Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 11, 2018

T. Lemon
Nibbhaya Consulting
S. Cheshire
Apple Inc.

May 10, 2018

Multicast DNS Discovery Relay
draft-ietf-dnssd-mdns-relay-00

Abstract

This document extends the specification of the Discovery Proxy for Multicast DNS-Based Service Discovery. It describes a lightweight relay mechanism, a Discovery Relay, which, when present on a link, allows Discovery Proxies not attached to that link to provide mDNS proxy service on that link.

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 http://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 November 11, 2018.

Table of Contents

1. Introduction

The Discovery Proxy for Multicast DNS-Based Service Discovery [I-D.ietf-dnssd-hybrid] is a mechanism for discovering services on a subnetted network through the use of Discovery Proxies, which issue Multicast DNS (mDNS) requests [RFC6762] on various multicast links in the network on behalf of a remote host performing DNS-Based Service Discovery [RFC6763].

In the original Discovery Proxy specification, it is imagined that for every multicast link on which services will be discovered, a host will be present running a full Discovery Proxy. This document introduces a lightweight Discovery Relay that can be used to provide discovery services on a multicast link without requiring a full Discovery Proxy on every multicast link.

Since the primary purpose of a Discovery Relay is providing remote virtual interface functionality to Discovery Proxies, this document is written with that in mind. However, in principle, a Discovery Relay could be used by any properly authorized client. In the context of this specification, a Discovery Proxy is a client to the Discovery Relay. This document uses the terms "Discovery Proxy" and "Client" to mean the same thing; the term "Client" is used when we are talking about the communication between the client and the Relay, and the term "Discovery Proxy" when we are referring specifically to something that is acting as a Discovery Proxy.

The Discovery Relay operates by listening for TCP connections from Clients. When a Client connects, the connection is authenticated and secured using TLS. The Client can then specify one or more multicast links from which it wishes to receive mDNS traffic. The Client can also send messages to be transmitted on its behalf on one or more of those multicast links. DNS Stateful Operations (DSO) [I-D.ietf-dnsop-session-signal] is used as a framework for conveying interface and IP header information associated with each message.

The Discovery Relay functions essentially as a set of one or more remote virtual interfaces for the Client, one on each multicast link to which the Discovery Relay is connected. In a complex network, it is possible that more than one Discovery Relay will be connected to the same multicast link; in this case, the Client ideally should only be using one such Relay Proxy per multicast link, since using more than one will generate duplicate traffic.

How such duplication is detected and avoided is out of scope for this document; in principle it could be detected using HNCP [RFC7788] or configured using some sort of orchestration software in conjunction with NETCONF [RFC6241] or CPE WAN Management Protocol [TR-069].

2. Terminology

The following definitions may be of use:

Client A network service that uses a Discovery Relay to receive mDNS multicast traffic and/or communicate with mDNS Agents.

mDNS Agent A host which sends and/or responds to mDNS queries.

Discovery Proxy A network service which receives well-formed questions using the DNS protocol, performs multicast DNS queries to answer those questions, and responds with those answers using the DNS protocol. A Discovery Proxy that can communicate with mDNS Agents using a Discovery Relay is also a Client.

Discovery Relay A network service which relays received mDNS messages to a Client, and can transmit mDNS messages on behalf of that Client.

multicast link A maximal set of network connection points, such that any host connected to any connection point in the set may send a packet with a link-local multicast destination address (specifically the mDNS link-local multicast destination address [RFC6762]) that will be received by all hosts connected to all other connection points in the set. Note that it is becoming increasingly common for a multicast link to be smaller than its corresponding unicast link. For example it is becoming common to have multiple Wi-Fi Access Points on a shared Ethernet backbone, where the multiple Wi-Fi Access Points and their shared Ethernet backbone form a single unicast link (a single IPv4 subnet, or single IPv6 prefix) but not a single multicast link. Unicast packets between two hosts on that IPv4 subnet or IPv6 prefix are correctly delivered, but multicast packets are not forwarded between the various Wi-Fi Access Points. Given the slowness of Wi-Fi multicast, the decision to not forward multicast packets between Wi-Fi Access Points is reasonable, and that further supports the need for technologies like Discovery Proxy and Discovery Relay to facilitate discovery on these networks.

whitelist A list of one or more IP addresses from which a Discovery Relay may accept connections.

silently discard When a message that is not supported or not permitted is received, and the required response to that message is to "silently discard" it, that means that no response is sent by the service that is discarding the message to the service that sent it. The service receiving the message may log the event, and

may also count such events: "silently" does not preclude such behavior.

3. Protocol Overview

This document describes a way for Discovery Proxies to communicate with mDNS agents on remote multicast links to which they are not directly connected, using a Discovery Relay. As such, there are two parts to the protocol: connections between Discovery Proxies and Discovery Relays, and communications between Discovery Relays and mDNS agents.

3.1. Connections between Proxies and Relays (overview)

Discovery Relays listen for incoming connection requests. Connections between Clients and Discovery Relays are established by Clients. Connections are authenticated and encrypted using TLS, with both client and server certificates. Connections are long-lived: a Client is expected to send many queries over a single connection, and Discovery Relays will forward all mDNS traffic from subscribed interfaces over the connection.

The stream encapsulated in TLS will carry DNS frames as in the DNS TCP protocol [RFC1035] Section 4.2.2. However, all messages will be DSO messages [I-D.ietf-dnsop-session-signal]. There will be four types of such messages between Discovery Relays and Clients:

- Control messages from Client to Relay
- Link status messages from Relay to Client
- mDNS messages from Client to Relay
- mDNS messages from Relay to Client

Clients can send four different control messages to Relays: the Link Request, Link Discontinue, Report Link Changes and Stop Reporting Link Changes messages. The first two are used by the Client to indicate to the Discovery Relay that mDNS messages from one or more specified multicast links are to be relayed to the Discovery Proxy, and to subsequently stop such relaying. The second two are used by the Client to request that the Relay report on the set of links that can be requested, and to request that it discontinue such reporting.

Link Status messages from the Relay to the Client inform the client that a link has become available, or that a formerly-available link is no longer available.

mDNS messages from a Discovery Proxy to a Discovery Relay cause the Discovery Relay to transmit the mDNS message on one or more multicast links to which the Discovery Relay host is directly attached.

mDNS messages from a Discovery Relay to a Discovery Proxy are sent whenever an mDNS message is received on a multicast link to which the Discovery Relay has subscribed.

During periods with no traffic flowing, Clients are responsible for generating any necessary keepalive traffic, as stated in the DSO specification [I-D.ietf-dnsop-session-signal].

3.2. mDNS Messages On Multicast Links

Discovery Relays listen for mDNS traffic on all configured multicast links that have at least one active subscription from a Discovery Proxy. When an mDNS message is received on a multicast link, it is forwarded on every open Client connection that is subscribed to mDNS traffic on that multicast link. In the event of congestion, where a particular Client connection has no buffer space for an mDNS message that would otherwise be forwarded to it, the mDNS message is not forwarded to it. Normal mDNS retry behavior is used to recover from this sort of packet loss. Discovery Relays are not expected to buffer more than a few mDNS packets. Excess mDNS packets are silently discarded. In practice this is not expected to be a issue. Particularly on networks like Wi-Fi, multicast packets are transmitted at rates ten or even a hundred times slower than unicast packets. This means that even at peak multicast packets rates, it is likely that a unicast TCP connection will able to carry those packets with ease.

Clients send mDNS messages they wish to have sent on their behalf on remote multicast link(s) on which the Client has an active subscription. A Discovery Relay will not transmit mDNS packets on any multicast link on which the Client does not have an active subscription, since it makes no sense for a Client to ask to have a query sent on its behalf if it's not able to receive the responses to that query.

4. Connections between Proxies and Relays (details)

When a Discovery Relay starts, it opens a passive TCP listener to receive incoming connection requests from Clients. This listener may be bound to one or more source IP addresses, or to the wildcard address, depending on the implementation. When a connection is received, the relay must first validate that it is a connection to an IP address to which connections are allowed. For example, it may be that only connections to ULAs are allowed, or to the IP addresses

configured on certain interfaces. If the listener is bound to a specific IP address, this check is unnecessary.

If the relay is using an IP address whitelist, the next step is for the relay to verify that that the source IP address of the connection is on its whitelist. If the connection is not permitted either because of the source address or the destination address, the Discovery Relay responds to the TLS Client Hello message from the Client with a TLS user_canceled alert ([I-D.ietf-tls-tls13] Section 6.1).

Otherwise, the Discovery Relay will attempt to complete a TLS handshake with the Client. Clients are required to send the post_handshake_auth extension ([I-D.ietf-tls-tls13] Section 4.2.5). If a Discovery Relay receives a ClientHello message with no post_handshake_auth extension, the Discovery Relay rejects the connection with a certificate_required alert ([I-D.ietf-tls-tls13] Section 6.2).

Once the TLS handshake is complete, the Discovery Relay MUST request post-handshake authentication as described in ([I-D.ietf-tls-tls13] Section 4.6.2). If the Client refuses to send a certificate, or the key presented does not match the key associated with the IP address from which the connection originated, or the CertificateVerify does not validate, the connection is dropped with the TLS access_denied alert ([I-D.ietf-tls-tls13] Section 6.2).

Once the connection is established and authenticated, it is treated as a DNS TCP connection [RFC1035].

Aliveness of connections between Clients and Relays is maintained as described in Section 4 of [I-D.ietf-dnsop-session-signal]. Clients must also honor the 'Retry Delay' TLV (section 5 of [I-D.ietf-dnsop-session-signal]) if sent by the Discovery Relay.

Clients may establish more than one connection to a specific Discovery Relay. This would happen in the case that a TCP connection stalls, and the Client is able to reconnect before the previous connection has timed out. It could also happen as a result of a server restart. It is not likely that two active connections from the same Client would be present at the same time, but it must be possible for additional connections to be established. The Discovery Relay may drop the old connection when the new one has been fully established, including a successful TLS handshake. What it means for two connections to be from the same Client is that the connections both have source addresses that belong to the same Client, and that they were authenticated using the same client certificate.

In order to know what links to request, the Client can be configured with a list of links supported by the Relay. However, in some networking contexts, dynamic changes in the availability of links are likely; therefore Clients may also use the Report Link Changes TLV to request that the Relay report on the availability of its links. In some contexts, for example when debugging, a Client may operate with no information about the set of links supported by a relay, simply relying on the relay to provide one.

5. Traffic from Relays to Clients

The mere act of connecting to a Discovery Relay does not result in any mDNS traffic being forwarded. In order to request that mDNS traffic from a particular multicast link be forwarded on a particular connection, the Client must send one or more DSO messages, each containing a single mDNS Link Request TLV (Section 8.1) indicating the multicast link from which traffic is requested.

When an mDNS Link Request message is received, the Discovery Relay validates that it is connected to the specified multicast link, and that forwarding is enabled for that link. If both checks are successful, it MUST send a response with RCODE=0 (NOERROR). If the Relay is not connected to the specified link, or forwarding from that link to the Client is not enabled, it sends a response with RCODE=3 (NXDOMAIN/Name Error). It is not an error to request the same link more than once; the Relay MUST NOT reject an mDNS Link Request on that basis. If the relay cannot satisfy the request for some reason other than that the link is invalid, for example resource exhaustion, it sends a response with RCODE=2 (SERVFAIL).

If the requested link is valid, the Relay begins forwarding all mDNS traffic from that link to the Client. Delivery is not guaranteed: if there is no buffer space, packets will be dropped. It is expected that regular mDNS retry processing will take care of retransmission of lost packets. The amount of buffer space is implementation dependent, but generally should not be more than the bandwidth delay product of the TCP connection [RFC1323]. The Discovery Relay should use the TCP_NOTSENT_LOWAT mechanism [NOTSENT][PRIO] or equivalent, to avoid building up a backlog of data in excess of the amount necessary to have in flight to fill the bandwidth delay product of the TCP connection.

mDNS messages from Relays to Clients are framed within DSO messages. Each DSO message can contain multiple TLVs, but only a single mDNS message is conveyed per DSO message. Each forwarded mDNS message is contained in an mDNS Message TLV (Section 8.4). The layer two source address of the message, if known, MAY be encoded in a Layer Two Source TLV (Section 8.5). The source IP address and port of the

message MUST be encoded in an IP Source TLV (Section 8.6). The multicast link on which the message was received MUST be encoded in a Link Identifier TLV (Section 8.3). The Client MUST silently ignore unrecognized TLVs in mDNS messages, and MUST NOT discard mDNS messages that include unrecognized TLVs.

A Client may discontinue listening for mDNS messages on a particular multicast link by sending a DSO message containing an mDNS Link Discontinue TLV (Section 8.2). Subsequent messages from that link that had previously been queued may arrive after listening has been discontinued. The Client should silently discard such messages. The Discovery Relay MUST discontinue generating such messages as soon as the request is received. The Discovery Relay does not respond to this message other than to discontinue forwarding mDNS messages from the specified links.

6. Traffic from Clients to Relays

Like mDNS traffic from relays, each mDNS message sent by a Client to a Discovery Relay is encapsulated in an mDNS Message TLV (Section 8.4) within a DSO message. Each message MUST contain one or more Link Identifier TLVs (Section 8.3). The Discovery Relay will transmit the message to the mDNS port and multicast address on each link specified in the message using the specified IP address family.

Although the communication between Clients and Relays uses the DNS stream protocol and DNS Stateless Operations, there is no case in which a Client would legitimately send a DNS query (something other than a DSO message) to a Relay. Therefore, if a Relay receives a message other than a DSO message, it MUST respond with a REFUSED result code. The reason not to simply drop the connection is that it might result in a continual reconnection loop.

7. Discovery Proxy Behavior

Discovery Proxies treat multicast links for which Discovery Relay service is being used as if they were virtual interfaces; in other words, a Discovery Proxy serving multiple multicast links using multiple Discovery Relays behaves the same as a Discovery Proxy serving multiple multicast links using multiple physical network interfaces. In this section we refer to multicast links served directly by the Discovery Proxy as locally-connected links, and multicast links served through the Discovery Relay as relay-connected links.

What this means is that when a Discovery Proxy receives a DNSSD query from a client via unicast, it will generate mDNS query messages on the relevant multicast link(s) for which it is acting as a proxy.

For locally-connected link(s), those query messages will be sent directly. For relay-connected link(s), the query messages will be sent through the Discovery Relay that is being used to serve that multicast link.

Responses from devices on locally-connected links are processed normally. Responses from devices on relay-connected links are received by the Discovery Relay, encapsulated, and forwarded to the Discovery Proxy; the Discovery Proxy then processes these messages using the link-identifying information included in the encapsulation.

Discovery Proxies do not generally respond to mDNS queries on relay-connected links. The one exception is responding to the Domain Enumeration queries used to bootstrap unicast service discovery ("lb._dns-sd._udp.local", etc.) [RFC6763]. Apart from these Domain Enumeration queries, if any other mDNS query is received from a Discovery Relay, the Discovery Proxy silently discards it.

In principle it could be the case that some device is capable of performing service discovery using Multicast DNS, but not using traditional unicast DNS. Responding to mDNS queries received from the Discovery Relay could address this use case. However, continued reliance on multicast is counter to the goals of the current work in service discovery, and to benefit from wide-area service discovery such client devices should be updated to support service discovery using unicast queries.

8. DSO TLVs

This document defines a modest number of new DSO TLVs.

8.1. mDNS Link Request

The mDNS Link Request TLV conveys a link identifier from which a Client is requesting that a Discovery Relay forward mDNS traffic. The link identifier comes from the provisioning configuration (see Section 9). The DSO-TYPE for this TLV is TBD-R. DSO-LENGTH is always 5. DSO-DATA is the 8-bit address family followed by the 32-bit link identifier, in network byte order, as described in Section 9. An address family value of 1 indicates IPv4 and 2 indicates IPv6, as recorded in the IANA Registry of Address Family Numbers [AdFam].

The mDNS Link Request TLV can only be used as a primary TLV, and requires an acknowledgement.

At most one mDNS Link Request TLV may appear in a DSO message. To request multiple link subscriptions, multiple separate DSO messages are sent, each containing a single mDNS Link Request TLV.

8.2. mDNS Link Discontinue

The mDNS Link Discontinue TLV is used by Clients to unsubscribe to mDNS messages on the specified multicast link. DSO-TYPE is TBD-D. DSO-LENGTH is always 5. DSO-DATA is the 8-bit address family followed by the 32-bit link identifier, in network byte order, as described in Section 9.

The mDNS Link Discontinue TLV can only be used as a primary TLV, and is not acknowledged.

At most one mDNS Link Discontinue TLV may appear in a DSO message. To unsubscribe from multiple links, multiple separate DSO messages are sent, each containing a single mDNS Link Discontinue TLV.

8.3. Link Identifier

This option is used both in DSO messages from Discovery Relays to Clients that contain received mDNS messages, and from Clients to Discovery Relays that contain mDNS messages to be transmitted on the multicast link. In the former case, it indicates the multicast link on which the message was received; in the latter case, it indicates the multicast link on which the message should be transmitted. DSO-TYPE is TBD-L. DSO-LENGTH is always 5. DSO-DATA is the 8-bit

address family followed by the 32-bit link identifier, in network byte order, as described in Section 9.

The Link Identifier TLV can only be used as an additional TLV.

8.4. mDNS Message

The mDNS Message TLV is used to encapsulate an mDNS message that a Relay is forwarding from a multicast link to a Client, or that a Client is sending to a Relay for transmission on a multicast link. Only the application layer payload of the mDNS message is carried in the DSO mDNS Message TLV, i.e., just the DNS message itself, beginning with the DNS Message ID, not the IP or UDP headers. The DSO-TYPE for this TLV is TBD-M. DSO-LENGTH is the length of the encapsulated mDNS message. DSO-DATA is the content of the encapsulated mDNS message.

The mDNS Message TLV can only be used as a primary TLV, and is not acknowledged.

8.5. Layer Two Source Address

The Layer Two Source Address TLV is used to report the link-layer address from which an mDNS message was received. This TLV is optionally present in DSO messages from Discovery Relays to Clients that contain mDNS messages when the source link-layer address is known. The DSO-TYPE is TBD-2. DSO-LENGTH is variable, depending on the length of link-layer addresses on the link from which the message was received. DSO-DATA is the link-layer address as it was received on the link.

The Layer Two Source Address TLV can only be used as an additional TLV.

8.6. IP Source

The IP Source TLV is used to report the IP source address and port from which an mDNS message was received. This TLV is present in DSO messages from Discovery Relays to Clients that contain mDNS messages. DSO-TYPE is TBD-A. DSO-LENGTH is either 6, for an IPv4 address, or 18, for an IPv6 address. DSO-DATA is the two-byte source port, followed by the 4- or 16-byte IP Address, in network byte order.

The IP Source TLV can only be used as an additional TLV.

8.7. Report Link Changes

The Report Link Changes TLV requests that the Discovery Relay report link changes. When the relay is reporting link changes and a new link becomes available, it sends a Link Available message to the Client. When a link becomes unavailable, it sends a Link Unavailable message to the Client. If there are links available when the request is received, then for each such link the relay immediately sends a Link Available Message to the Client. DSO-TYPE is TBD-P. DSO-LENGTH is 0.

The mDNS Report Link Changes TLV can only be used as a primary TLV, and requires an acknowledgement. The acknowledgment does not contain a Link Available TLV: it is just a response to the Report Link Changes message.

8.8. Stop Reporting Link Changes

The Stop Reporting Link Changes TLV requests that the Discovery Relay stop reporting on the availability of links supported by the relay. This cancels the effect of a Report Link Changes TLV. DSO-TYPE is TBD-S. DSO-LENGTH is 0.

The mDNS Stop Reporting Link Changes TLV can only be used as a primary TLV, and is not acknowledged.

8.9. Link Available

The Link Available TLV is used by Discovery Relays to indicate to clients that a new link has become available. The format is the same as the Link Identifier TLV. DSO-TYPE is TBD-V. The Link Available TLV may be accompanied by one or more Link Prefix TLVs which indicate IP prefixes the Relay knows to be present on the link.

The mDNS Link Available TLV can only be used as a primary TLV, and is not acknowledged.

8.10. Link Unavailable

The Link Unavailable TLV is used by Discovery Relays to indicate to clients that an existing link has become unavailable. The format is the same as the Link Identifier TLV. DSO-TYPE is TBD-U.

The mDNS Link Unavailable TLV can only be used as a primary TLV, and is not acknowledged.

8.11. Link Prefix

The Link Prefix TLV represents an IP address or prefix configured on a link. The length is 17 for an IPv6 address or prefix, and 5 for an IPv4 address or prefix. The TLV consists of a prefix length, between 0 and 32 for IPv4 or between 0 and 128 for IPv6, represented as a single byte. This is followed by the IP address, either four or sixteen bytes. DSO-TYPE is TBD-K.

The Link Prefix TLV can only be used as a secondary TLV.

9. Provisioning

In order for a Discovery Proxy to use Discovery Relays, it must be configured with sufficient information to identify multicast links on which service discovery is to be supported and connect to discovery relays supporting those multicast links, if it is not running on a host that is directly connected to those multicast links.

A Discovery Relay must be configured both with a set of multicast links to which the host on which it is running is connected, on which mDNS relay service is to be provided, and also with a list of one or more Clients authorized to use it.

On a network supporting DNS Service Discovery using Discovery Relays, more than one different Discovery Relay implementation is likely be present. While it may be that only a single Discovery Proxy is present, that implementation will need to be able to be configured to interoperate with all of the Discovery Relays that are present. Consequently, it is necessary that a standard set of configuration parameters be defined for both Discovery Proxies and Discovery Relays.

DNS Service Discovery generally operates within a constrained set of links, not across the entire internet. This section assumes that what will be configured will be a limited set of links operated by a single entity or small set of cooperating entities, among which services present on each link should be available to users on that link and every other link. This could be, for example, a home network, a small office network, or even a network covering an entire building or small set of buildings. The set of Discovery Proxies and Discovery Relays within such a network will be referred to in this section as a 'Discovery Domain'.

Depending on the context, several different candidates for configuration of Discovery Proxies and Discovery relays may be applicable. The simplest such mechanism is a manual configuration file, but regardless of provisioning mechanism, certain configuration

information needs to be communicated to the devices, as outlined below.

9.1. Provisioned Objects

Three types of objects must be described in order for Discovery Proxies and Discovery Relays to be provisioned: Discovery Proxies, Multicast Links, and Discovery Relays. "Human-readable" below means actual words or proper names that will make sense to an untrained human being. "Machine-readable" means a name that will be used by machines to identify the entity to which the name refers. Each entity must have a machine-readable name and may have a human-readable name. No two entities can have the same human-readable name. Similarly, no two entities can have the same machine-readable name.

9.1.1. Multicast Link

The description of a multicast link consists of:

link-identifier A 32-bit identifier that uniquely identifies that link within the Discovery Domain. Each link MUST have exactly one such identifier. Link Identifiers do not have any special semantics, and are not intended to be human-readable.

ldh-name A fully-qualified domain name for the multicast link that is used to form an LDH domain name as described in section 5.3 of the Discovery Proxy specification [I-D.ietf-dnssd-hybrid]. This name is used to identify the link during provisioning, and must be present.

hr-name A human-readable user-friendly fully-qualified domain name for the multicast link. This name MUST be unique within the Discovery Domain. Each multicast link MUST have exactly one such name. The hr-name MAY be the same as the ldh-name. (The hr-name is allowed to contain spaces, punctuation and rich text, but it is not required to do so.)

The ldh-name and hr-name can be used to form the LDH and human-readable domain names as described in [I-D.ietf-dnssd-hybrid], section 5.3.

Note that the ldh-name and hr-name can be used in two different ways.

On a small home network with little or no human administrative configuration, link names may be directly visible to the user. For example, a search in 'home.arpa' on a small home network may discover services on both ethernet.home.arpa and wi-fi.home.arpa. In the case

of a home user who has one Ethernet-connected printer and one Wi-Fi-connected printer, discovering that they have one printer on ethernet.home.arpa and another on wi-fi.home.arpa is understandable and meaningful.

On a large corporate network with hundreds of Wi-Fi Access Points, the individual link names of the hundreds of multicast links are less likely to be useful to end users. In these cases, Discovery Broker functionality [I-D.sctl-discovery-broker] is used to translate the many link names to something more meaningful to users. For example, in a building with 50 Wi-Fi Access Points, each with their own link names, services on all the different physical links may be presented to the user as appearing in 'headquarters.example.com'. In this case, the individual link names can be thought of similar to MAC addresses or IPv6 addresses. They are used internally by the software as unique identifiers, but generally are not exposed to end users.

9.1.2. Discovery Proxy

The description of a Discovery Proxy consists of:

name a machine-readable name used to reference this Discovery Proxy in provisioning.

hr-name an optional human-readable name which can appear in provisioning, monitoring and debugging systems. Must be unique within a Discovery Domain.

public-key a public key that identifies the Discovery Proxy. This key can be shared across services on the Discovery Proxy Host. The public key is used both to uniquely identify the Discovery Proxy and to authenticate connections from it.

private-key the private key corresponding to the public key.

source-ip-addresses a list of IP addresses that may be used by the Discovery Proxy when connecting to Discovery Relays. These addresses should be addresses that are configured on the Discovery Proxy Host. They should not be temporary addresses. All such addresses must be reachable within the Discovery Domain.

public-ip-addresses a list of IP addresses that may be used to submit DNS queries to the Discovery Proxy. This is not used for interoperation with Discovery Relays, but is mentioned here for completeness: this list of addresses may differ from the 'source-ip-addresses' list. If any of these addresses are reachable from

outside of the Discovery Domain, services in that domain will be discoverable outside of the domain.

multicast links a list of multicast links on which this Discovery Proxy is expected to provide service

The private key should never be distributed to other hosts; all of the other information describing a Discovery Proxy can be safely shared with Discovery Relays.

In some configurations it may make sense for the Discovery Relay not to have a list of links, but simply to support the set of all links available on relays to which the Proxy is configured to communicate.

9.1.3. Discovery Relay

The description of a Discovery Relay consists of:

name a required machine-readable identifier used to reference the relay

hr-name an optional human-readable name which can appear in provisioning, monitoring and debugging systems. Must be unique within a Discovery Domain.

public-key a public key that identifies the Discovery Relay. This key can be shared across services on the Discovery Relay Host. Indeed, if a Discovery Proxy and Discovery Relay are running on the same host, the same key may be used for both. The public key uniquely identifies the Discovery Relay and is used by the Discovery Proxy to verify that it is talking to the intended Discovery Relay after a TLS connection has been established.

private-key the private key corresponding to the public key.

connect-tuples a list of IP address/port tuples that may be used to connect to the Discovery Relay. The relay may be configured to listen on all addresses on a single port, but this is not required, so the port as well as the address must be specified.

multicast links a list of multicast links to which this relay is physically connected.

The private key should never be distributed to other hosts; all of the other information describing a Discovery Relay can be safely shared with Discovery Proxies.

In some cases a Relay may not be configured with a static list of links, but may simply discover links by monitoring the set of available interfaces on the host on which the Relay is running. In that case, the relay could be configured to identify links based on the names of network interfaces, or based on the set of available prefixes seen on those interfaces. This sort of configuration is out of scope for this discussion and is left as an exercise for the reader.

9.2. Configuration Files

For this discussion, we assume the simplest possible means of configuring Discovery Proxies and Discovery Relays: the configuration file. Any environment where changes will happen on a regular basis will either require some automatic means of generating these configuration files as the network topology changes, or will need to use a more automatic method for configuration, such as HNCP [RFC7788].

There are many different ways to organize configuration files. This discussion assumes that multicast links, relays and proxies will be specified as objects, as described above, perhaps in a master file, and then the specific configuration of each proxy or relay will reference the set of objects in the master file, referencing objects by name. This approach is not required, but is simply shown as an example. In addition, the private keys for each proxy or relay must appear only in that proxy or relay's configuration file.

The master file contains a list of Discovery Relays, Discovery Proxies and Multicast Links. Each object has a name and all the other data associated with it. We do not formally specify the format of the file, but it might look something like this:

```
Relay upstairs
  public-key xxx
  connect-tuple 192.0.2.1 1917
  connect-tuple fd00::1 1917
  link upstairs-wifi
  link upstairs-wired
Relay downstairs
  public-key yyy
  connect-tuple 192.51.100.1 2088
  connect-tuple fd00::2 2088
  link downstairs-wifi
  link downstairs-wired
Proxy main
  public-key zzz
  address 203.1.113.1
Link upstairs-wifi
  id 1
  name Upstairs Wifi
Link upstairs-wired
  id 2
  hr-name Upstairs Wired
Link downstairs-wifi
  id 3
  name Downstairs Wifi
Link downstairs-wired
  id 4
  hr-name Downstairs Wired
```

9.3. Discovery Proxy Configuration

The Discovery Proxy configuration contains enough information to identify which Discovery Proxy is being configured, enumerate the list of multicast links it is intended to serve, and provide keying information it can use to authenticate to Discovery Relays. It may also contain custom information about the port and/or IP address(es) on which it will respond to DNS queries.

An example configuration, following the convention used in this section, might look something like this:

```
Proxy main
  private-key zzz
  subscribe upstairs-wifi
  subscribe downstairs-wifi
  subscribe upstairs-wired
  subscribe downstairs-wired
```

When combined with the master file, this configuration is sufficient for the Discovery Proxy to identify and connect to the relay proxies that serve the links it is configured to support.

9.4. Discovery Relay Configuration

The discovery relay configuration just needs to tell the discovery relay what name to use to find its configuration in the master file, and what the private key is corresponding to its public key in the master file. For example:

```
Relay Downstairs
  private-key yyy
```

10. Security Considerations

Part of the purpose of the Multicast DNS Discovery Relay protocol is to place a simple relay, analogous to a BOOTP relay, into routers and similar devices that may not be updated frequently. The BOOTP [RFC0951] protocol has been around since 1985, and continues to be useful today. The BOOTP protocol uses no encryption, and in many enterprise networks this is considered acceptable. In contrast, the relay protocol requires TLS 1.3. A concern is that after 20 or 30 years TLS 1.3, or some of the encryption algorithms it uses, may become obsolete, rendering devices that require it unusable. Our assessment is that TLS 1.3 probably will be around for many years to come. TLS 1.0 [RFC2246] was used for about a decade, and similarly TLS 1.2 [RFC5246] was also used for about a decade. We expect TLS 1.3 [I-D.ietf-tls-tls13] to have at least that lifespan. In addition, recent IETF efforts are pushing for better software update practices for devices like routers, for other security reasons, making less likely that in ten years time it will be less common to be using routers that haven't had a software update for ten years. However, authors of encryption specifications and libraries should be aware of the potential backwards compatibility issues if an encryption algorithm becomes deprecated. This specification RECOMMENDS that if an encryption algorithm becomes deprecated, then rather than remove that encryption algorithm entirely, encryption libraries should disable that encryption algorithm by default, but leave the code present with an option for client software to enable it in special cases, such as a Multicast DNS Discovery Relay client talking to an ancient Multicast DNS Discovery Relay server. Using no encryption like BOOTP would eliminate this backwards compatibility concern, but we feel that in such a future hypothetical scenario, using even a weak encryption algorithm still makes passive eavesdropping and tampering harder, and is preferable to using no encryption at all.

11. IANA Considerations

The IANA is kindly requested to update the DSO Type Codes Registry [I-D.ietf-dnsop-session-signal] by allocating codes for each of the TBD type codes listed in the following table, and by updating this document, here and in Section 8. Each type code should list this document as its reference document.

Opcode	Status	Name
TBD-R	Standard	mDNS Link Request
TBD-D	Standard	mDNS Discontinue
TBD-L	Standard	Link Identifier
TBD-M	Standard	mDNS Messsage
TBD-2	Standard	Layer Two Source Address
TBD-A	Standard	IP Source
TBD-P	Standard	Report Link Changes
TBD-S	Standard	Stop Reporting Link Changes
TBD-V	Standard	Link Available
TBD-U	Standard	Link Unavailable
TBD-K	Standard	Link Prefix

DSO Type Codes to be allocated

12. Acknowledgments

13. References

13.1. 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>.

[RFC1323] Jacobson, V., Braden, R., and D. Borman, "TCP Extensions for High Performance", RFC 1323, DOI 10.17487/RFC1323, May 1992, <https://www.rfc-editor.org/info/rfc1323>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[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>.

[RFC7788] Stenberg, M., Barth, S., and P. Pfister, "Home Networking Control Protocol", RFC 7788, DOI 10.17487/RFC7788, April 2016, <https://www.rfc-editor.org/info/rfc7788>.

[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.sctl-discovery-broker] Cheshire, S. and T. Lemon, "Service Discovery Broker", draft-sctl-discovery-broker-00 (work in progress), July 2017.

[I-D.ietf-dnsop-session-signal] Bellis, R., Cheshire, S., Dickinson, J., Dickinson, S., Lemon, T., and T. Pusateri, "DNS Stateful Operations", draft-ietf-dnsop-session-signal-07 (work in progress), March 2018.

[I-D.ietf-tls-tls13]
Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-28 (work in progress), March 2018.

13.2. Informative References

[RFC0951] Croft, W. and J. Gilmore, "Bootstrap Protocol", RFC 951, DOI 10.17487/RFC0951, September 1985, <https://www.rfc-editor.org/info/rfc951>.

[RFC2246] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, DOI 10.17487/RFC2246, January 1999, <https://www.rfc-editor.org/info/rfc2246>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[TR-069] Broadband Forum, "CPE WAN Management Protocol", November 2013, <https://www.broadband-forum.org/technical/download/TR-069_Amendment-5.pdf>.

[NOTSENT] "TCP_NOTSENT_LOWAT socket option", July 2013, <https://lwn.net/Articles/560082/>.

[PRIO] "Prioritization Only Works When There's Pending Data to Prioritize", January 2014, <https://insouciant.org/tech/prioritization-only-works-when-theres-pending-data-to-prioritize/>.

[AdFam] "IANA Address Family Numbers Registry", <https://www.iana.org/assignments/address-family-numbers/>.

Authors' Addresses

Ted Lemon
Nibbhaya Consulting
P.O. Box 958
Brattleboro, Vermont 05301
United States of America

Email: mellon@fugue.com

Stuart Cheshire
Apple Inc.
1 Infinite Loop
Cupertino, California 95014
USA

Phone: +1 408 974 3207
Email: cheshire@apple.com