SCHC Working Group
Internet-Draft
Intended status: Informational
Expires: 30 July 2026

E. Ramos
L. Corneo
Ericsson
A. Minaburo
Consultant
R. Munoz-Lara
Departamento de Ingenieria Electrica, Universidad de Chile
S. Cespedes
Concordia University

26 January 2026

Static Context Header Compression and Fragmentation over networks prone to disruptions
draft-ietf-schc-over-networks-prone-to-disruptions-03

Abstract

This document describes the use of SCHC over different network topologies and devices regardless of their capabilities and configurations. The use of SCHC will bring connectivity to devices with disruptive connections caused by restrained use of battery and connectionless setups with long delays and latency.

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 30 July 2026.

Copyright Notice

Table of Contents

1. Introduction

A network prone to disruptions (NPD) is a type of communications network where the devices (Dev) may be in the presence of long delays or intermittent connectivity. Unlike conventional networks that depend on uninterrupted, low-latency connectivity, networks prone to disruptions are designed to manage extended interruptions and substantial delays in data transmission. By employing methods like buffering and data forwarding, they ensure that information

ultimately arrives at its destination, even if a direct connection is not consistently present. NPDs are especially useful in scenarios like space exploration, ZE devices, or emergency situations where standard communication infrastructure is either lacking or unreliable. This document explains the different topologies and how SCHC can improve communication in such networks.

This document normatively references [RFC5234] and has more information in 3GPPdocA and 3GPPdocB. (REPLACE)

2. Conventions and Definitions

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.

3. Devices Types

ZE-Devices based in cellular Zero Energy (ZE) devices are ultra-low-power small electronic circuits that can be used in Internet of Things (IoT) applications. Typically, a ZE device solely relies on the energy that is harvested from the surrounding environment through an energy harvester, e.g., a small solar panel or Radio Frequencies (RF). The harvested energy is often stored in small rechargeable batteries or super-capacitors. However, the most constrained ZE devices are completely passive and could lack energy storage. ZE energy devices typically contain sensors, e.g., temperature, as well as a radio interface to offload sensor readings.

ZE devices do not require any battery replacement, or manual charging, as they harvest energy from their surrounding environment. ZE devices might be small, and come in the form of sensors (which report on data from readings and measurements), trackers (which report on the location of an object or a living being), or actuators (which prompt other machines to operate).

The widespread adoption of ZE devices will lead to a massive reduction in both the cost and power needed to run and maintain IoT systems, making them more scalable. Gathering data from these devices also has the potential to drive higher productivity, pollution reduction, and enriched lifestyles, without requiring any additional energy. Furthermore, battery-less devices are better for the environment and can be managed with simple processes, from manufacturing to disposal.

3.1. 3GPP device classification

At the time of writing, the 3GPP TR 38.848 collects decisions regarding "Ambient IoT", which is another name for ZE IoT used throughout this draft. In that document, three different types of ZE devices are specified based on their energy storage capacity and their RF transmission capabilities.

* Device type A: Fully passive devices, without any energy storage capability. The peak power consumption is expected to be less than 10 uW. The wireless communication technology used is backscatter communication.

* Device type B. Semi-passive devices with limited energy storage, e.g., super-capacitor or coin-cell battery. The peak power consumption is expected to be in the order of few hundreds of uW. The wireless communication technology used is backscatter communication with the stored energy possible to be used for amplification of the backscattered signal.

* Device type C. Active devices with energy storage. The peak power consumption is expected to be less than 10 mW. The wireless communication technology used is active communication and independent signal generation.

The type of devices A, B, and C are able to demodulate control, data, etc from the relevant entity in RAN according to connectivity topology.

3.1.1. 3GPP ZE IoT topologies

3GPP currently discusses four topologies to enable communication between ZE devices and the cellular network. Most capable ZE devices may be able to communicate directly with a base station (BS). On the other hand, more constrained ZE devices may need the assistance of intermediary nodes, for example, to provide carrier signals or energy to excite and power up the device. We would focus so far on the topology 1 in this document.

3.1.1.1. Topology 1

In Topology 1, see Figure 1, the ZE device directly and bidirectionally communicates with a base station (BS). The communication between the BS and the ZE device includes device data and/or signaling.

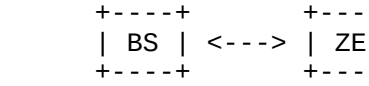

Figure 1: Topology 1. The base station (BS) and ZE device communicate directly.

3.1.1.2. Topology 2

In Topology 2, see Figure 2, the ZE device communicates bidirectionally with an intermediate node (IN) between the device and BS. In this topology, the intermediate node can be a ZE-enabled relay, such as a user equipment (UE), meaning other mobile device or equipment, or a repeater. The IN transfers ZE data and/or signaling between BS and the ZE device.

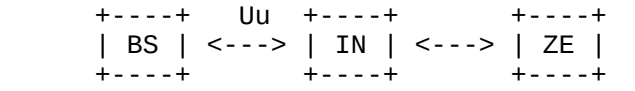

Figure 2: Topology 2. The base station (BS) and ZE device communicate through an intermediary node (IN).

3.1.1.3. Topology 3

In Topology 3, see Figure 3 and Figure 4, the ZE device transmits data/signalling to a BS, and receives data/signalling from the assisting node (AN). Alternatively, the ZE device receives data/ signaling from a BS and transmits data/signaling to the AN. In this topology, the AN can be a ZE-enabled relay, for example, another UE.

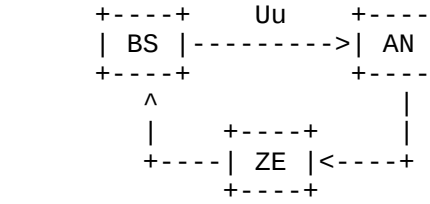

Figure 3: Topology 3 (downlink assistance). The base station (BS) utilizes an assisting node (AN) to transmit data to the ZE device.

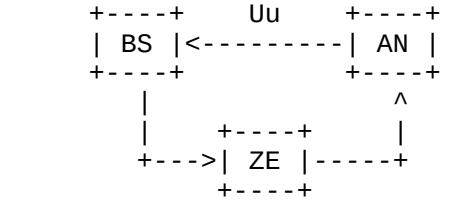

Figure 4: Topology 3 (uplink assistance). An assisting node (AN) relays to the base station (BS) the ZE UL transmission.

3.1.1.4. Topology 4

In Topology 4, see Figure 5, the ZE device communicates bidirectionally with a UE. The communication between UE and the ZE device includes ZE data and/or signaling.

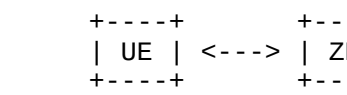

Figure 5: Topology 4. A user equipment (UE) and ZE device communicate directly.

3.1.2. User plane characteristics for a Cellular ZE-devices

The nature of the ZE devices requires some changes in the architecture of the radio network protocol stack to minimize the power consumption on the transmissions and simplify operations. The reception of data, even control signaling, also requires energy.

In a design for ZE devices design, the energy that is harvested is preferred to be used for the device’s transmissions. Since the ZE devices are expected to have highly uplink-dominated traffic, and therefore the minimization of downlink transmissions (including feedback) can be anticipated.

Also, the transmission opportunities and characteristics require that the handling of the packets is tolerant to delays in the reception and reassembling due to the inherent unreliability of the source of power for such transmissions. Even so, these devices coexist with legacy and the more capable devices that will be utilizing the same mobile networks, and the changes should be compatible with the type of equipment that is typically utilized for cellular networks to favor adoption and economy of scale.

Due to the restricted power on ZE devices, the user plane is expected to be simplified and optimized to reduce the overhead and the need for handling multiple levels of feedback. The power restriction itself and the possible lack of link adaptation and reduction of the feedback might increase the probability of packet loss and in some scenarios also the probability of interference. This is due to the deployment of many devices in close vicinity that are power-charged by the same type of energy source and therefore possibly activated simultaneously, which may cause access collision to the network as well as interference to other cells.

The mentioned restrictions make the design of the user plane for these kinds of devices is challenging and requires compromises on the current design. This would imply an iterative approach on what components and procedures are kept and which ones are new with respect to the regular cellular devices' operation.

For example, to increase the efficiency, the transmissions may be done at the same time as accessing the network, meaning the utilization of the RACH (Random Access Channel) to reduce the control signaling. Transmissions using RACH are susceptible to collision since they are mostly multiplexed by preambles and timing chosen randomly by the device and currently are not scheduled as the traditional user plane transmission are. The minimization of downlink signaling may have an impact on the possibility of having scheduled traffic, in addition to the impossibility of the network of knowing if a device has enough energy to monitor a particular downlink signaling channel.

The need to reduce overhead and optimize the number of bits over the air to reduce the power required to transmit is a clear requirement of the ZE devices. Consequently, the use of SCHC (Static Context Header Compression) [RFC8724] has a great potential to reduce the quantity of data needed to be sent over the air, as well as provide elements that can be used to increase reliability, support for fragmentation, and potentially manage the problem of the long delays between transmissions. The delays may happen when a device has just enough energy for transmitting certain packets but not enough to empty the buffer. Part of the energy might be needed for the reception of packets from the network.

The network is capable of managing the possibility that a full object might not be received soon after a transmission is started. This increases the requirement of how long the fragments and packet might need to be kept in buffers, so it is avoided to lose the energy that the devices have used in the initial transmission(s). This enables that the device can continue with the rest of the packets once the power for a new transmission has been harvested. Of course, the

buffers should be stored as long as it makes sense for the use case of the device, and therefore it might require certain degree of configuration, in some cases at the devices and in others at the network, or both.

The possibility of collisions between transmissions and the lack of power control and link adaptation may affect the reliability of the delivery of packets. But still, the restriction of power for transmitting and reception and the delays make challenging the support for reliability based on retransmissions. In this respect, we could think that there is a trade-off between the reliability and additional delay in receiving the data. In some scenarios, these delays could make sense and in others, the delay could make the packets irrelevant to their use case. In that sense equally to the previous point, the configuration of the delays targeting for reliability is important.

From the required characteristics outlined for the user plane, the use of SCHC becomes relevant to fulfill them. SCHC offers fragmented packet corruption detection, and delivery reliability window-based mechanisms, such as ACK-always (Each fragment delivery is explicitly acknowledged) and ACK-on Error (only detected losses trigger delivery reports outlining the fragment loss).

The requirements can be addressed with some additional complements to support the deployment of SCHC into the cellular Zero Energy device scenarios. For example, adding support for object transport in contrast to only IP packet support, and providing better management of long delays. In addition, a solution to enable the set up of the contexts and rules that make sure there is alignment between the network and the devices on the management of packets. Part of this can be accomplished by imagining that a complete object fits an imaginary jumbo IP package and SCHC would then fragment such packet into pieces that can be fitted in the radio transport block.

In this way, a great part of the overhead is removed and the SCHC services would take care of the reliability and delay-friendly transmission of packets. In addition, there is the possibility of integrating even further SCHC to the cellular lower protocol layers, for example by not relying on feedback from MAC for the reliability of transmission of packets but instead using the fragment bitmap from SCHC. This also may improve the power efficiency of each transmission since the device does not need to monitor the feedback channel after each transmission.

The big challenge in using SCHC in this fashion is how to configure the SCHC fragmentation and reassembly entities. A Dev using SCHC and the endpoint where SCHC is terminated in the network with the

relevant context information so the transmitter and the receiver have an understanding of what are the parameters of operation for this particular case, which would depend on the network load and devices power availability for transmission and the maximum allowed delay configuration.

At the moment it is unclear if the backscattering devices (devices type A and B) will support IP connectivity from the device itself, the current cases being analyzed are leaning towards the transmission of one ID when the backscatter signal is activated. In that sense, the applications would require intermediate platforms to fetch the data and the onboarding procedure would require an association of the device to an identifier that could be exposed through an API or IP tunneling from non-IP traffic services.

3.1.2.1. End-to-end view

The traffic characteristics of ZE devices and their use case might drive the development of the end-to-end interactions and protocol stack. In mostly uplink-dominated cases, the device would produce information that needs to be collected due to the potential delays by a platform instead of being transmitted to a particular application due to the requirement of availability. In the case of applications using the generated data, it would in most cases fetch the data from such platforms, and therefore the connectivity towards the final application might not be direct. Therefore, it is highly probable that the direct communication stack can in most cases be assumed to be mediated by a data collection platform.

One option is that such a platform is provided by operators. In that case, it makes sense to incorporate SCHC as part of the protocol stack between the network and the terminal. This option would require some knowledge of the application protocol stack by the mobile network so that effective compression can be realized. This type of deployment would maximize the energy efficiency by optimizing the compression up to the transport block level reducing additional overhead from padding and lower layers headers. In this scenario the application would only receive the payload whenever a packet or an object is fully assembled, reducing the need for additional implementation to application logic. When transmitting a complete object in full, SCHC could be utilized in a similar way to a transport protocol due to its fragmentation features. They enable transmissions over long periods of time and reconstruct the full object after receiving all fragments and also provide some reliability control on the fragments transmitted.

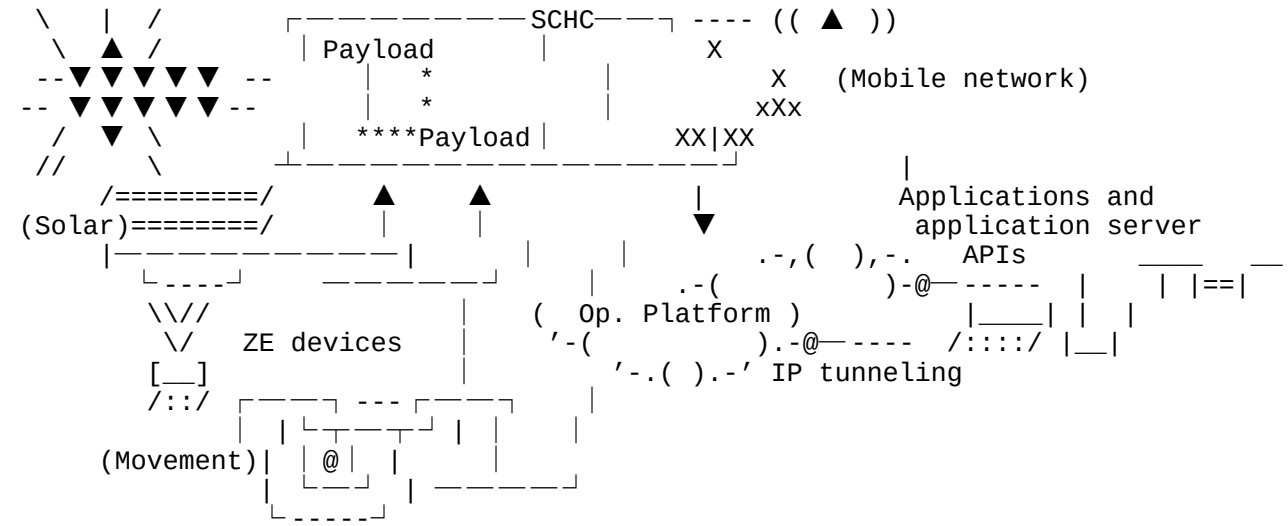

Figure 6: Platform exposing the ZE devices data

This means that the device has to be onboarded in the platform with a unique identifier that is associated with the SCHC flow. Then such an identifier is utilized to map the transmissions to the applications requiring the data. In some cases, this identifier may be supplied and configured out-of-band by auxiliary procedures, since the device might not have the capability to onboard itself to and endpoint.

The applications require support to notifications of data available in a similar fashion to a pub-sub system. In this way, the application can request the information from the corresponding API. In the case of an IP tunnel, since the connection may not be up during the whole time, it would require forwarding the object to a specific location where the application can fetch the transmitted object.

Another option is the enabling of configurable data collection platforms, which would imply providing SCHC support over the top in the application layer. For this option, the SCHC packets would look like non-IP traffic for the network, and the reliability of the packets, delay management, and reassembling of fragments need to be handled by the application. Therefore, the delays in transmissions and changes in network connection points need to be handled and accounted for.

3.2. Direct-to-satellite IoT (DtS-IoT) devices

Direct-to-satellite IoT communication (DtS-IoT) is the direct communication between end devices and the satellite in an IoT environment without using a terrestrial LPWAN gateway as an intermediate connection element (radio gateway for LoRaWAN and eNodeB for NB-IoT). In DtS-IoT, the LPWAN gateway is located on the satellite. When DtS-IoT technology uses a single low earth orbit (LEO) satellite or a sparse constellation of LEO satellites, the communication between IoT nodes and the satellite is prone to disruptions.

In order to deal with disruptions, end devices and LEO satellites use a mechanism for sending messages called store and forward. For uplink communications, when an end device transmits and is not in satellite coverage, the end device stores messages in a queue until there is visibility. Similarly, when the satellite receives the end-devices message, it is stored until connection to the ground station becomes available.

3.2.1. DtS-IoT Topology

A DtS-IoT network is composed of three types of nodes:

* End-devices: These are devices located at the edge of the network. They are usually memory and power-constrained devices. The end devices access the network through LEO satellites.

* LEO satellite: a satellite orbiting the Earth at altitudes between 160 and 2,000 kilometers. Due to its proximity, the visibility window between an end-device and a LEO satellite is in the order of minutes and its periodicity is generally less than two hours.

* Ground Station: A network element that interconnects a LEO satellite with a terrestrial network (e.g., Internet).

In the figure Figure 7, the end device communicates bidirectionally with the Ground Station via a LEO satellite.

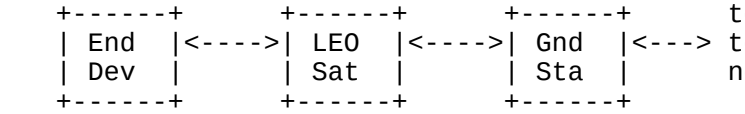

Figure 7: Topology DtS-IoT.

3.2.2. Disruptions in DtS-IoT

In a DtS-IoT environment, disruptions between the ground nodes (end device and ground station) and LEO satellite occur due to the fast speed and short line-of-sight duration between the satellite and ground stations.

From the point of view of the ground nodes (end device and ground station), there are two-time windows associated with interrupts.

* Visibility window (visibility time): the time during which a device on the ground can communicate with a satellite.

* Pass-to-pass window (revisit time): is the time between the end of a visibility window (pass i) and the beginning of the next visibility window (pass $i+1$) for the same device on the ground. During this time the ground device cannot communicate with the satellite.

Due to the asynchronous communication provided by the two time windows, the *transfer delay* of a SCHC packet increases.
Figure Figure 8 shows the message flow for sending a SCHC window with its corresponding acknowledgement through a DtS-IoT environment.
Note that the increase in the transfer delay of a SCHC packet is directly related to the revisit time, since the fragmenter (end-device) waits for a SCHC ACK when an SCHC window has been completely transmitted or when an ACK REQ has been sent, as defined in [RFC8724].

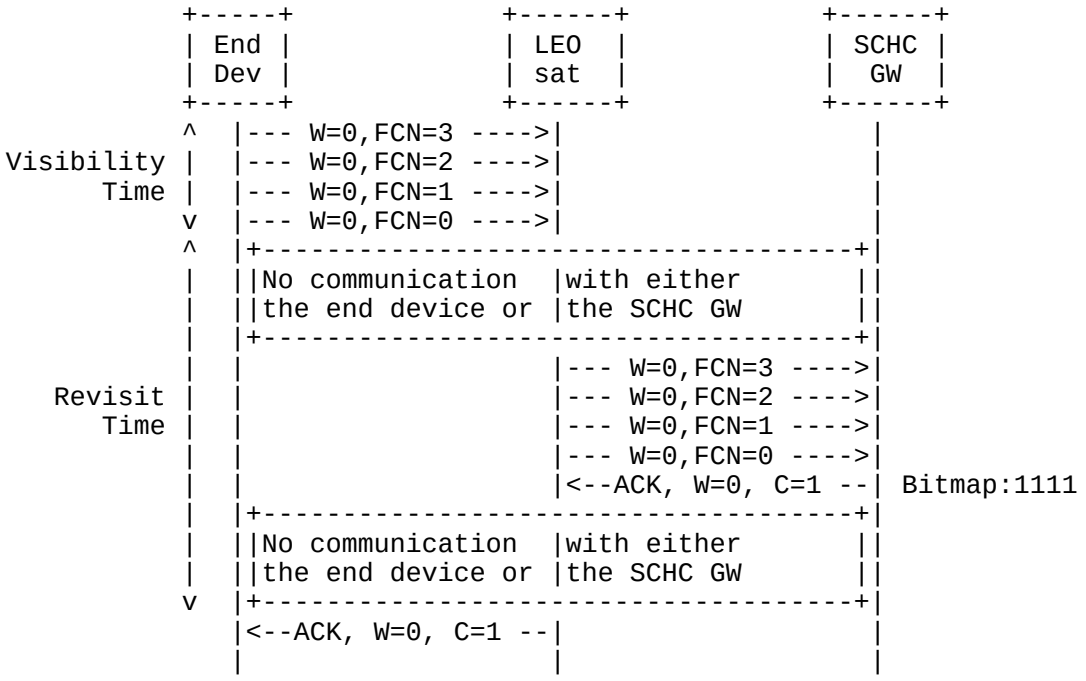

Figure 8: Message flow for an SCHC session in a DtS-IoT environment.

4. SCHC as a size and delay-optimized transmission mechanism

SCHC mechanisms can be used to provide reliability and segmentation and then extended to provide delay-tolerant transmissions of large objects. This can be done by using the SCHC Fragmentation/Reassembly mechanism Ack on Error [RFC8724] which divides the object into smaller chunks called tiles that are transmitted according to a network's scheduled occasions considering the device power saving and state configuration. The configuration and setup of SCHC object transfer session considering the network and terminal states according to the needs of each device matching to their use case becomes a critical functionality to address. [new text]For this case SCHC would become a simple transport protocol for the whole object instead of only fragmenting IP packets which is different from what has been specified by RFC [RFC8724].

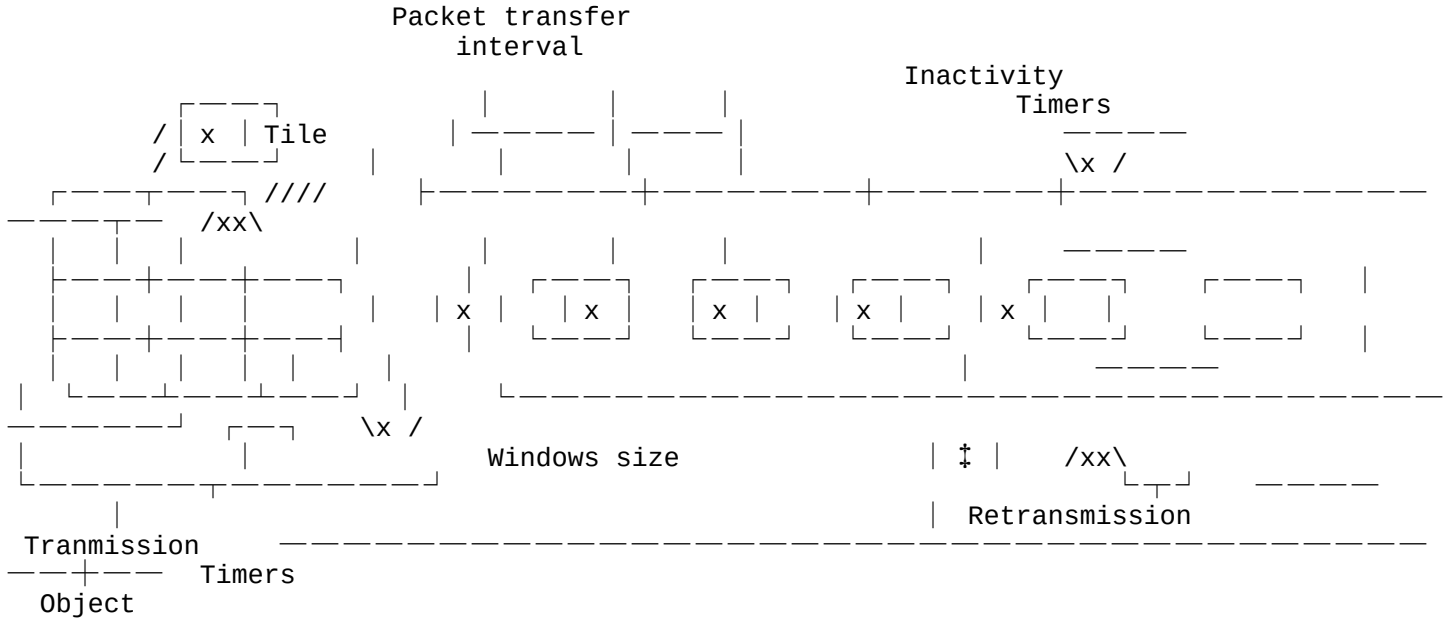

Figure 9: Object Fragmentation utilizing SCHC fragmentation

4.1. General architecture

The Figure 10 shows a high configuration of the network communication between a Device and an Application Server (App). Dev has short-live intermittent connections and needs a middle host called proxy that will maintain the connection state even if the communication is discontinued with the Dev and continued communication with the Application Server. The proxy may answer some requests instead of the Dev.

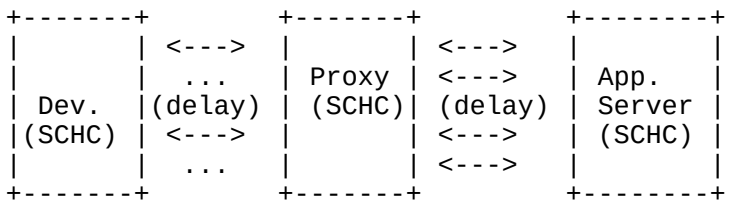

Figure 10: High Level Communication Architecture

4.2. SCHC in ZE-Devices based on cellular

4.2.1. Device-initiated transmissions

Once a device is onboarded into a network, or during the network connection procedure, it must be configured with a new threshold value MAX_OBJECT_SIZE, measured in bytes). This configuration could be also pre-defined and notified to the network using out-of-band methods. This is used to compare the object size to be transmitted. If the object size exceeds such threshold, it means that it is required to operate with a delay-friendly transmission configuration and it will use the most adequate SCHC delay values that are capable of handling the object size to be transmitted by the device. The most adequate configuration is such that can handle (bigger or equal) the size of the object to be transmitted according to the MAX_OBJECT_SIZE associated configuration.

To avoid collisions and help with the network management of multiple devices accessing the network simultaneously, the configuration could include a Best Effort Transfer Interval (BETI). A BETI configuration is meant to provide pacing information to the SCHC device. After each BETI the device attempts to transfer a number of SCHC tiles. The value of BETI could be based on a timer (send new fragment every X second), transmission occasions (send every X occasion), or radio events (paging, DRX/DTX cycle, etc.). Also, the values of BETI can be also determined by a random timer given by a configured range. The number of tiles to send in each BETI, a Tile Count (TC) parameter, is by default 1 but can be configured by the network to be a higher number.

The SCHC Rule for these devices may be a well-known rule that will not need to be updated. If the Proxy has several devices attached, it must recognize which one is sending.

4.2.2. Network initiated transmission

If there is a need for the network to transmit data to a device in some cases may require transmitting to a large number of devices and potentially even the same network delivery points (e.g., radio base stations). To accomplish this in a scenario where the compressor entity is in the cellular network, it will need to have a copy of the object to be delivered to the device to transmit it to the device according to a suitable scheduling and agreed configuration. As mentioned before, this would require the network to provide APIs to Applications Servers (AS) that either provide an interface to upload to the network the object to be transferred beforehand or a proxy IP address for large object transfers that would buffer the object for further transmission if the data were from the application layer. The delivery may reuse the same mechanisms used to provide IP tunneling transmissions or non-IP transmissions already specified in

the cellular standards.

4.2.3. SCHC context configuration and additional parameters for ZE transmission

4.2.3.1. Context provisioning

SCHC successful header compression happens only when a common context is shared between sender and receiver. Typically, context provisioning is outside the scope of SCHC RFC documents, mainly because there may be several ways to implement it. However, the most constrained ZE devices, e.g., 3GPP ZE type 0, may not be able to receive packets from the network, thus dramatically restricting context provisioning possibilities. Hence, this document also discusses how a SCHC context may be provisioned to ZE devices with no reception capabilities.

Discussion of the possibilities:

* Standardized set of rules that ZE device manufacturers include in their firmware. Viable solution but may lead to even more heterogeneity in the IoT ecosystem. In fact, different vendors may support different non-overlapping subsets of SCHC contexts or none at all.

* Third-party entities or device owners upload and maintain the SCHC contexts, for example flashing the MCU. Manual process and not really scalable.

* NFC or equivalent interfaces for SCHC context provisioning. Add costs for the interface, it is a non-scalable manual process.

* Use of well-known rules, provisioned at device configuration.

4.2.3.2. Context updating

Since SCHC works with static context information, it is not likely (or desired) to update the SCHC delay tolerant configurations very often (e.g., more than once a week -- what exactly is "often" depends on the device capabilities and typical communication frequency), so the most feasible options are that the network would produce a set of pre-configured configurations that are addressed individually with a configuration ID. This means that the network could configure, for example, rules for one device for maximum SCHC packet size large, medium, and small and use three context groups where it applies this parameter setting. In turn, the SCHC MAX_PACKET_SIZE will be set to such values.

In the case of SCHC being utilized as a transport protocol to transmit an object, the size of the tiles used to fragment the object could be set to the MTU of the bearer where the transmission will be realized, for example, if the data is transmitted using regular transmission channels, the MTU would be 1358 bytes in most of the cases. The SCHC standard fragmentation inactivity timers and fragmentation retransmission timers can be also set according to the scheduling calculation and the expected time of delivery (based on the schedule) for the large packets. Those timers are applied to the fragments that are transmitted and their acknowledgments.

The network can use the expected scheduling time for one of the rule groups and set several parameters according to multiple scheduling situations, for example, extra-long delay, long delay, medium delay, sort delay, and no delay. In a situation with a delay configuration, the retransmission timer and the inactivity timer would be set to a reasonable value (e.g., 24 hours), meanwhile, in no delay settings, the timers would be set to significantly smaller values (e.g., 10 minutes). The values of the timers would be also correlated to the SCHC window (i.e., successive tiles in a group) size selected which translates to how many transmissions of the tiles are expected to check the correct reception of the tiles belonging to one window. Shorter timers would correspond to shorter window sizes (i.e., a smaller number of tiles would be sent, and hence shorter retransmission/inactivity time is appropriate), meanwhile, larger timer values would correspond to larger window sizes. The window size would also depend on how many tiles the object is fragmented into.

The profile also would have information in reference to the maximum number of Attempts, meaning how many retransmissions of one packet (after the retransmission timer has expired) should be attempted before aborting the transmission. In cases of devices with a history of bad coverage (known from, e.g., connectivity logs for that device), this setting could be set to a higher number (for example 10), and in more common cases for a cellular network where reliability is high, to just one retransmission. Similarly, if the uplink seems to be the problem, then the adjustment could be done in the MAX_ACK_REQUESTS, where the sender would poll the receiver to transmit a bitmap with the received packets if needed and retransmit the request if the retransmit timer expires the number of times that MAX_ACK_REQUESTS is configured to.

4.2.3.3. Payload compression

This section describes how the SCHC framework may be used to compress payload, in addition to the headers for which it was initially designed. As ZE devices must minimize the number of transmitted bits, due to their energy constraints, payload compression may provide significant gains in that respect. Since the compression (and decompression) functionality is already implemented in the device, then the same engine could be re-utilized to provide compression of the payload when possible. This would mean that a section of the context MUST be dedicated to the payload and separated from the header compression part.

An example of payload compression targeting key-value-based formats is now provided. Specifically, SenML [RFC8428] is used to encode a typical IoT payload, as shown below:

json [{"bn":"2001:db8:1234:5678::1/", "n":"temperature", "u":"Cel", "v":25.2}, {"n":"humidity", "u":"%RH", "v":30}]

The above SenML pack includes two SenML records, JSON objects, that describe the temperature and humidity of two sensors.

A SCHC rule defined for the above SenML payload may be defined as follows:

FID	FL	FP	DI	TV	MO	CDA	Sent [bits]
application/senml+json.bn.1	22	1	Up	2001:...	equal	not-sent	0
application/senml+json.n.1	11	1	Up	temp...	equal	not-sent	0
application/senml+json.u.1	3	1	Up	Cel	equal	not-sent	0
...	..	..	..	...	...	...	...
application/senml+json.n.2	8	1	Up	hum...	equal	not-sent	0
...	..	..	..	...	...	...	...

The next paragraphs demonstrate how the SCHC compressor may perform the matching of the SenML payload presented above.

The compressor starts from the first entry in the table above, where the first FID is application/senml+json.bn.1. Here, the FID's name has been encoded in a way to describe the content type, application/ senml+json, the SenML field, bn, and the identifier of the object containing such field, 1. To be noticed, a dot, . has been used as a separator, although other symbols may be used.

The compressor must now inspect the payload looking for the field bn of the first object, 1, in the SenML payload that is being compressed. When the field bn is found, the MO is performed against the TV, the base name of the sensor, and the CDA is performed. The compressor now moves to the next FID in the SCHC rule and repeats the above.

This type of payload compression is devised specifically for key-value-based formats, such as JSON. However, other types of formats may be supported as long as the compressor implements the logic to parse the semantics behind the FID.

4.2.3.4. Fragmentation parameters

Due to the different types of devices and their energy harvesting capabilities, the actual parameters to fragment the objects have to consider these differences. As it is difficult to reconfigure these devices because energy is needed for additional processing and receiving data, the best approach is to create some profiles that match the different types of devices. The profiles could depend on the size of packets that the device could manage as well as the expected time that the device might need to collect to send such packets. One proposal is to have 4 categories of time-based profiles:

* Latency mapping hours. The devices that map to this kind of profile would have a source of energy that can recharge the device at least once per hour. Therefore the retransmission timers can be set to a maximum of 6 hours, for example, and inactivity timers that may last 3 to 4 hours.

* Latency mapping a day. The devices may require a full day to recharge before sending a packet. Therefore the retransmission timer may take 4 or 7 days and an inactivity timer of 2 or 3 days.

* Latency mapping a week. In this case, very infrequently packets are sent and therefore it is not expected that a lot of data would be transmitted at once. Therefore retransmission timer and inactivity timer would be quite close to the transmission time. Could be 2 weeks for an inactivity timer and 3 weeks for a retransmission timer.

* Latency mapping a month. Similarly to the previous case, the values should also map close to transmission expectation. 2 months for the inactivity timer and 3 months for the retransmission timer.

Profiles related to packet sizes: * Single value packet * Multiple values in one packet * Multiple objects

4.3. SCHC for Low Power Wide Area (LPWA) Devices

The LPWA devices are devices whose architecture could vary a lot, ranging from small sensors to more complex devices with actuators, or even meters. Most of those devices are operated through batteries and are duty-cycled to reduce their power consumption. In some cases, they may sleep 10 hours and some implementation even switch off their receivers to save beyond what the network configuration could provide.

On one hand, the deployment of SCHC may enable transmissions to the device when it is back in service. On the other hand, SCHC may enable a device to receive a transmission as soon as the device resumes operation from dormant mode. If a device is not reachable, the network could cache the object to transmit, and transmit it using the delay-friendly features of SCHC. As such, the device can receive the data without triggering timeouts or packet loss. This avoids creating additional network traffic due to retransmissions and timeouts even before any packet has been sent. In addition to the uplink traffic, the data could be more efficiently sent in terms of power and with retransmissions based on nacked packets.

4.4. SCHC in DtS-IoT devices

When an end device in a DtS-IoT scenario uses SCHC as an optimized transmission mechanism, the main objective is to reduce the transfer delay of the SCHC packet. As indicated above, the transfer delay is directly proportional to the time that the fragmenter (end device) waits for the reception of a SCHC ACK. This document proposes two mechanisms to reduce this transfer delay.

1. LEO Satellite as a SCHC Proxy.

2. SCHC with FEC mechanism.

4.4.1. LEO Satellite as a SCHC Proxy

In this mechanism, the LEO satellite has the function of SCHC proxy. The SCHC proxy has two features to improve SCHC performance with local acknowledgments and retransmissions. These messages are employed to trigger local (and faster) error recovery. In both communication directions, it is recommended to use the fragmentation mode assigned by each profile.

* In uplink communications, the SCHC Proxy locally acknowledges SCHC regular fragments with an SCHC ACK message. Local acknowledgments split the SCHC connection between the end-device and SCHC Gateway. When local acknowledgments are used, the responsibility for retrieving any fragments after the proxy SCHC has acknowledged them lies with the proxy.

* In downlink communications, the SCHC Proxy retransmits locally the regular SCHC fragments that losses between SCHC Proxy and end-device.

4.4.1.1. Device-initiated transmissions

The figure Figure 11 shows the transmission of a SCHC window in an uplink communication using the SCHC message flows defined in [RFC8724]. The transmission of a SCHC window can be divided into two phases:

* *Phase 1:* In the visibility window, the end device sends the tiles to the LEO satellite. The tiles are carried by SCHC regular fragments. The tiles are stored in the LEO satellite. When the SCHC Proxy receives all tiles of a SCHC window, it sends to the end device a SCHC ACK. If the end device detects the loss of a tile(s) and the remaining visibility window time allows the tiles to be sent, the end device immediately retransmits the lost tile(s) without waiting for another visibility window.

* *Phase 2:* The LEO satellite sends the tiles stored in phase 1 to the SCHC gateway. The SCHC gateway receives the tiles and responds to the end device with an SCHC ACK message.

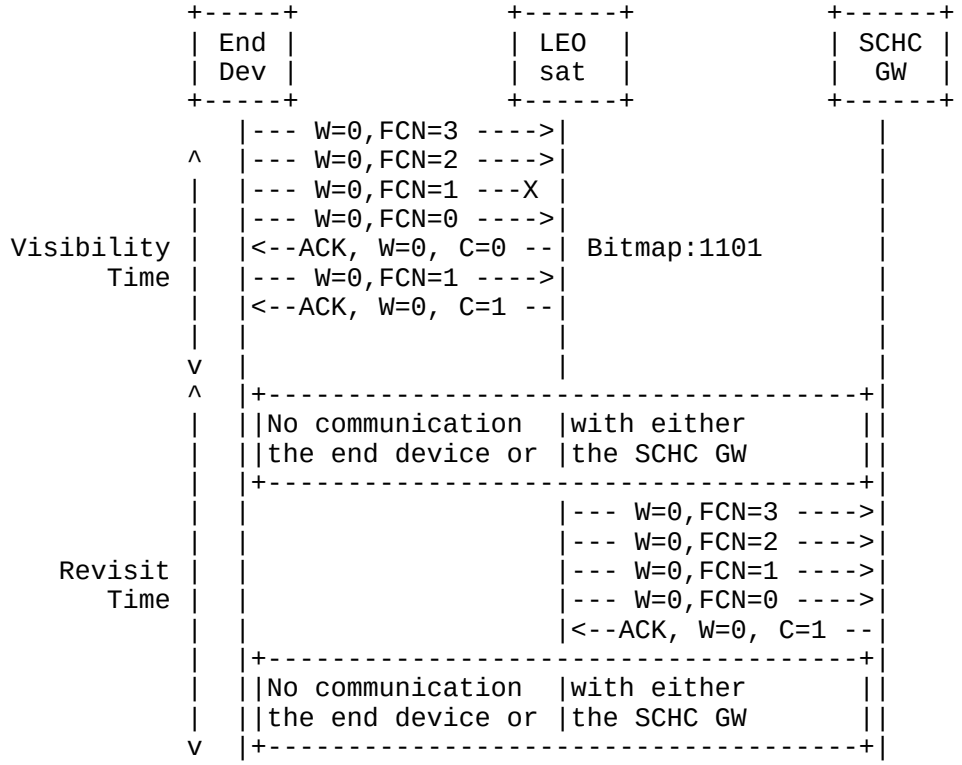

Figure 11: SCHC over DtS-IoT

4.4.2. SCHC with FEC mechanism

In this mechanism, the end-device (fragmenter) and the SCHC gateway (reassembler) use a Forward Error Correction mechanism (FEC) to protect the tiles. In this mechanism, the successful transmission of tiles MAY be confirmed by a SCHC ACK message. Figure Figure 12 shows the transmission of a SCHC session with FEC mechanism in a DtS-IoT environment. More details over the implementation in draft-munoz-schc-over-dts-iot

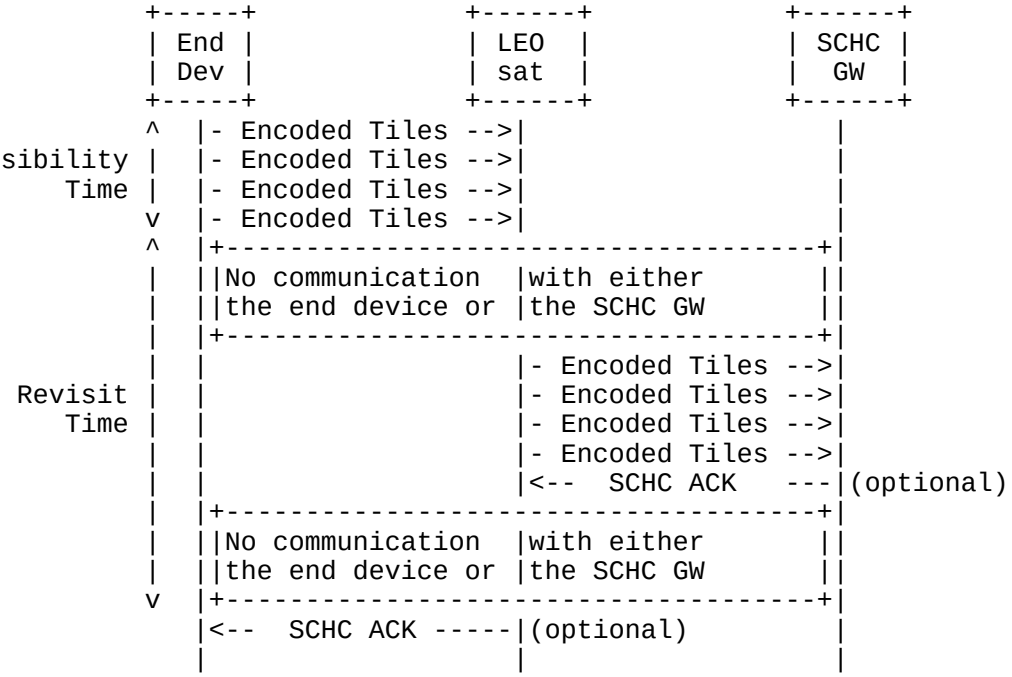

Figure 12: Transmission of an SCHC session using an FEC mechanism

5. IANA considerations

This document has no IANA actions.

6. Security considerations

This document does not add any security considerations and follows the [RFC8724].

7. 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/rfc/rfc2119>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/rfc/rfc5234>.

[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/rfc/rfc8174>.

[RFC8428] Jennings, C., Shelby, Z., Arkko, J., Keranen, A., and C. Bormann, "Sensor Measurement Lists (SenML)", RFC 8428, DOI 10.17487/RFC8428, August 2018, <https://www.rfc-editor.org/rfc/rfc8428>.

[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/rfc/rfc8724>.

Appendix A. Appendix A

This becomes an Appendix (REPLACE)

Appendix B. Acknowledgements

The authors would like to thank (in alphabetic order): ToDo

Authors' Addresses

Edgar Ramos
Ericsson
Hirsalantie 11
FI- 02420 Jorvas, Kirkkonummi
Finland
Email: edgar.ramos@ericsson.com

Lorenzo Corneo
Ericsson
Hirsalantie 11
FI- 02420 Jorvas, Kirkkonummi
Finland
Email: lorenzo.corneo@ericsson.com

Ana Minaburo
Consultant
35510 Cesson-Sevigne
France
Email: anaminaburo@gmail.com

Rodrigo Munoz-Lara
Departamento de Ingenieria Electrica, Universidad de Chile
Av. Tupper 2007, Santiago
Chile
Email: rmunozlara@ing.uchile.cl

Sandra Cespedes
Concordia University
1455 De Maisonneuve Blvd. W., Montreal QC, H3G 1M8
Canada
Email: sandra.cespedes@concordia.ca