ReliaBLE: Towards Reliable Communication via Bluetooth Low Energy Advertisement Networks

Thomas Zachariah*, Neal Jackson*, Branden Ghena†, Prabal Dutta*
*University of California, Berkeley, †Northwestern University
tzachari@berkeley.edu, neal.jackson@berkeley.edu, branden@northwestern.edu, prabal@berkeley.edu

Abstract

Bluetooth Low Energy (BLE) has emerged as a prominent low-power communication protocol for Internet of Things (IoT) devices due to its affordability, simplicity, efficiency, and compatibility with mobile platforms. We consider the use of advertisements, small periodic packets broadcast by BLE devices, as a potentially useful transport for sensor network data, worthy of deeper investigation. In this paper, we analyze a network paradigm, which facilitates communication between IoT devices and gateways through BLE advertisements—an approach suitable for low-power devices due to its simplicity and compatibility with existing BLE stacks. To understand the behavior of such BLE advertisement networks, we derive and empirically validate analytical models to predict their performance. We also evaluate real-world deployments of BLE advertisement networks, first exploring a dataset from a prior deployment of BLE sensors accounting for over 600 sensed days across multiple locations. We show that with redundancy and suitable gateway hardware, BLE advertisements can be used to create a network achieving 99% data reception, and with a dynamic adaptation protocol, can do so in networks with varying contention. We design and deploy our own advertisement networks that incorporate our models, demonstrating that they can predict average network performance, that advertisements can be used as the base for a reliable network, and that BLE advertisement networks may prove to be a valuable tool for the sensor network community.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless Communication*

General Terms

Design, Measurement, Performance, Reliability

Keywords

BLE, Advertisement Network, Deployment, Adaptation

1 Introduction

Bluetooth Low Energy (BLE) beacons are short-range broadcast transmitters that are widely used as a method of enabling consumer devices like smartphones, computers, and gateways to detect the presence of interactable objects and locations [46]. Academic projects are exploring the use of BLE as well in applications such as long-term health tracking [48], environmental monitoring [25], and indoor localization [12].

One communication mechanism of BLE is the advertisement — a simple, periodic, broadcast message intended for device discovery. Advertisements reduce or eliminate listening costs for energy constrained devices, avoid interference via channel diversity, and are simple to specify in software. With advertisements, we can create a single-hop, star-topology network in full compliance with the BLE specification that allows any number of devices to send data to any number of gateways, with little software complexity. We refer to this as a BLE Advertisement Network.

While they only provide unidirectional communication, advertisements are useful for collecting data from sensor deployments. Placing sensor measurements in advertisement payloads also allows smartphones to receive and interpret the data, enabling easy network introspection. Deployed devices from asset trackers [44] to plug-load power meters [9] are already using advertisements for data transfer, but we find that these types of networks have not been rigorously studied in literature. In this paper, we explore the BLE advertisement primitive to understand how well it performs in various conditions and how it can best be used for sensor networks.

To predict expected performance before deployment, analytical models are needed that explain the impacts of network configurations, such as transmission frequency and number of deployed devices. While advertisements are ALOHA transmissions at heart [1], they are not identical to them. BLE advertisements are periodic, which means that the probability of repeat collisions is greater than the normal ALOHA expectation. We extend the efforts of prior work [20, 21] to analytically describe reception rates for periodic transmissions in terms of BLE advertisement parameters. We also experimentally validate our models, demonstrating that they accurately represent reality through controlled studies.

While these models do not fully describe deployed networks, they are useful for determining expected performance. The simple access control mechanism of BLE leads to significant packet loss as the number of devices in a deployment

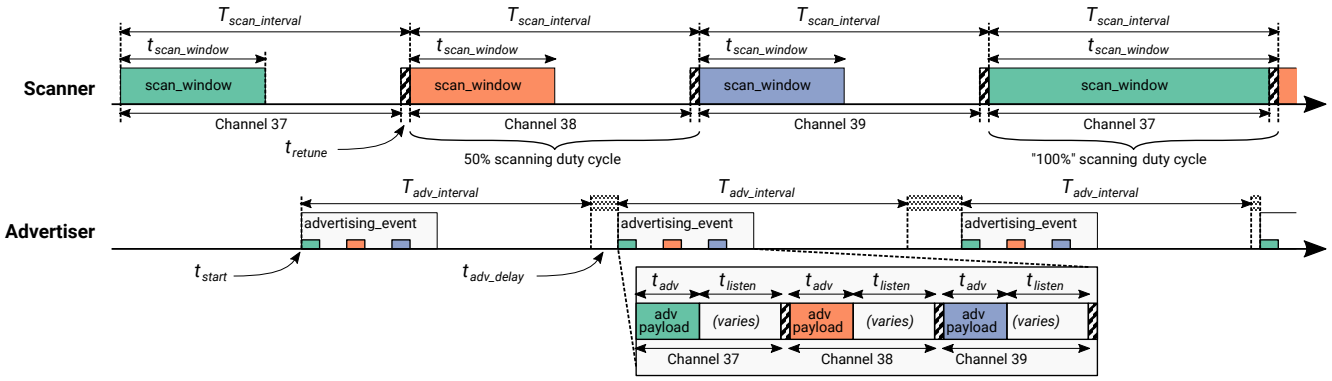

Figure 1: BLE Advertisement Overview. During BLE advertising, the advertiser and scanner are unsynchronized and operate independently. The scanner rotates listening across the three advertising channels, changing channels every $T_{scan\_interval}$. Energy-conscious scanners can separately configure t_{scan_window} which controls the duty-cycle of listening on each channel. At any point t_{start} an advertiser may begin sending advertisement events. Inside an advertisement event, advertisers send an identical payload on all three channels. The delay between advertisement events is controlled by $T_{adv\_interval}$, which dictates the minimum interval as a random $0-10$ ms t_{adv_delay} which is added to each interval. A scanner will successfully receive an advertisement when the advertisement payload transmission and scan window align, which occurs once on Channel 39 in this example.

increases, but we find that through the addition of redundancy, data reception rates can remain high in many environments. Deployment-specific factors, such as external interference or distance from nodes to the gateway, may hinder the connectivity of the network, but descriptions of network capacity alone allow a baseline performance expectation to be determined.

Using our models, we can identify when real-world deployments are underperforming. We analyze a dataset collected from a previous deployment of sensors using a BLE advertisement network with 335 power meters [9] installed across nine locations for an average of 68 days in each location. In applying the models to this dataset, we observe increased network performance, falling short, however, of anticipated theoretical reception. An investigation reveals the problem not in the network, but rather in shortcomings of the hardware and software of the BLE gateway. We explore the gateway issues, and in new deployments we show that with improved gateways we can accurately predict average network performance, demonstrating the efficacy of our models for planning successful sensor network deployments. Furthermore, we design and implement the ReliaBLE adaptation protocol, which allows devices to dynamically adjust transmissions to maintain high data reception in networks with varying contention.

Building on the success of the original Bluetooth protocol, BLE has blossomed to a point of stability and ubiquity. This work moves towards deeply understanding BLE for sensor network applications, demonstrating that BLE advertisements can be used as a reliable transport for real-world deployments.

2 Background

Bluetooth Low Energy (BLE) is defined by the Bluetooth Special Interest Group [7]. While it shares a name and some similarities with classic Bluetooth networks, it is a distinct protocol. We discuss the primitives of BLE networking and explore prior ideas for how BLE could be adapted to sensor networking applications. Figure 1 depicts the major protocol elements of BLE advertising discussed in this section.

Advertising. In BLE, advertising is nominally the method for device discovery. An advertisement event is made up of three packets sent in rapid succession at periodic intervals. Each advertisement packet is followed by a brief window for listening. This limited listening for end devices is the "low energy" part of BLE, allowing devices to primarily sleep with their radios entirely off. Reasonably frequent advertisement events enable quick device discovery and interaction.

Advertisement packets include: a fixed 16 bytes of preamble, address, CRC, and other headers, and up to 31 bytes of payload. Sent over the 1 Mbps physical layer, advertisement packets have an on-air time of 128–376 µs. In an advertising event, identical packets are sent redundantly on three channels to mitigate interference, with a listening period before changing channels. The BLE specification reserves three channels for these advertisements, placed in the spectrum to avoid WiFi channels 1, 6, and 11.[1] Devices emit advertisement events at a set interval, $T_{adv\_interval}$, (configurable from 20 ms to 10.24 s) plus a stochastic jitter, t_{adv_delay} (selected uniformly from 0–10 ms). The injected jitter randomly distributes transmissions in time and prevents collisions from repeating indefinitely.

Broadcasts are sent without collision avoidance or channel sensing. As a network of unsynchronized blind transmitters, the aggregate throughput of BLE advertisements can thus be modeled as an ALOHA network [28]. The use of three advertisement channels provides no collision avoidance in the common case as they are iterated in the same order by all transmitters and there is no entropy in t_{listen}. In the absence of significant jitter or fading, a collision on one channel will result in a collision on the others.

Scanning. BLE scanners are devices listening for advertisements. They listen on one advertisement channel at a time, periodically rotating to the next channel. The BLE specification allows scanners to set a configuration for the receiver duty cycle on each channel, which can be scaled

[1]The advertisement channels are named 37, 38, and 39, but are not adjacent in the spectrum. They are at 2402, 2426, and 2480 MHz respectively.

from 0 to 100%. In practice, gateways running on wall power select 100% duty cycle to receive all advertisements. Power-constrained devices, such as smartphones, can select lower duty cycles to conserve energy. Android, for instance, defines multiple scanning modes that applications can select, which correspond to 10%, 25%, and 100% duty cycle scanning [2].

Reducing the duty cycle can have a large impact on the success of receiving data. Particular choices of scanning and advertising intervals may result in poor synchronization that results in a high discovery latency [20]. Some platforms, such as iOS, provide no application control over the scan window [26] which can require advertisers to set aggressive advertisement intervals to realize responsive designs, e.g. Apple recommends a 20 ms advertising interval for discovery [4].

In practice, even scanning hardware set to 100% duty cycle does not achieve the theoretical 100% reception rate. Perez-Diaz et al. study the performance of the major BLE chipsets, finding that, in practice, receivers fail to receive packets for brief periods when switching channels or decoding received packets, and periodically throughout the scan [38]. The losses from these "blind spots" can be as high as 10% depending on packet size, advertising interval, and scanning interval.[2]

Scan Requests. When a scanner receives an advertisement, it may choose to request additional data from the advertiser by sending a scan request. During the advertiser's listening period following a transmission, if it receives a scan request, it responds by sending an additional advertisement payload of up to 31 bytes, termed a scan response. Scan responses are generally used to provide additional data that may not have fit in the original advertisement payload.

Hernandez et al. suggest that the presence of a scan request could be used as a form of acknowledgment, allowing transmitters to reduce or cease transmitting for some duration after receiving one, reducing overall contention in the network [18]. However, Harris and Kravets explain how the BLE backoff protocol works against this idea [17, 24]. If a scanner makes a request and does not receive a response, it assumes there was a collision with another scanner and adds a random delay before requesting again. This backoff mechanism cannot distinguish the case where the request (or response) collided with another advertiser, which becomes increasingly likely as network density grows. This is further confounded as scanners back off exponentially, thus even a modest number of collisions will result in an artificially low number of scan request "acknowledgments", incorrectly underestimating link quality and necessitating yet more advertisements. Rather than send additional data in scan responses, Kravets et al. recommend splitting data across successive advertisement payloads [24].

Connections. Connections are the method for high throughput, bi-directional communication in BLE. After receiving an advertisement, a scanner can send a connection request to the advertiser. Both devices then move into a hopping pattern across the 37 channels reserved for connections.

[2]For example, while nRF52832 hardware is capable of tuning frequencies in 40 µs [37], in practice the Nordic softdevice takes roughly 800 µs to switch scan channels [40]. The Noble BLE library sets a default scan interval of 10 ms [32].Were one to scan with Noble atop a Nordic softdevice, it would have an effective duty cycle of only 92%.

The scanner, which initiated the connection, becomes the master in charge of scheduling connection events—when packets are actually exchanged. A master connected to multiple slaves schedules them with both time division and channel division multiplexing. BLE connections do not scale in practice, but theoretically, the only limit to the number of connected devices is the ability to schedule them. At connection time, the master adds an offset—in 1.25 ms steps—to the start time of the first communication event. The specification allows connection periods from 7.5 ms to 4000 ms, which translates to a maximum number of 6 to 3200 devices that can be connected to at one time without overlap. Real-world BLE chipsets are significantly more constrained than this theoretical limit. Firmware on many BLE radios limit the number of simultaneous connections to less than ten [32]. The open source MyNewt BLE stack supports the most simultaneous connections of any we survey at 32 [3].

BLE 5.0 adds "periodic advertisements", which use connection channels to allow payloads up to 255 bytes and higher data rates [7]. However, these are still initiated via the BLE 4.2 advertisement mechanism we focus on in this work.

Gateways. Gateways are bridge devices that translate the communication protocols of low-power devices to the Internet at large. For BLE networks, gateways are effectively scanners that have a connection to the Internet. A wide range of options can be used to fulfill this requirement—from a Linux box to a barebones microcontroller with BLE and WiFi. These choices, as we discuss, can significantly impact network performance.

3 Related Work

This work is not the first to have explored an architecture for BLE networking. Various techniques for mesh networks on top of BLE have been explored. Darroudi and Gomez provide an overview of the academic, commercial, and standards-organization approaches to BLE mesh [8]. BlueFlood and BLEmesh implement mesh networks on top of BLE advertisements [34, 22]. These techniques could be improved through the use of the advertising models discussed in this paper. IPv6 networking over a BLE connection is studied by BLEach [41]. It does not expand to handle large deployment sizes, however, exploring connections to 4–8 devices total. We expand these works with our investigation into BLE advertisement networks, which provide different tradeoffs for deployments, instead emphasizing simplicity and scalability.

The theoretical models for advertisement networks are informed by several prior works. The ALOHA system first describes the access control method that will later be used by BLE [1]. Liu et al. first study the probability of collisions for BLE advertisements using the Poisson distribution, much like ALOHA [28]. They calculate delay before device discovery rather than packet reception rates. Similarly motivated, Jeon et al. create an iterative model for determining discovery latency [20]. Harris et al. adopt a probabilistic model for packet collisions, similar to our own models but missing the effect of the delay added to each interval [16]. Perez-Diaz et al. [38] include that random delay, coming to the same result we do in Equation (1). Our models go beyond this prior work by accounting for heterogeneous node configurations and observing the increased probability of repeat collisions in BLE.

Rather than just presenting models for collisions, we also use BLE parameters to describe network reception rates. While our models only encompass network performance, prior work also models energy use of BLE [20, 29], which is valuable for a wholistic understanding of BLE deployments.

We also draw inspiration from studies of other network technologies. Similar to this work, Kohvakka et al. create analytical models for ZigBee networks, including probabilistic models of collisions [23]. Szewczyk et al. analyze data from their deployment on Great Duck Island to detect and determine causes of network failures [43]. The Roofnet deployment [6] evaluates a real-world deployment of an 802.11b mesh network with an unplanned topology. Like this work, they evaluate success of a simple-to-deploy network in the real world. Srinivasan et al. [42] make empirical measurements of 802.15.4 networks to challenge assumptions common to protocol design and inform future design efforts. They suggest that a gap exists between research work in networking and performance of real-world network deployments which needs to be solved by understanding root causes of deployment failures. This work is notably inspired by LoRaWAN [30], a communication protocol whose network architecture shares many traits with BLE advertisement networks, but facilitates longer-range needs with generally lower throughput and higher dedicated infrastructure costs. Studies have explored alternative access control methods and gateway improvements for LoRaWAN [10, 11, 14, 45]. We expand these works by exploring BLE networks, and follow their spirit by investigating results of real-world network deployments.

4 BLE Advertisement Networks

Prior efforts have considered neighbor discovery protocols utilizing BLE [21], how BLE gateways could provide general-purpose services to sensors [50], and IPv6 networking over BLE connections [41]. However, exploration of BLE advertisements for IoT networking at scale is lacking. In this section, we motivate BLE advertisements as a mechanism atop which to build an IoT network, explore the design space of possible network architectures, and develop models for BLE advertisement-based networking to investigate how BLE parameters affect the resulting network performance.

4.1 Why Network with Advertisements?

Using BLE advertisements for communication has its advantages. Advertisements, and BLE in general, enable communication directly with people through presence in personal devices. Anyone with a phone can easily discover and collect transmissions from BLE devices, unlike other low-power networks. Second, advertisements are simple. For radios implementing BLE, the interface for sending data over advertisements can be as straightforward as providing a payload and interval. Even when using a raw radio interface, BLE advertisements do not requiring tight timing or complex access control. Due in part to this simplicity, BLE advertisements are very low energy. This makes them a good choice for power-constrained devices, like intermittent computing systems, which cannot reliably participate in scheduled networks as devices may not have energy when needed [31, 47].

Finally, advertisements scale to many devices in a way that connections do not in practice. While a single gateway can theoretically connect to many devices at once, BLE firmware has much lower limits. Users report that common USB dongle chipsets allow less than ten simultaneous connections [32]. In contrast, there is no limit to the number of devices from which a scanner can receive advertisements, apart from channel utilization. Additionally, use of advertisements does not preclude use of connections. The latter may be useful for infrequent, complex operations such as updates to device configuration or firmware. In this study, we focus on steady-state operation of networks during data collection, assuming such bi-directional interactions are rare.

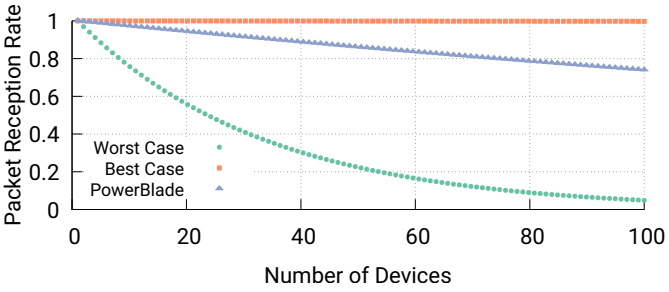

Figure 2: Packet reception as deployment size increases. Both the worst case (31 byte payload broadcast every 20 ms) and best case (0 byte payload every 10240 ms) are displayed. PowerBlade [9], a research BLE sensor node that transmits 23 bytes every 200 ms, is also shown as a real-world example. Small deployments (<10 nodes) often yield acceptable PRR. Larger ones must balance desired throughput and reception.

4.2 Communication with Advertisements

BLE advertisements have a user-configurable payload of 0–31 bytes. While some payload bytes have specialized purposes, such as a leading flags field or a company identifier, in practice advertisers can send 31 arbitrary bytes.[3] Advertisers may send identical data in every packet (e.g. to act as a proximity beacon) or change packet contents and length on every message (e.g. to report instantaneous sensor data). To improve reliability at the expense of latency and throughput, an advertiser could repeat the same data for several packets in a row before updating the data. The frequency of advertising packets is controlled by the advertisement interval, $T_{adv\_interval} \in [20\,\text{ms}, 10.24\,\text{s}]$, plus a random additional delay, t_{adv_delay} drawn uniformly from $[0, 10]$ ms at each period. The maximum theoretical goodput for one node is thus 9.92 kb/s.

4.3 Advertisement Collision Model

We explore the potential of advertisement-based networking by modeling collisions of BLE advertisers. First, let us assume that distance and channel do not affect packet reception—that all sensor nodes in the network are within range of the gateway receiving their data and that the received signal strength from each is identical on all advertising channels. This is the worst-case for collisions; in practice the capture effect will mitigate some collisions. Next, consider a wall-powered gateway that is always listening and always

[3] Desktop OSes & Android permit scanning apps to access raw advertisements. iOS requires the first 4 bytes (flags & company ID) be well-formed to recover advertisements.

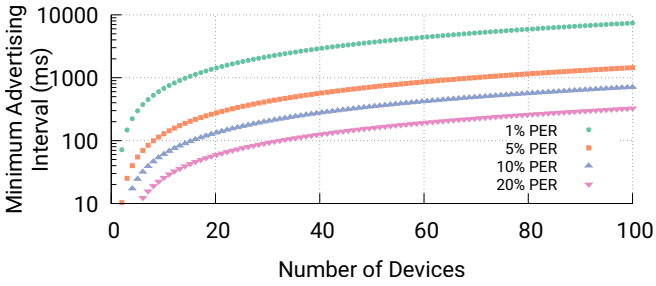

Figure 3: Intervals required for target packet error rates. Given a fixed payload (here 31 bytes), to realize a target packet error rate, the minimum advertising interval must grow with the number of devices. Even small deployments require several hundred milliseconds between transmissions to achieve a 1% packet error rate. Accepting 10% error rates allows sub-second intervals even as deployments expand to 100 devices.

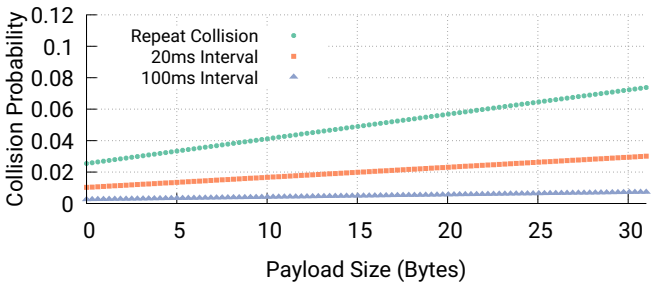

Figure 4: Probability of collision. The likelihood of a collision between two BLE transmitters grows as the payload size of the packet increases. The probability of a repeat collision is increased due to the periodicity of BLE. Given a collision on the previous packet, the probability of collision for the current transmission in BLE is twice as high as the normal probability for even the fastest advertisement interval.

able to receive packets (such a gateway is quite reasonable, effectively $t_{scan\_window} = T_{scan\_interval}$ and $t_{scan\_window} >> t_{retune}$). As explained in Section 2, because a collision on any advertising channel will collide on all advertising channels and we are treating the propagation of each channel as identical, we can therefore ignore the channel of the scanner altogether. Finally, to start we will assume that all advertisements have identical payload sizes and that there is no other interference.

Probabilistic Model. Literature describes the basic model for BLE advertisement collisions [38]. The probability of collision, P_c, for N advertising BLE devices, is:

$$P_c = 1 - \left(1 - \frac{2 \times t_{adv}}{T_{adv\_interval} + \mathbb{E}(t_{adv\_delay})}\right)^{N-1} \quad (1)$$

Additional losses will come from interference with other technologies, P_i. We can express the packet reception rate, PRR, more generally then as the probability of neither colliding nor being interfered with:

$$PRR = (1 - P_c)(1 - P_i) \quad (2)$$

In general, the BLE advertisement channels are positioned so they fall outside of the bands of the normal WiFi channels (1, 6, and 11). Narendra et al. study BLE interference from a single WiFi access point and find little to no packet loss when using BLE channels that avoid the main WiFi channels [35]. However, empirical studies find that many real-world WiFi deployments use every channel [33] and further interference may still be caused by 802.15.4 traffic or other transmissions in the 2.4 GHz ISM band. We simplify by assuming that P_i is zero for the remainder of this analysis and focus on refining the estimate of collision probability, P_c.

Using the Probabilistic Model. In Figure 2 we use this model to explore the effects of number of devices, payload length, and advertising interval on packet reception rate. Configuring devices for the highest throughput—full payloads transmitted at the highest frequency—results in the highest probability of collision and the lowest packet reception rate.

Certain applications may have acceptable packet error rates, PER, to meet their requirements. We can solve Equation (1) to determine the minimum advertising interval that satisfies a given packet error rate and number of devices:

$$T_{adv\_interval} = \frac{2 \times t_{adv}}{1 - (1 - PER)^{\frac{1}{N-1}}} - \mathbb{E}(t_{adv\_delay}) \quad (3)$$

Figure 3 plots the impact on latency as network density scales for various target error rates. As expected for an ALOHA-style network, high throughput for many devices can only be achieved by sacrificing reliability for any given packet.

Extending to Heterogeneous Configurations. One assumption made previously was that all devices on the network act identically. But we can extend the model from Equation (1) to remove this requirement. Assume a primary device with an advertisement transmission time of t_{adv_0} and a second possibly colliding device with a transmission time of t_{adv_i}. A collision occurs if the second device begins transmitting any time during t_{adv_0} or up to t_{adv_i} before the primary device begins transmitting. If the transmission from the second device is uniformly distributed in time, it is equally likely to occur during any point in its advertising interval. Generalizing this second device to all nodes in the network, we can express the probability of collision with the primary device, P_{c_0}, as:

$$P_{c_0} = 1 - \prod_{i=1}^{N-1} \left(1 - \frac{t_{adv_0} + t_{adv_i}}{T_{adv\_int_i} + \mathbb{E}(t_{adv\_delay})}\right) \quad (4)$$

4.4 Packet Reception to Data Reception

In real-world deployments, we are not interested in the reception of individual packets, but rather the eventual recovery of their payload. If we want to increase the probability that any particular payload is received, we can repeat it. Then, for data to be lost all redundant packets sent must be lost. A naïve model for this would use P_c as the probability for each failure. However, repeat collisions in BLE are *not* independent. Given that the first packet collided, the probability of a second collision is greater than a random collision. Indeed, absent t_{adv_delay} (or clock drift), once a single packet suffered

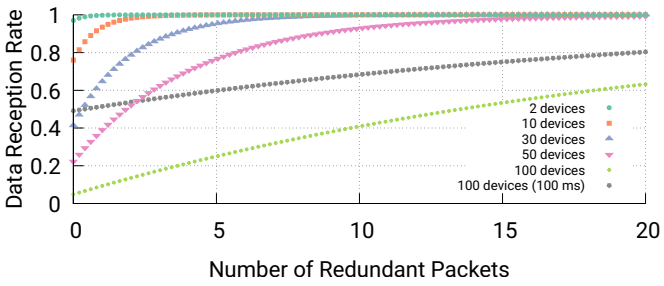

Figure 5: Data reception for redundant transmissions. The number of redundant packets transmitted is varied for multiple deployment sizes, each transmitting at a 20 ms interval, with an extra 100 device scenario at 100 ms interval. As deployment density and network contention grow, advertisers must send more redundant packets to maintain data reception reliability. Slower advertising (e.g. 100 ms case) improves reception at the expense of responsiveness and throughput.

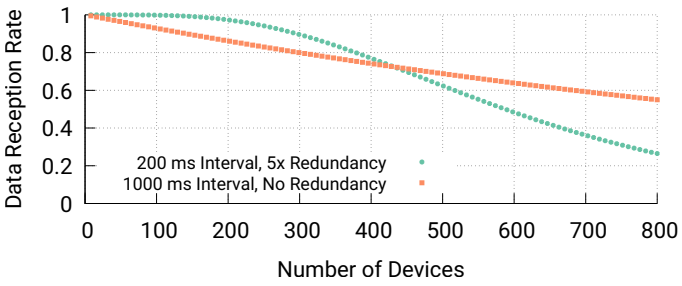

Figure 6: Data reception rate favors redundancy. Data reception rates are compared across deployment sizes for one network configured to send a packet every 1000 ms versus another that sends five redundant packets per second at 200 ms intervals. A crossover point occurs where data reception gains due to redundancy is not enough to overcome losses due to network congestion, but the expected maximum deployment size is typically far below this point for many applications.

a collision all future packets would collide too, as long as each device is using the same $T_{adv\_interval}$ (i.e. a network of homogeneous devices). This is a change from traditional ALOHA analysis, which does not have an assumption of periodicity, and results in an increased probability of a repeat collision.

After a collision, a repeat collision occurs if the sum of the differences in initial transmission times, advertising intervals, and selected random delays for two devices are less than the duration of the advertisement. For the homogeneous deployment case (advertising interval difference and average difference in initial transmission time are both zero), a repeat collision occurs if the difference in random delays has not moved one advertisement outside of the transmission time of the other. The difference of random delays creates a triangular distribution which we can integrate to determine the probability of collision. The full equation for the probability of a repeat collision is demonstrated in Equation (5).

$$P_{rc} = 1 - \left[\left(1 - 2 \int_0^{t_{adv}} \frac{1}{t_{adv\_delay}} - \frac{x}{t_{adv\_delay}^2} \mathrm{d}x \right) \times \left(1 - \frac{2 \times t_{adv}}{T_{adv\_interval} + \mathbb{E}(t_{adv\_delay})} \right)^{N-2} \right] \quad (5)$$

Note that this does not account for the case where more than one node collides with the primary transmission, which means it slightly underestimates the number of repeat collisions.

Figure 4 visualizes the impact of repeat collisions for the two-node case and finds that the odds of a repeat collision range from 2% to 7%, more than twice as much as the naïve ALOHA assumption. This increased probability of collision can have significant impact on device discovery, and may lead to extended periods when all packets from a device collide.

To extend this to model the probability that data is received, note that the probability of a second collision is the same as the probability of a third collision, and so on. The probability of the first collision is the normal probability of collision from Equation (4), while the second and onward are the probability of a repeated collision from Equation (5). For a series of M packets with the same data payload, we express data reception rate, DRR, as the odds that data is received at least once:

$$DRR = 1 - (P_c)(P_{rc})^{M-1} \quad (6)$$

Figure 5 applies this model to demonstrate the impact of redundant advertising on data reception rate. Even at the fastest advertising frequency, redundant transmissions allow devices to overcome poor packet reception rates.

Redundancy is not always beneficial, however. To preserve the same data update rate, devices sending redundant advertisements must advertise more frequently. This increases transmission contention, which raises the probability of collision, and may defeat the goal of more reliable receptions.

Consider an application that wishes to reliably transmit data each second. Is it more reliable for each node to send five copies (once every 200 ms) or to minimize potential contention and send only one copy (once every 1000 ms)? Figure 6 shows how the expected data reception rate for these scenarios responds as network density grows. Advertising redundantly is initially more successful than advertising slowly but at 432 devices in a deployment there is a crossover. With more devices, the added contention from redundant packets actually reduces data reliability. Since we are considering deployments where all devices are within range of each other, more than 432 is unlikely. For practical network scenarios, redundant transmissions are the better choice for reliability.

The worst case for BLE collisions occurs when an event triggers data transmission, a common architecture in sensing systems. Per the BLE specification, no random delay is added before the first packet. As such, transmitting devices triggered by the same event may collide on their first transmission, and do so repeatedly on each successive interval. For such triggered systems, a random delay should be added before beginning advertising to avoid this failure mode.

4.5 Advertisement Network Takeaways

We find that advertisement-based sensor networks should be capable of achieving data reception rates over 99% given

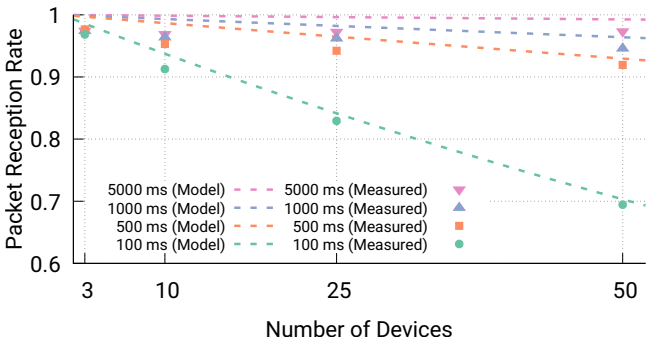

Figure 7: Analytical and experimental packet reception. Packet reception rate is measured across a range of number of transmitters and a selection of advertising intervals. Note that the y-axis starts from 0.6 PRR. We find that the analytical model tracks well with reality, but that it overestimates the true reception rate, possibly due to interference.

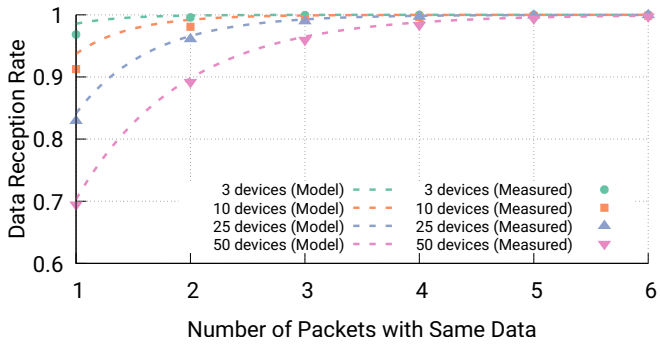

Figure 8: Analytical and experimental data reception. Data reception rate is measured across a range of number of redundantly transmitted packets for a selection of deployment sizes. Note that the y-axis starts from 0.6 DRR. Maximum sized packets are transmitted at 100 ms intervals. The experimental results closely match the analytical model.

the right parameter selection. Such networks benefit from the ease of implementation afforded by the advertisement primitive, the ubiquity of BLE radios, and the extensibility afforded by connections for infrequent maintenance tasks. To maximize the performance of advertisement-based networks, advertisers should add packet redundancy. Applications demanding very high density (100 or more devices in the same broadcast domain) and sub-second latency constraints may not be well served by advertisement-based networks.

Additionally, lack of acknowledgments makes guarantees of performance probabilistic. The networks are best for applications that may need high probability of data reception, but do not require successful reception of any particular packet.

5 Empirical Analysis of BLE Networks

To validate the analytical model of advertising network performance, we test networks of up to fifty devices and compare their performance with predictions from the model. For advertising, we use a programmable beacon platform built atop the nRF51822 BLE radio [36]. All nodes are placed in a 10×5 grid with 15 cm spacing. To recover packets, we use a Bluetooth Protocol Analyzer [13] to eliminate any potential variance from the Bluetooth stack. The analyzer sits 1 m away from the grid and listens on all three advertising channels concurrently. The experiment is run in a residential building, with general interference expected from other devices such as WiFi or Bluetooth transmitters. We dwell in each configuration for ten minutes, discarding the start and end of each trace.

The experiment varies multiple configuration parameters. We sweep the number of advertising devices from 3 to 50. For each deployment size, we test advertising intervals from 100 ms to 10 s. To detect missing packets, each advertising payload is sent with a monotonically increasing sequence number. The remainder of the packet payload is padded to a total payload size of 31 bytes. Packets received with invalid CRC values are considered missing.

To reduce jitter induced by receiving scan requests, all transmitters are configured to not listen for scan or connection requests. Otherwise, the presence of nearby smartphones scanning for BLE packets may influence the behavior of the experiment. As a side effect, this limits the minimum advertising interval of the network to 100 ms per version 4.2 of the BLE specification [7].

Figure 7 shows the results of measuring packet reception rate for this experiment. Each point is the average reception rate across the three advertising channels for all deployed devices. While 95% confidence intervals are calculated, they are too small to visualize. Although the model for BLE advertising should be conservative (as it ignores the capture effect), we find the model strictly overestimates performance in this experiment. We hypothesize that this is due to interference with other transmitters such as WiFi, Bluetooth Classic, or other BLE devices. This error is minimal, however, accounting for less than 5% deviation from the expected result.

Figure 8 adapts the raw packet reception information to estimate data recovery rate with redundant packets. Integer division of raw sequence number by a redundancy rate (number of repeated packets) yields a new stream of possibly-redundant sequence numbers. If a sequence number is seen at least once in this adapted stream, we count that data item as having been received. For a redundancy rate of one, the results are identical to the 100 ms interval line from Figure 7. Adding even modest redundancy quickly results in 100% data reception rates. Again, we find that the experimental data tracks the analytical model.

6 Real-World Deployments

While modeling the behavior of a network is valuable for informing system designers, there is no substitute for measurements of a real-world deployment. A recent power meter deployment provides us an opportunity to analyze long-running in-situ BLE networks that have collected useful data.

PowerBlade is a research power meter that measures voltage and current waveforms in real time, transmitting the data via BLE advertisements [9]. Its design results in a miniscule energy budget, preventing use of mesh network architectures. BLE also allows deployers to easily label devices and users to observe power draw of metered appliances on a phone app.

Location	1	2	3	4	5	6	7	8	9
Duration (Days)	66	168	7	110	87	83	40	24	23
# of PowerBlades	68	84	12	23	37	35	29	21	26
Expected PRR (%)	63.5	57.0	92.8	86.1	78.3	79.4	82.7	87.3	84.4
Expected DRR (%)	96.8	94.5	100	99.8	99.3	99.4	99.6	99.8	99.7

Table 1: PowerBlade deployments. 355 BLE power meters are deployed in 9 locations. Given the network configuration, our models predict data reception greater than 99% for most locations. This provides an opportunity to compare real-world network performance to theoretical expectations.

PowerBlade has been deployed in 8 residential homes and a commercial office to study contribution of various loads. Table 1 details the 9 deployments, which include 335 devices over 608 deployment-days and result in 1.5 billion recorded measurements. We study the data from these deployments to evaluate performance of their BLE advertisement networks and test the efficacy of our models on a real-world dataset.

In each location, one or more gateways are deployed to collect measurements. The gateway is a BeagleBone Black running Linux with an attached USB Bluetooth dongle [5]. It runs a simple NodeJS application atop the Noble Bluetooth library [32] to parse advertisements into data packets. Packets are de-duplicated and sent to a cloud database. This data can be used to measure data reception rates for the deployments.

PowerBlade records a new measurement once per second. It transmits with an advertising interval of 200 ms, sending 4 redundant packets for each measurement. The 5th packet each second is an Eddystone beacon [15] that links to a phone app to interact with the device. Each measurement has an attached sequence number, which helps to detect missed data. The data payload is 27 bytes and Eddystone is 26 bytes. PowerBlade also places its name, a 12-byte payload, in the scan response, to allow phones to identify it. This causes more collision by increasing packet transmission time when a scan request is received. While collection of scan response is optional, the gateways' BLE library does not allow disabling scan requests.

6.1 Expected Reception Rates

We apply models from Section 4 to predict performance. We use an average t_{adv} of 342.4 µs and $T_{adv\_interval}$ is 200 ms. Number of redundant transmissions is 4. We assume all devices in a deployment are within transmit range of each other.

While the actual probability of a scan request occurring is unclear due to scanner backoff policies, we can assume a worst case in which every advertisement has a corresponding scan request which is properly received and results in a scan response. We can model this worst case as an extension to the duration of the collision window. So rather than $2 \times t_{adv}$, the numerators of Equations (4) and (5) become $2 \times t_{adv} + 2 \times t_{IFS} + t_{scan\_req} + t_{scan\_resp}$ where t_{IFS} is inter-frame spacing of 150 µs, t_{scan_req} is scan request on-air duration of 176 µs, and t_{scan_resp} is PowerBlade's scan response duration of 224 µs.

Accounting for scan requests and responses in this manner, Table 1 lists expected packet and data reception rates for a network the size of each deployment. While the deployment size is large enough to produce more than 40% packet error in the worst case, all deployments are expected to have data reception rates above 90% and most expect above 99%.

Figure 9: Data reception rate by deployment location. While expected DRR is greater than 99% for the majority of locations, we find that locations receive 50–80%. Revised DRR loosely accounts for loss due to infrastructure failure by liberally removing gaps in reception longer than 1 hour.

6.2 Measured Data Reception Rates

We can determine the data reception rate for each deployment location by observing the sequence number in each measurement, counting how many unique measurements were received and dividing by how many were expected. As shown in Figure 9, data reception is much lower than expected. The best performance is in Location 4 with 83.4% DRR.

To attempt to identify why the network is underperforming, we measure the data loss pattern. Figure 10 shows the probability of occurrence for increasing lengths of missed measurements for each device in the deployment. As run length increases, the probability falls off, however there is a relatively high occurrence of long runs of dropped measurements. It is possible that such periods represent infrastructure failures rather than problems with devices. This includes problems like gateway crashes and network outages that are all too common in real world deployments [19].

To investigate whether infrastructure failures could be the source of most of the packet loss in these deployments, we omit packet loss that accounts for large gaps in data. The revised DRR in Figure 9 shows each result if we discount all gaps in data of an hour or more. Upon investigation, Locations 1 and 3 each have multi-day gaps during their deployment period, which are likely true outages. However, even liberally removing all 1-hour gaps does not yield satisfactory results, which suggests that another factor is at fault.

6.3 Gateway Analysis

Another cause of packet loss could be problems in the receiver chain. It is possible that packets are being dropped by the USB dongle hardware or firmware, in the Linux BLE stack, or in the Noble BLE library. To test for this, we can compare the performance of various gateway configurations compared to the professional scanner in a controlled setting.

Re-using the experimental setup from Section 5, we place twenty-five nRF51822 BLE beacons in a single room. At five different advertising intervals, we simultaneously record received packets on a sniffer and other gateway designs placed next to each other. Table 2 shows percent error from the theoretical model for the gateways at each advertising interval.

"Original Gateway" is the configuration used in the original PowerBlade deployments, a USB BLE dongle with the

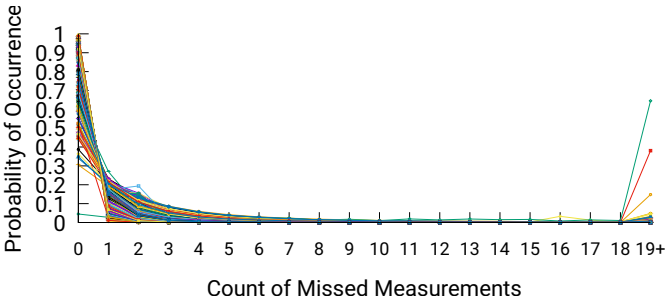

Figure 10: Measurement loss probability. For each received measurement, we count the number of immediately preceding measurements that were missed. A line is plotted for each device in the deployment. The most common count is 0 (last measurement was received). Brief streaks of 1 to several missed measurements are not rare. Very long gaps—19+ missed—suggest possible infrastructure or device failure.

Advertising Interval (ms)	100	500	1000	5000	10000
	Deviation from Theoretical Performance (%)				
Original Gateway, Noble	-46.6	-42.8	-39.5	-36.4	-34.4
Modified Gateway, Noble	-32.9	-27.2	-23.9	-21.3	-18.3
nRF52DK Gateway, Noble	-33.3	-22.6	-17.1	-14.4	-12.3
ESP32 Serial	-8.3	-14.2	-12.9	-12.4	-11.6
nRF52DK Serial	-4.3	-9.9	-8.9	-8.5	-7.8
Professional Sniffer	-1.3	-3.0	-2.7	-3.4	-3.1

Table 2: Additional packet loss in tested receivers. For a series of advertising intervals, all packets from 25 BLE beacons are recorded by: a BLE gateway identical to those in the PowerBlade deployment, a modified gateway with increased scan interval, a gateway with different BLE hardware, an ESP32 BLE scanner over serial, a nRF52DK BLE scanner over serial, and a professional BLE sniffer. Reception is most improved by avoiding the Linux BLE stack and Noble library and instead collecting with a scanning microcontroller. But all configurations yield packet loss above the theoretical amount.

Noble BLE library. We find that this gateway configuration performs far more poorly than the professional scanner with a 30 to 40 percentage point lower reception rate. A low scan interval on the gateway could result in frequent scanning gaps while channel-switching [38]. "Modified Gateway" increases the scan interval from 10 to 100 ms, resulting in some improvement, but still 20–30% loss. Another issue might be the receiver hardware or firmware [49]. Using an open-source HCI stack implementation [27], we use a nRF52 development kit to receive packets for the Noble library with an increased scan interval. "nRF52DK Gateway" shows these results, which still demonstrate loss of 10–30%. The most successful technique tested is to avoid the Noble library and Linux BLE stack altogether. We investigate the nRF52 development kit and the ESP32, both microcontrollers combined with BLE radios. For the nRF52, we find 7.88% additional packet loss on average from the theoretical model.

The particular choice of hardware and libraries used in the PowerBlade deployment gateways, while the straightforward path, introduces the most packet loss of any configuration tested, likely accounting for many of the packet reception problems discovered. Here we find that the ubiquity of BLE is a double-edge sword. Gateways are easy to build due to available commercial BLE dongles and installable radio stacks, but the presence of multiple, layered, opaque stacks makes tracking down the source of possible losses difficult.

7 Statically Planned Deployments

While the results from unplanned BLE deployments fall below expectations, combining knowledge of BLE advertisement models with the experimentally determined packet loss expected at the gateway can allow deployment success to be estimated. We demonstrate this in two additional deployments, planned in advance to see if reception rates improve.

7.1 Revising Deployment Parameters

We can adjust the data reception model to account for packet loss at the gateway similar to interference. Equation (7) shows the probability of receiving data sent M times where P_l is probability of packet loss at the gateway, P_c is probability of a collision, and P_{rc} is probability of a repeat collision.

$$DRR = 1 - [1 - (1 - P_c)(1 - P_l)]\,[1 - (1 - P_{rc})(1 - P_l)]^{M-1} \quad (7)$$

The first deployment we plan targets a 99% data reception rate and is referred to as the "99%" deployment. Assuming 7.88% loss at the gateway (average for nRF52DK), 25 devices deployed, and new data once per second, we can solve for expected data reception rate given a number of redundant packets per second. Sending 3 per second (333 ms interval) should result in a reception rate of 99.6%. For the second deployment, we target a deployment with 80% data reception and refer to it as the "80%" deployment. With the same number of devices and expected loss at the gateway, an interval of 112.5 ms with no redundancy should lead to this outcome.

25 PowerBlades are deployed throughout an apartment, attached to devices and plugged into outlets. A gateway using a nRF52 development kit over a serial connection is deployed at the intersection of the kitchen and living room. Nodes in the bedroom are the furthest away from the gateway, transmitting through a wall for the shortest-path link. The PowerBlades are first configured for the "99%" deployment and data is collected for 24 hours. Following that, devices are reprogrammed for the "80%" deployment and then replaced in the same locations for an additional 24 hours.

7.2 Deployment Results

The results of the real-world deployments are messier than the managed setup used for empirical testing, but far more predictable than the unplanned BLE deployment as we minimize and account for gateway issues. Figure 11 shows the results of the "99%" deployment. Out of the 25 deployed devices, 17 reach 99% data reception, with another 4 greater than 90% and a median of 99.6%. Devices in the bedroom are furthest away from the gateway and also have the highest prevalence of weaker than expected DRR. Devices 19 and 20 are physically close to the gateway, but positioned behind a microwave, which likely results in weak signal strength.

Without changing the locations of deployed devices, the "80%" deployment, which has a target data reception rate of 80%, obtains a measured median of 78.5%. Eight devices

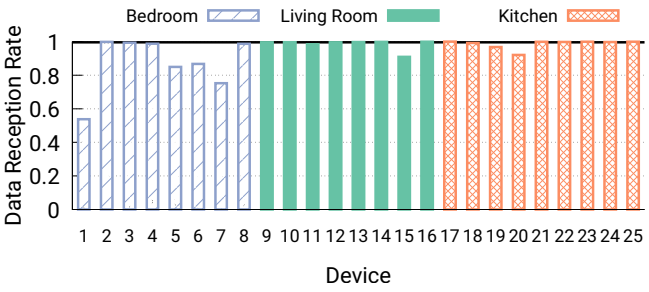

Figure 11: DRR for each device in "99%" deployment. A gateway is deployed between the living room and kitchen of an apartment, with an adjacent bedroom through a wall. Expected DRR is marked with a black line. Over two-thirds of the devices reach 99% data reception. Some devices, especially those further away, suffer degraded performance due to poor connectivity.

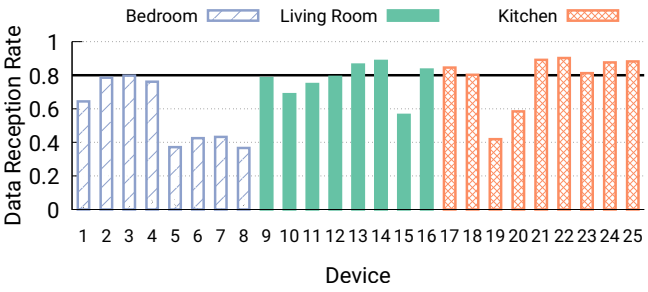

Figure 12: DRR for each device in "80%" deployment. Over half of the devices meet the target rate of 80% (black line). Devices close to the gateway perform better than expected, likely due to the capture effect. Without changing device locations, resulting DRR is poorer than in the "99%" deployment as expected due to more frequent packet collisions without redundant transmissions.

have rates within the range of 76–84%, as displayed in Figure 12. Seven more devices have better than predicted DRR, between 84–90%. Based on received device signal strength, capture effect is the likely cause. Prior work evaluates BLE capture effect suggests that stronger colliding packets can be received successfully at any point during the weaker packet (not just if transmitted during the preamble) as long as the packet is 14 dB higher in strength [39]. We find that in our "80%" deployment, all devices above 84% reception are above this threshold in comparison to at least two other deployed devices. Device 22 is the strongest case of this, with its transmissions averaging 14 dB higher than 12 other devices.

Neither deployment was fully successful in achieving target reception rates for all devices. Though reduced reception does correlate with distance, determining in advance which devices will perform poorly is difficult. Even between the two planned deployments we find that which devices do poorly changes (e.g. devices 1, 8, and 19). Devices 1 and 2 are also inches from each other, but have very different results in the "99%" deployment. We hypothesize that these issues are due to the antenna gain pattern, with slight adjustments in placement resulting in poor signal and fading effects. In these cases, a second gateway could be deployed, which would improve reception rates without any changes to deployed devices.

8 Adapting to the BLE Environment

In the prior examples, we statically configured advertisement parameters based on our expectations of transmission contention. In many real-world scenarios however, the number of communicating devices is often not known in advance. This is especially true for long-term deployments, where devices may be added or removed over time. Automatically adapting to the environment would allow a device to ensure reliable communication under any scenario, while also minimizing transmissions and therefore conserving energy. We implement an approach for adaptation, named ReliaBLE.

8.1 Measurement

The first step of adapting to the environment is measuring it. Many BLE radios support dual mode operation, where a device can be both an advertiser and a scanner. By entering scanner mode, the device can receive all valid BLE advertisements being broadcast around it. Doing so for a certain period will give a good sense of what the transmission contention looks like. Usefully, the ability to do a brief scan is part of the BLE API on microcontrollers and smartphones, meaning this method will work on almost all devices.

While scanning is a much higher energy cost than transmitting, periodically performing a brief scan for advertisements can be a low energy cost if rare and brief enough. Scanning for one second every ten minutes increases an nRF51822's average power draw by about 70 µW—roughly the same cost as increasing advertisement rate by one packet per second. This may be a reasonable price to pay for applications expecting rapid changes in the environment. A daily or weekly scan rate would further diminish energy impact.

8.2 Adaptation

Two steps are needed for ReliaBLE adaptation: estimation of collision probability and determination of redundancy. Given a reasonably accurate measurement capability, the receptions can be transformed into an estimated packet reception rate. It can then be used, along with a desired data reception rate, to select transmission redundancy.

As calculations will be performed on microcontrollers, some simplifying assumptions are helpful. First, we can utilize the naïve model for packet collisions from Section 4.4. This will increase error slightly, but is unlikely to majorly impact redundancy selection. For another simplification, we can measure packets per second, rather than actual devices and advertisement intervals. The difference in collision rate between sending, for example, 200 packets from one device or 2 packets each from 100 devices, is around 1%. Packet duration is more important for determining collisions, but packet size is provided to higher layers of most BLE libraries.

Combining these simplifications, Equation (4) can be revised to a product across all received packets of the device's own on-air duration (t_{adv_0}) plus the received on-air duration (t_{adv_i}), divided by the scan duration (t_{scan}).

Algorithm 1 Number of redundant transmissions for a DRR.

```
1: procedure CalculateRedundancy(estimatedPRR, desiredDRR)
2:     drr ← estimatedPRR
3:     advCount ← 1
4:     while drr < desiredDRR do
5:         drr ← 1 − ((1 − drr) ∗ (1 − estimatedPRR))
6:         advCount ← advCount + 1
7:     end while
8:     return advCount
9: end procedure
```

Transmissions per Second		1	2	3	4	5	6	7
Desired DRR	90%	140	252	276	274	265	253	241
	95%	68	168	203	212	211	206	200
	99%	13	70	107	126	134	138	138

Table 3: Number of devices supported at a desired DRR. As more devices are added, each must increase transmissions to maintain reliability. At a certain threshold (shaded), more transmissions reduce reliability. Deployments with fewer devices than this in a single broadcast domain will be stable.

$$estP_c = 1 - \prod_{i=1}^{Packets} \frac{t_{adv_0} + t_{adv_i}}{t_{scan}} \quad (8)$$

After calculating an estimated probability of collision, a desired data reception rate can be used to determine redundancy configuration. Algorithm 1 demonstrates the steps for doing so. Eliding repeated collisions, DRR is a product of PRR for each redundant transmission. This algorithm can be used on microcontrollers with fixed point arithmetic without major loss in accuracy. An upper limit on redundancy can also be imposed based on maximum acceptable energy use.

A concern with automatic adaptation is stability when many devices employ the same algorithm. There are regions of stability where a number of devices can all be following the same algorithm and meet the same reliability. Adding enough devices will push the entire deployment to add an additional packet per second. Eventually, this can cascade to the point where increasing transmissions harms overall reliability. The number of devices this occurs at depends on the desired DRR.

Table 3 demonstrates maximum deployment size and stable regions for 90%, 95%, and 99% desired DRR. For example, 80 devices can achieve 99% DRR at 3 transmissions per second. 140 devices cannot stably meet 99% DRR, but can yield 90% or 95% DRR with 1 to 2 transmissions per second. High reliability deployments of over 100 devices in a single area may need to investigate other protocols.

8.3 Experimental Results

To test, we implement the algorithm on an nRF51822. The adapting device starts by transmitting one BLE advertisement per second. Every ten minutes, it acts as a BLE scanner for one second. After each scan, it adjusts its transmission rate to reach 99% DRR. We also account for expected gateway losses when calculating packet loss as discussed in Section 7.1.

The environment the adapting device is deployed in has 20-100 BLE advertisements per second (background noise from nearby iOS devices). We additionally enable up to 49 other BLE transmitters, all placed within two meters of the adapting device, in order to increase noise in the environment. These transmitters are changed between ten-minute scan periods and transmit ten packets per second each. A gateway is placed within two meters of the adapting device to collect results.

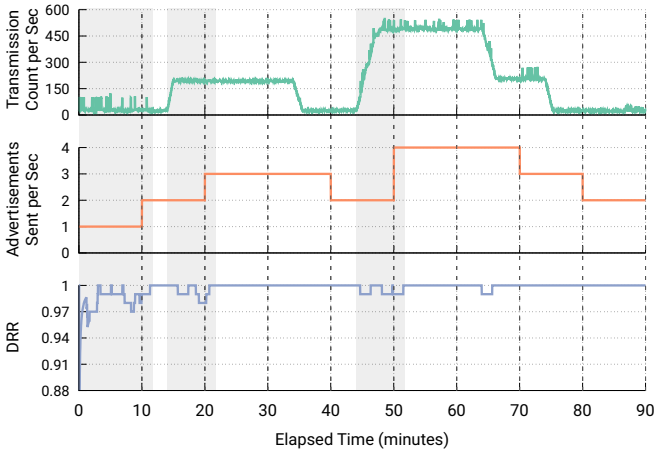

Figure 13: Runtime adaptation to the BLE environment. Over 90 minutes, the number of advertisements transmitted per second in an environment varies from 20 to over 500. An adapting device is deployed, scanning and modifying its behavior every ten minutes. Recorded are the number of advertisements it sends per second and the data reception rate (as a running average of the last 100 seconds) for those advertisements. The shaded regions are periods when the adapting device underestimates transmissions in the environment and poorer performance is likely. The next time the device scans the environment, it increases redundancy to account for the heavier traffic and maintain 99% DRR. Similarly, when it overestimates the environment, it reduces redundancy to save energy. Adaptation allows the device to maintain reliability even when the environment changes by an order of magnitude.

Figure 13 demonstrates the results from this test. The top plot (in green) visualizes the number of BLE transmissions per second in the environment. The minor count spikes throughout are due to nearby iOS device activity. At 15 minutes, 20 transmitters are added to the environment. At 35, they are disabled. Then, from 45–65, all transmitters are enabled.

The middle (orange) and bottom (blue) plots respectively display the number of advertisements per second sent by the adapting device and the resulting data reception rate for the adapting device. Vertical dashed lines every 10 minutes denote when the adapting device scans the environment and then adapts its transmission rate. Gray regions from 0–12, 15–22, and 45–52 minutes mark points where the adapting device has underestimated the environment and poor performance is expected. DRR shown is a running average of 100 seconds.

Initial background traffic is enough to warrant transmission of 2 packets per second to maintain 99% DRR. With more than 500 transmissions at once, 4 packets are necessary. Sending only 3 would have resulted in 95% DRR. Sending 2 packets or a single packet would have resulted in 86% or 63% reception rates respectively. As the environment reduces in contention, the adapting device follows that as well, eventually returning to 2 transmissions per second by the end.

9 Conclusions

In this paper, we investigate BLE advertisement networks—useful for low-power IoT due to their simplicity. The models we describe and validate allow for a theoretical understanding of the performance a network can expect. Moving into the real world, we demonstrate the power of these models to measure whether existing deployments are working as expected and to select network parameters to meet a desired performance. We explore the dataset from a large deployment of BLE power meters, finding problems in the BLE gateway that cause poor reception. In new deployments, we show that with redundancy, BLE advertisements can be used to create a network achieving 99% data reception. Further, we develop and implement the ReliaBLE adaptation protocol, which dynamically adjusts redundancy to maintain 99% data reception even with drastic real-time changes in network contention.

Not all issues faced by advertisement networks are solved. Connectivity is a challenging deployment problem and is exacerbated by BLE's short range, necessitating the deployment of smarter, ubiquitous gateways to solve it. This paper, however, provides an important first step towards BLE networking for reliable, interoperable, and ubiquitous sensor networks.

10 Acknowledgments

This work was supported in part by the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA. This material is based upon work supported by the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE) under the award number DE-EE0008220. This work is also supported by the National Science Foundation under grants CNS-1824277, CNS-1822332, and DGE-1256260. We thank Sam DeBruin and Pat Pannuto for their invaluable support with this work, and the reviewers for their feedback.

11 References

[1] N Abramson. The ALOHA system: another alternative for computer communications. *AFIPS'70 (Fall)*, 281–285. ACM, 1970.
[2] Android. Scan settings. https://developer.android.com/reference/android/bluetooth/le/ScanSettings.html, 2022.
[3] Apache. NimBLE. https://mynewt.apache.org/latest/network/, 2019.
[4] Apple. Accessory design guidelines for devices. https://developer.apple.com/accessories/Accessory-Design-Guidelines.pdf, 2022.
[5] BeagleBone. Beaglebone black. https://beagleboard.org/black, 2021.
[6] J Bicket, D Aguayo, S Biswas, and R Morris. Architecture of an unplanned 802.11b mesh network. *MobiCom'05*, 31–42. ACM, 2005.
[7] Bluetooth. Core 4.2. https://bluetooth.com/specifications/specs, 2014.
[8] S Darroudi and C Gomez. Bluetooth low energy mesh networks: A survey. *Sensors*, 17(7):1467. MDPI, 2017.
[9] S DeBruin, B Ghena, Y Kuo, and P Dutta. Powerblade: A low-profile, true-power, plug-thru energy meter. *SenSys'15*, 17–29. ACM, 2015.
[10] A Dongare, R Narayanan, A Gadre, A Luong, A Balanuta, S Kumar, B Iannucci, and A Rowe. Charm: Exploiting geographical diversity through coherent combining in LPWAN. *IPSN'18*, 60–71. IEEE, 2018.
[11] R Eletreby, D Zhang, S Kumar, and O Yağan. Empowering LPWANs in urban settings. *SIGCOMM'17*, 309–321. ACM, 2017.
[12] R Faragher and R Harle. Location fingerprinting with Bluetooth low energy beacons. *J-SAC*, 33(11):2418–2428. IEEE, 2015.
[13] Frontline. BPA. http://fte.com/products/BPAlowenergy.aspx, 2020.
[14] B Ghena, J Adkins, L Shangguan, K Jamieson, P Levis, and P Dutta. Challenge: Unlicensed LPWANs are not yet the path to ubiquitous connectivity. *MobiCom'19*. ACM, 2019.
[15] Google. Eddystone. https://github.com/google/eddystone, 2018.
[16] A Harris, V Khanna, G Tuncay, and R Kravets. Smart LaBLEs: Proximity, autoconfiguration, and a constant supply of Gatorade (TM). *SEC'16*, 142–154. IEEE, 2016.
[17] A Harris, V Khanna, G Tuncay, R Want, and R Kravets. BLE in dense IoT environments. *Communications Mag*, 54(12):30–36. IEEE, 2016.
[18] Á Hernández, D Cerio, A Valdovinos, and J Valenzuela. Proposal and evaluation of BLE discovery process based on new features of Bluetooth 5.0. *Sensors*, 17(9):1988. MDPI, 2017.
[19] T Hnat, V Srinivasan, J Lu, T Sookoor, R Dawson, J Stankovic, and K Whitehouse. The hitchhiker's guide to successful residential sensing deployments. *SenSys'11*, 232–245. ACM, 2011.
[20] W Jeon, M Dwijaksara, and D Jeong. Performance analysis of neighbor discovery process in BLE networks. *TVT*, 66(2):1865–71. IEEE, 2017.
[21] C Julien, C Liu, A Murphy, and G Picco. BLEnd: Practical continuous neighbor discovery for Bluetooth LE. *IPSN'17*, 105–116. ACM, 2017.
[22] H Kim, J Lee, and J Jang. BLEmesh: A wireless mesh network protocol for Bluetooth devices. *FiCloud'15*, 558–563. IEEE, 2015.
[23] M Kohvakka, M Kuorilehto, M Hännikäinen, and T Hämäläinen. Performance analysis of ieee 802.15.4 and zigbee for large-scale wireless sensor network applications. *PE-WASUN'06*, 48–57. ACM, 2006.
[24] R Kravets, A Harris, and R Want. Beacon trains: blazing a trail through dense BLE environments. *CHANTS'16*, 69–74. ACM, 2016.
[25] F Lin, Y Kuo, J Hsieh, H Tsai, Y Liao, and H Lee. A self-powering wireless environment monitoring system using soil energy. *Sensors Journal*, 15(7):3751–3758. IEEE, 2015.
[26] J Linde. Advertising and scanning states in BLE. https://lists.apple.com/archives/bluetooth-dev/2012/Feb/msg00037.html, 2012.
[27] Linux Foundation. Zephyr project. https://zephyrproject.org, 2022.
[28] J Liu, C Chen, Y Ma, and Y Xu. Adaptive device discovery in Bluetooth low energy networks. *VTC'13 (Spring)*, 1–5. IEEE, 2013.
[29] J Liu, C Chen, Y Ma, and Y Xu. Energy analysis of device discovery for Bluetooth low energy. *VTC'13 (Fall)*, 1–5. IEEE, 2013.
[30] LoRa. V1.1 specification. https://resources.lora-alliance.org, 2017.
[31] B Lucia, V Balaji, A Colin, K Maeng, and E Ruppel. Intermittent Computing: Challenges and Opportunities. *SNAPL'17*, 8:1–8:14, 2017.
[32] S Mistry. Noble. https://github.com/noble/noble, 2018.
[33] M Mohammad, X Guo, and M Chan. Oppcast: Exploiting spatial and channel diversity for robust data collection in urban environments. *IPSN'16*, 1–12. IEEE, 2016.
[34] B Nahas, S Duquennoy, and O Landsiedel. Concurrent transmissions for multi-hop Bluetooth 5. *EWSN'19*, 130–141. Junction, 2019.
[35] P Narendra, S Duquennoy, and T Voigt. BLE and 802.15.4 in the IoT: Evaluation and interoperability. *InterIoT'15*, 427–438. Springer, 2015.
[36] Nordic. nRF51822 datasheet. http://infocenter.nordicsemi.com/pdf/nRF51822_PS_v3.1.pdf, 2014.
[37] Nordic. Radio. https://infocenter.nordicsemi.com/topic/com.nordic.infocenter.nrf52832.ps.v1.1/radio.html#unique_576569782, 2021.
[38] D Pérez, Á Hernández, J Valenzuela, and A Valdovinos. Analytical and experimental performance evaluation of BLE neighbor discovery including non-idealities of chipsets. *Sensors*, 17(3):499. MDPI, 2017.
[39] C Roest. Enabling the chaos networking primitive on Bluetooth LE. Master's thesis, Delft University of Technology, 2015.
[40] B Spockeli. Devzone. https://devzone.nordicsemi.com/f/nordic-q-a/25553/nrf52832-scan-channel-switch-time/100747#100747, 2017.
[41] M Spörk, C Boano, M Zimmerling, and K Römer. BLEach: Exploiting the full potential of IPv6 over BLE in constrained embedded IoT devices. *SenSys'17*, 15–29. ACM, 2017.
[42] K Srinivasan, P Dutta, A Tavakoli, and P Levis. An empirical study of low-power wireless. *TOSN*, 6(2):16. ACM, 2010.
[43] R Szewczyk, J Polastre, A Mainwaring, and D Culler. Lessons from a sensor network expedition. *EWSN'04*, 307–322. Springer, 2004.
[44] Tile. App and bluetooth tracker device. https://thetileapp.com, 2022.
[45] R Trüb and L Thiele. Increasing throughput and efficiency of LoRaWAN class A. *UBICOMM'18*, 54–64. IARIA, 2018.
[46] R Want, B Schilit, and S Jenson. Enabling the Internet of Things. *Computer*, 48(1):28–35. IEEE, 2015.
[47] L Yerva, B Campbell, A Bansal, T Schmid, and P Dutta. Grafting energy-harvesting leaves on a sensornet tree. *IPSN*, 197–208. ACM, 2012.
[48] B Yu, L Xu, and Y Li. Bluetooth low energy (BLE) based mobile electrocardiogram monitoring system. *ICIA'12*, 763–767. IEEE, 2012.
[49] T Zachariah, N Jackson, and P Dutta. The Internet of Things still has a gateway problem. *HotMobile'22*, 109–115. ACM, 2022.
[50] T Zachariah, N Klugman, B Campbell, J Adkins, N Jackson, and P Dutta. The Internet of Things has a gateway problem. *HotMobile'15*, 27–32. ACM, 2015.