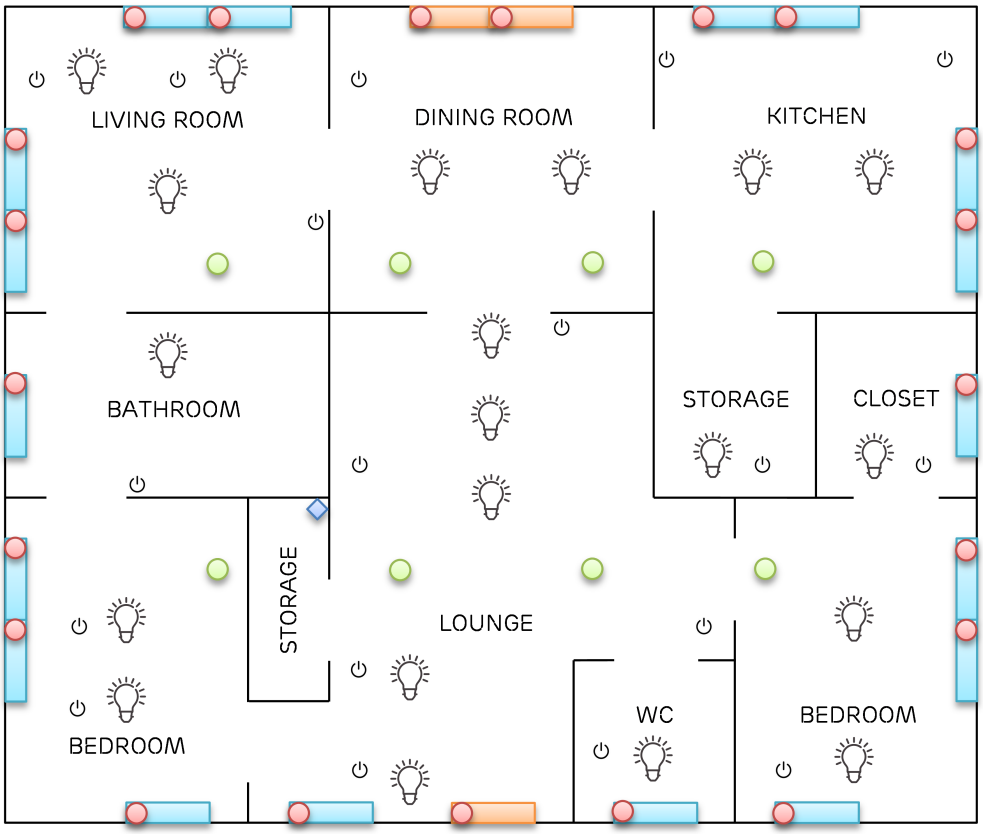
CHALMERS
UNIVERSITY OF TECHNOLOGY



Evaluating IPv6 Connectivity for Bluetooth Low Energy and IEEE 802.15.4

A Study on Radio Standards for the Internet of Things

Master's thesis in Communication Engineering

PATRIK TRELSMO

Department of Signals and Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2016

Master's thesis 2016:EX071

Evaluating IPv6 Connectivity for Bluetooth Low Energy and IEEE 802.15.4

A Study on Radio Standards for the Internet of Things

PATRIK TRELSMO

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Signals and Systems
Chalmers University of Technology
Gothenburg, Sweden 2016

Evaluating IPv6 Connectivity for Bluetooth Low Energy and IEEE 802.15.4
A Study on Radio Standards for the Internet of Things
PATRIK TRELSMO

Examiner: Fredrik Brännström, Department of Signals and Systems, Chalmers
Supervisor: Piergiuseppe Di Marco, Ericsson Research
Supervisor: Johan Östman, Department of Signals and Systems, Chalmers
Supervisor: Roman Chirikov, Ericsson Research

Master's Thesis 2016:EX071
Department of Signals and Systems
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: The simulated home automation scenario used as basis for the analysis of Bluetooth Low Energy and IEEE 802.15.4. The scenario is described in Appendix B.

Typeset in LaTeX
Printed by Teknologtryck
Gothenburg, Sweden 2016

Evaluating IPv6 Connectivity for Bluetooth Low Energy and IEEE 802.15.4
A Study on Radio Standards for the Internet of Things
PATRIK TRELSMO
Department of Signals and Systems
Chalmers University of Technology

Abstract

IPv6 is seen as a key building block for the Internet of Things (IoT). It enables interoperability between networks based on different physical layers, introduces a vast address space and has methods that automatically configure device addresses. However, there are issues with running IPv6 over low power radio technologies such as those used for IoT. The IPv6 protocol has large overheads, with a header size of 40 bytes. There is also a communication overhead to establish and maintain connections. Finally, the protocols running on top of IPv6 might not be optimised for low-power applications, which may add overhead.

This work is focused on two relevant low power radio technologies for IoT - Bluetooth Low Energy (BLE) and IEEE 802.15.4 (802.15.4). The technologies are benchmarked with respect to energy efficiency, latency and service ratio under IPv6 traffic by means of simulations in Ericsson's system simulator. It is found that the latency under IPv6 traffic is similar, but BLE outperforms 802.15.4 in energy efficiency and service ratio. The header compression introduced in IPv6 over Low power Wireless Personal Area Networks (6LoWPAN) and IPv6 over Bluetooth Low Energy (6LoBTLE) increases all performance indicators measured but requires a shared context to be set up and managed. Because of this, one must consider the amount of traffic individual devices will transmit when deciding whether to use header compression or not.

IPv6 multicast addresses are also studied. When using IPv6 multicast, packets can be either broadcasted or sequentially unicasted on the MAC layer. The results show that broadcasting decreases the latency and the loss ratio. There are however issues with broadcasting which include reliability and the lack of methods to encrypt transmissions on the link layer. Careful consideration should therefore be made when choosing how to handle IPv6 multicast addresses.

Finally, the impact of IPv6 connection establishment and maintenance is studied. This is particularly relevant for BLE devices using random link layer addresses. To be able to use header compression, a BLE device must trigger IPv6 control message exchange when the link layer address is changed. Results show that this affects the latency negatively for devices with low traffic intensity. A solution for ensuring reachable end devices by implementing a classification of messages and special mode of operation is described in the thesis.

Keywords: 6LoWPAN, Bluetooth Low Energy, ICMP, IEEE 802.15.4, IPv6, Multicast, Standardisation, Performance evaluation

Acknowledgements

First of all I would like to thank my supervisor at Ericsson Research, Piergiuseppe Di Marco, for guidance during my work and for always taking the time to answer questions. I also want to thank my supervisor at Chalmers, Johan Östman, for keeping me on track and providing valuable feedback.
There are several people at Ericsson Research who have also contributed greatly to this report. Among them are Roman Chirikov, Karin Lagergren, Pontus Arvidson and Per Skillermark, who have all been very helpful in answering questions and explaining technical details.

Patrik Trelsmo
Gothenburg, August 2016

Contents

List of Figures

List of Tables

Acronyms

6LoBTLE 6LoWPAN for Bluetooth Low Energy
6LoWPAN IPv6 over Low power Wireless Personal Area Networks
802.15.4 IEEE 802.15.4

AFH Adaptive Frequency Hopping
AWGN Additive White Gaussian Noise

BER Bit Error Rate
BLE Bluetooth Low Energy

CSMA Carrier Sense Multiple Access
CSMA-CA Carrier Sense Multiple Access with Collision Avoidance

DAD Duplicate Address Detection
DSSS Direct Sequence Spread Spectrum
DTLS Datagram Transport Layer Security

GATT Generic Attribute Protocol
GFSK Gaussian Frequency Shift Keying
GTS Guaranteed Time Slot

ICMP Internet Control Message Protocol
IETF Internet Engineering Task Force
IoT Internet of Things
IP Internet Protocol
IPv6 Internet Protocol version 6
ISI Inter-Symbol Interference

LoWPAN Low Power Wireless Personal Area Network

MIC Message Integrity Check
MTU Maximum Transmission Unit

PAN Personal Area Network
PDU Protocol Data Unit

RFC Request For Comments
RPA Resolvable Private Address

SDU Service Data Unit

UDP User Datagram Protocol
UI User Interface

1
Introduction

1.1 Background

The Internet of Things (IoT) is one of the most widely anticipated paradigm shifts today, where everyday items such as thermostats, lamps, household appliances and sensor networks will be connected to the internet. This connectivity will allow for a wide range of new business opportunities, and also enable more efficient use of resources such as automated heating, better use of lighting, smarter irrigation and much more. Connecting sensors to virtually everything enables systems to be built that that can optimise usage, detect leakage or enable control of systems in hazardous environments, for example. The IoT has already started to take shape and there are today many examples of early implementations such as smart electricity meters, lamps, door locks, body sensors, temperature sensors, thermostats and more.

Since many of these devices are mobile (e.g. body sensors), placed in remote areas (e.g. flow sensors in sewer systems) and/or are ad-hoc, wireless access is required. To keep the cost of individual devices low, large scale manufactured batteries with limited capacity are used. At the same time devices require long lifespans, meaning that the power consumption must be held extremely low. For example, a device powered by a CR2032 battery with a capacity of 200 mAh can have a lifetime requirement of several years. For comparison, a WiFi (IEEE 802.11a/b/g/n/ac) router can have an average power consumption in the range of watts, which would give a battery life of less than one hour using the aforementioned battery type. It is obvious that WiFi is not suitable and that other standards therefore must be used.

Low power wireless access technologies have been, and are still a hot research topic. During the last few years we have seen a huge increase in low power wireless access technologies specifically developed for the Internet of Things. These include IEEE 802.15.4 (802.15.4), Bluetooth Low Energy (BLE) and IEEE 802.11ah (Wi-Fi HaLow) among others.

Many of these technologies do however only provide access to isolated networks where all devices use the same wireless access technologies and are not connected to what we today refer to as the internet. To allow for interoperability between this abundance of different wireless access technologies, the Internet Protocol (IP) can be used. In previous years, solutions for bringing IP connectivity to low power radio access technologies have been developed.

The target has been Internet Protocol version 6 (IPv6) due to its large address space and autoconfiguration abilities for addresses. The large address space can accommodate all possible IoT devices and the autoconfiguration of their addresses

makes it easy to manage large networks of devices with limited User Interface (UI). IPv6 also enables interoperability between networks and is a key building block for creating the IoT [1]. When developing IPv6, high speed wired networks without power restrictions, such as Ethernet, was the intended physical network considered [2]. As a result, IPv6 is not suitable for the IoT as is. Therefore, research is being performed on how to enable IP traffic over low power wireless access networks. The adaptation layer 6LoWPAN was developed to bring IPv6 to 802.15.4 by providing header compression and fragmentation of large IP Service Data Units (SDUs) [3, 4]. This was later also brought to BLE under the name 6LoWPAN for Bluetooth Low Energy (6LoBTLE), slightly modified because of the differences in the technologies [5].

Even though solutions exist for connecting BLE and 802.15.4 networks to the internet, it is still an open research topic. Future developments of the radio access technologies and use cases add new challenges for supporting IPv6. Also, questions remain regarding how different wireless access technologies compare to each other under IP traffic. Since one main focus of technologies for the IoT is to keep the energy consumption low, it is of great interest to see if it is possible to enable IPv6 without forsaking this aspect. A benchmark of technologies is of great use for implementers and also for developers to identify strengths and weaknesses for future improvements.

1.2 Motivation

The thesis will focus on BLE and 802.15.4. BLE is interesting because of the number of devices shipped (2.8 billion devices 2015) and the activity regarding standardisation and development of new applications [6]. The other radio technology, 802.15.4, is seen as a reference standard for home and industrial automation with many large companies engaged in the development of standards on top of it [7]. It is also one of the earliest standards which is still being updated, with the latest version being released in 2015. While Wi-Fi HaLow is also assumed to influence the IoT ecosystem, it is not yet fully standardised at the time of this work and it is therefore not possible to measure the performance compared to existing technologies without making uncertain assumptions.

Internet connectivity for 802.15.4 is provided with the adaptation layer 6LoWPAN, released in 2007 [3]. For BLE, there exists a similar adaptation layer with the name 6LoBTLE. This was released in October 2015 [5]. While solutions that enable IPv6 for both BLE and 802.15.4 exist, there is still much activity in the standardisation work around improving IPv6 connectivity for low power radio technologies. This thesis will focus on evaluating the current standards, identifying issues and proposing improvements for future iterations of the standards.

The study will not consider a network in a mesh topology. The targeted Bluetooth version is 4.1, which only supports a star topology for IPv6. Therefore the study will only consider star networks without IPv6 routing. A more detailed description of assumptions and limitations of the study is provided in section 3.

1.3 Purpose

The purpose of this thesis is to evaluate and develop solutions for Internet connectivity over BLE and to benchmark the performance of existing and developed solutions against 802.15.4 in specific use cases. More specifically, the performance of BLE and 802.15.4 will be benchmarked for IPv6 traffic with and without 6LoWPAN or 6LoBTLE header compression. Specific issues, such as how to handle IPv6 multicast destinations and how to establish and maintain IPv6 networks will also be studied.

1.4 Previous Work

For BLE and 802.15.4 there exist many studies on the performance in terms of latency [8–11] and energy efficiency [9, 11–16]. Some studies only consider a single device disturbed with interference [13] while others consider larger networks of devices [8, 14]. There are also studies that only measure the performance of a single device without any interference [9, 12]. In [15], IPv6 over BLE and 802.15.4 is studied, but only for single devices in a controlled environment. There are no studies where the performance of these radio access technologies is benchmarked in a real use case and where IPv6 traffic is considered. Therefore the novel contributions of this thesis are to consider IP traffic with and without 6LoWPAN solutions in a homogeneous set up for both BLE and 802.15.4, and to propose improvements for existing standards.
Other pieces of work that have particularly contributed to this thesis are referenced throughout the report.

1.5 Outline

The thesis is organised as follows.
Chapter 2 describes the theory. The aim of this chapter is to give the reader the necessary theoretical background required to follow the remaining thesis with ease. It includes some background on BLE, 802.15.4, IPv6 and 6LoWPAN, with focus on the parts that are relevant to the studies performed. The interested reader is directed to the standard documents in [2–5, 17–19] for more information.
Chapter 3 describes the method used when performing the studies. This includes defining performance metrics, assumptions made and a detailed description of the implementation of the simulator.
The simulation results are presented in chapter 4 and the thesis is concluded with a discussion of the results and suggestions for future research in chapter 5.
Appendices include a closer description of the Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA) functionality in 802.15.4 in appendix A and details of the simulation scenarios in appendix B. Appendix C covers the 6LoWPAN parameter settings used in the simulations.

2
Theoretical Background

This chapter presents the necessary theory required to follow the remaining report. The simulated protocol stack is shown in figure 2.1 and the corresponding protocol layers are described here. The two radio technologies are described with a focus on the parts that have been implemented, as the entire standards were not implemented. The specifications used for this work are available in [17, 18]. A brief description of the most important features of IPv6 for this study is presented, but the reader is assumed to have a basic knowledge of this protocol. For more information about IPv6, [20] is recommended. Finally, the adaptation layer 6LoWPAN is described in some detail.[1]For more information on 6LoWPAN, [3–5, 19] are recomended.

2.1 Bluetooth Low Energy

Bluetooth Low Energy is a Low Power Wireless Personal Area Network (LoWPAN) technology developed by the Bluetooth Special Interest Group (SIG). It operates in the unlicensed 2.4 GHz Industrial, Scientific & Medical (ISM) band and the standard defines functionality from the physical layer to the application layer. The version targeted for this work is 4.1, adopted in December 2013 [17]. Subsequent releases add functionality such as extended packet size and improved security measures, which can be of interest for improving the performance under IPv6 traffic. However, the majority of devices on the market today implement version 4.1 and this is also the version implemented in the simulator.
The description of the technology will begin with the physical layer and will then continue up the protocol stack to describe the implemented layers. For IPv6 traffic, the BLE-specific Transport and Application layers are not used and are therefore not covered in the theory.

2.1.1 Physical Layer

The physical layer is responsible for transmitting bits over the air. The physical channels used by BLE are located in the 2.4GHz ISM band with a spacing of 2 MHz. This gives a total of 40 channels with center frequencies according to

$$f_{\mathrm{k}} = 2402 + k \cdot 2 \text{ MHz} \ , \ k = 0, ..., 39 \, , \tag{2.1}$$

Where k indicates the channel index and f_{k} is the center frequency of the channel with index k. The bandwidth of the transmissions is 1 MHz and the output power

[1]With 6LoWPAN, both 6LoWPAN and 6LoBTLE are meant throughout this report.

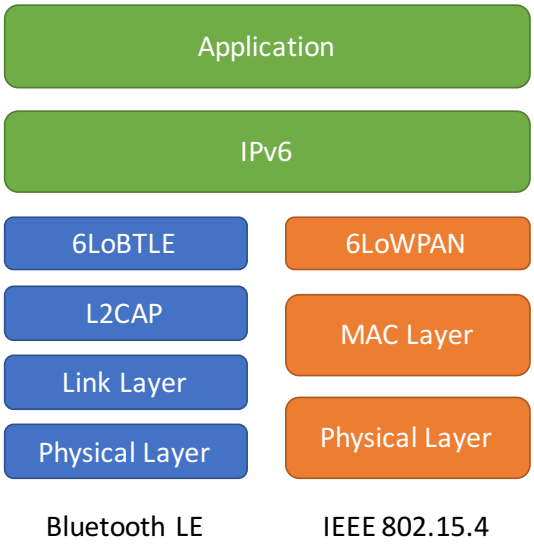


Figure 2.1: The simulated protocol stack.

lies in the range between -20 dBm and +10 dBm (0.01 to 10 mW). The modulation scheme used is Gaussian Frequency Shift Keying (GFSK), which is a variant of the standard FSK modulation scheme where the input signal to the modulator is filtered with a Gaussian filter that smooths the transitions between the input baseband bits. The result is a transmission that does not use as much bandwidth but is more prone to Inter-Symbol Interference (ISI). FSK is also useful since the amplitude of the output signal is constant, which allows the hardware to be designed using low-cost amplifiers that need not be linear. The resulting bit rate is 1 Mbps.
In the simulations, the physical layer was modelled using the physical channel indexes, the bit rate and the Bit Error Rate (BER). By not implementing the entire functionality of the physical layer, the execution speed of the simulator was increased. The reduced level of detail reduced the complexity of the channel model and implementation while still providing an accurate model of the physical layer.
The channel indices allowed interfering transmissions to be identified and the bit rate was used to calculate the duration of a transmission. A model for the BER in an Additive White Gaussian Noise (AWGN) channel is expressed as

$$\text{BER} = erfc(\sqrt{0.5 \cdot \text{SINR}}), \tag{2.2}$$

where SINR is the signal to interference and noise ratio (received signal power divided by noise and interference power) [21].

2.1.2 Link Layer

The BLE link layer is responsible for establishing and maintaining connections between two devices. In BLE, the link layer follows a state machine, shown in figure 2.2. The scanning state is not used for establishing connections and will therefore not be described here.

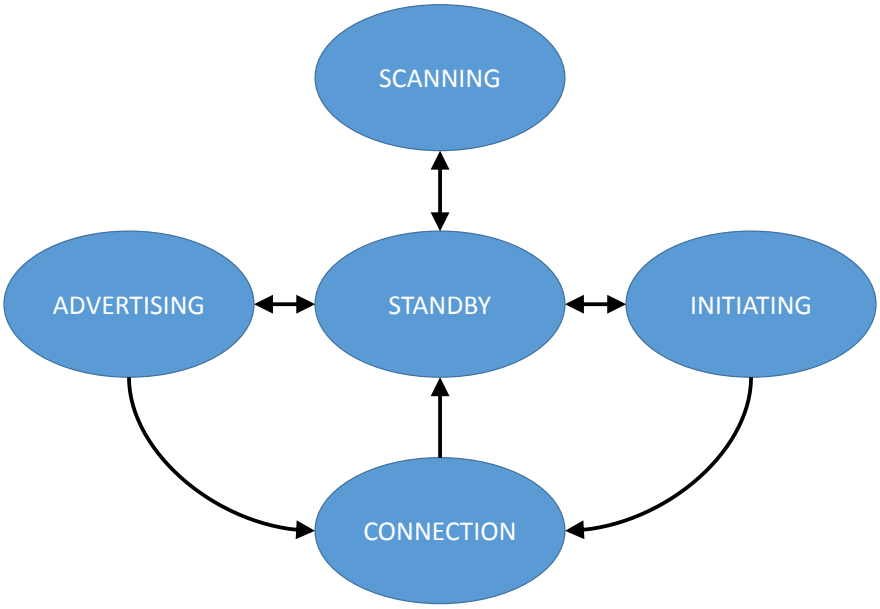


Figure 2.2: State Machine used in link layer of Bluetooth Low Energy.

The most important state is the standby state. In this state the device uses very small amounts of power and can therefore stay functional for a long period of time. As can be seen in figure 2.2, all other states can transition to the standby state.
The other states are more complex than the standby state as the functionality is more advanced. We provide here a short example of a sensor device transmitting data to explain how the different states function.
A device that wishes to transmit a packet will transition to the advertising state. In the advertising state the device will send periodic advertisements on three shared advertising channels, with physical channel index 0,12 and 39 shown in figure 2.3. These values are chosen to spread the transmissions across the available spectrum, and to avoid interference with WiFi[2] [22]. The periodicity is a parameter which can be modified in implementations. For our simulations we chose 20 ms, which is the lowest value allowed by the standard. The maximum value is 10.24 seconds. In addition, a random delay between 0 and 10 ms is added to each advertisement event to avoid synchronised transmissions between different devices in the same network. The advertisements can be of different types; connectable undirected, connectable directed, non-connectable undirected and scannable undirected. In the studies performed, only the connectable directed advertisement Protocol Data Unit (PDU) has been used. It contains the source link layer address of the device wanting to establish a connection and the target link layer address of the device it wishes to connect with.
The advertising device will continue to transmit advertisements until it receives a connection request from the targeted device or until the advertisement supervision timer expires. The supervision timer expires after 200 ms in this study, which is

[2]BLE channel 12 lies between WiFi channels 1 and 6.

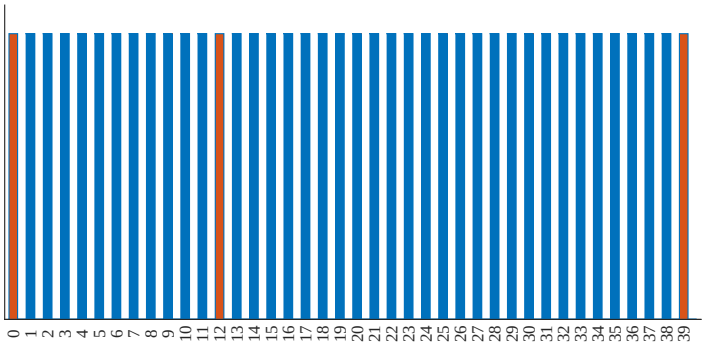


Figure 2.3: BLE data and advertisement channels with physical layer indexing. Channels marked in blue are data channels and channels marked in red are advertisement channels. The channel indexes are mapped to carrier frequency according to (2.1)

chosen to minimise the energy consumption.
The target device must be in the initiating state to receive advertisements (advertisements can also be received in the scanning state, but devices cannot transition to the connected state from there). In the initiating state the device scans the advertising channels for advertisements. Upon receiving a connectable directed advertisement PDU with it's link layer address as the target address, it will reply with a connection request on the same advertising channel. The connection request contains parameters for the connection, such as a delay until the connection begins, the periodicity of connection events and more. When the connection request is transmitted, the initiating device transitions to the connected state. When the advertising device receives the connection request, it transitions to the connected state.
In the connected state, there are two sub-states; master and slave. The initiating device becomes the master and the advertising device becomes the slave. In the connected state, the master device controls the timing of connection events, which are events where the devices exchange data. Between the connection events the devices can turn of transmitters and receivers to save energy.
After the connection request is transmitted and received, there is a delay until the first connection event. This delay is set to 1.25 ms in the simulations. The connection event is started when the master device transmits a data packet on one of the dedicated data channels. Then data is exchanged either until there is no more data to exchange or until the connection event closes because the next connection event is coming up. In the simulations, the periodicity of connection events is 7.5 ms, meaning that there is a connection event every 7.5 ms. The slave device does not need to wake up for each connection event, but can use slave latency to only wake up for a reduced number of connection events. For instance, a slave latency value of 2 allows the slave to only listen for the master on every third connection event. This allows the slave to save more energy but still being able to wake up to transmit data if needed.
Each connection event starts when the slave receives a packet from the master. If the master does not have any data to transmit, it transmits an empty data packet, 10

bytes. This empty data packet is also used as acknowledgement when data is being exchanged during the connection event. During the connection event, transmissions are spaced with an inter frame space of 150 μs.
To terminate a connection, either the slave or master device can transmit a termination command. When receiving such a command, a device shall send an acknowledgement then transition to the idle state.
This example is very simplified and there is much more details regarding the BLE link layer in [17]. In the standard there are more figures describing events, there is more explanation behind all parameters and also more details on what is allowed and not. In order to understand the simulations, this should provide sufficient background on the BLE link layer state machine and states.

2.1.2.1 Resolvable Private Addresses

In the previous example, a device that has data to transmit sends a connectable directed advertisement containing the link layer address of the destination and itself. These link layer addresses can be of different types; static/public, resolvable private or non-resolvable private [17].
A public link layer address is an address that is static and never changes. It can be exchanged and stored in a white-list for reconnection at a later time.
A non-resolvable private address is usually a completely random address used for a single connection. As the name implies, it cannot be resolved by a receiving device and can therefore not be used to verify the identity of a device attempting to connect. A device can use multiple non-resolvable private addresses for different connections, and can also change address upon reconnecting to a previous peer.
The resolvable private address can also be a random address, a device may use different addresses for different connections and it may also change an address at will. The difference compared to a non-resolvable address is that after bonding with a peer device, the peer device can resolve a new address if it changes, but a third device (or eavesdropper) will not be able to identify the original device based on the new link layer address [17].
This is achieved by splitting the link layer address into two 24-bit blocks where the first block is randomly generated and the second half generated by taking the hash of the first block with a known key. On bonding, the key can be exchanged in an encrypted message exchange, giving the peer device access to the key. When a new address is generated, the peer device can then uses the key to calculate the hash on the first part of the new address. If the result of the hash matches the second part of the new address, it corresponds to the device associated to the key [17].

2.1.2.2 Packet Structure

The general link layer packet structure is shown in figure 2.4. From the figure it is clear that the shortest packet size is 80 bits (10 bytes) and the largest is 376 bits (47 bytes). These short packets combined with the high bit rate reduce the hardware complexity since the frequency drift due to heating during the transmission of a packet will be small and the receiver does therefore not need to track the frequency drift [22]. In the general link layer packet, different types of packets can

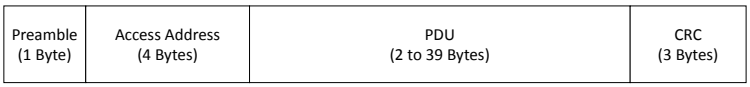


Figure 2.4: Packet structure for Bluetooth Low Energy.

be encapsulated. These are either used on the advertising or data channels. On the advertising channels, we are interested in advertising and initiating PDUs. There also exist scanning PDUs, but those are not used in the simulations and are therefore not considered here. As previously mentioned, there are four different advertisement PDU types. All advertisement channel PDUs begin with a 2 byte header, followed by the payload.
For the connectable directed advertisement PDU, the payload is 12 bytes. The first 6 bytes make up the link layer address of the advertiser and the following 6 the link layer address of the target. The connection request PDU sent in response consists of the same first 12 bytes, followed by 22 bytes carrying parameters for the connection.
For the data channel PDUs, an additional 4 bytes are reserved in the payload for a potential Message Integrity Check (MIC) in case link layer security is used. The header is still 2 bytes but the fields are not the same as for advertising channel PDUs. The data packets used to transmit data during connection events can carry 27 bytes of payload. The 4 bytes for MIC can not be used for additional data when link layer security is not used, which is a decision made to simplify the functionality in the receiver. In the header, the length field consists of 5 bits, giving a range of 0 to 31 bytes. Since 4 bytes are used for MIC, the remaining size is 27 bytes.
For link layer command PDUs, which are also carried in data channel PDUs, the format is as follows; the first byte specifies an operation code, where for example 0x02 is a termination command for terminating a link layer connection. The remaining 0-26 bytes are data carried with the command. For the termination command the command data is a single byte carrying an error code which details why the connection was terminated.
To accommodate larger packets from higher layers, there is functionality for Fragmentation and Recombination in the L2CAP layer. As a packet from a higher layer arrives that does not fit in a single link layer data packet, it is fragmented as follows; first, 4 bytes if L2CAP header is prepended to the packet. Then the packet is simply fragmented and encapsulated in a number of link layer data packets. The link layer header carries a field related to fragmentation, which describes whether the payload is the first fragment in a stream or not. This method is therefore very lightweight in terms of overhead.

2.1.2.3 Frequency Hopping

As there are 37 data channels to use, BLE implements Adaptive Frequency Hopping (AFH) to reduce interference. For each connection event, a new data channel is used for communication. The adaptive part is because the master can decide to remove some data channels from use, if for instance WiFi is being used in the vicinity. When

establishing a connection, the master device decides which channels to use and can therefore remove channels with high interference from use. By only using the "best" channels, interference can be suppressed. For instance, if WiFi channel 1 is in use, it will interfere with BLE physical channels 1 through 8. With AFH, these channels will not be used and thereby the two technologies can coexist.

2.2 IEEE 802.15.4

The IEEE 802.15.4-2011 standard specifies the MAC layer and several different physical layers. Higher layers are defined by implementers of the standard, such as Zigbee. In this work, only the Direct Sequence Spread Spectrum (DSSS) O-QPSK physical layer is considered.

2.2.1 Physical Layer

The physical layer considered is the DSSS O-QPSK physical version defined in [18]. It operates in the 2.4 GHz ISM frequency band as well as sub-GHz ISM bands. Only the 2.4 GHz mode is mandatory and it is the only mode considered in this study. In contrast to BLE, 802.15.4 does not use any frequency hopping techniques [18]. All communication within the same Personal Area Network (PAN) therefore happens on the same channel.

The modulation happens in several steps. In the first step, 4 bits are mapped to one out of 16 pseudo-random noise sequences consisting of 32 chips. The resulting chip stream is modulated using O-QPSK with a half-sine pulse shaping. The result is a signal which is spread across additional spectrum. The bandwidth used is 2 MHz and the resulting bit rate is 250 kbps. Another useful value is the symbol rate, which is used when timing events in the MAC layer. It is 62.5 kbps. The standard does not specify a span for the output power, but a lower limit for the maximum output power is defined to be -3 dBm. The minimum allowed transmit power is not defined.

Again, the details of the physical layer are not simulated. Instead it is modelled using the bit rate and bit error rate. The bit error rate for a DSSS O-QPSK receiver in an AWGN channel is given by

$$BER = \frac{M_s/2}{M_s - 1} \sum_{i=1}^{M_s-1} \left(\binom{M_s - 1}{i} \frac{(-1)^{i+1}}{i+1} \exp\left(\frac{-i \cdot K_b}{i+1} \cdot \mathrm{SINR} \right) \right), \tag{2.3}$$

where $M_s = 16$ and $K_b = \log_2 M_s = 4$ [23]. M_s is the number of symbols in the modulation, in this case the number of pseudo-random noise sequences that are used for mapping bits to chips. K_b is the number of bits per symbol (i.e. pseudo-random noise sequence).

The physical packet structure is shown in figure 2.5. From the figure it is clear that the physical layer imposes an overhead of 6 bytes to the transmission of each packet.

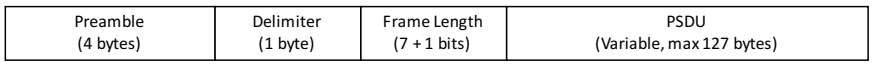


Figure 2.5: IEEE 802.15.4 physical header format.

2.2.2 MAC Layer

The MAC layer in 802.15.4 can operate in two modes; beacon-enabled and non beacon-enabled. In this study, only the non beacon-enabled version has been implemented. Furthermore, 802.15.4 devices can be either full-function devices (FFD) or reduced-function devices (RFD). There are some differences in the complexity of these devices, where FFDs can be coordinators of networks while RFDs can only be end-devices. In the simulations performed for this study, no management of the 802.15.4 network was simulated. Therefore all devices implemented identical functionality.
In the non beacon-enabled mode, the MAC layer of 802.15.4 is not very complex compared to the link layer in BLE. There is no state machine or connection establishments in the link layer. The parameters of the network are broadcast in beacons. In the non beacon-enabled mode, a device that wishes to join a network requests the transmission of a beacon and then begins the process of registering with the network coordinator. The connection process is not considered in this study and will therefore not be presented here. It is however worth mentioning that devices are addressed using a 2 byte PAN identifier together with either a 64-bit extended unique identifier or a 2 byte assigned device identifier.
The channel access is handled with a CSMA-CA process, which is described in more detail in appendix A. Simply put, a device with data to transmit will initiate a random backoff after which it will perform a clear channel assessment (CCA). If the channel is idle, it will transmit the data, otherwise it updates the CSMA parameters and initiates a new random backoff. The maximum number of retries if the channel is busy is by default 4, after which a channel access failure occurs and the packet is discarded (a total of 5 attempts). The CSMA parameters include a counter of the number of times the channel was sensed busy, and a backoff exponent. The backoff exponent is used to generate the random backoff according to

$$\#\text{ of backoff periods} = \text{Random}(0, 2^{BE} - 1)\,,$$

where BE is the backoff exponent and the backoff period corresponds to 20 symbol periods [18]. The initial value for BE is 3 and the maximum value is 5. The value is incremented when the channel is sensed busy according to

$$BE = \min(BE + 1, macMaxBE)\,,$$

where $macMaxBE$ is the maximum value allowed with the default value 5.
Whether to acknowledge transmissions or not is optional and depends on the setting in the received packet. If an acknowledgement is required, the receiving device shall

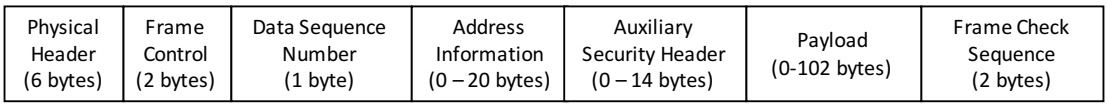


Figure 2.6: IEEE 802.15.4 general MAC packet format, including the physical header.

reply with an acknowledgement 12 symbol periods after receiving the packet (i.e. the turnaround time). The acknowledgement is sent without using Carrier Sense Multiple Access (CSMA), as it is assumed it can be transmitted in connection to a successful transmission. Because of this, the timeout value when awaiting an acknowledgement can be set very short. The value used is based on the size of the physical header and calculated as

$$\begin{aligned} \text{Timeout} &= \text{Turnaround} + \text{Physical Header} + \text{Acknowledgement payload} \\ &= 12 + 10 + 12 + 1 = 35 \text{ Symbols} \\ &= 560\mu s\,. \end{aligned} \tag{2.4}$$

The additional symbol in the second row is introduced as a safety measure such that the timeout does not occur at the exact same instant as the acknowledgement is received. The values in (2.4) are valid for a non-beacon enabled implementation of 802.15.4.
After the acknowledgement the next transmission takes place either 20 or 40 symbol periods later, depending on the size of the first packet. If the previous MAC layer PDU was larger than 18 bytes, the larger value is used as inter frame spacing.
If no acknowledgement is received within 560 μs, up to 3 retransmissions may be performed before a retry failure occurs and the packet is discarded (a total of 4 attempts). A retransmission resets the parameters used for CSMA, meaning that the backoff exponent and busy count are reset to the default values for each retransmission.

2.2.2.1 Packet Structure

The general MAC layer packet structure is shown in figure 2.6. From the figure it seems that many fields can be left out, such as the address fields and the auxiliary security header. This is true in some cases. For instance, when link layer security is not used, the auxiliary security header is not carried. When communication is local to the PAN, either the source or destination PAN identifier can be elided.
There are four different frame types; beacon, data, acknowledgement and MAC command. Since the network is assumed to be non beacon-enabled, we assume that beacon frames are not transmitted. Neither are MAC command frames, which are used to establish and maintain the PAN on MAC layer level. This leaves data and acknowledgement frames. The structure of data frames is very similar to the general MAC packet format shown in figure 2.6. The only difference is that the addressing

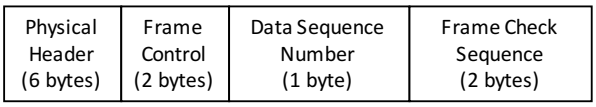


Figure 2.7: IEEE 802.15.4 acknowledgement frame format, including the physical header.

fields have a minimum total size of 6 bytes. The maximum size of the payload is 102 bytes without security enabled and 81 bytes with.
Finally, the acknowledgement frame is shown in figure 2.7. It is 5 bytes, where the identification is performed using the sequence number. The sequence number is a byte that is randomly generated in each device. The probability of two devices having the same sequence number and both expecting an acknowledgement simultaneously is therefore very low.

2.3 IPv6

IPv6 is a network layer defined originally in Request For Comments (RFC) 2460 [2]. Since then, many RFCs have defined additional functionality. In this study we do not consider the entire IPv6 standard, but focus on the header, autoconfiguration of addresses and neighbour discovery. Since the network topology is considered to be a star, no routing protocols are studied. Security is provided by higher layers so the security functions of IPv6 are not considered.

2.3.1 IPv6 Header

The IPv6 header is shown in figure 2.8 and has a static size of 40 bytes. Additional information is carried in extension headers, which are carried in the payload but referenced from the default header. The IPv6 header includes the following fields:

Version 4 bits. Always set to 6.

Traffic Class 8 bits that identifies the class of traffic. Used to provide priority between packets.

Flow Label 20 bits identifying a flow of packets. Still experimental usage.

Payload Length 16 bits specifying the length of the payload in bytes.

Next Header 8 bits specifying the next header, either an extension header or the higher layer header.

Hop Limit 8 bits counting the remaining number of hops until the packet is discarded. Decremented by each router.

Source Address 128 bits specifying the source of the IPv6 packet.

Destination Address 128 bits specifying the next hop destination of the IPv6 packet.

Extension headers often relate to routing with types such as Hop-By-Hop Options or Routing headers, or security with types such as Authentication header. Also

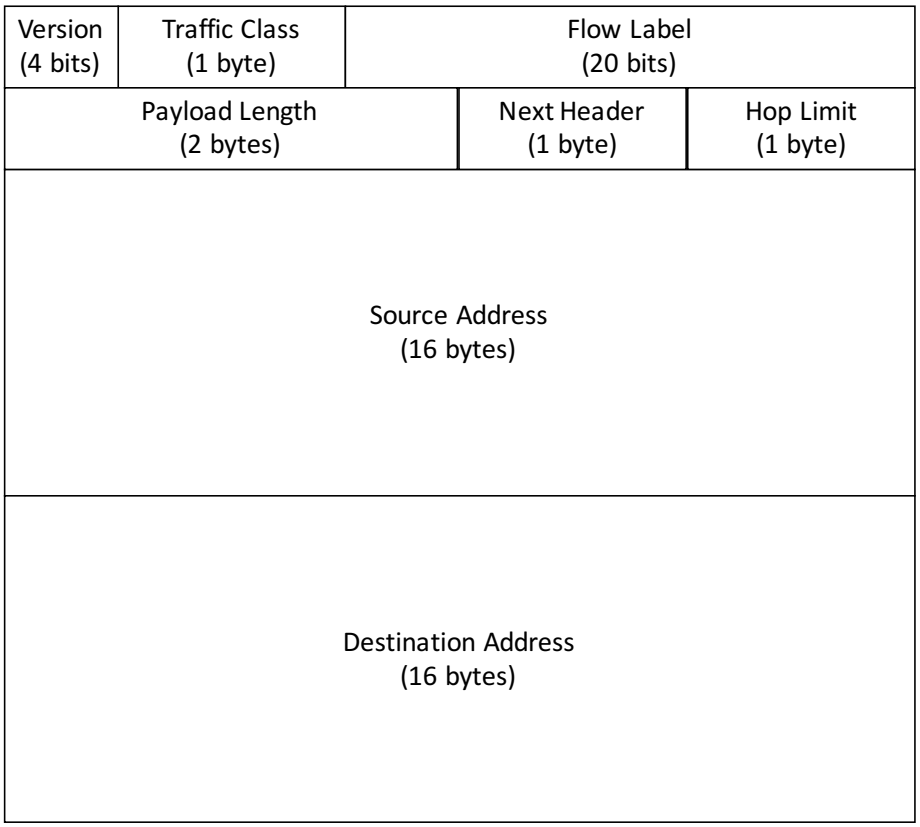


Figure 2.8: IPv6 Header format.

mobility is handled by extension headers, using the Mobility header. In this study, no extension headers were necessary with the exception of Internet Control Message Protocol (ICMP) headers. ICMP headers are not strictly extension headers, but do not qualify as higher layer headers either since ICMP is an internal protocol of IPv6. The ICMP protocol is described in more detail in section 2.3.3.1.

2.3.2 IPv6 Address

The IPv6 address consists of 128 bits represented by 8 groups of 4 hexadecimal numbers separated by colons, for example:

$$2001 : 0DB8 : 0000 : 0000 : 0000 : 0000 : 0123 : 4ABC.$$

To shorten the address, leading zeros can be omitted:

$$2001 : DB8 : 0 : 0 : 0 : 0 : 123 : 4ABC.$$

Finally, the longest consecutive string of zeros can be replaced by a double colon as follows:

$$2001 : DB8 :: 123 : 4ABC.$$

The first part of the address is the prefix, whose length is denoted by a trailing slash:

$$2001 : DB8 :: 123 : 4ABC/64.$$

In this example, the first 64 bits constitute the prefix.
IPv6 addresses can be classified into three groups; unicast, multicast and anycast. Simply described, a unicast address belongs to a single interface on a single device. A multicast address identifies a number of interfaces belonging to one or several devices. A packet sent to a multicast address is delivered to all interfaces it identifies. Finally, the anycast address identifies a number of interfaces similar to the multicast address, but a packet sent to an anycast address is only delivered to a single interface.
In this study we have considered unicast and multicast addresses. Inside these groups, addresses can be of several types. For unicast addresses, there are three addresses of special interest for this study; link-local, global and the unspecified address. The link-local address is defined by the prefix

$$FE80 :: /10\,.$$

It is only valid on a single physical link and can therefore not be routed. Every interface must have a link-local address. In a star topology, a link-local address is valid in the entire star. The global unicast address is defined by the prefix

$$2000 :: /3\,.$$

A global unicast address is most commonly made up by a 64-bit prefix assigned by an authority such as an Internet Service Provider (ISP), followed by a 64-bit interface identifier. The unspecified address is the all-zeros address

$$:: /128\,,$$

and is used before any IPv6 address has been configured. It can not be used as destination address, only as source.
IPv6 multicast addresses have the prefix

$$\text{FF00} :: /8\,.$$

The multicast address type considered in this study is based on the global unicast prefix and is defined in RFC 3306 [24]. It follows the following format

$$\text{FF3E} : 40 : \{\text{64 bit prefix}\} : \{\text{32 bit group ID}\}\,,$$

where 3 is the default value of flags for this type of address, E identifies the address as belonging to a global scope, 40 is hexadecimal for 64 decimal and represents the prefix length. The prefix itself is the same as for the IPv6 subnet and the group ID is a unique identifier for the multicast address.

2.3.2.1 Autoconfiguration of IPv6 Addresses

In IPv6, devices can configure their IPv6 address in multiple ways. To perform autoconfiguration, a device first needs a 64-bit Extended Unique Identifier (EUI-64).

EUI-64 The EUI-64 can be used as a unique interface identifier in IPv6 addresses. If the interface has a 48-bit MAC address, the EUI-64 is created by first inserting the 16-bit FFFE after the first 24 bits of the MAC address. After this, the seventh bit in the EUI-64 shall be inverted to comply with local or global scope. The details for this can be found in RFC 4291 [25].
An EUI-64 can also be randomly generated. Because of the large address space, with 200 devices in a network all randomly generating EUI-64s, the probability of a collision is practically zero. With 1 million devices, the probability of a collision is in the order of 10^{-8}, assuming that the devices have perfect random number generators. Even if not, the probability of a collision is very low.

Stateless autoconfiguration Stateless autoconfiguration is performed without the need for Dynamic Host Control Protocol version 6 (DHCPv6) services. The interface configures its link-local address using the link-local prefix and appends its EUI-64. For a global unicast address, a device must first obtain the network prefix through Neighbour Discovery, then insert the EUI-64 to create the address. To verify the uniqueness of the generated address, Duplicate Address Detection (DAD) is performed. DAD is a method for verifying the uniqueness of an IPv6 address and is described in more detail in [19, 20, 26]. RFC 6775 [19] optimises DAD for low power networks, reducing the process to a simple look-up in a table in the router.
Stateless autoconfiguration is fully autonomous and does not require any manual steps in order to function. This is very useful if the devices in a network are numerous and/or lack input functionality. This makes stateless autoconfiguration desired for IoT applications with IPv6.

Stateful autoconfiguration Stateful autoconfiguration uses the services provided by DHCPv6, or similar services. In this study stateful autoconfiguration has not been used and will therefore not be described in further detail.

2.3.3 IPv6 Connection

Neighbour discovery is a process for a device to join and stay connected to an IPv6 network. To understand the process we must first discuss ICMP.

2.3.3.1 ICMP

ICMP is an integral part of the Internet Protocol and was defined in RFC 792 [27]. It enables diagnostics such as the *ping* command and error reports such as destination unreachable messages. The messages related to ICMP are divided into two categories; informational and error messages. The messages are carried in IPv6 packets and contain a header and a payload. The header contains three fields; type, code and checksum. The type field defines which kind of message it is, for example destination unreachable, echo request or router solicitation. The code field provides additional granularity, for example specifying why the destination is not reachable. Finally the checksum provides some integrity of the message. The payload varies with the types of messages and no length field is required because the length is specified by the IPv6 header.

2.3.3.2 Establishing an IPv6 Connection

In order to establish an IPv6 connection, Neighbour discovery is used. Neighbour discovery is presented in RFC 4861 [26], but RFC 6775 [19] introduces optimisations for low power networks such as those studied here. Therefore the background here will cover the process described in RFC 6775.
When a device is powered on, it does not have any information about routers, network prefix or similar. Therefore, the first step is to transmit a router solicitation from the device.
A router solicitation message is an ICMP informational message carrying a Source Link-Layer Address Option (SLLAO) which contains the link layer address of the originating device.
In a low power network, the EUI-64 is assumed to be unique. Therefore, the link-local address can be generated before knowing anything about the IPv6 network and does not need to be verified using duplicate address detection. The link-local address is used as source address in the router solicitation.
Upon receiving a router solicitation, an IPv6 router will respond to the address in the SLLAO with a router advertisement. The router advertisement contains parameters for the IPv6 network such as current hop limit, flags for how devices generate IPv6 addresses, prefix information, context information (related to 6LoWPAN) and more. The router advertisement therefore becomes quite large in comparison to the router solicitation. An approximate size of the router advertisement (without IPv6 header) is 112 bytes, while the router solicitation is 24 bytes.

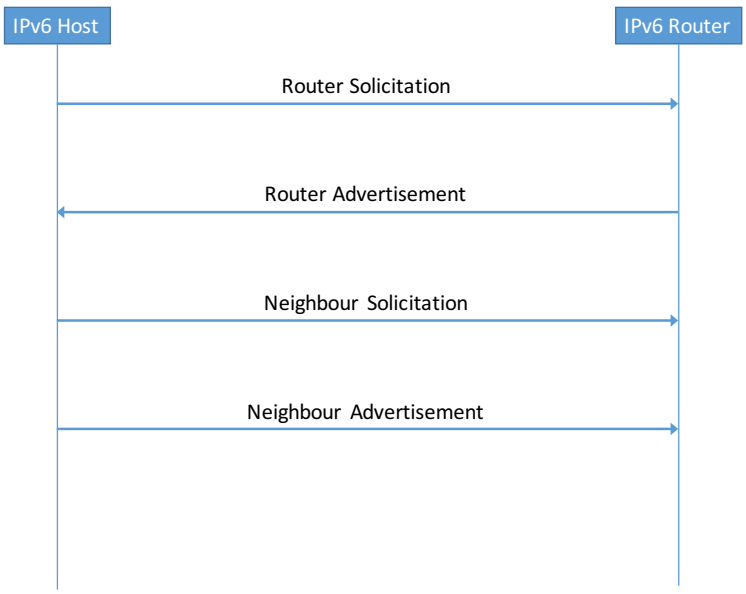


Figure 2.9: Messaging chart for establishing an IPv6 connection.

When the device that wishes to join the IPv6 network receives a router advertisement, it reads the payload and sets parameters in the IPv6 layer implementation. Depending on the flags for address generation, it will either use stateless or stateful autoconfiguration to create an IPv6 address and in the next step attempt to register the generated IPv6 address with the router. The registration is done by sending a neighbour solicitation message with an Address Registration Option (ARO) and an SLLAO. The neighbour solicitation is sent with the configured IPv6 address as source address and the ARO contains the EUI-64 used to create the address.
The router performs DAD by looking up the IPv6 address being registered. The router maintains a mapping from link layer address to IPv6 address for all devices in the network in order to perform duplicate address detection without having to transmit any queries to other devices. If the IPv6 address does not exist in the neighbour cache, the address is unique and not a duplicate. If it does exist, the corresponding link layer address is checked against the one carried in the SLLAO. If these are different, the address is a duplicate and an error message is returned. If they are the same, it means that the device is re-registering the address and it is therefore not a duplicate.
After performing the DAD, a neighbour solicitation message is returned with the status field set to the result of DAD.
The message exchange sequence is shown in figure 2.9.

2.3.3.3 Maintaining Ipv6 Connections

When establishing an IPv6 connection by locating a router, receiving network information and registering addresses, timers are being set in many places. For instance, when registering an IPv6 address, a lifetime for the registration is chosen. When this

lifetime expires, the address is no longer valid and it must be re-registered before it can be used again. The maximum registration lifetime is approximately 45 days, so this is a rare event. But if for some reason the maximum registration lifetime cannot be used, re-registration might happen more frequently.

To re-register an IPv6 address, a device needs to send a neighbour solicitation with an ARO and SLLAO as when registering the same address for the first time. The router responds with a neighbour advertisement message. In case the default router expires, or context information must be updated, a router solicitation must be sent from the device to the router in order to update the network information. These events are all based on different registration lifetimes for addresses, network information and more.

2.4 6LoWPAN

The adaptation layer 6LoWPAN was first introduced for 802.15.4 networks in 2007 with RFC 4944 - Transmission of IPv6 Packets over IEEE 802.15.4 Networks. It has since been updated by RFC 6282 - Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks and RFC 6775 - Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs). In 2015, RFC 7668 - IPv6 over Bluetooth(R) Low Energy was finalised, bringing 6LoWPAN to BLE as well, under the name 6LoBTLE. The goal is to enable IPv6 packets to be transmitted over low power, lossy networks with devices that are sleeping (i.e. transceiver turned off) most of the time.

To achieve this, the IPv6 header is compressed by not transmitting all fields and by making use of how IPv6 addresses are created to perform stateless or context-based compression of IPv6 addresses. To comply with the IPv6 standard, a Maximum Transmission Unit (MTU) of 1280 bytes must be supported. To achieve this, fragmentation functionality was included in the adaptation layer.

In this thesis, we use the term 6LoWPAN universally for both 6LoWPAN and 6LoBTLE. In a star topology, the functionality does not differ much. In 6LoBTLE, no fragmentation is required since it is performed by the L2CAP layer. The header compression is identical, but the internal workings for recreating compressed IPv6 header fields differ. The details regarding how an address is recreated are not considered here, with the exception of noting that in the case of an IPv6 address being based on an EUI-64 generated from a link layer address, the address is reconstructed using the source link layer address in the received packet.

2.4.1 Header Compression

6LoWPAN provides methods for compression and decompression of IPv6 headers, and also some higher layer headers. In this work we have considered IPv6 header compression and UDP header compression. The methods are described below.

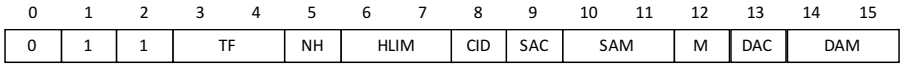


Figure 2.10: 6LoWPAN_IPHC header format.

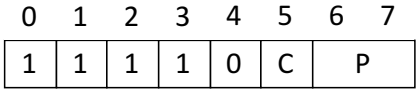


Figure 2.11: Compressed UDP header format used in 6LoWPAN next header compression.

2.4.1.1 IPv6 Header Compression

Header compression is based on clever compression of fields in the IPv6 header, possibly based on a context shared across the LoWPAN. The 6LoWPAN header is shown in figure 2.10. The header fields are described in more detail in appendix C. The details of the compression can be found in RFC 6282 [4].
Only the compression of IPv6 addresses requires a context to be established [3, 4]. Fields such as traffic class and flow label can be entirely elided in a stateless manner. This is because these fields are still somewhat experimental and not widely used. While the fields enable smart scheduling of packets based on priority and flow, such mechanisms are assumed to be of minor benefit to IoT networks. The compression of the hop limit field is very useful in link-local communication such as in a star topology where the hop limit is set to one. The use of an extended context identifier is implementation specific, as there is no standardised way of handling the context in a specific implementation. It is however assumed that a single context identifier does not suffice for compression of all datagrams in a network.
The modes for source and destination address compression are somewhat more complicated. However, for link-local addresses based on EUI-64, stateless autoconfiguration can be used, and the interface identifier can be inferred from the link layer address. Therefore, link-local addresses can be fully elided if the link layer address is the one used when generating an EUI-64. The parameters used for this scenario is SAC = 0, SAM = 3.

2.4.1.2 UDP Header Compression

In addition to IPv6 headers, also User Datagram Protocol (UDP) headers can be compressed using 6LoWPAN. The compressed header format for UDP is shown in figure 2.11. The header fields are explained below.
C Checksum compression
- 0: Checksum not compressed, all 16 bits in-line.
- 1: Checksum elided, 0 bits in-line.

P UDP Port compression

0: No port compression. 16 bits for both source and destination ports in-line.
1: Destination port partially compressed. 24 bits in-line.
2: Source port partially compressed. 24 bits in-line.
3: Source and Destination ports partially compressed. 8 bits in-line.

The UDP length field is always elided and is inferred from the lower layers such as the IPv6 or 802.15.4 headers. The UDP checksum can be compressed when additional integrity checks are used, such as a lower layer MIC, and if it is authorised by a higher layer providing at least the same strength of integrity check on the UDP payload, such as Datagram Transport Layer Security (DTLS).
The compression of UDP ports is stateless and does not require any additional information to be carried in the shared context. By compressing UDP ports, the risk of multiple applications using the same port is increased. It is therefore recommended to only use port compression together with some kind of transport layer security, such as DTLS.

2.4.1.3 Shared Context

Each device maintains a data structure for context information used for compression of IPv6 addresses. The structure is called the context table and contains, as the name implies, context entries. Each context entry is associated to a context ID, the prefix/partial address, a compression bit and the valid lifetime. Table 2.1 shows an example of a context table. The context is shared across the network and each device in the network shall have the exact same information in the context table.
The compression bit is a flag used to determine if this entry is allowed to be used for transmission (compression) or only reception (decompression). When registering a context entry, the compression bit is set to zero meaning it can not be used for transmission. After a time, when it can be assumed that the context has been distributed throughout the network, the entry is updated with the compression bit set to one and can now be used for compression. The same procedure is defined for when an update to a context is performed. In preparation for the update of the actual context information, an update is sent to change the compression bit back to zero. After a number of seconds ([19] defines the time to be more than 300 seconds) the context information is updated, but the compression bit remains unchanged. After an additional time, the compression bit is updated to one [19]. This strict life cycle reduces the probability of a device using a version of the context that is not valid, but introduces delays when updating or setting up the context. The time between creating a context and updating the compression bit to one must also be larger than 300 seconds [19].
The context ID is a 4 bit field that uniquely identifies the context entry in the network. In the 6LoWPAN header, the extended context identifier is 8 bits, meaning that different context entries can be used to compress the source and destination addresses.
The registration lifetime is a 16-bit unsigned integer counter, specifying the time until a specifix context entry is valid. The counter is decreased every 60 seconds.

Context ID	Prefix	C	Valid Lifetime

0x00	2001:db8::/64	1	0xFF13
0xB2	2016:C0DE::/64	1	0xAFE3
0x56	212F:F58E::1/64	0	0xFFFE
0x13	2001:db8::1864:ACFF:CAFE:FEED/128	1	0xDD42

Table 2.1: Example of a context table used for 6LoWPAN header compression.

2.4.2 Fragmentation and Reassembly

Fragmentation and reassembly functionality is only mandatory for 6LoWPAN for 802.15.4 networks, as 802.15.4 does not include any functionality for handling packets larger than the MTU of the link layer.
Fragmentation in 6LoWPAN is performed as follows. On each fragment, a fragmentation header of 4-5 bytes is added. The fragmentation header contains the size of the packet to be fragmented, a unique tag for this packet and finally an offset for the current fragment. The first fragment does not carry the offset field and therefore only carries 4 bytes overhead. The offset field specifies the offset for the current fragment in increments of 8 bytes, which means that each fragment carries a maximum of 96 bytes without security and 72 bytes with.
The choice to include the length in all fragments is made to allow the receiver to allocate memory regardless of which fragment arrives first.
This fragmentation method is less efficient than the Fragmentation and Recombination implemented in the L2CAP layer of BLE. However, the MTU for 802.15.4 is larger than for BLE, thus fewer fragments are required.

3
Method

3.1 Performance Evaluation

To benchmark the two radio technologies against each other, a number of key performance indicators were evaluated. The first performance indicator was the traffic service ratio, or reliability. It was measured on the application layer by comparing the amount of generated and served traffic in terms of bits.

The second performance indicator was the latency, or delay. It was assumed that the requirement was a maximum end to end delay on the application layer of 500 ms. The latency was also measured on the application layer, by associating each transmitted packet with the time of generation and comparing it to the time of reception.

Finally, the energy efficiency, or battery life expectancy, was considered. Since devices are powered with small batteries with limited energy, it is crucial that the power consumption is kept low. The energy consumption was based on measurements from the physical layer in the simulator.

All performance indicators were measured under varying traffic loads as a sensitivity analysis. In the default scenario, the traffic load was approximately 120 bps, as described in appendix B. A scenario with low traffic load (12 bps)and three scenarios with increasing traffic load (240, 480 & 960 bps) were simulated to measure the performance under varying traffic load.

Based on these key performance indicators, further analysis was performed on the performance increase with 6LoWPAN, and how it affects the system. For instance, the header compression reduces the size of the IP packets which in turn can increase the reliability of the network. It can also have an impact on the latency if the number of fragments in lower layers is reduced as a result of header compression, and the energy efficiency should increase when reducing the amount of overhead.

The performance indicators were studied for both BLE and 802.15.4 in several scenarios. The base scenario was a home automation setup specified in appendix B. This base scenario was first simulated with a basic model of IPv6, where the only implemented feature was the overhead in terms of bits. This simulation gave a base performance used as a benchmark between the two technologies and as a baseline for measuring the performance increase from 6LoWPAN. Later simulations studied more advanced models of IPv6 by implementing IPv6 multicast addresses and studying the effects of establishing and maintaining IPv6 networks. For all simulations, a sensitivity analysis was performed to measure the performance when decreasing and increasing the traffic load.

Metric	Description
Service Ratio	Fraction of successfully received transmissions
Latency	Single Hop and End to End Delay
Energy Efficiency	Power consumption and Expected Battery Lifetime

Table 3.1: Key Performance Indicators used for performance evaluation.

Mode	BLE	802.15.4
Idle	2.5 μA	2.5 μA
Tx	9.1 mA	9.1 mA
Rx	6.1 mA	6.1 mA

Table 3.2: Current consumption for BLE and 802.15.4 devices.

3.1.1 Performance Indicators

A more detailed description of the performance indicators, listed in table 3.1, follows here.
The latency was measured on the application layer, to reflect the latency that a user would experience when using the system. Only successful transmissions were used. The single hop delay was measured for all transmissions in the network, while the end to end delay was only measured for traffic from light switches to the corresponding lamps, via the central device. Therefore the end to end delay only contained data from a subset of devices, while the single hop delay was based on every single successful transmission in the network.
The energy efficiency was calculated based on the fraction of time devices spent in different states. The states were idle, transmitter active and receiver active. The current consumption in the different states is shown in table 3.2. The values are taken from the data sheets for the Texas Instruments CC2630 and CC2640 micro-controllers. The values for current consumption were found to be identical between the devices [28, 29]. The micro-controllers are further described in section 3.2.1
The logging of the fraction of time was performed in the physical layer of the implementation. The calculations were performed in the post-processing step in Matlab, assuming a CR2032 battery with a voltage of 3 V and a capacity of 200 mAh.
The service ratio was measured on the application layer. The amount of generated traffic in terms of bits was logged for each device. This was then compared to the amount of received traffic for each device. The amount of generated traffic that exceeded the amount of received traffic was considered to be lost.

3.2 Simulator

To capture all these performance indicators, simulations were performed using Ericsson's system simulator. It simulates real networks and allows the capture of metrics such as throughput, dropped packets, time spent in different states and more.
The capture of performance metrics is done by outputting logs from the simulator

at run time. The logs are implemented as seen fit and can include instantaneous metrics and events such as block error probabilities, latency of a single packet, number of retransmissions and more. However, since the logs are created at run time the aggregation of data is performed in a post-processing stage. For this purpose, Matlab was used.

3.2.1 Assumptions

The most important assumptions are given here with a brief explanation of why the assumption was made, why it is reasonable and how it impacts the results.
First off, the devices in the simulations were assumed to be stationary. This made the implementation easier and it is also be the case for many IoT devices, such as temperature sensors, lamps, proximity sensors and more. In a home automation scenario, which was the use cases studied, the majority of devices are assumed to be stationary and possible mobile devices move slowly. Adding mobility in this scenario with only one central device could introduce coverage issues if a device moves out of range. To avoid such issues, and to simplify the implementation, it was assumed that all devices were stationary.
The channel model used incorporated path loss and shadowing, but no fast fading. It can therefore be assumed to model a slow fading channel, which holds for stationary devices and devices with low mobility. Based on this, it was assumed that only considering stationary devices did not affect the results much while greatly simplifying the simulations and implementation.
Another assumption made was that there were no interfering transmissions from other technologies, such as WiFi. This assumption was made to simplify the implementation and to isolate the core differences between the technologies. Both BLE and 802.15.4 define techniques for avoiding interference, namely AFH in BLE and CSMA + DSSS in 802.15.4. The performance of these techniques is not identical, so adding interfering technologies such as WiFi to the simulations would add another layer of complexity to the benchmark. While interesting for future research, it was considered out of scope for this work.
On the same note, it was assumed that the network simulated was not interfered by another network based on the same technology.
Regarding the energy consumption, values for current consumption for 802.15.4 were taken from the datasheet of Texas Instruments CC2630 [28], a wireless microcontroller supporting the DSSS physical layer of 802.15.4 and has the MAC layer implemented on the ROM. The microcontroller has a CPU which can implement higher layers of the protocol stack (such as 6LoWPAN and IPv6) and applications. For BLE, values are taken from the datasheet of CC2640, also from Texas Instruments [29]. It too has a CPU for implementing higher layers and applications. However, the values for current consumption used in the simulations are based on the assumption that there was no application implemented and does not consider the increased current consumption from processing in higher layers. It was assumed that the main current consumption comes from Rx/Tx operations and that the implementation of higher layers and applications is identical for the two technologies. This assumption simplifies logging and implementation, since the higher layer implementation can be

kept simpler. It however influences the absolute values in the results, such as the expected battery lifetime. Including the power consumption of higher layers would decrease the expected lifetime, but not affect the difference between the technologies much.
As previously mentioned, the network topology was a star topology with a single central device. This assumption removed the need to implement any routing schemes and allowed us to focus on the differences in the lower layers for the performance evaluation and benchmark.
Finally, the simulations were run assuming a steady state of the network. This means that there was no PAN management, introduction of devices to the network or similar. This assumption removed the transient events that only occur once while setting up and focused the simulations on actually running a network. It did not affect the results much, assuming that the majority of devices are stationary and once connected will not change network. While this might affect the results because the technologies have different methods for maintaining the PAN, these effects were assumed to be minor compared to the introduction of IPv6.

3.2.2 Implementation

The simulator is implemented in Java and allows the user to specify the entire system from the positioning of the devices, propagation characteristics, protocol stack implementation and traffic models. A high level description of the implementation follows here.
Of the implementation described below, the implementation of 802.15.4, 6LoWPAN fragmentation and reassembly and some ICMP functionality was performed as a part of this study. The remaining implementation was already in place before this study.

3.2.2.1 Deployment

All devices are assumed to be stationary. Therefore the propagation can be calculated in advance to make the simulations run faster and to keep consistency between iterations. The propagation is calculated between all devices based on the position of devices and the floor plan of the home. Details regarding device placement, floor plan and more can be found in appendix B.

3.2.2.2 IEEE 802.15.4

The implementation of 802.15.4 consists of the MAC and physical layers, as discussed in section 2.
The physical layer is implemented as a subset of the IEEE 802.15.4-2011 standard [18]. The model for the physical layer includes the bit rate to calculate the duration of transmissions and a model of the BER to calculate the packet error rate.
When receiving a packet from the MAC layer, the packet is first encapsulated in a physical transmission containing information about the transmit power, physical channel index, transmitting antenna and more. The transmitting antenna contains the location of the originating device. The transmission object is then passed to a

transmission manager which handles all transmissions. The transmission manager passes all active transmissions on a certain channel to all active receivers which then attempts to receive the transmissions.

The receiver, upon receiving a transmission from the transmission manager, first makes sure that it is tuned to the same channel and has been so for the duration of the transmission. It then calculates the block error probability based on the received power and interfering transmissions. From the block error probability, the frame check sequence is set to true or false based on a Bernoulli trial. The frame check sequence is assumed to be perfect and detects all errors. The packet is finally delivered to the higher layer.

The physical receiver is also responsible to perform the clear channel assessment in the CSMA process. To do so, a dummy transmission is used to extract all active transmissions from the transmission manager. Based on all active transmissions, the received power is calculated and compared to a threshold value. If the received power is above the threshold value, the channel is assumed to be busy, otherwise free. The threshold value used in the study was -75 dBm, in accordance with the standard [18].

The MAC layer follows the non beacon enabled functionality described in section 2.2. As with the physical layer, the implementation is to be seen as a subset of the standard, limited by the assumptions and scope of this study. Upon receiving a packet from a higher layer, the MAC layer entity prepares the packet for transmission by encapsulating it in a 802.15.4 data frame. It is assumed that acknowledgements are required, therefore a flag is set to signal this to the receiving MAC layer entity. If the transmitting entity is not busy with another packet (receiving or transmitting), it initiates the unslotted CSMA process in order to transmit the packet. If the CSMA process ends in a success, the encapsulated packet is forwarded to the lower layer. Otherwise it is discarded.

Upon receiving a packet, the receiving entity checks the Frame Check Sequence to verify that the received packet is not corrupted. If the packet is not corrupted, it checks whether it is the target destination of the packet (or if the packet was broadcast). If the packet should be processed, it is processed and delivered to the higher layer, then an acknowledgement is transmitted if requested. As can be noted, the implementation follows the standard and the limitations presented in section 2.2.

One implementation specific decision is the introduction of sleepy devices. These are assumed to be in an idle state a packet is received from the traffic model. Only then do the devices enable transmitter or receiver in order to exchange data with the network. By doing so, the energy consumption can be kept low, but such devices can not be reached with data from the central device. In the simulations, all sensor devices were assumed to be sleepy. The central device and actuator devices always had their receiver turned on when idle to remain reachable. The MAC layer parameters used for the 802.15.4 implementation in the simulator are shown in table 3.3.

Higher layers are common for both 802.15.4 and BLE and are described further down.

Parameter	Value
macMaxBE	5
macMaxCSMABackoffs	4
macMaxFrameRetries	3
macMinBE	3
macAckWaitDuration	560 μs
macSecurityEnabled	FALSE

Table 3.3: 802.15.4 MAC layer parameters used in simulations

3.2.2.3 Bluetooth Low Energy

The BLE implementation used for the study includes the physical, link and L2CAP layers. The implementation is a subset of the standard, tailored for the needs of this study.
The physical layer follows the same logic as the implementation of the physical layer for 802.15.4. The duration of the transmission is calculated based on the physical bit rate and the size of the packet to transmit. It is then sent to a transmission manager that manages all transmissions on the channel. The receiver calculates the block error probability based on received power and interfering transmissions and then delivers the received packet to the higher layer.
The link layer in the BLE implementation is somewhat more complex. The number of devices in the simulation was relatively high, and BLE devices on the market today have a limit of the number of devices that can be connected simultaneously. Because of this, the link layer implementation terminates connections when both master and slave devices have no more data to transmit in order to not overflow the number of connected devices. This is usually managed by the Generic Attribute Protocol (GATT) layer in a BLE implementation, but this level of detail in the simulations was not assumed to be necessary.
As for 802.15.4, the devices were implemented as sleepy or non-sleepy. The sleepy devices in the BLE implementation never entered the initiating state and were therefore not reachable unless they transmitted data of their own. Non-sleepy devices transitioned to the initiating state after terminating a connection and did not stay in the idle state.
The L2CAP implementation was limited to providing fragmentation and recombination services when receiving packets unable to fit in one packet.

3.2.2.4 6LoWPAN

The 6LoWPAN implementation provided header compression on IPv6 headers and UDP headers (if applicable). It also provided fragmentation and reassembly on top of the 802.15.4 implementation.
The header compression was based on parameters that were set for an entire simulation, meaning that each device used the same header compression throughout the simulation. Since the traffic from each device was well defined, this was not an issue. The 6LoWPAN entity calculated a new IPv6 header size based on the header compression parameters for the IPv6 and UDP headers and performed the opposite

process when receiving a packet.
Fragmentation was performed according to the theory presented in section 2.4.1, but only when the lower layers were 802.15.4, since L2CAP fragmentation and recombination was implemented for BLE.

3.2.2.5 IPv6 and ICMP

The highest layer implemented was IPv6, including ICMP. The ICMP layer is a part of IPv6, but was implemented as a separate layer because of programming reasons. This did not affect the simulation results.
The Ipv6 implementation simply encapsulated received data in IPv6 packets and forwarded said packets to the lower layer entities. The additional functionality was introduced in the ICMP implementation.
The ICMP implementation was limited to neighbour discovery. When a packet is received from the IPv6 entity, a check is performed to see whether the device is connected to the IPv6 network. If not, the received packet is buffered and a Router Solicitation is sent. Upon receiving a Router Advertisement from the lower layer, a Neighbour Solicitation is sent. When a Neighbour Advertisement is received, a connection is assumed to be established and a timer is started for the validity of the virtual IPv6 address. The packet(s) in the queue is transmitted when the ICMP entity enters the connected state.
When the timer for the IPv6 address expires, the device will trigger a re-registration if it expects downlink traffic (i.e. is an actuator device). Otherwise, re-registration will occur when the next packet arrives from the IPv6 entity. The re-registration process followed the message exchange sequence shown in figure 2.9.

3.2.2.6 Traffic Model

The traffic model simulated the application layer by generating packets of a predetermined size with a Poisson arrival rate. In the default case the traffic load generated is approximately 120 bps (on the application layer). The exact values for all cases are given in appendix B. In the traffic model, the performance in terms of latency was measured.

4
Results

4.1 Benchmark

The main focus of this work has been on the benchmark between BLE and 802.15.4, where the two technologies have been implemented in Ericsson's system simulator and evaluated in a home automation scenario with IPv6 traffic. To isolate the effects that correspond to differences between the technologies, minimal additional complexity in higher layers was included in the first simulation. For instance, no 6LoWPAN header compression was included, and neither was any overhead for maintaining or setting up IPv6 connections. The only overhead from IPv6 was therefore in larger packets from the headers.

4.1.1 Steady State Operation with Uncompressed IPv6

The initial simulations were run with limited IPv6 and minimal 6LoWPAN functionality. The header compression, which is a core feature of 6LoWPAN, requires management and setup of a shared context across the PAN. To remove as much complexity as possible and to find a benchmark that can be used as a baseline for future improvements, features such as these were not included in this simulation. For 802.15.4, fragmentation and reassembly is mandatory since the MAC layer only supports a MTU of between 81 and 102 Bytes compared to the required 1280 Bytes of IPv6 [18, 20]. In the simulations, no link-layer security was implemented and the MTU for 802.15.4 was therefore 102 Bytes. In the simulation the packets received never required fragmentation and the 6LoWPAN layer could therefore be left out in order to simulate plain IPv6.
The networks were assumed to be in a steady, connected state with no requirements for Neighbour Discovery, Address Resolution or other IPv6 processes for setup or maintenance. In short, the only overhead of IPv6 in these simulations was the increased packet sizes. The details of parameters used and scenario setup are given in appendix B.
Looking first at the service ratio as a function of the traffic load in figure 4.1, it is clear that BLE serves a higher load with more reliability than 802.15.4. This is also shown in figure 4.2, which shows the generated traffic and loss ratio for each type of device. From left to right the device types are; window sensors, temperature sensors, asset tags, indoor lights and light switches. The device types are further described in appendix B.
The reasons why BLE is more reliable are many, for instance BLE has a bit rate of 1 Mbps while 802.15.4 has a bit rate of 250 kbps. This means that the central

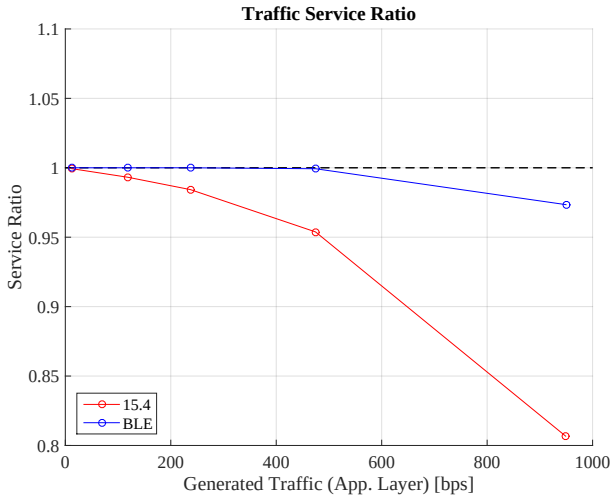


Figure 4.1: Benchmark: Traffic Service Ratio as a function of traffic load. The default scenario corresponds to the second data points (120 bps). The service ratio is calculated as the ratio of the served traffic divided by the generated traffic.

device spends less time in a busy state and is less congested by traffic that it needs to relay. Also, in BLE, data transmissions take place on dedicated channels that do not interfere with advertisements while in 802.15.4 all transmissions take place in the same frequency band leading to collisions and congestion.

The main reason for packet loss in 802.15.4 is because of the retry limit, not the channel access failure. This can be explained with the hidden terminal issue. Because the energy detection threshold is -75 dBm, some devices are hidden from each other due to path loss and shadowing (walls etc.). If there is a collision and retransmission is performed, the CSMA parameters are reset. This limits the maximum backoff to 2.24 ms (140 symbols). The transmission of the packets in the simulation (119 bytes) takes approximately 3.8 ms, which means that the probability of colliding on the next transmission again is high since both devices will attempt transmitting again within 2.24 ms of the acknowledgement wait duration. So, if a collision occurs because the devices cannot detect transmissions from each other, there is a high probability that it leads to a retry limit.

Some back-of-an-envelope calculations on this gives the following. In the default scenario, the load on the channel is approximately 5.25 %. Assuming that the average fraction of hidden terminals is 10 % (7-8 devices) and transmissions are uniformly distributed, collisions occur due to hidden terminals in $5.25 \cdot 0.1 = 0.525\%$ of transmissions. If all collisions due to hidden terminals lead to both transmissions being lost, the traffic loss ratio should be about 1 % (because 2 transmissions are lost for each collision). The simulated traffic service ratio is 99.32 %, which corresponds to a loss ratio of approximately 0.7 %. These 0.7 % include losses because of other reasons and is somewhat lower than the calculations give. However, not all collisions result in losses because there is a probability that the backoff results in a successful transmission later, and the fraction of hidden devices might be lower than 10 %. The majority of losses however occur due to issues with hidden terminals.

The reason for losses in BLE is because of congestion on the advertising channels.

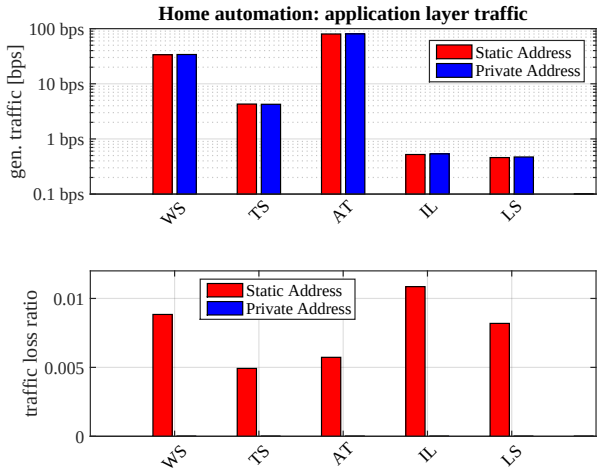
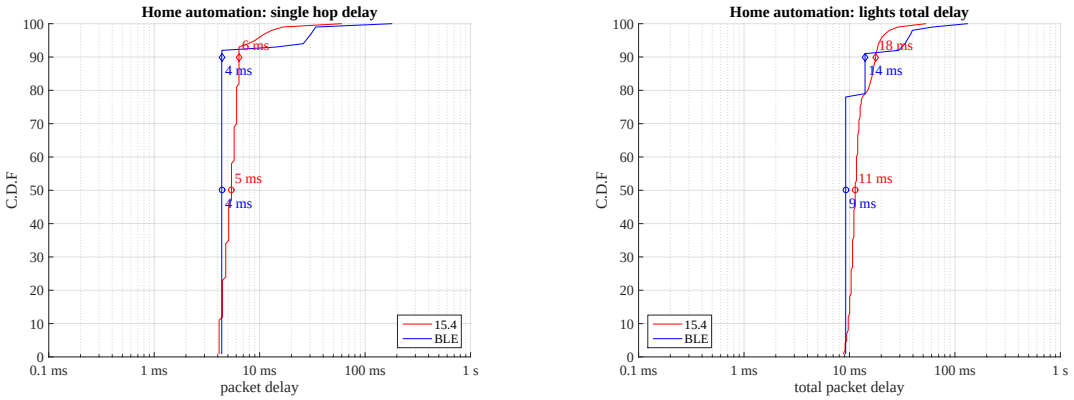


Figure 4.2: Benchmark: Generated and lost traffic per device type.



(a) Single Hop Delay.

(b) End to End Delay.

Figure 4.3: Benchmark: CDF plots of latency in default scenario.

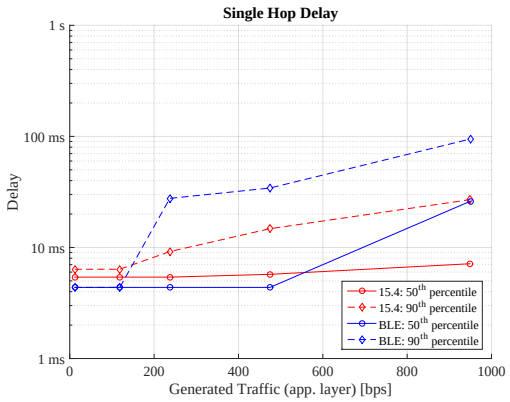
When advertising, there is a supervision timer set to 200 ms in the simulations. If no connection is established within that time, the receiving device is assumed to be (temporarily) unavailable and all packets destined for that device are discarded. For the default scenario, the randomness introduced in advertising events (uniformly distributed between 0 and 10 ms) manages to solve collisions. If a connection is established, the transfer of data takes place on a dedicated channel without interference. Therefore there are no collisions once the connection is established in the simulation.

Moving on to the single hop latency (for successful transmissions) in figure 4.3a, we note that the values are very similar. The single hop latency for BLE contains establishing a new connection by advertising, receiving a connection request and then transmitting the data in the first connection event. For most of the transmissions this takes approximately 4 ms, while some transmissions take an additional 20 - 30 ms because of a lost advertisement or connection request. For 802.15.4, the CDF shows signs of more underlying variance in the delay, since the randomness of the

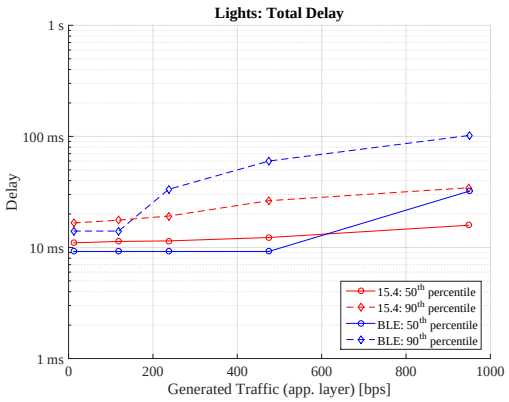
backoff period in the CSMA-CA makes the delay spread out. While 802.15.4 does not require connections to be established, the actual bit rate is lower and the delay is because of this higher. This can be connected to the large packet sizes, noticing that the overhead of establishing a connection in BLE is reduced as the amount of data increases. The payload fraction depends on the payload size, but is comparable for the two technologies. For BLE the maximum payload fraction is 73% and for 802.15.4 (with the simulation settings) it is 78%.

Looking closer at the lights we can study the end to end delay between a light switch and the corresponding light(s), as seen in figure 4.3b. There are 3 light switches that control 2 lamps each while the rest of the switches control only a single lamp each. The majority of data therefore comes from a light switch sending commands to a single light through the central device. Looking first at the latency for BLE we note that the end to end delay is slightly higher than the double single hop delay. The single hop delay does not include the transmission of an acknowledgement, as it is assumed that the received data is delivered to the higher layer in the BLE host while the controller transmits the acknowledgement. But before the relay transmission can begin in the central device, the acknowledgement for the received data must be transmitted and the connection terminated. The termination of a connection requires an additional transmission from either the master or slave device, which introduces an additional delay to the sequential transmissions. This introduces a slight delay. Looking at the 80^{th} to 90^{th} percentiles we see an additional delay of approximately 5 ms. This relates to the switches that control two lamps. Since BLE cannot broadcast IPv6 packets in current implementations, the central device must establish a connection to each lamp in succession, as follows; the central device receives a command from a switch and shall relay this to the two registered lamps. It first connects to one of the lamps and transmits the command. When acknowledged (or failed) it terminates the connection with the first lamp and connects to the second. Therefore the second lamp also experiences the delay of the first. This step in the CDF is not present in the latency for 802.15.4 because of the CSMA-CA. The random backoff smooths the curve since some transmissions experience a longer backoff and others a shorter. Between the two technologies, for the single traffic load analysed, the end to end latency is very similar.

In figure 4.4a and 4.4b, the single hop and end to end delay is shown for different traffic loads. In this figure it is clear that the advertising channels experience congestion when increasing the load and that in more than 10% of the transmissions, connecting on the first advertisement fails when the total load of the network is increased. When increasing the application layer traffic load to about 950 bps, even 50% of the connections fail to establish on the first attempt. Here the CSMA-CA of 802.15.4 performs better as it can avoid some congestion by sensing the channel and backing off. However, this procedure is short and does not manage to solve collisions due to hidden terminals which means that some packets are instead lost rather than arriving late, as can be seen in figure 4.1. For instance, when increasing the load in the 802.15.4 network, the central device needs to relay more transmissions to the lamps. During transmission, the central device cannot receive packets from other devices, meaning that they might be discarded due to retry limit during the time. Also, all transmissions are made on the same channel, leading to collisions



(a) Single Hop Delay as a function of traffic load.



(b) End to End Delay as a function of traffic load.

Figure 4.4: Benchmark: Sensitivity analysis of latency.

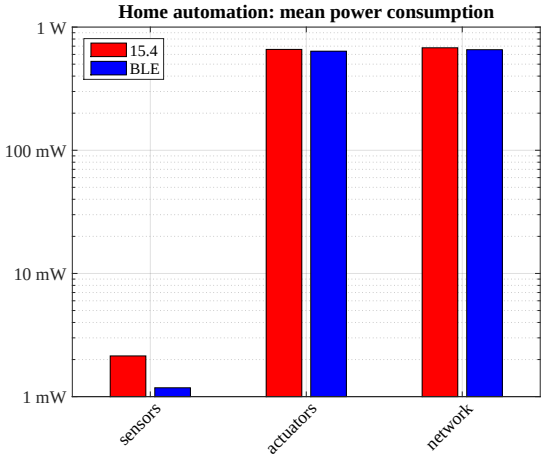


Figure 4.5: Benchmark: Mean device power consumption.

when the traffic load is increased. In total, even for very high levels of traffic, the latency is within 500 ms and should not give the user an experience of slow service for the packets that are successful. However, the service ratio for 15.4 is low when the traffic load is high, which would give a poor experience nonetheless.

The mean power consumption for sensors, actuators and the entire network is shown in figure 4.5. Since the actuators are always reachable and don't turn off their receivers, the power consumption is high in comparison to the sensors. This is also reflected in the average power consumption for the network. Since the sensors don't consume nearly as much power as the actuators, the average power consumption of the network is not affected much by them. From now on we will only focus on the power consumption and expected device lifetime of sensor devices.

A more detailed figure of the sensor devices power consumption is shown in figure 4.6. There it is clear that BLE devices consume less energy than the 802.15.4 devices. This can be explained by the same reasoning for the latency, since BLE

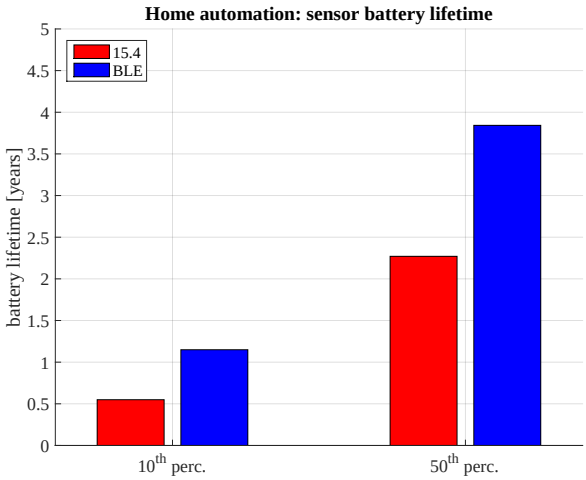


Figure 4.6: Benchmark: Expected device lifetime for sensors.

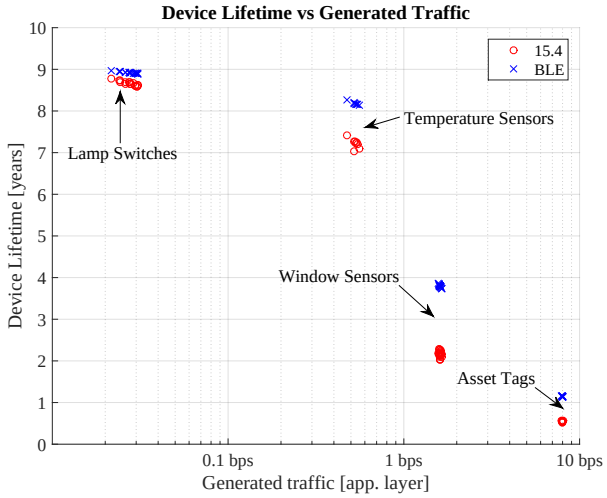


Figure 4.7: Benchmark: Device lifetime as function of generated traffic.

devices transmit with 4 times the bit rate of 802.15.4 devices and the packets are large (reducing the overhead of establishing connections), BLE devices spend less time in active states. The current consumption in the different states is given in table 3.2. The current consumed when transmitting or receiving is more than 1000 times the current consumed in idle state, which is explained by having to power a high-frequency oscillator [22].

Figure 4.7 shows the expected device lifetime as a function of generated application layer traffic per device. From this figure it is clear that the device lifetime is reduced as the traffic load (or rather the frequency of transmissions) is increased. Note that for the temperature sensors the device lifetime seems somewhat higher than expected. This is explained by the fact that those devices generate 4 Bytes per application layer packet, and that they therefore generate more traffic per transmission. It is also noted that the difference in expected lifetime between BLE and 802.15.4 devices is smaller when the generated traffic is low. This is reasonable as

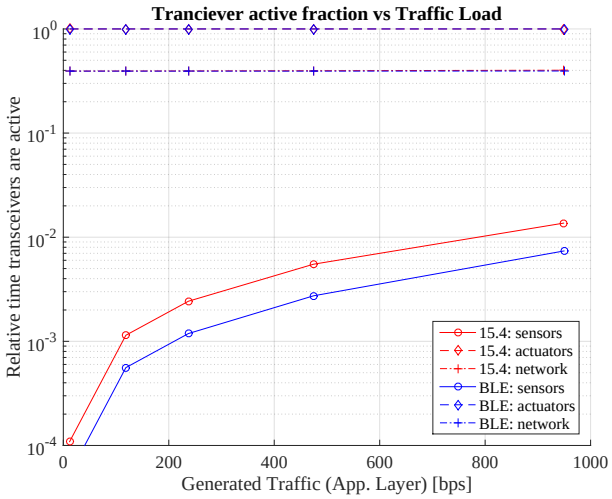


Figure 4.8: Benchmark: Device active fraction as function of traffic load.

the devices then spend most of the time in idle state. While the expected device lifetime appears to be more similar for the asset tags than for the window sensors even though asset tags generate more traffic, the relative difference is actually larger. In figure 4.8, the relative active fraction of devices is shown. Note that the active fraction translates directly to energy consumption, a larger fraction of time spent in active states means higher energy consumption.
Again it is clear that an increased load has a big impact on the devices in terms of active fraction and thereby also energy. We note that for actuators and network there is no major difference across different traffic loads while the relative transceiver active fraction for sensor devices increases as the traffic increases. There is no difference between the two technologies in terms of how the active fraction changes as traffic load is varied. BLE devices in general have a lower time in active states, which is explained with the same reasoning as above regarding energy consumption and expected battery lifetime.

4.1.2 6LoWPAN Header Compression

From the benchmark we learnt that BLE outperforms 802.15.4 in terms of energy consumption and traffic service ratio. The latency is very similar in the base scenario, but for higher traffic levels, the CSMA in 802.15.4 outperforms the ALOHA-approach for establishing connections in BLE. Since we did not simulate the technologies without IPv6, there are no values on how the performance is degraded by the overhead from IPv6 packets. Even so, the overhead in bytes from IPv6 and higher layers (CoAP, DTLS, UDP) sums up to 89 bytes. This means that one byte on the application layer generates 90 bytes transmitted by the IP layer. This is a huge overhead and one of the issues with running IPv6 over low-power radio technologies. This issue was identified early and included in RFC 4919 - IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals [30]. To combat this issue, IPv6 header compression was developed. Therefore the next logical step in simulations was to include this func-

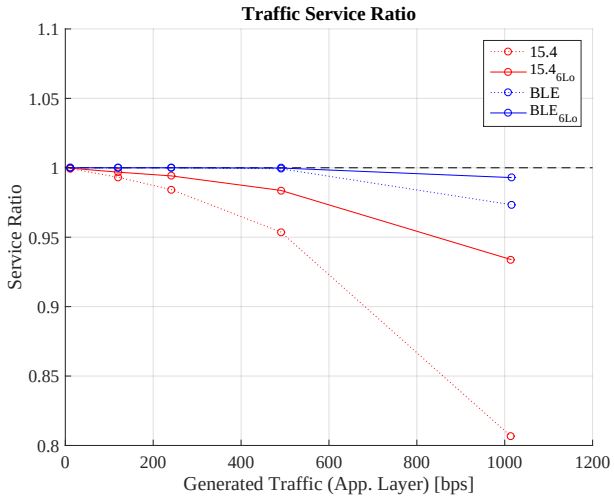


Figure 4.9: 6LoWPAN: Traffic Service Ratio as a function of traffic load.

tionality and see how the performance was affected.
So, for the second simulation, IPv6 header compression was included. It was assumed that the shared context had been previously set up and did not expire, thereby reducing the complexity of the implementation. The header compression parameters used are described in appendix C and are based on the definitions in RFC 7668 and RFC 4944 together with the assumptions for the traffic.
RFC 6282 also defines compression of UDP headers [4]. Since the modelled protocol stack included UDP this was also implemented, reducing the UDP overhead from 8 Bytes to 2 Bytes. The UDP checksum was elided since verification was performed by the DTLS layer.
Looking at the traffic service ratio in figure 4.9 it is clear that introducing header compression increases the amount of traffic being served in the network. Especially for 802.15.4 the service ratio is increased substantially for higher traffic loads. Because all transmissions take place on the same channel, reducing the size of packets leads to less congestion and thereby increases the service ratio. For BLE, the size of the advertisements does not change, neither does the frequency of transmissions. Therefore the congestion on the advertising channels is only affected by the reduced duration of data transmissions that gives less queued transmissions.
Regarding latency, as can be seen in figure 4.10, it is only marginally reduced when introducing header compression. Even if the number of packets transmitted over the air in BLE is reduced after introducing header compression, the overhead of setting up a connection is not reduced. This results in the minor reduction of 2-3 ms (for generated traffic of $\approx$ 120 bps) in the end to end delay. For 802.15.4, the largest delay is introduced by the overhead and transmitting bits over the air, since the packet size is large enough to carry the generated IPv6 packets without fragmentation and there is no need to set up connections. The reduction of 2-4 ms (for $\approx$ 120 bps) is explained partly because of reduced contention and thereby also backoffs, but also by the reduced number of bits being transmitted over the air.
The introduction of header compression reduces the amount of traffic on the link layer because of the reduced IPv6 and UDP overhead. The same 6LoWPAN settings

were used for both BLE and 802.15.4 which is reflected in the similar increase of energy efficiency, as can be seen in figure 4.11. The median lifetime is increased by 23.7 % for BLE and by 33.5 % for 802.15.4. Again the difference in increase relates to the difference in bit rate and MTU size. While the number of bits over the air is reduced more for BLE devices, the higher bit rate does not reduce the active time as much as for 802.15.4 devices since the overhead of establishing connections becomes more prominent. Therefore the improvement is slightly larger for 802.15.4.

4.2 IPv6 Multicast

As we previously saw, the performance is improved by introducing header compression. This however, requires a shared context being managed across the network. How the context is structured is described further in RFC 6775 but not considered further in this study as how to handle it is implementation specific. Instead we focused on another issue, regarding IPv6 Multicast addresses.
IPv6 Multicast addresses are used for transmitting one packet to several destinations without replicating the transmission on the IP layer for the individual destinations. This type of address can be used to reduce transmissions for applications such as lighting, where one switch controls multiple lamps.
To fully support IPv6, multicast transmissions must also be supported by the underlying technology, such as BLE or 802.15.4. On the MAC layer, transmission of multicast packets (in a star topology) can be done by either sending a single broadcast transmission or by sequentially transmitting copies of the original packet to one receiver at a time. For the considered version of BLE (4.1), the only way to transmit IPv6 packets is by setting up a connection to another device and send the IPv6 packets as BLE data packets. The only broadcast support in BLE is advertising packets, with a maximum data size of 31 bytes [17]. This can not be assumed to fit an IPv6 packet, as the uncompressed IPv6 header is 40 bytes and compression can not be assumed to apply to all transmissions. Therefore, IPv6 multicast packets

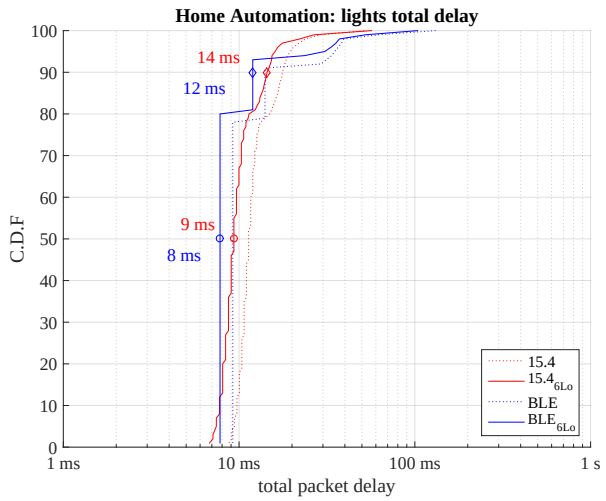


Figure 4.10: 6LoWPAN: End to End Latency.

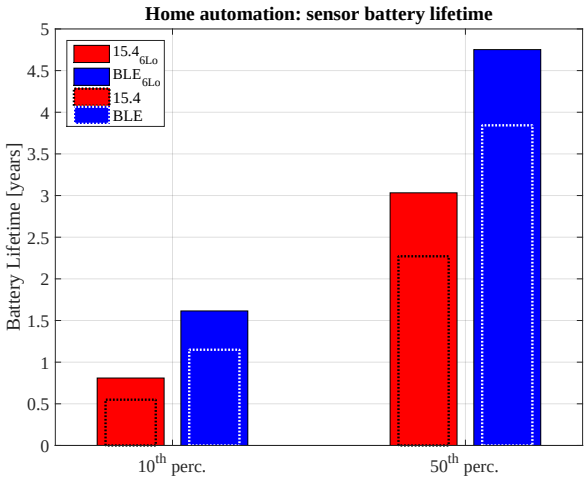


Figure 4.11: 6LoWPAN: Expected device lifetime for sensors.

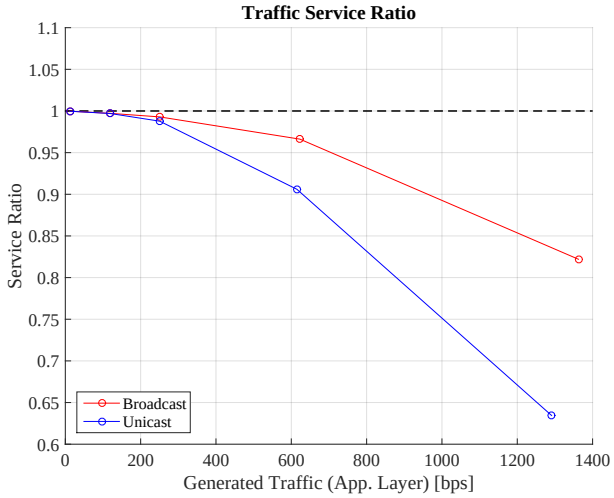


Figure 4.12: Multicast: Traffic Service Ratio as a function of traffic load. Note that this is for 802.15.4 devices only.

must be transmitted in a unicast fashion over BLE.

For 802.15.4, there exists broadcast functionality on the MAC layer. It is therefore possible to transmit IPv6 multicast packets either by broadcasting on the MAC layer or by transmitting sequential copies to the devices that are registered with the multicast IP address. This introduces an implementation choice where the tradeoffs between the two approaches must be considered. For instance, broadcasting should reduce the amount of traffic in the network and also the latency of the packet being broadcast since all recipients receive simultaneously. But at the same time, broadcast transmissions cannot be acknowledged in 802.15.4 which might reduce the reliability since there is no way of knowing whether the transmission is successful or not [18]. This can be countered by a repetition scheme where broadcast packets are repeated to add redundancy or some other scheme for increasing reliability, but this is considered to be out of the scope for this work.

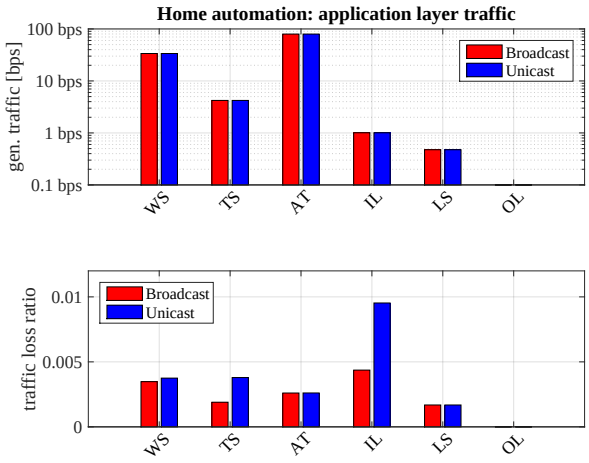


Figure 4.13: Multicast: Generated and lost traffic per device type.

Based on the discussion above, it is not obvious what the effects of these different approaches are. Therefore this has been simulated in a real use case. The use case is the same home automation set up as previously, but in these simulations there is also one additional light switch that controls all lights. The lights are assumed to be registered to multicast IPv6 addresses, even if there is only one light in the multicast group. Then the scenario is simulated when the multicast transmissions are broadcast on the MAC layer and when they are transmitted to each of the registered listeners one at a time, i.e. unicast. The same performance metrics are studied to show the effects on latency, energy efficiency and throughput. Please note here that we have increased the fraction of traffic from light switches to accentuate the differences between broadcasting and unicasting at the MAC layer. The values should therefore not be compared to the previous simulations as-is, with exception for the default scenario which is unaltered.

Looking at the service ratio in figure 4.12, we note a somewhat surprising result. Even though broadcast transmissions cannot be acknowledged, the service ratio is higher when the multicast transmissions are broadcast on the MAC layer than when they are unicast. One explanation for this is that the channel quality is high and that retransmissions due to bad channel conditions are rare. Transmitting fewer packets then reduces the risk of collisions.

Reducing the traffic load also reduces collisions which can occur due to hidden terminals. As discussed in section 4.1.1, the main reason for lost packets in 802.15.4 is because of hidden terminals. We argue there that retransmissions do not manage to solve all collisions due to the limited backoff, which is further verified by these results.

In figure 4.13 we also note that the loss rate for indoor lights is higher for the unicast mode. This is most likely a result of transmissions being sent sequentially. When one transmission is finished, be it successful or not, the next is initiated. Therefore, the transmissions over the channel do not follow a Poisson distribution, but are instead clustered together. This can also be seen as bursty traffic, where the average is low but the instantaneous load can be high. As a result, the system experiences more

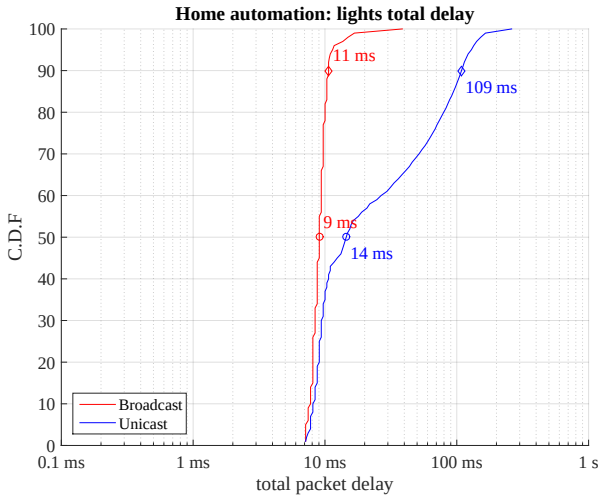


Figure 4.14: Multicast: End to End Latency.

congestion and more errors occur. This behaviour should also appear if some larger packets are transmitted, since fragmentation leads to the same bursty behaviour of the traffic.

The increased reliability with broadcast transmissions holds if the coverage is good, such as the indoor-only scenario used, since the packets are likely to be received correctly on the first attempt if no collision occurs. Had the coverage been worse, the acknowledged unicast transmissions could increase the reliability of transmissions by retransmitting packets that were not received correctly on the first attempt. This would however increase the congestion with more packets being sent over the air. The increased congestion would lead to more collisions due to hidden terminals, which can lead to packet loss. In the current setup, it is difficult to give any clear predictions on how the performance changes in different scenarios since there are many factors in play.

The end to end latency for lamps is shown in figure 4.14. There we note the obvious result that broadcasting results in a lower latency for successful transmissions than unicasting. This is explained simply by noting that in broadcasting mode, only one transmission is sent over the air, regardless of the number of destinations. In unicasting mode, one or several transmissions are sent over the air, depending on the number of destinations. Note that this latency is only measured for the traffic from light switches to lamps and only considers successful transmissions.

Looking at the single hop latency as a function of the traffic load in figure 4.15, we note that the latency increases more for the unicasting mode because of the increased congestion, leading to more back-offs and delays. With more packets being sent over the air (multiple copies of same IPv6 packet), the probability of finding the channel busy is higher.

Figure 4.16 shows the mean power consumption in the default case. Note that the amount of traffic being multicast is low compared to the total amount of traffic and that the broadcasting is performed in the central device which is always active. This is the reason for the negligible difference in energy consumption between the unicast and broadcast modes. Looking at figure 4.17 showing the relative active

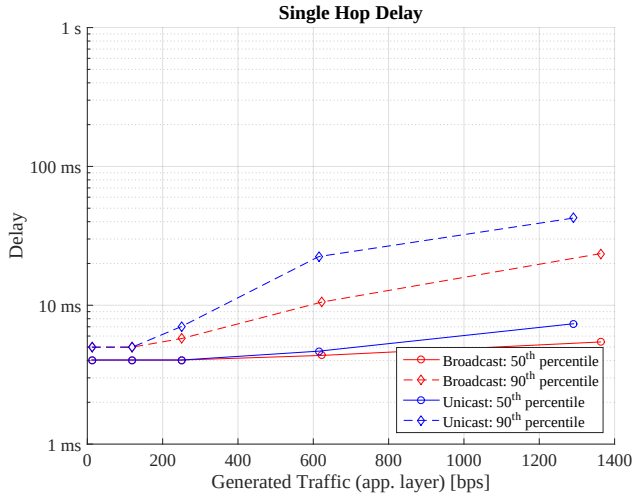


Figure 4.15: Multicast: Single Hop Latency as a function of traffic load.

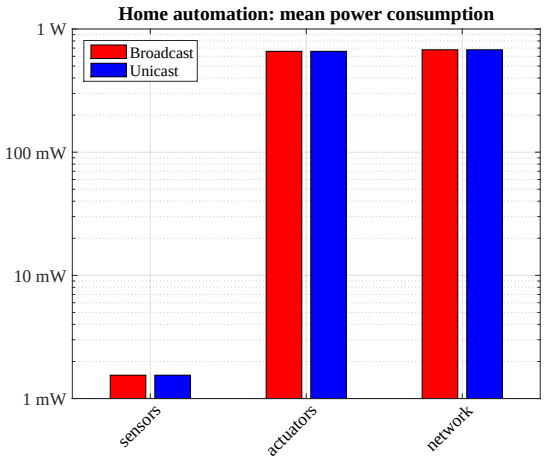


Figure 4.16: Multicast: Mean device power consumption.

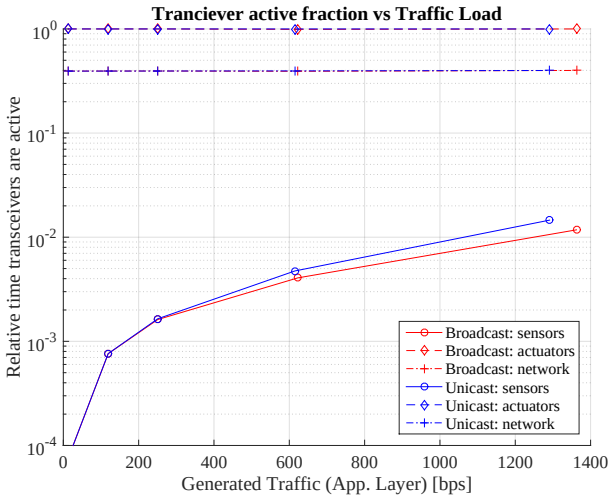


Figure 4.17: Multicast: Device active fraction as function of traffic load.

fraction of transceivers as a function of traffic load, we note a difference between the two modes as the traffic load increases. This is because of the increased number of transmissions over the air introducing additional back-offs and retransmissions in the unicast mode.

In previous works, such as [31], it has been shown that broadcasting is not useful in a generic mesh scenario. When considering routing and a mesh topology, the overhead of having many devices receive the same data outweighs the benefits from only having to transmit each packet once. However, in the scenario considered for this study it is instead shown than there are benefits of using link layer broadcasting for IPv6 multicast packets. It can reduce latency, increase service ratio and in some cases also increase the lifetime expectancy of devices (see figure 4.17 for high traffic load).

The benefits of broadcasting depend on several factors, such as the coverage/fading caracteristics, the number of devices registered to each multicast group and the amount of traffic these groups receive.

There are also issues with broadcasting packets. The obvious issue is that transmissions are not reliable. In a low-power network this is more of an issue than in a wired, reliable network. There is no way of knowing whether a transmission is successful or not and therefore also impossible to perform retransmissions based on errors. A retransmission scheme such as a repetition code can be implemented to increase reliability if the channel quality is bad, but this also increases the energy consumption.

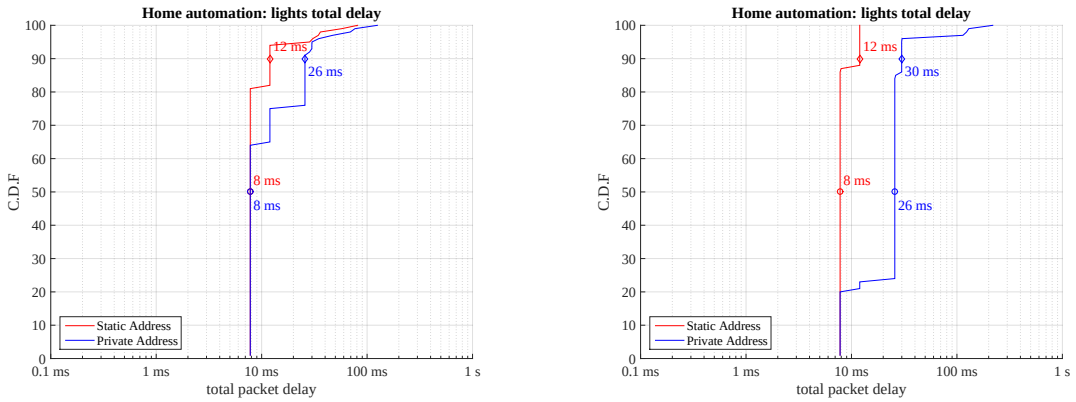
In current standards there is no way of securing broadcast transmissions to provide encryption, authentication or privacy. Therefore, link layer security cannot be used in conjunction with broadcasting on the MAC layer. This might become an issue when security flaws are found in higher layer security implementations.

4.3 Establishing and Managing IPv6 Connections

Setting up an IPv6 network requires devices to acquire IPv6 addresses. The functionality for this is Neighbour Discovery, which defines how devices discover routers, acquire addresses and register these with the router. This process is not energy efficient and relies heavily on broadcasting control messages. This has spurred the work on optimising this for LoWPANs which lead to RFC 6775 - Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs) [19]. The optimisations introduced are discussed in section 2.3.3.

4.3.1 Address Valid Lifetime and Context Management

When registering an IPv6 address, the maximum lifetime that can be chosen is approximately 45 days [19]. The valid lifetime should be chosen in such a way that the device can refresh it at the next wake-up. However, IPv6 is implemented independently of the sleeping schedule of a device which makes it difficult to know whether a device will wake up before the address expires or not.
In BLE, there is a possible issue regarding IPv6 address validity. BLE devices can use resolvable private addresses on the link layer to make it difficult for someone to track the movement of the device based on link layer address, as introduced in section 2.1. The resolvable private address is used in connectable advertisements. The recommendation in the standard is to change this address every 15 minutes [17]. When changing a private address, RFC 7668 requires new non link-local IPv6 addresses to be registered using ICMP messaging, as described in section 2.3.3 of this report and section 3.2.3 of RFC 7668 [5]. This reduces the maximum registration lifetime of an IPv6 address from 45 days to only 15 minutes and increases the frequency of ICMP messages to maintain the IP connections.
This does however not apply to link-local addresses, since they are assumed to be unique and not registered in the neighbour cache. And when changing link layer address, if the BLE connection on link layer is maintained, the connection can be referenced using the host controller interface connection handle [5]. However, if the BLE connection is terminated, issues may arise when reconnecting if the devices have not bonded and exchanged keys for resolving the private link layer addresses or their unique device identities. It is therefore assumed that an exchange of ICMP messages is required when a BLE device changes link layer address.
In order to study the effects of this, a simulation was run for BLE devices with additional ICMP functionality for enabling neighbour discovery messaging. Two cases were considered. One where the devices used public link layer addresses and one where devices used resolvable private link layer addresses. Because of the limited number of simultaneous connections currently supported for BLE , the assumption made was that devices disconnected when neither master nor slave had more data to transmit. Because of this, it was assumed that when using private link layer addresses, neighbour discovery had to be performed every 15 minutes to maintain the IPv6 connection despite the device using a link-local IPv6 address instead of a global unicast address.
Furthermore, it was assumed that devices did not know when the next wake-up



(a) Total Traffic Intensity: 120 bps. **(b)** Total Traffic Intensity: 12 bps.

Figure 4.18: End to end delay for different values of traffic intensity for BLE with and without Resolvable Private Addresses.

event was. Therefore, upon waking up and attempting to transmit data, a sensor device would transmit if its IPv6 address was still valid. Otherwise, if the device had a new link layer address, it first had to register a new IPv6 address before sending the data. Actuator devices initiated the exchange of ICMP messages when the registration lifetime expired.

In the implementation of the proposed scenario, a reachability issue for sleepy BLE devices was observed. When transmitting a router solicitation, the response is not immediate. Therefore the BLE connection was terminated before the router (BLE master) had prepared the router advertisement, since it was prepared in a higher layer implementation. The IPv6 host device (BLE slave) was not reachable when idle and could therefore not be reached with the router advertisement. To avoid this issue, all devices were implemented as non-sleepy devices and were assumed to be always reachable. As a result, no measurements could be performed on energy efficiency and the focus was instead on latency and service ratio. The impact on latency was that between each message exchange on the IPv6 layer, a new BLE connection had to be established, introducing approximately 3 ms of delays to each message. In a real system, it is not efficient to establish connections for each message exchange like this, since it increases the energy consumption. The shortest delay that can be achieved without tearing down and setting up a connection is therefore 7.5 ms, which is the shortest available connection interval. Therefore, the values in figures 4.18a, 4.18b, 4.19 show lower latency values than would be possible in a real implementation. However, the amount of traffic that is affected by latency is not affected.

4.3.2 Simulation Results

Figure 4.18 shows the CDF of the end to end delay for traffic from lamp switches for two different traffic intensity values. On the left hand side we see the delay for the default scenario. It is noted that the latency for some transmissions is higher due to the overhead from registering an IPv6 address and refreshing context entries ($\approx$ 18

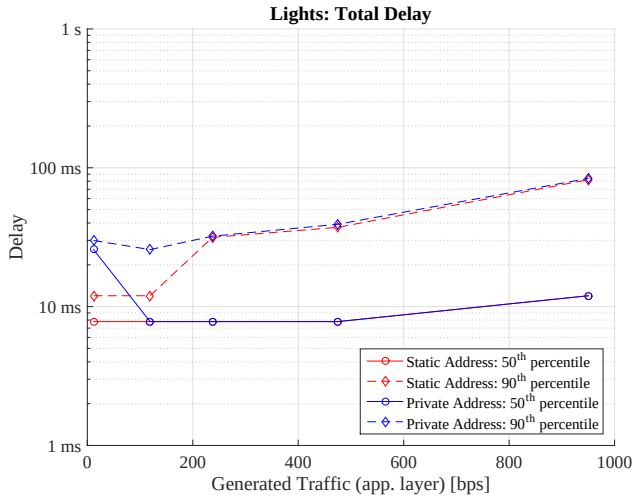


Figure 4.19: End to end delay as a function of traffic intensity for BLE using static or resolvable private link layer addresses.

ms). However, while the increase in latency is high in relation to the latency with static link layer addresses, it is small in absolute value and the resulting latency is still within perceptible values. In the same figure, on the right hand side, we see the same metric, but now for a lower traffic intensity. Here we note a larger impact of the management of IP connections. This is explained by the fact that we have fewer transmissions before the link layer address is changed. In this scenario, the mean packet arrival interval for lamp switches is 3000 seconds (50 minutes), meaning that on average each transmission is queued behind the process of establishing an IP connection in the ICMP entity. This is also reflected in the figure. Since the arrivals are Poisson distributed, we note that some (approximately 25 %) transmissions occur within 15 minutes after the previous one and therefore do not experience an additional delay.

Looking at figure 4.19, we note that the performance in terms of latency converges as the traffic intensity increases. When the traffic intensity increases, the fraction of transmissions that are queued behind the process of establishing an IP connection decreases. From this we conclude that managing IP connections is an issue for networks where devices use private link layer addresses, wish to communicate using global addresses and the traffic intensity is low. The increased number of transmissions will reduce the device lifetime, as can be concluded by looking again at figure 4.7, where it is clear that increasing the frequency of transmissions has a negative impact on the expected device lifetime.

Finally, the service ratio was also studied. There was no difference found between the two scenarios. For low traffic intensity when the ICMP messaging introduced some overhead, the total traffic load was sufficiently low and the small increase did not lower the service ratio. For high traffic intensity, the overhead from ICMP messaging was low in comparison, meaning that it did not affect the service ratio noticeably.

Solicitation Message	Solicited Message(s)	Timeout
ICMPv6 Router Solicitation	ICMPv6 Router Advertisement	3 s
ATT Request	ATT Response	300 ms

Table 4.1: Example of mapping table associating solicitation messages to corresponding solicited messages.

4.3.3 Sensor Reachability

Energy efficiency was not measured in this simulation. During implementation in the simulator, an issue was observed regarding connectivity from the master to the slave. In the simulation scenario, there is a total of 77 devices. This is more than what is currently allowed as simultaneous connections in BLE, which is solved by disconnecting after the devices have exchanged the information they currently have buffered. If the transmission is a router solicitation this means that the host (i.e. BLE slave) will disconnect after finishing this transmission, since the link layer does not have any data in the buffer (it is buffered in the IP layer pending a registered IPv6 address). After disconnecting, it will sleep until it receives a new packet from a higher layer. Meanwhile, the IP layer is awaiting a router advertisement in response to the router solicitation. The router (i.e BLE master) can not reach the IPv6 host and the transaction fails. To circumvent this issue, all devices were implemented as actuators and were therefore always reachable. As a result, the energy consumption was not measured.
To handle this issue in a real scenario, we propose a solution for ensuring slave reachability. The proposed solution is a method for selecting an appropriate sleeping policy of the radio controller in a peripheral device. The sleeping policy is based on the type of message that is generated and the connection status with the peer (central) device.
The message types are given by:

1. *Solicitation message*: the message triggers a response from the receiving device
2. *Solicited message*: the message is generated upon receiving a solicitation
3. *Non-solicited message*: the message does not require a response from the receiving device.

A message in the first category should not belong to any other category, but a message may belong to both the second and third category.
Applied to the problem with ICMP messaging for establishing and maintaining an IPv6 connection, the router solicitation and neighbour solicitation messages belong to the first category. Router advertisement and neighbour advertisement messages belong to categories 2 and 3. This categorisation can be expanded to also include BLE control messages defined in the BLE stack.
Next, a **mapping table** is implemented in the peripheral device. The table maps the solicitation message to the corresponding type(s) of solicitation message(s) and also associates the entry with a timeout associated to the message exchange, as shown in table 4.1
When a message is generated by the peripheral device, the mapping table is checked for an entry corresponding to the message type. If none is found, the device continues as normal without any change in functionality. If a matching entry is found, the

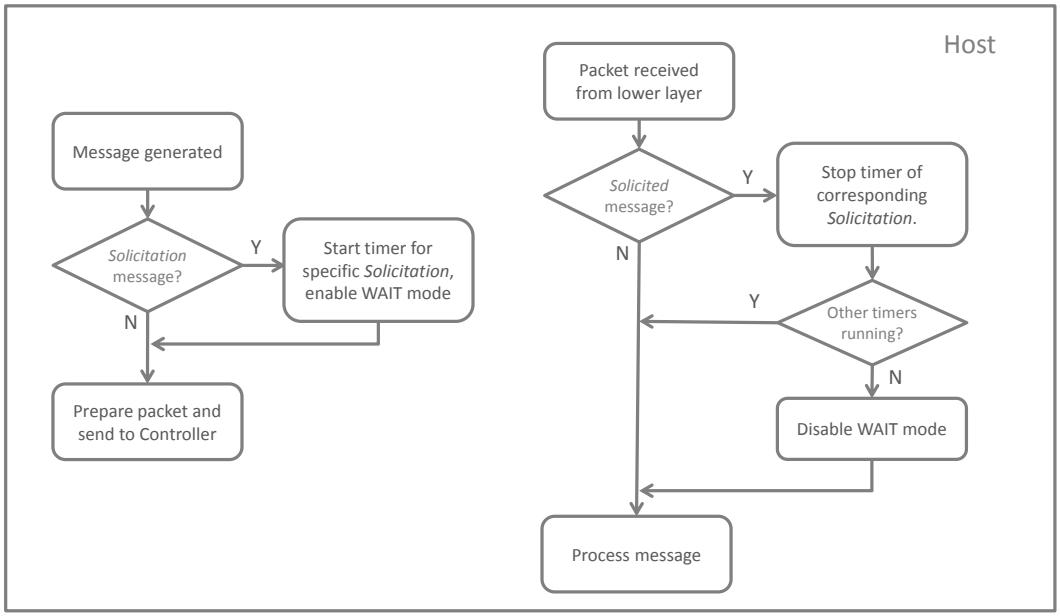


Figure 4.20: Flow chart describing the process for ensuring reachability in the host of a peripheral device.

associated timeout is activated in a timer and a special mode of operation, here called the WAIT mode, is initiated. The packet is then prepared and delivered to the lower layer.

Upon receiving a message, the message type is checked against the solicited message type associated to possible running timers. If a match is found, the expected solicited message has been received and the timer is stopped. The WAIT mode is disabled when there are no more active timers. The flow chart in figure 4.20 describes this process. The WAIT mode ensures that the peripheral device is reachable. In an active BLE connection, a peripheral device is by default in the slave subset of the connected state. As a result, it will go to sleep when it does not have any data to transmit according to the *connSlaveLatency*. The *connSlaveLatency* defines the number of connection events that a device is not required to listen for messages from the master. In the WAIT mode, the device bypasses this setting and listens for messages on every connection event. By doing so, the latency of the message exchange is minimised while also keeping the energy consumption low, since the peripheral device will enable slave latency when the solicited response is received. The process is also described in figure 4.21. If the peer device (BLE master) terminates the connection before the solicited response is received, the peripheral device in WAIT mode can follow several implementation directions:

1. The peripheral device transitions to the *Initiating* state and waits for the peer

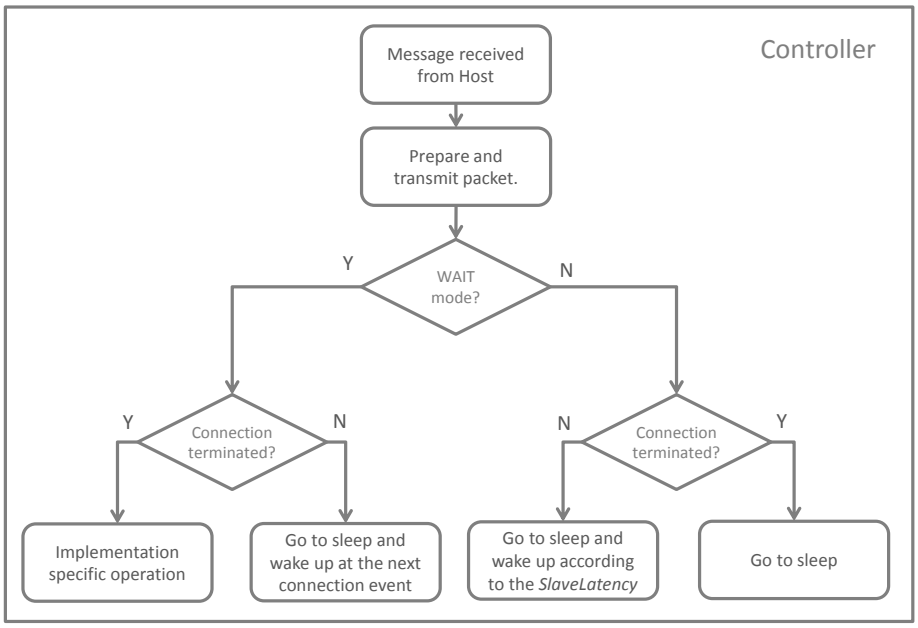


Figure 4.21: Flow chart describing the process for ensuring reachability in radio controller of a peripheral device.

device to send directed connectable advertisements. The slave/master modes will switch upon establishing the next connection.
2. The peripheral device transitions to the *Advertising* state and attempts to re-establish the connection with the peer device. The slave/master modes remain upon establishing the next connection.
3. The peripheral device transitions to the *Scanning* state and listens to the advertisement channels for transmissions from the previous peer device. The peer device sends a *reconnection request*[1] which makes the peripheral device transition to the *Advertising* state and send connectable directed advertisements to the previous peer. This ensures that the slave/master modes are not exchanged and that the connection is not re-established prematurely.

By implementing this method, the BLE connection can be established with the parameters required for continuous operation such as a large *connSlaveLatency* for devices transmitting rarely, but when exchanging messages in a sequence that requires two-way communication, the latency is minimised. In summary, the method optimises the trade off between energy efficiency and latency in an elegant way.

[1]Not currently defined in standards.

5
Conclusions and Discussion

In this chapter, the simulation results are summarised and discussed. The validity of results is discussed and key take home messages emphasised. The identified issues are presented and our proposed solutions to some issues are summarised. Some future research topics are identified and finally we discuss ethical and environmental considerations.

5.1 Simulations

The first simulation, with steady state operation and uncompressed IPv6, provided us with a benchmark between BLE and 802.15.4. It was found that BLE outperforms 802.15.4 in terms of service ratio because of the higher bit rate and dedicated data channels. 802.15.4 has issues with hidden terminals, which does not appear in BLE because of the ALOHA style of access on the advertising channels (technically all terminals are hidden in ALOHA). It should be noted that the values used for the energy detection threshold are the maximum values allowed by the standard. If values from the data sheet for TI CC2630 were used instead, the energy detection would be improved by 15 dB (lowered threshold), which would reduce the amount of hidden terminals. This would improve the reliability for 802.15.4, but not change the other results.
In terms of latency, the performance is similar in the default scenario. BLE experiences an overhead because connections must be established, which is somewhat hidden by the large packets. While 802.15.4 does not require any connections to be established, the bit rate is 4 times lower than for BLE, which results in a similar total latency. When increasing the traffic load, BLE experiences more congestion on the advertising channels, which results in advertisements colliding. This introduces an additional delay of 20-30 ms and as a result the latency for BLE is more sensitive to the traffic load than 802.15.4. It is a known fact that CSMA is more efficient than ALOHA and this is once again verified by the results, as expected.
Finally, the energy efficiency was studied. It was shown that BLE devices use less energy to run the same network setup and that as a result the BLE devices have a longer expected lifetime. This is directly related to the high bit rate which means that BLE devices spend less time in active states than 802.15.4 devices.
When 6LoWPAN header compression was implemented, the size of packets was reduced. As a result, all performance indicators measured were improved. The service ratio was improved mainly for 802.15.4, since the shorter packets reduced the congestion and also reduced the probability of consecutive collisions when a

collision occurred because of a hidden terminal.
The energy consumption decreased similarly between the technologies because of reduced time in active states. The total decrease of number of bits over the air was lower for 802.15.4 since BLE also benefited from reduced fragmentation, but since the bit rate is lower for 802.15.4, each elided bit improved the energy efficiency more than for BLE.
IPv6 multicast addresses were studied next to identify the effects on the performance. Because BLE did not support broadcast of IPv6 messages, only 802.15.4 was considered. It was, in contrast to previous works, found that the service ratio was improved by broadcasting messages with IPv6 multicast destinations, despite the inability to acknowledge broadcast transmissions. The positive effects from reduced congestion outweighed the negative effects from not being able to perform retransmissions. The conclusions from this is that the retransmission process in 802.15.4 does not provide good protection against collisions from hidden terminals, and that the errors experienced when unicasting all transmissions were of types that retransmissions would not help against.
The latency was improved by broadcasting, because no sequential transmissions were performed and because the number of transmissions was reduced, leading to less congestion and less backoffs.
Another effect of the reduced number of transmissions when using broadcasting functionality was increased expected device lifetime. This was simply an effect of devices spending less time in active states.
In the final simulation the effects of establishing and managing IPv6 connections were studied by considering BLE devices using public or resolvable private link layer addresses. No considerable difference was found in the service ratio, as when the traffic load was low and the overhead from ICMP messaging high, there was little or no congestion and when the traffic load was high, the overhead from ICMP messaging was low.
The latency was affected by using private addresses in the default scenario, because the overhead from maintaining the IPv6 connection in terms of messages sent was considerable. In the default scenario, approximately 15-20 % of transmissions from the light switches (i.e. application layer) were affected by messages exchanged in the neighbour detection process. In the entire network, other sensor devices transmitted more frequently and the overhead for those devices was less prominent. Therefore the composite effect on the total traffic was negligible. When reducing the traffic intensity by a factor 10, a larger fraction of transmissions from lamp switches (approximately 60 %) experienced an increased delay because of ICMP messages. When increasing the traffic intensity, the performance in terms of latency converged between networks where devices used public or resolvable private link layer addresses.

5.2 Identified Issues and Proposed Solutions

One purpose of the study was to perform a benchmark the performance between BLE and 802.15.4 under IPv6 traffic. Another purpose was to identify areas with issues and to propose solutions to these issues. This section summarises the work on the second purpose.

In the first simulation, with no IPv6 functionality or 6LoWPAN header compression, the main issue identified was the large overhead introduced by the IPv6 and higher layers. A total of 89 bytes were added to the application layer data before the technology specific overhead below the IPv6 layer. This resulted in a very low fraction of application data being carried in every data packet and is not efficient. The solution for this issue is today considered to be 6LoWPAN header compression, which can reduce the IPv6 header size from 40 down to 2 bytes and also provides compression of UDP headers from 8 to 2 bytes.

This was implemented and it was noted that it did in fact improve the performance. However, there are issues regarding the maintenance of the shared context required for stateful header compression. One issue is that when updating or changing a compression entry, the entry becomes stale for a long time (at least 300 seconds). It is not possible to update context entries on the fly. Another possible issue is that the context must be distributed across the network, there is no way of distributing only parts of the context that will be used. Only the devices communicating need to have the same entry in the context, but there is today no standardised way of distributing only parts of the context. So, if a network consists of many devices and each device registers a global unicast address with the router to be able to use this when communicating with a note outside the subnet (for instance a remote server), every device in the network will receive this context entry. This may result in memory issues in constrained devices and also energy issues since this information must be transmitted and received. We have not considered any solutions to these issues as the handling of 6LoWPAN context entries is implementation specific and out of scope in the standards.

Instead we focused on an area with improvement potential, which is how to handle IPv6 multicast addresses. The issue is that there is no standardised way of handling IPv6 multicast destinations on the link layers of the underlying technology. We performed a study on the two possible methodologies to provide suggestions for future directions of standardisation.

It was found that being able to broadcast large transmissions can be beneficial for the service ratio, latency and energy consumption. However, it is acknowledged that the benefits are dependent on the scenario. Since broadcast transmissions generally cannot be acknowledged, there is no reliability in such transmissions. If the channel quality is bad, then performance might be better when not broadcasting transmissions with IPv6 multicast destinations. If the multicast groups are small, then the improvements in terms of energy efficiency and latency will be smaller when broadcasting than if the multicast groups are large.

When broadcasting transmissions, there is no method for providing link layer security today. Therefore, if using broadcast functionality to serve higher layer data transmissions, one must rely on higher layer security.

Based on this discussion, there is no clear method to use for transmitting packets with IPv6 multicast destinations in an optimal manner. However, we do support the development of increased broadcast capacity presented for Bluetooth 5.0, since broadcasting can be useful [32]. These conclusions also give some guidelines for anyone attempting to implement an algorithm for automatically deciding when to use either broadcasting or sequential unicasting.

The final issue that was identified was regarding managing IPv6 connections. It was found that the ICMP messaging required for establishing and managing IPv6 connections does not reduce the performance in terms of service ratio or latency much for BLE devices. However, an issue was found during the implementation. It was noted that because a connection event is closed when neither device has more data to transmit, a response to a message from a higher layer than the link layer would have to be sent in the next connection event. It was also noted that devices might terminate the link layer connection immediately after exchanging all available data, effectively not being reachable unless transmitting something.
The issue with this is not only if devices disconnect and thereby fail to connect, but since the response comes in the next connection event the latency for a transmission queued behind the establishment of an IPv6 connection will be large. In case the initiating device implements slave latency, the latency will be even higher. Therefore we proposed a method for ensuring reachability of BLE slave devices based on higher layer messaging. By implementing our solution, the slave device will be reachable even if the master disconnects, and if the BLE connection is active it will not apply slave latency. This ensures that BLE devices can connect to IPv6 networks and minimises the latency when doing so. It also keeps the energy consumption low.
Another suggested improvement is the possibility to entirely avoid having to perform additional messaging when changing a BLE link layer address to a new resolvable private address. This can be done by using the identity address that is returned after resolving a private address. This identity address is static and can be used as reference to a device that has previously bonded. The details of such a solution are not presented in this thesis.

5.3 Proposed Future Research

Throughout this study, several simplifying assumptions have been made. Because of these assumptions, the results from the simulations are not entirely truthful compared to a real setup. Hardware limitations such as buffer and memory sizes, processing of packets and similar has not been considered and might affect the results in a real network.
Furthermore, we have assumed that the network does not experience any interference from other technologies. Since both BLE and 802.15.4 operate in ISM bands, this assumption does truly not hold in a real scenario. Based on this, an interesting future study would be to set up a network with existing hardware for both BLE and 802.15.4 and measure the performance. The problem with such a study is that the improvement suggestions offered here cannot be implemented easily.
In this study, we have not considered any link layer security, as it is assumed that the security requirements are covered by DTLS. To further improve the benchmark between BLE and 802.15.4, a study where link layer security is considered would be interesting as well as very useful to identify security issues.
When this study was performed, WiFi HaLow was not yet released. A broader benchmark also including more radio technologies designed for the IoT gives more insight to the available choices and would be useful for future improvements to standards.

5.4 Ethical and Environmental Considerations

When performing scientific work it is always a good idea to take a minute and think about the consequences of the work one is performing. So also with this study.
This study further enables connecting a wide range of devices to the internet by proposing solutions to current issues. By doing so, this study essentially creates security and privacy issues that must be considered.
A debatable assumption is that there will always be security flaws in systems created by people. Assuming this is the case, it is debatable whether enabling people to connect their homes with devices such as alarm systems, stoves, ovens, locks and more to the internet is a good idea. If people with malicious intent manage to compromise a home automation system connected to the internet, great damage can be done. It is therefore of utmost importance that the systems used for this kind of applications are designed with high requirements on security and are kept up to date with continuous updates.
This is one of the reasons why security considerations have not been considered in this study. Because the author does not have any education or experience in security it would do more harm than good to study the security of the solutions proposed. Instead it is urged that people with such experience are included in the standardisation and implementation of new solutions.
One security issue that is created when enabling IPv6 connectivity for battery powered devices is the opportunity for an attacker to attack and kill nodes by overloading them with data. Since devices are battery powered and we have seen in the results that transmissions deplete the battery quickly this is a simple way of disrupting a network. For a network connected to power outlets, a so call Denial-of-Service attack will render the network unusable for the duration of the attack. But the same attack targeted at a battery-powered network could effectively disable the network by depleting the batteries of the devices in it. Earlier, networks for IoT were limited by the radio coverage of the devices in them, but connecting these networks to the internet removes this barrier and an attack can originate from anywhere.
A suggestion that is somewhat out of scope of this study is that everyone should be made aware of this kind of issues before experiencing it the wrong way. Because even if the systems are designed with security in mind, in many cases the user does not identify potential security risks and thereby create openings for people with malicious intent. One very simple example is the task of choosing a strong password which is exchanged on a regular basis and not used for multiple services.
There are also environmental considerations to be made with the emergence of IoT. While the benefits from automation and optimisation of processes is desirable, there are some parts of IoT that are not so clear from an environmental point of view. One example is the case of light switches. It is somewhat difficult to see the use case of connecting a stationary light switch controlling a stationary lamp to the internet. The benefit is that no cable is required to the lamp, but since the lamp itself is mains-powered, it still requires a cable. In addition, the light switch requires a battery, instead of being a passive switch. A light switch installed 50 years ago still functions to this day, but considering the rapid evolution of electronics hardware such as processors it is questionable if a light switch built with a wireless transceiver

will ever achieve the same lifetime as a regular switch.
While the increased use cases, such as making sure all lights are turned off using the internet, may justify connecting the light itself to the internet, also connecting the switch seems somewhat unnecessary. If the switch must be exchanged every 10 years instead of every 50, more material will have to be produce to meet that demand.

Bibliography

[1] A. Williams. IoT gets smart with Bluetooth. *Electronics Weekly*, (2642):12, 2015. ISSN 00135224.

[2] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460, RFC Editor, December 1998. `http://www.rfc-editor.org/rfc/rfc2460.txt`.

[3] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944, RFC Editor, September 2007. `http://www.rfc-editor.org/rfc/rfc4944.txt`.

[4] J. Hui and P. Thubert. Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks. RFC 6282, RFC Editor, September 2011. `http://www.rfc-editor.org/rfc/rfc6282.txt`.

[5] J. Nieminen, T. Savolainen, M. Isomaki, B. Patil, Z. Shelby, and C. Gomez. IPv6 over BLUETOOTH(R) Low Energy. RFC 7668, RFC Editor, October 2015. `http://www.rfc-editor.org/rfc/rfc7668.txt`.

[6] HighBeam Research. ABI Research Predicts Wireless Connectivity Gap to Widen as Bluetooth Enabled Device Shipments Reach 19 Billion over the Next Five Years. *Telecommunications Weekly*, page 45, Dec 30 2015.

[7] A. Willig. Recent and Emerging Topics in Wireless Industrial Communications: A Selection. *IEEE Transactions on Industrial Informatics*, 4(2):102–124, May 2008.

[8] P. Di Marco, C. Fischione, F. Santucci, and K. H. Johansson. Modeling IEEE 802.15.4 Networks Over Fading Channels. *IEEE Transactions on Wireless Communications*, 13(10):5366–5381, Oct 2014.

[9] K. Mikhaylov, N. Plevritakis, and J. Tervonen. Performance Analysis and Comparison of Bluetooth Low Energy with IEEE 802.15.4 and SimpliciTI. *Journal of Sensor and Actuator Networks*, 2(3):589–613, 2013.

[10] M. Petrova, J. Riihijarvi, P. Mahonen, and S. Labella. Performance study of IEEE 802.15.4 using measurements and simulations. In *IEEE Wireless Communications and Networking Conference, 2006. WCNC 2006.*, volume 1, pages 487–492, April 2006. doi: 10.1109/WCNC.2006.1683512.

[11] H. Wang, M. Xi, J. Liu, and C. Chen. Transmitting IPv6 packets over Bluetooth low energy based on BlueZ. In *Advanced Communication Technology (ICACT), 2013 15th International Conference on*, pages 72–77, Jan 2013.

[12] J. Nieminen, C. Gomez, M. Isomaki, T. Savolainen, B. Patil, Z. Shelby, M. Xi, and J. Oller. Networking Solutions for Connecting Bluetooth Low Energy Enabled Machines to the Internet of Things. *IEEE Network*, pages 83–90, 11 2014.

[13] E. Casilari, J. M. Cano-García, and G. Campos-Garrido. Modeling of Current Consumption in 802.15.4/ZigBee Sensor Motes. *Sensors*, 10(6):5443–5468, 2010.

[14] W. Du, D. Navarro, and F. Mieyeville. Performance evaluation of IEEE 802.15.4 sensor networks in industrial applications. *International Journal of Communication Systems*, 28(10):1657–1674, 2015.

[15] M. Siekkinen, M. Hiienkari, J. Nurminen, and J. Nieminen. How low energy is Bluetooth Low Energy? Comparative measurements with ZigBee/802.15.4. In *Wireless Communications and Networking Conference Workshops (WCNCW), 2012 IEEE*, pages 232–237. IEEE, April 2012. ISBN 9781467306812;1467306819;.

[16] C. J. Hansen. Internetworking with Bluetooth Low Energy. *GetMobile: Mobile Comp. and Comm.*, 19(2):34–38, August 2015. ISSN 2375-0529. doi: 10.1145/2817761.2817774.

[17] *Bluetooth® Core Specification 4.1*. Bluetooth SIG, 2014. [Online]. Available: `https://www.bluetooth.com/specifications/adopted-specifications`.

[18] *IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)*. IEEE 802.15.4-2011, 2011. [Online]. Available: `https://standards.ieee.org/about/get/802/802.15.html`.

[19] Z. Shelby, S. Chakrabarti, E. Nordmark, and C. Bormann. Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs). RFC 6775, RFC Editor, November 2012. `http://www.rfc-editor.org/rfc/rfc6775.txt`.

[20] R. Graziani. *IPv6 Fundamentals: A Straightforward Approach to Understanding IPv6*. Cisco Press, 800 East 96th Street, Indianapolis, IN 46240 USA, 2013. ISBN 9781587143137.

[21] S. Ghnimi, A. Rajhi, and A. Gharsallah. BER perfermance of GMSK modulation under radio mobile propagation environments. In *2011 11th Mediterranean Microwave Symposium (MMS)*, pages 305–308, Sept 2011. doi: 10.1109/MMS.2011.6068586.

[22] R. Heydon. *Bluetooth Low Energy: The Developer's Handbook*. Pearson Education, Inc., One Lake Street, Upper Saddle River, New Jersey 07458, 2013. ISBN 9780132888363.

[23] S. Fang, S. M. Berber, and A. K. Swain. Energy consumption evaluations of cluster-based sensor nodes with IEEE 802.15.4 transceiver in flat Rayleigh fading channel. In *Wireless Communications Signal Processing, 2009. WCSP 2009. International Conference on*, pages 1–5, Nov 2009. doi: 10.1109/WCSP.2009.5371679.

[24] B. Haberman and D. Thaler. Unicast-Prefix-based IPv6 Multicast Addresses. RFC 3306, RFC Editor, August 2002. `http://www.rfc-editor.org/rfc/rfc3306.txt`.

[25] R. Hinden and S. Deering. IP Version 6 Addressing Architecture. RFC 4291, RFC Editor, February 2006. `http://www.rfc-editor.org/rfc/rfc4291.txt`.

[26] T. Narten, E. Nordmark, W. Simpson, and H. Soliman. Neighbor Discovery for IP version 6 (IPv6). RFC 4861, RFC Editor, September 2007. `http://www.rfc-editor.org/rfc/rfc4861.txt`.

[27] J. Postel. Internet Control Message Protocol. STD 5, RFC Editor, September 1981. `http://www.rfc-editor.org/rfc/rfc792.txt`.

[28] *CC2630 SimpleLink™ 6LoWPAN, ZigBee® Wireless MCU*. Texas Instruments, 2015. [Online.] Available: `http://www.ti.com/product/CC2630`.

[29] *CC2640 SimpleLink™ Bluetooth® Smart Wireless MCU*. Texas Instruments, 2015. [Online.] Available: `http://www.ti.com/product/CC2640`.

[30] N. Kushalnagar, G. Montenegro, and C. Schumacher. IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals. RFC 4919, RFC Editor, August 2007. `http://www.rfc-editor.org/rfc/rfc4919.txt`.

[31] X. Wang. Multicast for 6LoWPAN Wireless Sensor Networks. *IEEE Sensors Journal*, 15(5):3076–3083, 2015.

[32] *Bluetooth 5 Coming Soon*. Bluetooth SIG, Jun 2016. [Online.] Available: `https://www.bluetooth.com/news/pressreleases/2016/06/16/-bluetooth5-quadruples-rangedoubles-speedincreases-data-broadcasting-capacity-by-800`, Accessed: 11 July 2016.

A

CSMA-CA in IEEE 802.15.4

The CSMA functionality in the non beacon-enabled mode for 802.15.4 is described by the flowchart in figure A.1. The process begins with resetting the counter for the number of failed channel accesses (NB) and the backoff exponent used for randomly generating the backoff duration (BE). Next, a random backoff is generated as a random number between 0 and $2^{BE} - 1$. This is the number of unit backoff periods to remain idle before sensing the channel. A unit backoff period is equivalent to 20 symbol periods or 320 μs. With a default value for *macMinBE* of 3, the backoff is between 0 and 140 symbol periods, which in time units is between 0 and 2.24 ms in increments of 320 μs.
After the random backoff, the channel sensing procedure is performed. During 8 symbol periods (128 μs), the receiver is turned on and the received power is measured. If the average power received during the channel sensing is below the threshold (-75 dBm in simulations), the channel is considered clear and the CSMA process exits with a successful status. The transceiver then turns around from receiving to transmitting (12 symbol periods or 192 μs) and the packet is transmitted.
If the power received is above the threshold, the channel is assumed to be busy. The counter for the number of failed channel accesses (NB) is incremented by one and the backoff exponent is also incremented by one. The backoff exponent is limited by *macMaxBE*, which in the simulations was set to 5. If the number of failed channel accesses is larger than *macMaxCSMABackoffs* (default value 4), the process exits because of a channel access failure. If not, then a new random backoff period is generated and the process is run again.

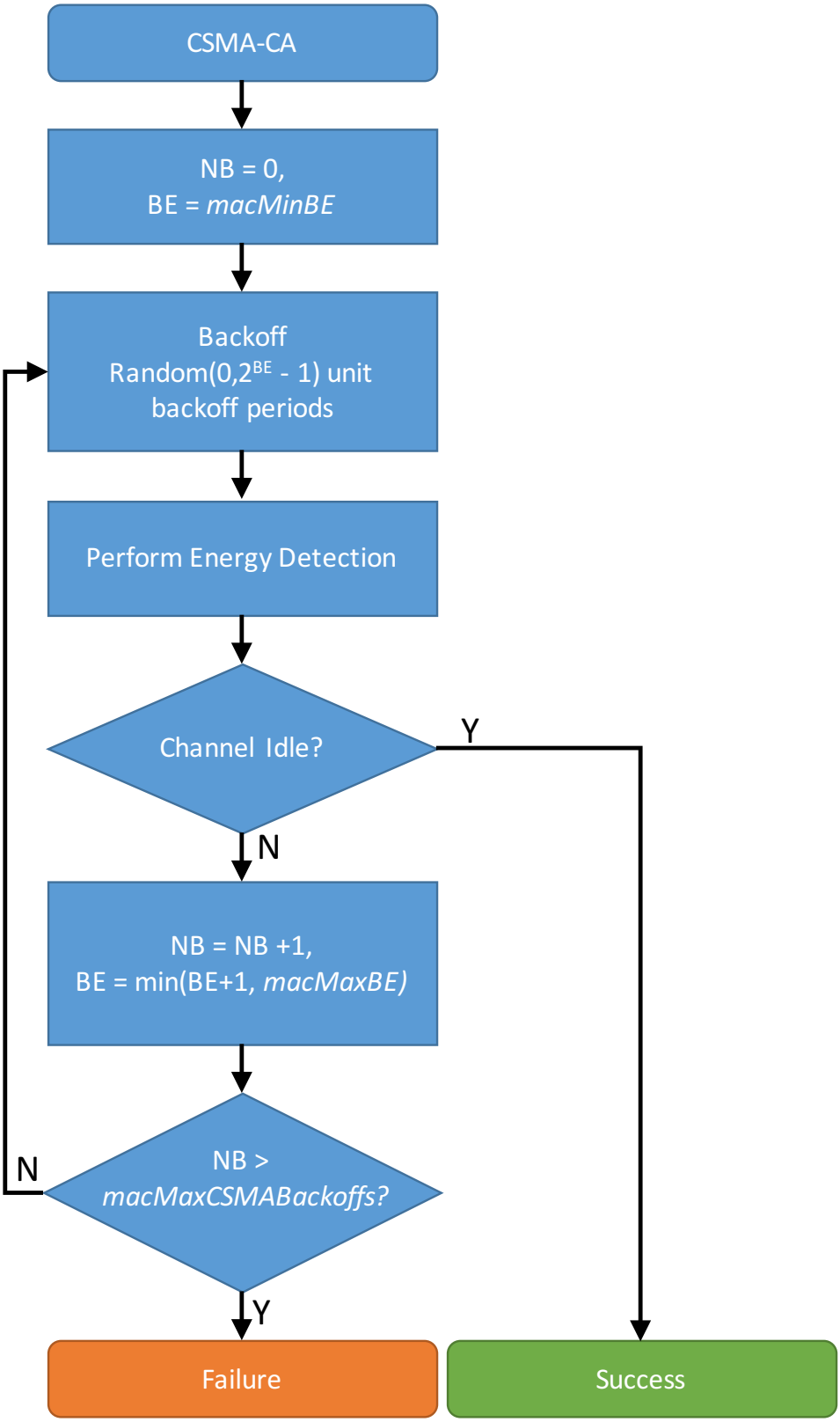


Figure A.1: Flowchart of the unslotted CSMA used in 802.15.4.

B

Simulation Scenario

Details of the simulation scenario used are presented here. The deployment of devices and the floor plan of the assumed apartment/house is shown in figure B.1. In table B.1, the devices are described with packet size and mean packet arrival interval. Note that the asset tags are not shown in the figure. For the sensitivity analysis, the traffic load was varied. The values in table B.1 are valid for the default case. For low traffic, the arrival rates were 1/10 of the default values. For the 3 cases with higher traffic, the arrival rates were doubled for each iteration. Thus, the normalised traffic load in the different cases were 1/10, 1, 2, 4, 8.

Icon	Device Type	# of Devices	Payload Size	Packet Interval
	Lamp	20	1 B	5 min
	Switch	18	1 B	5 min
	Window Sensor	21	1 B	5 s
	Temperature Sensor	8	4 B	1 min
	Asset Tag[1]	10	1 B	1 s
	Gateway	1	-	-

Table B.1: Description of the simulated devices used in the study.

[1]Not included in figure - uniformly distributed across the simulated area.

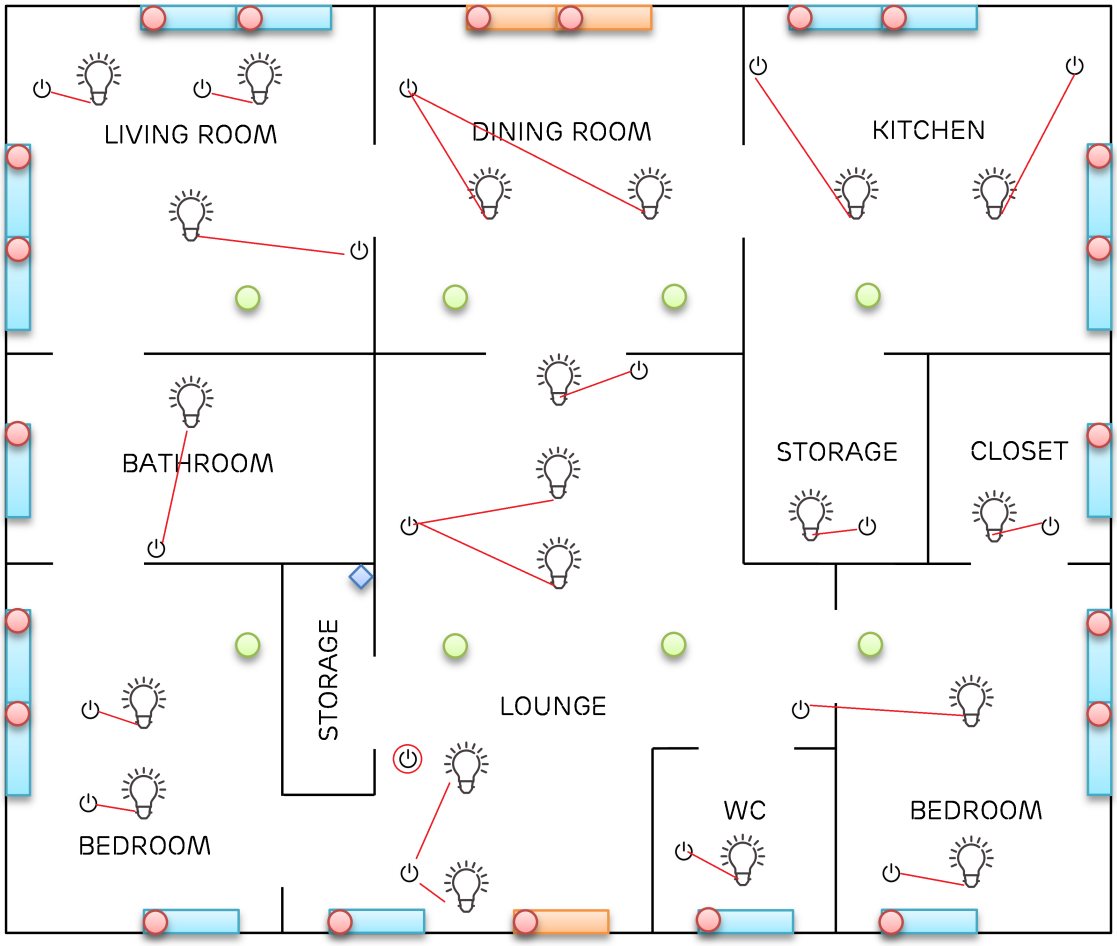


Figure B.1: Deployment of devices for the simulation scenario used. Red lines represent logical connections between lamps and lamp switches. All communication goes through the central device, meaning that a command from a lamp switch is sent to the gateway where it is forwarded to the corresponding lamp(s) according to the logical connections.

C

Parameter settings for 6LoWPAN Header Compression

Here we describe the parameter settings for 6LoWPAN Header Compression in more detail. The header compression fields and corresponding number of additional bits carried in-line are given below.

TF Traffic class and flow label.
- 0: 4 bytes carried in-line.
- 1: 3 bytes carried in-line.
- 2: 1 byte carried in-line.
- 3: Traffic class and flow label are elided.

NH Next Header
- 0: Full 8 bits carried in-line
- 1: Next header field compressed with 6LoWPAN next header compression

HLIM Hop Limit
- 0: Full 8 bits carried in-line
- 1: Hop limit is 1 (0 bits in-line)
- 2: Hop limit is 64 (0 bits in-line)
- 3: Hop limit is 255 (0 bits in-line)

CID Context Identifier Extension
- 0: Default (0) or no context used.
- 1: 8 bit context identifier extension carried in-line.

SAC Source Address Compression, specifies whether to use context based or stateless compression for the source address.
- 0: Stateless compression used.
- 1: Context based compression used.

SAM Source Address Mode, specifies the compression mode used.

SAC = 0 :
- 0: Full address carried in-line.
- 1: First 64 bits are link-local prefix. Remaining 64 bits carried in-line.
- 2: First 64 bits are link-local prefix. Remaining bits are 0000:00FF:FE00:XXXX. X's carried in-line.
- 3: Full address elided. Prefix is link-local. Remaining bits calculated from encapsulating header.

SAC = 1 :
- 0: Full address elided. The address is ::, the unspecified address.

1: 64 bits carried in-line. Remaining bits covered by context and encapsulating header.
2: 16 bits carried in-line. Remaining bits covered by context and encapsulating header.
3: Full address elided. Address is recovered using context and encapsulating header.

M Multicast, specifies whether the destination is a multicast address or not.
0: Destination address is not a multicast address.
1: Destination address is a multicast address.

DAC Destination Address Compression, as SAC, but for destination address.
0: Stateless compression used.
1: Context based compression used.

DAM Destination Address Mode, like SAM, but for destination address.

DAC = 0, M = 0 :
0: Full address carried in-line.
1: First 64 bits are link-local prefix. Remaining 64 bits carried in-line.
2: First 64 bits are link-local prefix. Remaining bits are 0000:00FF:FE00:XXXX, X's carried in-line.
3: Full address elided. Prefix is link-local. Remaining bits calculated from encapsulating header.

DAC = 1, M = 0 :
0: Reserved.
1: 64 bits carried in-line. Remaining bits covered by context.
2: 16 bits carried in-line. Remaining bits covered by context.
3: Full address elided. Address is recovered using context and encapsulating header.

DAC = 0, M = 1 :
0: Full address carried in-line.
1: Address is FFXX::00XX:XXXX:XXXX, 48 bits in-line.
2: Address is FFXX::00XX:XXXX, 32 bits in-line.
3: Address is FF02::00XX, 8 bits in-line.

DAC = 1, M = 1 :
0: Address is FFXX:XXLL:PPPP:PPPP:PPPP:PPPP:XXXX:XXXX, 48 bits in-line. P and L derived from context information [4, 24].
1: Reserved.
2: Reserved.
3: Reserved.

The parameter values used in the simulations are given in table C.1 below. Some header field values were exchanged between simulations, the corresponding fields have multiple values in the table.

Field	Value	Comments
Asset Tags		
TF	3	Traffic Class and Flow Label not used.
NH	1	UDP header compression used.
HLIM	3	Single-hop transmissions.
C	1	Assume context identifier is required.
SAC	1	Assume context-based compression.
SAM	3	Stateless autoconfiguration used.
M	0	
DAC	1	Assume context-based compression.
DAM	3	Stateless autoconfiguration used.
C (UDP)	1	Checksum elided.
P (UDP)	3	Full port compression.
Window Sensors		
TF	3	Traffic Class and Flow Label not used.
NH	1	UDP header compression used.
HLIM	3	Single-hop transmissions.
C	1	Assume context identifier is required.
SAC	1	Assume context-based compression.
SAM	3	Stateless autoconfiguration used.
M	0	
DAC	1	Assume context-based compression.
DAM	3	Stateless autoconfiguration used.
C (UDP)	1	Checksum elided.
P (UDP)	3	Full port compression.
Temperature Sensors		
TF	3	Traffic Class and Flow Label not used.
NH	1	UDP header compression used.
HLIM	3	Single-hop transmissions.
C	1	Assume context identifier is required.
SAC	1	Assume context-based compression.
SAM	3	Stateless autoconfiguration used.
M	0	
DAC	1	Assume context-based compression.
DAM	3	Stateless autoconfiguration used.
C (UDP)	1	Checksum elided.
P (UDP)	3	Full port compression.
Light Switches		
TF	3	Traffic Class and Flow Label not used.
NH	1	UDP header compression used.
HLIM	3/0	Hop limit used for multicast.
C	1	Assume context identifier is required.
SAC	1	Assume context-based compression.
SAM	3	Stateless autoconfiguration used.

M	0/1	1 for multicast simulation.
DAC	1	Assume context-based compression.
DAM	3/0	0 for multicast simulation.
C (UDP)	1	Checksum elided.
P (UDP)	3	Full port compression.
Lamps		
TF	3	Traffic Class and Flow Label not used.
NH	1	UDP header compression used.
HLIM	3	Single-hop transmissions.
C	0	Stateless compression.
SAC	0	Stateless compression.
SAM	3	Stateless autoconfiguration used.
M	0	
DAC	0	Stateless compression.
DAM	3	Stateless autoconfiguration used.
C (UDP)	1	Checksum elided.
P (UDP)	3	Full port compression.
Central Device		
TF	3	Traffic Class and Flow Label not used.
NH	1	UDP header compression used.
HLIM	3	Single-hop transmissions.
C	1	Assume context identifier is required.
SAC	1	Assume context-based compression.
SAM	3	Stateless autoconfiguration used.
M	0/1	
DAC	1	Assume context-based compression.
DAM	3/0	Stateless autoconfiguration used.
C (UDP)	1	Checksum elided.
P (UDP)	3	Full port compression.

Table C.1: Parameter settings used for 6LoWPAN header compression.