SCHC Working Group
Internet-Draft
Intended status: Standards Track
Expires: 24 November 2025

M. Dumay
Orange Research

23 May 2025

SCHC Context lifecycle and management
draft-dumay-schc-context-lifecycle-00

Abstract

Static Context Header Compression and fragmentation (SCHC) framework provides both a header compression mechanism and an optional fragmentation mechanism.

SCHC operation is based on a common static Context stored at the two (or more) ends of a communication. As the scope of SCHC becomes broader, the static Context needs to evolve towards a more dynamic Context.

This purpose of this document is to initiate a discussion on the lifecycle of a Context. It describes the main steps in the lifecycle of a SCHC Context, and more broadly, it aims at documenting the related Requirements and Terminology for the proper use of SCHC. These goals are in line with the Working Group’s objectives, as set out in the Charter:

* "Produce an informational document describing how a carried protocol can use SCHC",

* "Produce Standard Track documents for SCHC Rule Discovery and Parameter Negotiation",

* "Produce Standard Track documents for SCHC Rule Provisioning".

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/.

Copyright Notice

Table of Contents

1. Introduction

The SCHC Working Group is tasked with extending the benefits of the [RFC8724] SCHC technology beyond Low-Power Wide-Area (LPWA) networks for which it was originally designed. The compression and fragmentation mechanisms described in [RFC8724] are based on a shared static Context between entities participating in the same communication.

A Context contains a set of rules that specify how messages are to be compressed/decompressed and/or fragmented/reassembled.

The new SCHC roadmap requires an evolution in the concept of Context, which must be able to adapt to a dynamic environment.

This brings new challenges, in particular maintaining Context synchronization between end-points. Indeed, Context synchronization is a prerequisite for ensuring proper communication between two (or more) SCHC end-points.

This document provides a starting point for discussion on how to manage the various steps in a dynamic Context’s lifecycle:

* how a Context is provisionned,

* how to make a Context evolve over time,

* how to ensure that a Context keeps synchronized among all end-points,

* how to detect and correct possible errors,

* how to manage a context’s end-of-life.

2. Requirements Notation

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here. and indicate requirement levels for compliant SCHC implementations.

3. Terminology

This section defines terminology and abbreviations used in this document. It extends the terminology of [RFC8724].

Rule : A description of the header fields to be compressed/ decompressed or fragmented/reassembled, and the operations to be performed on the fields.

Context / Set of Rules : A context, also known as Set of Rules, contains one or more Rules of various types such as compression, no compression, fragmentation, and management.

Profile: A Profile indicates a particular set of parameters. Both ends of a SCHC communication must be provisioned with the same Profile information and with the same Context before the communication starts, so that there is no ambiguity in how they expect to communicate. Profile parameters are for example the RuleID numbering scheme (fixed-size or variable-size RuleIDs), the fragmentation-related settings, the list of protocols to be processed within the present SCHC Session, and the associated data model(s).

Instance: The SCHC Instance is defined by a Profile and Context pair, selected and parameterized for a given communication. In particular, the Context includes customized Target Values needed to compress/ decompress headers, or to fragment/reassemble a packet.

Instance Manager: The Instance Manager is in charge of creating a SCHC Instance, i.e. loading a Profile and a Context with the appropriate initial parameters/values for a given communication.

Session: The SCHC Session is an active relationship between hosts sharing the same Instance. The session between two end-points whose Instances are static is implicit. Otherwise, a Session is a dynamic space with a beginning and an end.

Context Manager: The Context Manager is in charge of managing the Context within a SCHC Instance, in particular handling Rule modifications during a SCHC Session.

Parser : A parser analyzes and interprets packets. Its main role is to break them down into understandable and structured elements such as fields, allowing for the processing or extraction of relevant information. A given protocol can be broken down in different manners depending on the Parser used, and/or the underlying data model, potentially leading to different SCHC processing (different Rules).

Data Model : A Data Model allows a formal description of the format of packets to be processed with SCHC, as well as a formal description of Rules. Relying on formal models facilitates Context provisioning and interoperability between SCHC nodes.

4. SCHC Architecture Reminder

TODO:

* Basic SCHC operation

* SCHC Data Plane and Control Plane

5. SCHC Context Lifecycle Overview

The SCHC Context is the foundation on which all SCHC operations are based. In this document, we describe the different lifecycle steps of a Context, in a static and dynamic environment, and we aim to identify the key elements for the efficient operation of SCHC, in particular to ensure that the Context remains synchronized between end-points. We study the interactions of the Context with related entities (Profile, Data Models) and how the Context Manager enables updates so the Context remains adapted to the target communication.

5.1. Relationship between Context, Profile and Data Model

In this section, we take a closer look at the notions of Profile, Context and Data Model, as well as the relationships between these entities.

A SCHC Profile defines the features, scope, and parameters, of the SCHC operation for a communication between two (or more) end-points. The parameters that are present in the SCHC Profile are for example the RuleID numbering scheme (fixed-size or variable-size RuleIDs), the fragmentation-related settings, the list of protocols to be processed within the present SCHC Session (scope), and the associated Header Data Model(s). The SCHC Profile then serves as the basis for setting up an adequate Context, i.e., for selecting and adapting relevant Rules for the communication.

The Context contains the Rules that define SCHC operation for a given communication. These Rules can be of different natures such as compression, no compression, fragmentation, and management. Compression Rules all have the same format: a Rule is a list of Field Descriptors
composed of a Field Identifier (FID), a Field Length (FL), a Field Position (FP), a Direction Indicator (DI), a Target Value (TV), a Matching Operator (MO), and a Compression/Decompression Action (CDA). However, the content of compression Rules is specific to the protocols to which compression applies. Prior to the SCHC operation, a protocol Parser identifies the fields encountered in the headers to be compressed and labels them with a Field ID. The way the Parser analyzes header fields (e.g. semantic or syntactic) depends on the

Data Model(s) considered in the Profile and it results in a specific list of FieldsIDs. Within a Rule, Field Descriptors are listed in the order in which fields are returned by the Parser, and indicate their length and position, the functions to be applied to these fields for compression and decompression, etc.

Data Models are an important part of the SCHC ecosystem, to ensure interoperability between SCHC end-points, in particular the compatibility of Parsing at both ends of the communication. Data Models serve as a common reference to indicate how the header of a given protocol should be represented, and consequently the structure of the associated Rules.
In addition, [RFC9363] defines a Data Model for SCHC Rules. It allows to check the validity of the format of Rules in a Context. The name and version of the Data Models used are provided in the Profile.

Figure 1 shows the relationship between Data Models, Context, Profile, and how they feed the Parser and the SCHC Engine to achieve the Compression of incoming Packets. The targeted protocol stack is IP/UDP/CoAP, we assume that the header structure of each protocol is described in separate Data Models. This modularity allows to select the desired Data Model for each protocol, for example a syntactic or a semantic model for CoAP representation.

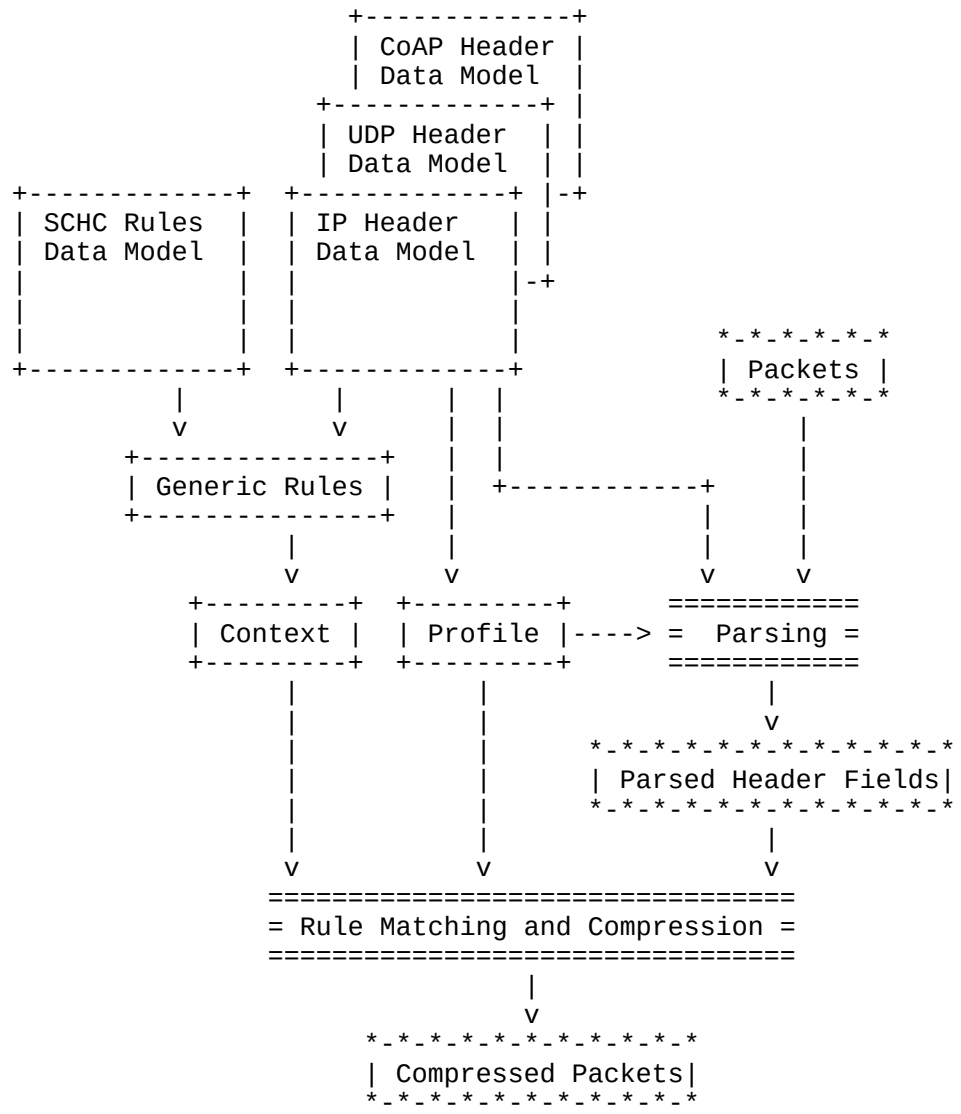

Figure 1: Profile and Context Overview

5.2. Instance Initialization and Context Provisionning

This section explains how a SCHC Instance is initialized before a communication is started.

In the context of LPWANs, the expectation is that SCHC Rules are associated with a physical device that is deployed in a network, the traffic flows to be processed are known in advance. Pre-configured Profile and Context are loaded on both end-points (for example statically on the Device side and dynamically on the SCHC gateway side), which contain the same parameters.

In an unconstrained environment, Profile and Context may be loaded either statically or dynamically, e.g. from an online library where generic or specific Profiles and Contexts are published to a reachable repository, and uniquely identified using Uniform Resource Names and versionned. Generic Profiles and Contexts must be adapted to the communication, in particular by customizing Target Values.

According to the application/communication needs, the Instance Manager of the end-point initiating the communication may select a corresponding Profile and Context. They are obtained either locally or online, or they can retrieve the Profile and Context directly from the partner end-point. An end-point can advertise the support of SCHC, as well as the Profile(s) and Context(s) it owns, and/or intends to use.

A three-way handshake is performed, similarly to the TCP handshake, in order to synchronize the Profile and Context information. Figure 2 shows the generic steps of this handshake. The first step consists of one end-point sending an advertisement message with its SCHC preferences or capabilities. The other end-point may reply with an acknowledgement and additional parameters so that the interlocutor can fine-tune his Context. Finally, the end-point that initiated the handshake concludes the transaction with an acknowledgement, then the SCHC operation can start. The handshake may include additional steps corresponding to the loading of a Context on a given end-point from a remote location,
for example from an external repository or directly from the other end-point. This loading procedure can take place before or during the advertisement phase and may happen twice, once for each end-point.

In Figure 2, a Profile and a Context have already been loaded on end-point A.

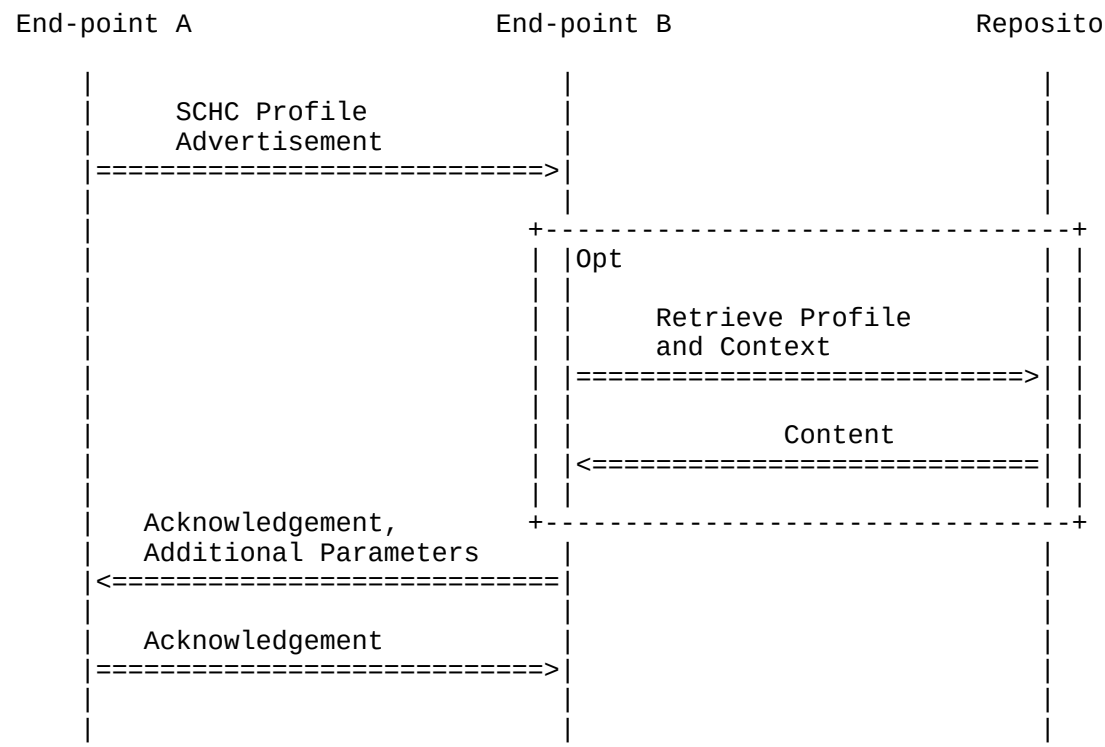

Figure 2: Three-way handshake for Profile and Context initialization

Once the end-points have agreed on the Profile, Context and parameters, an identical SCHC Instance exists on both sides. Since several SCHC Instances can co-exist within the same end-point, each instance has an Instance ID.

In the case of a static Instance or when an Instance is resumed, the Instance ID is sent in theadverstisement. This allows you to check the validity of an instance without having to send all the Profile parameters.

An empty advertisement may also be sent in order to ask for the capabilities of the second end-point. It responds by sending the parameters of its preferred Profile and the associated Context parameters. Finally, the first end-point concludes the handshake by sending an Acknowledgement, and additional parameters if necessary.

5.3. Normal SCHC Operation

This section describes the normal operation of SCHC when no rule modification is needed.

At the sender side, the Parser analyzes packets to be sent and creates a list of fields that is processed by the SCHC Engine, i.e. the Context is browsed to find matching Rules. When several Rules match the field list, implementation must choose one (this is not specified in the standard). The selected matching Rule is used to compress and/or fragment the packet, the rule identifier (RuleID) is transmitted together with the compression residue or fragments to the recipient. When no Rule matches the field list, the No Compression Rule is used (the corresponding RuleID is added in front of the original packet).

Based on the RuleID and the residue/fragments received, the recevier uses the SCHC Decompression or Reassembly functions to rebuild the original packet and forwards it in its uncompressed form to the Application or to the next layer or next hop.

5.4. Context modification

This section explains how a Context can be modified to fit the traffic all along the communication process.

A Context should be dynamically configurable using a network management protocols in order to adapt to in order to adapt to changing traffic patterns and characteristics. Netconf, Restconf and Coreconf are candidate management protocols, suited for specific use cases.

Common operations on a SCHC Context are:

* Rule addition
* Rule modification
* Rule deletion

The Context Manager is also responsible for:

* Verifying and respecting the rights specified in the related ACLs
* Checking for compliance with the Data Model
* Handling errors
  - RuleID not found
  - Error during decompression (examples?)
  - No valid Instance

- Detection of Context desynchronization and recovery

- ...

Management traffic can also be compressed with SCHC using OAM Rules that are part of the Context (see section Management Rules).

5.5. SCHC Session end

SCHC Instance (Profile and Context) end-of-life

The end of a SCHC Session can be explicit (by sending a SCHC control message) or implicit (reset message received from another layer, timeout).

All data associated with this instance should be deleted on all end-points.

5.6. Management Rules

Signaling (control), configuration, and error messages are exchanged between SCHC end-points in order to allow proper SCHC operation. These messages may use CORECONF which is based on YANG data models, uses CoAP for message exchange and CBOR encoding.

Specific OAM Sets of Rules should be locally installed on SCHC end-points or available on an online library, to define how to compress the management traffic (e.g. CORECONF messages).

6. Use cases

6.1. Static Instances

6.2. Dynamic Instance creation and Session management using the Context Manager

7. IANA Considerations

8. Security Considerations

9. Acknowledgements

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", RFC 8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC8724] Minaburo, A. and L. Toutain, "SCHC: Generic Framework for Static Context Header Compression and Fragmentation", RFC 8724, April 2020, <https://www.rfc-editor.org/rfc/rfc8724>.

[RFC9363] Petrov, I., "LPWAN Static Context Header Compression (SCHC) over LoRaWAN", RFC 9363, March 2023, <https://www.rfc-editor.org/rfc/rfc9363>.

10.2. Informative References

[REST] Toutain, L., "CORECONF Rule management for SCHC", 2025.

Appendix A. Some Note

(This appendix to be deleted by the RFC editor.)

Author’s Address

Marion Dumay
Orange Research
CS 10098, 22 chemin du Vieux Ch棚ne
38243 Meylan Cedex
France
Phone: +33-78-612-4892
Email: marion.dumay@orange.com