RPL: IPv6 Routing Protocol for Low Power and Lossy Networks

Tsvetko Tsvetkov
Betreuer: Alexander Klein
Seminar Sensorknoten: Betrieb, Netze und Anwendungen SS 2011
Lehrstuhl Netzarchitekturen und Netzdienste, Lehrstuhl Betriebssysteme und Systemarchitektur
Fakultät für Informatik, Technische Universität München
Email: tsvetko.tsvetkov@gmail.com

ABSTRACT

Today, Low Power and Lossy Networks (LLNs) represent one of the most interesting research areas. They include Wireless Personal Area Networks (WPANs), low-power Power Line Communication (PLC) networks and Wireless Sensor Networks (WSNs). Such networks are often optimized to save energy, support traffic patterns different from the standard unicast communication, run routing protocols over link layers with restricted frame-sizes and many others [14].

This paper presents the IPv6 Routing Protocol for Low power and Lossy Networks (RPL) [19], which has been designed to overcome routing issues in LLNs. It implements measures to reduce energy consumption such as dynamic sending rate of control messages and addressing topology inconsistencies only when data packets have to be sent. The protocol makes use of IPv6 and supports not only traffic in the upward direction, but also traffic flowing from a gateway node to all other network participants.

This paper focuses on the employment of RPL in WSNs and gives a brief overview of the protocol's performance in two different testbeds.

Keywords

RPL, Sensor Network, Low-Power Network, Lossy Link, Routing, Data Collection, Data Dissemination

1. INTRODUCTION

Over the last years WSNs have become a very important and challenging research field. Such networks consist of spatially distributed autonomous devices which usually operate untethered and additionally have limited power resources. This limits all aspects of their construction, architecture and communication capabilities. Several studies such as [2] and [11] reveal the impact of wireless lossy links on the overall reliability, power efficiency and maximum achievable throughput. There are cases where a network can only achieve approximately the half of the throughput of the corresponding lossless network. Moreover, lossy links effect the power consumption due to packet retransmissions and broadcasting. Zhao and Govindan [20] have estimated the impact of such links and concluded that 50% to 80% of the communication energy is wasted in overcoming packet collisions and environmental effects in indoor and outdoor scenarios.

Such LLNs are additionally characterized by connections that are not restricted to two endpoints. Many scenarios may include Point-to-Multipoint (P2MP) or Multipoint-to-Point (MP2P) traffic patterns. Such networks are also known for their asymmetric link properties. The communication is realized by using a separate uplink and downlink. Because each unidirectional link provides only one way traffic, the bandwidths in the two directions may differ substantially, possibly by many orders of magnitude.

In order to meet these requirements and challenges, the Internet Engineering Task Force (IETF) ROLL Working Group designed a new routing protocol, called RPL [18]. The highest goal of RPL is to provide efficient routing paths for P2MP and MP2P traffic patters in LLNs. The protocol successfully supports the latest version of the Internet Protocol which results from the research made by different organizations.

The IP for Smart Objects (IPSO) Alliance has made a great effort to promote the use of IP for small devices [4]. It is the leading organization for defining the *Internet of Things* and supports the use of the layered IP architecture for small computers. The cooperation with the IETF organization further accelerates the adoption of IPv6 on LLNs. IETF has specified the IPv6 over Low power Wireless Personal Area Networks (6LoWPAN) standard [12] which supports the idea of applying IPv6 even to the smallest machines. In this way, devices with limited hardware resources are able to participate in the Internet of Things. This standard also enables the use of standard web services without application gateways.

The rest of this paper is organized as follows. Section 2 gives an overview of RPL's basic features and describes the terminology of the protocol. Section 3 discusses topics such as topology construction and structure of the used control message. An introduction to RPL's loop avoidance and detection mechanisms is presented in Section 4. Section 5 gives information about the different routing metrics. Section 6 describes how the support of P2MP traffic is realized and Section 7 gives an overview of the protocol's performance. Finally, the paper is concluded in Section 8.

2. RPL DESIGN OVERVIEW

RPL is a distance vector routing protocol for LLNs that makes use of IPv6. Network devices running the protocol are connected in such a way that no cycles are present. For this purpose a Destination Oriented Directed Acyclic Graph (DODAG), which is routed at a *single* destination, is built. The RPL specification calls this specific node a DODAG root. The graph is constructed by the use of an Objective

doi: 10.2313/NET-2011-07-1_09

Function (OF) which defines how the *routing metric* is computed. In other words, the OF specifies how routing constraints and other functions are taken into account during topology construction.

In some cases a network has to be optimized for different application scenarios and deployments. For example, a DODAG may be constructed in a way where the Expected Number of Transmissions (ETX) or where the current amount of battery power of a node is considered. For this reason, RPL allows building a logical routing topology over an existing physical infrastructure. It specifies the so-called RPL Instance which defines an OF for a set of one or more DODAGs.

The protocol tries to avoid routing loops by computing a node's position relative to other nodes with respect to the DODAG root. This position is called a *Rank* and increases if nodes move away from the root and decreases when nodes move in the other direction, respectively. The Rank may be equal to a simple hop-count distance, may be calculated as a function of the routing metric or it may be calculated with respect to other constraints.

The RPL specification defines four types of control messages for topology maintenance and information exchange. The first one is called DODAG Information Object (DIO) and is the main source of routing control information. It may store information like the current Rank of a node, the current RPL Instance, the IPv6 address of the root, etc. The second one is called a Destination Advertisement Object (DAO). It enables the support of down traffic and is used to propagate destination information upwards along the DODAG. The third one is named DODAG Information Solicitation (DIS) and makes it possible for a node to require DIO messages from a reachable neighbor. The fourth type is a DAO-ACK and is sent by a DAO recipient in response to a DAO message. The RPL specification defines all four types of control messages as ICMPv6 information messages with a requested type of 155. This new type has been officially confirmed by IANA [6]. Note that the last two are not further described in this paper.

Another important fact about the protocol's design is the maintenance of the topology. Since most of devices in a LLN are typically battery powered, it is crucial to limit the amount of sent control messages over the network. Many routing protocols broadcast control packets at a fixed time interval which causes energy to be wasted when the network is in a stable condition. Thus, RPL adapts the sending rate of DIO messages by extending the Trickle algorithm [10]. In a network with stable links the control messages will be rare whereas an environment in which the topology changes frequently will cause RPL to send control information more often.

3. UPWARD ROUTING

Upward routing is a standard procedure which enables network devices to send data (e.g. temperature measurements) to a common data sink, also called sometimes a gateway or root node. In a typical WSN scenario, nodes periodically generate data packets (e.g. each minute) which have to find their way through the network. In this section, the RPL topology construction process is discussed and the structure of a DIO message is presented.

3.1 DIO Message Structure

As previously mentioned, a DIO message is the main source of information which is needed during topology construction. Figure 1 represents the structure of the message.

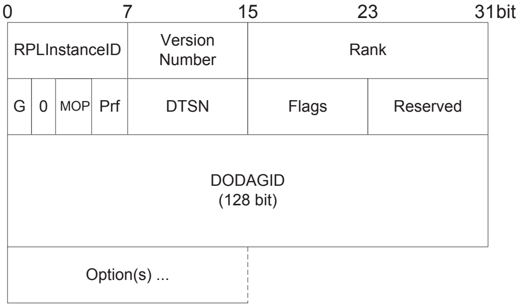

Figure 1: DIO Message Structure

A DIO first allows a node to discover the RPL Instance by storing the corresponding one in the first data field. The second and the third field include the DODAG Version and the Rank of the sender of the message. Section 3.2 describes the purpose of the RPL Instance, version number and the Rank update process. The next byte includes the 'G' flag which defines whether a DODAG is grounded. Grounded means that it can satisfy an application-defined goal. If it is not set, the DODAG is said to be floating. This may happen when a DODAG is disconnected from the rest of the network and supports only connectivity to its nodes. The MOP field (size of 3 bits) is set by the DODAG root and defines the used mode of operation for downward routing. Section 6 gives more information about it. The Prf field (size of 3 bits) defines how preferable the root node is compared to other root nodes. Such a node is identified by the DODAGID field. The last used field is DTSN and it is needed for saving a sequence number. Such a number is maintained by the node issuing the DIO message and guarantees the freshness of the message.

A DIO message may be extended by the use of options. In this paper, only the DODAG Configuration option is discussed, since it plays a crucial role for parameter exchange. Figure 2 outlines its structure.

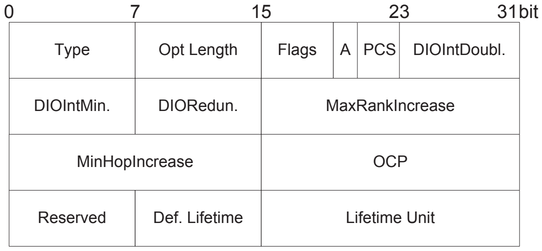

Figure 2: DODAG Configuration Option

The first two bytes always include the type (0x04) and the option's length (14 bytes). The next byte includes the 'A' flag, the Flags field and the PCS field. They will be not further discussed since they are not important for this paper. The next two bytes define the maximum timer value τ_{max} and the minimum timer value τ_{min} needed for the trickle timer setup. More information can be found in Section 3.2. The DIORedun. field defines a constant k used for suppressing DIO messages. If a node receives more than k DIOs, it may suppress them. The MaxRankIncrease field defines an upper limit for the Rank and is further discussed in Section 4. The MinHopIncrease field stores the minimum increase of the Rank between a node and any of its parent nodes. It creates a trade-off between the maximum number of hops and the hop cost precision. The DODAG Configuration Option concludes with the OCP field (OF identification), the Default Lifetime for all routes and the Lifetime Unit. The latter one defines in seconds the length of a time unit.

3.2 Constructing Topologies

In general, there are three[1] types of nodes in a RPL network. The first type are root nodes which are commonly referred in literature as gateway nodes that provide connectivity to another network. The second type are routers. Such nodes may advertise topology information to their neighbors. The third type are leafs that do not send any DIO messages and have only the ability to join an existing DODAG.

The construction of the topology starts at a root node that begins to send DIO messages. Each node that receives the message runs an algorithm to choose an appropriate parent. The choice is based on the used metric and constraints defined by the OF. Afterwards each of them computes its own Rank and in case a node is a router, it updates the Rank in the DIO message and sends it to all neighboring peers. Those nodes repeat the same steps and the process terminates when a DIO message hits a leaf or when no more nodes are left in range. A possible Rank computation is shown in Equation 1. In this example, *floor(v)* evaluates v to the greatest integer less than or equal to v.

$$DAGRank(rank) = floor\left(\frac{rank}{MinHopIncrease}\right) \tag{1}$$

However, in most sensor node deployments several data collection points (root nodes) are needed. Thus, three values have to be considered in order to uniquely identify a DODAG: (1) RPL Instance ID for identification of an independent set of DODAGs, optimized for a given scenario; (2) DODAG ID which is a routable IPv6 address belonging to the root; (3) DODAG version number which is incremented each time a DODAG reconstruction is needed. The RPL specification defines the combination of those three values as a DODAG Version. An example is shown in Figure 3. Each of the three constructed topologies may be positioned in different rooms where the construction of a single DODAG is impossible. Due to the fact that the three graphs belong to the same instance, their construction is realized in a similar way (e.g. by considering ETX values).

The RPL specification distinguishes between three logical sets when building upward routes. First, the candidate neighbor set which includes all reachable nodes. They may belong to different DODAG Versions. For example, in Figure 3 node 10 may store node 11 in its candidate neighbor set. Second, the parent set which is a subset of the candidate neighbor set. It includes only nodes that belong to the same DODAG Version. When a node stores a neighbor into the parent set, it becomes attached to the given DODAG. The last one contains one[2] element: the most preferred next hop taken from the parent set. Note that a node may belong to more than one RPL Instance. In this case, it must join a DODAG for each RPL Instance.

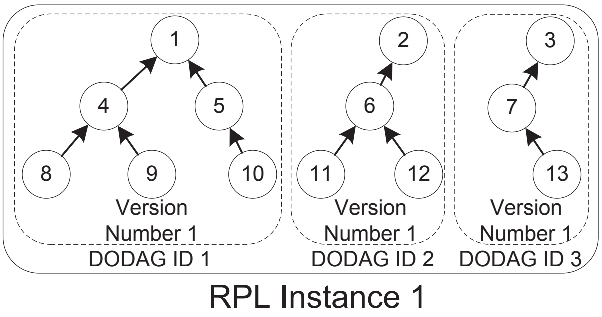

Figure 3: RPL Topology Partition

As previously mentioned, RPL dynamically adapts the sending rate of its control DIO messages. To achieve this, two values need to be used: one for defining the minimum sending time interval τ_{min} and another for defining the maximum sending interval τ_{max}. Whenever the sending timer expires, RPL doubles it up to the maximum value τ_{max}. Whenever RPL detects an event which indicates that the topology needs active maintenance, it resets the timer to τ_{min}. Such events are a found inconsistency when forwarding a data packet, joining of a new node, leaving the current DODAG and triggering topology repair.

4. ROUTING LOOPS

The formation of routing loops is a common problem in all kinds of networks. Due to topology changes caused by failure or mobility, a node may pick a new route to a given destination. If the new route includes a network participant which is a descendant, loops may occur. This leads to network congestion, packet drops, energy waste and delays. However, a quick and reliable detection of such topology inconsistencies is not a sufficient solution for LLNs. For example, even in a complete static sensor node deployment a malfunctioning antenna of a node may cause frequent changes of the node's distance to the root. Child nodes may be picked as next hops by their parents and a topology repair mechanism may be triggered. This leads to further energy consumption and waste of bandwidth. Therefore, a routing protocol for LLNs has to define a loop avoidance strategy considered during topology construction.

4.1 Avoidance Mechanisms

So far, it was only said that a Rank represents a node's position in the graph and that router nodes forward DIO control messages for topology maintenance. However, such messages are sent in a multicast manner to the neighboring nodes. If each node in range accepts such messages and takes the sender of the DIO into account for computation of the parent set, it may happen that child nodes are selected as best next hops. Consider the example shown in Figure 4. The topology is constructed by taking ETX into account. Further, the Rank value equals the ETX metric.

[1] Virtual roots are not considered in this paper.

[2] RPL also allows multiple equally preferred parents.

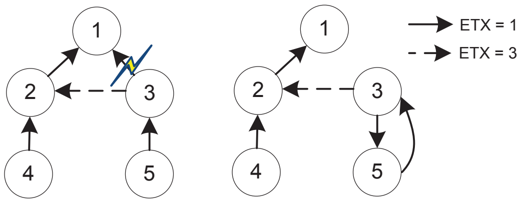

Figure 4: Loop Creation

If node 3 processes a DIO control message from node 5 it will consider it as a valid possible parent. If the link between the root node and node 3 fails, node 3 will simply pick its child node as next hop and a loop will occur, since the ETX cost to the root is 3. Compared to the other possible way to the root (through node 2) the ETX cost is 4.

Therefore, a RPL node does not process DIO messages from nodes deeper (higher Rank) than itself because such nodes may belong to its sub-DODAG. Even if node 4 is reachable for node 3, it should not be considered as next hop. Instead, node 3 has to declare an invalid state, poison its routes and join the DODAG Version again.

RPL also limits the movement of a node within a DODAG Version. Since moving up in a DODAG does not present the risk of creating a loop but moving down might, the RPL specification suggests that a node must never advertise within a DODAG Version a Rank higher than $Rank_{Lowest} + Rank_{MaxInc}$. $Rank_{Lowest}$ is the lowest Rank the node has advertised within a DODAG Version and $Rank_{MaxInc}$ is a predefined constant received via a DIO. Figure 5 illustrates a simple topology where node 1 is the DODAG root.

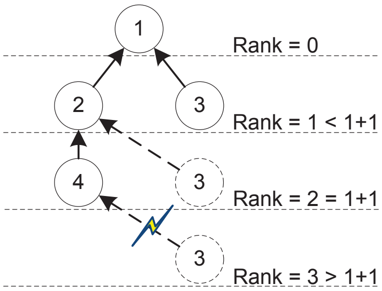

Figure 5: Movement Limitation within a DODAG Version

Let node 2 and 3 have a Rank of 1 and node 4 a Rank of 2. In addition, let $Rank_{MaxInc} = 1$. If node 3 moves away and increases its Rank to 2, it is allowed to choose node 2 as next hop. However, if node 3 moves even further away and increases its Rank to 3, it must not pick node 4 as a next hop. The RPL document does not specify what node 3 should do in this situation. A possible solution is to join another DODAG or advertise a DODAG rebuild request.

4.2 Detection Mechanisms

Usually, in LLNs nodes rarely generate data traffic which makes keeping the topology consistent all the time wasteful in terms of energy consumption. For example, Koen Langendoen et al. introduced in their paper [9] a large-scale experiment in which several sensor nodes were disseminated over a potato field programmed to send data every 60 seconds. Even for this relatively large time interval they have experienced loss of battery power after three weeks of operation. If the topology is kept consistent all the time, it may happen that nodes experience lack of energy after a shorter period of time. Thus, changes in connectivity need not to be addressed until data packets are present.

RPL loop detection uses additional information that is transported in the data packets. It places a RPL Packet Information in the IPv6 option field which is updated and examined on each hop. There are five control fields within the RPL Packet Information. The first one indicates whether the packet is sent in a upward or downward direction. The second one reports if a Rank mismatch has been detected. It is used to indicate whenever the Rank of the sender, stored in the packet, is lower than the Rank of the receiver. The RPL specification suggests that packets should not be immediately dropped if such a inconsistency is detected. Instead, the packet should be forwarded. However, if an inconsistency is detected on the packet for the second time, it must be dropped and the trickle timer must be reset. In this way, route repair is triggered. The third one is the forwarding error field and is used by a child node to report that it does not have a valid route to the destination of the packet. The last two are the Rank of the sender and the RPL Instance ID.

5. RPL METRICS

Many of today's routing protocols use link metrics that do not take a node's current status into account. The status includes typical resources such as CPU usage, available memory and left energy. This may be crucial for LLNs where network devices are usually battery powered and have limited hardware resources. For example, if a chain topology occurs in a sensor network deployment, the last node before the root will usually experience a higher traffic load and forwarding overhead than the others. If in Figure 6 all nodes frequently generate data packets and send them to the root, node 2 may fail very quickly due to the lack of energy. Even if the link between node 3 and 7 is not an optimal solution, it may be reasonable to send data packets through node 7 since it may offer a more stable node condition. Otherwise, if node 2 fails it may take some time until node 3 picks node 7 as next hop and packet drop may occur.

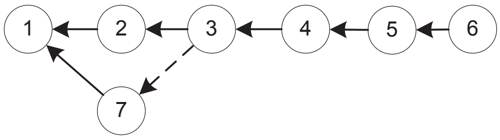

Figure 6: Chain Topology

Therefore, the ROLL Working Group defined several metric categories in [7] that may be considered when selecting the next hop. Some of the metrics defined in the document are carried in messages that are optional from the point of view of a RPL implementation. They have to be additionally specified and are not further described in this paper. In the following two possible metric computations from the RPL metrics document will be shortly discussed.

5.1 Node Energy Consumption

This method suggests that a node should consider the energy level of its neighbors before picking them as possible parents. For this purpose, two units of information are used: (1) the type of the node which indicates how it is supplied with power and (2) the Energy Estimation (EE). The RPL metric specification defines three possible states for the first information field: powered, on batteries and scavenger. If a network device is powered it means that it may be the root node connected to a PC or it may be some sort of special data collector (e.g. cluster-heads in hierarchical routing). Such nodes may report a maximum EE value and, in general, are preferable during parent selection. If a node is on batteries, it has to compute its EE value by using Equation 2. The $Power_{now}$ value is the remaining energy and $Power_{max}$ is the power estimation reported at boot up.

$$EE = \frac{Power_{now}}{Power_{max}} \cdot 100 \tag{2}$$

However, if a node derives energy from external sources [13] it may report EE as a quantity value that is computed by dividing the amount of power the node has acquired by the power consumed. This may be a rough estimation of how much load a node experiences for a given period of time.

5.2 ETX

This metric is an approximation of the expected number of transmissions until a data packet reaches the gateway node. A node that is one hop away from the root, with perfect signal strength and very little interference, may have an ETX of 1. Another node with a less reliable connection to a root may have a higher ETX.
ETX is a bidirectional single-hop link quality computation between two neighbor nodes [1]. For the computation a metric called Packet Reception Rate (PRR) is used. PRR is calculated at the receiver node for each window ρ of received packets, as follows:

$$PRR(\rho) = \frac{Number\ of\ received\ packets}{Number\ of\ sent\ packets} \tag{3}$$

In literature the value computed in Equation 3 is also defined as *in-quality*, which is the quality from node A to node B measured by node B by counting the successfully received packets from A among all transmitted. In this paper it will be called PRR_{down}. For the actual ETX estimation the *out-quality* is further needed. This is the in-quality estimated by node A and is defined as PRR_{up} at node B. In this way node B can calculate ETX as shown in Equation 4:

$$ETX = \frac{1}{PRR_{down} \cdot PRR_{up}} \tag{4}$$

6. DOWNWARD ROUTING

The support of downward routing is another important key feature of the protocol. By supporting P2MP traffic it is possible for a network administrator to control nodes that are even not in range. This is very useful for performance evaluation purposes where usually several hundred nodes are spread over a large area. If such traffic is not supported, even the slightest changes, such as a timer value, may require to find the node, disconnect it from the network and upload a new code image. Moreover, if the idea of the Internet of Things is considered, P2MP becomes a must for LLN routing protocols [17].
The RPL specification defines two modes of operation for supporting P2MP. First, the non-storing mode which makes use of source routing. In this mode each node has to propagate its parent list up to the root. After receiving such topology information, the root computes the path to the destinations. Second, the storing mode which is fully stateful. Here, each non-root and non-leaf network participant has to maintain a routing table for possible destinations. Note that any given RPL Instance is either storing or non-storing.

6.1 DAO Message Structure

As mentioned in Section 2, DAO messages are used by RPL nodes to propagate routing information in order to enable P2MP traffic. Figure 7 represents the structure of a DAO message.

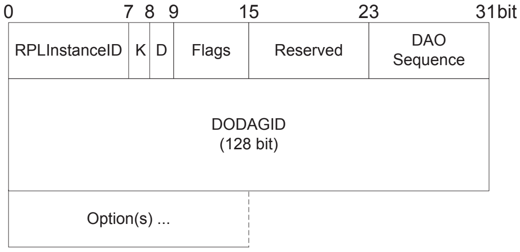

Figure 7: DAO Message Structure

Similar to the DIO message, the DAO message includes an RPL Instance ID. This is the same one that the node has learned from a received DIO. The next field is the Flags field where only the first two bits are used. The first one is the 'K' flag which indicates whether the sender of the DAO expects to receive a DAO-ACK in response. The second one is the 'D' flag which indicates if the DODAGID field is present. Due to the fact that the DODAGID field represents the IPv6 address of the root it may be omitted if there is only one root node present. The DAO Sequence field is a sequence number that is incremented for each outgoing DAO message by the sender. The sequence number ensures the freshness of a DAO message and is echoed back by the parent when DAO-ACKs are used.
The message can be further extended by the use of options. In this paper, only two will be discussed: the Target option and the Transit Information option. The first one is used to indicate a target IPv6 address, prefix or multicast group. In terms of routing, it represents reachability information. RPL defines the structure of it, as shown in Figure 8.

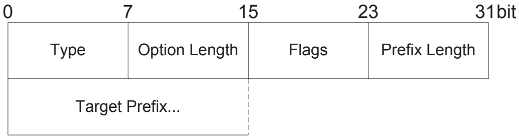

Figure 8: DAO Target Option

First of all, it stores the type of the option (0x05). Afterwards, the length of the option and the length of the prefix are included. In the latter case, the number of valid leading bits of the routing prefix is meant. The last field is the target prefix and it may identify, for example, a single node or a whole group of nodes that can be reached via a common prefix. The Flags field is experimental and is not used.
The second type is the Transit Information option. It is used to indicate attributes for a path to one or more destinations. Destinations are indicated by one or more Target options that precede the Transit Information option(s). Figure 9 outlines its structure.

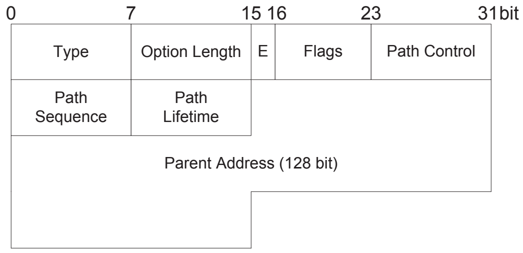

Figure 9: DAO Transit Information Option

The first field represents the type of the option and is always set to 0x06. The second field is the Option Length field which indicates if the Parent Address field is present. Note that in case of storing mode the IPv6 address of the parent may be omitted. The next field is the Flags field where only the first bit is used. The 'E' flag indicates whether the parent router redistributes external targets into the RPL network. Such targets may have been learned by an external protocol. However, they do not play a crucial role for this paper. The next three fields are needed for reachability control. First, the Path Control field is used to limit the number of parents to which a DAO message may be sent. Second, the Path Sequence field indicates if a Target option with updated information has been issued. Third, the Path Lifetime defines how long a prefix for a destination should be kept valid. The time is measured in Lifetime Units and is implementation specific.

6.2 Non-Storing Mode

In the non-storing mode each node generates a DAO message and sends it to the DODAG root. The time generation interval in which DAO messages are sent depends on the implementation. However, the RPL specification suggests that the needed delay between two DAO sending operations may be inversely proportional to the Rank. In this way, if a node is far away from the root it will generate DAOs more often than a node that is closely positioned to the gateway. Furthermore, each node has to extend the DAO message by using the aforementioned Transit Information option. In the Parent Address field the IPv6 address of a parent node is stored. It should be kept in mind that a typical non-storing node may use multiple Transit Information options in order to report its complete parent set to the root node. The resulting DAO message is sent directly to the DODAG root along the default route created during parent selection. Figure 10 illustrates this process.

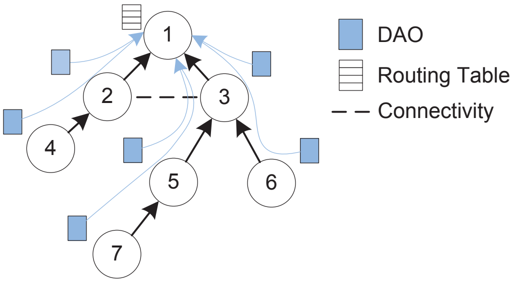

Figure 10: RPL Non-Storing Mode

Usually, intermediate nodes inspect DAO messages and compare the stored path sequence number with the last seen. In this way, a node can distinguish between stale and up-to-date routing information.
After collecting the needed information, the root pieces the downward route together. If it needs to send a data packet to a given destination the IPv6 Source Routing header is used. Thus, network nodes can easily forward a data packet until it reaches the given destination or the IPv6 Hop Limit reaches 0.

6.3 Storing Mode

Similar to the non-storing mode, the storing mode also requires the generation of DAO messages. The configuration of the timer triggering such messages may be implemented in the same way as it was mentioned above. However, a DAO is no longer propagated to the DODAG root. Instead, it is sent as unicast to all parent nodes which maintain additional downward routing tables. Figure 11 gives an overview of this process.

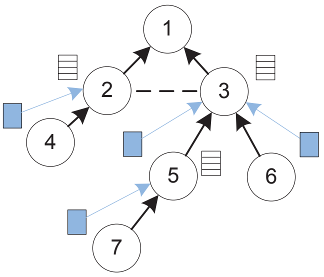

Figure 11: RPL Storing Mode

When a node sends a DAO message it has to keep the Parent address field in the Transit Information option empty since a node's responsibility is not to advertise its parent set, but to announce prefixes that are reachable through it. If the device is a router it has to use the Target option in order to advertise a prefix. In case it has multiple prefixes to advertise, it must extent the DAO by multiple Target options.
If a data packet is sent from the DODAG root node, it must be sent only to all one-hop neighbors. Afterwards, through table look-up the packet is routed downwards until it reaches its destination or the Hop Limit in the IPv6 header reaches 0. Such networks may also be hierarchically organized and route aggregation may be performed. Moreover, a sub-DODAG may run another RPL Instance which makes it possible to

combine storing and non-storing mode. In this way, nodes with constraints regarding memory may be grouped together and operate only in non-storing mode.

7. PERFORMANCE EVALUATION

Gnawali et al. [8] introduced a RPL implementation, called TinyRPL. In their work they use the Berkeley Low-power IP (BLIP) stack in TinyOS 2.x [15] which interacts with their protocol implementation. More precisely, the control plane of TinyRPL communicates with the BLIP stack which offers a forwarding plane implementation. It should be kept in mind that BLIP also offers an implementation in TinyOS of a number of IP-based protocols such as TCP and UDP. In this way, during one test run several transport protocol configurations can be evaluated and compared.

TinyRPL is further compared with the Collection Tree Protocol (CTP) [5], the de-facto routing protocol standard for TinyOS. For this purpose, a 51-node TelosB [3] testbed scenario is build where only one node is acting as a root. Overall, they use two network configurations: one that has a data packet generation interval of 5 seconds and another that has a data packet time interval of 10 seconds. For each of them the protocols are tested against each other and an estimation of the packet reception ratio and the average number of control packets is made. Each testbed run lasts 24 hours. Since CTP uses ETX for topology maintenance, the OF of TinyRPL is set to use the same method for metric computation. In order to make a fair comparison the downstream routing options are disabled since the standard CTP configuration does not define a mechanism similar to the one realized with DAO messages.

In all test runs both protocols managed to achieve a reception rate of approximately 99 %. The amount of control traffic reported also stays at the same level since both protocols make use of the trickle timer and are evaluated in static scenarios.

8. CONCLUSION

LLNs and, in particular, WSNs are rapidly emerging as a new type of distributed systems, with applications in different areas such as target tracking, building environment, traffic management, etc. However, to achieve a reliable communication, to guarantee a high delivery ratio and to be at same time energy efficient requires special mechanisms realized at the network layer.

As a result, RPL was specified and developed in order to overcome these requirements. The protocol is an end-to-end IP-based solution which does not require translation gateways in order to address nodes within the network from the outside world. Moreover, the use of IPv6 allows deploying RESTful web services for sensor networks [16]. In this way, a client (e.g. PC connected to the root) may initiate requests by using HTTP to nodes within the network which will return the appropriate responses. It is even easier to use such a feature, since RPL defines in its specification the support of downward traffic. Because of P2MP a root node can easily propagate such requests to the nodes.

It should also be mentioned that RPL may run on nodes that have limited energy and memory capabilities. The protocol dynamically adapts the sending rate of routing control messages which will be frequently generated only if the network is in a unstable condition. In addition, the protocol allows the use of source routing when P2MP is needed which reduces the memory overhead on intermediate nodes.

RPL also allows optimization of the network for different application scenarios and deployments. For example, it may take into account the link quality between nodes or their current amount of energy which makes it an efficient solution for WSN deployments.

9. REFERENCES

[1] N. Baccour, A. Koubaa, M. B. Jamaa, H. Youssef, M. Zuniga, and M. Alves. A comparative simulation study of link quality estimators in wireless sensor networks. Technical Report TR-09-03-30, open-ZB, 2009.

[2] A. Cerpa, J. Wong, L. Kuang, M. Potkonjak, and D. Estrin. Statistical model of lossy links in wireless sensor networks. In *Information Processing in Sensor Networks, IPSN 2005*, pages 81–88, 2005.

[3] Crossbow Technology. TelosB datasheet. http://www.willow.co.uk/TelosB_Datasheet.pdf, 2010.

[4] A. Dunkels and J. P. Vasseur. Why IP. IPSO Alliance White Paper #1, 2008.

[5] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection Tree Protocol. Technical Report SING-09-01, Stanford University, 2009.

[6] Internet Assigned Numbers Authority (IANA). Internet Control Message Protocol version 6 (ICMPv6) Parameters. http://www.iana.org/assignments/, 2011.

[7] M. Kim, J. P. Vasseur, K. Pister, N. Dejean, and D. Barthel. Routing Metrics used for Path Calculation in Low Power and Lossy Networks. Internet draft, http://datatracker.ietf.org/wg/roll/, 2011.

[8] J. Ko, S. Dawson-Haggerty, O. Gnawali, D. Culler, and A. Terzis. Evaluating the performance of RPL and 6LoWPAN in TinyOS. In *Proceedings of the Workshop on Extending the Internet to Low power and Lossy Networks (IP+SN)*, 2011.

[9] K. Langendoen, A. Baggio, and O. Visser. Murphy loves potatoes: Experiences from a pilot sensor network deployment in precision agriculture. In *The International Workshop on Parallel and Distributed Real-Time Systems*, 2006.

[10] P. Levis, N. Patel, D. Culler, and S. Shenker. Trickle: A self-regulating algorithm for code maintenance and propagation in wireless sensor networks. In *Proceedings of the USENIX NSDI Conference*, pages 15–28, San Francisco, CA, USA, 2004.

[11] Y. Li, J. Harms, and R. Holte. Impact of lossy links on performance of multihop wireless networks. In *Computer Communications and Networks, 2005. ICCCN 2005*, pages 303–308, 2005.

[12] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler. RFC 4944: Transmission of IPv6 packets over IEEE 802.14 networks, 2007.

[13] S. J. Roundy. *Energy scavenging for wireless sensor nodes with Focus on vibration to electricity conversion.* PhD thesis, University of California at Berkeley, 2003.

[14] Routing Over Low power and Lossy networks (ROLL). Description of Working Group. http://datatracker.ietf.org/wg/roll/charter/, 2011.

[15] The TinyOS Alliance. Poster abstract: TinyOS 2.1, adding threads and memory protection to TinyOS. In *Proceedings of the 6th ACM Conference on Embedded Networked Sensor Systems (SenSys'08)*, Raleigh, NC, USA, 2008.

[16] TinyOS community. BLIP tutorial. http://docs.tinyos.net/index.php/BLIP_Tutorial, 2010.

[17] J. P. Vasseur. The Internet of Things: Dream or Reality. SENSORCOMM 2010: The Fourth International Conference on Sensor Technologies and Applications, 2010.

[18] J. P. Vasseur, N. Agarwal, J. Hui, Z. Shelby, P. Bertrand, and C. Chauvenet. RPL: IPv6 Routing Protocol for Low power and Lossy Networks. IPSO Alliance, 2011.

[19] T. Winter, P. Thubert, A. Brandt, T. Clausen, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, and J. P. Vasseur. RPL: IPv6 Routing Protocol for Low power and Lossy Networks. ROLL Working Group, 2011.

[20] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, SenSys '03, pages 1–13, New York, USA, 2003.