ï»¿

        On Forwarding 6LoWPAN Fragments over a Multi-Hop IPv6 Network

Abstract

   This document provides generic rules to enable the forwarding of an
   IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) fragment
   over a route-over network. Forwarding fragments can improve both
   end-to-end latency and reliability as well as reduce the buffer
   requirements in intermediate nodes; it may be implemented using RFC
   4944 and Virtual Reassembly Buffers (VRBs).

Table of Contents

1. Introduction

    The original 6LoWPAN fragmentation is defined in [RFC4944] for use
    over a single Layer 3 hop, though multiple Layer 2 hops in a mesh-
    under network is also possible, and was not modified by the update in
    [RFC6282]. 6LoWPAN operations including fragmentation depend on a
    link-layer security that prevents any rogue access to the network.

    In a route-over 6LoWPAN network, an IP packet is expected to be
    reassembled at each intermediate hop, uncompressed, pushed to Layer 3
    to be routed, and then compressed and fragmented again. This
    document introduces an alternate approach called 6LoWPAN Fragment
    Forwarding (6LFF) whereby an intermediate node forwards a fragment
    (or the bulk thereof, MTU permitting) without reassembling if the
    next hop is a similar 6LoWPAN link. The routing decision is made on
    the first fragment of the datagram, which has the IPv6 routing
    information. The first fragment is forwarded immediately, and a
    state is stored to enable forwarding the next fragments along the
    same path.

    Done right, 6LoWPAN Fragment Forwarding techniques lead to more
    streamlined operations, less buffer bloat, and lower latency. But it
    may be wasteful when fragments are missing, leading to locked
    resources and low throughput, and it may be misused to the point that
    the end-to-end latency of one packet falls behind that of per-hop
    reassembly.

    This specification provides a generic overview of 6LFF, discusses
    advantages and caveats, and introduces a particular 6LFF technique
    called "Virtual Reassembly Buffer" (VRB) that can be used while
    retaining the message formats defined in [RFC4944]. Basic
    recommendations such as the insertion of an inter-frame gap between
    fragments are provided to avoid the most typical caveats.

2. Terminology

2.1. 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.

2.2. Background

    Past experience with fragmentation, e.g., as described in "IPv4
    Reassembly Errors at High Data Rates" [RFC4963] and references
    therein, has shown that misassociated or lost fragments can lead to
    poor network behavior and, occasionally, trouble at the application
    layer. That experience led to the definition of the "Path MTU
    Discovery for IP version 6" [RFC8201] protocol that limits
    fragmentation over the Internet.

    "IP Fragmentation Considered Fragile" [RFC8900] discusses security
    threats that are linked to using IP fragmentation. The 6LoWPAN
    fragmentation takes place underneath the IP Layer, but some issues
    described there may still apply to 6LoWPAN fragments (as discussed in
    further details in Section 7).

    Readers are expected to be familiar with all the terms and concepts
    that are discussed in "IPv6 over Low-Power Wireless Personal Area
    Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and
    Goals" [RFC4919] and "Transmission of IPv6 Packets over IEEE 802.15.4
    Networks" [RFC4944].

    "Multiprotocol Label Switching Architecture" [RFC3031] states that
    with MPLS,

packets are "labeled" before they are forwarded. At subsequent
hops, there is no further analysis of the packet's network layer
header. Rather, the label is used as an index into a table which
specifies the next hop, and a new label.

The MPLS technique is leveraged in the present specification to
forward fragments that actually do not have a network-layer header,
since the fragmentation occurs below IP.

2.3. New Terms

This specification uses the following terms:

6LoWPAN Fragment Forwarding Endpoints: The 6LFF endpoints are the
   first and last nodes in an unbroken string of 6LFF nodes. They
   are also the only points where the fragmentation and reassembly
   operations take place.

Compressed Form: This specification uses the generic term
   "compressed form" to refer to the format of a datagram after the
   action of [RFC6282] and possibly [RFC8138] for Routing Protocol
   for Low-Power and Lossy Network (RPL) [RFC6550] artifacts.

Datagram_Size: The size of the datagram in its compressed form
   before it is fragmented.

Datagram_Tag: An identifier of a datagram that is locally unique to
   the Layer 2 sender. Associated with the link-layer address of the
   sender, this becomes a globally unique identifier for the datagram
   within the duration of its transmission.

Fragment_Offset: The offset of a fragment of a datagram in its
   compressed form.

3. Overview of 6LoWPAN Fragmentation

Figure 1 illustrates 6LoWPAN fragmentation. We assume node A
forwards a packet to node B, possibly as part of a multi-hop route
between 6LoWPAN Fragment Forwarding endpoints, which may be neither A
nor B, though 6LoWPAN may compress the IP header better when they are
both the 6LFF and the 6LoWPAN compression endpoints.

```
               +---+                      +---+
  ... ---| A |-------------------->| B |--- ...
               +---+                      +---+
                        # (frag. 5)


        123456789                    123456789
        +---------+                  +---------+
        |   #  ###|                  |###  #   |
        +---------+                  +---------+
           outgoing                     incoming
         fragmentation                 reassembly
             buffer                      buffer
```
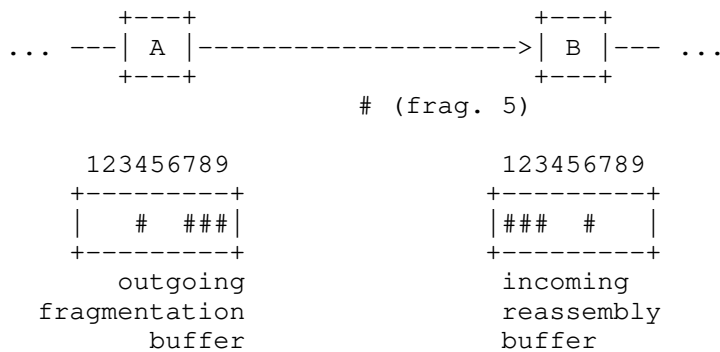
        Figure 1: Fragmentation at Node A, and Reassembly at Node B

Typically, node A starts with an uncompressed packet and compacts the
IPv6 packet using the header compression mechanism defined in
[RFC6282]. If the resulting 6LoWPAN packet does not fit into a
single link-layer frame, node A's 6LoWPAN sub-layer cuts it into
multiple 6LoWPAN fragments, which it transmits as separate link-layer
frames to node B. Node B's 6LoWPAN sub-layer reassembles these
fragments, inflates the compressed header fields back to the original
IPv6 header, and hands over the full IPv6 packet to its IPv6 layer.

In Figure 1, a packet forwarded by node A to node B is cut into nine
fragments, numbered 1 to 9 as follows:

* Each fragment is represented by the '#' symbol.

* Node A has sent fragments 1, 2, 3, 5, and 6 to node B.

* Node B has received fragments 1, 2, 3, and 6 from node A.

* Fragment 5 is still being transmitted at the link layer from node
   A to node B.

The reassembly buffer for 6LoWPAN is indexed in node B by:

* a unique identifier of node A (e.g., node A's link-layer address).

* the Datagram_Tag chosen by node A for this fragmented datagram.

Because it may be hard for node B to correlate all possible link-
layer addresses that node A may use (e.g., short versus long
addresses), node A must use the same link-layer address to send all
the fragments of the same datagram to node B.

Conceptually, the reassembly buffer in node B contains:

* a Datagram_Tag as received in the incoming fragments, associated
   with the interface and the link-layer address of node A for which
   the received Datagram_Tag is unique,

* the actual packet data from the fragments received so far, in a
   form that makes it possible to detect when the whole packet has
   been received and can be processed or forwarded,

* a state indicating the fragments already received,

* a Datagram_Size, and

* a timer that allows discarding a partially reassembled packet
   after some timeout.

A fragmentation header is added to each fragment; it indicates what
portion of the packet that fragment corresponds to. Section 5.3 of
[RFC4944] defines the format of the header for the first and
subsequent fragments. All fragments are tagged with a 16-bit
"Datagram_Tag", used to identify which packet each fragment belongs
to. Each datagram can be uniquely identified by the sender link-
layer addresses of the frame that carries it and the Datagram_Tag
that the sender allocated for this datagram. [RFC4944] also mandates
that the first fragment is sent first and with a particular format
that is different than that of the next fragments. Each fragment
except for the first one can be identified within its datagram by the
datagram-offset.

Node B's typical behavior, per [RFC4944], is as follows. Upon
receiving a fragment from node A with a Datagram_Tag previously
unseen from node A, node B allocates a buffer large enough to hold
the entire packet. The length of the packet is indicated in each
fragment (the Datagram_Size field), so node B can allocate the buffer
even if the fragment it receives first is not the first fragment. As
fragments come in, node B fills the buffer. When all fragments have
been received, node B inflates the compressed header fields into an
IPv6 header and hands the resulting IPv6 packet to the IPv6 layer,
which performs the route lookup. This behavior typically results in
per-hop fragmentation and reassembly. That is, the packet is fully
reassembled, then (re-)fragmented, at every hop.

4. Limitations of Per-Hop Fragmentation and Reassembly

There are at least two limitations to doing per-hop fragmentation and
reassembly. See [ARTICLE] for detailed simulation results on both
limitations.

4.1. Latency

When reassembling, a node needs to wait for all the fragments to be

received before being able to re-form the IPv6 packet and possibly
forwarding it to the next hop. This repeats at every hop.

This may result in increased end-to-end latency compared to a case
where each fragment is forwarded without per-hop reassembly.

4.2. Memory Management and Reliability

Constrained nodes have limited memory. Assuming a reassembly buffer
for a 6LoWPAN MTU of 1280 bytes as defined in Section 4 of [RFC4944],
typical nodes only have enough memory for 1-3 reassembly buffers.

To illustrate this, we use the topology from Figure 2, where nodes A,
B, C, and D all send packets through node E. We further assume that
node E's memory can only hold 3 reassembly buffers.

```
                +---+          +---+
      ... --->| A |------>| B |
                +---+          +---+\
                                     \
                          +---+      +---+
                          | E |--->| F |  ...
                          +---+      +---+
                         /
                        /
            +---+          +---+
      ... --->| C |------>| D |
                +---+          +---+
```
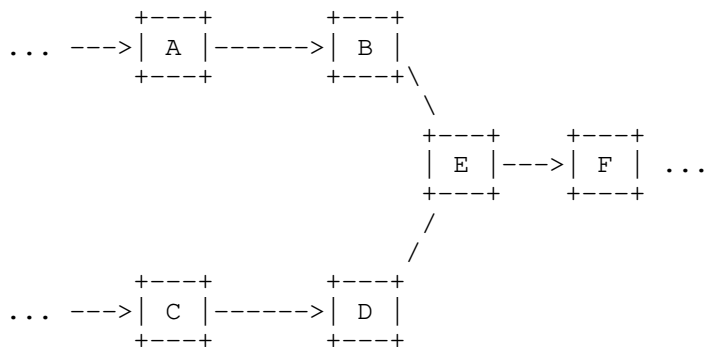
              Figure 2: Illustrating the Memory Management Issue

When nodes A, B, and C concurrently send fragmented packets, all
three reassembly buffers in node E are occupied. If, at that moment,
node D also sends a fragmented packet, node E has no option but to
drop one of the packets, lowering end-to-end reliability.

5. Forwarding Fragments

A 6LoWPAN Fragment Forwarding technique makes the routing decision on
the first fragment, which is always the one with the IPv6 address of
the destination. Upon receiving a first fragment, a forwarding node
(e.g., node B in an A->B->C sequence) that does fragment forwarding
MUST attempt to create a state and forward the fragment. This is an
atomic operation, and if the first fragment cannot be forwarded, then
the state MUST be removed.

Since the Datagram_Tag is uniquely associated with the source link-
layer address of the fragment, the forwarding node MUST assign a new
Datagram_Tag from its own namespace for the next hop and rewrite the
fragment header of each fragment with that Datagram_Tag.

When a forwarding node receives a fragment other than a first
fragment, it MUST look up state based on the source link-layer
address and the Datagram_Tag in the received fragment. If no such
state is found, the fragment MUST be dropped; otherwise, the fragment
MUST be forwarded using the information in the state found.

Compared to Section 3, the conceptual reassembly buffer in node B now
contains the following, assuming that node B is neither the source
nor the final destination:

*   a Datagram_Tag as received in the incoming fragments, associated
    with the interface and the link-layer address of node A for which
    the received Datagram_Tag is unique.

*   the link-layer address that node B uses as the source to forward
    the fragments.

*   the interface and the link-layer address of the next-hop C that is
    resolved on the first fragment.

* a Datagram_Tag that node B uniquely allocated for this datagram
  and that is used when forwarding the fragments of the datagram.

* a buffer for the remainder of a previous fragment left to be sent.

* a timer that allows discarding the stale 6LFF state after some
  timeout. The duration of the timer should be longer than that
  which covers the reassembly at the receiving endpoint.

A node that has not received the first fragment cannot forward the
next fragments. This means that if node B receives a fragment, node
A was in possession of the first fragment at some point. To keep the
operation simple and consistent with [RFC4944], the first fragment
MUST always be sent first. When that is done, if node B receives a
fragment that is not the first and for which it has no state, then
node B treats it as an error and refrains from creating a state or
attempting to forward. This also means that node A should perform
all its possible retries on the first fragment before it attempts to
send the next fragments, and that it should abort the datagram and
release its state if it fails to send the first fragment.

Fragment forwarding obviates some of the benefits of the 6LoWPAN
header compression [RFC6282] in intermediate hops. In return, the
memory used to store the packet is distributed along the path, which
limits the buffer-bloat effect. Multiple fragments may progress
simultaneously along the network as long as they do not interfere.
An associated caveat is that on a half-duplex radio, if node A sends
the next fragment at the same time as node B forwards the previous
fragment to node C down the path, then node B will miss it. If node
C forwards the previous fragment to node D at the same time and on
the same frequency as node A sends the next fragment to node B, this
may result in a hidden terminal problem. In that case, the
transmission from node C interferes at node B with that from node A,
unbeknownst to node A. Consecutive fragments of a same datagram MUST
be separated with an inter-frame gap that allows one fragment to
progress beyond the next hop and beyond the interference domain
before the next shows up. This can be achieved by interleaving
packets or fragments sent via different next-hop routers.

6. Virtual Reassembly Buffer (VRB) Implementation

   The VRB [LWIG-VRB] is a particular incarnation of a 6LFF that can be
   implemented without a change to [RFC4944].

   VRB overcomes the limitations listed in Section 4. Nodes do not wait
   for the last fragment before forwarding, reducing end-to-end latency.
   Similarly, the memory footprint of VRB is just the VRB table,
   reducing the packet drop probability significantly.

   However, there are other caveats:

   Non-zero Packet Drop Probability: The abstract data in a VRB table
      entry contains at a minimum the link-layer address of the
      predecessor and the successor, the Datagram_Tag used by the
      predecessor, and the local Datagram_Tag that this node will swap
      with it. The VRB may need to store a few octets from the last
      fragment that may not have fit within MTU and that will be
      prepended to the next fragment. This yields a small footprint
      that is 2 orders of magnitude smaller, compared to needing a
      1280-byte reassembly buffer for each packet. Yet, the size of the
      VRB table necessarily remains finite. In the extreme case where a
      node is required to concurrently forward more packets than it has
      entries in its VRB table, packets are dropped.

   No Fragment Recovery: There is no mechanism in VRB for the node that
      reassembles a packet to request a single missing fragment.
      Dropping a fragment requires the whole packet to be resent. This
      causes unnecessary traffic, as fragments are forwarded even when
      the destination node can never construct the original IPv6 packet.

   No Per-Fragment Routing: All subsequent fragments follow the same

sequence of hops from the source to the destination node as the
first fragment, because the IP header is required in order to
route the fragment and is only present in the first fragment. A
side effect is that the first fragment must always be forwarded
first.

The severity and occurrence of these caveats depend on the link layer
used. Whether they are acceptable depends entirely on the
requirements the application places on the network.

If the caveats are present and not acceptable for the application,
alternative specifications may define new protocols to overcome them.
One example is [RFC8931], which specifies a 6LFF technique that
allows the end-to-end fragment recovery between the 6LFF endpoints.

7. Security Considerations

An attacker can perform a Denial-of-Service (DoS) attack on a node
implementing VRB by generating a large number of bogus "fragment 1"
fragments without sending subsequent fragments. This causes the VRB
table to fill up. Note that the VRB does not need to remember the
full datagram as received so far but only possibly a few octets from
the last fragment that could not fit in it. It is expected that an
implementation protects itself to keep the number of VRBs within
capacity, and that old VRBs are protected by a timer of a reasonable
duration for the technology and destroyed upon timeout.

Secure joining and the link-layer security that it sets up protects
against those attacks from network outsiders.

"IP Fragmentation Considered Fragile" [RFC8900] discusses security
threats and other caveats that are linked to using IP fragmentation.
The 6LoWPAN fragmentation takes place underneath the IP Layer, but
some issues described there may still apply to 6LoWPAN fragments.

* Overlapping fragment attacks are possible with 6LoWPAN fragments,
   but there is no known firewall operation that would work on
   6LoWPAN fragments at the time of this writing, so the exposure is
   limited. An implementation of a firewall SHOULD NOT forward
   fragments but instead should recompose the IP packet, check it in
   the uncompressed form, and then forward it again as fragments if
   necessary. Overlapping fragments are acceptable as long as they
   contain the same payload. The firewall MUST drop the whole packet
   if overlapping fragments are encountered that result in different
   data at the same offset.

* Resource-exhaustion attacks are certainly possible and a sensitive
   issue in a constrained network. An attacker can perform a DoS
   attack on a node implementing VRB by generating a large number of
   bogus first fragments without sending subsequent fragments. This
   causes the VRB table to fill up. When hop-by-hop reassembly is
   used, the same attack can be more damaging if the node allocates a
   full Datagram_Size for each bogus first fragment. With the VRB,
   the attack can be performed remotely on all nodes along a path,
   but each node suffers a lesser hit. This is because the VRB does
   not need to remember the full datagram as received so far but only
   possibly a few octets from the last fragment that could not fit in
   it. An implementation MUST protect itself to keep the number of
   VRBs within capacity and to ensure that old VRBs are protected by
   a timer of a reasonable duration for the technology and destroyed
   upon timeout.

* Attacks based on predictable fragment identification values are
   also possible but can be avoided. The Datagram_Tag SHOULD be
   assigned pseudorandomly in order to reduce the risk of such
   attacks. A larger size of the Datagram_Tag makes the guessing
   more difficult and reduces the chances of an accidental reuse
   while the original packet is still in flight, at the expense of
   more space in each frame. Nonetheless, some level of risk remains
   because an attacker that is able to authenticate to and send
   traffic on the network can guess a valid Datagram_Tag value, since

there are only a limited number of possible values.

    * Evasion of Network Intrusion Detection Systems (NIDSs) leverages
       ambiguity in the reassembly of the fragment. This attack makes
       little sense in the context of this specification since the
       fragmentation happens within the Low-Power and Lossy Network
       (LLN), meaning that the intruder should already be inside to
       perform the attack. NIDS systems would probably not be installed
       within the LLN either but rather at a bottleneck at the exterior
       edge of the network.

8. IANA Considerations

    This document has no IANA actions.

9. References

9.1. Normative References

    [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>.

    [RFC4919] Kushalnagar, N., Montenegro, G., and C. Schumacher, "IPv6
               over Low-Power Wireless Personal Area Networks (6LoWPANs):
               Overview, Assumptions, Problem Statement, and Goals",
               RFC 4919, DOI 10.17487/RFC4919, August 2007,
               <https://www.rfc-editor.org/info/rfc4919>.

    [RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler,
               "Transmission of IPv6 Packets over IEEE 802.15.4
               Networks", RFC 4944, DOI 10.17487/RFC4944, September 2007,
               <https://www.rfc-editor.org/info/rfc4944>.

    [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>.

9.2. Informative References

    [ARTICLE] Tanaka, Y., Minet, P., and T. Watteyne, "6LoWPAN Fragment
               Forwarding", IEEE Communications Standards Magazine, Vol.
               3, Issue 1, pp. 35-39, DOI 10.1109/MCOMSTD.2019.1800029,
               March 2019,
               <https://ieeexplore.ieee.org/abstract/document/8771317>.

    [LWIG-VRB] Bormann, C. and T. Watteyne, "Virtual reassembly buffers
               in 6LoWPAN", Work in Progress, Internet-Draft, draft-ietf-
               lwig-6lowpan-virtual-reassembly-02, 9 March 2020,
               <https://tools.ietf.org/html/draft-ietf-lwig-6lowpan-
               virtual-reassembly-02>.

    [RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol
               Label Switching Architecture", RFC 3031,
               DOI 10.17487/RFC3031, January 2001,
               <https://www.rfc-editor.org/info/rfc3031>.

    [RFC4963] Heffner, J., Mathis, M., and B. Chandler, "IPv4 Reassembly
               Errors at High Data Rates", RFC 4963,
               DOI 10.17487/RFC4963, July 2007,
               <https://www.rfc-editor.org/info/rfc4963>.

    [RFC6282] Hui, J., Ed. and P. Thubert, "Compression Format for IPv6
               Datagrams over IEEE 802.15.4-Based Networks", RFC 6282,
               DOI 10.17487/RFC6282, September 2011,
               <https://www.rfc-editor.org/info/rfc6282>.

    [RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J.,
               Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur,
               JP., and R. Alexander, "RPL: IPv6 Routing Protocol for

                   Low-Power and Lossy Networks", RFC 6550,
                   DOI 10.17487/RFC6550, March 2012,
                   <https://www.rfc-editor.org/info/rfc6550>.

   [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>.

   [RFC8201]   McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed.,
               "Path MTU Discovery for IP version 6", STD 87, RFC 8201,
               DOI 10.17487/RFC8201, July 2017,
               <https://www.rfc-editor.org/info/rfc8201>.

   [RFC8900]   Bonica, R., Baker, F., Huston, G., Hinden, R., Troan, O.,
               and F. Gont, "IP Fragmentation Considered Fragile",
               BCP 230, RFC 8900, DOI 10.17487/RFC8900, September 2020,
               <https://www.rfc-editor.org/info/rfc8900>.

   [RFC8931]   Thubert, P., Ed., "IPv6 over Low-Power Wireless Personal
               Area Network (6LoWPAN) Selective Fragment Recovery",
               RFC 8931, DOI 10.17487/RFC8931, November 2020,
               <https://www.rfc-editor.org/info/rfc8931>.

Acknowledgments

Authors' Addresses

   Thomas Watteyne (editor)
   Analog Devices
   32990 Alvarado-Niles Road, Suite 910
   Union City, CA 94587
   United States of America

   Email: thomas.watteyne@analog.com

   Pascal Thubert (editor)
   Cisco Systems, Inc
   Building D
   45 Allee des Ormes - BP1200
   06254 Mougins - Sophia Antipolis
   France

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

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

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