Demo: One Step Closer to Ultra-stable Bluetooth Low Energy Multi-hop Networks

Bozheng Pang
TU Dresden
Dresden, Germany
bozheng.pang@tu-dresden.de

Thomas C. Schmidt
HAW Hamburg
Hamburg, Germany
t.schmidt@haw-hamburg.de

Matthias Wählisch
TU Dresden
Dresden, Germany
m.waehlisch@tu-dresden.de

Abstract

Clock drift is a challenge in Bluetooth Low Energy (BLE) multi-hop networks. It leads to the drift among connections managed by a single BLE device, and eventually the overlap of radio events. Such overlaps impact the performance of the BLE multi-hop network, e.g., by connection loss and network collapse. In this demo, we present an extremely lightweight and highly effective solution to mitigate the drift among connections. The core idea is to fine-tune the anchor point of the connections managed by every BLE node. Our results show that our solution can help BLE multi-hop networks avoid connection drift, thus achieving the stability.

CCS Concepts

• **Networks** → **Topology analysis and generation**; **Network manageability**; • **Hardware** → **Clock generation and timing**.

Keywords

Bluetooth Low Energy (BLE), Multi-hop, Clock drift, Stability

1 Introduction

Bluetooth Low Energy (BLE) is a point-to-point communication technology [7]. To make it more suitable for Internet of Things (IoT) deployment, BLE Mesh appears. However, the performance of BLE Mesh is not ideal due to its connectionless characteristic, especially when the network is with a high density and high amount of data [8]. As a more promising solution, researchers establish a multi-hop network on top of the connection-oriented mode of BLE [6].

A BLE multi-hop network is connection-based, so it expected to have a more predictable performance than BLE Mesh. However, after the realization, researchers notice a key challenge, i.e., clock drift [6]. Since each BLE node connects to multiple other BLE nodes, the clock difference among them makes the connections existing on each node drift slowly. Such drifts accumulate and eventually lead to the overlap between radio events of the connections on the BLE node. The overlap forces the BLE node to drop some radio events and leads to the performance drop, instead of commonly assumed external interference [4, 5]. Hence, the multi-hop network faces connection loss and even network collapse.

In this demo, we showcase an extremely lightweight and highly effective method to mitigate the impact of clock drift in BLE multi-hop networks. The idea is to introduce minor adjustment to the anchor point of the BLE connections on each BLE node. We implement BLE multi-hop networks without and with the method, to show the effect.

We briefly describe the method and its implementation in section 2. We show the demo setup and description in section 3. Afterwards, we illustrate the expected results of the demo in section 4. A brief discussion is in section 5. Future work is given in section 6.

2 Method and Implementation

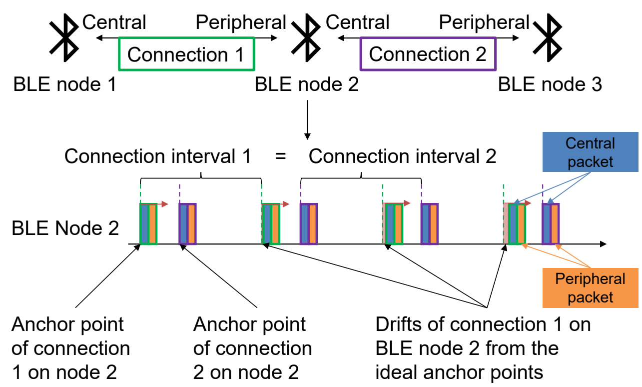

Figure 1: An illustration of the relative drift between two connections existing on a BLE node. BLE node 2 calibrates its anchor point of connection 1 according to BLE node 1, and does not do so for connection 2. As a result, on BLE node 2, the time length between the neighboring anchor points of the two connections changes.

In BLE communication, the data exchange between two nodes (central role and peripheral role) occurs according to a fixed time interval, i.e., connection interval. The starting point of each connection interval is an anchor point. When a BLE node plays different roles in multiple connections, these connections relatively drift, shown as anchor points moving on the BLE node (see Figure 1).

The core idea of our method is to fine-tune the anchor point of the connections managed by every BLE device. In Figure 1, node 2 plays different roles in the two connections, i.e., peripheral in connection 1 and central in connection 2. As a result, the BLE node must calibrate its anchor point in connection 1 according to its central, but it can decide the anchor point for connection 2. This offers the opportunity to mitigate the relative drift by fine-tuning the anchor points of connection 2. Since the BLE node manages both connections, it naturally has the information of both of them. This way, without extra data exchange, the BLE node knows the drift details, such as the drift direction and speed. By such information, we can manually manipulate the anchor point of connection 2 to drift in the same direction and speed of connection 1. We achieve this by injecting microsecond-level adjustments into anchor point calculation in the link layer of BLE. Hence, the two connections realize a relatively static status on the BLE node.

Our method needs two conditions to support its proper functionality. First, there must be a leader device (the node with the largest MAC address) defined in the network to be the clock source. The leader device should only be central in all of its connections. Second, all the other BLE nodes should hold both the central and peripheral roles. We realize this by repeated disconnection and reconnection.

3 Demo Setup and Description

Table 1: Demo hardware list

Hardware	Number
Adafruit Feather nRF52840 Sense	15
Micro USB cable	15
USB hub + power supply	1
Monitor + typec/HDMI cable + power supply	1
Power socket	1

We give the description of our demo setup in this section, to illustrate the requirements. We list all the required hardware in Table 1, and explain their usage in later contents.

The demo bases itself on Adafruit Feather nRF52840 Sense development boards. We need 15 of them to establish two BLE networks separately, one with only 5 BLE nodes and the other one with 10 BLE nodes. The 5 BLE nodes use the pure BLE stack to establish the network without the implementation of our method. The 10-node BLE network implements our method on top the BLE stack, thus as a comparison to the behavior and performance of the 5-node BLE network. We use 5 more nodes in the second BLE network to further show the effectiveness of our method. All the implementations are based on RIOT OS [2].

Due to the 15 BLE nodes, we need 15 micro USB cables to program and power them. To collect and plot data, these 15 nodes must be connected to our laptop. Hence, we need a USB hub with minimum 15 ports. We collect and plot all the data through a Python script. To offer a decent demo experience to our audiences, we expect an external monitor to show the results and dynamic figures. To power all the devices, i.e., USB hub, monitor, and laptop, we also need a power socket.

For the development boards (Adafruit Feather nRF52840 Sense), micro USB cables, and USB hub, we can prepare and bring them to the conference. However, regarding the monitor and power socket, we expect the conference to offer them.

In Figure 2, we show the demo setup in our office environment. All the 15 BLE nodes are connected to the USB hub. We manually distinguish the 15 BLE nodes into two groups, one group with 5 BLE nodes and the other one with 10 BLE nodes. These two groups of nodes are two BLE multi-hop networks. They share the same parameters and settings, except the implementation of our method on the 10-node BLE network. We program all the nodes to periodically print out formatted data, e.g., every 1 s. The connection between our laptop and USB hub allows the data transmission from all the BLE nodes to our laptop. The laptop runs a Python script to monitor, collect, and process the data from the two networks. Then, it plots the data and updates the plot every second on the external monitor, with a 1-h window.

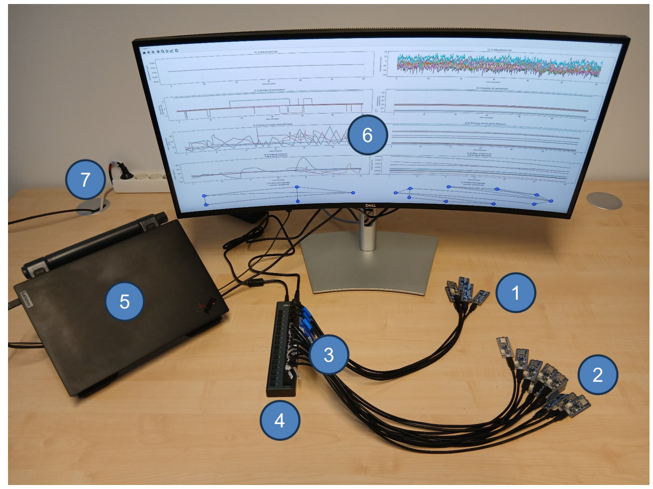

Figure 2: Demo setup in the office environment. The numbers and their corresponding devices are: 5-node BLE network without implementing our method ①, 10-node BLE network with the implementation of our method ②, micro USB cables ③, USB hub ④, laptop that runs the Python script ⑤, external monitor ⑥, and power socket ⑦.

During the demo, we would like to give the following information to our audiences. First, BLE has the capability of establishing a multi-hop network, but cannot maintain the network in a stable status. The instability is from the clock drift among the BLE nodes, and BLE specification does not offer any mechanisms to resolve the problem. As a result, after establishing a BLE multi-hop network, we can see frequent connection loss and topology change. We will show and explain all these through the 5-node network. Second, we explain the principle and implementation of our method to mitigate the impact of clock drift on BLE multi-hop networks. The core idea is that we introduce a proportional integral (PI) controller on each BLE node. The PI controller monitors the drifts among all the connections, and introduces minor adjustments accordingly to fine-tune them. Third, we compare the 10-node BLE network with the 5-node one. Our results should illustrate that our method can prevent connection loss and topology change in BLE multi-hop networks. The phenomenon is that we can see frequent disconnection, reconnection, and unstable topology even when there are only 5 BLE nodes in the network. However, for the 10-node BLE network, we will not see any disconnection or reconnection. Besides, the topology will stay consistent throughout the whole demo. We also use other parameters shown in Figure 3 to illustrate the mitigation of clock drift.

4 Demo Results

We demonstrate a comparison between two BLE multi-hop networks. The two multi-hop networks have different sizes, e.g., 5 nodes in the multi-hop network without our method and 10 nodes with our method. This is to illustrate the effectiveness of our solution. Except the implementation of our method, all the other parameters and settings are the same, such as connection intervals and data rates.

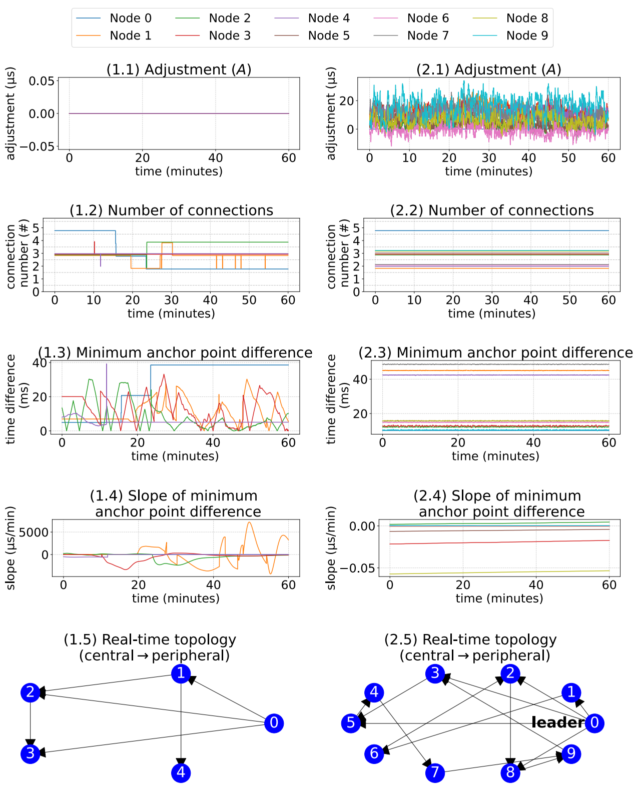

Figure 3: The expected results for the demo. The left column plots the real-time data of the 5-node network. The right column plots the real-time data of the 10-node network. Both use a time window of 60 minutes.

For the 5-node network, we expect unstable BLE connections, shown as disconnection from time to time (see left column of Figure 3). The varying number of connections shows the frequent disconnection and reconnection. We observe minimum anchor point difference changing on each BLE node, meaning relative drift among connections. We also keep an eye on the slope of the minimum anchor point difference to see its trend by time. The real-time topology plot can show the topology change due to disconnection and reconnection.

For the 10-node network, we expect to see a stable BLE multi-hop network, shown as no connection loss at all (see right column of Figure 3). Our method introduces adjustment (A) to the anchor point of connections. The number of connections stays constant on each BLE node. The minimum anchor point difference on each BLE node fluctuates around its initial value, meaning our method prevents connections from drifting. The slope of the minimum anchor point difference converges to 0 on each node, which suggests that the time lengths among connections stay constant from the long-term point of view. As a result, a stable BLE multi-hop network is realized, without modifying the timer/clock of hardware or software (IoT OS). We have also tested the method on larger networks, such 20 nodes, and the method still functions. However, we notice some performance drop of our method, such as a larger slope of minimum anchor point difference.

5 Discussion

There is no existing solutions in BLE standards to cope with the clock drift in BLE multi-hop networks. The current time calibration mechanism from BLE standards only targets the time synchronization between two nodes, i.e., a central and a peripheral. As a result, BLE cannot manage multi-hop networks properly. State-of-the-art solutions highly rely on new time synchronization mechanisms [1, 3]. However, they either introduce extra communication load or changes to IoT clock/timer systems, or even both.

During the development phase, we tried different setups, such as network size and connection number on each node. We notice that the increase of network size and connection number on each node negatively impacts the effectiveness of our method. For instance, the slope of minimum anchor point difference becomes larger and converges more slowly. This can be from the additional coordination overhead due to managing more concurrent connections.

6 Future Work

First, we aim to increase the connection number allowed on each BLE node. Second, we would like to investigate the possibility to remove the leader. Third, the uniform distribution of connections on each node can be interesting.

References

[1] Farzad Asgarian and Khalil Najafi. 2022. BlueSync: Time Synchronization in Bluetooth Low Energy With Energy-Efficient Calculations. *IEEE Internet of Things Journal* 9, 11 (June 2022), 8633–8645. doi:10.1109/JIOT.2021.3116921

[2] Emmanuel Baccelli, Cenk Gündoğan, Oliver Hahm, Peter Kietzmann, Martine S. Lenders, Hauke Petersen, Kaspar Schleiser, Thomas C. Schmidt, and Matthias Wählisch. 2018. RIOT: An Open Source Operating System for Low-End Embedded Devices in the IoT. *IEEE Internet of Things Journal* 5, 6 (Dec. 2018), 4428–4440. doi:10.1109/JIOT.2018.2815038

[3] Ryotaro Ohara, Shintaro Izumi, Shoya Imanaka, Tetsuo Yamamura, Ishii Toru, and Hiroshi Kawaguchi. 2024. MicroSync: Sub-Micro Second Accuracy Wireless Time Synchronization Service. *IEEE Access* 12 (2024), 124478–124494. doi:10.1109/ACCESS.2024.3446668

[4] Bozheng Pang, Tim Claeys, Hans Hallez, and Jeroen Boydens. 2023. Modeling the Trade-off between Throughput and Reliability in a Bluetooth Low Energy Connection. In *Proceedings of the 2023 International Conference on embedded Wireless Systems and Networks (EWSN '23)*. International Conference on Embedded Wireless Systems and Networks (EWSN), Rende, Italy, 32–39.

[5] Bozheng Pang, Tim Claeys, Davy Pissoort, Hans Hallez, and Jeroen Boydens. 2020. A Study on the Impact of the Number of Devices on Communication Interference in Bluetooth Low Energy. In *2020 XXIX International Scientific Conference Electronics (ET)*. 1–4. doi:10.1109/ET50336.2020.9238240

[6] Hauke Petersen, Thomas C. Schmidt, and Matthias Wählisch. 2021. Mind the gap: multi-hop IPv6 over BLE in the IoT. In *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies (CoNEXT '21)*. Association for Computing Machinery, New York, NY, USA, 382–396. doi:10.1145/3485983.3494847

[7] Bluetooth SIG. 2025. Bluetooth Core Specification v6.2. *Bluetooth SIG* (2025), 3897.

[8] Yize Zhao, Lin Wang, Zijuan Liu, Yifan Xu, Fan Dang, Xu Wang, Haitian Zhao, and Xin Miao. 2024. A Comprehensive Evaluation of Bluetooth Low Energy Mesh. In *2024 IEEE 30th International Conference on Parallel and Distributed Systems (ICPADS)*. 544–551. doi:10.1109/ICPADS63350.2024.00077 ISSN: 2690-5965.