6TiSCH
Internet-Draft
Intended status: Informational
Expires: April 21, 2016

D. Dujovne, Ed.
Universidad Diego Portales
LA. Grieco
Politecnico di Bari
MR. Palattella
University of Luxembourg
N. Accettura
University of California Berkeley
October 19, 2015

6TiSCH 6top Scheduling Function Zero (SF0)
draft-dujovne-6tisch-6top-sf0-00

Abstract

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

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft will expire on April 21, 2016.

Table of Contents

1. Introduction

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

This document addresses the requirements for a scheduling function listed in [I-D.wang-6tisch-6top-sublayer], Section 4.2, and follows the recommended outline from Section 4.3.

2. Scheduling Function Identifier

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

3. Rules for Adding/Deleting Cells

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

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

3.1. SF0 Triggering Events

We RECOMMEND SF0 to monitor the bandwidth usage on the node (local node bandwidth) and bandwidth requests from neighbour nodes (incoming bandwidth). This allows SF0 to be triggered by any change in local node bandwidth and/or incoming bandwidth. The exact mechanism of when SF0 is triggered is implementation-specific.

3.2. SF0 Bandwidth Estimation Algorithm

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

1. Collect the Incoming Bandwidth Requirements from neighbour nodes (IBR).
2. Collect the Local node Bandwidth Requirements (LBR).
3. Calculate the updated total Outgoing Bandwidth Requirement (OBR) as: OBR=LBR+IBR and submit the request to the allocation policy.
4. Return to step 1.

3.3. SF0 Allocation Policy

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

SF0 uses the following parameters:

SCHEDULEDCELLS: The number of cells scheduled from the current cell to a particular neighbor.
REQUIREDCELLS: The number of cells calculated by the Bandwidth Estimation Algorithm from the current node to that neighbor.
SF0THRESH: Threshold parameter introducing cell over-provisioning in the allocation policy. It is a non-negative value expressed as number of cells. The definition of this value is implementation-specific; however, it is RECOMMENDED a SF0THRESH value of 3 cells. A setting of SF0THRESH>0 will cause the node to allocate at least SF0THRESH cells to each of its' neighbours.

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

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

Figure 1: The SF0 Allocation Policy

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

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

4. Rules for CellList

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

> The Transaction Source node, for each of the cells to be put in the CellList field, first selects the slotOffset randomly; second, it verifies if the slotOffset is free and third it chooses the channelOffset randomly.
> The Transaction Destination node goes through the cells in the CellList in order. For each one, it verifies whether it has a cell schedule with the same slotOffset. If yes, it skips the cell. If not, it allocates the cell. This stops when either (1) it has scheduled NumCells cells or (2) there are no more cells in the CellList.

5. 6P Timeout Value

The 6P Timeout Value provided by SF0 allows the maximum number of TSCH link-layer retries. Given the TSCH parameters for the backoff mechanism, macMinBE and macMaxBE, and the length in seconds of the minimal Slotframe, SM, the timeout value is computed as: timeout = (2^(macMaxBE+1)-2^macMinBE) * SM

6. Meaning of Container Field

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

7. Node Behavior at Boot

In order to define a known state after the node is restarted, a CLEAR command is issued to each of the neighbour nodes to enable a new allocation process.

8. Relocating Cells

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

9. 6P Error Handling

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

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

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

RC_ERR_6OFID: The node MUST NOT retry immediately. The node MAY add the neighbor node on a blacklist. The node MAY retry to contact this neighbor later.

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

RC_ERR_BUSY: Issue a RESET command.

10. Examples

TODO

11. 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.

12. Security Considerations

TODO

13. IANA Considerations

- o IANA_SFID_SF0

14. 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).

15. References

15.1. 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>.

[IEEE802154e]
IEEE standard for Information Technology, "IEEE std. 802.15.4e, Part. 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendament 1: MAC sublayer", April 2012.

[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", June 2011.

15.2. Informative References

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

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

[I-D.ietf-6tisch-terminology]
Palattella, M., Thubert, P., Watteyne, T., and Q. Wang, "Terminology in IPv6 over the TSCH mode of IEEE 802.15.4e", draft-ietf-6tisch-terminology-05 (work in progress), July 2015.

[I-D.wang-6tisch-6top-sublayer]
Wang, Q. and X. Vilajosana, "6TiSCH Operation Sublayer (6top)", draft-wang-6tisch-6top-sublayer-02 (work in progress), October 2015.

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

Authors' Addresses

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

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

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

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

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

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

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

Email: nicola.accettura@eecs.berkeley.edu

6TiSCH
Internet-Draft
Intended status: Informational
Expires: December 13, 2015

P. Thubert, Ed.
Cisco

June 11, 2015

6TiSCH requirements for DetNet
draft-thubert-6tisch-4detnet-01

Abstract

This document builds on the 6TiSCH architecture that defines, among others, mechanisms to establish and maintain deterministic routing and scheduling in a centralized fashion. The document details dependencies on DetNet and PCE controller to express topologies and capabilities, as well as abstract state that the controller must be able to program into the network devices to enable deterministic forwarding operations.

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

Copyright Notice

Table of Contents

1. Introduction

The emergence of wireless technology has enabled a variety of new devices to get interconnected, at a very low marginal cost per device, at any distance ranging from Near Field to interplanetary, and in circumstances where wiring may not be practical, for instance on fast-moving or rotating devices.

At the same time, a new breed of Time Sensitive Networks is being developed to 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. Such traffic is not limited to professional Audio/ Video networks, but is also found in command and control operations such as industrial automation and vehicular sensors and actuators.

At IEEE802.1, the Audio/Video Task Group [IEEE802.1TSNTG] Time Sensitive Networking (TSN) to address Deterministic Ethernet. The Medium access Control (MAC) of IEEE802.15.4 [IEEE802154] has evolved with the new TimeSlotted Channel Hopping (TSCH) [I-D.ietf-6tisch-tsch] mode for deterministic industrial-type applications. TSCH was introduced with the IEEE802.15.4e [IEEE802154e] amendment and will be wrapped up in the next revision of the IEEE802.15.4 standard. For all practical purpose, this document is expected to be insensitive to the future versions of the IEEE802.15.4 standard, which is thus referenced undated.

Though at a different time scale, both TSN and TSCH standards provide 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, as a train that leaves intermediate stations at precise times along its path. With TSCH, time is formatted into timeSlots, and an individual cell is 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 multi-path fading and co-channel interferences (for example by Wi-Fi emitters).

The 6TiSCH Architecture [I-D.ietf-6tisch-architecture] defines a remote monitoring and scheduling management of a TSCH network by a Path Computation Element (PCE), which cooperates with an abstract Network Management Entity (NME) to manage timeSlots and device resources in a manner that minimizes the interaction with and the load placed on the constrained devices.

This Architecture applies the concepts of Deterministic Networking on a TSCH network to enable the switching of timeSlots in a G-MPLS manner. This document details the dependencies that 6TiSCH has on PCE [PCE] and DetNet [I-D.finn-detnet-architecture] to provide the necessary capabilities that may be specific to such networks. In turn, DetNet is expected to integrate and maintain consistency with the work that has taken place and is continuing at IEEE802.1TSN and AVnu.

2. Terminology

Readers are expected to be familiar with all the terms and concepts that are discussed in "Multi-link Subnet Support in IPv6" [I-D.ietf-ipv6-multilink-subnets].

The draft uses terminology defined or referenced in [I-D.ietf-6tisch-terminology] and [I-D.ietf-roll-rpl-industrial-applicability].

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

3. 6TiSCH Overview

The scope of the present work is a subnet that, in its basic configuration, is made of a TSCH [I-D.ietf-6tisch-tsch] MAC Low Power Lossy Network (LLN).

```
            ---+-------- ............ ------------
               |      External Network       |
               |                          +-----+
            +-----+                       | NME |
            |     | LLN Border            |     |
            |     | router                +-----+
            +-----+
          o    o   o
   o     o   o     o
      o   o LLN   o    o     o
         o   o   o       o
                 o
```

Figure 1: Basic Configuration of a 6TiSCH Network

In the extended configuration, a Backbone Router (6BBR) federates multiple 6TiSCH in a single subnet over a 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.

```
                 ---+-------- ............ ------------
                    |      External Network       |
                    |                           +-----+
                    |             +-----+       | NME |
                 +-----+          |  +-----+    |     |
                 |     | Router   |  | PCE |    +-----+
                 |     |          +--|     |
                 +-----+             +-----+
                    |                   |
                    | Subnet Backbone   |
              +--------------------+------------------+
              |                    |                  |
           +-----+             +-----+             +-----+
           |     | Backbone    |     | Backbone    |     | Backbone
      o    |     | router      |     | router      |     | router
           +-----+             +-----+             +-----+
      o                  o                   o                 o   o
          o    o   o         o   o  o   o         o  o   o    o
     o            o        o  LLN      o      o         o      o
        o   o    o      o      o o     o  o   o    o    o     o
```

Figure 2: Extended Configuration of a 6TiSCH Network

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. This SHOULD be done in conformance to the DetNet Architecture [I-D.finn-detnet-architecture] which studies Layer-3 aspects of Deterministic Networks, and covers networks that span multiple Layer-2 domains. One particular requirement is that the PCE MUST be able to compute a deterministic path and to end across the TSCH network and an IEEE802.1 TSN Ethernet backbone, and DetNet MUST enable end-to-end deterministic forwarding.

6TiSCH defines the concept of a Track, which is a complex form of a uni-directional Circuit ([I-D.ietf-6tisch-terminology]). As opposed to a simple circuit that is a sequence of nodes and links, a Track is shaped as a directed acyclic graph towards a destination to support multi-path forwarding and route around failures. A Track may also branch off and rejoin, for the purpose of the so-called Packet Replication and Elimination (PRE), over non congruent branches. PRE may be used to complement layer-2 Automatic Repeat reQuest (ARQ) to meet industrial expectations in Packet Delivery Ratio (PDR), in particular when the Track extends beyond the 6TiSCH network.

```
                    +-----+
                    | IoT |
                    | G/W |
                    +-----+
                       ^  <---- Elimination
                      | |
       Track branch   | |
              +-------+ +--------+ Subnet Backbone
              |                  |
           +--|--+            +--|--+
           |  |  | Backbone   |  |  | Backbone
      o    |  |  | router     |  |  | router
           +--/--+            +--|--+
      o     /    o     o---o----/       o
          o    o---o--/   o      o   o  o   o
     o     \  /     o              o   LLN    o
        o   v  <---- Replication
            o
```

Figure 3: End-to-End deterministic Track

In the example above, a Track is laid out from a field device in a 6TiSCH network to an IoT gateway that is located on a IEEE802.1 TSN backbone.

The Replication function in the field device 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.

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 current deployments, a TSCH Track does not necessarily support PRE but is 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.

3.1. TSCH and 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.wang-6tisch-6top-sublayer].

The 6top data model and management interfaces are further discussed in [I-D.ietf-6tisch-6top-interface] and [I-D.ietf-6tisch-coap].

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.

3.2. SlotFrames and Priorities

A slotFrame is the base object that the PCE needs to manipulate to program a schedule into an LLN node. Elaboration on that concept can be fond in section "SlotFrames and Priorities" of [I-D.ietf-6tisch-architecture]

IEEE802.15.4 TSCH avoids contention on the medium by formatting time and frequencies in cells of transmission of equal duration. 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.

The frequency used by a cell in the matrix rotates in a pseudo-random fashion, from an initial position at an epoch time, as the matrix iterates over and over.

A CDU matrix is computed by the PCE, but unallocated timeSlots may be used opportunistically by the nodes for classical best effort IP traffic. The PCE has precedence in the allocation in case of a conflict.

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 PCE MUST compute the CDU matrices and shared that knowledge with all the nodes. The matrices are 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.

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 PCE MUST compute the partitioning of CDU matrices into chunks and shared that knowledge with all the nodes in a 6TiSCH network.

```
             +-----+-----+-----+-----+-----+-----+-----+     +-----+
chan.Off. 0  |chnkA|chnkP|chnk7|chnkO|chnk2|chnkK|chnk1| ... |chnkZ|
             +-----+-----+-----+-----+-----+-----+-----+     +-----+
chan.Off. 1  |chnkB|chnkQ|chnkA|chnkP|chnk3|chnkL|chnk2| ... |chnk1|
             +-----+-----+-----+-----+-----+-----+-----+     +-----+
               ...
             +-----+-----+-----+-----+-----+-----+-----+     +-----+
chan.Off. 15 |chnkO|chnk6|chnkN|chnk1|chnkJ|chnkZ|chnkI| ... |chnkG|
             +-----+-----+-----+-----+-----+-----+-----+     +-----+
                0     1     2     3     4     5     6          M
```

Figure 4: CDU matrix Partitioning in Chunks

The appropriation of a chunk can be requested explicitly by the PCE to any node. After a successful appropriation, the PCE owns the cells in that chunk, and may use them as hard cells to set up Tracks.

3.3. Schedule Management by a PCE

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.

Section "Schedule Management Mechanisms" of the 6TiSCH architecture describes 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. The Track operation for DetNet corresponds to a remote monitoring and scheduling management by a PCE.

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.

[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.

[I-D.ietf-6tisch-coap] also defines a basic set CoAP resources and associated RESTful access methods (GET/PUT/POST/DELETE). The payload (body) of the CoAP messages is encoded using the CBOR format. The PCE commands are expected to be issued directly as CoAP requests or to be mapped back and forth into CoAP by a gateway function at the edge of the 6TiSCH network. For instance, it is 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. This architecture will be refined to comply with DetNet [I-D.finn-detnet-architecture] when the work is formalized.

3.4. Track Forwarding

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 the routing computation of a Track by a PCE. The 6TiSCH architecture supports three different forwarding model, G-MPLS Track Forwarding (TF), 6LoWPAN Fragment Forwarding (FF) and IPv6 Forwarding (6F) which is the classical IP operation. The DetNet case relates to the Track Forwarding operation under the control of a PCE.

A Track is a unidirectional path between a source and a destination. In a Track cell, the normal operation of IEEE802.15.4 Automatic Repeat-reQuest (ARQ) usually happens, though the acknowledgment may be omitted in some cases, for instance if there is no scheduled cell for a retry.

Track Forwarding is the simplest and fastest. A bundle of cells set 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 that can be used regardless of the network layer protocol. This model can effectively 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 IEEE802.15.4, this means effectively broadcast, so that along the Track the short address for the destination of the frame is set to 0xFFFF.

A 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, and a cell in such a bundle belongs to at most one 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. The 6TiSCH architecture provides additional means to avoid waste of cells as well as overflows in the transmit bundle, 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 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 in the IPv6 header is set to Deterministic Forwarding, as discussed in Section 4.1. 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.

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

3.4.1. 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.

```
                        |                                   ^
   +--------------+     |                                   |
   |     IPv6     |     |                                   |
   +--------------+     |                                   |
   |  6LoWPAN HC  |     |                                   |
   +--------------+  ingress                              egress
   |     6top     |   sets     +----+          +----+     restores
   +--------------+  dmac to   |    |          |    |     dmac to
   |   TSCH MAC   |   brdcst   |    |          |    |      self
   +--------------+     |      |    |          |    |       |
   |   LLN PHY    |     +-------+    +--...-----+    +-------+
   +--------------+
```

Track Forwarding, Transport Mode

3.4.2. 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.

```
      +--------------+
      |     IPv6     |
      +--------------+
      |  6LoWPAN HC  |
      +--------------+             set            restore
      |     6top     |            +dmac+          +dmac+
      +--------------+          to|brdcst       to|nexthop
      |   TSCH MAC   |            |    |           |    |
      +--------------+            |    |           |    |
      |   LLN PHY    |    +-------+    +--...-----+    +-------+
      +--------------+    |   ingress                 egress   |
                          |                                    |
      +--------------+    |                                    |
      |   LLN PHY    |    |                                    |
      +--------------+    |                                    |
      |   TSCH MAC   |    |                                    |
      +--------------+    | dmac =                             | dmac =
      |ISA100/WiHART |    | nexthop                            v nexthop
      +--------------+
```

Figure 5: 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.

3.4.3. 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. Operations of Interest for DetNet and PCE

In a classical system, the 6TiSCH device does not place the request for bandwidth between self and another device in the network. Rather, an Operation Control System invoked through an Human/Machine Interface (HMI) indicates the Traffic Specification, in particular in terms of latency and reliability, and the end nodes. With this, the PCE must compute a Track between the end nodes and provision the network with per-flow state that describes the per-hop operation for a given packet, the corresponding timeSlots, and the flow identification that enables to recognize when a certain packet belongs to a certain Track, sort out duplicates, etc...

For a static configuration that serves a certain purpose for a long period of time, it is expected that a node will be provisioned in one shot with a full schedule, which incorporates the aggregation of its behavior for multiple Tracks. 6TiSCH expects that the programing of the schedule will be done over COAP as discussed in 6TiSCH Resource Management and Interaction using CoAP [I-D.ietf-6tisch-coap].

But an Hybrid mode may be required as well whereby a single Track is added, modified, or removed, for instance if it appears that a Track does not perform as expected for, say, PDR. For that case, the expectation is that a protocol that flows along a Track (to be), in a fashion similar to classical Traffic Engineering (TE) [CCAMP], may be used to update the state in the devices. 6TiSCH provides means for a device to negotiate a timeSlot with a neighbor, but in general that flow was not designed and no protocol was selected and it is expected that DetNet will determine the appropriate end-to-end protocols to be used in that case.

```
                        Stream Management Entity

                        Operational System and HMI

      -+-+-+-+-+-+-+ Northbound -+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-

                PCE         PCE              PCE              PCE

      -+-+-+-+-+-+-+ Southbound -+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-

              --- 6TiSCH------6TiSCH------6TiSCH------6TiSCH--
     6TiSCH /     Device      Device      Device      Device   \
     Device-                                                    - 6TiSCH
            \     6TiSCH      6TiSCH      6TiSCH      6TiSCH   /  Device
              ----Device------Device------Device------Device--
```

Figure 6

4.1. Packet Marking and Handling

Section "Packet Marking and Handling" of [I-D.ietf-6tisch-architecture] describes the packet tagging and marking that is expected in 6TiSCH networks.

4.1.1. Tagging Packets for Flow Identification

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

It results that the tagging that is used for a DetNet flow outside the 6TiSCH LLN MUST be swapped into 6TiSCH formats and back as the packet enters and then leaves the 6TiSCH network.

Note: The method and format used for encoding the RPLInstanceID at 6lo is generalized to all 6TiSCH topological Instances, which includes Tracks.

4.1.2. 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 MUST enable correlated timeSlots to be grouped for transmit (and respectively receive) with a 'OR' relations, and then a 'AND' relation MUST 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.

4.1.3. 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].

4.2. Topology and capabilities

6TiSCH nodes are usually IoT devices, characterized by very limited amount of memory, just enough buffers to store one or a few IPv6 packets, and limited bandwidth between peers. It results that a node will maintain only a small number of peering information, and will not be able to store many packets waiting to be forwarded. Peers can be identified through MAC or IPv6 addresses, but a Cryptographically Generated Address [RFC3972] (CGA) may also be used.

Neighbors can be discovered over the radio using mechanism such as beacons, but, though the neighbor information is available in the 6TiSCH interface data model, 6TiSCH does not describe a protocol to pro-actively push the neighborhood information to a PCE. This protocol should be described and should operate over CoAP. The protocol should be able to carry multiple metrics, in particular the same metrics as used for RPL operations [RFC6551]

The energy that the device consumes in sleep, transmit and receive modes can be evaluated and reported. So can the amount of energy that is stored in the device and the power that it can be scavenged from the environment. The PCE SHOULD be able to compute Tracks that will implement policies on how the energy is consumed, for instance balance between nodes, ensure that the spent energy does not exceeded the scavenged energy over a period of time, etc...

5. IANA Considerations

This specification does not require IANA action.

6. Security Considerations

On top of the classical protection of control signaling that can be expected to support DetNet, it must be noted that 6TiSCH networks operate on limited resources that can be depleted rapidly if an attacker manages to operate a DoS attack on the system, for instance by placing a rogue device in the network, or by obtaining management control and to setup extra paths.

7. Acknowledgments

This specification derives from the 6TiSCH architecture, which 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: Kris Pister, Thomas Watteyne, Xavier Vilajosana, Qin Wang, Tom Phinney, Robert Assimiti, Michael Richardson, Zhuo Chen, Malisa Vucinic, Alfredo Grieco, Martin Turon, Dominique Barthel, Elvis Vogli, Guillaume Gaillard, Herman Storey, Maria Rita Palattella, Nicola Accettura, Patrick Wetterwald, Pouria Zand, Raghuram Sudhaakar, and Shitanshu Shah for their participation and various contributions.

8. References

8.1. Normative References

[I-D.ietf-6tisch-6top-interface]
Wang, Q., Vilajosana, X., and T. Watteyne, "6TiSCH Operation Sublayer (6top) Interface", draft-ietf-6tisch-6top-interface-03 (work in progress), March 2015.

[I-D.ietf-6tisch-architecture]
Thubert, P., "An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4", draft-ietf-6tisch-architecture-08 (work in progress), May 2015.

[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-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-04 (work in progress), March 2015.

[I-D.ietf-6tisch-tsch]
Watteyne, T., Palattella, M., and L. Grieco, "Using IEEE802.15.4e TSCH in an IoT context: Overview, Problem Statement and Goals", draft-ietf-6tisch-tsch-06 (work in progress), March 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

8.2. Informative References

[I-D.finn-detnet-architecture]
Finn, N., Thubert, P., and M. Teener, "Deterministic Networking Architecture", draft-finn-detnet-architecture-01 (work in progress), March 2015.

[I-D.ietf-ipv6-multilink-subnets]
Thaler, D. and C. Huitema, "Multi-link Subnet Support in IPv6", draft-ietf-ipv6-multilink-subnets-00 (work in progress), July 2002.

[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-03 (work in progress), March 2015.

[I-D.thubert-6lowpan-backbone-router]
Thubert, P., "6LoWPAN Backbone Router", draft-thubert-6lowpan-backbone-router-03 (work in progress), February 2013.

[I-D.wang-6tisch-6top-sublayer]
Wang, Q., Vilajosana, X., and T. Watteyne, "6TiSCH Operation Sublayer (6top)", draft-wang-6tisch-6top-sublayer-01 (work in progress), July 2014.

[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, December 1998.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

[RFC3444] Pras, A. and J. Schoenwaelder, "On the Difference between Information Models and Data Models", RFC 3444, January 2003.

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, March 2005.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC4903] Thaler, D., "Multi-Link Subnet Issues", RFC 4903, June 2007.

[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, August 2007.

[RFC6282] Hui, J. and P. Thubert, "Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks", RFC 6282, September 2011.

[RFC6550] Winter, T., Thubert, P., 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, March 2012.

[RFC6551] Vasseur, JP., Kim, M., Pister, K., Dejean, N., and D. Barthel, "Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks", RFC 6551, March 2012.

[RFC6775] Shelby, Z., Chakrabarti, S., Nordmark, E., and C. Bormann, "Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6775, November 2012.

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/>.

[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".

[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
Internet-Draft
Intended status: Informational
Expires: April 21, 2016

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

October 19, 2015

6TiSCH Operation Sublayer (6top)
draft-wang-6tisch-6top-sublayer-03

Abstract

This document defines the 6TiSCH Operation Sublayer (6top), which offers mechanisms for distributed scheduling in 6TiSCH networks. The 6top sublayer is the next higher layer of the IEEE802.15.4e TSCH medium access control layer. The 6top Protocol (6P) defined in this document allows neighbor nodes to add/delete TSCH cells to one another. To be able to match different application requirements, the 6top Scheduling Function (SF) decides when to add/delete cells. The SF is left out of scope, and will be specified in one or more companion documents.

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

Copyright Notice

Table of Contents

1. Introduction

All communication in a 6TiSCH network is orchestrated by a schedule [RFC7554]. This specification defines the mechanisms offered by the 6TiSCH Operation Sublayer (6top) sublayer. These mechanisms allow a node to communicate with its neighbor node(s) to agree on a TSCH schedule in a distributed manner.

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

Figure 1: A simple 6TiSCH network.

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

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

This results in a distributed schedule management solution.

The mechanisms needed to enable this interaction are defined by the 6TiSCH Operation Sublayer (6top) sublayer, described in Section 2. The 6top Protocol (6P), specified in Section 3, defines the communication between neighbor nodes in this context. The 6top sublayer includes a 6top Scheduling Function (SF) which defines the policy of 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. One or more SFs will be defined in one or more companion documents. Section 4 provides some guidelines on how to design an SF.

2. 6TiSCH Operation Sublayer (6top)

As depicted in Figure 2, the 6TiSCH Operation Sublayer (6top) sits directly above the IEEE802.15.4e TSCH medium access control layer [IEEE802154e].

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

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

The roles of the 6top sublayer are:

- o Implement and terminate the 6top Protocol (6P), which allows neighbor nodes to communicate to add/delete cells to one another.
- o Run a 6top Scheduling Function (SF) which defines the algorithm to decide when to add/delete cells.
- o Offer a way for a neighbor node to discover which SF is being used.

2.1. Hard/Soft Cells

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

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

In the context of this specification, all the cells used by 6top are Soft Cells. Hard cells can be used for example when "hard-coding" a cell (e.g. the 6TiSCH Configuration [I-D.ietf-6tisch-minimal]).

2.2. Using 6top with the Minimal 6TiSCH Configuration

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

- o Slotframe 0 (SFR0) 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 (SFR1) is used by 6top to allocate cells from. In Figure 3, this slotframe is 10 slots long, but it can be of any length.

.

SFR0 SHOULD be of higher priority than SFR1. 6top MAY support further slotframes; how to use more slotframes is out of the scope for this document.

```
          | 0    1    2    3    4  | 0    1    2    3    4  |
          +-------------------------+-------------------------+
     SFR0 | EB |    |    |    |    | EB |    |    |    |    |
          |    |    |    |    |    |    |    |    |    |    |
          +-------------------------------------------------+
     SFR1 |    |A->B|    |    |    |    |    |    |B->A|    |
          |    |    |    |    |    |    |    |    |    |    |
          +-------------------------------------------------+
          | 0    1    2    3    4    5    6    7    8    9  |
```

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

3. 6top Protocol (6P)

The 6top Protocol (6P) allows two neighbor nodes to pass information to add/delete cells to their TSCH schedule. This information is carried as IEEE802.15.4 Information Elements (IE) [IEEE802154e] and travels only a single hop.

Conceptually, two neighbor nodes "negotiate" the location of the cells to add/delete. We reuse the topology in Figure 1 to illustrate how the protocol works.

When node A wants to add (resp. delete) 2 cells to node B:

1. Node A sends a message to node B indicating it wants to add (resp. delete) 2 cells to node B to its schedule, and listing 2 or more candidate cells.
2. Node B responds with a message indicating that the operation succeeded, and specifying which cells from the candidate list it added (resp. deleted). This allows node A to add (resp. delete) the same cells to/from its schedule.

Figure 4 is a sequence diagram which illustrates this exchange. Here, node A requests 2 cells to node B. It sends a 6P ADD Request to node be indicatig it wishes to add 2 cells (the "NumCells" value), and specifying a list of 3 candidate cells from which node B can choose (the "CellList" value). Each cell in the CellList is a tuple with the (slotOffset,channelOffset) coordinates of the candidate cell in the TSCH schedule. Node B selects 2 of the 3 cells in the CellList of the 6P ADD Request, and sends a 6P Response back to node A specifying the cells it selected from the specified container (e.g Slotframe, Chunk, etc ...). This allow nodes A and B to add those two cells to their schedule.

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

Figure 4: Sequence diagram to illustrate the 6P negotiation.

We call "6P Transaction" the action of two neighbor nodes exchanging a 6P Request Message and the corresponding 6P Reply message.

3.1. Message Format

3.1.1. 6top Information Element

The messages exchanges as part of the 6P protocol are carried in a 6top Information Element. The 6top Information Element is a IETF IE with Group ID IANA_6TOP_IE_GROUP_ID. The length of the 6top

Information Element is variable. The content of the 6top Information Element is specified in Section 3.1. TODO: IETF IE specified in Appendix A for now, but to be specified in separate draft in the future.

3.1.2. General Message Format

All 6P messages have the following format:

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

Ver (6P Version): The version of the 6P protocol. Only version IANA_6P_VERSION is defined in this document. Future specification might define further version of the 6P protocol.
Code: Command to carry out, or response code. The list of command identifiers and return codes is defined only for version IANA_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 4.1.
Other Fields: The list of other fields depends on the value of the code field, as detailed below.

3.1.3. 6P Command Identifiers

Figure 5 lists the 6P command identifiers.

Value	Command ID	Description
IANA_CMD_ADD	CMD_ADD	add one or more cells
IANA_CND_DELETE	CMD_DELETE	delete one or more cells
IANA_CMD_COUNT	CMD_COUNT	count scheduled cells
IANA_CMD_LIST	CMD_LIST	list the scheduled cells
IANA_CMD_CLEAR	CMD_CLEAR	clear all cells
TODO-0xf	reserved	

Figure 5: 6P Command Identifiers

3.1.4. 6P Return Codes

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

Value	Return Code	Description
IANA_RC_SUCCESS	RC_SUCCESS	operation succeeded
IANA_RC_VER_ERR	RC_VER_ERR	unsupported 6P version
IANA_RC_SFID_ERR	RC_SFID_ERR	unsupported SFID
IANA_RC_ERR_BUSY	RC_ERR_BUSY	handling previous request
IANA_RC_RESET	RC_RESET	abort 6P transaction
IANA_RC_ERR	RC_ERR	operation failed
TODO-0xf	reserved	

Figure 6: 6P Return Codes

3.1.5. 6P Cell Format

The 6P Cell is an element which is present in several messages. It is a 4-byte field formatted as:

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

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

3.1.6. 6P ADD Request Format

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Ver   |  Code |    SFID       | NumCells      | Container     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| CellList ...
+-+-+-+-+-+-+-+-
```

Ver: Set to IANA_6P_VERSION.

Code: Set to IANA_CMD_ADD for a 6P ADD Request.
SFID: Identifier of the SF to be used by the receiver to handle the message.
NumCells: The number of additional TX cells the sender wants to schedule to the receiver.
Container: An indication of where in the schedule to take the cells from (which slotframe, which chunk, etc.). This value is an indication to the SF. The meaning of this field depends on the SF, and is hence out of scope of this document.
CellList: A list of 0, 1 or multiple 6P Cells. The format of a 6P Cell is defined in Section 3.1.5

3.1.7. 6P DELETE Request Format

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

3.1.8. 6P COUNT Request Format

```
                         1                   2
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | Ver   |  Code |    SFID       | Container     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Ver: Set to IANA_6P_VERSION.
Code: Set to IANA_CMD_COUNT for a 6P COUNT Request.
SFID: Identifier of the SF to be used by the receiver to handle the message.
Container: An indication of where in the schedule to take the cells from (which slotframe, which chunk, etc.). This value is an indication to the SF. The meaning of this field depends on the SF, and is hence out of scope of this document.

3.1.9. 6P LIST Request Format

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

3.1.10. 6P CLEAR Request Format

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

3.1.11. 6P Response Format

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

Ver: Set to IANA_6P_VERSION.
SFID: Identifier of the SF to be used by the receiver to handle the message.
Code: One of the 6P Return Codes listed in Section 3.1.4.
Other Fields: The fields depends on what command the request is for:

Response to an ADD, DELETE or LIST command: A list of 0, 1 or multiple 6P Cells. The format of a 6P Cell is defined in Section 3.1.5.
Response to COUNT command: The number of cells scheduled from the requestor to the receiver by the 6P protocol, encoded as a 2-octet unsigned integer.
Response to CLEAR command: No other fields are present in the response.

3.2. Protocol Behavior

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

3.2.1. 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_6P_VERSION), a node MAY implement multiple protocol versions at the same time. When receiving a 6P message with a Version number it does not implement, a node MUST reply with a 6P Response and a return code of IANA_RC_VER_ERR. The Version field in the 6P Response MUST be the same as the Version field in the corresponding 6P Request.

3.2.2. SFID Checking

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

3.2.3. Concurrent 6P Transactions

Only a single 6P Transaction between two neighbors, in a given direction, can take place at the same time. That is, a node MUST NOT issue a new 6P Request to a given neighbor before having received the 6P Response for a previous request to that neighbor. The only exception to this rule is when the previous 6P Transaction has timed out. If a node receives a 6P Request from a given neighbor before having sent the 6P Response to the previous 6P Request from that neighbor, it MUST send back a 6P Response with a return code of IANA_RC_ERR.

A node MAY support concurrent 6P Transactions from different neighbors. In this case, in Figure 1, node C can have a different ongoing 6P Transaction with nodes B and E. In case a node does not have enough resources to handle concurrent 6P Transactions from different neighbors, when it receives a 6P Request from a neighbor while already handling a different request from a different neighbor, it MUST reply to that second request with a 6P Response with return code IANA_RC_BUSY.

3.2.4. Timeout

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

3.2.5. Adding cells

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

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

Upon receiving the request, node A’s SF verifies which of the cells in the CellList it can add as receive cells from node C in its own schedule. How that selection is done is specified in the SF and out of scope of this document. That verification can succeed (NumCell cells from the CellList can be used), fail (none of the cells from the CellList can be used) or partially succeed (less than NumCell

cells from the CellList can be used). In all cases, node A MUST send a 6P Response with return code set to IANA_RC_SUCCESS, and which specifies the list of cells that were scheduled as receive cells from C. That can contain 0 elements (when the verification failed), NumCell elements (succeeded) or between 0 and NumCell elements (partially succeeded).

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

3.2.6. Aborting a 6P Transaction

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

3.2.7. Deleting cells

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

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

3.2.8. Handling error responses

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

3.3. 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.

4. Guidelines for 6top Scheduling Functions (SF)

4.1. 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	reserved
0x01-0xef	managed
0xf0-0xfe	unmanaged
0xff	reserved

Figure 7: SFID range.

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

4.2. Requirements for an SF

The specification for an SF

- o MUST specify an identifier for that SF.
- o SHOULD clearly state the application domain the SF is created for.
- o MUST specify the rule for a node to decide when to add/delete one or more cells to a neighbor.
- o MUST specify the rule for a Transaction source to select cells to add to the CellList field in the 6P ADD Request.
- o MUST specify the rule for a Transaction destination to select cells from CellList to add to its schedule.
- o MUST specify a value for the 6P Timeout, or a rule to calculate it.
- o MUST specify a meaning for the "Container" field in the 6P ADD Request.
- o MUST specify the behavior of a node when it boots.
- o MUST specify what to do after an error has occurred (either the node sent a 6P Response with an error code, or received one).

- o SHOULD contain examples which highlight normal and error scenarios.
- o SHOULD contain a list of current implementations, at least during the I-D state of the document, per [RFC6982].
- o SHOULD contain a performance evaluation of the scheme, possibly through references to external documents.

4.3. Recommended Structure of an SF Specification

The following section structure for a SF document is RECOMMENDED:

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

5. 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.

6. Security Considerations

TODO: analyze risks

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

7. IANA Consideration

- o TODO: IANA_6TOP_IE_GROUP_ID
- o TODO: IANA_6P_VERSION
- o TODO: IANA_CMD_ADD
- o TODO: IANA_CMD_DELETE
- o TODO: IANA_RC_SUCCESS
- o TODO: IANA_RC_VER_ERR
- o TODO: IANA_RC_ERR

8. References

8.1. 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>.

[IEEE802154e] IEEE standard for Information Technology, "IEEE std. 802.15.4e, Part. 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC sublayer", April 2012.

8.2. 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-12 (work in progress), September 2015.

[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-05 (work in progress), July 2015.

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

Appendix A. [TEMPORARY] IETF IE

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

The IETF IE is a IEEE802.15.4 Payload Information Element with the Group ID set to IANA_6TOP_IE_GROUP_ID. The value of IANA_6TOP_IE_GROUP_ID is defined by the IEEE, communicated to the IETF, and noted by IANA. The format of the IETF IE is exactly the same as the format of an MLME Information Element, as specified in [IEEE802154e], Section 5.2.4.5. The difference is that the space of Sub-IDs is managed by the IETF/IANA.

Appendix B. [TEMPORARY] IEEE Liaison Considerations

If the specification described in this document is supported by the 6TiSCH WG, the authors of this document ask the 6TiSCH WG chairs to liaise with the IEEE to request a Payload Information Element Group ID to be assigned to the IETF (Group ID IANA_6TOP_IE_GROUP_ID described in Appendix A).

Appendix C. [TEMPORARY] Terms for the Terminology Draft

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

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

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

SFID: The "6top Scheduling Function Identifier" (SFID) is a 4-bit field identifying a SF. It is defined in TODO_LINK_draft-wang-6tisch-6top-sublayer.

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

6P Transaction: Part of the "6top Protocol" (6P), the action of two neighbors exchanging a 6P request message and the corresponding 6P response message. It is defined in TODO_LINK_draft-wang-6tisch-6top-sublayer.

Appendix D. [TEMPORARY] Changelog

- -03
  -
  - https://bitbucket.org/6tisch/draft-wang-6tisch-6top-sublayer/issues/32/missing-command-list
  - https://bitbucket.org/6tisch/draft-wang-6tisch-6top-sublayer/issues/31/missing-command-count
  - https://bitbucket.org/6tisch/draft-wang-6tisch-6top-sublayer/issues/30/missing-command-clear
  - https://bitbucket.org/6tisch/draft-wang-6tisch-6top-sublayer/issues/37/6top-atomic-transaction-6p-transaction

* https://bitbucket.org/6tisch/draft-wang-6tisch-6top-sublayer/issues/35/separate-opcode-from-rc
* https://bitbucket.org/6tisch/draft-wang-6tisch-6top-sublayer/issues/36/add-length-field-in-ie
* https://bitbucket.org/6tisch/draft-wang-6tisch-6top-sublayer/issues/27/differentiate-rc_err_busy-and
* https://bitbucket.org/6tisch/draft-wang-6tisch-6top-sublayer/issues/29/missing-rc-rc_reset
* https://bitbucket.org/6tisch/draft-wang-6tisch-6top-sublayer/issues/28/the-sf-must-specify-the-behavior-of-a-mote
* https://bitbucket.org/6tisch/draft-wang-6tisch-6top-sublayer/issues/26/remove-including-their-number
* https://bitbucket.org/6tisch/draft-wang-6tisch-6top-sublayer/issues/34/6of-sf
* https://bitbucket.org/6tisch/draft-wang-6tisch-6top-sublayer/issues/33/add-a-figure-showing-the-negociation

o -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