Poster: An Open-Source IPv6 over BLE Stack for Contiki

Michael Spörk, Markus Schuß, Carlo Alberto Boano, and Kay Römer
Institute for Technical Informatics, Graz University of Technology, Austria
{michael.spoerk, markus.schuss, cboano, roemer}@tugraz.at

Abstract

Bluetooth Low Energy (BLE) has gained increasing popularity over the past years because of its energy efficiency, reliability under interference, and wide adoption in consumer devices such as smartphones, tablets, and wearables. In 2015, the IETF formalized the RFC 7668 standard that defines how BLE connections can be used to exchange IPv6 packets, practically allowing any BLE device to seamlessly connect to the Internet of Things. To date, however, very few open-source implementations of this standard are available to the research community. To fill this gap, in this poster we introduce *ContikiBLE*, the first fully open-source IPv6 over BLE communication stack for the Contiki operating system, and present an evaluation of its main features using off-the-shelf TI CC2650 SensorTag wireless sensor nodes.

1 Motivation

An increasing number of real-world objects are becoming accessible and manageable through the Internet. A large portion of these devices is based on IEEE 802.15.4, the defacto link and physical layer standard for low-power wireless personal area networks, and uses 6LoWPAN as key enabling technology to allow integration into IPv6 networks.

Recently, a number of wireless technologies are becoming appealing alternatives to IEEE 802.15.4. In particular, Bluetooth Low Energy [1] fits the requirements of smart objects (such as limited processing capabilities, memory size, and constrained battery capacity) and comparative measurements have shown that its power consumption is lower than that of IEEE 802.15.4 [3, 6]. Another key advantage of BLE is its wide adoption in consumer electronic devices such as high-end smartphones, wearables, and laptops. These devices are typically connected to the Internet, and may hence be used as border routers to easily provide smart objects supporting BLE with an access to the Internet of Things.

In answer to this raising popularity of BLE, the IETF released in October 2015 the RFC 7668 standard (also called IPv6 over BLE) describing how IPv6 packets can be exchanged using BLE connections [4]. According to the standard, an IPv6 over BLE network has a border router (e.g., a BLE-enabled smartphone) and one or more end-nodes (e.g., BLE-enabled sensors that publish their measurements to the Internet). The border router provides the node devices with an IPv6 prefix and Internet access. The standard also defines fragmentation and reassembly of large IPv6 packets BLE's L2CAP protocol, and the compression of IPv6 base header fields that can be derived from the BLE connection.

Although the standard was released more than a year ago, very few open-source implementations of RFC 7668 exist, and no entirely open-source IPv6 over BLE stack for the Contiki OS has been released yet. An IPv6 over BLE open-source stack for Contiki would significantly support the research community in carrying out further research on BLE, in the same way as Contiki's *sicslowpan* over IEEE 802.15.4 implementation drove a large body of works in the past years.

To fill this gap, we have designed **ContikiBLE**[1], the first fully open-source IPv6 over BLE communication stack for the Contiki OS [2], and created a prototype implementation for the TI CC2650 SensorTag platform. The design of ContikiBLE is driven by four main goals. First, the communication stack is *fully compliant* to the RFC 7668 standard, fulfilling all aforementioned specifications. Second, any smart object running ContikiBLE is *interoperable* with other IPv6 devices and hence is able to seamlessly connect to the Internet via a border router. Third, ContikiBLE is easily *portable* to other Contiki hardware platforms with BLE support by simply adding a new hardware abstraction layer. Furthermore, the architecture of ContikiBLE was designed to be *fully compatible* with the existing architecture of Contiki. This allows to design applications that are agnostic to the network stack, i.e., developers can use the same application and simply change the network stack configuration in the `project-conf.h` file of the Contiki application.

2 ContikiBLE

ContikiBLE provides an IPv6 over BLE implementation that is fully compliant to the RFC 7668 and that uses the same layered architecture of the existing network stack of Contiki. Figure 1 provides an overview of ContikiBLE.

[1]https://github.com/spoerk/contiki/tree/sensortag_rfc7668

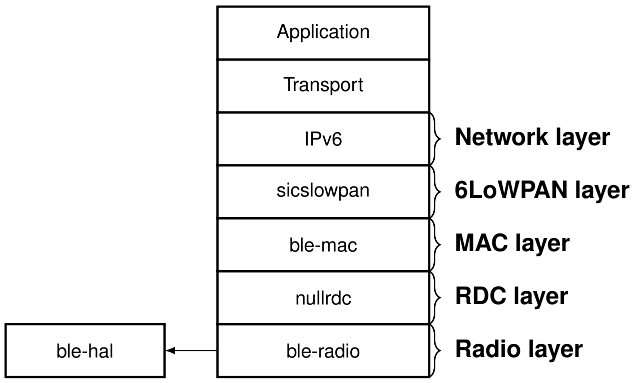

Figure 1. ContikiBLE network stack.

Radio layer. The radio layer implements all functionality needed to setup a BLE connection and to exchange IPv6 packets. It is split into two parts: a hardware abstraction layer (`ble-hal`) implementing all hardware-specific BLE code, and a generic part (`ble-radio`) connecting the abstraction layer to the upper layers of the network stack. Porting ContikiBLE to other Contiki hardware with BLE support only requires to adapt the `ble-hal` to the new platform: all other parts of ContikiBLE are hardware-independent.

RDC layer. BLE connections already implement basic duty cycling of the radio, therefore the existing `nullrdc` is used for our prototype implementation. Future revisions of ContikiBLE may implement an RDC layer that controls the connection parameters of the BLE connection and that allows to close BLE connections for extended periods of time.

MAC layer. The `ble-mac` layer implements L2CAP's connection setup, fragmentation, and reassembly functionality, making it possible to split large IPv6 packets into smaller fragments that can be exchanged over BLE connections with limited packet length. The current version of `ble-mac` supports a maximum IPv6 packet length of 1280 bytes.

6LoWPAN layer. ContikiBLE makes use of the existing `sicslowpan` implementation of Contiki and its IPv6 header compression to elide the fields that can be derived from the BLE connection. Since packet fragmentation and reassembly are performed at the MAC layer, the fragmentation mechanism of `sicslowpan` is disabled.

Network layer. ContikiBLE uses the existing IPv6 neighbor discovery implementation of Contiki to exchange subnet information between BLE routers and end-devices.

3 Evaluation

We evaluate our prototype implementation of ContikiBLE using the TI SensorTag CC2650 hardware platform.

IPv6 interoperability. First, we connect a SensorTag to a border router (Raspberry Pi running Raspbian Linux) using a LogiLink BT0015 BLE/USB dongle, and verify that it can be pinged from a laptop connected via Ethernet. Our measurements show that ICMPv6 echo request and response messages can be successfully exchanged for a packet length of up to the minimum required MTU size (1280 bytes), with an average latency below 350 ms, as shown in Figure 2.

Energy consumption. We measure the energy consumption of a SensorTag exchanging 100 UDP packets with the

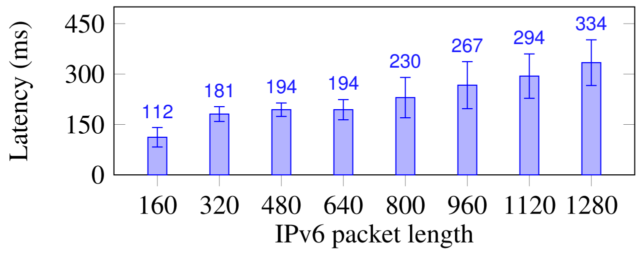

Figure 2. Ping latency for different IPv6 packet lengths.

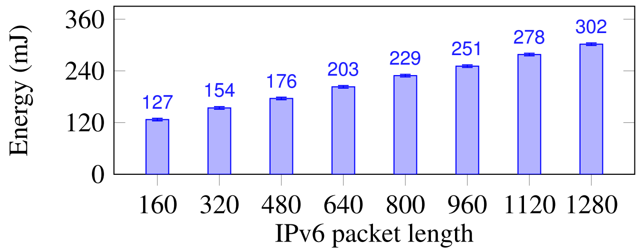

Figure 3. Energy consumption of ContikiBLE.

border router (one packet/second) in hardware using D-Cube [5]. Figure 3 shows the average energy consumption as a function of the length of the exchanged IPv6 packets. One can notice that the energy consumption increases linearly with the exchanged IPv6 packet length.

Reliability. Next, we introduce Wi-Fi interference in our experimental setup and measure the robustness of BLE's frequency hopping algorithm. Our measurements show that BLE is able to escape interference and sustain a packet reception rate of 100% regardless of the payload length.

4 Next Steps

This prototype version of ContikiBLE should act as a seed for further research on IPv6 over BLE. As a next step, we plan to port ContikiBLE to other hardware platforms with BLE support, such as the Nordic Semiconductor nRF52, and to create a version of the stack that supports connectionless BLE communication. Furthermore, we plan to compare ContikiBLE to the existing IPv6 over IEEE 802.15.4 communication stack of Contiki in regard to communication latency, energy efficiency, and reliability under interference.

5 References

[1] Bluetooth SIG. Specification of the Bluetooth System – Covered Core Package version: 4.1. `https://www.bluetooth.org/en-us/specification/adopted-specifications`, 2013.

[2] A. Dunkels, B. Grönvall, and T. Voigt. Contiki – a Lightweight and Flexible Operating System for Tiny Networked Sensors. In *Proc. of the 1st EmNetS Workshop*, 2004.

[3] P. Narendra, S. Duquennoy, and T. Voigt. BLE and IEEE 802.15.4 in the IoT: Evaluation and Interoperability Considerations. *Proc. of the 1st EAI InterIoT Conference*, 2015.

[4] J. Nieminen et al. RFC 7668 - IPv6 over BLUETOOTH(R) Low Energy. `https://tools.ietf.org/html/rfc7668`, 2015.

[5] M. Schuß, C. A. Boano, M. Weber, and K. Römer. A competition to push the dependability of low-power wireless protocols to the edge. In *Proc. of the 14th EWSN Conference*, 2017.

[6] M. Siekkinen et al. How low energy is Bluetooth Low Energy? Comparative measurements with Zigbee/802.15.4. In *Proc. of the Wireless Communications and Networking Conf. Workshops (WCNCW)*, 2012.