

ROLL Working Group  
Internet-Draft  
Updates: [6553](#), [6550](#), [8138](#) (if approved)  
Intended status: Standards Track  
Expires: November 16, 2019

M. Robles  
Aalto  
M. Richardson  
SSW  
P. Thubert  
Cisco  
May 15, 2019

Using RPL Option Type, Routing Header for Source Routes and IPv6-in-IPv6  
encapsulation in the RPL Data Plane  
[draft-ietf-roll-useofrplinfo-26](#)

## 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 [RFC 6553](#) (RPL Option Type), [RFC 6554](#) (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 [RFC 6553](#) adding a change to the RPL Option Type. Additionally, this document updates [RFC 6550](#) 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 November 16, 2019.

Internet-Draft

RPL-data-plane

May 2019

## 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

<a href="#">1.</a>	Introduction . . . . .	<a href="#">3</a>
<a href="#">1.1.</a>	Overview . . . . .	<a href="#">4</a>
<a href="#">2.</a>	Terminology and Requirements Language . . . . .	<a href="#">5</a>
<a href="#">3.</a>	Updates to <a href="#">RFC6553</a> , <a href="#">RFC6550</a> and <a href="#">RFC8138</a> . . . . .	<a href="#">6</a>
<a href="#">3.1.</a>	Updates to <a href="#">RFC6553</a> : Indicating the new RPI value. . . . .	<a href="#">6</a>
<a href="#">3.2.</a>	Updates to <a href="#">RFC6550</a> : Indicating the new RPI in the DODAG Configuration Option Flag. . . . .	<a href="#">10</a>
<a href="#">3.3.</a>	Updates to <a href="#">RFC8138</a> : Indicating the way to decompress with the new RPI value. . . . .	<a href="#">11</a>
<a href="#">4.</a>	Sample/reference topology . . . . .	<a href="#">11</a>
<a href="#">5.</a>	Use cases . . . . .	<a href="#">14</a>
<a href="#">6.</a>	Storing mode . . . . .	<a href="#">16</a>
<a href="#">6.1.</a>	Storing Mode: Interaction between Leaf and Root . . . . .	<a href="#">17</a>
<a href="#">6.1.1.</a>	SM: Example of Flow from RPL-aware-leaf to root . . . . .	<a href="#">17</a>
<a href="#">6.1.2.</a>	SM: Example of Flow from root to RPL-aware-leaf . . . . .	<a href="#">19</a>
<a href="#">6.1.3.</a>	SM: Example of Flow from root to not-RPL-aware-leaf . . . . .	19
<a href="#">6.1.4.</a>	SM: Example of Flow from not-RPL-aware-leaf to root . . . . .	20
<a href="#">6.2.</a>	Storing Mode: Interaction between Leaf and Internet. . . . .	<a href="#">21</a>
<a href="#">6.2.1.</a>	SM: Example of Flow from RPL-aware-leaf to Internet . . . . .	21
<a href="#">6.2.2.</a>	SM: Example of Flow from Internet to RPL-aware-leaf . . . . .	22
<a href="#">6.2.3.</a>	SM: Example of Flow from not-RPL-aware-leaf to Internet . . . . .	<a href="#">23</a>
<a href="#">6.2.4.</a>	SM: Example of Flow from Internet to not-RPL-aware- leaf. . . . .	<a href="#">24</a>
<a href="#">6.3.</a>	Storing Mode: Interaction between Leaf and Leaf . . . . .	<a href="#">25</a>
<a href="#">6.3.1.</a>	SM: Example of Flow from RPL-aware-leaf to RPL-aware-	

leaf . . . . .	<a href="#">25</a>
6.3.2. SM: Example of Flow from RPL-aware-leaf to not-RPL-aware-leaf . . . . .	<a href="#">26</a>
6.3.3. SM: Example of Flow from not-RPL-aware-leaf to RPL-aware-leaf . . . . .	<a href="#">27</a>

6.3.4. SM: Example of Flow from not-RPL-aware-leaf to not-RPL-aware-leaf . . . . .	<a href="#">29</a>
<a href="#">7.</a> Non Storing mode . . . . .	<a href="#">30</a>
<a href="#">7.1.</a> Non-Storing Mode: Interaction between Leaf and Root . . . . .	<a href="#">31</a>
7.1.1. Non-SM: Example of Flow from RPL-aware-leaf to root . . . . .	<a href="#">32</a>
7.1.2. Non-SM: Example of Flow from root to RPL-aware-leaf . . . . .	<a href="#">32</a>
7.1.3. Non-SM: Example of Flow from root to not-RPL-aware-leaf . . . . .	<a href="#">33</a>
7.1.4. Non-SM: Example of Flow from not-RPL-aware-leaf to root . . . . .	<a href="#">34</a>
7.2. Non-Storing Mode: Interaction between Leaf and Internet . . . . .	<a href="#">35</a>
7.2.1. Non-SM: Example of Flow from RPL-aware-leaf to Internet . . . . .	<a href="#">35</a>
7.2.2. Non-SM: Example of Flow from Internet to RPL-aware-leaf . . . . .	<a href="#">36</a>
7.2.3. Non-SM: Example of Flow from not-RPL-aware-leaf to Internet . . . . .	<a href="#">37</a>
7.2.4. Non-SM: Example of Flow from Internet to not-RPL-aware-leaf . . . . .	<a href="#">38</a>
<a href="#">7.3.</a> Non-Storing Mode: Interaction between Leafs . . . . .	<a href="#">39</a>
7.3.1. Non-SM: Example of Flow from RPL-aware-leaf to RPL-aware-leaf . . . . .	<a href="#">39</a>
7.3.2. Non-SM: Example of Flow from RPL-aware-leaf to not-RPL-aware-leaf . . . . .	<a href="#">41</a>
7.3.3. Non-SM: Example of Flow from not-RPL-aware-leaf to RPL-aware-leaf . . . . .	<a href="#">42</a>
7.3.4. Non-SM: Example of Flow from not-RPL-aware-leaf to not-RPL-aware-leaf . . . . .	<a href="#">43</a>
8. Operational Considerations of supporting not-RPL-aware-leaves . . . . .	<a href="#">44</a>
<a href="#">9.</a> Operational considerations of introducing 0x23 . . . . .	<a href="#">45</a>
<a href="#">10.</a> IANA Considerations . . . . .	<a href="#">46</a>
<a href="#">11.</a> Security Considerations . . . . .	<a href="#">47</a>
<a href="#">12.</a> Acknowledgments . . . . .	<a href="#">50</a>
<a href="#">13.</a> References . . . . .	<a href="#">50</a>
<a href="#">13.1.</a> Normative References . . . . .	<a href="#">50</a>

[13.2. Informative References](#) . . . . . [51](#)  
Authors' Addresses . . . . . [54](#)

[1.](#) Introduction

RPL (IPv6 Routing Protocol for Low-Power and Lossy Networks) [[RFC6550](#)] is a routing protocol for constrained networks. [RFC 6553](#) [[RFC6553](#)] defines the "RPL option" (RPI), carried within the IPv6 Hop-by-Hop header to quickly identify inconsistencies (loops) in the routing topology. [RFC 6554](#) [[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 RPI option value to make [RFC8200](#) routers ignore this option by default.

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 [\[I-D.ietf-intarea-tunnels\]](#).

## [1.1.](#) Overview

The rest of the document is organized as follows: [Section 2](#) describes the used terminology. [Section 3](#) describes the updates to [RFC6553](#), [RFC6550](#) and [RFC 8138](#). [Section 4](#) provides the reference topology used for the uses cases. [Section 5](#) describes the uses cases included. [Section 6](#) describes the storing mode cases and [section 7](#) the non-storing mode cases. [Section 8](#) describes the operational considerations of supporting not-RPL-aware-leaves. [Section 9](#) depicts operational considerations for the proposed change on RPL Option type, [section 10](#) the IANA considerations and then [section 11](#) describes the security aspects.

Robles, et al.

Expires November 16, 2019

[Page 4]

---

Internet-Draft

RPL-data-plane

May 2019

## [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: LBR, LLN, RPL, RPL Domain and ROLL.

**RPL-node:** A device which implements RPL, thus the device is RPL-aware. Please note that the device can be found inside the LLN or outside LLN. In this document a RPL-node which is a leaf of a (Destination Oriented Directed Acyclic Graph) DODAG is called RPL-aware-leaf (Raf).

**RPL-not-capable:** 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. In this document a not-RPL-aware node which is a leaf of a DODAG is called not-RPL-aware-leaf (~Raf).

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.

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."

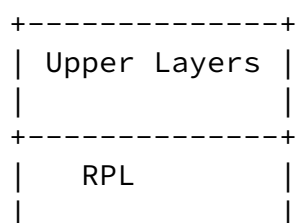
6LBR: [[RFC6775](#)] defines it as:"A border router located at the junction of separate 6LoWPAN networks or between a 6LoWPAN network and another IP network. There may be one or more 6LBRs at the 6LoWPAN network boundary. A 6LBR is the responsible authority for IPv6 prefix propagation for the 6LoWPAN 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 RPL Option Type. Thus the network does not work correctly.

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.

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.



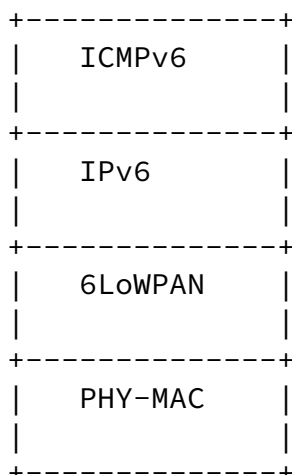


Figure 1: RPL Stack.

RPL supports two modes of Downward traffic: in storing mode (RPL-SM), it is fully stateful; in non-storing mode (RPL-NSM), 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.

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

3.1. Updates to [RFC6553](#): Indicating the new RPI value.

This modification is required to be able to send, for example, IPv6 packets from a RPL-aware-leaf to a not-RPL-aware node through Internet (see [Section 6.2.1](#)), 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 header, 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 Value	Binary Value			Description	Reference
	act	chg	rest		
----- 0x63	01	1	00011	RPL Option	[ <a href="#">RFC6553</a> ]

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 IP-in-IP 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 IP-in-IP 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". This means that while it should be avoided, the impact on the Internet of leaking a Hop-by-Hop header is acceptable.

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 field 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 remaining bits serve as the option type and remain as 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 [RFC6553] RPL Hop-by-Hop option known as RPI.

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 RPL-aware-leaf to Internet (see Section 6.2.1).

This is a significant update to [RFC6553].

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

Figure 3: Revised Option Type in RPL Option.

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

When forwarding packets, implementations SHOULD use the same value as it was received (This is required because, RPI type code can not be

---

changed by [[RFC8200](#)]). 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.

A network which is switching from straight 6lowpan compression mechanism to those described in [[RFC8138](#)] will experience a flag day in the data compression anyway, and if possible this change can be deployed at the same time.

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 option 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 RPI 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.

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](#)].

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 IPv6-in-IPv6 headers for traffic originating from the root to 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.

### 3.2. 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 (0x23) and old RPI (0x63) nodes, this section defines a flag in the DIO Configuration Option, to indicate when then new RPI value can be safely used. This means, the flag is going to indicate the type of RPI that the network is using. 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 RPI. A node that forwards a packet with an RPI MUST not modify the option type of the RPI.

This is done via a DODAG Configuration Option flag which will propagate through the network. 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 (0x63) value.

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.

The DODAG Configuration Option has a Flag field which is modified by this document. 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".

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 if the flag is set, so DIO messages would be set with the flag unset until a DIO is received with the flag set.

**3.3.** Updates to [RFC8138](#): Indicating the way to decompress with the new RPI value.

This modification is required to be able to decompress the RPL RPI option with the new value (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 RPL 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 3.2](#). 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 RPL-aware-leaf to not-RPL-aware-leaf and not-RPL-aware-leaf to not-RPL-aware-leaf comprise an IPv6-in-IPv6 and RPI compression headers. The use of the IPv6-in-IPv6 header is MANDATORY in this case, and it SHOULD be compressed with [[RFC8138](#)] [section 7](#). As exemplification of compressing the RPI, section A.1 of [[RFC8138](#)] illustrates the case in Storing mode where the packet is received from the Internet, then the root encapsulates the packet to insert the RPI. The result is shown in Figure 5.

++ ... -+-+...-+-+-- ... -+-+--+ ... -+-+ ... -+-+ ... -+-+...

```

|11110001| RPI- | IP-in-IP | NH=1          |11110CPP| Compressed | UDP
|Page 1  | 6LoRH | 6LoRH  | LOWPAN_IPHC | UDP    | UDP header | Payld
+-+ ... +-+...+-+... +-+... +-+... +-+... +-+... +-+... +-+... +-+...

```

Figure 5: RPI Inserted by the Root in Storing Mode

#### 4. Sample/reference topology

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

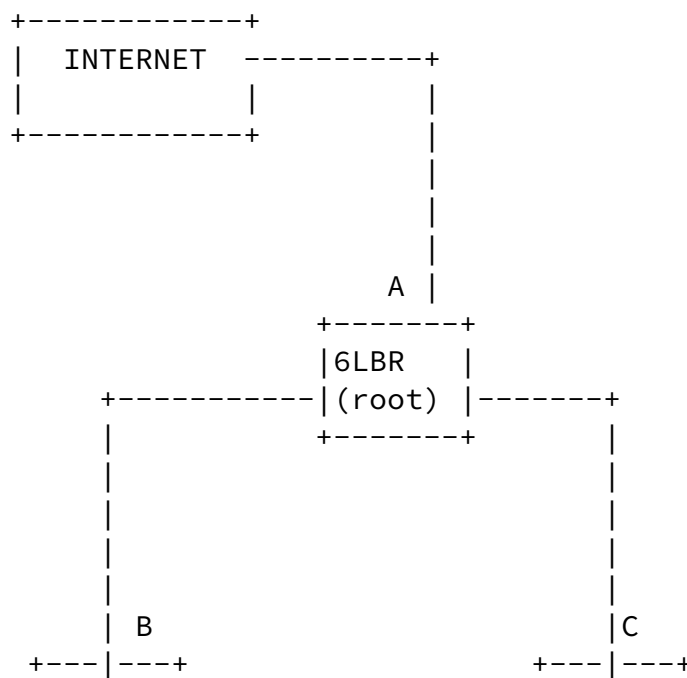
Figure 6 shows the reference RPL Topology for this document. The letters above the nodes are there so that they may be referenced in subsequent sections. In the figure, 6LR represents a full router

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, thus the 6BBR is not present in the figure.

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

The leafs marked as ~Raf (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.



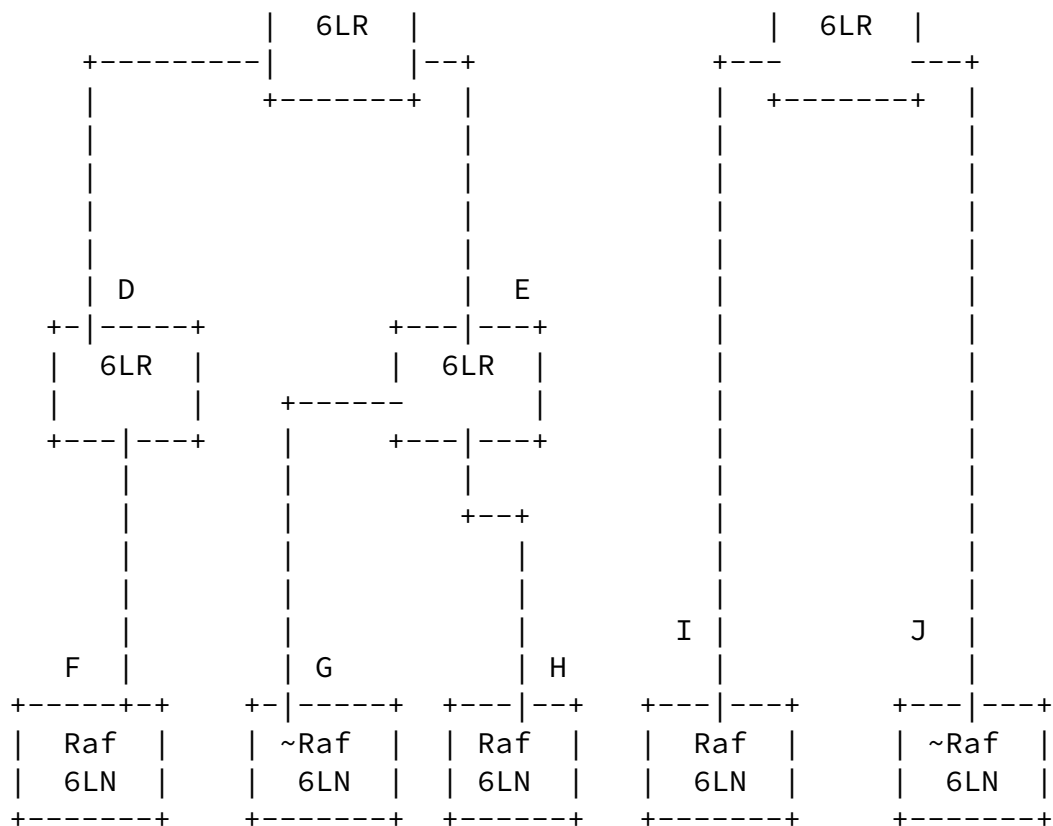


Figure 6: A reference RPL Topology.

## 5. 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 (0x23).

The uses cases describe the communication between RPL-aware-nodes, with the root (6LBR), and with Internet. This document also

describes the communication between nodes acting as leaves that do not understand RPL (~Raf nodes), but are part of the LLN. (e.g. [Section 6.1.4](#) Flow from not-RPL-aware-leaf to root) This document depicts as well the communication inside of the LLN when it has the final destination addressed outside of the LLN e.g. with destination to Internet. For example, [Section 6.2.3](#) Flow from not-RPL-aware-leaf to Internet

The uses cases comprise as follow:

Interaction between Leaf and Root:

RPL-aware-leaf(Raf) to root  
root to RPL-aware-leaf(Raf)  
not-RPL-aware-leaf(~Raf) to root  
root to not-RPL-aware-leaf(~Raf)

Interaction between Leaf and Internet:

RPL-aware-leaf(Raf) to Internet  
Internet to RPL-aware-leaf(Raf)  
not-RPL-aware-leaf(~Raf) to Internet  
Internet to not-RPL-aware-leaf(~Raf)

Interaction between Leafs:

RPL-aware-leaf(Raf) to RPL-aware-leaf(Raf) (storing and non-storing)  
RPL-aware-leaf(Raf) to not-RPL-aware-leaf(~Raf) (non-storing)

not-RPL-aware-leaf(~Raf) to RPL-aware-leaf(Raf) (storing and non-storing)

not-RPL-aware-leaf(~Raf) to not-RPL-aware-leaf(~Raf) (non-storing)



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](#)]. Extensions headers may not be added or removed except by the sender or the receiver.

However, unlike [[RFC6553](#)], the Hop-by-Hop Option Header used for the RPI artifact has the first two bits set to '00'. This means that the RPI artifact will be ignored when received by a host or router that does not understand that option ( [Section 4.2 \[\[RFC8200\]\(#\)\]](#)).

This means that when the no-drop RPI option code 0x23 is used, a packet that leaves the RPL domain of an LLN (or that leaves the LLN entirely) will not be discarded when it contains the [[RFC6553](#)] RPL Hop-by-Hop option known as RPI. Thus, the RPI Hop-by-Hop option is left in place even if the end host does not understand it.

NOTE: No clear attack has been described when the RPI information is released to the Internet. At a minimum, it is clear that the RPI option would waste some network bandwidth when it escapes. This is traded off against the savings in the LLN by not having to encapsulate the packet in order to remove the artifact. Please check the Security Considerations sections [Section 11](#) for further details.

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 RPI 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 RPI 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 RPI and 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.

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.

## 6. Storing mode

In storing mode (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 Raf node that is the target(tgt)), to the "root" or if a hop-by-hop header must be added (indicated by "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 headers are 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<sub>i</sub> is the intermediate router 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).

Internet-Draft

RPL-data-plane

May 2019

Interaction between	Use Case	IPv6-in-IPv6	IPv6-in-IPv6 dst
Leaf - Root	Raf to root	No	No
	root to Raf	No	No
	root to ~Raf	No	No
	~Raf to root	must	root
Leaf - Internet	Raf to Int	No	No
	Int to Raf	must	Raf (tgt)
	~Raf to Int	must	root
	Int to ~Raf	must	hop
Leaf - Leaf	Raf to Raf	No	No
	Raf to ~Raf	No	No
	~Raf to Raf	must	Raf (tgt)
	~Raf to ~Raf	must	~Raf

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

### 6.1. Storing Mode: Interaction between Leaf and Root

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

RPL-aware-leaf to root

root to RPL-aware-leaf

not-RPL-aware-leaf to root

root to not-RPL-aware-leaf

### [6.1.1.](#) SM: Example of Flow from RPL-aware-leaf to root

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

Robles, et al.

Expires November 16, 2019

[Page 17]

---

Internet-Draft

RPL-data-plane

May 2019

As stated in [Section 16.2 of \[RFC6550\]](#) a RPL-aware-leaf node does not generally issue DIO messages; a leaf node accepts DIO messages from upstream. (When the inconsistency in routing occurs, a leaf node will generate a DIO with an infinite rank, to fix it). It may issue DAO and DIS messages though it generally ignores DAO and DIS messages.

In this case the flow comprises:

RPL-aware-leaf (6LN) --> 6LR\_i --> root(6LBR)

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

As it was mentioned in this document 6LRs, 6LBR are always full-fledged RPL routers.

The 6LN (Node F) inserts the RPI header, and sends the packet to 6LR (Node E) which decrements the rank in 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 header can be removed by the 6LBR because the packet is addressed to the 6LBR. The 6LN must know that it is communicating with the 6LBR to make use of this scenario. The 6LN 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	6LN src	6LR_i	6LBR dst
Inserted headers	RPI	--	--
Removed headers	--	--	RPI
Re-added headers	--	--	--
Modified headers	--	RPI	--
Untouched headers	--	--	--

Table 1: Storing: Summary of the use of headers from RPL-aware-leaf to root

### [6.1.2.](#) SM: Example of Flow from root to RPL-aware-leaf

In this case the flow comprises:

root (6LBR) --> 6LR\_i --> RPL-aware-leaf (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 header 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 6LN 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	6LR_i	6LN
Inserted headers	RPI	--	--
Removed headers	--	--	RPI
Re-added headers	--	--	--
Modified headers	--	RPI	--
Untouched headers	--	--	--

+-----+-----+-----+-----+

Table 2: Storing: Summary of the use of headers from root to RPL-aware-leaf

[6.1.3.](#) SM: Example of Flow from root to not-RPL-aware-leaf

In this case the flow comprises:

root (6LBR) --> 6LR\_i --> not-RPL-aware-leaf (IPv6)

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 not-RPL-aware leaf, 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	IPv6 dst node
Inserted headers	RPI	--	--
Removed headers	--	--	--
Re-added headers	--	--	--
Modified headers	--	RPI	--
Untouched headers	--	--	RPI (Ignored)

Table 3: Storing: Summary of the use of headers from root to not-RPL-aware-leaf

[6.1.4.](#) SM: Example of Flow from not-RPL-aware-leaf to root

In this case the flow comprises:

not-RPL-aware-leaf (IPv6) --> 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 header, 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. In the figure, IP6-IP6 refers to IPv6-in-IPv6.

Header	IPv6 src node	6LR_1	6LR_i	6LBR dst
Inserted headers	--	IP6-IP6(RPI)	IP6-IP6(RPI)[1]	--
Removed headers	--	--	--	IP6-IP6(RPI)[1][2]
Re-added	--	--	IP6-IP6(RPI)[1]	--

headers				
Modified headers	--	--	IP6-IP6(RPI) [2]	--
Untouched headers	--	--	--	--

Figure 8: Storing mode: Summary of the use of headers from not-RPL-aware-leaf to root. [[1] Case where the IPv6-in-IPv6 header is addressed to the next hop (Node B). [2] Case where the IPv6-in-IPv6 header is addressed to the root (Node A).

## 6.2. Storing Mode: Interaction between Leaf and Internet.

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

RPL-aware-leaf to Internet

Internet to RPL-aware-leaf

not-RPL-aware-leaf to Internet

Internet to not-RPL-aware-leaf

### 6.2.1. SM: Example of Flow from RPL-aware-leaf 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:

RPL-aware-leaf (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-node type (e.g. 6LR)

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

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

Table 4: Storing: Summary of the use of headers from RPL-aware-leaf to Internet

### 6.2.2. SM: Example of Flow from Internet to RPL-aware-leaf

In this case the flow comprises:

Internet --> root (6LBR) --> 6LR\_i --> RPL-aware-leaf (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 header is added in a outer IPv6-in-IPv6 header (with the IPv6-in-IPv6 destination address set to the 6LR) and sent to 6LR, which modifies the rank in the RPI. When the packet arrives at 6LN the RPI header is removed and the packet processed.

The Figure 9 shows the table that summarizes what headers are needed for this use case. In the figure, IP6-IP6 refers to IPv6-in-IPv6.

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

Figure 9: Storing mode: Summary of the use of headers from Internet to RPL-aware-leaf.

### 6.2.3. SM: Example of Flow from not-RPL-aware-leaf to Internet

In this case the flow comprises:

not-RPL-aware-leaf (IPv6) --> 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 header 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.

The Figure 10 shows the table that summarizes what headers are needed for this use case. In the figure, IP6-IP6 refers to IPv6-in-IPv6.

Internet-Draft

RPL-data-plane

May 2019

Header	IPv6 src node	6LR_1	6LR_i [i=2,...,n]	6LBR	Internet dst
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: Storing mode: Summary of the use of headers from not-RPL-aware-leaf to Internet. [1] Case when packet is addressed to the root. [2] Case when the packet is addressed hop-by-hop.

#### 6.2.4. SM: Example of Flow from Internet to not-RPL-aware-leaf.

In this case the flow comprises:

Internet --> root (6LBR) --> 6LR\_i --> not-RPL-aware-leaf (IPv6)

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 header within an IPv6-in-IPv6 header. The IPv6-in-IPv6 is addressed hop-by-hop.

The final node should be able to remove one or more IPv6-in-IPv6 headers which are all addressed to it. The final node does not process the RPI, the node ignores the RPI. Further details about this are mentioned in [[I-D.thubert-roll-unaware-leaves](#)], which specifies RPL routing for a 6LN acting as a plain host and not 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.

The Figure 11 shows the table that summarizes what headers are needed for this use case. In the figure, IP6-IP6 refers to IPv6-in-IPv6.

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

Figure 11: Storing mode: Summary of the use of headers from Internet to not-RPL-aware-leaf.

### 6.3. Storing Mode: Interaction between Leaf and Leaf

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

RPL-aware-leaf to RPL-aware-leaf

RPL-aware-leaf to not-RPL-aware-leaf

not-RPL-aware-leaf to RPL-aware-leaf

not-RPL-aware-leaf to not-RPL-aware-leaf

### 6.3.1. SM: Example of Flow from RPL-aware-leaf to RPL-aware-leaf

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:

6LN --> 6LR\_ia --> common parent (6LR\_x) --> 6LR\_id --> 6LN

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

6LR\_ia (Node D) is the intermediate router 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 6LN (Node F) to the common parent (6LR\_x).

6LR\_id (Node E) is the intermediate router from the common parent (6LR\_x) (Node B) to destination 6LN (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 6LN.

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 increasing to decreasing 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	6LN	6LR_ia	6LR_x (common parent)	6LR_id	6LN
	src				dst

Inserted headers	RPI	--	--	--	--
Removed headers	--	--	--	--	RPI
Re-added headers	--	--	--	--	--
Modified headers	--	RPI	RPI	RPI	--
Untouched headers	--	--	--	--	--

Table 5: Storing: Summary of the use of headers for RPL-aware-leaf to RPL-aware-leaf

### 6.3.2. SM: Example of Flow from RPL-aware-leaf to not-RPL-aware-leaf

In this case the flow comprises:

6LN --> 6LR\_ia --> common parent (6LR\_x) --> 6LR\_id --> not-RPL-aware 6LN (IPv6)

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

6LR\_ia is the intermediate router from source (6LN) 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 6LN to the common parent (6LR\_x).

6LR\_id (Node E) is the intermediate router from the common parent (6LR\_x) (Node B) to destination not-RPL-aware 6LN (IPv6) (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 6LN.

This situation is identical to the previous situation [Section 6.3.1](#)

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

Header	6LN src	6LR_ia	6LR_x(common parent)	6LR_id	IPv6 dst node
Inserted headers	RPI	--	--	--	--
Removed headers	--	--	--	--	--
Re-added headers	--	--	--	--	--
Modified headers	--	RPI	RPI	RPI	--
Untouched headers	--	--	--	--	RPI(Ignored)

Table 6: Storing: Summary of the use of headers for RPL-aware-leaf to not-RPL-aware-leaf

### 6.3.3. SM: Example of Flow from not-RPL-aware-leaf to RPL-aware-leaf

In this case the flow comprises:

not-RPL-aware 6LN (IPv6) --> 6LR\_ia --> common parent (6LR\_x) --> 6LR\_id --> 6LN

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

6LR\_ia (Node E) is the intermediate router from source (not-RPL-aware 6LN (IPv6)) (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 ges through from source to the common parent.

6LR\_id (Node D) is the intermediate router from the common parent (6LR\_x) (Node B) to destination 6LN (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 destination 6LN.

The 6LR\_ia ( $ia=1$ ) (Node E) receives the packet from the the IPv6 node

(Node G) and inserts and the RPI header encapsulated in IPv6-in-IPv6 header. The IPv6-in-IPv6 header is addressed to the destination 6LN (Node F).

The Figure 12 shows the table that summarizes what headers are needed for this use case. In the figure, IP6-IP6 refers to IPv6-in-IPv6.

Header	IPv6 src node	6LR_ia	Common Parent (6LRx)	6LR_id	6LN dst
Inserted headers	--	IP6-IP6(RPI)	--	--	--
Removed headers	--	--	--	--	IP6-IP6(RPI)
Re-added headers	--	--	--	--	--
Modified headers	--	--	IP6-IP6(RPI)	IP6-IP6(RPI)	--
Untouched headers	--	--	--	--	--

Figure 12: Storing mode: Summary of the use of headers from not-RPL-aware-leaf to RPL-aware-leaf.

#### [6.3.4.](#) SM: Example of Flow from not-RPL-aware-leaf to not-RPL-aware-leaf

In this case the flow comprises:

```
not-RPL-aware 6LN (IPv6 src)--> 6LR_1--> 6LR_ia --> 6LBR --> 6LR_id
--> not-RPL-aware 6LN (IPv6 dst)
```



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 not-RPL-aware 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 IPv6 src to the root.

6LR\_id (C) is the intermediate router from the root (Node A) to the destination 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 (IPv6 dst).

Note that this flow is identical to [Section 6.3.3](#), except that the RPI is ignored at the IPv6 dst node.

The 6LR\_1 (Node E) receives the packet from the the IPv6 node (Node G) and inserts the RPI header (RPI), encapsulated in an IPv6-in-IPv6 header. The IPv6-in-IPv6 header is addressed to the final destination (Node J).

The Figure 13 shows the table that summarizes what headers are needed for this use case. In the figure, IP6-IP6 refers to IPv6-in-IPv6.

Header	IPv6 src node	6LR_1	6LR_ia	6LR_m	IPv6 dst node
Inserted headers	--	IP6-IP6(RPI)	--	--	--
Removed headers	--	--	--	--	IP6-IP6(RPI) RPI Ignored
Re-added headers	--	--	--	--	--
Modified headers	--	--	IP6-IP6(RPI)	IP6-IP6(RPI)	--
Untouched headers	--	--	--	--	--

Figure 13: Storing mode: Summary of the use of headers from not-RPL-aware-leaf to not-RPL-aware-leaf

## 7. 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 not-RPL aware nodes. Only the 6LBR needs to act if compensation is necessary for not-RPL aware receivers.

The following 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 (indicated by "Raf"), 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 3). In the Figure the (1) indicates a 6tisch case [RFC8180], where the RPI header may still be needed for the instanceID to be available for priority/channel selection at each hop.

Internet-Draft

RPL-data-plane

May 2019

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

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

### 7.1. Non-Storing Mode: Interaction between Leaf and Root

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

RPL-aware-leaf to root

root to RPL-aware-leaf

not-RPL-aware-leaf to root

[7.1.1.](#) Non-SM: Example of Flow from RPL-aware-leaf to root

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

RPL-aware-leaf (6LN) --> 6LR\_i --> root(6LBR)

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

6LR\_i is the intermediate router from source to destination. In this case, "1 <= i <= n", n is the number of routers (6LR) that the packet goes through from source (6LN) 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	6LN src	6LR_i	6LBR dst
Inserted headers	RPI	--	--
Removed headers	--	--	RPI
Re-added headers	--	--	--
Modified headers	--	RPI	--
Untouched headers	--	--	--

Table 7: Non Storing: Summary of the use of headers from RPL-aware-leaf to root

[7.1.2.](#) Non-SM: Example of Flow from root to RPL-aware-leaf

In this case the flow comprises:

root (6LBR) --> 6LR\_i --> RPL-aware-leaf (6LN)

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

6LR\_i is the intermediate router 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 (6LN).

The 6LBR inserts an RH3, and a RPI header. 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	6LN dst
Inserted headers	RPI, RH3	--	--
Removed headers	--	--	RH3, RPI
Re-added headers	--	--	--
Modified headers	--	RPI, RH3	--
Untouched headers	--	--	--

Table 8: Non Storing: Summary of the use of headers from root to RPL-aware-leaf

### [7.1.3.](#) Non-SM: Example of Flow from root to not-RPL-aware-leaf

In this case the flow comprises:

root (6LBR) --> 6LR\_i --> not-RPL-aware-leaf (IPv6)

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

6LR\_i is the intermediate router 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 (IPv6).

In 6LBR the RH3 is added, it is modified at each intermediate 6LR (6LR<sub>1</sub> and so on) and it is fully consumed in the last 6LR (6LR<sub>n</sub>), 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 <sub>i</sub> i=(1,..,n-1)	6LR <sub>n</sub>	IPv6 dst node
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 Storing: Summary of the use of headers from root to not-RPL-aware-leaf

7.1.4. Non-SM: Example of Flow from not-RPL-aware-leaf to root

In this case the flow comprises:

not-RPL-aware-leaf (IPv6) --> 6LR\_1 --> 6LR\_i --> root (6LBR)

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

6LR\_i is the intermediate router from source to destination. In this case, " $1 < i \leq n$ ", n is the number of routers (6LR) that the packet goes through from source (IPv6) to destination (6LBR). For example, 6LR\_1 (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 entire packet is consumed by the root.

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

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

headers				
+-----+	+-----+	+-----+	+-----+	+-----+
Untouched	--	--	--	--
headers				
+-----+	+-----+	+-----+	+-----+	+-----+

Figure 16: Non Storing: Summary of the use of headers from not-RPL-aware-leaf to root

## 7.2. Non-Storing Mode: Interaction between Leaf and Internet

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

RPL-aware-leaf to Internet

Internet to RPL-aware-leaf

not-RPL-aware-leaf to Internet

Internet to not-RPL-aware-leaf

### 7.2.1. Non-SM: Example of Flow from RPL-aware-leaf to Internet

In this case the flow comprises:

RPL-aware-leaf (6LN) --> 6LR<sub>i</sub> --> root (6LBR) --> Internet

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

6LR<sub>i</sub> is the intermediate router 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 (6LN) to 6LBR.

This case is identical to storing-mode case.

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



Internet.

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

Header	6LN 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 Storing: Summary of the use of headers from RPL-aware-leaf to Internet

### 7.2.2. Non-SM: Example of Flow from Internet to RPL-aware-leaf

In this case the flow comprises:

Internet --> root (6LBR) --> 6LR\_i --> RPL-aware-leaf (6LN)

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

6LR\_i is the intermediate router 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(6LN).

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.

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

Header	Internet dst	6LBR	6LR <sub>i</sub>	6LN src
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 Storing: Summary of the use of headers from Internet to RPL-aware-leaf

### 7.2.3. Non-SM: Example of Flow from not-RPL-aware-leaf to Internet

In this case the flow comprises:

not-RPL-aware-leaf (IPv6) --> 6LR<sub>1</sub> --> 6LR<sub>i</sub> --> root (6LBR) --> Internet

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

6LR<sub>i</sub> is the intermediate router from source to destination. In this case, " $1 < i \leq n$ ",  $n$  is the number of routers (6LR) that the packet goes through from source(IPv6) to 6LBR. e.g 6LR<sub>1</sub> ( $i=1$ ).

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<sub>1</sub>) will add a RPI header 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 6.2.3](#)).

The Figure 17 shows the table that summarizes what headers are needed for this use case. In the figure, IP6-IP6 refers to IPv6-in-IPv6.

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

Figure 17: Non Storing: Summary of the use of headers from not-RPL-aware-leaf to Internet

#### 7.2.4. Non-SM: Example of Flow from Internet to not-RPL-aware-leaf

In this case the flow comprises:

Internet --> root (6LBR) --> 6LR\_i --> not-RPL-aware-leaf (IPv6)

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

6LR\_i is the intermediate router from source to destination. In this case, " $1 < i \leq n$ ", n is the number of routers (6LR) that the packet goes through from 6LBR to not-RPL-aware-leaf (IPv6).

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.

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

Internet-Draft

RPL-data-plane

May 2019

Header	Internet src	6LBR	6LR <sub>1</sub>	6LR <sub>i</sub> (i=2,...,n)	IPv6 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-Storing mode: Summary of the use of headers from Internet to not-RPL-aware-leaf [1] The last 6LR before the IPv6 node.

### 7.3. Non-Storing Mode: Interaction between Leafs

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

RPL-aware-leaf to RPL-aware-leaf

RPL-aware-leaf to not-RPL-aware-leaf

not-RPL-aware-leaf to RPL-aware-leaf

not-RPL-aware-leaf to not-RPL-aware-leaf

#### 7.3.1. Non-SM: Example of Flow from RPL-aware-leaf to RPL-aware-leaf

In this case the flow comprises:

6LN src --> 6LR\_ia --> root (6LBR) --> 6LR\_id --> 6LN 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 is the intermediate router 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 6LN to the root.

6LR\_id is the intermediate router 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 header to the original packet, and send the packet upwards.

The originating node must put the RPI 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 option.

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, as it was contained in an IPv6-in-IPv6 header addressed to it. Otherwise, there may be a RPI header buried inside the inner IP header, which should get ignored.

Networks that use the RPL P2P extension [[RFC6997](#)] are essentially non-storing DODAGs and fall into this scenario or scenario [Section 7.1.2](#), with the originating node acting as 6LBR.

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

Internet-Draft

RPL-data-plane

May 2019

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

Figure 19: Non Storing mode: Summary of the use of headers for RPL-aware-leaf to RPL-aware-leaf

### 7.3.2. Non-SM: Example of Flow from RPL-aware-leaf to not-RPL-aware-leaf

In this case the flow comprises:

6LN --> 6LR\_ia --> root (6LBR) --> 6LR\_id --> not-RPL-aware (IPv6)

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

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

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

As in the previous case, the 6LN will insert a RPI (RPI<sub>1</sub>) 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 6LR\_id.

The Figure 20 shows the table that summarizes what headers are needed for this use case. In the figure, IP6-IP6 refers to IPv6-in-IPv6.

Header	6LN src	6LR_ia	6LBR	6LR_id	6LR_m	IPv6 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 Storing: Summary of the use of headers from RPL-aware-leaf to not-RPL-aware-leaf.

### 7.3.3. Non-SM: Example of Flow from not-RPL-aware-leaf to RPL-aware-leaf

In this case the flow comprises:

not-RPL-aware 6LN (IPv6) --> 6LR\_1 --> 6LR\_ia --> root (6LBR) --> 6LR\_id --> 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 is the intermediate router from source to the root. In this case, "1 <= ia <= n", n is the number of intermediate routers (6LR)

6LR\_id is the intermediate router from the root to the destination. In this case, "1 <= id <= m", m is the number of the intermediate routers (6LR).

This scenario is mostly identical to the previous one. The RPI 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 (RPI\_2).

The Figure 21 shows the table that summarizes what headers are needed for this use case. In the figure, IP6-IP6 refers to IPv6-in-IPv6.

Header	IPv6 src	6LR_1	6LR_ia	6LBR	6LR_id	6LN 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 Storing: Summary of the use of headers from not-RPL-aware-leaf to RPL-aware-leaf.

#### 7.3.4. Non-SM: Example of Flow from not-RPL-aware-leaf to not-RPL-aware-leaf

In this case the flow comprises:

not-RPL-aware 6LN (IPv6 src) --> 6LR\_1 --> 6LR\_ia --> root (6LBR) --> 6LR\_id --> not-RPL-aware (IPv6 dst)

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

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

6LR\_id is the intermediate router 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. In the figure, IP6-IP6 refers to IPv6-in-IPv6.

Header	IPv6 src node	6LR_1	6LR_ia	6LBR	6LR_id	6LR_m	IPv6 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 22: Non Storing: Summary of the use of headers from not-RPL-aware-leaf to not-RPL-aware-leaf

## 8. Operational Considerations of supporting not-RPL-aware-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.

## [9.](#) Operational considerations of introducing 0x23

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

During bootstrapping the node gets the DIO with the information of RPL 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 RPL

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

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 it is triggered when the DIO is sent with the flag indicating the new RPI value. 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.

## 10. 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.

[RFCXXXX] represents this document.

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.

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

Robles, et al. Expires November 16, 2019 [Page 46]

---

Internet-Draft RPL-data-plane May 2019

Bit number	Description	Reference
3	RPI 0x23 enable	This document

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

## 11. 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 the target of the 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! 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 header, 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 header 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 header therefore will not be seen by the network. Upon reaching the destination node the RPI header has no further meaning and is just skipped; the presence of a second RPI header will have no meaning to

the end node as the packet has already been identified as being at its final destination.

The RH3 and RPI headers 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.

## [12.](#) Acknowledgments

This work is done thanks to the grant by the Stand.ICT project.

A special BIG thanks to C. M. Heard for the help with the [Section 3](#). 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, Matthias Kovatsch, Peter van der Stok, Xavier Vilajosana and Thomas Watteyne.

## [13.](#) References

### [13.1.](#) 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>>.

Robles, et al.

Expires November 16, 2019

[Page 50]

---

Internet-Draft

RPL-data-plane

May 2019

- [RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", [RFC 6040](#), DOI 10.17487/RFC6040, November 2010, <<https://www.rfc-editor.org/info/rfc6040>>.
- [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>>.
- [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>>.

## [13.2](#). 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-11](#) (work in progress), February 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-19](#) (work in progress), March 2019.

[I-D.ietf-anima-bootstrapping-keyinfra]

Pritikin, M., Richardson, M., Behringer, M., Bjarnason, S., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", [draft-ietf-anima-bootstrapping-keyinfra-20](#) (work in progress), May 2019.

[I-D.ietf-intarea-tunnels]

Touch, J. and M. Townsley, "IP Tunnels in the Internet Architecture", [draft-ietf-intarea-tunnels-09](#) (work in progress), July 2018.

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

Thubert, P., "Routing for RPL Leaves", [draft-thubert-roll-unaware-leaves-07](#) (work in progress), April 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>>.

Internet-Draft

RPL-data-plane

May 2019

- [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>>.
- [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>>.

Robles, et al.

Expires November 16, 2019

[Page 53]

---

Internet-Draft

RPL-data-plane

May 2019

#### Authors' Addresses

Maria Ines Robles  
Aalto University  
Innopoly  
Espoo 02150  
Finland

Email: [mariainesrobles@gmail.com](mailto:mariainesrobles@gmail.com)

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

Email: [mcr+ietf@sandelman.ca](mailto:mcr+ietf@sandelman.ca)

URI: <http://www.sandelman.ca/mcr/>

Pascal Thubert  
Cisco Systems, Inc  
Village d'Entreprises Green Side 400, Avenue de Roumanille  
Batiment T3, Biot - Sophia Antipolis 06410  
France

Email: [pthubert@cisco.com](mailto:pthubert@cisco.com)

