Collaborative Attendance Tracking using 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

May 18, 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

May 18, 2017

Summary

This thesis aimed to design and implement an attendance tracking protocol using Bluetooth Low Energy (BLE). The attendance tracking protocol would be implemented in a BLE capable microcontroller which would be incorporated in an ID card along with a coin cell battery.

Attendance tracking is performed in areas of high population densities such as lecture theatres and evacuation assembly points. These are ideal network topologies for multi-hop routing protocols, which can take advantage of the high density, reducing the required transmission power, and consequently, power consumption of devices in the network.

The Ad-hoc on Demand Distance Vector (AODV) routing protocol is chosen as a suitable protocol for attendance tracking after some analysis of existing routing protocols and research investigating their performance. Two attendance tracking protocols are designed, *Roll Call* and *Route-toZero*, that are based on AODV's route discovery mechanism. This mechanism is implemented on Nordic Semiconductor's nRF5 series devices using their nRF SDK. These two attendance tracking protocol designs are then implemented on top of the AODV implementation.

The attendance tracking applications are evaluated in a physical network of ten nodes comprised of nRF51 devices. Metrics including the node discovery rate, number of packets sent and processed per node, and number of packets sent and processed per hop are measured. The viability of the attendance tracking protocol and AODV implementation are shown, with a best recorded node discovery rate of 1.34 nodes discovered per second.

Based on these results, a number of areas for further work are outlined. One of these is the improvement of the AODV implementation to include route maintenance to handle route errors. This would be required in larger networks and would allow for mobility in nodes while tracking attendance. Another area for further work is the implementation of these designs in a simulator in order to measure performance in larger and denser networks. The final area is the implementation of other protocols which might be suitable for attendance tracking.

Collaborative Attendance Tracking using Bluetooth Low Energy

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

Supervisor: Jonathan Dukes

The possibilities to improve the digitisation of attendance tracking have grown with the recent explosion of the Internet of Things. Technologies such as Bluetooth Low Energy (BLE) allow for the development of applications that can run on extremely small microcontrollers and can last for years on a coin cell battery.

The aim of this thesis is to design an attendance tracking application based on an existing low power ad-hoc routing protocol that is capable of lasting for the lifetime of an ID card, which a microcontroller and batter could be incorporated into.

Two attendance tracking protocols are designed that are based on the Ad-hoc On Demand Distance Vector (AODV) routing protocol's route discovery mechanism. This route discovery mechanism is implemented on Nordic Semiconductor's BLE stack. Both of these attendance tracking protocol designs are then built on top of this AODV implementation.

The attendance tracking application is evaluated in a physical network of ten nodes, comprised of nRF51 devices. The viability of the attendance tracking protocol and AODV implementation are shown, with a best recorded node discovery rate of 1.34 nodes discovered per second.

Acknowledgments

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

BEN LYNCH

University of Dublin, Trinity College
September 2017

Contents

List of Figures

Chapter 1

Introduction

1.1 Introduction

Attendance tracking is commonly used in places such as universities, to determine student attendance in lectures, and in organisations, for taking employee attendance and for assembly scenarios such as in fire evacuation procedures.

Attendance tracking has mainly been achieved by performing manual roll calls using pen and paper, but has had implementations using technologies such as barcodes, Radio-frequency identification (RFID), and Near field communication (NFC). In the case of barcodes and NFC, these technologies require users to manually scan themselves for their attendance to be taken. In the case of RFID, tag collision can occur when numerous tags in the same area respond at the same time. Although implementations exist using these technologies, they have limitations and are not always effortless for the user.

Bluetooth Low Energy (BLE) is a version of Bluetooth which was designed specifically with the Internet of Things (IoT) in mind. It has ultra-low peak, average, and idle power consumption and operates in the industrial, scientific, and medical (ISM) radio band (2.400 - 2.4835 GHz). BLE has two modes of communication, broadcasting and connections, with broadcasting most commonly being used to discover devices in order to establish connections.

Microcontrollers with BLE radios, such as Nordic Semiconductor's nRF51 series devices, are extremely small (6mm x 6mm x 1mm) and relatively cheap (from 2 euro).

It is feasible to incorporate these microcontrollers into student or organisation ID cards. With the addition of energy efficient applications with a focus on radio duty cycling, a microcontroller should be able to run on a coin cell battery for the lifecycle of the ID card.

High crowd density is one of the main characteristics of scenarios where attendance tracking is commonly used. With the use of multi-hop routing, this high crowd density can be exploited to reduce the required transmission range of devices, which in turn reduces the transmission power required.

The combination of a low power routing protocol, such as the Ad-hoc On Demand Distance Vector routing protocol, with the exclusive use of BLE broadcasting results in an implementation with small packet sizes, low packet processing times, and removes the overhead of connection establishment, creating an ideal framework for efficient and long lasting attendance tracking applications.

1.2 Problem Area

As mentioned, the attendance tracking is focused in areas of high crowd densities. Devices in the location where attendance tracking is taking place will form wireless, mobile ad-hoc networks (MANET).

Attendance tracking in MANETs is a form of network member discovery and so one of the most important requirements in the selection of an appropriate routing protocol is its performance in the discovery of new routes. Once a route is established, member presence knowledge will have been acquired and the route will only be used in the establishment of other routes.

Energy efficiency will be one of the main requirements for an attendance tracking application, as it will have to last for the lifetime of the user's ID card. For the purpose of this research we consider that lifetime to be a year. The main mechanism to achieve this energy efficiency is radio duty cycling. That is, the radio of the device will remain in a low power mode for the majority of its life, only entering a higher power mode when attendance is being taken.

1.3 Motivation for this research

The Internet of Things (IoT) has recently been in the spotlight as one of the fastest growing information technology sectors. According to a report [1] from the financial services company IHS Markit, the number of IoT devices is expected to rise by fifteen percent year-over-year to reach twenty billion in 2017.

According to the IoT Developer Survey 2017 [2], Bluetooth Low Energy has seen a growth of twelve percent in its use as a connectivity protocol for IoT solutions since 2015. In addition to this, the majority of major android branded smartphones (Android OS 4.3 and above), iOS devices (model 4S and onwards), and Windows based smart phones are BLE enabled. The early adoption of BLE in Apple's iBeacon has also aided in the wider adoption of BLE technology.

The explosion of the IoT and the growing adoption rate of BLE make it an exciting technology to work with. It is still a relatively new technology and further research into its capabilities and possible applications is still required.

1.4 Objectives

The goals of this dissertation are as follows:

- The design of an attendance tracking protocol based on an existing routing protocol suitable for member identification in an ad-hoc network.
- The implementation of a module encapsulating the functionality of the chosen routing protocol.
- The implementation of an application which makes use of the above module to track attendance.
- An analysis of the performance of the protocol in BLE.

1.5 Dissertation Outline

This dissertation begins with a literature review in Chapter 2 which provides an understanding of radio duty cycling, an overview of appropriate routing protocols, and a review of their performance. Chapter 3 provides a description of the key aspects of BLE for the development of applications, a discussion of design decisions related to the implementation of a routing protocol for BLE, as well as a description of two protocols for attendance tracking. Chapter 4 discusses the implementation of these protocols and provides some security considerations with regards to the implementation. Chapter 5 provides an overview of the evaluation methodology, results from medium scale testing, and an analysis of these results. Finally, Chapter 6 summarises the contributions made by this dissertation, a general discussion on the outcome of this thesis, and areas for future work.

Chapter 2

Background

2.1 Introduction

This chapter investigates some of the techniques currently used in radio duty cycling, a key component in low energy applications. Specifically, an overview of the X-MAC, Box-MAC-2, and ContikiMAC radio duty cycling protocols is provided along with the benefits associated with the techniques used.

An overview of well known ad-hoc routing protocols is also provided with a focus on route discovery as a mechanism for attendance tracking, and packet size, as BLE advertisement packets are limited in size. A brief overview on research related to the performance of these protocols is then provided.

2.2 Radio duty cycling

Power consumption is extremely important for nodes that wish to achieve a long network lifetime. It is not possible to achieve this long life time if a radio transceiver is permanently powered on as even low-power transceivers consume too much power. By turning the radio transceiver off a node can conserve energy and achieve a long network lifetime, however when the transceiver is off, the node cannot send or receive messages. This is solved by keeping the radio off only in-between the reception and transmission of messages.

2.2.1 Low Power Listening

Low Power Listening (LPL) [3] is a duty cycling technique in which receivers periodically turn on their radios to identify if there is activity in the radio medium. If activity is detected, the receiving node's radio is kept on for a longer period so that data can be exchanged. an illustration of LPL is provided in figure 2.1.

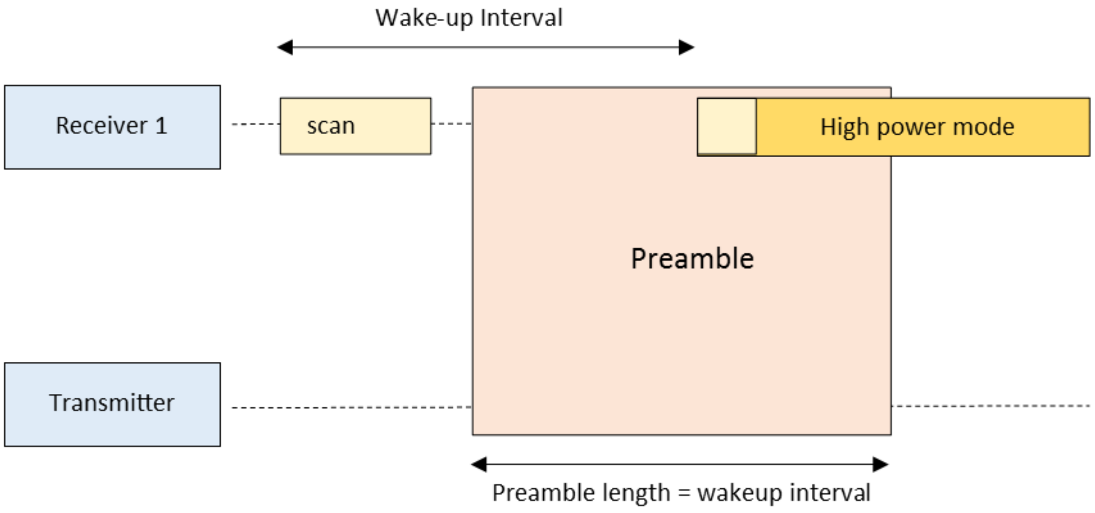


Figure 2.1: Low Power Listening

Transmitting nodes send a preamble stream which is at least as long as the receiving node's wake-up interval, ensuring that the receiver will have their radio on during the transmission of the preamble. When a receiving node detects a preamble stream, they respond with an acknowledgement, and the transmitting node terminates the preamble transmission.

LPL was designed for single node wake-up. Broadcast mode is possible, however, this requires the transmission of maximum length preamble streams as the transmitting node does not know when all intended receivers have woken up, and single node wake-up is rendered impossible as any node detecting the preamble will keep their radio on afterwards.

2.2.2 Low Power Probing

Low Power Probing (LPP) [4] is a duty cycling technique in which the roles from LPL are reversed. Instead of receivers scanning at wake-up intervals, they periodically broadcast a probe package. an illustration of LPP is provided in figure 2.2.

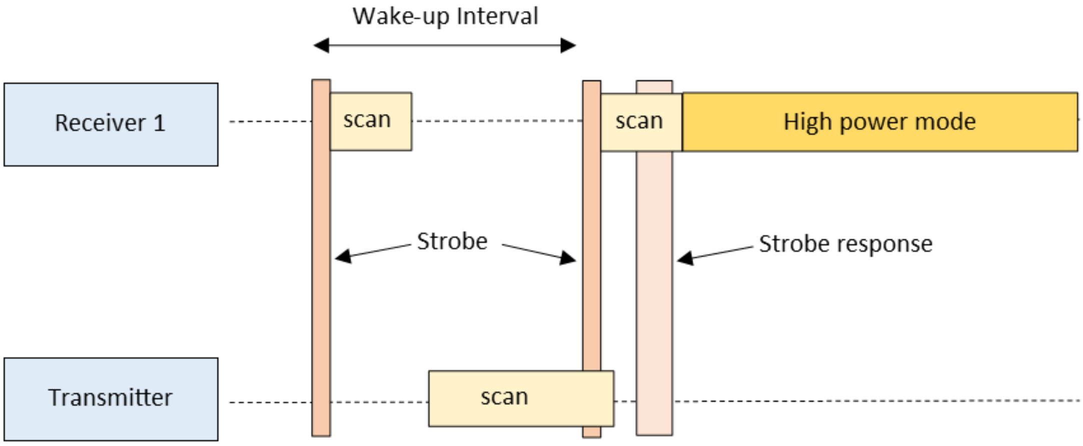


Figure 2.2: Low Power Probing

Transmitting nodes scan until a probe is detected, at which point they immediately transmit a probe response. The receiver scans for a short period after sending a probe, ensuring it will receive any probe responses intended for it. Upon receiving data, the receiver will enter a high power mode, keeping their radio on for a longer period.

2.2.3 X-MAC

X-MAC [5] is an LPL based radio duty cycling protocol. X-MAC implements a variation of LPL in which the transmitting node sends a series of short preamble packets which include the address of the target destination. Periodic gaps are used during the transmission of these shorter preambles. By including the destination address, receiving nodes are able to turn their radios off immediately if they are not the destination specified. The periodic gaps in the transmitting node's preamble stream are used by receiving nodes to send acknowledgements. This series of short preambles approximates the continuous preamble from standard LPL. The X-MAC protocol is illustrated in figure 2.3.

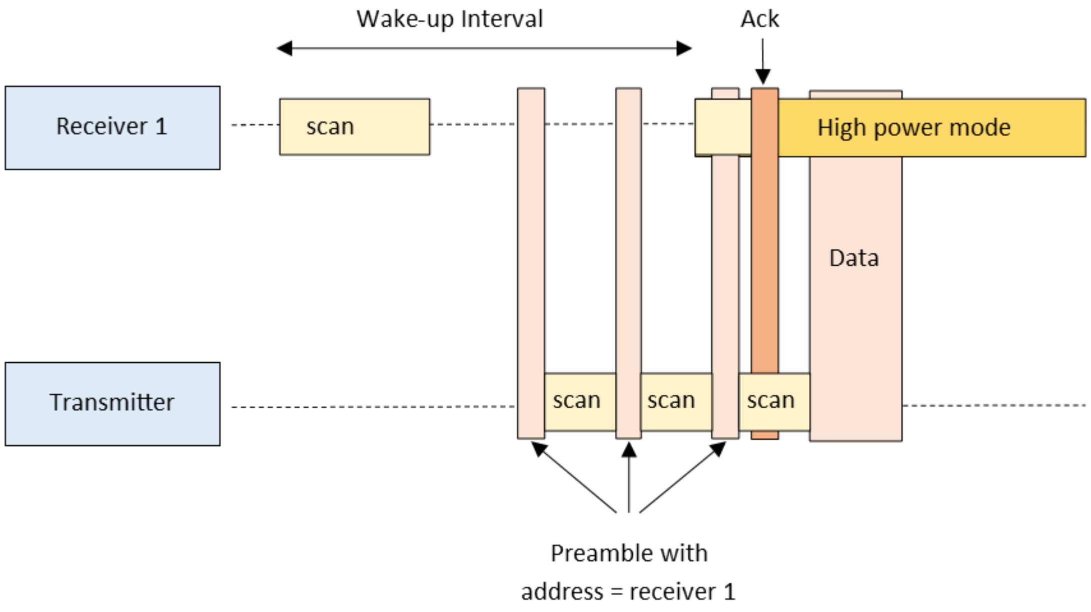


Figure 2.3: X-MAC

The inclusion of a destination address reduces the wake-up cost for nodes which preambles are not intended for, making single-node wake-up feasible using broadcasting.

2.2.4 BoX-MAC-2

BoX-MAC-2 [6] is an adaptation of X-MAC in which a transmitting node includes the entire data packet within the preamble, instead of the destination address. This eliminates the need for a transmitting node to send further data after receiving an acknowledgement from a receiving node. The Box-MAC-2 protocol is illustrated in figure 2.4.

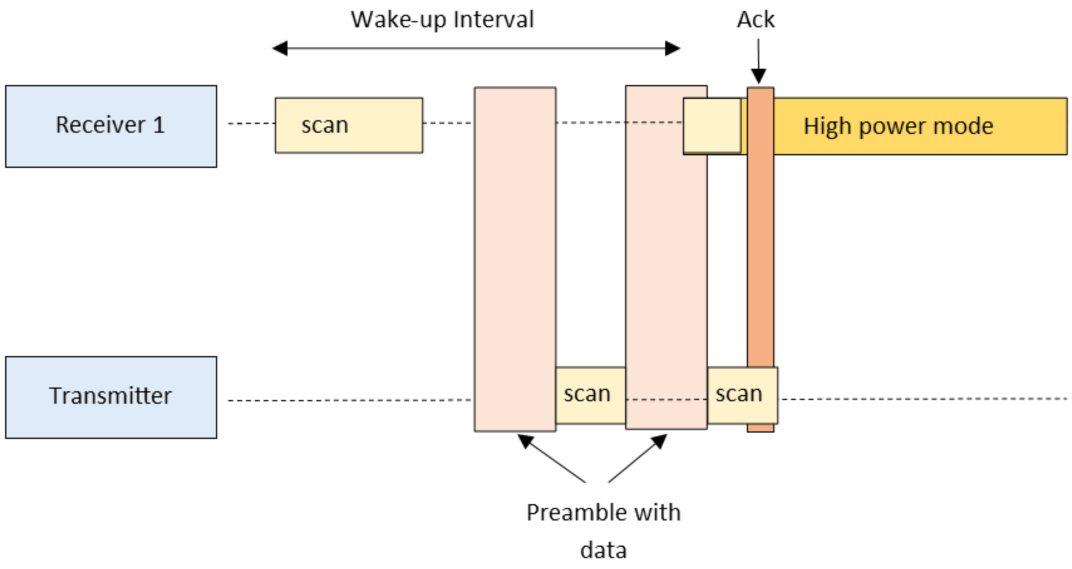


Figure 2.4: Box-MAC-2

2.2.5 ContikiMAC

ContikiMAC [7] is a radio duty cycling protocol that uses LPL and is based on X-MAC. It has two unique mechanisms to reduce energy consumption which can be seen in figure 2.5.

The first mechanism is *Fastsleep*. With this, nodes turn off their radios if one of three conditions are met:

- if a node detects radio activity longer than the largest packet length in the protocol.
- if detected radio activity is followed by a silence period which is longer than two successive transmission intervals.
- if valid radio activity is identified but no start of packet could be detected.

The second mechanism is *Transmission Phase Lock*. With this, each node maintains the wake-up phases of their neighbours by noting the time at which it saw link layer acknowledgements from them during its wake-up phase. With this information a node can send information to its neighbour when it knows the neighbour will be awake.

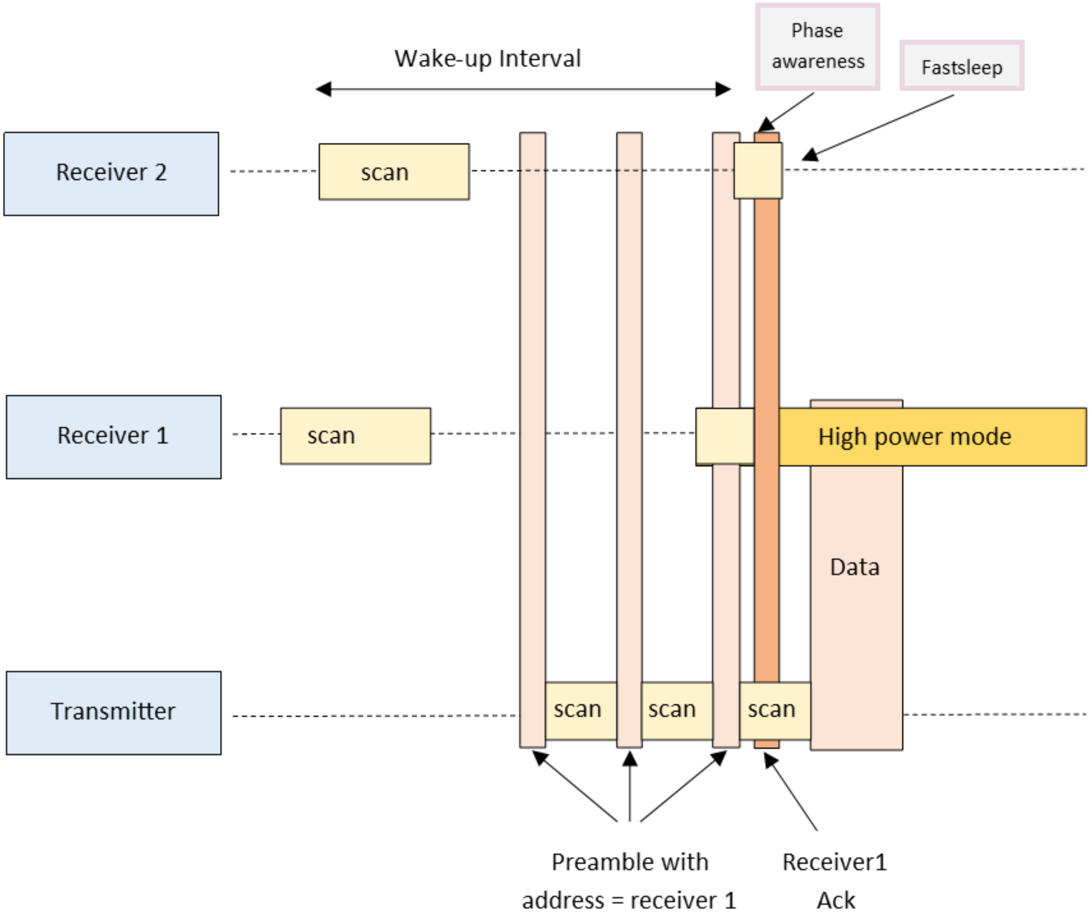


Figure 2.5: ContikMAC

According to the ContikiMAC radio duty cycling protocol report [7], in which the protocol was evaluated in the Contiki simulation environment, it was found that the Fastsleep and Phase Lock mechanisms reduce network power consumption by between 10 and 80 percent, depending on the wakeup frequencies of devices.

2.3 Routing Protocols

2.3.1 Ad-hoc On Demand Distance Vector routing

Overview

Ad-hoc On Demand Distance Vector routing (AODV) [8] is a reactive routing protocol designed for use in mobile ad-hoc 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 route discovery in AODV can be seen in figure 2.6.

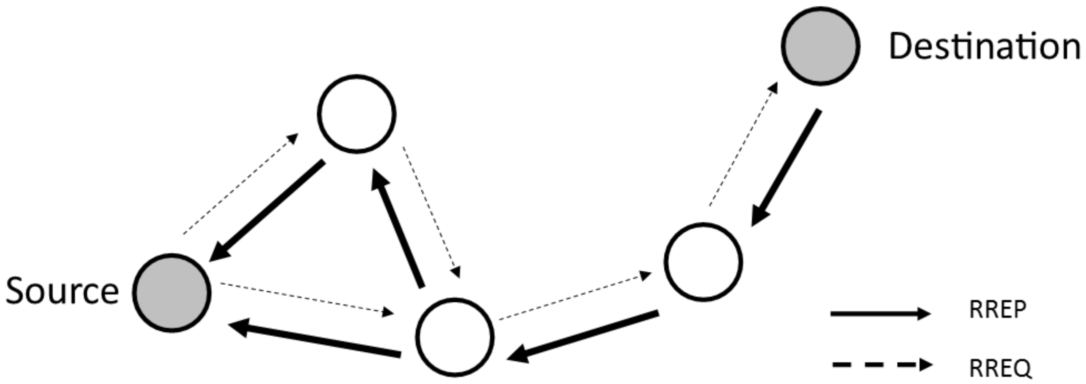


Figure 2.6: AODV route discovery

The AODV protocol has built in sequencing numbers in each of its control packets 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. Entries in the routing table have a time to live and if not used will expire, limiting the memory overhead to routes that are being used, as opposed to all possible routes.

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, the following processing takes place:

- The node creates or updates a route to the previous hop from which the RREQ was received.
- If a RREQ with the same source address and RREQ ID has been previously received within a certain time period, the RREQ is discarded.
- The node increments the Hop Count within the RREQ by one.
- The node creates or updates a reverse route to the source address of the RREQ. This is required in the event a RREP is received with the source of the RREQ as the destination.
- If the node is the destination of the RREQ, it discards the RREQ, generates a RREP, and sends the RREP to the next hop of the reverse route.
- If the node is not the destination but has a routing table entry for the destination, it can generate a route reply (RREP) which it sends to the RREQ source.
- If the node is not the destination and does not have any routing information regarding the destination, it re-broadcasts the RREQ message.

At each node a RREP arrives at, the following processing takes place:

- The node creates or updates a route to the previous hop from which the RREP was received.
- The node increments the Hop Count within the RREP by one.
- The node creates or updates a route to the source of the RREP.
- The node sends the RREP to the next hop indicated in the routing table entry for the RREP's destination address.

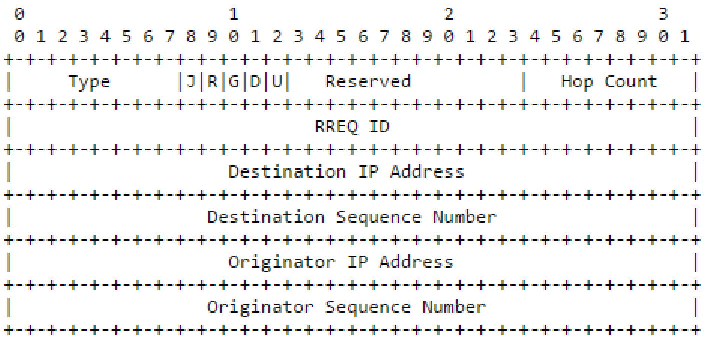
In summary, a route is considered found when the RREQ message arrives at either the desired host, or to an intermediary node with a valid route entry for the destination. In either case, a RREP is sent back to the originator of the RREQ message. As the

RREP message propagates back to the originator, each intermediary node creates a route to the destination (the source of the RREP).

Route maintenance is performed using Hello messages which detect link breaks. Nodes periodically broadcast these Hello messages to their neighbours and in the event that a node fails to receive a Hello message within a certain period 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.

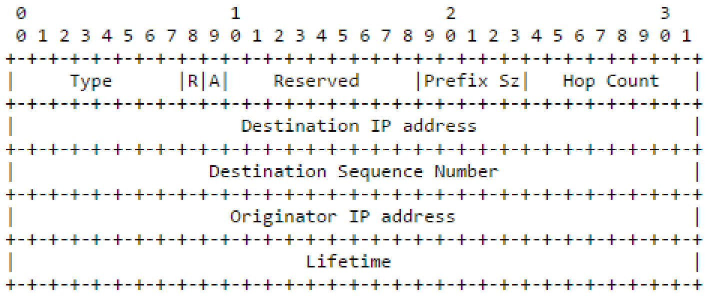
Control Packets

Figures 2.7 and 2.8 show the format of AODV's two primary control packets, RREQs and RREPs.



```
 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      |J|R|G|D|U|   Reserved          |   Hop Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            RREQ ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination IP Address                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Destination Sequence Number                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Originator IP Address                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Originator Sequence Number                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2.7: AODV RREQ packet format



```
 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      |R|A|    Reserved     |Prefix Sz|   Hop Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Destination IP address                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Destination Sequence Number                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Originator IP address                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Lifetime                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2.8: AODV RREP packet format

These are static sized packets with the RREQ packet having a total size of twenty-four bytes and the RREP packet having a total size of twenty bytes.

2.3.2 Dynamic Source Routing

Dynamic Source Routing (DSR) [9] is a routing protocol designed for multi-hop wireless ad-hoc networks of mobile devices. DSR is based on a type of routing called *source routing*. In source routing, the sender of a packet determines the complete route in which a packet is to be forwarded through.

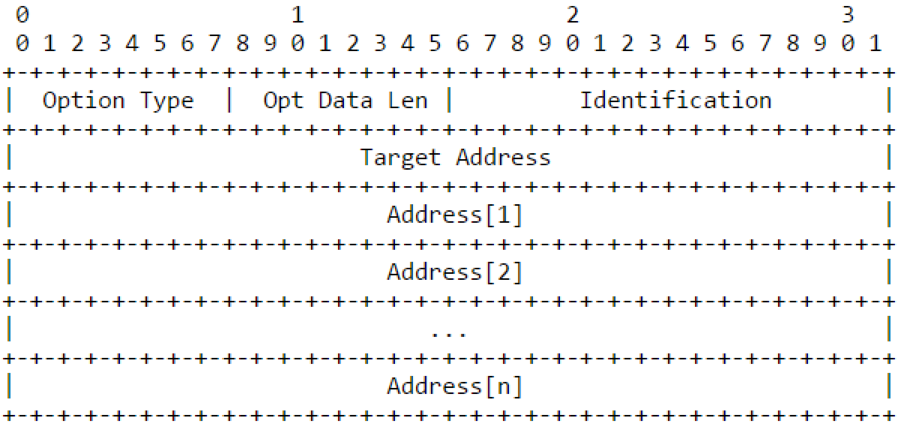
Each node in the network maintains a route cache in which it caches source routes which it has learned of. When sending a packet, the node first checks for a valid route to the destination in its route cache. If a route is found, it is used to transmit the packet. If no route is found, the node can discover the route using DSRs route discovery protocol.

Nodes initiating route discovery broadcast a route request (RREQ) packet identifying both the source of the RREQ and the target node. Each RREQ message contains a *route record*, which is the accumulated sequence of hops taken by the RREQ packet as it propagates through the network. Upon reaching the target destination, a *route response* (RREP) packet is generated containing the route by which the RREQ arrived through. This RREP is then returned via a route specified in the nodes route cache if one exists, or by reversing the route contained in the RREQ.

Route maintenance is performed by every node in a route. If a node exceeds it's maximum number of retransmissions for a packet it is forwarding without receiving a response, it must send a *route error* message to the original sender of the packet.

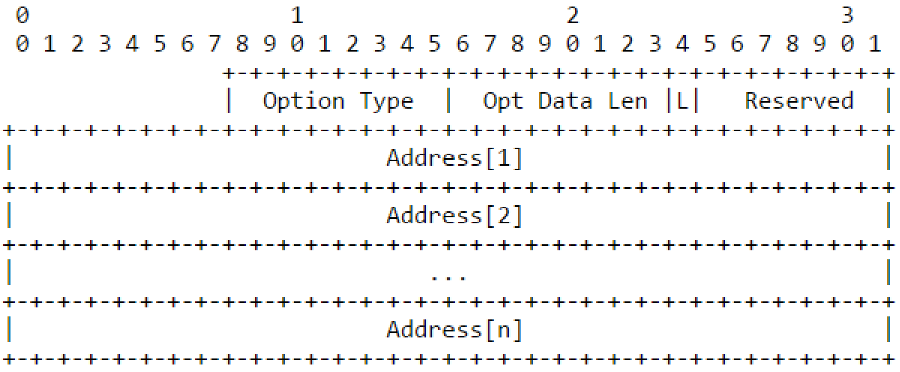
Control Packets

Figure 2.10 and figure 2.10 show the format of DSR's two primary control packets, RREQs and RREPs.



```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Option Type  |  Opt Data Len |         Identification        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Target Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Address[1]                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Address[2]                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Address[n]                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2.9: DSR RREQ packet format



```
 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
                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                |  Option Type  |  Opt Data Len |L|   Reserved   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Address[1]                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Address[2]                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Address[n]                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2.10: DSR RREP packet format

These are dynamic sized packets, with the RREQ having an eight byte header with n possible addresses each of which are four bytes. This list grows as the RREQ propagates through the network. The RREP packet has a four byte header with n possible addresses, again, each of which are four bytes each. This list identifies the route the packet should take.

2.3.3 Optimised Link State Routing

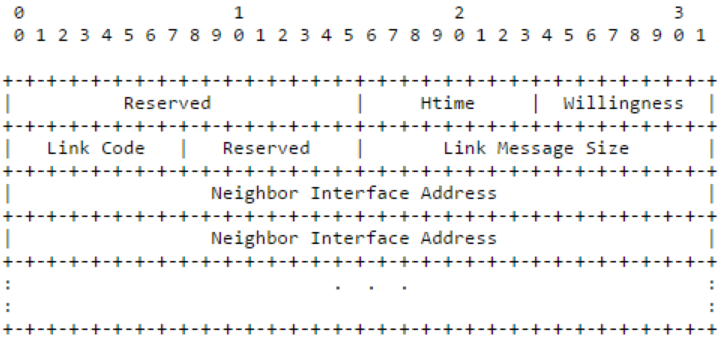
Optimised Link State Routing (OLSR) [10] is proactive IP routing protocol optimized for mobile ad hoc networks. Proactive protocols are those in which routing information for all nodes in the network is discovered and maintained before the transmission of packets. It is an optimisation of a pure link state protocol for mobile ad-hoc networks. In pure link state protocols, all adjacent network links are flooded through the entire network. In DSDV, only a subset of these links are declared. This subset is known as the *multipoint relay selectors*. In addition to this, flooding of control traffic is minimised as nodes only use the selected nodes, called *multipoint relays* (MPR) to transmit traffic to the rest of the network.

HELLO messages are used for neighbour sensing. They are broadcast to all nodes within one-hop but are not relayed to further nodes. They contain information relating to neighbours and their link status. This allows a node to learn about nodes up to two hops away.

Topology Control (TC) messages are periodically sent throughout the network using MPRs. These messages contain a list of neighbours who have selected the transmitting node as an MPR. The information in these nodes are used by receiving nodes to build topology tables, which identify the MPRs used by each node in the network, and are used to calculate routing tables.

Control Packets

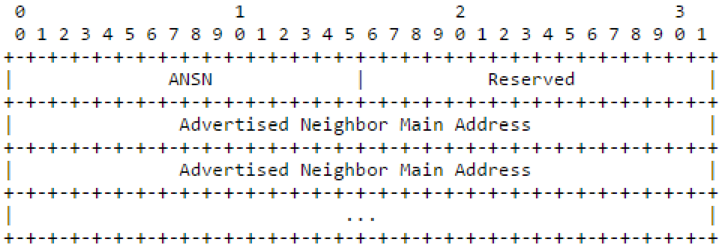
Figure 2.11 and figure 2.12 show the format for OSLR's two primary control packets, HELLO and TC.



```
 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

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Reserved             |     Htime     |  Willingness  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Link Code   |   Reserved    |       Link Message Size       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Neighbor Interface Address                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Neighbor Interface Address                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                             .  .  .                           :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2.11: OLSR HELLO packet format



```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              ANSN             |           Reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Advertised Neighbor Main Address                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Advertised Neighbor Main Address                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2.12: OLSR TC packet format

The HELLO packet contains eight bytes of header information and a list of n neighbour addresses which are four bytes each. The TC packet contains a four byte header and a list of n node addresses which are again four bytes each.

2.3.4 Destination Sequenced Distance Vector Routing

Destination Sequenced Distance Vector Routing (DSDV) [11] is a table driven routing protocol designed for ad hoc mobile networks. DSDV uses the Bellman-Ford algorithm with a hop count metric to calculate paths. The Bellman-Ford algorithm [12] computes the shortest path from a source vertex to all other vertices in a weighted graph. DSDV is a proactive protocol, meaning the routing information for all nodes in the network is maintained in the routing table and routes are added and updated by nodes exchanging table information at regular intervals and through triggers mechanisms employed by the protocol.

Each node in the network maintains its own sequence number which is independently chosen. Each time a periodic update is made, a node increments its sequence number by two. Every update sent by a node includes their sequence number to eliminate routing loops.

Control Packets

There are two defined packets in DSDV for advertising routing table information.

The first type of packet is the *full dump*. Full dumps carry all available routing information from a node and are intended for infrequent use. Full dump messages will most likely require multiple network protocol data units (NPDU).

The second type of packet is the *incremental*. Incremental packets contain routing information which has changed since the last full dump. Incremental messages are intended to fit within a single NPDU.

2.3.5 IPv6 Routing Protocol for Low-Power and Lossy Networks

Overview

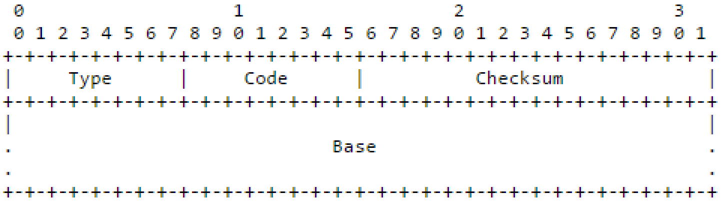
The Routing Protocol for Low-Power and Lossy Networks (RPL) [13] is a distance vector IPv6 routing protocol designed for wireless ad-hoc sensor networks. The protocol specifies how to build a *Destination Oriented Directed Acyclic Graph* (DODAG) using an objective function and a set of metrics and constraints. The objective function operates on this set of metrics and constraints. An example of an DODAG objective is to "Find the path with the lowest hop count(metric) while avoiding non-encrypted links (constraint)". An *RPLInstance* is comprised of one or more DODAGs and multiple RPLInstances can be active in the network at a time, with nodes having multiple objective functions active at once.

The graph building process begins at a root node. The root broadcasts information about the DODAG formation using DODAG Information Objects (DIO) packets. These packets carry relevant network information that allows nodes to discover the new RPLInstance, learn its configuration parameters, and to select a set of parent nodes. Nodes receiving DIOs decide whether to join the DODAG according to their objective function. Nodes joining the DODAG compute their rank within the DODAG, which is an indication of their co-ordinates within the graph hierarchy, and re-broadcasts the DIO message to their neighbours. The transmission of DIO packets build routes in the downwards direction from the root to leaf nodes. Nodes in the network can broadcast DODAG Information Solicitation (DIS) messages to solicit a DIO from surrounding RPL nodes.

Point-to-point communications is achieved by sending packets 'up' the graph to a common ancestor, at which point the packet is forwarded 'down' to the desired destination.

Control Packets

Figure 2.13 illustrates the general format of RPL control packets. These packets have a four byte header and a base which contains DIS and DIO packets.



```
 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                              .
.                                                               .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2.13: RPL control packet format

DIS packets have a length of four bytes and can carry up to three IPV6 packet options which make up one byte.

DIO packets are seven bytes long, including a 128 bit DODAGID, which is the IPv6 address belonging to the DODAG root.

2.4 Protocol Performance

This section provides an overview of some of the research conducted comparing the protocols found in section 2.3.

A performance comparison was performed between AODV and DSR in [14]. The comparison was performed using a simulation based on the NS-2 network simulator and focuses on low mobility networks. The performance metrics measured include: Packet Delivery Ratio (PDR) and End to end delay. The authors observed that AODV and DSR have very similar PDRs in static networks, but AODV has improved PDR with increased node movement. With respect to end to end delay, they observed that DSR marginally outperformed AODV in their simulations. Overall, they concluded that AODV is preferred to DSR due to its more efficient use of bandwidth.

A performance comparison was performed between AODV and DSDV in [15]. The comparison was performed using a simulator called MobiREAL which is also based on the NS2 network simulator. The performance metrics measured include: Network throughput and End to end delay. The authors observed that AODV had a PDR of seventy to ninety percent, while DSDV had a PDR of fifty to seventy-five percent.

They observed high end to end delay initially in AODV that reduces with time, and the inverse in DSDV, with a low end to end delay in the beginning that gradually increases. They concluded that the performance of AODV was better than DSDV for real time applications.

A performance comparison was performed between RPL and a lightweight version of AODV called LOADng in [16]. The simulated network is a static network which reflects the Home Automation [17] scenario. The comparison was performed using the Cooja simulator on the Contiki OS and uses the ContikiRPL implementation and a basic implementation of LOADng. The performance metrics measured include: End to end delay, Average hop distance, and Routing overhead. The authors observed that RPL provides shorter routes and a smaller spanning tree depth compared to LOADng in dense network topologies. They observed that the routing overhead of RPL largely depends on the choice of network parameters chosen but has less memory requirements than LOADng. They concluded that RPL performed better than LOADng but has a much higher implementation complexity. It is also worth noting that the LOADng implementation was unoptimised. If it was optimised it may have had better performance.

A performance comparison was performed between DSDV, AODV, and DSR in [18]. The comparison was performed using the NS-2.34 simulation tool. The performance metrics measured include: PDR, Network throughput, End to end delay, and Routing overhead. The authors observed that DSDV has a higher routing load and lower throughput compared to both AODV and DSR. AODV and DSR have extremely similar PDRs, end to end delays, and throughput in scenarios of low and high mobility, but AODV has a lower routing load in all tested scenarios.

A performance comparison was performed between OLSR, AODV, and DSDV in [19]. The comparison was performed with an NS-2 simulation with the simulated topologies being based on tactical networks for ships and sensor-based network nodes. The performance metrics used include: PDR, End to end delay, Routing overhead, and Normalised routing load. The authors observed that in scenarios of mobility, AODV outperforms both of the other protocols, with OLSR being a vast improvement over DSDV. In the scenario of a static sensor network OLSR outperformed AODV in all cases, with AODV performing poorly with high node density but still outperforming DSDV.

From the above papers, AODV either outperforms or performs similarly to DSR regardless of network density or node mobility. AODV is superior to DSDV with respect to PDR, Routing Load, and Network throughput. It has worse end to end delay initially, but its end to end delay gradually improves, surpassing DSDV. AODV outperforms OLSR in all metrics in networks with node mobility, but the reverse is true in static networks. An unoptimised implementation of AODV is outperformed by RPL in all metrics within the Home Automation environment, a fully static sensor network. Although the above research does not focus on route discovery, it shows AODV as the most suitable protocol for networks with mobility and that it still performs well in static networks.

2.5 Summary

There are two primary methods of radio duty cycling, Low Power Listening and Low Power Probing. In LPL, duty cycling nodes scan for preamble streams at wake-up intervals. In LPP, duty cycling nodes advertise strobe packets at wake-up intervals followed by a short period of scanning. The X-MAC and Box-MAC-2 protocols have improved preamble usage, and ContikiMAC implements Fast Sleep and Transmission Phase Lock techniques to further improve the effectiveness of LPL radio duty cycling.

There are many suitable routing protocols for low power ad-hoc networks. AODV is a reactive routing protocol that establishes routes on demand, uses the routing table at each intermediate device to establish routes, and has static sized packets. DSR is another reactive routing protocol but uses source routing in which the route is known before transmitting and so has dynamic sized packets. OLSR is a proactive protocol, meaning every node maintains a routing table representing the entire topology of the network. It is an optimisation of a pure link state protocol which uses Multipoint Relays to reduce the flooding of table information throughout the network. DSDV is another proactive protocol in which routing table information is periodically shared between neighbouring nodes and a Bellman-Ford algorithm is used to calculate paths. Finally, RPL is a protocol which specifies how to build a Destination Oriented Directed Acyclic Graph for sensor networks. Nodes in the network represent vertices and maintain information about their parents so that a path exists between any node and the root node.

There have been some performance comparisons made between some of the protocols investigated in this chapter, all of which have been made using NS-2 or Cooja simulations. The main metrics measured are the Packet Delivery Ratio, End to end delay, and Routing overhead. AODV and DSR appear to have similar performance in static and mobile networks with DSR having a worse routing overhead. In the one experiment featuring RPL, it appears to outperform a light weight, unoptimised version of AODV, but has a much higher implementation complexity. In the one experiment featuring OLSR, it appears to have better overall performance than AODV in static topologies. DSDV appears to have the poorest performance out of all the routing protocols.

Chapter 3

Design

3.1 Introduction

This chapter provides an overview of the aspects of BLE that will effect the design of any protocols for the technology. Design decisions regarding certain aspects of the AODV routing protocol are discussed, and the designs for two attendance tracking protocols are outlined, *Roll Call* and *Route to Zero*, which are built on top of AODV's route discovery mechanism.

3.2 Bluetooth Low Energy

Bluetooth standards are governed by the Bluetooth Special Interest Group (SIG) [20]. This includes the functionality of the technology, its technical operations, certification, interoperability, and standard evolution. In 2009 SIG announced the Bluetooth Core Specification version 4.0 which included BLE. BLE was a radical departure from standard Bluetooth making both technologies incompatible with one another. The Bluetooth Core Specification 5.0 [21] was released in December 2016 and provides improved range, speed, and message capacity over the previous specifications.

3.2.1 The BLE Protocol Stack

BLE devices are divided into three parts: controller, host, and application. Each of these parts are subdivided further into layers that provide the various functionality that is required to operate. The full stack is illustrated in figure 3.1.

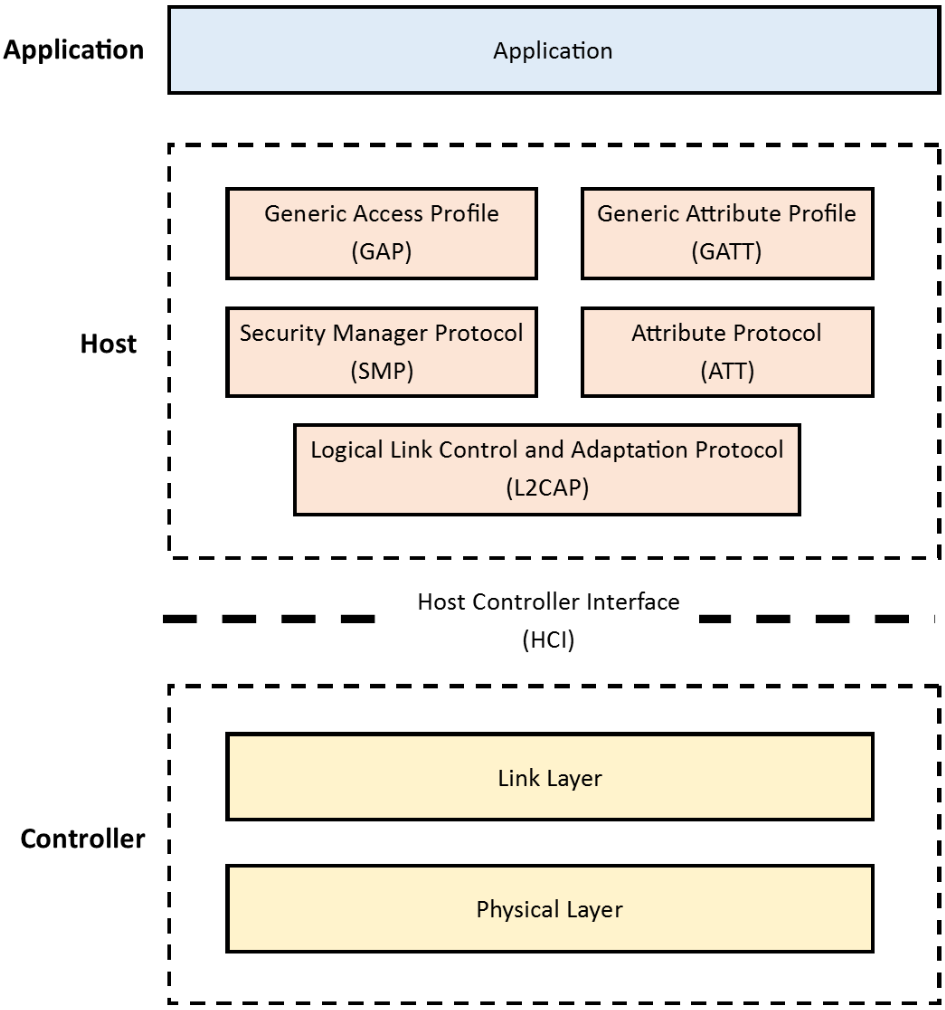

Application
Application
Generic Access Profile (GAP)
Generic Attribute Profile (GATT)
Host
Security Manager Protocol (SMP)
Attribute Protocol (ATT)
Logical Link Control and Adaptation Protocol (L2CAP)
Host Controller Interface (HCI)
Link Layer
Controller
Physical Layer

Figure 3.1: The BLE protocol stack

The application is the highest layer of the stack and is responsible for the management of data and implementation of logic that is relevant to the current use case.

The host covers the upper layers of the BLE protocol stack and includes both protocols and profiles which are described in more detail below.

The controller is comprised of the two lowest layers in the stack: the physical layer and the link layer.

The host can communicate with the controller through the Host Controller Interface (HCI). The HCI decouples the host and controller, allowing them to be swapped without affecting the other. It exposes enough BLE functionality that allows the host to perform operations such as scanning and advertising.

3.2.2 Physical Layer

The physical layer is comprised of a device's radio, which performs the modulation and demodulation of analog signals. The radio uses the ISM frequency band with 40 channels, from 2.4 GHz to 2.4835 GHz. Three of these channels are used for advertising, and the remaining 37 are used for connections. The advertisement channels are located at the start, middle, and end of the band. If any single advertisement channel is blocked, the others are likely to be free due to the frequency difference in channel placement. For this reason, advertising on all channels is recommended for any single advertisement packet.

3.2.3 Link Layer

The Link layer (LL) directly interfaces with the physical layer and is a combination of hardware and software. The LL manages Bluetooth Device addresses, 48 bit values which uniquely identify devices. It is also in charge of establishing connections. In the case of connections, the LL manages connection intervals and is capable of configuring encryption. Consequently, it defines the following roles for devices:

- Advertiser - a device which sends advertisement packets
- Scanner - a device which scans for advertisement packets
- Master - a device which initiates and manages connections
- Slave - a device which accepts connection requests

3.2.4 Protocols

Attribute Protocol

The Attribute Protocol (ATT) is a transport protocol which defines a basic data unit and how it can be exchanged between devices.

The *attribute* is the basic data unit of the protocol and is composed of three elements:

- a 16 bit handle - uniquely identifies an attribute.
- a UUID - defines the type of attribute.
- a value - a data value of a certain length.

The protocol is client-server focused. Devices can operate as servers, storing attributes in non-volatile memory, and clients can use the protocol to send read and write requests to servers. Clients specify the attribute they wish to read or write using the attribute's handle value.

Security Manager Protocol

The Security Manager Protocol (SMP) provides functionality for devices to generate and exchange security keys. It defines the following two roles:

- Initiator - the link layer master
- Responder - the link layer slave

The SMP provides support for *Pairing*, *Bonding*, and *Encryption Re-establishment.* In pairing, devices generate a temporary common security encryption key which is used to switch to a secure encrypted link. In bonding, the pairing procedure is completed, followed by the generation and exchange of permanent security keys. Bonding establishes a permanent bond, as the keys are stored in non-volatile memory. Encryption Re-establishment defines how the keys generated during the bonding process can be re-used in future connections.

Logical Link Control and Adaptation Protocol

The Logical Link Control and Adaptation Protocol (L2CAP) is responsible for transporting data for higher layer protocols, it is responsible for the fragmentation and recombination of packets, and L2CAP performs quality of service management for higher layer protocols.

3.2.5 Profiles

Profiles in BLE encapsulate basic modes of operation or specific use cases required by devices. Profiles are definitions of how a protocol should be used. The Generic Access Profile (GAP) and the Generic Attribute Profile (GATT) are the two main profiles in BLE which are fundamental to ensure interoperability between devices from different vendors.

GAP

At its core, GAP allows devices to discover one another, establish connections, and broadcast data. GAP defines four roles which are higher level versions of those specified by the Link layer, and are as follows:

- Broadcaster - periodically sends out advertising packets with data.
- Observer - periodically scans for advertising packets.
- Central - initiates connections with peer devices (link layer Master).
- Peripheral - establishes connections with centrals (link layer slave).

A set of modes are also defined by GAP which are a further refinement of roles and relate to device discoverability and connectibility.

GATT

GATT is the top most layer in the BLE stack and deals with data exchanges between devices. It is built on top of the ATT and defines a basic data model and functions that allow devices to discover, read, write, and push data between one another. The Bluetooth SIG has developed a wide range of GATT profiles including:

- Cycling Speed and Cadence Profile - transfer speed and cadence data from bicycle sensors to a phone.
- Glucose Profile - transfer glucose levels over BLE.
- Health Thermometer Profile - transfer body temperature readings over BLE

GATT is not required for an AODV implementation as all necessary data can be transmitted in packets, and host data is irrelevant to other nodes.

3.2.6 Intervals and Windows

When a device has the broadcaster role, advertisement packets are sent periodically on each of the three advertisement channels. The time interval between the sending of advertisement packets is known as the *advertisment interval*. This interval is comprised of a fixed interval and a random delay.

The fixed interval can be set from 20ms to 10.24s, in steps of 0.625ms. The random delay is a value from 0ms to 10ms that is automatically added by the link layer. The purpose of this random delay is to reduce collisions between the advertisements of different devices.

When a device has the observer role, the device scans each of the three channels periodically. The time interval between the scanning of each channel is known as the *scan interval* and the time spent scanning each channel is called the *scan window*.

Radio duty cycling is accomplished by reducing the scan interval and scan window. In the context of attendance tracking, the window is always set to the minimum possible value, while the interval can be set to the maximum possible value. This means that the radio duty cycling preamble has to be at least as long as the maximum scan interval.

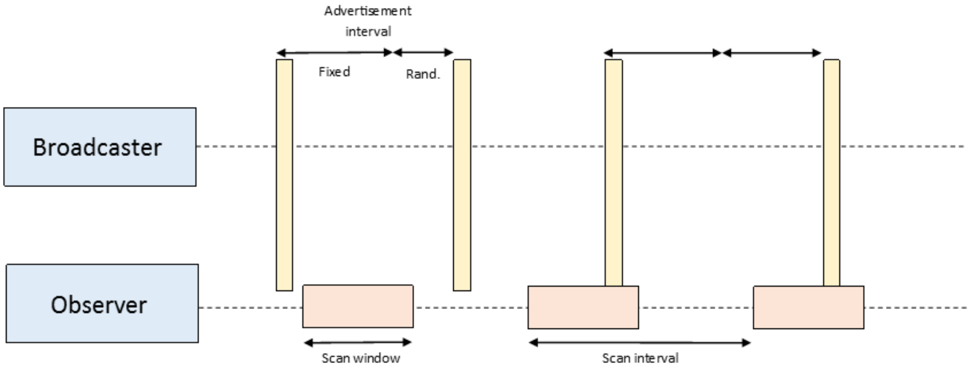


Figure 3.2: BLE Broadcaster and Observer

Figure 3.2 shows two devices with the broadcaster and observer roles. It illustrates the importance of interval and window selection, as an observer's scan window must overlap with a broadcaster's advertisement interval in order to receive the advertisement data. However, the shorter the interval and longer the window, the more power is consumed by the radio.

The designs featured in this section make use of devices with both the broadcaster and observer roles. This is to avoid the overhead of establishing connections which are unnecessary if a very low number of packet exchanges are sufficient, and device location is not guaranteed with each use of the application.

3.2.7 Connections

Connections are a sequence of data exchanges between central and peripheral devices at predefined times. Connections are established by the central device scanning neighbouring nodes and identifying if any are currently accepting connection requests. If a suitable peripheral is identified, the central sends a connection request packet which contains the following:

- Frequency hop count - determines the hopping sequence that will be followed by both devices throughout the connection period.
- Connection interval - the time between two consecutive connection events.
- Slave latency - the number of connection events a peripheral can ignore without causing a disconnection.
- Connection supervision timeout - the maximum time between two received data packets before the connection is considered lost.

Devices can perform a pairing or bonding procedure at the beginning of connections using the SMP.

No connections are established in the attendance tracking protocols outlined in this section as the AODV control packets fit within a single advertisement packet, and continuous and frequent communications on the level of BLE connections is not required, making it an unnecessary overhead.

3.2.8 BLE Data Packets

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 which is illustrated in figure 3.3. The header includes the actual length of the packet and the PDU type. The payload length depends on the advertising PDU type. For beacon-like devices, and for attendance tracking, the PDU type used is the *ADV_NONCONN_IND* type. This type indicates a non-connectable advertisement, which means central devices will not attempt to establish connections with it.

The PDU payload includes a 6 byte Bluetooth MAC address, leaving 31 bytes for advertisement data structures. These advertisement structures each contain a type, length, and data section. The types of these structures are pre-defined and the most commonly used include service UUIDs, shortened and complete local names, and manufacturer specific data.

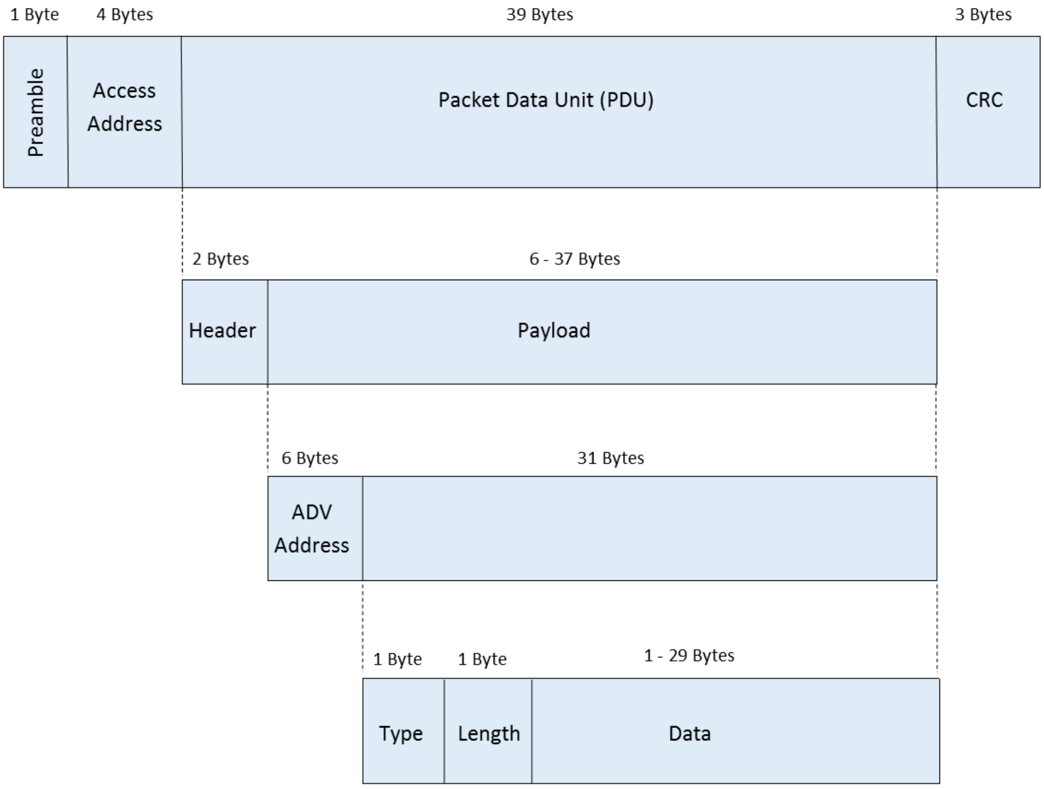


Figure 3.3: BLE Advertisment Packet Structure

3.3 AODV BLE module

The attendance tracking applications are based on AODV's route discovery mechanism and so the main adaptations required for a Bluetooth Low Energy AODV module are discussed.

3.3.1 Advertisement Periods

Unlike the multicast described in AODV's specification [8], advertising in BLE is not instantaneous. As mentioned previously, a scanner's scan window must overlap with an advertiser's advertisement interval in order to receive data packets. Unlike connections, with broadcasting these intervals and windows are not synchronised. If a device wishes to broadcast a data packet, it is required to advertise that specific packet for a certain period. By selecting an appropriate period, the advertising device can be confident that the scanning windows of neighbouring devices will overlap with their advertisement interval within the specified period.

3.3.2 Packet Buffering

The receiving of an AODV control packet may require the receiver to broadcast a packet in response. As a result of the advertisement period discussed in section 3.3.1, packets which a device needs to send must be buffered. Without buffering, any currently advertising packets could go unreceived by not being broadcast for a sufficient amount of time.

3.4 Attendance Tracking

In the following sections the root node is the device tracking the attendance of user nodes and the root node's address is known globally.

3.4.1 Roll Call

The *Roll Call* protocol is designed for lecture theatre-type scenarios. That is, a list of users is in our possession and we wish to determine which of these are present and which are not. An illustration of the protocol is provided in 3.4.

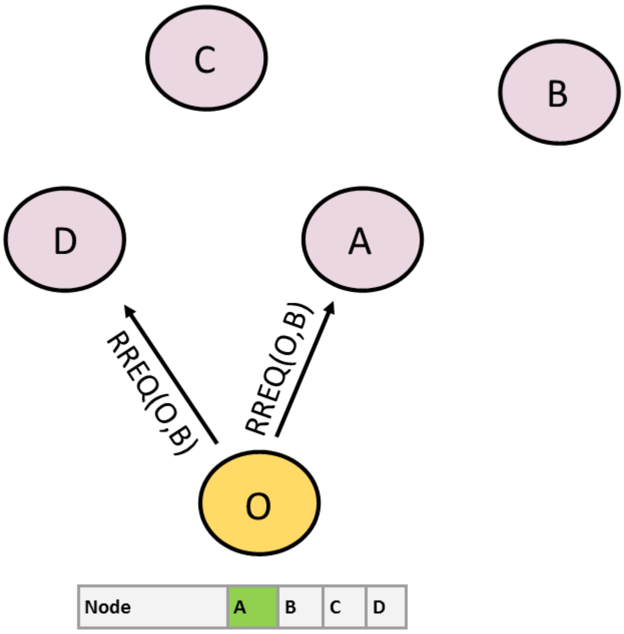


Figure 3.4: Roll Call

To identify if a user is present, the root node broadcasts a RREQ into the network with the destination address set to that of the user. When a RREQ arrives in a node, if the node is not the destination, it re-broadcasts the RREQ according to AODV's specification. When a RREQ arrives at the destination node, they send a RREP back to the root node indicating their presence. If a RREP is not received before the end of the protocol, the user is counted as absent. The duration of the protocol is dependent on the size of the user list.

In this protocol Low Power Listening radio duty cycling is performed. User nodes periodically scan to see if roll call is occurring. If they detect that it is occurring, they enter a high power state. The first RREQ sent by the root node has an advertisement period sufficiently long to ensure nearby nodes will detect it. This is similar to the method used by Box-MAC-2 in section 2.2.4. The first RREQ is the preamble stream containing all the required data and is re-broadcast by receiving nodes.

3.4.2 Route to Zero

The *Route-to-Zero* protocol is designed for scenarios in which the attendance of an otherwise unknown device can be taken. That is, a list of users is built during the protocol. An illustration of the protocol is provided in 3.5.

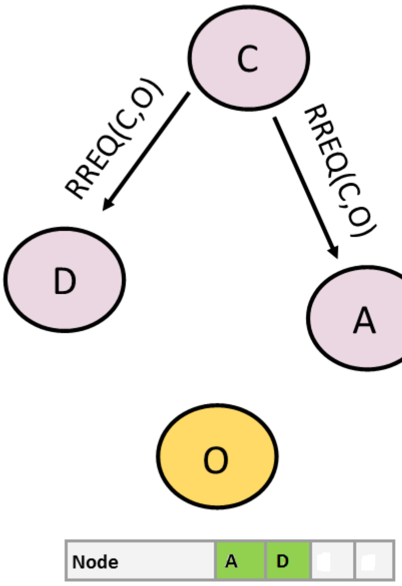


Figure 3.5: Route to Zero

In this protocol, the responsibility of tracking attendance is placed on the users. User nodes attempt to discover a route to the root node by broadcasting a RREQ with the destination address set to that of the root node. When the RREQ arrives at the root, it adds the user to its list and sends a RREP back to the user node to acknowledge its presence.

In this protocol, once again, Low Power Listening radio duty cycling is performed. User nodes periodically scan to see if Route-to-Zero is occurring. If they detect that it is occurring, they enter a high power state. The preamble in this scenario is a RREQ sent by the root with the destination and source address both equal to the root node's address. This RREQ is re-broadcast by all nodes that receive it. Upon receiving the RREQ, the user node enters a high power state and begins the protocol, identifying a route to the root node.

3.4.3 Design Discussion

There may exist the opportunity to use both the Roll Call and Route-to-Zero protocols in conjunction with one another. This might occur if the root node has a pre-determined list of users, additional users (not in the pre-determined list) can be in attendance, and users that might not all arrive at the same time.

The route-to-zero protocol can be used initially, generating a list of present users. This list can be compared with the pre-determined list and if a user is missing, the Roll Call protocol can be initialised at a later point to verify the absence of the missing users.

3.5 Summary

When designing applications for BLE, careful consideration has to be taken with regards to the GAP communication types, whether broadcasting is sufficient or if connections are required. When broadcasting, the advertising intervals and scanning intervals and windows have to be selected with care. This is so the chance of a scan window overlapping with the advertisement interval is sufficiently high and so that too much power is not consumed. BLE advertisement packets have to be formed such that they adhere to the GAP packet specification, and it is important to choose advertisement data carefully as packets are limited in size.

In order to translate AODV into a BLE module, some design decisions were required. The first of which is Advertisement Periods, to ensure that an advertisement packet is being broadcast for a sufficiently long time so that we can be confident that it will be received by neighbouring nodes. The second of which is Packet Buffering, a consequence of advertisement periods.

Two protocols are proposed for attendance tracking, both of which are based on AODV's route discovery mechanism. The Roll Call protocol is designed for scenarios in which the attendance of a pre-determined group of users is being taken. The Route-to-zero protocol is designed for scenarios in which there is not a pre-determined group of users. These protocols can be used in conjunction if the scenario allows and requires it.

Chapter 4

Implementation

4.1 Introduction

This chapter provides a brief overview of the hardware used, an overview of the implementation architecture as well as descriptions of the Broadcaster and Observer GAP role implementations. A brief discussion is provided on the necessity of route table entry timers and route maintenance in AODV. Finally, a discussion of security considerations for the implementation is provided.

4.2 Hardware

The hardware used for the implementation of the attendance tracking application is Nordic Semiconductor's nRF51 and nRF52 development kit. This kit is used for Bluetooth Low Energy, ANT, and 2.4GHz applications.

The kit provides access to all I/O and interfaces and has four programmable LEDs and push buttons. It also has a Segger J-Link interface commonly used for debugging.

4.3 Architecture

4.3.1 Overview

The attendance tracking application was implemented using the Nordic SDK [22]. The Nordic SDK exposes all the functionality of the BLE stack, such as functions to start and stop advertising and scanning, in addition to various device modules such as software timers. Custom scanning and advertising files were created to encapsulate the available functionality in the SDK, mainly to initialise parameters, start, stop, and in the case of advertising, to update the advertisement packet currently being broadcast.

An AODV module was written which sits on top of the Nordic SDK. The AODV module contains packet type definitions and functions to send and process control packets. These functions adhere to the relevant areas of the AODV specification [8]. A simplified version of the architecture is illustrated in 4.1.

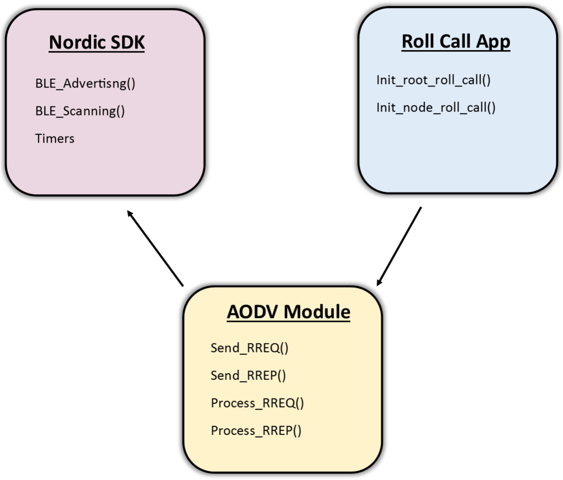


Figure 4.1: Implementaion Architecture

The AODV module contains a file which encapsulates packet management and functionality. This includes high level structures for RREQ and RREP packets, functions to convert these high level structures to and from byte arrays for the BLE stack, utility functions to generate packets, copy packets, access individual packet bytes, and finally, functions for improved debugging.

The module contains a file which encapsulates routing table management and functionality. This includes a structure for routing table entries, and functions for routing table lookups and removals.

The module contains a main file (aodv_module.h) which encapsulates the core functionality of AODV. This includes module initialisation and reset functions, functions to send and process RREQs and RREPs, a function to process advertisement reports specific to AODV, timer handlers and a number of global static member data such as a list of routing table entries as the routing table, a list of previously received packets, and a packet buffer made from a first-in-first-out queue.

4.3.2 Broadcasting

Figure 4.2 illustrates the flow of execution for the broadcaster GAP role implemented in the AODV module.

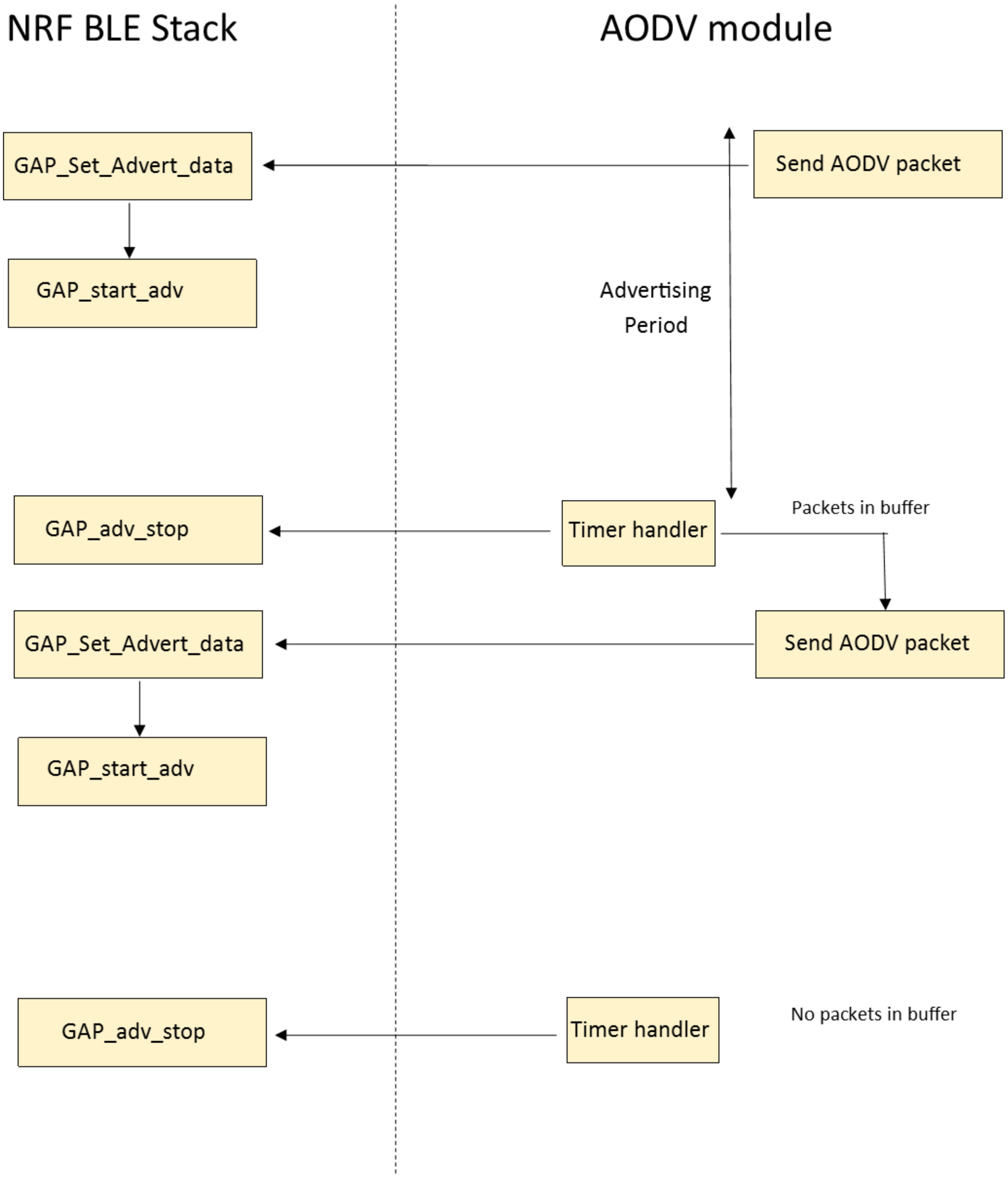


Figure 4.2: Broadcasting Implementation

As seen in figure 4.2, advertising is initialised and started with the sending of the first RREQ packet. Each time a send function is invoked, the packet is placed within the packet buffer. When the AODV module is initialised, a recurring timer is started which represents the advertising period. Each time the timer expires, its handler turns advertisements off, and takes the next packet to be sent from the packet buffer. If the buffer is empty, the handler exits, leaving advertisements turned off. If there is a packet in the buffer, the handler passes it to the appropriate send function (RREQ or RREP).

4.3.3 Observing

Figure 4.3 illustrates the flow of execution for the observer GAP role implemented in the AODV module.

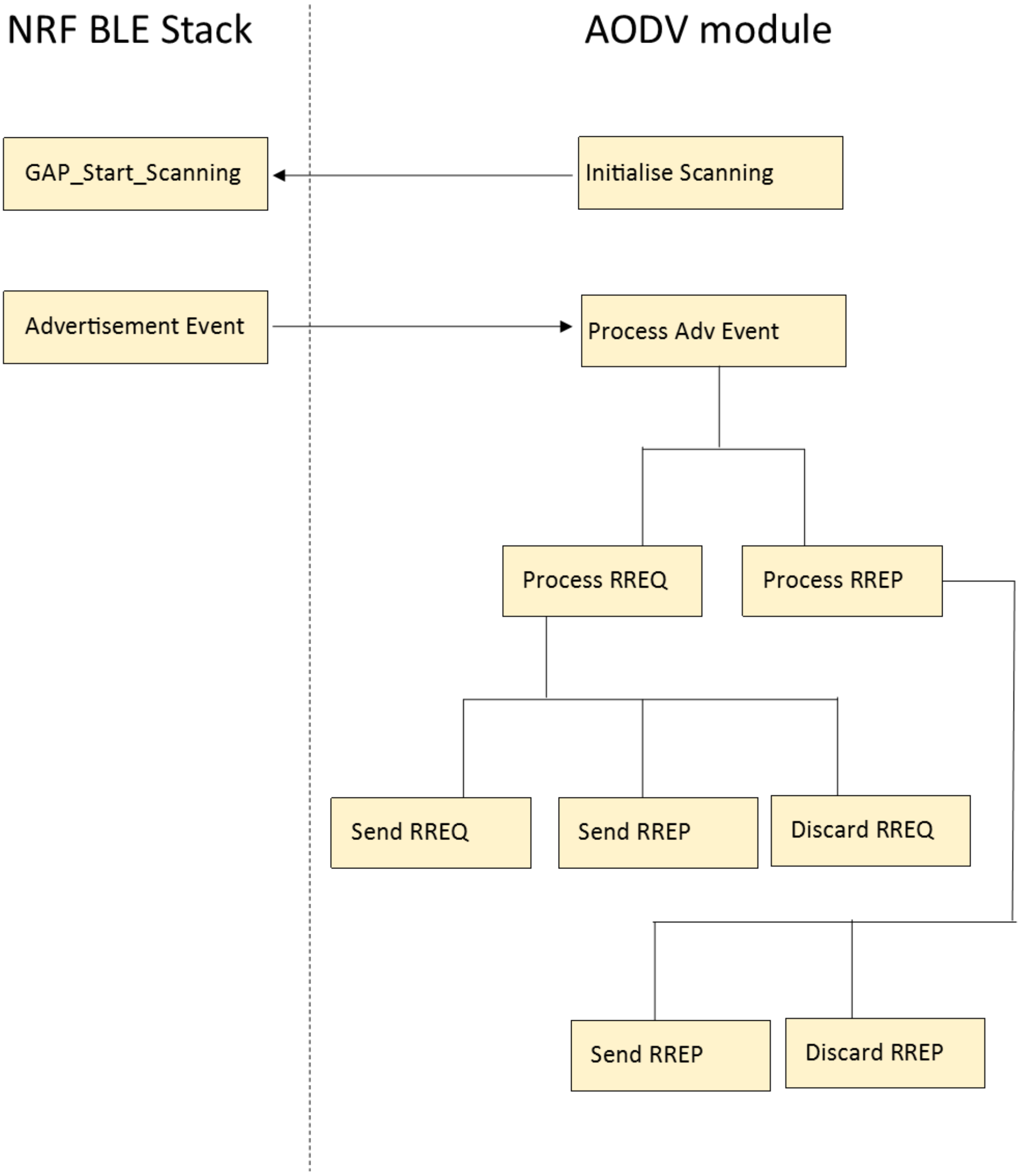


Figure 4.3: Observer Implementation

As seen in figure 4.3, scanning is initialised and started during the initialisation of the AODV module. All devices scan throughout the lifetime of the protocol. The BLE stack generates an advertisement event when an advertisement packet is scanned. This generated event contains the actual packet and all advertisement events are processed by the AODV module. the packet is sent to a processing function based on the type of advertisement (RREQ/RREP).

When processing RREQs, to prevent unnecessary processing, if the source address is equal to that of the node, the packet is discarded. This is required as the node broadcasting the RREQ may scan the same RREQ a number of times, possibly more than the number of surrounding nodes re-broadcasting it as they repeatedly advertise the duration of the advertisement period.

If the node processing the RREQ is not the destination, and does not contain a route to the destination in their routing table, they buffer the RREQ for re-broadcasting.

If the node processing the RREQ is the destination or has a route to the destination in their routing table, they generate a RREP and send it to the source of the RREQ.

4.3.4 Routing table

Entries in an AODV routing table have a time-to-live associated with them. This is to reduce routing overhead by discarding routes that are not actively being used.

This functionality has not been implemented in the AODV module. The reason for this is that the routing table is reset at the start of each period of attendance tracking. This is Because nodes are not guaranteed to be in the same location with the same neighbours each time the protocol is active, which would result in stale routing tables. Another reason is that the implementation was originally targeted to be tested in networks with a maximum of 16 nodes so any reduction on routing table overhead would be minimal.

4.3.5 Route Maintenance

AODV's route maintenance is performed at each node in the network by periodically broadcasting HELLO messages. If a node fails to receive a HELLO message from another node in one of its routing table entries, the node broadcasts a route error message to its neighbours allowing any nodes with an entry to the broken link to mark it as invalid.

Route maintenance is not required when only the route discovery mechanism of the protocol is used, and the target networks are semi-static. Not having route maintenance still allows for small amounts of mobility, as long as nodes stay within range of each other.

4.4 Security Considerations

This section discusses some of the security issues with the described implementation and possible methods of securing the protocols.

Passive Eavesdropping

In BLE, when the two devices are initially paired, they derive a long-term key using a key-exchange protocol. All communication from that point on is encrypted with the derived key. Both AODV based protocols use BLE advertising exclusively, no device bonding occurs. Without bonding, it is not possible to achieve secure advertising using the BLE stack. It is also worth noting that it is important that all nodes in a network can read all packets to update their own routing table entries and to properly route packets to the correct destinations. Full packet encryption could be used to provide confidentiality and to combat eavesdropping. A shared secret key may be programmed into devices which can then be used to perform cryptographic operations. This is suitable when application distribution is controlled i.e. organisation ID cards with a built in microcontroller.

Identity Tracking

If a BLE device broadcasts a unique id such as its device address, that address can be associated with the device owner/user. This enables the physical tracking of the user based on the presence of the BLE device. The purpose of the attendance tracking application is to associate devices and users during a gathering (lecture/conference/fire evacuation), but malicious nodes eavesdropping during these gatherings could then perform identity tracking at any time.

With BLE, the device address can be manually changed. Changing the device address at a specific interval can prevent the association of an address and a user. In the context of an attendance tracking application, if a root node attempts to identify which devices from a list of device addresses are present, a device with a changed address would not be considered. If a device changes its address, the root nodes address list needs to be updated. Separate unique identifiers should be used in addition to device addresses. These identifiers can then be encrypted and the device addresses can be changed at regular intervals. This way, only those capable of decryption can associate an ID from a device with a user.

Impersonation Attacks

AODV is susceptible to nodes performing malicious activities while masquerading as other nodes.

AODV specifications [8] security considerations section advises the following in regards to impersonation attacks:

- Route Response (RREP) packets SHOULD be authenticated to prevent spurious routes to a desired destination. This prevents attackers masquerading as a desired destination and denying service to the destination or inspecting all traffic intended for the destination.

- Route Error (RERR) packets SHOULD be authenticated to prevent malicious nodes from disrupting valid routes.

- Keyed-hash Message Authentication Codes (HMAC) MAY be used to authenticate messages given a shared secret key. One issue with hashes in the context of BLE advertising is the limited packet size. A BLE advertising packet can contain 31 bytes of data. AODV RREP packets require 20 bytes of data, leaving room for an 11 Byte MAC.

Active Attacks in AODV

AODV control packets contain hop counts and sequence numbers which are used by nodes to identify the freshness of a route. The mutability of these values is a vulnerability, in that nodes can modify these values to maliciously advertise better routes.

By deceptively increasing the sequence number in a packet, a fresher route to a particular destination can be advertised. By deceptively decreasing the hop count, inefficient routes will be used, increasing network resource usage. Authentication using HMACS, as described above, can be used to prevent these attacks.

Chapter 5

Results and Analysis

5.1 Introduction

This chapter provides an overview of the methodology used in the evaluation of the implementation of the Roll Call [1] and Route-to-Zero protocols. In addition to this, the results from a medium scale evaluation are shown and discussed.

5.2 Evaluation Methodology

The network topology used for the evaluation is a three by four grid with a root and nine user nodes as illustrated in figure 5.1. With this topology, the implementation can be tested with a hop count of three. A single hop is the maximum distance a node can broadcast data. From experiments with the nRF51 we discovered a hop distance of 4.32m when using a transmission power of -30 dBm (minimum transmission power of the nRF51).

For this experiment the root node sends the RREQs alphabetically, beginning with node A and ending with node I.

[1] a short video demonstration of the Roll Call implementation is available at [23]

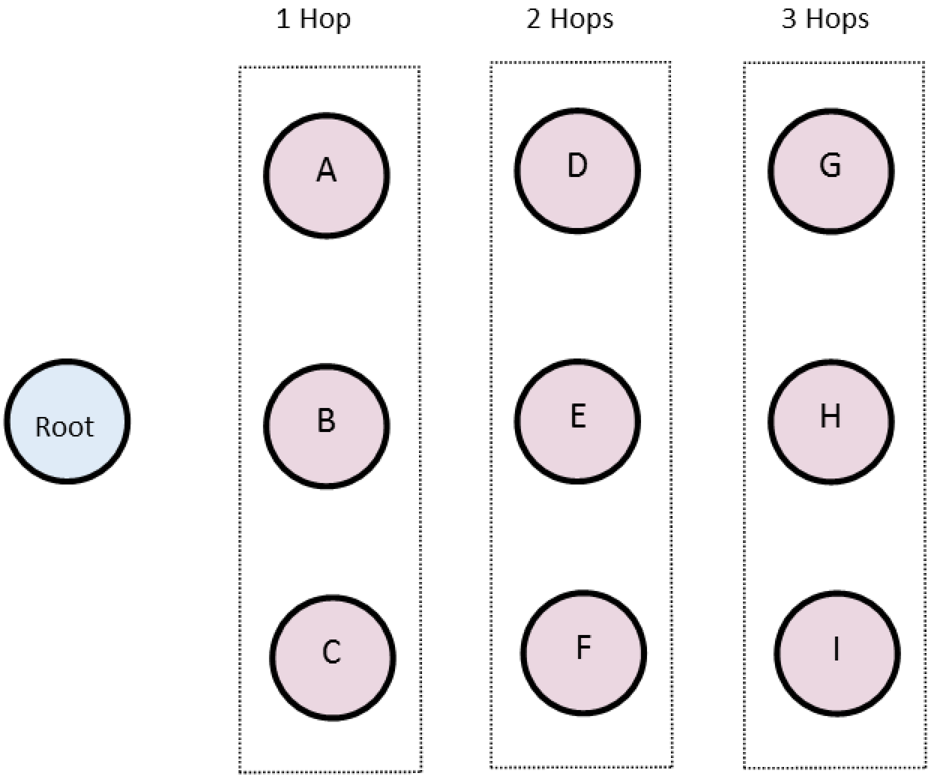


Figure 5.1: Evaluation network topology

The metrics measured include:

- Node discovery rate. This is the rate at which nodes are discovered, measured from the beginning of the protocol until the final node has been identified. For the purpose of this experiment, the Route-to-Zero implementation knew the total number of nodes in the network.

- Total packets sent/processed per node. This is the total number of RREQ and RREP packets that are sent and processed by each node. Repeat packets are not processed.

- Average packets sent/processed per hop. This is the average number of packets sent and processed for each hop away from the root that a node is. Nodes A, B, and C are one hop away, nodes D, E, and F are two hops away, and nodes G, H, and I are three hops away.

These metrics were recorded using a statistics packet. This packet is sent to the root node in the same manner as a RREP packet. This packet contains the number of RREQs and RREPS that were received, sent, and processed by the node (repeat packets are not counted towards processed packet count). For the purpose of this experiment, all statistic packets were sent twenty seconds after the receipt of the first RREQ.

5.3 Medium Scale Evaluation Results

5.3.1 Evaluation Settings

The BLE settings used in this experiment are as follows:

- Advertisement Interval: 100ms
- Advertisement Period: 500ms
- Scan Interval: 50ms
- Scan Window: 20ms

5.3.2 Node discovery rate

The graph in figure 5.2 shows the number of nodes that were discovered with respect to time.

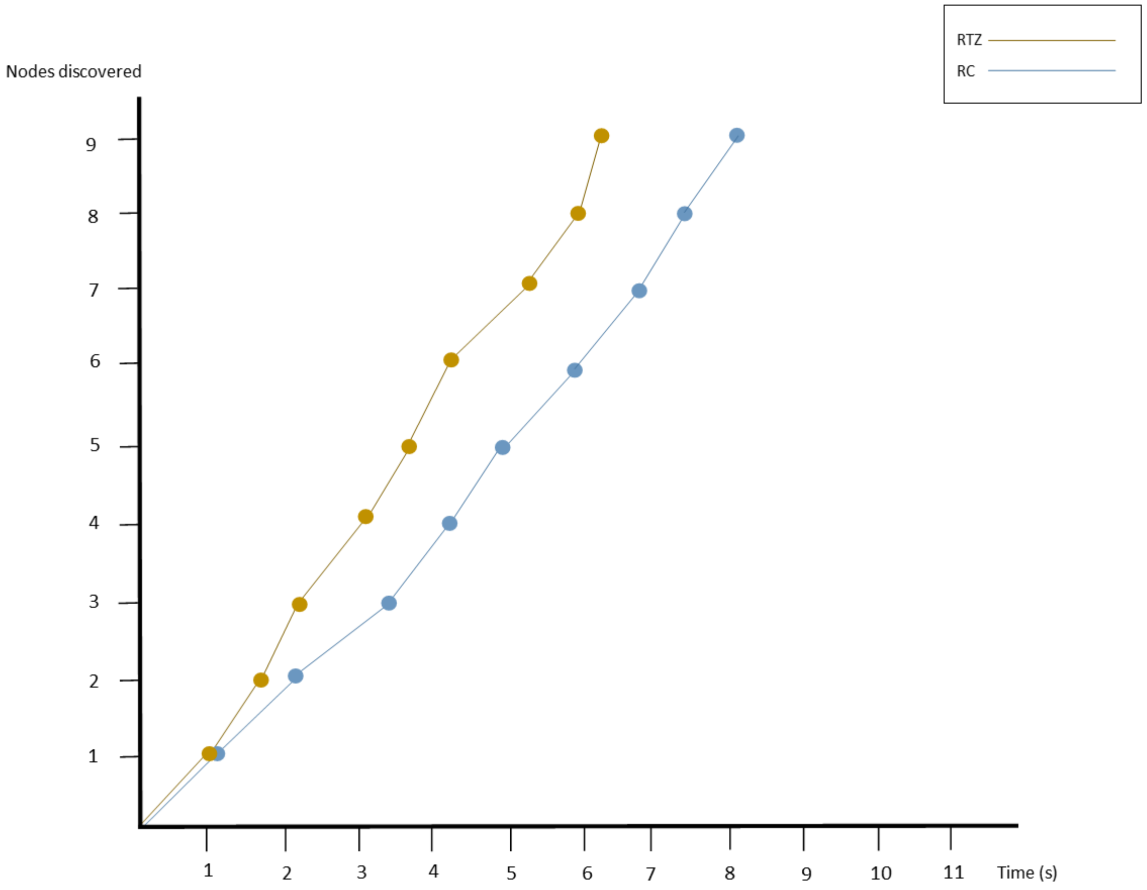


Figure 5.2: Number of nodes discovered vs time

Roll Call (RC) has a discovery rate of 1.06 nodes per second while Rout-to-Zero (RTZ) has a discovery rate of 1.34 nodes per second. The reason why RTZ has an improved discovery rate is that only a single RREQ is flooded through the network initially, meaning that nodes in the first hop can each send their own RREP after forwarding the RREQ, and as the wake-up RREQ arrives at subsequent hops, these nodes will have already performed a large chunk of the RREQ forwarding from other nodes and will we able to send their RREPs sooner than those in RC. In RC, the last RREQ from the root is only sent after 4.5 seconds (500ms per RREQ and 9 nodes), so when a RREQ arrives at its destination, the node will have its buffer filled with other messages to forward and so won't be able to respond with its own RREP as quickly. One important thing to note in this scenario is that when all nodes are discovered in

RC no more packets will be broadcast within the network, whereas in RTZ, there may be RREP messages from the root still propagating through the network.

5.3.3 Packets per node

The graph in figure 5.3 shows the number of packets sent and processed at each node in the network.

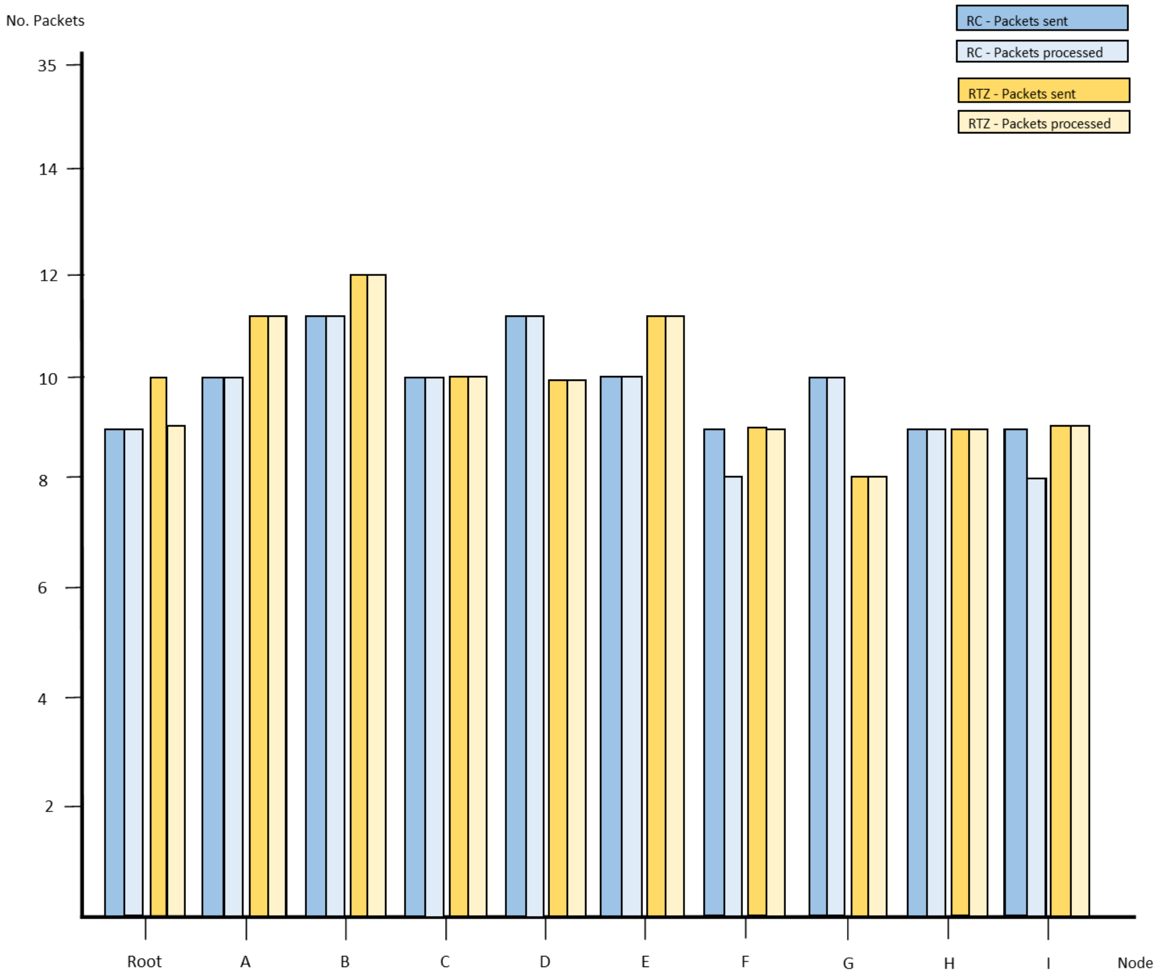


Figure 5.3: Total packets sent/received per node

The number of packets sent and processed at each node is very similar in both protocols and the number of sent packets is largely the same as the number of processed packets at each node. The reason for this is that if a RREQ is processed, it is either forwarded or a RREP is returned. If a RREP is processed, it will be forwarded as RREPs are only processed by nodes in the RREP route.

The network will be flooded with repeat packets and broadcasted RREPs due to the nature of BLE advertising, but these packets will not have much impact on nodes as they are instantly discarded if repeated or not relevant.

5.3.4 Packets per hop

The graph in figure 5.4 shows the average number of packets sent and received for nodes at different hop counts from the root.

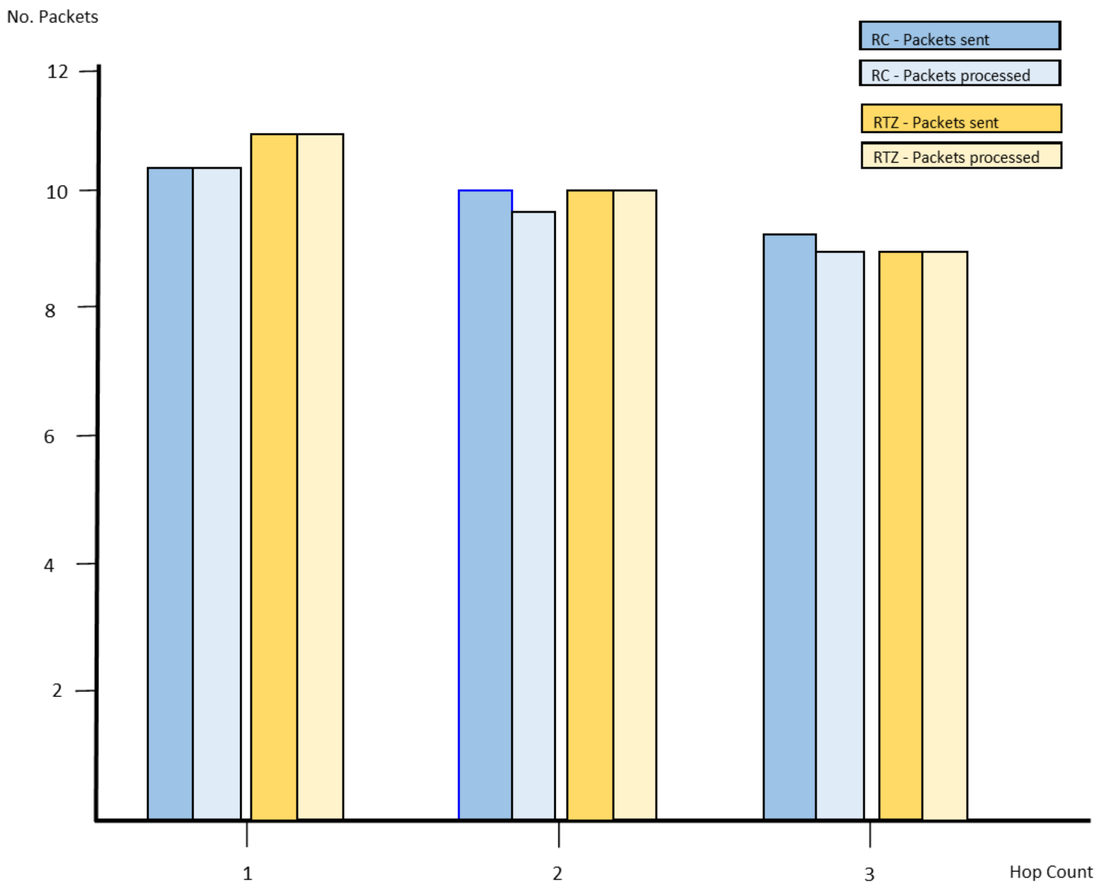


Figure 5.4: Average packets sent/received vs hops from root

The number of packets sent and received increases marginally the closer a node is to the root node. The reason for this is that nodes three hops away will not generally have any RREPs broadcast to them, so any that are scanned will likely be discarded and as a result, throughout the lifetime of the protocol the node will not have to forward any RREP packets. With an increased number of hops, and devices further apart, I expect this would result in more noticeable differences between hops.

A difference in packet processing implies that nodes closer to the root consume more power than farther nodes as they are advertising for longer periods (if a node has no packets in its buffer advertising is turned off). This would only come into effect in networks with many more nodes than are present in this network. Nodes 1 hop from the root only advertise for ˜1s longer than those that are 3 hops away.

Chapter 6

Conclusion

6.1 Overview

In this thesis I have provided the design for two protocols that make use of AODV's route discovery mechanism to perform attendance tracking. The first protocol *Roll Call*, is designed for lecture theatre-like scenarios in which the attendance of a predetermined list of users is being taken. The second protocol *Route to zero*, is designed for scenarios in which the attendance of a group of possibly unknown users is being taken.

An AODV module with route discovery functionality was implemented for use in Bluetooth Low Energy. In addition to this, the Roll Call and Route-to-Zero protocols were implemented using the AODV module and tested in a small to medium scale network of nRF51 devices. The results of the evaluation show the protocol working with three hops in the network, but further fine tuning of the BLE setting is required to improve the performance of the protocol.

6.2 Discussion

The two designs were capable of performing node discovery, but it is difficult to say how successful they were without any other references. Based on the recorded data, it would take the RTZ protocol 90 seconds to take the attendance in a lecture theatre of 120 students. This would be further increased in networks with mobility and in areas with high interference - an aspect of networks untested in this thesis.

The AODV discovery mechanism was suitable for the task but in both designs, the amount of RREQ packets flooded through the system was quite high. It is possible that the metrics measured could be further improved by reducing the advertisement period and increasing the scanning interval, allowing for packets to be flooded through the network at a faster pace. It would also be necessary to experiment in larger networks - either with physical devices, or through simulation.

Overall, AODV seems to have been a good choice. It outperformed the other investigated protocols in mobile networks, which could be desirable in attendance tracking applications, but was not tested in this thesis. For an approach focused on purely static networks, RPL or OLSR, would likely be better choices, but have a higher implementation complexity and their performance in BLE is difficult to predict.

6.3 Future Work

6.3.1 Improvements to AODV

There are a number of improvements that could be made to the AODV module, namely the implementation of route maintenance. This would allow for attendance tracking in scenarios of high mobility and would be necessary in larger networks were there is a much higher chance of route failure.

6.3.2 Simulation

The testing of the designs outlined in this thesis was only performed in a small ten node network. It is difficult to test in larger physical networks as the devices have to be acquired, programmed, and space has to be located. With a minimum range of 4.32m, a very large space would be required to test networks with hop counts in the double digits and higher. One solution to this which is commonly used, is a simulation environment that replicates these larger networks which provides the advantage of not requiring physical devices or space.

One issue with this is that there are few BLE simulation tools. There is a BLE simulator [24] implemented in the OMNeT++ simulator [25] but there are no instructions, no updates, and it has quite poor code quality as it was initially intended to quickly check some of the author's ideas. As a result of this, in the early stages of this thesis, I decided to implement a higher level BLE simulator, which accurately simulated BLE channels in the physical layer, and advertising and scanning. The implementation was written in Java and drew from the OMNeT++ simulator. Unfortunately, there wasn't sufficient time to implement this as well as implementing the designs on physical hardware and so simulation development was halted to prioritise the physical implementation.

6.3.3 Implementation of other protocols

It would be interesting to implement some of the other protocols discussed in chapter 2, especially RPL. RPL outperformed AODV in section 2.4 so it would be interesting to see its performance when used for attendance tracking. The adaptation of RPL for BLE would be difficult as it has a very high implementation complexity but perhaps performance gains would outweigh this complexity.

Bibliography

[1] IHS Markit, "Iot trend watch 2017," jan 2017.

[2] I. Skerrett, "Iot developer survey 2017," apr 2017.

[3] D. Moss, J. Hui, and K. Klues, "Low power listening," *TinyOS Core Working Group*, vol. TEP 105, 2007.

[4] R. Musaloiu-E., C.-J. M. Liang, and A. Terzis, "Koala: Ultra-low power data retrieval in wireless sensor networks," *International Conference on Information Processing in Sensor Networks*, pp. 421–432, 2008.

[5] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks," in *Proceedings of the 4th international conference on Embedded networked sensor systems*, pp. 307–320, Association for Computing Machinery, 2006.

[6] D. Moss and P. Levis, "Box-macs: Exploiting physical and link layer boundaries in low-power networking," *Computer Systems Laboratory Stanford University*, 2008.

[7] A. Dunkels, "The contikimac radio duty cycling protocol," *SICS Technical Report*, dec 2011.

[8] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (aodv) routing," RFC 3561, RFC Editor, July 2003. `http://www.rfc-editor.org/rfc/rfc3561.txt`.

[9] D. Johnson, Y. Hu, and D. Maltz, "The dynamic source routing protocol (dsr) for mobile ad hoc networks for ipv4," RFC 4728, RFC Editor, February 2007. `http://www.rfc-editor.org/rfc/rfc4728.txt`.

[10] T. Clausen and P. Jacquet, "Optimized link state routing protocol (olsr)," RFC 3626, RFC Editor, October 2003. http://www.rfc-editor.org/rfc/rfc3626.txt.

[11] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers," *ACM SIGCOMM Computer Communication Review*, vol. 24, pp. 234–244, oct 1994.

[12] R. Bellman, "On a routing problem," *Quarterly of applied mathematics*, vol. 16.1, pp. 87–90, 1958.

[13] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and R. Alexander, "Rpl: Ipv6 routing protocol for low-power and lossy networks," RFC 6550, RFC Editor, March 2012. http://www.rfc-editor.org/rfc/rfc6550.txt.

[14] M. Bouhorma, H. Bentaouit, and A. Boudhir, "Performance comparison of ad-hoc routing protocols aodv and dsr," in *International Conference on Multimedia Computing and Systems*, vol. 2 of *2*, pp. 511 – 514, apr 2009.

[15] S. K. Gupta and R. K. Saket, "Performance metric comparison of aodv and dsdv routing protocols in manets using ns-2," in *International Journal of Research and Reviews in Applied Sciences*, vol. 7.3, pp. 339–350, jun 2011.

[16] M. Vucinic, B. Tourancheau, and A. Duda, "Performance comparison of the rpl and loadng routing protocols in a home automation scenario," in *IEEE Wireless Communications and Networking Conference (WCNC): NETWORKS*, IEEE, apr 2013.

[17] A. Brandt and J. Buron, "Home automation routing requirements in low power and lossy networks," Internet-Draft draft-ietf-roll-home-routing-reqs-11, IETF Secretariat, January 2010. http://www.ietf.org/internet-drafts/draft-ietf-roll-home-routing-reqs-11.txt.

[18] A. Tuteja, R. Gujral, and S. Thalia, "Comparative performance analysis of dsdv, aodv and dsr routing protocols in manet using ns2," in *International Conference on Advances in Computer Engineering*, IEEE, jun 2010.

[19] P. G. Lye and J. C. McEachen, “A comparison of optimized link state routing with traditional ad-hoc routing protocols,” in *Operations Research, Systems Engineering and Industrial Engineering Commons*, US Department of Defense, dec 2006.

[20] “The bluetooth special interest group.” https://www.bluetooth.com.

[21] “Bluetooth core specifications.” https://www.bluetooth.com/specifications/adopted-specifications.

[22] “nrf5 sdk.” https://www.nordicsemi.com/eng/Products/Bluetooth-low-energy/nRF5-SDK.

[23] “Roll call demonstration.” https://www.youtube.com/watch?v=Kjj2l9KPVzk.

[24] K. Mikhaylov, “Bluetooth low energy (bluetooth smart) network simulation tool.” http://cc.oulu.fi/~kmikhayl/BLE.html.

[25] “Omnet++ dsicrete even simulator.” https://omnetpp.org/.