Hasan Qaq

Performance evaluation of Bluetooth mesh networking in building environment

June 2022

Master's thesis
2022
Hasan Qaq

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Information Security and Communication Technology

NTNU
Norwegian University of Science and Technology

NTNU
Norwegian University of
Science and Technology

Performance evaluation of Bluetooth mesh networking in building environment

Hasan Qaq

Communication technology and digital security
Submission date: June 2022
Supervisor: Yuming Jiang
Co-supervisor: Omkar Kulkarni

Norwegian University of Science and Technology
Department of Information Security and Communication Technology

NTNU – Trondheim
Norwegian University of
Science and Technology

Performance evaluation of Bluetooth mesh networking in building environment

Hasan Qaq

Submission date: June 2022
Supervisor: Yuming Jiang, NTNU
Co-supervisor: Omkar Kulkarni, Nordic Semiconductor ASA

Norwegian University of Science and Technology
Department of Information Security and Communication Technology

Title: Performance evaluation of Bluetooth mesh networking in building environment
Student: Hasan Qaq

Problem description:

Data gathering networks can be highly valuable to commercial buildings and office spaces for monitoring and reacting to the physical characteristics of buildings' environment. For instance, a wireless sensor network may be used to monitor the building environment and control heating ventilation and air conditioning (HVAC) systems, and lighting systems to optimize the energy consumption of the building and to improve the comfort level of building users. Such networks have a multitude of sensors that collect and measure various quantities and send this information to a gateway for further processing, e.g. sending the data packets over the Internet to a system that controls for example the heating or lighting system in the building. Bluetooth mesh is an upcoming short-range wireless networking standard that provides an infrastructure to build such wireless sensor networks.

The objective of the thesis project is to conduct an exploration of how the Bluetooth mesh networking technology performs for a data collecting sensor network in order to assess the reliability of the network. Specifically, the project has the following major objectives:

1. Conducting different tests on a Bluetooth mesh test network available at Nordic Semiconductor, including data gathering and analyzing.
2. Investigating and proposing potential improvements, e.g. location and configuration of sensor nodes, to achieve improved reliability and data throughput. This includes theoretically analyzing the proposals and estimating the impact, verifying the proposals with execution on the test setup, and gathering and analyzing the test results.
3. Conducting a comparative study of existing wireless technologies that can be used for building data-gathering wireless sensor networks, based on which, discuss pros and cons of using Bluetooth mesh.

Date approved:	2022-01-21
Responsible professor:	Yuming Jiang, NTNU
Supervisor(s):	Omkar Kulkarni, Nordic Semiconductor ASA

Abstract

Bluetooth mesh is a recent addition to the IoT connectivity landscape. It provides a simple and efficient wireless networking solution. Bluetooth mesh provides mesh connectivity to many nodes without any coordination.

This thesis evaluates the performance of Bluetooth mesh technology in IoT data-gathering networks in an office environment. Through carrying out multiple tests on an actual network, the reliability of this network, in terms of packet loss i.e. packet delivery ratio, is evaluated. The reliability is evaluated during the analysis of the effect of some protocol-related and protocol-non-related parameters. The protocol-related parameters include the Publish Retransmit Count (PRC), Network Transmit Count (NTC) and Relay Retransmit Count (RRC). The protocol-non-related parameters include the packet sending interval, the packet sending randomization, and the packet payload redundancy.

Results show that when using unsegmented packets, a high degree of reliability is achieved when the PRC, NTC and RRC values are tuned, and when the relay locations and the non-protocol-related parameters in the network are chosen properly.

Sammendrag

Bluetooth mesh er et tilskudd til mesh teknologien. Bluetooth mesh gir en enkel, effektiv og trådløs nettverksløsning der mange nettverksnoder kan kommunisere uten noen overordnet styringsmekanisme.

Denne oppgaven evaluerer ytelsen til Bluetooth mesh-teknologien i IoT-datainnsamlingsnettverk i et kontormiljø. Gjennom testing på et fullskallert datainnsamlingsnettverk blir påliteligheten til nettverket med, fokus på pakketap, evaluert ved å analyse effekten av et utvalg av protokollrelaterte og ikke-protokollrelaterte parametere. De protokollrelaterte parametrene inkluderer Publish Retransmit Count(PRC), the Network Transmit Count (NTC) and the relay Retransmit Count (RRC). De ikke-protokoll-relaterte parameterne inkluderer pakkesendingsintervall, bruk av ekstra tilfeldig tid mellom sendinger og nyttelast-reliabilitet.

Resultatene viser, ved bruk av usegmenterte pakker, en høy grad av nettverkspålitelighet ved bruk av regulerte verdier av PRC, NTC og RRC, protokollrelaterte parameterne, samt nøye utvalgte relay-noder for nettverket.

Preface

This master's thesis is my final assignment for Master of Science degree in Communication Technology at the Department of Information Security and Communication Technology at the Norwegian University of Science and Technology (NTNU). The project is conducted throughout the spring semester 2022 as an in-depth evolution of the specialization project carried out one semester earlier.

The topic of the research is suggested by Nordic Semiconductor ASA, a Norwegian fabless semiconductor company specializing in wireless communication technology that powers the IoT, and thus the research itself is heavily practice-oriented. The work is supervised by prof. Yuming Jiang from NTNU and co-supervised by Omkar Kulkarni from Nordic Semiconductor.

Acknowledgements

I would like to thank the following people, without whom I would not have been able to complete this research, and without whom I would not have made it through my masters degree.

My professor Yuming Jiang (IIK, NTNU) and supervisor Omkar Kulkarni (Nordic Semiconductor) whose insight and knowledge into the subject matter steered me through this research.

My colleagues at the Bluetooth Mesh group at Nordic Semiconductor: Stine Åkredalen for the great discussions and insights through the thesis, and Anders Storrø for the help with the test network.

Big thanks to my family for all the love and support, and to Norway for opening its doors to me.

Contents

List of Figures

List of Tables

List of Acronyms

ADV Advertisement.

AppKey Application Key.

BLE Bluetooth Low Energy.

Bluetooth SIG Bluetooth Special Interest Group.

BTM Bluetooth Mesh.

DFU Device Firmware Update.

GATT Generic Attribute.

GW Gateway.

ICT Information and Communication Technology.

IoT Internet of Things.

IV Initialization Vector.

LPN Low Power Node.

NetKey Network Key.

NTC Network Transmit Count.

PDU Protocol Data Unit.

POE Power Over Ethernet.

PRC Publish Retransmit Count.

QoS Quality of Service.

RRC Relay Retransmit Count.

RTOS Real-Time Operating System.

RTT Round Trip Time.

SoC System on a Chip.

TTL Time-To-Live.

Chapter 1
Introduction

1.1 Problem description and contribution

Several wireless communication technologies are available today for deploying customer and commercial Internet of Things (IoT) applications. Some examples of these technologies include ZigBee, Thread, 6LoWPAN, and Bluetooth Low Energy (BLE). In terms of protocols, performance, reliability, latency, price, and coverage, these technologies are extremely heterogeneous [RMGG20]. This means that the particular technology to be selected depends on the specifics of the application scenario. There can be some characteristics in which each of them is optimal and others in which they are otherwise sub-optimal. Therefore, it is not possible to conclude that one technology performs better than another for all scenarios [HPG+20].

The ultra-low power consumption of BLE [RGL17] makes it a good option for power-limited devices. BLE has even demonstrated the capability to meet the tight Quality of Service (QoS) requirements of single-hop real-time industrial applications [RGL17]. Thus, in the past years, BLE has become a popular choice for many applications including wearable devices, home automation, and IoT [RMGG20]. Due to BLE being a native implementation feature in the majority of today's devices (such as laptops and smartphones), users can interact with BLE objects in the environment without requiring additional tools [CDBF19].

Bluetooth Mesh (BTM) is a recent technology developed by the Bluetooth Special Interest Group (Bluetooth SIG), released in June 2017. Using the existing BLE protocol stack, BTM allows many-to-many device communication over BLE radio [RMGG20]. The BTM functionality is entirely dependent on the BLE protocol stack. Using BTM, thousands of devices can be connected to create a mesh network, with greatly enhancing coverage. The simplicity of its structure, the low cost of implementation, and the ability to work with existing BLE devices make BTM an appealing technology for use in mid-range IoT applications [RMGG20].

Since BTM was introduced in the market, not many studies have been performed evaluating the BTM technology in IoT networks so far. These studies were either based on simulation and not an actual network like [DSLA20], or based on real-world experiments like [LN20] and [Lab] but examining the latency and reliability of BTM networks based only on the protocol parameters. The aspect of sensor data collection over BTM is not explored in depth. Therefore, this thesis will focus on a wider exploration of how BTM networking can be used for data collecting in IoT networks. This includes the exploration of performance of the BTM sensor data collection in the context of some BTM protocol-related and protocol-non-related parameters. The protocol-related parameters are the Publish Retransmit Count (PRC), the Network Transmit Count (NTC) and the Relay Retransmit Count (RRC). The protocol-non-related parameters are the packet sending interval, the packet sending randomization and the packet payload redundancy. The project is done in collaboration with Nordic Semiconductor ASA, a Norwegian fabless semiconductor company specializing in wireless communication technology that powers the IoT [NOD]. The work includes tests that were conducted on the Nordic Semiconductor large-scale test network in Trondheim to determine the reliability, in terms of data loss i.e. packet delivery ratio, of BTM networks.

The importance of the reliability assessment in data collection sensor networks comes from that these networks are used in environments where sensors have functional and operational challenges. The challenges are related to resource limitation e.g. energy supply [VBPP17], and to the physical barriers and the noise in the real environment of deployment. Additionally, the sensors are left unattended to do their job properly and efficiently where data is collected in real time. This can cause packet loss which hamper the network activity. Thus, continuous delivery of data requires reliable data transmission.

This thesis reports the results of several tests carried on an actual BTM network consisting of 100 nodes distributed in a 1400 m^2 office area. One node in the network works a collector or a gateway. The other nodes, i.e. sensors, periodically generate data packets and propagate them, using BTM, to the gateway. The gateway sends the content of the received packets over the Ethernet to a dedicated PC for further analysis.

The results are analyzed to investigate how the different protocol-related and protocol-non-related parameters affect the packet delivery ratio, hereafter referred to as reliability, and how to achieve high reliability in this network. The results show that BTM networks, when using unsegmented packets, have a high degree of reliability when the relays and the non-protocol-related parameters, i.e. the packet sending randomization and the payload redundancy, are chosen properly, and when the protocol-related parameters, PRC, NTC and RRC, are tuned. A paper, attached

in the appendix, has been drafted and is under submission.

1.2 Outline

This thesis is divided into nine chapters as follows:

- Chapter 1 presents this introduction.
- Chapter 2 is used to establish the knowledge context essential for understanding BTM.
- Chapter 3 introduces the test network that was used to do this thesis, while chapter 4 discusses the related works.
- Chapter 5 presents the methods that were used to conduct the tests on the test network.
- Chapter 6 shows the results gathered through running the tests, and chapter 7 discusses these results.
- Chapter 8 contains the conclusion of this thesis and chapter 9 contains future work.

Appendix includes the paper that has been drafted and is under submission.

Chapter 2
Bluetooth Mesh basics

This section gives a closer look at the Bluetooth Mesh (BTM) architecture, its stack layers and their respective responsibilities, and the packet structure. The features and the design principles of BTM will also be covered in this section.

BTM is a short-range wireless mesh networking standard which allows for many-to-many communication over a BLE radio [RMGG20]. It enables the creation of large-scale low-data-rate short-range wireless network [Mesh-NW]. The mesh devices communicate directly with each other without the use of a centralized controller. This gives the mesh network the advantage of better scalability and reliability as the network has no single point of failure [Mesh-NW].

A device that is a part of a mesh network and able to transmit and receive packets in this network is called a *node* or a *provisioned device*, and that which is not is called *unprovisioned device*. Thus, the *provisioning process* is the mechanism that transforms an unprovisioned device into a node. Each node is required to have at least one *element* which is an addressable item within a node or a device. A node could be composed of several elements. An example of an BTM device is a light bulb and a temperature sensor.

To exchange the data between BTM nodes, BTM uses a publish/subscribe paradigm. This paradigm requires that the element has an address. Three types of addresses are defined: Unicast, group, and virtual addresses. The sender *publishes* a packet to a certain address. If this address is unicast, the destination is a single element inside a node and it is automatically processed by this element upon reception. But, if the address is a group/virtual address, elements that are interested in receiving the packet will *subscribe* to such group/virtual address, and only they would process that packet.

2.1 Architecture

BTM offers a *full-stack* connectivity solution i.e. everything from the physical low-level BLE radio layer to the high-level mesh model application layer is defined. The Mesh Profile Specification [Gro19] defines as a layered architecture as shown in Figure 2.1.

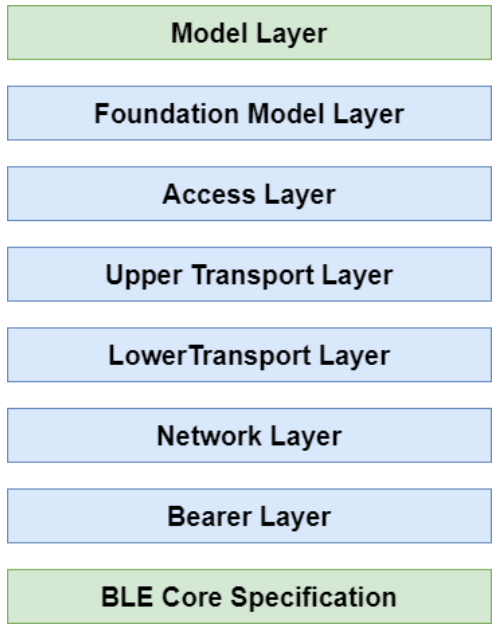

Figure 2.1: Bluetooth Mesh system architecture

- **Model Layer**

  Model layer defines the application functionality. Models are essentially collections of states, state transitions, bindings, messages, and other behaviors. An example of a model is a lighting model.

- **Foundation Model Layer**

  As part of the foundation model layer, models that are related to managing and configuring mesh devices are implemented.

- **Access Layer**

  The access layer defines how higher layer applications can use the upper transport layer. It defines and controls the application data encryption and decryption performed in the upper transport layer. The access layer checks also whether the incoming application data has been received in the context of the right network and application keys before forwarding it to the higher layer.

- **Upper Transport Layer**

  In this layer, application data is encrypted, decrypted, and authenticated before it is passed to and from the access layer. The upper transport layer is designed to provide confidentiality of access packets. The upper transport layers generate and send packets between upper transport layers on different peer nodes. These packets are called as transport control packets.

- **Lower Transport Layer**

  The Protocol Data Unit (PDU)s sent from the upper transport layer are taken by this layer and sent to the lower transport layer on a peer device. If a PDU does not fit into a single TransportPDU (See section 2.2), the lower transport layer splits this PDU into *segments*, that is, breaking the PDU into multiple TransportPDUs. On the receiving device, the low transport layer *reassembles* the received segments into a single upper transport layer PDU.

- **Network layer**

  The packet address types and the network packet format are defined by the network layer. The network layer decides whether incoming and outgoing packets should be forwarded for further processing or dropped. It is also responsible for the network-level encryption and authentication in addition to packet relaying.

- **Bearer Layer**

  The bearer layer defines how the different BTM packets are handled. There are two types of bearers in BTM, namely the Advertisement (ADV) bearer and the Generic Attribute (GATT) bearer. The ADV bearer takes care of handling packets using the advertisement mode of BLE. The GATT bearer is provided to enable communication with devices that are not capable of supporting the advertising bearer to participate in a mesh network e.g. a mobile phone.

- **BLE Core specification**: This is not a layer. This is the full BLE stack which is essential for the basic wireless communications capabilities delivered by the BTM.

2.2 Packet format

The BTM network layer packet is shown in Table 2.1 and illustrated in Figure 2.2:

- IVI : It is used to authenticate and encrypt the PDU.
- NID: It is used to identify the Encryption Key and Privacy Key used to secure the PDU.

Field Name	Bits	Notes
IVI	1	Least significant bit of Initialization Vector (IV) Index.
NID	7	Value derived from the Network Key (NetKey) used to identify the Encryption Key and Privacy Key used to secure the PDU.
CTL	1	Network Control.
TTL	7	Time To Live.
SEQ	24	Sequence Number.
SRC	16	Source Address.
DST	16	Destination Address.
Transport PDU	8 to 128	Packet payload from the model layer.
NW MIC	32 or 64	Packet Integrity Check for Network.

Table 2.1: BTM packet fields definitions

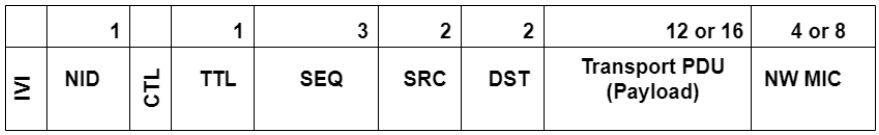

Figure 2.2: Bluetooth Mesh network layer packet format. Each number represents the size of the BTM packet field in bytes.

- CTL: It is used to determine if the packet is part of a Control packet (CTL is 1), then NW MIC is 8 bytes, or an Access packet (CTL is 0), then NW MIC is 4 bytes.
- TTL: TTL is set by the origination element so that the PDU can be relayed in a mesh network. The PDU can be relayed only if TTL > 1. Every time the PDU is relayed in a mesh network, TTL is decremented by 1. If TTL is 0 or 1, the PDU will not be relayed.
- SEQ: A unique number for each new PDU sent from the origination element.
- SRC: Defines the originating element. It shall be a unicast address. SRC is set by the originating element.
- DST: Defines the element(s) that this PDU is directed toward. It shall be a unicast address. DST is set by the originating element.
- TransportPDU: It is the payload sent from the application layer on the originating element. When the CTL bit is 0, then the maximum size of the TransportPDU is 16 bytes, while it is 12 bytes when CTL is 1. If the payload

is higher than the maximum size of TransportPDU, then the packet will be divided into *segments* where the maximum size of each segment does not exceed the maximum size of the TransportPDU. Otherwise, the packet is called *unsegmented*. One byte of the payload of the unsegmented packet will be used to specify the model specific packet type, this byte is called the **opcode**. The TransportPDU field is set by the originating lower transport layer.

- NW MIC: It is used to authenticate that the DST and TransportPDU have not been changed. The NW MIC is set by the network layer at each node that transmits or relays the PDU.

2.3 Node types and design principles

2.3.1 Node types

The functionality of a BTM node can be determined by the features it supports. A BTM node can optionally support one or more additional features [Gro19]. There are 4 features that a node can support; namely relay, proxy, low power and friend.

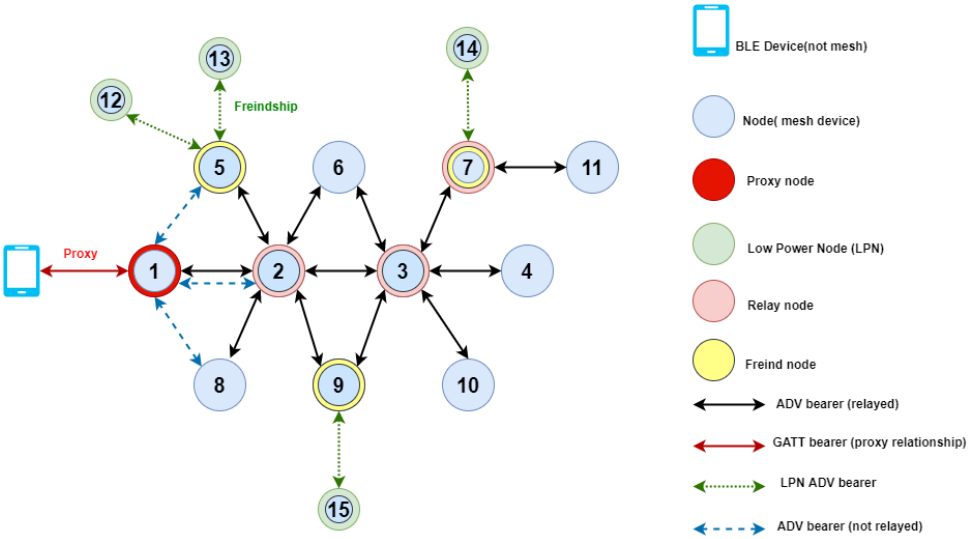

Figure 2.3: Nodes types and intercommunication in a BTM network.

- Relay feature: The relay feature may be implemented by the network layer. A node that supports the relay feature is called a *relay node* or just a *relay*, hereafter referred to as relay. A relay is a node that receives and then retransmits mesh packets over the advertising bearer to enable larger networks. Nodes 2 and 3 in Figure 2.3 are relay nodes.

- Proxy feature: Like the relay feature, the proxy feature may also be implemented by the network layer. A node that supports the proxy feature is called a *proxy node*. A proxy node is able to relay/forward packets between non-mesh-supported BLE devices and a mesh network. It acts as an intermediary and uses GATT operations to allow devices from outside the mesh network to interact with the network. Nodes 1 in Figure 2.3 is a proxy node.

- Low Power feature: A node that supports the low power feature is called a *Low Power Node (LPN)*. An LPN is a node that has the ability to operate within a mesh network at significantly reduced reception duty cycles to preserve energy as much as possible. In order to enable an LPN to reduce its receiver duty cycle and save energy, it needs a friendship relationship with another node, called *friend node*. Nodes 12, 13, 14 and 15 in Figure 2.3 are LPN nodes.

- Friend feature: The *friend node*, supporting the friend feature, provides assistance to an LPN in reception, by storing and forwarding packets destined to that node. Forwarding by the friend node is made on-demand when the LPN polls the friend for packets awaiting delivery. Nodes 5 and 9 in Figure 2.3 are friend nodes.

2.3.2 Design principles

The features discussed above give BTM many characteristics that make it unique:

- The publish/subscribe paradigm: As discussed earlier, BTM follows a publish/subscribe paradigm for communication between nodes. This gives flexible address assignment and group casting [DSLA20].

- Two-layer security: A packet sent from a BTM node is encrypted and authenticated in two layers; the application layer and the network layer. In the application layer, an Application Key (AppKey) is used to provide confidentiality and authentication of application data sent from the node. In the network layer, a NetKey is used to provide security within a mesh network. This makes it possible for intermediary devices to be used to relay packets without allowing these devices to read or alter the application data in the relayed packets.

- Power saving with "friendship": Friendship is established by an LPN to a friend node. Once established, the friend node performs actions that help to reduce the power consumption on the LPN. The friend node maintains a cache that stores all the incoming packets addressed to the LPN and delivers those packets to the LPN when requested. In addition, the friend node delivers security updates to the LPN.

- Managed flooding: Managed flooding is a process between flooding[1] and routing[2] where the relay node broadcasts packets to all nodes within its radio range. In managed flooding when a packet is relayed, the Time-To-Live (TTL) that is assigned to it gets decremented by one with each retransmission. A packet is only retransmitted when its TTL is greater than one. Additionally, the packets are cached. Therefore, a received packet that already exists in the cache of the relay node is automatically discarded and not retransmitted.

As a way of providing an appropriate level of reliability in a mesh network, the Bluetooth Mesh Profile [Gro19] allows for packets repetitions at the application and the network layer. These repetitions of the BTM packets can be controlled by three parameters, hereafter called *retransmission parameters*:

- Publish Retransmit Count (PRC): It controls the number of replicas of a message published by a model. These replicas are spaced between them depending on a time interval specified by the model, hereafter called *PRC interval*. Each replica has a unique sequence number.
- Network Transmit Count (NTC): It controls the number of packet transmissions of the Network PDU originating from the node. The space between the transmissions depends on a time interval, in milliseconds, hereafter called *NTC interval*. NTC is an integer that can be chosen from the range [0, 7], while NTC interval is also an integer in the range [0, 330] and must be a multiple of 10.
- Relay Retransmit Count (RRC): It controls the number of packet retransmissions of the Network PDU relayed by the node. The space between the transmissions depends on a time interval, in milliseconds, hereafter called *RRC interval*. Like NTC, RRC is an integer that can be chosen from the range [0, 7]. RRC interval is also an integer in the range [0, 330] and must be a multiple of 10.

Figure 2.4 shows how retransmission parameters in a BTM network works. Having PRC = 1, two messages with time-space equals PRC interval will be generated at the application layer of the originating node. Each message has a unique sequence number. With NTC = 1, each message received from the application layer, will be transmitted twice. The time-space between these two packets is the NTC interval.

[1] Flooding is a very simple mechanism for propagating packets in a network using broadcast [DSLA20].

[2] Routing is the process of selecting a path for traffic in a network or between or across multiple networks.

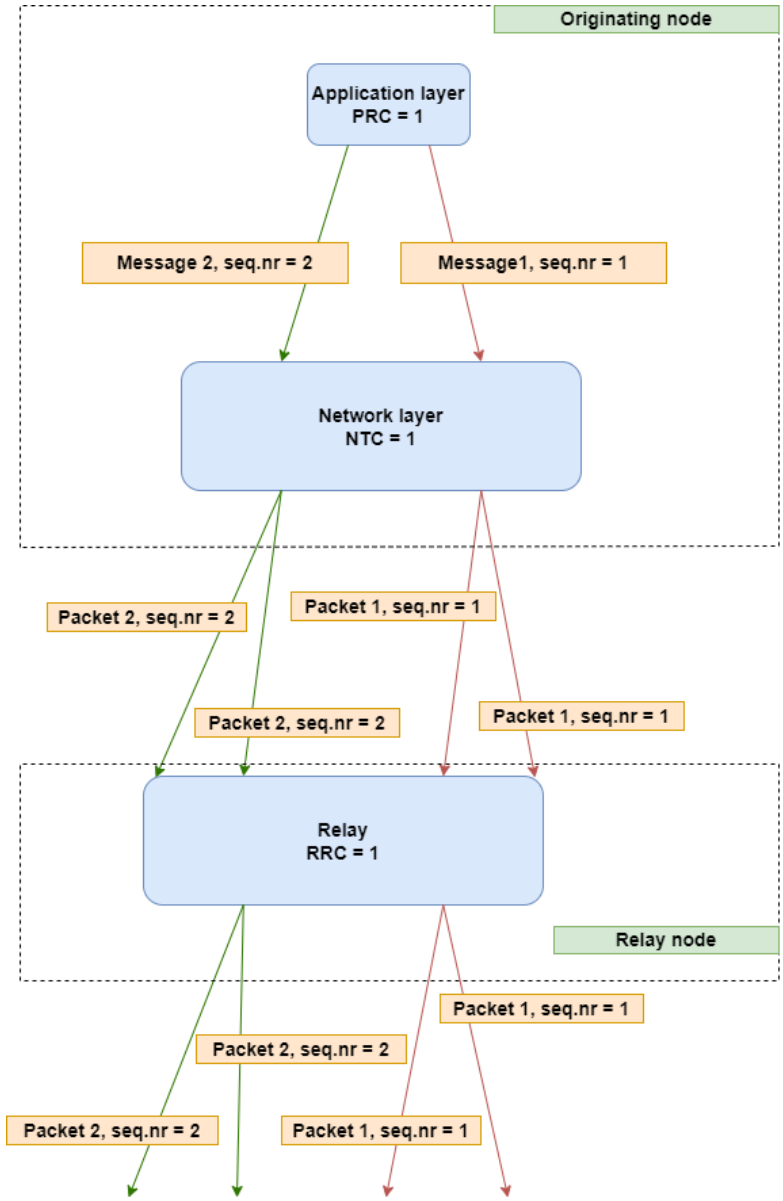

Figure 2.4: Retransmission parameters in a BTM network.

Thus, 4 packets with two unique sequence numbers are sent from the originating node. When these packets, whose TTL > 1, reach a relay node whose RRC $= 1$, only one packet of the two that have the same sequence number will be retransmitted twice. The time-space between the retransmission of these packets equals the RRC interval.

Chapter 3
Related works

This section presents previous works on BTM networks. These works focus mainly on the performance evaluation of BTM networks.

3.1 Silicon Labs: Bluetooth Mesh Network Performance

Silicon Labs, hereafter referred to as SiLabs, is a fabless technology company that designs and manufactures semiconductors, silicon devices and software [Si-Labs].

In its report [Lab], Bluetooth Mesh Network Performance, SiLabs deliver a performance evaluation of BTM. This includes the testing of latency, reliability and scalability in BTM networks. The testing was conducted over actual wireless devices in test networks. The BTM devices were deployed in hallways, offices and open areas, Figure 3.1, in a 2230 m^2 office area with active wireless networks like Wi-Fi in range. The testing network consisted of 192 nodes. Several tests were conducted using different payloads with both segmented and unsegmented messaging. The results show that the reliability of BTM networks is greater than 99% when the application payload can fit in a single packet, i.e. using unsegmented packets, and the relay count RRC is low.

The report concludes that in order to get low latency and high reliability in BTM applications:

- packets should be unsegmented.
- A bigger payload results in segmentation which directly affects latency and, for this reason, multi-casting should avoid using packet segmentation.
- When network size and number of hops increase, relay selection becomes critical for network performance.

Figure 3.1: Testing clusters in the Silicon Labs R&D Facility, source [Lab]

3.2 Ericsson: Bluetooth Mesh Networking

Ericsson is a founding member of the Bluetooth SIG and a provider of Information and Communication Technology (ICT) worldwide [Ericsson]. In September 2020, Ericsson published a white paper [DSLA20] called "Bluetooth mesh networking". The white paper shows the results of a series of performance tests conducted on a BTM network.

Unlike SiLabs, the tests were carried out through *simulation* where they carried out a full stack implementation of the Bluetooth Mesh Profile in a system-level simulator. No specifics were given regarding the virtual environment. The performance evaluation was conducted on a large-scale office scenario: 879 devices, including window sensors, occupancy sensors, HVAC[1] sensors and actuators, light switches and light bulbs, all deployed in an area of approximately 2000 m^2. The network has one or more gateways (GW). The sensors send *unacknowledged*[2] packets to the Gateway (GW) which forwards these packets to the cloud for further analysis. The relays in the network are selected only from the powered nodes which are located in open areas like corridors.

The results show that the best reliability performance was higher than 99.9%, meaning that more than 99.9% of the packets sent from the sensors reached their

[1]HVAC: Heating, ventilation, and air-conditioning.
[2]The purpose of the acknowledgment packet is to let the sending node know that the sent packet was received by the intended receiver.

destination successfully. The paper concludes that the best results were obtained when only 6 (1.5%) relays were deployed in a 1000 m^2 area.

3.3 Aijaz *et. al*: Experimental Evaluation and Optimization of Bluetooth Mesh

A paper published by Aijaz *et. al* [ASLM21] studies the impact of latency performance for 100% reliability (In terms of packet delivery), the impact of packet segmentation and the performance optimization with adjustments for advertising interval[3] and scanning interval[4] in BTM networks.

Figure 3.2: Test-bed for experimental evaluation of BTM, source [ASLM21]

Several tests were conducted on an actual network. The test-bed, Figure 3.2, consists of 20 nodes over two floors in an office environment covering approximately 600 m^2. The test-bed setup depicts a challenging multi-hop mesh scenario due to weak link between the two floors. The nodes in the test-bed experience shadowing from humans, and external interference due to the existence of other wireless networks. All the nodes are relays and packets are acknowledged and unsegmented.

The paper concludes that the latency performance for perfect reliability can be optimized through adjustment of advertising/scanning parameters, extended advertisements, simple power control techniques, and customized relaying. It also states that BTM provides a flexible solution that can be applied to various monitoring and control applications

[3]The time interval between BLE advertising packets sent periodically on advertising channels when a BLE device is in advertising mode.

[4]Scan interval defines the frequency that the scanner hops between a BLE device advertising channels.

3.4 De Leon *et. al*: Bluetooth Mesh in Monitoring Applications

A paper published by De Leon *et. al* [LN20] performs an experimental performance evaluation of BTM for monitoring applications aiming to investigate the limits of BTM. The paper is based on a real-world setup. The main focus of the paper is packet delivery ratio. All the nodes in the test setup periodically generate data packets and propagate them towards a base station. The focus is on the cases where multiple mobile nodes are present in coexistence with other sensor nodes and other wireless devices such as Wi-Fi. The amount of relays is minimized according to the physical limitations of the environment.

The test setup, Figure 3.3, consists of 33 nodes in a 500 m^2 office area. The packets are unsegmented, and the transmission power[5] of the nodes is fixed to -4 dBm and to 0 dBm for broadcasters and relay nodes respectively in order to reduce interference between radios.

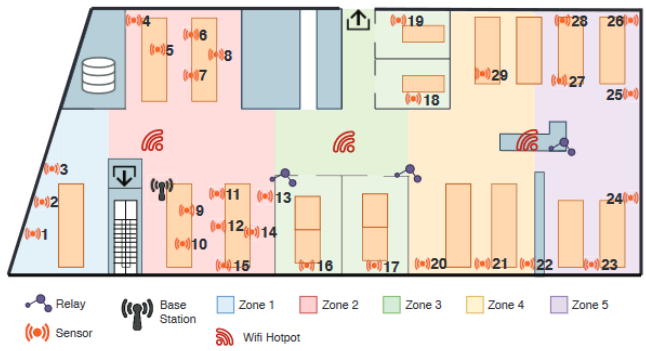

Figure 3.3: Node placement in the experimental setup, source [LN20]

The results show that to achieve a reliable packet delivery service, relay nodes should not get saturated. The paper suggests that the BTM technology has serious limitations for large-scale monitoring applications in which the nodes have frequent data generation, while it can be a promising option for networks with low traffic load (e.g. event-driven applications).

[5]Transmission power is the actual amount of power of radio frequency energy that a transmitter produces at its output. It is measured in Watt or dBm, where 1 mW = 0 dBm.

3.5 Rondón *et. al*: Understanding the Performance of Bluetooth Mesh

A paper published by Rondón *et. al* [RMGG20] performs an evaluation of the QoS performance and scalability of BTM. The most important BTM protocol parameters and their impact on system performance were defined.

The tests were carried out through *extensive simulations*. The test-bed consists of 28 nodes spread around an office environment in which interference data, which are collected from a real office environment, was recorded.

The results show that BTM is particularly vulnerable to network congestion and increased packet collision probability for densely populated deployments. The randomization of the timing parameters of the protocol proved to be a solid starting point to counteract network congestion. The performance of BTM is not notably affected by interference from other BTM devices in the network. However, BTM is still considerably susceptible to interference from other common wireless technologies, such as WLAN.

3.6 Baert *et. al*: An Overview and Experimental Evaluation of Bluetooth Mesh

Baert *et. al* published a paper [BRSH18] concerned with measuring the performance of BTM in terms of latency through theoretical modeling and physical experimentation. The test-bed, Figure 3.4, consists of 22 nodes in a 1000 m^2 office environment at Ghent University, Belgium. Measurements were done considering the Round Trip Time (RTT) against the number of relay nodes and the number of hops.

Conclusions state that the positive impact of a proper backoff mechanism as well as network density in RTT. They also point out the fact that relay features as the BTM backbone rely on non-power-limited devices which makes the deployment of pure low-power devices unrealistic at this current state.

3.7 Reflection on previous works

Through the previous sections, several works concerned with BTM were presented. Many of these works have focused on the performance evaluation of BTM, in terms of latency and reliability, by exploring only the effect of the protocol parameters. The tests in many of these works were not carried out on an actual network, or a data-collecting network. Thus, the most two relevant previous works for this thesis is Ericsson, section 3.2, and De Leon *et. al*, section 3.4, because they study a BTM data-collecting network.

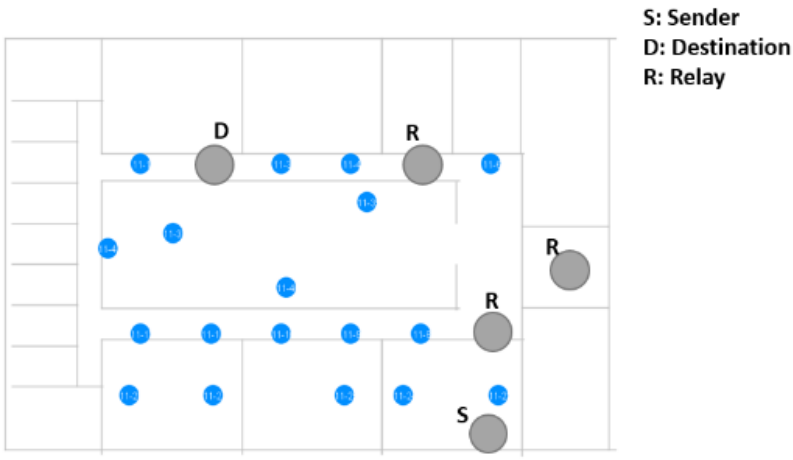

Figure 3.4: The test-bed used to conduct experiments at Ghent University, source [BRSH18]

However, Ericsson's white paper is based on *simulation* where no specifics were given regarding the virtual environment. Choosing the GW(s) and exploring the effect of multiple protocol-non-related parameters on the network reliability were not covered in the paper.

Despite that the work done by De Leon *et. al* was carried out on a real-world office environment and studied the packet delivery ratio, it uses a GW which is not a node in the network. It also does not explore the effect of protocol-non-related parameters on the packet delivery ratio.

This thesis will focus on a wider exploration of BTM in a real-world data-collecting IoT network in an office environment. This includes choosing of the GW and the relays in the network in addition to the exploration of the effect of some protocol-related and protocol-non-related parameters on the packet delivery ratio, i.e. the reliability, in the network.

Chapter 4
The test network

This section presents the large-scale test network that is used to carry out the tests performed in this thesis. This includes the hardware tools, the software tools, and the specifics of the testing environment.

4.1 Hardware tools

The Nordic Semiconductor test network, Figure 4.3, consists of 100 kits installed on an area of 1400 m^2 in Nordic Semiconductor office in Trondheim. Each kit is a custom-made internal Power Over Ethernet (POE) development board labeled POE dev board, Figure 4.1. The POE dev board was specially designed for the test network, and it is not a commercial product. The POE dev board consists of (See Figure 4.2): A High Power(HP) LED, an nRF52840 chip, an antenna, a POE supply, and a Wiznet W5500 chip. The nRF52840 chip is a multi-protocol Bluetooth 5.3 System on a Chip (SoC) supporting among other things BLE and BTM [nRF52840]. Wiznet W5500 chip is a hardwired TCP/IP embedded Ethernet controller that enables easier internet connection for embedded systems [wiznet]. The kits are installed on the ceiling plates in the Nordic Semiconductor office, Figure 4.4, and are roughly evenly distributed throughout the office floor. The kits were distributed in this way to simulate lights inside the office. Each kit has an ID which is affiliated with its MAC address in order to have a better overview and control of the kits in the test network, Figure 4.3. The kits are connected to three stacked[1] POE switches connected to a dedicated computer, hereafter referred to as the remote PC. The test network can be controlled by using the remote PC.

4.2 Software tools

The nRF software development kit (SDK) in combination with one of the nRF52 series hardware development kits (DK) by Nordic Semiconductor provide a total

[1] They operate as one virtual device.

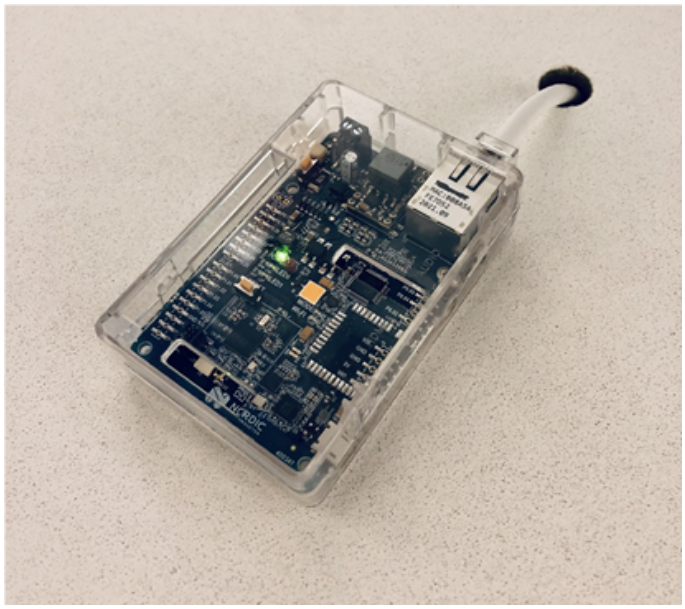

Figure 4.1: POE dev board installed on the ceiling. Source: Private photo

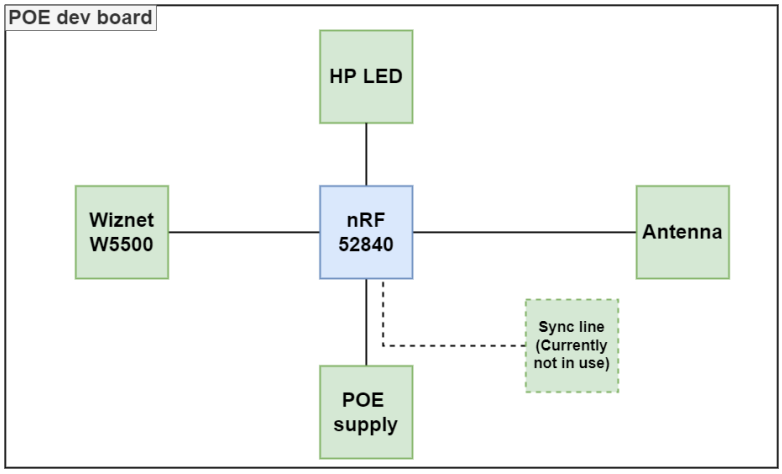

Figure 4.2: The components of the POE dev board. Source: Nordic Semiconductor - internal

Figure 4.3: Nordic test network. The circles with numbers indicate the approximate placement of each node and the node ID. Source: Nordic Semiconductor - internal

Figure 4.4: POE dev boards placement in the office hallway. Source: Private photo

solution for application development with BTM. The nRF Connect SDK (NCS), by Nordic Semiconductor, is an open-source library containing the Nordic-specific source code and open-source project Zephyr which is a Real-Time Operating System (RTOS), a boot loader (MCUboot)[2], drivers and libraries. These software tools, together with the toolchain as described in the NCS documentation [nCS], is everything that is needed to build BTM applications with Nordic Semiconductor nRF52 series device.

- NCS is available through GitHub [nCS].
- nRF SDK documentation [nrf-Connect].
- nRF52840 SoC documentation [nRF52840]

The firmware of all the kits is updated using the same code. This is called Device Firmware Update (DFU). The test firmware is based on v1.7.1 of NCS. In the test network, the kits are DFU'ed over the same Ethernet connection as the rate of data transfer using Ethernet is much higher than in BLE[3]. The DFU process is done by executing a Python script on the remote PC. When the kits are successfully DFU'ed, the kits do not undergo provisioning, but instead they boot up as pre-provisioned devices with fixed unicast addresses. After DFU completes, all nodes boot up as a part of single mesh network.

4.3 The testing environment

The 4^{th} floor of the Nordic Semiconductor building in Trondheim, Norway is the testing environment. It has an area of 1400 m^2. The floor has many walls which are made of concrete or other soft opaque material. Many employees work in the testing environment. Additionally, the test setup experiences external interference from Wi-Fi access points and other Bluetooth devices operating on the floor.

Figure 4.5 shows the Bluetooth traffic on advertisement channels per minute around each node in the test network. This represents the interference from other Bluetooth devices in the test environment. The Bluetooth traffic was measured at 10:00 am on Mondays. This time slot was chosen as the majority of employees are in the office and working which means that the interference could be at its peak.

[2]MCUboot is a secure bootloader for 32-bits microcontrollers [MCUboot].

[3]Data transfer rate in BLE 5.x is 1.4 Mbps [BT] while it ranges from 1 Mbps to 400 Gbps in the Ethernet [IEEE-Ethernet]

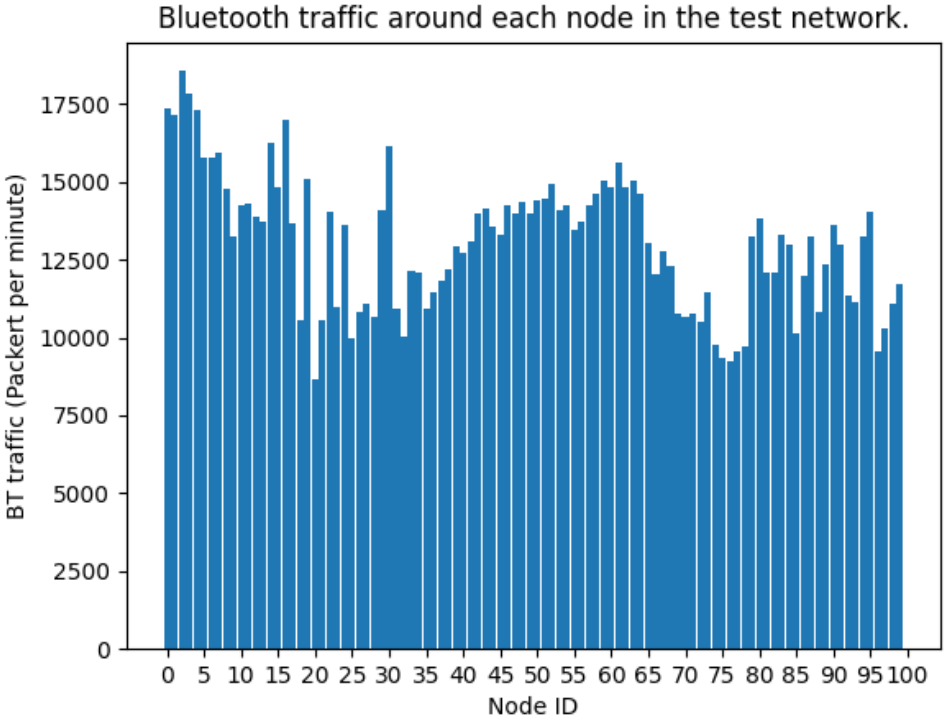

Figure 4.5: Bluetooth traffic around each node in the test-bed

Chapter 5
Methods

This section introduces the tests that are carried out on the test network, and the challenges that are faced during conducting these tests.

The tests are conducted in a way where different parameters are used firstly in isolation from other parameters, and then in combination with other parameters to see the effect of each parameter on the reliability. The parameters, that have a positive effect on the reliability when they are used in isolation, are tested later in combinations. The combinations of the parameters that have the best effect on the reliability are used later in the test. In order not to exclude the effect of different combinations of the parameters, other combinations of all the parameters are tested.

5.1 Design and specification

As mentioned previously, the aim of this thesis is to study the reliability in BTM sensor data networks. Thus, the tests are run on the Nordic Semiconductor test network, hereafter referred to as the test-bed, will have a gateway (GW) and sensors. The GW is a node that receives BTM packets from the other nodes, i.e. sensors, and forwards these packets to a Python script on the remote PC for further analysis.

In other words, in each test there are one GW and 99 sensors in the test-bed where the sensors periodically send BTM packets to the GW. Each packet has a unique tag and a payload which simulates a random value measured by the sensor. When the GW gets a packet, it forwards the packet payload and the sender's address to the Python script on the remote PC. The Python script counts the packets sent from each sensor. When the test is over, each sensor sends the total number of the packets it sent to the GW to the Python script. The Python script then calculates the ratio between the total number of the received packets to the total number of the sent packets from each sensor. This ratio represents the packet delivery ratio, or in other words represents the reliability.

The procedure of the tests, Figure 5.1, to be conducted on the test-bed, is designed as follows:

1. All the kits in the test-bed are DFU'ed with the test code.
2. After DFU'ing, the user sends a command as a data packet over the Ethernet from the test Python script on the remote PC to all the nodes in the test-bed. The data packet, Figure 5.2, consists of:
   a) The packet identifier: This is a 4 bytes field used by the nodes in order to identify that the received packet is an Ethernet packet for running a test.
   b) The test opcode: A 1 byte field used to identify the test. Upon receiving this opcode, the node will start the procedure of choice.
   c) The destination: It is the broadcast MAC address FF:FF:FF:FF:FF:FF which means that the packet is destined to all the nodes in the test-bed.
   d) The test duration: A 2 bytes field specifies how long the test will last.
   e) The sending interval: A 2 bytes field used to specify how often the sensors send a BTM packet to the GW.
   f) The gateway (GW): A 6 bytes field, i.e. the MAC address, that identifies which node in the test-bed is the gateway during the test.
3. Upon receiving the Ethernet packet, the node checks if its MAC address matches the received MAC address of the GW. If not, the node is a sensor otherwise it is a GW. Sensors start the test immediately and a BTM packet is sent to the GW, using the GW unicast address, every sending interval. The packets sent by sensors are unsegmented packets where the available payload is 11 bytes[1]. Of these 11 bytes, a 2 bytes field is used as a packet tag [2] that identifies the packet, and a 2 bytes field is used to simulate a random value measured by the sensor. Thus, of the 11 bytes available on the payload, only 4 bytes are used which means that there are still 7 bytes available, Figure 5.3.
4. When the GW receives a packet from a sensor, it does not acknowledge the packet i.e. it does not send a confirmation back to the sender telling that its sent packet was successfully received.

   Instead of that, the GW *"logs"* the packet over the Ethernet. This means that the GW sends a data packet, which contains the content of the received BTM packet and its sender's unicast address, over the Ethernet to the test Python

[1] The original available payload for the used unsegmented packets is 12 bytes, but one byte is used for the opcode.

[2] A number starts at "1" and increases by "1" every time the sensor sends a new packet to the GW.

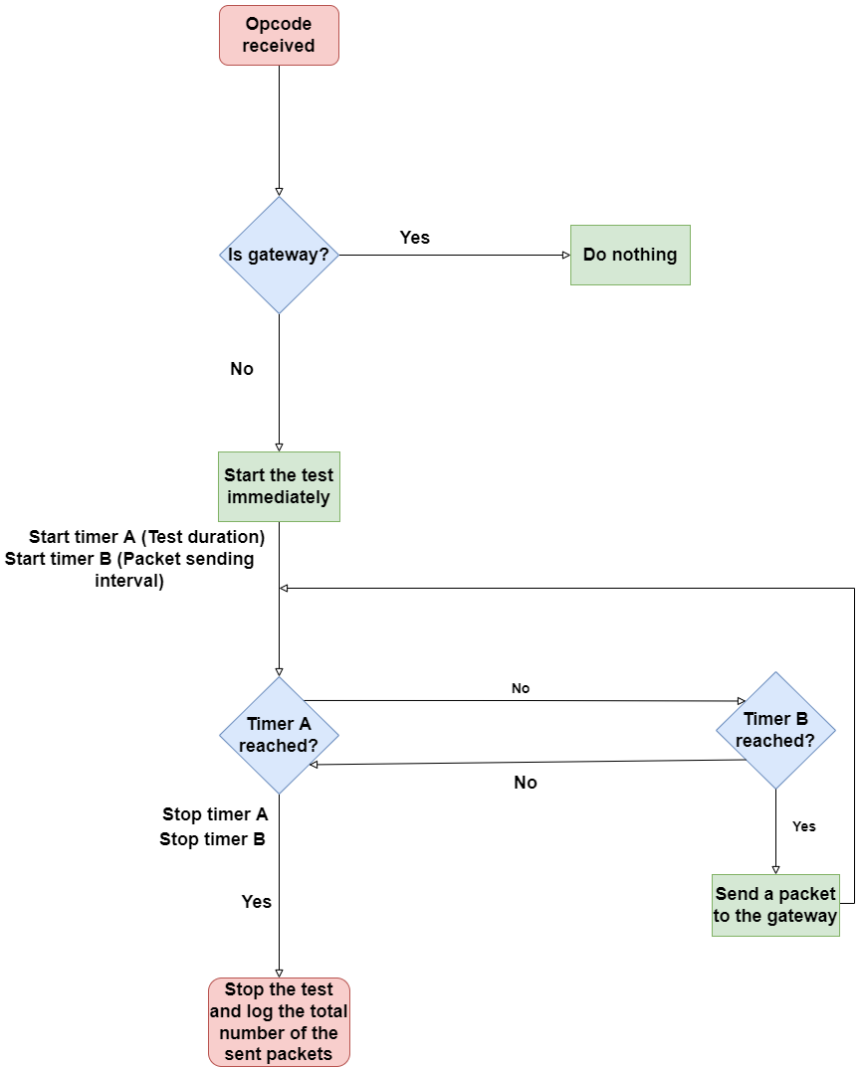

Figure 5.1: Test logic flow chart

4	6	1	2	2	6
Identifier	Destination	Opcode	Sending interval	Test duration	Gateway

Figure 5.2: The packet structure of the test command sent from the remote PC. Each number represents the size of the field in bytes.

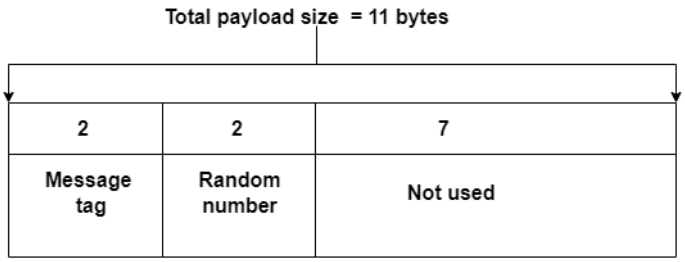

Figure 5.3: The structure of the BTM packet sent from the sensors to the gateway. Each number represent the size of the field in bytes

script on the remote PC. The script parses the content to identify which sensor sent what as it has the unicast addresses of all the nodes in the test-bed[3].

5. When the test duration is over, the sensors stop sending packets to the GW. Then, each sensor logs the total number of packets that were sent to the GW. The test Python script parses the content sent from the sensors, using the nodes MAC addresses, compares it to the logged packets from the GW and saves the results.

5.2 Network preset

Before starting the test, a few assessments and decisions must be made: Choosing a specific node in the network to be the gateway, and which nodes to act as relays in the test-bed. When the gateway and the relays are chosen, the effect of the retransmission parameters on reliability are tested. After that, the effect of some protocol-non-related parameters; namely the sending interval, the packet sending randomization and the packet payload redundancy, on the reliability are tested as well.

In order to check if the test logic works as it should, a small network of "4" POE dev boards was used. These kits were connected to a PC via a stackable switch to simulate the test-bed. The results of running the tests on this small set of nodes, hereafter referred to as desk setup, are not considered to be valid due to that the nodes are very close to each other, a few centimeters. Therefore, packet loss is very unlikely, and the packet delivery ratio is 100% [Qaq21].

5.2.1 Choosing the location of the gateway

Choosing the GW was not important when conducting the test on the desk setup as the nodes are very close to each other [Qaq21]. However, running the test on the

[3]The unicast address of a node is derived from its MAC during the DFU process and it is static.

test-bed can give different results depending on choosing the location of the GW taking into consideration the area of the testing environment and the number of kits. Thus, choosing the GW can play an important role in achieving better reliability and at the same time decrease the traffic in the network as the closer the sensor to the GW is, the fewer hops are needed to get its packets delivered to the GW. For sensors that are far away from the GW, the number of hops needed to forward the packets to the GW may increase.

To choose a GW in the test-bed, the following is done:

- Having no relays in the test-bed, test duration is 10 minutes, sending interval is 5 seconds, NTC = RRC = PRC = 0, the transmission power of the nodes is 0 dBm (1 mW), and the rest of the nodes configuration options are according to the Zephyr-Nordic Semiconductor implementation of BTM [Zephyr-Nordic]. This will apply also in the coming sections unless something else is specified.
- Run a Python script to choose a GW as following: For every node in the test-bed:
  1. Choose this node as a GW.
  2. Run the test explained in section 5.1, and see how many sensors could communicate with the chosen GW. If some sensors are not able to communicate with the GW, find out what is the reason; whether it's for example the physical obstacles or distance to the GW.
  3. The node that receives higher number of packets from higher number of sensors, compared to other nodes, is the default GW.
  4. If two nodes have similar number of responses from similar sensors, choose the one which is located in open areas like corridors. If two nodes are in the open areas, then calculate the average distance to all the sensors. The node with the smaller average distance is chosen as a GW.

5.2.2 Choosing the relays

As the test-bed has 100 kits distributed on 1400 m^2 where there are concrete wall, there may not be a node which can communicate with all the other nodes in the test-bed. Thus, having relay nodes, which forward the packets to the GW from the sensors which do not have direct contact with the GW, can mitigate this problem. The question that rises then: Which nodes in the test-bed can be chosen as relays?

None of the previous works except [DSLA20] mentioned how the relays were chosen. [DSLA20] concluded that only 1.5% relays in a 1000 m^2 is needed for a reliability performance higher than 99.9%, where relays are located in open areas.

Having 100 nodes in an area of 1400 m^2 means that 2.1%[4] relays are needed, i.e. 2 relays. Therefore, 2 nodes are chosen in the network as relays. If using 2 relays is not enough to get the packets from all the sensors delivered to the GW, more nodes from the open areas are chosen. The nodes are chosen by having two relays around the GW from different directions. Then, the number of relays is increased gradually in the two directions till each sensor in the test-bed is covered by at least one relay. The number of relays should be increased gradually to ensure connectivity between relays and at the same time not increase the traffic in the network. Relays have a direct connection to each other i.e. there are no barriers between them. It should, therefore, be possible to find a path between an arbitrary pair of nodes.

5.3 The protocol-non-related parameters

As mentioned earlier, the protocol-non-related parameters that are used in this thesis are: The packet sending interval, the packet sending randomization and the payload redundancy. These parameters are used because the test logic gives a space to test them.

The packet sending randomization is inspired by some previous works like [RMGG20], [BRSH18] and [DSLA20]. However, [RMGG20] used randomization of the timing parameters of the protocol, while [BRSH18] used back-off mechanism for RTT and [DSLA20] used advertising randomization. On the other hand, the sending interval and the payload redundancy are not done by any of the previous works.

5.3.1 The packet sending randomization

In the test described in section 5.1, the sensors start sending packets to the GW immediately upon receiving the command from the remote PC. Additionally, all the sensors use the same sending interval, meaning that they all send packets to the GW approximately at the same time. This could result in packet collision at the GW and the relays, as the GW and the relays could not be able to process all the incoming packets at the same time. Therefore, if the GW and the relays receive the packets at different times, they could be able to process all the incoming packets. This could increase the reliability in the test-bed.

The packet sending randomization is implemented as follows:

- Upon receiving the command from the remote PC, the sensor waits for a random time, hereafter referred to as the random start time.

[4] 1400 * 0.016 / 1000 = 0.021

- After waiting for a random start time, the sensor sends its first packet to the GW. For the next packets, the sensor will pick another random time, hereafter referred to as back-off time, every time it sends a packet to the GW. Hence, the sensor sends a packet to the GW every sending interval + back-off time.

5.3.2 The packet sending interval

Sending interval is a parameter that specifies how often a sensor sends a packet to the GW. It is sent in the Python command from the remote PC to the sensors. Changing the sending interval could have an impact on the reliability as it may lead to less packet collision at the GW and the relays.

5.3.3 The payload redundancy

The structure of the packet sent from the senors to the GW was explained in section 5.1. As explained, the sent packets are unsegmented packets where the available payload is 11 bytes. Of these 11 bytes, a 2 bytes field is used as a packet tag that identifies the packet, and a 2 bytes field is used to simulate a random value measured by the sensor. Thus, of the 11 bytes available on the payload, only 4 bytes are used which means that there are still 7 bytes available.

The payload redundancy means that for every packet sent from a sensor, except for the first one, the payload of the packet will include its payload and the payload of the previous packet. The actual payload of a BTM packet sent from a sensor to the GW, as explained above, is 4 bytes. Since the payload field of each packet has 7 bytes available, another 4 bytes payload can fit in. Thus, the actual payload of each packet would be 8 byes; 4 bytes for current packet and 4 bytes for the previous packet.

Using the payload redundancy can compensate the dropped/missed packets sent by a sensor.

5.4 Challenges related to the work

Like any other project, it did not work as planned and some unexpected challenges were met during the execution of the tests. There are mainly two challenges: Not being able to run tests and not all the nodes are DFU'ed successfully.

5.4.1 Not being able to run tests

As mentioned earlier, the nodes are POE'ed and connected to the remote PC through a stackable switch. Commands to start the test are sent from the remote PC to the nodes over the Ethernet, Figure 5.4. During the first month of the master thesis,

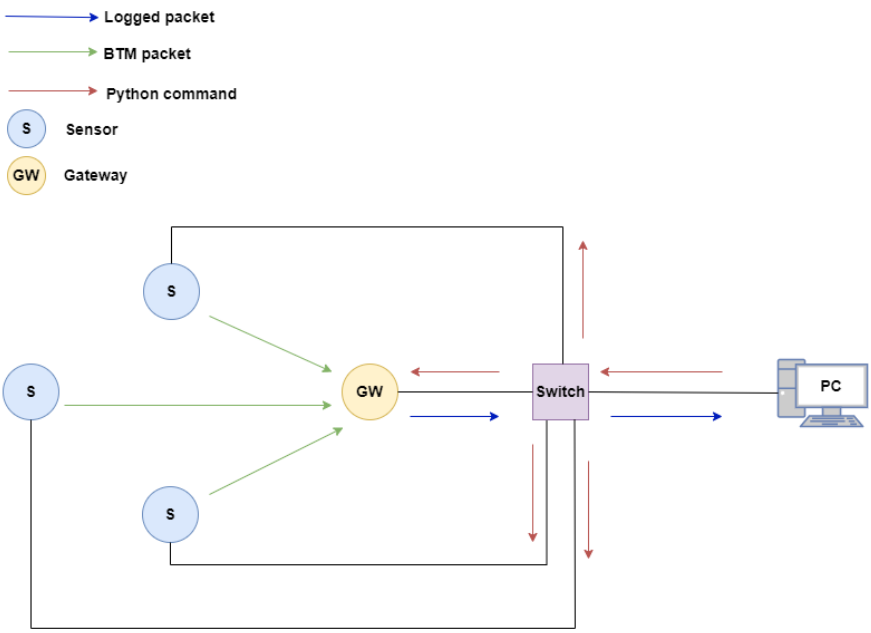

Figure 5.4: Test structure

Windows 10 was used to DFU the kits and run the tests. However, this had to be changed afterward. The reason for that is that the connection to the kits became unstable meaning that the command packets sent from the Python script on the remote PC to the nodes were sometimes received, and sometimes they were not. This seemed to be a problem for all the users of the test-bed on the 4th floor at Nordic. Many attempts were made to find out how to solve this problem, but none of them was successful.

Many various alternative solutions to similar problems reported by other people were tried. This included using a different IP range for the test-bed, resetting Winsock[5], updating the NIC[6] driver, using only IPv4 and changing the firewall settings. Unfortunately, none of these suggestions worked. After two weeks of working on the issue, a workaround was found. The workaround is using Linux instead of Windows 10. Linux was installed on the remote PC. DFU'ing and running the tests from now on are done using Linux. This worked perfectly well. The problem has never been experienced since switching to Linux.

[5]Winsock is a programming interface and a supporting program that handles input/output requests for Internet applications in a Windows operating system.

[6]A network interface card (NIC) is a hardware component without which a computer cannot be connected over a network.

5.4.2 Not all the nodes are DFU'ed successfully

The test-bed was installed to conduct different types of tests. Every time a new test is run on the network, the user DFU's the kits in the test-bed with his/her code, checks if the kits were successfully DFU'ed, and then runs his/her test. But before that, the test must be run successfully on a smaller set of nodes. If the nodes in the test-bed are not successfully DFU'ed, then the user DFU's them again. If this did not work, the nodes are DFU'ed with a very simple firmware that ensures the connectivity between the nodes and the remote PC. After DFU'ing with this simple firmware, the nodes should be able to be successfully DFU'ed with a new test. This was not the case when the tests of this thesis were going to be conducted on the test-bed.

After running the test on the desk setup where there are 4 nodes, the test was going to be conducted on the test-bed. After DFU'ing the kits, it was noticed that 30 nodes were not successfully DFU'ed. These nodes were re-DFU'ed, but nothing changed. Several attempts were made to DFU all the kits in the test-bed using the simple firmware, but it did not work. After spending almost 3 days trying to DFU the nodes, it was decided to reset them completely by re-flashing the firmware manually. The 30 nodes to be flashed manually were defined, and then flashed one by one by directly connecting each node to a laptop and flashing it with the the test code. It took 2-3 hours, but afterward, all the nodes were able to be DFU'ed with the test. This problem has never experienced since then. A new mechanism was added to the DFU'ing process to avoid the problem. The Mesh team helped to improve the Ethernet DFU procedure. A new mechanism was added to the firmware. If the new firmware after DFU fails to reach Ethernet gateway, it will be reverted. This mechanism solved the problem with nodes getting broken due to problematic firmware.

Chapter 6
Results and analysis

This section shows the node that was chosen as a GW, the relays, and how reliability in the network is affected by changing some protocol-related and protocol-non-related parameters using the test described in section 5.1.

6.1 The gateway and the relays

6.1.1 The gateway

Node 90 was chosen as a GW after implementing the method explained in subsection 5.2.1. Figure 6.1 shows the responses node 90 got from all the sensors in the test-bed.

6.1.2 The relays

As seen from Figure 6.1, the GW did not get responses from all the sensors in the network. This means that the packets sent from some sensors do not reach the GW. Having relays in the test-bed can mitigate this problem.

Using 2 relays in the test-bed did not result in packets from all the sensors reaching the GW. Some sensors were still not able to get their packets delivered to the GW despite choosing different sets of relays, Figure 6.2. Using different sets of 3 relays gave slightly better results, Figure 6.3.

However, choosing more relays as explained in subsection 5.2.2, with TTL for packets sent from the sensors is 6[1], resulted in a set of *16 relays* was chosen, Figure 6.4. Using this set of relays, packets from all sensors reached the gateway, Figure 6.5. Removing any relay of this set would result in a noticeable packet loss for some sensors. The reliability in the test-bed using only this set of relays is 63.4%. This

[1]The number of hops needed to forward a packet from a sensor that is furthest from the GW, like node 5.

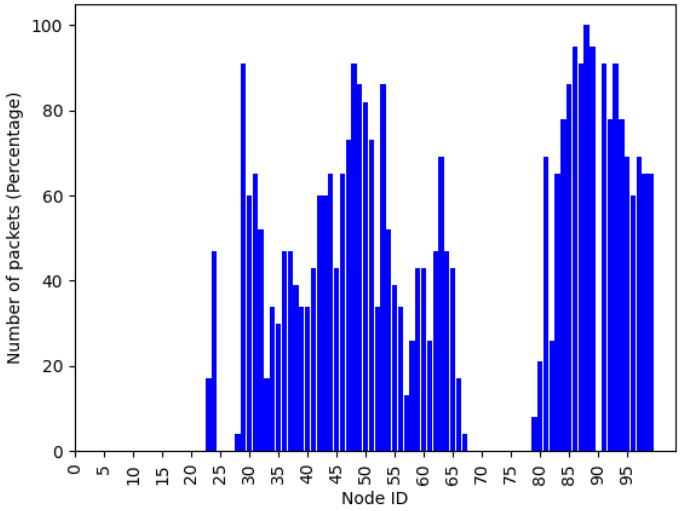

Figure 6.1: Responses from all the sensors in the test-bed sent to node 90. Each sensor sends 120 packets.

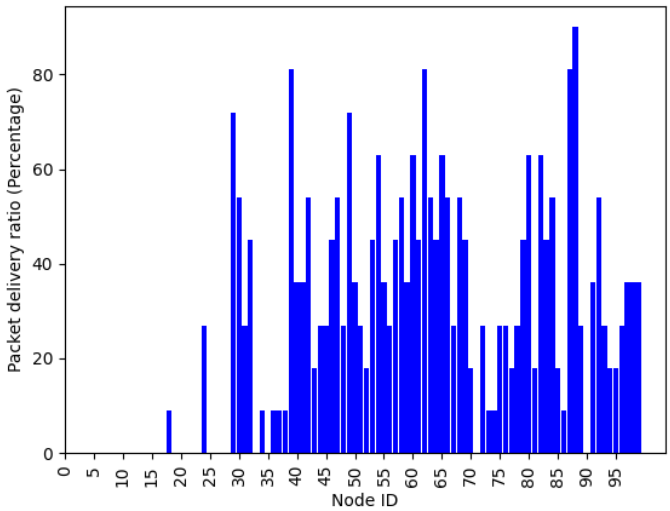

Figure 6.2: Responses from all the sensors in the test-bed using nodes 24 and 62 as relays.

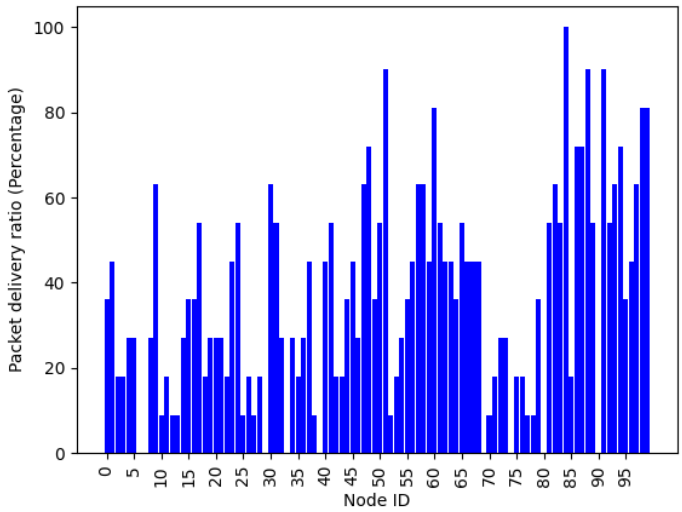

Figure 6.3: Responses from all the sensors in the test-bed using nodes 19, 24 and 80 as relays.

reliability is achieved with: PRC = NTC = RRC = 0, sending interval is 5 seconds, 120 packets sent to the GW and the transmission power of each sensor is 0 dBm, and the rest of the nodes configuration options are according to the Zephyr-Nordic Semiconductor implementation of BTM [Zephyr-Nordic].

6.2 Protocol-related parameters: Retransmission parameters

As explained in section 2.3, retransmission parameters are used to provide an appropriate level of reliability in BTM networks. These parameters are NTC, RRC, and PRC.

- PRC: It controls the number of replicas of a packet published by a model.
- NTC: It controls the number of packet transmissions of the Network PDU originating from the node.
- RRC: It controls the number of packet retransmissions of the Network PDU relayed by the node.

The tests were conducted on the test-bed firstly using PRC, NTC, or RRC and then using permutations of these parameters with the following applies, in this section and next sections, unless something else is specified:

Figure 6.4: The sensors, relays, and the gateway in the test-bed. Each circle with a number inside represents the approximate placement of each node and the number represents the node ID.

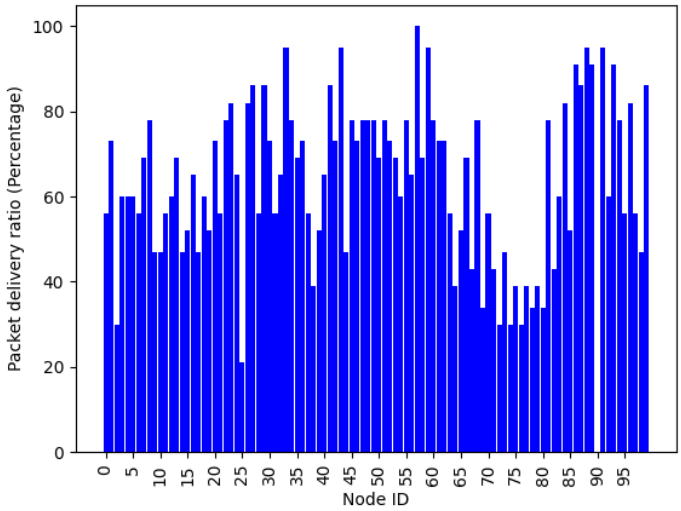

Figure 6.5: Responses from all the sensors to the GW, with 120 packets sent to the gateway, using relays from Figure 6.4.

- Having the relays from Figure 6.4.
- 120 packets[2] sent from each sensor to the GW. This number, 120 packets, was chosen as each test for different parameter(s) is conducted several time. Thus, having a higher number means more time to spend on each test, i.e. the process will be time consuming as the time dedicated for this thesis is limited; 21 weeks.
- The transmission power of each node in the test-bed is 0 dBm (1 mW).
- The rest of the nodes configuration options are according to the Zephyr-Nordic Semiconductor implementation of BTM [Zephyr-Nordic].

6.2.1 PRC

Running the test on the test-bed using different values for PRC with
PRC interval = 0 milliseconds[3], NTC = 0 and RRC = 0 gave the following results:

PRC	0	1	2	3	4
Reliability	63.4 %	59.3%	51.5%	55.3%	59.5%

Table 6.1: Reliability in the test-bed using only relays and PRC

Table 6.1 shows that using PRC *decreases* the reliability. With PRC = 2 the reliability drops by more than 12%, compared to the reliability when PRC = 0. Then, the reliability increases as PRC increases but it is still lower than what it is when PRC = 0.

6.2.2 NTC

Running the test on the test-bed using different values for NTC with
NTC interval = 20 milliseconds[4], PRC = 0 and RRC = 0 gave the following results:

NTC	0	1	2	3	4
Reliability	63.4 %	68.7%	69.7%	67.6%	65.5%

Table 6.2: Reliability in the test-bed using only relays and NTC

Table 6.2 shows that using NTC *increases* the reliability. The reliability reaches its peak when NTC = 2. Increasing the NTC after that decreases the reliability which still better than what it is when NTC = 0.

[2]Test duration = 10 minutes and sending intervals = 5 second
[3]The packet and its replica(s) are sent at the same time.
[4]According to the Zephyr-Nordic Semiconductor implementation of BTM [Zephyr-Nordic].

6.2.3 RRC

Running the test on the test-bed using different values for RRC with
RRC interval = 20 milliseconds[4], PRC = 0 and NTC = 0 gave the following results:

RRC	0	1	2	3	4
Reliability	63.4%	67,7%	70.4%	69.8%	66.1%

Table 6.3: Reliability in the test-bed using only relays and RRC

Table 6.3 shows that using RRC *increases* the reliability. The reliability reaches its peak when RRC = 2. Increasing the RRC after that decreases the reliability which still better than what it is when RRC = 0.

6.2.4 PRC, NTC and RRC combinations

Table 6.1, Table 6.2 and Table 6.3 show that NTC or RRC increases the reliability in the test-bed while PRC decreases it. Thus, combinations of both NTC and RRC, whose values gave better reliability, are tested to see how the reliability would be affected. PRC will be tested with some combinations of NTC and RRC to see if it still gives a negative effect on the packet delivery ratio.

NTC	RRC	PRC	Reliability
1	1	0	69.3%
1	1	1	49.1%
1	2	0	73.8%
1	2	1	62.5%
1	3	0	65.6%
1	3	1	51.5%

Table 6.4: Reliability in the test-bed using relays and different combinations of the retransmission parameters.

Table 6.4 shows that the reliability increases when using a combination of NTC and RRC but only when PRC = 0. Table 6.4 and Table 6.1 show that using PRC in isolation or in combination with the other two retransmission parameters decreases the reliability. Therefore, more combinations of NTC and RRC will be tested but with PRC = 0.

NTC	RRC	PRC	Reliability
2	1	0	69.5%
2	2	0	70%
2	3	0	67.1%
3	1	0	63.8%
3	2	0	66.8%
3	3	0	57.3%

Table 6.5: Reliability in the test-bed using relays and different combinations of NTC and RRC.

NTC	RRC	PRC	Reliability
1	2	0	73.8%
2	1	0	69.5%
2	2	0	70%

Table 6.6: Highest reliability in the test-bed using the retransmission parameters

Table 6.6 shows that the highest reliability in the test-bed was reached when NTC = 1, RRC = 2 and PRC = 0. However, reliability is still not high enough, and as it is seen in Table 6.4 and Table 6.5 , increasing NTC and RRC did not increase the reliability. So, how to enhance the reliability? The answer could be in using some protocol-non-related parameters.

6.3 Protocol-non-related parameters

As discussed earlier, using the retransmission parameters enhanced the reliability to a certain extent. However, the highest reliability is still lower than the aim of 100% reliability.

In this section, the reliability in the test-bed will be tested using the protocol-non-related parameters in combination with the retransmission parameters, used in Table 6.6. The protocol-non-related parameters to be used are: The packet sending interval, the packet sending randomization and the payload redundancy.

6.3.1 Sending interval

Using different sending intervals, with 120 packets sent from each sensor to the GW, gave the following results:

NTC	RRC	PRC	Sending interval(sec)	Reliability
1	2	0	8	71.3%
1	2	0	7	71.6%
1	2	0	6	71.8%
1	2	0	5	73.8%
1	2	0	4	60.9%
1	2	0	3	51.5%
1	2	0	2	40.3%

Table 6.7: Reliability in the test-bed using different sending intervals, with NTC = 1 and RRC = 2.

As seen in Table 6.7, the higher the sending interval is the higher the reliability, but to a specific extent. However, the reliability decreases noticeably as the sending interval decreases. The reason for that could be a higher packet collision at the GW and the relays as more packets reach them, almost at the same time, when the sending interval is low. Thus, the GW and the relays could be not able to process all the incoming packets, and some packets could be dropped. But what if the packets do not reach the GW and the relays at the same time?

6.3.2 The packet sending randomization

Different intervals for the random start time and the back-off time were tested. As a result, an interval of 0 - 10 seconds was used for both the random start time and the back-off time. The used interval was chosen based on that the reliability, for different sending intervals, is higher when using this interval compared to when using smaller intervals.

Conducting the test on the test-bed using the packet sending randomization as explained in subsection 5.3.1, with 120 packets sent from the sensors to the GW gave the following results:

NTC	RRC	PRC	Reliability
1	2	0	97.3 %
2	1	0	99.7%
2	2	0	98.4%

Table 6.8: Reliability in the test-bed using the sending packet randomization, with different retransmission parameters and different sending intervals.

Using the packet sending randomization boosted the reliability in the test-bed significantly. As illustrated in Table 6.8, the reliability increased to 97.3% when NTC = 1 and RRC = 2, while it is 99.7% when NTC = 2 and RRC = 1 and 98.4% when NTC = 2 and RRC = 2. This is a big enhancement compared to when utilizing the retransmission parameters only.

However, there still a place for improvement despite getting a reliability of 99.7% because this reliability was achieved only when NTC = 1 and RRC = 2. For the other retransmission parameters combinations, the reliability was 97.3% and 98.4%. Could payload redundancy be the solution for getting a reliability of 100%?

6.3.3 Payload redundancy

Conducting the tests on the test-bed using the payload redundancy described in subsection 6.3.3, with the packet sending randomization and 120 packets sent from the sensors to the GW, gave the following:

NTC	RRC	PRC	Reliability
1	1	0	99.7%
1	2	0	99.7%
2	1	0	99.99%
2	2	0	99.7%

Table 6.9: Reliability in the test-bed using the payload redundancy, the packet sending randomization, with different retransmission parameters and different sending intervals.

The results in Table 6.9 were achieved using different packet sending intervals in the range [1-8] seconds. As seen in the Table 6.9, the reliability in the test-bed raised when using the packet redundancy implementation. A minimum reliability of 99.7% was achieved regardless the combinations of NTC and RRC as long as they are not zero.

6.4 Testing more combinations of different parameters

Using the payload redundancy and the packet sending randomization with retransmission parameters gave high reliability. But, what parameter(s) had the highest effect on reliability, the protocol-related or the protocol-non-related parameters?

To answer this question, many tests were carried out on the test-bed. The tests were first run using only the protocol-non-related parameters, i.e. the payload

NTC	RRC	PRC	Reliability
0	0	0	98.1%
1	0	0	99.7%
2	0	0	99.94%
3	0	0	99.99%
4	0	0	99.94%
0	1	0	99.7%
0	2	0	99.7%
0	3	0	98.7%
0	4	0	93.2%
0	0	1	95.1%
0	0	2	94.1%
0	0	3	90.4%

Table 6.10: Reliability in the test-bed using the payload redundancy, the packet sending randomization, with different retransmission parameters where only one parameter is used at time, and different sending intervals.

redundancy and the packet sending randomization[5], then using NTC, RRC or PRC to see how the packet delivery ratio is affected.

Table 6.10 illustrates that using only the protocol-non-related parameters boosted the reliability from 63.4% to 98.1%. On the other hand, using NTC and RRC increased the reliability from 63.4% to 73.8%, Table 6.4.

Table 6.10 shows also that using NTC only gave a minimum reliability of 99.7%. Increasing the NTC gave slightly better reliability to the contrary of increasing RRC which gave a higher reliability when RRC = 1, 2 but a lower reliability when RRC = 3, 4. On the other hand, using PRC decreased the overall reliability. The higher the PRC, the lower the reliability.

With PRC = NTC = RRC = 0, sending intervals = 5 seconds, 120 packets, and the relays from Figure 6.4, the achieved reliability was:

- 94.9% when using only the packet sending randomization.
- 83.4% when using only the payload redundancy.

[5]The sending interval has almost no effect on reliability when the other two protocol-non-related parameters are used

Chapter 7
Discussion

The overall goal of this thesis is to explore how reliable BTM is, in terms of packet loss, when it is used in IoT data sensor networks. This includes the exploration of protocol-related and protocol-non-related parameters and its effect on the packet delivery ratio. To do so, many tests were conducted on the test-bed and results were gained. These results will be discussed in this chapter.

7.1 Relays selection

Packets sent from many sensors did not reach the GW without relays. Without having relays in the test-bed, the GW did not receive packets from approximately 30% of the sensors despite that it got packets form a higher number of sensors compared to the other nodes. However, by using relays, the GW got packets from all the sensors in the test-bed, and the packet delivery ratio was 63.4%. Hence, choosing the relays plays a major role in maintaining a connection between the sensors and the GW.

Relay selection was not discussed in many previous works; only [DSLA20] mentioned that relays were chosen from the powered nodes in the open areas like corridors. [DSLA20] concluded that the best reliability performance, higher than 99.9%, was achieved when 1.5% relays were implemented in 1000 m^2. Implementing this conclusion on the test-bed by using 2 or 3 relays, Figure 6.2 and Figure 6.3, did not give a reliability higher than 99.9%. Packets from many sensors did not reach the network. Thus, the number of relays increased gradually till the GW got packets from all the sensors in the test-bed. However, the reliability in the network was only 63.4% after choosing 16 relays when no other parameters are used.

7.2 Retransmission parameters, the packet sending randomization and the payload redundancy

Using the retransmission parameters only had an impact on the reliability. The reliability drops when using only PRC, while when utilizing only NTC or RRC it rises to reach a peak and then it declines. Figure 7.1 shows that reliability goes up to 69.7% and 70.4% when NTC = 2 and RRC = 2 respectively. It shows also that the reliability goes down to 51.5% when PRC = 2.

When using the retransmission parameters in combination with the protocol-non-related parameters, the packet sending randomization and the payload redundancy, the retransmission parameters had also an impact on the reliability. Using PRC decreases the reliability, NTC increases it and RRC increases it to reach a peak then it starts declining, Figure 7.2.

Using combinations of NTC, RRC and PRC, without the protocol-non-related parameters, had also an impact on the reliability. Like when it used without NTC and RRC, PRC caused a decrement between 11% - 20% in reliability. On the other hand, a combination of NTC and RRC gave better reliability, namely 73.8%. However, when using the NTC and RRC with the packet sending randomization and the payload redundancy, a minimum reliability of 99.7% was achieved and the best reliability, namely 99.99%, was achieved when NTC = 2 and RRC = 1. The same reliability was achieved when NTC = 3 and RRC = PRC = 0.

Despite that there is a marginal difference between 99.99% and 99.7%, this difference says something about the packet collision at the GW and the relays, and the traffic in the network. Using NTC will increase the traffic on the *first hop*, i.e. from the sensor to the relay node. This is because if a packet reaches a relay node, the relay checks its cache before forwarding the packet. If this packet was already forwarded, it will not be forwarded again, i.e. it will be dropped. Hence, no matter how many replicas of a packet are sent from the network layer of a sensor to a relay, only one replica will be forwarded as long as all the replicas have the same sequence number.

When it comes to RRC, the traffic could be increased on the *second hop*, i.e. from the relay to the other relays, because a packet could be forwarded till it reaches a relay which has already forwarded it before, or till its TTL is 1. This could lead to a higher traffic in the network and thus a higher packet collision at the GW and the relays, resulting in a lower reliability which is not the case when using only NTC.

Based on the discussion above, using PRC could increase the traffic on both the first and the subsequent hop as the model layer sends "x" replicas of the packet. This means that traffic in the network could be "x" times doubled compared to when

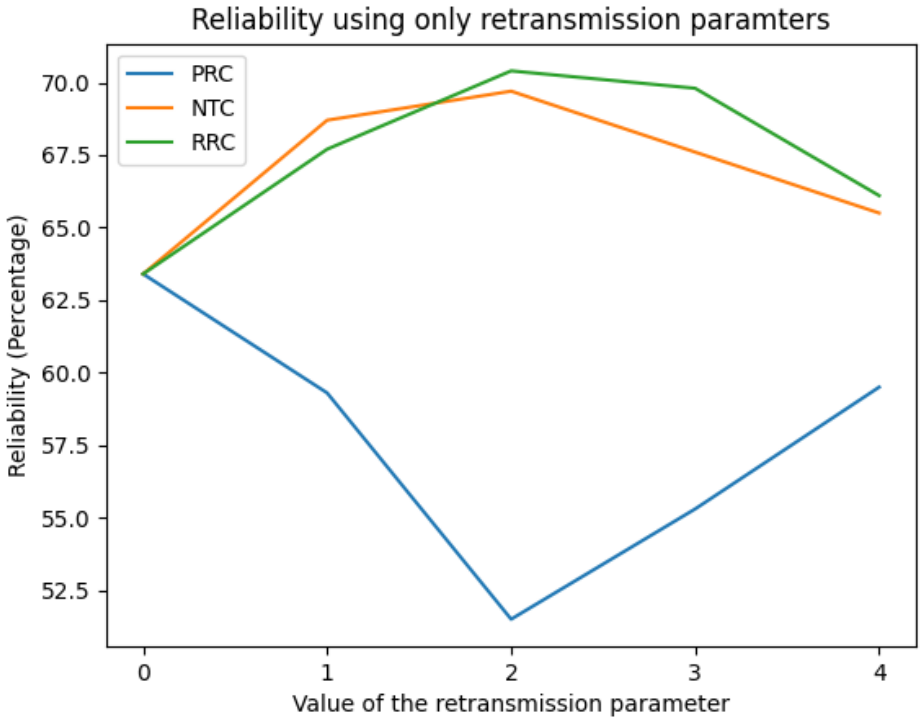

Figure 7.1: Reliability in the test-bed using only retransmission parameters. It shows how reliability is affected when using NTC, RRC or PRC.

PRC = 0, i.e. the model layer sends only one packet. Having NTC = y and RRC = z in addition, could double the traffic "y" times and "z" times on the first and the second hop respectively. This could result in more traffic in the network causing packet drop and collision at the GW and the relays resulting in a lower reliability.

Hence, less traffic in the network could enhance the reliability. Thus, using the packet sending randomization and the payload redundancy without the retransmission parameters gave as high reliability as 98.1%. The reason for that can be less packet collision at the GW and the relays as packets reach them at different times and the GW and the relays are able to process all the incoming packets, hence fewer packets are dropped. When using NTC with the packet sending randomization and the payload redundancy, the reliability was slightly higher than when using RRC meaning that the traffic in the network is higher when using RRC compared to when using NTC causing a bit higher packet collision at the GW and the relays.

However, increasing the traffic in the network using NTC or/and RRC gave reliability ranging from 98.7% to 99.99%. Thus, using these two parameters increases

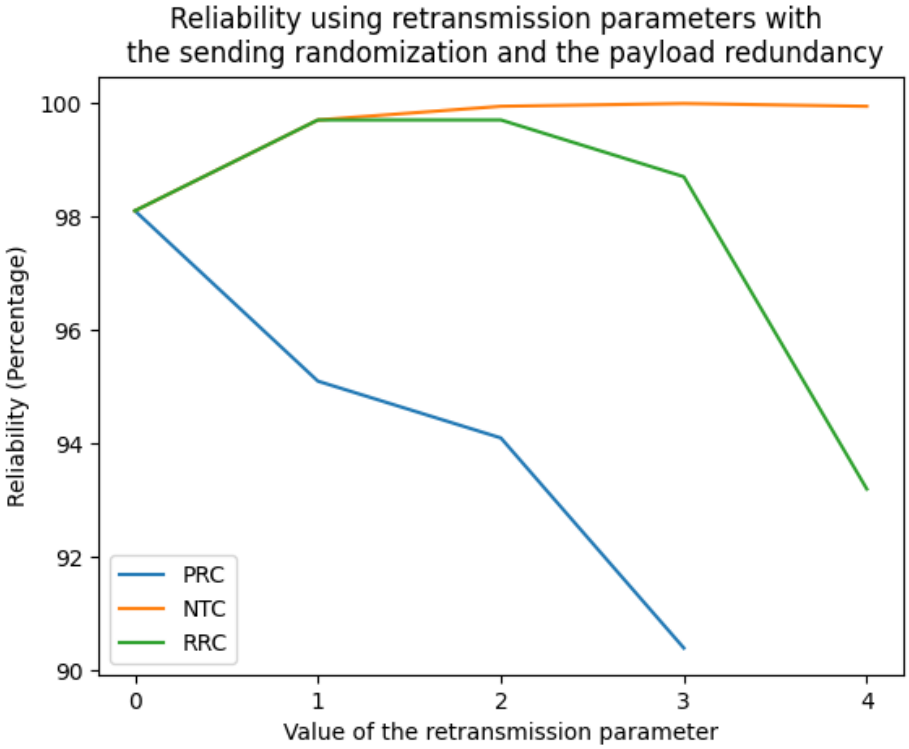

Figure 7.2: The impact of retransmission parameters on reliability when using the packet randomization and the payload redundancy. It shows how reliability is affected when using NTC, RRC or PRC.

the reliability despite increasing the traffic in the network. The reason for that could be the high random start time and the high back-off time meaning that in spite of the higher traffic, the GW and the relays are still able to process all the packets that reach them because these packets reach at different times. One more reason could be that even if some packets could be lost because of the packet collision, these packets are compensated as the their payload is carried by the next packets sent from the same source.

The reliability was 94.9% and 83.4% when using only the packet sending randomization and the payload redundancy respectively, while it was 98.1% when using these two parameters. Thus, the the packet sending randomization and/ or the payload redundancy have a higher effect on the reliability in the network than the retransmission parameters.

The results were obtained when running the test multiple times with a duration

of 10 minutes and sending interval of 5 seconds for each run i.e 120 packets. However, after using the packet sending randomization and the payload redundancy with NTC and RRC tuned and PRC = 0, the reliability was always higher than 99% regardless the duration of the test, whether it is 10 minutes or multiple hours.

Compared to previous works; [DSLA20] and [Lab] , the same reliability was achieved; 99.99%. However, none of these works studied the effect of each retransmission parameter and each protocol-non-related parameter which was studied in this thesis on the reliability. [DSLA20] used the first hop packet repetitions, NTC, and randomization of the advertising triplet. [Lab] used NTC = RRC = 3 which they describe as typical values for these two parameters, while the NTC interval and RRC interval were 10 ms. However, [Lab] do not discuss the effect of these two parameters on reliability.

7.3 Sending interval

Increasing the traffic, by using NTC and RRC, when using the packet sending randomization and the payload redundancy did not affect the reliability. This was also the case when using different sending intervals as it had an effect on the reliability only when the packet sending randomization and the payload redundancy were not used. However, when these two parameters were used, using any sending interval (Higher or equal to 1 second) did not affect the reliability. Thus, sending interval has no impact on the reliability when the packet sending randomization and the payload redundancy are set up properly. As mentioned earlier, none of the previous works studied the effect of the sending interval on reliability.

Chapter 8
Conclusion

The Bluetooth Mesh technology, when using unsegmented packets, has proven to be a reliable technology that can be used in actual large-scale IoT sensor data networks in office environment.

The reliability, in terms of packet loss i.e. the packet delivery ratio, of Bluetooth Mesh networks depends on how relays, the protocol-non-related parameters like the randomization of the packet sending and the payload redundancy, and the retransmission parameters are chosen. The key factors for better reliability are: Relays, the packet sending randomization and the packet payload redundancy.

Choosing the relays in these networks is the main key factor. The relays should be chosen in a way where packets from any node can reach any other node in the network, and at the same time the traffic in the network is as low as possible.

The packet sending randomization and the packet payload redundancy is the second most important factor. Using these parameters has a bigger effect on reliability than using retransmission parameters. Using these two parameters does not increase the traffic in the network. Thus, the packet collision at the gateway and the relays is minimal resulting in fewer packets dropped. Additionally, if some packets are lost or dropped, they are compensated by the next coming packets.

The retransmission parameters have an effect on reliability too. While PRC has a negative effect, NTC and RRC has a positive effect, but to a certain extent. However, the impact of NTC and RRC on the reliability is much smaller that the impact of the packet sending randomization and the packet payload redundancy.

All in all, Bluetooth Mesh provides a reliability higher than 99.9% reliability in IoT data-collecting networks when using unsegmented packets. In order to get this reliability, the following should be done:

- Firstly, chose a gateway which is located in open areas, like hallways. The gateway can communicate directly with a higher number of sensors compared to any other node in the network.
- Secondly, chose relays also from the open areas. The number of relays and the TTL of the relayed packets should be chosen in a way where packets sent from any node in the network can reach their destinations.
- Thirdly, sensors should send packets to the gateway at random times. Each time a sensor sends a packet, it waits for a random back-off time. Additionally, each packet sent from the sensor except the first packet should have the payload of the previous packet in addition to its own payload.
- Finally, use reransmission parameters. It is not recommended to use PRC. Instead, start by using NTC and see if the reliability increases to be higher than 99%. If it does not, use RRC with or without NTC.

Chapter 9
Future work

Many ideas and suggestions have been seen throughout this thesis.

A future work that could be done is running the same tests on the same test-bed but with taking into consideration the relation between the network traffic and the selection of the relays and TTL, the sending packet randomization and the payload redundancy, and the retransmission parameters. This could be about choosing different parameters for the sensors depending on their placement to the GW, that is, having different TTL, NTC and RRC values for each sensor and relay depending on the distance from the GW and the obstacles between the sensor and the GW.

Another future work could be about finding a better way to choose the relays and/or the gateway in such networks taking into consideration the traffic in the network. This means choosing a minimum set of relays where packets from any sensor can reach the gateway and the traffic in the network is as low as possible.

One more future work that could be done is exploring the impact of different protocol-related and protocol-non-related parameters, like the transmission power of the nodes and the signal interference, on reliability.

The future works mentioned above are all concerned with the BTM networking. Conducting a comparative study of BTM and another wireless technology like Thread on the test-bed could be another future work. The aim of the work could be to compare BTM and Thread performance in building data-gathering wireless sensor networks.

References

[ASLM21] A. Aijaz, A. Stanoev, *et al.*, «Demystifying the performance of bluetooth mesh: Experimental evaluation and optimization», Jun. 2021.

[BRSH18] M. Baert, J. Rossey, *et al.*, «The bluetooth mesh standard: An overview and experimental evaluation», *Sensors*, vol. 18, Jul. 2018.

[BT] The Bluetooth website, https://www.bluetooth.com/blog/exploring-bluetooth-5-how-fast-can-it-be/, visited on 2022-05-06.

[CDBF19] A. Cilfone, L. Davoli, *et al.*, «Wireless mesh networking: An iot-oriented perspective survey on relevant technologies», *Future Internet*, vol. 11, no. 99, 2019.

[DSLA20] P. Di Marco, P. Skillermark, *et al.*, «Bluetooth mesh networking», *Ericsson*, Sep. 2020.

[Ericsson] The Ericsson website, https://www.ericsson.com/en/about-us, visited on 2022-05-04.

[Gro19] M. W. Group, «Mesh profile (mesh)», Jan. 2019.

[HPG+20] Á. Hernández-Solana, D. Pérez-Díaz-De-Cerio, *et al.*, «Bluetooth mesh analysis, issues, and challenges», *IEEE Access*, vol. 8, 2020.

[IEEE-Ethernet] The IEEE website, https://standards.ieee.org/ieee/802.3bt/6749/, visited on 2022-05-06.

[Lab] S. Labs, «An1137: Bluetooth mesh network performance. rev. 0.2»,

[LN20] E. Leon and M. Nabi, «An experimental performance evaluation of bluetooth mesh technology for monitoring applications», pp. 1–6, May 2020.

[MCUboot] The MCUboot website, https://docs.mcuboot.com/, visited on 2022-05-06.

[Mesh-NW] The Bluetooth website, https://www.bluetooth.com/learn-about-bluetooth/recent-enhancements/mesh/, visited on 2022-04-29.

[nCS] The Github website, https://github.com/nrfconnect/sdk-nrf, visited on 2022-05-06.

[NOD] The Nordic Semiconductor website, https://www.nordicsemi.com/About-us, visited on 2022-04-27.

[nrf-Connect] The Nordic Semiconductor website, https://developer.nordicsemi.com/nRF_Connect_SDK/doc/latest/nrf/, visited on 2022-05-06.

[nRF52840] The Nordic Semiconductor website, https://www.nordicsemi.com/Products/nRF52840, visited on 2022-05-05.

[Qaq21] H. Qaq, «Performance evaluation of bluetooth mesh network in building environment, specialization project», Nov. 2021.

[RGL17] R. Rondon, M. Gidlund, and K. Landernäs, «Evaluating bluetooth low energy suitability for time-critical industrial iot applications», *International Journal of Wireless Information Networks*, vol. 24, Sep. 2017.

[RMGG20] R. Rondón, A. Mahmood, *et al.*, «Understanding the performance of bluetooth mesh: Reliability, delay, and scalability analysis», *IEEE Internet of Things Journal*, vol. 7, no. 3, 2020.

[Si-Labs] The Silicon Labs website, https://www.silabs.com/about-us, visited on 2022-05-04.

[VBPP17] A.-V. Vladuta, I. Bica, *et al.*, «Reliable data collection for wireless sensor networks using unmanned aerial vehicles», in Jan. 2017, pp. 323–337.

[wiznet] The Wiznet website, https://www.wiznet.io/product-item/w5500/, visited on 2022-05-05.

[Zephyr-Nordic] The Github website, hhttps://github.com/nrfconnect/sdk-zephyr/blob/main/subsys/bluetooth/mesh/Kconfig, visited on 2022-05-07.

(Appendix)
An Experimental Study for Reliable Sensor Data Gathering in a Bluetooth Mesh Network

Hasan Qaq[a], Omkar Kulkarni[b], Yuming Jiang[a]
[a]NTNU, Norwegian University of Science and Technology, Trondheim, Norway
[b]Nordic Semiconductor, Trondheim, Norway

***Abstract*—Bluetooth mesh is a recent addition to the IoT connectivity landscape. It provides a simple and efficient wireless networking solution. Bluetooth mesh provides mesh connectivity to many nodes without any coordination. This paper evaluates the performance of Bluetooth mesh technology in IoT data-gathering networks in an office environment. Through carrying out multiple tests on an actual data-collecting network, the reliability of this network, in terms of packet loss i.e. packet delivery ratio, is evaluated during the analysis of the effect of some protocol-related and protocol-non-related parameters. The protocol-related parameters include the Publish Re-transmit Count (PRC), the Network Transmit Count (NTC) and the relay Re-transmit Count (RRC). The protocol-non-related parameters include the message sending interval, the message sending randomization, and the message payload redundancy. Results show that when using unsegmented messages, a high degree of reliability is achieved when the PRC, NTC and RRC values are tuned, and when the relays and some non-protocol-related parameters, i.e. the message sending randomization and the payload redundancy, in the network are chosen properly.**

***Index Terms*—IoT, Bluetooth mesh, Bluetooth Low Energy, Data gathering networks, Wireless sensor networks**

I. Introduction

Several wireless communication technologies are available today for deploying Internet of Things (IoT) applications. Some examples of these technologies include Thread and Bluetooth Low Energy (BLE). The ultra-low power consumption of BLE [1] makes it a good option for power-limited devices and has even demonstrated the capability to meet the tight Quality-of-Service (QoS) requirements of single-hop real-time industrial applications [1].

Bluetooth mesh (BTM) is a recent technology developed by the Bluetooth Special Internet Group (Bluetooth SIG), released in June 2017. Using the existing BLE protocol stack, BTM allows many-to-many device communication over BLE radio [2]. The BTM functionality is entirely dependent on the BLE protocol stack. Using BTM, thousands of devices can be connected to create a mesh network, with greatly enhancing coverage.

Since BTM was introduced in the market, not many studies have been performed evaluating the BTM technology in IoT networks so far. These studies were either based on simulation and not an actual network like [3] or based on real-world experiments like [4] and [5] but examining the latency and reliability of BTM networks based only on the protocol parameters. However, this paper focuses on a wider exploration of how BTM networking can be used for data collecting in IoT networks. This includes the exploration of some BTM protocol-related parameters namely the Publish Re-transmit Count (PRC), the Network Transmit Count (NTC) and the relay Re-transmit Count (RRC), and some protocol-non-related parameters namely the message sending interval, the message sending randomization, and the message payload redundancy. The work includes tests that are conducted on a large-scale test network to determine the reliability in terms of data loss i.e. packet delivery ratio, of BTM networks.

The importance of the reliability assessment in sensor data networks comes from that these networks are used in environments where sensors have functional and operational challenges. The challanges are related to resource limitation e.g. energy supply [6], and to the physical barriers and the noise in the real environment of deployment. Additionally, the sensors are left unattended to do their job properly, efficiently and in real-time. This, in addition to the harsh environment of deployment, can cause packet loss which hamper the network activity. Thus, continuous delivery of data requires reliable data transmission.

This paper reports the results of several tests carried on an actual BTM network consisting of 100 nodes distributed in a 1400 m^2 office area. The nodes are installed on the ceiling plates in the Nordic Semiconductors office. They are roughly evenly distributed throughout the office floor and were distributed in this way to simulate lights inside the office, Figure 1.One node in the network works a collector or a gateway where the other nodes, i.e. sensors, periodically generate data packets and propagate them, using BTM, to the gateway. The gateway sends the content of the received packets over the Ethernet to a dedicated PC for further analysis. The results are analyzed to investigate how the different protocol-related and protocol-non-related parameters affect the packet delivery ratio, hereafter referred to as reliability, and how to achieve high reliability in this network. The results show that Bluetooth Mesh provides 99.99% reliability in IoT data-collecting networks when using unsegmented packets. In order to get this reliability, the following should be done:

- Firstly, choose a gateway which is located in open areas, like corridors. The gateway has a higher number of responses from the sensors compared to any other node

Fig. 1. Nodes placement in the office hallway

in the network.

- Secondly, choose relays also from the open areas. The number of relays and the TTL of messages should be chosen in a way where messages sent from any node in the network can reach their destinations.
- Thirdly, sensors should send messages to the gateway at random times. Each time a sensor sends a packet, it waits for a random back-off time. Additionally, each packet sent from the sensor except the first one, should have the payload of the previous packet in addition to its own payload.
- Finally, use NTC and/ or RRC. It is not recommended to use PRC. Start by using NTC and see if the reliability increases to be higher than 99%. If it does not, use RRC with or without NTC.

The remainder of this paper is organized as follows. Section II shortly introduces the BTM technology. Section III presents the test setup and the test environment. The experimental results and discussions are presented in Section IV. Section V includes the conclusion with summary remarks.

II. Bluetooth Mesh

BTM enables the creation of large-scale low-data-rate short-range wireless network [7]. The mesh devices communicate directly with each other without the use of a centralized controller. A device that is a part of a mesh network and able to transmit and receive packets in this mesh network is called a *node* or a *provisioned device*, and that which is not is called *unprovisioned devices*.

To exchange the data between nodes, BTM follows a publish/subscribe paradigm. The sender *publishes* a packet to a certain address. Node(s) that are interested in receiving the packet will *subscribe* to that address. In a BTM packet, the maximum size of the Transport Protocol Data Unit (Transport-PDU), i.e. payload, is either 12 or 16 bytes. If the payload of a message is higher than the maximum size of TransportPDU, then the message will be divided into *segments* where the

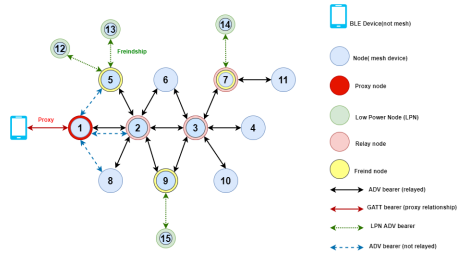

Fig. 2. Nodes types in a BTM network.

maximum size of each segment does not exceed the maximum size of the TransportPDU. Otherwise, the message is called *unsegmented*.

The functionality of a BTM node can be determined by the features it supports. A BTM node can optionally support one or more additional features [8]. There are 4 features that a node can support; namely relay, proxy, low power and friend, Figure 2. The relay node forwards messages to other nodes in the network. The proxy nodes enables the communication between a non-mesh-device and the BTM network. The friend node forwards messages to and from the Low Power Node (LPN) to the rest of the network.

To relay packets, BTM uses managed flooding. Managed flooding is a process between flooding and routing where the relay node broadcasts packets to all nodes within its radio range. In managed flooding when a packet is relayed, the Time-To-Live (TTL) that is assigned to it gets decremented by one with each re-transmission. A packet is only re-transmitted when its TTL is greater than one. Additionally, the packets are cached. Therefore, a received packet that already exists in the cache of the relay node is automatically discarded and not re-transmitted.

As a way of providing an appropriate level of reliability in a mesh network, the Bluetooth Mesh Profile [8] allows for messages repetitions at the application and the network layer. These repetitions of the BTM messages can be controlled by three parameters: PRC, NTC and RRC. PRC controls the number of replicas of a packet published by a model where each replica has an unique sequence number. NTC controls the number of message transmissions of the Network PDU originating from the node, while RRC controls the number of message re-transmissions of the Network PDU relayed by the node.

III. Related works

[5] deliver a performance evaluation of BTM. This includes the testing of latency, reliability and scalability in BTM networks. The testing was conducted over actual wireless devices in test networks. The testing network consisted of 192 nodes in a 2230 m^2 office area with active wireless networks, like Wi-Fi, in range. Several tests were conducted using different payloads with both segmented and unsegmented messaging. The results show that the reliability of BTM networks is

greater than 99% when the application payload can fit in a single packet, i.e. using unsegmented messages, and the relay count RRC is low.

[3] was carried out through simulation where no specifics were given regarding the virtual environment. The performance evaluation was conducted on a large-scale office scenario: 879 devices in an area of approximately 2000 m2. The network has one or more Gateway. The sensors send unacknowledged messages to the GW which forwards these messages to the cloud for further analysis. The relays in the network are selected only from the powered nodes which are located in open areas like corridors. The results show that the best reliability performance was higher than 99.9%. [3] concludes that the best results were obtained when only 6 (1.5%) relays were deployed in a 1000 m^2 area.

[2] performs an evaluation of the Quality-of-Service performance and scalability of BTM. The tests were carried out through extensive simulations. The test-bed consists of 28 nodes spread around an office environment in which interference data, which are collected from a real office environment, was recorded. The results show that BTM is particularly vulnerable to network congestion and increased packet collision probability for densely populated deployments. The randomization of the timing parameters of the protocol proved to be a solid starting point to counteract network congestion. The performance of BTM is not notably affected by interference from other BTM devices in the network. However, BTM is still considerably susceptible to interference from other common wireless technologies, such as WLAN.

IV. The test environment and structure

The Nordic Semiconductors test network, herefater referred to as test-bed, consists of 100 kits installed on an area of 1400 m^2 in Nordic Semiconductors office in Trondheim. Each kit is a custom-made internal Powered Over Ethernet (POE) development board labeled POE dev board. The POE dev board was specially designed for the test network, and it is not a commercial product. It consists among other things of nRF52840 chip and Wiznet W5500 chip. The kits are connected to three stacked POE switches connected to a dedicated computer, hereafter referred to as the remote PC. The test setup can be controlled by using the remote PC.

The firmware of all the kits is updated using the same code. This is called Device Firmware Update (DFU). The test environment has many walls made of concrete or other opaque objects. Many employees work in the testing environment. The test setup experiences external interference from Wi-Fi access points and other Bluetooth devices operating on the floor.

The GW is the node that is located in open areas like hallways, and receives higher number of responses compared to the other nodes in the test-bed. The relays are also chosen from nodes in open areas where they have direct connection to each other i.e. there are no barriers between them. Each node in the test-bed is covered by at least one relay. Without relays in the test-bed, the GW did not receive messages from approximately 30% of the sensors despite that it gets messages

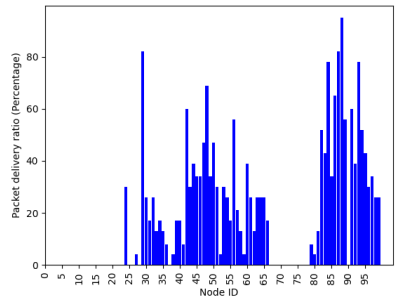

Fig. 3. Responses from all the sensors in the test-bed sent to node 90 with no relays.

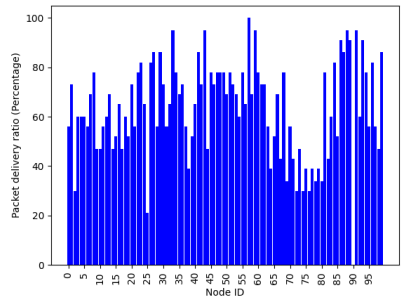

Fig. 4. Responses from all the sensors to the GW using relays.

form a higher number of sensors compared to other nodes, Figure 3.

However, by using relays with TTL = 6, the GW got messages from all the sensors in the test-bed, and the packet delivery ratio was 63.4%. This reliability was achieved using only relays without any of the protocol-related or protocol-non-related parameters mentioned above. Hence, choosing the relays plays a major role in maintaining a connection between the sensors and the GW, Figure 4. Figure 5 shows the GW and the relays used in the tests.

The tests that are carried out on the test-bed work as follows: The remote PC sends a command over the Ethernet to all the nodes in the test-bed. The command specifies which node is the gateway (GW), the sending interval (How often the sensors send a packet to the GW) and the test duration. Upon receiving this command, the sensors simultaneously start sending unacknowledged and unsegmented BTM messages to the GW simulating that many lights are switched on at the same time. These BTM messages has an available payload of 11 bytes; the original available payload is 12 bytes, but 1 byte is used for the model opcode. Of these 11 bytes, a 2-bytes field is used as a unique message tag, and a 2-bytes field is used to simulate a random value measured by the sensor. Thus, the

Fig. 5. The sensors, relays, and the gateway in the test-bed. Each circle with a number inside represents the approximate placement of each node and the number represents the node ID.

PRC	0	1	2	3	4
Reliability	63.4 %	59.3%	51.5%	55.3%	59.5%
NTC	0	1	2	3	4
Reliability	63.4 %	68.7%	69.7%	67.6%	65.5%
RRC	0	1	2	3	4
Reliability	63.4%	67,7%	70.4%	69.8%	66.1%

TABLE I
RELIABILITY IN THE TEST-BED USING ONLY RELAYS AND PRC

NTC	RRC	PRC	Reliability
1	1	0	69.3%
1	1	1	49.1%
1	2	0	73.8%
1	2	1	62.5%
1	3	0	65.6%
1	3	1	51.5%
2	1	0	69.5%
2	2	0	70%
2	3	0	67.1%
3	1	0	63.8%
3	2	0	66.8%
3	3	0	57.3%

TABLE II
RELIABILITY IN THE TEST-BED USING RELAYS AND DIFFERENT COMBINATIONS OF RE-TRANSMISSION PARAMETERS.

unused part of the message payload is 7 bytes.

When the GW receives the BTM messages from the sensors, it logs the content of each message and the sender's address over the Ethernet to the remote PC. The test script on the remote PC parses these message to identify which sensor sent what. When the test is over, each sensor logs the number of the BTM messages it sent to the GW over the Ethernet to the test script. The test script calculates the ratio, for each sensor, between the logged messages from the GW and the logged messages from sensors. This ratio is the packet delivery ratio, i.e. the reliability.

The tests are conducted in a way where different parameters will be used firstly in isolation, from other parameters, and then in combination with other parameters to see the effect of each parameter on the reliability. The parameters that have a positive effect on the reliability when they are used in isolation, will be tested later in combinations. The parameters to be tested: PRC, NTC, RRC, sending interval, message sending randomization and payload redundancy. The message sending randomization means that the sensors send packets to the GW asynchronously . The payload redundancy means that every packet has its payload and the payload of the previous packet. The transmission power for each node in the network is fixed to 0 dBm, the sending interval is 5 sec, 120 messages, and the rest of the nodes configuration options, except for NTC and RRC, are according to the Zephyr-Nordic Semiconductors implementation of BTM [9]. This includes the NTC interval and the RRC interval which are fixed to 20 ms.

V. RESULTS AND ANALYSIS

In this section the results for different experiments are presented and discussed.

A. The protocol-related parameters

The protocol-related parameters, hereafter referred to as the re-transmission parameters, covered in this paper are: PRC, NTC, and RRC. The effect of these parameters on the reliability, in isolation and in combination, was tested. Table **??** shows the reliability when each parameter was tested in isolation. For PRC, the PRC interval was zero i.e. messages from the model layer are sent at the same time. For NTC and RRC the interval was 20 ms.

Table I shows that using PRC decreases the reliability. With PRC = 2 the reliability drops by more than 12%, compared to the reliability when PRC = 0, then it increases as PRC increases but still lower than what is when PRC = 0. NTC or RRC increase reliability. The reliability reaches its peak when NTC = 2. However, increasing the NTC after that decreases the reliability which is still better than what it is when NTC = 0. On the other hand, the reliability reaches its peak when RRC = 2. Like NTC, increasing the RRC after that decreases the reliability which still be better than what it is when RRC = 0.

Based on the results from Table I, combinations of both NTC and RRC, whose values gave better reliability, are tested to see how the reliability would be affected. Thus, PRC will be tested with some combinations of NTC and RRC to see if it still gives a negative effect on the packet delivery ratio. Table II shows that the reliability increases when using a combination of NTC and RRC but only when PRC = 0. Hence, using only PRC or using PRC with the other two re-transmission parameters decreases the reliability.

Table II also shows that the highest reliability in the test-bed was reached when NTC = 1, RRC = 2 and PRC = 0. However, reliability is still not high enough, and as it is seen in the table, increasing NTC and RRC did not increase the

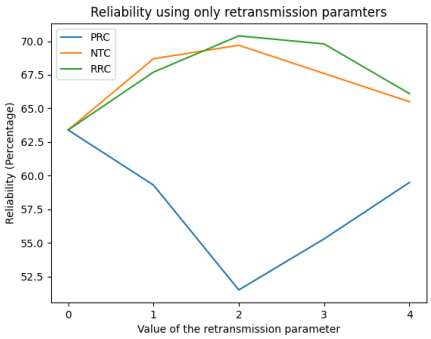

Fig. 6. Reliability in the test-bed using only re-transmission parameters

NTC	RRC	PRC	Sending interval(sec)	Reliability
1	2	0	8	71.3%
1	2	0	7	71.6%
1	2	0	6	71.8%
1	2	0	5	73.8%
1	2	0	4	60.9%
1	2	0	3	51.5%
1	2	0	2	40.3%

TABLE III
RELIABILITY IN THE TEST-BED USING DIFFERENT SENDING INTERVALS.

NTC	RRC	PRC	Reliability
0	0	0	94.9 %
1	2	0	97.3 %
2	1	0	99.7%
2	2	0	98.4%

TABLE IV
RELIABILITY IN THE TEST-BED USING THE SENDING MESSAGE RANDOMIZATION

reliability.

Using the re-transmission parameters only had an impact on the reliability. The reliability drops when using only PRC, while when utilizing only NTC or RRC it rises to reach a peak and then it declines. Figure 6 shows that reliability goes up to 69.7% and 70.4% when NTC = 2 and RRC = 2 respectively. It shows also that the reliability goes down to 51.5% when PRC = 2.

Using combinations of NTC, RRC and PRC, without the protocol-non-related parameters, had also an impact on the reliability. Like when it used without NTC and RRC, PRC caused a decrement between 11% - 20% in reliability. On the other hand, a combination of NTC and RRC gave better reliability, namely 73.8%.

When using the re-transmission parameters in combination with the protocol-non-related parameters, the message sending randomization and the payload redundancy, the re-transmission parameters had also an impact on the reliability. Using PRC decreases the reliability, NTC increases it, and RRC increases it to reach a peak then it starts declining, 7.

B. The non-protocol-related parameters

Using the re-transmission parameters enhanced the reliability to a certain extent. However, the highest reliability is still lower than the aim of 100% . Thus, the reliability in the test-bed will be tested using non-protocol-related parameters in isolation and then in combination with the NTC and RRC values that increased reliability . The non-protocol-related parameters to be used are: The message sending interval, the message sending randomization and the payload redundancy.

Using different sending intervals gave the following results:

Table III shows that the higher the sending interval is the higher the reliability, but to a specific extent. However, the reliability decreases noticeably as the sending interval decreases. The reason for this could be higher packet collision at the GW and the relays as more packets reach them, almost at the same time, when the sending interval is low. Thus, the GW and the relays could be not able to process all the incoming messages, and some messages could be dropped. But what if the messages do not reach the GW and the relays at the same time?

In all the tests that have been conducted so far, the sensors start sending BTM messages to the GW at the same time, which is the worst case, resulting in packet collision at the GW, hence lower reliability.

Using the packet sending randomization boosted the reliability in the test-bed significantly. As illustrated in Table IV, the reliability increased to 94.9% using only this parameter, and to 97.3% when NTC = 1 and RRC = 2, while it is 99.7% when NTC = 2 and RRC = 1 and 98.4% when NTC = 2 and RRC = 2. This is a big enhancement compared to when utilizing the re-transmission parameters only. However, the reliability 99.7% was achieved only when NTC = 1 and RRC = 2. For the other re-tranmission parameters combinations, the reliability was 97.3% and 98.4%. Could payload redundancy increase the reliability additionally?

The actual payload of a BTM message sent from a sensor to the GW, as explained above, is 4 bytes. Since the payload field of each message has 7 bytes available, another 4 bytes payload can fit in. Thus, the actual payload of each message would be 8 byes; 4 bytes for current message and 4 bytes for the previous message. Using the payload redundancy with the packet sending randomization with NTC and RRC gave the following results:

The results in Table V were achieved using different message sending intervals in the range [1-8] seconds. As seen in the Table V the reliability in the test-bed raised when

NTC	RRC	PRC	Reliability
1	1	0	99.7%
1	2	0	99.7%
2	1	0	99.99%
2	2	0	99.7%

TABLE V
RELIABILITY IN THE TEST-BED USING THE PAYLOAD REDUNDANCY, THE SENDING MESSAGE RANDOMIZATION, WITH DIFFERENT RE-TRANSMISSION PARAMETERS AND DIFFERENT SENDING INTERVALS.

NTC	RRC	PRC	Reliability
0	0	0	98.1%
1	0	0	99.7%
2	0	0	99.94%
3	0	0	99.99%
4	0	0	99.94%
0	1	0	99.7%
0	2	0	99.7%
0	3	0	98.7%
0	4	0	93.2%
0	0	1	95.1%
0	0	2	94.1%
0	0	3	90.4%

TABLE VI
RELIABILITY IN THE TEST-BED USING THE PAYLOAD REDUNDANCY, THE SENDING MESSAGE RANDOMIZATION, WITH DIFFERENT RE-TRANSMISSION PARAMETERS, WHERE ONLY ONE PARAMETERS IS USED AT TIME, AND DIFFERENT SENDING INTERVALS.

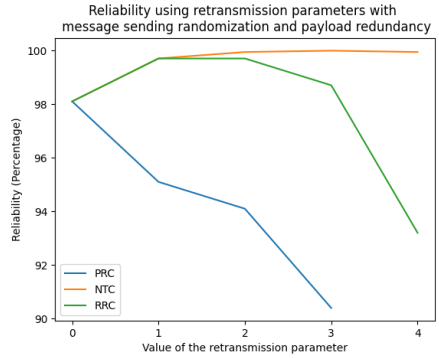

Fig. 7. The impact of re-transmission parameters on reliability when using the sending packet randomization and the payload redundancy. It shows how reliability is affected when using NTC, RRC or PRC.

using the packet redundancy implementation. A minimum reliability of 99.7% was achieved regardless the combinations of NTC and RRC as long as they are not zero. Using the payload redundancy in isolation gave a reliability of 83.4% when the re-transmission parameters and the packet sending randomization were not used.

‘

C. Testing more combinations of different parameters

Using the payload redundancy and the sending message randomization with re-transmission parameters gave very good reliability. But, what parameter(s) had the highest effect on reliability, the protocol-related or protocol-non-related parameters?

Table VI illustrates that using only the packet sending randomization and the payload redundancy[1] parameters are used boosted the reliability from 63.4% to 98.1%. On the other hand, using NTC and RRC increased the reliability from 63.4% to 73.8%, Table III. Table VI shows also that using NTC only gave a minimum reliability of 99.7%. Increasing the NTC gave slightly better reliability to the contrary of increasing RRC which gave higher reliability when RRC = 1, 2 but lower reliability when RRC = 3, 4. On the other hand, using PRC decreased the overall reliability. The higher the PRC, the lower the reliability, Figure 7.

Using the packet sending randomization only gave a reliability of 94.5%, while using the payload redundancy only gave a reliability of 83.4%.

By combining the packet sending randomization, the payload redundancy, NTC and/or RRC the reliability is always higher than 99%. This applies to messages sent from the sensors to the GW and vice versa.

VI. DISCUSSION

Using NTC increases the traffic on the *first hop*, i.e. from the sensor to the relay node. This is because if a message reaches a relay node, the relay checks its cache before forwarding the message. If this message was already forwarded, it will not be forwarded again, i.e. it will be dropped. Hence, no matter how many replicas of a message are sent from the network layer of a sensor to a relay, only one replica will be forwarded as long as all the replicas has the same sequence number.

When it comes to RRC, the traffic could be increased on the *second hop*, i.e. from the relay to the other relays, because a message could be forwarded till it reaches a relay which it has forwarded it before, or till its TTL = 1. This could lead to higher traffic in the network and thus higher packet collision at the GW and the relays, resulting in lower reliability which is not the case when using only NTC.

Based on the discussion above, using PRC could increase the traffic on both the first and the second hop as the model layer sends ”x” replicas of the message. This means that traffic in the network could be ”x” times doubled compared to when PRC = 0, i.e. the model message sends only one message. Having NTC = y and RRC = z in addition, could double the traffic ”y” times and ”z” times on the first and the second hop respectively. This could result in more traffic in the network causing packet drop and collision at the GW and the relays resulting in lower reliability.

Hence, less traffic in the network could enhance the reliability. Thus, using the message sending randomization and the payload redundancy without the re-transmission parameters, gave as high reliability as 98.1%. The reason for that can be less packet collision at the GW and the relays as messages reach them at different times and they are able to process all the messages, hence fewer messages are dropped. When using NTC with the sending packet randomization and the payload redundancy, the reliability was slightly higher than when using RRC meaning that the traffic in the network is higher when using RRC compared to when using NTC causing a bit higher packet collision at the GW and the relays.

However, increasing the traffic in the network using NTC or/and RRC gave reliability ranging 98.7% - 99.99%. Thus, using these two parameters increases the reliability despite

[1]The sending interval has almost no effect on reliability when the other two protocol-non-related

increasing the traffic in the network. The reason for that could be the high random start time and the high back-off time meaning that in spite of the higher traffic, the GW and the relays are still able to process all the messages that reach it because these messages reach it at different times. One more reason could be that even if some messages could be lost because of the packet collision at the GW, these messages are compensated as the their payload is carried by next messages sent from the same source.

The reliability was 94.9% and 83.4% when using only the message sending randomization and the payload redundancy respectively, while it was 98.1% when using these two parameters. Thus, the the message sending randomization and/or the payload redundancy have higher effect on the reliability in the network than the re-transmission parameters.

The results were obtained when running the test multiple times with a duration of 10 minutes and sending interval of 5 sec for each run i.e 120 messages. However, after using the message sending randomization and the payload redundancy with NTC and RRC tuned, the reliability was always higher than 99% regardless the duration of the test, whether it is 10 minutes or multiple hours.

VII. Conclusion

The Bluetooth Mesh technology, when using unsegmented packets, has proven to be a reliable technology that can be used in actual large-scale IoT sensor data networks in office environment.

The reliability, in terms of packet loss i.e. the packet delivery ratio, of Bluetooth Mesh networks depends on how relays, the protocol-non-related parameters like the randomization of the packet sending and the payload redundancy, and the re-transmission parameters are chosen. The key factors for better reliability are: Relays, the message sending randomization and the packet payload redundancy.

Choosing the relays in these networks is the main key factor. The relays should be chosen in a way where packets from any node can reach any other node in the network, and at the same time the traffic in the network is as low as possible.

The message sending randomization and the packet payload redundancy is the second most important factor. Using these parameters has a bigger effect on reliability than using re-transmission parameters. Using these two parameters does not increase the traffic in the network. Thus, the packet collision at the gateway and the relays is minimal resulting in fewer packets dropped. Additionally, if some packets are lost or dropped, they are compensated by the next coming packets.

The re-transmission parameters have an effect on reliability too. While prc has a negative effect, ntc and rrc has a positive effect, but to a certain extent. However, the impact of ntc and rrc on reliability is much smaller that the impact of the message sending randomization and the message payload redundancy.

References

[1] R. Rondon, M. Gidlund, and K. Landernäs, "Evaluating bluetooth low energy suitability for time-critical industrial iot applications," *International Journal of Wireless Information Networks*, vol. 24, 09 2017.

[2] R. Rondón, A. Mahmood, S. Grimaldi, and M. Gidlund, "Understanding the performance of bluetooth mesh: Reliability, delay, and scalability analysis," *IEEE Internet of Things Journal*, vol. 7, no. 3, 2020.

[3] P. Di Marco, P. Skillermark, A. Larmo, and P. Arvidson, "Bluetooth mesh networking," *Ericsson*, 09 2020.

[4] E. Leon and M. Nabi, "An experimental performance evaluation of bluetooth mesh technology for monitoring applications," pp. 1–6, 05 2020.

[5] S. Labs, "An1137: Bluetooth mesh network performance. rev. 0.2."

[6] A.-V. Vladuta, I. Bica, V.-V. Patriciu, and F. Pop, *Reliable Data Collection for Wireless Sensor Networks Using Unmanned Aerial Vehicles*, 01 2017, pp. 323–337.

[7] "The Bluetooth website," https://www.bluetooth.com/learn-about-bluetooth/recent-enhancements/mesh/, visited on 2022-04-29.

[8] M. W. Group, "Mesh profile (mesh)," 01 2019.

[9] "The Github website," hhttps://github.com/nrfconnect/sdk-zephyr/blob/main/subsys/bluetooth/mesh/Kconfig, visited on 2022-05-07.