Internet Engineering Task Force   M. Goyal, Ed.
Internet-Draft   University of Wisconsin Milwaukee
Intended status: Experimental   E. Baccelli
Expires: August 27, 2011   INRIA
                                                           J. Martocci
                                                      Johnson Controls
                                                    February 23, 2011

                       Identifying Defunct DAGs in RPL
                       draft-goyal-roll-defunct-dags-00

Abstract

   This document specifies a mechanism for an RPL node to identify
   defunct directed acyclic graphs.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

   RPL [I-D.ietf-roll-rpl], an IPv6 routing protocol for low power and
   lossy networks (LLNs), allows the formation of directed acyclic
   graphs (DAGs) in an LLN. These DAGs are used for routing data
   traffic within the LLN as well as to reach destinations outside the
   LLN via the DAG root(s). A DAG can be categorized as "grounded" or
   "floating" based on whether joining the DAG allows a node to meet an
   application specific goal or not. A DAG can be categorized as
   "global" or "local" depending on whether the RPL Instance (identified
   by the RPLInstanceID), to which the DAG belongs, is globally unique
   or not. A DAG may be permanent in nature or exist temporarily
   [I-D.ietf-roll-rpl] [I-D.ietf-roll-p2p-rpl]. A DAG is uniquely
   identified by the combination of its RPLInstanceID, DODAGID and
   DODAGVersionNumber. A node, running RPL, can join at most one DAG
   within an RPL Instance.

   As described in Section 17.4.2 in [I-D.ietf-roll-rpl], an RPL node
   needs to maintain a certain state about each DAG it belongs to. This
   state includes the tuple (RPLInstanceID, DODAGID, DODAGVersionNumber)
   to identify the DAG, the node's current Rank as well as the minimum
   Rank (L) the node has had in this DAG, the set of parents the node
   has in the DAG and the Trickle timers that govern the sending of
   DODAG Information Object (DIO) messages by the node for the DAG
   [I-D.ietf-roll-trickle]. This state, except the Trickle timers,
   needs to be maintained for a certain time duration even when the node
   has no parent left in the DAG. This is done to ensure that the node
   does not join an earlier version of the DAG and it does not rejoin
   the DAG version represented by the DODAGVersionNumber value at a rank
   higher than L + DAGMaxRankIncrease, where DAGMaxRankIncrease is a
   configurable RPL parameter [I-D.ietf-roll-rpl].

   Given the strict memory constraints faced by nodes in an LLN
   [RFC5548] [RFC5673] [RFC5826] [RFC5867], it is imperative that RPL
   protocol has a mechanism that allows a node to identify defunct DAGs
   and delete the state it maintains for such DAGs. This document
   specifies such a mechanism.

1.1. Defunct DAGs

   An RPL node removes a neighbor from its parent set for a DAG:

   o If the neighbor is no longer reachable, as determined using a
      mechanism such as Neighbor Unreachanility Detection (NUD)
      [RFC4861], Bidirectional Forwarding Detection (BFD) [RFC5881] or
      L2 triggers [RFC5184]; or

o If the neighbor advertises in its DIO an infinite rank in the DAG;
   or

o If keeping the neighbor as a parent would required the node to
   increase its rank beyond L + DAGMaxRankIncrease; or

o If the neighbor advertises in its DIO membership in a different
   DAG within the same RPL Instance, where a different DAG is
   recognised by a different DODAGID or a different
   DODAGVersionNumber.

Even if the conditions listed above exist, an RPL node may fail to
remove a neighbor from its parent set because:

o The node fails to receive the neighbor's DIOs advertising an
   increased rank or the neighbor's membership in a different DAG;

o The node may not check, and hence may not detect, the neighbor's
   unreachability for a long time. For example, the node may not
   have any data to send to this neighbor and hence may not encounter
   any event (such as failure to send data to this neighbor) that
   would trigger a check for the neighbor's reachability.

In such cases, a node would continue to consider itself attached to a
DAG even if all its parents in the DAG are unreachable or have moved
to different DAGs. Such a DAG can be characterized as being defunct
from the node's perspective. If the node maintains state about a
large number of defunct DAGs, such state may prevent a considerable
portion of the total memory in the node from being available for more
useful purposes.

To alleviate the problem described above, this document specifies a
mechanism for an RPL node to identify the defunct DAGs and delete the
state it maintains for such DAGs. Note that, given the proactive
nature of RPL protocol, the lack of data traffic using a DAG can not
be considered a reliable indication of the DAG's defunction.
Further, the Trickle timer based control of DIO transmissions means
the possibility of an indefinite delay in the receipt of a new DIO
from a functional DAG parent. Hence, the mechanism specified in this
document is based on the use of a multicast DODAG Information
Solicitation (DIS) message to solicit DIOs about a DAG suspected of
defunction.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and

"OPTIONAL" in this document are to be interpreted as described in
[RFC2119].

Additionally, this document uses terminology from
[I-D.ietf-roll-terminology] and [I-D.ietf-roll-rpl]. Specifically,
the term RPL node refers to an RPL router or an RPL host as defined
in [I-D.ietf-roll-rpl].

3. The No Inconsistency Flag in the DIS Base Object

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Flags    |N|   Reserved   | Option(s)...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
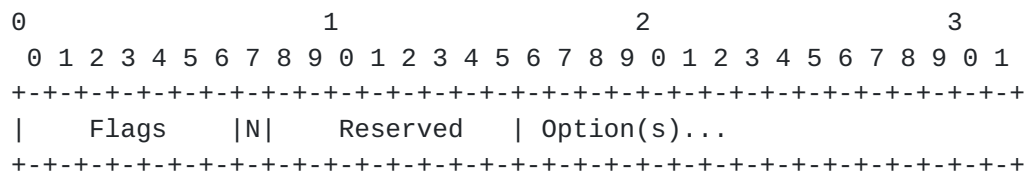
Figure 1: The No Inconsistency Flag in DIS Base Object

An RPL node can use a DODAG Information Solicitation (DIS) message to
solicit DODAG Information Object (DIO) messages from its neighbors.
A DIS may carry a Solicited Information option that specifies the
predicates of the DAG(s) the node is interested in. In the absence
of a Solicited Information option, it is assumed that the node
generating the DIS is interested in receiving DIOs for all the DAGs.
In the following discussion, we use the term "DIS predicates" to
refer to both cases. If the DIS does not contain a Solicited
Information option, all DAGs will match the DIS predicates; otherwise
only those DAGs match the DIS predicates that satisfy the predicates
specified in the Solicited Information option contained in the DIS.

A DIS can be multicast to all the in-range neighbors or it can be
unicast to a specific neighbor. Unless restricted by a DIS flag, an
RPL node must consider the receipt of a multicast DIS as an
inconsistency and hence reset its Trickle timers
[I-D.ietf-roll-trickle] for the DAGs that match the DIS predicates.
The receipt of a unicast DIS causes an RPL node to generate the DIOs
for all the DAGs matching the DIS predicates without resetting the
Trickle timers.

This document defines a "No Inconsistency" (N) flag inside the DIS
base object. The modified DIS base object format is shown in
Figure 1. An RPL node, generating a DIS, MUST set this flag if it
solicits DIOs for the purpose of identifying the defunct DAGs as
specified in this document. On receiving a unicast/multicast DIS
with N flag set, an RPL node MUST NOT reset the trickle timers for
the DAGs that match the DIS predicates. For each DAG matching the

predicates of a multicast DIS received with N flag set, an RPL node
MUST schedule a DIO transmission after a time duration between Imin/2
and Imin, where Imin is the minimum Trickle interval size
[I-D.ietf-roll-trickle] associated with the DAG. For each DAG
matching the predicates of a unicast DIS received with N flag set, an
RPL node MUST immediately generate a DIO.

4. Identifying A Defunct DAG

When an RPL node has not received a DIO from any of its parents in a
DAG for more than MaxSilence*Imax seconds, where MaxSilence is a
configurable parameter greater than 1 and Imax is the maximum Trickle
interval size [I-D.ietf-roll-trickle] associated with the DAG:

o The node MUST generate a multicast DIS message that carries a
   Solicited Information option and has N flag set. The Solicited
   Information option MUST have the I and D flags set and the
   RPLInstanceID/DODAGID fields MUST be set to values identifying the
   DAG. The V flag inside the Solicited Information option SHOULD
   NOT be set so as to allow neighbors to send DIOs advertising the
   latest version of the DAG.

o After sending the DIS, the node MUST wait for Imin duration, where
   Imin is the minimum Trickle interval size associated with the DAG,
   to receive the DIOs generates by its neighbors.

o At the conclusion of the wait period:

   * If the node has received one or more DIOs advertising newer
      version(s) of the DAG, it MUST join the latest version of the
      DAG, select a new parent set among the neighbors advertising
      the latest DAG version and mark the DAG status as functional.

   * Otherwise, if the node has not received a DIO advertising the
      current version of the DAG from a neighbor in the parent set,
      it MUST remove that neighbor from the parent set. As a result,
      if the node has no parent left in the DAG, it MUST mark the DAG
      as defunct and schedule the deletion of the state it has
      maintained for the DAG after DAGHoldTime duration, a
      configurable parameter.

An RPL node SHOULD check the functional status of a DAG it belongs to
in the manner described above at least once during a
CheckDAGStatusTime interval, which is a configurable parameter.

5. Security Considerations

   TBA

6. IANA Considerations

   TBA

7. Acknowledgements

   We gratefully acknowledge Thomas Clausen for motivating this draft.

8. References

8.1. Normative References

   [I-D.ietf-roll-rpl]
             Winter, T., Thubert, P., Brandt, A., Clausen, T., Hui, J.,
             Kelsey, R., Levis, P., Pister, K., Struik, R., and J.
             Vasseur, "RPL: IPv6 Routing Protocol for Low power and
             Lossy Networks", draft-ietf-roll-rpl-18 (work in
             progress), February 2011.

   [I-D.ietf-roll-trickle]
             Levis, P., Clausen, T., Hui, J., Gnawali, O., and J. Ko,
             "The Trickle Algorithm", draft-ietf-roll-trickle-08 (work
             in progress), January 2011.

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
             Requirement Levels", BCP 14, RFC 2119, March 1997.

8.2. Informative References

   [I-D.ietf-roll-p2p-rpl]
             Goyal, M., Baccelli, E., Brandt, A., Cragie, R., Martocci,
             J., and C. Perkins, "Reactive Discovery of Point-to-Point
             Routes in Low Power and Lossy Networks",
             draft-ietf-roll-p2p-rpl-02 (work in progress),
             February 2011.

   [I-D.ietf-roll-terminology]
             Vasseur, J., "Terminology in Low power And Lossy
             Networks", draft-ietf-roll-terminology-04 (work in
             progress), September 2010.

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

   [RFC5184]   Teraoka, F., Gogo, K., Mitsuya, K., Shibui, R., and K.
               Mitani, "Unified Layer 2 (L2) Abstractions for Layer 3
               (L3)-Driven Fast Handover", RFC 5184, May 2008.

   [RFC5548]   Dohler, M., Watteyne, T., Winter, T., and D. Barthel,
               "Routing Requirements for Urban Low-Power and Lossy
               Networks", RFC 5548, May 2009.

   [RFC5673]   Pister, K., Thubert, P., Dwars, S., and T. Phinney,
               "Industrial Routing Requirements in Low-Power and Lossy
               Networks", RFC 5673, October 2009.

   [RFC5826]   Brandt, A., Buron, J., and G. Porcu, "Home Automation
               Routing Requirements in Low-Power and Lossy Networks",
               RFC 5826, April 2010.

   [RFC5867]   Martocci, J., De Mil, P., Riou, N., and W. Vermeylen,
               "Building Automation Routing Requirements in Low-Power and
               Lossy Networks", RFC 5867, June 2010.

   [RFC5881]   Katz, D. and D. Ward, "Bidirectional Forwarding Detection
               (BFD) for IPv4 and IPv6 (Single Hop)", RFC 5881,
               June 2010.

Authors' Addresses

   Mukul Goyal (editor)
   University of Wisconsin Milwaukee
   3200 N Cramer St
   Milwaukee, WI 53201
   USA

   Phone: +1 414 2295001
   Email: mukul@uwm.edu

   Emmanuel Baccelli
   INRIA

   Phone: +33-169-335-511
   Email: Emmanuel.Baccelli@inria.fr
   URI: http://www.emmanuelbaccelli.org/

Jerald Martocci
Johnson Controls
507 E Michigan St
Milwaukee, WI 53202
USA

Phone: +1 414-524-4010
Email: jerald.p.martocci@jci.com