6lo
Internet-Draft
Updates: 4944 (if approved)
Intended status: Standards Track
Expires: January 3, 2016

P. Thubert, Ed.
Cisco
C. Bormann
Uni Bremen TZI
L. Toutain
IMT-TELECOM Bretagne
R. Cragie
ARM

July 02, 2015

# A Routing Header Dispatch for 6LoWPAN
draft-thubert-6lo-routing-dispatch-05

## Abstract

This specification provides a new 6LoWPAN dispatch type for use in Route-over and mixed Mesh-under and Route-over topologies, that reuses the encoding of the mesh type defined in RFC 4944 for pure Mesh-under topologies. This specification also defines a method to compress RPL Option (RFC6553) information and Routing Header type 3 (RFC6554), an efficient IP-in-IP technique and opens the way for further routing techniques. This extends 6LoWPAN Transmission of IPv6 Packets (RFC4944), and is applicable to new link-layer types where 6LoWPAN is being defined.

## Status of This Memo



## Copyright Notice

Table of Contents



# 1. Introduction

The design of Low Power and Lossy Networks (LLNs) is generally focused on saving energy, which is the most constrained resource of all. The other constraints, such as the memory capacity and the duty cycling of the LLN devices, derive from that primary concern. Energy is often available from primary batteries that are expected to last for years, or is scavenged from the environment in very limited quantities. Any protocol that is intended for use in LLNs must be designed with the primary concern of saving energy as a strict requirement.

Controlling the amount of data transmission is one possible venue to save energy.  In a number of LLN standards, the frame size is limited to much smaller values than the IPv6 maximum transmission unit (MTU) of 1280 bytes.  In particular, an LLN that relies on the classical Physical Layer (PHY) of IEEE 802.14.5 [IEEE802154] is limited to 127 bytes per frame.  The need to compress IPv6 packets over IEEE 802.14.5 led to the 6LoWPAN Header Compression [RFC6282] work (6LoWPAN-HC).

Innovative Route-over techniques have been and are still being developed for routing inside a LLN.  In a general fashion, such techniques require additional information in the packet to provide loop prevention and to indicate information such as flow identification, source routing information, etc.

For reasons such as security and the capability to send ICMP errors back to the source, an original packet must not be tampered with, and any information that must be inserted in or removed from an IPv6 packet must be placed in an extra IP-in-IP encapsulation.  This is the case when the additional routing information is inserted by a router on the path of a packet, for instance a mesh root, as opposed to the source node.  This is also the case when some routing information must be removed from a packet that will flow outside the LLN.

As an example, the Routing Protocol for Low Power and Lossy Networks [RFC6550] (RPL) is designed to optimize the routing operations in constrained LLNs.  As part of this optimization, RPL requires the addition of RPL Packet Information (RPI) in every packet, as defined in Section 11.2 of [RFC6550].

The RPL Option for Carrying RPL Information in Data-Plane Datagrams [RFC6553] specification indicates how the RPI can be placed in a RPL Option for use in an IPv6 Hop-by-Hop header.  This representation demands a total of 8 bytes when in most cases the actual RPI payload requires only 19 bits.  Since the Hop-by-Hop header must not flow outside of the RPL domain, it must be removed from packets that leave the domain, and be inserted in packets entering the domain.  In both cases, this operation implies an IP-in-IP encapsulation.

```
              ------+---------                            ^
                    |          Internet                   |
                    |                                     | Native IPv6
                 +-----+                                  |
                 |     | Border Router (RPL Root)    ^    |    ^
                 |     |                             |    |    |
                 +-----+                             |    |    | IPv6 in
                    |                                |    |    | IPv6
              o    o   o    o                        |    |    | + RPI
          o o   o  o   o  o  o o   o                 |    |    |  or RH3
         o  o o  o o    o   o   o  o  o              |    |    |
         o   o    o  o     o  o    o  o  o           |    |    |
        o  o   o  o   o         o   o o              v    v    v
        o          o             o     o
                          LLN
```

Figure 1: IP-in-IP Encapsulation within the LLN

Additionally, in the case of the Non-Storing Mode of Operation (MOP), RPL requires a Routing Header type 3 (RH3) as defined in the IPv6 Routing Header for Source Routes with RPL [RFC6554] specification, for all packets that are routed down a RPL graph.  With Non-Storing RPL, even if the source is a node in the same LLN, the packet must first reach up the graph to the root so that the root can insert the RH3 to go down the graph.  In any fashion, whether the packet was originated in a node in the LLN or outside the LLN, and regardless of whether the packet stays within the LLN or not, as long as the source of the packet is not the root itself, the source-routing operation also implies an IP-in-IP encapsulation at the root to insert the RH3.

6TiSCH [I-D.ietf-6tisch-architecture] specifies the operation of IPv6 over the TimeSlotted Channel Hopping [I-D.ietf-6tisch-tsch] (TSCH) mode of operation of IEEE 802.14.5.  The architecture requires the use of both RPL and the 6lo adaptation layer framework ([RFC4944], [RFC6282]) over IEEE 802.14.5.  Because it inherits the constraints on the frame size from the MAC layer, 6TiSCH cannot afford to spend 8 bytes per packet on the RPI.  Hence the requirement for a 6LoWPAN header compression of the RPI.

The type of information that needs to be present in a packet inside the LLN but not outside of the LLN varies with the routing operation, but there is overall a need for an extensible compression technique that would simplify the IP-in-IP encapsulation, when needed, and optimally compress existing routing artifacts found in LLNs.

This specification extends 6LoWPAN [RFC4944] and in particular reuses the Mesh Header formats that are defined for the Mesh-under use cases so as to carry routing information for Route-over use cases.  The specification includes the formats necessary for RPL and is extensible for additional formats.

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

The Terminology used in this document is consistent with and incorporates that described in ‘Terminology in Low power And Lossy Networks’ [RFC7102] and [RFC6550].

The terms Route-over and Mesh-under are defined in [RFC6775].

Other terms in use in LLNs are found in [RFC7228].

The term "byte" is used in its now customary sense as a synonym for "octet".

## 3. Updating RFC 4944

This draft proposes 3 ways to adapt 6LoWPAN while maintaining backward compatibility with IPv6 over IEEE 802.15.4 [RFC4944].

Option 1 considers that a network where this specification applies is physically separate from a network where the Mesh Header defined in [RFC4944] is used.  With that assumption, the Mesh Header dispatch space can be reused.  A variation is proposed whereby the NALP pattern 00xxxxxx is reused instead of the Mesh Header pattern.

Option 2 defines a new Separator Dispatch value that indicates that no Mesh Header is present in the remainder of the packet.  If the 10xxxxxx pattern is found in the packet after this new Separator Dispatch, then this specification applies.  It is suggested that the new Separator Dispatch would also enable to reuse patterns 00xxxxxx and 11xxxxxx in the future.

Option 3 uses values in pattern 11xxxxxx that are free to this date, avoiding patterns 11000xxx and 11100xxx that are used for the Fragmentation Header as defined in [RFC4944].

### 3.1. Reusing Mesh Header (or NALP) Dispatch Space

Section 5.1 of the IPv6 over IEEE 802.15.4 [RFC4944] specification defines various Dispatch Types and Headers, and in particular a Mesh Header that corresponds to a pattern 10xxxxxx and effectively consumes one third of the whole 6LoWPAN dispatch space for Mesh-under specific applications.

This specification reuses the Dispatch space for Route-over and mixed operations. This means that a device that use the Mesh Header as specified in [RFC4944] should not be placed in a same network as a device which operates per this update. This is generally not a problem since a network is classically either Mesh-under OR Route-over.

A new implementation of Mesh-under MAY support both types of encoding, and if so, it SHOULD provide a management toggle to enable either mode and it SHOULD use this specification as the default mode.

A dispatch space of equivalent size to the Mesh Header was reserved in [RFC4944] for external specifications Not A LowPan (NALP), hoping that such specification could coexist harmlessly on a same network as early 6LoWPAN.

It is unclear that this disposition was useful at some point and that NALP was effectively used in a network where 6LoWPAN is deployed. A variation of the suggestion above would be, to use pattern 10xxxxxx instead of pattern 10xxxxxx If deemed necessary, it would be possible to move NALP to some other (smaller) dispatch space.

### 3.2. Add A New Dispatch

The suggestion here is not to use the Escape Dispatch, which is not entirely defined at this point, but to block one other dispatch value (say 11111111) to indicate that from that point on, the parsing of the packet should use this specification if the pattern 10xxxxxx is found.

The expectation is that if there is a Mesh Header, it is placed early in the packet and from there this specification will apply to any other appearance of the 10xxxxxx pattern. On the other hand, if there is no mesh header, there is a need to indicate so with this new dispatch value, and then any appearance of the 10xxxxxx pattern will be parsed per this specification.

It must be noted that the NALP space is really reserved for the first dispatch in the 6LoWPAN packet. Once a packet is identified as a 6LoWPAN packet by a first dispatch, the NALP range could be used.

Finally, the specification indicates that Fragments Headers must always preceed Routing header.

As a result, the 11111111 pattern could be considered a delimiminator between a portion of the frame that is formatted per [RFC4944] on the left, and a portion from which the space for Mesh Header, Fragment Header and NALP can be reused, on the right. This specification would reuse the Mesh Header or the NALP as discuused above, so the text in this specification is not impacted.

### 3.3. Use Free Space the the FRAG range

With the third proposal, the 6LoRH uses free bit patterns that are defined in [RFC4944] in the 11 xxxxxx range, avoiding FRAG1 of 11 000xxx and FRAGN of 11 100xxx.

The third bit, which differentiates FRAG1 from FRAGN in their particular ranges, indicates Elective vs. Critical; the fourth bit is always set to ensure that the 6LoRH does not collision with FRAG1 or FRAGN. The net result is one bit less than in the other proposals for the encoding space in the 6LoRH, which means only 4 bits to encode the length in the Elective format, as discussed below, and only 4 bits TSE.

The resulting formats and consequences are detailed in the relevant sections.

## 4. Placement of 6LoRH

One or more 6LoRHs MAY be placed in a 6LoWPAN packet and MUST always be placed before the LOWPAN_IPHC [RFC6282]. A 6LoRH MUST always be placed after Fragmentation Header and Mesh Header [RFC6282].

## 5. General Format

The 6LoWPAN Routing Header (6LoRH) may contain source routing information such as a compressed form of RH3, or other sorts of routing information such as the RPL RPI, source and/or destination address, and is extensible for future uses, with the given example of BIER bitmap encoding in Section 10.

There are two forms for 6LoRH:

   Elective (6LoRHE)

   Critical (6LoRHC)

This specification proposes several alternatives for the 6LoRH encoding:

Reuse Mesh Header Space in route over mode

Same as above, signaled by an initial escape byte

a more complex encoding using other coding space that is still free in the 6lo adaptation layer framework

The layout of the Elective and Critical forms depends on the encoding of the 6LoRH itself.

With the Mesh Header reuse proposal, the 6LoRH reuses the bit patterns that are defined in [RFC4944] for the Mesh Header, specifically the Dispatch Value Bit Pattern of 10xxxxxx.

With the Escaped Mesh Header reuse, the 6LoRH also reuses the bit patterns that are defined in [RFC4944] for the Mesh Header, but an ESC dispatch, with a value of 11111111, must be placed before the first 6LoRH.  The ESC indicates that the parsing of the pattern of 10xxxxxx will now be performed following this specification.  There is no need to place an ESC before each 6LoRH, since the ESC influences the parsing of the rest of the packet.

## 5.1.  Elective Format

With the first and second proposals, the 6LoRHE uses the Dispatch Value Bit Pattern of 101xxxxx.
A 6LoRHE may be ignored and skipped in parsing.
If it is ignored, the 6LoRHE is forwarded with no change inside the LLN.

```
 0                   1
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-       ...        -+
|1|0|1| Length  |      Type     |                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-       ...        -+
                                 <--    Length    -->
```

Figure 2: Elective 6LoWPAN Routing Header

With the third proposal 6LoRHE uses the Dispatch Value Bit Pattern of 1111xxxx.

```
       0                   1
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-       ...        -+
      |1|1|1|1| Length|      Type     |                    |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-       ...        -+
                                       <--    Length    -->
```

Figure 3: Elective 6LoWPAN Routing Header

Length:
Length of the 6LoRHE expressed in bytes, excluding the first 2 bytes. This is done to enable a node to skip a 6LoRH that it does not support and/or cannot parse, for instance if the Type is not known.

Type:
Type of the 6LoRHE

### 5.2. Critical Format

With the first and second proposals, the 6LoRHC uses the Dispatch Value Bit Pattern of 100xxxxx.
A node which does not support the 6LoRHC Type MUST silently discard the packet (note that there is no provision for the exchange of error messages; such a situation should be avoided by judicious use of administrative control and/or capability indications).

```
     0                   1
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-              ...             -+
    |1|0|0|   TSE   |      Type     |                                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-              ...             -+
                                     <-- Length implied by Type/TSE -->
```

Figure 4: Critical 6LoWPAN Routing Header

With the third proposal 6LoRHE uses the Dispatch Value Bit Pattern of 1101xxxx.

```
     0                   1
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-              ...             -+
    |1|1|0|1|  TSE  |      Type     |                                 |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-              ...             -+
                                     <-- Length implied by Type/TSE -->
```

TSE:

Type Specific Extension. The meaning depends on the Type, which must be known in all of the nodes. The interpretation of the TSE depends on the Type field that follows. For instance, it may be used to transport control bits, the number of elements in an array, or the length of the remainder of the 6LoRHC expressed in a unit other than bytes.

Type:
Type of the 6LoRHC

## 6. The Routing Header type 3 (RH3) 6LoRH

The Routing Header type 3 (RH3) 6LoRH (RH3-6LoRH) is a Critical 6LoWPAN Routing Header that provides a compressed form for the RH3, as defined in [RFC6554] for use by RPL routers. Routers that need to forward a packet with a RH3-6LoRH are expected to be RPL routers and expected to support this specification. If a non-RPL router receives a packet with a RPI-6LoRH, this means that there was a routing error and the packet should be dropped so the Type cannot be ignored.

```
  0                   1
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+    -+-    -+ ... +-    -+
 |1|0|0|  Size   |6LoRH Type 0..4| Hop1 | Hop2 |     | HopN |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+    -+-    -+ ... +-    -+

       Size indicates the number of compressed addresses
```

Figure 5: The RH3-6LoRH

The values for the RH3-6LoRH Type are an enumeration, 0 to 4. The form of compression is indicated by the Type as follows:

| Type | Size Unit |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |

Figure 6: The RH3-6LoRH Types

In the case of a RH3-6LoRH, the TSE field is used as a Size, which encodes the number of hops minus 1; so a Size of 0 means one hop, and the maximum that can be encoded is 32 hops. (If more than 32 hops need to be expressed, a sequence of RH3-6LoRH can be employed.)

The next Hop is indicated in the first entry of the first RH3-6LoRH. Upon reception, the entry is checked whether it refers to the processing router itself. If it so, the entry is removed from the RH3-6LoRH and the Size is decremented. If the Size is now zero, the whole RH3-6LoRH is removed. If there is no more RH3-6LoRH, the processing node is the last router on the way, which may or may not be collocated with the final destination.

The last hop in the last RH3-6LoRH is the last router prior to the destination in the LLN. So even when there is a RH3-6LoRH in the frame, the address of the final destination is in the LoWPAN_IPHC [RFC6282].

If some bits of the first address in the RH3-6LoRH can be derived from the final destination is in the LoWPAN_IPHC, then that address may be compressed, otherwise is is expressed in full. Next addresses only need to express the delta from the previous address.

All addresses in a RH3-6LoRH are compressed in a same fashion, down to the same number of bytes per address. In order to get different forms of compression, multiple consecutive RH3-6LoRH must be used.

## 7. The RPL Packet Information 6LoRH

[RFC6550], Section 11.2, specifies the RPL Packet Information (RPI) as a set of fields that are to be added to the IP packets for the purpose of Instance Identification, as well as Loop Avoidance and Detection.

In particular, the SenderRank, which is the scalar metric computed by an specialized Objective Function such as [RFC6552], indicates the Rank of the sender and is modified at each hop. The SenderRank allows to validate that the packet progresses in the expected direction, either upwards or downwards, along the DODAG.

RPL defines the RPL Option for Carrying RPL Information in Data-Plane Datagrams [RFC6553] to transport the RPI, which is carried in an IPv6 Hop-by-Hop Options Header [RFC2460], typically consuming eight bytes per packet.

With [RFC6553], the RPL option is encoded as six Octets; it must be placed in a Hop-by-Hop header that consumes two additional octets for a total of eight. In order to limit its range to the inside the RPL domain, the Hop-by-Hop header must be added to (or removed from) packets that cross the border of the RPL domain.

The 8-bytes overhead is detrimental to the LLN operation, in particular with regards to bandwidth and battery constraints. These bytes may cause a containing frame to grow above maximum frame size, leading to Layer 2 or 6LoWPAN [RFC4944] fragmentation, which in turn cause even more energy spending and issues discussed in the LLN Fragment Forwarding and Recovery [I-D.thubert-6lo-forwarding-fragments].

An additional overhead comes from the need, in certain cases, to add an IP-in-IP encapsulation to carry the Hop-by-Hop header. This is needed when the router that inserts the Hop-by-Hop header is not the source of the packet, so that an error can be returned to the router. This is also the case when a packet originated by a RPL node must be stripped from the Hop-by-Hop header to be routed outside the RPL domain.

This specification defines an IPinIP-6LoRH in Section 8 for that purpose, but it must be noted that stripping a 6LoRH does not require a manipulation of the packet in the LOWPAN_IPHC, and thus, if the source address in the LOWPAN_IPHC is the node that inserted the IPinIP-6LoRH then this alone does not mandate an IPinIP-6LoRH.

As a result, a RPL packet may bear only a RPI-6LoRH and no IPinIP-6LoRH. In that case, the source and destination of the packet are located in the LOWPAN_IPHC.

As with [RFC6553], the fields in the RPI include an 'O', an 'R', and an 'F' bit, an 8-bit RPLInstanceID (with some internal structure), and a 16-bit SenderRank.

The remainder of this section defines the RPI-6LoRH, a Critical 6LoWPAN Routing Header that is designed to transport the RPI in 6LoWPAN LLNs.

## 7.1. Compressing the RPLInstanceID

RPL Instances are discussed in [RFC6550], Section 5. A number of simple use cases will not require more than one instance, and in such a case, the instance is expected to be the global Instance 0. A global RPLInstanceID is encoded in a RPLInstanceID field as follows:

```
 0 1 2 3 4 5 6 7
+-+-+-+-+-+-+-+-+
|0|     ID      |  Global RPLInstanceID in 0..127
+-+-+-+-+-+-+-+-+
```

Figure 7: RPLInstanceID Field Format for Global Instances

For the particular case of the global Instance 0, the RPLInstanceID field is all zeros. This specification allows to elide a RPLInstanceID field that is all zeros, and defines a I flag that, when set, signals that the field is elided.

## 7.2. Compressing the SenderRank

The SenderRank is the result of the DAGRank operation on the rank of the sender; here the DAGRank operation is defined in [RFC6550], Section 3.5.1, as:

```
DAGRank(rank) = floor(rank/MinHopRankIncrease)
```

If MinHopRankIncrease is set to a multiple of 256, the least significant 8 bits of the SenderRank will be all zeroes; by eliding those, the SenderRank can be compressed into a single byte. This idea is used in [RFC6550] by defining DEFAULT_MIN_HOP_RANK_INCREASE as 256 and in [RFC6552] that defaults MinHopRankIncrease to DEFAULT_MIN_HOP_RANK_INCREASE.

This specification allows to encode the SenderRank as either one or two bytes, and defines a K flag that, when set, signals that a single byte is used.

## 7.3. The Overall RPI-6LoRH encoding

The RPI-6LoRH provides a compressed form for the RPL RPI. Routers that need to forward a packet with a RPI-6LoRH are expected to be RPL routers and expected to support this specification. If a non-RPL router receives a packet with a RPI-6LoRH, this means that there was a routing error and the packet should be dropped so the Type cannot be ignored.

Since the I flag is not set, the TSE field does not need to be a length expressed in bytes. The field is fully reused for control bits so as to encode the O, R and F flags from the RPI, and the I and K flags that indicate the compression that is taking place.

The Type for the RPI-6LoRH is 5 with the first proposal and in the range 5-8 with the third proposal.

The RPI-6LoRH is immediately followed by the RPLInstanceID field, unless that field is fully elided, and then the SenderRank, which is either compressed into one byte or fully in-lined as the whole 2 bytes. The I and K flags in the RPI-6LoRH indicate whether the RPLInstanceID is elided and/or the SenderRank is compressed and depending on these bits, the Length of the RPI-6LoRH may vary as described hereafter.

```
     0                   1                   2
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  ...  -+-+-+
    |1|0|0|O|R|F|I|K| 6LoRH Type=5  |   Compressed fields  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+  ...  -+-+-+
```

Figure 8: The Generic RPI-6LoRH Format

O, R, and F bits:
The O, R, and F bits as defined in [RFC6550], Section 11.2.

I bit:
If it is set, the Instance ID is elided and the RPLInstanceID is the Global RPLInstanceID 0. If it is not set, the octet immediately following the type field contains the RPLInstanceID as specified in [RFC6550] section 5.1.

K bit:
If it is set, the SenderRank is be compressed into one octet, and the lowest significant octet is elided. If it is not set, the SenderRank, is fully inlined as 2 octets.

In Figure 9, the RPLInstanceID is the Global RPLInstanceID 0, and the MinHopRankIncrease is a multiple of 256 so the least significant byte is all zeros and can be elided:

```
     0                   1                   2
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |1|0|0|O|R|F|1|1| 6LoRH Type=5  | SenderRank    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
              I=1, K=1
```

Figure 9: The most compressed RPI-6LoRH

In Figure 10, the RPLInstanceID is the Global RPLInstanceID 0, but both bytes of the SenderRank are significant so it can not be compressed:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |1|0|0|O|R|F|1|0| 6LoRH Type=5  |        SenderRank             |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
              I=1, K=0
```

Figure 10: Eliding the RPLInstanceID

In Figure 11, the RPLInstanceID is not the Global RPLInstanceID 0, and the MinHopRankIncrease is a multiple of 256:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |1|0|0|O|R|F|0|1| 6LoRH Type=5  | RPLInstanceID |  SenderRank   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
             I=0, K=1
```

Figure 11: Compressing SenderRank

In Figure 12, the RPLInstanceID is not the Global RPLInstanceID 0, and both bytes of the SenderRank are significant:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |1|0|0|O|R|F|0|0| 6LoRH Type=5  | RPLInstanceID |    Sender-...
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     ...-Rank      |
   +-+-+-+-+-+-+-+-+
             I=0, K=0
```

Figure 12: Least compressed form of RPI-6LoRH

A typical packet in RPL non-storing mode going down the RPL graph requires an IPinIP encapsulating the RH3, whereas the RPI is usually omitted, unless it is important to indicate the RPLInstanceID. To match this structure, an optimized IPinIP 6LoRH is defined in Section 8.

With the third approach, the format becomes:

```
    0                   1
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
   |1|1|0|1|O|R|F|r|6LoRH Type 5-8 | ...
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
              r = reserved
```

Figure 13: Third encoding

And the types include the setting of I and K as follows:

| Type | I | K |
|---|---|---|
| 5 | 0 | 0 |
| 6 | 0 | 1 |
| 7 | 1 | 0 |
| 8 | 1 | 1 |

Figure 14: The RPI-6LoRH Types

## 8. The IP-in-IP 6LoRH

The IP-in-IP 6LoRH (IPinIP-6LoRH) is an Elective 6LoWPAN Routing Header that provides a compressed form for the encapsulating IPv6 Header in the case of an IP-in-IP encapsulation.

An IPinIP encapsulation is used to insert a field such as a Routing Header or an RPI at a router that is not the source of the packet. In order to send an error back regarding the inserted field, the address of the router that performs the insertion must be provided.

The encapsulation can also enable a router down the path removing a field such as the RPI, but this can be done in the compressed form by removing the RPI-6LoRH, so an IPinIP-6LoRH encapsulation is not required for that sole purpose.

This field is not critical for routing so the Type can be ignored, and the TSE field contains the Length in bytes.

```
  0                   1                   2
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-       ...      -+
 |1|0|1| Length  | 6LoRH Type 9  |  Hop Limit    | Encaps. Address  |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-       ...      -+
```

Figure 15: The IPinIP-6LoRH

The Length of an IPinIP-6LoRH is expressed in bytes and MUST be at least 1, to indicate a Hop Limit (HL), that is decremented at each hop. When the HL reaches 0, the packet is dropped per [RFC2460]

If the Length of an IPinIP-6LoRH is exactly 1, then the Encapsulator Address is elided, which means that the Encapsulator is a well-known router, for instance the root in a RPL graph.

If the Length of an IPinIP-6LoRH is strictly more than 1, then an Encapsulator Address is placed in a compressed form after the Hop Limit field. The value of the Length indicates which compression is performed on the Encapsulator Address. For instance, a Size of 3 indicates that the Encapsulator Address is compressed to 2 bytes.

When it cannot be elided, the destination IP address of the IP-in-IP header is transported in a RH3-6LoRH as the first address of the list.

With RPL, the destination address in the IP-in-IP header is implicitly the root in the RPL graph for packets going upwards, and the destination address in the IPHC for packets going downwards. If the implicit value is correct, the destination IP address of the IP-in-IP encapsulation can be elided.

If the final destination of the packet is a leaf that does not support this specification, then the chain of 6LoRH must be stripped by the RPL/6LR router to which the leaf is attached. In that example, the destination IP address of the IP-in-IP header cannot be elided.

In the special case where the 6LoRH is used to route 6LoWPAN fragments, the destination address is not accessible in the IPHC on all fragments and can be elided only for the first fragment and for packets going upwards.

## 9. The Mesh Header 6LoRH

The Mesh Header 6LoRH (MH-6LoRH) is an Elective 6LoWPAN Routing Header that provides an alternate form for the Mesh Addressing Type and Header defined in [RFC4944] with the same semantics.

The MH-6LoRH is introduced as replacement for use in potentially mixed Route_Over and Mesh-under environments. LLN nodes that need to forward a packet with a MH-6LoRH are expected to support this specification. If a router that supports only Route-over receives a packet with a MH-6LoRH, this means that there was a routing error and the packet should be dropped, so the Type cannot be ignored.

The HopsLft field defined in [RFC4944] is encoded in the TSE, so this specification doubles the potential number of hops vs. [RFC4944] in the first proposal and is conserved in the third proposal.

The HopsLft value of 0x1F is reserved and signifies an 8-bit Deep Hops Left field immediately following the Type, and allows a source node to specify a hop limit greater than 30 hops.

```
    0                   1
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-              ...            -+
   |1|0|0| HopsLft |6LoRHType 11-14| originator address, final address |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-              ...            -+
```

Figure 16: The MH-6LoRH

The V and F flags defined in [RFC4944] are encoded in the MH-6LoRH Type as follows:

| Type | V | F |
|---|---|---|
| 11 | 0 | 0 |
| 12 | 0 | 1 |
| 13 | 1 | 0 |
| 14 | 1 | 1 |

Figure 17: The MH-6LoRH Types

## 10. The BIER 6LoRH

(Note that the current contents of this section is a proof of concept only; the details for this encoding need to be developed in parallel with defining the semantics of a constrained version of BIER.)

The Bit Index Explicit Replication (BIER) 6LoRH (BIER-6LoRH) is an Elective 6LoWPAN Routing Header that provides a variable-size container for a BIER Bitmap. BIER can be used to route downwards a RPL graph towards one or more LLN node, as discussed in the BIER Architecture [I-D.wijnands-bier-architecture] specification. The capability to parse the BIER Bitmap is necessary to forward the packet so the Type cannot be ignored.

```
       0                   1
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ... -+-+-+-    ...      -+
      |1|0|0|  Size   |6LoRHType 15-19| Control Fields |    bitmap     |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ... -+-+-+-    ...      -+
```

Figure 18: The BIER-6LoRH

The Type for a BIER-6LoRH indicates the size of words used to build the bitmap and whether the bitmap is operated as an uncompressed bit-by-bit mapping, or as a Bloom filter.

In the bit-by-bit case, each bit is mapped in an unequivocal fashion with a single addressable resource in the network.  This may rapidly lead to large bitmaps, and BIER allows to divide a network into groups that partition the network so that a given bitmap is locally significant to one group only.  This specification allows to encode a 1-byte Group ID in the BIER-6LoRH Control Fields.

A Bloom Filter can be seen as a compression technique for the bitmap. A Bloom Filter may generate false positives, which, in the case of BIER, result in undue forwarding of a packet down a path where no listener exists.

As an example, the Constrained-Cast [I-D.bergmann-bier-ccast] specification employs Bloom Filters as a compact representation of a match or non-match for elements in a large set.

In the case of a Bloom Filter, a number of Hash functions must be run to obtain a multi-bit signature of an encoded element.  This specification allows to signal an Identifier of the Hash functions being used to generate a certain bitmap, so as to enable a migration scenario where Hash functions are renewed.  A Hash ID is signaled as a 1-byte value, and, depending on the Type, there may be up to 2 or up to 8 Hash IDs passed in the BIER-6LoRH Control Fields associated with a Bloom Filter bitmap, as follows:

```
+-----------+-------------+-----------------+-----------+
|   Type    |   encoding  |  Control Fields | Word Size |
+-----------+-------------+-----------------+-----------+
|    15     | bit-by-bit  |      none       |  32 bits  |
|    16     | Bloom filter| 2* 1-byte HashID|  32 bits  |
|    17     | bit-by-bit  |      none       |  128 bits |
|    18     | Bloom filter| 8* 1-byte HashID|  128 bits |
|    19     | bit-by-bit  |  1-byte GroupID |  128 bits |
+-----------+-------------+-----------------+-----------+
```

Figure 19: The BIER-6LoRH Types

In order to address a potentially large number of devices, the bitmap may grow very large.  Yet, the maximum frame size for a given MAC layer may limit the number of bits that can be dedicated to routing. The Size indicates the number of words in the bitmap minus one, so a size of 0 means one word, a Size of 1 means 64 2 words, up to a size of 31 which means 32 words.

## 11. Security Considerations

The security considerations of [RFC4944], [RFC6282], and [RFC6553] apply.

Using a compressed format as opposed to the full in-line format is logically equivalent and does not create an opening for a new threat when compared to [RFC6550], [RFC6553] and [RFC6554].

## 12. IANA Considerations

This document creates a IANA registry for the 6LoWPAN Routing Header Type, and assigns the following values:

0..4 : RH3-6LoRH [RFCthis]

5 : RPI-6LoRH [RFCthis]

9 : IPinIP-6LoRH [RFCthis]

11..14 : MH-6LoRH [RFCthis]

15..19 : BIER-6LoRH [RFCthis]

## 13. Acknowledgments

The authors wish to thank Martin Turon, James Woodyatt and Ralph Droms for constructive reviews to the design in the 6lo Working Group. The overall discussion involved participants to the 6MAN, 6TiSCH and ROLL WGs, thank you all. Special thanks to the chairs of the ROLL WG, Michael Richardson and Ines Robles, and Brian Haberman, Internet Area A-D, and Adrian Farrel, Routing Area A-D, for driving this complex effort across Working Groups and Areas.

## 14. References

### 14.1. Normative References

[IEEE802154]
IEEE standard for Information Technology, "IEEE std. 802.15.4, Part. 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks", 2015.

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

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, September 2007.

[RFC6282] Hui, J. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, September 2011.

[RFC6550] Winter, T., Thubert, P., 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, March 2012.

[RFC6552] Thubert, P., "Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6552, March 2012.

[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, March 2012.

[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, March 2012.

[RFC7102] Vasseur, JP., "Terms Used in Routing for Low-Power and Lossy Networks", RFC 7102, January 2014.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, May 2014.

## 14.2. Informative References

[I-D.bergmann-bier-ccast]
Bergmann, O., Bormann, C., and S. Gerdes, "Constrained-Cast: Source-Routed Multicast for RPL", draft-bergmann-bier-ccast-00 (work in progress), November 2014.

[I-D.ietf-6tisch-architecture]
Thubert, P., "An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4", draft-ietf-6tisch-architecture-08 (work in progress), May 2015.

[I-D.ietf-6tisch-tsch]
Watteyne, T., Palattella, M., and L. Grieco, "Using IEEE802.15.4e TSCH in an IoT context: Overview, Problem Statement and Goals", draft-ietf-6tisch-tsch-06 (work in progress), March 2015.

[I-D.thubert-6lo-forwarding-fragments]
Thubert, P. and J. Hui, "LLN Fragment Forwarding and Recovery", draft-thubert-6lo-forwarding-fragments-02 (work in progress), November 2014.

[I-D.wijnands-bier-architecture]
Wijnands, I., Rosen, E., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast using Bit Index Explicit Replication", draft-wijnands-bier-architecture-05 (work in progress), March 2015.

[RFC6775] Shelby, Z., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, November 2012.

Authors' Addresses

Pascal Thubert (editor)
Cisco Systems
Village d'Entreprises Green Side
400, Avenue de Roumanille
Batiment T3
Biot - Sophia Antipolis  06410
FRANCE

Phone: +33 4 97 23 26 34
Email: pthubert@cisco.com

Carsten Bormann
Universitaet Bremen TZI
Postfach 330440
Bremen  D-28359
Germany

Phone: +49-421-218-63921
Email: cabo@tzi.org

Laurent Toutain
Institut MINES TELECOM; TELECOM Bretagne
2 rue de la Chataigneraie
CS 17607
Cesson-Sevigne Cedex  35576
France

Email: Laurent.Toutain@telecom-bretagne.eu

Robert Cragie
ARM Ltd.
110 Fulbourn Road
Cambridge  CB1 9NJ
UK

Email: robert.cragie@gridmerge.com