6TiSCH
Internet-Draft
Intended status: Standards Track
Expires: May 4, 2017

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

October 31, 2016

6top Protocol (6P)
draft-ietf-6tisch-6top-protocol-03

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

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 http://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 May 4, 2017.

Table of Contents

1. TEMPORARY EDITORIAL NOTES 3
2. Introduction 3
3. 6TiSCH Operation Sublayer (6top) 5
3.1. Hard/Soft Cells 5
3.2. Using 6top with the Minimal 6TiSCH Configuration 5
4. 6top Protocol (6P) 6
4.1. 6top Transaction 6
4.1.1. 2-step 6top Transaction 7
4.1.2. 3-step 6top Transaction 8
4.2. Message Format 10
4.2.1. 6top Information Element 10
4.2.2. General Message Format 10
4.2.3. 6P Message Types 11
4.2.4. 6P Command Identifiers 12
4.2.5. 6P Return Codes 12
4.2.6. 6P CellOptions 13
4.2.7. 6P Cell Format 15
4.2.8. 6P ADD Request Format 15
4.2.9. 6P DELETE Request Format 16
4.2.10. 6P STATUS Request Format 17
4.2.11. 6P LIST Request Format 17
4.2.12. 6P CLEAR Request Format 18
4.2.13. 6P Response Format 19
4.2.14. 6P Confirmation Format 20
4.3. Protocol Behavior 20
4.3.1. Version Checking 20
4.3.2. SFID Checking 20
4.3.3. Concurrent 6P Transactions 21
4.3.4. Timeout 21
4.3.5. SeqNum Mismatch 21
4.3.6. Clearing the Schedule 22

4.3.7. Adding Cells with 2-way Transaction 22
4.3.8. Aborting a 6P Transaction 22
4.3.9. Deleting Cells 23
4.3.10. Listing Cells 23
4.3.11. Generation Management 24
4.3.12. Handling error responses 25
4.4. Security 25
5. Guidelines for 6top Scheduling Functions (SF) 25
5.1. SF Identifier (SFID) 26
5.2. Requirements for an SF 26
5.3. Recommended Structure of an SF Specification 27
6. Implementation Status 27
7. Security Considerations 28
8. IANA Consideration 28
9. References 29
9.1. Normative References 29
9.2. Informative References 29
Appendix A. [TEMPORARY] Changelog 30
Authors' Addresses 32

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, etc). The statement does indicate that the text will be written at some time.
- "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.
- The string "REMARK" is put before a remark (questions, suggestion, etc) from an author, editor of contributor. These are on-going discussions at the time to writing, NOT part of the final text.
- 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). 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.

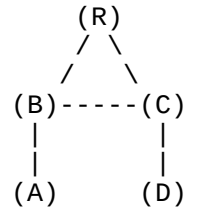


Figure 1: A simple 6TiSCH network.

The example network depicted in Figure 1 is used to describe the interactions between nodes. We consider the canonical case where node "A" issues 6P requests to node "B". We keep this example throughout this document. Throughout the discussions, node A will always represent the node that issues a 6P request; node B the node that receives this request.

We consider node A in Figure 1 monitoring the communication cells it has in its schedule to node B.

- If node A determines that the number of link-layer frames it is sending to B per unit of time is larger than the capacity offered by the TSCH cells it has scheduled to B, it triggers a 6P Transaction with node B to add one or more cells to B's TSCH schedule.
- If the traffic is lower than the capacity, node A triggers a 6P Transaction with node B to delete one or more cells in the TSCH schedule of both nodes.
- Node A MAY also monitor statistics to determine whether collisions are happening on a particular cell to node B. If this feature is enabled, node A communicates with node B 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 handled by the scheduling function (see below).

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].

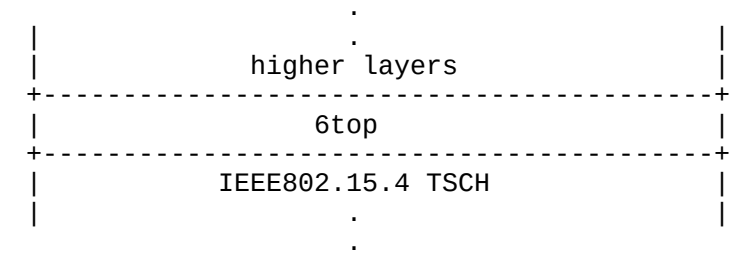


Figure 2: The 6top sublayer in the protocol stack.

The roles of the 6top sublayer are to:

- Implement and terminate the 6top Protocol (6P), which allows neighbor nodes to communicate to add/delete cells to one another.
- Run one or more 6top Scheduling Functions (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":

- a soft cell can be read, added, deleted or updated by 6top.
- 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

6P 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 to avoid for cells in slotframe 1 to "mask" cells in slotframe 0. 6top MAY support further slotframes; how to use more slotframes is out of the scope for this document.

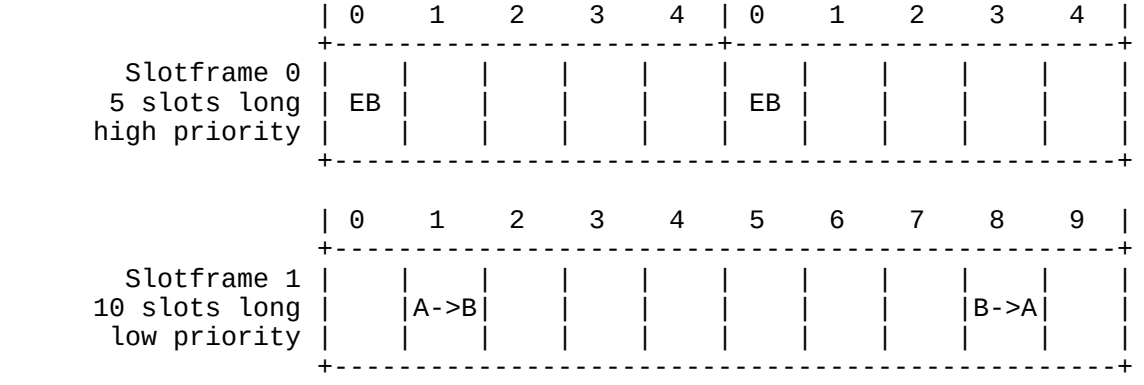


Figure 3: 2-slotframe structure when using 6top alongside the Minimal 6TiSCH Configuration.

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 6P 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 neighbors, or when the 6P Transaction has failed.

A 6P 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.

Consistency between the schedules of two neighbor nodes is of utmost importance. A loss of consistency (e.g. node A has a transmit cell

to node B, but node B does not have the corresponding reception cell) can cause loss of connectivity. To verify consistency, neighbors nodes increment the "schedule generation" number of their schedule each time they add/remove a cell. Neighbor nodes exchange generation numbers at each 6P Transaction to detect possible inconsistencies. This mechanism is explained in Section 4.3.11.

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

4.1.1. 2-step 6top Transaction

Figure 4 is a sequence diagram to help understand the 2-step 6top transaction (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.

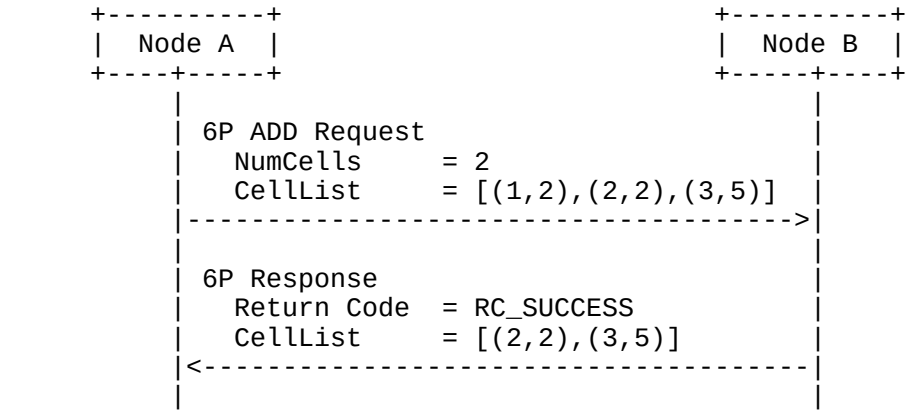


Figure 4: A 2-step 6P Transaction.

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 cells (the "CellList" value). Each cell in the CellList is a [slotOffset,channelOffset] tuple.
3. Node A at the same time sets a timeout timer to abort the transaction if no response has been received when it expires. The value of the timeout is out of the scope of this document and MAY be defined by the SF. More details are given in Section 4.3.8.

4. 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.
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.

4.1.2. 3-step 6top Transaction

Figure 5 is a sequence diagram to help understand the 3-step 6top transaction. 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.

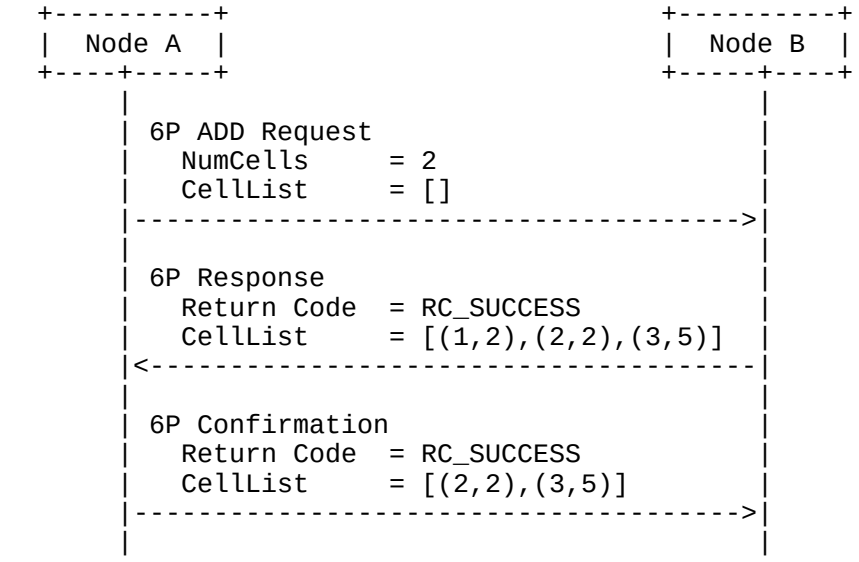


Figure 5: A 3-step 6P Transaction.

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 an empty "CellList".
3. Node A at the same time sets a timeout timer to abort the transaction if no response has been received when it expires. The value of the timeout is out of the scope of this document and MAY be defined by the SF. More details are given in Section 4.3.8.
4. 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.

5. Node B at the same time sets a timeout timer to abort the transaction if no response has been received when it expires. The value of the timeout is out of the scope of this document and MAY be defined by the SF. More details are given in Section 4.3.8.
6. The SF running on node A selects 2 cells. Node A sends back a 6P Confirmation to node B, indicating the cells it selected.
7. 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.

When in a transaction, node A proposes a candidate CellList to node B and B cannot allocate any of those cells. Node B SHOULD respond with a CellList suggesting alternatives. This approach facilitates the agreement between A and B and enables A to not guess what cells may be not used in B. The following figure ilustrated these 3-step transaction.

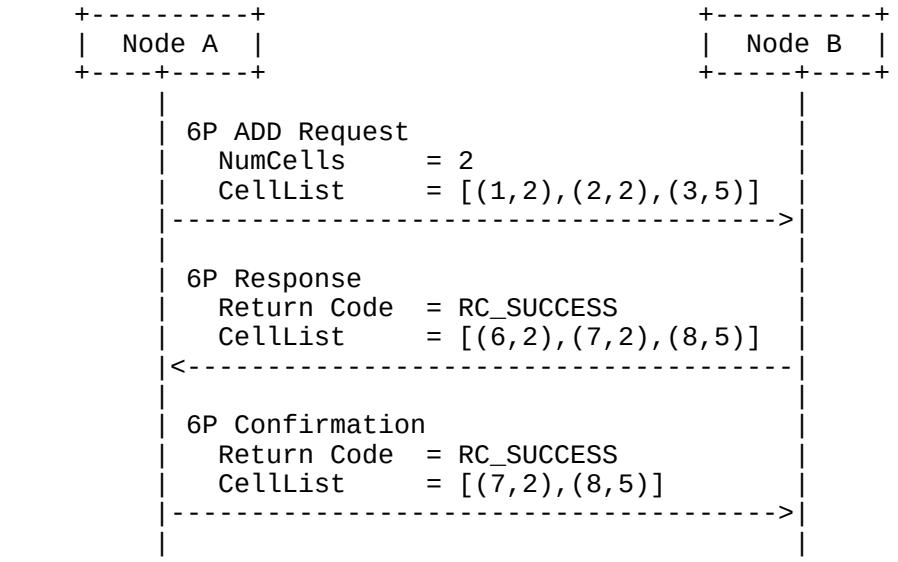


Figure 6: A 3-step 6P Transaction with cell suggestion.

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, and selects a candidate list of cells.
2. Node A sends a 6P ADD Request to node B, indicating it wishes to add 2 cells (the "NumCells" value), with an the proposed "CellList".
3. Node A at the same time sets a timeout timer to abort the transaction if no response has been received when it expires.

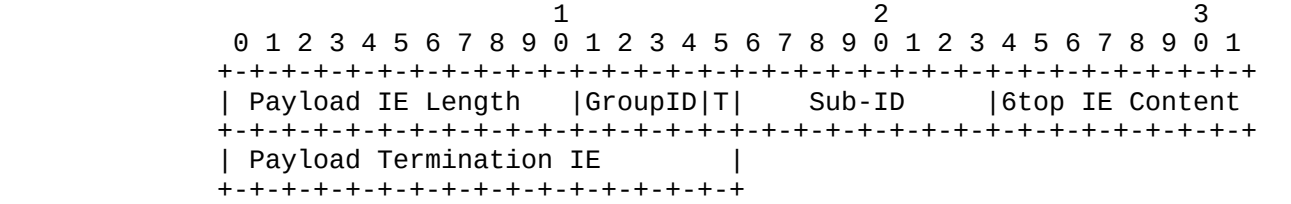
The value of the timeout is out of the scope of this document and MAY be defined by the SF. More details are given in Section 4.3.8.

4. The SF running on node B cannot match any of the proposed cells and selects 3 alternative candidate cells. Node B sends back a 6P Response to node A, indicating the 3 candidate alternative cells it selected.
5. Node B at the same time sets a timeout timer to abort the transaction if no response has been received when it expires. The value of the timeout is out of the scope of this document and MAY be defined by the SF. More details are given in Section 4.3.8.
6. The SF running on node A selects 2 cells from the proposed CellList. Node A sends back a 6P Confirmation to node B, indicating the cells it selected.
7. 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

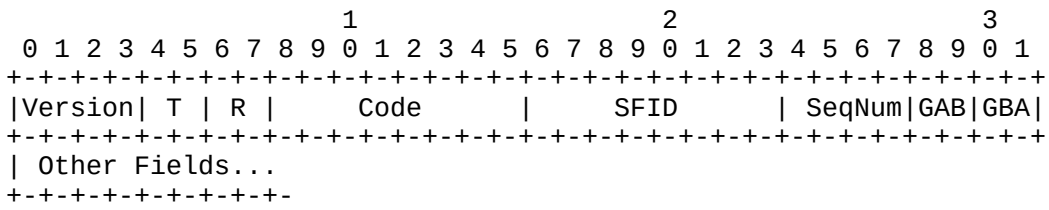
6P messages are carried as payload of IEEE802.15.4 Payload Information Elements (IE) [IEEE802154-2015]. 6p messages travel over a single hop.



The 6top IE is an IEEE Payload IE with GroupID IANA_IETF_IE_GROUP_ID. The 6top IE complies with the IE format defined in [I-D.kivinen-802-15-ie]. 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].

4.2.2. General Message Format

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



Version (6P Version): The version of the 6P protocol. Only version IANA_6TOP_6P_VERSION is defined in this document. Future specifications MAY define further versions of the 6P protocol.

Type (T): Type of message. The possible messages types are defined in Section 4.2.3.

Reserved (R): These two bits SHOULD be set to zero when sending the message and MUST be ignored on reception.

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 the 6P Request, 6P Response and 6P Confirmation of the same 6P Transaction. The value of SeqNum MUST increment by exactly one at each new 6P request issued to the same neighbor.

GAB: Schedule Generation for the cells scheduled from node A to node B. The generation is used to ensure consistency between the schedule of the two neighbors. Section 4.3.11 details how schedule generation is managed.

GBA: Schedule Generation for the cells scheduled from node B to node A.

Other Fields: The list of other fields depends on the value of the code field, as detailed below.

4.2.3. 6P Message Types

Figure 7 lists the 6P message types.

Value of the "Type" field	Meaning
b00	6P Request
b01	6P Response
b10	6P Confirmation (3-step 6top Transaction only)
b11	Reserved

Figure 7: 6P Message Types

4.2.4. 6P Command Identifiers

The Code field contains a 6P Command Identifier when 6P Message is a 6P Request. Figure 8 lists the 6P command identifiers.

Command ID	Value	Description
CMD_ADD	IANA_6TOP_CMD_ADD	add one or more cells
CMD_DELETE	IANA_6TOP_CMD_DELETE	delete one or more cells
CMD_STATUS	IANA_6TOP_CMD_STATUS	status of the schedule
CMD_LIST	IANA_6TOP_CMD_LIST	list the scheduled cells in node B
CMD_CLEAR	IANA_6TOP_CMD_CLEAR	clear all cells on both node A and node B
reserved	TODO-0xf	reserved

Figure 8: 6P Command Identifiers

4.2.5. 6P Return Codes

The Code field contains a 6P Return Code when 6P Message is a 6P Response or a 6P Confirmation. Figure 9 lists the 6P Return Codes and their meaning.

Return Code	Value	Description
RC_SUCCESS	IANA_6TOP_RC_SUCCESS	operation succeeded
RC_ERR_VER	IANA_6TOP_RC_ERR_VER	unsupported 6P version
RC_ERR_SFID	IANA_6TOP_RC_ERR_SFID	unsupported SFID
RC_ERR_GEN	IANA_6TOP_RC_ERR_GEN	schedule generation error
RC_ERR_BUSY	IANA_6TOP_RC_ERR_BUSY	handling previous request
RC_ERR_NORES	IANA_6TOP_RC_ERR_NORES	not enough resources
RC_ERR_RESET	IANA_6TOP_RC_ERR_RESET	error in state machine wrong sequence of commands
RC_ERR	IANA_6TOP_RC_ERR	generic error
reserved	TODO-0xf	

Figure 9: 6P Return Codes

4.2.6. 6P CellOptions

The 6P CellOptions field is present in the 6P ADD, the 6P DELETE, the 6P STATUS and the 6P LIST requests. The 6P CellOptions apply to all elements contained in the CellList field. Hence all cells in the CellList will be of the same type. In the 6P ADD request, it is used to specify what type of cell to add. In the 6P DELETE request, it is used to specify what type of cell to delete. In the 6P STATUS and the 6P LIST requests, it is used as a selector of particular types of cells. Figure 10 contains the RECOMMENDED format of the 6P CellOptions field. Figure 11 contains the RECOMMENDED meaning of the 6P CellOptions field for the 6P STATUS and 6P LIST requests.

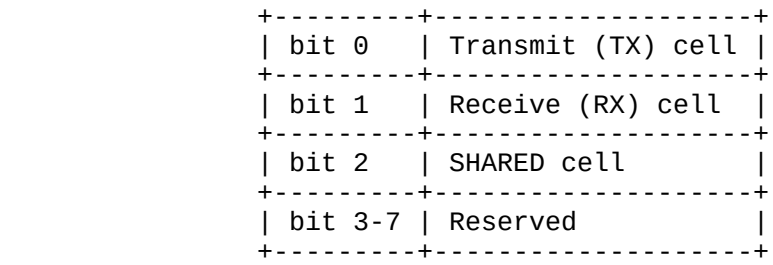


Figure 10: Format of the CellOptions field

Note: assuming node A issues the 6P command to node B.

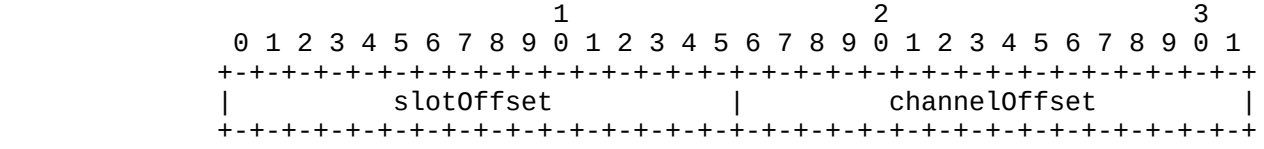
CellOptions Value	B's action when receiving a 6P message from A
TX=0,RX=0,S=0	select all cells scheduled with A
TX=1,RX=0,S=0	select the cells scheduled with A and marked as RX
TX=0,RX=1,S=0	select the cells scheduled with A and marked as TX
TX=1,RX=1,S=0	select the cells scheduled with A and marked as TX and RX
TX=0,RX=0,S=1	select the cells scheduled with A and marked as SHARED
TX=1,RX=0,S=1	select the cells scheduled with A and marked as RX and SHARED
TX=0,RX=1,S=1	select the cells scheduled with A and marked as TX and SHARED
TX=1,RX=1,S=1	select the cells scheduled with A and marked as TX and RX and SHARED

Figure 11: Meaning of the 6P CellOptions field for the 6P STATUS and the 6PLIST requests

The CellOptions is an opaque set of bits, sent unmodified to the SF. The SF MAY redefine the format of the CellOptions field. The SF MAY redefine the meaning of the CellOptions field.

4.2.7. 6P Cell Format

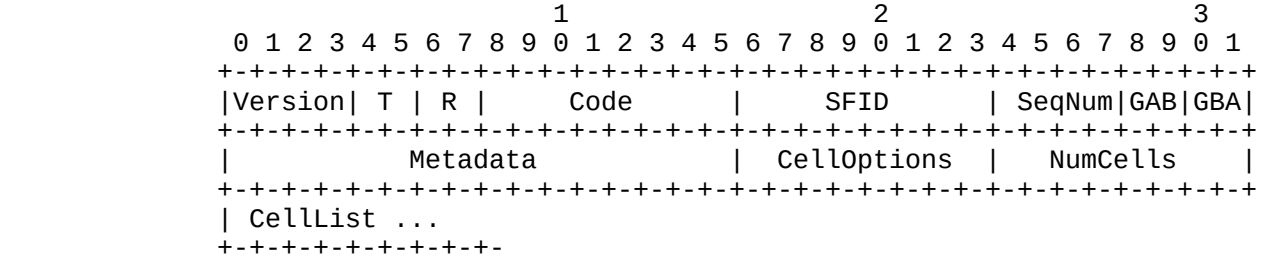
A CellList field MAY be present in a 6P ADD Request, 6P DELETE Request, a 6P Response or a 6P Confirmation. It is composed of zero, one of more 6P Cell containers. The CellOptions field defines the type of the cells in a particular CellList. All cells in the CellList will be of the same type. The 6P Cell is a 4-byte field, its RECOMMENDED format is:



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.

4.2.8. 6P ADD Request Format



Version: Set to IANA_6TOP_6P_VERSION.
Type: Set to 6P Request (see Figure 7).
Reserved: Set to 0.
Code: Set to CMD_ADD (see Section 4.2.4).
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.
GAB: Schedule Generation for the cells scheduled from node A to node B.
GBA: Schedule Generation for the cells scheduled from node B to node A.
Metadata: Metadata used as extra signaling to the SF. 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

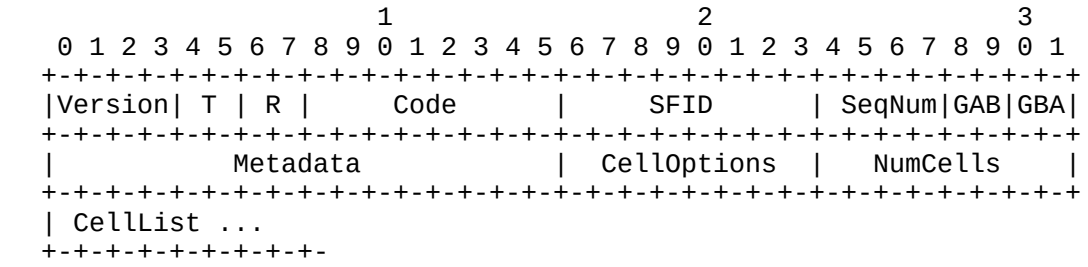
SF, and is out of scope of this document. One example use can be to specify which slotframe to schedule the cells to.

CellOptions: Indicates the type of cells to add. All cells in the CellList for a particular request will use the same CellOption. When different types of cells need to be allocated those need to be handled in separate ADD requests using different CellOptions. The CellOptions is an opaque set of bits, sent unmodified to the SF. The RECOMMENDED format of the CellOptions field is defined in Section 4.2.6. The SF MAY redefine the format or the meaning of the CellOptions field.

NumCells: The number of additional cells the sender wants to schedule to the receiver according.

CellList: A list of 0, 1 or multiple 6P Cells. The CellList is an opaque set of bytes, sent unmodified to the SF. The RECOMMENDED format of each 6P Cell is defined in Section 4.2.7. The SF MAY redefine the format of the CellList field.

4.2.9. 6P DELETE Request Format



Version: Set to IANA_6TOP_6P_VERSION.

Type: Set to 6P Request (see Figure 7).

Reserved: Set to 0.

Code: Set to CMD_DELETE (see Section 4.2.4).

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.

GAB: Schedule Generation for the cells scheduled from node A to node B.

GBA: Schedule Generation for the cells scheduled from node B to node A.

Metadata: Metadata used as extra signaling to the SF. 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. One example use can be to specify which slotframe to delete the cells from.

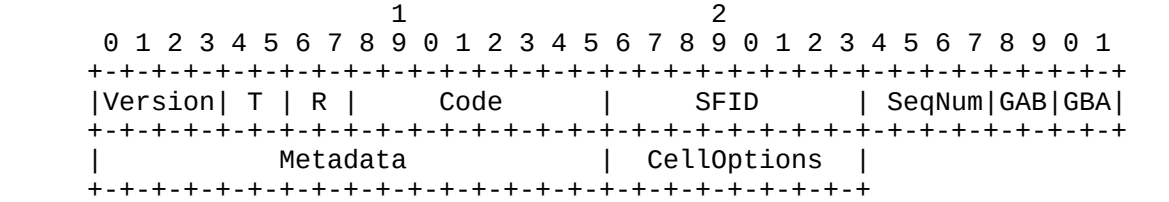
CellOptions: Indicates the type of cells to delete. The CellOptions is an opaque set of bits, sent unmodified to the SF. The RECOMMENDED format of the CellOptions field is defined in

Section 4.2.6. The SF MAY redefine the format or the meaning of the CellOptions field.

NumCells: The number of cells from the specified CellList the sender wants to delete from the schedule of both sender and receiver.

CellList: A list of 0, 1 or multiple 6P Cells. The CellList is an opaque set of bytes, sent unmodified to the SF. The RECOMMENDED format of each 6P Cell is defined in Section 4.2.7. The SF MAY redefine the format of the CellList field.

4.2.10. 6P STATUS Request Format



Version: Set to IANA_6TOP_6P_VERSION.

Type: Set to 6P Request (see Figure 7).

Reserved: Set to 0.

Code: Set to CMD_STATUS (see Section 4.2.4).

SFID: Identifier of the SF to be used by the receiver to handle the message.

SeqNum: Packet identifier to match request and response.

GAB: Schedule Generation for the cells scheduled from node A to node B.

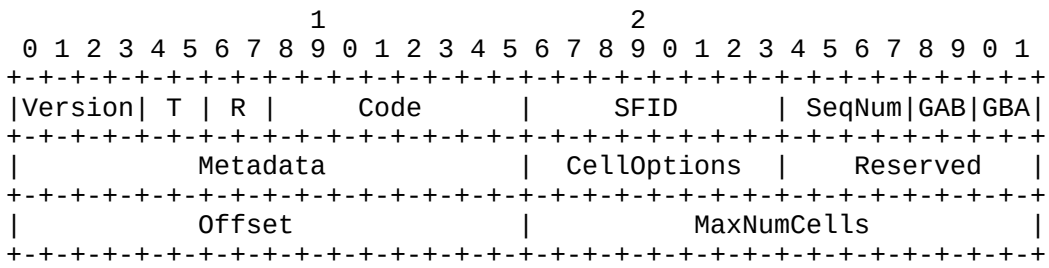
GBA: Schedule Generation for the cells scheduled from node B to node A.

Metadata: Metadata used as extra signaling to the SF. 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. One example use can be to specify which slotframe to get the status from.

CellOptions: Further selects which types of cells to be considered. The CellOptions is an opaque set of bits, sent unmodified to the SF. The RECOMMENDED format and meaning of the CellOptions field is defined in Section 4.2.6. The SF MAY redefine the format or the meaning of the CellOptions field.

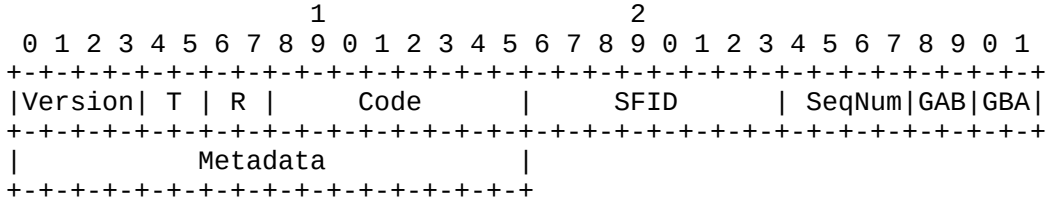
4.2.11. 6P LIST Request Format

The command lists the cells scheduled from node A to node B according to the specified CellOptions.



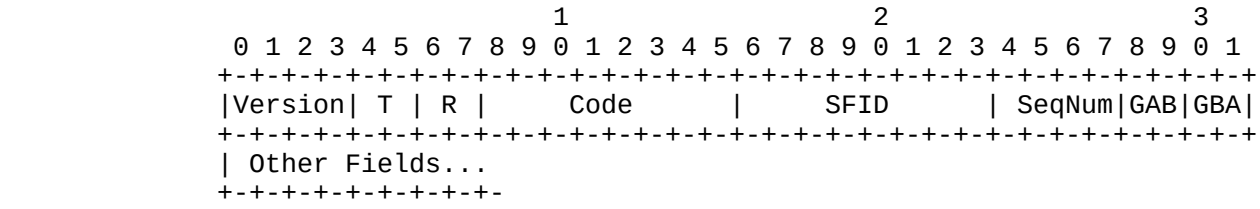
Version: Set to IANA_6TOP_6P_VERSION.
Type: Set to 6P Request (see Figure 7).
Reserved: Set to 0.
Code: Set to CMD_LIST (see Section 4.2.4).
SFID: Identifier of the SF to be used by the receiver to handle the message.
SeqNum: Packet identifier to match request and response.
GAB: Schedule Generation for the cells scheduled from node A to node B.
GBA: Schedule Generation for the cells scheduled from node B to node A.
Metadata: Metadata used as extra signaling to the SF. One example use can be to specify which slotframe to list the cells from. 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.
CellOptions: Further selects which types of cells to be considered. The CellOptions is an opaque set of bits, sent unmodified to the SF. The RECOMMENDED format and meaning of the CellOptions field is defined in Section 4.2.6. The SF MAY redefine the format or the meaning of the CellOptions field.
Reserved: Set to 0.
Offset: The Offset of the first scheduled cell that is requested. The mechanism assumes cells are ordered according to some rule. The ordering rule is defined by the SF.
MaxNumCells: The maximum number of requested cells. Less cells than MaxNumCells can be returned if they do not fit in the packet.

4.2.12. 6P CLEAR Request Format



Version: Set to IANA_6TOP_6P_VERSION.
Type: Set to 6P Request (see Figure 7).
Reserved: Set to 0.
Code: Set to CMD_CLEAR (see Section 4.2.4).
SFID: Identifier of the SF to be used by the receiver to handle the message.
SeqNum: Packet identifier to match request and response.
GAB: Schedule Generation for the cells scheduled from node A to node B.
GBA: Schedule Generation for the cells scheduled from node B to node A.
Metadata: Metadata used as extra signaling to the SF. One example use can be to specify which slotframe to be cleared. 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.

4.2.13. 6P Response Format



Version: Set to IANA_6TOP_6P_VERSION.
Type: Set to 6P Response (see Figure 7).
Reserved: Set to 0.
Code: One of the 6P Return Codes listed in Section 4.2.5.
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.
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.
GAB: Schedule Generation for the cells scheduled from node A to node B.
GBA: Schedule Generation for the cells scheduled from node B to node A.
Other Fields: The contents depends on the Code field in the request, and listed below.

When responding to an ADD, DELETE, LIST request, the "Other Field" contains a list of 0, 1 or multiple 6P Cells. The format of a 6P Cell is defined in Section 4.2.7.

When responding to an STATUS request, the "Other Field" contains the number of cells scheduled between node A and node B that match the CellOptions field, encoded as a 2-octet unsigned integer. This is shown in Figure 12.

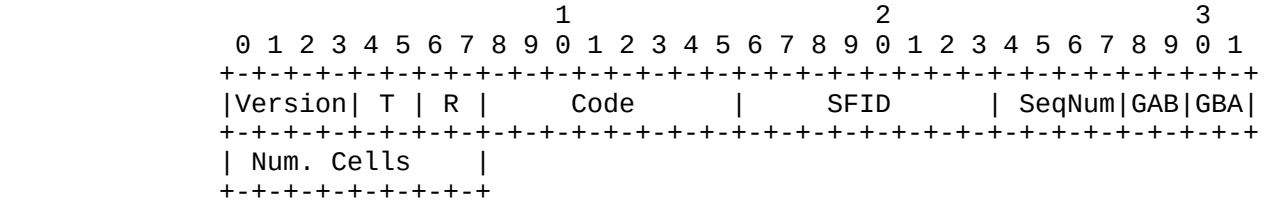


Figure 12

When responding to an CLEAR request, the "Other Field" is empty.

4.2.14. 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, except that the Type field is set to 6P Confirmation (see Figure 7). The same Return Codes are used in both 6P Response and 6P Confirmation messages. The confirmation MUST contain the same SeqNum value as the value in the SeqNum field of the 6P Request and 6P Response of the same transaction.

4.3. Protocol Behavior

We use the topology in Figure 1 for illustration. We assume 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 RC_ERR_VER. 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. 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

RC_ERR_SFID. The Version field in the 6P Response MUST be the same as the Version field in the corresponding 6P Request. In a 3-step transaction, the Version field in the 6P Confirmation MUST match that of the 6P Request and 6P Response in the same transaction.

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 RC_ERR_RESET. A node receiving RC_ERR_RESET MUST abort the transaction and consider it never happened.

Nodes A and B MAY support having two transactions going on at the same time, one in each direction. Similarly, a node MAY support concurrent 6P Transactions from different neighbors. In this case, the cells involved in an 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 R. In case a node does not have enough resources to handle concurrent 6P Transactions from different neighbors it MUST reply with a 6P Response with return code RC_ERR_NORES. In case the requested cells are locked, it MUST reply to that request with a 6P Response with return code RC_ERR_BUSY. The node receiving RC_ERR_BUSY or an RC_ERR_NORES may implement a retry mechanism, as defined by the SF.

4.3.4. Timeout

A timeout happens when the node sending the 6P Request has not received the 6P Response. The timeout should be longer than the longest possible time it can take for the 6P Transaction to finish. The value of the timeout hence depends on the number of cells schedule between the neighbor nodes, on the maximum number of link-layer retransmissions, etc. 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. This rules applies as well to a 6P Confirmation with a SeqNum value different from that of the 6P Request or 6P Response of the same transaction.

4.3.6. Clearing the Schedule

When a 6P CLEAR command is issued from node A to node B, both nodes A and B MUST remove all the cells scheduled between them. That is, node A MUST remove all the cells is has scheduled with B, and node B MUST remove all the cells is has scheduled with A. In a 6P CLEAR command, the generation counters GAB and GBA MUST NOT be checked. That is, their value is "don't care". In particular, even if a schedule generation mismatch is detected, it MUST NOT cause the transaction to abort.

4.3.7. Adding Cells with 2-way Transaction

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

Node A's SF selects NumCandidate>=NumCells cells from its schedule as candidate cells to node B. The CellOptions field specifies the type of this cells. NumCandidate MUST be larger or equal to NumCells. 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 A sends a 6P ADD Request to node B which contains the CellOptions, the value of NumCells and a seleciton of NumCandidate cells in the CellList.

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

Upon receiving the response, node A adds the cells specified in the CellList according to the request CellOptions field.

4.3.8. 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 RC_ERR_RESET. Upon receiving this 6top reply, the initiator of the 6P Transaction MUST consider the 6P Transaction as failed.

4.3.9. Deleting Cells

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

- The nodes delete the cells they agree upon rather than adding them.
- All cells in the CellList MUST already be scheduled between the two nodes and match the CellOptions field. If node A puts cells in its CellList that are not already scheduled between the two nodes and match the CellOptions field, node B replies with a RC_ERR_RESET return code.
- If the CellList in the 6P Request is empty, the SF on the receiving node is free to delete any cell from the sender, as long as it matches the CellOptions field.
- The CellList in a 6P Request (2-step transaction) or 6P Response (3-step transaction) MUST either be empty, contain exactly NumCells cells, or more than NumCells cells. The case where the CellList is not empty but contains less than NumCells cells is not supported.

4.3.10. Listing Cells

When a node A issues a LIST command, it specifies:

- Through the CellOptions field, the type of cells to list, according to Section 4.2.6.
- Through the Offset field, the offset of the first CellOptions type cell to be present in the returned list. The cell ordering policy is defined by the SF.
- Through the MaxNumCells field, the maximum number of cells to be present in the response.

When receiving a LIST command, node B returns the cells in its schedule that match the CellOptions field as specified in Section 4.2.6 The RECOMMENDED format of each 6P Cell is defined in Section 4.2.7. The SF MAY redefine the format of the CellList field.

When node B receives a LIST request, the returned CellList in the 6P Response contains between 1 and MaxNumCells cells, starting from the specified Offset, as many as fit in the frame. Node B MUST return at least one cell, unless the specified Offset is beyond the end of B's cell list in its schedule. If node B has less than Offset cells of CellOptions type, the CellList it returns is empty.

4.3.11. Generation Management

For each neighbor, a node maintains 2 two-bit generation numbers. These numbers are variables internal to the node.

- GTX is the generation number for the transmission cells to the neighbor.
- GRX is the generation number for the receive cells from the neighbor.

4.3.11.1. Incrementing GTX and GRX

GTX and GRX are 2-bit variables. Their possible values are:

Value	Meaning
0b00	Clear or never scheduled
0b01-0b10	Lollipop Counter values
0b11	Reserved

Figure 13: Possible values of the GRX and GTX generation numbers.

GTX and GRX are set to 0 upon initialization, and after a 6P CLEAR command. GTX and GRX are incremented by 1 after each time a cell with that neighbor is added/deleted from the schedule (e.g. after a successful 6P ADD or 6P DELETE transactions). The value rolls from 0b10 to 0b01. This results in a lollipop counter with 0x00 as the start value, and 0b01 and 0b10 the count values.

4.3.11.2. Setting GAB and GBA fields

Each 6P message contains a GAB and a GBA field, used to indicate the current generation counters of the node transmitting the message. The value of the GAB and GBA fields MUST be set according to the following rules:

- When node A sends a 6P Request or 6P confirmation to node B, node A sets GAB to its GTX and GBA to its GRX.
- When node B sends a 6P Response to node A, node B sets GAB to its GRX and GBA to its GTX.

4.3.11.3. Detecting and Handling Schedule Generation Inconsistencies

Upon receiving a 6P message, a node MUST do the following checks:

- When node B receives a 6P Request of 6P confirmation from node A, it verifies that GAB==GRX and GBA==GTX.
- When node A receives a 6P Response from node B, it verifies that GAB==GTX and GBA==GRX.

If any of these comparisons is false, the node has detected a schedule generation inconsistency.

When a schedule generation inconsistency is detected:

- If the code of the 6P Request is different from CMD_CLEAR, the node MUST reply with error code RC_ERR_GEN.
- If the code of the 6P Request is CMD_CLEAR, the schedule generation inconsistency MUST be ignored.

It is up to the Scheduling Function to define the action to take when an schedule generation inconsistency is detected. The RECOMMENDED action is to issue a 6P CLEAR command.

4.3.12. Handling error responses

A return code with a name starting with "RC_ERR" in Figure 9 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. Defining what to do after an error has occurred is out of scope of this document. The SF defines what to do after an error has occurred.

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 14: SFID range.

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

5.2. Requirements for an SF

The specification for an SF

- MUST specify an identifier for that SF.
- MUST specify the rule for a node to decide when to add/delete one or more cells to a neighbor.
- MUST specify the rule for a Transaction source to select cells to add to the CellList field in the 6P ADD Request.
- MUST specify the rule for a Transaction destination to select cells from CellList to add to its schedule.
- MUST specify a value for the 6P Timeout, or a rule/equation to calculate it.
- MUST specify a meaning for the "Metadata" field in the 6P ADD Request.
- MUST specify the behavior of a node when it boots.
- MUST specify what to do after an error has occurred (either the node sent a 6P Response with an error code, or received one).
- 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.
- SHOULD clearly state the application domain the SF is created for.
- SHOULD contain examples which highlight normal and error scenarios.
- SHOULD contain a list of current implementations, at least during the I-D state of the document, per [RFC6982].
- SHOULD contain a performance evaluation of the scheme, possibly through references to external documents.
- MAY redefine the format of the CellList field.

- MAY redefine the format of the CellOptions field.
- MAY redefine the meaning of the CellOptions field.

5.3. Recommended Structure of an SF Specification

The following section structure for a SF document is RECOMMENDED:

- Introduction
- Scheduling Function Identifier
- Rules for Adding/Deleting Cells
- Rules for CellList
- 6P Timeout Value
- Meaning of the Metadata Field
- Node Behavior at Boot
- 6P Error Handling
- Examples
- Implementation Status
- Security Considerations
- 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/6lo plugtests: 6P was one of the protocols addressed during the ETSI 6TiSCH #3 plugtests organized on 15-17 July 2016 in Berlin, Germany. 15 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/1077-6tisch-6lo-plugtests.

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

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_STATUS
- TODO: IANA_6TOP_CMD_LIST
- TODO: IANA_6TOP_CMD_CLEAR
- TODO: IANA_6TOP_RC_SUCCESS

- TODO: IANA_6TOP_RC_ERR_VER
- TODO: IANA_6TOP_RC_ERR_SFID
- TODO: IANA_6TOP_RC_ERR_GEN
- TODO: IANA_6TOP_RC_ERR_BUSY
- TODO: IANA_6TOP_RC_ERR_NORES
- TODO: IANA_6TOP_RC_ERR_RESET
- 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>.

[I-D.kivinen-802-15-ie] Kivinen, T. and P. Kinney, "IEEE 802.15.4 Information Element for IETF", draft-kivinen-802-15-ie-04 (work in progress), October 2016.

[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-16 (work in progress), June 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] Changelog

- draft-ietf-6tisch-6top-protocol-03
  - Added a reference to [I-D.kivinen-802-15-ie].
  - Added the Type field.
  - Editorial changes (figs, typos, ...)
- draft-ietf-6tisch-6top-protocol-02
  - Rename COUNT to STATUS
  - Split LIST to LIST AB and LIST BA
  - Added generation counters and describing generation tracking of the schedule
  - Editorial changes (figs, typos, ...)
- draft-ietf-6tisch-6top-protocol-01
  - Clarifying locking of resources in concurrent transactions
  - Clarifying return of RC_ERR_BUSY in case of concurrent transactions without enough resources
- draft-ietf-6tisch-6top-protocol-00
  - Informational to Std track
- 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
Internet-Draft
Intended status: Standards Track
Expires: October 1, 2018

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

March 30, 2018

6top Protocol (6P)
draft-ietf-6tisch-6top-protocol-11

Abstract

This document defines the 6top Protocol (6P), which enables distributed scheduling in 6TiSCH networks. 6P allows neighbor nodes to add/delete TSCH cells to one another. 6P is part of the 6TiSCH Operation Sublayer (6top), the next higher layer to the IEEE Std 802.15.4 TSCH medium access control layer. The 6top layer terminates the 6top Protocol defined in this document, and runs one of more 6top Scheduling Function(s). A 6top Scheduling Function (SF) decides when to add/delete cells, and triggers 6P Transactions. This document lists the requirements for an SF, but leaves the definition of SFs out of scope.

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

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 October 1, 2018.

Table of Contents

1. Introduction 3
2. 6TiSCH Operation Sublayer (6top) 4
  2.1. Hard/Soft Cells 5
  2.2. Using 6P with the Minimal 6TiSCH Configuration 5
3. 6top Protocol (6P) 6
  3.1. 6P Transactions 6
    3.1.1. 2-step 6P Transaction 7
    3.1.2. 3-step 6P Transaction 9
  3.2. Message Format 11
    3.2.1. 6top Information Element (IE) 11
    3.2.2. Generic 6P Message Format 11
    3.2.3. 6P CellOptions 12
    3.2.4. 6P CellList 14
  3.3. 6P Commands and Operations 15
    3.3.1. Adding Cells 15
    3.3.2. Deleting Cells 17
    3.3.3. Relocating Cells 18
    3.3.4. Counting Cells 24
    3.3.5. Listing Cells 25
    3.3.6. Clearing the Schedule 27
    3.3.7. Generic Signaling Between SFs 28
  3.4. Protocol Functional Details 28
    3.4.1. Version Checking 28
    3.4.2. SFID Checking 29
    3.4.3. Concurrent 6P Transactions 29
    3.4.4. 6P Timeout 30
    3.4.5. Aborting a 6P Transaction 30
    3.4.6. SeqNum Management 30
    3.4.7. Handling Error Responses 36
  3.5. Security 36
4. Requirements for 6top Scheduling Functions (SF) 36

4.1. SF Identifier (SFID) 36
4.2. Requirements for an SF 36
5. Security Considerations 37
6. IANA Considerations 37
6.1. IETF IE Subtype '6P' 37
6.2. 6TiSCH parameters sub-registries 38
6.2.1. 6P Version Numbers 38
6.2.2. 6P Message Types 38
6.2.3. 6P Command Identifiers 39
6.2.4. 6P Return Codes 40
6.2.5. 6P Scheduling Function Identifiers 41
6.2.6. 6P CellOptions bitmap 42
7. References 43
7.1. Normative References 43
7.2. Informative References 43
Appendix A. Recommended Structure of an SF Specification 44
Authors' Addresses 44

1. Introduction

All communication in a 6TiSCH network is orchestrated by a schedule [RFC7554]. The schedule is composed of cells, each identified by a [slotOffset,channelOffset]. This specification defines the 6top Protocol (6P), terminated by the 6TiSCH Operation sublayer (6top). 6P allows a node to communicate with a neighbor node to add/delete TSCH cells to each other. This results in distributed schedule management in a 6TiSCH network. The 6top layer terminates the 6top Protocol, and runs one or more 6top Scheduling Functions (SFs) that decide when to add/delete cells and trigger 6P Transactions. The SF is out of scope of this document but the requirements for an SF are defined here.

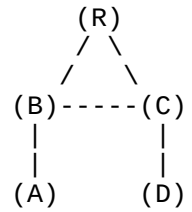


Figure 1: A simple 6TiSCH network.

The example network depicted in Figure 1 is used to describe the interaction between nodes. We consider the canonical case where node "A" issues 6P requests to node "B". We keep this example throughout this document. Throughout the document, node A always represents the node that issues a 6P request; node B the node that receives this request.

We consider that node A monitors the communication cells it has in its schedule to node B:

- If node A determines that the number of link-layer frames it is sending to node B per unit of time exceeds the capacity offered by the TSCH cells it has scheduled to node B, it triggers a 6P Transaction with node B to add one or more cells to the TSCH schedule of both nodes.
- If the traffic is lower than the capacity, node A triggers a 6P Transaction with node B to delete one or more cells in the TSCH schedule of both nodes.
- Node A MAY also monitor statistics to determine whether collisions are happening on a particular cell to node B. If this feature is enabled, node A communicates with node B to "relocate" the cell which suffers collisions to a different [slotOffset,channelOffset] location in the TSCH schedule.

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. 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. At least one SF MUST be running. The SFID field contained in all 6P messages allows a node to invoke the appropriate SF on a per-6P Transaction basis.

Section 2 describes the 6TiSCH Operation Sublayer (6top). Section 3 defines the 6top Protocol (6P). Section 4 provides guidelines on how to define an SF.

2. 6TiSCH Operation Sublayer (6top)

As depicted in Figure 2, the 6TiSCH Operation Sublayer (6top) is the next higher layer to the IEEE Std 802.15.4 TSCH medium access control (MAC) layer [IEEE802154]. We use "802.15.4" as a short version of "IEEE Std 802.15.4" in this document.

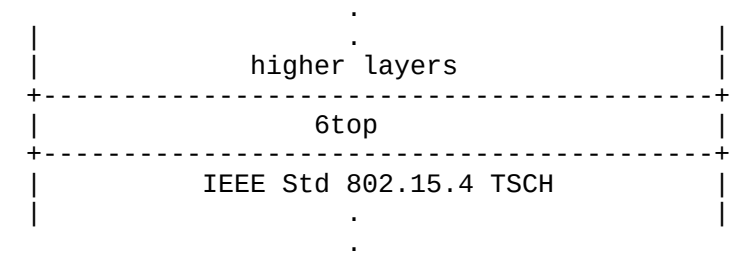


Figure 2: The 6top sublayer in the protocol stack.

The roles of the 6top sublayer are to:

- Terminate the 6top Protocol (6P), which allows neighbor nodes to communicate to add/delete cells to one another.
- Run one or multiple 6top Scheduling Functions (SFs), which define the rules that decide when to add/delete cells.

2.1. Hard/Soft Cells

Each cell in the schedule is either "hard" or "soft":

- a soft cell can be read, added, deleted or updated by 6top.
- 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 schedule [RFC8180].

2.2. Using 6P with the Minimal 6TiSCH Configuration

6P MAY be used alongside the Minimal 6TiSCH Configuration [RFC8180]. In this case, it is RECOMMENDED to use 2 slotframes, as depicted in Figure 3:

- Slotframe 0 is used for traffic defined in the Minimal 6TiSCH Configuration. In Figure 3, this slotframe is 5 slots long, but the slotframe can be shorter or longer.
- 6P allocates cells from Slotframe 1. In Figure 3, Slotframe 1 is 10 slots long, but the slotframe can be shorter or longer.

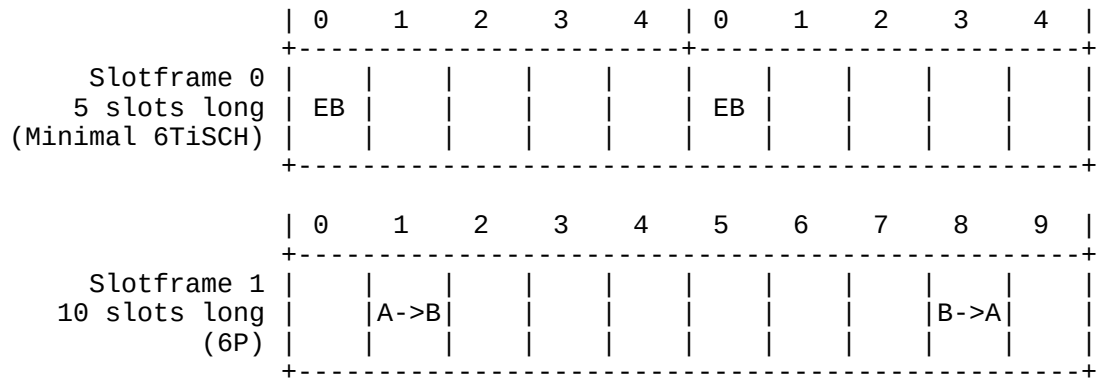


Figure 3: 2-slotframe structure when using 6P alongside the Minimal 6TiSCH Configuration.

The Minimal 6TiSCH Configuration cell SHOULD be allocated from a slotframe of higher priority than the slotframe used by 6P for dynamic cell allocation. This way, dynamically allocated cells cannot "mask" the cells used by the Minimal 6TiSCH Configuration. 6top MAY support additional slotframes; how to use additional slotframes is out of scope for this document.

3. 6top Protocol (6P)

The 6top Protocol (6P) enables two neighbor nodes to add/delete/relocate cells in their TSCH schedule. Conceptually, two neighbor nodes "negotiate" the location of the cells to add, delete, or relocate in their TSCH schedule.

3.1. 6P Transactions

We call "6P Transaction" a complete negotiation between two neighbor nodes. A 6P Transaction starts when a node wishes to add/delete/relocate one or more cells with one of its neighbors. A 6P Transaction ends when the cell(s) have been added/deleted/relocated in the schedule of both nodes, or when the 6P Transaction has failed.

6P messages exchanged between nodes A and B during a 6P Transaction SHOULD be exchanged on non-shared unicast cells ("dedicated" cells) between A and B. If no dedicated cells are scheduled between nodes A and B, shared cells MAY be used.

Keeping consistency between the schedules of the two neighbor nodes is important. A loss of consistency can cause loss of connectivity. One example is when node A has a transmit cell to node B, but node B does not have the corresponding reception cell. To verify

consistency, neighbor nodes maintain a Sequence Number (SeqNum). Neighbor nodes exchange the SeqNum as part of each 6P Transaction to detect possible inconsistency. This mechanism is explained in Section 3.4.6.2.

An implementation MUST include a mechanism to associate each scheduled cell with the SF that scheduled it. This mechanism is implementation-specific and out of scope of this document.

A 6P Transaction can consist of 2 or 3 steps. A 2-step transaction is used when node A selects the cells to be allocated. A 3-step transaction is used when node B selects the cells to be allocated. An SF MUST specify whether to use 2-step transactions, 3-step transactions, or both.

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

3.1.1. 2-step 6P Transaction

Figure 4 shows an example 2-step 6P Transaction. In a 2-step transaction, node A selects the candidate cells. Several elements are left out to simplify understanding.

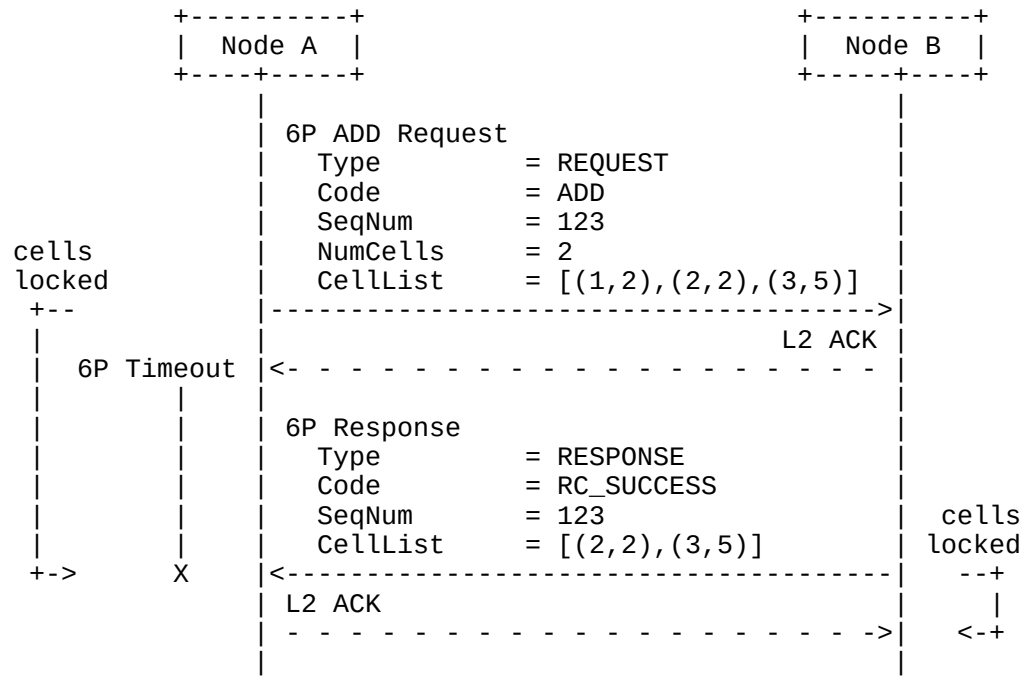


Figure 4: An example 2-step 6P Transaction.

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

1. The SF running on node A determines that 2 extra cells need to be scheduled to node B.
2. The SF running on node A selects 3 candidate cells. Node A locks the candidate cells in it schedule until it receives a 6P response.
3. 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 cells (the "CellList" value). Each cell in the CellList is a [slotOffset,channelOffset] tuple. This 6P ADD Request is link-layer acknowledged by node B (labeled "L2 ACK" in Figure 4).
4. After having successfully sent the 6P ADD Request (i.e. receiving the link-layer acknowledgment), node A starts a 6P Timeout to abort the 6P Transaction in case no response is received from node B.
5. The SF running on node B selects 2 out of the 3 cells from the CellList of the 6P ADD Request. Node B locks those cells in it schedule until the transmission is successfull (i.e. node B receives a link-layer ACK from node A). Node B sends back a 6P

Response to node A, indicating the cells it has selected. The response is link-layer acknowledged by node A.

6. Upon completion of this 6P Transaction, 2 cells from A to B have been added to the TSCH schedule of both nodes A and B.
7. An inconsistency in the schedule can happen if the 6P Timeout expires when the 6P Response is in the air, if the last link-layer ACK for the 6P Response is lost, or if one of the nodes is power cycled. 6P provides an inconsistency detection mechanism described in Section 3.4.6.1 to cope with such situations.

3.1.2. 3-step 6P Transaction

Figure 5 shows an example 3-step 6P Transaction. In a 3-step transaction, node B selects the candidate cells. Several elements are left out to simplify understanding.

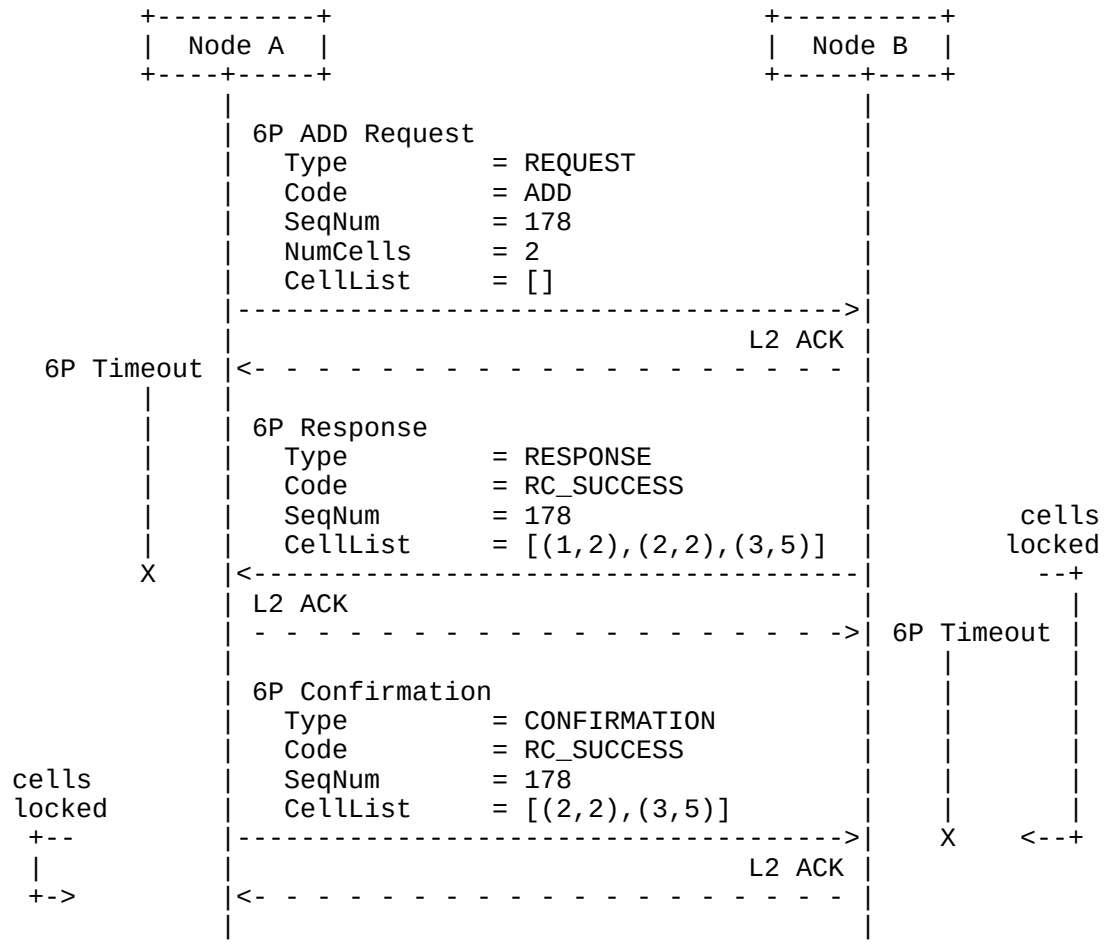


Figure 5: An example 3-step 6P Transaction.

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

1. The SF running on node A determines that 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 an empty "CellList". This 6P ADD Request is link-layer acknowledged by node B.
3. After having successfully sent the 6P ADD Request, node A starts a 6P Timeout to abort the transaction in case no 6P Response is received from node B.
4. The SF running on node B selects 3 candidate cells, and locks them. Node B sends back a 6P Response to node A, indicating the

3 cells it has selected. The response is link-layer acknowledged by node A.

5. After having successfully sent the 6P Response, node B starts a 6P Timeout to abort the transaction in case no 6P Confirmation is received from node A.
6. The SF running on node A selects 2 cells from the CellList field in the 6P Response, and locks those. Node A sends back a 6P Confirmation to node B, indicating the cells it selected. The confirmation is link-layer acknowledged by node B.
7. Upon completion of the 6P Transaction, 2 cells from A to B have been added to the TSCH schedule of both nodes A and B; other cells are unlocked.
8. An inconsistency in the schedule can happen if the 6P Timeout expires when the 6P Confirmation is in the air, if the last link-layer ACK for the 6P Confirmation is lost, or if one of the nodes is power cycled. 6P provides an inconsistency detection mechanism described in Section 3.4.6.1 to cope with such situations.

3.2. Message Format

3.2.1. 6top Information Element (IE)

6P messages travel over a single hop. 6P messages are carried as payload of an 802.15.4 Payload Information Element (IE) [IEEE802154]. The messages are encapsulated within the Payload IE Header. The Group ID is set to the IETF IE value defined in [RFC8137]. The content is encapsulated by a SubType ID, as defined in [RFC8137].

Since 6P messages are carried in IEs, IEEE bit/byte ordering applies. Bits within each field in the 6top IE are numbered from 0 (leftmost and least significant) to k-1 (rightmost and most significant), where the length of the field is k bits. Fields that are longer than a single octet are copied to the packet in the order from the octet containing the lowest numbered bits to the octet containing the highest numbered bits (little endian).

This document defines the "6top IE", a SubType of the IETF IE defined in [RFC8137], with subtype ID IANA_6TOP_SUBIE_ID. The SubType Content of the "6top IE" is defined in Section 3.2.2. The length of the "6top IE" content is variable.

3.2.2. Generic 6P Message Format

All 6P messages follow the generic format shown in Figure 6.

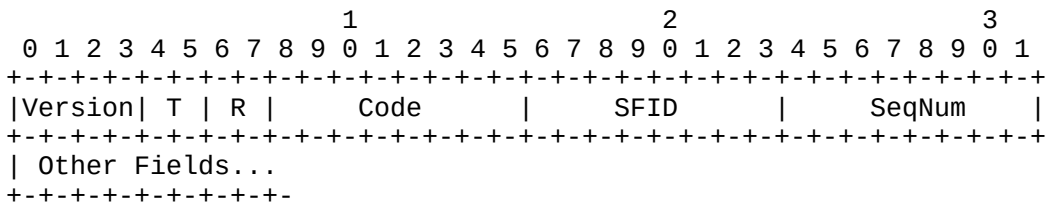


Figure 6: Generic 6P Message Format.

6P Version (Version): The version of the 6P protocol. Only version 0 is defined in this document. Future specifications MAY define further versions of the 6P protocol.

Type (T): Type of message. The message types are defined in Section 6.2.2.

Reserved (R): Reserved bits. These two bits SHOULD be set to zero when sending the message, and MUST be ignored upon reception.

Code: The Code field contains a 6P Command Identifier when the 6P message is of Type REQUEST. Section 6.2.3 lists the 6P command identifiers. The Code field contains a 6P Return Code when the 6P message is of Type RESPONSE or CONFIRMATION. Section 6.2.4 lists the 6P Return Codes. The same return codes are used in both 6P Response and 6P Confirmation messages.

6top Scheduling Function Identifier (SFID): The identifier of the SF to use to handle this message. The SFID is defined in Section 4.1.

SeqNum: Sequence number associated with the 6P Transaction, used to match the 6P Request, 6P Response and 6P Confirmation of the same 6P Transaction. The value of SeqNum MUST be different at each new 6P Request issued to the same neighbor. The SeqNum is also used to ensure consistency between the schedules of the two neighbors. Section 3.4.6 details how the SeqNum is managed.

Other Fields: The list of other fields and how they are used is detailed in Section 3.3.

3.2.3. 6P CellOptions

An 8-bit 6P CellOptions bitmap is present in the following 6P requests: ADD, DELETE, COUNT, LIST, RELOCATE.

- In the 6P ADD request, the 6P CellOptions bitmap is used to specify what type of cell to add.
- In the 6P DELETE request, the 6P CellOptions bitmap is used to specify what type of cell to delete.
- In the 6P COUNT and the 6P LIST requests, the 6P CellOptions bitmap is used as a selector of a particular type of cells.

o In the 6P RELOCATE request, the 6P CellOptions bitmap is used to specify what type of cell to relocate.

The contents of the 6P CellOptions bitmap apply to all elements in the CellList field. Section 6.2.6 contains the RECOMMENDED format of the 6P CellOptions bitmap. Figure 7 contains the RECOMMENDED meaning of the 6P CellOptions bitmap for the 6P ADD, DELETE, RELOCATE requests. Figure 8 contains the RECOMMENDED meaning of the 6P CellOptions bitmap for the 6P COUNT, LIST requests.

Note: assuming node A issues the 6P command to node B.

CellOptions Value	the type of cells B adds/deletes/relocates to its schedule when receiving a 6P ADD/DELETE/RELOCATE Request from A.
TX=0,RX=0,S=0	Does not apply. RC_ERR is returned.
TX=1,RX=0,S=0	add/delete/relocate RX cells at B (TX cells at A)
TX=0,RX=1,S=0	add/delete/relocate TX cells at B (RX cells at A)
TX=1,RX=1,S=0	add/delete/relocate TX|RX cells at B (and at A)
TX=0,RX=0,S=1	Does not apply. RC_ERR is returned.
TX=1,RX=0,S=1	add/delete/relocate RX|SHARED cells at B (TX|SHARED cells at A)
TX=0,RX=1,S=1	add/delete/relocate TX|SHARED cells at B (RX|SHARED cells at A)
TX=1,RX=1,S=1	add/delete/relocate TX|RX|SHARED cells at B (and at A)

Figure 7: Meaning of the 6P CellOptions bitmap for the 6P ADD, DELETE, RELOCATE requests.

Note: assuming node A issues the 6P command to node B.

CellOptions Value	the type of cells B selects from its schedule when receiving a 6P COUNT or LIST Request from A, from all the cells B has scheduled with A
TX=0,RX=0,S=0	all cells
TX=1,RX=0,S=0	all cells marked as RX only
TX=0,RX=1,S=0	all cells marked as TX only
TX=1,RX=1,S=0	all cells marked as TX and RX only
TX=0,RX=0,S=1	all cells marked as SHARED (regardless of TX, RX)
TX=1,RX=0,S=1	all cells marked as RX and SHARED only
TX=0,RX=1,S=1	all cells marked as TX and SHARED only
TX=1,RX=1,S=1	all cells marked as TX and RX and SHARED

Figure 8: Meaning of the 6P CellOptions bitmap for the 6P COUNT, LIST requests.

The CellOptions is an opaque set of bits, sent unmodified to the SF. The SF MAY redefine the format and meaning of the CellOptions field.

3.2.4. 6P CellList

A CellList field MAY be present in a 6P ADD Request, a 6P DELETE Request, a 6P RELOCATE Request, a 6P Response, or a 6P Confirmation. It is composed of a concatenation of zero, one or more 6P Cells as defined in Figure 9. The content of the CellOptions field specifies the options associated with all cells in the CellList. This necessarily means that the same options are associated with all cells in the CellList.

A 6P Cell is a 4-byte field, its RECOMMENDED format is:

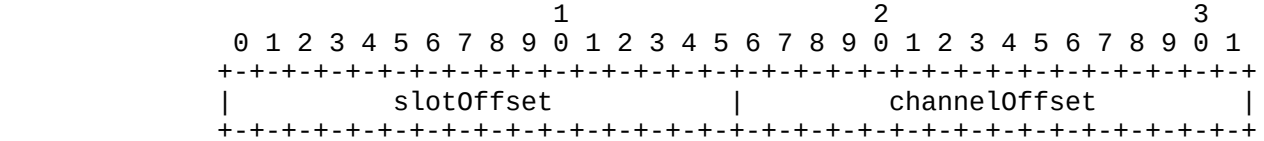


Figure 9: 6P Cell Format.

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 length of the CellList field is implicit, and determined by the IE Length field, present in the Payload IE header as defined by the IEEE 802.15.4 standard [IEEE802154]. The SF MAY redefine the format of the CellList field.

3.3. 6P Commands and Operations

3.3.1. Adding Cells

Cells are added by using the 6P ADD command. The Type field (T) is set to REQUEST. The Code field is set to ADD. Figure 10 defines the format of a 6P ADD Request.

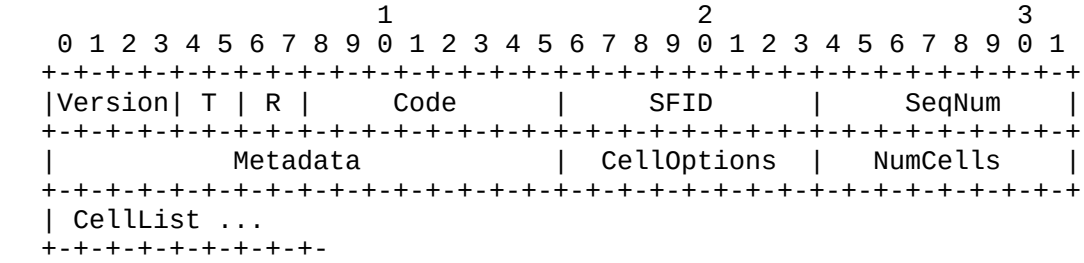


Figure 10: 6P ADD Request Format.

Metadata: Used as extra signaling to the SF. The contents of the Metadata field is an opaque set of bytes passed unmodified to the SF. The meaning of this field depends on the SF, and is out of scope of this document. For example, Metadata can specify in which slotframe to add the cells.
CellOptions: Indicates the options to associate with the cells to be added. If more than one cell is added (NumCells>1), the same options are associated with each one. This necessarily means that, if node A needs to add multiple cells with different options, it needs to initiate multiple 6P ADD Transactions.
NumCells: The number of additional cells node A wants to schedule to node B.
CellList: A list of 0, 1 or multiple candidate cells. Its length is implicit and determined by the Length field of the Payload IE header.

Figure 11 defines the format of a 6P ADD Response and Confirmation.

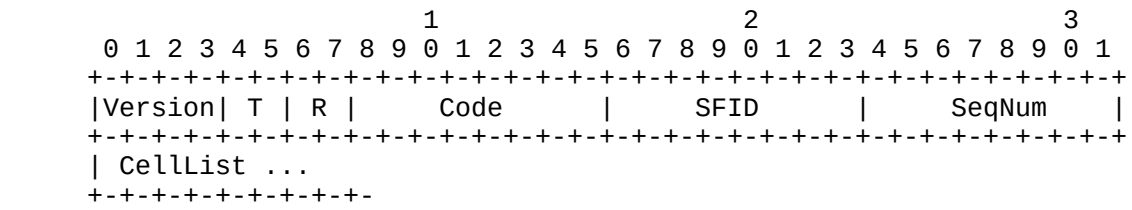


Figure 11: 6P ADD Response and Confirmation Formats.

CellList: A list of 0, 1 or multiple 6P Cells.

Consider the topology in Figure 1 where the SF on node A decides to add NumCells cells to node B.

Node A's SF selects NumCandidate cells from its schedule. These are cells that are candidates to be scheduled with node B. The CellOptions field specifies the type of these cells. NumCandidate MUST be larger or equal to NumCells. How many cells node A selects (NumCandidate) and how that selection is done is specified in the SF and out of scope of this document. Node A sends a 6P ADD Request to node B which contains the CellOptions, the value of NumCells, and a selection of NumCandidate cells in the CellList. In case the NumCandidate cells do not fit in a single packet, this operation MUST be split into multiple independent 6P ADD Requests, each for a subset of the number of cells that eventually need to be added. In case of a 3-step transaction, the SF is reponsible of ensuring that the returned candidate celllist fit in the 6P Response packet.

Upon receiving the request, node B checks whether the cellOptions are set to a legal value as noted by Figure 7. If this is not the case, a Response with code RC_ERR is returned. Otherwise, node B's SF verifies which of the cells in the CellList it can install in node B's schedule, following the specified CellOptions field. How that selection is done is specified in the SF and out of scope of this document. The verification can succeed (NumCells cells from the CellList can be used), fail (none of the cells from the CellList can be used), or partially succeed (less than NumCells cells from the CellList can be used). In all cases, node B MUST send a 6P Response with return code set to RC_SUCCESS, and which specifies the list of cells that were scheduled following the CellOptions field. That can contain NumCells elements (succeed), 0 elements (fail), or between 0 and NumCells elements (partially succeed).

Upon receiving the response, node A adds the cells specified in the CellList according to the CellOptions field.

3.3.2. Deleting Cells

Cells are deleted by using the 6P DELETE command. The Type field (T) is set to REQUEST. The Code field is set to DELETE. Figure 12 defines the format of a 6P DELETE Request.

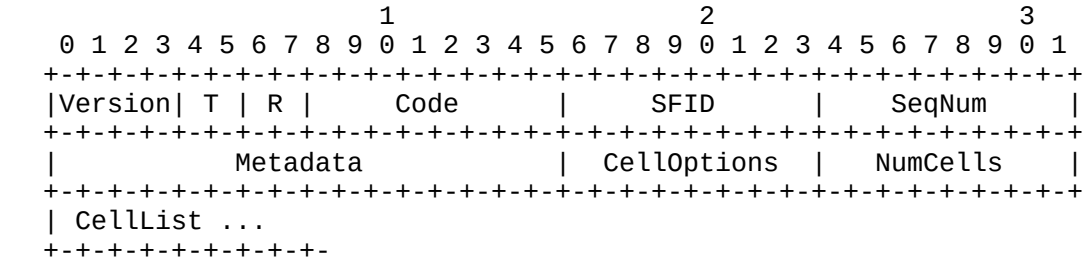


Figure 12: 6P DELETE Request Format.

Metadata: Same usage as for the 6P ADD command, see Section 3.3.1. Its format is the same as that in the 6P ADD command, but its content could be different.

CellOptions: Indicates the options that need to be associated to the cells to delete. Only cells matching the CellOptions can are deleted.

NumCells: The number of cells from the specified CellList the sender wants to delete from the schedule of both sender and receiver.

CellList: A list of 0, 1 or multiple 6P Cells. Its length is determined by the Length field of the Payload IE header.

Figure 13 defines the format of a 6P DELETE Response and Confirmation.

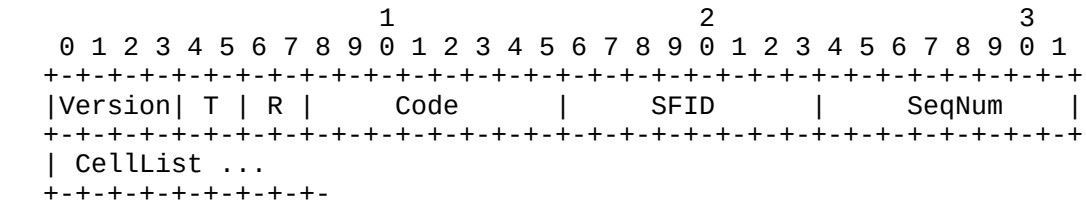


Figure 13: 6P DELETE Response and Confirmation Formats.

CellList: A list of 0, 1 or multiple 6P Cells.

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

- The nodes delete the cells they agree upon rather than adding them.
- All cells in the CellList MUST already be scheduled between the two nodes and MUST match the CellOptions field. If node A puts cells in its CellList that are not already scheduled between the two nodes and match the CellOptions field, node B MUST reply with a RC_ERR_CELLLIST return code.
- If the CellList in the 6P Request is empty, the SF on the receiving node SHOULD delete any cell from the sender, as long as it matches the CellOptions field.
- The CellList in a 6P Request (2-step transaction) or 6P Response (3-step transaction) MUST either be empty, contain exactly NumCells cells, or more than NumCells cells. The case where the CellList is not empty but contains less than NumCells cells is not supported.

3.3.3. Relocating Cells

Cell relocation consists in moving a cell to a different [slotOffset,channelOffset] location in the schedule. The Type field (T) is set to REQUEST. The Code is set to RELOCATE. Figure 14 defines the format of a 6P RELOCATE Request.

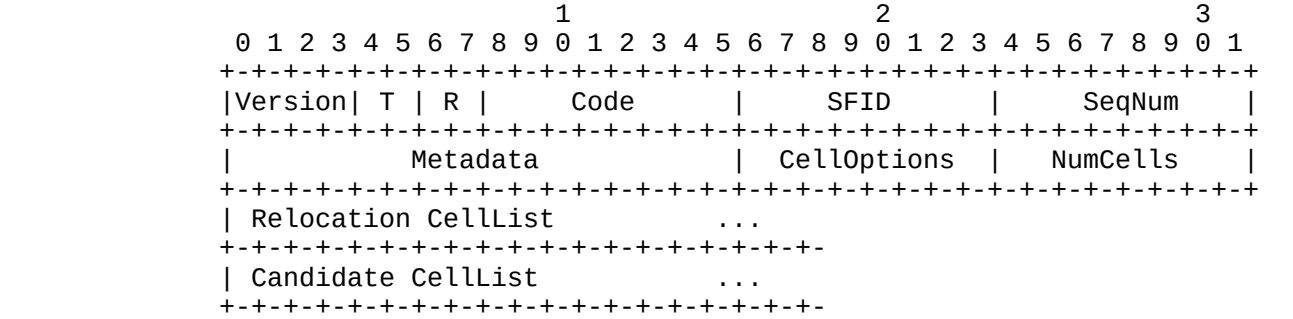


Figure 14: 6P RELOCATE Request Format.

Metadata: Same usage as for the 6P ADD command, see Section 3.3.1.
CellOptions: Indicates the options that need to be associated with cells to be relocated.
NumCells: The number of cells to relocate, which MUST be equal or greater than 1.
Relocation CellList: The list of NumCells 6P Cells to relocate.
Candidate CellList: A list of NumCandidate candidate cells for node B to pick from. NumCandidate MUST be 0, equal to NumCells, or greater than NumCells. Its length is determined by the Length field of the Payload IE header.

In a 2-step 6P RELOCATE Transaction, node A specifies both the cells it needs to relocate, and the list of candidate cells to relocate to. The Relocation CellList MUST contain exactly NumCells entries. The Candidate CellList MUST contain at least NumCells entries (that is NumCandidate>=NumCells).

In a 3-step 6P RELOCATE Transaction, node A specifies only the cells it needs to relocate, but not the list of candidate cells to relocate to. The Candidate CellList MUST therefore be empty.

Figure 15 defines the format of a 6P RELOCATE Response and Confirmation.

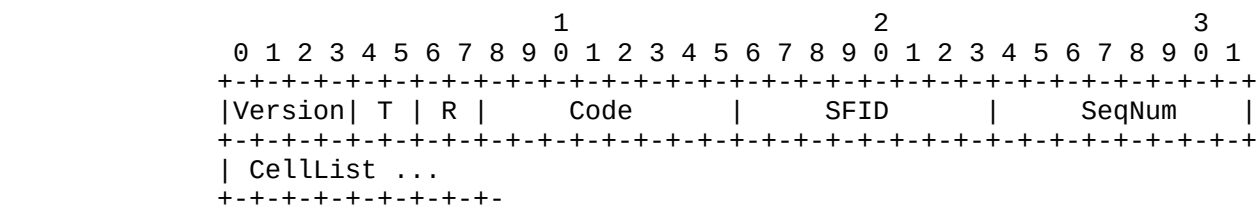


Figure 15: 6P RELOCATE Response and Confirmation Formats.

CellList: A list of 0, 1 or multiple 6P Cells.

Node A's SF wants to relocate NumCells particular cells. Node A creates a 6P RELOCATE Request, and indicates the cells it wants to relocate in the Relocation CellList. It also selects NumCandidate cells from its schedule as candidate cells to relocate the cells to, and puts those in the Candidate CellList. The CellOptions field specifies the type of the cell(s) to relocate. NumCandidate MUST be larger or equal to NumCells. 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 A sends the 6P RELOCATE Request to node B.

Upon receiving the request, Node B checks if the length of the Candidate CellList is larger or equal to NumCells. Node B's SF verifies that all the cells in the Relocation CellList are indeed scheduled with node A, and are associate the options specified in the CellOptions field. If that check fails, node B MUST send a 6P Response to node A with return code RC_ERR_CELLLIST. If that check passes, node B's SF verifies which of the cells in the Candidate CellList it can install in its schedule. How that selection is done is specified in the SF and out of scope of this document. That verification on Candidate CellList can succeed (NumCells cells from the Candidate CellList can be used), fail (none of the cells from the Candidate CellList can be used) or partially succeed (less than

NumCells cells from the Candidate CellList can be used). In all cases, node B MUST send a 6P Response with return code set to RC_SUCCESS, and which specifies the list of cells that will be re-scheduled following the CellOptions field. That can contain NumCells elements (succeed), 0 elements (fail), between 0 and NumCells elements (partially succeed). If N < NumCells cells appear in the CellList, this means first N cells in the Relocation CellList have to be relocated, the remainder have not.

Upon receiving the response with Code RC_SUCCESS, node A relocates the cells specified in Relocation CellList of its RELOCATE Request to the new locations specified in the CellList of the 6P Response, in the same order. In case the received Response Code is RC_ERR_CELLLIST, the transaction is aborted and no cell is relocated. Upon receiving the L2 Ack of the 6P Response or 6P Confirmation in case of a 3-step transaction, Node B relocates the selected cells.

The SF SHOULD take into account situations such as the relocation of all cells at a time between two nodes. If the operation fails the schedules of both nodes may completely diverge. Is up to the SF the handling of such situation.

Figure 16 shows an example of a successful 2-step 6P RELOCATION Transaction.

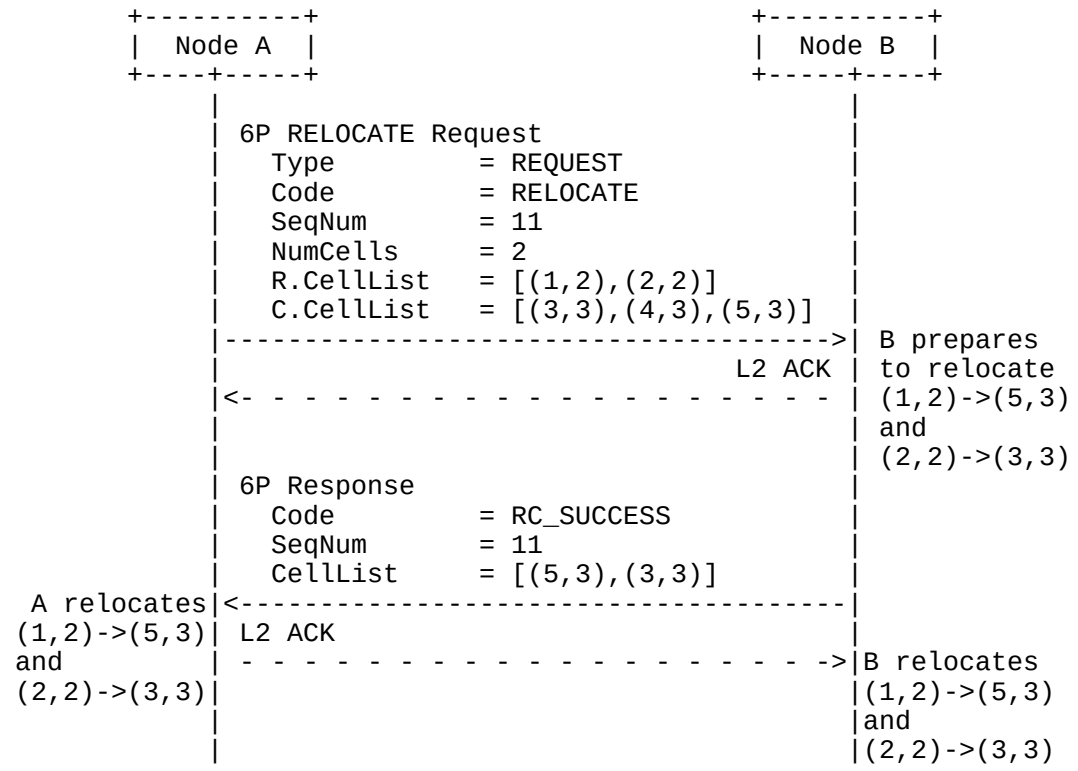


Figure 16: Example of a successful 2-step 6P RELOCATION Transaction.

Figure 17 shows an example of a partially successful 2-step 6P RELOCATION Transaction.

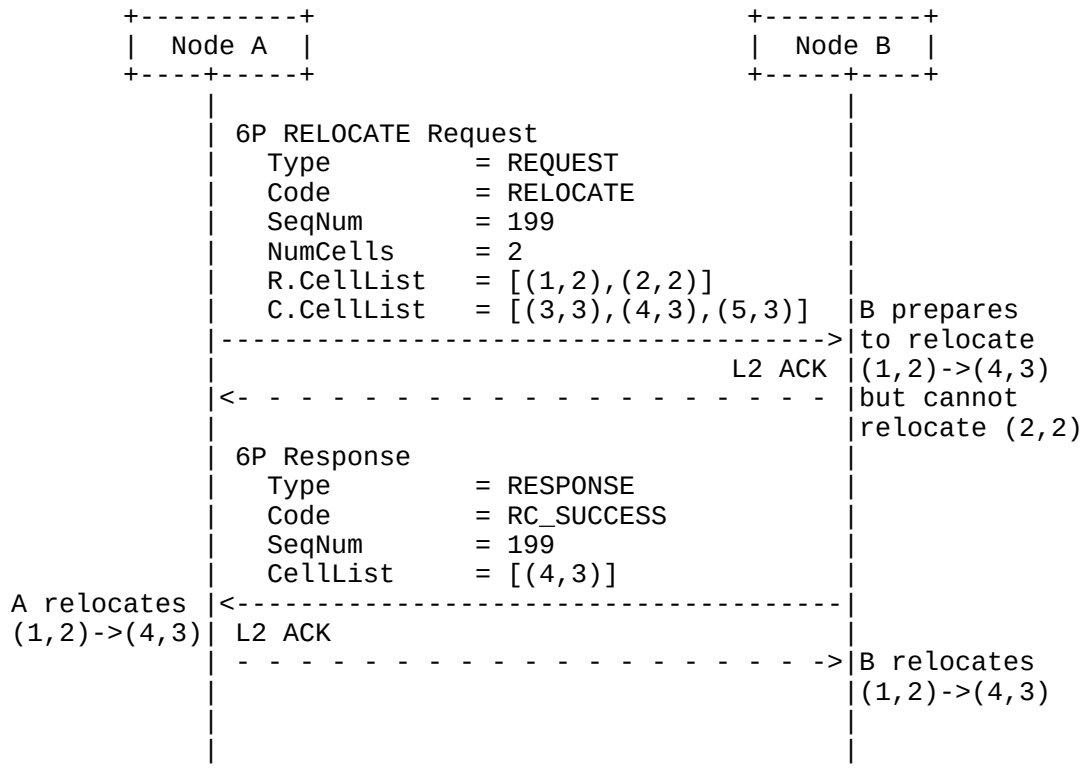


Figure 17: Example of a partially successful 2-step 6P RELOCATION Transaction.

Figure 18 shows an example of a failed 2-step 6P RELOCATION Transaction.

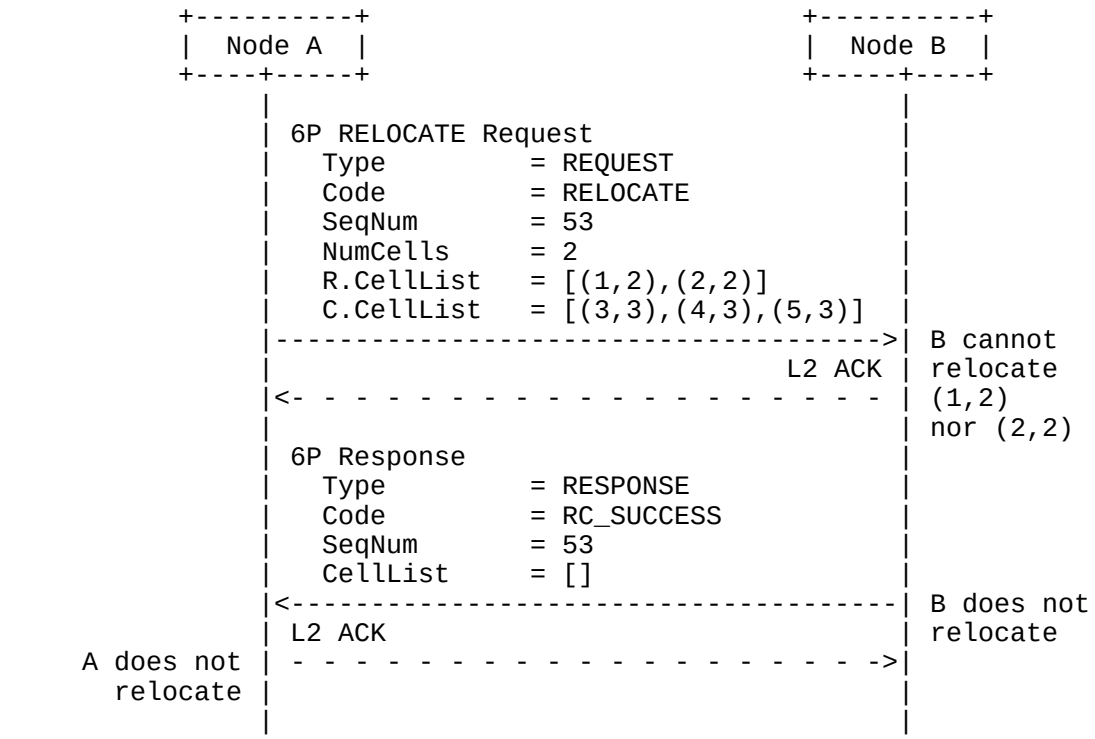


Figure 18: Failed 2-step 6P RELOCATION Transaction Example.

Figure 19 shows an example of a successful 3-step 6P RELOCATION Transaction.

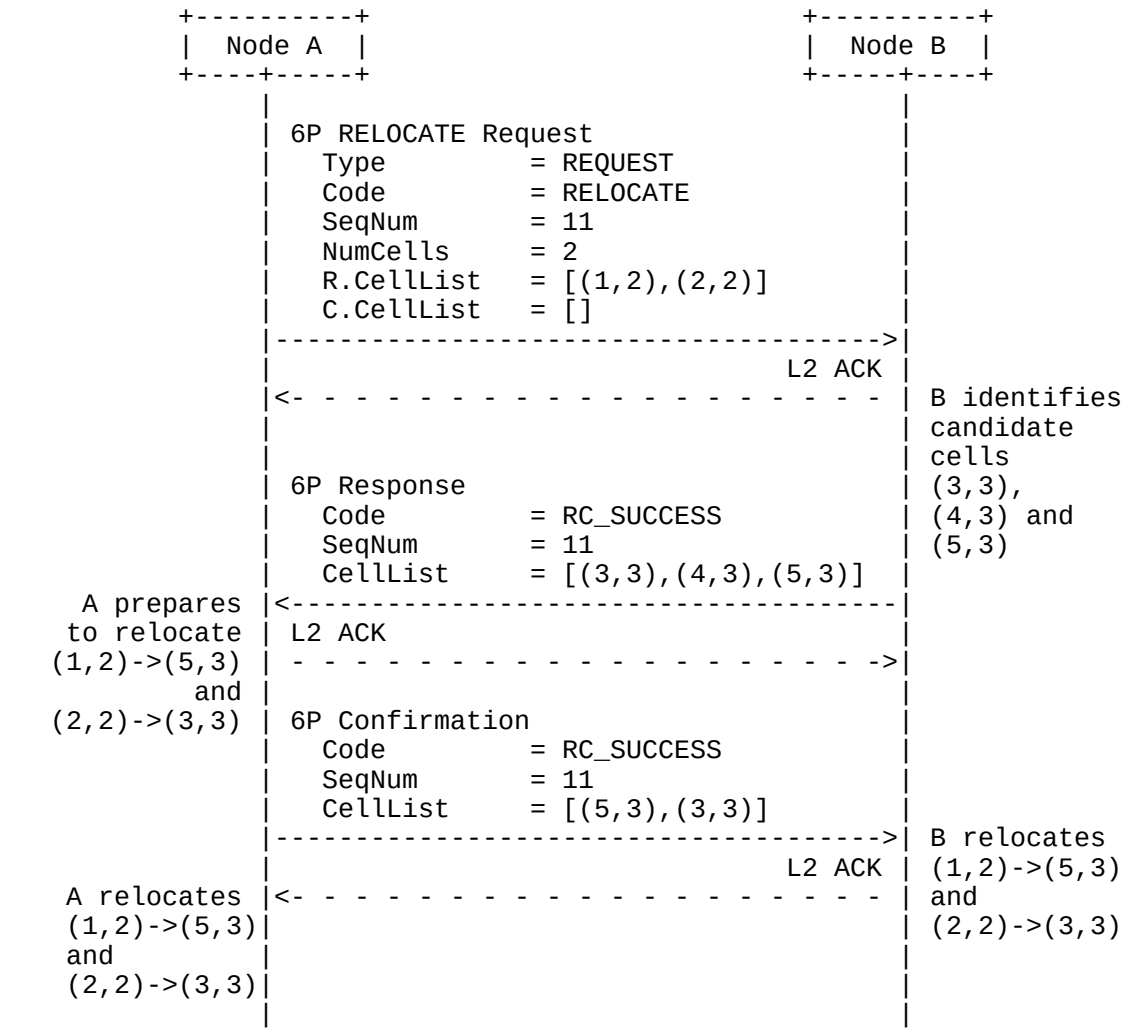


Figure 19: Example of a successful 3-step 6P RELOCATION Transaction.

3.3.4. Counting Cells

To retrieve the number of scheduled cells a node A has with B, node A issues a 6P COUNT command. The Type field (T) is set to REQUEST. The Code field is set to COUNT. Figure 20 defines the format of a 6P COUNT Request.

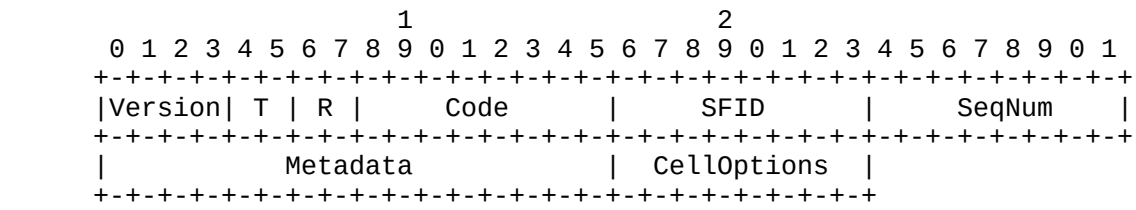


Figure 20: 6P COUNT Request Format.

Metadata: Same usage as for the 6P ADD command, see Section 3.3.1. Its format is the same as that in the 6P ADD command, but its content could be different.
CellOptions: Specifies which type of cell to be counted.

Figure 21 defines the format of a 6P COUNT Response.

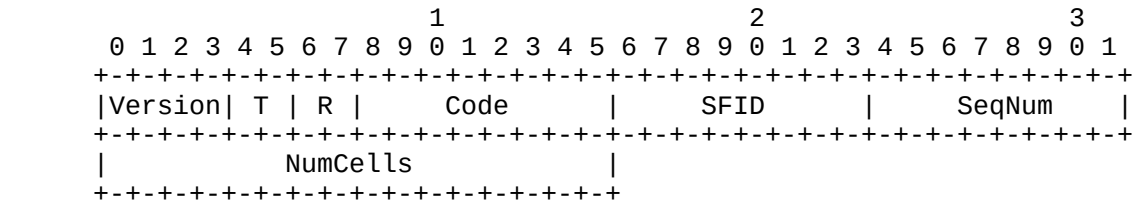


Figure 21: 6P COUNT Response Format.

NumCells: The number of cells which correspond to the fields of the request.

Node A issues a COUNT command to node B, specifying cell options. Upon receiving the 6P COUNT request, node B goes through its schedule and counts the number of cells scheduled with node A in its own schedule, and which match the cell options in the CellOptions field of the request. Section 3.2.3 details the use of the CellOptions field.

Node B issues a 6P response to node A with return code set to RC_SUCCESS, and with NumCells containing the number of cells that match the request.

3.3.5. Listing Cells

To retrieve a list of scheduled cells node A has with node B, node A issues a 6P LIST command. The Type field (T) is set to REQUEST. The Code field is set to LIST. Figure 22 defines the format of a 6P LIST Request.

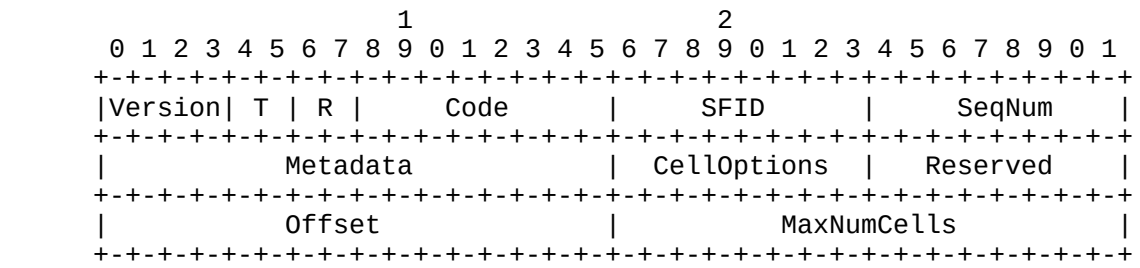


Figure 22: 6P LIST Request Format.

Metadata: Same usage as for the 6P ADD command, see Section 3.3.1. Its format is the same as that in the 6P ADD command, but its content could be different.

CellOptions: Specifies which type of cell to be listed.

Reserved: Reserved bits. These bits SHOULD be set to zero when sending the message, and MUST be ignored upon reception.

Offset: The Offset of the first scheduled cell that is requested. The mechanism assumes cells are ordered according to a rule defined in the SF. The rule MUST always order the cells in the same way.

MaxNumCells: The maximum number of cells to be listed. Node B MAY return less than MaxNumCells cells, for example if MaxNumCells cells do not fit in the frame.

Figure 23 defines the format of a 6P LIST Response.

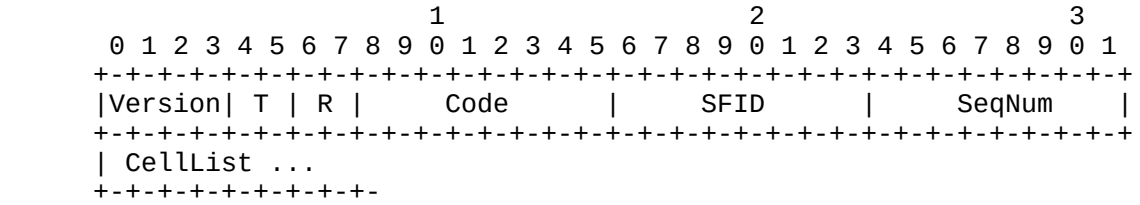


Figure 23: 6P LIST Response Format.

CellList: A list of 0, 1 or multiple 6P Cells.

When receiving a LIST command, node B returns the cells scheduled with A in its schedule that match the CellOptions field as specified in Section 3.2.3.

When node B receives a LIST request, the returned CellList in the 6P Response contains between 1 and MaxNumCells cells, starting from the specified offset. Node B SHOULD include as many cells as fit in the frame. If the response contains the last cell, Node B MUST set the

Code field in the response to RC_EOL ("End of List", as per Figure 37), indicating to Node A that there no more cells that match the request. Node B MUST return at least one cell, unless the specified Offset is beyond the end of B's cell list in its schedule. If node B has less than Offset cells that match the request, node B returns an empty CellList and a Code field set to RC_EOL.

3.3.6. Clearing the Schedule

To clear the schedule between nodes A and B (for example after a schedule inconsistency is detected), node A issues a CLEAR command. The Type field (T) is set to 6P Request. The Code field is set to CLEAR. Figure 24 defines the format of a 6P CLEAR Request.

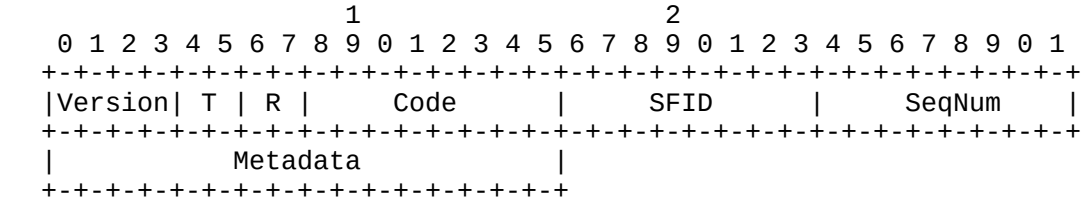


Figure 24: 6P CLEAR Request Format.

Metadata: Same usage as for the 6P ADD command, see Section 3.3.1. Its format is the same as that in the 6P ADD command, but its content could be different.

Figure 25 defines the format of a 6P CLEAR Response.

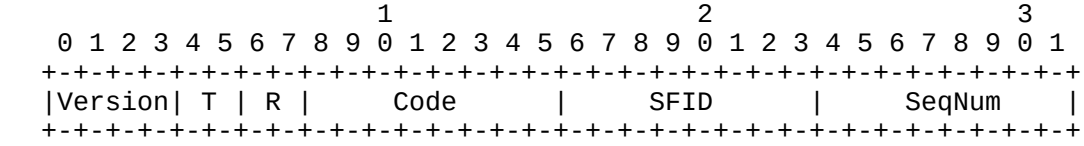


Figure 25: 6P CLEAR Response Format.

When a 6P CLEAR command is issued from node A to node B, both nodes A and B MUST remove all the cells scheduled between them. That is, node A MUST remove all the cells scheduled with node B, and node B MUST remove all the cells scheduled with node A. In a 6P CLEAR command, the SeqNum MUST NOT be checked. In particular, even if the request contains a SeqNum value that would normally cause node B to detect a schedule inconsistency, the transaction MUST NOT be aborted. Upon 6P CLEAR completion, the value of SeqNum MUST be reset to 0.

The Response Code to a 6P CLEAR command SHOULD be RC_SUCCESS unless the operation cannot be executed. When the CLEAR operation cannot be executed, the Response Code MUST be set to RC_RESET.

3.3.7. Generic Signaling Between SFs

The 6P SIGNAL message allows the SF implementations on two neighbor nodes to exchange generic commands. The payload in a received SIGNAL message is an opaque set of bytes passed unmodified to the SF. How the generic SIGNAL command is used is specified by the SF, and outside the scope of this document. The Type field (T) is set to REQUEST. The Code field is set to SIGNAL. Figure 26 defines the format of a 6P SIGNAL Request.

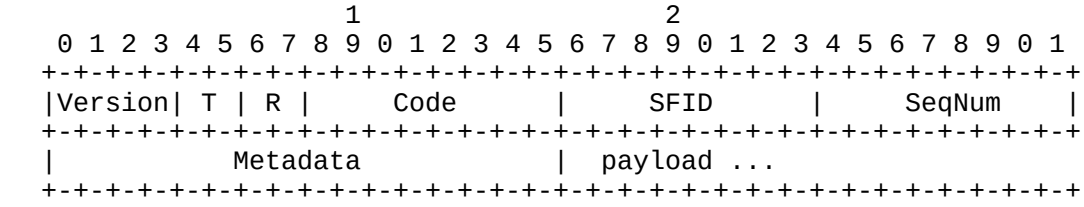


Figure 26: 6P SIGNAL Request Format.

Metadata: Same usage as for the 6P ADD command, see Section 3.3.1. Its format is the same as that in the 6P ADD command, but its content could be different.

Figure 27 defines the format of a 6P SIGNAL Response.

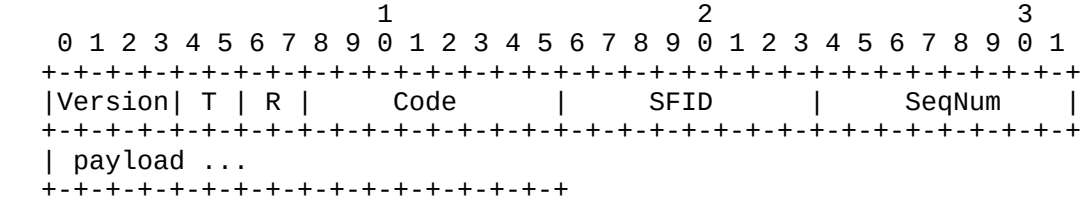


Figure 27: 6P SIGNAL Response Format.

3.4. Protocol Functional Details

3.4.1. 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 from 0), a node MAY implement multiple protocol versions at the same time. When a node receives a 6P message with a

Version number it does not implement, the node MUST reply with a 6P Response with a Return Code field set to RC_ERR_VERSION. The format of this 6P Response message MUST be compliant with Version 0 and MUST be supported by all future versions of the protocol. This ensures that, when node B sends a 6P Response to node A indicating it does not implement the 6P version in the 6P Request, node A can successfully parse that response.

When a node supports a version number received in a 6P Request message, the Version field in the 6P Response MUST be the same as the Version field in the corresponding 6P Request. Similarly, in a 3-step transaction, the Version field in the 6P Confirmation MUST match that of the 6P Request and 6P Response of the same transaction.

3.4.2. SFID Checking

All messages contain an SFID field. 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 RC_ERR_SFID. The SFID field in the 6P Response MUST be the same as the SFID field in the corresponding 6P Request. In a 3-step transaction, the SFID field in the 6P Confirmation MUST match that of the 6P Request and the 6P Response of the same transaction.

3.4.3. 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, except 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 RC_RESET (as per Figure 37) and discard this ongoing second transaction. A node receiving a RC_RESET code MUST abort the second transaction and consider it never happened.

Nodes A and B MAY support having two transactions going on at the same time, one in each direction. Similarly, a node MAY support concurrent 6P Transactions with different neighbors. In this case, the cells involved in an 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 R. In case a node does not have enough resources to handle concurrent 6P Transactions from different neighbors it MUST reply with a 6P Response with return code RC_ERR_BUSY (as per Figure 37). In case the requested cells are locked, it MUST reply to that request with a

6P Response with return code RC_ERR_LOCKED (as per Figure 37). The node receiving RC_ERR_BUSY or a RC_ERR_LOCKED MAY implement a retry mechanism, defined by the SF.

3.4.4. 6P Timeout

A timeout occurs when the node have successfully sent a 6P Request does not receive the corresponding 6P Response within an amount of time specified by the SF. In a 3-step transaction, a timeout also occurs when a node sending the 6P Response does not receive a 6P Confirmation. When a timeout occurs, the transaction MUST be canceled at the node where the timeout occurs. The value of the 6P Timeout should be larger than the longest possible time it takes to receive the 6P Response or Confirmation. The value of the 6P Timeout hence depends on the number of cells scheduled between the neighbor nodes, the maximum number of link-layer retransmissions, etc. The SF MUST determine the value of the timeout. The value of the timeout is out of scope of this document.

3.4.5. Aborting a 6P Transaction

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

Similarly, in the case of 3-step transaction, when the receiver of a 6P Response fails during the 6P Transaction and is unable to complete it, it MUST reply to that 6P Response with a 6P Confirmation with return code RC_RESET. Upon receiving this 6P Confirmation, the sender of the 6P Response MUST consider the 6P Transaction as failed.

3.4.6. SeqNum Management

The SeqNum is the field in the 6top IE header used to match Request, Response and Confirmation. The SeqNum is used to detect and handle duplicate commands (Section 3.4.6.1) and schedule inconsistencies (Section 3.4.6.2). Each node remembers the last used SeqNum for each neighbor. That is, a node stores as many SeqNum values as it has neighbors. In case of supporting multiple SFs at a time, a SeqNum value is maintained per SF and per neighbor. In the remainder of this section, we describe the use of SeqNum between two neighbors; the same happens for each other neighbor, independently.

When a node resets or after a CLEAR transaction, it MUST reset SeqNum to 0. The 6P Response and 6P Confirmation for a transaction MUST use

the same SeqNum value as that in the Request. After every transaction, the SeqNum MUST be incremented by exactly 1.

Specifically, if node A receives the link-layer acknowledgment for its 6P Request, it commits to incrementing the SeqNum by exactly 1 after the 6P Transaction ends. This ensure that, at the next 6P Transaction where it sends a 6P Request, that 6P Request will have a different SeqNum.

Similarly, a node B increments the SeqNum by exactly 1 after having received the link-layer acknowledgment for the 6P Response (2-step 6P Transaction), or after having sent the link-layer acknowledgment for the 6P Confirmation (3-step 6P Transaction) .

When a node B receives a 6P Request from node A with SeqNum equal to 0, it checks the stored SeqNum for A. If A is a new neighbor, the stored SeqNum in B will be 0. The transaction can continue. If the stored SeqNum for A in B is different than 0, a potential inconsistency is detected. In this case B MUST return RC_ERR_SEQNUM with SeqNum=0. The SF at A MAY decide what to do next as described in Section 3.4.6.2.

The SeqNum MUST be implemented as a lollipop counter: it rolls over from 0xFF to 0x01 (not to 0x00). This is used to detect a neighbor reset. Figure 28 lists the possible values of the SeqNum.

Value	Meaning
0x00	Clear or After device Reset
0x01-0xFF	Lollipop Counter values

Figure 28: Possible values of the SeqNum.

3.4.6.1. Detecting and Handling Duplicate 6P Messages

All 6P commands are link-layer acknowledged. A duplicate message means that a node receives a second 6P Request, Response or Confirmation. This happens when the link-layer acknowledgment is not received, and a link-layer retransmission happens. Duplicate messages are normal and unavoidable.

Figure 29 shows an example 2-step transaction in which Node A receives a duplicate 6P Response.

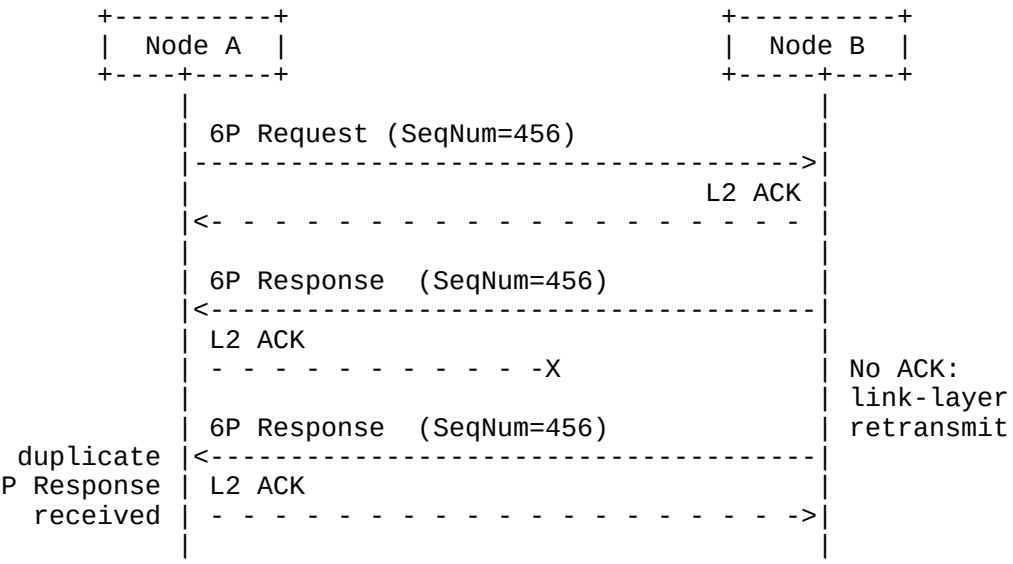


Figure 29: Example duplicate 6P message.

Figure 30 shows example 3-step transaction in which Node A receives a out-of-order duplicate 6P Response after having sent a 6P Confirmation.

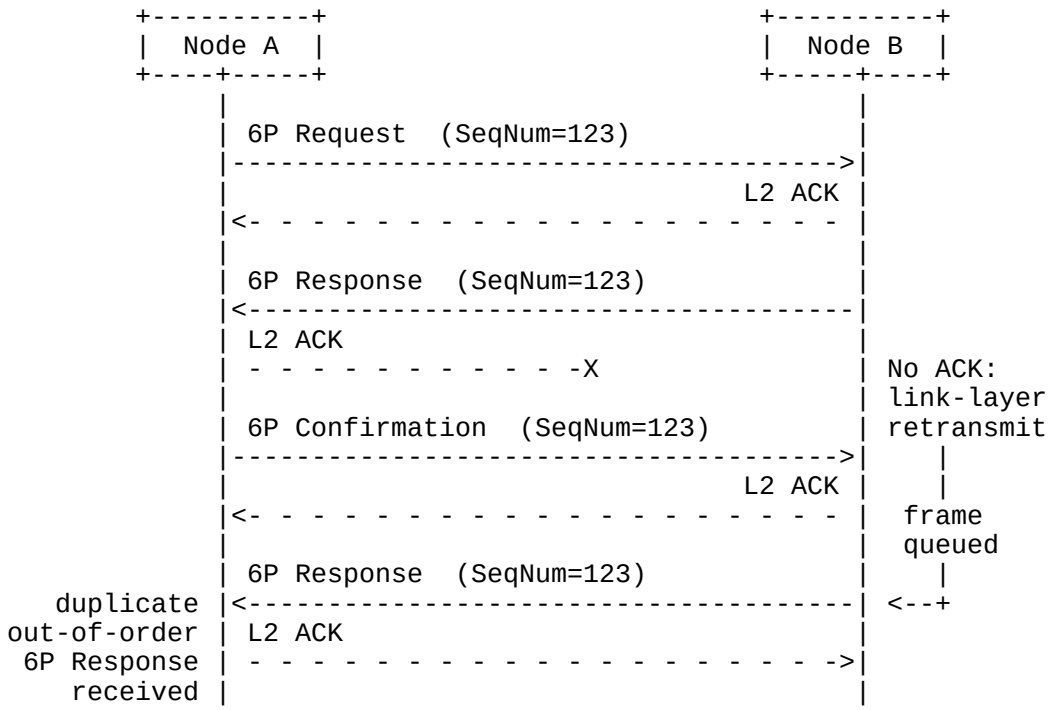


Figure 30: Example out-of-order duplicate 6P message.

A node detects a duplicate 6P message when it has the same SeqNum and type as the last frame received from the same neighbor. When receiving a duplicate 6P message, a node MUST send a link-layer acknowledgment, but MUST silently ignore the 6P message at the 6top sublayer.

3.4.6.2. Detecting and Handling a Schedule Inconsistency

A schedule inconsistency happens when the schedules of nodes A and B are inconsistent. For example, when node A has a transmit cell to node B, but node B does not have the corresponding receive cell, and therefore isn't listening to node A on that cell. A schedule inconsistency results in loss of connectivity.

The SeqNum field, which is present in each 6P message, is used to detect an inconsistency. The SeqNum field increments by 1 at each message. A node computes the expected SeqNum field for the next 6P Transaction. If a node receives a 6P Request with a SeqNum value that is not the expected one, it has detected an inconsistency.

There are at least 2 cases in which a schedule inconsistency happens.

The first case is when a node loses state, for example when it is power cycled (turned off, then on). In that case, its SeqNum value is reset to 0. Since the SeqNum is a lollipop counter, its neighbor detects an inconsistency at the next 6P transaction. This is illustrated in Figure 31.

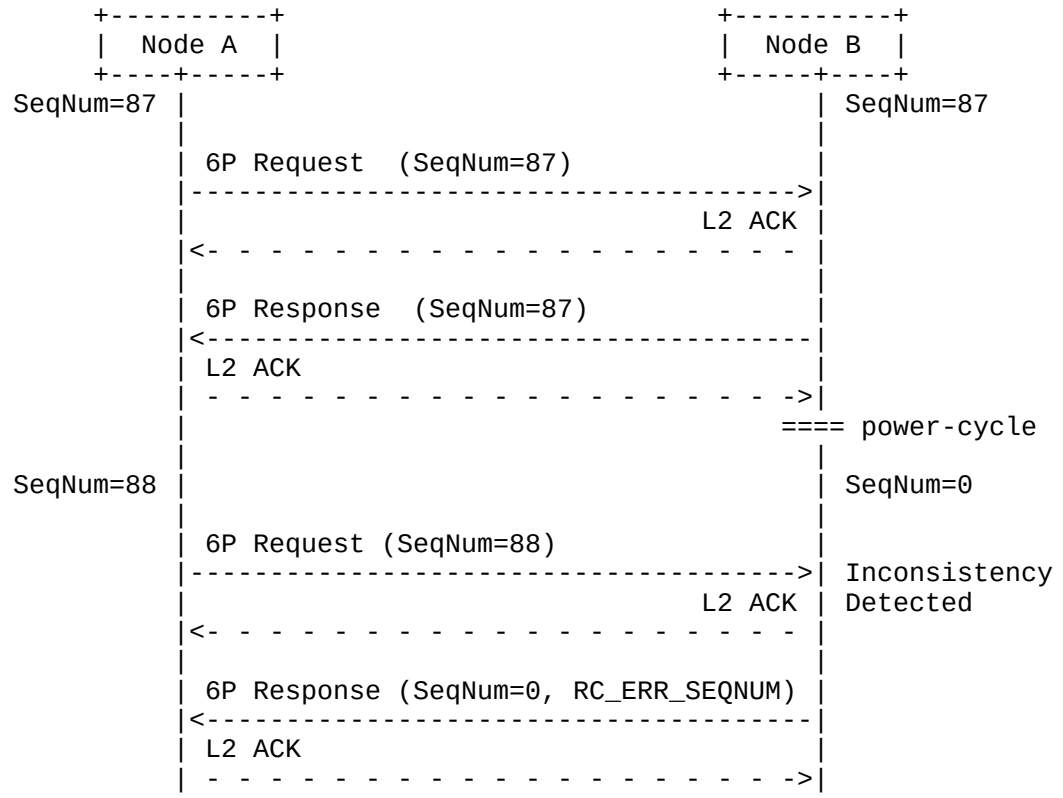


Figure 31: Example of inconsistency because of node reset.

The second case is when the maximum number of link-layer retransmissions is reached on the 6P Response of a 2-step transaction (or equivalently on a 6P Confirmation of a 3-step transaction). This is illustrated in Figure 32.

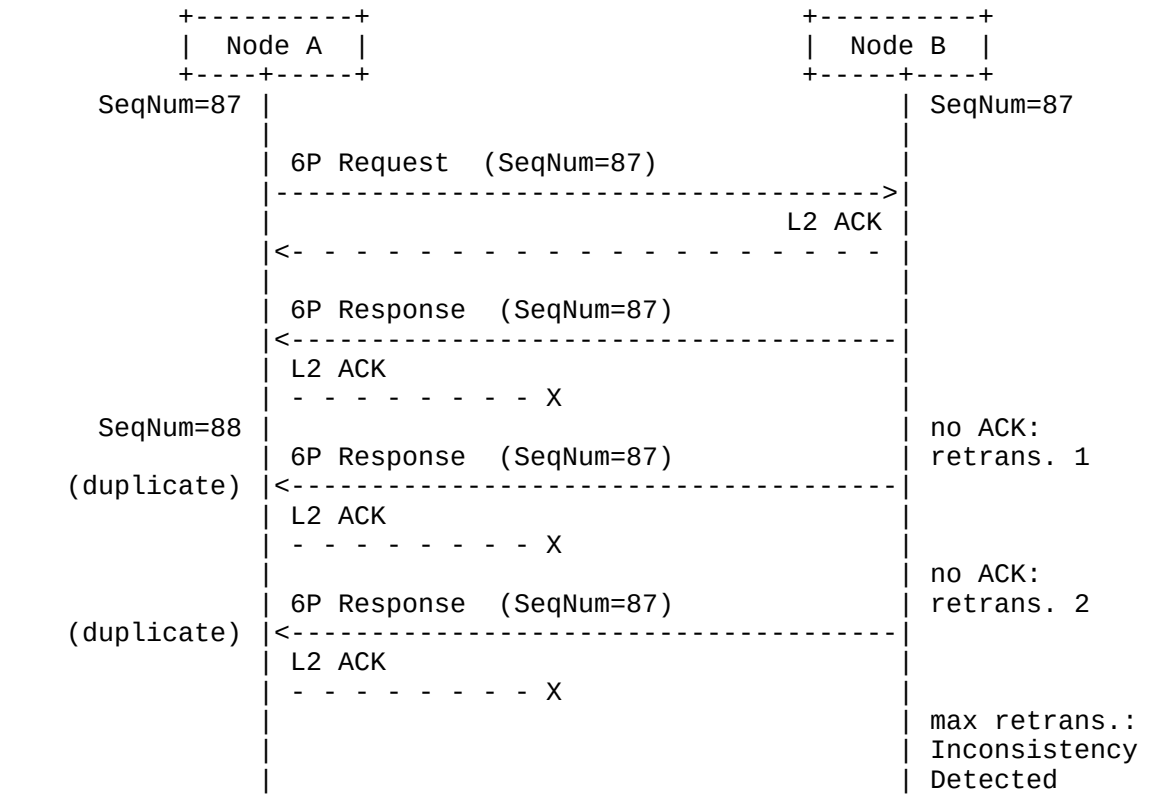


Figure 32: Example inconsistency because of maximum link-layer retransmissions (here 2).

In both cases, node B detects the inconsistency.

If the inconsistency is detected during a 6P Transaction (Figure 31), the node that has detected it MUST send back a 6P Response or 6P Confirmation with an error code of RC_ERR_SEQNUM. In this 6P Response or 6P Confirmation, the SeqNum field MUST be set to the value of the sender of the message (0 in the example in Figure 31).

The SF of the node which has detected the inconsistency MUST define how to handle the inconsistency. A first possibility is to issue a 6P CLEAR request to clear the schedule, and rebuild. A second possibility is to issue a 6P LIST request to retrieve the schedule. A third possibility is to internally "roll-back" the schedule. How to handle an inconsistency is out of scope of this document. The SF defines how to handle an inconsistency.

3.4.7. Handling Error Responses

A return code marked as Yes in the "Is Error" column in Figure 37 indicates an error. When a node receives a 6P Response or 6P Confirmation with an error, it MUST consider the 6P Transaction as failed. In particular, if this was a response to a 6P ADD, DELETE or RELOCATE Request, the node MUST NOT add, delete or relocate any of the cells involved in this 6P Transaction. Similarly, a node sending a 6P Response or a 6P Confirmation with an error code MUST NOT add, delete, relocate any cells as part of that 6P Transaction. Defining what to do after an error has occurred is out of scope of this document. The SF defines what to do after an error has occurred.

3.5. 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 IEs.

4. Requirements for 6top Scheduling Functions (SF)

4.1. SF Identifier (SFID)

Each SF has a 1-byte identifier. Section 6.2.5 defines the rules for applying for an SFID.

4.2. Requirements for an SF

The specification for an SF

- MUST specify an identifier for that SF.
- MUST specify the rule for a node to decide when to add/delete one or more cells to a neighbor.
- MUST specify the rule for a Transaction source to select cells to add to the CellList field in the 6P ADD Request.
- MUST specify the rule for a Transaction destination to select cells from CellList to add to its schedule.
- MUST specify a value for the 6P Timeout, or a rule/equation to calculate it.
- MUST specify the rule for ordering cells.
- MUST specify a meaning for the "Metadata" field in the 6P ADD Request.
- MUST specify the SF behavior of a node when it boots.
- MUST specify how to handle a schedule inconsistency.
- MUST specify what to do after an error has occurred (either the node sent a 6P Response with an error code, or received one).
- MUST specify the list of statistics to gather. Example statistics include 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.

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

- MAY redefine the format of the CellList field.
- MAY redefine the format of the CellOptions field.
- MAY redefine the meaning of the CellOptions field.

5. Security Considerations

6P messages are carried inside 802.15.4 Payload Information Elements (IEs). Those Payload IEs are encrypted and authenticated at the link layer through CCM* [CCM-Star]. 6P benefits from the same level of security as any other Payload IE. The 6P protocol does not define its own security mechanisms. In particular, although a key management solution is out of scope of this document, the 6P protocol will benefit for the key management solution used in the network.

The 6P protocol does not provide protection against DOS attacks. Example attacks include, not sending confirmation messages in 3-step transaction, sending wrongly formatted requests, etc. These cases SHOULD be handled by an appropriate policy, such as blacklisting the attacker after several attempts. The effect on the overall network is mostly localized to those two nodes, as communication happens in dedicated cells.

6. IANA Considerations

6.1. IETF IE Subtype ’6P’

This document adds the following number to the "IEEE Std 802.15.4 IETF IE subtype IDs" registry defined by [RFC8137]:

Subtype	Name	Reference
IANA_6TOP_SUBIE_ID	6P	RFCXXXX

Figure 33: IETF IE Subtype ’6P’.

6.2. 6TiSCH parameters sub-registries

This section defines sub-registries within the "IPv6 over the TSCH mode of IEEE 802.15.4e (6TiSCH) parameters" registry, hereafter referred to as the "6TiSCH parameters" registry. Each sub-registry is described in a subsection.

6.2.1. 6P Version Numbers

The name of the sub-registry is "6P Version Numbers".

A Note included in this registry should say: "In the 6top Protocol (6P) [RFCXXXX] there is a field to identify the version of the protocol. This field is 4 bits in size."

Each entry in the sub-registry must include the Version in the range 0-15, and a reference to the 6P version’s documentation.

The initial entry in this sub-registry is as follows:

Version	Reference
0	RFCXXXX

Figure 34: 6P Version Numbers.

All other Version Numbers are Unassigned.

The IANA policy for future additions to this sub-registry is "IETF Review or IESG Approval" as described in [RFC8126].

6.2.2. 6P Message Types

The name of the sub-registry is "6P Message Types".

A note included in this registry should say: "In the 6top Protocol (6P) version 0 [RFCXXXX], there is a field to identify the type of message. This field is 2 bits in size."

Each entry in the sub-registry must include the Type in range b00-b11, the corresponding Name, and a reference to the 6P message type's documentation.

Initial entries in this sub-registry are as follows:

Type	Name	Reference
b00	REQUEST	RFCXXXX
b01	RESPONSE	RFCXXXX
b10	CONFIRMATION	RFCXXXX

Figure 35: 6P Message Types.

All other Message Types are Reserved.

The IANA policy for future additions to this sub-registry is "IETF Review or IESG Approval" as described in [RFC8126].

6.2.3. 6P Command Identifiers

The name of the sub-registry is "6P Command Identifiers".

A Note included in this registry should say: "In the 6top Protocol (6P) version 0 [RFCXXXX], there is a Code field which is 8 bits in size. In a 6P Request, the value of this Code field is used to identify the command."

Each entry in the sub-registry must include an Identifier in the range 0-255, the corresponding Name, and a reference to the 6P command identifier's documentation.

Initial entries in this sub-registry are as follows:

Identifier	Name	Reference
0	Reserved	
1	ADD	RFCXXXX
2	DELETE	RFCXXXX
3	RELOCATE	RFCXXXX
4	COUNT	RFCXXXX
5	LIST	RFCXXXX
6	SIGNAL	RFCXXXX
7	CLEAR	RFCXXXX
8-254	Unassigned	
255	Reserved	

Figure 36: 6P Command Identifiers.

The IANA policy for future additions to this sub-registry is "IETF Review or IESG Approval" as described in [RFC8126].

6.2.4. 6P Return Codes

The name of the sub-registry is "6P Return Codes".

A Note included in this registry should say: "In the 6top Protocol (6P) version 0 [RFCXXXX], there is a Code field which is 8 bits in size. In a 6P Response or 6P Confirmation, the value of this Code field is used to identify the return code."

Each entry in the sub-registry must include a Code in the range 0-255, the corresponding Name, the corresponding Description, and a reference to the 6P return code’s documentation.

Initial entries in this sub-registry are as follows:

Code	Name	Description	Is Error?
0	RC_SUCCESS	operation succeeded	No
1	RC_EOL	end of list	No
2	RC_ERR	generic error	Yes
3	RC_RESET	critical error, reset	Yes
4	RC_ERR_VERSION	unsupported 6P version	Yes
5	RC_ERR_SFID	unsupported SFID	Yes
6	RC_ERR_SEQNUM	schedule inconsistency	Yes
7	RC_ERR_CELLLIST	cellList error	Yes
8	RC_ERR_BUSY	busy	Yes
9	RC_ERR_LOCKED	cells are locked	Yes

Figure 37: 6P Return Codes.

All other Message Types are Unassigned.

The IANA policy for future additions to this sub-registry is "IETF Review or IESG Approval" as described in [RFC8126].

6.2.5. 6P Scheduling Function Identifiers

6P Scheduling Function Identifiers.

A Note included in this registry should say: "In the 6top Protocol (6P) version 0 [RFCXXXX], there is a field to identify the scheduling function to handle the message. This field is 8 bits in size."

Each entry in the sub-registry must include an SFID in the range 0-255, the corresponding Name, and a reference to the 6P Scheduling Function's documentation.

Initial entries in this sub-registry are as follows:

SFID	Name	Reference
0	Minimal Scheduling Function (MSF)	draft-chang-6tisch-msf
1	Experimental Scheduling Function (SFX)	draft-ietf-6tisch-6top-sfx

Figure 38: SF Identifiers (SFID).

All other Message Types are Unassigned.

The IANA policy for future additions to this sub-registry depends on the value of the SFID, as defined in Figure 39. These specifications must follow the guidelines of Section 4.

Range	Registration Procedures
0-127	IETF Review or IESG Approval
128-255	Expert Review

Figure 39: SF Identifier (SFID): Registration Procedure.

6.2.6. 6P CellOptions bitmap

The name of the sub-registry is "6P CellOptions bitmap".

A Note included in this registry should say: "In the 6top Protocol (6P) version 0 [RFCXXXX], there is an optional CellOptions field which is 8 bits in size."

Each entry in the sub-registry must include a bit position in the range 0-7, the corresponding Name, and a reference to the bit's documentation.

Initial entries in this sub-registry are as follows:

bit	Name	Reference
0	TX (Transmit)	RFCXXXX
1	RX (Receive)	RFCXXXX
2	SHARED	RFCXXXX
3-7	Reserved	

Figure 40: 6P CellOptions bitmap.

All other Message Types are Reserved.

The IANA policy for future additions to this sub-registry is "IETF Review or IESG Approval" as described in [RFC8126].

7. References

7.1. Normative References

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

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

[RFC8137] Kivinen, T. and P. Kinney, "IEEE 802.15.4 Information Element for the IETF", RFC 8137, DOI 10.17487/RFC8137, May 2017, <https://www.rfc-editor.org/info/rfc8137>.

7.2. Informative References

[CCM-Star]
Struik, R., "Formal Specification of the CCM* Mode of Operation, IEEE P802.15 Working Group for Wireless Personal Area Networks (WPANs).", September 2005.

[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.

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

[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, <https://www.rfc-editor.org/info/rfc7554>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8180] Vilajosana, X., Ed., Pister, K., and T. Watteyne, "Minimal IPv6 over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration", BCP 210, RFC 8180, DOI 10.17487/RFC8180, May 2017, <https://www.rfc-editor.org/info/rfc8180>.

Appendix A. Recommended Structure of an SF Specification

The following section structure for a SF document is RECOMMENDED:

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

Authors' Addresses

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

Email: wangqin@ies.ustb.edu.cn

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

Email: xvilajosana@uoc.edu

Thomas Watteyne
Analog Devices
32990 Alvarado-Niles Road, Suite 910
Union City, CA 94587
USA

Email: thomas.watteyne@analog.com

6TiSCH
Internet-Draft
Intended status: Standards Track
Expires: September 14, 2017

D. Dujovne, Ed.
Universidad Diego Portales
LA. Grieco
Politecnico di Bari
MR. Palattella
Luxembourg Institute of Science and Technology (LIST)
N. Accettura
LAAS-CNRS
March 13, 2017

6TiSCH 6top Scheduling Function Zero (SF0)
draft-ietf-6tisch-6top-sf0-03

Abstract

This document defines a Scheduling Function called "Scheduling Function Zero" (SF0). SF0 dynamically adapts the number of allocated cells between neighbor nodes, based on the amount of currently allocated cells and the neighbor nodes' cell requirements. Neighbor nodes negotiate in a distributed neighbor-to-neighbor basis the number of cell(s) to be added/deleted. SF0 uses the 6P signaling messages to add/delete cells in the schedule. This function selects the candidate cells from the schedule, defines which cells will be added/deleted and triggers the allocation/deallocation process.

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

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 http://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 14, 2017.

Table of Contents

1. TEMPORARY EDITORIAL NOTES 3
2. Introduction 3
3. Scheduling Function Identifier 4
4. SF0 Triggering Events 4
5. SF0 Cell Estimation Algorithm 4
6. SF0 Allocation Policy 5
7. Rules for CellList 6
8. 6P Timeout Value 7
9. Meaning of Metadata Information 7
10. Node Behavior at Boot 7
11. Cell Type 7
12. SF0 Statistics 8
13. Relocating Cells 8
14. Forced Cell Deletion Policy 8
15. 6P Error Handling 8
16. Examples 9
17. Implementation Status 9
18. Security Considerations 9
19. IANA Considerations 9
20. 6P Compliance 10
21. Acknowledgments 10
22. References 10
22.1. Normative References 10
22.2. Informative References 11
Appendix A. [TEMPORARY] Changelog 11
Authors' Addresses 12

1. TEMPORARY EDITORIAL NOTES

This document is an Internet Draft, so it is 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, etc). The statement does indicate that the text will be written at some time.
- "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.
- The string "REMARK" is put before a remark (questions, suggestion, etc) from an author, editor of contributor. These are on-going discussions at the time to writing, NOT part of the final text.
- This section will be removed in the final text.

2. Introduction

This document defines a minimal Scheduling Function for the 6top sublayer [I-D.ietf-6tisch-6top-protocol], called "Scheduling Function Zero" (SF0). SF0 is designed to offer the minimal set of functionalities to be usable in a wide range of applications. SF0 defines two algorithms: The Scheduling Algorithm defines the number of cells to allocate/delete between two neighbours and the relocation algorithm defines when to relocate a cell.

The Scheduling Algorithm: A number of TX and/or RX cells must be allocated between neighbor nodes in order to enable data transmission among them. A portion of these allocated cells will be utilized by neighbors, while the remaining cells can be over-provisioned to handle unanticipated increases in cell requirements. The Scheduling Algorithm collects the cell allocation/deletion requests from the neighbors and the number of cells which are currently under usage. First, the Cell Estimation Algotithm calculates the number of required cells by adding the collected values and second, the calculated value is given to the Allocation Policy, which provides stability by adding hystheresis and overprovisioning by deciding when to schedule the new number of cells, according to a threshold. In order to reduce consumption, this algorithm is triggered only when there is a change on the number of effectively used cells or if there is a change on the number of requested cells from a particular node.

The Relocation Algorithm: Allocated cells may experience packet loss from different sources, such as noise, interference or cell collision

(after the same cell is allocated by other nodes in range on the network). In order to avoid this problem, Packet Delivery Rate (PDR) is monitored periodically for each allocated cell. A relocation is triggered when the PDR value drops below a certain threshold, compared to the average PDR of the rest of allocated cells. The destination location on the schedule is defined randomly.

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

1. It waits for a triggering event (Section 4).
2. It applies the Cell Estimation Algorithm (CEA) for a particular neighbor to determine how many cells are required to that neighbor (Section 5).
3. It applies the Allocation Policy to compare the number of required cells to the number of already scheduled cells, and determines the number of cells to add/delete (Section 6).

We expect additional SFs, offering more functionalities for a more specific use case, to be defined in future documents. SF0 addresses the requirements for a scheduling function listed in Section 5.2 from [I-D.ietf-6tisch-6top-protocol], and follows the recommended outline listed in Section 5.3 of [I-D.ietf-6tisch-6top-protocol]. This document follows the terminology defined in [I-D.ietf-6tisch-terminology].

3. Scheduling Function Identifier

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

4. SF0 Triggering Events

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

1. If there is a change in the current number of required cells
2. If there is a successful cell allocation/deallocation process with the neighbour.

This allows SF0 to be triggered by any change in locally generated or incoming traffic. The exact mechanism of when SF0 is triggered is implementation-specific.

5. SF0 Cell Estimation Algorithm

The Cell Estimation Algorithm takes into account the new incoming cell requirements from the neighbor node and the current outgoing number of used cells. This allows the algorithm to estimate a new

number of cells to be allocated. As a consequence, the Cell Estimation Algorithm for SF0 follows the steps described below:

1. Collect the current number of used cells
2. Calculate the new number of cells to be allocated by adding the current number of used cells plus an OVERPROVISION number of cells
3. Submit the request to the allocation policy as REQUIREDCELLS
4. Return to step 1 and wait for a triggering event.

The OVERPROVISION parameter is a percentage of the currently allocated cells which is added to the used cells to guarantee that the growth on the number of used cells can be detected without packet loss. This percentage value is implementation-specific. A value of OVERPROVISION equal to zero leads to queue growth and possible packet loss, since there are no overprovisioned cells to detect if there is a growth on the number of used cells.

6. 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 cell 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 Cell 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' neighbors.

The SF0 allocation policy compares REQUIREDCELLS with SCHEDULEDCELLS and decides to add/delete cells taking into account SF0THRESH. This is illustrated in Figure 1. The number of cells to be scheduled/deleted is out of the scope of this document and it is implementation-dependent.

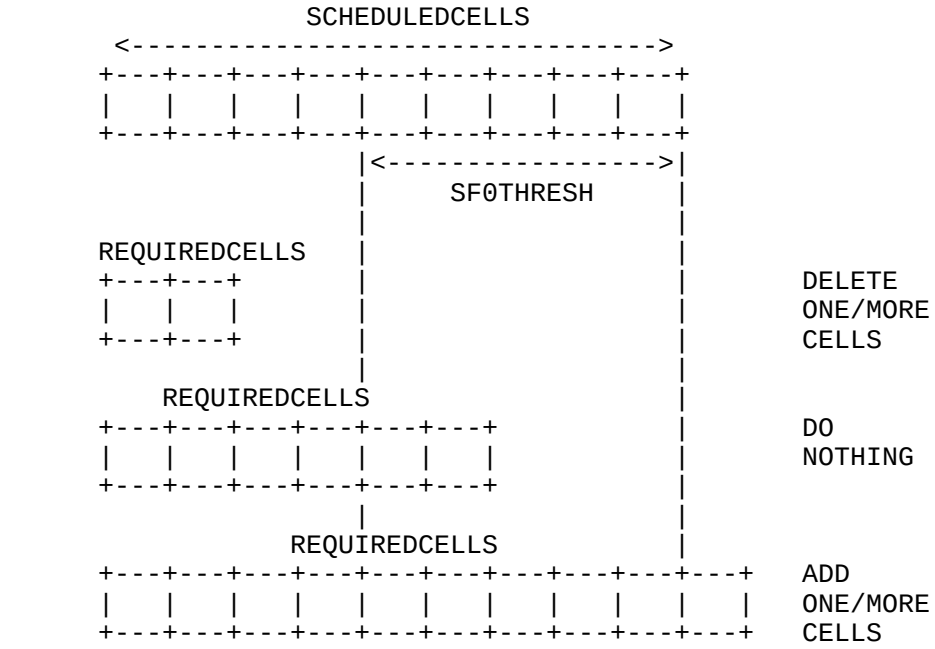


Figure 1: The SF0 Allocation Policy

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.

7. Rules for CellList

There are two methods to define the CellList: The Whitelist method, which fills the CellList with the number of proposed cells to the neighbour, and the Blacklist, which fills the CellList with the cells which cannot be used by the neighbour. The rule to select the method is implementation-specific. 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.

8. 6P Timeout Value

The general timeout equals the equivalent time of the number of slots until the next scheduled cell. TODO/REMARK: The exact calculation is currently under discussion on the Mailing List.

9. 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).

10. 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 neighbor nodes to enable a new allocation process. The 6P Initial Timeout Value provided by SF0 should allow for the maximum number of TSCH link-layer retries, as defined by Section 4.3.4 of [I-D.ietf-6tisch-6top-protocol]. TODO/REMARK: The initial timeout is currently under discussion on the Mailing List.

11. Cell Type

SF0 uses TX (Transmission) cell type only, thus defining celloptions as TX=0, RX=1 and S=0 according to section 4.2.6 of [I-D.ietf-6tisch-6top-protocol].

12. SF0 Statistics

Packet Delivery Rate (PDR) is calculated per cell, as the quotient of the number of successfully delivered packets to 10, for the last 10 packet transmission attempts, without counting retransmissions.

13. Relocating Cells

SF0 uses Packet Delivery Rate (PDR) statistics to monitor the currently allocated cells for cell relocation (by changing their slotOffset and/or channelOffset). When the PDR of one or more softcells is below PDR_THRESHOLD, defined as a percentage of the average of the PDR of the rest of the scheduled cells, SF0 relocates each of the cell(s) to a number of available cells selected randomly. PDR_THRESHOLD is out of the scope of this document and it is implementation-dependent.

14. Forced Cell Deletion Policy

When all the cells are scheduled, we need a policy to free cells, for example, under alarm conditions or if a node disappears from the neighbor list. The action to follow this condition is out of scope of this document and it is implementation-dependent.

15. 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 ADD 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 ADD Request, the neighbor has received the request, but less than NumCells of the cells in the CellList were allocated. 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_ERR_VER: The node MUST NOT retry immediately. The node MAY add the neighbor node to a blacklist. The node MAY retry to contact this neighbor later.

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

RC_ERR_GEN: The node MUST issue a CLEAR command to the neighbour.

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

RC_ERR_NORES: Wait for a timeout and restart the scheduling process.
RC_ERR_RESET: Abort 6P Transaction
RC_ERR: Abort 6P Transaction. The node MAY retry to contact this neighbor later.

16. Examples

TODO

17. 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.

18. Security Considerations

TODO

19. IANA Considerations

- IANA_SFID_SF0

20. 6P Compliance

- MUST specify an identifier for that SF. OK
- MUST specify the rule for a node to decide when to add/delete one or more cells to a neighbor. OK
- MUST specify the rule for a Transaction source to select cells to add to the CellList field in the 6P ADD Request. OK
- MUST specify the rule for a Transaction destination to select cells from CellList to add to its schedule. OK
- MUST specify a value for the 6P Timeout, or a rule/equation to calculate it. OK
- MUST specify a meaning for the "Metadata" field in the 6P ADD Request. OK
- MUST specify the behavior of a node when it boots. OK
- MUST specify what to do after an error has occurred (either the node sent a 6P Response with an error code, or received one). OK
- 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. OK
- SHOULD clearly state the application domain the SF is created for. OK
- SHOULD contain examples which highlight normal and error scenarios.
- SHOULD contain a list of current implementations, at least during the I-D state of the document, per [RFC6982].
- SHOULD contain a performance evaluation of the scheme, possibly through references to external documents.
- MAY redefine the format of the CellList? field. OK

21. 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).

22. References

22.1. 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>.

22.2. Informative References

[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-08 (work in progress), December 2016.

[I-D.ietf-6tisch-6top-protocol] Wang, Q. and X. Vilajosana, "6top Protocol (6P)", draft-ietf-6tisch-6top-protocol-03 (work in progress), October 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] Changelog

- draft-ietf-6tisch-6top-sf0-02
  - Editorial changes (figs, typos, ...)
- draft-ietf-6tisch-6top-sf0-03
  - Fixed typos
  - Removed references to "effectively used cells"
  - Changed Cell Estimation Algorithm to the third proposed alternative on IETF97
  - Forced cell deletion becomes implementation specific
  - Added PDR calculation formula
  - Added PDR_THRESHOLD as implementation specific value

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
Luxembourg Institute of Science and Technology (LIST)
Department 'Environmental Research and Innovation' (ERIN)
41, rue du Brill
Belvaux L-4422
Grand-duchy of Luxembourg

Phone: +352 275 888-5055
Email: mariarita.palattella@list.lu

Nicola Accettura
LAAS-CNRS
7, avenue du Colonel Roche
Toulouse 31400
France

Phone: +33 5 61 33 69 76
Email: nicola.accettura@laas.fr

6TiSCH
Internet-Draft
Intended status: Experimental
Expires: January 3, 2018

D. Dujovne, Ed.
Universidad Diego Portales
LA. Grieco
Politecnico di Bari
MR. Palattella
Luxembourg Institute of Science and Technology (LIST)
N. Accettura
LAAS-CNRS
July 2, 2017

6TiSCH 6top Scheduling Function Zero (SF0)
draft-ietf-6tisch-6top-sf0-05

Abstract

This document defines a Scheduling Function called "Scheduling Function Zero" (SF0). SF0 dynamically adapts the number of scheduled cells between neighbor nodes, based on the amount of currently allocated cells and the neighbor nodes' cell requirements. Neighbor nodes negotiate in a distributed neighbor-to-neighbor basis the number of cell(s) to be added/deleted. SF0 uses the 6P signaling messages to add/delete cells in the schedule. This function selects the candidate cells from the schedule, defines which cells will be added/deleted and triggers the allocation/deallocation process.

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

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 http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 3, 2018.

Table of Contents

1. TEMPORARY EDITORIAL NOTES 3
2. Introduction 3
3. Scheduling Function Identifier 4
4. Allocated and Used Cells 4
5. Overprovisioning 4
6. Scheduling Algorithm 4
6.1. SF0 Triggering Events 4
6.2. SF0 Cell Estimation Algorithm 4
6.3. SF0 Allocation Policy 6
7. Rules for CellList 8
8. 6P Timeout Value 8
9. Meaning of Metadata Information 9
10. Node Behavior at Boot 9
11. Cell Type 9
12. SF0 Statistics 9
13. Relocating Cells 9
14. Forced Cell Deletion Policy 10
15. 6P Error Handling 10
16. Examples 10
17. Implementation Status 10
18. Security Considerations 11
19. IANA Considerations 11
20. 6P Compliance 11
21. Acknowledgments 12
22. References 12
22.1. Normative References 12
22.2. Informative References 12
Appendix A. [TEMPORARY] Changelog 13
Authors' Addresses 13

1. TEMPORARY EDITORIAL NOTES

This document is an Internet Draft, so it is 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, etc). The statement does indicate that the text will be written at some time.
- "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.
- The string "REMARK" is put before a remark (questions, suggestion, etc) from an author, editor of contributor. These are on-going discussions at the time to writing, NOT part of the final text.
- This section will be removed in the final text.

2. Introduction

This document defines a minimal Scheduling Function using the 6P protocol [I-D.ietf-6tisch-6top-protocol], called "Scheduling Function Zero" (SF0). SF0 is designed to offer a number of functionalities to be usable in a wide range of applications. SF0 defines two algorithms: The Scheduling Algorithm defines the number of cells to allocate/delete between two neighbours and the Relocation Algorithm defines when to relocate a cell.

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

1. It waits for a triggering event (Section 6.1).
2. It applies the Cell Estimation Algorithm (CEA) for a particular neighbor to determine how many cells are required to that neighbor (Section 6.2).
3. It applies the Allocation Policy to compare the number of required cells to the number of already scheduled cells, and determines the number of cells to add/delete (Section 6.3).

We expect additional SFs, offering more functionalities for a more specific use case, to be defined in future documents. SF0 addresses the requirements for a scheduling function listed in Section 5.2 from [I-D.ietf-6tisch-6top-protocol], and follows the recommended outline listed in Section 5.3 of [I-D.ietf-6tisch-6top-protocol]. This document follows the terminology defined in [I-D.ietf-6tisch-terminology].

3. Scheduling Function Identifier

The Scheduling Function Identifier (SFID) of SF0 is IANA_6TISCH_SFID_SF0.

4. Allocated and Used Cells

An allocated cell is assigned as a TX, RX or Shared cell on the schedule, as a reserved resource. This reservation does not imply that a packet will be transmitted during the scheduled cell time. A used cell is a cell where a packet has been transmitted during the scheduled cell time on the last slotframe.

5. Overprovisioning

Overprovisioning is the action and effect of increasing a value representing an amount of resources. In the case of SF0, overprovisioning is done as a provision to reduce traffic variability effects on packet loss, to the expense of artificially allocating a number of cells.

6. Scheduling Algorithm

A number of TX cells must be allocated between neighbor nodes in order to enable data transmission among them. A portion of these allocated cells will be used by neighbors, while the remaining cells can be over-provisioned to handle unanticipated increases in cell requirements. The Scheduling Algorithm collects the cell allocation/ deallocation requests from the neighbors and the number of cells which are currently under usage. First, the Cell Estimation Algorithm calculates the number of required cells and second, the calculated number is transferred to the Allocation Policy. In order to reduce consumption, this algorithm is triggered only when there is a change on the number of used cells from a particular node.

6.1. SF0 Triggering Events

We RECOMMEND SF0 to be triggered at by the following event: If there is a change on the number of used cells towards any of the neighbours. The exact mechanism of when SF0 is triggered is implementation-specific.

6.2. SF0 Cell Estimation Algorithm

The Cell Estimation Algorithm takes into account the number of current used cells to the neighbour. This allows the algorithm to estimate a new number of cells to be scheduled to the neighbour. As

a consequence, the Cell Estimation Algorithm for SF0 follows the steps described below:

1. Collect the current number of used cells to the neighbour
2. Calculate the new number of cells to be scheduled to the neighbour by adding the current number of used cells plus an OVERPROVISION number of cells
3. Transfer the request to the allocation policy as REQUIREDCELLS
4. Return to step 1 and wait for a triggering event.

The Cell Estimation Algorithm is depicted on figure Figure 1. The OVERPROVISION parameter is calculated as a percentage of the number of currently scheduled cells to the neighbour. OVERPROVISION is added to the amount of used cells to the neighbour to reduce the probability of packet loss given a sudden growth on the number of used cells to the neighbour. The OVERPROVISION value is implementation-specific. A value of OVERPROVISION equal to zero leads to queue growth and possible packet loss: In this case, there are no overprovisioned cells where a sudden growth on the number of cells can absorbed and detected.

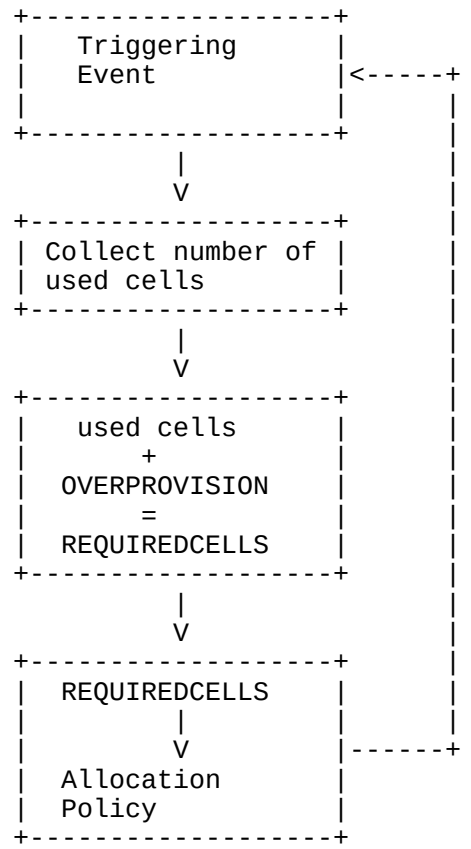


Figure 1: The SF0 Estimation Algorithm

6.3. 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 cell 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 Cell 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' neighbors.

The SF0 allocation policy compares REQUIREDCELLS with SCHEDULEDCELLS and decides to add/delete cells taking into account SF0THRESH. This is illustrated in Figure 2. The number of cells to be added/deleted is out of the scope of this document and it is implementation-dependent.

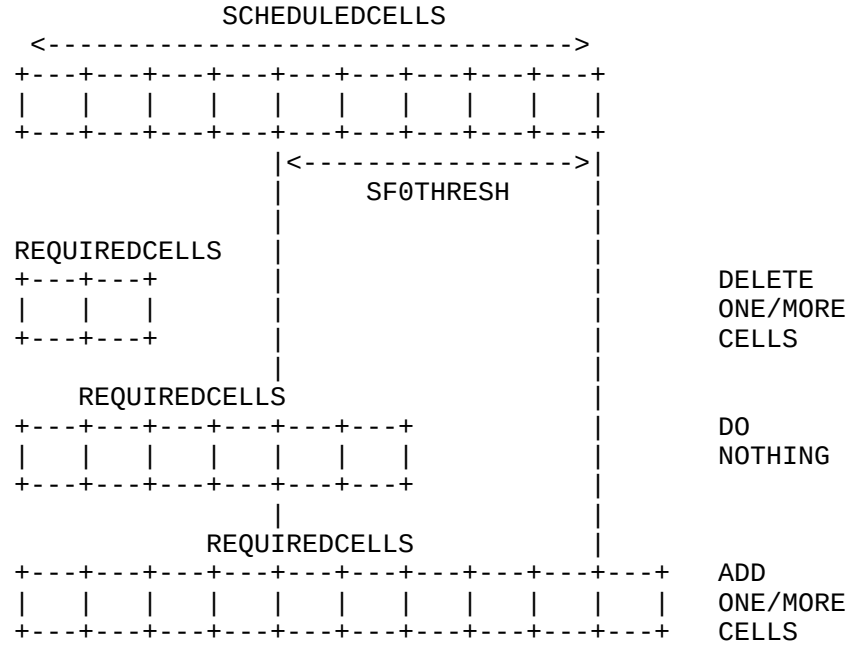


Figure 2: The SF0 Allocation Policy

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 number of cells to add or delete is implementation-specific.

7. Rules for CellList

There are two methods to define the CellList: The Whitelist method, which fills the CellList with the number of proposed cells to the neighbour, and the Blacklist, which fills the CellList with the cells which cannot be used by the neighbour. The rule to select the method is implementation-specific. 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 is 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 is not occupied from the ones on the CellList.

SF0 does not include any transaction retry process. If the transaction is not successful, SF0 will be retriggered on the next slotframe if the number of used cells changes.

8. 6P Timeout Value

The timeout value is implementation-specific. The timeout value MAY be different for each transaction and each neighbour. The timeout range is from 0 to 128. The timeout MUST be added as an 7-bit on the Metadata header to the neighbour. There is no measurement unit associated to the timeout value. If the timeout expires, the node issues a RESET return code will be issued to the neighbour. SF0 has no retry policy. Timeout examples are depicted on Figure 3 and Figure 4.

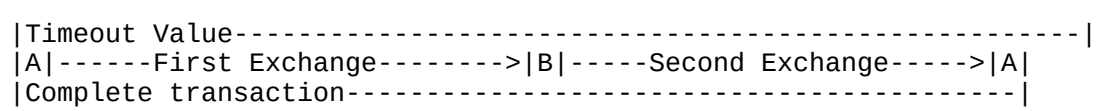


Figure 3: Example Transaction where the timeout does not expire

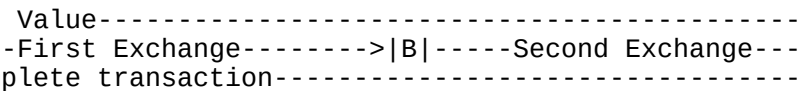


Figure 4: Example Transaction where the timeout expires

9. 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 [TIMEOUT] represents the Timeout value
BIT 15 [WBLIST] is used to indicate that the CellList provided is a Whitelist (value=0) or a Blacklist (value=1).

10. 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 neighbor nodes to enable a new allocation process and at least a SF0THRESH number of cells MUST be allocated to each of the neighbours.

11. Cell Type

SF0 uses TX (Transmission) cell type only, thus defining celloptions as TX=0, RX=1 and S=0 according to section 4.2.6 of [I-D.ietf-6tisch-6top-protocol].

12. SF0 Statistics

Packet Delivery Rate (PDR) is calculated per cell, as the percentage of acknowledged packets, for the last 10 packet transmission attempts. There is no retransmission policy on SF0.

13. Relocating Cells

Allocated cells may experience packet loss from different sources, such as noise, interference or cell collision (after the same cell is allocated by other nodes in range on the network).

SF0 uses Packet Delivery Rate (PDR) statistics to monitor the currently allocated cells for cell relocation (by changing their slotOffset and/or channelOffset). When the PDR of one or more softcells is below PDR_THRESHOLD, SF0 relocates each of the cell(s) to a number of available cells selected randomly. PDR_THRESHOLD is out of the scope of this document and it is implementation-dependent.

14. Forced Cell Deletion Policy

When all the cells are scheduled, we need a policy to free cells, for example, under alarm conditions or if a node disappears from the neighbor list. The action to follow this condition is out of scope of this document and it is implementation-dependent.

15. 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 ADD 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 ADD Request, the neighbor has received the request, but less than NumCells of the cells in the CellList were allocated. 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_ERR_VER:
: The node MUST NOT retry immediately. The node MAY add the neighbor node to a blacklist. The node MAY retry to contact this neighbor later.

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

RC_ERR_GEN:
: The node MUST issue a CLEAR command to the neighbour.

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

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

RC_ERR_RESET:
: Abort 6P Transaction

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

16. Examples

TODO

17. 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.

18. Security Considerations

TODO

19. IANA Considerations

- IANA_6TiSCH_SFID_SF0

20. 6P Compliance

- MUST specify an identifier for that SF. OK
- MUST specify the rule for a node to decide when to add/delete one or more cells to a neighbor. OK
- MUST specify the rule for a Transaction source to select cells to add to the CellList field in the 6P ADD Request. OK
- MUST specify the rule for a Transaction destination to select cells from CellList to add to its schedule. OK
- MUST specify a value for the 6P Timeout, or a rule/equation to calculate it. OK
- MUST specify a meaning for the "Metadata" field in the 6P ADD Request. OK
- MUST specify the behavior of a node when it boots. OK
- MUST specify what to do after an error has occurred (either the node sent a 6P Response with an error code, or received one). OK
- 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. OK
- SHOULD clearly state the application domain the SF is created for. OK
- SHOULD contain examples which highlight normal and error scenarios.
- SHOULD contain a list of current implementations, at least during the I-D state of the document, per [RFC6982].
- SHOULD contain a performance evaluation of the scheme, possibly through references to external documents.
- MAY redefine the format of the CellList? field. OK

21. 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).

22. References

22.1. 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>.

22.2. Informative References

[I-D.ietf-6tisch-6top-protocol] Wang, Q., Vilajosana, X., and T. Watteyne, "6top Protocol (6P)", draft-ietf-6tisch-6top-protocol-07 (work in progress), June 2017.

[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-09 (work in progress), June 2017.

[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.

[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>.

Appendix A. [TEMPORARY] Changelog

- draft-ietf-6tisch-6top-sf0-02
  - Editorial changes (figs, typos, ...)
- draft-ietf-6tisch-6top-sf0-03
  - Fixed typos
  - Removed references to "effectively used cells"
  - Changed Cell Estimation Algorithm to the third proposed alternative on IETF97
  - Forced cell deletion becomes implementation specific
  - Added PDR calculation formula
  - Added PDR_THRESHOLD as implementation specific value

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
Luxembourg Institute of Science and Technology (LIST)
Department 'Environmental Research and Innovation' (ERIN)
41, rue du Brill
Belvaux L-4422
Grand-duchy of Luxembourg

Phone: +352 275 888-5055
Email: mariarita.palattella@list.lu

Nicola Accettura
LAAS-CNRS
7, avenue du Colonel Roche
Toulouse 31400
France

Phone: +33 5 61 33 69 76
Email: nicola.accettura@laas.fr

6TiSCH
Internet-Draft
Intended status: Informational
Expires: July 31, 2017

P. Thubert, Ed.
Cisco

January 27, 2017

An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4
draft-ietf-6tisch-architecture-11

Abstract

This document describes a network architecture that provides low-latency, low-jitter and high-reliability packet delivery. It combines a high speed powered backbone and subnetworks using IEEE 802.15.4 time-slotted channel hopping (TSCH) to meet the requirements of LowPower wireless deterministic applications.

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 http://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 July 31, 2017.

Table of Contents

1. Introduction 3
2. Terminology 4
3. High Level Architecture 5
3.1. 6TiSCH Stack 5
3.2. TSCH: A Deterministic MAC Layer 6
3.3. Scheduling TSCH 7
3.4. Routing and Forwarding Over TSCH 9
3.5. A Non-Broadcast Multi-Access Radio Mesh Network 10
3.6. A Multi-Link Subnet Model 12
3.7. Join Process and Registration 13
3.8. Dependencies on Work In Progress 14
4. Architecture Components 16
4.1. 6LoWPAN (and RPL) 16
4.1.1. RPL Leaf Support in 6LoWPAN ND 16
4.1.2. RPL Root And 6LBR 17
4.2. TSCH and 6top 18
4.2.1. 6top 18
4.2.2. Scheduling Functions and the 6P protocol 18
4.2.3. 6top and RPL Objective Function operations 19
4.2.4. Network Synchronization 20
4.2.5. SlotFrames and Priorities 21
4.2.6. Distributing the reservation of cells 22
4.3. Communication Paradigms and Interaction Models 24
4.4. Schedule Management Mechanisms 25
4.4.1. Static Scheduling 25
4.4.2. Neighbor-to-neighbor Scheduling 26
4.4.3. Remote Monitoring and Schedule Management 26
4.4.4. Hop-by-hop Scheduling 29
4.5. On Tracks 29
4.5.1. General Behavior of Tracks 29
4.5.2. Serial Track 30
4.5.3. Complex Track with Replication and Elimination . . . 31
4.5.4. DetNet End-to-end Path 31
4.5.5. Cell Reuse 32
4.6. Forwarding Models 33
4.6.1. Track Forwarding 33
4.6.2. Fragment Forwarding 36
4.6.3. IPv6 Forwarding 37
4.7. Centralized vs. Distributed Routing 38
4.7.1. Packet Marking and Handling 38
4.7.2. Replication, Retries and Elimination 39
4.7.3. Differentiated Services Per-Hop-Behavior 40
5. IANA Considerations 40
6. Security Considerations 40
6.1. Join Process Highlights 40
7. Acknowledgments 43

7.1. Contributors 43
7.2. Special Thanks 43
7.3. And Do not Forget 44
8. References 44
8.1. Normative References 44
8.2. Informative References 46
8.3. Other Informative References 51
Appendix A. Personal submissions relevant to upcoming work . . . 52
Author's Address 53

1. Introduction

Wireless Networks enable a wide variety of devices of any size to get interconnected, often at a very low marginal cost per device, at any distance ranging from Near Field to interplanetary, and in circumstances where wiring may be impractical, for instance on fast-moving or rotating devices.

In the other hand, Deterministic Networks enable traffic that is highly sensitive to jitter, quite sensitive to latency, and with a high degree of operational criticality so that loss should be minimized at all times. Applications that need such networks are presented in [I-D.ietf-detnet-use-cases]. They include Professional Media and Operation Technology (OT) Industrial Automation Control Systems (IACS).

The Medium access Control (MAC) of IEEE std 802.15.4 [IEEE802154] has evolved with the IEEE std 802.15.4e Timeslotted Channel Hopping (TSCH) [RFC7554] mode to provide deterministic properties on wireless networks. TSCH was initially introduced with the IEEE std 802.15.4e amendment [IEEE802154e] of the IEEE std 802.15.4 standard and constituted a part of the standard from that day. For all practical purpose, this document is expected to be insensitive to the revisions of the IEEE std 802.15.4 standard, which is thus referenced undated.

Proven Deterministic Networking standards for use in Process Control, including ISA100.11a [ISA100.11a] and WirelessHART [WirelessHART], have demonstrated the capabilities of the IEEE std 802.15.4 TSCH MAC for high reliability against interference, low-power consumption on well-known flows, and its applicability for Traffic Engineering (TE) from a central controller.

In order to enable the convergence of IT and OT in LLN environments, 6TiSCH ports the IETF suite of protocol that are defined for such environments over the TSCH MAC. 6TiSCH also provides large scaling capabilities, which, in a number of scenarios, require the addition of a high speed and reliable backbone and the use of IP version 6 (IPv6). The 6TiSCH Architecture introduces an IPv6 Multi-Link subnet

model that is composed of a federating backbone and a number of IEEE std 802.15.4 TSCH low-power wireless networks attached and synchronized by Backbone Routers.

The architecture defines mechanisms to establish and maintain routing and scheduling in a centralized, distributed, or mixed fashion, for use in multiple OT environments. It is applicable in particular to industrial control systems, building automation that leverage distributed routing to address multipath over a large number of hops, in-vehicle command and control that can be as demanding as industrial applications, commercial automation and asset Tracking with mobile scenarios, home automation and domotics which become more reliable and thus provide a better user experience, and resource management (energy, water, etc.).

2. Terminology

The draft uses domain-specific terminology defined or referenced in [I-D.ietf-6tisch-terminology], [I-D.ietf-6lo-backbone-router], and [I-D.ietf-roll-rpl-industrial-applicability].

Readers are expected to be familiar with all the terms and concepts that are discussed in "Neighbor Discovery for IP version 6" [RFC4861], "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals" [RFC4919], and Neighbor Discovery Optimization for Low-power and Lossy Networks [RFC6775] where the 6LoWPAN Router (6LR) and the 6LoWPAN Border Router (6LBR) are introduced.

Readers may benefit from reading the "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks" [RFC6550] specification; "Multi-Link Subnet Issues" [RFC4903]; "Mobility Support in IPv6" [RFC6275]; "Neighbor Discovery Proxies (ND Proxy)" [RFC4389]; "IPv6 Stateless Address Autoconfiguration" [RFC4862]; "FCFS SAVI: First-Come, First-Served Source Address Validation Improvement for Locally Assigned IPv6 Addresses" [RFC6620]; and "Optimistic Duplicate Address Detection" [RFC4429] prior to this specification for a clear understanding of the art in ND-proxying and binding.

The draft also conforms to the terms and models described in [RFC3444] and [RFC5889] and uses the vocabulary and the concepts defined in [RFC4291] for the IPv6 Architecture and refers [RFC4080] for reservation signaling and [RFC5191] for authentication.

3. High Level Architecture

3.1. 6TiSCH Stack

The 6TiSCH architecture presents a reference stack that is implemented and interop tested by a conjunction of opensource, IETF and ETSI efforts. One goal is to help other bodies to adopt the stack as a whole, making the effort to move to an IPv6-based IOT stack easier. Now, for a particular, environment, some of the choices that are made in this architecture may not be relevant. For instance, RPL is not required for star topologies and mesh-under Layer-2 routed networks, and the 6LoWPAN compression may not be sufficient for ultra-constrained cases such as some Low Power Wide Area (LPWA) networks. In such cases, it is perfectly doable to adopt a subset of the selection that is presented hereafter and then select alternate components to complete the solution wherever needed.

The IETF proposes multiple techniques for implementing functions related to routing, transport or security. In order to control the complexity of the possible deployments and device interactions, and to limit the size of the resulting object code, the architecture limits the possible variations of the stack and recommends a number of base elements for LLN applications. In particular, UDP [RFC0768] [RFC2460] and the Constrained Application Protocol [RFC7252] (CoAP) are used as the transport / binding of choice for applications and management as opposed to TCP and HTTP.

The resulting stack is represented below:

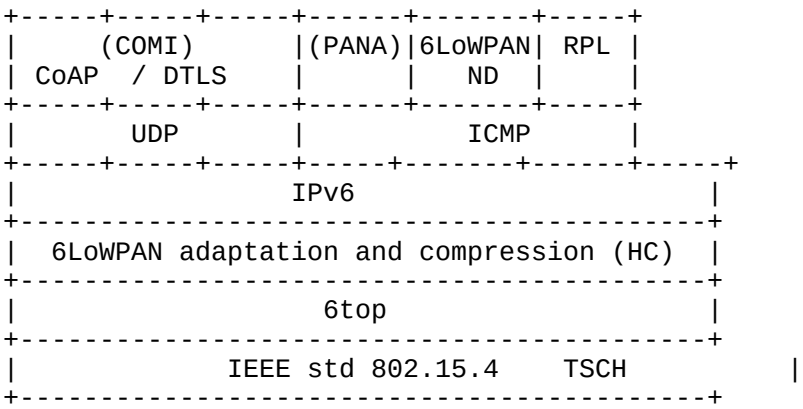


Figure 1: 6TiSCH Protocol Stack

RPL is the routing protocol of choice for LLNs. So far, there was no identified need to define a 6TiSCH specific Objective Function. The Minimal 6TiSCH Configuration [I-D.ietf-6tisch-minimal] describes the operation of RPL over a static schedule used in a slotted aloha fashion, whereby all active slots may be used for emission or reception of both unicast and multicast frames.

The 6LoWPAN Header Compression [RFC6282] is used to compress the IPv6 and UDP headers, whereas the 6LoWPAN Routing Header [I-D.ietf-roll-routing-dispatch] is used to compress the RPL artifacts in the IPv6 data packets, including the RPL Packet Information (RPI), the IP-in-IP encapsulation to/from the RPL root, and the Source Route Header (SRH) in non-storing mode.

6TiSCH has adopted the general direction of CoAP Management Interface (COMI) [I-D.vanderstok-core-comi] for the management of devices. This is leveraged for instance for the implementation of the generic data model for the 6top sublayer management interface [I-D.ietf-6tisch-6top-interface]. The proposed implementation is based on CoAP and CBOR, and specified in 6TiSCH Resource Management and Interaction using CoAP [I-D.ietf-6tisch-coap].

The Datagram Transport Layer Security (DTLS) [RFC6347] is represented as an example of a protocol that could be used to protect CoAP datagrams, but the exact stack is not determined at the time of this writing..

Similarly, the Protocol for Carrying Authentication for Network access (PANA) [RFC5191] is represented as an example of a protocol that could be leveraged to secure the join process, as a Layer-3 alternate to IEEE std 802.1x/EAP. Regardless, the security model ensures that, prior to a join process, packets from a untrusted device are controlled in volume and in reachability. In particular, a PANA stack should be separated from the main protocol stack to avoid attacks during the join process that is introduced in Section 3.7. An overview of the security aspects of the join process can be found in Section 6.

The 6TiSCH Operation sublayer (6top) [I-D.wang-6tisch-6top-sublayer] is a sublayer of a Logical Link Control (LLC) that provides the abstraction of an IP link over a TSCH MAC and schedules packets over TSCH cells,as further discussed in the next sections.

3.2. TSCH: A Deterministic MAC Layer

Though at a different time scale (several orders of magnitude), both IEEE std 802.1TSN and IEEE std 802.15.4TSCH standards provide Deterministic capabilities to the point that a packet that pertains

to a certain flow may traverse a network from node to node following a very precise schedule, as a train that enters and then leaves intermediate stations at precise times along its path. With TSCH, time is formatted into timeslots, and individual communication cells are allocated to unicast or broadcast communication at the MAC level. The time-slotted operation reduces collisions, saves energy, and enables to more closely engineer the network for deterministic properties. The channel hopping aspect is a simple and efficient technique to combat multipath fading and external interference (for example by Wi-Fi emitters).

6TiSCH builds on the IEEE std 802.15.4TSCH MAC and inherits its advanced capabilities to enable them in multiple environments where they can be leveraged to improve automated operations. The 6TiSCH Architecture also inherits the capability to perform a centralized route computation to achieve deterministic properties, though it relies on the IETF DetNet Architecture [I-D.ietf-detnet-architecture], and IETF components such as the Path Computation Element (PCE) [PCE], for the protocol aspects.

On top of this inheritance, 6TiSCH adds capabilities for distributed routing and scheduling operations based on the RPL routing protocol and capabilities to negotiate schedule adjustments between peers. These distributed routing and scheduling operations simplify the deployment of TSCH networks and enable wireless solutions in a larger variety of use cases from operational technology in general. Examples of such use-cases in industrial environments include plant setup and decommissioning, as well as monitoring of lots of lesser importance measurements such as corrosion and events. RPL also enables mobile use cases such as mobile workers and cranes, as presented in [I-D.ietf-roll-rpl-industrial-applicability].

3.3. Scheduling TSCH

A scheduling operation attributes cells in a Time-Division-Multiplexing (TDM) / Frequency-Division Multiplexing (FDM) matrix called the Channel distribution/usage (CDU) to either individual transmissions or as multi-access shared resources (see the 6TiSCH Terminology [I-D.ietf-6tisch-terminology] for more on these terms). Scheduling effectively enables multiple communications at a same time in a same interference domain using different channels; but a node equipped with a single radio can only transmit or receive on one channel at any given point of time.

From the standpoint of a 6TiSCH node (at the MAC layer), its schedule is the collection of the times at which it must wake up for transmission, and the channels to which it should either send or listen at those times. The schedule is expressed as one or more

slotframes that repeat over and over. Slotframes may collision and require a device to wake at a same time, in which case a priority indicates which slotframe is actually activated.

The 6top sublayer hides the complexity of the schedule to the upper layers. The Link that IP may utilize between the 6TiSCH node and a peer may in fact be composed of a pair of cell bundles, one to receive and one to transmit. Some of the cells may be shared, in which case the 6top sublayer must perform some arbitration.

The 6TiSCH architecture identifies four ways a schedule can be managed and CDU cells can be allocated: Static Scheduling, Neighbor-to-Neighbor Scheduling, Remote Monitoring and Schedule Management, and Hop-by-hop Scheduling.

Static Scheduling: This refers to the minimal 6TiSCH operation whereby a static schedule is configured for the whole network for use in a slotted-aloha fashion. The static schedule is distributed through the native methods in the TSCH MAC layer. This operation leverages RPL to maintain a loopless graph for routing and time distribution. It is specified in the Minimal 6TiSCH Configuration [I-D.ietf-6tisch-minimal] specification. and does not preclude other scheduling operations to co-exist on a same 6TiSCH network.

Neighbor-to-Neighbor Scheduling: This refers to the dynamic adaptation of the bandwidth of the Links that are used for IPv6 traffic between adjacent routers. Scheduling Functions such as SF0 [I-D.ietf-6tisch-6top-sf0] influence the operation of the 6top sublayer [I-D.wang-6tisch-6top-sublayer] to add and remove cells in peers schedule, using the 6top protocol [I-D.ietf-6tisch-6top-protocol] for the negotiation on the MAC resources.

Remote Monitoring and Schedule Management: This refers to the central computation of a schedule and the capability to forward a frame based on the cell of arrival. In that case, the related portion of the device schedule as well as other device resources are managed by an abstract Network Management Entity (NME), which may cooperate with the PCE in order to minimize the interaction with and the load on the constrained device. This model is the TSCH adaption of the DetNet Architecture [I-D.ietf-detnet-architecture], and it enables Traffic Engineering with deterministic properties.

Hop-by-hop Scheduling: This refers to the possibility to reserves cells along a path for a particular flow using a distributed mechanism.

It is not expected that all use cases will require all those mechanisms. Static Scheduling with minimal configuration one is the only one that is expected in all implementations, since it provides a simple and solid basis for convergecast routing and time distribution.

A deeper dive in those mechanisms can be found in Section 4.4.

3.4. Routing and Forwarding Over TSCH

6TiSCH leverages the RPL routing protocol for interoperable distributed routing operations. RPL is applicable to Static Scheduling and Neighbor-to-Neighbor Scheduling. The architecture also supports a centralized routing model for Remote Monitoring and Schedule Management. It is expected that a routing protocol that is more optimized for point-to-point routing than RPL, such as the Reactive Discovery of Point-to-Point Routes in Low-Power and Lossy Networks [RFC6997](P2P RPL), or the Ad Hoc On-demand Distance Vector Routing (AODV) [I-D.ietf-manet-aodvv2] will be selected for Hop-by-hop Scheduling.

The 6TiSCH architecture supports three different forwarding models, the classical IPv6 Forwarding, where the node selects a feasible successor at Layer-3 on a per packet basis and based on its routing table, G-MPLS Track Forwarding, which switches a frame received at a particular Timeslot into another Timeslot at Layer-2, and 6LoWPAN Fragment Forwarding, which allows to forward individual 6loWPAN fragments along the route set by the first fragment.

IPv6 Forwarding: This is the classical IP forwarding model, with a Routing Information Based (RIB) that is installed by the RPL routing protocol and used to select a feasible successor per packet. The packet is placed on an outgoing Link, that the 6top layer maps into a (Layer-3) bundle of cells, and scheduled for transmission based on QoS parameters. On top of RPL, this model also applies to any routing protocol which may be operated in the 6TiSCH network, and corresponds to all the distributed scheduling models, Static, Neighbor-to-Neighbor and Hop-by-Hop Scheduling.

G-MPLS Track Forwarding: This model corresponds to the Remote Monitoring and Schedule Management. In this model, A central controller (hosting a PCE) computes and installs the schedules in the devices per flow. The incoming (Layer-2) bundle of cells from the previous node along the path determines the outgoing (Layer-2) bundle towards the next hop for that flow as determined by the PCE. The programmed sequence for bundles is called a Track and can assume shapes that are more complex than a simple direct sequence of nodes.

6LoWPAN Fragment Forwarding: This is an hybrid model that derives from IPv6 forwarding for the case where packets must be fragmented at the 6LoWPAN sublayer. The first fragment is forwarded like any IPv6 packet and leaves a state in the intermediate hops to enable forwarding of the next fragments that do not have a IP header without the need to recompose the packet at every hop.

This can be broadly summarized in the following table:

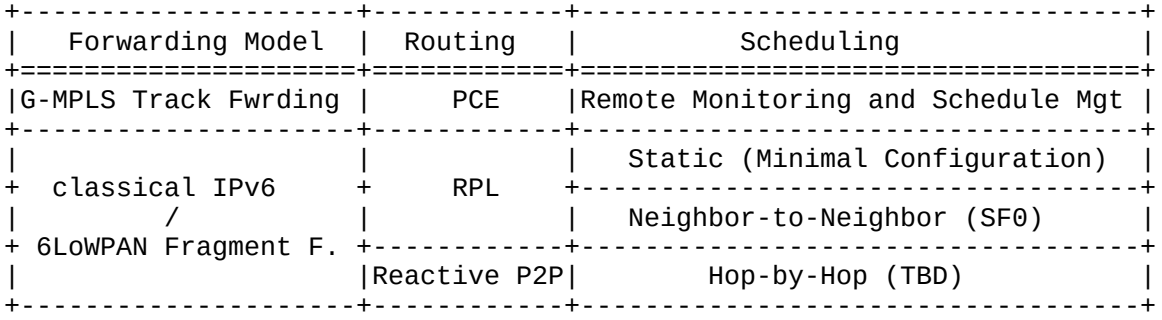


Figure 2: Routing, Forwarding and Scheduling

3.5. A Non-Broadcast Multi-Access Radio Mesh Network

A 6TiSCH network is an IPv6 [RFC2460] subnet which, in its basic configuration, is a single Low Power Lossy Network (LLN) operating over a synchronized TSCH-based mesh.

Inside a 6TiSCH LLN, nodes rely on 6LoWPAN Header Compression (6LoWPAN HC) [RFC6282] to encode IPv6 packets. From the perspective of the network layer, a single LLN interface (typically an IEEE std 802.15.4-compliant radio) may be seen as a collection of Links with different capabilities for unicast or multicast services.

6TiSCH nodes are not necessarily reachable from one another at Layer-2 and an LLN may span over multiple links. This effectively forms an homogeneous non-broadcast multi-access (NBMA) subnet, which is beyond the scope of existing IPv6 ND methods. Extensions to IPv6 ND have to be introduced.

Within that subnet, neighbor devices are discovered with 6LoWPAN Neighbor Discovery [RFC6775] (6LoWPAN ND), whereas RPL [RFC6550] enables routing in the so called Route Over fashion, either in storing (stateful) or non-storing (stateless, with routing headers) mode.

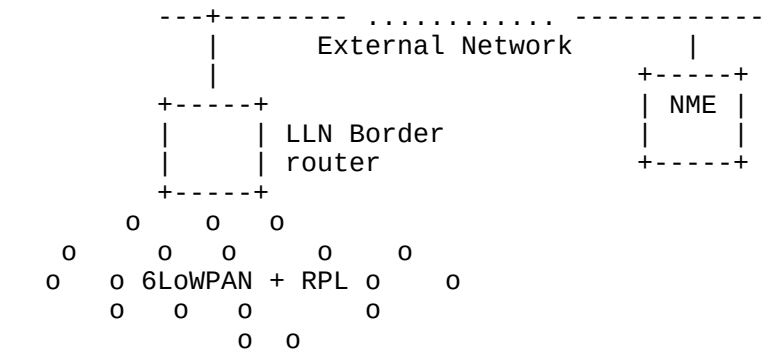


Figure 3: Basic Configuration of a 6TiSCH Network

6TiSCH nodes join the mesh by attaching to nodes that are already members of the mesh. Some nodes act as routers for 6LoWPAN ND and RPL operations, as detailed in Section 4.1. Security aspects of the join process by which a device obtains access to the network are discussed in Section 6.

With TSCH, devices are time-synchronized at the MAC level. The use of a particular RPL Instance for time synchronization is discussed in Section 4.2.4. With this mechanism, the time synchronization starts at the RPL root and follows the RPL DODAGs with no timing loop.

RPL forms Destination Oriented Directed Acyclic Graphs (DODAGs) within Instances of the protocol, each Instance being associated with an Objective Function (OF) to form a routing topology. A particular 6TiSCH node, the LLN Border Router (LBR), acts as RPL root, 6LoWPAN HC terminator, and Border Router for the LLN to the outside. The LBR is usually powered. More on RPL Instances can be found in section 3.1 of RPL [RFC6550], in particular "3.1.2. RPL Identifiers" and "3.1.3. Instances, DODAGs, and DODAG Versions". RPL adds artifacts in the data packets that are compressed with a 6LoWPAN addition 6LoRH [I-D.ietf-roll-routing-dispatch].

Additional routing and scheduling protocols may be deployed to establish on-demand Peer-to-Peer routes with particular characteristics inside the 6TiSCH network. This may be achieved in a centralized fashion by a PCE [PCE] that programs both the routes and the schedules inside the 6TiSCH nodes, or by in a distributed fashion using a reactive routing protocol and a Hop-by-Hop scheduling protocol.

A Backbone Router may be connected to the node that acts as RPL root and / or 6LoWPAN 6LBR and provides connectivity to the larger campus / factory plant network over a high speed backbone or a back-haul

link. A Backbone Router may perform proxy IPv6 Neighbor Discovery (ND) [RFC4861] operations over the backbone on behalf of the 6TiSCH nodes so they can share a same IPv6 subnet and appear to be connected to the same backbone as classical devices. A Backbone Router may alternatively redistribute the registration in a routing protocol such as OSPF [RFC5340] or BGP [RFC2545], or inject them in a mobility protocol such as MIPv6 [RFC6275], NEMO [RFC3963], or LISP [RFC6830].

This architecture expects that a 6LoWPAN node can connect as a leaf to a RPL network, where the leaf support is the minimal functionality to connect as a host to a RPL network without the need to participate to the full routing protocol. The architecture also expects that a 6LoWPAN node that is not aware at all of the RPL protocol may also connect as a host but the specifications for this to happen are not available at the time of this writing.

3.6. A Multi-Link Subnet Model

An extended configuration of the subnet comprises multiple LLNs. The LLNs are interconnected and synchronized over a backbone, that can be wired or wireless. The backbone can be a classical IPv6 network, with Neighbor Discovery operating as defined in [RFC4861] and [RFC4862]. This architecture requires work to standardize the the registration of 6LoWPAN nodes to the Backbone Routers.

In the extended configuration, a Backbone Router (6BBR) operates as described in [I-D.ietf-6lo-backbone-router]. The 6BBR performs ND proxy operations between the registered devices and the classical ND devices that are located over the backbone. 6TiSCH 6BBRs synchronize with one another over the backbone, so as to ensure that the multiple LLNs that form the IPv6 subnet stay tightly synchronized.

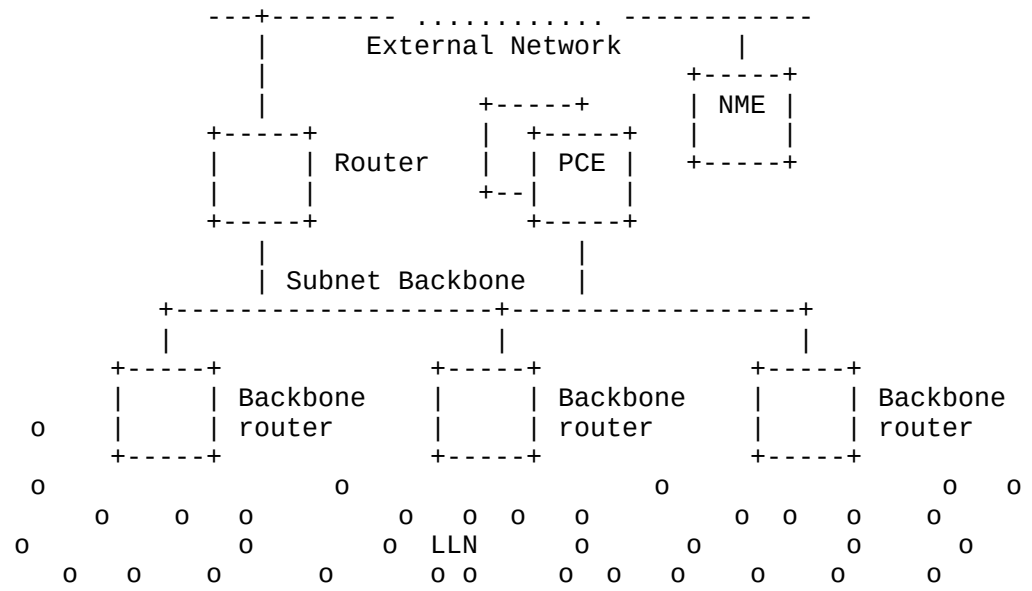


Figure 4: Extended Configuration of a 6TiSCH Network

As detailed in Section 4.1 the 6LoWPAN ND 6LBR and the root of the RPL network need to be collocated and share information about the devices that is learned through either protocol but not both. The combined RPL root and 6LBR may be collocated with the 6BBR, or directly attached to the 6BBR. In the latter case, it leverages the extended registration process defined in [I-D.ietf-6lo-backbone-router] to proxy the 6LoWPAN ND registration to the 6BBR on behalf of the LLN nodes, so that the 6BBR may in turn perform proxy classical ND operations over the backbone.

If the Backbone is Deterministic (such as defined by the Time Sensitive Networking WG at IEEE), then the Backbone Router ensures that the end-to-end deterministic behavior is maintained between the LLN and the backbone. The DetNet Architecture [I-D.ietf-detnet-architecture] studies Layer-3 aspects of Deterministic Networks, and covers networks that span multiple Layer-2 domains.

3.7. Join Process and Registration

As detailed in Section 4.1 the combined 6LoWPAN ND 6LBR and root of the RPL network learn information such as the device Unique ID (from 6LoWPAN ND) and the updated Sequence Number (from RPL), and perform 6LoWPAN ND proxy registration to the 6BBR of behalf of the LLN nodes. Figure 5 illustrates the periodic signaling that starts at the leaf

node with 6LoWPAN ND, is then carried over RPL to the RPL root, and then to the 6BBR. Efficient ND being an adaptation of 6LoWPAN ND, it makes sense to keep those two homogeneous in the way they use the source and the target addresses in the Neighbor Solicitation (NS) messages for registration, as well as in the options that they use for that process.

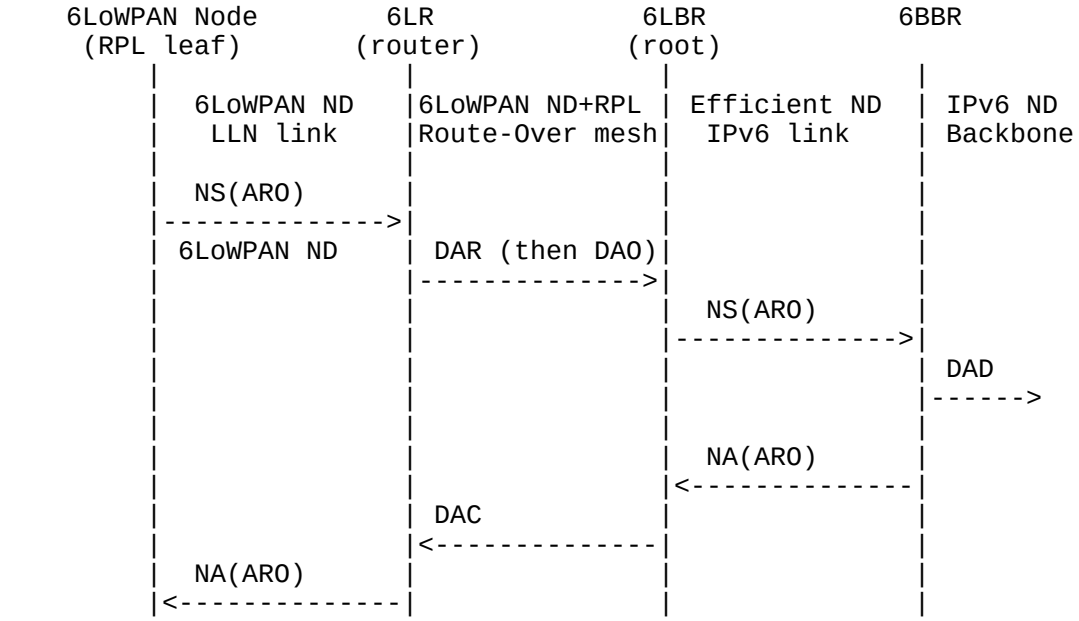


Figure 5: (Re-)Registration Flow over Multi-Link Subnet

As the network builds up, a node should start as a leaf to join the RPL network, and may later turn into both a RPL-capable router and a 6LR, so as to accept leaf nodes to recursively join the network.

3.8. Dependencies on Work In Progress

In order to control the complexity and the size of the 6TiSCH work, the architecture and the associated IETF work are staged and the WG is expected to recharter multiple times. This document is incremented as the work progresses following the evolution of the WG charter and the availability of dependent work. The intent is to publish when the WG concludes.

At the time of this writing:

- The architecture of the operation of RPL over a dynamic schedule is being studied at 6TISCH as the second iteration of the charter.

- The need of a reactive routing protocol to establish on-demand constraint-optimized routes and a reservation protocol to establish Layer-3 Tracks is being discussed at 6TiSCH but not chartered for.

- The components and protocols that are required to implement this stage of architecture are being standardized at the IETF. An Update to 6LoWPAN ND [I-D.thubert-6lo-rfc6775-update] covers the evolution of 6LoWPAN Neighbor Discovery that is needed to implement the Backbone Router [I-D.ietf-6lo-backbone-router]. In addition the protection of registered addresses against impersonation and take over can be guaranteed by Address Protected Neighbor Discovery for Low-power and Lossy Networks [I-D.ietf-6lo-ap-nd].

- The work on centralized Track computation is deferred to a subsequent iteration of the 6TiSCH charter. The idea at the time of this writing is that 6TiSCH will apply the concepts of Deterministic Networking on a Layer-3 network. The 6TiSCH Architecture should thus inherit from the DetNet [I-D.ietf-detnet-architecture] architecture and thus depends on it. The Path Computation Element (PCE) should be a core component of that architecture. Around the PCE, a protocol such as an extension to a TEAS [TEAS] protocol will be required to expose the 6TiSCH node capabilities and the network peers to the PCE, and a protocol such as a lightweight PCEP or an adaptation of CCAMP [CCAMP] G-MPLS formats and procedures will be used to publish the Tracks, as computed by the PCE, to the 6TiSCH nodes.

- BIER-TE-based OAM, Replication and Elimination [I-D.thubert-bier-replication-elimination] leverages Bit Index Explicit Replication - Traffic Engineering to control in the data plane the DetNet Replication and Elimination activities, and to provide traceability on links where replication and loss happen, in a manner that is abstract to the forwarding information, whereas a 6loRH for BitStrings [I-D.thubert-6lo-bier-dispatch] proposes a 6LoWPAN compression for the BIER Bitstring based on 6LoWPAN Routing Header [I-D.ietf-roll-routing-dispatch].

- The security model and in particular the join process are being discussed at 6lo and 6TiSCH. PANA is presented in Section 3.1 as a candidate of choice for the join process but alternatives are discussed. Work resulting from [ACE] could be considered as well. Related contributions are presented in Appendix A.

- The current charter positions 6TiSCH on IEEE std 802.15.4 only. Though most of the design should be portable on other link types, 6TiSCH has a strong dependency on IEEE std 802.15.4 and its evolution. At the time of this writing, a revision of the IEEE std 802.15.4 standard is expected early 2016. That revision should integrate TSCH as well as other amendments and fixes into the main specification. The impact on this Architecture should be minimal to non-existent, but deeper work such as 6top and security may be impacted. A 6TiSCH Interest Group was formed at IEEE to maintain the synchronization and help foster work at the IEEE should 6TiSCH demand it.

- Work is being proposed at IEEE (802.15.12 PAR) for an LLC that would logically include the 6top sublayer. The interaction with the 6top sublayer and the Scheduling Functions described in this document are yet to be defined.

- ISA100 [ISA100] Common Network Management (CNM) is another external work of interest for 6TiSCH. The group, referred to as ISA100.20, defines a Common Network Management framework that should enable the management of resources that are controlled by heterogeneous protocols such as ISA100.11a [ISA100.11a], WirelessHART [WirelessHART], and 6TiSCH. Interestingly, the establishment of 6TiSCH Deterministic paths, called Tracks, are also in scope, and ISA100.20 is working on requirements for DetNet.

4. Architecture Components

4.1. 6LoWPAN (and RPL)

4.1.1. RPL Leaf Support in 6LoWPAN ND

RPL needs a set of information in order to advertise a leaf node through a DAO message and establish reachability.

At the bare minimum the leaf device must provide a sequence number that matches the RPL specification in section 7. Section 5.3 of [I-D.ietf-6lo-backbone-router], on the Extended Address Registration Option (EARO), already incorporates that addition with a new field in the option called the Transaction ID.

If for some reason the node is aware of RPL topologies, then providing the RPL InstanceID for the instances to which the node wishes to participate would be a welcome addition. In the absence of such information, the RPL router must infer the proper instanceID from external rules and policies.

On the backbone, the InstanceID is expected to be mapped onto a an overlay that matches the instanceID, for instance a VLANID.

This architecture leverages [I-D.ietf-6lo-backbone-router] that extends 6LoWPAN ND [RFC6775] to carry the counter as an abstract Transaction ID (TID).

4.1.2. RPL Root And 6LBR

6LoWPAN ND is unclear on how the 6LBR is discovered, and how the liveliness of the 6LBR is asserted over time. On the other hand, the discovery and liveliness of the RPL root are obtained through the RPL protocol. This architecture suggests to collocate these functions by default, in which case the discovery of the 6LBR is automatic for RPL leaves.

When 6LoWPAN ND is coupled with RPL, the 6LBR and RPL root functionalities are co-located in order that the address of the 6LBR be indicated by RPL DIO messages and to associate the unique ID from the DAR/DAC exchange with the state that is maintained by RPL. The DAR/DAC exchange becomes a preamble to the DAO messages that are used from then on to reconfirm the registration, thus eliminating a duplication of functionality between DAO and DAR messages.

Even though the root of the RPL network is integrated with the 6LBR, it is logically separated from the Backbone Router (6BBR) that is used to connect the 6TiSCH LLN to the backbone. This way, the root has all information from 6LoWPAN ND and RPL about the LLN devices attached to it.

This architecture also expects that the root of the RPL network (proxy-)registers the 6TiSCH nodes on their behalf to the 6BBR, for whatever operation the 6BBR performs on the backbone, such as ND proxy, or redistribution in a routing protocol. This relies on an extension of the 6LoWPAN ND registration described in [I-D.ietf-6lo-backbone-router].

This model supports the movement of a 6TiSCH device across the Multi-Link Subnet, and allows the proxy registration of 6TiSCH nodes deep into the 6TiSCH LLN by the 6LBR / RPL root. This requires an alteration from [RFC6775] whereby the Target Address of the NS message is registered as opposed to the Source, which, in the case of a proxy registration, is that of the 6LBR / RPL root itself.

4.2. TSCH and 6top

4.2.1. 6top

6top is a logical link control sitting between the IP layer and the TSCH MAC layer, which provides the link abstraction that is required for IP operations. The 6top operations are specified in [I-D.ietf-6tisch-6top-protocol]. In particular, 6top provides a management interface that enables an external management entity to schedule cells and slotFrames, and allows the addition of complementary functionality, for instance to support a dynamic schedule management based on observed resource usage as discussed in Section 4.4.2.

The 6top data model and management interfaces are further discussed in Section 4.4.3.

4.2.1.1. Hard Cells

The architecture defines "soft" cells and "hard" cells. "Hard" cells are owned and managed by an separate scheduling entity (e.g. a PCE) that specifies the slotOffset/channelOffset of the cells to be added/moved/deleted, in which case 6top can only act as instructed, and may not move hard cells in the TSCH schedule on its own.

4.2.1.2. Soft Cells

6top contains a monitoring process which monitors the performance of cells, and can move a cell in the TSCH schedule when it performs poorly. This is only applicable to cells which are marked as "soft". To reserve a soft cell, the higher layer does not indicate the exact slotOffset/channelOffset of the cell to add, but rather the resulting bandwidth and QoS requirements. When the monitoring process triggers a cell reallocation, the two neighbor devices communicating over this cell negotiate its new position in the TSCH schedule.

4.2.2. Scheduling Functions and the 6P protocol

In the case of soft cells, the cell management entity that controls the dynamic attribution of cells to adapt to the dynamics of variable rate flows is called a Scheduling Function (SF). There may be multiple SFs with more or less aggressive reaction to the dynamics of the network. The 6TiSCH 6top Scheduling Function Zero (SF0) [I-D.ietf-6tisch-6top-sf0] provides a simple scheduling function that can be used by default by devices that support dynamic scheduling of soft cells.

The SF may be seen as divided between an upper bandwidth adaptation logic that is not aware of the particular technology that is used to obtain and release bandwidth, and an underlying service that maps those needs in the actual technology, which means mapping the bandwidth onto cells in the case of TSCH.

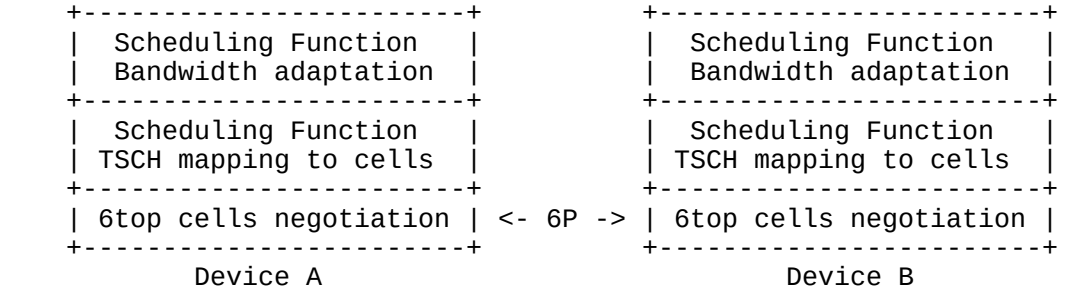


Figure 6: SF/6P stack in 6top

The SF relies on 6top services that implement the 6top Protocol (6P) [I-D.ietf-6tisch-6top-protocol] to negotiate the precise cells that will be allocated or freed based on the schedule of the peer. It may be for instance that a peer wants to use a particular time slot that is free in its schedule, but that timeslot is already in use by the other peer for a communication with a third party on a different cell. The 6P protocol enables the peers to find an agreement in a transactional manner that ensures the final consistency of the nodes state.

4.2.3. 6top and RPL Objective Function operations

An implementation of a RPL [RFC6550] Objective Function (OF), such as the RPL Objective Function Zero (OF0) [RFC6552] that is used in the Minimal 6TiSCH Configuration [I-D.ietf-6tisch-minimal] to support RPL over a static schedule, may leverage, for its internal computation, the information maintained by 6top.

Most OFs require metrics about reachability, such as the ETX. 6top creates and maintains an abstract neighbor table, and this state may be leveraged to feed an OF and/or store OF information as well. In particular, 6top creates and maintains an abstract neighbor table. A neighbor table entry contains a set of statistics with respect to that specific neighbor including the time when the last packet has been received from that neighbor, a set of cell quality metrics (e.g. RSSI or LQI), the number of packets sent to the neighbor or the number of packets received from it. This information can be obtained through 6top management APIs as detailed in the 6top sublayer

specification [I-D.wang-6tisch-6top-sublayer] and used for instance to compute a Rank Increment that will determine the selection of the preferred parent.

6top provides statistics about the underlying layer so the OF can be tuned to the nature of the TSCH MAC layer. 6top also enables the RPL OF to influence the MAC behaviour, for instance by configuring the periodicity of IEEE std 802.15.4 Extended Beacons (EB's). By augmenting the EB periodicity, it is possible to change the network dynamics so as to improve the support of devices that may change their point of attachment in the 6TiSCH network.

Some RPL control messages, such as the DODAG Information Object (DIO) are ICMPv6 messages that are broadcast to all neighbor nodes. With 6TiSCH, the broadcast channel requirement is addressed by 6top by configuring TSCH to provide a broadcast channel, as opposed to, for instance, piggybacking the DIO messages in Enhance Beacons. Consideration was given towards finding a way to embed the Route Advertisements and the RPL DIO messages (both of which are multicast) into the IEEE std 802.15.4 Enhanced Beacons. It was determined that this produced undue timer coupling among layers, that the resulting packet size was potentially too large, and required it is not yet clear that there is any need for Enhanced Beacons in a production network.

4.2.4. Network Synchronization

Nodes in a TSCH network must be time synchronized. A node keeps synchronized to its time source neighbor through a combination of frame-based and acknowledgment-based synchronization. In order to maximize battery life and network throughput, it is advisable that RPL ICMP discovery and maintenance traffic (governed by the trickle timer) be somehow coordinated with the transmission of time synchronization packets (especially with enhanced beacons). This could be achieved through an interaction of the 6top sublayer and the RPL objective Function, or could be controlled by a management entity.

Time distribution requires a loop-less structure. Nodes taken in a synchronization loop will rapidly desynchronize from the network and become isolated. It is expected that a RPL DAG with a dedicated global Instance is deployed for the purpose of time synchronization. That Instance is referred to as the Time Synchronization Global Instance (TSGI). The TSGI can be operated in either of the 3 modes that are detailed in section 3.1.3 of RPL [RFC6550], "Instances, DODAGs, and DODAG Versions". Multiple uncoordinated DODAGs with independent roots may be used if all the roots share a common time source such as the Global Positioning System (GPS). In the absence

of a common time source, the TSGI should form a single DODAG with a virtual root. A backbone network is then used to synchronize and coordinate RPL operations between the backbone routers that act as sinks for the LLN. Optionally, RPL's periodic operations may be used to transport the network synchronization. This may mean that 6top would need to trigger (override) the trickle timer if no other traffic has occurred for such a time that nodes may get out of synchronization.

A node that has not joined the TSGI advertises a MAC level Join Priority of 0xFF to notify its neighbors that is not capable of serving as time parent. A node that has joined the TSGI advertises a MAC level Join Priority set to its DAGRank() in that Instance, where DAGRank() is the operation specified in section 3.5.1 of [RFC6550], "Rank Comparison".

A root is configured or obtains by some external means the knowledge of the RPLInstanceID for the TSGI. The root advertises its DagRank in the TSGI, that must be less than 0xFF, as its Join Priority (JP) in its IEEE std 802.15.4 Extended Beacons (EB). We'll note that the JP is now specified between 0 and 0x3F leaving 2 bits in the octet unused in the IEEE std 802.15.4e specification. After consultation with IEEE authors, it was asserted that 6TiSCH can make a full use of the octet to carry an integer value up to 0xFF.

A node that reads a Join Priority of less than 0xFF should join the neighbor with the lesser Join Priority and use it as time parent. If the node is configured to serve as time parent, then the node should join the TSGI, obtain a Rank in that Instance and start advertising its own DagRank in the TSGI as its Join Priority in its EBs.

4.2.5. SlotFrames and Priorities

6TiSCH enables in essence the capability to use IPv6 over a MAC layer that enables to schedule some of the transmissions. In order to ensure that the medium is free of contending packets when time arrives for a scheduled transmission, a window of time is defined around the scheduled transmission time where the medium must be free of contending energy.

One simple way to obtain such a window is to format time and frequencies in cells of transmission of equal duration. This is the method that is adopted in IEEE std 802.15.4 TSCH as well as the Long Term Evolution (LTE) of cellular networks.

In order to describe that formatting of time and frequencies, the 6TiSCH architecture defines a global concept that is called a Channel Distribution and Usage (CDU) matrix; a CDU matrix is a matrix of

cells with an height equal to the number of available channels (indexed by ChannelOffsets) and a width (in timeslots) that is the period of the network scheduling operation (indexed by slotOffsets) for that CDU matrix. The size of a cell is a timeslot duration, and values of 10 to 15 milliseconds are typical in 802.15.4 TSCH to accommodate for the transmission of a frame and an ack, including the security validation on the receive side which may take up to a few milliseconds on some device architecture.

A CDU matrix iterates over and over with a pseudo-random rotation from an epoch time. In a given network, there might be multiple CDU matrices that operate with different width, so they have different durations and represent different periodic operations. It is recommended that all CDU matrices in a 6TiSCH domain operate with the same cell duration and are aligned, so as to reduce the chances of interferences from slotted-aloha operations. The knowledge of the CDU matrices is shared between all the nodes and used in particular to define slotFrames.

A slotFrame is a MAC-level abstraction that is common to all nodes and contains a series of timeslots of equal length and precedence. It is characterized by a slotFrame_ID, and a slotFrame_size. A slotFrame aligns to a CDU matrix for its parameters, such as number and duration of timeslots.

Multiple slotFrames can coexist in a node schedule, i.e., a node can have multiple activities scheduled in different slotFrames, based on the precedence of the 6TiSCH topologies. The slotFrames may be aligned to different CDU matrices and thus have different width. There is typically one slotFrame for scheduled traffic that has the highest precedence and one or more slotFrame(s) for RPL traffic. The timeslots in the slotFrame are indexed by the SlotOffset; the first cell is at SlotOffset 0.

When a packet is received from a higher layer for transmission, 6top inserts that packet in the outgoing queue which matches the packet best (Differentiated Services [RFC2474] can therefore be used). At each scheduled transmit slot, 6top looks for the frame in all the outgoing queues that best matches the cells. If a frame is found, it is given to the TSCH MAC for transmission.

4.2.6. Distributing the reservation of cells

6TiSCH expects a high degree of scalability together with a distributed routing functionality based on RPL. To achieve this goal, the spectrum must be allocated in a way that allows for spatial reuse between zones that will not interfere with one another. In a

large and spatially distributed network, a 6TiSCH node is often in a good position to determine usage of spectrum in its vicinity.

Use cases for distributed routing are often associated with a statistical distribution of best-effort traffic with variable needs for bandwidth on each individual link. With 6TiSCH, the abstraction of an IPv6 link is implemented as a pair of bundles of cells, one in each direction; the size of a bundle is optimal when both the energy wasted idle listening and the packet drops due to congestion loss are minimized. This can be maintained if the number of cells in a bundle is adapted dynamically, and with enough reactivity, to match the variations of best-effort traffic. In turn, the agility to fulfill the needs for additional cells improves when the number of interactions with other devices and the protocol latencies are minimized.

6TiSCH limits that interaction to RPL parents that will only negotiate with other RPL parents, and performs that negotiation by groups of cells as opposed to individual cells. The 6TiSCH architecture allows RPL parents to adjust dynamically, and independently from the PCE, the amount of bandwidth that is used to communicate between themselves and their children, in both directions; to that effect, an allocation mechanism enables a RPL parent to obtain the exclusive use of a portion of a CDU matrix within its interference domain. Note that a PCE is expected to have precedence in the allocation, so that a RPL parent would only be able to obtain portions that are not in-use by the PCE.

The 6TiSCH architecture introduces the concept of chunks [I-D.ietf-6tisch-terminology]) to operate such spectrum distribution for a whole group of cells at a time. The CDU matrix is formatted into a set of chunks, each of them identified uniquely by a chunk-ID. The knowledge of this formatting is shared between all the nodes in a 6TiSCH network. 6TiSCH also defines the process of chunk ownership appropriation whereby a RPL parent discovers a chunk that is not used in its interference domain (e.g lack of energy detected in reference cells in that chunk); then claims the chunk, and then defends it in case another RPL parent would attempt to appropriate it while it is in use. The chunk is the basic unit of ownership that is used in that process.

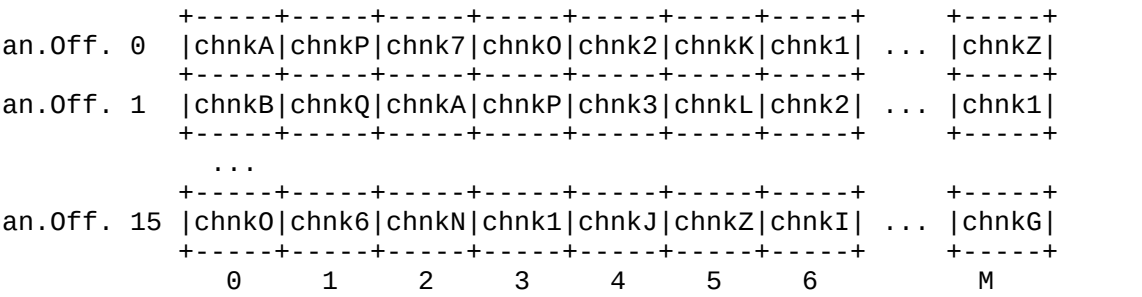


Figure 7: CDU matrix Partitioning in Chunks

As a result of the process of chunk ownership appropriation, the RPL parent has exclusive authority to decide which cell in the appropriated chunk can be used by which node in its interference domain. In other words, it is implicitly delegated the right to manage the portion of the CDU matrix that is represented by the chunk. The RPL parent may thus orchestrate which transmissions occur in any of the cells in the chunk, by allocating cells from the chunk to any form of communication (unicast, multicast) in any direction between itself and its children. Initially, those cells are added to the heap of free cells, then dynamically placed into existing bundles, in new bundles, or allocated opportunistically for one transmission.

The appropriation of a chunk can also be requested explicitly by the PCE to any node. In that case, the node still may need to perform the appropriation process to validate that no other node has claimed that chunk already. After a successful appropriation, the PCE owns the cells in that chunk, and may use them as hard cells to set up Tracks.

4.3. Communication Paradigms and Interaction Models

[I-D.ietf-6tisch-terminology] defines the terms of Communication Paradigms and Interaction Models, which can be placed in parallel to the Information Models and Data Models that are defined in [RFC3444].

A Communication Paradigms would be an abstract view of a protocol exchange, and would come with an Information Model for the information that is being exchanged. In contrast, an Interaction Models would be more refined and could point on standard operation such as a Representational state transfer (REST) "GET" operation and would match a Data Model for the data that is provided over the protocol exchange.

section 2.1.3 of [I-D.ietf-roll-rpl-industrial-applicability] and next sections discuss application-layer paradigms, such as Source-sink (SS) that is a Multipeer to Multipeer (MP2MP) model primarily used for alarms and alerts, Publish-subscribe (PS, or pub/sub) that is typically used for sensor data, as well as Peer-to-peer (P2P) and Peer-to-multipeer (P2MP) communications. Additional considerations on Duocast and its N-cast generalization are also provided. Those paradigms are frequently used in industrial automation, which is a major use case for IEEE std 802.15.4 TSCH wireless networks with [ISA100.11a] and [WirelessHART], that provides a wireless access to [HART] applications and devices.

This specification focuses on Communication Paradigms and Interaction Models for packet forwarding and TSCH resources (cells) management. Management mechanisms for the TSCH schedule at Link-layer (one-hop), Network-layer (multithop along a Track), and Application-layer (remote control) are discussed in Section 4.4. Link-layer frame forwarding interactions are discussed in Section 4.6, and Network-layer Packet routing is addressed in Section 4.7.

4.4. Schedule Management Mechanisms

6TiSCH uses 4 paradigms to manage the TSCH schedule of the LLN nodes: Static Scheduling, neighbor-to-neighbor Scheduling, remote monitoring and scheduling management, and Hop-by-hop scheduling. Multiple mechanisms are defined that implement the associated Interaction Models, and can be combined and used in the same LLN. Which mechanism(s) to use depends on application requirements.

4.4.1. Static Scheduling

In the simplest instantiation of a 6TiSCH network, a common fixed schedule may be shared by all nodes in the network. Cells are shared, and nodes contend for slot access in a slotted aloha manner.

A static TSCH schedule can be used to bootstrap a network, as an initial phase during implementation, or as a fall-back mechanism in case of network malfunction. This schedule is pre-established, for instance decided by a network administrator based on operational needs. It can be pre-configured into the nodes, or, more commonly, learned by a node when joining the network using standard IEEE std 802.15.4 Information Elements (IE). Regardless, the schedule remains unchanged after the node has joined a network. RPL is used on the resulting network. This "minimal" scheduling mechanism that implements this paradigm is detailed in [I-D.ietf-6tisch-minimal].

4.4.2. Neighbor-to-neighbor Scheduling

In the simplest instantiation of a 6TiSCH network described in Section 4.4.1, nodes may expect a packet at any cell in the schedule and will waste energy idle listening. In a more complex instantiation of a 6TiSCH network, a matching portion of the schedule is established between peers to reflect the observed amount of transmissions between those nodes. The aggregation of the cells between a node and a peer forms a bundle that the 6top layer uses to implement the abstraction of a link for IP. The bandwidth on that link is proportional to the number of cells in the bundle.

If the size of a bundle is configured to fit an average amount of bandwidth, peak traffic is dropped. If the size is configured to allow for peak emissions, energy is be wasted idle listening.

The 6top sublayer [I-D.wang-6tisch-6top-sublayer] defines a protocol for neighbor nodes to reserve soft cells to transmit to one another. Because this reservation is done without global knowledge of the schedule of nodes in the LLN, scheduling collisions are possible. 6top defines a monitoring process which continuously Tracks the packet delivery ratio of soft cells. It uses these statistics to trigger the reallocation of a soft cell in the schedule, using a negotiation protocol between the neighbors nodes communicating over that cell.

In the most efficient instantiations of a 6TiSCH network, the size of the bundles that implement the links may be changed dynamically in order to adapt to the need of end-to-end flows routed by RPL. An optional Scheduling Function (SF) such as SF0 [I-D.ietf-6tisch-6top-sf0] is used to monitor bandwidth usage and perform requests for dynamic allocation by the 6top sublayer. The SF component is not part of the 6top sublayer. It may be collocated on the same device or may be partially or fully offloaded to an external system.

Monitoring and relocation is done in the 6top layer. For the upper layer, the connection between two neighbor nodes appears as an number of cells. Depending on traffic requirements, the upper layer can request 6top to add or delete a number of cells scheduled to a particular neighbor, without being responsible for choosing the exact slotOffset/channelOffset of those cells.

4.4.3. Remote Monitoring and Schedule Management

The 6top interface document [I-D.ietf-6tisch-6top-interface] specifies the generic data model that can be used to monitor and manage resources of the 6top sublayer. Abstract methods are

suggested for use by a management entity in the device. The data model also enables remote control operations on the 6top sublayer.

The capability to interact with the node 6top sublayer from multiple hops away can be leveraged for monitoring, scheduling, or a combination of thereof. The architecture supports variations on the deployment model, and focuses on the flows rather than whether there is a proxy or a translation operation en-route.

[I-D.ietf-6tisch-coap] defines an mapping of the 6top set of commands, which is described in [I-D.ietf-6tisch-6top-interface], to CoAP resources. This allows an entity to interact with the 6top layer of a node that is multiple hops away in a RESTful fashion.

The entity issuing the CoAP requests can be a central scheduling entity (e.g. a PCE), a node multiple hops away with the authority to modify the TSCH schedule (e.g. the head of a local cluster), or a external device monitoring the overall state of the network (e.g. NME). It is also possible that a mapping entity on the backbone transforms a non-CoAP protocol such as PCEP into the RESTful interfaces that the 6TiSCH devices support.

With respect to Centralized routing and scheduling, the 6TiSCH Architecture is (expected to be) be an extension of the detnet work Deterministic Networking Architecture [I-D.ietf-detnet-architecture], which studies Layer-3 aspects of Deterministic Networks, and covers networks that span multiple Layer-2 domains. The DetNet architecture is a form of SDN Architecture and is composed of three planes, a (User) Application Plane, a Controller Plane (where the PCE operates), and a Network Plane which in our case is the 6TiSCH LLN. The generic SDN architecture is discussed in Software-Defined Networking (SDN): Layers and Architecture Terminology [RFC7426] and is represented below:

SDN Layers and Architecture Terminology per RFC 7426

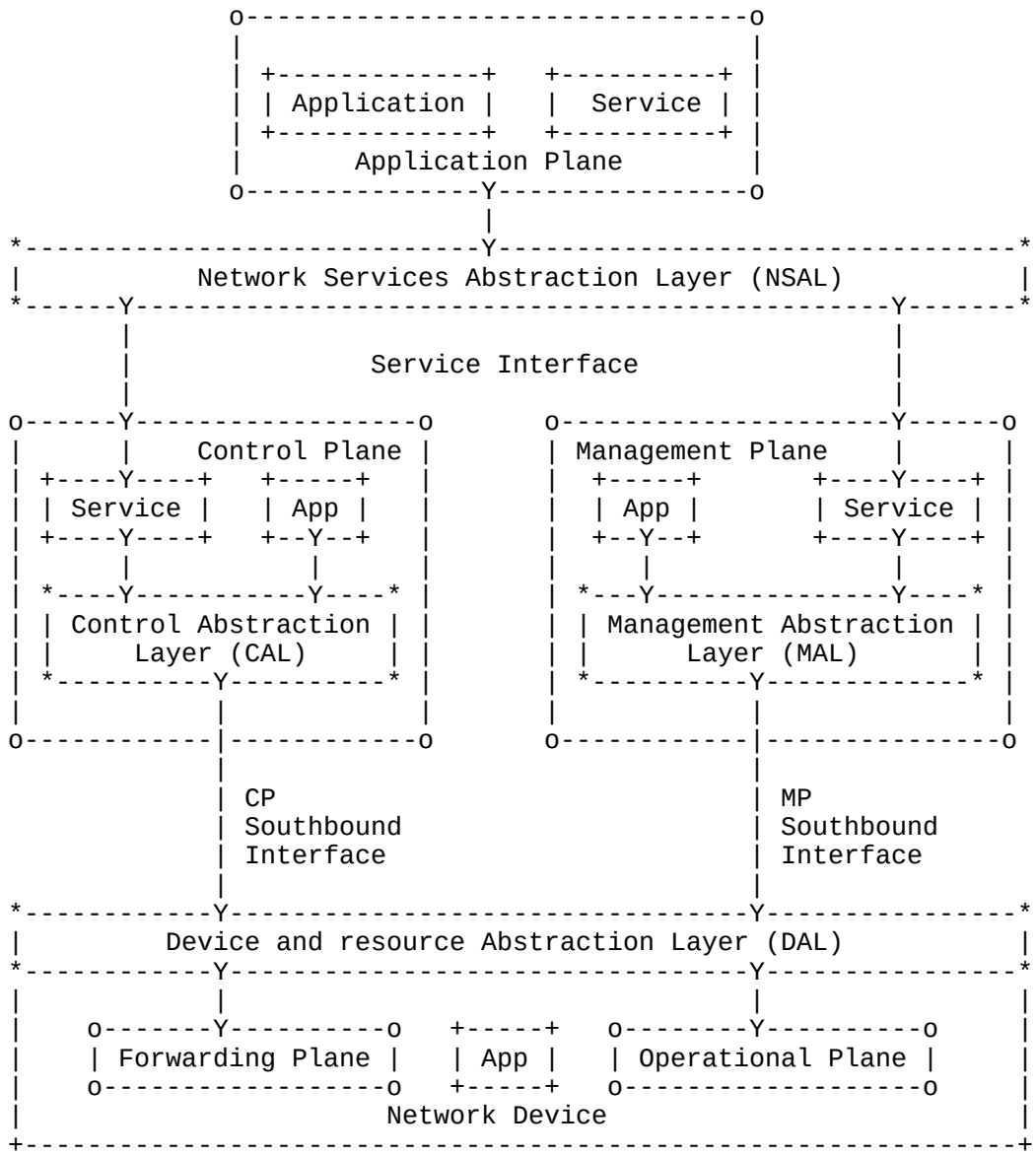


Figure 8

The PCE establishes end-to-end Tracks of hard cells, which are described in more details in Section 4.6.1. The DetNet work is expected to enable end to end Deterministic Path across heterogeneous

network (e.g. a 6TiSCH LLN and an Ethernet Backbone). This model fits the 6TiSCH extended configuration, whereby a 6BBR federates multiple 6TiSCH LLN in a single subnet over a backbone that can be, for instance, Ethernet or Wi-Fi. In that model, 6TiSCH 6BBRs synchronize with one another over the backbone, so as to ensure that the multiple LLNs that form the IPv6 subnet stay tightly synchronized.

If the Backbone is Deterministic, then the Backbone Router ensures that the end-to-end deterministic behavior is maintained between the LLN and the backbone. It is the responsibility of the PCE to compute a deterministic path and to end across the TSCH network and an IEEE std 802.1 TSN Ethernet backbone, and that of DetNet to enable end-to-end deterministic forwarding.

4.4.4. Hop-by-hop Scheduling

A node can reserve a Track (Section 4.5) to a destination node multiple hops away by installing soft cells at each intermediate node. This forms a Track of soft cells. It is the responsibility of the 6top sublayer of each node on the Track to monitor these soft cells and trigger relocation when needed.

This hop-by-hop reservation mechanism is expected to be similar in essence to [RFC3209] and/or [RFC4080]/[RFC5974]. The protocol for a node to trigger hop-by-hop scheduling is not yet defined.

4.5. On Tracks

4.5.1. General Behavior of Tracks

The architecture introduces the concept of a Track, which is a directed path from a source 6TiSCH node to a destination 6TiSCH node across a 6TiSCH LLN. A Track is the 6TiSCH instantiation of the concept of a Deterministic Path as described in [I-D.ietf-detnet-architecture]. Constrained resources such as memory buffers are reserved for that Track in intermediate 6TiSCH nodes to avoid loss related to limited capacity. A 6TiSCH node along a Track not only knows which bundles of cells it should use to receive packets from a previous hop, but also knows which bundle(s) it should use to send packets to its next hop along the Track.

A Track is composed of bundles of cells with related schedules and logical relationships and that ensure that a packet that is injected in a Track will progress in due time all the way to destination. Multiple cells may be scheduled in a Track for the transmission of a single packet, in which case the normal operation of IEEE std 802.15.4 Automatic Repeat-reQuest (ARQ) can take place; the

acknowledgment may be omitted in some cases, for instance if there is no scheduled cell for a possible retry.

There are several benefits for using a Track to forward a packet from a source node to the destination node.

1. Track forwarding, as further described in Section 4.6.1, is a Layer-2 forwarding scheme, which introduces less process delay and overhead than Layer-3 forwarding scheme. Therefore, LLN Devices can save more energy and resource, which is critical for resource constrained devices.

2. Since channel resources, i.e. bundles of cells, have been reserved for communications between 6TiSCH nodes of each hop on the Track, the throughput and the maximum latency of the traffic along a Track are guaranteed and the jitter is maintained small.

3. By knowing the scheduled time slots of incoming bundle(s) and outgoing bundle(s), 6TiSCH nodes on a Track could save more energy by staying in sleep state during in-active slots.

4. Tracks are protected from interfering with one another if a cell belongs to at most one Track, and congestion loss is avoided if at most one packet can be presented to the MAC to use that cell. Tracks enhance the reliability of transmissions and thus further improve the energy consumption in LLN Devices by reducing the chances of retransmission.

4.5.2. Serial Track

A Serial (or simple) Track is the 6TiSCH version of a circuit; a bundle of cells that are programmed to receive (RX-cells) is uniquely paired to a bundle of cells that are set to transmit (TX-cells), representing a Layer-2 forwarding state which can be used regardless of the network layer protocol.

A Serial Track is thus formed end-to-end as a succession of paired bundles, a receive bundle from the previous hop and a transmit bundle to the next hop along the Track. For a given iteration of the device schedule, the effective channel of the cell is obtained by adding a pseudo-random number to the channelOffset of the cell, which results in a rotation of the frequency that used for transmission.

The bundles may be computed so as to accommodate both variable rates and retransmissions, so they might not be fully used at a given iteration of the schedule.

4.5.3. Complex Track with Replication and Elimination

As opposed to a Serial Track that is a sequence of nodes and links, a Complex Track is shaped as a directed acyclic graph towards a destination to support multi-path forwarding and route around failures.

A Complex Track may also branch off and rejoin, for the purpose of the DetNet Packet Replication and Elimination (PRE), over non congruent branches. PRE may be used to complement Layer-2 ARQ to meet industrial expectations in Packet Delivery Ratio (PDR), in particular when the Track extends beyond the 6TiSCH network in a larger DetNet network.

The art of Deterministic Networks already include PRE techniques. Example standards include the Parallel Redundancy Protocol (PRP) and the High-availability Seamless Redundancy (HSR) [IEC62439].

At each 6TiSCH hop along the Track, the PCE may schedule more than one timeslot for a packet, so as to support Layer-2 retries (ARQ). It is also possible that the field device only uses the second branch if sending over the first branch fails.

In the art of TSCH, a path does not necessarily support PRE but it is almost systematically multi-path. This means that a Track is scheduled so as to ensure that each hop has at least two forwarding solutions, and the forwarding decision is to try the preferred one and use the other in case of Layer-2 transmission failure as detected by ARQ.

4.5.4. DetNet End-to-end Path

Ultimately, DetNet should enable to extend a Track beyond the 6TiSCH LLN. Figure 9 illustrates a Track that is laid out from a field device in a 6TiSCH network to an IoT gateway that is located on an 802.1 Time-Sensitive Networking (TSN) backbone.

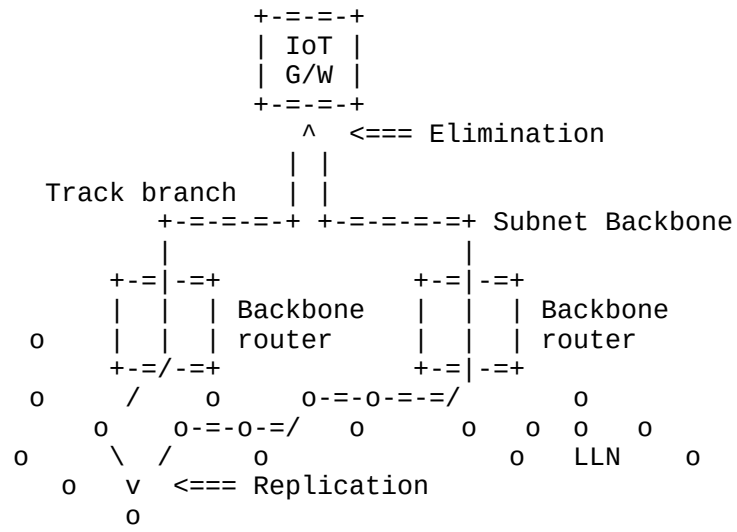


Figure 9: End-to-End deterministic Track

The Replication function in the 6TiSCH Node sends a copy of each packet over two different branches, and the PCE schedules each hop of both branches so that the two copies arrive in due time at the gateway. In case of a loss on one branch, hopefully the other copy of the packet still makes it in due time. If two copies make it to the IoT gateway, the Elimination function in the gateway ignores the extra packet and presents only one copy to upper layers.

4.5.5. Cell Reuse

The 6TiSCH architecture provides means to avoid waste of cells as well as overflows in the transmit bundle pof a Track, as follows:

In one hand, a TX-cell that is not needed for the current iteration may be reused opportunistically on a per-hop basis for routed packets. When all of the frame that were received for a given Track are effectively transmitted, any available TX-cell for that Track can be reused for upper layer traffic for which the next-hop router matches the next hop along the Track. In that case, the cell that is being used is effectively a TX-cell from the Track, but the short address for the destination is that of the next-hop router. It results that a frame that is received in a RX-cell of a Track with a destination MAC address set to this node as opposed to broadcast must be extracted from the Track and delivered to the upper layer (a frame with an unrecognized

destination MAC address is dropped at the lower MAC layer and thus is not received at the 6top sublayer).

On the other hand, it might happen that there are not enough TX-cells in the transmit bundle to accommodate the Track traffic, for instance if more retransmissions are needed than provisioned. In that case, the frame can be placed for transmission in the bundle that is used for Layer-3 traffic towards the next hop along the Track as long as it can be routed by the upper layer, that is, typically, if the frame transports an IPv6 packet. The MAC address should be set to the next-hop MAC address to avoid confusion. It results that a frame that is received over a Layer-3 bundle may be in fact associated to a Track. In a classical IP link such as an Ethernet, off-Track traffic is typically in excess over reservation to be routed along the non-reserved path based on its QoS setting. But with 6TiSCH, since the use of the Layer-3 bundle may be due to transmission failures, it makes sense for the receiver to recognize a frame that should be re-Tracked, and to place it back on the appropriate bundle if possible. A frame should be re-Tracked if the Per-Hop-Behavior group indicated in the Differentiated Services Field of the IPv6 header is set to Deterministic Forwarding, as discussed in Section 4.7.1. A frame is re-Tracked by scheduling it for transmission over the transmit bundle associated to the Track, with the destination MAC address set to broadcast.

4.6. Forwarding Models

By forwarding, this specification means the per-packet operation that allows to deliver a packet to a next hop or an upper layer in this node. Forwarding is based on pre-existing state that was installed as a result of a routing computation Section 4.7. 6TiSCH supports three different forwarding model, G-MPLS Track Forwarding (TF), 6LoWPAN Fragment Forwarding (FF) and IPv6 Forwarding (6F).

4.6.1. Track Forwarding

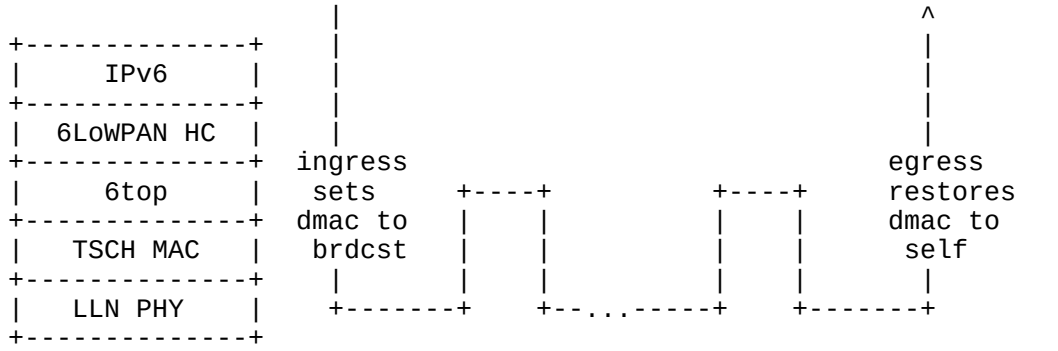
Forwarding along a Track can be seen as a Generalized Multi-protocol Label Switching (G-MPLS) operation in that the information used to switch a frame is not an explicit label, but rather related to other properties of the way the packet was received, a particular cell in the case of 6TiSCH. As a result, as long as the TSCH MAC (and Layer-2 security) accepts a frame, that frame can be switched regardless of the protocol, whether this is an IPv6 packet, a 6LoWPAN fragment, or a frame from an alternate protocol such as WirelessHART or ISA100.11a.

A data frame that is forwarded along a Track normally has a destination MAC address that is set to broadcast - or a multicast address depending on MAC support. This way, the MAC layer in the intermediate nodes accepts the incoming frame and 6top switches it without incurring a change in the MAC header. In the case of IEEE std 802.15.4, this means effectively broadcast, so that along the Track the short address for the destination of the frame is set to 0xFFFF.

There are 2 modes for a Track, transport mode and tunnel mode.

4.6.1.1. Transport Mode

In transport mode, the Protocol Data Unit (PDU) is associated with flow-dependant meta-data that refers uniquely to the Track, so the 6top sublayer can place the frame in the appropriate cell without ambiguity. In the case of IPv6 traffic, this flow identification is transported in the Flow Label of the IPv6 header. Associated with the source IPv6 address, the Flow Label forms a globally unique identifier for that particular Track that is validated at egress before restoring the destination MAC address (DMAC) and punting to the upper layer.



Track Forwarding, Transport Mode

4.6.1.2. Tunnel Mode

In tunnel mode, the frames originate from an arbitrary protocol over a compatible MAC that may or may not be synchronized with the 6TiSCH network. An example of this would be a router with a dual radio that is capable of receiving and sending WirelessHART or ISA100.11a frames with the second radio, by presenting itself as an access Point or a Backbone Router, respectively.

In that mode, some entity (e.g. PCE) can coordinate with a WirelessHART Network Manager or an ISA100.11a System Manager to specify the flows that are to be transported transparently over the Track.

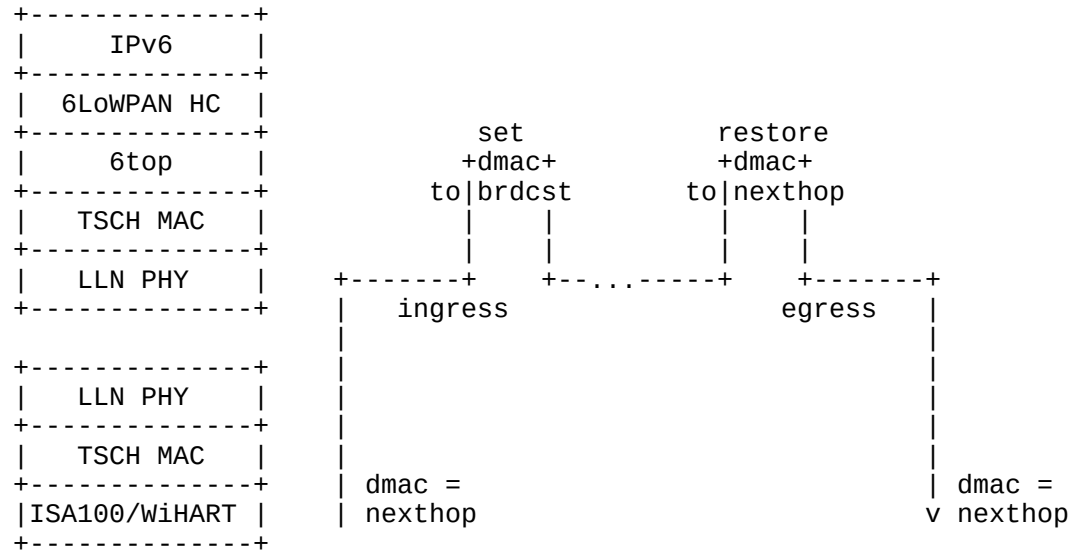


Figure 10: Track Forwarding, Tunnel Mode

In that case, the flow information that identifies the Track at the ingress 6TiSCH router is derived from the RX-cell. The dmac is set to this node but the flow information indicates that the frame must be tunneled over a particular Track so the frame is not passed to the upper layer. Instead, the dmac is forced to broadcast and the frame is passed to the 6top sublayer for switching.

At the egress 6TiSCH router, the reverse operation occurs. Based on metadata associated to the Track, the frame is passed to the appropriate link layer with the destination MAC restored.

4.6.1.3. Tunnel Metadata

Metadata coming with the Track configuration is expected to provide the destination MAC address of the egress endpoint as well as the tunnel mode and specific data depending on the mode, for instance a service access point for frame delivery at egress. If the tunnel egress point does not have a MAC address that matches the configuration, the Track installation fails.

In transport mode, if the final Layer-3 destination is the tunnel termination, then it is possible that the IPv6 address of the destination is compressed at the 6LoWPAN sublayer based on the MAC address. It is thus mandatory at the ingress point to validate that the MAC address that was used at the 6LoWPAN sublayer for compression matches that of the tunnel egress point. For that reason, the node that injects a packet on a Track checks that the destination is effectively that of the tunnel egress point before it overwrites it to broadcast. The 6top sublayer at the tunnel egress point reverts that operation to the MAC address obtained from the tunnel metadata.

4.6.2. Fragment Forwarding

Considering that 6LoWPAN packets can be as large as 1280 bytes (the IPv6 MTU), and that the non-storing mode of RPL implies Source Routing that requires space for routing headers, and that a IEEE std 802.15.4 frame with security may carry in the order of 80 bytes of effective payload, an IPv6 packet might be fragmented into more than 16 fragments at the 6LoWPAN sublayer.

This level of fragmentation is much higher than that traditionally experienced over the Internet with IPv4 fragments, where fragmentation is already known as harmful.

In the case to a multihop route within a 6TiSCH network, Hop-by-Hop recomposition occurs at each hop in order to reform the packet and route it. This creates additional latency and forces intermediate nodes to store a portion of a packet for an undetermined time, thus impacting critical resources such as memory and battery.

[I-D.thubert-6lo-forwarding-fragments] describes a mechanism whereby the datagram tag in the 6LoWPAN Fragment is used as a label for switching at the 6LoWPAN sublayer. The draft allows for a degree of flow control based on an Explicit Congestion Notification, as well as end-to-end individual fragment recovery.

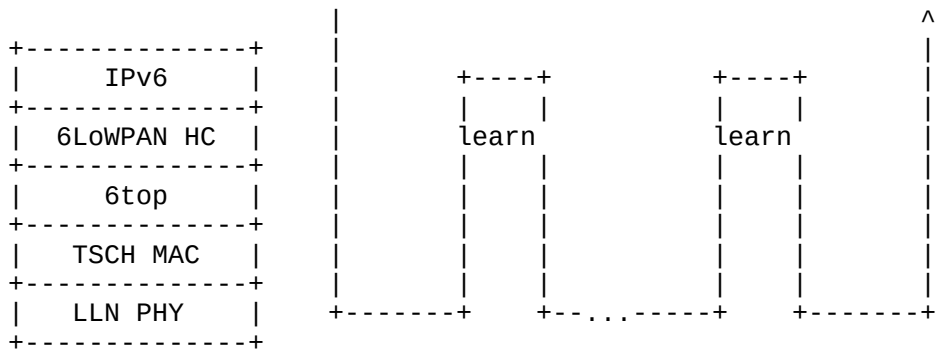


Figure 11: Forwarding First Fragment

In that model, the first fragment is routed based on the IPv6 header that is present in that fragment. The 6LoWPAN sublayer learns the next hop selection, generates a new datagram tag for transmission to the next hop, and stores that information indexed by the incoming MAC address and datagram tag. The next fragments are then switched based on that stored state.

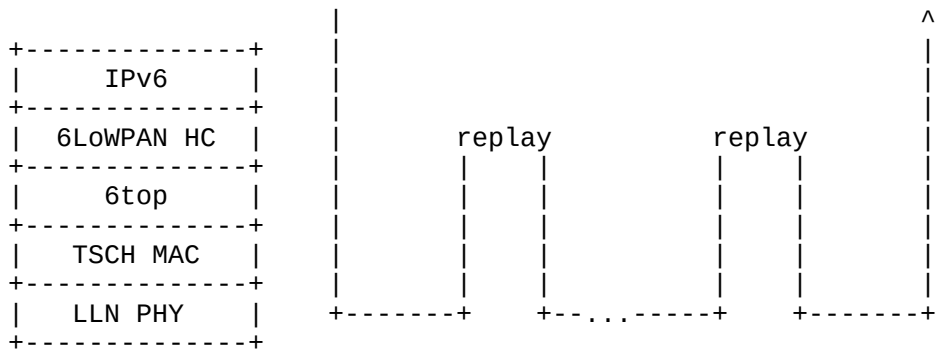


Figure 12: Forwarding Next Fragment

A bitmap and an ECN echo in the end-to-end acknowledgment enable the source to resend the missing fragments selectively. The first fragment may be resent to carve a new path in case of a path failure. The ECN echo set indicates that the number of outstanding fragments should be reduced.

4.6.3. IPv6 Forwarding

As the packets are routed at Layer-3, traditional QoS and RED operations are expected to prioritize flows; the application of

Differentiated Services is further discussed in [I-D.svshah-tsvwg-lln-diffserv-recommendations].

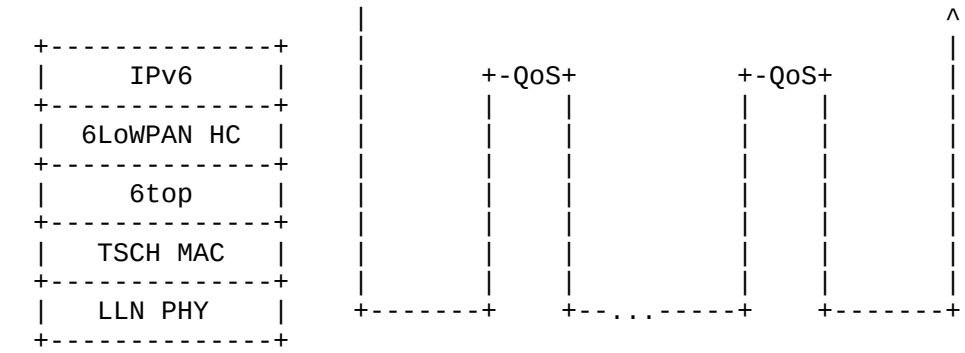


Figure 13: IP Forwarding

4.7. Centralized vs. Distributed Routing

6TiSCH supports a mixed model of centralized routes and distributed routes. Centralized routes can for example be computed by a entity such as a PCE. Distributed routes are computed by RPL.

Both methods may inject routes in the Routing Tables of the 6TiSCH routers. In either case, each route is associated with a 6TiSCH topology that can be a RPL Instance topology or a Track. The 6TiSCH topology is indexed by a Instance ID, in a format that reuses the RPLInstanceID as defined in RPL [RFC6550].

Both RPL and PCE rely on shared sources such as policies to define Global and Local RPLInstanceIDs that can be used by either method. It is possible for centralized and distributed routing to share a same topology. Generally they will operate in different slotFrames, and centralized routes will be used for scheduled traffic and will have precedence over distributed routes in case of conflict between the slotFrames.

4.7.1. Packet Marking and Handling

All packets inside a 6TiSCH domain must carry the Instance ID that identifies the 6TiSCH topology that is to be used for routing and forwarding that packet. The location of that information must be the same for all packets forwarded inside the domain.

For packets that are routed by a PCE along a Track, the tuple formed by the IPv6 source address and a local RPLInstanceID in the packet identify uniquely the Track and associated transmit bundle.

For packets that are routed by RPL, that information is the RPLInstanceID which is carried in the RPL Packet Information, as discussed in section 11.2 of [RFC6550], "Loop Avoidance and Detection".

The RPL Packet Information (RPI) is carried in IPv6 packets as a RPL option in the IPv6 Hop-By-Hop Header [RFC6553].

A compression mechanism for the RPL packet artifacts that integrates the compression of IP-in-IP encapsulation and the Routing Header type 3 [RFC6554] with that of the RPI in a 6LoWPAN dispatch/header type is specified in [RFC8025] and [I-D.ietf-roll-routing-dispatch].

Either way, the method and format used for encoding the RPLInstanceID is generalized to all 6TiSCH topological Instances, which include both RPL Instances and Tracks.

4.7.2. Replication, Retries and Elimination

6TiSCH expects elimination and replication of packets along a complex Track, but has no position about how the sequence numbers would be tagged in the packet.

As it goes, 6TiSCH expects that timeSlots corresponding to copies of a same packet along a Track are correlated by configuration, and does not need to process the sequence numbers.

The semantics of the configuration will enable correlated timeSlots to be grouped for transmit (and respectively receive) with a 'OR' relations, and then a 'AND' relation would be configurable between groups. The semantics is that if the transmit (and respectively receive) operation succeeded in one timeSlot in a 'OR' group, then all the other timeSLots in the group are ignored. Now, if there are at least two groups, the 'AND' relation between the groups indicates that one operation must succeed in each of the groups.

On the transmit side, timeSlots provisioned for retries along a same branch of a Track are placed a same 'OR' group. The 'OR' relation indicates that if a transmission is acknowledged, then further transmissions should not be attempted for timeSlots in that group. There are as many 'OR' groups as there are branches of the Track departing from this node. Different 'OR' groups are programmed for the purpose of replication, each group corresponding to one branch of the Track. The 'AND' relation between the groups indicates that transmission over any of branches must be attempted regardless of whether a transmission succeeded in another branch. It is also possible to place cells to different next-hop routers in a same 'OR'

group. This allows to route along multi-path tracks, trying one next-hop and then another only if sending to the first fails.

On the receive side, all timeSlots are programmed in a same 'OR' group. Retries of a same copy as well as converging branches for elimination are converged, meaning that the first successful reception is enough and that all the other timeSlots can be ignored.

4.7.3. Differentiated Services Per-Hop-Behavior

Additionally, an IP packet that is sent along a Track uses the Differentiated Services Per-Hop-Behavior Group called Deterministic Forwarding, as described in [I-D.svshah-tsvwg-deterministic-forwarding].

5. IANA Considerations

This specification does not require IANA action.

6. Security Considerations

This architecture operates on IEEE std 802.15.4 and expects link-layer security to be enabled at all times between connected devices, except for the very first step of the device join process, where a joining device may need some initial, unsecured exchanges so as to obtain its initial key material. Work has already started at the 6TiSCH Security Design Team and an overview of the current state of that work is presented in Section 6.1.

Future work on 6TiSCH security and will examine in deeper detail how to secure transactions end-to-end, and to maintain the security posture of a device over its lifetime. The result of that work will be described in a subsequent volume of this architecture.

6.1. Join Process Highlights

The architecture specifies three logical elements to describe the join process:

Joining Node (JN): Node that wishes to become part of the network;

Join Coordination Entity (JCE) : A Join Coordination Entity (JCE) that arbitrates network access and hands out network parameters (such as keying material);

Join Assistant (JA), a one-hop (radio) neighbor of the joining node that acts as proxy network node and may provide connectivity with the JCE.

The join protocol consists of three major activities:

Device Authentication: The JN and the JA mutually authenticate each other and establish a shared key, so as to ensure on-going authenticated communications. This may involve a server as a third party.

Authorization: The JA decides on whether/how to authorize a JN (if denied, this may result in loss of bandwidth). Conversely, the JN decides on whether/how to authorize the network (if denied, it will not join the network). Authorization decisions may involve other nodes in the network.

Configuration/Parameterization: The JA distributes configuration information to the JN, such as scheduling information, IP address assignment information, and network policies. This may originate from other network devices, for which the JA may act as proxy. This step may also include distribution of information from the JN to the JA and other nodes in the network and, more generally, synchronization of information between these entities.

The device joining process is depicted in Figure 14, where it is assumed that devices have access to certificates and where entities have access to the root CA keys of their communicating parties (initial set-up requirement). Under these assumptions, the authentication step of the device joining process does not require online involvement of a third party. Mutual authentication is performed between the JN and the JA using their certificates, which also results in a shared key between these two entities.

The JA assists the JN in mutual authentication with a remote server node (primarily via provision of a communication path with the server), which also results in a shared (end-to-end) key between those two entities. The server node may be a JCE that arbitrages the network authorization of the JN (where the JA will deny bandwidth if authorization is not successful); it may distribute network-specific configuration parameters (including network-wide keys) to the JN. In its turn, the JN may distribute and synchronize information (including, e.g., network statistics) to the server node and, if so desired, also to the JA. The actual decision of the JN to become part of the network may depend on authorization of the network itself.

The server functionality is a role which may be implemented with one (centralized) or multiple devices (distributed). In either case, mutual authentication is established with each physical server entity with which a role is implemented.

Note that in the above description, the JA does not solely act as a relay node, thereby allowing it to first filter traffic to be relayed based on cryptographic authentication criteria - this provides first-level access control and mitigates certain types of denial-of-service attacks on the network at large.

Depending on more detailed insight in cost/benefit trade-offs, this process might be complemented by a more "relaxed" mechanism, where the JA acts as a relay node only. The final architecture will provide mechanisms to also cover cases where the initial set-up requirements are not met or where some other out-of-sync behavior occurs; it will also suggest some optimizations in case JCE-related information is already available with the JA (via caching of information).

When a device rejoins the network in the same authorization domain, the authorization step could be omitted if the server distributes the authorization state for the device to the JA when the device initially joined the network. However, this generally still requires the exchange of updated configuration information, e.g., related to time schedules and bandwidth allocation.

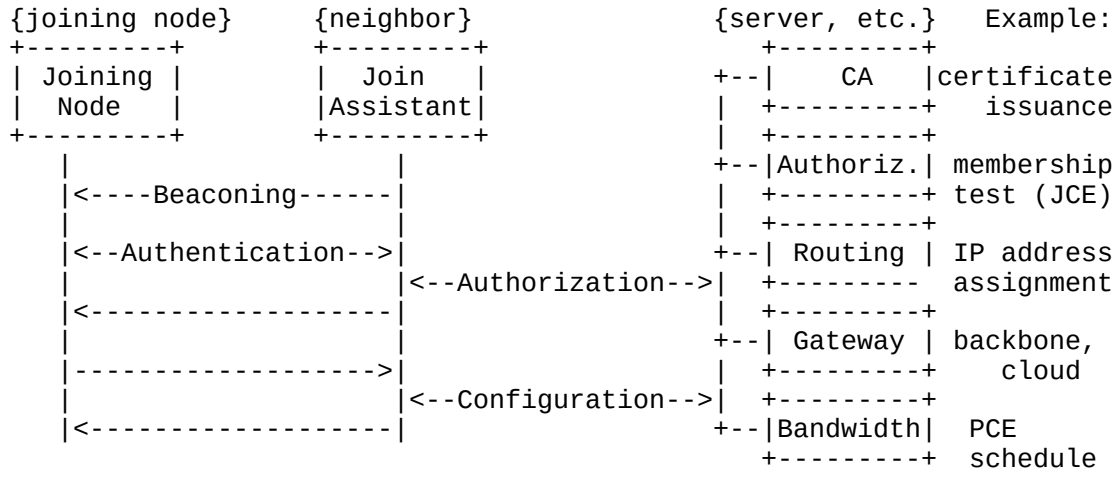


Figure 14: Network joining, with only authorization by third party

7. Acknowledgments

7.1. Contributors

The co-authors of this document are listed below:

Robert Assimiti for his breakthrough work on RPL over TSCH and initial text and guidance.

Kris Pister for creating it all and his continuing guidance through the elaboration of this design.

Michael Richardson for his leadership role in the Security Design Team and his contribution throughout this document.

Rene Struik for the security section and his contribution to the Security Design Team.

Xavier Vilajosana who lead the design of the minimal support with RPL and contributed deeply to the 6top design and the G-MPLS operation of Track switching.

Qin Wang who lead the design of the 6top sublayer and contributed related text that was moved and/or adapted in this document.

Thomas Watteyne for his contribution to the whole design, in particular on TSCH and security.

7.2. Special Thanks

Special thanks to Tero Kivinen, Jonathan Simon, Giuseppe Piro, Subir Das and Yoshihiro Ohba for their deep contribution to the initial security work, and to Diego Dujovne for starting and leading the SF0 effort.

Special thanks also to Pat Kinney for his support in maintaining the connection active and the design in line with work happening at IEEE std 802.15.4.

Special thanks to Ted Lemon who was the INT Area A-D while this specification was developed for his great support and help throughout.

Also special thanks to Ralph Droms who performed the first INT Area Directorate review, that was very deep and through and radically changed the orientations of this document.

7.3. And Do not Forget

This specification is the result of multiple interactions, in particular during the 6TiSCH (bi)Weekly Interim call, relayed through the 6TiSCH mailing list at the IETF.

The authors wish to thank: Alaeddine Weslati, Chonggang Wang, Georgios Exarchakos, Zhuo Chen, Alfredo Grieco, Bert Greevenbosch, Cedric Adjih, Deji Chen, Martin Turon, Dominique Barthel, Elvis Vogli, Geraldine Texier, Malisa Vucinic, Guillaume Gaillard, Herman Storey, Kazushi Muraoka, Ken Bannister, Kuor Hsin Chang, Laurent Toutain, Maik Seewald, Maria Rita Palattella, Michael Behringer, Nancy Cam Winget, Nicola Accettura, Nicolas Montavont, Oleg Hahm, Patrick Wetterwald, Paul Duffy, Peter van der Stock, Rahul Sen, Pieter de Mil, Pouria Zand, Rouhollah Nabati, Rafa Marin-Lopez, Raghuram Sudhaakar, Sedat Gormus, Shitanshu Shah, Steve Simlo, Tengfei Chang, Tina Tsou, Tom Phinney, Xavier Lagrange, Ines Robles and Samita Chakrabarti for their participation and various contributions.

8. References

8.1. Normative References

[I-D.ietf-6lo-backbone-router]
Thubert, P., "IPv6 Backbone Router", draft-ietf-6lo-backbone-router-03 (work in progress), January 2017.

[I-D.ietf-6tisch-minimal]
Vilajosana, X., Pister, K., and T. Watteyne, "Minimal 6TiSCH Configuration", draft-ietf-6tisch-minimal-19 (work in progress), January 2017.

[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-08 (work in progress), December 2016.

[I-D.ietf-detnet-architecture]
Finn, N. and P. Thubert, "Deterministic Networking Architecture", draft-ietf-detnet-architecture-00 (work in progress), September 2016.

[I-D.ietf-roll-routing-dispatch]
Thubert, P., Bormann, C., Toutain, L., and R. Cragie, "6LoWPAN Routing Header", draft-ietf-roll-routing-dispatch-05 (work in progress), October 2016.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <http://www.rfc-editor.org/info/rfc768>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <http://www.rfc-editor.org/info/rfc2460>.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <http://www.rfc-editor.org/info/rfc4861>.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <http://www.rfc-editor.org/info/rfc4862>.

[RFC6282] Hui, J., Ed. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, DOI 10.17487/RFC6282, September 2011, <http://www.rfc-editor.org/info/rfc6282>.

[RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, DOI 10.17487/RFC6550, March 2012, <http://www.rfc-editor.org/info/rfc6550>.

[RFC6551] Vasseur, JP., Ed., Kim, M., Ed., Pister, K., Dejean, N., and D. Barthel, "Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks", RFC 6551, DOI 10.17487/RFC6551, March 2012, <http://www.rfc-editor.org/info/rfc6551>.

[RFC6552] Thubert, P., Ed., "Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6552, DOI 10.17487/RFC6552, March 2012, <http://www.rfc-editor.org/info/rfc6552>.

[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>.

[RFC6554] Hui, J., Vasseur, JP., Culler, D., and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6554, DOI 10.17487/RFC6554, March 2012, <http://www.rfc-editor.org/info/rfc6554>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <http://www.rfc-editor.org/info/rfc6775>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <http://www.rfc-editor.org/info/rfc7252>.

[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>.

[RFC8025] Thubert, P., Ed. and R. Cragie, "IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Paging Dispatch", RFC 8025, DOI 10.17487/RFC8025, November 2016, <http://www.rfc-editor.org/info/rfc8025>.

8.2. Informative References

[I-D.ietf-6lo-ap-nd]
Sarikaya, B., Thubert, P., and M. Sethi, "Address Protected Neighbor Discovery for Low-power and Lossy Networks", draft-ietf-6lo-ap-nd-00 (work in progress), November 2016.

[I-D.ietf-6tisch-6top-interface]
Wang, Q. and X. Vilajosana, "6TiSCH Operation Sublayer (6top) Interface", draft-ietf-6tisch-6top-interface-04 (work in progress), July 2015.

[I-D.ietf-6tisch-6top-protocol]
Wang, Q. and X. Vilajosana, "6top Protocol (6P)", draft-ietf-6tisch-6top-protocol-03 (work in progress), October 2016.

[I-D.ietf-6tisch-6top-sf0]
Dujovne, D., Grieco, L., Palattella, M., and N. Accettura, "6TiSCH 6top Scheduling Function Zero (SF0)", draft-ietf-6tisch-6top-sf0-02 (work in progress), October 2016.

[I-D.ietf-6tisch-coap]
Sudhaakar, R. and P. Zand, "6TiSCH Resource Management and Interaction using CoAP", draft-ietf-6tisch-coap-03 (work in progress), March 2015.

[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., Schmitt, J., Vilajosana, X., Mahmoodi, T., Spirou, S., and P. Vizarreta, "Deterministic Networking Use Cases", draft-ietf-detnet-use-cases-11 (work in progress), October 2016.

[I-D.ietf-manet-aodvv2]
Perkins, C., Ratliff, S., Dowdell, J., Steenbrink, L., and V. Mercieca, "Ad Hoc On-demand Distance Vector Version 2 (AODVv2) Routing", draft-ietf-manet-aodvv2-16 (work in progress), May 2016.

[I-D.ietf-roll-rpl-industrial-applicability]
Phinney, T., Thubert, P., and R. Assimiti, "RPL applicability in industrial networks", draft-ietf-roll-rpl-industrial-applicability-02 (work in progress), October 2013.

[I-D.richardson-6tisch-security-architecture]
Richardson, M., "security architecture for 6top: requirements and structure", draft-richardson-6tisch-security-architecture-02 (work in progress), April 2014.

[I-D.struik-6tisch-security-architecture-elements]
Struik, R., Ohba, Y., and S. Das, "6TiSCH Security Architectural Elements, Desired Protocol Properties, and Framework", draft-struik-6tisch-security-architecture-elements-01 (work in progress), October 2014.

[I-D.svshah-tsvwg-deterministic-forwarding]
Shah, S. and P. Thubert, "Deterministic Forwarding PHB", draft-svshah-tsvwg-deterministic-forwarding-04 (work in progress), August 2015.

[I-D.svshah-tsvwg-lln-diffserv-recommendations]
Shah, S. and P. Thubert, "Differentiated Service Class Recommendations for LLN Traffic", draft-svshah-tsvwg-lln-diffserv-recommendations-04 (work in progress), February 2015.

[I-D.thubert-6lo-bier-dispatch]
Thubert, P., Brodard, Z., Jiang, H., and G. Texier, "A 6loRH for BitStrings", draft-thubert-6lo-bier-dispatch-02 (work in progress), January 2017.

[I-D.thubert-6lo-forwarding-fragments]
Thubert, P. and J. Hui, "LLN Fragment Forwarding and Recovery", draft-thubert-6lo-forwarding-fragments-04 (work in progress), January 2017.

[I-D.thubert-6lo-rfc6775-update]
Thubert, P., Nordmark, E., and S. Chakrabarti, "An Update to 6LoWPAN ND", draft-thubert-6lo-rfc6775-update-01 (work in progress), October 2016.

[I-D.thubert-bier-replication-elimination]
Thubert, P., Brodard, Z., and H. Jiang, "BIER-TE-based OAM, Replication and Elimination", draft-thubert-bier-replication-elimination-00 (work in progress), September 2016.

[I-D.vanderstok-core-comi]
Stok, P., Bierman, A., Veillette, M., and A. Pelov, "CoAP Management Interface", draft-vanderstok-core-comi-11 (work in progress), January 2017.

[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.

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, DOI 10.17487/RFC2474, December 1998, <http://www.rfc-editor.org/info/rfc2474>.

[RFC2545] Marques, P. and F. Dupont, "Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing", RFC 2545, DOI 10.17487/RFC2545, March 1999, <http://www.rfc-editor.org/info/rfc2545>.

[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>.

[RFC3444] Pras, A. and J. Schoenwaelder, "On the Difference between Information Models and Data Models", RFC 3444, DOI 10.17487/RFC3444, January 2003, <http://www.rfc-editor.org/info/rfc3444>.

[RFC3610] Whiting, D., Housley, R., and N. Ferguson, "Counter with CBC-MAC (CCM)", RFC 3610, DOI 10.17487/RFC3610, September 2003, <http://www.rfc-editor.org/info/rfc3610>.

[RFC3963] Devarapalli, V., Wakikawa, R., Petrescu, A., and P. Thubert, "Network Mobility (NEMO) Basic Support Protocol", RFC 3963, DOI 10.17487/RFC3963, January 2005, <http://www.rfc-editor.org/info/rfc3963>.

[RFC3971] Arkko, J., Ed., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, DOI 10.17487/RFC3971, March 2005, <http://www.rfc-editor.org/info/rfc3971>.

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, DOI 10.17487/RFC3972, March 2005, <http://www.rfc-editor.org/info/rfc3972>.

[RFC4080] Hancock, R., Karagiannis, G., Loughney, J., and S. Van den Bosch, "Next Steps in Signaling (NSIS): Framework", RFC 4080, DOI 10.17487/RFC4080, June 2005, <http://www.rfc-editor.org/info/rfc4080>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <http://www.rfc-editor.org/info/rfc4291>.

[RFC4389] Thaler, D., Talwar, M., and C. Patel, "Neighbor Discovery Proxies (ND Proxy)", RFC 4389, DOI 10.17487/RFC4389, April 2006, <http://www.rfc-editor.org/info/rfc4389>.

[RFC4429] Moore, N., "Optimistic Duplicate Address Detection (DAD) for IPv6", RFC 4429, DOI 10.17487/RFC4429, April 2006, <http://www.rfc-editor.org/info/rfc4429>.

[RFC4903] Thaler, D., "Multi-Link Subnet Issues", RFC 4903, DOI 10.17487/RFC4903, June 2007, <http://www.rfc-editor.org/info/rfc4903>.

[RFC4919] Kushalnagar, N., Montenegro, G., and C. Schumacher, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals", RFC 4919, DOI 10.17487/RFC4919, August 2007, <http://www.rfc-editor.org/info/rfc4919>.

[RFC5191] Forsberg, D., Ohba, Y., Ed., Patil, B., Tschofenig, H., and A. Yegin, "Protocol for Carrying Authentication for Network Access (PANA)", RFC 5191, DOI 10.17487/RFC5191, May 2008, <http://www.rfc-editor.org/info/rfc5191>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <http://www.rfc-editor.org/info/rfc5340>.

[RFC5889] Baccelli, E., Ed. and M. Townsley, Ed., "IP Addressing Model in Ad Hoc Networks", RFC 5889, DOI 10.17487/RFC5889, September 2010, <http://www.rfc-editor.org/info/rfc5889>.

[RFC5974] Manner, J., Karagiannis, G., and A. McDonald, "NSIS Signaling Layer Protocol (NSLP) for Quality-of-Service Signaling", RFC 5974, DOI 10.17487/RFC5974, October 2010, <http://www.rfc-editor.org/info/rfc5974>.

[RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <http://www.rfc-editor.org/info/rfc6275>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <http://www.rfc-editor.org/info/rfc6347>.

[RFC6620] Nordmark, E., Bagnulo, M., and E. Levy-Abegnoli, "FCFS SAVI: First-Come, First-Served Source Address Validation Improvement for Locally Assigned IPv6 Addresses", RFC 6620, DOI 10.17487/RFC6620, May 2012, <http://www.rfc-editor.org/info/rfc6620>.

[RFC6655] McGrew, D. and D. Bailey, "AES-CCM Cipher Suites for Transport Layer Security (TLS)", RFC 6655, DOI 10.17487/RFC6655, July 2012, <http://www.rfc-editor.org/info/rfc6655>.

[RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, DOI 10.17487/RFC6830, January 2013, <http://www.rfc-editor.org/info/rfc6830>.

[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>.

[RFC7426] Haleplidis, E., Ed., Pentikousis, K., Ed., Denazis, S., Hadi Salim, J., Meyer, D., and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and Architecture Terminology", RFC 7426, DOI 10.17487/RFC7426, January 2015, <http://www.rfc-editor.org/info/rfc7426>.

8.3. Other Informative References

[ACE] IETF, "Authentication and Authorization for Constrained Environments", <https://dataTracker.ietf.org/doc/charter-ietf-ace/>.

[CCAMP] IETF, "Common Control and Measurement Plane", <https://dataTracker.ietf.org/doc/charter-ietf-ccamp/>.

[DETNET] IETF, "Deterministic Networking", <https://datatracker.ietf.org/doc/charter-ietf-detnet/>.

[DICE] IETF, "DTLS In Constrained Environments", <https://dataTracker.ietf.org/doc/charter-ietf-dice/>.

[HART] www.hartcomm.org, "Highway Addressable remote Transducer, a group of specifications for industrial process and control devices administered by the HART Foundation".

[IEC62439] IEC, "Industrial communication networks - High availability automation networks - Part 3: Parallel Redundancy Protocol (PRP) and High-availability Seamless Redundancy (HSR) - IEC62439-3", 2012, <https://webstore.iec.ch/publication/7018>.

[IEEE802.1TSNTG] IEEE Standards Association, "IEEE 802.1 Time-Sensitive Networks Task Group", March 2013, <http://www.ieee802.org/1/pages/avbridges.html>.

[IEEE802154] IEEE standard for Information Technology, "IEEE std. 802.15.4, Part. 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks".

[IEEE802154e]
IEEE standard for Information Technology, "IEEE standard for Information Technology, IEEE std. 802.15.4, Part. 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks, June 2011 as amended by IEEE std. 802.15.4e, Part. 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC sublayer", April 2012.

[ISA100] ISA/ANSI, "ISA100, Wireless Systems for Automation", <https://www.isa.org/isa100/>.

[ISA100.11a]
ISA/ANSI, "Wireless Systems for Industrial Automation: Process Control and Related Applications - ISA100.11a-2011 - IEC 62734", 2011, <http://www.isa.org/Community/SP100WirelessSystemsforAutomation>.

[PCE] IETF, "Path Computation Element", <https://dataTracker.ietf.org/doc/charter-ietf-pce/>.

[TEAS] IETF, "Traffic Engineering Architecture and Signaling", <https://dataTracker.ietf.org/doc/charter-ietf-teas/>.

[WirelessHART]
www.hartcomm.org, "Industrial Communication Networks - Wireless Communication Network and Communication Profiles - WirelessHART - IEC 62591", 2010.

Appendix A. Personal submissions relevant to upcoming work

This document covers a portion of the total work that is needed to cover the full 6TiSCH architecture. Missing portions at this time include Deterministic Networking with Track Forwarding, Dynamic Scheduling, and Security.

[I-D.richardson-6tisch-security-architecture] elaborates on the potential use of 802.1AR certificates, and some options for the join process are presented in more details.

[I-D.struik-6tisch-security-architecture-elements] describes 6TiSCH security architectural elements with high level requirements and the security framework that are relevant for the design of the 6TiSCH security solution.

Author’s Address

Pascal Thubert (editor)
Cisco Systems, Inc
Building D
45 Allee des Ormes - BP1200
MOUGINS - Sophia Antipolis 06254
FRANCE

Phone: +33 497 23 26 34
Email: pthubert@cisco.com

6TiSCH
Internet-Draft
Intended status: Informational
Expires: October 27, 2018

P. Thubert, Ed.
Cisco

April 25, 2018

An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4
draft-ietf-6tisch-architecture-14

Abstract

This document describes a network architecture that provides low-latency, low-jitter and high-reliability packet delivery. It combines a high speed powered backbone and subnetworks using IEEE 802.15.4 time-slotted channel hopping (TSCH) to meet the requirements of LowPower wireless deterministic applications.

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 October 27, 2018.

Table of Contents

1. Introduction 3
2. Terminology 4
2.1. BCP 14 4
2.2. References 4
2.3. Subset of a 6LoWPAN Glossary 5
3. High Level Architecture 6
3.1. 6TiSCH Stack 6
3.2. TSCH: A Deterministic MAC Layer 8
3.3. Scheduling TSCH 9
3.4. Routing and Forwarding Over TSCH 10
3.5. A Non-Broadcast Multi-Access Radio Mesh Network 12
3.6. A Multi-Link Subnet Model 14
3.7. Join Process and Registration 15
3.8. Dependencies on Work In Progress 18
4. Architecture Components 20
4.1. 6LoWPAN (and RPL) 20
4.1.1. RPL Leaf Support in 6LoWPAN ND 20
4.1.2. RPL Root And 6LBR 21
4.2. TSCH and 6top 22
4.2.1. 6top 22
4.2.2. Scheduling Functions and the 6P protocol 22
4.2.3. 6top and RPL Objective Function operations 23
4.2.4. Network Synchronization 24
4.2.5. SlotFrames and Priorities 25
4.2.6. Distributing the reservation of cells 26
4.3. Communication Paradigms and Interaction Models 28
4.4. Schedule Management Mechanisms 29
4.4.1. Static Scheduling 29
4.4.2. Neighbor-to-neighbor Scheduling 30
4.4.3. Remote Monitoring and Schedule Management 30
4.4.4. Hop-by-hop Scheduling 33
4.5. On Tracks 33
4.5.1. General Behavior of Tracks 33
4.5.2. Serial Track 34
4.5.3. Complex Track with Replication and Elimination . . . 35
4.5.4. DetNet End-to-end Path 35
4.5.5. Cell Reuse 36
4.6. Forwarding Models 37
4.6.1. Track Forwarding 37
4.6.2. Fragment Forwarding 40
4.6.3. IPv6 Forwarding 41
4.7. Centralized vs. Distributed Routing 42
4.7.1. Packet Marking and Handling 42
4.7.2. Replication, Retries and Elimination 43
4.7.3. Differentiated Services Per-Hop-Behavior 44
5. IANA Considerations 44

6. Security Considerations 44
6.1. Join Process Highlights 45
7. Acknowledgments 47
7.1. Contributors 47
7.2. Special Thanks 48
7.3. And Do not Forget 48
8. References 49
8.1. Normative References 49
8.2. Informative References 51
8.3. Other Informative References 56
Author's Address 57

1. Introduction

Wireless Networks enable a wide variety of devices of any size to get interconnected, often at a very low marginal cost per device, at any distance ranging from Near Field to interplanetary, and in circumstances where wiring may be impractical, for instance on fast-moving or rotating devices.

In the other hand, Deterministic Networks enable traffic that is highly sensitive to jitter, quite sensitive to latency, and with a high degree of operational criticality so that loss should be minimized at all times. Applications that need such networks are presented in [I-D.ietf-detnet-use-cases]. They include Professional Media and Operation Technology (OT) Industrial Automation Control Systems (IACS).

The Medium access Control (MAC) of IEEE Std 802.15.4 [IEEE802154] has evolved with the IEEE Std 802.15.4e Timeslotted Channel Hopping (TSCH) [RFC7554] mode to provide deterministic properties on wireless networks. TSCH was initially introduced with the IEEE Std 802.15.4e amendment [IEEE802154e] of the IEEE Std 802.15.4 standard and constituted a part of the standard from that day. For all practical purpose, this document is expected to be insensitive to the revisions of the IEEE Std 802.15.4 standard, which is thus referenced undated.

Proven Deterministic Networking standards for use in Process Control, including ISA100.11a [ISA100.11a] and WirelessHART [WirelessHART], have demonstrated the capabilities of the IEEE Std 802.15.4 TSCH MAC for high reliability against interference, low-power consumption on well-known flows, and its applicability for Traffic Engineering (TE) from a central controller.

In order to enable the convergence of IT and OT in LLN environments, 6TiSCH ports the IETF suite of protocol that are defined for such environments over the TSCH MAC. 6TiSCH also provides large scaling capabilities, which, in a number of scenarios, require the addition

of a high speed and reliable backbone and the use of IP version 6 (IPv6). The 6TiSCH Architecture introduces an IPv6 Multi-Link subnet model that is composed of a federating backbone and a number of IEEE Std 802.15.4 TSCH low-power wireless networks attached and synchronized by Backbone Routers.

The architecture defines mechanisms to establish and maintain routing and scheduling in a centralized, distributed, or mixed fashion, for use in multiple OT environments. It is applicable in particular to industrial control systems, building automation that leverage distributed routing to address multipath over a large number of hops, in-vehicle command and control that can be as demanding as industrial applications, commercial automation and asset Tracking with mobile scenarios, home automation and domotics which become more reliable and thus provide a better user experience, and resource management (energy, water, etc.).

2. Terminology

2.1. BCP 14

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119][RFC8174] when, and only when, they appear in all capitals, as shown here.

2.2. References

The draft uses domain-specific terminology defined or referenced in:

"Terms Used in IPv6 over the TSCH mode of IEEE 802.15.4e" [I-D.ietf-6tisch-terminology],

"Neighbor Discovery Optimization for Low-power and Lossy Networks" [RFC6775],

"Registration Extensions for 6LoWPAN Neighbor Discovery" [I-D.ietf-6lo-rfc6775-update], and

"Terms Used in Routing for Low-Power and Lossy Networks (LLNs)" [RFC7102].

Other terms in use in LLNs are found in "Terminology for Constrained-Node Networks" [RFC7228].

Readers are expected to be familiar with all the terms and concepts that are discussed in

- "Neighbor Discovery for IP version 6" [RFC4861],
- "IPv6 Stateless Address Autoconfiguration" [RFC4862],
- "Problem Statement and Requirements for IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Routing" [RFC6606].

The draft also conforms to the terms and models described in [RFC3444] and [RFC5889] and uses the vocabulary and the concepts defined in [RFC4291] for the IPv6 Architecture and refers [RFC4080] for reservation

In addition, readers would benefit from reading:

- "Multi-Link Subnet Issues" [RFC4903],
- "Mobility Support in IPv6" [RFC6275],
- "RPL applicability in industrial networks" [I-D.ietf-roll-rpl-industrial-applicability],
- "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals" [RFC4919].
- "Optimistic Duplicate Address Detection" [RFC4429],
- "Neighbor Discovery Proxies (ND Proxy)" [RFC4389],
- "FCFS SAVI: First-Come, First-Served Source Address Validation Improvement for Locally Assigned IPv6 Addresses" [RFC6620], and
- "Optimistic Duplicate Address Detection" [RFC4429]

prior to this specification for a clear understanding of the art in ND-proxying and binding.

2.3. Subset of a 6LoWPAN Glossary

This document often uses the following acronyms:

6BBR: 6LoWPAN Backbone Router (proxy for the registration)

6LBR: 6LoWPAN Border Router (authoritative on DAD)

6LN: 6LoWPAN Node

6LR: 6LoWPAN Router (relay to the registration process)

6CIO: Capability Indication Option

(E)ARO: (Extended) Address Registration Option

(E)DAR: (Extended) Duplicate Address Request

(E)DAC: (Extended) Duplicate Address Confirmation

DAD: Duplicate Address Detection

DODAG: Destination-Oriented Directed Acyclic Graph

LLN: Low-Power and Lossy Network (a typical IoT network)

NA: Neighbor Advertisement

NCE: Neighbor Cache Entry

ND: Neighbor Discovery

NDP: Neighbor Discovery Protocol

NS: Neighbor Solicitation

ROVR: Registration Ownership Verifier (pronounced rover)

RPL: IPv6 Routing Protocol for LLNs (pronounced ripple)

RA: Router Advertisement

RS: Router Solicitation

TSCH: Timeslotted Channel Hopping

TID: Transaction ID (a sequence counter in the EARO)

3. High Level Architecture

3.1. 6TiSCH Stack

The 6TiSCH architecture presents a reference stack that is implemented and interop tested by a conjunction of opensource, IETF and ETSI efforts. One goal is to help other bodies to adopt the stack as a whole, making the effort to move to an IPv6-based IOT stack easier. Now, for a particular, environment, some of the choices that are made in this architecture may not be relevant. For instance, RPL is not required for star topologies and mesh-under Layer-2 routed networks, and the 6LoWPAN compression may not be

sufficient for ultra-constrained cases such as some Low Power Wide Area (LPWA) networks. In such cases, it is perfectly doable to adopt a subset of the selection that is presented hereafter and then select alternate components to complete the solution wherever needed.

The IETF proposes multiple techniques for implementing functions related to routing, transport or security. In order to control the complexity of the possible deployments and device interactions, and to limit the size of the resulting object code, the architecture limits the possible variations of the stack and recommends a number of base elements for LLN applications. In particular, UDP [RFC0768] [RFC8200] and the Constrained Application Protocol [RFC7252] (CoAP) are used as the transport / binding of choice for applications and management as opposed to TCP and HTTP.

The resulting protocol stack is represented below:

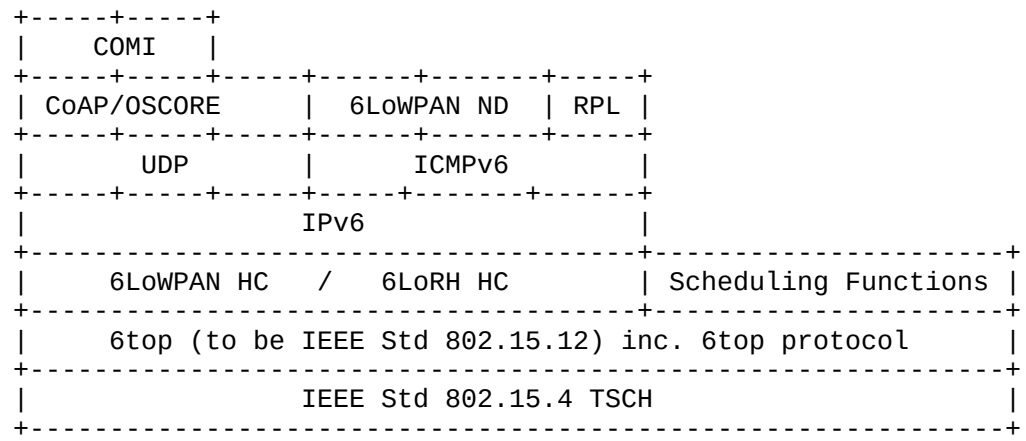


Figure 1: 6TiSCH Protocol Stack

RPL is the routing protocol of choice for LLNs. So far, there was no identified need to define a 6TiSCH specific Objective Function. The Minimal 6TiSCH Configuration [RFC8180] describes the operation of RPL over a static schedule used in a slotted aloha fashion, whereby all active slots may be used for emission or reception of both unicast and multicast frames.

The 6LoWPAN Header Compression [RFC6282] is used to compress the IPv6 and UDP headers, whereas the 6LoWPAN Routing Header (6LoRH) [RFC8138] is used to compress the RPL artifacts in the IPv6 data packets, including the RPL Packet Information (RPI), the IP-in-IP

encapsulation to/from the RPL root, and the Source Route Header (SRH) in non-storing mode.

6TiSCH has adopted the general direction of CoAP Management Interface (COMI) [I-D.ietf-core-comi] for the management of devices.

The Datagram Transport Layer Security (DTLS) [RFC6347] sitting either under CoAP or over CoAP so as to traverse proxies, as well as Object Security for Constrained RESTful Environments (OSCORE) [I-D.ietf-core-object-security], are examples of protocol that could be used to protect application payload, and OSCORE is used in particular by the "Minimal Security Framework for 6TiSCH" [I-D.ietf-6tisch-minimal-security] for the the Join Process.

An overview of the the initial steps of a device in a network can be found in Section 3.7; the security aspects of the join process are further detailed in Section 6.

The 6TiSCH Operation sublayer (6top) [I-D.wang-6tisch-6top-sublayer] is a sublayer of a Logical Link Control (LLC) that provides the abstraction of an IP link over a TSCH MAC and schedules packets over TSCH cells,as further discussed in the next sections.

3.2. TSCH: A Deterministic MAC Layer

Though at a different time scale (several orders of magnitude), both IEEE Std 802.1TSN and IEEE Std 802.15.4TSCH standards provide Deterministic capabilities to the point that a packet that pertains to a certain flow may traverse a network from node to node following a very precise schedule, as a train that enters and then leaves intermediate stations at precise times along its path. With TSCH, time is formatted into timeslots, and individual communication cells are allocated to unicast or broadcast communication at the MAC level. The time-slotted operation reduces collisions, saves energy, and enables to more closely engineer the network for deterministic properties. The channel hopping aspect is a simple and efficient technique to combat multipath fading and external interference (for example by Wi-Fi emitters).

6TiSCH builds on the IEEE Std 802.15.4TSCH MAC and inherits its advanced capabilities to enable them in multiple environments where they can be leveraged to improve automated operations. The 6TiSCH Architecture also inherits the capability to perform a centralized route computation to achieve deterministic properties, though it relies on the IETF DetNet Architecture [I-D.ietf-detnet-architecture], and IETF components such as the Path Computation Element (PCE) [PCE], for the protocol aspects.

On top of this inheritance, 6TiSCH adds capabilities for distributed routing and scheduling operations based on the RPL routing protocol and capabilities to negotiate schedule adjustments between peers. These distributed routing and scheduling operations simplify the deployment of TSCH networks and enable wireless solutions in a larger variety of use cases from operational technology in general. Examples of such use-cases in industrial environments include plant setup and decommissioning, as well as monitoring of lots of lesser importance measurements such as corrosion and events. RPL also enables mobile use cases such as mobile workers and cranes, as presented in [I-D.ietf-roll-rpl-industrial-applicability].

3.3. Scheduling TSCH

A scheduling operation attributes cells in a Time-Division-Multiplexing (TDM) / Frequency-Division Multiplexing (FDM) matrix called the Channel distribution/usage (CDU) to either individual transmissions or as multi-access shared resources (see the 6TiSCH Terminology [I-D.ietf-6tisch-terminology] for more on these terms). Scheduling effectively enables multiple communications at a same time in a same interference domain using different channels; but a node equipped with a single radio can only transmit or receive on one channel at any given point of time.

From the standpoint of a 6TiSCH node (at the MAC layer), its schedule is the collection of the times at which it must wake up for transmission, and the channels to which it should either send or listen at those times. The schedule is expressed as one or more slotframes that repeat over and over. Slotframes may collide and require a device to wake at a same time, in which case a priority indicates which slotframe is actually activated.

The 6top sublayer hides the complexity of the schedule to the upper layers. The Link that IP may utilize between the 6TiSCH node and a peer may in fact be composed of a pair of cell bundles, one to receive and one to transmit. Some of the cells may be shared, in which case the 6top sublayer must perform some arbitration.

The 6TiSCH architecture identifies four ways a schedule can be managed and CDU cells can be allocated: Static Scheduling, Neighbor-to-Neighbor Scheduling, Remote Monitoring and Schedule Management, and Hop-by-hop Scheduling.

Static Scheduling: This refers to the minimal 6TiSCH operation whereby a static schedule is configured for the whole network for use in a slotted-aloha fashion. The static schedule is distributed through the native methods in the TSCH MAC layer. This operation leverages RPL to maintain a loopless graph for

routing and time distribution. It is specified in the Minimal 6TiSCH Configuration [RFC8180] specification. and does not preclude other scheduling operations to co-exist on a same 6TiSCH network.

Neighbor-to-Neighbor Scheduling: This refers to the dynamic adaptation of the bandwidth of the Links that are used for IPv6 traffic between adjacent routers. Scheduling Functions such as the "6TiSCH Minimal Scheduling Function (MSF)" [I-D.chang-6tisch-msf] influence the operation of the MAC layer to add, update and remove cells in peers schedule, using the "6top Protocol (6P)" [I-D.ietf-6tisch-6top-protocol] for the negotiation of the MAC resources.

Remote Monitoring and Schedule Management: This refers to the central computation of a schedule and the capability to forward a frame based on the cell of arrival. In that case, the related portion of the device schedule as well as other device resources are managed by an abstract Network Management Entity (NME), which may cooperate with the PCE in order to minimize the interaction with and the load on the constrained device. This model is the TSCH adaption of the "DetNet Architecture" [I-D.ietf-detnet-architecture], and it enables Traffic Engineering with deterministic properties.

Hop-by-hop Scheduling: This refers to the possibility to reserves cells along a path for a particular flow using a distributed mechanism.

It is not expected that all use cases will require all those mechanisms. Static Scheduling with minimal configuration one is the only one that is expected in all implementations, since it provides a simple and solid basis for convergecast routing and time distribution.

A deeper dive in those mechanisms can be found in Section 4.4.

3.4. Routing and Forwarding Over TSCH

6TiSCH leverages the RPL routing protocol for interoperable distributed routing operations. RPL is applicable to Static Scheduling and Neighbor-to-Neighbor Scheduling. The architecture also supports a centralized routing model for Remote Monitoring and Schedule Management. It is expected that a routing protocol that is more optimized for point-to-point routing than RPL [RFC6550], such as the "Asymmetric AODV-P2P-RPL in Low-Power and Lossy Networks" [I-D.ietf-roll-aodv-rpl] (AODV-RPL), which derives from the Ad Hoc

On-demand Distance Vector Routing (AODV) [I-D.ietf-manet-aodvv2] will be selected for Hop-by-hop Scheduling.

The 6TiSCH architecture supports three different forwarding models, the classical IPv6 Forwarding, where the node selects a feasible successor at Layer-3 on a per packet basis and based on its routing table, G-MPLS Track Forwarding, which switches a frame received at a particular Timeslot into another Timeslot at Layer-2, and 6LoWPAN Fragment Forwarding, which allows to forward individual 6loWPAN fragments along the route set by the first fragment.

IPv6 Forwarding: This is the classical IP forwarding model, with a Routing Information Based (RIB) that is installed by the RPL routing protocol and used to select a feasible successor per packet. The packet is placed on an outgoing Link, that the 6top layer maps into a (Layer-3) bundle of cells, and scheduled for transmission based on QoS parameters. On top of RPL, this model also applies to any routing protocol which may be operated in the 6TiSCH network, and corresponds to all the distributed scheduling models, Static, Neighbor-to-Neighbor and Hop-by-Hop Scheduling.

G-MPLS Track Forwarding: This model corresponds to the Remote Monitoring and Schedule Management. In this model, A central controller (hosting a PCE) computes and installs the schedules in the devices per flow. The incoming (Layer-2) bundle of cells from the previous node along the path determines the outgoing (Layer-2) bundle towards the next hop for that flow as determined by the PCE. The programmed sequence for bundles is called a Track and can assume shapes that are more complex than a simple direct sequence of nodes.

6LoWPAN Fragment Forwarding: This is an hybrid model that derives from IPv6 forwarding for the case where packets must be fragmented at the 6LoWPAN sublayer. The first fragment is forwarded like any IPv6 packet and leaves a state in the intermediate hops to enable forwarding of the next fragments that do not have a IP header without the need to recompose the packet at every hop.

This can be broadly summarized in the following table:

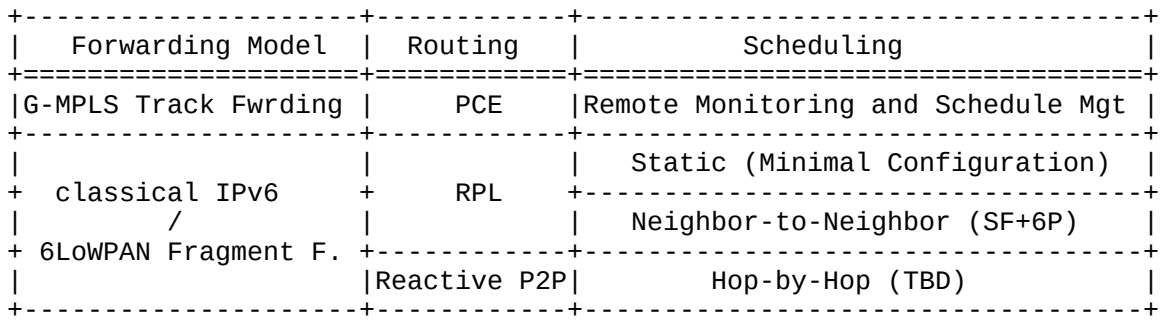


Figure 2: Routing, Forwarding and Scheduling

3.5. A Non-Broadcast Multi-Access Radio Mesh Network

A 6TiSCH network is an IPv6 [RFC8200] subnet which, in its basic configuration, is a single Low Power Lossy Network (LLN) operating over a synchronized TSCH-based mesh.

Inside a 6TiSCH LLN, nodes rely on 6LoWPAN Header Compression (6LoWPAN HC) [RFC6282] to encode IPv6 packets. From the perspective of the network layer, a single LLN interface (typically an IEEE Std 802.15.4-compliant radio) may be seen as a collection of Links with different capabilities for unicast or multicast services.

6TiSCH nodes are not necessarily reachable from one another at Layer-2 and an LLN may span over multiple links. This effectively forms an homogeneous non-broadcast multi-access (NBMA) subnet, which is beyond the scope of existing IPv6 ND methods. Extensions to IPv6 ND have to be introduced.

Within that subnet, neighbor devices are discovered with 6LoWPAN Neighbor Discovery [RFC6775] (6LoWPAN ND), whereas RPL [RFC6550] enables routing in the so called Route Over fashion, either in storing (stateful) or non-storing (stateless, with routing headers) mode.

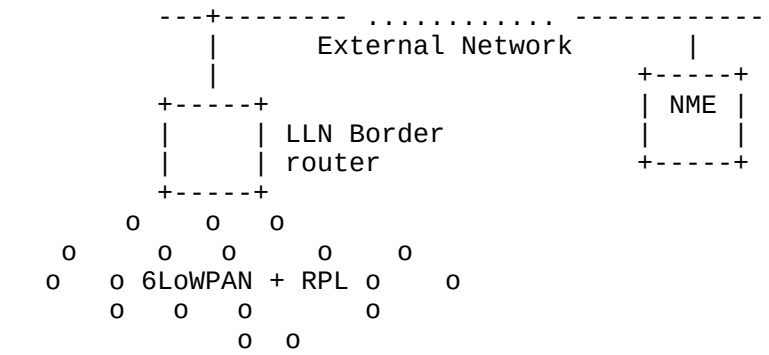


Figure 3: Basic Configuration of a 6TiSCH Network

6TiSCH nodes join the mesh by attaching to nodes that are already members of the mesh. Some nodes act as routers for 6LoWPAN ND and RPL operations, as detailed in Section 4.1. Security aspects of the join process by which a device obtains access to the network are discussed in Section 6.

With TSCH, devices are time-synchronized at the MAC level. The use of a particular RPL Instance for time synchronization is discussed in Section 4.2.4. With this mechanism, the time synchronization starts at the RPL root and follows the RPL DODAGs with no timing loop.

RPL forms Destination Oriented Directed Acyclic Graphs (DODAGs) within Instances of the protocol, each Instance being associated with an Objective Function (OF) to form a routing topology. A particular 6TiSCH node, the LLN Border Router (LBR), acts as RPL root, 6LoWPAN HC terminator, and Border Router for the LLN to the outside. The LBR is usually powered. More on RPL Instances can be found in section 3.1 of RPL [RFC6550], in particular "3.1.2. RPL Identifiers" and "3.1.3. Instances, DODAGs, and DODAG Versions". RPL adds artifacts in the data packets that are compressed with a 6LoWPAN addition 6LoRH [RFC8138].

Additional routing and scheduling protocols may be deployed to establish on-demand Peer-to-Peer routes with particular characteristics inside the 6TiSCH network. This may be achieved in a centralized fashion by a PCE [PCE] that programs both the routes and the schedules inside the 6TiSCH nodes, or by in a distributed fashion using a reactive routing protocol and a Hop-by-Hop scheduling protocol.

A Backbone Router may be connected to the node that acts as RPL root and / or 6LoWPAN 6LBR and provides connectivity to the larger campus / factory plant network over a high speed backbone or a back-haul

link. A Backbone Router may perform proxy IPv6 Neighbor Discovery (ND) [RFC4861] operations over the backbone on behalf of the 6TiSCH nodes so they can share a same IPv6 subnet and appear to be connected to the same backbone as classical devices. A Backbone Router may alternatively redistribute the registration in a routing protocol such as OSPF [RFC5340] or BGP [RFC2545], or inject them in a mobility protocol such as MIPv6 [RFC6275], NEMO [RFC3963], or LISP [RFC6830].

This architecture expects that a 6LoWPAN node can connect as a leaf to a RPL network, where the leaf support is the minimal functionality to connect as a host to a RPL network without the need to participate to the full routing protocol. The architecture also expects that a 6LoWPAN node that is not aware at all of the RPL protocol may also connect as a host but the specifications for this to happen are not available at the time of this writing.

3.6. A Multi-Link Subnet Model

An extended configuration of the subnet comprises multiple LLNs. The LLNs are interconnected and synchronized over a backbone, that can be wired or wireless. The backbone can be a classical IPv6 network, with Neighbor Discovery operating as defined in [RFC4861] and [RFC4862]. This architecture requires work to standardize the the registration of 6LoWPAN nodes to the Backbone Routers.

In the extended configuration, a Backbone Router (6BBR) operates as described in [I-D.ietf-6lo-backbone-router]. The 6BBR performs ND proxy operations between the registered devices and the classical ND devices that are located over the backbone. 6TiSCH 6BBRs synchronize with one another over the backbone, so as to ensure that the multiple LLNs that form the IPv6 subnet stay tightly synchronized.

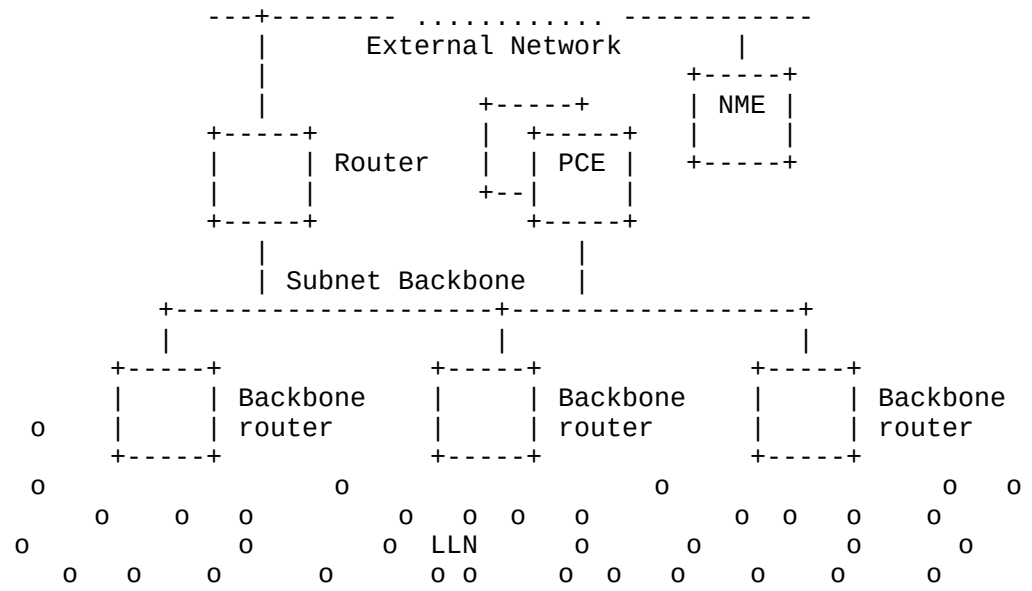


Figure 4: Extended Configuration of a 6TiSCH Network

As detailed in Section 4.1 the 6LoWPAN ND 6LBR and the root of the RPL network need to be collocated and share information about the devices that is learned through either protocol but not both. The combined RPL root and 6LBR may be collocated with the 6BBR, or directly attached to the 6BBR. In the latter case, it leverages the extended registration process defined in [I-D.ietf-6lo-backbone-router] to proxy the 6LoWPAN ND registration to the 6BBR on behalf of the LLN nodes, so that the 6BBR may in turn perform proxy classical ND operations over the backbone.

If the Backbone is Deterministic (such as defined by the Time Sensitive Networking WG at IEEE), then the Backbone Router ensures that the end-to-end deterministic behavior is maintained between the LLN and the backbone. The DetNet Architecture [I-D.ietf-detnet-architecture] studies Layer-3 aspects of Deterministic Networks, and covers networks that span multiple Layer-2 domains.

3.7. Join Process and Registration

As detailed in Section 6, a node that wishes to join the 6TiSCH network with a preshared key (PSK) performs the role of the pledge in the 6TiSCH Join Protocol (6JP) [I-D.ietf-6tisch-minimal-security] protocol. In order to join, the pledge is helped by a Join Proxy (JP) that relays the link-scope 6JP Join request over the IP network

to the Join Registrar/Coordinator (JRC) that can authenticate the pledge and validate that it is attached to the appropriate network. As a result of this exchange the pledge is in possession of a link-layer material including a key and a short address, and all traffic is secured at the link-layer .

Figure 5 illustrates that very initial step.

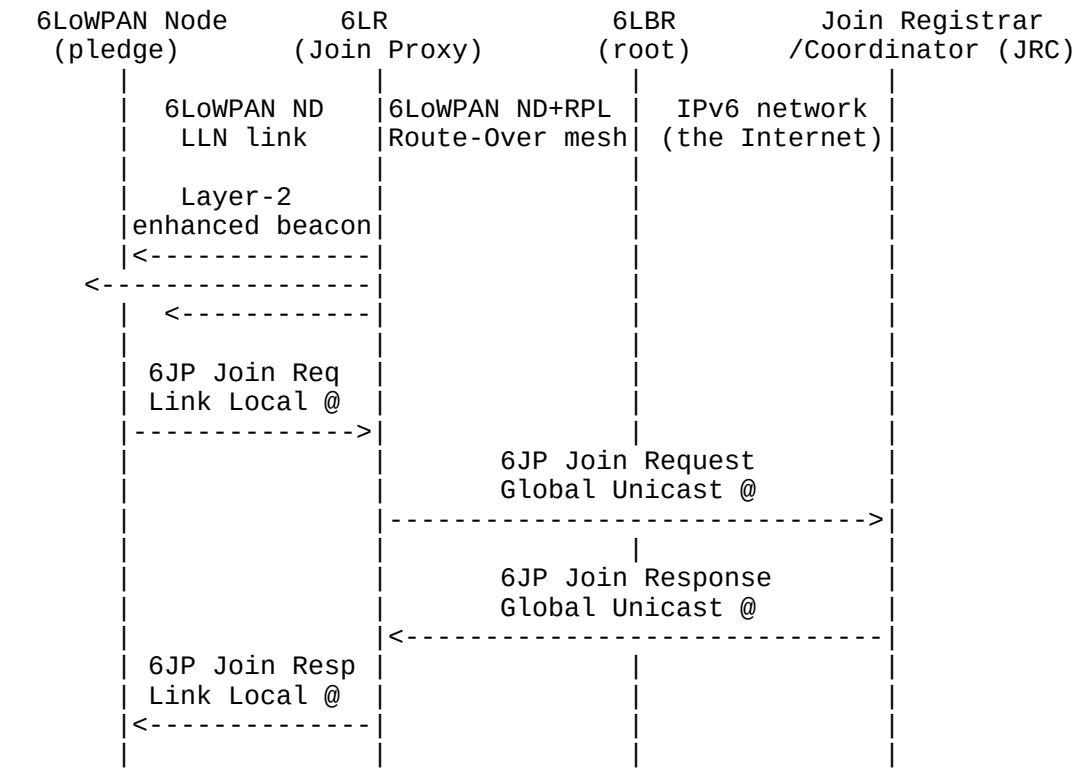


Figure 5: (Re-)Registration Flow over Multi-Link Subnet

As detailed in Section 4.1, the combined 6LoWPAN ND 6LBR and root of the RPL network learn information such as the device Unique ID (from 6LoWPAN ND) and the updated Sequence Number (from RPL), and perform 6LoWPAN ND proxy registration to the 6BBR of behalf of the LLN nodes.

Figure 6 illustrates the initial IPv6 signaling that eables a 6LN to form a global address and register it to a 6LBR using 6LoWPAN ND [I-D.ietf-6lo-rfc6775-update], is then carried over RPL to the RPL root, and then to the 6BBR.

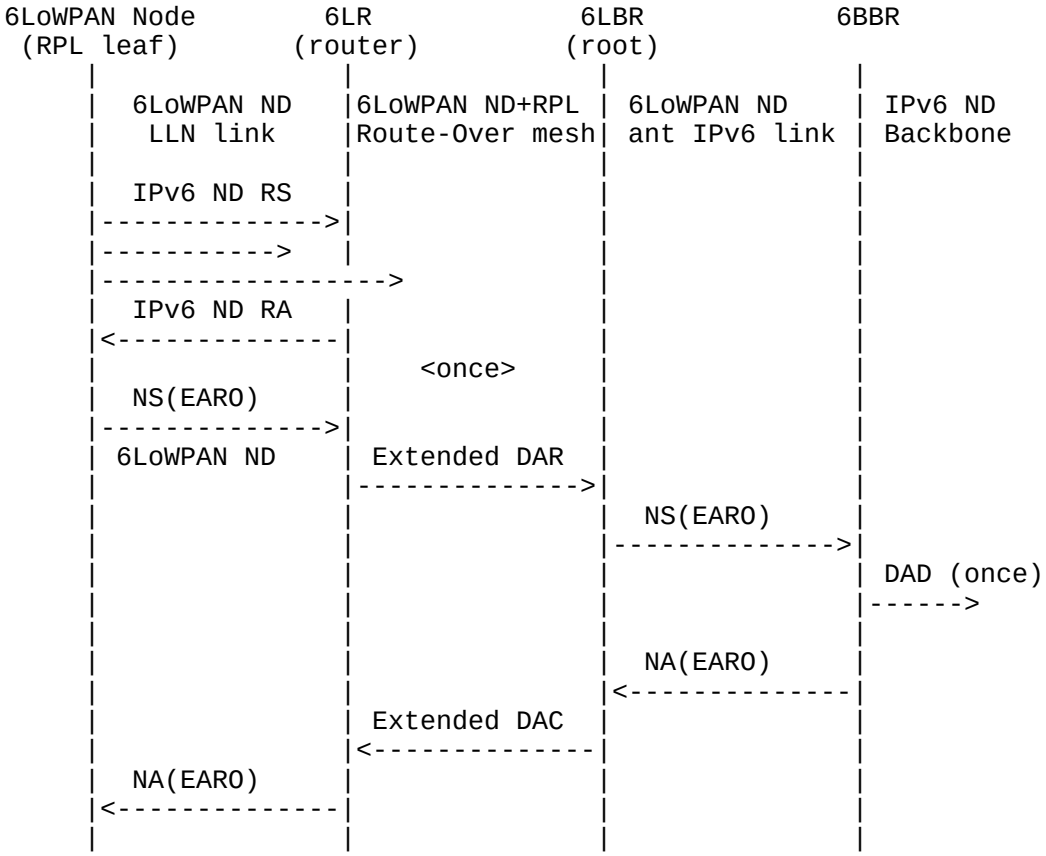


Figure 6: Initial Registration Flow over Multi-Link Subnet

Figure 7 illustrates the repeating IPv6 signaling that enables a 6LN to keep a global address alive and registered to its 6LBR using 6LoWPAN ND [I-D.ietf-6lo-rfc6775-update], using 6LoWPAN ND ot the 6LR, RPL to the RPL root, and then 6LoWPAN ND again to the 6BBR.

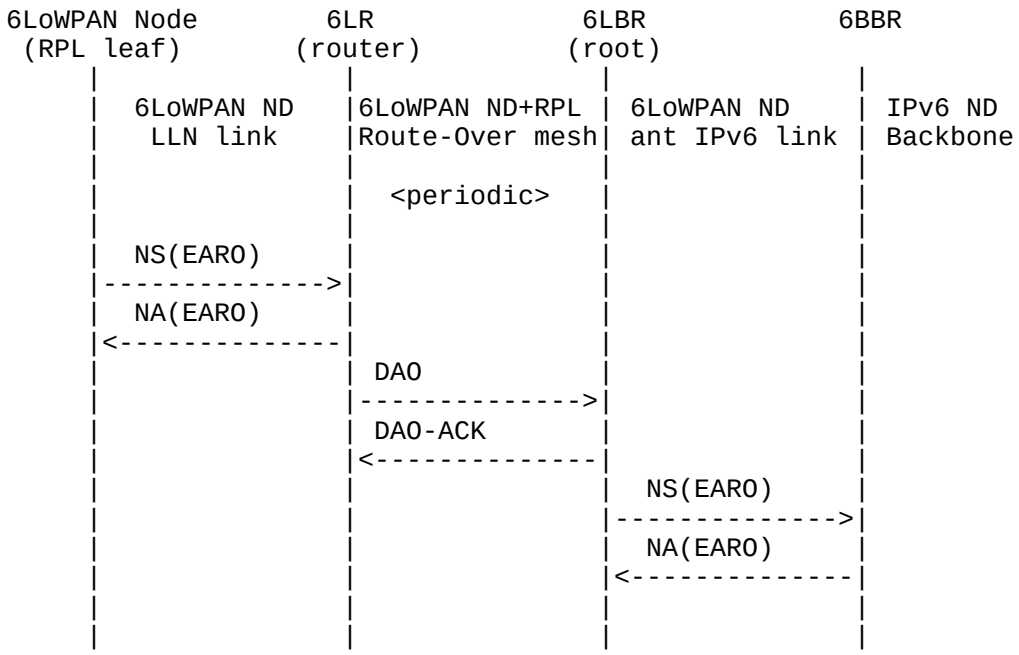


Figure 7: Next Registration Flow over Multi-Link Subnet

As the network builds up, a node should start as a leaf to join the RPL network, and may later turn into both a RPL-capable router and a 6LR, so as to accept leaf nodes to recursively join the network.

3.8. Dependencies on Work In Progress

In order to control the complexity and the size of the 6TiSCH work, the architecture and the associated IETF work are staged and the WG is expected to recharter multiple times. This document is incremented as the work progresses following the evolution of the WG charter and the availability of dependent work. The intent is to publish when the WG concludes.

At the time of this writing:

- The architecture of the operation of RPL over a dynamic schedule is being studied at 6TISCH as the second iteration of the charter.

- The need of a reactive routing protocol to establish on-demand constraint-optimized routes and a reservation protocol to

establish Layer-3 Tracks is being discussed at 6TiSCH but not chartered for.

- The components and protocols that are required to implement this stage of architecture are being standardized at the IETF. An Update to 6LoWPAN ND [I-D.ietf-6lo-rfc6775-update] covers the evolution of 6LoWPAN Neighbor Discovery that is needed to implement the Backbone Router [I-D.ietf-6lo-backbone-router]. In addition the protection of registered addresses against impersonation and take over can be guaranteed by Address Protected Neighbor Discovery for Low-power and Lossy Networks [I-D.ietf-6lo-ap-nd].

- The work on centralized Track computation is deferred to a subsequent iteration of the 6TiSCH charter. The idea at the time of this writing is that 6TiSCH will apply the concepts of Deterministic Networking on a Layer-3 network. The 6TiSCH Architecture should thus inherit from the DetNet [I-D.ietf-detnet-architecture] architecture and thus depends on it. The Path Computation Element (PCE) should be a core component of that architecture. Around the PCE, a protocol such as an extension to a TEAS [TEAS] protocol will be required to expose the 6TiSCH node capabilities and the network peers to the PCE, and a protocol such as a lightweight PCEP or an adaptation of CCAMP [CCAMP] G-MPLS formats and procedures will be used to publish the Tracks, as computed by the PCE, to the 6TiSCH nodes.

- BIER-TE-based OAM, Replication and Elimination [I-D.thubert-bier-replication-elimination] leverages Bit Index Explicit Replication - Traffic Engineering to control in the data plane the DetNet Replication and Elimination activities, and to provide traceability on links where replication and loss happen, in a manner that is abstract to the forwarding information, whereas a 6loRH for BitStrings [I-D.thubert-6lo-bier-dispatch] proposes a 6LoWPAN compression for the BIER Bitstring based on 6LoWPAN Routing Header [RFC8138].

- The security model and in particular the join process depends on the ANIMA [ANIMA] Bootstrapping Remote Secure Key Infrastructures (BRSKI) [I-D.ietf-anima-bootstrapping-keyinfra] in order to enable zero-touch security provisionning; for highly constrained nodes, a minimal model based on pre-shared keys (PSK) is also available.

- The current charter positions 6TiSCH on IEEE Std 802.15.4 only. Though most of the design should be portable on other link types, 6TiSCH has a strong dependency on IEEE Std 802.15.4 and its evolution. At the time of this writing, a revision of the IEEE Std 802.15.4 standard is expected early 2016. That revision

should integrate TSCH as well as other amendments and fixes into the main specification. The impact on this Architecture should be minimal to non-existent, but deeper work such as 6top and security may be impacted. A 6TiSCH Interest Group was formed at IEEE to maintain the synchronization and help foster work at the IEEE should 6TiSCH demand it.

- Work is being proposed at IEEE (802.15.12 PAR) for an LLC that would logically include the 6top sublayer. The interaction with the 6top sublayer and the Scheduling Functions described in this document are yet to be defined.

- ISA100 [ISA100] Common Network Management (CNM) is another external work of interest for 6TiSCH. The group, referred to as ISA100.20, defines a Common Network Management framework that should enable the management of resources that are controlled by heterogeneous protocols such as ISA100.11a [ISA100.11a], WirelessHART [WirelessHART], and 6TiSCH. Interestingly, the establishment of 6TiSCH Deterministic paths, called Tracks, are also in scope, and ISA100.20 is working on requirements for DetNet.

4. Architecture Components

4.1. 6LoWPAN (and RPL)

4.1.1. RPL Leaf Support in 6LoWPAN ND

RPL needs a set of information in order to advertise a leaf node through a DAO message and establish reachability.

At the bare minimum the leaf device must provide a sequence number that matches the RPL specification in section 7. Section 5.3 of [I-D.ietf-6lo-backbone-router], on the Extended Address Registration Option (EARO), already incorporates that addition with a new field in the option called the Transaction ID.

If for some reason the node is aware of RPL topologies, then providing the RPL InstanceID for the instances to which the node wishes to participate would be a welcome addition. In the absence of such information, the RPL router must infer the proper instanceID from external rules and policies.

On the backbone, the InstanceID is expected to be mapped onto a an overlay that matches the instanceID, for instance a VLANID.

This architecture leverages [I-D.ietf-6lo-backbone-router] that extends 6LoWPAN ND [RFC6775] to carry the counter as an abstract Transaction ID (TID).

4.1.2. RPL Root And 6LBR

With [RFC6775], information on the 6LBR is disseminated via an Authoritative Border Router Option (ABRO) in RA messages. The discovery and liveliness of the RPL root are obtained through the RPL protocol [RFC6550]. The capability to support the update to RFC6775 [I-D.ietf-6lo-rfc6775-update] is indicated in the 6LoWPAN Capability Indication Option (6CIO).

"Routing for RPL Leaves" [I-D.thubert-roll-unaware-leaves] details the basic interaction of 6LoWPAN ND and RPL and enables a plain 6LN that supports [I-D.ietf-6lo-rfc6775-update] to obtain return connectivity via the RPL network as a non-RPL-aware leaf. Though the above specification enables a model where the separation is possible, this architecture recommends to collocate the functions of LBR and RPL root.

When 6LoWPAN ND is coupled with RPL, the 6LBR and RPL root functionalities are co-located in order that the address of the 6LBR be indicated by RPL DIO messages and to associate the unique ID from the DAR/DAC exchange with the state that is maintained by RPL. The DAR/DAC exchange becomes a preamble to the DAO messages that are used from then on to reconfirm the registration, thus eliminating a duplication of functionality between DAO and DAR messages.

Even though the root of the RPL network is integrated with the 6LBR, it is logically separated from the Backbone Router (6BBR) that is used to connect the 6TiSCH LLN to the backbone. This way, the root has all information from 6LoWPAN ND and RPL about the LLN devices attached to it.

This architecture also expects that the root of the RPL network (proxy-)registers the 6TiSCH nodes on their behalf to the 6BBR, for whatever operation the 6BBR performs on the backbone, such as ND proxy, or redistribution in a routing protocol. This relies on an extension of the 6LoWPAN ND registration described in [I-D.ietf-6lo-backbone-router].

This model supports the movement of a 6TiSCH device across the Multi-Link Subnet, and allows the proxy registration of 6TiSCH nodes deep into the 6TiSCH LLN by the 6LBR / RPL root. This requires an alteration from [RFC6775] whereby the Target Address of the NS message is registered as opposed to the Source, which, in the case of a proxy registration, is that of the 6LBR / RPL root itself.

4.2. TSCH and 6top

4.2.1. 6top

6top is a logical link control sitting between the IP layer and the TSCH MAC layer, which provides the link abstraction that is required for IP operations. The 6top operations are specified in [I-D.ietf-6tisch-6top-protocol]. In particular, 6top provides a management interface that enables an external management entity to schedule cells and slotFrames, and allows the addition of complementary functionality, for instance to support a dynamic schedule management based on observed resource usage as discussed in Section 4.4.2.

The 6top data model and management interfaces are further discussed in Section 4.4.3.

4.2.1.1. Hard Cells

The architecture defines "soft" cells and "hard" cells. "Hard" cells are owned and managed by an separate scheduling entity (e.g. a PCE) that specifies the slotOffset/channelOffset of the cells to be added/moved/deleted, in which case 6top can only act as instructed, and may not move hard cells in the TSCH schedule on its own.

4.2.1.2. Soft Cells

6top contains a monitoring process which monitors the performance of cells, and can move a cell in the TSCH schedule when it performs poorly. This is only applicable to cells which are marked as "soft". To reserve a soft cell, the higher layer does not indicate the exact slotOffset/channelOffset of the cell to add, but rather the resulting bandwidth and QoS requirements. When the monitoring process triggers a cell reallocation, the two neighbor devices communicating over this cell negotiate its new position in the TSCH schedule.

4.2.2. Scheduling Functions and the 6P protocol

In the case of soft cells, the cell management entity that controls the dynamic attribution of cells to adapt to the dynamics of variable rate flows is called a Scheduling Function (SF). There may be multiple SFs with more or less aggressive reaction to the dynamics of the network. The "6TiSCH Minimal Scheduling Function (MSF)" [I-D.chang-6tisch-msf] provides a simple scheduling function that can be used by default by devices that support dynamic scheduling of soft cells.

The SF may be seen as divided between an upper bandwidth adaptation logic that is not aware of the particular technology that is used to obtain and release bandwidth, and an underlying service that maps those needs in the actual technology, which means mapping the bandwidth onto cells in the case of TSCH.

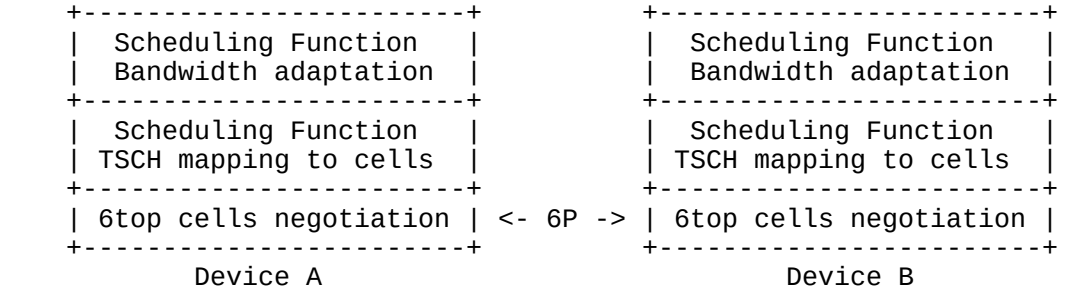


Figure 8: SF/6P stack in 6top

The SF relies on 6top services that implement the 6top Protocol (6P) [I-D.ietf-6tisch-6top-protocol] to negotiate the precise cells that will be allocated or freed based on the schedule of the peer. It may be for instance that a peer wants to use a particular time slot that is free in its schedule, but that timeslot is already in use by the other peer for a communication with a third party on a different cell. The 6P protocol enables the peers to find an agreement in a transactional manner that ensures the final consistency of the nodes state.

4.2.3. 6top and RPL Objective Function operations

An implementation of a RPL [RFC6550] Objective Function (OF), such as the RPL Objective Function Zero (OF0) [RFC6552] that is used in the Minimal 6TiSCH Configuration [RFC8180] to support RPL over a static schedule, may leverage, for its internal computation, the information maintained by 6top.

Most OFs require metrics about reachability, such as the ETX. 6top creates and maintains an abstract neighbor table, and this state may be leveraged to feed an OF and/or store OF information as well. In particular, 6top creates and maintains an abstract neighbor table. A neighbor table entry contains a set of statistics with respect to that specific neighbor including the time when the last packet has been received from that neighbor, a set of cell quality metrics (e.g. RSSI or LQI), the number of packets sent to the neighbor or the number of packets received from it. This information can be obtained through 6top management APIs as detailed in the 6top sublayer

specification [I-D.wang-6tisch-6top-sublayer] and used for instance to compute a Rank Increment that will determine the selection of the preferred parent.

6top provides statistics about the underlying layer so the OF can be tuned to the nature of the TSCH MAC layer. 6top also enables the RPL OF to influence the MAC behaviour, for instance by configuring the periodicity of IEEE Std 802.15.4 Extended Beacons (EB's). By augmenting the EB periodicity, it is possible to change the network dynamics so as to improve the support of devices that may change their point of attachment in the 6TiSCH network.

Some RPL control messages, such as the DODAG Information Object (DIO) are ICMPv6 messages that are broadcast to all neighbor nodes. With 6TiSCH, the broadcast channel requirement is addressed by 6top by configuring TSCH to provide a broadcast channel, as opposed to, for instance, piggybacking the DIO messages in Enhance Beacons. Consideration was given towards finding a way to embed the Route Advertisements and the RPL DIO messages (both of which are multicast) into the IEEE Std 802.15.4 Enhanced Beacons. It was determined that this produced undue timer coupling among layers, that the resulting packet size was potentially too large, and required it is not yet clear that there is any need for Enhanced Beacons in a production network.

4.2.4. Network Synchronization

Nodes in a TSCH network must be time synchronized. A node keeps synchronized to its time source neighbor through a combination of frame-based and acknowledgment-based synchronization. In order to maximize battery life and network throughput, it is advisable that RPL ICMP discovery and maintenance traffic (governed by the trickle timer) be somehow coordinated with the transmission of time synchronization packets (especially with enhanced beacons). This could be achieved through an interaction of the 6top sublayer and the RPL objective Function, or could be controlled by a management entity.

Time distribution requires a loop-less structure. Nodes taken in a synchronization loop will rapidly desynchronize from the network and become isolated. It is expected that a RPL DAG with a dedicated global Instance is deployed for the purpose of time synchronization. That Instance is referred to as the Time Synchronization Global Instance (TSGI). The TSGI can be operated in either of the 3 modes that are detailed in section 3.1.3 of RPL [RFC6550], "Instances, DODAGs, and DODAG Versions". Multiple uncoordinated DODAGs with independent roots may be used if all the roots share a common time source such as the Global Positioning System (GPS). In the absence

of a common time source, the TSGI should form a single DODAG with a virtual root. A backbone network is then used to synchronize and coordinate RPL operations between the backbone routers that act as sinks for the LLN. Optionally, RPL's periodic operations may be used to transport the network synchronization. This may mean that 6top would need to trigger (override) the trickle timer if no other traffic has occurred for such a time that nodes may get out of synchronization.

A node that has not joined the TSGI advertises a MAC level Join Priority of 0xFF to notify its neighbors that is not capable of serving as time parent. A node that has joined the TSGI advertises a MAC level Join Priority set to its DAGRank() in that Instance, where DAGRank() is the operation specified in section 3.5.1 of [RFC6550], "Rank Comparison".

A root is configured or obtains by some external means the knowledge of the RPLInstanceID for the TSGI. The root advertises its DagRank in the TSGI, that must be less than 0xFF, as its Join Priority (JP) in its IEEE Std 802.15.4 Extended Beacons (EB). We'll note that the JP is now specified between 0 and 0x3F leaving 2 bits in the octet unused in the IEEE Std 802.15.4e specification. After consultation with IEEE authors, it was asserted that 6TiSCH can make a full use of the octet to carry an integer value up to 0xFF.

A node that reads a Join Priority of less than 0xFF should join the neighbor with the lesser Join Priority and use it as time parent. If the node is configured to serve as time parent, then the node should join the TSGI, obtain a Rank in that Instance and start advertising its own DagRank in the TSGI as its Join Priority in its EBs.

4.2.5. SlotFrames and Priorities

6TiSCH enables in essence the capability to use IPv6 over a MAC layer that enables to schedule some of the transmissions. In order to ensure that the medium is free of contending packets when time arrives for a scheduled transmission, a window of time is defined around the scheduled transmission time where the medium must be free of contending energy.

One simple way to obtain such a window is to format time and frequencies in cells of transmission of equal duration. This is the method that is adopted in IEEE Std 802.15.4 TSCH as well as the Long Term Evolution (LTE) of cellular networks.

In order to describe that formatting of time and frequencies, the 6TiSCH architecture defines a global concept that is called a Channel Distribution and Usage (CDU) matrix; a CDU matrix is a matrix of

cells with an height equal to the number of available channels (indexed by ChannelOffsets) and a width (in timeslots) that is the period of the network scheduling operation (indexed by slotOffsets) for that CDU matrix. The size of a cell is a timeslot duration, and values of 10 to 15 milliseconds are typical in 802.15.4 TSCH to accommodate for the transmission of a frame and an ack, including the security validation on the receive side which may take up to a few milliseconds on some device architecture.

A CDU matrix iterates over and over with a pseudo-random rotation from an epoch time. In a given network, there might be multiple CDU matrices that operate with different width, so they have different durations and represent different periodic operations. It is recommended that all CDU matrices in a 6TiSCH domain operate with the same cell duration and are aligned, so as to reduce the chances of interferences from slotted-aloha operations. The knowledge of the CDU matrices is shared between all the nodes and used in particular to define slotFrames.

A slotFrame is a MAC-level abstraction that is common to all nodes and contains a series of timeslots of equal length and precedence. It is characterized by a slotFrame_ID, and a slotFrame_size. A slotFrame aligns to a CDU matrix for its parameters, such as number and duration of timeslots.

Multiple slotFrames can coexist in a node schedule, i.e., a node can have multiple activities scheduled in different slotFrames, based on the precedence of the 6TiSCH topologies. The slotFrames may be aligned to different CDU matrices and thus have different width. There is typically one slotFrame for scheduled traffic that has the highest precedence and one or more slotFrame(s) for RPL traffic. The timeslots in the slotFrame are indexed by the SlotOffset; the first cell is at SlotOffset 0.

When a packet is received from a higher layer for transmission, 6top inserts that packet in the outgoing queue which matches the packet best (Differentiated Services [RFC2474] can therefore be used). At each scheduled transmit slot, 6top looks for the frame in all the outgoing queues that best matches the cells. If a frame is found, it is given to the TSCH MAC for transmission.

4.2.6. Distributing the reservation of cells

6TiSCH expects a high degree of scalability together with a distributed routing functionality based on RPL. To achieve this goal, the spectrum must be allocated in a way that allows for spatial reuse between zones that will not interfere with one another. In a

large and spatially distributed network, a 6TiSCH node is often in a good position to determine usage of spectrum in its vicinity.

Use cases for distributed routing are often associated with a statistical distribution of best-effort traffic with variable needs for bandwidth on each individual link. With 6TiSCH, the abstraction of an IPv6 link is implemented as a pair of bundles of cells, one in each direction; the size of a bundle is optimal when both the energy wasted idle listening and the packet drops due to congestion loss are minimized. This can be maintained if the number of cells in a bundle is adapted dynamically, and with enough reactivity, to match the variations of best-effort traffic. In turn, the agility to fulfill the needs for additional cells improves when the number of interactions with other devices and the protocol latencies are minimized.

6TiSCH limits that interaction to RPL parents that will only negotiate with other RPL parents, and performs that negotiation by groups of cells as opposed to individual cells. The 6TiSCH architecture allows RPL parents to adjust dynamically, and independently from the PCE, the amount of bandwidth that is used to communicate between themselves and their children, in both directions; to that effect, an allocation mechanism enables a RPL parent to obtain the exclusive use of a portion of a CDU matrix within its interference domain. Note that a PCE is expected to have precedence in the allocation, so that a RPL parent would only be able to obtain portions that are not in-use by the PCE.

The 6TiSCH architecture introduces the concept of chunks [I-D.ietf-6tisch-terminology]) to operate such spectrum distribution for a whole group of cells at a time. The CDU matrix is formatted into a set of chunks, each of them identified uniquely by a chunk-ID. The knowledge of this formatting is shared between all the nodes in a 6TiSCH network. 6TiSCH also defines the process of chunk ownership appropriation whereby a RPL parent discovers a chunk that is not used in its interference domain (e.g lack of energy detected in reference cells in that chunk); then claims the chunk, and then defends it in case another RPL parent would attempt to appropriate it while it is in use. The chunk is the basic unit of ownership that is used in that process.

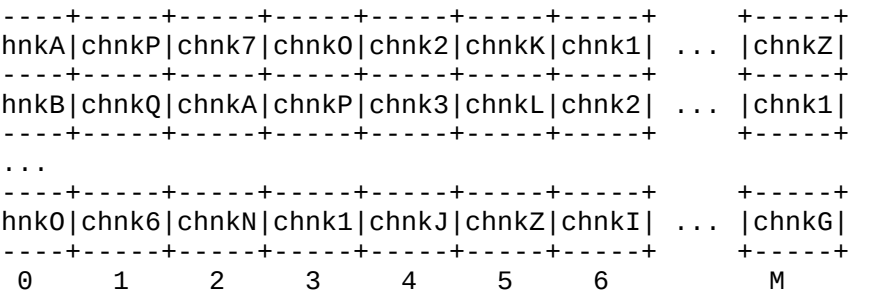


Figure 9: CDU matrix Partitioning in Chunks

As a result of the process of chunk ownership appropriation, the RPL parent has exclusive authority to decide which cell in the appropriated chunk can be used by which node in its interference domain. In other words, it is implicitly delegated the right to manage the portion of the CDU matrix that is represented by the chunk. The RPL parent may thus orchestrate which transmissions occur in any of the cells in the chunk, by allocating cells from the chunk to any form of communication (unicast, multicast) in any direction between itself and its children. Initially, those cells are added to the heap of free cells, then dynamically placed into existing bundles, in new bundles, or allocated opportunistically for one transmission.

The appropriation of a chunk can also be requested explicitly by the PCE to any node. In that case, the node still may need to perform the appropriation process to validate that no other node has claimed that chunk already. After a successful appropriation, the PCE owns the cells in that chunk, and may use them as hard cells to set up Tracks.

4.3. Communication Paradigms and Interaction Models

[I-D.ietf-6tisch-terminology] defines the terms of Communication Paradigms and Interaction Models, which can be placed in parallel to the Information Models and Data Models that are defined in [RFC3444].

A Communication Paradigms would be an abstract view of a protocol exchange, and would come with an Information Model for the information that is being exchanged. In contrast, an Interaction Models would be more refined and could point on standard operation such as a Representational state transfer (REST) "GET" operation and would match a Data Model for the data that is provided over the protocol exchange.

section 2.1.3 of [I-D.ietf-roll-rpl-industrial-applicability] and next sections discuss application-layer paradigms, such as Source-sink (SS) that is a Multipeer to Multipeer (MP2MP) model primarily used for alarms and alerts, Publish-subscribe (PS, or pub/sub) that is typically used for sensor data, as well as Peer-to-peer (P2P) and Peer-to-multipeer (P2MP) communications. Additional considerations on Duocast and its N-cast generalization are also provided. Those paradigms are frequently used in industrial automation, which is a major use case for IEEE Std 802.15.4 TSCH wireless networks with [ISA100.11a] and [WirelessHART], that provides a wireless access to [HART] applications and devices.

This specification focuses on Communication Paradigms and Interaction Models for packet forwarding and TSCH resources (cells) management. Management mechanisms for the TSCH schedule at Link-layer (one-hop), Network-layer (multithop along a Track), and Application-layer (remote control) are discussed in Section 4.4. Link-layer frame forwarding interactions are discussed in Section 4.6, and Network-layer Packet routing is addressed in Section 4.7.

4.4. Schedule Management Mechanisms

6TiSCH uses 4 paradigms to manage the TSCH schedule of the LLN nodes: Static Scheduling, neighbor-to-neighbor Scheduling, remote monitoring and scheduling management, and Hop-by-hop scheduling. Multiple mechanisms are defined that implement the associated Interaction Models, and can be combined and used in the same LLN. Which mechanism(s) to use depends on application requirements.

4.4.1. Static Scheduling

In the simplest instantiation of a 6TiSCH network, a common fixed schedule may be shared by all nodes in the network. Cells are shared, and nodes contend for slot access in a slotted aloha manner.

A static TSCH schedule can be used to bootstrap a network, as an initial phase during implementation, or as a fall-back mechanism in case of network malfunction. This schedule is pre-established, for instance decided by a network administrator based on operational needs. It can be pre-configured into the nodes, or, more commonly, learned by a node when joining the network using standard IEEE Std 802.15.4 Information Elements (IE). Regardless, the schedule remains unchanged after the node has joined a network. RPL is used on the resulting network. This "minimal" scheduling mechanism that implements this paradigm is detailed in [RFC8180].

4.4.2. Neighbor-to-neighbor Scheduling

In the simplest instantiation of a 6TiSCH network described in Section 4.4.1, nodes may expect a packet at any cell in the schedule and will waste energy idle listening. In a more complex instantiation of a 6TiSCH network, a matching portion of the schedule is established between peers to reflect the observed amount of transmissions between those nodes. The aggregation of the cells between a node and a peer forms a bundle that the 6top layer uses to implement the abstraction of a link for IP. The bandwidth on that link is proportional to the number of cells in the bundle.

If the size of a bundle is configured to fit an average amount of bandwidth, peak traffic is dropped. If the size is configured to allow for peak emissions, energy is be wasted idle listening.

The 6top Protocol [I-D.ietf-6tisch-6top-protocol] specifies the exchanges between neighbor nodes to reserve soft cells to transmit to one another. Because this reservation is done without global knowledge of the schedule of other nodes in the LLN, scheduling collisions are possible. An optional Scheduling Function (SF) such as MSF [I-D.chang-6tisch-msf] is used to monitor bandwidth usage and perform requests for dynamic allocation by the 6top sublayer. The SF component is not part of the 6top sublayer. It may be collocated on the same device or may be partially or fully offloaded to an external system.

Monitoring and relocation is done in the 6top layer. For the upper layer, the connection between two neighbor nodes appears as an number of cells. Depending on traffic requirements, the upper layer can request 6top to add or delete a number of cells scheduled to a particular neighbor, without being responsible for choosing the exact slotOffset/channelOffset of those cells.

4.4.3. Remote Monitoring and Schedule Management

The work at the 6TiSCH WG is focused on non-deterministic traffic and does not provide the generic data model that would be necessary to monitor and manage resources of the 6top sublayer. It is recognized that CoAP can be appropriate to interact with the 6top layer of a node that is multiple hops away across a 6TiSCH mesh.

The entity issuing the CoAP requests can be a central scheduling entity (e.g. a PCE), a node multiple hops away with the authority to modify the TSCH schedule (e.g. the head of a local cluster), or a external device monitoring the overall state of the network (e.g. NME). It is also possible that a mapping entity on the backbone

transforms a non-CoAP protocol such as PCEP into the RESTful interfaces that the 6TiSCH devices support.

With respect to Centralized routing and scheduling, it is envisionned that the related component of the 6TiSCH Architecture would be an extension of the Deterministic Networking Architecture [I-D.ietf-detnet-architecture], which studies Layer-3 aspects of Deterministic Networks, and covers networks that span multiple Layer-2 domains. The DetNet architecture is a form of SDN Architecture and is composed of three planes, a (User) Application Plane, a Controller Plane (where the PCE operates), and a Network Plane which in our case is the 6TiSCH LLN. The generic SDN architecture is discussed in Software-Defined Networking (SDN): Layers and Architecture Terminology [RFC7426] and is represented below:

SDN Layers and Architecture Terminology per RFC 7426

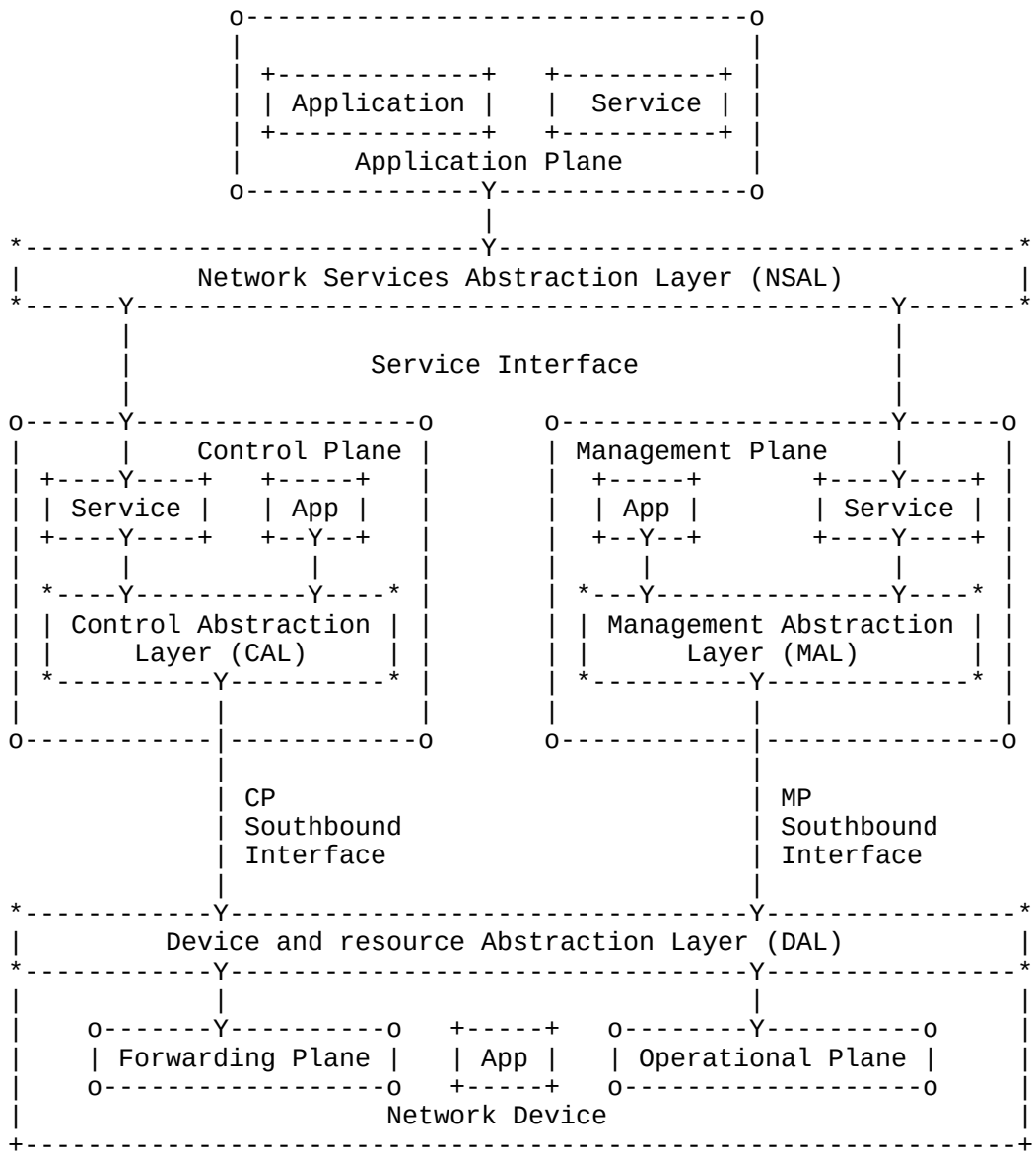


Figure 10

The PCE establishes end-to-end Tracks of hard cells, which are described in more details in Section 4.6.1. The DetNet work is expected to enable end to end Deterministic Path across heterogeneous

network (e.g. a 6TiSCH LLN and an Ethernet Backbone). This model fits the 6TiSCH extended configuration, whereby a 6BBR federates multiple 6TiSCH LLN in a single subnet over a backbone that can be, for instance, Ethernet or Wi-Fi. In that model, 6TiSCH 6BBRs synchronize with one another over the backbone, so as to ensure that the multiple LLNs that form the IPv6 subnet stay tightly synchronized.

If the Backbone is Deterministic, then the Backbone Router ensures that the end-to-end deterministic behavior is maintained between the LLN and the backbone. It is the responsibility of the PCE to compute a deterministic path and to end across the TSCH network and an IEEE Std 802.1 TSN Ethernet backbone, and that of DetNet to enable end-to-end deterministic forwarding.

4.4.4. Hop-by-hop Scheduling

A node can reserve a Track (Section 4.5) to a destination node multiple hops away by installing soft cells at each intermediate node. This forms a Track of soft cells. It is the responsibility of the 6top sublayer of each node on the Track to monitor these soft cells and trigger relocation when needed.

This hop-by-hop reservation mechanism is expected to be similar in essence to [RFC3209] and/or [RFC4080]/[RFC5974]. The protocol for a node to trigger hop-by-hop scheduling is not yet defined.

4.5. On Tracks

4.5.1. General Behavior of Tracks

The architecture introduces the concept of a Track, which is a directed path from a source 6TiSCH node to a destination 6TiSCH node across a 6TiSCH LLN. A Track is the 6TiSCH instantiation of the concept of a Deterministic Path as described in [I-D.ietf-detnet-architecture]. Constrained resources such as memory buffers are reserved for that Track in intermediate 6TiSCH nodes to avoid loss related to limited capacity. A 6TiSCH node along a Track not only knows which bundles of cells it should use to receive packets from a previous hop, but also knows which bundle(s) it should use to send packets to its next hop along the Track.

A Track is composed of bundles of cells with related schedules and logical relationships and that ensure that a packet that is injected in a Track will progress in due time all the way to destination. Multiple cells may be scheduled in a Track for the transmission of a single packet, in which case the normal operation of IEEE Std 802.15.4 Automatic Repeat-reQuest (ARQ) can take place; the

acknowledgment may be omitted in some cases, for instance if there is no scheduled cell for a possible retry.

There are several benefits for using a Track to forward a packet from a source node to the destination node.

1. Track forwarding, as further described in Section 4.6.1, is a Layer-2 forwarding scheme, which introduces less process delay and overhead than Layer-3 forwarding scheme. Therefore, LLN Devices can save more energy and resource, which is critical for resource constrained devices.

2. Since channel resources, i.e. bundles of cells, have been reserved for communications between 6TiSCH nodes of each hop on the Track, the throughput and the maximum latency of the traffic along a Track are guaranteed and the jitter is maintained small.

3. By knowing the scheduled time slots of incoming bundle(s) and outgoing bundle(s), 6TiSCH nodes on a Track could save more energy by staying in sleep state during in-active slots.

4. Tracks are protected from interfering with one another if a cell belongs to at most one Track, and congestion loss is avoided if at most one packet can be presented to the MAC to use that cell. Tracks enhance the reliability of transmissions and thus further improve the energy consumption in LLN Devices by reducing the chances of retransmission.

4.5.2. Serial Track

A Serial (or simple) Track is the 6TiSCH version of a circuit; a bundle of cells that are programmed to receive (RX-cells) is uniquely paired to a bundle of cells that are set to transmit (TX-cells), representing a Layer-2 forwarding state which can be used regardless of the network layer protocol.

A Serial Track is thus formed end-to-end as a succession of paired bundles, a receive bundle from the previous hop and a transmit bundle to the next hop along the Track. For a given iteration of the device schedule, the effective channel of the cell is obtained by adding a pseudo-random number to the channelOffset of the cell, which results in a rotation of the frequency that used for transmission.

The bundles may be computed so as to accommodate both variable rates and retransmissions, so they might not be fully used at a given iteration of the schedule.

4.5.3. Complex Track with Replication and Elimination

As opposed to a Serial Track that is a sequence of nodes and links, a Complex Track is shaped as a directed acyclic graph towards a destination to support multi-path forwarding and route around failures.

A Complex Track may also branch off and rejoin, for the purpose of the DetNet Packet Replication and Elimination (PRE), over non congruent branches. PRE may be used to complement Layer-2 ARQ to meet industrial expectations in Packet Delivery Ratio (PDR), in particular when the Track extends beyond the 6TiSCH network in a larger DetNet network.

The art of Deterministic Networks already include PRE techniques. Example standards include the Parallel Redundancy Protocol (PRP) and the High-availability Seamless Redundancy (HSR) [IEC62439].

At each 6TiSCH hop along the Track, the PCE may schedule more than one timeslot for a packet, so as to support Layer-2 retries (ARQ). It is also possible that the field device only uses the second branch if sending over the first branch fails.

In the art of TSCH, a path does not necessarily support PRE but it is almost systematically multi-path. This means that a Track is scheduled so as to ensure that each hop has at least two forwarding solutions, and the forwarding decision is to try the preferred one and use the other in case of Layer-2 transmission failure as detected by ARQ.

4.5.4. DetNet End-to-end Path

Ultimately, DetNet should enable to extend a Track beyond the 6TiSCH LLN. Figure 11 illustrates a Track that is laid out from a field device in a 6TiSCH network to an IoT gateway that is located on an 802.1 Time-Sensitive Networking (TSN) backbone.

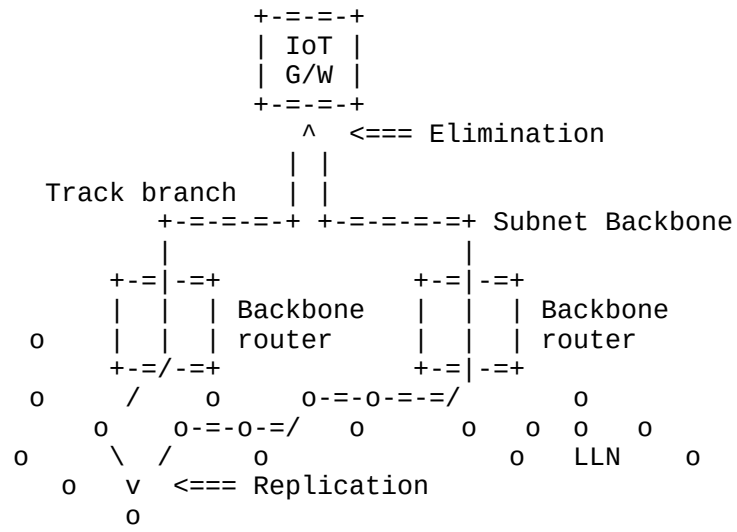


Figure 11: End-to-End deterministic Track

The Replication function in the 6TiSCH Node sends a copy of each packet over two different branches, and the PCE schedules each hop of both branches so that the two copies arrive in due time at the gateway. In case of a loss on one branch, hopefully the other copy of the packet still makes it in due time. If two copies make it to the IoT gateway, the Elimination function in the gateway ignores the extra packet and presents only one copy to upper layers.

4.5.5. Cell Reuse

The 6TiSCH architecture provides means to avoid waste of cells as well as overflows in the transmit bundle pof a Track, as follows:

In one hand, a TX-cell that is not needed for the current iteration may be reused opportunistically on a per-hop basis for routed packets. When all of the frame that were received for a given Track are effectively transmitted, any available TX-cell for that Track can be reused for upper layer traffic for which the next-hop router matches the next hop along the Track. In that case, the cell that is being used is effectively a TX-cell from the Track, but the short address for the destination is that of the next-hop router. It results that a frame that is received in a RX-cell of a Track with a destination MAC address set to this node as opposed to broadcast must be extracted from the Track and delivered to the upper layer (a frame with an unrecognized

destination MAC address is dropped at the lower MAC layer and thus is not received at the 6top sublayer).

On the other hand, it might happen that there are not enough TX-cells in the transmit bundle to accommodate the Track traffic, for instance if more retransmissions are needed than provisioned. In that case, the frame can be placed for transmission in the bundle that is used for Layer-3 traffic towards the next hop along the Track as long as it can be routed by the upper layer, that is, typically, if the frame transports an IPv6 packet. The MAC address should be set to the next-hop MAC address to avoid confusion. It results that a frame that is received over a Layer-3 bundle may be in fact associated to a Track. In a classical IP link such as an Ethernet, off-Track traffic is typically in excess over reservation to be routed along the non-reserved path based on its QoS setting. But with 6TiSCH, since the use of the Layer-3 bundle may be due to transmission failures, it makes sense for the receiver to recognize a frame that should be re-Tracked, and to place it back on the appropriate bundle if possible. A frame should be re-Tracked if the Per-Hop-Behavior group indicated in the Differentiated Services Field of the IPv6 header is set to Deterministic Forwarding, as discussed in Section 4.7.1. A frame is re-Tracked by scheduling it for transmission over the transmit bundle associated to the Track, with the destination MAC address set to broadcast.

4.6. Forwarding Models

By forwarding, this specification means the per-packet operation that allows to deliver a packet to a next hop or an upper layer in this node. Forwarding is based on pre-existing state that was installed as a result of a routing computation Section 4.7. 6TiSCH supports three different forwarding model, G-MPLS Track Forwarding (TF), 6LoWPAN Fragment Forwarding (FF) and IPv6 Forwarding (6F).

4.6.1. Track Forwarding

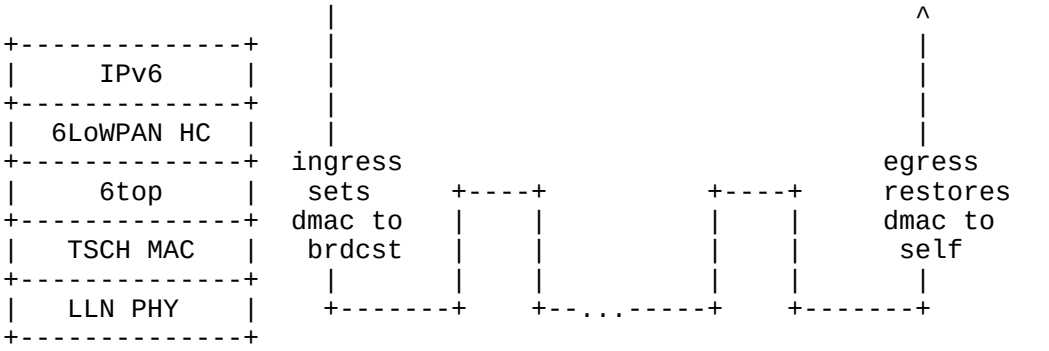
Forwarding along a Track can be seen as a Generalized Multi-protocol Label Switching (G-MPLS) operation in that the information used to switch a frame is not an explicit label, but rather related to other properties of the way the packet was received, a particular cell in the case of 6TiSCH. As a result, as long as the TSCH MAC (and Layer-2 security) accepts a frame, that frame can be switched regardless of the protocol, whether this is an IPv6 packet, a 6LoWPAN fragment, or a frame from an alternate protocol such as WirelessHART or ISA100.11a.

A data frame that is forwarded along a Track normally has a destination MAC address that is set to broadcast - or a multicast address depending on MAC support. This way, the MAC layer in the intermediate nodes accepts the incoming frame and 6top switches it without incurring a change in the MAC header. In the case of IEEE Std 802.15.4, this means effectively broadcast, so that along the Track the short address for the destination of the frame is set to 0xFFFF.

There are 2 modes for a Track, transport mode and tunnel mode.

4.6.1.1. Transport Mode

In transport mode, the Protocol Data Unit (PDU) is associated with flow-dependant meta-data that refers uniquely to the Track, so the 6top sublayer can place the frame in the appropriate cell without ambiguity. In the case of IPv6 traffic, this flow identification is transported in the Flow Label of the IPv6 header. Associated with the source IPv6 address, the Flow Label forms a globally unique identifier for that particular Track that is validated at egress before restoring the destination MAC address (DMAC) and punting to the upper layer.



Track Forwarding, Transport Mode

4.6.1.2. Tunnel Mode

In tunnel mode, the frames originate from an arbitrary protocol over a compatible MAC that may or may not be synchronized with the 6TiSCH network. An example of this would be a router with a dual radio that is capable of receiving and sending WirelessHART or ISA100.11a frames with the second radio, by presenting itself as an access Point or a Backbone Router, respectively.

In that mode, some entity (e.g. PCE) can coordinate with a WirelessHART Network Manager or an ISA100.11a System Manager to specify the flows that are to be transported transparently over the Track.

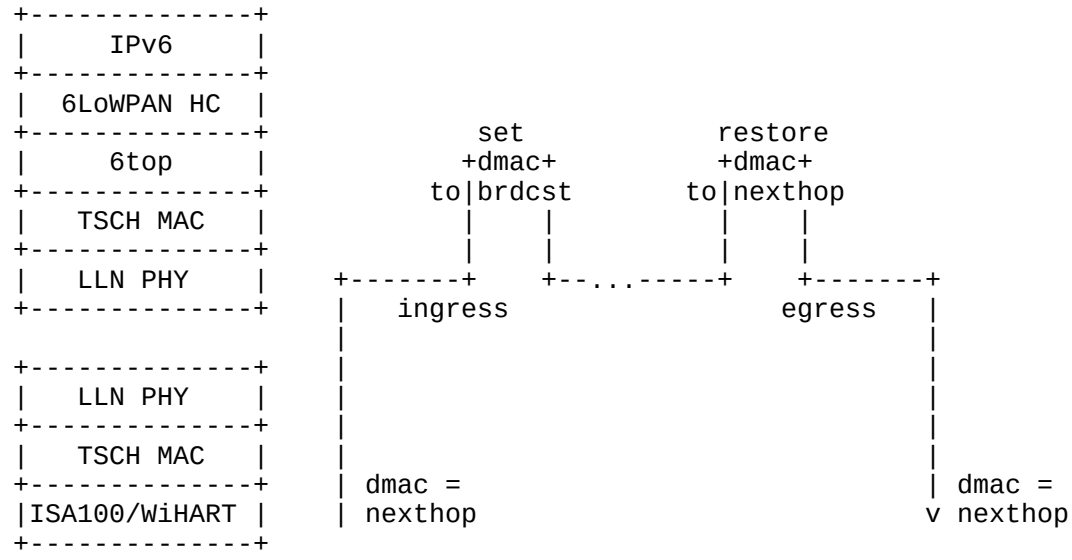


Figure 12: Track Forwarding, Tunnel Mode

In that case, the flow information that identifies the Track at the ingress 6TiSCH router is derived from the RX-cell. The dmac is set to this node but the flow information indicates that the frame must be tunneled over a particular Track so the frame is not passed to the upper layer. Instead, the dmac is forced to broadcast and the frame is passed to the 6top sublayer for switching.

At the egress 6TiSCH router, the reverse operation occurs. Based on metadata associated to the Track, the frame is passed to the appropriate link layer with the destination MAC restored.

4.6.1.3. Tunnel Metadata

Metadata coming with the Track configuration is expected to provide the destination MAC address of the egress endpoint as well as the tunnel mode and specific data depending on the mode, for instance a service access point for frame delivery at egress. If the tunnel egress point does not have a MAC address that matches the configuration, the Track installation fails.

In transport mode, if the final Layer-3 destination is the tunnel termination, then it is possible that the IPv6 address of the destination is compressed at the 6LoWPAN sublayer based on the MAC address. It is thus mandatory at the ingress point to validate that the MAC address that was used at the 6LoWPAN sublayer for compression matches that of the tunnel egress point. For that reason, the node that injects a packet on a Track checks that the destination is effectively that of the tunnel egress point before it overwrites it to broadcast. The 6top sublayer at the tunnel egress point reverts that operation to the MAC address obtained from the tunnel metadata.

4.6.2. Fragment Forwarding

Considering that 6LoWPAN packets can be as large as 1280 bytes (the IPv6 MTU), and that the non-storing mode of RPL implies Source Routing that requires space for routing headers, and that a IEEE Std 802.15.4 frame with security may carry in the order of 80 bytes of effective payload, an IPv6 packet might be fragmented into more than 16 fragments at the 6LoWPAN sublayer.

This level of fragmentation is much higher than that traditionally experienced over the Internet with IPv4 fragments, where fragmentation is already known as harmful.

In the case to a multihop route within a 6TiSCH network, Hop-by-Hop recomposition occurs at each hop in order to reform the packet and route it. This creates additional latency and forces intermediate nodes to store a portion of a packet for an undetermined time, thus impacting critical resources such as memory and battery.

[I-D.watteyne-6lo-minimal-fragment] describes a framework for forwarding fragments end-to-end across a 6TiSCH route-over mesh. Within that framework, [I-D.bormann-lwig-6lowpan-virtual-reassembly] details a virtual reassembly buffer mechanism whereby the datagram tag in the 6LoWPAN Fragment is used as a label for switching at the 6LoWPAN sublayer. Building on this technique, [I-D.thubert-6lo-fragment-recovery] introduces a new format for 6LoWPAN fragments that enables the selective recovery of individual fragments, and allows for a degree of flow control based on an Explicit Congestion Notification.

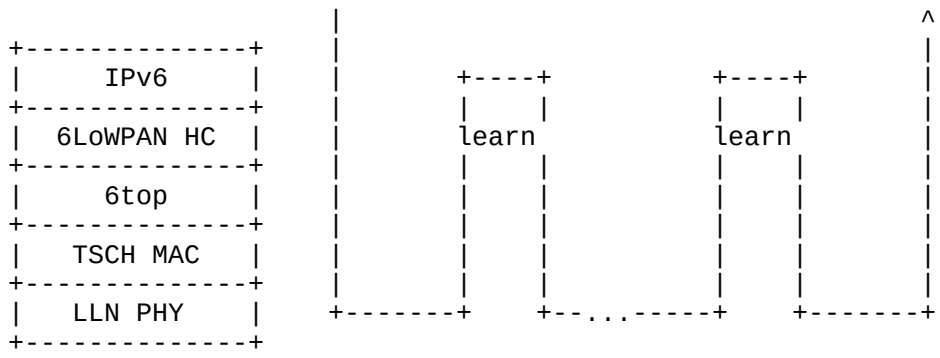


Figure 13: Forwarding First Fragment

In that model, the first fragment is routed based on the IPv6 header that is present in that fragment. The 6LoWPAN sublayer learns the next hop selection, generates a new datagram tag for transmission to the next hop, and stores that information indexed by the incoming MAC address and datagram tag. The next fragments are then switched based on that stored state.

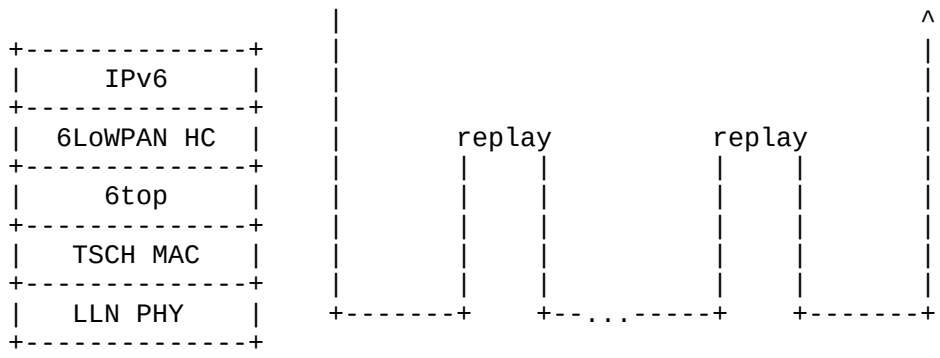


Figure 14: Forwarding Next Fragment

A bitmap and an ECN echo in the end-to-end acknowledgment enable the source to resend the missing fragments selectively. The first fragment may be resent to carve a new path in case of a path failure. The ECN echo set indicates that the number of outstanding fragments should be reduced.

4.6.3. IPv6 Forwarding

As the packets are routed at Layer-3, traditional QoS and Active Queue Management (AQM) operations are expected to prioritize flows;

the application of Differentiated Services is further discussed in [I-D.svshah-tsvwg-lln-diffserv-recommendations].

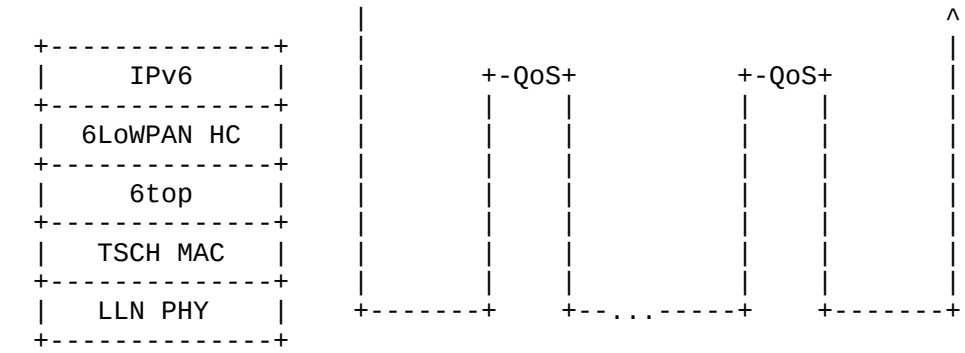


Figure 15: IP Forwarding

4.7. Centralized vs. Distributed Routing

6TiSCH supports a mixed model of centralized routes and distributed routes. Centralized routes can for example be computed by a entity such as a PCE. Distributed routes are computed by RPL.

Both methods may inject routes in the Routing Tables of the 6TiSCH routers. In either case, each route is associated with a 6TiSCH topology that can be a RPL Instance topology or a Track. The 6TiSCH topology is indexed by a Instance ID, in a format that reuses the RPLInstanceID as defined in RPL [RFC6550].

Both RPL and PCE rely on shared sources such as policies to define Global and Local RPLInstanceIDs that can be used by either method. It is possible for centralized and distributed routing to share a same topology. Generally they will operate in different slotFrames, and centralized routes will be used for scheduled traffic and will have precedence over distributed routes in case of conflict between the slotFrames.

4.7.1. Packet Marking and Handling

All packets inside a 6TiSCH domain must carry the Instance ID that identifies the 6TiSCH topology that is to be used for routing and forwarding that packet. The location of that information must be the same for all packets forwarded inside the domain.

For packets that are routed by a PCE along a Track, the tuple formed by the IPv6 source address and a local RPLInstanceID in the packet identify uniquely the Track and associated transmit bundle.

For packets that are routed by RPL, that information is the RPLInstanceID which is carried in the RPL Packet Information, as discussed in section 11.2 of [RFC6550], "Loop Avoidance and Detection".

The RPL Packet Information (RPI) is carried in IPv6 packets as a RPL option in the IPv6 Hop-By-Hop Header [RFC6553].

A compression mechanism for the RPL packet artifacts that integrates the compression of IP-in-IP encapsulation and the Routing Header type 3 [RFC6554] with that of the RPI in a 6LoWPAN dispatch/header type is specified in [RFC8025] and [RFC8138].

Either way, the method and format used for encoding the RPLInstanceID is generalized to all 6TiSCH topological Instances, which include both RPL Instances and Tracks.

4.7.2. Replication, Retries and Elimination

6TiSCH expects elimination and replication of packets along a complex Track, but has no position about how the sequence numbers would be tagged in the packet.

As it goes, 6TiSCH expects that timeSlots corresponding to copies of a same packet along a Track are correlated by configuration, and does not need to process the sequence numbers.

The semantics of the configuration will enable correlated timeSlots to be grouped for transmit (and respectively receive) with a 'OR' relations, and then a 'AND' relation would be configurable between groups. The semantics is that if the transmit (and respectively receive) operation succeeded in one timeSlot in a 'OR' group, then all the other timeSLots in the group are ignored. Now, if there are at least two groups, the 'AND' relation between the groups indicates that one operation must succeed in each of the groups.

On the transmit side, timeSlots provisioned for retries along a same branch of a Track are placed a same 'OR' group. The 'OR' relation indicates that if a transmission is acknowledged, then further transmissions should not be attempted for timeSlots in that group. There are as many 'OR' groups as there are branches of the Track departing from this node. Different 'OR' groups are programmed for the purpose of replication, each group corresponding to one branch of the Track. The 'AND' relation between the groups indicates that transmission over any of branches must be attempted regardless of whether a transmission succeeded in another branch. It is also possible to place cells to different next-hop routers in a same 'OR'

group. This allows to route along multi-path tracks, trying one next-hop and then another only if sending to the first fails.

On the receive side, all timeSlots are programmed in a same 'OR' group. Retries of a same copy as well as converging branches for elimination are converged, meaning that the first successful reception is enough and that all the other timeSlots can be ignored.

4.7.3. Differentiated Services Per-Hop-Behavior

Additionally, an IP packet that is sent along a Track uses the Differentiated Services Per-Hop-Behavior Group called Deterministic Forwarding, as described in [I-D.svshah-tsvwg-deterministic-forwarding].

5. IANA Considerations

This specification does not require IANA action.

6. Security Considerations

This architecture operates on IEEE Std 802.15.4 and expects link-layer security to be enabled at all times between connected devices, except for the very first step of the device join process, where a joining device may need some initial, unsecured exchanges so as to obtain its initial key material.

The Minimal Security Framework for 6TiSCH [I-D.ietf-6tisch-minimal-security] describes the minimal mechanisms required to support secure enrollment of a pledge to a 6TiSCH network based on PSK. The specification enables to establish of link-layer keys, typically used in combination with a variation of Counter with CBC-MAC (CCM) [RFC3610], and set up a secure end-to-end session between the joining node (called the pledge) and the join registrar/ coordinator (JRC) in charge of authenticating the node via a Join Proxy (JP). It can also be used to obtain a link layer short address as a side effect. The 6TiSCH Joined Process (6JP) used shared slots which are a constrained resource, so it is optimized to limit the number of messages to the strict minimum. As an example, Neighbor Discovery between the pledge and the JP can be skipped when the IPv6 Link Local addresses that are used derive from the node's EUI-64 address.

The "6tisch Zero-Touch Secure Join protocol" [I-D.ietf-6tisch-dtsecurity-zerotouch-join] wraps the minimal security draft with a flow inspired from ANIMA "Bootstrapping Remote Secure Key Infrastructures (BRSKI)" [I-D.ietf-anima-bootstrapping-keyinfra].

6.1. Join Process Highlights

The BRSKI architecture specifies three logical elements to describe the join process:

Pledge: Node that wishes to become part of the network;

Join Registrar/Coordinator (JRC) : An entity that arbitrates network access and hands out network parameters (such as keying material);

Join Proxy (JP), a one-hop (radio) neighbor of the joining node that acts as proxy network node and may provide connectivity with the JRC.

The join protocol consists of three major activities:

Device Authentication: The Pledge and the JP mutually authenticate each other and establish a shared key, so as to ensure on-going authenticated communications. This may involve a server as a third party.

Authorization: The JP decides on whether/how to authorize a Pledge (if denied, this may result in loss of bandwidth). Conversely, the Pledge decides on whether/how to authorize the network (if denied, it will not join the network). Authorization decisions may involve other nodes in the network.

Configuration/Parameterization: The JP distributes configuration information to the Pledge, such as scheduling information, IP address assignment information, and network policies. This may originate from other network devices, for which the JP may act as proxy. This step may also include distribution of information from the Pledge to the JP and other nodes in the network and, more generally, synchronization of information between these entities.

The device joining process is depicted in Figure 16, where it is assumed that devices have access to certificates and where entities have access to the root CA keys of their communicating parties (initial set-up requirement). Under these assumptions, the authentication step of the device joining process does not require online involvement of a third party. Mutual authentication is performed between the Pledge and the JP using their certificates, which also results in a shared key between these two entities.

The JP assists the Pledge in mutual authentication with a remote server node (primarily via provision of a communication path with the

server), which also results in a shared (end-to-end) key between those two entities. The server node may be a JRC that arbitrages the network authorization of the Pledge (where the JP will deny bandwidth if authorization is not successful); it may distribute network-specific configuration parameters (including network-wide keys) to the Pledge. In its turn, the Pledge may distribute and synchronize information (including, e.g., network statistics) to the server node and, if so desired, also to the JP. The actual decision of the Pledge to become part of the network may depend on authorization of the network itself.

The server functionality is a role which may be implemented with one (centralized) or multiple devices (distributed). In either case, mutual authentication is established with each physical server entity with which a role is implemented.

Note that in the above description, the JP does not solely act as a relay node, thereby allowing it to first filter traffic to be relayed based on cryptographic authentication criteria - this provides first-level access control and mitigates certain types of denial-of-service attacks on the network at large.

Depending on more detailed insight in cost/benefit trade-offs, this process might be complemented by a more "relaxed" mechanism, where the JP acts as a relay node only. The final architecture will provide mechanisms to also cover cases where the initial set-up requirements are not met or where some other out-of-sync behavior occurs; it will also suggest some optimizations in case JRC-related information is already available with the JP (via caching of information).

When a device rejoins the network in the same authorization domain, the authorization step could be omitted if the server distributes the authorization state for the device to the JP when the device initially joined the network. However, this generally still requires the exchange of updated configuration information, e.g., related to time schedules and bandwidth allocation.

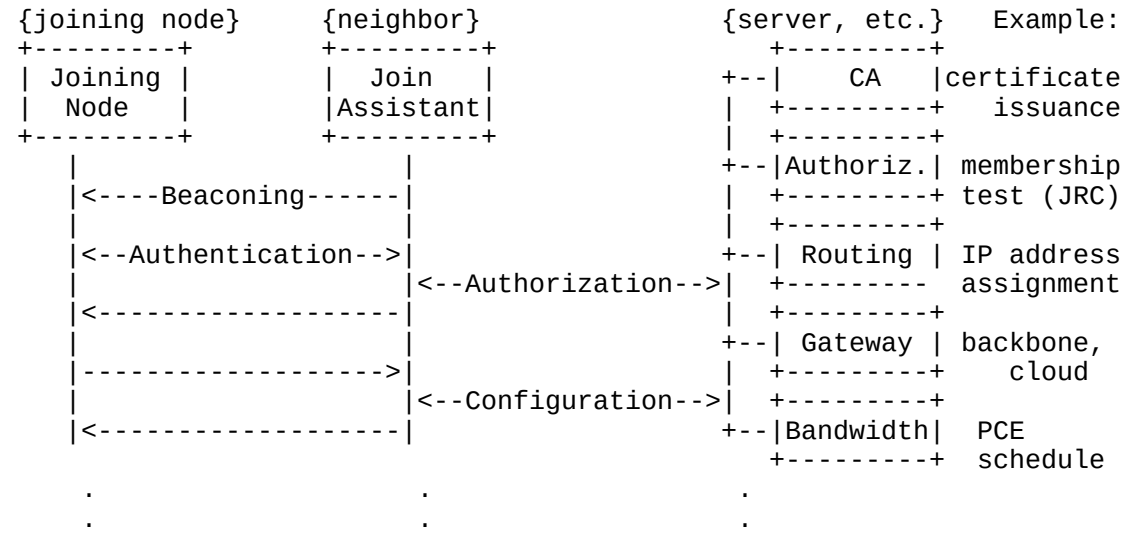


Figure 16: Network joining, with only authorization by third party

7. Acknowledgments

7.1. Contributors

The co-authors of this document are listed below:

Robert Assimiti for his breakthrough work on RPL over TSCH and initial text and guidance;

Kris Pister for creating it all and his continuing guidance through the elaboration of this design;

Michael Richardson for his leadership role in the Security Design Team and his contribution throughout this document;

Rene Struik for the security section and his contribution to the Security Design Team;

Malisa Vucinic for the work on the one-touch join process and his contribution to the Security Design Team;

Xavier Vilajosana who lead the design of the minimal support with RPL and contributed deeply to the 6top design and the G-MPLS operation of Track switching;

Qin Wang who lead the design of the 6top sublayer and contributed related text that was moved and/or adapted in this document;

Thomas Watteyne for his contribution to the whole design, in particular on TSCH and security.

7.2. Special Thanks

Special thanks to Tero Kivinen, Jonathan Simon, Giuseppe Piro, Subir Das and Yoshihiro Ohba for their deep contribution to the initial security work, to Diego Dujovne for starting and leading the SF0 effort and to Tengfei Chang for evolving it in the MSF.

Special thanks also to Pat Kinney for his support in maintaining the connection active and the design in line with work happening at IEEE Std 802.15.4.

Special thanks to Ted Lemon who was the INT Area A-D while this specification was developed for his great support and help throughout.

Also special thanks to Ralph Droms who performed the first INT Area Directorate review, that was very deep and through and radically changed the orientations of this document.

7.3. And Do not Forget

This specification is the result of multiple interactions, in particular during the 6TiSCH (bi)Weekly Interim call, relayed through the 6TiSCH mailing list at the IETF.

The authors wish to thank: Alaeddine Weslati, Chonggang Wang, Georgios Exarchakos, Zhuo Chen, Alfredo Grieco, Bert Greevenbosch, Cedric Adjih, Deji Chen, Martin Turon, Dominique Barthel, Elvis Vogli, Geraldine Texier, Malisa Vucinic, Guillaume Gaillard, Herman Storey, Kazushi Muraoka, Ken Bannister, Kuor Hsin Chang, Laurent Toutain, Maik Seewald, Maria Rita Palattella, Michael Behringer, Nancy Cam Winget, Nicola Accettura, Nicolas Montavont, Oleg Hahm, Patrick Wetterwald, Paul Duffy, Peter van der Stock, Rahul Sen, Pieter de Mil, Pouria Zand, Rouhollah Nabati, Rafa Marin-Lopez, Raghuram Sudhaakar, Sedat Gormus, Shitanshu Shah, Steve Simlo, Tengfei Chang, Tina Tsou, Tom Phinney, Xavier Lagrange, Ines Robles and Samita Chakrabarti for their participation and various contributions.

8. References

8.1. Normative References

[I-D.ietf-6tisch-terminology]
Palattella, M., Thubert, P., Watteyne, T., and Q. Wang, "Terms Used in IPv6 over the TSCH mode of IEEE 802.15.4e", draft-ietf-6tisch-terminology-10 (work in progress), March 2018.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

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

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <https://www.rfc-editor.org/info/rfc4861>.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <https://www.rfc-editor.org/info/rfc4862>.

[RFC6282] Hui, J., Ed. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, DOI 10.17487/RFC6282, September 2011, <https://www.rfc-editor.org/info/rfc6282>.

[RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, DOI 10.17487/RFC6550, March 2012, <https://www.rfc-editor.org/info/rfc6550>.

[RFC6552] Thubert, P., Ed., "Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6552, DOI 10.17487/RFC6552, March 2012, <https://www.rfc-editor.org/info/rfc6552>.

[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, <https://www.rfc-editor.org/info/rfc6553>.

[RFC6554] Hui, J., Vasseur, JP., Culler, D., and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6554, DOI 10.17487/RFC6554, March 2012, <https://www.rfc-editor.org/info/rfc6554>.

[RFC6606] Kim, E., Kaspar, D., Gomez, C., and C. Bormann, "Problem Statement and Requirements for IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Routing", RFC 6606, DOI 10.17487/RFC6606, May 2012, <https://www.rfc-editor.org/info/rfc6606>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <https://www.rfc-editor.org/info/rfc6775>.

[RFC7102] Vasseur, JP., "Terms Used in Routing for Low-Power and Lossy Networks", RFC 7102, DOI 10.17487/RFC7102, January 2014, <https://www.rfc-editor.org/info/rfc7102>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <https://www.rfc-editor.org/info/rfc7228>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[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, <https://www.rfc-editor.org/info/rfc7554>.

[RFC8025] Thubert, P., Ed. and R. Cragie, "IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Paging Dispatch", RFC 8025, DOI 10.17487/RFC8025, November 2016, <https://www.rfc-editor.org/info/rfc8025>.

[RFC8138] Thubert, P., Ed., Bormann, C., Toutain, L., and R. Cragie, "IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Routing Header", RFC 8138, DOI 10.17487/RFC8138, April 2017, <https://www.rfc-editor.org/info/rfc8138>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8180] Vilajosana, X., Ed., Pister, K., and T. Watteyne, "Minimal IPv6 over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration", BCP 210, RFC 8180, DOI 10.17487/RFC8180, May 2017, <https://www.rfc-editor.org/info/rfc8180>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

8.2. Informative References

[I-D.bormann-lwig-6lowpan-virtual-reassembly]
Bormann, C., "Virtual reassembly buffers in 6LoWPAN", draft-bormann-lwig-6lowpan-virtual-reassembly-00 (work in progress), January 2018.

[I-D.chang-6tisch-msf]
Chang, T., Vucinic, M., and X. Vilajosana, "6TiSCH Minimal Scheduling Function (MSF)", draft-chang-6tisch-msf-01 (work in progress), March 2018.

[I-D.ietf-6lo-ap-nd]
Thubert, P., Sarikaya, B., and M. Sethi, "Address Protected Neighbor Discovery for Low-power and Lossy Networks", draft-ietf-6lo-ap-nd-06 (work in progress), February 2018.

[I-D.ietf-6lo-backbone-router]
Thubert, P., "IPv6 Backbone Router", draft-ietf-6lo-backbone-router-06 (work in progress), February 2018.

[I-D.ietf-6lo-rfc6775-update]
Thubert, P., Nordmark, E., Chakrabarti, S., and C. Perkins, "Registration Extensions for 6LoWPAN Neighbor Discovery", draft-ietf-6lo-rfc6775-update-19 (work in progress), April 2018.

[I-D.ietf-6tisch-6top-protocol]
Wang, Q., Vilajosana, X., and T. Watteyne, "6top Protocol (6P)", draft-ietf-6tisch-6top-protocol-11 (work in progress), March 2018.

[I-D.ietf-6tisch-dtsecurity-zerotouch-join]
Richardson, M. and B. Damm, "6tisch Zero-Touch Secure Join protocol", draft-ietf-6tisch-dtsecurity-zerotouch-join-01 (work in progress), October 2017.

[I-D.ietf-6tisch-minimal-security]
Vucinic, M., Simon, J., Pister, K., and M. Richardson, "Minimal Security Framework for 6TiSCH", draft-ietf-6tisch-minimal-security-05 (work in progress), March 2018.

[I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., Bjarnason, S., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-14 (work in progress), April 2018.

[I-D.ietf-core-comi]
Veillette, M., Stok, P., Pelov, A., and A. Bierman, "CoAP Management Interface", draft-ietf-core-comi-02 (work in progress), December 2017.

[I-D.ietf-core-object-security]
Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)", draft-ietf-core-object-security-12 (work in progress), March 2018.

[I-D.ietf-detnet-architecture]
Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", draft-ietf-detnet-architecture-04 (work in progress), October 2017.

[I-D.ietf-detnet-use-cases]
Grossman, E., "Deterministic Networking Use Cases", draft-ietf-detnet-use-cases-15 (work in progress), April 2018.

[I-D.ietf-manet-aodvv2]
Perkins, C., Ratliff, S., Dowdell, J., Steenbrink, L., and V. Mercieca, "Ad Hoc On-demand Distance Vector Version 2 (AODVv2) Routing", draft-ietf-manet-aodvv2-16 (work in progress), May 2016.

[I-D.ietf-roll-aodv-rpl]
Anamalamudi, S., Zhang, M., Sangi, A., Perkins, C., Anand, S., and B. Liu, "Asymmetric AODV-P2P-RPL in Low-Power and Lossy Networks (LLNs)", draft-ietf-roll-aodv-rpl-03 (work in progress), March 2018.

[I-D.ietf-roll-rpl-industrial-applicability]
Phinney, T., Thubert, P., and R. Assimiti, "RPL applicability in industrial networks", draft-ietf-roll-rpl-industrial-applicability-02 (work in progress), October 2013.

[I-D.svshah-tsvwg-deterministic-forwarding]
Shah, S. and P. Thubert, "Deterministic Forwarding PHB", draft-svshah-tsvwg-deterministic-forwarding-04 (work in progress), August 2015.

[I-D.svshah-tsvwg-lln-diffserv-recommendations]
Shah, S. and P. Thubert, "Differentiated Service Class Recommendations for LLN Traffic", draft-svshah-tsvwg-lln-diffserv-recommendations-04 (work in progress), February 2015.

[I-D.thubert-6lo-bier-dispatch]
Thubert, P., Brodard, Z., Jiang, H., and G. Texier, "A 6loRH for BitStrings", draft-thubert-6lo-bier-dispatch-04 (work in progress), January 2018.

[I-D.thubert-6lo-fragment-recovery]
Thubert, P., "6LoWPAN Selective Fragment Recovery", draft-thubert-6lo-fragment-recovery-00 (work in progress), February 2018.

[I-D.thubert-bier-replication-elimination]
Thubert, P., Eckert, T., Brodard, Z., and H. Jiang, "BIER-TE extensions for Packet Replication and Elimination Function (PREF) and OAM", draft-thubert-bier-replication-elimination-03 (work in progress), March 2018.

[I-D.thubert-roll-unaware-leaves]
Thubert, P., "Routing for RPL Leaves", draft-thubert-roll-unaware-leaves-04 (work in progress), March 2018.

[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.

[I-D.watteyne-6lo-minimal-fragment]
Watteyne, T., Bormann, C., and P. Thubert, "LLN Minimal Fragment Forwarding", draft-watteyne-6lo-minimal-fragment-01 (work in progress), March 2018.

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, DOI 10.17487/RFC2474, December 1998, <https://www.rfc-editor.org/info/rfc2474>.

[RFC2545] Marques, P. and F. Dupont, "Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing", RFC 2545, DOI 10.17487/RFC2545, March 1999, <https://www.rfc-editor.org/info/rfc2545>.

[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, <https://www.rfc-editor.org/info/rfc3209>.

[RFC3444] Pras, A. and J. Schoenwaelder, "On the Difference between Information Models and Data Models", RFC 3444, DOI 10.17487/RFC3444, January 2003, <https://www.rfc-editor.org/info/rfc3444>.

[RFC3610] Whiting, D., Housley, R., and N. Ferguson, "Counter with CBC-MAC (CCM)", RFC 3610, DOI 10.17487/RFC3610, September 2003, <https://www.rfc-editor.org/info/rfc3610>.

[RFC3963] Devarapalli, V., Wakikawa, R., Petrescu, A., and P. Thubert, "Network Mobility (NEMO) Basic Support Protocol", RFC 3963, DOI 10.17487/RFC3963, January 2005, <https://www.rfc-editor.org/info/rfc3963>.

[RFC4080] Hancock, R., Karagiannis, G., Loughney, J., and S. Van den Bosch, "Next Steps in Signaling (NSIS): Framework", RFC 4080, DOI 10.17487/RFC4080, June 2005, <https://www.rfc-editor.org/info/rfc4080>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <https://www.rfc-editor.org/info/rfc4291>.

[RFC4389] Thaler, D., Talwar, M., and C. Patel, "Neighbor Discovery Proxies (ND Proxy)", RFC 4389, DOI 10.17487/RFC4389, April 2006, <https://www.rfc-editor.org/info/rfc4389>.

[RFC4429] Moore, N., "Optimistic Duplicate Address Detection (DAD) for IPv6", RFC 4429, DOI 10.17487/RFC4429, April 2006, <https://www.rfc-editor.org/info/rfc4429>.

[RFC4903] Thaler, D., "Multi-Link Subnet Issues", RFC 4903, DOI 10.17487/RFC4903, June 2007, <https://www.rfc-editor.org/info/rfc4903>.

[RFC4919] Kushalnagar, N., Montenegro, G., and C. Schumacher, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals", RFC 4919, DOI 10.17487/RFC4919, August 2007, <https://www.rfc-editor.org/info/rfc4919>.

[RFC5191] Forsberg, D., Ohba, Y., Ed., Patil, B., Tschofenig, H., and A. Yegin, "Protocol for Carrying Authentication for Network Access (PANA)", RFC 5191, DOI 10.17487/RFC5191, May 2008, <https://www.rfc-editor.org/info/rfc5191>.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, DOI 10.17487/RFC5340, July 2008, <https://www.rfc-editor.org/info/rfc5340>.

[RFC5889] Baccelli, E., Ed. and M. Townsley, Ed., "IP Addressing Model in Ad Hoc Networks", RFC 5889, DOI 10.17487/RFC5889, September 2010, <https://www.rfc-editor.org/info/rfc5889>.

[RFC5974] Manner, J., Karagiannis, G., and A. McDonald, "NSIS Signaling Layer Protocol (NSLP) for Quality-of-Service Signaling", RFC 5974, DOI 10.17487/RFC5974, October 2010, <https://www.rfc-editor.org/info/rfc5974>.

[RFC6275] Perkins, C., Ed., Johnson, D., and J. Arkko, "Mobility Support in IPv6", RFC 6275, DOI 10.17487/RFC6275, July 2011, <https://www.rfc-editor.org/info/rfc6275>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC6620] Nordmark, E., Bagnulo, M., and E. Levy-Abegnoli, "FCFS SAVI: First-Come, First-Served Source Address Validation Improvement for Locally Assigned IPv6 Addresses", RFC 6620, DOI 10.17487/RFC6620, May 2012, <https://www.rfc-editor.org/info/rfc6620>.

[RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, DOI 10.17487/RFC6830, January 2013, <https://www.rfc-editor.org/info/rfc6830>.

[RFC7426] Haleplidis, E., Ed., Pentikousis, K., Ed., Denazis, S., Hadi Salim, J., Meyer, D., and O. Koufopavlou, "Software-Defined Networking (SDN): Layers and Architecture Terminology", RFC 7426, DOI 10.17487/RFC7426, January 2015, <https://www.rfc-editor.org/info/rfc7426>.

8.3. Other Informative References

[ACE] IETF, "Authentication and Authorization for Constrained Environments", <https://dataTracker.ietf.org/doc/charter-ietf-ace/>.

[ANIMA] IETF, "Autonomic Networking Integrated Model and Approach", <https://dataTracker.ietf.org/doc/charter-ietf-anima/>.

[CCAMP] IETF, "Common Control and Measurement Plane", <https://dataTracker.ietf.org/doc/charter-ietf-ccamp/>.

[DETNET] IETF, "Deterministic Networking", <https://datatracker.ietf.org/doc/charter-ietf-detnet/>.

[DICE] IETF, "DTLS In Constrained Environments", <https://dataTracker.ietf.org/doc/charter-ietf-dice/>.

[HART] www.hartcomm.org, "Highway Addressable remote Transducer, a group of specifications for industrial process and control devices administered by the HART Foundation".

[IEC62439] IEC, "Industrial communication networks - High availability automation networks - Part 3: Parallel Redundancy Protocol (PRP) and High-availability Seamless Redundancy (HSR) - IEC62439-3", 2012, <https://webstore.iec.ch/publication/7018>.

[IEEE802.1TSNTG] IEEE Standards Association, "IEEE 802.1 Time-Sensitive Networks Task Group", March 2013, <http://www.ieee802.org/1/pages/avbridges.html>.

[IEEE802154]
IEEE standard for Information Technology, "IEEE Std. 802.15.4, Part. 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks".

[IEEE802154e]
IEEE standard for Information Technology, "IEEE standard for Information Technology, IEEE Std. 802.15.4, Part. 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks, June 2011 as amended by IEEE Std. 802.15.4e, Part. 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC sublayer", April 2012.

[ISA100] ISA/ANSI, "ISA100, Wireless Systems for Automation", <https://www.isa.org/isa100/>.

[ISA100.11a]
ISA/ANSI, "Wireless Systems for Industrial Automation: Process Control and Related Applications - ISA100.11a-2011 - IEC 62734", 2011, <http://www.isa.org/Community/SP100WirelessSystemsforAutomation>.

[PCE] IETF, "Path Computation Element", <https://dataTracker.ietf.org/doc/charter-ietf-pce/>.

[TEAS] IETF, "Traffic Engineering Architecture and Signaling", <https://dataTracker.ietf.org/doc/charter-ietf-teas/>.

[WirelessHART]
www.hartcomm.org, "Industrial Communication Networks - Wireless Communication Network and Communication Profiles - WirelessHART - IEC 62591", 2010.

Author's Address

Pascal Thubert (editor)
Cisco Systems, Inc
Building D
45 Allee des Ormes - BP1200
MOUGINS - Sophia Antipolis 06254
FRANCE

Phone: +33 497 23 26 34
Email: pthubert@cisco.com

6tisch Working Group
Internet-Draft
Intended status: Informational
Expires: August 29, 2017

M. Richardson
Sandelman Software Works

February 25, 2017

6tisch Secure Join protocol
draft-ietf-6tisch-dtsecurity-secure-join-01

Abstract

This document describes a zero-touch mechanism to enroll a new device (the "pledge") into a IEEE802.15.4 TSCH network using the 6tisch signaling mechanisms. The resulting device will obtain a domain specific credential that can be used with either 802.15.9 per-host pair keying protocols, or to obtain the network-wide key from a coordinator. The mechanism describe her is an augmentation to the one-touch mechanism described in [I-D.ietf-6tisch-minimal-security].

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 http://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 August 29, 2017.

Copyright Notice

Table of Contents

1. Introduction 2
1.1. Terminology 3
1.2. Credentials 4
1.2.1. One-Touch Assumptions 4
1.2.2. Factory provided credentials (if any) 4
1.2.3. Credentials to be introduced 5
1.3. Network Assumptions 5
1.3.1. Security above and below IP 5
1.3.2. Join network assumptions 6
1.3.3. Number and cost of round trips 6
1.3.4. Size of packets, number of fragments 7
1.4. Target end-state for join process 7
2. Join Protocol 7
2.1. Key Agreement process 8
2.2. Provisional Enrollment process 8
2.3. Key Distribution Process 9
3. YANG model for BRSKI objects 9
3.1. Description of Pledge States in Join Process 10
4. Definition of managed objects for zero-touch bootstrap . . . 10
5. Privacy Considerations 11
5.1. Privacy Considerations for Production network 11
5.2. Privacy Considerations for New Pledges 11
5.2.1. EUI-64 derived address for join time IID 12
5.3. Privacy Considerations for Join Assistant 12
6. Security Considerations 12
7. IANA Considerations 12
8. Protocol Definition 12
9. Acknwoledgements 12
10. References 12
10.1. Normative References 12
10.2. Informative References 15
10.3. URIs 16
Appendix A. appendix 16
Author’s Address 16

1. Introduction

Enrollment of new nodes into LLNs present unique challenges. The constrained nodes has no user interfaces, and even if they did, configuring thousands of such nodes manually is undesireable from a human resources issue, as well as the difficulty in getting consistent results.

This document is about a standard way to introduce new nodes into a 6tisch network that does not involve any direct manipulation of the nodes themselves. This act has been called "zero-touch" provisioning, and it does not occur by chance, but requires coordination between the manufacturer of the node, the service operator running the LLN, and the installers actually taking the devices out of the shipping boxes.

The act of doing "one-touch" provisioning, where a node undergoes a site-specific indoctrination process is described in [I-D.ietf-6tisch-minimal-security].

The mechanism described here and in [I-D.ietf-6tisch-minimal-security] can be discovered by a new node in a running network, so a device which has received a network-specific "one-touch" setup, but which is located in another network, and is capable of "zero-touch" operation could discovery this fact and operate in other mode.

Many of the components of the zero-touch mechanisms described here are in common with [I-D.ietf-anima-bootstrapping-keyinfra] and [I-D.ietf-netconf-zerotouch]. The on-the-wire pledge to join registrar protocols are different in this protocol from those described in ANIMA, but conceptually operate identically. The vouchers are identical. It is expected that the back-end network operator infrastructure would be able to bootstrap ANIMA-type devices over ethernet, while also being able bootstrap 6tisch devices over 802.15.4 with few changes.

1.1. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [RFC2119] and indicate requirement levels for compliant STuPiD implementations.

The reader is expected to be familiar with the terms and concepts defined in [I-D.ietf-6tisch-terminology], [RFC7252], [I-D.ietf-core-object-security], and [I-D.ietf-anima-bootstrapping-keyinfra]. The following terms are imported: drop ship, imprint, enrollment, pledge, join proxy, ownership voucher, join registrar/coordinator. The following terms are repeated here for readability, but this document is not authoritative for their definition:

pledge the prospective device, which has the identity provided to at the factory. Neither the device nor the network knows if the device yet knows if this device belongs with this network.

Joined Node the prospective device, after having completing the join process, often just called a Node.

Join Proxy (JP): a stateless relay that provides connectivity between the pledge and the join registrar/coordinator.

Join Registrar/Coordinator (JRC): central entity responsible for authentication and authorization of joining nodes.

Audit Token A signed token from the manufacturer authorized signing authority indicating that the bootstrapping event has been successfully logged. This has been referred to as an "authorization token" indicating that it authorizes bootstrapping to proceed.

Ownership Voucher A signed voucher from the vendor vouching that a specific domain "owns" the new entity as defined in [I-D.ietf-netconf-zerotouch].

MIC manufacturer installed certificate. An [ieee802-1AR] identity.

1.2. Credentials

In the zero-touch scenario, every device expected to be drop shipped would have an [ieee802-1AR] manufacturer installed certificate (MIC). The private key part of the certificate would either be generated in the device, or installed securely (and privately) as part of the manufacturing process. [cullenCiscoPhoneDeploy] provides an example of process which has been active for a good part of a decade.

The MIC would be signed by the manufacturer’s CA, the public key component of that would be included in the firmware.

1.2.1. One-Touch Assumptions

This document interacts with the one-touch solution described in [I-D.ietf-6tisch-minimal-security].

1.2.2. Factory provided credentials (if any)

When a manufacturer installed certificate is provided as the IDevID, it SHOULD contain a number of fields.
[I-D.ietf-anima-bootstrapping-keyinfra] provides a detailed set of requirements.

A manufacturer unique serial number MUST be provided in the serialNumber SubjectAltName extension, and MAY be repeated in the Common Name. There are no sequential or numeric requirements on the serialNumber, it may be any unique value that the manufacturer wants to use. The serialNumber SHOULD be printed on the packaging and/or on the device in a discrete way so that failures can be physically traced to the relevant device.

1.2.3. Credentials to be introduced

The goal of the bootstrap process is to introduce one or more new locally relevant credentials:

1. a certificate signed by a local certificate authority/registrar. This is the LDevID of [ieee802-1AR].

2. alternatively, a network-wide key to be used to secure L2 traffic.

3. alternatively, a network-wide key to be used to authenticate per-peer keying of L2 traffic using a mechanism such as provided by [ieee802159].

1.3. Network Assumptions

This document is about enrollment of constrained devices [RFC7228] to a constrained network. Constrained networks is such as [ieee802154], and in particular the time-slotted, channel hopping (tsch) mode, feature low bandwidths, and limited opportunities to transmit. A key feature of these networks is that receivers are only listening at certain times.

1.3.1. Security above and below IP

802.15.4 networks have three kinds of layer-2 security:

- a network key that is shared with all nodes and is used for unicast and multicast. The key may be used for privacy, and it may be used in some cases for authentication only (in the case of enhanced beacons).

- a series of network keys that are shared (agreed to) between pairs of nodes (the per-peer key)

- a network key that is shared with all nodes (through a group key management system), and is used for multicast traffic only, while a per-pair key is used for unicast traffic

Setting up the credentials to bootstrap one of these kinds of security, (or directly configuring the key itself for the first case) is required. This is the security below the IP layer.

Security is required above the IP layer: there are three aspects which the credentials in the previous section are to be used.

- to provide for secure connection with a Path Computation Element [RFC4655], or other LLC (see ({RFC7554}} section 3).

- to initiate a connection between a Resource Server (RS) and an application layer Authorization Server (AS and CAS from [I-D.ietf-ace-actors]).

1.3.1.1. Perfect Forward Secrecy

Perfert Forward Secrecy (PFS) is the property of a protocol such that complete knowledge of the crypto state (for instance, via a memory dump) at time X does not imply that data from a disjoint time Y can also be recovered. ([PFS]).

PFS is important for two reasons: one is that it offers protection against the compromise of a node. It does this by changing the keys in a non-deterministic way. This second property also makes it much easier to remove a node from the network, as any node which has not participated in the key changing process will find itself no longer connected.

1.3.2. Join network assumptions

The network which the new pledge will connect to will have to have the following properties:

- a known PANID. The PANID 0xXXXX where XXXX is the assigned RFC# for this document is suggested.

- a minimal schedule with some Aloha time. This is usually in the same slotframe as the Enhanced Beacon, but a pledge MUST listen for an unencrypted Enhanced Beacon to so that it can synchronize.

1.3.3. Number and cost of round trips

TBD.

1.3.4. Size of packets, number of fragments

1.4. Target end-state for join process

At the end of the zero-touch join process there will be a symmetric key protected channel between the Join Registrar/Coordinator and the pledge, now known as a Joined Node. This channel may be rekeyed via new exchange of asymmetric exponents (ECDH for instance), authenticated using the domain specific credentials created during the join process.

This channel is in the form of an OSCOAP protected connection with [I-D.ietf-core-comi] encoded objects. This document includes definition of a [I-D.ietf-netconf-keystore] compatible objects for encoding of the relevant [I-D.ietf-anima-bootstrapping-keyinfra] objects.

2. Join Protocol

The pledge join protocol state machine is described in [I-D.ietf-6tisch-minimal-security], in section XYZ. The pledge recognizes that it is in zero-touch configuration by the following situation:

- no PSK has been configured for the network in which it has joined.
- the pledge has no locally defined certificate (no LDevID), only an IDevID.
- the network asserts an identity that the pledge does not recognize.

All of these conditions MUST be true. If any of these are not true, then the pledge has either been connected to the wrong network, or it has already been bootstrapped into a different network, and it should wait until it finds that network.

The zero-touch process consists of three stages:

1. the key agreement process
2. the provisional enrollment process
3. the key distribution process

2.1. Key Agreement process

The key agreement process is identical to [I-D.ietf-6tisch-minimal-security]. The process uses EDHOC with certificates.

The pledge will have to trust the JRC provisionally, as described in [I-D.ietf-anima-bootstrapping-keyinfra], section 3.1.2, and in section 4.1.1 of [RFC7030].

The JRC will be able to validate the IDevID of the pledge using the manufacturer's CA.

The pledge may not know if it is in a zero-touch or one-touch situation: the pledge may be able to verify the JRC based upon trust anchors that were installed at manufacturing time. In that case, the pledge runs the simplified one-touch process.

The pledge signals in the EDHOC message_2 if it has accepted the JRC certificate. The JRC will in general, not trust the pledge with the network keys until it has provided the pledge with a voucher. The pledge will notice the absence of the provisioning keys.

XXX - there could be some disconnect here. May need additional signals here.

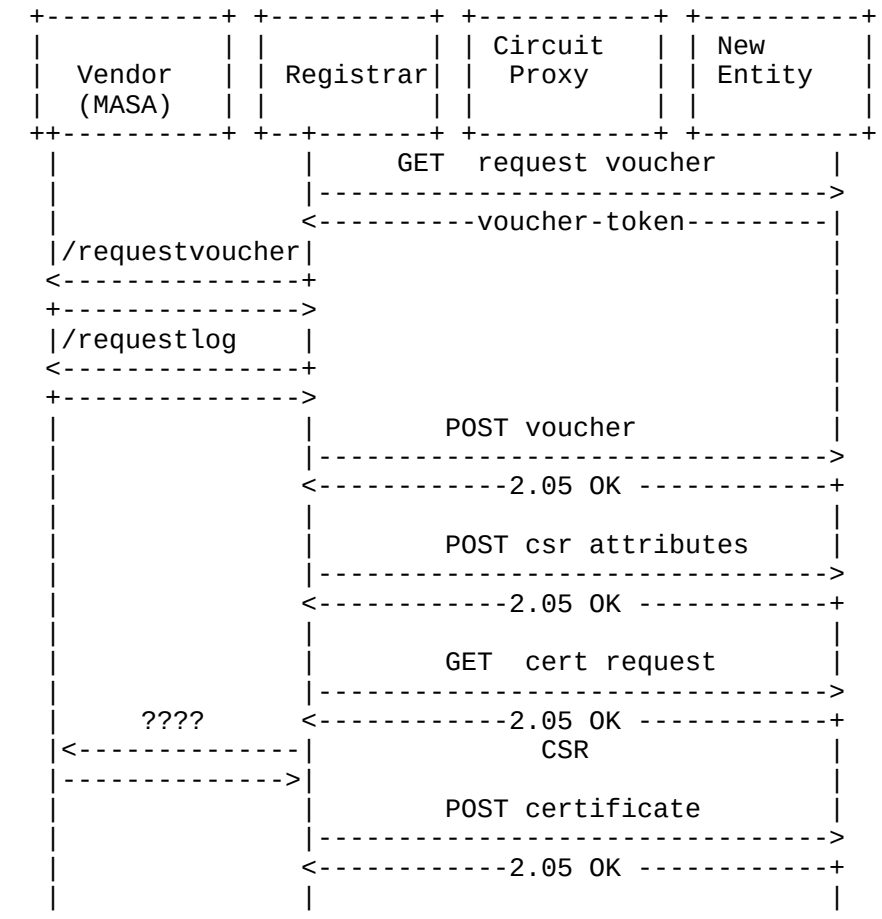
2.2. Provisional Enrollment process

When the pledge determines that it can not verify the certificate of the JRC using built-in trust anchors, then it enters a provisional state. In this state, it keeps the channel created by EDHOC open.

A new EDHOC key derivation is done by the JRC and pledge using a new label, "6tisch-provisional".

The pledge runs as a passive CoMI server, leaving the JRC to drive the enrollment process. The JRC can interrogate the pledge in a variety of fashions as shown below: the process terminates when the JRC provides the pledge with an ownership voucher and the pledge leaves the provisional state.

A typical interaction involves the following requests:



2.3. Key Distribution Process

The key distribution process utilizes the protocol described [I-D.richardson-6tisch-minimal-rekey]. The process starts with the initial key, rather than an actual rekey.

This protocol remains active for subsequent rekey operations.

3. YANG model for BRSKI objects

```
module ietf-6tisch-brski { yang-version 1.1;

namespace "urn:ietf:params:xml:ns:yang:6tisch-brski"; prefix
"ietf6brski";
```

```
//import ietf-yang-types { prefix yang; } //import ietf-inet-types {
prefix inet; }

organization "IETF 6tisch Working Group";

contact "WG Web: http://tools.ietf.org/wg/6tisch/ WG List:
6tisch@ietf.org [2] Author: Michael Richardson mcr+ietf@sandelman.ca
[3]";

description "This module defines an interface to set and retrieve
BRSKI objects using CoMI.  This interface is used as part of an
enrollment process for constrained nodes and networks.";

revision "2017-03-01" { description "Initial version"; reference "RFC
XXXX: 6tisch zero-touch bootstrap"; }

// top-level container container ietf6brski { leaf requestnonce {
type binary; length XX; // how big can/should it be? mandatory true;
description "Request Nonce."; } leaf voucher { type binary;
description "The voucher as a serialized COSE object"; }

leaf csrattributes {
  type binary;
  description "A list of attributes that MUST be in the CSR";
}

leaf certificaterequest {
  type binary;
  description "A PKIX format Certificate Request";
}

leaf certificate {
  type binary;
  description "The LDevID certificate";
}   } }
```

3.1. Description of Pledge States in Join Process

TBD

4. Definition of managed objects for zero-touch bootstrap

The following is relevant YANG for use in the bootstrap protocol. The objects identified are identical in format to the named objects from [I-D.ietf-anima-bootstrapping-keyinfra].

5. Privacy Considerations

[I-D.ietf-6lo-privacy-considerations] details a number of privacy considerations important in Resource Constrained nodes. There are two networks and three sets of constrained nodes to consider. They are: 1. the production nodes on the production network. 2. the new pledges, which have yet to enroll, and which are on a join network. 3. the production nodes which are also acting as proxy nodes.

5.1. Privacy Considerations for Production network

The details of this are out of scope for this document.

5.2. Privacy Considerations for New Pledges

New Pledges do not yet receive Router Advertisements with PIO options, and so configure link-local addresses only based upon layer-2 addresses using the normal SLAAC mechanisms described in [RFC4191].

These link-local addresses are visible to any on-link eavesdropper (who is synchronized to the same Join Assistant), so regardless of what is chosen they can be seen. This link-layer traffic is encapsulated by the Join Assistant into IPIP packets and carried to the JCE. The traffic SHOULD never leave the operator's network, and no outside traffic should enter, so it should not be possible to do any ICMP scanning as described in [I-D.ietf-6lo-privacy-considerations].

The join process described herein requires that some identifier meaningful to the network operator be communicated to the JCE via the Neighbor Advertisement's ARO option. This need not be a manufacturer created EUI-64 as assigned by IEEE; it could be another value with higher entropy and less interesting vendor/device information. Regardless of what is chosen, it can be used to track where the device attaches.

For most constrained device, network attachment occurs very infrequently, often only once in their lifetime, so tracking opportunities may be rare.

Further, during the enrollment process, a DTLS connection connection will be created. Unless TLS1.3 is used, the device identity will be visible to passive observers in the 802.11AR IDevID certificate that is sent. Even when TLS1.3 is used, an active attacker could collect the information by simply connecting to the device; it would not have to successful complete the negotiation either, or even attempt to Man-In-The-Middle the device.

There is, at the same time, significant value in avoiding a link-local DAD process by using an IEEE assigned EUI-64, and there is also significant advantage to the operator being able to see what the vendor of the new device is.

5.2.1. EUI-64 derived address for join time IID

It is therefore suggested that the IID used in the link-local address used during the join process be a vendor assigned EUI-64. After the join process has concluded, the device SHOULD be assigned a unique randomly generated long address, and a unique short address (not based upon the vendor EUI-64) for use at link-layer. At that point, all layer-3 content is encrypted by the layer-2 key.

5.3. Privacy Considerations for Join Assistant

6. Security Considerations

7. IANA Considerations

This document allocates one value from the subregistry "Address Registration Option Status Values": ND_NS_JOIN_DECLINED Join Assistant, JOIN DECLINED (TBD-AA)

8. Protocol Definition

9. Acknwoledgements

Kristofer Pister helped with many non-IETF references.

10. References

10.1. Normative References

[cullenCiscoPhoneDeploy]
Jennings, C., "Transitive Trust Enrollment for Constrained Devices", 2012, <http://www.lix.polytechnique.fr/hipercom/SmartObjectSecurity/papers/CullenJennings.pdf>.

[I-D.ietf-6lo-privacy-considerations]
Thaler, D., "Privacy Considerations for IPv6 Adaptation Layer Mechanisms", draft-ietf-6lo-privacy-considerations-04 (work in progress), October 2016.

[I-D.ietf-6tisch-minimal]
Vilajosana, X., Pister, K., and T. Watteyne, "Minimal 6TiSCH Configuration", draft-ietf-6tisch-minimal-21 (work in progress), February 2017.

[I-D.ietf-6tisch-minimal-security]
Vucinic, M., Simon, J., and K. Pister, "Minimal Security Framework for 6TiSCH", draft-ietf-6tisch-minimal-security-01 (work in progress), February 2017.

[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-08 (work in progress), December 2016.

[I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., Bjarnason, S., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-04 (work in progress), October 2016.

[I-D.ietf-anima-grasp]
Bormann, C., Carpenter, B., and B. Liu, "A Generic Autonomic Signaling Protocol (GRASP)", draft-ietf-anima-grasp-09 (work in progress), December 2016.

[I-D.ietf-core-comi]
Stok, P., Bierman, A., Veillette, M., and A. Pelov, "CoAP Management Interface", draft-ietf-core-comi-00 (work in progress), January 2017.

[I-D.ietf-core-object-security]
Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security of CoAP (OSCOAP)", draft-ietf-core-object-security-01 (work in progress), December 2016.

[I-D.ietf-netconf-keystore]
Watsen, K. and G. Wu, "Keystore Model", draft-ietf-netconf-keystore-00 (work in progress), October 2016.

[I-D.ietf-netconf-zerotouch]
Watsen, K. and M. Abrahamsson, "Zero Touch Provisioning for NETCONF or RESTCONF based Management", draft-ietf-netconf-zerotouch-12 (work in progress), January 2017.

[I-D.richardson-6tisch-join-enhanced-beacon]
Richardson, M., "802.15.4 Informational Element encapsulation of 6tisch Join Information", draft-richardson-6tisch-join-enhanced-beacon-00 (work in progress), February 2017.

[I-D.richardson-6tisch-minimal-rekey]
Richardson, M., "Minimal Security rekeying mechanism for 6TiSCH", draft-richardson-6tisch-minimal-rekey-00 (work in progress), February 2017.

[I-D.richardson-anima-6join-discovery]
Richardson, M., "GRASP discovery of Registrar by Join Assistant", draft-richardson-anima-6join-discovery-00 (work in progress), October 2016.

[iec62591]
IEC, ., "62591:2016 Industrial networks - Wireless communication network and communication profiles - WirelessHART", 2016, <https://webstore.iec.ch/publication/24433>.

[ieee802-1AR]
IEEE Standard, ., "IEEE 802.1AR Secure Device Identifier", 2009, <http://standards.ieee.org/findstds/standard/802.1AR-2009.html>.

[ieee802154]
IEEE Standard, ., "802.15.4-2015 - IEEE Standard for Low-Rate Wireless Personal Area Networks (WPANs)", 2015, <http://standards.ieee.org/findstds/standard/802.15.4-2015.html>.

[ieee802159]
IEEE Standard, ., "802.15.9-2016 - IEEE Approved Draft Recommended Practice for Transport of Key Management Protocol (KMP) Datagrams", 2016, <http://standards.ieee.org/findstds/standard/802.15.9-2016.html>.

[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>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <http://www.rfc-editor.org/info/rfc6775>.

[RFC7030] Pritikin, M., Ed., Yee, P., Ed., and D. Harkins, Ed., "Enrollment over Secure Transport", RFC 7030, DOI 10.17487/RFC7030, October 2013, <http://www.rfc-editor.org/info/rfc7030>.

[RFC7217] Gont, F., "A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC)", RFC 7217, DOI 10.17487/RFC7217, April 2014, <http://www.rfc-editor.org/info/rfc7217>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <http://www.rfc-editor.org/info/rfc7228>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <http://www.rfc-editor.org/info/rfc7252>.

10.2. Informative References

[duckling] Stajano, F. and R. Anderson, "The resurrecting duckling: security issues for ad-hoc wireless networks", 1999, <https://www.cl.cam.ac.uk/~fms27/papers/1999-StajanoAnd-duckling.pdf>.

[I-D.ietf-ace-actors] Gerdes, S., Seitz, L., Selander, G., and C. Bormann, "An architecture for authorization in constrained environments", draft-ietf-ace-actors-04 (work in progress), September 2016.

[I-D.ietf-roll-useofrplinfo] Robles, I., Richardson, M., and P. Thubert, "When to use RFC 6553, 6554 and IPv6-in-IPv6", draft-ietf-roll-useofrplinfo-10 (work in progress), December 2016.

[ISA100] "The Technology Behind the ISA100.11a Standard", June 2010, <http://www.isa100wci.org/Documents/PDF/The-Technology-Behind-ISA100-11a-v-3_pptx>.

[PFS] Wikipedia, ., "Forward Secrecy", August 2016, <https://en.wikipedia.org/w/index.php?title=Forward_secrecy&oldid=731318899>.

[pledge] Dictionary.com, ., "Dictionary.com Unabridged", 2015, <http://dictionary.reference.com/browse/pledge>.

[RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, DOI 10.17487/RFC4191, November 2005, <http://www.rfc-editor.org/info/rfc4191>.

[RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, DOI 10.17487/RFC4655, August 2006, <http://www.rfc-editor.org/info/rfc4655>.

[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>.

[RFC7731] Hui, J. and R. Kelsey, "Multicast Protocol for Low-Power and Lossy Networks (MPL)", RFC 7731, DOI 10.17487/RFC7731, February 2016, <http://www.rfc-editor.org/info/rfc7731>.

10.3. URIs

[2] mailto:6tisch@ietf.org

[3] mailto:mcr+ietf@sandelman.ca

Appendix A. appendix

insert appendix here

Author’s Address

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca

6TiSCH
Internet-Draft
Intended status: Best Current Practice
Expires: August 24, 2017

X. Vilajosana, Ed.
Universitat Oberta de Catalunya
K. Pister
University of California Berkeley
T. Watteyne
Linear Technology

February 20, 2017

Minimal 6TiSCH Configuration
draft-ietf-6tisch-minimal-21

Abstract

This document describes a minimal mode of operation for an IPv6 over the TSCH mode of IEEE 802.15.4e (6TiSCH) Network. This minimal mode of operation specifies the baseline set of protocols that need to be supported, recommended configurations and modes of operation sufficient to enable a 6TiSCH functional network. 6TiSCH provides IPv6 connectivity over a Time Synchronized Channel Hopping (TSCH) mesh composed of IEEE Std 802.15.4 TSCH links. This minimal mode uses a collection of protocols with the respective configurations, including the 6LoWPAN framework, enabling interoperable IPv6 connectivity over IEEE Std 802.15.4 TSCH. This minimal configuration provides the necessary bandwidth for network and security bootstrap, and defines the proper link between the IETF protocols that interface to IEEE Std 802.15.4 TSCH. This minimal mode of operation should be implemented by all 6TiSCH compliant devices.

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 http://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 August 24, 2017.

Table of Contents

1. Introduction 3
2. Requirements Language 4
3. Terminology 4
4. IEEE Std 802.15.4 Settings 4
4.1. TSCH Schedule 5
4.2. Cell Options 7
4.3. Retransmissions 7
4.4. Timeslot Timing 7
4.5. Frame Contents 7
4.5.1. IEEE Std 802.15.4 Header 8
4.5.2. Enhanced Beacon Frame 8
4.5.3. Acknowledgment Frame 9
4.6. Link-Layer Security 9
5. RPL Settings 10
5.1. Objective Function 10
5.1.1. Rank Computation 11
5.1.2. Rank Computation Example 12
5.2. Mode of Operation 13
5.3. Trickle Timer 13
5.4. Packet Contents 13
6. Network Formation and Lifetime 13
6.1. Value of the Join Metric Field 13
6.2. Time Source Neighbor Selection 14
6.3. When to Start Sending EBs 14
6.4. Hysteresis 14
7. Implementation Recommendations 15
7.1. Neighbor Table 15
7.2. Queues and Priorities 15
7.3. Recommended Settings 16
8. Security Considerations 16
9. IANA Considerations 18

10. Acknowledgments 18
11. References 18
11.1. Normative References 18
11.2. Informative References 20
11.3. External Informative References 21
Appendix A. Examples 21
A.1. Example: EB with Default Timeslot Template 21
A.2. Example: EB with Custom Timeslot Template 23
A.3. Example: Link-layer Acknowledgment 25
A.4. Example: Auxiliary Security Header 25
Authors' Addresses 26

1. Introduction

A 6TiSCH network provides IPv6 connectivity [RFC2460] over a Time Synchronized Channel Hopping (TSCH) mesh [RFC7554] composed of IEEE Std 802.15.4 TSCH links [IEEE802154-2015]. IPv6 connectivity is obtained by the use of the 6LoWPAN framework ([RFC4944], [RFC6282], [RFC8025],[I-D.ietf-roll-routing-dispatch] and [RFC6775]), RPL [RFC6550], and its Objective Function 0 (OF0) [RFC6552].

This specification defines operational parameters and procedures for a minimal mode of operation to build a 6TiSCH Network. Any 6TiSCH compliant device should implement this mode of operation. This operational parameter configuration provides the necessary bandwidth for nodes to bootstrap the network. The bootstrap process includes initial network configuration and security bootstrap. In this specification, the 802.15.4 TSCH mode, the 6LoWPAN framework, RPL [RFC6550], and its Objective Function 0 (OF0) [RFC6552] are used unmodified. Parameters and particular operations of TSCH are specified to guarantee interoperability between nodes in a 6TiSCH Network.

In a 6TiSCH network, nodes follow a communication schedule as per 802.15.4 TSCH. In it, nodes learn the schedule of the network when joining. When following this specification, the learned schedule is the same for all nodes and does not change over time. Future specifications may define mechanisms for dynamically managing the communication schedule. Dynamic scheduling solutions are out of scope of this document.

IPv6 addressing and compression are achieved by the 6LoWPAN framework. The framework includes [RFC4944], [RFC6282], [RFC8025], the 6LoWPAN Routing Header dispatch [I-D.ietf-roll-routing-dispatch] for addressing and header compression, and [RFC6775] for duplicate address detection (DAD) and address resolution.

More advanced work is expected in the future to complement the Minimal Configuration with dynamic operations that can adapt the schedule to the needs of the traffic at run time.

2. Requirements Language

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 RFC 2119 [RFC2119].

3. Terminology

This document uses terminology from [I-D.ietf-6tisch-terminology]. The following concepts are used in this document:

802.15.4: We use "802.15.4" as a short version of "IEEE Std 802.15.4" in this document.

SFD: Start of Frame Delimiter.

RX: Reception.

TX: Transmission.

IE: Information Element.

EB: Enhanced Beacon.

ASN: Absolute Slot Number.

Join Metric: Field in the TSCH Synchronization IE representing the topological distance between the node sending the EB and the PAN coordinator.

4. IEEE Std 802.15.4 Settings

An implementation compliant to this specification MUST implement IEEE Std 802.15.4 [IEEE802154-2015] in "timeslotted channel hopping" (TSCH) mode.

The remainder of this section details the RECOMMENDED TSCH settings, which are summarized in Figure 1. Any of the properties marked in the EB column are announced in the Enhanced Beacons (EB) the nodes send [IEEE802154-2015] and learned by those joining the network. Changing their value hence means changing the contents of the EB.

In case of discrepancy between the values in this specification and IEEE Std 802.15.4 [IEEE802154-2015], the IEEE standard has precedence.

Property	Recommended Setting	EB*
Slotframe Size	Tunable. Trades-off bandwidth against energy.	X
Number of scheduled cells** (active)	1 Timeslot 0x0000 Channel Offset 0x0000 Link Options = (TX Link = 1, RX Link = 1, Shared Link = 1, Timekeeping = 1)	X
Number of unscheduled cells (off)	All remaining cells in the slotframe	X
Max Number MAC retransmissions	3 (4 transmission attempts)	
Timeslot template	IEEE Std 802.15.4 default (macTimeslotTemplateId=0)	X
Enhanced Beacon Period (EB_PERIOD)	Tunable. Trades-off join time against energy.	
Number used frequencies (2.4 GHz O-QPSK PHY)	IEEE Std 802.15.4 default (16)	X
Channel Hopping sequence (2.4 GHz O-QPSK PHY)	IEEE Std 802.15.4 default (macHoppingSequenceID = 0)	X

* an "X" in this column means this property's value is announced in the EB; a new node hence learns it when joining.
** This cell LinkType is set to ADVERTISING.

Figure 1: Recommended IEEE Std 802.15.4 TSCH Settings.

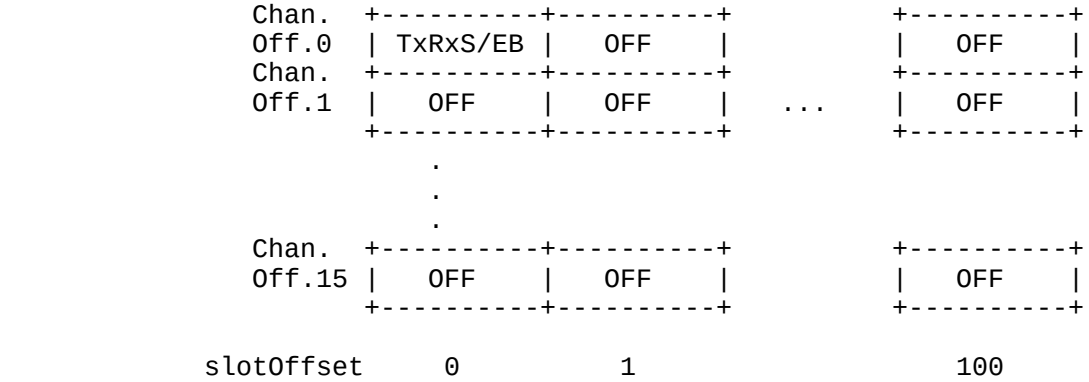
4.1. TSCH Schedule

This minimal mode of operation uses a single slotframe. The TSCH slotframe is composed of a tunable number of timeslots. The slotframe size (i.e. the number of timeslots it contains) trades off bandwidth for energy consumption. The slotframe size needs to be tuned; the way of tuning it is out of scope of this specification. The slotframe size is announced in the EB. The RECOMMENDED value for

the slotframe handle (macSlotframeHandle) is 0x00. An implementation MAY choose to use a different slotframe handle, for example to add other slotframes with higher priority. The use of other slotframes is out of the scope of this document.

There is only a single scheduled cell in the slotframe. This cell MAY be scheduled at any slotOffset/channelOffset within the slotframe. The location of that cell in the schedule is announced in the EB. The LinkType of the scheduled cell is ADVERTISING to allow EBs to be sent on it.

Figure 2 shows an example of a slotframe of length 101 timeslots, resulting in a radio duty cycle below 0.99%.



EB: Enhanced Beacon
Tx: Transmit
Rx: Receive
S: Shared
OFF: Unscheduled by this specification

Figure 2: Example slotframe of length 101 timeslots.

A node MAY use the scheduled cell to transmit/receive all types of link-layer frames. EBs are sent to the link-layer broadcast address and not acknowledged. Data frames are sent unicast, and acknowledged by the receiving neighbor.

All remaining cells in the slotframe are unscheduled. Dynamic scheduling solutions may be defined in the future which schedule those cells. One example is the 6top Protocol (6P) [I-D.ietf-6tisch-6top-protocol]. Dynamic scheduling solutions are out of scope of this document.

The default values of the TSCH Timeslot template (defined in [IEEE802154-2015] Section 8.4.2.2.3) and Channel Hopping sequence (defined in [IEEE802154-2015] Section 6.2.10) SHOULD be used. A node MAY use different values by properly announcing them in its Enhanced Beacon.

4.2. Cell Options

In the scheduled cell, a node transmits if there is a packet to transmit, listens otherwise (both "TX" and "RX" bits are set). When a node transmits, requesting a link-layer acknowledgment per [IEEE802154-2015], and does not receive it, it uses a back-off mechanism to resolve possible collisions ("Shared" bit is set). A node joining the network maintains time synchronization to its initial time source neighbor using that cell ("Timekeeping" bit is set).

This translates into a Link Option for this cell:

```
b0 = TX Link = 1 (set)
b1 = RX Link = 1 (set)
b2 = Shared Link = 1 (set)
b3 = Timekeeping = 1 (set)
b4 = Priority = 0 (clear)
b5-b7 = Reserved = 0 (clear)
```

4.3. Retransmissions

Per Figure 1, the RECOMMENDED maximum number of link-layer retransmissions is 3. This means that, for packets requiring an acknowledgment, if none are received after a total of 4 attempts, the transmission is considered failed and the link layer MUST notify the upper layer. Packets not requiring an acknowledgment (including EBs) are not retransmitted.

4.4. Timeslot Timing

Per Figure 1, the RECOMMENDED timeslot template is the default one (macTimeslotTemplateId=0) defined in [IEEE802154-2015].

4.5. Frame Contents

[IEEE802154-2015] defines the format of frames. Through a set of flags, [IEEE802154-2015] allows for several fields to be present or not, to have different lengths, and to have different values. This specification details the RECOMMENDED contents of 802.15.4 frames, while strictly complying to [IEEE802154-2015].

4.5.1. IEEE Std 802.15.4 Header

The Frame Version field MUST be set to 0b10 (Frame Version 2). The Sequence Number field MAY be elided.

EB Destination Address field MUST be set to 0xFFFF (short broadcast address). The EB Source Address field SHOULD be set as the node's short address if this is supported. Otherwise the long address MUST be used.

The PAN ID Compression bit SHOULD indicate that the Source PAN ID is "Not Present" and the Destination PAN ID is "Present". The value of the PAN ID Compression bit is specified in Table 7-2 of the IEEE Std 802.15.4-2015 specification, and depends on the type of the destination and source link-layer addresses (short, extended, not present).

Nodes follow the reception and rejection rules as per Section 6.7.2 of [IEEE802154-2015].

The Nonce is formatted according to [IEEE802154-2015]. In the IEEE Std 802.15.4 specification [IEEE802154-2015], nonce generation is described in Section 9.3.2.2, and byte ordering in Section 9.3.1, Annex B.2 and Annex B.2.2.

4.5.2. Enhanced Beacon Frame

After booting, a TSCH node starts in an unsynchronized, unjoined state. Initial synchronization is achieved by listening for EBs. EBs from multiple networks may be heard. Many mechanisms exist for discrimination between networks, the details of which are out of scope.

The IEEE Std 802.15.4 specification does not define how often EBs are sent, nor their contents [IEEE802154-2015]. In a minimal TSCH configuration, a node SHOULD send an EB every EB_PERIOD. Tuning EB_PERIOD allows a trade-off between joining time and energy consumption.

EBs should be used to obtain information about local networks, and to synchronize ASN and time offset of the specific network that the node decides to join. Once joined to a particular network, a node MAY choose to continue to listen for EBs, to gather more information about other networks, for example. During the joining process, before secure connections to time parents have been created, a node MAY maintain synchronization using EBs. [RFC7554] discusses different time synchronization approaches.

The IEEE Std 802.15.4 specification requires EBs to be send in order to enable nodes to join the network. The EBs SHOULD carry the Information Elements (IEs) listed below [IEEE802154-2015].

TSCH Synchronization IE: Contains synchronization information such as ASN and Join Metric. The value of the Join Metric field is discussed in Section 6.1.

TSCH Timeslot IE: Contains the timeslot template identifier. This template is used to specify the internal timing of the timeslot. This specification RECOMMENDS the default timeslot template.

Channel Hopping IE: Contains the channel hopping sequence identifier. This specification RECOMMENDS the default channel hopping sequence.

TSCH Slotframe and Link IE: Enables joining nodes to learn the initial schedule to be used as they join the network. This document RECOMMENDS the use of a single cell.

If a node strictly follows the recommended setting from Figure 1, the EB it sends has the exact same contents as an EB it has received when joining, except for the Join Metric field in the TSCH Synchronization IE.

When a node has already joined a network, i.e. it has received an EB, synchronized to the EB sender and configured its schedule following this specification, the node SHOULD ignore subsequent EBs which try to change the configured parameters. This does not preclude listening EBs from other networks.

4.5.3. Acknowledgment Frame

Per [IEEE802154-2015], each acknowledgment contain an ACK/NACK Time Correction IE.

4.6. Link-Layer Security

When securing link-layer frames, link-layer frames MUST be secured by the link-layer security mechanisms defined in IEEE Std 802.15.4 [IEEE802154-2015]. Link-layer authentication MUST be applied to the entire frame, including the 802.15.4 header. Link-layer encryption MAY be applied to 802.15.4 payload IEs and the 802.15.4 payload.

This specification assumes the existence of two cryptographic keys:

Key K1 is used to authenticate EBs. EBs MUST be authenticated only (no encryption), and their contents is defined in Section 4.5.2.

Key K2 is used to authenticate and encrypt DATA and ACKNOWLEDGMENT frames.

These keys can be pre-configured, or learned during a key distribution phase. Key distribution mechanisms are defined for example in [I-D.ietf-6tisch-minimal-security] and [I-D.ietf-6tisch-dtsecurity-secure-join]. Key distribution is out of scope of this document.

The behavior of a Joining Node (JN) is different depending on which key(s) are pre-configured:

If both keys K1 and K2 are pre-configured, the JN does not rely on a key distribution phase to learn K1 or K2.

If key K1 is pre-configured but not key K2, the JN authenticates EBs using K1, and relies on the key distribution phase to learn K2.

If neither key K1 nor key K2 is pre-configured, the JN accepts EBs as defined in Section 6.3.1.2 of IEEE Std 802.15.4 [IEEE802154-2015], i.e., they are passed forward even "if the status of the unsecuring process indicated an error". The JN then runs key distribution phase to learn K1 and K2. During that process, the node JN is talking to uses the secExempt mechanism (IEEE Std 802.15.4, Section 9.2.4) to process frames from JN. Once the key distribution phase is done, the node which has installed secExempts for the JN MUST clear the installed exception rules.

In the event of a network reset, the new network MUST either use new cryptographic keys, or ensure that the ASN remains monotonically increasing.

5. RPL Settings

In a multi-hop topology, the RPL routing protocol [RFC6550] MAY be used.

5.1. Objective Function

If RPL is used, nodes MUST implement the RPL Objective Function Zero (OF0) [RFC6552].

5.1.1. Rank Computation

The Rank computation is described at [RFC6552], Section 4.1. A node's Rank (see Figure 4 for an example) is computed by the following equations:

R(N) = R(P) + rank_increment

rank_increment = (Rf*Sp + Sr) * MinHopRankIncrease

Figure 3 lists the OF0 parameter values that MUST be used if RPL is used.

OF0 Parameters	Value
Rf	1
Sp	(3*ETX)-2
Sr	0
MinHopRankIncrease	DEFAULT_MIN_HOP_RANK_INCREASE (256)
MINIMUM_STEP_OF_RANK	1
MAXIMUM_STEP_OF_RANK	9
ETX limit to select a parent	3

Figure 3: OF0 parameters.

The step_of_rank (Sp) uses Expected Transmission Count (ETX) [RFC6551].

An implementation MUST follow OF0's normalization guidance as discussed in Section 1 and Section 4.1 of [RFC6552]. Sp SHOULD be calculated as (3*ETX)-2. The minimum value of Sp (MINIMUM_STEP_OF_RANK) indicates a good quality link. The maximum value of Sp (MAXIMUM_STEP_OF_RANK) indicates a poor quality link. The default value of Sp (DEFAULT_STEP_OF_RANK) indicates an average quality link. Candidate parents with ETX greater than 3 SHOULD NOT be selected. This avoids having ETX values on used links which are larger that the maximum allowed transmission attempts.

5.1.2. Rank Computation Example

This section illustrates the use of the Objective Function Zero (see Figure 4). We have:

rank_increment = ((3*numTx/numTxAck)-2)*minHopRankIncrease = 512

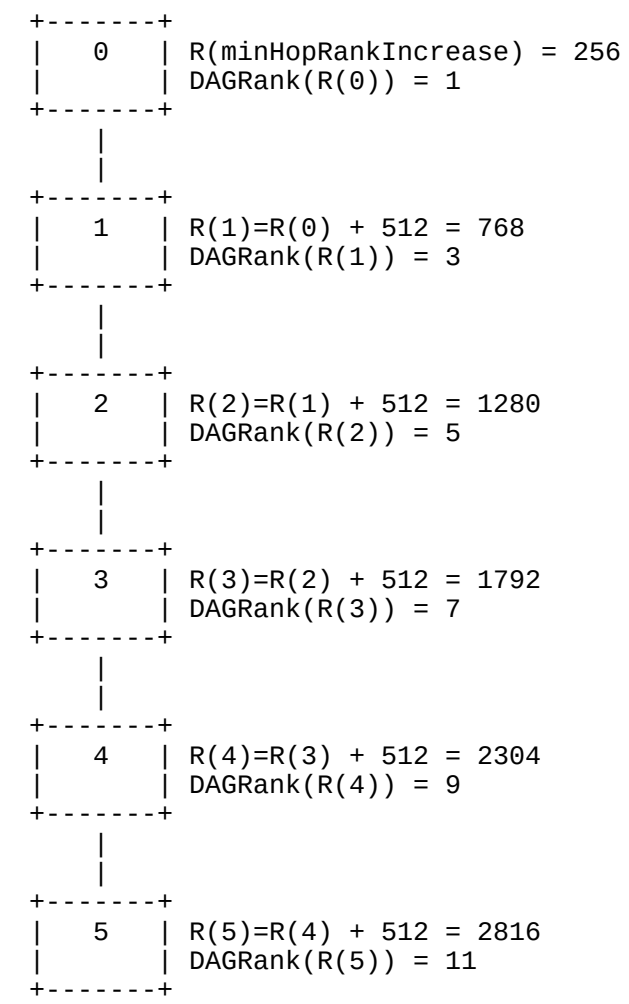


Figure 4: Rank computation example for 5-hop network where numTx=100 and numTxAck=75 for all links.

5.2. Mode of Operation

When RPL is used, nodes MUST implement the non-storing ([RFC6550] Section 9.7) mode of operation. The storing ([RFC6550] Section 9.8) mode of operation SHOULD be implemented by nodes with enough capabilities. Nodes not implementing RPL MUST join as leaf nodes.

5.3. Trickle Timer

RPL signaling messages such as DIOs are sent using the Trickle Algorithm [RFC6550] (Section 8.3.1) and [RFC6206] (Section 4.2). For this specification, the Trickle Timer MUST be used with the RPL defined default values [RFC6550] (Section 8.3.1).

5.4. Packet Contents

RPL information and hop-by-hop extension headers MUST follow [RFC6553] and [RFC6554]. For cases in which the packets formed at the LLN need to cross through intermediate routers, these MUST follow the IP-in-IP encapsulation requirement specified by [RFC6282] and [RFC2460]. Routing extension headers such as RPI [RFC6550] and SRH [RFC6554], and outer IP headers in case of encapsulation MUST be compressed according to [I-D.ietf-roll-routing-dispatch] and [RFC8025].

6. Network Formation and Lifetime

6.1. Value of the Join Metric Field

The Join Metric of the TSCH Synchronization IE in the EB MUST be calculated based on the routing metric of the node, normalized to a value between 0 and 255. A lower value of the Join Metric indicates the node sending the EB is topologically "closer" to the root of the network. A lower value of the Join Metric hence indicates higher preference for a joining node to synchronize to that neighbor.

In case the network uses RPL, the Join Metric of any node (including the DAG root) MUST be set to DAGRank(rank)-1. According to Section 5.1.1, DAGRank(rank(0)) = 1. DAGRank(rank(0))-1 = 0 is compliant with 802.15.4’s requirement of having the root use Join Metric = 0.

In case the network does not use RPL, the Join Metric value MUST follow the rules specified by [IEEE802154-2015].

6.2. Time Source Neighbor Selection

When a node joins a network, it may hear EBs sent by different nodes already in the network. The decision of which neighbor to synchronize to (e.g. which neighbor becomes the node's initial time source neighbor) is implementation-specific. For example, after having received the first EB, a node MAY listen for at most MAX_EB_DELAY seconds until it has received EBs from NUM_NEIGHBOURS_TO_WAIT distinct neighbors. Recommended values for MAX_EB_DELAY and NUM_NEIGHBOURS_TO_WAIT are defined in Figure 5. When receiving EBs from distinct neighbors, the node MAY use the Join Metric field in each EB to select the initial time source neighbor, as described in IEEE Std 802.15.4 [IEEE802154-2015], Section 6.3.6.

At any time, a node MUST maintain synchronization to at least one time source neighbor. A node's time source neighbor MUST be chosen among the neighbors in its RPL routing parent set when RPL is used. In the case a node cannot maintain connectivity to at least one time source neighbor, the node looses synchronization and needs to join the network again.

6.3. When to Start Sending EBs

When a RPL node joins the network, it MUST NOT send EBs before having acquired a RPL Rank to avoid inconsistencies in the time synchronization structure. This applies to other routing protocols with their corresponding routing metrics. As soon as a node acquires routing information (e.g. a RPL Rank, see Section 5.1.1), it SHOULD start sending Enhanced Beacons.

6.4. Hysteresis

Per [RFC6552] and [RFC6719], the specification RECOMMENDS the use of a boundary value (PARENT_SWITCH_THRESHOLD) to avoid constant changes of the parent when ranks are compared. When evaluating a parent that belongs to a smaller path cost than the current minimum path, the candidate node is selected as new parent only if the difference between the new path and the current path is greater than the defined PARENT_SWITCH_THRESHOLD. Otherwise, the node MAY continue to use the current preferred parent. Per [RFC6719], the PARENT_SWITCH_THRESHOLD SHOULD be set to 192 when ETX metric is used (in the form 128*ETX), the recommendation for this document is to use PARENT_SWITCH_THRESHOLD equal to 640 if the metric being used is ((3*ETX)-2)*minHopRankIncrease, or a proportional value. This deals with hysteresis both for routing parent and time source neighbor selection.

7. Implementation Recommendations

7.1. Neighbor Table

The exact format of the neighbor table is implementation-specific. The RECOMMENDED per-neighbor information is (taken from the [openwsn] implementation):

identifier: Identifier(s) of the neighbor (e.g. EUI-64).

numTx: Number of link-layer transmission attempts to that neighbor.

numTxAck: Number of transmitted link-layer frames that have been link-layer acknowledged by that neighbor.

numRx: Number of link-layer frames received from that neighbor.

timestamp: When the last frame was received from that neighbor. This can be based on the ASN counter or any other time base. It can be used to trigger a keep-alive message.

routing metric: Such as the RPL Rank of that neighbor.

time source neighbor: A flag indicating whether this neighbor is a time source neighbor.

7.2. Queues and Priorities

The IEEE Std 802.15.4 specification [IEEE802154-2015] does not define the use of queues to handle upper-layer data (either application or control data from upper layers). The following rules are RECOMMENDED:

A node is configured to keep in the queues a configurable number of upper-layer packets per link (default NUM_UPPERLAYER_PACKETS) for a configurable time that should cover the join process (default MAX_JOIN_TIME).

Frames generated by the 802.15.4 layer (including EBs) are queued with a priority higher than frames coming from higher-layers.

Frame type BEACON is queued with higher priority than frame types DATA.

7.3. Recommended Settings

Figure 5 lists RECOMMENDED values for the settings discussed in this specification.

Parameter	RECOMMENDED Value
MAX_EB_DELAY	180
NUM_NEIGHBOURS_TO_WAIT	2
PARENT_SWITCH_THRESHOLD	640
NUM_UPPERLAYER_PACKETS	1
MAX_JOIN_TIME	300

Figure 5: Recommended Settings.

8. Security Considerations

This document is concerned only with link-layer security.

By their nature, many IoT networks have nodes in physically vulnerable locations. We should assume that nodes will be physically compromised, their memories examined, and their keys extracted. Fixed secrets will not remain secret. This impacts the node joining process. Provisioning a network with a fixed link key K2 is not secure. For most applications, this implies that there will be a joining phase during which some level of authorization will be allowed for nodes which have not been authenticated. Details are out of scope, but the link layer must provide some flexibility here.

If an attacker has obtained K1 it can generate fake EBs to attack whole network by sending authenticated EBs. The attacker can cause the joining node to initiate the joining process to the attacker. In the case that the joining process includes authentication and distribution of a K2, then the joining process will fail and the JN will notice the attack. If K2 is also compromised the JN will not notice the attack and the network will be compromised.

Even if an attacker does not know the value of K1 and K2 (Section 4.6), it can still generate fake EB frames, authenticated with an arbitrary key. We here discuss the impact these fake EBs can have, depending on what key(s) are pre-provisioned.

If both K1 and K2 are pre-provisioned, a joining node can distinguish legitimate from fake EBs, and join the legitimate network. The fake EBs have no impact.

The same holds if K1 is pre-provisioned but not K2.

If neither K1 nor K2 is pre-provisioned, a joining node may mistake a fake EB for a legitimate one and initiate a joining process to the attacker. That joining process will fail, as the joining node will not be able to authenticate the attacker during the security handshake. This will force the joining node to start over listening for an EB. So while the joining node never joins the attacker, this costs the joining node time and energy, and is a vector of attack.

Choosing what key(s) to pre-provision need to balance the different discussions above.

Once the joining process is over, the node that has joined can authenticate EBs (it knows K1). This means it can process their contents and use EBs for synchronization.

ASN provides a nonce for security operations in a slot. Any re-use of ASN with a given key exposes information about encrypted packet contents, and risks replay attacks. Replay attacks are prevented because, when the network resets, either the new network uses new cryptographic key(s), or ensures that the ASN increases monotonically (Section 4.6).

Maintaining accurate time synchronization is critical for network operation. Accepting timing information from unsecured sources MUST be avoided during normal network operation, as described in Section 4.5.2. During joining, a node may be susceptible to timing attacks before key K1 and K2 are learned. During network operation, a node MAY maintain statistics on time updates from neighbors and monitor for anomalies.

Denial of Service (DoS) attacks at the MAC layer in an LLN are easy to achieve simply by RF jamming. This is the base case against which more sophisticated DoS attacks should be judged. For example, sending fake EBs announcing a very low Join Metric may cause a node to waste time and energy trying to join a fake network even when legitimate EBs are being heard. Proper join security will prevent the node from joining the false flag, but by then the time and energy will have been wasted. However, the energy cost to the attacker would be lower and the energy cost to the joining node higher if the attacker simply sent loud short packets in the middle of any valid EB that it hears.

ACK reception probability is less than 100%, due to changing channel conditions and unintentional or intentional jamming. This will cause the sending node to retransmit the same packet until it is acknowledged or a retransmission limit is reached. Upper layer protocols should take this into account, possibly using a sequence number to match retransmissions.

The 6TiSCH layer SHOULD keep track of anomalous events and report them to a higher authority. For example, EBs reporting low Join Metrics for networks which cannot be joined, as described above, may be a sign of attack. Additionally, in normal network operation, message integrity check failures on packets with valid CRC will occur at a rate on the order of once per million packets. Any significant deviation from this rate may be a sign of network attack. Along the same lines, time updates in ACKs or EBs that are inconsistent with the MAC-layer's sense of time and its own plausible time error drift rate may also be a result of network attack.

9. IANA Considerations

This document requests no immediate action by IANA.

10. Acknowledgments

The authors acknowledge the guidance and input from Rene Struik, Pat Kinney, Michael Richardson, Tero Kivinen, Nicola Accettura, Malisa Vucinic and Jonathan Simon. Thanks to Charles Perkins, Brian E. Carpenter, Ralph Droms, Warren Kumari, Mirja Kuehlewind, Ben Campbell, Benoit Claise and Suresh Krishnan for the exhaustive and detailed reviews. Thanks to Simon Duquennoy, Guillaume Gaillard, Tengfei Chang and Jonathan Munoz for the detailed review of the examples section. Thanks to 6TiSCH co-chair Pascal Thubert for his guidance and advice.

11. References

11.1. Normative References

[I-D.ietf-roll-routing-dispatch]
Thubert, P., Bormann, C., Toutain, L., and R. Cragie, "6LoWPAN Routing Header", draft-ietf-roll-routing-dispatch-05 (work in progress), October 2016.

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

[RFC8025] Thubert, P., Ed. and R. Cragie, "IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Paging Dispatch", RFC 8025, DOI 10.17487/RFC8025, November 2016, <http://www.rfc-editor.org/info/rfc8025>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <http://www.rfc-editor.org/info/rfc6775>.

[RFC6719] Gnawali, O. and P. Levis, "The Minimum Rank with Hysteresis Objective Function", RFC 6719, DOI 10.17487/RFC6719, September 2012, <http://www.rfc-editor.org/info/rfc6719>.

[RFC6554] Hui, J., Vasseur, JP., Culler, D., and V. Manral, "An IPv6 Routing Header for Source Routes with the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6554, DOI 10.17487/RFC6554, March 2012, <http://www.rfc-editor.org/info/rfc6554>.

[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>.

[RFC6552] Thubert, P., Ed., "Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6552, DOI 10.17487/RFC6552, March 2012, <http://www.rfc-editor.org/info/rfc6552>.

[RFC6551] Vasseur, JP., Ed., Kim, M., Ed., Pister, K., Dejean, N., and D. Barthel, "Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks", RFC 6551, DOI 10.17487/RFC6551, March 2012, <http://www.rfc-editor.org/info/rfc6551>.

[RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, DOI 10.17487/RFC6550, March 2012, <http://www.rfc-editor.org/info/rfc6550>.

[RFC6282] Hui, J., Ed. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, DOI 10.17487/RFC6282, September 2011, <http://www.rfc-editor.org/info/rfc6282>.

[RFC6206] Levis, P., Clausen, T., Hui, J., Gnawali, O., and J. Ko, "The Trickle Algorithm", RFC 6206, DOI 10.17487/RFC6206, March 2011, <http://www.rfc-editor.org/info/rfc6206>.

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, DOI 10.17487/RFC4944, September 2007, <http://www.rfc-editor.org/info/rfc4944>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <http://www.rfc-editor.org/info/rfc2460>.

[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>.

11.2. Informative References

[I-D.ietf-6tisch-6top-protocol]
Wang, Q. and X. Vilajosana, "6top Protocol (6P)", draft-ietf-6tisch-6top-protocol-03 (work in progress), October 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-08 (work in progress), December 2016.

[I-D.ietf-6tisch-minimal-security]
Vucinic, M., Simon, J., and K. Pister, "Minimal Security Framework for 6TiSCH", draft-ietf-6tisch-minimal-security-01 (work in progress), February 2017.

[I-D.ietf-6tisch-dtsecurity-secure-join]
Richardson, M., "6tisch Secure Join protocol", draft-ietf-6tisch-dtsecurity-secure-join-00 (work in progress), December 2016.

[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>.

11.3. External Informative References

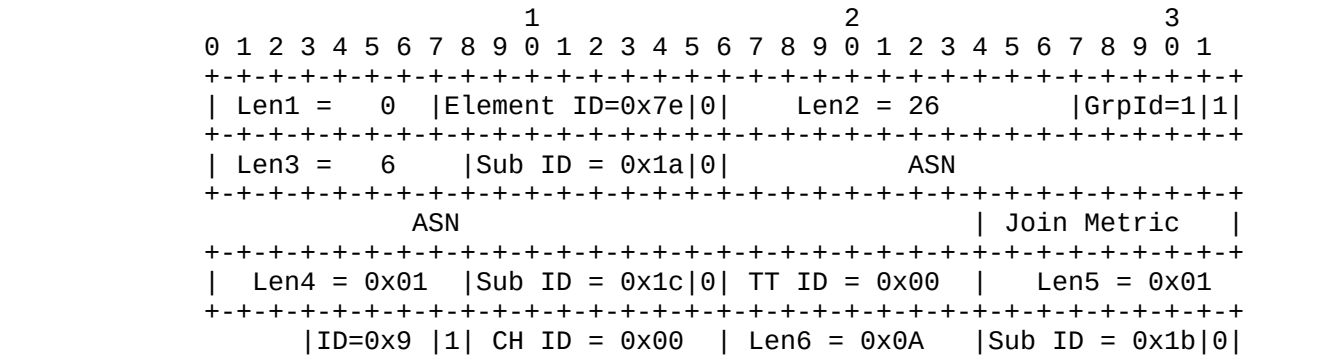
[openwsn] Watteyne, T., Vilajosana, X., Kerkez, B., Chraim, F., Weekly, K., Wang, Q., Glaser, S., and K. Pister, "OpenWSN: a Standards-Based Low-Power Wireless Development Environment", Transactions on Emerging Telecommunications Technologies , August 2012.

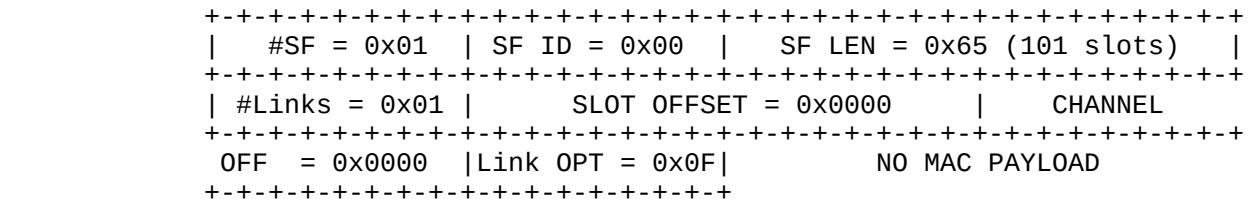
Appendix A. Examples

This section contains several example packets. Each example contains (1) a schematic header diagram, (2) the corresponding bytestream, (3) a description of each of the IEs that form the packet. Packet formats are specific for the [IEEE802154-2015] revision and may vary in future releases of the IEEE standard. In case of differences between the packet content presented in this section and [IEEE802154-2015], the latter has precedence.

The MAC header fields are described in a specific order. All field formats in this examples are depicted in the order in which they are transmitted, from left to right, where the leftmost bit is transmitted first. Bits within each field are numbered from 0 (leftmost and least significant) to k - 1 (rightmost and most significant), where the length of the field is k bits. Fields that are longer than a single octet are sent to the PHY in the order from the octet containing the lowest numbered bits to the octet containing the highest numbered bits (little endian).

A.1. Example: EB with Default Timeslot Template





Bytestream:

```
00 3F 1A 88 06 1A ASN#0 ASN#1 ASN#2 ASN#3 ASN#4 JP 01 1C 00
01 C8 00 0A 1B 01 00 65 00 01 00 00 00 00 0F
```

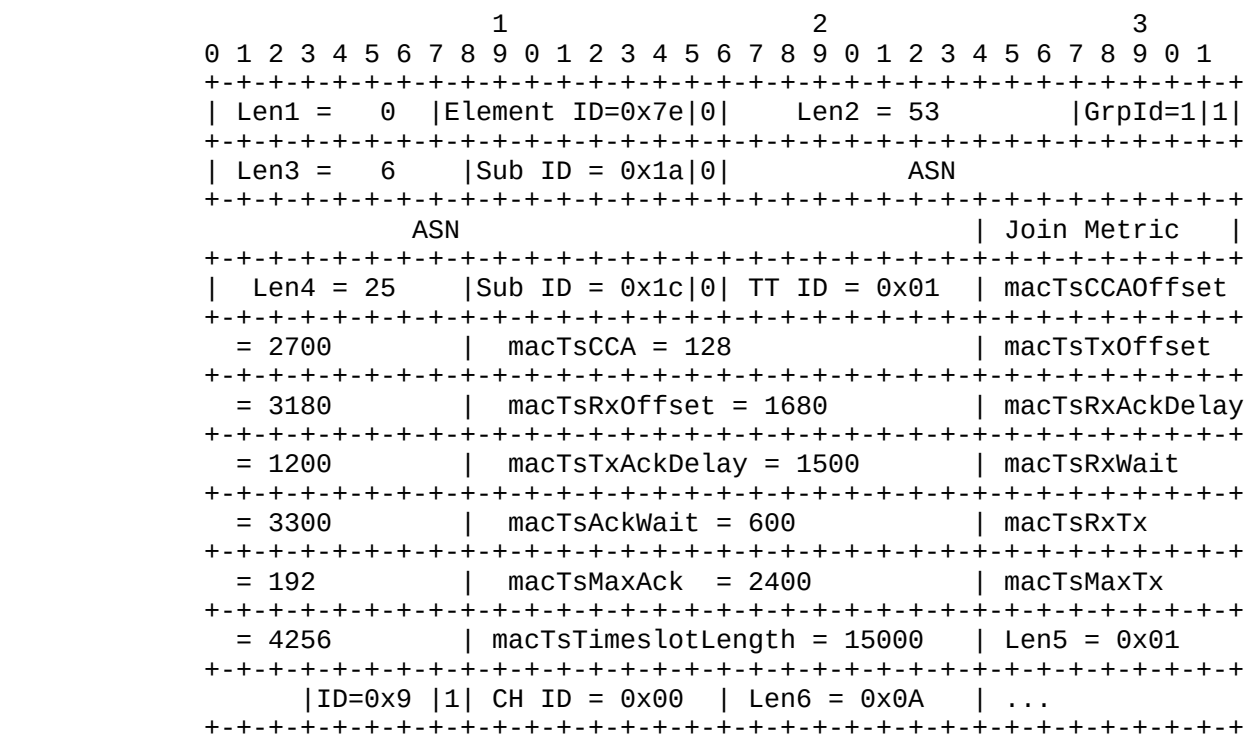
Description of the IEs:

```
#Header IE Header
    Len1 = Header IE Length (0)
    Element ID = 0x7e - termination IE indicating Payload IE
        coming next
    Type 0

#Payload IE Header (MLME)
    Len2 = Payload IE Len (26 Bytes)
    Group ID = 1 MLME (Nested)
    Type = 1

#MLME-SubIE TSCH Synchronization
    Len3 = Length in bytes of the sub-IE payload (6 Bytes)
    Sub-ID = 0x1a (MLME-SubIE TSCH Synchronization)
    Type = Short (0)
    ASN  = Absolute Sequence Number (5 Bytes)
    Join Metric = 1 Byte

#MLME-SubIE TSCH Timeslot
    Len4 = Length in bytes of the sub-IE payload (1 Byte)
    Sub-ID = 0x1c (MLME-SubIE Timeslot)
    Type = Short (0)
    Timeslot template ID = 0x00 (default)

#MLME-SubIE Channel Hopping
    Len5 = Length in bytes of the sub-IE payload (1 Byte)
    Sub-ID = 0x09 (MLME-SubIE Channel Hopping)
    Type = Long (1)
    Hopping Sequence ID = 0x00 (default)

#MLME-SubIE TSCH Slotframe and Link
    Len6 = Length in bytes of the sub-IE payload (10 Bytes)
    Sub-ID = 0x1b (MLME-SubIE TSCH Slotframe and Link)
```

```
Type = Short (0)
Number of slotframes = 0x01
Slotframe handle = 0x00
Slotframe size = 101 slots (0x65)
Number of Links (Cells) = 0x01
Timeslot = 0x0000 (2B)
Channel Offset = 0x0000 (2B)
Link Options = 0x0F
(TX Link = 1, RX Link = 1, Shared Link = 1,
 Timekeeping = 1 )
```

A.2. Example: EB with Custom Timeslot Template

Using a custom timeslot template in EBs: setting timeslot length to 15ms.



Bytestream:

```
00 3F 1A 88 06 1A ASN#0 ASN#1 ASN#2 ASN#3 ASN#4 JP 19 1C 01 8C 0A 80
00 6C 0C 90 06 B0 04 DC 05 E4 0C 58 02 C0 00 60 09 A0 10 98 3A 01 C8
00 0A ...
```

Description of the IEs:

#Header IE Header
Len1 = Header IE Length (none)
Element ID = 0x7e - termination IE indicating Payload IE coming next
Type 0

#Payload IE Header (MLME)
Len2 = Payload IE Len (53 Bytes)
Group ID = 1 MLME (Nested)
Type = 1

#MLME-SubIE TSCH Synchronization
Len3 = Length in bytes of the sub-IE payload (6 Bytes)
Sub-ID = 0x1a (MLME-SubIE TSCH Synchronization)
Type = Short (0)
ASN = Absolute Sequence Number (5 Bytes)
Join Metric = 1 Byte

#MLME-SubIE TSCH Timeslot
Len4 = Length in bytes of the sub-IE payload (25 Bytes)
Sub-ID = 0x1c (MLME-SubIE Timeslot)
Type = Short (0)
Timeslot template ID = 0x01 (non-default)

The 15ms timeslot announced:

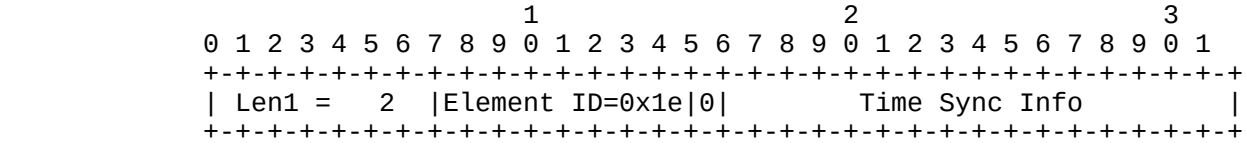
IEEE 802.15.4 TSCH parameter	Value (us)
macTsCCAOffset	2700
macTsCCA	128
macTsTxOffset	3180
macTsRxOffset	1680
macTsRxAckDelay	1200
macTsTxAckDelay	1500
macTsRxWait	3300
macTsAckWait	600
macTsRxTx	192

```
| macTsMaxAck                    |       2400 |
+--------------------------------+------------+
| macTsMaxTx                     |       4256 |
+--------------------------------+------------+
| macTsTimeslotLength            |      15000 |
+--------------------------------+------------+
```

#MLME-SubIE Channel Hopping
Len5 = Length in bytes of the sub-IE payload. (1 Byte)
Sub-ID = 0x09 (MLME-SubIE Channel Hopping)
Type = Long (1)
Hopping Sequence ID = 0x00 (default)

A.3. Example: Link-layer Acknowledgment

Enhanced Acknowledgment packets carry the Time Correction IE (Header IE).



Bytestream:

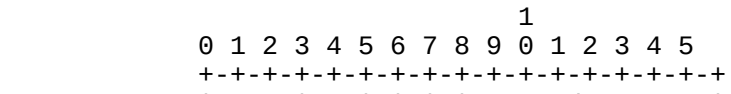
02 0F TS#0 TS#1

Description of the IEs:

#Header IE Header
Len1 = Header IE Length (2 Bytes)
Element ID = 0x1e - ACK/NACK Time Correction IE
Type 0

A.4. Example: Auxiliary Security Header

802.15.4 Auxiliary Security Header with security Level set to ENC-MIC-32.



Bytestream:

6D IDX#0

Security Auxiliary Header fields in the example:

#Security Control (1 byte)
L = Security Level ENC-MIC-32 (5)
M = Key Identifier Mode (0x01)
Frame Counter Suppression = 1 (omitting Frame Counter field)
ASN in Nonce = 1 (construct Nonce from 5 byte ASN)
Reserved = 0

#Key Identifier (1 byte)
Key Index = IDX (deployment-specific KeyIndex parameter that identifies the cryptographic key)

Authors' Addresses

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

Email: xvilajosana@uoc.edu

Kris Pister
University of California Berkeley
512 Cory Hall
Berkeley, California 94720
USA

Email: pister@eecs.berkeley.edu

Thomas Watteyne
Linear Technology
32990 Alvarado-Niles Road, Suite 910
Union City, CA 94587
USA

Email: twatteyne@linear.com

6TiSCH Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 13, 2017

M. Vucinic
Inria
J. Simon
Linear Technology
K. Pister
University of California Berkeley
M. Richardson
Sandelman Software Works
March 12, 2017

Minimal Security Framework for 6TiSCH
draft-ietf-6tisch-minimal-security-02

Abstract

This document describes the minimal mechanisms required to support secure enrollment of a pledge, a device being added to an IPv6 over the TSCH mode of IEEE 802.15.4e (6TiSCH) network. It assumes that the pledge has been provisioned with a credential that is relevant to the deployment - the "one-touch" scenario. The goal of this configuration is to set link-layer keys, and to establish a secure end-to-end session between each pledge and the join registrar who may use that to further configure the pledge. Additional security behaviors and mechanisms may be added on top of this minimal framework.

Table of Contents

1. Introduction 3
2. Terminology 3
3. One-Touch Assumptions 4
4. Join Overview 4
4.1. Step 1 - Enhanced Beacon 5
4.2. Step 2 - Neighbor Discovery 6
4.3. Step 3 - Security Handshake 6
4.4. Step 4 - Simple Join Protocol - Join Request 8
4.5. Step 5 - Simple Join Protocol - Join Response 8
5. Architectural Overview and Communication through Join Proxy . 8
5.1. Stateless-Proxy CoAP Option 9
6. Security Handshake 10
6.1. Discovery Message 11
7. Simple Join Protocol Specification 11
7.1. OSCOAP Security Context Instantiation 12
7.2. Specification of Join Request 13
7.3. Specification of Join Response 13
8. Link-layer Requirements 15
9. Asymmetric Keys 15
10. Rekeying and Rejoin 16
11. Key Derivations 16
12. Security Considerations 16
13. Privacy Considerations 17
14. IANA Considerations 18
14.1. CoAP Option Numbers Registry 18
15. Acknowledgments 18
16. References 18
16.1. Normative References 18
16.2. Informative References 19
Appendix A. Example 21
Authors' Addresses 23

1. Introduction

This document describes the minimal feature set for a new device, termed pledge, to securely join a 6TiSCH network. As a successful outcome of this process, the pledge is able to securely communicate with its neighbors, participate in the routing structure of the network or establish a secure session with an Internet host.

When a pledge seeks admission to a 6TiSCH [RFC7554] network, it first needs to synchronize to the network. The pledge then configures its link-local IPv6 address and authenticates itself, and also validates that it is joining the right network. At this point it can expect to interact with the network to configure its link-layer keying material. Only then may the node establish an end-to-end secure session with an Internet host using OSCOAP [I-D.ietf-core-object-security] or DTLS [RFC6347]. Once the application requirements are known, the node interacts with its peers to request additional resources as needed, or to be reconfigured as the network changes [I-D.ietf-6tisch-6top-protocol].

This document presumes a network as described by [RFC7554], [I-D.ietf-6tisch-6top-protocol], and [I-D.ietf-6tisch-terminology]. It assumes the pledge pre-configured with either a:

- pre-shared key (PSK),
- raw public key (RPK),
- or a locally-valid certificate and a trust anchor.

As the outcome of the join process, the pledge expects one or more link-layer key(s) and optionally a temporary network identifier.

2. Terminology

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 [RFC2119]. These words may also appear in this document in lowercase, absent their normative meanings.

The reader is expected to be familiar with the terms and concepts defined in [I-D.ietf-6tisch-terminology], [RFC7252], [I-D.ietf-core-object-security], and [I-D.ietf-anima-bootstrapping-keyinfra]. The following terms are imported: pledge, join proxy, join registrar/coordinator, drop ship, imprint, enrollment, ownership voucher.

Pledge: the prospective device, which has the identity provided to at the factory.

Joined Node: the prospective device, after having completed the join process, often just called a Node.

Join Proxy (JP): a stateless relay that provides connectivity between the pledge and the join registrar/coordinator.

Join Registrar/Coordinator (JRC): central entity responsible for authentication and authorization of joining nodes.

3. One-Touch Assumptions

This document assumes the one-touch scenario, where devices are provided with some mechanism by which a secure association may be made in a controlled environment. There are many ways in which this might be done, and detailing any of them is out of scope for this document. But, some notion of how this might be done is important so that the underlying assumptions can be reasoned about.

Some examples of how to do this could include:

- JTAG interface
- serial (craft) console interface
- pushes of physical buttons simultaneous to network attachment
- unsecured devices operated in a Faraday cage

There are likely many other ways as well. What is assumed is that there can be a secure, private conversation between the Join Registrar/Coordinator, and the pledge, and that the two devices can exchange some trusted bytes of information.

4. Join Overview

This section describes the steps taken by a pledge in a 6TiSCH network. When a previously unknown device seeks admission to a 6TiSCH [RFC7554] network, the following exchange occurs:

1. The pledge listens for an Enhanced Beacon (EB) frame [IEEE8021542015]. This frame provides network synchronization information, and tells the device when it can send a frame to the node sending the beacons, which plays the role of Join Proxy (JP) for the pledge, and when it can expect to receive a frame.

2. The pledge configures its link-local IPv6 address and advertizes it to Join Proxy (JP).

3. The pledge sends packets to JP in order to securely identify itself to the network. These packets are directed to the Join Registrar/Coordinator (JRC), which may be co-located on the JP or another device.

4. The pledge receives one or more packets from JRC (via the JP) that sets up one or more link-layer keys used to authenticate subsequent transmissions to peers.

From the pledge's perspective, minimal joining is a local phenomenon - the pledge only interacts with the JP, and it need not know how far it is from the DAG root, or how to route to the JRC. Only after establishing one or more link-layer keys does it need to know about the particulars of a 6TiSCH network.

The handshake is shown as a transaction diagram in Figure 1:

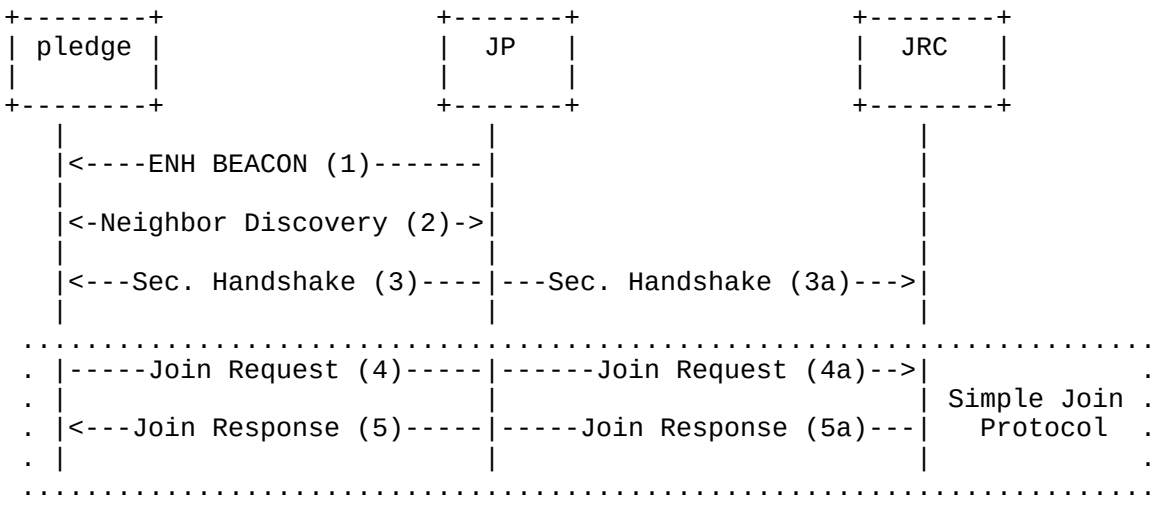


Figure 1: Overview of the join process.

The details of each step are described in the following sections.

4.1. Step 1 - Enhanced Beacon

Due to the channel hopping nature of 6TiSCH, transmissions take place on physical channels in a circular fashion. For that reason, Enhanced Beacons (EBs) are expected to be found by listening on a single channel. However, because some channels may be blacklisted, a

new pledge must listen for Enhanced Beacons for a certain period on each of the 16 possible channels. This search process entails having the pledge keep the receiver portion of its radio active for the entire period of time.

Once the pledge hears an EB from a JP, it synchronizes itself to the joining schedule using the cells contained in the EB. The selection of which beacon to start with is outside the scope of this document. Implementers SHOULD make use of information such as: whether the L2 address of the EB has been tried before, any Network Identifier [I-D.richardson-6tisch-join-enhanced-beacon] seen, and the strength of the signal. The pledge can be configured with the Network Identifier to seek when it is configured with the PSK.

Once a candidate network has been selected, the pledge can transition into a low-power duty cycle, waking up only when the provided schedule indicates shared slots which the pledge may use for the join process.

At this point the pledge may proceed to step 2, or continue to listen for additional EBs.

A pledge which receives only Enhanced Beacons containing Network ID extensions [I-D.richardson-6tisch-join-enhanced-beacon] with the initiate bit cleared, SHOULD NOT proceed with this protocol on that network. The pledge SHOULD consider that it is in a network which manages join traffic, it SHOULD switch to [I-D.ietf-6tisch-dtsecurity-secure-join].

4.2. Step 2 - Neighbor Discovery

At this point, the pledge forms its link-local IPv6 address based on EUI64 and may register it at JP, in order to bootstrap the IPv6 neighbor tables. The Neighbor Discovery exchange shown in Figure 1 refers to a single round trip Neighbor Solicitation / Neighbor Advertisement exchange between the pledge and the JP. The pledge may further follow the Neighbor Discovery (ND) process described in Section 5 of [RFC6775].

4.3. Step 3 - Security Handshake

The security handshake between pledge and JRC uses Ephemeral Diffie-Hellman over COSE (EDHOC) [I-D.selander-ace-cose-ecdhe] to establish the shared session secret used to encrypt the Simple Join Protocol.

The security handshake step is OPTIONAL in case PSKs are used, while it is REQUIRED for RPKs and certificates.

When using certificates, the process continues as described in [I-D.selander-ace-cose-ecdhe], but MAY result in no network key being returned. In that case, the pledge enters a provisional situation where it provides access to an enrollment mechanism described in [I-D.ietf-6tisch-dtsecurity-secure-join].

If using a locally relevant certificate, the pledge will be able to validate the certificate of the JRC via a local trust anchor. In that case, the JRC will return networks keys as in the PSK case. This would typically be the case for a device which has slept so long that it no longer has valid network keys and must go through a partial join process again.

In case the handshake step is omitted, the shared secret used for protection of the Simple Join Protocol in the next step is the PSK.

A consequence is that if the long-term PSK is compromised, keying material transferred as part of the join response is compromised as well. Physical compromise of the pledge, however, would also imply the compromise of the same keying material, as it is likely to be found in node’s memory.

4.3.1. Pre-Shared Symmetric Key

The Diffie-Hellman key exchange and the use of EDHOC is optional, when using a pre-shared symmetric key. This cuts down on traffic between JRC and pledge, but requires pre-configuration of the shared key on both devices.

It is REQUIRED to use unique PSKs for each pledge. If there are multiple JRCs in the network (such as for redundancy), they would have to share a database of PSKs.

4.3.2. Asymmetric Keys

The Security Handshake step is required, when using asymmetric keys. Before conducting the Diffie-Hellman key exchange using EDHOC [I-D.selander-ace-cose-ecdhe] the pledge and JRC need to receive and validate each other’s public key certificate. As detailed above, this can only be done for locally relevant (LDevID) certificates. IDevID certificates require entering a provisional state as described in [I-D.ietf-6tisch-dtsecurity-secure-join].

When RPKs are pre-configured at pledge and JRC, they can directly proceed to the handshake.

4.4. Step 4 - Simple Join Protocol - Join Request

The Join Request that makes part of the Simple Join Protocol is sent from the pledge to the JP using the shared slot as described in the EB, and forwarded to the JRC. Which slot the JP uses to transmit to the JRC is out of scope: some networks may wish to dedicate specific slots for this join traffic.

The join request is typically authenticated/encrypted end-to-end using AES-CCM-16-64-128 algorithm from [I-D.ietf-cose-msg] and a key derived from the shared secret from step 3. Algorithm negotiation is described in detail in [I-D.selander-ace-cose-ecdhe].

The nonce is derived from the shared secret, the pledge's EUI64 and a monotonically increasing counter initialized to 0 when first starting.

4.5. Step 5 - Simple Join Protocol - Join Response

The Join Response that makes part of the Simple Join Protocol is sent from the JRC to the pledge through JP that serves as a stateless relay. Packet containing the Join Response travels on the path from JRC to JP using pre-established routes in the network. The JP delivers it to the pledge using the slot information from the EB. JP operates as the application-layer proxy and does not keep any state to relay the message. It uses information sent in the clear within the join response to decide where to forward to.

The join response is typically authenticated/encrypted using AES-CCM-16-64-128 algorithm from [I-D.ietf-cose-msg] and a key derived from the shared secret from step 3.

The nonce is derived from the shared secret, pledge's EUI64 and a monotonically increasing counter matching that of the join request.

The join response contains one or more link-layer key(s) that the pledge will use for subsequent communication. Each key that is provided by the JRC is associated with an 802.15.4 key identifier. In other link-layer technologies, a different identifier may be substituted. Join Response optionally also contains an IEEE 802.15.4 short address [IEEE8021542015] assigned to pledge by JRC.

5. Architectural Overview and Communication through Join Proxy

The protocol in Figure 1 is implemented over Constrained Application Protocol (CoAP) [RFC7252]. The Pledge plays the role of a CoAP client, JRC the role of a CoAP server, while JP implements CoAP forward proxy functionality [RFC7252]. Since JP is also likely a

constrained device, it does not need to implement a cache but rather process forwarding-related CoAP options and make requests on behalf of pledge that is not yet part of the network.

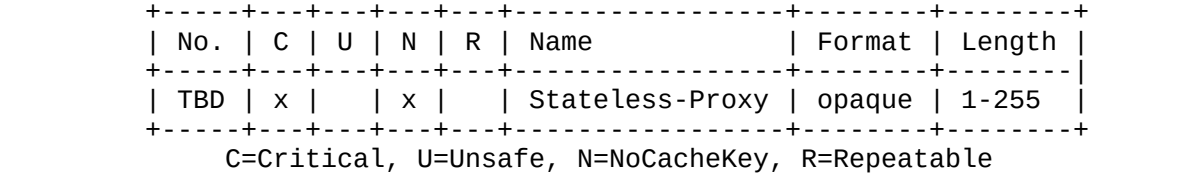
The pledge communicates with a Join Proxy (JP) over link-local IPv6 addresses. The pledge designates a JP as a proxy by including in the CoAP requests to the JP the Proxy-Scheme option with value "coap" (CoAP-to-CoAP proxy). The pledge MUST include the Uri-Host option with its value set to the well-known JRC's alias - "6tisch.arpa". The pledge does not need to learn the actual IPv6 address of JRC at any time during the join protocol. The JP knows the address of the JRC, via a provisioning process that occured when the JP, acting as a pledge, joined. The initial bootstrap of the DODAG root would require explicit provisioning of the JRC address.

5.1. Stateless-Proxy CoAP Option

The CoAP proxy by default keeps per-client state information in order to forward the response towards the originator of the request (client). This state information comprises CoAP token, but the implementations also need to keep track of the IPv6 address of the host, as well as the corresponding UDP source port number. In the setting where the proxy is a constrained device and there are potentially many clients, as in the case of JP, this makes it prone to Denial of Service (DoS) attacks, due to the limited memory.

The Stateless-Proxy CoAP option (c.f. Figure 2) allows the proxy to insert within the request the state information necessary for relaying the response back to the client. Note that the proxy still needs to keep some state, such as for performing congestion control or request retransmission, but what is aimed with Stateless-Proxy option is to free the proxy from keeping per-client state.

Stateless-Proxy option is critical, Safe-to-Forward, not part of the cache key, not repeatable and opaque. When processed by OSCOAP, Stateless-Proxy option is neither encrypted nor integrity protected.



No.	C	U	N	R	Name	Format	Length
TBD	x		x		Stateless-Proxy	opaque	1-255

C=Critical, U=Unsafe, N=NoCacheKey, R=Repeatable

Figure 2: Stateless-Proxy CoAP Option

Upon reception of a Stateless-Proxy option, the CoAP server MUST echo it in the response. The value of the Stateless-Proxy option is

internal proxy state that is opaque to the server. Example state information includes IPv6 address of the client, its UDP source port, and the CoAP token. For security reasons, the state information MUST be authenticated, MUST include a freshness indicator (e.g. a sequence number or timestamp) and MAY be encrypted. The proxy may use an appropriate COSE structure [I-D.ietf-cose-msg] to wrap the state information as the value of the Stateless-Proxy option. The key used for encryption/authentication of the state information may be known only to the proxy.

Once the proxy has received the CoAP response with Stateless-Proxy option present, it decrypts/authenticates it, checks the freshness indicator and constructs the response for the client, based on the information present in the option value.

Note that a CoAP proxy using the Stateless-Proxy option is not able to return 5.04 Gateway Timeout error in case the request to the server times out. Likewise, if the response to the proxy's request does not contain the Stateless-Proxy option, for example when the option is not supported by the server, the proxy is not able to return the response to the client.

6. Security Handshake

In order to derive a shared session key, pledge and JRC run the EDHOC protocol [I-D.selander-ace-cose-ecdhe]. During this process, pledge and JRC mutually authenticate each other and verify authorization information before proceeding with the Simple Join Protocol. In case certificates are used for authentication, this document assumes that a special certificate with role attribute set has been provisioned to the JRC. This certificate is verified by pledge in order to authorize JRC to continue with the join process. How such a certificate is issued to the JRC is out of scope of this document.

Figure 3 details the exchanges between the pledge and JRC that take place during the execution of the security handshake. Format of EDHOC messages is specified in [I-D.selander-ace-cose-ecdhe].

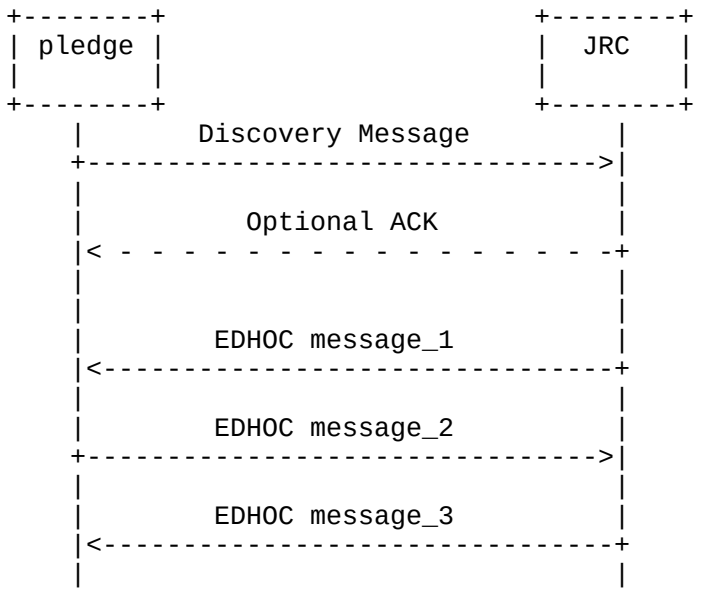


Figure 3: Transaction diagram of the security handshake.

6.1. Discovery Message

Pledge triggers the security handshake by sending a discovery message to the JRC. This initial message does not make part of the EDHOC handshake. JRC is the initiator of the EDHOC run and is able to schedule the execution of many concurrent enrollments at will by acknowledging the request and sending a separate, delayed response. The Discovery Message SHALL be mapped to a CoAP request:

- The request method is POST.
- The Proxy-Scheme option is set to "coap".
- The Uri-Host option is set to "6tisch.arpa".
- The Uri-Path option is set to "edhoc".
- The payload is optional and contains pledge’s EUI-64.

7. Simple Join Protocol Specification

Simple Join Protocol is a single round trip protocol (c.f. Figure 4) that facilitates secure enrollment of a pledge, based on a shared symmetric secret. In case the pledge was provisioned by an

asymmetric key (certificate or RPK), Simple Join Protocol is preceded by a security handshake, described in Section 6. When the pledge is provisioned with a PSK, Simple Join Protocol may be run directly.

Pledge and JRC MUST protect their exchange end-to-end (i.e. through the proxy) using Object Security of CoAP (OSCOAP) [I-D.ietf-core-object-security].

Figure 4: Transaction diagram of the Simple Join Protocol.

7.1. OSCOAP Security Context Instantiation

The OSCOAP security context MUST be derived at pledge and JRC as per Section 3.2 of [I-D.ietf-core-object-security] using HKDF [RFC5869] as the key derivation function.

- Master Secret MUST be the secret generated by the run of EDHOC as per Appendix B of [I-D.selander-ace-cose-ecdhe], or the PSK in case EDHOC step was omitted.
- Sender ID of the pledge MUST be set to the concatenation of its EUI-64 and byte string 0x00.
- Recipient ID (ID of JRC) MUST be set to the concatenation of pledge’s EUI-64 and byte string 0x01. The construct uses pledge’s EUI-64 to avoid nonce reuse in the response in the case same PSK is shared by a group of pledges.
- Algorithm MUST be set to AES-CCM-16-64-128 from [I-D.ietf-cose-msg]. CoAP messages are therefore protected with an 8-byte CCM authentication tag and the algorithm uses 13-byte long nonces.

The hash algorithm that instantiates HKDF MUST be SHA-256 [RFC4231]. The derivation in [I-D.ietf-core-object-security] results in traffic

keys and static IVs for each side of the conversation. Nonces are constructed by XOR'ing the static IV with current sequence number. The context derivation process occurs exactly once.

Implementations MUST ensure that multiple CoAP requests to different JRCs result in the use of the same OSCOAP context so that sequence numbers are properly incremented for each request. This may happen in a scenario where there are multiple 6TiSCH networks present and the pledge tries to join one network at a time.

7.2. Specification of Join Request

Message Join Request SHALL be mapped to a CoAP request:

- The request method is GET.
- The Proxy-Scheme option is set to "coap".
- The Uri-Host option is set to "6tisch.arpa".
- The Uri-Path option is set to "j".
- The object security option SHALL be set according to [I-D.ietf-core-object-security] and OSCOAP parameters set as described above.

7.3. Specification of Join Response

If OSCOAP processing is a success and the pledge is authorized to join the network, message Join Response SHALL be mapped to a CoAP response:

- The response Code is 2.05 (Content).
- Content-Format option is set to application/cbor.
- The payload is a CBOR [RFC7049] array containing, in order:
  - COSE Key Set, specified in [I-D.ietf-cose-msg], containing one or more link-layer keys. The mapping of individual keys to 802.15.4-specific parameters is described in Section 7.3.1.
  - Optional short address that is assigned to the pledge. The format of the short address follows Section 7.3.2.

```
payload = [
    COSE_KeySet,
    ? short_address,
]
```

7.3.1. Link-layer Keys Transported in COSE Key Set

Each key in the COSE Key Set [I-D.ietf-cose-msg] SHALL be a symmetric key. If "kid" parameter of the COSE Key structure is present, the corresponding keys SHALL belong to an IEEE 802.15.4 KeyIdMode 0x01 class. In that case, parameter "kid" of COSE Key structure SHALL be used to carry IEEE 802.15.4 KeyIndex value. If the "kid" parameter is not present in the transported key, the application SHALL consider the key to be an IEEE 802.15.4 KeyIdMode 0x00 (implicit) key. This document does not support IEEE 802.15.4 KeyIdMode 0x02 and 0x03 class keys.

7.3.2. Short Address

Optional "short_address" structure transported as part of the join response payload represents IEEE 802.15.4 short address assigned to the pledge. It is encoded as CBOR array object, containing in order:

- Byte string, containing the 16-bit address.

- Optional lease time parameter, "lease_asn". The value of the "lease_asn" parameter is the 5-byte Absolute Slot Number (ASN) corresponding to its expiration, carried as a byte string in network byte order.

```
short_address = [
    address : bstr,
    ? lease_asn : bstr,
]
```

It is up to the joined node to request a new short address before the expiry of its previous address. The mechanism by which the node requests renewal is the same as during join procedure, as described in Section 10. The assigned short address is used for configuring both Layer 2 short address and Layer 3 addresses.

7.3.3. Error Handling

In the case JRC determines that pledge is not supposed to join the network (e.g. by failing to find an appropriate security context), it should respond with a 4.01 Unauthorized error. Upon reception of a 4.01 Unauthorized, the pledge SHALL attempt to join the next advertised 6TiSCH network. If all join attempts have failed at

pledge, the pledge SHOULD signal to the user by an out-of-band mechanism the presence of an error condition.

In the case that the JRC determines that the pledge is not (yet) authorized to join the network, but a further zero-touch process might permit it, the JRC responds with a 2.05 (Content) code, but the payload contains the single CBOR string "prov" (for "provisional"). No link-layer keys or short address is returned.

This response is typically only expected when in asymmetric certificate mode using 802.1AR IDevID certificates. But for reasons of provisioning or device reuse, this could occur even when a one-touch PSK authentication process was expected.

8. Link-layer Requirements

In an operational 6TiSCH network, all frames MUST use link-layer frame security. The frame security options MUST include frame authentication, and MAY include frame encryption.

Link-layer frames are protected with a 16-byte key, and a 13-byte nonce constructed from current Absolute Slot Number (ASN) and the source (the JP for EBs) address, as shown in Figure 5:

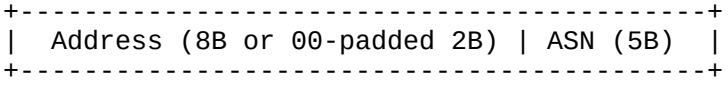


Figure 5: Link-layer CCM* nonce construction

The pledge does not initially do any authentication of the EB frames, as it does not know the K1 key. When sending frames, the pledge sends unencrypted and unauthenticated frames. JP accepts these frames (exempt mode in 802.15.4) for the duration of the join process. How JP learns whether the join process is ongoing is out of scope of this specification.

As the EB itself cannot be authenticated by pledge, an attacker may craft a frame that appears to be a valid EB, since the pledge can neither know the ASN a priori nor verify the address of the JP. This permits a Denial of Service (DoS) attack at the pledge. Beacon authentication keys are discussed in [I-D.ietf-6tisch-minimal].

9. Asymmetric Keys

Certificates or pre-configured RPKs may be used to exchange public keys between the pledge and JRC. The key pair is generated using

elliptic curve secp256r1, and the certificate containing the public key is signed using ECDSA. (XXX: would be nice to move to EdDSA)

The certificate itself may be a compact representation of an X.509 certificate, or a full X.509 certificate. Compact representation of X.509 certificates is out of scope of this specification. The certificate is signed by a root CA whose certificate is installed on all nodes participating in a particular 6TiSCH network, allowing each node to validate the certificate of the JRC or pledge as appropriate.

10. Rekeying and Rejoin

This protocol handles initial keying of the pledge. For reasons such as rejoining after a long sleep, or expiry of the short address, the joined node MAY send a new Join Request over the previously established secure end-to-end session with JRC. JRC responds with up-to-date keys and a short address. The node may also use the Simple Join Protocol exchange for node-initiated rekeying. How node learns that it should be rekeyed is out of scope. Additional work, such as in [I-D.richardson-6tisch-minimal-rekey] can be used.

11. Key Derivations

When EDHOC is used to derive keys, the cost of the asymmetric operation can be amortized over any additional connections that may be required between the node (during or after joining) and the JRC.

Each application SHOULD use a unique session key. EDHOC was designed with this in mind. In order to accomplish this, the EDHOC key derivation algorithm can be run with a different label. Other users of this key MUST define the label.

12. Security Considerations

In case PSKs are used, this document mandates that the pledge and JRC are pre-configured with unique keys. The uniqueness of generated nonces is guaranteed under the assumption of unique EUI64 identifiers for each pledge. Note that the address of the JRC does not take part in nonce construction. Therefore, even should an error occur, and a PSK shared by a group of nodes, the nonces constructed as part of the different responses are unique. The PSK is still important for authentication of the pledge and authentication of the JRC to the pledge. Should an attacker come to know the PSK, then a man-in-the-middle attack is possible. The well known problem with Bluetooth headsets with a "0000" pin applies here. The design differentiates between nonces constructed for requests and nonces constructed for responses by different sender identifiers (0x00 for pledge and 0x01 for JRC).

Being a stateless relay, JP blindly forwards the join traffic into the network. While the exchange between pledge and JP takes place over a shared cell, join traffic is forwarded using dedicated cells on the JP to JRC path. In case of distributed scheduling, the join traffic may therefore cause intermediate nodes to request additional bandwidth. (EDNOTE: this is a problem that needs to be solved) Because the relay operation of JP is implemented at the application layer, JP is the only hop on the JP-6LBR path that can distinguish join traffic from regular IP traffic in the network. It is therefore recommended to implement stateless rate limiting at JP: a simple bandwidth (in bytes or packets/second) cap would be appropriate.

The shared nature of the "minimal" cell used for join traffic makes the network prone to DoS attacks by congesting the JP with bogus radio traffic. As such an attacker is limited by emitted radio power, redundancy in the number of deployed JPs alleviates the issue and also gives the pledge a possibility to use the best available link for join. How a network node decides to become a JP is out of scope of this specification.

At the time of the join, the pledge has no means of verifying the content in the EB and has to accept it at "face value". In case the pledge tries to join an attacker’s network, the join response message in such cases will either fail the security check or time out. The pledge may implement a blacklist in order to filter out undesired beacons and try to join the next seemingly valid network. The blacklist alleviates the issue but is effectively limited by the node’s available memory. Such bogus beacons will prolong the join time of the pledge and so the time spent in "minimal" [I-D.ietf-6tisch-minimal] duty cycle mode.

13. Privacy Considerations

This specification relies on the uniqueness of EUI64 that is transferred in clear as part of the security context identifier. (EDNOTE: should we say IID here?) Privacy implications of using such long-term identifier are discussed in [RFC7721] and comprise correlation of activities over time, location tracking, address scanning and device-specific vulnerability exploitation. Since the join protocol is executed rarely compared to the network lifetime, long-term threats that arise from using EUI64 are minimal. In addition, the join response message contains an optional short address which can be assigned by JRC to the pledge. The short address is independent of the long-term identifier EUI64 and is encrypted in the response. For that reason, it is not possible to correlate the short address with the EUI64 used during the join. Use of short addresses once the join protocol completes mitigates the aforementioned privacy risks. In addition, EDHOC may be used for

identity protection during the join protocol by generating a random context identifier in place of the EUI64 [I-D.selander-ace-cose-ecdhe].

14. IANA Considerations

Note to RFC Editor: Please replace all occurrences of "[[this document]]" with the RFC number of this specification.

This document allocates a well known name under the .arpa name space according to the rules given in: [RFC3172]. The name "6tisch.arpa" is requested. No subdomains are expected. No A, AAAA or PTR record is requested.

14.1. CoAP Option Numbers Registry

The Stateless-Proxy option is added to the CoAP Option Numbers registry:

Number	Name	Reference
TBD	Stateless-Proxy	[[this document]]

15. Acknowledgments

The work on this document has been partially supported by the European Union’s H2020 Programme for research, technological development and demonstration under grant agreement No 644852, project ARMOUR.

The authors are grateful to Thomas Watteyne and Goeran Selander for reviewing the draft. The authors would also like to thank Francesca Palombini and Ludwig Seitz for participating in the discussions that have helped shape the document.

16. References

16.1. Normative References

[I-D.ietf-core-object-security]
Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security of CoAP (OSCOAP)", draft-ietf-core-object-security-01 (work in progress), December 2016.

[I-D.ietf-cose-msg]
Schaad, J., "CBOR Object Signing and Encryption (COSE)", draft-ietf-cose-msg-24 (work in progress), November 2016.

[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>.

[RFC3172] Huston, G., Ed., "Management Guidelines & Operational Requirements for the Address and Routing Parameter Area Domain ("arpa")", BCP 52, RFC 3172, DOI 10.17487/RFC3172, September 2001, <http://www.rfc-editor.org/info/rfc3172>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <http://www.rfc-editor.org/info/rfc7049>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <http://www.rfc-editor.org/info/rfc7252>.

16.2. Informative References

[I-D.ietf-6tisch-6top-protocol]
Wang, Q. and X. Vilajosana, "6top Protocol (6P)", draft-ietf-6tisch-6top-protocol-03 (work in progress), October 2016.

[I-D.ietf-6tisch-dtsecurity-secure-join]
Richardson, M., "6tisch Secure Join protocol", draft-ietf-6tisch-dtsecurity-secure-join-01 (work in progress), February 2017.

[I-D.ietf-6tisch-minimal]
Vilajosana, X., Pister, K., and T. Watteyne, "Minimal 6TiSCH Configuration", draft-ietf-6tisch-minimal-21 (work in progress), February 2017.

[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-08 (work in progress), December 2016.

[I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., Bjarnason, S., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-04 (work in progress), October 2016.

[I-D.richardson-6tisch-join-enhanced-beacon]
Dujovne, D. and M. Richardson, "IEEE802.15.4 Informational Element encapsulation of 6tisch Join Information", draft-richardson-6tisch-join-enhanced-beacon-01 (work in progress), March 2017.

[I-D.richardson-6tisch-minimal-rekey]
Richardson, M., "Minimal Security rekeying mechanism for 6TiSCH", draft-richardson-6tisch-minimal-rekey-01 (work in progress), February 2017.

[I-D.selander-ace-cose-ecdhe]
Selander, G., Mattsson, J., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", draft-selander-ace-cose-ecdhe-04 (work in progress), October 2016.

[IEEE8021542015]
IEEE standard for Information Technology, ., "IEEE Std 802.15.4-2015 Standard for Low-Rate Wireless Personal Area Networks (WPANs)", 2015.

[RFC4231] Nystrom, M., "Identifiers and Test Vectors for HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512", RFC 4231, DOI 10.17487/RFC4231, December 2005, <http://www.rfc-editor.org/info/rfc4231>.

[RFC5869] Krawczyk, H. and P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC 5869, DOI 10.17487/RFC5869, May 2010, <http://www.rfc-editor.org/info/rfc5869>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <http://www.rfc-editor.org/info/rfc6347>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <http://www.rfc-editor.org/info/rfc6775>.

[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>.

[RFC7721] Cooper, A., Gont, F., and D. Thaler, "Security and Privacy Considerations for IPv6 Address Generation Mechanisms", RFC 7721, DOI 10.17487/RFC7721, March 2016, <http://www.rfc-editor.org/info/rfc7721>.

Appendix A. Example

Figure 6 illustrates a join protocol exchange in case PSKs are used. Pledge instantiates the OSCOAP context and derives the traffic keys and nonces from the PSK. It uses the instantiated context to protect the CoAP request addressed with Proxy-Scheme option and well-known host name of JRC in the Uri-Host option. The example assumes a JP that is already aware of JRC's IPv6 address and does not need to resolve the well-known "6tisch.arpa" host name. Triggered by the presence of Proxy-Scheme option, JP forwards the request to the JRC and adds the Stateless-Proxy option with value set to the internally needed state, authentication tag, and a freshness indicator. Once JRC receives the request, it looks up the correct context based on the Sender ID (sid) parameter. It reconstructs OSCOAP's external Additional Authenticated Data (AAD) needed for verification based on:

- Version field of the received CoAP header.
- Code field of the received CoAP header.
- Algorithm being the AES-CCM-16-64-128 from [I-D.ietf-cose-msg].
- Request ID being set to pledge's EUI-64 concatenated with 0x00.
- Request Sequence number set to the value of "Partial IV" of the received COSE object.

Replay protection is ensured by OSCOAP and the tracking of sequence numbers at each side. In the example below, the response contains sequence number 7 meaning that there have already been some attempts to join under a given context, not coming from the pledge. Once JP receives the response, it authenticates the Stateless-Proxy option before deciding where to forward. JP sets its internal state to that found in the Stateless-Proxy option. Note that JP does not posses the key to decrypt the COSE object present in the payload so the join_response object is opaque to it. The response is matched to the

request and verified for replay protection at pledge using OSCOAP processing rules.

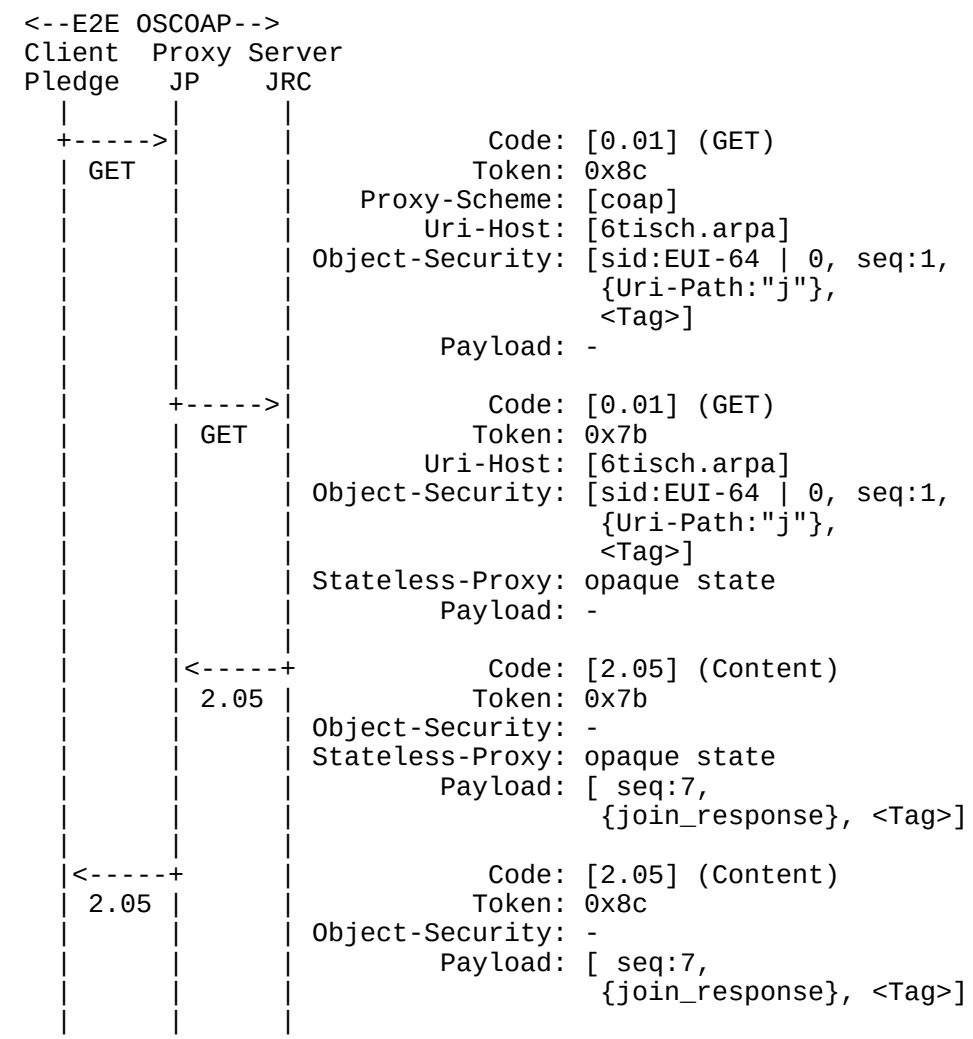


Figure 6: Example of a join protocol exchange with a PSK. {} denotes encryption and authentication, [] denotes authentication.

Where join_response is as follows.

```
join_response:
[
    [   / COSE Key Set array with a single key /
        {
             1 : 4, / key type symmetric /
             2 : h'01', / key id /
            -1 : h'e6bf4287c2d7618d6a9687445ffd33e6' / key value /
        }
    ],
    [
        h'af93' / assigned short address /
    ]
]
```

Encodes to h'8281a301040241012050e6bf4287c2d7618d6a9687445ffd33e68142af93' with a size of 30 bytes.

Authors' Addresses

Malisa Vucinic
Inria
2 Rue Simone Iff
Paris 75012
France

Email: malisa.vucinic@inria.fr

Jonathan Simon
Linear Technology
32990 Alvarado-Niles Road, Suite 910
Union City, CA 94587
USA

Email: jsimon@linear.com

Kris Pister
University of California Berkeley
512 Cory Hall
Berkeley, CA 94720
USA

Email: pister@eecs.berkeley.edu

Michael Richardson
Sandelman Software Works
470 Dawson Avenue
Ottawa, ON K1Z5V7
Canada

Email: mcr+ietf@sandelman.ca

6TiSCH Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 6, 2018

M. Vucinic, Ed.
University of Montenegro
J. Simon
Analog Devices
K. Pister
University of California Berkeley
M. Richardson
Sandelman Software Works
March 05, 2018

Minimal Security Framework for 6TiSCH
draft-ietf-6tisch-minimal-security-05

Abstract

This document describes the minimal framework required for a new device, called "pledge", to securely join a 6TiSCH (IPv6 over the TSCH mode of IEEE 802.15.4e) network. The framework requires that the pledge and the JRC (join registrar/coordinator, a central entity), share a symmetric key. How this key is provisioned is out of scope of this document. Through a single CoAP (Constrained Application Protocol) request-response exchange secured by OSCORE (Object Security for Constrained RESTful Environments), the pledge requests admission into the network and the JRC configures it with link-layer keying material and a short link-layer address. This specification defines the message format, a new Stateless-Proxy CoAP option, and configures the rest of the 6TiSCH communication stack for this join process to occur in a secure manner. Additional security mechanisms may be added on top of this minimal framework.

Table of Contents

1. Introduction 3
2. Terminology 4
3. Identifiers 4
4. One-Touch Assumption 5
5. Join Overview 5
5.1. Step 1 - Enhanced Beacon 7
5.2. Step 2 - Neighbor Discovery 7
5.3. Step 3 - Join Request 8
5.4. Step 4 - Join Response 8
6. Link-layer Configuration 9
7. Network-layer Configuration 9
7.1. Identification of Join Request Traffic 10
7.2. Identification of Join Response Traffic 11
8. Application-level Configuration 11
8.1. OSCORE Security Context 12
9. 6TiSCH Join Protocol 13
9.1. Specification of the Join Request 14
9.2. Specification of the Join Response 15
9.3. Error Handling and Retransmission 17
9.4. Rekeying and Rejoining 18
9.5. Parameters 18
9.6. Mandatory to Implement Algorithms 18
10. Stateless-Proxy CoAP Option 19
11. Security Considerations 20
12. Privacy Considerations 21
13. IANA Considerations 21
13.1. CoAP Option Numbers Registry 21
14. Acknowledgments 22
15. References 22
15.1. Normative References 22
15.2. Informative References 23

Appendix A. Example 24
Authors' Addresses 27

1. Introduction

This document presumes a 6TiSCH network as described by [RFC7554] and [RFC8180]. By design, nodes in a 6TiSCH network [RFC7554] have their radio turned off most of the time, to conserve energy. As a consequence, the link used by a new device for joining the network has limited bandwidth [RFC8180]. The secure join solution defined in this document therefore keeps the number of over-the-air exchanges for join purposes to a minimum.

The micro-controllers at the heart of 6TiSCH nodes have a small amount of code memory. It is therefore paramount to reuse existing protocols available as part of the 6TiSCH stack. At the application layer, the 6TiSCH stack already relies on CoAP [RFC7252] for web transfer, and on OSCORE [I-D.ietf-core-object-security] for its end-to-end security. The secure join solution defined in this document therefore reuses those two protocols as its building blocks.

This document defines a secure join solution for a new device, called "pledge", to securely join a 6TiSCH network. The specification defines a 6TiSCH Join Protocol (6JP) used by the pledge to request admission into a network managed by the JRC, and for the JRC to configure the pledge with the necessary parameters, a new CoAP option, and configures different layers of the 6TiSCH protocol stack for the join process to occur in a secure manner.

It assumes the presence of a JRC (join registrar/coordinator), a central entity. It further assumes that the pledge and the JRC share a symmetric key, called PSK (pre-shared key). The PSK is used to configure OSCORE to provide a secure channel to 6JP. How the PSK is installed is out of scope of this document.

When the pledge seeks admission to a 6TiSCH network, it first synchronizes to it, by initiating the passive scan defined in [IEEE802.15.4-2015]. The pledge then exchanges messages with the JRC; these messages can be forwarded by nodes already part of the 6TiSCH network. The messages exchanged allow the JRC and the pledge to mutually authenticate, based on the PSK. They also allow the JRC to configure the pledge with link-layer keying material and a short link-layer address. After this secure join process successfully completes, the joined node can interact with its neighbors to request additional bandwidth using the 6top Protocol [I-D.ietf-6tisch-6top-protocol] and start sending the application traffic.

2. Terminology

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 [RFC2119]. These words may also appear in this document in lowercase, absent their normative meanings.

The reader is expected to be familiar with the terms and concepts defined in [I-D.ietf-6tisch-terminology], [RFC7252], [I-D.ietf-core-object-security], and [RFC8152].

The specification also includes a set of informative examples using the CBOR diagnostic notation [I-D.ietf-cbor-cddl].

The following terms defined in [I-D.ietf-6tisch-terminology] are used extensively throughout this document:

- pledge
- joined node
- join proxy (JP)
- join registrar/coordinator (JRC)
- enhanced beacon (EB)
- join protocol
- join process

In addition, we use the generic terms "network identifier" and "pledge identifier". See Section 3.

3. Identifiers

The "network identifier" uniquely identifies the 6TiSCH network in the namespace managed by a JRC. Typically, this is the 16-bit Personal Area Network Identifier (PAN ID) defined in [IEEE802.15.4-2015]. Companion documents can specify the use of a different network identifier for join purposes, but this is out of scope of this specification. Such identifier needs to be carried within Enhanced Beacon (EB) frames.

The "pledge identifier" uniquely identifies the pledge in the namespace managed by a JRC. The pledge identifier is typically the globally unique 64-bit Extended Unique Identifier (EUI-64) of the

IEEE 802.15.4 device. This identifier is used to generate the IPv6 addresses of the pledge and to identify it during the execution of the join protocol. For privacy reasons, it is possible to use an identifier different from the EUI-64 (e.g. a random string). See Section 12.

4. One-Touch Assumption

This document assumes a one-touch scenario. The pledge is provisioned with certain parameters before attempting to join the network, and the same parameters are provisioned to the JRC.

There are many ways by which this provisioning can be done. Physically, the parameters can be written into the pledge using a number of mechanisms, such as a JTAG interface, a serial (craft) console interface, pushing buttons simultaneously on different devices, over-the-air configuration in a Faraday cage, etc. The provisioning can be done by the vendor, the manufacturer, the integrator, etc.

Details of how this provisioning is done is out of scope of this document. What is assumed is that there can be a secure, private conversation between the JRC and the pledge, and that the two devices can exchange the parameters.

Parameters that are provisioned to the pledge include:

- Pre-Shared Key (PSK). The JRC additionally needs to store the identifier of the pledge bound to the given PSK. The PSK SHOULD be at least 128 bits in length, generated uniformly at random. It is RECOMMENDED to generate the PSK with a cryptographically secure pseudorandom number generator. Each pledge SHOULD be provisioned with a unique PSK.

- Optionally, a network identifier. Provisioning the network identifier to the pledge is RECOMMENDED, as it significantly speeds up the join process. In case this parameter is not provisioned, the pledge attempts to join one network at a time.

- Optionally, any non-default algorithms. Mandatory to implement and default algorithms are specified in Section 9.6.

5. Join Overview

This section describes the steps taken by a pledge in a 6TiSCH network. When a pledge seeks admission to a 6TiSCH network, the following exchange occurs:

1. The pledge listens for an Enhanced Beacon (EB) frame [IEEE802.15.4-2015]. This frame provides network synchronization information, and tells the device when it can send a frame to the node sending the beacons, which plays the role of join proxy (JP) for the pledge, and when it can expect to receive a frame.

2. The pledge configures its link-local IPv6 address and advertises it to the join proxy (JP).

3. The pledge sends a Join Request to the JP in order to securely identify itself to the network. The Join Request is directed to the JRC, which may be co-located on the JP or another device.

4. In case of successful processing of the request, the pledge receives a join response from JRC (via the JP) that sets up one or more link-layer keys used to authenticate and encrypt subsequent transmissions to peers, and a short link-layer address for the pledge.

From the pledge's perspective, joining is a local phenomenon - the pledge only interacts with the JP, and it needs not know how far it is from the 6LBR, or how to route to the JRC. Only after establishing one or more link-layer keys does it need to know about the particulars of a 6TiSCH network.

The join process is shown as a transaction diagram in Figure 1:

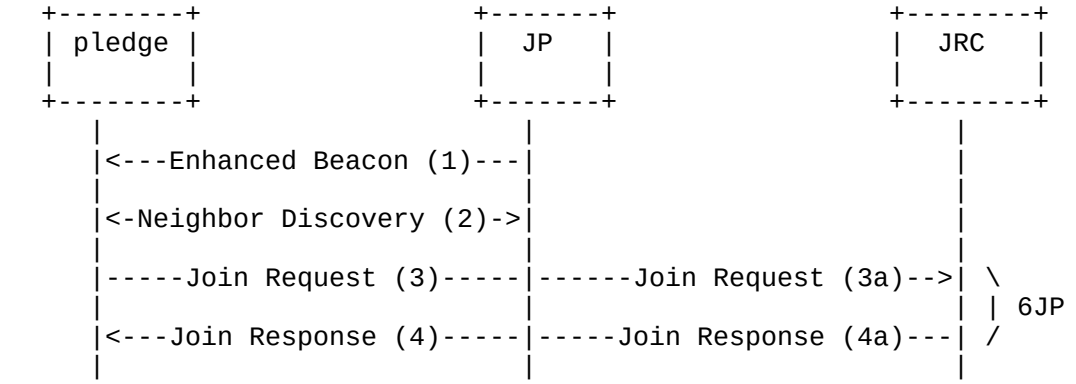


Figure 1: Overview of a successful join process. 6JP stands for 6TiSCH Join Protocol.

The details of each step are described in the following sections.

5.1. Step 1 - Enhanced Beacon

The pledge synchronizes to the network by listening for, and receiving, an Enhanced Beacon (EB) sent by a node already in the network. This process is entirely defined by [IEEE802.15.4-2015], and described in [RFC7554].

Once the pledge hears an EB, it synchronizes to the joining schedule using the cells contained in the EB. The pledge can hear multiple EBs; the selection of which EB to use is out of the scope for this document, and is discussed in [RFC7554]. Implementers should make use of information such as: what network identifier the EB contains, whether the source link-layer address of the EB has been tried before, what signal strength the different EBs were received at, etc. In addition, the pledge may be pre-configured to search for EBs with a specific network identifier.

If the pledge is not provisioned with the network identifier, it attempts to join one network at a time, as described in Section 9.3.

Once the pledge selects the EB, it synchronizes to it and transitions into a low-power mode. It deeply duty cycles its radio, switching the radio on when the provided schedule indicates slots which the pledge may use for the join process. During the remainder of the join process, the node that has sent the EB to the pledge plays the role of JP.

At this point, the pledge may proceed to step 2, or continue to listen for additional EBs.

5.2. Step 2 - Neighbor Discovery

The pledge forms its link-local IPv6 address based on the interface identifier, as per [RFC4944]. The pledge MAY perform the Neighbor Solicitation / Neighbor Advertisement exchange with the JP, as per Section 5.5.1 of [RFC6775]. The pledge and the JP use their link-local IPv6 addresses for all subsequent communication during the join process.

Note that Neighbor Discovery exchanges at this point are not protected with link-layer security as the pledge is not in possession of the keys. How JP accepts these unprotected frames is discussed in Section 6.

5.3. Step 3 - Join Request

The Join Request is a message sent from the pledge to the JP, and which the JP forwards to the JRC. The JP forwards the Join Request to the JRC on the existing 6TiSCH network. How exactly this happens is out of scope of this document; some networks may wish to dedicate specific slots for this join traffic.

The Join Request is authenticated/encrypted end-to-end using an AEAD (Authenticated Encryption with Associated Data) algorithm from [RFC8152] and a key derived from the PSK, the pledge identifier and a request-specific constant value. Algorithms which MUST be implemented are specified in Section 9.6.

The nonce used when securing the Join Request is derived from the PSK, the pledge identifier and a monotonically increasing counter initialized to 0 when first starting.

Join Request message is specified in Section 9.1, while the details on security processing can be found in Section 7 of [I-D.ietf-core-object-security].

5.4. Step 4 - Join Response

The Join Response is sent by the JRC to the pledge, and is forwarded through the JP as it serves as a stateless relay. The packet containing the Join Response travels from the JRC to JP using the operating routes in the 6TiSCH network. The JP delivers it to the pledge. The JP operates as the application-layer proxy, and does not keep any state to relay the message. It uses information sent in the clear within the Join Response to decide where to forward to.

The Join Response is authenticated/encrypted end-to-end using an AEAD algorithm from [RFC8152]. The key used to protect the response is different from the one used to protect the request (both are derived from the PSK, as explained in Section 8.1). The response is protected using the same nonce as in the request.

The Join Response contains one or more link-layer key(s) that the pledge will use for subsequent communication. Each key that is provided by the JRC is associated with an 802.15.4 key identifier. In other link-layer technologies, a different identifier may be substituted. The Join Response also contains an IEEE 802.15.4 short address [IEEE802.15.4-2015] assigned by the JRC to the pledge, and optionally the IPv6 address of the JRC.

Join Response message is specified in Section 9.2, while the details on security processing can be found in Section 7 of [I-D.ietf-core-object-security].

6. Link-layer Configuration

In an operational 6TiSCH network, all frames MUST use link-layer frame security [RFC8180]. The IEEE 802.15.4 security attributes MUST include frame authenticity, and MAY include frame confidentiality (i.e. encryption).

As specified in [RFC8180], the network uses a key termed as K1 to authenticate EBs and a key termed as K2 to authenticate and optionally encrypt DATA and ACKNOWLEDGMENT frames. The keys K1 and K2 MAY be the same key (same value and IEEE 802.15.4 index). How the JRC communicates these keys to 6LBR is out of scope of this specification.

The pledge does not initially do any authenticity check of the EB frames, as it does not know the K1 key. The pledge is still able to parse the contents of the received EBs and synchronize to the network, as EBs are not encrypted [RFC8180].

When sending frames during the join process, the pledge sends unencrypted and unauthenticated frames. The JP accepts these frames (using the "exempt mode" in 802.15.4) for the duration of the join process. How the JP learns whether the join process is ongoing is out of scope of this specification.

As the EB itself cannot be authenticated by the pledge, an attacker may craft a frame that appears to be a valid EB, since the pledge can neither know the ASN a priori nor verify the address of the JP. This opens up a possibility of DoS attack, as discussed in Section 11. Beacon authentication keys are discussed in [RFC8180].

7. Network-layer Configuration

The pledge and the JP SHOULD keep a separate neighbor cache for untrusted entries and use it to store each other’s information during the join process. Mixing neighbor entries belonging to pledges and nodes that are part of the network opens up the JP to a DoS attack. How the pledge and the JP decide to transition each other from untrusted to trusted cache, once the join process completes, is out of scope. One implementation technique is to use the information whether the incoming frames are secured at the link layer.

The pledge does not communicate with the JRC at the network layer. This allows the pledge to join without knowing the IPv6 address of

the JRC. Instead, the pledge communicates with the JP at the network layer, and with the JRC at the application layer, as specified in Section 8.

The JP communicates with the JRC over global IPv6 addresses. The JP discovers the network prefix and configures its global IPv6 address upon successful completion of the join process and the obtention of link-layer keys. The pledge learns the actual IPv6 address of the JRC from the Join Response, as specified in Section 9.2; it uses it once joined in order to operate as a JP.

The JRC can be co-located on the 6LBR. In this special case, the IPv6 address of the JRC can be omitted from the Join Response message for space optimization. The 6LBR then MUST set the DODAGID field in RPL DIOs [RFC6550] to its IPv6 address. The pledge learns the address of the JRC once joined and upon the reception of a first RPL DIO message, and uses it to operate as a JP.

Before the 6TiSCH network is started, the 6LBR MUST be provisioned with the IPv6 address of the JRC.

7.1. Identification of Join Request Traffic

The join request traffic that is proxied by the Join Proxy comes from unauthenticated nodes, and there may be an arbitrary amount of it. In particular, an attacker may send fraudulent traffic in attempt to overwhelm the network.

When operating as part of a [RFC8180] 6TiSCH minimal network using reasonable scheduling algorithms, the join request traffic present may cause intermediate nodes to request additional bandwidth. An attacker could use this property to cause the network to overcommit bandwidth (and energy) to the join process.

The Join Proxy is aware of what traffic is join request traffic, and so can avoid allocating additional bandwidth itself. The Join Proxy SHOULD implement a bandwidth cap on outgoing join request traffic. This cap will not protect intermediate nodes as they can not tell join request traffic from regular traffic. Despite the bandwidth cap implemented separately on each Join Proxy, the aggregate join request traffic from many Join Proxies may cause intermediate nodes to decide to allocate additional cells. It is undesirable to to so in response to the join request traffic. In order to permit the intermediate nodes to avoid this, the traffic needs to be tagged in some way.

[RFC2597] defines a set of per-hop behaviors that may be encoded into the Diffserv Code Points (DSCPs). The Join Proxy SHOULD set the DSCP

of join request packets that it produces as part of the relay process to AF43 code point (See Section 6 of [RFC2597]).

A Join Proxy that does not set the DSCP on traffic forwarded should set it to zero so that it is compressed out.

A Scheduling Function (SF) running on 6TiSCH nodes SHOULD NOT allocate additional cells as a result of traffic with code point AF43. Companion SF documents SHOULD specify how this recommended behavior is achieved.

7.2. Identification of Join Response Traffic

The JRC SHOULD set the DSCP of join response packets addressed to the Join Proxy to AF42 code point. Join response traffic can not be induced by an attacker as it is generated only in response to legitimate pledges (see Section 9.3). AF42 has lower drop probability than AF43, giving join response traffic priority in buffers over join request traffic.

When the JRC is not co-located with the 6LBR, then the code point provides a clear indication to the 6LBR that this is join response traffic.

Due to the convergecast nature of the DODAG, the 6LBR links are often the most congested, and from that point down there is progressively less (or equal) congestion. If the 6LBR paces itself when sending join response traffic then it ought to never exceed the bandwidth allocated to the best effort traffic cells. If the 6LBR has the capacity (if it is not constrained) then it should provide some buffers in order to satisfy the Assured Forwarding behavior.

Companion SF documents SHOULD specify how traffic with code point AF42 is handled with respect to cell allocation.

8. Application-level Configuration

The Join Request/Join Response exchange in Figure 1 is carried over CoAP [RFC7252] and secured using OSCORE [I-D.ietf-core-object-security]. The pledge plays the role of a CoAP client; the JRC plays the role of a CoAP server. The JP implements CoAP forward proxy functionality [RFC7252]. Because the JP can also be a constrained device, it cannot implement a cache. Rather, the JP processes forwarding-related CoAP options and makes requests on behalf of the pledge, in a stateless manner by using the Stateless-Proxy option defined in this document.

The pledge designates a JP as a proxy by including the Proxy-Scheme option in CoAP requests it sends to the JP. The pledge also includes in the requests the Uri-Host option with its value set to the well-known JRC's alias, as specified in Section 9.1.

The JP resolves the alias to the IPv6 address of the JRC that it learned when it acted as a pledge, and joined the network. This allows the JP to reach the JRC at the network layer and forward the requests on behalf of the pledge.

The JP MUST add a Stateless-Proxy option to all the requests that it forwards on behalf of the pledge as part of the join process.

The value of the Stateless-Proxy option is set to the internal JP state, needed to forward the Join Response message to the pledge. The Stateless-Proxy option handling is defined in Section 10.

The JP also tags all packets carrying the Join Request message at the network layer, as specified in Section 7.1.

8.1. OSCORE Security Context

Before the pledge and the JRC may start exchanging CoAP messages protected with OSCORE, they need to derive the OSCORE security context from the parameters provisioned out-of-band, as discussed in Section 4.

The OSCORE security context MUST be derived at the pledge and the JRC as per Section 3 of [I-D.ietf-core-object-security].

- the Master Secret MUST be the PSK.
- the Master Salt MUST be the pledge identifier.
- the Sender ID of the pledge MUST be set to byte string 0x00.
- the Recipient ID (ID of the JRC) MUST be set to byte string 0x01.
- the Algorithm MUST be set to the value from [RFC8152], agreed out-of-band by the same mechanism used to provision the PSK. The default is AES-CCM-16-64-128.
- the Key Derivation Function MUST be agreed out-of-band. Default is HKDF SHA-256 [RFC5869].

The derivation in [I-D.ietf-core-object-security] results in traffic keys and a common IV for each side of the conversation. Nonces are constructed by XOR'ing the common IV with the current sequence number

and sender identifier. For details on nonce construction, refer to [I-D.ietf-core-object-security].

Implementations MUST ensure that multiple CoAP requests to different JRCs result in the use of the same OSCORE context, so that the sequence numbers are properly incremented for each request. The pledge typically sends requests to different JRCs if it is not provisioned with the network identifier and attempts to join one network at a time. A simple implementation technique is to instantiate the OSCORE security context with a given PSK only once and use it for all subsequent requests. Failure to comply will break the confidentiality property of the AEAD algorithm due to the nonce reuse.

8.1.1. Persistency

Implementations MUST ensure that mutable OSCORE context parameters (Sender Sequence Number, Replay Window) are stored in persistent memory. A technique that prevents reuse of sequence numbers, detailed in Section 6.5.1 of [I-D.ietf-core-object-security], MUST be implemented. Each update of the OSCORE Replay Window MUST be written to persistent memory.

This is an important security requirement in order to guarantee nonce uniqueness and resistance to replay attacks across reboots and rejoins. Traffic between the pledge and the JRC is rare, making security outweigh the cost of writing to persistent memory.

9. 6TiSCH Join Protocol

6TiSCH Join Protocol (6JP) is a lightweight protocol over CoAP [RFC7252] and a secure channel provided by OSCORE [I-D.ietf-core-object-security]. 6JP allows the pledge to request admission into a network managed by the JRC, and for the JRC to configure the pledge with the parameters necessary for joining the network. These parameters are: link-layer keys in use, IEEE 802.15.4 short address assigned to the pledge, and the IPv6 address of the JRC.

This section specifies the 6JP bindings to CoAP and OSCORE, 6JP message formats and the semantics of different fields.

6JP relies on the security properties provided by OSCORE. This includes end-to-end confidentiality, data authenticity, replay protection, and a secure binding of responses to requests.

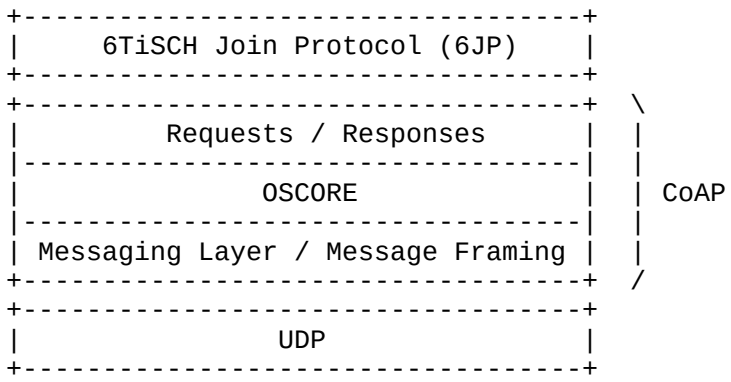


Figure 2: Abstract layering of 6JP.

6JP consists of two messages:

- the Join Request message, sent by the pledge to the JRC, proxied by the JP. The Join Request message and its mapping to CoAP is specified in Section 9.1.

- the Join Response message, sent by the JRC to the pledge if the JRC successfully processes the Join Request using OSCORE and it determines through a mechanism that is out of scope of this specification that the pledge is authorized to join the network. The Join Response message is proxied by the JP. The Join Response message and its mapping to CoAP is specified in Section 9.2.

The payload of 6JP messages is encoded with CBOR [RFC7049], with some parameters being optional. The first byte of the CBOR-encoded byte string contains the CBOR major type and additional information (e.g. the number of elements in an array). In case of an array, the CBOR decoder decides based on this additional information if a certain optional parameter is present or not.

9.1. Specification of the Join Request

The Join Request the pledge sends SHALL be mapped to a CoAP request:

- The request method is POST.

- The type is Non-confirmable (NON).

- The Proxy-Scheme option is set to "coap".

- The Uri-Host option is set to "6tisch.arpa".

- The Uri-Path option is set to "j".

- The Object-Security option SHALL be set according to [I-D.ietf-core-object-security]. The OSCORE security context used is the one derived in Section 8.1. The OSCORE Context Hint SHALL be set to the pledge identifier. The OSCORE Context Hint allows the JRC to retrieve the security context for a given pledge.

- The payload is a CBOR array [RFC7049] containing, in order:

  * Byte string, containing the identifier of the network that the pledge is attempting to join. This enables the JRC to manage multiple 6TiSCH networks.

```
request_payload = [
    network_identifier : bstr,
]
```

9.2. Specification of the Join Response

The Join Response the JRC sends SHALL be mapped to a CoAP response:

- The response Code is 2.04 (Changed).

- The payload is a CBOR array [RFC7049] containing, in order:

  * the COSE Key Set, specified in [RFC8152], containing one or more link-layer keys. The mapping of individual keys to 802.15.4-specific parameters is described in Section 9.2.1.

  * the link-layer short address to be used by the pledge. The format of the short address follows Section 9.2.2.

  * optionally, the IPv6 address of the JRC, encoded as a byte string, with the length of 16 bytes. If the IPv6 address of the JRC is not present in the Join Response, this indicates the JRC is co-located with the 6LBR, and has the same IPv6 address as the 6LBR. See Section 7.

```
response_payload = [
    COSE_KeySet,
    short_address,
    ? JRC_address : bstr,
]
```

9.2.1. Link-layer Keys Transported in the COSE Key Set

Each key in the COSE Key Set [RFC8152] SHALL be a symmetric key. The first key in the COSE Key Set SHALL be used as the K1 key from [RFC8180]. The second key in the COSE Key Set SHALL be used as the K2 key from [RFC8180]. In the case where the network uses the same key for K1 and K2, the COSE Key Set SHALL carry a single key.

If the COSE Key Set carries more than 2 keys, the implementation SHOULD consider the response as malformed.

If the "kid" parameter of the COSE Key structure is present, the corresponding key SHALL be used as IEEE 802.15.4 KeyIdMode 0x01 (index). In that case, parameter "kid" of the COSE Key structure SHALL be used to carry the IEEE 802.15.4 KeyIndex value.

If the length of the "kid" parameter is more than 1 byte (length defined by [IEEE802.15.4-2015]), the implementation SHOULD consider the response as malformed.

If the "kid" parameter is not present in the transported key, the implementation SHALL consider the key to be an IEEE 802.15.4 KeyIdMode 0x00 (implicit) key.

This document does not support IEEE 802.15.4 KeyIdMode 0x02 and 0x03 class keys. In the case that the response is considered malformed, the implementation SHOULD indicate to the user through an out-of-band mechanism the presence of an error condition.

9.2.2. Short Address

The "short_address" structure transported as part of the join response payload represents the IEEE 802.15.4 short address assigned to the pledge. It is encoded as a CBOR array object, containing, in order:

- Byte string, containing the 16-bit address.
- Optionally, the lease time parameter, "lease_asn". The value of the "lease_asn" parameter is the 5-byte Absolute Slot Number (ASN) corresponding to its expiration, carried as a byte string in network byte order.

```
short_address = [
    address : bstr,
    ? lease_asn : bstr,
]
```

It is up to the joined node to request a new short address before the expiry of its previous address. The mechanism by which the node requests renewal is the same as during join procedure, as described in Section 9.4.

9.3. Error Handling and Retransmission

Since the Join Request is mapped to a Non-confirmable CoAP message, OSCORE processing at the JRC will silently drop the request in case of a failure. This may happen for a number of reasons, including failed lookup of an appropriate security context (e.g. the pledge attempting to join a wrong network), failed decryption, positive replay window lookup, formatting errors (possibly due to malicious alterations in transit). Silently dropping the Join Request at the JRC prevents a DoS attack where an attacker could force the pledge to attempt joining one network at a time, until all networks have been tried.

Using a Non-confirmable CoAP message to transport the Join Request also helps minimize the required CoAP state at the pledge and the Join Proxy, keeping it to a minimum typically needed to perform CoAP congestion control. It does, however, introduce some complexity as the pledge needs to implement a retransmission mechanism.

The following binary exponential back-off algorithm is inspired by the one described in [RFC7252]. For each Join Request the pledge sends while waiting for a Join Response, the pledge MUST keep track of a timeout and a retransmission counter. For a new Join Request, the timeout is set to a random value between TIMEOUT_BASE and (TIMEOUT_BASE * TIMEOUT_RANDOM_FACTOR), and the retransmission counter is set to 0. When the timeout is triggered and the retransmission counter is less than MAX_RETRANSMIT, the Join Request is retransmitted, the retransmission counter is incremented, and the timeout is doubled. Note that the retransmitted Join Request passes new OSCORE processing, such that the sequence number in the OSCORE context is properly incremented. If the retransmission counter reaches MAX_RETRANSMIT on a timeout, the pledge SHOULD attempt to join the next advertised 6TiSCH network. If the pledge receives a Join Response that successfully passes OSCORE processing, it cancels the pending timeout and processes the response. The pledge MUST silently discard any response not protected with OSCORE, including error codes. For default values of retransmission parameters, see Section 9.5.

If all join attempts to advertised networks have failed, the pledge SHOULD signal to the user the presence of an error condition, through some out-of-band mechanism.

9.4. Rekeying and Rejoining

This specification handles initial keying of the pledge. For reasons such as rejoining after a long sleep, expiry of the short address, or node-initiated rekeying, the joined node MAY send a new Join Request using the already-established OSCORE security context. The JRC then responds with up-to-date keys and a (possibly new) short address. How the joined node decides when to rekey is out of scope of this document. Mechanisms for rekeying the network are defined in companion specifications.

9.5. Parameters

6JP uses the following parameters:

Name	Default Value
TIMEOUT_BASE	10 s
TIMEOUT_RANDOM_FACTOR	1.5
MAX_RETRANSMIT	4

The values of TIMEOUT_BASE, TIMEOUT_RANDOM_FACTOR, MAX_RETRANSMIT may be configured to values specific to the deployment. The default values have been chosen to accommodate a wide range of deployments, taking into account dense networks.

9.6. Mandatory to Implement Algorithms

The mandatory to implement AEAD algorithm for use with OSCORE is AES-CCM-16-64-128 from [RFC8152]. This is the algorithm used for securing 802.15.4 frames, and hardware acceleration for it is present in virtually all compliant radio chips. With this choice, CoAP messages are protected with an 8-byte CCM authentication tag, and the algorithm uses 13-byte long nonces.

The mandatory to implement hash algorithm is SHA-256 [RFC4231].

The mandatory to implement key derivation function is HKDF [RFC5869], instantiated with a SHA-256 hash.

10. Stateless-Proxy CoAP Option

The CoAP proxy defined in [RFC7252] keeps per-client state information in order to forward the response towards the originator of the request. This state information includes at least the CoAP token, the IPv6 address of the host, and the UDP source port number. If the JP used the stateful CoAP proxy defined in [RFC7252], it would be prone to Denial-of-Service (DoS) attacks, due to its limited memory.

The Stateless-Proxy CoAP option Figure 3 allows the JP to be entirely stateless. This option inserts, in the request, the state information needed for relaying the response back to the client. The proxy still keeps some general state (e.g. for congestion control or request retransmission), but no per-client state.

The Stateless-Proxy CoAP option is critical, Safe-to-Forward, not part of the cache key, not repeatable and opaque. When processed by OSCORE, the Stateless-Proxy option is neither encrypted nor integrity protected.

No.	C	U	N	R	Name	Format	Length
TBD	x		x		Stateless-Proxy	opaque	1-255

C=Critical, U=Unsafe, N=NoCacheKey, R=Repeatable

Figure 3: Stateless-Proxy CoAP Option

Upon reception of a Stateless-Proxy option, the CoAP server MUST echo it in the response. The value of the Stateless-Proxy option is internal proxy state that is opaque to the server. Example state information includes the IPv6 address of the client, its UDP source port, and the CoAP token. For security reasons, the state information MUST be authenticated, MUST include a freshness indicator (e.g. a sequence number or timestamp) and MAY be encrypted. The proxy may use an appropriate COSE structure [RFC8152] to wrap the state information as the value of the Stateless-Proxy option. The key used for encryption/authentication of the state information may be known only to the proxy.

Once the proxy has received the CoAP response with a Stateless-Proxy option present, it decrypts/authenticates it, checks the freshness indicator and constructs the response for the client, based on the information present in the option value.

Note that a CoAP proxy using the Stateless-Proxy option is not able to return a 5.04 Gateway Timeout Response Code in case the request to the server times out. Likewise, if the response to the proxy's request does not contain the Stateless-Proxy option, for example when the option is not supported by the server, the proxy is not able to return the response to the client.

11. Security Considerations

This document recommends that the pledge and JRC are provisioned with unique PSKs. The nonce used for the Join Request and the Join Response is the same, but used under a different key. The design differentiates between keys derived for requests and keys derived for responses by different sender identifiers (0x00 for pledge and 0x01 for JRC). Note that the address of the JRC does not take part in nonce or key construction. Even in the case of a misconfiguration in which the same PSK is used for several pledges, the keys used to protect the requests/responses from/towards different pledges are different, as they are derived using the pledge identifier as Master Salt. The PSK is still important for mutual authentication of the pledge and the JRC. Should an attacker come to know the PSK, then a man-in-the-middle attack is possible. The well-known problem with Bluetooth headsets with a "0000" pin applies here.

Being a stateless relay, the JP blindly forwards the join traffic into the network. A simple bandwidth cap on the JP prevents it from forwarding more traffic than the network can handle. This forces attackers to use more than one Join Proxy if they wish to overwhelm the network. Marking the join traffic packets with a non-zero DSCP allows the network to carry the traffic if it has capacity, but encourages the network to drop the extra traffic rather than add bandwidth due to that traffic.

The shared nature of the "minimal" cell used for the join traffic makes the network prone to DoS attacks by congesting the JP with bogus traffic. Such an attacker is limited by its maximum transmit power. The redundancy in the number of deployed JPs alleviates the issue and also gives the pledge a possibility to use the best available link for joining. How a network node decides to become a JP is out of scope of this specification.

At the beginning of the join process, the pledge has no means of verifying the content in the EB, and has to accept it at "face value". In case the pledge tries to join an attacker's network, the Join Response message will either fail the security check or time out. The pledge may implement a blacklist in order to filter out undesired EBs and try to join using the next seemingly valid EB. This blacklist alleviates the issue, but is effectively limited by

the node's available memory. Bogus beacons prolong the join time of the pledge, and so the time spent in "minimal" [RFC8180] duty cycle mode.

12. Privacy Considerations

The join solution specified in this document relies on the uniqueness of the pledge identifier within the namespace managed by the JRC. This identifier is transferred in clear as an OSCORE Context Hint. The use of the globally unique EUI-64 as pledge identifier simplifies the management but comes with certain privacy risks. The implications are thoroughly discussed in [RFC7721] and comprise correlation of activities over time, location tracking, address scanning and device-specific vulnerability exploitation. Since the join protocol is executed rarely compared to the network lifetime, long-term threats that arise from using EUI-64 as the pledge identifier are minimal. In addition, the Join Response message contains a short address which is assigned by the JRC to the pledge. The assigned short address SHOULD be uncorrelated with the long-term pledge identifier. The short address is encrypted in the response. Once the join process completes, the new node uses the short addresses for all further layer 2 (and layer-3) operations. This mitigates the aforementioned privacy risks as the short layer-2 address (visible even when the network is encrypted) is not traceable between locations and does not disclose the manufacturer, as is the case of EUI-64.

13. IANA Considerations

Note to RFC Editor: Please replace all occurrences of "[[this document]]" with the RFC number of this specification.

This document allocates a well-known name under the .arpa name space according to the rules given in [RFC3172]. The name "6tisch.arpa" is requested. No subdomains are expected. No A, AAAA or PTR record is requested.

13.1. CoAP Option Numbers Registry

The Stateless-Proxy option is added to the CoAP Option Numbers registry:

Number	Name	Reference
TBD	Stateless-Proxy	[[this document]]

14. Acknowledgments

The work on this document has been partially supported by the European Union's H2020 Programme for research, technological development and demonstration under grant agreement No 644852, project ARMOUR.

The authors are grateful to Thomas Watteyne and Goeran Selander for reviewing, and to Klaus Hartke for providing input on the Stateless-Proxy CoAP option. The authors would also like to thank Francesca Palombini, Ludwig Seitz and John Mattsson for participating in the discussions that have helped shape the document.

15. References

15.1. Normative References

[I-D.ietf-core-object-security]
Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)", draft-ietf-core-object-security-08 (work in progress), January 2018.

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

[RFC2597] Heinanen, J., Baker, F., Weiss, W., and J. Wroclawski, "Assured Forwarding PHB Group", RFC 2597, DOI 10.17487/RFC2597, June 1999, <https://www.rfc-editor.org/info/rfc2597>.

[RFC3172] Huston, G., Ed., "Management Guidelines & Operational Requirements for the Address and Routing Parameter Area Domain ("arpa")", BCP 52, RFC 3172, DOI 10.17487/RFC3172, September 2001, <https://www.rfc-editor.org/info/rfc3172>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <https://www.rfc-editor.org/info/rfc7049>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC8152] Schaad, J., "CBOR Object Signing and Encryption (COSE)", RFC 8152, DOI 10.17487/RFC8152, July 2017, <https://www.rfc-editor.org/info/rfc8152>.

15.2. Informative References

[I-D.ietf-6tisch-6top-protocol]
Wang, Q., Vilajosana, X., and T. Watteyne, "6top Protocol (6P)", draft-ietf-6tisch-6top-protocol-09 (work in progress), October 2017.

[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-09 (work in progress), June 2017.

[I-D.ietf-cbor-cddl]
Birkholz, H., Vigano, C., and C. Bormann, "Concise data definition language (CDDL): a notational convention to express CBOR data structures", draft-ietf-cbor-cddl-02 (work in progress), February 2018.

[I-D.richardson-6tisch-minimal-rekey]
Richardson, M., "Minimal Security rekeying mechanism for 6TiSCH", draft-richardson-6tisch-minimal-rekey-02 (work in progress), August 2017.

[IEEE802.15.4-2015]
IEEE standard for Information Technology, ., "IEEE Std 802.15.4-2015 Standard for Low-Rate Wireless Personal Area Networks (WPANs)", 2015.

[RFC4231] Nystrom, M., "Identifiers and Test Vectors for HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512", RFC 4231, DOI 10.17487/RFC4231, December 2005, <https://www.rfc-editor.org/info/rfc4231>.

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, DOI 10.17487/RFC4944, September 2007, <https://www.rfc-editor.org/info/rfc4944>.

[RFC5869] Krawczyk, H. and P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC 5869, DOI 10.17487/RFC5869, May 2010, <https://www.rfc-editor.org/info/rfc5869>.

[RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, DOI 10.17487/RFC6550, March 2012, <https://www.rfc-editor.org/info/rfc6550>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <https://www.rfc-editor.org/info/rfc6775>.

[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, <https://www.rfc-editor.org/info/rfc7554>.

[RFC7721] Cooper, A., Gont, F., and D. Thaler, "Security and Privacy Considerations for IPv6 Address Generation Mechanisms", RFC 7721, DOI 10.17487/RFC7721, March 2016, <https://www.rfc-editor.org/info/rfc7721>.

[RFC8180] Vilajosana, X., Ed., Pister, K., and T. Watteyne, "Minimal IPv6 over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration", BCP 210, RFC 8180, DOI 10.17487/RFC8180, May 2017, <https://www.rfc-editor.org/info/rfc8180>.

Appendix A. Example

Figure 4 illustrates a successful join protocol exchange. The pledge instantiates the OSCORE context and derives the traffic keys and nonces from the PSK. It uses the instantiated context to protect the Join Request addressed with a Proxy-Scheme option, the well-known host name of the JRC in the Uri-Host option, and its EUI-64 as pledge identifier and OSCORE Context Hint. Triggered by the presence of a Proxy-Scheme option, the JP forwards the request to the JRC and adds the Stateless-Proxy option with value set to the internally needed state, authentication tag, and a freshness indicator. The JP has learned the IPv6 address of the JRC when it acted as a pledge and joined the network. Once the JRC receives the request, it looks up the correct context based on the Context Hint parameter. It reconstructs OSCORE's external Additional Authenticated Data (AAD) needed for verification based on:

- the Version of the received CoAP header.

- the Algorithm value agreed out-of-band, default being AES-CCM-16-64-128 from [RFC8152].
- the Request ID being set to the value of the "kid" field of the received COSE object.
- the Join Request sequence number set to the value of "Partial IV" field of the received COSE object.
- Integrity-protected options received as part of the request.

Replay protection is ensured by OSCORE and through persistent handling of mutable context parameters. Once the JP receives the Join Response, it authenticates the Stateless-Proxy option before deciding where to forward. The JP sets its internal state to that found in the Stateless-Proxy option, and forwards the Join Response to the correct pledge. Note that the JP does not possess the key to decrypt the COSE object (join_response) present in the payload. The Join Response is matched to the Join Request and verified for replay protection at the pledge using OSCORE processing rules. In this example, the Join Response does not contain the IPv6 address of the JRC, the pledge hence understands the JRC is co-located with the 6LBR.

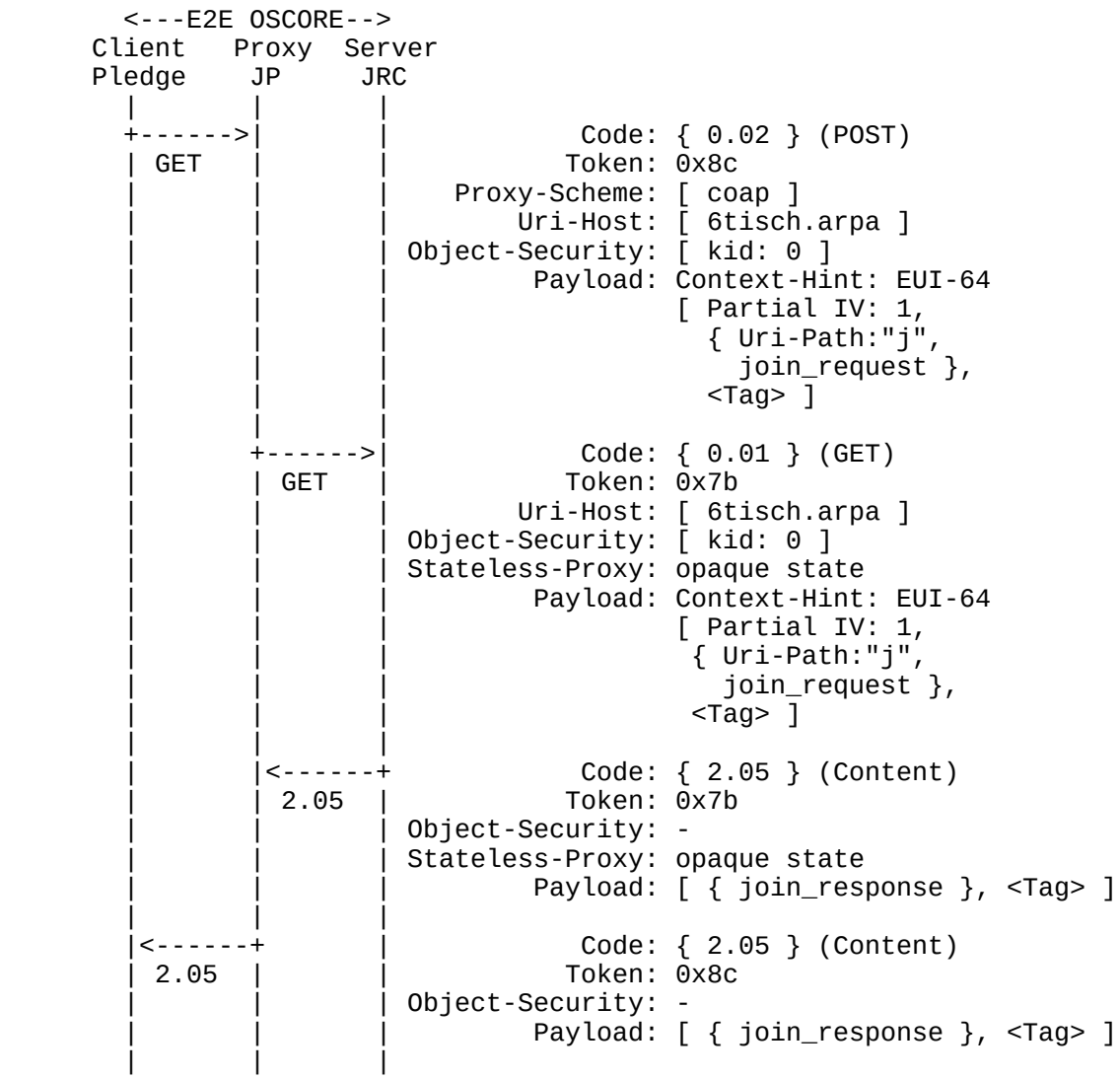


Figure 4: Example of a successful join protocol exchange. { ... } denotes encryption and authentication, [...] denotes authentication.

Where join_request is:

```
join_request:
[
    h'cafe' / PAN ID of the network pledge is attempting to join /
]
```

The join_request encodes to h'8142cafe' with a size of 4 bytes.

And join_response is:

```
join_response:
[
    [   / COSE Key Set array with a single key /
        {
             1 : 4, / key type symmetric /
             2 : h'01', / key id /
            -1 : h'e6bf4287c2d7618d6a9687445ffd33e6' / key value /
        }
    ],
    [
        h'af93' / assigned short address /
    ]
]
```

The join_response encodes to h'8281a301040241012050e6bf4287c2d7618d6a9687445ffd33e68142af93' with a size of 30 bytes.

Authors' Addresses

Malisa Vucinic (editor)
University of Montenegro
Dzordza Vasingtona bb
Podgorica 81000
Montenegro

Email: malisav@ac.me

Jonathan Simon
Analog Devices
32990 Alvarado-Niles Road, Suite 910
Union City, CA 94587
USA

Email: jonathan.simon@analog.com

Kris Pister
University of California Berkeley
512 Cory Hall
Berkeley, CA 94720
USA

Email: pister@eecs.berkeley.edu

Michael Richardson
Sandelman Software Works
470 Dawson Avenue
Ottawa, ON K1Z5V7
Canada

Email: mcr+ietf@sandelman.ca

6TiSCH
Internet-Draft
Intended status: Informational
Expires: June 19, 2017

MR. Palattella, Ed.
LIST
P. Thubert
cisco
T. Watteyne
Linear Technology / Dust Networks
Q. Wang
Univ. of Sci. and Tech. Beijing
December 16, 2016

Terminology in IPv6 over the TSCH mode of IEEE 802.15.4e
draft-ietf-6tisch-terminology-08

Abstract

This document provides a glossary of terminology used in IPv6 over the TSCH mode of IEEE 802.15.4e (6TiSCH). This document extends existing terminology documents for Low-power and Lossy Networks.

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 http://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 June 19, 2017.

Copyright Notice

Table of Contents

1. Introduction 2
2. Terminology 2
3. Security Considerations 8
4. References 8
4.1. Normative References 8
4.2. Informative References 9
4.3. External Informative References 10
Authors' Addresses 10

1. Introduction

The IEEE802.15.4 Medium Access Control (MAC) has evolved with the Time Slotted Channel Hopping (TSCH) mode for industrial-type applications. It provides deterministic capabilities to the point that a packet that pertains to a certain flow crosses the network from node to node following a very precise schedule, like a train leaves intermediate stations at precise times along its path.

This document provides additional terminology elements to cover terms that are new to the context of TSCH wireless networks and other deterministic networks.

2. Terminology

The draft extends [RFC7102] and use terms from [RFC6550] and [RFC6552], which are all included here by reference.

The draft does not reuse terms from IEEE802.15.4e such as "path" or "link" which bear a meaning that is quite different from classical IETF parlance.

This document adds the following terms:

6TiSCH: IPv6 over the Timeslotted Channel Hopping (TSCH) mode of IEEE802.15.4e. It defines (i) the 6top sublayer; (ii) a set of protocols for setting up a TSCH schedule in distributed approach, for managing the allocation of resources; and (iii) the architecture to bind them together, for use in IPv6 TSCH based networks.

6top: The "6TiSCH Operation Sublayer" (6top) is the next highest layer of the IEEE802.15.4e TSCH medium access

control layer. It implements and terminates the "6top Protocol" (6P), and contains a "6top Scheduling Function" (SF).

SF: The "6top Scheduling Function" (SF) "is the cell management entity that add or delete cells dynamically based on its allocation policy in order to fulfill cell requirements. The cell negotiation with a neighbor is done using 6P. General guidelines for designing a SF are provided in [I-D.ietf-6tisch-6top-protocol].

SFID: The "6top Scheduling Function Identifier" (SFID) is a 4-bit field identifying a SF. Defined in [I-D.ietf-6tisch-6top-protocol].

6P: The "6top Protocol" (6P) allows neighbor nodes to communicate to add/delete cells to one another in their TSCH schedule. Defined in [I-D.ietf-6tisch-6top-protocol].

6P Transaction: Part of the "6top Protocol" (6P), the action of two neighbors exchanging a 6P request message and the corresponding 6P response message. Defined in [I-D.ietf-6tisch-6top-protocol].

ASN: Absolute Slot Number, the total number of timeslots that have elapsed since the PAN coordinator has started the TSCH network. Incremented by one at each timeslot. It is wide enough to not roll over in practice. See [IEEE802154-2015] and [RFC7554].

Blacklist of Frequencies: A set of frequencies which should not be used for communication. See [IEEE802154-2015] and [RFC7554].

BBR: Backbone Router. In the 6TiSCH architecture, an LBR and also a IPv6 ND-efficiency-aware Router (NEAR) [I-D.chakrabarti-nordmark-6man-efficient-nd]. Performs ND proxy operations between registered devices and classical ND devices that are located over the backbone.

Broadcast Cell: A scheduled cell used for broadcast transmission.

Bundle: A group of equivalent scheduled cells, i.e. cells identified by different [slotOffset, channelOffset], which are scheduled for a same purpose, with the same neighbor, with the same flags, and the same slotframe. The size of the bundle refers to the number of cells it

contains. For a given slotframe length, the size of the bundle translates directly into bandwidth. A bundle is a local abstraction thar represents a half-duplex link for either sending or receiving, with bandwidth that amounts to the sum of the cells in the bundle. A bundle is globally identified by (source MAC, destination MAC, TrackID). At Layer 3, a pair of bundles forms a link. By using a well-known constant, NULLT, as TrackId for a L3 link, the IP link between adjacent nodes A and B comprises 2 bundles: (macA, macB, NULLT) and (macB, macA, NULLT). At Layer 2, a pair of bundles forms a switching state. Considered a segment A-B-C along a track, there are two bundles in node B, one incoming = (macA, macB, trackId) and one outgoing = (macB, macC, trackId).

CCA: Clear Channel Assessment. Mechanism defined in [IEEE802154-2015], section 6.2.5.2. In a TSCH network, CCA can be used to detect other radio networks in vicinity. Nodes listen the channel before sending, to detect other ongoing transmissions. Because the network is synchronized, CCA cannot be used to detect colliding transmission within the same network. CCA is necessary for the 6TiSCH minimal configuration [I-D.ietf-6tisch-minimal] in shared slots, and in presence of multiple instances of 6TiSCH networks.

Cell: A single element in the TSCH schedule, identified by a slotOffset, a channelOffset, a slotframeHandle. A cell can be scheduled or unscheduled.

Centralized Cell Reservation: A reservation of a cell done by a centralized entity (e.g., a PCE) in the network.

Centralized Track Reservation: A reservation of a track done by a centralized entity (e.g., a PCE) in the network.

ChannelOffset: Identifies a row in the TSCH schedule. The number of available channelOffset values is equal to the number of available frequencies. The channelOffset translates into a frequency when the communication takes place, resulting in channel hopping. See [RFC7554].

Channel Distribution/Usage (CDU) matrix: : Matrix of cells (i,j) representing the spectrum (channel) distribution among the different nodes in the 6TiSCH network. The CDU matrix has width in timeslots, equal to the period of the network scheduling operation, and height equal to the number of available channels. Every cell (i,j) in the

CDU, identified by (slotOffset, channelOffset), belongs to a specific chunk. It has to be noticed that such a matrix which includes all the cells grouped in chunks, belonging to different slotframes, is different from the TSCH schedule.

Chunk: A well-known list of cells, distributed in time and frequency, within a CDU matrix; a chunk represents a portion of a CDU matrix. The partition of the CDU in chunks is globally known by all the nodes in the network to support the appropriation process, which is a negotiation between nodes within an interference domain. A node that manages to appropriate a chunk gets to decide which transmissions will occur over the cells in the chunk within its interference domain (i.e., a parent node will decide when the cells within the appropriated chunk are used and by which node, among its children.

Dedicated Cell: A cell that is reserved for a given node to transmit to a specific neighbor.

Deterministic Network: The generic concept of deterministic network is defined in [I-D.ietf-detnet-architecture]. When applied to 6TiSCH it refers to the reservation of tracks which guarantee an end to end latency and optimize the PDR for well-characterized flows.

Distributed Cell Reservation: A reservation of a cell done by one or more in-network entities (typically a connection endpoint).

Distributed Track Reservation: A reservation of a track done by one or more in-network entities (typically a connection endpoint).

EB: Enhanced Beacon frame used by a node to announce the presence of the network. It contains enough information for a joining node to synchronize to the network. See [IEEE802154-2015] and [RFC7554].

Hard Cell: A scheduled cell which the 6top sublayer cannot relocate.

Hopping Sequence: Ordered sequence of frequencies, identified by a Hopping_Sequence_ID, used for channel hopping, when translating the channel offset value into a frequency (i.e., PHY channel). See [IEEE802154-2015] and [RFC7554].

IE: Information Element, a Type-Length-Value containers placed at the end of the MAC header, used to pass data between layers or devices. Some IE identifiers are managed by the IEEE [IEEE802154-2015]. Some IE identifiers are managed by the IETF [I-D.kivinen-802-15-ie].

JCE: The Join Coordination Entity (JCE) is a central entity that coordinates the joining of new nodes in the network. See [I-D.ietf-6tisch-minimal-security] and [I-D.ietf-6tisch-dtsecurity-secure-join].

JA: The Join Assistant (JA) is a one-hop neighbor of a joining node that may facilitate it to become meaningful part of the network (e.g., by serving as a local connectivity point to the remainder of the network). JA emits EBs, used by JNs to synchronize to the network. See [I-D.ietf-6tisch-minimal-security] and [I-D.ietf-6tisch-dtsecurity-secure-join].

JN: The Joining Node (JN) is a device attempting to join a particular 6TiSCH network. See [I-D.ietf-6tisch-minimal-security].

Join Protocol: The protocol which secures initial communication between a joining node and the JCE.

LBR: Low-power Lossy Network (LLN) Border Router. It is an LLN device, usually powered, that acts as a Border Router to the outside within the 6TiSCH architecture.

Link: A communication facility or medium over which nodes can communicate at the link layer, i.e., the layer immediately below IP. Thus, the IETF parlance for the term "Link" is adopted, as opposed to the IEEE802.15.4e terminology.

Operational Network: A IEEE802.15.4e network whose encryption/ authentication keys are determined by some algorithms/ protocols. There may be network-wide group keys, or per-link keys.

(to) Relocate a Cell: The action operated by the 6top sublayer of changing the slotOffset and/or channelOffset of a soft cell.

(to) Schedule a Cell: The action of turning an unscheduled cell into a scheduled cell.

Scheduled cell: A cell which is assigned a neighbor MAC address (broadcast address is also possible), and one or more of the following flags: TX, RX, shared, timeskeeping. A scheduled cell can be used by the IEEE802.15.4e TSCH implementation to communicate. A scheduled cell can be either a hard or a soft cell.

Shared Cell: A cell marked with both the "TX" and "shared" flags. This cell can be used by more than one transmitter node. A back-off algorithm is used to resolve contention. See [IEEE802154-2015] and [RFC7554].

SlotOffset: Identifies a column in the TSCH schedule, i.e., the number of timeslots since the beginning of the current iteration of the slotframe. See [IEEE802154-2015] and [RFC7554].

Slotframe: A collection of timeslots repeating in time, analogous to a superframe in that it defines periods of communication opportunities. It is characterized by a slotframe_ID, and a slotframe_size. Multiple slotframes can coexist in a node’s schedule, i.e., a node can have multiple activities scheduled in different slotframes, based on the priority of its packets/traffic flows. The timeslots in the Slotframe are indexed by the SlotOffset; the first timeslot is at SlotOffset 0. See [IEEE802154-2015] and [RFC7554].

Soft Cell: A scheduled cell which the 6top sublayer can relocate.

Timeslot: A basic communication unit in TSCH which allows a transmitter node to send a frame to a receiver neighbor, and that receiver neighbor to optionally send back an acknowledgment. See [IEEE802154-2015] and [RFC7554].

Time Source Neighbor: A neighbor that a node uses as its time reference, and to which it needs to keep its clock synchronized. See [IEEE802154-2015] and [RFC7554].

Track: A determined sequence of cells along a multi-hop path. It is typically the result of a track reservation. The node that initializes the process of establishing a track is the owner of the track. The latter assigns a unique identifier to the track, called TrackID.

TrackID: Unique identifier of a track, assigned by the owner of the track.

TSCH: Time Slotted Channel Hopping, a medium access mode of the [IEEE802154-2015] standard which uses time synchronization to achieve ultra low-power operation and channel hopping to enable high reliability. See [IEEE802154-2015] and [RFC7554].

TSCH Schedule: A matrix of cells, each cell indexed by a slotOffset and a channelOffset. The TSCH schedule contains all the scheduled cells from all slotframes and is sufficient to qualify the communication in the TSCH network. The number of channelOffset values (the "height" of the matrix) is equal to the number of available frequencies. See [IEEE802154-2015] and [RFC7554].

Unscheduled Cell: A cell which is not used by the IEEE802.15.4e TSCH implementation. See [IEEE802154-2015] and [RFC7554].

3. Security Considerations

Since this document specifies terminology and does not specify new procedures or protocols, it raises no new security issues.

4. References

4.1. Normative References

[RFC2309] Braden, B., Clark, D., Crowcroft, J., Davie, B., Deering, S., Estrin, D., Floyd, S., Jacobson, V., Minshall, G., Partridge, C., Peterson, L., Ramakrishnan, K., Shenker, S., Wroclawski, J., and L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet", RFC 2309, DOI 10.17487/RFC2309, April 1998, <http://www.rfc-editor.org/info/rfc2309>.

[RFC3444] Pras, A. and J. Schoenwaelder, "On the Difference between Information Models and Data Models", RFC 3444, DOI 10.17487/RFC3444, January 2003, <http://www.rfc-editor.org/info/rfc3444>.

[RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, DOI 10.17487/RFC6550, March 2012, <http://www.rfc-editor.org/info/rfc6550>.

[RFC6552] Thubert, P., Ed., "Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6552, DOI 10.17487/RFC6552, March 2012, <http://www.rfc-editor.org/info/rfc6552>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <http://www.rfc-editor.org/info/rfc6775>.

[RFC7102] Vasseur, JP., "Terms Used in Routing for Low-Power and Lossy Networks", RFC 7102, DOI 10.17487/RFC7102, January 2014, <http://www.rfc-editor.org/info/rfc7102>.

[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>.

4.2. Informative References

[I-D.chakrabarti-nordmark-6man-efficient-nd]
Chakrabarti, S., Nordmark, E., Thubert, P., and M. Wasserman, "IPv6 Neighbor Discovery Optimizations for Wired and Wireless Networks", draft-chakrabarti-nordmark-6man-efficient-nd-07 (work in progress), February 2015.

[I-D.ietf-6tisch-6top-protocol]
Wang, Q. and X. Vilajosana, "6top Protocol (6P)", draft-ietf-6tisch-6top-protocol-03 (work in progress), October 2016.

[I-D.ietf-6tisch-dtsecurity-secure-join]
Richardson, M., "6tisch Secure Join protocol", draft-ietf-6tisch-dtsecurity-secure-join-00 (work in progress), December 2016.

[I-D.ietf-6tisch-minimal]
Vilajosana, X. and K. Pister, "Minimal 6TiSCH Configuration", draft-ietf-6tisch-minimal-17 (work in progress), November 2016.

[I-D.ietf-6tisch-minimal-security]
malisa.vucinic@st.com, m., Simon, J., and K. Pister, "Minimal Security Framework for 6TiSCH", draft-ietf-6tisch-minimal-security-00 (work in progress), December 2016.

[I-D.ietf-detnet-architecture]
Finn, N. and P. Thubert, "Deterministic Networking Architecture", draft-ietf-detnet-architecture-00 (work in progress), September 2016.

[I-D.kivinen-802-15-ie]
Kivinen, T. and P. Kinney, "IEEE 802.15.4 Information Element for IETF", draft-kivinen-802-15-ie-04 (work in progress), October 2016.

[I-D.thubert-6lo-rfc6775-update-reqs]
Thubert, P. and P. Stok, "Requirements for an update to 6LoWPAN ND", draft-thubert-6lo-rfc6775-update-reqs-07 (work in progress), April 2016.

[I-D.thubert-roll-forwarding-frags]
Thubert, P. and J. Hui, "LLN Fragment Forwarding and Recovery", draft-thubert-roll-forwarding-frags-02 (work in progress), September 2013.

4.3. External Informative References

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

Authors' Addresses

Maria Rita Palattella (editor)
Luxembourg Institute of Science and Technology
Department 'Environmental Research and Innovation' (ERIN)
41, rue du Brill
Belvaux L-4422
Luxembourg

Phone: (+352) 275 888-5055
Email: mariarita.palattella@list.lu

Pascal Thubert
Cisco Systems, Inc
Village d'Entreprises Green Side
400, Avenue de Roumanille
Batiment T3
Biot - Sophia Antipolis 06410
France

Phone: +33 497 23 26 34
Email: pthubert@cisco.com

Thomas Watteyne
Linear Technology / Dust Networks
30695 Huntwood Avenue
Hayward, CA 94544
USA

Phone: +1 (510) 400-2978
Email: twatteyne@linear.com

Qin Wang
Univ. of Sci. and Tech. Beijing
30 Xueyuan Road
Beijing 100083
China

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

6TiSCH
Internet-Draft
Intended status: Informational
Expires: September 3, 2018

MR. Palattella, Ed.
LIST
P. Thubert
cisco
T. Watteyne
Analog Devices
Q. Wang
Univ. of Sci. and Tech. Beijing
March 2, 2018

Terms Used in IPv6 over the TSCH mode of IEEE 802.15.4e
draft-ietf-6tisch-terminology-10

Abstract

This document provides a glossary of terminology used in IPv6 over the TSCH mode of IEEE 802.15.4e (6TiSCH). This document extends existing terminology documents for Low-power and Lossy Networks.

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 3, 2018.

Copyright Notice

Table of Contents

1. Introduction 2
2. Terminology 2
3. Security Considerations 7
4. References 7
4.1. Normative References 7
4.2. Informative References 8
4.3. External Informative References 8
Authors' Addresses 8

1. Introduction

The IEEE802.15.4 Medium Access Control (MAC) has evolved with the Time Slotted Channel Hopping (TSCH) mode for industrial-type applications.

This document provides additional terminology elements to cover terms that are new to the context of TSCH wireless networks and other deterministic networks.

2. Terminology

The draft extends [RFC7102] and use terms from [RFC6550] and [RFC6552], which are all included here by reference.

The draft does not reuse terms from IEEE802.15.4 such as "path" or "link" which bear a meaning that is quite different from classical IETF parlance.

This document adds the following terms:

6TiSCH (IPv6 over the TSCH mode of IEEE 802.15.4e): It defines the 6top sublayer, a set of protocols for setting up a TSCH schedule in distributed approach, and a security solution.

6top (6TiSCH Operation Sublayer): The next highest layer of the IEEE802.15.4 TSCH medium access control layer. It implements and terminates 6P, and contains at least one SF.

6P (6top Protocol): Allows neighbor nodes to communicate to add/ delete cells to one another in their TSCH schedule.

6P Transaction: Part of 6P, the action of two neighbors exchanging a 6P request message and the corresponding 6P response message.

ASN (Absolute Slot Number): The total number of timeslots that have elapsed since the PAN coordinator has started the TSCH network. Incremented by one at each timeslot. It is wide enough to not roll over in practice.

BBR (Backbone Router): An LBR and also a IPv6 ND-efficiency-aware Router (NEAR) [I-D.chakrabarti-nordmark-6man-efficient-nd]. Performs ND proxy operations between registered devices and classical ND devices that are located on the backbone.

blacklist of frequencies: A set of frequencies which should not be used for communication.

broadcast cell: A scheduled cell used for broadcast transmission.

bundle: A group of equivalent scheduled cells, i.e. cells identified by different [slotOffset, channelOffset], which are scheduled for a same purpose, with the same neighbor, with the same flags, and the same slotframe. The size of the bundle refers to the number of cells it contains. For a given slotframe length, the size of the bundle translates directly into bandwidth. A bundle is a local abstraction that represents a half-duplex link for either sending or receiving, with bandwidth that amounts to the sum of the cells in the bundle.

CCA (Clear Channel Assessment): Mechanism defined in [IEEE802154-2015], section 6.2.5.2. In a TSCH network, CCA can be used to detect other radio networks in vicinity. Nodes listen the channel before sending, to detect other ongoing transmissions. Because the network is synchronized, CCA cannot be used to detect colliding transmission within the same network.

cell: A single element in the TSCH schedule, identified by a slotOffset, a channelOffset, a slotframeHandle. A cell can be scheduled or unscheduled.

centralized cell reservation: A reservation of a cell done by a centralized entity (e.g., a PCE) in the network.

centralized track reservation: A reservation of a track done by a centralized entity (e.g., a PCE) in the network.

Channel Distribution/Usage (CDU) matrix: : Matrix of cells (i,j) representing the spectrum (channel) distribution among the different nodes in the 6TiSCH network. The CDU matrix has width in timeslots, equal to the period of the network scheduling operation, and height equal to the number of available channels. Every cell (i,j) in the CDU, identified by (slotOffset, channelOffset), belongs to a specific chunk. It has to be noticed that such a matrix which includes all the cells grouped in chunks, belonging to different slotframes, is different from the TSCH schedule.

channelOffset: Identifies a row in the TSCH schedule. The number of available channelOffset values is equal to the number of available frequencies. The channelOffset translates into a frequency when the communication takes place, resulting in channel hopping.

chunk: A well-known list of cells, distributed in time and frequency, within a CDU matrix. A chunk represents a portion of a CDU matrix. The partition of the CDU matrix in chunks is globally known by all the nodes in the network to support the appropriation process, which is a negotiation between nodes within an interference domain. A node that manages to appropriate a chunk gets to decide which transmissions will occur over the cells in the chunk within its interference domain (i.e., a parent node will decide when the cells within the appropriated chunk are used and by which node, among its children.

dedicated cell: A cell that is reserved for a given node to transmit to a specific neighbor.

deterministic network: The generic concept of deterministic network is defined in [I-D.ietf-detnet-architecture]. When applied to 6TiSCH, it refers to the reservation of tracks which guarantee an end-to-end latency and optimize the PDR for well-characterized flows.

distributed cell reservation: A reservation of a cell done by one or more in-network entities.

distributed track reservation: A reservation of a track done by one or more in-network entities.

EB (Enhanced Beacon): A special frame defined used by a node, including the JP, to announce the presence of the

network. It contains enough information for a pledge to synchronize to the network.

hard cell: A scheduled cell which the 6top sublayer cannot relocate.

hopping sequence: Ordered sequence of frequencies, identified by a Hopping_Sequence_ID, used for channel hopping when translating the channel offset value into a frequency.

IE (Information Element): Type-Length-Value containers placed at the end of the MAC header, used to pass data between layers or devices. Some IE identifiers are managed by the IEEE [IEEE802154-2015]. Some IE identifiers are managed by the IETF [I-D.kivinen-802-15-ie].

join process: The overall process that includes the discovery of the network by pledge(s) and the execution of the join protocol.

join protocol: The protocol that allows the pledge to join the network. The join protocol encompasses authentication, authorization and parameter distribution. The join protocol is executed between the pledge and the JRC.

joined node: The new device, after having completed the join process, often just called a node.

JP (Join Proxy): Node already part of the 6TiSCH network that serves as a relay to provide connectivity between the pledge and the JRC. The JP announces the presence of the network by regularly sending EB frames.

JRC (Join Registrar/Coordinator): Central entity responsible for the authentication, authorization and configuration of the pledge.

LBR: Low-power Lossy Network (LLN) Border Router. It is an LLN device, usually powered, that acts as a Border Router to the outside within the 6TiSCH architecture.

link: A communication facility or medium over which nodes can communicate at the link layer, the layer immediately below IP. The IETF parlance for the term "Link" is adopted, as opposed to the IEEE802.15.4 terminology.

pledge: A new device that attempts to join a 6TiSCH network.

(to) relocate a cell: The action operated by the 6top sublayer of changing the slotOffset and/or channelOffset of a soft cell.

(to) schedule a cell: The action of turning an unscheduled cell into a scheduled cell.

scheduled cell: A cell which is assigned a neighbor MAC address (broadcast address is also possible), and one or more of the following flags: TX, RX, shared, timeskeeping. A scheduled cell can be used by the IEEE802.15.4 TSCH implementation to communicate. A scheduled cell can either be a hard or a soft cell.

SF (6top Scheduling Function): The cell management entity that adds or deletes cells dynamically based on application networking requirements. The cell negotiation with a neighbor is done using 6P.

SFID (6top Scheduling Function Identifier): A 4-bit field identifying an SF.

shared cell: A cell marked with both the "TX" and "shared" flags. This cell can be used by more than one transmitter node. A back-off algorithm is used to resolve contention.

slotframe: A collection of timeslots repeating in time, analogous to a superframe in that it defines periods of communication opportunities. It is characterized by a slotframe_ID, and a slotframe_size. Multiple slotframes can coexist in a node's schedule, i.e., a node can have multiple activities scheduled in different slotframes, based on the priority of its packets/traffic flows. The timeslots in the Slotframe are indexed by the SlotOffset; the first timeslot is at SlotOffset 0.

slotOffset: A column in the TSCH schedule, i.e. the number of timeslots since the beginning of the current iteration of the slotframe.

soft cell: A scheduled cell which the 6top sublayer can relocate.

time source neighbor: A neighbor that a node uses as its time reference, and to which it needs to keep its clock synchronized.

timeslot: A basic communication unit in TSCH which allows a transmitter node to send a frame to a receiver neighbor,

and that receiver neighbor to optionally send back an acknowledgment.

track: A determined sequence of cells along a multi-hop path. It is typically the result of a track reservation. The node that initializes the process of establishing a track is the owner of the track. The latter assigns a unique identifier to the track, called TrackID.

TrackID: Unique identifier of a track.

TSCH (6top Scheduling Function Identifier): A medium access mode of the [IEEE802154-2015] standard which uses time synchronization to achieve ultra low-power operation, and channel hopping to enable high reliability.

TSCH Schedule: A matrix of cells, each cell indexed by a slotOffset and a channelOffset. The TSCH schedule contains all the scheduled cells from all slotframes and is sufficient to qualify the communication in the TSCH network. The number of channelOffset values (the "height" of the matrix) is equal to the number of available frequencies.

Unscheduled Cell: A cell which is not used by the IEEE802.15.4 TSCH implementation.

3. Security Considerations

Since this document specifies terminology and does not specify new procedures or protocols, it raises no new security issues.

4. References

4.1. Normative References

[RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, DOI 10.17487/RFC6550, March 2012, <https://www.rfc-editor.org/info/rfc6550>.

[RFC6552] Thubert, P., Ed., "Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)", RFC 6552, DOI 10.17487/RFC6552, March 2012, <https://www.rfc-editor.org/info/rfc6552>.

[RFC7102] Vasseur, JP., "Terms Used in Routing for Low-Power and Lossy Networks", RFC 7102, DOI 10.17487/RFC7102, January 2014, <https://www.rfc-editor.org/info/rfc7102>.

4.2. Informative References

[I-D.chakrabarti-nordmark-6man-efficient-nd]
Chakrabarti, S., Nordmark, E., Thubert, P., and M. Wasserman, "IPv6 Neighbor Discovery Optimizations for Wired and Wireless Networks", draft-chakrabarti-nordmark-6man-efficient-nd-07 (work in progress), February 2015.

[I-D.ietf-detnet-architecture]
Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", draft-ietf-detnet-architecture-04 (work in progress), October 2017.

[I-D.kivinen-802-15-ie]
Kivinen, T. and P. Kinney, "IEEE 802.15.4 Information Element for IETF", draft-kivinen-802-15-ie-06 (work in progress), March 2017.

4.3. External Informative References

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

Authors' Addresses

Maria Rita Palattella (editor)
Luxembourg Institute of Science and Technology
Department 'Environmental Research and Innovation' (ERIN)
41, rue du Brill
Belvaux L-4422
Luxembourg

Phone: (+352) 275 888-5055
Email: mariarita.palattella@list.lu

Pascal Thubert
Cisco Systems, Inc
Village d'Entreprises Green Side
400, Avenue de Roumanille
Batiment T3
Biot - Sophia Antipolis 06410
France

Phone: +33 497 23 26 34
Email: pthubert@cisco.com

Thomas Watteyne
Analog Devices
32990 Alvarado-Niles Road, Suite 910
Union City, CA 94587
USA

Email: thomas.watteyne@analog.com

Qin Wang
Univ. of Sci. and Tech. Beijing
30 Xueyuan Road
Beijing 100083
China

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

6lo Working Group
Internet-Draft
Intended status: Informational
Expires: September 12, 2017

D. Dujovne
Universidad Diego Portales
M. Richardson
Sandelman Software Works

March 11, 2017

IEEE802.15.4 Informational Element encapsulation of 6tisch Join Information
draft-richardson-6tisch-join-enhanced-beacon-01

Abstract

In TSCH mode of IEEE802.15.4, as described by [I-D.ietf-6tisch-minimal], opportunities for broadcasts are limited to specific times and specific channels. Nodes in a TSCH network typically frequently send Enhanced Beacon (EB) frames to announce the presence of the network. This document provides a mechanism by which small details critical for new nodes (pledges) and long sleeping nodes may be carried within the Enhanced Beacon.

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 http://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, 2017.

Copyright Notice

Table of Contents

1. Introduction 2
1.1. Terminology 2
1.2. Layer-2 Synchronization 3
1.3. Layer-3 synchronization IPv6 Router solicitations and advertisements 3
2. Protocol Definition 4
2.1. Protocol Example 4
3. Security Considerations 5
4. Privacy Considerations 5
5. IANA Considerations 5
6. Acknowledgements 5
7. References 5
7.1. Normative References 5
7.2. Informative References 6
Appendix A. Change history 6
Authors' Addresses 7

1. Introduction

[RFC7554] describes the use of the time-slotted channel hopping (TSCH) mode of [ieee802154]. As further details in [I-D.ietf-6tisch-minimal], an Enhanced Beacon is transmitted during a slot designated a broadcast slot.

EDNOTE: Explain why broadcasts are rare, and why we need them. What the Enhanced Beacon is, and what Information Elements are, and how the IETF has a subtype for that area. Explain what kind of things could be placed in Information Elements, how big they could be, and how they could be compressed.

1.1. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [RFC2119] and indicate requirement levels for compliant STuPiD implementations.

1.2. Layer-2 Synchronization

As explained in section 6 of [I-D.ietf-6tisch-minimal], the Enhanced Beacon has a number of purposes: synchronization of ASN and Join Metric, timeslot template identifier, the channel hopping sequence identifier, TSCH SlotFrame and Link IE.

The Enhanced Beacon (EB) is used by nodes already part of a TSCH network to annouce its existance. Receiving an EB allows a Joining Node (pledge) to learn about the network and synchronize to it. The EB may also be used as a means for a node already part of the network to re-synchronize [RFC7554].

There are a limited number of timeslots designated as a broadcast slot by each router. These slots are rare, and with 10ms slots, with a slot-frame length of 100, there may be only 1 slot/s for the beacon.

1.3. Layer-3 synchronization IPv6 Router solicitations and advertisements

At layer 3, [RFC2461] defines a mechanism by which nodes learn about routers by listening for multicasted Router Advertisements (RA). If no RA is heard within a set time, then a Router Solicitation (RS) may be multicast, to which an RA will be received, usually unicast.

Although [RFC6775] reduces the amount of multicast necessary to do address resolution via Neighbor Solicitation messages, it still requires multicast of either RAs or RS. This is an expensive operation for two reasons: there are few multicast timeslots for unsolicited RAs; if a pledge node does not hear an RA, and decides to send a RS (consuming a broadcast aloha slot with unencrypted traffic), many unicast RS may be sent in response.

This is a particularly acute issue for the join process for the following reasons:

1. use of a multicast slot by even a non-malicious unauthenticated node for a Router Solicitation may overwhelm that time slot.

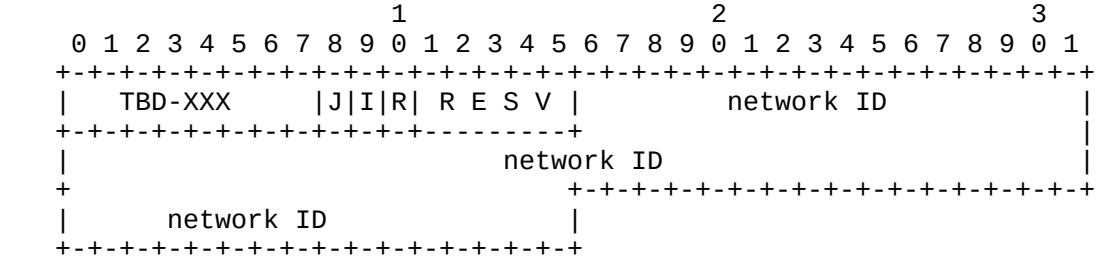
2. it may require many seconds of on-time before a new pledge hears a Router Soliciation that it can use.

3. a new pledge may listen to many Enhanced Beacons before it can pick an appropriate network and/or closest Join Assistant to attach to. If it must listen for a RS as well as find the Enhanced Beacon, then the process may take a very long time.

2. Protocol Definition

[I-D.kivinen-802-15-ie] creates a registry for new IETF IE subtypes. This document allocates a new subtype TBD-XXX.

This document documents a new IE subtype structure is as follows. As explained in [I-D.kivinen-802-15-ie] the length of the Sub-Type Content can be calculated from the container, so no length information is necessary.



J the Join Proxy flag is set if the sending node will operate as a Join Proxy according to [I-D.ietf-6tisch-minimal-security].

I the Initiate Join flag is set if this network supports pledges initiating the join process themselves according to [I-D.ietf-6tisch-minimal-security]. If not set, then the pledge should do an NS DAD operation ([RFC6775] section 4.3, explained in [I-D.ietf-6tisch-dtsecurity-secure-join]) and then remain silent, to wait to be contacted.

R the Router Advertisement flag is set if the sending node will act as a Router for host-only nodes that need addressing via unicast Router Solicitation messages.

network ID this is an opaque 16-byte identifier that uniquely identifies this network, potentially among many networks that are operating in the same frequencies in overlapping physical space.

In a 6tisch network, where RPL is used as the mesh routing protocol, the network ID SHOULD be constructed from a SHA256 hash of the DODAGID of the network. The result will be a 32-byte hash, and the lower 16-bytes should be used.

2.1. Protocol Example

Here are three examples of processing.

3. Security Considerations

All of the contents of this Information Element are sent in the clear. The containing Enhanced Beacon is not encrypted, but may be authenticated to nodes which have already received network-wide keying material.

4. Privacy Considerations

The use of a network ID may reveal information about the network. The use of a SHA256 hash of the DODAGID, rather than using the DODAGID directly provides some cover the addresses used within the network. The DODAGID is usually the IPv6 address of the root of the RPL mesh.

An interloper with a radio sniffer would be able to use the network ID to map out the extend of the mesh network.

5. IANA Considerations

Allocate a new number TBD-XXX from Registry IETF IE Sub-type ID. This entry should be called 6tisch-Join-Info.

6. Acknowledgements

Thomas Watteyne provided extensive editorial comments on the document.

7. References

7.1. Normative References

[I-D.ietf-6tisch-architecture]
Thubert, P., "An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4", draft-ietf-6tisch-architecture-11 (work in progress), January 2017.

[I-D.ietf-6tisch-minimal]
Vilajosana, X., Pister, K., and T. Watteyne, "Minimal 6TiSCH Configuration", draft-ietf-6tisch-minimal-21 (work in progress), February 2017.

[I-D.kivinen-802-15-ie]
Kivinen, T. and P. Kinney, "IEEE 802.15.4 Information Element for IETF", draft-kivinen-802-15-ie-06 (work in progress), March 2017.

[ieee802154]
IEEE Standard, ., "802.15.4-2015 - IEEE Standard for Low-Rate Wireless Personal Area Networks (WPANs)", 2015, <http://standards.ieee.org/findstds/standard/802.15.4-2015.html>.

[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>.

[RFC2461] Narten, T., Nordmark, E., and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, DOI 10.17487/RFC2461, December 1998, <http://www.rfc-editor.org/info/rfc2461>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <http://www.rfc-editor.org/info/rfc6775>.

[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>.

7.2. Informative References

[I-D.ietf-6tisch-dtsecurity-secure-join]
Richardson, M., "6tisch Secure Join protocol", draft-ietf-6tisch-dtsecurity-secure-join-01 (work in progress), February 2017.

[I-D.ietf-6tisch-minimal-security]
Vucinic, M., Simon, J., and K. Pister, "Minimal Security Framework for 6TiSCH", draft-ietf-6tisch-minimal-security-01 (work in progress), February 2017.

Appendix A. Change history

This is an evolution of an earlier proposal which provided for storing an entire IPv6 Router Adverisement in an Informational Element. It was deemed too general a solution, possibly subject to mis-use. This proposal restricts the use to just the key pieces of information required.

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

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca

6lo Working Group
Internet-Draft
Intended status: Informational
Expires: July 20, 2018

D. Dujovne
Universidad Diego Portales
M. Richardson
Sandelman Software Works
January 16, 2018

IEEE802.15.4 Informational Element encapsulation of 6tisch Join Information
draft-richardson-6tisch-join-enhanced-beacon-03

Abstract

In TSCH mode of IEEE802.15.4, as described by [RFC8180], opportunities for broadcasts are limited to specific times and specific channels. Nodes in a TSCH network typically frequently send Enhanced Beacon (EB) frames to announce the presence of the network. This document provides a mechanism by which small details critical for new nodes (pledges) and long sleeping nodes may be carried within the Enhanced Beacon.

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 July 20, 2018.

Copyright Notice

Table of Contents

1. Introduction 2
1.1. Terminology 2
1.2. Layer-2 Synchronization 3
1.3. Layer-3 synchronization IPv6 Router solicitations and advertisements 3
2. Protocol Definition 4
2.1. Protocol Example 4
3. Security Considerations 5
4. Privacy Considerations 5
5. IANA Considerations 5
6. Acknowledgements 5
7. References 5
7.1. Normative References 5
7.2. Informative References 6
Appendix A. Change history 6
Authors' Addresses 7

1. Introduction

[RFC7554] describes the use of the time-slotted channel hopping (TSCH) mode of [ieee802154]. As further details in [RFC8180], an Enhanced Beacon is transmitted during a slot designated a broadcast slot.

EDNOTE: Explain why broadcasts are rare, and why we need them. What the Enhanced Beacon is, and what Information Elements are, and how the IETF has a subtype for that area. Explain what kind of things could be placed in Information Elements, how big they could be, and how they could be compressed.

1.1. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [RFC2119] and indicate requirement levels for compliant STuPiD implementations.

1.2. Layer-2 Synchronization

As explained in section 6 of [RFC8180], the Enhanced Beacon has a number of purposes: synchronization of ASN and Join Metric, timeslot template identifier, the channel hopping sequence identifier, TSCH SlotFrame and Link IE.

The Enhanced Beacon (EB) is used by nodes already part of a TSCH network to annouce its existance. Receiving an EB allows a Joining Node (pledge) to learn about the network and synchronize to it. The EB may also be used as a means for a node already part of the network to re-synchronize [RFC7554].

There are a limited number of timeslots designated as a broadcast slot by each router. These slots are rare, and with 10ms slots, with a slot-frame length of 100, there may be only 1 slot/s for the beacon.

1.3. Layer-3 synchronization IPv6 Router solicitations and advertisements

At layer 3, [RFC2461] defines a mechanism by which nodes learn about routers by listening for multicasted Router Advertisements (RA). If no RA is heard within a set time, then a Router Solicitation (RS) may be multicast, to which an RA will be received, usually unicast.

Although [RFC6775] reduces the amount of multicast necessary to do address resolution via Neighbor Solicitation messages, it still requires multicast of either RAs or RS. This is an expensive operation for two reasons: there are few multicast timeslots for unsolicited RAs; if a pledge node does not hear an RA, and decides to send a RS (consuming a broadcast aloha slot with unencrypted traffic), many unicast RS may be sent in response.

This is a particularly acute issue for the join process for the following reasons:

1. use of a multicast slot by even a non-malicious unauthenticated node for a Router Solicitation may overwhelm that time slot.

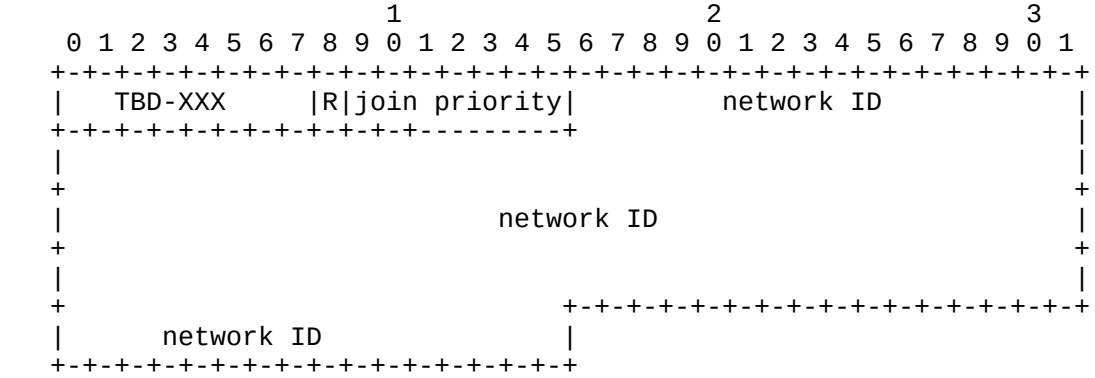
2. it may require many seconds of on-time before a new pledge hears a Router Soliciation that it can use.

3. a new pledge may listen to many Enhanced Beacons before it can pick an appropriate network and/or closest Join Assistant to attach to. If it must listen for a RS as well as find the Enhanced Beacon, then the process may take a very long time.

2. Protocol Definition

[RFC8137] creates a registry for new IETF IE subtypes. This document allocates a new subtype TBD-XXX.

This document documents a new IE subtype structure is as follows. As explained in [RFC8137] the length of the Sub-Type Content can be calculated from the container, so no length information is necessary.



J the Join prority value contains a number from 0 to 0x7f. Lower numbers are considered to be a higher preference. A priority of 0x7f indicates that the announcer should never be considered as a viable proxy. Lower value indicates willing to act as a Join Proxy as described in [I-D.ietf-6tisch-minimal-security].

R the Router Advertisement flag is set if the sending node will act as a Router for host-only nodes that need addressing via unicast Router Solicitation messages.

network ID this is an opaque 16-byte identifier that uniquely identifies this network, potentially among many networks that are operating in the same frequencies in overlapping physical space.

In a 6tisch network, where RPL is used as the mesh routing protocol, the network ID SHOULD be constructed from a SHA256 hash of the DODAGID of the network. The result will be a 32-byte hash, and the right-most 16-bytes should be used as the network ID.

2.1. Protocol Example

Here will be three examples of processing.

3. Security Considerations

All of the contents of this Information Element are sent in the clear. The containing Enhanced Beacon is not encrypted, but may be authenticated to nodes which have already received network-wide keying material.

4. Privacy Considerations

The use of a network ID may reveal information about the network. The use of a SHA256 hash of the DODAGID, rather than using the DODAGID directly provides some cover the addresses used within the network. The DODAGID is usually the IPv6 address of the root of the RPL mesh.

An interloper with a radio sniffer would be able to use the network ID to map out the extend of the mesh network.

5. IANA Considerations

Allocate a new number TBD-XXX from Registry IETF IE Sub-type ID. This entry should be called 6tisch-Join-Info.

6. Acknowledgements

Thomas Watteyne provided extensive editorial comments on the document.

7. References

7.1. Normative References

[I-D.ietf-6tisch-architecture]
Thubert, P., "An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4", draft-ietf-6tisch-architecture-13 (work in progress), November 2017.

[I-D.ietf-6tisch-minimal-security]
Vucinic, M., Simon, J., Pister, K., and M. Richardson, "Minimal Security Framework for 6TiSCH", draft-ietf-6tisch-minimal-security-04 (work in progress), October 2017.

[ieee802154]
IEEE Standard, ., "802.15.4-2015 - IEEE Standard for Low-Rate Wireless Personal Area Networks (WPANs)", 2015, <http://standards.ieee.org/findstds/standard/802.15.4-2015.html>.

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

[RFC2461] Narten, T., Nordmark, E., and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, DOI 10.17487/RFC2461, December 1998, <https://www.rfc-editor.org/info/rfc2461>.

[RFC6775] Shelby, Z., Ed., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, DOI 10.17487/RFC6775, November 2012, <https://www.rfc-editor.org/info/rfc6775>.

[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, <https://www.rfc-editor.org/info/rfc7554>.

[RFC8137] Kivinen, T. and P. Kinney, "IEEE 802.15.4 Information Element for the IETF", RFC 8137, DOI 10.17487/RFC8137, May 2017, <https://www.rfc-editor.org/info/rfc8137>.

7.2. Informative References

[I-D.ietf-6tisch-dtsecurity-secure-join] Richardson, M., "6tisch Secure Join protocol", draft-ietf-6tisch-dtsecurity-secure-join-01 (work in progress), February 2017.

[RFC8180] Vilajosana, X., Ed., Pister, K., and T. Watteyne, "Minimal IPv6 over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration", BCP 210, RFC 8180, DOI 10.17487/RFC8180, May 2017, <https://www.rfc-editor.org/info/rfc8180>.

Appendix A. Change history

This is an evolution of an earlier proposal which provided for storing an entire IPv6 Router Adverisement in an Informational Element. It was deemed too general a solution, possibly subject to mis-use. This proposal restricts the use to just the key pieces of information required.

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

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca

6TiSCH Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 29, 2017

M. Richardson
Sandelman Software Works

February 25, 2017

Minimal Security rekeying mechanism for 6TiSCH
draft-richardson-6tisch-minimal-rekey-01

Abstract

This draft describes a mechanism to rekey the networks used by 6TISCH nodes. It leverages the security association created during an enrollment protocol. The rekey mechanism permits incremental deployment of new sets of keys, followed by a rollover to a new key.

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 http://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 August 29, 2017.

Copyright Notice

Table of Contents

1. Introduction 2
2. Terminology 2
3. An approach to rekeying 3
4. YANG model for keys 4
5. YANG model for short-address 6
6. Security of CoMI link 8
7. Rekey of master connection 8
8. Privacy Considerations 8
9. Security Considerations 8
10. IANA Considerations 9
11. Acknowledgments 9
12. References 9
12.1. Normative References 9
12.2. Informative References 9
Appendix A. Example 10
Author's Address 10

1. Introduction

6TiSCH networks of nodes often use a pair of keys, K1/K2 to authenticate beacons (K1), encrypt broadcast traffic (K1) and encrypt unicast traffic (K2). These keys need to occasionally be refreshed for a number of reasons:

- cryptographic hygiene: the keys must be replaced before the ASN roles over or there could be repeated use of the same key.
- to remove nodes from the group: replacing the keys excludes any nodes that are suspect, or which are known to have left the network
- to recover short-addresses: if the JRC is running out of short (2-byte) addresses, it can rekey the network in order to garbage collect the set of addresses.

This protocol uses the CoMI [I-D.ietf-core-comi] to present the set of 127 key pairs.

In addition to providing for rekey, this protocol includes access to the allocated short-address.

2. Terminology

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 [RFC2119]. These

words may also appear in this document in lowercase, absent their normative meanings.

The reader is expected to be familiar with the terms and concepts defined in [I-D.ietf-6tisch-terminology], [RFC7252], [I-D.ietf-core-object-security], and [I-D.ietf-anima-bootstrapping-keyinfra].

3. An approach to rekeying

Rekeying of the network requires that all nodes be updated with the new keys. This can take time as the network is constrained, and this management traffic is not highest priority.

The JRC must reach out to all nodes that it is aware of. As the JRC has originally provided the keys via either zero-touch [I-D.ietf-6tisch-dtsecurity-secure-join] or [I-D.ietf-6tisch-minimal-security] protocol, and in each case, the JRC assigned the short-address to the node, so it knows about all the nodes.

The data model presented in this document provides for up to 127 K1/K2 keys, as each key requires a secKeyId, which is allocated from a 255-element palette provides by [IEEE8021542015]. Keys are to be updated in pairs, and the pairs are associated in the following way: the K1 key is always the odd numbered key (1,3,5), and the K2 key is the even numbered key that follows (2,4,6). A secKeyId value of 0 is invalid, and the secKeyId value of 255 is unused in this process.

Nodes MAY support up to all 127 key pair slots, but MUST support a minimum of 6 keys (3 slot-pairs). When fewer than 127 are supported, the node MUST support secKeyId values from 1 to 254 in a sparse array fashion.

A particular key slot-pair is considered active, and this model provides a mechanism to query and also to explicitely set the active pair.

Nodes decrypt any packets for which they have keys, but MUST continue to send using only the keypair which is considered active. Receipt of a packet which is encrypted (or authenticated in the case of a broadcast) with a secKeyId larger (taking consideration that secKeyId wraps at 254) than the active slot-pair causes the node to change active slot pairs.

This mechanism permits the JRC to provision new keys into all the nodes while the network continues to use the existing keys. When the JRC is certain that all (or enough) nodes have been provisioned with

the new keys, then the JRC causes a packet to be sent using the new key. This can be the JRC sending the next Enhanced Beacon or unicast traffic using the new key if the JRC is also a regular member of the LLN. In the likely case that the JRC has no direct connection to the LLN, then the JRC updates the active key to the new key pair using a CoMI message.

The frame goes out with the new keys, and upon receipt (and decryption) of the new frame all receiving nodes will switch to the new active key. Beacons or unicast traffic leaving those nodes will then update additional peers, and the network will switch over in a flood-fill fashion.

((EDNOTE: do we need an example?))

4. YANG model for keys

```
module ietf-6tisch-symmetric-keying {
  yang-version 1.1;

  namespace
    "urn:ietf:params:xml:ns:yang:6tisch-keys";
  prefix "ietf6keys";

  //import ietf-yang-types { prefix yang; }
  //import ietf-inet-types { prefix inet; }

  organization
   "IETF 6tisch Working Group";

  contact
   "WG Web:   <http://tools.ietf.org/wg/6tisch/>
    WG List:  <mailto:6tisch@ietf.org>
    Author:   Michael Richardson
              <mailto:mcr+ietf@sandelman.ca>";

  description
    "This module defines the format for a set of network-wide 802.15.4
    keys used in 6tisch networks.  There are 128 sets of key pairs,
    with one keypair (K1) used to authenticate (and sometimes encrypt)
    multicast traffic, and another keypair (K2) used to encrypt unicast
    traffic.  The 128 key pairs are numbered by the (lower) odd
    keyindex, which otherwise is a 0-255 value.  Keyindex 0 is
    not valid.  This module is a partial expression of the tables in
    https://mentor.ieee.org/802.15/dcn/15/15-15-0106-07-0mag-security-section-pi
ctures.pdf";

  revision "2017-03-01" {
    description
```

```
     "Initial version";
    reference
     "RFC XXXX: 6tisch minimal security";
  }

  // top-level container
  container ietf6tischkeys {
    config false;
    description
      "A voucher that can be used to assign one or more
       devices to an owner.";

    // secKeyIdMode is always 1, do not describe it here.
    leaf secKeyIndex {
      type uint8;
      description
        "The keyIndex for this keySet. A number between 1 and 255.";

      reference
        "IEEE802.15.4";
    }

    container secKeyUsage {
      leaf txPacketsSent {
        type uint32;
        description "Number of packets sent with this key.";
      }
      leaf rxPacketsSuccess {
        type uint32;
        description "Number of packets received with this key that were
                     successfully decrypted and authenticated.";
      }
      leaf rxPacketsReceived {
        type uint32;
        description "Number of packets received with this key, both
            successfully received, and unsuccessfully.";
      }

    }

    container secKeyDescriptor {
      description
        "This container describes the details of a specific cipher key";
      leaf secKey {
        type binary;
        description "The actual encryption key.
        This value is write only, and is not returned in a
        read, or returns all zeroes.";
```

```
      }
    }
  }
}
```

5. YANG model for short-address

```
module ietf-6tisch-symmetric-keying {
  yang-version 1.1;

  namespace
    "urn:ietf:params:xml:ns:yang:6tisch-keys";
  prefix "ietf6keys";

  //import ietf-yang-types { prefix yang; }
  //import ietf-inet-types { prefix inet; }

  organization
   "IETF 6tisch Working Group";

  contact
   "WG Web:   <http://tools.ietf.org/wg/6tisch/>
    WG List:  <mailto:6tisch@ietf.org>
    Author:   Michael Richardson
              <mailto:mcr+ietf@sandelman.ca>";

  description
    "This module defines the format for a set of network-wide
     802.15.4  keys used in 6tisch networks.  There are 128
     sets of key pairs, with one keypair (K1) used to
     authenticate (and sometimes encrypt) multicast traffic,
     and another keypair (K2) used to encrypt unicast traffic.
     The 128 key pairs are numbered by the (lower) odd
     keyindex, which otherwise is a 0-255 value.
     Keyindex 0 is not valid.  This module is a partial
     expression of the tables in
https://mentor.ieee.org/802.15/dcn/15/15-15-0106-07-0mag-security-section-pictur
es.pdf";

  revision "2017-03-01" {
    description
     "Initial version";
    reference
     "RFC XXXX: 6tisch minimal security";
  }

  // top-level container
  container ietf6tischkeys {
    config false;
```

```
    description
      "A voucher that can be used to assign one or more
       devices to an owner.";

    // secKeyIdMode is always 1, do not describe it here.
    leaf secKeyIndex {
      type uint8;
      description
        "The keyIndex for this keySet. A number between
         1 and 255.";

      reference
        "IEEE802.15.4";
    }

    container secKeyUsage {
      leaf txPacketsSent {
        type uint32;
        description "Number of packets sent with this key.";
      }
      leaf rxPacketsSuccess {
        type uint32;
        description "Number of packets received with this key
            that were successfully decrypted and authenticated.";
      }
      leaf rxPacketsReceived {
        type uint32;
        description "Number of packets received with this key,
            both successfully received, and unsuccessfully.";
      }

    }

    container secKeyDescriptor {
      description
        "This container describes the details of a specific
         cipher key";
      leaf secKey {
        type binary;
        description "The actual encryption key";
      }
    }
  }
}
```

6. Security of CoMI link

The CoMI resources presented here are protected by OSCOAP ([I-D.ietf-core-object-security]), secured using the EDHOC connection used for joining. A unique application key is generated using an additional key generation process with the unique label "6tisch-rekey".

7. Rekey of master connection

Should the OSCOAP connection need to be rekeyed, a new EDHOC process will be necessary. This will need access to trusted authentication keys, either the PSK used from a one-touch process, or the locally significant domain certificates installed during a zero-touch process.

8. Privacy Considerations

The rekey protocol itself runs over a network encrypted with the K2 key. The end to end protocol from JRC to node is also encrypted using OSCOAP, so the keys are not visible, nor is the keying traffic distinguished in anyway to an observer.

As the secKeyId is not confidential in the underlying 802.15.4 frames, an observer can determine what sets of keys are in use, and when a rekey is activated by observing the change in the secKeyId.

The absolute value of the monitonically increasing secKeyId could provide some information as to the age of the network.

9. Security Considerations

This protocol permits the underlying network keys to be set. Access to all of the portions of this interface MUST be restricted to an ultimately trusted peer, such as the JRC.

An implementation SHOULD not permit reading the network keys. Those fields should be write-only.

The OSCOAP security for this interface is initialized by a join mechanism, and so depends upon the initial credentials provided to the node. The initial network keys would have been provided during the join process; this protocol permits them to be updated.

10. IANA Considerations

This document allocates a SID number for the YANG model. There is no IANA action required for this document.

11. Acknowledgments

12. References

12.1. Normative References

[I-D.ietf-core-comi]
Stok, P., Bierman, A., Veillette, M., and A. Pelov, "CoAP Management Interface", draft-ietf-core-comi-00 (work in progress), January 2017.

[I-D.ietf-core-object-security]
Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security of CoAP (OSCOAP)", draft-ietf-core-object-security-01 (work in progress), December 2016.

[I-D.ietf-cose-msg]
Schaad, J., "CBOR Object Signing and Encryption (COSE)", draft-ietf-cose-msg-24 (work in progress), November 2016.

[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>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <http://www.rfc-editor.org/info/rfc7049>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <http://www.rfc-editor.org/info/rfc7252>.

12.2. Informative References

[I-D.ietf-6tisch-6top-protocol]
Wang, Q. and X. Vilajosana, "6top Protocol (6P)", draft-ietf-6tisch-6top-protocol-03 (work in progress), October 2016.

[I-D.ietf-6tisch-dtsecurity-secure-join]
Richardson, M., "6tisch Secure Join protocol", draft-ietf-6tisch-dtsecurity-secure-join-00 (work in progress), December 2016.

[I-D.ietf-6tisch-minimal-security]
Vucinic, M., Simon, J., and K. Pister, "Minimal Security Framework for 6TiSCH", draft-ietf-6tisch-minimal-security-01 (work in progress), February 2017.

[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-08 (work in progress), December 2016.

[I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., Bjarnason, S., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-04 (work in progress), October 2016.

[IEEE8021542015]
IEEE standard for Information Technology, ., "IEEE Std 802.15.4-2015 Standard for Low-Rate Wireless Personal Area Networks (WPANs)", 2015.

Appendix A. Example

Example COMI requests/responses.

Author's Address

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca

6TiSCH Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 1, 2018

M. Richardson
Sandelman Software Works

August 28, 2017

Minimal Security rekeying mechanism for 6TiSCH
draft-richardson-6tisch-minimal-rekey-02

Abstract

This draft describes a mechanism to rekey the networks used by 6TISCH nodes. It leverages the security association created during an enrollment protocol. The rekey mechanism permits incremental deployment of new sets of keys, followed by a rollover to a new key.

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 http://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 March 1, 2018.

Table of Contents

1. Introduction 2
2. Terminology 3
3. Tree diagram notation 3
4. An approach to rekeying 3
5. YANG models 4
5.1. Tree diagram 5
5.2. YANG model for keys 5
5.3. YANG model for short-address 8
6. Security of CoMI link 10
7. Rekey of master connection 10
8. Privacy Considerations 10
9. Security Considerations 10
10. IANA Considerations 11
11. Acknowledgments 11
12. References 11
12.1. Normative References 11
12.2. Informative References 12
Appendix A. Example 12
Author's Address 13

1. Introduction

6TiSCH networks of nodes often use a pair of keys, K1/K2 to authenticate beacons (K1), encrypt broadcast traffic (K1) and encrypt unicast traffic (K2). These keys need to occasionally be refreshed for a number of reasons:

- cryptographic hygiene: the keys must be replaced before the ASN roles over or there could be repeated use of the same key.
- to remove nodes from the group: replacing the keys excludes any nodes that are suspect, or which are known to have left the network
- to recover short-addresses: if the JRC is running out of short (2-byte) addresses, it can rekey the network in order to garbage collect the set of addresses.

This protocol uses the CoMI [I-D.ietf-core-comi] to present the set of 127 key pairs.

In addition to providing for rekey, this protocol includes access to the allocated short-address.

2. Terminology

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 [RFC2119]. These words may also appear in this document in lowercase, absent their normative meanings.

The reader is expected to be familiar with the terms and concepts defined in [I-D.ietf-6tisch-terminology], [RFC7252], [I-D.ietf-core-object-security], and [I-D.ietf-anima-bootstrapping-keyinfra].

3. Tree diagram notation

A simplified graphical representation of the data models is used in this document. The meaning of the symbols in these diagrams is as follows:

- Brackets "[" and "]" enclose list keys.
- Braces "{" and "}" enclose feature names, and indicate that the named feature must be present for the subtree to be present.
- Abbreviations before data node names: "rw" (read-write) represents configuration data and "ro" (read-only) represents state data.
- Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- Ellipsis ("...") stands for contents of subtrees that are not shown.

4. An approach to rekeying

Rekeying of the network requires that all nodes be updated with the new keys. This can take time as the network is constrained, and this management traffic is not highest priority.

The JRC must reach out to all nodes that it is aware of. As the JRC has originally provided the keys via either zero-touch [I-D.ietf-6tisch-dtsecurity-secure-join] or [I-D.ietf-6tisch-minimal-security] protocol, and in each case, the JRC assigned the short-address to the node, so it knows about all the nodes.

The data model presented in this document provides for up to 127 K1/ K2 keys, as each key requires a secKeyId, which is allocated from a 255-element palette provides by [IEEE8021542015]. Keys are to be updated in pairs, and the pairs are associated in the following way: the K1 key is always the odd numbered key (1,3,5), and the K2 key is the even numbered key that follows (2,4,6). A secKeyId value of 0 is invalid, and the secKeyId value of 255 is unused in this process.

Nodes MAY support up to all 127 key pair slots, but MUST support a minimum of 6 keys (3 slot-pairs). When fewer than 127 are supported, the node MUST support secKeyId values from 1 to 254 in a sparse array fashion.

A particular key slot-pair is considered active, and this model provides a mechanism to query and also to explicitely set the active pair.

Nodes decrypt any packets for which they have keys, but MUST continue to send using only the keypair which is considered active. Receipt of a packet which is encrypted (or authenticated in the case of a broadcast) with a secKeyId larger (taking consideration that secKeyId wraps at 254) than the active slot-pair causes the node to change active slot pairs.

This mechanism permits the JRC to provision new keys into all the nodes while the network continues to use the existing keys. When the JRC is certain that all (or enough) nodes have been provisioned with the new keys, then the JRC causes a packet to be sent using the new key. This can be the JRC sending the next Enhanced Beacon or unicast traffic using the new key if the JRC is also a regular member of the LLN. In the likely case that the JRC has no direct connection to the LLN, then the JRC updates the active key to the new key pair using a CoMI message.

The frame goes out with the new keys, and upon receipt (and decryption) of the new frame all receiving nodes will switch to the new active key. Beacons or unicast traffic leaving those nodes will then update additional peers, and the network will switch over in a flood-fill fashion.

((EDNOTE: do we need an example?))

5. YANG models

5.1. Tree diagram

A diagram of the two YANG modules looks like:

```
1700  module: ietf-6tisch-symmetric-keying
1701      +--rw ietf6tischkeypairs* [counter]
1702      |  +--rw counter           uint16
1703      |  +--rw ietf6tischkey1
1704      |  |  +--rw secKeyDescriptor
1705      |  |  |  +--rw secKey?   binary
1706      |  |  +--rw secKeyIndex?        uint8
1707      |  +--rw ietf6tischkey2
1708      |     +--rw secKeyDescriptor
1709      |     |  +--rw secKey?   binary
1710      |     +--rw secKeyIndex?        uint8
1711      +--ro secKeyUsage
1712      |  +--ro txPacketsSent?       uint32
1713      |  +--ro rxPacketsSuccess?    uint32
1714      |  +--ro rxPacketsReceived?   uint32
1715      +--rw newKey?               binary

         rpcs:
1716       +---x installNextKey

1717  module: ietf-6tisch-short-address
1718      +--ro ietf6shortaddresses
1719         +--ro shortaddress    binary
1720         +--ro validuntil      uint32
1721         +--ro effectiveat?    uint32
```

Figure 1: Tree diagrams of two rekey modules

5.2. YANG model for keys

```
module ietf-6tisch-symmetric-keying {
  yang-version 1.1;

  namespace
    "urn:ietf:params:xml:ns:yang:ietf-6tisch-symmetric-keying";
  prefix "ietf6keys";

  //import ietf-yang-types { prefix yang; }
  //import ietf-inet-types { prefix inet; }

  organization
   "IETF 6tisch Working Group";
```

```
  contact
   "WG Web:   <http://tools.ietf.org/wg/6tisch/>
    WG List:  <mailto:6tisch@ietf.org>
    Author:   Michael Richardson
              <mailto:mcr+ietf@sandelman.ca>";

  description
    "This module defines the format for a set of network-wide 802.15.4
    keys used in 6tisch networks.  There are 128 sets of key pairs,
    with one keypair (K1) used to authenticate (and sometimes encrypt)
    multicast traffic, and another keypair (K2) used to encrypt unicast
    traffic.  The 128 key pairs are numbered by the (lower) odd
    keyindex, which otherwise is a 0-255 value.  Keyindex 0 is
    not valid.  This module is a partial expression of the tables in
    https://mentor.ieee.org/802.15/dcn/15/15-15-0106-07-0mag-security-section-pi
ctures.pdf.
To read and write the key pairs, a monotonically increasing counter is added. A
new key pair must be added with current_counter = last_counter+1. The current sp
ecification allows overwriting of earlier key pairs. It is up to the server to r
emove old key pairs, such that only the last three (two) pairs are stored and vi
sible to the client.";

  revision "2017-03-01" {
    description
     "Initial version";
    reference
     "RFC XXXX: 6tisch minimal security";
  }

  // list of key pairs
  list ietf6tischkeypairs {
  key counter;
  description
    "a list of key pairs with unique index: counter.";
  leaf  counter {
  type uint16{
     range "0..256";  // for the moment 256 items
    }
  mandatory "true";
  description
    "unique reference to the key pair for client access.";
  }  // counter

  // key descriptor for FIRST part of pair
  container ietf6tischkey1 {
    description
      "A voucher that can be used to assign one or more
       devices to an owner.";
// this container is pretty empty, a leaf would do the job.

    container secKeyDescriptor {
  // I assume this needs to be extended, why else a container?
      description
```

```
        "This container describes the details of a
           specific cipher key";
      leaf secKey {
        type binary;
        description "The actual encryption key.
          This value is write only, and is not returned in a
          read, or returns all zeroes.";
      } // secKey
    }  // secKeyDescriptor

    // leaf secKeyIdMode is always 1, not described here.
    leaf secKeyIndex {
      type uint8;
      description
        "The keyIndex for this keySet.
            A number between 1 and 255.";
      reference
        "IEEE802.15.4";
    } // secKeyIndex
 } // ietf6tischkey1

// key descriptor for SECOND part of pair
  container ietf6tischkey2 {
    description
      "A voucher that can be used to assign one or more
       devices to an owner.";
    container secKeyDescriptor {
  // I assume this needs to be extended, why else a container?
      description
        "This container describes the details of a
           specific cipher key";
      leaf secKey {
        type binary;
        description "The actual encryption key.
          This value is write only, and is not returned in a
          read, or returns all zeroes.";
      } // secKey
    }  // secKeyDescriptor

    // leaf secKeyIdMode is always 1, not described here.
    leaf secKeyIndex {
      type uint8;
      description
        "The keyIndex for this keySet.
           A number between 1 and 255.";
      reference
        "IEEE802.15.4";
    } // secKeyIndex
```

```
   } // ietf6tischkey2
 } //ietf6tischkeypairs

// the usage is over all pairs
    container secKeyUsage {
    config false; // cannot be set by client
    description
      "statistics of sent and received packets.";
      leaf txPacketsSent {
        type uint32;
        description "Number of packets sent with this key.";
      } // txPacketsSent
      leaf rxPacketsSuccess {
        type uint32;
        description "Number of packets received with this key that were
                     successfully decrypted and authenticated.";
      }// rxPacketsSuccess
      leaf rxPacketsReceived {
        type uint32;
        description "Number of packets received with this key, both
            successfully received, and unsuccessfully.";
      } // rxPacketsReceived

    } // secKeyUsage

 // setting new key, and validation of new key
     leaf newKey{
        type binary;
        description
        "new key value to be set by client.";
     }  // newKey
     rpc installNextKey{
         description
         "Client informs server that newKey is to be
         used as current key.";
     }  // installNextKey


  } // module ietf-6tisch-symmetric-keying
```

5.3. YANG model for short-address

```
   module ietf-6tisch-short-address {
     yang-version 1.1;

     namespace
       "urn:ietf:params:xml:ns:yang:ietf-6tisch-short-address";
     prefix "ietf6shortaddr";
```

```
     //import ietf-yang-types { prefix yang; }
     //import ietf-inet-types { prefix inet; }

     organization
      "IETF 6tisch Working Group";

     contact
      "WG Web:   <http://tools.ietf.org/wg/6tisch/>
       WG List:  <mailto:6tisch@ietf.org>
       Author:   Michael Richardson
                 <mailto:mcr+ietf@sandelman.ca>";

     description
      "This module defines an interface to set and interrogate
       the short (16-bit) layer-2 address used in 802.15.4
       TSCH mode networks.  The short addresses are used
       in L2 frames to save space.  A lifetime is included
       in terms of TSCH Absolute Slot Number, which acts
       as a monotonically increasing clock.  ";

     revision "2017-03-01" {
       description
        "Initial version";
       reference
        "RFC XXXX: 6tisch minimal security";
     }

     // top-level container
     container ietf6shortaddresses {
       config false;
       description
         "A 16-bit short address for use by the node.";

       leaf shortaddress {
         type binary{
            length 1..2;}
         mandatory true;
         description
           "The two byte short address to be set.";
       }
       leaf validuntil {
         type uint32;
         mandatory true;
         description "The Absolute Slot Number/256 at which
                      the address ceases to be valid.";
       }

       leaf effectiveat {
```

```
        type uint32;
        description "The Absolute Slot Number/256 at which
                     time the address was originally set.
                     This is a read-only attribute that
                     records the ASN when the shortaddress
                     element was last written or updated.";
      }
    }
  }
```

6. Security of CoMI link

The CoMI resources presented here are protected by OSCOAP ([I-D.ietf-core-object-security]), secured using the EDHOC connection used for joining. A unique application key is generated using an additional key generation process with the unique label "6tisch-rekey".

7. Rekey of master connection

Should the OSCOAP connection need to be rekeyed, a new EDHOC process will be necessary. This will need access to trusted authentication keys, either the PSK used from a one-touch process, or the locally significant domain certificates installed during a zero-touch process.

8. Privacy Considerations

The rekey protocol itself runs over a network encrypted with the K2 key. The end to end protocol from JRC to node is also encrypted using OSCOAP, so the keys are not visible, nor is the keying traffic distinguished in anyway to an observer.

As the secKeyId is not confidential in the underlying 802.15.4 frames, an observer can determine what sets of keys are in use, and when a rekey is activated by observing the change in the secKeyId.

The absolute value of the monitonically increasing secKeyId could provide some information as to the age of the network.

9. Security Considerations

This protocol permits the underlying network keys to be set. Access to all of the portions of this interface MUST be restricted to an ultimately trusted peer, such as the JRC.

An implementation SHOULD not permit reading the network keys. Those fields should be write-only.

The OSCOAP security for this interface is initialized by a join mechanism, and so depends upon the initial credentials provided to the node. The initial network keys would have been provided during the join process; this protocol permits them to be updated.

10. IANA Considerations

This document allocates a SID number for the YANG model. There is no IANA action required for this document.

11. Acknowledgments

12. References

12.1. Normative References

[I-D.ietf-core-comi]
Veillette, M., Stok, P., Pelov, A., and A. Bierman, "CoAP Management Interface", draft-ietf-core-comi-01 (work in progress), July 2017.

[I-D.ietf-core-object-security]
Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security of CoAP (OSCOAP)", draft-ietf-core-object-security-04 (work in progress), July 2017.

[I-D.ietf-cose-msg]
Schaad, J., "CBOR Object Signing and Encryption (COSE)", draft-ietf-cose-msg-24 (work in progress), November 2016.

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

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <https://www.rfc-editor.org/info/rfc7049>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

12.2. Informative References

[I-D.ietf-6tisch-6top-protocol]
Wang, Q., Vilajosana, X., and T. Watteyne, "6top Protocol (6P)", draft-ietf-6tisch-6top-protocol-07 (work in progress), June 2017.

[I-D.ietf-6tisch-dtsecurity-secure-join]
Richardson, M., "6tisch Secure Join protocol", draft-ietf-6tisch-dtsecurity-secure-join-01 (work in progress), February 2017.

[I-D.ietf-6tisch-minimal-security]
Vucinic, M., Simon, J., Pister, K., and M. Richardson, "Minimal Security Framework for 6TiSCH", draft-ietf-6tisch-minimal-security-03 (work in progress), June 2017.

[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-09 (work in progress), June 2017.

[I-D.ietf-anima-bootstrapping-keyinfra]
Pritikin, M., Richardson, M., Behringer, M., Bjarnason, S., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructures (BRSKI)", draft-ietf-anima-bootstrapping-keyinfra-07 (work in progress), July 2017.

[IEEE8021542015]
IEEE standard for Information Technology, ., "IEEE Std 802.15.4-2015 Standard for Low-Rate Wireless Personal Area Networks (WPANs)", 2015.

Appendix A. Example

In the examples below, a new key value is set in the server example.com; followed by the setting of the new key value as the current key value. The SID values of new Key and installNextKey are 1715 and 1716 respectively. The corresponding base64 values are: ez and e0 respectively.

The setting of the new key value is done with the PUT request with the binary value 1234567890.

```
   PUT coap://example.com/c/ez
      (Content-Format :application/yang-value+cbor)
   h'1234567890'

   RES: 2.01 Created

   Payload in CBOR:
   45            # bytes(5)
      1234567890 # "\x124Vx\x90"
```

Consecutively, the RPC is invoked with a POST method to validate the new key value.

```
   POST coap://example.com/c/e0
      (Content-Format :application/yang-value+cbor)

   RES: 2.05 Content
```

The client can query how many TX packets have been received. The SID of secKeyUsage/txPacketsSent is 1712, corresponding with base64 ew.

```
   GET coap://example.com/c/ew

   RES: 2.05 Content (Content-Format :application/yang-value+cbor)
   3

   Payload in CBOR:
   03 # unsigned(3)
```

Author's Address

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca

DetNet
Internet Draft
Interned status: Standards Track
Expires: June 22, 2017

H. Wang
P. Wang
C. Zhang
Y. Yang
Chongqing University of
Posts and Telecommunications

December 19, 2016

Joint Scheduling Architecture for Deterministic Industrial Field/Backhaul Networks
draft-wang-detnet-backhaul-architecture-00

Abstract

Joint scheduling of industrial field network and backhaul network is significant for end-to-end deterministic delay requirements of data flows in factories. This document describes a joint scheduling architecture for deterministic industrial field and backhaul networks. Taking WIA-PA wireless field network and IPv6-based backhaul network as an example, this document shows how the joint scheduling architecture works.

Table of Contents

1. Introduction 2
2. Joint Scheduling Architecture..... 3
2.1. Distributed Architecture..... 4
2.2. Centralized Architecture..... 5
2.3. Joint Scheduling Architecture..... 6
3. Joint Scheduling Scheme..... 8
3.1. WIA-PA Network Joint Scheduling 9
3.2. Protocol Conversion..... 9
3.3. Industrial Backhaul Network Scheduling 11
4. Security Considerations..... 13
5. IANA Considerations 13
6. References 13
6.1. Normative References..... 13
6.2. Informative References..... 13

1. Introduction

Deterministic network is an essential element of the industrial network. Using deterministic network in the industrial field can enhance the network performance and greatly reduce the network packet loss. Thus, it is the future development direction of industrial network technology to use deterministic networks in the whole industrial network. Deterministic networks in industrial networks are mainly concentrated on the industrial field networks, such as ISA100.11a[IEC62734], WirelessHART[IEC62591] and WIA-PA[IEC62601], and there is little joint scheduling scheme that can be applied to industrial networks.

Nowadays, in the use case document[draft-bas-usecase-detnet] and architecture document[draft-finn-detnet-architecture] submitted by the IETF DetNet working group, a deterministic network based on Ethernet has already been researched. The document proposes a network architecture based on SDN technology, which can accurately control the transmission of data streams. However, the document does not consider the characteristics of the industrial backhaul networks and the actual situation of other industrial field deterministic networks. First of all, the data flow of industrial backhaul network is highly sensitive to the uncertainty of time. Therefore, it is very important that how to apply the deterministic networks based on Ethernet to industrial backhaul networks. Secondly, the existing deterministic networks in the industrial field have been widely deployed in the factory, and Deterministic network technology is already very mature, and direct replacement will consume a lot of manpower and material resources.

Based on existing work in the architecture document[draft-finn-detnet-architecture], this document proposes a joint scheduling architecture for deterministic industrial field networks. This framework will firstly replace the industrial backhaul networks and other non-deterministic networks of industrial networks into deterministic Ethernet-based network, and then on the basis of SDN technology, this document proposes a joint scheduler, which can be used for joint scheduling on other deterministic networks in deterministic Ethernet-based network and industrial field network. Through deploying the deterministic network throughout the industrial network based on the joint scheduling architecture, it can realize the end-to-end deterministic scheduling between different industrial field networks, and ensure data stream indicators as well as save manpower and material resources.

2. Joint Scheduling Architecture

For industrial networks, there are many network controllers in the network, which together constitute the control plane for the whole industrial network. The control plane is very important in the entire network, especially when it comes to cross domain transfer of time-sensitive data. So the control plane architecture will greatly affect the performance of the network, therefore it is becoming a research hotspot on how to give full play to the performance of their respective networks when the multiple controllers are in the joint cooperation. However, there is not a unified standard of joint architecture of multiple controllers in the industry at present. The main frameworks are the following two kinds: the distributed architecture and the centralized architecture. The WIA-PA network, which is the typical of WSNs standards which has become an

international standard for industrial field networks approved by IEC, is used as an example to illustrate these architectures.

2.1. Distributed Architecture

Distributed architecture is also known as East-West architecture. In the architecture, the status of all network controller is equal, these controllers are connected to each other to form an unstructured network, and achieve cross domain transfer task deployment through the mutual transmission of information, as shown in Figure 1.

In the distributed architecture, the controller can exchange different network topologies and the accessibility of information through the east-west interface, and each controller can build a global network topology. In the access to the global network topology, since each controller is equal, it can serve as a server role at the same time, as well as has the service capacity of starting deterministic cross-network transmission.

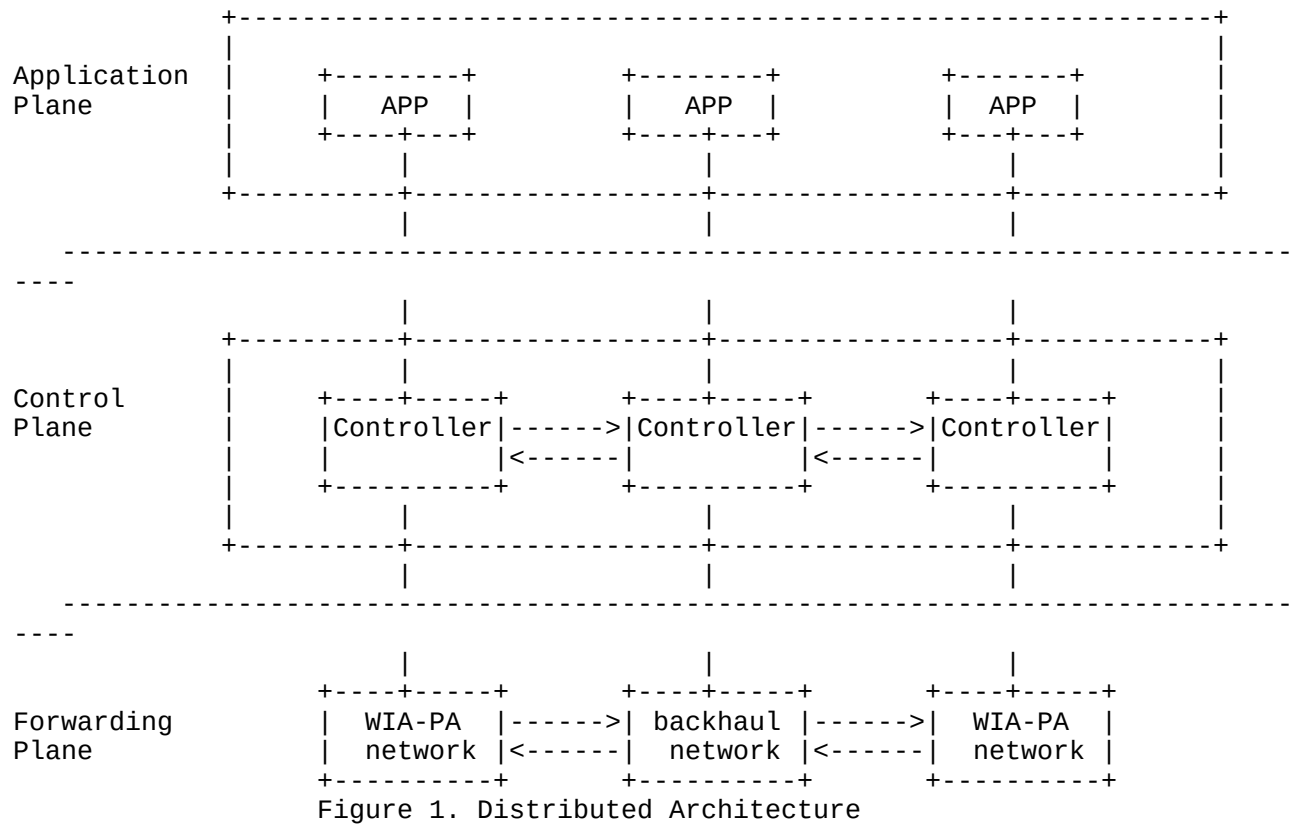


Figure 1. Distributed Architecture

2.2. Centralized Architecture

Centralized architecture is also known as vertical multi-level architecture. In this architecture, the control plane is divided into two parts, one is the basic control plane composed of a variety of network controllers; another part is a network controller composed of the main controller, which is responsible for controlling the basic control plane, as shown in Figure 2.

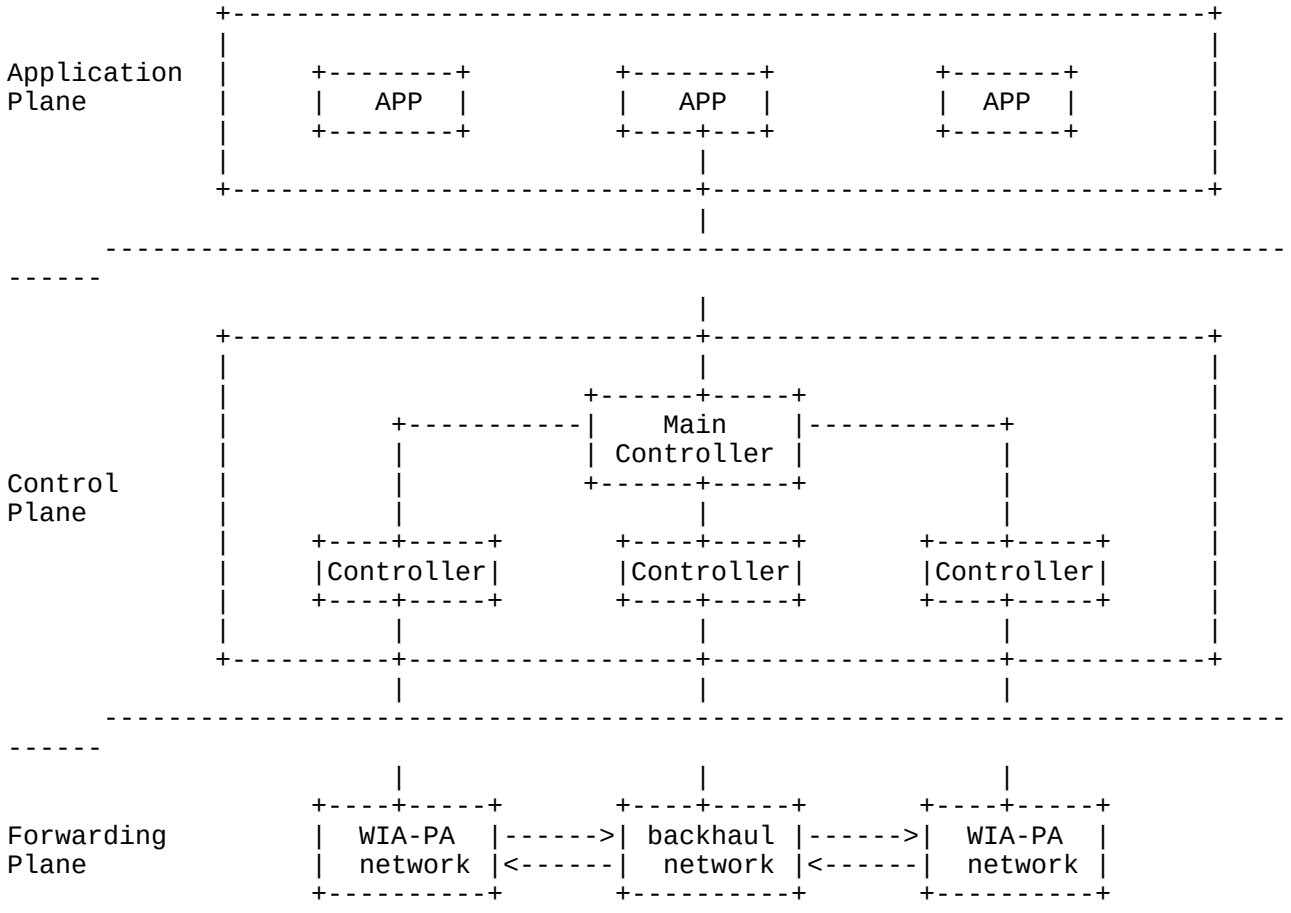


Figure 2. Centralized Architecture

The centralized architecture needn’t to expand the east-west interface. It only needs to establish a connection with the basic controllers through the southbound interface. After the connection

is established, the main controller obtains the every domain network topology through the API interface provided by the basic controllers, and storages global network topology on its own. It can also assign tasks to basic controllers through the API interface.

2.3. Joint Scheduling Architecture

In the practical application, distributed architecture not only needs to extend the east-west interface, but also maintains a global network topology in each controller. Only each controller maintains such a global network topology, it can ensure the deterministic control of the control plane for the whole network.

Though the centralized architecture does not have the above requirements, for the deterministic industrial network, the scale of the network is not very large, in the industrial backhaul network, a single SDN controller is sufficient to meet the control demands of industrial backhaul network. If centralized architecture is directly applied to an industrial network, it will not only be unable to give full play to the advantages of the architecture in multi controllers collaboration, but also cause meaningless information interaction between the controllers, which will waste network resource.

In view of the problems existing in these two architectures, this document takes the WIA-PA network as an example and proposes a joint scheduling architecture based on the architecture document[draft-finn-detnet-architecture]. The architecture is optimized according to the characteristics of deterministic industrial network, so that a single SDN controller can unite the WIA-PA network systems manager to manage the entire industrial network, and provide support for the deterministic scheduling of data streams across network transmission through industrial backhaul network located in different domains of WIA-PA network.

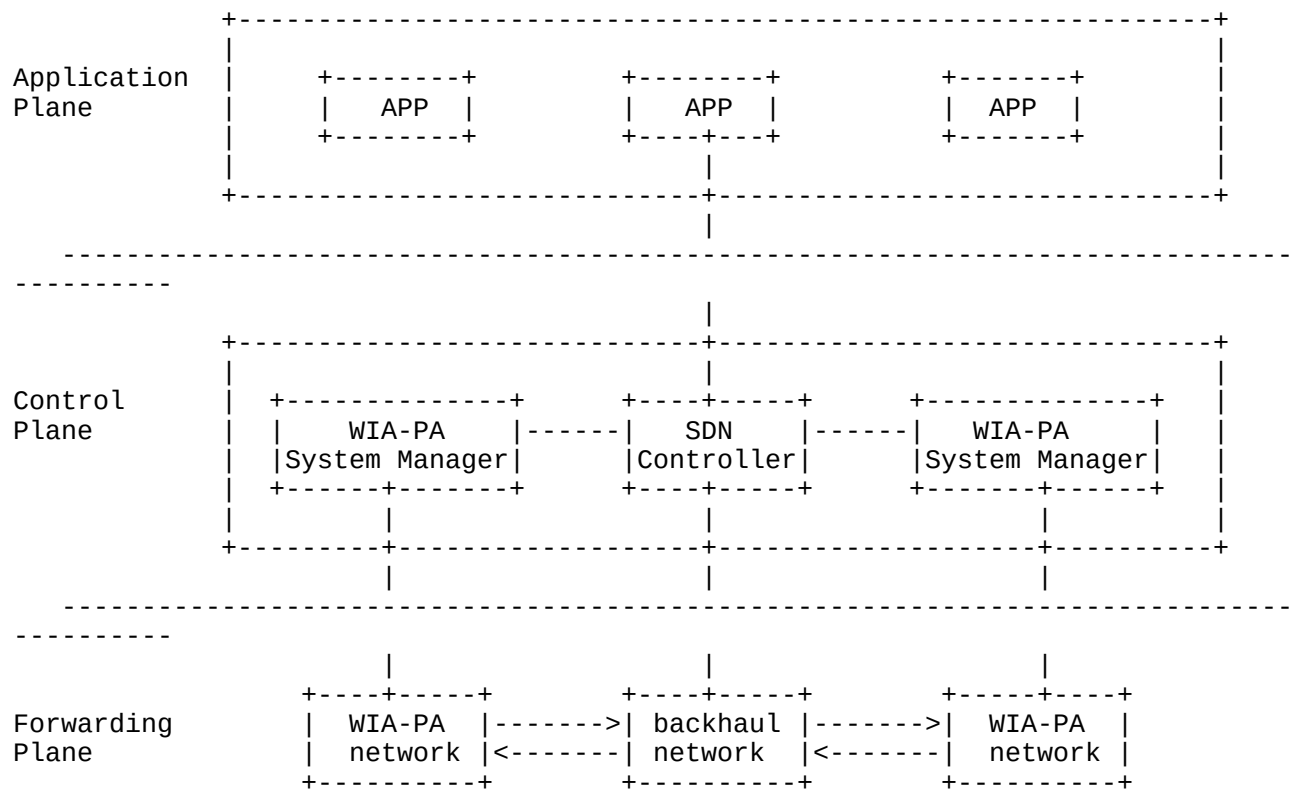


Figure 3. Joint scheduling architecture

As shown in Figure 3, joint scheduling architecture can be mainly classified into three planes:

- Forwarding plane: this plane contains various types of network equipment in different networks. It is the physical entities of the network transmission. In general, to achieve the desired network functions for the network manager, these devices are specific factors of management control operation, which makes their own resources abstract for their own control elements to manage and configure.

- Control plane: this plane is formed by the WIA-PA System Manager and the SDN controller. Joint scheduler is integrated into the SDN controller in the form of plugin, and other WIA-PA System Managers accept joint management scheduler by establishing a connection with the SDN controller. Meanwhile, inside the SDN controller, joint scheduler achieves the management of industrial backhaul network by directly calling the corresponding module of SDN controller.

o Application plane: this plane provides users with a unified interface about a variety of resources for the whole network. At the same time, it also provides users with an intuitive, user-friendly interface, which can shield the complex network information of the original.

Joint Scheduling Architecture defines an architecture that when industrial networks contain other deterministic networks, these deterministic networks and deterministic Ethernet-based networks are jointly scheduling. On the basis of this architecture, control and scheduling for the entire industrial network can be realized by joint scheduler, so as to provide a real-time protection for each data stream.

3. Joint Scheduling Scheme

Taking WIA-PA wireless field network and IPv6-based backhaul network as an example, this section shows how the joint scheduling architecture works. Existing WIA-PA scheduling scheme only applies to WIA-PA field network. Scheduling scheme will fail once the data is transferred to backhaul networks. Joint scheduling scheme is innovation and expansion of WIA-PA scheduling scheme.

Firstly, scheduling scheme based on SDN in industry backhaul network is added to the original scheduling scheme, so that data can flow in the industrial backhaul network, and the data can be identified and assigned existing backhaul network resource according to their requirements for the network resources.

Secondly, conducting an optimization for original WIA-PA scheduling scheme enables scheduling scheme based on WIA-PA networks plays together joint scheduler, and scheduling scheme can simultaneously apply to two non-adjacent domains so that it can be adapt to the cross-border joint operation based on SDN.

Thirdly, due to the specificity of cross-border transmission services, the joint scheduling scheme for WIA-PA network VCR_ID and Route ID is reclassified.

Finally, since the system manager allocates a short address to the field device on the basis of the network address information about its own domain in WIA-PA networks. Thus resulting in the entire network short address field device is uncertain. In order to identify the field device on different network domains and domain, the network identifier (PAN_ID) is applied to the joint scheduling scheme to identify WIA-PA network.

After the SDN controller initiates joint scheduling module, WIA-PA system manager will actively establish a connection with the united scheduler. After the scheduler receives a cross-border transmission request, joint scheduler will send a request for obtaining topology information and node information to WIA-PA System Manager. Then, the scheduler will assign paths and network resources according to this information by pre-defined scheduling algorithm.

After the routing and network resources have been calculated, joint scheduler will configure and deploy networks by the corresponding network controller.

3.1. WIA-PA Network Joint Scheduling

In the united scheduling process, path deployment and resource allocation for WIA-PA network are performed by calling the WIA-PA network system manager API interface. System manager will query the corresponding information of the field device in the network upon receiving the acquisition command of joint operation for the network information, and then return the received information to the united scheduler. The system manager will configure communication resources for the corresponding gateway device, routing equipment and field equipment if the system manager receives configuration commands from joint scheduler. After receiving a successful response, it will send a successful reply to the united scheduler.

3.2. Protocol Conversion

In the process of cross-border transmission, since industrial backhaul network is different from WIA-PA network, which is not an IP-based Ethernet. Protocol conversion of gateway for WIA-PA packet is needed when the data of WIA-PA network needs to transmit to another network through cross-border industrial backhaul. Meanwhile, according to the joint scheduling scheme, SDN controller is able to identify the WIA-PA Ethernet data stream, and allocate resources according to the data stream type and level of the data stream. Therefore, in the protocol conversion process of gateway, scheduling and control of WIA-PA data flow can be realized by SDN controller unless the VCR of WIA-PA data stream and the priority are filled in the IPv6 header.

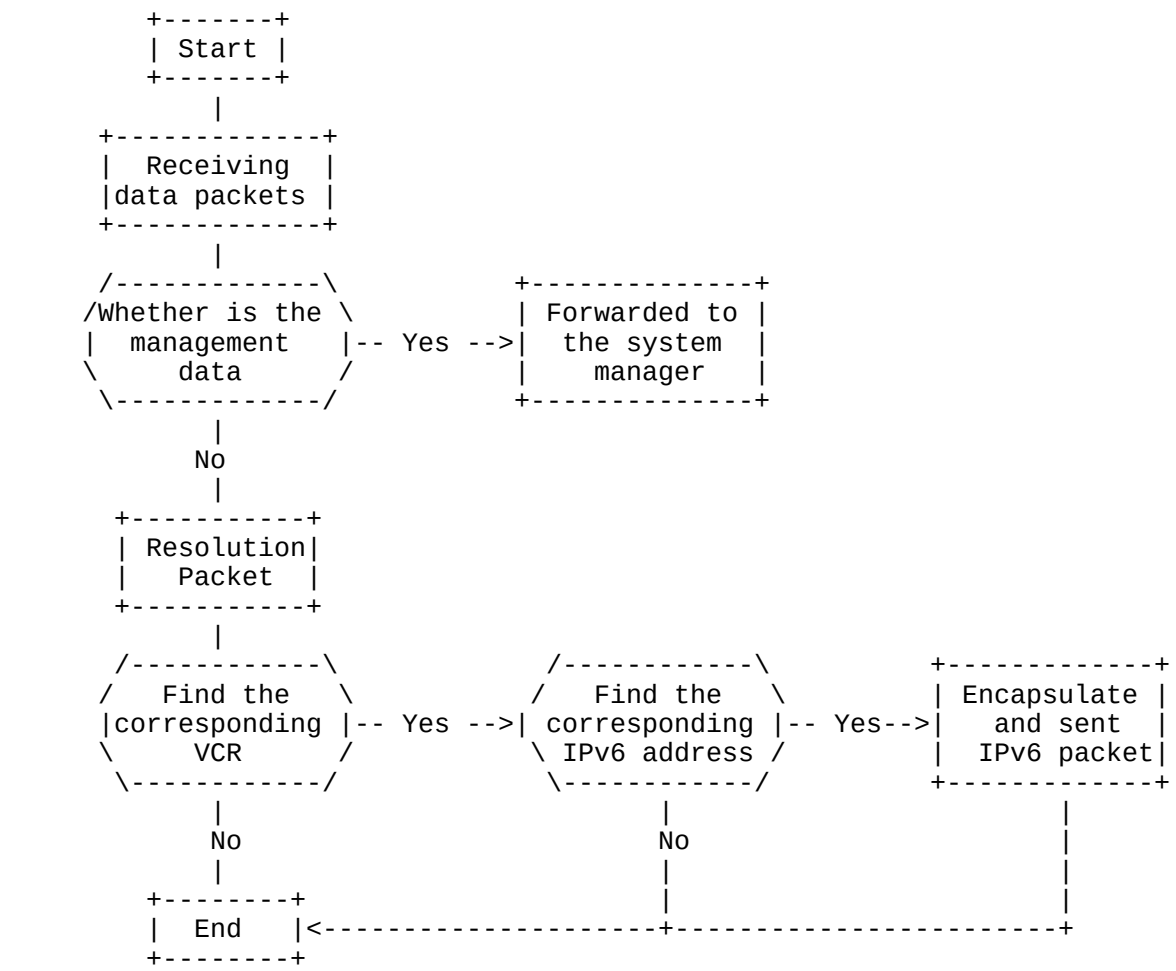


Figure 4. The conversion process of gateway protocol

As shown in Figure 4, according to the above section, the gateway will receive the address mapping of joint scheduler configure when configuration WIA-PA network. After that, VCR tables and IPv6 address-mapping tables will be formed according to this information. When the gateway receives WIA-PA packets, it will firstly parse out Route ID, Object ID and Instance ID, and find corresponding VCR from VCR tables. Meanwhile, the gateway finds the corresponding IPv6 address according to Route ID in IPv6 address mapping table. Then, the gateway begins to encapsulate WIA-PA packets based on IPv6 format, fill VCR_ID in IPv6 header flow label field, and fill the priority of WIA-PA packet in communication category of IPv6 header

fields, zero is used to fill up insufficient bytes. Then, the protocol conversion for WIA-PA data is completed.

When the gateway receives IPv6 packets from the industrial backhaul networks, the gateway will make out VCR_ID from IPv6 packet header, and find packets VCR in the domain WIA-PA network according to the VCR ID in its own maintenance VCR table, and replace it with the information of original packet. Then, the protocol conversion for IPv6 packet is completed.

3.3. Industrial Backhaul Network Scheduling

In deterministic network based on SDN, joint scheduler can recognize WIA-PA data stream through matching on IPv6 flow label field. According to priority of IPv6 and VCR_ID type, joint scheduling can allocate the necessary resources to communication, and ensure that the key data flow is not affected when adding new data flow in the existing network. It can also monitor the real-time data flow of the network. To protect critical data flows from affected, switching paths is also considered when necessary. The scheduling process of industrial backhaul network is shown in Figure 5.

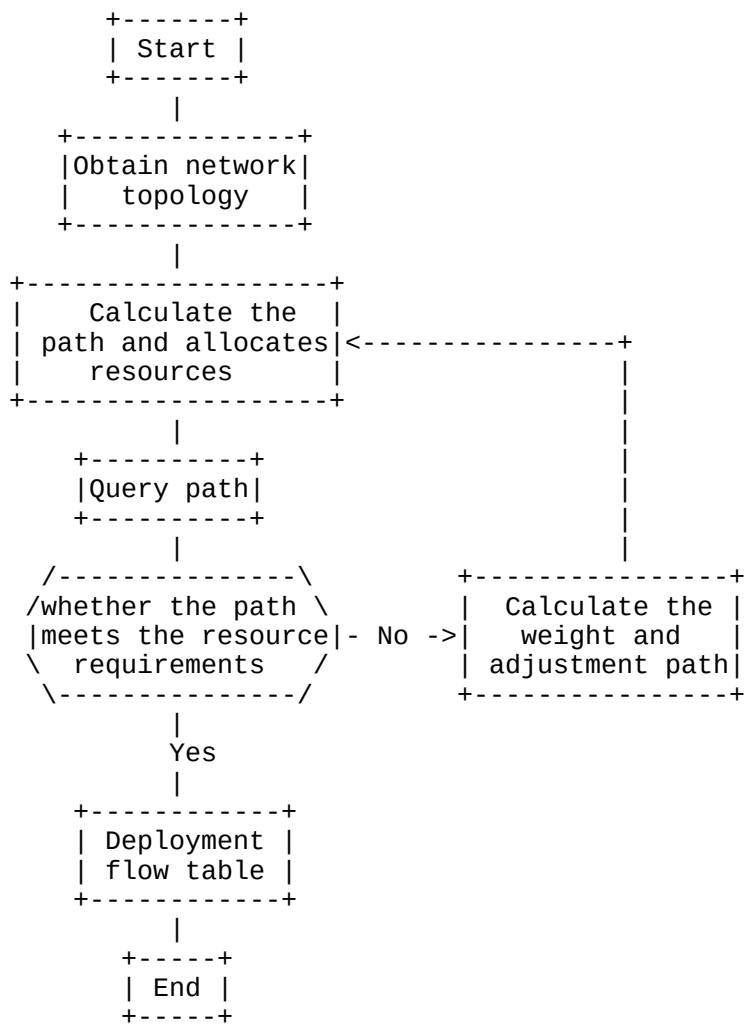


Figure 5. The scheduling process of Industrial backhaul network

After receiving the request for service, the joint scheduler will calculate the route information and network resource allocation. Once the path information and resource allocation are determined, joint dispatcher will confirm whether the resource path is capable of meeting business requirements through the inside module of SDN controller. If it meets business requirements, then the flow table is deployed by SDN controller. Otherwise, the path information and resource allocation are recalculated to choose the other paths to transmit data flow.

4. Security Considerations

5. IANA Considerations

This memo includes no request to IANA.

6. References

6.1. Normative References

6.2. Informative References

[IEC62734]
ISA/IEC, "ISA100.11a, Wireless Systems for Automation, also IEC 62734", 2011, <http://www.isa100wci.org/enUS/Documents/PDF/3405-ISA100-WirelessSystems-Future-brochWEB-ETSI.aspx>.

[IEC62591]
IEC, "Industrial Communication Networks - Wireless Communication Network and Communication Profiles - WirelessHART - IEC 62591", 2010, <https://webstore.iec.ch/p-preview/info_iec62591%7Bed1.0%7Den.pdf>

[IEC62601]
IEC, "Industrial networks - Wireless communication network and communication profiles - WIA-PA - IEC 62601", 2015, <https://webstore.iec.ch/preview/info_iec62601%7Bed2.0%7Db.pdf>

[I-D.finn-detnet-problem-statement]
Finn, N. and P. Thubert, "Deterministic Networking Problem Statement", draft-finn-detnet-problem-statement-04 (work in progress), October 2015.

[I-D.finn-detnet-architecture]
Finn, N., Thubert, P., and M. Teener, "Deterministic Networking Architecture", draft-finn-detnetarchitecture-03 (work in progress), March 2016.

[I-D.bas-usecase-detnet]
Kaneko, Y., Toshiba and Das, S, "Building Automation Use Cases and Requirements for Deterministic Networking", draft-bas-usecase-detnet-00 (work in progress), April 2016.

Authors' Addresses

Heng Wang
Chongqing University of Posts and Telecommunications
2 Chongwen Road
Chongqing, 400065
China

Phone: (86)-23-6248-7845
Email: wangheng@cqupt.edu.cn

Ping Wang
Chongqing University of Posts and Telecommunications
2 Chongwen Road
Chongqing, 400065
China

Phone: (86)-23-6246-1061
Email: wangping@cqupt.edu.cn

Chang Zhang
Chongqing University of Posts and Telecommunications
2 Chongwen Road
Chongqing, 400065
China

Phone: (86)-23-6246-1061
Email: zc910522@126.com

Yi Yang
Chongqing University of Posts and Telecommunications
2 Chongwen Road
Chongqing, 400065
China

Phone: (86)-23-6246-1061
Email: 15023705316@163.com

DetNet
Internet Draft
Intended status: Standards Track
Expires: June 11, 2018

H. Wang
P. Wang
C. Zhang
Y. Yang
Chongqing University of
Posts and Telecommunications

December 8, 2017

Joint Scheduling Architecture for Deterministic Industrial Field/Backhaul Networks
draft-wang-detnet-backhaul-architecture-02

Abstract

The joint scheduling between industrial field network and backhaul network is important to satisfy the requirements of deterministic delay for data flows in factories. This document describes a joint scheduling architecture for deterministic industrial field/backhaul networks. Taking WIA-PA, an international standard about industrial wireless field network, and IPv6-based backhaul network as an example, this document depicts how the joint scheduling architecture works in detail.

Table of Contents

1. Introduction 2
2. Joint Scheduling Architecture 3
2.1. Distributed Architecture 4
2.2. Centralized Architecture 5
2.3. Joint Scheduling Architecture 6
3. Joint Scheduling Scheme 9
3.1. WIA-PA Network Joint Scheduling 10
3.2. Protocol Conversion 10
3.3. Industrial Backhaul Network Scheduling 12
3.4. Bandwidth guarantee method 14
4. Security Considerations 14
5. IANA Considerations 14
6. References 14
6.1. Normative References 14
6.2. Informative References 14
Authors' Addresses 16

1. Introduction

Deterministic network is one of essential elements of industrial network. With the help of deterministic network, industrial field network can greatly enhance the network performance in terms of reducing transmission delay. Thus, applying deterministic network into the whole industrial network has attracted a lot of attention recently. Deterministic network is mainly focused on the industrial field networks, such as ISA100.11a [IEC62734], WirelessHART [IEC62591] and WIA-PA [IEC62601]. In order to solve the problem of data transmission in different industrial field networks and the issue of data flows between industrial field networks and wide area

networks, industrial backhaul network is deployed in factory. However, there are little considerations about joint scheduling scheme that can be applied to industrial networks.

The emerging Software Defined Networks (SDN) technology on the Internet brings a new choice to solve joint scheduling problem. SDN has been proposed as a new network architecture in recent years. The network architecture separates the network control plane from the forwarding plane, which brings a revolution for the network architecture. By separating control plane from forwarding plane ,and the open communication protocol, SDN breaks the closure of traditional network device provider. Besides, open interfaces and free programmability also make network management more efficient and flexible.

In document [I-D.bas-usecase-detnet] and [I-D.finn-detnet-architecture] submitted by the IETF DetNet working group, deterministic network based on Ethernet has been researched already. They propose a network architecture based on SDN technology that can accurately control the transmission of data streams. However, the characteristics of the industrial backhaul network and the actual condition of industrial field deterministic networks are not considered. Firstly, the data that transmits in industrial backhaul network is highly sensitive to transmission delay. Secondly, the existing deterministic networks have been widely deployed in industrial field environment, thus the direct replacement for original networks will consume many workers and material resources.

Based on existing research in document [I-D.finn-detnet-architecture], this document proposes a joint scheduling architecture for deterministic industrial networks. It will firstly replace the industrial backhaul networks and other non-deterministic networks located in industrial networks with deterministic Ethernet network. Then this document proposes a joint scheduler based on SDN technology. By deploying the deterministic network in complete industrial network, it can realize the end-to-end deterministic scheduling between different industrial field networks.

2. Joint Scheduling Architecture

There are many types of network controllers in industrial networks, which constitute the control plane of the whole industrial network together. The control plane is very important in the entire network, especially when it refers to cross-domain transmission of time-sensitive data. The control plane architecture affects the performance of the network greatly. It is becoming a hot research on

how to give full play to the performance of their respective networks when multiple controllers are in the joint work.

However, there is no unified standard for the joint architecture of multiple controllers in the industry presently. The mainstream of architecture includes distributed architecture and centralized architecture.

2.1. Distributed Architecture

Distributed architecture is known as East-West architecture. In the architecture, the status of all network controllers are equal, these controllers connect with each other to form an unstructured network, and implement cross-domain transmission task by exchanging information, as shown in Figure 1.

In distributed architecture, controller can exchange different network topologies and the accessibility of information by east-west interface, and each controller can establish a global network topology. From a global network perspective, each controller is equal, thus it can serve as a server role as well as the ability to start deterministic cross-network transmission task.

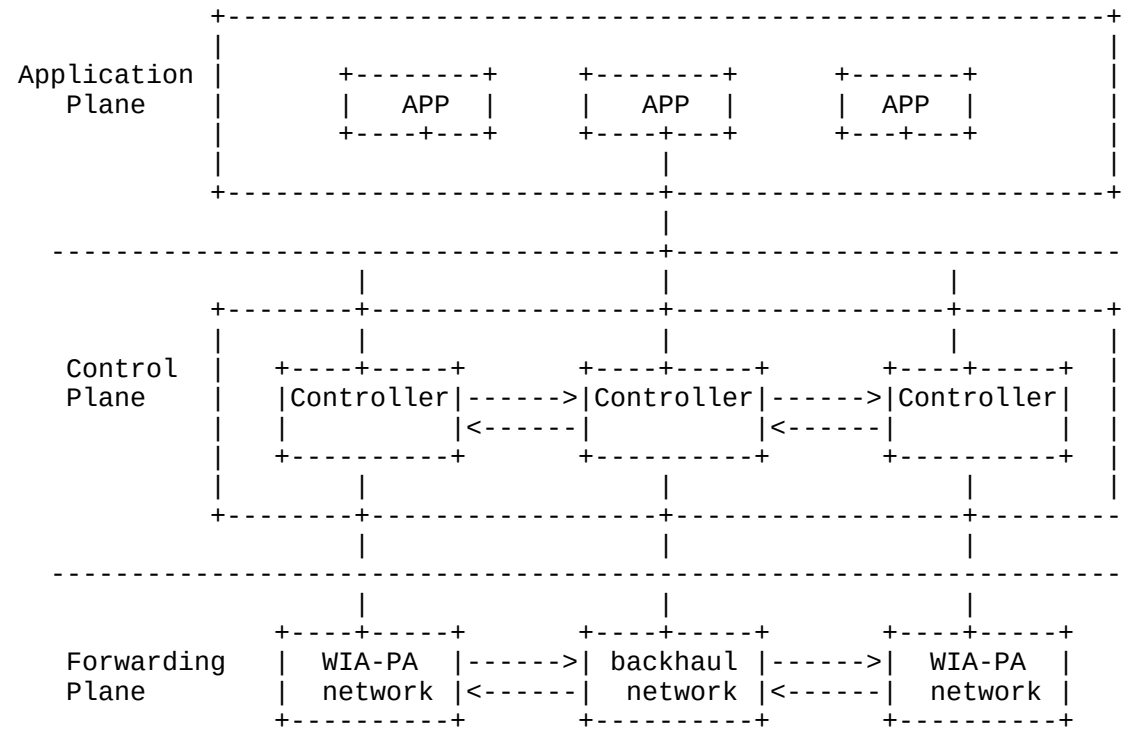


Figure 1. Distributed Architecture

2.2. Centralized Architecture

Centralized architecture is also called a vertical multi-level architecture. In this architecture, the control plane is divided into two parts, one part is the basic control plane, which is composed of a variety of network controllers, and another is a main network controller, which is responsible for controlling the basic control plane. The detailed description of centralized architecture is shown in Figure 2.

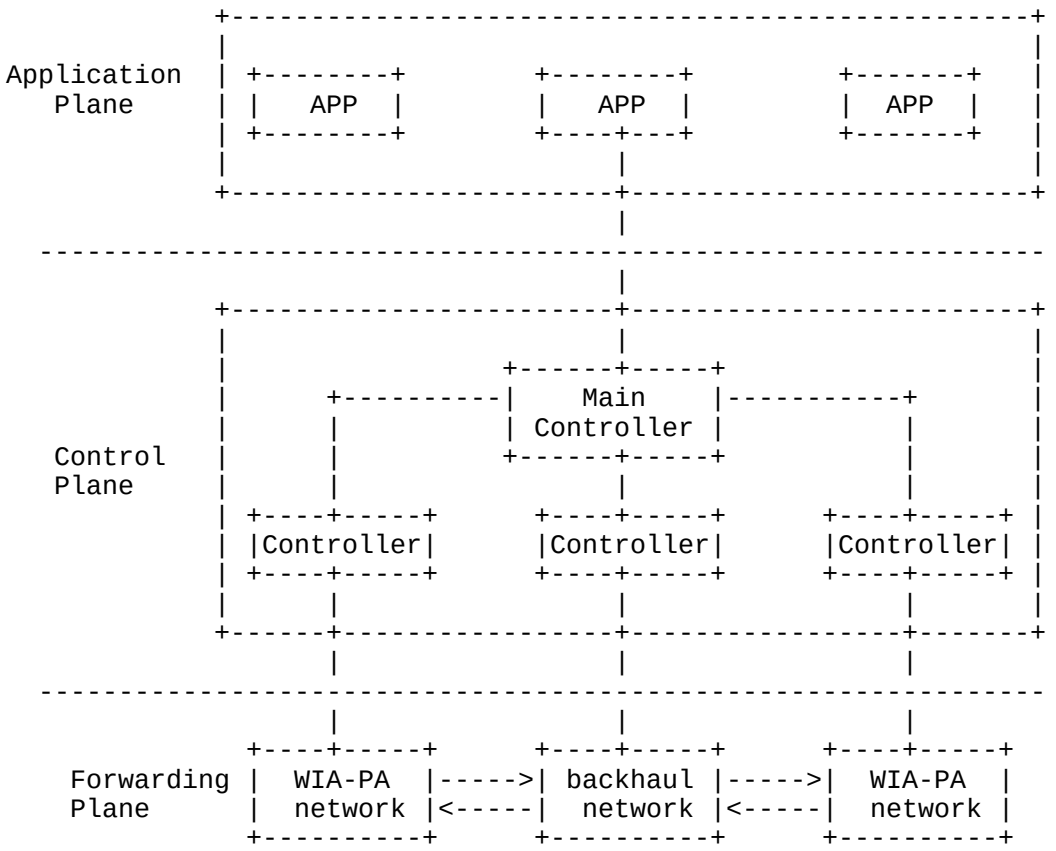


Figure 2. Centralized Architecture

The centralized architecture does not need to expand east-west interface. It only needs to establish a connection with the basic controllers using southbound interface. After the connection is established, the main controller obtains the every topology of network domain by the API interface provided by the basic controllers, and storages global network topology. Main controller can also assign tasks to basic controllers by API interface.

2.3. Joint Scheduling Architecture

In practical application, distributed architecture not only needs to extend the east-west interface, but also maintains a global network

topology in each controller. Only each controller maintains such a global network topology, can it ensure the deterministic control of whole network.

For deterministic industrial network, the scale of network is not very large. Besides, in industrial backhaul network, a single SDN controller is sufficient to meet the demands of control. If centralized architecture is directly applied to an industrial network, it will not only be unable to make full use of advantages of the multi-controller architecture, but also cause unnecessary information interaction between controllers wasting network resource.

Considering the problems existing in above two architectures, this document proposes a joint scheduling architecture based on the architecture document [I-D.finn-detnet-architecture]. The architecture is optimized according to the characteristics of deterministic industrial network. A single SDN controller can unite the WIA-PA network system manager to manage the entire industrial network, and provide support for the deterministic scheduling of cross-network data transmission through industrial backhaul network located in different WIA-PA networks.

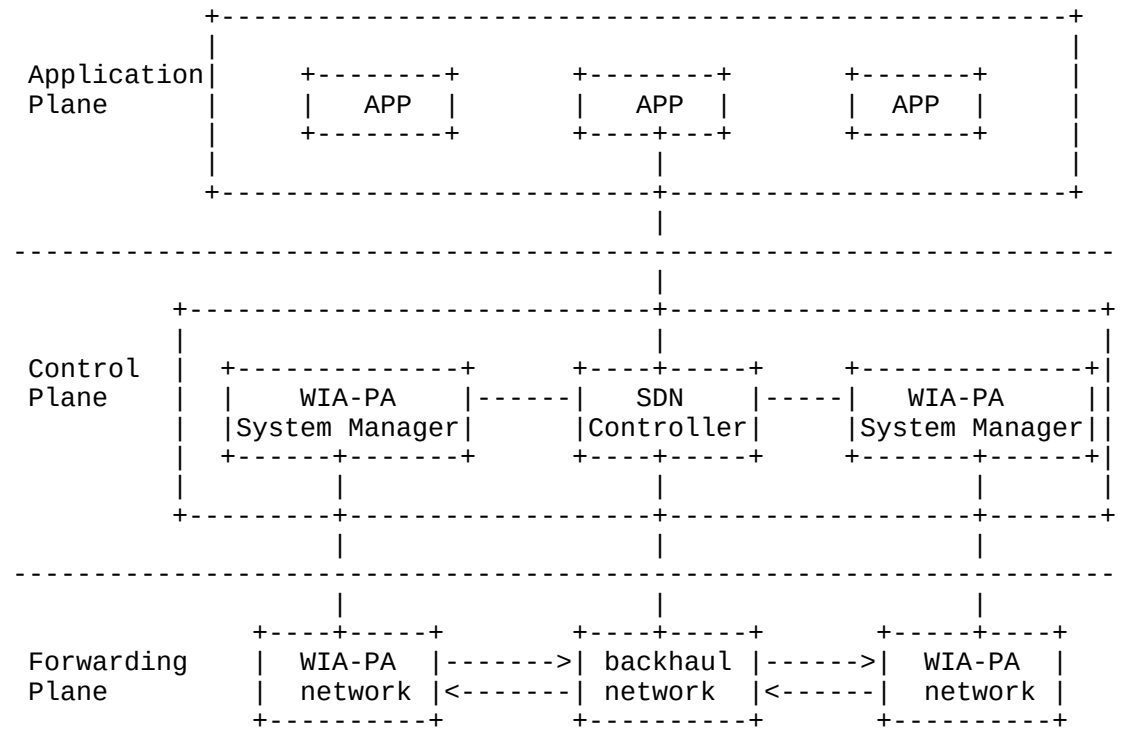


Figure 3. Joint scheduling architecture

Figure 3 depicts the joint scheduling architecture. The architecture can be mainly classified into three planes:

- Forwarding plane: this plane contains various types of network equipment belonged to different networks. It is the physical entities for network transmission. To obtain the desired network functions from network manager, devices should abstract their own resources to provide to its network manager respectively.

- Control plane: this plane is composed by WIA-PA system manager and SDN controller. Joint scheduler is integrated into the SDN controller by a way of plugin, and other WIA-PA system managers communicate with joint scheduler by establishing a connection with the SDN controller. Meanwhile, joint scheduler implement the management of industrial backhaul network by directly invoking the corresponding module of SDN controller.

- Application plane: this plane provides users with a unified interface about many variety of resources for the whole network. At the same time, it also provides an intuitive and user-friendly interface, which shields the complex network information of the original.

When application plane triggers a joint scheduling task, SDN controller calculates path and allocates resource according to the task request from the application plane. Upon finishing calculation, SDN controller sends them using the unified joint scheduling interface to corresponding network manager, and then the network manager sends them to the industrial field network.

Based on joint scheduling architecture, joint scheduler can realize control and scheduling for the entire industrial network, thus it can provide a real-time guarantee for each data stream.

3. Joint Scheduling Scheme

Taking WIA-PA network and IPv6-based backhaul network as an example, this section describes how the joint scheduling architecture works. Existing WIA-PA scheduling scheme only applies to WIA-PA field network. Scheduling scheme will fail once data transfers to backhaul networks. Joint scheduling scheme is an innovation and expansion compared to WIA-PA scheduling scheme.

Firstly, original scheduling scheme based on SDN in industry backhaul network is added to the proposed scheduling scheme, thus, data can flow in the industrial backhaul network.

Secondly, by conducting an optimization for original WIA-PA scheduling scheme, original scheduling scheme can work with joint scheduler, and simultaneously be applied to cross-domain network.

Thirdly, due to the specificity of cross-border transmission services, the joint scheduling scheme for WIA-PA network VCR_ID and route ID is reclassified.

Finally, due to system manager allocates short address to field device based on WIA-PA network address information independently. Thus the short address of field device in entire industrial network is uncertain. In order to identify the field device belonged to different network domains, the network identifier (PAN_ID) is applied to the joint scheduling scheme to identify different WIA-PA networks.

After the SDN controller initiates joint scheduling module, WIA-PA system manager will actively establish a connection with the united scheduler. After the scheduler receives a cross-border transmission request, joint scheduler will send a request for obtaining topology information and node information to WIA-PA System Manager. Then, the scheduler will assign paths and network resources according to this information by pre-defined scheduling algorithm. After the path and network resources have been calculated, joint scheduler will configure and deploy networks by the corresponding network controller.

3.1. WIA-PA Network Joint Scheduling

In joint scheduling process, path deployment and resource allocation for WIA-PA network are performed by employing the WIA-PA system manager API interface. System manager will query the corresponding information of the field device in the network upon receiving the command about joint operation for the network information, and then return the received information to the joint scheduler. The system manager will configure communication resources for the corresponding gateway device, routing equipment and field equipment when it receives configuration commands from joint scheduler.

3.2. Protocol Conversion

For cross-domain transmission, industrial backhaul network is different from WIA-PA network which is not an IP-based Ethernet. Protocol conversion for WIA-PA packet in gateway is needed when data generated from WIA-PA network needs to transmit to another field network through industrial backhaul. Meanwhile, according to the joint scheduling scheme, SDN controller is able to recognize the WIA-PA data stream and allocate resources according to data stream type. Therefore, in the protocol conversion process, scheduling and control of WIA-PA data flow can be realized by SDN controller by combining the VCR of WIA-PA data stream and the priority filled in IPv6 header.

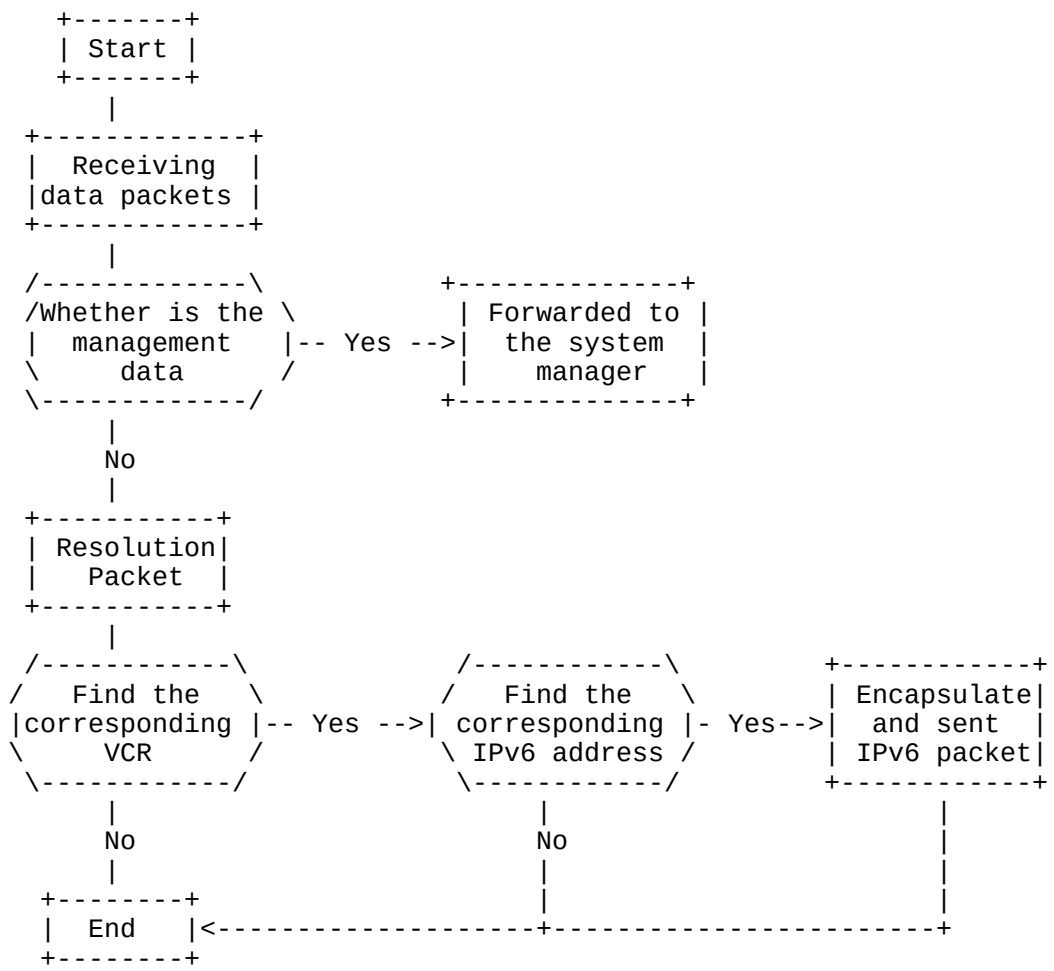


Figure 4. The process of protocol conversion in gateway

As shown in Figure 4, gateway will receive the address mapping configuration command from joint scheduler. Then VCR tables and IPv6 address-mapping tables will be formed according to this information. When gateway receives WIA-PA packets, it will firstly parse Route ID, Object ID and Instance ID, and find corresponding VCR from VCR tables. Meanwhile, the gateway finds the corresponding IPv6 address according to Route ID in IPv6 address mapping table. Then, gateway begins to encapsulate WIA-PA packets based on IPv6 format, fill VCR_ID in IPv6 header flow label field and the priority of WIA-PA packet in IPv6 header fields.

When receiving IPv6 packets from industrial backhaul networks, gateway will recognize VCR_ID from IPv6 packet header, and obtain packet VCR according to the VCR ID in VCR table, then replace it with the information of original packet.

3.3. Industrial Backhaul Network Scheduling

In deterministic network based on SDN, joint scheduler can recognize WIA-PA data stream by matching IPv6 flow label field. According to priority in IPv6 header field and VCR_ID type, joint scheduling can allocate the necessary resources to communication and ensure that important data flow is not affected when adding new data flows in existing network. It can also monitor the real-time data flow. To guarantee the real-time performance of critical data flows, redundant paths are also considered when necessary. The scheduling process of industrial backhaul network is shown in Figure 5.

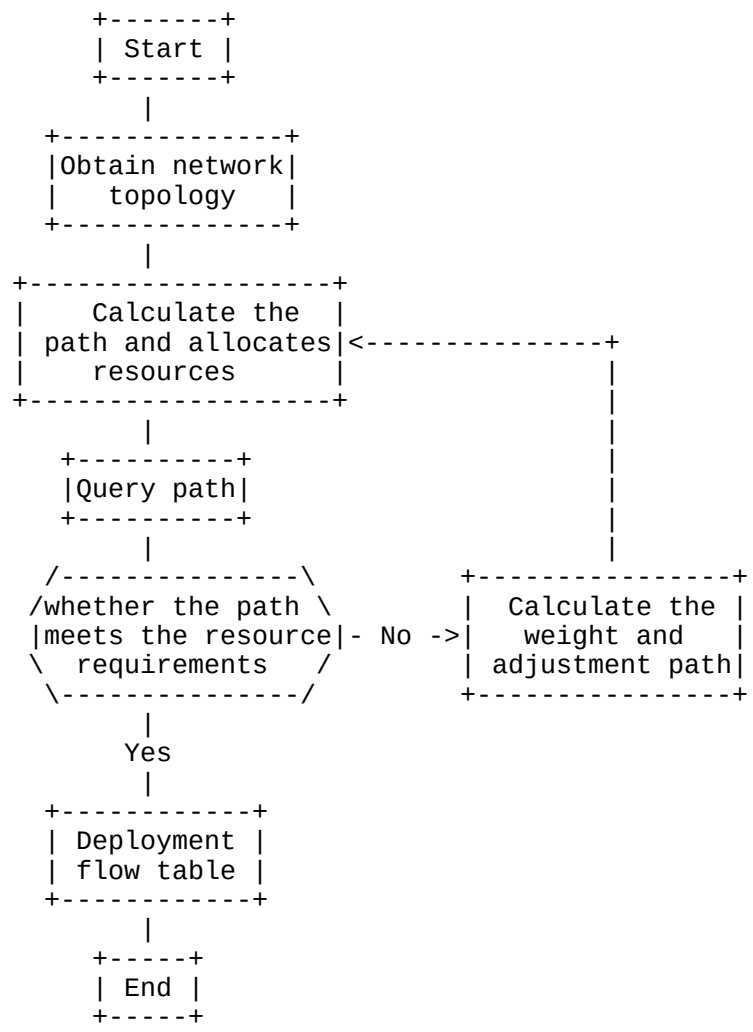


Figure 5. The scheduling process of industrial backhaul network

When receiving the request for service, the joint scheduler will calculate route information and resource allocation. Once the path and resource allocation are determined, joint scheduling will confirm whether the resource and path are capable of meeting business requirements by the inside module of SDN controller. If it meets, then the flow table is deployed by SDN controller. Otherwise, the path information and resource allocation are recalculated to choose the correct paths to transmit data flow.

3.4. Bandwidth guarantee method

Bandwidth guarantee method is implemented on the basis of joint scheduling mechanism, in order to solve the problem that industrial backhaul networks can not identify fine-grained and cross-network data transmission. By filling the priority information and the RouteID defined in WIA-PA network into the IPv6 header, the SDN controller can not only identify cross-network transmission of the WIA-PA data stream, but also obtain priority information about the WIA-PA data stream. In industrial backhaul network, the SDN switch employs the mechanism of priority queue to allocate network bandwidth. Thus SND switch can distribute WIA-PA data streams into corresponding queues of ports according to the received flow table from SDN controller. Therefore, the bandwidth of data stream is guaranteed.

By using the above methods, joint scheduling mechanism can distinguish data streams in a fine-grained way and guarantee bandwidth when data transmits in industrial backhaul network. For example, real-time data in WIA-PA network is sensitive to delay, thus it should be allocated more bandwidth to reduce transmission delay. For not urgent data, it can be assigned less bandwidth to reserve bandwidth for real-time data. Meanwhile, SDN controller can flexibly adjust bandwidth allocation strategy to relieve network congestion.

4. Security Considerations

5. IANA Considerations

This memo includes no request to IANA.

6. References

6.1. Normative References

6.2. Informative References

[IEC62734] ISA/IEC, "ISA100.11a, Wireless Systems for Automation, also IEC 62734", 2011, <http://www.isa100wci.org/enUS/Documents/PDF/3405-ISA100-WirelessSystems-Future-brochWEB-ETSI.aspx>.

[IEC62591]
IEC, "Industrial Communication Networks - Wireless Communication Network and Communication Profiles - WirelessHART - IEC 62591", 2010, <https://webstore.iec.ch/p-preview/info_iec62591%7Bed1.0%7Den.pdf>

[IEC62601]
IEC, "Industrial networks - Wireless communication network and communication profiles - WIA-PA - IEC 62601", 2015, < https://webstore.iec.ch/preview/info_iec62601%7Bed2.0%7Db .pdf>

[I-D.finn-detnet-problem-statement]
Finn, N. and P. Thubert, "Deterministic Networking Problem Statement", draft-finn-detnet-problem-statement-05 (work in progress), March 2016.

[I-D.finn-detnet-architecture]
Finn, N., Thubert, P., and M. Teener, "Deterministic Networking Architecture", draft-finn-detnet-architecture-08 (work in progress), August 2016.

[I-D.bas-usecase-detnet]
Kaneko, Y., Toshiba and Das, S, "Building Automation Use Cases and Requirements for Deterministic Networking", draft-bas-usecase-detnet-00 (work in progress), October 2015.

Authors' Addresses

Heng Wang
Chongqing University of Posts and Telecommunications
2 Chongwen Road
Chongqing, 400065
China

Phone: (86)-23-6248-7845
Email: wangheng@cqupt.edu.cn

Ping Wang
Chongqing University of Posts and Telecommunications
2 Chongwen Road
Chongqing, 400065
China

Phone: (86)-23-6246-1061
Email: wangping@cqupt.edu.cn

Chang Zhang
Chongqing University of Posts and Telecommunications
2 Chongwen Road
Chongqing, 400065
China

Phone: (86)-23-6246-1061
Email: zc910522@126.com

Yi Yang
Chongqing University of Posts and Telecommunications
2 Chongwen Road
Chongqing, 400065
China

Phone: (86)-23-6246-1061
Email: 15023705316@163.com