Workgroup: ROLL
Internet-Draft: draft-ietf-roll-rnfd-00
Published: 1 March 2022
Intended Status: Standards Track
Expires: 2 September 2022
Authors: K. Iwanicki
University of Warsaw

RNFD: Fast border router crash detection in RPL

Abstract

By and large, a correct operation of a RPL network requires border routers to be up. In many applications, it is beneficial for the nodes to detect a crash of a border router as soon as possible to trigger fallback actions. This document describes RNFD, an extension to RPL that expedites border router failure detection, even by an order of magnitude, by having nodes collaboratively monitor the status of a given border router. The extension introduces an additional state at each node, a new type of RPL Control Message Options for synchronizing this state among different nodes, and the coordination algorithm itself.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction
   1.1. Effects of LBR Crashes
   1.2. Design Principles
2. Terminology
3. Overview
   3.1. Protocol State Machine
   3.2. Counters and Communication
4. The RNFD Option
   4.1. General CFRC Requirements
   4.2. Format of the Option
5. RPL Router Behavior
   5.1. Joining a DODAG Version and Changing the RNFD Role
   5.2. Detecting and Verifying Problems with the DODAG Root
   5.3. Disseminating Observations and Reaching Agreement
   5.4. DODAG Root's Behavior
   5.5. Activating and Deactivating the Protocol on Demand
   5.6. Processing CFRCs of Incompatible Lengths
   5.7. Summary of RNFD's Interactions with RPL
   5.8. Summary of RNFD's Constants
6. Manageability Considerations
   6.1. Role Assignment and CFRC Size Adjustment
   6.2. Virtual DODAG Roots
7. Security Considerations
8. IANA Considerations
9. Acknowledgements
10. References
   10.1. Normative References
   10.2. Informative References
Author's Address

1. Introduction

RPL is an IPv6 routing protocol for low-power and lossy networks (LLNs) [RFC6550]. Such networks are usually constrained in device energy and channel capacity. They are formed largely of nodes that offer little processing power and memory, and links that are of variable qualities and support low data rates. Therefore, the main challenge that a routing protocol for LLNs has to address is minimizing resource consumption without sacrificing reaction time to network changes.

One of the main design principles adopted in RPL to minimize node resource consumption is delegating much of the responsibility for routing to LLN border routers (LBRs). A network is organized into destination-oriented directed acyclic graphs (DODAGs), each corresponding to an LBR and having all its paths terminate at the LBR. To this end, every node is dynamically assigned a rank representing its distance, measured in some metric, to a given LBR, with the LBR having the minimal rank, which reflects its role as the DODAG root. The ranks allow each non-LBR node to select from among its neighbors (i.e., nodes to which the node has links) those ones that are closer to the LBR than the node itself: the node's parents in the graph. The resulting DODAG paths, consisting of the node-parent links, are utilized for routing packets upward: to the LBR and outside the LLN. They are also used by nodes to periodically report their connectivity upward to the LBR, which allows in turn for directing packets downward, from the LBR to these nodes, for instance, by means of source routing [RFC6554]. All in all, not only do LBRs participate in routing but also drive the process of DODAG construction and maintenance underlying the protocol.

To play this central role, LBRs are expected to be more capable than regular LLN nodes. They are assumed not to be constrained in computing power, memory, and energy, which often entails a more involved hardware-software architecture and tethered power supply. This, however, also makes them more prone to failures, especially since in large deployments it is often difficult to ensure a backup power supply for every LBR.

1.1. Effects of LBR Crashes

When an LBR crashes, the nodes in its DODAG lose the ability to communicate with other Internet hosts. In addition, a significant fraction of DODAG paths interconnecting the nodes become invalid, as they pass through the LBR. The others also degenerate as a result of DODAG repair attempts, which are bound to fail. In effect, routing inside the DODAG also becomes largely impossible. Consequently, it is desirable that an LBR crash be detected by the nodes fast, so that they can leave the broken DODAG and join another one or trigger additional application- or deployment-dependent fallback mechanisms, thereby minimizing the negative impact of the disconnection.

Since all DODAG paths lead to the corresponding LBR, detecting its crash by a node entails dropping all parents and adopting an infinite rank, which reflects the node's inability to reach the LBR. Depending on the deployment settings, the node can then remain in such a state, join a different DODAG, or even become itself the root of a floating DODAG. In any case, however, achieving this state for all nodes is slow, can generate heavy traffic, and is difficult to implement correctly [Iwanicki16] [Paszkowska19] [Ciolkosz19].

To start with, tearing down all DODAG paths requires each of the LBR's neighbors to detect that its link with the LBR is no longer up. Otherwise, any of the neighbors unaware of this fact can keep advertising a finite rank and can thus be other nodes' parent or ancestor in the DODAG: such nodes will incorrectly believe they have a valid path to the LBR. Detecting a crash of a link by a node normally happens when the node has observed sufficiently many forwarding failures over the link. Therefore, considering the low-data-rate applications of LLNs, the period from the crash to the moment of eliminating from the DODAG the last link to the LBR may be long. Subsequently learning by all nodes that none of their links can form any path leading to the LBR also adds latency, partly due to parent changes that the nodes independently perform in attempts to repair their broken paths locally. Since a non-LBR node has only local knowledge of the network, potentially inconsistent with that of other nodes, such parent changes often produce paths containing loops, which have to be broken before all nodes can conclude that no path to the LBR exists globally. Even with RPL's dedicated loop detection mechanisms [RFC6553], this also requires traffic, and hence time. Finally, switching a parent or discovering a loop can also generate cascaded bursts of control traffic, owing to the adaptive Trickle algorithm for exchanging DODAG information [RFC6202]. Overall, the behavior of the network when handling an LBR crash is highly suboptimal, thereby not being in line with RPL's goal of minimizing resource consumption and reaction latencies.

1.2. Design Principles

To address this issue, this document proposes an extension to RPL, dubbed Root Node Failure Detector (RNFD). To minimize the time and traffic required to handle an LBR crash, the RNFD algorithm adopts the following design principles, derived directly from the previous observations:

1. Explicitly coordinating LBR monitoring between nodes instead of relying only on the emergent behavior resulting from their independent operation.

2. Avoiding probing all links to the dead LBR so as to reduce the tail latency when eliminating these links from the DODAG.

3. Exploiting concurrency by prompting proactive checking for a possible LBR crash when some nodes suspect such a failure may have taken place, which aims to further reduce the critical path.

4. Minimizing changes to RPL's existing algorithms by operating in parallel and largely independently (in the background), and introducing few additional assumptions.

While these principles do improve RPL's performance under a wide range of LBR crashes, their probabilistic nature precludes hard guarantees for all possible corner cases. In particular, in some scenarios, RNFD's operation may result in false negatives, but these situations are peculiar and will eventually be handled by RPL's own aforementioned mechanisms. Likewise, in some scenarios, notably involving highly unstable links, false positives may occur, but they can be alleviated as well. In any case, the principles also guarantee that RNFD can be deactivated at any time, if needed, in which case RPL's operation is unaffected.

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The Terminology used in this document is consistent with and incorporates that described in "Terms Used in Routing for Low-Power and Lossy Networks (LLNs)" [RFC7102], "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks" [RFC6550], and "The Routing Protocol for Low-Power and Lossy Networks (RPL) Option for Carrying RPL Information in Data-Plane Datagrams" [RFC6553]. Other terms in use in LLNs can be found in "Terminology for Constrained-Node Networks" [RFC7228].

In particular, the following acronyms appear in the document:

DIO DODAG Information Object (a RPL message)

DIS DODAG Information Solicitation (a RPL message)

DODAG Destination-Oriented Directed Acyclic Graph

LLN Low-power and Lossy Network

LBR LLN Border Router

In addition, the document introduces the following concepts:

Sentinel One of the two roles that a node can play in RNFD. For a given DODAG Version, a Sentinel node is the DODAG root's neighbor that monitors the DODAG root's status. There are normally multiple Sentinels for a DODAG root. However, being the DODAG root's neighbor need not imply being Sentinel.

Acceptor The other of the two roles that a node can play in RNFD. For a given DODAG Version, an Acceptor node is a node that is not Sentinel.

Locally Observed DODAG Root's State (LORS) A node's local knowledge of the DODAG root's status, specifying in particular whether the DODAG root is up.

Conflict-Free Replicated Counter (CFRC) Conceptually represents a dynamic set whose cardinality can be estimated. It defines a partial order on its values and supports element addition and union. The union operation is order- and duplicate-insensitive, that is, idempotent, commutative, and associative.

3. Overview

As mentioned previously, LBRs are DODAG roots in RPL, and hence a crash of an LBR is global in that it affects all nodes in the corresponding DODAG. Therefore, each node running RNFD for a given DODAG explicitly tracks the DODAG root's current condition, which is referred to as Locally Observed DODAG Root's State (LORS), and synchronizes its local knowledge with other nodes.

Since monitoring the condition of the DODAG root is performed by tracking the status of its links (i.e., whether they are up or down), it must be done by the root's neighbors; other nodes must accept their observations. Consequently, depending on their roles, non-root nodes are divided in RNFD into two disjoint groups: Sentinels and Acceptors. A Sentinel node is the DODAG root's neighbor that monitors its link with the root. The DODAG root thus normally has multiple Sentinels but being its neighbor need not imply being Sentinel. An Acceptor node is in turn a node that is not Sentinel. Acceptors thus mainly collect and propagate Sentinels' observations. More information on Sentinel selection can be found in Section 6.1.

3.1. Protocol State Machine

The possible values of LORS and transitions between them are depicted in Figure 1. States "UP" and "GLOBALLY DOWN" can be attained by both Sentinels and Acceptors; states "SUSPECTED DOWN" and "LOCALLY DOWN"--by Sentinels only.

```
  +---------------------------------------------------------+
  |                      |---------------------------+   3a |
  |    +-----------------+---------+              3b |      |
  |    | 2b              |         v                 v      v
+-+----+-+   1 +---------+-+     +-----------+     +-+------+-+
|   UP   +---->+ SUSPECTED +---->+  LOCALLY  +---->+ GLOBALLY |
|        +<----+    DOWN   | 2a  |    DOWN   | 3c  |   DOWN   |
+-+----+-+  4a +-----------+     +-+---------+     +-+--------+
  ^    ^                           |                 |
  |    |                        4b |                 |
  |    +---------------------------+               5 |
  +--------------------------------------------------+
```

Figure 1: RNFD States and Transitions

To begin with, when any node joins a DODAG Version, the DODAG root must appear alive, so the node initializes RNFD with its LORS equal to "UP". For a properly working DODAG root, the node remains in state "UP".

However, when a node--acting as Sentinel--starts suspecting that the root may have crashed, it changes its LORS to "SUSPECTED DOWN" (transition 1 in Figure 1). The transition from "UP" to "SUSPECTED DOWN" can happen based on the node's observations at either the data plane, for instance, link-layer triggers about missing hop-by-hop acknowledgments for packets forwarded over the node's link to the root, or the control plane, for example, a significant growth in the number of Sentinels already suspecting the root to be dead. In state "SUSPECTED DOWN", the Sentinel node may verify its suspicion and/or inform other nodes about the suspicion. When this has been done, it changes its LORS to "LOCALLY DOWN" (transition 2a). In some cases, the verification need not be performed and, as an optimization, a direct transition from "UP" to "LOCALLY DOWN" (transition 2b) can be done instead.

If sufficiently many Sentinels have their LORS equal to "LOCALLY DOWN", all nodes--Sentinels and Acceptors--consent globally that the DODAG root must have crashed and set their LORS to "GLOBALLY DOWN", irrespective of the previous value (transitions 3a, 3b, and 3c). State "GLOBALLY DOWN" is terminal in that the only transition any node can perform from this to another state (transition 5) takes place when the node joins a new DODAG version. When a node is in state "GLOBALLY DOWN", RNFD forces RPL to maintain an infinite rank and no parent, thereby preventing routing packets upward in the DODAG. In other words, this state represents a situation in which all non-root nodes agree that the current DODAG version is unusable, and hence, to recover, the root has to give a proof of being alive by initiating a new DODAG Version.

In contrast, if a node--either Sentinel or Acceptor--is in state "UP", RNFD does not influence RPL's packet forwarding: a node can route packets upward if it has a parent. The same is true for a Sentinel node in states "SUSPECTED DOWN" and "LOCALLY DOWN". Finally, while in any of the two states, a Sentinel node may observe some activity of the DODAG root, and hence decide that its suspicion is a mistake. In such a case, it returns to state "UP" (transitions 4a and 4b).

3.2. Counters and Communication

To enable arriving at a global conclusion that the DODAG root has crashed (i.e., transiting to state "GLOBALLY DOWN"), all nodes count locally and synchronize among each other the number of Sentinels considering the root to be dead (i.e., those in state "LOCALLY DOWN"). This process employs structures referred to as conflict-free replicated counters (CFRCs). They are stored and modified independently by each node and are disseminated throughout the network in options added to RPL link-local control messages: DODAG Information Objects (DIOs) and DODAG Information Solicitations (DISs). Upon reception of such an option from its neighbor, a node merges the received counter with its local one, thereby obtaining a new content for its local counter.

The merging operation is idempotent, commutative, and associative. Moreover, all possible counter values are partially ordered. This enables ensuring eventual consistency of the counters acros all nodes, irrespective of the particular sequence of merges, shape of the DODAG, or general network topology.

Each node in RNFD maintains two CFRCs for a DODAG:

- PositiveCFRC, counting Sentinels that have considered or still consider the root node as alive in the current DODAG Version,

- NegativeCFRC, counting Sentinels that have considered or still consider the root node as dead in the current DODAG Version.

PositiveCFRC is always greater than or equal to the NegativeCFRC in terms of the partial order defined for the counters. The difference between the value of PositiveCFRC and the value of NegativeCFRC is thus nonnegative and estimates the number of Sentinels that still consider the DODAG root node as alive.

4. The RNFD Option

RNFD state synchronization between nodes takes place through the RNFD Option. It is a new type of RPL Control Message Options that is carried in link-local RPL control messages, notably DIOs and DISs.

Its main task is allowing the receivers to merge their two CFRCs with the sender's CFRCs.

4.1. General CFRC Requirements

CFRCs in RNFD MUST support the following operations:

value(c) Returns a non-negative integer value corresponding to the number of nodes counted by a given CFRC, c.

zero() Returns a CFRC that counts no nodes, that is, has its value equal to 0.

self() Returns a CFRC that counts only the node executing the operation.

infinity() Returns a CFRC that counts all possible nodes and represents a special value, infinity.

merge(c1, c2) Returns a CFRC that is a union of c1 and c2 (i.e., counts all nodes that are counted by either c1, c2, or both c1 and c2).

compare(c1, c2) Returns the result of comparing c1 to c2.

saturated(c) Returns TRUE if a given CFRC, c, is saturated (i.e., no more new nodes should be counted by it), or FALSE otherwise.

The partial ordering of CFRCs implies that the result of compare(c1, c2) can be either:

- smaller, if c1 is ordered before c2 (i.e., c2 counts all nodes that c1 counts and at least one node that c1 does not count);

- greater, if c1 is ordered after c2 (i.e., c1 counts all nodes that c2 counts and at least one node that c2 does not count);

- equal, if c1 and c2 are the same (i.e., they count the same nodes);

- incomparable, otherwise.

In particular, zero() is smaller than all other values and infinity() is greater than any other value.

The properties of merging in turn can be formalized as follows for any c1, c2, and c3:

- idempotence: c1 = merge(c1, c1);

- *commutativity: merge(c1, c2) = merge(c2, c1);

- *associativity: merge(c1, merge(c2, c3)) = merge(merge(c1, c2), c3).

In particular, merge(c, zero()) always equals c while merge(c, infinity()) always equals infinity().

There are many algorithmic structures that can provide the aforementioned properties of CFRC. Although in principle RNFD does not rely on any specific one, the option adopts so-called linear counting [Whang90].

4.2. Format of the Option

The format of the RNFD Option conforms to the generic format of RPL Control Message Options:

```
     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
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |   Type = TBD1 | Option Length |                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
    |                                                               |
    +                                                               +
    |               PosCFRC, NegCFRC (Variable Length*)             |
    .                                                               .
    .                                                               .
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The '*' denotes that the fields have equal lengths of at least one octet each. In effect, together they occupy at least two octets.

Figure 2: Format of the RNFD Option

Option Type TBD1

Option Length 8-bit unsigned integer. Denotes the length of the option in octets excluding the Option Type and Option Length fields. Its value MUST be even. A value of 0 denotes that RNFD is disabled in the current DODAG Version.

PosCFRC, NegCFRC Two variable-length, octet-aligned bit arrays carrying the sender's PositiveCFRC and NegativeCFRC, respectively.

The length of the arrays constituting the PosCFRC and NegCFRC fields is the same and is derived from Option Length as follows. The value of Option Length is divided by 2 to obtain the number of octets each of the two arrays occupies. The resulting number of octets is multiplied by 8 which yields an upper bound on the number of bits in each array. As the actual bit length of each of the arrays, the largest prime number less than the upper bound is assumed. For example, if the value of Option Length is 16, then each array occupies 8 octets, and its actual bit length is 61, as this is the largest prime number less than 64.

Furthermore, for any bit equal to 1 in the NegCFRC, the bit with the same index MUST be equal to 1 also in the PosCFRC. Any unused bits (i.e., the bits beyond the actual bit length of each of the arrays) MUST be equal to 0. Finally, if PosCFRC has all its bits equal to 1, then NegCFRC MUST also have all its bits equal to 1.

The CFRC operations are defined for such bit arrays of a given length as follows:

value(c) Returns the smallest integer value not less than -LT*ln(L0/LT), where ln() is the natural logarithm function, L0 is the number of bits equal to 0 in the array corresponding to c and LT is the bit length of the array.

zero() Returns an array with all bits equal to 0.

self() Returns an array with a single bit, selected uniformly at random, equal to 1.

infinity() Returns an array with all bits equal to 1.

merge(c1, c2) Returns a bit array that constitutes a bitwise OR of c1 and c2, that is, a bit in the resulting array is equal to 0 only if the same bit is equal to 0 in both c1 and c2.

compare(c1, c2) Returns:

*equal if each bit of c1 is equal to the corresponding bit of c2;

*less if c1 and c2 are not equal and, for each bit equal to 1 in c1, the corresponding bit in c2 is also equal to 1;

*greater if c1 and c2 are not equal and, for each bit equal to 1 in c2, the corresponding bit in c1 is also equal to 1;

*incomparable, otherwise.

saturated(c) Returns TRUE, if more than 63% of the bits in c are equal to 1, or FALSE, otherwise.

5. RPL Router Behavior

Although RNFD operates largely independently of RPL, it does need interact with RPL and the overall protocol stack. These interactions are described next and can be realized, for instance, by means of event triggers.

5.1. Joining a DODAG Version and Changing the RNFD Role

Whenever RPL running at a node joins a DODAG Version, RNFD--if active--MUST assume for the node the role of Acceptor. Accordingly, it MUST set its LORS to "UP" and its PositiveCFRC and NegativeCFRC to zero().

The role MAY then change between Acceptor and Sentinel at any time. However, while a switch from Sentinel to Acceptor has no preconditions, for a switch from Acceptor to Sentinel to be possible, *all* of the following conditions MUST hold:

1. LORS is "UP";

2. saturated(PositiveCFRC) is FALSE;

3. a neighbor entry for the DODAG root is present in RPL's DODAG parent set;

4. the neighbor is considered reachable via its link-local IPv6 address.

A role change also REQUIRES appropriate updates to LORS and CFRCs, so that the node is properly accounted for. More specifically, when changing its role from Acceptor to Sentinel, the node MUST add itself to its PositiveCFRC as follows. It MUST generate a new CFRC value, selfc = self(), and MUST replace its PositiveCFRC, denoted oldpc, with newpc = merge(oldpc, selfc). In contrast, the effects of a switch from Sentinel to Acceptor vary depending on the node's value of LORS before the switch:

- for "GLOBALLY DOWN", the node MUST NOT modify its LORS, PositiveCFRC, and NegativeCFRC;

- for "LOCALLY DOWN", the node MUST set its LORS to "UP" but MUST NOT modify its PositiveCFRC and NegativeCFRC;

- for "UP" and "SUSPECTED DOWN", the node MUST set its LORS to "UP", MUST NOT modify it PositiveCFRC, but MUST add itself to NegativeCFRC, that is, replace its NegativeCFRC, denoted oldnc, with newnc = merge(oldnc, selfc), where selfc is the counter generated with self() when the node last added itself to its PositiveCFRC.

5.2. Detecting and Verifying Problems with the DODAG Root

Only nodes that are Sentinels take active part in detecting crashes of the DODAG Root; Acceptors just disseminate their observations, reflected in the CFRCs.

The DODAG root monitoring SHOULD be based on both internal inputs, notably the values of CFRCs and LORS, and external inputs, such as triggers from RPL and other protocols. External input monitoring SHOULD be performed preferably in a reactive fashion, also independently of RPL, and at both data plane and control plane. In particular, it is RECOMMENDED that RNFD be directly notified of events relevant to the routing adjacency maintenance mechanisms on which RPL relies, such as Layer 2 triggers [RFC5184] or the Neighbor Unreachability Detection [RFC4861] mechanism. Only events concerning the DODAG root need be monitored to this end. For example, RNFD can conclude that there may be problems with the DODAG root if it observes a lack of multiple consecutive L2 acknowledgments for packets transmitted by the node via the link to the DODAG root. Internally, in turn, it is RECOMMENDED that RNFD take action whenever there is a change to its local CFRCs, so that a node can have a chance to participate in detecting potential problems even when normally it would not exchange packets over the link with the DODAG root during some period. In particular, RNFD SHOULD conclude that there may be problems with the DODAG root, when the fraction value(NegativeCFRC)/value(PositiveCFRC) has grown by at least RNFD_SUSPICION_GROWTH_THRESHOLD since the node last set its LORS to "UP".

Whenever having its LORS set to "UP" RNFD concludes--based on either external or internal inputs--that there may be problems with the link with the DODAG root, it MUST set its LORS to either "SUSPECTED DOWN" or, as an optimization, to "LOCALLY DOWN".

The "SUSPECTED DOWN" value of LORS is temporary: its aim is to give RNFD an additional opportunity to verify whether the link with the DODAG root is indeed down. Depending on the outcome of such verification, RNFD MUST set its LORS to either "UP", if the link has been confirmed not to be down, or "LOCALLY DOWN", otherwise. The verification can be performed, for example, by transmitting RPL DIS or ICMPv6 Echo Request messages to the DODAG root's link-local IPv6 address and expecting replies confirming that the root is up and reachable through the link. Care SHOULD be taken not to overload the DODAG root with traffic due to simultaneous probes, for instance, random backoffs can be employed to this end. It is RECOMMENDED that the "SUSPECTED DOWN" value of LORS is attained and verification takes place if RNFD's conclusion on the state of the DODAG root is based only on indirect observations, for example, the aforementioned growth of the CFRC values. In contrast, for direct observations, such as missing L2 acknowledgments, the verification MAY be skipped, with the node's LORS effectively changing from "UP" directly to "LOCALLY DOWN".

For consistency with RPL, when detecting potential problems with the DODAG root, RNFD also MUST make use of RPL's independent knowledge. More specifically, a node MUST switch its LORS from "UP" or "SUSPECTED DOWN" directly to "LOCALLY DOWN" if a neighbor entry for the DODAG root is removed from RPL's DODAG parent set or the neighbor ceases to be considered reachable via its link-local IPv6 address.

Finally, while having its LORS already equal to "LOCALLY DOWN", a node may make an observation confirming that its link with the DODAG root is actually up. In such a case, it SHOULD set its LORS back to "UP" but MUST NOT do this before the previous conditions 2-4 necessary for a node to change its role from Acceptor to Sentinel all hold.

To appropriately account for the node's observations on the state of the DODAG root, the aforementioned LORS transitions are accompanied by changes to the node's local CFRCs as follows. Changes between "UP" and "SUSPECTED DOWN" do not affect any of the two CFRCs. During a switch from "UP" or "SUSPECTED DOWN" to "LOCALLY DOWN", in turn, the node MUST add itself to its NegativeCFRC, as explained previously. By symmetry, a transition from "LOCALLY DOWN" to "UP" REQUIRES the node to add itself to its PositiveCFRC, again, as explained previously.

5.3. Disseminating Observations and Reaching Agreement

Nodes disseminate their observations by exchanging CFRCs in the RNFD Options embedded in link-local RPL control messages, notably DIOs and DISs. When processing such a received option, a node--acting as Sentinel or Acceptor--MUST update its PositiveCFRC and NegativeCFRC to respectively newpc = merge(oldpc, recvpc) and newnc = merge(oldnc, recvnc), where oldpc and oldnc are the values of the node's PositiveCFRC and NegativeCFRC before the update, while recvpc and recvnc are the received values of option fields PosCFRC and NegCFRC, respectively.

In effect, the node's value of fraction value(NegativeCFRC)/ value(PositiveCFRC) may change. If the fraction reaches at least RNFD_CONSENSUS_THRESHOLD (with value(PositiveCFRC) being greater than zero), then the node consents on the DODAG root being down. Accordingly, it MUST change its LORS to "GLOBALLY DOWN" and set its PositiveCFRC and NegativeCFRC both to infinity().

The "GLOBALLY DOWN" value of LORS is terminal: the node MUST NOT change it and MUST NOT modify its CFRCs until it joins a new DODAG Version. With this value of LORS, RNFD at the node MUST also prevent RPL from having any DODAG parent and advertising any Rank other than INFINITE_RANK.

Since the RNFD Option is embedded, among others, in RPL DIO control messages, updates to a node's CFRCs may affect the sending schedule of these messages, which is driven by the DIO Trickle timer [RFC6206]. It is RECOMMENDED to use for RNFD a dedicated Trickle timer, different from RPL's DIO Trickle timer. In such a setting, whenever RNFD's timer fires and no DIO message containing the RNFD Option has been sent to the link-local all-RPL-nodes multicast IPv6 address since the previous firing, the node sends a DIO message containing the RNFD Option to the address. In contrast, in the absence of a dedicated Trickle timer for RNFD, an implementation SHOULD ensure that the RNFD Option is present in multicast DIO messages sufficiently often to quickly propagate changes to the node's CFRCs. In either case, a node MUST reset its Trickle timer when it changes its LORS to "GLOBALLY DOWN", so that information about the detected crash of the DODAG root is disseminated in the DODAG fast. Likewise, a node SHOULD reset its Trickle timer when any of its local CFRCs changes significantly.

5.4. DODAG Root's Behavior

The DODAG root node MUST assume the role of Acceptor in RNFD and MUST NOT ever switch this role. It MUST also monitor its LORS and local CFRCs, so that it can react to various events.

To start with, the DODAG root MUST generate a new DODAG Version, thereby restarting the protocol, if it changes its LORS to "GLOBALLY DOWN", which may happen when the root has restarted after a crash or the nodes have falsely detected its crash. It MAY also generate a new DODAG Version if fraction value(NegativeCFRC)/ value(PositiveCFRC) approaches RNFD_CONSENSUS_THRESHOLD, so as to avoid potential interruptions to routing.

Furthermore, the DODAG root SHOULD either generate a new DODAG Version or increase the bit length of its CFRCs if saturated(PositiveCFRC) becomes TRUE. This is a self-regulation mechanism that helps adjust the CFRCs to a potentially large number of Sentinels (see Section 6.1).

In general, issuing a new DODAG Version effectively restarts RNFD. The DODAG root MAY thus perform this operation also in other situations.

5.5. Activating and Deactivating the Protocol on Demand

RNFD CAN be activated and deactivated on demand, once per DODAG Version. The particular policies for activating and deactivating the protocol are outside the scope of this document. However, the activation and deactivation SHOULD be done at the DODAG root node; other nodes MUST comply.

More specifically, when a non-root node joins a DODAG Version, RNFD at the node is initially inactive. The node MUST NOT activate the protocol unless it receives for this DODAG Version a valid RNFD Option containing some CFRCs, that is, having its Option Length field positive. In particular, if the option accompanies the message that causes the node to join the DODAG Version, the protocol SHOULD be active from the moment of the joining. RNFD then remains active at the node until it is explicitly deactivated or the node joins a new DODAG Version. An explicit deactivation MUST take place when the node receives an RNFD Option for the DODAG Version with no CFRCs, that is, having its Option Length field equal to zero. When explicitly deactivated, RNFD MUST NOT be reactivated unless the node joins a new DODAG Version. In particular, when the first RNFD Option received by the node has its Option Length field equal to zero, the protocol MUST remain deactivated for the entire time the node belongs to the current DODAG Version.

When RNFD at a node is initially inactive for a DODAG Version, the node MUST NOT attach any RNFD Option to the messages it sends (in particular, because it may not know the desired CFRC length--see Section 5.6). When the protocol has been explicitly deactivated, the node MAY also decide not to attach the option to its outgoing messages. However, it is RECOMMENDED that it sends sufficiently many messages with the option to the link-local all-RPL-nodes multicast IPv6 address to allow its neighbors to learn that RNFD has been deactivated in the current DODAG version. In particular, it MAY reset its Trickle timer to this end but also MAY use some reactive mechanisms, for example, replying with a unicast DIO or DIS containing the RNFD Option with no CFRCs to a message from a neighbor that contains the option with some CFRCs, as such a neighbor appears not to have learned about the deactivation of RNFD.

5.6. Processing CFRCs of Incompatible Lengths

The merge() and compare() operations on CFRCs require both arguments to be compatible, that is, to have the same bit length. However, the processing rules for the RNFD Option (see Section 4.2) do not necessitate this. This fact is made use of not only in the mechanisms for activating and deactivating the protocol (see Section 5.5), but also in mechanisms for dynamic adjustments of CFRCs, which aim to enable deployment-specific policies (see Section 6.1). A node thus MUST be prepared to receive the RNFD Option with fields PosCFRC and NegCFRC of a different bit length than the node's own PositiveCFRC and NegativeCFRC. Assuming that it has RNFD active and that fields PosCFRC and NegCFRC in the option have a positive length, the node MUST react as follows.

If the bit length of fields PosCFRC and NegCFRC is the same as that of the node's local PositiveCFRC and NegativeCFRC, then the node MUST perform the merges, as detailed previously (see Section 5.3).

If the bit length of fields PosCFRC and NegCFRC is smaller than that of the node's local PositiveCFRC and NegativeCFRC, then the node MUST ignore the option and MAY reset its Trickle timer.

If the bit length of fields PosCFRC and NegCFRC is greater than that of the node's local PositiveCFRC and NegativeCFRC, then the node MUST extend the bit length of its local CFRCs to be equal to that in the option and set the CFRCs as follows:

- If the node's LORS is "GLOBALLY DOWN", then both its local CFRCs MUST be set to infinity().

- Otherwise, they both MUST be set to zero(), and the node MUST account for itself in so initialized CFRCs. More specifically, if the node is Sentinel, then it MUST add itself to its PositiveCFRC, as detailed previously. In addition, if its LORS is "LOCALLY DOWN", then it MUST also add itself to its NegativeCFRC, again, as explained previously. Finally, the node MUST perform merges of its local CFRCs and the ones received in the option (see Section 5.3) and MAY reset its Trickle timer.

In contrast, if the node is unable to extend its local CFRCs, for example, because it lacks resources, then it MUST stop participating in RNFD, that is, until it joins a new DODAG Version, it MUST NOT send the RNFD Option and MUST ignore this option in received messages.

5.7. Summary of RNFD's Interactions with RPL

In summary, RNFD interacts with RPL in the following manner:

- While having its LORS equal to "GLOBALLY DOWN", RNFD prevents RPL from routing packets and advertising upward routes in the corresponding DODAG (see Section 5.3).

- In some scenarios, RNFD triggers RPL to issue a new DODAG Version (see Section 5.4).

- Depending on the implementation, RNFD may cause RPL's DIO Trickle timer resets (see Section 5.3, Section 5.5, and Section 5.6).

- RNFD monitors events relevant to routing adjacency maintenance as well as those affecting RPL's DODAG parent set (see Section 5.1 and Section 5.2).

- Using RNFD entails embedding the RNFD Option into link-local RPL control messages (see Section 4.2).

5.8. Summary of RNFD's Constants

The following is a summary of RNFD's constants:

RNFD_SUSPICION_GROWTH_THRESHOLD A threshold concerning the value of fraction value(NegativeCFRC)/value(PositiveCFRC). If the value at a Sentinel node grows at least by this threshold since the time the node's LORS was last set to "UP", then the node's LORS is set to "SUSPECTED DOWN" or "LOCALLY DOWN", which implies that the node suspects or assumes a crash of the DODAG root (see Section 5.2). The default value of the threshold is 0.12. The higher the value the longer the detection period but the lower risk of increased traffic due suspicion verification.

RNFD_CONSENSUS_THRESHOLD A threshold concerning the value of fraction value(NegativeCFRC)/value(PositiveCFRC). If the value at a Sentinel or Acceptor node reaches the threshold, then the node's LORS is set to "GLOBALLY DOWN", which implies that consensus has been reached on the DODAG root node being down (see Section 5.3). The default value of the threshold is 0.51. The higher the value the longer the detection period but the lower the risk of false positives.

The means of configuring the constants at individual nodes are outside the scope of this document.

6. Manageability Considerations

RNFD is largely self-managed, with the exception of protocol activation and deactivation, as well as node role assignment and the related CFRC size adjustment, for which only the aforementioned mechanisms are defined, so as to enable adopting deployment-specific policies. This section outlines some of the possible policies.

6.1. Role Assignment and CFRC Size Adjustment

One approach to node role and CFRC size selection is to manually designate specific nodes as Sentinels in RNFD, assuming that they will have chances to satisfy the necessary conditions for attaining this role (see Section 5.1), and fixing the CFRC bit length to accommodate these nodes.

Another approach is to automate the selection process: in principle, any node satisfying the necessary conditions for becoming Sentinel (see Section 5.1) can attain this role. However, in networks where the DODAG root node has many neighbors, this approach may lead to saturated(PositiveCFRC) quickly becoming TRUE, which--without additional measures--may degrade RNFD's performance. This issue can be handled with a probabilistic solution: if PositiveCFRC becomes saturated with little or no increase in NegativeCFRC, then a new DODAG Version can be issued and a node satisfying the necessary conditions can become Sentinel in this version only with probability 1/2. This process can be continued with the probability being halved in each new DODAG Version until PositiveCFRC is no longer quickly saturated. Another solution is to increase, potentially multiple times the bit length of the CFRCs by the DODAG root if PositiveCFRC becomes saturated with little or no growth in NegativeCFRC, which does not require issuing a new DODAG Version but lengthens the RNFD Option. In this way, again, a sufficient bit length can be dynamically discovered or the root can conclude that a given bit length is excessive for (some) nodes and resort to the previous solution. Increasing the bit length can be done, for instance, by doubling it, respecting the condition that it has to be a prime number (see Section 4.2).

In either of the solutions, Sentinel nodes SHOULD preferably be stable themselves and have stable links to the DODAG root. Otherwise, they may often exhibit LORS transitions between "UP" and "LOCALLY DOWN" or switches between Acceptor and Sentinel roles, which gradually saturates CFRCs. Although as a mitigation the number of such transitions and switches per node MAY be limited, having Sentinels stable SHOULD be preferred.

6.2. Virtual DODAG Roots

RPL allows a DODAG to have a so-called virtual root, that is, a collection of nodes coordinating to act as a single root of the DODAG. The details of the coordination process are left open in the specification [RFC6550] but, from RNFD's perspective, two possible realizations are worth consideration:

- Just a single (primary) node of the nodes comprising the virtual root acts as the actual root of the DODAG. Only when this node fails, does another (backup) node take over. As a result, at any time, at most one of the nodes comprising the virtual root is the actual root.

- More than one of the nodes comprising the virtual root act as actual roots of the DODAG, all advertising the same Rank in the DODAG. When some of the nodes fail, the other nodes may or may not react in any specific way. In other words, at any time, more than one node can be the actual root.

In the first realization, RNFD's operation is largely unaffected. The necessary conditions for a node to become Sentinel (Section 5.1) guarantee that only the current primary root node is monitored by the protocol. This SHOULD be taken into account in the policies for node role assignment, CFRC size selection, and, possibly, the setting of the two thresholds (Section 5.8). Moreover, when a new primary has been elected, to avoid polluting CFRCs with observations on the previous primary, it is RECOMMENDED to issue a new DODAG Version, especially if the new primary has different neighbors compared to the old one.

In the second realization, the fact that the virtual root consists of multiple nodes is transparent to RNFD. Therefore, employing RNFD is such a setting can be beneficial only if the nodes comprising the virtual root may suffer from correlated crashes, for instance, due to global power outages.

7. Security Considerations

RNFD is an extension to RPL and is thus both vulnerable to and benefits from the security issues and solutions described in [RFC6550] and [RFC7416]. Its specification in this document does not introduce new traffic patterns or new messages, for which specific mitigation techniques would be required beyond what can already be adopted for RPL.

In particular, RNFD depends on information exchanged in the RNFD Option. If the contents of this option were compromised, then failure misdetection may occur. One possibility is that the DODAG root may be falsely detected as crashed, which would result in an inability of the nodes to route packets, at least until a new DODAG Version is issued by the root. Another possibility is that a crash of the DODAG root may not be detected by RNFD, in which case RPL would have to rely on its own mechanisms. Moreover, compromising the contents of the RNFD Option may also lead to increased traffic due to DIO Trickle timer resets. Consequently, RNFD deployments are RECOMMENDED to use RPL security mechanisms if there is a risk that control information might be modified or spoofed.

In this context, RNFD's two features are worth highlighting. First, unless a DODAG root's all neighbors are compromised, a false positive can always be detected by the root based on its local CFRCs. If the frequency of such false positives becomes problematic, RNFD can be disabled altogether, for instance, until the problem has been diagnosed. This procedure can be largely automated at LBRs. Second, some types of false negatives can also be detected this way.

Those that pass undetected, in turn, are likely not to have major negative consequences on RPL apart from the lack of improvement to its performance upon a DODAG root's crash, at least if RPL's other components are not attacked as well.

8. IANA Considerations

To represent the RNFD Option, IANA is requested to allocate the value TBD1 from the "RPL Control Message Options" sub-registry of the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry.

9. Acknowledgements

The authors would like to acknowledge Piotr Ciolkosz and Agnieszka Paszkowska. Agnieszka contributed to deeper understanding and formally proving various aspects of RPL's behavior upon an LBR crash. Piotr in turn developed a prototype implementation of RNFD dedicated for RPL to verify earlier performance claims.

TODO More likely to follow.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC6206] Levis, P., Clausen, T., Hui, J., Gnawali, O., and J. Ko, "The Trickle Algorithm", RFC 6206, DOI 10.17487/RFC6206, March 2011, <https://www.rfc-editor.org/info/rfc6206>.

[RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, DOI 10.17487/RFC6550, March 2012, <https://www.rfc-editor.org/info/rfc6550>.

[RFC6553] Hui, J. and JP. Vasseur, "The Routing Protocol for Low-Power and Lossy Networks (RPL) Option for Carrying RPL Information in Data-Plane Datagrams", RFC 6553, DOI

10.17487/RFC6553, March 2012, <https://www.rfc-editor.org/info/rfc6553>.

[RFC6554] Hui, J., Vasseur, JP., Culler, D., and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6554, DOI 10.17487/RFC6554, March 2012, <https://www.rfc-editor.org/info/rfc6554>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

10.2. Informative References

[Ciolkosz19] Ciolkosz, P., "Integration of the RNFD Algorithm for Border Router Failure Detection with the RPL Standard for Routing IPv6 Packets", Master's Thesis, University of Warsaw, 2019.

[Iwanicki16] Iwanicki, K., "RNFD: Routing-layer detection of DODAG (root) node failures in low-power wireless networks", In IPSN 2016: Proceedings of the 15th ACM/IEEE International Conference on Information Processing in Sensor Networks, IEEE, pp. 1--12, DOI 10.1109/IPSN.2016.7460720, 2016, <https://doi.org/10.1109/IPSN.2016.7460720>.

[Paszkowska19] Paszkowska, A. and K. Iwanicki, "Failure Handling in RPL Implementations: An Experimental Qualitative Study", In Mission-Oriented Sensor Networks and Systems: Art and Science (Habib M. Ammari ed.), Springer International Publishing, pp. 49--95, DOI 10.1007/978-3-319-91146-5_3, 2019, <https://doi.org/10.1007/978-3-319-91146-5_3>.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <https://www.rfc-editor.org/info/rfc4861>.

[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, DOI 10.17487/RFC5184, May 2008, <https://www.rfc-editor.org/info/rfc5184>.

[RFC6202] Loreto, S., Saint-Andre, P., Salsano, S., and G. Wilkins, "Known Issues and Best Practices for the Use of Long Polling and Streaming in Bidirectional HTTP", RFC 6202, DOI 10.17487/RFC6202, April 2011, <https://www.rfc-editor.org/info/rfc6202>.

[RFC7102]
Vasseur, JP., "Terms Used in Routing for Low-Power and Lossy Networks", RFC 7102, DOI 10.17487/RFC7102, January 2014, <https://www.rfc-editor.org/info/rfc7102>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <https://www.rfc-editor.org/info/rfc7228>.

[RFC7416] Tsao, T., Alexander, R., Dohler, M., Daza, V., Lozano, A., and M. Richardson, Ed., "A Security Threat Analysis for the Routing Protocol for Low-Power and Lossy Networks (RPLs)", RFC 7416, DOI 10.17487/RFC7416, January 2015, <https://www.rfc-editor.org/info/rfc7416>.

[Whang90] Whang, K.-Y., Vander-Zanden, B.T., and H.M. Taylor, "A Linear-time Probabilistic Counting Algorithm for Database Applications", In ACM Transactions on Database Systems, DOI 10.1145/78922.78925, 1990, <https://doi.org/10.1145/78922.78925>.

Author's Address

Konrad Iwanicki
University of Warsaw
Banacha 2
02-097 Warszawa
Poland

Phone: +48 22 55 44 428
Email: iwanicki@mimuw.edu.pl