Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 29, 2013

T. Clausen
A. Colin de Verdiere
J. Yi
LIX, Ecole Polytechnique
U. Herberg
Fujitsu Laboratories of America
Y. Igarashi
Hitachi, Ltd., Yokohama Research Laboratory

February 25, 2013

Observations of RPL: IPv6 Routing Protocol for Low power and Lossy Networks
draft-clausen-lln-rpl-experiences-06

Abstract

With RPL - the "IPv6 Routing Protocol for Low-power Lossy Networks" - having been published as a Proposed Standard after a ~2-year development cycle, this document presents an evaluation of the resulting protocol, of its applicability, and of its limits. The documents presents a selection of observations of the protocol characteristics, exposes experiences acquired when producing various prototype implementations of RPL, and presents results obtained from testing this protocol - by way of network simulations, in network testbeds and in deployments. The document aims at providing a better understanding of possible limits of RPL, notably the possible directions that further protocol developments should explore, in order to address these.

Table of Contents

1. Introduction

RPL - the "Routing Protocol for Low Power and Lossy Networks" [RFC6550] - is a proposal for an IPv6 routing protocol for Low-power Lossy Networks (LLNs), by the ROLL Working Group in the Internet Engineering Task Force (IETF). This routing protocol is intended to be the IPv6 routing protocol for LLNs and sensor networks, applicable in all kinds of deployments and applications of LLNs.

The objective of RPL and ROLL is to provide routing in networks which "comprise up to thousands of nodes" [roll-charter], where the majority of the nodes have very constrained resources [I-D.ietf-roll-terminology], and where handling mobility is not an explicit design criteria [RFC5867], [RFC5826], [RFC5673], [RFC5548].

[roll-charter] states that "Typical traffic patterns are not simply unicast flows (e.g. in some cases most if not all traffic can be point to multipoint)", and [I-D.ietf-roll-terminology] further categorizes the supported traffic types into "upward" traffic from sensors to a collection sink or LBR (LLN Border Router) (denoted multipoint-to-point), "downward" traffic from the collection sink or LBR to the sensors (denoted point-to-multipoint) and traffic from "sensor to sensor" (denoted point-to-point traffic), and establishes this terminology for these traffic types. Thus, while the target for RPL and ROLL is to support all of these traffic types, the emphasis among these, according to [roll-charter], appears to be to optimize for multipoint-to-point traffic, while also supporting point-to-multipoint and point-to-point traffic.

With approximately one year past since publication of RPL as [RFC6550], it is opportune to document observations of the protocol, in order to understand which aspects of it work well and which necessitate further investigations. Understanding possible limitations is important to identify issues which may restrict the deployment scope of the protocol and which may need further protocol work or enhancements.

The observations made in this document, except for when explicitly noted otherwise, do not depend on any specific implementation or deployment, but can be understood from simply analyzing the protocol specification [RFC6550]. That said, all observations made have been confirmed to also be present in, at least, some deployments or test platforms with RPL, i.e., have been experimentally confirmed.

This document is explicitly not an implementation guidebook for RPL. It has as objective to document observations of behaviors of [RFC6550], in the spirit of better understanding the characteristics and limits of the protocol.

2. Terminology

This document uses the terminology and notation defined in [RFC6550].

Additionally, this document uses terminology from [I-D.ietf-roll-terminology], specifically the terms defined for the traffic types "MP2P" (Multipoint-to-Point), "P2P" (Point To Point) and "P2MP" (Point-to-Multipoint).

Finally, this document introduces the following terminology:

RPL Router - A device, running the RPL protocol, as specified by [RFC6550].

3. RPL Overview

The basic construct in RPL is a "Destination Oriented Directed Acyclic Graph" (DODAG), depicted in Figure 1, with a single RPL Router acting as DODAG Root. The DODAG Root has responsabilities in addition to those of other RPL Routers, including for initiating, configuring, and managing the DODAG, and (in some cases) acting as a

central relay for traffic through and between RPL Routers in the LLN.

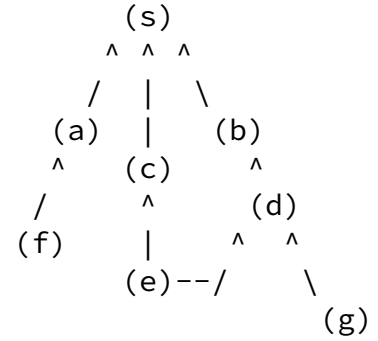

Figure 1: RPL DODAG

In an LLN, in which RPL has converged to a stable state, each RPL Router has identified a stable set of parents, each of which is a potential next-hop on a route towards the DODAG Root. One of the parents is selected as preferred parent. Each RPL Router, which is part of a DODAG (i.e., which has selected parents and a preferred parent) will emit DODAG Information Object (DIO) messages, using link-local multicast, indicating its respective rank in the DODAG (i.e., distance to the DODAG Root according to some metric(s), in the simplest form hop-count). Upon having received a (number of such) DIO messages, an RPL Router will calculate its own rank such that it is greater than the rank of each of its parents, select a preferred parent and then itself start emitting DIO messages.

DODAG formation thus starts at the DODAG Root (initially, the only RPL Router which is part of a DODAG), and spreads gradually to cover the whole LLN as DIOs are received, parents and preferred parents are selected, and further RPL Routers participate in the DODAG. The DODAG Root also includes, in DIO messages, a DODAG Configuration Object, describing common configuration attributes for all RPL Routers in that network - including their mode of operation, timer characteristics etc. RPL Routers in a DODAG include a verbatim copy of the last received DODAG Configuration Object in their DIO messages, permitting also such configuration parameters propagating through the network.

As a Distance Vector protocol, RPL restricts the ability for an RPL Router to change rank. An RPL Router can freely assume a smaller rank than previously advertised (i.e., logically move closer to the

DODAG Root) if it discovers a parent advertising a lower rank, and must then disregard all previous parents of ranks higher than the router's new rank. The ability for an RPL Router to assume a greater rank (i.e., logically move farther from the DODAG Root) than previously advertised is restricted in order to avoid count-to-infinity problems. The DODAG Root can trigger "global recalculation" of the DODAG by increasing a sequence number, DODAG version, in DIO messages.

The DODAG so constructed is used for installing routes: the "preferred parent" of an RPL Router can serve as a default route towards the DODAG Root, and the DODAG Root can embed in its DIO messages the destination prefixes, included by DIOs generated by RPL Routers through the LLN, to which connectivity is provided by the DODAG Root. Thus, RPL by way of DIO generation provides "upward routes" or "multipoint-to-point routes" from the sensors inside the LLN and towards the DODAG Root (and, possibly, to destinations reachable through the DODAG Root).

"Downward routes" are enabled by having sensors issue Destination Advertisement Object (DAO) messages, propagating as unicast via preferred parents towards the DODAG Root. These describe which prefixes belong to, and can be reached via, which RPL Router. In a network, all RPL Routers must operate in either of storing mode or non-storing mode, specified by way of a "Mode of Operation" (MOP) flag in the DODAG Configuration Object from the DODAG Root. Those two modes are non-interoperable, i.e., a mixture of RPL Routers running in different modes is impossible in the same routing domain. Depending on the MOP, DAO messages are forwarded differently towards the DODAG Root:

- In "non-storing mode", an RPL Router originates a DAO messages, advertising one or more of its parents, and unicasts these to the

DODAG Root. Once the DODAG Root has received DAOs from an RPL Router, and from all RPL Routers on the route between it and the DODAG Root, it can use source routing for reaching advertised destinations inside the LLN.

- In "storing mode", each RPL Router on the route between the originator of a DAO and the DODAG Root records a route to the prefixes advertised in the DAO, as well as the next-hop towards

these (the RPL Router, from which the DAO was received), then forwards the DAO to its preferred parent.

"Point-to-point routes", for communication between devices inside the LLN and where neither of the communicating devices are the DODAG Root, are as default supported by having the source sensor transmit a data packet, via its default route to the DODAG Root (i.e., using the upward routes), which will then, depending on the "Mode of Operation" for the DODAG, either add a source-route to the received data packet for reaching the destination sensor (downward routes in non-storing mode), or simply use hop-by-hop routing (downward routes in storing mode) for forwarding the data packet. In the case of storing mode, if the source and the destination for a point-to-point data packet share a common ancestor other than the DODAG Root, a downward route may be available in an RPL Router (and, thus, used) before the data packet reaches the DODAG Root.

3.1. RPL Message Emission Timing - Trickle Timers

RPL message generation is timer-based, with the DODAG Root being able to configure back-off of message emission intervals using Trickle [RFC6206]. Trickle, as used in RPL, stipulates that an RPL Router transmits a DIO "every so often" - except if receiving a number of DIOs from neighbor RPL Routers, enabling the RPL Router to determine if its DIO transmission is redundant.

When an RPL Router transmits a DIO, there are two possible outcomes: either every neighbor RPL Router that hears the message finds that the information contained is consistent with its own state (i.e., the received DODAG version number corresponds with that which the RPL Router has recorded, and no better rank is advertised than that which is recorded in the parent set) - or, a recipient RPL Router detects that either the sender of the DIO or itself has out-of-date information. If the sender has out-of-date information, then the recipient RPL Router schedules transmission of a DIO to update this information. If the recipient RPL Router has out-of-date information, then it updates based on the information received in the DIO.

With Trickle, an RPL Router will schedule emission of a DIO at some

time, t, in the future. When receiving a DIO containing information

consistent with its own information, the RPL Router will record that "redundant information has been received" by incrementing a redundancy counter, c. At the time t, if c is below some "redundancy threshold", then it transmits its DIO. Otherwise, transmission of a DIO at this time is suppressed, c is reset and a new t is selected to twice as long time in the future - bounded by a pre-configured maximum value for t. If, on the other hand, the RPL Router has received an out-of-date DIO from one of its neighbors, t is reset to a pre-configured minimum value and c is set to zero. In both cases, at the expiration of t, the RPL Router will verify if c is below the "redundancy threshold" and if so transmit - otherwise, increase t and stay quiet.

4. Requirement Of DODAG Root

As indicated in Section 3, the DODAG Root has both a special responsibility and is subject to special requirements. The DODAG Root is responsible for determining and maintaining the configuration parameters for the DODAG, and for initiating DIO emissions.

The DODAG Root is also responsible (in both storing and non-storing mode) for being able to, when downward routes are supported, maintain sufficient topological information to be able to construct routes to all destinations in the network.

When operating in non-storing mode, this entails that the DODAG Root is required to have sufficient memory and sufficient computational resources to be able to record a network graph containing all routes from itself and to all destinations and to calculate routes.

When operating in storing mode, this entails that the DODAG Root needs enough memory to keep a list of all RPL Routers in the RPL instance, and a next hop for each of those RPL Routers. If aggregation is used, the memory requirements can be reduced in storing mode (see Section 8 for observations about aggregation in RPL).

The DODAG Root is also required to have sufficient energy available so as to be able to ensure the relay functions required. This, especially for non-storing mode, where all data packets transit through the DODAG Root.

4.1. Observations

In a given deployment, select RPL Routers can be provisioned with the required energy, memory and computational resources so as to serve as

DODAG Roots, and be administratively configured as such - with the remainder of the RPL Routers in the network being of typically lesser capacity. In storing mode, the DODAG root needs to keep a routing entry for all RPL Routers in the RPL instance. In non-storing mode, the resource requirements on the DODAG Root are likely much higher than in storing mode, as the DODAG Root needs to store a network graph containing complete routes to all destinations in the RPL instance, in order to calculate the routing table (whereas in storing mode, only the next hop for each destination in the RPL instance needs to be stored, and aggregation may be used to further reduce the resource requirements).

RPL Routers provisioned with resources to act as DODAG Roots, and administratively configured to act as such, represent a single point of failure in the network. As the memory requirements for the DODAG Root and for other RPL Routers are substantially different, unless all RPL Routers are provisioned with resources (memory, energy, ...) to act as DODAG Roots, effectively if the designated DODAG Root fails, the network fails and RPL is unable to operate. Even if electing another RPL Router as temporary DODAG Root (e.g., for forming a "Floating" DODAG) for providing internal connectivity between RPL Routers, this RPL Router may not have the necessary resources to satisfy this role as (temporary) DODAG Root.

Thus, although in principle RPL provides, by way of "Floating DODAGs", protocol mechanisms for establishing a DODAG for providing internal connectivity even in case of failure of the administratively provisioned DODAG Root, all (or at least a large number) of the RPL routers need to have resources to act as roots to support floating DODAG, especially in non-storing mode.

Another possible LLN scenario is that only internal point-to-point connectivity is sought, and no RPL Router has a more "central" role than any other - a self-organizing LLN. In those cases, it would be hard to specify such "super-device" as DODAG root, and can result in non-optimal routes.

5. RPL Data Traffic Flows

RPL makes a-priori assumptions of data traffic types, and explicitly defines three such [I-D.ietf-roll-terminology] traffic types: sensor-to-root data traffic (multipoint-to-point) is predominant, root-to-sensor data traffic (point-to-multipoint) is rare and sensor-to-sensor (point-to-point) data traffic is extremely rare. While not specifically called out thus in [RFC6550], the resulting protocol

design, however, reflects these assumptions in that the mechanism constructing multipoint-to-point routes is efficient in terms of

control traffic generated and state required, point-to-multipoint route construction much less so - and point-to-point routes subject to potentially significant route stretch (routes going through the DODAG Root in non-storing mode) and over-the-wire overhead from using source routing (from the DODAG Root to the destination) (see Section 7) - or, in case of storing mode, considerable memory requirements in all LLN routers inside the network (see Section 7).

An RPL Router selects from among its parents a "preferred parent", to serve as a default route towards the DODAG Root (and to prefixes advertised by the DODAG Root). Thus, RPL provides "upward routes" or "multipoint-to-point routes" from the RPL Routers below the DODAG Root and towards the DODAG Root.

An RPL Router which wishes to act as a destination for data traffic ("downward routes" or "point-to-multipoint") issues DAOs upwards in the DODAG towards the DODAG Root, describing which prefixes belong to, and can be reached via, that RPL Router.

Point-to-Point routes between RPL Routers below the DODAG Root are supported by having the source RPL Router transmit, via its default route, data traffic towards the DODAG Root. In non-storing mode, the data traffic will reach the DODAG Root, which will reflect the data traffic downward towards the destination RPL Router, adding a strict source routing header indicating the precise route for the data traffic to reach the intended destination RPL Router. In storing mode, the source and the destination may possibly (although, may also not) have a common ancestor other than the DODAG Root, which may provide a downward route to the destination before data traffic reaching the DODAG Root.

5.1. Observations

RPL is suited for networks where sensor-to-root traffic is dominante, by distribution of DIO messages and building of a collection tree. The one way traffic from the sensor to the root can be forwarded through the preferred parent.

However, the data traffic characteristics, assumed by RPL, do not

represent a universal distribution of traffic types in LLNs:

- There are scenarios where sensor-to-sensor traffic is a more common occurrence, documented, e.g., in [RFC5867] ("Building Automation Routing Requirements in Low Power and Lossy Networks").

- There are scenarios, where all traffic is bi-directional, e.g., in case sensor devices in the LLN are, in majority, "actively read": a request is issued by the DODAG Root to a specific sensor, and

the sensor value is expected returned. In fact, unless all traffic in the LLN is unidirectional, without acknowledgements (e.g., as in UDP), and no control messages (e.g., for service discovery) or other data packets are sent from the DODAG Root to the RPL Routers, traffic will be bi-directional. The IETF protocol for use in constrained environments, CoAP [I-D.ietf-core-coap], makes use of acknowledgements to control packet loss and ensure that packets are received by the packet destination. In the four message types defined for CoAP: confirmable, acknowledgement, reset and non-confirmable, the first three are dedicate for sending/acknowledgement cycle. Another example is that the ZigBee Alliance SEP 2.0 specification [SEP2.0] describes the use of HTTP over TCP over ZigBeeIP, between RPL Routers and the DODAG Root - and with the use of TCP inherently causing bidirectional traffic by way of data-packets and their corresponding acknowledgements. In fact, current Internet protocols generally require some form of acknowledgment, and foregoing an acknowledgment probably means a trade-off in the area of reliable transmission or repeated retransmissions or both.

For the former, all sensor-to-sensor routes include the DODAG Root, possibly causing congestions on the communication medium near the DODAG Root, and draining energy from the intermediate RPL Routers on an unnecessarily long route. If sensor-to-sensor traffic is common, RPL Routers near the DODAG Root will be particularly solicited as relays, especially in non-storing mode.

For the latter, as there is no provision for on-demand generation of routing information from the DODAG Root to a proper subset of all RPL Routers, each RPL Router (besides the Root) is required to generate DAOs. In particular in non-storing mode, each RPL Router will unicast a DAO to the DODAG Root (whereas in storing mode, the DAOs

propagate upwards towards the Root). The effects of the requirement to establish downward routes to all RPL Routers are:

- Increased memory and processing requirements at the DODAG Root (in particular in non-storing mode) and in RPL Routers near the DODAG Root (in storing mode).

- A considerable control traffic overhead [bidir], in particular at and near the DODAG Root, therefore:

- Potentially congested channels, and:

- Energy drain from the RPL Routers.

6. Fragmentation Of RPL Control Messages And Data Packet

Link layers, used in LLNs, are often unable to provide an MTU of, at least, 1280 octets - as otherwise required for IPv6 [RFC2460]. In such LLNs, link-specific fragmentation and reassembly of IP packets at a layer below IPv6 is used to transport larger IP packets, providing the required minimum 1280 octet MTU [RFC4919].

When such below-the-IP-layer fragmentation is used, the IP packet has to be reassembled at every hop. Every fragment must be received successfully by the receiving device, or the entire IP packet is lost. Moreover, the additional link-layer frame overhead (and IPv6 Fragment header overhead in case of IP fragmentation) for each of the fragments increases the capacity required from the medium, and may consume more energy for transmitting a higher number of frames on the network interface.

RPL is an IPv6 routing protocol, designed to operate on constrained link layers, such as [ieee802154], with a maximum frame size of 127 bytes - a much smaller value than the specified minimum MTU of 1280 bytes for IPv6 [RFC2460]. Reducing the need of fragmentation of IP datagrams on such a link layer, 6LoWPAN provides an adaptation layer [RFC4944], [RFC6282], providing "Layer 2.5 fragmentation" in order to accommodate IPv6 packet transmissions over the maximum IEEE 802.15.4 frame size of 127 octets, as well as compressing the IPv6 header,

reducing the overhead of the IPv6 header from at least 40 octets to a minimum of 2 octets. Given the IEEE 802.15.4 frame size of 127 octets, a maximum frame overhead of 25 octets and 21 octets for link layer security [RFC4944], 81 octets remain for L2 payload. Further subtracting 2 octets for the compressed IPv6 header leaves 79 octets for L3 data payload if link-layer fragmentation is to be avoided.

The second L in LLN indicating Lossy [roll-charter], higher loss rates than typically seen in IP networks are expected, rendering fragmentation important to avoid. This, in particular because, as mentioned above, the whole IP packet is dropped if only a single fragment is lost.

6.1. Observations

[RFC4919] makes the following observation regarding using IP in LoWPAN networks based on IEEE 802.15.4 frames:

> Applications within LoWPANs are expected to originate small packets. Adding all layers for IP connectivity should still allow transmission in one frame, without incurring excessive fragmentation and reassembly. Furthermore, protocols must be designed or chosen so that the individual "control/protocol

> packets" fit within a single 802.15.4 frame. Along these lines, IPv6's requirement of sub-IP reassembly [...] may pose challenges for low-end LoWPAN devices that do not have enough RAM or storage for a 1280-octet packet.

In order to avoid the link-layer fragmentation and thus to adhere to the recommendation in [RFC4919], each control packet of RPL must fit into the remaining 79 octets of the 802.15.4 frame. While 79 octets may seem to be sufficient to carry RPL control messages, consider the following: RPL control messages are carried in ICMPv6, and the mandatory ICMPv6 header consumes 4 octets. The DIO base another 24 octets. If link metrics are used, that consumes at least another 8 octets - and this is when using a simple hop count metric; other metrics may require more. The DODAG Configuration Object consumes up to a further 16 octets, for a total of 52 octets. Adding a Prefix Information Object for address configuration consumes another 32 octets, for a total of 84 octets - thus exceeding the 79 octets available for L3 data payload and causing link-layer fragmentation of

such a DIO. As a point of reference, the ContikiRPL [rpl-contiki] implementation includes both the DODAG Configuration option and the Prefix Information option in all DIO messages. Any other options, e.g., Route Information options indicating prefixes reachable through the DODAG Root, increase the overhead and thus the probability of fragmentation.

RPL may further increase the probability of link-layer fragmentation of data traffic: for non-storing mode, RPL employs source-routing for all downward traffic. [RFC6554] specifies the RPL Source Routing header, which imposes a fixed overhead of 8 octets per IP packet leaving 71 octets remaining from the link-layer MTU in order to contain the whole IP packet into a single frame - from which must be deducted a variable number of octets, depending on the length of the route. With fewer octets available for data payload, RPL thus increases the probability for link-layer fragmentation of also data packets. This, in particular, for longer routes, e.g., for point-to-point data traffic between sensors inside the LLN, where data traffic transit through the DODAG Root and is then source-routed to the destination. The overhead of source routing is further detailed in Section 7.

Given the minimal packet size of LLNs, the routing protocol must impose low (or no) overhead on data packets, hopefully independently of the number of hops [RFC4919]. However, source-routing not only causes increased overhead in the IP header, it also leads to a variable available payload for data (depending on how long the source route is). In point-to-point communication and when non-storing mode is used for downward traffic, the source of a data packet will be unaware of how many octets will be available for payload (without

incurring L2.5 fragmentation) when the DODAG Root relays the data packet and adds the source routing header. Thus, the source may choose an inefficient size for the data payload: if the data payload is large, it may exceed the link-layer MTU at the DODAG Root after adding the source-routing header; on the other hand, if the data payload is low, the network resources are not used efficiently, which introduces more overhead and more frame transmissions.

Unless the DODAG Root is the source of an IPv6 packet to be forwarded through an RPL LLN, the IPv6 packet must be encapsulated in IPv6-in-IPv6 tunneling, with the RPL extension added to the outer IPv6

header. Similarly, in non-storing mode, the original IPv6 packet must be carried in IPv6-in-IPv6 tunneling, with the RPL routing header added to the outer IPv6 header. Both of these mechanisms add additional overhead, increasing the likelihood that link-layer fragmentation will be required to deliver the IPv6 packet. In addition, even IPv6 packets that are the minimum MTU size of 1280 octets will require IPv6 fragmentation to accommodate the RPL tunnel and headers on a deployment using the [RFC4944] specification to carry IPv6 over IEEE 802.15.4, because RFC4944 defines the MTU for such deployments to be 1280 octets. The ZigBee Alliance is considering relaxing [RFC4944] to use an MTU of 1360 octets in its specification for IPv6 over IEEE 802.15.4 to accommodate 1280 octet IPv6 packets with the required tunnel overhead without fragmentation.

7. The DAO Mechanism: Downward and Point-to-Point Routes

RPL specifies two distinct and incompatible "modes of operation" for downward traffic: storing mode, where each RPL Router is assumed to maintain routes to all destinations in its sub-DODAG, i.e., RPL Routers that are "deeper down" in the DODAG, and non-storing mode, where only the DODAG Root stores routes to destinations inside the LLN, and where the DODAG Root employs strict source routing in order to route data traffic to the destination RPL Router.

7.1. Observations

In addition to possible fragmentation, as occurs when using potentially long source routing headers over a medium with a small MTU - similar to what is discussed in Section 6 - the maximum length of the source routing header [RFC6554] is limited to 136 octets, including an 8 octet long header. As each IPv6 address has a length of 16 octets, not more than 8 hops from the source to the destination are possible for "raw IPv6". Using address compression (e.g., as specified in [RFC4944]), the maximum route length may not exceed 64 hops. This excludes deployment of RPL for scenarios with long "chain-like" topologies, such as traffic lights along a street.

In storing mode, each RPL Router has to store routes for destinations in its sub-DODAG. This implies that, for RPL Routers near the DODAG Root, the required storage is only bounded by the number of destinations in the network. As RPL targets constrained devices with

little memory, but also has as ambition to be operating networks consisting of thousands of routers [roll-charter], the storing capacity on these RPL Routers may need to be the same as DODAG root - or, at least, the storage requirements in RPL Routers "near the DODAG Root" and "far from the DODAG Root" is not homogenous, thus some sort of administrative deployment, and continued administrative maintenance of devices, as the network evolves, is needed.

In an experimental testbed, [rpl-eval-UCB] argues that practical experiences suggest that RPL in storing mode, with RPL Routers having 10kB of RAM (TELOSB mote with TinyOS, 16-bit RISC, 48 kB program flash memory, 16 kB configuration EEPROM), should be limited to networks of less than ~30 RPL Routers. Note that observation of less than 30 RPL Routers only presents the results obtained from specified testbed and implementation in [rpl-eval-UCB]. Aggregation / summarization of addresses may be advanced as a possible argument that this issue is of little significance - Section 8 discusses why such an argument does not apply. Moreover, if the LoWPAN adaption layer [RFC4944] is used in the LLN, route aggregation is not possible since the same /64 is applied across the entire network.

In short, the mechanisms in RPL force the choice between requiring all RPL Routers to have sufficient memory to store route entries for all destinations (storing mode) - or, suffer increased risk of fragmentation, and thus loss of data packets, while consuming network capacity by way of source routing through the DODAG Root (non-storing mode).

In RPL, the "mode of operation" stipulates that either downward routes are not supported (MOP=0), or that they are supported by way of either storing or non-storing mode. In case downward routes are supported, RPL does not provide any mechanism for discriminating between which routes should or should not be maintained. In particular, in order to calculate routes to a given destination, all intermediaries between the DODAG Root and that destination must themselves be reachable - effectively rendering downward routes in RPL an "all-or-none" situation. In case a destination is unreachable, all the DODAG Root may do is increase DTSN (Destination Advertisement Trigger Sequence Number) to trigger DAO message transmission, or eventually increase the DODAG version number in case the destination is still unreachable, which possibly provokes a broadcast-storm-like situation. This, in particular, as [RFC6550] does not specify DAO message transmission constraints, nor any mechanism for adapting DAO emission to the network capacity.

In storing mode, a DTSN increment by the DODAG Root works only if all RPL Routers, on the path from the DODAG Root to the "lost" target RPL Router, have kept their routing table up-to-date by triggering DAO updates, and thus have a route to the target RPL Router. In non-storing mode, the DODAG Root incrementing its DTSN will trigger global DAO updates, and thus extra overhead in the network and delay in the recalculation of the missing route.

Furthermore, DTSN increments are carried by way of DIO messages. In case the "lost" target RPL Router has lost all of its parents, it will not be able to receive DIO messages from them, and thus will have to wait until it has poisoned its sub-DODAG and joined the DODAG through another parent. The only way the DODAG Root can speed up this process is by incrementing the DODAG version number, thus triggering global recalculation of the DODAG.

Even in case the DTSN increment is carried to the "lost" target RPL Router through another parent, the triggered DAO will need to go up the DODAG to the DODAG Root via another route, which might itself be broken. This would necessitate the use of local repair mechanisms, potentially causing loops in the network (see Section 14) and eventually global DODAG recalculation.

8. Address Aggregation and Summarization

As indicated in Section 7, in storing mode, an RPL Router is expected to be able to store routing entries for all destinations in its "sub-DODAG", i.e., routing entries for all destinations in the network where the route to the DODAG Root includes that RPL Router.

In the Internet, no single router stores explicit routing entries for all destinations. Rather, IP addresses are assigned hierarchically, such that an IP address does not only uniquely identify a network interface, but also its topological location in the network, as illustrated in Figure 2. All addresses with the same prefix are reachable by way of the same router - which can, therefore, advertise only that prefix. Other routers need only record a single routing entry for that prefix, knowing that as the IP packet reaches the router advertising that prefix, more precise routing information is available.

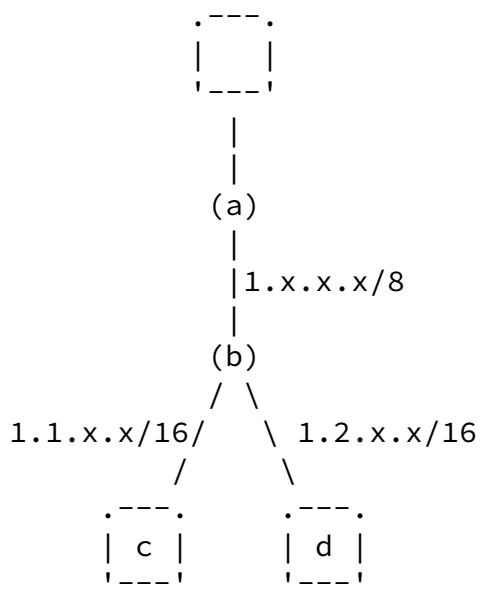

Figure 2: Address Hierarchies

Any aggregated routes require the use of a prefix shorter than /64, and subsequent hierarchical assignment of prefixes down to a /64 (as any RPL Router itself provides a /64 subnet to any hosts connected to the RPL Router).

Moreover, if the 6lowpan adaption layer [RFC4944] is used in the LLN, route aggregation is not possible since the same /64 is applied across the entire network.

8.1. Observations

In RPL, each RPL Router acquires a number of parents, as described in Section 3, from among which it selects one as its preferred parent and, thus, next-hop on the route to the DODAG Root. RPL Routers maintain a parent set containing possibly more than a single parent so as to be able to rapidly select an alternative preferred parent, should the previously selected such become unavailable. Thus expected behavior is for an RPL Router to be able to change its point of attachment towards the DODAG Root. If IP addresses are assigned in a strictly hierarchical fashion, and if scalability of the routing

state maintained in storing mode is based on this hierarchy, then this entails that each time an RPL Router changes its preferred parent, it must also change its own IP address - as well as cause RPL Routers in its "sub-DODAG" to do the same. RPL does not specify signaling for reconfiguring addresses in a sub-DODAG, while [RFC6550] specifically allows for aggregation (e.g., in Section 18.2.6.: "[...] it is recommended to delay the sending of DAO message to DAO parents in order to maximize the chances to perform route aggregation").

A slightly less strict hierarchy can be envisioned, where an RPL Router can change its preferred parent without necessarily changing addresses of itself and of its sub-DODAG, provided that its former and new preferred parents both have the same preferred parent, and have addresses hierarchically assigned from that - from the "preferred grandparent". With reference to Figure 1, this could be e changing its preferred parent from d to c, provided that both d and c have b as preferred parent. Doing so would impose a restriction on the parent-set selection, admitting only parents which have themselves the same parent, losing redundancy in the network connectivity. RPL does not specify rules for admitting only parents with identical grand-parents into the parent set - although such is not prohibited either, if the loss of redundancy is acceptable.

The DODAG Root incrementing the DODAG version number is the mechanism by which RPL enables global reconfiguration of the network, reconstructing the DODAG with (intended) more optimal routes. In case of addressing hierarchies being enforced, so as to enable aggregation, this will either restrict the ability for an optimal DODAG construction, or will also have to trigger global address autoconfiguration so as to ensure addressing hierarchies.

Finally, with IP addresses serving a dual role of an identifier of both an end-point for communication and a topological location in the network, changing the IP address of a device, so as to reflect a change in network topology, also entails interrupting ongoing communication to or through that device. Additional mechanisms (e.g., a DNS-like system) mapping "communications identifiers" and "IP addresses" are required.

9. Link Bidirectionality Verification

Parents (and the preferred parent) are selected based on receipt of DIOs. This, alone, does not guarantee the ability of an RPL Router to successfully communicate with the parent. However, the basic use of links is for "upward" routes, i.e., for the RPL Router to use a parent (the preferred parent) as relay towards the DODAG Root - in the opposite direction of the one in which the DIO was received.

9.1. Observations

Unidirectional links are no rare occurrence, such as is known from wireless multi-hop networks. Preliminary results from a test-bed of AMI (Automated Metering Infrastructure) devices using 950MHz radio interfaces, and with a total of 22 links, show that 36% of these links are unidirectional. If an RPL Router receives a DIO on such a unidirectional link, and selects the originator of the DIO as parent,

which would be a bad choice: unicast traffic in the upward direction would be lost. If the RPL Router had verified the bidirectionality of links, it might have selected a better parent, to which it has a bidirectional link.

[RFC6550] discusses some mechanisms which can (if deemed needed) be used to verify that a link is bidirectional before choosing an RPL Router as a parent. While requiring one mechanism for bidirectional verification to be used, the document does not specify which method to be used, and how to be used. The mechanisms discussed include NUD [RFC4861], BFD [RFC5881] and [RFC5184]. BFD is explicitly called out as "often not desirable" as it uses a proactive approach (exchange of periodic HELLO messages), and thus would "lead to excessive control traffic". Furthermore, not all L2 protocols provide L2 acknowledgements; even less so for multicast packets - and so, not on RPL DIOs, the multicast transmission of which is a requirement for the Trickle timer flooding reduction to be effective (see Section 3.1). This has as consequence that such L2 acknowledgements can only be used to determine if a given link is bidirectional or unidirectional once the RPL Router already has selected parents AND actually has data traffic to forward by way of these parents - in contradiction with RPL's stated design principle that require that the reachability of an RPL Router be verified before choosing it as a parent ([RFC6550], Section 1.1). Absent any mechanism specified by RPL to verify the bidirectionality of links, RPL Routers thus have to rely on NUD to choose their parent correctly (see Section 10).

10. Neighbor Unreachability Detection For Unidirectional Links

[RFC6550] suggests using Neighbor Unreachability Detection (NUD) [RFC4861] to detect and recover from the situation of unidirectional links between an RPL Router and its (preferred) parent(s). When, e.g., an RPL Router tries (and fails) to actually use another RPL Router for forwarding traffic, NUD is supposed engaged to detect and prompt corrective action, e.g., by way of selecting an alternative preferred parent.

NUD is based upon observing if a data packet is making forward progress towards the destination, either by way of indicators from upper-layer protocols (such as TCP and, though not called out in [RFC4861], also from lower-layer protocols such as Link Layer ACKs) or - failing that - by unicast probing by way of transmitting a unicast Neighbor Solicitation message and expecting that a solicited Neighbor Advertisement message be returned.

10.1. Observations

An RPL Router may receive, transiently, a DIO from an RPL Router, closer (in terms of rank) to the DODAG Root than any other RPL Router from which a DIO has been received. Some, especially wireless, link layers may exhibit different transmission characteristics between multicast and unicast transmissions (such is the case for some implementations of IEEE 802.11b, where multicast/broadcast transmissions are sent at much lower bit-rates than are unicast; IEEE 802.11b is, of course, not suggested as a viable L2 for LLNs, but serves to illustrate that such asymmetric designs exist), leading to a (multicast) DIO being received from farther away than a unicast transmission can reach. DIOs are sent (downward) using link-local multicast, whereas the traffic flowing in the opposite direction (upward) is unicast. Thus, a received (multicast) DIO may not be indicative of useful unicast connectivity - yet, RPL might cause this RPL Router to select this seemingly attractive RPL Router as its preferred parent. This may happen both at initialization, or at any time during the LLN lifetime as RPL allows attachment to a "better

parent" over the network lifetime.

A DODAG so constructed may appear stable and converged until such time that unicast traffic is to be sent and, thus, NUD invoked. Detecting only at that point that unicast connectivity is not maintained, and causing local (and possibly global) repairs exactly at that time, may lead to traffic not being deliverable. As indicated in Section 8, if scalability is dependent on addresses being assigned hierarchically, changing point-of-attachment may entail more than switching preferred parent.

An RPL Router may detect that its preferred parent is lost by way of NUD, when trying to communicate to the DODAG Root. If that RPL Router has no other parents in its parent set, all it can do is wait: RPL does not provide other mechanisms for an RPL Router to react to such an event. In the case where there is no downward traffic (i.e., no data or acknowledgements are sent from the DODAG Root), neither the DODAG Root nor the preferred parent, to which upward connectivity was lost, will be able to detect and react to the event of connectivity loss.

In other words, for upward traffic, the RPL Routers that by way of NUD detect connectivity loss, will be unable to act in order to restore connectivity (e.g., by way of a signaling mechanism to the DODAG Root, to request DODAG reconstruction by way of version number increase). The RPL Routers, which could react (the "preferred parents") will for upward traffic not generate any traffic "downward" allowing NUD to engage and detect connectivity loss.

It is worth noting that RPL is optimized for upward traffic (multipoint-to-point traffic), and that this is exactly the type of traffic where NUD is not applicable as a mechanism for detecting and reacting to connectivity loss.

Also, absent all RPL Routers consistently advertising their reachability through DAO messages, a protocol requiring bidirectional flows between the communicating devices, such as TCP or CoAP confirmable-acknowledgement exchange, will be unable to operate.

Finally, upon having been notified by NUD that the "next hop" is unreachable, an RPL Router must discard the preferred parent and

select another - hoping that this time, the preferred parent is actually reachable. Also, if NUD indicates "no forward progress" based on an upper-layer protocol, there is no guarantee that the problem stems exclusively from the preferred parent being unreachable. Indeed, it may be a problem further ahead, possibly outside the LLN, thus changing preferred parent will not alleviate the situation. Moreover, using information from an upper-layer protocol, e.g., to return TCP ACKs back to the source, requires established downward routes in the DODAG (i.e., each RPL Router needs to send DAO messages to the DODAG Root, as described in Section 7).

Incidentally, this stems from a fundamental difference between "fixed links in the Internet" and "wireless links": whereas the former, as a rule, are reliable, predictable and with losses being rare exceptions, the latter are characterized by frequent losses and general unpredictability.

11. RPL Implementability and Complexity

RPL is designed to operate on "RPL Routers [...] with constraints on processing power, memory, and energy (battery power)" [RFC6550]. However, the 163 pages long specification of RPL, plus additional specifications for routing headers [RFC6554], Trickle timer [RFC6206], routing metrics [RFC6551] and objective function [RFC6552], describes complex mechanisms (e.g., the upwards and downward data traffic, a security solution, manageability of RPL Routers, auxiliary functions for autoconfiguration of RPL Routers, etc.), and provides no less than 9 message types, and 10 different message options.

To give one example, the ContikiRPL implementation (http://www.sics.se/contiki), which provides only storing mode and no security features, consumes about 50 KByte of memory. Sensor hardware, such as MSP430 sensor platforms, does not contain much more memory than that, i.e., there may not be much space left to deploy

any application on the RPL Router.

11.1. Observations

Since RPL is intended as the routing protocol for LLNs, which covers

all the diverse applications requirements listed in [RFC5867], [RFC5673], [RFC5826], [RFC5548], it is likely that (i) due to limited memory capacity of the RPL Routers, and (ii) due to expensive development cost of the routing protocol implementation, RPL implementations will only support a partial set of features from the specification, leading to non-interoperable implementations.

In order to accommodate for the verbose exchange format, route stretching and source routing for point-to-point traffic, several additional Internet-Drafts are being discussed for adoption in the ROLL Working Group - adding complexity to an already complex specification which, it is worth recalling, was intended to be of a protocol for low-capacity devices.

12. Underspecification

While [RFC6550] provides various options and extensions in many parts, which makes a complex protocol, as described in Section 11, some mechanisms are underspecified.

While for DIOs, the Trickle timer specifies a relatively efficient and easy-to-understand timing for message transmission, the timing of DAO transmission is not explicit. As each DAO may have a limited lifetime, one "best guess" for implementers would be to send DAO periodically, just before the life-time of the previous DAO expires. Since DAOs may be lost, another "best guess" would be to send several DAOs shortly one after the other in order to increase probability that at least one DAO is successfully received.

The same underspecification applies for DAO-ACK messages: optionally, on reception of a DAO, an RPL Router may acknowledge successful reception by returning a DAO-ACK. Timing of DAO-ACK messages is unspecified by RPL.

12.1. Observations

By not specifying details about message transmission intervals and required actions when receiving DAO and DAO-ACKs, implementations may exhibit a bad performance if not carefully implemented. Some examples are:

1. If DAO messages are not sent in due time before the previous DAO expires (or if the DAO is lost during transmission), the routing entry will expire before it is renewed, leading to a possible data traffic loss.

2. RPL does not specify to use jitter [RFC5148] (i.e., small random delay for message transmissions). If DAOs are sent periodically, adjacent RPL Routers may transmit DAO messages at the same time, leading to link layer collisions.

3. In non-storing mode, the "piece-wise calculation" of routes to a destination from which a DAO has been received, relies on previous reception of DAOs from intermediate RPL Routers along the route. If not all of these DAOs from intermediate RPL Routers have been received, route calculation is not possible, and DAO-ACKs or data traffic cannot be sent to that destination.

Other examples of underspecification include detection of connectivity loss, as described in Section 10, as well as the local repair mechanism, which may lead to loops and thus data traffic loss, if not carefully implemented: an RPL Router discovering that all its parents are unreachable, may - according to the RPL specification - "detach" from the DODAG, i.e., increase its own rank to infinity. It may then "poison" its sub-DODAG by advertising its infinite rank in its DIOs. If, however, the RPL Router receives a DIO before it transmits the "poisoned" DIO, it may attach to its own sub-DODAG, creating a loop. If, instead, it had waited some time before processing DIOs again, chances are it would have succeeded in poisoning its sub-DODAG and thus avoided the loop.

13. Protocol Convergence

Trickle [RFC6206] is used by RPL to schedule transmission of DIO messages, with the objective of minimizing the amount of transmitted DIOs while ensuring a low convergence time of the network. The theoretical behavior of Trickle is well understood, and the convergence properties are well studied. Simulations of the mechanism, such as documented [trickle-multicast], confirm these theoretical studies.

In real-world environments, however, varying link qualities may cause the algorithm to converge less well: frequent message losses entail resets of the Trickle timer and more frequent and unpredicted message emissions.

13.1. Observations

The varying link quality in real-world environments results in frequent changes of the best parent, which triggers a reset of the Trickle timer and thus the emission of DIOs. Therefore Trickle does not converge as well for links that are fluctuating in quality as in theory.

This has been observed, e.g., in an experimental testbed: 69 RPL Routers (MSP430-based wireless sensor routers with IEEE 802.15.4, using [rpl-contiki] IPv6 stack and RPL without downward routes; the parameters of the Trickle timer were set to the implementation defaults (minimum DIO interval: 4 s, DIO interval doublings: 8, redundancy constant: 10) were positioned in a fixed grid topology. This resulted in DODAGs being constructed with an average of 2.45 children per RPL Router and an average rank of 3.58.

In this small test network, the number of DIO messages emitted - expectedly - spiked within the first ~10 seconds. Alas, rather than taper off to become zero (as the simulation studies would suggest), the DIO emission rate remained constant at about 70 DIOs per second. Details on this experiment can be found in [rpl-eval].

The resulting higher control overhead due to frequent DIO emission, leads to higher bandwidth and energy consumption as well as possibly to an increased number of collisions of frames, as observed in [trickle-multicast].

13.2. Caveat

Note that these observations do not claim that it is impossible to parametrize Trickle timers so that a given deployment exhibits the theoretical characteristics (or, characetristics sufficiently close thereto) of the Trickle mechanism. These observations suggest that the default parameter values, provided for Trickle timers in [RFC6550], did not apply to the small network tested. These observations also suggest that special care is required when selecting the values for the parameters for Trickle timers, and that these values likely are to be determined experimentally, and individually for each deployment.

14. Loops

[RFC6550] states that it "guarantees neither loop free route selection nor tight delay convergence times, but can detect and repair a loop as soon as it is used. RPL uses this loop detection to ensure that packets make forward progress [...] and trigger repairs

when necessary". This implies that a loop may only then be detected and fixed when data traffic is sent through the network.

In order to trigger a local repair, RPL relies on the "direction" information (with values "up" or "down"), contained in an IPv6 hop-by-hop option header from received a data packet. If an "upward" data packet is received by an RPL Router, but the previous hop of the packet is listed with a lower rank in the neighbor set, the RPL Router concludes that there must be a routing loop and it may therefore trigger a local repair. For downward traffic in non-storing mode, the DODAG Root can detect loops if the same RPL Router identifier (i.e., IP address) appears at least twice in the route towards a destination.

14.1. Observations

The reason for RPL to repair loops only when detected by a data traffic transmission is to reduce control traffic overhead. However, there are two problems in repairing loops only when so triggered: (i) the triggered local repair mechanism delays forward progress of data packets, increasing end-to-end delays, and (ii) the data packet has to be buffered during repair.

(i) may seem as the lesser of the two problems, since in a number of applications, such as data acquisition in smart metering applications, an increased delay may be acceptable. However, for applications such as alarm signals or in home automation (e.g., a light switch), increased delay may be undesirable.

As for (ii), RPL is supposed to run on LLN routers with "constraints on [...] memory" [RFC6550]; buffering incoming packets during the route repair may not be possible for all incoming data packets, leading to dropped packets. Depending on the transport protocol, these data packets must be retransmitted by the source or are definitely lost.

If carefully implemented with respect to avoiding loops before they occur, the impact of the loop detection in RPL may be minimized. However, it can be observed that with current implementations of RPL, such as the ContikiRPL implementation, loops do occur - and, frequently. During the same experiments described in Section 13, a snapshot of the DODAG was made every ten seconds. In 74.14% of the 4114 snapshots, at least one loop was observed. Further investigation revealed that in all these cases the DODAG was partitioned, and the loop occurred in the sub-DODAG that no longer had a connection to the DODAG Root. When the link to the only parent of an RPL Router breaks, the RPL Router may increase its rank and - when receiving a DIO from an RPL Router in its sub-DODAG - attach

itself to its own sub-DODAG, thereby creating a loop - as detailed in Section 12.1.

While it can be argued that the observed loops are harmless since they occur in a DODAG partition that has no connection to the DODAG Root, they show that the routes are not built correctly. Even worse, when the broken link re-appears, it is possible that in certain situations, the loop is only repaired when data traffic is sent, possibly leading to data loss (as described above). This can occur if the link to the previous parent is reestablished, but the rank of that previous parent has increased in the meantime.

Another problem with the loop repair mechanism arises in non-storing mode when using only downward traffic: while the DODAG Root can easily detect loops (as described above), it has no direct means to trigger a local repair where the loop occurs. Indeed, it can only trigger a global repair by increasing the DODAG version number, leading to a Trickle timer reset and increased control traffic overhead in the network caused by DIO messages, and therefore a possible energy drain of the RPL Routers and congestion of the channel.

Finally, loop detection for every data packet increases the processing overhead. RPL is targeted for deployments on very constrained devices with little CPU power, therefore a loop detection for every packet reduces available resources of the LLN router for other tasks (such as sensing). Moreover, each IPv6 packet needs to contain the "RPL Option for Carrying RPL Information in Data-Plane

Datagrams" [RFC6553] in order to use loop detection (as well as determining the RPL instance), which in turn implies an extra IPv6 header (and thus overhead) for IPv6-in-IPv6 tunneling. As this RPL option is a hop-by-hop option, it needs to be in an encapsulating IPv6-in-IPv6 tunnel and then regenerated at each hop.

15. Security Considerations

This document does currently not specify any security considerations. This document also does not provide any evaluation of the security mechanisms of RPL.

16. IANA Considerations

This document has no actions for IANA.

17. Acknowledgements

The authors would like to thank Matthias Philipp (INRIA) for his contributions to conducting many of the experiments, revealing or confirming the issues described in this document.

Moreover, the authors would like to express their gratitude to Ralph Droms (Cisco) for his careful review of various versions of this document, for many long discussions, and for his considerable contributions to both the content and the quality of this document.

18. Informative References

[I-D.ietf-core-coap]
Shelby, Z., Hartke, K., Bormann, C., and B. Frank, "Constrained Application Protocol (CoAP)", draft-ietf-core-coap-13 (work in progress), December 2012.

[I-D.ietf-roll-terminology]
Vasseur, J., "Terminology in Low power And Lossy

Networks", draft-ietf-roll-terminology-11 (work in progress), February 2013.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, Decemer 1998.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC4919] Kushalnagar, N., Montenegro, G., and C. Schumacher, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals", RFC 4919, August 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.

[RFC5148] Clausen, T., Dearlove, C., and B. Adamson, "Jitter Considerations in Mobile Ad Hoc Networks (MANETs)", RFC 5148, February 2008.

[RFC5184] Aggarwal, R., Kompella, K., Nadeau, T., and G. Swallow, "Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5184, June 2010.

[RFC5548] Dohler, M., Watteyne, T., Winter, T., and D. Barthel, "Routing Requirements for Urban Low-Power and Lossy Networks", RFC 5548, May 2009.

[RFC5673] Pister, K., Thubert, P., Dwars, S., and T. Phinney, "Industrial Routing Requirements in Low-Power and Lossy Networks", RFC 5673, October 2009.

[RFC5826] Brandt, A., Buron, J., and G. Porcu, "Home Automation Routing Requirements in Low-Power and Lossy Networks", RFC 5826, April 2010.

[RFC5867] Martocci, J., Mi, P., Riou, N., and W. Vermeylen, "Building Automation Routing Requirements in Low Power and

Lossy Networks", RFC 5867, June 2010.

[RFC5881] Ward, D. and D. Katz, "Bidirectional Forwarding Detection (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881, June 2010.

[RFC6206] Levis, P., Clausen, T., Gnawali, O., and J. Ko, "The Trickle Algorithm", RFC 6206, March 2011.

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

[RFC6550] Winther, T., Thubert, P., Hui, J., Vasseur, J., Brandt, A., Kelsey, R., Levis, P., Piester, K., Struik, R., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, March 2012.

[RFC6551] Vasseur, J., Pister, K., Dejan, N., and D. Barthel, "Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks", RFC 6551, March 2012.

[RFC6552] Thubert, P., "Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6552, March 2012.

[RFC6553] Hui, J. and J. Vasseur, "The Routing Protocol for Low-Power and Lossy Networks (RPL) Option for Carrying RPL Information in Data-Plane Datagrams", RFC 6553, March 2012.

[RFC6554] Hui, J., Vasseur, J., Culler, D., and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6554,

March 2012.

[SEP2.0] Alliance, Zigbee., "ZigSmart Energy version 2.0 (SEP 2.0) draft 0.7", July 2011.

[bidir] Clausen, T. and U. Herberg, "A Comparative Performance Study of the Routing Protocols LOAD and RPL with Bi-

Directional Traffic in Low-power and Lossy Networks (LLN)", Proceedings of the Eighth ACM International Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN), 2011.

[ieee802154]
Computer Society, IEEE., "IEEE Std. 802.15.4-2003", October 2003.

[roll-charter]
"ROLL Charter", web http://datatracker.ietf.org/wg/roll/charter/, February 2012.

[rpl-contiki]
Tsiftes, N., Eriksson, J., and A. Dunkels, "Low-Power Wireless IPv6 Routing with ContikiRPL", Proceedings Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (ISPN), 2011.

[rpl-eval]
Clausen, T., Herberg, U., and M. Philipp, "A Critical Evaluation of the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL)", Proceedings of the 5th IEEE International Conference on Wireless & Mobile Computing, Networking & Communication (WiMob), 2011.

[rpl-eval-UCB]
Ko, J., Dawson-Haggerty, S., Culler, D., and A. Terzis, "Evaluating the Performance of RPL and 6LoWPAN in TinyOS", Proceedings of the Workshop on Extending the Internet to Low power and Lossy Networks (IP+SN), 2011.

[trickle-multicast]
Clausen, T. and U. Herberg, "Study of Multipoint-to-Point and Broadcast Traffic Performance in the 'IPv6 Routing Protocol for Low Power and Lossy Networks' (RPL)", Journal of Ambient Intelligence and Humanized Computing, 2011.

Authors' Addresses

Thomas Clausen
LIX, Ecole Polytechnique
91128 Palaiseau Cedex,
France

Phone: +33 6 6058 9349
Email: T.Clausen@computer.org
URI: http://www.thomasclausen.org

Axel Colin de Verdiere
LIX, Ecole Polytechnique
91128 Palaiseau Cedex,
France

Phone: +33 6 1264 7119
Email: axel@axelcdv.com
URI: http://www.axelcdv.com/

Jiazi Yi
LIX, Ecole Polytechnique
91128 Palaiseau Cedex,
France

Phone: +33 1 6933 4031
Email: jiazi@jiaziyi.com
URI: http://www.jiaziyi.com/

Ulrich Herberg
Fujitsu Laboratories of America
1240 E Arques Ave
Sunnyvale, CA 94085
USA

Email: ulrich@herberg.name
URI: http://www.herberg.name/

Yuichi Igarashi
Hitachi, Ltd., Yokohama Research Laboratory

Phone: +81 45 860 3083
Email: yuichi.igarashi.hb@hitachi.com
URI: http://www.hitachi.com/