# RFC 4944: Per-hop Fragmentation and Reassembly Issues

Georgios Z. Papadopoulos*, Pascal Thubert†, Sotirios Tsakalidis§ and Nicolas Montavont*

*IMT Atlantique, IRISA, UBL, France, {name.lastname}@imt-atlantique.fr
†Cisco Systems, France, {pthubert}@cisco.com
§Aristotle University of Thessaloniki, Greece, {stsakalid}@ece.auth.gr

***Abstract*—The first Internet Engineering Task Force (IETF) Working Group (WG) on Internet of Things (IoT), IPv6 over Low power WPAN (6LoWPAN), was established in 2005 to enable mechanisms to transmit Internet Protocol version 6 (IPv6) packets over IEEE 802.15.4. Since IPv6 supports packet sizes larger than the IEEE 802.15.4 maximum frame size, 6LoWPAN WG defined an adaptation layer by standardizing several documents such RFC 4944, RFC 6282 and RFC 6775. RFC 4944 describes the frame format for transmission of IPv6 packets and it defines mechanisms for header compression and fragmentation. RFC 6282 updates RFC 4944, however, it does not reconsider the 6LoWPAN fragment forwarding method. Indeed, considering that IEEE 802.15.4 comes with a Maximum Transmission Unit (MTU) size of 127 bytes, while a IPv6 datagrams with a 1280 byte MTU, an IPv6 packet could be fragmented into more than ten fragments at the 6LoWPAN adaptation layer. Then, using the 6LoWPAN route-over mesh network, the fragmented 6LoWPAN packet must be reassembled at every hop, which eventually causes latency, congestion, interference and packet losses. In this paper, we first provide a thorough background on RFC 4944. We then detail the potential fragment forwarding issues when employing the route over scheme. Finally, we present the ongoing efforts at the IETF to address the undesired issues.**

***Index Terms*—Internet of Things; RFC 4944; 6LoWPAN; Fragmentation; Route Over; Fragment Forwarding;**

## I. Introduction

The Internet Protocol (IP) can be considered as the "thin waist" of the Internet. Indeed, it is the key enabler of the Internet's explosive evolution and growth during the last 40 years. Thanks to widespread usage of IP version 6 (IPv6) [1], more and more constrained devices are getting connected to the Internet, which eventually converged into a new paradigm called the Internet of Things (IoT) [2]. There has been essential interest in designing and deploying IoT-based applications, with business sectors ranging from Smart Grid [3] to industrial IoT [4], where low cost and easily deployed IoT devices can provide significant benefits.

IoT, also referred to as Low-power and Lossy Network (LLN), is usually composed of hundreds of small, uniquely identifiable and limited in memory capacity objects. Typically, these devices are employed with low-power and lossy communication technologies.

Therefore, at the Internet Engineering Task Force (IETF), a number of Working Groups (WG) have been established to define a set of protocols for various layers of the LLN protocol stack to mitigate the potential issues. The CoRE WG defined the web transfer protocol, CoAP [5], the ROLL WG specified the routing protocol, RPL [6], while the 6TiSCH WG [7] focuses on enabling IPv6 over the IEEE 802.15.4-TSCH standard [8]. In Fig. 1, an LLN-based stack is depicted.

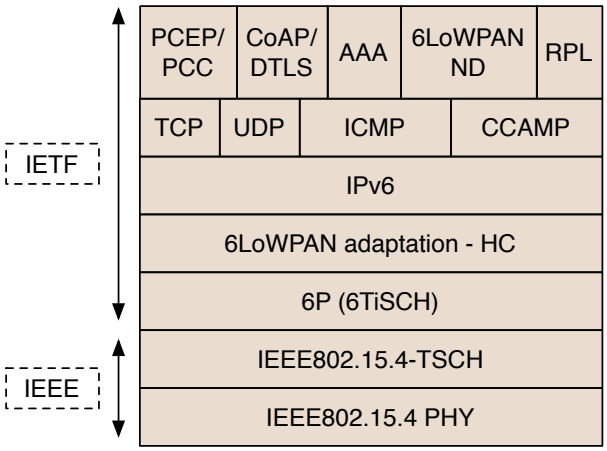



Fig. 1. LLNs protocol stack.

The IEEE 802.15.4 technology comes with a number of limitations, among which are limited communication range and a Maximum Transmission Unit (MTU) of 127 $bytes$. Considering that IPv6 comes with a MTU of 1280 $bytes$ [1], it would be impossible to transmit an IPv6 datagram over IEEE 802.15.4. Therefore, the IETF 6LoWPAN WG was chartered to fulfill the IPv6 requirements and, thus, enable IPv6 packet transmissions over LLNs [9]. The 6LoWPAN WG specified an intermediate layer between layers two and three, called the IPv6 adaptation layer [10]. The 6LoWPAN adaptation layer defines compression, fragmentation, reassembling and forwarding mechanisms for IPv6 datagrams that do not fit in the MTU of 127 $bytes$. The 6LoWPAN WG was replaced by IPv6 over Networks of Resource-constrained Nodes (6lo) WG which focuses on facilitating IPv6 connectivity over wider range of radio technologies such as Bluetooth Low Energy (BLE) RFC 7668 [11] and Z-Wave RFC 7428 [12].

The compression algorithm is necessary to reduce the 40 $byte$ IPv6 header. Additionally, fragmentation and reassembly are required to split the large IPv6 datagrams into multiple short fragments. Finally, RFC 4944 comes with hop-by-hop forwarding solutions, where at each hop a node reassembles and fragments again the entire datagram before transmitting to the next hop along the path, which is not ideal for a forwarder.

RFC 6282 [13] updates RFC 4944, however, it does not fully replace RFC 4944 since it does not redefine the 6LoWPAN fragment forwarding method. RFC 6282 only replaces the compression piece while the fragments and the mesh header are not updated (yet). Thus, RFC 4944 is still the current reference for fragments.

In this paper, we focus on the fragment forwarding technique of 6LoWPAN, when considering IPv6 fragmented datagrams in multi-hop networks. After providing an overview of LLN Fragment Forwarding, we thoroughly present all the issues that the fragment forwarding scheme of RFC 4944 introduces in mesh networks. Our study shows that there are essential problems in terms of network reliability, end-to-end delay, resource usage and implementation. This work is positioned as a problem statement article.

The paper is organized as follows. In Section II, we provide a thorough background on the 6LoWPAN mechanism. Then, in Section III, we list the potential issues that the *route over* forwarding scheme of 6LoWPAN may cause. In Section IV, the proposed solutions from the standardization community are presented. Finally, Section V concludes our paper.

## II. Background: RFC 4944

In this section, the 6LoWPAN fragmentation and reassembly mechanisms as well as the fragments forwarding scheme are presented in detail.

### A. Fragmentation

As specified in [10], the fragmentation procedure takes place only when an IPv6 datagram does not fit within a single IEEE 802.15.4 frame. Indeed, if the IPv6 data packet does fit, then it will be transmitted unfragmented and, thus, the LoWPAN encapsulation will not contain the fragmentation header. On the opposite hand, the transmitter node splits the datagram into multiple link fragments when it does not fit in 127 $bytes$, the maximum physical layer frame size. The length of each link fragment is defined in multiples of eight bytes. To enable the fragmentation and reassembly mechanisms, the fragmentation header comes with the following fields:

- The *datagram size* to identify the size of the IPv6 datagram.
- The *datagram tag* to identify all fragments of a single datagram.
- The *datagram offset* to identify the location of the received fragment.

In Fig. 2, the layout of the 6LoWPAN headers is illustrated.

### B. Reassembly

The receiving node, once it receives the first fragment, initiates the reassembly procedure to construct the actual IPv6 packet. In order to achieve this, the receiver checks the *datagram tag* field to identify the fragments that belong to a given IPv6 data packet, while it checks the *datagram offset* to identify the offset (i.e., location) of the received fragment within the original datagram. The size of the original unfragmented data packet as well as the size of the reassembly buffer can be identified by the *datagram size* field. Once a node receives a fragment with a certain *datagram tag* value, it starts a reassembly timer. This timeout value must be configured to 60 seconds maximum, which is the timeout in the IPv6 reassembly procedure [1]. When it expires, if the datagram has not been reconstructed, the received fragments are discarded, while the reassembly procedure is aborted.

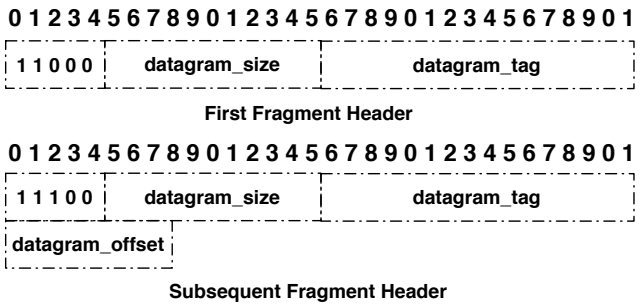



Fig. 2. The Fragmentation Header consists of 4 bytes for the first fragment and 5 bytes for subsequent fragments.

### C. Fragment Forwarding Schemes in 6LoWPAN

Furthermore, 6LoWPAN comes with two Fragment Forwarding (FF) mechanisms: ***mesh under*** and ***route over***. The first scheme operates at the adaptation layer, while the later at the network layer.

*1) Mesh under:* In mesh-under operation, the network (i.e., IP) layer does not proceed with any IP routing. In fact, the 6LoWPAN adaptation layer executes the routing and forwarding over the mesh network. To transmit a datagram to a certain destination, the EUI 64 bit address or the 16 bit short address is employed. Thus, in order to forward the received frame, the 6LoWPAN layer employs the mesh header as well as the link layer source and destination addressees that are included in the IEEE 802.15.4 header. Therefore, it is not necessary to unpack the IPv6 header.

*2) Route Over:* In the route-over scheme, all routing and forwarding operations are performed in the network layer. Considering that IEEE 802.15.4 [8] operates over mesh networks, often a routing protocol is operating at Layer 3. For example, the well known IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [6] is one of the most adopted protocols for routing packets in a multi-hop network. In RPL, each node can act as a relay for others. Thus, since the frames are routed at the network layer in [10], it is straightforward that the 6LoWPAN adaptation layer processes the received fragments at each hop to reassembly the original IPv6 data packet and then to fragment again, before forwarding to the next hop as it is depicted in Fig. 3. In order to achieve this, in addition to the hop-by-hop source and destination, the link-layer addresses of the transmitter and the final receiver should be included.

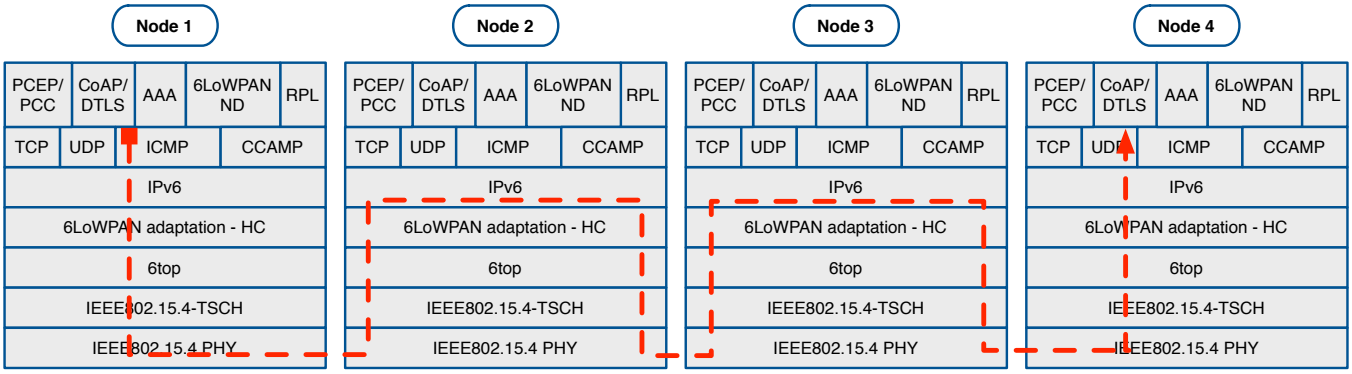



Fig. 3. L3 forwarding: IPv6 forwards the fragments over multiple hops.

## III. Per Hop Fragmentation & Reassembly Issues

6LoWPAN comes with a fragment forwarding solution that is ill-suited for a 6LoWPAN *Route-Over* (that is routed, Layer-3) mesh; in that case, reassembly and fragmentation of the entire datagram need to take place at every IPv6 hop along the path. There is thus a desire to streamline the forwarding of fragments hop-by-hop along the mesh, but this procedure presents a number of caveats that we detail in this section.

### A. Reliability

In RFC 4944 [10], when a node receives the first fragment of a given datagram, it initiates the *reassembly timer*, i.e., 60 seconds. As stated earlier, once the reassembly timer expires, if all fragments have not arrived and, thus, the whole data packet has not been reassembled, then the received fragments are discarded and the buffer is cleared. Indeed, even if only one fragment is missing at the receiver side, it can not reassemble the datagram and, consequently, it will drop the whole packet. Considering the nature of wireless communication where potential losses are common, such behavior (i.e., reassembly timer) may reduce the network reliability performance. Even though there are solutions at the IETF community [14] that propose selective acknowledgement or reset, these require new signaling and, thus, are not compatible with RFC 4944.

### B. Resource Usage

Fragmentation causes inefficient resource usage, since it introduces significant buffering requirements at the forwarding devices. Indeed, to perform complete reassembly at each hop, the forwarder requires at least 1280 *bytes* of buffer space per complete IPv6 datagram. Sensor devices are usually extremely constrained in terms of memory, which indicates that they have limited number of potential buffers for the reassembly procedure, *e.g.*, for 1 or maybe 2 complete IPv6 datagrams. Thus, given two datagrams (A and B) in circulation, the currently being reassembled datagram A will be discarded when a new fragment is received of datagram B, while the previous datagram A is not entirely reassembled yet. As a result, such scenario will introduce more losses in the wireless constrained network.

### C. Bloat & Congestion

Without flow control, buffer bloat at the core of the network or near the root becomes increasingly probable as flows are created that traverse the same intermediate nodes. Moreover, considering that RFC 4944 does not come with an end-to-end feedback loop for pacing, the larger the number of fragments the higher the chances for overloading the network and causing congestion loss. Furthermore, in a wireless network, multiple flows could traverse through intermediate routers which eventually may cause congestion. As a result, the fragments are destroyed, end-points time out, packets are retried and, thus, throughput is reduced. A potential solution could be similar to Transmission Control Protocol (TCP), dynamic windowing. However, it would require additional signaling, which will not be compatible with RFC 4944.

### D. Latency

Reassembly and fragmentation at each hop will increase the end-to-end latency [15]. This is because each device is required to "wait" for 60 seconds, which is the timeout period to perform the reassembly of the entire datagram, see Fig. 4a. Then, there is an additional computation time to fragment again the previously reassembled datagram. Thus, the more hops in the path toward the root the higher the cost to pay.

This latency can be reduced by introducing a streamlining technique whereby a node may forward each individual fragment as opposed to reassembling the whole packet first. For instance, in Fig. 4b the Forwarder 1, retransmits the fragment to Forwarder 2 in the following time slot, while in RFC 4944, the Forwarder 1 wastes two extra time slots to receive all necessary fragments. Furthermore, when a fragment forwarding process with larger than one window is used, parallel transmissions are enables, as illustrated in Fig. 4c. Equations (1) and (2) calculate the latency in timeslots of RFC 4944 and the streamlining technique, respectively:

$$T_{RFC} = (N-1)F \tag{1}$$

$$T_S = (N-1) + 2(F-1) \tag{2}$$

where N indicates the number of nodes and F the number of data fragments. It is noticeable that the latency is reduced when the streamlining technique is employed, see Fig. 5.

| | Source | Forwarder 1 | Forwarder 2 | Destination |
|---|---|---|---|---|
| T = 0 | F F F | | | |
| T = 1 | F F (F) | F | | |
| T = 2 | F (F) | F F | | |
| T = 3 | (F) | F F F | | |
| T = 4 | | F F (F) | F | |
| T = 5 | | F (F) | F F | |
| T = 6 | | (F) | F F F | |
| T = 7 | | | F F (F) | F |
| T = 8 | | | F (F) | F F |
| T = 9 | | | (F) | F F F |

(a) RFC 4944.

| | Source | Forwarder 1 | Forwarder 2 | Destination |
|---|---|---|---|---|
| T = 0 | F F F | | | |
| T = 1 | F F (F) | F | | |
| T = 2 | F F | (F) | F | |
| T = 3 | F F | | (F) | F |
| T = 4 | F (F) | F | | F |
| T = 5 | F | (F) | F | F |
| T = 6 | F | | (F) | F F |
| T = 7 | (F) | F | | F F |
| T = 8 | | (F) | F | F F |
| T = 9 | | | (F) | F F F |

(b) Streamlining with a window of size 1.

| | Source | Forwarder 1 | Forwarder 2 | Destination |
|---|---|---|---|---|
| T = 0 | F F F | | | |
| T = 1 | F F (F) | F | | |
| T = 2 | F F | (F) | F | |
| T = 3 | F (F) | F | (F) | F |
| T = 4 | F | (F) | F | F |
| T = 5 | (F) | F | (F) | F F |
| T = 6 | | (F) | F | F F |
| T = 7 | | | (F) | F F F |
| T = 8 | | | | |
| T = 9 | | | | |

(c) Streamlining with a larger window.

Fig. 4. RFC 4944 per hop reassembly and fragmentation versus streamlining behavior: considering three hops with four nodes.

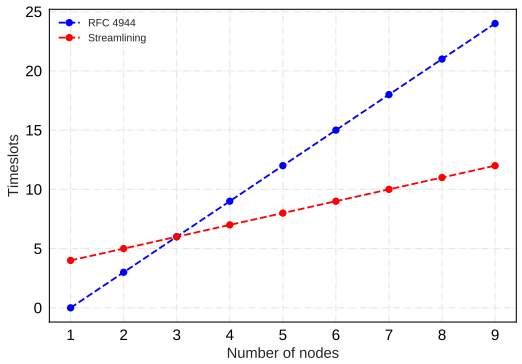



Fig. 5. Delay performance evaluation of a packet with 3 fragments.

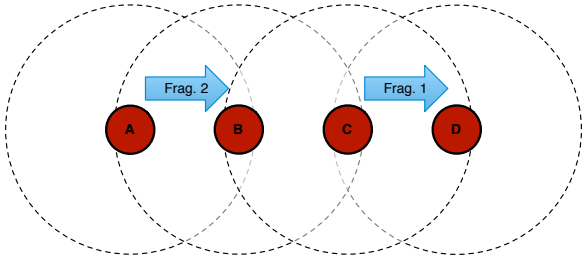



Fig. 6. Potential issues with interference.

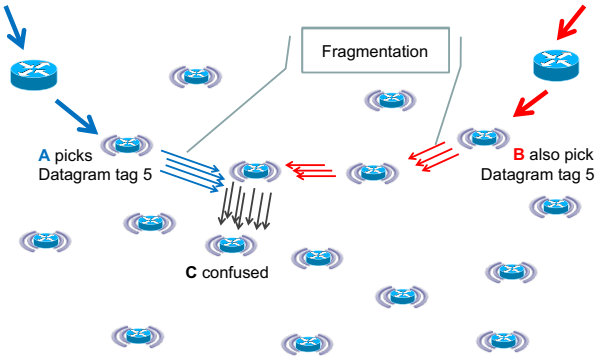



Fig. 7. Multiple simultaneous flows from different sources.

### *E. Potential Fragment Interference*

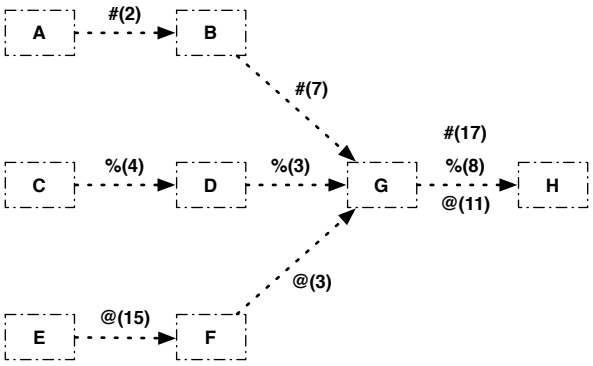
Considering a multi-hop wireless network using a single radio channel over Carrier-Sense Multiple Access (CSMA) as a Medium Access Control (MAC) protocol, where A $\rightarrow$ B $\rightarrow$ C $\rightarrow$ D, see Fig. 6, two consecutive fragments may interfere with each other (on node B in this example). Indeed, since the fragments are transmitted consecutively in a very short period of time, there is high probability that B may end up with unexpected interference when the source node A transmits its second fragment, while the relay node C forwards the first fragment to D. This is similar to the hidden node problem. This problem could be potentially mitigated to a certain degree if there was an end-to-end feedback loop. Thus, the source node would be aware of the status of each transmitted fragment along the path toward the destination. For instance, applying a minimum flow control, by introducing time delays between consecutive fragments, would allow the node A to know when to transmit its second fragment.

### *F. Implementation (Datagram tag issue)*

Finally, there are certain implementation issues regarding RFC 4944 [10]. Note that as per RFC 4944, only the first fragment of the datagram comes with the source and destination IPv6 addresses, while the following fragments are routed based on *datagram tag*, which is actually misleading, since the *tag* is unique only to the 6LoWPAN end points. Thus, two different flows may have the same *datagram tag*, which eventually causes implementation problems during the storing fragment forward state. For instance, in Fig. 7 two different traffic flows take place in the network through nodes A and B. Consider the potential case where both A and B select the same *datagram tag* 5 to initiate the fragmentation procedure. Then, at a certain point, the two flows arrive to a common intermediate node C. At this stage, C is the confused node, since it does not know where to forward the fragments.

## IV. Proposed Solutions from the Standardization Community

Fragmentation and reassembly procedures are very active topics in a number of IETF WGs such as 6lo, 6TiSCH, and LPWAN. As a result, several solutions have been proposed so far. In this section we will summarize the ongoing efforts at the IETF standardization community.




| incoming | | outgoing | |
|---|---|---|---|
| L2 src | tag | L2 dest | tag |
| B | 7 | F | 17 |
| D | 3 | F | 8 |
| F | 3 | F | 11 |
| empty | | | |

Node G's VRB table

Fig. 8. VRB table of node G: #(2), %(4) and @(15) are fragments from packets coming from nodes A, C and E, with datagram_tag configured to 2, 4 and 15, respectively.

### A. LLN Minimal Fragment Forwarding

Recently, in the 6lo WG, a design team was established to tackle the previously listed problems from [10]. They published a new Internet Draft that proposes a Virtual Reassembly Buffer (VRB) technique which reduces the end-to-end delay while improving the reliability in route-over forwarding [16]. The VRB method allows 6LoWPAN fragments to be delivered over multiple hops without requiring actual fragmentation or reassembly at each hop.

*1) Fragment Forwarding:* At its core, VRB allows a node to immediately forward a fragment that it receives, without reassembling the complete packet first. Originally, this concept was introduced in [17], where a node, once it obtained all required information about the data, may retransmit it in the form of a forwarded fragment. To do so, it is necessary that all fragments are transmitted with the same outgoing address and datagram tag, otherwise the final receiver will not be able to proceed with a full reassembly and, thus, it would discard the received fragments.

Each node in the network maintains a VRB table, similarly to a switching table with a maximum pre-allocated memory which is implementation dependent. In the beginning, all VRB tables are empty. For instance, in Fig. 8, when node G receives the first fragment from node B with datagram $tag = 7$, it analyzes the content of the fragment to find out the IPv6 destination address. If it is not the final destination, it identifies the next hop node to forward the fragment. To do so, it builds an entry in the VRB table with the following 4 fields:

- link-layer address of the previous hop
- locally unique datagram tag of the received fragment
- link-layer address of the next hop
- locally unique datagram tag for the transmitting fragment

Thus, the following fragments that match the "incoming" columns of the receiving node VRB table are forwarded based on the "outgoing" columns. As a result, based on VRB at each node, the packets are virtually reassembled and fragmented, without actually reserving additional memory.

Finally, once the last fragment is forwarded to the next hop, the node may clear the entry in its table. However, if the last fragment is never received, the node may set a timer maximum of 60 sec, as it is defined in [10], and the VRB table entry may be cleared after the expiration of the timeout period.

*2) Drawbacks:* Even though the VRB method overcomes certain issues that were presented in Section III, it comes with some limitations:

- No Fragment Recovery: by employing VRB, there is no mechanism to request a single missing fragment to proceed with full reassembly.
- No Per-Fragment Routing: all follow-up fragments follow the same path to the destination as the first fragment.
- Non-zero Packet Drop Probability: the size of the VRB table is finite, thus, in scenarios, e.g., nodes closer to the root, where a node needs to retransmit more packets that it has entries in its VRB table, the packets will be dropped.

### B. 6LoWPAN Selective Fragment Recovery

In [14], the authors present the "6LoWPAN Selective Fragment Recovery" specification that updates the fragmentation mechanism that is specified in RFC 4944 [10] for use in route-over LLNs. They propose a lightweight protocol, similar to Multiprotocol Label Switching (MPLS), to forward individual fragments across a route-over LLN. Moreover, the authors propose a recovery mechanism for potential lost fragments between LLN endpoints as well as a minimal flow control to prevent bloat in the mesh network.

*1) Fragment Forwarding:* This proposed specification extends the previously presented VRB scheme to forward fragments with no intermediate reconstruction and fragmentation of the entire datagram. In fact, only the first fragment carries the IPv6 header from the source (i.e., fragmenting end point) to the destination (reassembling end point). Note that this proposal considers that the first fragment is large enough to carry the IPv6 header to support routing decisions. Once this first fragment is successfully received, the intermediate routers between the two end points install a Label-Switched Path (LSP). Then, the remaining $n-1$ fragments are all forwarded along the path by label-switching. To enable the LSP, the datagram tag is employed as a label, that is swapped at each intermediate router. The label that is placed in the datagram tag is built based on the original source MAC address and is only valid for that source MAC, which eventually solves

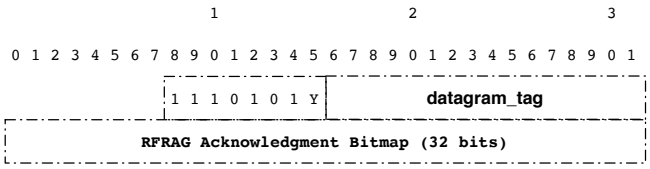



Fig. 9. RFRAG Acknowledgment Dispatch type and Header: where the $Y$ 1 bit is the ECN Echo, when it is set, the transmitter indicates that at least one of the acknowledged fragments was received with an ECN which indicates that the path is subject to congestion. Below is the RFRAG Acknowledgment Bitmap encoding, which indicates with 0 and 1 bit if the fragments are received successfully or not.

the datagram tag issue. Finally, all fragments follow the same route, while they are delivered to the reassembly end point in the same order of transmission from the fragmentation end point. As a result, the individual fragments will not be able to employ alternative paths.

*2) Fragment Recovery and Flow Control:* Furthermore, this proposal comes with a fragment recovery scheme to ensure the successful end-to-end fragment transmission. In order to achieve this, it introduces the following new Dispatch types:

- Recoverable Fragment (RFRAG).
- RFRAG with ACK Request (RFRAG-ARQ).
- RFRAG Acknowledgment (RFRAG-ACK).
- RFRAG-ACK with or without Explicit Congestion Notification (ECN) Echo (RFRAG-ECHO) [18].

The 6LoWPAN fragmenting endpoint includes RFRAG and optionally RFRAG-ARQ within the transmitted fragments to request the acknowledgement of the successful reception of the fragments. It also mitigates the potential congestion by controlling the number of outstanding fragments; the fragments that have been transmitted by the fragmenting endpoint without receiving either positive or negative confirmation.

On the other side, when the reassembling endpoint receives a fragment with the ACK Request flag set to $ON$, it reconstructs the datagram and transmits back to the fragmenting endpoint an RFRAG Acknowledgment to acknowledge the successfully received fragments. When the RFRAG-ARQ flag is set, the reassembling end point sends back the RFRAG-ACK which may optionally carry an ECN, see Fig. 9, to ensure that the original MAC address and the datagram tag are sufficient information to transmit back to the source 6LoWPAN endpoint the RFRAG Acknowledgment. Moreover, it indicates that the path to the destination is congested.

*3) Drawbacks:* LLN Fragment Forwarding and Recovery draft overcomes most of the issues that were listed in Section III, however, it comes with some limitations:

- New Dispatch types: indeed, this draft proposes new dispatch types which endangers the compatibility with the existing RFC 4944 standard.
- Additional traffic: the proposal comes with fragment recovery and congestion notification mechanisms, both of which introduce additional transmissions and bidirectional traffic. Thus, it increases the overall energy consumption in the wireless network, as well as affecting the end-to-end latency.

## V. Conclusions

Given that the IEEE 802.15.4 technology comes with a 127 $byte$ MTU, while IPv6 packets can have a 1280 $byte$ MTU, it is normally not possible to send an IPv6-based datagram over IEEE 802.15.4. Therefore, the 6LoWPAN WG standardized several mechanisms to enable such communications. Indeed, it specified an adaptation layer that defines header compression, fragmentation, reassembly and forwarding methods. In the 6LoWPAN adaptation layer, a relay forwarding a packet has to reassemble the entire packet and then fragment it again. In this paper, we first thoroughly presented the potential limits of per-hop fragmentation and reassembly as it is standardized in RFC 4944. We then presented the ongoing standardization efforts at the IETF to overcome such issues. Indeed, we summarized two leading drafts that currently are proposed at the 6lo WG. Finally, we highlighted the pros and cons of each proposal.

## References

[1] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," RFC 2460, Dec. 1998.
[2] M. Wollschlaeger, T. Sauter, and J. Jasperneite, "The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0," *IEEE Industrial Electronics Magazine*, vol. 11, no. 1, pp. 17–27, 2017.
[3] A. Paventhan, B. D. Darshini, H. Krishna, N. Pahuja, M. F. Khan, and A. Jain, "Experimental evaluation of IETF 6TiSCH in the context of Smart Grid," in *Proceedings of the 2nd IEEE World Forum on Internet of Things (WF-IoT)*, 2015, pp. 530–535.
[4] D. Dujovne, T. Watteyne, X. Vilajosana, and P. Thubert, "6TiSCH: deterministic IP-enabled industrial internet (of things)," *IEEE Communications Magazine*, vol. 52, no. 12, pp. 36–41, 2014.
[5] Z. Shelby, K. Hartke, C. Bormann, and B. Frank, "Constrained Application Protocol (CoAP)," IETF CoRE Working Group, Feb. 2011.
[6] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and A. R., "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC 6550, March 2012.
[7] P. Thubert, T. Watteyne, M. R. Palattella, X. Vilajosana, and Q. Wang, "IETF 6TSCH: Combining IPv6 Connectivity with Industrial Performance," in *Proceedings of the 7th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*, 2013, pp. 541–546.
[8] "IEEE Standard for Low-Rate Wireless Personal Area Networks (LR-WPANs)," IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011), April 2016.
[9] N. Kushalnagar, G. Montenegro, and C. Schumacher, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals," IETF RFC 4919, August 2007.
[10] G. Montenegro, N. Kushalnagar, and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks," RFC 4944, September 2007.
[11] J. Nieminen, T. Savolainen, M. Isomaki, B. Patil, Z. Shelby, and C. Gomez, "IPv6 over BLUETOOTH(R) Low Energy," RFC 7668, October 2015.
[12] A. Brandt and J. Buron, "Transmission of IPv6 Packets over ITU-T G.9959 Networks," RFC 7428, February 2015.
[13] J. Hui and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks," RFC 6282, September 2011.
[14] P. Thubert, "6LoWPAN Selective Fragment Recovery," draft-ietf-6lo-fragment-recovery-00, IETF, September 2018.
[15] A. Ludovici, A. Calveras, and J. Casademont, "Forwarding Techniques for IP Fragmented Packets in a Real 6LoWPAN Network," *Sensors*, vol. 11, no. 1, pp. 992–1008, 2011.
[16] T. Watteyne, C. Bormann, and P. Thubert, "LLN Minimal Fragment Forwarding," draft-watteyne-6lo-minimal-fragment-02, IETF, July 2018.
[17] Z. Shelby and C. Bormann, *6LoWPAN*. John Wiley & Sons, November 2009.
[18] G. Fairhurst and M. Welzl, "The Benefits of Using Explicit Congestion Notification (ECN)," RFC 8087, March. 2017.