

ROLL Working Group
Internet-Draft
Updates: [6553](#), [6550](#), [8138](#) (if approved)
Intended status: Standards Track
Expires: June 15, 2020

M. Robles
Aalto
M. Richardson
SSW
P. Thubert
Cisco
December 13, 2019

Using RPI Option Type, Routing Header for Source Routes and IPv6-in-IPv6
encapsulation in the RPL Data Plane
[draft-ietf-roll-useofrplinfo-33](#)

Abstract

This document looks at different data flows through LLN (Low-Power and Lossy Networks) where RPL (IPv6 Routing Protocol for Low-Power and Lossy Networks) is used to establish routing. The document enumerates the cases where [RFC6553](#) (RPI Option Type), [RFC6554](#) (Routing Header for Source Routes) and IPv6-in-IPv6 encapsulation is required in data plane. This analysis provides the basis on which to design efficient compression of these headers. This document updates [RFC6553](#) adding a change to the RPI Option Type. Additionally, this document updates [RFC6550](#) defining a flag in the DIO Configuration Option to indicate about this change and updates [RFC8138](#) as well to consider the new Option Type when the RPL Option is decompressed.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on June 15, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	3
1.1.	Overview	4
2.	Terminology and Requirements Language	5
3.	RPL Overview	6
4.	Updates to RFC6553 , RFC6550 and RFC8138	7
4.1.	Updates to RFC6550 : Advertising External Routes with Non- Storing Mode Signaling.	7
4.2.	Updates to RFC6553 : Indicating the new RPI Option Type. .	8
4.3.	Updates to RFC6550 : Indicating the new RPI in the DODAG Configuration Option Flag.	11
4.4.	Updates to RFC8138 : Indicating the way to decompress with the new RPI Option Type.	12
5.	Sample/reference topology	14
6.	Use cases	16
7.	Storing mode	19
7.1.	Storing Mode: Interaction between Leaf and Root	20
7.1.1.	SM: Example of Flow from RAL to root	21
7.1.2.	SM: Example of Flow from root to RAL	21
7.1.3.	SM: Example of Flow from root to RUL	22
7.1.4.	SM: Example of Flow from RUL to root	23
7.2.	SM: Interaction between Leaf and Internet.	23
7.2.1.	SM: Example of Flow from RAL to Internet	24
7.2.2.	SM: Example of Flow from Internet to RAL	24
7.2.3.	SM: Example of Flow from RUL to Internet	25
7.2.4.	SM: Example of Flow from Internet to RUL.	26
7.3.	SM: Interaction between Leaf and Leaf	27
7.3.1.	SM: Example of Flow from RAL to RAL	28
7.3.2.	SM: Example of Flow from RAL to RUL	29
7.3.3.	SM: Example of Flow from RUL to RAL	30
7.3.4.	SM: Example of Flow from RUL to RUL	31
8.	Non Storing mode	32

8.1.	Non-Storing Mode: Interaction between Leaf and Root . . .	33
8.1.1.	Non-SM: Example of Flow from RAL to root	34
8.1.2.	Non-SM: Example of Flow from root to RAL	34
8.1.3.	Non-SM: Example of Flow from root to RUL	35
8.1.4.	Non-SM: Example of Flow from RUL to root	36
8.2.	Non-Storing Mode: Interaction between Leaf and Internet .	37
8.2.1.	Non-SM: Example of Flow from RAL to Internet	37
8.2.2.	Non-SM: Example of Flow from Internet to RAL	38
8.2.3.	Non-SM: Example of Flow from RUL to Internet	39
8.2.4.	Non-SM: Example of Flow from Internet to RUL	40
8.3.	Non-SM: Interaction between Leafs	41
8.3.1.	Non-SM: Example of Flow from RAL to RAL	41
8.3.2.	Non-SM: Example of Flow from RAL to RUL	43
8.3.3.	Non-SM: Example of Flow from RUL to RAL	44
8.3.4.	Non-SM: Example of Flow from RUL to RUL	45
9.	Operational Considerations of supporting RUL-leaves	46
10.	Operational considerations of introducing 0x23	47
11.	IANA Considerations	48
12.	Security Considerations	49
13.	Acknowledgments	52
14.	References	52
14.1.	Normative References	52
14.2.	Informative References	54
	Authors' Addresses	56

[1.](#) Introduction

RPL (IPv6 Routing Protocol for Low-Power and Lossy Networks) [[RFC6550](#)] is a routing protocol for constrained networks. [[RFC6553](#)] defines the RPL Option carried within the IPv6 Hop-by-Hop Header to carry the RPLInstanceID and quickly identify inconsistencies (loops) in the routing topology. The RPL Option is commonly referred to as the RPL Packet Information (RPI) though the RPI is really the abstract information that is defined in [[RFC6550](#)] and transported in the RPL Option. [RFC6554](#) [[RFC6554](#)] defines the "RPL Source Route Header" (RH3), an IPv6 Extension Header to deliver datagrams within a RPL routing domain, particularly in non-storing mode.

These various items are referred to as RPL artifacts, and they are seen on all of the data-plane traffic that occurs in RPL routed networks; they do not in general appear on the RPL control plane traffic at all which is mostly hop-by-hop traffic (one exception being DAO messages in non-storing mode).

It has become clear from attempts to do multi-vendor interoperability, and from a desire to compress as many of the above

artifacts as possible that not all implementers agree when artifacts are necessary, or when they can be safely omitted, or removed.

The ROLL WG analyzed how [\[RFC2460\]](#) rules apply to storing and non-storing use of RPL. The result was 24 data plane use cases. They are exhaustively outlined here in order to be completely unambiguous. During the processing of this document, new rules were published as [\[RFC8200\]](#), and this document was updated to reflect the normative changes in that document.

This document updates [RFC6553](#), changing the value of the Option Type of the RPL Option to make [RFC8200](#) routers ignore this option when not recognized.

A Routing Header Dispatch for 6LoWPAN (6LoRH)([\[RFC8138\]](#)) defines a mechanism for compressing RPL Option information and Routing Header type 3 (RH3) [\[RFC6554\]](#), as well as an efficient IPv6-in-IPv6 technique.

Since some of the uses cases here described, use IPv6-in-IPv6 encapsulation. It MUST take in consideration, when encapsulation is applied, the [RFC6040](#) [\[RFC6040\]](#), which defines how the explicit congestion notification (ECN) field of the IP header should be constructed on entry to and exit from any IPV6-in-IPV6 tunnel. Additionally, it is recommended the reading of [\[I-D.ietf-intarea-tunnels\]](#) that explains the relationship of IP tunnels to existing protocol layers and the challenges in supporting IP tunneling.

Non-constrained uses of RPL are not in scope of this document, and applicability statements for those uses may provide different advice, E.g. [\[I-D.ietf-anima-autonomic-control-plane\]](#).

1.1. Overview

The rest of the document is organized as follows: [Section 2](#) describes the used terminology. [Section 3](#) provides a RPL Overview. [Section 4](#) describes the updates to [RFC6553](#), [RFC6550](#) and [RFC 8138](#). [Section 5](#) provides the reference topology used for the uses cases. [Section 6](#) describes the uses cases included. [Section 7](#) describes the storing mode cases and [section 8](#) the non-storing mode cases. [Section 9](#) describes the operational considerations of supporting RPL-unaware-leaves. [Section 10](#) depicts operational considerations for the proposed change on RPI Option Type, [section 11](#) the IANA considerations and then [section 12](#) describes the security aspects.

2. Terminology and 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.

Terminology defined in [[RFC7102](#)] applies to this document: LLN, RPL, RPL Domain and ROLL.

RPL Leaf: An IPv6 host that is attached to a RPL router and obtains connectivity through a RPL Destination Oriented Directed Acyclic Graph (DODAG). As an IPv6 node, a RPL Leaf is expected to ignore a consumed Routing Header and as an IPv6 host, it is expected to ignore a Hop-by-Hop header. It results that a RPL Leaf can correctly receive a packet with RPL artifacts. On the other hand, a RPL Leaf is not expected to generate RPL artifacts or to support IP-in-IP encapsulation. For simplification, this document uses the standalone term leaf to mean a RPL leaf.

RPL Packet Information (RPI): The abstract information that [[RFC6550](#)] places in IP packets. The term is commonly used, including in this document, to refer to the RPL Option [[RFC6553](#)] that transports that abstract information in an IPv6 Hop-by-Hop Header.

RPL-aware-node (RAN): A device which implements RPL. Please note that the device can be found inside the LLN or outside LLN.

RPL-Aware-Leaf(RAL): A RPL-aware-node that is also a RPL Leaf.

RPL-unaware-node: A device which does not implement RPL, thus the device is not-RPL-aware. Please note that the device can be found inside the LLN.

RPL-Unaware-Leaf(RUL): A RPL-unaware-node that is also a RPL Leaf.

6LoWPAN Node (6LN): [[RFC6775](#)] defines it as: "A 6LoWPAN node is any host or router participating in a LoWPAN. This term is used when referring to situations in which either a host or router can play the role described.". In this document, a 6LN acts as a leaf.

6LoWPAN Router (6LR): [[RFC6775](#)] defines it as: "An intermediate router in the LoWPAN that is able to send and receive Router Advertisements (RAs) and Router Solicitations (RSs) as well as forward and route IPv6 packets. 6LoWPAN routers are present only in route-over topologies."

6LoWPAN Border Router (6LBR): [[RFC6775](#)] defines it as: "A border router located at the junction of separate 6LoWPAN networks or between a 6LoWPAN network and another IP network. There may be one or more 6LBRs at the 6LoWPAN network boundary. A 6LBR is the responsible authority for IPv6 prefix propagation for the 6LoWPAN network it is serving. An isolated LoWPAN also contains a 6LBR in the network, which provides the prefix(es) for the isolated network."

Flag Day: A transition that involves having a network with different values of RPI Option Type. Thus the network does not work correctly (Lack of interoperation).

Hop-by-hop re-encapsulation: The term "hop-by-hop re-encapsulation" header refers to adding a header that originates from a node to an adjacent node, using the addresses (usually the GUA or ULA, but could use the link-local addresses) of each node. If the packet must traverse multiple hops, then it must be decapsulated at each hop, and then re-encapsulated again in a similar fashion.

Non-Storing Mode (Non-SM): RPL mode of operation in which the RPL-aware-nodes send information to the root about its parents. Thus, the root know the topology, then the intermediate 6LRs do not maintain routing state so that source routing is needed.

Storing Mode (SM): RPL mode of operation in which RPL-aware-nodes (6LRs) maintain routing state (of the children) so that source routing is not needed.

Note: Due to lack of space in some figures (tables) we refers IPv6-in-IPv6 as IP6-IP6.

3. RPL Overview

RPL defines the RPL Control messages (control plane), a new ICMPv6 [[RFC4443](#)] message with Type 155. DIS (DODAG Information Solicitation), DIO (DODAG Information Object) and DAO (Destination Advertisement Object) messages are all RPL Control messages but with different Code values. A RPL Stack is shown in Figure 1.

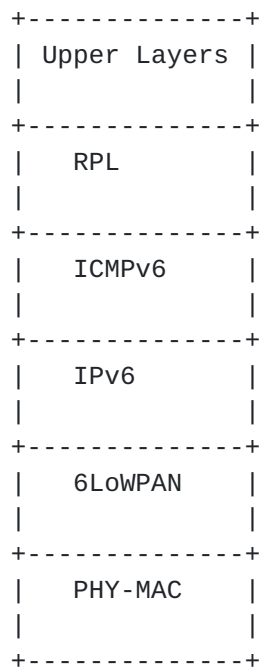


Figure 1: RPL Stack.

RPL supports two modes of Downward traffic: in storing mode (SM), it is fully stateful; in non-storing mode (Non-SM), it is fully source routed. A RPL Instance is either fully storing or fully non-storing, i.e. a RPL Instance with a combination of storing and non-storing nodes is not supported with the current specifications at the time of writing this document.

4. Updates to [RFC6553](#), [RFC6550](#) and [RFC8138](#)

4.1. Updates to [RFC6550](#): Advertising External Routes with Non-Storing Mode Signaling.

[Section 6.7.8. of \[RFC6550\]](#) introduces the 'E' flag that is set to indicate that the 6LR that generates the DAO redistributes external targets into the RPL network. An external Target is a Target that has been learned through an alternate protocol, for instance a route to a prefix that is outside the RPL domain but reachable via a 6LR. Being outside of the RPL domain, a node that is reached via an external target cannot be guaranteed to ignore the RPL artifacts and cannot be expected to process the [\[RFC8138\]](#) compression correctly. This means that the RPL artifacts should be contained in an IP-in-IP encapsulation that is removed by the 6LR, and that any remaining compression should be expanded by the 6LR before it forwards a packet outside the RPL domain.

This specification updates [[RFC6550](#)] to RECOMMEND that external targets are advertised using Non-Storing Mode DAO messaging even in a Storing-Mode network. This way, external routes are not advertised within the DODAG and all packets to an external target reach the Root like normal Non-Storing Mode traffic. The Non-Storing Mode DAO informs the Root of the address of the 6LR that injects the external route, and the root uses IP-in-IP encapsulation to that 6LR, which terminates the IP-in-IP tunnel and forwards the original packet outside the RPL domain free of RPL artifacts. This whole operation is transparent to intermediate routers that only see traffic between the 6LR and the Root, and only the Root and the 6LRs that inject external routes in the network need to be upgraded to add this function to the network.

A RUL is a special case of external target when the target is actually a host and it is known to support a consumed Routing Header and to ignore a HbH header as prescribed by [[RFC8200](#)]. The target may have been learned through as a host route or may have been registered to the 6LR using [[RFC8505](#)]. IP-in-IP encapsulation MAY be avoided for Root to RUL communication if the RUL is known to process the packets as forwarded by the parent 6LR without decapsulation.

In order to enable IP-in-IP all the way to a 6LN, it is beneficial that the 6LN supports decapsulating IP-in-IP, but that is not assumed by [[RFC8504](#)]. If the 6LN is a RUL, the Root that encapsulates a packet SHOULD terminate the tunnel at a parent 6LR unless it is aware that the RUL supports IP-in-IP decapsulation.

A node that is reachable over an external route is not expected to support [[RFC8138](#)]. Whether a decapsulation took place or not and even when the 6LR is delivering the packet to a RUL, the 6LR that injected an external route MUST uncompress the packet before forwarding over that external route.

4.2. Updates to [RFC6553](#): Indicating the new RPI Option Type.

This modification is required to be able to send, for example, IPv6 packets from a RPL-Aware-Leaf to a RPL-unaware node through Internet (see [Section 7.2.1](#)), without requiring IPv6-in-IPv6 encapsulation.

[[RFC6553](#)] ([Section 6](#), Page 7) states as shown in Figure 2, that in the Option Type field of the RPL Option, the two high order bits must be set to '01' and the third bit is equal to '1'. The first two bits indicate that the IPv6 node must discard the packet if it doesn't recognize the Option Type, and the third bit indicates that the Option Data may change in route. The remaining bits serve as the Option Type.

Hex	Binary Value			Description	Reference
Value					
	act	chg	rest		
0x63	01	1	00011	RPL Option	[RFC6553]

Figure 2: Option Type in RPL Option.

This document illustrates that it is not always possible to know for sure at the source that a packet will only travel within the RPL domain or may leave it.

At the time [RFC6553] was published, leaking a Hop-by-Hop header in the outer IPv6 header chain could potentially impact core routers in the internet. So at that time, it was decided to encapsulate any packet with a RPL Option using IPv6-in-IPv6 in all cases where it was unclear whether the packet would remain within the RPL domain. In the exception case where a packet would still leak, the Option Type would ensure that the first router in the Internet that does not recognize the option would drop the packet and protect the rest of the network.

Even with [RFC8138] that compresses the IPv6-in-IPv6 header, this approach yields extra bytes in a packet which means consuming more energy, more bandwidth, incurring higher chances of loss and possibly causing a fragmentation at the 6LoWPAN level. This impacts the daily operation of constrained devices for a case that generally does not happen and would not heavily impact the core anyway.

While intention was and remains that the Hop-by-Hop header with a RPL Option should be confined within the RPL domain, this specification modifies this behavior in order to reduce the dependency on IPv6-in-IPv6 and protect the constrained devices. [Section 4 of \[RFC8200\]](#) clarifies the behaviour of routers in the Internet as follows: "it is now expected that nodes along a packet's delivery path only examine and process the Hop-by-Hop Options header if explicitly configured to do so".

When unclear about the travel of a packet, it becomes preferable for a source not to encapsulate, accepting the fact that the packet may leave the RPL domain on its way to its destination. In that event, the packet should reach its destination and should not be discarded by the first node that does not recognize the RPL Option. But with the current value of the Option Type, if a node in the Internet is configured to process the Hop-by-Hop header, and if such node encounters an option with the first two bits set to 01 and conforms

to [\[RFC8200\]](#), it will drop the packet. Host systems should do the same, irrespective of the configuration.

Thus, this document updates the Option Type of the RPL Option [\[RFC6553\]](#), abusively naming it RPI Option Type for simplicity, to (Figure 3): the two high order bits MUST be set to '00' and the third bit is equal to '1'. The first two bits indicate that the IPv6 node MUST skip over this option and continue processing the header ([\[RFC8200\] Section 4.2](#)) if it doesn't recognize the Option Type, and the third bit continues to be set to indicate that the Option Data may change en route. The five rightmost bits remain at 0x3. This ensures that a packet that leaves the RPL domain of an LLN (or that leaves the LLN entirely) will not be discarded when it contains the RPL Option.

With the new Option Type, if an IPv6 (intermediate) node (RPL-not-capable) receives a packet with an RPL Option, it should ignore the Hop-by-Hop RPL Option (skip over this option and continue processing the header). This is relevant, as it was mentioned previously, in the case that there is a flow from RAL to Internet (see [Section 7.2.1](#)).

This is a significant update to [\[RFC6553\]](#).

Hex	Binary Value			Description	Reference
Value					
	act	chg	rest		
0x23	00	1	00011	RPL Option	[RFCXXXX](*)

Figure 3: Revised Option Type in RPL Option. (*)represents this document

Without the signaling described below, this change would otherwise create a lack of interoperation (flag day) for existing networks which are currently using 0x63 as the RPI Option Type value. A move to 0x23 will not be understood by those networks. It is suggested that RPL implementations accept both 0x63 and 0x23 when processing the header.

When forwarding packets, implementations SHOULD use the same value as it was received. This is required because, RPI Option Type can not be changed by [\[RFC8200\] - Section 4.2](#). It allows to the network to be incrementally upgraded, and for the DODAG root to know which parts of the network are upgraded.

When originating new packets, implementations SHOULD have an option to determine which value to originate with, this option is controlled by the DIO option described below.

The change of RPI Option Type from 0x63 to 0x23, makes all [\[RFC8200\] Section 4.2](#) compliant nodes tolerant of the RPL artifacts. There is therefore no longer a necessity to remove the artifacts when sending traffic to the Internet. This change clarifies when to use an IPv6-in-IPv6 header, and how to address them: The Hop-by-Hop Options Header containing the RPI MUST always be added when 6LRs originate packets (without IPv6-in-IPv6 headers), and IPv6-in-IPv6 headers MUST always be added when a 6LR find that it needs to insert a Hop-by-Hop Options Header containing the RPL Option. The IPv6-in-IPv6 header is to be addressed to the RPL root when on the way up, and to the end-host when on the way down.

In the non-storing case, dealing with not-RPL aware leaf nodes is much easier as the 6LBR (DODAG root) has complete knowledge about the connectivity of all DODAG nodes, and all traffic flows through the root node.

The 6LBR can recognize not-RPL aware leaf nodes because it will receive a DAO about that node from the 6LR immediately above that not-RPL aware node. This means that the non-storing mode case can avoid ever using hop-by-hop re-encapsulation headers for traffic originating from the root to the leafs.

The non-storing mode case does not require the type change from 0x63 to 0x23, as the root can always create the right packet. The type change does not adversely affect the non-storing case.

4.3. Updates to [RFC6550](#): Indicating the new RPI in the DODAG Configuration Option Flag.

In order to avoid a Flag Day caused by lack of interoperation between new RPI Option Type (0x23) and old RPI Option Type (0x63) nodes, this section defines a flag in the DIO Configuration Option, to indicate when the new RPI Option Type can be safely used. This means, the flag is going to indicate the value of Option Type that the network is using for the RPL Option. Thus, when a node join to a network will know which value to use. With this, RPL-capable nodes know if it is safe to use 0x23 when creating a new RPL Option. A node that forwards a packet with an RPI MUST NOT modify the Option Type of the RPL Option.

This is done using a DODAG Configuration Option flag which will signal "RPI 0x23 enable" and propagate through the network. [Section 6.3.1. of \[RFC6550\]](#) defines a 3-bit Mode of Operation (MOP)

in the DIO Base Object. The flag is defined only for MOP value between 0 to 6. For a MOP value of 7 or above, the flag MAY indicate something different and MUST NOT be interpreted as "RPI 0x23 enable" unless the specification of the MOP indicates to do so.

As stated in [RFC6550] the DODAG Configuration option is present in DIO messages. The DODAG Configuration option distributes configuration information. It is generally static, and does not change within the DODAG. This information is configured at the DODAG root and distributed throughout the DODAG with the DODAG Configuration option. Nodes other than the DODAG root do not modify this information when propagating the DODAG Configuration option.

Currently, the DODAG Configuration Option in [RFC6550] states: "the unused bits MUST be initialize to zero by the sender and MUST be ignored by the receiver". If the flag is received with a value zero (which is the default), then new nodes will remain in [RFC6553](#) Compatible Mode; originating traffic with the old-RPI Option Type (0x63) value. If the flag is received with a value of 1, then the option value for the RPL Option MUST be set to 0x23.

Bit number three of the flag field in the DODAG Configuration option is to be used as shown in Figure 4 :

Bit number	Description	Reference
3	RPI 0x23 enable	This document

Figure 4: DODAG Configuration Option Flag to indicate the RPI-flag-day.

In case of rebooting, the node (6LN or 6LR) does not remember the RPI Option Type, that is if the flag is set, so DIO messages sent by the node would be set with the flag unset until a DIO message is received with the flag set indicating the new RPI Option Type. The node sets to 0x23 if the node supports this feature.

4.4. Updates to [RFC8138](#): Indicating the way to decompress with the new RPI Option Type.

This modification is required to be able to decompress the RPL Option with the new Option Type of 0x23.

RPI-6LoRH header provides a compressed form for the RPL RPI [RFC8138] in [section 6](#). A node that is decompressing this header MUST decompress using the RPI Option Type that is currently active: that is, a choice between 0x23 (new) and 0x63 (old). The node will know which to use based upon the presence of the flag in the DODAG Configuration Option defined in [Section 4.3](#). E.g. If the network is in 0x23 mode (by DIO option), then it should be decompressed to 0x23.

[RFC8138] [section 7](#) documents how to compress the IPv6-in-IPv6 header.

There are potential significant advantages to having a single code path that always processes IPv6-in-IPv6 headers with no conditional branches.

In Storing Mode, for the examples of Flow from RAL to RUL and RUL to RUL comprise an IPv6-in-IPv6 and RPI compressed headers. The use of the IPv6-in-IPv6 header is MANDATORY in this case, and it SHOULD be compressed with [\[RFC8138\] section 7](#). Figure 5 illustrates the case in Storing mode where the packet is received from the Internet, then the root encapsulates the packet to insert the RPI. In that example, the leaf is not known to support [RFC 8138](#), and the packet is encapsulated to the 6LR that is the parent and last hop to the final destination.

```

+-+ ... -+-+ ... +-+ ... -+-+ +-+--+ ... +-+ ... -+++ ... +-...
|11110001|SRH-6LoRH| RPI- |IP-in-IP| NH=1      |11110CPP| UDP | UDP
|Page 1  |Type1 S=0| 6LoRH |6LoRH  |LOWPAN_IPHC| UDP   | hdr |Payld
+-+ ... -+-+ ... +-+ ... -+-+ .+-+--+--+ ... +-+ ... -+ ... +-...
      <-4bytes->                <-      RFC 6282      ->
                                   No RPL artifact

```

Figure 5: RPI Inserted by the Root in Storing Mode

In Figure 5, the source of the IPv6-in-IPv6 encapsulation is the Root, so it is elided in the IP-in-IP 6LoRH. The destination is the parent 6LR of the destination of the inner packet so it cannot be elided. It is placed as the single entry in an SRH-6LoRH as the first 6LoRH. There is a single entry so the SRH-6LoRH Size is 0. In that example, the type is 1 so the 6LR address is compressed to 2 bytes. It results that the total length of the SRH-6LoRH is 4 bytes. Follows the RPI-6LoRH and then the IP-in-IP 6LoRH. When the IP-in-IP 6LoRH is removed, all the router headers that precede it are also removed. The Paging Dispatch [\[RFC8025\]](#) may also be removed if there was no previous Page change to a Page other than 0 or 1, since the

LOWPAN_IPHC is encoded in the same fashion in the default Page 0 and in Page 1. The resulting packet to the destination is the inner packet compressed with [[RFC6282](#)].

5. Sample/reference topology

A RPL network in general is composed of a 6LBR, Backbone Router (6BBR), 6LR and 6LN as leaf logically organized in a DODAG structure.

Figure 6 shows the reference RPL Topology for this document. The letters above the nodes are there so that they may be referenced in subsequent sections. In the figure, 6LR represents a full router node. The 6LN is a RPL aware router, or host (as a leaf). Additionally, for simplification purposes, it is supposed that the 6LBR has direct access to Internet and is the root of the DODAG, thus the 6BBR is not present in the figure.

The 6LN leaves (RAL) marked as (F, H and I) are RPL nodes with no children hosts.

The leafs marked as RUL (G and J) are devices which do not speak RPL at all (not-RPL-aware), but uses Router-Advertisements, 6LowPAN DAR/DAC and efficient-ND only to participate in the network [[RFC6775](#)]. In the document these leafs (G and J) are also referred to as an IPv6 node.

The 6LBR ("A") in the figure is the root of the Global DODAG.

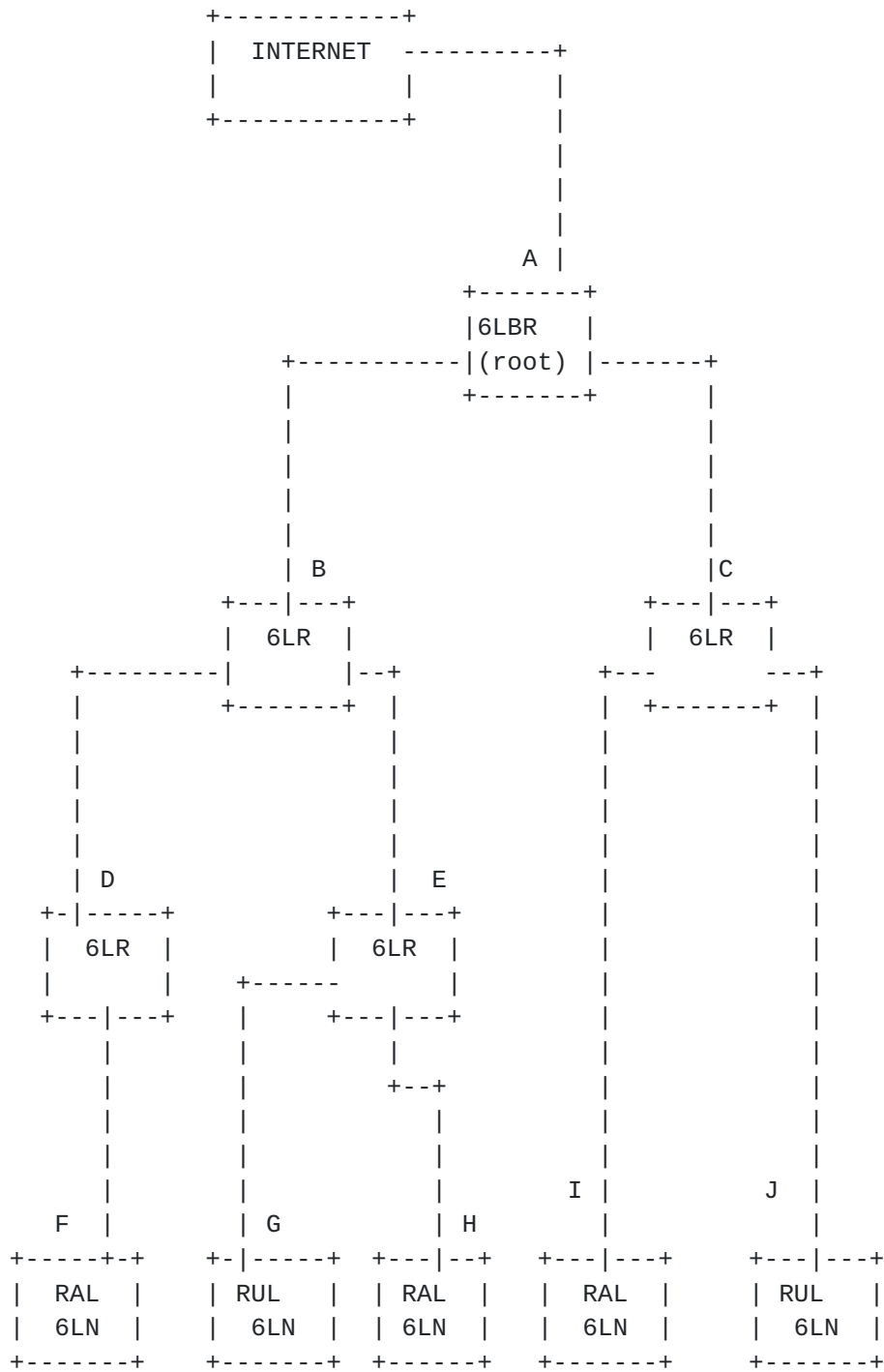


Figure 6: A reference RPL Topology.

6. Use cases

In the data plane a combination of [RFC6553](#), [RFC6554](#) and IPv6-in-IPv6 encapsulation are going to be analyzed for a number of representative traffic flows.

This document assumes that the LLN is using the no-drop RPI Option Type of 0x23.

The use cases describe the communication in the following cases: - Between RPL-aware-nodes with the root (6LBR) - Between RPL-aware-nodes with the Internet - Between RUL nodes within the LLN (e.g. see [Section 7.1.4](#)) - Inside of the LLN when the final destination address resides outside of the LLN (e.g. see [Section 7.2.3](#)).

The uses cases are as follows:

Interaction between Leaf and Root:

RAL to root

root to RAL

RUL to root

root to RUL

Interaction between Leaf and Internet:

RAL to Internet

Internet to RAL

RUL to Internet

Internet to RUL

Interaction between Leafs:

RAL to RAL

RAL to RUL

RUL to RAL

RUL to RUL

This document is consistent with the rule that a Header cannot be inserted or removed on the fly inside an IPv6 packet that is being routed. This is a fundamental precept of the IPv6 architecture as outlined in [[RFC8200](#)].

As the rank information in the RPI artifact is changed at each hop, it will typically be zero when it arrives at the DODAG root. The DODAG root MUST force it to zero when passing the packet out to the Internet. The Internet will therefore not see any SenderRank information.

Despite being legal to leave the RPI artifact in place, an intermediate router that needs to add an extension header (e.g. RH3 or RPL Option) MUST still encapsulate the packet in an (additional) outer IP header. The new header is placed after this new outer IP header.

A corollary is that an RH3 or RPL Option can only be removed by an intermediate router if it is placed in an encapsulating IPv6 Header, which is addressed TO the intermediate router. When it does so, the whole encapsulating header must be removed. (A replacement may be added). This sometimes can result in outer IP headers being addressed to the next hop router using link-local address.

Both the RPL Option and the RH3 headers may be modified in very specific ways by routers on the path of the packet without the need to add and remove an encapsulating header. Both headers were designed with this modification in mind, and both the RPL RH3 and the RPL Option are marked mutable but recoverable: so an IPsec AH security header can be applied across these headers, but it can not secure the values which mutate.

The RPI MUST be present in every single RPL data packet.

Prior to [[RFC8138](#)], there was significant interest in removing the RPI for downward flows in non-storing mode. The exception covered a very small number of cases, and causes significant interoperability challenges, yet costed significant code and testing complexity. The ability to compress the RPI down to three bytes or less removes much of the pressure to optimize this any further [[I-D.ietf-anima-autonomic-control-plane](#)].

The earlier examples are more extensive to make sure that the process is clear, while later examples are more concise.

The uses cases are delineated based on the following requirements:

The RPI has to be in every packet that traverses the LLN.

- Because of the previous requirement, packets from the Internet have to be encapsulated.
- A Header cannot be inserted or removed on the fly inside an IPv6 packet that is being routed.
- Extension headers may not be added or removed except by the sender or the receiver.
- RPI and RH3 headers may be modified by routers on the path of the packet without the need to add and remove an encapsulating header.
- An RH3 or RPL Option can only be removed by an intermediate router if it is placed in an encapsulating IPv6 Header, which is addressed to the intermediate router.
- Non-storing mode requires downstream encapsulation by root for RH3.

The uses cases are delineated based on the following assumptions:

This document assumes that the LLN is using the no-drop RPI Option Type (0x23).

- Each IPv6 node (including Internet routers) obeys [[RFC8200](#)] [RFC 8200](#), so that 0x23 RPI Option type can be safely inserted.
- All 6LRs obey [RFC 8200](#) [[RFC8200](#)].
- The RPI is ignored at the IPv6 dst node (RUL).
- In the uses cases, we assume that the RAL supports IP-in-IP encapsulation.
- In the uses cases, we dont assume that the RUL supports IP-in-IP encapsulation.
- Non-constrained uses of RPL are not in scope of this document.
- Compression is based on [[RFC8138](#)].
- The flow label [[RFC6437](#)] is not needed in RPL.

7. Storing mode

In storing mode (SM) (fully stateful), the sender can determine if the destination is inside the LLN by looking if the destination address is matched by the DIO's Prefix Information Option (PIO) option.

The following table (Figure 7) itemizes which headers are needed in each of the following scenarios. It indicates if the IPv6-in-IPv6 header that is added, must be addressed to the final destination (the RAL node that is the target(tgt)), to the "root" or if a hop-by-hop header must be added (indicated by "hop"). In the hop-by-hop basis, the destination address for the next hop is the link-layer address of the next hop.

In cases where no IPv6-in-IPv6 header is needed, the column states as "No". If the IPv6-in-IPv6 header is needed is a "must".

In all cases the RPI is needed, since it identifies inconsistencies (loops) in the routing topology. In all cases the RH3 is not needed because it is not used in storing mode.

In each case, 6LR_i are the intermediate routers from source to destination. " $1 \leq i \leq n$ ", n is the number of routers (6LR) that the packet goes through from source (6LN) to destination.

The leaf can be a router 6LR or a host, both indicated as 6LN. The root refers to the 6LBR (see Figure 6).

Interaction between	Use Case	IPv6-in-IPv6	IPv6-in-IPv6 dst
Leaf - Root	RAL to root	No	No
	root to RAL	No	No
	root to RUL	No	No
	RUL to root	must	hop or root
Leaf - Internet	RAL to Int	No	No
	Int to RAL	must	RAL (tgt)
	RUL to Int	must	hop or root
	Int to RUL	must	6LR
Leaf - Leaf	RAL to RAL	No	No
	RAL to RUL	No	No
	RUL to RAL	must	RAL (tgt)
	RUL to RUL	must	6LR

Figure 7: Table of IPv6-in-IPv6 encapsulation in Storing mode.

7.1. Storing Mode: Interaction between Leaf and Root

In this section is described the communication flow in storing mode (SM) between,

RAL to root

root to RAL

RUL to root

root to RUL

7.1.1. SM: Example of Flow from RAL to root

In storing mode, [RFC 6553](#) (RPI) is used to send RPL Information instanceID and rank information.

In this case the flow comprises:

RAL (6LN) --> 6LR_i --> root(6LBR)

For example, a communication flow could be: Node F --> Node D --> Node B --> Node A root(6LBR)

The RAL (Node F) inserts the RPI, and sends the packet to 6LR (Node D) which decrements the rank in the RPI and sends the packet up. When the packet arrives at 6LBR (Node A), the RPI is removed and the packet is processed.

No IPv6-in-IPv6 header is required.

The RPI can be removed by the 6LBR because the packet is addressed to the 6LBR. The RAL must know that it is communicating with the 6LBR to make use of this scenario. The RAL can know the address of the 6LBR because it knows the address of the root via the DODAGID in the DIO messages.

The Table 1 summarizes what headers are needed for this use case.

Header	RAL src	6LR_i	6LBR dst
Inserted headers	RPI	--	--
Removed headers	--	--	RPI
Re-added headers	--	--	--
Modified headers	--	RPI	--
Untouched headers	--	--	--

Table 1: SM: Summary of the use of headers from RAL to root

7.1.2. SM: Example of Flow from root to RAL

In this case the flow comprises:

root (6LBR) --> 6LR_i --> RAL (6LN)

For example, a communication flow could be: Node A root(6LBR) --> Node B --> Node D --> Node F

In this case the 6LBR inserts RPI and sends the packet down, the 6LR is going to increment the rank in RPI (it examines the instanceID to identify the right forwarding table), the packet is processed in the RAL and the RPI removed.

No IPv6-in-IPv6 header is required.

The Table 2 summarizes what headers are needed for this use case.

Header	6LBR src	6LR_i	RAL dst
Inserted headers	RPI	--	--
Removed headers	--	--	RPI
Re-added headers	--	--	--
Modified headers	--	RPI	--
Untouched headers	--	--	--

Table 2: SM: Summary of the use of headers from root to RAL

[7.1.3.](#) SM: Example of Flow from root to RUL

In this case the flow comprises:

root (6LBR) --> 6LR_i --> RUL (IPv6 dst node)

For example, a communication flow could be: Node A root(6LBR) --> Node B --> Node E --> Node G

As the RPI extension can be ignored by the RUL, this situation is identical to the previous scenario.

The Table 3 summarizes what headers are needed for this use case.

Header	6LBR src	6LR_i	RUL (IPv6 dst node)
Inserted headers	RPI	--	--
Removed headers	--	--	--
Re-added headers	--	--	--
Modified headers	--	RPI	--
Untouched headers	--	--	RPI (Ignored)

Table 3: SM: Summary of the use of headers from root to RUL

7.1.4. SM: Example of Flow from RUL to root

In this case the flow comprises:

RUL (IPv6 src node) --> 6LR_1 --> 6LR_i --> root (6LBR)

For example, a communication flow could be: Node G --> Node E --> Node B --> Node A root(6LBR)

When the packet arrives from IPv6 node (Node G) to 6LR_1 (Node E), the 6LR_1 will insert a RPI, encapsulated in a IPv6-in-IPv6 header. The IPv6-in-IPv6 header can be addressed to the next hop (Node B), or to the root (Node A). The root removes the header and processes the packet.

The Figure 8 shows the table that summarizes what headers are needed for this use case. [1] refers the case where the IPv6-in-IPv6 header is addressed to the next hop (Node B). [2] refers the case where the IPv6-in-IPv6 header is addressed to the root (Node A).

Header	RUL	6LR_1	6LR_i	6LBR dst
	src node			
Inserted headers	--	IP6-IP6(RPI)	IP6-IP6(RPI)[1]	--
Removed headers	--	--	IP6-IP6(RPI)[1]	IP6-IP6(RPI)[1][2]
Re-added headers	--	--	--	--
Modified headers	--	--	IP6-IP6(RPI)[2]	--
Untouched headers	--	--	--	--

Figure 8: SM: Summary of the use of headers from RUL to root.

7.2. SM: Interaction between Leaf and Internet.

In this section is described the communication flow in storing mode (SM) between,

RAL to Internet

Internet to RAL

RUL to Internet

Internet to RUL

7.2.1. SM: Example of Flow from RAL to Internet

RPL information from [RFC 6553](#) may go out to Internet as it will be ignored by nodes which have not been configured to be RPI aware.

In this case the flow comprises:

RAL (6LN) --> 6LR_i --> root (6LBR) --> Internet

For example, the communication flow could be: Node F --> Node D --> Node B --> Node A root(6LBR) --> Internet

No IPv6-in-IPv6 header is required.

Note: In this use case it is used a node as leaf, but this use case can be also applicable to any RPL-aware-node type (e.g. 6LR)

The Table 4 summarizes what headers are needed for this use case.

Header	RAL src	6LR_i	6LBR	Internet dst
Inserted headers	RPI	--	--	--
Removed headers	--	--	--	--
Re-added headers	--	--	--	--
Modified headers	--	RPI	--	--
Untouched headers	--	--	RPI	RPI (Ignored)

Table 4: SM: Summary of the use of headers from RAL to Internet

7.2.2. SM: Example of Flow from Internet to RAL

In this case the flow comprises:

Internet --> root (6LBR) --> 6LR_i --> RAL (6LN)

For example, a communication flow could be: Internet --> Node A root(6LBR) --> Node B --> Node D --> Node F

When the packet arrives from Internet to 6LBR the RPI is added in a outer IPv6-in-IPv6 header (with the IPv6-in-IPv6 destination address set to the RAL) and sent to 6LR, which modifies the rank in the RPI. When the packet arrives at the RAL the RPI is removed and the packet processed.

The Figure 9 shows the table that summarizes what headers are needed for this use case.

Header	Internet	6LBR	6LR_i	RAL dst
src				
Inserted headers	--	IP6-IP6(RPI)	--	--
Removed headers	--	--	--	IP6-IP6(RPI)
Re-added headers	--	--	--	--
Modified headers	--	--	IP6-IP6(RPI)	--
Untouched headers	--	--	--	--

Figure 9: SM: Summary of the use of headers from Internet to RAL.

7.2.3. SM: Example of Flow from RUL to Internet

In this case the flow comprises:

RUL (IPv6 src node) --> 6LR_1 --> 6LR_i --> root (6LBR) --> Internet

For example, a communication flow could be: Node G --> Node E --> Node B --> Node A root(6LBR) --> Internet

The 6LR_1 (i=1) node will add an IPv6-in-IPv6(RPI) header addressed either to the root, or hop-by-hop such that the root can remove the RPI before passing upwards. The IPv6-in-IPv6 addressed to the root cause less processing overhead. On the other hand, with hop-by-hop the intermediate routers can check the routing tables for a better routing path, thus it could be more efficient and faster. Implementation should decide which approach to take.

The originating node will ideally leave the IPv6 flow label as zero so that the packet can be better compressed through the LLN. The 6LBR will set the flow label of the packet to a non-zero value when sending to the Internet, for details check [[RFC6437](#)].

The Figure 10 shows the table that summarizes what headers are needed for this use case. In the table, [1] shows the case when packet is addressed to the root. [2] shows the case when the packet is addressed hop-by-hop.

Header	IPv6	6LR_1	6LR_i	6LBR	Internet
	src		[i=2,...,n]		dst
	node				
	(RUL)				
Inserted headers	--	IP6-IP6(RPI)	IP6-IP6(RPI) [2]	--	--
Removed headers	--	--	IP6-IP6(RPI) [2]	IP6-IP6(RPI) [1][2]	--
Re-added headers	--	--	--	--	--
Modified headers	--	--	IP6-IP6(RPI) [1]	--	--
Untouched headers	--	--	--	--	--

Figure 10: SM: Summary of the use of headers from RUL to Internet.

[7.2.4.](#) SM: Example of Flow from Internet to RUL.

In this case the flow comprises:

Internet --> root (6LBR) --> 6LR_i --> RUL (IPv6 dst node)

For example, a communication flow could be: Internet --> Node A
root(6LBR) --> Node B --> Node E --> Node G

The 6LBR will have to add an RPI within an IPv6-in-IPv6 header. The IPv6-in-IPv6 is addressed to the 6LR parent of the 6LR_i.

Further details about this are mentioned in [\[I-D.ietf-roll-unaware-leaves\]](#), which specifies RPL routing for a 6LN acting as a plain host and not being aware of RPL.

The 6LBR may set the flow label on the inner IPv6-in-IPv6 header to zero in order to aid in compression [\[RFC8138\]](#)[\[RFC6437\]](#).

The Figure 11 shows the table that summarizes what headers are needed for this use case.

Header	Internet src	6LBR	6LR_i	IPv6 dst node
Inserted headers	--	IP6-IP6(RPI)	IP6-IP6(RPI)	--
Removed headers	--	--	IP6-IP6(RPI)	--
Re-added headers	--	--	--	--
Modified headers	--	--	--	--
Untouched headers	--	--	--	--

Figure 11: SM: Summary of the use of headers from Internet to RUL.

7.3. SM: Interaction between Leaf and Leaf

In this section is described the communication flow in storing mode (SM) between,

RAL to RAL

RAL to RUL

RUL to RAL

RUL to RUL

7.3.1. SM: Example of Flow from RAL to RAL

In [RFC6550] RPL allows a simple one-hop optimization for both storing and non-storing networks. A node may send a packet destined to a one-hop neighbor directly to that node. See [section 9 in \[RFC6550\]](#).

When the nodes are not directly connected, then in storing mode, the flow comprises:

```
RAL src (6LN) --> 6LR_ia --> common parent (6LR_x) --> 6LR_id --> RAL
dst (6LN)
```

For example, a communication flow could be: Node F --> Node D --> Node B --> Node E --> Node H

6LR_ia (Node D) are the intermediate routers from source to the common parent (6LR_x) (Node B). In this case, $1 \leq ia \leq n$, n is the number of routers (6LR) that the packet goes through from RAL (Node F) to the common parent 6LR_x (Node B).

6LR_id (Node E) are the intermediate routers from the common parent (6LR_x) (Node B) to destination RAL (Node H). In this case, $1 \leq id \leq m$, m is the number of routers (6LR) that the packet goes through from the common parent (6LR_x) to destination RAL (Node H).

It is assumed that the two nodes are in the same RPL Domain (that they share the same DODAG root). At the common parent (Node B), the direction of RPI is changed (from decreasing to increasing the rank).

While the 6LR nodes will update the RPI, no node needs to add or remove the RPI, so no IPv6-in-IPv6 headers are necessary.

The Table 5 summarizes what headers are needed for this use case.

Header	RAL src	6LR_ia	6LR_x (common parent)	6LR_id	RAL dst
Inserted headers	RPI	--	--	--	--
Removed headers	--	--	--	--	RPI
Re-added headers	--	--	--	--	--
Modified headers	--	RPI	RPI	RPI	--
Untouched headers	--	--	--	--	--

Table 5: SM: Summary of the use of headers for RAL to RAL

7.3.2. SM: Example of Flow from RAL to RUL

In this case the flow comprises:

RAL src (6LN) --> 6LR_ia --> common parent (6LR_x) --> 6LR_id --> RUL
(IPv6 dst node)

For example, a communication flow could be: Node F --> Node D -->
Node B --> Node E --> Node G

6LR_ia are the intermediate routers from source (RAL) to the common parent (6LR_x) In this case, $1 \leq ia \leq n$, n is the number of routers (6LR) that the packet goes through from RAL to the common parent (6LR_x).

6LR_id (Node E) are the intermediate routers from the common parent (6LR_x) (Node B) to destination RUL (Node G). In this case, $1 \leq id \leq m$, m is the number of routers (6LR) that the packet goes through from the common parent (6LR_x) to destination RUL.

This situation is identical to the previous situation [Section 7.3.1](#)

The Table 6 summarizes what headers are needed for this use case.

Header	RAL src	6LR_ia	6LR_x(common parent)	6LR_id	RUL dst
Inserted headers	RPI	--	--	--	--
Removed headers	--	--	--	--	--
Re-added headers	--	--	--	--	--
Modified headers	--	RPI	RPI	RPI	--
Untouched headers	--	--	--	--	RPI(Ignored)

Table 6: SM: Summary of the use of headers for RAL to RUL

7.3.3. SM: Example of Flow from RUL to RAL

In this case the flow comprises:

RUL (IPv6 src node) --> 6LR_ia --> common parent (6LR_x) --> 6LR_id
--> RAL dst (6LN)

For example, a communication flow could be: Node G --> Node E -->
Node B --> Node D --> Node F

6LR_ia (Node E) are the intermediate routers from source (RUL) (Node G) to the common parent (6LR_x) (Node B). In this case, $1 \leq ia \leq n$, n is the number of routers (6LR) that the packet goes through from source to the common parent.

6LR_id (Node D) are the intermediate routers from the common parent (6LR_x) (Node B) to destination RAL (Node F). In this case, $1 \leq id \leq m$, m is the number of routers (6LR) that the packet goes through from the common parent (6LR_x) to the destination RAL.

The 6LR_ia ($ia=1$) (Node E) receives the packet from the RUL (Node G) and inserts the RPI encapsulated in a IPv6-in-IPv6 header. The IPv6-in-IPv6 header is addressed to the destination RAL (Node F).

The Figure 12 shows the table that summarizes what headers are needed for this use case.

Header	RUL	6LR_ia	Common	6LR_id	RAL
	src		Parent		dst
	node		(6LRx)		
Inserted	--	IP6-IP6(RPI)	--	--	--
headers					
Removed	--	--	--	--	IP6-IP6(RPI)
headers					
Re-added	--	--	--	--	--
headers					
Modified	--	--	IP6-IP6(RPI)	IP6-IP6(RPI)	--
headers					
Untouched	--	--	--	--	--
headers					

Figure 12: SM: Summary of the use of headers from RUL to RAL.

7.3.4. SM: Example of Flow from RUL to RUL

In this case the flow comprises:

RUL (IPv6 src node)--> 6LR_1--> 6LR_ia --> 6LBR --> 6LR_id --> RUL
(IPv6 dst node)

For example, a communication flow could be: Node G --> Node E -->
Node B --> Node A (root) --> Node C --> Node J

Internal nodes 6LR_ia (e.g: Node E or Node B) is the intermediate router from the RUL source (Node G) to the root (6LBR) (Node A). In this case, " $1 < ia \leq n$ ", n is the number of routers (6LR) that the packet goes through from the RUL to the root.

6LR_id (Node C) are the intermediate routers from the root (Node A) to the destination RUL dst node (Node J). In this case, " $1 \leq id \leq m$ ", m is the number of routers (6LR) that the packet goes through from the root to destination RUL.

The RPI is ignored at the RUL dst node.

The 6LR_1 (Node E) receives the packet from the RUL (Node G) and inserts the RPI (RPI), encapsulated in an IPv6-in-IPv6 header

directed to the root. The root removes the RPI and inserts a new RPI addressed to the 6LR father of the RUL.

The Figure 13 shows the table that summarizes what headers are needed for this use case.

Header	RUL	6LR_1	6LR_ia	6LBR	6LR_id	RUL
	src					dst
	node					node
Inserted headers	--	IP6-IP6 (RPI)	IP6-IP6 (RPI)	IP6-IP6 (RPI2)	IP6-IP6 (RPI2)	--
Removed headers	--	--	IP6-IP6 (RPI)	IP6-IP6 (RPI1)	IP6-IP6 (RPI2)	
Re-added headers	--	--	--	--	--	--
Modified headers	--	--				--
Untouched headers	--	--	--	--	--	--

Figure 13: SM: Summary of the use of headers from RUL to RUL

8. Non Storing mode

In Non Storing Mode (Non-SM) (fully source routed), the 6LBR (DODAG root) has complete knowledge about the connectivity of all DODAG nodes, and all traffic flows through the root node. Thus, there is no need for all nodes to know about the existence of RPL-unaware nodes. Only the 6LBR needs to act if compensation is necessary for not-RPL aware receivers.

The table (Figure 14) summarizes what headers are needed in the following scenarios, and indicates when the RPI, RH3 and IPv6-in-IPv6 header are to be inserted. It depicts the target destination address possible to a 6LN (indicated by "RAL"), to a 6LR (parent of a 6LN) or to the root. In cases where no IPv6-in-IPv6 header is needed, the column states as "No". There is no expectation on RPL that RPI can

be omitted, because it is needed for routing, quality of service and compression. This specification expects that is always a RPI Present.

The leaf can be a router 6LR or a host, both indicated as 6LN (Figure 6). In the table (Figure 14) the (1) indicates a 6tisch case [RFC8180], where the RPI may still be needed for the instanceID to be available for priority/channel selection at each hop.

Interaction between	Use Case	RPI	RH3	IPv6-in-IPv6	IPv6-in-IPv6 dst
Leaf - Root	RAL to root	Yes	No	No	No
	root to RAL	Yes	Yes	No	No
	root to RUL	Yes (1)	Yes	must	6LR
	RUL to root	Yes	No	must	root
Leaf - Internet	RAL to Int	Yes	No	No	No
	Int to RAL	Yes	Yes	must	RAL
	RUL to Int	Yes	No	must	root
	Int to RUL	Yes	Yes	must	6LR
Leaf - Leaf	RAL to RAL	Yes	Yes	must	root/RAL
	RAL to RUL	Yes	Yes	must	root/6LR
	RUL to RAL	Yes	Yes	must	root/RAL
	RUL to RUL	Yes	Yes	must	root/6LR

Figure 14: Table that shows headers needed in Non-Storing mode: RPI, RH3, IPv6-in-IPv6 encapsulation.

8.1. Non-Storing Mode: Interaction between Leaf and Root

In this section is described the communication flow in Non Storing Mode (Non-SM) between,

RAL to root

root to RAL

RUL to root

root to RUL

8.1.1. Non-SM: Example of Flow from RAL to root

In non-storing mode the leaf node uses default routing to send traffic to the root. The RPI must be included since it contains the rank information, which is used to avoid/detect loops.

RAL (6LN) --> 6LR_i --> root(6LBR)

For example, a communication flow could be: Node F --> Node D --> Node B --> Node A (root)

6LR_i are the intermediate routers from source to destination. In this case, " $1 \leq i \leq n$ ", n is the number of routers (6LR) that the packet goes through from source (RAL) to destination (6LBR).

This situation is the same case as storing mode.

The Table 7 summarizes what headers are needed for this use case.

Header	RAL src	6LR_i	6LBR dst
Inserted headers	RPI	--	--
Removed headers	--	--	RPI
Re-added headers	--	--	--
Modified headers	--	RPI	--
Untouched headers	--	--	--

Table 7: Non-SM: Summary of the use of headers from RAL to root

8.1.2. Non-SM: Example of Flow from root to RAL

In this case the flow comprises:

root (6LBR) --> 6LR_i --> RAL (6LN)

For example, a communication flow could be: Node A (root) --> Node B --> Node D --> Node F

6LR_i are the intermediate routers from source to destination. In this case, " $1 \leq i \leq n$ ", n is the number of routers (6LR) that the packet goes through from source (6LBR) to destination (RAL).

The 6LBR inserts an RH3, and a RPI. No IPv6-in-IPv6 header is necessary as the traffic originates with an RPL aware node, the 6LBR. The destination is known to be RPL-aware because the root knows the whole topology in non-storing mode.

The Table 8 summarizes what headers are needed for this use case.

Header	6LBR src	6LR _i	RAL dst
Inserted headers	RPI, RH3	--	--
Removed headers	--	--	RH3, RPI
Re-added headers	--	--	--
Modified headers	--	RPI, RH3	--
Untouched headers	--	--	--

Table 8: Non-SM: Summary of the use of headers from root to RAL

8.1.3. Non-SM: Example of Flow from root to RUL

In this case the flow comprises:

root (6LBR) --> 6LR_i --> RUL (IPv6 dst node)

For example, a communication flow could be: Node A (root) --> Node B --> Node E --> Node G

6LR_i are the intermediate routers from source to destination. In this case, " $1 \leq i \leq n$ ", n is the number of routers (6LR) that the packet goes through from source (6LBR) to destination (RUL).

In 6LBR the RH3 is added, it is modified at each intermediate 6LR (6LR₁ and so on) and it is fully consumed in the last 6LR (6LR_n), but left there. As the RPI is added, then the IPv6 node which does not understand the RPI, will ignore it (following [RFC8200](#)), thus encapsulation is not necessary.

The Figure 15 depicts the table that summarizes what headers are needed for this use case.

Header	6LBR	6LR _i i=(1,...,n-1)	6LR _n	IPv6 dst node (RUL)
Inserted headers	RPI, RH3	--	--	--
Removed headers	--	--		--
Re-added headers	--	--	--	--
Modified headers	--	RPI, RH3	RPI, RH3(consumed)	--
Untouched headers	--	--	--	RPI, RH3 (both ignored)

Figure 15: Non-SM: Summary of the use of headers from root to RUL

8.1.4. Non-SM: Example of Flow from RUL to root

In this case the flow comprises:

RUL (IPv6 src node) --> 6LR₁ --> 6LR_i --> root (6LBR) dst

For example, a communication flow could be: Node G --> Node E --> Node B --> Node A (root)

6LR_i are the intermediate routers from source to destination. In this case, " $1 \leq i \leq n$ ", n is the number of routers (6LR) that the packet goes through from source (RUL) to destination (6LBR). For example, 6LR₁ ($i=1$) is the router that receives the packets from the IPv6 node.

In this case the RPI is added by the first 6LR (6LR₁) (Node E), encapsulated in an IPv6-in-IPv6 header, and is modified in the following 6LRs. The RPI and the entire packet is consumed by the root.

The Figure 16 shows the table that summarizes what headers are needed for this use case.

+-----+-----+-----+-----+-----+-----+					
	RUL				
Header	src	6LR_1	6LR_i	6LBR	dst
	node				
+-----+-----+-----+-----+-----+-----+					
Inserted	--	IPv6-in-IPv6(RPI)	--	--	
headers					
+-----+-----+-----+-----+-----+-----+					
Removed	--	--	--	IPv6-in-IPv6(RPI)	
headers					
+-----+-----+-----+-----+-----+-----+					
Re-added	--	--	--	--	
headers					
+-----+-----+-----+-----+-----+-----+					
Modified	--	--	IPv6-in-IPv6(RPI)	--	
headers					
+-----+-----+-----+-----+-----+-----+					
Untouched	--	--	--	--	
headers					
+-----+-----+-----+-----+-----+-----+					

Figure 16: Non-SM: Summary of the use of headers from RUL to root

8.2. Non-Storing Mode: Interaction between Leaf and Internet

This section will describe the communication flow in Non Storing Mode (Non-SM) between:

RAL to Internet

Internet to RAL

RUL to Internet

Internet to RUL

8.2.1. Non-SM: Example of Flow from RAL to Internet

In this case the flow comprises:

RAL (6LN) src --> 6LR_i --> root (6LBR) --> Internet dst

For example, a communication flow could be: Node F --> Node D --> Node B --> Node A --> Internet

6LR_i are the intermediate routers from source to destination. In this case, " $1 \leq i \leq n$ ", n is the number of routers (6LR) that the packet goes through from source (RAL) to 6LBR.

This case is identical to storing-mode case.

The IPv6 flow label should be set to zero to aid in compression [RFC8138], and the 6LBR will set it to a non-zero value when sending towards the Internet [RFC6437].

The Table 9 summarizes what headers are needed for this use case.

Header	RAL src	6LR_i	6LBR	Internet dst
Inserted headers	RPI	--	--	--
Removed headers	--	--	--	--
Re-added headers	--	--	--	--
Modified headers	--	RPI	--	--
Untouched headers	--	--	RPI	RPI (Ignored)

Table 9: Non-SM: Summary of the use of headers from RAL to Internet

8.2.2. Non-SM: Example of Flow from Internet to RAL

In this case the flow comprises:

Internet --> root (6LBR) --> 6LR_i --> RAL dst (6LN)

For example, a communication flow could be: Internet --> Node A (root) --> Node B --> Node D --> Node F

6LR_i are the intermediate routers from source to destination. In this case, " $1 \leq i \leq n$ ", n is the number of routers (6LR) that the packet goes through from 6LBR to destination (RAL).

The 6LBR must add an RH3 header. As the 6LBR will know the path and address of the target node, it can address the IPv6-in-IPv6 header to that node. The 6LBR will zero the flow label upon entry in order to aid compression [RFC8138].

The Table 10 summarizes what headers are needed for this use case.

Header	Internet src	6LBR	6LR_i	RAL dst
Inserted headers	--	IPv6-in-IPv6 (RH3,RPI)	--	--
Removed headers	--	--	--	IPv6-in-IPv6 (RH3,RPI)
Re-added headers	--	--	--	--
Modified headers	--	--	IPv6-in-IPv6 (RH3,RPI)	--
Untouched headers	--	--	--	--

Table 10: Non-SM: Summary of the use of headers from Internet to RAL

8.2.3. Non-SM: Example of Flow from RUL to Internet

In this case the flow comprises:

RUL (IPv6 src node) --> 6LR_1 --> 6LR_i --> root (6LBR) --> Internet dst

For example, a communication flow could be: Node G --> Node E --> Node B --> Node A --> Internet

6LR_i are the intermediate routers from source to destination. In this case, " $1 \leq i \leq n$ ", n is the number of routers (6LR) that the packet goes through from source (RUL) to 6LBR, e.g. 6LR_1 ($i=1$).

In this case the flow label is recommended to be zero in the IPv6 node. As RPL headers are added in the IPv6 node packet, the first 6LR (6LR_1) will add a RPI inside a new IPv6-in-IPv6 header. The IPv6-in-IPv6 header will be addressed to the root. This case is identical to the storing-mode case (see [Section 7.2.3](#)).

The Figure 17 shows the table that summarizes what headers are needed for this use case.

Header	RUL	6LR_1	6LR_i	6LBR	Internet
	src		[i=2,...,n]		dst
	node				
Inserted headers	--	IP6-IP6(RPI)	--	--	--
Removed headers	--	--	--	IP6-IP6(RPI)	--
Re-added headers	--	--	--	--	--
Modified headers	--	--	IP6-IP6(RPI)	--	--
Untouched headers	--	--	--	--	--

Figure 17: Non-SM: Summary of the use of headers from RUL to Internet

8.2.4. Non-SM: Example of Flow from Internet to RUL

In this case the flow comprises:

Internet src --> root (6LBR) --> 6LR_i --> RUL (IPv6 dst node)

For example, a communication flow could be: Internet --> Node A (root) --> Node B --> Node E --> Node G

6LR_i are the intermediate routers from source to destination. In this case, " $1 \leq i \leq n$ ", n is the number of routers (6LR) that the packet goes through from 6LBR to RUL.

The 6LBR must add an RH3 header inside an IPv6-in-IPv6 header. The 6LBR will know the path, and will recognize that the final node is not an RPL capable node as it will have received the connectivity DAO from the nearest 6LR. The 6LBR can therefore make the IPv6-in-IPv6 header destination be the last 6LR. The 6LBR will set to zero the flow label upon entry in order to aid compression [[RFC8138](#)].

The Figure 18 shows the table that summarizes what headers are needed for this use case.

Header	Internet	6LBR	6LR_1	6LR_i (i=2,...,n)	RUL
	src				dst node
Inserted headers	--	IPv6-in-IPv6 (RH3,RPI)	--	--	--
Removed headers	--	--	--	IPv6-in-IPv6 (RH3,RPI)[1]	--
Re-added headers	--	--	--	--	--
Modified headers	--	--	IPv6-in-IPv6 (RH3,RPI)	IPv6-in-IPv6 (RH3,RPI)	--
Untouched headers	--	--	--	--	--

Figure 18: Non-SM: Summary of the use of headers from Internet to RUL
[1] The last 6LR before the IPv6 node.

8.3. Non-SM: Interaction between Leafs

In this section is described the communication flow in Non Storing Mode (Non-SM) between,

RAL to RAL

RAL to RUL

RUL to RAL

RUL to RUL

8.3.1. Non-SM: Example of Flow from RAL to RAL

In this case the flow comprises:

RAL src --> 6LR_ia --> root (6LBR) --> 6LR_id --> RAL dst

For example, a communication flow could be: Node F --> Node D -->
Node B --> Node A (root) --> Node B --> Node E --> Node H

6LR_{ia} are the intermediate routers from source to the root. In this case, $1 \leq ia \leq n$, n is the number of routers (6LR) that the packet goes through from RAL to the root.

6LR_{id} are the intermediate routers from the root to the destination. In this case, " $1 \leq ia \leq m$ ", m is the number of the intermediate routers (6LR).

This case involves only nodes in same RPL Domain. The originating node will add a RPI to the original packet, and send the packet upwards.

The originating node must put the RPI (RPI1) into an IPv6-in-IPv6 header addressed to the root, so that the 6LBR can remove that header. If it does not, then additional resources are wasted on the way down to carry the useless RPI.

The 6LBR will need to insert an RH3 header, which requires that it add an IPv6-in-IPv6 header. It should be able to remove the RPI(RPI1), as it was contained in an IPv6-in-IPv6 header addressed to it. Otherwise, there may be a RPI buried inside the inner IP header, which should get ignored. The root inserts a RPI (RPI2) alongside the RH3.

Networks that use the RPL P2P extension [[RFC6997](#)] are essentially non-storing DODAGs and fall into this scenario or scenario [Section 8.1.2](#), with the originating node acting as 6LBR.

The Figure 19 shows the table that summarizes what headers are needed for this use case.

Header	RAL src	6LR_ia	6LBR	6LR_id	RAL dst
Inserted headers	IPv6-in-IPv6 (RPI1)		IPv6-in-IPv6 (RH3-> RAL, RPI2)	--	--
Removed headers	--	--	IPv6-in-IPv6 (RPI1)	--	IPv6-in-IPv6 (RH3, RPI2)
Re-added headers	--	--	--	--	--
Modified headers	--	IP6-in-IP6 (RPI1)	--	IP6-in-IP6 (RPI2)	--
Untouched headers	--	--	--	--	--

Figure 19: Non-SM: Summary of the use of headers for RAL to RAL.
IP6-in-IP6 refers to IPv6-in-IPv6.

8.3.2. Non-SM: Example of Flow from RAL to RUL

In this case the flow comprises:

RAL --> 6LR_ia --> root (6LBR) --> 6LR_id --> RUL (IPv6 dst node)

For example, a communication flow could be: Node F --> Node D -->
Node B --> Node A (root) --> Node B --> Node E --> Node G

6LR_ia are the intermediate routers from source to the root In this case, $1 \leq ia \leq n$, n is the number of intermediate routers (6LR)

6LR_id are the intermediate routers from the root to the destination. In this case, " $1 \leq id \leq m$ ", m is the number of the intermediate routers (6LRs).

As in the previous case, the RAL (6LN) will insert a RPI (RPI_1) header which must be in an IPv6-in-IPv6 header addressed to the root so that the 6LBR can remove this RPI. The 6LBR will then insert an RH3 inside a new IPv6-in-IPv6 header addressed to the last 6LR_id (6LR_id = m).

The Figure 20 shows the table that summarizes what headers are needed for this use case.

Header	RAL src node	6LR_ia	6LBR	6LR_id	6LR_m	RUL dst node
Inserted headers	IP6-IP6 (RPI1)		IP6-IP6 (RH3, RPI2)	--	--	--
Removed headers	--	--	IP6-IP6 (RPI1)	--	IP6-IP6 (RH3, RPI2)	--
Re-added headers	--	--	--	--	--	--
Modified headers	--	IP6-IP6 (RPI1)	--	IP6-IP6 (RH3, RPI2)		--
Untouched headers	--	--	--	--	--	--

Figure 20: Non-SM: Summary of the use of headers from RAL to RUL.

8.3.3. Non-SM: Example of Flow from RUL to RAL

In this case the flow comprises:

RUL (IPv6 src node) --> 6LR_1 --> 6LR_ia --> root (6LBR) --> 6LR_id
--> RAL dst (6LN)

For example, a communication flow could be: Node G --> Node E -->
Node B --> Node A (root) --> Node B --> Node E --> Node H

6LR_ia are the intermediate routers from source to the root. In this case, $1 \leq ia \leq n$, n is the number of intermediate routers (6LR)

6LR_id are the intermediate routers from the root to the destination. In this case, " $1 \leq ia \leq m$ ", m is the number of the intermediate routers (6LR).

This scenario is mostly identical to the previous one. The RPI (RPI1) is added by the first 6LR (6LR_1) inside an IPv6-in-IPv6

header addressed to the root. The 6LBR will remove this RPI, and add it's own IPv6-in-IPv6 header containing an RH3 header and an RPI (RPI2).

The Figure 21 shows the table that summarizes what headers are needed for this use case.

Header	RUL src node	6LR_1	6LR_ia	6LBR	6LR_id	RAL dst node
Inserted headers	--	IP6-IP6 (RPI1)	--	IP6-IP6 (RH3, RPI2)	--	--
Removed headers	--		--	IP6-IP6 (RPI1)	--	IP6-IP6 (RH3, RPI2)
Re-added headers	--		--	--	--	--
Modified headers	--		IP6-IP6 (RPI1)	--	IP6-IP6 (RH3, RPI2)	--
Untouched headers	--		--	--	--	--

Figure 21: Non-SM: Summary of the use of headers from RUL to RAL.

8.3.4. Non-SM: Example of Flow from RUL to RUL

In this case the flow comprises:

RUL (IPv6 src node) --> 6LR_1 --> 6LR_ia --> root (6LBR) --> 6LR_id
--> RUL (IPv6 dst node)

For example, a communication flow could be: Node G --> Node E -->
Node B --> Node A (root) --> Node C --> Node J

6LR_ia are the intermediate routers from source to the root. In this case, $1 \leq ia \leq n$, n is the number of intermediate routers (6LR)

6LR_id are the intermediate routers from the root to the destination. In this case, " $1 \leq ia \leq m$ ", m is the number of the intermediate routers (6LR).

This scenario is the combination of the previous two cases.

The Figure 22 shows the table that summarizes what headers are needed for this use case.

Header	RUL	6LR_1	6LR_id	6LBR	6LR_id	6LR_m	RUL
	src						dst
	node						node
Inserted headers	--	IP6-IP6 (RPI1)	--	IP6-IP6 (RH3, RPI2)	--	--	--
Removed headers	--	--	--	IP6-IP6 (RPI1)	--	IP6-IP6 (RH3, RPI2)	--
Re-added headers	--	--	--	--	--	--	--
Modified headers	--	--	IP6-IP6 (RPI1)	--	IP6-IP6 (RH3, RPI2)	--	--
Untouched headers	--	--	--	--	--	--	--

Figure 22: Non-SM: Summary of the use of headers from RUL to RUL

9. Operational Considerations of supporting RUL-leaves

Roughly half of the situations described in this document involve leaf ("host") nodes that do not speak RPL. These nodes fall into two further categories: ones that drop a packet that have RPI or RH3 headers, and ones that continue to process a packet that has RPI and/or RH3 headers.

[RFC8200] provides for new rules that suggest that nodes that have not been configured (explicitly) to examine Hop-by-Hop headers, should ignore those headers, and continue processing the packet. Despite this, and despite the switch from 0x63 to 0x23, there may be hosts that are pre-RFC8200, or simply intolerant. Those hosts will

drop packets that continue to have RPL artifacts in them. In general, such hosts can not be easily supported in RPL LLNs.

There are some specific cases where it is possible to remove the RPL artifacts prior to forwarding the packet to the leaf host. The critical thing is that the artifacts have been inserted by the RPL root inside an IPv6-in-IPv6 header, and that the header has been addressed to the 6LR immediately prior to the leaf node. In that case, in the process of removing the IPv6-in-IPv6 header, the artifacts can also be removed.

The above case occurs whenever traffic originates from the outside the LLN (the "Internet" cases above), and non-storing mode is used. In non-storing mode, the RPL root knows the exact topology (as it must be create the RH3 header), and therefore knows what the 6LR prior to the leaf. For example, in Figure 5, node E is the 6LR prior to the leaf node G, or node C is the 6LR prior to the leaf node J.

traffic originating from the RPL root (such as when the data collection system is co-located on the RPL root), does not require an IPv6-in-IPv6 header (in either mode), as the packet is originating at the root, and the root can insert the RPI and RH3 headers directly into the packet, as it is formed. Such a packet is slightly smaller, but only can be sent to nodes (whether RPL aware or not), that will tolerate the RPL artifacts.

An operator that finds itself with a lot of traffic from the RPL root to RPL-not-aware-leaves, will have to do IPv6-in-IPv6 encapsulation if the leaf is not tolerant of the RPL artifacts. Such an operator could otherwise omit this unnecessary header if it was certain of the properties of the leaf.

As storing mode can not know the final path of the traffic, intolerant (that drop packets with RPL artifacts) leaf nodes can not be supported.

10. Operational considerations of introducing 0x23

This section describes the operational considerations of introducing the new RPI Option Type of 0x23.

During bootstrapping the node gets the DIO with the information of RPI Option Type, indicating the new RPI in the DODAG Configuration Option Flag. The DODAG root is in charge to configure the current network to the new value, through DIO messages and when all the nodes are set with the new value. The DODAG should change to a new DODAG version. In case of rebooting, the node does not remember the RPI

Option Type. Thus, the DIO is sent with a flag indicating the new RPI Option Type.

The DODAG Configuration option is contained in a RPL DIO message, which contains a unique DTSN counter. The leaf nodes respond to this message with DAO messages containing the same DTSN. This is a normal part of RPL routing; the RPL root therefore knows when the updated DODAG Configuration Option has been seen by all nodes.

Before the migration happens, all the RPL-aware nodes should support both values. The migration procedure is triggered when the DIO is sent with the flag indicating the new RPI Option Type. Namely, it remains at 0x63 until it is sure that the network is capable of 0x23, then it abruptly change to 0x23. This options allows to send packets to not-RPL nodes, which should ignore the option and continue processing the packets.

In case that a node join to a network that only process 0x63, it would produce a flag day as was mentioned previously. Indicating the new RPI in the DODAG Configuration Option Flag is a way to avoid the flag day in a network. It is recommended that a network process both options to enable interoperability.

11. IANA Considerations

This document updates the registration made in [\[RFC6553\]](#) Destination Options and Hop-by-Hop Options registry from 0x63 to 0x23 as shown in Figure 23.

Hex Value	Binary Value			Description	Reference
	act	chg	rest		
0x23	00	1	00011	RPL Option	[RFCXXXX] (*)
0x63	01	1	00011	RPL Option(DEPRECATED)	[RFC6553] [RFCXXXX] (*)

Figure 23: Option Type in RPL Option.(*)represents this document

DODAG Configuration option is updated as follows (Figure 24):

Bit number	Description	Reference
3	RPI 0x23 enable	This document

Figure 24: DODAG Configuration Option Flag to indicate the RPI-flag-day.

12. Security Considerations

The security considerations covered in [\[RFC6553\]](#) and [\[RFC6554\]](#) apply when the packets are in the RPL Domain.

The IPv6-in-IPv6 mechanism described in this document is much more limited than the general mechanism described in [\[RFC2473\]](#). The willingness of each node in the LLN to decapsulate packets and forward them could be exploited by nodes to disguise the origin of an attack.

While a typical LLN may be a very poor origin for attack traffic (as the networks tend to be very slow, and the nodes often have very low duty cycles) given enough nodes, they could still have a significant impact, particularly if attack is targeting another LLN. Additionally, some uses of RPL involve large backbone ISP scale equipment [\[I-D.ietf-anima-autonomic-control-plane\]](#), which may be equipped with multiple 100Gb/s interfaces.

Blocking or careful filtering of IPv6-in-IPv6 traffic entering the LLN as described above will make sure that any attack that is mounted must originate from compromised nodes within the LLN. The use of [BCP38](#) [\[BCP38\]](#) filtering at the RPL root on egress traffic will both alert the operator to the existence of the attack, as well as drop the attack traffic. As the RPL network is typically numbered from a single prefix, which is itself assigned by RPL, [BCP38](#) filtering involves a single prefix comparison and should be trivial to automatically configure.

There are some scenarios where IPv6-in-IPv6 traffic should be allowed to pass through the RPL root, such as the IPv6-in-IPv6 mediated communications between a new Pledge and the Join Registrar/Coordinator (JRC) when using [\[I-D.ietf-anima-bootstrapping-keyinfra\]](#) and [\[I-D.ietf-6tisch-dtsecurity-secure-join\]](#). This is the case for the RPL root to do careful filtering: it occurs only when the Join Coordinator is not co-located inside the RPL root.

With the above precautions, an attack using IPv6-in-IPv6 tunnels can only be by a node within the LLN on another node within the LLN. Such an attack could, of course, be done directly. An attack of this kind is meaningful only if the source addresses are either fake or if the point is to amplify return traffic. Such an attack, could also be done without the use of IPv6-in-IPv6 headers using forged source addresses. If the attack requires bi-directional communication, then IPv6-in-IPv6 provides no advantages.

Whenever IPv6-in-IPv6 headers are being proposed, there is a concern about creating security issues. In the security section of [\[RFC2473\]](#), it was suggested that tunnel entry and exit points can be secured, via "Use IPsec". This recommendation is not practical for RPL networks. [\[RFC5406\]](#) goes into some detail on what additional details would be needed in order to "Use IPsec". Use of ESP would prevent [RFC8183](#) compression (compression must occur before encryption), and [RFC8183](#) compression is lossy in a way that prevents use of AH. These are minor issues. The major issue is how to establish trust enough such that IKEv2 could be used. This would require a system of certificates to be present in every single node, including any Internet nodes that might need to communicate with the LLN. Thus, "Use IPsec" requires a global PKI in the general case.

More significantly, the use of IPsec tunnels to protect the IPv6-in-IPv6 headers would in the general case scale with the square of the number of nodes. This is a lot of resource for a constrained nodes on a constrained network. In the end, the IPsec tunnels would be providing only [BCP38](#)-like origin authentication! That is, IPsec provides a transitive guarantee to the tunnel exit point that the tunnel entry point did [BCP38](#) on traffic going in. Just doing [BCP38](#) origin filtering at the entry and exit of the LLN provides a similar level amount of security without all the scaling and trust problems of using IPsec as [RFC2473](#) suggested. IPsec is not recommended.

An LLN with hostile nodes within it would not be protected against impersonation with the LLN by entry/exit filtering.

The RH3 header usage described here can be abused in equivalent ways (to disguise the origin of traffic and attack other nodes) with an IPv6-in-IPv6 header to add the needed RH3 header. As such, the attacker's RH3 header will not be seen by the network until it reaches the end host, which will decapsulate it. An end-host should be suspicious about a RH3 header which has additional hops which have not yet been processed, and SHOULD ignore such a second RH3 header.

In addition, the LLN will likely use [\[RFC8138\]](#) to compress the IPv6-in-IPv6 and RH3 headers. As such, the compressor at the RPL-root will see the second RH3 header and MAY choose to discard the packet

if the RH3 header has not been completely consumed. A consumed (inert) RH3 header could be present in a packet that flows from one LLN, crosses the Internet, and enters another LLN. As per the discussion in this document, such headers do not need to be removed. However, there is no case described in this document where an RH3 is inserted in a non-storing network on traffic that is leaving the LLN, but this document should not preclude such a future innovation. It should just be noted that an incoming RH3 must be fully consumed, or very carefully inspected.

The RPI, if permitted to enter the LLN, could be used by an attacker to change the priority of a packet by selecting a different RPLInstanceID, perhaps one with a higher energy cost, for instance. It could also be that not all nodes are reachable in an LLN using the default instanceID, but a change of instanceID would permit an attacker to bypass such filtering. Like the RH3, a RPI is to be inserted by the RPL root on traffic entering the LLN by first inserting an IPv6-in-IPv6 header. The attacker's RPI therefore will not be seen by the network. Upon reaching the destination node the RPI has no further meaning and is just skipped; the presence of a second RPI will have no meaning to the end node as the packet has already been identified as being at it's final destination.

The RH3 and RPIs could be abused by an attacker inside of the network to route packets on non-obvious ways, perhaps eluding observation. This usage is in fact part of [\[RFC6997\]](#) and can not be restricted at all. This is a feature, not a bug.

[\[RFC7416\]](#) deals with many other threats to LLNs not directly related to the use of IPv6-in-IPv6 headers, and this document does not change that analysis.

Nodes within the LLN can use the IPv6-in-IPv6 mechanism to mount an attack on another part of the LLN, while disguising the origin of the attack. The mechanism can even be abused to make it appear that the attack is coming from outside the LLN, and unless countered, this could be used to mount a Distributed Denial Of Service attack upon nodes elsewhere in the Internet. See [\[DDOS-KREBS\]](#) for an example of such attacks already seen in the real world.

If an attack comes from inside of LLN, it can be alleviated with SAVI (Source Address Validation Improvement) using [\[RFC8505\]](#) with [\[I-D.ietf-6lo-ap-nd\]](#). The attacker will not be able to source traffic with an address that is not registered, and the registration process checks for topological correctness. Notice that there is an L2 authentication in most of the cases. If an attack comes from outside LLN IPv6-in- IPv6 can be used to hide inner routing headers, but by construction, the RH3 can typically only address nodes within

the LLN. That is, a RH3 with a CmprI less than 8 , should be considered an attack (see [RFC6554, section 3](#)).

Nodes outside of the LLN will need to pass IPv6-in-IPv6 traffic through the RPL root to perform this attack. To counter, the RPL root SHOULD either restrict ingress of IPv6-in-IPv6 packets (the simpler solution), or it SHOULD walk the IP header extension chain until it can inspect the upper-layer-payload as described in [\[RFC7045\]](#). In particular, the RPL root SHOULD do [\[BCP38\]](#) processing on the source addresses of all IP headers that it examines in both directions.

Note: there are some situations where a prefix will spread across multiple LLNs via mechanisms such as the one described in [\[I-D.ietf-6lo-backbone-router\]](#). In this case the [BCP38](#) filtering needs to take this into account, either by exchanging detailed routing information on each LLN, or by moving the [BCP38](#) filtering further towards the Internet, so that the details of the multiple LLNs do not matter.

[13. Acknowledgments](#)

This work is done thanks to the grant given by the StandICT.eu project.

A special BIG thanks to C. M. Heard for the help with the [Section 4](#). Much of the redaction in that section is based on his comments.

Additionally, the authors would like to acknowledge the review, feedback, and comments of (alphabetical order): Robert Cragie, Simon Duquennoy, Ralph Droms, Cenk Guendogan, Rahul Jadhav, Benjamin Kaduk, Matthias Kovatsch, Charlie Perkins, Alvaro Retana, Peter van der Stok, Xavier Vilajosana, Eric Vyncke and Thomas Watteyne.

[14. References](#)

[14.1. Normative References](#)

- [BCP38] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", [BCP 38](#), [RFC 2827](#), DOI 10.17487/RFC2827, May 2000, <<https://www.rfc-editor.org/info/bcp38>>.
- [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>>.

- [RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", [RFC 6040](#), DOI 10.17487/RFC6040, November 2010, <<https://www.rfc-editor.org/info/rfc6040>>.
- [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>>.
- [RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", [RFC 6550](#), DOI 10.17487/RFC6550, March 2012, <<https://www.rfc-editor.org/info/rfc6550>>.
- [RFC6553] Hui, J. and JP. Vasseur, "The Routing Protocol for Low-Power and Lossy Networks (RPL) Option for Carrying RPL Information in Data-Plane Datagrams", [RFC 6553](#), DOI 10.17487/RFC6553, March 2012, <<https://www.rfc-editor.org/info/rfc6553>>.
- [RFC6554] Hui, J., Vasseur, JP., Culler, D., and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", [RFC 6554](#), DOI 10.17487/RFC6554, March 2012, <<https://www.rfc-editor.org/info/rfc6554>>.
- [RFC7045] Carpenter, B. and S. Jiang, "Transmission and Processing of IPv6 Extension Headers", [RFC 7045](#), DOI 10.17487/RFC7045, December 2013, <<https://www.rfc-editor.org/info/rfc7045>>.
- [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>>.
- [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>>.

- [RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, [RFC 8200](#), DOI 10.17487/RFC8200, July 2017, <<https://www.rfc-editor.org/info/rfc8200>>.
- [RFC8504] Chown, T., Loughney, J., and T. Winters, "IPv6 Node Requirements", [BCP 220](#), [RFC 8504](#), DOI 10.17487/RFC8504, January 2019, <<https://www.rfc-editor.org/info/rfc8504>>.

14.2. Informative References

- [DDOS-KREBS]
Goodin, D., "Record-breaking DDoS reportedly delivered by >145k hacked cameras", September 2016, <<http://arstechnica.com/security/2016/09/botnet-of-145k-cameras-reportedly-deliver-internets-biggest-ddos-ever/>>.
- [I-D.ietf-6lo-ap-nd]
Thubert, P., Sarikaya, B., Sethi, M., and R. Struik, "Address Protected Neighbor Discovery for Low-power and Lossy Networks", [draft-ietf-6lo-ap-nd-12](#) (work in progress), April 2019.
- [I-D.ietf-6lo-backbone-router]
Thubert, P., Perkins, C., and E. Levy-Abegnoli, "IPv6 Backbone Router", [draft-ietf-6lo-backbone-router-13](#) (work in progress), September 2019.
- [I-D.ietf-6tisch-dtsecurity-secure-join]
Richardson, M., "6tisch Secure Join protocol", [draft-ietf-6tisch-dtsecurity-secure-join-01](#) (work in progress), February 2017.
- [I-D.ietf-anima-autonomic-control-plane]
Eckert, T., Behringer, M., and S. Bjarnason, "An Autonomic Control Plane (ACP)", [draft-ietf-anima-autonomic-control-plane-21](#) (work in progress), November 2019.
- [I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Eckert, T., Behringer, M., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", [draft-ietf-anima-bootstrapping-keyinfra-30](#) (work in progress), November 2019.
- [I-D.ietf-intarea-tunnels]
Touch, J. and M. Townsley, "IP Tunnels in the Internet Architecture", [draft-ietf-intarea-tunnels-10](#) (work in progress), September 2019.

[I-D.ietf-roll-unaware-leaves]

Thubert, P. and M. Richardson, "Routing for RPL Leaves", [draft-ietf-roll-unaware-leaves-07](#) (work in progress), November 2019.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", [RFC 2460](#), DOI 10.17487/RFC2460, December 1998, <<https://www.rfc-editor.org/info/rfc2460>>.

[RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", [RFC 2473](#), DOI 10.17487/RFC2473, December 1998, <<https://www.rfc-editor.org/info/rfc2473>>.

[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>>.

[RFC5406] Bellovin, S., "Guidelines for Specifying the Use of IPsec Version 2", [BCP 146](#), [RFC 5406](#), DOI 10.17487/RFC5406, February 2009, <<https://www.rfc-editor.org/info/rfc5406>>.

[RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", [RFC 6437](#), DOI 10.17487/RFC6437, November 2011, <<https://www.rfc-editor.org/info/rfc6437>>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", [RFC 6775](#), DOI 10.17487/RFC6775, November 2012, <<https://www.rfc-editor.org/info/rfc6775>>.

[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>>.
- [RFC8180] Vilajosana, X., Ed., Pister, K., and T. Watteyne, "Minimal IPv6 over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration", [BCP 210](#), [RFC 8180](#), DOI 10.17487/RFC8180, May 2017, <<https://www.rfc-editor.org/info/rfc8180>>.
- [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>>.

Authors' Addresses

Maria Ines Robles
Aalto University, Finland

Email: mariainesrobles@gmail.com

Michael C. Richardson
Sandelman Software Works
470 Dawson Avenue
Ottawa, ON K1Z 5V7
CA

Email: mcr+ietf@sandelman.ca
URI: <http://www.sandelman.ca/mcr/>

Pascal Thubert
Cisco Systems, Inc
Building D
45 Allée des Ormes - BP1200
MOUGINS - Sophia Antipolis 06254
FRANCE

Phone: +33 497 23 26 34
Email: pthubert@cisco.com

