Concurrent Transmissions for Multi-Hop Bluetooth 5

Beshr Al Nahas[1], Simon Duquennoy[2], Olaf Landsiedel[3,1]
[1] Chalmers University of Technology, Sweden
[2] RISE SICS, Sweden
[3] Kiel University, Germany
{beshr, olafl}@chalmers.se, simon.duquennoy@ri.se, ol@informatik.uni-kiel.de

Abstract

Bluetooth is an omnipresent communication technology, available on billions of connected devices today. While it has been traditionally limited to peer-to-peer and star network topology, the recent Bluetooth 5 standard introduces new operating modes to allow for increased reliability and Bluetooth Mesh supports multi-hop networking based on message flooding. In this paper, we present BlueFlood. It adapts concurrent transmissions, as introduced by Glossy, to Bluetooth. The result is fast and efficient network-wide data dissemination in multi-hop Bluetooth networks. Moreover, we show that BlueFlood floods can be reliably received by off-the-shelf Bluetooth devices such as smartphones, opening new applications of concurrent transmissions and a seamless integration with existing technologies.

We present an in-depth experimental feasibility study of concurrent transmissions over Bluetooth PHY in a controlled environment. Further, we build a small-scale testbed where we evaluate BlueFlood in real-world settings of a residential environment. We show that BlueFlood achieves 99% end-to-end delivery ratio in multi-hop networks with a duty cycle of 0.13% for 1-second intervals.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Wireless Communication

General Terms

Design, Experimentation, Measurement, Performance

Keywords

Constructive Interference, Synchronous Transmissions, Capture Effect, BLE, WSN, IoT

1 Introduction

Bluetooth is an omnipresent communication technology. In 2017, more than 3.6 Billion Bluetooth-enabled devices were sold and the overall installed-base of Bluetooth devices is estimated to be roughly 10 Billion [3]. This makes Bluetooth predominant in our modern, connected society.

While Bluetooth has been available for many years, the release of Bluetooth Low Energy (BLE) in 2010 brought significant improvements in terms of energy efficiency for Bluetooth. Today, many wireless peripherals for; *e.g.*, health, fitness and home automation use BLE as main communication technology. With the recent release of Bluetooth 5 and Bluetooth Mesh, the yearly growth of the deployment of Bluetooth devices is likely to further increase. With its new transmission modes, Bluetooth 5 aims to offer a performance in terms of reliability, range, and energy efficiency that is on-par with IEEE 802.15.4 [47].

In the past decade, the research community has designed a plethora of MAC, routing, and dissemination protocols for low-power wireless networking. However, the focus for networking in low-power wireless has been nearly exclusively on IEEE 802.15.4. For example, Glossy [14] made a breakthrough in low-power wireless in disseminating information at network-scale quickly and efficiently. It utilizes concurrent transmissions of tightly synchronized packets to realize flooding and synchronization services. As of today, Glossy is practically limited to 802.15.4 in the 2.4 GHz band and – to a smaller degree – ultra-wide band communication (UWB) [6, 10] and 802.15.4 in the sub-GHz band [9].

Concurrent transmissions for Bluetooth, however, have been overlooked until today. It is, for example, not shown whether the concepts of concurrent transmissions are applicable to Bluetooth. The key differences between the Bluetooth PHY and IEEE 802.15.4 in the 2.4 GHz band, are (i) the use of GFSK and O-QPSK modulation, respectively, (ii) the lack of DSSS in Bluetooth and (iii) the support of four data rates in Bluetooth: 125 Kbps, 500 Kbps, 1 Mbps, 2 Mbps versus 250 Kbps for 802.15.4. This design makes Bluetooth less sophisticated in terms of physical layer features when compared to IEEE 802.15.4. Moreover, analytic and experimental results indicate that the coding robustness provided by DSSS in 802.15.4 is essential to the reliability of Glossy [46, 28]. The recently adopted standard Bluetooth 5 provides additional encoding, but still operates with GFSK modulation and without DSSS.

We argue that adapting the concepts of concurrent transmissions to Bluetooth can open a variety of new application scenarios due to the ubiquitous availability of Bluetooth-enabled devices. In this paper, we evaluate concurrent trans-

International Conference on Embedded Wireless
Systems and Networks (EWSN) 2019
25–27 February, Beijing, China

ISBN: 978-0-9949886-3-8

missions on top of Bluetooth PHY and exploit them in BlueFlood to provide network-wide flooding. For example, in case of a fire in a building, we see the opportunity to use BlueFlood to disseminate a warning message with evacuation information as extended Bluetooth beacons. As we show in this paper, such a flood of Bluetooth beacons is received at low-latency and high reliability by, for example, off-the-shelf smartphones. Similarly, Bluetooth Mesh extensively builds on network-wide flooding of messages which can benefit from concurrent transmissions to improve energy efficiency and reliability while reducing latency.

Finally, while Glossy was originally implemented on TelosB hardware utilizing the MSP430 MCU and a CC2420 radio, we now have modern SoCs with integrated radios available. We show in this paper that these strongly simplify the design and implementation of protocols where transmissions need to be timed in the order of parts of a microsecond; *i.e.*, down to the individual ticks of the micro-controller, such as the case of concurrent transmissions.

Contributions This paper makes six key contributions:

1. We demonstrate the practical feasibility of concurrent transmissions on Bluetooth PHY.
2. We evaluate the performance trade-offs of the four different transmissions modes provided by Bluetooth 5 of 1 and 2 Mbps and coded long range with 500 and 125 Kbps, for concurrent transmissions.
3. We introduce BlueFlood: a multi-hop, low-power concurrent flooding protocol for Bluetooth PHY.
4. We demonstrate that BlueFlood is received by off-the-shelf receivers; *e.g.*, smartphones.
5. We illustrate how modern System-On-Chip (SoC) hardware simplifies the design of protocols based on concurrent transmissions.
6. We evaluate BlueFlood in a residential environment and show that BlueFlood achieves 99% end-to-end delivery ratio in multi-hop networks with a duty cycle of 0.13% for 1-second intervals. Moreover, we show the fragility of CT over Bluetooth and the associated practical challenges. BlueFlood is available as open source[1]. This includes the code, the experimental data and the scripts needed to reproduce our results.

Outline The remainder of this paper is structured as follows: we review the related technical background about low power communication, Bluetooth and concurrent transmissions in §*2*, then we evaluate the feasibility of concurrent transmissions over Bluetooth in §*3*. In §*4*, we introduce the design of BlueFlood, next we evaluate it in §*5*. Finally, we discuss the related state of the art research in §*6* and conclude in §*7*.

2 Background

In this section, we provide the necessary technical background on Bluetooth and concurrent transmissions. Also, we relate to essential state of the art on both modeling and utilizing concurrent transmissions in low-power wireless. With these we identify the key challenges for concurrent transmissions on Bluetooth PHY. Later, §*6* provides a deeper discussion of the state of the art in the broader field of concurrent transmissions.

2.1 Low-Power Wireless: 802.15.4 vs. BLE

Bluetooth Low Energy (BLE) and ZigBee/802.15.4 are today's widespread technologies for low-power wireless communication in the unlicensed 2.4 GHz spectrum. Each of them was initially designed for unique and distinct goals: While Bluetooth traditionally targets low-range single-hop communication with a bit-rate suitable for *e.g.*, wearable and multimedia applications, ZigBee targets longer ranges and reliable multi-hop communication with a lower bit-rate suitable for *e.g.*, home automation applications or industrial control. To this end, the IEEE 802.15.4-2015 standard introduces a physical layer that utilizes O-QPSK modulation and DSSS for forward error correction (FEC) in the 2.4 GHz band offering a bit-rate of 250 Kbps in 16 channels of 5 MHz and supports packet size of up to 127 bytes. On the other hand, both Bluetooth and 802.15.4 in sub-GHz use variants of FSK modulation, and both support *uncoded detection* for demodulation. BLE 4 uses GFSK and the latter uses 2-FSK — both modulation schemes represent bits 0 and 1 by using a $\pm F$ frequency shift from the central frequency. BLE 4 offers a bit-rate of 1 Mbps in 40 channels with a bandwidth of 2 MHz each without FEC and supports packets with PDU up to 39 bytes. Overall, the design choices of the narrower channels, a simpler modulation scheme and the lack of DSSS make Bluetooth the less robust communication scheme of the two. Next, we discuss how the recent Bluetooth 5 changes this.

2.2 Bluetooth 5

With the widespread availability of Bluetooth and an estimated number of 10 billion Bluetooth devices sold, there is an increasing interest to use Bluetooth beyond the originally targeted domain of low-range, single hop communication. For this, the recent Bluetooth 5 standard [47] introduces (i) new long-range communication modes and (ii) supports longer packets up to 255 bytes.

The physical layer of Bluetooth 5 supports four PHY modes: (i) two modes without forward error correction (FEC): a new, 2 Mbps mode in addition to the backward compatible 1 Mbps, and (ii) two new long range modes that utilize FEC driven by a convolutional code: 500 Kbps and 125 Kbps, with up to 4× longer range when compared to uncoded 1-Mbps. We note selected low-level details: (i) the different modes have different preamble lengths: one byte for 1 M, two bytes for 2 M and ten bytes for the coded modes 500 K and 125 K, (ii) the two coded modes 500 K and 125 K always transmit the header with FEC 1:8, and only afterwards changes the coding rate to FEC 1:2 for the 500 K mode, and (iii) all modes share a symbol rate of 1 M except for the 2 M mode. Table *1* summarizes the operation modes. When compared to 802.15.4, the physical layer of Bluetooth 5 still maintains the narrow channels of 2 MHz and does not employ DSSS. Nonetheless, the standard has the potential to be an enabler for IoT applications with a performance in terms of range, reliability, and energy-efficiency

[1] https://github.com/iot-chalmers/BlueFlood

comparable to 802.15.4.

2.3 Bluetooth Mesh

Bluetooth Mesh, part of the Bluetooth 4 standard, introduces multi-hop communication to Bluetooth: Bluetooth Mesh follows a publish/subscribe paradigm where messages are flooded in the network so that all subscribers can receive them. Thus, Bluetooth Mesh does not employ routing nor does it maintain paths in the network. To reduce the burden on battery-powered devices, forwarding of messages in a Bluetooth Mesh is commonly handled by mains-powered devices. In recent studies with always-on, *i.e.*, mains-powered, nodes as backbone, Bluetooth Mesh reaches a reliability of above 99% both in simulation [31] and experiments [39], and latencies of 200 milliseconds, in netowrks of up to 6 hops with payload of 16 bytes [39].

Because Bluetooth Mesh employs flooding, it differs strongly from established mesh and routing protocols in 802.15.4 such as CTP [17] or RPL [43]. We see the fact that Bluetooth Mesh is based on flooding is an additional motivation to evaluate the feasibly and performance of concurrent transmissions for network-wide flooding in Bluetooth 5.

2.4 Bluetooth Advertisements

Traditionally, Bluetooth targets single-hop communication. For this, it operates in two modes: advertisement mode and connected mode. In the advertisement mode, a Bluetooth device broadcasts short pieces of information. This is commonly used by low-power devices such as, for example, temperature sensors, to share their sensor readings, and localization beacons, to announce their presence and their physical location. Moreover, this mode is used to advertise the availability of a device so that other devices can connect to it. The second mode, the connected mode, establishes a connection between a master and a slave. Here, a master and slave communicate in time-synchronized *connection events*. In this paper, we focus on Bluetooth advertisements and refer the reader to Bluetooth core specifications [7] for details about each mode.

In this paper we use non-connectable beacons for lightweight flooding while staying compatible with off-the-shelf devices. Bluetooth 5 extends this further by allowing a packet up to 255 bytes versus the legacy 39 bytes limit. Moreover, it allows advertising on any of the 40 channels instead of limiting it to three channels as in previous Bluetooth versions. While the Bluetooth specifications do not define the beacon payload format, there are several industrial standards, with two main formats [45]: (i) Apple's iBeacon (shown in Figure *1*) and the open source alternative AltBeacon (by Radius Networks), which carries a Universally Unique Identifier (UUID) that defines an event or a geographical location ID with minor and major fields which allow to define sub-events or other attributes, and (ii) Google's Eddystone, which supports URL and telemetry beaconing in addition to UID beaconing (similar to iBeacons).

2.5 Concurrent Transmissions and Capture

In this section, we discuss concurrent transmissions (CT) in a generic context that applies to both IEEE 802.15.4 (ZigBee PHY) and Bluetooth 5 PHY.

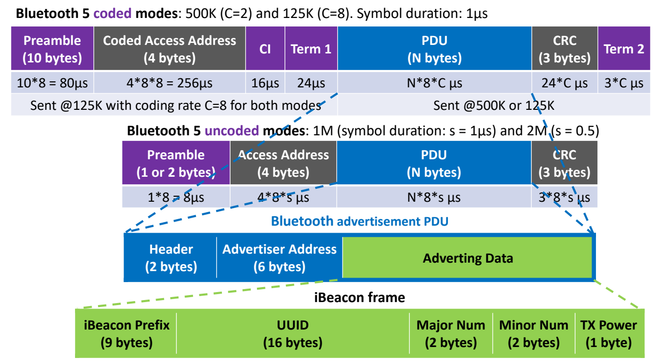


Figure 1. Bluetooth packet structure for the coded and uncoded modes. *Bluetooth advertisements formats are defined in defacto industrial standards such as iBeacon.*

Definition In concurrent transmissions, multiple nodes synchronously transmit the data they want to share. Nodes overhearing the concurrent transmissions receive one of them with high probability, due to the capture effect [27].

Factors Affecting the Performance of CT In summary, the performance and practical feasibility of CT depends on four factors [46]: (i) the time delta between the two packets, and (ii) the received signal strength RSS delta. (iii) The choices of the radio technology (modulation and encoding), and (iv) the option to send identical payload or not determines the range of the first two parameters for a successful reception and the final robustness of the CT link.

In practice, the carrier frequencies of the different transmitters are never exactly equal. As a result, the concurrent transmission of the *same data* leads to a *beating* radio signal, where the signal magnitude alternates between peaks and valleys instead of being uniform. These variations in frequency and phase distort the signal; thus, CT might become destructive if the signal distortion is severe. It shall be noted that the radios transmit preamble bytes to synchronize the frequency and phase of the receiver to that of the transmitter. In case of CT, the receiver would synchronize to the effective sum of the different preambles. On the other hand, the concurrent transmission of *different data* causes destructive interference to the signal that is only recoverable when one transmitter signal has a RSS delta over the sum of the other CT as long as they are received within the duration of the signal preamble. 802.15.4 radios in the 2.4 GHz band utilize DSSS, where bits are encoded redundantly into *chips* with a 1:8 FEC redundancy; *i.e.*, 2 M chips/sec encode a 250 Kbps data stream. This encoding helps recovering bits from the distorted signal in both cases of CT of the same and different data. Typically, in 802.15.4, the radio receives the stronger one of the concurrent transmissions if its signal is 3 dBm stronger, the so-called *co-channel rejection*, if they are synchronized within the preamble of 5 bytes: 160 μs [26]. However, in the case of CT of the same data over 802.15.4, if the nodes transmit within 0.5 μs, then no signal strength delta is necessary [14]. On the other hand, radio standards that lack FEC mechanisms experience challenges when it comes to receiving CT as we explain in §*3*.

Table 1. Bluetooth 5: PHY parameters and modes. *Note that the two coded modes 500 K and 125 K use the 1 M PHY mode beneath. τ stands for period.*

Bitrate	*Symbol rate*	*Symbol τ*	*bit τ*	*FEC*	*Preamble*
[*bps*]	[Symbol/s]	[μs]	[μs]	ratio	[byte]
2 M	2 M	0.5	0.5	-	2
1 M	1 M	1	1	-	1
500 K	1 M	1	2	1:2	10
125 K	1 M	1	8	1:8	10

2.6 Glossy

Glossy [14] is a flooding protocol for network-wide synchronization and data dissemination. It established the design principle of concurrent transmissions of the same data in low-power wireless networks that are based on the IEEE 802.15.4 standard as it proved to be a highly reliable and efficient protocol. Glossy operates in rounds, with a designated node, the initiator, that starts the concurrent flooding. Nodes hearing the transmission synchronize to the network and join the flooding wave by repeating the packet. The transmissions are tightly synchronized in order to achieve non-destructive CT. Every node alternates between reception and transmission and repeat this multiple times to spread the information and achieve one-to-many data dissemination from the initiator to the rest of the network.

3 Feasibility of CT over Bluetooth

After providing the required background on both Bluetooth and concurrent transmissions, we set out to analyze and evaluate whether concurrent transmissions are practical on the Bluetooth physical layer. We begin by outlining the lessons-learned from the state-of-the-art, discuss why CT shall work, identify the challenges and discuss how they materialize for Bluetooth, then we proceed to our experimental evaluation of this feasibility study.

3.1 CT Opportunities and Challenges

In this section, we outline why CT should fundamentally work over Bluetooth before discussing the practical challenges and limitations of achieving successful CT over Bluetooth. While recent studies by Carlson *et al.* [9], Wilhelm *et al.* [46] and Liao *et al.* [28] [29], among others, discuss CT over 15.4 both in 2.4 GHz and in sub-GHz, we draw lessons that are applicable to Bluetooth due to the similarities in the modulation employed in Bluetooth and 802.15.4 sub-GHz as discussed in §*2.1*. We need to differentiate between the following cases: same versus different data, and with versus without FEC. In the following discussions we start by focusing on the *same data* case without FEC, then we discuss the *different data* case and discuss the benefits of having FEC.

3.1.1 Opportunities in the Bluetooth Baseband

Bluetooth uses Gaussian-filtered Frequency Shift Keying (GFSK). We can describe it with a non-distorting simplification: in the base-band frequency spectrum of the modulated signal, bits 0 and 1 are shown as $\pm F$ frequency shifts from the central frequency. In the case of *ideal* synchronous concurrent transmissions (*i.e.*, no time, frequency or phase delta in the carrier band) of the same data, the two signals would overlay perfectly and a receiver would not notice a difference from the case of a single transmitter. On the other hand, with different data, the sum of the two signals of the two different bits need to be distinguishable from the center frequency, and lay on either $\pm F$ sides; *i.e.*, one signal needs to be sufficiently stronger than the other.

Nevertheless, real-life concurrent transmissions are not as simple, as different transmitters have slightly different frequencies, drift independently and signals sum-up at the receiver with different phases. We discuss these challenges in the next section.

3.1.2 Bluetooth CT Challenges

Per the study of Liao *et al.* [29] on CT over 802.15.4 in the sub-GHz band, the most critical operation zone for CT is when both transmissions reach the receiver with the same power; *i.e.*, zero power delta. The authors argue, that in this case the timing offset needs to be smaller than 1 μs. Wilhelm *et al.* [46] suggest that the combination of the carrier-phase offset and the timing offset is detrimental to the reception of CT. Their paper gives bounds of the tolerable timing offset to be half of the symbol period; *i.e.*, $\tau/2$. For Bluetooth, this translates to 0.25 μs for the 2 Mbps mode and 0.5 μs for the other modes (as they share the same symbol rate). On the other hand, the tolerable carrier phase offset is estimated to be 0.4π. While we cannot control the phase offset in software with off-the-shelf radios, we can synchronize the transmissions to be within the bounds noted above. In addition, the signal preamble helps the receiver to synchronize the phase offset. Thereby a longer preamble can help a receiver to lock on a specific phase-offset and thereby improve the reception of this particular transmissions.

In the case of *different data* on the other hand, and under similar conditions, there needs to be a power delta of about 10 dB to have a packet reception rate of $20-30\%$, especially when not protected by FEC [46].

Overall, these studies demonstrate a degraded receiver sensitivity and subsequently the declined reliability with CT when done over uncoded, non-DSSS communication, *i.e.*, without the protection of FEC. On the other hand, the studies indicate that use of FEC improves reliability and relaxes the conditions for successful reception as well.

3.1.3 Summary

Based on the existing models and analysis, we can summarize the status of CT over Bluetooth in the following: (i) since Bluetooth employs non-DSSS communication, it is expected to suffer under CT when compared to, for example, 802.15.4. (ii) The timing offset shall be kept under 0.25 μs for the 2 Mbps mode and 0.5 μs for the other modes, (iii) the phase offset shall be below $\pm 0.4\pi$, which we do not control directly. However, we argue that we can potentially increase the robustness by using Bluetooth modes with longer preambles and thereby improve the synchronization of the receiver onto a specific phase-offset of a signal. (iv) The capture of CT of different data is not possible without a major signal power delta; especially without FEC (see also §*2.5*), and (v) the use of FEC is expected to improve the performance, but obviously incurs a non-trivial overhead of 1:2 or 1:8 per the two modes 500 K and 125 K, respectively. Next, we experimentally evaluate CT performance over Bluetooth.

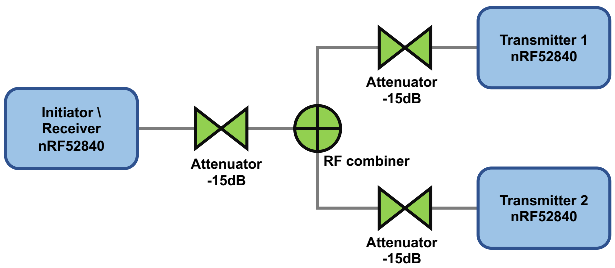


Figure 2. Feasibility of CT over Bluetooth PHY: *micro-evaluation setup of two concurrent transmitters and one receiver connected via coaxial cables and attenuators through their antenna connectors.*

3.2 CT over Bluetooth: Experimental Study

Before devising and implementing a full system for concurrent transmissions in Bluetooth, we begin with a series of controlled and reproducible experiments, which have the following goals: to demonstrate that concurrent transmissions in Bluetooth are feasible, to evaluate their reliability, and to derive first insights on how the different Bluetooth modes ranging from coded 125 Kbps to non-coded 2 Mbps impact performance.

Objectives In this section, we show the feasibility of CT over Bluetooth by answering three questions: (i) How reliable is a Bluetooth CT link depending on the difference in the received signal strength of two concurrent transmitters? (ii) How does timing accuracy affect the reliability of CT? (iii) How does CT in the Bluetooth PHY perform when sending same vs. different data? While there are multiple papers that give analytic answers; *e.g.*, [28] and [46] (see also §*3.1*), there is a gap of experimental evaluation that we seek to fill.

Setup To ensure a controlled communication channel free from external interference and to enable reproducible results, all nodes in this feasibility study are connected via coaxial cables and attenuators through their antenna connectors, as in Figure *2*. For simplicity and without loss of generality, we focus in our feasibility study on the case of two concurrent transmitters and one receiver. This setup ensures symmetry and is common in related work; *e.g.*, [46].

We evaluate the performance of concurrent transmissions over Bluetooth using a setup of three nodes equipped with nRF52840 SoCs (see Table 2) capable of Bluetooth 5 communication: (i) an initiator node that starts periodic rounds by transmitting a packet, then switches to receive mode, and (ii) two CT nodes that transmit concurrently after hearing the first packet. We send iBeacon packets of 38 bytes. We test both cases of sending the same data and different data. Each experiment is run until at least 2000 packets are sent.

3.2.1 *CT Performance vs. TX Power Delta*

We fix the transmission power of one CT node to 0 dB and vary the transmission power of the second to sweep all the factory calibrated TX power settings: [-40, -20, -16, -8, -4, 0, 2, 3, 4, 5, 6, 7, 8] dB. We cross check at the receiver (initiator) and confirm that the received signals have a matching power delta as the configuration. We repeat this experiment on the four modes of Bluetooth 5; namely, 2 M, 1 M, 500 K and 125 K.

Figure *3a* shows the results of the experiments. We can summarize the results in the following takeaways:

- The first take away of this experiment is that CT of the same data is feasible over all the Bluetooth 5 modes regardless of the power delta.
- Secondly, while the long range mode 125 Kbps with FEC 1:8 has the best performance, the other modes perform well once there is a difference in the CT signal strength.
- Thirdly, in the case of different data, capture is only feasible when there is a power delta greater than or equal to 8 dB.

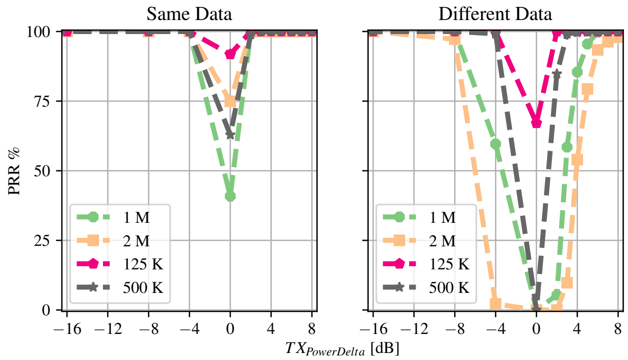
For these reasons, we base our design on CT of the same data. It should be noted, however, that the performance of concurrent transmissions over Bluetooth PHY is considerably weaker than over 802.15.4 (as reported in *e.g.*, [14]) and it is greatly affected by the setup. Nevertheless, we show in this paper that we can utilize it to build efficient end-to-end flooding.

3.2.2 *CT Performance vs. TX Time Delta*

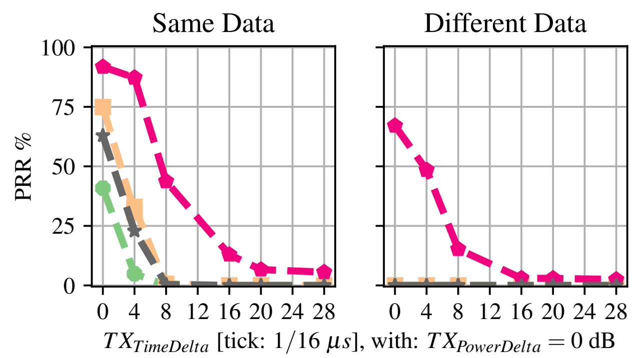
We inject a constant delay in the transmission time of one CT node, and vary it to [0, 4, ... 28] clock ticks; *i.e.*, [0, 0.25, ... 1.75] μs, and fix the transmission power of both nodes. Note that one clock tick is $1/16 = 0.0625\,\mu s$, and the symbol period of the 1 Mbps PHY is equal to 1 μs, which is the same for the Bluetooth modes (1 M, 500 K, 125 K), while the 2 M mode has a symbol period of 0.5 μs. We repeat the test with different TX power deltas (as we did in the previous section: fix one node to TX power of 0 dB and change the other) to study the combined effect of signal power and transmission delay.

Figure *3b*– Figure *3e* show the results of the experiments. We distinguish the following phenomena:

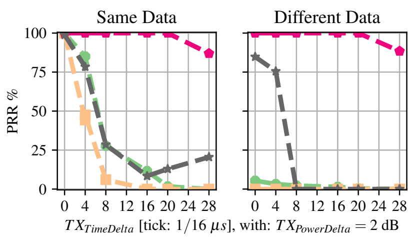
- *Destructive interference: 0-2 dB*. The first take away of this experiment is that the performance of CT of the same data drops significantly with TX time delta when we operate in the 0-2 dB power delta zone as Figures *3b–3c* show. The reason is that the two signals interfere destructively when the symbols are misaligned.
- *Power capture for the coded 125 K mode: 0-2 dB*. We notice in the case of 0 dB tx power and different data that only the high fidelity 125 K mode survives up to time delta of 8 ticks, which equals half of PHY symbol. On the other hand, having as little as 2 dB power makes the time delta effect on performance insignificant for the 125 K mode.
- *Slightly destructive: 4 dB and half a symbol delay*. We notice that the CT performance drops with the time delta up to 8 ticks (half a symbol for 500 K and 1 M), to start recovering partially after crossing the symbol boundary. The 2 M mode exhibit a performance drop similar to 1 M ($\approx 60\%$) but at the 4 ticks (half a symbol at 2 M), but does not recover, as Figure *3d* shows.
- *Power capture at 8 dB*. We notice that the time delta effect on CT performance is almost negligible except for the 2 M mode where we see a drop of PRR to 80-



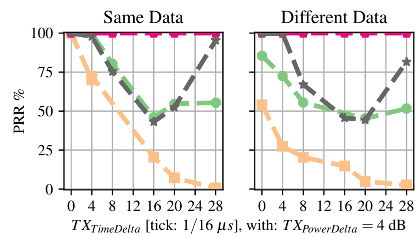
(a) TX power delta effect: *CT of the same data is feasible even when the two signals have the same strength. However, the capture of different data suffers greatly when the different packets have a power delta less than 8 dB.*



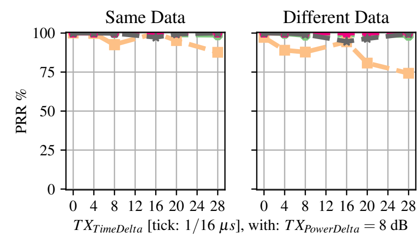
(b) TX time delta effect without power delta: *CT performance drops significantly when time delta is introduced. The x-axis is in ticks with* $1/16$ *μs granularity.*



(c) TX time delta effect with 2 dB power delta: *CT performance drops significantly when time delta is introduced.*



(d) TX time delta effect with 4 dB power delta: *CT performance drops significantly when time delta is introduced, then recovers when the delay crosses the symbol boundaries (16 ticks for 1 M) as it becomes in the power capture zone.*



(e) TX time delta effect with 8 dB power delta: *CT performance is penalized lightly when time delta is introduced as it is operating in the power capture zone.*

Figure 3. Micro-evaluation of CT over Bluetooth PHY: *effect of power delta and time delta when transmitting identical or independent payloads.*

90%. Thus, we conclude that at this power difference we mainly have capture.

3.3 Conclusion

Our main conclusion from the analysis and experiments is that CT of the same data over Bluetooth is feasible for all Bluetooth 5 modes; even when we have as little as 2 dB power delta and it can tolerate a time delta of a couple of ticks while keeping a good link quality. On the other hand, CT of different data needs to have a relatively high (8 dB) signal power difference in order to work. Therefore, we focus on data dissemination in this paper, and utilize CT of the same data to build BlueFlood: a reliable end-to-end flooding protocol, as we show next.

We would like to highlight, the results of experimental study and analytic models discussed in §*3.1.2* differ slightly: For example, our results indicate Bluetooth CT cannot tolerate more than $\tau/4 = 0.25$ μs time delta as opposed to the expected $\tau/2 = 0.5$ μs in the case of 0 dB power delta. Moreover, we see that CT of different data is successful with 8 *dB* Tx power delta as opposed the the expected 10 *dB* for uncoded modes (1-2 Mbps). We do not focus in this paper on obtaining an accurate theoretical model of CT over Bluetooth, but rather to show it is a viable design choice and to utilize it.

4 BlueFlood

In this section, we motivate the use of concurrent flooding and how we tackle its challenges over Bluetooth, then we introduce the design of BlueFlood.

Motivation We seek to design a low-power protocol for multi-hop data dissemination that can be received with unmodified smart devices. Thus, in BlueFlood, a backbone of BlueFlood enabled devices flood Bluetooth-compliant advertisements through concurrent transmissions, which are then received by off-the-shelf Bluetooth enabled devices. Based on our insights from the feasibility study of concurrent transmissions in Bluetooth, see §*3*, we adopt a design that borrows from Glossy and other flooding protocols.

Challenges and Solutions As explained in §*3.1*, concurrent transmissions are challenging over Bluetooth. Mainly, (i) the concurrent transmissions need to be synchronized down to 250 ns, and (ii) the CT links are fragile under the near zero power delta condition. However, since the link quality stays above 30% in the worst case in Figure *3a*, we argue that CT stays a viable strategy. On the positive side, the frequency diversity over 40 channels in Bluetooth helps surviving the external interference. Moreover, the various Bluetooth 5 modes give an interesting reliability–energy trade-off and widen the design space. Plus, the modern SoCs simplify the realization of the required tight synchronization, as we show later.

Overview We build BlueFlood, a synchronous flooding protocol that utilizes CT of the same data, as depicted in Figure *4*. We take inspiration from Glossy and A^2 [4] and design our protocol to be a round-based and time-slotted protocol.

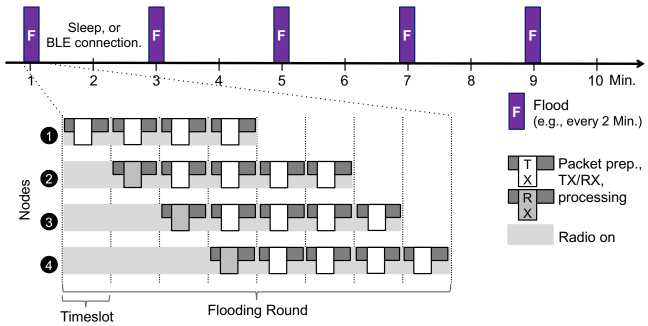


Figure 4. Overview of BlueFlood operation: *synchronous flooding that utilizes CT with RX / 4 TX transmission policy.*

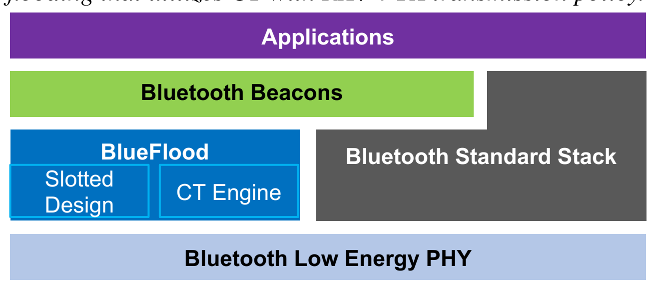


Figure 5. System architecture: *BlueFlood operates over the Bluetooth PHY, but it is transparent to the application which interfaces with standard Bluetooth beacons.*

Thus, just like in Glossy, Chaos and A^2, we schedule individual communication rounds on a network-wide scale. In the beginning of a round, all nodes wake up aiming to receive. A round is further split into time slots in which nodes either transmit, listen or sleep, according to a so called transmission policy.

From a system integration perspective, we design BlueFlood to be transparent to the application. In our example, the applications interact with a standard Bluetooth beacon library without having to know about the existence of BlueFlood, see Figure 5. As a result, BlueFlood distributes Bluetooth beacons on network-wide scale instead of the traditional on-hop announcements, enabling the application scenarios discussed in §*1*.

Next, we discuss the logical components of the protocol: time-slotted design, synchronization, transmission policy and frequency agility. Later, we discuss the design simplifications on modern SoCs.

4.1 Design Elements

In this section we discuss the design elements of BlueFlood. We take inspirations from Glossy, A^2 and the winners of the EWSN dependability competition in the years 2016 [38], 2017 [30], and 2018 [2].

4.1.1 Time-Slotted Design

Each slot fits one packet transmission/reception and processing, as shown in Figure 6. Within each slot, a node transmits, receives or sleeps according to the selected transmission policy. The default transmission policy in BlueFlood is to concurrently transmit a packet N times, *i.e.*, in the N slots following the reception of a packet, before completing the round and entering a deep-sleep mode until the beginning of the next round.

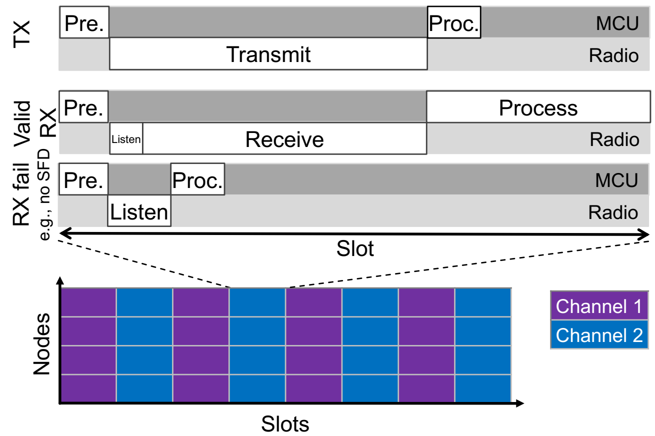


Figure 6. Overview of BlueFlood timeslot: *the timeslot accommodates one packet transmission or reception and handling, and each node hops the channel every timeslot.*

Power-saving To save power, every node turns the radio off as soon as the transmission or reception has ended or in case it fails to detect a valid packet at the beginning of the slot. The combination of CT with these simple power-saving techniques allows BlueFlood to provide a backbone of energy-efficient flooding devices. This is in contrast to Bluetooth Mesh, which restricts the forwarding task to mains-powered devices.

4.1.2 Frequency Agility

Glossy and related CT approaches see their performance degrade in presence of interference [15, 18]. We address this by employing Bluetooth frequency agility over the 40 available channels. Thus, in BlueFlood, nodes switch to a new channel to transmit or receive in each timeslot following a network-wide schedule. The round and slot numbers are used to index this hopping sequence. Once the node is synchronized, it has the same view of the slot and round numbers as the rest of the network; thus, it does not need to start each round on the same channel. This is similar to the channel-hopping of TSCH [20], Bluetooth [7] and has proven its robustness even under strong interference in the EWSN dependability competitions [30, 1, 5].

4.1.3 Synchronization

A key requirement is to keep the nodes tightly synchronized for a complete round within the bounds of 250 ns to successfully achieve CT. We merely require each node to receive a single valid packet during each round, which we then use for the per-round synchronization based on the radio-registered timestamp.

Scanning for Networks When a node wants to join the network, it listens on one frequency for $2 \cdot N$ periods, where N is the number of channels. Until it receives a valid packet, it hops the channel and repeats. Upon receiving a valid packet, it uses the slot number to synchronize to the beginning of the round.

Re-synchronization If a node does not receive a packet for X rounds, it assumes it lost the synchronization. Subsequently, it switches to the scanning mode.

4.1.4 Transmission Policy

Since we only require one valid packet per round to keep the synchronization, we utilize a transmission policy that follows the pattern: one valid RX, then N consecutive TX; *i.e.*, we wait for the first valid packet then transmit N times in a raw. This has a lower overhead of $N+1$ slots instead of $2*N$ for the original Glossy transmission policy (N times RX–TX) as BlueFlood eliminates the need to listen to repeated packets; thus, needs half the slots plus one to do N transmissions.

Power Budget With the aforementioned transmission policy, a node stays on for receive guard time ($Rx_{GuardTime}$) each slot until it receives the first valid packet, then it transmits N times. This strategy gives an average power budget P_{Avg} as a function of Tx and Rx power P_{Tx}, P_{Rx}, and average radio time R_{Avg} per node per successful round of:

$$P_{Avg} = (Avg_{HopCount} \times Rx_{GuardTime} + AirTime) \times P_{Rx} + N \times AirTime \times P_{Tx} \quad (1)$$

$$R_{Avg} = (Avg_{HopCount} \times Rx_{GuardTime}) + (N+1) \times AirTime \quad (2)$$

Bluetooth Modes Trade-off We discuss the trade-offs between different Bluetooth modes. The fastest mode 2 M has the shortest radio air-time. Thus, it has the lowest energy budget, but a lower reliability and shorter range than the coded 125 K mode which has up to $2-4\times$ longer range in comparison. In the same time, the coded 125 K mode has 1:8 FEC, which means $16-8\times$ longer air-time and higher energy budget than the 2 M and 1 M modes, respectively. In other words, N times TX in the 125 K mode costs as much as $16 \times N$ TX in the 2 M mode. In our evaluation in §*5*, we show how the different transmission modes impact the reliability of CT.

4.1.5 Bluetooth Compatibility and Packet Structure

To keep receive compatibility with off-the-shelf devices; *e.g.*, smartphones, we utilize the standard Bluetooth beacons; *e.g.*, non-connectable undirected advertisements, to flood the events. In particular, we use iBeacons (see Figure *1*) and override the major and minor numbers to designate the round and slot numbers, respectively.

4.2 Simplified Design on Modern SoCs

The modern SoCs integrate the MCU and the radio and provide a RAM-mapped packet buffer. Moreover, some provide configurable triggering of peripherals based on HW events to eliminate SW delays of processing interrupts. For example, on the nRF51 and 52 series, it is possible to wire the timer events to the radio and even to control the radio based on radio-generated events without the need for SW interaction in between [32]. Moreover, it sources all the peripherals from a divider of the CPU clock to achieve synchronous HW events. The main clock is a high resolution crystal of up to 64 MHz that sources the peripherals with 16 MHz.

In BlueFlood, we utilize both the direct wiring of events and the high resolution clock to strongly simplify our design and implementation when compared to Glossy. Practically, it allows us to avoid many of the SW complexities that original design of Glossy deals with to achieve the tight timing requirements on older-generation systems such as TelosB motes. For example, due to these limitations of the platform, the implementation of Glossy (i) relies on a radio-driven execution model, (ii) builds on a complex management of execution timing to minimize the packet transfer delay between the radio and the MCU, and (iii) relies on a Virtual High-resolution Timer (VHT) [37] for synchronization. In our experience, this makes Glossy and protocols building on Glossy such as, for example, LWB [15], Chaos [26], Crystal [21] hard to port to new platforms. We want to note that Glossy was later ported to several SoC platforms such as the CC2538 [19] and the subGHz CC430 SoC [16]. To our best knowledge, the synchronous transmission kernel of these ports stays complex due to the lack of the ability to wire hardware events on these platforms.

Table 2. Supported platforms details

SoC	*CPU*		*RAM*	*Firmware*	*Bluetooth*
nRF	*Cortex*	*Freq.*		*storage*	*modes*
Version		*[MHz]*	*[KB]*	*[KB]*	*[bps]*
51822	M0	16	16	128	1 M
52832	M4	64	64	512	1-2 M
52840	M4	64	64	512	125-500 K, 1-2 M

5 BlueFlood Evaluation

In this section we describe our implementation briefly and evaluate BlueFlood performance in a multihop mesh scenario.

5.1 Evaluation Setup

We present our BlueFlood implementation, the scenario, the metrics and the testbed used for evaluation.

Implementation We implement BlueFlood[2] in C for the Contiki OS [12] targeting Nordic Semiconductor nRF nodes equipped with an on-SoC Bluetooth radio. Table *2* lists the nodes specifications.

Scenario The evaluation scenario is a connection-less multihop dissemination. We use standard Bluetooth channels; as a result, we run BlueFlood with co-existing Bluetooth traffic and other sources such as WiFi as our testbed is deployed in an apartment. For the single channel experiments, we use the Bluetooth advertising channel 37. Unless otherwise mentioned, dissemination rounds repeat at a 0.5 s period. We run each experiment until we get about 1900-2000 rounds.

Configuration Depending on the Bluetooth mode, the slot size varies between 1 and 7 ms, with half of the slot length allocated to radio tx/rx. We configure the receive guard time to be half of that radio slot time. It shall be noted that our implementation is far from optimal as suggested by the difference in slot length and the minimal radio-on time shown in Table *3*.

[2] The code, raw data and parsing scripts are available at: `https://github.com/iot-chalmers/BlueFlood`

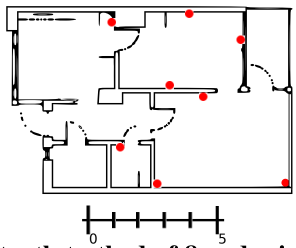


Figure 7. Bluetooth testbed of 8 nodes in a 60 m^2 apartment, where each circle represents a node.

Transmission Policy We use the transmission policy $NTx = 4$, but with a custom policy for the initiator. This custom policy is unnecessary for the dissemination functionality, and the sole goal is to be able to evaluate the reception of concurrent transmissions on the initiator side as well. We configure the initiator to alternate between sending and receiving until it gets a valid packet. Then, it stays on receive mode; thus, it is guaranteed to listen to concurrent transmissions.

Goals We evaluate BlueFlood performance on a testbed (described next) and test reception on a smartphone. Moreover, we evaluate how the different parameters affect the performance. Namely, we look at the effect of different transmission power, number of channels and packet size.

Testbed and Interference We deploy a testbed of 8 nodes of the type nRF52840 in a 60 m^2 residential apartment. This setup, although small, represents a typical household with 2-3 devices per room as depicted in Figure 7.
The testbed has interference from neighbouring homes as well as EMI from home and kitchen appliances such as a connected TV, microwave oven and fridge, etc. The 2.4 GHz spectrum is crowded with 16 coexisting WiFi networks covering all the 13 WiFi channels. The resulting mesh has 1-3 hops depending on the transmission power.

Metrics We focus on the following performance metrics:

- *Packet Reception Rate per slot (PRR per slot)*: to evaluate how reliable a link is. It is the ratio of the received valid packets over the number of receive slots. This gives an indication of the reliability of flooding using CT in a micro-level *i.e.*, per hop.
- *End-to-End Delivery Rate (E2E PDR)*: to evaluate how reliable a protocol is. We consider a round reliable as long as the node receives the disseminated value at least once;
- *Radio-on time*: is the total time the radio is active during a round, as a proxy for the energy consumed during a round;
- *Latency*: is the duration of a round until each node receives the disseminated value.

5.2 Transmission Power

In this section, we evaluate BlueFlood performance with varying transmission powers. We use all the 40 available channels to send iBeacon packets; *i.e.*, 38-byte packets with 30 bytes payload and 46 bytes = 368 symbols on air including PHY headers on the 1 Mbps PHY. We vary the TX power in [-20, -16, -12, -8, -4, 0] dB and repeat the experiments using the four Bluetooth modes.

Table 3. BlueFlood slot length needed to send a single iBeacon (38 bytes) for the different Bluetooth modes. *Air time: is the air time for the packet and represents the relative power budget for each mode. The radio slot is longer than the air time as we need to setup the radio and to compensate for the various SW delays.*

Mode	*PHY symbols*	*Air time*	*Radio slot*	*Guard*	*Slot*
[*bps*]	[symbol]	[millisecond]			
2 M	376	0.188	0.45	0.225	0.9
1 M	368	0.368	0.65	0.325	1.3
500 K	1134	1.134	1.4	0.7	2.8
125 K	3408	3.408	3.7	1.85	7.2

Table 4. The lists of used Bluetooth channels. *Note that the channel number indicates a consecutive frequency except for the channels 37, 38, 39 which are spread in the spectrum.*

# Channels	*Channel list*
3	37, 38, 39
9	0, 5, 10, 15, 20, 25, 37, 38, 39
18	0, 5, 10, 15, 20, 25, 37, 38, 39 3, 8, 13, 18, 23, 28, 33, 7, 17

Figure 8 summarizes the results. The end-to-end reliability stays over 85% for all modes and all transmission powers. We see an outlier for the 2 M mode at -12 dB with a sudden drop in reliability, which happened due to unfortunate – yet expected – noise on the channels as we are evaluating on standard Bluetooth channels in an apartment. For transmission powers greater than 0 dB the topology essentially becomes a single hop star network with regard to the initiator.

The -16 dB configuration is particularly interesting, as it leads to a 1.5 hops network; *i.e.*, it takes 1.5 slots to get the packet. In these settings, the average links quality is $> 60\%$ (as shown in the PRR Figure 8), while the end-to-end reliability is greater than 99%. In the same time, the 2 M mode offers a 16 times less radio-on time and about 7 times less latency as compared to the 125 K mode which has close to 99.9% PDR, but up to 10 ms in latency.

Overall, BlueFlood offers an attractive low-latency power-saving alternative – something that Bluetooth Mesh can fundamentally not achieve, as relay nodes must be always-on.

Estimated Duty-cycle For the 2 Mbps mode, we use Equation 2 with $NTx = 4$, 1.5 hops on average and 0.188 ms and 0.225 ms air time and guard time, respectively to get $R_{Avg} = 1.2775$ ms average radio time per node per round and about 1.5 ms of latency. For rounds that repeat every second, this represents an average radio duty cycle of $D_C = 0.13\%$.

5.3 Channel Hopping

In this section, we evaluate BlueFlood performance with different number of channels. We employ a pseudo-random hopping sequence of length 128. We send iBeacon packets with -16 dB TX power with $NTx = 4$. We vary the number of channels in [1, 3, 9, 18, 40] and repeat the experiments using

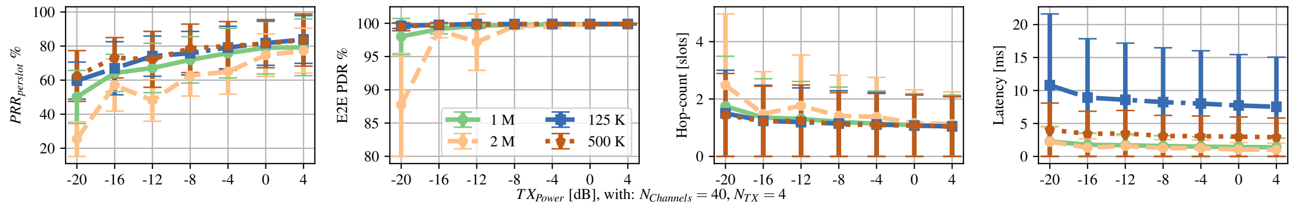


Figure 8. BlueFlood dissemination at different transmission power, over 40 channels: *while all modes have a good reliability > 97% for Tx powers -16 dB and higher, the 2 Mbps mode is particularly interesting as it is 16 times faster and thus uses 16 times less radio-on time than 125 Kbps.*

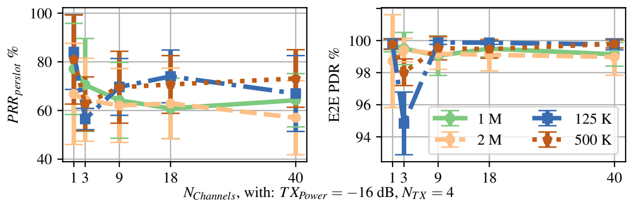


Figure 9. BlueFlood dissemination with different number of channels: *more channels allow for a robust operation, but in the same time, using all the available 40 channels means using more polluted channels.*

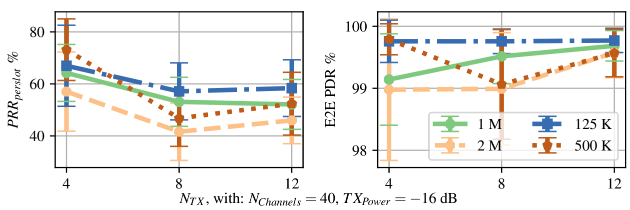


Figure 10. BlueFlood performance with longer repetition bursts. *Link reliability drops with longer transmission bursts due to the increased transmissions density and the end-to-end performance improves minimally as a result.*

the four Bluetooth modes. The channels we use in every case are spread over the spectrum as shown in Table *4*

Figure *9* shows the results. We notice the large standard deviation when using only one channel. This indicates varying reliability both for links and end-to-end. The deviation decreases and reliability increases when using up to 18 channels, but decreases slightly again when using all 40 channels. We attribute this simply to the fact that the 40-channel case includes poor channels that happened not to be present in the 18-channel case. Nevertheless, reliability stays in the 99% range, and using more channels lowers the possibility of having correlated losses as it is less likely to hit channels polluted with the same interference.

5.4 Repetitions: Number of Transmissions

In this section, we evaluate BlueFlood performance with different number of transmissions NTx. We use 40 channels in a pseudo-random hopping sequence of length 128 to send iBeacon packets with -16 dB TX power. We vary the number of transmissions in [4, 8, 12] and repeat the experiments using the four Bluetooth modes.

Figure *10* shows the results. We notice that the link quality drops with higher number of repetitions, but the end-to-end performance, on the other hand, improves. The reason is that repeated transmissions improve the end-to-end packet delivery ratio exponentially: $PDR = 1 - (1 - PRR)^N$.

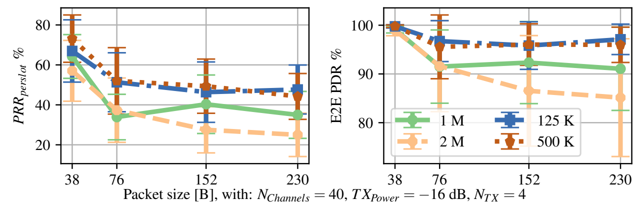


Figure 11. BlueFlood performance with different packet sizes. *The longer the packet, the lower the reliability.*

Thus, despite a reduced per-hop PDR, this leads to end-to-end $PDR > 99.5\%$ for all modes, at the expanse of energy. On the other hand, we see an interesting energy trade-off for the different modes: The 2 M mode reliability with 12 Tx is very close to that of 125 K, yet the former costs about 16 times less energy to send; *i.e.*, the cost of sending the whole round with the packet repeated 12 times is less than that for sending one packet in the mode 125 K.

5.5 Packet Size

In this section, we evaluate BlueFlood performance when sending larger packets. We use 40 channels in a pseudo-random hopping sequence of length 128 to send Beacon packets with -16 dB TX power with $NTx = 4$. We vary the size of the packet in [38, 76, 152, 230] bytes and repeat the experiments using the four Bluetooth modes. Notice that packets larger than 38 bytes are not compatible with iBeacons although we use the same format with longer payload, but they are still Bluetooth 5 compliant.

Figure *11* summarizes the results. We notice that the link quality drops with the larger packet size, and so does the end-to-end reliability. With the larger packet size, the probability of corruption due to either interference or reflections increases, as does the packet air time. The protection of FEC helps retaining a reliability close to 95% for the two coded modes. The most affected are the 2 M, then 1 M modes, which show the fragility of CT over Bluetooth for packets larger than classic 38-Byte beacons.

5.6 Compatibility with Unmodified Phones

We are using the testbed to run BlueFlood and test the reception of the CT of iBeacons from our testbed using an unmodified Samsung Galaxy S9. We run BlueFlood using $NTx = 4$ and Tx power -16 dB on channel 37 in the legacy 1 Mbps mode. We install a Bluetooth beacon scanner application and we enable the scanning mode. We place the phone in several locations in the apartment, and it is able to correctly decode our beacons. Due to the tight timing requirements of BlueFlood, we cannot reliably initiate the flood from the phone, but we can receive it.

6 Related Work

In this section, we discuss the state of the art of the broader field of concurrent transmissions and related phenomena such as constructive interference and capture effect and the protocols that based on these concepts in wireless sensor networks (WSNs) and Internet of Things (IoT). We provide the necessary technical background on Bluetooth and concurrent transmissions earlier in §2.

Concurrent Transmissions Protocols A-MAC [13] and Glossy [14] pioneered the field of concurrent transmissions in WSNs. LWB [15], Splash [11] and Choco [42] base on Glossy to schedule individual network-floods to provide data collection while Crystal [21] and its multichannel version [22] reduce the number of Glossy floods by relying on data prediction. CXFS [9], Sparkle [49] and others [23, 35, 8, 25, 48] limit the number of concurrent transmitters in Glossy or LWB while Sleeping Beauty [36] combines both limiting the number of transmitters by putting them to sleep and scheduling Glossy floods to improve energy efficiency.

SurePoint [6] builds an efficient concurrent network-wide flooding similar to Glossy in UWB and leverage it to provide a localization service, while Corbalán and Picco [10] employ the concurrent transmissions for ranging on UWB.

Chaos [26] on the other hand extends the design of Glossy to utilize the capture effect on 802.15.4 in the 2.4 GHz to let node transmit different data and efficiently calculate network-wide aggregated by employing in-network data processing. A^2 [4] takes this further by introducing communication primitives for network-wide consensus. However, since both of they base on capture of different data rather than flooding the same data, they are more difficult to support on uncoded communication technologies such as the Bluetooth modes 1 and 2 Mbps.

Overall, concurrent transmissions enable low-latency network-wide communication. While none of the aforementioned protocols support Bluetooth, the concepts are generally extendable to other technologies given that concurrent transmissions are supported. BlueFlood builds on these results to bring efficient network flooding to Bluetooth mesh networks.

Understanding Concurrent Transmissions While the capture effect is not new and was first observed for FM transmitters [27], the capture effect in low-power wireless networking was first studied by Son *et al.* [40] experimentally. The success of concurrent transmissions in Glossy started a debate on how these work and what underlying physical phenomena enable it. The authors of Glossy argue that the signals interfere constructively. Later, this was underlined by Rao *et al.* [33] who utilize Glossy style flooding and through precise timing can also achieve destructive interference to provide negative feedback.

In contrast, Wilhelm *et al.* [46] introduce analytical models backed with experiments to parameterize concurrent transmissions and show that these are rather non-destructive interference instead. Thus, they argue that the signals get degraded due to concurrent transmissions but still can be decoded. Moreover, they argue that coding is essential to improve the reliablity of concurrent transmissions. Similarly, Liao *et al.* [28] argue it is DSSS ans its coding that lets CT survive beating. While the mentioned papers are limited to 802.15.4 in the 2.4 GHz, Liao *et al.* [29] has a limited study on CT over 802.15.4 in subGHz. Roest [34] studies the capture effect and evaluates Chaos on BLE, 1 Mbps. To the best of our knowledge, no prior research has evaluated and utilized CT over Bluetooth 5 as this paper does.

Low-power Channel Hopping Using frequency diversity techniques has proven to be effective for combating interference [44]. It is wide-spread both in the established standards; such as Bluetooth [7], TSCH [20] and in the state of the art such as the top solutions in the dependability competition [38] and BLEach [41]. BLEach not only enables adaptive channels black-listing and adaptive duty cycling to provide quality of service guarantees, but implements IPv6 over BLE as in RFC 7668 [24]. However, it only supports star networks as opposed to BlueFlood which supports multihop Bluetooth mesh networks.

7 Conclusion

This paper introduces and evaluates concurrent transmissions over Bluetooth PHY. We argue that the recent approaches to concurrent transmissions based on Glossy, are key enablers for such protocols. We present BlueFlood: a network stack based on concurrent transmissions to provide low power, low-latency and reliable flooding and data dissemination to Bluetooth mesh networks that are battery operated. Our experimental evaluation shows that: (i) Although CT is more fragile over Bluetooth PHY than it is over 802.15.4, it is a viable communication strategy for network-wide dissemination; (ii) BlueFlood achieves data dissemination with high reliability, low power and low latency; (iii) the choice of the transmissions mode provides a tradeoff between reliability, energy, and latency; and (iv) BlueFlood floods can be received on unmodified phones.

8 Acknowledgments

We would like to thank Coen Roest for his master thesis work on Enabling the Chaos Networking Primitive on Bluetooth LE. This work was supported by the Swedish Research Council VR through the ChaosNet project, the Swedish Foundation for Strategic Research SSF through the LoWi project, and Sweden's innovation agency VINNOVA.

9 References

[1] A. Escobar et al. Competition: Redfixhop with channel hopping. In *Proc. of the Intl. Conf. on Embedded Wireless Systems and Networks (EWSN)*, 2017.

[2] A. Escobar et al. Competition: Bigbangbus. In *Proc. of the Intl. Conf. on Embedded Wireless Systems and Networks (EWSN)*, 2018.

[3] ABI Research. Installed Base of Bluetooth-enabled Devices Worldwide in 2012 and 2018. Statista - The Statistics Portal, 2018.

[4] B. Al Nahas, S. Duquennoy, and O. Landsiedel. Network-wide Consensus Utilizing the Capture Effect in Low-power Wireless Networks. In *Proc. of the Conf. on Embedded Networked Sensor Systems (ACM SenSys)*, 2017.

[5] B. Al Nahas and O. Landsiedel. Competition: Towards low-power wireless networking that survives interference with minimal latency. In *Proc. of the Intl. Conf. on Embedded Wireless Systems and Networks (EWSN)*, 2017.

[6] B. Kempke et al. Surepoint: Exploiting ultra wideband flooding and diversity to provide robust, scalable, high-fidelity indoor localization.

In *Proc. of the Conf. on Embedded Networked Sensor Systems (ACM SenSys)*, 2016.
[7] Bluetooth SIG. Bluetooth 5 Core Specifications., 2016.
[8] M. Brachmann, O. Landsiedel, and S. Santini. Concurrent transmissions for communication protocols in the internet of things. In *Proc. of the Conf. on Local Computer Networks (IEEE LCN)*, 2016.
[9] D. Carlson, M. Chang, A. Terzis, Y. Chen, and O. Gnawali. Forwarder Selection in Multi-transmitter Networks. In *Proc. of the Conf. Distributed Computing in Sensor Systems (DCOSS)*, 2013.
[10] P. Corbalán and G. P. Picco. Concurrent ranging in ultra-wideband radios: Experimental evidence, challenges, and opportunities. In *Proc. of the Intl. Conf. on Embedded Wireless Systems and Networks (EWSN)*, 2018.
[11] M. Doddavenkatappa, M. C. Chan, and B. Leong. Splash: Fast data dissemination with constructive interference in wireless sensor networks. In *Proc. of the Symposium on Networked Systems Design & Implementation (USENIX NSDI)*, 2013.
[12] A. Dunkels, B. Gronvall, and T. Voigt. Contiki - A Lightweight and Flexible Operating System for Tiny Networked Sensors. In *Proc. of the Conf. on Local Computer Networks (IEEE LCN)*, 2004.
[13] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.-J. M. Liang, and A. Terzis. Design and Evaluation of a Versatile and Efficient Receiver-Initiated Link Layer for Low-Power Wireless. In *Proc. of the Conf. on Embedded Networked Sensor Systems (ACM SenSys)*, 2010.
[14] F. Ferrari et al. Efficient Network Flooding and Time Synchronization with Glossy. In *Proc. of the Conf. on Information Processing in Sensor Networks (ACM/IEEE IPSN)*, 2011.
[15] F. Ferrari, M. Zimmerling, L. Mottola, and L. Thiele. Low-Power Wireless Bus. In *Proc. of the Conf. on Embedded Networked Sensor Systems (ACM SenSys)*, 2012.
[16] F. Ferrari, M. Zimmerling, L. Mottola, and L. Thiele. LWB and Glossy Source Code for CC430 SoC. `https://github.com/ETHZ-TEC/LWB`, 2016.
[17] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection Tree Protocol. In *Proc. of the Conf. on Embedded Networked Sensor Systems (ACM SenSys)*, 2009.
[18] Z. He, K. Hewage, and T. Voigt. Arpeggio: A penetration attack on glossy networks. In *Proc. of the Conf. on Sensor, Mesh and Ad Hoc Communications and Networks (IEEE SECON)*, 2016.
[19] K. Hewage. LWB and Glossy Source Code for CC2538 SoC. `https://github.com/kasunch/lwb-cc2538`, 2017.
[20] IEEE. 802.15.4-2015: IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) 1: MAC sublayer, 22 April 2016.
[21] T. Istomin, A. L. Murphy, G. P. Picco, and U. Raza. Data prediction + synchronous transmissions = ultra-low power wireless sensor networks. In *Proc. of the Conf. on Embedded Networked Sensor Systems (ACM SenSys)*, 2016.
[22] T. Istomin, M. Trobinger, A. L. Murphy, and G. P. Picco. Interference-resilient ultra-low power aperiodic data collection. In *Proc. of the Conf. on Information Processing in Sensor Networks (ACM/IEEE IPSN)*, 2018.
[23] J. Jeong et al. Low-power and topology-free data transfer protocol with synchronous packet transmissions. In *Proc. of the Conf. on Sensor, Mesh and Ad Hoc Communications and Networks (IEEE SECON)*, 2014.
[24] J. Nieminen et al. IPv6 over Bluetooth Low Energy. RFC 7668, 2015.
[25] J. Zhang et al. RFT: Identifying Suitable Neighbors for Concurrent Transmissions in Point-to-Point Communications. In *Proc. of the Conf. on Modeling, Analysis and Simulation of Wireless and Mobile Systems (ACM MSWiM)*, 2015.
[26] O. Landsiedel, F. Ferrari, and M. Zimmerling. Chaos: Versatile and Efficient All-to-All Data Sharing and In-Network Processing at Scale. In *Proc. of the Conf. on Embedded Networked Sensor Systems (ACM SenSys)*, 2013.
[27] K. Leentvaar and J. Flint. The Capture Effect in FM Receivers. *IEEE Trans. Commun.*, 24(5), 1976.
[28] C.-H. Liao, Y. Katsumata, M. Suzuki, and H. Morikawa. Revisiting the So-Called Constructive Interference in Concurrent Transmission. In *Proc. of the Conf. on Local Computer Networks (IEEE LCN)*, 2016.
[29] C.-H. Liao, M. Suzuki, and H. Morikawa. Poster Abstract: Toward Robust Concurrent Transmission for sub-GHz Non-DSSS Communication. In *Proc. of the Conf. on Embedded Networked Sensor Systems (ACM SenSys)*, 2016.
[30] R. Lim, R. D. Forno, F. Sutton, and L. Thiele. Competition: Robust flooding using back-to-back synchronous transmissions with channel-hopping. In *Proc. of the Intl. Conf. on Embedded Wireless Systems and Networks (EWSN)*, 2017.
[31] P. D. Marco, P. Skillermark, A. Larmo, and P. Arvidson. Bluetooth mesh networking. Ericsson AB, 2017.
[32] Nordic Semiconductor. nRF52840 Product Specification, 2018.
[33] V. S. Rao, M. Koppal, R. V. Prasad, T. V. Prabhakar, C. Sarkar, and I. Niemegeers. Murphy loves ci: Unfolding and improving constructive interference in wsns. In *Proc. of the Conf. on Computer Communications (IEEE INFOCOM)*, 2016.
[34] C. Roest. Enabling the Chaos Networking Primitive on Bluetooth LE. Master's thesis, Chalmers Univ. of Tech. and TU Delft, 2015.
[35] C. Sarkar. Lwb and fs-lwb implementation for sky nodes using contiki. arXiv preprint, 2016.
[36] C. Sarkar, R. V. Prasad, R. T. Rajan, and K. Langendoen. Sleeping beauty: Efficient communication for node scheduling. In *Proc. of the Conf. on Mobile Ad Hoc and Sensor Systems (IEEE MASS)*, 2016.
[37] T. Schmid, P. Dutta, and M. B. Srivastava. High-resolution, low-power time synchronization an oxymoron no more. In *Proc. of the Conf. on Information Processing in Sensor Networks (ACM/IEEE IPSN)*, 2010.
[38] M. Schuß, C. Boano, M. Weber, and K. Römer. A Competition to Push the Dependability of Low-Power Wireless Protocols to the Edge. In *Proc. of the Intl. Conf. on Embedded Wireless Systems and Networks (EWSN)*, 2017.
[39] Silicon Labs. Bluetooth Mesh Network Performance. `https://www.silabs.com/documents/login/application-notes/an1137-bluetooth-mesh-network-performance.pdf`, 2018.
[40] D. Son, B. Krishnamachari, and J. Heidemann. Experimental Study of Concurrent Transmission in Wireless Sensor Networks. In *Proc. of the Conf. on Embedded Networked Sensor Systems (ACM SenSys)*, 2005.
[41] M. Spörk, C. A. Boano, M. Zimmerling, and K. Römer. Bleach: Exploiting the full potential of ipv6 over ble in constrained embedded iot devices. In *Proc. of the Conf. on Embedded Networked Sensor Systems (ACM SenSys)*, 2017.
[42] M. Suzuki, Y. Yamashita, and H. Morikawa. Low-power, end-to-end reliable collection using glossy for wireless sensor networks. In *IEEE Vehicular Technology Conf. (VTC Spring)*, 2013.
[43] T. Winter et al. RPL: IPv6 routing protocol for low-power and lossy networks. RFC 6550 (Proposed Standard), 2012.
[44] T. Watteyne, S. Lanzisera, A. Mehta, and K. Pister. Mitigating Multipath Fading through Channel Hopping in Wireless Sensor Networks. In *Proc. of the IEEE Intl. Conf. on Communications (IEEE ICC)*, 2010.
[45] Wikipedia contributors. Bluetooth low energy beacon — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Bluetooth_low_energy_beacon&oldid=851685016`, 2018.
[46] M. Wilhelm, V. Lenders, and J. B. Schmitt. On the Reception of Concurrent Transmissions in Wireless Sensor Networks. *IEEE Trans. Wirel. Commun.*, 2014.
[47] M. Woolley. Bluetooth 5: Go Faster. Go Further. Bluetooth SIG, 2018.
[48] D. Yuan and M. Hollick. Ripple: High-throughput, reliable and energy-efficient network flooding in wireless sensor networks. In *Proc. of the Symposium on a World of Wireless Mobile and Multimedia Networks (IEEE WoWMoM)*, 2015.
[49] D. Yuan, M. Riecker, and M. Hollick. Making 'Glossy' Networks Sparkle: Exploiting Concurrent Transmissions for Energy Efficient, Reliable, Ultra-Low Latency Communication in Wireless Control Networks. In *Proc. of the Intl. Conf. on Embedded Wireless Systems and Networks (EWSN)*, 2014.