s) represented by the SOA MNAME master server, then the Update proxy can assume that DNS UPDATES are not available for the subdomain and listening to mDNS announcements on the IP subnet would be unproductive.

5.2. DNS UPDATE Sections

A DNS UPDATE message contains four sections as specified in [RFC2136].

5.2.1. Zone Section

When an Update proxy is adding or removing services to/from a subdomain, the zone section MUST contain a single zone (ZOCOUNT = 1) and the ZNAME MUST be the subdomain being updated. ZTYPE MUST be SOA and ZCLASS MUST be the same class as the records being added/removed.

DNS UPDATES to multiple subdomains MUST be performed in separate DNS UPDATE messages with one subdomain per message.

If a new subdomain is being created for a domain by the Update proxy, the subdomain's parent zone should be used for the ZNAME. ZTYPE MUST be SOA and ZCLASS MUST be the same class as the subdomain's NS record CLASS that is going to be added. Similarly for removing a subdomain.

5.2.2. Prerequisite Section

It is not necessary for the Update proxy to include any prerequisites when adding/removing records. However, if the Update proxy wants to have better error handling, it can add prerequisites to ensure the state of the authoritative server is consistent.

Given that multiple Update proxies may exist for the same IP subnet (and subdomain), it is possible that similar records may be added or deleted to/from the authoritative server before the Update proxy's own messages are processed. This is not to be considered a fatal error and may happen during normal operation of redundant proxies. The use of prerequisite can be used to identify these cases if desired.

5.2.3. Update Section

The Update section contains all of the records that the proxy wants to be added/removed in a single subdomain. If TIMEOUT resource records are being manually added to the authoritative server, they MUST be included as regular resource records in the Update section. See Section 6.3 below for more information.

5.2.4. Additional Data Section

The Update proxy may include additional data as needed. Instances where additional data might be included are:

1. When creating a subdomain by adding new NS records to a domain, A or AAAA glue records MAY be needed. Though, in most cases, the same authoritative server name / IP addresses should be used as in the parent domain.
2. If including a lease lifetime as discussed below in Section 6.3, the OPT recording containing the Update lease will be sent in the additional data section.
3. The TSIG cryptographic signature of the DNS UPDATE message should be the last resource record in the additional data section.

6. DNS Authoritative Server Behavior

The Update proxy will rely on the authoritative server to update the SERIAL number for the zone after each update is completed.

6.1. DNS Push Notifications

An authoritative unicast DNS server MAY support DNS Push notifications [I-D.ietf-dnssd-push] for client queries in order to provide more timely and more efficient responses. While this is outside of the scope of the Update proxy, it is mentioned here for completeness.

6.2. DNSSEC Compatibility

With mDNS, the next domain name field in an NSEC record could not reference the next record in the zone because it was not possible to know all of the records in the zone a priori. By mapping all known records into a unicast subdomain, the NSEC next domain name field can contain the next known record as defined. As new services are discovered and UPDATED in the authoritative unicast DNS server, the NSEC records can be kept up to date by the authoritative server.

The Update proxy will assume that DNS updates sent to zones with DNSSEC enabled will be updated as needed as specified in [RFC3007].

6.3. DNS Update Record Lifetimes

When the Update proxy sends a DNS UPDATE message to an authoritative unicast DNS server, it MAY include a lease lifetime to indicate how long the UPDATE server should keep the resource records active in the zone. This is different from the TTL which tells resolvers how long to keep the records in their cache. Lease lifetimes may be based on different origin data. For example, when an IP address is assigned to a host via DHCP, the DHCP server will provide a time period for which the address is assigned to the host.

There are several possibilities for how a DNS UPDATE server may limit the lifetime of records added via an update message.

1. The DNS update server MAY be configured to automatically delete the records after a certain fixed time period (such as 24 hours). This is a failsafe mechanism in case the origin of the record data goes offline and does not ever try to remove the records.
2. A lease lifetime can be communicated via an OPT record as defined in Dynamic DNS Update Leases [I-D.sekar-dns-ul]. This provides a timeout period for all of the records added in the update message

and is controlled by the sender of the update. This is a work in progress and does not yet have widespread adoption among authoritative unicast DNS server software.

3. Individual DNS TIMEOUT resource records
[I-D.pusateri-dnsop-update-timeout] can be added to the update message to indicate the timeout value for one or any number of the resource records contained in the update message. This is the most flexible but also does not have any adoption among authoritative unicast DNS server software. One advantage of the TIMEOUT resource records is that they are stored in the authoritative server like any other record and synchronized to secondary servers as well. Therefore, if the primary server were to restart or experience an extended failure, the lease lifetime would not be lost.

Note that it is possible to use both the Dynamic DNS Update leases to communicate the lease lifetime and for the authoritative unicast DNS server to create TIMEOUT resource records on demand to achieve the same result if the Update proxy does not include TIMEOUT resource records natively.

7. Transitioning to Unicast

The design of the Update proxy is intended to work within the existing mDNS model and allow it to scale to multiple subnets using existing unicast DNS infrastructure. It is careful not to increase the amount of multicast traffic used for service discovery but it is also capable of providing a transition path to actually reduce multicast usage incrementally. Hosts have generally not been able to directly register their own services through DNS UPDATE because of the security scaling problem of shared passwords needed by TSIG.

By creating trusted infrastructure in the form of an Update proxy, services can now be registered through the Update proxy directly via unicast. This alleviates the need to advertise over mDNS at all.

Therefore, if we allow service providers to discover the Update proxy using either unicast or multicast queries, we can then perform all subsequent communication over unicast. This greatly reduces the multicast usage and the need for every host on the network to wake up and listen to multicast responses and periodic announcements.

In order to accomodate this transition, the Update proxy MAY announce and respond to PTR queries for the update proxy service using service name "_dns-updateproxy-tls._tcp.local." pointing to a local instance name. As host implementations are updated to locate an Update proxy, they can switch entirely to unicast for all of the services they

provide. The SRV record for the instance name can provide a target name and port for which the service provider can connect directly using TLS and send unicast registrations for all of its services. The TXT record for the instance name can contain vendor specific information associated with the Update proxy if desired. There are no required keys in the TXT record and while the record MUST exist (see Section 6 of [RFC6763]), it can consist of a single zero length byte. Multiple announcements can be sent across a single TLS connection. There are no responses expected from the Update proxy after sending the announcements. However, Update proxies MUST only announce this service if they intend to provide a unicast interface for service providers on the local subnet.

Discovery of the Update proxy can also be entirely using unicast for service providers that prefer or are not multicast capable. The service provider would use address-derived domain enumeration as described in Section 3.1 to determine the subdomain name of the local link and then perform a normal unicast PTR query through the local resolver for "_dns-updateproxy-tls._tcp.<subdomain>." to find the Update proxy instance. In the case of redundant proxies, multiple PTR records with different instance names may exist. The SRV records priority and weight fields can be used to determine the preferred Update proxy.

Once these services are initially registered with the Update proxy, the proxy may occasionally sent unicast queries to confirm these services are still active. A 60 second interval with appropriate jitter is recommended for confirmation. If, in the meantime, the service is discontinued, the service provider SHOULD reconnect to the Update proxy and send "goodbye" announcements with TTL 0 as normally would be done with mDNS to expire the services as described in Section 8.4 of [RFC6762]. The Update proxy will not send any response to these "goodbye" announcements, therefore, the connection MUST be closed gracefully to ensure proper delivery.

8. Security Considerations

When a secure DNS UPDATE is sent to an authoritative server, it should not be construed that this information is any more reliable than the original mDNS announcement was for which it was based. Care should always be taken when receiving mDNS announcements to ensure they are source IP address is one that belongs to an IP subnet on the received interface of the Update proxy. In addition, the TTL of the received link local announcement MUST be 1 to ensure it was not forwarded from a remote network.

Each Update proxy requires configuration of a shared secret for creation of the TSIG signature resource record contained as the last record in the UPDATE message.

9. References

9.1. 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>>.
- [RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, DOI 10.17487/RFC2782, February 2000, <<https://www.rfc-editor.org/info/rfc2782>>.
- [RFC2845] Vixie, P., Gudmundsson, O., Eastlake 3rd, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, DOI 10.17487/RFC2845, May 2000, <<https://www.rfc-editor.org/info/rfc2845>>.
- [RFC3007] Wellington, B., "Secure Domain Name System (DNS) Dynamic Update", RFC 3007, DOI 10.17487/RFC3007, November 2000, <<https://www.rfc-editor.org/info/rfc3007>>.
- [RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", STD 88, RFC 3596, DOI 10.17487/RFC3596, October 2003, <<https://www.rfc-editor.org/info/rfc3596>>.
- [RFC3927] Cheshire, S., Aboba, B., and E. Guttman, "Dynamic Configuration of IPv4 Link-Local Addresses", RFC 3927, DOI 10.17487/RFC3927, May 2005, <<https://www.rfc-editor.org/info/rfc3927>>.
- [RFC4635] Eastlake 3rd, D., "HMAC SHA (Hashed Message Authentication Code, Secure Hash Algorithm) TSIG Algorithm Identifiers", RFC 4635, DOI 10.17487/RFC4635, August 2006, <<https://www.rfc-editor.org/info/rfc4635>>.

- [RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <<https://www.rfc-editor.org/info/rfc6762>>.
- [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>>.
- [RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <<https://www.rfc-editor.org/info/rfc7858>>.
- [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>>.

9.2. Informative References

- [I-D.ietf-dnssd-hybrid] Cheshire, S., "Discovery Proxy for Multicast DNS-Based Service Discovery", draft-ietf-dnssd-hybrid-08 (work in progress), March 2018.
- [I-D.ietf-dnssd-mdns-relay] Lemon, T. and S. Cheshire, "Multicast DNS Discovery Relay", draft-ietf-dnssd-mdns-relay-01 (work in progress), July 2018.
- [I-D.ietf-dnssd-push] Pusateri, T. and S. Cheshire, "DNS Push Notifications", draft-ietf-dnssd-push-16 (work in progress), November 2018.
- [I-D.pusateri-dnsop-update-timeout] Pusateri, T. and T. Wattenberg, "DNS TIMEOUT Resource Record", draft-pusateri-dnsop-update-timeout-02 (work in progress), March 2019.
- [I-D.sekar-dns-ul] Cheshire, S. and T. Lemon, "Dynamic DNS Update Leases", draft-sekar-dns-ul-02 (work in progress), August 2018.
- [RFC7719] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", RFC 7719, DOI 10.17487/RFC7719, December 2015, <<https://www.rfc-editor.org/info/rfc7719>>.

Appendix A. Comparison to Discovery Proxy

The Update Proxy defined in this document is an alternative to the Discovery Proxy [I-D.ietf-dnssd-hybrid] and the Discovery Relay [I-D.ietf-dnssd-mdns-relay]. This solution makes different trade-offs than the ones made by the Discovery Proxy which offer some advantages at a cost of increased state.

The main difference is that the Discovery Proxy builds the list of matching services on demand by querying over mDNS and collecting the announcements in response to client queries. Whereas the Update proxy tries to build a complete list of services by listening for all announcements, discovering and refreshing them, and then inserting them into subdomains using DNS UPDATE.

The main advantages of the Update proxy include limiting further propagation of IP multicast across the campus, providing a pathway to eliminate multicast entirely, faster response time to client queries, and the ability to provide DNSSEC signed security responses for client queries.

While the Discovery Proxy increases multicast queries and responses based on received unicast queries " $O(n^2)$ ", the Update proxy can reduce or eliminate mDNS traffic on the local links " $O(1)$ ".

Another key difference is that the Update proxy never becomes an authoritative unicast DNS server for the attached subdomain. It simply updates the existing authoritative server for the domain. Therefore, the administrator is free to use existing authoritative DNS server infrastructure.

Author's Address

Tom Pusateri
Unaffiliated
Raleigh NC 27608
USA

Phone: +1 (919) 867-1330
Email: pusateri@bangj.com