WiLe: Fault Tolerance Mesh Network based on Bluetooth Low Energy for Reliable Wireless Sensor Networks

Hai Ninh Dang
Faculty of Engineering Mechanics and Automation
University of Engineering and Technology
Vietnam National University, Hanoi
Hanoi, Vietnam
mr.ninh.vn@outlook.com

Ngoc Linh Nguyen*
International School
Vietnam National University, Hanoi
Hanoi, Vietnam
nlnguyen@vnu.edu.vn
*Corresponding author

Abstract—Bluetooth Low Energy (BLE) has become a major wireless technology for the Internet of Things. Recent efforts by academia, industry, and standards development organizations have focused on creating BLE mesh network solutions. However, due to the technical complexity of Bluetooth and the fact that the majority of Bluetooth stacks are proprietary, the practical implementation of a BLE-based multi-hop network is very limited, mostly being tested on simulation. In this paper, a new mesh network stack based on the BLE Link Layer is proposed to create a reliable and self-correcting network. The network is then simulated using Matlab and deployed on real devices, and ready for mass commercial device.

Keywords—Mesh networks, wireless sensor networks, fault tolerance, BLE, IPv6.

I. INTRODUCTION

Bluetooth Low Energy (BLE) is a wireless communications technology created by the Bluetooth Special Interest Group (SIG), and introduced as part of the Bluetooth 4.0 core specification. In contrast with previous Bluetooth versions, BLE is mainly intended for applications involving simple devices (e.g., battery-enabled sensors) that need to infrequently transfer small units of data with remarkably low power consumption. Thanks to these characteristics and fueled by its common support in consumer electronics devices (e.g., smartphones), BLE has become a major communications technology for the Internet of Things (IoT).

Originally, BLE only supported the star network topology. However, many competing technologies in the IoT space support the mesh network topology as well, which provides advantages in terms of robustness and covered area.

In early efforts, Bluetooth SIG introduced a basic mesh protocol called Bluetooth mesh [1]. This protocol proves to be very suitable when controlling a large number of IoT devices, especially lighting devices. However, due to its simplicity, the standard Bluetooth Mesh protocol suffers from many reliability problems, including latency, message loss, and so on.

In this paper, and to the best of our knowledge, we provide a protocol stack, called WiLe, for end-to-end communication in a BLE mesh network topology. Then evaluate the performance of the new protocol against the standard Bluetooth mesh.

General Characteristics of WiLe protocol:
- BLE-base: Build on top of BLE Link Layer. Allows connection to standard Bluetooth devices.
- No single point of failure: Networks are auto-configuring and self-healing, so will continue to provide secure and reliable communication even if individual devices fail.
- Low power: Host devices can operate for several years on small batteries using suitable duty cycles.

II. WiLe PROTOCOL SPECIFICATION

A. WiLe Protocol Stack

Fig. 1 illustrates the protocol stack for WiLe protocol over Bluetooth LE links. In our proposal stack, the standard stack of BLE [2] is maintained, with the addition of the WiLe-Controller module, which is responsible for managing mesh connections and routing, along with an IPv6 module that communicates with the network stack layer.

The functions of the WiLe-Controller will be presented in chapter C. Routing and Network Connectivity.

The IPv6 module is responsible for assigning IP addresses using Dynamic Host Configuration Protocol (DHCP), as well as processing messages between the two network layers, includes header compression as defined in RFC6282 [3], which specifies the compression format for IPv6 datagrams on top of IEEE 802.15.4, to ensure message size when transmitted over BLE link.
WiLe: Fault Tolerance Mesh Network based on Bluetooth Low Energy for Reliable Wireless Sensor Networks

![Fig. 1. Protocol stack for WiLe network](image)

WiLe interferes with the following layers on the standard BLE stack:

- **Generic Access Profile (GAP):** Allows BLE devices to interoperate with each other. It provides a set of rules or procedures so that devices can discover each other, broadcast data, establish secure connections, perform functional operations and set device configurations.
- **Generic Attribute Profile (GATT):** Defines the structure in which data is exchanged between two devices.
- **Attribute Protocol (ATT):** Defines how data is represented in a BLE server database and the methods by which that data can be read or written.

**B. WiLe Network Topology**

In a WiLe network, nodes are split into two forwarding roles: A **Router** is a node that forwards packets for network devices and keeps its transceiver enabled at all times; an **End Device (ED)** is a node that communicates primarily with a single Router, does not forward packets for other network devices and can disable its transceiver to reduce power.

![Fig. 2. Basic WiLe Network Topology and Devices](image)

Besides that, the **Controller** is a Router that is responsible for managing the set of routers in a WiLe network. It is dynamically self-elected for fault tolerance, and aggregates and distributes network-wide configuration information.

A **Border Router** is a device that can forward information between a WiLe network and internet network (for example, WiFi).

The **Controller** is a Router that is responsible for managing the set of Routers in network. It is dynamically self-elected for fault tolerance.

**C. Routing and Network Connectivity**

BLE devices can use privacy addresses, which are updated frequently, in order to counter threats such as activities correlation over time, location tracking, or exploiting vendor-specific vulnerabilities. However, this approach may have negative impact on a routing mechanism. A possible approach to avoid this issue might be a coordinated, network-wide scheme whereby nodes share information that allows the address in use for each node at a given moment to be determined.

Until now, the only implementation that meets this requirement is the RPL protocol. However, RPL meta data is based on a tree-like structure, so it is not really suitable for a mesh multi-connection network structure. For the experiments presented in this work, we implemented a connection manager called WiLe-Controller, which enables static connection management.

At start, a node starts advertising its presence, or starts scanning for advertisements sent by the configured peer and initiates a connection. The WiLe-Controller module monitors the health of each configured connection. If a connection is dropped, the module goes back into advertising/scanning mode to reopen the lost connection. At the same time, the WiLe-Controller also performs the coordinator function based on distance-vector routing [4]. That works by having each router maintain a routing table, giving the best-known distance from source to destination and which route is used to get there.

Routing databases contain the data sets described in the following subsections:

- **Router ID Set:** The current set of valid Router IDs and the sequence number assigned to the current ID_set.
- **Link Set:** Records information about other Routers that are, or recently were, neighbors on that interface.

Each network will have a single WiLe-Controller, which is also a router. Routers acting as Controller maintain an additional database for tracking Router ID assignments. Like the Routing Database, a Controller maintains a separate Controller Database for each interface on which a Device is acting as a Controller. Any other router in the network can become a WiLe-Controller in case the old controller fails.

**D. IPv6 Addressing Architecture**

All WiLe devices use many different IPv6 unicast addresses for communication. For addresses that have scope larger than link-local, WiLe defines three different types of address:

- **Routing Locator (RLOC):** An IPv6 address that identifies the location of a device within a network partition.
- **Anycast Locator (ALOC):** An IPv6 address that identifies the location of one or more device within a network partition.
- **Endpoint Identifier (EID):** An IPv6 address that uniquely identifies a WiLe device.

WiLe also uses three scopes for unicast addressing:

- **Link-local:** For discover neighbors, configure links, and exchange routing information.
- **Mesh-local:** For reach device within the same network partition.
- Global (including Domain): for enable communication with devices outside the WiLe network.

All devices are assigned a Router ID and a End ID. Each router maintains a table of all their end node, the combination of which uniquely identifies a device within the topology.

E. Fault tolerance method

In a WiLe network, none of these devices represents a single point of failure. While there are a number of devices in the system that perform special functions, the WiLe design is such that they can be replaced without impacting the ongoing communication within the network.

Router selection is done in a distributed fashion: each router and end device choose whether to change state based on its information about the local network topology. In case a node tries to connect to an end device, that device sends a request to the WiLe controller to become a router. WiLe controller decides to upgrade end device to router based on the number of routers currently in the network, so the network stays connected and routes are not stretched out to more hops than necessary.

The process for an end device to become a router is as follows:

1. End device sends an address solicit message to the network Controller, asking for a Router ID. If the Controller accepts, it responds with a Router ID and the node upgrades itself to a Router.

2. The new Router sends a multicast Link Request to neighboring Routers. If received, the router responds with Link Accept messages.

3. The new Router responds to each Router with a unicast Link Accept to establish the Router-Router link.

When a Router downgrades to a End Device, its Router-Router links are disconnected. The device runs the network setup process again to re-establish the connection to the nearest router.

If the controller of a WiLe network becomes inoperable, one or more of the remaining routers will determine that they are no longer connected to the controller and form new network partitions. Network partitions advertise their presence and merge with other partitions if they are reachable.

III. SIMULATION AND EXPERIMENT RESULTS

In this section, we present how we deployed the network and measured its performance in the real world.

A. Simulation results

In order to verify the connectivity of the network with the number of devices and device distribution in various conditions. We designed a Matlab model that simulates the connection process of the entire network.

We also simulate network connectivity with a large number of different spatially distributed capabilities. With 1000 allocated to each set of nodes 94, 275 and 844, the simulated network establishes a stable connection for all nodes.
WiLe: Fault Tolerance Mesh Network based on Bluetooth Low Energy for Reliable Wireless Sensor Networks

The simulations perform random allocation of devices in spaces of 50x50 m and 150x150 m, showing the connection is stable established.

Fig. 8. Simulation network with 275 nodes in an area of 50x50m

Fig. 9. Simulation network with 844 nodes in an area of 150x150 m

B. Experiment results

In the next step, we deploy the network on the hardware device. We deployed 4 devices, of which 2 devices, NRF52840 and NRF5340, use BLE connection, while the remaining 2 devices, ESP32-C6 and ESP32-C3 connections use BLE and Wifi connection simultaneously.

The ESP32-C6 board are equipped with ESP32-C6 SoCs featuring a high-performance 32-bit RISC-V processor, which can be clocked up to 160 MHz, along with 512KB SRAM. Plays the role of WiLe Controller. At the same time, with integrated Wifi 6 connection, the ESP-C6 node also acts as a border router, a bridge between the WiLe network and the Internet.

ESP32-C3 board integrates a 32-bit core RISC-V microcontroller and 400 KB of internal RAM, it also has Wifi and BLE connectivity. Acts as a router and has the ability to become a replacement controller for the ESP32-C6 node.

NRF5340 and NRF52840 are two power-saving SoCs from Nordic. With BLE connectivity, they act as BLE router and end device to measure energy consumption with battery-powered devices.

Deployment on a variety of hardware to test the flexibility and suitability of the WiLe protocol on different hardware in practice.

We also implemented a web interface running on the Raspberry Pi to visually simulate the network's connectivity. At the same time, we simulate 9 other devices operating simultaneously with 4 hardware devices. The purpose of using a combination of real devices and simulated devices is to easily simulate error cases that can occur at random nodes during controlled periods of time.

Fig. 10. Four WiLe devices connect to Raspberry Pi board to record logs

Fig. 11. The web interface visually demonstrates the connections of 13 devices

We tested 4 days of activity logging with 64 times the normal message volume without any disconnection issues. In case the controller device emulator disconnects, the neighboring device will likely take on the role of the new network controller.

Fig. 12. The new controller is replaced (hexagon with red border) when the previous controller is disconnected
The experimental implementation also passed the 4 tests shown in Table I, includes no lost messages when routing, fault tolerance in cases of node allocation changes, connection disconnections on 1 or 2 router nodes.

<table>
<thead>
<tr>
<th>Table I. WiLe implementations test cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>√ = pass, x = fail</td>
</tr>
<tr>
<td>0% dropped messages</td>
</tr>
<tr>
<td>√</td>
</tr>
</tbody>
</table>

C. Energy Efficiency

To better understand whether battery-driven WiLe devices, we measure the current consumption for a single node in different connection states and scenarios for a Nordic SoC board using the Current Ranger Kit [5], along with an oscilloscope. Allows us to see not only average current consumption but also instantaneous consumption with nA resolution.

![Fig. 13. Power measurement setup](image)

Equation (1) indicates how the WiLe device lifetime (L).

\[ L = \frac{E_{\text{Battery}}}{I_{\text{avg}}} \]  \hspace{1cm} Eq. 1

In our medium traffic load scenario (producer interval of 2s±0.5 s, connection interval 75 ms), a subordinate that acts as forwarder with three active connections shows an additional current consumption of 100 μA caused by the BLE connections. And current consumption to the board’s average idle current consumption of 12 μA. We assume an \( E_{\text{Battery}} \) of 235 mAh, which is typical for a button cell battery, allow to run this configuration for 2 years.

This shows that despite a considerable overhead in software complexity and CPU processing, WiLe nodes can compete with plain BLE mesh in terms of energy while at the same time offering more network features and reliability.

IV. COMMERCIALIZATION

Our proposed WiLe protocol is currently being commercialized in the pre-release phase for the purpose of testing stability in practice.

With about 10,000 devices being deployed, including smarthome devices branded FPT, Casper, Rossi, with device types including: Power switch, power socket, universal infrared remote, air conditioner, water heater. These devices have been installed in personal apartments as well as offices and hotels. Feedback from end users is being captured so that future improvements can be made.

![Fig. 14. Devices that are implementing the WiLe Protocol](image)

In addition, the commercial WiLe network is also expanded based on the ability to connect with Bluetooth Mesh standard devices, including smart lighting and sensor devices under the Rang Dong brand. Demonstrates good compatibility by taking advantage of the advantages of BLE connection.

V. CONCLUSION

In this study, we have implemented a new fully functional network protocol based on BLE connection, includes proposed software stack, routing protocol, fault tolerance method, and perform deployment using simulation as well as real devices. The results show that the network operates to meet the proposed requirements.

Compared with existing implementations and research, our WiLe network structure has many advantages thanks to the implementation of a mesh structure. Details are compared in Table II.

<table>
<thead>
<tr>
<th>Table II. Compare some IP over BLE implementations</th>
</tr>
</thead>
<tbody>
<tr>
<td>√ = supported</td>
</tr>
<tr>
<td>Implementation</td>
</tr>
<tr>
<td>WiLe</td>
</tr>
<tr>
<td>NimBLE [6]</td>
</tr>
<tr>
<td>Zephyr [7]</td>
</tr>
<tr>
<td>BLEach [8][9][10]</td>
</tr>
</tbody>
</table>

WiLe protocol implementation has been deployed on commercial devices and demonstrated effective operation on devices using power walls as well as rechargeable batteries. The next development direction of the research is to implement a WiLe network architecture for sensor devices powered by non-rechargeable batteries, with the goal of minimum energy consumption to operate for 5-10 years.

BLE is considered a promising deployment option for low-power IoT scenarios as it offers popular network access available in the mass market and WiLe has demonstrated full potential in future mesh scenarios. In the future, we plan to expand the scope to include mobile systems such as a WiLe device. From there, the overall solution is completed and ready for commercial implementation.
WiLe: Fault Tolerance Mesh Network based on Bluetooth Low Energy for Reliable Wireless Sensor Networks

REFERENCES