Simplified neighbour cache implementation in RPL/6LoWPAN

Dominique Barthel
France Telecom Orange
Meylan, France
dominique.barthel@orange-ftgroup.com

Abstract

We focus on implementing RPL on resource-constrained nodes, such as 6LoWPAN nodes. In addition to the routing table, RPL requires to maintain a parent set and a candidate (parent) neighbour set. In general, a neighbour cache is also maintained that includes reachability information about all neighbours.

Reducing the control traffic needed to maintain reachability information about all these nodes is called for. Furthermore, memory-constrained nodes placed in a high density environment simply don't have enough memory to store all neighbours anyway[1].

It is advantageous to reduce the neighbour cache mission to only storing information about those nodes that contribute to the routing topology, which is a much simpler subgraph of the physical topology.

Therefore, the mechanisms by which they are maintained can be those provided by the routing protocol.

Context

In this paper, we assume that RPL is used for upward and downward routing, and that downward routing is operated in storing mode. The discussion below directly applies to routers. However, because a dense wireless environment provides many routers to select from, we advocate that even leaf-nodes should be able to understand the metrics used by a RPL instance, and therefore be able to decode Dodag Information Objects (DIOs). Therefore, leaf nodes (which are not hosts in the traditional sense) also benefit from this proposal.

Neighbour cache content

In RPL, for upward routing, a node stores a parent set used for reaching the root. Among this parent set, one is chosen as 'preferred parent'. In addition to the set of parents in use, a few candidate neighbours can be stored for swift replacement of gone-away parents.

Because the storing mode of operation is considered, the node will receive some DAOs originating at nodes in its sub-DODAG. These DAOs can be of two types:

- DAOs originating at direct children having selected the node as their 'preferred parent',
- DAOs propagating information originated at indirect children, i.e. nodes that are part of the sub-DODAGs of direct children.

[1] In commercial Smart Metering deployments, we have witnessed densities of 200-300 in some settings. On the other hand, the available RAM is a few KBs.

When the received DAO comes from a direct child, the originator must be registered into the neighbour cache and flagged as a child. When the received DAO comes from an indirect child, the latter is not registered into the neighbour cache but only into the routing table.

It is proposed that the neighbour cache be limited to parents, parent candidates and children that have sent DAOs. This limits their population to a manageable number. We acknowledge that this simplification restricts the opportunities for P2P routing through direct connection between neighbouring nodes, which is one of the cases considered by RPL. However, in the case of high node density, it is a fact that not all neighbours can be memorized by a memory-limited node anyway.

Neighbour cache management

The neighbour cache now contains two kinds of neighbours

- Children,
- Parents and parent candidates,

which are managed differently using the existing mechanisms in RPL.

Parents are necessarily routers and thus send pseudo-periodic DIOs (only "pseudo" periodic because of the use of Trickle timers). The parent cache management can therefore be based on the detection of DIOs. When a node does not receive a DIO within the expected period (typically a few times the maximum Trickle period, depending on the Trickle transmission suppression factor), it can decide to directly ‘ping’ the parent by using a unicast DIS. If it receives a DIO in answer, the ‘ping’ is successful and the parent is still valid. If the ‘ping’ fails, the parent can be flagged as unreachable in the neighbour cache or even removed (policy dependant). When the parent is removed from the neighbour cache, it must also be removed from the routing table and a no-path DAO must be sent to the root through an alternate route. A new DAO must also be sent to the alternate parent.

Children might be leaf nodes and thus may not transmit pseudo-periodic DIOs. The neighbour cache management cannot be based on the missing DIOs. In order to avoid an energy-costly periodic polling, a reactive approach is proposed. On a failed attempt to forward downward data, the next-hop child will be declared unreachable and the whole route to this child will be cleared by sending a no-path DAO with the target defined as the unreachable child. It is expected that the child will send a DAO by itself when it gets connectivity again.

Conclusions

In low-power and lossy networks where RPL is known to be used as a routing protocol, such as networks of sensors nodes with software in Flash memory, the neighbour cache can be reduced to only storing information about those neighbours relevant to the routing topology, and said cache can be maintained through existing routing control messages. This eases the use of IPv6 protocols on such resource-constrained devices.