Concerns

1. Inefficient parsing
   • Orthogonal concepts unnecessarily intertwined
   • Byte fields not byte aligned

2. Unnecessary bits in simplest form
   • 1 hop, no frag packets contain frag and mesh bits

3. Leaves little room for extensions
   – No provision for additional routing parameters (except seqno)
   – Dangerous when there is no consensus
   – How do we add support for new protocols in the future?
Past Extension Attempts

- Final address determines if seqno appears
  - Tricky to decode, especially with variable sized fields
- Add \( B \) bit to indicate broadcast/multicast
  - Required more bits in subsequent fragments
  - Broke byte-alignment of some fields
    - Added padding to restore alignment in subsequent headers
- What about prot_type field?
  - Does not exist in subsequent fragments
- Modifying the current adaptation header is difficult
  - Especially if it becomes standard and widely adopted

IPv6 Header Format

- Basic Header (addressing, hops left, etc.)
  - Hop-by-Hop Options
  - Routing
  - Fragment
  - Destination Options
- Each header contains the type of the following header

| Addressing | HBH Options | Routing | Fragment | Dest Options | Payload... |

- What can we learn from IPv6?
  - With header stacking:
    - Separate orthogonal concepts
    - Clean ordering of headers
    - Clean extensibility
Current Header Format

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0
LF proto_type datagram_size dgram_tag B src pad OHops Left F
Address Source

6lowpan IPv6

| L2 Addressing | Addressing |
| L2 Mesh Options | Hop-by-Hop Options |
| L2 Fragmentation | Fragmentation |
| Upper Layer Protocol | Upper Layer Protocol |

Why not use IPv6 to guide header format?

Goals

• Preserve current functionality
• Reduce complexity
• Better byte alignment
• Reduce header size
• Clean extensibility
Proposed Header Stack

- [Header Type] + [Header]
  - Dispatch Type  ??? Header…
  - Mesh Type  Mesh Header…
  - Frag Type  Frag Header…

- Typical Header Stacks (preserve IPv6 ordering)
  - Single Hop, No Frag  HC1 Dispatch HC1 Hdr…
  - Multi Hop, No Frag  Mesh Dispatch Mesh Hdr… HC1 Dispatch HC1 Hdr…
  - Single Hop, Frag  Frag Dispatch Frag Hdr… HC1 Dispatch HC1 Hdr…
  - Multi Hop, Frag  Mesh Dispatch Mesh Hdr… Frag Dispatch Frag Hdr… HC1 Dispatch HC1 Hdr…

Proposed Header Details

- Dispatch Header (extendable)
  - 0 Dispatch IPv6, LOWPAN_HC1, Source Route, ???

- Mesh Header
  - 1 0 OF  Hops Originator Addr, Final Addr
  - 1 0 OF  0xF  Hops Originator Addr, Final Addr (15-255 hops)

- Fragmentation Header
  - 1 1 0  dgram_tag  dgram_size
  - 1 1 1  dgram_tag  dgram_size  dgram_offset
Comparison: Compactness & Functionality

- **Single Hop, No Fragmentation**
  - 2 Bytes: \[ \text{LF prot_type MB rsv} \]
  - 1 Byte: \[ \text{Dispatch} \]

- **Multi Hop, No Fragmentation**
  - 3 Bytes: \[ \text{LF prot_type MB rsv OF Hops Left} \]
  - 2 Bytes: \[ \text{10 OF Hops 0 Dispatch} \]

- **Single Hop, Fragmentation**
  - 5 Bytes: \[ \text{LF prot_type MB rsv datagram_size dgram_tag pad} \]
  - 4 Bytes: \[ \text{110 dgram_tag dgram_size 0 Dispatch} \]

- **Multi Hop, Fragmentation**
  - 6 Bytes: \[ \text{LF prot_type MB rsv datagram_size dgram_tag pad OF Hops Left} \]
  - 5 Bytes: \[ \text{10 OF Hops 110 dgram_tag dgram_size 0 Dispatch} \]

- **Multi Hop > 14 hops**
  - 6 Bytes: \[ \text{LF prot_type MB rsv datagram_size dgram_tag pad OF Hops Left} \]
  - 6 Bytes: \[ \text{10 OF 0xF Hops 110 dgram_tag dgram_size 0 Dispatch} \]

\[ \text{Proposed format handles 255 hops vs 63} \]
Comparison: Compactness & Functionality

- Byte savings in most common cases
- Equal in large diameter network
  - Proposed format supports >63 hops
- Keeps every bit of functionality
  - Fragmentation fields unchanged
  - Mesh fields unchanged (except hops left)

Comparison: Extensibility

- Deep Networks
  - 1 0xF
    - Hops
    - Addr
    - HC1 Dispatch
    - HC1 Hdr...
  - 1 0xF 0xF
    - Hops
    - Addr
    - HC1 Dispatch
    - HC1 Hdr...

- Mesh Protocols (LOAD, AODV, DYMO, Source route…)
  - Mesh Dispatch
  - Mesh Hdr...
  - HC1 Dispatch
  - HC1 Hdr...

- Other Upper Layer Protocols
  - Mesh Dispatch
  - Mesh Hdr...
  - HCX Dispatch
  - HCX Hdr...

- Anything Else…
  - Mesh Dispatch
  - Mesh Hdr...
  - HC1 Dispatch
  - HC1 Hdr...
  - Mesh Dispatch
  - Mesh Hdr...
  - XXX Dispatch
  - XXX Hdr...
  - YYY Dispatch
  - YYY Hdr...
  - ZZZ Dispatch
  - ZZZ Hdr...
In a Nutshell

• Reformatted 6lowpan header format

• Preserved all current capabilities

• Strictly more compact
  – Especially in the most frequent cases

• Follows IPv6 header stacking methodology
  – Strictly more expressive and extensible
    • Can cleanly address current concerns expressed in WG (e.g., Mesh Delivery, B, …) and potential future concerns (e.g., diagnostics)
  – Easier to parse
    • Clean, orthogonal, byte aligned

• Questions / Comments?
Generalize Dispatch?

- Adds 1 byte to Mesh and Frag header each
- Other advantages:
  - Full 11-bits for frag offset
  - Full 6-bits for hops

```
+---------+------+
|         |      |
| 0 1 2 3 | 4 5 6 7 8 9 | 0 1 2 3 4 5 6 7 8 9 0 1 |
| IPv6 Dispatch | IPv6 Header |
| HC1 Dispatch | HC1 Header |
| Mesh Dispatch | Hops | Originator Addr, Final Addr |
| Frag Dispatch | dgram_tag | dgram_size | dgram_offset |
```

- 1-hop, no frag: -1 byte
- Mesh or Frag: Same
- Mesh and Frag: +1 byte

Proposed Header Details

```
+---------+------+
|         |      |
| 0 1 2 3 | 4 5 6 7 8 9 | 0 1 2 3 4 5 6 7 8 9 0 1 |
| Dispatch | IPv6, LOWPAN_HC1, Source Route, ??? |
| 0x7F | Dispatch | IPv6, LOWPAN_HC1, ??? |
| Mesh Header | Hops | Originator Addr, Final Addr |
| 0xF | Hops | Originator Addr, Final Addr |
| Fragmentation Header | dgram_tag | dgram_size | dgram_offset |
```

(15-255 hops)
### Comparison: Mesh Broadcast

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Multi Hop, No Fragmentation, Broadcast**

<table>
<thead>
<tr>
<th>3 Bytes</th>
<th>LF</th>
<th>prot_type</th>
<th>MB</th>
<th>rsv</th>
<th>OF</th>
<th>Hops Left</th>
<th>Seqno</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 Bytes</td>
<td>1 0 OF</td>
<td>Hops</td>
<td>0 1 B Dispatch</td>
<td>Seqno</td>
<td>0</td>
<td>Dispatch</td>
<td></td>
</tr>
</tbody>
</table>

- Is an 8-bit seqno field really correct?
  - “Winner takes all” when there is no consensus of how to do routing
- Dispatch type leaves flexibility

### Comparison: Compactness & Functionality

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Multi Hop > 14 hops (First Fragment)**

<table>
<thead>
<tr>
<th>6 Bytes</th>
<th>LF</th>
<th>prot_type</th>
<th>MB</th>
<th>rsv</th>
<th>datagram_size</th>
<th>dgram_tag</th>
<th>OF</th>
<th>Hops</th>
<th>dgram_size</th>
<th>Dispatch</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 Bytes</td>
<td>1 0 OF</td>
<td>0xF</td>
<td>Hops</td>
<td>1 1 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Multi Hop > 14 hops (Subsequent Fragments)**

<table>
<thead>
<tr>
<th>6 Bytes</th>
<th>LF</th>
<th>dgram_offset</th>
<th>MB</th>
<th>rsv</th>
<th>datagram_size</th>
<th>dgram_tag</th>
<th>OF</th>
<th>Hops</th>
<th>dgram_size</th>
<th>dgram_offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 Bytes</td>
<td>1 0 OF</td>
<td>0xF</td>
<td>Hops</td>
<td>1 1 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Type/Dispatch Allocation?

- **Proposal 1**
  - 0xxxxxxx: Dispatch
    - 00000000: Not a LoWPAN packet
    - 11111111: A full 8-bit dispatch follows
  - 10xxxxxx: Mesh
  - 11xxxxxx: Fragmentation

- **Proposal 2**
  - 00xxxxxx: Reserved (maybe not a LoWPAN)
    - 00000000: Not a LoWPAN packet
  - 01xxxxxx: Mesh
  - 10xxxxxx: Fragmentation
  - 11xxxxxx: Dispatch
    - 11111111: A full 8-bit dispatch follows