Enabling High-Goodput Backscatter Communication with Commodity BLE

Maoran Jiang[†*], Yunyun Feng[†*], Amiya Nayak[‡], and Wei Gong[†**]
[†]University of Science and Technology of China; [‡]University of Ottawa
mrjiang@mail.ustc.edu.cn, yunyunf@mail.ustc.edu.cn, anayak@site.uottawa.ca, weigong@ustc.edu.cn

***Abstract*—Recently, backscatter technology has attracted much interest in wireless communication due to its novel low-cost and battery-free design. Since Bluetooth Low Energy (BLE) was born to be low energy consumption, great efforts have been made in BLE-based backscatter systems, like FreeRider, RBLE. In this paper, we present BonusBlue, a BLE backscatter system that enables high-goodput communication with commercial BLE devices. BonusBlue tag generates BLE packets by modulating data on excitation signals and set up a data connection with BLE receiver using a state machine, thus achieving a high-goodput communication link. We present this state machine design and build a prototype of our tag using an FPGA, and evaluate its performance with BLE devices. Our evaluation shows that the backscatter tag can build a robust data connection link with commodity BLE device in the guidance of our state machine and transmit tag data on this link. Experimental results show that our backscatter system can achieve a goodput of up to 16.9 kbps.**

I. Introduction

Recent years have witnessed a dramatic development in the field of Internet-of-Things (IoT). It depicts a world with tremendous mobile sensors deployed in all walks of life. During this progress, there is an increasing demand for cheaper sensor nodes that can maintain a long working time. Because it is hard to replace batteries for the massive nodes deployed in the ocean.

Fortunately, backscatter technology has presented a novel low-cost and low power consumption communication design. One of the most well-known backscatter systems is RFID. A typical RFID system consists of reader and tags. RFID reader generates carrier and receives backscattered signal from RFID tag and decodes out the transmitted data. However, the high cost of dedicated readers obstructs the development of RFID. For this reason, a bunch of backscatter paradigms that leverage different carrier signals have been proposed [1]- [14]. Recent studies have shown that backscatter systems can work with ambient signals of wireless TV, visible light, Bluetooth, WiFi, ZigBee and LoRa [1], [2], [4], [5], [6], [7], [9], [10], [11], [12], [13], [14]. And the most popular branch is to modulate data on the radio signals around us, which achieves great compatibility.

Bluetooth is one of the most popular wireless technology in the market. In 2019, the total annual Bluetooth shipments have astonishingly achieved 4.2 billion [15]. There are mainly two types of Bluetooth device in today's market, Bluetooth Classic (BR/EDR) and Bluetooth Low Energy (BLE), which are incompatible with each other. Since BLE can provide considerably low power consumption, it has become a commonly used communication protocol in IoT systems. Meanwhile, BLE-based backscatter system also benefits from this and there are some distinguished works on this, e.g., Ble-backscatter [4], FreeRider [16], RBLE [17]. FreeRider utilizes codeword translation technique and modulates its data on the payload of the excitation packet. However, its unreliable modulation scheme makes backscattered packets fail to pass CRC in commercial BLE receivers. Ble-backscatter and RBLE only support direct communication with commodity BLE devices in a low rate of advertisement packets, which makes them be limited to a low-goodput transmission.

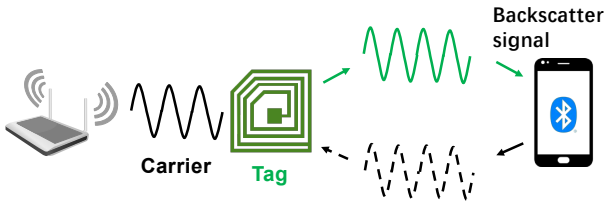


Fig. 1. BounsBlue design. The BonusBlue regenerates BLE packets from the excitation signal and fabricates a connection with a commercial BLE device using a state machine.

In this paper, we present a novel design for backscatter communication: setup a data connection between backscatter tag and BLE receiver, thus achieving a high-goodput communication. As shown in Fig. 1, when the tag receives carrier from excitation device, it modulates data on the carrier to generate new BLE packets. Then receiver will get these packets and response to them. They alternate to send packets on agreed channels.

There are mainly two challenges in building a data connection:

- BLE Spec defines the procedure of establishing data connection between BLE devices. The device that is going to be connected will first advertise on the advertisement channel and the peer device that wants to initiate a connection should send a request at a specific time after the advertising packet. The two devices primarily negotiate the connection parameters in the request

*Co-primary authors: Maoran Jiang, Yunyun Feng; **Corresponding author: Wei Gong.

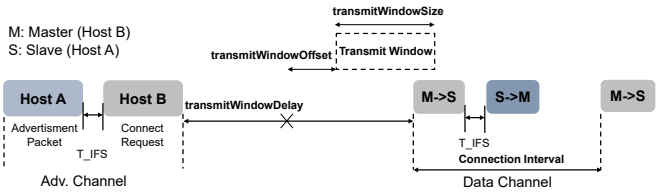


Fig. 2. Bluetooth link layer connection setup procedure timing.

packet. Then they transmit packets in different channels in different time slots. Thus the tag should follow the time and channel hopping manner defined by Bluetooth specification when building a data connection. So far, the tag is not able to initiate a data connection with commodity BLE receiver.

- BLE is a multi-layer protocol. It adopts acknowledgment mechanism and flow control in link layer. During data connection, the slave (to whom was connected) will respond to every packet sent by the master (who initiates the connection). In BLE packets, proper data fields are set to notify the peer device that the current packet is whether a new packet or an old packet. Since backscatter tag does not adopt an active radio, it cannot properly set this data field based on the response from the peer device. Thus the packets backscattered by the tag may be dropped by the link layer of BLE receiver.

We build the first backscatter system that supports building data connections with commodity BLE receivers. Our backscatter tag can initiate connection request to the BLE receiver and transmit data in different channels following the hopping manner agreed by both devices. This connection can support a high-goodput communication link.

We have made two key contributions to this design.

- **State machine for data connection.** Data connection requires the packet to be sent at the correct time window and channel. Our key solution is to build a state machine to guide tag to initiate a connection. This state machine is used to control tag's behavior and state during connection. It manages tag to transmit data in correct channel in different time slots.
- **Comprehensive evaluation.** We provide a comprehensive prototype evaluation for our high-goodput backscatter system. We compare its goodput with state-of-the-art BLE backscatter systems and also show the variability of the data connection built by our tag system. Results show that our tag can achieve a goodput of 16.9 kbps, which is 2x larger than RBLE [17]. Meanwhile, our tag is also able to adapt to different transmission rates.

II. BLE Primer

BLE has been popular in applications of healthcare, fitness, and wearable devices due to its significantly reduced power consumption. Generally, we can classify communication links

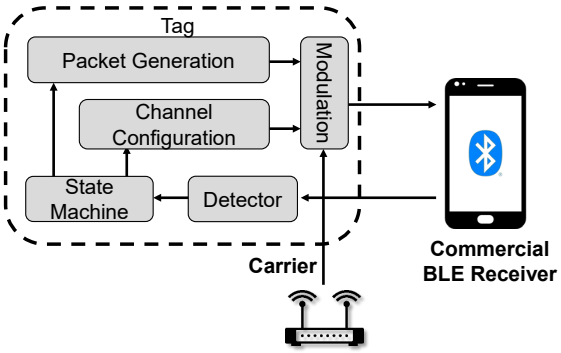


Fig. 3. BonusBlue overview. Upon receiving the signal from receiver, BonusBlue tag parses this command and does packet generation channel configuration following data connection manner. All these behaviors are controlled by a state machine.

in BLE into two types: connectionless and connection-oriented. BLE advertising is considered as a connectionless link, which is used to transmit a small amount of data. However, connection-oriented link is more common in BLE applications and it supports higher packets transmission rate. In the following, we present a typical example of the setup process of BLE connection.

How does commercial BLE devices setup connection?We illustrate this process from three parts: interactions in advertising channels, process of channel hopping, and interactions in data channels. The details are shown in Fig. 2.

Host A first periodically broadcasts data on three advertising channels, indicating that it is ready to be connected. Host B will scan these three advertising channels one by one and initiate a connection by sending a connect request immediately after T_IFS=150 μs when detecting the advertising packet.

The second part is channel hopping. Host A, to be connected, is denoted as the slave, and Host B, which initiates the request, is denoted as the master. When the slave receives the request, it then hops to the data channel and listens for packet from the master within a transmit window. This window is determined by several parameters including *transmitWindowDelay*, *transmitWindowOffset*, and *transmitWindowSize*. These parameters are either specified by BLE protocol or declared in the connect request packet. The time point at which the master transmits the packet is denoted as the anchor point, which is a time reference for the remaining actions.

The third part is about data exchange in data channels. Starting from the anchor point, the timeline is divided into several equal time slots, where each one is called connection event and corresponds to a channel index. In each connection event, the master and the slave should transmit packets in their corresponding channel. It is worth noting that the slave will send out a response packet if it receives a packet from the master. What's more, it is worth noting that BLE adopts acknowledgment mechanism. When the connection was setup, both the master and slave will maintain two local variables, and they compare them with the data field in incoming packets,

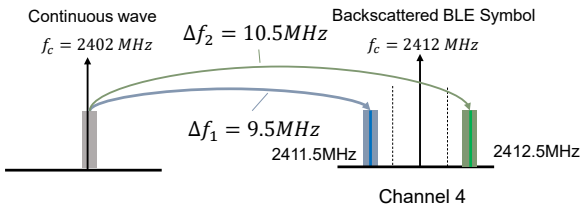


Fig. 4. Example of frequency modulation. BonusBlue tag shifts a continuous-wave to the target channel to construct a BLE symbol.

thus knowing that the packet is whether a new packet or an old one. And the device will drop the old packet. We will discuss it later.

III. System Design

In this section, we first give an overview of our system design. Then we briefly discuss the modulation scheme our backscatter tag uses, followed by the introduction of our state machine design for the data connection. Finally, the process of using redundant sequence number to confront acknowledgment mechanism is proposed.

A. Overview

Fig. 3 shows the framework of our high-goodput backscatter system. The system consists of three parts: Backsactter tag, excitation generator, and BLE receiver. Upon receiving BLE signal from the receiver, tag detects and parses the command. The state machine will receive internal command and control tag to configure the channel and generate raw packet bits corresponding to a different channel. Then the modulation module will modulate the raw bits on the carrier from excitation generator to regenerate BLE packets. The state machine makes sure the tag follows the manner of data connection of a commercial BLE device, thus a data connection can be built between the tag and the commodity receiver.

B. Modulation Scheme

We adopt RBLE's [17] frequency shift modulation scheme but use continuous-wave as excitation signal. As shown in Fig. 4, the frequency of continuous-wave is 2402 MHz, which will be reflected by RF switch of tag. We control the on-off pattern of RF switch at a specific frequency to generate a frequency shift, which can be formulated as multiplying excitation signal with a baseband signal. Since BLE adopts BFSK modulation, to generate BLE symbol '0', we use 9.5 MHz frequency clock to control the RF switch and the excitation signal will be shifted to 2409.5 MHz. And we use 10.5 MHz to generate symbol '1' in the same way. By generating this baseband signal following the raw bits of packet, the tag can reflect out a BLE packet that can be received by commercial receiver.

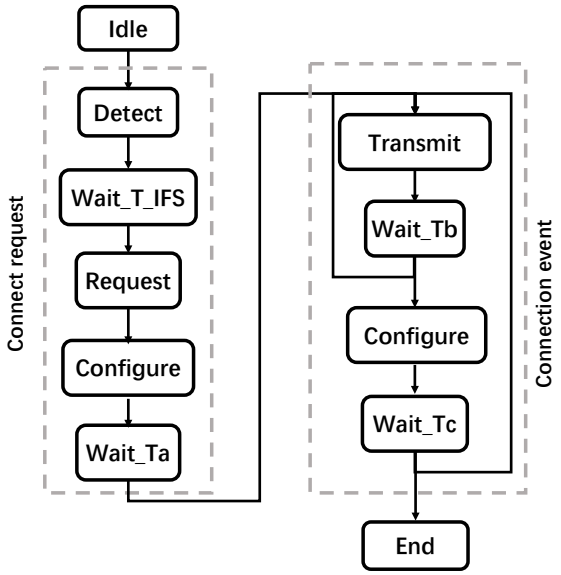


Fig. 5. State machine on tag for building a data connection. The procedure is classified into two stages: connection request and connection event. Connection request stage consists of detecting advertising packets and initiating request. Connection event stage consists of transmitting packets, waiting for responses, and configuring frequency clocks for different channels. In different waiting states, $Wait\_$ is followed with different waiting time. T_IFS = 150 μs, Ta is determined by the transmit window, Tb is determined by the time interval between consecutive packets from tag, and Tc is determined by the connection interval of each connection event.

C. State Machine on Tag

As we introduced in section II, BLE specification has defined an explicit procedure about data connection setup, which involves connection request and data exchanging in different channels. The recent work RBLE [17] first involves a channel hopping mechanism to confront interference, but it fails to satisfy the accurate hopping routine since data connection has a strict requirement of transmitting in the right channels at right time. We designed a state machine for tag to manage its radio state and timing requirements during data connection.

As shown in Fig. 5, the left dotted box is the procedure of connection request and the right dotted box is the procedure in connection event part. The tag stays at *Idle* state when it doesn't transmit data. If receiver sends a particular sequence of packets, the tag parses this command and is waked up to *Detect* state, ready to setup a connection. If the advertisement packet of receiver is detected, tag enters *WAIT_T_IFS* state to waits T_IFS = 150 μs, which is a packet interval defined by the protocol, otherwise, tag will back to *Idle* state. The tag will backscatter a request packet to receiver after T_IFS. Then it configures frequency clock and enters *Wait_Ta* state to wait time Ta for transmit window. During this process, the timer is set according to BLE Specification.

Since the tag cannot tell the differences between packets in different advertising channels, we need to make the receiver broadcast on only one channel and make it known to tag,

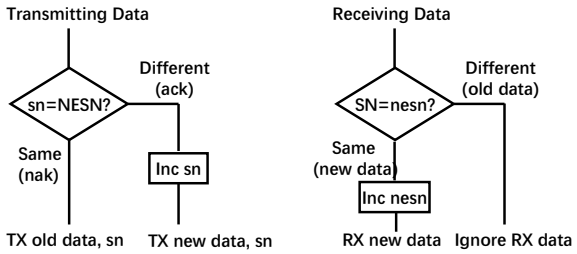


Fig. 6. Link layer acknowledgment mechanism.

thus the peer BLE device can receive the connection request from tag. When this connect request is accepted, both the receiver and tag will move to data channel. They alternate to send packets in each connection event. The tag will first be in *Transmit* state to send the first packet and waits for receiver's response in state *Wait_Tb*, and then it transmits next packet. At the end of each connection event, tag will enter *Configure* state to configure the frequency clock that controls RF switch and then enters *Wait_Tc* state to wait for the start of next connection event.

In each connection event, tag and receiver will hop to different channels. This channel sequence is determined by the channel map and channel selection algorithm, which was declared in connect request packet. There are two channel selection algorithms currently. Algorithm #1 is a simple incremental algorithm, which produces a uniform sequence of channels. Compared to Algorithm #1, #2 is more complex and it produces a randomized sequence of channels. Here we choose Algorithm #1 to reduce the complexity of implementation.

To implement the channel hopping mechanism, we cannot generate all the frequency clocks for RF switch at one time, since BLE occupies 40 channels, and RF switch needs two frequency clocks for backscattering to each target channel, which leads to 80 frequency clocks in total. Generating all these clocks at one time is unrealistic. Thus we adopt FPGA's Mixed-Mode Clock Manager (MMCM) to provide clocks for RF switch. We can dynamically change the output clock of MMCM through dynamic configuration port it provides.

It is worth noting that BLE also adopts whitening to avoid transmitting constant symbols. Whitening is an XOR operation between data bits and whitening sequence, which was generated using a whitening seed. In BLE, channel index is used to produce this whitening seed. So in each connection event, our tag must apply whitening to the raw bits of packet according to the current channel index before transmitting.

D. Redundant Sequence Number

As we mentioned above, BLE is a multi-layer protocol. In link layer, BLE device maintains several data fields to indicate that the incoming packets are new ones or old ones. The device will drop these old ones, which could be a problem in our backscatter systems. As shown in Fig. 6, when the

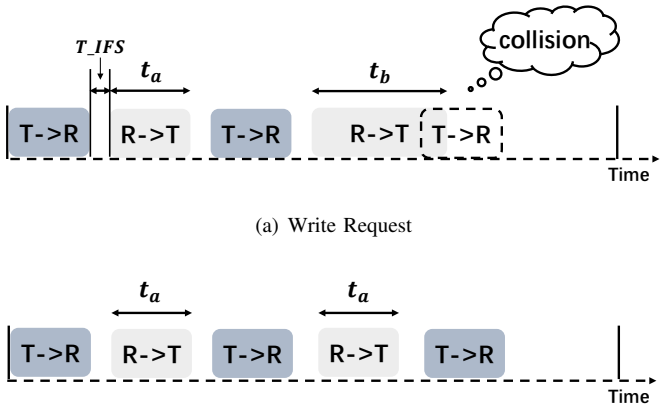
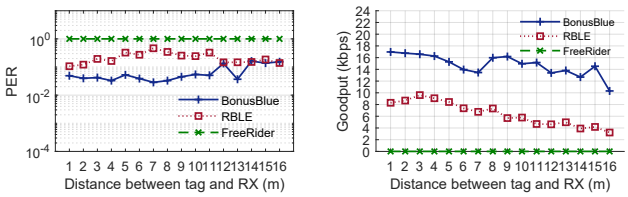


Fig. 7. Example of two different write operations in a connection event. 'T→R' denotes that it's a packet send from tag to receiver. The length of rectangle denotes the length of this packet.

connection is built up, both the master and slave will maintain two local variables, transmitSeqNum (denoted as sn), and nextExpectedSeqNum (denoted as $nesn$), initiated with bit "0". For packets, they contain two fields: SN and $NESN$. Before the master (or slave) sends a packet, it will assign sn to SN, $nesn$ to $NESN$. Upon receiving a packet, the receiver compares the $NESN$ with local sn. If they are equal, the old packet will be transmitted, otherwise the receiver increments local sn and transmits a new packet. If SN is equal to $nesn$, the receiver will accept the data, otherwise, the data will be ignored.

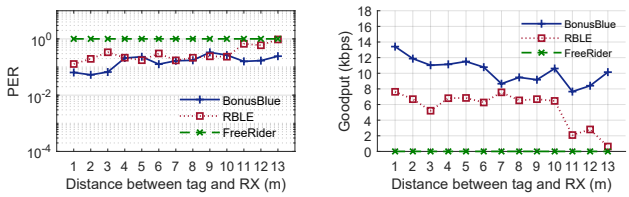
From the above procedure, we can see that the data fields (transmitSeqNum and nextExpectedSeqNum, each occupies 1 bit) of each packet are set based on the previous packet. However, backscatter tag doesn't employ an active radio and cannot decode BLE packets. So we let tag transmit multiple packets with different data fields (sequence number) in each connection event. The two data fields (transmitSeqNum, nextExpectedSeqNum) lead to 4 cases. We can simply transmit these packets with different data fields one by one in a connection event. Although our tag cannot decode the response from receiver and some packets will be dropped, there will always be a packet be recognized as new data by the receiver.

To transmit multiple packets in one connection event, we need to design the transmit time of each packet. Because in data connection, the slave will always respond with a packet when it receives a packet from the master. In each connection event, if tag knows the length of each packet from the peer device, then it can transmit packet at the correct time, which is T_IFS after the previous received packet. This can be achieved using ATT Protocol, which is used to discover, read and write the attributes (one kind of data structure) of peer device. We can make the commercial devices provide writable attributes as a server. And BonusBlue tag, as a client, will write these attributes. However, there are two different write operations, *Write Request* and *Write Command*. *Write Request* operation



(a) PER (b) Goodput

Fig. 8. Backscatter PER, and goodput across distances in LOS scenario.



(a) PER (b) Goodput

Fig. 9. Backscatter PER, and goodput across distances in NLOS scenario.

will ask for a write response, which will change the length of response packet, as shown in Fig. 7(a). When the tag transmits packet after $t_a + 2 * T\_IFS$, there would be a collision. However, *Write Command* operation doesn't ask for this write response as shown in Fig. 7(b), thus the peer BLE device will always respond with an empty packet, which has a fixed length known to the tag. And tag just transmit next packet after $t_a + 2 * T\_IFS$. And tag's packet can successfully be in the receiving window of receiver.

IV. Implementation

We build a prototype of BonusBlue tag using FPGA, off-the-shelf ICs. The implementation is detailed as follows. The backscatter tag consists of a signal detector, RF switch, and control logic. The signal detector is constructed using AD8313, and TLV3501, which is used to build a threshold tuning circuit. The RF switch is constructed using ADG902. We implement the control logic using Zynq-7010, which integrates ARM processor and Xilinx 7-series FPGA. ARM processor is used to provide application data for packets. FPGA is used to control RF switch to reflect excitation signal.

We use TI CC2650 to transmit continuous-wave at 5 dBm, and Nordic nRF52840 with an amplifier of 20 dBm gain to act as a receiver. The packet error rate (PER) is evaluated using TI SmartRF software with CC2640R2F.

V. Evaluation

In this section, we evaluate the performance of BonusBlue and compare it with prior BLE backscatter system FreeRider [16] and RBLE [17] in line-of-sight (LOS) and non-line-of-sight (NLOS) scenarios. We also conduct experiments to evaluate the variability of connection. Performance is evaluated through PER and goodput.

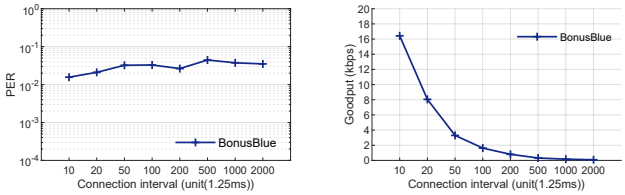
A. End-to-End Performance

To evaluate the end-to-end performance, we conduct experiments in LOS and NLOS scenarios in indoor environment. We deploy the backscatter tag 0.2m from the excitation, and then gradually move the receiver far away from the tag. In NLOS scenarios, we set a metal barrier between the tag and receiver. We measure PER and goodput at each point and modify the application of receiver (Nordic nRF52840) to make it advertise only in channel 38. Our tag will detect and send a request to the receiver in this channel. Meanwhile, for simplification, we set tag and receiver to transmit data only on data channel 14 and 28. We transmit 4 redundant packets in each connection event, and each connection event is set to 12.5 ms. Since the receiver running a BLE application will drop the CRC failed packets, we use TI CC2640R2F to sniff the packets in channel 14 and 28, and calculate the PER. The goodput of BonusBlue is measured through the printed information on the laptop connecting to the receiver.

From Fig. 8(a) and Fig. 9(a), we can see the PER of backscattered packets gradually increases as distance increases. PER of BonusBlue tag in LOS is better than that of NLOS. The maximum communication range of BonusBlue tag is 16m for LOS and 13m for NLOS. This is because to build a data connection, the tag needs to detect the advertising packet of the receiver. The signal detector we use is constructed using AD8313, which is a complete multistage demodulating logarithmic amplifier. In this indoor environment with RF noise, the signal detector can detect the packet at the maximum distance of 16m, and 13m in NLOS.

Generally, when the distance between tag and receiver (TI sniffer) is within 11 m, PERs for BonusBlue in LOS environments are lower than 10%. However in NLOS scenario, when the receiver is just 3 m away, the PER of BonusBlue rises over 10%. We notice that the PERs of FreeRider [16] in LOS and NLOS are both 100%. Because we evaluate the performance of communication with commercial BLE device. FreeRider adopts codeword translation technique. During redundant encoding, the excitation packet will be shifted to the target channel and the payload will be modified. But the CRC of backscattered packet is not set properly, so backscattered packets fail to pass the CRC check on commodity receivers. Thus PERs of FreeRider will always be 100%. The PERs of RBLE are higher than BonusBlue in LOS and NLOS scenarios to some extent. This could be caused by the difference of excitation signal. BonusBlue needs continuous-wave to be carrier, whose frequency could be more accurate, while RBLE takes the BLE packets as excitation signal.

Despite the small gap of PERs between RBLE and BonusBlue, our system has a significant advantage in goodput. As depicted in Fig. 8(b) and Fig. 9(b), the maximum goodput of BonusBlue is 16.9 kbps and 13.4 kbps in LOS and NLOS scenarios, respectively. In contrast, the counterparts of RBLE are 8.3 kbps and 7.6 kbps for LOS and NLOS cases, respectively. We notice that the goodput of FreeRider are 0 in LOS and NLOS scenarios. Because FreeRider's packets cannot pass CRC and its PERs are all 100%. So for commercial



(a) PER (b) Goodput

Fig. 10. Backscatter PER, and goodput across different connection interval.

receiver, its packets cannot be received and the goodput are all 0. Considering the maximum goodput, BonusBlue is 2.03x and 1.54x better than RBLE. This is because our tag generates more packets than RBLE, and the key point is that our tag uses data connection to communicate with receiver. Thus the receiver can receive data at such a high rate, which is not supported in advertisement. The above experiment results show that our tag can achieve a high-goodput communication link with commercial BLE receiver using data connection.

B. Adaptability of BonusBlue

Next, we evaluate the flexibility of BonusBlue tag. Our system utilizes data connection to transmit data. We measure the PERs and Goodput with different connection intervals, which are defined in unit time of 1.25 ms. We conduct experiments between 12.5 ms and 2500 ms. The receiver is fixed at 1 m from tag, and other setups are the same as the end-to-end evaluation. As depicted in Fig. 10(a), PERs under different connection interval are all below 5%. For goodput shown in Fig. 10(b), we can see that as the connection interval increases, the goodput decreases rapidly, since the transmitting rate decreases. The goodput with a connection interval of 12.5 ms is 16.42 kbps, and when connection interval is 2500 ms, the goodput is 0.082 kbps, which is almost near zero.

VI. Related work

Our work is inspired by the previous progress on Bluetooth-based backscatter systems. We will discuss the closely related work and differences between them as follows.

Interscatter [7] first transforms a commercial BLE device into a continuous-wave (CW) generator and regenerate 802.11b signals on it. However it fails to generate BLE packets. FreeRider [16] realizes a backscatter system entirely using BLE radios. However, since FreeRider adopts codeword translation, it fails to have direct interactions between backscatter tag and commercial BLE. First, it only modifies the payload of BLE packet, thus the backscattered packets cannot pass the CRC check and be dropped by commercial devices without extra modifications. Second, FreeRider needs packets in original channel to decode the tag data, which adds complexities to the BLE receivers. BLE-Backscatter [4] and RBLE [17] enable direct interaction with commercial BLE devices. But their applications are both limited in advertising. Although they can regenerate data channel packets, the commercial receivers (not in test mode) cannot receive them since BLE devices will not directly listen for packets in data channels.

VII. Conclusion

In this paper, we propose BonusBlue, a high-goodput backscatter system that works with commodity BLE device. BonusBlue is able to establish a data connection with standard BLE device using a state machine, and it also adopts redundant sequence numbers to pass the acknowledgment mechanism of receiver. We present the design of the first connection-enabled tag and build a prototype. Our evaluation shows that the tag can successfully build a data connection with a commercial BLE receiver and achieve a goodput of up to 16.9 kbps with a communication range (tag-to-receiver) up to 16 m.

Acknowledgment

This work was supported by NSFC Grant No. 61932017 and 61971390, and Hefei Healthcare Grant No. J2020Y03, and the Fundamental Research Funds for the Central Universities No. WK5290000002.

References

[1] V. Liu, A. Parks, V. Talla, S. Gollakota, D. Wetherall, and J. R. Smith, "Ambient backscatter: wireless communication out of thin air," in *Proc. of ACM SIGCOMM*, 2013.

[2] X. Xu, Y. Shen, J. Yang, C. Xu, G. Shen, G. Chen, and Y. Ni, "Passivevlc: Enabling practical visible light backscatter communication for battery-free iot applications," in *Proc. of ACM MobiCom*, 2017.

[3] D. Bharadia, K. R. Joshi, M. Kotaru, and S. Katti, "Backfi: High throughput wifi backscatter," in *Proc. of ACM SIGCOMM*, 2015.

[4] J. F. Ensworth and M. S. Reynolds, "Ble-backscatter: Ultra low-power iot nodes compatible with bluetooth 4.0 low energy (ble) smartphones and tablets,"

[5] B. Kellogg, A. Parks, S. Gollakota, J. R. Smith, and D. Wetherall, "Wi-fi backscatter: Internet connectivity for rf-powered devices," in *Proc. of ACM SIGCOMM*, 2014.

[6] B. Kellogg, V. Talla, S. Gollakota, and J. R. Smith, "Passive wi-fi: Bringing low power to wi-fi transmissions," in *Proc. of USENIX NSDI*, 2016.

[7] V. Iyer, V. Talla, B. Kellogg, S. Gollakota, and J. Smith, "Inter-technology backscatter: Towards internet connectivity for implanted devices," in *Proc. of ACM SIGCOMM*, 2016.

[8] P. Zhang, M. Rostami, P. Hu, and D. Ganesan, "Enabling practical backscatter communication for on-body sensors," in *Proc. of ACM SIGCOMM*, 2016.

[9] P. Zhang, D. Bharadia, K. Joshi, and S. Katti, "Hitchhike: Practical backscatter using commodity wifi," in *Proc. of ACM SenSys*, 2016.

[10] J. Zhao, W. Gong, and J. Liu, "X-tandem: Towards multi-hop backscatter communication with commodity wifi," in *Proc. of ACM Mobicom*, 2018.

[11] Q. Wang, S. Chen, J. Zhao, and W. Gong, "RapidRider: Efficient WiFi Backscatter with Uncontrolled Ambient Signals," in *Proc. of IEEE INFOCOM*, 2021.

[12] V. Talla, M. Hessar, B. Kellogg, A. Najafi, J. R. Smith, and S. Gollakota, "Lora backscatter: Enabling the vision of ubiquitous connectivity," in *Proc. of ACM IMWUT*, 2017.

[13] W. Gong, L. Yuan, Q. Wang, and J. Zhao, "Multiprotocol backscatter for personal iot sensors," in *Proc. of ACM CoNEXT*, 2020

[14] L. Yuan, C. Xiong, S. Chen, and W. Gong, "Embracing self-powered wireless wearables for smart healthcare," in *2021 IEEE International Conference on Pervasive Computing and Communications (PerCom).* IEEE, 2021, pp. 1–7.

[15] "Bluetooth market update" https://www.bluetooth.com/wp-content/uploads/2019/03/Bluetooth_Market_Update_2018.pdf

[16] P. Zhang, C. Josephson, D. Bharadia, and S. Katti, "Freerider: Backscatter communication using commodity radios," in *Proc. of ACM CONEXT*, 2017.

[17] M. Zhang, J. Zhao, S. Chen, and W. Gong, "Reliable backscatter with commodity BLE," in *Proc. of IEEE INFOCOM*, 2020.