Low-Power Wireless Advertising Software Library for Distributed M2M and Contextual IoT

Mohamed Imran Jameel
McGill University
Montréal, Québec, Canada
Email: imran.jameel@mail.mcgill.ca

Jeffrey Dungen
reelyActive
Montréal, Québec, Canada
Email: jeff@reelyactive.com

***Abstract*—Bluetooth Smart is emerging as arguably the first global low-power wireless standard for the Internet of Things, bringing with it billions of devices, or "Things", capable of spontaneously broadcasting short messages to any potential receiving devices in range. If a widespread infrastructure of such receiving devices were to exist, these broadcast messages could be reliably captured, parsed, and forwarded in IP packets via the Internet to any and all concerned parties, enabling connectionless, distributed low-power M2M networks. In this paper we present advlib, a software library for parsing low-power wireless broadcast (also known as advertising) packets, with this objective. Experimental results indicate that, coupled with the necessary receiver infrastructure, in many cases at least the device vendor can be identified, validating the potential for M2M forwarding. Moreover, results suggest that sufficient semantically-meaningful information may be extracted by the library to support contextual IoT applications even at a local scale. Development continues on extending the support of known protocols and establishing the necessary relationships with device vendors.**

***Keywords*-IoT; Bluetooth Smart; BLE; Library; Active RFID; Wireless; M2M; Contextual Awareness**

I. Introduction

Bluetooth Smart (historically referred to as BLE) is a low-power wireless protocol for short range communications [1]. One specific feature of BLE not found in other versions of Bluetooth is the ability for a device to spontaneously broadcast short messages containing tens of bytes to any receiving devices which may be in range. This is known as advertising and it is similar in function to that of active RFID, where devices periodically emit unsolicited messages including their identifier and possibly sensor and/or status data. For instance, our company, reelyActive has successfully developed a proprietary active RFID protocol in 2012 with similar characteristics. However, the key differentiator is that BLE is today shipping over 3 billion devices annually [2] which are used globally, while proprietary active RFID technologies, such as our own, are relegated to specific applications in specific geographies. As a result, BLE is arguably the first global low-power wireless standard and, as such, is likely to play a significant role in the Internet of Things (IoT).

While the phrase Internet of Things was coined in 1999 [3] extensive use of the term is a far more recent occurrence, paralleling the proliferation of inexpensive, interoperable and embeddable wireless chipsets which make their inclusion in everyday objects increasingly economically viable. While industry veterans such as Cisco suggest that 50 billion devices will connect to the IoT by 2020 [4], other research takes an alternative view, suggesting 212 billion computerized devices by 2020, of which 15% will be connected [5]. Should a significant number of those extra 162 billion devices possess a BLE chip or similar technology, it is indeed foreseeable that they could at least occasionally be IoT-connected when in range of edge routers capable of receiving their periodic broadcasts. Admitting this case, the total number of IoT devices by 2020 may indeed be higher than widely predicted.

In 2013, we at reelyActive published a paper entitled "Towards a Simple, Versatile, Distributed Low-Power Wireless M2M Infrastructure" [6] in which we presented edge routers capable of receiving and forwarding over IP networks both BLE and proprietary active RFID broadcast packets. At that time, the number of advertising BLE devices detected in a crowded public space was typically low, and often zero. Compare that with today where the authors have often observed, in range of a single receiver, over 100 advertising devices at technology conferences. The proliferation of Apple iBeacons [7], wearables, and emerging standards such as Eddystone [8], an open beacon format from Google, all based on BLE, means that it is not uncommon for even sparsely deployed infrastructure to discover a new advertising device every week.

In light of this proliferation, there is a need for an open tool which can parse and extract semantic meaning [9] from the plethora of advertisement data. In this paper we present a software library for parsing low-power wireless broadcast packets which supports both BLE and our own proprietary active RFID protocol. Our motivation is twofold: first to facilitate low-power, connectionless M2M and, second, to facilitate novel, contextual IoT applications.

II. Low-Power Wireless Advertising

In this section we present the characteristics of low-power wireless advertising. First we present the BLE advertising protocol followed by protocols built on top thereof. Lastly we present the reelyActive active RFID protocol.

A detailed examination of BLE can be found in [10]. For instance, BLE has two ways of communicating. Firstly, using advertisements, where a BLE peripheral device (advertiser) broadcasts packets to every device around it. The receiving device (scanner) can then act on this information or connect to receive more information. Secondly, is to receive packets using a connection, where both the peripheral and central send packets.

The BLE channel plan is split into 37 data communication channels and 3 dedicated advertising channels used for device discovery, connection initiation and broadcasting. Advertising channels are allocated in different parts of the spectrum in order to avoid interference from 802.11/Wi-Fi signal. This is demonstrated in Figure 1.

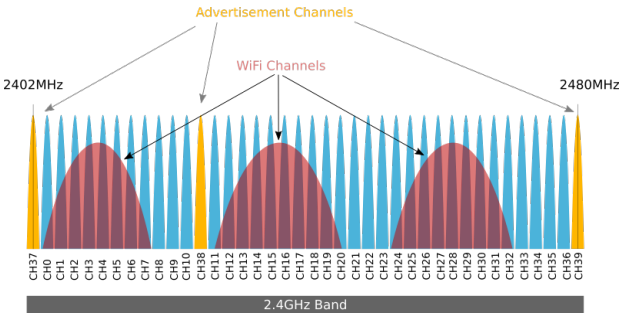

Figure 1. The 2.4GHz spectrum for Bluetooth extends from 2402MHz to 2480MHz. LE uses 40 1MHz wide channels, numbered 0 to 39. Each is separated by 2MHz. Channels 37, 38, and 39 are used only for sending advertisement packets. The rest are used for data exchange during a connection. [11]

A. BLE Advertising Packets

A BLE device in advertising mode may periodically transmit packets carrying advertising information as Payload Data Units (PDU) on the RF channels dedicated for this purpose.

Table I
ALLOWABLE RESPONSES TO ADVERTISING EVENT TYPES

Advertising Event Type	PDU Used	Scannable	Connectable
Connectable Undirected	ADV_IND	YES	YES
Connectable Directed	ADV_DIRECT_IND	NO	YES
Non-Connectable Undirected	ADV_NONCONN_IND	NO	NO
Discoverable Undirected	ADV_DISCOVER_IND	YES	NO

As Table 1 explains, there are several PDU types for the advertisements which indicate whether the device accepts connections and/or whether it will respond to a scan request in which it may send a second complimentary packet containing additional information.

As shown in Figure 2 and 3, a standard BLE advertising packets consist of the Preamble (1 byte), Access Address (4 bytes), Payload Data Unit (PDU) (up to 39 bytes) and CRC (3 bytes). Only the PDU contains information pertinent to our motivations, the other bytes can be seen as overhead.

An advertising packet PDU has a header and a variable payload. The header contains information about the size of the payload and its type. The payload supports a variable number of advertisement data structures which are limited to 31 bytes.

A one-bit flag (TxAdd) contained in the two-byte header indicates whether this address is public or random. Public addresses are uniquely assigned to the device by the IEEE. Random addresses are, as the name implies, randomly selected and may change over time.

Payload data structure types are specified by the Generic Access Profile (GAP) and include service UUIDs, name strings, service data as well as manufacturer specific data for generic information [12].

B. Protocols built on BLE

1) iBeacon: In 2013, Apple unveiled the iBeacon protocol as a way to add real world context to smartphone applications. An iBeacon is a BLE advertiser with manufacturer specific data defined by Apple. This data includes a 16 byte UUID, four additional identification bytes and transmission power indication, as shown in Figure 2.

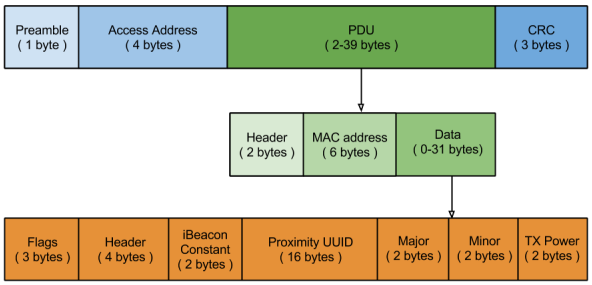

Figure 2. Bluetooth Packet with iBeacon

2) Eddystone: In July 2015, Google released their very own cross platform, open-source, Eddystone Beacon. More interestingly, in addition to broadcasting a UUID, Eddystone also broadcasts two more packets (or frame types) - a uniform resource locator (URL), and a Telemetry (TLM) stream. The URL frame type is a lot more flexible than broadcasting the UUID (which is just a unique string of numbers and letters). Therefore, instead of being a pointer to a back-end database, uninterpretable by anyone except the company and the native app with which the beacon is associated, a URL can be interpreted by any piece of software that can decode the beacon. Furthermore, the TLM frame type broadcasts data obtained from sensors. This allows the triggering of different actions based on sensors

data such as temperature, humidity, sound, and proximity, as is common in traditional M2M applications.

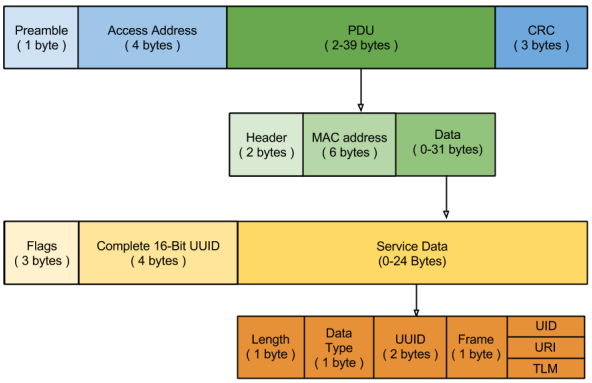

Figure 3. Bluetooth Packet with Eddystone

C. reelyActive Advertising Packets

The reelyActive proprietary active RFID protocol currently supports two packet types: identification, and identification plus sensor data.

Both packet types support a globally unique EUI-64 identifier where the OUI-36, belonging to reelyActive, is implicit. As a result, any advertising device can be positively and uniquely identified. In the case of the latter packet, both the temperature and battery voltage of the device are also present.

III. advlib

Here we present the design and implementation of advlib, our open source library for parsing low-power wireless broadcast packets. We first present the design considerations followed by the final implementation.

A. Design Considerations

The majority of reelyActive software is open source and written in JavaScript. Server-side code is based on the Node.js framework and client-side code makes extensive use of the AngularJS framework. While these precedents contribute largely to the design of advlib, we nonetheless present the specific considerations for the project.

1) Open-Source: Given that the library will evolve continuously as the BLE and other low-power wireless protocols evolve, and as new devices come online, it is imperative that the project be open source and moreover accept contributions from outside developers with timely domain-specific knowledge.

2) Lightweight server-side operation: Given that the library will run not only on powerful cloud servers but also on the resource-constrained embedded hardware of the edge devices of the IoT, it is imperative that the library support lightweight server-side operation.

3) Client-side operation: The ability to run the library in a standard web browser with a simple user interface (UI) is not imperative, but highly desirable as it permits users to parse and manipulate raw packets without the need for installing and running software.

In all cases it is imperative that the tools and frameworks be licensed for free use in order to facilitate widespread adoption for the IoT.

B. Implementation

Based on the design considerations, advlib is written in JavaScript for the Node.js framework. A bundled version is available as an npm package which installs in a single line [13]. The open source code is available on GitHub [14] under an MIT License. A client-side version is equally available and hosted on GitHub Pages for anyone to use live [15].

advlib ingests PDUs as raw hexadecimal strings and outputs the parsed data as JavaScript Object Notation (JSON). The following code snippet demonstrates the API call required to process a BLE PDU.

```
var advlib = require('advlib');
var rawPDU = '421655daba50...';
var output = advlib.ble.process(rawPDU);
console.log(output);
```

Examples of the JSON output are provided in the following section, and additional documentation can be found on the GitHub and npm project pages presented previously.

IV. Experimental Results

We have used advlib in conjunction with hardware receivers in both private and public environments to capture and parse real-world BLE advertising packets as well as those of our own active RFID devices. In this section we present what identification and sensor data can typically be extracted, specifically that which is non-proprietary and can be decoded.

A. Identification by Advertiser Address

Each BLE advertisement packet contains, at a minimum, a 48-bit address which identifies the advertising device. The following JSON is the output of advlib for a minimal packet.

```
{
  type: "ADVA-48",
  value: "5278795d1474",
  advHeader: {
    type: "SCAN_REQ",
    length: 12,
    txAdd: "random",
    rxAdd: "public"
  },
  advData: {}
}
```

In this case the 48-bit address, represented in hexadecimal, is 52:78:79:5d:14:74. Here the address is random, which we have observed as typical for smartphones from which this packet originated.

More specifically, this packet is of the SCAN_REQ type. We have observed that when edge routers advertise using the ADV_DISCOVER_IND packet type (see Table 1) which is scannable but not connectable, BLE-enabled smartphones in range will indeed often scan the edge router with a SCAN_REQ packet. In this special case, a device such as a smartphone, which we have not observed sending unsolicited advertisement packets, is nonetheless incited to divulge its advertiser address in the scan request. From our observations, devices running the latest mobile operating systems such as iOS8 and Android 5 use random advertiser addresses which change regularly, and therefore these cannot be uniquely identified.

B. Identification by Service UUIDs

1) iBeacon UUID: The following example demonstrates the BLE packet for a Kontakt.io iBeacon. This is classified according to its 'manufacturerSpecificData.iBeacon.uuid' property, which the authors determined experimentally through device isolation, although it resides with Apple in their iBeacon classification.

```
advData: {
  manufacturerSpecificData: {
    companyName: "Apple, Inc.",
    companyIdentifierCode: "004c",
    data: "0215f7826da64fa24e988024bc5b71e0893ef7e84be5b3",
    iBeacon: {
      uuid: "f7826da64fa24e988024bc5b71e0893e",
      major: "f7e8",
      minor: "4be5",
      txPower: "-77dBm",
      licenseeName: "Kontakt.io"
    }
  }
}
```

2) BLE UUID: The following example demonstrates the parsed BLE packet data for the FitBit. This is classified according to its 'nonComplete128BitUUIDs' property, which the authors determined experimentally through their observation of multiple FitBit devices.

```
advData: {
  nonComplete128BitUUIDs: "adabfb006e7d4601bda2bffaa68956ba",
  serviceData: {
    uuid: "180a",
    data: "1204eb150000"
    specificationName: "Device Information"
  }
}
```

3) Local Name: The following example demonstrates the parsed BLE packet data for the Xiaomi Mi Band. This is classified according to its 'completeLocalName' property, which the authors determined experimentally through their observation of multiple Mi Bands.

```
advData: {
  completeLocalName: "MI"
}
```

C. Contextual Data by URL

As previously described in the Eddystone subsection, an advertised URL can serve to provide far more contextual information when accessed than an identifier alone.

The following example demonstrates the parsed BLE packet data for the UriBeacon, precursor to the Eddystone. The URL is encoded in the 'serviceData' object.

```
advData: {
  serviceData: {
    uuid: "fed8",
    data: "00f20275726962656163 6f6e08",
    companyName: "Google",
    uriBeacon: {
      invisibleHint: "false",
      txPower: "-14dBm",
      url: "http://uribeacon.org"
    }
  }
}
```

D. Identification by EUI-64

Each reelyActive advertisement packet contains, at a minimum, a 28-bit address which, coupled with the implicit OUI-36 (0x001bc5094), forms an EUI-64 identifying the advertising device. The following JSON is the output of advlib for a minimal packet.

```
{
  type: "EUI-64",
  value: "001bc50941234567"
}
```

V. Present and Future Applications

As presented in Section III, advlib is freely available for use as an open source software package, and is integrated in the reelyActive software stack. As such it currently serves in applications ranging from retail analytics to social discovery in co-working spaces to physical collision detection. In

this section we present how advlib currently contributes to contextual IoT applications and how it may contribute to low-power wireless M2M and might even be used for reverse look-up.

A. Contextual Smart Spaces

In 2013, we at reelyActive published a paper entitled Hyperlocal Context to Facilitate an Internet of Things Understanding of the World [16] in which we presented a contextual visualisation of people present at a location, as detected by their uniquely identifiable active RFID tags. This visualisation has evolved as a platform entitled Smart Spaces [17] which is a commercially viable product.

Currently the information about the device vendor as extracted by advlib serves to visualise the variety of devices which are in range of receiver infrastructure, as shown in Figure 4. For instance, an Estimote iBeacon is displayed as such because it can be uniquely identified by its iBeacon UUID, as described in Section II.

Figure 4. Devices at Bluetooth World 2015

The URI Beacon protocol, now integrated into Eddystone as described in Section II, is especially promising because compatible devices can simply advertise a URL at which all the information required for visualisation in the Smart Spaces interface can be provided. This technique is successfully used today.

B. Distributed M2M

Given that the vendor of many low-power wireless devices can be determined from an advertising packet, it would be feasible to forward those packets to the corresponding vendor via an API. In this case, the vendor could process and, in turn, forward the data to any concerned parties. Since it is not uncommon for receiver infrastructure owned by one party to communicate with software running advlib operated by a second party, such a forwarding arrangement would in effect represent distributed M2M. A relevant example involves BLE tracking devices such as those sold by Tile [18] or TrackR [19]. Third party receiver infrastructure could capture such devices' broadcast packets, and forward these to the corresponding vendor. This can be seen as an extension of "crowd GPS" applications leveraging mobile phones as receivers, not unlike those discussed in [20] and [21].

C. Reverse operation

The current advlib software converts raw hexadecimal strings into meaningful JSON, however practical applications of the reverse can be imagined. For instance, it would be desirable to create an advertising packet via a GUI in a web browser which would convert the packet to raw bytes which would be understood by transmitter infrastructure. This feature is prioritized for future development.

VI. Conclusion

In this paper we have presented the capabilities of an open source software library for parsing low-power wireless broadcast packets. We have demonstrated that unique identification of devices and/or their vendors is often possible from these packets, as is sensor data. As a result, this information is currently being used successfully for contextual IoT applications and is in the pilot phase for distributed M2M applications such as those presented herein. Thanks to low-power wireless standards such as BLE, the IoT is growing by billions of devices per year, and the advlib is well positioned as an open, extensible library, accessible to the scientific and industrial communities alike, for the interpretation of any and all data these devices may advertise.

Acknowledgment

The authors would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) for its continued support via the Industrial Undergraduate Student Research Award (IUSRA).

References

[1] Bluetooth SIG. *Specification of the Bluetooth System, Version 4.2*, 2014.

[2] Bluetooth SIG, "Building for the IoT? The Bluetooth Developer Studio Just Made Your Job Easier". (2015, Apr. 14). [Online]. Available: http://www.bluetooth.com/pages/press-releases-detail.aspx?itemid=228.

[3] K. Ashton. (2009, Jun. 22). "That 'Internet of Things' Thing". [Online]. Available: http://www.rfidjournal.com/articles/pdf?4986.

[4] D. Evans. (2011). "How the Next Evolution of the Internet Is Changing Everything". [Online]. Available at: http://www.iotsworldcongress.com/documents/4643185/3e968a44-2d12-4b73-9691-17ec508ff67b.

[5] V. Turner *et al*. (2014). "The Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things". [Online]. Available: http://www.emc.com/leadership/digital-universe/2014iview/internet-of-things.htm.

[6] J. Dungen *et al* "Towards a simple, versatile, distributed low-power wireless M2M infrastructure", in *2013 IEEE 38th Conf. on Local Computer Networks (LCN)*, Sydney, Australia, pp.890-895, 21-24.

[7] Apple Developers, "iBeacons for Developers". [Online]. Available: https://developer.apple.com/ibeacon/.

[8] Google Developers, "Beacons", [Online]. Available: https://developers.google.com/beacons/.

[9] D. Guinard *et al*, "From the Internet of Things to the Web of Things: Resource-oriented Architecture and Best Practices" in *Architecting the Internet of Things*, Heidelberg, Germany: Springer Berlin Heidelberg, 2011, pp. 97–129,.

[10] C. Gomez *et al*, "Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-Power Wireless Technology", *Sensors*, vol. 12, no. 12, pp. 11734–11753, Aug. 2012.

[11] Argenox Technologies, "A BLE Advertising Primer". [Online]. Available: http://www.argenox.com/bluetooth-low-energy-ble-v4-0-development/library/a-ble-advertising-primer.

[12] Bluetooth SIG, "Company Identifiers". [Online]. Available: https://www.bluetooth.org/en-us/specification/assigned-numbers/company-identifiers.

[13] M. Jameel and J. Dungen. (2015). "Library for Decoding Wireless Advertising Packets.", *npm*. [Online]. Available: https://www.npmjs.com/package/advlib.

[14] M. Jameel and J. Dungen. (2015). "Advlib", *GitHub*, 2015. [Online]. Available: https://github.com/reelyactive/advlib.

[15] M. Jameel and J. Dungen. (2015). "Advlib: An Open Library for Wireless Advertising Packets", *reelyActive*, 2015. [Online]. Available: http://reelyactive.github.io/advlib/.

[16] J. Dungen and P.-O. Genest, "Hyperlocal Context to Facilitate an Internet of Things Understanding of the World" in *International Workshop on Identification, Information and Knowledge in the Internet of Things*, Beijing, China, 2013. [Online]. Available: http://reelyactive.com/science/reelyActive-IIKI2013.pdf

[17] reelyActive, "Smart Spaces". [Online]. Available: http://smartspac.es.

[18] Tile, "Never Lose Your Keys, Wallet Or Anything Again". [Online]. Available: https://www.thetileapp.com.

[19] TrackR, "Track your phone, wallet, keys anything else with TrackR". [Online]. Available: https://www.thetrackr.com.

[20] U. Blanke *et al*, "Capturing crowd dynamics at large scale events using participatory GPS-localization" in *2014 IEEE 9th International Conf. on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, Singapore, pp.1-7, 21-24.

[21] J. Weppner and P. Lukowicz, "Bluetooth based collaborative crowd density estimation with mobile phones" in *2013 IEEE International Conf. on Pervasive Computing and Communications (PerCom)*, San Diego, CA, pp.193-200.