lpwan Working Group
Internet-Draft
Intended status: Informational
Expires: January 3, 2019

N. Sornin, Ed.
M. Coracin
Semtech
I. Petrov
Acklio
A. Yegin
Actility
J. Catalano
Kerlink
V. Audebert
EDF R&D

July 02, 2018

Static Context Header Compression (SCHC) over LoRaWAN
draft-petrov-lpwan-ipv6-schc-over-lorawan-02

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.

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 January 3, 2019.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

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

[I-D.ietf-lpwan-overview]. Even though those technologies share a great number of common features like start-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

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 device during the procedure while joining the network (Join Procedure)

o DevAddr: a 32-bit non-unique identifier assigned to a 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, which is the most bandwidth-consuming operation in other header compression mechanisms such as RoHC [RFC5795]. 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      |
|     IPv6     |                                  |    IPv6      |
|              |                                  |              |
|   SCHC C/D   |                                  |              |
|   (context)  |                                  |              |
+-------+------+                                  +-------+------+
         |   +--+     +----+     +---------+              .
         +~~ |RG| === |NGW | === |SCHC C/D |... Internet ..
             +--+     +----+     |(context)|
                                 +---------+
```

Figure 1: Architecture

Figure 1 represents the architecture for compression/decompression, it is based on [I-D.ietf-lpwan-overview] terminology. The Device is sending applications flows using IPv6 or IPv6/UDP protocols. These flows are 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 C/D for decompression which shares the same rules with the Dev. The SCHC 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 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 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 can be embedded in different places, for example in the Network Server and/or the Application Server.

Next steps for this section: detailed overview of the LoRaWAN architecture and its mapping to the SCHC architecture.

4. LoRaWAN Architecture

An overview of LoRaWAN [lora-alliance-spec] protocol and architecture is described in [I-D.ietf-lpwan-overview]. 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. 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.

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

Figure 1: LPWAN/LoRaWAN Architecture

SCHC C/D (Compressor/Decompressor) and SCHC Fragmentation are performed on the LoRaWAN End-device and the Application Server. 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 acts as the first hop IP router for the End-device. Note that 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. Device classes (A, B, C) and interactions

TBD

4.2. Device addressing

TBD

4.3. General Message Types

TBD

4.4. LoRaWAN MAC Frames

TBD

5. SCHC over LoRaWAN

5.1. Rule ID management

Rule ID can be stored and transported in the FPort field of the LoRaWAN MAC frame or as the first bytes of the payload.

TBD

5.2. IID computation

TBD

5.3. Fragmentation

TBD

5.3.1. Reliability options

5.3.1.1. Uplinks: From device to gateway

In that case the device is the fragmentation transmitter, and the SCHC gateway the fragmentation receiver.

- o SCHC fragmentation reliability mode : "ACK_ALWAYS"

- o Window size: 8, the FCN field is encoded on 3 bits

- o DTag : 1bit. this field is used to clearly separate two consecutive fragmentation sessions. A LoRaWAN device cannot interleave several fragmented SCHC datagrams.

- o MIC calculation algorithm: CRC32 using 0xEDB88320 (i.e. the reverse representation of the polynomial used e.g. in the Ethernet standard [RFC3385])

o Retransmission Timer and inactivity Timer: LoRaWAN devices do not implement a "retransmission timer". At the end of a window the ACK corresponding to this window is transmitted by the network gateway in the RX1 or RX2 receive slot of the device. If this ACK is not received the device sends an all-0 (or an all-1) fragment with no payload to request an 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 gateway implements an "inactivity timer". The default recommended duration of this timer is 12h. This value is mainly driven by application requirements and may be changed.

```
| RuleID | DTag  | W     | FCN    | Payload |
+ ------ + ----- + ----- | ------ + ------- +
| 3 bits | 1 bit | 1 bit | 3 bits |         |
```

Figure 2: All fragment except the last one. Header size is 8 bits.

```
| RuleID | DTag  | W     | FCN    | MIC     | Payload |
+ ------ + ----- + ----- | ------ + ------- + ------- +
| 3 bits | 1 bit | 1 bit | 3 bits | 32 bits |         |
```

Figure 3: All-1 fragment detailed format for the last fragment.
Header size is 8 bits.

The format of an all-0 or all-1 acknowledge is:

```
| RuleID | DTag  | W     | Encoded bitmap | Padding (0s) |
+ ------ + ----- + ----- | -------------- + ------------ +
| 3 bits | 1 bit | 1 bit | up to 8 bits   | 0 to 3 bits  |
```

Figure 4: ACK format for All-0 windows. Header size is 1 or 2 bytes.

```
| RuleID | DTag  | W     | C     | Encoded bitmap (if C = 0) | Padding (0s) |
+ ------ + ----- + ----- + ----- + ------------------------- + ------------ +
| 3 bits | 1 bit | 1 bit | 1 bit | up to 8 bits              | 0 to 2 bits  |
```

Figure 5: ACK format for All-1 windows. Header size is 1 or 2 bytes.

5.3.1.2. Downlinks: From 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.

- SCHC fragmentation reliability mode : ACK_ALWAYS
- Window size : 1 , The FCN field is encoded on 1 bits
- DTag : 1bit. This field is used to clearly separate two consecutive fragmentation sessions. A LoRaWAN device cannot interleave several fragmented SCHC datagrams.
- MIC calculation algorithm: CRC32 using 0xEDB88320 (i.e. the reverse representation of the polynomial used e.g. in the Ethernet standard [RFC3385])
- MAX_ACK_REQUESTS : 8

RuleID	DTag	W	FCN	Payload	Padding
3 bits	1 bit	1 bit	1 bits	X bytes	2 bits

Figure 6: All fragments but the last one. Header size is 6 bits.

RuleID	DTag	W	FCN	MIC	Payload	Padding
3 bits	1 bit	1 bit	1 bits	32 bits	X bytes	2 bits

Figure 7: All-1 Fragment Detailed Format for the Last Fragment. Header size is 6 bits.

The format of an all-0 or all-1 acknowledge is:

RuleID	DTag	W	Encoded bitmap	Padding (0s)
3 bits	1 bit	1 bit	1 bit	2 bits

Figure 8: ACK format for All-0 windows. Header size is 8 bits.

```
| RuleID | DTag  | W     | C = 1 | Padding (0s) |
+ ------ + ----- + ----- + ----- + ------------ +
| 3 bits | 1 bit | 1 bit | 1 bit | 2 bits       |
```

Figure 9: ACK format for All-1 windows, MIC is correct. Header size is 8 bits.

```
| RuleID | DTag  | W     | b'111  | 0xFF (all 1's) |
+ ------ + ----- + ----- + ------ + -------------- +
| 3 bits | 1 bit | 1 bit | 3 bits | 8 bits         |
```

Figure 10: Receiver ABORT packet (following an all-1 packet with incorrect MIC). Header size is 16 bits.

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

5.3.1.2.1. Class A devices

Class A devices can only receive in an RX slot following the transmission of an uplink. Therefore there cannot be a concept of "retransmission timer" for a gateway talking to classA devices for downlink fragmentation.

The device replies with an ACK fragment to every single fragment received from the 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 transmit the ACK fragment until it receives the fragment of the next window. The device shall transmit up to MAX_ACK_REQUESTS ACK fragments before aborting. The device should transmit those ACK as soon as possible while taking into consideration eventual 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=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. This message might be lost therefore the 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 from the gateway different from an ACK-request indicating that the gateway has received the ACK message.

Following the reception of a FCN=1 fragment (the last fragment of a datagram) and if 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 from the gateway different from an ACK-request indicating that the gateway has received the ABORT message. The fragmentation receiver (device) does not implement retransmission timer and inactivity timer.

The fragmentation sender (the gateway) implements an inactivity timer with default duration 12 hours. Once a fragmentation session is started, if the gateway has not received any ACK or receiver-ABORT message 12 hours fater the last message from the device was received, the 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.

5.3.1.3. Class B or C devices

Class B&C devices can receive in scheduled RX slots or in RX slots following the transmission of an uplink. The device replies with an ACK fragment to every single fragment received from the 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 ACK fragment even if that fragment has already been received. The ACK bitmap is 1 bit long and is always 1. If the gateway receives this ACK, it proceeds to send the next window fragment If the retransmission timer elapses and the gateway has not received the ACK of the current window it retransmits the last fragment. The gateway tries retransmitting up to MAX_ACK_REQUESTS times before aborting.

Following the reception of a FCN=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 gateway receives this ACK, the current fragmentation session has succeeded and its context can be cleared.

If the retransmission timer elapses and the gateway has not received the all-1 ACK it retransmits the last fragment with the payload (not an ACK-request without payload). The gateway tries retransmitting up to MAX_ACK_REQUESTS times before aborting.

The device SHALL keep the all-1 ACK message in memory until it receives a downlink from the gateway different from the last (FCN=1) fragment indicating that the gateway has received the ACK message. Following the reception of a FCN=1 fragment (the last fragment of a datagram) and if the MIC is NOT correct, the device shall transmit a

receiver-ABORT fragment. The retransmission timer is used by the gateway (the sender), the optimal value is very much application specific but here are some recommended default values. For classB 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. (modify 128sec) For classC devices which are nearly constantly receiving, the recommended value is 30 seconds. This means that the device shall try to transmit the ACK within 30 seconds of the reception of each fragment. The inactivity timer is implemented by the device to flush the context in-case it receives nothing from the gateway over an extended period of time. The recommended value is 12 hours for both classB&C devices.

5.3.2. Supporting multiple window sizes

TBD

5.3.3. Downlink fragment transmission

TBD

5.3.4. SCHC behavior for devices in class A, B and C

TBD

6. Security considerations

TBD

7. Acknowledgements

TBD

8. References

8.1. Normative References

[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>.

[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>.

8.2. Informative References

[I-D.ietf-lpwan-ipv6-static-context-hc]
Minaburo, A., Toutain, L., Gomez, C., and D. Barthel, "LPWAN Static Context Header Compression (SCHC) and fragmentation for IPv6 and UDP", draft-ietf-lpwan-ipv6-static-context-hc-16 (work in progress), June 2018.

[I-D.ietf-lpwan-overview]
Farrell, S., "LPWAN Overview", draft-ietf-lpwan-overview-10 (work in progress), February 2018.

[lora-alliance-spec]
Alliance, L., "LoRaWAN Specification Version V1.0.2", <http://portal.lora-alliance.org/DesktopModules/Inventures_Document/FileDownload.aspx?ContentID=1398>.

Appendix A. Examples

Appendix B. Note

Authors' Addresses

Nicolas Sornin (editor)
Semtech
14 Chemin des Clos
Meylan
France

Email: nsornin@semtech.com

Michael Coracin
Semtech
14 Chemin des Clos
Meylan
France

Email: mcoracin@semtech.com

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

Email: ivaylo@ackl.io

Alper Yegin
Actility
.
Paris, Paris
France

Email: alper.yegin@actility.com

Julien Catalano
Kerlink
1 rue Jacqueline Auriol
35235 Thorigne-Fouillard
France

Email: j.catalano@kerlink.fr

Vincent AUDEBERT
EDF R&D
7 bd Gaspard Monge
91120 PALAISEAU
FRANCE

Email: vincent.audebert@edf.fr