Network Working Group   A. Minaburo
Internet-Draft   Acklio
Intended status: Informational   L. Toutain
Expires: September 12, 2019   Institut MINES TELECOM ; IMT Atlantique
                                                        March 11, 2019

Data Model for Static Context Header Compression (SCHC) draft-toutain-lpwan-schc-yang-data-model-00

Abstract

This document describes a YANG data model for the SCHC (Static Context Header Compression). A generic module is defined, that can be applied for any headers and also a specific model for the IPv6 UDP protocol stack is also proposed. Note that this draft is a first attempt to define a YANG data module for SCHC, more work is needed to use all the YANG facilities.

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 September 12, 2019.

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

1. Introduction

SCHC [I-D.ietf-lpwan-ipv6-static-context-hc] defines a compression technique for LPWAN networks based on static context. The context contains a list of rules (cf. Figure 1). Each rule contains itself a list of field descriptions composed of a field identifier (FID), a field length (FID), a field position (FP), a field direction (DI), a target value (TV), a matching operator (MO) and a Compression/ Decompression Action (CDA).

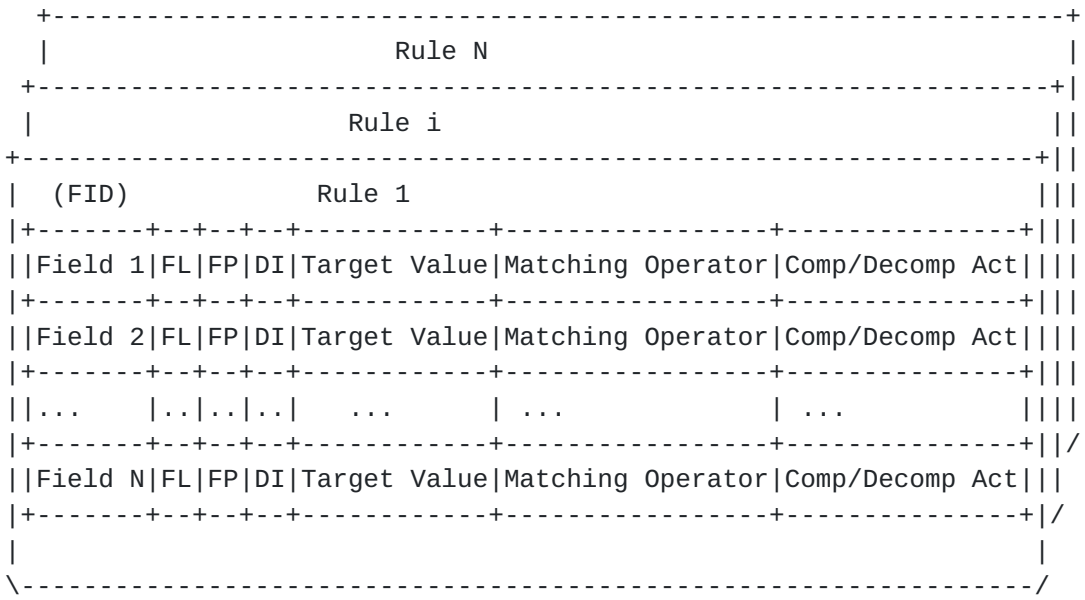

Figure 1: Compression Decompression Context

The goal of this document is to provide an YANG data model to represent SCHC Compression and Fragmentation rules, to allow management over a LPWAN network. The main constraints are:

- since the device may be managed through the LPWAN network, management messages must be compact. COREconf offers a representation based on CBOR.

- this data model can be extended with new values, such as new field id, new MO or CDA.

2. YANG types

2.1. Field Identifier

The field identifier is used to identify a specific field. It is viewed as an uint32.

2.2. Target Value field

A value may be associated for each field in a rule. The value's type depends on the field. It can be an integer, a prefix, a string, or any other type carried by the field. The LPWA-types regroups all the possibles values. Figure 2 gives its definition.

```
typedef lpwan-types {
  type union {
    type uint8;
    type uint16;
    type uint32;
    type uint64;
    type inet:ipv6-prefix;
    type string;
   }
}
```

Figure 2: Value types

Note that as defined in [I-D.ietf-lpwan-ipv6-static-context-hc], Dev and App Prefixes can be of type inet:ipv6-prefix-type, but this type derives from ASCII characters, a binary representation such as uint64 will be more compact.

2.3. Matching Operators

A matching operator is used to check the field value stored in the rule against the value contained in the header field. If there is no matching the rule is not selected. Two instances of matching operator are defined to allow the rule selection from informations contained either in the compressed header or the uncompressed header. The SCHC document [I-D.ietf-lpwan-ipv6-static-context-hc] defines four operators:

- equal: The rule's value must be equal to the packet header value for a specific field.

- ignore: There is no check for this field.

- MSB(x): This operator compare the most significant bits. The operator takes one argument representing the length of least significant bit part, which will be ignored during the matching but sent if the rule matches.

- match-mapping: From the list of values of the Target Value, This operator will match if one of those values is equal to the field value and will send the index of the list representing this value.

```
/*********************************/
/* Matching operator type         */
/*********************************/
typedef matching-operator-type {
    type enumeration {
        enum equal;
        enum ignore;
        enum msb;
        enum match-mapping;
    }
}
```

Figure 3: Matching operators

Figure 3 represents the Matching Operator type definition.

2.4. Compression Decompression Actions

The SCHC document [I-D.ietf-lpwan-ipv6-static-context-hc] defines some compression decompression actions (CDA). The CDA tells how to compress and decompress the field. They are defined in Figure 4. they are coded the same way as MO.

```
        /*********************************************/
        /* Compression-Decompression action type       */
        /*********************************************/
        typedef compression-decompression-action-type {
                        type enumeration {
                                enum not-sent;
                                enum value-sent;
                                enum lsb;
                                enum mapping-sent;
                                enum compute-length;
                                enum compute-checksum;
                                enum esiid-did;
                                enum laiid-did;
                        }
        }
```

Figure 4: Action functions

3. Generic rule definition

Each rule's row is defined by several leaves, composed of:

- a field key which will be used as a key,
- a field name that can be used for debugging purpose,
- a field length that containing the length of the field,
- a field position that gives the number of instances,
- a field direction indicates the packet direction,
- a field target value containing the value that will be compared,
- a matching operators for rule selection
- an compression/decompression action to compress/decompress the field.

Figure 5 defines the format.

```
      grouping rule-entry {
              leaf field-id {
                      type int32;
                      description "Field ID unique value representing the
Field";
              }
```

```
                leaf field-length {
                        type uint8;
                        description "size in bits of the field";
                }

                leaf field-position {
                        type uint8;
                        description "For repeated fields, we need to be able to
                                     distinguish between successive
occurences";
                }

                leaf direction {
                     type direction-type;
                }
                list target-values {
                     key tv-key;
                     leaf tv-key {
                        type int8;
                     }
                     leaf target-value {
                        type lpwan-types;
                     }
                     description "Target Values can be a list of value, for
                                  match-mapping. For other MO, only one entry
is specified";
                }
                leaf matching-operator {
                     type matching-operator-type;
                }

                leaf matching-operator-parameter {
                     type lpwan-types;
                     description "If the matching operator requires a parameter
                                  (for example lsb or msb), the value is
provided here.";
                }

                leaf compression-decompression-action {
                     type compression-decompression-action-type;
                }

                leaf compression-decompression-action-parameter {
                     type lpwan-types;
                     description "If the matching operator requires a parameter
                                  (for example lsb or msb), the value is
provided here.";
                }
```

```
        }
```

Figure 5: Action functions

4. YANG static context model

This lead to the generic rule definition, represented Figure 7. It defines a set of rules.

```
  grouping compression-rule {

    leaf rule-id {
            type uint8;
            description "The number of the context rule that should be
applied.";
    }
    leaf rule-id-length {
            type uint8;
    }

    list rule-fields {
      key "field-id field-position direction";
      uses rule-entry;
    }

  }
```

Figure 6: YANG definition of the generic module

```
 module: ietf-lpwan-schc-rule
   +--rw rule-id?          uint8
   +--rw rule-id-length?   uint8
   +--rw rule-fields* [field-id field-position direction]
      +--rw field-id                                      int32
      +--rw field-length?                                 uint8
      +--rw field-position                                uint8
      +--rw direction                                     direction-type
      +--rw target-values* [tv-key]
      |  +--rw tv-key          int8
      |  +--rw target-value?   lpwan-types
      +--rw matching-operator?                            m.-o.-type
      +--rw matching-operator-parameter?                  lpwan-types
      +--rw compression-decompression-action?             c.-d.-a.-type
      +--rw compression-decompression-action-parameter?   lpwan-types
```

Figure 7: Generic module tree

The YANG tree is given Figure 7.

```
SID        Assigned to
---------  -------------------------------------------------
60000      node /rule-fields
60001      node /rule-fields/compression-decompression-action
60002      node /rule-fields/compression-decompression-action-parameter
60003      node /rule-fields/direction
60004      node /rule-fields/field-id
60005      node /rule-fields/field-length
60006      node /rule-fields/field-position
60007      node /rule-fields/matching-operator
60008      node /rule-fields/matching-operator-parameter
60009      node /rule-fields/target-values
60010      node /rule-fields/target-values/target-value
60011      node /rule-fields/target-values/tv-key
60012      node /rule-id
60013      node /rule-id-length


File ietf-lpwan-schc-rule@2016-10-31.sid created
Number of SIDs available : 100
Number of SIDs assigned : 14
```

Figure 8: Example of SID allocation

Figure 8 gives a simple allocation for SID value. SID values from 100 to 113 are used for /generic-rules/context-rules/rule-fields/field-compression-decompression-action. SID value from 1009 to 1012 are used in /generic-rules/context-rules/rule-fields/field-matching-operator.

5. Acknowledgement

The authors would like to thank Michel Veillette, Alexander Pelov, Antoni Markovski for their help on COMI/CoOL and YANG.

6. Normative References

[I-D.ietf-core-comi]
Veillette, M., Stok, P., Pelov, A., and A. Bierman, "CoAP Management Interface", draft-ietf-core-comi-04 (work in progress), November 2018.

[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.

Authors' Addresses

Ana Minaburo
Acklio
1137A Avenue des Champs Blancs
35510 Cesson-Sevigne Cedex
France

Email: ana@ackl.io

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

Email: Laurent.Toutain@imt-atlantique.fr