

ROLL Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: October 11, 2009

P. Thubert  
Cisco  
T. Watteyne  
UC Berkeley  
Z. Shelby  
Sensinode  
D. Barthel  
Orange Labs  
April 9, 2009

LLN Routing Fundamentals  
draft-thubert-roll-fundamentals-01

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on October 11, 2009.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents in effect on the date of publication of this document (<http://trustee.ietf.org/license-info>). Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Internet-Draft

LLN Routing Fundamentals

April 2009

## Abstract

This document describes a basic set of fundamental mechanisms for routing on a Low-power and Lossy Network (LLN). It does not intend to specify a full-blown protocol. It is rather offered as a basis to support the discussion while designing the ROLL protocol.

## Table of Contents

<a href="#">1.</a>	<a href="#">Introduction . . . . .</a>	<a href="#">4</a>
<a href="#">1.1.</a>	<a href="#">Terminology . . . . .</a>	<a href="#">5</a>
<a href="#">1.2.</a>	<a href="#">Needs . . . . .</a>	<a href="#">6</a>
<a href="#">2.</a>	<a href="#">Tree Discovery . . . . .</a>	<a href="#">7</a>
<a href="#">2.1.</a>	<a href="#">Overview . . . . .</a>	<a href="#">8</a>
<a href="#">2.2.</a>	<a href="#">Discovery Information . . . . .</a>	<a href="#">10</a>
<a href="#">2.3.</a>	<a href="#">Stability . . . . .</a>	<a href="#">11</a>
<a href="#">3.</a>	<a href="#">Route Dissemination . . . . .</a>	<a href="#">11</a>
<a href="#">3.1.</a>	<a href="#">Overview . . . . .</a>	<a href="#">11</a>
<a href="#">3.2.</a>	<a href="#">Disseminated Information . . . . .</a>	<a href="#">12</a>
<a href="#">3.3.</a>	<a href="#">LLN Router Operation . . . . .</a>	<a href="#">13</a>
<a href="#">4.</a>	<a href="#">Forwarding . . . . .</a>	<a href="#">15</a>
<a href="#">4.1.</a>	<a href="#">Upstream Forwarding . . . . .</a>	<a href="#">15</a>
<a href="#">4.2.</a>	<a href="#">Downstream Forwarding . . . . .</a>	<a href="#">17</a>
<a href="#">5.</a>	<a href="#">Multicast Support . . . . .</a>	<a href="#">18</a>
<a href="#">5.1.</a>	<a href="#">Overview . . . . .</a>	<a href="#">18</a>
<a href="#">5.2.</a>	<a href="#">Receiver Flow . . . . .</a>	<a href="#">18</a>
<a href="#">5.3.</a>	<a href="#">Source flow . . . . .</a>	<a href="#">19</a>
<a href="#">6.</a>	<a href="#">Advanced Features . . . . .</a>	<a href="#">19</a>
<a href="#">6.1.</a>	<a href="#">Interaction with other routing protocols . . . . .</a>	<a href="#">19</a>
<a href="#">6.1.1.</a>	<a href="#">AODV/DYMO . . . . .</a>	<a href="#">19</a>
<a href="#">6.1.2.</a>	<a href="#">OSPF/OLSR . . . . .</a>	<a href="#">20</a>
<a href="#">6.1.3.</a>	<a href="#">MIP6/NEMO . . . . .</a>	<a href="#">21</a>
<a href="#">6.2.</a>	<a href="#">Route Optimization . . . . .</a>	<a href="#">21</a>
<a href="#">6.2.1.</a>	<a href="#">Node-to-node routing . . . . .</a>	<a href="#">21</a>
<a href="#">6.2.2.</a>	<a href="#">Offline Path Computation . . . . .</a>	<a href="#">21</a>
<a href="#">6.2.3.</a>	<a href="#">Graph forwarding . . . . .</a>	<a href="#">22</a>
<a href="#">6.3.</a>	<a href="#">Density . . . . .</a>	<a href="#">23</a>
<a href="#">6.4.</a>	<a href="#">Digraph Dissemination . . . . .</a>	<a href="#">24</a>
<a href="#">6.5.</a>	<a href="#">Multiple LBRs and Trees . . . . .</a>	<a href="#">24</a>
<a href="#">6.6.</a>	<a href="#">Aggregation for Route Dissemination . . . . .</a>	<a href="#">24</a>
<a href="#">6.7.</a>	<a href="#">Advanced Forwarding . . . . .</a>	<a href="#">25</a>
<a href="#">7.</a>	<a href="#">Security Considerations . . . . .</a>	<a href="#">25</a>

<a href="#">8.</a>	IANA Considerations . . . . .	<a href="#">26</a>
<a href="#">9.</a>	Acknowledgments . . . . .	<a href="#">26</a>
<a href="#">10.</a>	References . . . . .	<a href="#">26</a>
<a href="#">10.1.</a>	Normative References . . . . .	<a href="#">26</a>
<a href="#">10.2.</a>	Informative References . . . . .	<a href="#">26</a>

Thubert, et al.	Expires October 11, 2009	[Page 2]
-----------------	--------------------------	----------

---

Internet-Draft	LLN Routing Fundamentals	April 2009
----------------	--------------------------	------------

Authors' Addresses . . . . .	<a href="#">28</a>
------------------------------	--------------------

## 1. Introduction

This document describes a basic set of fundamental mechanisms for routing on a Low-power and Lossy Network (LLN) appropriate for scenarios identified by the ROLL working group. It does not intend to specify a full-blown protocol. It is rather offered as a basis to support the discussion while designing the ROLL protocol. The fundamental mechanisms proposed stem from our analysis that current academic, industrial and IETF protocols suitable to ROLL scenarios are reduceable to those basic mechanisms.

Those mechanisms provide a core set of functionality that can be complemented by specific extensions to implement the needs expressed in the ROLL routing requirement drafts:

- o Urban WSNs Routing Requirements in Low Power and Lossy Networks [[I-D.ietf-roll-urban-routing-reqs](#)]
- o Building Automation Routing Requirements in Low Power and Lossy Networks [[I-D.ietf-roll-building-routing-reqs](#)]
- o Home Automation Routing Requirements in Low Power and Lossy Networks [[I-D.ietf-roll-home-routing-reqs](#)]
- o Industrial Routing Requirements in Low Power and Lossy Networks [[I-D.ietf-roll-indus-routing-reqs](#)]

The constraints expressed in the routing requirement documents (such as on node memory and communication cost) narrow the choice of

fundamental mechanisms down to very simple ones.

Due to the highly directed flows in LLNs, a tree structure comes naturally to mind as a bare minimum. In a slightly more elaborate mechanism, we propose that each router memorizes a few best neighbor routers (not only among its parents up the tree, but also among its siblings), to choose from (using some routing metric) when routing towards LLN Border Routers (LBR). However, to reduce complexity, we propose that only the best parent be advertised up the structure towards the LBRs, giving each of them a simple tree representation to be used for routing downstream traffic or for making other global decisions. Since links and nodes are expected to come and go over time, mechanisms for tree reorganization are described. However, on a shorter time scale, transient link failures are bound to happen. In such a case, we recommend that the link-layer passes packets back to the network layer for re-routing along alternate paths.

In terms of routing, the basic fundamental methods include uni/anycast routing up the graph and unicast routing down the tree

(either hop-by-hop or source-based). The best neighbor selection mechanism is left to the protocol design phase. We even suggest that it be left as a plug-in for future evolution. However, a set of basic tree discovery and forwarding rules, described here, prevents loops from forming, in most cases, whatever the routing algorithm eventually implemented.

More advanced mechanisms which can be built upon the fundamental mechanisms are also described. They include route optimizations, dissemination of a digraph, dissemination and maintenance of multiple overlapping trees, prefix aggregation and advanced forwarding rules.

This document is organized as follows:

[Section 1.1](#) defines the terminology used in this document.

[Section 2](#) concentrates on the basic tree discovery and maintenance mechanism.

[Section 3](#) introduces the basic distance-vector route dissemination mechanism.

[Section 4](#) describes the upstream and downstream forwarding rules.

[Section 5](#) describes multicast support.

[Section 6](#) describes advanced mechanisms which can be built upon these fundamentals.

### 1.1. Terminology

The terminology used in this document is consistent with and incorporates that described in [[I-D.ietf-roll-terminology](#)]. This terminology is extended in this document as follows:

to Attach: the action of establishing a child-to-parent relationship in Tree Discovery.

Tree Depth: the maximum number of edges that need to be traversed from any tree node to the root.

Discovery: a mechanism by which a logical representation of the network is built.

Floating, Grounded: a tree is said to be Grounded if it is connected to a high-capacity backbone or backhaul link to a network such as the Internet. By contrast, a tree is said to be Floating if it is not Grounded.

Graph: a set of vertices and edges to represent a network of nodes and links. A Directed Acyclic Graph (DAG) is a graph with directional edges where no loop is formed.

Uniform Path Metric: A scalar measure for the quality of the bi-directional path between the LLN Router and the root.

Route Dissemination: the action of establishing state within the network so that routers know how to forward packets related to some source-destination pairs.

Router: a network node that is capable of forwarding packets on behalf of other nodes. In ROLL routing requirement documents, it appears that most nodes are expected to be routers.

Default Router: the router to turn to when a node has no information on where to forward a packet.

## [1.2.](#) Needs

The ROLL working group has identified typical scenarios and their related requirements for LLN routing. The main requirements on any fundamental mechanisms used for achieving the ROLL protocol can be summarized as follows:

- o Support for operation in both full IPv6 [[RFC2460](#)] and minimal 6LoWPAN [[RFC4944](#)] networks.
- o Optimized for traffic directed between nodes and LBRs.
- o The discovery of multiple disjoint routing paths to increase reliability.
- o Support for multiple LBRs out of the LLN.
- o Minimal network state needed by routers, with a hard bound better than  $O(D)$ ,  $D$  being the number of destinations.
- o Support for complex unicast, anycast and multicast flows.
- o Localized response upon link failures without requiring global updates.
- o Minimal control overhead scaling within  $O(\log(L))$  of the data rate.
- o Support for link and node costs along routes.

## [2.](#) Tree Discovery

A tree is the simplest and most basic acyclic graph structure. Even if it is not sufficient to ensure by itself the multipath forwarding proposed below, a tree provides the ideal structure for best path routing between source and sink in a convergecast.

In many occasions, LLNs do not have a clear and stable physical

structure and it becomes necessary to overlay a logical representation to define links and enable IPv6 operations. LLN Tree Discovery is the component of the LLN fundamentals that builds and maintains logical tree structures over the LLN.

The nodes in an LLN discovery tree are Routers; the root is an arbitrary elected Router if the tree is Floating; it is a LLN Border Router (LBR) if the tree is Grounded, that is the root is connected to the infrastructure via a backhaul link or a federating backbone.

A federating backbone such as an extended LoWPAN backbone is the virtual root of the federated tree. In that case, the LBRs are attached at a depth of one and are in charge of performing the root operations on behalf of that virtual root.

A tree is identified by a Tree ID which can take the form of an IPv6 address: in the case of a LoWPAN configuration with a federating backbone, the LoWPAN prefix is used as the Tree ID. If there is no backbone, the tree ID will be an address of the root or a prefix owned by the root. A router attaching to a tree sets a route to the treeID via its parent in the tree.

A router may attach to and may advertise more than one tree, but it uses and advertises at most one tree as Default tree. A router sets up its default route via its parent in its Default tree.

This section describes

1. a minimum extension to IPv6 Neighbor Discovery Router Advertisements in order to ensure that LLN Routers organize in a tree structure, and
2. a minimum common algorithmic part that all LLN Routers are required to implement in order to ensure that, whatever the individual routing decisions, routing loops between LLN Routers are avoided and basic optimization is achieved.

LLN Discovery is based on an autonomous decision by each Router with no global state convergence such as traditionally found in IGP. In order to enable backward compatibility and interoperability, LLN

Discovery allows Routers to make different decisions from identical



inputs, based on their own configuration and their own algorithms, though it is highly preferable that the decision algorithm be consistent in a given deployment to achieve the specific goals of that deployment.

The signalling mechanism that is used to form the trees is an extension to the ICMP Router Advertisement (RA) message, namely the Tree Information Option (TIO). The TIO allows LLN Routers to advertise the tree they belong to, and to select and move to the best location within the available trees. LLN Routers propagate the TIO in RA messages down the tree, updating some metrics such as the Tree Depth while leaving other information such as the Tree ID unchanged. This is compatible with RA period reduction techniques such as the use of Trickle.

## [2.1.](#) Overview

LLN Tree Discovery is a form of distance vector protocol for use in wireless meshed networks. Tree Discovery locates the nearest exit and forms Directed Graphs towards that exit, composed of a best path tree and alternate forwarding options.

By introducing the concept of routing plug-ins, LLN Tree Discovery enables LLN Routers to implement different policies for selecting their preferred parent in the Tree. Tree Discovery does not specify the plug-in operation, but rather specifies a set of rules to be implemented by all plug-ins to ensure interoperability.

The Tree Depth is the underlying criterion that guarantees loop-free operations even if plug-ins implement different policies, and even if these policies do not use Depth as a routing metric.

In order to organize and maintain a loopfree structure, the parent selection plug-ins in the LLN Routers MUST obey the following rules and definitions:

1. The root of a tree exposes the tree in the Router Advertisement (RA) Tree Information Option (TIO) and LLN Routers propagate the TIO down.
2. An LLN Border Router that is attached to a federating backbone acts as root and advertises a depth of one. An LBR that is not attached to a federating backbone is a root and exposes a depth of zero.
3. An LLN Router that is not a Border Router may be the root of its own Floating tree. Its depth is zero in that tree. An LLN

---

Router that loses its current parent and has no alternate parent is back to that same state, but it needs to remember the Tree ID and the sequence counter in the TIO of the lost parent for a period of time which covers multiple TIOs.

4. An LLN Router announces its tree as Default in a TIO unless it also announces its participation to another tree that it uses as Default. An LBR announces its tree as Default and sets up its default route over the backhaul or the backbone. An LLN router that attaches to a tree that is announced as Default may select that tree as Default in which case it will propagate the Default information in the TIO for that tree and set up a default route via its parent in that tree. If the route attaches to other trees that are also announced as Default, it will reset the Default for the corresponding TIOs.
5. A router sending an RA without TIO is considered a Grounded Default Router at depth 0.
6. An LLN Router that is already part of a tree MAY move at any time and with no delay in order to get closer to the root of its current tree, i.e. in order to reduce its own tree depth. But an LLN Router MUST NOT move down the tree that it is attached to unless the potential parent advertises a Sequence Number that is newer than the last Sequence Number known for that tree, indicating that the potential parent is not within this router subtree.
7. A LLN Router may move from its current default tree into any different default tree at any time and whatever the depth it reaches in the new tree but, before it can do so, it may have to wait for a Tree Hop Timer to elapse. If the router was root of its own floating tree, it may join its previous tree (identified by the last parent Tree ID) only if the sequence number in the TIO was incremented since the LLN Router left that tree, indicating that the candidate parent was not attached behind this LLN Router and kept getting subsequent TIOs from the same tree. The LLN Router will join that other tree if it is preferable for reasons of connectivity, configured preference, available medium time, size, security, bandwidth, tree depth, or whatever metrics the LLN Router cares to use.
8. If an LLN Router has selected a new parent router but has not moved yet (because it is waiting for Tree Hop Timer to elapse), it is said to be unstable and refrains from sending Router Advertisement - Tree Information Options.

9. When a LLN Router joins a tree, moves within its own tree or receives a modified TIO from its current parent router, it sends out an unsolicited Router Advertisement message with TIO that propagates the new tree information.
10. This allows the new higher parts of the tree to be updated first, eventually dragging their sub-tree with them, and allowing stepped sub-tree reconfigurations, limiting relative movements.

## [2.2.](#) Discovery Information

The Tree Information Option carries a number of metrics and other information that allows an LLN Router to discover a tree and select its parent while avoiding loop generation.

### TIO Base option

The Tree Information Option is a container option, which might contain a number of suboptions. The base option regroups the minimum information set that is mandatory to operate the LLN Discovery Algorithm.

**Default (D):** The Default (D) flag is set when the tree is used to set up the default route. A router that participates to multiple trees (including self-rooted) announces at most one tree as Default.

**Grounded (G):** The Grounded (G) flag is set when the tree is attached to a fixed network infrastructure (such as the Internet).

**Sequence Number:** An integer that is incremented by the root for each TIO sent on a link. It is propagated unchanged down the tree.

**Tree Depth:** If the root is attached to a federating backbone, its Tree Depth is 1, otherwise it is 0. The Tree Depth of an LLN Router is the depth of its parent as received in a TIO,

incremented by at least one. All the nodes in the tree advertise their Tree Depth in the Tree Information Options that they append to the RA messages as part of the propagation process.

**Tree ID:** An IPv6 address which uniquely identifies a tree. This value is set by the root to one of its ULA or global addresses or prefixes.

**Uniform Path Metric:** A scalar measure for the quality of the bi-directional path between the LLN Router and the root.

The following values MUST not change during the propagation of the TIO down the tree: G, Sequence Number, Tree Delay and Tree ID. The Default flag MAY only be reset. All other fields are updated at each hop of the propagation.

In addition to the minimum set of information required, a number of options can be used, e.g. for bandwidth, stability, preference etc.

### [2.3.](#) Stability

An LLN Router is instable when it is prepared to move shortly to another parent Router. This happens typically when the LLN Router has selected a more preferred candidate parent Router and has to wait for the Tree Hop Timer to elapse before roaming. Instability may also occur when the current parent Router is lost and the next best one is still held up. Instability is resolved when the Tree Hop Timer of all the parent Router(s) causing instability elapse.

Instability is transient (on the order of Tree Hop Timers). When an LLN Router is unstable, it MUST NOT send RAs with TIO. This reduces the likelihood of loops when LLN Router A wishes to attach to LLN Router B and LLN Router B wishes to attach to LLN Router A. Unless RAs crisscross, a LLN Router only receives TIO from stable parent Routers, which do not plan to attach to it, so it can safely attach to one of them.

## [3.](#) Route Dissemination

### [3.1.](#) Overview

Route Dissemination is the second component of the LLN fundamental mechanisms. As explained previously, the first component, LLN Tree Discovery, establishes a logical tree structure over the LLN and sets up default routes towards the root of its Default Tree. To establish the routing states towards the nodes in the LLN and enable complete reachability along the tree, it suffices for Route Dissemination to advertise up the tree the host ID, prefix and multicast routes.

As a result, the Default Router for an LLN Router is its parent up in the Default tree (upstream); and the more specific routes are always oriented down the tree (downstream).

LLN Tree Discovery does not only provide loop avoidance for the Route

Thubert, et al.

Expires October 11, 2009

[Page 11]

---

Internet-Draft

LLN Routing Fundamentals

April 2009

Dissemination protocol; LLN Tree Discovery also triggers Route Dissemination each time a topological change occurs. The loopfree structure must be restored before Route Dissemination can operate again and repaint the tree with prefixes, addresses and group membership.

Each logical tree that LLN Tree Discovery forms is considered a separate routing topology. If an LLN Router belongs to multiple of such topologies, then it is expected that both the Route Dissemination signaling and the data packets are flagged to follow the topology for which the packet was introduced in the network.

The ROLL Route Dissemination protocol defines a new information vector called the Route Information Option (RIO) to disseminate atomic routing information towards the root of the tree.

A parent maintains a state for each information it learns from Route Dissemination. Advertisements are sequenced and the last sequence number is kept. An out-of-sequence RIO must be disregarded. If the RIO information appears valid, it is forwarded to the parent's parent in the next burst, carried by a RIO, together with the parent's own information.

### [3.2.](#) Disseminated Information

Route Dissemination extends [RFC4861](#) and [RFC4191](#) to allow a node to include a new Route Information Option in ND messages such as Neighbor Advertisements (NAs).

In order to track the freshness of an advertisement, the RIO includes a sequence counter that is incremented each time the advertisement is reissued.

An NA is also sent to the new parent once it has been selected after a movement, or when the list of advertised information has changed.

Route Dissemination may advertise positive (prefix is present) or negative (removed) RIOs.

The RIO base option carries sequenced route information for unicast and multicast; it contains:

Resource type: Prefix, host, or multicast group

Prefix Length: Number of valid leading bits in the IPv6 Prefix.

**RIO Lifetime:** The length of time in seconds (relative to the time the packet is sent) that the prefix is valid for route determination.

**RIO Depth:** Set to 0 by the router that owns the resource and issues the RIO. Incremented by all routers that propagate the RIO towards the root.

**RIO Sequence:** Incremented by the router that owns the resource for each new RIO for that prefix. Left unchanged by all routers that propagate the RIO.

**Prefix:** Variable-length field containing a prefix, an IPv6 address or a multicast group id.

### [3.3.](#) LLN Router Operation

Route Dissemination information can be redistributed in another

routing protocol, e.g. MANET or IGP. But the MANET or the IGP route information SHOULD NOT be redistributed into Route Dissemination. This creates a hierarchy of routing protocols where Route Dissemination routes stand somewhere between connected and IGP routes. See Section [Section 6.1](#) for more discussion on integration with other routing protocols.

As a result:

- o LLN Tree Discovery establishes a tree using extended Neighbor Discovery RS/RA flows.
- o A routing algorithm exploits the tree to optimally move upstream traffic out of the LLN (default route).
- o Route Dissemination extends Neighbor Discovery in order to quickly establish hop-by-hop routes down the tree.
- o Source Routing can be used to provide additional routes towards nodes in the LLN. When and where there exists hop-by-hop state in routers, the source routing information can be made sparse.

Route Dissemination maintains abstract lists of known information. An entry contains the following abstract information:

- o A reference to the adjacency that was created for that prefix.
- o The IPv6 address of the advertising Neighbor.

- o The logical equivalent of the full Route Dissemination information.
- o A 'reported' Boolean to keep track whether this prefix was already reported to the parent's parent.
- o A counter of retries to count how many TIOs were sent to the neighbor without reachability confirmation for the prefix.

Route Dissemination stores the entries in either one of 3 abstract lists; the Connected, the Reachable and the Unreachable lists. In

practice all are part of a route table.

The Connected list corresponds to the resources owned by the LLN Router.

As long as a router keeps receiving timely RIOs for a given information, its entry is listed in the Reachable list.

Once scheduled to be destroyed, an entry is moved to the Unreachable list if the router has a parent to which it sends RIOs, otherwise the entry is cleaned up right away. The entry is removed from the Unreachable list when the parent changes or after a no-RIO has been sent to the parent indicating the loss of the prefix.

### RIO Processing

When ND sends an NA to the parent, Route Dissemination extends the message with RIO options for:

- \* All entries that are not deleted.
- \* All entries in the removed list, using a no-RIO.
- \* All entries in the advertised list that are 'not reported yet'. The entries are then set to 'reported'.

If an information is advertised as a no-RIO, the associated route is removed, and the entry is transferred to the removed list. Otherwise, the proper routing table is looked up:

- \* If a preferred route to that source from another protocol already exists, the RIO is ignored.
- \* If a new route can be created, a new entry is allocated to track it, as CONFIRMED, but not reported.

- \* If a Route Dissemination route existed already via the same Neighbor, it is CONFIRMED.
- \* If an older unicast route existed via a different Neighbor,



this is equivalent to a no-RI0 for the previous entry followed by a new RI0 for the new entry. So the old entry is scheduled to be destroyed, whereas the new one is installed.

#### Unicast Route Dissemination messages from child to parent

When sending Route Dissemination to its parent, a router includes the RI0s about not already reported entries in the Reachable and Connected lists, as well as no-RI0s for all the entries in the Unreachable list.

The TI0 from the root is used to synchronize the whole tree. Its period is expected to range from 500ms to hours, depending on the stability of the configuration and the bandwidth available.

The design choice behind this is NOT TO synchronize the parent and children databases, but instead to update them regularly to cover from the loss of packets. The rationale for that choice is network dynamicity. If the topology can be expected to change frequently, synchronization might be an excessive goal in terms of exchanges and protocol complexity. This results in a simple protocol with no real peering.

## [4.](#) Forwarding

The fundamental mechanisms described in this draft build a DAG to enable communication from the LLN Router nodes to the LLN Border Routers (upstream); a second mechanism informs LLN Routers about their children in the tree, hence enabling LLN Boarder Router to LLN Router communication (downstream) and node-to-node routing along the tree. While the previous sections focus on how routing information is disseminated throughout the LLN and used for routing, this section focuses on the forwarding policies used by LLN Routers.

Reliability is increased by allowing a node to try several potential next-hop nodes in upstream traffic; downstream traffic is sent along the tree formed by route dissemination.

### [4.1.](#) Upstream Forwarding

Forwarding in a LLN needs to account for requirements that are unusual in the IP world:

perfect loop freedom is a non-goal

the specification allows for the 'wheel model' where a packet circulates a bit around the destination till it finally makes it.

transient forwarding failure are commonplace

This specification introduces the capability for the layer 2 to give a packet back to layer 3 in order to try another adjacency.

Using the LLN Tree Discovery procedure, LLN Routers expose their path metrics using the Uniform Path Metric field in the TIO. Neighbor LLN Routers with a lesser depth in the tree than self are forwarding parents. Neighbor LLN Routers with a same depth in the tree are siblings. Forwarding via parents ensures a loop free operation whereas forwarding via siblings may not be loopfree unless additional measures are taken.

The approach taken in this specification is to favor forwarding via parents but still enable forwarding via siblings as a backup option. Preferring the parents enables a forwarding gradient towards the LBR that limits the chances of multiple consecutive hops over siblings. This specification also prevents from returning a packet back to the neighbor that just passed it. This simple rule coupled with the forwarding gradient protect against loops for a vast majority of cases, and the specification relies on an appropriate setting of the TTL in a given deployment to protect against meltdowns.

In more details:

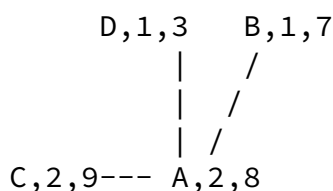
- o A LLN router MUST send upstream data to its forwarding parent with smallest metric. Note that, depending on the way the routing protocol determines this metric, and the dynamics of the tree, the best forwarding parent at a given point of time is not necessarily the parent with the smallest depth or the parent in the logical tree defined by the Tree Discovery procedure.
- o If the transmission of an upstream packet to that preferred parent fails (due to a node or link failure, or mobility), the LLN router MAY attempt to forward the packet again via other parents, as ordered by best metric.
- o If the transmission to both primary and secondary forwarding parents fails, the LLN Router MAY forward the packet via siblings, as ordered by best metric.
- o When the transmission fails and the packet is retried via a

different neighbor, the router MUST decrease the TTL by one.

In order to enable these rules, a LLN router maintains a blacklist per packet being forwarded that contains:

- o the neighbor that forwarded the packet to self
- o neighbors to which forwarding of this packet failed

These rules are illustrated in the following figure which represents a subset of an LLN.



An LLN Router is identified by <Id,Depth,Metric>. LLN Router A has three neighbors B,C,D. D is A's primary forwarding parent as it is the neighbor with the smallest Metric among neighbors with smaller depth. If transmission to D fails, A sends the packet to B, which is of smaller depth. If transmission to B fails, A transmits to C. Because C is at the same depth as A, a blacklisting policy is used to avoid that C retransmits to A.

#### [4.2.](#) Downstream Forwarding

Downstream routing using LLN fundamental mechanisms can occur using either hop-by-hop state, source routing or a combination thereof (loose source route). By default, the LLN Route Dissemination mechanism builds up hop-by-hop distance-vector routing information in each of the routers along the tree up to the root for each address, prefix or group ID.

Source routing can optionally be supported by either requesting a route record header from a node, or by having nodes send periodic route record headers up to the root. If a Route Dissemination route exists to the first entry in the Record Route header via the source of the packet, then the router can override the source of the packet with its address without adding the original source to the Record

Route. At that point, the routing header operation becomes loose, in other words an hybrid between transparent hop-by-hop (stateful) and source routing.

Therefore three different downstream techniques are supported:

- o Hop-by-hop forwarding. When only partial route dissemination data reaches a LLN Border Router, it only knows the next-hop to a given

LLN Router in the network. In this case, each LLN Router relaying downstream data will select the next-hop according to the information it receives during route dissemination.

- o Full source routing. When all the route dissemination data reaches a LLN Border Router, it one can choose to specify the full list of LLN Routers to be traversed in each downstream data packet.
- o Loose source routing. When the source route information is compressed because of existing state in the routers along the path.

## [5. Multicast Support](#)

### [5.1. Overview](#)

Wherever we mention <MLD>, one can read MLDv2,3 for IPv6. Doing IGMP over the LLN involves:

- o LLN Border Router acting as a local Rendez-vous Point (RP) for the LLN and as source towards the Internet for all multicast flows started in the LLN.
- o transporting <MLD> in Route Dissemination and recursive coalescence of the multicast requests.

### [5.2. Receiver Flow](#)

The LBR is considered as a Rendezvous Point (RP) for all multicast flows issued from inside the LLN. Multicast packets are passed up the tree to the LBR.

Nodes talk <MLD> to their parent router. The parent router forward the registration and inject their own as a special type of RIO for multicast groups, towards the LBR. The LBR MAY participate to multicast in the infrastructure it is connected to and forward all the packets coming from the LLN.

Between the parent router and the LBR, <MLD> requests are transported in the RIO; each hop aggregates the requests in a fashion that is similar to proxy IGMP, but this happens recursively between child node to parent router up to the LBR. On the way, multicast routing states are installed in each router from the receiver to the root, enabling multicast routing down the LLN tree.

### [5.3.](#) Source flow

As a Node, the source is unaware of the ROLL protocol, and it uses standard protocols with the router (say in IPv6: Neighbor Discovery, <MLD> etc...). So when it has a multicast packet to send, the source just forwards it to its default router, which is the expected standard behavior. Routers on the way recursively forward to their parent. At each hop, if a multicast route indicates that a listener is reachable via another child (different from that through which the packet was received) then the packet is duplicated and forwarded to that child down the tree.

If the LLN Border Router is configured to do so, it will source the packet to a real RP in the Internet.

## [6.](#) Advanced Features

The fundamental mechanisms described in this document are sufficient to allow for upstream and downstream communication inside the LLN. They form a common basis upon which future LLN routing protocols can be designed. This section indicates some possible advanced features which can be integrated to increase efficiency for a particular usage scenarios.

### [6.1.](#) Interaction with other routing protocols

While network design and specific use cases are out of scope for this document, it must be noted that the LLN fundamental mechanisms described herein might be used in conjunction with other routing protocols in order to fulfill the requirements of a particular deployment. Here follows a non exhaustive series of examples illustrating such interactions.

#### 6.1.1. AODV/DYMO

In the example of a closed loop between a sensor and a switch, a constrained optimized route must be installed between the 2 devices.

Defining such a specific route is costly and should be performed on-demand when the bulk of the traffic is buffered data from source to sink.

A reactive MANET protocol such as AODV [[RFC3561](#)], DSR [[RFC4728](#)] or DYMO [[I-D.ietf-manet-dymo](#)] can be deployed to enable such routing, though the QoS-constrained approach for AODV is stalled as a draft ([[I-D.perkins-manet-aodvgos](#)]).

#### 6.1.2. OSPF/OLSR

A federating backbone is the virtual root of a collection of trees that forms a single routing topology. If that topology shares a same prefix, a sensor device can move freely within the topology without renumbering. The 6LoWPAN backbone link is an example of such a federating backbone and in that case, the protocol that enables any to any reachability is simply IPv6 Neighbor Discovery [[RFC4861](#)].

In a generalized case with routing and multiple subnets, a traditional IGP such as OSPF [[RFC2740](#)] or a MANET protocol such as OLSR [[RFC3626](#)] can be deployed within the federating backbone between the LBR to advertise the routes learnt from the LLN fundamentals dissemination protocol through the redistribution of route information.

In turn, the routed federating backbone is just the instantiation at Depth 0 of the more general concept of beltlines. A beltline is a set of routers of a same depth in a same tree that form a subarea

where an IGP is run and route information from the LLN Route Dissemination protocol is redistributed. This creates routes around the root and reduces the load that routing along the tree imposes on the lower depth of the tree.

Note that in turn, beltline routes ARE NOT redistributed into LLN Route Dissemination information. As a result, the beltlines routes are orthogonal to the route dissemination routes, and they should never collide, which optimizes the value of the control plane of the combination.

Beltline routes should be used with caution in order to maintain stability and optimize the resulting routes:

- o beltline routes should only be used when a certain topological stability was asserted
- o using beltline routes discourages the reorganization of the tree, mostly when that causes a router to change its depth
- o a divide and conquer approach to limit the size of a beltline enables to manage the cost of the control plane
- o a beltline of depth 2 or more should be an arc as opposed to full circle. In the example of a closed loop between a sensor and a switch, a constrained optimized route must be installed between the 2 devices.

### [6.1.3.](#) MIP6/NEMO

MIP6 [[RFC3775](#)] and NEMO [[RFC3963](#)] enable a subtree to move away from the tree and maintain reachability as if the nodes in the subtree were still located in their topologically correct position. This can be useful when a RIO aggregation is performed (see [Section 6.6](#)) to enable reachability of a stray device. MIP6 can also be useful to enable a mobile display device such as a PDA to keep accessing a sensor network remotely without injecting the sensor network prefix into the infrastructure for security reasons.

## [6.2.](#) Route Optimization

Whereas upstream and downstream communication is made possible by the fundamental mechanisms described in this document, applications may require more require traffic engineering, which may include:

#### [6.2.1.](#) Node-to-node routing

Node-to-node routing is ensured along the tree by the Route Dissemination protocol, and the packets flow via the first common parent. This can be optimized if the LLN Border Router has a clear view of the topology (see 'Offline Path Computation' section). In this case, the LLN Border Router can indicate the direct path between both LLN Routers, calculated offline, to the source, the destination, or both. This technique induces a trade-off between multi-hop route efficiency and signaling overhead to setup this direct node-to-node path for instance as suggested in [Section 6.1.1](#).

#### [6.2.2.](#) Offline Path Computation

Whereas nodes might not have the capacity to store and manage enough information to perform constrained routing, it is possible for nodes to report their neighborhood information to the LLN Border routers. LLN Border routers can then share their partial topology databases and get a full picture of the network.

From there, it is possible to get LLN Border routers to compute shorter or constrained paths and either distribute them (e.g. LDP) or pass the source route information to the end nodes.

An OSPF example of that goes like this. Nodes run HELLO or similar, and send their LSA in unicast to their LLN Border routers. The LLN Border routers act as proxy for the nodes and share those LSAs with other LLN Border routers over the backbone. At some point they converge and an LLN Border router will run SPF on behalf of all its registered nodes, one at a time. The SPF computation should end at a certain distance from the node for which it makes more sense to go

through the backbone anyway. Then the LLN Border router sends the set of routes to the node as an new topology that can be used in a MTR fashion.

#### [6.2.3.](#) Graph forwarding



Distance Vector and Link State routing protocols are traditionally designed in terms of:

Links -> Metrics -> Routes -> network runtime

Unless traffic engineering kicks in, either the routes are established over the shortest path and the alternate links are wasted or the traffic is load balanced in a fashion that represents the ratio of costs as opposed to the ratio of capacity of the paths.

Also, the runtime of the network is opaque to the forwarding plane, so the only way to guarantee some end-to-end bandwidth for a class of traffic is to blindly reserve it, leading to even more waste of bandwidth when the reservation is not fully utilized.

In order to optimize the network utilization, it would be beneficial to detect the saturation of the shortest path and load balance the extra traffic over alternate routes. In the case of ROLL, it is also critical to be able to make a reroute decision on a per packet basis when hop by hop retries are exhausted. Arpanet introduced a feedback loop into the routing protocol by making the metrics dynamic:

```
Links -> Metrics -> Routes -> network runtime
          ^                                     |
          |-----|
```

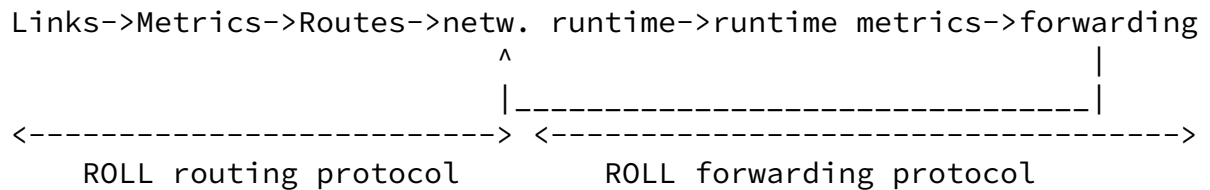
But this approach was unsuccessful, causing instabilities and disrupting the network. With dynamic metrics, the duration of the convergence time - or frozen time -, increases with the number of links and the frequency of the metric updates. During that time, the response of the network is undefined and temporary loops occur.

An approach to solve this problem is having 2 independent sets of metrics: on the one hand, the topological metrics that are rather static and mostly administratively set; and on the other hand, the volatile metrics that are based on dynamic measurements of the network characteristics.

The topological metrics are used by the LLN routing protocol to initially build the tree as described in this specification. The

volatile metrics are then used by a forwarding protocol to balance the traffic for that destination over the upstream links, thus modifying the way the graph is being used in runtime, without changing its structure.

To get there, the control plane operates in 2 phases, in a lollipop fashion:



The LLN fundamentals proposal builds shortest path trees to the exits but adds the capability to forward over another branch if sending a packet to a parent fails, either via any alternate parent or a sibling. So the paths that we really want to monitor are along the tree itself and one hop away from the tree. To get there, the root emits a beacon that is multicasted down the tree and heard one hop away. That beacon gathers the metrics that will be used for alternate parents and siblings selection and nodes keep track of the beacon they hear for all the parents and siblings they want to track. From the beacon, they can infer the quality of the path through all the alternates and compare them.

### [6.3.](#) Density

In a dense environment, it is useless that all routers that can provide backhauling service actually do so; in practice, limiting the number of routers that accept attached nodes saves memory in the attached nodes and reduces the cost of signalling. Also, limiting the number of forwarding LLN Routers in the tree improves the multicast operations.

Algorithms such a Trickle could be used by a LLN Router to decide to stop providing its access services for attached nodes if there are a number of neighboring routers that provide similar services. The simplest abstraction of such similarity is that a multiple routers advertising a same depth, though such a simple similarity does not address the specifics of a router selection in the plugins. In a more general fashion, a LLN Router can associate the concept of similarity with the characteristics of its own parent router selection plug in.

#### [6.4.](#) Digraph Dissemination

The fundamental techniques described in this draft overlay a tree for source/sink traffic over the physical topology. This tree could be converted into a (bi)graph with additional overhead. A LLN Router would therefore send route dissemination data to both its primary and secondary forwarding parents, hence informing an LLN Border Router of disjoint paths. This makes sense in applications where the gains in increase downstream reliability outweigh the additional signaling overhead.

#### [6.5.](#) Multiple LBRs and Trees

The LLN Tree Discovery technique propagates increasing depths and metrics throughout the network; upstream messages travel on a decreasing metric path back to the LLN Border Router. When the LLN features multiple LBRs, the following options appear:

- o If the different LBRs share the same TreeID, an LLN Router implicitly sends its upstream data to the LBR which is closest in terms of aggregated metric. This should be used whenever LBRs play the same role.
- o Different LBRs may choose to use different TreeIDs. In this case, a LLN Router is part of multiple trees, one for each TreeID. When sending an upstream message, a LLN Router chooses on which TreeID it wishes to send, i.e. to which LBR.
- o A hybrid case can exist in which some LBRs share the same TreeID while others have their dedicated Tree ID.

An alternative when having multiple LBRs is to construct multiple trees (e.g. one for each LBR) and choose a default tree for forwarding data. Using an alternate tree is possible only when labeling the data packet accordingly; an unlabeled packet is forwarded on the default tree.

#### [6.6.](#) Aggregation for Route Dissemination

Aggregation of prefixes on a same router

When deploying a router with multiple interfaces, it makes sense

to assign an aggregation prefix (shorter than /64) to the router and partition it as /64 prefixes over the router interfaces. A router that owns a contiguous set of prefixes should only report the aggregation of these prefixes through Route Dissemination.

### Aggregation of prefixes by a parent acting as ROLL Home

There are also a number of cases where a ROLL aggregation is shared within a platoon of LLN Routers. In that case, it is still possible to use aggregation techniques with Route Dissemination and improve its scalability. In that case, the parent is configured as the Route Dissemination aggregator for the group prefix. At run time, it absorbs the individual RIO information it receives from the platoon members down its subtree and only reports the aggregation up the TD tree. This works fine when the whole platoon is attached within the parent's subtree.

But other cases might occur for which additional support is required:

1. the aggregator is attached within the subtree of one of its platoon members.
2. a platoon member is somewhere else within the TD tree.
3. a platoon member is somewhere else in the Internet.

In all those cases, a node situated above the aggregator in the TD tree but not above the platoon member will see the advertisements for the aggregation owned by the aggregator but not that of the individual platoon member prefix. So it will route all the packets for the platoon member towards the aggregator, but the aggregator will have no route to the platoon and will fail to forward.

### [6.7.](#) Advanced Forwarding

A blacklisting policy can be used to avoid routing loops when an upstream data packet is sent between neighbor LLN Routers of the same depth. Alternatively, more general techniques can be used to avoid

loops. One is to record the sequence of already traversed nodes in the data packet as it travels along a multi-hop path. When receiving a packet, a LLN Router may know whether it has already relayed that packet; if yes, it can know from which neighbors it had received it and to which it had sent. A distributed version of depth first search can then be used to avoid routing loops. This extension enables upstream packets to be sent to neighbors with a larger depth.

## [7.](#) Security Considerations

As this draft suggests the use of new options carried in ICMP ND messages; the same security considerations as in [[RFC4861](#)] apply, in

Thubert, et al.

Expires October 11, 2009

[Page 25]

---

Internet-Draft

LLN Routing Fundamentals

April 2009

particular with regards to the use of Secure ND [[RFC3971](#)] to protect against address theft. Additionally link-layer security should be applied in the case of 6LoWPAN where SeND is not typically possible.

## [8.](#) IANA Considerations

This draft would require two new ICMP options for use with ND: the Tree Information Option (TIO) and the Route Information Option (RIO).

## [9.](#) Acknowledgments

The authors would like to thank Richard Kelsey, Robert Assimiti, Kris Pister, Mischa Dohler, Julien Abeille, Ryuji Wakikawa, Teco Boot, Patrick Wetterwald, Bryan McLaughlin and Carlos J. Bernardos for useful design considerations and reviews.

## [10.](#) References

### [10.1.](#) Normative References

- [RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", [RFC 2460](#), December 1998.
- [RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4

Networks", [RFC 4944](#), September 2007.

## [10.2.](#) Informative References

[I-D.ietf-manet-dymo]

Chakeres, I. and C. Perkins, "Dynamic MANET On-demand (DYMO) Routing", [draft-ietf-manet-dymo-17](#) (work in progress), March 2009.

[I-D.ietf-roll-building-routing-reqs]

Martocci, J., Riou, N., Mil, P., and W. Vermeulen, "Building Automation Routing Requirements in Low Power and Lossy Networks", [draft-ietf-roll-building-routing-reqs-05](#) (work in progress), February 2009.

[I-D.ietf-roll-home-routing-reqs]

Porcu, G., "Home Automation Routing Requirements in Low Power and Lossy Networks", [draft-ietf-roll-home-routing-reqs-06](#) (work in progress), November 2008.

Thubert, et al.

Expires October 11, 2009

[Page 26]

---

Internet-Draft

LLN Routing Fundamentals

April 2009

[I-D.ietf-roll-indus-routing-reqs]

Networks, D., Thubert, P., Dwars, S., and T. Phinney, "Industrial Routing Requirements in Low Power and Lossy Networks", [draft-ietf-roll-indus-routing-reqs-04](#) (work in progress), January 2009.

[I-D.ietf-roll-terminology]

Vasseur, J., "Terminology in Low power And Lossy Networks", [draft-ietf-roll-terminology-00](#) (work in progress), October 2008.

[I-D.ietf-roll-urban-routing-reqs]

Dohler, M., Watteyne, T., Winter, T., Barthel, D., Jacquenet, C., Madhusudan, G., and G. Chegaray, "Urban WSNs Routing Requirements in Low Power and Lossy Networks", [draft-ietf-roll-urban-routing-reqs-05](#) (work in progress), March 2009.

[I-D.perkins-manet-aodvqos]

Perkins, C. and E. Belding-Royer, "Quality of Service for Ad hoc On-Demand Distance Vector Routing",

[draft-perkins-manet-aodvqos-01](#) (work in progress),  
November 2001.

- [RFC2740] Coltun, R., Ferguson, D., and J. Moy, "OSPF for IPv6", [RFC 2740](#), December 1999.
- [RFC3561] Perkins, C., Belding-Royer, E., and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing", [RFC 3561](#), July 2003.
- [RFC3626] Clausen, T. and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)", [RFC 3626](#), October 2003.
- [RFC3775] Johnson, D., Perkins, C., and J. Arkko, "Mobility Support in IPv6", [RFC 3775](#), June 2004.
- [RFC3963] Devarapalli, V., Wakikawa, R., Petrescu, A., and P. Thubert, "Network Mobility (NEMO) Basic Support Protocol", [RFC 3963](#), January 2005.
- [RFC3971] Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", [RFC 3971](#), March 2005.
- [RFC4728] Johnson, D., Hu, Y., and D. Maltz, "The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4", [RFC 4728](#), February 2007.

Thubert, et al.	Expires October 11, 2009	[Page 27]
-----------------	--------------------------	-----------

---

Internet-Draft	LLN Routing Fundamentals	April 2009
----------------	--------------------------	------------

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

#### Authors' Addresses

Pascal Thubert  
Cisco Systems  
Village d'Entreprises Green Side  
400, Avenue de Roumanille  
Batiment T3  
Biot - Sophia Antipolis 06410  
FRANCE

Phone: +33 4 97 23 26 34  
Email: pthubert@cisco.com

Thomas Watteyne  
UC Berkeley  
497 Cory Hall #1774  
Berkeley Sensor & Actuator Center  
Berkeley, California 94720-1774  
USA

Phone: +1 (510) 333-4437  
Email: watteyne@eecs.berkeley.edu

Zach Shelby  
Sensinode  
Kidekuja 2  
Vuokatti 88600  
FINLAND

Phone: +358407796297  
Email: zach@sensinode.com

Dominique Barthel  
Orange Labs  
28 chemin du Vieux Chene, BP98  
BP98  
Meylan 38243  
FRANCE



Phone: +33476764522

Email: dominique.barthel@orange-ftgroup.com