6TiSCH
Internet-Draft
Intended status: Informational
Expires: December 29, 2016

Q. Wang, Ed.
Univ. of Sci. and Tech. Beijing
X. Vilajosana
Universitat Oberta de Catalunya

June 27, 2016

6top Protocol (6P)
draft-ietf-6tisch-6top-protocol-01

Abstract

This document defines the 6top Protocol (6P), which enables distributed scheduling in 6TiSCH networks. 6P allows neighbor nodes in a 6TiSCH network to add/delete TSCH cells to one another. 6P is part of the 6TiSCH Operation Sublayer (6top), the next higher layer of the IEEE802.15.4 TSCH medium access control layer. The 6top Scheduling Function (SF) decides when to add/delete cells, and triggers 6P transactions. Several SFs can be defined, each identified by a different 6top Scheduling Function Identifier (SFID). This document lists the requirements for an SF, but leaves the definition of the SF out of scope. Different SFs are expected to be defined in future companion specifications.

Requirements Language

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 RFC 2119 [RFC2119].

Status of This Memo

Table of Contents

1. TEMPORARY EDITORIAL NOTES

This document is an Internet Draft, so work-in-progress by nature. It contains the following work-in-progress elements:

- "TODO" statements are elements which have not yet been written by the authors for some reason (lack of time, ongoing discussions with no clear consensus yet, etc). The statement does indicate that the text will be written.
- "TEMPORARY" Appendices are there to capture current ongoing discussions or the changelog of the document. These appendices will be removed in the final text.
- "IANA_" identifiers are placeholders for numbers assigned by IANA. These placeholders are to be replaced by the actual values they represent after their assignment by IANA.
- This section will be removed in the final text.

2. Introduction

All communication in a 6TiSCH network is orchestrated by a schedule [RFC7554]. This specification defines the 6top Protocol (6P), part of the 6TiSCH Operation Sublayer (6top) sublayer. 6P allow a node to communicate with a neighbor to add/delete a TSCH cell to one another. 6P hence enables distributed scheduling in a 6TiSCH network.

```
          (A)
          / \
         /   \
      (B)-----(C)
       |       |
       |       |
      (D)     (E)
```

Figure 1: A simple 6TiSCH network.

For example, node C in Figure 1 monitors the communication cells to node A it has in its schedule.

- o If node C determines the number of frames it is sending to A per unit of time is larger than the capacity offered by the TSCH cells it has scheduled to A, it triggers a 6P transaction with node A to add one or more cells to A in the TSCH schedule.
- o If the traffic is lower than the capacity, node C triggers a 6P transaction with node A to delete one or more cells to A in the TSCH schedule.
- o Node C might also monitor statistics to determine whether collisions are happening on a particular cell to node A. If this feature is enabled, node C communicates with node A to add a new cell and delete the cell which suffered from collisions. This conceptually results in "relocating" the cell which suffered from collisions to a different slotOffset/channelOffset location in the TSCH schedule. The mechanism to handle cell relocation is out of the scope of this document and might be defined in a future document.

This results in distributed schedule management in a 6TiSCH network.

The 6top Scheduling Function (SF) defines when to add/delete a cell to a neighbor. The SF functions as a (required) add-on to 6P. Different applications require different SFs, so the SF is left out of scope of this document. Different SFs are expected to be defined in future companion specifications. A node MAY implement multiple SFs and run them at the same time. The SFID field contained in all 6P messages allows a node to switch between SFs on a per-transaction basis.

Section 3 describes the 6TiSCH Operation Sublayer (6top). Section 4 defines the 6top Protocol (6P). Section 5 provides guidelines on how to design an SF.

3. 6TiSCH Operation Sublayer (6top)

As depicted in Figure 2, the 6TiSCH Operation Sublayer (6top) is the next higher layer to the IEEE802.15.4 TSCH medium access control layer [IEEE802154-2015].

```
                            .
           |                .                       |
           |          next higher layer             |
           +----------------------------------------+
           |                6top                    |
           +----------------------------------------+
           |          IEEE802.15.4 TSCH             |
           |                .                       |
                            .
```

Figure 2: The 6top sublayer in the protocol stack.

The roles of the 6top sublayer are:

- o Implement and terminate the 6top Protocol (6P), which allows neighbor nodes to communicate to add/delete cells to one another.
- o Run one or more 6top Scheduling Function (SF), which define the algorithm to decide when to add/delete cells.

3.1. Hard/Soft Cells

6top qualifies each cell in the schedule as either "hard" or "soft":

- o a Soft Cell can be read, added, deleted or updated by 6top.
- o a Hard Cell is read-only for 6top.

In the context of this specification, all the cells used by 6top are Soft Cells. Hard cells can be used for example when "hard-coding" a scheduling. This is done, for example, in the Minimal 6TiSCH Configuration [I-D.ietf-6tisch-minimal].

3.2. Using 6top with the Minimal 6TiSCH Configuration

6top MAY be used alongside the Minimal 6TiSCH Configuration [I-D.ietf-6tisch-minimal]. In this case, it is RECOMMENDED to use 2 slotframes, as depicted in Figure 3:

- o Slotframe 0 is used for traffic defined in the Minimal 6TiSCH Configuration. In Figure 3, this slotframe is 5 slots long, but it can be of any length.
- o Slotframe 1 is used by 6top to allocate cells from. In Figure 3, this slotframe is 10 slots long, but it can be of any length.

Slotframe 0 SHOULD be of higher priority than Slotframe 1. 6top MAY support further slotframes; how to use more slotframes is out of the scope for this document.

```
                 | 0    1    2    3    4 | 0    1    2    3    4  |
                 +-----------------------+------------------------+
     Slotframe 0 |    |    |    |    |    |    |    |    |    |    |
    5 slots long | EB |    |    |    |    | EB |    |    |    |    |
   high priority |    |    |    |    |    |    |    |    |    |    |
                 +------------------------------------------------+

                 | 0    1    2    3    4    5    6    7    8    9 |
                 +------------------------------------------------+
     Slotframe 1 |    |    |    |    |    |    |    |    |    |    |
   10 slots long |    |A->B|    |    |    |    |    |    |B->A|    |
    low priority |    |    |    |    |    |    |    |    |    |    |
                 +------------------------------------------------+
```

Figure 3: 2-slotframe structure when using 6top alongside the Minimal 6TiSCH Configuration.

4. 6top Protocol (6P)

The 6top Protocol (6P) allows two neighbor nodes to communicate to add/delete cells to their TSCH schedule. Conceptually, two neighbor nodes "negotiate" the location of the cell(s) to add/delete.

4.1. 6top Transaction

We call "6top Transaction" a complete negotiation between two neighbor nodes. A transaction starts when a node wishes to add/ delete one or more cells to one of its neighbors; it ends when the cell(s) have been added removed from the schedule of both neighbor, or when the transaction has failed.

A transaction can consist of 2 or 3 steps. It is the SF which determines whether to use 2-step or 3-step transactions. An SF MAY use both 2-step and 3-step transactions.

We reuse the topology in Figure 1 to illustrate 2-step and 3-step transactions.

4.1.1. 2-step 6top Transaction

6P supports both 2- and 3-step transactions; the SF determinisms which to use. Without loss of generality, this section illustrates 2-step transaction through an example.

Figure 4 is a sequence diagram to help understand the core principle of 6P (several elements are left out to simplify understanding). We assume the SF running on node A determines 2 extra cells need to be scheduled to node B. In this example, node A proposes the cells to use.

```
+----------+                           +----------+
|  Node A  |                           |  Node B  |
+----+-----+                           +-----+----+
     |                                       |
     | 6P ADD Request                        |
     |   NumCells     = 2                    |
     |   CellList     = [(1,2),(2,2),(3,5)]  |
     |-------------------------------------->|
     |                                       |
     | 6P Response                           |
     |   Return Code  = IANA_6TOP_RC_SUCCESS |
     |   CellList     = [(2,2),(3,5)]        |
     |<--------------------------------------|
     |                                       |
```

Figure 4: A 2-step 6P transaction.

In this example, the 2-step transaction occurs as follows:

1. The SF running on node A selects 3 candidate cells.
2. Node A sends a 6P ADD Request to node B, indicating it wishes to add 2 cells (the "NumCells" value), and specifying the list of 3 candidate (the "CellList" value). Each cell in the CellList is a (slotOffset,channelOffset) tuple.
3. The SF running on node B selects 2 of the 3 cells in the CellList of the 6P ADD Request. Node B sends back a 6P Response to node A, indicating the cells it selected.
4. The result of this 6P transaction is that 2 cells from A to B have been added to the TSCH schedule of both nodes A and B.

4.1.2. 3-step 6top Transaction

6P supports both 2- and 3-step transactions; the SF determinisms which to use. Without loss of generality, this section illustrates 3-step transaction through an example.

Figure 5 is a sequence diagram to help understand the core principle of 6P (several elements are left out to simplify understanding). We assume the SF running on node A determines 2 extra cells need to be scheduled to node B. In this example, node B proposes the cells to use.

```
      +----------+                       +----------+
      |  Node A  |                       |  Node B  |
      +----+-----+                       +-----+----+
           |                                   |
           | 6P ADD Request                    |
           |   NumCells     = 2                |
           |   CellList     = []               |
           |---------------------------------->|
           |                                   |
           | 6P Response                       |
           |   Return Code  = IANA_6TOP_RC_SUCCESS |
           |   CellList     = [(1,2),(2,2),(3,5)]  |
           |<----------------------------------|
           |                                   |
           | 6P Confirmation                   |
           |   Return Code  = IANA_6TOP_RC_SUCCESS |
           |   CellList     = [(2,2),(3,5)]    |
           |---------------------------------->|
           |                                   |
```

Figure 5: A 3-step 6P transaction.

In this example, the 3-step transaction occurs as follows:

1. The SF running on node A determines 2 extra cells need to be scheduled to node B, but does not select candidate cells.
2. Node A sends a 6P ADD Request to node B, indicating it wishes to add 2 cells (the "NumCells" value), with en empty "CellList".
3. The SF running on node B selects 3 candidate cells. Node B sends back a 6P Response to node A, indicating the 3 cells it selected.
4. The SF running on node A selects 2 cells. Node A sends back a 6P Confirmation to node B, indicating the cells it selected.
5. The result of this 6P transaction is that 2 cells from A to B have been added to the TSCH schedule of both nodes A and B.

4.2. Message Format

4.2.1. 6top Information Element

6P messages are carried as payload of IEEE802.15.4 Information Elements (IE) [IEEE802154-2015]. 6p messages travel over a single hop.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Payload IE Length   |GroupID|T|    Sub-ID     |6top IE Content
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Payload Termination IE        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The 6top IE is an IETF IE with GroupID IANA_IETF_IE_GROUP_ID. The Sub-ID used by the 6top IE is IANA_6TOP_SUBIE_ID. The length of the 6top IE content is variable. The content of the 6top IE is specified in Section 4.2. The Payload Termination IE is defined by the IEEE802.15.4 standard [IEEE802154-2015]. TODO: IETF IE specified in Appendix A for now, but to be specified in a separate draft in the future.

4.2.2. General Message Format

In all 6P messages, the 6top IE content has the following format:

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version| Code  |     SFID      |    SeqNum     | Other Fields...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Version (6P Version): The version of the 6P protocol. Only version IANA_6TOP_6P_VERSION is defined in this document. Future specifications MIGHT define further version of the 6P protocol.
Code: Command to carry out or response code. The list of command identifiers and return codes is defined only for version IANA_6TOP_6P_VERSION in this document.
SFID (6top Scheduling Function Identifier): The identifier of the SF to use to handle this message. The SFID is defined in Section 5.1.
SeqNum: An identifier of the packet, used to match request and response. The value of SeqNum MUST increment by exactly one at each new 6P request issued to the same neighbor.
Other Fields: The list of other fields depends on the value of the code field, as detailed below.

4.2.3. 6P Command Identifiers

Figure 6 lists the 6P command identifiers.

Value	Command ID	Description
IANA_6TOP_CMD_ADD	CMD_ADD	add one or more cells
IANA_6TOP_CMD_DELETE	CMD_DELETE	delete one or more cells
IANA_6TOP_CMD_COUNT	CMD_COUNT	count scheduled cells
IANA_6TOP_CMD_LIST	CMD_LIST	list the scheduled cells
IANA_6TOP_CMD_CLEAR	CMD_CLEAR	clear all cells
TODO-0xf	reserved	

Figure 6: 6P Command Identifiers

4.2.4. 6P Return Codes

Figure 7 lists the 6P Return Codes and their meaning.

Value	Return Code	Description
IANA_6TOP_RC_SUCCESS	RC_SUCCESS	operation succeeded
IANA_6TOP_RC_VER_ERR	RC_VER_ERR	unsupported 6P version
IANA_6TOP_RC_SFID_ERR	RC_SFID_ERR	unsupported SFID
IANA_6TOP_RC_BUSY	RC_BUSY	handling previous request
IANA_6TOP_RC_RESET	RC_RESET	abort 6P transaction
IANA_6TOP_RC_ERR	RC_ERR	operation failed
TODO-0xf	reserved	

Figure 7: 6P Return Codes

4.2.5. 6P Cell Format

The 6P Cell is an element which is present in several messages. It is a 4-byte field, its RECOMMENDED format is:

```
                      1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |          slotOffset           |         channelOffset         |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

slotOffset: The slot offset of the cell.
channelOffset: The channel offset of the cell.

The CellList is an opaque set of bytes, sent unmodified to the SF. The SF MAY redefine the format of the CellList field.

4.2.6. 6P ADD Request Format

```
                      1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |Version|  Code |     SFID      |     SeqNum    |   NumCells    |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |           Metadata            | CellList ...
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Version: Set to IANA_6TOP_6P_VERSION.
Code: Set to IANA_6TOP_CMD_ADD for a 6P ADD Request.
SFID: Identifier of the SF to be used by the receiver to handle the message.
SeqNum: Packet identifier to match 6P Request and 6P Response.
NumCells: The number of additional TX cells the sender wants to schedule to the receiver.
Metadata: Metadata used as extra signaling to the SF. One example use can be to specify which slotframe to schedule the cells to. The contents of the Metadata field is an opaque set of bytes, and passed unmodified to the SF. The meaning of this field depends on the SF, and is hence out of scope of this document.
CellList: A list of 0, 1 or multiple 6P Cells. The RECOMMENDED format of each 6P Cell is defined in Section 4.2.5. The CellList is an opaque set of bytes, sent unmodified to the SF. The SF MAY redefine the format of the CellList field.

4.2.7. 6P DELETE Request Format

The 6P DELETE Request has the exact same format as the 6P ADD Request, except for the code which is set to IANA_6TOP_CMD_DELETE.

4.2.8. 6P COUNT Request Format

```
                     1                   2
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  Code |    SFID       |     SeqNum    |   Metadata
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    Metadata    |
+-+-+-+-+-+-+-+-+
```

Version: Set to IANA_6TOP_6P_VERSION.
Code: Set to IANA_6TOP_CMD_COUNT for a 6P COUNT Request.
SFID: Identifier of the SF to be used by the receiver to handle the message.
SeqNum: Packet identifier to match request and response.
Metadata: Metadata used as extra signaling to the SF. One example use can be to specify which slotframe to schedule the cells to. The contents of the Metadata field is an opaque set of bytes, and passed unmodified to the SF. The meaning of this field depends on the SF, and is hence out of scope of this document.

4.2.9. 6P LIST Request Format

The 6P LIST Request has the exact same format as the 6P COUNT Request, except for the code which is set to IANA_6TOP_CMD_LIST.

4.2.10. 6P CLEAR Request Format

The 6P CLEAR Request has the exact same format as the 6P COUNT Request, except for the code which is set to IANA_6TOP_CMD_CLEAR.

4.2.11. 6P Response Format

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  Code |    SFID       |     SeqNum    | Other Fields...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Version: Set to IANA_6TOP_6P_VERSION.
SFID: Identifier of the SF to be used by the receiver to handle the message. The response MUST contain the same SFID value as the value in the SFID field of the 6P Request is responds to.
Code: One of the 6P Return Codes listed in Section 4.2.4.
SeqNum: Packet identifier to match request and response. The response MUST contain the same SeqNum value as the value in the SeqNum field of the 6P Request is responds to.
Other Fields: The fields depends on what command the request is for:

Response to an ADD, DELETE or LIST command: A list of 0, 1 or multiple 6P Cells. The format of a 6P Cell is defined in Section 4.2.5.

Response to COUNT command: The number of cells scheduled from the requesting node to the receiver node by the 6P protocol, encoded as a 2-octet unsigned integer.

Response to CLEAR command: No other fields are present in the response.

4.2.12. 6P Confirmation Format

A 6P Confirmation is only used in a 3-step transaction, as the third step. A 6P Confirmation Message has the exact same format as a 6P Response Message. It is only the fact that it appears as the third step in a 3-step transaction that distinguishes it from a 6P Response. In particular, the same Return Codes are used in both 6P Response and 6P Confirmation messages.

4.3. Protocol Behavior

For illustration, we assume we use the topology in Figure 1, and that node A negotiates to add/delete cells to node B.

4.3.1. Version Checking

All messages contain a Version field. If multiple Versions of the 6P protocol have been defined (in future specifications for Version values different than IANA_6TOP_6P_VERSION), a node MAY implement multiple protocol versions at the same time. When receiving a 6P message with a Version number it does not implement, a node MUST reply with a 6P Response and a return code of IANA_6TOP_RC_VER_ERR. The Version field in the 6P Response MUST be the same as the Version field in the corresponding 6P Request.

4.3.2. SFID Checking

All messages contain a SFID field. If multiple SFs has been defined, a node MAY support multiple SFs at the same time. When receiving a 6P message with an unsupported SFID, a node MUST reply with a 6P Response and a return code of IANA_6TOP_RC_SFID_ERR. The Version field in the 6P Response MUST be the same as the Version field in the corresponding 6P Request.

4.3.3. Concurrent 6P Transactions

Only a single 6P Transaction between two neighbors, in a given direction, can take place at the same time. That is, a node MUST NOT issue a new 6P Request to a given neighbor before having received the

6P Response for a previous request to that neighbor. The only exception to this rule is when the previous 6P Transaction has timed out. If a node receives a 6P Request from a given neighbor before having sent the 6P Response to the previous 6P Request from that neighbor, it MUST send back a 6P Response with a return code of IANA_6TOP_RC_ERR.

A node MAY support concurrent 6P Transactions from different neighbors. In this case the cells involved in the ongoing 6P transaction MUST be locked until the transaction finishes. For example, in Figure 1, node C can have a different ongoing 6P Transaction with nodes B and E. In case a node does not have enough resources to handle concurrent 6P Transactions from different neighbors or if the cells requested are locked, it MUST reply to that second request with a 6P Response with return code IANA_6TOP_RC_BUSY. The node receiving IANA_6TOP_RC_BUSY may implement a retry mechanism, as decided by the Scheduling Function.

4.3.4. Timeout

A timeout happens when the node sending the 6P Request has not received the 6P Response. The value of the timeout is coupled with how the cells between the nodes are scheduled. The SF determines the value of the timeout. The value of the timeout is out of scope of this document.

4.3.5. SeqNum Mismatch

When a node receives a 6P Response with SeqNum value different from the SeqNum value in the 6P Request, it MUST drop the packet and consider the 6P Transaction as having failed.

4.3.6. Adding cells

We assume the topology in Figure 1 where the SF on node C decides to add NumCell cells to node A.

Node C’s SF selects NumCandidate>=NumCell cells from its schedule as candidate transmit cells to node A. NumCandidate MUST be larger or equal to NumCell. How many cells it selects (NumCandidate) and how that selection is done is specified in the SF and out of scope of this document. Node C sends a 6P ADD Request to node A which contains the value of NumCells and the NumCandidate cells in the CellList.

Upon receiving the request, node A’s SF verifies which of the cells in the CellList it can add as receive cells from node C in its own schedule. How that selection is done is specified in the SF and out of scope of this document. That verification can succeed (NumCell cells from the CellList can be used), fail (none of the cells from the CellList can be used) or partially succeed (less than NumCell cells from the CellList can be used). In all cases, node A MUST send a 6P Response with return code set to IANA_6TOP_RC_SUCCESS, and which specifies the list of cells that were scheduled as receive cells from C. That can contain 0 elements (when the verification failed), NumCell elements (succeeded) or between 0 and NumCell elements (partially succeeded).

Upon receiving the response, node C adds the cells specified in the CellList as transmit cells to node A.

4.3.7. Aborting a 6P Transaction

In case the receiver of a 6top request fails during a 6P Transaction and is unable to complete it, it SHOULD reply to that request with a 6P Response with return code IANA_6TOP_RC_RESET. Upon receiving this 6top reply, the initiator of the 6P Transaction MUST consider the 6P Transaction as failed.

4.3.8. Deleting cells

The behavior for deleting cells is equivalent to that of adding cells except that:

- o The nodes delete the cells they agree upon rather than adding them.
- o All cells in the CellList MUST be already scheduled between the two nodes.
- o If the CellList in the 6P Request is empty, the SF on the receiving node is free to delete any cell from the sender.
- o The CellList MUST either be equal, contain exactly NumCell cells, or more than NumCell cells. The case where the CellList is not empty but contains less than NumCell cells is not supported.

4.3.9. Handling error responses

A return code with a name starting with "RC_ERR" in Figure 7 indicates an error. When a node receives a 6P Response with such an error, it MUST consider the 6P Transaction failed. In particular, if this was a response to a 6P ADD/DELETE Request, the node MUST NOT add/delete any of the cells involved in this 6P Transaction. Similarly, a node sending a 6P Response with an "RC_ERR" return code MUST NOT add/delete any cells as part of that 6P Transaction. The SF defines what to do after an error has occurred. Defining what to do after an error has occurred is out of scope of this document.

4.4. Security

6P messages are secured through link-layer security. When link-layer security is enabled, the 6P messages MUST be secured. This is possible because 6P messages are carried as Payload IE.

5. Guidelines for 6top Scheduling Functions (SF)

5.1. SF Identifier (SFID)

Each SF has an identifier. The identifier is encoded as a 1-byte field. The identifier space is divided in the following ranges.

Range	Meaning
0x00-0xef	managed
0xf0-0xfe	unmanaged
0xff	reserved

Figure 8: SFID range.

SF identifiers in the managed space MUST be managed by IANA.

5.2. Requirements for an SF

The specification for an SF

- o MUST specify an identifier for that SF.
- o MUST specify the rule for a node to decide when to add/delete one or more cells to a neighbor.
- o MUST specify the rule for a Transaction source to select cells to add to the CellList field in the 6P ADD Request.
- o MUST specify the rule for a Transaction destination to select cells from CellList to add to its schedule.
- o MUST specify a value for the 6P Timeout, or a rule/equation to calculate it.
- o MUST specify a meaning for the "Metadata" field in the 6P ADD Request.
- o MUST specify the behavior of a node when it boots.
- o MUST specify what to do after an error has occurred (either the node sent a 6P Response with an error code, or received one).
- o MUST specify the list of statistics to gather. An example statistic if the number of transmitted frames to each neighbor. In case the SF requires no statistics to be gathered, the specific of the SF MUST explicitly state so.

- o SHOULD clearly state the application domain the SF is created for.
- o SHOULD contain examples which highlight normal and error scenarios.
- o SHOULD contain a list of current implementations, at least during the I-D state of the document, per [RFC6982].
- o SHOULD contain a performance evaluation of the scheme, possibly through references to external documents.
- o MAY redefine the format of the CellList field.

5.3. Recommended Structure of an SF Specification

The following section structure for a SF document is RECOMMENDED:

- o Introduction
- o Scheduling Function Identifier
- o Rules for Adding/Deleting Cells
- o Rules for CellList
- o 6P Timeout Value
- o Meaning of the Metadata Field
- o Node Behavior at Boot
- o 6P Error Handling
- o Examples
- o Implementation Status
- o Security Considerations
- o IANA Considerations

6. Implementation Status

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC6982]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC6982], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

ETSI 6TiSCH #2 plugtests: 6P was one of two protocols addressed during the ETSI 6TiSCH #2 plugtests organized on 2-4 February 2016 in Paris, France. 14 entities participated in this event, verifying the compliance and interoperability of their implementation of 6P. This event happened under NDA, so neither the name of the entities nor the test results are public. This event is, however, a clear indication of the maturity of 6P, and the interest it generates. More information about the event at http://www.etsi.org/news-events/events/1022-6TiSCH-2-plugtests.

OpenWSN: 6P is implemented in the OpenWSN project [OpenWSN] under a BSD open-source license. The authors of this document are collaborating with the OpenWSN community to gather feedback about the status and performance of the protocols described in this document. Results from that discussion will appear in this section in future revision of this specification. More information about this implementation at http://www.openwsn.org/.

Wireshark Dissector: A Wireshark dissector for 6P is implemented under a BSD open-source license. It is not yet merged into the main Wireshark build, but can be downloaded at https://github.com/openwsn-berkeley/dissectors/.

7. Security Considerations

TODO: explicit risks

6P messages are carried inside IEEE802.15.4 Payload Information Elements (IEs). Those Payload IEs are encrypted and authenticated at the link layer through CCM*. 6P benefits from the same level of security as any other Payload IE. The 6P protocol does not define its own security mechanisms. A key management solution is out of scope for this document. The 6P protocol will benefit for the key management solution used in the network.

8. IANA Consideration

TODO: write out this section as soon as the discussion with the IEEE about a possible IETF IE ID has concluded.

- TODO: IANA_IETF_IE_GROUP_ID
- TODO: IANA_6TOP_SUBIE_ID
- TODO: IANA_6TOP_6P_VERSION
- TODO: IANA_6TOP_CMD_ADD
- TODO: IANA_6TOP_CMD_DELETE
- TODO: IANA_6TOP_CMD_LIST
- TODO: IANA_6TOP_CMD_COUNT
- TODO: IANA_6TOP_CMD_CLEAR
- TODO: IANA_6TOP_RC_SUCCESS
- TODO: IANA_6TOP_RC_VER_ERR

- o TODO: IANA_6TOP_RC_SFID_ERR
- o TODO: IANA_6TOP_RC_BUSY
- o TODO: IANA_6TOP_RC_RESET
- o TODO: IANA_6TOP_RC_ERR

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[IEEE802154-2015] IEEE standard for Information Technology, "IEEE Std 802.15.4-2015 - IEEE Standard for Low-Rate Wireless Personal Area Networks (WPANs)", October 2015.

9.2. Informative References

[RFC7554] Watteyne, T., Ed., Palattella, M., and L. Grieco, "Using IEEE 802.15.4e Time-Slotted Channel Hopping (TSCH) in the Internet of Things (IoT): Problem Statement", RFC 7554, DOI 10.17487/RFC7554, May 2015, <http://www.rfc-editor.org/info/rfc7554>.

[RFC6982] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", RFC 6982, DOI 10.17487/RFC6982, July 2013, <http://www.rfc-editor.org/info/rfc6982>.

[I-D.ietf-6tisch-minimal] Vilajosana, X. and K. Pister, "Minimal 6TiSCH Configuration", draft-ietf-6tisch-minimal-15 (work in progress), February 2016.

[I-D.ietf-6tisch-terminology] Palattella, M., Thubert, P., Watteyne, T., and Q. Wang, "Terminology in IPv6 over the TSCH mode of IEEE 802.15.4e", draft-ietf-6tisch-terminology-07 (work in progress), March 2016.

[OpenWSN] Watteyne, T., Vilajosana, X., Kerkez, B., Chraim, F., Weekly, K., Wang, Q., Glaser, S., and K. Pister, "OpenWSN: a Standards-Based Low-Power Wireless Development Environment", Transactions on Emerging Telecommunications Technologies , August 2012.

Appendix A. [TEMPORARY] IETF IE

This section contains a proposal for the specification of an IETF IE. If this proposal is supported by the 6TiSCH WG, the authors of this draft recommend for the specification of the IETF IE to be its own draft, possibly developed in the 6TiSCH WG. The reason for having it a separated document is that the scope of the IETF IE is wider that the 6P protocol defined in this document.

The proposal is to use an IETF IE, a IEEE802.15.4 Payload Information Element with the Group ID set to IANA_IETF_IE_GROUP_ID. The value of IANA_IETF_IE_GROUP_ID is defined by the IEEE, communicated to the IETF, and noted by IANA. The format of the IETF IE is exactly the same as the format of an MLME Information Element, as specified in [IEEE802154-2015], Section 5.2.4.5. The difference is that the space of Sub-IDs is managed by the IETF/IANA. The Sub-ID used by 6top commands is IANA_6TOP_SUBIE_ID with value 0x00.

Other options are being discussed between the IETF 6TiSCH WG and the IEEE 6TiSCH IG, and listed in https://www.ietf.org/mail-archive/web/6tisch/current/msg04469.html. These options concern the way 6P Messages are transported as IEEE802.15.4 IEs, and do not impact the format of those messages.

Appendix B. [TEMPORARY] IEEE Liaison Considerations

If the specification described in this document is supported by the 6TiSCH WG, the authors of this document ask the 6TiSCH WG chairs to liaise with the IEEE to request a Payload Information Element Group ID to be assigned to the IETF (Group ID IANA_IETF_IE_GROUP_ID described in Appendix A).

Appendix C. [TEMPORARY] Terms for the Terminology Draft

Terms introduced by this document, and which needs to be added to [I-D.ietf-6tisch-terminology]:

6top: The "6TiSCH Operation Sublayer" (6top) is the next highest layer of the IEEE802.15.4 TSCH medium access control layer. It implements and terminates the "6top Protocol" (6P), and contains one or more "6top Scheduling Function" (SF). It is defined in TODO_LINK_draft-wang-6tisch-6top-protocol.

SF: The "6top Scheduling Function" (SF) is the policy inside the "6TiSCH Operation Sublayer" (6top) which decides when to add/delete cells. It is defined in TODO_LINK_draft-wang-6tisch-6top-protocol.

SFID: The "6top Scheduling Function Identifier" (SFID) is a 1-byte field identifying a SF. It is defined in TODO_LINK_draft-wang-6tisch-6top-protocol.

6P: The "6top Protocol" (6P) allows neighbor nodes to communicate to add/delete cells to one another in their TSCH schedule. It is defined in TODO_LINK_draft-wang-6tisch-6top-protocol.

6P Transaction: Part of the "6top Protocol" (6P), we call "6top Transaction" a complete negotiation between two neighbor nodes. A transaction starts when a node wishes to add/ delete one or more cells to one of its neighbors; it ends when the cell(s) have been added removed from the schedule of both neighbor, or when the transaction has failed. It is defined in TODO_LINK_draft-wang-6tisch-6top-protocol.

Appendix D. [TEMPORARY] Changelog

- o draft-ietf-6tisch-6top-protocol-01
    * Clarifying locking of resources in concurrent transactions
    * Clarifying return of RC_BUSY in case of concurrent transactions without enough resources
- o draft-ietf-6tisch-6top-protocol-00
    * Informational to Std track
- o draft-wang-6tisch-6top-protocol-00
    * Editorial overhaul: fixing typos, increasing readability, clarifying figures.
    * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/47
    * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/54
    * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/55
    * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/49
    * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/53
    * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/44
    * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/48
    * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/43
    * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/52

  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/45
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/51
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/50
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/46
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/41
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/42
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/39
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/40

- draft-wang-6tisch-6top-sublayer-05

  * Specifies format of IE
  * Adds token in messages to match request and response

- draft-wang-6tisch-6top-sublayer-04

  * Renames IANA_6TOP_IE_GROUP_ID to IANA_IETF_IE_GROUP_ID.
  * Renames IANA_CMD and IANA_RC to IANA_6TOP_CMD and IANA_6TOP_RC.
  * Proposes IANA_6TOP_SUBIE_ID with value 0x00 for the 6top sub-IE.

- draft-wang-6tisch-6top-sublayer-03

  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/32/missing-command-list
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/31/missing-command-count
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/30/missing-command-clear
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/37/6top-atomic-transaction-6p-transaction
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/35/separate-opcode-from-rc
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/36/add-length-field-in-ie
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/27/differentiate-rc_err_busy-and
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/29/missing-rc-rc_reset
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/28/the-sf-must-specify-the-behavior-of-a-mote
  * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/26/remove-including-their-number

   * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/34/6of-sf
   * https://bitbucket.org/6tisch/draft-wang-6tisch-6top-protocol/issues/33/add-a-figure-showing-the-negociation

- draft-wang-6tisch-6top-sublayer-02

   * introduces the 6P protocol and the notion of 6top Transaction.
   * introduces the concept of 6OF and its 6OFID.

Authors' Addresses

Qin Wang (editor)
Univ. of Sci. and Tech. Beijing
30 Xueyuan Road
Beijing, Hebei 100083
China

Phone: +86 (10) 6233 4781
Email: wangqin@ies.ustb.edu.cn

Xavier Vilajosana
Universitat Oberta de Catalunya
156 Rambla Poblenou
Barcelona, Catalonia 08018
Spain

Phone: +34 (646) 633 681
Email: xvilajosana@uoc.edu

6TiSCH D. Dujovne, Ed.
Internet-Draft Universidad Diego Portales
Intended status: Informational LA. Grieco
Expires: January 9, 2017 Politecnico di Bari
MR. Palattella
University of Luxembourg
N. Accettura
University of California Berkeley
July 8, 2016

6TiSCH 6top Scheduling Function Zero (SF0)
draft-ietf-6tisch-6top-sf0-01

Abstract

This document defines a 6top Scheduling Function called "Scheduling Function Zero" (SF0). SF0 dynamically adapts the number of reserved cells between neighbor nodes, based on the currently allocated bandwidth and the neighbour nodes' requirements. Neighbor nodes negotiate in a distributed neighbor-to-neighbor basis the cell(s) to be added/deleted. SF0 uses the 6P signaling messages to add/delete cells in the schedule. Some basic rules for deciding when to add/ delete cells and for selecting the cells to be added/deleted within the schedule are also provided.

Requirements Language

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 RFC 2119 [RFC2119].

Table of Contents

1. Introduction

This document defines the Scheduling Function for the 6top sublayer [I-D.wang-6tisch-6top-sublayer] called "Scheduling Function Zero" (SF0).

This document addresses the requirements for a scheduling function listed in [I-D.wang-6tisch-6top-sublayer], Section 4.2, and follows the recommended outline from Section 4.3. This draft agrees with the terminology defined on [I-D.ietf-6tisch-terminology] and is designed within the context of [RFC7554]

2. Scheduling Function Identifier

The Scheduling Function Identifier (SFID) of SF0 is IANA_SFID_SF0.

3. Rules for Adding/Deleting Cells

A node running SF0 determines when to add/delete cells in a three-step process:

1. It waits for a triggering event (Section 3.1).
2. It applies the Bandwidth Estimation Algorithm (BEA) for a particular neighbor to determine how much bandwidth is required to that neighbor (Section 3.2).
3. It applies the Allocation Policy to compare the number of required cells to the number of already scheduled cells, and determine the number of cells to add/delete (Section 3.3).

3.1. SF0 Triggering Events

We RECOMMEND SF0 to be triggered at least by the following events:

1. If there is a change in the Current Outgoing Bandwidth Usage (COBU)
2. If there is any New Incoming Bandwidth Requirements from neighbour nodes (NIBR)

This allows SF0 to be triggered by any change in local outgoing bandwidth and/or incoming bandwidth. A relocation request from the neighbour is considered as an Incoming Bandwidth Request, given that it is expected to increase packet delivery rate on the relocated cells, thus increasing the required bandwidth. The exact mechanism of when SF0 is triggered is implementation-specific.

3.2. SF0 Bandwidth Estimation Algorithm

The Bandwidth Estimation Algorithm takes into account the sum of the incoming bandwidth requirements from the neighbour nodes and also the current outgoing bandwidth value. This allows the node to estimate the total outgoing bandwidth requirement. As a consequence, the Bandwidth Estimation Algorithm for SF0 follows the steps described below:

1. Collect the New Incoming Bandwidth Requirements from neighbour nodes (NIBR)
2. Obtain the Current Outgoing Bandwidth Usage (COBU)
3. Calculate the New Outgoing Bandwidth (NOB) as: NOB=COBU+NIBR
4. Submit the request to the allocation policy
5. Return to step 1 and wait for a triggering event.

3.3. SF0 Allocation Policy

The "Allocation Policy" is the set of rules used by SF0 to decide when to add/delete cells to a particular neighbor to satisfy the bandwidth requirements.

SF0 uses the following parameters:

SCHEDULEDCELLS: The number of cells scheduled from the current node to a particular neighbor.

REQUIREDCELLS: The number of cells calculated by the Bandwidth Estimation Algorithm from the current node to that neighbor.

SF0THRESH: Threshold parameter introducing cell over-provisioning in the allocation policy. It is a non-negative value expressed as number of cells. The definition of this value is implementation-specific. A setting of SF0THRESH>0 will cause the node to allocate at least SF0THRESH cells to each of its’ neighbours.

The SF0 allocation policy compares REQUIREDCELLS with SCHEDULEDCELLS and decides to add/delete cells taking into account SF0THRESH. This is illustrated in Figure 1.

```
             SCHEDULEDCELLS
 <--------------------------------->
+---+---+---+---+---+---+---+---+---+
|   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+
                |<--------->|
                | SF0THRESH |
                |           |
REQUIREDCELLS   |           |
+---+---+       |           |      DELETE
|   |   |       |           |      ONE/MORE
+---+---+       |           |      CELLS
                |           |
    REQUIREDCELLS           |
+---+---+---+---+---+---+   |      DO
|   |   |   |   |   |   |   |      NOTHING
+---+---+---+---+---+---+   |
                |           |
            REQUIREDCELLS   |
+---+---+---+---+---+---+---+---+---+---+   ADD
|   |   |   |   |   |   |   |   |   |   |   ONE/MORE
+---+---+---+---+---+---+---+---+---+---+   CELLS
```

Figure 1: The SF0 Allocation Policy

1. If REQUIREDCELLS<(SCHEDULEDCELLS-SF0THRESH), delete one or more cells.
2. If (SCHEDULEDCELLS-SF0THRESH)<=REQUIREDCELLS<=SCHEDULEDCELLS, do nothing.
3. If SCHEDULEDCELLS<=REQUIREDCELLS, add one or more cells.

When SF0THRESH equals 0, any discrepancy between REQUIREDCELLS and SCHEDULEDCELLS triggers an action to add/delete cells. Positive values of SF0THRESH reduce the number of 6P Transactions.

The Allocation Policy also translates the bandwidth requirement into cells according to their PDR. For example, if a cell with a 100% PDR is equivalent to 1Kbps, and the required bandwith is 8Kbps, then, the number of scheduled cells will be 8. However, if two of the allocated cells have a 70% PDR, there number of scheduled cells will be 9.

4. Rules for CellList

When issuing a 6top ADD Request, SF0 executes the following sequence:

Whitelist case:

The Transaction Source node: Prepares the CellList field by selecting randomly the required cells, verifying that the slot offset and channel offset are not occupied and choose channelOffset randomly for each cell.
The Transaction Destination node: Goes through the cells in the CellList in order, verifying whether there are no slotOffset conflicts.

Blacklist case:

The Transaction Source node: Prepares the CellList field by building a list of currently scheduled cells into the CellList.
The Transaction Destination node: Selects randomly the required cells from the unallocated cells on the schedule, verifying that the slot offset and channel offset are not occupied from the ones on the CellList.

5. 6P Timeout Value

The general timeout equals the equivalent time of the number of slots until the next scheduled cell.

6. Meaning of Metadata Information

The Metadata 16-bit field is used as follows:

BITS 0-7 [SLOTFRAME] are used to identify the slotframe number
BITS 8-14 are RESERVED
BIT 15 [WBLIST] is used to indicate that the CellList provided is a Whitelist (value=0) or a Blacklist (value=1).

TODO: length of the SlotFrame SHOULD be an integer multiple of the length of the minimal SlotFrame.

7. Node Behavior at Boot

In order to define a known state after the node is restarted, a CLEAR command is issued to each of the neighbour nodes to enable a new allocation process. The 6P Initial Timeout Value provided by SF0 allows the maximum number of TSCH link-layer retries. Given the TSCH parameters for the backoff mechanism, macMinBE and macMaxBE, and the length in seconds of the minimal Slotframe, SM, the timeout value is computed as: timeout = (2^(macMaxBE+1)-2^macMinBE) * SM

8. Relocating Cells

SF0 uses Packet Delivery Rate (PDR) statistics to monitor the currently allocated cells for cell re-allocation (by changing their slotOffset and/or channelOffset) when it finds out that the PDR of one or more softcells below 20% of the average PDR.

9. Forced Cell Deletion Policy

TODO: When all the cells are scheduled, we need a policy to free cells, for example, under alarm conditions or if a node dissappears from the neighbour list.

10. 6P Error Handling

A node implementing SF0 handles a 6P Response depending on the Return Code it contains:

RC_SUCCESS:
: If the number of elements in the CellList is the number of cells specified in the NumCells field of the 6P ALL Request, the operation is complete. The node does not take further action. If the number of elements in the CellList is smaller (possibly 0) than the number of cells specified in the NumCells field of the 6P ALL Request, the neighbor has received the request, but less than NumCells of the cells in the CellList were. In that case, the node MAY retry immediately with a different CellList if the amount of storage space permits, or build a new (random) CellList.

RC_VER_ERR: The node MUST NOT retry immediately. The node MAY add the neighbor node on a blacklist. The node MAY retry to contact this neighbor later.

RC_SFID_ERR: The node MUST NOT retry immediately. The node MAY add the neighbor node on a blacklist. The node MAY retry to contact this neighbor later.

RC_BUSY: Wait for a timeout and restart the scheduling process.

RC_RESET: Abort 6P Transaction

RC_ERR: Abort 6P Transaction. The node MAY retry to contact this neighbor later.

11. Examples

TODO

12. Implementation Status

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC6982]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC6982], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

OpenWSN: This specification is implemented in the OpenWSN project [OpenWSN]. The authors of this document are collaborating with the OpenWSN community to gather feedback about the status and performance of the protocols described in this document. Results from that discussion will appear in this section in future revision of this specification.

13. Security Considerations

TODO

14. IANA Considerations

- IANA_SFID_SF0

15. Acknowledgments

Thanks to Kris Pister for his contribution in designing the default Bandwidth Estimation Algorithm. Thanks to Qin Wang and Thomas Watteyne for their support in defining the interaction between SF0 and the 6top sublayer.

This work is partially supported by the Fondecyt 1121475 Project, the Inria-Chile "Network Design" group, and the IoT6 European Project (STREP) of the 7th Framework Program (Grant 288445).

16. References

16.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

16.2. Informative References

[RFC7554] Watteyne, T., Ed., Palattella, M., and L. Grieco, "Using IEEE 802.15.4e Time-Slotted Channel Hopping (TSCH) in the Internet of Things (IoT): Problem Statement", RFC 7554, DOI 10.17487/RFC7554, May 2015, <http://www.rfc-editor.org/info/rfc7554>.

[RFC6982] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", RFC 6982, DOI 10.17487/RFC6982, July 2013, <http://www.rfc-editor.org/info/rfc6982>.

[I-D.ietf-6tisch-terminology] Palattella, M., Thubert, P., Watteyne, T., and Q. Wang, "Terminology in IPv6 over the TSCH mode of IEEE 802.15.4e", draft-ietf-6tisch-terminology-07 (work in progress), March 2016.

[I-D.wang-6tisch-6top-sublayer] Wang, Q. and X. Vilajosana, "6TiSCH Operation Sublayer (6top)", draft-wang-6tisch-6top-sublayer-04 (work in progress), November 2015.

[OpenWSN] Watteyne, T., Vilajosana, X., Kerkez, B., Chraim, F., Weekly, K., Wang, Q., Glaser, S., and K. Pister, "OpenWSN: a Standards-Based Low-Power Wireless Development Environment", Transactions on Emerging Telecommunications Technologies , August 2012.

Authors’ Addresses

Diego Dujovne (editor)
Universidad Diego Portales
Escuela de Informatica y Telecomunicaciones
Av. Ejercito 441
Santiago, Region Metropolitana
Chile

Phone: +56 (2) 676-8121
Email: diego.dujovne@mail.udp.cl

Luigi Alfredo Grieco
Politecnico di Bari
Department of Electrical and Information Engineering
Via Orabona 4
Bari 70125
Italy

Phone: 00390805963911
Email: a.grieco@poliba.it

Maria Rita Palattella
University of Luxembourg
Interdisciplinary Centre for Security, Reliability and Trust
4, rue Alphonse Weicker
Luxembourg L-2721
LUXEMBOURG

Phone: (+352) 46 66 44 5841
Email: maria-rita.palattella@uni.lu

Nicola Accettura
University of California Berkeley
Berkeley Sensor & Actuator Center
490 Cory Hall
Berkeley, California 94720
USA

Email: nicola.accettura@eecs.berkeley.edu

6tisch
Internet-Draft
Intended status: Standards Track
Expires: January 19, 2017

S. Anamalamudi
M. Zhang
Huawei Technologies
C. Perkins
Futurewei
S.V.R.Anand
Indian Institute of Science
July 18, 2016

Scheduling Function One (SF1) for hop-by-hop Scheduling in 6tisch Networks
draft-satish-6tisch-6top-sf1-01

Abstract

This document defines a 6top Scheduling Function called "Scheduling Function One" (SF1) to reserve, label and schedule the end-to-end resources hop-by-hop through distributed Resource Reservation Protocol(RSVP). SF1 uses the 6P signaling messages with a global TrackID to add/delete cells in end-to-end L2-bundles of isolated instance.

Table of Contents

1. Introduction

With Scheduling Function Zero (SF0) [I-D.dujovne-6tisch-6top-sf0], on-the-fly cell scheduling (ADD/DELETE) to 1-hop neighbors can be achieved for aggregated (best-effort) traffic flows. In other words, all the instances from nodeA to nodeB in Fig. 1 are scheduled in a single L3-bundle (IP link).

```
               L3-bundle(Instance-1,Instance-2,...Instance-n)
            -------------------------------------------------->
      nodeA<-------------------------------------------------  nodeB
               L3-bundle(Instance-1,Instance-2,...Instance-n)
```

Figure 1: L3-bundle for aggregated traffic flows in 1-hop with SF0.

Some applications (e.g. Industrial M2M) require end-to-end dedicated L2-bundles to protect the control/data streams for time-critical applications [I-D.ietf-detnet-use-cases]. For such applications, per-instance based L2-bundles need to be scheduled hop-by-hop in between application sender and receiver nodes [I-D.ietf-6tisch-architecture]. In addition, cells in the scheduled end-to-end L2-bundles of each instance have to be dynamically adapted for bursty time-critical traffic flows. To achieve, end-to-end track has to be installed with a global TrackID that is associated with the

L2-bundles of each instance. With 1-hop based SF0 cell scheduling, it is difficult to schedule dedicated end-to-end cells for isolated traffic flows. In addition, global bandwidth estimation through Resource Reservation protocol is required for bandwidth allocation in multi-hop cell scheduling. This draft proposes a Scheduling Function One (SF1) to schedule end-to-end dedicated L2-bundles for each instance, and to dynamically adapt the cells in scheduled L2-bundles of ongoing instance through RSVP protocol(see Fig. 2).

```
                L2-bundle(Instance-1)      L2-bundle(Instance-1)
             ------------------------>    ------------------->
            <------------------------     <--------------------
                L2-bundle(Instance-1)      L2-bundle(Instance-1)

                L2-bundle(Instance-2)      L2-bundle(Instance-2)
             ----------------------->     ----------------->
      Sender<-----------------------nodeB <----------------- Receiver
                L2-bundle(Instance-2)     L2-bundle(Instance-2)
                        .                          .
                        .                          .
                L2-bundle(Instance-n)     L2-bundle(Instance-n)
             ------------------------>   -------------------->
            <------------------------    <--------------------
                L2-bundle(Instance-n)     L2-bundle(Instance-n)
```

Figure 2: Dedicated L2-bundles for end-to-end isolated traffic flows with SF1

2. Operation of Scheduling Function one (SF1)

With SF1, Sender determines when to reserve the end-to-end resources, support implicit label switching(GMPLS), schedule the labeled L2-bundles hop-by-hop, associate the global TrackID for labeled L2-bundles, and dynamically adapt the cells in ongoing instance through distributed Resource Reservation Protocol (RSVP-lite). The triggering events in SF1 are as follows :

1. If Sender has any Outgoing Bandwidth Requirement for new instance to transmit data to Receiver.

2. If Sender has a New Outgoing Bandwidth Requirement for Ongoing Instance to transmit data to Receiver.

In both cases, distributed RSVP-lite (explained in Section .2.1) is triggered to provide end-to-end resource reservations along with scheduling operations.

2.1. Resource Reservation Protocol(RSVP-lite)

In this specification, an end-to-end route path is assumed to be available with reactive P2P-RPL (Storing or non-storing mode) protocols. A distributed Resource Reservation Protocol (RSVP-lite) with 6tisch scheduling capability is designed to schedule the labeled reserved resources hop-by-hop for isolated instance. SF1 of application sender will trigger the RSVP-lite operation, whenever it has time critical traffic flow towards the receiver. The RSVP-lite has two messages namely (1) RSVP-PATH message(Sender to Receiver) and (2) RSVP-RESV message(Receiver to Sender).

2.2. RSVP-PATH message

The basic RSVP-PATH message [RFC2205] is used to carry the "Sender Traffic Specification" along with "characterization parameters" from sender to receiver. Since RSVP treat objects as opaque data, it is valid to assume other protocol(eg., GMPLS, 6P) as an object in RSVP-PATH messages.

The format of PATH message with the support of 6tisch scheduling capabilities (6P and SF1) is as follows :

```
 <Path Message> ::=    <Common Header> [ <INTEGRITY> ]
                       [ [<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ... ]
                       [ <MESSAGE_ID> ]
                       <SESSION> <RSVP_HOP>
                       <TIME_VALUES>
                       [ <EXPLICIT_ROUTE> ]
                       <LABEL_REQUEST>
                       [ <PROTECTION> ]
                       [ <LABEL_SET> ... ]
                       [<SF1 OPERATION REQUEST>]
                       [<6P OPERATION REQUEST>]
                       [ <SESSION_ATTRIBUTE> ]
                       [ <NOTIFY_REQUEST> ]
                       [ <ADMIN_STATUS> ]
                       [ <POLICY_DATA> ... ]
                       <sender descriptor>
```

"SF1 OPERATION REQUEST" and "6P OPERATION REQUEST" are added in the PATH message to check for 6tisch scheduling capabilities within the intermediate nodes from sender to receiver. "Cell Switching Capability(CSC)" is proposed to make use of cell as an implicit label in RSVP-GMPLS [RFC3473]. The message format of the "CSC" is out-of-scope in this specification. "LABEL_REQUEST" in path message should set to "Cell Switching Capability". "RPLInstanceID" is added in the "SENDER_TEMPLATE" to create Global TrackID during 6P transactions of

RSVP-RESV message. Whenever an intermediate node does not support the "Cell switching Capability" or "6P transactions" or "SF1 operation" then it needs to send a "PathErr" message back to the application sender. The message format for "PathErr" codes with respect to Scheduling capability is out-of-scope in this specification.

2.3. RSVP-RESV message

The basic RSVP-RESV messages [RFC2205] are transmitted upstream from receiver to sender to provide resource reservation along with "Label Distribution". In this specification, hop-by-hop scheduling is extended to support both resource reservation and label distribution. The current specification is only defined for unicast point-to-point traffic flows, i.e., Fixed Filter (FF) reservation style.

The format of RESV message with the support of 6tisch scheduling capabilities (6P and SF1) is as follows :

```
 <Resv Message> ::=    <Common Header> [ <INTEGRITY> ]
                       [ [<MESSAGE_ID_ACK> | <MESSAGE_ID_NACK>] ... ]
                       [ <MESSAGE_ID> ]
                       <SESSION> <RSVP_HOP>
                       <TIME_VALUES>
                       <LABEL>
                       [<SF1 OPERATION>]
                       [<6P OPERATION>]
                       [ <RESV_CONFIRM> ]  [ <SCOPE> ]
                       [ <NOTIFY_REQUEST> ]
                       [ <ADMIN_STATUS> ]
                       [ <POLICY_DATA> ... ]
                       <STYLE> <flow descriptor list>
```

Upon arrival of the PATH message at an application receiver, the SENDER_TSPEC and ADSPEC objects are interpreted to select the resource reservation parameters. Since the RSVP-lite provides receiver initiated resource reservation setup, the scheduling operation needs to be performed upstream from receiver to sender. Subsequently, the reserved resources (bandwidth) are mapped into 6tisch cells through Scheduling Function and corresponding L2-bundle is created. An aggregation of cells is called "bundle"(the directional link to a next-hop neighbor). Every L2-bundle is associated with a global trackID to dynamically adapt the cells "hop-by-hop" to a scheduled instance. In addition, the TrackID is used as a "packet filter" to switch the incoming tracks to outgoing tracks. The receiver will generate the TrackID with the combination of "Sender/Receiver IP address" and "RPLInstanceID" that is obtained from "SENDER_TEMPLATE/FILTER_SPEC" and "SESSION" object.

```
                  next-hop node            Receiver
                 +--------------+      +--------------+
                 |     IPv6     |      |     IPv6     |
                 +--------------+      +--------------+
                 |   6LoWPAN    |      |   6LoWPAN    |
                 +--------------+      +--------------+
                 |     6top     |      |     6top     |
                 +--------------+      +--------------+
                 |   TSCH MAC   |      |   TSCH MAC   |
                 +--------------+      +--------------+
                 |   LLN PHY    |      |   LLN PHY    |
                 +--------------+      +--------------+
                      |                           |
                      |                           | Rspec:Reserves
                      |                           | bandwith
                      |                           |
                      |                           | SF1:Maps bandwidth to
                      |                           | cells
                      | RESV + 6P Request(TrackID)|
                      |<------------------------- |
 Rspec:Reserves       |                           |
 bandwith             |                           |
                      |                           |
 SF1:Maps bandwidth to|                           |
 cells                |6P Response (CellList[..]) |
                      |-------------------------->|
                      |                           |
                      |                           |
                      |                           |
                      |   6P confirmation         |LABEL SET
                      |   CellList[..]+ Label     |label=Channel+Slot
                      |<--------------------------|
      Resv state:"Cell|                           |
      label"          |                           |
                      |                           |
                      |                           |
                      |                           |
```

Figure 3: Operation of RSVP-RESV message with 6P transactions.

From [RFC6997], it is noteworthy that application sender that initiates the point-to-point (P2P) traffic is called "Parent node" and application receiver that receives the data is called "Child node". Since the receiver (child node) performs the Scheduling operation upstream towards the sender, "3-step transaction" of 6P protocol needs to be triggered hop-by-hop to schedule the reserved resources(see Fig. 3). Hence, "6P Request" with associated TrackID (in metadata field) is transmitted in "RESV" message from Receiver to next-hop node. The "NumCells" field in the 6P Request is set to required number of cells whereas receive "CellList" should be empty. Once the outgoing interface of next-hop node receive the "RESV" message, it checks the service request specification(Rspec) and perform the resource reservation. Subsequently, Scheduling Function of next-hop neighbor map the reserved resources into transmit cells. Later, "6P Response" with transmit "CellList" (slotOffset,channelOffset) is downstream to receiver. When the receiver has cells (to receive data) available with the "CellList" specified in the "6P Response" then "6P Confirmation" with "IANA_6TOP_RC_SUCCESS" is upstream towards next-hop node. Otherwise, "ResvErr" message should send back to the receiver with specific error. Since the cell characteristics(slotOffset,channelOffset) is used as a label, the next-hop node will store the "SlotOffset+ChannelOffset" as a label to switch the traffic flow to receiver. For the multiple cells (Bundle), a generalized label set is created where each label represents one cell to forward data to receiver. The "Generalized Label Set" is transmitted in the metadata field of "6P confirmation" from receiver to upstream node. Once the 6P transaction is successful in between next-hop node and receiver, a labeled L2-bundle is created with the associated TrackID. Subsequently, "cell label set" is stored in the Resv state block at the next-hop node. Later, SF1 of "next-hop node" maps the reserved bandwidth to the "receiving cells" to receive the data from its upstream node. The "RESV" message with "6P Request" along with TrackID is transmitted upstream towards sender node. With this, end-to-end Track is installed with a succession of paired L2-bundles(a receive bundle from the previous hop and a transmit bundle to the next hop) for a specific instance from sender to receiver(See Fig. 4).

```
 +--------------+  <-Data transmission in end-to-end Track->
 |     IPv6     | Sender                            Receiver
 +--------------+   |                                     |
 |   6LoWPAN    |   |                                     |
 +--------------+   |                nodeB                |
 |     6top     |   |               +----+                |
 +--------------+   |               |    |                |
 |   TSCH MAC   |   |               |    |                |
 +--------------+   |               |    |                |
 |   LLN PHY    |   |   L2-Bundle   |    |   L2-Bundle    |
 +--------------+   +---------------+    +----------------+
                    <--Dedicated cells for each Instance-->
```

Figure 4: End-to-end cell scheduling with SF1 Scheduling

During data transmission, SF1 of sender at 6top identifies the TrackID based on "Sender/Receiver IP address, RPLInstanceID" from the received IP packet with RPI header information [RFC6553]. Subsequently, an associated L2-bundle is scheduled to forward the data to next-hop neighbor (nodeB in Fig.4). Later, SF1 of the next-hop neighbor identifies the TrackID based on "Sender/Receiver IP address, RPLInstanceID" of the received data to switch the track towards receiver. With this, end-to-end data transmission is achieved through "Track forwarding" at the 6top sub-layer (see Fig. 4). With "Cell Switching Capability" of RSVP-GMPLS [RFC3473], cells in paired L2-bundles are used as an implicit labels to label switch the data from Sender to Receiver at the 6top sub-layer.

2.4. Reroute and Bandwidth Increase mechanism

Whenever, the sender needs to setup a new tunnel that is capable of maintaining resource reservations without double counting(at same intermediate node) the resources with existing tunnel then "RSVP reroute mechanism" need to be initiated [RFC3209]. With this operation, bandwidth can increase/decrease end-to-end in the ongoing tunnel. The detailed explanation of "Reroute mechanism" is explained in [RFC3209].

2.5. Error Codes

The detailed explanation of PathErr and ResvErr with different ERROR_SPEC to handle Scheduling and 6P operation errors will be described in later specification.

3. Scheduling Function Identifier

The Scheduling Function Identifier (SFID) of SF1 is IANA_SFID_SF1(TBD).

4. IANA Considerations

IANA is requested to allocate a new Scheduling Function (IANA_SFID_SF1) from the SF space of Scheduling Functions defined in [I-D.dujovne-6tisch-6top-sf0]

5. Security Considerations

TODO

6. References

6.1. References

[RFC2205] Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, DOI 10.17487/RFC2205, September 1997, <http://www.rfc-editor.org/info/rfc2205>.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <http://www.rfc-editor.org/info/rfc3209>.

[RFC3473] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, DOI 10.17487/RFC3473, January 2003, <http://www.rfc-editor.org/info/rfc3473>.

[RFC6553] Hui, J. and JP. Vasseur, "The Routing Protocol for Low-Power and Lossy Networks (RPL) Option for Carrying RPL Information in Data-Plane Datagrams", RFC 6553, DOI 10.17487/RFC6553, March 2012, <http://www.rfc-editor.org/info/rfc6553>.

[RFC6997] Goyal, M., Ed., Baccelli, E., Philipp, M., Brandt, A., and J. Martocci, "Reactive Discovery of Point-to-Point Routes in Low-Power and Lossy Networks", RFC 6997, DOI 10.17487/RFC6997, August 2013, <http://www.rfc-editor.org/info/rfc6997>.

6.2. Informative References

[I-D.dujovne-6tisch-6top-sf0]
Dujovne, D., Grieco, L., Palattella, M., and N. Accettura, "6TiSCH 6top Scheduling Function Zero (SF0)", draft-dujovne-6tisch-6top-sf0-01 (work in progress), March 2016.

[I-D.ietf-6tisch-6top-protocol]
Wang, Q. and X. Vilajosana, "6top Protocol (6P)", draft-ietf-6tisch-6top-protocol-01 (work in progress), June 2016.

[I-D.ietf-6tisch-architecture]
Thubert, P., "An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4", draft-ietf-6tisch-architecture-10 (work in progress), June 2016.

[I-D.ietf-detnet-use-cases]
Grossman, E., Gunther, C., Thubert, P., Wetterwald, P., Raymond, J., Korhonen, J., Kaneko, Y., Das, S., Zha, Y., Varga, B., Farkas, J., Goetz, F., and J. Schmitt, "Deterministic Networking Use Cases", draft-ietf-detnet-use-cases-10 (work in progress), July 2016.

Authors' Addresses

Satish Anamalamudi
Huawei Technologies
No. 156 Beiqing Rd. Haidian District
Beijing 100095
China

Email: satish.anamalamudi@huawei.com

Mingui Zhang
Huawei Technologies
No. 156 Beiqing Rd. Haidian District
Beijing 100095
China

Email: zhangmingui@huawei.com

Charles E. Perkins
Futurewei
2330 Central Expressway
Santa Clara 95050
Unites States

Email: charliep@computer.org

S.V.R Anand
Indian Institute of Science
Bangalore
560012
India

Email: anand@ece.iisc.ernet.in