Network Working Group   C. Huitema
Internet-Draft   Private Octopus Inc.
Intended status: Informational   D. Kaiser
Expires: September 13, 2020   University of Luxembourg
                                                         March 12, 2020

DNS-SD Privacy and Security Requirements
draft-ietf-dnssd-prireq-08

Abstract

   DNS-SD (DNS Service Discovery) normally discloses information about
   devices offering and requesting services. This information includes
   host names, network parameters, and possibly a further description of
   the corresponding service instance. Especially when mobile devices
   engage in DNS Service Discovery at a public hotspot, serious privacy
   problems arise. We analyze the requirements of a privacy-respecting
   discovery service.

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 13, 2020.

Copyright Notice

Table of Contents

[1](#). Introduction

DNS Service Discovery (DNS-SD) [RFC6763] over Multicast DNS (mDNS)
[RFC6762] enables zero-configuration service discovery in local
networks. It is very convenient for users, but it requires the
public exposure of the offering and requesting identities along with
information about the offered and requested services. Parts of the
published information can seriously breach the user's privacy. These
privacy issues and potential solutions are discussed in [KW14a],
[KW14b] and [K17]. While the multicast nature of mDNS makes these
risks obvious, most risks derive from the observability of
transactions. These risks also need to be mitigated when using
server-based variants of DNS-SD.

There are cases when nodes connected to a network want to provide or
consume services without exposing their identities to the other
parties connected to the same network. Consider, for example, a
traveler wanting to upload pictures from a phone to a laptop when
both are connected to the Wi-Fi network of an Internet cafe, or two
travelers who want to share files between their laptops when waiting
for their plane in an airport lounge.

We expect that these exchanges will start with a discovery procedure
using DNS-SD over mDNS. One of the devices will publish the
availability of a service, such as a picture library or a file store
in our examples. The user of the other device will discover this
service, and then connect to it.

When analyzing these scenarios in Section 3.1, we find that the DNS-
SD messages leak identifying information such as the service instance
name, the host name, or service properties. We use the following
definitions:

Identity In this document, the term "identity" refers to the
   identity of the entity (legal person) operating a device.

Disclosing an Identity In this document "disclosing an identity"
   means showing the identity of operating entities to devices
   external to the discovery process; e.g., devices on the same
   network link that are listening to the network traffic but are not
   actually involved in the discovery process. This document focuses
   on identity disclosure by data conveyed via messages on the
   service discovery protocol layer. Still, identity leaks on deeper
   layers, e.g., the IP layer, are mentioned.

Disclosing Information In this document "disclosing information" is
   also focused on disclosure of data conveyed via messages on the

service discovery protocol layer, such as generic non-identity but
still potentially sensitive data.

2. Threat Model

This document considers the following attacker types sorted by
increasing power. All these attackers can either be passive (they
just listen to network traffic they have access to) or active (they
additionally can craft and send malicious packets).

external An external attacker is not on the same network link as
   victim devices engaging in service discovery; thus, the external
   attacker is in a different multicast domain.

on-link An on-link attacker is on the same network link as victim
   devices engaging in service discovery; thus, the on-link attacker
   is in the same multicast domain. This attacker can also mount all
   attacks an external attacker can mount.

MITM A Man in the Middle (MITM) attacker either controls (parts of)
   a network link or can trick two parties to send traffic via the
   attacker; thus, the MITM attacker has access to unicast traffic
   between devices engaging in service discovery. This attacker can
   also mount all attacks an on-link attacker can mount.

3. Threat Analysis

In this section we analyse how the attackers described in the
previous section might threaten the privacy of entities operating
devices engaging in service discovery. We focus on attacks
leveraging data transmitted in service discovery protocol messages.

3.1. Service Discovery Scenarios

In this section, we review common service discovery scenarios and
discuss privacy threats and their privacy requirements. In all three
of these common scenarios the attacker is of the type passive on-
link.

3.1.1. Private Client and Public Server

Perhaps the simplest private discovery scenario involves a single
client connecting to a public server through a public network. A
common example would be a traveler using a publicly available printer
in a business center, in an hotel, or at an airport.

```
                                     ( Taking notes:
                                     ( David is printing
                                     ( a document
                                      ~~~~~~~~~~~
                                                 o
                                             o     ___
       ___                                     _|___|_
      /   \                                    |* *|
      |   |   client              server       \_/
       \_/         __                            |
        |      / /   Discovery   +----------+     |
       /|\    /_/  <-----------> |  +----+  |    /|\
      / | \__/                   +--|    |--+   / | \
     /  |                           |___/      /  |  \
    /   |                                     /   |   \
    / \                                       / \
   /   \                                     /   \
  /     \                                   /     \
 /       \                                 /       \
/         \                               /         \

      David                                    adversary
```
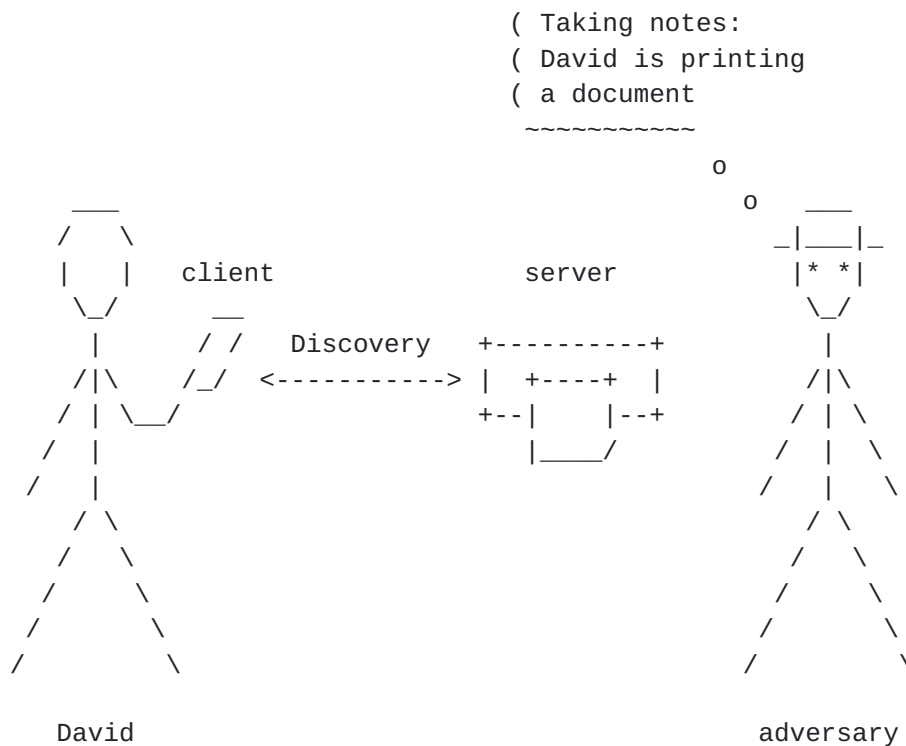
   In that scenario, the server is public and wants to be discovered,
   but the client is private. The adversary will be listening to the
   network traffic, trying to identify the visitors' devices and their
   activity. Identifying devices leads to identifying people, either
   for surveillance of these individuals in the physical world or as a
   preliminary step for a targeted cyber attack.

   The requirement in that scenario is that the discovery activity
   should not disclose the identity of the client.

3.1.2. Private Client and Private Server

   The second private discovery scenario involves a private client
   connecting to a private server. A common example would be two people
   engaging in a collaborative application in a public place, such as an
   airport's lounge.

```
                                                  ( Taking notes:
                                                  ( David is meeting
                                                  ( with Stuart
                                                   ~~~~~~~~~~~
                                                                o
         ___                          ___          o   ___
        /   \                        /   \          _|___|_
        |   |   server      client   |   |         |* *|
         \_/          __        __     \_/           \_/
          |      / /   Discovery  \ \    |            |
         /|\    /_/  <----------->  \_\   /|\          /|\
        / | \__/                  \__/ | \        / | \
       /  |                           |  \      /  |  \
      /   |                           |   \    /   |   \
       / \                           / \        / \
      /   \                         /   \      /   \
     /     \                       /     \    /     \
    /       \                     /       \  /       \
   /         \                   /         \/         \

        David                         Stuart      Adversary
```
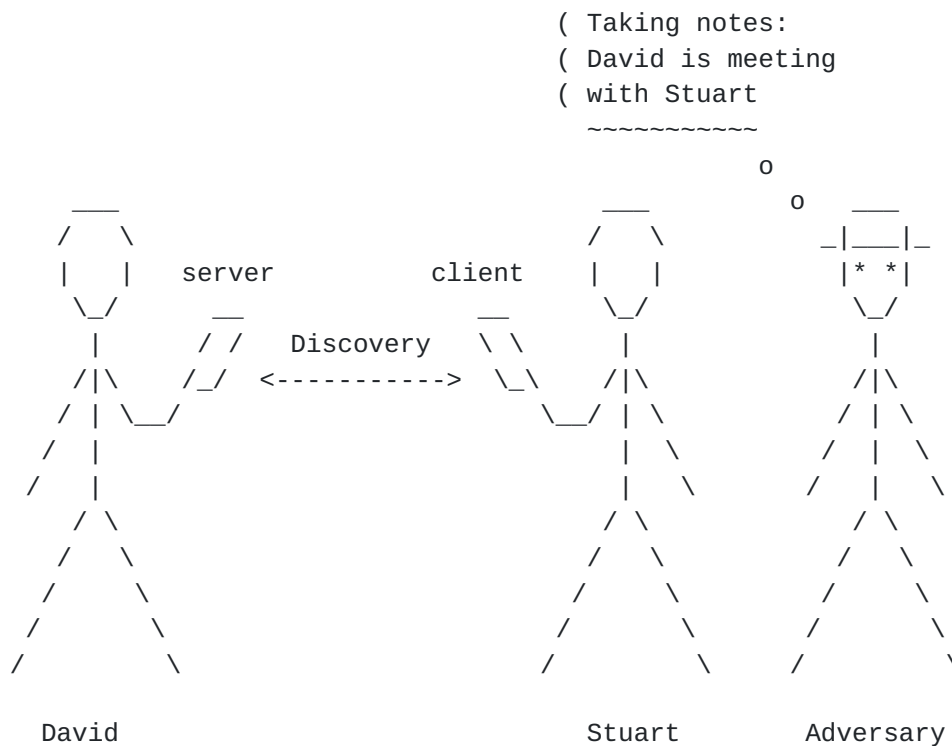
   In that scenario, the collaborative application on one of the devices
   will act as a server, and the application on the other device will
   act as a client. The server wants to be discovered by the client,
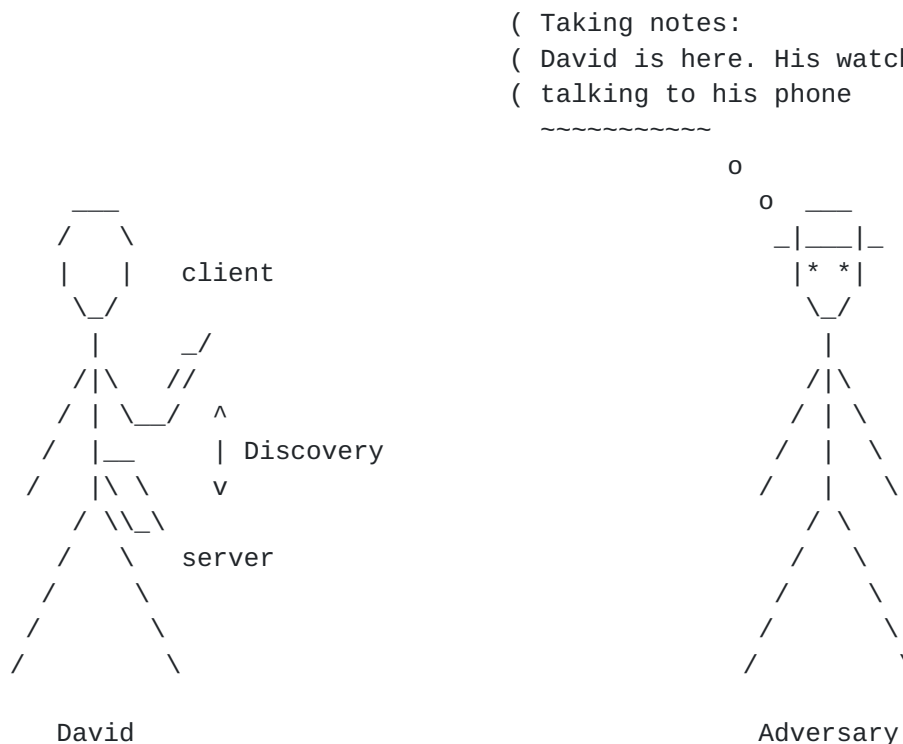   but has no desire to be discovered by anyone else. The adversary
   will be listening to network traffic, attempting to discover the
   identity of devices as in the first scenario, and also attempting to
   discover the patterns of traffic, as these patterns reveal the
   business and social interactions between the owners of the devices.

   The requirement in that scenario is that the discovery activity
   should not disclose the identity of either the client or the server,
   nor reveal the business and social interactions between the owners of
   the devices.

3.1.3. Wearable Client and Server

   The third private discovery scenario involves wearable devices. A
   typical example would be the watch on someone's wrist connecting to
   the phone in their pocket.

```
                                    ( Taking notes:
                                    ( David is here. His watch is
                                    ( talking to his phone
                                      ~~~~~~~~~~~
                                                     o
           ___                          o    ___
          /   \                            _|___|_
          |   |    client                  |* *|
           \_/                              \_/
            |      _/                        |
           /|\    //                        /|\
          / | \__/  ^                      / | \
         /  |__      | Discovery          /  |  \
        /   |\ \     v                   /   |   \
         / \\_\                         / \
          /   \    server              /   \
         /     \                      /     \
        /       \                    /       \
       /         \                  /         \

          David                          Adversary
```

   This third scenario is in many ways similar to the second scenario.
   It involves two devices, one acting as server and the other acting as
   client, and it leads to the same requirement of the discovery traffic
   not disclosing the identity of either the client or the server. The
   main difference is that the devices are managed by a single owner,
   which can lead to different methods for establishing secure relations
   between the devices. There is also an added emphasis on hiding the
   type of devices that the person wears.

   In addition to tracking the identity of the owner of the devices, the
   adversary is interested in the characteristics of the devices, such
   as type, brand, and model. Identifying the type of device can lead
   to further attacks, from theft to device-specific hacking. The
   combination of devices worn by the same person will also provide a
   "fingerprint" of the person, risking identification.

   This scenario also represents the general case of bringing private
   IoT devices into public places. A wearable IoT device might act as a
   DNS-SD/mDNS client which allows attackers to infer information about
   devices' owners. While the attacker might be a person as in the
   example figure, this could also be abused for large scale data
   collection installing stationary IoT-device-tracking servers in
   frequented public places.

   The issues described in Section 3.1.1 such as identifying people or
   using the information for targeted attacks apply here too.

3.2. **DNS-SD Privacy Considerations**

   While the discovery process illustrated in the scenarios in
   Section 3.1 most likely would be based on [RFC6762] as a means for
   making service information available, this document considers all
   kinds of means for making DNS-SD resource records available. These
   means comprise but are not limited to mDNS [RFC6762], DNS servers
   ([RFC1033] [RFC1034], [RFC1035]), using SRP [I-D.ietf-dnssd-srp], and
   multi-link [RFC7558] networks.

   The discovery scenarios in Section 3.1 illustrate three separate
   abstract privacy requirements that vary based on the use case. These
   are not limited to mDNS.

   1. Client identity privacy: Client identities are not leaked during
       service discovery or use.

   2. Multi-entity, mutual client and server identity privacy: Neither
       client nor server identities are leaked during service discovery
       or use.

   3. Single-entity, mutual client and server identity privacy:
       Identities of clients and servers owned and managed by the same
       legal person are not leaked during service discovery or use.

   In this section, we describe aspects of DNS-SD that make these
   requirements difficult to achieve in practice. While it is intended
   to be thorough, it is not possible to be exhaustive.

   Client identity privacy, if not addressed properly, can be thwarted
   by a passive attacker (see Section 2). The type of passive attacker
   necessary depends on the means of making service information
   available. Information conveyed via multicast messages can be
   obtained by an on-link attacker. Unicast messages are easy to access
   if the transmission is not encrypted, but could still be accessed by
   an attacker with access to network routers or bridges. Using multi-
   link service discovery solutions [RFC7558], external attackers have
   to be taken into consideration as well, e.g., when relaying multicast
   messages to other links.

   Server identity privacy can be thwarted by a passive attacker in the
   same way as client identity privacy. Additionally, active attackers
   querying for information have to be taken into consideration as well.
   This is mainly relevant for unicast-based discovery, where listening
   to discovery traffic requires a MITM attacker; however, an external
   active attacker might be able to learn the server identity by just
   querying for service information, e.g. via DNS.

3.2.1. Information made available via DNS-SD Resource Records

DNS-Based Service Discovery (DNS-SD) is defined in [RFC6763]. It
allows nodes to publish the availability of an instance of a service
by inserting specific records in the DNS ([RFC1033], [RFC1034],
[RFC1035]) or by publishing these records locally using multicast DNS
(mDNS) [RFC6762]. Available services are described using three types
of records:

PTR Record: Associates a service type in the domain with an
   "instance" name of this service type.

SRV Record: Provides the node name, port number, priority and weight
   associated with the service instance, in conformance with
   [RFC2782].

TXT Record: Provides a set of attribute-value pairs describing
   specific properties of the service instance.

3.2.2. Privacy Implication of Publishing Service Instance Names

In the first phase of discovery, clients obtain all PTR records
associated with a service type in a given naming domain. Each PTR
record contains a Service Instance Name defined in Section 4 of
[RFC6763]:

   Service Instance Name = <Instance> . <Service> . <Domain>

The <Instance> portion of the Service Instance Name is meant to
convey enough information for users of discovery clients to easily
select the desired service instance. Nodes that use DNS-SD over mDNS
[RFC6762] in a mobile environment will rely on the specificity of the
instance name to identify the desired service instance. In our
example of users wanting to upload pictures to a laptop in an
Internet Cafe, the list of available service instances may look like:

Alice's Images   . _imageStore._tcp . local
Alice's Mobile Phone   . _presence._tcp   . local
Alice's Notebook   . _presence._tcp   . local
Bob's Notebook   . _presence._tcp   . local
Carol's Notebook   . _presence._tcp   . local

Alice will see the list on her phone and understand intuitively that
she should pick the first item. The discovery will "just work".
(Note that our examples of service names conform to the specification
in section 4.1 of [RFC6763], but may require some character escaping
when entered in conventional DNS software.)

However, DNS-SD/mDNS will reveal to anybody that Alice is currently
visiting the Internet Cafe. It further discloses the fact that she
uses two devices, shares an image store, and uses a chat application
supporting the _presence protocol on both of her devices. She might
currently chat with Bob or Carol, as they are also using a _presence
supporting chat application. This information is not just available
to devices actively browsing for and offering services, but to
anybody passively listening to the network traffic, i.e. a passive
on-link attacker.

There is, of course, also no authentication requirement to claim a
particular instance name, so an active attacker can provide resources
that claim to be Alice's but are not.

[3.2.3](). **Privacy Implication of Publishing Node Names**

The SRV records contain the DNS name of the node publishing the
service. Typical implementations construct this DNS name by
concatenating the "host name" of the node with the name of the local
domain. The privacy implications of this practice are reviewed in
[[RFC8117]]. Depending on naming practices, the host name is either a
strong identifier of the device, or at a minimum a partial
identifier. It enables tracking of both the device, and, by
extension, the device's owner.

[3.2.4](). **Privacy Implication of Publishing Service Attributes**

The TXT record's attribute-value pairs contain information on the
characteristics of the corresponding service instance. This in turn
reveals information about the devices that publish services. The
amount of information varies widely with the particular service and
its implementation:

o Some attributes, such as the paper size available in a printer,
   are the same on many devices, and thus only provide limited
   information to a tracker.

o Attributes that have freeform values, such as the name of a
   directory, may reveal much more information.

Combinations of individual attributes have more information power
than specific attributes, and can potentially be used for
"fingerprinting" a specific device.

Information contained in TXT records not only breaches privacy by
making devices trackable, but might directly contain private
information about the user. For instance the _presence service

reveals the "chat status" to everyone in the same network. Users
might not be aware of that.

Further, TXT records often contain version information about
services, allowing potential attackers to identify devices running
exploit-prone versions of a certain service.

[3.2.5](#). Device Fingerprinting

The combination of information published in DNS-SD has the potential
to provide a "fingerprint" of a specific device. Such information
includes:

o List of services published by the device, which can be retrieved
   because the SRV records will point to the same host name.

o Specific attributes describing these services.

o Port numbers used by the services.

o Priority and weight attributes in the SRV records.

This combination of services and attributes will often be sufficient
to identify the version of the software running on a device. If a
device publishes many services with rich sets of attributes, the
combination may be sufficient to identify the specific device and
track its owner.

An argument is sometimes made that devices providing services can be
identified by observing the local traffic, and that trying to hide
the presence of the service is futile. However, there are good
reasons for the discovery service layer to avoid unnecessary
exposure:

1. Providing privacy at the discovery layer is of the essence for
    enabling automatically configured privacy-preserving network
    applications. Application layer protocols are not forced to
    leverage the offered privacy, but if device tracking is not
    prevented at the deeper layers, including the service discovery
    layer, obfuscating a certain service's protocol at the
    application layer is futile.

2. Further, in the case of mDNS based discovery, even if the
    application layer does not protect privacy, typically services
    are provided via unicast which requires a MITM attacker, while
    identifying services based on multicast discovery messages just
    requires an on-link attacker.

The same argument can be extended to say that the pattern of services
offered by a device allows for fingerprinting the device. This may
or may not be true, since we can expect that services will be
designed or updated to avoid leaking fingerprints. In any case, the
design of the discovery service should avoid making a bad situation
worse, and should as much as possible avoid providing new
fingerprinting information.

3.2.6. Privacy Implication of Discovering Services

The consumers of services engage in discovery, and in doing so reveal
some information such as the list of services they are interested in
and the domains in which they are looking for the services. When the
clients select specific instances of services, they reveal their
preference for these instances. This can be benign if the service
type is very common, but it could be more problematic for sensitive
services, such as some private messaging services.

One way to protect clients would be to somehow encrypt the requested
service types. Of course, just as we noted in Section 3.2.5, traffic
analysis can often reveal the service.

3.3. Security Considerations

For each of the operations described above, we must also consider
security threats we are concerned about.

3.3.1. Authenticity, Integrity & Freshness

Can devices (both servers and clients) trust the information they
receive? Has it been modified in flight by an adversary? Can
devices trust the source of the information? Is the source of
information fresh, i.e., not replayed? Freshness may or may not be
required depending on whether the discovery process is meant to be
online. In some cases, publishing discovery information to a shared
directory or registry, rather than to each online recipient through a
broadcast channel, may suffice.

3.3.2. Confidentiality

Confidentiality is about restricting information access to only
authorized individuals. Ideally this should only be the appropriate
trusted parties, though it can be challenging to define who are "the
appropriate trusted parties." In some use cases, this may mean that
only mutually authenticated and trusting clients and servers can read
messages sent for one another. The process of service discovery in
particular is often used to discover new entities that the device did
not previously know about. It may be tricky to work out how a device

can have an established trust relationship with a new entity it has
never previously communicated with.

3.3.3. Resistance to Dictionary Attacks

It can be tempting to use (publicly computable) hash functions to
obscure sensitive identifiers. This transforms a sensitive unique
identifier such as an email address into a "scrambled" but still
unique identifier. Unfortunately simple solutions may be vulnerable
to offline dictionary attacks.

3.3.4. Resistance to Denial-of-Service Attacks

In any protocol where the receiver of messages has to perform
cryptographic operations on those messages, there is a risk of a
brute-force flooding attack causing the receiver to expend excessive
amounts of CPU time and, where appliciable, battery power just
processing and discarding those messages.

Also, amplification attacks have to be taken into consideration.
Messages with larger payloads should only be sent as an answer to a
query sent by a verified client.

3.3.5. Resistance to Sender Impersonation

Sender impersonation is an attack wherein messages such as service
offers are forged by entities who do not possess the corresponding
secret key material. These attacks may be used to learn the identity
of a communicating party, actively or passively.

3.3.6. Sender Deniability

Deniability of sender activity, e.g., of broadcasting a discovery
request, may be desirable or necessary in some use cases. This
property ensures that eavesdroppers cannot prove senders issued a
specific message destined for one or more peers.

3.4. Operational Considerations

3.4.1. Power Management

Many modern devices, especially battery-powered devices, use power
management techniques to conserve energy. One such technique is for
a device to transfer information about itself to a proxy, which will
act on behalf of the device for some functions, while the device
itself goes to sleep to reduce power consumption. When the proxy
determines that some action is required which only the device itself

can perform, the proxy may have some way to wake the device, as
described for example in [SLEEP-PROXY].

In many cases, the device may not trust the network proxy
sufficiently to share all its confidential key material with the
proxy. This poses challenges for combining private discovery that
relies on per-query cryptographic operations, with energy-saving
techniques that rely on having (somewhat untrusted) network proxies
answer queries on behalf of sleeping devices.

3.4.2. Protocol Efficiency

Creating a discovery protocol that has the desired security
properties may result in a design that is not efficient. To perform
the necessary operations the protocol may need to send and receive a
large number of network packets, or require an inordinate amount of
multicast transmissions. This may consume an unreasonable amount of
network capacity, particularly problematic when it is a shared
wireless spectrum. Further, it may cause an unnecessary level of
power consumption which is particularly problematic on battery
devices, and may result in the discovery process being slow.

It is a difficult challenge to design a discovery protocol that has
the property of obscuring the details of what it is doing from
unauthorized observers, while also managing to perform efficiently.

3.4.3. Secure Initialization and Trust Models

One of the challenges implicit in the preceding discussions is that
whenever we discuss "trusted entities" versus "untrusted entities",
there needs to be some way that trust is initially established, to
convert an "untrusted entity" into a "trusted entity".

The purpose of this document is not to define the specific way in
which trust can be established. Protocol designers may rely on a
number of existing technologies, including PKI, Trust On First Use
(TOFU), or using a short passphrase or PIN with cryptographic
algorithms such as Secure Remote Password (SRP) [RFC5054] or a
Password Authenticated Key Exchange like J-PAKE [RFC8236] using a
Schnorr Non-interactive Zero-Knowledge Proof [RFC8235].

Protocol designers should consider a specific usability pitfall when
trust is established immediately prior to performing discovery.
Users will have a tendency to "click OK" in order to achieve their
task. This implicit vulnerability is avoided if the trust
establishment requires more significant participation of the user,
such as entering a password or PIN.

3.4.4. External Dependencies

Trust establishment may depend on external parties. Optionally, this might involve synchronous communication. Systems which have such a dependency may be attacked by interfering with communication to external dependencies. Where possible, such dependencies should be minimized. Local trust models are best for secure initialization in the presence of active attackers.

4. Requirements for a DNS-SD Privacy Extension

Given the considerations discussed in the previous sections, we state requirements for privacy preserving DNS-SD in the following subsections.

Defining a solution according to these requirements is intended to lead to a solution that does not transmit privacy-violating DNS-SD messages and further does not open pathways to new attacks against the operation of DNS-SD.

However, while this document gives advice on which privacy protecting mechanisms should be used on deeper-layer network protocols and on how to actually connect to services in a privacy-preserving way, stating corresponding requirements is out of the scope of this document. To mitigate attacks against privacy on lower layers, both servers and clients must use privacy options available at lower layers, and for example avoid publishing static IPv4 or IPv6 addresses, or static IEEE 802 MAC addresses. For services advertised on a single network link, link local IP addresses should be used; see [RFC3927] and [RFC4291] for IPv4 and IPv6, respectively. Static servers advertising services globally via DNS can hide their IP addresses from unauthorized clients using the split mode topology shown in [I-D.ietf-tls-esni]. Hiding static MAC addresses can be achieved via MAC address randomization (see [RFC7844]).

4.1. Private Client Requirements

For all three scenarios described in Section 3.1, client privacy requires DNS-SD messages to:

1. Avoid disclosure of the client's identity, either directly or via inference, to nodes other than select servers.

2. Avoid exposure of linkable identifiers that allow tracing client devices.

3. Avoid disclosure of the client's interest in specific service instances or service types to nodes other than select servers.

When listing and resolving services via current DNS-SD deployments, clients typically disclose their interest in specific services types and specific instances of these types, respectively.

In addition to the exposure and disclosure risks noted above, protocols and implementations will have to consider fingerprinting attacks (see Section 3.2.5) that could retrieve similar information.

4.2. Private Server Requirements

Servers like the "printer" discussed in scenario 1 are public, but the servers discussed in scenarios 2 and 3 are by essence private. Server privacy requires DNS-SD messages to:

1. Avoid disclosure of the server's identity, either directly or via inference, to nodes other than authorized clients. In particular, Servers must avoid publishing static identifiers such as host names or service names. When those fields are required by the protocol, servers should publish randomized values. (See [RFC8117] for a discussion of host names.)

2. Avoid exposure of linkable identifiers that allow tracing servers.

3. Avoid disclosure to unauthorized clients of service instance names or service types of offered services.

4. Avoid disclosure to unauthorized clients of information about the services they offer.

5. Avoid disclosure of static IPv4 or IPv6 addresses.

When offering services via current DNS-SD deployments, servers typically disclose their hostnames (SRV, A/AAAA), instance names of offered services (PRT, SRV), and information about services (TXT). Heeding these requirements protects a server's privacy on the DNS-SD level.

The current DNS-SD user interfaces present the list of discovered service names to the users, and let them pick a service from the list. Using random identifiers for service names renders that UI flow unusable. Privacy-respecting discovery protocols will have to solve this issue, for example by presenting authenticated or decrypted service names instead of the randomized values.

4.3. Security and Operation

   In order to be secure and feasible, a DNS-SD privacy extension needs
   to consider security and operational requirements including:

   1. Avoiding significant CPU overhead on nodes or significantly
       higher network load. Such overhead or load would make nodes
       vulnerable to denial of service attacks. Further, it would
       increase power consumption which is damaging for IoT devices.

   2. Avoiding designs in which a small message can trigger a large
       amount of traffic towards an unverified address, as this could be
       exploited in amplification attacks.

5. IANA Considerations

   This draft does not require any IANA action.

6. Acknowledgments

   This draft incorporates many contributions from Stuart Cheshire and
   Chris Wood. Thanks to Florian Adamsky for extensive review and
   suggestions on the organization of the threat model. Thanks to Barry
   Leiba for an extensive review. Thanks to Roman Danyliw, Ben Kaduk,
   Adam Roach and Alissa Cooper for their comments during IESG review.

7. References

7.1. Normative References

   [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>.

7.2. Informative References

   [I-D.ietf-dnssd-srp]
              Cheshire, S. and T. Lemon, "Service Registration Protocol
              for DNS-Based Service Discovery", draft-ietf-dnssd-srp-02
              (work in progress), July 2019.

   [I-D.ietf-tls-esni]
              Rescorla, E., Oku, K., Sullivan, N., and C. Wood,
              "Encrypted Server Name Indication for TLS 1.3", draft-
              ietf-tls-esni-05 (work in progress), November 2019.

   [K17]   Kaiser, D., "Efficient Privacy-Preserving
              Configurationless Service Discovery Supporting Multi-Link
              Networks", 2017,
              <http://nbn-resolving.de/urn:nbn:de:bsz:352-0-422757>.

   [KW14a]   Kaiser, D. and M. Waldvogel, "Adding Privacy to Multicast
              DNS Service Discovery", DOI 10.1109/TrustCom.2014.107,
              2014, <http://ieeexplore.ieee.org/xpl/
              articleDetails.jsp?arnumber=7011331>.

   [KW14b]   Kaiser, D. and M. Waldvogel, "Efficient Privacy Preserving
              Multicast DNS Service Discovery",
              DOI 10.1109/HPCC.2014.141, 2014,
              <http://ieeexplore.ieee.org/xpl/
              articleDetails.jsp?arnumber=7056899>.

   [RFC1033] Lottor, M., "Domain Administrators Operations Guide",
              RFC 1033, DOI 10.17487/RFC1033, November 1987,
              <https://www.rfc-editor.org/info/rfc1033>.

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

   [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>.

   [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>.

   [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>.

   [RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing
              Architecture", RFC 4291, DOI 10.17487/RFC4291, February
              2006, <https://www.rfc-editor.org/info/rfc4291>.

   [RFC5054] Taylor, D., Wu, T., Mavrogiannopoulos, N., and T. Perrin,
              "Using the Secure Remote Password (SRP) Protocol for TLS
              Authentication", RFC 5054, DOI 10.17487/RFC5054, November
              2007, <https://www.rfc-editor.org/info/rfc5054>.

   [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>.

   [RFC7844] Huitema, C., Mrugalski, T., and S. Krishnan, "Anonymity
              Profiles for DHCP Clients", RFC 7844,
              DOI 10.17487/RFC7844, May 2016,
              <https://www.rfc-editor.org/info/rfc7844>.

   [RFC8117] Huitema, C., Thaler, D., and R. Winter, "Current Hostname
              Practice Considered Harmful", RFC 8117,
              DOI 10.17487/RFC8117, March 2017,
              <https://www.rfc-editor.org/info/rfc8117>.

   [RFC8235] Hao, F., Ed., "Schnorr Non-interactive Zero-Knowledge
              Proof", RFC 8235, DOI 10.17487/RFC8235, September 2017,
              <https://www.rfc-editor.org/info/rfc8235>.

   [RFC8236] Hao, F., Ed., "J-PAKE: Password-Authenticated Key Exchange
              by Juggling", RFC 8236, DOI 10.17487/RFC8236, September
              2017, <https://www.rfc-editor.org/info/rfc8236>.

   [SLEEP-PROXY]
              Cheshire, S., "Understanding Sleep Proxy Service", 2009,
              <http://stuartcheshire.org/SleepProxy/index.html>.

Authors' Addresses

   Christian Huitema
   Private Octopus Inc.
   Friday Harbor, WA 98250
   U.S.A.

   Email: huitema@huitema.net
   URI: http://privateoctopus.com/

Daniel Kaiser
University of Luxembourg
6, avenue de la Fonte
Esch-sur-Alzette 4364
Luxembourg

Email: daniel.kaiser@uni.lu
URI: https://secan-lab.uni.lu/