On Practical Selective Jamming of Bluetooth Low Energy Advertising

Sebastian Bräuer, Anatolij Zubow, Sven Zehl
Technical University of Berlin
{braeuer, zubow, zehl}@tkn.tu-berlin.de

Mehran Roshandel, Soroush Mashhadi-Sohi
Deutsche Telekom, T-Labs
{Mehran.Roshandel, Soroush.Mashhadi-Sohi}@telekom.de

***Abstract*—Bluetooth Low Energy (BLE) Advertising is an important component of BLE as it is used to broadcast connection information to other devices nearby, usually smartphones. Furthermore, it is the foundation of a variety of services, such as indoor localization.**

We present a selective jammer against BLE beacons. The jammer selectively jams only BLE advertising beacons for which it was configured (API) on just the BLE advertising channels actually being used (narrowband). Such a jammer is hard to detect and hence requires a sophisticated Intrusion Detection System (IDS).

The prototypic implementation uses off-the-shelf embedded hardware which is low-cost, small-size and very power efficient. Experiment results demonstrate the feasibility of the proposed solution and reveal the impact of the distance between jammer and receiver on the advertising success rate. At short distances the jammer is able to successfully corrupt all BLE advertisement beacon frames making the BLE beacon source undiscoverable.

***Index terms*—** Jamming, Bluetooth Low Energy.

I. Introduction

Bluetooth Low Energy (BLE) Advertising is an important component of BLE [1]. It is the basis for so called *beacons*, small devices that periodically broadcast information to other devices nearby, usually smartphones. In combination with a smartphone app an installed beacon network can provide a variety of services, such as indoor localization [2], proximity-triggered advertising [3] or mobile payment systems [4].

The commercial interest in these applications (and therefore in BLE Advertising) is growing. By the end of 2015 there were already 4 million deployed beacons [5] and by 2020 deployment of over 400 million beacons is projected [6]. With increasing use of BLE Advertising security aspects become more important, especially on the physical layer, where all wireless communication is prone to attacks.

While attacking BLE advertising using a continuous jammer is fairly straightforward, such an approach has a number of disadvantages. First, it can easily be detected by an Intrusion Detection System (IDS). Second, it is nonselective, i.e. it jams all beacon transmissions and not only those of the beacon source to be attacked, and very energy inefficient, i.e. the wideband jamming signal is sent permanently. In contrast a *selective narrowband* jammer is arguably more sophisticated; it is very efficient and very hard to detect [7]. By jamming only selected frames an attacker can stealthy prevent those frames to reach their destination. This is achieved by decoding the frame header of each frame on the air and then deciding whether to jam the frame payload or not. However, the design and implementation of such a jammer is challenging as it requires on-the-fly frame header inspection while meeting strict timing requirements in order to make sure that a sufficient part of the remaining content is jammed. Moreover, as BLE advertising beacons are sent on different channels redundantly [1] such a jammer must be able to predict and quickly follow the channel hopping sequence used by the beacon source to be attacked. Finally, an *energy efficient* solution which can be implemented using *low cost* off-the-shelf small-sized hardware is desirable.

Contributions: In this paper we present the design of a selective jammer for BLE advertising. The proposed jammer can be programmed so that only specific beacons, e.g. those with a particular device address, are being jammed. As BLE beacons are sent on different advertisement channels the proposed jammer contains a discovery component which scans all advertisement channels in order to estimate the channels actually being used by the beacon sources to be attacked. The proposed jammer is energy-efficient and hard to detect, as a short jamming signal is only emitted during transmission and only on the channel being used by the beacon frame under attack. The proposed jammer is prototypically implemented using low-cost, off-the-shelf and small-sized hardware. By means of experiments the feasibility and efficiency with respect to two identified performance metrics is demonstrated.

The rest of the paper is organized as follows. Section II of this paper gives a short introduction into BLE Advertising. Section III presents the system model and problem formulation. Section IV gives an overview about possible jamming attacks on BLE Advertisement frames known from literature. The proposed jamming solution is described in section V. Section VI elaborates the experiment setup and testbed evaluation of the proposed jammer. Finally, the sections VII and VIII describe the related work and give a conclusion.

II. Primer BLE Advertising

This section gives a brief overview of the relevant part of the BLE protocol stack [1] with a focus on the advertising process described in the Link Layer specification as well as the BLE framing.

[1] In order to mitigate narrowband interference.

978-1-5090-3862-6/16/$31.00 ©2016 IEEE

A. BLE Physical Layer

The BLE physical layer uses the 2.4GHz (2402-2483.5 MHz) spectrum and GFSK modulation at a bitrate of 1 Mbps. BLE divides its spectrum into 40 channels, numbered 0 to 39, with a spacing of 2 MHz each (Fig. 1). Channels 37, 38, and 39 are so-called *advertisement channels* and are used for device discovery, broadcasting information and establishing a connection, whereas the remaining channels are described as *data channels* and are used for data exchange during a connection. The three advertisement channels are spread across the 2.4 GHz spectrum in order to achieve frequency diversity and being robust against interference from Wi-Fi, classic Bluetooth, Microwaves, etc.

Every frame has a known preamble followed by a known access address which can be used to synchronize and correlate against. On the three advertisement channels the access address and the preamble are fixed to enable broadcast reception.

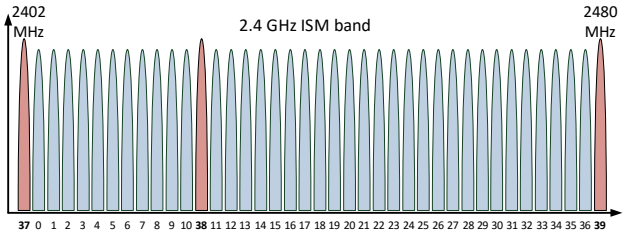

Fig. 1. BLE channelization with advertisement channels (red) and data channels (blue).

B. BLE Link Layer

The Bluetooth Low Energy Link Layer distinguishes between three basic device roles, namely, i) advertiser, ii) scanner and iii) initiator.

A device in the advertiser role will periodically broadcast frames on each advertising channel (Fig. 2). These frames may contain information about the advertiser such as services and features the advertiser supports as well as manufacturer specific information. The time interval between frames has both a fixed interval and a random delay. The fixed interval can be set from 20 ms to 10.24 s whereas the random delay is a pseudo-random value from interval [0, 10] ms. The purpose of the latter is to reduce the possibility of collisions between advertisement frames from different co-located devices. In order to save power the BLE specification allows sending advertisement frames on just one or two channels at the cost of reduced robustness against interference [2].

A device in the scanner role will passively listen for frames sent by advertisers in proximity. It may or may not request further information from an advertiser by sending a scan request (SCAN_REQ) on the same channel it received the advertiser's frame. The advertiser will respond to this by sending a scan response (SCAN_RSP) containing services that were previously unmentioned or data that did not fit in the advertising frame, e.g. the device name.

[2]Apple, for example, recommends advertising on all 3 channels, as do other manufacturers.

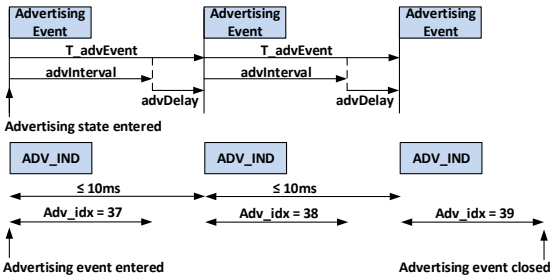

Fig. 2. Bluetooth LE advertising - during each advertising event the beacon is transmitted on all used advertising channels in ascending order.

A device that wishes to form a connection with an advertiser is called an initiator. The initiator therefore sends a connection request (CONNECT_REQ) to the advertiser. If the CONNECT_REQ is accepted by the advertiser, both devices will leave the advertising channel and form a connection as specified in the CONNECT_REQ frame.

C. BLE Advertisement Packets

The BLE specification defines the framing structure. A BLE frame consists of a preamble, an access address, a frame data unit (PDU) and a cyclic redundancy checksum (CRC). Unlike Bluetooth Classic, there is no mechanism for Forward Error Correction (FEC).

The frame data unit for the advertising channel consists of a 2-byte header and a variable payload from 6 to 37 bytes. The length of the payload is defined by the 6-bit length field in the header of the Advertising Channel PDU. The structure of the PDU is defined by its type, which is indicated in the first 4 bits of the header. All PDU types include the device address, BTADDr, of the transmitting device (which can be public or randomly chosen) in the first 6 bytes of the payload.

An advertising device can choose from 4 different PDU types depending on its functionality and intent: i) ADV_IND, ii) ADV_SCAN_IND, iii) ADV_NONCONN_IND, iv) ADV_DIRECT_IND.

The first three types all share the same format and shall only indicate whether the device will accept scan requests or connection requests. They include a field (AdvData) with a variable number of advertisement data structures (AD) up to 31 bytes. An AD structure can contain information about the services offered, transmission power levels and other characteristics. Many higher layer protocols that are built on top of BLE use the AD structures to encode their information. E.g. Apple's proprietary *iBeacon* format encodes its data into the Manufacturer Specific Data field and Google's open *Eddystone* protocol encodes its functionality into a Service Data field.

The last one (ADV_DIRECT_IND) contains no AD structures and indicates that the advertiser will only accept connections by an initiator specified in the PDU.

As previously mentioned, an advertiser can also respond to a SCAN_REQ frame sent by an active scanner with a SCAN_RSP frame, whose format is equivalent to the ADV_IND/ADV_SCAN_IND/ADV_NONCONN_IND frames. That way an advertiser can reduce the size of his advertising frames to only include basic information and deliver further information on demand , thus save energy.

III. System Model and Problem Formulation

A. System Model

We consider the basic scenario shown in Fig. 3 consisting of a single BLE beacon source emitting BLE advertisement beacons, a receiver which performs passive scanning for those beacons and a jammer whose goal is to block (jam) the beacon reception at the receiver node.

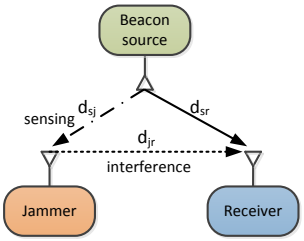

Fig. 3. System model.

B. Problem Formulation

Our objective is to minimize the overall energy consumption at the jammer node while guaranteeing the jamming constraint, i.e. successful jamming of the signal of the BLE beacon source to be attacked so that none of the BLE advertising frames transmitted can be correctly decoded at the receiver, e.g. received frames are corrupted.

Besides energy efficiency we aim for a jammer which is selective, i.e. jamming just the BLE beacons to be attacked, hard to detect by any IDS, is of low-cost and small-sized, i.e. can be inconspicuously installed for a longer time.

IV. Types of Jammer

There are several types of jammers that can be used against BLE advertising beacons. According to the generic jammer models proposed by Xu et al. [8] we can distinguish between: i) constant jammer, ii) deceptive jammer, iii) random jammer and iv) reactive jammer. Moreover, a jammer can be selective if it can be programmed to attack just specific frames. As BLE advertising beacons are sent on different channels redundantly we have to further distinguish between narrowband and wideband jamming depending on whether only a single or multiple BLE channels can be jammed at the same time.

The different jammer types vary with regard to their efficiency, power-consumption and complexity. In the following we will briefly describe them and discuss their pros and cons when used for jamming against BLE advertising beacons.

A constant wideband jammer emits noise for an indefinite amount of time over a large frequency range. Although it has a low complexity, its efficiency is low and energy consumption is high. This is because such a jammer has to jam all the three advertisement channels simultaneously which are spread over the whole 2.4 GHz ISM band (Fig. 1). Furthermore, the jammer emits the jamming signal even at times when no BLE beacon frame is transmitted. Such a jammer can be easily detected by an IDS, e.g. with a spectrum analyzer.

A constant narrow-band jammer emits the jamming signal permanently but only on a single BLE advertising channel. As only a single BLE channel can be jammed at a same time frequency hopping need to be applied.

A reactive wideband jammer is based on the observation that BLE beacons are only sent at certain points in time, e.g. every second. Therefore, it is sufficient to emit the jamming signal only during frame transmission. Hence, such a periodic jammer needs to synchronize with the BLE source to be attacked which requires a sniffing or channel sensing component.

A reactive narrow-band jammer emits the jamming signal on a single BLE advertising channel only when a frame transmission to be attacked has been detected. Again frequency hopping need to be performed in order to not miss beacon frames transmitted on other channels.

V. Proposed Solution

A. Design Principles

The main goal is to design an efficient low-cost, low-power, hard to detect, selective jammer using commercial off-the-shelf hardware. This is achieved by designing a jammer which is:

- reactive,
- selective,
- narrow-band,
- power efficient,
- low cost & small-sized.

The proposed jammer is *reactive* as only a short jamming signal is only emitted during the transmission of a beacon frame which is long enough to corrupt the frame (CRC fail). Selectivity is achieved by using higher-layer information to decide whether to jam a frame already in the air or not. In the proposed solution the *selectivity* is based on BTAddr. Moreover, the jamming signal is *narrow-band*, i.e. it is only emitted on the frequency channels being used by the selected beacon. Therefore all available BLE advertisement channels need to be scanned in order to estimate the channels being actually used by the beacons source to be attacked. Moreover, we exploit the fact, that according to the BLE specification the hopping sequence over the advertising channels is deterministic and hence can be easily followed by a jammer node. The beacons are sent on the selected set of advertising channels always in the same ascending order, i.e. a beacon source that is using all the three advertising channels 37, 38 and 39 will transmit its beacon first on channel 37, then 38 and finally on 39. For the jammer we use commercial off-the-shelf (COTS) embedded BLE-capable hardware[3]. There is no need for expensive and

[3]RedBearLab BLE Nano http://redbearlab.com/blenano/

power hungry Software-defined Radio hardware (e.g. USRP). The selected hardware is flexible, *low cost*, and *very small*, i.e. size of a quarter dollar coin and *power efficient*, i.e. can be powered by button cell. Still it provides a fast TX/RX turn-around time of just 140 μs which is sufficient for selective jamming and fast channel hopping.

B. Detailed Specification

Several steps are necessary to be able to jam BLE advertising of a particular beacon source. First, we need to find out the specific configuration parameters used by a particular beacon source we want to attack. In case of Apple's iBeacon we scan the BTLE advertisement channels for beacons having the particular company ID and record the corresponding BRAddr. For generic beacons we have to know the BRAddr upfront. Second, we have to scan the BLE advertisement channels to see whether the beacon source is using all the three advertisement channels or just a subset, e.g. only channel 37 and 38. Third, we have to configure (program) the jammer by telling it the set of BRAddr to be jammed (blacklist) or not be jammed (whitelist) as well as the BLE advertisement channels used.

The actual jammer consists of two components, namely, i) beacon detection and ii) jamming. In the detection phase the jammer decodes received beacon frame headers on-the-fly to decide based on configured blacklisting / whitelisting of BRAddrs whether to jam the remaining frame (payload) or not. On successful detection the jammer starts the jamming phase by emitting a very short jamming signal on the channel used. Thereafter, the jammer switches to the next used advertising channel and restarts the detection phase. The Finite State Machine (FSM) model of the proposed jammer showing the configuration when all three advertisement channels are being used is depicted in Fig. 4. Note, that the jammer enters the initial state after timeout.

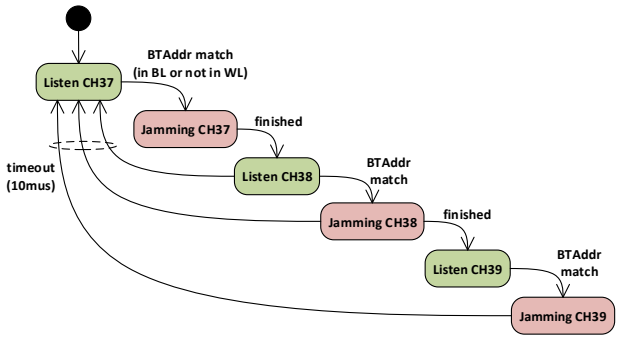

Fig. 4. FSM model of the proposed jammer for the configuration when all three advertisement channels are used.

The API which is used to program the jammer is shown in Tab. I.

C. Implementation Details

1) Jammer Hardware: For the jammer node we used a *RedBearLab BLE Nano* Development Board, which is equipped with a *Nordic nRF51822* SoC and an integrated antenna (Fig. 5). The *nRF51822* runs on a 16 MHz ARM Cortex-M0 processor and has an embedded 2.4 GHz transceiver, supporting BLE and the *Nordic Gazell* protocol stack. The transceiver has a turn-around time (the time needed to switch from receiving mode to transmitting mode) of only 140 μs. Therefore it fits into the timing constraints of reactive jamming BLE. The maximum transmission power of the transceiver is 4dBm, which was used throughout all measurements.

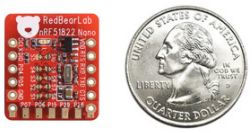

Fig. 5. Jamming hardware: RedBearLab BLE Nano, adapted from [9]

2) Jamming Program: The jamming program is an interrupt-controlled state machine running on the nRF51822. At startup the jammer tunes its receiver onto the first advertising channel using the standard advertising access address 0x8E89BED6. It is important to note that the access address is used as correlation code, so any non-standard communication using another on the advertising channel is hidden from the jammer. If a frame is detected (so after preamble and access address are received) the transceiver will start an integrated bit counter that counts every received bit and will generate a CPU interrupt (BCMATCH) when it reaches a predefined value. The bit counter value has to be big enough to ensure the filter characteristic of the frame is received at BCMATCH, e.g. 64 bits for the device address (16 bits PDU header plus 48 bits address). Using the fact that the transceiver constantly writes received bytes to memory, we can then use BCMATCH interrupt to check whether the frame characteristic matches our filter. If not, the transceiver is switched to transmitting mode and sends a fake frame, again using the internal bit counter to stop transmission after one bit of frame data. Unfortunately, the bit counter can only be started after preamble (which is not configurable) and access address is transmitted, so we send a total of 41 bits. After the transmission is stopped, the transceiver is tuned to the next advertising channel to receive and to repeat the process. To avoid being stuck at one channel (if the channel is not used by the targeted devices, or if the transceiver is unable to detect frames) a 10 ms timeout is started (which is the maximum time between two advertising frames on consecutive used advertising channels sent in one advertising event [1]). On a timeout the jammer will return to the first advertising channel. If no timeout occurs the jamming process continues as on the previous channel and repeats on the last channel afterwards.

VI. Countermeasures

A. Attack Mitigation

Although the proposed jammer is suitable to attack any standard-complying beacon, our attack can be mitigated using a slight modification to the advertising process of the beacon.

TABLE I
JAMMER API DESCRIPTION

Function	Description
get_braddrs_by(companyID, ...)	scans all BTLE advertisement channels for beacons having the particular company ID, e.g. Apple iBeacons. Reports the BRAddr being used.
scan_for_bdaddr(BRAddr)	scans for beacons with address BTAddr on all BTLE advertisement channels. Reports the channels being used.
configure_jammer(blacklist of BRAddr, whitelist of BRAddr, specification of channel hopping)	configures the jammer using either a blacklist or whitelist of BRAddr to be jammed and not jammed respectively.

If the advertiser chooses a pseudo-random channel hopping pattern based on a common secret between sender and receiver, as proposed by Emek and Wattenhofer [10], instead of transmitting in a round-robin fashion over the advertising channels, the proposed jammer would not be able to adjust its channel hopping pattern to be able to jam every beacon frame. However, such an approach would require a preliminary key establishment session which could also be jammed as discussed in more detail in Strasser et al. [11].

Nevertheless, in any case, the required modification to the BLE advertising process, even though it is incompatible with the BLE standard, would not affect the receiver's ability scan for the beacon frames.

When extending the proposed jammer to use three jammer nodes, one for each BLE advertising channel, channel hopping becomes unnecessary. Such an approach is eligible as the used jammer hardware is low-cost. With such a jammer, to the best of our knowledge, there is no countermeasure against the jamming attack possible, without modifying the BLE specification completely, e.g. like proposed in Zhang et al. [12].

B. Attack Detection

Despite the lack of active countermeasures, the proposed jamming attack can still be detected by sending very short decoy beacons (just header up to BRAddr) and having a sniffer which collects all overheard frames. However, as the sniffer is unable to guess the access address used by the jammer he will fail to decode the jamming frames. For attack detection a more sophisticated IDS is required. In particular it has to analyze the energy on the channel just after the end of the decoy frame where an elevated value might be an indicator for an ongoing jamming attack.

VII. EVALUATION

The proposed jammer is analyzed by means of experiments in a small testbed. First, we present the methodology used. Second, the performance metrics are identified and results are presented.

A. Methodology

The hardware used for the jammer and receiver node was already described in V-C1. As BLE beacon source we used regular commercially available hardware, the *Gigaset G-tag*. The beacon transmits ADV_IND frames at an advertising interval of one frame per second with a power of 0 dBm. Each frame has a total length of 46 bytes. The receiver node consists of three *RedBearLab BLE Nano* boards - one for each BLE advertising channel. By using a dedicated receiver for each BLE advertising channel we make sure that we do not lose frames due to channel hopping. Note, this represents the optimal receiver; a normal smartphone will receive less beacons as it cannot simultaneously listen on all BLE advertisement channels and has therefore do channel hopping. All boards are flashed with the *Nordic nRF Sniffer* firmware and connected via UART to a laptop. The receivers are controlled by a Python script on the laptop via the *Nordic Sniffer API*[4]. Each receiver is set to listen for incoming frames on one channel only, so all advertisement channels are covered and no hopping is performed. Through the Python script all received frames, even invalid or malformed frames are written to a PCAP file for post-analysis with *MatLab*. Further, the jammer was configured to transmit with 4 dBm transmit power.

B. Performance Metrics

As primary metric we identified the Advertising Success Rate (ASR) which is the ratio between the number of correctly received BLE advertising events at the receiver side and the total number of transmitted BLE advertisements. Note, a BLE advertising event is successful if in a given advertisement period at least a single BLE advertisement frame is correctly received on either channel 37, 38 or 39. The lower the ASR the more successful the jammer is, i.e. ASR=0 is perfect jamming. Another important metric is the jamming area or coverage of a jammer which is defined as the spatial area around the jammer with a ASR less than some threshold, e.g. 1%.

C. Results

Experiment: (Analyzing ASR and jammer coverage) The objective is to analyze the efficiency of the proposed BLE advertisement jammer in terms of ASR and jammer coverage. We set-up an outdoor experiment in which we put the three nodes, namely beacon source, jammer and receiver, on a line elevated by 1 m from the ground (grass field). Here we fixed the distance between the beacon source and the receiver to 3.7 m which corresponds to a receive power of -77 dBm. The distance between the jammer and the receiver nodes was varied from around 1 to 10 meters. The advertising interval of the beacon source was set to 1 s. Moreover the beacon source was configured to sent a beacon on all the three advertisment channels during the advertisement interval.

Result: Fig. 6 shows the estimated ASR. At a distance of 76 cm the ASR is zero, i.e. the jammer is able to successfully

[4]https://goo.gl/I3myff

jam each transmitted BLE advertisement frame on each channel (here: 37, 38 and 39). By increasing the distance between the jammer and the receiver the ASR increases. At a distance of around 10 m there is no significant impact of the jammer as the comparison with the no jamming case shows. Note that the jamming distance is depending on the transmission power of the jammer, e.g. here 4 dBm, if a greater distance is needed the transmission power has to be increased.

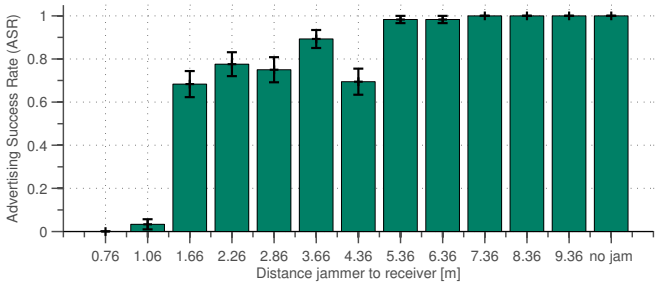

Fig. 6. Impact of distance between jammer and receiver (distance beacon source to receiver is 3.7 m corresponding to receive power of around -77 dBm). The mean and standard error value is presented.

VIII. Related Work

In [13], [14] a survey on different jamming attacks in WSNs is given. Jamming classical Bluetooth was analyzed by Koeppel et al. [15] where he showed that it is possible to jam a specific Bluetooth connection without any prior knowledge about it. The parameters needed to compute the Bluetooth channel hopping sequence (master's LAP/UAP and clock) can be determined by analyzing the observed traffic of the piconet [16]. With this information the jammer follows the channel hopping sequence and jams the piconet. The prototype was implemented using the inexpensive Ubertooth hardware platform. Reactive and selective jammers have also been developed to target wireless protocols which do not apply Frequency Hopping Spread Spectrum (FHSS). Reactive jammers using expensive USRP-SDR platform were studied in the context of 802.11 [17], [18], 802.11p (VANET) [14] and 802.15.4 [19]. Moreover, in WiFire [20], the authors presented a wireless firewall for 802.15.4 based networks that utilizes a USRP-SDR based reactive jammer to classify and effectively block undesired communication without interfering with desired communication. Selective jamming of 802.11 using USRP-SDR was analyzed by Cassola et al. [21]. The first solution using cheap off-the-shelf Wi-Fi dongles was given by Vanhoef et al. [7].

IX. Conclusions

In this paper we presented a selective jammer against BLE advertising beacons. The jammer selectively jams only BLE advertising beacons for which it was configured using either blacklisting or whitelisting of BTAddrs. Moreover, only the BLE advertising channels actually being used are jammed (narrowband). The prototypic implementation is low-cost, small-size and very power efficient. Experiment results demonstrate the feasibility of the proposed solution and reveal the impact of the distance between jammer and receiver on the frame delivery rate.

X. Acknowledgment

This work has been supported by the STEUERUNG project funded by Deutsche Telekom.

References

[1] "Bluetooth v4.2 specification," Bluetooth Special Interest Group. [Online]. Available: https://www.bluetooth.org/en-us/specification/adopted-specifications

[2] J. Bruins, "What's New In Core Location?" Apple World Wide Developers Conference, Apple Inc., 2013. [Online]. Available: http://devstreaming.apple.com/videos/wwdc/2013/307xex4xl1ey243ksyxqfip0xowr/307/307-SD.mov?dl=1

[3] M. S. Gast, *Building applications with IBeacon: proximity and location services with bluetooth low energy*. " O'Reilly Media, Inc.", 2014.

[4] C. Gomez, J. Oller, and J. Paradells, "Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology," *Sensors 2012*, vol. 12, no. 9, pp. 11 734–11 753, 2012.

[5] Unacast, "The Proxbook Report -The State Of The Proximity Industry," Tech. Rep., 2016.

[6] ABI Research, "BLE Tags: The Location of Things (LOT)," Tech. Rep., 2015.

[7] M. Vanhoef and F. Piessens, "Advanced wi-fi attacks using commodity hardware," in *Proceedings of the 30th Annual Computer Security Applications Conference*. ACM, 2014, pp. 256–265.

[8] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," in *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*. ACM, 2005, pp. 46–57.

[9] redbearlab.com, "Ble nano comparision to quarter dollar coin," *http://redbearlab.com/blenano/*.

[10] Y. Emek and R. Wattenhofer, *Frequency Hopping against a Powerful Adversary*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[11] M. Strasser, C. Pöpper, S. Capkun, and M. Cagalj, "Jamming-resistant key establishment using uncoordinated frequency hopping," in *2008 IEEE Symposium on Security and Privacy (sp 2008)*. IEEE, 2008.

[12] L. Zhang, H. Wang, and T. Li, "Anti-jamming message-driven frequency hopping—part i: System design," *IEEE Transactions on Wireless Communications*, vol. 12, no. 1, pp. 70–79, 2013.

[13] A. Mpitziopoulos, D. Gavalas, C. Konstantopoulos, and G. Pantziou, "A survey on jamming attacks and countermeasures in wsns," *Communications Surveys & Tutorials, IEEE*, vol. 11, no. 4, pp. 42–56, 2009.

[14] O. Punal, C. Pereira, A. Aguiar, and J. Gross, "Experimental characterization and modeling of rf jamming attacks on vanets," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 2, pp. 524–540, Feb 2015.

[15] S. Köppel, "Bluetooth Jamming," Bachelors Thesis, ETH Zürich, Switzerland, 2013, ftp://ftp.tik.ee.ethz.ch/pub/students/2012-HS/BA-2012-16.pdf.

[16] D. Spill and A. Bittau, "Bluesniff: Eve meets alice and bluetooth," in *Proceedings of the First USENIX Workshop on Offensive Technologies*, ser. WOOT '07. Berkeley, CA, USA: USENIX Association, 2007.

[17] E. Bayraktaroglu, C. King, X. Liu, G. Noubir, R. Rajaraman, and B. Thapa, "Performance of ieee 802.11 under jamming," *Mobile Networks and Applications*, vol. 18, no. 5, pp. 678–696, 2013.

[18] S. Prasad and D. J. Thuente, "Jamming attacks in 802.11 g – a cognitive radio based approach," in *MILITARY COMMUNICATIONS CONFERENCE, 2011-MILCOM 2011*. IEEE, 2011, pp. 1219–1224.

[19] M. Wilhelm, I. Martinovic, J. B. Schmitt, and V. Lenders, "Short paper: reactive jamming in wireless networks: how realistic is the threat?" in *Proceedings of the fourth ACM conference on Wireless network security*. ACM, 2011, pp. 47–52.

[20] M. Wilhelm, I. Martinovic, J. Schmitt, and V. Lenders, "Wifire: A firewall for wireless networks," in *Proceedings of the ACM SIGCOMM 2011 Conference*, ser. SIGCOMM '11. New York, NY, USA: ACM, 2011.

[21] A. Cassola, W. K. Robertson, E. Kirda, and G. Noubir, "A practical, targeted, and stealthy attack against wpa enterprise authentication." in *NDSS*, 2013.