Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: January 3, 2019

S. Cheshire
Apple Inc.
T. Lemon
Nibbhaya Consulting

July 2, 2018

Service Registration Protocol for DNS-Based Service Discovery
draft-sctl-service-registration-01

Abstract

The DNS-SD Service Registration Protocol uses the standard DNS Update mechanism to enable DNS-Based Service Discovery using only unicast packets. This eliminates the dependency on Multicast DNS as the foundation layer, which greatly improves scalability and improves performance on networks where multicast service is not an optimal choice, particularly 802.11 (WiFi) and 802.15 (IoT) networks. DNS-SD Service registration uses public keys and SIG(0) to allow services to defend their registrations against attack.

1. Introduction

DNS-Based Service Discovery [RFC6763] is a component of Zero Configuration Networking [RFC6760] [ZC] [I-D.cheshire-dnssd-roadmap].

DNS-Based Service Discovery (DNS-SD) allows services to advertise the fact that they provide service, and to provide the information required to access that service. Clients can then discover the set of services of a particular type that are available. They can then select a service from among those that are available and obtain the information required to use it.

The DNS-SD Service Registration protocol, described in this document, provides a reasonably secure mechanism for publishing this information: what services are offered, and how to use them. Once published, these services can be readily discovered by clients using standard DNS lookups.

In the DNS-Based Service Discovery specification [RFC6763] Section 10 "Populating the DNS with Information" briefly discusses ways that services can publish their information in the DNS namespace. In the case of Multicast DNS [RFC6762], allows clients to directly query services on the local link for names in the ".local" namespace.

RFC6763 also allows clients to discover services using the DNS protocol [RFC1035]; this is done either by having a system administrator manually configure service information in the DNS, or by using a Discovery Proxy [I-D.ietf-dnssd-hybrid], which performs mDNS queries on behalf clients issuing queries using DNS. This eliminats the "link local" limitation of mDNS, but provides no additional security, and still relies on multicast.

Manual configuration of DNS servers is costly and failure-prone, and requires a knowledgable network administrator. Consequently, although all DNS-SD implementations of which we are aware support it, it is much less frequently used than mDNS. This document describes a solution: a way to provide DNS-SD using DNS that can be as automatic as multicast DNS, but with better performance, scalability and security.

2. Service Registration Protocol

Services using the DNS-SD Service Registration Protocol use DNS Update [RFC2136] [RFC3007] to publish service information in the DNS. We will discuss several parts to this process: how to know what to publish, how to know where to publish it (under what name), how to publish it, how to secure its publication, and how to maintain the information once published.

2.1. What to publish

RFC 6763 describes the details of what is to be published. In general terms, a service will have a name under which it offers services ([RFC6763] section 4.1.1) and one or more service names under which that instance name appears ([RFC6763] section 4.1.2). The full details of how this works are described in section 4 of that document in its entirety. A service publishes its contact information using an SRV record on the Service Instance Name. It can also publish TXT records with additional information about the service; this is discussed in section 6 of RFC 6763.

RFC 6763 is the definitive source for information about what to publish; the reason for mentioning it here is that the reader may prefer to have an overview of the whole service registration process before digging into the details. Also, the "Service Instance Name" is an important aspect of first-come, first-serve naming, which we describe later on in this document.

2.2. Where to publish it

Multicast DNS uses a single namespace, ".local", which is valid on the local link. This convenience is not available for DNS-SD using the DNS protocol: the portion of the DNS namespace in which services on the local network are to be published must be discovered by the service before it can register itself.

Names published using DNS-SD service registration will be published under some name other than .local. However, the process of discovering what that name is is complicated, and for any given network it should always be the case that there will be just one namespace in which registered names will be published. Rather than requiring the service to discover this name before issuing a registration, the service SHOULD simply use the name ".local." The DNS server that receives the registration request will rewrite all instances of the terminal label ".local" to use the local registration domain name. The response to the DNS Update being used to register the service will contain the rewritten names, instead of ".local". Subsequent updates should still use the ".local" domain

and not the registration domain, since the registration domain may change over time or when the service is physically moved to a new network.

2.3. How to publish it

DNS Updates are very flexible. As a consequence, it is possible to do the entire registration in a single DNS message. The update consists of two elements. The first updates the Service Name by adding a PTR record pointing to the Service Instance Name. The second updates the Service Instance Name. The second creates or updates the Service Instance Name update adds an SRV record and a KEY record, and optionally adds a TXT record with extra information about the service. The contents of the KEY record are described in the section on First-Come First-Served Naming (Section 2.4.1). The update is signed using the private key that corresponds to the public key in the KEY record, using the SIG(0) protocol [RFC2931].

The update may be rejected. If the chosen service instance name is not permitted, or is already taken, the update will be returned with the error code YXDOMAIN. In this case, the service will need to choose a new instance name and try again.

2.4. How to secure it

Traditional DNS update is secured using the TSIG protocol, which uses a secret key shared between the client (which issues the update) and the server (which authenticates it). This model does not work for automatic service registration.

The goal of securing the DNS-SD Registration Protocol is to provide the best possible security given the constraint that service registration has to be automatic. It is possible to layer more operational security on top of what we describe here, but what we describe here improves upon the security of mDNS. The goal is not to provide the level of security of a network managed by a skilled operator.

2.4.1. First-Come First-Served Naming

First-Come First-Serve naming provides a limited degree of security: a service that registers its service using DNS-SD Registration protocol is given ownership of a name for an extended period of time based on the key used to authenticate the DNS Update. As long as the registration service remembers the Service Instance Name and the key used to register that Service Instance Name, no other service can add or update the information associated with that Service Instance Name.

2.4.1.1. Service Behavior

The service generates a public/private key pair. This key pair MUST be stored in stable storage; if there is no writable stable storage on the client, the client MUST be pre-configured with a public/private key pair that can be used.

When sending DNS updates, the service includes a KEY record containing the public portion of the key, which is stored as an RRset under the Service Instance Name. It is permissible for a device that offers more than one service under more than one Service Instance Name to use the same KEY on each such name.

The update is signed using SIG(0), using the private key that corresponds to the public key in the KEY record.

The lifetime of the DNS-SD PTR, SRV and TXT records [RFC6763] is typically set to two hours. This means that if a device is disconnected from the network, it does not appear in the user interfaces of devices looking for services of that type for too long.

However, the lifetime of its DNS SIG(0) public key should be set to a much longer time, typically 14 days. The result of this is that even though a device may be temporarily unplugged, disappearing from the network for a few days, it makes a claim on its name that lasts much longer.

This way, even if a device is unplugged from the network for a few days, and its services are not available for that time, no other rogue device can come along and immediately claim its name the moment it disappears from the network, and yet the name is eventually cleaned up and made available for re-use.

2.4.1.2. Registration Server Behavior

The Registration server checks that the DNS update contains a Service Instance Name. In principle, each name in the update should be evaluated as a candidate Service Instance Name. However, some names will obviously be Service Names, and these can be skipped when evaluating candidates. In order for a candidate to actually be a service instance name, the following conditions must be true:

o There is at least one name that turns out NOT to be a Service Instance Name for which there is a PTR RRset update that includes a record pointing to the candidate.

o The candidate includes an SRV record

o The candidate includes a KEY record

If an update does not contain a valid Service Instance Name, or if it contains an update to a PTR RRset that references a name that is not the Service Instance Name being updated, the update is rejected with the NOTAUTH RCODE.

If an update contains an SRV record that contains an IP address other than the IP address from which the update was recieved, the update is rejected with the NOTAUTH RCODE.

Once each name for which there are updates in the DNS Update has been considered as a candidate, it should be the case that only one name is actually a possible Service Instance Name. If more than one name is still a possible candidate, then the DNS Update is rejected with the FORMERR RCODE.

If there is only one candidate, then the server checks to see if that name already exists. If it does already exist, then the server checks to see if the KEY record on the name is the same as the one in the update for that name. If it is not, then the DNS Update is rejected with the YXDOMAIN RCODE.

Otherwise, the server validates the update using SIG(0). If the validation fails, the update is rejected with NOTAUTH. Otherwise, the update is evaluated according to the rules described in RFC2136 for processing DNS updates, and whatever the correct result is is returned.

The server MAY apply additional criteria when accepting updates. In some networks, it may be possible to do out-of-band registration of keys, and only accept updates from pre-registered keys. In this case, an update for a key that has not been registered should be rejected using NOTAUTH.

There are at least two benefits to doing this rather than simply using normal SIG(0) DNS updates. First, the same registration protocol can be used in both cases, so both use cases can be addressed by the same implementation. Second, the registration protocol includes maintenance functionality not present with normal DNS updates.

The server may also have a dictionary of names or name patterns that are not permitted. If such a list is used, updates for Service Instance Names that match entries in the dictionary are rejected with YXDOMAIN.

2.5. Maintenance

2.5.1. Cleaning up stale data

Because the DNS-SD registration protocol is automatic, and not managed by humans, some additional bookkeeping is required. When an update is constructed by the client, it MUST include include an EDNS(0) Update Lease option and an EDNS(0) Instance Lease option.

These leases are promises, similar to DHCP leases [RFC2131], from the client that it will send a new update for the service registration before the lease time expires. The Update Lease time is chosen to represent the time after the update during which the registered records other than the KEY record should be assumed to be valid. The Instance Lease time represents the time after the update during which the KEY record should be assumed to be valid.

The reasoning behind the different lease times is discussed in the section on first-come, first-served naming Section 2.4.1. DNS-SD Registration Protocol servers may be configured with limits for these values. A default limit of two hours for the Update Lease and 30 days for the Instance Lease are currently thought to be good choices. Clients that are going to continue to use names on which they hold leases should update well before the lease ends, in case the registration service is unavailable or under heavy load.

Clients should assume that each lease ends N seconds after the update was first transmitted, where N is the number included in the option. Servers should assume that each lease ends N seconds after the update that was successfully processed was received. Because the server will always receive the update after the client sent it, this avoids the possibility of misunderstandings.

DNS-SD Registration Protocol servers SHOULD reject updates that do not include a DNS update lease time. Dual-use servers may accept updates that don't include leases, but SHOULD differentiate between DNS-SD registration protocol updates and other updates, and SHOULD reject updates that are known to be DNS-SD registration protocol updates if they do not include leases.

2.5.2. Sleep Proxy

Another use of Service Registration Protocol is for devices that sleep to reduce power consumption.

In this case, in addition to the DNS Update Lease option [I-D.sekar-dns-ul] described above, the device includes an EDNS(0) OWNER Option [I-D.cheshire-edns0-owner-option].

The DNS Update Lease option constitutes a promise by the device that it will wake up before this time elapses, to renew its records and thereby demonstrate that it is still attached to the network. If it fails to renew the records by this time, that indicates that it is no longer attached to the network, and its records should be deleted.

The EDNS(0) OWNER Option indicates that the device will be asleep, and will not be receptive to normal network traffic. When a DNS server receives a DNS Update with an EDNS(0) OWNER Option, that signifies that the DNS server should act as a proxy for any IPv4 or IPv6 address records in the DNS Update message. This means that the DNS server should send ARP or ND messages claiming ownership of the IPv4 and/or IPv6 addresses in the records in question. In addition, the DNS server should answer future ARP or ND requests for those IPv4 and/or IPv6 addresses, claiming ownership of them. When the DNS server receives a TCP SYN or UDP packet addressed to one of the IPv4 or IPv6 addresses for which it proxying, it should then wake up the sleeping device using the information in the EDNS(0) OWNER Option. At present version 0 of the OWNER Option specifies the "Wake-on-LAN Magic Packet" that needs to be sent; future versions could be extended to specify other wakeup mechanisms.

3. Security Considerations

DNS-SD Service Registration Protocol updates have no authorization semantics other than first-come, first-served. This means that if an attacker from outside of the administrative domain of the server knows the server's IP address, it can in principle send updates to the server that will be processed successfully. Servers should therefore be configured to reject updates from source addresses outside of the administrative domain of the server.

Note that these rules only apply to the validation of DNS-SD registration protocol updates. A server that accepts updates from DNS-SD registration protocol clients may also accept other DNS updates, and those DNS updates may be validated using different rules. However, in the case of a DNS service that accepts automatic updates, the intersection of the DNS-SD service registration update rules and whatever other update rules are present must be considered very carefully.

4. Privacy Considerations

5. References

5.1. Normative 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>.

5.2. Informative 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>.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <https://www.rfc-editor.org/info/rfc2131>.

[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>.

[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>.

[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, <https://www.rfc-editor.org/info/rfc6760>.

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

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

[I-D.cheshire-dnssd-roadmap]
Cheshire, S., "Service Discovery Road Map", draft-cheshire-dnssd-roadmap-01 (work in progress), March 2018.

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

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

Authors' Addresses

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

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

Ted Lemon
Nibbhaya Consulting
P.O. Box 958
Brattleboro, Vermont 05302
United States of America

Email: mellon@fugue.com