# Investigating the Energy Consumption of Member Discovery in Bluetooth Low Energy

by

**Ben Lynch, B.A.(Mod.)**

**Thesis**

Presented to the

University of Dublin, Trinity College

in fulfillment

of the requirements

for the Degree of

**Master in Computer Science**

**University of Dublin, Trinity College**

September 2017

# Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

Ben Lynch

February 23, 2017

# Permission to Lend and/or Copy

I, the undersigned, agree that Trinity College Library may lend or copy this thesis upon request.

Ben Lynch

February 23, 2017

# Acknowledgments

I would like to that my supervisor Jonathan Dukes for suggesting such an interesting project and assisting me throughout.

Ben Lynch

*University of Dublin, Trinity College*
*September 2017*

# Investigating the Energy Consumption of Member Discovery in Bluetooth Low Energy

Ben Lynch, MCS
University of Dublin, Trinity College, 2017

Supervisor: Jonathan Dukes

...ABSTRACT...

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 Introduction

Internet of things/WSNs; Emedded systems with limited power - low energy protocols - BLE [1]

## 1.2 Problem Area

Member discovery/member consensus in semi static Ad-hoc netoworks - role call

## 1.3 Motivation for this research

The rise of the Internet of things and Bluetooth Low Energy playing a large part in it.

## 1.4 Dissertation Structure

# Chapter 2

# Background

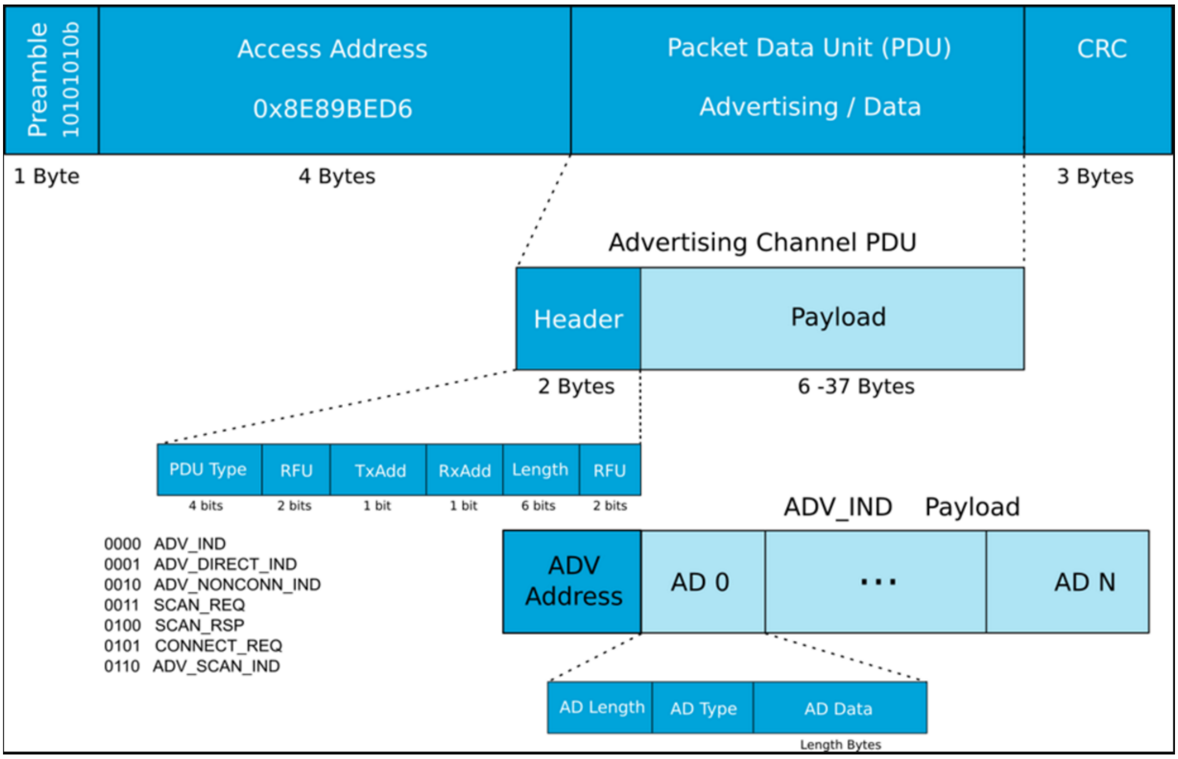
## 2.1 Introduction

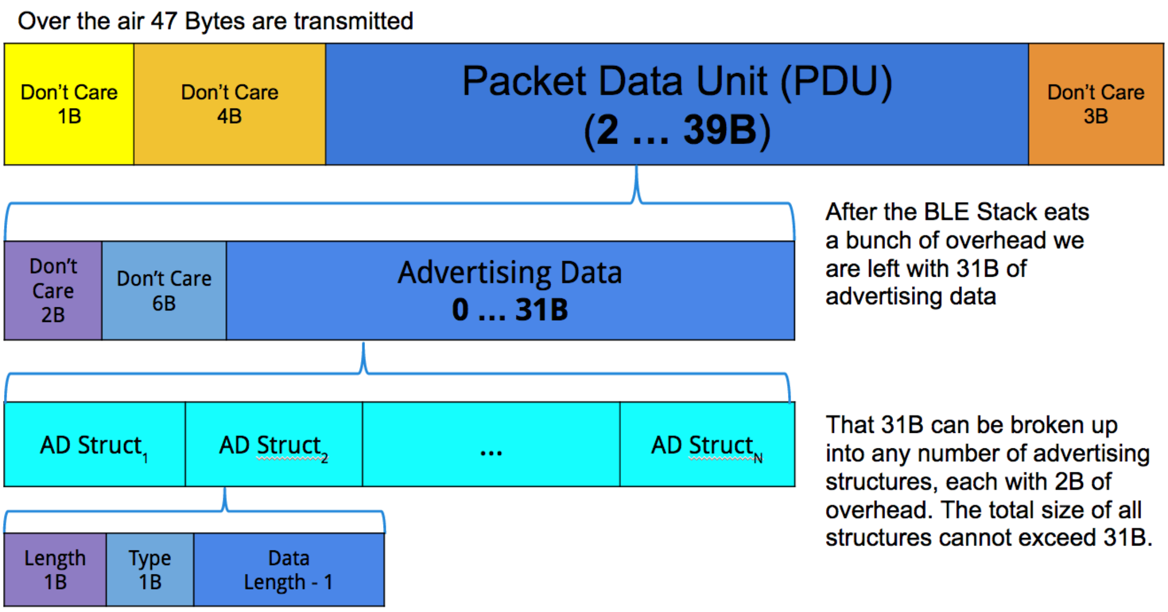
## 2.2 Bluetooth Low Energy

### 2.2.1 Scanning and Advertising

The Packet data unit for the advertising channel (called the Advertising Channel PDU) includes a 2-byte header and a variable payload from 6 to 37 bytes. The actual length of the payload is defined by the 6-bit Length field in the header of the Advertising Channel PDU.

there are several PDU types for the advertisements, but here we will focus ADV_NONCONN_IN - peripheral does not want to accept connections, which is typical in Beacons.


Preamble
10101010b
Access Address
0x8E89BED6
Packet Data Unit (PDU)
Advertising / Data
CRC
1 Byte
4 Bytes
3 Bytes
Advertising Channel PDU
Header
Payload
2 Bytes
6 -37 Bytes
PDU Type
RFU
TxAdd
RxAdd
Length
RFU
4 bits
2 bits
1 bit
1 bit
6 bits
2 bits
0000 ADV_IND
0001 ADV_DIRECT_IND
0010 ADV_NONCONN_IND
0011 SCAN_REQ
0100 SCAN_RSP
0101 CONNECT_REQ
0110 ADV_SCAN_IND
ADV_IND Payload
ADV Address
AD 0
...
AD N
AD Length
AD Type
AD Data
Length Bytes

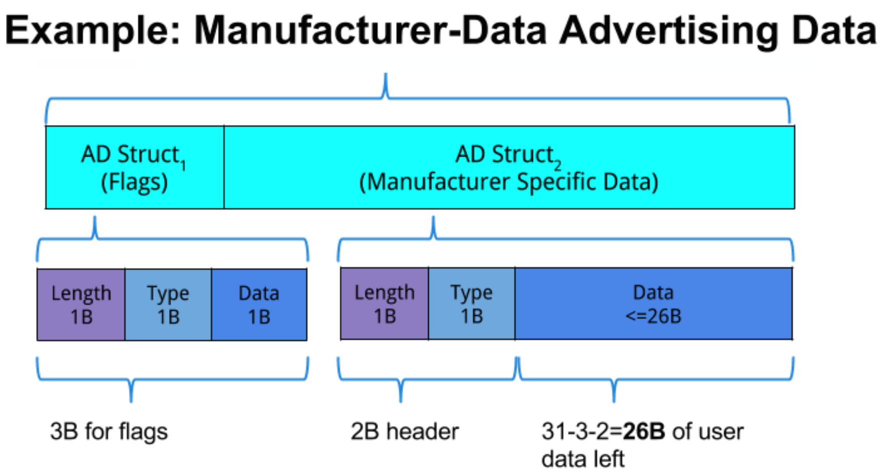

Over the air 47 Bytes are transmitted
Don't Care 1B
Don't Care 4B
Packet Data Unit (PDU) (2 … 39B)
Don't Care 3B
Don't Care 2B
Don't Care 6B
Advertising Data 0 … 31B
After the BLE Stack eats a bunch of overhead we are left with 31B of advertising data
AD Struct1
AD Struct2
...
AD StructN
That 31B can be broken up into any number of advertising structures, each with 2B of overhead. The total size of all structures cannot exceed 31B.
Length 1B
Type 1B
Data Length - 1

Every BLE package can contain a maximum of 47 bytes (which isn't much), but:

1. The BLE stack requires 8 bytes (1 + 4 + 3) for its own purposes.
2. The advertising packet data unit (PDU) therefore has at maximum 39 bytes. But the BLE stack once again requires some overhead, taking up another 8 bytes (2 + 6).
3. The PDU's advertising data field has 31 bytes left, divided into advertising data (AD) structures. Then:
   - The GAP broadcast must contain flags that tell the device about the type of advertisement we're sending. The flag structure uses three bytes in total: one for data length, one for data type and one for the data itself. The reason we need the first two bytes (the data length and type indications) is to help the parser work correctly with our flag information. We have 28 bytes left.
   - Now we're finally sending our own data in its own data structure. But our own data structure also needs an indication of length and type (two bytes in total). So we have 26 bytes left.

All of which means that we have only 26B to use for the data we want to send over GAP.

And here's what the bottom two layers of structure look like for our particular example - sending manufacturer data:




*The example we use here only requires two data structures, one of 3B, one of 28B (of which two are used for data length and type indications)*

http://www.argenox.com/a-ble-advertising-primer/
https://docs.mbed.com/docs/ble-intros/en/latest/Advanced/CustomGAP/

## 2.3 Investigated Protocols

### 2.3.1 AODV

#### Overview

AODV is a reactive routing protocol designed for use in ad hoc networks. Ad hoc networks are networks which have no pre-existing infrastructure and so AODV is particularly suitable for peer-to-peer BLE networks. In AODV, topology information is only transmitted on demand. Only a single route is ever recorded between a source and destination, and is only maintained while active. An example of this protocol can be seen in figure 1.

When a node wishes to send traffic to a host and no route is known, a route request (RREQ) message is broadcast. At each node the RREQ arrives at, if the node is not the destination and does not have a route recorded, it rebroadcasts the RREQ. A route is considered found when the RREQ message arrives at either the desired host, or to an intermediary node with a valid route entry for the destination. In either case, a route reply message (RREP) is sent back to the originator of the RREQ message. As the RREP message propagates back to the originator, each intermediary node creates a route to the destination.

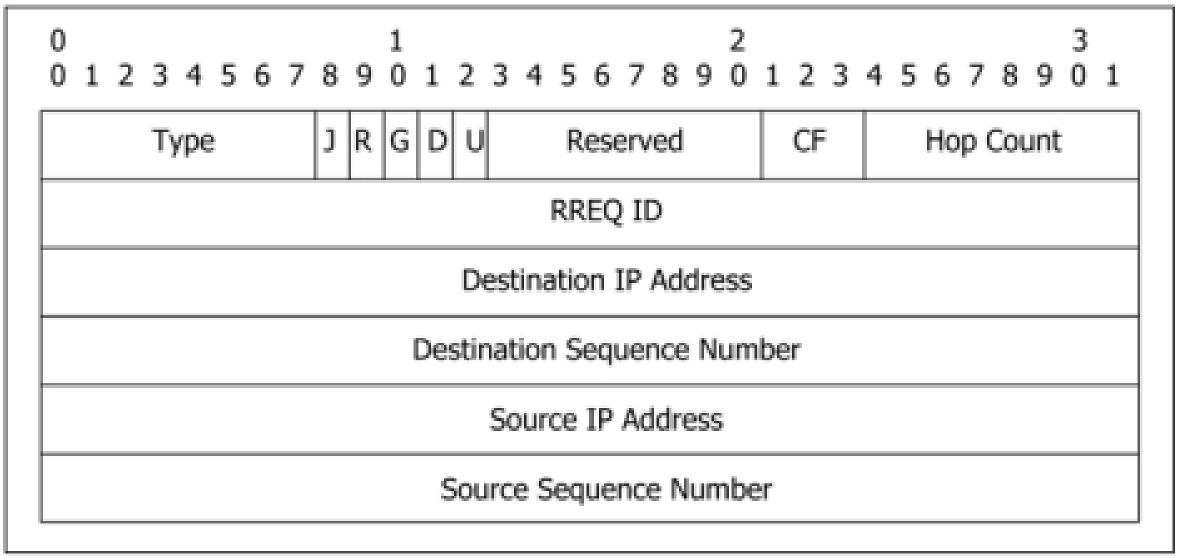
Hello messages can be used to detect link breaks. Nodes periodically broadcast these Hello messages to their neighbours and in the event that a node fails to receive several Hello message from its neighbour, a break is detected. In the event that a node detects an error in one of its known routes, it sends a route error (RERR) message to each of its neighbours.

The AODV protocol has built in sequencing numbers in each RREQ and RREP packet which prevents routing loops being formed, a challenge faced by many routing algorithms. In addition to this, each node maintains its own routing table, keeping the routing process minimal if the host has the required route information in its own routing table.

C. E. Perkins, E. M. Belding-Royer, and S. Das. "Ad hoc OnDemand Distance Vector (AODV) Routing" RFC 3561, July 2003.

**Evaluation**

**BLE Considerations**




Total size: 24 Bytes

### 2.3.2 LOADng

T. Clausen et al. The Lightweight On-demand Ad hoc Distance-vector Routing Protocol - Next Generation (LOADng). In: Internet Engineering Task Force (IETF) Draft (2016)

### 2.3.3 Trickle

P. Levis, T. Clausen, J. Hui, O. Gnawali, J. Ko, The Trickle Algorithm, IETF Internet Draft: draft-ietf-roll-trickle-08, 2011

### 2.3.4 LOAD

K. Kim, S. Daniel Park, G. Montenegro, S. Yoo, N. Kushalnagar, 6LoWPAN Ad Hoc On-Demand Distance Vector Routing (LOAD), IETF Internet Draft: draft-daniel-6lowpan-load-adhoc-routing-03, 2007.

### 2.3.5 ZigBee Cluster Tree protocol

S.C. Erge, ZigBee/IEEE 802.15.4 Summary, 2004. ¡http:// staff.ustc.edu.cn/ustcsse/papers/SR10.

### 2.3.6 Hilow

K. Kim, S. Yoo, J. Park, S.D. Park, J. Lee, Hierarchical Routing over 6LoWPAN (HiLow), IETF: Internet Draft: draft-deniel-6lowpanhilow-hierarchical-routing-00.txt, vol. 38, December 2005.

### 2.3.7 Dymo-Low

K. Kim, S. Park, I. Chakeres, C. Perkins, Dynamic MANET On-demand for 6LoWPAN (DYMO-low) Routing, Internet Draft: draftmontenegro-6lowpan-dymo-low-routing-03, June 2007

### 2.3.8 RPL

#### Overview

RPL is based on the topological concept of Directed Acyclic Graphs (DAGs). The DAG defines a tree-like structure that specifies the default routes between nodes in the LLN.

RPL organizes nodes as Destination-Oriented DAGs (DODAGs), where most popular destination nodes (i.e. sinks) or those providing a default route to the Internet (i.e. gateways) act as the roots of the DAGs.

A network may consist of one or several DODAGs, which form together an RPL instance identified by a unique ID, called RPLInstanceID.

Olfa Gaddour, Anis Kouba, "RPL in a nutshell: A survey", Computer Networks Volume 56, Issue 14, 28 September 2012, Pages 31633178

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
.                             Base                              .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
.                           Option(s)                           .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: RPL Control Message

**Evaluation**

Aishwarya Parasuram, "An Analysis of the RPL Routing Standard for Low Power and Lossy Networks", May 14, 2016

- RPL does not support applications which require a fixed MTU (Maximum Transmission Unit). The control message size can vary greatly and RPL does not give any guarantees on the size.

- RPL does not cater well to a request-response traffic pattern such as utility metering[24]

  Ulrich Herberg and Thomas Clausen. A Comparative Performance Study of the Routing Protocols LOAD and RPL with Bi-Directional Trac in Low-power and Lossy Networks (LLN)

- RPL does not cater to emergency scenarios where there is a high data trac in the network. In case of an emergency, a number of messages might be sent out causing congestion at the DODAG root. It also causes delay and possible packet loss [68]. RPL doesnt specify any mechanism of dealing with data packet loss.

  Weisheng Tang et al. Toward Improved RPL: A Congestion Avoidance Multipath Routing Protocol with Time Factor for Wireless Sensor Networks. In: Journal of Sensors 2016 (2015).

- RPL cannot be used for topologies with a long chain-like structure that contains paths length greater than eight (if raw IPv6 is used) or 64 (while using IPv6 address compression [31]), while being used in non-storing mode. The source routing header can be a maximum of 136 octets which includes an eight octet long header. An IPv6 address is 16 octets long. Hence no more than eight can be accommodated unless address compression is used. If the addresses are compressed, then the path length may not exceed 64.

  Ed. J. Hui and P. Thubert. Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks. In: Internet Engineering Task Force (IETF) 4944 (2011).

- In the current specification there is no support provided for broadcast. RPL only supports unicast to and from the DODAG root. In case broadcast is required, the root has to deliver the data to all nodes, which is very inecient.

- RPL is not suitable for WSNs that contain sensor nodes that can harvest ambient energy from the environment. Such networks are a possible solution for sustaining sensor networks for decades without maintenance, since they can recharge on their own using ambient energy. However in [47], it was found that energy-harvesting sensor networks running RPL produce 40-45% lower goodput than battery-operated sensor networks as the harvesters drop

  Wilbert Nestor Michael C. Tiglao, China Kimberly Paige S. Yu, and Carl C. Dizon. Performance Analysis of RPL in an Ambient Energy Harvesting Wireless Sensor Network. In: The 4th International Conference on Internet Applications, Protocols and Services (NETAPPS) (2015).

## 2.4 Summary

# Chapter 3

# Methodology

## 3.1 Introduction

## 3.2 Protocol Assessment

Research - simulation

## 3.3 Custom Simulation

### 3.3.1 Existing Simulators

### 3.3.2 Custom Simulator Discussion

## 3.4 Analysis of BLE Data Packets

## 3.5 Summary

# Chapter 4

# Implementation

## 4.1 Introduction

## 4.2 nrf51

### 4.2.1 nrf51 DK

### 4.2.2 nrf5 SDK

### 4.2.3 Nordic Python API

## 4.3 Summary

# Chapter 5

# Results and Analysis

## 5.1 Introduction

## 5.2 Energy Measurement Methods

## 5.3 Simulation Results

### 5.3.1 AODV

## 5.4 Medium Scale Evaluation Results

6-8 devices of AODV role call

## 5.5 Summary

# Chapter 6

# Conclusion

## 6.1 Conclusion

# Appendix

...

# Bibliography

[1] G. D. Greenwade, "The Comprehensive Tex Archive Network (CTAN)," *TUGBoat*, vol. 14, no. 3, pp. 342–351, 1993.