Static Context Header Compression (SCHC) over LoRaWAN
draft-ietf-lpwan-schc-over-lorawan-01

Abstract

The Static Context Header Compression (SCHC) specification describes generic header compression and fragmentation techniques for LPWAN (Low Power Wide Area Networks) technologies. SCHC is a generic mechanism designed for great flexibility, so that it can be adapted for any of the LPWAN technologies.

This document provides the adaptation of SCHC for use in LoRaWAN networks, and provides elements such as efficient parameterization and modes of operation. This is called a profile.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Static Context Header Compression (SCHC) specification [I-D.ietf-lpwan-ipv6-static-context-hc] describes generic header compression and fragmentation techniques that can be used on all LPWAN (Low Power Wide Area Networks) technologies defined in [RFC8376]. Even though those technologies share a great number of common features like star-oriented topologies, network architecture, devices with mostly quite predictable communications, etc; they do have some slight differences in respect of payload sizes, reactiveness, etc.

SCHC gives a generic framework that enables those devices to communicate with other Internet networks. However, for efficient performance, some parameters and modes of operation need to be set appropriately for each of the LPWAN technologies.

This document describes the efficient parameters and modes of operation when SCHC is used over LoRaWAN networks.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

This section defines the terminology and acronyms used in this document. For all other definitions, please look up the SCHC specification [I-D.ietf-lpwan-ipv6-static-context-hc].

o DevEUI: an IEEE EUI-64 identifier used to identify the end-device during the procedure while joining the network (Join Procedure)

o DevAddr: a 32-bit non-unique identifier assigned to a end-device statically or dynamically after a Join Procedure (depending on the activation mode)

o TBD: all significant LoRaWAN-related terms.

3. Static Context Header Compression Overview

This section contains a short overview of Static Context Header Compression (SCHC). For a detailed description, refer to the full specification [I-D.ietf-lpwan-ipv6-static-context-hc].

Static Context Header Compression (SCHC) avoids context synchronization, based on the fact that the nature of data flows is highly predictable in LPWAN networks, some static contexts may be stored on the Device (Dev). The contexts must be stored in both ends, and it can either be learned by a provisioning protocol or by out-of-band means or it can be pre-provisioned, etc. The way the context is learned on both sides is out of the scope of this document.

```
          Dev                                                 App
  +----------------+                                +----+ +----+ +----+
  | App1 App2 App3 |                                |App1| |App2| |App3|
  |                |                                |    | |    | |    |
  |       UDP      |                                |UDP | |UDP | |UDP |
  |      IPv6      |                                |IPv6| |IPv6| |IPv6|
  |                |                                |    | |    | |    |
  |SCHC C/D and F/R|                                |    | |    | |    |
  +--------+-------+                                +----+ +----+ +----+
           |  +--+     +----+    +----+    +----+     .      .      .
           +~ |RG| === |NGW | == |SCHC| == |SCHC|...... Internet ....
              +--+     +----+    |F/R |    |C/D |
                                 +----+    +----+
```

Figure 1: Architecture

Figure 1 represents the architecture for compression/decompression, it is based on [RFC8376] terminology. The Device is sending applications flows using IPv6 or IPv6/UDP protocols. These flow might be fragemented (SCHC F/R), and compressed by an Static Context Header Compression Compressor/Decompressor (SCHC C/D) to reduce headers size. Resulting information is sent on a layer two (L2) frame to a LPWAN Radio Network (RG) which forwards the frame to a Network Gateway (NGW). The NGW sends the data to a SCHC F/R for defragmentation, if required, then C/D for decompression which shares the same rules with the device. The SCHC F/R and C/D can be located on the Network Gateway (NGW) or in another place as long as a tunnel is established between the NGW and the SCHC F/R, then SCHC F/R and SCHC C/D. The SCHC C/D in both sides must share the same set of Rules. After decompression, the packet can be sent on the Internet to one or several LPWAN Application Servers (App).

The SCHC F/R and SCHC C/D process is bidirectional, so the same principles can be applied in the other direction.

In a LoRaWAN network, the RG is called a Gateway, the NGW is Network Server, and the SCHC C/D is an Application Server. It can be provided by the Network Server or any third party software. Figure 1 can be map in LoRaWAN terminology to:

```
        Dev                                                        App
+----------------+                                          +-----+ +-----+ +----
+
| App1 App2 App3 |                                          |App1| |App2| |
App3|
|                |                                          |    | |    | |
|
|       UDP      |                                          |UDP | |UDP | |UDP
|
|      IPv6      |                                          |IPv6| |IPv6| |
IPv6|
|                |                                          |    | |    | |
|
|SCHC C/D and F/R|                                          |    | |    | |
|
+--------+-------+                                          +-----+ +-----+ +----
+
         |  +--------+     +--------+    +----------------+     .      .      .
         +~ |Gateway| === |Network| == |Application     |...... Internet ....
            +--------+     |server |    |server F/R - C/D|
                           +--------+    +----------------+
```

Figure 2: Architecture

4. LoRaWAN Architecture

An overview of LoRaWAN [lora-alliance-spec] protocol and architecture is described in [RFC8376]. Mapping between the LPWAN architecture entities as described in [I-D.ietf-lpwan-ipv6-static-context-hc] and the ones in [lora-alliance-spec] is as follows:

o Devices (Dev) are the end-devices or hosts (e.g. sensors, actuators, etc.). There can be a very high density of devices per radio gateway (LoRaWAN gateway). This entity maps to the LoRaWAN End-Device.

o The Radio Gateway (RGW), which is the end point of the constrained link. This entity maps to the LoRaWAN Gateway.

o The Network Gateway (NGW) is the interconnection node between the Radio Gateway and the Internet. This entity maps to the LoRaWAN Network Server.

o LPWAN-AAA Server, which controls the user authentication and the applications. This entity maps to the LoRaWAN Join Server.

o Application Server (App). The same terminology is used in LoRaWAN. In that case, the application server will be the SCHC gateway, doing C/D and F/R.

```
       ()   ()   ()       |                      +------+
        ()  () () ()     / \       +---------+   | Join |
       () () () () ()   /   \======|    ^    |===|Server|  +-----------+
        () ()  ()      |           | <--|--> |   +------+  |Application|
       () ()  ()  ()  / \==========|    v    |=============|  Server   |
        ()  ()  ()   /   \         +---------+             +-----------+
       End-Devices  Gateways     Network Server
```

Figure 3: LPWAN Architecture

SCHC C/D (Compressor/Decompressor) and SCHC F/R (Fragmentation/ Reassembly) are performed on the LoRaWAN End-Device and the Application Server (called SCHC gateway). While the point-to-point link between the End-Device and the Application Server constitutes single IP hop, the ultimate end-point of the IP communication may be an Internet node beyond the Application Server. In other words, the LoRaWAN Application Server (SCHC gateway) acts as the first hop IP router for the End-Device. The Application Server and Network Server may be co-located, which effectively turns the Network/Application Server into the first hop IP router.

4.1. End-Device classes (A, B, C) and interactions

The LoRaWAN MAC layer supports 3 classes of end-devices named A, B and C. All end-devices implement the classA, some end-devices implement classA+B or class A+C. ClassB and classC are mutually exclusive.

o *ClassA*: The classA is the simplest class of end-devices. The end-device is allowed to transmit at any time, randomly selecting a communication channel. The network may reply with a downlink in one of the 2 receive windows immediately following the uplinks. Therefore, the network cannot initiate a downlink, it has to wait for the next uplink from the end-device to get a downlink opportunity. The classA is the lowest power end-device class.

o *ClassB*: classB end-devices implement all the functionalities of classA end-devices, but also schedule periodic listen windows. Therefore, as opposed the classA end-devices, classB end-devices can receive downlink that are initiated by the network and not following an uplink. There is a trade-off between the periodicity of those scheduled classB listen windows and the power consumption of the end-device. The lower the downlink latency, the higher the power consumption.

o *ClassC*: classC end-devices implement all the functionalities of classA end-devices, but keep their receiver open whenever they are

not transmitting. ClassC end-devices can receive downlinks at any time at the expense of a higher power consumption. Battery powered end-devices can only operate in classC for a limited amount of time (for example for a firmware upgrade over-the-air). Most of the classC end-devices are main powered (for example Smart Plugs).

4.2. End-Device addressing

LoRaWAN end-devices use a 32 bits network address (devAddr) to communicate with the network over-the-air. However, that address might be reused several time on the same network at the same time for different end-devices. End-devices using the same devAddr are distinguish by the Network Server based on the cryptographic signature appended to every single LoRaWAN MAC frame, as all end-devices use different security keys. To communicate with the SCHC gateway the Network Server MUST identify the end-devices by a unique 64bits device ID called the devEUI. Unlike devAddr, devEUI is guaranteed to be unique for every single end-device across all networks. The devEUI is assigned to the end-device during the manufacturing process by the end-device's manufacturer. It is built like an Ethernet MAC address by concatenating the manufacturer's IEEE OUI field with a vendor unique number. ex: 24bits OUI is concatenated with a 40 bits serial number. The Network Server translates the devAddr into a devEUI in the uplink direction and reciprocally on the downlink direction.

```
+--------+         +----------+        +---------+           +----------+
| End-   | <=====> | Network  | <====> | SCHC    | <========> | Internet |
| Device | devAddr | Server   | devEUI | Gateway |  IPv6/UDP  |          |
+--------+         +----------+        +---------+           +----------+
```

Figure 4: LoRaWAN addresses

4.3. General Message Types

o *Confirmed messages*: The sender asks the receiver to acknowledge the message.

o *Unconfirmed messages*: The sender does not ask the receiver to acknowledge the message.

As SCHC defines its own acknowledgment mechanisms, SCHC does not require to use confirmed messages.

4.4. LoRaWAN MAC Frames

- *JoinRequest*: This message is used by a end-device to join a network. It contains the end-device's unique identifier devEUI and a random nonce that will be used for session key derivation.

- *JoinAccept*: To on-board a end-device, the Network Server responds to the JoinRequest end-device's message with a JoinAccept message. That message is encrypted with the end-device's AppKey and contains (amongst other fields) the major network's settings and a network random nonce used to derive the session keys.

- *Data*

5. SCHC-over-LoRaWAN

5.1. LoRaWAN FPort

The LoRaWAN MAC layers features a frame port field in all frames. This field (FPort) is 8-bit long and the values from 1 to 223 can be used. It allows LoRaWAN network and application to identify data.

A fragmentation session with application payload transferred from device to server, is called uplink fragmentation session. It uses FPortUpShort or FPortUpDefault for data uplink and its associated SCHC control downlinks. The other way, a fragmentation session with application payload transferred from server to device, is called downlink fragmentation session. It uses FPortDown for data downlink and its associated SCHC control uplinks.

FPorts can use arbitrary values inside the allowed FPort range and must be shared by the end-device, the Network Server and SCHC gateway. The uplink and downlink SCHC ports must be different. In order to improve interoperability, it is recommended to use:

- FPortUpShort = 20

- FPortUpDefault = 21

- FPortDown = 22

Those are recommended values and are application defined. Also application can have multiple fragmentation session between a device and one or several SCHC gateways. A set of three FPort values is required for each gateway instance the device is required to communicate with.

The only uplink messages using the FPortDown port are the fragmentation SCHC control messages of a downlink fragmentation session (ex ACKs). Similarly, the only downlink messages using the FPortUpShort or FPortUpDefault ports are the fragmentation SCHC control messages of an uplink fragmentation session.

5.2. Rule ID management

SCHC-over-LoRaWAN SHOULD support encoding RuleID on 6 bits (64 possible rules).

The RuleID 0 is reserved for fragmentation. The RuleID 63 is used to tag packets for which SCHC compression was not possible (no matching Rule was found).

The remaining RuleIDs are available for compression. RuleIDs are shared between uplink and downlink sessions. A RuleID different from 0 means that the fragmentation is not used, thus the packet should be send to C/D layer.

5.3. IID computation

As LoRaWAN network uses unique EUI-64 per end-device, the Interface IDentifier is the LoRaWAN DevEUI. It is compliant with [RFC4291] and IID starting with binary 000 must enforce the 64-bits rule. TODO: Derive IID from DevEUI with privacy constraints ? Ask working group ?

5.4. Fragmentation

The L2 word size used by LoRaWAN is 1 byte (8 bits). The SCHC fragmentation over LoRaWAN uses the ACK-on-Error for uplink fragmentation and Ack-Always for downlink fragmentation. A LoRaWAN end-device cannot support simultaneous interleaved fragmentation sessions in the same direction (uplink or downlink). This means that only a single fragmented IPv6 datagram may be transmitted and/or received by the end-device at a given moment.

The fragmentation parameters are different for uplink and downlink fragmentation sessions and are successively described in the next sections.

5.5. DTag

A LoRaWAN device cannot interleave several fragmented SCHC datagrams. This one bit field is used to distinguish two consecutive fragmentation sessions.

Note: While it is used to recover faster from transmission errors, it SHALL not be considered as the only way to distinguish two fragmentation sessions.

5.5.1. Uplink fragmentation: From device to SCHC gateway

In that case the device is the fragmentation transmitter, and the SCHC gateway the fragmentation receiver. Two fragmentation rules are defined regarding the *FPort*:

o *FPortUpShort*: SCHC header is only one byte. Used when fragmentation is required and payload size is less than 381 bytes.

o *FPortUpDefault*: SCHC header is two bytes. Used for all other cases: no fragmentation required or payload size is between 382 and 1524 byte.

Both rules share common parameters:

o *SCHC fragmentation reliability mode*: "ACK-on-Error"

o *DTag*: size is 1 bit.

o *FCN*: The FCN field is encoded on N = 7 bits, so WINDOW_SIZE = 127 tiles are allowed in a window (FCN=All-1 is reserved for SCHC).

o *MIC calculation algorithm*: CRC32 using 0xEDB88320 (i.e. the reverse representation of the polynomial used e.g. in the Ethernet standard [RFC3385]) as suggested in [I-D.ietf-lpwan-ipv6-static-context-hc].

o *MAX_ACK_REQUESTS*: 8

o *Tile*: size is 3 bytes (24 bits)

o *Retransmission and inactivity timers*: LoRaWAN end-devices do not implement a "retransmission timer". At the end of a window or a fragmentation session, corresponding ACK(s) is (are) transmitted by the network gateway (LoRaWAN application server) in the RX1 or RX2 receive slot of end-device. If this ACK is not received the end-device sends an all-0 (or an all-1) fragment with no payload to request an SCHC ACK retransmission. The periodicity between retransmission of the all-0/all-1 fragments is device/application specific and may be different for each device (not specified). The SCHC gateway implements an "inactivity timer". The default recommended duration of this timer is 12 hours. This value is

mainly driven by application requirements and may be changed by the application.

The following fields are different:

o RuleID size

o Window index size W

5.5.1.1. FPortUpShort - 1 byte header

In that case RuleID size is 0, the rule is the FPort=FPortUpShort and only fragmented payload can be transported.

o *RuleID*: size is 0 bit in SCHC header, not used.

o *Window index*: encoded on W = 0 bit, not used

With this set of parameters, the SCHC fragment header overhead is 1 byte (8 bits). MTU is: _127 tiles * 3 bytes per tile = 381 bytes_

Regular fragments

```
| DTag  | FCN    | Payload |
+ ----- + ------ + ------- +
| 1 bit | 7 bits |         |
```

Figure 5: All fragment except the last one. Header size is 8 bits (1 byte).

SCHC ACK

```
| RuleID | DTag  | W     | C     | Encoded bitmap (if C = 0) | Padding (0s) |
+ ------ + ----- + ----- + ----- + ------------------------- + ------------ +
| 6 bits | 1 bit | 2 bit | 1 bit | 0 to 127 bits             | 7 or 0 bits  |
```

Figure 6: SCHC ACK format, failed mic check.

5.5.1.2. FPortUpDefault - 2 bytes header

o *RuleID*: size is 6 bits (64 possible rules, 62 available for compression)

o *Window index*: encoded on W = 2 bits. So 4 windows are available.

With this set of parameters, the SCHC fragment header overhead is 2 bytes (16 bits). MTU is: _4 windows * 127 tiles * 3 bytes per tile = 1524 bytes_

Note: Even if it is less efficient, this rule can also be used for fragmented payload size less than 382 bytes.

Regular fragments

```
| RuleID | DTag  | W      | FCN    | Payload |
+ ------ + ----- + ------ + ------ + ------- +
| 6 bits | 1 bit | 2 bits | 7 bits |         |
```

Figure 7: All fragment except the last one. Header size is 16 bits (2 bytes).

Last fragment (All-1)

```
| RuleID | DTag  | W      | FCN=All-1 | MIC     | Payload           |
+ ------ + ----- + ------ + --------- + ------- + ----------------- +
| 6 bits | 1 bit | 2 bits | 7 bits    | 32 bits | Last tile, if any |
```

Figure 8: All-1 fragment detailed format for the last fragment.

SCHC ACK

```
| RuleID | DTag  | W     | C     | Encoded bitmap (if C = 0) |
+ ------ + ----- + ----- + ----- + ------------------------- +
| 6 bits | 1 bit | 2 bit | 1 bit | 0 to 127 bits             |
```

Figure 9: SCHC formats, failed MIC check.

Receiver-Abort

```
| RuleID | DTag  | W = b'11 | C = 1 | b'111111 | 0xFF (all 1's) |
+ ------ + ----- + -------- + ------+--------- + ---------------+
| 6 bits | 1 bit | 2 bits   | 1 bit | 6 bits   | 8 bits         |
```

Figure 10: Receiver-Abort format.

SCHC acknowledge request

```
   | RuleID | DTag  | W      | FCN = b'0000000 |
   + ------ + ----- + ------ + --------------- +
   | 6 bits | 1 bit | 2 bits | 7 bits          |
```

Figure 11: SCHC ACK REQ format.

5.5.2. Downlink fragmentation: From SCHC gateway to device

In that case the device is the fragmentation receiver, and the SCHC gateway the fragmentation transmitter. The following fields are common to all devices.

o *SCHC fragmentation reliability mode*: ACK-Always.

o *RuleID*: size is 6 bits (64 possible rules, 62 for compression).

o *Window index*: encoded on W=1 bit, as per [I-D.ietf-lpwan-ipv6-static-context-hc].

o *DTag*: Not used, so its size is 0 bit.

o *FCN*: The FCN field is encoded on N=1 bits, so WINDOW_SIZE = 1 tile (FCN=All-1 is reserved for SCHC).

o *MIC calculation algorithm*: CRC32 using 0xEDB88320 (i.e. the reverse representation of the polynomial used e.g. in the Ethernet standard [RFC3385]), as per [I-D.ietf-lpwan-ipv6-static-context-hc].

o *MAX_ACK_REQUESTS*: 8

As only 1 tile is used, its size can change for each downlink, and will be maximum available MTU minus header (1 byte)

Note: The Fpending bit included in LoRaWAN protocol SHOULD not be used for SCHC-over-LoRaWAN protocol. It might be set by the Network Server for other purposes in but not SCHC needs.

Regular fragments

```
   | RuleID | W     | FCN = b'0 | Payload |
   + ------ + ----- + --------- + ------- +
   | 6 bits | 1 bit | 1 bits    | X bytes |
```

Figure 12: All fragments but the last one. Header size 1 byte (8 bits).

Last fragment (All-1)

```
   | RuleID | W     | FCN = b'1 | MIC     | Payload          |
   + ------ + ----- + --------- + ------- + ---------------- +
   | 6 bits | 1 bit | 1 bit     | 32 bits | Last tile, if any |
```

Figure 13: All-1 SCHC ACK detailed format for the last fragment.

SCHC acknowledge

```
   | RuleID | W     | C = b'1 |
   + ------ + ----- + ------- +
   | 6 bits | 1 bit | 1 bit   |
```

Figure 14: SCHC ACK format, MIC is correct.

Receiver-Abort

```
   | RuleID | W    | C = b'0 | b'11111111 |
   + ------ + ----- + ------- + ---------- +
   | 6 bits | 1 bit | 1 bits  | 8 bits     |
```

Figure 15: Receiver-Abort packet (following an all-1 packet with incorrect MIC).

Class A and classB&C end-devices do not manage retransmissions and timers in the same way.

5.5.2.1. ClassA end-devices

Class A end-devices can only receive in an RX slot following the transmission of an uplink. Therefore there cannot be a concept of "retransmission timer" for an SCHC gateway. The SCHC gateway cannot initiate communication to a classA end-device.

The device replies with an ACK message to every single fragment received from the SCHC gateway (because the window size is 1).

Following the reception of a FCN=0 fragment (fragment that is not the last fragment of the packet or ACK-request, but the end of a window), the device MUST transmit the SCHC ACK fragment until it receives the fragment of the next window. The device shall transmit up to MAX_ACK_REQUESTS ACK messages before aborting. The device should transmit those ACK as soon as possible while taking into consideration potential local radio regulation on duty-cycle, to progress the fragmentation session as quickly as possible. The ACK bitmap is 1 bit long and is always 1.

Following the reception of a FCN=All-1 fragment (the last fragment of a datagram) and if the MIC is correct, the device shall transmit the ACK with the "MIC is correct" indicator bit set (C=1). This message might be lost therefore the SCHC gateway may request a retransmission of this ACK in the next downlink. The device SHALL keep this ACK message in memory until it receives a downlink, on SCHC FPortDown from the SCHC gateway different from an ACK-request: it indicates that the SCHC gateway has received the ACK message.

Following the reception of a FCN=All-1 fragment (the last fragment of a datagram), if all fragments have been received and the MIC is NOT correct, the device shall transmit a Receiver-Abort fragment. The device SHALL keep this Abort message in memory until it receives a downlink, on SCHC FPortDown, from the SCHC gateway different from an ACK-request indicating that the SCHC gateway has received the Abort message. The fragmentation receiver (device) does not implement retransmission timer and inactivity timer.

The fragmentation sender (the SCHC gateway) implements an inactivity timer with default duration of 12 hours. Once a fragmentation session is started, if the SCHC gateway has not received any ACK or Receiver-Abort message 12 hours after the last message from the device was received, the SCHC gateway may flush the fragmentation context. For devices with very low transmission rates (example 1 packet a day in normal operation) , that duration may be extended, but this is application specific.

5.5.2.2. Class B or C end-devices

Class B&C end-devices can receive in scheduled RX slots or in RX slots following the transmission of an uplink. The device replies with an ACK message to every single fragment received from the SCHC gateway (because the window size is 1). Following the reception of a FCN=0 fragment (fragment that is not the last fragment of the packet or ACK-request), the device MUST always transmit the corresponding SCHC ACK message even if that fragment has already been received. The ACK bitmap is 1 bit long and is always 1. If the SCHC gateway receives this ACK, it proceeds to send the next window fragment. If

the retransmission timer elapses and the SCHC gateway has not received the ACK of the current window it retransmits the last fragment. The SCHC gateway tries retransmitting up to MAX_ACK_REQUESTS times before aborting.

Following the reception of a FCN=All-1 fragment (the last fragment of a datagram) and if the MIC is correct, the device shall transmit the ACK with the "MIC is correct" indicator bit set. If the SCHC gateway receives this ACK, the current fragmentation session has succeeded and its context can be cleared.

If the retransmission timer elapses and the SCHC gateway has not received the SCHC ACK it retransmits the last fragment with the payload (not an ACK-request without payload). The SCHC gateway tries retransmitting up to MAX_ACK_REQUESTS times before aborting.

The device SHALL keep the SCHC ACK message in memory until it receives a downlink from the SCHC gateway different from the last (FCN>0 and different DTag) fragment indicating that the SCHC gateway has received the ACK message.

Following the reception of a FCN=All-1 fragment (the last fragment of a datagram), if all fragments have been received and if the MIC is NOT correct, the device shall transmit a Receiver-Abort fragment. The retransmission timer is used by the SCHC gateway (the sender), the optimal value is very much application specific but here are some recommended default values. For classB end-devices, this timer trigger is a function of the periodicity of the classB ping slots. The recommended value is equal to 3 times the classB ping slot periodicity. For classC end-devices which are nearly constantly receiving, the recommended value is 30 seconds. This means that the end-device shall try to transmit the ACK within 30 seconds of the reception of each fragment. The inactivity timer is implemented by the end-device to flush the context in-case it receives nothing from the SCHC gateway over an extended period of time. The recommended value is 12 hours for both classB&C end-devices.

6. Security considerations

This document is only providing parameters that are expected to be better suited for LoRaWAN networks for [I-D.ietf-lpwan-ipv6-static-context-hc]. As such, this parameters does not contribute to any new security issues in addition of those identified in [I-D.ietf-lpwan-ipv6-static-context-hc].

Acknowledgements

Thanks to all those listed in the Contributors section for the excellent text, insightful discussions, reviews and suggestions.

Contributors

Contributors ordered by family name.

o ins: V. Audebert name: Vincent AUDEBERT org: EDF R&D street: 7 bd Gaspard Monge city: 91120 PALAISEAU country: FRANCE email: vincent.audebert@edf.fr

o ins: J. Catalano name: Julien Catalano org: Kerlink street: 1 rue Jacqueline Auriol city: 35235 Thorigne-Fouillard country: France email: j.catalano@kerlink.fr

o ins: M. Coracin name: Michael Coracin org: Semtech street: 14 Chemin des Clos city: Meylan country: France email: mcoracin@semtech.com

o ins: M. Le Gourrierec name: Marc Le Gourrierec org: SagemCom street: 250 Route de l'Empereur city: 92500 Rueil Malmaison country: FRANCE email: marc.legourrierec@sagemcom.com

o ins: N. Sornin name: Nicolas Sornin org: Semtech street: 14 Chemin des Clos city: Meylan country: France email: nsornin@semtech.com

o ins: A. Yegin name: Alper Yegin org: Actility street: . city: Paris, Paris country: France email: alper.yegin@actility.com

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>.

[RFC3385] Sheinwald, D., Satran, J., Thaler, P., and V. Cavanna, "Internet Protocol Small Computer System Interface (iSCSI) Cyclic Redundancy Check (CRC)/Checksum Considerations", RFC 3385, DOI 10.17487/RFC3385, September 2002, <https://www.rfc-editor.org/info/rfc3385>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <https://www.rfc-editor.org/info/rfc4291>.

[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>.

[RFC5795] Sandlund, K., Pelletier, G., and L-E. Jonsson, "The RObust Header Compression (ROHC) Framework", RFC 5795, DOI 10.17487/RFC5795, March 2010, <https://www.rfc-editor.org/info/rfc5795>.

[RFC7136] Carpenter, B. and S. Jiang, "Significance of IPv6 Interface Identifiers", RFC 7136, DOI 10.17487/RFC7136, February 2014, <https://www.rfc-editor.org/info/rfc7136>.

[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>.

9.2. Informative References

[I-D.ietf-lpwan-ipv6-static-context-hc]
Minaburo, A., Toutain, L., Gomez, C., Barthel, D., and J. Zuniga, "LPWAN Static Context Header Compression (SCHC) and fragmentation for IPv6 and UDP", draft-ietf-lpwan-ipv6-static-context-hc-18 (work in progress), December 2018.

[lora-alliance-spec]
Alliance, L., "LoRaWAN Specification Version V1.0.3", <https://lora-alliance.org/sites/default/files/2018-07/lorawan1.0.3.pdf>.

Appendix A. Examples

A.1. Uplink - Compression example - No fragmentation

Figure 16 is representing an applicative payload going through SCHC, no fragmentation required

An applicative payload of 78 bytes is passed to SCHC compression layer using rule 1, allowing to compress it to 40 bytes: 2 bytes residue + 38 bytes payload.

RuleID	Compression residue	Payload
1	18 bits	38 bytes

The current LoRaWAN MTU is 51 bytes, although 2 bytes FOpts are used by LoRaWAN protocol: 49 bytes are available for SCHC payload; no need for fragmentation. The payload will be transmitted through FPortUpDefault

LoRaWAN Header	RuleID	Compression residue	Payload
XXXX	1	18 bits	38 bytes

Figure 16: Uplink example: compression without fragmentation

A.2. Uplink - Compression and fragmentation example

Figure 17 is representing an applicative payload going through SCHC, with fragmentation.

An applicative payload of 478 bytes is passed to SCHC compression layer using rule 1, allowing to compress it to 440 bytes: 18 bits residue + 138 bytes payload.

RuleID	Compression residue	Payload
1	18 bits	138 bytes

Given the size of the payload, FPortUpDefault will be used.
The current LoRaWAN MTU is 11 bytes, although 2 bytes FOpts are used by LoRaWAN protocol: 9 bytes are available for SCHC payload.
SCHC header is 2 bytes so 2 tiles are send in first fragment.

LoRaWAN Header	FOpts	RuleID	DTag	W	FCN	2 tiles
XXXX	2 bytes	0	0	0	126	6 bytes

Content of the two tiles is:

RuleID	Compression residue	Payload

```
+ ------ + ------------------- + --------- +
|   1    |       18 bits       |  3 bytes  |
```

Next transmission MTU is 242 bytes, no FOpts. 80 tiles are transmitted:

```
| LoRaWAN Header | RuleID | DTag  |   W    |  FCN   | 80 tiles  |
+ -------------- + ------ + ----- + ------ + ------ + --------- +
|       XXXX     |   0    |   0   |   0    |  124   | 240 bytes |
```

Next transmission MTU is 242 bytes, no FOpts. All 65 remaining tiles are transmitted, last tile is only 2 bytes.

```
| LoRaWAN Header | RuleID | DTag  |   W    |  FCN   |  MIC  | 65 tiles  |
+ -------------- + ------ + ----- + ------ + ------ + ----- + --------- +
|       XXXX     |   0    |   0   |   0    |  127   | CRC32 | 194 bytes |
```

All packets have been received by the SCHC gateway, computed MIC is correct so the following ACK is send to the device:

```
| LoRaWAN Header | RuleID | DTag  |   W    |  C  |
+ -------------- + ------ + ----- + ------ + --- +
|       XXXX     |   0    |   0   |   0    |  1  |
```

Figure 17: Uplink example: compression and fragmentation

A.3. Downlink

TODO

Appendix B. Note

Authors' Addresses

Olivier Gimenez (editor)
Semtech
14 Chemin des Clos
Meylan
France

Email: ogimenez@semtech.com

Ivaylo Petrov (editor)
Acklio
2bis rue de la Chataigneraie
35510 Cesson-Sevigne Cedex
France

Email: ivaylo@ackl.io