

Link performance measurement and improvement using Bluetooth mesh network

DURGAPRASAD SRINIVASA

HARSHAVARDHAN MANJUNATH

MASTER'S THESIS

DEPARTMENT OF ELECTRICAL AND INFORMATION TECHNOLOGY

FACULTY OF ENGINEERING | LTH | LUND UNIVERSITY



Link performance measurement and improvement using Bluetooth mesh network

Durgaprasad Srinivasa , Harshavardhan Manjunath
du2602sr-s@student.lu.se , ma5552gg-s@student.lu.se

Department of Electrical and Information Technology
Lund University

Supervisor: Michael Lentmaier(LTH),
Peter Karlsson(u-blox),
Magnus Persson(u-blox)

Examiner: Fredrik Rusek

November 26, 2020

Abstract

Bluetooth is a short-range technology, it helps with wireless communication between many electronic devices. Bluetooth low energy (BLE) is the advanced version of Bluetooth which uses low energy compared to the standard Bluetooth. Bluetooth mesh technology play a crucial role in the Internet of things (IoT). The IoT mesh technology enables many-to-many and one-to-many communication. This project deals with creating a Bluetooth mesh network model using MATLAB as a tool to replicate the readings obtained from real-world measurements. The purpose of this model is to give an overview of how the nodes in the Bluetooth mesh network will work by giving performance such as Round trip time, Time to live and throughput of each individual node, this in turn will help a consumer who will purchase the Bluetooth nodes for the mesh network about their performance and number of nodes that would be required for a given area with optimal performance.

Popular Science Summary

“Bluetooth”, it has become very common and has been used widely. Bluetooth technology has developed very much and recently there is a new part of Bluetooth technology, which is the Bluetooth Mesh Network. The Bluetooth Mesh Network is similar to wireless mesh networks used in offices or industries to connect all the computers to the internet, but, here in Bluetooth mesh network, it is used to send small information from time to time like the temperature in the building or for some applications like switching lights on and off. Bluetooth Mesh forms a crucial part of IoT.

The structure of the project was to measure the latency of nodes in the Bluetooth Mesh network. The measurements are carried out with the help of Bluetooth modules. This Bluetooth Mesh Network is for an office of area 50mx60m. The measurements are undertaken with the help of 4 test points (Bluetooth modules explained later in the report) which are placed at 4 corners of the office, there are 30 nodes spread out randomly in the office. These test points send a random message to each node and the node responds with an acknowledgment, this, in turn, finds the RTT (Round Trip Time) between the test point and the node, this measurement is carried out for all 30 nodes from each of the different test points to get different RTT. Before this measurement, we can configure the Time to Live (TTL) for the mesh network and the node properties. To limit these hops to a certain value we set the TTL to 5 (this is considered in the measurements as well). And, the throughput is also measured, but in this case, it is measured only between the test points. There are two different types of nodes used in here three different background traffic types, therefore this gives us 6 different measurement values. This measurement is carried out using the C-Sharp platform. Our task was to create a theoretical Bluetooth Mesh model using MATLAB based on the readings obtained from the measurement.

The main aim of the project to give an idea of how the mesh network works and can be used as a base to figure out the number of nodes that would be required for any given area and for the given latency requirement for a particular application which is achieved by the simulation model.

Acknowledgement

We would first like to thank our thesis supervisor Michael Lentmaier and examiner Fredrik Rusek of the faculty of engineering at Lund University. The door to Prof. Lentmaier office was always open whenever we ran into a trouble spot or had a question about our research or writing. He steered us in the right direction whenever he thought we needed it.

We would also like to thank the experts who were involved and helped us in this project at u-blox: Peter Karlsson, Magnus Persson and Pär-Olof Håkansson. Without their passionate participation and input, this project would not have been successfully conducted.

Finally, we must express our very profound gratitude to all our companions for providing us with unfailing support and continuous encouragement throughout our years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Acronyms and abbreviations

ATT	Attribute Profile
EVK	Evaluation Kit
GATT	Generic Attribute Profile
HCI	Host Control Interface
IoT	Internet of Things
ISM	Industrial, Scientific and Medical
L2CAP	Logical Link Control and Adoption Protocol
LPN	Low Power Node
PDU	Protocol Data Unit
RTT	Round Trip Time
SEQ	Sequence Number
SIG	Special Interest Group
SMP	Security Manager Protocol
TTL	Time To Live

Table of Contents

1	Introduction	1
1.1	Background and motivation	1
1.2	Project aims and main challenges	2
1.3	Approach and methodology	2
1.4	Literature survey	3
2	Bluetooth low energy and Bluetooth mesh network overview	5
2.1	Bluetooth low energy	5
2.2	Bluetooth mesh concepts	7
2.3	Bluetooth mesh nodes	11
3	Bluetooth mesh architecture and security	15
3.1	Overview of Bluetooth mesh architecture	15
3.2	Bluetooth mesh protocol	17
3.3	Security of Bluetooth mesh network	19
4	Methodology and measurement of Bluetooth mesh network	21
4.1	Proposed mesh network layout	21
4.2	Considered parameters for the Bluetooth mesh network	22
4.3	Scenarios considered	23
4.4	Measurements	25
5	Simulation model and results	29
5.1	Layout of the nodes in mesh network	29
5.2	Implementation	29
5.3	Results	32
6	Conclusion	45
	References	49
A	Extra Materials	51
A.1	AT commands	51
A.2	OP-CODE used for measurements, specified by Bluetooth SIG	53

List of Figures

1.1	Bluetooth mesh network. [19]	1
2.1	Channel distribution in Bluetooth Low Energy.[5]	5
2.2	Advertising mode in Bluetooth Low Energy.[6]	6
2.3	Connection mode in Bluetooth Low Energy.[7]	6
2.4	Bluetooth mesh standard through a publish/subscribe model. [8]	7
2.5	Node Composition. [8]	8
2.6	Provisioning of a new device for Bluetooth mesh. [10]	10
2.7	Different features of nodes. [20]	11
3.1	Bluetooth mesh network architecture. [11]	15
3.2	Bluetooth mesh protocol stacks.[15]	18
4.1	Bluetooth mesh network layout	21
4.2	Round trip time of Bluetooth mesh network	22
4.3	Throughput of Bluetooth mesh network	23
4.4	EVK NINA-B3 evaluation board	26
4.5	NINA-B3 microcontroller	26
4.6	Virtual-Here [14]	27
4.7	S-Center	27
5.1	Simulated layout of Bluetooth mesh network	30
5.2	ALLTreon-off RTT vs distance(Measurement)	33
5.3	ALLTreon-off RTT vs distance(Simulation model)	33
5.4	BestGuess-off RTT vs distance(Measurement)	34
5.5	BestGuess-off RTT vs distance(Simulation model)	34
5.6	ALLTreon-1second RTT vs distance(Measurement)	35
5.7	ALLTreon-1second RTT vs distance(Simulation model)	35
5.8	BestGuess-1second RTT vs distance(Measurement)	36
5.9	BestGuess-1second RTT vs distance(Simulation model)	36
5.10	ALLTreon-10seconds RTT vs distance(Measurement)	37
5.11	ALLTreon-10seconds RTT vs distance(Simulation model)	37
5.12	BestGuess-10seconds RTT vs distance(Measurement)	38
5.13	BestGuess-10seconds RTT vs distance(Simulation model)	38

5.14	ALLTreon TTL vs distance(Measurement)	40
5.15	ALLTreon TTL vs distance(Simulation model)	40
5.16	BestGuess TTL vs distance(Measurement)	41
5.17	BestGuess TTL vs distance(Simulation model)	41
5.18	Throughput comparison of measurement vs model	42
5.19	Scenario of map with reduced nodes	43
5.20	Alltreon scenario with reduced nodes(Simulation model)	44
5.21	Bestguess scenario with reduced nodes(Simulation model)	44
A.1	AT-Command (UBTMMOD)	51
A.2	AT-Command (UBTMPAD)	51
A.3	AT-Command (UBTMCCB)	52
A.4	AT-Command (UBTMRPUB)	52
A.5	Server reply (UUBTMRCV)	52

List of Tables

4.1	Measurement traffic models	23
5.1	Average RTT for measured values	39
5.2	Average RTT for simulation model values	39
5.3	Average RTT retransmission for measured values	39
5.4	Average RTT retransmission for simulation model values	39
5.5	Average RTT values	43
A.1	Messages name and its opcode	53

Technology advancement in Bluetooth has made tremendous improvements. The Bluetooth mesh was first introduced in 4.0, in which it makes difference between normal Bluetooth and low energy Bluetooth. The Bluetooth LE will save energy and serves a good purpose for the Internet of things (IoT) and this has been implemented in many applications. Bluetooth 4.0 is the source for the Bluetooth mesh, Bluetooth 5.0 is the advanced version of the Bluetooth Low Energy and with the more specification and also supports Bluetooth mesh.

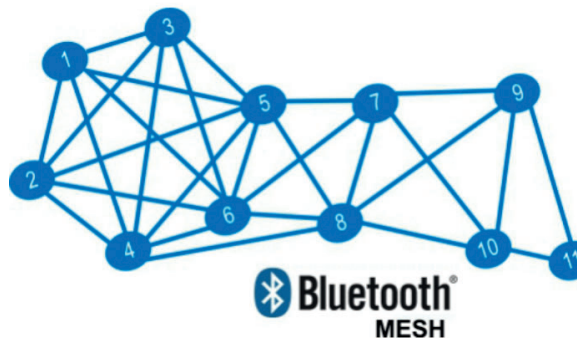


Figure 1.1: Bluetooth mesh network. [19]

Bluetooth mesh enables many-to-many communication between different nodes. These different nodes in the network will communicate with each other directly or via multi-hop communication. This mesh technology is used in different applications like IoT, building automation, sensor networks, and so on.

1.1 Background and motivation

IoT has increased drastically and is driving the growth of connected devices to an estimated value of 30 billion devices by 2020 [4]. Short-range technologies are typically operating on the license-free industrial, scientific, and medical (ISM) frequency bands. IoT devices can be used to monitor and control the mechanical, electrical, and electronic systems used in various types of buildings in home automation and building automation system. With devices such as IEEE 802.15.4

based ZigBee and thread technology, Bluetooth LE, etc., are easily disturbed by the other networks on the 2.4GHz bands.

Bluetooth LE is the low-power variant of classic Bluetooth which operates in the same 2.4 GHz ISM band and uses frequencies between 2402 and 2480 MHz. Bluetooth LE offers two modes of communication between devices: advertising mode or connection-oriented mode. Both communication techniques have their use and both of them can be used to realize Bluetooth mesh networking. However, the Bluetooth special interest group (SIG) has chosen to use the advertising mode as the core underlying technique for the Bluetooth mesh standard.

The communication in a Bluetooth mesh network uses a flooding mechanism, which makes sure that all the nodes in the network repeat themselves so that they are relayed further until the destination is reached. The standard uses a new type of BLE advertisement packet to communicate in a mesh network, which is only supported by devices that support both Bluetooth LE and Bluetooth mesh. Fortunately, the standard also defines a backward compatibility feature to ensure that BLE devices that do not support Bluetooth mesh can also be part of a Bluetooth mesh network.

1.2 Project aims and main challenges

The main aim of this master's thesis is to study the working of indoor Bluetooth mesh and its layers by building a theoretical model and to obtain the results like RTT (round trip time), TTL (time to live), throughput, and an approximate distance. This theoretical model is then compared with the previously measured results. We differentiate this model by transmitting packets of different lengths (maximum and minimum). Nodes are placed according to the floor plan and by using the Bluetooth module (NINA-B3), which supports Bluetooth 5.0 and mesh technology we measure above mentioned parameters. Based on the results we obtain, we may try to modify and improve the theoretical model for better performance.

The main challenges are to replicate the real-life scenario in the theoretical model and obtain values that match the measured values.

1.3 Approach and methodology

We initially build a theoretical model based on the floor plan using MATLAB and evaluate the key performance parameters such as RTT, TTL, and throughput. We compare the values which are obtained from the theoretical model with the parameter values provided to us.

In this method, we are going to consider the flooding technique for the communication between Bluetooth nodes. With this model, we differentiate the performance by sending messages of different lengths and we examine the parameter values to check for variations.

1.4 Literature survey

After a various literature search, we were able to find a few papers which were related to our work and which gave us more insight into what we were doing and the challenges and problems that we were going to face during this project.

Utilizing Bluetooth LE as an underlying technology to implement mesh networks has gained such importance in recent years. The paper [1] presents a complete background on Bluetooth mesh network, such as its network layer, how the communication takes place between nodes in a closed environment, the effects of RTT, TTL, and so on. And also showing the evaluation of Bluetooth Mesh performance utilizing three approaches, namely an experimental evaluation, a statistical approach, and a graph-based simulation model, which can be used as the basis for our thesis work. Paper [2] depicts how problems can be caused in a mesh network in commercial or industrial buildings such as offices, factories with a high density of Wi-Fi networks and other wireless equipment, microwave ovens, etc. In particular, [2] shows the mechanism (flooding) of how messages are transmitted in a Bluetooth mesh network and also showing the effects of RTT by transmitting packets of different sizes (Bytes). Paper [3] shows the auto-configuring and topology mapping in which any low energy Bluetooth protocol can be utilized.

We are given the values of parameters that were obtained from previous summer work and we hold these values as a reference and to obtain measurements similar to these values.

Bluetooth low energy and Bluetooth mesh network overview

In this chapter we discuss the Bluetooth low energy and the concepts of Bluetooth mesh network. Several concepts are introduced and explained in depth about the Bluetooth mesh network concept.

2.1 Bluetooth low energy

Bluetooth LE is first introduced in Bluetooth version 4.0 by Bluetooth SIG, which is also known as Bluetooth smart. Bluetooth LE uses the simple modulation techniques and also works on the same ISM band of normal Bluetooth which is 2.4 GHz. The frequency lies between 2402 Mhz - 2480 Mhz. It is used in health care, fitness, IoT, and home automation. Bluetooth LE has over the air data rate of 125 kbps - 1 Mbps - 2 Mbps, enables the single hop, star topology network, and provides low energy consumption.

Bluetooth LE spectrum has several channels which are divided into 2 Mhz each with the spacing of 2 Mhz, this gives 40 channels. The channels of Bluetooth LE is shown in the figure below. It has an advertisement channel of 37, 38, and 39, a total of 37 main channels that are used for the connection. Bluetooth Low Energy works in two modes: connection-oriented mode and advertising mode. Figure 2.1 show the channel distribution in Bluetooth Low Energy.

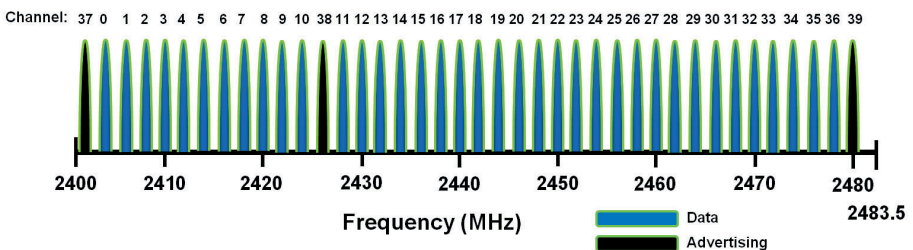
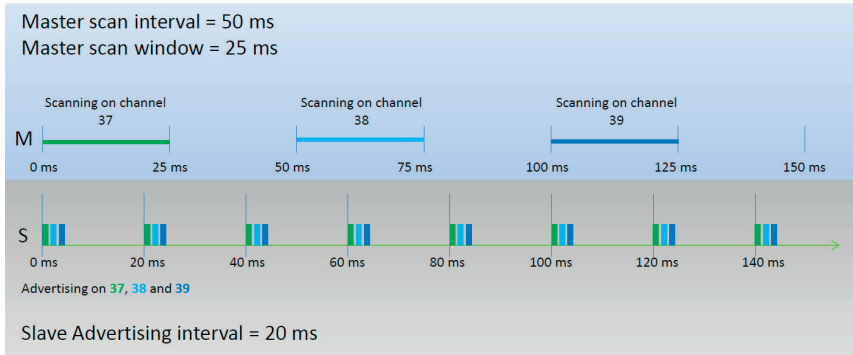


Figure 2.1: Channel distribution in Bluetooth Low Energy.[5]

In the advertising mode, the Bluetooth LE send full packets to the nearby devices to receive and process from the three advertising channels. In this process,

the advertiser will transmit data and the other receiver device will scan for the advertiser which is to receive the transmitted information. The advertising mode is explained in the Figure 2.2 below.

BLE Advertising



From this we can see that the following advertising packets will be picked up by the scanning device: $t=0$ ch=37, $t=20$ ch=37, $t=60$ ch=38, $t=100$ ch=39 and $t=120$ ch=39.

Figure 2.2: Advertising mode in Bluetooth Low Energy.[6]

In the connection mode, the remaining 37 channels are used to send data packets. This mode follows the master-slave operation with the time-division multiplexing (TDMA) scheme. The data is sent on a different channel with a certain interval. The connection mode is explained in the Figure 2.3 given below.

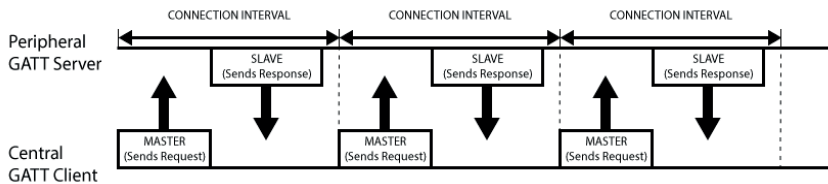


Figure 2.3: Connection mode in Bluetooth Low Energy.[7]

Both the techniques explained above can be used for the Bluetooth LE operations. In advertising mode, we can configure each node to scanning and broadcast the incoming message for further network. The connection mode depends on the central and peripheral at the same time. Depending on the application each has its own best results. As suggested by the Bluetooth special interest group (SIG) advertising mode is the most suitable mode for Bluetooth mesh standard.

2.2 Bluetooth mesh concepts

Bluetooth mesh standards depend on the subscribe and publish model. In this the publish will transmit the required data and then the subscriber can subscribe to one or more data by its requirement. The publish and subscribe model is illustrated in the Figure 2.4 below. Here in Bluetooth mesh network we have three address types which are unicast address, multicast address, and virtual address. The unicast address identifies the single node in which the unique address and this unicast address are given to each node in the network. A multicast or group address describes the group of nodes in the network. These are names as relay nodes, friendly nodes, and proxy nodes, these nodes are explained later in this report. A virtual address is assigned to one or more nodes in the network, these virtual addresses are configured in the manufacturing process. [7]

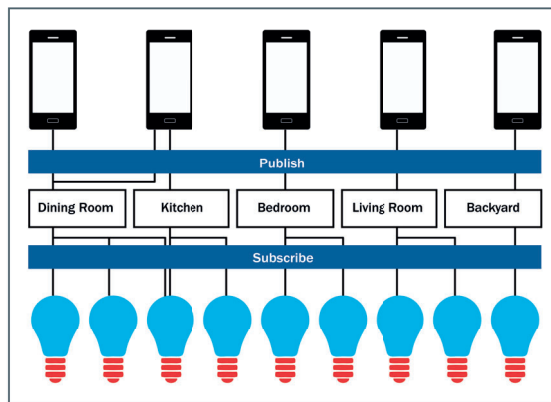


Figure 2.4: Bluetooth mesh standard through a publish/subscribe model. [8]

In the Figure 2.4 illustrated above, the device 1 will publish the group address to the dining room, light 1, light 2, and light 3 are subscribed to the dining room, i.e the device 1 will operate all the 3 lights which are connected to the dining room. Likewise, device 2 is published to the group's dining room and kitchen, light 3 and light 4 are subscribed to the kitchen, i.e device 2 can handle lights from 1 to 4 in both the groups.

The use of group addresses with the publish/subscribe model has the advantage of adding, removing, and re-configuring the other nodes. The new nodes can be added with the help of the provisioning process and configure the new nodes to subscribe to the previously configured groups, from this other nodes and network won't affect. The normal relay nodes, have the radio-enabled all the time and always in the listening mode, which leads to power consumption.

2.2.1 States and properties

The state is defined as the value of a particular type, which is in the server. States have their behavior. For example, if a light in one area either in on or off state.

The state of generic on/off will be reflected on light to be switched on/off. [7]

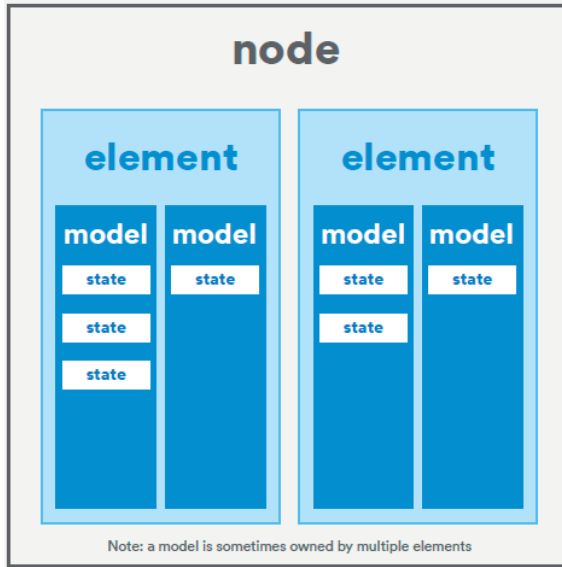


Figure 2.5: Node Composition. [8]

The property provides the context for interpreting characteristics. Use of context which is related to properties is explained in the example, an 8-bit temperature state type with some associated properties, which also includes present indoor ambient temperature and present outdoor ambient temperature. These two properties make a sensor to publish sensor readings in a way that allows a receiving client to determine the context of the temperature value. Properties are of two categories, which are read-only and read-write.

2.2.2 Messages, States and properties

Operation in a mesh network is done through messages which are circulated in the Bluetooth mesh network. A message represents an operation on a state value or collection of multiple states' values. All messages are of three types on which the Bluetooth mesh network supports. The three types of messages are GET, SET, and STATUS.

GET messages request from one or more nodes for the value of a given state in the network. To get the required response to a GET and also have relevant state value a STATUS message is sent. To change the value of the given state in the network we use SET message. Acknowledged SET messages in the network will be returned as a STATUS message in response to the SET message, but unacknowledged SET messages need not to get a response. In response to GET message STATUS messages are sent in the network which contains the relevant state value.

2.2.3 State transition and bound state

Changing from one state to another state is called the transition of states. Transitions are of two types, it may be instantaneous or it takes time to execute, this time is called transition time. State transition affects the application layer depending on the nodes.

When there is a change in one state and this change is affected to another state, this is the relationship lies between the states, this relationship is called state binding. one state can be bind to many other states. For example, consider a light controlled by a dimmer switch. The light would possess the two states, *generic On/Off* and *generic level* with each bound to the other. Reducing the brightness of the light until generic level has a value of zero fully dimmed results in generic of transitioning from On to Off.

2.2.4 Models

Models will define the functionality and characteristics of an element which is related to the mesh network by taking all the preceding concepts. There are three categories of the model, which are server model, client model, and control models.

The collection of states is defined in the server model, also the server model will take care of state transition, state binding, and messages of an element that contains the model may transmit or receive. This server takes care of the behavior of messages, states, and state transitions.

The client model does not elaborate on any states. It will define the server model messages, in which it will transmit or receive to GET, SET, or know the STATUS of states.

The control model has both client and server model, it allows a server model to communicate with other client models, and also it allows a client model to communicate with the other server model. By extending the models we can create the new models, a model that is not extended is called the root model. Adding or removing the behavior of the model is not possible. The only way to create a new model is by extending the existing model.

2.2.5 Generics

In the Bluetooth mesh network, there are several types of devices that use generic states and generic messages. Bluetooth mesh model defines many generic states such as generic on/off and generic level. Generic states and generic levels are utilized by the generalized model and generic server models.

Generics are used to create new models in the Bluetooth mesh network, it has a wide range of devices that support the network. As mentioned before, the models can be created by extending other models, generic models have the basis for creating the new devices quickly.

2.2.6 Scenes

A scene is stored in different types of states which can be called at a specific time and also for a particular type of message. It is a 16-bit numbers, they are all

unique in the Bluetooth mesh network. For example, you want to have a certain temperature and certain kind of lighting at home for a party, you can store these settings as scenes by using a configuration application and can recall this scene whenever it is needed. It can be recalled by sending scene-related mesh messages or automatically at a scheduled time.

2.2.7 Provisioning

In the paper [9], provisioning is the process of the device becoming a node by joining a Bluetooth mesh network. This process involves many processes and it is a secured process. Provisioning is accomplished by devices like tablets and these types of devices are called the provisioner. The provisioning steps are mentioned below [10].

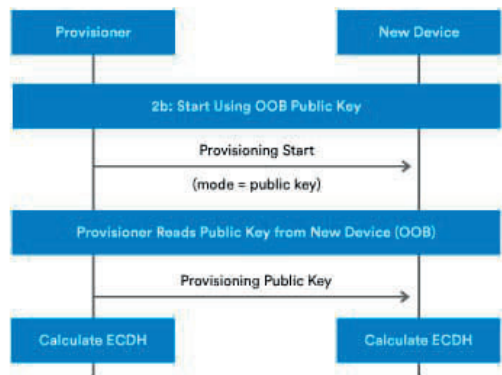


Figure 2.6: Provisioning of a new device for Bluetooth mesh. [10]

Beaconing

Beaconing is a type of advertisement process, in this process, the unprovisioned device will advertise its availability by using "Mesh Beacon" AD type in the advertising packets. Mesh beacons use a version of Bluetooth that came out last December that turn the current generation of location-broadcasting beacons into a two-way, net-connected network. For example, users can make a Bluetooth device to search for pairing for another Bluetooth device by using the pairing option.

Invitation

In this step, the provisioner will send an invite to the device which is to be provisioned in the form of Protocol Data Unit (PDU). The beaconing device will send a reply with the information about itself and its availability.

Exchanging public keys

In this step, both the beaconing device and the provisioner will exchange the public keys, which is ephemeral or static. This process of exchanging key is done by the out-of-band (OOB) method.

Authentication

In the authentication process, the device which is to be paired (to be provisioned) gives out the random numbers, it may be single or multiple digit numbers with a suitable action. For example, while authentication the device to be paired will blink LED to show that the device is ready for authentication and the cryptography communication takes place in between the user device and provisioned device, this will complete the authentication.

Distribution of provisioning data

After completing the authentication process a session key is obtained by each of the two devices from their private keys and the exchanged, peer public keys. This session key is used to complete the provisioning process, this will protect the subsequent distribution of data which includes the security key, also known as a network (NetKey). The provisioned device possesses the NetKey, IV Index (mesh security parameter), and unicast address, all these are given to the provisioner. This whole thing forms a node. The provisioning process is explained in the Figure 2.5.

2.3 Bluetooth mesh nodes

Nodes in the Bluetooth mesh network will receive and transmit the messages as discussed before. There are several types of nodes that have their capability, application, and working procedure. There are four different features in nodes for Bluetooth mesh networks, which are a relay, proxy, friend, Low-power and End nodes. These features can be enabled and disabled any time in the network.

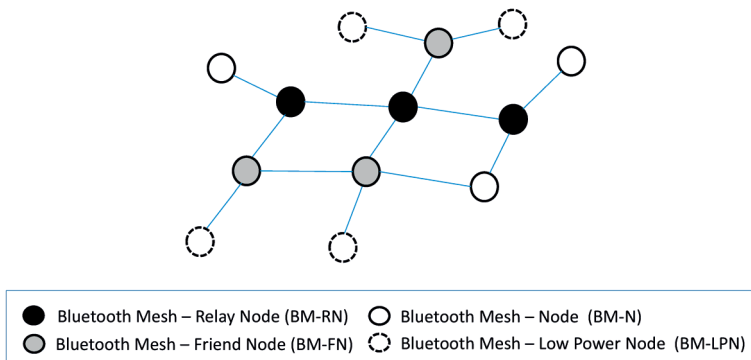


Figure 2.7: Different features of nodes. [20]

2.3.1 Relay Nodes

Relay nodes are the nodes that support relay features. These nodes will re-transmit the packets which are received. By this relaying process, the message can travel through the whole Bluetooth mesh network by taking the multiple hops between relay nodes. The Bluetooth mesh network has the PDU called the TTL (Time To Live). This TTL is used to limit the number of hops for a packet to travel from source to destination in the network. For example, if TTL is set to 3 then the packets take 3 hops to reach from source to destination, if TTL is set to 0 then there will be no hops and also no relay of the packets, packets take a direct link from source to destination. The TTL Relay nodes can be effectively used in the Bluetooth mesh network. TTL will be explained in Chapter 4.

2.3.2 Proxy Nodes

As we can see in the present world many devices support Bluetooth Low Energy. These devices connect with the help of the GATT, the generic attribute profile. The GATT helps two Bluetooth Low Energy devices to communicate with each other with the help of service and characteristics.

Proxy nodes use the GATT interface for its process in the BLE network. A Proxy Protocol, this protocol is used with a connection-oriented bearer, such as GATT is defined in the device. The devices which use GATT will read and write Proxy Protocol PDUs from within GATT characteristics implemented by the Proxy node. The Proxy node transforms these PDUs to/from mesh PDUs. These proxy nodes will allow BLE devices to interact with the mesh network even though they do not have the Bluetooth mesh stack.

2.3.3 Low Power Nodes and Friend Nodes

Messages when the temperature is above or below the given threshold. If the temperature is in the threshold, nodes remain silent. The nodes used in temperature sensing are referred to as low power nodes (LPN). These LPN are used in many applications like motion sensors, temperature monitoring, etc.

Low power nodes will work connected to another node that has a permanent power supply. Another node in which LPN is connected is called the friend node. This friend node will store the data or messages which is required by the LPN, when the LPN will poll the friend nodes, the data which is stored in the friend nodes are sent to the LPN in order by using a flag known as MD (More Data), this MD indicates if there is more messages or data to be sent from friend nodes to LPN. As mentioned in section 2.2, the relay nodes consume more power, the low power nodes play a crucial in Bluetooth mesh network as it consumes less power.

This relationship of the friend nodes and low power nodes is called friendship. The friendship will allow LPN to receive the messages when it is required to be active in the Bluetooth mesh network with very little power consumption.

2.3.4 End Nodes

End nodes are the nodes that are present in the Bluetooth mesh network, these nodes will receive the message and do not re-transmit that message, the message will be ended in that node.

2.3.5 Nodes Configuration

The nodes which are used in the Bluetooth mesh network are implemented within the configuration server model and they are accessed using the configuration client model. The configuration server state will indicate features such as a relay, proxy, low power, and friend, subscription list has all the addresses in which the nodes are subscribed. The application keys are in the nodes. But, if the nodes are counted in a network or not is indicated by network keys and subnet keys. Configuration messages permit the configuration client model and configuration server model. These models will support GET, SET, and STATUS operations.

Bluetooth mesh architecture and security

In this chapter, the Bluetooth mesh architecture is elaborated with a detailed explanation and the importance of the different layers will be explained. As the application increases in the Bluetooth mesh network, it is obvious for the network to have a secured connection to preserve the data, security becomes an important part of the network. In this chapter, we take a close look at Bluetooth mesh network Security.

3.1 Overview of Bluetooth mesh architecture

The Bluetooth mesh network will follow the layers as shown in the Figure 3.1 below. Bluetooth Mesh has a special layer called Bluetooth Low Energy Core Specification along with Bearer Layer, Network Layer, Lower transport layer, Upper transport layer, Access Layer, Foundation Model Layer, and Model Layer. Bluetooth mesh basic is Bluetooth Low Energy, devices in the Bluetooth mesh does not connect as it happens in Bluetooth Low Energy. But, it uses advertising and scanning to transmit the message with the relay feature. [16]

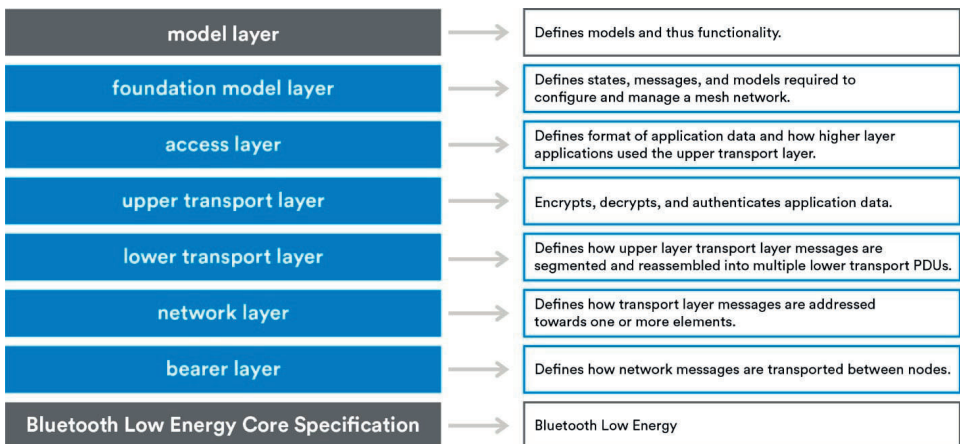


Figure 3.1: Bluetooth mesh network architecture. [11]

3.1.1 Bluetooth Low Energy Core Specifications

As mentioned before the whole architecture has a base of Bluetooth Low Energy layer, this is not just a single layer to be specified it is more than a layer of Bluetooth mesh architecture. This BLE Core Specifications will provide all the support to have the basic wireless communication for the mesh network, it takes the higher position in the architecture. The mesh system is directly dependent on the availability of the Bluetooth Low Energy mesh stack.

3.1.2 Bearer layer

The bearer layer explains how the different mesh message packets(Protocol Data Units or PDU) are transmitted in the network. Two types of bearers are present, they are Advertising bearer and GATT bearer.

In the advertising bearer, the scanning and advertising states of Bluetooth Low Energy devices are used to transmit and receive the mesh message packets

GATT bearer uses a protocol from the proxy node called the Proxy Protocol which performs GATT operations, it allows the devices to communicate indirectly with the nodes in the mesh which does not support the advertising bearer. The GATT operation has the GATT characteristics, as mentioned before the Proxy nodes have these GATT characteristics, it helps both the GATT bearer and the advertising bearer so that the message packets are suitable for both the bearer.

3.1.3 Network layer

In the Network Layer, network message format, and various message address are defined in here, the message format will transport layer PDUs to be transported by the bearer layer. The network layer also implements the relay and proxy features, it supports multiple bearers which has multiple network interfaces. To communicate between the devices which are part of the same node network layer uses the local interface. It will decide the network interface to output messages over. To determine whether the messages from the bearer layer should be delivered to the network layer for the processing. A filter controls the output messages whether to drop or deliver to the bearer layer.

3.1.4 Lower Transport Layer

Sending PDUs from the upper transport layer to the lower transport layer on a peer device is carried out by the Lower transport layer. The longer packets which do not fit into a single transport PDU, the lower transport layer will divide PDUs into multiple Transport PDUs by performing the segmentation process. The receiver lower transport layer will rejoin the segmented PDUs as a single upper transport layer PDU and this is sent to the stack.

3.1.5 Upper Transport layer

The application data passing to and from the access layer is encrypting, decrypting, and authenticated takes place in the upper transport layer. It will take care of

the transport control messages which are related to friendship and heartbeat, these messages are generated inside and sent to different peer nodes of the upper transport layer.

3.1.6 Access layer

This access layer defines how the application layer is responsible for defining how the application can use the upper transport layer. It Defines the application data format. It will control the encryption and decryption process which happens in the upper transport layer. Verify the data which is from the upper transport layer before forwarding the data to the stack.

3.1.7 Foundation Models Layer and Models Layer

The Foundation models layer will handle by implementing the network configuration and network management models. The models' layer will address the implementation of models, which include behaviors, messages, states, and state binding.

3.2 Bluetooth mesh protocol

In the previous section, we discussed the Bluetooth mesh architecture with different layers. Since Mesh network messages are inside the payload of Bluetooth Low Energy advertisement packets. Therefore, it is possible to compare the Bluetooth Low Energy and mesh protocol stacks. The Figure 3.2 shows the Bluetooth mesh protocol stacks.

As depicted in the Figure 3.2 the Bluetooth low energy consists of Generic Attribute Profile (GATT), Attribute Protocol (ATT), Security Manager Protocol, Logical Link Control and Adaption Protocol (L2CAP), Link Layer, and Physical Layer.

GATT is elaborated when two BLE devices exchange data using service and characteristics. It makes use of ATT to store services. Server/client relationship which gives a clear image of GATT. GATT server has the ATT lookup data and service and characteristics definitions and GATT client will send requests to the server. The transactions take place in the GATT client a response is received in the GATT server.

Transferring of data is handled by a low-level layer Attribute protocol. For a device, it identifies its discovery, reading, and writing properties. Generic Attribute Profile gives high-level services to the manufacturer for implementing Bluetooth Low Energy. The main aim of the service is to transfer data with a systematic procedure. For example, GATT defines if a device's role is going to be Server or Client [15].

L2CAP is used in the Bluetooth protocol stack as shown in the figure 3.2. The packets are passed to Host Controller Interface (HCI) or the Link manager. L2CAP also includes the functions like Multiplex the data between different higher layer protocols, Segmentation of packets in a server, and reassembling of packets in clients, for a group of Bluetooth devices it provides the one-way transmission

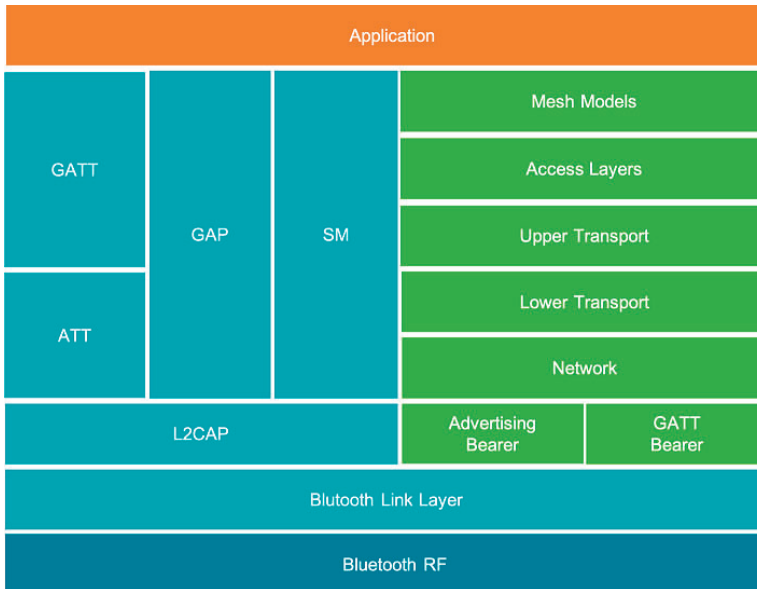


Figure 3.2: Bluetooth mesh protocol stacks.[15]

management of multiple data and L2CAP provides Quality of Service (QOS) for higher-layer protocols.

GAP gives many imports to the Central devices and Peripheral devices and defines various tasks for other devices too. Central devices are high power and memory units like mobile phones and tablets. Peripheral devices are small, low power, and fewer resource devices, these peripheral devices are can be connected to powerful central devices. Peripheral devices are proximity tags.

The Security Manager Protocol (SMP) contains two independent subsystems which are: initiator (SMPI) and responder (SMRP). SMPI will help multiple connections simultaneously by implementing the initiator features of the security manager protocol. SMRP helps for only one connection by implementing the responder feature of the security manager protocol. It also implements the cryptographic toolbox.

The link layers will help to discover the new Bluetooth devices, by assigning one device as a master and one device as a slave, the link-layer will establish the connection between various Bluetooth devices. The link layer will handle the interference, noise, and deep fades. It will broadcast the data by managing the connections between the devices.

As specified by the IEEE 802.15.1 Bluetooth physical layer consists of baseband and radio features. In a small region called the piconet Bluetooth network consists of one master and several slave devices. The master device selects the channel and time to transmit the data packet, this process can be used by other devices in the same piconet. A Bluetooth device in the piconet can act as both master or slave to the other devices in another piconet. The overlapping region is called the scatternet. These processes are all handled in the physical layer of the Bluetooth

mesh network.

3.3 Security of Bluetooth mesh network

As mentioned before the security of the Bluetooth mesh network is so important, in the coming section we have explained the mesh security fundamentals, key security features, and few attacks. The manufacturer of the Bluetooth mesh device will take care of the security required by researching on the threats of the Bluetooth mesh network. The security and protection of the Bluetooth mesh network are mandatory. The network, individual applications, and devices should be secured and these cannot be switched off or decremented for any reason [6].

3.3.1 Fundamentals of Mesh Security

The following fundamental security statements apply to all Bluetooth mesh networks:

1. All mesh messages are encrypted and authenticated.
2. Network security, application security, and device security are addressed independently.
3. Security keys can be changed during the life of the mesh network via a Key Refresh procedure.
4. Message obfuscation makes it difficult to track messages sent within the network providing a privacy mechanism to make it difficult to track nodes.
5. Mesh security protects the network against replay attacks.
6. The process by which devices are added to the mesh network to become nodes is itself secure.
7. Nodes can be removed from the network securely, in a way which prevents trashcan attacks.

3.3.2 Separation of Concerns and Mesh Security Keys

Bluetooth mesh security mainly consists of 3 keys, necessary for mesh security and other crucial functions. To understand in detail let us consider a mesh light act as a relay. As a relay, it can find itself handling messages related to the Bluetooth mesh door and security system. Light has no idea regarding these messages but needs to rely on other nodes. At the network layer mesh uses different security keys to overcome the above conflict. The nodes in the mesh should contain a network key(NetKey) so that it is part of the network. Encryption and the private key are obtained from Netkey, with the help of the Netkey node can decrypt and authenticate to the network layer hence relaying is obtained. The Network is divided into subnets each sub-net has its Netkey. Nodes having a specific Application key (Appkey) can possess decryption of respective application data. Nodes in a mesh possess many Appkeys, but each Appkey belongs to a small subset of nodes

of a particular function. App-keys are used for decryption and authentication of messages before passing them to the access layer. Appkey can work only on one specific network i.e. key binding, but the network can organize various applications. The last type of key is the Device key, only the provisioner device knows this key, is used in the process of secured communication between the node and provisioner.

3.3.3 Nodes Removal, attacks Attacks and privacy

An owner can take a call to sell the node to another owner but he has to make sure the security of the network from which it is taken. Hence, it's necessary to follow up on some methods for detaching node. Initially, a node that has to remove is added to the blacklist from the provisioner application later Key Refresh takes place in which the complete set of security key is replaced but the node which is detached does not contain the new NetKey and AppKey because the Key Refresh process depends on the network. Therefore, the old node does not pose any threat.[13]

NetKey gives rise to other important keys i.e. privacy key main function is to confound some of the crucial information such as source address (PDU header values) and to make sure that the max security is provided so that tracking via passive eavesdropping becomes difficult. Attacks based on traffic analysis is complex.

Replay Attacks

As discussed in the previous section we node contains security keys. A replay attack is one of the kinds of attacks, where valid data is transmitted repeatedly or delayed maliciously due to interrupt by an eavesdropper to track the recipient where the attacking device is unauthorized. For example, various applications use voice recognition to verify the identity. In text-dependent systems, an attacker can record the target individual's speech that was correctly verified by the system, then play the recording again to be verified by the system and access the application so he can track the data in the application. As a counter to protect from this kind of attack Bluetooth uses 2 network PDU fields i.e. SEQ and IV Index. Sequence Number (SEQ) is incremented on publication of the message every time. If the node receives a message from an element with an SEQ value less than or equal to SEQ of the last valid message then its discarded because it's likely to be a Replay attack. IV Index is different, it is valued within messages from a given element which has to be always greater than or equal to the last valid message. [13]

Methodology and measurement of Bluetooth mesh network

To improve the link performance of the Bluetooth mesh network we require a way to achieve it. The methodology being the basic key to the process, it is explained in detail along with the measurements of the Bluetooth mesh network.

4.1 Proposed mesh network layout

Figure 4.1 shows the physical layout of the u-blox office space and the position of each node which are spread out in the area. The configuration of each node is depicted in this figure. (Red and blue dots have no significance)

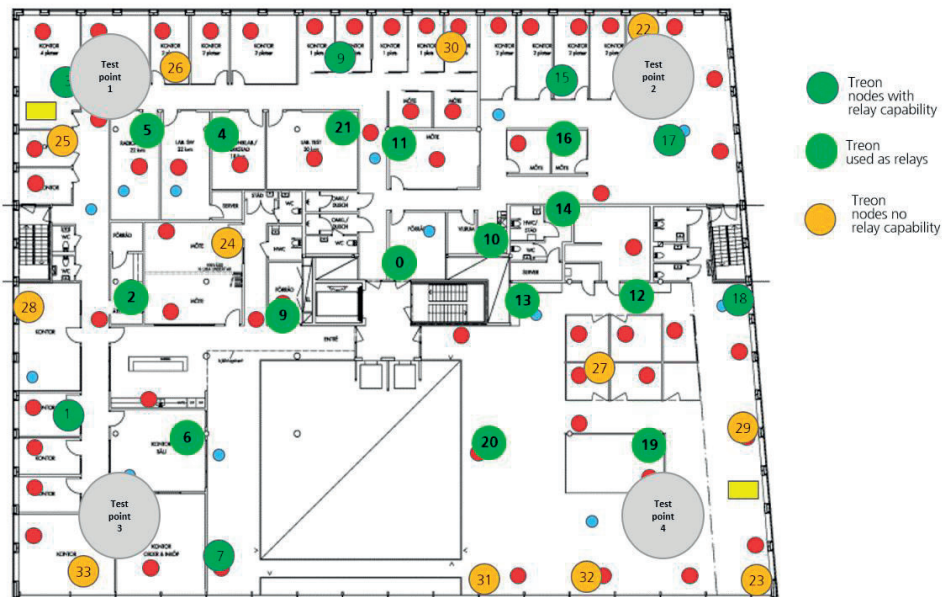


Figure 4.1: Bluetooth mesh network layout

There are 4 test points, each test point is an evaluation kit (EVK) with a NINA-B3 module (explained later in this chapter) connected to a raspberry-pie, which makes each test point available on the network and can be connected by another device. One more EVK, which is connected with the PC/laptop as shown in Figure 4.3, collects all the basic parameters and logs them on the PC/Laptop. In this Bluetooth mesh network, we have a total of 30 nodes in that we have two types of nodes, they are relay and end nodes, based on the requirements we can configure them as the end or relay nodes. Figure 4.1 just depicts the layout of the nodes and not the actual configuration as the configurations are varied based on the measurements.

4.2 Considered parameters for the Bluetooth mesh network

The basic parameters which are considered in the Bluetooth mesh network are hops, TTL, RTT, and throughput.

1. Hops: Number of hops between relay nodes from source to destination.
2. TTL: It is a field in mesh network protocol data units (PDUs). Its purpose is to control whether or not a received message should be relayed and to limit the total number of hops over which a message is ultimately relayed within the network.
3. RTT: The round-trip delay or round-trip time is the length of time it takes for a signal to be sent plus the length of time it takes for an acknowledgment of that signal to be received. This time delay includes the propagation times for the paths between the two communication endpoints. The commands shown in 4.2 are explained in section 4.4.3.

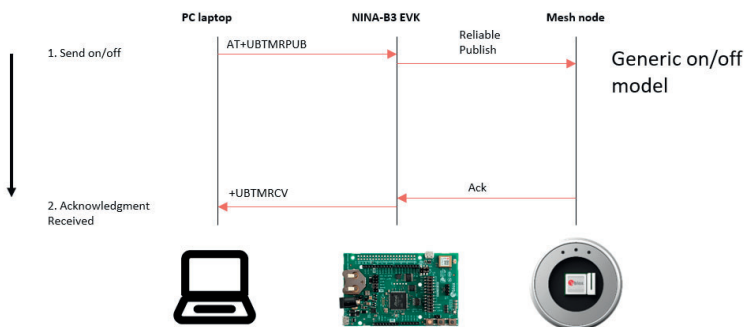


Figure 4.2: Round trip time of Bluetooth mesh network

4. Throughput: Elapsed time and amount of data sent from one node to another is used to calculate link throughput. Max throughput is achieved when the shortest possible time interval between packet transmission is used. Number of packets is directly proportional to the link throughput.

The throughput measurement is carried out between the test points only, i.e. from test point 1 to test point 4 and test point 2 to test point 3. Packet size = N bytes = $N \times 8 = n$ (bits).

Packets p sent during measurement time.

round trip time = $RTT(s)$

Link throughput = $n \times p / RTT$ (b/s)

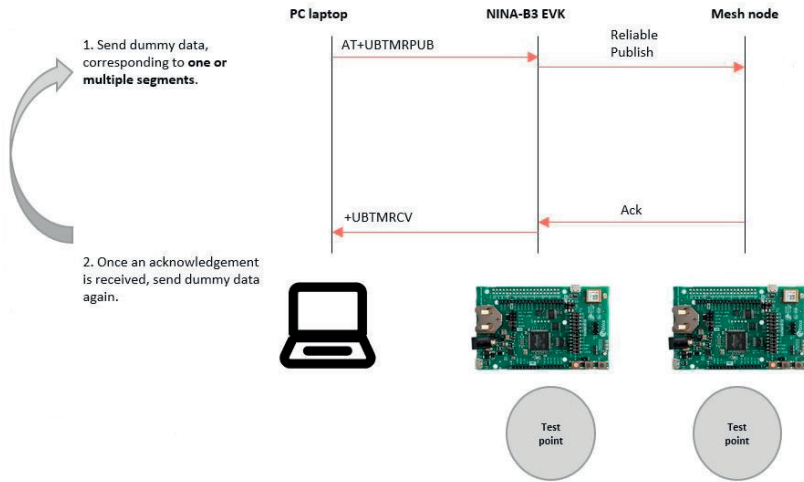


Figure 4.3: Throughput of Bluetooth mesh network

4.3 Scenarios considered

In our Bluetooth mesh network test we are considering 2 types of sensor nodes, they are *ALL Treon* and *best guess*. We are considering 3 background traffic models for each sensor nodes, this makes a total of six different types of scenarios which are shown in the table 4.1 below.

Background traffic model	Sensor cadence	TTL	Sensor nodes type
1	OFF	5	Best guess
2	OFF	5	ALL Treon
3	10 seconds	5	Best guess
4	1 second	5	Best guess
5	10 seconds	5	ALL Treon
6	1 second	5	ALL Treon

Table 4.1: Measurement traffic models

4.3.1 Sensor Nodes Types

A Treon node is a wireless sensor device for mesh networks. It is equipped with a variety of sensors to monitor environmental quality. It calculates an air quality index (via measuring a broad range of gases including Volatile Organic Compounds) and keeps track of temperature, humidity, ambient light, and barometric pressure. A Treon node can also measure acceleration and detect the proximity of a magnet. Once the node is turned on, it starts automatically to measure and transmit data at pre-configured intervals. A Treon node operates in a mesh network transmitting sensor values directly or via other nodes to a gateway, such as a Treon gateway. Typically, the data is sent from the gateway to a cloud back end for storage and analysis. [17]

Best guess: In best guess configuration the center few nodes are configured as relay nodes and rest of the nodes are configured as the end nodes

ALL Treon: In ALL Treon configuration all the nodes are set as relays and there are no end nodes, so that all the nodes receive and transmit packets.

4.3.2 Sensor Cadence

The Sensor Cadence state controls the cadence of sensor reports. It allows a sensor to be configured to send measured values using Sensor Status messages at a different cadence for a range of measured values. It also allows a sensor to be configured to send measured values when the value changes up or down by more than a configured delta value. If the *Fast Cadence High* value is equal or higher than the *Fast Cadence Low* value, and the measured value is within the closed interval of [Fast Cadence Low, Fast Cadence High], the *Sensor Status* messages are published more frequently. The messages shall be published every *Publish Period* divided by the Fast Cadence Period Divisor state. [12]

If the Fast Cadence High value is lower than the Fast Cadence Low value, and the measured value is lower than the Fast Cadence High value or is higher than the Fast Cadence Low value, the Sensor Status messages are published more frequently. The messages shall be published every Publish Period (configured for the model) divided by the Fast Cadence Period Divisor state [12].

The Fast Cadence Period Divisor field is a 7-bit value that shall control the increased cadence of publishing Sensor Status messages. The value is represented as a 2^n divisor of the Publish Period. For example, the value 0x04 would have a divisor of 16, and the value 0x00 would have a divisor of 1.

4.3.3 Flow of the project

The main aim of the project is to measure the latency of nodes in the Bluetooth Mesh network. The measurements are carried out with the help of Bluetooth modules. This Bluetooth Mesh Network is for an office of area 50m x 60m. The measurements are undertaken with the help of 4 test points (Bluetooth modules explained later in the report) which are placed at 4 corners of the office, there are 30 nodes spread out randomly in the office. These test points send a random message to each node and the node responds with an acknowledgment, this in

turn finds the RTT (Round Trip Time) between the test point and the node, this measurement is carried out for all 30 nodes from each of the different test points to get different RTT. Before this measurement, we can configure the Time to Live (TTL) for the mesh network and the node properties. And, the throughput is also measured, but in this case, it is measured only between the testpoints. There are 2 different types of nodes used in here 3 different background traffic types, therefore this gives us 6 different measurement values. This measurement is carried out using the C-Sharp platform. Our task was to create a theoretical Bluetooth Mesh model using MATLAB based on the readings obtained from the measurement. For this model, we had to take into consideration the distance between each node and test points to make a map of the mesh network layout. To make this model as similar to the readings obtained from the measurements, several parameters were considered, the key ones were the number of hops each message took to reach from source to destination (that is from test point to node). To limit these hops to a certain value we set the TTL to 5 (this is considered in the measurements as well). The main focus was on the link layer of the Bluetooth LE stack. Most importantly a model will not be like real value, since, during measurements, there are several factors like noise and interference which affects the readings, hence, in our model, we also considered noise parameters to make the values similar to the measured values. There were several other configurations based on Bluetooth SIG (Special Interest Group) that were used to make the model work and be as close as possible to the measured value.

4.4 Measurements

The measurements are made concerning 4 test points (or 4 EVKs). Each test point sends a message to all the nodes and finds the basic parameters such as TTL, RTT, and throughput. Since 4 test points are hooked up to raspberry-pie they are available on the network, these test points are added as COM-ports on the PC/Laptop using software called 'virtual-here' before the measurements. Once the COM-ports are added, the specific COM-port numbers are to be mentioned in the configuration JSON file (JavaScript Object Notation). These are the initial setup before the measurements. The test cases have been written on C-Sharp code, we used Visual Studio as the IDE (Integrated Development Environment) to run this. We can configure parameters such as TTL and message type. This measurement is carried out for 300 iterations as suggested in u-blox.

In case, if a problem occurs in any of the COM-ports they can be verified by running AT-commands on company-specific software called S-Center. We can also use this to check if the nodes are working properly.

4.4.1 Evaluation Kit and NINA-B3

The NINA-B3 evaluation kit is a platform that allows simulating and developing low power IoT applications using full Bluetooth 5, Near Field Communication (NFC), and IEEE 802.15.4.

The NINA-B3 module used in the EVK is stand-alone Bluetooth low energy Microcontroller Unit (MCU). It includes Arm® Cortex® -M4F microcontroller

with 1 MB internal flash and 256 KB RAM and having a system clock speed of 64 MHz and state of the art performance which improves power consumption by enabling optimum power saver modes. With a 1MB flash and 256 KB RAM, they provide the best capacity for applications on top of Bluetooth LE stacks such as beacons, GATT, and mesh. These are widely used in smart buildings, smart cities which include smart lighting systems, sensor networks, asset tracking solutions, and automation systems. Figure 4.4 shows the evaluation kit and Figure 4.5 shows the NINA-B3 microcontroller.

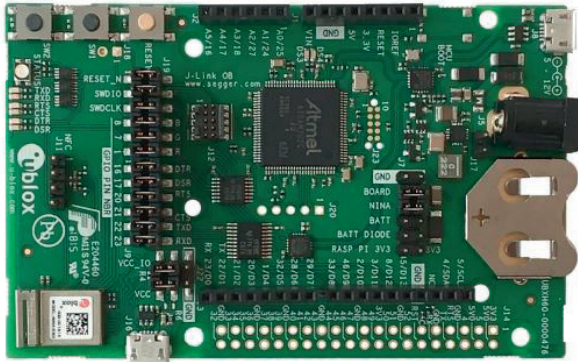


Figure 4.4: EVK NINA-B3 evaluation board



Figure 4.5: NINA-B3 microcontroller

4.4.2 Virtual-here

Normally the Universal Serial Bus (USB) devices are connected to the PC/Laptops by cables. With the use of virtual-here we don't need any cables to connect the devices, the network in which the virtual here is connected acts as the cable to connect the devices from PC/Laptop. This USB server solution helps the USB device to work remotely over a Local Area Network (LAN) network, Internet, and cloud network without the USB is physically connected to PC/Laptop.

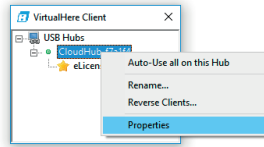


Figure 4.6: Virtual-Here [14]

4.4.3 S-center u-connectXpress Software

The u-connectXpress software is a powerful and easy-to-use tool for evaluating, configuring, and testing u-blox short range modules. It includes an intuitive, easy-to-understand and easy-to-use graphical interface. S-center provides a convenient means to configure the u-blox short range modules, to save the configuration in the flash memory of the module, and to restore factory settings if needed. Toolbar but-

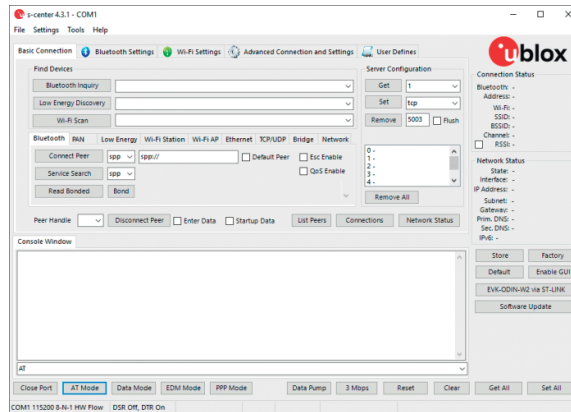


Figure 4.7: S-Center

tons are available to control settings. Each command that is executed by s-center is an AT command that is described in the Short Range Modules AT Command Manual. S-center also includes AT command terminal with user defined commands, support for Universal asynchronous receiver-transmitter (UART), Transmission Control Protocol (TCP) client and TCP server interface, firmware update feature for u-blox short range modules, chat tool, TCP reflector, and data pump.

Simulation model and results

The main purpose of creating this simulation model for the application team at u-blox is to have theoretical descriptive values on the link performance of the Bluetooth mesh network. So that, the consumers who implement the Bluetooth mesh network for their specific needs would have an appropriate idea of the performance and reliability of the Bluetooth mesh nodes and also have a brief picture of the number of nodes that they would require for a good performance in a given area.

5.1 Layout of the nodes in mesh network

Our first task was to replicate the node positions similar to the map provided as shown in Figure 4.1. To achieve this task we found out the distance between all the nodes and the test points to plot the x and y coordinates, so that we could have a visual map in our model and to trace the path that the message packet would take when transmitted from a source to reach the destination. The map obtained from simulation model is as shown below. As seen from the figure, the peach colour dots represent the relay nodes and blue dots represent the end nodes.

5.2 Implementation

This implementation is carried out in MATLAB. The main aim of this simulation was to make it as similar to the measurement values. To make this possible we used the Communication toolbox in MATLAB. It consists of all the Bluetooth functions having a Bluetooth LE stack on top of which the Bluetooth mesh works.

The Bluetooth mesh works on the principle of the flooding mechanism. As the name suggests the transmitter floods the packet through the entire network and chooses the path with the least hops. To control the hops, TTL is set, as the packet takes a hop, the TTL is decremented by 1, this continues until the TTL has become zero. This is a controlled flooding mechanism, as the transmitted packet is attached with the address and a sequence number, this sequence number is saved in the cache memory of each node and it prevents sending the message to the same node twice. [18]

As shown in Figure 3.2 the functioning of each layer was modeled as a separate object to create an entire model of the Bluetooth mesh. Following the Bluetooth

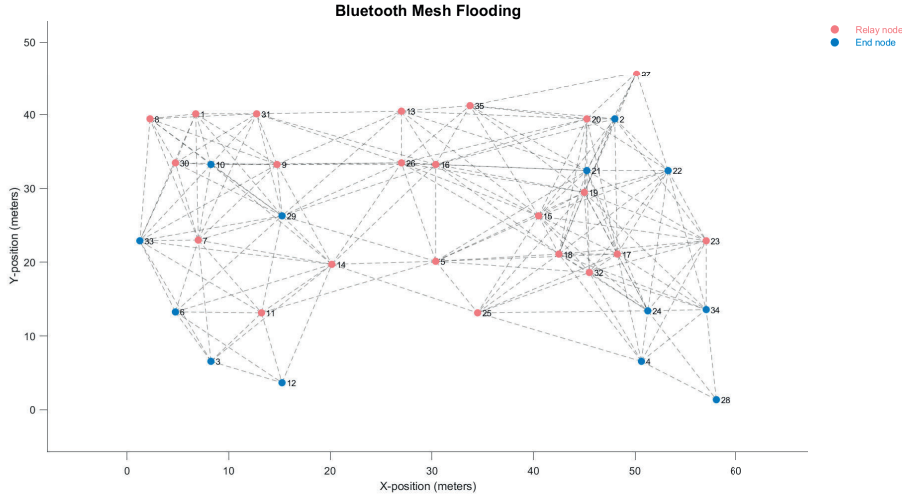


Figure 5.1: Simulated layout of Bluetooth mesh network

SIG documents several parameters were implemented, our Bluetooth mesh works on the Generic on-Off model (mentioned in Appendix).

Now coming to the parameters, TTL is set to 5 as mentioned in the measurement Table 4.1, based on this the hops the message needs to reach the destination is found. For each hop, the TTL is decremented by 1, once the TTL reaches 0 the message is discarded and no values are obtained to this reading. To identify each node they are addressed using hexadecimal values referred to as unicast address and each node is given an ID, which can be used to mention the source from which the packet will be transmitted and the destination to which it is supposed to reach. Based on the ID's and the unicast address of that destination node is obtained and the packet is flooded throughout the network until it reaches the destination. The packet takes the shortest route to reach the destination.

As we know that RTT refers to the total time of sending a message from the source until it reaches the destination and the destination sends an acknowledgment of receiving the message. Here we assumed that the acknowledgment will take the same path as the original message took to reach the destination. For this, we took the time it took to reach the destination and multiplied by 2. Of course, in theory, it will be well and good, but, in real life, there are so many factors that are affecting this.

To simulate the messages sent, we used the *Generic on-off set* and *Generic on-off status* opcodes specified by Bluetooth SIG (opcodes are mentioned in Appendix). The equation 5.1 was the base foundation used to find the RTT and equation 5.2 used to find the retransmission time.

$$RTT = \text{Number of hops} * 10ms. \quad (5.1)$$

In the above equation, the 10 ms contribute to the advertising intervals, scanning interval, and transmission delay of the network. If the packet is lost then the re-

transmission takes place in the network. The retransmission formula is mentioned below.

$$\text{Retransmission} = 200\text{ms} + 50\text{ms} * TTL \quad (5.2)$$

In equation 5.1 the 10ms hold for the advertising delay, scanning delay, and transmission times, and in the equation 5.2, 200 ms is the time the source waits if the message fails to reach the destination in the first attempt and the 50 ms makes up for the scan interval, advertising delay and transmission time, here the scanning delay is longer since it is for the retransmission. For this, the re-transmit count is set to 2, so, the source tries one more time before giving up. So, for the second attempt it waits another 150ms before it transmits, if this attempt also is unsuccessful, it stops transmitting and gives zero (meaning packet failed).

As mentioned earlier, all of these formulas are good, but cannot replicate the real measurement values. The first attempt to find RTT in our model was based on these equations, but the readings were not close to the measured value.

For the RTT simulation, we compared our RTT model which was based on the formula to the measurement values. For this, we considered RTT per-hop which was 10ms from the formula to the RTT per-hop of the measurement which was ranged from 12.332 ms to 14.6 ms, we found the delta between the simulation model and the measurement and added it to RTT calculation. Since, in reality, many elements are affecting the RTT, like interference from other wireless devices, let's just call it noise to simplify things. We introduced noise elements like random Gaussian noise such that the values from equation 5.1 deviate approximately 30 milliseconds to 100 milliseconds to get the values closer to the measured values. To set these noise values, we added random noise with the value of how much the simulation model was varying with the measurement and we set a threshold value of 160ms, that is once the RTT of any iteration reaches 160ms or beyond, it takes the retransmission path, that is it follows equation 5.2. For the second retransmission, the threshold was set for 500ms, for any iteration with RTT of 500ms or beyond would wait for 150ms and transmit again. Usually, the second retransmission is not considered because of very high values and only very few iterations around 2% take the second retransmission. After few corrections to noise levels, the RTT obtained were close enough to the measured values.

The message sent is of size 11 Bytes. As seen in the simulation model map Figure 5.1, we are considering points 1 to 4 as test-points and the rest as normal nodes. Each node can be configured to being either a relay or an end node. In the simulation model, the RTT is carried out for 100 iterations.

Similarly, after obtaining the round trip time from the diagonally opposite test points (in this simulation we considered these testpoints as extra nodes). we calculate the throughput using the throughput formula. The throughput is calculated only between the testpoints 1 to 4 and 2 to 3.

Packet size = N bytes = N x 8 = n (bits).

Packets p sent during measurement time.

round trip time = RTT(s)

Link throughput = n x p / RTT (b/s)

We are sending a message of size 11 Bytes and the corresponding throughput is found.

5.3 Results

5.3.1 Round Trip Time

Now, coming to the main part of the project is to compare the results of the measurements and the simulation model. we compare the results of RTT concerning distance, TTL concerning distance, and Throughput concerning RTT. Here we use a curve-fitting tool to see the trend of how each of them varies with the other parameter. The base is that the round trip time increases as the distance increases, TTL decreases as the distance increases, and throughput increases and the RTT decreases. Here, the distance is measured in meters, RTT is in units of milliseconds, throughput with units of bits/second (b/s) and there are no units for TTL.

The upper graphs represent the values of the measurement and the lower graphs represent the values of the simulation model, this representation is the same for all six readings. These graphs are plotted using the curve fitting tool in MATLAB. It shows the overall average values of the readings and shows how they are varying. As mentioned earlier we can see from the graph that the RTT increases as the distance increases.

For ALL Treon the number of relay nodes is more (27 of 30 nodes). This makes it more reliable and when the sensor cadence setting of 'off' makes the background traffic less. When it is 1 second, this means that the nodes send a message after every one second, this leads to an increase in background traffic and when the sensor cadence is set to 10 seconds, the messages are sent every 10 seconds, this reduces the traffic and chances for retransmission.

For best guess as seen from the graphs below the number of zeros are more, this is because there are more end nodes compared to all Treon (21 of 30 nodes are relays). The sensor cadence settings will be the same as it follows 'off' makes the background traffic less and the rest of the settings of sensor cadence follows the same principle of ALL Treon. Here, the red cross marks in the plots are the zero values, it means that the packets are lost or not delivered. This can happen because of few reasons, one is when the packets get lost due to more interference or if the destination node is far and the packet has to take more hops (more than 5) than the set value of TTL, it becomes zero. Another case is when a message is sent to an end node, so, the end nodes cannot relay or transmit an acknowledgment, therefore the source node does not receive any acknowledgment, it assumes the packet delivery has failed.

$$RTT = (c1 * distance/3e8) + (10 * hops) + c2 + r + N \quad (5.3)$$

The simulation model follows equation 5.3, where c1 is the slope of the linear region in the plot of RTT vs Distance with a slope of 2.63, this slope is varying for each simulation, but the mentioned slope is average of all the slope values, c2 is the offset value between the measurement and the simulation model, N is the Gaussian noise and r is the initial values that represents whether it is a transmission or retransmission, r=0 with a probability of 0.698, r=200 with probability 0.081 and r=500 with a probability of 0.02. The probability of packet failing is about 0.201. The 5.3 for all of the simulation trials carried out in this project.

RTT vs distance comparison of ALLTreon-off from measurement and simulation model

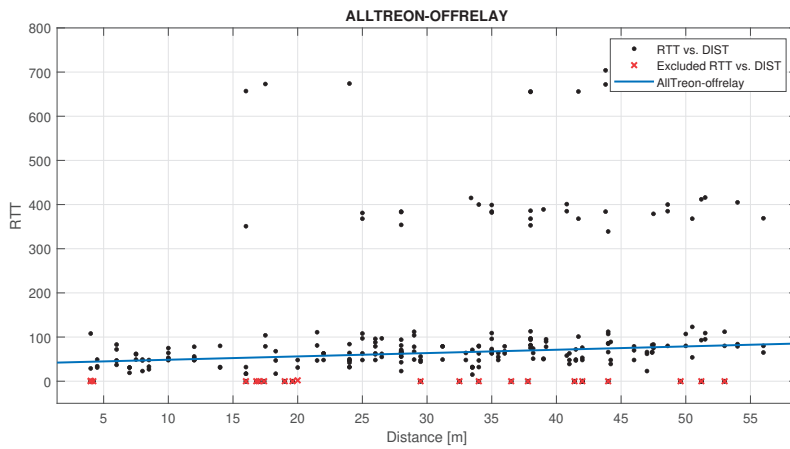


Figure 5.2: ALLTreon-off RTT vs distance(Measurement)

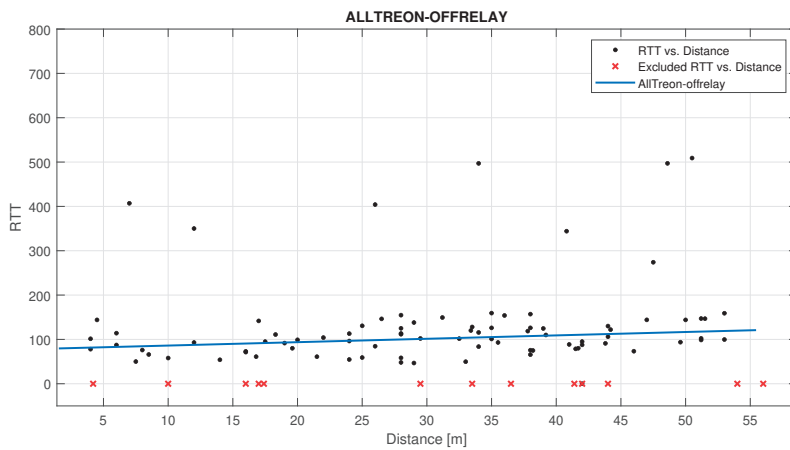


Figure 5.3: ALLTreon-off RTT vs distance(Simulation model)

RTT vs distance comparison of BestGuess-off from measurement and simulation model

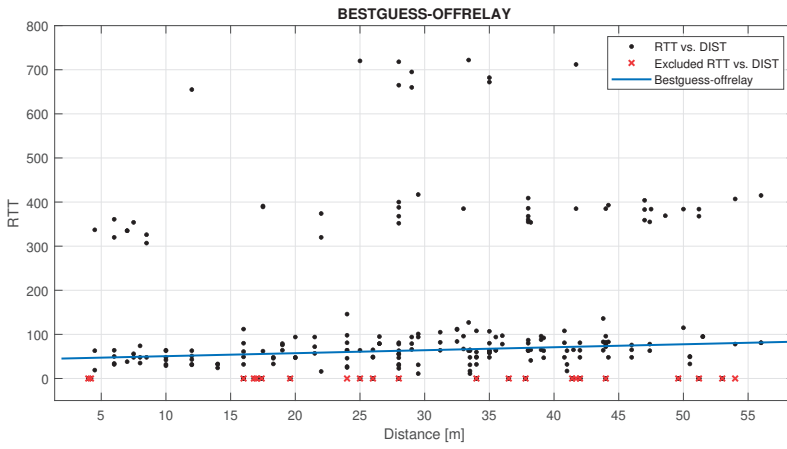


Figure 5.4: BestGuess-off RTT vs distance(Measurement)

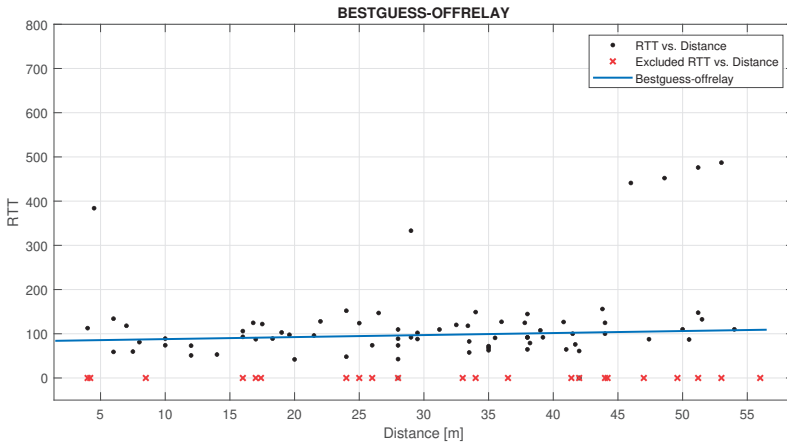


Figure 5.5: BestGuess-off RTT vs distance(Simulation model)

RTT vs distance comparison of ALLTreon-1second from measurement and simulation model

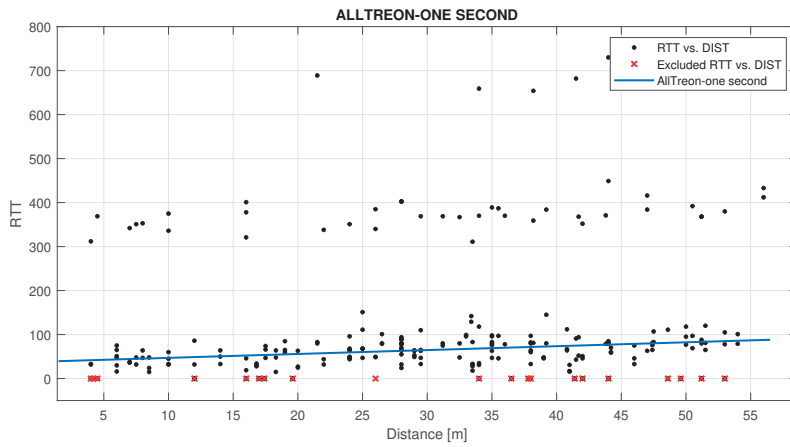


Figure 5.6: ALLTreon-1second RTT vs distance(Measurement)

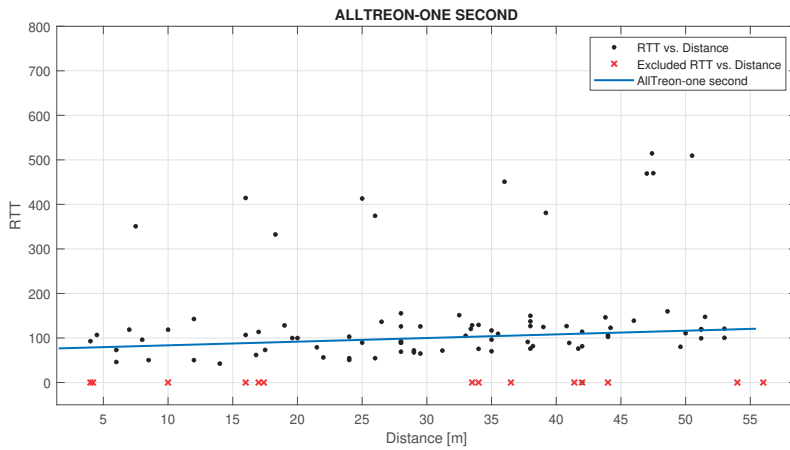


Figure 5.7: ALLTreon-1second RTT vs distance(Simulation model)

RTT vs distance comparison of BestGuess-1second from measurement and simulation model

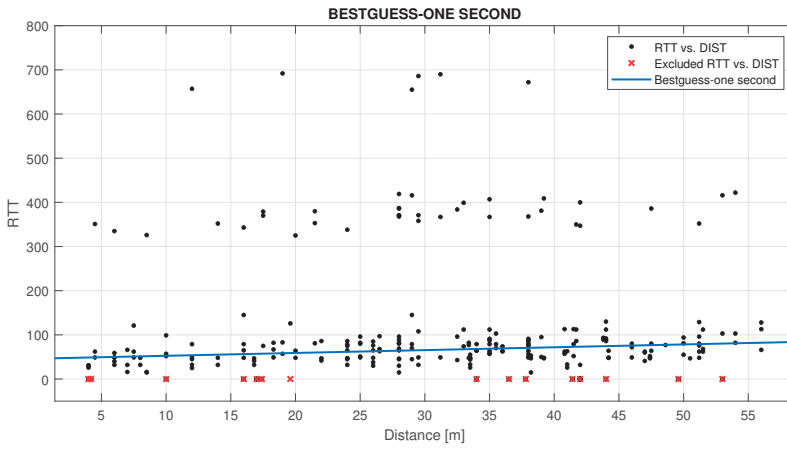


Figure 5.8: BestGuess-1second RTT vs distance(Measurement)

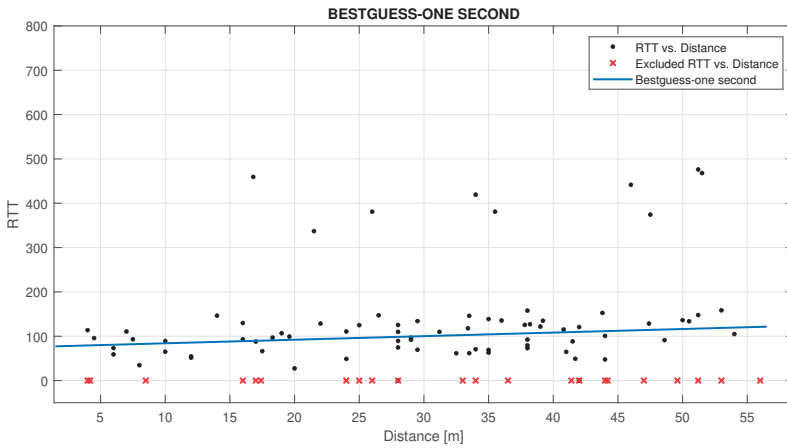


Figure 5.9: BestGuess-1second RTT vs distance(Simulation model)

RTT vs distance comparison of ALLTreon-10seconds from measurement and simulation model

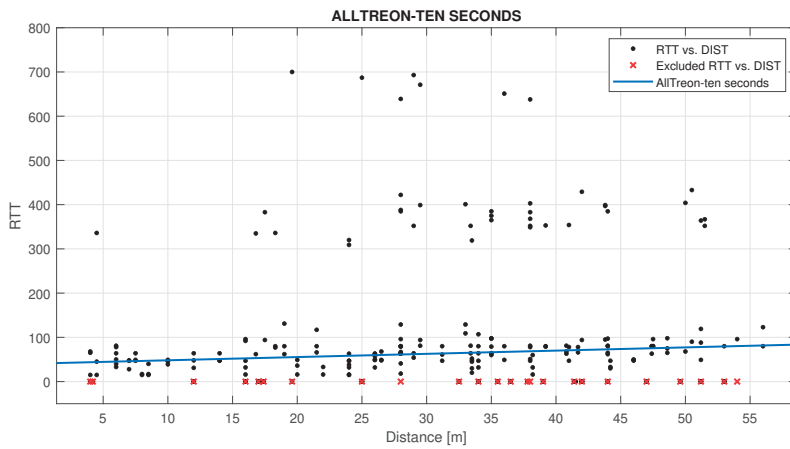


Figure 5.10: ALLTreon-10seconds RTT vs distance(Measurement)

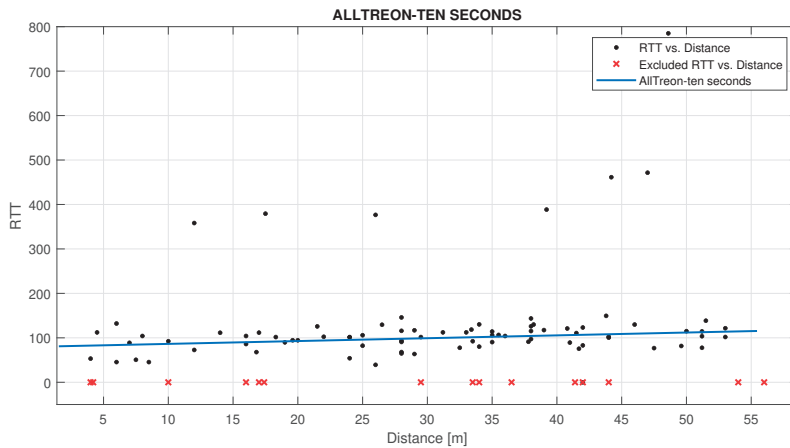


Figure 5.11: ALLTreon-10seconds RTT vs distance(Simulation model)

RTT vs distance comparison of BestGuess-10seconds from measurement and simulation model

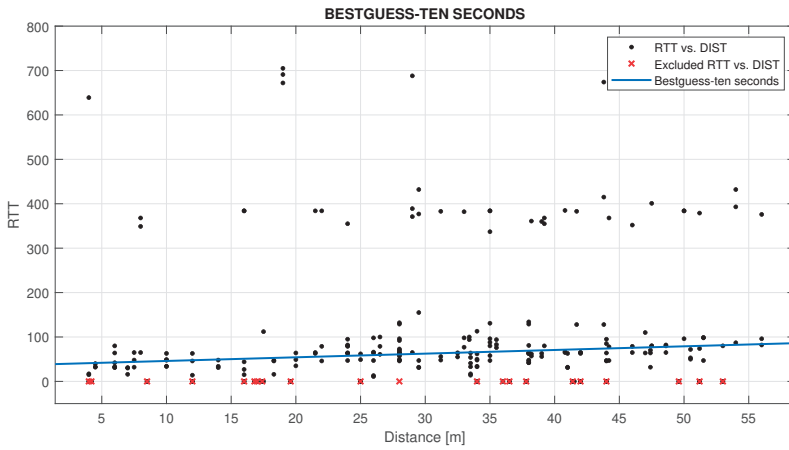


Figure 5.12: BestGuess-10seconds RTT vs distance(Measurement)

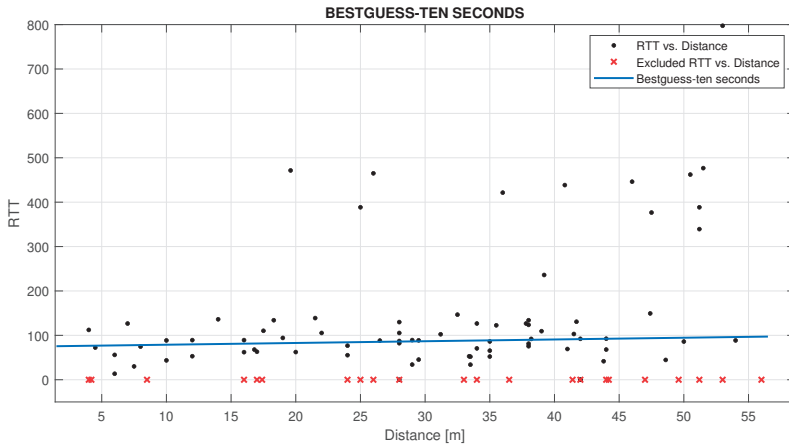


Figure 5.13: BestGuess-10seconds RTT vs distance(Simulation model)

To make it simple, we averaged out the RTT values. We have separated the values which are below 200 ms. They are the transmission with the first attempt, that is no retransmission, and values above 200ms are the ones where the packets are lost and retransmission occurs.

In these two tables 5.1 and 5.2, we have compared only the first transmission values of the measurement and simulation model. For the measurement, there are about 242 values compared to 76 values of the simulation model, hence the average values of the simulation model are little higher than the measurement. But, the closeness of the simulation model to the measurement achieved is about 90.1%, that is the All Treon model around 85.37% similar to the measurement and Best Guess model is around 95.2% close to the measurement. From this we can say that the simulation model is reliable.

TYPE	offrelay(ms)	1second(ms)	10seconds(ms)
ALLTreon	63.03	64.6	61.513
BestGuess	63.7215	66.015	62.514

Table 5.1: Average RTT for measured values

TYPE	offrelay(ms)	1second(ms)	10seconds(ms)
ALLTreon	85.89	83.2018	81.8
BestGuess	70.37	72.412	62.89

Table 5.2: Average RTT for simulation model values

These two tables 5.3 and 5.4 have the average retransmission RTT, which is above 200ms. The simulation model values are almost close to measurement values.

TYPE	offrelay(ms)	1second(ms)	10seconds(ms)
ALLTreon	445.093	412.36	429.3902
BestGuess	429.042	435.61	429.108

Table 5.3: Average RTT retransmission for measured values

TYPE	offrelay(ms)	1second(ms)	10seconds(ms)
ALLTreon	410.36	415.29	409.899
BestGuess	428.8731	425.5039	409.183

Table 5.4: Average RTT retransmission for simulation model values

It is quite difficult to come to any conclusion based on these since RTT varies very because of interference present by other elements. So, to get a proper understanding of the node performance, we find their throughput and compare them.

5.3.2 Time To Live

In TTL we consider only two node configuration settings, one for ALL Treon and another for Best guess. As mentioned earlier since the number of nodes in Best guess is less the TTL for certain iterations will be more compared to ALL Treon. The TTL remains the same for all the background traffic model. The below graph shows the TTL vs distance.

The representation holds the same, the upper graphs represent the values of the measurement and the lower graphs represent the values of the simulation model.

TLL vs distance comparison of AllTreon from measurement and simulation model

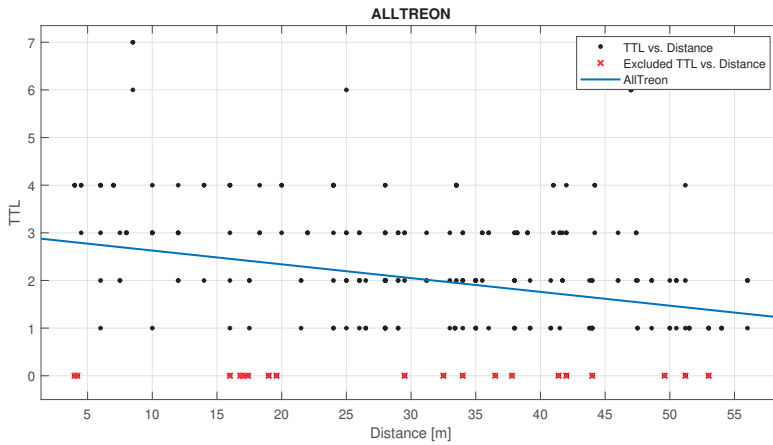


Figure 5.14: ALLTreon TTL vs distance(Measurement)

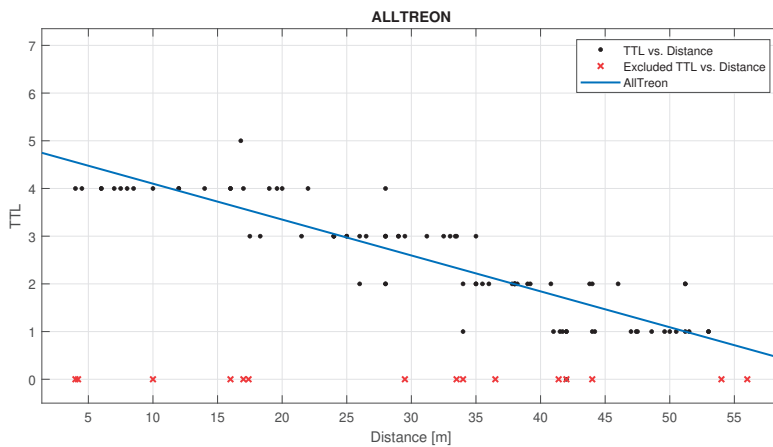


Figure 5.15: ALLTreon TTL vs distance(Simulation model)

TLL vs distance comparison of BestGuess from measurement and simulation model

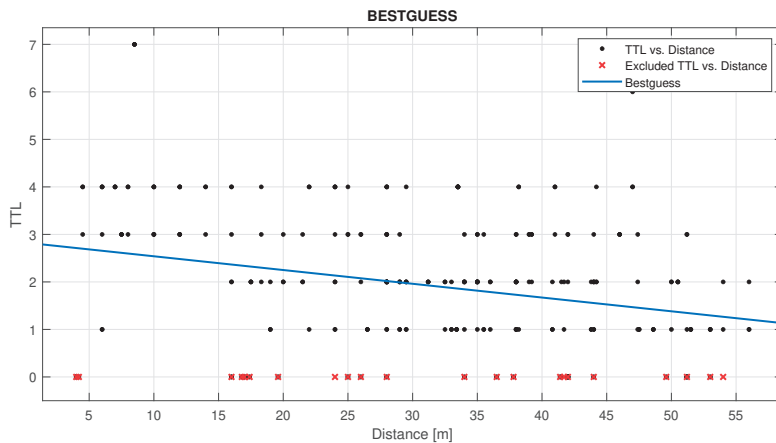


Figure 5.16: BestGuess TTL vs distance(Measurement)

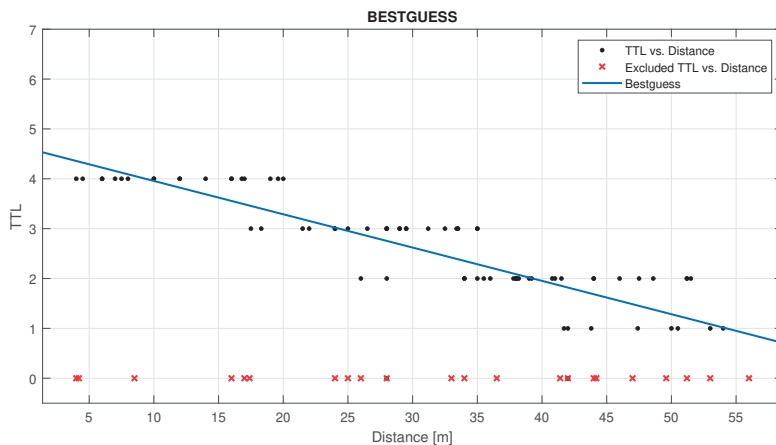


Figure 5.17: BestGuess TTL vs distance(Simulation model)

The TTL is a little different than measurement values because, in measurement, the node configurations have been varied consistently, but, in our simulation, we are keeping the node configurations constant, that is one configuration setting for ALL Treon and one configuration setting for Best Guess for all the iterations.

5.3.3 Throughput

To get a better idea of how the nodes are working we take the throughput measurements since the RTT is quite unreliable due to noise. But from these results, we can properly compare the result and choose the node with better performance.

The throughput measurement is carried out between the test points 1 to 4 and 2 to 3. In the measurement there are 30 iterations for each test run, this is carried out for both the type of sensor nodes and the three background traffic models. In the simulation we have considered 30 iterations same as measurement, and the throughput is calculated using the formula mentioned in the Section 4.2. For the throughput measurement and in the simulation model 30 message packets are sent and each message packet is of size 11 Bytes. To show the comparison we have taken the average value of the throughput for both measurement and simulation model in Figure 5.18.



Figure 5.18: Throughput comparison of measurement vs model

As we can see from the above figure ALL Treon nodes has the better performance and one second sensor cadence setting achieves peak throughput. From this we can say that the Treon nodes have better throughput and in turn lesser round trip time.

5.3.4 Reliability of simulation mesh model

To check the reliability of the model and to show that it is not just replicating the measurement values, we simulated another set of scenarios for RTT values. In this scenario, we reduced the number of nodes almost by half (16 nodes), we simulated for one set of values for each sensor node types, that is for *ALL Treon* and *Bestguess*. There were few changes made for the configurations, the TTL was set to 6 in ALL Treon case and TTL was set to 7 in Bestguess case. The map in Figure 5.19 shows the layout of the nodes. The plots show the values of RTT

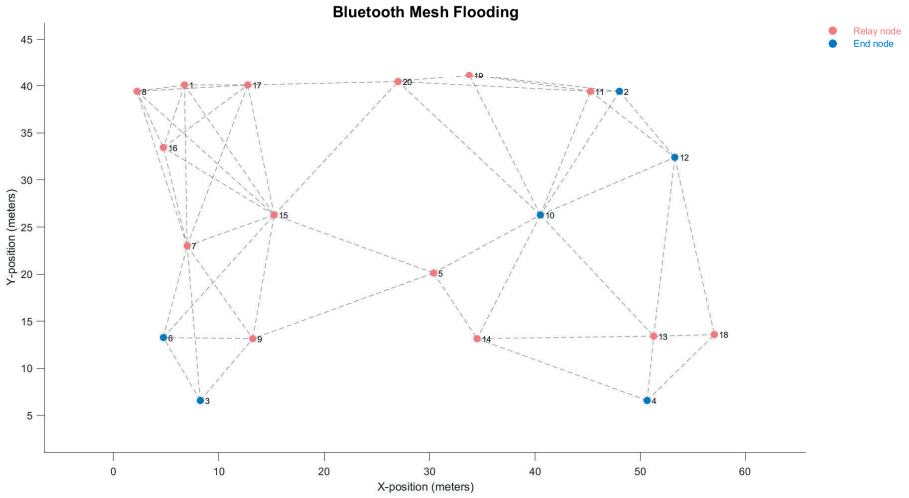


Figure 5.19: Scenario of map with reduced nodes

vs Distance. To be more accurate, we tabulated the average values of RTT for the transmission time and one for retransmission time. As seen from the table, the average RTT values are higher in this case, this is because, the nodes are spread apart and it will take the packet more hops to reach the destination, this in turn increases the RTT. There are more chances of retransmission because, the packets delivery gets failed, due to more number of hops.

TYPE	Avg transmission Time (ms)	Avg re-transmission time (ms)
ALLTreon	108.4629	531.6361
BestGuess	109.9756	686.3855

Table 5.5: Average RTT values

Alltreon and Bestguess scenarios in simulation model with reduced nodes

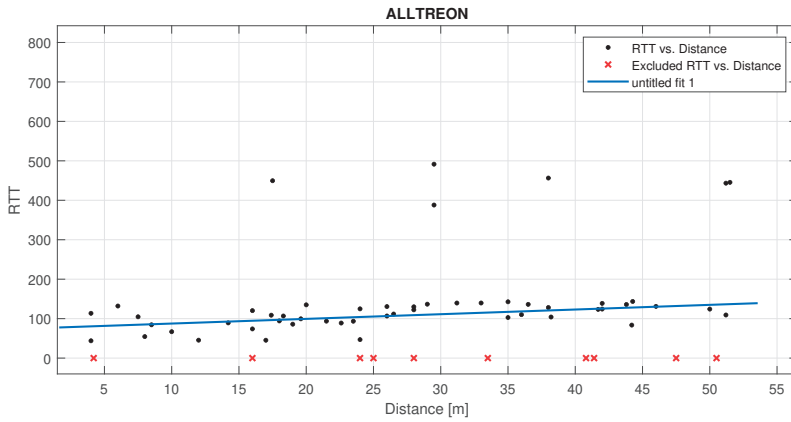


Figure 5.20: Alltreon scenario with reduced nodes(Simulation model)

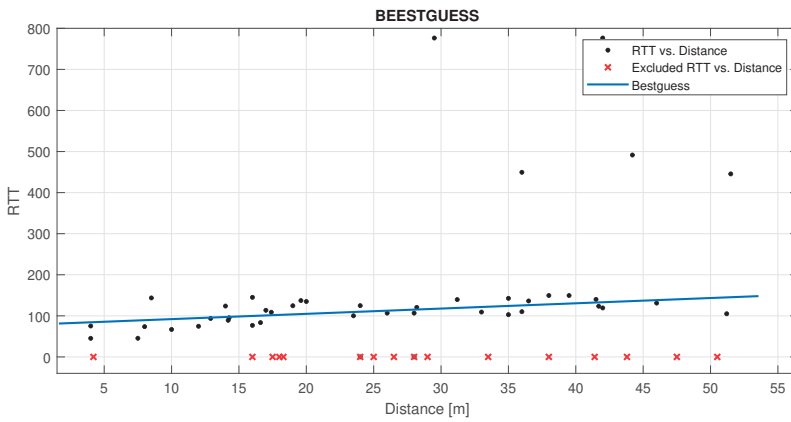


Figure 5.21: Bestguess scenario with reduced nodes(Simulation model)

All of the results of measurements and simulation models were done to know the optimal performance of the Bluetooth Mesh Network and node performance. These results will give a brief idea for the customers who purchase these Bluetooth nodes about its performance and the number of nodes that would be required for optimal performance in a given work environment.

Now, coming to the approximation of the number of nodes that would be required in a Bluetooth mesh network in a given area, we have made a brief calculation based on our measurements and simulation and with few assumptions.

Considering the node requirements for an office/building of area (50m x 50m). From the measurements and simulation, we have a base RTT for a single hop from source to destination to be around 11.854milliseconds(including all the background traffic) and the vicinity range of each node to be 15 meters, with these as a key factor, we calculate the number of nodes that would be required.

Let's consider a scenario when the requirements of RTT is below 100milliseconds. To fulfill these criteria, we can assume to have TTL configured to 4 or 5 (can be max up to 128). Since the distance per hop is around 12milliseconds, approximately 16-17 nodes would be sufficient, with an average RTT of 35-40 milliseconds and average RTT from corner to corner nodes to be around 72 milliseconds(these are the values between node to node). The number of nodes can be increased to around 20 for better performance. If the RTT requirements are not too tight, then, nodes can be reduced by half, with increased TTL, this scenario of the effect of reducing the nodes has been shown using the simulation model. Depending on the application of the nodes in the Bluetooth Mesh Network, if the information is to be sent more frequently (for example, frequent heartbeat readings), the sensor cadence value can be set to a lower threshold value.

Future Work

During certain applications (Temperature sensing mostly) Low Power nodes are used, this reduces the power consumption by remaining silent when it is not needed. This LPN works in connection with another node called the Friend node. The friend feature is currently not available in the current Treon node which was used in the measurements, this can be implemented in the future when the nodes are updated. Also, when the transmitting message sizes increase, which gives rise to the segmentation of messages. The feature of segmentation can be enabled on the Bluetooth LE stack and should check if there is any significant impact on parameters such as RTT and throughput.

References

- [1] Baert, M.; Rossey, J.; Shahid, A.; Hoebeke, J. *"The Bluetooth Mesh Standard: An Overview and Experimental Evaluation."* vol. 18, pp. 2409, Sensors 2018.
- [2] Michael Karlsson, *"MiraMesh, Thread and Bluetooth Mesh; A study on wireless mesh network disturbance resilience."* LumenRadio AB, Sweden, 2019.
- [3] Ali, Hassan.; Anna, Ahlquist. *Previous thesis work on "Using BLE mesh network for indoor tracking."* Master Thesis, Malmö universitet/Teknik och samhälle, Sweden, 2019.
- [4] https://en.wikipedia.org/wiki/Internet_of_things
- [5] <https://microchipdeveloper.com/wireless:ble-link-layer-channels>
- [6] *"Bluetooth Smart and Nordic's Softdevices - Part 1, GAP Advertising."* It is available at: <https://devzone.nordicsemi.com/nordic/short-range-guides/b/bluetooth-low-energy/posts/bluetooth-smart-and-the-nordics-softdevices-part-1>
- [7] *"Bluetooth Low Energy (BLE) overview."* It is available at: <https://www.global-tag.com/bluetooth-low-energy-ble-overview/>
- [8] *"Getting Started with Bluetooth Mesh - AN227069."* It is available at: https://www.cypress.com/html_doc/AppNotes/AN227069/index.html#getting-started-with-bluetooth-mesh-an227069
- [9] Ngela Hernández-Solna.; David Pérez-Díaz-De-Cerio.; Mario García-Lozano.; Antonio Valdovinos Bardají.; José-Luis Valenzuela. *"Bluetooth Mesh Analysis, Issues, and Challenges."* pp. 1-1, IEEE Access, 2020.
- [10] <https://www.bluetooth.com/blog/provisioning-a-bluetooth-mesh-network-part-1/>
- [11] *"The Fundamental Concepts of Bluetooth Mesh Networking."* It is available at: <https://www.bluetooth.com/blog/the-fundamental-concepts-of-bluetooth-mesh-networking-part-2/>
- [12] *"Bluetooth Mesh Profile specification, Version 1.0 or later."*
- [13] A.A. Pammi. *"Threats, Countermeasures, and Research Trends for BLE based IoT Devices."* Master Thesis, Arizona State University, US, 2017.

-
- [14] <http://www.isagecomm.com/virtual-usb-server-setup-guide/>
 - [15] <https://www.eetimes.com/the-highlights-of-bluetooth-mesh-networking-technology/#>
 - [16] "*Mesh Network Performance Comparison Accessed.*" Silicon Labs, AN1142, Mar.3, 2020. It is available at: <https://www.silabs.com/documents/public/application-notes/an1142-mesh-network-performancecomparison.pdf>
 - [17] <https://www.treon.fi/project/treon-node/>
 - [18] <https://www.geeksforgeeks.org/fixed-and-flooding-routing-algorithms/>
 - [19] "*Bluetooth SIG announces mesh networking capability.*" It is available at: <https://www.asmag.com/showpost/26924.aspx>
 - [20] <https://www.mdpi.com/1424-8220/19/5/1238/htm>

Extra Materials

A.1 AT commands

AT command mode allows you to interrogate the Bluetooth module and to change some of the settings; things like the name, the baud rate, whether or not it operates in slave mode or master mode.

When used as a master device AT commands allow you to connect to other Bluetooth slave devices

Some of the AT- Commands used and their description is given below.

A.3 Add Model +UBTMMOD

AT command	Description
AT+UBTMMOD=<model index>,<Type>,<company ID>,<model ID>	Adds a model
AT+UBTMMOD =<model index>	Reads model at index <model index>

A.3.1 Description

Defines and adds a model. The model can later be instantiated to an element through AT+UBTMELM.

A.3.2 Syntax

Response	Description
+UBTMMOD:<number of opcodes>,<Type>,<Company Id> <Model Id>,<opcode >[<opcode>[...]]	Read response
OK	Successful write response
ERROR	Error Response


Figure A.1: AT-Command (UBTMMOD)

A.24 Add publish address +UBTMPAD

AT command	Description
AT+UBTMPAD = <element index>, <model index> ,<publish address>	Add local publish address

A.24.1 Description

Set the publish address for the given model instance.

 See Appendix B for limitations regarding number of publish and subscription addresses.

A.24.2 Syntax

Response	Description
OK	Successful write response
ERROR	Error Response

Figure A.2: AT-Command (UBTMPAD)

A.5 Bind device key to remote configuration server +UBTMCCB

AT command	Description
AT+UBTMCCB=<element index>, <model index>, <remote address>, <device key>	Bind device key to a remote configuration server

A.5.1 Description

Binds a device key of a remote Configuration Server to a Configuration Client on the local node. The local node shall first define a Configuration Client model (e.g. using AT+UBTMMODG=1,0001). The model shall then be instantiated on element 1 (or higher).

After defining the model AT+UBMCCB can be used as preparation to send Configuration Server commands to the remote node.

Example:

```
AT+UBTMRPUB=1,1,8026,8028,5,99
```

Will request the relay status of the remote node using Config Relay Get with the expected answer message Config Relay Status.

Added device keys are stored in persistent memory. Only one remote Configuration Server can be configured at the time. AT+UBTMCCB sets the current remote server and will return ERROR if all device key positions are occupied. AT+UBTMCCU can be called to free up a device key position.

For more information on the configuration commands available please see the Mesh Profile Specification [2].

Figure A.3: AT-Command (UBTMCCB)

A.12 Reliable publish of opcode +UBTMRPUB

AT command	Description
AT+UBTMRPUB=<element index>, <model index>, <opcode>, <reply opcode>, <timeout>, <reliable ID>, [<data>]	Starts publishing a reliable message

A.12.1 Description

Start publishing a reliable message.

A.12.2 Syntax

Response	Description
OK	Successful write response
ERROR	Error Response

Figure A.4: AT-Command (UBTMRPUB)

A.14 Message event +UUBTMRCV

AT Event	Description
+UUBTMRCV:<event handle>, <element index>, <model index>, <opcode>, <source address type>, <source address>, <destination address type>, <destination address>, <TTL value>, <RSSI>, <data>	An unsolicited event from the mesh stack

A.14.1 Description

An unsolicited event from the mesh stack. It contains a message to an instance of a model with the given opcode. The user must always, with the exceptions described below, reply to this event with an AT+UBTMRPY command. The reply shall use the event handle.

- Unsolicited messages that are received from a group address do not need to be replied to.
- Reply messages themselves, for example a Generic OnOff Status message, do not need a reply.
- These kind of messages will have the event handle "FF".

Figure A.5: Server reply (UUBTMRCV)

A.2 OP-CODE used for measurements, specified by Bluetooth SIG

A message is defined as having an opcode, associated parameters, and behavior. An opcode may be a single octet (for special messages that require maximum possible payload for parameters), 2 octets (for standard messages), or 3 octets (for vendor-specific messages).

A total message size, including an opcode, is determined by the underlying transport layer, which may use a Segmentation and Reassembly (SAR) mechanism. To maximize performance and avoid the overhead of SAR, a design goal is to fit messages in a single segment. The transport layer provides up to 11 octets for a non-segmented message, leaving up to 10 octets that are available for parameters when using a 1-octet opcode, up to 9 octets available for parameters when using a 2-octet opcode, and up to 8 octets available for parameters when using a vendor-specific 3-octet opcode.

Message Name	Opcode
Config Node Reset	0x80 0x49
Config Node Reset Status	0x80 0x4A
Config Relay Get	0x80 0x26
Config Relay Set	0x80 0x27
Config Relay Status	0x80 0x28

Table A.1: Messages name and its opcode



LUND
UNIVERSITY

Series of Master's theses
Department of Electrical and Information Technology
LU/LTH-EIT 2020-797
<http://www.eit.lth.se>