Stream:	Internet Engineering Task Force (IETF)
RFC:	9914
Updates:	6550, 6553, 8138
Category:	Standards Track
Published:	April 2026
ISSN:	2070-1721
Authors:	P. Thubert, Ed.	R.A. Jadhav	M. Richardson
---	---	---	---
	Independent	*AccuKnox*	*Sandelman*

RFC 9914
Root-Initiated Routing State in the Routing Protocol for Low-Power and Lossy Networks (RPL)

Abstract

The Routing Protocol for Low-Power and Lossy Networks (RPL) (RFC 6550) enables data packet routing along a Destination-Oriented Directed Acyclic Graph (DODAG). However, the default route establishment mechanism relies on hop-by-hop forwarding along the DODAG, which may not always provide optimal routing efficiency. This document introduces the concept of Destination Advertisement Object (DAO) Projection, a mechanism that allows a RPL Root or an external controller to install optimized routes within the RPL domain. DAO Projections enable the creation of optimized unicast or multicast routes that do not strictly follow the DODAG structure, thereby improving routing efficiency, reliability, availability, and resource utilization in the RPL domain. This document specifies two types of Projected Routes (P-Routes) -- Storing Mode and Non-Storing Mode -- and outlines the signaling procedures necessary to establish, maintain, and remove these routes. This document updates RFCs 6550, 6553, and 8138.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9914.

Copyright Notice

Table of Contents

1. Introduction 5
2. Terminology 6
2.1. Requirements Language 6
2.2. Terms and Concepts 6
2.3. Glossary 7
2.4. Domain Terms 8
2.4.1. Projected Route 8
2.4.2. Projected DAO 8
2.4.3. Path 8
2.4.4. Routing Stretch 9
2.4.5. Track 9
3. Context and Goal 12
3.1. RPL Applicability 12
3.2. Multi-Topology Routing and Loop Avoidance 13
3.3. Requirements 15
3.3.1. Loose Source Routing 15
3.3.2. Forward Routes 16
3.4. On Tracks 17
3.4.1. Building Tracks with RPL 17
3.4.2. Tracks and RPL Instances 18
3.5. Path Signaling 19
3.5.1. Using Storing Mode Segments 20
3.5.2. Using Non-Storing Mode Joining Tracks 25
3.6. Complex Tracks 31

3.7. Scope and Expectations 33
3.7.1. External Dependencies 33
3.7.2. Relationship to Other IETF Specifications 33
4. Extending and Amending Existing RFCs 34
4.1. Extending RFC 6550 35
4.1.1. Projected DAO 35
4.1.2. Projected DAO Acknowledgment 37
4.1.3. Via Information Option 37
4.1.4. Sibling Information Option 38
4.1.5. P-DAO Request 38
4.1.6. Amending the RPI 39
4.1.7. Additional Flag in the RPL DODAG Configuration Option 39
4.2. Extending RFC 6553 40
4.3. Extending RFC 8138 41
5. New RPL Control Messages and Options 42
5.1. New P-DAO Request Control Message 42
5.2. New PDR-ACK Control Message 43
5.3. Via Information Options 44
5.4. Sibling Information Option 46
6. Root-Initiated Routing State 48
6.1. RPL Network Setup 48
6.2. Requesting a Track 49
6.3. Identifying a Track 49
6.4. Installing a Track 50
6.4.1. Signaling a Projected Route 51
6.4.2. Installing a Track Segment with a Storing Mode P-Route 52
6.4.3. Installing a Protection Path with a Non-Storing Mode P-Route 53
6.5. Tearing Down a P-Route 55
6.6. Maintaining a Track 55
6.6.1. Maintaining a Track Segment 55

6.6.2. Maintaining a Protection Path 56
6.7. Encapsulating and Forwarding Along a Track 56
6.8. Compression of RPL Artifacts 59
7. Less-Constrained Variations 60
7.1. Storing Mode Main DODAG 60
7.2. A Track as a Full DODAG 62
8. Profiles 63
9. Backwards Compatibility 64
10. Security Considerations 65
11. IANA Considerations 65
11.1. RPL DODAG Configuration Option Flag 65
11.2. Elective 6LoWPAN Routing Header Type 66
11.3. Critical 6LoWPAN Routing Header Type 66
11.4. Registry for RPL Option Flags 66
11.5. RPL Control Codes 67
11.6. RPL Control Message Options 67
11.7. Registry for Projected DAO Request Flags 67
11.8. Registry for Projected DAO Request Acknowledgment (PDR-ACK) Flags 68
11.9. Registry for PDR-ACK Acceptance Status Values 68
11.10. Registry for PDR-ACK Rejection Status Values 69
11.11. Registry for Via Information Options Flags 69
11.12. Registry for Sibling Information Option Flags 69
11.13. Destination Advertisement Object Flag 70
11.14. Destination Advertisement Object Acknowledgment Flag 70
11.15. ICMPv6 Error Code 70
11.16. RPL Rejection Status Values 71
12. References 71
12.1. Normative References 71
12.2. Informative References 72
Acknowledgments 74

Authors' Addresses 75

1. Introduction

The Routing Protocol for Low-Power and Lossy Networks (RPL) [RPL], is a Distance Vector protocol that is well-suited for application in a variety of low-energy Internet of Things (IoT) networks where constrained nodes cannot maintain the full view of the topology and stretched paths are acceptable (as opposed to the signaling and state overhead involved in maintaining the shortest paths across). Additionally, RPL is anisotropic, meaning that its operation depends on the orientation of the links, down from or up towards the Root, with the default route advertised down and more-specific paths advertised up along a limited set of links.

RPL forms Destination-Oriented Directed Acyclic Graphs (DODAGs) in which the Root often acts as the border router to connect the RPL domain to the IP backbone. Routers inside the DODAG route along the graph up towards the Root for the default route and down towards destinations in the RPL domain for more-specific routes. As a prerequisite, this specification expects a pre-existing RPL Instance with an associated DODAG and RPL Root, which are referred to as the main Instance, main DODAG, and main Root, respectively. The main Instance is operated in RPL Non-Storing Mode of Operation (MOP).

With this specification, an abstract routing function called a Path Computation Element (PCE) (e.g., located in a central controller or collocated with the main Root) interacts with the main Root to compute additional paths between nodes in the main Instance. In Non-Storing Mode, the base topological information to be passed to the PCE, i.e., the knowledge of the main DODAG, is already available at the Root. This specification introduces protocol extensions that enrich the topological information available to the Root with sibling relationships that are usable but not leveraged to form the main DODAG.

Based on usage, path length, and knowledge of available resources such as battery levels and reservable buffers in the nodes, the PCE, which has a global visibility of the system, can optimize the computed routes for application needs, including the capability to provide path redundancy. This specification also introduces protocol extensions that enable the Root to project (i.e., advertise from a remote location) the computed paths in the RPL domain as Projected Routes (a.k.a. P-Routes) on behalf of the PCE.

A P-Route may be installed in either Storing or Non-Storing Mode, potentially resulting in hybrid situations where the Mode in which the P-Route operates is different from that of the RPL main Instance. P-Routes can be used as stand-alone segments meant to reduce the size of the Source Routing Headers (SRHs), leveraging loose source routing operations down the main RPL DODAG. A P-Route can also be used as a protection path, and it can be combined and interleaved with other P-Routes to form a recovery graph called a Track. A Track is signaled as a separate RPL Instance that is associated with a main RPL Instance such that the RPL routers that form the Track are also members of the main DODAG. The Track provides underlay shortcuts using its own RIB, which is separate from the RIB of the main Instance and has a higher precedence.

2. Terminology

2.1. Requirements Language

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.

In addition, the terms "Extends" and "Amends" are used as per [NEW-TAGS], Section 3.

2.2. Terms and Concepts

In this document, readers will encounter terms and concepts that are discussed in the following:

- "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks" [RPL]
- "An Architecture for IPv6 over the Time-Slotted Channel Hopping Mode of IEEE 802.15.4 (6TiSCH)" [RFC9030]
- "Deterministic Networking Architecture" [RFC8655]
- "Using RPI Option Type, Routing Header for Source Routes, and IPv6-in-IPv6 Encapsulation in the RPL Data Plane" [RFC9008]
- "Reliable and Available Wireless (RAW) Architecture" [RAW-ARCH]
- "Terms Used in Routing for Low-Power and Lossy Networks" [RFC7102]

The 6TiSCH, Deterministic Networking (DetNet), and RAW architectures utilize the terms "Track" and "recovery graph" to represent the same concept even though they are in different environments. This document uses "Track" to represent that concept and only builds Tracks that are DODAGs, meaning that all links are oriented from ingress to egress. This specification also utilizes the terms "segment" and "protection path", which are also defined in the RAW architecture.

As opposed to routing trees, RPL DODAGs are typically constructed to provide redundancy and dynamically adapt the forwarding operation to the state of the Low-Power and Lossy Network (LLN) links. Note that the plain forwarding operation over DODAGs does not provide redundancy for all nodes, since at least the node nearest to the Root does not have an alternate feasible successor.

RAW solves that problem by defining protection paths that can be interleaved to form new paths that can be activated dynamically upon failures. This requires additional control in order to make the routing decision early enough along the Track to route around the failure.

RAW only uses single-ended DODAGs, meaning that they can be reversed in another DODAG by reversing all the links. The ingress of the Track is the Root of the DODAG, whereas the egress is the Root of the reversed DODAG. From the RAW perspective, single-ended DODAGs are special Tracks that only have forward links, and that can be leveraged to provide protection services by defining destination-oriented protection paths within the DODAG.

2.3. Glossary

This document often uses the following abbreviations:

6LoRH:	6LoWPAN Routing Header
6LR:	6LoWPAN Router (e.g., a RPL router in an LLN)
ARQ:	Automatic Repeat Request (in other words, retries)
CMO:	Control Message Option
DAG:	Directed Acyclic Graph
DAO:	Destination Advertisement Object
DIO:	DODAG Information Object
DODAG:	Destination-Oriented Directed Acyclic Graph. A DAG with only one vertex (i.e., node) that has no outgoing edge (i.e., link).
FEC:	Forward Error Correction
GUA:	Global Unicast Address
HARQ:	Hybrid Automatic Repeat Request (combines FEC and ARQ)
LLN:	Low-Power and Lossy Network
MOP:	Mode of Operation
NSM-VIO:	Non-Storing Mode Via Information Option. Source-routed VIO used in Non-Storing Mode P-DAO messages.
P-DAO:	Projected DAO
P-DAO-ACK:	Projected DAO Acknowledgment
P-DAO-REQ:	Projected DAO Request
P-Route:	Projected Route
PCE:	Path Computation Element
PDR-ACK	Projected DAO Request Acknowledgment
PLR:	Point of Local Repair

RAL: RPL-Aware Leaf

RAN: RPL-Aware Node (either a RPL router or a RPL-Aware Leaf)

RH: Routing Header

RIB: Routing Information Base (i.e., the routing table)

RPI: RPL Packet Information

RPL: Routing Protocol for Low-Power and Lossy Networks

RTO: RPL Target Option

RUL: RPL-Unaware Leaf

SIO: Sibling Information Option

SLO: Service Level Objective

SM-VIO: Storing Mode Via Information Option. Strict VIO used in Storing Mode P-DAO messages.

SRH: Source Routing Header (i.e., IPv6 RH type 3); see Section 2.4.5.7.2.

SRH-6LoRH: Source Routing Header 6LoRH. A compressed form of SRH defined in "IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Routing Header" [RFC8138].

TIO: Transit Information Option

ULA: Unique Local Address

VIO: Via Information Option. It can be an SM-VIO or NSM-VIO.

2.4. Domain Terms

This specification uses the terminology defined in the sections that follow.

2.4.1. Projected Route

A RPL P-Route is a RPL route that is computed remotely by a PCE and installed and maintained by a RPL Root on behalf of the PCE. It is installed as a state that signals that destinations (i.e., Targets) are reachable via or along a sequence of nodes.

2.4.2. Projected DAO

A Projected DAO (P-DAO) is a DAO message that is used to install a P-Route.

2.4.3. Path

Quoting (non-normatively) the definition of path in Section 1.3.3 of [INT-ARCH]:

> At a given moment, all the IP datagrams from a particular source host to a particular destination host will typically traverse the same sequence of gateways. We use the term "path" for this sequence. Note that a path is uni-directional; it is not unusual to have different paths in the two directions between a given host pair.

See Section 3.1.1 of [RAW-ARCH] for a longer, more modern definition of path.

It follows that the general acceptance of a path is a linear sequence of nodes, as opposed to a multi-dimensional graph. In the context of this document, a path is observed by following one copy of a packet that is injected in a Track and possibly replicated within.

2.4.4. Routing Stretch

RPL is anisotropic, meaning that it is directional or, more precisely, polar. RPL does not behave the same way "downwards" (Root towards leaves) with *multicast* DODAG Information Object (DIO) messages that form the DODAG versus "upwards" (leaves towards Root) with *unicast* DAO messages that follow the DODAG. This is in contrast with traditional IGPs that operate the same way in all directions and are thus called isotropic.

The term "routing stretch" denotes the length of a path, in comparison to the length of the shortest path, which can be an abstract concept in RPL when the metrics are statistical and dynamic, and the concept of distance varies with the Objective Function.

The RPL DODAG optimizes Point-to-Multipoint (P2MP) paths (from the Root) and Multipoint-to-Point (MP2P) paths (towards the Root), but the Point-to-Point (P2P) traffic has to follow the same DODAG. Following the DODAG, the RPL datapath passes via a common parent in Storing Mode and via the Root in Non-Storing Mode. This typically involves more hops and more latency than the minimum possible for a directional (i.e., forward) P2P path that an isotropic protocol would compute. We refer to this elongated path as stretched.

2.4.5. Track

The concept of Track is inherited from the 6TiSCH architecture [RFC9030] and equals that of a recovery graph in the RAW architecture [RAW-ARCH]. A Track is a networking graph that can be followed to transport packets with equivalent treatment; as opposed to other definitions of a path (see Section 1.3.3 of [INT-ARCH] and Section 3.1.1 of [RAW-ARCH], a Track is not necessarily linear. It may contain multiple paths that may fork and rejoin and that may enable RAW Packet Replication, Elimination, and Ordering Functions (PREOF).

Figure 1 illustrates the mapping of the DODAG with the generic concept of a Track, with the DODAG Root acting as the ingress for the Track, and the mapping of protection paths and segments, i.e., only forward segments, meaning that they are directional and progressing towards the destination. Note that East is represented on the left since the packets are forwarded East-West.

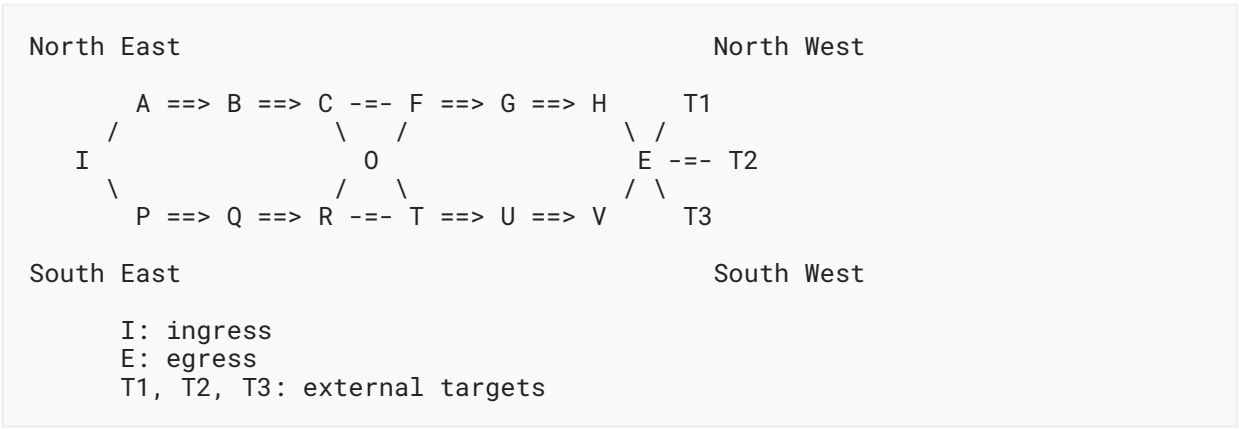

Figure 1: A Track and Its Components

Of note:

I ==> A ==> B ==> C: A segment to targets F and O

I --> F --> E: A protection path to targets T1, T2, T3

I, A, B, C, F, G, H, E: A path to T1, T2, T3

This specification builds Tracks that are DODAGs oriented towards a Track ingress, and the forward direction for packets is from the Track ingress to one of the possible multiple Track egress nodes, which is also down the DODAG.

The Track may be strictly connected, meaning that the vertices are adjacent, or loosely connected, meaning that the vertices are connected using segments that are associated to the same Track.

2.4.5.1. TrackID

A RPLInstanceID (typically of a Local Instance) identifies a Track using the namespace owned by the Track ingress. For Local Instances, the TrackID is associated with the IPv6 address of the Track ingress that is used as the DODAGID, and together they form a unique identification of the Track (see the definition of DODAGID in Section 2 of [RPL]).

2.4.5.2. Namespace

The term "namespace" is used to refer to the scope of the TrackID. The TrackID is locally significant within its namespace. For Local Instances, the namespace is identified by the DODAGID for the Track, and the tuple (DODAGID, TrackID) is globally unique. For Global Instances, the namespace is the whole RPL domain.

2.4.5.3. Complex Track

A Complex Track is a Track that can be traversed via more than one path (e.g., a DODAG).

2.4.5.4. Stand Alone

Stand alone refers to a segment or a protection path that is installed with a single P-DAO that fully defines the path, e.g., a stand-alone segment is installed with a single Storing Mode Via Information Option (SM-VIO) all the way between the ingress and egress.

2.4.5.5. Stitching

This specification uses the term "stitching" to indicate that a Track is piped to another one, meaning that traffic out of the first Track is injected into the other Track.

2.4.5.6. Protection Path

The concept of protection path is defined in the RAW architecture [RAW-ARCH] as an end-to-end forward serial path. With this specification, a protection path is installed by the Root of the main DODAG using a Non-Storing Mode P-DAO message, e.g., I --> F --> E in Figure 1.

As the Non-Storing Mode Via Information Option (NSM-VIO) can only signal sequences of nodes, it takes one Non-Storing Mode P-DAO message per protection path to signal the structure of a Complex Track.

Each NSM-VIO for the same TrackID but with a different SegmentID signals a different protection path that the Track ingress adds to the topology.

2.4.5.7. Segment

A segment is a serial path formed by a strict sequence of nodes along which a P-Route is installed, e.g., I ==> A ==> B ==> C in Figure 1. With this specification, a segment is typically installed by the Root of the main DODAG using Storing Mode P-DAO messages. A segment is used as the topological edge of a Track joining the loose steps along the protection paths that form the structure of a Complex Track. The same segment may be leveraged by more than one protection path where the protection paths overlap.

Since this specification builds only DODAGs, all segments are oriented from the ingress (East) to egress (West), as opposed to the general Track model in the RAW architecture [RAW-ARCH], which allows North/South segments that can be bidirectional as well.

2.4.5.7.1. Section of a Segment

The section of a segment refers to a continuous subset of a segment that may be replaced while the segment remains. For instance, in segment A=>B=>C=>D=>E=>F, say that the link C to D might be misbehaving. The section B=>C=>D=>E in the segment may be replaced by B=>C'=>D'=>E to route around the problem. The segment becomes A=>B=>C'=>D'=>E=>F.

2.4.5.7.2. Segment Routing and SRH

In a Non-Storing Mode RPL domain, the IPv6 Routing Header used for source routing is the RPL Source Route Header as defined in [RFC6554]. This specification operates in that context and uses the acronym SRH to mean IPv6 RH type 3, as opposed to IPv6 RH type 4 defined in [RFC8754] for Segment Routing over IPv6 (SRv6) operation.

If the network is a 6LoWPAN network, the expectation is that the SRH is compressed and encoded as a 6LoWPAN Routing Header (6LoRH), as specified in Section 5 of [RFC8138].

This specification uses the term "Segment Routing" generically to refer to using source routing to hop over segments. As such, forwarding along segments as specified hereafter can be seen as a form of Segment Routing [RFC8402] that leverages the RPL Source Route Header for its operation.

Outside of LLNs, the RPL network may be less constrained and operated in Storing Mode, as discussed in Section 7.1. In that case, this specification could be extended to accommodate the SRv6 RH.

3. Context and Goal

3.1. RPL Applicability

RPL is optimized for situations where the power is scarce, the bandwidth is constrained, and the transmissions are unreliable. This matches the use case of an IoT LLN where RPL is typically used today and also situations of high relative mobility between the nodes in the network (a.k.a. swarming), e.g., within a variable set of vehicles with a similar global motion or a platoon of drones. In contrast, this specification only applies when the platoon has a relatively stable topology where the segments can be attributed reliability and availability for a certain lifetime; see [RAW-ARCH].

To reach this goal, RPL is primarily designed to minimize the control plane activity (i.e., the relative amount of routing protocol exchanges versus data traffic) and the amount of state that is maintained in each node. RPL does not need to converge, and it provides connectivity to most nodes most of the time.

RPL may form multiple topologies called Instances. Instances can be created to enforce various optimizations through Objective Functions or to reach out through different Root nodes. The concept of Objective Function allows adapting the activity of the routing protocol to the use case, e.g., type, speed, and quality of the LLN links.

RPL Instances operate in parallel, unaware of one another. Yet, it is possible to define a model whereby if a route cannot be found in the current Instance A where a packet is being forwarded, then the router may look up the routing table (i.e., the RIB) in Instance B and forward along Instance B if the route is found there. To avoid loops, this must happen in such a way that the Instances themselves form a Directed Acyclic Graph (DAG) leading to the last resort Instance, which is the "lowest" Instance if Instance A is considered "higher" then Instance B. This specification uses underlay Tracks as "lower" Instances, with the main Instance being the "highest" of all.

The RPL Root is responsible for selecting the RPL Instance that is used to forward a packet coming from the backbone into the RPL domain and for setting the related RPL information in the packets. Each Instance creates its own routing table (i.e., a RIB) in participating nodes, and

the RIB associated to the Instance must be used end to end in the RPL domain. To that effect, RPL tags the packets with the Instance ID in a Hop-by-Hop extension header. 6TiSCH leverages RPL for its distributed routing operations.

To reduce the routing exchanges, RPL leverages an anisotropic Distance Vector approach, which does not need global knowledge of the topology and only optimizes the routes to and from the RPL Root, allowing P2P paths to be stretched. Although RPL installs its routes proactively, it only maintains them lazily, in reaction to actual traffic or as a slow background activity.

This is simple and efficient in situations where the traffic is mostly directed from or to a central node, such as the control traffic between routers and a controller of a Software-Defined Networking (SDN) infrastructure or an Autonomic Control Plane (ACP).

But stretch in P2P routing is counter-productive to both reliability and latency as it introduces additional delay and chances of loss. As a result, [RPL] is not a good fit for the use cases listed in the RAW use cases document [RFC9450], which demand high availability and reliability and, as a consequence, require both short and diverse paths.

3.2. Multi-Topology Routing and Loop Avoidance

RPL first forms a default route in each node towards the Root, and those routes together coalesce as a DAG oriented upwards. RPL then constructs routes to destinations signaled as Targets in the reverse direction, down the same DODAG. To do so, a RPL Instance can be operated in either RPL Storing or Non-Storing MOP. The default route towards the Root is maintained aggressively and may change while a packet progresses without causing loops, so the packet will still reach the Root.

In Non-Storing Mode, each node advertises itself as a Target directly to the Root, indicating the parents that may be used to reach itself. Recursively, the Root builds and maintains an image of the whole DODAG in memory and leverages that abstraction to compute source route paths for the packets to their destinations down the DODAG. When a node changes its point(s) of attachment to the DODAG, it takes a single unicast packet to the Root along the default route to update it, and the connectivity to the node is restored immediately; this mode is preferable for use cases where internet connectivity is dominant or when the Root controls the network activity in the nodes, which is the case in this specification.

In Storing Mode, the routing information percolates upwards, and each node maintains the routes to the subDAG of its descendants down the DODAG. The maintenance is lazy; it is either reactive upon receiving traffic or a slow background process. Packets flow via the common parent and the routing stretch is reduced, compared to the Non-Storing MOP, for better P2P connectivity. However, a new route takes a longer time to propagate to the Root, since it takes time for the Distance Vector protocol to operate hop by hop, and the connectivity from the Internet to the node is restored more slowly upon node movement.

Either way, the RPL routes are injected by the Target nodes in a distributed fashion. To complement RPL and eliminate routing stretch, this specification introduces a hybrid mode that combines Storing and Non-Storing operations to build and project routes onto the nodes where they should be installed. This specification uses the term "P-Route" to refer to those routes.

In the simplest mode of this specification, Storing Mode P-Routes can be deployed to complete the path between the hops described in the loose SRH in the main DODAG. In that case, all the routes (source routed and P-Routes) belong to the Routing Information Base (RIB) associated with the main Instance. Storing Mode P-Routes are referred to as segments in this specification.

A set of P-Routes can also be projected to form a dotted-line underlay of the main Instance and provide Traffic-Engineered paths for an application. In that case, the P-Routes are installed in Non-Storing Mode, and the set of P-Routes is called a Track. A Track is associated with its own RPL Instance and, as any RPL Instance, with its own RIB. As a result, each Track defines a routing topology in the RPL domain. As for the main DODAG, segments associated to the Track Instance may be deployed to establish the complete path between loose source routing hops using segments expressed as Storing Mode P-Routes.

Routing in a multi-topology domain may cause loops unless strict rules are applied. This specification defines two strict orders to ensure loop avoidance when P-Routes are used in a RPL domain: one between forwarding methods and one between RPL Instances, which are routing topologies.

The first order is strict and complete and relates to the forwarding method and, more specifically, the origin of the information used in the next-hop computation. The possible forwarding methods are: 1) to a direct next hop, 2) to an indirect neighbor via a common neighbor, 3) along a segment, and 4) along a nested Track. The methods are strictly ordered as listed above; see more in Section 6.7. A forwarding method may leverage any of the lower-order ones, but never one with a higher order; for instance, when forwarding a packet along a segment, the router may use direct or indirect neighbors but cannot use a Track. The lower-order methods have a strict precedence, so the router will always prefer a direct neighbor over an indirect one or a segment within the current RPL Instance over another Track.

The second order is strict and partial and applies between RPL Instances. It allows the RPL node to detect an error in the state installed by the PCE, e.g., after a desynchronization. That order must be defined by the administrator for the RPL domain and defines a DODAG of underlay RPL Instances with the main Instance as the Root. The relation of RPL Instances may be represented as a DODAG of Instances where the main Instance is the Root. The rule is that a RPL Instance may leverage another RPL Instance as an underlay if and only if that other Instance is one of its descendants in the graph. Supporting this method is **OPTIONAL** for nested Tracks and **REQUIRED** between a Track Instance and the main Instance. It may be done using network management or future extensions to this specifications. When the DODAG of underlay Instances is not provided, the RPL nodes consider by default that all Track Instances are children of the main Instance, and they do not attempt to validate the order for nested Tracks, trusting the PCE implicitly. As a result, a packet that is being forwarded along the main Instance may be encapsulated in any Track, but a packet that was forwarded along a Track **MUST NOT** be forwarded along the default route of the main Instance.

3.3. Requirements

3.3.1. Loose Source Routing

A RPL implementation operating in a very constrained LLN typically uses the Non-Storing MOP as represented in Figure 2. In that mode, a RPL node indicates a parent-child relationship to the Root, using a Destination Advertisement Object (DAO) that is unicast from the node directly to the Root, and the Root typically builds a source-routed path to a destination down the DODAG by recursively concatenating this information.

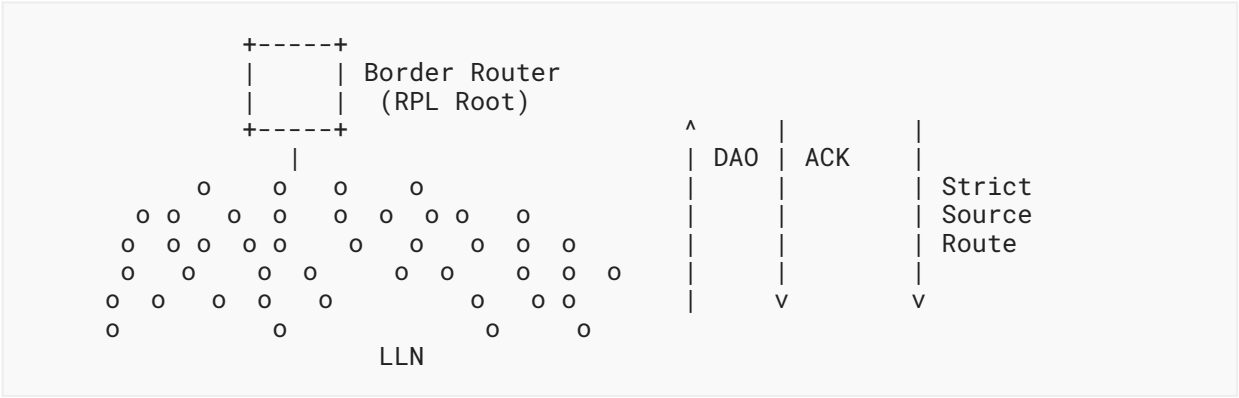

Figure 2: RPL Non-Storing Mode of Operation

Based on the parent-children relationships expressed in the Non- Storing DAO messages, the Root possesses topological information about the whole network, though this information is limited to the structure of the DODAG for which it is the destination. A packet that is generated within the domain will always reach the Root, which can then apply source routing information to reach the destination if the destination is also in the DODAG. Similarly, a packet coming from the outside of the domain for a destination that is expected to be in a RPL domain reaches the Root. This results in the wireless bandwidth near the Root being the limiting factor for all transmissions towards or within the domain, and the Root is a single point of failure for all connectivity to nodes within its domain.

The RPL Root must add a Source Routing Header to all downward packets. As a network grows, the size of the Source Routing Header increases with the depth of the network. In some use cases, a RPL network forms long lines along physical structures like streets with lighting. Limiting the packet size is beneficial to the energy budget, directly for the current transmission and also indirectly since it reduces the chances of frame loss and energy spent in retries, e.g., by ARQ over one hop at Layer 2 or end to end at upper layers. Using smaller packets also reduces the chances of packet fragmentation, which is highly detrimental to the LLN operation, in particular when fragments are forwarded but not recovered; see [RFC8930] compared to [RFC8931] for more details.

A limited amount of well-targeted routing state would allow the source routing operation to be loose as opposed to strict and would reduce the overhead of routing information in packets. Because the capability to store routing state in every node is limited, the decision of which route is installed where can only be optimized with global knowledge of the system, knowledge that the Root or an associated PCE may possess by means that are outside the scope of this specification.

Being on path for all packets in Non-Storing Mode, the Root may determine the number of P2P packets in its RPL domain per source and destination, the latency incurred, and the amount of energy and bandwidth that is consumed to reach itself and then back down, including possible fragmentation when encapsulating larger packets. Enabling a shorter path that would not traverse the Root for select P2P sources/destinations may improve the latency, lower the consumption of constrained resources, free bandwidth at the bottleneck near the Root, improve the delivery ratio, and reduce the latency for those P2P flows; this would be a global benefit for all flows by reducing the load at the Root.

To limit the need for RPL Source Route Headers in deep networks, one possibility is to store a routing state associated with the main DODAG in select RPL routers down the path. The Root may elide the sequence of routers that is installed in the network from its RPL Source Route Header, which therefore becomes loose, in contrast to being strict in [RPL].

3.3.2. Forward Routes

[RPL] optimizes P2MP routes from the Root, MP2P routes towards the Root, and routes from/to the outside of the RPL domain when the Root also serves as the border router. All routes are installed North-South (a.k.a. up/down) along the RPL DODAG. Peer-to-Peer forward routes in a RPL network will generally experience elongated (stretched) paths rather than direct (optimized) paths, since routing between two nodes always happens via a common parent, as illustrated in Figure 3:

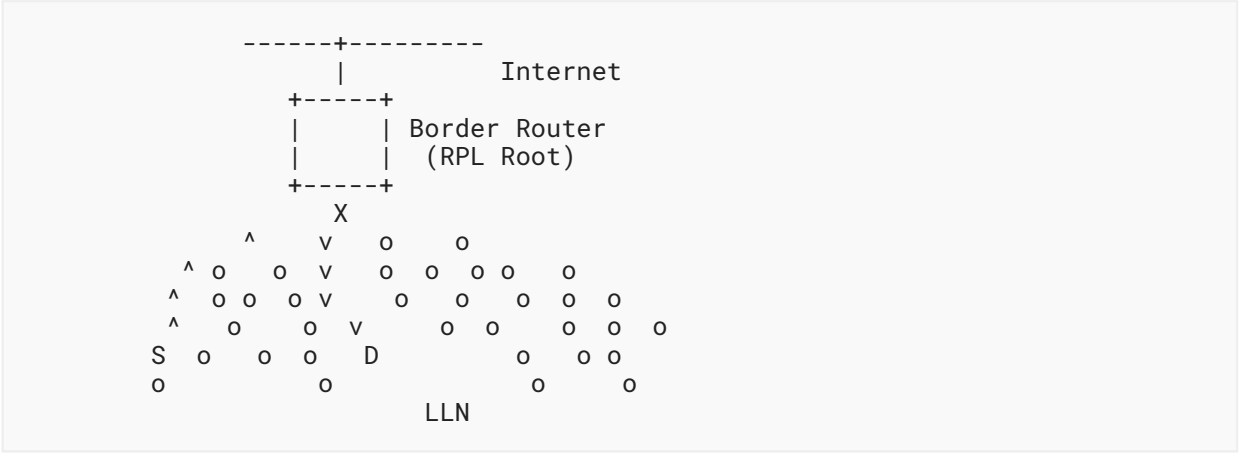

Figure 3: Routing Stretch Between S and D via Common Parent X Along North-South Paths

As described in [RFC9008], the amount of stretch depends on the MOP:

- In Non-Storing Mode, all packets routed within the DODAG flow all the way up to the Root of the DODAG. If the destination is in the same DODAG, the Root must encapsulate the packet to place an RH that has the strict source route information down the DODAG to the destination. This will be the case even if the destination is relatively close to the source and the Root is relatively far off.
- In Storing Mode, unless the destination is a child of the source, the packets will follow the default route up the DODAG as well. If the destination is in the same DODAG, they will eventually reach a common parent that has a route to the destination; at worst, the common parent may also be the Root. From that common parent, the packet will follow a path down the DODAG that is optimized for the Objective Function that was used to build the DODAG.

It turns out that it is often beneficial to enable direct P2P routes if either the RPL route presents a stretch from the shortest path or the new route is engineered with a different objective, and this is even more critical in Non-Storing Mode than it is in Storing Mode because the routing stretch is wider. For that reason, earlier work within the IETF was introduced: the "Reactive Discovery of Point-to-Point Routes in Low-Power and Lossy Networks" [RFC6997], which specifies a distributed method for establishing optimized P2P routes. This specification proposes an alternative based on centralized route computation.

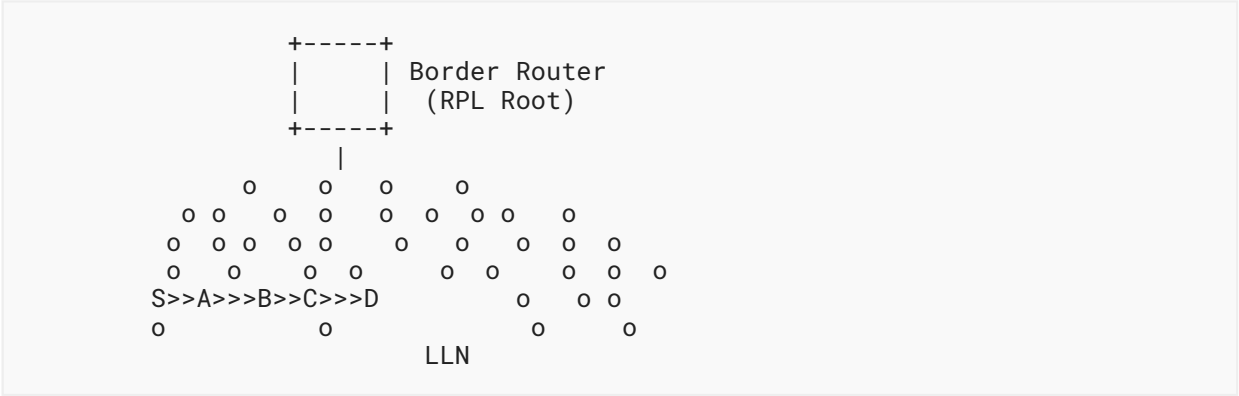

Figure 4: More Direct Forward Route Between S and D

The requirement is to install additional routes in the RPL routers, to reduce the stretch of some P2P routes and maintain the characteristics within a given Service Level Objective (SLO), e.g., in terms of latency and/or reliability.

3.4. On Tracks

3.4.1. Building Tracks with RPL

The concept of a Track was introduced in the 6TiSCH architecture [RFC9030] as a collection of potential protection paths that leverage redundant forwarding solutions along the way. This can be a DODAG or a more complex structure that is only partially acyclic (e.g., per packet).

With this specification, a Track is shaped as a DODAG, and following the directed edges leads to a Track ingress. Storing Mode P-DAO messages follow the direction of the edges to set up routes for traffic that flows the other way, towards the Track egress(es). If there is a single Track egress, then the Track is reversible so that another DODAG may be formed by reversing the direction of each edge. A node at the ingress of more than one segment in a Track may use one or more of these segments to forward a packet inside the Track.

A RPL Track is a collection of (one or more) parallel loose source-routed sequences of nodes ordered from ingress to egress, each forming a protection path. The nodes in a Track are directly connected, reachable via existing Tracks as illustrated in Section 3.5.2.3 or joined with strict segments of other nodes as shown in Section 3.5.1.3. The protection paths are expressed in RPL Non-Storing Mode and require an encapsulation to add a RPL Source Route Header, whereas the segments are expressed in RPL Storing Mode.

A path provides only one path between the ingress and egress. It comprises exactly one protection path. A stand-alone segment implicitly defines a path from its ingress to egress.

A Complex Track forms a graph that provides a collection of potential paths to provide redundancy for the packets, either as a collection of protection paths that may be parallel or interleaved at certain points or as a more generic DODAG.

3.4.2. Tracks and RPL Instances

Section 5.1 of [RPL] describes the RPL Instance and its encoding. There can be up to 128 Global RPL Instances, for which there can be one or more DODAGs, and there can be 64 Local RPL Instances, with a namespace that is indexed by a DODAGID, where the DODAGID is a Unique Local Address (ULA) or a Global Unicast Address (GUA) of the Root of the DODAG. Bit 0 (most significant) is set to 1 to signal a Local RPLInstanceID, as shown in Figure 5. By extension, this specification expresses the value of the RPLInstanceID as a single integer between 128 and 191, representing both the Local RPLInstanceID in 0..63 in the rightmost bits and bit 0 set.

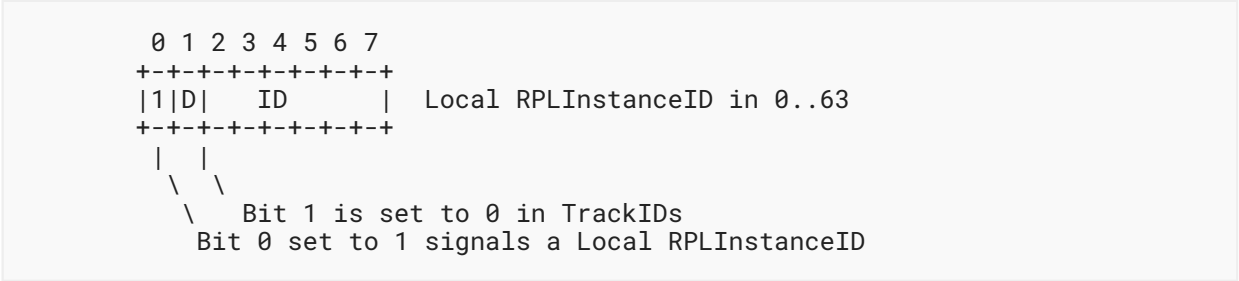

Figure 5: Local RPLInstanceID Encoding

A Track typically forms an underlay to the main Instance and is associated with a Local RPL Instance from which the RPLInstanceID is used as the TrackID. When a packet is placed on a Track, it is IP-in-IP encapsulated with a RPL Option containing RPL Packet Information (RPI) that signals the RPLInstanceID. The encapsulating source IP address and RPI Instance are set to the Track ingress IP address and Local RPLInstanceID, respectively; see more in Section 6.3.

A Track typically offers service protection across several protection paths. As a degraded form of a Track, a path made of a single protection path (i.e., offering no protection) can be used as an alternative to a segment for forwarding along a RPL Instance. In that case, instead of following native routes along the Instance, the packets are encapsulated to signal a more-specific source-routed path between the loose hops in the encapsulated Source Routing Header.

If the encapsulated packet follows a Global Instance, then the protection path may be part of that Global Instance as well, e.g., the Global Instance of the main DODAG. This can only be done for Global Instances because the ingress node that encapsulates the packets over the protection path is not the Root of the Instance, so the source address of the encapsulated packet cannot be used to determine the Track along the way.

3.5. Path Signaling

This specification enables setting up a P-Route along either a protection path or a segment. A P-Route is installed and maintained by the Root of the main DODAG using an extended RPL DAO message called a P-DAO, and a Track is composed of the combination of one or more P-Routes. In order to clarify the techniques that may be used to install a P-Route, this section uses the simple case of the path illustrated in Figure 6. Thus, the goal is to build a path from node A to E for packets towards E's neighbors F and G along A, B, C, D, and E as opposed to via the Root:

Figure 6: Reference Track

A P-DAO message for a Track signals the TrackID in the RPLInstanceID field. In the case of a Local RPL Instance, the address of the Track ingress is used as the source to encapsulate packets along the Track. The Track is signaled in the DODAGID field of the P-DAO Base Object; see Figure 8.

This specification introduces the Via Information Option (VIO) to signal a sequence of hops in a protection path or a segment in the P-DAO messages, either in Storing Mode (SM-VIO) or in Non-Storing Mode (NSM-VIO). One P-DAO message contains a single VIO, which is associated to one or more RPL Target Options that signal the destination IPv6 addresses that can reached along the Track (see more in Section 5.3).

Before diving deeper into Track and segment signaling and operation, this section provides examples of how route projection works through variations of a simple example. This simple example illustrates the case of host routes, though RPL Targets can also be prefixes.

Conventionally, we use ==> to represent a strict hop and --> for a loose hop. We use "-to-", such as in C==>D==>E-to-F, to represent comma-separated Targets, e.g., F is a Target for segment C==>D==>E. In the example below, A is the Track ingress and E is the Track egress. C is a stitching

point. F and G are "external" Targets for the Track and become reachable from A via Track A (ingress) to E (egress and implicit Target in Non-Storing Mode), leading to F and G (explicit Targets).

In a general manner, the desired outcome is as follows:

- Targets are E, F, and G
- P-DAO 1 signals C==>D==>E
- P-DAO 2 signals A==>B==>C
- P-DAO 3 signals F and G via the A-->E Track

P-DAO 3 may be omitted if P-DAOs 1 and 2 signal F and G as Targets.

Loose sequences of hops are expressed in Non-Storing Mode; this is why P-DAO 3 contains an NSM-VIO. With this specification:

- The DODAGID to be used by the ingress as the source address is signaled in the DAO Base Object (see Figure 8).
- The via list in the VIO is encoded as an SRH-6LoRH (see Figure 16), and it starts with the address of the first-hop node after the ingress node in the loose hop sequence.
- The via list ends with the address of the egress node.

> Note 1: The egress of a Non-Storing Mode P-Route is implicitly a target; it is not listed in the RPL Target Options but is still accounted for as if it was. The only exception is when the egress is the only address listed in the VIO, in which case it would indicate via itself, which would be nonsensical.

> Note 2: By design, the list of nodes in a VIO in Non-Storing Mode is exactly the list that shows in the encapsulation SRH. So in the cases detailed below, if the Mode of the P-DAO is Non-Storing, then the VIO row can be read as indicating the SRH as well.

3.5.1. Using Storing Mode Segments

A==>B==>C and C==>D==>E are segments of the same Track. Note that the Storing Mode signaling imposes strict continuity in a segment, since the P-DAO is passed hop by hop, as a classical DAO is, along the reverse datapath that it signals. One benefit of strict routing is that loops are avoided along the Track.

3.5.1.1. Stitched Segments

In this formulation:

- P-DAO 1 signals C==>D==>E-to-F,G
- P-DAO 2 signals A==>B==>C-to-F,G

Storing Mode P-DAO 1 is sent to E, and when it is successfully acknowledged, Storing Mode P-DAO 2 is sent to C as follows:

Field	P-DAO 1 to E	P-DAO 2 to C
Mode	Storing	Storing
Track ingress	A	A
(DODAGID, TrackID)	(A, 129)	(A, 129)
SegmentID	1	2
VIO	C, D, E	A, B, C
Targets	F, G	F, G

Table 1: P-DAO Messages

As a result, the RIBs are set as follows:

Node	Destination	Origin	Next Hop(s)	TrackID
E	F, G	P-DAO 1	Neighbor	(A, 129)
D	E	P-DAO 1	Neighbor	(A, 129)
"	F, G	P-DAO 1	E	(A, 129)
C	D	P-DAO 1	Neighbor	(A, 129)
"	F, G	P-DAO 1	D	(A, 129)
B	C	P-DAO 2	Neighbor	(A, 129)
"	F, G	P-DAO 2	C	(A, 129)
A	B	P-DAO 2	Neighbor	(A, 129)
"	F, G	P-DAO 2	B	(A, 129)

Table 2: RIB Settings

> Note: The " sign is used throughout the tables in this document to indicate the same value as in the row above.

Packets originating at A and going to F or G do not require encapsulation as the RPI can be placed in the native header chain. For packets that it routes, A must encapsulate to add the RPI that signals the TrackID; the outer headers of the packets that are forwarded along the Track have the following settings:

Header	IPv6 Source Address	IPv6 Destination Address	TrackID in RPI
Outer	A	F or G	(A, 129)
Inner	Any but A	F or G	N/A

Table 3: Packet Header Settings

As an example, say that A has a packet for F. Using the RIB in Table 2:

- From P-DAO 2: A forwards to B, and B forwards to C.
- From P-DAO 1: C forwards to D, and D forwards to E.
- From Neighbor Cache Entry: E delivers the packet to F.

3.5.1.2. External Routes

In this example, we consider F and G as destinations that are external to the Track as a DODAG, as discussed in Section 4.1.1 of [RFC9008]. We then apply the directives for encapsulating in that case (see more in Section 6.7).

In this formulation, we set up the protection path explicitly, which creates less routing state in intermediate hops at the expense of larger packets to accommodate source routing:

- P-DAO 1 signals C==>D==>E-to-E
- P-DAO 2 signals A==>B==>C-to-E
- P-DAO 3 signals F and G via the A-->E-to-F,G Track

Storing Mode P-DAOs 1 and 2 and Non-Storing Mode P-DAO 3 are sent to E, C, and A, respectively, as follows:

	P-DAO 1 to E	P-DAO 2 to C	P-DAO 3 to A
Mode	Storing	Storing	Non-Storing
Track ingress	A	A	A
(DODAGID, TrackID)	(A, 129)	(A, 129)	(A, 129)
SegmentID	1	2	3
VIO	C, D, E	A, B, C	E
Targets	E	E	F, G

Table 4: P-DAO Messages

Note in the above that E is not an implicit Target in Storing Mode, so it must be added in the RPL Target Option (RTO) for P-DAOs 1 and 2. E is not an implicit Target for P-DAO 3 either, since E is the only entry in the VIO.

As a result, the RIBs are set as follows:

Node	Destination	Origin	Next Hop(s)	TrackID
E	F, G	P-DAO 1	Neighbor	(A, 129)
D	E	P-DAO 1	Neighbor	(A, 129)
C	D	P-DAO 1	Neighbor	(A, 129)
"	E	P-DAO 1	D	(A, 129)
B	C	P-DAO 2	Neighbor	(A, 129)
"	E	P-DAO 2	C	(A, 129)
A	B	P-DAO 2	Neighbor	(A, 129)
"	E	P-DAO 2	B	(A, 129)
"	F, G	P-DAO 3	E	(A, 129)

Table 5: RIB Settings

Packets from A to E do not require an encapsulation. In the tables below, this is why E may show as an IPv6 destination address only if the IPv6 source address X is different from A. Conversely, the encapsulation is always done when the IPv6 destination address is F or G. Other destination addresses do not match this P-Route and are not subject to encapsulation.

The outer headers of the packets that are forwarded along the Track have the following settings:

Header	IPv6 Source Address	IPv6 Destination Address	TrackID in RPI
Outer	A	E	(A, 129)
Inner	X	Either F or G. If X!=A, E is also permitted.	N/A

Table 6: Packet Header Settings

As an example, say that A has a packet for F. Using the RIB in Table 5:

- From P-DAO 3: A encapsulates the packet and sends it down the Track signaled by P-DAO 3, with the outer header above. Now the packet destination is E.
- From P-DAO 2: A forwards to B, and B forwards to C.
- From P-DAO 1: C forwards to D, and D forwards to E; E decapsulates the packet.
- From Neighbor Cache Entry: E delivers packets to F or G.

3.5.1.3. Segment Routing

In this formulation, protection paths are leveraged to combine segments and form a graph. The packets are source routed from a segment to the next to adapt the path:

- P-DAO 1 signals C==>D==>E-to-E
- P-DAO 2 signals A==>B-to-B,C
- P-DAO 3 signals F and G via the A-->C-->E-to-(E),F,G Track

Storing Mode P-DAOs 1 and 2 and Non-Storing Mode P-DAO 3 are sent to E, B, and A, respectively, as follows:

	P-DAO 1 to E	P-DAO 2 to B	P-DAO 3 to A
Mode	Storing	Storing	Non-Storing
Track ingress	A	A	A
(DODAGID, TrackID)	(A, 129)	(A, 129)	(A, 129)
SegmentID	1	2	3
VIO	C, D, E	A, B	C, E
Targets	E	B, C	F, G

Table 7: P-DAO Messages

Note in the table above that the segment can terminate at the loose hop as used in the example of P-DAO 1 or at the previous hop as done with P-DAO 2. Both methods are possible on any segment joined by a loose protection path. P-DAO 1 generates more signaling since E is the segment egress when D could be, but a benefit is that it validates that the connectivity between D and E still exists.

As a result, the RIBs are set as follows:

Node	Destination	Origin	Next Hop(s)	TrackID
E	F, G	P-DAO 1	Neighbor	(A, 129)
D	E	P-DAO 1	Neighbor	(A, 129)
C	D	P-DAO 1	Neighbor	(A, 129)
"	E	P-DAO 1	D	(A, 129)
B	C	P-DAO 2	Neighbor	(A, 129)
A	B	P-DAO 2	Neighbor	(A, 129)

Node	Destination	Origin	Next Hop(s)	TrackID
"	C	P-DAO 2	B	(A, 129)
"	E, F, G	P-DAO 3	C, E	(A, 129)

Table 8: RIB Settings

Packets originated at A to E do not require an encapsulation, but they carry an SRH via C. The outer headers of the packets that are forwarded along the Track have the following settings:

Header	IPv6 Source Address	IPv6 Destination Address	TrackID in RPI
Outer	A	C until C then E	(A, 129)
Inner	X	Either F or G. If X!=A, E is also permitted.	N/A

Table 9: Packet Header Settings

As an example, say that A has a packet for F. Using the RIB in Table 8:

- From P-DAO 3: A encapsulates the packet the Track signaled by P-DAO 3, with the outer header above. Now the destination in the IPv6 header is C, and an SRH signals that the final destination is E.
- From P-DAO 2: A forwards to B, and B forwards to C.
- From P-DAO 3: C processes the SRH and sets the destination in the IPv6 header to E.
- From P-DAO 1: C forwards to D, and D forwards to E; E decapsulates the packet.
- From the Neighbor Cache Entry: E delivers packets to F or G.

3.5.2. Using Non-Storing Mode Joining Tracks

In this formulation:

- P-DAO 1 signals C==>D==>E-to-(E),F,G
- P-DAO 2 signals A==>B==>C-to-(C),E,F,G

A==>B==>C and C==>D==>E are Tracks expressed as Non-Storing Mode P-DAOs.

3.5.2.1. Stitched Tracks

Non-Storing Mode P-DAO 1 and 2 are sent to C and A, respectively, as follows:

	P-DAO 1 to C	P-DAO 2 to A
Mode	Non-Storing	Non-Storing
Track ingress	C	A

	P-DAO 1 to C	P-DAO 2 to A
(DODAGID, TrackID)	(C, 131)	(A, 131)
SegmentID	1	1
VIO	D, E	B, C
Targets	F, G	E, F, G

Table 10: P-DAO Messages

As a result, the RIBs are set as follows (using "ND" to indicate that the address is discovered by IPv6 Neighbor Discovery [RFC4861] [RFC8505] or an equivalent method):

Node	Destination	Origin	Next Hop(s)	TrackID
E	F, G	ND	Neighbor	Any
D	E	ND	Neighbor	Any
C	D	ND	Neighbor	Any
"	E, F, G	P-DAO 1	D, E	(C, 131)
B	C	ND	Neighbor	Any
A	B	ND	Neighbor	Any
"	C, E, F, G	P-DAO 2	B, C	(A, 131)

Table 11: RIB Settings

Packets originated at A to E, F, and G could be generated with the RPI and the SRH and no encapsulation. Alternatively, A may generate a native packet to the target and then encapsulate it with an RPI and an SRH indicating the source-routed path leading to E, as it would for a packet that it routes coming from another node. This is effectively the same case as for packets generated by the Root in a RPL network in Non-Storing Mode; see Section 8.1.3 of [RFC9008]. The latter is often preferred since it leads to a single code path, and when the destination is F or G, it does not need to understand and process the RPI or the SRH. Either way, the packets to E, F, or G carry an SRH via B and C, and when they reach C, C needs to encapsulate them again to add an SRH via D and E. The encapsulating headers of packets that are forwarded along the Track between C and E have the following settings:

Header	IPv6 Source Address	IPv6 Destination Address	TrackID in RPI
Outer	C	D until D then E	(C, 131)

Header	IPv6 Source Address	IPv6 Destination Address	TrackID in RPI
Inner	X	E, F, or G	N/A

Table 12: Packet Header Settings Between C and E

As an example, say that A has a packet for F. Using the RIB in Table 11:

- From P-DAO 2: A encapsulates the packet with a destination of F in the Track signaled by P-DAO 2. The outer header has source A, destination B, an SRH that indicates C as the next loose hop, and an RPI indicating a TrackID of 131 from A's namespace, which is distinct from a TrackID of 131 from C's.
- From the SRH: Packets forwarded by B have source A, destination C, a consumed SRH, and an RPI indicating a TrackID of 131 from A's namespace. C decapsulates.
- From P-DAO 1: C encapsulates the packet with a destination of F in the Track signaled by P-DAO 1. The outer header has source C, destination D, an SRH that indicates E as the next loose hop, and an RPI indicating a TrackID of 131 from C's namespace. E decapsulates.

3.5.2.2. External Routes

In this formulation:

- P-DAO 1 signals C==>D==>E-to-(E)
- P-DAO 2 signals A==>B==>C-to-(C),E
- P-DAO 3 signals F and G via the A-->E-to-F,G Track

Non-Storing Mode P-DAO 1 is sent to C, and Non-Storing Mode P-DAOs 2 and 3 are sent to A, as follows:

	P-DAO 1 to C	P-DAO 2 to A	P-DAO 3 to A
Mode	Non-Storing	Non-Storing	Non-Storing
Track ingress	C	A	A
(DODAGID, TrackID)	(C, 131)	(A, 129)	(A, 141)
SegmentID	1	1	1
VIO	D, E	B, C	E
Targets		E	F, G

Table 13: P-DAO Messages

Note in the table above that E is an implicit Target in P-DAO 1 and so is C in P-DAO 2. As Non-Storing Mode egress node addresses, they are not listed in the respective RTOs.

As a result, the RIBs are set as follows:

Node	Destination	Origin	Next Hop(s)	TrackID
E	F, G	ND	Neighbor	Any
D	E	ND	Neighbor	Any
C	D	ND	Neighbor	Any
"	E	P-DAO 1	D, E	(C, 131)
B	C	ND	Neighbor	Any
A	B	ND	Neighbor	Any
"	C, E	P-DAO 2	B, C	(A, 129)
"	F, G	P-DAO 3	E	(A, 141)

Table 14: RIB Settings

The encapsulating headers of packets that are forwarded along the Track between C and E have the following settings:

Header	IPv6 Source Address	IPv6 Destination Address	TrackID in RPI
Outer	C	D until D then E	(C, 131)
Middle	A	E	(A, 141)
Inner	X	E, F, or G	N/A

Table 15: Packet Header Settings

As an example, say that A has a packet for F. Using the RIB in Table 14:

- From P-DAO 3: A encapsulates the packet with a destination of F in the Track signaled by P-DAO 3. The outer header has source A, destination E, and an RPI indicating a TrackID of 141 from A's namespace. This recurses with the following.
- From P-DAO 2: A encapsulates the packet with a destination of E in the Track signaled by P-DAO 2. The outer header has source A, destination B, an SRH that indicates C as the next loose hop, and an RPI indicating a TrackID of 129 from A's namespace.
- From the SRH: Packets forwarded by B have source A, destination C, a consumed SRH, and an RPI indicating a TrackID of 129 from A's namespace. C decapsulates.
- From P-DAO 1: C encapsulates the packet with a destination of E in the Track signaled by P-DAO 1. The outer header has source C, destination D, an SRH that indicates E as the next loose hop, and an RPI indicating a TrackID of 131 from C's namespace. E decapsulates.

3.5.2.3. Segment Routing

In this formulation:

- P-DAO 1 signals C==>D==>E-to-(E)
- P-DAO 2 signals A==>B-to-C
- P-DAO 3 signals F and G via the A-->C-->E-to-(E),F,G Track

Non-Storing Mode P-DAO 1 is sent to C, and Non-Storing Mode P-DAOs 2 and 3 are sent to A, as follows:

	P-DAO 1 to C	P-DAO 2 to A	P-DAO 3 to A
Mode	Non-Storing	Non-Storing	Non-Storing
Track ingress	C	A	A
(DODAGID, TrackID)	(C, 131)	(A, 129)	(A, 141)
SegmentID	1	1	1
VIO	D, E	B	C, E
Targets		C	F, G

Table 16: P-DAO Messages

As a result, the RIBs are set as follows:

Node	Destination	Origin	Next Hop(s)	TrackID
E	F, G	ND	Neighbor	Any
D	E	ND	Neighbor	Any
C	D	ND	Neighbor	Any
"	E	P-DAO 1	D, E	(C, 131)
B	C	ND	Neighbor	Any
A	B	ND	Neighbor	Any
"	B, C	P-DAO 2	C	(A, 129)
"	E, F, G	P-DAO 3	C, E	(A, 141)

Table 17: RIB Settings

The encapsulating headers of packets that are forwarded along the Track between A and B have the following settings:

Header	IPv6 Source Address	IPv6 Destination Address	TrackID in RPI
Outer	A	B until D then E	(A, 129)
Middle	A	C	(A, 141)
Inner	X	E, F, or G	N/A

Table 18: Packet Header Settings

The encapsulating headers of packets that are forwarded along the Track between B and C have the following settings:

Header	IPv6 Source Address	IPv6 Destination Address	TrackID in RPI
Outer	A	C	(A, 141)
Inner	X	E, F, or G	N/A

Table 19: Packet Header Settings

The encapsulating headers of packets that are forwarded along the Track between C and E have the following settings:

Header	IPv6 Source Address	IPv6 Destination Address	TrackID in RPI
Outer	C	D until D then E	(C, 131)
Middle	A	E	(A, 141)
Inner	X	E, F, or G	N/A

Table 20: Packet Header Settings

As an example, say that A has a packet for F. Using Table 18:

- From P-DAO 3: A encapsulates the packet with a destination of F in the Track signaled by P-DAO 3. The outer header has source A, destination C, an SRH that indicates E as the next loose hop, and an RPI indicating a TrackID of 141 from A's namespace. This recurses with the following.
- From P-DAO 2: A encapsulates the packet with a destination of C in the Track signaled by P-DAO 2. The outer header has source A, destination B, and an RPI indicating a TrackID of 129 from A's namespace. B decapsulates forwards to C based on a sibling connected route.
- From the SRH: C consumes the SRH and makes the destination E.

- From P-DAO 1: C encapsulates the packet with a destination of E in the Track signaled by P-DAO 1. The outer header has source C, destination D, an SRH that indicates E as the next loose hop, and an RPI indicating a TrackID of 131 from C's namespace. E decapsulates.

3.6. Complex Tracks

To increase the reliability of the P2P transmission, this specification enables building a collection of protection paths between the same ingress and egress Nodes and combining them within the same TrackID, as shown in Figure 7. Protection paths may be interleaved at the edges of loose hops or remain parallel.

The segments that join the loose hops of a protection path are installed with the same TrackID as the protection path. But each individual protection path and segment has its own P-RouteID that allows it to be managed separately. Two protection paths of the same Track may cross at a common node that is a member of a segment of each protection path or may be joined by additional segments. The final path of a packet may then be the result of interleaving those two (and possibly more) protection paths. In that case, the common node has more than one next hop in its RIB associated to the Track but no specific signal in the packet to indicate which segment is being followed. A next hop that can reach the loose hop is selected.

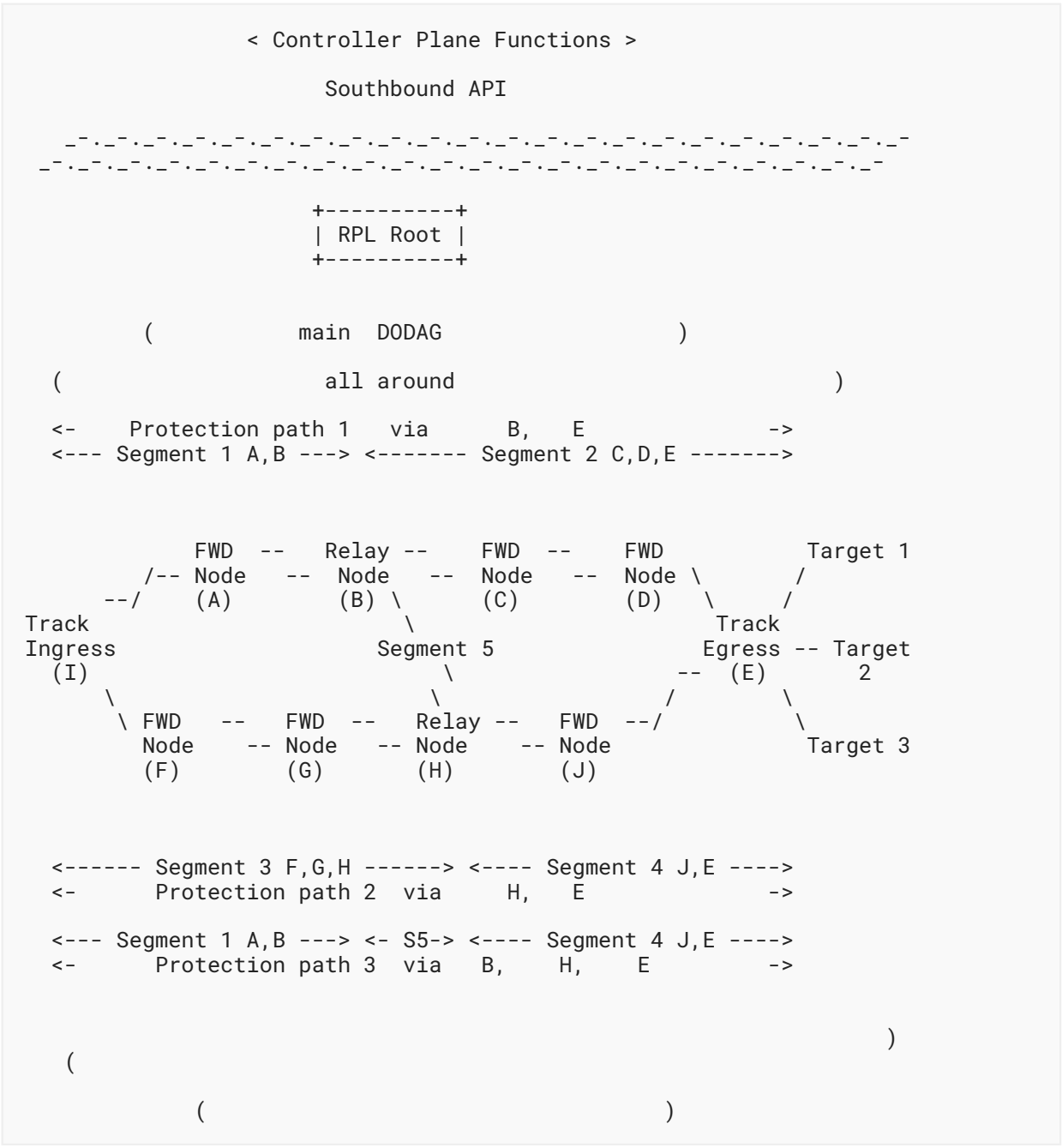

Figure 7: Segments and Tracks

Note that while this specification enables building both segments inside a protection path, for instance, segment 2 above (which is within protection path 1) and Inter-protection-path segments (i.e., North-South) such as segment 5 above (which joins protection paths 1 and 2), it does not signal which Inter-protection-path segments are available to the ingress, so the use of North-South segments and associated path redundancy functions is currently limited. The only

possibility available at this time is to define overlapping protection paths as illustrated in Figure 7, with protection path 3 that is congruent with protection path 1 until node B and that is congruent with protection path 2 from node H on, abstracting segment 5 as a forward segment.

3.7. Scope and Expectations

3.7.1. External Dependencies

This specification expects that the main DODAG is operated in RPL Non-Storing Mode to sustain the exchanges with the Root. Based on its comprehensive knowledge of the parent-child relationship, the Root can form an abstracted view of the whole DODAG topology. This document adds the capability for nodes to advertise additional sibling information to complement the topological awareness of the Root to be passed on to the PCE and enables the PCE to build more/better paths that traverse those siblings.

P-Routes require resources such as routing table space in the routers and bandwidth on the links; the amount of state that is installed in each node must be computed to fit within the node's memory, and the amount of rerouted traffic must fit within the capabilities of the transmission links. The methods used to learn the node capabilities and the resources that are available in the devices and in the network are out of scope for this document. The method to capture and report the LLN link capacity and reliability statistics are also out of scope. They may be fetched from the nodes through network management functions or other forms of telemetry such as Operations, Administration, and Maintenance (OAM).

3.7.2. Relationship to Other IETF Specifications

3.7.2.1. Extending 6TiSCH

The 6TiSCH architecture [RFC9030] leverages a centralized model that is similar to that of the DetNet architecture [RFC8655], whereby the device resources and capabilities are exposed to an external controller that installs routing states into the network based on its own Objective Functions that reside in that external entity.

3.7.2.2. Mapping to DetNet

DetNet forwarding nodes only understand the simple 1-to-1 forwarding sublayer transport operation along a segment whereas the more sophisticated relay nodes can also provide service sublayer functions such as Replication and Elimination.

One possible mapping between DetNet and this specification is to signal the relay nodes as the hops of a protection path and the forwarding nodes as the hops in a segment that join the relay nodes as illustrated in Figure 7.

3.7.2.3. Leveraging PCE

With DetNet and 6TiSCH, the component of the controller that is responsible for computing routes is a PCE. The PCE computes its routes based on its own Objective Functions, as described in [RFC4655], and typically controls the routes using the PCE Communication Protocol (PCEP) [RFC5440]. While this specification expects a PCE, and while PCEP might effectively be used between the Root and the PCE, the control protocol between the PCE and the Root is out of scope.

This specification also expects a single PCE with a full view of the network. Distributing the PCE function for a large network is out of scope. This specification uses the RPL Root as a proxy to the PCE. The PCE may be collocated with the Root or may reside in an external Controller. In the latter case, the protocol between the Root and the PCE is out of scope and mapped to RPL inside the DODAG; one possible control protocol between the Root and external PCE is for the Root to transmit the information it received in the RPL DAOs, including all the SIOs that detail the parent/child and sibling information, to the PCEs.

The algorithm to compute the paths, the protocol used by the PCE, and the metrics and link statistics involved in the computation are also out of scope. The effectiveness of the route computation by the PCE depends on the quality of the metrics that are reported from the RPL network. Which metrics are used and how they are reported are out of scope, but the expectation is that they are mostly of a long-term, statistical nature and provide visibility on link throughput, latency, stability, and availability over relatively long periods.

3.7.2.4. Providing for RAW

A recovery graph as in the RAW architecture [RAW-ARCH] can be composed of forward East-West directional segments and North-South bidirectional segments to enable additional path diversity using PREOF to select the protection paths to be used for a given datagram. This provides a dynamic balance between the reliability and availability requirements of the flows and the need to conserve energy and spectrum. This specification prepares for RAW by setting up the Tracks, but it only forms DODAGs, which are composed of aggregated end-to-end loose source-routed protection paths, joined by strict routed segments, all oriented forward.

The RAW architecture defines a data plane extension of the PCE called the Point of Local Repair (PLR) that adapts the use of the path redundancy within a Track to defeat the diverse causes of packet loss. The PLR controls the forwarding operation of the packets within a Track. This specification can use but does not impose a PLR and does not provide the policies that would select which packets are routed through which path within a Track (in other words, how the PLR may use the path redundancy within the Track). By default, the use of the available redundancy is limited to simple load balancing, and all the segments are forward unidirectional only.

A Track may be set up to reduce the load around the Root or to enable urgent traffic to flow more directly. This specification does not provide the policies that would decide which flows are routed through which Track. In a Non-Storing Mode RPL Instance, the main DODAG provides a default route via the Root, and the Tracks provide more-specific routes to the Track Targets.

4. Extending and Amending Existing RFCs

This section explains which changes are extensions and which are amendments to existing specifications. It is expected that extensions to existing specifications will not cause existing code on legacy 6LRs to malfunction, as the extensions will simply be ignored. New code is required for an extension. Those 6LRs will be unable to function in the new mechanisms and may also make the P-DAOs impossible to install. Amendments to existing specifications are situations where there are semantic changes required to existing code and where new unit tests may be required to confirm that legacy operations will continue unaffected.

4.1. Extending RFC 6550

This specification Extends RPL [RPL] to enable the Root to install forward routes inside a main DODAG that is operated as Non-Storing Mode. The Root issues a P-DAO message (see Section 4.1.1) to the Track ingress; the P-DAO message contains a new VIO that installs a strict or a loose sequence of hops to form a Track segment or a protection path, respectively.

The Projected DAO Request (P-DAO-REQ) is a new message detailed in Section 5.1. As per Section 6 of [RPL], if a node receives this message and it does not understand this new code, it discards the message. When the Root initiates communication to a node that it has not communicated with before and that it has not ascertained to implement this specification (by means such as capabilities), then the Root **SHOULD** request a Projected DAO Request Acknowledgment (PDR-ACK).

A P-DAO-REQ message enables a Track ingress to request the Track from the Root. The resulting Track is also a DODAG for which the Track ingress is the Root, and the owner is the address that serves as the DODAGID and is authoritative for the associated namespace from which the TrackID is selected. In the context of this specification, the installed route appears as a more-specific route to the Track Targets, and the Track ingress forwards the packets toward the Targets via the Track using normal longest match IP forwarding.

To ensure that the P-DAO-REQ and P-DAO messages can flow at most times, it is **RECOMMENDED** that the nodes involved in a Track maintain multiple parents in the main DODAG, advertise them all to the Root, and use them in turn to retry similar packets. It is also **RECOMMENDED** that the Root uses diverse source route paths to retry similar messages to the nodes in the Track.

4.1.1. Projected DAO

Section 6 of [RPL] introduces the RPL Control Message Options (CMOs), including the RPL Target Option (RTO) and Transit Information Option (TIO), which can be placed in RPL messages such as the DAO. A DAO message signals routing information to one or more Targets indicated in the RTOs and provides one and only one via-node in the TIO, with the via-node being the tunnel endpoint to reach the targets.

This document Amends the specification of the DAO to create the P-DAO message. This Amended DAO is signaled with a new "Projected DAO" (P) flag; see Figure 8.

A P-DAO is a special DAO message generated by the Root to install a P-Route formed of multiple hops in its DODAG. This provides a RPL-based method to install the Tracks as a collection of multiple P-Routes as expected by the 6TiSCH architecture [RFC9030].

The Root **MUST** source the P-DAO message with its address that serves as the DODAGID for the main DODAG. The receiver **MUST NOT** accept a P-DAO message that is not sent by the Root of its DODAG and **MUST** ignore such messages silently.

The 'P' flag is encoded in bit position 2 of the Flags field in the DAO Base Object. The Root **MUST** set it to 1 in a P-DAO message. Otherwise, it **MUST** be set to 0. It is set to 0 in legacy implementations as specified, respectively, in Sections 20.11 and 6.4 of [RPL].

The P-DAO is a part of control plane signaling and should not be stuck behind high traffic levels. The expectation is that the P-DAO message be sent at a high QoS level, above that of data traffic, typically with the Network Control precedence.

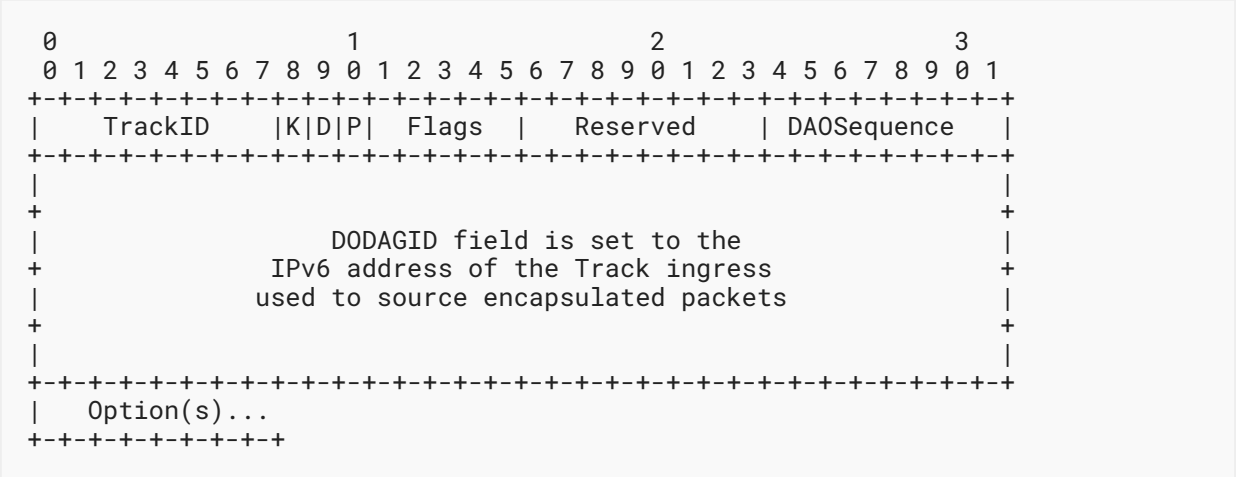

Figure 8: Projected DAO Base Object

New fields:

TrackID: The Local or Global RPLInstanceID of the DODAG that serves as the Track (see more in Section 6.3).

P: 1-bit flag.

The 'P' flag is set to 1 by the Root to signal a P-DAO; otherwise, it is set to 0.

The 'D' flag is set to 1 to signal that the DODAGID field is present. It may be set to 0 if and only if the destination address of the Projected DAO Acknowledgment (P-DAO-ACK) message is set to the IPv6 address that serves as the DODAGID, and it **MUST** be set to one otherwise, meaning that the DODAGID field **MUST** then be present.

In RPL Non-Storing Mode, the TIO and RTO are combined in a DAO message to inform the DODAG Root of all the edges in the DODAG, which are formed by the directed parent-child relationships. The DAO message signals to the Root that a given parent can be used to reach a given child. The P-DAO message generalizes the DAO to signal to the Track ingress that a Track, for which the sender is the Root, can be used to reach children and siblings of the Track egress. In both cases, options may be factorized and multiple RTOs may be present to signal a collection of children that can be reached through the parent or the Track, respectively.

4.1.2. Projected DAO Acknowledgment

This document also Amends the DAO-ACK message. The new 'P' flag signals the projected form.

The format of the P-DAO-ACK message is thus illustrated in Figure 9:

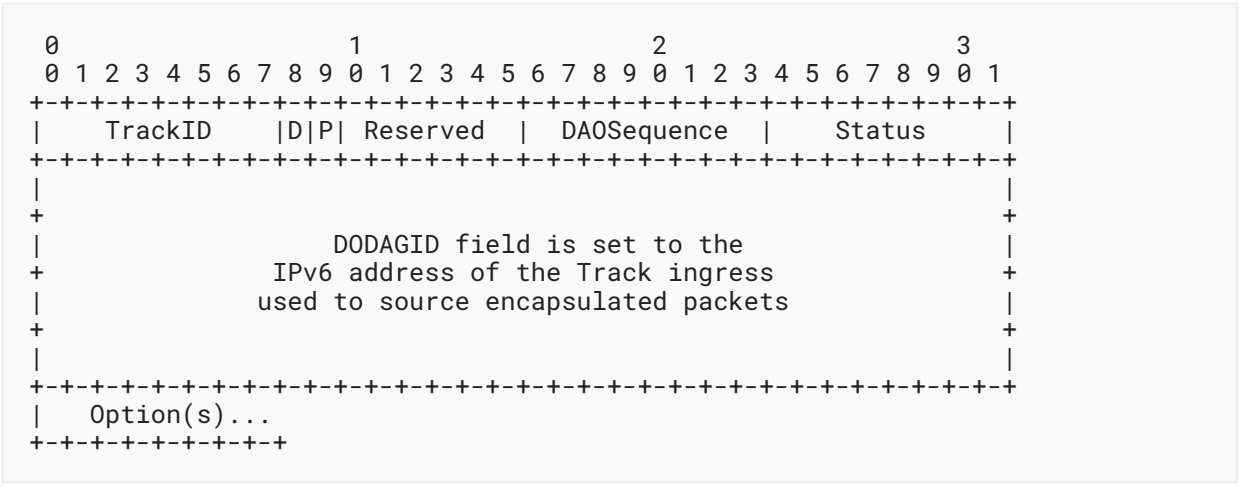

Figure 9: P-DAO-ACK Base Object

New fields:

TrackID: The Local or Global RPLInstanceID of the DODAG that serves as the Track (see more in Section 6.3).

P: 1-bit flag.

The 'P' flag is set to 1 by the Root to signal a P-DAO; otherwise, it is set to 0.

The 'D' flag is set to 1 to signal that the DODAGID field is present. It may be set to 0 if and only if the source address of the P-DAO-ACK message is set to the IPv6 address that serves as the DODAGID, and it **MUST** be set to one otherwise, meaning that the DODAGID field **MUST** then be present.

4.1.3. Via Information Option

This document Extends the CMO to create new objects called Via Information Options (VIOs). The VIOs are the multi-hop alternative to the TIOs (see more in Section 5.3). One VIO is the stateful Storing Mode VIO (SM-VIO); an SM-VIO installs a strict hop-by-hop P-Route called a Track segment. The other is the Non-Storing Mode VIO (NSM-VIO); the NSM-VIO installs a loose source-routed P-Route called a protection path at the Track ingress, which uses that state to encapsulate an IP-in-IP packet with a new Routing Header (RH) to the Track egress (see more in Section 6.7).

A P-DAO contains one or more RTOs to indicate the Target (destinations) that can be reached via the P-Route, followed by exactly one VIO that signals the sequence of nodes to be followed (see more in Section 6). There are two modes of operation for the P-Routes: Storing Mode and Non-Storing Mode (see more in Sections 6.4.2 and 6.4.3, respectively).

4.1.4. Sibling Information Option

This specification Extends the CMO to create the Sibling Information Option (SIO). The SIO is used by a RPL-Aware Node (RAN) to advertise a selection of its candidate neighbors as siblings to the Root (see more in Section 5.4). The SIO is placed in DAO messages that are sent directly to the main Root, including multicast DAO (see Section 9.10 of [RPL]).

This specification Amends rules 1 and 2 listed in Section 9.10 of [RPL] for the multicast DAO operation as follows:

OLD:

> 1. A node **MAY** multicast a DAO message to the link-local scope all-RPL-nodes multicast address.
> 2. A multicast DAO message **MUST** be used only to advertise information about the node itself, i.e., prefixes directly connected to or owned by the node, such as a multicast group that the node is subscribed to or a global address owned by the node

NEW:

> 1. A multicast DAO message **MUST** be used only to advertise information about the node (using the Target Option) and direct Link Neighbors such as learned by Neighbor Discovery (using the SIO).
> 2. The multicast DAO may be used to enable direct and indirect (via a common neighbor) P2P communication without needing the DODAG to relay the packets. The multicast DAO exposes the sender's addresses as Targets in RTOs and the sender's neighbors addresses as siblings in SIOs; this tells the sender's neighbors that the sender is willing to act as a relay between those of its neighbors that are too far apart.

4.1.5. P-DAO Request

The set of RPL Control Messages is Extended to include the P-DAO-REQ and PDR-ACK. These two new RPL Control Messages enable a RAN to request the establishment of a Track between itself (as the Track ingress Node) and a Track egress. The node makes its request by sending a new P-DAO-REQ message to the Root. The Root confirms with a new PDR-ACK message back to the requester RAN; see Section 5.1 for more.

4.1.6. Amending the RPI

Sending a packet within a RPL Local Instance requires the presence of the abstract RPI described in Section 11.2 of [RPL] in the outer IPv6 header chain (see [RFC9008]). The RPI carries a Local RPLInstanceID that, in association with either the source or the destination address in the IPv6 header, indicates the RPL Instance that the packet follows.

This specification Amends [RPL] to create a new flag that signals when a packet is forwarded along a P-Route.

Projected-Route 'P': 1-bit flag. It is set to 1 in the RPI that is added in the encapsulation when a packet is sent over a Track. It is set to 0 when a packet is forwarded along the main DODAG (as a Track), including when the packet follows a segment that joins loose hops of the main DODAG. The flag is not mutable en route.

The encoding of the 'P' flag in native format is shown in Section 4.2 while the compressed format is indicated in Section 4.3.

4.1.7. Additional Flag in the RPL DODAG Configuration Option

The DODAG Configuration option is defined in Section 6.7.6 of [RPL]. Its purpose is extended to distribute configuration information affecting the construction and maintenance of the DODAG, as well as operational parameters for RPL on the DODAG, through the DODAG. This option was originally designed with four bit positions reserved for future use as Flags.

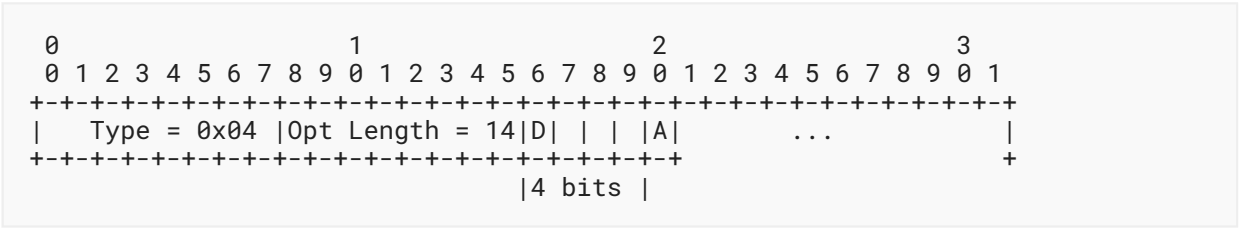

Figure 10: DODAG Configuration Option (Partial View)

This specification Amends [RPL] to define the new "Projected Routes Support" (D) flag. The 'D' flag is encoded in bit position 0 of the reserved Flags in the DODAG Configuration option (this is the most significant bit). It is set to 0 in legacy implementations as specified respectively in Sections 20.14 and 6.7.6 of [RPL].

The 'D' flag is set to 1 to indicate that this specification is enabled in the network and that the Root will install the requested Tracks when feasible upon receiving a P-DAO-REQ message.

Section 4.1.2 of [RFC9008] Amends [RPL] to indicate that the definition of the Flags applies to MOP values from zero (0) to six (6) only. For a MOP value of 7, the implementation **MUST** consider that the Root accepts P-DAO-REQ messages and will install P-Routes.

The RPL DODAG Configuration option is typically placed in a DIO message. The DIO message propagates down the DODAG to form and then maintain its structure. The DODAG Configuration option is copied unmodified from parents to children.

[RPL] states that:

> Nodes other than the DODAG root **MUST NOT** modify this information when propagating the DODAG Configuration option.

Therefore, a legacy parent propagates the 'D' flag as set by the Root, and when the 'D' flag is set to 1, it is transparently flooded to all the nodes in the DODAG.

4.2. Extending RFC 6553

"The Routing Protocol for Low-Power and Lossy Networks (RPL) Option for Carrying RPL Information in Data-Plane Datagrams" [RFC6553] describes the RPL Option for use among RPL routers to include the abstract RPI described in Section 11.2 of [RPL] in data packets.

The RPL Option is commonly referred to as the RPI even though the RPI is really the abstract information that is transported in the RPL Option. [RFC9008] updated the Option Type from 0x63 to 0x23.

This specification Extends the RPL Option to encode the 'P' flag as follows:

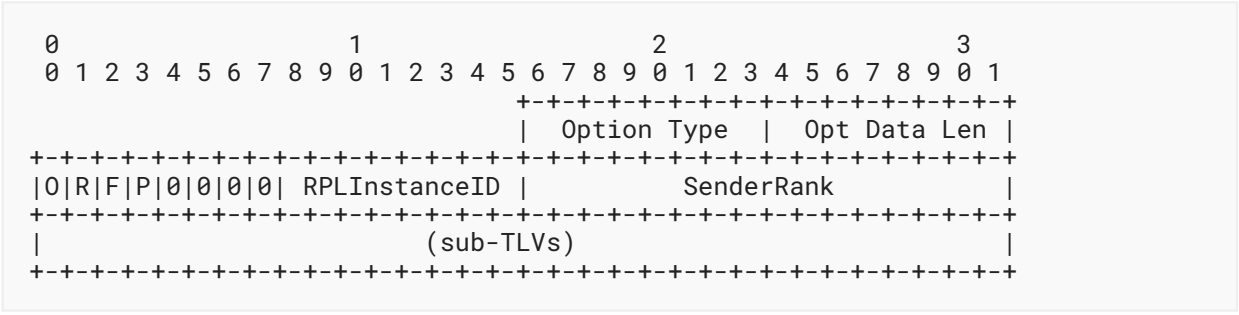

Figure 11: Amended RPL Option Format

Option Type: 0x23 or 0x63; see [RFC9008].

Opt Data Len: See [RFC6553].

'O', 'R', and 'F' flags: See [RFC6553]. These flags **MUST** be set to 0 by the sender and **MUST** be ignored by the receiver if the 'P' flag is set.

Projected-Route 'P': 1-bit flag as defined in Section 4.1.6.

RPLInstanceID: See [RFC6553]. Indicates the TrackID if the 'P' flag is set, as discussed in Section 4.1.1.

SenderRank: See [RFC6553]. This field **MUST** be set to 0 by the sender and **MUST** be ignored by the receiver if the 'P' flag is set.

4.3. Extending RFC 8138

The 6LoWPAN Routing Header specification [RFC8138] introduces a new 6LoWPAN [RFC6282] dispatch type for use in 6LoWPAN route-over topologies, which initially covers the needs of RPL data packet compression.

Section 4 of [RFC8138] presents the generic formats of the 6LoRH in two forms: Elective, which can be ignored and skipped when the router does not understand it, and Critical, which causes the packet to be dropped when the router cannot process it. The 'E' flag in the 6LoRH indicates its form. In order to skip the Elective 6LoRHs, their format imposes a fixed expression of the size, whereas the size of a Critical 6LoRH may be signaled in variable forms to enable additional optimizations.

When compression as described in [RFC8138] is used, the Root of the main DODAG that sets up the Track also constructs the compressed Routing Header (SRH-6LoRH) on behalf of the Track ingress, which avoids the complexities of optimizing SRH-6LoRH encoding in constrained code. In that case, the SRH-6LoRH is signaled in the NSM-VIO, and it is expressed in a fashion that can be placed as is in the packet encapsulation by the Track ingress.

Section 6.3 of [RFC8138] presents the formats of the 6LoWPAN RH of type 5 (RPI-6LoRH) that compresses the RPI for normal RPL operation. The format of the RPI-6LoRH is not suited for P-Routes since the 'O', 'R', and 'F' flags are not used and the Rank is unknown and ignored.

This specification Extends [RFC8138] to introduce a new 6LoRH, the P-RPI-6LoRH, that can be used in either Elective or Critical 6LoRH form; see Tables 22 and 23, respectively. The new 6LoRH **MUST** be used as a Critical 6LoRH, unless an SRH-6LoRH is present and controls the routing decision, in which case it **MAY** be used in Elective form.

The P-RPI-6LoRH is designed to compress the RPI along RPL P-Routes. Its format is as follows:

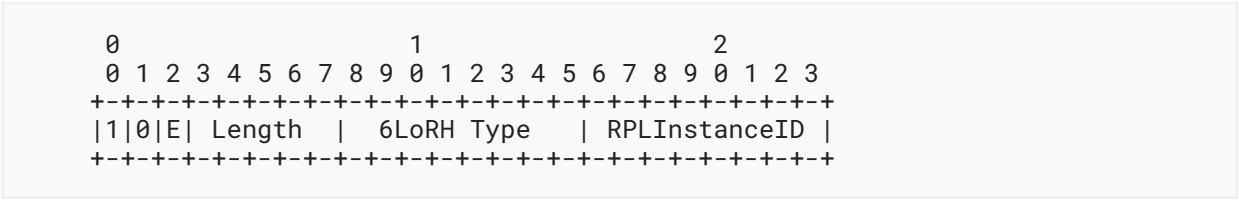

Figure 12: P-RPI-6LoRH Format

6LoRH Type: IANA has defined the value 8 for both the Elective and Critical forms.

Elective 'E': See [RFC8138]. The 'E' flag is set to 1 to indicate an Elective 6LoRH, meaning that it can be ignored when forwarding.

RPLInstanceID : In the context of this specification, the RPLInstanceID field signals the TrackID; see Sections 3.4 and 6.3.

Section 6.8 details how a Track ingress leverages the P-RPI-6LoRH header as part of the encapsulation of a packet to place it into a Track.

5. New RPL Control Messages and Options

5.1. New P-DAO Request Control Message

The P-DAO-REQ message is sent by a node in the main DODAG to the Root. It is a request to establish or refresh a Track where the node sending the P-DAO-REQ is the Track ingress, and it signals whether or not an acknowledgment called PDR-ACK is requested. A positive PDR-ACK indicates that the Track was built and that the Root commits to maintaining the Track for the negotiated lifetime.

The main Root **MAY** indicate to the Track ingress that the Track was terminated before its time; to do so, it **MUST** use an asynchronous PDR-ACK with a negative status. A status of "Transient Failure" (see Section 11.10) is an indication that the P-DAO-REQ may be retried after a reasonable time that depends on the deployment. Other negative status values indicate a permanent error; the attempt must be abandoned until a corrective action is taken at the application layer or through network management.

The Track ingress to be of the requested Track is indicated in the source IPv6 address of the P-DAO-REQ, and the TrackID is indicated in the message itself. At least one RPL Target Option **MUST** be present in the message. If more than one RPL Target Option is present, the Root will provide a Track that reaches the first listed Target and a subset of the other Targets; the details of the subset selection are out of scope. The RTO signals the Track egress (see more in Section 6.2).

The RPL Control Code for the P-DAO-REQ is 0x09. The format of the P-DAO-REQ Base Object is as follows:

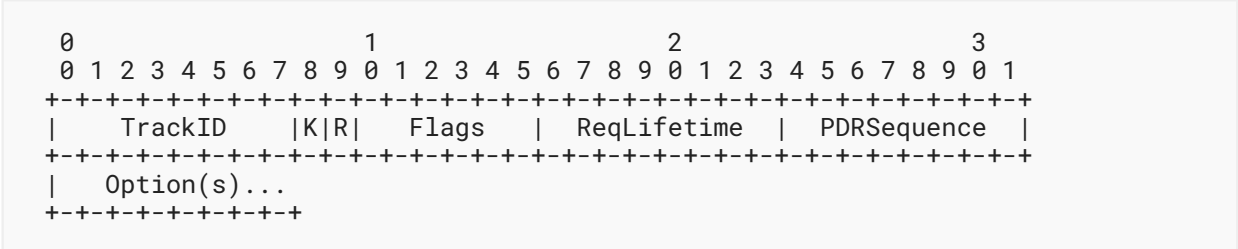

Figure 13: New P-DAO Request Format

TrackID: 8-bit field. In the context of this specification, the TrackID field signals the RPLInstanceID of the DODAG formed by the Track; see Sections 3.4 and 6.3. To allocate a new Track, the ingress Node must provide a value that is not in use at this time.

K: The 'K' flag is set to indicate that the recipient is expected to send a PDR-ACK back.

R: The 'R' flag is set to request a Complex Track for redundancy.

Flags: Reserved. The Flags field **MUST** be initialized to zero by the sender and **MUST** be ignored by the receiver.

ReqLifetime: 8-bit unsigned integer. The requested lifetime for the Track expressed in Lifetime Units (obtained from the DODAG Configuration option). The value of 255 (0xFF) represents infinity (never time out).

A P-DAO-REQ with a fresher PDRSequence refreshes the lifetime, and a ReqLifetime of 0 indicates that the Track **MUST** be destroyed, e.g., when the application that requested the Track terminates.

PDRSequence: 8-bit wrapping sequence number, obeying the operation in Section 7.2 of [RPL]. The PDRSequence is used to correlate a PDR-ACK message with the P-DAO-REQ message that triggered it. It is incremented at each P-DAO-REQ message and echoed in the PDR-ACK by the Root.

5.2. New PDR-ACK Control Message

The new PDR-ACK is sent as a response to a P-DAO-REQ message with the 'K' flag set. The RPL Control Code for the PDR-ACK is 0x0A. Its format is as follows:

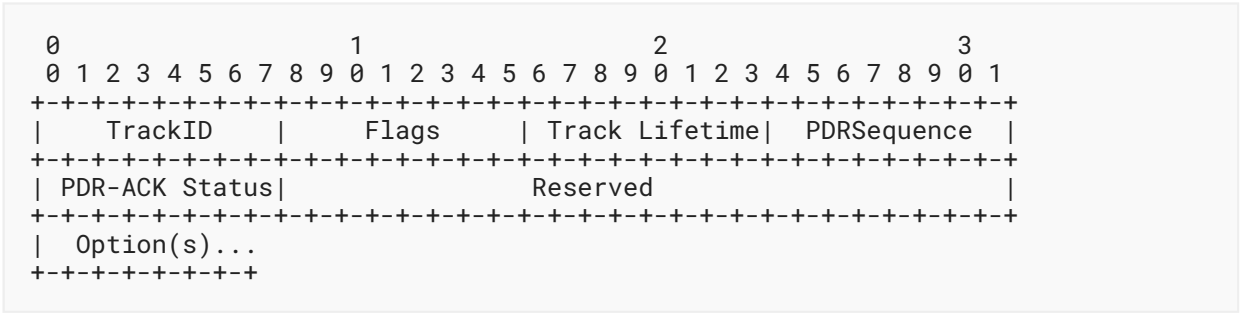

Figure 14: New PDR-ACK Control Message Format

TrackID: Set to the TrackID indicated in the TrackID field of the P-DAO-REQ messages that this replies to.

Flags: Reserved. The Flags field **MUST** be initialized to zero by the sender and **MUST** be ignored by the receiver.

Track Lifetime: Indicates the remaining lifetime for the Track, expressed in Lifetime Units. The value of 255 (0xFF) represents infinity. The value of zero (0x00) indicates that the Track was destroyed or not created.

PDRSequence: 8-bit wrapping sequence number. It is incremented at each P-DAO-REQ message and echoed in the PDR-ACK.

PDR-ACK Status: 8-bit field indicating the completion. The PDR-ACK Status is substructured as indicated in Figure 15:

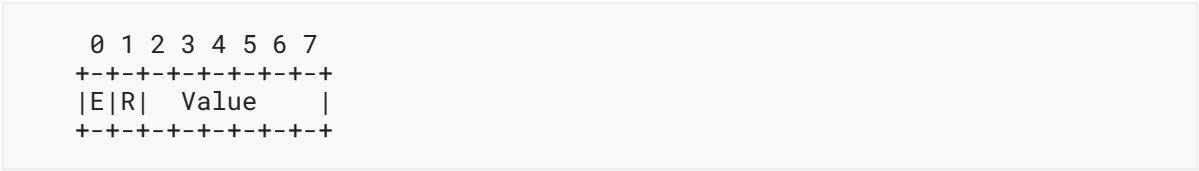

Figure 15: PDR-ACK Status Format

E: 1-bit flag. Set to indicate a rejection. When not set, a Value field that is set to 0 indicates Success/Unqualified Acceptance, and other values indicate "not an outright rejection".

R: 1-bit flag. Reserved; **MUST** be set to 0 by the sender and **MUST** be ignored by the receiver.

Status Value: 6-bit unsigned integer. Values depend on the setting of the 'E' flag; see Tables 28 and 29.

Reserved: The Reserved field **MUST** be initialized to zero by the sender and **MUST** be ignored by the receiver.

5.3. Via Information Options

A VIO signals the ordered list of IPv6 Via Addresses that constitutes the hops of either a protection path (using Non-Storing Mode) or a segment (using Storing Mode) of a Track. A Storing Mode P-DAO contains one SM-VIO whereas a Non-Storing Mode P-DAO contains one NSM-VIO.

The duration of the validity of a VIO is indicated in a Segment Lifetime field. A P-DAO message that contains a VIO with a Segment Lifetime of 0 is referred as a No-Path P-DAO.

The VIO contains one or more SRH-6LoRH headers, each formed of an SRH-6LoRH head and a collection of compressed Via Addresses, except in the case of a Non-Storing Mode No-Path P-DAO where the SRH-6LoRH header is not present.

In the case of an SM-VIO, or if [RFC8138] is not used in the data packets, then the Root **MUST** use only one SRH-6LoRH per Via Information Option, and the compression is the same for all the addresses, as shown in Figure 16, for simplicity.

In case of an NSM-VIO, and if [RFC8138] is in use in the main DODAG, the Root **SHOULD** optimize the size of the NSM-VIO if using different SRH-6LoRH Types would make the VIO globally shorter; this means that more than one SRH-6LoRH may be present.

The format of the VIO is as follows:

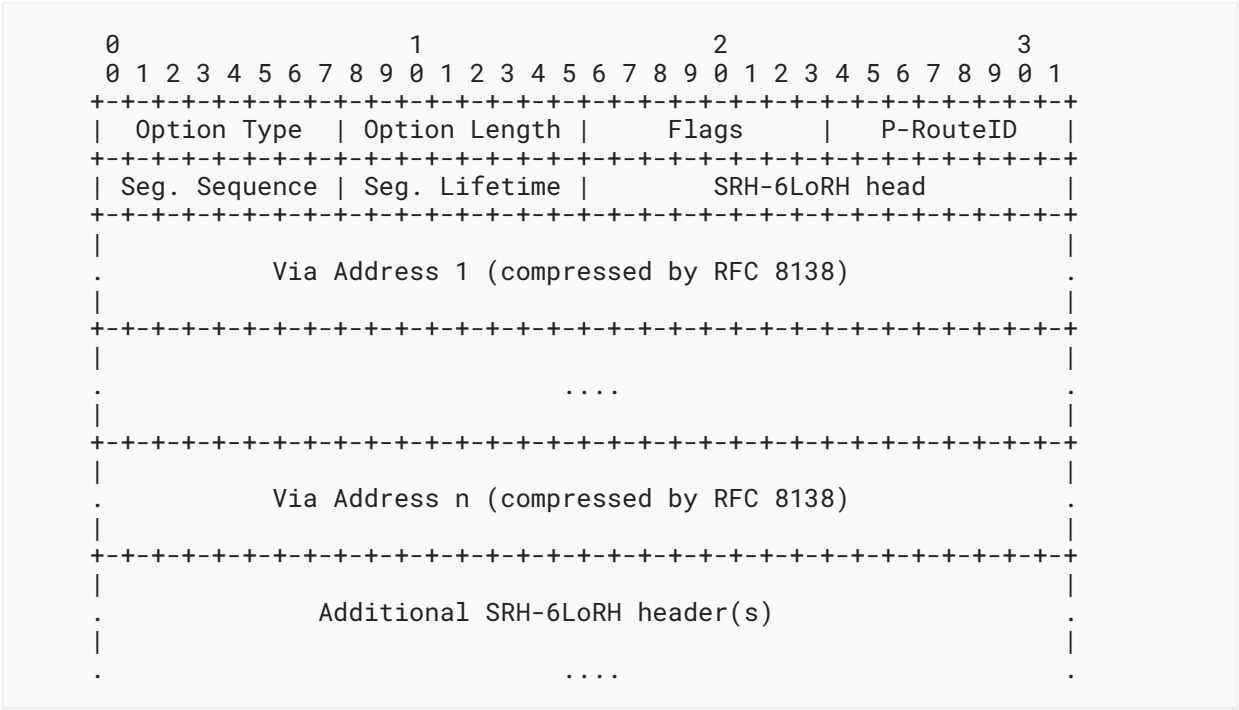

Figure 16: VIO Format

Option Type: 0x0F for SM-VIO and 0x10 for NSM-VIO (see Table 26).

Option Length: 8-bit unsigned integer, representing the length in octets of the option, not including the Option Type and Length fields (see Section 6.7.1 of [RPL]); the Option Length is variable, depending on the number of Via Addresses and the compression applied.

Flags: 8-bit field. No flag is defined in this specification. The field **MUST** be set to 0 by the sender and **MUST** be ignored by the receiver.

P-RouteID: 8-bit field that identifies a component of a Track or the main DODAG as indicated by the TrackID field. The value of 0 is used to signal a path, i.e., made of a single segment/protection path. In an SM-VIO, the P-RouteID indicates a SegmentID. In an NSM-VIO, it indicates the ID of a protection path that is added (or updated) to the overall topology of the Track.

Segment Sequence: 8-bit unsigned integer. The Segment Sequence obeys the operation in Section 7.2 of [RPL], and the initial value is 255.

When the Root of the DODAG needs to refresh or update a segment in a Track, it increments the Segment Sequence individually for that segment.

The segment information indicated in the VIO deprecates any state for the segment indicated by the P-RouteID within the indicated Track and provides the new information about the segment.

A VIO with a Segment Sequence that is not as fresh as the current one is ignored.

A VIO for a given DODAGID with the same (TrackID, P-RouteID, Segment Sequence) indicates a retry; it **MUST NOT** change the segment and **MUST** be propagated or answered as the first copy.

Segment Lifetime: 8-bit unsigned integer. The length of time in Lifetime Units (obtained from the Configuration option) that the segment is usable.

The period starts when a new Segment Sequence is seen. The value of 255 (0xFF) represents infinity. The value of zero (0x00) indicates a loss of reachability.

SRH-6LoRH head: The first 2 bytes of the (first) SRH-6LoRH as shown in Figure 6 of [RFC8138]. As an example, a 6LoRH of type 4 means that the Via Addresses are provided in full with no compression.

Via Address: An IPv6 ULA or GUA of a node along the segment. The VIO contains one or more IPv6 Via Addresses listed in the datapath order from ingress to egress. The list is expressed in a compressed form as signaled by the preceding SRH-6LoRH header.

In a Storing Mode P-DAO that updates or removes a section of an already existing segment, the list in the SM-VIO may represent only the section of the segment that is being updated; at the extreme, the SM-VIO updates only one node, in which case it contains only one IPv6 address. In all other cases, the list in the VIO **MUST** be complete.

In the case of an SM-VIO, the list indicates a sequential (strict) path through direct neighbors; the complete list starts at the ingress and ends at the egress, and the nodes listed in the VIO, including the egress, **MAY** be considered as implicit Targets.

In the case of an NSM-VIO, the complete list can be loose and excludes the ingress node, starting at the first loose hop and ending at a Track egress; the Track egress **MUST** be considered as an implicit Target, so it **MUST NOT** be signaled in a RPL Target Option.

5.4. Sibling Information Option

The Sibling Information Option (SIO) provides information about siblings that could be used by the Root to form P-Routes. One or more SIOs may be placed in the DAO messages that are sent to the Root in Non-Storing Mode.

To advertise a neighbor node, the router **MUST** have an active Address Registration from that sibling per [RFC8505] for an address (ULA or GUA) that serves as an identifier for the node. If this router also registers an address to that sibling, and the link has similar properties in both directions, only the router with the lowest Interface ID in its registered address needs to report the SIO, with the 'B' flag set, and the Root will assume symmetry.

The SIO carries a flag (B) that is set when similar performance can be expected in both directions; this flag indicates to the routing that the information provided for one direction is valid for both. If the SIO is effectively received from both sides, then the 'B' flag **MUST** be ignored. The policy that describes the performance criteria and how they are asserted is out of scope. In the absence of an external protocol to assert the link quality, the flag **SHOULD NOT** be set.

The format of the SIO is as follows:

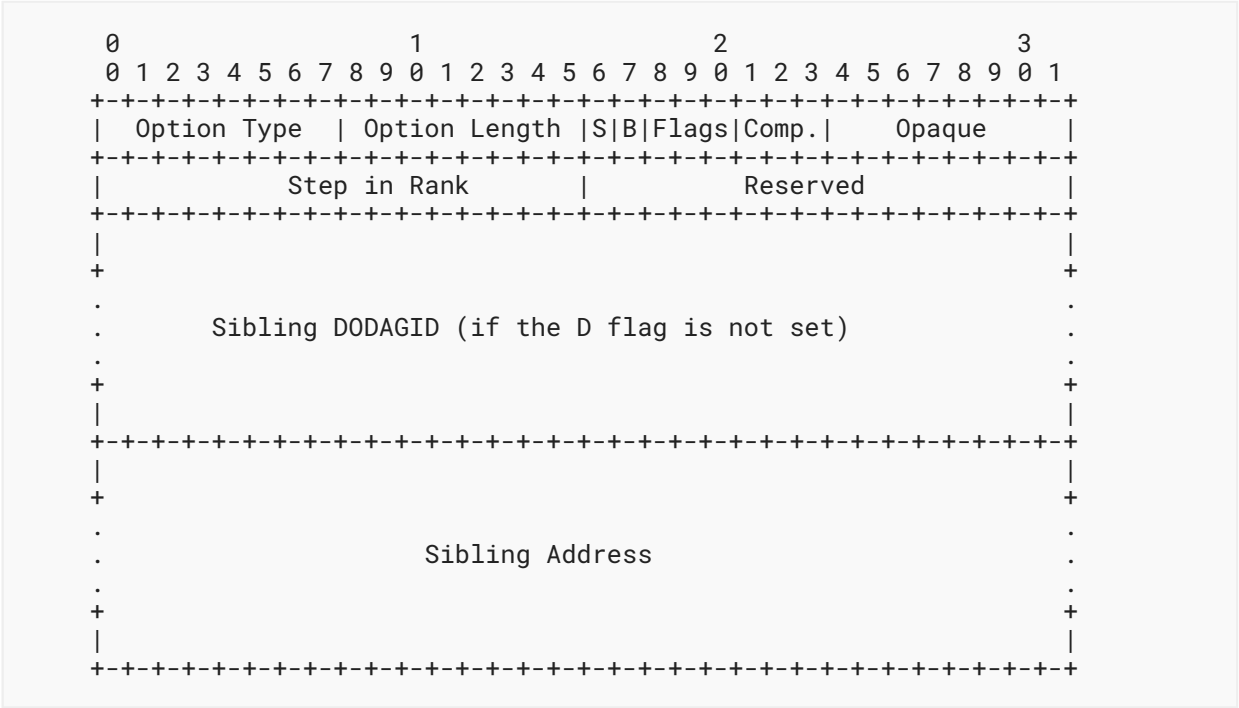

Figure 17: Sibling Information Option Format

Option Type: 0x11 for SIO (see Table 26).

Option Length: 8-bit unsigned integer, representing the length in octets of the option, not including the Option Type and Length fields (see Section 6.7.1 of [RPL]).

Reserved for Flags: **MUST** be set to 0 by the sender and **MUST** be ignored by the receiver.

B: 1-bit flag that is set to indicate that the connectivity to the sibling is bidirectional and roughly symmetrical. In that case, only one of the siblings needs to report the SIO for the hop. If 'B' is not set, then the SIO only indicates connectivity from the sibling to this node, and it does not provide information on the hop from this node to the sibling.

S: 1-bit flag that is set to indicate that the sibling belongs to the same DODAG. When not set, the Sibling DODAGID is indicated.

Flags: Reserved. The Flags field **MUST** be initialized to zero by the sender and **MUST** be ignored by the receiver.

Comp.: Compression Type; a 3-bit unsigned integer. This is the SRH-6LoRH Type as defined in Figure 7 in Section 5.1 of [RFC8138] that corresponds to the compression used for the Sibling Address and its DODAGID if present. The Compression reference is the Root of the main DODAG.

Opaque: **MAY** be used to carry information that the node and the Root understand, e.g., a particular representation of the link properties such as a proprietary Link Quality Information for packets received from the sibling. In some scenarios such as Industrial Alliances that use RPL for a particular use/environment, this field **MAY** be redefined to fit the needs of the case.

Step in Rank: 16-bit unsigned integer. This is the Step in Rank [RPL] as computed by the Objective Function between this node and the sibling, which reflects the abstract Rank increment that would be computed by the Objective Function if the sibling was the preferred parent.

Reserved: The Reserved field **MUST** be initialized to zero by the sender and **MUST** be ignored by the receiver

Sibling DODAGID: 2 to 16 bytes. The DODAGID of the sibling in a compressed form [RFC8138] as indicated by the Compression Type field. This field is present if and only if the 'D' flag is not set.

Sibling Address: 2 to 16 bytes. An IPv6 address of the sibling with a scope that **MUST** make it reachable from the Root, e.g., it cannot be a Link Local Address. The IPv6 address is encoded in the compressed form [RFC8138] indicated by the Compression Type field.

An SIO **MAY** be immediately followed by a DAG Metric Container. In that case, the DAG Metric Container provides additional metrics for the hop from the Sibling to this node.

6. Root-Initiated Routing State

6.1. RPL Network Setup

To avoid the need of Path MTU Discovery by 6LoWPAN endpoints, 6LoWPAN links are normally defined with an MTU of 1280 (see Section 4 of [6LoWPAN]). Injecting packets in a Track typically involves an IP-in-IP encapsulation and additional IPv6 extension headers. This may cause fragmentation if the resulting packets exceed the MTU that is defined for the RPL domain.

Though fragmentation is possible in a 6LoWPAN LLN, e.g., using [6LoWPAN], [RFC8930], and/or [RFC8931], it is **RECOMMENDED** to define an MTU that is larger than 1280 between the RPL routers that form the main DODAG to allow for the necessary header additions, while still exposing 1280 to the 6LoWPAN endpoint stacks.

6.2. Requesting a Track

This specification introduces the P-DAO-REQ message, which is used by an LLN node to request the formation of a new Track for which the LLN node is the ingress. Note that the namespace for the TrackID is owned by the ingress node, and in the absence of a P-DAO-REQ, there must be some procedure for the Root to assign TrackIDs in that namespace while avoiding collisions (see more in Section 6.3).

The P-DAO-REQ signals the desired TrackID and the duration for which the Track should be established. Upon a P-DAO-REQ, the Root **MAY** install the Track as requested, in which case it answers with a PDR-ACK indicating the granted Track Lifetime. All the segments **MUST** be of the same mode, either Storing or Non-Storing. All the segments **MUST** be created with the same TrackID and the same DODAGID signaled in the P-DAO.

The Root designs the Track as it sees fit and updates/changes the segments over time to serve the Track as needed. Note that there is no protocol element to notify the requesting Track ingress when changes happen deeper down the Track, so they are transparent to the Track ingress. If the main Root cannot maintain an expected service level, then it needs to tear down the Track completely. The Segment Lifetime in the P-DAO messages does not need to be aligned to the Requested Lifetime in the P-DAO-REQ or between P-DAO messages for different segments. For example, the Root may use shorter lifetimes for the segments and renew them or change them during the lifetime of the Track. All the components (protection paths and segments) of a Track **MUST** be destroyed (or have their lifetime elapsed) before the TrackID can be reused.

When the Track Lifetime is relatively close to elapse -- meaning in the order of the trip time from the node to the Root -- the requesting node **SHOULD** resend a P-DAO-REQ using the TrackID in the PDR-ACK to extend the lifetime of the Track; otherwise, the Track will time out, and the Root will tear down the whole structure.

If the Track fails and cannot be restored, the Root notifies the requesting node asynchronously with a PDR-ACK with a Track Lifetime of 0, indicating that the Track has failed, and a PDR-ACK Status, indicating the reason of the fault.

6.3. Identifying a Track

RPL defines the concept of an Instance to signal an individual routing topology, and multiple topologies can coexist in the same network. The RPLInstanceID is tagged in the RPI of every packet to signal which topology the packet actually follows.

This specification leverages the RPL Instance model as follows:

- The main Root **MAY** use P-DAO messages to add better routes in the main Instance in conformance with the routing objectives in that Instance.

To achieve this, the main Root **MAY** install a segment along a path down the main DODAG, which is operated in Non-Storing Mode. This enables loose source routing and reduces the size of the Routing Header; see Section 3.3.1. The main Root **MAY** also install a protection path across the main DODAG to complement the routing topology.

When adding a P-Route to the RPL main DODAG, the main Root **MUST** set the RPLInstanceID field of the P-DAO Base Object (see Section 6.4.1 of [RPL]) to the RPLInstanceID of the main DODAG, and it **MUST NOT** use the DODAGID field. A P-Route provides a longer match to the Target Address than the default route via the main Root, so it is preferred.

- The main Root **MAY** also use P-DAO messages to install a Track as an independent routing topology (say, Traffic Engineered) to achieve particular routing characteristics from ingress to egress endpoints. To achieve this, the main Root **MUST** set up a Local RPL Instance (see Section 5 of [RPL]), and the Local RPLInstanceID serves as the TrackID. The TrackID **MUST** be unique for the IPv6 ULA or GUA of the Track ingress that serves as the DODAGID for the Track.

  This way, a Track is uniquely identified by the tuple (DODAGID, TrackID) where the TrackID is always represented with the 'D' flag set to 0 (see also Section 5.1 of [RPL]), indicating that when used in an RPI, the source address of the IPv6 packet signals the DODAGID.

  The P-DAO Base Object **MUST** indicate the tuple (DODAGID, TrackID) that identifies the Track as shown in Figure 8, and the P-RouteID that identifies the P-Route **MUST** be signaled in the VIO as shown in Figure 16.

  The Track ingress is the Root of the DODAGID formed by the Local RPL Instance. It owns the namespace of its TrackIDs, so it can pick any unused value to request a new Track with a P-DAO-REQ. In a particular deployment where P-DAO-REQs are not used, a portion of the namespace can be administratively delegated to the main Root, meaning that the main Root is authoritative for assigning the TrackIDs for the Tracks it creates.

  With this specification, the main Root is aware of all the active Tracks, so it can also pick any unused value to form Tracks without a P-DAO-REQ. To avoid a collision of the main Root and the Track ingress picking the same value at the same time, it is **RECOMMENDED** that the Track ingress starts allocating the ID value of the Local RPLInstanceID (see Section 5.1 of [RPL]) used as TrackIDs with the value 0 incrementing, while the Root starts with 63 decrementing.

6.4. Installing a Track

A path can be installed by a single P-Route that signals the sequence of consecutive nodes either in Storing Mode as a single-segment Track or in Non-Storing Mode as a single-protection-path Track. A single-protection-path Track can be installed as a loose Non-Storing Mode P-Route, in which case the next loose entry must recursively be reached over a path.

A Complex Track can be installed as a collection of P-Routes with the same DODAGID and TrackID. The ingress of a Non-Storing Mode P-Route is the owner and Root of the DODAGID. The ingress of a Storing Mode P-Route must be either the owner of the DODAGID or a hop of a protection path of the same Track. In the latter case, the Targets of the P-Route must include the

next hop of the protection path if there is one to ensure forwarding continuity. In the case of a Complex Track, each segment is maintained independently and asynchronously by the Root, with its own lifetime that may be shorter, the same, or longer than that of the Track.

A route along a Track for which the TrackID is not the RPLInstanceID of the main DODAG **MUST** be installed with a higher precedence than the routes along the main DODAG, meaning that:

- The longest match **MUST** be the prime comparison for routing.
- For an equal-length match, the route along the Track **MUST** be preferred over the one along the main DODAG.
- There **SHOULD NOT** be two different Tracks leading to the same Target from same ingress node, unless there's a policy for selecting which packets use which Track; such a policy is out of scope.
- A packet that was routed along a Track **MUST NOT** be routed along the main DODAG again; if the destination is not reachable as a neighbor by the node where the packet exits the Track, then the packet **MUST** be dropped.

6.4.1. Signaling a Projected Route

This specification adds a capability whereby the Root of a main DODAG installs a Track as a collection of P-Routes, using a P-DAO message for each individual protection path or segment. The P-DAO signals a collection of Targets in one or more RTOs. Those Targets can be reached via a sequence of routers indicated in a VIO.

Like a classical DAO message, a P-DAO causes a change of state only if it is "new" per Section 9.2.2 ("Generation of DAO Messages") of the RPL specification [RPL]; this is determined using the Segment Sequence information from the VIO as opposed to the Path Sequence from a TIO. Also, a Segment Lifetime of 0 in a VIO indicates that the P-Route associated to the segment is to be removed. There are two Modes of operation for the P-Routes: Storing and Non-Storing.

A P-DAO message **MUST** be sent from the address of the Root that serves as the DODAGID for the main DODAG. It **MUST** contain either exactly one sequence of one or more RTOs followed by one VIO or any number of sequences of one or more RTOs followed by one or more TIOs. The former is the normal expression for this specification, whereas the latter corresponds to the variation for less-constrained environments described in Section 7.2.

A P-DAO that creates or updates a protection path **MUST** be sent to a GUA or a ULA of the ingress of the protection path; it **MUST** contain the full list of hops in the protection path unless the protection path is being removed. A P-DAO that creates a new Track segment **MUST** be sent to a GUA or a ULA of the segment egress and **MUST** signal the full list of hops in a segment; a P-DAO that updates (including deletes) a section of a segment **MUST** be sent to the first node after the modified segment and **MUST** signal the full list of hops in the section starting at the node that immediately precedes the modified section.

In Non-Storing Mode, as discussed in Section 6.4.3, the Root sends the P-DAO to the Track ingress where the source routing state is applied, whereas in Storing Mode, the P-DAO is sent to the last node on the installed path and forwarded in the reverse direction, installing a Storing Mode state at each hop, as discussed in Section 6.4.2. In both cases, the Track ingress is the owner of the Track, and it generates the P-DAO-ACK when the installation is successful.

If the 'K' flag is present in the P-DAO, the P-DAO **MUST** be acknowledged using a P-DAO-ACK that is sent back to the address of the Root from which the P-DAO was received. In most cases, the first node of the protection path, segment, or updated section of the segment is the node that sends the acknowledgment. The exception to the rule is when an intermediate node in a segment fails to forward a Storing Mode P-DAO to the previous node in the SM-VIO.

In a No-Path Non-Storing Mode P-DAO, the SRH-6LoRH **MUST NOT** be present in the NSM-VIO; the state in the ingress is erased regardless. In all other cases, a VIO **MUST** contain at least one Via Address, and a Via Address **MUST NOT** be present more than once, which would create a loop.

A node that processes a VIO **MAY** verify whether any of these conditions happen, and when one does, it **MUST** ignore the P-DAO and reject it with a RPL rejection status of "Error in VIO" in the DAO-ACK; see Section 11.16.

Errors, other than those discussed explicitly, that prevent the installation of the route are acknowledged with a RPL rejection status of "Unqualified Rejection" in the P-DAO-ACK.

6.4.2. Installing a Track Segment with a Storing Mode P-Route

As illustrated in Figure 18, a Storing Mode P-DAO installs a route along the segment signaled by the SM-VIO towards the Targets indicated in the Target Options. The segment is to be included in a DODAG indicated by the P-DAO Base Object, which may be the one formed by the main DODAG, or a Track associated with a Local RPL Instance.

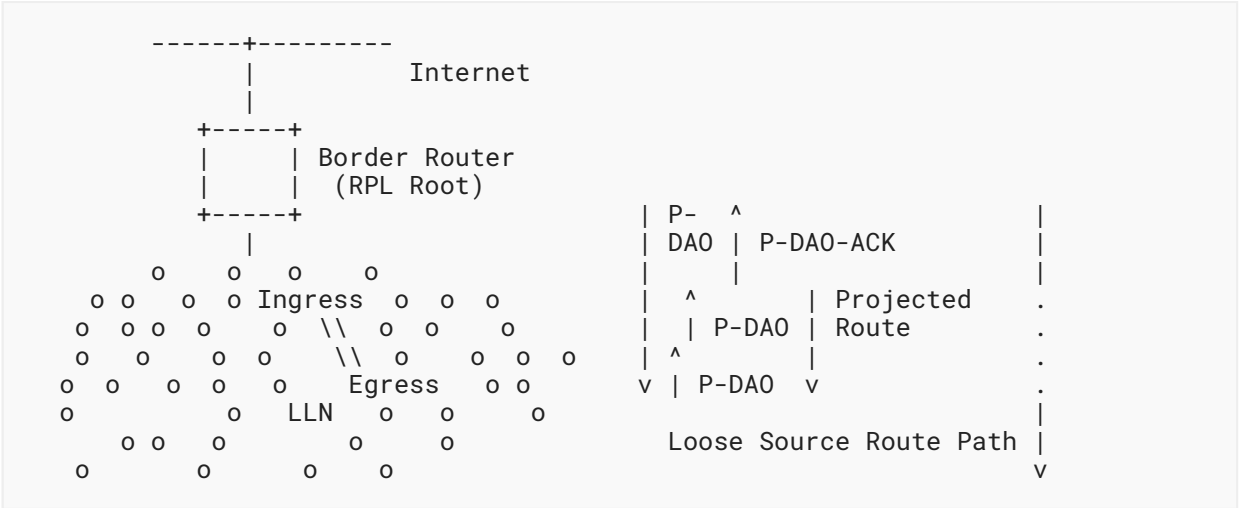

Figure 18: Projecting a Route

In order to install the relevant routing state along the segment, the Root sends a unicast P-DAO message to the Track egress router of the routing segment that is being installed. The P-DAO message contains an SM-VIO with a strict sequence of Via Addresses. The SM-VIO follows one or more RTOs indicating the Targets to which the Track leads. The SM-VIO contains a Segment Lifetime for which the state is to be maintained.

The Root sends the P-DAO directly to the egress node of the segment. In that P-DAO, the destination IP address matches the last Via Address in the SM-VIO. This is how the egress recognizes its role. In a similar fashion, the segment ingress node recognizes its role because it matches the first Via Address in the SM-VIO.

The egress node of the segment is the only node in the path that does not install a route in response to the P-DAO; it is expected to be already able to route to the Target(s) based on its existing tables. If one of the Targets is not known, the node **MUST** answer to the Root with a P-DAO-ACK listing the unreachable Target(s) in an RTO and a rejection status of "Unreachable Target".

If the egress node can reach all the Targets, it forwards the P-DAO with unchanged content to its predecessor in the segment as indicated in the list of VIOs, and the message is recursively propagated unchanged along the sequence of routers indicated in the P-DAO, but in the reverse order, from egress to ingress.

The address of the predecessor to be used as the destination of the propagated DAO message is found in the Via Address list at the position preceding the one that contains the address of the propagating node, which is used as the source of the message.

Upon receiving a propagated DAO, all except the egress router **MUST** install a route towards the DAO Target(s) via their successor in the SM-VIO. A router that cannot store the routes to all the Targets in a P-DAO **MUST** reject the P-DAO by sending a P-DAO-ACK to the Root with a rejection status of "Out of Resources" as opposed to forwarding the DAO to its predecessor in the list. The router **MAY** install additional routes towards the Via Addresses that appear in the SM-VIO after its own address, if any, but in case of a conflict or a lack of resource, the route(s) to the Target(s) **MUST** be installed in priority.

If a router cannot reach its predecessor in the SM-VIO, the router **MUST** send the P-DAO-ACK to the Root with a rejection status of "Predecessor Unreachable".

The process continues until the P-DAO is propagated to the ingress router of the segment, which answers with a P-DAO-ACK to the Root. The Root always expects a P-DAO-ACK, either from the Track ingress with a positive status or from any node along the segment with a negative status. If the P-DAO-ACK is not received, the Root may retry the DAO with the same TrackID or tear down the route.

6.4.3. Installing a Protection Path with a Non-Storing Mode P-Route

As illustrated in Figure 19, a Non-Storing Mode P-DAO installs a source-routed path within the Track indicated by the P-DAO Base Object towards the Targets indicated in the Target Options. The source-routed path requires a Source Routing Header, which implies an IP-in-IP

encapsulation is needed to add the SRH to an existing packet. It is sent to the Track ingress, which creates a tunnel associated with the Track and connected routes over the tunnel to the Targets in the RTO. The tunnel encapsulation **MUST** incorporate a Routing Header via the list addresses listed in the VIO in the same order. The content of the NSM-VIO starting at the first SRH-6LoRH header **MUST** be used verbatim by the Track ingress when it encapsulates a packet to forward it over the Track.

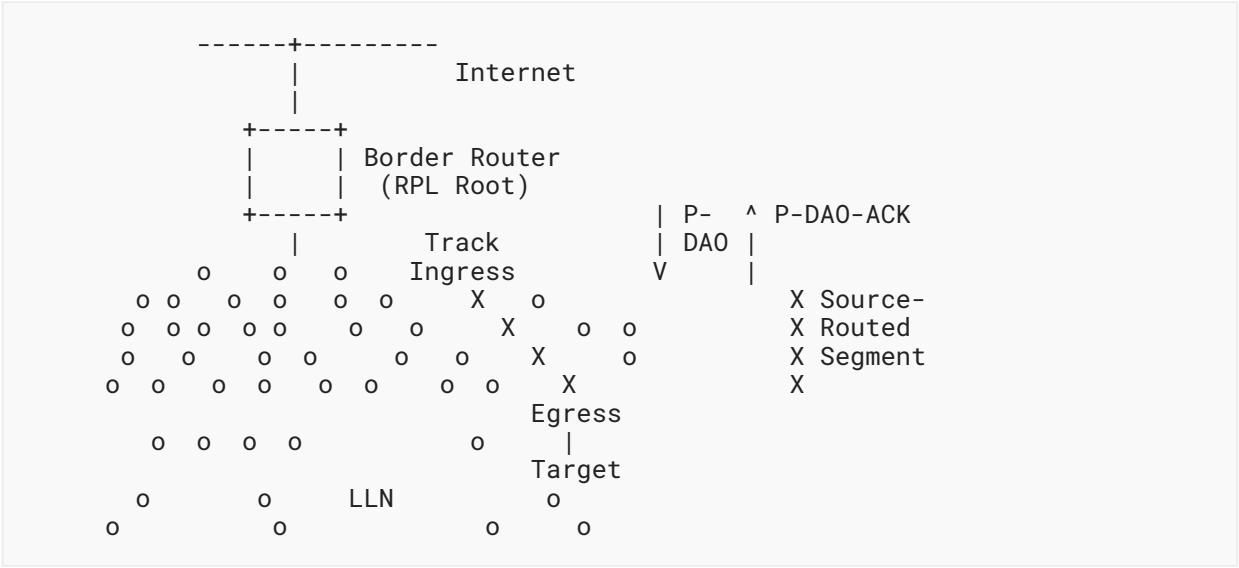

Figure 19: Projecting a Non-Storing Route

The next entry in the source-routed path must be either a neighbor of the previous entry or reachable as a Target via another P-Route, either Storing or Non-Storing, which implies that the nested P-Route has to be installed before the loose sequence is and that P-Routes must be installed from the last to the first along the datapath. For instance, a segment of a Track must be installed before the protection path(s) of the same Track that uses it, and stitched segments must be installed in order from the last to the first to reach the Targets.

If the next entry in the loose sequence is reachable over a Storing Mode P-Route, it **MUST** be the Target of a segment and the ingress of a next segment, which are both already set up; the segments are associated with the same Track, which avoids needing an additional encapsulation. For instance, in Section 3.5.1.3, segments A==>B-to-C and C==>D==>E-to-F must be installed with Storing Mode P-DAO messages 1 and 2 before the Track A-->C-->E-to-F that joins them can be installed with Non-Storing Mode P-DAO 3.

Conversely, if it is reachable over a Non-Storing Mode P-Route, the next loose source-routed hop of the inner Track is a Target of a previously installed Track and the ingress of a next Track, which requires de- and re-encapsulation when switching the outer Tracks that join the loose hops. This is exemplified in Section 3.5.2.3 where Non-Storing Mode P-DAOs 1 and 2 install strict Tracks that Non-Storing Mode P-DAO 3 joins as a super Track. In such a case, packets are subject to double IP-in-IP encapsulation.

6.5. Tearing Down a P-Route

A P-DAO with a lifetime of 0 is interpreted as a No-Path DAO. Its function is to clean up an existing state as opposed to refreshing it or installing a new one. To tear down a Track, the Root must tear down all the Track segments and protection paths that compose it one by one.

Since the protection path state of a Track is located only on the ingress Node, the Root cleans up the protection path by sending an NSM-VIO to the ingress to indicate the TrackID and the P-RouteID of the protection path being removed, a Segment Lifetime of 0, and a newer Segment Sequence. The SRH-6LoRH with Via Addresses in the NSM-VIO is not needed; it **SHOULD NOT** be placed in the message and **MUST** be ignored by the receiver. Upon that NSM-VIO, the ingress node removes all state for that Track, if any, and replies positively anyway.

The Root cleans up a section of a segment by sending an SM-VIO to the last node of the segment with an updated TrackID and P-RouteID, a Segment Lifetime of 0, and a newer Segment Sequence. The Via Addresses in the SM-VIO indicate the section of the segment being modified, from the first to the last node that is impacted. This can be the whole segment if it is totally removed or a sequence of one or more nodes that have been bypassed by a segment update.

The No-Path P-DAO is forwarded normally along the reverse list, even if the intermediate node does not find a segment state to clean up. This results in cleaning up the existing segment state, if any, as opposed to refreshing an existing one or installing a new one.

6.6. Maintaining a Track

Repathing a Track segment or protection path may cause jitter and packet misordering. For critical flows that require timely and/or in-order delivery, it might be necessary to deploy PREOF [RAW-ARCH] over a highly redundant Track. This specification allows the use of more than one protection path for a Track and 1+N packet redundancy.

This section provides the steps to ensure that no packet is lost due to the operation itself. This is ensured by installing the new section from its last node to the first, so when an intermediate node installs a route along the new section, all the downstream nodes in the section have already installed their own. The disabled section is removed as well when the in-flight packets are forwarded along the new section.

6.6.1. Maintaining a Track Segment

To modify a section of a segment between the first node and a second downstream node (which can be the ingress and egress, respectively) while retaining those nodes in the segment, the Root sends an SM-VIO to the second node indicating the sequence of nodes in the new section of the segment. The SM-VIO indicates the TrackID and the P-RouteID of the segment being updated and a newer Segment Sequence. The P-DAO is propagated from the second to the first node, and on the way, it updates the state on the nodes that are common to the old and new section of the segment and creates a state in the new nodes.

When the state is updated in an intermediate node, that node might still receive packets that were in flight from the ingress to self over the old section of the segment. Since the remainder of the segment is already updated, the packets are forwarded along the new version of the segment from that node on.

After a reasonable amount of time, the Root tears down the remaining section(s) of the old segments as described in Section 6.5 to enable the deprecated sections to drain their traffic.

6.6.2. Maintaining a Protection Path

This specification allows the Root to add protection paths to a Track by sending a Non-Storing Mode P-DAO to the ingress associated to the same TrackID and a new SegmentID. If the protection path is loose, then the segments that join the hops must be created first. It makes sense to add a new protection path before removing one that is becoming excessively lossy and switch to the new protection path before removing the old. Dropping a Track before the new one is installed would reroute the traffic via the Root; this may increase the latency beyond acceptable thresholds and overload the network near the Root. This may also cause loops in the case of stitched Tracks: The packets that cannot be injected in the second Track might be routed back and reinjected at the ingress of the first Track.

It is also possible to update a protection path by sending a Non-Storing Mode P-DAO to the ingress with the same SegmentID, an incremented Segment Sequence, and the new complete list of hops in the NSM-VIO. Updating a live protection path means changing one or more of the intermediate loose hops, and it involves laying out new segments from and to the new loose hops before the NSM-VIO is issued for the new protection path.

Packets that are in flight over the old version of the protection path still follow the old source route path over the old segments. After a reasonable time, the Root tears down those segments as described in Section 6.5 to enable the deprecated segments to drain their traffic.

6.7. Encapsulating and Forwarding Along a Track

When injecting a packet in a Track, the ingress router must encapsulate the packet using IP-in-IP to add the Source Routing Header with the final destination set to the Track egress.

All properties of a Track's operations are inherited from the main Instance that is used to install the Track. For instance, the use of compression per [RFC8138] is determined by whether it is used in the RPL main DODAG, e.g., by setting the 'T' flag [RFC9035] in the RPL Configuration option.

When the Track ingress places a packet in a Track, it encapsulates it with an additional IPv6 header, a Routing Header, and an IPv6 Hop-by-Hop Option Header that contains the RPI as follows:

- In the uncompressed form, the source of the packet is the address that this router uses as the DODAGID for the Track, the destination is the first Via Address in the NSM-VIO, and the RH is an SRH [RFC6554] that contains the list of the remaining Via Addresses, ending with the Track egress.

- In a network where 6LoWPAN header compression [RFC6282] is in use, it is preferred to implement "IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Paging Dispatch" [RFC8025] and compress the RPL artifacts as indicated in [RFC8138].

  In that case, the RPL Source Route Header is the exact copy of the (chain of) SRH-6LoRH found in the NSM-VIO, also ending with the Track egress. The RPI-6LoRH is appended next, followed by an IP-in-IP 6LoRH header that indicates the ingress router in the Encapsulator Address field; see a similar case in Figure 20 of [RFC8138].

To signal the Track in the packet, this specification leverages the RPL forwarding model as follows:

- In the data packets, the Track DODAGID and the TrackID **MUST** be respectively signaled as the IPv6 source address, and the RPLInstanceID field of the RPI **MUST** be placed in the outer chain of IPv6 headers.

  The RPI carries a Local RPLInstanceID called the TrackID, which, in association with the DODAGID, indicates the Track along which the packet is forwarded.

  The 'D' flag in the RPLInstanceID **MUST** be set to 0 to indicate that the source address in the IPv6 header is set to the DODAGID (see more in Section 6.3).

- This specification conforms to the principles of [RFC9008] with regard to packet forwarding and encapsulation along a Track as follows:

  - With this specification, the Track is a RPL DODAG. From the perspective of that DODAG, the Track ingress is the Root, the Track egress is a RPL-Aware 6LR, and neighbors of the Track egress that can be reached via the Track, but are external to it, are external destinations and treated as RPL-Unaware Leaves (RULs). The encapsulation rules in [RFC9008] apply.
  - If the Track ingress is the originator of the packet and the Track egress is the destination of the packet, there is no need for an encapsulation.
  - Thus, the Track ingress must encapsulate the traffic that it did not originate, and it must include an RPI in the encapsulation to signal the TrackID.

  A packet that is being routed over the RPL Instance associated to a first Non-Storing Mode Track **MAY** be placed recursively in a second Track to cover one loose hop of the first Track, as discussed in more detail in Section 3.5.2.3. On the other hand, a Storing Mode segment must be strict, and a packet that it placed in a Storing Mode segment **MUST** follow that segment till the segment egress.

It is known that a packet is forwarded along a Track by the source address and the RPI in the encapsulation. The TrackID is used to identify the RIB entries associated to that Track, which, in intermediate nodes, correspond to the P-Routes for the segments of the Track that the forwarding router is aware of. Packet processing uses the following precedence: 1) self-delivery or Routing Header handling when one is present, 2) delivery to direct neighbors, 3) delivery to indirect neighbors, 4) routing along a segment along the Track, and 5) injecting the packet in another Track, as a last resort.

To achieve this, the packet handling logic **MUST** happen in the following order:

- If the destination of the packet is self:
  1. If the header chain contains a RPL Source Route Header that is not fully consumed, then the packet is forwarded along the Track as prescribed by [RFC6554], meaning that the next entry in the Routing Header becomes the destination.
  2. Otherwise, if the packet was encapsulated, then the packet is decapsulated and the forwarding process recurses; else, the packet is delivered to the stack.
- Otherwise, the packet is forwarded as follows:
  1. If the destination of the packet is a direct neighbor, e.g., installed by IPv6 Neighbor Discovery, then the packet **MUST** be forwarded to that neighbor.
  2. Else, if the destination of the packet is an indirect neighbor, e.g., installed by a multicast DAO message from a common neighbor (see Section 4.1.4), then the packet **MUST** be forwarded to the common neighbor.
  3. Else, if there is a RIB entry for the same Track (e.g., installed by an SM-VIO in a DAO message with the destination as the target) and the next hop in the RIB entry is a direct neighbor, then the packet is passed to that neighbor.
  4. Else, if there is a RIB entry for the different Track (e.g., installed by an NSM-VIO in a DAO message with the destination as the target), then the packet is encapsulated to be forwarded along that Track and the forwarding process recurses; otherwise, the packet is dropped.
  5. To avoid loops, and as opposed to packets that were not encapsulated, a packet that was decapsulated from a Track **MUST NOT** be routed along the default route of the main DODAG; this would mean that the end-to-end path is uncontrolled. The node that discovers the fault **MUST** discard the packet.

The node that drops a packet in any of the steps above **MUST** send an ICMPv6 error message [RFC4443] to the Root, with a new code "Error in P-Route" (see Section 11.15). The Root can then repair by updating the broken segment and/or Tracks. In the case of a broken segment, the Root removes the leftover sections of the segment using SM-VIOs with a lifetime of 0, indicating the section where one or more nodes are being removed (see Section 6.6).

In case of a permanent forwarding error along a source route path, the node that fails to forward **SHOULD** send an ICMP error with the code "Error in Source Routing Header" back to the source of the packet, as described in Section 11.2.2.3 of [RPL]. Upon receiving this message, the encapsulating node **SHOULD** stop using the source route path for a reasonable period of time, which depends on the deployment, and it **SHOULD** send an ICMP message with the code "Error in P-Route" to the Root. Failure to follow these steps may result in packet loss and wasted resources along the source route path that is broken.

Either way, the ICMP message **MUST** be throttled in case of consecutive occurrences. It **MUST** be sourced at the ULA or GUA that is used in this Track for the source node, so the Root can establish where the error happened.

The portion of the invoking packet that is sent back in the ICMP message **SHOULD** record at least up to the RH if one is present, and the hop of the RH **SHOULD** be consumed by this node so that the destination in the IPv6 header is the next hop that this node could not reach. If a 6LoRH [RFC8138] is used to carry the IPv6 routing information in the outer header, then the whole 6LoRH information **SHOULD** be present in the ICMP message.

6.8. Compression of RPL Artifacts

When the main DODAG in a 6LoWPAN LLN is operated in Non-Storing Mode and the data packets are compressed using [RFC8138], a typical packet that circulates in the main DODAG is formatted as shown in Figure 20, representing the case where an IP-in-IP encapsulation is needed (see Table 19 of [RFC9008]):

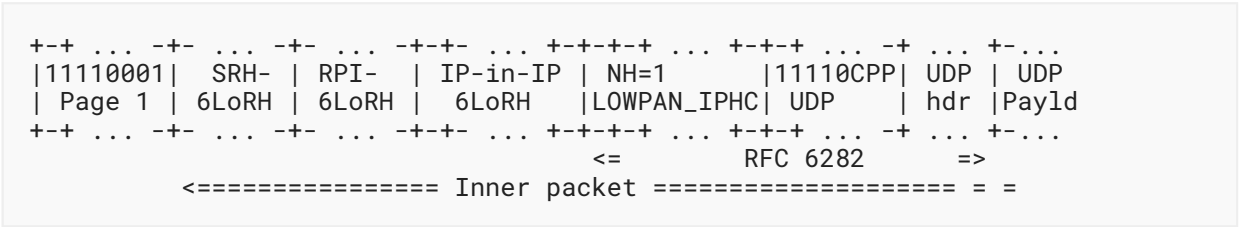

Figure 20: A Packet as Forwarded Along the Main DODAG

Since there is no page switch between the encapsulated packet and the encapsulation, the first octet of the compressed packet that acts as the page selector is actually removed at encapsulation; therefore, the inner packet used in the descriptions below starts with the SRH-6LoRH and is exactly the packet represented in Figure 20, from the second octet onward.

When encapsulating the inner packet to place in the Track, the first header that the ingress appends at the head of the inner packet is an IP-in-IP 6LoRH header; in that header, the encapsulator address, which maps to the IPv6 source address in the uncompressed form, contains a GUA or ULA IPv6 address of the ingress node that serves as the DODAGID for the Track, expressed in a compressed form using the DODAGID of the main DODAG as a reference for the compression. If the address is compressed to 2 bytes, the resulting value for the Length field (shown in Figure 21) is 3, meaning that the SRH-6LoRH as a whole is 5 octets long.

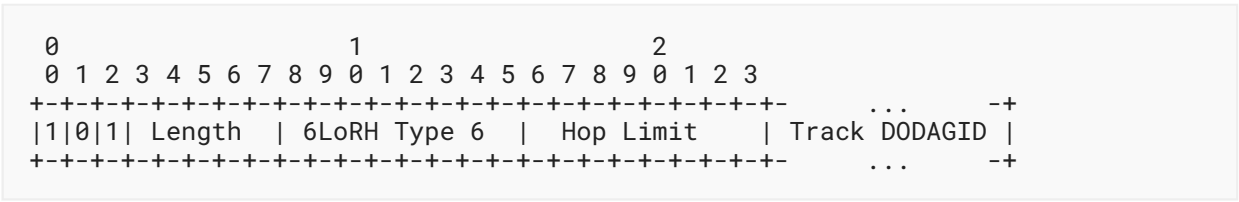

Figure 21: The IP-in-IP 6LoRH Header

At the head of the resulting sequence of bytes, the Track ingress then adds a P-RPI-6LoRH header to transport the RPI in its compressed form as illustrated in Figure 12. The RPI carries the TrackID as RPLInstanceID. Combined with the IP-in-IP 6LoRH header, this allows identifying the Track without ambiguity.

The SRH-6LoRH is then added at the head of the resulting sequence of bytes as a verbatim copy of the content of the SM-VIO that signaled the selected protection path.

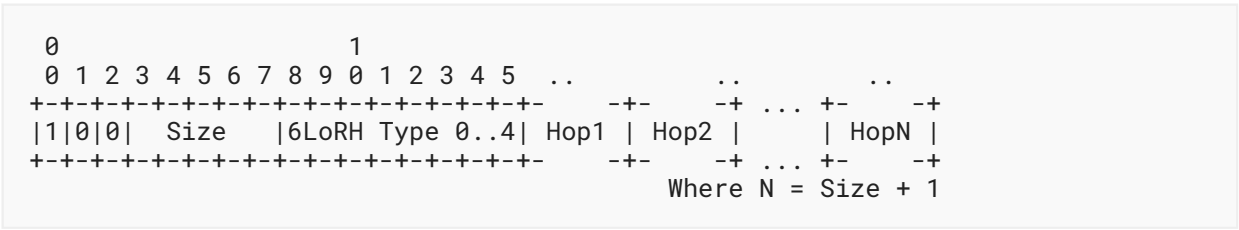

Figure 22: The SRH-6LoRH Header

The format of the resulting encapsulated packet, which is in compressed form per [RFC8138], is illustrated in Figure 23:

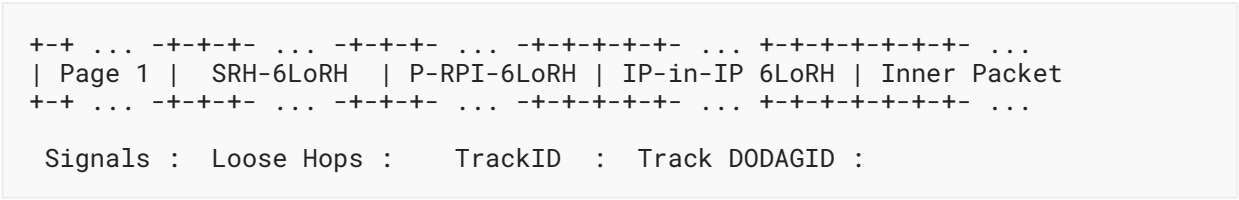

Figure 23: A Packet as Forwarded Along a Track

7. Less-Constrained Variations

7.1. Storing Mode Main DODAG

This specification expects that the main DODAG is operated in Non-Storing Mode. The reasons for that limitation are mostly related to LLN operations, power, and spectrum conservation:

- In Non-Storing Mode, the Root already knows the DODAG topology, so the additional topological information is reduced to the siblings.
- The downward routes are updated with unicast messages to the Root, which ensures that the Root can reach back to the LLN nodes after a repair faster than in the case of Storing Mode. Also, the Root can control the use of path diversity in the DODAG to reach the LLN nodes. For both reasons, Non-Storing Mode provides better capabilities for the Root to maintain the P-Routes.
- When the main DODAG is operated in Non-Storing Mode, P-Routes enable loose source routing, which is only an advantage in that mode. Storing Mode does not use Source Routing Headers and does not derive the same benefits from this capability.

On the other hand, since RPL is a Layer 3 routing protocol, its applicability extends beyond LLNs to a generic IP network. RPL requires fewer resources than alternative IGPs such as OSPF, IS-IS, the Enhanced Interior Gateway Routing Protocol (EIGRP), BABEL, or RIP when using routing stretch rather than the shortest path routes to a destination that those protocols compute. P-Routes add the capability to install the shortest and/or constrained routes to special destinations as discussed in Appendix A.9.4 of "An Autonomic Control Plane (ACP)" [RFC8994].

In a powered and wired network, when enough memory to store the needed routes is available, the RPL Storing Mode proposes a better trade-off than the Non-Storing Mode, as it reduces the routing stretch and lowers the load on the Root. In that case, the control path between the Root and the RPL nodes can be maintained more aggressively and with more redundancy than in LLNs, and the nodes can be reached to maintain the P-Routes at most times for a finer and more reactive control.

This section specifies the additions that are needed to support P-Routes when the main DODAG is operated in Storing Mode. As long as the RPI can be processed adequately by the data plane, the changes in this specification are limited to the DAO message. The Track structure, routes, and forwarding operations remain the same. Since there is no capability negotiation, the expectation is that all the nodes in the network support this specification in the same fashion or are configured the same way through management.

In Storing Mode, the Root misses the Child-to-Parent relationship that forms the main DODAG as well as the sibling information. To provide that knowledge, the nodes in the network **MUST** send additional DAO messages that are unicast to the Root just like Non-Storing DAO messages are.

In the DAO message, the originating router advertises a set of neighbor nodes using SIOs, regardless of the relative position in the DODAG of the advertised node versus this router.

The DAO message **MUST** be formed as follows:

- The originating router is identified by the source address of the DAO. That address **MUST** be the one that this router registers to the neighbor routers so the Root can correlate the DAOs from those routers when they advertise this router as their neighbor. The DAO contains one or more sequences of one TIO and one or more SIOs. There is no RPL Target Option so that the Root is not confused into adding a Storing Mode route to the Target.
- The TIO is formed as in Storing Mode, and the Parent Address is not present. The Path Sequence and Path Lifetime fields are aligned with the values used in the Address Registration of the node(s) advertised in the SIO, as explained in Section 9.1 of [RFC9010]. Having similar values in all nodes allows factorizing the TIO for multiple SIOs as done in [RPL].
- The TIO is followed by one or more SIOs that provide an address (ULA or GUA) of the advertised neighbor node.

However, the RPL routing information headers may not be supported on all types of routed network infrastructures, especially not in high-speed routers. When the RPI is not supported in the data plane, there cannot be Local RPL Instances and RPL can only operate as a single topology (the main DODAG). The RPL Instance is the one that runs the main DODAG, and the

ingress node that encapsulates the RPL Instance is not the Root. The routes along the Tracks are alternate routes to those available along the main DODAG. They **MAY** conflict with routes to children and **MUST** take precedence in the routing table. The Targets **MUST** be adjacent to the Track egress to avoid loops that may form if the packet is reinjected in the main DODAG.

7.2. A Track as a Full DODAG

This specification builds Tracks with parallel or interleaved protection paths as opposed to a more complex DODAG with interconnections at any place desirable. The reason for that limitation is related to constrained node operations and the capability to store a large amount of topological information and compute complex paths:

- With this specification, the node in the LLN has no topological awareness and does not need to maintain dynamic information about the link quality and availability.
- The Root has complete topological information and statistical metrics that allow it, or its PCE, to perform a global optimization of all Tracks in its DODAG. Based on that information, the Root computes the protection path and produces the source route paths.
- The node merely selects one of the proposed paths and applies the associated pre-computed Routing Header in the encapsulation. This alleviates both the complexity of computing a path and the compressed form of the Routing Header.

The RAW architecture [RAW-ARCH] actually expects the PLR at the Track ingress to react to changes in the forwarding conditions along the Track and reroute packets to maintain the required degree of reliability. To achieve this, the PLR needs the full richness of a DODAG to form any path that could meet the SLO.

This section specifies the additions that are needed to turn the Track into a full DODAG and enable the main Root to provide the necessary topological information to the Track ingress. The expectation is that the PLR's metrics will be in a different order than the PCE's metrics because of the difference in the timescale between routing and forwarding; see more in [RAW-ARCH]. It follows that the PLR will learn the metrics it needs from an alternate source, e.g., OAM frames.

To pass the topological information to the ingress, the Root uses a P-DAO message that contains sequences of Targets and TIOs that collectively represent the Track, expressed in the same fashion as in classical Non-Storing Mode. The difference is that the Root is the source as opposed to the destination, and the Root can report information on many Targets, possibly the full Track, with one P-DAO.

Note that the Path Sequence and Lifetime in the TIO are selected by the Root and that the Target/Transit information tuples in the P-DAO are not those received by the Root in the DAO messages about the said Targets. The Track may follow sibling routes and does not need to be congruent with the main DODAG.

8. Profiles

This document provides a set of tools that may or may not be needed by an implementation depending on the type of application it serves. This section describes profiles that can be implemented separately, e.g., using only a portion of this specification to meet a particular use case, and can be used to discriminate what an implementation can and cannot do.

Profiles 0 to 2 operate in the main Instance and do not require the support of Local RPL Instances or the indication of the RPL Instance in the data plane. Profile 3 and above leverage Local RPL Instances to build arbitrary Tracks rooted at the Track ingress, using the namespace of the Track ingress for the TrackID.

Profiles 0 and 1 are **REQUIRED** by all implementations that may be used in LLNs; Profile 1 leverages Storing Mode to reduce the size of the RPL Source Route Header in the most common LLN deployments. Profile 2 is **RECOMMENDED** in a high-speed (e.g., wired) environment to enable Traffic Engineering and network automation. All the other profile/environment combinations are **OPTIONAL**.

Profile 0:
: Profile 0 is the legacy support of [RPL] Non-Storing Mode, with default routing Northwards (up) and strict source routing Southwards (down the main DODAG). It provides the minimal common functionality that must be implemented as a prerequisite to all the Track-supporting profiles. The other profiles extend Profile 0 with selected capabilities that this specification introduces.

Profile 1 (Storing Mode P-Route segments along the main DODAG):
: Profile 1 does not create new paths; compared to Profile 0, it combines Storing and Non-Storing Modes to balance the size of the Routing Header in the packet and the amount of state in the intermediate routers in a Non-Storing Mode RPL DODAG.

Profile 2 (Non-Storing Mode P-Route segments along the main DODAG):
: Profile 2 extends Profile 0 with strict source-routed Non-Storing Mode P-Routes along the main DODAG, which is the same as Profile 1 but using NSM-VIOs as opposed to SM-VIOs. Profile 2 provides the same capability to compress the SRH in packets down the main DODAG as Profile 1, but it requires an encapsulation in order to insert an additional SRH between the loose source routing hops. With Profile 2, the Tracks **MUST** be installed as subTracks of the main DODAG, and the main Instance **MUST** be used as the TrackID. Note that the ingress node encapsulates but is not the Root, as it does not own the DODAGID.

Profile 3:
: In order to form the best path possible, this profile requires the support of an SIO to inform the Root of additional possible hops. Profile 3 extends Profile 1 with additional Storing Mode P-Routes that install segments that do not follow the main DODAG. If the segment ingress (in

the SM-VIO) is the same as the IPv6 address of the Track ingress (in the Projected DAO Base Object), the P-DAO creates an implicit Track between the segment ingress and the segment egress.

Profile 4:
: Profile 4 extends Profile 2 with strict source-routed Non-Storing Mode P-Routes to form forward Tracks that are inside the main DODAG but do not necessarily follow it. A Track is formed as one or more strict source-routed paths between the Root that is the Track ingress and the Track egress that is the last node.

Profile 5:
: Profile 5 combines Profile 4 with Profile 1 and enables loose source routing between the ingress and the egress of the Track. As in Profile 1, Storing Mode P-Routes form the connections in the loose source route.

Profile 6:
: Profile 6 combines Profile 4 with Profile 2 and enables loose source routing between the ingress and the egress of the Track.

Profile 7:
: Profile 7 implements Profile 5 in a main DODAG that is operated in Storing Mode as presented in Section 7.1. As in Profiles 1 and 2, the TrackID is the RPLInstanceID of the main DODAG. Longest match rules decide whether a packet is sent along the main DODAG or rerouted in a Track.

Profile 8:
: Profile 8 is offered in preparation of the RAW work and for use cases where an arbitrary node in the network can afford the same code complexity as the RPL Root in a traditional deployment. It offers a full DODAG visibility to the Track ingress, as specified in Section 7.2, in a Non-Storing Mode main DODAG.

Profile 9:
: Profile 9 combines Profiles 7 and 8, operating the Track as a full DODAG within a Storing Mode main DODAG, using only the main DODAG RPLInstanceID as the TrackID.

9. Backwards Compatibility

This specification can operate in a mixed network where some nodes support it and some do not. There are restrictions, though. All nodes that need to process a P-DAO **MUST** support this specification. As discussed in Section 3.7.1, how the Root knows the node capabilities and whether they support this specification are out of scope.

This specification defines the 'D' flag in the RPL DODAG Configuration option (see Section 4.1.7) to signal that the RPL nodes can request the creation of Tracks. The requester may not know whether the Track can effectively be constructed or whether enough nodes along the preferred paths support this specification. Therefore, it makes sense to only set the 'D' flags in the DIO when the conditions for success are in place, in particular when all the nodes that could be on the path of the Tracks are upgraded.

10. Security Considerations

It is worth noting that per [RPL], every node in the LLN is RPL-aware and can inject any RPL-based attack in the network. This specification uses messages that are already present in RPL [RPL] with optional secured versions. The same secured versions may be used with this specification, and whatever security is deployed for a given network also applies to the flows in this specification.

The LLN nodes depend on the RPL Root and RANs for their operation. A trust model is necessary to ensure that the right devices are acting in these roles, avoiding sinkhole attacks (as is done in Section 7 of [RFC7416]). This trust model could be, at a minimum, based on a Layer 2 secure joining and link-layer security. This is a generic 6LoWPAN requirement; see Req-5.1 in Appendix B.5 of [RFC8505].

In a general manner, the Security Considerations in [RPL] and [RFC7416] apply to this specification as well. In particular, link-layer security is needed to prevent denial-of-service attacks, whereby a rogue router creates a high churn in the RPL network by constantly injecting forged P-DAO messages and using up all the available storage in the attacked routers.

When applied to radio LLNs such as IEEE Std 802.15.4, the lower-layer frame protection can be leveraged with an appropriate join protocol. The 6TiSCH Constrained Join Protocol (CoJP) [RFC9031] uses the RPL Root as 6LBR. The join protocol could be extended to provide additional key material for pledges to 6LBR communication when additional end-to-end security is desired beyond the hop-by-hop security from the lower layer.

With this specification, the Root **MAY** generate P-DAO messages but other nodes **MUST NOT** do so. P-DAO-REQ messages **MUST** be sent to the Root. This specification expects that the communication with the Root is authenticated but does not enforce which method is used.

Additionally, the trust model could include a role validation (e.g., using a role-based authorization) to ensure that the node that claims to be a RPL Root is entitled to do so. That trust should propagate from egress to ingress in the case of a Storing Mode P-DAO.

This specification suggests some validation of the VIO to prevent basic loops, i.e., by avoiding a node that appears twice. But that is only a minimal protection. Arguably, an attacker that can inject P-DAOs can reroute any traffic and rapidly deplete critical resources such as the spectrum and battery in the LLN.

11. IANA Considerations

11.1. RPL DODAG Configuration Option Flag

IANA has assigned a flag in the "DODAG Configuration Option Flags for MOP 0..6" registry [RFC9008] under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL] as follows:

Bit Number	Capability Description	Reference
0	Projected Routes Support (D)	RFC 9914

Table 21: New DODAG Configuration Option Flag

IANA has added this RFC as an additional reference for MOP 7 in the "Mode of Operation" registry under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL].

11.2. Elective 6LoWPAN Routing Header Type

IANA has updated the "Elective 6LoWPAN Routing Header Type" registry [RFC8138] under the "IPv6 Low Power Personal Area Network Parameters" registry group [IANA-6LO] as follows:

Value	Description	Reference
8	P-RPI-6LoRH	RFC 9914

Table 22: New Elective 6LoWPAN Routing Header Type

11.3. Critical 6LoWPAN Routing Header Type

IANA has updated the "Critical 6LoWPAN Routing Header Type" registry [RFC8138] under the "IPv6 Low Power Personal Area Network Parameters" registry group [IANA-6LO] as follows:

Value	Description	Reference
8	P-RPI-6LoRH	RFC 9914

Table 23: New Critical 6LoWPAN Routing Header Type

11.4. Registry for RPL Option Flags

IANA has created the "RPL Option Flags" registry, for the 8-bit RPL Option flags field as detailed in Figure 11, under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL]. The bits are indexed from 0 (leftmost) to 7. Each bit is tracked with the following qualities:

- Bit number (counting from bit 0 as the most significant bit)
- Indication when set
- Reference

The registration procedure is Standards Action [RFC8126]. The initial allocation is as indicated in Table 24:

Bit Number	Indication When Set	Reference
0	Down (O)	[RFC6553]
1	Rank-Error (R)	[RFC6553]
2	Forwarding-Error (F)	[RFC6553]
3	Projected-Route (P)	RFC 9914
4..255	Unassigned	

Table 24: Initial P-DAO-REQ Flags

11.5. RPL Control Codes

IANA has updated the "RPL Control Codes" registry under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL] as indicated in Table 25:

Code	Description	Reference
0x09	Projected DAO Request (P-DAO-REQ)	RFC 9914
0x0A	Projected DAO Request Acknowledgment (PDR-ACK)	RFC 9914

Table 25: New RPL Control Codes

11.6. RPL Control Message Options

IANA has updated the "RPL Control Message Options" registry under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL] as indicated in Table 26:

Value	Meaning	Reference
0x0F	Stateful Storing Mode VIO (SM-VIO)	RFC 9914
0x10	Source-Routed Non-Storing Mode VIO (NSM-VIO)	RFC 9914
0x11	Sibling Information Option (SIO)	RFC 9914

Table 26: RPL Control Message Options

11.7. Registry for Projected DAO Request Flags

IANA has created the "Projected DAO Request (P-DAO-REQ) Flags" registry under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL]. The bits are indexed from 0 (leftmost) to 7. Each bit is tracked with the following qualities:

- Bit number (counting from bit 0 as the most significant bit)
- Capability description

- Reference

The registration procedure is Standards Action [RFC8126]. The initial allocation is as indicated in Table 27:

Bit Number	Capability Description	Reference
0	PDR-ACK requested (K)	RFC 9914
1	Requested path should be redundant (R)	RFC 9914
2..255	Unassigned	

Table 27: Initial P-DAO-REQ Flags

11.8. Registry for Projected DAO Request Acknowledgment (PDR-ACK) Flags

IANA has created the "Projected DAO Request Acknowledgment (PDR-ACK) Flags" registry under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL]. The bits are indexed from 0 (leftmost) to 7. Each bit is tracked with the following qualities:

- Bit number (counting from bit 0 as the most significant bit)
- Capability description
- Reference

The registration procedure is Standards Action [RFC8126]. At the time of publication of this document, no bit has been assigned in this registry.

11.9. Registry for PDR-ACK Acceptance Status Values

IANA has created the "PDR-ACK Acceptance Status Values" registry under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL]. Each value is tracked with the following qualities:

- Value
- Meaning
- Reference

The possible values are expressed as a 6-bit unsigned integer (0..63). The registration procedure is Standards Action [RFC8126]. The initial allocation is as indicated in Table 28:

Value	Meaning	Reference
0	Unqualified Acceptance	RFC 9914
1..63	Unassigned	

Table 28: Acceptance Values of the PDR-ACK Status

11.10. Registry for PDR-ACK Rejection Status Values

IANA has created the "PDR-ACK Rejection Status Values" registry under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL]. Each value is tracked with the following qualities:

- Value
- Meaning
- Reference

The possible values are expressed as a 6-bit unsigned integer (0..63). The registration procedure is Standards Action [RFC8126]. The initial allocation is as indicated in Table 29:

Value	Meaning	Reference
0	Unqualified Rejection	RFC 9914
1	Transient Failure	RFC 9914
2..63	Unassigned	

Table 29: PDR-ACK Rejection Status Values

11.11. Registry for Via Information Options Flags

IANA has created the "Via Information Options (VIO) Flags" registry under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL]. The bits are indexed from 0 (leftmost) to 7. Each bit is tracked with the following qualities:

- Bit number (counting from bit 0 as the most significant bit)
- Capability description
- Reference

The registration procedure is Standards Action [RFC8126]. At the time of publication of this document, no bit has been assigned in this registry (see more in Section 5.3).

11.12. Registry for Sibling Information Option Flags

IANA has created the "Sibling Information Option (SIO) Flags" registry under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL]. The bits are indexed from 0 (leftmost) to 4. Each bit is tracked with the following qualities:

- Bit number (counting from bit 0 as the most significant bit)
- Capability description
- Reference

The registration procedure is Standards Action [RFC8126]. The initial allocation is as indicated in Table 30 (see more in Figure 17):

Bit Number	Capability Description	Reference
0	'S' flag: Sibling in same DODAG as self	RFC 9914
1	'B' flag: Connectivity to the sibling is roughly symmetrical	RFC 9914
2..4	Unassigned	

Table 30: Initial SIO Flags

11.13. Destination Advertisement Object Flag

IANA has updated the "Destination Advertisement Object (DAO) Flags" registry, created in Section 20.11 of [RPL], under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL] as indicated in Table 31 (see more in Section 4.1.1):

Bit Number	Capability Description	Reference
2	Projected DAO (P)	RFC 9914

Table 31: New Destination Advertisement Object (DAO) Flag

11.14. Destination Advertisement Object Acknowledgment Flag

IANA has updated the "Destination Advertisement Object (DAO) Acknowledgment Flags" registry, created in Section 20.12 of [RPL], under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL] as indicated in Table 32 (see more in Section 4.1.2):

Bit Number	Capability Description	Reference
1	Projected DAO Acknowledgment (P)	RFC 9914

Table 32: New Destination Advertisement Object Acknowledgment Flag

11.15. ICMPv6 Error Code

In some cases, RPL will return an ICMPv6 error message when a message cannot be forwarded along a P-Route.

Per this specification, IANA has updated the "Type 1 - Destination Unreachable" registry, in the "ICMPv6 'Code' Fields" registry, under the "Internet Control Message Protocol version 6 (ICMPv6) Parameters" registry group [IANA-ICMP] as indicated in Table 33.

Code	Name	Reference
9	Error in P-Route	RFC 9914

Table 33: New ICMPv6 Error Code

11.16. RPL Rejection Status Values

IANA has updated the "RPL Rejection Status" registry under the "Routing Protocol for Low Power and Lossy Networks (RPL)" registry group [IANA-RPL] as indicated in Table 34:

Value	Meaning	Reference
2	Out of Resources	RFC 9914
3	Error in VIO	RFC 9914
4	Predecessor Unreachable	RFC 9914
5	Unreachable Target	RFC 9914
6..63	Unassigned	

Table 34: RPL Rejection Status Values

12. References

12.1. 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>.

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <https://www.rfc-editor.org/info/rfc4443>.

[RFC6282] Hui, J., Ed. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, DOI 10.17487/RFC6282, September 2011, <https://www.rfc-editor.org/info/rfc6282>.

[RPL] 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>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8138] Thubert, P., Ed., Bormann, C., Toutain, L., and R. Cragie, "IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Routing Header", RFC 8138, DOI 10.17487/RFC8138, April 2017, <https://www.rfc-editor.org/info/rfc8138>.

[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>.

[RFC9008] Robles, M.I., Richardson, M., and P. Thubert, "Using RPI Option Type, Routing Header for Source Routes, and IPv6-in-IPv6 Encapsulation in the RPL Data Plane", RFC 9008, DOI 10.17487/RFC9008, April 2021, <https://www.rfc-editor.org/info/rfc9008>.

[RFC9030] Thubert, P., Ed., "An Architecture for IPv6 over the Time-Slotted Channel Hopping Mode of IEEE 802.15.4 (6TiSCH)", RFC 9030, DOI 10.17487/RFC9030, May 2021, <https://www.rfc-editor.org/info/rfc9030>.

[RAW-ARCH] Thubert, P., Ed., "Reliable and Available Wireless (RAW) Architecture", RFC 9912, DOI 10.17487/RFC9912, April 2026, <https://www.rfc-editor.org/info/rfc9912>.

12.2. Informative References

[INT-ARCH] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <https://www.rfc-editor.org/info/rfc1122>.

[RFC4655] Farrel, A., Vasseur, J.-P., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <https://www.rfc-editor.org/info/rfc4655>.

[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>.

[6LoWPAN] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, DOI 10.17487/RFC4944, September 2007, <https://www.rfc-editor.org/info/rfc4944>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", RFC 5440, DOI 10.17487/RFC5440, March 2009, <https://www.rfc-editor.org/info/rfc5440>.

[RFC6997] Goyal, M., Ed., Baccelli, E., Philipp, M., Brandt, A., and J. Martocci, "Reactive Discovery of Point-to-Point Routes in Low-Power and Lossy Networks", RFC 6997, DOI 10.17487/RFC6997, August 2013, <https://www.rfc-editor.org/info/rfc6997>.

[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>.

[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>.

[RFC8025] Thubert, P., Ed. and R. Cragie, "IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Paging Dispatch", RFC 8025, DOI 10.17487/RFC8025, November 2016, <https://www.rfc-editor.org/info/rfc8025>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

[RFC8505] Thubert, P., Ed., Nordmark, E., Chakrabarti, S., and C. Perkins, "Registration Extensions for IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Neighbor Discovery", RFC 8505, DOI 10.17487/RFC8505, November 2018, <https://www.rfc-editor.org/info/rfc8505>.

[RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <https://www.rfc-editor.org/info/rfc8754>.

[RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", RFC 8655, DOI 10.17487/RFC8655, October 2019, <https://www.rfc-editor.org/info/rfc8655>.

[RFC8930] Watteyne, T., Ed., Thubert, P., Ed., and C. Bormann, "On Forwarding 6LoWPAN Fragments over a Multi-Hop IPv6 Network", RFC 8930, DOI 10.17487/RFC8930, November 2020, <https://www.rfc-editor.org/info/rfc8930>.

[RFC8931] Thubert, P., Ed., "IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Selective Fragment Recovery", RFC 8931, DOI 10.17487/RFC8931, November 2020, <https://www.rfc-editor.org/info/rfc8931>.

[RFC8994] Eckert, T., Ed., Behringer, M., Ed., and S. Bjarnason, "An Autonomic Control Plane (ACP)", RFC 8994, DOI 10.17487/RFC8994, May 2021, <https://www.rfc-editor.org/info/rfc8994>.

[RFC9010] Thubert, P., Ed. and M. Richardson, "Routing for RPL (Routing Protocol for Low-Power and Lossy Networks) Leaves", RFC 9010, DOI 10.17487/RFC9010, April 2021, <https://www.rfc-editor.org/info/rfc9010>.

[RFC9031] Vučinić, M., Ed., Simon, J., Pister, K., and M. Richardson, "Constrained Join Protocol (CoJP) for 6TiSCH", RFC 9031, DOI 10.17487/RFC9031, May 2021, <https://www.rfc-editor.org/info/rfc9031>.

[RFC9035] Thubert, P., Ed. and L. Zhao, "A Routing Protocol for Low-Power and Lossy Networks (RPL) Destination-Oriented Directed Acyclic Graph (DODAG) Configuration Option for the 6LoWPAN Routing Header", RFC 9035, DOI 10.17487/RFC9035, April 2021, <https://www.rfc-editor.org/info/rfc9035>.

[RFC9450] Bernardos, CJ., Ed., Papadopoulos, G., Thubert, P., and F. Theoleyre, "Reliable and Available Wireless (RAW) Use Cases", RFC 9450, DOI 10.17487/RFC9450, August 2023, <https://www.rfc-editor.org/info/rfc9450>.

[NEW-TAGS] Kühlewind, M. and S. Krishnan, "Definition of new tags for relations between RFCs", Work in Progress, Internet-Draft, draft-kuehlewind-rswg-updates-tag-02, 8 July 2024, <https://datatracker.ietf.org/doc/html/draft-kuehlewind-rswg-updates-tag-02>.

[IANA-6LO] IANA, "IPv6 Low Power Personal Area Network Parameters", <https://www.iana.org/assignments/_6lowpan-parameters>.

[IANA-RPL] IANA, "Routing Protocol for Low Power and Lossy Networks (RPL)", <https://www.iana.org/assignments/rpl/>.

[IANA-ICMP] IANA, "Internet Control Message Protocol version 6 (ICMPv6) Parameters", <https://www.iana.org/assignments/icmpv6-parameters/>.

Acknowledgments

The authors wish to acknowledge JP. Vasseur, Remy Liubing, James Pylakutty, and Patrick Wetterwald for their contributions to the ideas developed here. Many thanks to Dominique Barthel and S.V.R. Anand for their global contribution to 6TiSCH, RAW, and this RFC, as well as text suggestions that were incorporated. Also, special thanks to Remous-Aris Koutsiamanis, Li Zhao, Dominique Barthel, and Toerless Eckert for their in-depth reviews, with many excellent suggestions that improved the readability and the content of the specification. Many thanks to Remous-Aris Koutsiamanis for his review during WG Last Call and to Maria Ines Robles for her thorough shepherd review. Many thanks to Warren Kumari, Ran Chen, Murray Kucherawy, Roman Danyliw, Klaas Wierenga, Deb Cooley, Éric Vyncke, Gunter Van de Velde, Sue Hares, and John Scudder for their comments and suggestions during the IETF Last Call and IESG review cycle.

Authors' Addresses

Pascal Thubert (EDITOR)
Independent
06330 Roquefort-les-Pins
France
Email: pascal.thubert@gmail.com

Rahul Arvind Jadhav
AccuKnox
Kundalahalli Village, Whitefield
Bangalore 560037
Karnataka
India
Phone: +91-080-49160700
Email: rahul.ietf@gmail.com

Michael C. Richardson
Sandelman Software Works
Email: mcr+ietf@sandelman.ca
URI: http://www.sandelman.ca/