MT-RPL: a cross-layer approach for mobility support in RPL

Cosmin Cobârzan, Julien Montavont, Thomas Noel

EAI Endorsed Transactions
on Internet of Things
Research Article ICST.ORG

MT-RPL: a cross-layer approach for mobility support in RPL

Cosmin Cobârzan[1], Julien Montavont[1,*] and Thomas Noël[1]

[1]ICube laboratory (UMR CNRS 7357), University of Strasbourg, France

Abstract

Low Power and Lossy Networks (LLNs) are inherently dynamic - nodes move or experience link perturbations. Routing packets in LLNs is generally performed by the IETF IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL). To face the dynamics of LLN, RPL is helped by external mechanisms such as Neighbor Unreachability Detection (NUD) or Bidirectional Forwarding Detection (BFD). In this article, we focus our analysis on mobility support. We first show that NUD and BFD fail to mitigate node disconnection. We therefore propose a new cross-layer protocol operating between the MAC and routing layers known as Mobility-Triggered RPL (MT-RPL). MT-RPL has been implemented in Contiki OS and is evaluated together with NUD and BFD through an extensive experimentation campaign. Results show that our solution significantly reduces the disconnection time, which increases the packet delivery ratio from the mobile node to the root and reduces control traffic in the network.

Received on 17 March 2016; accepted on 21 June 2016; published on 01 December 2016

Keywords: Internet of Things, RPL, Dynamics, Mobility, Reproducible Experiments

doi:10.4108/eai.1-12-2016.151712

1. Introduction

Low-Power and Lossy Networks (LLN) are a new class of constrained wireless networks that allow a set of objects (sensors, actuators, etc.) to exchange relevant data in a multi-hop fashion. Interconnecting LLNs together with Internet opened the road to a large variety of applications and gave birth to what is now called the Internet of Things.

In this context, seamless mobility support is one of the keys to a widespread adoption of LLNs. First, a large variety of applications, ranging from target tracking [1] to wildlife monitoring [2] require the support of node mobility. Additionally, LLNs will be fully integrated into the future 5G networks and as such should cope with the notion of ubiquitous connectivity. In a general view, node mobility can be managed with three different approaches: relaying, Software Defined Networking (SDN) and routing [3]. In this article, we investigate how mobility support in LLNs could be achieved with the IPv6 routing protocol for Low-Power and Lossy Networks (RPL) [4]. This protocol, standardized by the IETF, is further detailed in Section 2.

RPL is designed to cope with network dynamics while maintaining connectivity by suggesting the usage of three unreachability detection mechanisms: Neighbor Unreachability Detection (NUD) [5], Bidirectional Forwarding Detection [6] and hints from lower layers via Layer 2 (L2) triggers such as [7]. Naturally, we could rely on those mechanisms to detect the movement of nodes and update the routes accordingly. However, we have shown in [8] that both NUD and BFD are unable to prevent mobile nodes from being disconnected for long period of time, which significantly increases the overall packet loss together with the contention at the MAC layer. In addition, L2 triggers are only a general framework that allows layer 2 to offer its services to layer 3 and vice versa. Those observations lead us to propose an innovative cross-layer protocol known as Mobility-Triggered RPL (MT-RPL) [8]. The present article is the first synthesis of this solution, that presents all of its concepts together with a thorough experimental evaluation.

MT-RPL is an implementation of L2 triggers, which operates alongside RPL at the routing layer, and

*Corresponding author. Email: montavont@unistra.fr

leverages X-Machiavel [9] operations at the MAC layer. X-Machiavel is a preamble sampling MAC protocol which favors mobile node's access to the transmission resources. Our preliminary performance evaluation [8] showed that MT-RPL significantly reduces the disconnection time, increases the packet delivery ratio while limiting the energy consumption. Those observations were however based on results obtained by simulations. It is quite delicate to simulate properly the characteristics of wireless communications together with mobility. Due to their instability, wireless links may lead to a constantly changing network topology, making organization of nodes a very difficult task and endangering MAC and network layer operations. Furthermore, some of the reasons why node disconnection occurs in the first place are closely related to implementation, platform or operating system specifics that are most often ignored in simulators. In this document, we chose an entirely empiric approach in order to further investigate and validate the effectiveness of our cross-layer protocol in comparison to BFD and NUD. We therefore implemented NUD, BFD and MT-RPL in Contiki OS and performed an extensive experimentation campaign with mobile robots. All experimentations are made on the Equipex FIT IoT-LAB [10] experimental platform, which is a large scale deployment of open wireless sensor network platform. To the best of our knowledge, this article also represents the first experimental evaluation of RPL with mobile nodes.

The rest of this article is organized as follows. We give a brief introduction to RPL in Sect. 2. Sect. 3 presents an overview of mobility management solutions and focuses on how RPL behaves in presence of mobile nodes. Our contribution MT-RPL is described in Sect. 4. Experimental parameters and performance evaluation in the FIT IoT-LAB experimental platform are detailed in Sect. 5. Finally, Sect. 6 concludes this article.

2. RPL basics

The IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [4] is the IETF standard to compute and disseminate IPv6 routes in LLNs. In RPL, routes are built along a DODAG, oriented to the root and shaped by an objective function. The graph is built after sending and receiving new ICMPv6 messages: DODAG Information Object (DIO), DODAG Information Solicitation (DIS) and DODAG Destination Advertisement Object (DAO). The root of the graph may act as a border router between the LLN and an external IPv6 domain such as the Internet. The root starts building the DODAG when it sends the first DIO in the neighborhood. Any node that receives a DIO will attach to the graph by computing a rank and build a parent set (a list of potential next hops to the root). From the parent set the node will select a preferred parent according to the advertised objective function. The default forwarding rule uses this preferred parent as next hop. Now, the attached nodes can advertise further the DODAG by sending their own DIO in their neighborhood, at intervals given by the trickle timer [11]. Nodes that do not want to wait for the next DIO can speed up the attachment process once they send a DIS requesting information about neighboring DODAG. After receiving a DIS, nodes already attached to a DODAG reply with a DIO and so, the requesting node will be able to attach to the DODAG. Once upward routes are established (i.e. the default route toward the root), optional downward routes can be build thanks to DAO messages. DAO will advertise the nodes destination information to the root, establishing point-to-point and point-to-multipoint communications. Before a node changes its preferred parent, a no-path DAO is sent to the former parent in order to remove downward routes that have been established through this parent.

3. Mobility support in RPL

3.1. Motivation

Mobility support allows a node to maintain ongoing communications and initiate future communications while on the move. We can find in the literature three categories of mobility support solutions: relaying, Software Defined Networking (SDN) and routing [3]. Solutions based on relaying use a dedicated node that tracks the movements of mobile nodes and acts as a relay station to forward incoming traffic to their current locations. Outgoing traffic are generally also transmitted toward the relay station before being forwarded to the final destination. The most well-known relaying solution is the Mobile IPv6 protocol standardized by the IETF. However, such solution creates a single point of failure (the relay station) and was not designed to cope with the characteristics of LLNs [12]. Solutions based on SDN will use the network controller to dynamically configure address translation and forwarding rules on access routers. For example, an access router can rewrite the destination address (translation rule) for packets destined to a mobile node with its current IPv6 address before forwarding them to its current access router (forwarding rule). SDN-based solutions have the advantage to distribute the data plane (the traffic no longer goes through a single router) but centralize the control plane (into the network controller) and increase the complexity of router operations in a way not suitable for constrained nodes.

Solutions based on routing consist of updating routing tables along with the movement of mobile nodes. Here, mobile nodes keep their IPv6 addresses

(and therefore their prefixes) unchanged during their movements across IP networks. Once a mobile node enters a new network, the router that manages this network learns the IPv6 prefix of the mobile node and starts announcing itself as the next hop for this prefix. Although such solution is fully distributed, the convergence of the routing protocol could be slow in legacy IPv6 networks (e.g. with BGP). Nevertheless, we are convinced that such category is particularly well suited for LLNs. In LLNs, each node is potentially a router and therefore should already participate in the routing process. This explain why we focused our attention on RPL which is the routing protocol supported by the IETF.

3.2. Problem statement

RPL is designed to mitigate the network dynamics inherent to LLNs: connectivity of nodes can be sporadic (due to link perturbations), nodes can disappear (due to energy outage), etc. RPL mitigates such situations by allowing nodes to change their preferred parent in order to reconnect to the graph. The reconnection occurs when a node receives a DIO and computes a better rank than its current rank. However, DIO are scheduled regarding the trickle algorithm [11]. In a stable network, each DIO may be separated by a very large period of time (up to 2.3 hours calculated using default values from [4]). In the worst scenario, a mobile node can therefore move to a new location and waits for 2.3 hours before detecting this movement, changing its preferred parent and updating the corresponding routing paths. During this period of time, the mobile node keeps trying to send its data to a node no longer reachable, which is likely to increase contention on the medium, energy consumption and packet loss. In addition, the root of the graph is no longer able to reach the mobile node as the advertised downward path lifetime expires, resulting in packet loss. Still, a mobile node can send multicast DIS to solicit fresh DIO. But such transmission will reset the trickle timers on neighboring nodes, increasing the control traffic together with the energy consumption and contention on the medium. In addition, DIS are optional and RPL does not specify how and when a node should send such messages. Also, receiving fresh DIO does not necessarily trigger a parent change, even if the current preferred parent of a mobile node is unreachable. With specific objective functions and metrics (e.g. MinHop), a mobile node can find itself in a situation where all neighboring nodes present a rank greater (worst) than its current rank. In such a situation, a mobile node will not change its preferred parent to a node that makes it moving backwards from the root of the graph. To resolve this problem, nodes should be able to reset their rank whenever their preferred parents become unreachable.

To keep track of the reachability of a neighbor, RPL suggests to use external mechanisms such as Neighbor Unreachability Detection (NUD) [5], Bidirectional Forwarding Detection [6] and hints from lower layers via Layer 2 (L2) triggers [7]. All of those mechanisms can detect when a node becomes unreachable, the preferred parent in particular, enabling the node to start searching for a new parent. The node will first search a suitable candidate in its parent set and if there are no parents available, it performs a local repair: the node removes first all parents from the parent set, then announce its disconnection from the DODAG (by sending a DIO advertising an infinite rank to poison upward routes in its sub-DODAG) and re-connect to the graph upon fresh DIO reception. In our previous work we evaluated how NUD and BFD could help RPL to support mobile nodes [8] (hints for lower layer via layer 2 abstraction triggers only defines abstractions to exchange information between layers 2 and 3, enabling cross-layer optimization). NUD is a key element of the Neighbor Discovery Protocol [5] that allows the maintenance of reachability information about active IPv6 neighbors. Once a node confirms the reachability of a neighbor, this neighbor is considered as reachable for 30*s* (using default value). Then, a new reachability confirmation is postponed until the node wants to send a message to this neighbor. From here, the node still waits for 5*s* (by default) before sending neighbor solicitations to confirm reachability. Neighbor solicitations are sent until reachability is confirmed through the reception of a neighbor advertisement or the maximum allowed solicitations (3 by default) are sent and no response is received. In the latter case, the neighbor is considered as unreachable. Note that the IETF recently proposed 6LoWPAN optimizations to Neighbor Discovery [13] in which NUD is only performed to verify that the default routers are still reachable. The procedure itself is very similar to the legacy one (exchange of neighbor solicitations / advertisements) and uses the same default values for the timers. On the other side, BFD is a simple solution to detect failures in the forwarding plane towards a next hop. Reachability between two nodes is confirmed by periodically exchanging BFD packets between those two nodes. If a node stops receiving BFD packet for a certain period of time (not defined by BFD), it considers the neighbor as unreachable. BFD packets are encapsulated into UDP datagrams and may be asynchronously transmitted between the two neighbors.

We showed in [8] that both NUD and BFD fail to prevent serious disconnection of mobile nodes (up to 40*s*), which significantly increases the packet loss (up to 92% of in certain scenarios). BFD presented the highest

signaling overhead while NUD was the most energy-consuming solution. NUD and BFD was designed to provide unreachability detection in networks with different characteristics than LLNs, which explains those underachievements. For example, BFD is based upon periodic transmissions which have a significant impact on networks with limited throughput. On the other side, NUD was not designed to operate over networks with energy constraints. In the next section, we present alternative solutions for mobility support in RPL.

3.3. Related Work

Managing the parent set of RPL and keeping it up to date is a topic that has drawn much attention, mainly when mobility of nodes is also present. The proposed solution in [14] is applied in a vehicular Ad Hoc network. To mitigate high dynamics due to vehicle movement, the authors propose to eliminate the trickle timer and send DIO at a constant rate, between 2 and 10 seconds. By this means, they manage to decrease the disconnection time, as DIO are received more frequently. Parent change is left up to a better rank received in DIO, relying thus on RPL procedures. Even if they use the expected transmission count metric (ETX), this does not always mean received DIO will actually trigger a parent change when needed. In addition, they introduce a constant control overhead that may significantly reduce the network lifetime.

Another approach to ensure up to date parent set and avoid disconnection of mobile nodes is to periodically send DIS in multicast [15]. Depending on the dynamics experienced by mobile nodes (e.g. the frequency of parent change), the interval between DIS messages is adjusted: if several parents are changed during a defined observation time window (inter-DIS interval), the inter-DIS interval is shortened, while if the mobile node maintains the same preferred parent during this time, the inter-DIS interval is widened. However, the transmission of a multicast DIS will reset the trickle timer of all neighboring nodes, increasing the control traffic and potentially changing the topology, making the established paths unstable. In addition, the parent change is left entirely up to RPL: only a better rank in a received DIO would trigger a parent change.

Co-RPL [16] is an extension of RPL that keeps track of the relative position of mobile nodes. This proposal divides the network into circular areas, known as coronas, centered at the DAG roots. Nodes can belong only to one corona at a time. Each corona is identified by an ID that serves as a relative coordinate to localize mobile nodes. In addition, each node tracks its neighbors by maintaining a neighbor table. This table is filled upon DIO reception. Node mobility is detected by two means: when a mobile node moves to a new corona or when its neighbor table changes, both situations triggering a parent change. Simulation results show that Co-RPL decreases the packet loss ratio by 45% and lower the energy consumption by 50% when compared to standard RPL. This evaluation can be valuable for networks where all nodes are mobile and only the root is fixed. However, many parameters of Co-RPL are not given by the authors, which makes difficult any comparison with other solutions. Also, the benefit of introducing coronas is not clear as the RPL rank already defines the node's position relative to other nodes with respect to a DODAG root.

Authors of [17] develop a mobility mechanism for RPL (mRPL) by integrating a mobility detection mechanism based on received RSSI levels. Once connected to a parent, a mobile node will send several data packets to the parent, after which the parent will send back a unicast DIO. This DIO contains the average RSSI level and implicitly filters asymmetric links. As long as the received RSSI levels are above a threshold, data transmission continues. When the mobile node detects that the RSSI value drops under a threshold, it will start searching for a new parent. For this, the mobile node sends a burst of multicast DIS messages. The receiving nodes will reply with DIO messages in unicast, delaying their reply in such a way that collisions do not occur at the mobile node. This process continues until the mobile node finds a new parent with a high quality link (the received RSSI above a threshold). Simulation and experimentation results show that mRPL improves the mobility management in several areas: high packet delivery ratio, responsiveness to network dynamics, effectiveness at high data transmission rate. However, mRPL needs high data rate to maintain the connectivity of mobile nodes (the packet delivery ratio drops by 24% if the data rate is reduced from 100ms to 5s). Generally, applications in LLN only require a low data rate (e.g. 1 packet/15s for vehicle tracking applications [18]). In addition, the RSSI is known to be unstable and interference sensitive. Operations based on such a versatile parameter are likely to give unreliable results, as shown in [19].

We also envisaged in [20] a solution to detect preferred parent disconnection based entirely on RPL control messages. Our approach advocates that the mobile node connects only as leaf to the DODAG and advertises its mobility status through a flag (Mobility Flag - MF) in the sent DAO. The preferred parent of the mobile node that receives this DAO will pause its trickle timer and switch to what we call a reverse trickle timer. The reverse trickle timer starts with a large interval (I_{max}) that will successively be divided by 2. When each interval expires a DIO is sent. Dividing the intervals happens until I_{min} is reached. Then, the parent requests from any attached mobile nodes a new DAO by

sending a DIO message with an increased Destination Advertisement Trigger Sequence Number (DTSN). If no new DAO with MF set arrives at the parent, the parent will turn back to the regular trickle timer. The mobile node will monitor the interval between received DIO from the parent. When a threshold (given by the number of missed DIO from the preferred parent) is crossed, the mobile node will reset to infinity its rank and start sending DIS in multicast to discover new parents. Simulation results show a decreased disconnection time and control traffic overhead when compared to [14] and [15]. Those results are inline with other surveys on the topic [21].

4. Mobility-Triggered RPL

This section presents our contribution for supporting mobile nodes in RPL, referred to as Mobility-Triggered RPL (MT-RPL). MT-RPL is a cross-layer protocol that follows L2 triggers rule-book enabling communication between RPL at routing layer and X-Machiavel [9], a preamble sampling MAC protocol, at the MAC layer. MT-RPL is based upon the following assumptions: the network is composed of fixed (i.e. non-moving) and mobile nodes, and a node is able to determine in which category (fixed or mobile) it belongs. Note that the MT-RPL may be used with any asynchronous MAC protocol that allows opportunistic forwarding at layer 2. By contrast, synchronous MAC protocols generally offer a poor support for opportunistic forwarding, preventing MT-RPL to operate over this category of protocols.

4.1. X-Machiavel

X-Machiavel is a mobility oriented variation of X-MAC preamble sampling MAC protocol [22]. The idea lying behind X-Machiavel is to favor mobile node transmissions by allowing them to steal the medium from fixed nodes. A transmission in X-MAC starts when the sending node transmits the first strobes of the preamble in the neighborhood. Once the destination receives the preamble it will send an ACK, notifying the sender to stop the strobes and proceed with the data packet. Now the two nodes are synchronized and data can be transferred. After the data is successfully received, the destination sends a new ACK to the sender. X-Machiavel will change X-MAC behavior to give mobile nodes a head start for data packet transmission. When the channel is idle, packets from the mobile node can be opportunistically forwarded by fixed nodes to the destination. When the channel is busy, the mobile node will overhear ongoing transmission of other fixed nodes and will be able to steal the channel in order to send its own data. These operations are possible as X-Machiavel adds two new fields in the packet header.

In the type field, a packet will be identified as being: a preamble frame (type P0, P1 or P2), a data packet (type DATA), an acknowledgment for a preamble (type PK0 or PK1) or an acknowledgment for a data packet (type ACK). Preamble strobes of type P0 are used by the mobile nodes to forbid channel stealing or to allow fixed nodes to opportunistically accept pending data on behalf of the destination. P1 type preamble strobes are sent by fixed nodes to advertise the availability of channel for stealing by mobile nodes. Lastly, preamble strobes of type P2 sent from the fixed nodes grant their data transmission as no node can steal the channel anymore. Acknowledging P0 preambles by fixed nodes is done with PK0 acknowledgment, when the P0 preamble is received but not destined to them. In this situation the fixed node acts as an opportunistic forwarder which is ready to receive data from a mobile node. Preamble received by the intended destination will be acknowledged by PK1 type acknowledgment. More in depth information about how X-Machiavel works is available in [9]. Next, we will present how information from X-Machiavel is used at RPL to create MT-RPL.

4.2. Integration with RPL

The idea of MT-RPL is to use information from layer 2 to trigger actions at layer 3 and vice versa. Furthermore, X-Machiavel events (e.g. opportunistic forwarding or channel stealing) will be reported asynchronously at RPL in order to trigger parent change. In addition, layer 3 information such as the RPL rank is included in the layer 2 header in order to prevent the creation of loops in the network. By this means, MT-RPL makes sure that packets always progress forward in the graph towards the root. With this information, a fixed node only acts as an opportunistic forwarder for packets originated from a mobile node located further away in the DODAG. Mobile nodes on the other hand will also know if stealing the channel from a fixed node is worthwhile. In the following, a detailed presentation of the different operational modes of MT-RPL is done.

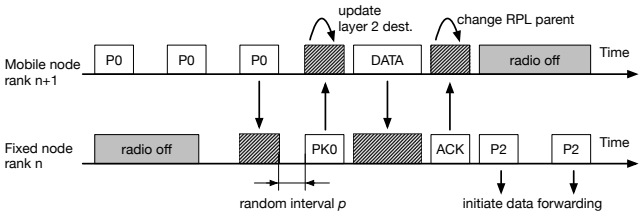

Figure 1. Preamble is acknowledged - Opportunistic forwarding

On an idle channel, a P0 preamble with the rank computed at RPL layer is sent from the mobile node in the neighborhood. If the intended destination is

reachable, X-Machiavel mode of operation is followed: PK1 acknowledgment is sent from the destination, then the data from the mobile node is claimed. On the other hand, if a node different than the intended destination overhears a P0 preamble, it can decide to act as an opportunistic forwarder. The logic behind the choice is based on the RPL rank advertised in the P0 preamble - if the rank of the sender is greater than the rank of the potential forwarder (i.e. the fixed node, which would forward the data, is closer to the root than the mobile node), the potential forwarder can acknowledge the P0 preamble and send back a PK0 acknowledgment (Fig. 1). PK0 acknowledgements are sent after a random interval p in order to always favor the transmission of PK1 acknowledgment from the original destination (i.e. the current preferred parent can always claim first the pending data) and to limit collisions between several opportunistic forwarders. At the mobile node, upon reception of PK0, the mobile node will change the destination towards the new forwarder and send its data (Fig. 1). Upon successful transmission, information about the next hop for the data packet - RPL rank and address - is provided asynchronously from L2 to RPL via L2 triggers. At RPL, if the forwarder is in the mobile node's parent set, the preferred parent is changed accordingly, reflecting the reality from L2. Next, RPL control packets are sent if needed (new DAO and/or DIO). Following a successful opportunistic forwarding, the data from the mobile node will be routed up to the root with P2 preambles in order to ensure no channel stealing. All other nodes with a rank greater or equal to the one announced in the preamble will simply discard the received preamble.

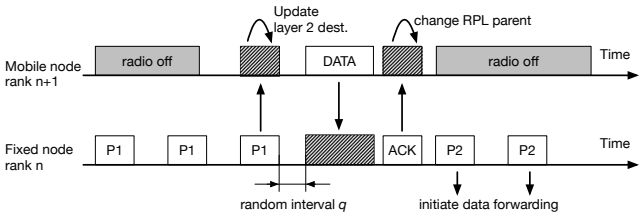

Figure 2. Preamble is overhead – Channel stealing

If the mobile node finds itself in the position to transmit on an occupied channel (Fig. 2), it will fit its transmission between two consecutive strobes of the preamble sent by fixed nodes. X-Machiavel mode of operation states that after receiving a preamble, the destination must also send back an acknowledgment between two strobes to the sender. This will notify the sender to stop sending the preamble and move on to the actual data. MT-RPL allows mobile nodes to send data packets to the sender of the preamble before an acknowledgment from the destination of the preamble can be sent (see Fig. 2). However, mobile nodes with MT-RPL can benefit from this only if the rank of the sender of the preamble is lower than their own rank, in order to achieve forward progress towards the root. If this condition is validated, the mobile node will steal the channel as follows. First, a mobile node overhears a P1 preamble destined to another node and advertising a RPL rank lower than its own RPL rank. After this, the mobile node will update the layer 2 destination of the data packet towards the sender of the overheard preamble and send the packet before the transmission of the next strobe or the PK1 acknowledgment. To prevent collision between multiple mobile nodes, data packets are sent after a short random interval q. Let assume that T is the minimum delay between P1 preamble strobe and its acknowledgment. Mobile nodes randomly draw $q \in [0; T[$ and waits for the expiration of q before transmitting the data. If the channel is busy during this period of time, the mobile node considers that the channel was stolen by another mobile node and postpones its transmission. This procedure allows collision avoidance (to a certain extent) between multiple mobile nodes trying to steal the channel from the same fixed node at the same time.

The fixed node, after receiving a data packet between two P1 preamble strobes, will first acknowledge the successful reception of the data then advertise P2 preamble for both the mobile node data packet and its own data packet, which still needs to be sent. Upon reception of the acknowledgment, the mobile node updates its RPL parent. Further along the path, X-Machiavel principles apply. Any fixed node receiving a P2 preamble will forward the data further using the same preamble, thus data from the mobile node will have priority. Figure 2 shows the transmission of a mobile node on an occupied channel.

Finally, the mobile node can find itself in an area where its preamble is not acknowledged. For example, such situation occurs when the preferred parent is no longer reachable and all nodes in the neighborhood have greater RPL rank than the one currently set on the mobile node. After sending the whole preamble, the mobile node will reset its RPL rank to infinite. At the next scheduled transmission any neighbor will be able to acknowledge the mobile nodes preamble and forward further to the root the data using P2 preambles.

In the former paragraphs we described how MT-RPL leverages X-Machiavel actions at the networking layer in RPL protocol. However, the fixed node acting as an opportunistic forwarder or from which the channel is stolen may not already be in the mobile node parent set. In such situation, the mobile node is missing information to properly re-attach to the DODAG (all the necessary parameters are usually propagated through DIO). Nevertheless, the mobile node can compute the IPv6 address of the fixed node from its MAC address (as 6LoWPAN is regularly used with RPL) to solicit, by

sending a unicast DIS, the transmission of a unicast DIO from this node to receive the missing RPL parameters, such as metric information. By contrast to multicast DIS / DIO, unicast DIS / DIO do not reset trickle timers on neighboring nodes, thus preventing the transmission of a large number control messages in the area.

In conclusion, regardless if the mobile node finds an opportunistic forwarder or steals the medium from another node, MT-RPL accelerates the response to network dynamics by enabling mobile node to maintain DODAG connectivity without generating extra control traffic. The next section will present the performance evaluation of MT-RPL based on real experiments.

5. Experimentation Campaign

5.1. Experimental Setup

To go beyond simulations (which can suffer from various simplifications) or home-made experiments (which are generally non-reproducible), the evaluation of MT-RPL is done through experimentations using FIT IoT-LAB [10]. FIT IoT-LAB is an experimental platform which provides the infrastructure facility suitable for scientific evaluation (experiment automation and reproducibility, precise time sampling, etc.) of IoT communication protocols. More than 2500 wireless nodes are deployed across 6 different sites in France and offer researchers different topological networks. A variety of nodes are available, both in terms of processor architecture (MSP430, ST2M32 - Cortex M3 and Cortex A8) as well as in terms of wireless chips (860 Mhz and 2.4 Ghz 802.15.4 PHY). Our evaluation also includes the other mechanisms suggested by RPL to verify the reachability of nodes (i.e. NUD and BFD) in order to present a thorough comparison with MT-RPL. After implementing all mechanisms presented in both Sect. 3.2 and 4 in Contiki OS, the deployment on the platform was done on Cortex M3 nodes (ST2M32 processor and 2.4 GHz 802.15.4 PHY). The source code of this implementation is available on git [23].

FIT IoT-LAB also provides the infrastructure for mobile node support. Different mobility types are available and trajectories are reproducible. Mobile nodes are Turtlebot2 robots equipped with Cortex M3 node. In our experiments, the movement of the robots can be viewed as a random waypoint model with the following constraints. First, the speed of the robots varies, as they will slow down before an obstacle so that the direction can be adjusted. They also need time to accelerate when departing from a waypoint and decelerate before they arrive to the next waypoint. The waypoints are considered reached when the robot arrives within a predefined range to the exact position of the waypoint. Once reached, the robot will stop and orient itself towards the next waypoint. It is therefore possible that the path between waypoints is not always a straight line, as in the random waypoint model.

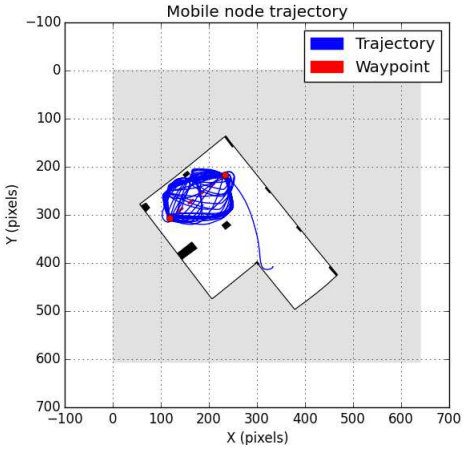

Figure 3. Trajectory of mobile node in Strasbourg

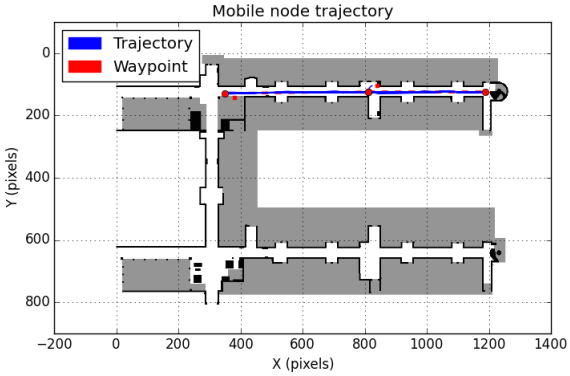

Figure 4. Trajectory of mobile node in Grenoble

Experimentations take place on two different locations (Grenoble and Strasbourg) providing different characteristics. In the Strasbourg site, the fixed nodes are deployed in a form of a 3D grid, with a 2m spacing between each other on all directions (X, Y and Z). Of the two layers available of sensors, we have chosen one layer only, forming a 5x5 node grid, that will act as forwarders between the mobile node and the root. However, as shown in [22], all wireless links in the grid are not necessarily stable due to neighbor interferences and environmental conditions. As a result, the mobile node is not necessarily able to communicate with all fixed nodes at a time. In addition, the quality of the wireless links may vary, resulting in neighbors having different quality. Finally, the mobile node can not choose the root as parent in order to use a minimum of 2-hops paths. The robot roams on the floor of the room, between 2 waypoints, as seen in Fig. 3. In the Grenoble site, the fixed nodes are deployed in corridors, beneath the walkway. The robot moves in the corridor, as illustrated in Fig. 4. We took advantage of the topology and moved

Experiments parameter	Value
Topology - Strasbourg site Grenoble site	1 root, 5 x 5 grid of fixed nodes, 1 mobile node 1 root, 10 fixed nodes beneath the walkway, 1 mobile node
Data collection scheme	Time driven, 1 packet/30s fixed nodes → root 1 packet/5s mobile nodes → root and root → mobile nodes
Data packet size	127 bytes
Mobility model	Modified Random Waypoint, speed up to 0.8 m/s
Routing model	RPL in storing mode using ETX
RPL default values	DIO - given by trickle timer algorithm [11]; min. 4s, max. 8 doublings DIS - 60s or after each data packet if empty parent set, until attached to DODAG DAO - after parent attachment/change or when a DAO from a child arrives
Service interval	3-5 minutes - fixed node can become parent for mobile node 1-4 minutes - fixed node stops receiving data from mobile node
MAC model	X-MAC (NUD and BFD), X-Machiavel (MT-RPL) Maximum number of retransmissions - 4 Duty cycle - 1/64s
Microcontroller unit	ARM Cortex M3, 32-bits, 72 Mhz, 64kB RAM (ST2M32F103REY)
Radio communication	802.15.4 AT86RF231 transceiver 250 kB/s bandwidth, TX power: -17 dBm, Sensitivity -101 dBm
Antenna model	Omnidirectional, modulation O-QPSK
Experimentation setup	10 experiments/mechanism/site, 3 mechanisms, 2 sites, 1 hour/experimentation
Values for parameters of unreachability detection mechanisms	
NUD (RFC 4861)	Maximum number of NS transmission - 3, Delay first probe - 5s, Reachable time - 30s, Retransmission time - 1s
BFD (RFC 5880)	Desired TX interval - 30s, Missed BFD packets that bring session DOWN - 1

Table 1. Experiments parameters

the root outside the reach of the mobile node for the same reason as in Strasbourg.

To collect reliable and precise time measurements during experiments, we force the mobile node to change parents by stopping the service of fixed nodes toward mobile nodes at random time intervals. The fixed nodes will serve the mobile nodes between 3-5 minutes, then they will stop serving the mobile node between 1-4 minutes. With these values of service provision, the mobile node will always find a fixed node that can serve as preferred parent in the network. A more detailed view of the different parameters of experimentation can be explored in Table 1. NUD and BFD are analyzed using X-MAC protocol so that only the receiving node can acknowledge the data packet. Once the preferred parent is considered as unreachable, the parent set is dropped and new DIO are requested from the neighborhood. These DIO are requested by sending multicast DIS messages from the mobile node. On the other hand, MT-RPL uses X-Machiavel protocol, where any node with a better rank than the parent of the node can act as an opportunistic forwarder, or where a mobile node can steal the channel from a fixed node. MT-RPL will only exchange unicast DIS/DIO if needed, as we explained earlier. All mechanisms are used only between the mobile node and its respective parent.

The root to mobile node path is kept up to date with DAO messages. Each change in topology is reported

to the root. Nodes will also update their local routing table, as we operate RPL in storing mode. Mobile nodes will start communicating with the DODAG after 5 minutes from the start of the experiments. This period, considering the size of the topology, ensures a stable DODAG with few changes in the fixed part of the network.

5.2. Results analysis

For each mechanism presented above we made 10 experiments at each site (Strasbourg and Grenoble), leading to 60 experiments of an hour each. With a 95% confidence interval, our measured experimental results are averaged over the 10 experiments for each unreachability detection method and site. During the experiments we evaluated the following parameters: mobile node disconnection time from the preferred parent, packet delivery ratio (PDR) and total number of control messages.

The disconnection time is illustrated in Fig. 5 and represents the time between the preferred parent stopping serving the mobile node, the unreachability detection mechanism reacting and the exchange of RPL control messages (DIS and DIO) with neighboring nodes in order to re-attach to the DODAG (i.e. choose a new preferred parent) plus the time needed by each unreachability detection mechanism to exchange specific control messages until reachability is confirmed. Please note that a mobile node transmits a multicast DIS whenever a data packet should be sent but no next hop is set at the routing layer (i.e. the node does have a preferred parent set).

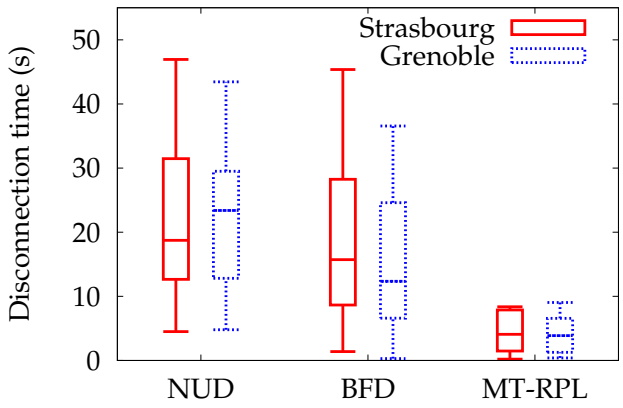

Figure 5. Average disconnection time from parent

As we can see, MT-RPL shows the lower disconnection times in both Strasbourg and Grenoble sites. The large disconnection times observed for NUD (between 31 and 47s for one-quarter of the measurements in Strasbourg) is due to the moment when the mobile node enters in the probe state which depends on the last reachability confirmation and data sending frequency. In the worst case, the mobile node confirmed the reachability of its preferred parent right before being disconnected from this node. With default timer values, the mobile node takes 38*s* to detect the unreachability of its parent. Next, the mobile node should search for a new parent (through the exchange of multicast DIS / DIO) and confirm the reachability of the selected preferred parent (through the exchange of neighbor solicitation and advertisement) to re-attach to the DODAG. In the Grenoble site, the disconnection time experienced with NUD is slightly reduced thanks to the network topology (the nodes experienced a lower medium contention in comparison with Strasbourg).

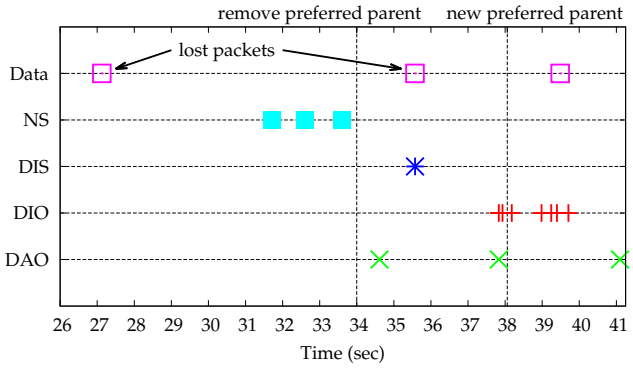

Figure 6. Parent change with NUD

How the mobile node uses NUD to change parent its preferred parent is more clear in Fig. 6. Each dot represents the transmission or reception of a message at the time indicated on the X-axis, while message types are reported on the Y-axis. Results presented in this figure are extracted from one of the most representative trials in Strasbourg. In Fig. 6, reachability is confirmed just before the fixed node, acting as the mobile node preferred parent, stops serving the mobile node (at $t = 0$ sec). After 31*s*, the mobile node moves to the probe state of NUD and sends 3 neighbor solicitations to its current preferred parent. After no response, the preferred parent is considered as unreachable, triggering the reset of the parent set and the transmission of new multicast DIS. Upon reception of new DIO, the mobile node selects a new preferred parent and reconnects to the graph. In this example, the mobile node was disconnected from the graph during 38.05*s*.

With BFD, the disconnection time is slightly lower than the one observed with NUD thanks to the lower timer settings (fixed to 30*s* in our experiments). However, after detecting the unreachability of the preferred parent (the mobile node has not received the expected BFD packet), there can be time variations until a node manages to regain connectivity with the

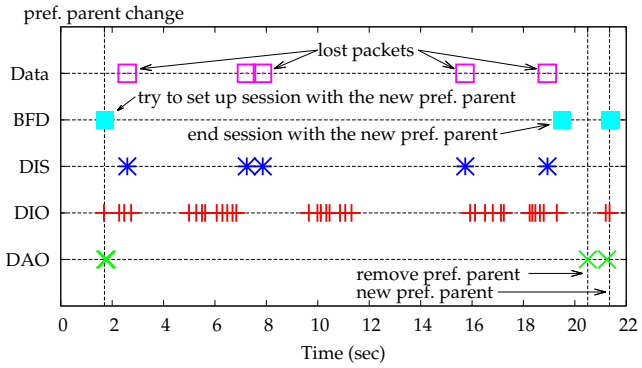

Figure 7. Parent change with BFD

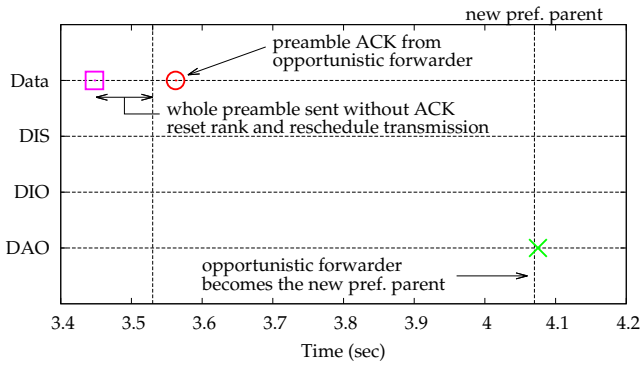

Figure 8. Parent change with MT-RPL

DODAG. The mobile node, as it does not have a valid preferred parent, needs to exchange DIS/DIO to search for a new parent, after which BFD control packets need to be exchanged in order to establish a BFD session. If BFD control packets do not arrive in a timely manner between nodes, delay in reconnection time can spike. We illustrate such situation in Fig. 7 which is constructed similarly to Fig 6. In Fig. 6, 7 and 8, the current preferred parent of the mobile node stops serving the mobile node at $t = 0s$. So the disconnection time starts from $t = 0s$ to the time at which the mobile node re-attaches to a new preferred parent. In Fig. 7 , we can see that even though the detection of disconnection from the preferred parent is done quickly (after $1.7s$), BFD packets are not always exchanged successfully with the new preferred parent to establish a BFD session (e.g. due to the poor quality of the wireless link with the selected parent). As a result, the mobile node starts to search again for a new parent at $t = 20.5s$. Finally, the mobile node manages to set up a BFD session with a new preferred parent after $21.35s$. since the disconnection, re-attaching itself to the DODAG.

With MT-RPL, the disconnection time is in the interval $[0.08s - 5.86s]$, independently of the analyzed topology. As expected, the detection time is close to the data packet transmission rate, because MT-RPL only changes the preferred parent upon channel stealing or opportunistic forwarding. In the worst case (i.e. the upper bound of the disconnection time), the mobile node finds itself surrounded by nodes that can not act as opportunistic forwarders due to their RPL rank. This situation is much clearer in Fig. 8 which shows that the mobile node needs to send a full preamble before resetting its RPL rank and removing its preferred parent at $t = 3.52s$. Next, it reschedules its data transmission. An opportunistic forwarder is now able to claim the data packet. Even in this unfavorable situation, we can see that MT-RPL reduces the disconnection time to $4.07s$ and is not required to send extra control packets (only one DAO to update the downward route). We can also observe a delay of $500ms$ before the mobile node changes the parent due to processing delays specific to the implementation on the Cortex M3 nodes in the FIT IoT-LAB. The mobile node, after it receives the acknowledgement from the opportunistic forwarder will need to send the data packet, so other operations must be postponed (i.e. change of preferred parent). The introduced delay may be reduced or eliminated depending on the hardware capabilities and the operating system that runs on nodes. We chose to postpone the change of parent, as given the periodicity of data exchange (every 5 sec), there will be no negative impact on performance.

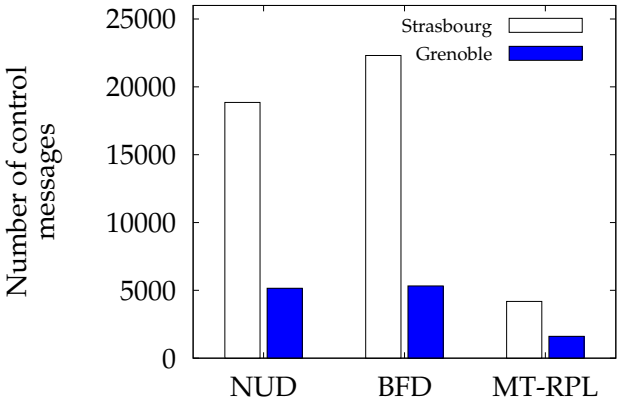

Figure 9. Average number of control messages sent

The overall signaling overhead for each mechanism is presented in Fig. 9. As we can see, NUD and BFD presents the highest signaling overhead for both sites. Enabling an external unreachability detection mechanism introduces not only new control packets, specific for each mechanism, but increases the number of RPL control packets in both experimental sites. With NUD and BFD, once a mobile node is disconnected from the DODAG, it starts sending multicast DIS to solicit

fresh DIO. Any fixed node that receives a multicast DIS resets its trickle timer and sends new DIO at the maximum allowed rate. This process significantly increases the number of transmitted DIO, mainly in Strasbourg where multicast DIS are received by more fixed nodes than in Grenoble. In addition, BFD and NUD need to regularly check the connectivity with the preferred parent through dedicated control messages (BFD packets for BFD and neighbor solicitations / advertisements for NUD). BFD shows the highest signaling overhead because of the number of messages required to set up a BFD session between a mobile node and its preferred parent. By contrast, MT-RPL reduces the number of control messages (by a factor of 4 in comparison with BFD or NUD) together with the disconnection time. First, MT-RPL does not introduce new control messages. Next, in the best cases, no DIS/DIO exchanges are required when the mobile node changes its preferred parent via opportunistic forwarding or channel stealing. In the worst cases, the mobile node should send a unicast DIS to trigger the transmission of a unicast DIO from its new preferred parent. This allows the mobile node to retrieve the missing parameters to properly re-connects to the DODAG. As the mobile node uses unicast transmission, the other neighbors are not involved in the process and therefore do not reset their trickle timer, keeping their current DIO transmission rate low.

The disconnection time of the mobile node from the preferred parent (and thus from the DODAG) is likely to impact the Packet Delivery Ratio (PDR) experienced by the mobile node and the root. At the application layer of the mobile node runs a Constant Bit Rate (CBR) application, which will send packets to the root at regular interval (each 5s). On the root, the same CBR application runs, sending packets to the mobile node. Note that the fixed nodes also run a CBR application to allow channel stealing of MT-RPL. The PDR presented in Table 2 are calculated from the application layer. In Strasbourg, the PDR values for paths between the mobile node and the root are high for NUD and MT-RPL (cf. Table 2). Obviously, MT-RPL shows a higher PDR than NUD thanks to its lowest disconnection time. Also, the grid topology increases the probability that a fixed node is only 1 hop away from the root, reducing the probability to drop packet on the way. On the other hand, BFD experiences an increased packet loss (41% of the packets are lost while both NUD and MT-RPL limit the loss to 12% and 5% respectively). With BFD, the mobile node needs to set up a BFD session with the chosen parent to enforce the parent change. We have previously seen in Fig. 7 that such confirmation may not come in a timely manner, which leads to an increased packet loss. However, the paths from the root to the mobile node experience lower PDR for all solutions. Furthermore, such paths have to be up-to-date in order to route packet at the current location of the mobile node. With NUD and BFD, the mobile node keeps its preferred parent (being reachable or not) for longer periods of time. Advertised downward routes are therefore more stable but do not necessarily reflect the current position of the mobile node. By contrast, MT-RPL changes the preferred parent more often, generating a large amount of DAO transmissions, making downward routes unstable. In addition, local conditions may lead to loss of DAO messages and so, an intermediate router (being the root or not) is likely to find itself with an expired entry (no route to host) or an outdated route that can no longer reach the mobile node. For now, we can say that the paths from the root to the mobile node are still unreliable, regardless of the unreachability mechanism used.

Strasbourg site		NUD	BFD	MT-RPL
Mobile node to root	Avg. (%)	87.90	58.81	94.72
	± (%)	3.38	3.69	3.07
Root to mobile node	Avg. (%)	28.65	26.04	19.28
	± (%)	5.40	1.94	4.59
Grenoble site				
Mobile node to root	Avg. (%)	26.46	10.53	34.15
	± (%)	6.89	1.68	2.36
Root to mobile node	Avg. (%)	9.99	6.48	7.72
	± (%)	5.51	4.52	4.16

Table 2. Packet delivery ratio with 95% confidence intervals

In Grenoble, the different topology and the environmental conditions impact the PDR values. Even if we have comparable disconnection time in Strasbourg, PDR values decrease. Such drops are mainly due to losses on the links between fixed nodes towards the root. In Grenoble, nodes are located into corridors in which people walk by and may interfere with transmissions. In addition, only one path is available to reach the root. As a result, all traffic is carried by the same path, which generates network congestion. Moreover, the mobile node is always between 2 and 4 hops away from the root which further increases the probability of packet loss. Nevertheless, MT-RPL doubles the PDR achieved by BFD and delivers ∼29% more packets than NUD between the mobile node and the root. Such results are explained by two factors. First, MT-RPL limits the signaling overhead, thus reducing the overall contention on the network. In addition, MT-RPL allows the mobile node to send packets opportunistically, so fixed nodes closer to the root may receive these packets, which will reduce the possibility of packet loss. However, PDR values of the path from the root to the mobile node are still low. We plan to investigate solutions to make downward routes more stable and reliable.

6. Conclusions and Perspectives

In this article, we analyzed how node mobility affects key parameters of communication transmitted over a Low Power and Lossy Network (LLN). Our study focus on the IETF IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL). We showed that RPL operations can result in situations in which a mobile node can be disconnected from the network for long periods of time or is even unable to reconnect to the network. To avoid such situations, RPL suggests to use external mechanisms such as Neighbor Unreachability Detection (NUD) or Bidirectional Forwarding Detection (BFD). In a preliminary work, we showed by simulation that those solutions are unable to prevent nodes from being disconnected for long periods of time. In this article, we proposed a new cross-layer protocol operating between the MAC and routing layer known as Mobility-Triggered RPL (MT-RPL). Our solution uses X-Machiavel, a MAC protocol that allows mobile nodes to use opportunistic forwarders or to steal the medium from fixed nodes. In short, MT-RPL enables X-Machiavel operations to be reported to the network layer in order to trigger the necessary RPL operations to remain connected to the graph. Although some aspects of MT-RPL are parts of our previous work, the present article is the first synthesis that put all the pieces of MT-RPL together. This article also represents the first experimental analysis of MT-RPL on a real platform including mobile nodes.

Results presented in 5.2 confirmed that neither NUD nor BFD allow fast reconnection to the network, increasing packet loss together with the signaling overhead. By contrast, MT-RPL reacts quickly to topology changes, reducing the disconnection time together with the packet loss. In addition, MT-RPL reduces 2-5 times the control traffic compared to BFD or NUD. Nevertheless, there are still some areas where further improvements could be made, such as the path between the root and the mobile node, which is still under-performing when it comes to obtained PDR values or availability of the path. We are currently investigating solutions for this problem. We are also extending the FIT IoT-LAB framework to allow precise time measurements of the energy consumption together with increasing the number of supported mobile nodes.

References

[1] J. Leguay et al. (2008) An efficient service oriented architecture for heterogeneous and dynamic wireless sensor networks. In *IEEE International Workshop on Practial Issues in Building Sensor Network Applications*.

[2] J. Allred et al. (2007) Sensorflock: an airborne wireless sensor network of micro-air vehicles. In *5th ACM Conference on Embedded Networked Sensor Systems*.

[3] F. Giust et al. (2015) Distributed Mobility Management for Future 5G Networks: Overview and Analysis of Existing Approaches. *IEEE Communication Magazine* 53(1): 142–149.

[4] T. Winter et al. (2012), RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks, IETF RFC 6550.

[5] T. Narten et al. (2007), Neighbor Discovery for IP version 6 (IPv6), IETF RFC 4861.

[6] D. Katz et al. (2010), Bidirectional Forwarding Detection (BFD), IETF RFC 5880.

[7] F. Teraoka et al. (2008), Unified Layer 2 (L2) Abstractions for Layer 3 (L3)-Driven Fast Handover, IETF RFC 5184.

[8] C. Cobârzan et al. (2015) Integrating Mobility in RPL. In *12th European on Wireless Sensor Networks (EWSN)*.

[9] R. Kuntz et al. (2013) Improving the medium access in highly mobile Wireless Sensor Networks. *Springer Telecommunication Systems* 52(4): 2437–2458.

[10] Future Internet (FIT) - Internet of Things testbed. URL https://www.iot-lab.info. [visited on 2016-06].

[11] P. Levis et al. (2011), The Trickle Algorithm, IETF RFC 6206.

[12] J. Montavont et al. (2014) Mobile IPv6 in Internet of Things: Analysis, Experimentations and Optimizations. *Elsevier Ad Hoc Networks* 14: 15–25.

[13] J. Montavont et al. (2015) Theoretical Analysis of IPv6 Stateless Address Autoconfiguration in Low-power and Lossy Wireless Networks. In *11th IEEE RIVF International Conf. on Computing and Communication Technologies*.

[14] K. C. Lee et al. (2012) RPL Under Mobility. In *IEEE Consumer Communications and Networking Conference*.

[15] I.E. Korbi et al. (2012) Mobility Enhanced RPL for Wireless Sensor Networks. In *IEEE International Conference on the Network of the Future (NOF)*.

[16] O. Gaddour et al. (2014) Co-RPL: RPL routing for mobile low power wireless sensor networks using Corona mechanism. In *9th IEEE International Symposium on Industrial Embedded Systems (SIES)*.

[17] H. Fotouhi et al. (2015) mRPL: Boosting mobility in the Internet of Things. *Elsevier Ad Hoc Networks* 26: 17–35.

[18] J. Krnic et al. (2008) Impact of WSN applications generated traffic on WCDMA access networks. In *IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*.

[19] Y. Chen et al. (2010) On the Mechanisms and Effects of Calibrating RSSI Measurements for 802.15.4 Radios. In *7th European on Wireless Sensor Networks (EWSN)*.

[20] C. Cobârzan et al. (2014) Analysis and performance evaluation of RPL under mobility. In *IEEE Symposium on Computers and Communication (ISCC)*.

[21] A. Oliveira et al. (2016) Low-power and lossy networks under mobility: A survey. *Elsevier Computer Networks* .

[22] J. Beaudaux et al. (2014) Thorough Empirical Analysis of X-MAC over a Large Scale Internet of Things Testbed. *IEEE Sensors Journal, Special Issue on Internet of Things (IoT): Architecture, Protocols and Services* 14: 383–392.

[23] MT-RPL implementation source code. URL https://icube-forge.unistra.fr/montavont/mt-rpl. [visited on 2016-06].