Demo: Scheduling Function Zero on a 6TiSCH Network

Tengfei Chang
Inria, EVA team
Paris, France
tengfei.chang@inria.fr

Pere Tuset
Universitat Oberta de Catalunya
WiNe team, Barcelona, Spain
peretuset@uoc.edu

Xavier Vilajosana
Universitat Oberta de Catalunya
WiNe team, Barcelona, Spain
xvilajosana@uoc.edu

Thomas Watteyne
Inria, EVA team
Paris, France
thomas.watteyne@inria.fr

Abstract

The IETF 6TiSCH standard combines the high reliability of Time Synchronized Channel Hopping and the ease-of-use of IPv6, for demanding industrial applications. Scheduling Function Zero (SF0) is a distributed scheduling function in charge of calculating the bandwidth needed by each of the nodes in a 6TiSCH network. A relocation policy detects and relocates cells which suffer from a schedule collision, to ensure the link reliability. This demo consists of OpenMote devices running the OpenWSN implementation, including the 6TiSCH minimal configuration, the 6top Protocol, and SF0. Visualization tools allow an attendee to see the relocation happening.

1 The 6TiSCH Standards

6TiSCH [2, 5] is an IETF working group standardizing the convergence of IPv6 and low-power industrial wireless communications rooted in the IEEE802.15.4-2015 TSCH standard [3]. 6TiSCH runs on top of IEEE802.15.4-2015 TSCH and exploits the industrial performance provided by Time Slotted Channel Hopping (TSCH) at the link layer. On top of TSCH, 6TiSCH uses the IoT upper stack designed at the IETF for constrained devices [4]. This combination of industrial performance and Internet-readiness enables nodes to acquire an IPv6 address (6LoWPAN), form a multi-hop topology (RPL) and have web interactions (CoAP).

TSCH defines how to transmit a link-layer frame to a neighbor, and get an acknowledgment, on a timeslot and on a frequency. The [timeslot, frequency offset] tuple is called a "cell". A collection of cells is organized in a slotframe that repeats over time. A scheduler allocates cells to pairs of neighbor nodes wanting to communicate. The IEEE802.15.4-2015 TSCH standard does not indicate how to create and manage this schedule.

International Conference on Embedded Wireless
Systems and Networks (EWSN) 2017
20–22 February, Uppsala, Sweden

ISBN: 978-0-9949886-1-4

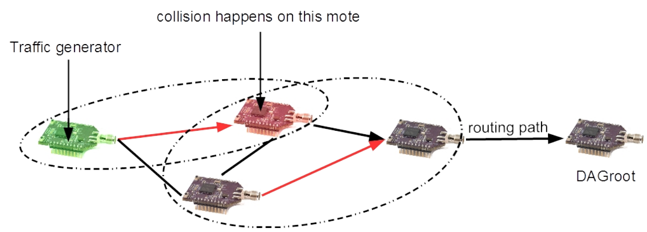

Figure 1. 6TiSCH SF0 demo setup. The green mote acts as a traffic generator. The arrow indicates the next hop on routing path. The red arrow indicates two links prone to collision. The cells are allocated with SF0.

The IETF 6TiSCH Working Group standardizes the 6top Protocol (6P) [7] and the Scheduling Function Zero (SF0) [1]. 6P allows neighbor nodes to add/remove cells between them; SF0 is an algorithm running on each mote which decides how many cells is needs to each of its neighbors. 6P and SF0 form a distributed schedule management solution for TSCH networks.

1.1 The 6top Protocol (6P)

6P defines the message formats and states machine to support different policies for distributed scheduling.

When node `A` needs to add one or more cells to communicate with its neighbor `B`, `A` determines a list of candidate cells according to its local information and forms a *6P Request* command which it sends to `B`. After receiving it, `B` checks the availability of those cells in its own schedule. `B` selects a subset of the candidate list received from `A`, and forms a *6P Response* command which it sends back to `A`. Once `A` receives it, both `A` and `B` add these cells to their local schedule.

6P defines a CellOptions field in the command so the type of cell can be selected. For example, `A` can schedule a transmission slot (Tx) to `B`; `B` then adds a Reception slot (Rx) from `A`. 6P provides other management commands such as `STATUS`, `LIST` and `RELOCATE` [8].

1.2 Scheduling Function Zero (SF0)

SF0 is the algorithm which decides when to add/remove cells to satisfy the bandwidth requirements of each node. SF0 defines a parameter called `SF0THRESH` to control the over-provisioning of cells during the operation. SF0 imposes three rules for adding or deleting a cell:

- if requiredCells $<$ scheduledCells-SF0THRESH, delete one or more cells,
- if scheduledCells-SF0THRESH$\leq$requiredCells and requiredCells$\leq$scheduledCells, do nothing,
- if scheduledCells$\leq$requiredCells, add one or more cells.

SF0 selects cells randomly from the local schedule of the node. Without global knowledge, pairs of neighbor nodes may choose the same cell, resulting in collisions. SF0 defines a relocation policy which detects such schedule collisions, and a relocation policy which "moves" collided cells elsewhere in the schedule.

2 Description of the Demo

We implement the protocols described in Section in OpenWSN. OpenWSN [9] is a standard-based open source protocol implementation of the IETF 6TiSCH stack and the IEEE802.15.4-2015 TSCH standard. It is the reference implementation of 6TiSCH protocol stack, and includes the 6P and SF0 protocols. The OpenWSN source code is separated into two parts. The *firmware* runs on constrained devices such as the OpenMote, and implements 6TiSCH. The *software* (the "OpenVisualizer") runs on the Raspberry Pi-like computer sitting at the egde of the mesh network. It monitors the network status and provides bridging capabilities from the 6TiSCH mesh network to the Internet. In the proposed demo, we use the OpenVisualizer to show attendees the network status, for example, the number of cell reserved through SF0.

The implementation runs on OpenMote devices. OpenMote [6] is an open-source hardware designed at UC Berkeley and commercialized by OpenMote Technologies[1]. It features the CC2538, a single-chip solution for low power wireless networks. The OpenMote is powered by the OpenUSB with a pair of AA batteries; the OpenUSB also allows a computer to reprogram the OpenMote.

The proposed demo consists of 10-20 OpenMotes deployed across the demo floor. All motes run the 6TiSCH protocol with 6P and SF0 implemented. The resulting wireless network is organized as a low-power multihop mesh network, the topology of which one can see on the OpenVisualizer. One of the motes (the green mote in Fig. 1) acts as a data generator. All other motes forward its packet until it reaches the DAGroot. By changing the rate at which data is generated, one can see SF0 calculating different bandwidth requirements, and cells being added/removed.

The demo demonstrates two important aspects. First, when changing traffic, the bandwidth used by the motes changes: all motes along the multi-hop path between source and destination add/remove cells, per the SF0 bandwidth estimation algorithm. Second, as shown in Fig. 1, there can be a schedule collision. In this case, SF0's relocation policy kicks in to relocate the cell to a different location in the TSCH schedule. Both behavior can be followed "live" on the OpenVisualizer.

[1] www.openmote.com

3 Conclusions

6TiSCH provides a protocol stack for industrial IoT applications. 6P and SF0 – part of 6TiSCH – calculate and allocate the networking resources needed by the motes. This demo consists of several OpenMote devices running OpenWSN, demonstrating 6P and SF0. By changing the traffic on one node, the nodes in the network dynamically change the number of cells they have scheduled to their neighbor. This is handled in a distributed transparent way by the 6P protocol and the SF0 scheduling policy. When a schedule collision happens, SF0 relocates the cells involved. All these operations can be seen "live" on the visualization tools of OpenWSN.

4 References

[1] D. Dujovne, L. Grieco, M. Palattella, and N. Accettura. 6TiSCH 6top Scheduling Function Zero (SF0), Oct. 2016.

[2] D. Dujovne, T. Watteyne, X. Vilajosana, and P. Thubert. 6TiSCH: deterministic IP-enabled industrial internet (of things). *IEEE Communications Magazine*, 52:36–41, Dec. 2014.

[3] IEEE. 02.15.4-2015: IEEE Approved Draft Standard for Low-Rate Wireless Personal Area Networks (WPANs), 2015.

[4] M. R. Palattella, N. Accettura, X. Vilajosana, T. Watteyne, L. A. Grieco, G. Boggia, and M. Dohler. Standardized protocol stack for the internet of (important) things. *IEEE Communications Surveys Tutorials*, 15(3):1389–1406, Third 2013.

[5] M. R. Palattella, P. Thubert, X. Vilajosana, T. Watteyne, Q. Wang, and T. Engel. *6TiSCH Wireless Industrial Networks: Determinism Meets IPv6*, pages 111–141. Springer International Publishing, Cham, 2014.

[6] X. Vilajosana, P. Tuset, T. Watteyne, and K. Pister. OpenMote: Open-Source Prototyping Platform for the Industrial IoT. *Ad Hoc Networks, Springer Link*, 155:211–222, November 2015.

[7] Q. Wang and X. Vilajosana. 6top Protocol (6P). Technical Report draft-ietf-6tisch-6top-protocol [work in progress], Internet Engineering Task Force, 2016.

[8] Q. Wang and X. Vilajosana. 6top Protocol (6P), Oct. 2016.

[9] T. Watteyne, X. Vilajosana, B. Kerkez, F. Chraim, K. Weekly, and Q. Wang. OpenWSN: a standards-based low-power wireless development environment. *Transactions on Emerging Telecommunications Technologies, Wiley*, 23:480–493, August 2012.