RFC4944

- Link-layer fragmentation only in route-over → reassembly at each hop
- Fragment header

- Reassembly timer:
  - Starts when node receives first fragment
  - Timeout value MUST be <60s
  - When times out, buffer cleared, packet dropped

Figure 4: First Fragment

Figure 5: Subsequent Fragments

Units of 8 octets

+1 on each new frag
No initial value specified
Section 5.3 of [RFC4944] also defines how to fragment compressed IPv6 datagrams that do not fit within a single link frame. Section 5.3 of [RFC4944] defines the fragment header’s datagram_size and datagram_offset values as the size and offset of the IPv6 datagram before compression. As a result, all fragment payload outside the first fragment must carry their respective portions of the IPv6 datagram before compression. **This document does not change that requirement.** When using the fragmentation mechanism described in Section 5.3 of [RFC4944], any header that cannot fit within the first fragment MUST NOT be compressed.
Problem statement

• Per-hop fragmentation and reassembly has 2 issues:
  • Latency:
    • Increases end-to-end latency as you need to wait for each fragment at each hop
  • Reliability:
    • Limited memory → limited number of buffers (1-2?) → packet dropped when new frag received and old not fully reassembled yet
    • No frag recovery: 1 frag loss == packet dropped

• Proposed solution:
  • Fragment forwarding:
    • Source fragments
    • Intermediate nodes relays
    • LBR reassembles

→ Creation in IETF101 of 6lo fragmentation DT
Drafts resulting from 6lo fragmentation DT

- **draft-ietf-6lo-minimal-fragment**
  - provides an overview of 6LoWPAN fragmentation
  - highlights limits of VRB

- **draft-ietf-lwig-6lowpan-virtual-reassembly**
  - details simple VRB implementation technique which results in fragment forwarding, but without fragment recovery

- **draft-ietf-6lo-fragment-recovery**
  - Defines a new protocol to do end-to-end ACK’ing and recover fragments
Status

• draft-ietf-6lo-minimal-fragment-01 posted 11 March 2019
• Informational

Table of Contents

1. Overview of 6LoWPAN Fragmentation .................................. 2
2. Limits of Per-Hop Fragmentation and Reassembly .................. 4
   2.1. Latency .................................................. 4
   2.2. Memory Management and Reliability ........................... 4
3. Virtual Reassembly Buffer (VRB) Implementation ................. 5
4. Security Considerations .............................................. 5
5. IANA Considerations .................................................. 6
6. Acknowledgments ..................................................... 6
7. Informative References ............................................... 6
Authors' Addresses ..................................................... 6
Change in -01

• Fixed typos
• Link to study:
  • Yasuyuki Tanaka, Pascale Minet, Thomas Watteyne, "6LoWPAN Fragment Forwarding", IEEE Communications Standards Magazine, 2019
Simulation Study
Yasuyuki Tanaka

- Using 6TiSCH simulator (https://bitbucket.org/6tisch/simulator/src)
- topology shown on the left
- RFC8180 with 101 slot slotframe, sufficient bandwidth, no 6P, no RPL
- all nodes generate data
  pkPeriod = U[54s,66s]
- One data point = 100 runs
- 95% confidence intervals

240 bytes of RAM

1,280 bytes of RAM

draft-ietf-6lo-minimal-fragment

RFC4944

End-to-End Reliability (%)

Number of fragments per packet

Per-hop reassembly, 1 reassembly buffer

Fragment Forwarding (8 VRB table Entries)

End-to-End Latency (s)

Number of fragments per packet

Per-hop reassembly, 1 reassembly buffer

Fragment Forwarding (12 VRB Table Entries)
Questions to the WG

• Ready for WGLC?
## Backup: simulation settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSCH slotframe length</td>
<td>101 slots</td>
</tr>
<tr>
<td>Slot duration</td>
<td>10 ms</td>
</tr>
<tr>
<td>Link reliability</td>
<td>100 %</td>
</tr>
<tr>
<td>Packet interval</td>
<td>Uniform in [54 s, 66 s]</td>
</tr>
<tr>
<td><strong>Cell Allocation</strong></td>
<td></td>
</tr>
<tr>
<td>Node A</td>
<td>1 TX cell</td>
</tr>
<tr>
<td>Node B</td>
<td>2 TX cells and 1 RX cell</td>
</tr>
<tr>
<td>Node C</td>
<td>3 TX cells and 2 RX cells</td>
</tr>
<tr>
<td>Node D</td>
<td>4 TX cells and 3 RX cells</td>
</tr>
<tr>
<td>Node E</td>
<td>1 TX cell</td>
</tr>
<tr>
<td>Node F</td>
<td>2 TX cells and 1 RX cell</td>
</tr>
<tr>
<td>Node G</td>
<td>3 TX cells and 2 RX cells</td>
</tr>
<tr>
<td>Node H</td>
<td>4 TX cells and 3 RX cells</td>
</tr>
<tr>
<td>Node I</td>
<td>9 TX cells and 8 RX cell</td>
</tr>
<tr>
<td>Node J</td>
<td>9 RX cells</td>
</tr>
<tr>
<td>(per.hop reas.) # reassembly buffers</td>
<td>1 (1280 B of memory)</td>
</tr>
<tr>
<td>(frag. forwd.) # VRBs</td>
<td>8 (160 B of memory)</td>
</tr>
<tr>
<td>Num. fragment per packet</td>
<td>between 1 and 10</td>
</tr>
<tr>
<td>Number of simulation runs</td>
<td>100</td>
</tr>
<tr>
<td>Duration of one simulation run</td>
<td>7000 s</td>
</tr>
</tbody>
</table>