Computer Networks 56 (2012) 3163–3178

Contents lists available at SciVerse ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Survey Paper

RPL in a nutshell: A survey

Olfa Gaddour [a,*], Anis Koubâa [b]

[a] CES Research Unit, National School of Engineers of Sfax, University of Sfax, Tunisia
[b] CISTER Research Unit, Polytechnic Institute of Porto (ISEP/IPP), Portugal

ARTICLE INFO

Article history:
Received 23 December 2011
Received in revised form 2 June 2012
Accepted 16 June 2012
Available online 7 July 2012

Keywords:
RPL
Routing protocols
Low power and lossy networks (LLNs)
ROLL IETF
Performance evaluation
Experimentation

ABSTRACT

IPv6 Routing Protocol for Low Power and Lossy Networks (RPL) is a routing protocol specifically designed for Low power and Lossy Networks (LLN) compliant with the 6LoWPAN protocol. It currently shows up as an RFC proposed by the IETF ROLL working group. However, RPL has gained a lot of maturity and is attracting increasing interest in the research community. The absence of surveys about RPL motivates us to write this paper, with the objective to provide a quick introduction to RPL. In addition, we present the most relevant research efforts made around RPL routing protocol that pertain to its performance evaluation, implementation, experimentation, deployment and improvement. We also present an experimental performance evaluation of RPL for different network settings to understand the impact of the protocol attributes on the network behavior, namely in terms of convergence time, energy, packet loss and packet delay. Finally, we point out open research challenges on the RPL design. We believe that this survey will pave the way for interested researchers to understand its behavior and contributes for further relevant research works.

1. Introduction

6LoWPAN [1] is a milestone protocol that bridged the gap between low-power devices and the IP world. It is an IP-based technology for Low-Power Wireless Personal Area Networks (LoWPANs), such as Wireless Sensor Networks, that combines IEEE 802.15.4 [2] and the IPv6 protocols [3]. This integration provided a new dimension in the design of LoWPANs as it allows for a full interoperability with the Internet.

Since the specification of 6LoWPAN [4], routing has been considered as one of the key issues in 6LoWPAN networks that are worth investigation. Indeed, there have been several endeavors for specifying an efficient routing protocol for 6LoWPAN-compliant LLNs, such as for instance, Hydro [5], Hilow [6], and Dymo-low [7]. All these proprietary solutions did not gain a lot of space in the arena, and an increasing need for a standard solution has arisen. To address this gap, the IETF ROLL working group [8,9] has proposed a routing protocol, referred to as *RPL* (Routing Protocol for Low power and lossy networks), which is the main candidate for acting as the standard routing protocol for IP smart object networks (also referred to as LLN). A key feature of RPL is that it is designed for networks with lossy links, which are those exposed to high Packet Error Rate (PER) and link outages. Although RPL is still a RFC, it has gained a lot of maturity turning it as a promising standardized routing solution for Low Power and Lossy Networks. In fact, several research works have focused on the design and deployment of RPL protocol and real-world implementations have been showing up [10–13], etc. The increasing popularity of RPL is due to several factors, including its flexibility to adapt to different topologies, QoS support and other interesting features that we present next.

To the best of our knowledge, this is the first comprehensive survey of the RPL protocol. In [14], the authors presented an overview on the 6LoWPAN and RPL technologies. As compared to this survey, Ref. [14] only provides a brief tutorial-like introduction of the RPL protocol and does not

* Corresponding author. Tel.: +216 96 819 500.
E-mail addresses: olfa.gaddour@enis.rnu.tn (O. Gaddour), aska@isep.ipp.pt (A. Koubâa).

1389-1286/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.comnet.2012.06.016

comprehensively describe the protocol behavior/mechanisms, nor does it discuss the recent works tied to RPL. This survey is roughly divided into two majors parts: The first part spans over Sections 2–5 which present an overview of RPL and its main features. The second part (i.e. Sections 6–7) provides a comprehensive review on the latest development and research works related to RPL and points out underlying research challenges.

The remainder of the paper is organized as follows. Section 2 presents the design objectives and network architecture of the RPL protocol. In Section 3, we present the routing protocol specification namely the protocol control headers and network construction operations. Section 4 describes network management mechanisms pertaining to fault-tolerance, QoS, and security. In Section 5, we present an experimental performance evaluation to understand the impact of RPL attributes on the network behavior. In Section 6, we compare RPL with its competitors and provide a literature review of relevant and recent works around the RPL protocol. Finally, Section 7 concludes the paper and discusses open research challenges with respect to RPL design and deployment.

2. Protocol overview

2.1. Design objectives

RPL is a distance–vector (DV) and a source routing protocol that is designed to operate on top of several link layer mechanisms including IEEE 802.15.4 PHY and MAC layers. It targets *collection-based networks,* where nodes periodically send measurements to a collection point, as well as point-to-multipoint traffic from the central point to the devices inside the LLN. Point-to-point traffic is also supported in RPL. A key feature in RPL is that it represents a specific routing solution for low power and lossy networks [14,15], which stand for networks with very limited resources in terms of energy, computation and bandwidth turning them highly exposed to packet losses. In fact, it has been specifically designed to meet the requirements of resource-constrained nodes as mentioned in the routing requirement terminology document [16]. In particular, RPL-enabled LLNs take into account two main features (i) the prospective data rate is typically low (less than 250 kbps), and (ii) communication is prone to high error rates, which results in low data throughput. A lossy link is not only characterized by a high Bit Error Rate (BER) but also the long inaccessibility time, which strongly impacts the routing protocol design. In fact, the protocol was designed to be highly adaptive to network conditions and to provide alternate routes, whenever default routes are inaccessible.

RPL is based on the topological concept of *Directed Acyclic Graphs* (DAGs). The DAG defines a tree-like structure that specifies the default routes between nodes in the LLN. However, a DAG structure is more than a typical tree in the sense that a node might associate to multiple parent nodes in the DAG, in contrast to classical trees where only one parent is allowed. More specifically, RPL organizes nodes as *Destination-Oriented DAGs* (DODAGs), where most popular destination nodes (i.e. sinks) or those providing a default route to the Internet (i.e. gateways) act as the roots of the DAGs.

A network may consist of one or several DODAGs, which form together an *RPL instance* identified by a unique ID, called *RPLInstanceID*. A network may run multiple RPL instances concurrently; but these instances are logically independent. An node may join multiple RPL instances, but must only belong to one DODAG within each instance. Fig. 1 shows an example of RPL instances with multiple DODAGs.

One of the relevant features of the RPL routing protocol is that it combines both *mesh* and *hierarchical* topologies. On the one hand, an RPL-based network topology is inherently hierarchical as it forces underlying nodes to self-organize as one or several *DODAGs*, based on parent-to-child relationship. On the other hand, RPL supports the mesh topology as it allows routing through siblings instead

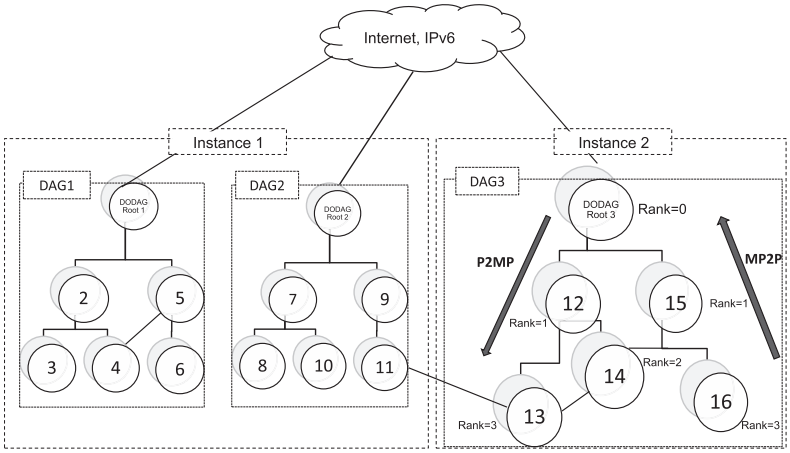

Fig. 1. A RPL network with three DODAGs in two instances.

of parents and children, when needed. This combination mesh/hierarchical provides a great flexibility in terms of routing and topology management.

RPL provides the following features [15]:

- *Auto-configuration*: As RPL is compliant with IPv6, the RPL-based LLN will benefit from basic IP routing characteristics mainly the dynamic discovery of network paths and destinations. This feature is guaranteed by the use of the Neighbor Discovery mechanisms.
- *Self-healing*: RPL proves its ability to adapt to logical network topology changes and node failures. In fact, links and nodes in LLNs are not stable and may vary frequently. RPL implements mechanisms that choose more than one parent per node in the DAG to eliminate/decrease the risks of failure.
- *Loop avoidance and detection*: A DAG is acyclic by nature as a node in a DAG must have a higher rank than all of its parents. RPL includes reactive mechanisms in order to detect loops in case of topology change. In addition, RPL triggers recovery mechanisms (global and local repair) when the loops occur.
- *Independence and Transparency*: As in the IP architecture, RPL is designed to operate over multiple link layers. RPL is able to operate in constrained networks, or in conjunction with highly constrained devices. Thus, RPL is then independent from data-link layer technologies.
- *Multiple edge routers*: It is possible to construct multiple DAGs in an RPL network and each DAG has a root. A node may belong to multiple instances, and may act different roles in each instance. Thus, the network will benefit from high availability and load balancing.

2.2. Network model

RPL defines three types of nodes:

- *Low Power and Lossy Border Routers* (LBRs): it refers to the root of a DODAG that represents a collection point in the network and has the ability to construct a DAG. The LBR also acts as a gateway (or edge router) between the Internet and the LLN.
- *Router*: it refers to a device that can forward and generate traffic. Such a router does not have the ability to create a new DAG, but associate to an existing one.
- *Host*: it refers to an end-device that is capable of generating data traffic, but is not able to forward traffic.

The basic topological component in RPL is the DODAG, a Destination Oriented DAG, rooted in a special node called *DODAG root*, as illustrated in Fig. 1. The DODAG root has the following properties: (i) it typically acts as an LBR, (ii) it represents the data sink within the directed acyclic graph, (iii) it is typically the final destination node in the DODAG, since it acts as a common transit point that bridges the LLN with IPv6 networks, (iv) it has the ability to generate a new DODAG that trickles downward to leaf nodes.

Each node in the DODAG is assigned a *rank*. The *rank* of a node is defined in [15] as "the node's individual position relative to other nodes with respect to a DODAG root". It is an integer that represents the location of a node within the DODAG. The rank strictly increases in the downstream direction of the DAG, and strictly decreases in the upstream direction. In other words, nodes on top of the hierarchy receive smaller ranks than those in the bottom and the smallest rank is assigned to the DODAG root. This is illustrated in Fig. 1.

The architecture of a DODAG is similar to a cluster-tree topology where all the traffic is collected in the root. However, the DODAG architecture differs from the cluster-tree in the sense that a node can be associated not only to its parent (with higher rank), but also to other sibling nodes (with equal ranks). The rank is used in RPL to avoid and detect routing loops (refer to Section 4), and allows nodes to distinguish between their parents and siblings in the DODAG. In fact, RPL enables nodes to store a list of candidate parents and siblings that can be used if the currently selected parent loses its routing ability.

In the construction process of network topology, each router identifies a stable set of parents on a path towards the DODAG root, and associates itself to a preferred parent, which is selected based on the *Objective Function*. The Objective Function defines how RPL nodes translate one or more metrics into ranks, and how to select and optimize routes in a DODAG. It is responsible for rank computation based on specific routing metrics (e.g. delay, link quality, connectivity, etc.) and specifying routing constraints and optimization objectives. The design of efficient Objective Functions is still an open research issue. A couple of drafts have been proposed. In [17], the draft proposes to use the Expected Number of Transmission (ETX) required to successfully transmit a packet on the link as the path selection criteria in RPL routing. The route from a particular node to the DODAG root represents the path that minimizes the sum of ETX from source to the DODAG root. In [18], the draft proposes Objective Function 0 (OF0), which is only based on the abstract information carried in an RPL packet, such as Rank. OF0 is agnostic to link layer metrics, such as ETX, and its goal is to foster connectivity among nodes in the network.

3. Routing protocol specification

In this Section, we present the main mechanisms and features provided in RPL as defined in IETF drafts.

3.1. RPL control messages

RPL messages are specified as a new type of ICMPv6 control messages, whose structure is depicted in Fig. 2.

According to [19], the RPL control message is composed of (i) an ICMPv6 header, which consists of three fields: Type, Code and Checksum, (ii) a message body comprising a message base and a number of options.

The *Type* field specifies the type of the ICMPv6 control message prospectively set to 155 in case of RPL (confirmed by the Internet Assigned Number Authority (IANA)). The *Code* field identifies the type of RPL control message. Four codes are currently defined:

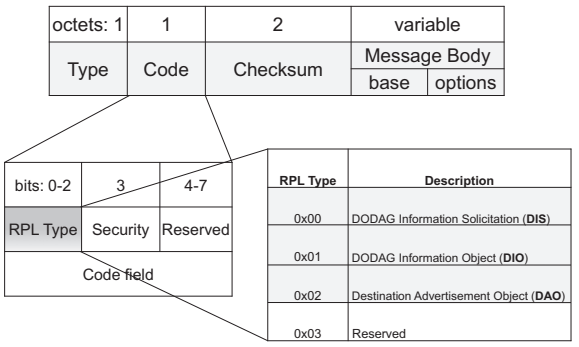

Fig. 2. RPL control message.

- *DODAG Information Solicitation* (DIS): The DIS message is mapped to 0×01, and is used to solicit a *DODAG Information Object* (DIO) from an RPL node. The DIS may be used to probe neighbor nodes in adjacent DODAGs. The current DIS message format contains non-specified flags and fields for future use.
- *DODAG Information Object* (DIO): The DIO message is mapped to 0x01, and is issued by the DODAG root to construct a new DAG and then sent in multicast through the DODAG structure. The DIO message carries relevant network information that allows a node to discover a RPL instance, learn its configuration parameters, select a DODAG parent set, and maintain the DODAG. The format of the DIO Base Object is presented in Fig. 3. The main DIO Base Object fields are: (i) *RPLInstanceID*, is an 8-bit information initiated by the DODAG root that indicates the ID of the RPL instance that the DODAG is part of, (ii) *Version Number*, indicates the version number of a DODAG that is typically incremented upon each network information update, and helps maintaining all nodes synchronized with new updates, (iii) *Rank*, a 16-bit field that specifies the rank of the node sending the DIO message, (vi) *Destination Advertisement Trigger Sequence Number* (DTSN) is an 8-bit flag that is used to maintain downward routes, (v) *Grounded* (G) is a flag indicating whether the current DODAG satisfies the application-defined objective, (vi) *Mode of Operation* (MOP) identifies the mode of operation of the RPL instance set by the DODAG root. Four operation modes have been defined and differ in terms of whether they support downward routes maintenance and multicast or not. Upward routes are supported by default. Any node joining the DODAG must be able to cope with the MOP to participate as a router, otherwise it will be admitted as a leaf node, (vii) *DODAGPreference* (Prf) is a 3-bit field that specifies the preference degree of the current DODAG root as compared to other DODAG roots. It ranges from 0×00 (default value) for the *least preferred* degree, to 0×07 for the *most preferred* degree, (viii) *DODAGID* is a 128-bit IPv6 address set by a DODAG root, which uniquely identifies a DODAG. Finally, DIO Base Object may also contain an Option field.

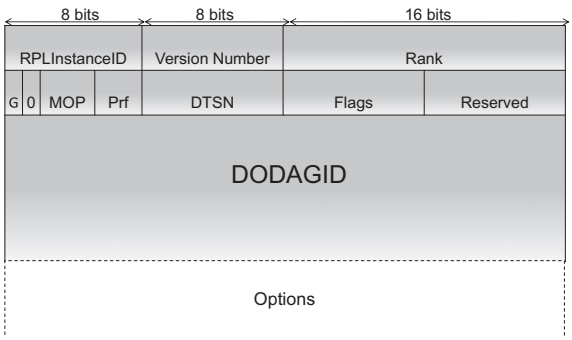

Fig. 3. The DIO message format.

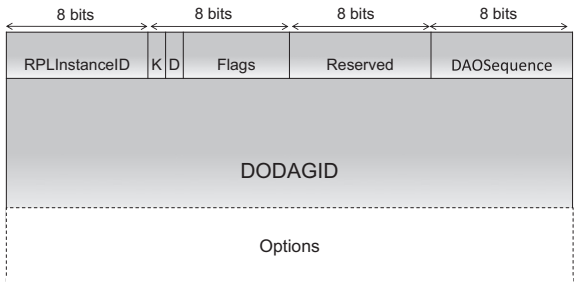

Fig. 4. The DAO message format.

- *Destination Advertisement Object* (DAO): The DAO message is mapped to 0×02, and is used to propagate reverse route information to record the nodes visited along the upward path. DAO messages are sent by each node, other than the DODAG root, to populate the routing tables with prefixes of their children and to advertise their addresses and prefixes to their parents. After passing this DAO message through the path from a particular node to the DODAG root through the default DAG routes, a complete path between the DODAG root and the node is established. Fig. 4 illustrates the format of the DAO Base Object.
  As shown in the figure, the main DAO message fields are: (i) *RPLInstanceID*, is an 8-bit information indicates the ID of the RPL instance as learned from the DIO, (ii) *K flag* that indicates whether and acknowledgment is required or not in response to a DAO message, (iii) *DAOSequence* is a sequence number incremented at each DAO message, (iv) *DODAGID* is a 128-bit field set by a DODAG root which identifies a DODAG. This field is present only when *flag D* is set to 1.
- *Destination Advertisement Object* (DAO-ACK): The DAO-ACK message is sent as a unicast packet by a DAO recipient (a DAO parent or DODAG root) in response to a unicast DAO message. It carries information about *RPLInstanceID*, *DAOSequence*, and *Status*, which indicate the completion. Status code are still not clearly defined, but codes greater than 128 mean a rejection and that a node should select an alternate parent.

3.2. DODAG construction

The DODAG construction is based on the Neighbor Discovery (ND) process, which consists in two main operation

(1) broadcast transmission of DIO control messages issued by the DODAG root to build routes in the downward direction from the root down to client nodes, (2) unicast of DAO control messages issued by client nodes and sent up to the DODAG root to build routes in the upward direction.

In order to construct a new DODAG, the DODAG root broadcasts a DIO message to announce its DODAGID, its Rank information to allow nodes to determine their positions in the DODAG, and the Objective Function identified by the *Objective Code Point* (OCP) within the *DIO Configuration* option fields. This message will be received by a *client node* which can be either a node willing to join or an already joined node.

When a node willing to join the DODAG receives the DIO message, it (i) adds the DIO sender address to its parent list, (ii) computes its rank according to the Objective Function specified in the OCP filed, such that the node's rank is greater than that of each of its parents, and (iii) forward the DIO message with the updated rank information. The client node chooses the most *preferred parent* among the list of its parents as the default node through which inward traffic is forwarded.

When a node already associated with DODAG receives another DIO message, it can proceed in three different ways (i) discard the DIO message according to some criteria specified by RPL, (ii) process the DIO message to either maintain its location in an existing DODAG or (iii) improve its location by getting a lower rank in the DODAG based on computing the path cost specified by the Objective Function. Whenever a node changes its rank, it must discard all nodes in the parents' list whose ranks are smaller than the new computed node's rank to avoid routing loops.

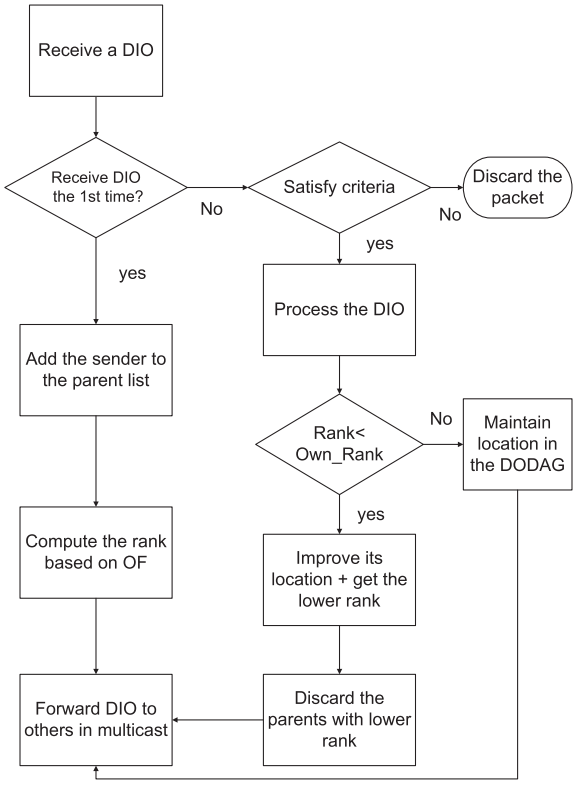

Fig. 5. The operation of a router in a DODAG.

The flowchart presented in Fig. 5 summarizes the operation of a router in a DODAG.

After the construction of the DODAG, each client node would have a default upward route through which it can transmit its inward traffic at the destination of the DODAG root. Obviously, the default route is formed by the most preferred parent of each node.

If the Mode of Operation flag in the DIO Base Object is different from zero, downward routes from the root to nodes are supported and have to be maintained. In this case, each client node must send a unicast DAO control message to determine the reverse route information. When traveling back to the DODAG root, visited nodes are recorded in the packet along the upward route, and complete route is then established between the DODAG root and the client node. RPL specifies two modes of operations to maintain downward routes in an RPL instance:

- *Storing mode*: in the storing mode, a DAO message is sent in unicast by the child to the selected parent, which is able to store DAO messages received by its children before sending the new DAO message with aggregate reachability information to its parent. The storing mode can enable or disable multicast mode.
- *Non-storing mode*: in the non-storing mode, the DAO message is sent in unicast to the DODAG root, thus, intermediate parents do not store DAO messages, but only insert their own addresses to the reverse route stack in the received DAO message, then forwards it to its parent.

To maintain the DODAG, each node periodically generates DIO messages triggered by a *trickle timer*. The key idea of the trickle timer technique is to optimize the message transmission frequency based on network conditions. In a nutshell, the frequency is increased whenever an inconsistent network management information is received for faster recovery from a potential failure, and decreased in the opposite case. The timer duration increases exponentially whenever the timer fires. Initially, the timer duration is set to I_{min}, which will be doubled $I^{doubling}$ times until it reaches the maximum value $I_{max} = I_{min} * I^{doubling}$. For any detected change in the DODAG (e.g. unreachable parent, new parent selection, new DODAG Sequence Number, routing loop, etc.), the trickle timer is reset to I_{min}, prescribed in DIO messages.

3.3. Communication paradigms

RPL supports three communication paradigms: (i) *Multi-Point-to-Point* (MP2P) (or many-to-one); (ii) *Point-To-Multi-Point* (P2MP) (or one-to-many); (iii) *Point-To-Point* (P2P) (or one-to-one). In what follows, we detail the operation of these communication patterns.

3.3.1. Multi-Point-to-Point Operation

RPL provides support for multipoint-to-point traffic which pertains to *data collection traffic* forwarded in the

upward route direction from multiple nodes towards the DODAG root. The data collection traffic is referred to as *inward unicast traffic*. The MP2P traffic is the main traffic flow in most of LLN applications [20–23]. The destinations of MP2P are mainly border routers that play important roles in the network and provide an interface for connectivity with the Internet. RPL supports MP2P traffic such that destinations can be reached via DODAG roots. Once the DODAG is constructed, it is used to build the upward routes from routers to the root. The default routes from nodes to the roots are established through the preferred parents chain. The DAG root can insert in its DIO messages the destination prefixes which may also be included by DIOs generated by the routers through the LLN, to which it can provide connectivity. The main advantage of MP2P traffic in RPL is that it is supported with little routing state as the node should only store the destination that leads to the DAG root.

3.3.2. Point-to-Multipoint Operation

RPL also defines the Point-to-Multipoint Operation, which represents the traffic transmitted in the downward route direction from the root down to multiple nodes. This configuration traffic is commonly known as outward *unicast traffic*, and it is essential for some LLN applications Home Automation [20], and Industrial Automation [21]. To support P2MP traffic, RPL uses a destination advertisement mechanism, which supplies down routes to routers until reaching the destination. To install downward routes, the routers send unicast DAO messages to their parents or to the DAG root. The DAO messages contain the prefixes within the network, and advertise the routes for each destination. Each intermediate router that forwards a DAO message towards the root adds its address to a reverse routing route in the DAO message, thus providing the source with the ability of performing source routing for reaching the child nodes in the network.

3.3.3. Point-to-Point Operation

RPL provides mechanisms for point-to-point (P2P) routing between any two nodes in the DODAG. In order to support P2P traffic in a RPL network, a LBR must be able to transit packets at which point it is source routed to the destination. Two cases arise (i) If the destination node is co-located with the sender node in the same transmission range, then the latter directly sends the message to the destination without passing it through its parent. (ii) Else, the P2P mechanism would depend on whether the network is pre-configured as storing or non-storing mode. In the non-storing mode, routers do not store any information about downward routes (no information about their descendants) and only the root possesses such information. In this case, any packet must be first sent through the DODAG upward routes to the root, which will forward it to its destination. In the storing mode, routers locally store the routing information about downward routes. If the destination is a descendant of the router, then it forwards the message down to the closest router to the destination. If the destination is not a descendent, the message is forwarded to the parent node, which will apply the same aforementioned rules to send the packet to its destination. As such, the packet will be forwarded in the uplink direction of the tree from child to parent until reaching the router that is the first ancestor of both the source and destination nodes.

3.4. Multicast routing

Multicast is supported by RPL only in the storing mode, when the (*MOP*) field in the DIO control message is set to 0×03.

RPL relies on the Multicast Listener Discovery (MLD) group registration protocol [24] for supporting multicast routing. In fact, it maps MLD queries into RPL DAO messages by transporting a multicast group in DAO target option. This mapping enables a DODAG root to act as a multicast router as if the listener were directly attached to it. Nodes that support the multicast operation can join the network as routers, but those that do not support multicast can only join as leaf nodes.

If a listener is not an RPL node, then it uses the typical MLD protocol to register to a multicast group. If this listener is attached to an RPL router while multicast is supported, then the RPL router map the MLD queries into a DAO message for the registration of the requesting node. If the listener is already RPL-enabled, then DAO message are used by default for group registration. MLD requests are then transported as DAO messages within the DODAG recursively between child and its parent up to the DODAG root.

Multicast routing information are located at each router on the way from the nodes to the DODAG root, enabling the root to send a multicast packet to all its children that had issued a DAO message requesting for that multicast group, as well as all the attached nodes that registered over this multicast group [15].

When node sends a multicast packet inside the DODAG, the packet is forwarded to the preferred parent, by default. If the transmission fails, then the packet will be routed to the alternate parents in the DODAG. The packet is also duplicated to all the registered children, except for the one that passed the packet.

The multicast operation is then centralized in the DODAG root which acts as an automatic proxy point (or Rendez-vous point) for the RPL network, and as source towards the non-RPL domain for all multicast flows started in the RPL domain.

4. RPL network management

RPL provides several mechanisms for network management. In what follows, we present an overview of such mechanisms that pertain to fault-tolerance, QoS-aware routing and security.

4.1. Fault-tolerance

RPL is a self-healing routing and topology control protocol. In fact, it presents mechanisms for (1) DODAG repair operation, triggered when an inconsistency is detected in

the DODAG, (2) loop detection and avoidance mechanisms to avoid loops in routing.

4.1.1. DODAG repair

Repair mechanisms are of paramount importance for a routing protocol to dynamically update routing decisions and adapt the network topology to potential node/link failure. To that end, RPL supports two complementary repair mechanisms: (i) *global repair* and (ii) *local repair*. When a node detects a network inconsistency (e.g. a link between two nodes fails or a local loop is detected), it triggers a local repair operation. It consists in urgently finding a backup path without trying to repair the whole DODAG. This alternate recovery path may not be an optimal path.

If local repairs are not efficient for network recovery due to several inconsistencies, the DODAG root may trigger a global repair operation and then it increments the DODAG version number and initiates a new DODAG version. The global repair operation leads to a fundamental reconstruction of the network topology. Nodes in the new DODAG version can choose a new position whose rank is neither constrained by nor dependent on their previous rank within the old DODAG version.

4.1.2. Loop avoidance and detection

Loops are a common undesirable problem in distance–vector routing protocols. To overcome it, RPL relies on rank-based and path-validation mechanisms for loop avoidance and loop detection. We alert the reader that loop avoidance and detection mechanisms are different from those defined in traditional IP networks [14]: In fact, in LLNs, the overreaction to loop detection is not recommended as opposed to IP networks where fast reaction is a must, since the prospective traffic in LLNs is very low, thus limiting the influence of potential loops.

Loop avoidance: A loop may occur when a node that loses all its parents, for some reasons, and attaches to another node in its own sub-DODAG. This may happen in particular when DIO messages are lost, and referred to as *DIO loops*. Several rules have been defined to avoid loops. The *max_depth* rule imposes that: a node cannot select a parent whose rank is higher than the node minimum rank plus *DAGMAXRankIncrease*. This rule does not prevent loops from occurring but avoid count-to-infinity when a loop is formed. For instance, considering the network in Fig. 1, assume that the link between $N15$ and the root was broken, and $N15$ selects node $N13$ as parent. This situation results in a loop as $N15$ has no means to know that $N13$ belongs to its sub-DODAG. A local repair operation will detect and repair the loop after a few network updates, which will end by dissociating $N15$ from $N13$. The repair operation works as follows: if $DAGMAXRankIncrease = 5$, this means that $N15$ can join any node with a rank $(1 + 5) = 6$. If $N15$ selects $N13$ (which satisfies the condition) as a parent, its new rank is increased to 4. Then, $N14$ updates in turn its rank to 5, $N13$ updates its rank to 6, and $N15$ updates again its rank to 7, which exceed the maximum allowed rank, that is 7. The loop is then detected, and thus avoided by breaking the child-parent relation between $N15$ and $N13$. RPL also prevents nodes looking for alternate parents to increase their ranks by selecting deeper parents, since this would very likely results in loops in the LLN.

Another way to avoid loops is that a node may poison its sub-DAG by advertising a rank of INFINITE_RANK without having to use *DAGMaxRankIncrease*. In addition to DIO loops, *DAO loops* may occur when a node fails to inform its parent that a destination is no longer reachable. In other words, the parent maintains a route installed towards a destination based on old DAO messages from a child, but the child is not able to update its parent about the non availability of that route, due to DAO message loss. In this case, if the child wants to send a packet to that destination, its non updated parent will keep sending back the packet to the child, thus forming a loop. The use of acknowledgment of DAO messages represents a fundamental solution to avoid DAO loops.

Loop detection: Loops are often hard to avoid. Thus, RPL specifies loop detection mechanisms to resolve them when they do occur. Detection mechanisms rely on the concept of *data path validation*, which consists in carrying control data in data packets to ensure that a packet is moving in forward direction and not experimenting any loop. Control data are flags making part of packet headers. For instance, the current RPL version allows one-hop sibling path, which means that a packet can be forwarded to a sibling only once along its path to the destination. The packet is then dropped in the second attempt to forward the packet to a sibling of another node, which can be easily encoded with a flag in packet header.

4.2. QoS-aware routing

RPL is a QoS-aware and constrained-based routing protocol. QoS-aware routing means that RPL is able to provide different levels of QoS based on the underlying QoS *metric* ruling routing decisions. Constraint-based routing means that RPL defines *constraints* that reduce the search space for possible path satisfying QoS requirements. QoS metrics and QoS constraints are typically different from each other. A *metric* is a quantitative value that helps to find the best path satisfying an Objective Function. For instance, an Objective Function based on the delay metric would be to find the path that minimizes the end-to-end delay from the source to the destination. On the other hand, a *constraint* is used to include or eliminate links or nodes that do not respect specific criteria [14]. For example, the Objective Function would recommend to prune/avoid links with poor quality. A routing metric or constraint can be additive or multiplicative, or selects a path that contains a maximum or a minimum value. In addition, routing metric can be either local, when it does not propagate along the DODAG, in contrast to global routing metrics which should be propagated.

In [25], the ROLL working group specified a set of links and nodes link's and node's routing metrics and constraints that are suitable to LLNs and recommended for the RPL routing protocol. RPL defines in the DIO control message a common optional header for metrics and constraints objects, called *DAG Metric Container object*, with several flags. Flags are used to specify the nature and features of routing objects, such as whether it represents a

metric or constraints, whether it is local or global, whether a metric is additive or others, etc.

The main routing metric and constraint objects are summarized in what follow:

- *Node State and Attribute (NSA) Object*: this metric reports information on node state and attributes such as CPU overload, available memory. The ROLL working group decided to set a 1-bit flag when a node faces a congestion situation, assessed according to a local policy. If the flag is set, traffic will be re-routed to avoid passing through the congested node.
- *Node Energy Object*: This object is used as a constraint to avoid using nodes with low power level. In [25], the Node Energy Object can be described by any combination of the following indicators: (1) the *node power mode*, encoded by 2-bit flag indicating the type of the node's power sources: main-powered, battery-powered, or scavenger (solar panel, mechanical, etc.), (2) the *estimated remaining lifetime*, which provides an estimation of the remaining power-level for nodes operating with batteries and scavengers, (3) other power-related metrics to be defined in the future.
- *Hop Count Object*: this metric simply reports the number of hops crossed along the path between a node and the destination;
- *Link Throughput Object*: this metric reports the range of throughput that the link can handle, in addition to the current link throughput. It can be used as a metric or constraint. When used as a metric, it may be considered as additive, or it may report a maximum or a minimum value.
- *Link latency Object*: this metric is used to report the path latency. The latency of the path is expressed as the sum of all latencies, or can be mapped to the maximum/minimum latency along the path. It can also be used as a constraint (e.g. pruning all links with a latency higher than a certain threshold).
- *Link Reliability Object*: this metric specifies the link reliability level, which can be expressed in several ways such as packet reception ratio, bit error rate, mean time between failures, and others. In [25], two links reliability metrics have been defined: (1) the Link Quality Level (LQL) object, which quantifies the link reliability using a discrete value, from 0 to 7 where 0 indicates that the link quality level is unknown and 1 reports the highest link quality level. The mechanisms specifying how to compute LQL has still not been defined and are implementation specific. (2) Expected Transmission count Metric (ETX), which provides an estimation of the number of transmission a node has to make along the path to the destination to deliver a packet. It is typically estimated as $1/(PRR_{up} \times PRR_{down})$ that is the inverse of the product of packet reception ratio (PRR) in upstream and downstream directions.
- *Link Color Object*: this constraint allows to assigning 10-bit encoded color to links, where the meaning of each color is implementation-specific. This administrative static link constraint is used to avoid or attract specific links for specific traffic types. For instance, it is possible to assign a blue color for links supporting encryption. This color object can be used to define specific Objective Function, such as selecting blue colored paths, or paths with maximum number of blue colored links, if encryption is an important criteria in the routing policy.

4.3. Security

Initially, security specification in RPL was not well elaborated in the RPL specification RFC [15]. Basically, RPL is expected to support message confidentiality and integrity. It has three basic security modes:

- *Unsecured*: In this mode, RPL control messages are sent without any additional security mechanisms; it could use other present security primitives to meet the application requirements.
- *Pre-installed*: in this mode, nodes joining a RPL instance have pre-installed keys that enable them to process and generate secured RPL messages.
- *Authenticated*: In authenticated mode, nodes have pre-installed keys as in pre-installed mode, but these pre-installed keys may only be used to join a RPL instance as a leaf.

Later, the IETF ROLL working group proposed in [26] a comprehensive security framework for routing over LLNs upon previous routing security protocols, and adapted them to fulfill the LLNs' constraints and requirements. The authors presented a set of security recommendations to be incorporated into LLN routing protocols. They firstly discussed the security issues in these networks. They also analyzed the security attacks that would threaten LLNs in terms of confidentiality, integrity, and availability. In addition, they proposed the countermeasures that could be considered to defend against these attacks. As an illustration, they demonstrated how this security framework can be applied to RPL. They pointed out the attacks that particularly threaten the RPL DAG management operations, namely the potential attacks that might affect the RPL control messages (DIO, DAO and DIS), and the information pertaining to the IPv6 hop-by-hop option header and the routing header. The most important security requirements that they recommended in this regards are integrity, authenticity, encryption and protection against the message replay attack. They also recommended to use per-message security mechanisms as an alternative to address these attacks.

Even though, the RPL security services proposed in [15] and [26] do not address all possible attacks and remain exposed to some threats that may compromise RPL security, such as a broadcasting fake messages by a compromised internal node. It results that security in RPL still deserves further investigation.

5. RPL performance evaluation

To evaluate the performance of the RPL protocol under different network settings, we deployed a real network composed of 30 TelosB motes comprising one sink node acting as the DAG root, and 29 RPL routers, all deployed

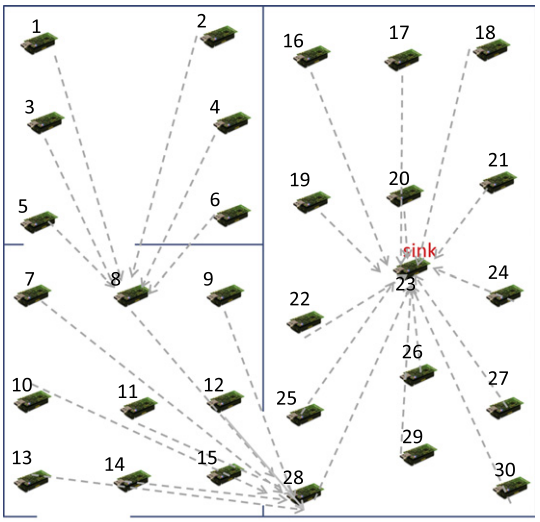

Fig. 6. Experimental testbed for multi-hop topology.

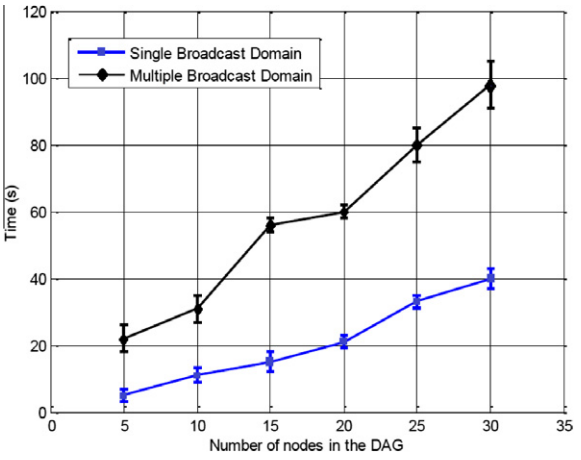

Fig. 7. Experimental convergence time for RPL networks.

in an indoor office environment. ContikiRPL implementation [27] was used in these experiments. The trickle timer parameters I_{min} and $I_{doubling}$ are set to 12 and 8, respectively, which gives a minimum interval for sending out control packets of 2^{12} ms (i.e. around 4 s) and a maximum interval of 2^{12+8} ms (i.e. around 17 min). The transmission power is set to −25 dBm. In all the experiments, we used the default Objective Function in Contiki which is MRHOF with ETX metric. We considered two scenarios: (i) a single broadcast domain scenario, where all nodes hear each other, and (ii) a multiple broadcast domain scenario, where nodes are placed in a multi-hop topology such that the routers are distant from each other with a range varying from 1 to 4 hops. Fig. 6 shows our experimental testbed for the evaluation of RPL in the multi-hop topology. We have deployed the motes in three rooms in the laboratory. We show in this figure one observed instantiation of the parent to child relationship within the formed DAG in the experiment. The node with ID 23 is the DAG root.

5.1. DAG convergence time

The DAG convergence time represents the time at which the DAG is completely constructed and all the nodes have joined the network. Fig. 9 shows the average measured convergence times for different network sizes, and for single and multiple broadcast domains. Each experiment is repeated five times and results are presented with 90% confidence interval (see Fig. 7).

We observe that the convergence time linearly increases with the number of nodes in the DAG for both single and multiple broadcast domains. However, the convergence time in the multiple broadcast domain is at least four times larger than that of the single broadcast domain. This illustrates the impact of the number of hops on the time needed to join the network. Further, we notice that the convergence time becomes remarkably large when the number of nodes increases. This is mainly due to three reasons: (i) the lossy nature of the channel, since control packets need to be retransmitted when they are lost. This confirms the results drawn in [28], where the authors demonstrated using COOJA simulator that the convergence time with a lossy channel is much larger than that with a perfect channel (ii) The duty cycle radio protocol used in ContikiMAC, which induces additional delays in particular when the topology grows, (iii) the impact of the trickle timer parameter ($I_{min} = 4$ s), which makes the DIO retransmissions occur after an important delay.

We have measured the convergence time of the state of the art Collection Tree Protocol (CTP) [29] using the TinyOS CTP implementation [30] in order to compare it with RPL. We have deployed a similar testbed in which we placed 10 TelosB motes in a single broadcast domain. We measured the convergence time which corresponds to the time that a node finds a parent in the tree topology. We found out that for a network composed of five nodes, the average convergence time is around 5 s, and for a network composed of 10 motes, the average convergence time is around 7 s. When comparing these results with those of RPL, the convergence time of RPL is slightly higher than that of CTP. However, tuning the internal parameters of each protocol such trickle timer in RPL or duty cycle may produce different results. In general, the network formation of the DAG RPL is comparable with the time needed to construct the tree in CTP. In [28], a comparative study between CTP and RPL is presented.

5.2. Power consumption

The power consumption represents the average energy consumed of each node in the DAG during the DAG construction process. Fig. 8 shows the average power consumption for different network sizes. The power consumption was measured during the joining process.

A straightforward observation is that the power consumption increases with the number of routers in the DAG, which is expected. We also observe that the multiple broadcast domain consumes more power in the network set-up process than the single broadcast domain does. However, the gap becomes smaller as the number of nodes increases and the average consumed energy converges to-

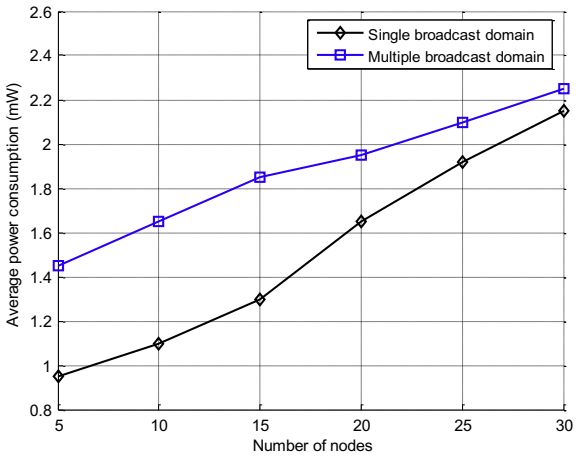

Fig. 8. Average power consumption.

wards 2 mW for both scenarios. This demonstrates that the effect of multi-hop becomes less important for large networks and that the number of nodes represents the major factor of energy dissipation.

5.3. Packet loss

For the quest to assess the reliability of RPL, we measured the packet loss ratio, which is defined as the ratio of the number of lost packets to the total number of packets. Packet losses occur when one or more data packets traveling across the DAG fail to reach their destinations. Each router randomly sends Hello data packets to the root at an average rate of 1 packet/min. We have chosen a large period to avoid overloading the network and prevent collisions. We have used the data collection tool of ContikiOS to collect the Hello data packets at the root. Fig. 9 depicts the packet loss ratio for different hop counts between the DAG root and a router mote.

Fig. 9 shows that the packet loss ratio for 1 and 2 hops is relatively low (between 1% and 4%). However, this ratio significantly increases with greater hop counts reaching 20% for 4 hops. This is due to the following reasons: (i) packet losses have a cumulative effect when the number of hops increases, due to increasing chances for a packet to get lost from one hop to another, (ii) link quality fluctuations of low-power and lossy links, which results in temporary losses of connectivity in particular for those disturbed by obstacles. This was the case of large hop counts in our experiment. These results raise questions about the effectiveness of the default Objective Function relying on the ETX metric to select routes. Considering more efficient link quality estimators such as 4-Bits [31] and F-LQE [32] is a promising approach to promote the reliability of RPL through the selection of more stable and higher-quality routes. This represents an open research issue in RPL design.

5.4. Packet delay

We evaluated the end-to-end delays in a RPL network to understand its timeliness behavior. We used the Ping application to measure the round-trip time between two nodes placed at a certain number of hops. The packet delay is defined as the duration between the transmission time of the Ping Request message and the reception time of the Ping Reply message. We have used the analyzer tools Wireshark [33] and Z-Monitor [34] for delay measurements. Fig. 10 shows the measured packet delays for different hop counts between the source and destination with a confidence interval of 90%.

Fig. 10 shows that the packet delay increases almost linearly with the number of hops between the source and destination. It varies from 1 s for single hop distant motes to 2.7 s for motes that are 4 hops away. The measured packet delays represent an acceptable real-time performance considering the low-power and resource limitation nature of sensor nodes. We alert the reader that such observed delays are not tightly dependant on the design of the RPL routing protocol as they are biased by other factors such as sleepy devices and the packet loss ratio; i.e. a higher loss ratio induces more retransmissions and thus increases the delay. Lower delays can be observed with better links and lower packet loss ratios. However, there

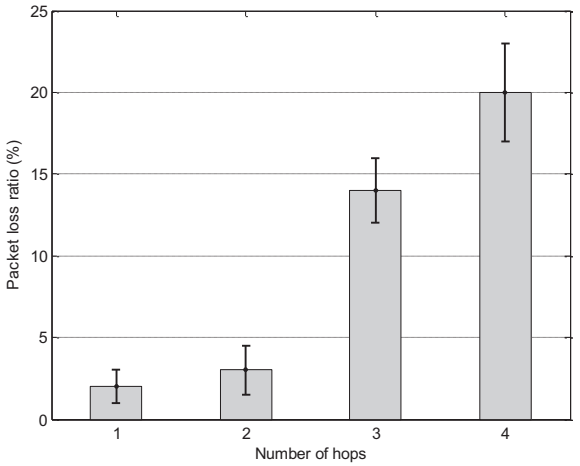

Fig. 9. Packet Loss Ratio for different hops.

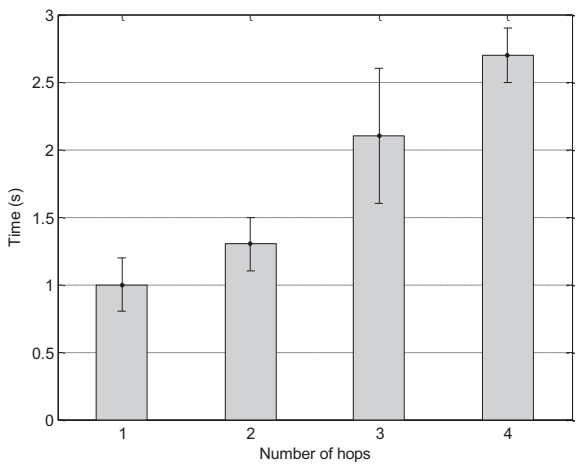

Fig. 10. Packet delay for different hop count.

is a room and a need to further improve the timeliness performance of lossy networks using RPL through the adoption of more sophisticated QoS mechanisms to optimize the route selection process and reduce end-to-end delays.

5.5. Fault tolerance

To evaluate the ability of RPL to respond to node failures, we forced the failure of some nodes (long term drop) in different depths and we evaluated the recovery time, which is the time required to establish a new route to the DAG root through another preferred parent instead of the failed parent.

From these experiments, we found that a node having its preferred parent failed finds another parent in different time duration varying between 24 and 65 s. By analyzing the exchanged messages during the node failure using WireShark and Z-Monitor sniffers, we notice that the node who had disconnected from the DAG due to the failure of its parent changes its preferred parent just after listening to a DIO sent from another neighbor that advertises a rank lower than the rank of its current parent. As we have previously mentioned, the DIO transmission is governed by a trickle timer, which explains the reason of having random recovery times in the experiments.

In RFC 6550 [15], RPL does not use proactive approaches for preventing faults and making recovery faster, but rather it favors the use of reactive approaches as they are more energy-efficient. Indeed, RFC 6550 states that "*In a general fashion, a detection mechanism that is reactive to traffic is favored in order to minimize the cost of monitoring links that are not being used*". Furthermore, the neighbor unreachability detection (NUD) is made not mandatory in RPL. ContikiRPL implementation does not include any active mechanism to probe neighbors regularly. The Objective Function in this implementation computes the rank of the node by adding the ETX of the preferred parent with the ETX of the link. When a failure occurs, it will increase the link ETX towards the preferred parent. When the failed parent gets its link-ETX attributed a high value, RPL will select another parent of lower rank and link ETX, assuming such a parent exists.

We also forced the failure of the DAG root, and we noticed that all the DAG was broken after a certain period (starting with the failed root). The nodes continue to send DIOs but the links between the nodes are all broken. If the instance contains more than one DAG root, the nodes will join another root in order to regain connectivity.

As a conclusion, it appears that the specification of proactive mechanisms in RPL to predict and detect node failures would be of interest for real-time applications as it will help accelerate recovery, and prevent inaccessibility times.

6. Related works

The ROLL working group has proposed several drafts tightening the design of the RPL protocol. However, there are still several issues that are left open for further investigation. Although RPL is still in its early days, several research works have tackled some of these open issues. In what follows, we first present the most relevant routing protocols for low power and lossy networks and we compare them against RPL. Then, we survey the most recent works related to RPL.

6.1. Comparison of existing LLN routing protocols

In this Section, we present the well-known routing protocols proposed for LLNs and related to standard protocols. The features of these routing protocols are populated in the Table 1 for comparative purpose with RPL. The Table presents an overview on how different routing protocols fit under different categories and also compares their characteristics. In what follows we summarize the details of these routing protocols.

In 2005, and before the birth of 6LoWPAN, ZigBee standard protocol was considered as the most prominent technology for LLNs. It has been characterized by its famous Cluster-Tree protocol [38] as a hierarchical routing protocol that uses link-state packets to form a two-tier cluster-based network, where each cluster is governed by one Cluster Head (CH). The ZigBee Cluster-Tree network is self-organized and it supports fault-tolerance and self-repair mechanisms. The cluster-tree is formed by several parent-to-child associations between ZigBee Routers until a certain depth. Clusters are supposed to operate in exclusive time slots to avoid interference between them. In 2007, with the emergence with ZigBee Pro 2007, the cluster-tree routing was no longer supported for complexity of maintenance issues, and the standard has adopted flat and mesh routing based on AODV.

With the emergence of IPv6-based networks for LLNs, a new wave of routing protocols has emerged. Several Internet drafts have been proposed [6,39,7]. Hilow [6] is one of the first routing protocols proposed as an Internet draft for 6LoWPAN networks that mainly addresses scalability issues. HiLow is a hierarchical and on-demand routing protocol that takes advantage of the 6LoWPAN capabilities in terms of dynamic assignment of 16-bit short addresses. The use of 16-bit short addressing scheme allows for using hierarchical routing which is very scalable. HiLow relies on parent to child relationship to construct the network tree. Once a 6LoWPAN node wants to join the network, it either associates with an existing neighbor parent router, which assigns it a 16-bit short address, or initiates a new 6LoWPAN network if no other neighbor 6LoWPAN device is found. Like in any other tree routing, any router just needs to determine whether the destination of the packet should be forwarded to parent (ascendant) or child (descendant) node based on the address, which is expressed as $Adrr = PA * MCN + N$ where PA is the parent address, MCN is the maximum number of child routers of each the parent node is allowed to associate with, and N is the index of the Nth child node. This addressing mechanism is very similar to ZigBee Cluster-Tree addressing scheme.

Later, LOAD was proposed in [39] as another Internet draft and represents a flat on-demand routing protocol designed for 6LoWPAN network. It is based on AODV and supports mesh network topologies. However, LOAD does not use the destination sequence number used in AODV.

Table 1
Comparison of LLN routing protocols.

	Topology	Type	Algorithm	Hello messages	Local repair	Mobility	Scalability	Memory usage	Energy usage	Supported traffic	IPv6 support	Available simulators	Citations
AODV	Flat	On demand	Distance vector	Use	Use	Mobile	Medium	High	High		P2P	No	
AODVjr	Flat	On demand	Distance vector	No use	No use	Mobile	Low	Low	Low		P2P	No	
ZigBee Cluster Tree	Hierarchical	Proactive	Link State	No use	No use	Static	High	High	High	MP2P P2MP	No	OPNET	open-zb [35]
TinyAODV	Flat	On Demand	Distance vector	No use	No use	Mobile	High	Low	Low	P2P	Yes	Avrora OMNET++ TOSSIM	TinyOS-1.x [36]
Hilow	Hierarchical	On demand	Tree based	No use	No use	Static Mobile	High	Low	Low	MP2P P2MP	Yes		
LOAD	Flat	On demand	Distance vector	No use	No use	Static Mobile	High	Low	Low	P2P	Yes		
Dimo-Low	Flat	On-demand		Use	No use	Static mobile	Low	High	High	P2P	Yes		
CTP	Hierarchical	Proactive	Source routing	No use	Use	Mobile	High	Low	Low	MP2P P2MP	No	TOSSIM	TinyOS-2.x [29]
Hydro	Hierarchical flat	On-demand	Source routing	No use	No use	Static	High	Medium	High	MP2P P2MP P2P	Yes		blip1.0 [37]
RPL	Hierarchical flat	Proactive	Distance–vector source-routing	No use	Use	Static Mobile	High	Low	Low	MP2P P2MP P2P	Yes	Cooja	TinyRPL [28] ContikiRPL [27]

For ensuring loop freedom, only the destination of a route should generate a Route Reply (RREP). LOAD uses LQI (Link Quality Indicator) and the number of hops as routing metrics for route selection. In case of a link failure, the upstream node of the broken link initiates a local route by using the route discovery mechanism. LOAD uses the link layer acknowledgments instead of Hello messages to save energy while keeping track of route connectivity.

In the same trend, Kim et al. proposed DYMO-low in [7] as a flat routing protocol. DYMO-low operates on the link layer to create a mesh network topology of 6LoWPAN devices. It performs route discovery and maintenance. Dymo-Low does not use local repair and uses Hello messages instead. It uses sequence numbers to ensure loop freedom.

On the other side, Hydro [5] is the default routing protocol that was proposed in the 6LoWPAN Berkeley implementation known as *BLIP*. This routing protocol represents a hybrid mechanism that provides both centralized control and local agility. Hydro uses a distributed algorithm to form a DAG for routing data from router nodes to border routers. Nodes report topologies periodically to the border router, allowing it to maintain a global view on the topology. All the nodes forward packets to the border router which forwards them to the appropriate destinations.

It can be noted that the completeness of RPL makes it a winning and promising routing protocol as compared to its competitors. In fact, one reason behind the success of RPL routing protocol in comparison with the other protocols is that it provides comprehensive features including support of various types of traffic (MP2P, P2MP and P2P) and its ability to directly connect to Internet nodes with global IPv6 addresses. This turns RPL very flexible and can be easily tuned for different applications' requirements. Furthermore, RPL constructs the topology proactively and thus discards the need to broadcast RREQ messages, which is used in several AODV-based protocols. In addition, it provides the benefits of both mesh and tree routing protocols as it supports both hierarchical and flat topologies. In addition, RPL combines the distance–vector and the source routing paradigms, and supports local and global repair, which make it suitable for fault-tolerant applications. In addition, the notion of Objective Function in RPL offers a great flexibility for supporting various application requirements and enables QoS-award routing. For a detailed overview of routing protocols in wireless ad hoc and sensor networks, we refer the reader to the survey paper [40].

Thanks to the attractive features of RPL, several recent research works have extensively investigated this protocol in several fronts.In the remaining part of this section, we survey the latest works related to RPL, which are roughly classified into two categories: (1) modeling and performance evaluation, (2) amendments and integration of RPL.

6.2. Modeling and performance evaluation of RPL

Several works have proposed simulation and experimental models for the sake of experimenting with RPL and evaluating its performance.

6.2.1. Simulation models

In [41], the authors tackled the problem of routing loops that may occur in DAGs when nodes choose another parent and increase their ranks. They have implemented a simulation model of RPL under NS-2 simulator, and simulated it for a large-scale network composed of 1 root and 1000 nodes deployed in a large space. The authors evaluated the RPL performance in terms of the network convergence time after a rank increase, and the routing message overhead. The simulation results showed that the majority of the routing loops are resolved rapidly and this process causes the generation of only a small number of DIOs. However, in some cases, the rank increase operation by one node triggers a sequence of events that generates multiple routing loops and the network may require a significant time to converge to a stable loop-free state. The affected nodes may produce a significant number of DIOs. They recommended some precautionary measures that can be used to avoid routing loops, and demonstrated through simulations how these measures affect the convergence time and routing message overhead involved in reaching a loop-free state following a rank increase operation. The authors found out that the turmoil imposed by the default loop avoidance mechanisms is more significant than routing loops. For that purpose, the paper concludes that loop avoidance mechanism are not practical for large-scale RPL networks.

In [42], the authors compared between the P2P routing based on RPL and the simple shortest path routing algorithm. They demonstrated via NS-2 simulations of a large-scale network that the P2P routes within a DAG are significantly sub-optimal as compared to the minimum cost (shortest) routes, mainly when the source and destination are close to each other. This result gets even worse when the DAG size increases. However, the authors did not propose any solution to this problem.

In [43], The authors proposed an implementation of the RPL protocol tailored for the Advanced Metering Infrastructure (AMI). They used ETX as the default link metric, and proposed a new ETX-based technique for rank computation. They validated their RPL routing protocol design for AMI networks using NS-2 simulation and compared its performance to AODV protocol, with a simulation scenario comprising 1000 meter nodes, and one gateway node (similarly to [41,42]). Simulation results showed that The Packet Delivery Ratio and the end-to-end delay for RPL are not sensitive to the distance. The authors evaluated the performance of RPL for AMI networks under different levels of the shadowing effect and results showed that RPL produces a satisfactory performance for both inward and outward unicast data traffic in AMI networks.

In [44], the authors presented a performance evaluation study of RPL through an OMNET++ simulation model. They used ETX as a default link metric to build the DAG. Simulation results showed that the performance of P2P routing in RPL for the considered topology is close to the shortest path between source and destination and it is better when the root is in the middle. In addition, they found that 90% of the nodes need to store less than 20 entries in their routing tables. They found also that for a node closer to sink, the data packet amount is much higher than control packet. For leaf nodes the amount of control packets are more than data packets. They demonstrated the efficiency of the trickle timer in controlling the packet overhead and stabilizing the network. In addition, the paper demonstrated the significant effect of the global repair duration on the number of control packet overhead. However, the simulations were performed for a small-scale network, preventing the generalization of results for large-scale networks.

In [13], the authors proposed a performance evaluation of the RPL protocol in the context of smart grid applications. Using OMNET++ simulation model of RPL as in [44], they demonstrated the capability of the trickle timer in RPL to bound the control overhead and reduce communication latency. In addition, they demonstrated that RPL quickly performs local repair of link outage and provides a path quality close to an optimized shortest path for an outdoor environment. Also, they showed that in RPL Point-to-Point (P2P) routing, the path quality is not drastically worse than the shortest path. This result even improves when the DAG root is located in the middle of the network. The results showed that, in most cases, the total end-to-end delay is in the order of milli-seconds. In [45], the authors have presented a performance evaluation study of the RPL routing protocol using the Contiki COOJA simulator [46]. They mainly evaluated the network overhead, the throughput and the end-to-end delays for two network sizes of 20 and 100 motes. They concluded that RPL leads to a fast network set-up and limited communication delays. The authors reported a network set-up time, i.e. the time required to let the control overhead drop from 100% to about 25%, of about 10 min for 20 nodes, which is relatively high for such a medium-scale networks. They also observed that DAO messages represent the main factor that increases the control overhead of RPL. It can be concluded that RPL is open to further amendments to optimize the network formation process in terms of convergence time and control overhead.

We alert the reader that simulation models are appropriate for providing insights on the RPL protocol behavior; however, in general, these models are not able to report the exact performance of the protocol as they rely on emulated channel models, which may differ from real channels, and make abstraction on hardware resources and their usage. As such, experimental models are of paramount importance to assess the real protocol behavior and performance. In the next section, we provide the main efforts that contributed to the implementation of experimental prototypes of the RPL protocol.

6.2.2. Experimental models

ContikiRPL [27] is the first real-world implementation of RPL developed under Contiki [46] operating system. In fact, the authors have developed a comprehensive framework for simulation, experimentation, and evaluation of RPL routing mechanisms under Contiki operating system. One of the main features of ContikiRPL is that it provides a simple programming interface for designing and evaluating Objective Functions. In [47], the authors evaluated ContikiRPL in terms of power-efficiency and implementation complexity, and they proved that the packet delivery ratio is high and the battery lifetime may last for several

years. The measured buffer size was shown to be low as compared to the available memory space of TmoteSky motes.

In [28], the authors proposed TinyRPL, a real implementation of RPL under TinyOS 2.x. They evaluated the performance of RPL and BLIP through real experimentation using TelosB motes and compared the performance of RPL with the CTP protocol. They found that the performance of TinyRPL is comparable to that of the CTP protocol in terms of Packet Reception Ratio (PRR) and Control Traffic overhead. The paper also reported that TinyRPL provides a packet loss ratio less than 2% and a latency no larger than 200*ms*.

In [48], the authors tested the interoperability of ContikiRPL and tinyRPL implementations. They presented experiences showing that the two implementations mixed together can inter-operate. They also concluded that the interoperability is not enough for guaranteeing the correct network behavior and they identified problems related to the variation of some components that may affect packet delivery and network churn. However, they did not test the interoperability of the downstream routing capabilities that RPL provides as they did not consider DAO messages.

The authors in [49] have designed and implemented the IPv6 stack including RPL for NanoQplus which is a multithreaded operating sytem for small wireless sensors and actuators. They implemented RPL with on Objective Function based on minimizing the number of hops. They performed experiments on real motes and showed that their implementation fits the memory constraints of senor motes. In addition, they demonstrated that RPL achieves a packet delivery ratio above 91%.

6.3. Amendment and integration of RPL

In [10], the authors raised the need for achieving the interpretability between networks in the context of the Internet-of-Things. They proposed a communication stack for powerline communication (PLC) based on open standards, namely IEEE 802.15.4/6LoWPAN protocol stack with the use of RPL as the underlying routing protocol. The authors designed and developed a PLC testbed and they implemented RPL in Contiki operating system [46]. It has been shown that RPL is adequate for PLCs since it enables efficient P2MP and MP2P traffic. The experimental study carried on demonstrated how to achieve the interpretability between IEEE 802.15.4 and PLC nodes. However, the experimental tests were limited to ping and global repair operations. Other RPL operations were not tested. In addition, the authors have used an Objective Function based on ETX, which is not suitable for PLC networks with real-time requirements.

In [11], the authors considered the problem pertaining to the lack of a proper broadcast mechanism in RPL and proposed some amendment mechanisms. The authors presented two broadcast mechanisms that aim at exploiting the existing routing state of RPL, while not requiring additional state maintenance. The first mechanism is the parent-based flooding, which restricts the retransmission of broadcast packets to those received from a parent node. The second mechanism is the preferred parent flooding, which restricts the retransmissions of broadcast packets to those received from the RPL router's preferred parent. The paper presented a NS-2 simulation study of RPL in which they analyzed several properties of unicast and multicast traffic. Simulations showed that the total retransmission overhead of broadcasted messages from the DAG root is much lower when a node retransmits packets from all of its parents than when it retransmits only those transiting through the preferred parent. However, the delivery ratio in the two proposed solutions is very low and needs further improvements.

In [12], the authors proposed other broadcast mechanisms, in addition to those presented in [11], and evaluated their performance. As part of this evaluation, the proposed solutions were compared against MPR Flooding, which is a broadcast mechanism widely employed in ad hoc networks. It has been shown that the delivery ratio and traffic overhead improved with the proposed broadcast mechanisms, at the cost of the need for the use of a Neighbor Discovery mechanism instead of signaling, which may slow down the message delivery.

In [50], the authors investigated the problem of sink mobility within the DAG and proposed a distributed and weighted moving mobility strategy for sinks in RPL networks. The main design objective was to improve the network lifetime by moving the sinks towards the leaf nodes. The idea consists in moving the sinks towards routers with highest weight, which is a function of the residual energy, number of neighbors and number of hops. Simulation results of 1600-node network with three mobile sinks showed that the proposed strategy increases significantly the network lifetime, since when moving the sink based on the weighted approach, the data traffic becomes more balanced.

In [51], the authors discussed the problem of node compromise in RPL and its impact on the network security. The authors proposed some countermeasures to overcome this problem in the update of the version number of the DAG which initiates the DAG construction. They proposed a new security service, which prevents a node from illegitimately increasing the version number and compromise illegal rank values.

In [52], the authors proposed RB-MAC, a preamble sampling MAC protocol specifically designed for the RPL protocol. This protocol is receiver based, and dynamically chooses the next hop according to the channel conditions and the sensor status. They demonstrated that their scheme is resilient against lossy links and reduces the number of retransmissions. They also showed that their scheme outperforms the other preamble sampling MAC protocols in terms of energy and delay. However, they did not integrate their proposed MAC in RPL which makes its evaluation incomplete.

7. Discussions and future challenges

Table 2 summarizes the main features of the RPL protocol.

In summary, RPL is a very promising routing protocol for LLNs as it provides a great level of flexibility to deal with different requirements of underlying applications. In

Table 2
RPL routing protocol main features.

Feature	Description
Target network	LLN; IPv6/6LowPAN networks
Routing type	Source-routing, Distance–vector
Topology	Mesh/hierarchical based on DAGs
Traffic flows	MP2P, P2MP and P2P
Message update	Trickle timer
Control messages	DIO, DAO, DIS
Neighbor discovery	like IPv6 ND mechanisms
Transmission	Unicast and multicast
Metrics and constraints	Dynamic, based on OF and Rank
Modes	Storing and non-storing

this paper, we have surveyed the most important features of the RPL routing protocol, with the aim to provide researchers with a quick yet comprehensive introduction of RPL.

Although the RPL specification has gained sufficient maturity, there are still several open research problems not addressed by the ROLL working group that have been stated as being out of the scope of the RFC specification. One of the most important issues still left open is the specification of the Objective Function. This is indeed a fundamental in RPL as the Objective Function greatly impacts the routing behavior and the path selection policies. In the current RFC specification, there is no clear relation on how to map an Objective Function to rank computation. In addition, RPL does not specify which metric/constraint to use to optimize an Objective Function. Several research areas in this respect can be defined, including (but not limited to) the specification of Objective Functions for certain class of applications tailored to their requirements (e.g. reliability, real-time), the design of new Objective Functions, the mapping between rank computation and Objective Functions.

In terms of topology control, the root selection mechanism in a DODAG is also another open problem. More work is needed to optimize the default path quality, as some study such in [42] argued the sub-optimality of paths under RPL. The mode of operation of RPL instances is not described in the IETF drafts. In fact, the mechanisms that allow a node to select a certain instance and to move from one instance to another, the time when the DODAG root increments the *DODAGVersionNumber*, the rules of routing outside an instance are not specified.

Security in RPL is immature. It is indeed important to define security mechanisms such as key management and authentication techniques to secure the join of an instance in the network. The process by which the key is received by a node as well as the key establishment and maintenance processes are not specified. In addition, the configuration of security mechanisms for the processing of the incoming packets is still an open issue.

Acknowledgment

This work is supported by the National Plan for Sciences and Technology under the Computer Science Research Program for the Grant No. 08-INF200-8.

References

[1] G. Montenegro, N. Kushalnagar, J. Hui, D. Culler, Transmission of IPv6 Packets Over IEEE 802.15.4 Networks. Internet Proposed Standard RFC 4944, September 2007.
[2] IEEE 802.15.4 Standard, Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-rate Wireless Personal Area Networks (LR-WPANs), IEEE Standard for Information Technology, IEEE-SA Standards Board, March 2003.
[3] J.W. Hui, D.E. Culler, IP is dead, long live IP for wireless sensor networks, in: The 6th International Conference on Embedded Networked Sensor Systems (SENSYS'08).
[4] N. Kushalnagar, G. Montenegro, C. Schumacher, IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals," IETF RFC: 4919, August 2007.
[5] A. Tavakoli, HYDRO: A Hybrid Routing Protocol for Lossy and Low Power Networks, IETF Internet Draft: draft-tavakoli-hydro-01, September 2009.
[6] K. Kim, S. Yoo, J. Park, S.D. Park, J. Lee, Hierarchical Routing over 6LoWPAN (HiLow), IETF: Internet Draft: draft-deniel-6lowpan-hilow-hierarchical-routing-00.txt, vol. 38, December 2005.
[7] K. Kim, S. Park, I. Chakeres, C. Perkins, Dynamic MANET On-demand for 6LoWPAN (DYMO-low) Routing, Internet Draft: draft-montenegro-6lowpan-dymo-low-routing-03, June 2007.
[8] The Internet Engineering Task Force (IETF), 2010. <http://www.ietf.org/>.
[9] Routing Over Low Power and Lossy Networks (ROLL), 2004. <https://datatracker.ietf.org/wg/roll/charter/>.
[10] C. Chauvenet, Tourancheau, B. Genon-Catalot, D. Goudet, M.P.-E. Pouillot, A communication stack over PLC for multi physical layer IPv6 networking, in: Smart Grid Communications (SmartGridComm), First IEEE International Conference on, November 2010.
[11] T. Clausen, U. Herberg, Multipoint to point and broadcast in RPL, in: 13th International Conference on Network-Based Information Systems, September 2010, pp. 1–6.
[12] T. Clausen, U. Herberg, Comparative study of RPL-enabled optimized broadcast in wireless sensor networks, in: Sixth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP2010), Brisbane, Australia, December 2010.
[13] J. Tripathi, J. de Oliveira, J. Vasseur, Applicability study of RPL with local repair in smart grid substation networks, in: Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on, November 2010, pp. 1–6.
[14] J.-P. Vasseur, A. Dunkels, Interconnecting Smart Objects with IP – The NextInternet, Morgan Kaufmann, 2010.
[15] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, RPL: IPv6 Routing Protocol for Low Power and Lossy Networks, IETF Request for Comments 6550, March 2012.
[16] J. Vasseur, Terminology in Low power and Lossy Networks, IETF Internet Draft: draft-ietf-roll-terminology-04.txt, September 2010.
[17] O. Gnawali, P. Levis, The ETX Objective Function for RPL, IETF Internet Draft: draft-gnawali-roll-etxof-00, 2010. <http://tools.ietf.org/html/draft-gnawali-roll-etxof-00>.
[18] E.P. Thubert, RPL Objective Function 0, IETF Internet Draft: draft-ietf-roll-of0-03, 2010. <http://tools.ietf.org/html/draft-ietf-roll-of0-12>.
[19] A. Conta, S. Deering, E.M. Gupta, Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification, RFC: 4443, March 2006.
[20] E.J. Martocci, P.D. Mil, N. Riou, W. Vermeylen, Building Automation Routing Requirements in Low-Power and Lossy Networks, RFC: 5867, March 2010.
[21] A. Brandt, J. Buron, G. Porcu, Home Automation Routing Requirements in Low-Power and Lossy Networks, Request for Comments: 5826, March 2010.
[22] E.K. Pister, E.P. Thubert, S. Dwars, T. Phinney, Industrial Routing Requirements in Low-Power and Lossy Networks, RFC: 5673, March 2009.
[23] E.M. Dohler, E.T. Watteyne, E.T. Winter, E.D. Barthel, Routing Requirements for Urban Low-Power and Lossy Networks, Request for Comments: 5548, March 2009.
[24] S. Deering, W. Fenner, B. Haberman, Multicast Listener Discovery (MLD) for IPv6, RFC: 2710, 1999. <http://www.faqs.org/rfcs/rfc2710.html>.
[25] J. Vasseur, M. Kim, K. Pister, N. Dejean, D. Barthel, Routing Metrics used for Path Calculation in Low Power and Lossy Networks, IETF

Internet Draft: draft-ietf-roll-routing-metrics-09, 2010. <http://tools.ietf.org/html/draft-ietf-roll-routing-metrics-09>.
[26] T. Tsao, R. Alexander, M. Dohler, V. Daza, A. Lozano, A Security Framework for Routing over Low Power and Lossy Networks, IETF Requirement Draft for Routing over Low Power and Lossy Networks (ROLL), Work in Progress.
[27] N. Tsiftes, J. Eriksson, N. Finne, O. Fredrik, J. Höglund, A. Dunkels, A Framework for Low-Power IPv6 Routing Simulation, Experimentation, and Evaluation, SIGCOMM'10, New Delhi, India, November 2010, pp. 479–480.
[28] J. Ko, S. Dawson-Haggerty, O. Gnawali, D. Culler, A. Terzis, Evaluating the Performance of RPL and 6LoWPAN in TinyOS," Workshop on Extending the Internet to Low Power and Lossy Networks (IP+SN), April 2011.
[29] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, P. Levis, Collection tree protocol, in: 7th ACM Conference on Embedded Networked Sensor Systems (SenSys), November 2009, pp. 1–14.
[30] R. Fonseca, O. Gnawali, K. Jamieson, P. Levis, TEP 119, TinyOS Documentation, 2006.
[31] R. Fonseca, O. Gnawali, K. Jamieson, P. Levis, Four bit wireless link estimation, in: Proceedings of the Sixth Workshop on Hot Topics in Networks (HotNets VI), 2007.
[32] N. Baccour, A. Koubaa, H. Youssef, M.B. Jamâa, D. do Rosário, M. Alves, L.B. Becker, F-lqe: A Fuzzy Link Quality Estimator for Wireless Sensor Networks, EWSN, 2010, pp. 240–255.
[33] Wireshark Tool. <http://www.wireshark.org/>.
[34] (2011) Z-Monitor: A Monitoring Tool for IEEE 802.15.4 WPANs. <http://www.z-monitor.org>.
[35] Open-ZB Open-Source Toolset for the IEEE 802.15.4/ZigBee Protocols. <http://www.open-zb.net>.
[36] TinyAODV Implementation under TinyOS-1.x Source Code. <http://tinyos-1.x/contrib/hsn/apps/TraceRouteTestAODV>.
[37] BLIP Implementation. <http://docs.tinyos.net/tinywiki/>.
[38] S.C. Erge, ZigBee/IEEE 802.15.4 Summary, 2004. <http://staff.ustc.edu.cn/ustcsse/papers/SR10.ZigBee.pdf>.
[39] K. Kim, S. Daniel Park, G. Montenegro, S. Yoo, N. Kushalnagar, 6LoWPAN Ad Hoc On-Demand Distance Vector Routing (LOAD), IETF Internet Draft: draft-daniel-6lowpan-load-adhoc-routing-03, 2007.
[40] T. Watteyne, A. Molinaro, M. Richichi, M. Dohler, From MANET To IETF ROLL Standardization: A Paradigm Shift in WSN Routing Protocols, IEEE Communications Surveys and Tutorials, in press.
[41] W. Xie, Goyal, M. Hosseini, H. Martocci, J. Bashir, Y. Baccelli, A.E. Durresi, Routing loops in DAG-based low power and lossy networks, in: Advanced Information Networking and Applications (AINA), 24th IEEE International Conference on, April 2010, pp. 888–895.
[42] W. Xie, M. Goyal, H. Hosseini, J. Martocci, Y. Bashir, E. Baccelli, A. Durresi, A performance analysis of point-to-point routing along a directed acyclic graph in low power and lossy networks, in: 2010 13th International Conference on Network-Based Information Systems, 2010, pp. 111–116.
[43] D. Wang, Z. Tao, J. Zhang, A. Abouzeid, RPL based routing for advanced metering infrastructure in smart grid, in: Communications Workshops (ICC), 2010 IEEE International Conference on, May 2010, pp. 1–6.
[44] J. Tripathi, J. de Oliveira, J. Vasseur, A performance evaluation study of RPL: routing protocol for low power and lossy networks, in: Information Sciences and Systems (CISS), 2010 44th Annual Conference on, May 2010, pp. 1–6.
[45] N. Accettura, L. Grieco, G. Boggia, P. Camarda, Performance analysis of the RPL routing protocol, in: Mechatronics (ICM), 2011 IEEE International Conference on, April 2011, pp. 767–772.
[46] Contiki Operating System. <http://www.sics.se/contiki/>.
[47] N. Tsiftes, J. Eriksson, A. Dunkels, Poster Abstract: Low-Power Wireless IPv6 Routing with ContikiRPL, IPSN'10, Stockholm, Sweden, April 2010, pp. 12–16,.
[48] J. Ko, J. Eriksson, N. Tsiftes, S. Dawson-Haggerty, A. Terzis, A. Dunkels, D. Culler, ContikiRPL and TinyRPL: Happy Together, Workshop on Extending the Internet to Low Power and Lossy Networks (IP+SN), April 2011. <http://www.sics.se/fros/cooja.php>.
[49] J. Jeong, J. Kim, P. Mah, Design and implementation of low power wireless IPv6 routing for NanoQplus, in: 13rd International Conference on Advanced Communication Technologies (ICACT), February 2011.
[50] L.B. Saad, B. Tourancheau, Sinks mobility strategy in IPv6-based WSNs for network lifetime improvement, in: International Conference on New Technologies, Mobility and Security IFIP NTMS, 2011.
[51] A. Dvir, T. Holczer, L. Buttyán, VeRA – Version number and rank authentication in RPL, in: 7th IEEE International Workshop on Wireless and Sensor Networks Security, IEEE, 2011.
[52] M. Akhavan, T. Watteyne, A. Aghvami, Enhancing the performance of RPL using a receiver-based MAC protocol in lossy WSNs, in: Telecommunications (ICT), 2011 18th International Conference on, may 2011, pp. 191 –194.

Olfa Gaddour was born in 1983. Currently, she is a Ph.D student at the National Engineering School of Sfax since January 2010. Her research activity is conducted within CES research unit. She has received the Engineering degree in Networks and Communication, from the Higher School of Telecommunications (Sup'Com), Tunis, Tunisia in 2007 and the Master degree in New Technologies of Dedicated Computer Science Systems, from the National Engineering School of Sfax, in 2009. Her current research interests are in the field of Wireless Sensor Networks (WSN) and the 6LowPAN protocol. They are focused on the QoS routing and the RPL routing protocol. She has several publications in many conferences (e.g. ICDCS'SN workshop).

Anis Koubaa was born in 1977. He received the Engineering degree in telecommunications from the Higher School of Telecommunications, Tunis, Tunisia, in 2000, and the M.Sc. and Ph.D. degrees in computer science from the National Polytechnic Institute of Lorraine (INPL), Nancy, France, in 2001 and 2004, respectively. He is currently an Assistant Professor at the College of Computer Science and Information Systems, Al-Imam Muhammad Ibn Saud University, Riyadh, Saudi Arabia, and a Research Associate at the CISTER/IPP-HURRAY Research Group, Porto, Portugal. He is actively working on the design of large-scale wireless sensor netwoks and cyber-physical systems, while maintaining QoS, security, mobility and reliability. He is particularly interested in the assessment and improvement of the IEEE 802.15.4/ZigBee standard protocol stack for large-scale wireless sensor networks. He has driven the research efforts in the context of the ART-WiSe and open-ZB research frameworks that have contributed to the release of an open-source toolset of the IEEE 802.15.4/ZigBee protocol stack. He is the Chair of the TinyOS ZigBee Working Group, whose purpose is achieving a standard-compliant open-source implementation of ZigBee over TinyOS. In addition, he is involved in the CONET European Network of Excellence, in particular with the research cluster COTS-based architecture for QoS in large-scale distributed embedded systems. He is also actively participating as a reviewer or a program committee member in some reputed international journals, conferences and workshops dealing with real-time networks, quality-of-service, wireless communications, and related issues.