CoRE P. van der Stok, Ed.
Internet-Draft vanderstok consultancy
Intended status: Informational K. Lynn
Expires: January 11, 2013 Consultant
A. Brandt
Sigma Designs
July 10, 2012

CoRE Discovery, Naming, and Addressing draft-vanderstok-core-dna-02

Abstract

This is a working document intended to focus discussion and refine draft language for the CoAP protocol specification (or other proposed standards) in the areas of discovery, naming, and addressing. Requirements on discovery are motivated. Special emphasis is placed on group definition and discovery. Examples show how groups can be represented in CoAP Resource Directories or DNS-SD.

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 11, 2013.

Copyright Notice

carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

The CoRE working group is chartered to design and standardize a Constrained Application Protocol (CoAP) for resource constrained devices and networks [I-D.ietf-core-coap]. The requirements for CoRE are documented in [I-D.shelby-core-coap-req]. This draft discusses the requirements on service discovery for M2M and interactive applications using resource constrained devices. We propose the use of DNS-SD [I-D.cheshire-dnsext-dns-sd] and Resource Directory (RD) [I-D.shelby-core-resource-directory] to satisfy the requirements. The proposal relies heavily on naming and addressing conventions. Special emphasis is placed on the definition, naming, and discovery of groups.

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 RFC 2119 [RFC2119]. Addtional privileged words are described below.

A "device" is a physical processor connected to at least one link through a network interface. Each interface has at least one IP unicast address. The IP address is optionally bound to a host name, which may be a Fully Qualified Domain Name (FQDN).

An "end-point" corresponds to a "service" and is indentified by a unique {protocol, host, port} tuple. The identity of an endpoint may be specified by the 'scheme' and 'authority' parts of a URI [RFC3986]. 'Protocol' is a label that indicates application and transport protocol bindings as well as default port (if port is not specified) and possibly default semantics such as web-linking [RFC5988]. 'Host' corresponds to the [RFC3986] ABNF production as updated by [I-D.ietf-6man-uri-zoneid].

A "service type", e.g. _bldg-ctrl._tcp, is equivalent to the 'protocol' label described above. It identifies an application protocol, typically defined by a Standards Development Organization (SDO), and is ultimately registered with IANA [I-D.cheshire-dnsext-dns-sd]. The SDO may additionally specify service subtypes (e.g. _light, _onoff-control, _air-flow ...) to designate units of functionality, the attributes of the subtypes, and the primitives acting on the attributes.

A "resource" is any attribute of an end-point that can be acted upon by REST methods. A resource is identified by a URI, that is, an end-point plus a 'path' component [RFC3986].

A "Function Set" is a service subtype with a set of resources and behaviors that may be accessed through a REST interface. A Function Set will typically be decribed by a base URI plus an interface definition as described in [I-D.shelby-core-interfaces]. The interface definition may specify the naming patterns of subordinate resources and the methods that act on them.

A "Profile" is a group of Function Sets defined by a specification.

A "Collection" is a set of two or more homogeneous subordinate resources that may be acted upon in the aggregate by sending messages to their parent resource, or individually by sending messages to the collection member.

A "Device type" describes a standardized set of Function Sets that satisfy a use case for a hosting device.

A "group" is a set of end-points.

A "multicast group" is a group of end-points that share a multicast address. The multicast address is optionally bound to a FQDN that identifies the multicast group. For the purposes of this document, a (multicast) group generally hosts one Function Set.

1.2. Motivation

In this draft, we focus and expand discussions on requirements pertaining to discovery, naming, and addressing, including REQ8 of the CoRE charter:

> REQ8: A definition of how to use CoAP to advertise about or query for a Device's description. This description may include the device name and a list of its Resources, each with a URL, an interface description URI (pointing e.g. to a Web Application Description Language (WADL) document) and an optional name or identifier. The name taxonomy used for this description will be consistent with other IETF work, e.g. draft-cheshire-dnsext-dns-sd. [charter]

The basis of this draft originated in [I-D.vanderstok-core-bc].

1.3. Applicability

This note applies to discovery of services for commissioning and auto-configuration in building networks (for example: residential and

office). The examples are inspired by the building environment. The proposed solutions and recommendations can be applied more generally.

2. Architecture

This section illustrates the aspects of naming schemes and their support by DNS-based Service Discovery (DNS-SD) [I-D.cheshire-dnsext-dns-sd] , Extended Multicast DNS (xmDNS) [I-D.lynn-homenet-site-mdns], and the Resource Directory (RD) [I-D.shelby-core-resource-directory] on a set of network architectures.

The basic network for low-power nodes can be composed of low-resource nodes sharing the same IPv6 prefix and connected to low-power links like IEEE 802.15.4, ITU-T G.9959, or Powerline. The "lowpan" is a good example of such a network [RFC4944], [RFC6282]. The network can be either isolated or connected. This draft assumes that application profiles are defined above coap or http, for example, applications as specified by the ZigBee Smart Energy Profile 2.0 (SEP2), Obix, or BACnet IT working groups. The naming and discovery solutions presented here are applicable to multiple interconnected subnets. Example network architectures are:

- An isolated lowpan consists of at least two lowpan devices, one of which is an edge router that is not connected to a backbone. A Resource Directory may be situated on the edge router. Alternatively, xmDNS responders may execute on each device.
- A connected lowpan consists of at least two lowpan devices, one of which is an edge router that is connected to a backbone. A Resource Directory may be situated on an edge router. Alternatively, xmDNS responders may execute on each device or the DNS-SD service may be available over the backbone.
- Interconnected lowpans consist of at least two lowpans connected via edge routers to the same backbone. A Resource Directory may be situated on each edge router. Alternatively, xmDNS responders may execute on each device or the DNS-SD service may be available over the backbone.
- A site is a set of interconnected subnets that is locally administered. A site may include zero or more lowpans. Border routers may prevent some messages from passing into or out of the site.

In certain scenarios, the domain may correspond to the network topology. In the general case, the domain and network subnet structure may differ.

3. Use Cases

The use of service discovery is presented in two environments: (1)

interactive service discovery, and (2) M2M service discovery. From

the use cases we derive the types of queries that service discovery should support. In particular, a primary motivation is the discovery of groups that support a given Function Set.

3.1. Discovery scope

In the simplest case the discovery scope is limited to the services and resources provided on the LowPAN. In more complex cases, discovery has a scope that can be defined with domain names. The authority part of a URI [RFC3986] can express the location of the device hosting the service. A common example is the naming associated with the structure of a building. A device may acquire a FQDN that relates directly to its location in the building. For example, power-strip.office4.floor1.example.com (or shorter ps.o4.f1.example.com) refers to a power-strip with device name "power-strip" (or short "ps") in office4 at floor1 in the building of the company example.com. Another naming scheme can be functional like TV1.media.IT.example.com, possibly refering to TV number 1 maintained by the media group of the IT department of the example organisation. Domain naming can be used to express that devices are situated in the same building area or belong to the same organisational units. Multiple FQDNs can identify a given device.

The DNS provides the mapping from Fully Qualified Domain Name (FQDN) to network address and vice-versa. The RD realtes domain names to end-points. The binding of domain names to a physical device (for example, assigning a given FQDN to the TV in the corner) depends on the operational conditions as described below.

3.2. Interactive mapping

In the residential context, naming of the device is done by the occupant of the home. After connecting the device to the network, an IP-address is assigned, possibly based on the EUI-64 value of the network interface. The occupant can use a remote control with a graphical user interface to display all devices that provide a given service (e.g. a "lighting" service). The remote control prompts the occupant to identify the (default named) devices, possibly accompanied by on/off switching, barcode scanning, or other manual intervention. The occupant can provide a meaningful name that will be bound to that device. The installation steps can be as follows: Service discovery returns all interfaces on which the specified service is available. The occupant, with or without additional physical support, establishes the binding between an IP address (based on MAC address or other unique identifier) and the name of the device providing the service, plus its relationship with other devices or function sets in the system. In some cases a device has multiple names, and an additional mapping between user specified

name

and an automatically generated DNS name is supported.

3.3. M2M mapping

In the commercial context (e.g. office buildings) it is usual to employ a Comissioning Tool (CT) to provide the mapping from physical device to IP network address or FQDN. The professional context is more rigid than the home because the absence of devices and also the unwanted presence of devices needs to be detected. Devices are named by an architect or installation contractor. Names can be generated automatically and need not be human-friendly. The CT contains the names and the physical locations of the devices. At commissioning time the interfaces have acquired a network address, possibly based on the EUI-64. The physical device is identified by reading in a unique identifier (e.g. EUI-64 of interface, UUID of device) with a reader (e.g. barcode reader). Consequently, the device name to network address binding is stored into DNS (or elsewhere). Alternatives for identifying devices are pushing buttons on the device or remotely switching on/off the device.

3.4. End-point grouping

Groups can be used to express that end-points are related by supporting a specific Function Set (e.g. HVAC equipment controlled by the closest temperature sensor). Grouping is also necessary when a set of end-points has to react together, more or less synchronously, to a sequence of commands sent by one or more devices.
A common example is provided by lighting applications where a subset of lights in the building are dimmed to the same level, set to the same colour, or switched off simultaneously. Another example is provided by a power-strip supporting a set of power-outlets. Power-outlets are switched on/off individually or all together. Other examples concern the home, such as a "sleep mode" setting of all media devices in the home when the user activates the night scene.

Group naming is done the same way as for device naming. Related end-points are grouped and named. The group name is constructed like a FQDN with the group name as prefix. Adressing the group can be done in two ways: (1) by addressing each end-point of the group individually (which requires serial access), or (2) by defining a multicast address for a multicast group. In the latter case, each hosting device must enable reception by a given end-point of the messages sent to the multicast address. The members of the multicast group must have identical port number and path, because the requests are specified in a single multicast message.

The Function Set specified in a message can contain a collection of resources of the same subtype. The Function Set path postfixed with

an identifier refers to the individual resources of the collection. For example: the /path/light points to the resource collection of the service subtype light. Each member of the collection can be identified with /path/light/x, with x in {1,2,3,..}. Consequently, the /path/light/1/onoff specifies the onoff resource of collection member 1 of Function Set with /path/light, and /path/light/onoff specifies the resource onoff of all collection members contained by the Function Set with /path/light. When /path/light/onoff is used in a multicast message, it is interpreted as a message to a single light resource by devices having only one, and to all members of the collection for devices having several light resources.

In [I-D.shelby-core-interfaces] the Batch interface is used to manipulate a collection of subresources with one message. Both the batch and the collection are static. The collection contains homogeneous resources where the batch can contain heterogeneous resources.

It is expected that SDOs will standardize resource types, profiles (path names) and group naming conventions. Manufacturers design the functions supported by a device and select the Profiles, resources and collections. Each manufacturer can precede the profile path names with a manufacturer defined path prefix; for example when a device supports multiple standards. Domain name, device name, group name and group members are installation dependent.

Figure 1 illustrates some of the concepts described above. A device is identified with the name "Power-strip". In the example, no domain names are associated with the device, and the name "power-strip" resolves to a Unique Local Address (ULA)[RFC4193]. The device provides two end-points: one delivers a http service and the second a CoAP service. The http server and CoAP server share the same IP-address and use different ports 80 and 61616 respectively. Two different service subtypes, one identified by Function Set, ps, and one identified by Function Set, pm, are supported by one end-point. The Function Set "Power strip" contains a collection of four resources, "Outlet 1" to "Outlet 4", each one accessible via the accompanying paths /ps/1 to ps/4. The path /ps interfaces to the entire resource collection. The attribute "output" is defined in the service subtype specification.


"Power-strip"
[device] has at least one NIC/IP address, may have a name
"HTTP server"
[End-point] http://power-strip (at default TCP port 80)
"CoAP server"
[End-point] coap://power-strip (at default UDP port 61616)
"Power Meter"
[Function Set] coap://power-strip/pm
"Power strip"
[Function Set] coap://power-strip/ps
"Outlet 1"
[Collection member] coap://power-strip/ps/1
[resource] coap://power-strip/ps/1/output

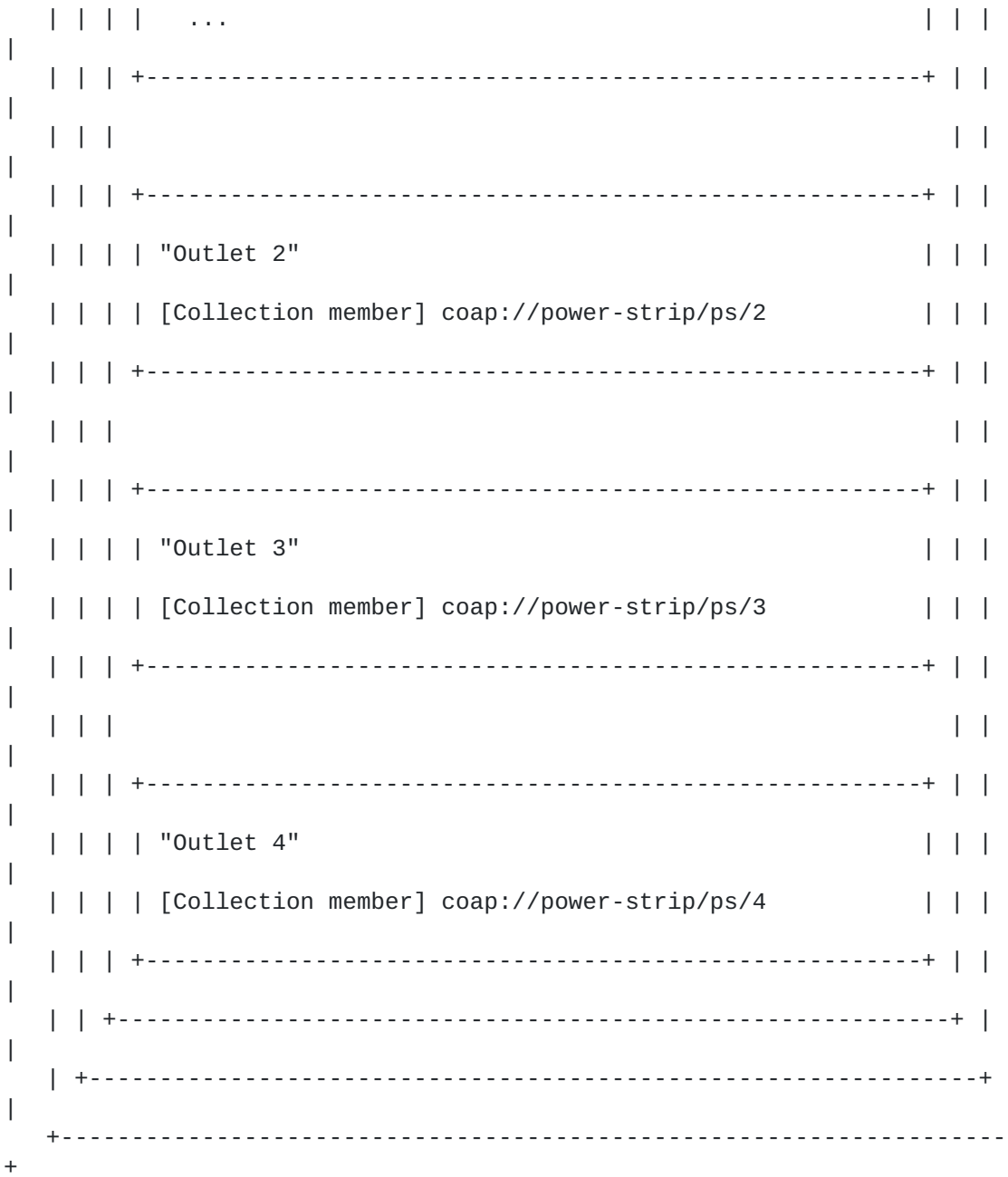

Figure 1: device with end-points, Function Sets and resources

3.5. Discovery queries

The following conditions are assumed to be valid. A device has acquired an IP address and a name, possibly stored in DNS. The resource naming (path) in the device obeys a standardized profile. The resource types of the Function Sets are also standardardized. A device may belong to a domain.

Service discovery should support that a device can learn its domain(s) and all the end-points within a domain providing a given service (e.g. temperature measurement). The device learns of these end-points the path name that prefixes the path names defined by the profile. Devices need to learn the groups to which they belong and learn all the members of those groups. This section motivates that a discovery service supports the following queries:

Goal	Description
Name_resolution	Resolve the group or device name to IP address and optional port number
Return_server	Return all end-points (Function-Sets) supporting a given service (sub-)type within a given domain
Create_group	Create a group of end-points providing a given service (sub-)type within a given domain
Enroll_member	Enroll an end-point as member of a given group
Remove_member	Remove an end-point as member of a given group
Return_group	Return all groups of which a given end-point (device) is a member
Return_member	Return end-points belonging to a given group

Name_resolution is supported by DNS and CoAP resource discovery. Names are required in the context of home control and manual setup of installations. Names are persistent and meaningful as compared to IP addresses and are preferably used in applications when IP addresses can change.

Return_server is the most common use of service discovery and was originally designed for interactive use. The canonical IT example is finding all printers within a zone, which allows a user to select the desired printer from the returned list. Another example is in the context of UPnP [UPNP], where all media players are returned on a screen and the user can select the desired media player on the screen and play the selected content. In M2M applications, the returned names are not displayed on a screen but an application uses the returned list to select a (set of) Function Set(s) to control. Consequently, names in M2M applications need not be human

interpretable (for example, they can be unique numbers).

Create_group is useful in commissioning scenarios, where end-points

need to be grouped to receive the same command in a possibly synchronous fashion. Groups can also be created to express relations between devices such as ownership. The command creates a group name and creates a list of the members of the group. When the group is a multicast group, the command defines a unique multicast address and port, and specifies the path.

Enroll_member supports network and device reconfiguration. When the physical lay-out of an installation changes because devices are added, changed or removed, the associated groups also need to be modified.

Remove_member, see motivation under Enroll_member

Return_group is needed to learn the groups of which an end-point is member. The command is necessary for commissioning purposes where a Commissioning Tool (CT) is used. The CT, on the basis of designs provided by architects, decorators, sound/light engineers, defines groups and group members and stores that information in the service discovery database. In the next phase, the members of a group may need to learn their membership from the service discovery to enable reception of multicast messages.

Return_member can be used to learn which end-points are member of a given group. This command is useful in connection with Return_group. The end-point knowing to which groups it belongs can establish communication with the group members. For example, membership of a group instructs new devices, replacing faulty ones, which other devices share access rights or need to be consulted regularly.

3.6. Sleeping devices

This section suggests that service discovery of sleeping devices is mostly a matter of discovering the proxy. It is expected that a proxy will handle communications for the sleeping device, as expressed in [I-D.giacomin-core-sleepy-option] and in [I-D.vial-core-mirror-proxy]. The message sent to the sleeping device is directed to the proxy. The proxy will send the message on with a delay, or send the result of a function on the history of messages, when the sleeping device is ready. Different communication protocols between proxy and sleeping device are described in [I-D.giacomin-core-sleepy-option] and in [I-D.vial-core-mirror-proxy]. The setting up of the proxy is preferably standardized for a large set of proxy types. During the setting-up process, (offline or online) the proxy will take over all the entry-points of the sleeping device. The entry-points of the proxy can be entered into the discovery respository and consequently discovered like any other device.

For groups, two cases need to be considered (1) sleeping device is member of a group and receives group messages, and (2) the sleeping device sends messages to a group. Ad (1), when the sleeping device needs to receive messages sent to a group, the proxy will receive those messages and the end-point of the proxy is entered as group member to receive the group messages. Ad (2) When the sleeping device sends messages to a group, it is preferable that the sleeping device sends just one multicast message to the group to minimize energy costs. It is required that when one member of the group receives the message, all other group members receive it as well (unanimity), covered by the "reliability" REQ1 in [I-D.ietf-core-groupcomm]). A simple broadcast over a lowpan will not always succeed and additional multicast algorithms like Trickle [RFC6206] need to be introduced.

4. DNS and RD examples

The following device configuration and environment are assumed for the examples. The devices are placed on floor-x (fx) in two rooms room-y (ry) and room-z (rz). Both rooms contain a powerstrip with a powermeter and four power outlets. In each room there are two luminaires and one presence sensor (PIR). Each luminaire contains a dimmable light and a light sensor. Per floor there is a clock to set day and night time modes of the devices. The domains are: ry.fx.bldg.org, rz.fx.bldg.org and fx.bldg.org. The device names of the 4 luminaires are lm00203, lm00204, lm00205, and lm00206. The device names of the two powerstrips are ps0057, and ps0078. The paths of the Function Sets of the luminaire are: /lamp with resource /lamp/dim for the dimmable light, and /light with resource /light/ lumen for the light sensor. The Function Set path for four outlets is /ps with resource /ps/output. The path of each individual outlet is /ps/x with x in {1,2,3,4}, and with resource ps/x/output. The names of the two PIRs are pir0 and pir1. The Function Set path of the PIR is /occup, also being the resource.

Relating location to the domain name is a relevant example of domain naming. Multiple domain names, related to other application aspects, can be specified and applied simultaneously.

As described in section 3, devices do not announce themselves to the discovery repository, as is usual for IT applications, but they are entered (partially) with the aid of a central tool, for example a Remote Control, dedicated device, IPAD or other means.

The examples are constructed such that DNS can be used as repository for the RD.

4.1. DNS-SD examples

4.1.1. Basic Concepts

In conformance with [I-D.cheshire-dnsext-dns-sd], DNS-based discovery uses A or AAAA, PTR, SRV, and TXT Resource Records (RR). The SRV RR [RFC2782] specifies an end-point. An associated (identically named) TXT RR can contain a URI path. The SRV/TXT pair can specify a Function Set. An A or AAAA RR [RFC1035] binds a device name or multicast group name to an IP address. The PTR RR binds a service type to an end-point or Function Sets.

In cases where the end-point port may be dynamic, e.g. in the IPHC [RFC6282] compressible range, a new 'coap+srv' scheme is proposed (after [I-D.jennings-http-srv]). The authority part of a coap+srv URI specifies the name and location of an SRV record, which in turn contains values for device (IP address) and port.

4.1.2. Commissioning devices

Commissioning is the process to store the relation between a FQDN and a device. It is assumed that either a Remote Control (RC) in the home or a Commissioning Tool (CT) in the professional domain store the relation in DNS.

In the professional domain, the CT is assumed to contain information about the devices as prescibed by architect or installation company. The information in the CT contains device name, device domain name, and location in the building, but the relation with the installed device, identified with an unique identifier (e.g. EUI-64) is not established. By reading a bar code (or pushing buttons, switching on/off equipment, etc.), the CT learns the unique identifier of the device to be commissioned. All kinds of techniques can be used to establish the relation between IP address and unique identifier. When the identifier is the EUI-64 value, the IP address of the device can be constructed. When the identifier is not the EUI-64, a proprietary procol can be used to ask a given device its identifier. Etc. etc. The CT can learn the end-points (services) available on the device by querying /.well-known/core. In some cases the CT already obtained the end-points from a configuration file. Given these data, the CT can enter the devices and its services into DNS. Either automatically, or on instructions of an operator, the CT defines the groups in the DNS.

The home domain is different from the professional domain in the sense that no configuration information exists. The RC can for example use xmDNS to learn the addresses of all the devices present in its site. The RC can query devices for the presence of a given

service. The RC can query DNS for its own domain name and use that for the other devices in the site. Once (new) devices are named, this information can be stored in DNS for use in the network.

4.1.3. device examples

The relation between device name and IP address is expressed for the example devices in the following table.

```
lm00203.ry.fx.bldg.org. IN AAAA fdfd::1234
lm00204.ry.fx.bldg.org. IN AAAA fdfd::1235
ps0057.ry.fx.bldg.org.  IN AAAA fdfd::1236
pir1.ry.fx.bldg.org.    IN AAAA fdfd::1237
lm00205.rz.fx.bldg.org. IN AAAA fdfd::1238
lm00206.rz.fx.bldg.org. IN AAAA fdfd::1239
ps0058.rz.fx.bldg.org.  IN AAAA fdfd::1240
pir2.rz.fx.bldg.org.    IN AAAA fdfd::1241
clock.fx.bldg.org.      IN AAAA fdfd::1242
```

The next part defines the Resource Records to specify the end-points and their Function Sets. The names of the SRV RRs need to be unique to the DNS server, and follow the naming suggested in [I-D.cheshire-dnsext-dns-sd]. An end-point is represented with one SRV RR with a name "_service.domain". A Function-Set is represented with a SRV/TXT pair with name "subtype._sub._service.domain".

The service type "_bc._udp" is assumed to exist with the service subtypes "lamp", "sensor", "power", "presence", and "timer". Unique names of the SRV RRs can be created from the service subtype prefixed by the EUI-64 value. With EUI-64 value "1234" the SRV name 1234_bc._udp.domain can be created for the corresponding end-point. Given that the service _bc._udp supports subtype lamp the name 1234lamp._sub._bc._udp.domain is created for for each SRV/TXT pair describing a Function-Set serving the lamp subtype. They are valid within the authority zone, bldg.org, of the name server. For presentation purposes, 1234_bc is short for 1234_bc._udp.bldg.org, 1234lamp is short for 1234lamp._sub._bc._udp.bldg.org, etc. The luminaires with name "lm00xxx" provide each one end-point hosting two Function Sets with the paths /lamp and /light.

```
            end-point                  host name
            1234_bc   IN SRV 0 0 Port  lm00203.ry.fx.bldg.org
                      IN TXT txtvers=1
            2345_bc   IN SRV 0 0 Port  lm00204.ry.fx.bldg.org
                      IN TXT txtvers=1
            3456_bc   IN SRV 0 0 Port  lm00205.rz.fx.bldg.org
                      IN TXT txtvers=1
            4567_bc   IN SRV 0 0 Port  lm00206.rz.fx.bldg.org
                      IN TXT txtvers=1
            5678_bc   IN SRV 0 0 Port  ps0057.ry.fx.bldg.org
                      IN TXT txtvers=1
            6789_bc   IN SRV 0 0 Port  ps0058.rz.fx.bldg.org
                      IN TXT txtvers=1
            7890_bc   IN SRV 0 0 Port  pir1.ry.fx.bldg.org
                      IN TXT txtvers=1
            8901_bc   IN SRV 0 0 Port  pir2.rz.fx.bldg.org
                      IN TXT txtvers=1
            9012_bc   IN SRV 0 0 Port  clock.fx.bldg.org
                      IN TXT txtvers=1
```

The above list of SRV RRs specifies the attributes of the end-points:
devices, port numbers, and IP addresses with related AAAA RR. The
Function Sets are specified with a another set of SRV/TXT pairs. The
accompanying TXT RR specify the paths of the associated Function
Set(s). The accompanying example table is:

```
  Function Set                                host name
  1234lamp     IN SRV 0 0 Port                lm00203.ry.fx.bldg.org
               IN TXT txtvers=1 path=/lamp
  1234sensor   IN SRV 0 0 Port                lm00203.ry.fx.bldg.org
               IN TXT txtvers=1 path=/light
  2345lamp     IN SRV 0 0 Port                lm00204.ry.fx.bldg.org
               IN TXT txtvers=1 path=/lamp
  2345sensor   IN SRV 0 0 Port                lm00204.ry.fx.bldg.org
               IN TXT txtvers=1 path=/light
  5678power    IN SRV 0 0 Port                ps0057.ry.fx.bldg.org
               IN TXT txtvers=1 path=/ps
  7890presence IN SRV 0 0 Port                pir1.ry.fx.bldg.org
               IN TXT txtvers=1 path=/occup
  3456lamp     IN SRV 0 0 Port                lm00205.rz.fx.bldg.org
               IN TXT txtvers=1 path=/lamp
  3456sensor   IN SRV 0 0 Port                lm00205.rz.fx.bldg.org
               IN TXT txtvers=1 path=/light
  4567lamp     IN SRV 0 0 Port                lm00206.rz.fx.bldg.org
               IN TXT txtvers=1 path=/lamp
  4567sensor   IN SRV 0 0 Port                lm00206.rz.fx.bldg.org
               IN TXT txtvers=1 path=/light
  6789power    IN SRV 0 0 Port                ps0058.rz.fx.bldg.org
               IN TXT txtvers=1 path=/ps
  8901presence IN SRV 0 0 Port                pir2.rz.fx.bldg.org
               IN TXT txtvers=1 path=/occup
  9012timer    IN SRV 0 0 Port                clock.fx.bldg.org
               IN TXT txtvers=1 path=/time
```

The names of the SRV records can be created automatically, as long as they identify the SRV records uniquely within the set of RR entries in the DNS zone. The SRV record with name xxxxpower stands for a power collection, accessed via the path /ps which refers to a Function Set that contains a collection of two or more resources.

PTR records specify the possible service discovery queries. The names of the PTR records are the names of the service (sub)types, defined by IANA, and their values refer to the names of the associated end-points (SRV records) or Function-Sets (SRV/TXT records). To support the query "all lamps within fx.bldg.org", the following PTR records need to be added for service subtype: lamp._sub._bc._udp.

```
lamp._sub._bc._udp.bldg.org IN PTR 1234lamp
lamp._sub._bc._udp.bldg.org IN PTR 2345lamp
lamp._sub._bc._udp.bldg.org IN PTR 3456lamp
lamp._sub._bc._udp.bldg.org IN PTR 4567lamp
```

Where xxxxlamp is still short for xxxxlamp._sub._bc._udp.bldg.org.

Equally to query all services within a domain, PTR records with as name the building control service, "_bc.udp", refer to all SRV records describing all discoverable end-points.

```
_bc._udp.bldg.org IN PTR 1234_bc._udp.bldg.org
_bc._udp.bldg.org IN PTR 2345_bc._udp.bldg.org
_bc._udp.bldg.org IN PTR 3456_bc._udp.bldg.org
_bc._udp.bldg.org IN PTR 4567_bc._udp.bldg.org
_bc._udp.bldg.org IN PTR 5678_bc._udp.bldg.org
_bc._udp.bldg.org IN PTR etc, etc.
```

It is shown above how PTR records support the queries filtered on service type. Filtering on domain can be done adding additional PTR records which select the Function Sets of a given type within a given domain. The set of PTR records below filters on all lamps within domain ry.fx.bldg.org.

```
lamp._sub._bc._udp.ry.fx.bldg.org. IN PTR 1234lamp
lamp._sub._bc._udp.ry.fx.bldg.org. IN PTR 2345lamp
```

4.1.4. Group examples

As an example, five multicast-groups are defined to group all lamps on floor "fx", all lamps in office "ry", all lamps in office "rz", all power-strips on floor "fx", and all devices in the building controlled by a central timer. The multicast-group names are entered into DNS like the device names to enable resolution from multicast-group name to multicast address.

```
lamp-fx.fx.bldg.org.    IN AAAA ff15::11
lamp-ry.ry.fx.bldg.org. IN AAAA ff15::12
lamp-rz.rz.fx.bldg.org. IN AAAA ff15::13
power-fx.fx.bldg.org.   IN AAAA ff15::14
timer-bldg.bldg.org.    IN AAAA ff15::15
```

It is expected that SDOs will specify naming conventions for group names, extending the service (sub)type names for end-points and Function Sets.

The path and port of the multicast-groups is defined with SRV and TXT RRs. Accordingly the group is bound to end-points (SRV RR) and Function Sets (SRV+TXT RR). For convenience we assume that a multicast group is bound to a service subtype. In the example below the groups concern the service subtypes "lamp", "power", and "timer".
(Remark gp1-lamp is short for gp1-lamp._sub._bc._udp.bldg.org, etc.)

```
gp1-lamp IN SRV 0 0 Port   lamp-fx.fx.bldg.org.
         IN TXT path=/lamp
gp2-lamp IN SRV 0 0 Port   lamp-ry.ry.fx.bldg.org.
         IN TXT path=/lamp
gp3-lamp IN SRV 0 0 Port   lamp-rz.rz.fx.bldg.org.
         IN TXT path=/lamp
gp-power IN SRV 0 0 Port   power-fx.fx.bldg.org.
         IN TXT path=/ps
gp-timer IN SRV 0 0 Port   timer-bldg.bldg.org.
         IN TXT path=/time
```

The groups for the power strips need extra attention because the power strips include a collection of resources. The path to the group can be defined as /ps or as /ps/x with x in {1,2,3,4}. When using /ps/x the group contains the outlet x of the power strips. Using the path /ps as done in the table above refers to all outlets of a powerstrip. When sending a message to the power-fx group with as path /ps/x then the message will be received by Function Sets with path ps/x only.

The members of the groups can be stored in DNS by using the reverse DNS resolution technique. It is not unusual that a given IP address refers to multiple FQDNs. Extrapolating to group names extends the reverse DNS resolution in a natural manner. Below the members of group lamp-fx.fx.bldg.org with IP address ff15::11 containing all four lamps is shown.

```
1.1.0.....0.5.1.f.f.IP6.arpa. IN PTR lm00206.rz.fx.bldg.org.
1.1.0.....0.5.1.f.f.IP6.arpa. IN PTR lm00205.rz.fx.bldg.org.
1.1.0.....0.5.1.f.f.IP6.arpa. IN PTR lm00204.ry.fx.bldg.org.
1.1.0.....0.5.1.f.f.IP6.arpa. IN PTR lm00203.ry.fx.bldg.org.
1.1.0.....0.5.1.f.f.IP6.arpa. IN PTR lamp-fx.fx.bldg.org.
```

With the above table queries like "return all members of lamp-fx" can be answered.

Additional tables are needed to specify the multicast groups to which the end-point of a device belongs. A PTR RR with the name of the Function Set can refer to the name of the group. In the table below the Function Set "1234lamp" is member of the groups "lamp-fx" and "lamp-ry":

```
1234lamp IN PTR lamp-fx.fx.bldg.org
1234lamp IN PTR lamp-ry.ry.fx.bldg.org
```

Consequently, a device can query DNS for the groups to which its Function Set belongs, and consecutively enable the reception of the associated multicast messages.

4.1.5. Discovery validation

This section describes how the disovery requirements are met with DNS-SD. It is assumed that the DNS server tables are correctly filled in.

Name_resolution: Group- or device-name is resolved to IP address by sending a query to DNS to return all AAAA records with the given name.

Return_server: Suppose the given service is defined by instance._sub._service.domain. All Function Sets supporting this service in the domain are found by sending a query to DNS to return all PTR records with the name instance._sub._service.domain. The returned PTR records refer to the names of the SRV/TXT pairs describing the port, FQDN and path of the associated Function Sets. Additional queries return all SRV and TXT records with the names returned by the first query. To query the end-points a query is sent to return all PTR records with name service.domain.

Create-Group: Groups are created by creating resource records in DNS as described in section 4.1.4. The filing of the DNS server tables is done by a trusted Remote Control or commissioning Device (see section 4.1.2). Section 6.1 discusses the security aspects.

Enroll-member, Remove-member: See Create-Group.

Return-Group: Suppose the given Function Set is given by the name instance._sub._bc._service.domain. All groups to which this Function Set belongs are found by sending a query to DNS to return all PTR records with the name instance._sub._bc._service.domain. The returned PTR records refer to the names of the associated groups. Name resolution provides the IP address.

Return-member: The members of a group with name group group.domain are found by resolving the group name to an IP address. The members of the group are obtained by sending a query to DNS to return all PTR records with as the name the inverted IP-address. The PRT records refer to the names of the associated devices.

4.2. RD examples

Resource discovery in CoAP handles resource paths (called links) for the resources hosted on the server, augmented with attributes of these resources. A well-known path "/.well-known/core" [RFC5785] is a default Function Set for requesting the list of links on a given server [I-D.shelby-core-resource-directory]. The Resource Directory (RD) stores links to resources hosted by other servers. The link-

format [I-D.ietf-core-link-format] defines link extensions to specify the service type and end-point as used by DNS-SD. When querying the Resource Directory for links, filters can be applied to return only links with specified attribute values. A node learns the IP-address of the RD by for example sending a multicast request to a predefined multicast address registered with IANA, or by assuming that the RD is located in the edge router.

Contrary to DNS-SD, the RD has not defined a process which permits SDOs to specify service (sub)types. Consequently, the same service type examples are used for DNS-SD as for RD, where service type postfixed with subtype (DNS) equals "resource type" (RD). The "host" name of DNS is used as "end-point" name for RD.

This draft adheres to the mapping between DNS and link-format, described in [I-D.lynn-core-discovery-mapping].

4.2.1. Commissioning devices

It is assumed that either a Remote Control (RC) in the home or a Commissioning Tool (CT) in the professional domain is used to fill in the RD. Devices cannot enter links into the RD contrary to the suggestion in [I-D.shelby-core-resource-directory]. Both the RC or the CT have to fill in the RD, because at link creation the RD generates a location of the link-entry (e.g. /ed/453), that is returned to the creator of the link. Consequently, the CT or the RD need to create the link, because they need the location to update the link with additional information.

In the professional domain, the CT learns the identity of the device as described earlier, reads the services of the devices with a GET to /.well-known/core on the device, and stores them into the RD.

In the home domain, the RC can read in the EUI-64 of the device to be entered. The user types in domain names and device names on the RC. Consecutively, the RC follows the same procedure as the CT.

4.2.2. Device examples

For convenience, it is assumed that DNS contains the mapping from FQDN to IP address with AAAA RRs and vice-versa with PTR RRs. In all examples the FQDN is used and not the IP-address. It is assumed that the service type "_bc" has the subtypes: "lamp", "sensor", "power",

"presence", and "timer". Registration is done with the following statements by CT or RC to the RD with authority: //rd.example.com/ and path /rd. Each POST command to the RD defines an end-point in the URI and defines the corresponding Function Sets in the payload.

```
POST coap://rd.example.com/rd?ep=lm00203;d=ry.fx.bldg.org
Etag: 0x21
Payload:
</lamp>;rt="lamp._sub._bc",
</light>;rt="sensor._sub._bc"
Res: 2.01 created
Location: /rd/1234
```

The other end-points are defined with (leaving out the response):

```
POST coap://rd.example.com/rd?ep=lm00204;d=ry.fx.bldg.org
Payload:
</lamp>;rt="lamp._sub._bc",
</light>;rt="sensor._sub._bc"

POST coap://rd.example.com/rd?ep=lm00205;d=rz.fx.bldg.org
Payload:
</lamp>;rt="lamp._sub._bc",
</light>;rt="sensor._sub._bc"

POST coap://rd.example.com/rd?ep=lm00206;d=rz.fx.bldg.org
Payload:
</lamp>;rt="lamp._sub._bc",
</light>;rt="sensor._sub._bc"
```

```
POST coap://rd.example.com/rd?ep=ps0057;d=ry.fx.bldg.org
Payload:
</ps>;rt="power._sub._bc",
</ps/1>;rt="power._sub._bc",
</ps/2>;rt="power._sub._bc",
</ps/3>;rt="power._sub._bc",
</ps/4>;rt="power._sub._bc"

POST coap://rd.example.com/rd?ep=ps0058;d=rz.fx.bldg.org
Payload:
</ps>;rt="power._sub._bc",
</ps/1>;rt="power._sub._bc",
</ps/2>;rt="power._sub._bc",
</ps/3>;rt="power._sub._bc",
</ps/4>;rt="power._sub._bc"

POST coap://rd.example.com/rd?ep=pir1;d=ry.fx.bldg.org
Payload:
</occup>;rt="presence._sub._bc"

POST coap://rd.example.com/rd?ep=pir2;d=rz.fx.bldg.org
Payload:
</occup>;rt="presence._sub._bc"

POST coap://rd.example.com/rd?ep=clock;d=fx.bldg.org
Payload:
</time>;rt="timer._sub._bc"
```

Update or removal of the groups is done by using the returned location (not shown above) as described in [I-D.shelby-core-resource-directory] for the end-points.

4.2.3. Group examples

The same five multicast groups of the DNS example are used. The group name to address mapping is specified in DNS with AAAA RRs. The group name is not easily identified with an end-point. Therefore the mnemonic gp is added as link-format attribute. The group members are included by citing the end-point names, which allows a unique identification of the members. In all examples the group name is used and not the IP-address. Registration is done with the following statements by CT or RC to the RD with authority: /rd.example.com/ and path /rd, leaving out Etag:, Res:, and Location: lines. The group name is defined in the URI, while all the members (end-points) are specified in the payload. The path in the payload defines together with the end-point the Function Set.

```
POST coap://rd.example.com/rd?gp=lamp-fx;d=fx.bldg.org
Payload:
</lamp>;rt="lamp._sub._bc";ep=lm00206,
</lamp>;rt="lamp._sub._bc";ep=lm00205,
</lamp>;rt="lamp._sub._bc";ep=lm00204,
</lamp>;rt="lamp._sub._bc";ep=lm00203

POST coap://rd.example.com/rd?gp=lamp-ry;d=ry.fx.bldg.org
Payload:
</lamp>;rt="lamp._sub._bc";ep=lm00204,
</lamp>;rt="lamp._sub._bc";ep=lm00203

POST coap://rd.example.com/rd?gp=lamp-rz;d=rz.fx.bldg.org
Payload:
</lamp>;rt="lamp._sub._bc";ep=lm00206,
</lamp>;rt="lamp._sub._bc";ep=lm00205

POST coap://rd.example.com/rd?gp=power-fx;d=fx.bldg.org
Payload:
</ps>;rt="power._sub._bc";ep=ps0057,
</ps>;rt="power._sub._bc";ep=ps0058

POST coap://rd.example.com/rd?gp=timer-bldg;d=bldg.org
Payload:
</time>;rt="timer._sub._bc";ep=clock
```

With the above statements the groups are defined in the RD.

4.2.4. Discovery validation

This section describes how the disovery requirements are met with the RD. It is assumed that the RD tables are correctly filled in.

Name_resolution: When LoWPAN is connected to backbone, it is assumed that DNS is used for name resolution. When LoWPAN is stand-alone without DNS server, IP addresses are used to connect to an interface.

Return_server: Suppose the given service is defined by instance._sub._service. The query

```
GET coap://rd.example.com/rd-lookup/res?rt=instance.
                                        _sub._service
Res: 2.05 content
<coap://name.domain/path>;rt="instance._sub._service"
```

returns all Function Sets providing the specified service. By changing the "res?" part in the query by "ep?" all end-points are returned.

Create-Group: Groups are created by sending POST commands to RD as described in section 4.2.2. The filling of the RD server tables is done by a trusted Remote Control or commissioning Device (see section 4.2.1). Section 6.1 discusses the security aspects.

Enroll-member, Remove-member: See Create-Group.

Return-Group: Suppose the given end-point is given by the name ep.domain. All groups to which this end-point belongs are found by sending a query

```
GET coap://rd.example.com/rd-lookup/gp?ep=ep.domain
Res: 2.05 content
<coap://name.domain/path>;gp="group.domain"
```

returns all groups of which the specified end-point, ep.domain, is a member.

Return-member: The members of a group with name group group.domain are found by the query

```
GET coap://rd.example.com/rd-lookup/ep?gp=group.domain
Res: 2.05 content
<coap://name.domain/path>;ep="ep.domain"
```

which returns all end-points which are member of the group gp.domain.

5. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

6. Security Considerations

Security considerations apply to use of DNS and RD separately, because they use different security mechanisms.

6.1. DNS Security considerations

Security extensions for DNS are provided by Domain Name System Security Extensions (DNSSEC) as described in [RFC4033], [RFC4034], and [RFC4035]. In particular [RFC4033] describes all documents relevant to DNSSEC. The central design decision of DNSSEC is to provide origin authentication and integrity protection for DNS data.

All transported DNS data are freely accessible. Consequently, all correctly functioning nodes will receive the domain and group names to which they belong, and will receive the addresses of the multicast and unicast destinations, and the multicast address of the groups to which they belong. Accordingly, commands will be sent to the intended destinations and accepted by the intended destinations. All resolvers MUST be configured with a security anchor. The anchor can be stored non-volatile memory or be provided by other out-of-band means.

Updating of DNS data, e.g,. zone data MUST be done using the secure dynamic updates mechanism. For example, a commissioning device is used to fill in the DNS server. Both the device sending the update requests (i.e., the commissioning device) and the DNS Server MUST share a Transaction Signature (TSIG) key [RFC2845]. The TSIG key is a symmetric-key and it can be generated using the dnssec-keygen primitive. The TSIG key is then used to sign the DNS update message, thus ensuring the authenticity of the request. An access control list can be defined in the DNS to authorize updates to the DNS data. The Berkeley Internet Name Daemon (BIND) implementation of the Internet System Consortium (ISC) provides a mechanism to specify access control lists, ensuring that updates of DNS zone data can only be performed by authorized parties. The allow-update option in a zone statement is used to control access to a zone. This option grants clients with a particular TSIG key the permission to update any record of any name in the zone. To allow for finer-granular access control, the update-policy option can be used within a zone statement, it specifies a set of rules where each rule either grants or denies permissions to one or more names to be updated by one or more identities. The identity can be determined based on the TSIG key in the TSIG record.

In the home and also during construction in the professional case, islands of security exist when the authorative DNS server has no connection to parent or children zones. In a later stage, access can be provided via DNS root servers involving a second security anchor. Alternatively, stub resolvers access contact recursive name servers via an secure channel as SIG(0) [RFC2931] or TSIG [RFC2845].

In spite of using DNSSEC, rogue devices can obtain valid addresses and accept commands for these destinations. This is not worse than rogue devices accepting any packet via a NIC in promiscuous mode. Rogue devices can send unwanted commands to the thus observed IP address. The same addresses can also be obtained by listening to the network traffic.

6.2. RD Security considerations

Security for Resource Direcory is provided by recommended CoAP security techniques: DTLS.

7. Acknowledgments

Zach Shelby wrote the original Discovery section in [I-D.ietf-core-coap] which forms the basis for this draft. This I-D has benefited from conversations with and comments from Emmanuel Frimout, Michael Verschoor, Jamie Mc Cormack, Esko Dijk, Dee Denteneer, Joop Talstra, Jerald Martocci, Matthieu Vial, Jerome Hamel, George Yianni, and Nicolas Riou.

Sye Loong Keoh, Sandeep Kumar, and Oscar Garcia Morchon contributed actively to security considerations.

8. Changelog

Changes from -02 to -03:

- adapted to resource direcory version -03
- DNS-SD examples follow better DNS-SD naming conventions
- added gp= link-format attribute
- added security considerations
- groups of end-points and groups of Function Sets instead of groups of devices
- less centered around DNS, more RD aspects

9. References

9.1. Normative References

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

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

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

[RFC2845] Vixie, P., Gudmundsson, O., Eastlake, D., and B. Wellington, "Secret Key Transaction Authentication for DNS (TSIG)", RFC 2845, May 2000.

[RFC2931] Eastlake, D., "DNS Request and Transaction Signatures (SIG(0)s)", RFC 2931, September 2000.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, March 2005.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, March 2005.

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC4605] Fenner, B., He, H., Haberman, B., and H. Sandick, "Internet Group Management Protocol (IGMP) / Multicast Listener Discovery (MLD)-Based Multicast Forwarding ("IGMP/MLD Proxying")", RFC 4605, August 2006.

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

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known Uniform Resource Identifiers (URIs)", RFC 5785, April 2010.

[RFC5988] Nottingham, M., "Web Linking", RFC 5988, October 2010.

[RFC6282] Hui, J. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, September 2011.

[RFC6206] Levis, P., Clausen, T., Hui, J., Gnawali, O., and J. Ko,

"The Trickle Algorithm", RFC 6206, March 2011.

9.2. Informative References

[I-D.cheshire-dnsext-dns-sd]
Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", draft-cheshire-dnsext-dns-sd-11 (work in progress), December 2011.

[I-D.eggert-core-congestion-control]
Eggert, L., "Congestion Control for the Constrained Application Protocol (CoAP)", draft-eggert-core-congestion-control-01 (work in progress), January 2011.

[I-D.ietf-6man-uri-zoneid]
Carpenter, B. and R. Hinden, "Representing IPv6 Zone Identifiers in Uniform Resource Identifiers", draft-ietf-6man-uri-zoneid-01 (work in progress), May 2012.

[I-D.ietf-core-coap]
Shelby, Z., Hartke, K., Bormann, C., and B. Frank, "Constrained Application Protocol (CoAP)", draft-ietf-core-coap-10 (work in progress), June 2012.

[I-D.ietf-core-groupcomm]
Rahman, A. and E. Dijk, "Group Communication for CoAP", draft-ietf-core-groupcomm-01 (work in progress), March 2012.

[I-D.ietf-core-link-format]
Shelby, Z., "CoRE Link Format", draft-ietf-core-link-format-14 (work in progress), June 2012.

[I-D.lynn-core-discovery-mapping]
Lynn, K. and Z. Shelby, "CoRE Link-Format to DNS-Based Service Discovery Mapping", draft-lynn-core-discovery-mapping-01 (work in progress), July 2011.

[I-D.lynn-homenet-site-mdns]
Lynn, K. and D. Sturek, "Extended Multicast DNS", draft-lynn-homenet-site-mdns-00 (work in progress), March 2012.

[I-D.shelby-core-coap-req]

Shelby, Z., Stuber, M., Sturek, D., Frank, B., and R. Kelsey, "CoAP Requirements and Features", draft-shelby-core-coap-req-02 (work in progress), October 2010.

[I-D.shelby-core-interfaces]
Shelby, Z. and M. Vial, "CoRE Interfaces", draft-shelby-core-interfaces-02 (work in progress), March 2012.

[I-D.shelby-core-resource-directory]
Shelby, Z. and S. Krco, "CoRE Resource Directory", draft-shelby-core-resource-directory-03 (work in progress), May 2012.

[I-D.vanderstok-core-bc]
Stok, P. and K. Lynn, "CoAP Utilization for Building Control", draft-vanderstok-core-bc-05 (work in progress), October 2011.

[I-D.jennings-http-srv]
Jennings, C., "DNS SRV Records for HTTP", draft-jennings-http-srv-05 (work in progress), March 2009.

[I-D.giacomin-core-sleepy-option]
Fossati, T., Giacomin, P., Loreto, S., and M. Rossini, "Sleepy Option for CoAP", draft-giacomin-core-sleepy-option-00 (work in progress), February 2012.

[I-D.vial-core-mirror-proxy]
Vial, M., "CoRE Mirror Proxy", draft-vial-core-mirror-proxy-00 (work in progress), March 2012.

[ZeroConf]
Cheshire, S. and D. Steinberg, "Zero Configuration Networking: The Definitive Guide", O'Reilly Media, Inc. , ISBN 0-596-10100-7, 2006.

[UPNP] "Universal Plug and Play", Web http://www.upnp.org, 2012.

Authors' Addresses

Peter van der Stok (editor)
vanderstok consultancy
Kamperfoelie 8
Helmond, 5708 DM
The Netherlands

Email: consultancy@vanderstok.org

Kerry Lynn
Consultant

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

Anders Brandt
Sigma Designs
Emdrupvej 26A, 1.
Copenhagen O, 2100
Denmark

Email: Anders_Brandt@sigmadesigns.com