IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 30, 2020, accepted March 3, 2020, date of publication March 13, 2020, date of current version March 26, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2980795

Bluetooth Mesh Analysis, Issues, and Challenges

ÁNGELA HERNÁNDEZ-SOLANA[1], DAVID PÉREZ-DÍAZ-DE-CERIO[2], MARIO GARCÍA-LOZANO[2], ANTONIO VALDOVINOS BARDAJÍ[1], AND JOSÉ-LUIS VALENZUELA[2]

[1]Aragón Institute of Engineering Research (I3A), University of Zaragoza, 50018 Zaragoza, Spain
[2]Department of Signal Theory and Communications, Universitat Politècnica de Catalunya, 08860 Barcelona, Spain

Corresponding author: Ángela Hernández-Solana (anhersol@unizar.es)

This work was supported in part by the Spanish Ministry of Science with ERFD funds under Project RTI2018-099880-B-C32, Project RTI2018-095684-B-I00, and Project RTI2018-099063-B-I00, and in part by the Government of Aragon (reference group T31_17R).

ABSTRACT BLE is a widely used short-range technology which has gained a relevant position inside the Internet-of-Things (IoT) paradigm development thanks to its simplicity, low-power consumption, low-cost and robustness. New enhancements over BLE have focused on supporting mesh network topology. Compared to other mesh networks, BLE mesh has only considered a managed flooding protocol in its first version. Managed flooding may generally seem inefficient in many contexts, but it is a high desirable option when data transmission is urgent, the network is small or its configuration changes in a very dynamic way. Knowing the interest to many application contexts, this paper analyses the impact of tweaking several features over the reliability and efficiency of the mesh network. These features are configured and controlled in different layers: message repetition schemes, the transmission randomization, the election of a scheme based on an acknowledged or unacknowledged transmission, etc. In order to estimate the real performance of a mesh network deployment, this paper evaluates the effects of the interaction of the chosen parameters, their appropriate adjustment in relation with the characteristics of real implementations and the true overhead related to the whole protocol stack. The paper identifies configuration challenges, proposes network tuning criteria and outlines possible standard improvements. For this purpose, a detailed assessment on the implementation and execution of real devices has been performed with their chipset limitations.

INDEX TERMS Bluetooth low energy, wireless mesh networks, BLE mesh, managed flooding, performance.

I. INTRODUCTION

Nowadays, there are many wireless communication technologies that can be used for the deployment of IoT applications in domestic, urban, and industrial scenarios. Prominent examples are ZigBee, Z-Wave, Thread, 6LoWPAN, Wi-Fi, Bluetooth Low Energy (BLE). These technologies are extremely heterogeneous in terms of protocols, performance, reliability, latency, cost and coverage. Thus, the choice of one specific technology depends on the particularities of the intended service and application scenario. Each of them may be optimal in some characteristics and be suboptimal in other goals. Therefore, it is not possible to conclude that a technology performs better than another for all conditions [1]–[3].

Among them, BLE is a widely used short-range technology, which has gained a dominant position thanks to its simplicity, low-power consumption, low-cost and robustness. BLE is currently present in almost all smartphones, tablets, computers and consumer electronics in general. This has enabled the development of a wide range of new services and applications in sectors such as healthcare, home automation, security or vehicular communications. BLE is particularly efficient in tracking things or people in indoors/outdoors scenarios with low-power requirements, high scalability and reliability.

However, unlike other technologies such as WiFi or ZigBee, until 2017, BLE lacked the capability of mesh networking. Mesh networks allow data transmission between pairs of nodes in a dynamic and non-hierarchical way. Nodes cooperate and allow an efficient relay of messages from/to other devices. Knowing that mesh topologies are an attractive alternative to traditional centralized or tree-based network topologies, adding mesh functionality to Bluetooth was a necessary step. The Bluetooth Special Interest Group (SIG)

The associate editor coordinating the review of this manuscript and approving it for publication was Abbas Jamalipour.

has formed the Bluetooth Smart Mesh Working Group to work on the standardization of such capabilities for BLE [4]. In fact, it is remarkable that mesh functionalities are not a part of the core Bluetooth standard [5].

Compared to other mesh networks or protocols (including ZigBee, Thread, Z-Wave, WiFi) that use routing techniques, the BLE SIG has only considered a flooding protocol in its first version. Indeed, a *managed* flooding, halfway between the basic flooding and routing. Nevertheless, the SIG specification itself contemplates the short-term incorporation of a routing mechanism [4]. The absence of a standardized and efficient algorithm limits BLE suitability for a variety of applications in spatially distributed networks. In fact, the success of Bluetooth mesh depends on the ability to provide unique features and the possibility to address a wider range of applications, equaling or exceeding those offered by competing technologies. This is the reason why there are many proprietary and academic solutions in order to include routing [6]–[8]. Flooding is not the only difference between BLE and other mesh networks. Mesh is built on top of BLE, just using its advertising/scanning states. It does not use the Internet Protocol (IP) either.

Comparison between BLE and the competing technologies for IoT, or between flooding and other routing alternatives, is not the matter of this paper. Overall characteristics, strengths and limitations of flooding and routing are widely known. Routing is robust and consumes little energy, but implies high delay and finding an optimal route is challenging. On the other hand, advantages of flooding are its simplicity, redundancy and that it does not require computing routing tables. Any new message is forwarded by multiple relay nodes. However, the number of relay nodes and retransmissions should be limited and finely tuned to control congestion, which results in packet loss and high delays due to the contention-based access. Otherwise, this yields high energy consumption and congestion, the main drawback of flooding. The managed flooding option considered by BLE improves basic flooding operation by adding some optimizations. Important examples are time-to-live (TTL) stamps, message caching, heartbeat messages and friend node features. Although flooding and even managed flooding may generally seem inefficient in many contexts, it is a high desirable option when data transmission is urgent, the network is small or its configuration changes in a very dynamic way. This is the case, for instance, of lighting applications. Currently, the main focus of the flooding based BLE mesh standard are lighting applications, for the sole reason that it is easy to apply in these systems. Nevertheless, it can work for other applications. The standard also defines a set of models for the operation on scenarios such as device configuration and sensor readings. Actually, dynamic switching between the two options (routing and flooding) may be contemplated to adapt the system to the network conditions in many scenarios.

Knowing that flooding-based configurations (core of the current specification) are of interest to many application contexts, the main objective of this paper is to identify configuration challenges, propose network tuning criteria and outline possible standard improvements. For this purpose, a detailed assessment on the implementation and execution of real devices has been performed with their chipset limitations.

A key point of BLE mesh is that it is defined to work over BLE core specification. Mesh PDUs are transported between nodes using the advertising bearer, and specifically using simple non-connectable and non-scannable undirected advertising events. Nevertheless, the structure of those packets and timing transmission parameters need to be adapted to mesh transport requirements. For instance, adaptations and randomization on advertising intervals and passive scanning with a duty cycle as close to 100 percent as possible in order to avoid missing any incoming mesh messages or PDUs are required. However, BLE devices present non-idealities in their operation that need to be well characterized. Often, BLE devices present blind times linked to decoding or switching frequency functions, which prevent a real continuous scanning and may result on higher PDU loss rate than expected.

The reliability and efficiency of the mesh network depend on several features such as message repetition schemes, the transmission randomization, the election of a scheme based on an acknowledged or, alternatively, on an unacknowledged transmission, which are configured and controlled in different layers of the protocol stack. Besides, to estimate the real performance of a mesh network deployment it is necessary to understand the effects of the interaction of the chosen parameters sets at the several involved layers on mesh specification (configuration needs be considered jointly to achieve the best performance), their appropriate adjustment in relation with the characteristics of real implementations (for instance, cache and buffering capacities are not ideal, and in fact, very limited values are often present in real chipsets), and the true overhead related to the whole protocol stack. Covering and discussing all these points are the contributions of this work.

The rest of the paper is organized as follows. First, section II briefly introduces the reader to the BLE mesh concept, topology, terminology and required features. Section III describes key design parameters linked to bearer services, network mesh layers and involved overhead. In section IV, we analyze the interactions of the adjustable functions and parameters. In some cases, the analysis allows us to provide configuration suggestions and, in other cases, it allows to identify open points that need further study. Section V briefly includes other additional research challenges linked to mesh. Finally, in section VI we give some concluding remarks.

II. OVERVIEW OF BLUETOOTH MESH BASIS AND ARCHITECTURE

In this section, we will introduce the key aspects that characterize Bluetooth Mesh [9]. Understanding BLE mesh networking first requires to define the terminology present in the standard, related with the BLE mesh concept and topology, and the required features to support them. An illustration of

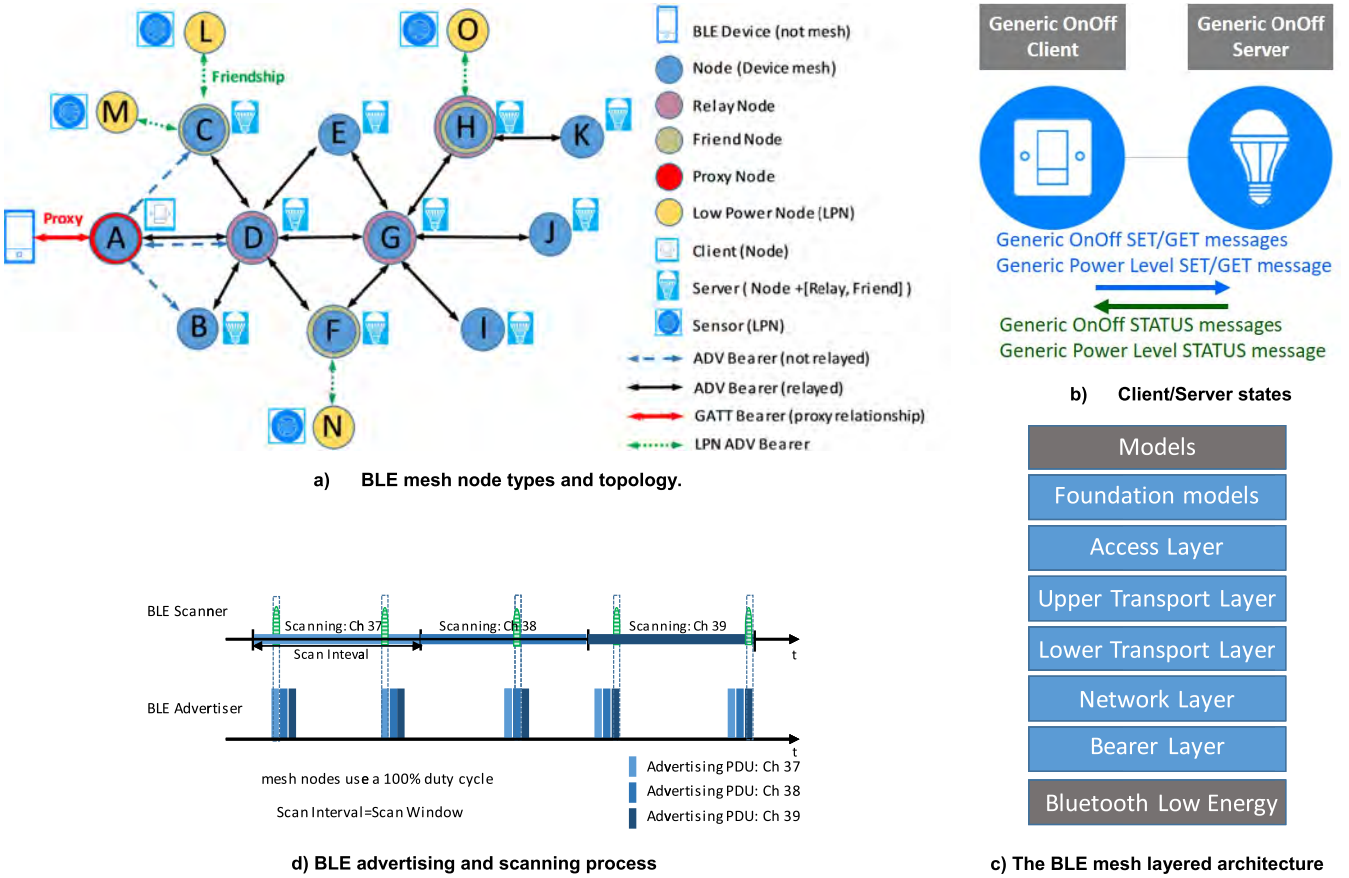

FIGURE 1. BLE mesh basis and architecture.

a mesh network, including the main terms and concepts that will be used along the paper, is shown in Fig. 1.a to ease its reading.

Devices which are part of a mesh network, specifically devices that are able to transmit and receive messages in a mesh network, are called "*Nodes*" or *"provisioned devices"*, and those which are not are called "*unprovisioned devices*". The "*provisioning process"* is the mechanism that transforms an *unprovisioned* device into a node.

Additionally, an *"Element"* is an addressable item within a device/node. A *node* is required to have at least one element (e.g., a light bulb, a security camera, smoke detector, temperature sensor, etc.). But, in fact, a device (*"node"*) could be composed of several elements (e.g., a light fixture composed by multiple light bulbs which can be turned on/off independently). Concerning the exchange of data between elements, it follows a publish/subscribe paradigm, and three types of addresses are contemplated: unicast, group and virtual addresses. The sender (e.g., node A in Figure 1.a) "publishes" a message to a certain address. If this address is unicast, the destination is a single element inside a node (e.g. node F) and it is automatically processed by this element upon reception. But, when a group/virtual address is used, elements that are interested in receiving the messages will subscribe to such group/virtual address, and only they would process that message (for example, all lamps in Fig. 1.a).

Following with the terminology, nodes have a number of optional features, giving them special capabilities:

Relay Node: A node that receives and then retransmits mesh messages over the advertising bearer to enable larger networks, using a *managed flooding* process. In this way, a node supporting the *relay feature* broadcasts messages to all nodes within its range but with a couple of optimizations: messages have a Time-To-Live (*TTL*) assigned which gets decremented with each retransmission. So, a message is only retransmitted when its *TTL* is greater than one. And, additionally, the messages are cached. Therefore, a received message that already exists in the cache is automatically discarded and not retransmitted.

Low Power Node (LPN) and Friend Node: A *low power node* is a node that has the ability to operate within a mesh network at significantly reduced reception duty cycles. In general, they are nodes that need to preserve energy as much as possible. Usually, they are nodes that spend most of their time transmitting data, that is, they are senders (e.g. a temperature sensor that sends data periodically or above a threshold), although occasionally can receive messages. In order to enable a LPN to reduce its receiver duty cycle and

save energy, it needs a friendship relationship to another node, called " *Friend Node*". The friend node, supporting the *friend feature*, provides assistance to a LPN in reception, by storing and forwarding messages destined to that node. Forwarding by the friend node is made on demand, when the LPN polls the friend for messages awaiting delivery.

Proxy Node: A node supporting *proxy feature* is able to relay/forward messages between non-mesh Bluetooth devices and a mesh network.

Provisioner Node: A device that is capable of adding a device to a mesh network (*provisioning process/feature*), providing among other information the necessary security keys. The provisioning process starts when an unprovisioned device sends mesh beacon advertisements announcing its availability to be configured. When the provisioner detects this device, it sends an information request to this unprovisioned device about supported security algorithms, number of elements the device supports, public key parameters, etc. Next, it starts the public key exchange (DevKey) and authentication between the device and the provisioner over a Bluetooth connection, other technology such as Near Field Communications (NFC) or even a QR (Quick Response) code. Once the authentication is complete, they exchange the provisioning data defined for that network: network and application keys, unicast address, default TTL, IV index, etc. When it is necessary to sell/change/dispose a node from a mesh network it is important to remember that it contains these security keys and should be removed properly from the mesh network.

Finally, each node supports a set of configuration states that concerns the node capabilities and behavior within the mesh. For example, the features supported by the node (proxy, relay and so on), the addresses the node has subscribed to, the security keys, the conditions under which an element can be found and their representation by a state value, etc.

Within BLE mesh specification, the *model paradigm* and, specifically, the term *model* organizes all these aspects. It addresses data among nodes for supporting applications in different scenarios and the definition of the behavior of nodes associated with the functionality. That is, similarly to the profiles in Bluetooth Classic, Bluetooth Mesh has its own specification for the so called models [9]. It completely defines the operation of typical usage scenarios for some specific functionalities, e.g., lighting or sensors. Elements can be in different *states* which also have associated properties. For example, a light may be represented through the *Generic OnOff* state. It is remarkable that *generic* states are reusable and allow a quick creation of new models. States can be managed by means of messages that can be of three types: GET and SET to request and change their value and the STATUS message. The latter is sent as a GET response, a SET acknowledgement, or independently as an *unsolicited* message. A node may include multiple models, while elements may be recognized as a server, client or control element.

Server is an element that implements the configuration server mode. This mode provides a device with the ability to be configured concerning various aspects (destination address to use when publishing messages or address subscribed to, parameters relating the periodic message publication, features like relay, LP o proxy roles), typically using an application that will implement the configuration client model (e.g. in Fig. 1.a, all the nodes except node A).

Client is an element that implements the Configuration Client model, i.e. a node that can change and monitor the configuration parameters of other nodes. It defines the messages which it may send or receive to GET, SET or acquire the STATUS of states defined in the servers. That is, an element exposing a state is referred to as a server and an element accessing a state is referred to as a client (see Fig. 1. b).

Control is an element that contains both a server model and a client model.

A. A LAYERED STACK OVERVIEW

Bluetooth mesh has been designed as a layered architecture with BLE 4.x backward compatibility. Thus, all certified Bluetooth Smart or Smart Ready devices could interact with a Bluetooth mesh network with suitable modifications to their software/firmware. Fig. 1.c) shows the layered stack. The main characteristic of the standard is that it is built on the top of the full BLE stack (physical and link layer). Data is transmitted consecutively on channels 37, 38, and 39 reserved for all non-connected state communications (see Fig. 1.d) and using non-connectable and non-scannable undirected advertising events, with some adaptations. On the top of this mesh stack, an application is implemented, being one of the most important key points of BLE mesh to standardize the specification of device behaviors according with the Model paradigm introduced above. To support it, the standard follows a layered architecture covering all the OSI layers from the application layer to the physical layer, in such a way that full BLE mesh stack concerns with: 1) How to define and implement these models; 2) How to address and deliver data across the mesh network; 3) How to abstract the underlying BLE Core [4] specification towards the layers above through the so called bearer concept.

We will concisely review each layer of the mesh architecture from top to bottom.

1) *Model layer*: Concerns the implementation of models and as such, the implementation of basic functionality of the nodes (behaviors, states, messages, and so on) in a specific and standardized application scenarios such as lightning and sensor. Each model is a part of the application and jointly with it and the foundation layer, they form a whole representation of the device.
2) *Foundation Model Layer:* It is responsible for the implementation of those models concerned with the configuration and management of a mesh network.
3) *Access Layer:* It defines how higher layer applications can use the more technical layers below (upper transport layer). It defines the format of the application data; it defines and controls the application data encryption

and decryption performed in the upper transport layer; and it checks 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.

4) *Upper Transport Layer:* The upper transport layer encrypts, decrypts, and authenticates application data and is designed to provide confidentiality of access messages.
5) *Lower Transport Layer:* It defines how upper transport layer messages are segmented and reassembled into multiple lower transport Packet Data Units (PDU).
6) *Network Layer:* It defines how transport layer messages are addressed towards one or more elements. It defines the network message format that allows Transport PDUs to be transported by the bearer layer. The network layer decides whether to relay/forward messages, accept them for further processing, or reject them. That is, relay and proxy features may be implemented by the network layer. It also defines how a network message is encrypted and authenticated.
7) *Bearer Layer:* It is the last layer before accessing the BLE core. It defines how network messages are transported between nodes. At the moment there are two bearers defined, the advertising bearer and the Generic Attribute Profile (GATT) bearer. The advertising bearer is the preferred bearer to deliver messages on a mesh network and defines the configuration and new content in non-connectable advertising PDUs and scanning configuration. On the other hand, the GATT bearer is provided to enable devices that are not capable of supporting the advertising bearer to participate in a mesh network. The GATT bearer uses the Proxy protocol to transmit and receive Proxy PDUs between two devices over a GATT connection.

Note that Bluetooth mesh networking was designed with security as one of its main priorities and it is mandatory: all Bluetooth mesh messages are encrypted and authenticated. In fact, it has different keys to address independently different concerns: network security (NetKey), application security (AppKey) and device security (DevKey). For example, being in possession of a NetKey allows a node to decrypt and authenticate messages up to the Network Layer to enable, for instance, relaying. However, this key does not allow data to be decrypted and an additional AppKey would be required in this case.

III. BLE MESH PDU TRANSPORT BASIS

Good throughput estimation requires to take into account real packet overheads linked to the whole protocol stack specification and time overheads associated to the delivering process across the network. On the other hand, the reliability of the network is based on repetition of messages. But repetition may be controlled by several redundant processes, managed by different parameters. In addition, BLE mesh supports transmission of unacknowledged and acknowledged messages.

The interaction between these processes and with flooding configuration parameters needs to be evaluated. In this section we will first define all the relevant procedures and parameters which affect the delivering data process across the mesh network. The analysis of the effects of the interaction of the chosen parameters setting will be included in section IV.

1) PROTOCOL STACK OVERHEAD ESTIMATION

Data is transmitted in sequence using at least one of the three advertising channels, while data transport capacity is limited by the advertising packet size and overhead introduced by the whole protocol stack from the bearer to the access layer. Although messages are usually short enough to be transported in only one ADV frame/PDU, segmentation could be also required. In any case, we will see that a low percentage of ADV PDU contains relevant data.

Fig. 2 shows an example of a non-connectable undirected advertising event. The payload length for this advertisement type is limited to 37 bytes. Nevertheless, the first 6 bytes of this payload are used to transmit the advertising address and the rest (31 bytes) must follow a concatenation of one or several specific structures (ADV structures). Each ADV structure is composed by a 1-byte length field and another 1-byte type field. For the specific case of mesh messages, the type should be $0 \times 2A$ as defined in [10]. Summarizing, this provides a maximum transport capacity for upper layers of 29 bytes from the total 47 bytes.

Fig. 3. depicts the PDU structure at the different layers defined in the mesh profile specification [4]. The Network Layer defines how packets are addressed over the network. It defines the structure to handle the parameters that allow the relaying functionality and network layer authentication using the NetKey. At this point, the following fields are needed:

- IVI: Least significant bit of the Initialization Vector Index.
- NID: Network Identification.
- CTL: Indicates the type of message (Access/Control).
- TTL: Time-To-Live.
- SEQ: Sequence number.
- SRC: Source element address.
- DST: Destination element address.
- Transport PDU: Next layer data.
- NetMIC: Network Message Integrity Check (MIC).

When transmitting an access message (CTL $= 0$) the NetMIC is reduced in 4 bytes because an additional MIC is added at the transport layer (TransMIC, Message Integrity Check for Transport).

The Lower Transport Layer defines a reliable transmission mechanism for Upper Transport PDU. To transmit Upper Transport PDUs larger than 15 bytes (11 bytes excluding TransMIC), the lower transport layer segments and reassembles them on the receiver.

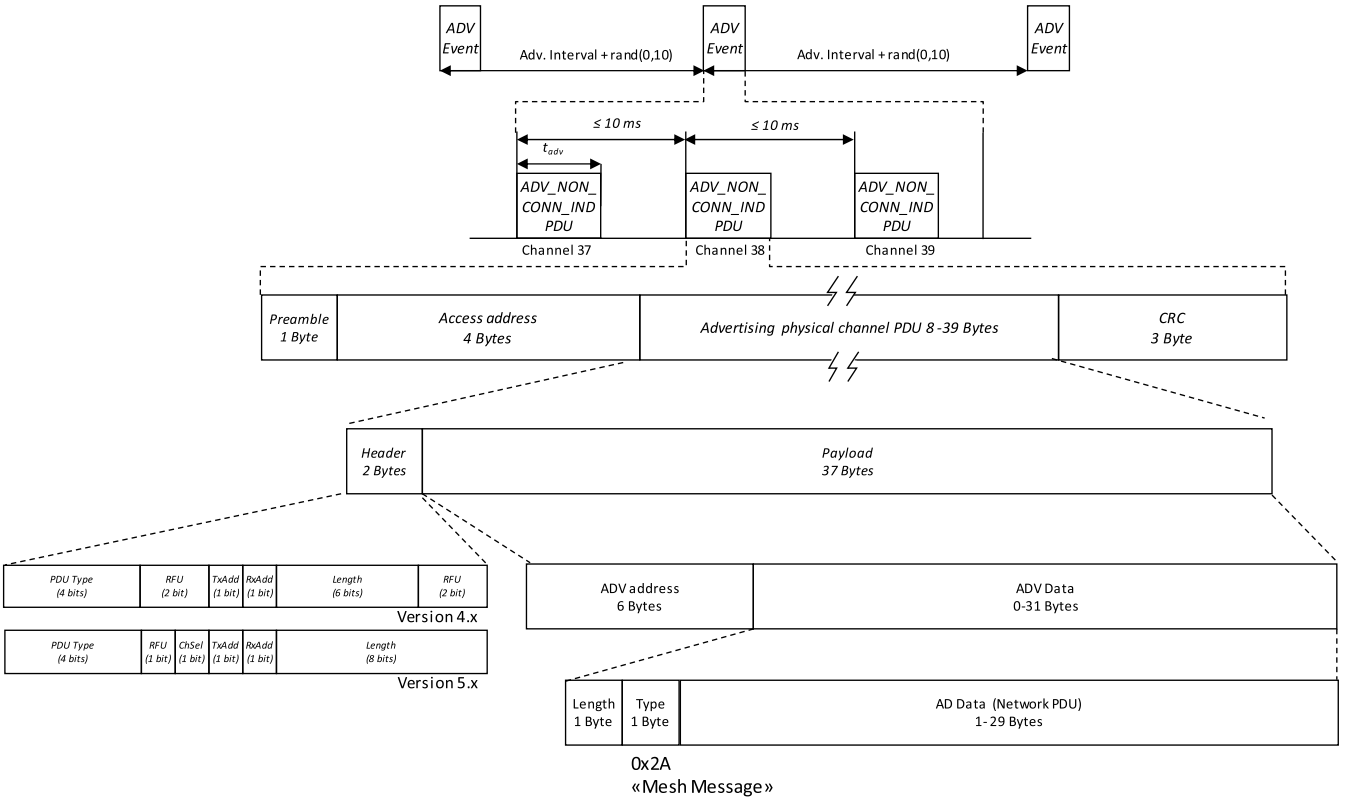

FIGURE 2. Use of BLE advertisements to send mesh messages.

Concerning the Upper Transport Layer, it takes internally generated Upper Transport Layer Control messages (up to 256 bytes PDU payload size) or access messages (up to 380 bytes PDU payload size). Access messages include a TransMIC, which allows the authentication and encryption using the AppKeys. Note that when the Upper Transport Access message is segmented at Lower Transport layer, the TransMIC is present only in the last segment. So, taking into account that the field SegN (Segment Number), which indicates the number of the current segment, has only 5 bits, it is possible to send up to 31 segments of 12 data bytes and the last with up to 8 data bytes. In conclusion, with Bluetooth mesh it is possible to transmit up to 380 data bytes from Upper Transport Layer. Yet, in the unsegmented case, the maximum available payload is only 11 bytes from the 47 bytes that composed the BLE advertisement. In addition, every Access Layer PDU, containing the Application data, shall be formed by an operation code (OpCode) plus some parameters. The list of the operation codes can be found in section 4.3.4 of [9]. The OpCode can have one byte (special messages), two bytes (standard messages) or three bytes (vendor specific). Therefore, at least one byte should be subtracted to this maximum of 380 bytes.

At the moment of this writing, most of the defined models use GET, SET and STATUS messages that are shorter than 11 bytes. Hence, they can be sent in a single unsegmented access message. As a result, the maximum user available payload is only 10 bytes from the 47 bytes that composed the BLE advertisement, a 21% efficiency. As an example, we provide the composition of the SET and STATUS messages for the Generic On/Off model in Fig. 3. It can be seen that only six and five bytes are needed, respectively.

When the PDU requires segmentation, it is necessary to take into account the limitations of the bearer. The time between segments is greater or equal than 20 ms and, therefore, the transmission efficiency is reduced even more.

2) BLE ADVERTISEMENT EVENT ADAPTATION TO MESH

Bluetooth mesh preferably uses advertising bearers, based on non-connectable and non-scannable advertising packets. A bearer is composed by the transmission of one advertising PDU in sequence (advertising event) using at least one of the three advertising channels (channel 37, 38 and/or 39).

The time between the beginning of two consecutive Advertising PDUs within an advertising event shall be less than or equal to 10 ms. The time between the start of two consecutive advertising events (affected by the imposed timing restrictions within the event) is controlled by the *advInterval* parameter ($\geq$20 ms) plus a random variable between 0 and 10 ms.

That is, according to core BLE specification, the Link Layer may enter in Advertising state, being possible to send several advertising PDUs in consecutive advertising events. Related to this, we note that reliability of managed flooding

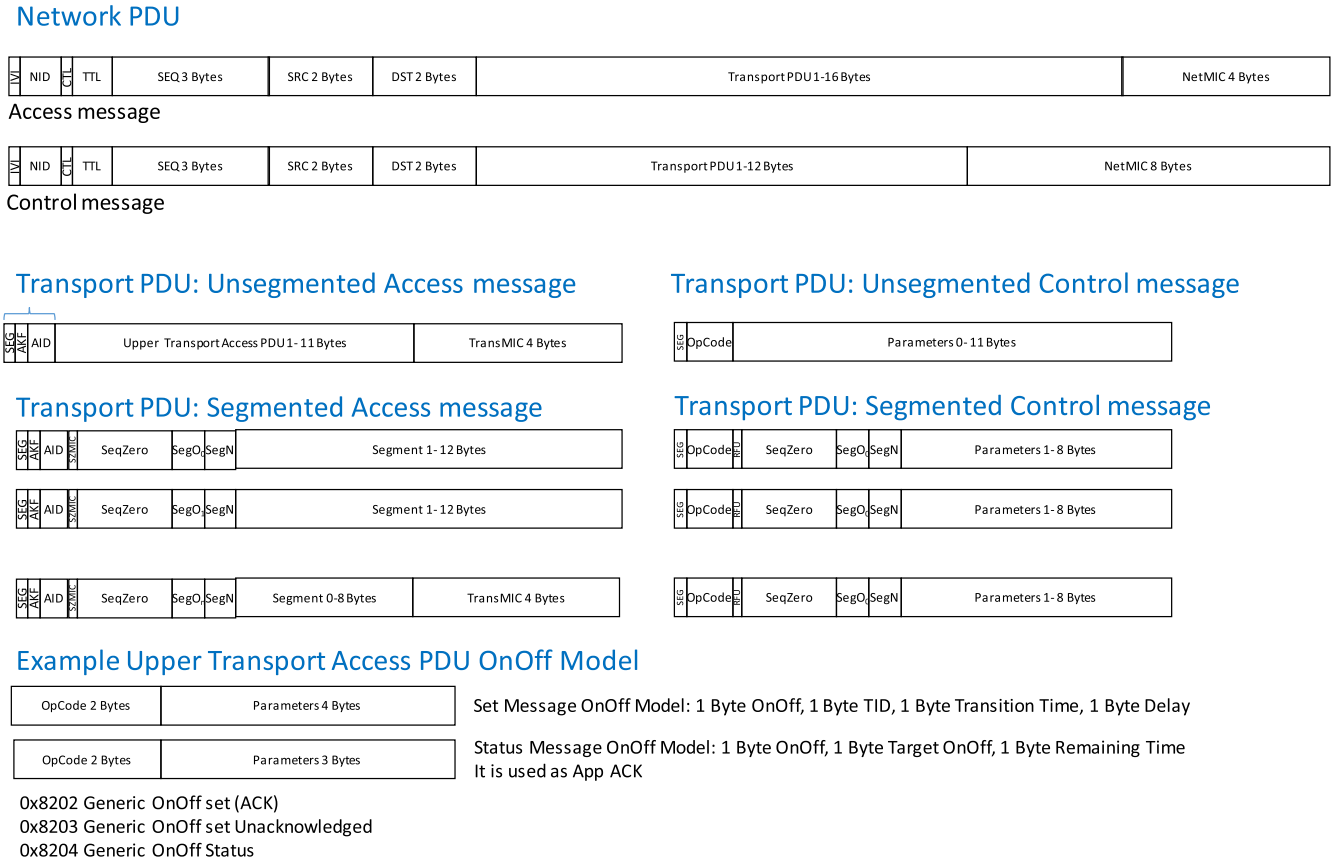

FIGURE 3. PDU structure at the different layers.

mesh networks could be improved by the repetition of Network PDUs (this implies to use several advertising events).

Nevertheless, mesh specifications indicate that BLE Link Layer shall exit the Advertising State within the advertising interval. That is, only one advertising event can be completed. This means that repetition of Network PDUs will not be supported by configuring the duration of the advertising state (link layer configuration). Instead, repetitions will be controlled by timers and parameters defined in the Network layer, being the *advInterval* parameter the minimum threshold for the defined parameters at this layer. Note that core 4.2 does not consider interleaving advertising events linked to various Network PDU transmissions. Thus, repetitions controlled at the link layer level when a node is involved in transmission or/and retransmission of several Network PDUs may not be very flexible. However, from Bluetooth 5.0 onwards it is possible to intercalate different advertising events. Advertising data belonging together are called an advertising data set. The Link Layer may support multiple sets, with each set having different advertising parameters such as advertising PDU type, advertising interval, and PHY. This feature can be used as an improvement in future mesh specifications. In fact, it could allow time between the start of two consecutive advertising PDU corresponding to different sets lower than 20 ms.

In addition, out of specification, the standard recommends that a small random delay should be introduced between receiving a mesh Network PDU and relaying a Network PDU to avoid collisions between multiple relays that have received the Network PDU at the same time.

Lastly, according to the specifications, all devices supporting only the advertising bearer should be scanning with a duty cycle as close as possible to 100%. However, many gaps affect the scanning process, disabling the radio reception: 1) time to switch between scanning channels; 2) time required to process the message (Network PDU contained on the ADV). After receiving it, the receiving node passes the message, at least, up to the network layer (if this node is not the destination, no matter it was a relay or not) or up to the application layer (if the node is the destination), where other events could be triggered (for example, sending an acknowledgement to the source node). In addition, when a node acts as relay node, scanning is affected by the required time to send the relaying message on the advertising channels.

3) BASIC MANAGED FLOODING SUPPORTING PARAMETERS ADN ADV PROCESSING

BLE mesh uses managed flooding technique, supported by two main features. Each message includes a TTL value that limits the number of times a message can be relayed and

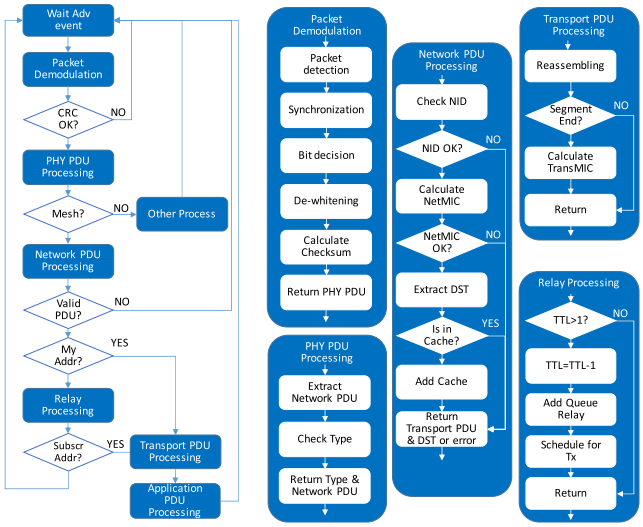

FIGURE 4. ADV processing block diagram.

to avoid unnecessary retransmissions, the nodes maintain a cache of the last received messages. The Bluetooth mesh specifications do not provide a specific value for the cache. It just requires it to be fixed to a value greater than one. However, the flooding efficiency, in addition to timing gaps, linked to the relay processing algorithm, are affected by chosen cache capacities. A low value for the cache in a dense network makes the flooding algorithm ineffective because the cache is renewed too fast. On the other side, if this value is increased, the processing delay and blind times also increase.

The Network PDU processing process is summarized in Fig. 4. Starting from the ADV detection, the receiver synchronizes during the preamble and makes the bit decision. Then, a de-whitening of the data is made with a sequence that depends on the frequency of the current scanned channel. Next, the checksum is calculated and if it matches, the advertisement data type is checked to determine whether it is a mesh message or not.

The next step is to determine if the message belongs to the same network as the node by checking the NID and NetMIC. At this point, a crucial step is to check whether the message has been received previously, i.e., it is present in the cache. If not, the message should be added to the cache and its processing should be continued. But, if it is in the cache, it should be automatically discarded.

Next, if the DST address is the unicast address of one of the elements in the node, the message should not be relayed and directly is passed to the upper layer.

The relay processing algorithm should check whether the TTL is greater than one. If true, the TTL should be decremented, the message added to a relay queue and scheduled for a future transmission.

Finally, the DST address is checked again, to see whether it matches one of the subscribed addresses of that node. If not, the process is terminated and the node should wait for the reception of the next ADV event. On the contrary, the message is delivered to the next layer, reassembled if necessary and the TransMIC calculated and checked before being passed to the application layer.

4) RELIABILITY AND MESH NETWORK PARAMETERS

The reliability of the network is based on the relay and repetition of the messages according to three different procedures and parameters, which could be redundant. One of them controlled by the model related layers and the other two by the network layer.

At the *model* level, as many replicas of an *access packet* (containing the model messages) are generated as the parameter *Publish retransmit count* establishes. These replicas are spaced between them depending on the *Publish retransmit interval steps (Pris)* parameter (5-bit value), being the retransmission interval equal to $(Pris + 1) \times 50$ ms.

Additionally, at the network layer, when a managed transport PDU (containing a segment or unsegmented *access* or *control* PDU) is originated in a node (source), it should be repeated several times as the parameter *Network transmit count* establishes and the space between these repetitions depends on the *Network transmit interval steps (Ntis)*. *Ntis* is a 5-bit value, being the transmission interval equal to $(Ntis + 1) \times 10$ ms.

Finally, also at the network layer, but linked to the relay feature, if the message comes from another node and needs to be relayed, the *Relay retransmit* procedure is applied. Similar to the previous *Network transmit*, this procedure has two parameters: *Relay retransmit count* and *Relay retransmit interval steps (Rris)*. *Rris* is a 5-bit value, being the retransmission interval equal to $(Rris + 1) \times 10$ ms.

Note that, according to the mesh specification, the minimum interval for the network transmit and relay retransmit states are 10 ms. But the bearer may impose restrictions on the set of intervals that it considers valid, for example, according with *advInterval*. Currently, the minimum required values of *Rris and Ntis* are one. Furthermore, at the link layer, each transmission managed by *Ntis* or *Rris* should be perturbed by a random value from 0 to 10 ms from the previous transmission. Random time value is not required to be added to the publish retransmission interval. Specification sets that upon receiving a *network PDU* at the Advertising Bearer Network Interface that is not tagged as relayed from the network layer (this is the case of the network PDU associated to the publish retransmit), it shall transmit the Network PDU over the advertising bearer using the value of the *Network Transmit state,* which includes *Ntis* and the random time requirement between each transmission. It is not clear that random affects the first transmission, but measurements performed with real devices implement it, according with Fig. 5. In any case, it is desirable that consecutive advertiser transmissions will be affected by a random time.

To better understand these procedures there is an example depicted in Fig. 5, assuming that unsegmented access messages are required for the application layer. In this example, the first node receives a trigger/event from the application

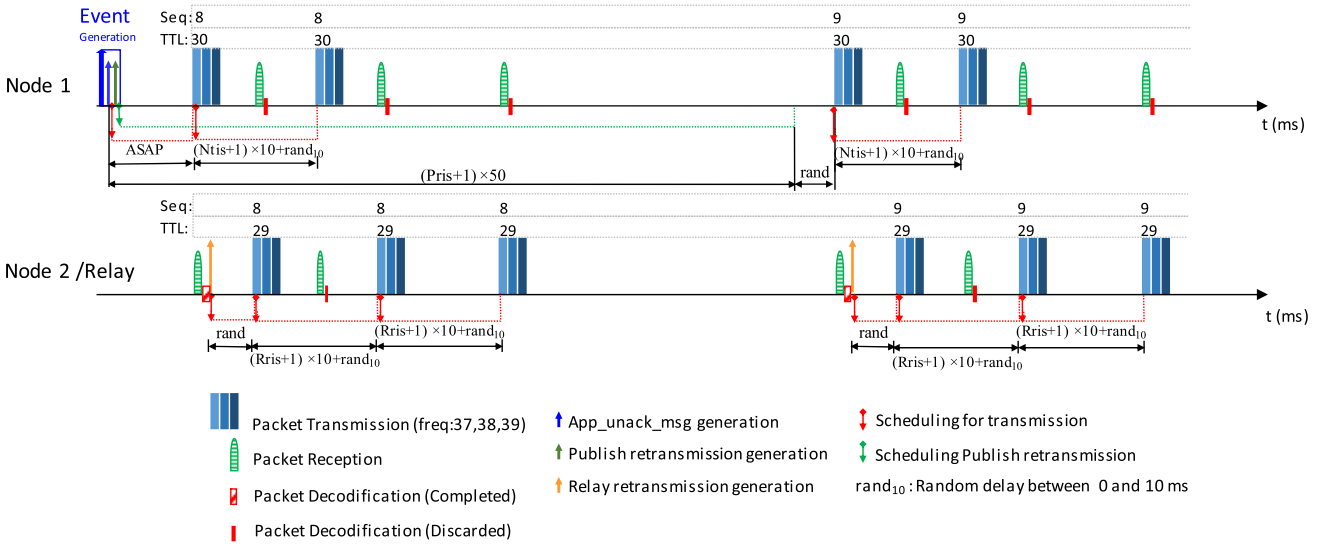

FIGURE 5. Unacknowledged message transmission and relay example.

layer that implies the transmission of one unacknowledged message (model layer) to another node. The destination address is supposed to be a group address or a unicast address not belonging to the elements of Node 1 or Node 2. Thus, Node 2, which has the relay feature enabled, should relay the message. For this example:

- *Publish retransmit count* = 1
- *Network transmit count* = 1
- *Relay retransmit count* = 2

The transmission of the message should be scheduled by the lowest layers as soon as possible. In the example, this would be the first group of three blue bars representing the three advertisement channels. This message is supposed to be set with a sequence number SEQ = 8 and a TTL = 30 at the network layer. Almost at the same time, the publish retransmission message is generated and buffered. Now, the network layer tag the corresponding transport PDU with SEQ = 9 and TTL = 30, and the link layer schedules its transmission $(Pris + 1) \times 50$ ms later.

When the first packet is effectively transmitted by node 1, following the *Network transmit* procedure, a repetition of this message (SEQ value remains equal to 8) is scheduled for $(Ntis + 1) \times 10 + rand(0, 10)$ ms later. When the packet is received and decoded by node 2 the *Relay retransmit* procedure starts. Then, the standard recommends to introduce a small random delay to avoid collisions between relays. In Fig. 5 we include this delay before transmitting the first relayed packet (SEQ = 8 and TTL = 29) and the second repetition (SEQ = 8 and TTL = 29) is scheduled $(Rris + 1) \times 10 + rand(0, 10)$ ms later. Note that prior to the transmission of this second packet, node 2 receives again a packet with SEQ = 8 and TTL = 30 from node 1. This reception is discarded because it was already in the cache. Finally, when the second relayed packet is transmitted, the third and last one is also scheduled. All these relayed packets are received and discarded by node 1.

Next, node 1 generates a new packet with SEQ = 9. This is due to the *Publish retransmit* procedure. Thus, the same *Network transmit* and *Relay retransmit* processes are repeated at node 1 and node 2.

5) ACKNOWLEDGED MESSAGES

As previously indicated, BLE mesh supports transmission of unacknowledged and acknowledged messages at the model related layer. When the unacknowledged option is set, reception is not ensured. The advertising bearer does not include acknowledgements. Nevertheless, the managed flooding protocol has an implicit redundancy. It allows the nodes to receive multiple copies of the original packet and repetitions controlled by parameters defined above. However, the redundancy level depends on node configuration. This must be suitable to the deployment scenario and traffic.

When a receiver gets a message requiring acknowledgment, it usually replies back with a STATUS message. This message is treated as any other, just having some timing differences. For instance, Fig. 6 represents an example of two nodes exchanging messages with acknowledgement. In this example:

- *Publish retransmit count* = 0
- *Network transmit count* = 2
- *Relay retransmit count* = N/A (No relay)

The process is very similar to the previous example. The main difference is that when the packet is received and processed at node 2 a new event in response is generated. This event schedules the transmission of the acknowledgment message (SEQ = 67) after a random delay ($rand_{ACK}$). When the acknowledgement is the response to a message that was sent to a unicast address, $rand_{ACK}$ should be between 20

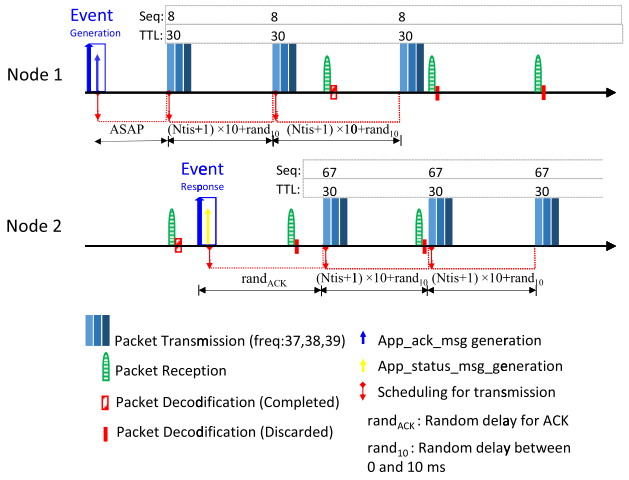

FIGURE 6. Acknowledged message transmission example.

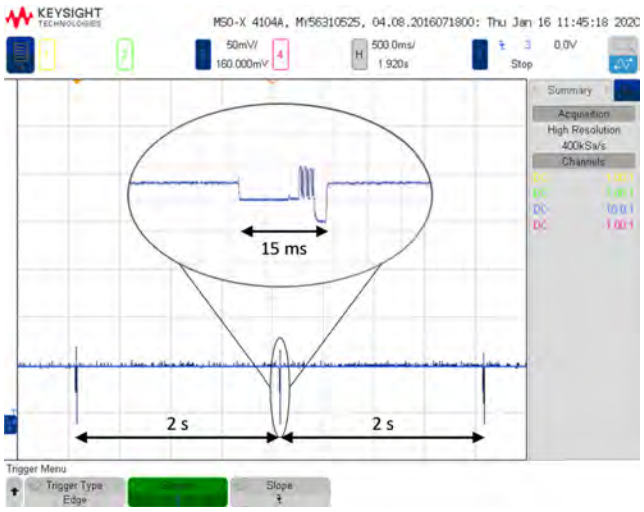

FIGURE 7. Blind times due to connectable advertising packets.

and 50 ms. But, if it was sent to a group address or a virtual address, rand_{ACK} should be between 20 and 500 ms to reduce the probability of collisions. Note that, if *Publish retransmit count* is greater than 0, at this moment another packet should be generated and scheduled following the *Publish retransmit* procedure as explained before.

IV. ISSUES AND IMPLEMENTATIONS

Depending on the selected configuration and firmware implementation several issues could affect the performance of a Bluetooth mesh network. We will analyze some of them, next.

A. BLIND TIMES

Real Bluetooth devices present non-idealities in their operation that need to be considered when performance is estimated although we cannot act modifying them. From the point of view of the overall network performance, maybe the most important impairments are the ones that affect to the scanning procedure of the Bluetooth devices. Specially, in a Bluetooth mesh network where the specifications state that the relays or the friend nodes should be scanning as close to 100% of the time as possible. In spite of this, as we stated before, due to firmware implementations or hardware limitations, during some time the scanners eventually leave this state and any packet received within these periods is lost. We called these intervals blind times. We analyzed some of these non-idealities in our previous works [11], [12]. They affect detection probabilities and thus, the probability that a message reaches the destination.

The reasons behind a blind time are diverse:

- When commuting from one scanning frequency to another. We have observed blind times of these type up to 16 ms [12].
- Caused by the processing of Bluetooth packets. When a Bluetooth packet is detected, the scanner exits the scanning state until it completely processes the packet and determines whether it belongs to the network and the actions required upon its reception. The cache size also affects to the processing delay and thus blind times.
- Excluding the expected time spent in retransmissions of received PDUs (when the node is a relay node), relay and destination nodes are also involved in the emission of connectable advertising packets. These packets are sent periodically to allow the reconfiguration of the network. For example, in the Nordic BLE mesh implementation the devices present a blind time of 15 ms within periods of 2 s as can be seen in Fig. 7.

B. BUFFERING CAPACITIES VS. NETWORK, RELAY AND PUBLISH RETRANSMISIONS PARAMETER SETTING

Apart from the cache, actual implementations may include other buffers used, for example, to store the packets before they can be transmitted at the physical interface. These buffers, depending on their size, may introduce additional issues. For example, increased delay vs. undesired discarding of messages. This we will be discussed in the following paragraphs.

A priori, mesh implementations compatible with core BLE 4.2 manage only one buffer of events at the link layer. However, from v5.0, several event sets can be managed and interleaved. The absence of details in the specification limits the definition of a general framework concerning the number of managed buffers in mesh implementation. Yet, based on experimental evaluation of real chipsets, we have identified at least two separated buffers at relay nodes that also act as source/destination nodes: one buffer for storing relayed PDUs and one for storing Network PDUs originated in the node.

Focusing exclusively on the relay buffer, reducing the size of the relay buffer increases the probability of discarding processed PDUs. However, this reduction limits flooding and potential congestion. In some real implementations, it has been observed that the size is limited to two PDUs. If no repetitions are configured in the model (*Publish, Network and Relay retransmit count* are equal to zero), since relay

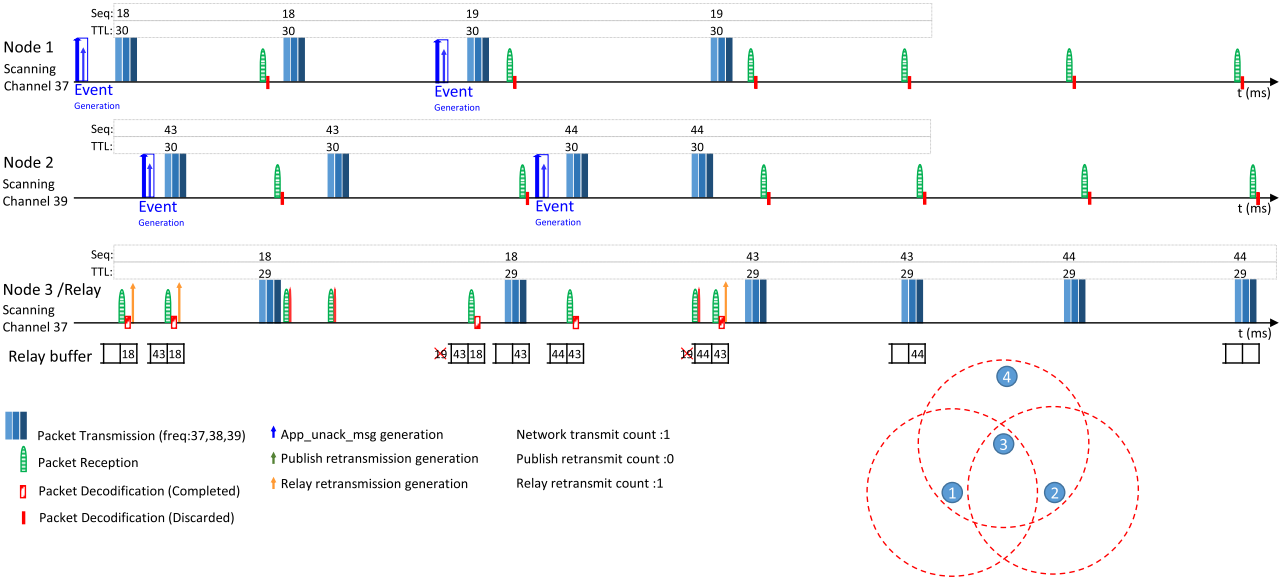

FIGURE 8. Relay buffer implementation example.

retransmits the received PDU as soon as possible (only perturbed by a small random delay to avoid collisions between multiple relays that have received the Network PDU at the same time), the number of discarded PDU could be negligible in most scenarios. Firstly, the processing time of Adv PDU is usually a blind time. As a result, the relay device is only able to detect new Network PDUs (Adv) in CH=37 in the random period defined after the previous Network PDU process. Secondly, the random period, if defined, is small. Thus, it is expected that the probability of receiving more than one PDU in the random period is low. The minimum configurable *advInterval* is 20 ms, so, buffering more than one PDU is only required when the relay node is involved in relaying PDUs from different source nodes.

If no repetitions are configured in the network, buffering more than one packet is only required when the relay node receives inputs from different source nodes. Nevertheless, when repetitions are configured, the relay node stores a received Network PDU until all repetitions are completed. That is, if R relay repetitions are configured (*Relay retransmit count* is equal to R), the time is $R\times[(Rris+1)\times 10+rand(0,10)]$ ms. This means that the probability of receiving a new Network PDU in this period is significantly greater, and thus, if the buffer size is limited, PDU discarding probability may be high.

Relay repetition could improve the reliability of one message node, since it provides diversity against possible errors or collisions. But, depending on the buffer size, it may affect the relay operation for additional transmissions.

The effect is significant if Access PDUs require segmentation. Even when a relay node is only simultaneously involved in the retransmission of messages from two source nodes. The time between the beginning of two consecutive transmissions of segments is expected to be as short as possible. That is, in BLE v. 4.2, this time will be probably *advInterval* or $(N+1)\times[(Ntis+1)\times 10+rand(0,10)]$ ms, when *Network transmit count* = N is greater than zero. This means that, depending on relay retransmit count and *Rris*, the probability of transmission of a new segment while the buffer is full is very high.

Note that concerning to Publish or Network repetition, the implications are different from the point of view of the relay operation. General discussion about Publish, Network and Relay repetition count parameter setting will be included later.

In Fig. 8 we depict an example scenario showing the behavior of a relay node, according with a buffer real implementation, where the buffer is able to store just two messages. Fig. 8 represents in an easy and visual way the result of a real monitoring of chipsets obtained in laboratory measurements.

At the lower right of the figure the topology and layout of the nodes is represented. Node 1, node 2 and 4 are out of coverage from each other. Node 1 and node 2 generate data with destination to node 4. Applications at node 1 and 2 generate unacknowledged messages. The data reaches this node through node 3, which acts as a relay. We do not make use of the Publish retransmit procedure (*Publish retransmit count = 0*). However, the *Network transmit count* and *Relay retransmit count* are configured to a value of 1. This means that when a new event is generated at the sources or a packet is received at the relay the packet is transmitted twice in both cases.

So, the example works as follows: first, an event is generated at Node 1 and a message with TTL=30 and Sequence number = 18 is prepared and sent over the three advertisement channels as soon as possible. This message is received and decoded successfully by the relay (node 3). Thus, the relay schedules the future transmission of the packet (Seq = 18, TTL = 29) and stores it in its buffer.

Then, before the relay can effectively transmit the first of the two retransmissions, Node 2 generates another event. This event has a message associated with Seq = 43 and TTL = 30 and it is the second message sent to the air interface. The packet is also received successfully by the relay. Hence, it stores the message at the second and last position of the buffer and schedules its future transmission.

The next message over the air is sent by the relay (Seq = 18, TTL = 29). Node 1 receives this message on channel 37 and node 2 on channel 39. Both of them discard the message because they do not have the relay feature set nor they are the destination of the message. Note that packet with Seq = 18 remains at the relay buffer because a second retransmission is still scheduled (*relay retransmit count* = 1). This message is also received and correctly processed by node 4 although this is not shown in the figure.

Next, Node 1 does the retransmission of packet with Seq = 18 and TTL = 30 (*network transmit count* = 1). The relay discards this packet because it was received recently, so it is in the cache. The same happens with the retransmission of packet with Seq = 43, TTL = 30 from node 2.

The key point of the example comes now: node 1 generates another event (Seq = 19, TTL = 30). This message is transmitted and received successfully by the relay. However, the relay buffer is full at this moment, so the device cannot store the message and it is discarded.

Just after this, the relay makes the second retransmission of packet (Seq=18, TTL=29), which is again received and discarded by node 1 and 2. As this was the last retransmission of this packet, it is removed from the relay buffer leaving a free slot.

To illustrate the issue under discussion, it is supposed that now, node 2 generates another event (Seq = 44, TTL = 30). This message is received at the relay and introduced at the relay queue, filling it up again.

Before the relay transmits the first packet with (Seq = 43, TTL = 29) both, node 2 and 1, make the second retransmission of their last packet. Both packets are discarded at the relay. The first one because the packet was in the cache and the second one because the buffer is completely full again. So, packet with Seq = 19 and TTL = 30 is definitely lost and will never arrive at its destination.

The example finishes transmitting twice the two packets stored and so emptying the buffer.

With this example we have illustrated that not only the cache is important. It is also necessary to take into account during the design other buffers that could affect unexpectedly to the network performance.

Assuming the relay buffer to be very limited, the questions to answer by the network configurator, depending on the context scenario, may include for example:

Is it better to improve the reliability using publish retransmission instead of network/relay repetition or both?

What is the probability that the repetition of a publish overlaps original public transmission process along the mesh network? *Pris* needs to be set according to the size of the mesh network?

C. NETWORK, RELAY AND PUBLISH RETRANSMISIONS PARAMETER SETTING

Network, relay and publish repetitions impact directly on the reliability of the network because they provide diversity against possible errors or collisions over the different links. However, high values in the number of repetitions and time intervals between repetitions produce increased delays, network saturation and a significant reduction of the throughput depending on the scenario.

Related to the increased delays, even if no concurrent message transmissions occur in the mesh network between several source and destination nodes, the major impact concerns to network and relay parameter configuration. Nevertheless, it depends on the link/physical level BLE core specification. When BLE v.4.2 is considered (mesh specification is defined compatible with this specification), the link layer manages advertising events sequentially. This means that, if a source node transmits a segmented Access PDU or a sequence of unsegmented Access PDUs, waiting PDUs are affected by repetition processes of the previous ones. Out of other components that impact reception/transmission of messages (interference, non-overlapping scanning and advertising channels, discarded received packets due to buffer overflow, etc.) and, in case of multihop communication involving n relays (equally configured), the minimum trip time without packet loss for the Mth subsequent Network PDU (or Mth segment) can be given (in a simplified way) by (1):

$$t = T_{adv} + \sum_{i=1}^{n} (t_{RXdelay,i} + t_{rand} + T_{adv,i}) + t_{processing,total} + (M-1) \cdot maximum(t_{NetworkTransmitConf}, t_{RelayTransmitConf}) \quad (1)$$

where, $t_{NetworkTransmitConf} = (N+1) \times [(Ntis+1) \times 10 + rand(0,10)]$ and $t_{RelayTransmitConf} = (R+1) \times [(Rris+1) \times 10 + rand(0,10)]$, being N and $Ntis$, the network transmit count and interval steps, respectively, and R and $Rris$, the relay retransmit count and number of interval steps, respectively. $t_{RXdelay,i}$ is a reception delay, depending on which channel (CH=37, 38, or 39) the ADV PDU is received on. $T_{adv} = T_{adv,i}$ is the transmission time of an ADV PDU and t_{rand} is the small random delay that the BLE specification recommends to be introduced between receiving a mesh Network PDU and relaying it. Finally $t_{processing,total}$ is the sum of the processing times in all the nodes. Note that if $Rris$=$Ntis$, no matter the N and R values, the delay depends on the highest of them (N or R), but the load depends on the sum of them ($N + R$).

The minimum delay would be significantly reduced if the Network Layer was able to instruct the Link Layer to interleave advertising events. Similar to the support of multiple advertising data sets, defined in BLE v.5.0 specification.

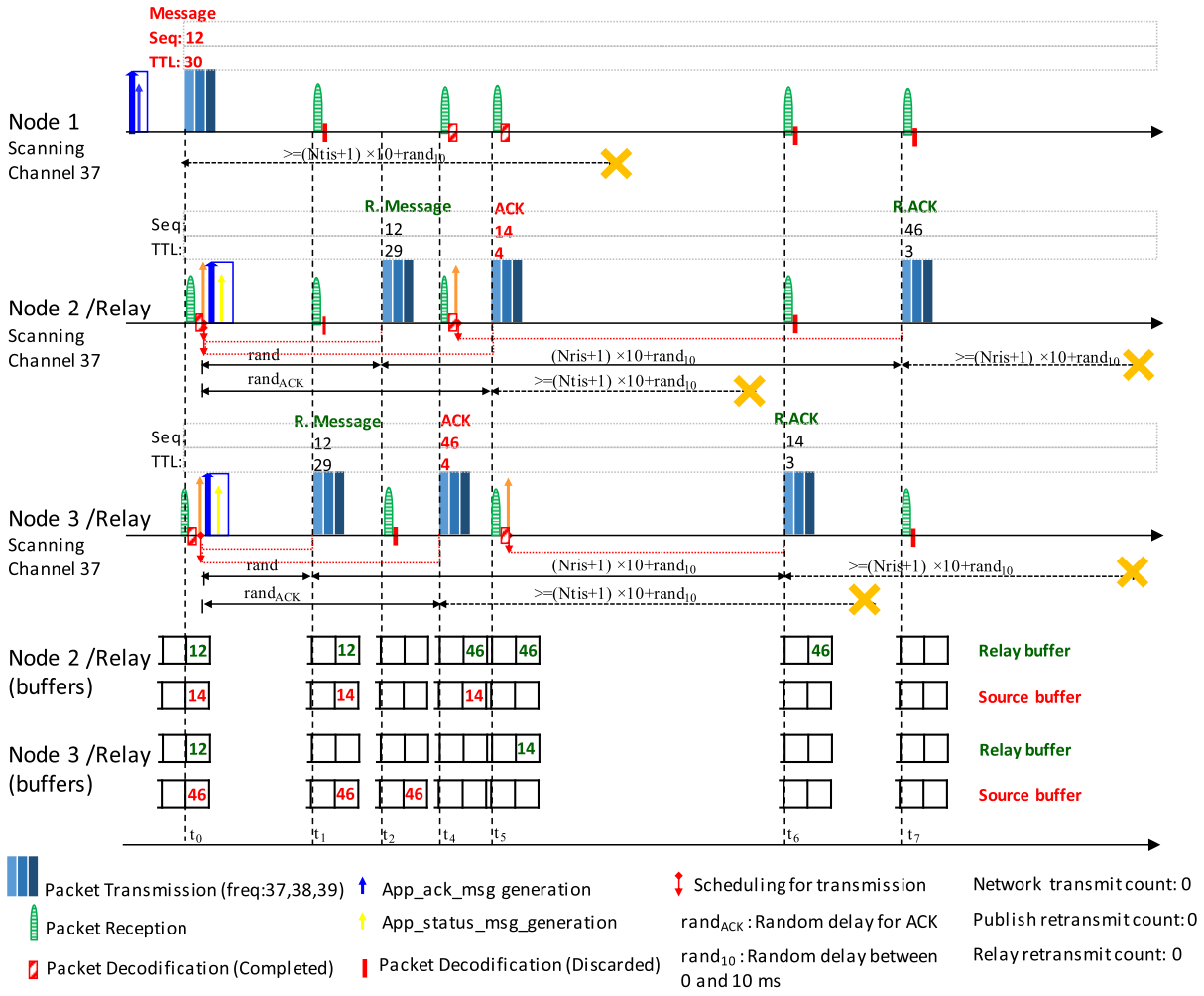

FIGURE 9. Timing between consecutive transmissions. Relay and source buffer state.

In ideal conditions, it would be given by (2).

$$t = T_{adv} + \sum_{i=1}^{n} (t_{RXdelay,i} + t_{rand} + T_{adv,i}) + t_{processing,total} \tag{2}$$

Obviously, when having multiple source/relay nodes and in real operation conditions, the exact timings depend on the specific channel conditions between neighbor nodes which are listening to each other and on collision probability between ADV transmissions, in addition to firmware implementation.

In this case, *N, R, Ntis, Rris* need to be chosen separately for each source and relay node depending on their own neighboring context (channel conditions and rate of interfering transmissions).

D. TIMING RELATIONSHIPS BETWEEN CONSECUTIVE TRANSMISSIONS AND APPROPRIATE SELECTION OF SOME RANDOM VALUES

The specification does not determine explicitly some random values such as the random delay before relaying a message (see Fig. 5 or Fig. 9). Inappropriate selection of these values and additional timing relationships between consecutive (re)transmissions could increase the collisions or the expected delay. A first restrictive condition appears between relay nodes that receive a mesh Network PDU at the same Advertising Event and relay it. Relay nodes may receive the advertising PDU in the same or a different channel, but a relay advertising event always starts at channel 37. We define the time between the beginning of two consecutive ADV PDUs within the Advertising Event as D. This means that the difference between receptions of relay nodes may be $T_{RX,delay} = D$ or $T_{RX,delay} = 2D$. The probability of collision depends on the random backoff applied (t_{rand}), but may also be conditioned by the D value, particularly if the D value is close to the backoff. D shall be less than or equal to 10 ms. Nevertheless, this value is very conservative. In real implementations, D is significantly lower. For example, when the highest ADV frame size is considered (47 bytes), the transmission time, $T_{adv,}$ is 376 μs, while measured D is even below 500 μs (exactly 430 μs).

The probability that a transmission from a reference node (started in a time instant t) collides with another one (P_{col}),

corresponds with the probability that one neighbor node starts its own transmission in the interval $[t-T_{adv}, t+T_{adv}]$. When $T_{RX,delay}$ between relays is 0, if random backoff is t_{rand}, the probability of collision between two relays is $2 \cdot T_{adv}/\overline{t_{rand}}$. And, if N_{relays} relays are involved, it is (3):

$$P_{col} = 1 - (1 - 2 \cdot T_{adv}/\overline{t_{rand}})^{N_{relay}-1} \quad (3)$$

In the specification, as we referred above, random is suggested but not specified. As an example, we observed in real implementations that a random value from 0 to 10 ms is set, similar to the added after each transmission managed by Network Transmit or Relay Retransmit parameters (*Ntis* or *Rris)*. If so, collision probability between two relays is 15%. The value is not negligible. This implies that not only random selection is important, but also selecting which nodes of the network should act as a relay, or even, physical channels selected for transmission on advertising events.

The second restrictive condition concerns timing relationships between consecutive (re)transmissions. To better illustrate some aspects to be considered, Fig. 9 depicts an example scenario showing the behavior of two relay nodes (node 2 and 3), acting also as destination of a message originated in node 1. For instance, they could represent light bulbs of a light application that can be turned on/off simultaneously. In this case, both send response messages (Acknowledgment). Fig. 9 represents the result of a real monitoring of chipsets obtained in laboratory measurements, where *Rris* and *Ntis* were equal to 1 That is, the minimum time interval is 20 ms.

The specification sets that the advertising bearer, upon receiving a Network PDU that is not tagged as relayed from the network layer, shall retransmit the Network PDU over de bearer using Network Transmit State (parameter *Ntis*); and, upon receiving a PDU tagged as relayed from the Network layer, using the value of Relay Retransmit state (parameter *Rris*). This means that not only Network Transmit Interval and Relay Retransmit Interval apply between repetitions of the same Network PDU as explained in section III. They control, respectively, the minimum time interval between any message transmissions originated from the node and between any Network PDU relayed by a node. Currently, the minimum *Ntis* and *Rris* are required to be 1 to meet v4.2 bearer restrictions. For example, in Fig. 9, once node 2 receives the ACK (Seq = 46, TTL = 4) from node 3, node 2 retransmits it *(Rris+1)×10+rand(0,10)* ms after the previous PDU retransmission (Seq = 12, TTL = 29), Specifically, this time is 20+*rand(0,10)* ms. However, as shown in Fig. 9, the time between two consecutive transmissions of the advertising bearer can be really shorter (in this case, shorter than 20 ms). Indeed, nodes can manage two buffers (for PDUs retransmitted and for PDUs originated in the node), similar to the interleaved Advertising Sets defined in v5.0. This is the reason why in node 2 the time between relaying the Message received from node 1 (Seq = 12, TTL = 29) and the originated ACK (Seq = 14, TTL = 4) is shorter than 20 ms. Thus, out of interleaved advertising sets, the minimum time between ADV transmissions is $20+rand_{10}$ ms. A fixed delay of 20 ms could be considered unnecessarily long in many scenarios. Nevertheless, the minimum collision probability between two nodes (configured equally, with $t_{interADV}$ = *(Xris+1)×10+rand(0,10)*, being *Xris = Ntis = Rris*) involved in independent transmissions is $2 \cdot T_{adv}/\overline{t_{\text{interADV}}}$. *If Xris = 1,* collision probability is around 3%. The value is not negligible. If the number of nodes (N_{nodes}) grows, the probability is (4):

$$P_{col} = 1 - (1 - 2 \cdot T_{adv}/\overline{t_{\text{interADV}}})^{N_{nodes}-1} \quad (4)$$

Besides, the collision probability may be greater if nodes are involved in relaying the same PDUs as explained above and considering the two interleaved Advertising sets. Thus, parameters *Ntis* and *Rris* need to be adequately set in each scenario to achieve a tradeoff between collisions and delay.

In addition, the random delay before transmitting an ACK message ($rand_{ACK}$) is specified to be longer than 20 ms (from 20 to 50 or 500 ms, for unicast or group destinations). Upper bound of this random variable needs to be adequately set according to the number of expected nodes to control collisions while avoiding unnecessary delays.

E. ADDRESSING AND ACKNOWLEDGEMENTS

Some configurations of the addressing and type of acknowledgement could generate unexpected issues. For example, let's suppose a case where a source sends a message to a group address and that message needs acknowledgement. When just one device subscribed to this group address answers with the ACK, the source would consider that the message has reached its destination. However, it is possible that many other devices that are also subscribed to this address may not have received the message. Considering multicast scenarios, it only makes sense to request acknowledgements if it is enough to just receive an ACK from one device.

F. OTHER NETWORK PARAMETER ADJUSTMENTS

There are several other network feature configuration and parameters that, if they are not adjusted correctly, could underperform or even disrupt the communications of the mesh network. The most important are:

TTL Adjustment: the results of a wrong setting of this parameter could produce from the messages not reaching its destination (low value) to the saturation of the network with unnecessary messages (high value). On the other hand, when acknowledgement messages are considered (for example, STATUS messages), TTL configuration for ACK has to be consistent with the network dimension.

The aim may be learning dynamically the number of required hops and set the TTL value for each model. It may be an easy objective considering the model paradigm, but requires cross layer communication between Network and Model Layers.

Relay Feature: apart from the collisions seen in the previous section, selecting which nodes of the network should

act as a relay affects the energy consumption, saturation and reliability of the network.

V. RESEARCH CHALLENGES

Mesh Bluetooth specification has achieved a simple and functional implementation. However, as it is shown in the previous sections, it has several issues that limit its performance. This section presents some research challenges that deserve some focus and that could help to outperform such problems.

1) SELF OPTIMIZED MESHED NETWORK (SON)

Along the previous sections, it has been shown that there is a great deal of parameters that affect the performance of Bluetooth mesh networks. Parameters such as network transmit, publish and relay retransmit counts, interval steps and time-to-life need to be correctly tuned. Thus, each network deployment may require a manual and individual configuration of each of the devices. This is a problem in networks with many nodes. In addition, the optimal values of these parameters will depend on the network layout, propagation conditions, interferences, QoS required for each service, etc.

Given this, dynamic and self-organized Bluetooth mesh networks constitute an interesting research topic. Such procedures should perform a correct tuning of all the parameters, without manual actions. The network would dynamically self-configure and allow an optimized plug-and-play operation. Also, they should react to failures (self-healing) and optimize the configuration over time as the network evolves in time with new devices, services or data traffic volume (self-optimization).

2) SOFT COMBINING

The managed flooding protocol allows the nodes to receive multiple copies of the original packet. The copies could be soft combined to increase the reliability of transmissions. The receiver could easily implement maximum ratio combining and improve the final code word error probabilities. Also, the transmitter can generate different sets of coded bits, different incremental redundancy versions. If systematic bits are always included, each packet would still be self-decodable. Such techniques would allow making the most of the inherent redundancy at the cost of a slight increase in the processing requirements. However, such redundancy pretty much depends on the network layout and parameter configuration.

Despite the standard is not contemplating this option, the frame structure would allow a simple implementation. This may be of particular interest in scenarios where it is necessary to increase the range or overcome difficult attenuation situations. Hence, studies about the limits and gains of this approach for Bluetooth mesh networks constitute an interesting research area.

3) BLUETOOTH 5 AND NEWBEARERS

The first version of Bluetooth mesh uses non-connectable (broadcast only) and non-scannable advertising packets as a bearer. This type of packet entails a limitation in the amount of information that can be sent, the latency, immunity against interferences, the implementation of ACK protocols and eventually, in the communication throughput. This may limit the scenarios where this technology suits in.

However, out of the mesh specifications, Bluetooth offers a wide variety of transmission modes that would increase current performance and achieve greater operation flexibility. For example, the use of scannable advertising beacons would allow to double the amount of bits sent. Moreover, any receiver could reply back through a scan request that could be used for ACK purposes. Also, Bluetooth 5 introduces new extended advertising PDUs with even larger payloads of up to 254 bytes (instead of 31). This can be further extended by using pointers to other packets, i.e. packet chains. In addition, secondary advertising messages have also been introduced. Advertising messages used to happen on 3 of the 40 available channels in the 2.4 GHz band. But the new secondary ones can use the 37 channels traditionally reserved for data. This fact, combined with multiple retransmissions, would allow the implementation of diversity techniques that would increase immunity against interference.

4) ANGLE OF ARRIVAL AND DEPARTURE

Bluetooth 5.1 specifications have incorporated procedures to estimate Angle of Arrival (AoA) and Angle of Departure (AoD) of the received and transmitted signal, respectively. This information is aimed for location services but could also be exploited by mesh deployments. AoA and AoD require that receiver and transmitter respectively have an array of antennas, which opens the door to introduce beamforming Such features could be used to increase the efficiency, performance and network range. Protection against interference can be increased and leads to lower bit error probability, lower consumption and enabling the implementation of routing protocols based on the topology.

5) ENERGY CONSUMPTION CONSIDERATION

The mesh Bluetooth standard contemplates the use of low power consumption nodes. This functionality is reserved for applications which emit information periodically and that are delay tolerant, e.g. sensor reporting. However, relay nodes require to be configured in continuous reception. The consumption of a device in continuous scanning mode is about 5 mA [13], [14] meaning that a node with a battery of 10^4 mAh would have an autonomy of just 83 days. This is clearly insufficient and indeed problematic in networks having many nodes. As a consequence, relay nodes would require an external power supply, either from the power grid or through energy harvesting techniques.

Hence, it remains an open issue how to deploy Bluetooth mesh networks in power limited scenarios. It is required to propose smart techniques that allow to reduce sensing times, for example, by examining the benefits of node syncing. Bluetooth periodic advertising could be rethought for mesh networking to allow the relay (acting as a scanner) to sync

with the transmitter. This means that both devices are able to wake up at the same time. Duty cycles and activity time slots could be tuned to avoid constant scanning. Synchronization information could be transferred through the mesh network by means of the new Bluetooth 5.1 periodic advertising sync transfer (PAST) feature [5]. Reducing advertising/scan (transmission/reception) times may yield to increased packet collision. For this reason, collision avoidance methods would require especial care, simple examples are random back-off times and random selection of advertising channels.

VI. CONCLUSION

Mesh specifications enable a variety of new applications, allowing BLE to enter even more into the world of IIoT, massive sensor networks, smart buildings and smart cities, etc.

However, the parameters to configure the network are so wide that optimizing them can become a brainteaser. Even some of them are not fully specified in the standard.

Along the paper we have discussed most of these parameters, pointing the relationships and interaction between them and the problems that arise when they are not properly configured. This is done covering all the layers of the protocol stack, from the bearer to the model. We have even considered and evaluated on a test-bed real device limitations such as blind times or buffering problems due to manufacturer implementations.

Finally, we have highlighted some research challenges that may improve BLE mesh networks: self-optimized networks, soft combining, the use of different bearers, the application of angle of arrival and angle of departure, etc. And last, but not least, we have placed the focus on energy consumption because, with mesh, BLE losses some of its low-energy objectives.

REFERENCES

[1] Silicon Labs. *AN1142: Mesh Network Performance Comparison*. Accessed: Mar. 3, 2020. [Online]. Available: https://www.silabs.com/documents/public/application-notes/an1142-mesh-network-performance-comparison.pdf

[2] A. Cilfone, L. Davoli, L. Belli, and G. Ferrari, "Wireless mesh networking: An IoT-oriented perspective survey on relevant technologies," *Future Internet*, vol. 11, no. 4, p. 99, 2019.

[3] J. Yin, Z. Yang, H. Cao, T. Liu, Z. Zhou, and C. Wu, "A survey on Bluetooth 5.0 and mesh: New milestones of IoT," *ACM Trans. Sen. Netw.*, vol. 15, May 2019, Art. no. 28.

[4] *Mesh Profile Bluetooth Specification V1.0.1*, Bluetooth SIG, Kirkland, WA, USA, 2019.

[5] Bluetooth SIG. (2019). *Bluetooth Core Specification 5.1*. Accessed: Oct. 17, 2019. [Online]. Available: https://www.bluetooth.com/specifications/bluetooth-core-specification/

[6] S. Darroudi and C. Gomez, "Bluetooth low energy mesh networks: A survey," *Sensors*, vol. 17, no. 7, p. 1467, May 2017.

[7] S. Sirur, P. Juturu, H. P. Gupta, P. R. Serikar, Y. K. Reddy, S. Barak, and B. Kim, "A mesh network for mobile devices using Bluetooth low energy," in *Proc. IEEE SENSORS*, Nov. 2015, pp. 1–4.

[8] Y. Murillo, B. Reynders, A. Chiumento, S. Malik, P. Crombez, and S. Pollin, "Bluetooth now or low energy: Should BLE mesh become a flooding or connection oriented network?" in *Proc. IEEE 28th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Oct. 2017, pp. 1–6.

[9] *Mesh Model Bluetooth Specification V1.0.1*, Bluetooth SIG, Kirkland, WA, USA, 2019.

[10] Bluetooth SIG. *Bluetooth SIG Assigned Numbers: Generic Access Profile*. Accessed: Mar. 3, 2020. [Online]. Available: https://www.bluetooth.com/specifications /assigned-numbers/generic-access-profile/

[11] Á. Hernández-Solana, D. Perez-Diaz-de-Cerio, A. Valdovinos, and J. L. Valenzuela, "Proposal and evaluation of BLE discovery process based on new features of Bluetooth 5.0," *Sensors*, vol. 17, no. 9, p. 1988, Aug. 2017.

[12] D. P.-D. de Cerio, Á. Hernández, J. Valenzuela, and A. Valdovinos, "Analytical and experimental performance evaluation of BLE neighbor discovery process including non-idealities of real chipsets," *Sensors*, vol. 17, no. 3, p. 499, 2017.

[13] Nordic Semiconductor. *nRF52840 Product Specification V1.1*. Accessed: Jan. 16, 2020. [Online]. Available: https://infocenter.nordicsemi.com/pdf/nRF52840_PS_v1.1.pdf

[14] Nordic Semiconductor. *nRF52832 Product Specification V1.4*. Accessed: Jan. 17, 2020. [Online]. Available: https://infocenter.nordicsemi.com/pdf/nRF52832_PS_v1.4.pdf

ÁNGELA HERNÁNDEZ-SOLANA received the degree in telecommunications engineering and the Ph.D. degree from the Universitat Politècnica de Catalunya (UPC), Spain, in 1997 and 2005, respectively. She has been working with the UPC and UNIZAR, where she has been an Associate Professor, since 2010. She is a member of the Aragón Institute of Engineering Research (I3A), University of Zaragoza. Her research interests include 5G/4G technologies, heterogeneous communication networks, and mission-critical communication networks, with an emphasis on transmission techniques, radio resource management and the quality of service, mobility management and planning, and the dimensioning of mobile networks.

DAVID PÉREZ-DÍAZ-DE-CERIO received the M.Sc. degree in telecommunications engineering, in 2003, and the Ph.D. degree in telecommunication from the Universitat Politècnica de Catalunya (UPC), in 2010. In 2003, he joined the WiComTec Research Group of the Department of Signal Theory and Communications as a Collaborating Lecturer. All his lectures are held at the Escola d'Enginyeria de Telecomunicació i Aeroespacial de Catalunya for the bachelor's degree, second-cycle, and MAST master's degree students. He has participated as a consultant of several local and European projects in addition to other public and private funded projects. His research interests are wireless communications systems, especially those based on the IEEE 802.X standards and their use in e-health applications and the IoT.

MARIO GARCÍA-LOZANO received the Ph.D. degree in telecommunications engineering from the Universitat Politècnica de Catalunya (UPC), Barcelona, in 2009. He has more than 20 years of experience in different radio network planning and optimization activities both at the academia and industry. He is currently an Associate Professor with the UPC, and his research activities are focused in the fields of radio resource management and the optimization wireless networks. He has actively participated in more than 25 competitive research projects and contracts with the industry. He was a recipient of three best paper awards and was the advisor of the student team that won the international competition in mobile network planning organized by the company ATD.

ANTONIO VALDOVINOS BARDAJÍ received the degree in telecommunications engineering and the Ph.D. degree from the Universitat Politècnica de Catalunya (UPC), Spain, in 1990 and 1994, respectively. He has been working with the UPC and the University of Zaragoza (UZ), where he has been a Full Professor, since 2003. His research interests include 5G/4G technologies, heterogeneous communication networks, and mission-critical communication networks, with an emphasis on transmission techniques, radio resource management and the quality of service, mobility management, and planning and dimensioning of mobile networks.

JOSÉ-LUIS VALENZUELA received the degree in telecommunications engineering and the Ph.D. degree from the Universitat Politècnica de Catalunya (UPC), Barcelona, Spain, in 1993 and 1997, respectively. He is currently an Associate Professor with the Department of Signal Theory and Communications, UPC. After graduation, he was concerned with equalization techniques for digital systems. He has also been working on the field of digital communications with a particular emphasis on digital radio and its performance under multipath propagation conditions. His research interests are in the fields of wireless sensor networks and wireless communications systems.