dnssd
Internet-Draft
Intended status: Informational
Expires: April 20, 2014

P. van der Stok
consultant
K. Lynn
Consultant

October 17, 2013

Building control requirements
draft-vanderstok-dnssd-building-requirements-00

Abstract

The draft describes an interface to the discovery of services on a bounded network segment from a building control perspective. Building control has special boundary conditions related to: (1) management of devices and services, (2) an installation involving stand-alone networks (3) (dis)connecting network (from) to Internet without renaming services and devices. Roaming devices are not considered in this version.

Note

Discussion and suggestions for improvement are requested, and should be sent to dnssd@ietf.org.

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 April 20, 2014.

Copyright Notice

Table of Contents

1. Introduction

The DNS-SD working group aims at service discovery without operator intervention on multi-link networks. The discovery of the services is restricted to a multi-link network segment of which the limits may be determined automatically.

In building control stable relations between application and server need to be created such that applications can communicate with servers during the lifetime of the installation without changes to the application code. Applications discover named service instances

on named devices connected to a network segment. At the start of its lifetime the network segment is not connected to Internet and its services. Before connection, after connection to Internet and during consecutive dis-/re-connections, the applications should be able to use the same names for devices and service instances.

At installation time the applications bind to the services that they want to use; e.g. a lighting application binds to the light sensor server. The location of the service is leading in selecting the required service instance from an offer of identical services. The supporting installation process leads to a number of requirements on the multi-link discovery. These requirements are expressed at the application level to a discovery interface which may hide the names which are actually transported over the network.

The service discovery is assumed to make use of DNS-Based service discovery [RFC6763]. This document complements the DNSSD requirements document [I-D.lynn-mdnsext-requirements].

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

This specification requires readers to be familiar with all the terms and concepts that are discussed in [RFC6763]. In addition, this specification makes use of the following terminology:

Network segment: A network of interconnected devices. The network may be composed of one link or multiple heterogeneous wired and wireless links interconnected by routers.

Stand-alone segment: a network segment that has no access or connection to Internet and its services, e.g. DNS service, DHCP service.

Connected segment: a network segment that is connected with one or more border routers to Internet.

TODO: other terms to be specified.

2. Identification of services

Devices providing a given service need to be identified and addressed, such that an application in another device can use the service instance hosted by the selected device. The physical location or other visual aspects of the device are often important for the service instance selection.

Two layers of communication can be distinguished: (1) the application communicating with the service instance, and (2) the messages exchanged between (ports, IP-addresses) over the network. Identifiers of the first layer need to be stable, while identifiers of the second layer can evolve.

Devices are connected to the network via one or more interfaces. On the network, the interfaces can be addressed by an IP address. The IP address comes with a scope. Often, the link-local IP address is constructed from the MAC address (full or short)[RFC4862], [RFC6775]. A global IP address can be constructed from the long MAC address and a prefix provided by a border router. On the other hand a global IP address can be allocated by the IP provider via DHCP. This latter IP address bears no relation at all to the MAC address. The IP address can change at any moment during the operational life of the device.

The applications make use of services. For inter-operability reasons services and their service type are standardized. The service type has the same life-time as the standard that defines it. In this way an application can find the service instance hosted by candidate devices.

Applications refer to a service instance on a given device identified with a host name. The host name, used by the client application, should remain stable during the life-time of the device and beyond. When a device fails, the application continues to use the same host name to minimize interventions on the network at device failure. The new replacing device keeps the same host name but has a new MAC address and IP address.

From the stable service type the client application selects a service instance with a host name and path. From the stable service instance name and host name the client application can find the currently valid IP address and port number.

For control purposes devices are grouped. For example a set of lights need to be turned off/on or dimmed simultaneously. A group is identified with a group name with has the same purpose and format as the host name. An IP multicast address or a set of IP unicast addresses can be associated with a group name.

The identifiers of Table 1 need to be considered when looking at the binding of a client application to a service instance:

Identifier	Lifetime determined by
Global IP address	IP provider decision
Link-local IP address	MAC address lifetime
Host name	Installation lifetime
Service instance name	Installation lifetime
Service type	service defining SDO existence
Group name	Installation lifetime

Table 1

During installation and commissioning of the devices, the host name and hosted service instance need to be related to each other and to the IP addresses and port number.

3. Network

For this specification we distinguish the logical network and the network segment.

A "logical network" is composed of "named" devices hosting "named" service-instances which provide "named" services to applications.

A network segment, defined in Section 1.1, provides IP addresses and ports.

The "logical network" is implemented with a network segment, which can be:

- A wireless mesh network with wireless IP routers
- single link without IP routers
- network segment containing several IP routers interconnecting meshes and single links but without IP border router.
- Any of the three networks above with one or more border routers connecting the network segment to Internet.

The installation process knows several phases:

Physical connections: Devices are unpacked, and installed at their location and connected to the mains (if appropriate) and connected to the network segment.

Logical connections: Devices are connected to the "logical network" and receive a host name such that the mapping: host name to IP address is established.

Grouping: Groups are defined.

Binding: Client applications bind to service instances (group of service instances) and the application is operational.

During installation, applications start operational life on a evolving "logical network".

During and after installation, the network segment goes through any sequence of the stages "Stand-alone", and "Connected". The limits of the "logical network" are defined as a function of the two network segment states:

1. All the devices connected to the stand-alone segment.

2. Border routers on the connected segment which delimit the boundaries of the "logical network".

Once the network segment is connected, the logical network represents a zone in DNS terms.

4. Naming conventions

In this and following sections design decisions are formulated to define the context in which the requirements can be made concrete.

4.1. Host name

Choosing an existing naming format limits the possibilities but also helps to profit from existing and proven techniques.

Design decision 1: host names use the naming scheme established for DNS.

This decision implies that the host name is composed of sequences of characters delimited by dots. Accordingly, the host name can reflect the hierarchical structure that is used for its identification within the installation (topological or organizational). The structure’s

use is explained in Section 4.3. The prefix of the host name is the name of the single unconnected device given during installation. The device name can be post-fixed with the installation structure. The latter can be post-fixed with its DNS domain name. For example, the host name can look like:

host name	scope
Device1	within the stand-alone network with a flat name structure.
Device1.floor1.bldg3	within the stand-alone network with a hierarchical name structure.
Device1.floor1.bldg3.example.com.	Fully Qualified Domain Name (FQDN) within the Internet connected network.

The prefix device1 can have human interpretable names like TV_in_kitchen which has meaning to the occupants of a home. Within the context of automated building control, the names are most likely machine interpretable and can represent any sequence of characters, like 213_abk_0004545588, which takes its meaning from the device's technical or organizational details, at the same time making the name unique within the installation.

This specification uses the naming convention name.ld.gd., where ld represents a "local domain" possibly with a structure like floor1.bldg3, and gd represents the "global domain" name supported by DNS like: example.com. The dots separating domain names in "global domain" are recognized by DNS for the definition of zones. The dots separating names in the "local domain" name are not seen by DNS but provide a consistent hierarchical naming convention. Its example use for service discovery is shown in Appendix A.

Design decision 2: Host and service instance names with a local scope are suffixed with global domain names to global scope.

The handling of roaming devices is out of scope of this requrement specification.

This is analogous to the scoping of telephone numbers. For example, within a small village all occupants can be reached by dialling a 6 digit number. Dialling nationally outside the village area necessitates a larger number, and the local 6-digit number is prefixed with the national area code. International access necessitates an international prefix code. This numbering policy has

a proven track record and conforms to a mental picture shared by a large majority of people.

Accordingly, applications use the names with the gd extension when access to services outside the local scope is needed. This approach has one large advantage:

Applications on devices connected to a "logical network" use names without global domain suffix for binding to servers on that network segment independent of the segment state (connected or stand-alone).

4.2. Service name

A device can host servers. The server provides a service instance.

Design decision 3: The name of the service instance follows the Instance.Service.Domain format defined in [RFC6763].

Where the "Service" part is composed of a name reserved by IANA for a given SDO. For example, the name _bc._udp (following [RFC2782]) can be used for the example building control service, where bc is (hypothetically) reserved by IANA. A service can be composed of subtypes (e.g. light) prefixed to the service type, resulting in: light._sub._bc._udp.

The "Instance" part can be the host name prefix, or the UID of the device. It is possible that there are multiple servers for a given device (e.g. multiple lights controlled by a device). In that case the service instance identifier must be extended to distinguish the service instances providing identical services on the same device.

The "Domain" part is composed of "ld.gd" as specified in Section 4.1. Similar to the host name, the service instance identifiers are known within the "logical network" without the "gd" suffix.

4.3. Discovery scope

Names of hosts and service instances are locally unique when they are unique within the "logical network". Names used as input to queries can include the suffixes "ld" and "gd". When a name includes the "gd" suffix the "ld" suffix MUST be included as well.

It should be noted that there is no 1-1 mapping from domain hierarchy to network segment topology. Given the hierarchy of Section 4.1, all devices of floor 1 to floor n can be connected to one switched ethernet segment. The lights in one floor (even one office) can be connected to different powerline segments.

The following rules SHOULD apply and be enforced by the service discovery service:

1. Names with the "gd" suffix are globally unique and are consequently locally unique.

2. Names with a "ld" suffix but without "gd" suffix are locally unique.

3. Names without "ld" suffix can be locally unique

4. Non-unique names without "ld" suffix should be locally unique with an appropriate "ld" suffix.

It is required that the underlying service discovery service guarantees the uniqueness of the names when they are declared to the network.

The local domain suffix is useful to group devices which are related. For example the location of the device can be communicated in the local domain name as proposed in Section 4.1. An application that wants to discover all light services in a given office, off3, queries all devices which provide service, light._sub._bc._udp, within domain off3.floor1.bldg3. Querying all devices within domain floor1.bldg3 or bldg3 returns all light services at floor 1 or within the whole building 3 respectively. The Resource Records for local service discovery are shown in Appendix A.

When the global domain is example.com, the queries to floor1.bldg3 and floor1.blg3.example.com should return the same results.

When in an early phase of the installation the local and the global domain names are not known, the query for light._sub._bc._udp should return all service instances present on the devices connected to the stand-alone segment.

5. Installation procedures

Two types of installations can be identified: "managed" and "plug-and-play".

In building control, often a tight control on device presence (and absence) and physical location is necessary to guarantee a smooth operation and maintenance. Mostly, the installation of devices for building control is "managed" accompanied by installation dependent naming guidelines.

In the home environment it is assumed that humans will notice absence and presence of devices visually, and names can be automatically chosen by devices, and may be changed to more personal names when experience invites users to do so. Mostly the installation for homes is "plug-and-play" without external installation dependent guidelines.

The terms "managed" and "plug-and play" are used when referring to any of these two installation approaches.

5.1. Managed Installation

Commissioning is the process of pairing the factory provided device identifier and the host name, and the consecutive binding of the application to service instances and groups. The factory provided device identifier can be the MAC address, used in the remainder of the text.

Design decision 4: A Commissioning Tool (CT) supports the storing of the relations in DNS-SD based structures.

The storage technique for DNS-SD relations should be transparent to the applications. Example storage techniques are:

- Distributed memory as implemented by mDNS [RFC6762]

- Hierarchic storage as implemented by DNS [RFC1035]

The CT contains information about the devices as prescribed by architect or Installation Company. The information in the CT describes host name -possibly suffixed with local domain-, location in the building, and the group names with its members. Before commissioning, the relation between the host name and the MAC address is unknown. The commissioning is based on two actions:

1. The CT learns the MAC address of a given device installed at a given location by reading a bar code (or pushing buttons, switching on/off equipment, etc.).

2. The CT learns the host name by pointing at a map of the building (or selection from a list, typing in the name or any other appropriate technique).

Uniqueness of the host name is assumed to be guaranteed by the installation process.

In this way the CT pairs the host name with the MAC address. Assuming the CT to be connected to the "logical network", all kinds

of techniques can be used to establish the relation between IP address and MAC address. For example, the link-local IP address of the device can be constructed from the MAC address. Having learnt the IP address, the CT can communicate the host name to the device.

The CT can learn the attributes of the services instances available on the device by querying /.well-known/core on the device. Alternatively, the CT already obtained the service instance attributes from a configuration file. Given these data, the CT can enter the host names and associated services into DNS-SD based storage structures.

Either automatically, or on instructions of an operator, or much later in the commissioning process, the CT can define the groups in DNS-SD. Before commissioning, the CT has a list of group names. For each group a service type and the host names of the members are specified. With additional information about the path of the services in the device and the port number of service, the groups can be fully specified in DNS-SD.

5.2. Plug-and-play Installation

The constraints on the host name and service instance name specified in Section 4.3 are assumed to be valid.

The relation between the factory provided device identifier (e.g. MAC address) and host name has to be established without any intervention by the installing person. The device is connected to the network and acquires its IP addresses (link-local, or global) according its installation characteristics (DHCP, or stateless). The device acquires a host name, either proposed by the manufacturer (for example, stored in the device at manufacturing location) or specified by the installing person. Uniqueness of the host name within the "logical network" must be ascertained by the underlying "discovery service". At the location of the device, the host name is now related to the MAC address and consequently to the IP-addresses of the device. The relation host name to IP address is published to the service discovery service. After successful publication, host name, IP address, and service instances are accessible to all other devices on the "logical network".

From this moment, applications on a device can discover service instances of a specified service offered by the devices connected to the "logical network".

5.3. Group declaration

Group declaration is managed both in managed and plug-and play environments. The CT defines the group in the DNS-SD repository.

A group has a name associated with a set of service instances. When the group is accessed with a multicast invocation, a multicast address, the port number, and a path to the service instance MUST be specified in the DNS repository. Multicast address, path, and port number are equal for all members of the group because transported in the same message to all destinations.

5.4. Binding

Applications need to bind to the service instances which provide a requested service. The service is identified with a service type, and with the service type the application knows how to access the service. The application learns the service instances present at the logical network, identified by:

- Path to the service instance.
- Host name of the device hosting the service instance, or group name of the set of service instances.

Binding can be done in mainly two ways:

Managed: a third device (CT) specifies the service instance to which the given application needs to bind.

Plug-and-play: the service requesting device learns the service instances of service providing devices via service discovery.

The device (group) name should remain valid during the lifetime of the installation independent of the state of the "logical network".

5.5. Device naming

Devices obtain their host name as function of the "managed" or the "plug-and play" mode.

Managed: The CT allocates the host name from a list. The host name can be flat or hierarchical expressed in the local domain suffix, but its global domain suffix is not necessarily known at allocation time. Given that networks can disconnect from and reconnect to the Internet the device is known within its "logical network" by its host name without the global domain suffix.

Plug-and-play: the device assumes a default host name given by the manufacturer. Uniqueness can be guaranteed by adding random

numbers when necessary. In a later stage users have the opportunity to change the host name to their liking. For the same reasons as the managed case, the device is known within its "logical network" by its host name without global domain suffix.

TODO: mobility of devices is not considered.

6. Service discovery requirements

Service discovery supports that an application can learn the names of service instances or group name of service instances and the names of the hosting devices. The discovery service supports the following:

Name_resolution: Resolve the group name, host name to IP address; resolve service instance name to host name, port number, and path.

Create_group: Create a group of end-points providing a given service

Enrol_member: Enrol an end-point as member of a given group.

Remove_member: Remove an end-point as member of a given group.

A summary of the requirements specified in the text is presented in Table 2. The first 4 requirements are the four design decisions.

Number	Requirement	Text
BC1	Host names follow DNS format	Section 4.1
BC2	Local names are suffixed for global uniqueness	Section 4.1
BC3	Service instance names follow DNS-SD conventions	Section 4.2
BC4	A Commissioning Tool assists managed installation	Section 5
BC5	SD queries with local names return same results for stand-alone and connected segments	Section 3
BC6	Limits of connected "logical network" are defined by border routers	Section 3
BC7	A network segment is composed of one or	Section 3

	several connected wired and wireless links		
BC8	SD service guarantees local uniqueness of names	Section 4.3	
BC9	The storage technology of DNS-SD relations is transparent to application	Section 5.1	
BC10	The SD service stores group names and group members	Section 5.3	

Table 2

6.1. Mapping to requirements I-D

The requirements of [I-D.lynn-mdnsext-requirements] are mapped to the requirements of Section 6 in Table 3.

lynn-mdnsext-requirements	This document
REQ 1	BC5, BC6, BC8
REQ 2	BC7, BC9
REQ 3	BC6, BC2
REQ 4	BC5
REQ 5	BC1, BC3

Table 3

The mapping of Table 3 is not one to one. BC10 is not covered by [I-D.lynn-mdnsext-requirements]. BC4 is not applicable to [I-D.lynn-mdnsext-requirements].

The requirements of [I-D.lynn-mdnsext-requirements] are requirements on the service discovery service solution, while the requirements in this specification are specified at the application level.

The [I-D.lynn-mdnsext-requirements] does not address the changing states of the logical network, but mentions incremental deployment in REQ5.

7. Security Considerations

TODO: follows DNS-SD security provisioning.

8. IANA Considerations

TODO

9. Acknowledgements

The draft has benefited from comments by Dee Denteneer, Esko Dijk, MIchael Verschoor, Gerhard Mekekamp, and Michael van Hartskamp.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

10.2. Informative References

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, September 2007.

[RFC6690] Shelby, Z., "Constrained RESTful Environments (CoRE) Link Format", RFC 6690, August 2012.

[RFC6762] Cheshire, S. and M. Krochmal, "Multicast DNS", RFC 6762, February 2013.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, February 2013.

[RFC6775] Shelby, Z., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, November 2012.

[I-D.ietf-core-coap]
Shelby, Z., Hartke, K., and C. Bormann, "Constrained Application Protocol (CoAP)", draft-ietf-core-coap-18 (work in progress), June 2013.

[I-D.lynn-mdnsext-requirements]
Lynn, K. and S. Cheshire, "Requirements for DNS-SD/mDNS Extensions", draft-lynn-mdnsext-requirements-02 (work in progress), July 2013.

Appendix A. RR for service discovery

This Appendix presents a simplified example to make the use of the local domain name more concrete. The naming used here is chosen to clarify the use of the local domain names for service discovery.

Suppose there are two areas area1 and area2 both containing two sensors providng a service with service type sensor._sub._bc._udp accessible via path /sns. Area1 and area2 are subdomains of floor4. The sensors are connected to different hosts with names device1 to device4. Suppose there are two devices app1 and app2 running client applications where app1 belongs to area1.floor4 and app2 belongs to floor2. App1 can discover all sensor service instances in its own domian area1.floor4 by doing a query for all PTR RR with name _sensor._sub._bc._udp.area1.floor4. App2 can discover all sensor service instances in floor4 doing a query for all PTR RR with name _sensor._sub._bc._udp.floor4. In the latter case App2 must be configured to target floor4.

```
; comment-- device names to IP addressses
device1.area1.floor4      IN AAAA              fdfd::01
device2.area1.floor4      IN AAAA              fdfd::02
device3.area2.floor4      IN AAAA              fdfd::03
device4.area2.floor4      IN AAAA              fdfd::04
app1.area1.floor4         IN AAAA              fdfd::05
app2.floor2               IN AAAA              fdfd::06

; comment-- service instances to end points
ii_sensor      IN SRV 0 0  61614     device1.area1.floor4
               IN TXT                txtver=1 path=/sns
jj_sensor      IN SRV 0 0  61614     device2.area1.floor4
               IN TXT                txtver=1 path=/sns
kk_sensor      IN SRV 0 0  61614     device3.area2.floor4
```

```
                IN TXT                txtver=1 path=/sns
   ll_sensor    IN SRV 0 0  61614     device4.area2.floor4
                IN TXT                txtver=1 path=/sns

   ; comment-- service in domain to service instances
   _sensor._sub._bc._udp.area1.floor4    IN PTR    ii_sensor
   _sensor._sub._bc._udp.area1.floor4    IN PTR    jj_sensor
   _sensor._sub._bc._udp.area2.floor4    IN PTR    kk_sensor
   _sensor._sub._bc._udp.area2.floor4    IN PTR    ll_sensor

   _sensor._sub._bc._udp.floor4          IN PTR    ii_sensor
   _sensor._sub._bc._udp.floor4          IN PTR    jj_sensor
   _sensor._sub._bc._udp.floor4          IN PTR    kk_sensor
   _sensor._sub._bc._udp.floor4          IN PTR    ll_sensor
```

According to [RFC6763], the service instance name yy_sensor (with y in {i,j,k,l}) should be yy_sensor._sub._bc._udp.areax.floor4. In the example the shorter (and also unique) name is used for clarity reasons.

The AAAA RR specify the IP addresses of the devices. The SRV RR combined with the TXT RR specify the hosting device of the service instances together with port number and path. The PTR RR specify the service instances associated with a service in a given domain.

Authors' Addresses

Peter van der Stok
consultant

Phone: +31-492474673 (Netherlands), +33-966015248 (France)
Email: consultancy@vanderstok.org
URI: www.vanderstok.org

Kerry Lynn
Consultant

Phone: +1-978-460-4253
Email: kerlyn@ieee.org