Workgroup: lpwan Working Group
Internet-Draft:
draft-ietf-lpwan-schc-over-sigfox-16
Published: 9 December 2022
Intended Status: Standards Track
Expires: 12 June 2023
Authors: JC. Zuniga C. Gomez
Universitat Politecnica de Catalunya
S. Aguilar
Universitat Politecnica de Catalunya
L. Toutain S. Cespedes
IMT-Atlantique Concordia University
D. Wistuba
NIC Labs, Universidad de Chile
J. Boite
Unabiz - Sigfox is now a Unabiz technology

SCHC over Sigfox LPWAN

Abstract

The Static Context Header Compression and fragmentation (SCHC) specification (RFC8724) describes a generic framework for application header compression and fragmentation modes designed for Low Power Wide Area Network (LPWAN) technologies. The present document defines a profile of SCHC (RFC8724) over Sigfox LPWAN, and provides optimal parameter values and modes of operation.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 12 June 2023.

Copyright Notice

Table of Contents

1. Introduction

The Generic Framework for Static Context Header Compression and Fragmentation (SCHC) specification [RFC8724] can be used on top of all the four LPWAN technologies defined in [RFC8376]. These LPWANs have similar characteristics such as star-oriented topologies,

network architecture, connected devices with built-in applications, etc.

SCHC offers a great level of flexibility to accommodate all these LPWAN technologies. Even though there are a great number of similarities between them, some differences exist with respect to the transmission characteristics, payload sizes, etc. Hence, there are optimal parameters and modes of operation that can be used when SCHC is used on top of a specific LPWAN technology.

Sigfox is an LPWAN technology that offers energy-efficient connectivity for devices at a very low cost. Sigfox brings a worldwide network composed of Base Stations that receive short (12 bytes) uplink messages sent by devices over the long-range Sigfox radio protocol. Base Stations then forward messages to the Sigfox Cloud infrastructure for further processing and final delivery to the customer. With SCHC functionalities, the Sigfox network offers more reliable communications (recovery of lost messages) and is able to convey extended-size payloads (fragmentation/reassembly).

This document describes the recommended parameters, settings, and modes of operation to be used when SCHC is implemented over a Sigfox LPWAN. This set of parameters are also known as a "SCHC over Sigfox profile" or simply "SCHC/Sigfox."

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

It is assumed that the reader is familiar with the terms and mechanisms defined in [RFC8376] and in [RFC8724].

3. SCHC over Sigfox

The Generic SCHC Framework described in [RFC8724] takes advantage of previous knowledge of traffic flows existing in LPWAN applications to avoid context synchronization.

Contexts need to be stored and pre-configured on both ends. This can be done either by using a provisioning protocol, by out of band means, or by pre-provisioning them (e.g., at manufacturing time). The way contexts are configured and stored on both ends is out of the scope of this document.

3.1. Network Architecture

Figure 1 represents the architecture for Compression/Decompression (C/D) and Fragmentation/Reassembly (F/R) based on the terminology defined in [RFC8376], where the Radio Gateway (RGW) is a Sigfox Base Station and the Network Gateway (NGW) is the Sigfox cloud-based Network.

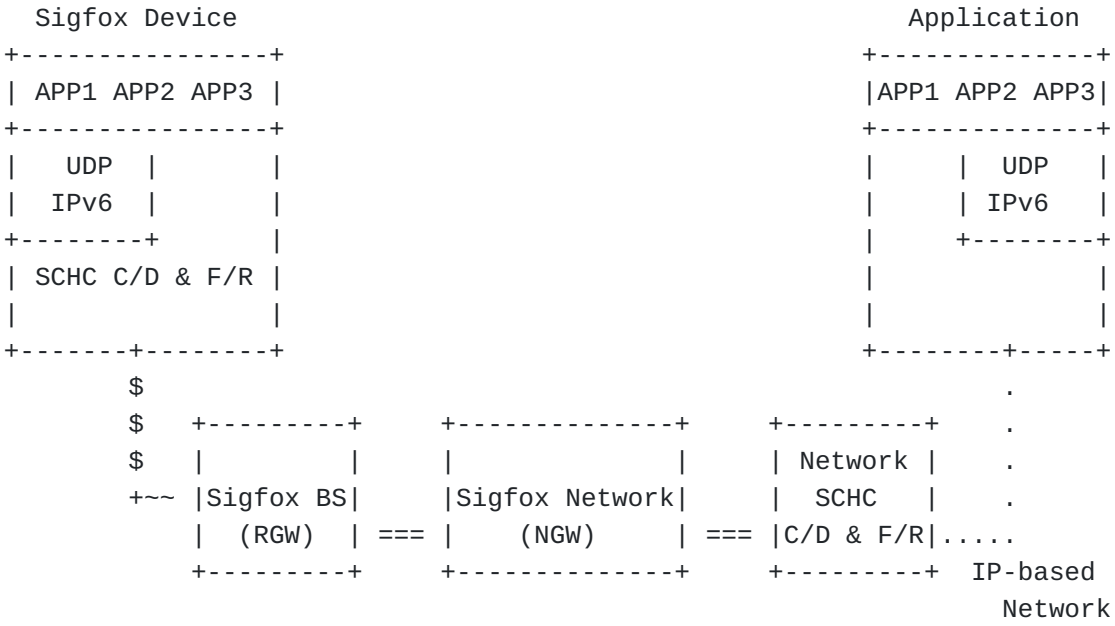


Figure 1: Network Architecture

In the case of the global Sigfox Network, RGWs (or Base Stations) are distributed over multiple countries wherever the Sigfox LPWAN service is provided. The NGW (or cloud-based Sigfox Core Network) is a single entity that connects to all RGWs (Sigfox Base Stations) in the world, providing hence a global single star network topology.

The Sigfox Device sends application packets that are compressed and/or fragmented by a SCHC C/D + F/R to reduce headers size and/or fragment the packet. The resulting SCHC Message is sent over a layer two (L2) Sigfox frame to the Sigfox Base Stations, which then forward the SCHC Message to the Network Gateway (NGW). The NGW then delivers the SCHC Message and associated gathered metadata to the Network SCHC C/D + F/R.

The Sigfox Network (NGW) communicates with the Network SCHC C/D + F/R for compression/decompression and/or for fragmentation/reassembly. The Network SCHC C/D + F/R shares the same set of rules as the Dev SCHC C/D + F/R. The Network SCHC C/D + F/R can be collocated with the NGW or it could be located in a different place, as long as a tunnel or secured communication is established between the NGW and the SCHC C/D + F/R functions. After decompression and/or reassembly, the packet can be forwarded over the Internet to one (or several) LPWAN Application Server(s) (App).

The SCHC C/D + F/R processes are bidirectional, so the same principles are applicable on both Uplink (UL) and Downlink (DL).

3.2. Uplink

Uplink Sigfox transmissions occur in repetitions over different times and frequencies. Besides time and frequency diversities, the Sigfox network also provides space diversity, as potentially an Uplink message will be received by several base stations.

Since all messages are self-contained and base stations forward all these messages back to the same Sigfox Network, multiple input copies can be combined at the NGW providing for extra reliability based on the triple diversity (i.e., time, space and frequency).

A detailed description of the Sigfox Radio Protocol can be found in [sigfox-spec].

Messages sent from the Device to the Network are delivered by the Sigfox network (NGW) to the Network SCHC C/D + F/R through a callback/API with the following information:

* Device ID
* Message Sequence Number
* Message Payload
* Message Timestamp
* Device Geolocation (optional)
* RSSI (optional)
* Device Temperature (optional)
* Device Battery Voltage (optional)

The Device ID is a globally unique identifier assigned to the Device, which is included in the Sigfox header of every message. The

Message Sequence Number is a monotonically increasing number identifying the specific transmission of this Uplink message, and it is also part of the Sigfox header. The Message Payload corresponds to the payload that the Device has sent in the Uplink transmission.

The Message Timestamp, Device Geolocation, RSSI, Device Temperature and Device Battery Voltage are metadata parameters provided by the Network.

A detailed description of the Sigfox callbacks/APIs can be found in [sigfox-callbacks].

Only messages that have passed the L2 Cyclic Redundancy Check (CRC) at network reception are delivered by the Sigfox Network to the Network SCHC C/D + F/R.

The L2 Word Size used by Sigfox is 1 byte (8 bits).

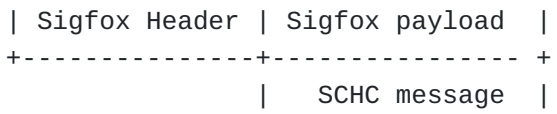


Figure 2: SCHC Message in Sigfox

Figure 2 shows a SCHC Message sent over Sigfox, where the SCHC Message could be a full SCHC Packet (e.g., compressed) or a SCHC Fragment (e.g., a piece of a bigger SCHC Packet).

3.3. Downlink

Downlink transmissions are Device-driven and can only take place following an Uplink communication that so indicates. Hence, a Sigfox Device explicitly indicates its intention to receive a Downlink message using a Downlink request flag when sending the preceding Uplink message to the network. The Downlink request flag is part of the Sigfox protocol headers. After completing the Uplink transmission, the Device opens a fixed window for Downlink reception. The delay and duration of the reception opportunity window have fixed values. If there is a Downlink message to be sent for this given Device (e.g., either a response to the Uplink message or queued information waiting to be transmitted), the network transmits this message to the Device during the reception window. If no message is received by the Device after the reception opportunity window has elapsed, the Device closes the reception window opportunity and gets back to the normal mode (e.g., continue Uplink transmissions, sleep, stand-by, etc.)

When a Downlink message is sent to a Device, a reception acknowledgement is generated by the Device and sent back to the Network through the Sigfox radio protocol and reported in the Sigfox Network backend.

A detailed description of the Sigfox Radio Protocol can be found in [sigfox-spec] and a detailed description of the Sigfox callbacks/ APIs can be found in [sigfox-callbacks]. Downlink request flag can be included in the information exchange between the Sigfox Network and Network SCHC.

3.4. SCHC-ACK on Downlink

As explained previously, Downlink transmissions are Device-driven and can only take place following a specific Uplink transmission that indicates and allows a following Downlink opportunity. For this reason, when SCHC bi-directional services are used (e.g., Ack-on-Error fragmentation mode) the SCHC protocol implementation needs to consider the times when a Downlink message (e.g., SCHC-ACK) can be sent and/or received.

For the Uplink ACK-on-Error fragmentation mode, a Downlink opportunity MUST be indicated by the last fragment of every window, which is signal by a specific the Fragment Compressed Number (FCN) value, i.e., FCN = All-0, or FCN = All-1. The FCN is the tile index in an specific window. FCN and window number combination allows to uniquely identified a SCHC Fragment as explained in [RFC8724]. The Device sends the fragments in sequence and, after transmitting the FCN = All-0 or FCN = All-1, it opens up a reception opportunity. The Network SCHC can then decide to respond at that opportunity (or wait for a further one) with a SCHC-ACK indicating in case there are missing fragments from the current or previous windows. If there is no SCHC-ACK to be sent, or if the network decides to wait for a further Downlink transmission opportunity, then no Downlink transmission takes place at that opportunity and after a timeout the Uplink transmissions continue. Intermediate SCHC fragments with FCN different from All-0 or All-1 MUST NOT use the Downlink request flag to request a SCHC-ACK.

3.5. SCHC Rules

The RuleID MUST be included in the SCHC header. The total number of rules to be used affects directly the Rule ID field size, and therefore the total size of the fragmentation header. For this reason, it is recommended to keep the number of rules that are defined for a specific device to the minimum possible.

RuleIDs can be used to differentiate data traffic classes (e.g., QoS, control vs. data, etc.), and data sessions. They can also be

used to interleave simultaneous fragmentation sessions between a Device and the Network.

3.6. Fragmentation

The SCHC specification [RFC8724] defines a generic fragmentation functionality that allows sending data packets or files larger than the maximum size of a Sigfox payload. The functionality also defines a mechanism to send reliably multiple messages, by allowing to resend selectively any lost fragments.

The SCHC fragmentation supports several modes of operation. These modes have different advantages and disadvantages depending on the specifics of the underlying LPWAN technology and application Use Case. This section describes how the SCHC fragmentation functionality should optimally be implemented when used over a Sigfox LPWAN for the most typical Use Case applications.

As described in section 8.2.3 of [RFC8724], the integrity of the fragmentation-reassembly process of a SCHC Packet MUST be checked at the receiver end. Since only Uplink/Downlink messages/fragments that have passed the Sigfox CRC-check are delivered to the Network/Sigfox Device SCHC C/D + F/R, integrity can be guaranteed when no consecutive messages are missing from the sequence and all FCN bitmaps are complete. With this functionality in mind, and in order to save protocol and processing overhead, the use of a Reassembly Check Sequence (RCS) as described in Section 3.6.1.5 MUST be used.

3.6.1. Uplink Fragmentation

Sigfox Uplink transmissions are completely asynchronous and take place in any random frequency of the allowed Uplink bandwidth allocation. In addition, devices may go to deep sleep mode, and then wake up and transmit whenever there is a need to send information to the network, as there is no need to perform any network attachment, synchronization, or other procedure before transmitting a data packet.

Since Uplink transmissions are asynchronous, a SCHC fragment can be transmitted at any given time by the Device. Sigfox Uplink messages are fixed in size, and as described in [RFC8376] they can carry 0-12 bytes payload. Hence, a single SCHC Tile size per fragmentation mode can be defined so that every Sigfox message always carries one SCHC Tile.

When the ACK-on-Error mode is used for Uplink fragmentation, the SCHC Compound ACK defined in [I-D.ietf-lpwan-schc-compound-ack]) MUST be used in the Downlink responses.

3.6.1.1. SCHC-Sender Abort

As defined in [RFC8724], a SCHC-Sender Abort can be triggered when the number of SCHC ACK REQ attempts is greater than or equal to MAX_ACK_REQUESTS. In the case of SCHC over Sigfox, a SCHC-Sender Abort MUST be sent if the number of repeated All-1s sent in sequence, without a Compound ACK reception inbetween, is greater than or equal to MAX_ACK_REQUESTS.

3.6.1.2. SCHC Receiver-Abort

As defined in [RFC8724], a SCHC Receiver-Abort is triggered when the receiver has no RuleID and DTag pairs available for a new session. In the case of SCHC/Sigfox a SCHC Receiver-Abort MUST be sent if, for a single device, all the RuleIDs are being processed by the receiver (i.e., have an active session) at a certain time and a new one is requested, or if the RuleID of the fragment is not valid.

A SCHC Receiver-Abort MUST be triggered when the Inactivity Timer expires.

MAX_ACK_REQUESTS can be increase when facing high error rates.

Although a SCHC Receiver-Abort can be triggered at any point in time, a SCHC Receiver-Abort Downlink message MUST only be sent when there is a Downlink transmission opportunity.

3.6.1.3. Single-byte SCHC Header for Uplink Fragmentation

3.6.1.3.1. Uplink No-ACK Mode: Single-byte SCHC Header

Single-byte SCHC Header No-ACK mode SHOULD be used for transmitting short, non-critical packets that require fragmentation and do not require full reliability. This mode can be used by Uplink-only devices that do not support Downlink communications, or by bidirectional devices when they send non-critical data. Note that sending non-critical data by using a reliable fragmentation mode (which is only possible for bidirectional devices) may incur unnecessary overhead.

Since there are no multiple windows in the No-ACK mode, the W bit is not present. However, it MUST use the FCN field to indicate the size of the data packet. In this sense, the data packet would need to be split into X fragments and, similarly to the other fragmentation modes, the first transmitted fragment would need to be marked with FCN = X-1. Consecutive fragments MUST be marked with decreasing FCN values, having the last fragment marked with FCN = (All-1). Hence, even though the No-ACK mode does not allow recovering missing fragments, it allows indicating implicitly the size of the expected packet to the Network and hence detect at the receiver side whether

all fragments have been received or not. In case the FCN field is not used to indicate the size of the data packet, the Network can detect whether all fragments have been received or not by using the integrity check.

When using the Single-byte SCHC Header for Uplink Fragmentation, the Fragmentation Header MUST be of 8 bit size, and it is composed as follows:

*RuleID size: 3 bits

*DTag size (T): 0 bit

*Fragment Compressed Number (FCN) size (N): 5 bits

*As per [RFC8724], in the No-ACK mode the W (window) field is not present.

*Regular tile size: 11 bytes

*All-1 tile size: 0 to 10 bytes

*Inactivity Timer: Application-dependent. The default value is 12 hours.

*RCS size: 5 bits

The maximum SCHC Packet size is of 340 bytes.

Section Section 3.7.1 presents SCHC Fragment format examples and Section Section 4.1 provides fragmentation examples, using Single-byte SCHC Header No-ACK mode.

3.6.1.3.2. Uplink ACK-on-Error Mode: Single-byte SCHC Header

ACK-on-Error with single-byte header SHOULD be used for short to medium size packets that need to be sent reliably. ACK-on-Error is optimal for reliable SCHC Packet transmission over Sigfox transmissions, since it leads to a reduced number of ACKs in the lower capacity Downlink channel. Also, Downlink messages can be sent asynchronously and opportunistically. In contrast, ACK-Always would not minimize the number of ACKs, and No-ACK would not allow reliable transmission.

Allowing transmission of packets/files up to 300 bytes long, the SCHC Uplink Fragmentation Header size is of 8 bits in size and is composed as follows:

*Rule ID size: 3 bits

*DTag size (T): 0 bit

*Window index (W) size (M): 2 bits

*Fragment Compressed Number (FCN) size (N): 3 bits

*MAX_ACK_REQUESTS: 5

*WINDOW_SIZE: 7 (with a maximum value of FCN=0b110)

*Regular tile size: 11 bytes

*All-1 tile size: 0 to 10 bytes

*Retransmission Timer: Application-dependent. The default value is 12 hours.

*Inactivity Timer: Application-dependent. The default value is 12 hours.

*RCS size: 3 bits

Section Section 3.7.2 presents SCHC Fragment format examples and Section Section 4.2 provides fragmentation examples, using ACK-on-Error with single-byte header.

3.6.1.4. Two-byte SCHC Header for Uplink Fragmentation

ACK-on-Error with two-byte header SHOULD be used for medium-large size packets that need to be sent reliably. ACK-on-Error is optimal for reliable SCHC Packet transmission over Sigfox, since it leads to a reduced number of ACKs in the lower capacity Downlink channel. Also, Downlink messages can be sent asynchronously and opportunistically. In contrast, ACK-Always would not minimize the number of ACKs, and No-ACK would not allow reliable transmission.

3.6.1.4.1. Uplink ACK-on-Error Mode: Two-byte SCHC Header Option 1

In order to allow transmission of medium-large packets/files up to 480 bytes long, the SCHC Uplink Fragmentation Header size is of 16 bits in size and composed as follows:

*Rule ID size is: 6 bits

*DTag size (T) is: 0 bit

*Window index (W) size (M): 2 bits

*Fragment Compressed Number (FCN) size (N): 4 bits.

*MAX_ACK_REQUESTS: 5

*WINDOW_SIZE: 12 (with a maximum value of FCN=0b1011)

*Regular tile size: 10 bytes

*All-1 tile size: 1 to 10 bytes

*Retransmission Timer: Application-dependent. The default value is 12 hours.

*Inactivity Timer: Application-dependent. The default value is 12 hours.

*RCS size: 4 bits

Note that WINDOW_SIZE is limited to 12. This because, 4 windows (M = 2) with bitmaps of size 12 can be fitted in a single SCHC Compound ACK.

Section Section 3.7.3 presents SCHC Fragment format examples, using ACK-on-Error with two-byte header Option 1.

3.6.1.4.2. Uplink ACK-on-Error Mode: Two-byte SCHC Header Option 2

In order to allow transmission of very large packets/files up to 2400 bytes long, the SCHC Uplink Fragmentation Header size is of be 16 bits in size and composed as follows:

*Rule ID size is: 8 bits

*DTag size (T) is: 0 bit

*Window index (W) size (M): 3 bits

*Fragment Compressed Number (FCN) size (N): 5 bits.

*MAX_ACK_REQUESTS: 5

*WINDOW_SIZE: 31 (with a maximum value of FCN=0b11110)

*Regular tile size: 10 bytes

*All-1 tile size: 0 to 9 bytes

*Retransmission Timer: Application-dependent. The default value is 12 hours.

*Inactivity Timer: Application-dependent. The default value is 12 hours.

*RCS size: 5 bits

Section Section 3.7.4 presents SCHC Fragment format examples, using ACK-on-Error with two-byte header Option 1.

3.6.1.5. All-1 and RCS behaviour

For ACK-on-Error, as defined in [RFC8724], it is expected that the last SCHC fragment of the last window will always be delivered with an All-1 FCN. Since this last window may not be full (i.e., it may be comprised of less than WINDOW_SIZE fragments), an All-1 fragment may follow a value of FCN higher than 1 (0b01). In this case, the receiver cannot determine from the FCN values alone whether there are or not any missing fragments right before the All-1 fragment.

For Rules where the number of fragments in the last window is unknown, an RCS field MUST be used, indicating the number of fragments in the last window, including the All-1. With this RCS value, the receiver can detect if there are missing fragments before the All-1 and hence construct the corresponding SCHC ACK Bitmap accordingly, and send it in response to the All-1.

3.6.2. Downlink Fragmentation

In some LPWAN technologies, as part of energy-saving techniques, Downlink transmission is only possible immediately after an Uplink transmission. This allows the device to go in a very deep sleep mode and preserve battery, without the need to listen to any information from the network. This is the case for Sigfox-enabled devices, which can only listen to Downlink communications after performing an Uplink transmission and requesting a Downlink.

When there are fragments to be transmitted in the Downlink, an Uplink message is required to trigger the Downlink communication. In order to avoid potentially high delay for fragmented datagram transmission in the Downlink, the fragment receiver MAY perform an Uplink transmission as soon as possible after reception of a Downlink fragment that is not the last one. Such Uplink transmission MAY be triggered by sending a SCHC message, such as a SCHC ACK. However, other data messages can equally be used to trigger Downlink communications. The fragment receiver SHOULD send an Uplink transmission (e.g., empty message) and request a Downlink every 24 hours when no SCHC session is started. The use or not of this Uplink transmission (and the transmission rate, if used) will depend on application specific requirements.

Sigfox Downlink messages are fixed in size, and as described in [RFC8376] they can carry up to 8 bytes payload. Hence, a single SCHC Tile size per mode can be defined so that every Sigfox message always carries one SCHC Tile.

For reliable Downlink fragment transmission, the ACK-Always mode SHOULD be used. Note that ACK-on-Error does not guarantee Uplink feedback (since no SCHC ACK will be sent when no errors occur in a window), and No-ACK would not allow reliable transmission.

The SCHC Downlink Fragmentation Header size is of 8 bits in size and is composed as follows:

- *RuleID size: 3 bits
- *DTag size (T): 0 bit
- *Window index (W) size (M) is: 0 bit
- *Fragment Compressed Number (FCN) size (N): 5 bits
- *MAX_ACK_REQUESTS: 5
- *WINDOW_SIZE: 31 (with a maximum value of FCN=0b11110)
- *Regular tile size: 7 bytes
- *All-1 tile size: 0 to 6 bytes
- *Retransmission Timer: Application-dependent. The default value is 12 hours.
- *Inactivity Timer: Application-dependent. The default value is 12 hours.
- *RCS size: 5 bits

3.7. SCHC-over-Sigfox F/R Message Formats

This section depicts the different formats of SCHC Fragment, SCHC ACK (including the SCHC Compound ACK defined in [I-D.ietf-lpwan-schc-compound-ack]), and SCHC Abort used in SCHC over Sigfox.

3.7.1. Uplink No-ACK Mode: Single-byte SCHC Header

3.7.1.1. Regular SCHC Fragment

Figure 3 shows an example of a regular SCHC fragment for all fragments except the last one. As tiles are of 11 bytes, padding MUST NOT be added. The penultimate tile of a SCHC Packet is of regular size.

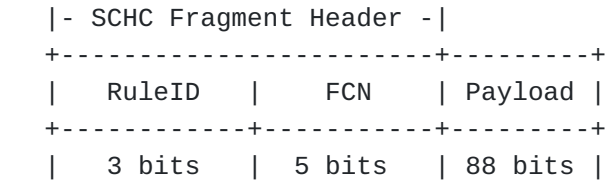


Figure 3: Regular SCHC Fragment format for all fragments except the last one

3.7.1.2. All-1 SCHC Fragment

Figure 4 shows an example of the All-1 message. The All-1 message MAY contain the last tile of the SCHC Packet. Padding MUST NOT be added, as the resulting size is L2-word-multiple.

The All-1 messages Fragment Header includes a 5-bit RCS, and 3 bits are added as padding to complete two bytes. The payload size of the All-1 message ranges from 0 to 80 bits.

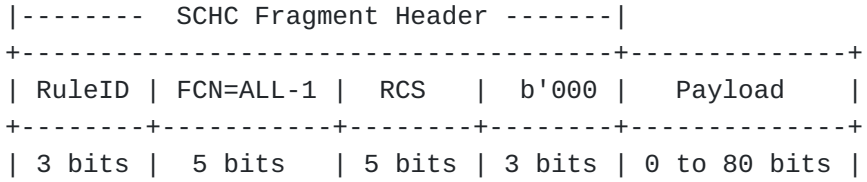


Figure 4: All-1 SCHC Message format with last tile

As per [RFC8724] the All-1 must be distinguishable from a SCHC Sender-Abort message (with same Rule ID, and N values). The All-1 MAY have the last tile of the SCHC Packet. The SCHC Sender-Abort message header size is of 1 byte, with no padding bits.

For the All-1 message to be distinguishable from the Sender-Abort message, the Sender-Abort message MUST be of 1 byte (only header with no padding). This way, the minimum size of the All-1 is 2 bytes, and the Sender-Abort message is 1 byte.

3.7.1.3. SCHC Sender-Abort Message format

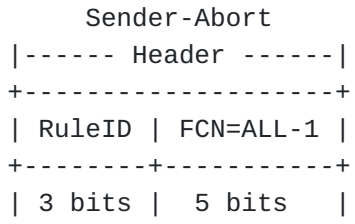


Figure 5: SCHC Sender-Abort message format

3.7.2. Uplink ACK-on-Error Mode: Single-byte SCHC Header

3.7.2.1. Regular SCHC Fragment

Figure 6 shows an example of a regular SCHC fragment for all fragments except the last one. As tiles are of 11 bytes, padding MUST NOT be added.

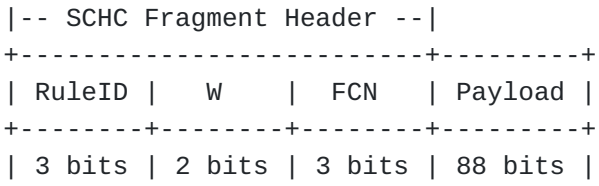


Figure 6: Regular SCHC Fragment format for all fragments except the last one

The SCHC ACK REQ MUST NOT be used, instead the All-1 SCHC Fragment MUST be used to request a SCHC ACK from the receiver (Network SCHC). As per [RFC8724], the All-0 message is distinguishable from the SCHC ACK REQ (All-1 message). The penultimate tile of a SCHC Packet is of regular size.

3.7.2.2. All-1 SCHC Fragment

Figure 7 shows an example of the All-1 message. The All-1 message MAY contain the last tile of the SCHC Packet. Padding MUST NOT be added, as the resulting size is L2-word-multiple.

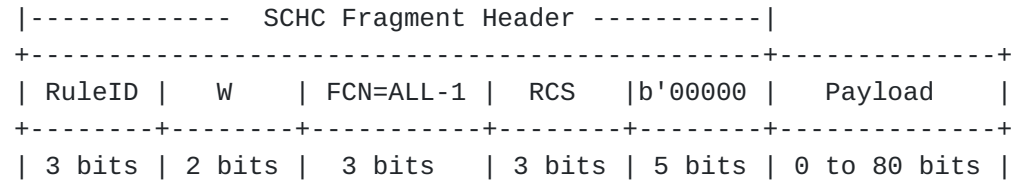


Figure 7: All-1 SCHC Message format with last tile

As per [RFC8724] the All-1 must be distinguishable from a SCHC Sender-Abort message (with same Rule ID, M, and N values). The All-1 MAY have the last tile of the SCHC Packet. The SCHC Sender-Abort message header size is of 1 byte, with no padding bits.

For the All-1 message to be distinguishable from the Sender-Abort message, the Sender-Abort message MUST be of 1 byte (only header with no padding). This way, the minimum size of the All-1 is 2 bytes, and the Sender-Abort message is 1 byte.

3.7.2.3. SCHC ACK Format

Figure 8 shows the SCHC ACK format when all fragments have been correctly received (C=1). Padding MUST be added to complete the 64-bit Sigfox Downlink frame payload size.

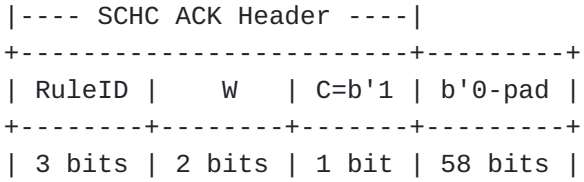


Figure 8: SCHC Success ACK message format

In case SCHC fragment losses are found in any of the windows of the SCHC Packet (C=0), the SCHC Compound ACK defined in [I-D.ietf-lpwan-schc-compound-ack] MUST be used. The SCHC Compound ACK message format is shown in Figure 9.

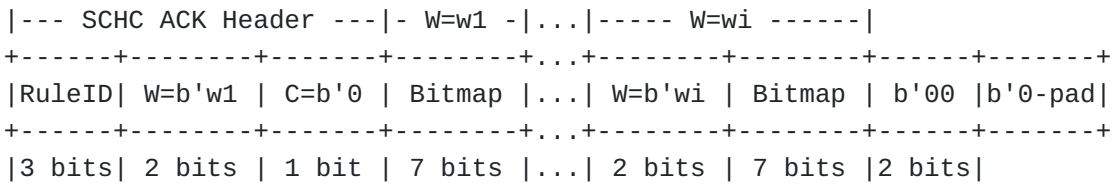


Figure 9: SCHC Compound ACK message format

3.7.2.4. SCHC Sender-Abort Message format

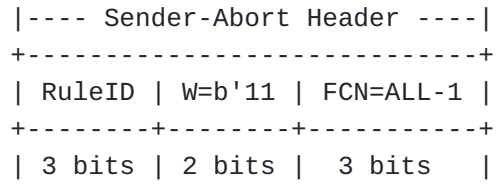


Figure 10: SCHC Sender-Abort message format

3.7.2.5. SCHC Receiver-Abort Message format

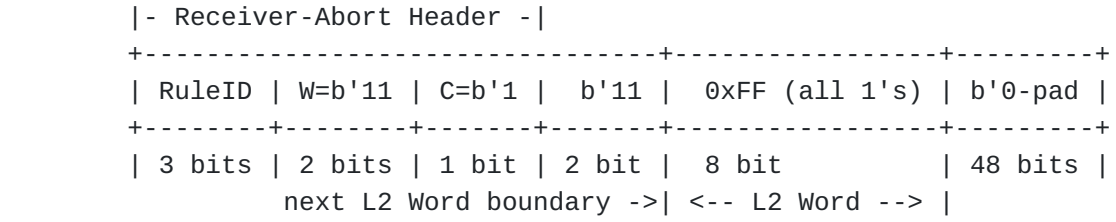


Figure 11: SCHC Receiver-Abort message format

3.7.3. Uplink ACK-on-Error Mode: Two-byte SCHC Header Option 1

3.7.3.1. Regular SCHC Fragment

Figure 12 shows an example of a regular SCHC fragment for all fragments except the last one. The penultimate tile of a SCHC Packet is of the regular size.

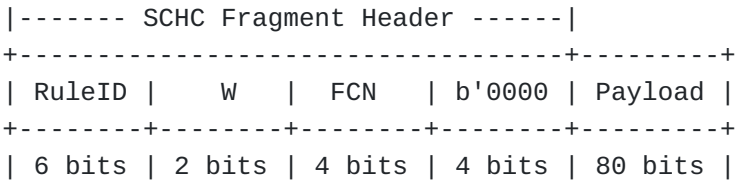


Figure 12: Regular SCHC Fragment format for all fragments except the last one

The SCHC ACK REQ MUST NOT be used, instead the All-1 SCHC Fragment MUST be used to request a SCHC ACK from the receiver (Network SCHC). As per [RFC8724], the All-0 message is distinguishable from the SCHC ACK REQ (All-1 message).

3.7.3.2. All-1 SCHC Fragment

Figure 13 shows an example of the All-1 message. The All-1 message MUST contain the last tile of the SCHC Packet.

The All-1 message Fragment Header contains a RCS of 4 bits to complete the two-byte size. The size of the last tile ranges from 8 to 80 bits.

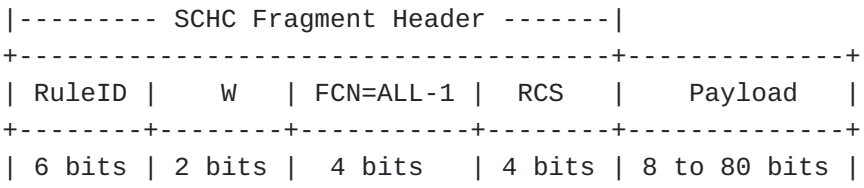


Figure 13: All-1 SCHC message format with last tile

As per [RFC8724] the All-1 must be distinguishable from the a SCHC Sender-Abort message (with same Rule ID, M and N values). The All-1 MUST have the last tile of the SCHC Packet, that MUST be of at least 1 byte. The SCHC Sender-Abort message header size is of 2 byte, with no padding bits.

For the All-1 message to be distinguishable from the Sender-Abort message, the Sender-Abort message MUST be of 2 byte (only header with no padding). This way, the minimum size of the All-1 is 3 bytes, and the Sender-Abort message is 2 bytes.

3.7.3.3. SCHC ACK Format

Figure 14 shows the SCHC ACK format when all fragments have been correctly received (C=1). Padding MUST be added to complete the 64-bit Sigfox Downlink frame payload size.

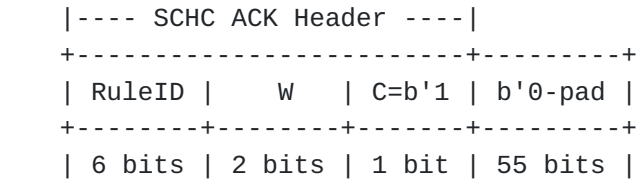
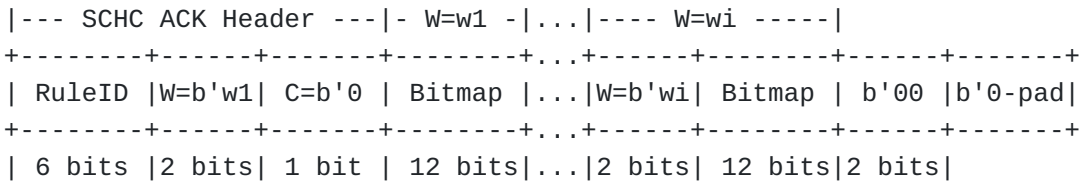


Figure 14: SCHC Success ACK message format

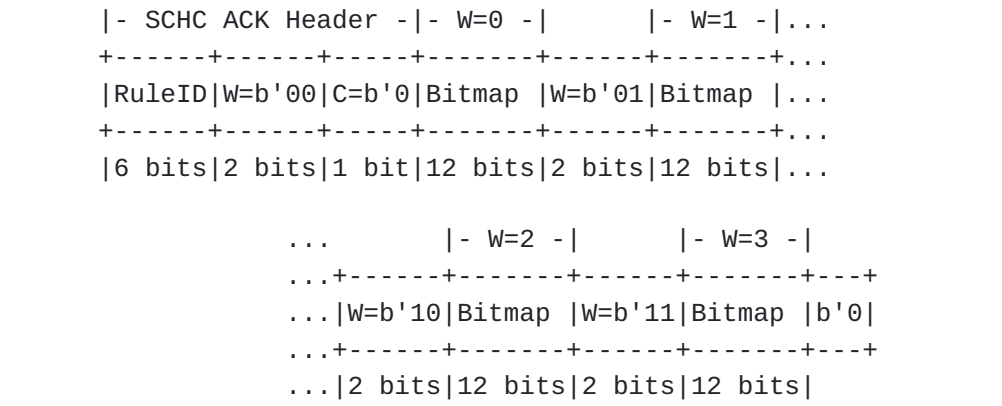
The SCHC Compound ACK message MUST be used in case SCHC fragment losses are found in any window of the SCHC Packet (C=0). The SCHC Compound ACK message format is shown in Figure 15. The SCHC Compound ACK can report up to 4 windows with losses. as shown in Figure 16.

When sent in the Downlink, the SCHC Compound ACK MUST be 0 padded (Padding bits must be 0) to complement the 64 bits required by the Sigfox payload.



Losses are found in windows W = w1,...,wi; where w1<w2<...<wi

Figure 15: SCHC Compound ACK message format



Losses are found in windows W = w1,...,wi; where w1<w2<...<wi

Figure 16: SCHC Compound ACK message format example with losses in all windows

3.7.3.4. SCHC Sender-Abort Messages

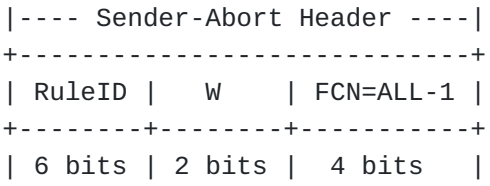


Figure 17: SCHC Sender-Abort message format

3.7.3.5. SCHC Receiver-Abort Message

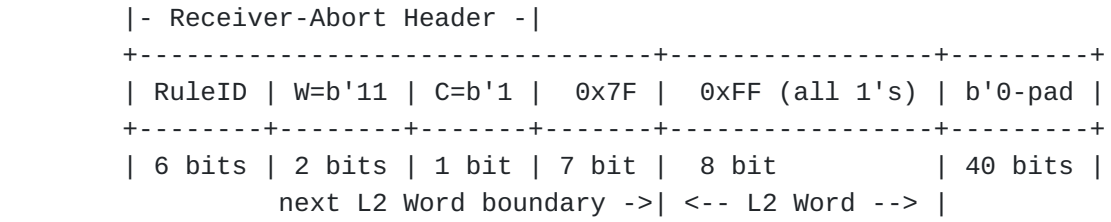


Figure 18: SCHC Receiver-Abort message format

3.7.4. Uplink ACK-on-Error Mode: Two-byte SCHC Header Option 2

3.7.4.1. Regular SCHC Fragment

Figure 19 shows an example of a regular SCHC fragment for all fragments except the last one. The penultimate tile of a SCHC Packet is of the regular size.

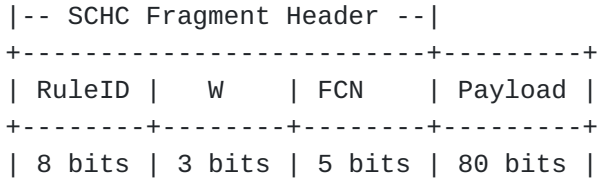


Figure 19: Regular SCHC Fragment format for all fragments except the last one

The SCHC ACK REQ MUST NOT be used, instead the All-1 SCHC Fragment MUST be used to request a SCHC ACK from the receiver (Network SCHC). As per [RFC8724], the All-0 message is distinguishable from the SCHC ACK REQ (All-1 message).

3.7.4.2. All-1 SCHC Fragment

Figure 20 shows an example of the All-1 message. The All-1 message MAY contain the last tile of the SCHC Packet.

The All-1 message Fragment Header contains an RCS of 5 bits, and 3 padding bits to complete a 3-byte Fragment Header. The size of the last tile, if present, ranges from 8 to 72 bits.

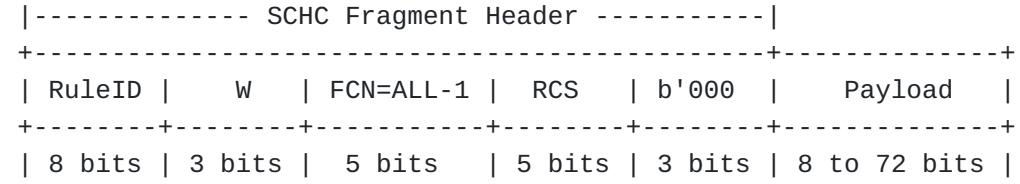


Figure 20: All-1 SCHC message format with last tile

As per [RFC8724] the All-1 must be distinguishable from the a SCHC Sender-Abort message (with same Rule ID, M and N values). The SCHC Sender-Abort message header size is of 2 byte, with no padding bits.

For the All-1 message to be distinguishable from the Sender-Abort message, the Sender-Abort message MUST be of 2 byte (only header with no padding). This way, the minimum size of the All-1 is 3 bytes, and the Sender-Abort message is 2 bytes.

3.7.4.3. SCHC ACK Format

Figure 21 shows the SCHC ACK format when all fragments have been correctly received (C=1). Padding MUST be added to complete the 64-bit Sigfox Downlink frame payload size.

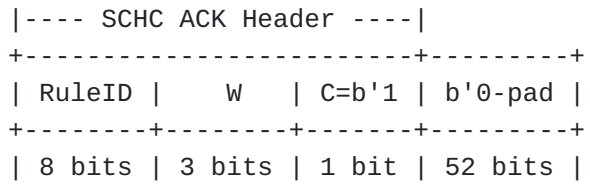


Figure 21: SCHC Success ACK message format

The SCHC Compound ACK message MUST be used in case SCHC fragment losses are found in any window of the SCHC Packet (C=0). The SCHC Compound ACK message format is shown in Figure 22. The SCHC Compound ACK can report up to 3 windows with losses.

When sent in the Downlink, the SCHC Compound ACK MUST be 0 padded (Padding bits must be 0) to complement the 64 bits required by the Sigfox payload.

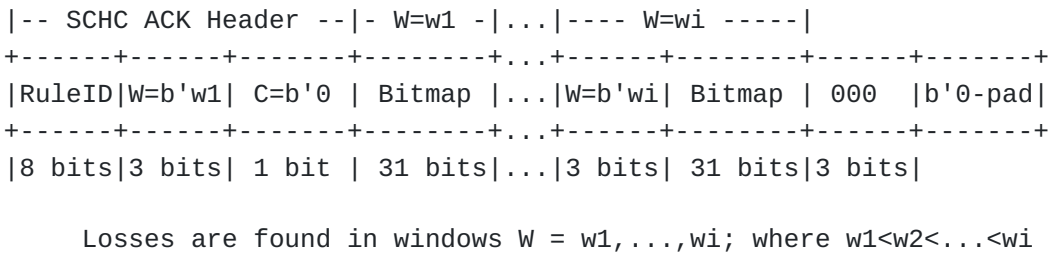


Figure 22: SCHC Compound ACK message format

3.7.4.4. SCHC Sender-Abort Messages

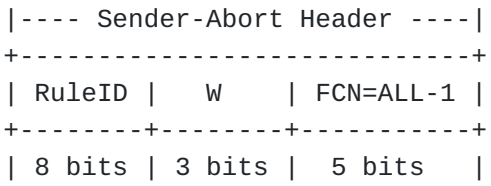


Figure 23: SCHC Sender-Abort message format

3.7.4.5. SCHC Receiver-Abort Message

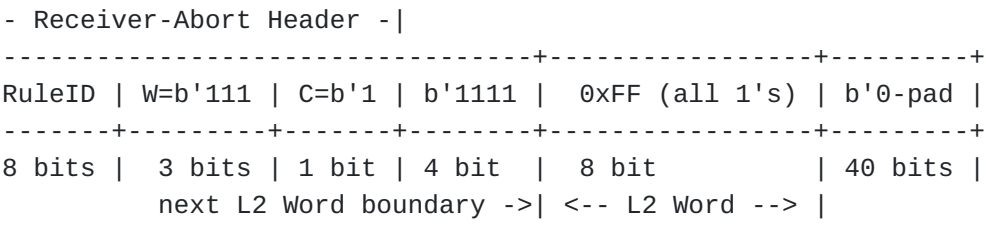


Figure 24: SCHC Receiver-Abort message format

3.7.5. Downlink ACK-Always Mode: Single-byte SCHC Header

3.7.5.1. Regular SCHC Fragment

Figure 25 shows an example of a regular SCHC fragment for all fragments except the last one. The penultimate tile of a SCHC Packet is of the regular size.

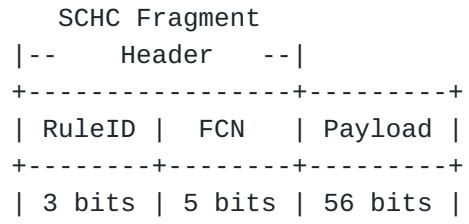


Figure 25: Regular SCHC Fragment format for all fragments except the last one

The SCHC ACK MUST NOT be used, instead the All-1 SCHC Fragment MUST be used to request a SCHC ACK from the receiver. As per [RFC8724], the All-0 message is distinguishable from the SCHC ACK REQ (All-1 message).

3.7.5.2. All-1 SCHC Fragment

Figure 26 shows an example of the All-1 message. The All-1 message MAY contain the last tile of the SCHC Packet.

The All-1 message Fragment Header contains an RCS of 5 bits, and 3 padding bits to complete a 2-byte Fragment Header. The size of the last tile, if present, ranges from 8 to 48 bits.

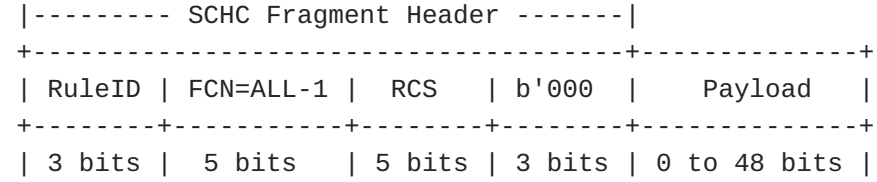


Figure 26: All-1 SCHC message format with last tile

As per [RFC8724] the All-1 must be distinguishable from the a SCHC Sender-Abort message (with same Rule ID and N values). The SCHC Sender-Abort message header size is of 1 byte, with no padding bits.

For the All-1 message to be distinguishable from the Sender-Abort message, the Sender-Abort message MUST be of 1 byte (only header with no padding). This way, the minimum size of the All-1 is 2 bytes, and the Sender-Abort message is 1 bytes.

3.7.5.3. SCHC ACK Format

Figure 27 shows the SCHC ACK format when all fragments have been correctly received (C=1). Padding MUST be added to complete 2 bytes.

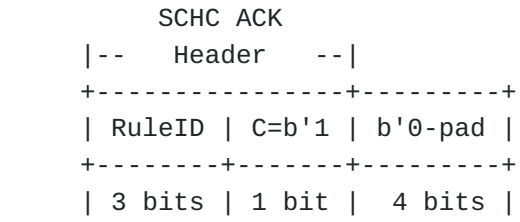


Figure 27: SCHC Success ACK message format

The SCHC ACK message format is shown in Figure 28.

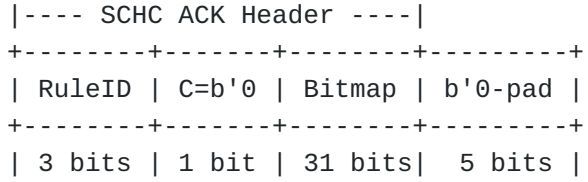


Figure 28: SCHC Compound ACK message format

3.7.5.4. SCHC Sender-Abort Messages

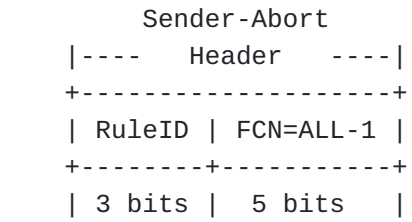


Figure 29: SCHC Sender-Abort message format

3.7.5.5. SCHC Receiver-Abort Message

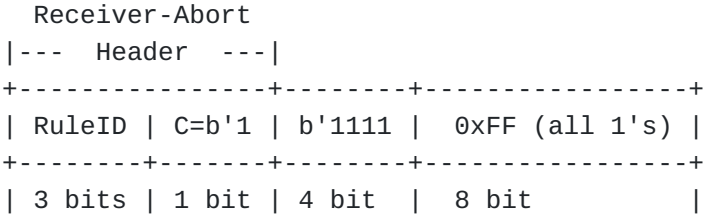


Figure 30: SCHC Receiver-Abort message format

3.8. Padding

The Sigfox payload fields have different characteristics in Uplink and Downlink.

Uplink frames can contain a payload size from 0 to 12 bytes. The Sigfox radio protocol allows sending zero bits, one single bit of information for binary applications (e.g., status), or an integer number of bytes. Therefore, for 2 or more bits of payload it is required to add padding to the next integer number of bytes. The reason for this flexibility is to optimize transmission time and hence save battery consumption at the device.

Downlink frames on the other hand have a fixed length. The payload length MUST be 64 bits (i.e., 8 bytes). Hence, if less information bits are to be transmitted, padding MUST be used with bits equal to 0. The receiver MUST removed the added padding bits before the SCHC reassembly process.

4. Fragmentation Sequence Examples

In this section, some sequence diagrams depicting messages exchanges for different fragmentation modes and use cases are shown. In the examples, 'Seq' indicates the Sigfox Sequence Number of the frame carrying a fragment.

4.1. Uplink No-ACK Examples

The FCN field indicates the size of the data packet. The first fragment is marked with FCN = X-1, where X is the number of fragments the message is split into. All fragments are marked with decreasing FCN values. Last packet fragment is marked with the FCN = All-1 (1111).

Case No losses - All fragments are sent and received successfully.

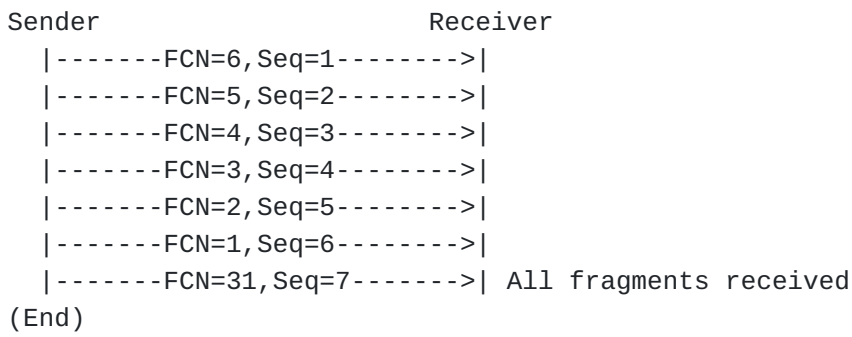


Figure 31: Uplink No-ACK No-Losses

When the first SCHC fragment is received, the Receiver can calculate the total number of SCHC fragments that the SCHC Packet is composed of. For example, if the first fragment is numbered with FCN=6, the receiver can expect six more messages/fragments (i.e., with FCN going from 5 downwards, and the last fragment with a FCN equal to 15).

Case losses on any fragment except the first.

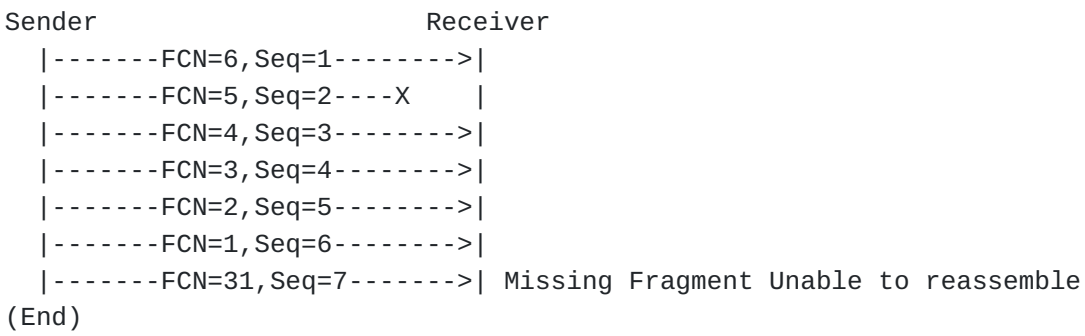


Figure 32: Uplink No-ACK Losses (scenario 1)

4.2. Uplink ACK-on-Error Examples: Single-byte SCHC Header

The single-byte SCHC header ACK-on-Error mode allows sending up to 28 fragments and packet sizes up to 300 bytes. The SCHC fragments may be delivered asynchronously and Downlink ACK can be sent opportunistically.

Case No losses

The Downlink flag must be enabled in the sender Uplink message to allow a Downlink message from the receiver. The Downlink Enable in the figures shows where the sender MUST enable the Downlink, and wait for an ACK.

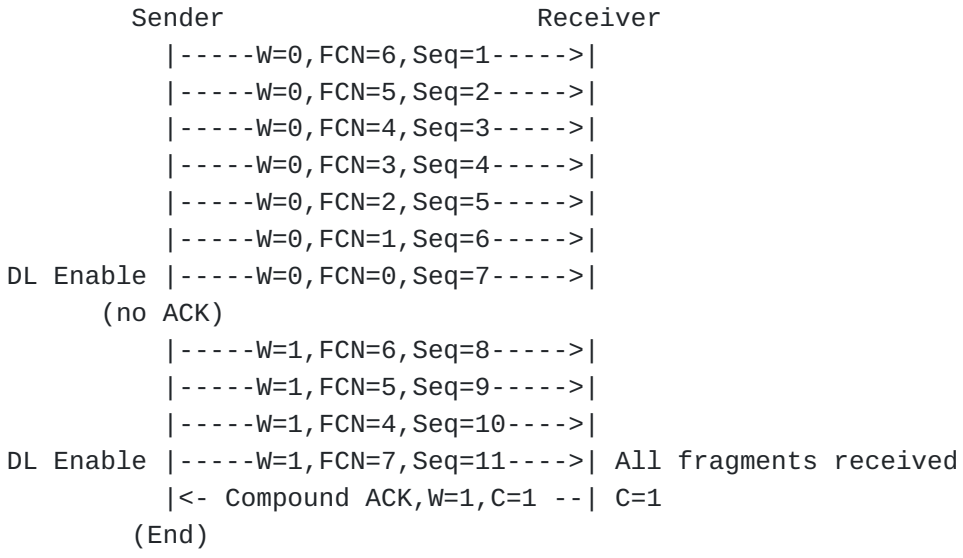


Figure 33: Uplink ACK-on-Error No-Losses

Case Fragment losses in first window

In this case, fragments are lost in the first window (W=0). After the first All-0 message arrives, the Receiver leverages the opportunity and sends a SCHC ACK with the corresponding bitmap and C=0.

After the loss fragments from the first window (W=0) are resent, the sender continues transmitting the fragments of the following window (W=1) without opening a reception opportunity. Finally, the All-1 fragment is sent, the Downlink is enabled, and the SCHC ACK is received with C=1. Note that the SCHC Compound ACK also uses a Sequence Number.

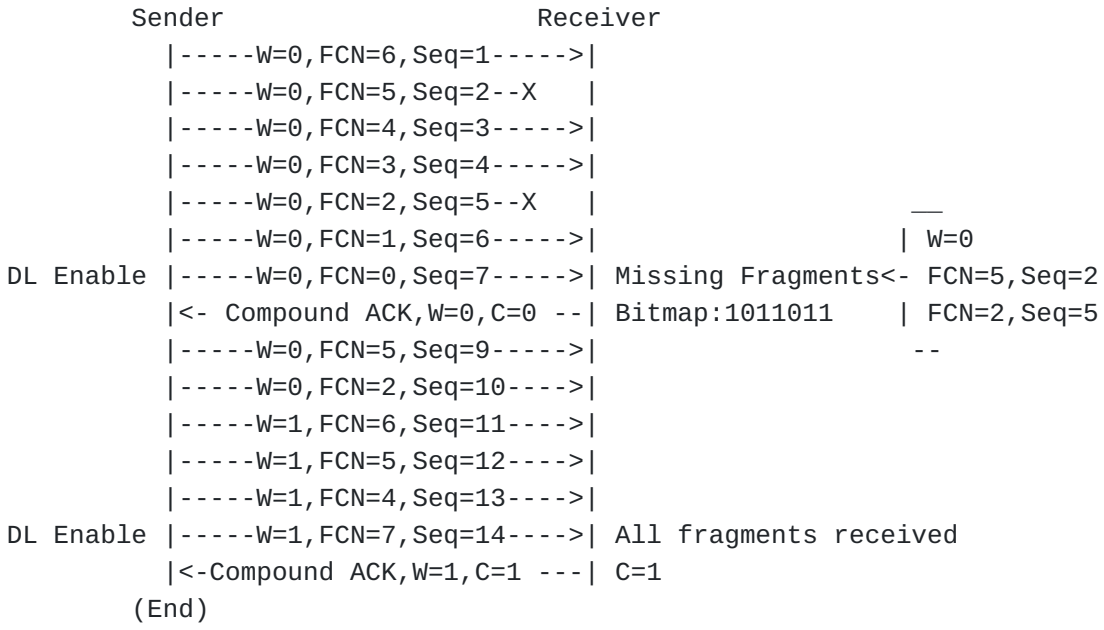


Figure 34: Uplink ACK-on-Error Losses on First Window

Case Fragment All-0 lost in first window (W=0)

In this example, the All-0 of the first window (W=0) is lost. Therefore, the Receiver waits for the next All-0 message of intermediate windows, or All-1 message of last window to generate the corresponding SCHC ACK, notifying the absence of the All-0 of window 0.

The sender resends the missing All-0 messages (with any other missing fragment from window 0) without opening a reception opportunity.

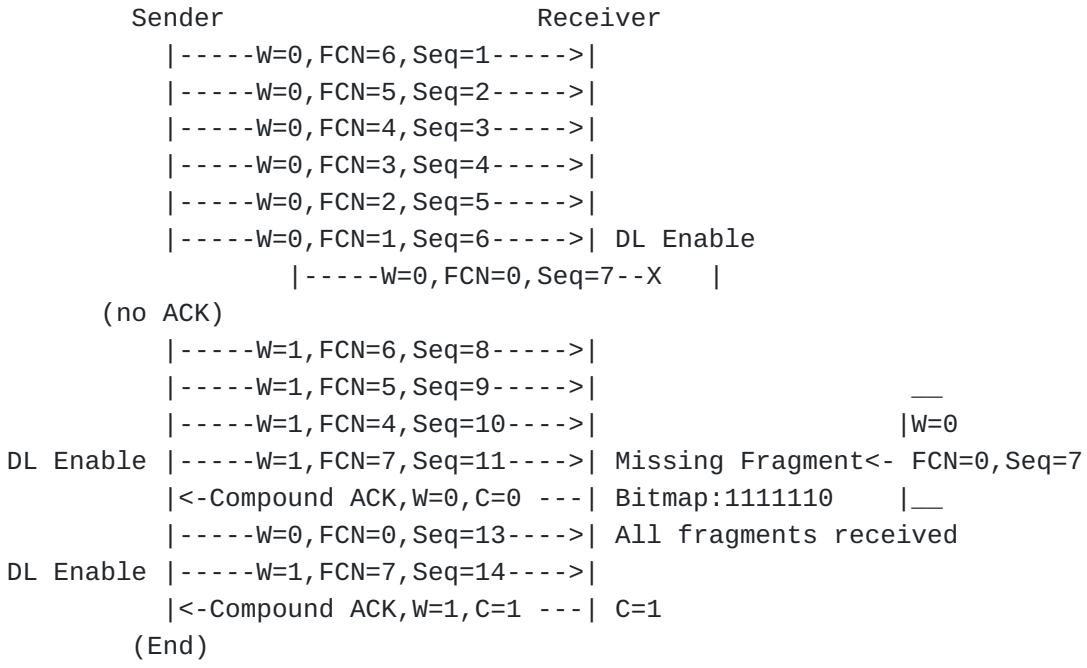


Figure 35: Uplink ACK-on-Error All-0 Lost on First Window

In the following diagram, besides the All-0 there are other fragment losses in the first window (W=0).

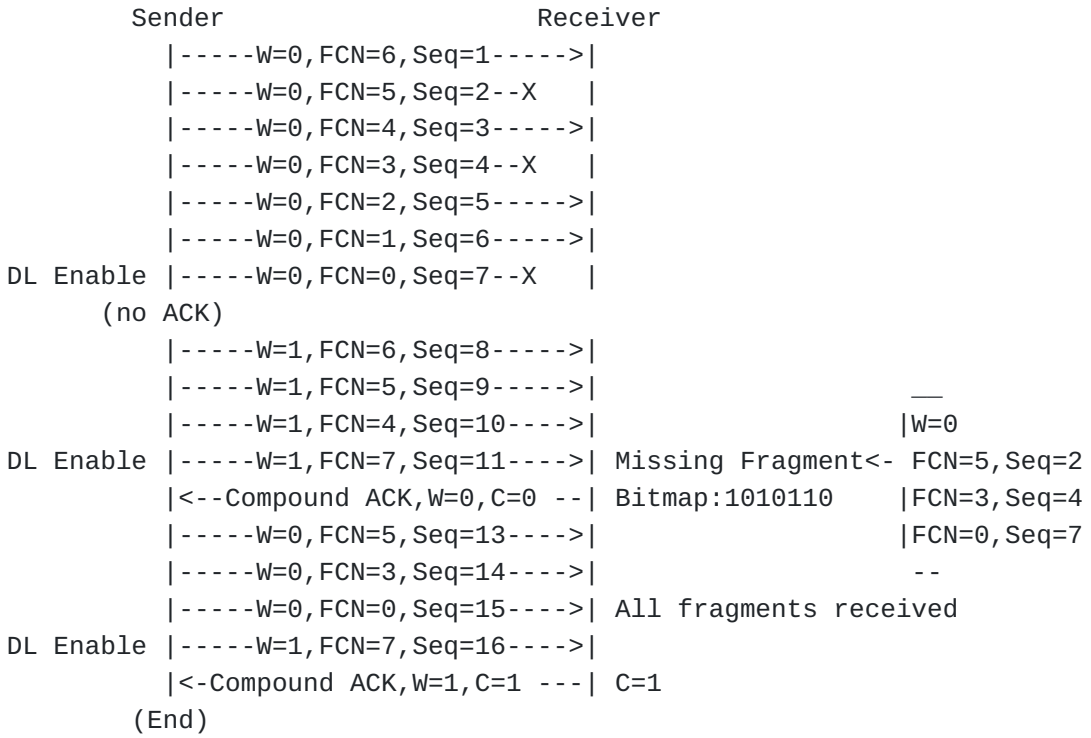


Figure 36: Uplink ACK-on-Error All-0 and other Fragments Lost on First Window

In the next examples, there are fragment losses in both the first (W=0) and second (W=1) windows. The retransmission cycles after the All-1 is sent (i.e., not in intermediate windows) MUST always finish with an All-1, as it serves as an ACK Request message to confirm the correct reception of the retransmitted fragments.

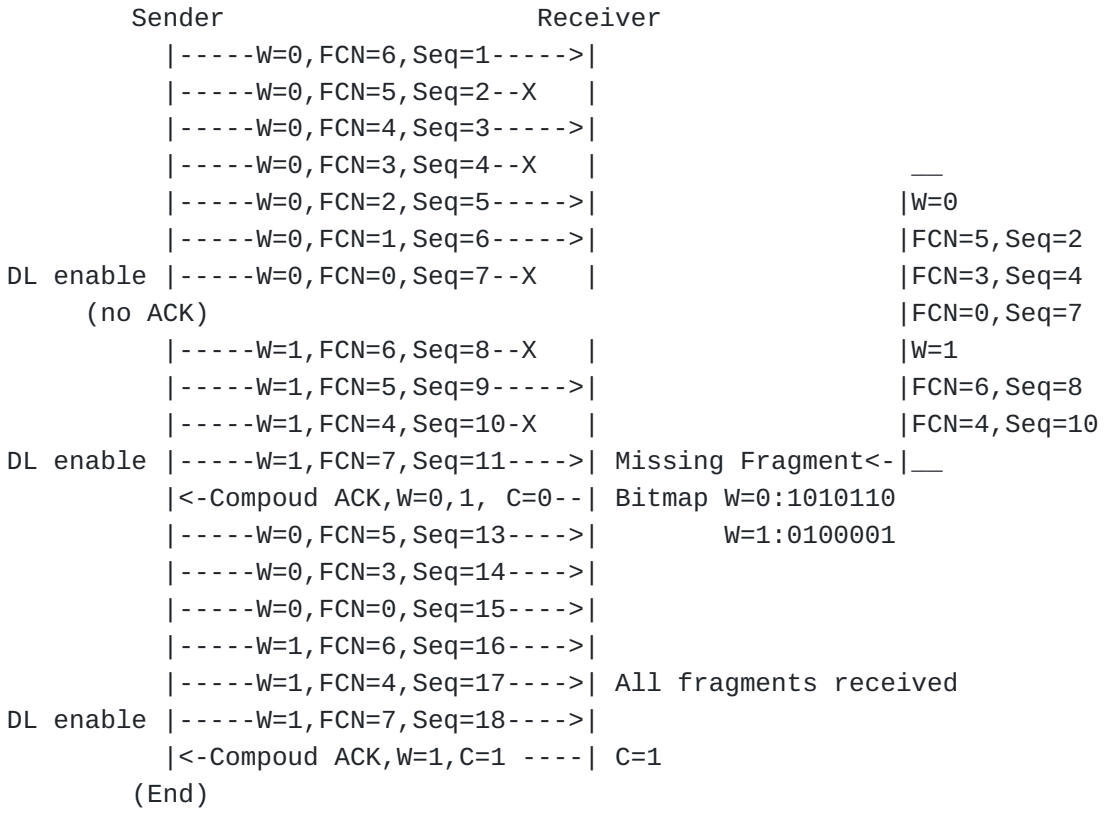


Figure 37: Uplink ACK-on-Error All-0 and other Fragments Lost on First and Second Windows (1)

Similar case as above, but with fewer fragments in the second window (W=1)

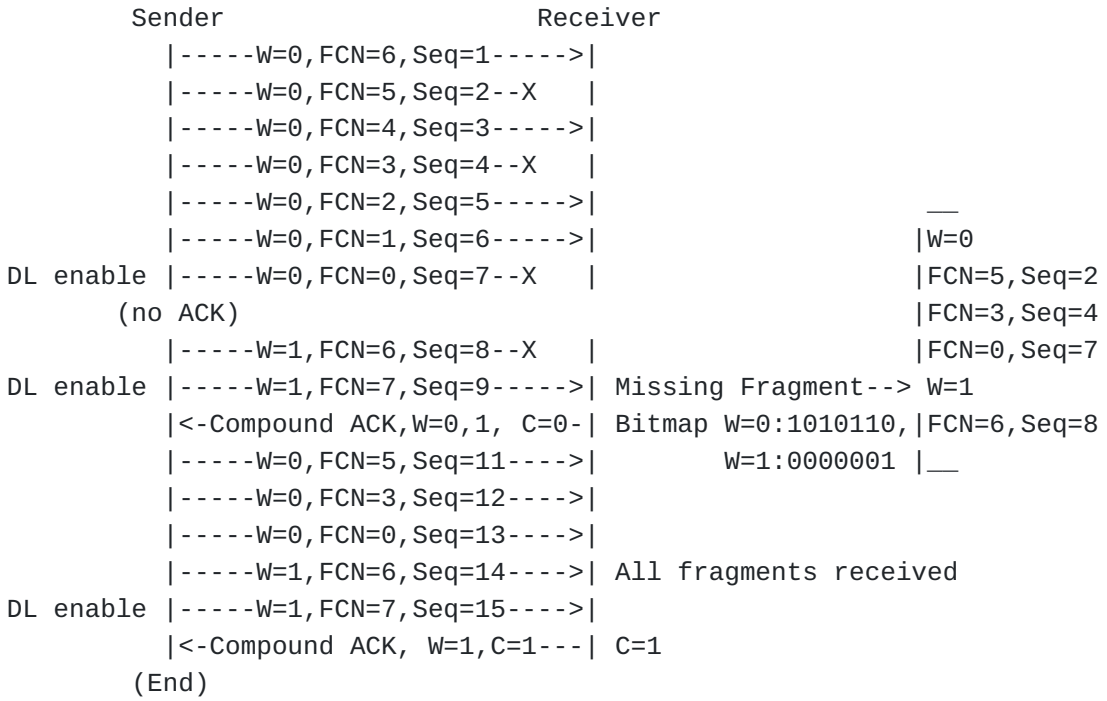


Figure 38: Uplink ACK-on-Error All-0 and other Fragments Lost on First and Second Windows (2)

Case SCHC ACK is lost

SCHC over Sigfox does not implement the SCHC ACK REQ message. Instead, it uses the SCHC All-1 message to request a SCHC ACK, when required.

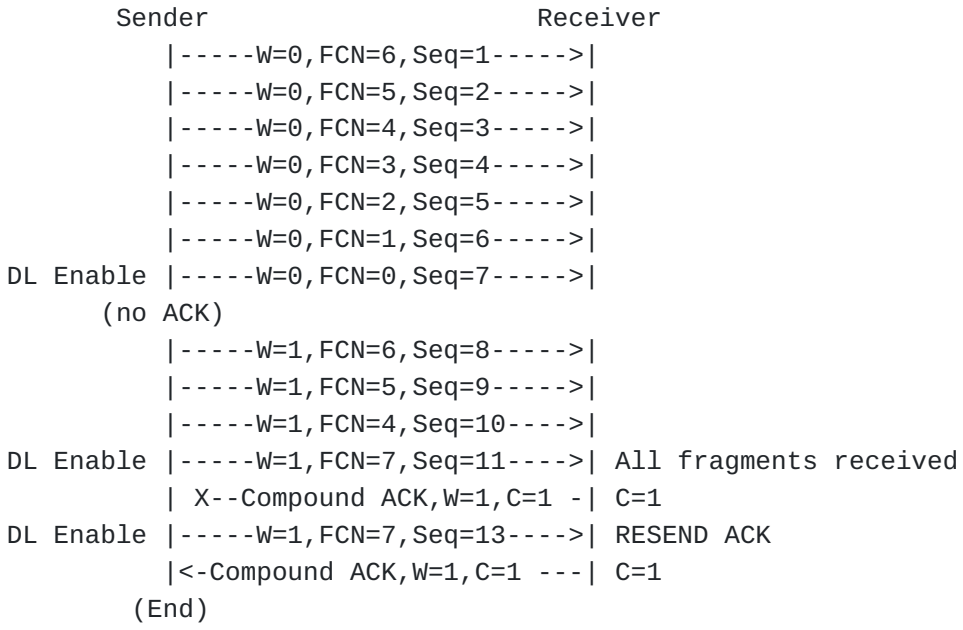


Figure 39: Uplink ACK-on-Error ACK Lost

Case SCHC Compound ACK at the end

In this example, SCHC Fragment losses are found in both window 0 and 1. However, the sender does not send a SCHC ACK after the All-0 of window 0. Instead, it sends a SCHC Compound ACK notifying losses of both windows.

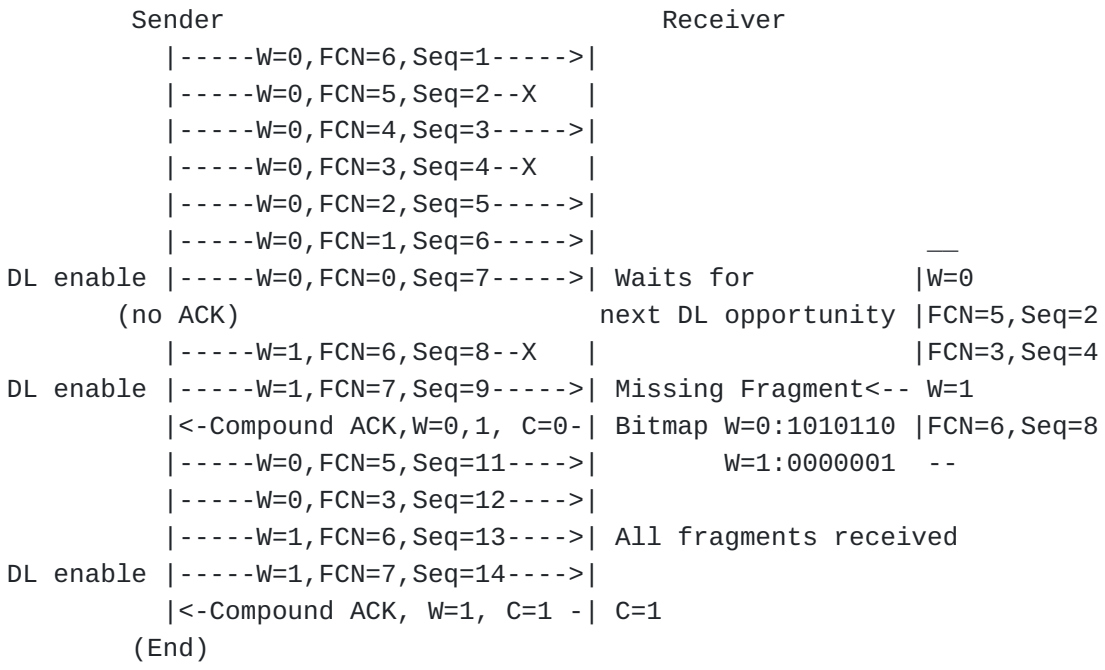


Figure 40: Uplink ACK-on-Error Fragments Lost on First and Second Windows with one Compound ACK

The number of times the same SCHC ACK message will be retransmitted is determined by the MAX_ACK_REQUESTS.

4.3. SCHC Abort Examples

Case SCHC Sender-Abort

The sender may need to send a Sender-Abort to stop the current communication. This may happen, for example, if the All-1 has been sent MAX_ACK_REQUESTS times.

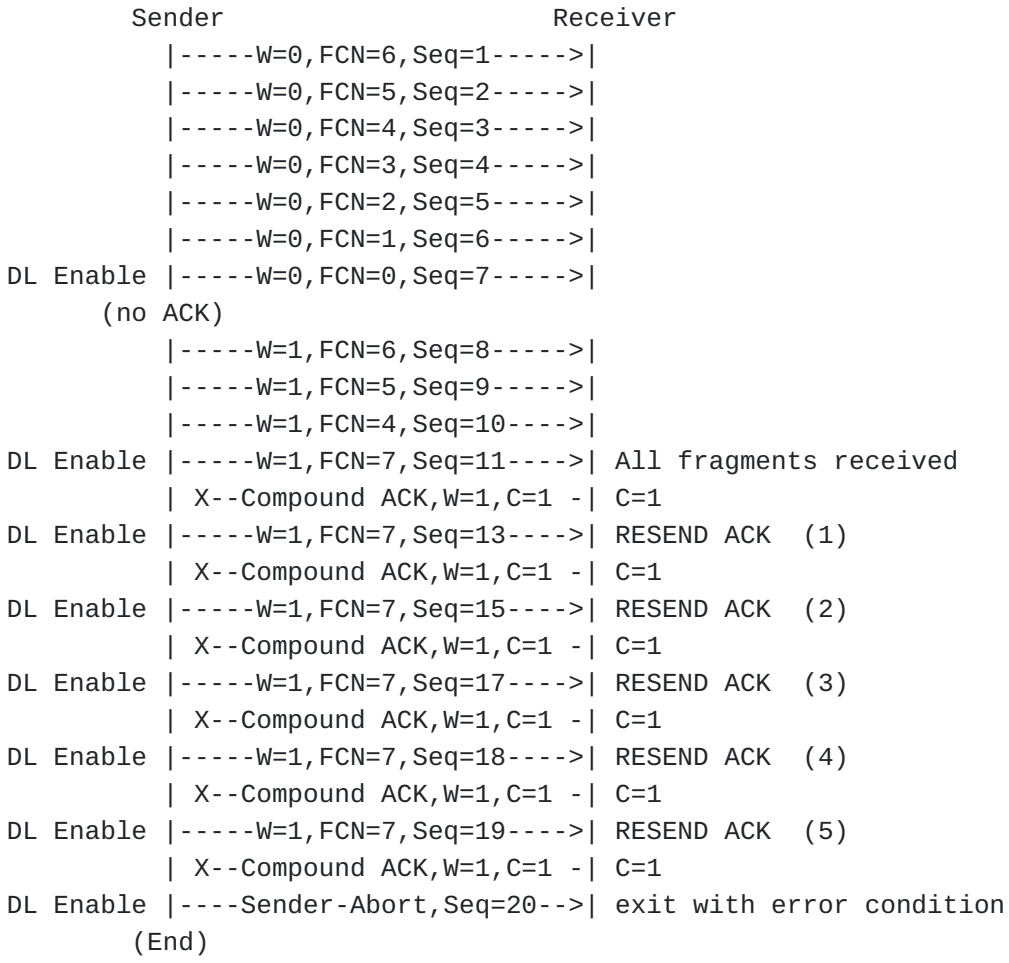


Figure 41: Uplink ACK-on-Error Sender-Abort

Case Receiver-Abort

The receiver may need to send a Receiver-Abort to stop the current communication. This message can only be sent after a Downlink enable.

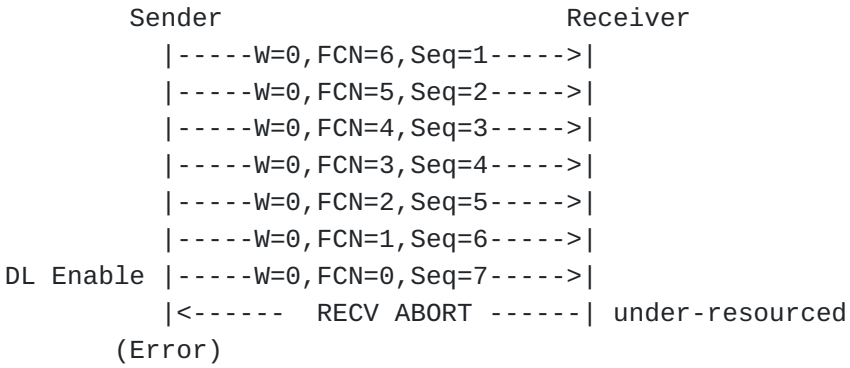


Figure 42: Uplink ACK-on-Error Receiver-Abort

5. Security considerations

The radio protocol authenticates and ensures the integrity of each message. This is achieved by using a unique device ID and an AES-128 based message authentication code, ensuring that the message has been generated and sent by the device or network with the ID claimed in the message.

Application data can be encrypted at the application layer or not, depending on the criticality of the use case. This flexibility allows providing a balance between cost and effort vs. risk. AES-128 in counter mode is used for encryption. Cryptographic keys are independent for each device. These keys are associated with the device ID and separate integrity and encryption keys are pre-provisioned. An encryption key is only provisioned if confidentiality is to be used.

The radio protocol has protections against replay attacks, and the cloud-based core network provides firewalling protection against undesired incoming communications.

6. IANA Considerations

This document has no IANA actions.

7. Acknowledgements

Carles Gomez has been funded in part by the Spanish Government through the Jose Castillejo CAS15/00336 grant, the TEC2016-79988-P grant, and the PID2019-106808RA-I00 grant, and by Secretaria d'Universitats i Recerca del Departament d'Empresa i Coneixement de la Generalitat de Catalunya 2017 through grant SGR 376.

Sergio Aguilar has been funded by the ERDF and the Spanish Government through project TEC2016-79988-P and project PID2019-106808RA-I00, AEI/FEDER, EU.

Sandra Cespedes has been funded in part by the ANID Chile Project FONDECYT Regular 1201893 and Basal Project FB0008.

Diego Wistuba has been funded by the ANID Chile Project FONDECYT Regular 1201893.

The authors would like to thank Ana Minaburo, Clement Mannequin, Rafael Vidal, Julien Boite, Renaud Marty, and Antonis Platis for their useful comments and implementation design considerations.

8. References

8.1. Normative References

[I-D.ietf-lpwan-schc-compound-ack] Zuniga, JC., Gomez, C., Aguilar, S., Toutain, L., Cespedes, S., and D. Wistuba, "SCHC Compound ACK", Work in Progress, Internet-Draft, draft-ietf-lpwan-schc-compound-ack-09, August 2021, <http://www.ietf.org/internet-drafts/draft-ietf-lpwan-schc-compound-ack-09.txt>.

[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>.

[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>.

[RFC8724] Minaburo, A., Toutain, L., Gomez, C., Barthel, D., and JC. Zuniga, "SCHC: Generic Framework for Static Context Header Compression and Fragmentation", RFC 8724, DOI 10.17487/RFC8724, April 2020, <https://www.rfc-editor.org/info/rfc8724>.

8.2. Informative References

[RFC8376] Farrell, S., Ed., "Low-Power Wide Area Network (LPWAN) Overview", RFC 8376, DOI 10.17487/RFC8376, May 2018, <https://www.rfc-editor.org/info/rfc8376>.

[sigfox-callbacks] Sigfox, "Sigfox Callbacks", <https://support.sigfox.com/docs/callbacks-documentation>.

[sigfox-spec]
Sigfox, "Sigfox Radio Specifications", <https://build.sigfox.com/sigfox-device-radio-specifications>.

Authors' Addresses

Juan Carlos Zuniga
Montreal QC
Canada

Email: j.c.zuniga@ieee.org

Carles Gomez
Universitat Politecnica de Catalunya
C/Esteve Terradas, 7
08860 Castelldefels
Spain

Email: carles.gomez@upc.edu

Sergio Aguilar
Universitat Politecnica de Catalunya
C/Esteve Terradas, 7
08860 Castelldefels
Spain

Email: sergio.aguilar.romero@upc.edu

Laurent Toutain
IMT-Atlantique
2 rue de la Chataigneraie
CS 17607
35576 Cesson-Sevigne Cedex
France

Email: Laurent.Toutain@imt-atlantique.fr

Sandra Cespedes
Concordia University
1455 De Maisonneuve Blvd. W.
Montreal QC, H3G 1M8
Canada

Email: sandra.cespedes@concordia.ca

Diego Wistuba
NIC Labs, Universidad de Chile
Av. Almte. Blanco Encalada 1975
Santiago
Chile

Email: wistuba@niclabs.cl

Julien Boite
Unabiz - Sigfox is now a Unabiz technology
Labege
France

Email: julien.boite@unabiz.com
URI: http://www.sigfox.com/