Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: January 3, 2018

S. Cheshire
Apple Inc.

July 2, 2017

Service Discovery Road Map
draft-cheshire-dnssd-roadmap-00

Abstract

Over the course of several years, a rich collection of technologies has developed around DNS-Based Service Discovery, described across multiple documents. This "Road Map" document gives an overview of how these separate but related technologies (and their documents) fit together, to facilitate Service Discovery in various environments.

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 January 3, 2018.

Copyright Notice

1. Road Map

DNS-Based Service Discovery [RFC6763] is a component of Zero Configuration Networking [RFC6760] [ZC] [Roadmap].

Over the course of several years, a rich collection of technologies has developed around DNS-Based Service Discovery. These various separate but related technologies are described across multiple documents. This "Road Map" document gives an overview of how these technologies (and their documents) fit together to facilitate Service Discovery across a broad range of operating environments, from small scale zero-configuration networks to large scale administered networks, from local area to wide area, and from low-speed wireless links in the kb/s range to high-speed wired links operating at multiple Gb/s.

Not all of the available components are necessary or appropriate in all scenarios. One goal of this "Road Map" document is to provide guidance about which components to use depending on the problem being solved.

2. Service Type Namespace

The single most important concept in Service Discovery is the namespace specifying how different service types are identified. This is how a client communicates what it needs, and how a server communicates what it offers. For a client to discover a server, client and server need to use the same namespace of service types, otherwise they may actually speak the same application protocol over the air or on the wire, and may in fact be completely compatible, and yet may be unable to detect this because they are using different names to refer to the same actual service. Hence, having a consistent namespace for referring to service types is vital.

IANA manages the registry of Service Types [RFC6335][SN]. This registry of Service Types can (and should) be used in any Service Discovery protocol as the vocabulary for describing *all* IP-based services, not only DNS-Based Service Discovery [RFC6763].

In this document we focus on the use of the IANA Service Type Registry [SN] in conjunction with DNS-Based Service Discovery, though that should not be taken in any way to imply any criticism of other Service Discovery protocols sharing the same namespace of service types. In different circumstances different Service Discovery protocols are appropriate.

For example, for Service Discovery of services potentially available via a Wi-Fi access point, prior to association with that Wi-Fi access point, when no IP link has yet been established, a Service Discovery protocol may use raw 802.11 frames, not necessarily IP, UDP, or DNS-formatted messages. For Service Discovery using peer-to-peer Wi-Fi technologies, without any Wi-Fi access point at all, it may also be preferable to use raw 802.11 frames instead of IP, UDP, or DNS-formatted messages. Service Discovery using IEEE 802.15.4 radios may use yet another over-the-air protocol. What is important is that they all share the same vocabulary to describe all IP-based services, so that client and server software, using agnostic APIs to consume and offer services on the network, has a common language to identify those services, independent of the medium or the particular Service Discovery protocol in use on that medium. Just as TCP/IP runs on many different link layers, and the concept of using an IP address to identify a particular peer is consistent across many different link layers, the concept of using a name from the IANA Service Type Registry to identify a particular service type also needs to be consistent across all IP-supporting link layers.

3. Service Discovery Operational Model

The three principal Service Discovery operations utilizing service types in the IANA Service Type Registry [SN] are:

1. Offer
2. Discover/Enumerate
3. Use

The first step, "Offer", is when a server is offering a service using some application-layer protocol on a listening TCP or UDP (or other transport protocol) port, and wishes to make that known to other devices.

The second step, "Discover", sometimes called, "Enumerate", is when a client device wishes to perform some action, but does not yet know which particular service instance will be used to perform that action. For example, when a user taps the "AirPrint" button on an iPhone, the iPhone knows that the user wishes to print, but not which particular printer to use. The desired *function* is known (IPP printing), but not the particular instance. In this case, the client device needs to enumerate the list of available service instances that are able to perform the desired task. In most cases this list of service instances is presented to a human user to choose from; in some cases it is software that examines the list of available service instances and determines the best one to use.

The third step, "Use", is when particular service instance has been selected, and the client wants to make use of that service instance, by opening a TCP connection to it or by sending UDP datagrams.

The second and third steps are intentionally separate. In the second step, a limited amount of information (typically just the name) is requested about a large number of service instances. In the third step more detailed information (e.g, target host IP address, port number, etc.) is requested about one specific service instance. Requesting all the detailed information about all available service instances would be inefficient and wasteful on the network. If the information about services on the network is imagined as a table, then the second step is requesting just one column from that table (the names) and the third step is requesting just one row from that table (the information pertaining to just one named service instance).

To give an example, clicking the "+" button in the printer settings on macOS is an operation performing the second step. It is requesting the names of all available printers. Once a print queue has been configured for the chosen printer, subsequent printing of

documents is an operation performing the third step. It only needs to request information about the specific printer in question. It is not necessary to repeatedly discover the list of every printer on the network if the device already knows which one it intends to use.

DNS-Based Service Discovery [RFC6763] implements these three principal Service Discovery operations using DNS records and queries, either using Multicast DNS [RFC6762] (for queries limited to the local link) or conventional unicast DNS [RFC1034] [RFC1035] (for queries beyond the local link).

Other Service Discovery protocol achieve the same semantics using different packet formats and mechanisms.

One incidental benefit of using DNS as the foundation layer is that Multicast DNS and conventional unicast DNS are also used provide name resolution (mapping host names to IP addresses) so there is some efficiency and code reuse in using the same underlying protocol for both naming and service discovery.

A final requirement is that the Service Discovery protocol perform not only discovery at a single moment in time, but also ongoing change notification (sometimes called "Publish & Subscribe"). Without support for ongoing change notification, clients would be forced to resort to polling to keep data up to date, which is inefficient and wasteful on the network.

Multicast DNS [RFC6762] implicitly includes change notification by virtue of announcing record changes via IP Multicast, which allows these changes to be seen by all peers on the same link (broadcast domain).

Conventional unicast DNS [RFC1034] [RFC1035] has historically not had broad support for change notification. This capability is added via the new mechanism for DNS Push Notifications [Push].

When using DNS-Based Service Discovery [RFC6763] there are two aspects to consider: firstly how the clients choose what DNS names to query, and what query mechanisms to use, and secondly how the relevant information got into the DNS namespace in the first place, so as to be available when clients query for it.

The available namespaces are discussed below in Section 4. Client operation is discussed in Section 5 and server operation is discussed in Section 6.

4. Service Discovery Namespace

When used with Multicast DNS [RFC6762] queries are automatically performed in the ".local" parent domain.

When used with conventional unicast DNS [RFC1034] [RFC1035] some other domain must be used.

For individuals and organizations with a globally-unique domain name registered to them, their globally-unique domain name, or a subdomain of it, can be used for service discovery.

However, it would be convenient for capable service discovery to be available even to people who haven’t taken the step of registering and paying for a globally-unique domain name. For these people it would be useful if devices arrived preconfigured with some suitable factory-default service discovery domain, such as "services.homenet" [I-D.ietf-homenet-dot]. Services published in this factory-default service discovery domain would not be globally unique or globally resolvable, but they could have scope larger than the single link provided by Multicast DNS.

5. Client Configuration and Operation

When using DNS-Based Service Discovery [RFC6763], clients have to choose what DNS names to query.

When used with Multicast DNS [RFC6762] queries are automatically performed in the ".local" parent domain.

For discovery beyond the local link, a unicast DNS domain must be used. This unicast DNS domain can be configured manually by the user, or it can be learned dynamically from the network (as has been done for many years at IETF meetings to facilitate discovery of the IETF Terminal Room printer from outside the IETF Terminal Room network). In the DNS-SD specification [RFC6763] section 11, "Discovery of Browsing and Registration Domains (Domain Enumeration)", describes how a client device learns one or more recommended service discovery domains from the network, using the special "lb._dns-sd._udp" query.

Given the service type that the user or client device is seeking (see Section 2) and one or more service discovery domains to look in, the client then sends its DNS queries, and processes the responses.

For some uses one-shot conventional DNS queries and responses are perfectly adequate, but for service discovery, where a list may be displayed on a screen for a user to see, it is desirable to keep that list up to date without the user having to repeatedly tap a "refresh" button, and without the software repeatedly polling the network on the user’s behalf.

And early solution to provide asynchronous change notifications for unicast DNS was the UDP-based protocol DNS Long-Lived Queries [DNS-LLQ]. This was used, among other things, by Apple’s Back to My Mac Service [RFC6281] introduced in Mac OS X 10.5 Leopard in 2007.

Recent experience has shown that an asynchronous change notification protocol built on TCP would be preferable, so the IETF is now developing DNS Push Notifications [Push].

Because DNS Push Notifications is built on top of a DNS TCP connection, rather than inventing its own session signaling mechanisms, DNS Push Notifications adopts the conventions specified by DNS Session Signaling [S-Sig].

6. Server Configuration and Operation

Section 5 above describes how clients perform their queries. The related question is how the relevant information got into the DNS namespace in the first place, so as to be available when clients query for it.

One way that relevant service discovery information can get into the DNS namespace is simply via manual configuration, creating the necessary PTR, SRV and TXT records [RFC6763], and indeed this is how the IETF Terminal Room printer has been advertised to IETF meeting attendees for many years. While this is easy for the experienced network operators at the IETF, it can be onerous to others less familiar with how to set up DNS-SD records.

Hence it would be convenient to automate this process of populating the DNS namespace with relevant service discovery information. Two efforts are underway to address this need, the Service Discovery Proxy [DisProx] and the Service Registration Protocol [RegProt].

The first effort is the Service Discovery Proxy [DisProx]. This technology is designed to work with today's devices that advertise services using Multicast DNS only (such as almost all network printers sold in the last decade). A Service Discovery Proxy is a device colocated on the same link as the devices we wish to be able to discover from afar. A remote client sends unicast queries to the Discovery Proxy, which performs local Multicast DNS queries on behalf of the remote client, and then sends back the answers it discovers.

Because the time it takes to receive Multicast DNS responses is uncertain, this mechanism benefits from being able to deliver asynchronous change notifications as new answers come in, using DNS Long-Lived Queries [DNS-LLQ] or the newer DNS Push Notifications [Push] on top of DNS Session Signaling [S-Sig].

As an alternative to having to be physically connected to the desired network link, a Service Discovery Proxy [DisProx] can use a Multicast DNS Discovery Relay [Relay] to give it a 'virtual' presence on a remote link. Indeed, when using Discovery Relays, a single Discovery Proxy can have a 'virtual' presence on hundreds of remote links. A single Discovery Proxy in the data center can serve the needs of an entire enterprise. This is modeled after the DHCP protocol. In simple residential scenarios the DHCP server resides on the local link. In complex enterprise networks, a single DHCP server resides in the data center, using simple lightweight BOOTP relay agents colocated with the routers on each physical link.

Finally, when clients are making TCP connections to multiple Service Discovery Proxies at the same time, this can be burdensome for the clients (which may be mobile and battery powered) and for the the Service Discovery Proxies (which may have to serve hundreds of clients). This situation is remedied by use of a Service Discovery Broker [Broker]. A Service Discovery Broker is an intermediary between client and server. A client can issue a single query to the Service Discovery Broker and have the Service Discovery Broker do the hard work of issuing multiple queries on behalf of the client. And a Service Discovery Broker can shield a Service Discovery Proxy from excessive load by colapsing multiple duplicate queries from different client down to a single query to the Service Discovery Proxy.

The second effort in this space, tacking the chalenge of automating the process of populating the DNS namespace with relevant service discovery information, is the Service Registration Protocol [RegProt]. This technology is designed to work with future devices that explicitly cooperate with the network to advertise their services.

The Service Registration Protocol is effectively DNS Update, with some minor additions.

One addition is the introduction of a lifetime on DNS Updates, using the the Dynamic DNS Update Lease EDNS(0) option [DNS-UL].

The second addition is the introduction of information that tells the Service Registration server that the device will be going to sleep to save power, combined with information specifying how to wake it up again on demand, using the EDNS(0) OWNER Option [Owner].

The use of an explicit Service Registration Protocol is beneficial in networks where multicast is expensive, inefficient, or outright blocked, such as many Wi-Fi networks. An explicit Service Registration Protocol is also beneficial in networks where multicast and broadcast are supported poorly, if at all, such as mesh networks like those using IEEE 802.15.4.

The use of power management information in the Service Registration messages allows devices to sleep to save power, which is especially beneficial for battery-powered devices in the home.

7. Informative References

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <http://www.rfc-editor.org/info/rfc1034>.

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

[RFC6281] Cheshire, S., Zhu, Z., Wakikawa, R., and L. Zhang, "Understanding Apple's Back to My Mac (BTMM) Service", RFC 6281, DOI 10.17487/RFC6281, June 2011, <http://www.rfc-editor.org/info/rfc6281>.

[RFC6760] Cheshire, S. and M. Krochmal, "Requirements for a Protocol to Replace the AppleTalk Name Binding Protocol (NBP)", RFC 6760, DOI 10.17487/RFC6760, February 2013, <http://www.rfc-editor.org/info/rfc6760>.

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, DOI 10.17487/RFC6762, February 2013, <http://www.rfc-editor.org/info/rfc6762>.

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

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <http://www.rfc-editor.org/info/rfc6335>.

[I-D.ietf-homenet-dot] Pfister, P. and T. Lemon, "Special Use Domain '.home.arpa'", draft-ietf-homenet-dot-06 (work in progress), June 2017.

[DisProx] Cheshire, S., "Discovery Proxy for Multicast DNS-Based Service Discovery", draft-ietf-dnssd-hybrid-06 (work in progress), March 2017.

[Push] Pusateri, T. and S. Cheshire, "DNS Push Notifications", draft-ietf-dnssd-push-12 (work in progress), July 2017.

[S-Sig] Bellis, R., Cheshire, S., Dickinson, J., Dickinson, S., Mankin, A., and T. Pusateri, "DNS Session Signaling", draft-ietf-dnsop-session-signal-03 (work in progress), July 2017.

[DNS-UL] Sekar, K., "Dynamic DNS Update Leases", draft-sekar-dns-ul-01 (work in progress), August 2006.

[DNS-LLQ] Sekar, K., "DNS Long-Lived Queries", draft-sekar-dns-llq-01 (work in progress), August 2006.

[Roadmap] Cheshire, S., "Service Discovery Road Map", draft-cheshire-dnssd-roadmap-00 (work in progress), July 2017.

[Owner] Cheshire, S. and M. Krochmal, "EDNS0 OWNER Option", draft-cheshire-edns0-owner-option-01 (work in progress), July 2017.

[RegProt] Cheshire, S. and T. Lemon, "Service Registration Protocol for DNS-Based Service Discovery", draft-sctl-service-registration-00 (work in progress), July 2017.

[Relay] Cheshire, S. and T. Lemon, "Multicast DNS Discovery Relay", draft-sctl-dnssd-mdns-relay-00 (work in progress), July 2017.

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

[SN] "Service Name and Transport Protocol Port Number Registry", <http://www.iana.org/assignments/service-names-port-numbers/>.

[ZC] Cheshire, S. and D. Steinberg, "Zero Configuration Networking: The Definitive Guide", O'Reilly Media, Inc. , ISBN 0-596-10100-7, December 2005.

Author's Address

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

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