

# THIẾT KẾ ỨNG DỤNG CHẠY ĐA GIAO THỨC ĐỘNG CHO HỆ THỐNG IOT

LUẬN VĂN KỸ SƯ

Phan Quang Thông – 1810558

Nguyễn Thị Mỹ Thu - 1814215

Giảng viên hướng dẫn

TS. Võ Quê Sơn



ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC BÁCH KHOA  
KHOA ĐIỆN – ĐIỆN TỬ, BỘ MÔN VIỄN THÔNG

05 – 2022



Số: \_\_\_\_\_ /BKĐT

Khoa: Điện – Điện tử

Bộ Môn: Viễn Thông

## NHIỆM VỤ LUẬN VĂN TỐT NGHIỆP

1. Họ và tên: Phan Quang Thông MSSV: 1810558  
Nguyễn Thị Mỹ Thu MSSV: 1814215
2. Ngành: Điện – Điện tử Chuyên ngành: Kỹ thuật Điện tử - Truyền thông
3. Đề tài: Thiết kế ứng dụng chạy đa giao thức động cho hệ thống IoT
4. Nhiệm vụ:
  - Thiết kế phần mềm ứng dụng dựa trên kỹ thuật dynamic multiprotocol (đa giao thức mạng)
  - Thiết kế hệ thống mạng chạy đa giao thức: Thread, Zigbee, Bluetooth Low Energy (BLE)
  - Mô phỏng hệ thống trong môi trường có điều kiện lý tưởng và thực tiễn
  - Đo đặc thực nghiệm các thông số đánh giá hệ thống
  - Phân tích và so sánh kết quả để rút ra kết luận
5. Ngày giao nhiệm vụ luận văn: 20/01/2022
6. Ngày hoàn thành nhiệm vụ: 31/05/2022
7. Họ và tên người hướng dẫn: Phản hướng dẫn  
TS. Võ Quê Sơn,  
BM Viễn Thông, Khoa Điện – Điện Tử 100%  
Nội dung và yêu cầu LVTN đã được thông qua Bộ Môn.

TP.HCM, ngày 31 tháng 05 năm 2022

**CHỦ NHIỆM BỘ MÔN**

PGS. TS. Hà Hoàng Kha

**NGƯỜI HƯỚNG DẪN CHÍNH**

TS. Võ Quê Sơn

**PHẦN DÀNH CHO KHOA, BỘ MÔN:**

Người duyệt (chấm sơ bộ): .....

Đơn vị: .....

Ngày bảo vệ : .....

Điểm tổng kết:.....

Nơi lưu trữ luận văn:.....

## LỜI CẢM ƠN

Lời đầu tiên, xin chân thành cảm ơn đến Khoa Điện - Điện Tử trường Đại Học Bách Khoa - ĐẠI HỌC QUỐC GIA TP.HCM đã tạo điều kiện cho nhóm em nghiên cứu và thực hiện luận văn tốt nghiệp này.

Nhóm em xin gửi cảm ơn đến thầy Võ Quê Sơn, người đã tận tình hướng dẫn và định hướng trong quá trình thực hiện luận văn của nhóm. Những lời nhận xét, hướng dẫn của thầy đã cung cấp cho nhóm rất nhiều kiến thức, kinh nghiệm trong quá trình thực hiện đề tài, từ đó cải thiện đáng kể những sai sót.

Nhóm cũng xin chân thành cảm ơn tất cả quý Thầy Cô đã nhiệt tình giảng dạy cho em trong suốt quá trình học tập tại trường. Cuối cùng, nhóm em xin được gửi lời cảm ơn chân thành đến gia đình cũng như bạn bè, người thân đã luôn ủng hộ, động viên em để em có thể có điều kiện tốt nhất cà về vật chất và tinh thần để học tập và nghiên cứu đến giây phút này.

Luận văn tốt nghiệp được thực hiện với sự phối hợp của hai thành viên, do vốn kiến thức còn hạn chế nên không thể tránh khỏi những thiếu sót. Rất mong nhận được sự đóng góp ý kiến từ phía thầy cô để chúng tôi học thêm được nhiều kinh nghiệm quý báu và có thể áp dụng vào công việc sau này.

Xin chân thành cảm ơn!

TP. HCM, ngày 01, tháng 06 năm 2022

Phan Quang Thông, Nguyễn Thị Mỹ Thu

## LỜI CAM ĐOAN

Thành viên của nhóm: Phan Quang Thông và Nguyễn Thị Mỹ Thu, là sinh viên chuyên ngành Kỹ thuật Điện tử - Viễn thông khóa 2018, tại Đại học Quốc gia thành phố Hồ Chí Minh – Trường Đại học Bách Khoa.

Nhóm xin cam đoan những nội dung sau đều là sự thật:

- (i) Công trình nghiên cứu này hoàn toàn do chính tôi thực hiện;
- (ii) Các tài liệu và trích dẫn trong luận văn này được tham khảo từ các nguồn thực tế, có uy tín và độ chính xác cao;
- (iii) Các số liệu và kết quả của công trình này được tôi tự thực hiện một cách độc lập và trung thực

Ngoài ra, trong đề tài còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc rõ ràng và cụ thể.

TP. HCM, ngày 01, tháng 06 năm 2022

Phan Quang Thông, Nguyễn Thị Mỹ Thu

## TÓM TẮT LUẬN VĂN

Trong luận văn này, nhóm em thực hiện thiết kế và xây dựng mô hình mạng đa giao thức trong các ứng dụng. Từ đó đánh giá được hiệu năng sử dụng của mạng đa giao thức trong môi trường hoạt động thực tiễn. Nhóm tập trung vào 2 nhiệm vụ chính:

- Thiết kế và xây dựng mô hình mạng đa giao thức
- Đánh giá hiệu suất hoạt động của giao thức trong mạng

*Đối với thiết kế và xây dựng hệ thống*, nhóm xây dựng mô hình gồm năm thành phần chính.

Khối Gateway	Giao tiếp với khối mạng OpenThread và khối mạng Zigbee qua module dongle và giao tiếp với khối Broker qua mạng Wifi và giao thức MQTT
Khối node Router	Sử dụng công nghệ Thread/Zigbee để truyền dữ liệu (hoặc nhận tín hiệu điều khiển) từ khói Gateway. Giao thức Thread/Zigbee chạy đồng thời với giao thức BLE để phục vụ việc truyền nhận dữ liệu khi Broker gặp sự cố
Khối End node	Chỉ sử dụng công nghệ Thread/Zigbee để truyền dữ liệu đồng thời tích hợp các giải thuật tiết kiệm năng lượng.
Khối Broker	Hoạt động trên Cloud của HiveMQ được hỗ trợ sẵn. Nhiệm vụ của khói Broker là truyền tải thông tin giữa khói Gateway và khói điều khiển thông qua công nghệ Wifi để có thể điều khiển từ xa.
Khối điều khiển	Phần mềm chạy trên máy tính/smartphone để nhận và gửi gói tin đến Broker để người dùng có thể cập nhật cũng như gửi lệnh điều khiển đến khói mạng Thread và Zigbee

*Đối với việc đánh giá hiệu suất hoạt động của mô hình mạng*, nhóm tiến hành thực nghiệm ở 2 khu vực khác nhau (trong nhà - nhiều vật cản, ngoài trời - ít vật cản) để có thể nhận xét về sự ảnh hưởng đáng kể của môi trường đến khả năng truyền nhận của mạng.

## ABSTRACT

In this thesis, we designed and developed applications based on multiprotocol technique. By performing experiments in different environments, conditions and hardware, we evaluated the efficiency of dynamic multiprotocol technique in practical settings. The thesis focused on two main points:

- Designing and building a network in which has dynamic multiprotocol nodes
- Evaluating the effectiveness of the protocols used in this network

The implementation of our system was divided into five main components:

Gateway	Exchanging packets with OpenThread network and Zigbee network through Thread and Zigbee dongles; communicating with the Broker using MQTT protocol through WiFi.
Routers	Running on Thread or Zigbee; exchanging data with Gateway. Some routers have dynamic multiprotocol (BLE/Thread or BLE/Zigbee) firmware to enable users' smartphones to access the network when Broker crashes.
End node	Running only on Thread or Zigbee for power saving purposes; receiving control commands from users
Broker	Running on the available Cloud service of HiveMQ. The main objective of the Broker is to act as a bridge between the Gateway and the control block (i.e. mobile application, website) by WiFi for remote control.
Control block	A mobile application on users' smartphone or laptop that allows them to access the Thread/Zigbee network by sending and receiving packets to and from the Broker (remote control) or a dynamic multiprotocol BLE/Thread node (local control).

To intensively evaluate the performance of this network, we performed our testing experiments in two different environments: indoor - almost no obstacles and outdoor - few obstacles, from which we come to an objective conclusion about how environmental factors affect the Thread or Zigbee connection.



## MỤC LỤC

<b>LỜI CÁM ƠN .....</b>	<b>i</b>
<b>LỜI CAM ĐOAN .....</b>	<b>ii</b>
<b>TÓM TẮT LUẬN VĂN .....</b>	<b>iii</b>
<b>ABSTRACT .....</b>	<b>iv</b>
<b>DANH SÁCH BẢNG .....</b>	<b>ix</b>
<b>DANH SÁCH HÌNH VẼ .....</b>	<b>x</b>
<b>DANH SÁCH TỪ VIẾT TẮT .....</b>	<b>xii</b>
<b>CHƯƠNG 1. GIỚI THIỆU .....</b>	<b>2</b>
1.1    Đặt vấn đề .....	2
1.2    Phạm vi và phương pháp nghiên cứu .....	2
1.3    Các đóng góp của luận văn .....	2
<b>CHƯƠNG 2. CƠ SỞ LÝ THUYẾT .....</b>	<b>3</b>
2.1    Tiêu chuẩn IEEE 802.15.4 .....	3
2.1.1    Lớp vật lý (PHY) .....	4
2.1.2    Lớp giao thức MAC .....	5
2.1.3    Lớp mạng .....	5
2.1.4    Lớp vận chuyển .....	7
2.1.5    Lớp ứng dụng .....	7
2.2    OpenThread .....	8
2.2.1    Tổng quát về công nghệ Thread .....	8
2.2.2    Kiến trúc mạng .....	9
2.2.3    Cơ chế đánh địa chỉ IPv6 .....	14
2.2.4    Tạo một mạng mới .....	20
2.2.5    Tham gia một mạng có sẵn .....	20
2.2.6    Lựa chọn Router .....	24
2.3    Giao thức Zigbee .....	29
2.3.1    Tổng quát về công nghệ Zigbee .....	29
2.3.2    Zigbee Mesh Networking .....	30
2.3.3    Phân loại thiết bị và vai trò .....	31
2.3.4    Định danh trong mạng .....	32
2.3.5    Phương pháp định tuyến .....	33
2.3.6    Bảo mật .....	37
2.3.7    Tạo lập mạng Zigbee .....	38
2.3.8    Thư viện Zigbee Cluster .....	40

<b>2.4 Giao thức MQTT .....</b>	<b>41</b>
2.4.1 Tổng quan về giao thức MQTT .....	41
2.4.2 Các node trong mô hình mạng MQTT .....	42
2.4.3 Cơ chế hoạt động .....	42
2.4.4 MQTT-SN.....	45
<b>2.5 Giao thức CoAP .....</b>	<b>47</b>
<b>2.6 Giao thức Bluetooth Low Energy .....</b>	<b>49</b>
2.6.1 Khái niệm.....	49
2.6.2 Đặc điểm .....	50
2.6.3 Ưu điểm và hạn chế của BLE .....	51
2.6.4 Kiến trúc của BLE .....	52
<b>2.7 Kỹ thuật đa giao thức động.....</b>	<b>61</b>
<b>CHƯƠNG 3. ĐẶC TẢ HỆ THỐNG.....</b>	<b>64</b>
<b>3.1 Khối Router .....</b>	<b>64</b>
<b>3.2 Khối End node.....</b>	<b>65</b>
<b>3.3 Khối Gateway .....</b>	<b>65</b>
<b>3.4 Khối Broker.....</b>	<b>66</b>
<b>3.5 Khối Điều khiển.....</b>	<b>66</b>
<b>CHƯƠNG 4. THIẾT KẾ VÀ THỰC HIỆN .....</b>	<b>67</b>
<b>4.1 Mô tả phần cứng .....</b>	<b>67</b>
4.1.1 nRF52840 Development Kit.....	67
4.1.2 CC2538 Customized Kit.....	68
4.1.3 Silabs EFR32MG12 + Wireless Starter Kit .....	69
4.1.4 Raspberry PI 3.....	70
<b>4.2 Thực hiện Firmware .....</b>	<b>71</b>
4.2.1 Khối gateway .....	71
4.2.2 Khối Router.....	74
4.2.3 Khối End Node .....	79
4.2.4 Khối Broker .....	81
4.2.5 Khối điều khiển.....	82
<b>CHƯƠNG 5. ĐÁNH GIÁ KẾT QUẢ.....</b>	<b>86</b>
<b>5.1 Đánh giá độ xa của đường truyền .....</b>	<b>86</b>
5.1.1 Kịch bản thử nghiệm.....	86
5.1.2 Môi trường thử nghiệm.....	86
5.1.3 Cách thức thực hiện .....	86
5.1.4 Kết quả .....	86
5.1.5 Nhận xét.....	87
<b>5.2 Xác định khoảng cách truyền tối ưu (PRR &gt; 95%) .....</b>	<b>87</b>
5.2.1 Kịch bản thử nghiệm.....	87
5.2.2 Môi trường thử nghiệm.....	88
5.2.3 Cách thức thực hiện .....	88
5.2.4 Kết quả .....	90
5.2.5 Nhận xét.....	91

<b>5.3 Đánh giá truyền tin đa chặng.....</b>	<b>91</b>
5.3.1 Kịch bản thử nghiệm 1 .....	91
5.3.2 Kịch bản thử nghiệm 2 .....	95
<b>5.4 Điều khiển qua MQTT .....</b>	<b>97</b>
5.4.1 Kịch bản thử nghiệm:.....	97
5.4.2 Cách thức thực hiện .....	98
5.4.3 Kết quả thu được .....	98
5.4.4 Nhận xét .....	98
<b>5.5 Điều khiển nội bộ .....</b>	<b>99</b>
5.5.1 Kịch bản thử nghiệm.....	99
5.5.2 Môi trường thực hiện .....	99
5.5.3 Cách thức thực hiện .....	99
5.5.4 Kết quả thu được .....	99
5.5.5 Nhận xét .....	99
<b>5.6 Đo công suất tiêu thụ và dự đoán thời gian hoạt động.....</b>	<b>100</b>
5.6.1 Kịch bản thử nghiệm.....	100
5.6.2 Cách thức thực hiện .....	100
5.6.3 Kết quả thu được .....	101
5.6.4 Nhận xét .....	101
<b>5.7 Đánh giá độ hiệu quả của ứng dụng đa giao thức.....</b>	<b>101</b>
5.7.1 Kịch bản thử nghiệm.....	101
5.7.2 Môi trường thử nghiệm .....	102
5.7.3 Cách bước thực hiện .....	102
5.7.4 Kết quả thực hiện .....	103
5.7.5 Nhận xét .....	103
<b>CHƯƠNG 6. KẾT LUẬN VÀ PHƯƠNG HƯỚNG.....</b>	<b>104</b>
<b>6.1 Kết luận.....</b>	<b>104</b>
<b>6.2 Phương hướng phát triển .....</b>	<b>104</b>
<b>TÀI LIỆU THAM KHẢO .....</b>	<b>105</b>

## DANH SÁCH BẢNG

Bảng 1. Thông số thiết kế của mạng Thread được nhà sản xuất cung cấp .....	9
Bảng 2. Giới hạn số thiết bị trong mạng .....	13
Bảng 3. Các loại địa chỉ Unicast .....	17
Bảng 4. Phân loại địa chỉ Multicast .....	19
Bảng 5. Nội dung Parent Request .....	21
Bảng 6. Nội dung Parent Response .....	22
Bảng 7. Nội dung Child ID Request .....	23
Bảng 8. Nội dung Child ID Response .....	24
Bảng 9. Thông tin Link Request .....	27
Bảng 10. Nội dung Link Accept .....	28
Bảng 11. Mô hình client-server giữa bóng đèn và công tắc .....	40
Bảng 12. Ví dụ một số lệnh toàn cầu .....	41
Bảng 13. So sánh CoAP và MQTT .....	49
Bảng 14. Mô tả các state machine của Bluetooth Low Energy .....	57
Bảng 15. Kết quả đánh giá độ xa của đường truyền .....	86
Bảng 16. Kết quả xác định khoảng cách truyền tối ưu .....	90
Bảng 17. Kết quả đánh giá truyền tin đa chặng ở môi trường không có vật cản .....	94
Bảng 18. Kết quả đánh giá truyền tin đa chặng ở môi trường có vật cản .....	95
Bảng 19. Kết quả đánh giá truyền tin đa chặng 2 .....	97
Bảng 20. Kết quả thử nghiệm điều khiển qua MQTT .....	98
Bảng 21. Kết quả thử nghiệm nội bộ .....	99
Bảng 22. Thông số cơ bản cung cấp bởi Nordic .....	100
Bảng 23. Kết quả thử nghiệm dự đoán thời gian hoạt động .....	101
Bảng 24. Kết quả thu được của thử nghiệm đánh giá ứng dụng đa giao thức .....	103

## DANH SÁCH HÌNH VẼ

Hình 1. Mô hình phân lớp các giao thức trong chuẩn IEEE 802.15.4 [1] .....	3
Hình 2. Khung dữ liệu lớp PHY và MAC [3].....	5
Hình 3. Vai trò Node trong mạng .....	9
Hình 4. Phân loại theo thiết bị .....	10
Hình 5. Quá trình thăng cấp.....	11
<i>Hình 6. Quá trình hạ cấp .....</i>	<i>12</i>
Hình 7. Vị trí của Thread leader và Border Router.....	12
Hình 8. Minh họa hiện tượng phân vùng mạng .....	13
Hình 9. Công thức Routing Locator .....	14
Hình 10. RLOC16.....	15
Hình 11. Ví dụ minh họa RLOC16.....	15
Hình 12. RLOC hoàn chỉnh .....	16
Hình 13. RLOC thay đổi khi Topology thay đổi .....	16
Hình 14. Parent Request .....	21
Hình 15. Parent Response .....	22
Hình 16. Child ID Request.....	23
Hình 17. Child ID Response .....	24
Hình 18. Minh họa Connected Dominating Set.....	25
Hình 19. Link Request .....	27
Hình 20. Link Accept.....	28
Hình 21. Link Accept và Request .....	28
Hình 22. Mô hình phân lớp của Zigbee .....	29
Hình 23. Ba loại topology trong mạng Zigbee .....	30
Hình 24. Centralized security và Distributed security .....	32
Hình 25. Quá trình khám phá đường đi 1 .....	35
Hình 26. Quá trình khám phá đường đi 2 .....	35
Hình 27. Quá trình sửa chữa đường đi 1 .....	36
Hình 28. Quá trình sửa chữa đường đi 2 .....	37
Hình 29. Mô hình kết nối MQTT.....	43
Hình 30. Minh họa sự khác biệt ở các mức QoS khác nhau .....	44
Hình 31. Quá trình client đăng ký và hủy đăng ký .....	45
Hình 32. Các gói tin trao đổi để duy trì, kết thúc kết nối MQTT .....	45
Hình 33. Mô hình mạng của giao thức MQTT-SN.....	47
Hình 34. Môi trường hoạt động của giao thức CoAP .....	48
Hình 35. Phân loại thiết bị Bluetooth.....	50
Hình 36. Phổ tần số.....	51
Hình 37. Kiến trúc BLE .....	53
Hình 38. Mô tả Frequency Hopping .....	55
Hình 39. Mô tả Channel Hopping.....	56
Hình 40. Bluetooth state machine .....	56
Hình 41. Mô hình kết nối giữa các thành phần trong lớp L2CAP .....	58
Hình 42. Cấu trúc GATT .....	60

Hình 43. Lược đồ thời gian của bộ vô tuyến chạy Dynamic Multiprotocol BLE và Thread .....	61
Hình 44. Kiến trúc đa giao thức động của Nordic .....	62
Hình 45. Độ ưu tiên gói tin khi chạy đa giao thức động.....	63
Hình 46. Topology hệ thống .....	64
Hình 47. nRF52840 Development Kit.....	67
Hình 48. CC2538 Customized Kit .....	68
Hình 49. EFR32MG12.....	69
Hình 50. Raspberry PI.....	71
Hình 51. Phần cứng Gateway .....	72
Hình 52. Mô hình phần mềm của Gateway .....	73
Hình 53. Thông tin mạng hiển thị trên Linux-host .....	74
Hình 54. Giải thuật nhận/gửi gói tin MQTT trên Gateway .....	75
Hình 55. BLE service điều khiển LED .....	76
Hình 56. Nguyên lý hoạt động của BLE/Thread Forwarding Service.....	77
Hình 57. Cấu trúc BLE/Thread Forwarding .....	78
Hình 58. Giải thuật gửi gói tin BLE/Thread Forwarding Service .....	79
Hình 59. Giải thuật xử lý gói tin BLE/Thread Forwarding Service .....	79
Hình 60. Stack của Thread End Node.....	80
Hình 61. Stack của Zigbee End Node.....	81
Hình 62. Kết quả thiết kế Gateway thành công .....	82
Hình 63. Giải thuật cho app điện thoại .....	83
Hình 64. Giao diện app cho phép người dùng quét và kết nối với nút Forwarder thông qua BLE .....	84
Hình 65. Giao diện app hiển thị kết quả cho quá trình discover và gửi các gói tin cho nút đích Thread .....	85
Hình 66. Mô tả môi trường thử nghiệm cho đánh giá đường truyền tối ưu .....	88
Hình 67. Kết quả thực hiện với giao thức OpenThread 1 .....	89
Hình 68. Kết quả thực hiện với giao thức Zigbee .....	89
Hình 69. Kết quả thực hiện với giao thức OpenThread 2 .....	90
Hình 70. Mô hình kết nối mạng đánh giá truyền tin đa chặng 1.....	92
Hình 71. Ảnh chụp thực tế quá trình thực hiện đo ngoài trời .....	93
Hình 72. Topology khi thực hiện đánh giá môi trường trong nhà .....	93
Hình 73. Kết quả thực hiện lệnh ping .....	94
Hình 74. Mô hình kết nối mạng đánh giá truyền tin đa chặng 2.....	96
Hình 75. MQTT script trên phần mềm MQTTLX .....	98
Hình 76. Mô tả quá trình đo dòng của thử nghiệm .....	101
Hình 77. Mô hình thử nghiệm đa giao thức .....	102
Hình 78. Kết quả thực hiện của thử nghiệm đánh giá ứng dụng đa giao thức.....	103

## DANH SÁCH TỪ VIẾT TẮT

6LoWPAN	IPv6 over Low power Wireless Personal Area Networks
ACK	Acknowledgement
ATT	Attribute Protocol
BLE	Bluetooth Low Energy
CoAP	Constrained Application Protocol
CLI	Command-line Interface
CSMA	Carrier Sense Multiple Access
DTLS	Datagram Transport Layer Security
ED	End device
EID	Endpoint Identifier
FTD	Full thread device
GAP	Generic Access Profile
GATT	Generic Attribute Profile
GTS	Guaranteed Time Slots
GUA	Global Unicast Address
HCI	Host Controller Interface
HTTP	Hypertext Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
MAC	Media access control
MLE	Mesh Link Establishment
MTD	Minimal thread device
NCP	Network Co-processor
L2CAP	Logical Link Control and Adaptation Protocol
LAN	Local area network
LR-WPAN	Low Rate – Wireless Personal Area Network
PRR	Packet Reception Rate
PAN	Personal Area Network
PHY	Physical layer
PPDU	Physical layer protocol data unit
PSDU	Physical layer service data unit
REED	Router eligible end device
RLOC	Routing Locator
RTT	Round-trip Time
SM	Security Manager
ULA	Unique Local Address
URI	Uniform Resource Identifier
ZCL	Zigbee Cluster Library



## CHƯƠNG 1. GIỚI THIỆU

### 1.1 Đặt vấn đề

Thread (OpenThread) là một trong những công nghệ truyền tin không dây mới ra mắt trong những năm gần đây. Thread được thiết kế đặc biệt cho các ứng dụng mạng công suất thấp, tập trung vào hiệu quả về vùng phủ sóng và năng lượng. Đặc biệt, Thread chạy trên nền IPv6, khác với các giao thức trước đây. Điểm hình cho các ứng dụng sử dụng công nghệ Thread là nhà thông minh, thành phố thông minh, trang trại thông minh, ... Đã có nhiều hãng sản xuất phần cứng tích hợp Thread trở thành một trong những giao thức chính bên cạnh các giao thức phổ biến như Bluetooth Low Energy (BLE) và Zigbee.

Trong luận văn tốt nghiệp này, nhóm chúng em sẽ tiến hành thiết kế và xây dựng mô hình mạng đa giao thức với mục tiêu:

- Đo đạc và kiểm tra độ hiệu quả của mạng Thread trong môi trường thực tế.
- Chạy đa giao thức giúp hỗ trợ công nghệ ra mắt từ lâu và tiết kiệm phần cứng.

### 1.2 Phạm vi và phương pháp nghiên cứu

- Nghiên cứu ba giao thức phổ biến trong ứng dụng IoT là Zigbee, Bluetooth và Thread
- Thiết kế các khối của hệ thống và hiện thực hóa trên các phần cứng khác nhau
- Thực hiện đo đạc thực tế để đánh giá hệ thống

### 1.3 Các đóng góp của luận văn

Luận văn này có các đóng góp như sau:

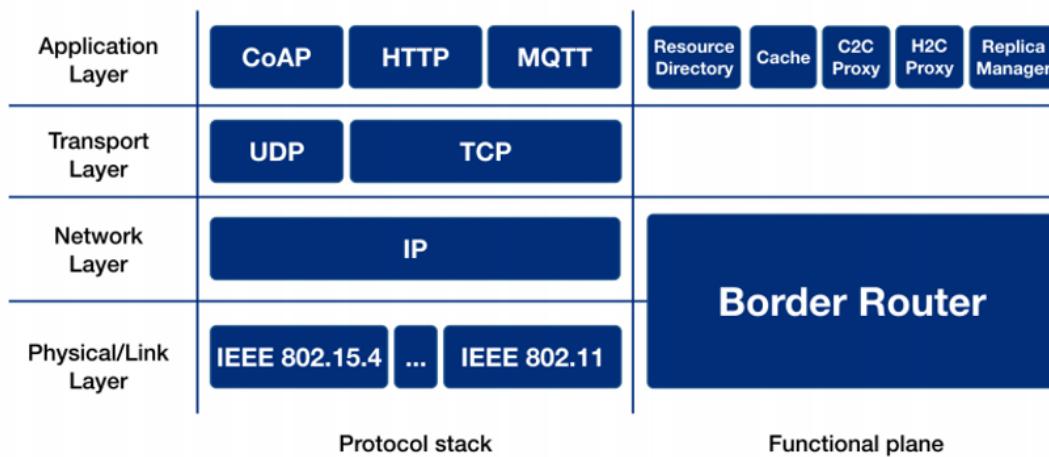
- Xây dựng Gateway chạy đa giao thức góp phần tích hợp các giao thức phổ biến vào một hệ thống IoT
- Xây dựng mô hình chạy đa giao thức và thực hiện đánh giá trong các điều kiện thực tế

## CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

## 2.1 Tiêu chuẩn IEEE 802.15.4

Để có cái nhìn rõ ràng về kiến trúc và phân lớp các giao thức của thiết bị IoT, nhóm đề xuất cách tiếp cận như hình bên dưới.

- Lớp ứng dụng (Application): CoAP, HTTP, MQTT
- Lớp vận chuyển (Transport): UDP, TCP
- Lớp mạng (Network): IPv6, 6LoWPAN
- Lớp vật lý (Physical/Link): Tiêu chuẩn IEEE 802.15.4, Tiêu chuẩn IEEE 802.11



Hình 1. Mô hình phân lớp các giao thức trong chuẩn IEEE 802.15.4 [1]

IEEE 802.15.4 là một tiêu chuẩn cung cấp quy định cho giao thức MAC và lớp vật lý (PHY) trong mô hình OSI một khuôn khổ và các lớp thấp hơn trong mô hình OSI, dùng cho các kết nối không dây chi phí nhỏ và công suất nhỏ. Hiện nay, IEEE 802.15.4 là nền tảng của Zigbee, 6LoWPAN, WirelessHART và Thread. Một số đặc điểm nổi bật của chuẩn truyền thông này [2].

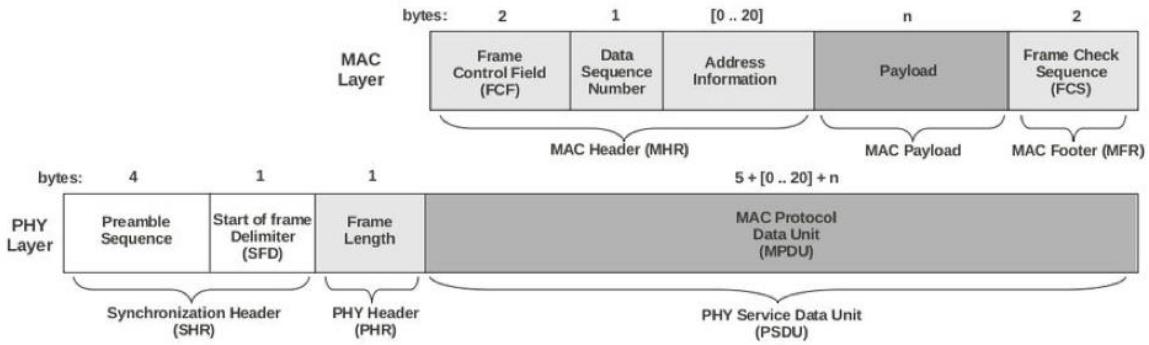
- Các phương pháp định tuyến đảm bảo năng lượng được bảo toàn và độ trễ truyền tin là ở mức thấp nhất có thể bằng cách dùng các khe thời gian bảo đảm (Guaranteed Time Slots)

- Tránh xung đột trong quá trình truyền tin nhờ vào cơ chế đa truy nhập tránh xung đột CSMA/CA (Carrier Sense Multiple Access Collision Avoidance)
- Hỗ trợ nhiều chuẩn bảo mật
- Hỗ trợ hàm quản lý năng lượng

### 2.1.1 Lớp vật lý (PHY)

Lớp vật lý (PHY) cung cấp lớp giao tiếp trung gian giữa lớp MAC và kênh truyền vật lý. Các lớp cao hơn sẽ kiểm soát phần cứng và phần mềm của bộ thu phát radio thông qua lớp vật lý. PHY Management Entity (PLME) là một giao diện cung cấp các chức năng quản lý quan trọng của lớp vật lý. PLME sẽ quản lý các tham số số, đối tượng và một số thông tin khác của lớp PHY. Một số tham số quan trọng bao gồm [2]:

- Tốc độ chip: 1600 kchip/s
- Lấy mẫu: O - QPSK, DSSS (Direct Sequence Spread Spectrum)
- Tốc độ bit: 250 kb/s
- Tốc độ mẫu: 62.5 ksymbol/s
- Số kênh truyền: 16
- Mỗi gói tin của lớp vật lý (PHY protocol data unit - PPDU) mang theo 4 phần chính:
  - Preamble (32 bits - đồng bộ ký tự)
  - Start of Packet delimiter (8 bits - đồng bộ khung)
  - PHY header/Độ dài gói tin (8 bits - thông báo độ dài trường PSDU)
  - PHY service data unit/PSDU (tối đa 127 bytes - trường data)



Hình 2. Khung dữ liệu lớp PHY và MAC [3]

Bên cạnh đó, lớp PHY hỗ trợ các hàm quản lý như Energy detection (ED), Link quality indicator (LQI), Radio signal strength indicator (RSSI) và Clear channel assessment (CCA).

### 2.1.2 Lớp giao thức MAC

Nhiệm vụ của giao thức MAC là xử lý các thông điệp quan trọng và kiểm soát tắc nghẽn. Cụ thể, nó lắng nghe các kênh truyền và các lớp liên kết để kiểm soát các lần thử lại, khung ACK (xác nhận) các thông báo để liên lạc tin cậy bằng các giao thức CSMA (Carrier Sense Multiple Access). Mã hóa và bảo vệ giao thức MAC được sử dụng trên các tin nhắn thông qua các khóa được thiết lập và cấu hình bởi các lớp mô hình OSI cao hơn. Theo đó, nó cung cấp cho lớp mạng sự giao tiếp cơ bản vững chắc [2].

Giao thức MAC có hai chế độ:

- Chế độ không báo hiệu (non-beacon mode): Truyền dữ liệu bằng thuật toán CSMA / CA không có vạch. Một số đặc điểm của chế độ này: Không sử dụng cấu trúc siêu khung (superframe), Không mở rộng, Tự tổ chức, Không đảm bảo nhận dữ liệu dữ liệu.
- Chế độ báo hiệu (beacon mode): Bộ điều phối PAN tạo báo hiệu thường xuyên để đồng bộ hóa các node được liên kết với nó. Siêu khung (superframe) được giới hạn giữa các thiết bị.

### 2.1.3 Lớp mạng

### 2.1.3.1 Giao thức IPv6

Giao thức Internet phiên bản 6 (IPv6) là phiên bản mới nhất của Giao thức Internet (IP). IP cung cấp một hệ thống nhận dạng và định vị cho tất cả các thiết bị (máy tính, router, bộ chuyển mạch, ...) trên Internet. IPv4 hiện được sử dụng trong mạng nội bộ và mạng cá nhân cũng như Internet. IPv6 là thừa kế của IPv4 nên hỗ trợ hầu hết các chức năng trên IPv4. Lợi ích đáng kể của IPv6 là mạng có thể đánh địa chỉ trực tiếp cho các thiết bị thông qua Border Router và Dịch vụ đám mây có thể đánh địa chỉ các thiết bị cục bộ trên mạng bằng phương pháp định địa chỉ IP tiêu chuẩn [4]. Địa chỉ IPv6 hỗ trợ nhiều hơn một địa chỉ cục bộ duy nhất (Unque local address - ULA) hoặc địa chỉ unicast toàn cầu (Global unicast address - GUA) dựa trên tài nguyên của chúng.

Các bit bậc cao (N bit) của địa chỉ IPv6 xác định mạng, trong khi phần còn lại chỉ định địa chỉ cụ thể trong mạng. N bit đó được gọi là 'prefix'. '/ 64' cho thấy N = 64 hoặc địa chỉ này có tiền tố 64 bit. Trong mạng Thread, mạng chọn một prefix sẽ được sử dụng trong toàn mạng [5]. Tiền tố đó được gọi là Locally assigned global ID hoặc mesh local ULA prefix. Các thiết bị trong mạng Thread sử dụng địa chỉ MAC mở rộng của riêng chúng để xác định mã nhận dạng giao diện của chúng và định cấu hình địa chỉ IPv6 liên kết cục bộ với tiền tố FE80 :: 0/64. Các thiết bị luồng cũng hỗ trợ địa chỉ phát multicast, bao gồm phát multicast liên kết cục bộ tất cả các node, phát multicast liên kết cục bộ tất cả router.

### 2.1.3.2 6LoWPAN

6LoWPAN là viết tắt của IPv6 Over Low Power Wireless Personal Networks. Giao thức 6LoWPAN xác định cơ chế kiểm soát gói tin khi giao tiếp qua chuẩn IPv6 (trên tiêu chuẩn IEEE 802.15.4) [6]

- Phân mảnh gói tin và lắp ráp gói tin IPv6: 6LoWPAN đáp ứng kích thước khung tối đa của IEEE 802.15.4. Ví dụ, với một gói 1280 byte (IPv6 MTU), 6LoWPAN sẽ phân mảnh gói ở người gửi và tập hợp lại ở người nhận. Do đó, nó hoạt động như một lớp thích ứng giữa mạng (Network layer) và lớp liên kết (Link layer) để giải quyết vấn đề.

- Nén dữ liệu: 6LoWPAN cung cấp cơ chế nén để giảm kích thước tiêu đề khung và giảm tiêu thụ năng lượng của thiết bị.
- Đóng gói tin: 6LoWPAN lấy các gói và bọc chúng bằng tiêu đề đóng gói, sau đó gửi chúng bằng IEEE 802.15.4 MAC và PHY
- Chuyển tiếp gói tin thuộc lớp liên kết: 6LoWPAN hỗ trợ chuyển tiếp gói lớp liên kết. Nó sử dụng một cơ chế low overhead để chuyển tiếp các gói multihop trong một mạng lưới. Đặc biệt, Thread sử dụng định tuyến IP để chuyển tiếp gói tin. Bảng định tuyến được duy trì với mỗi điểm đến và bước tiếp theo đến nó. Do đó, Thread sẽ sử dụng 6LoWPAN mesh header cho quá trình chuyển tiếp tiếp theo.

### 2.1.4 Lớp vận chuyển

User Datagram Protocol (UDP) là một giao thức cho phép các ứng dụng máy tính có thể gửi tin nhắn (datagram) đến các server khác qua mạng giao thức internet [7]. UDP sử dụng một mô hình giao tiếp đơn giản với các cơ chế không phức tạp. Ngoài ra, UDP cung cấp checksum cho dữ liệu và port numbers cho các chức năng. Tuy nhiên, nó không hỗ trợ handshake và có thể làm lộ dữ liệu của người dùng. UDP không đảm bảo việc truyền gửi và bảo vệ gói tin. Transmission Control Protocol (TCP) được thiết kế để cung cấp giao tiếp đáng tin cậy hơn. TCP là giao thức connection-oriented. server phải lắng nghe yêu cầu của client một cách thụ động trước khi thiết lập kết nối. Sau khi thiết lập kết nối giữa server và client, gói tin được phép gửi. TCP hỗ trợ three-way handshake, truyền lại và phát hiện lỗi.

Mạng Thread hỗ trợ UDP (theo chuẩn RFC 768) để truyền tin giữa các thiết bị. TCP có thể được triển khai tùy chọn.

### 2.1.5 Lớp ứng dụng

Lớp ứng dụng là lớp xác định các giao thức và phương thức được sử dụng bởi các server trong mạng. Ví dụ: công tắc và bóng đèn làm ví. Lớp ứng dụng sẽ quy định cách công tắc đó sẽ giao tiếp với bóng đèn. Trong trường hợp sử dụng mạng Thread, nó sẽ xác định việc hình thành và tham gia một mạng được gọi là commissioning [8]. Thread cung cấp các dịch vụ chính sau:

- Truyền tin qua UDP: Phương pháp giao tiếp sử dụng 16-bit port number và địa chỉ IPv6. Bởi vì UDP đơn giản hơn TCP, UDP cho phép thông lượng dữ liệu cao hơn và giảm mức tiêu thụ năng lượng của một ứng dụng. Ngoài ra, ứng dụng UDP cần ít mã hóa hơn TCP, điều này cung cấp cho chip nhiều bộ nhớ hơn cho các mục đích khác.
- Multicast: Gửi một thông điệp đến nhiều node trên mạng Thread. Multicast cung cấp một chức năng tích hợp để truyền tin với các node, router và toàn bộ mạng Thread với IPv6.
- Các lớp ứng dụng sử dụng các dịch vụ IP như UDP và CoAP (Constrained Application Protocol) cho phép giao tiếp qua mạng Internet.

## 2.2 OpenThread

### 2.2.1 Tổng quát về công nghệ Thread

Thread là một giao thức dựa trên IPv6, được thiết kế cho các thiết bị IoT tiêu thụ năng lượng thấp [9]. Thread là một phần của mạng khu vực cá nhân không dây (Wireless personal area network - WPAN) và độc lập với các giao thức mạng khác dựa trên IEEE 802.15 như ZigBee, Z-wave, BluetoothLE, v.v. Các đặc điểm chính của mạng Thread:

- Hỗ trợ địa chỉ IPv6 và phân vùng IP đơn giản: Quá trình cài đặt và hoạt động của mạng Thread rất đơn giản. Các giao thức trong mạng Thread cho phép hệ thống tự cấu hình và tự động sửa lỗi định tuyến.
- Bảo mật: Các thiết bị không thể tham gia mạng trừ khi chúng được xác minh qua ứng dụng di động hoặc Border Router. Mọi thông tin liên lạc đều được mã hóa và giữa các thiết bị.
- Khả năng mở rộng và phạm vi: Một phân vùng mạng Thread có thể hỗ trợ 250 node trở lên.
- Được tạo cho nhà thông minh và tự động hóa: Mạng Thread hỗ trợ ứng dụng có độ trễ thấp (100 ms) và hoạt động tin cậy kể cả khi mất kết nối với các thiết bị.

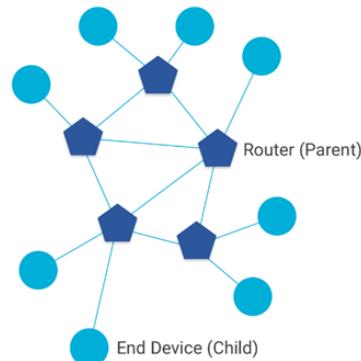
## CƠ SỞ LÝ THUYẾT

Bảng 1. Thông số thiết kế của mạng Thread được nhà sản xuất cung cấp

Thông số	Thread
Mục tiêu thiết kế	Tự động hóa trong nhà
Phân loại network	Mesh
Phạm vi	20 - 30 m
Số node tối đa trong mạng	250
Tần số hoạt động	2.4 GHz
Kỹ thuật trai phổ	Direct Sequence Spread Spectrum (DSSS)
Lưu lượng	250 kbps
Bảo mật	Banking - class, public - key cryptography
Lấy mẫu	O - QPSK

### 2.2.2 Kiến trúc mạng

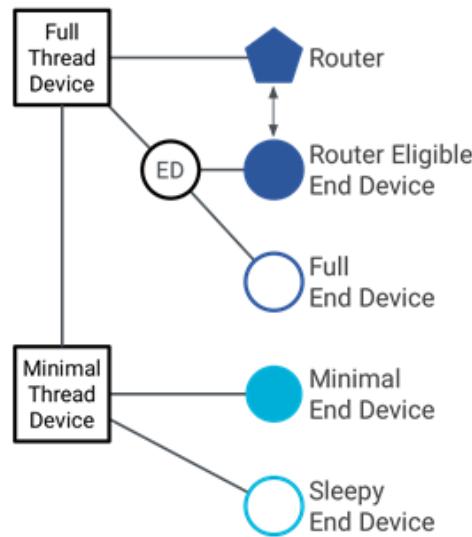
Tương tự với các mạng Mesh khác, node trong mạng Thread cũng được chia theo vai trò và loại thiết bị [9].



Hình 3. Vai trò Node trong mạng

#### 2.2.2.1 Vai trò thiết bị trong mạng

- Bộ định tuyến (Router): Bộ định tuyến phải luôn bật bộ thu phát của nó để đảm bảo các dịch vụ vận hành an toàn cho bộ kết hợp (thiết bị yêu cầu tham gia mạng Thread) và chuyển tiếp các gói cho các thiết bị trong mạng.
- Thiết bị đầu cuối (End Device - ED): Một ED giao tiếp chủ yếu với một router duy nhất và không chuyển tiếp các gói nhurable. Do đó, nó có thể vô hiệu hóa bộ thu phát của nó để tiết kiệm điện năng.



Hình 4. Phân loại theo thiết bị

Có một mối quan hệ giữa Router và ED. Router là parent và ED là child. Vì vậy, một ED có duy nhất một Router tương ứng.

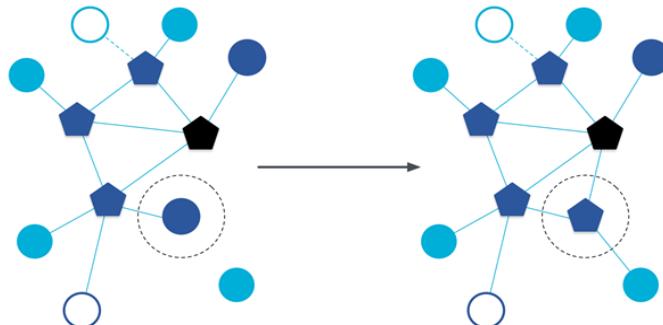
### 2.2.2.2 Loại thiết bị

- Full Thread Device (FTD): Loại thiết bị này luôn giữ cho các bộ thu phát của nó hoạt động, đăng ký địa chỉ multicast của tất cả các router và giữ bảng địa chỉ IPv6. FTD có thể hoạt động như một Router (parent) hoặc một ED (child). FTD được chia thành ba nhóm nhỏ:
  - Router
  - Router Eligible end device (REED) - thiết bị này có thể tự động hoán đổi vai trò giữa router và ED

- Full end device (FED) - thiết bị này không thể tự quảng bá như một router.
- Minimal Thread Device = (MTD): Thiết bị này không đăng ký các địa chỉ multicast và chuyển tiếp các gói đến parent của nó. MTD chỉ có thể hoạt động như một ED. MTD được chia thành hai nhóm:
  - Minimal End Device(MED) - bộ thu phát của nó luôn bật và không cần phải thăm dò các gói tin từ parent.
  - Sleepy End Device (SED): - thiết bị này thường ở chế độ ngủ và thỉnh thoảng sẽ thức dậy để thăm dò các gói tin từ parent. Do đó, nó có thể giữ cho mức tiêu thụ điện năng luôn ở mức thấp.

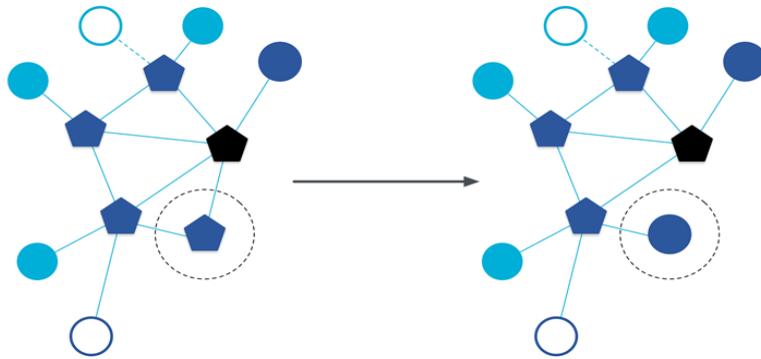
#### 2.2.2.3 Quá trình thăng cấp và hạ cấp

- **Thăng cấp:** Khi chỉ có một node REED trong mạng và một ED mới muốn tham gia, node REED sẽ tự động quảng bá chính nó như một Router. Ví dụ: Khi node màu xanh (góc dưới bên phải) muốn tham gia vào mạng, node màu xanh đậm (được khoanh tròn) sẽ trở thành một router và cho phép node màu xanh tham gia.



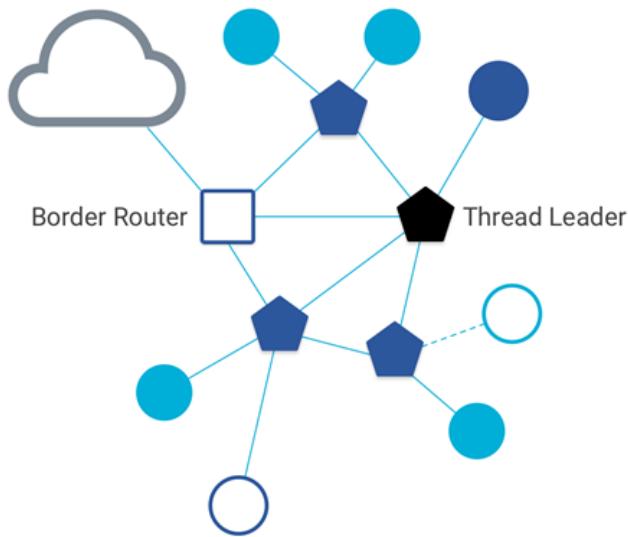
Hình 5. Quá trình thăng cấp

- **Hạ cấp:** Khi một người REED mất child của mình, nó sẽ tự hạ cấp thành một ED. Ví dụ: Sau khi mất con, node hình ngũ giác màu đen biến thành node ED.



Hình 6. Quá trình *hẹn cấp*

#### 2.2.2.4 Các vai trò khác trong mạng



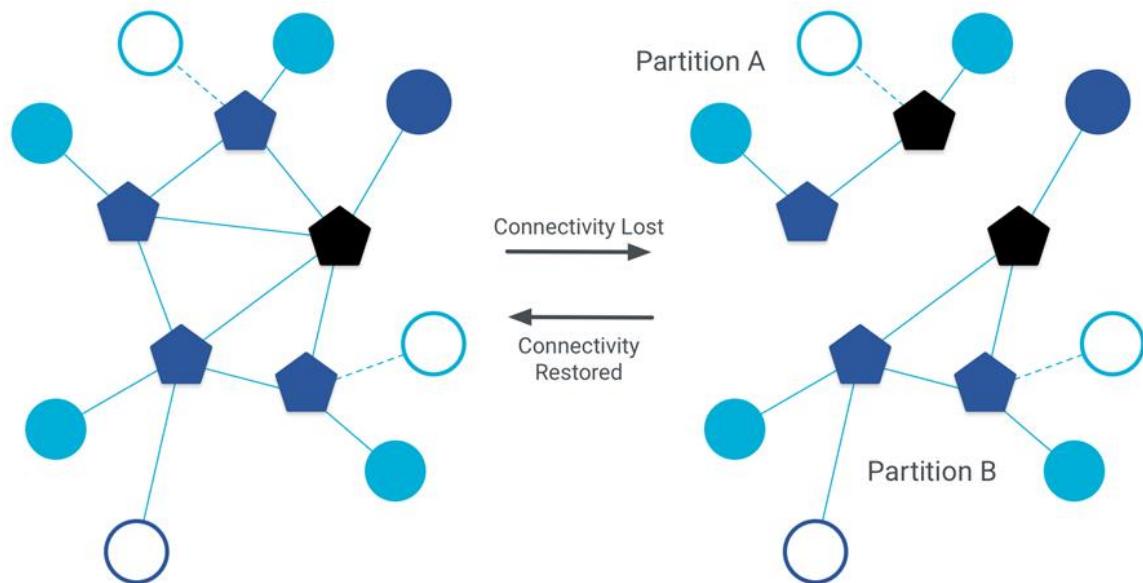
Hình 7. Vị trí của *Thread leader* và *Border Router*

**Thread Leader:** Node quản lý dữ liệu đăng ký các ID router được chỉ định và chấp nhận các yêu cầu từ REED trở thành router. Thread Leader như các router khác trong mạng, tất cả đều có child. Thread Leader quản lý các router khác bằng Constrained Application Protocol (CoAP). Đặc biệt, tất cả thông tin có trong Leader cũng hiển thị trong các Router khác. Do đó, nếu Thread Leader bị sập, Thread Leader khác sẽ được bầu mà không có lỗi mạng. Mỗi phân vùng mạng luôn có một Thread Leader.

**Border Router:** Có nhiều hơn một Border Router trong một mạng Thread. Nhiệm vụ của Border Router là cung cấp dịch vụ cho các thiết bị trong mạng. Border Router hoạt

động như một cổng, điều khiển kết nối giữa mạng Thread và mạng non-Thread (Wi-Fi và Ethernet). Điều này giúp mọi thiết bị nội mạng có thể kết nối với Internet.

### 2.2.2.5 Phân vùng mạng



Hình 8. Minh họa hiện tượng phân vùng mạng

Mạng Thread có thể được chia thành các phân vùng do sự ngắt kết nối giữa các thiết bị trong mạng. Khi điều này xảy ra, mỗi phân vùng hoạt động như một mạng Thread riêng biệt với Thread Leader, Router ID và dữ liệu mạng trong khi vẫn giữ nguyên các thông tin bảo mật. Nếu các phân vùng lấy lại kết nối, chúng sẽ hợp nhất thành một phân vùng duy nhất.

### 2.2.2.6 Giới hạn thiết bị

Bảng bên dưới mô tả số lượng thiết bị giới hạn trong một mạng Thread

Bảng 2. Giới hạn số thiết bị trong mạng

Vai trò	Giới hạn
Leader	1
Router	32

End Device	511 node/router
------------	-----------------

Mạng Thread giữ số lượng router nằm giữa 16 và 23. Nếu REED kết nối với D khi số lượng router dưới 16 thì REED sẽ thăng cấp thành Router.

### 2.2.3 Cơ chế đánh địa chỉ IPv6

#### 2.2.3.1 Phạm vi địa chỉ

Có 3 loại địa chỉ trong mạng Thread:

- Link-local: Tất cả các điểm cuối đều có thể truy cập được bằng đường truyền vô tuyến duy nhất. Địa chỉ link-local có prefix là ‘FE80 :: / 16’
- Mesh-local: Tất cả các điểm cuối đều có thể truy cập được trong cùng một mạng. Địa chỉ Mesh-local có prefix là ‘FE00 :: / 8’
- Global: Tất cả các điểm cuối đều có thể truy cập được từ ngoài mạng Thread

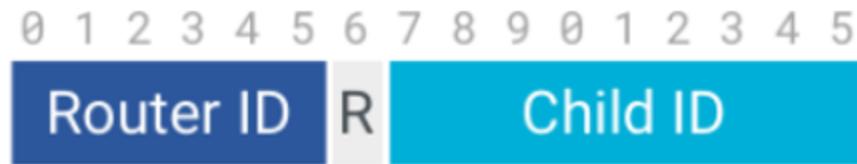
#### 2.2.3.2 Routing Locator

Routing Locator (RLOC) của các thiết bị giúp Thread xác định các điểm cuối dựa trên vị trí của chúng trong cấu trúc liên kết mạng. Công thức RLOC được trình bày dưới đây.



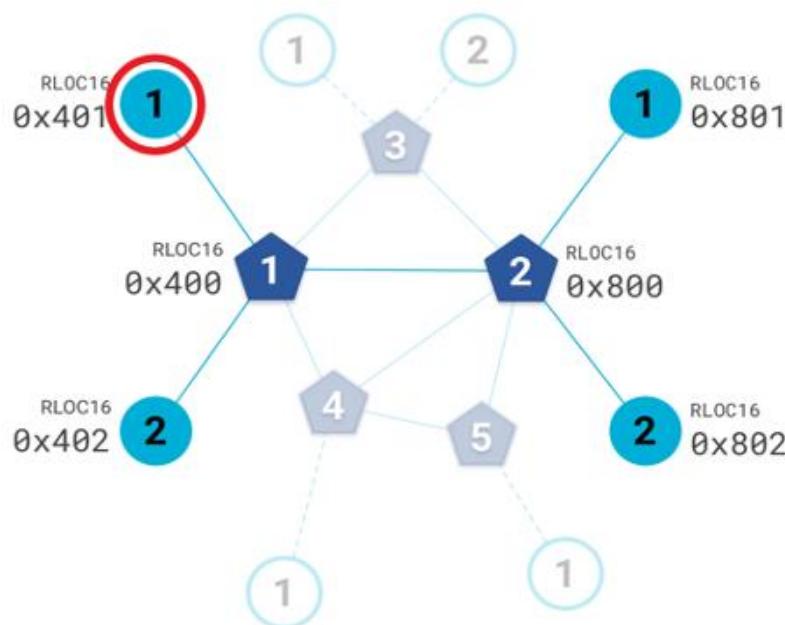
Hình 9. Công thức Routing Locator

**RLOC16:** Bao gồm: Router ID và Child ID. Tất cả Router đều lưu trữ Child ID của nó. Thread sử dụng các ID này để tính toán 16 bit cuối cùng của Routing Locator, còn được gọi là RLOC16.



Hình 10. RLOC16

Ví dụ: Tính toán RLOC16 của node màu đỏ trong hình dưới đây



Hình 11. Ví dụ minh họa RLOC16

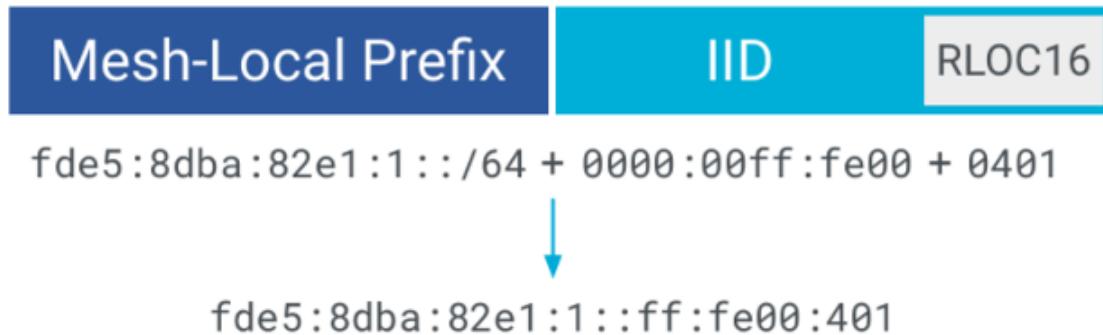
- Router ID = 1 (Child ID của ED tương ứng với Router ID)
- Child ID = 1 (Đây là thiết bị đầu tiên kết nối với Router 1)
- RLOC16 = 0000 1000 0000 0001 = 0x401

**Mesh-local prefix:** Người dùng có thể thay đổi prefix mesh-local của mạng Thread trong bước cài đặt mạng. Giá trị mặc định là FDE5: 8DBA: 82E1: 1 :: / 64

**IID:** RLOC16 là một phần của mã định danh ED, còn được gọi là Interface Identifier (IID). Công thức của IID là: 0000: 00FF: FE00: RLOC16. Sự kết hợp giữa IID và Mesh-Local Prefix tạo ra một RLOC hoàn chỉnh, là một địa chỉ IPv6. Ví dụ: Bằng cách sử dụng prefix

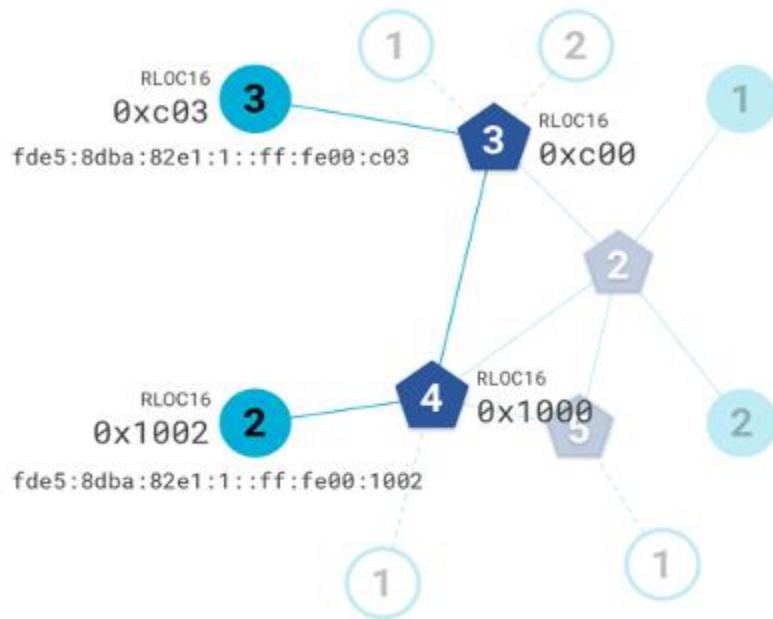
## CƠ SỞ LÝ THUYẾT

Mesh-Local là FDE5: 8DBA: 82E1: 1 :: / 64, RLOC cho một node có RLOC16 = 0x401 là:



Hình 12. RLOC hoàn chỉnh

Lưu ý: Khi topology mạng thay đổi, mã RLOC cũng sẽ được cập nhật. Hình bên dưới mô tả khi node 0x400 ngắt kết nối, node 0x401 và 0x402 tham gia mạng với RLOC mới.



Hình 13. RLOC thay đổi khi Topology thay đổi

### 2.2.3.3 Unicast

Các thiết bị Thread có thể có nhiều loại địa chỉ unicast, bao gồm cả RLOC. Một loại địa chỉ khác được gọi là Endpoint Identifiers (EIDs), xác định giao tiếp duy nhất trong

## CƠ SỞ LÝ THUYẾT

phân vùng mạng Thread. Trong topology mạng Thread, các EID là độc lập. Các loại unicast phổ biến được liệt kê bên dưới:

Bảng 3. Các loại địa chỉ Unicast

Link-Local Address (LLA)	
Một EID xác định thiết bị đầu cuối có thể truy cập được bằng một đường truyền vô tuyến.	
Ví dụ	FE80::54DB:881C:3845:57F4
IID	Dựa trên IEEE 802.15.4 mở rộng
Mục đích sử dụng	Tìm kiếm Router xung quanh, thiết lập kết nối với trao đổi thông tin định tuyến
	Không phải là địa chỉ định tuyến được
	Có prefix là FE80::/16

Mesh-Local EID (ML-EID)	
Một EID xác định một thiết bị đầu cuối độc lập với topology. Nó chỉ đường đến được một node Thread trong cùng một phân vùng. Nó còn được gọi là Unique Local Address (ULA).	
Ví dụ	FDE5:8DBA:82E1:1:416:993C:8399:35AB
IID	Ngẫu nhiên sau khi commissioning
Mục đích sử dụng	Địa chỉ định tuyến của Thread
	Không thay đổi khi topology thay đổi
	Có prefix là FD00::/8

Anycast Locator (ALOC)	
Xác định node Thread thông qua tra cứu RLOC khi mạng không thể tìm thấy mã RLOC của node cần thiết.	

## CƠ SỞ LÝ THUYẾT

Ví dụ	FDE5:8DBA:82E1:1::FF:Fe00:FC01
IID	0000:00FF:FE00:FCXX
Mục đích sử dụng	Không được sử dụng trong các ứng dụng Thread
	FCXX là ALOC.

<b>Routing Locator (RLOC)</b>	
Xác định địa chỉ node Thread trong mạng.	
Ví dụ	FDE5:8DBA:821:1::ff:fe00:1001
IID	0000:00FF:FE00:RLOC16
Mục đích sử dụng	Khởi tạo khi thiết bị kết nối vào mạng
	Dùng để gửi gói tin IPv6 qua mạng Thread
	Thay đổi khi topology mạng thay đổi

<b>Global Unicast Address (GUA)</b>	
Xác định địa chỉ node Thread trong mạng nhưng ở tầm ngoài mạng.	
Ví dụ	2000::54DB:881C:3845:57F4
IID	DHCP - gán bởi DHCPv6
	Thủ công - gán bởi lớp Application
	SLAAC - tạo ra ngẫu nhiên bởi node đó
Mục đích sử dụng	Địa chỉ public IPv6
	Có prefix 2000::/3

## 2.2.3.4 Multicast

Multicast cho phép chúng ta giao tiếp với nhiều thiết bị cùng một lúc. Có hai hoặc nhiều địa chỉ multicast trong mạng Thread cho các nhóm thiết bị khác nhau (FTD, MED, v.v.), tùy thuộc vào phạm vi.

Bảng 4. Phân loại địa chỉ Multicast

Địa chỉ IPv6	Scope	Sử dụng
FF02::1	Link-local	FTDs và MEDs
FF02::2	Link-local	FTDs
FF03::1	Mesh-local	FTDs và MEDs
FF03::2	Mesh-local	FTDs

## 2.2.3.5 Anycast

Sử dụng giao thức Mesh Link Establishment (MLE), Thread có thể thiết lập liên kết và phát thông tin về mạng tới tất cả các thiết bị trong cấu trúc liên kết mạng. Truyền tin trong mạng Thread hoạt động giống như Routing Information Protocol (RIP), một giao thức định tuyến theo vecto khoảng cách (distance vector).

Có ba loại thông tin mà MLE truyền đến các thiết bị muốn thiết lập liên kết:

- Thông tin định tuyến
- Thông tin về Leader bao gồm Leader RLOC, ID phân vùng và trọng số phân vùng
- Dữ liệu mạng bao gồm on-mesh prefix, tự động định cấu hình địa chỉ và một số đường đi cụ thể hơn.

MLE thường thực hiện các bước sau để thiết lập liên kết:

- Tìm liên kết đến các thiết bị lân cận
- Đánh giá chất lượng liên kết của các thiết bị lân cận
- Xây dựng liên kết đến các thiết bị lân cận

- Truyền các tham số liên kết với thiết bị, bao gồm loại thiết bị, bộ đếm khung hình và thời gian chờ.

### 2.2.4 *Tạo một mạng mới*

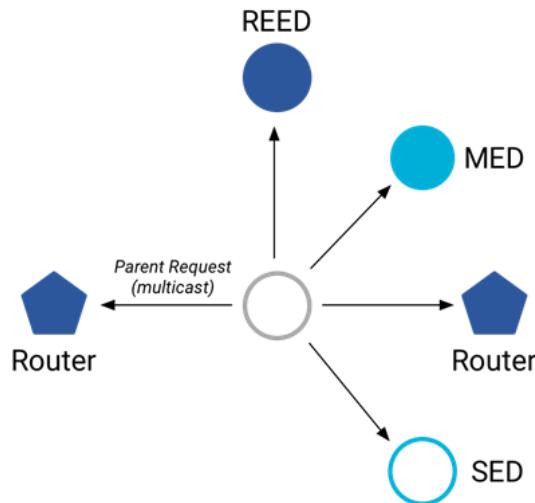
Nếu thiết bị phải tạo mạng mới, thiết bị sẽ chọn kênh ít bận nhất và PAN ID không được sử dụng. Sau đó, nó biến thành một Router và tự quảng bá mình với tư cách là Thread Leader. Thiết bị này phát các bản tin MLE Advertisement tới các thiết bị 802.15.4 khác để thông báo cho họ về trạng thái liên kết hiện tại của nó. Nó cũng phản hồi các Beacon Requests của quá trình quét hoạt động trên các thiết bị Thread khác.

### 2.2.5 *Tham gia một mạng có sẵn*

Nếu thiết bị phải tham gia vào một mạng hiện có, thiết bị sẽ điều chỉnh kênh, PAN ID, XPAN ID và Tên mạng của nó để khớp với tên của mạng mục tiêu bằng Thread Commissioning. Nó thực hiện quy trình MLE Attach để tham gia với tư cách ED (Child). Mọi thiết bị, bao gồm cả REED, ban đầu tham gia mạng Thread với tư cách là child. Toàn bộ quá trình này được gọi là liên kết child-parent.

- Child phát một parent request tới tất cả các Router và REED trong mạng.
- Tất cả các Router và REED lân cận đều gửi parent responses, bao gồm cả thông tin về chính nó.
- Child nhận node parent và chỉ gửi Child ID request đến thiết bị đó.
- Parent sau đó sẽ trả lời bằng Child ID respond để xác nhận việc thiết lập liên kết.

#### 2.2.5.1 Parent Request



Hình 14. Parent Request

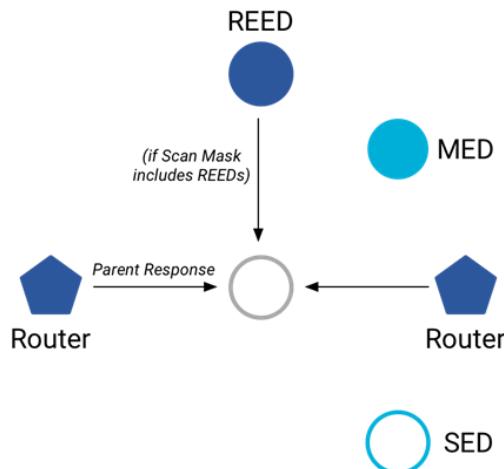
Parent request là một multicast request được gửi từ thiết bị muốn tham gia vào mạng. Mục đích của yêu cầu này là khám phá các Router và REED gần đó trong mạng.

Bảng 5. Nội dung Parent Request

<b>Mode</b>	Thông báo thông tin về thiết bị
<b>Challenge</b>	Kiểm tra Parent response để ngăn chặn relay attacks
<b>Scan Mask</b>	Giới hạn request đến Router và REED

#### 2.2.5.2 Parent Response

Parent Response là một unicast response đối với Parent request. Nó nhằm mục đích cung cấp thông tin chi tiết về một Router hoặc REED cho thiết bị muốn gia nhập mạng.



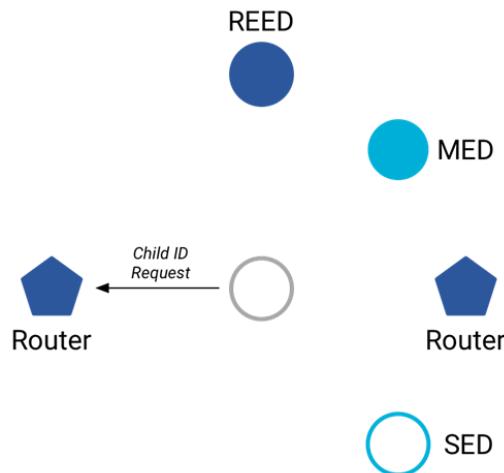
Hình 15. Parent Response

Bảng 6. Nội dung Parent Response

<b>Phiên bản</b>	Phiên bản Thread đang sử dụng
<b>Response</b>	Parent Request Challenge
<b>Link frame counter</b>	802.15.4 frame counter của Router/REED
<b>MLE frame counter</b>	MLE frame counter của Router/REED
<b>Địa chỉ nguồn</b>	RLOC16 của Router/REED
<b>Thông tin liên kết</b>	Chất lượng liên kết
<b>Kết nối</b>	Mô tả mức độ kết nối của Router/REED
<b>Challenge</b>	Kiểm tra Child ID Request để ngăn chặn relay attack

#### 2.2.5.3 Child ID Request

Child ID Request là một unicast request từ thiết bị muốn tham gia được gửi đến Router/REED. Nó dự định thiết lập một liên kết Child-Parent. Nếu REED nhận được yêu cầu, nó sẽ tự quảng bá như một Router sau đó chấp nhận yêu cầu.



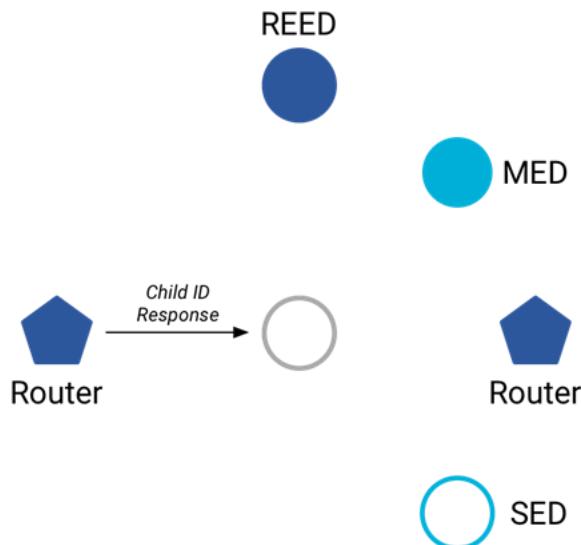
Hình 16. Child ID Request

Bảng 7. Nội dung Child ID Request

Phiên bản	Phiên bản Thread sử dụng
Phản hồi	Parent Request Challenge
Link frame counter	802.15.4 frame counter của child
MLE frame counter	MLE frame counter của child
Mode	Mô tả thông tin của child
Timeout	Thời gian ngừng hoạt động trước khi child được gia nhập vào mạng
Địa chỉ đăng kí (chỉ dành cho MEDs và SEDs)	Đăng kí địa chỉ IPv6

#### 2.2.5.4 Child ID Reponse

Child ID Response là unicast response từ parent được gửi đến child để xác nhận liên kết Child-Parent.



Hình 17. Child ID Response

Bảng 8. Nội dung Child ID Response

<b>Địa chỉ nguồn</b>	RLOC16 của Parent
<b>Địa chỉ 16</b>	RLOC16 của Child
<b>Thông tin về Leader</b>	Thông tin về Leader của parent bao gồm RLOC, ID của phân vùng mạng và hệ số phân vùng
<b>Thông tin về mạng</b>	Thông tin về Thread, bao gồm on-mesh prefix, đường đi và thiết lập địa chỉ
<b>Đường đi (nếu parent là REED)</b>	Bảng thông tin định tuyến
<b>Timeout</b>	Thời gian ngừng hoạt động trước khi child được gia nhập vào mạng
<b>Địa chỉ đăng ký (chỉ dành cho MEDs và SEDs)</b>	Xác nhận địa chỉ IPv6

## 2.2.6 Lựa chọn Router

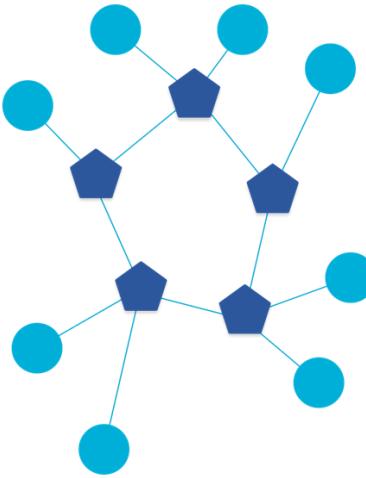
### 2.2.6.1 Connected Dominating Set (CDS)

CDS là tập hợp các Router kết nối với nhau và đảm bảo 3 yếu tố sau:

- Có một đường dẫn duy nhất giữa hai Router trong bộ.

- Bất kỳ Router nào trong mạng Thread có thể tiếp cận bất kỳ Router nào khác bằng cách ở hoàn toàn trong tập hợp.
- Mọi ED trong mạng Thread đều kết nối thẳng tới một Router

Ví dụ: Hình ảnh minh họa CDS trong cấu trúc liên kết mạng. Nó là bộ 5 router, kết nối và đảm bảo 3 yếu tố trên.



Hình 18. Minh họa Connected Dominating Set

Có một thuật toán phân tán duy trì CDS, đảm bảo mức độ dự phòng nhỏ. Mọi thiết bị ban đầu tham gia mạng với tư cách ED (child). Khi mạng Thread thay đổi, thuật toán sẽ thêm hoặc bớt Router để lưu CDS.

Mục tiêu của việc thêm Router vào mạng:

- Tăng phạm vi phủ sóng dựa trên ngưỡng Router (16 router)
- Cải thiện sự đa dạng của đường đi
- Giữ mức dự phòng nhỏ
- Mở rộng kết nối cộng với hỗ trợ nhiều child hơn

Mục tiêu của việc loại bỏ Router từ mạng:

- Giảm trạng thái định tuyến dựa trên ngưỡng Router (32 router)
- Cung cấp Router mới trong các phần khác của mạng nếu cần

### 2.2.6.2 Thăng cấp lên Router

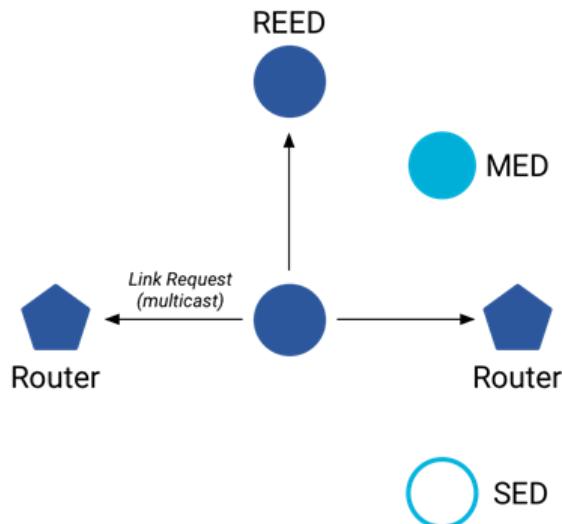
Sau khi tham gia mạng Thread, thiết bị child có thể được chọn để trở thành một Router. Trước khi tạo ra MLE Link Request, child sẽ gửi một gói tin địa chỉ của Leader để yêu cầu một Router ID. Nếu Leader chấp nhận, nó sẽ phản hồi bằng một Router ID. Do đó, child tự quảng bá mình như một Router.

Ngoài ra, thiết bị sử dụng MLE Link Request để thiết lập liên kết 2 chiều với router lân cận bằng các bước sau:

- Router mới sẽ gửi một multicast Link request đến các router gần đó.
- Router trả lời bằng Link Accept and Request.
- Router mới trả lời mỗi Router trong mạng bằng một unicast Link Accept để thiết lập Liên kết Router-Router.

### 2.2.6.3 Link Request

Link Request là một request từ một router mới gia nhập mạng đến tất cả các router trong mạng Thread. Thiết bị sẽ gửi multicast Link request tới FF02 :: 2. Sau khi phát hiện ra các router khác của MLE Advertisements, các thiết bị sẽ gửi unicast Link Requests.



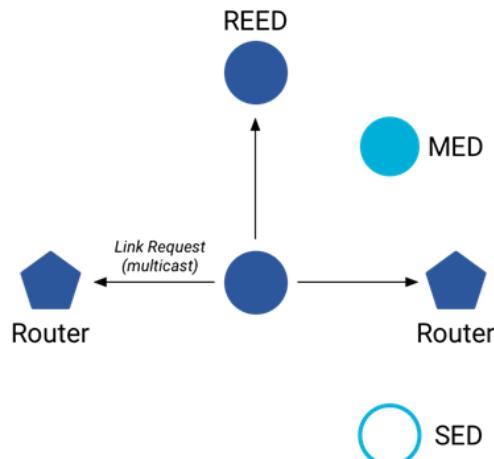
Hình 19. Link Request

Bảng 9. Thông tin Link Request

<b>Phiên bản</b>	Phiên bản Thread đang sử dụng
<b>Mục tiêu</b>	Kiểm tra Link response để ngăn chặn relay attack
<b>Địa chỉ nguồn</b>	RLOC16 của sender
<b>Thông tin Leader</b>	Thông tin về Router

#### 2.2.6.4 Link Accept

Link accept trả lời cho Link request từ router lân cận. Nó cung cấp dữ liệu về bản thân và đồng ý liên kết với router lân cận.



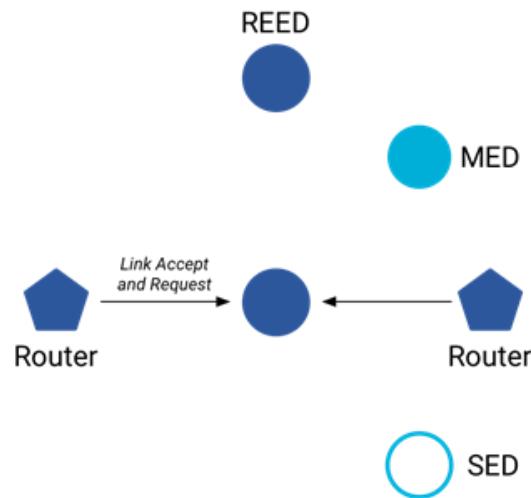
Hình 20. Link Accept

Bảng 10. Nội dung Link Accept

<b>Phiên bản</b>	Phiên bản Thread đang sử dụng
<b>Response</b>	Kiểm tra Link response để ngăn chặn relay attack
<b>Link frame counter</b>	802.15.4 Frame Counter của sender
<b>MLE frame counter</b>	MLE Frame Counter của sender
<b>Nguồn</b>	RLOC16 của sender
<b>Thông tin Leader</b>	Thông tin về router

#### 2.2.6.5 Link Accept và Request

Link Accept và Request là kết hợp của Link Accept và Link Request. Thread sử dụng yêu cầu này để giảm lượng gói tin trao đổi trong mạng.



Hình 21. Link Accept và Request

#### 2.2.6.6 Hạ cấp xuống REED

Nếu một router bị hạ cấp xuống REED, các liên kết router-router của nó sẽ bị ngắt và thiết bị bắt đầu quá trình MLE Attachment để tạo thành liên kết child-parent.

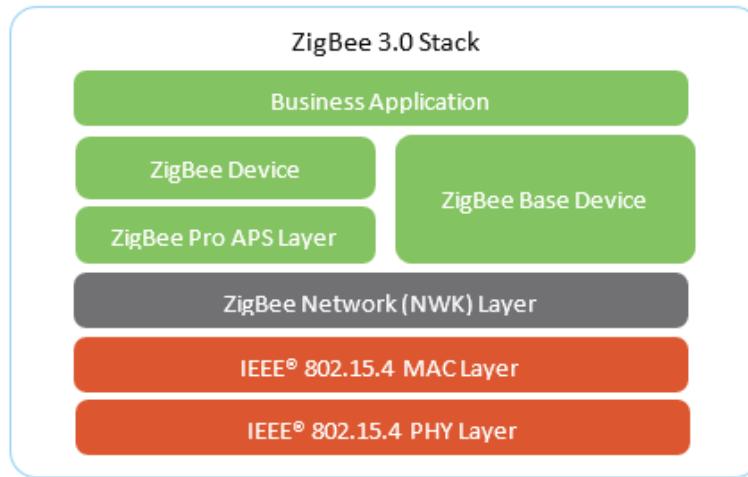
#### 2.2.6.7 Kết nối nhận 1 hướng

Sau khi router khởi động lại, những router lân cận kiểm tra lại kết nối với nó. Router sẽ gửi Link request với lân cận để thiết lập lại kết nối router-router.

### 2.3 Giao thức Zigbee

#### 2.3.1 Tổng quát về công nghệ Zigbee

Zigbee là một tiêu chuẩn mở không dây cho các mạng IoT. Lớp MAC và lớp PHY của Giao thức mạng Zigbee dựa theo quy định trong chuẩn IEEE 802.15.4.



Hình 22. Mô hình phân lớp của Zigbee

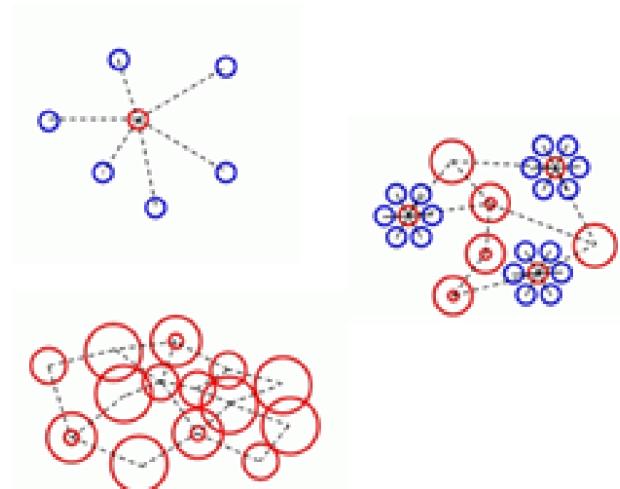
Về mặt kiến trúc, một mạng Zigbee có thể hỗ trợ tối đa 65.000 nodes trên mỗi mạng. Mạng Zigbee định tuyến lướt dựa trên Ad-hoc on-demand Distance Vector protocol (AODV protocol). Zigbee có hai kiến trúc mạng, mỗi kiến trúc có chế độ bảo mật riêng [10]:

- Distributed Security Network: không có Zigbee Coordinator, bất kỳ Zigbee Routers nào cũng có thể tạo thành mạng. Thiết bị tham gia mạng chỉ cần xác thực với Node parent của nó trước khi hoạt động.
- Centralized Security Network. bao gồm một thiết bị được gọi là Zigbee Coordinator tạo thành mạng Zigbee và là thực thể đáng tin cậy của mạng (Trust center). Thiết bị tham gia mạng cần xác thực với Trust Center trước khi hoạt động.

Khi mạng Zigbee được bảo mật, nó sử dụng mã hóa AES 128 bit để mã hóa dữ liệu trong mạng. Zigbee hỗ trợ nhiều cách để thêm thiết bị mới vào mạng Zigbee (gọi là Commissioning), bao gồm: chạy thử chế độ EZ, chạy thử Touchlink và chạy thử ngoài băng tần.

### 2.3.2 Zigbee Mesh Networking

Mạng mesh giúp các hệ thống vô tuyến đáng tin cậy hơn bằng cách cho phép node mạng chuyển tiếp tin nhắn cho các node lân cận. Ví dụ: nếu một node không thể gửi gói tin trực tiếp đến một node khác, mạng mesh hỗ trợ chuyển tiếp gói tin qua một hoặc nhiều node trung gian. Zigbee hỗ trợ ba loại topology liên kết mạng mesh [11]:



Hình 23. Ba loại topology trong mạng Zigbee

- Dạng hình sao: Trong một mạng sao, một node/hub là điểm trung tâm của tất cả các thông tin liên lạc. Cấu trúc liên kết này khiến việc truyền tin bị giới hạn bởi bán kính truyền thông trung tâm:
  - Node trung tâm là Coordinator, chịu trách nhiệm thành lập mạng.
  - Các node lá bên ngoài sử dụng nguồn tiết kiệm năng lượng.
- Dạng lưới đầy đủ: Trong một mạng lưới đầy đủ, tất cả các node là các node bộ định tuyến, bao gồm cả Coordinator sau khi nó tạo thành mạng vì tất cả các node có thể chuyển tiếp thông tin cho tất cả các node khác. Cấu trúc liên kết này ít bị tổn thương

khi liên kết gặp lỗi: Khi một thiết bị sự cố, các thiết bị xung quanh vẫn hoạt động bình thường.

- Dạng kết hợp (hybrid): Mô hình này kết hợp mạng hình sao và mạng lưới đầy đủ.

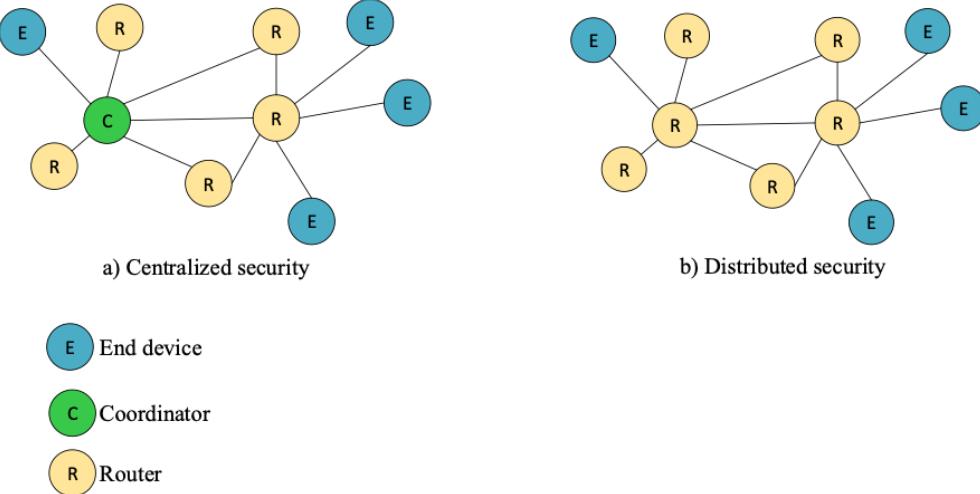
### 2.3.3 Phân loại thiết bị và vai trò

Mạng Zigbee có hai loại thiết bị [11]:

- Full function Device (FFD). Các FFD không ngủ và lý tưởng cho việc định tuyến các gói.
- Reduced Function Device (RFD). Các RFD ngủ giữa các lần truyền để tiết kiệm năng lượng, khiến chúng không phù hợp để định tuyến các gói.

Các thiết bị trong mạng Zigbee có 3 vai trò chính [11]:

- Zigbee coordinator: Tạo lập mạng Centralized Security Network và là bộ điều khiển chính. Trong quá trình Device Commissioning, Coordinator chịu trách nhiệm xác thực và ủy quyền cho các thiết bị tham gia. Coordinator cũng là Trust Center và thực hiện các hoạt động liên quan đến bảo mật,. Coordinator phải là FFD
- Zigbee router: FFD tham gia vào mạng Zigbee, không phải là Coordinator nhưng có thể đóng vai trò là Coordinator . Nó chịu trách nhiệm định tuyến gói tin trong mạng. Trong Centralized Security Network, trong khi thêm thiết bị mới vào mạng, Routers có thể chuyển tiếp tin nhắn giữa thiết bị mới và Coordinator.
- ZigBee end device (ZED): là các node lá. ZED giao tiếp qua các node cha và không giống như các thiết bị bộ định tuyến, không thể chuyển tiếp tin nhắn dành cho các node khác. Các End device có thể có nhiều loại:
  - Sleepy end device tắt nguồn bộ thu phát khi không hoạt động và do đó tiết kiệm. Tuy nhiên, phải thăm dò node cha để truyền nhận dữ liệu khi có yêu cầu. Các thiết bị này đôi khi còn được gọi là rx-off-when-idle.
  - Non-sleepy end device không định tuyến tin nhắn cho các thiết bị khác nhưng chúng vẫn được cấp nguồn trong quá trình hoạt động. Các thiết bị này được gọi là rx-on-when-idle.



Hình 24. Centralized security và Distributed security

Ngoài ra, trong quá trình thêm thiết bị mới vào mạng Zigbee, có thêm các vai trò bổ sung như sau:

- Joiner: Đây là một thiết bị Zigbee sẽ được thêm vào mạng.
- Trust Center (TC): Trong Centralized Security Network, ứng dụng TC chạy trên thiết bị đáng tin cậy của mạng Zigbee, tức là Zigbee coordinator. Chức năng chính của TC là:
  - Cấp phép cho Joiner tham gia vào mạng
  - Khởi tạo và quản lý các khóa mã hóa đối xứng sẽ được sử dụng để mã hóa tin nhắn giữa hai node trong mạng ở tầng ứng dụng (Application layer)
  - Xác định cấu hình chính sách bảo mật của mạng. Ví dụ thiết bị nào được phép tham gia mạng, mức độ bảo mật trong mạng.
- Parent: Bất kỳ Router nào nằm trong phạm vi radio của Joiner và chuyển tiếp tin nhắn giữa Joiner và Coordinator là một parent. Đối với RFD, Router cha của nó lưu các tin nhắn đến và gửi lại cho RFD khi thiết bị thức dậy.
- Initiator: Bộ khởi tạo là thiết bị kích hoạt quá trình Commissioning Touchlink.
- Target: Thiết bị đích sẽ tham gia vào mạng của bộ khởi tạo nếu bộ khởi tạo có khả năng hình thành mạng hoặc hình thành mạng để thêm bộ khởi tạo nếu bộ khởi tạo không có khả năng hình thành mạng.

#### 2.3.4 Định danh trong mạng

Định danh thiết bị [12]:

- MAC Address: Địa chỉ 64 bit duy nhất được gán cho thiết bị bởi nhà sản xuất và không bao giờ thay đổi.
- Extended Unique Identifier (EUI): một định danh duy nhất 48 bit cho mọi thiết bị Zigbee. Nó là duy nhất cho mọi thiết bị và được cung cấp trong quá trình sản xuất bởi thiết bị Zigbee
- Short Address: địa chỉ 16 bit được gán cho thiết bị khi thiết bị tham gia mạng. Nó là duy nhất trong một mạng.

Định danh mạng: Định danh mạng xác định bởi hai thông số [12]:

- PAN ID (Personal Area Network Identification) là mã định danh 16 bit được chọn ngẫu nhiên trong quá trình hình thành mạng. Địa chỉ này được sử dụng trong việc truyền các gói dữ liệu. Có thể PAN ID thay đổi khi Mesh Network vẫn hoạt động.
- EPID (Extended PAN ID) là định danh 64 bit của mạng. Cho đến khi một thiết bị rời khỏi mạng, ID này sẽ giữ nguyên. EPID được sử dụng trong quá trình Device Commissioning.

### 2.3.5 Phương pháp định tuyến

Các phương pháp định tuyến được sử dụng trong Zigbee là [12]:

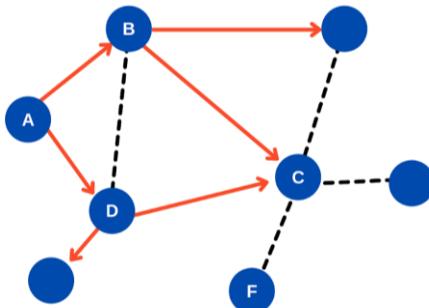
- Định tuyến bảng: Các tuyến đường được hình thành khi một node gửi yêu cầu tuyến đường để khám phá đường dẫn đến một node khác. Sau khi một tuyến đường được phát hiện giữa hai node, node nguồn sẽ gửi thông điệp của nó đến node đầu tiên trong tuyến đường, như được chỉ định trong bảng định tuyến của node nguồn. Mỗi node trung gian sử dụng bảng định tuyến riêng của mình để chuyển tiếp thông báo đến node tiếp theo (đó là "hop") dọc theo tuyến đường cho đến khi thông báo đến đích của nó. Nếu một tuyến đường không thành công, một lỗi tuyến đường được gửi lại cho người khởi tạo thông báo, người sau đó có thể khám phá lại tuyến đường.
- Định tuyến broadcast: Định tuyến broadcast là một cơ chế để gửi tin nhắn đến tất cả các thiết bị trong mạng. Một thông điệp broadcast được lặp lại bởi tất cả các thiết

bị có khả năng định tuyến trong mạng ba lần để đảm bảo phân phối đến tất cả các thiết bị. Broadcast là một phương thức đáng tin cậy để gửi một thông điệp, tuy nhiên nó nên được sử dụng một cách tiết kiệm vì tác động đến hiệu suất mạng. Broadcast lặp đi lặp lại có thể hạn chế bất kỳ lưu lượng truy cập khác điều đó có thể xảy ra trong mạng. Broadcast cũng không phải là một phương tiện giao hàng đáng tin cậy cho thiết bị ngủ vì node cha chịu trách nhiệm đệm tin nhắn cho thiết bị cuối đang ngủ nhưng có thể thả tin nhắn trước khi thiết bị cuối thức dậy để nhận nó.

- **Định tuyến multicast:** Định tuyến multicast hỗ trợ khả năng định tuyến one-to-many. Multicast được sử dụng khi một thiết bị muốn gửi tin nhắn đến một nhóm các thiết bị, chẳng hạn như một công tắc đèn gửi lệnh bật đến nhóm 10 đèn. Theo cơ chế này, tất cả các thiết bị được nối vào một nhóm multicast. Chỉ những thiết bị là thành viên của nhóm mới nhận được tin nhắn, mặc dù các thiết bị khác sẽ định tuyến các tin nhắn multicast này. Nó chỉ nên được sử dụng khi cần thiết trong các ứng dụng, vì sử dụng quá mức cơ chế multicast có thể làm giảm hiệu suất mạng.
- **Định tuyến nguồn/Many-to-one:** Định tuyến many-to-one là một cơ chế đơn giản để cho phép toàn bộ mạng có đường dẫn đến Coordinator. Trong định tuyến bảng thông thường, node trung tâm và các node xung quanh nó sẽ cần không gian bảng định tuyến để lưu trữ hop tiếp theo cho với mỗi thiết bị trong mạng. Với các thiết bị giới hạn bộ nhớ thường được sử dụng trong Zigbee, bảng định tuyến sẽ bị lược bỏ. Trong định tuyến many-to-one, thiết bị trung tâm trực tiếp gửi một thông điệp Discovery duy nhất thiết lập một bảng định tuyến đến tất cả các router để cung cấp hop tiếp theo cho thiết bị trung tâm. Điều này mang lại một kết quả tương tự như định tuyến bảng.

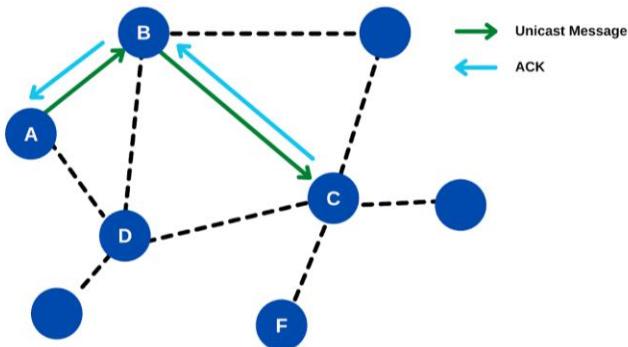
### 2.3.5.1 Quá trình khám phá đường đi (Route Discovery)

Khám phá đường đi bắt đầu khi một thông điệp Unicast được gửi từ thiết bị này sang thiết bị khác và không có tuyến đường tồn tại từ trước. Lúc này, phần mềm mạng sẽ bắt đầu quá trình khám phá tuyến đường. Giả sử rằng các bảng định tuyến của tất cả các thiết bị đều trống [12].



Hình 25. Quá trình khám phá đường đi 1

Ví dụ: A cần gửi tin nhắn đến C, như được hiển thị trong hình sau. A sẽ phát sóng nhắn tin cho toàn bộ mạng yêu cầu C trả lời. Thông điệp phát sóng này cũng phục vụ để thiết lập một tuyến đường tạm thời trả lại A, vì mỗi thiết bị trung gian ghi lại thiết bị mà nó nhận được tin nhắn. Các tuyến đường được cập nhật trên các nút trung gian - lưu ý rằng đây là những mục tạm thời có thời gian tồn tại ngắn hơn các mục thông thường và không nhằm mục đích sử dụng lại. Bởi vì A là một hàng xóm một bước, B và D không cần lưu trữ thông tin định tuyến về nó.



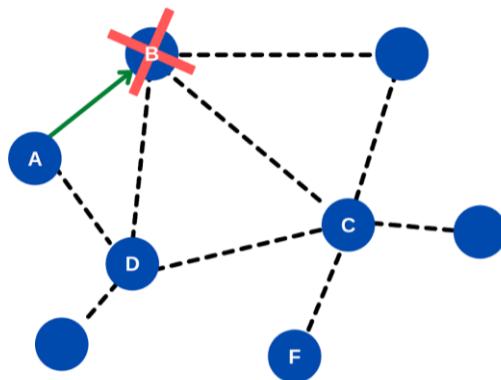
Hình 26. Quá trình khám phá đường đi 2

C có thể sử dụng B hoặc D làm bước nhảy tiếp theo trả lại A. Khi tin nhắn đến C, C sẽ gửi một tin nhắn Unicast đặc biệt (được gọi là tin nhắn Phản hồi tuyến đường (Route response message)) trả lại A bằng cách sử dụng tuyến đường tạm thời được xây dựng theo bước 1, như được hiển thị trong hình sau. Thông báo này được sử dụng bởi các thiết bị trung gian để thiết lập một tuyến đường trả lại C. Vì C là hàng xóm một bước, B không cần lưu trữ thông tin định tuyến về nó. D không liên quan đến phần quá trình khám phá vì

nó không được chọn bởi A trong bước trên. Khi thông báo đến thiết bị A, khám phá tuyến đường đã hoàn tất và tuyến mới có thể được sử dụng để gửi thông điệp dữ liệu từ A đến C.

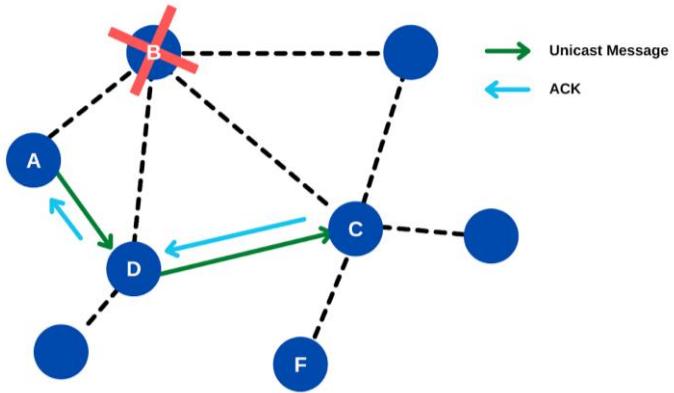
### 2.3.5.2 Quá trình sửa chữa đường đi (Route Repair)

Khi một tin nhắn unicast được gửi với yêu cầu ACK, thiết bị gửi sẽ được thông báo khi tin nhắn được gửi thành công. Nếu không nhận ACK, thì nó có thể thực hiện các bước để sửa chữa tuyến đường. Sửa chữa tuyến đường theo chính xác các bước tương tự như khám phá tuyến đường ở trên, nhưng nút bị hỏng (B, trong hình sau) không tham gia, dẫn đến một sự khác biệt trong lựa chọn tuyến đường [12].



Hình 27. Quá trình sửa chữa đường đi 1

Bảng định tuyến cho A được cập nhật để phản ánh rằng bước nhảy tiếp theo là D và thông báo được gửi thành công dọc theo đường dẫn mới, như được hiển thị trong hình sau.



Hình 28. Quá trình sửa chữa đường đi 2

### 2.3.6 Bảo mật

Bảo mật là tùy chọn trong quá trình khởi tạo mạng Zigbee. Mã hóa dữ liệu được áp dụng ở lớp mạng và lớp ứng dụng. Zigbee chỉ sử dụng mã hóa khóa đối xứng và tất cả mã hóa dựa trên AES-128 [13]. Bảo mật của mạng Zigbee là việc quản lý an toàn các khóa mã hóa được sử dụng trong mạng.

#### 2.3.6.1 Keys

Trong mạng Zigbee có thể có nhiều loại khóa để mã hóa và giải mã dữ liệu. Sau đây là danh sách các khóa được hỗ trợ trong Zigbee [14]:

- Network Key: Khóa này được sử dụng để mã hóa tất cả các tin nhắn trong mạng ở lớp mạng. Khóa được khởi tạo bởi thiết bị đầu tiên của mạng và được phân phối cho tất cả các thiết bị trong mạng trong quá trình Device Commissioning
- Global Link Keys: Khóa chung được định cấu hình trước cho mọi thiết bị Zigbee. Khóa này được sử dụng trong Device Commissioning để mã hóa phân phối Network Key. Trong trường hợp Distributed Security Network, khóa này được gọi là Khóa toàn cầu bảo mật phân tán. Trong trường hợp Commissioning Touchlink, khóa này được gọi là khóa Liên kết được cấu hình sẵn Touchlink. Khi giá trị của khóa này không được chỉ định, nó sẽ lấy giá trị của liên kết Trust Center toàn cầu mặc định, tức là 5A 69 67 42 65 65 41 6C 6C 69 61 6E 63 65 30 39 (ZigBeeAlliance09)

- Unique Link Key: Khóa được định cấu hình trước là duy nhất cho mọi thiết bị của Mạng Zigbee. Nó chỉ được chia sẻ giữa Trust Center và Thiết bị Zigbee. Chỉ Centralized Security Network mới có thể sử dụng khóa này trong quá trình device commissioning để mã hóa phân phối Network Key. Khóa này cũng có thể được gọi là Khóa dẫn xuất mã cài đặt nếu nó được lấy từ mã được tạo ngẫu nhiên (được gọi là Mã cài đặt) được cài đặt sẵn vào mọi Thiết bị Zigbee.
- Application Link Key: Khóa duy nhất được tạo bởi Trust Center trong Centralized Security Network. Nó chỉ được chia sẻ giữa Trust Center và Thiết bị Zigbee cụ thể để che giấu các tin nhắn giữa chúng với các thiết bị khác trong mạng.
- Application Link Key: Trong mạng Zigbee, giao tiếp giữa các thiết bị liên quan có thể được mã hóa ở Application layer bằng phím Liên kết ứng dụng. Trong Centralized Security Network, trung tâm Trust tạo khóa này và phân phối cho các thiết bị liên quan. Trong Distributed Security Network, các thiết bị tạo và chia sẻ chìa khóa này

Zigbee có thể sử dụng bất kỳ loại khóa liên kết nào và để mã hóa việc chuyển Network Key và Khóa liên kết ứng dụng (Application Linker Key). Các khóa liên kết được sử dụng thêm để lấy một cặp khóa:

- key-load key được sử dụng để mã hóa các khóa liên kết của Trung tâm ứng dụng hoặc Trust Center và được lấy từ Khóa liên kết bằng cách sử dụng chức năng bấm khóa chuyên dụng.
- key-transport key được sử dụng để mã hóa Network Key và được lấy từ Khóa liên kết bằng cách sử dụng chức năng bấm khóa chuyên dụng

Một Unique Key có lợi thế là cung cấp các tin nhắn che giấu từ các thiết bị khác trong mạng nhưng kích thước của bộ lưu trữ Khóa liên kết duy nhất trong Trust Center tăng theo số lượng thiết bị trong mạng. Khóa Liên kết toàn cầu có một lợi thế là bộ nhớ mà Trust Center yêu cầu không tăng theo số lượng thiết bị trong mạng Zigbee.

### 2.3.7 Tạo lập mạng Zigbee

Mạng Zigbee có 2 kiến trúc: Centralized Security Network và Distributed Security Network. Centralized Network chỉ có thể được khởi tạo từ Coordinator, trong khi bất kỳ thiết bị Router nào cũng có thể tạo thành một Distributed Security Network [12].

Quy trình tạo lập mạng: Một thiết bị thiết lập một mạng mới bằng cách xác định các kênh cần quét, Thời lượng quét và loại mạng (Centralized Security Network hoặc Distributed Security Network) sẽ được hình thành. Trong quá trình hình thành mạng Zigbee, các bước sau được thực hiện [15]:

- Một kênh phù hợp, từ danh sách kênh được chỉ định, được chọn sau khi thực hiện quét năng lượng và quét tích cực.
- Một cách ngẫu nhiên, một ID PAN nhỏ hơn 0xFFFF chưa được sử dụng trong kênh đó được chọn bởi thiết bị.
- Short Address của thiết bị được chọn dựa trên loại mạng được hình thành
  - Short Address của Zigbee Coordinator, node tạo thành Centralized Security Network, luôn có địa chỉ mạng 0x0000.
  - Trong Distributed Security Network, một địa chỉ ngắn ngẫu nhiên được gán cho thiết bị tạo thành mạng.
- EPID, globally unique ID cho mạng LAN, thường là địa chỉ IEEE 64 bit của thiết bị tạo thành mạng. Tuy nhiên, các giá trị khác là có thể và không có cấu trúc cần thiết cho địa chỉ

### 2.3.7.1 Khởi tạo bảo mật Zigbee

Sau khi mạng được hình thành, thiết bị khởi động mạng có trách nhiệm cấu hình bảo mật của mạng. Bước này được thực hiện trước khi bất kỳ thiết bị mới nào được thêm vào mạng. Các thuộc tính sau được gán một giá trị trong bước này:

- Security Level: Có 7 cấp độ bảo mật trong mạng Zigbee. Cấp độ bảo mật bằng 0, tức là, các tin nhắn được gửi qua mạng Zigbee sẽ không được bảo mật. Cấp độ bảo mật bằng 7 là cao nhất.
- Network Key: Khóa được sử dụng để bảo mật tin nhắn ở lớp mạng được tạo ngẫu nhiên bởi thiết bị tạo thành mạng. Nó là bất kỳ số ngẫu nhiên khác không.

- Trust Center Address: Trust Center cung cấp các dịch vụ bảo mật trong Centralized Security Network. Địa chỉ của Trust Center được đặt thành địa chỉ của Zigbee Coordinator. Trong trường hợp Distributed Security Network, địa chỉ Trust Center không được áp dụng.

### 2.3.8 *Thư viện Zigbee Cluster*

Trong Zigbee Cluster Library (ZCL), cụm (cluster) là một tập hợp các thông điệp được sử dụng để gửi và nhận các lệnh và dữ liệu liên quan qua Zigbee [16]. Ví dụ: cụm nhiệt độ sẽ chứa tất cả các thông báo OTA cần thiết để gửi và nhận thông tin về nhiệt độ. Để tạo điều kiện cho việc quản lý, các cụm này được nhóm lại theo chức năng, chẳng hạn như chức năng bảo mật, chiếu sáng, v.v. Các nhà phát triển cũng có thể xác định các cụm riêng của họ Lớp ứng dụng của Zigbee sau đó tham chiếu cụm nào được sử dụng trong các ứng dụng nhất định.

#### 2.3.8.1 Mô hình client-server

Mỗi cụm được chia thành hai đầu, một đầu client và một đầu server. Client và server có thể trao đổi thông tin theo 2 chiều. Trái ngược với nhiều hệ thống khác (ví dụ: HTTP), cả hai đều có cùng tiềm năng gửi và nhận tin nhắn. Ví dụ: Mô hình client-server giữa bóng đèn và công tắc. Bóng đèn đóng vai trò server và nhận lệnh điều khiển từ công tắc. Ngược lại, công tắc đóng vai trò client và gửi lệnh điều khiển đến bóng đèn [17].

Bảng 11. Mô hình client-server giữa bóng đèn và công tắc

<b>On/off cluster</b>	<b>Server (Light Bulb)</b>	<b>Client (Switch)</b>
Attributes	On/off	None
Received commands	ON OFF TOGGLE	None
Generated commands	None	ON OFF TOGGLE

#### 2.3.8.2 Đặc tính (Attribute)

Một thuộc tính là dữ liệu được liên kết hai đầu cụm. Mỗi cụm có thể có nhiều thuộc tính khác nhau. Mỗi thuộc tính khai báo một mã định danh 16 bit, kiểu dữ liệu, chỉ định đọc hoặc đọc / ghi, giá trị mặc định và chỉ thông báo.

### 2.3.8.3 Lệnh (Command)

Một lệnh bao gồm một định danh lệnh (8 bit) và payload. Giống như các thuộc tính, định danh là duy nhất trong một cụm. Các lệnh được chia thành hai loại [17]:

- Global command: Các lệnh toàn cầu được xác định trong ZCL và thực hiện được trên tất cả các cụm. Ví dụ:

Bảng 12. Ví dụ một số lệnh toàn cầu

Command	Response
Read attribute	Read attribute response
Write attribute	Write attribute response
Config reporting	Config reporting response
...	...

- Cluster-specific command: Các lệnh dành riêng cho cụm được xác định bên trong các định nghĩa cụm và là duy nhất cho cụm mà chúng được xác định. Ví dụ: Lệnh mô tả ở **phản mô hình client-server** (bao gồm ON, OFF, TOGGLE) là các cluster-specific commands vì các lệnh này chỉ hoạt động trong On/off cluster.

## 2.4 Giao thức MQTT

### 2.4.1 Tổng quan về giao thức MQTT

MQTT (Message Query Telemetry Transport) là một giao thức publish/subscribe cung cấp kết nối giữa các thiết bị trong môi trường mạng công suất thấp. Giao thức MQTT có hai đặc điểm nổi bật [18]:

- Mô hình publish/subscribe hoạt động bất đồng bộ, nghĩa là mô hình này tách rời 2 phía publish và subscribe. Publisher có thể gửi gói tin đi mà không cần quan tâm đến kết nối của Subscriber.
- Giao thức MQTT lọc các gói tin theo topic (chủ đề) của chúng (subject-base filtering). Chủ đề bao gồm một hoặc nhiều cấp chủ đề. Mỗi cấp độ chủ đề được phân tách bằng dấu '/' (dấu phân cách cấp độ chủ đề). Ngoài ra, MQTT còn cung cấp theo các ký tự đại diện (wildcards) để client có thể subscribe nhiều topic. Dấu '+' là ký tự đại diện mức độ đơn (single level wildcard). Dấu '#' là ký tự đại diện mức độ đa (multiple level wildcard).

Ví dụ:

- myhome/livingroom/temperature: Đăng ký chỉ chủ đề con /temperature trong chủ đề /livingroom
- myhome/+/lighting: Đăng ký tất cả topic về lighting trong nhà
- myhome/livingroom/#: Đăng ký tất cả topic con của livingroom, có thể bao gồm nhiệt độ, độ ẩm, trạng thái khóa cửa, ...

### 2.4.2 Các node trong mô hình mạng MQTT

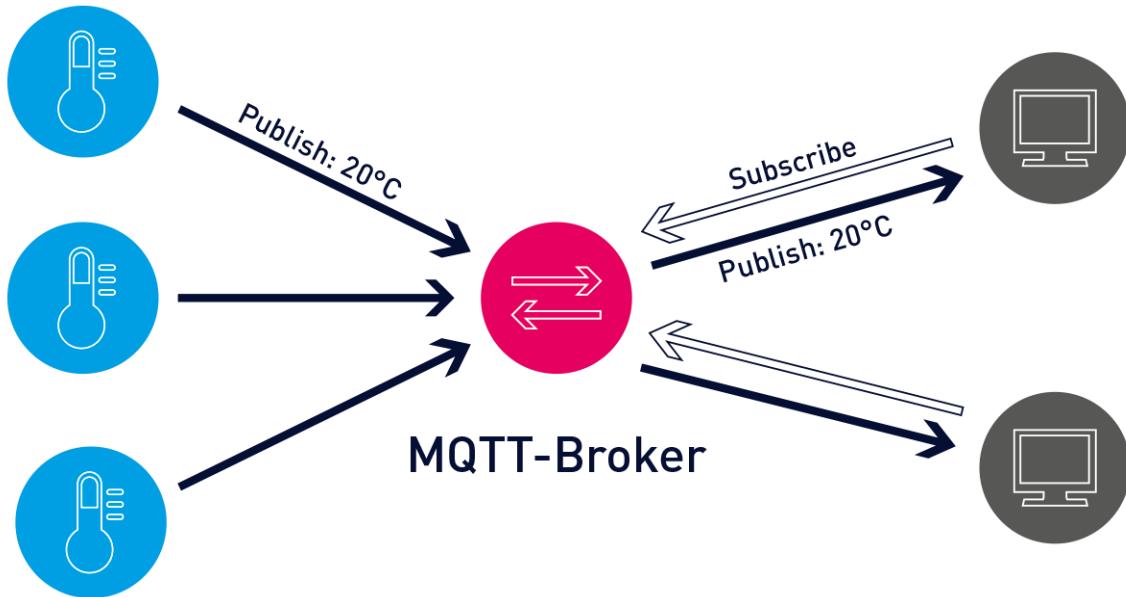
MQTT client có thể là thiết bị hiển thị thông tin (điện thoại thông minh, trung tâm giao tiếp gia đình), cảm biến (nhiệt độ, độ ẩm, v.v.) và các thiết bị có thể điều khiển (bóng đèn, khóa cửa, v.v.). MQTT client thực hiện subscribe đến topic để truyền và nhận các gói tin.

MQTT server hoặc MQTT broker. Thiết bị này thực hiện việc lọc các thông điệp và chuyển giao các thông điệp đã được gửi tới các Subscriber quan tâm chúng. Broker quyết định subscriber nào sẽ nhận được thông điệp gì dựa vào việc lọc các gói tin theo topic (chủ đề).

### 2.4.3 Cơ chế hoạt động

MQTT hoạt động theo cơ chế client/server, nơi mà mỗi cảm biến là một khách hàng (client) và kết nối đến một server, có thể hiểu như một server môi giới (broker), thông qua

giao thức TCP (Transmission Control Protocol). Broker chịu trách nhiệm điều phối tất cả các thông điệp giữa phía gửi đến đúng phía nhận.



Hình 29. Mô hình kết nối MQTT

### 2.4.3.1 Khởi tạo kết nối

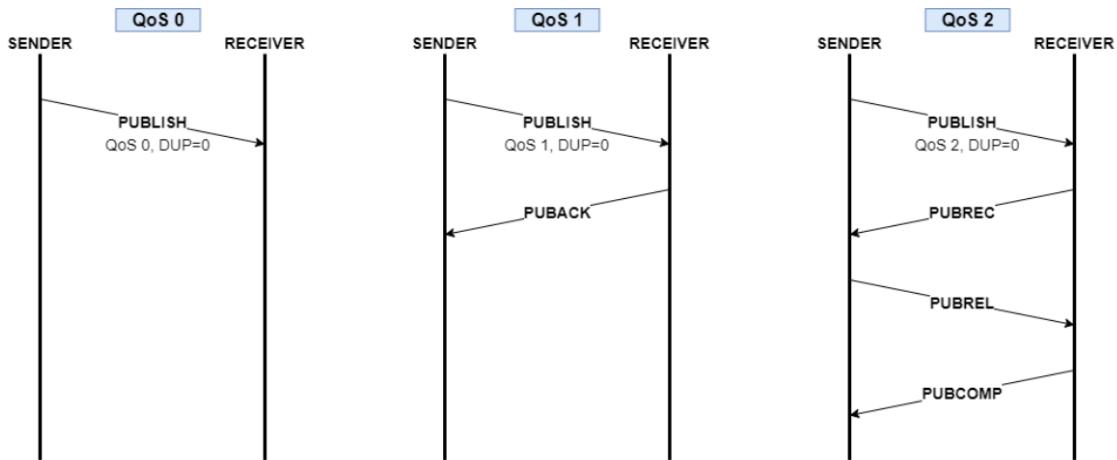
Client gửi gói tin yêu cầu kết nối (gói tin CONNECT). Khi MQTT - Broker nhận được gói tin sẽ trả lời bằng một gói tin CONNACK (Connecting Acknowledgement). Gói tin này mang một mã trả về (result code) thông báo kết quả của quá trình kết nối. Nếu kết nối thành công, client có thể publish gói tin lên topic xác định hoặc subscribe các topic có định nghĩa trên Broker.

### 2.4.3.2 Gửi gói tin đến topic xác định

Nếu client ở vai trò publisher, nó sẽ gửi một gói tin PUBLISH đến broker. Gói này chứa thông tin chi tiết về chất lượng dịch vụ QoS (Quality of Service), tên topic, payload, v.v. MQTT hỗ trợ ba mức QoS cho client:

- QoS = 0: Broker không gửi gói tin xác nhận ACK
- QoS = 1: Broker gửi lại gói tin xác nhận PUBACK (Publish acknowledgement)

- QoS = 2: Broker xác nhận gói tin PUBLISH với gói tin PUBREC (Publish received). Client tiếp tục gửi gói PUBREL (Publish release) và broker xác nhận lại bằng PUBCOMP (Publish complete).

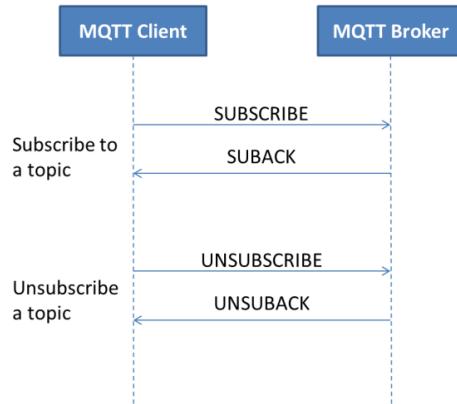


Hình 30. Minh họa sự khác biệt ở các mức QoS khác nhau

Ta nhận thấy QoS càng lớn, mức độ xác thực của gói tin càng cao. Tuy nhiên sẽ tồn tại thông tin cho việc gửi các gói tin xác nhận.

### 2.4.3.3 Đăng ký topic

Nếu client muốn đăng ký nhận gói tin của một chủ đề nào đó, nó sẽ gửi gói SUBSCRIBE mang theo tên chủ đề (mã UTF-8). Broker xác nhận đăng ký với gói SUBACK (Subscriber Acknowledgement) cùng với trạng thái của yêu cầu. Để hủy đăng ký một chủ đề, khách hàng sẽ gửi gói UNSUBSCRIBE đến broker. Broker xác nhận bằng gói UNSUBACK (Unsubscribe Acknowledgement)



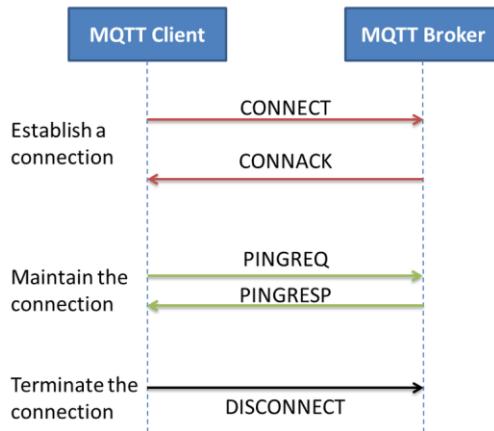
Hình 31. Quá trình client đăng ký và hủy đăng ký

### 2.4.3.4 Duy trì kết nối

Sau một khoảng thời gian chờ nhất định, kết nối giữa broker và client sẽ kết thúc. Để duy trì kết nối, client gửi gói tin PINGREQ đến broker. Broker phản hồi lại client với gói PINGRESP và duy trì kết nối tồn tại.

### 2.4.3.5 Kết thúc kết nối

Chấm dứt kết nối, Cleint gửi một gói DISCONNECT đến server. server sẽ không gửi lại gói xác nhận. Tuy nhiên tất cả các gói tin liên quan đến client đều bị xóa và client sẽ ngắt kết nối đến server.



Hình 32. Các gói tin trao đổi để duy trì, kết thúc kết nối MQTT

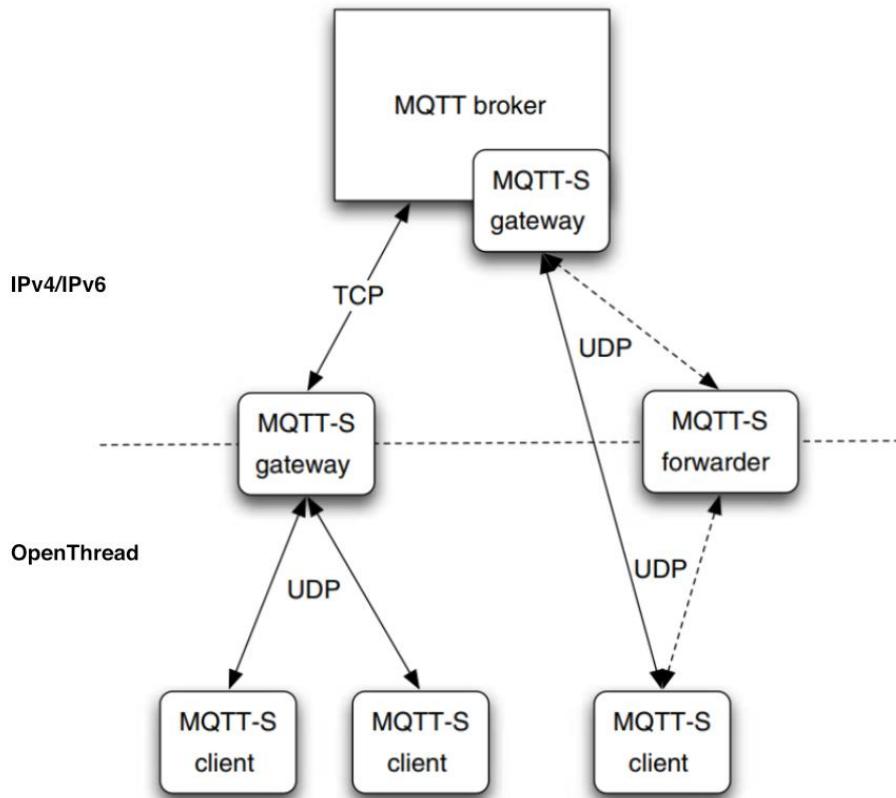
## 2.4.4 *MQTT-SN*

MQTT-SN (Message Query Telemetry Transport for Sensor Nodes) hướng đến các thiết bị nhúng trong môi trường công suất thấp. MQTT-SN được thiết kế cho tiêu chuẩn IEEE 802.15.4 với tốc độ thấp chỉ 250 kbps tại tần số 2.4 GHz, độ dài tối đa của gói tin là 127 bytes (ở lớp vật lý) và chỉ còn 88 bytes (ở lớp MAC) [19]

Ưu điểm của MQTT-SN so với MQTT:

- Gói tin CONNECT để thiết lập kết nối được chia nhỏ thành 3 gói tin. Hai gói tin thêm để truyền cho Will topic và Will message.
- Cung cấp cơ chế tìm kiếm broker xung quanh. Nếu có nhiều broker trong một mạng, chúng sẽ hoạt động ở chế độ chia tải (load-sharing mode)
- Clean session mở rộng với đặc tính Will, không chỉ có các topic được client subscribe cần duy trì, mà còn có Will topic và Will message. Client có thể thay đổi Will topic và Will message của nó trong suốt một session.
- Cung cấp cơ chế Offline Keep-Alive mới để hỗ trợ cho các sleeping clients tiết kiệm năng lượng. Các thiết bị hoạt động bằng pin có thể vào chế độ sleep, trong lúc đó, các gói tin hướng tới chúng sẽ được đệm (buffer) trên broker và chuyển đi khi chúng thức dậy.
- Trường tên topic trong gói tin PUBLISH sẽ được thay thế bằng topic id (2 bytes). Cung cấp thủ tục đăng ký topic cho client (Client đăng ký topic với broker, sau đó nhận lại topic id từ broker)
- Cho phép đặt trước topic id tương ứng với tên topic, do đó không cần thực hiện quá trình đăng ký. Ở trường hợp này, cả client và broker cần phải biết trước topic id và tên topic để giao tiếp với nhau. Đặc biệt, nếu tên topic ngắn (chỉ trong 2 bytes), client có thể sử dụng topic id giao tiếp với broker mà không cần thủ tục đăng ký
- Giảm tiêu thụ năng lượng vì sử dụng UDP để truyền tin thay cho TCP/IP

Trong mô hình mạng thực tế, MQTT-SN sẽ được áp dụng với các sensor để duy trì năng lượng và giảm công suất, đồng thời tích hợp MQTT-SN gateway vào Border Router để điều hướng các gói tin đến MQTT Broker qua mạng Internet.



Hình 33. Mô hình mạng của giao thức MQTT-SN

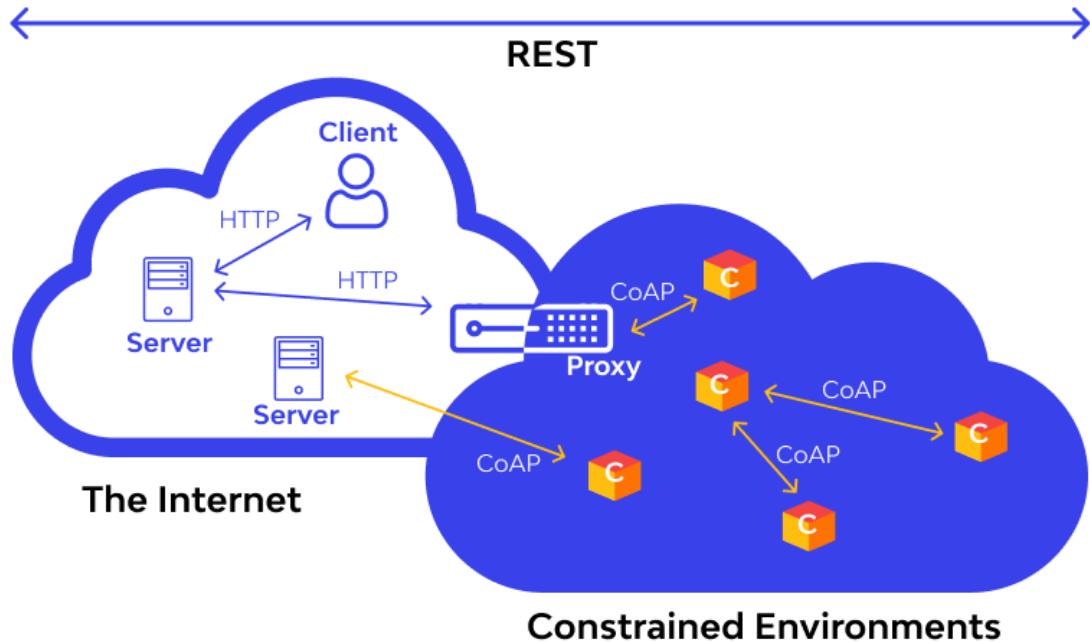
Kiến trúc mạng sử dụng giao thức MQTT-SN bao gồm 4 thành phần: MQTT broker, MQTT-SN gateway, MQTT-SN forwarder và MQTT-SN client. Client sẽ kết nối với broker thông qua gateway. Gateway thường được tích hợp trên broker hoặc cài đặt trong Border Router. Forwarder có thể không xuất hiện trong mạng vì client có thể gửi trực tiếp qua UDP đến gateway [20]:

- Gateway: Chuyển đổi giữa MQTT-SN sang MQTT.
- Forwarder: Định tuyến giữa các client và gateway, có thể đi qua nhiều đường dẫn và nhảy qua một số bộ định tuyến trên đường đi.
- Client và broker hoạt động tương tự MQTT

## 2.5 Giao thức CoAP

CoAP là một giao thức truyền tin theo mô hình client/server dự trên internet (tương tự như giao thức HTTP) nhưng được thiết kế cho các thiết bị công suất thấp [21]. Giao thức

này giao tiếp peer-to-peer giữa client và server. CoAP sử dụng UDP (User Datagram Protocol), không hỗ trợ TCP, ngoài ra còn hỗ trợ địa chỉ broadcast và multicast, truyền thông CoAP thông qua các datagram phi kết nối (connectionless) có thể được sử dụng trên các giao thức truyền thông dựa trên các gói. CoAP theo mô hình client/server. Client gửi yêu cầu đến server, sau đó server gửi lại phản hồi. Client có thể GET, PUT, POST và DELETE các tài nguyên [22].



Hình 34. Môi trường hoạt động của giao thức CoAP

### 2.5.1.1 Mô hình gói tin CoAP

Vì UDP là giao thức truyền tin không tin cậy, CoAP áp dụng một vài cơ chế ở lớp ứng dụng để tăng độ tin cậy cho quá trình truyền. Có 4 loại gói tin:

- Confirmable (CON): Gói tin CON yêu cầu xác nhận (ACK) và gói tin phản hồi có thể được gửi trong cùng ACK (đồng bộ) hoặc trong riêng biệt (không đồng bộ).
- Non-confirmable (NON): Gói tin NON không cần ACK
- Acknowledgement (ACK): được gửi để phản hồi lại các thông điệp CON, để xác nhận đã nhận được gói tin
- Reset (RST): phản hồi của một gói tin không thể xử lý.

### 2.5.1.2 So sánh MQTT và CoAP

## CƠ SỞ LÝ THUYẾT

Bảng 13. So sánh CoAP và MQTT

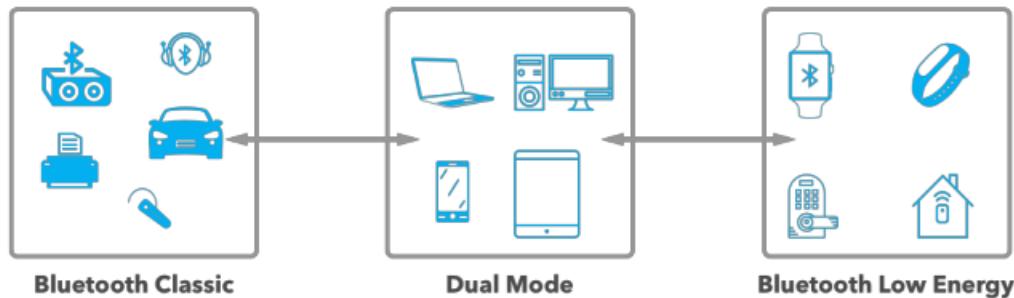
MQTT	CoAP
Mô hình subscribe/publish	Mô hình request/response
Broker điều hướng gói tin đến client tương ứng	Gói tin chỉ được truyền unicast giữa 2 thiết bị. Tương tự với HTTP
Chạy trên nền TCP	Sử dụng gói tin UDP để truyền tin và giao tiếp
Phân loại nhóm gói tin cho những mục đích khác nhau	Định nghĩa các gói tin để dễ dàng tìm kiếm

## 2.6 Giao thức Bluetooth Low Energy

### 2.6.1 Khái niệm

Bluetooth là một chuẩn giao tiếp không dây sử dụng trong việc truyền dữ liệu ở khoảng cách gần. Những ứng dụng dễ thấy của Bluetooth là bàn phím hoặc chuột không dây. Ngày nay, Bluetooth gần như có mặt ở mọi nơi, đặc biệt là đối với những thiết bị cần giao tiếp qua kết nối không dây.

Thiết bị Bluetooth được chia làm hai loại: Bluetooth Classic (BR/EDR) và Bluetooth Low Energy (BLE). Bluetooth Classic (BR/EDR) được sử dụng trong loa không dây, xe hơi hoặc tai nghe, những ứng dụng này cần tốc độ truyền dữ liệu lớn và liên tục. Bluetooth Low Energy (BLE) dành cho những ứng dụng đòi hỏi tiết kiệm năng lượng hoặc có nguồn năng lượng hạn chế như các thiết bị sử dụng pin hay các node cảm biến. Hai loại Bluetooth này không tương thích với nhau mà gần như là khác nhau. Một thiết bị Bluetooth Classic không thể giao tiếp trực tiếp với thiết bị BLE. Do đó, đối với các thiết bị như smartphone, người ta thường triển khai cả hai loại giao thức (Dual Mode Bluetooth devices), để cho phép chúng giao tiếp với cả hai loại Bluetooth [23].



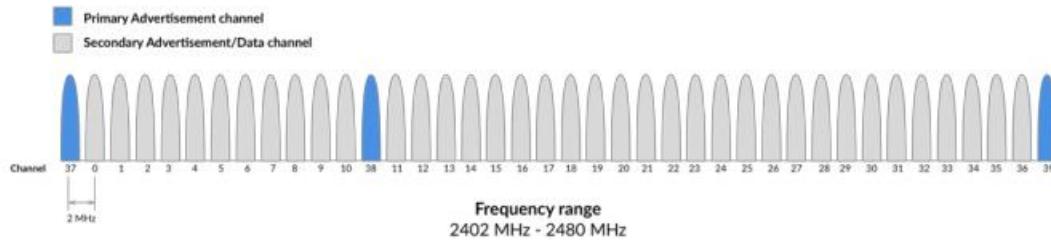
Hình 35. Phân loại thiết bị Bluetooth

Trong đề tài này, nhóm chỉ tập trung tìm hiểu về Bluetooth Low Energy (BLE) phục vụ cho các ứng dụng Internet vạn vật (Internet of things – IoT). Những hệ thống Internet of things (IoT) phát triển đòi hỏi nhiều hơn về số lượng thiết bị và cảm biến, những node cảm biến thường có nguồn năng lượng dữ trữ vô cùng hạn chế, chính vì thế, BLE – một giao thức tiết kiệm năng lượng, trở nên vô cùng phổ biến.

### 2.6.2 Đặc điểm

Một số thông số quan trọng về BLE:

- Phổ tần số hoạt động: 2.400 – 2.4835 GHz
- Phổ tần số này được chia thành 40 khe tần số 2MHz
- Data rate tối đa có thể đạt được (đối với Bluetooth version 5) là 2Mbps
- Khoảng cách truyền dao động tùy thuộc vào môi trường: 10-30m
- Công suất trung bình tiêu thụ cũng dao động tùy thuộc vào thông số BLE và chip sử dụng. Dòng điện tiêu thụ tối đa thường dưới 15mA.
- BLE có thể được bổ sung thêm các yếu tố bảo mật tùy thuộc vào nhu cầu của nhà phát triển
- BLE sử dụng mã hóa mã AES CCM 128 bit
- BLE được thiết kế cho các ứng dụng truyền dữ liệu băng hẹp. Sử dụng BLE cho các ứng dụng băng thông rộng có thể gây tiêu hao nhiều năng lượng.



Hình 36. Phổ tần số

### 2.6.3 Ưu điểm và hạn chế của BLE

#### 2.6.3.1 Hạn chế

##### *Throughput – Thông lượng*

Đối với những phiên bản Bluetooth 4.2 trở về trước, tốc độ truyền dữ liệu có định 1Mbps. Tuy nhiên, đối với phiên bản Bluetooth 5 hoặc mới hơn, tốc độ truyền phụ thuộc vào lớp vật lý hoặc chế độ hoạt động; tốc độ có thể là 1Mbps giống với các phiên bản trước hoặc 2Mbps nếu sử dụng tính năng truyền tốc độ cao. Nhưng đối với tính năng truyền khoảng cách xa, tốc độ giảm còn 500 hoặc 125Kbps [24].

Một số yếu tố ảnh hưởng đến tốc độ truyền dữ liệu:

- Gaps in between packets: Bluetooth quy định mỗi gói tin truyền đi phải cách nhau 150us. Đây được coi là khoảng thời gian hao phí vì không có data nào được truyền đi trong khoảng thời gian này.
- Packet overhead: Tất cả packet đều có vùng header chứa các thông tin của gói tin, nhưng không phải là data.
- Slave data packets requirement: Slave được yêu cầu gửi các gói tin rỗng kể cả khi không có dữ liệu cần trao đổi
- Retransmission of data packets: Trong trường hợp mất gói hoặc nhiều từ các thiết bị khác, những gói tin bị mất cần được truyền lại.

##### *Phạm vi truyền*

BLE được thiết kế cho các ứng dụng truyền khoảng cách ngắn, do đó phạm vi truyền của BLE rất hạn chế. Một số yếu tố ảnh hưởng đến phạm vi truyền của BLE:

- BLE hoạt động ở băng tần 2.4 GHz ISM nên rất dễ bị ảnh hưởng bởi các vật cản xung quanh như các vật dụng kim loại, tường hoặc nước.
- Thiết kế của anten cũng ảnh hưởng đến thiết bị BLE

### ***Đòi hỏi gateway trong trường hợp kết nối internet***

Để có thể truyền dữ liệu từ một thiết bị chỉ chạy BLE đến mạng Internet, cần một thiết bị BLE khác có kết nối IP để chuyển tiếp dữ liệu này đến mạng internet.

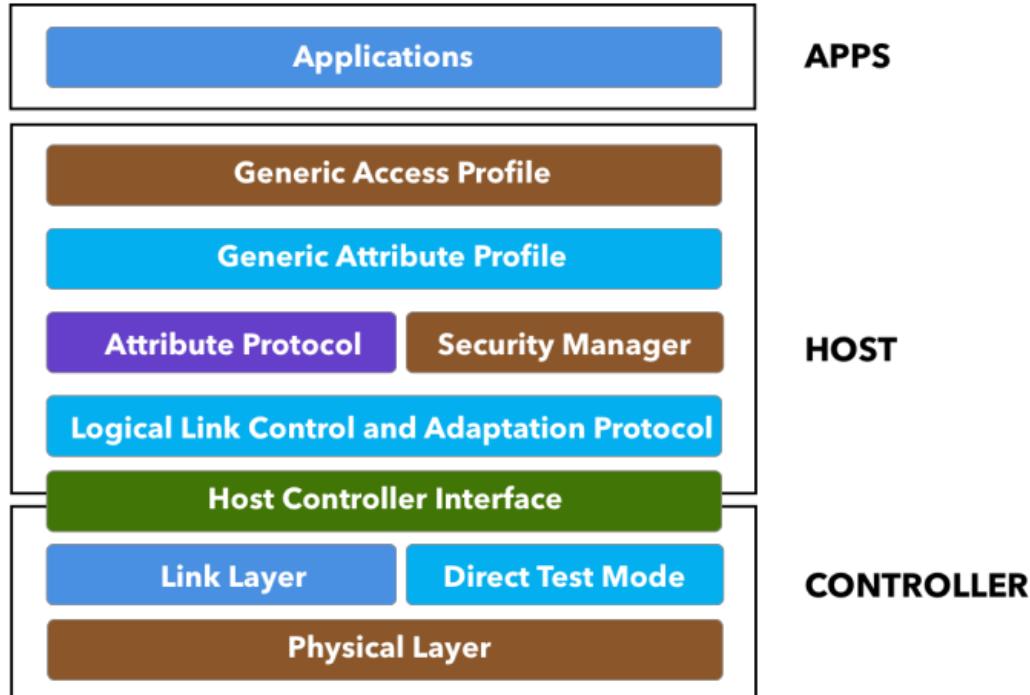
#### **2.6.3.2 Ưu điểm của BLE**

Mặc dù có một số hạn chế, BLE vẫn có những ưu điểm vượt trội và phù hợp với các hệ thống IoT. Một số ưu điểm bao gồm [24]:

- Công suất tiêu thụ thấp: BLE tiêu thụ công suất thấp hơn so với các đối thủ cạnh tranh. BLE đã được tối ưu hóa bằng cách tắt phát sóng Bluetooth và chỉ gửi một lượng nhỏ data với tốc độ truyền thấp
- BLE có sẵn trong hầu hết các smartphones trên thị trường. Đây là ưu điểm lớn nhất của BLE so với các đối thủ khác như ZigBee, Z-Wave và Thread.

#### ***2.6.4 Kiến trúc của BLE***

Bluetooth Low Energy (BLE) được chia thành các layers khác nhau, cụ thể, có ba khối chính: Application, Host và Controller [23].



Hình 37. Kiến trúc BLE

Có hai loại kết nối BLE: thiết bị ché độ kép và ché độ đơn thiết bị. Ché độ kép bao gồm Bluetooth cổ điển (Classical Bluetooth) và BLE. Ché độ kết nối đơn thiết bị chỉ hỗ trợ BLE. Các thiết bị ché độ đơn có thể giao tiếp với nhau và các thiết bị ché độ kép thông qua BLE, nhưng không phải với các thiết bị Bluetooth cổ điển.

- Application: tầng ứng dụng cung cấp giao diện người dùng
- Host: bao gồm các tầng sau
  - Generic Access Profile (GAP)
  - Generic Attribute Profile (GATT)
  - Attribute Protocol (ATT)
  - Security Manager (SM)
  - Logical Link Control and Adaptation Protocol (L2CAP)
  - Host Controller Interface (HCI) – Host side
- Controller: bao gồm các tầng sau
  - Physical layer (PHY)
  - Link layer
  - Direct Test Mode

- Host Controller Interface (HCI) – Controller side

### 2.6.4.1 Lớp vật lý

Lớp vật lý (PHY) là lớp thấp nhất trong kiến trúc. Nó cấu hình các thông số vật lý cho giao thức BLE để truyền tin qua môi trường truyền. BLE hoạt động trên băng tần 2.4GHz. Giao thức này chia phổ tần số thành 40 kênh (3 kênh dùng cho advertising và 37 kênh truyền dữ liệu) thay vì 79 kênh trong Bluetooth cổ điển. Mỗi kênh đều có băng thông 2 MHz. BLE sử dụng các kênh advertising chính để khám phá thiết bị, hình thành két nối và phát sóng. Trong khi đó, các kênh truyền dữ liệu hướng đến giao tiếp hai chiều giữa các thiết bị trong một mạng và chúng còn được gọi là kênh advertising thứ cấp.

Tốc độ truyền của BLE là khoảng 1 Mbps, với 1 bit/symbol. BLE là 1 giao thức được tối ưu hóa cho dữ liệu nhỏ và truyền tải nhanh chóng. Kỹ thuật điều chế BLE là Gaussian Frequency-Shift Keying (GFSK), nghĩa là các xung phải đi qua bộ lọc Gaussian trước khi thay đổi thành tần số sóng mang.

### 2.6.4.2 Lớp Liên kết

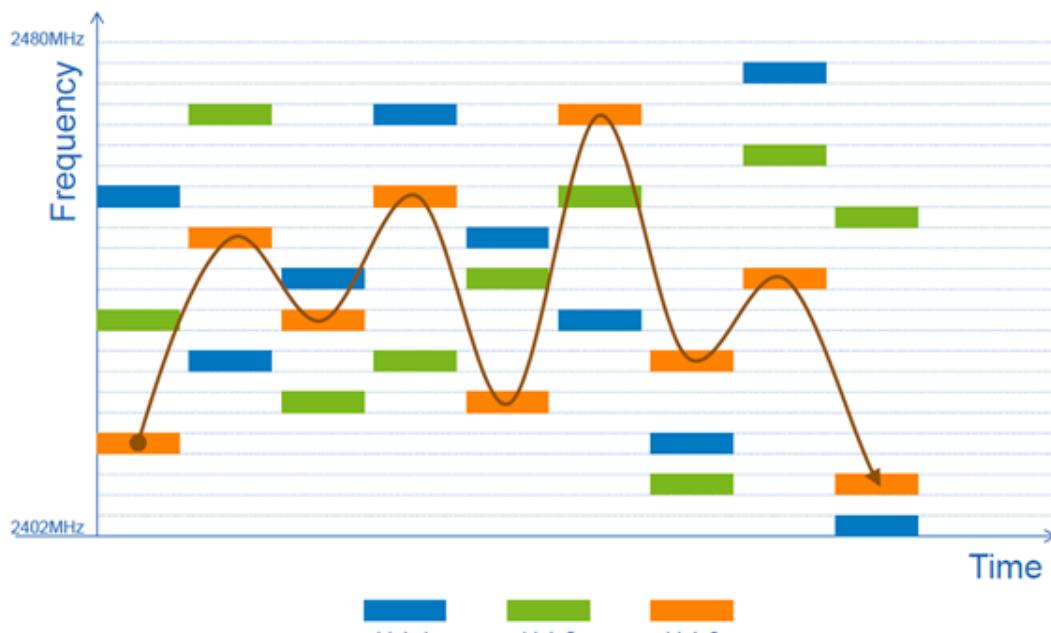
#### **Frequency hopping**

BLE sử dụng giải thuật Adaptive frequency hopping algorithm (AFH) xoay vòng trong 37 kênh truyền dữ liệu theo công thức:

$$fn + 1 = (fn + hop) \bmod 37$$

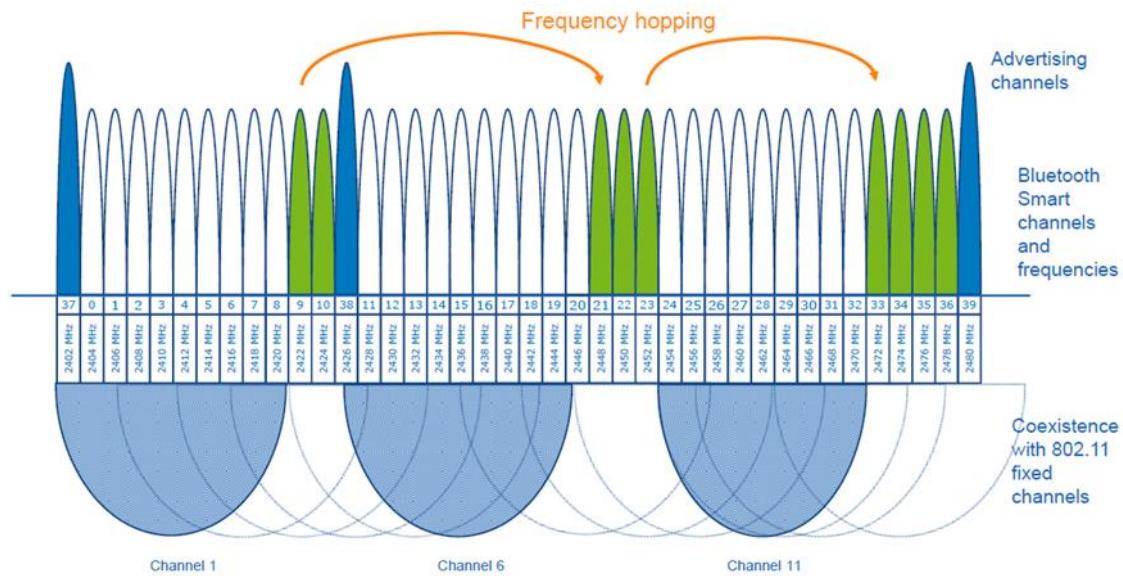
- Với  $fn + 1$  là kênh truyền sử dụng cho kết nối tiếp theo
- Hop là giá trị đặt (khoảng 5 đến 16) khi thiết bị khởi tạo kết nối.

Biểu đồ mô tả 3 kết nối BLE đồng thời và các bước nhảy tầng số tương ứng



Hình 38. Mô tả Frequency Hopping

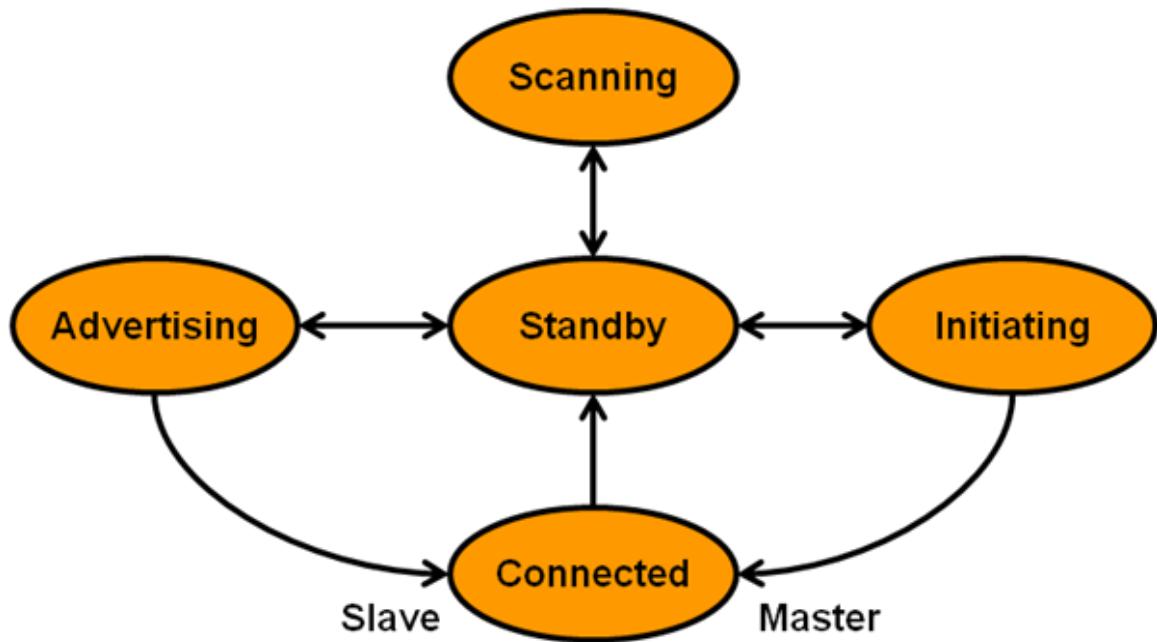
Cơ chế này cũng giúp lớp Liên kết remap một gói tin từ một kênh nhiễu thành một kênh truyền rồi để giảm nhiễu từ các thiết bị khác. Ví dụ: một thiết bị BLE hoạt động trong cùng một khu vực có nhiều mạng Wi-Fi trên các kênh 1, 6 và 11. Thiết bị BLE sẽ đánh dấu các kênh 0-8, 11-20 và 24-32 là các kênh nhiễu nhiễu. Khi hai thiết bị giao tiếp, chúng sẽ chuyển đổi qua lại và ánh xạ lại các kênh này.



Hình 39. Mô tả Channel Hopping

#### 2.6.4.3 BLE state machine

State machine mô tả hoạt động thiết yếu của lớp Liên kết, lớp này bao gồm năm trạng thái khác nhau. Sau đây là mô tả ngắn gọn về từng trạng thái và cách nó thay đổi giữa năm trạng thái.



Hình 40. Bluetooth state machine

Bảng 14. Mô tả các state machine của Bluetooth Low Energy

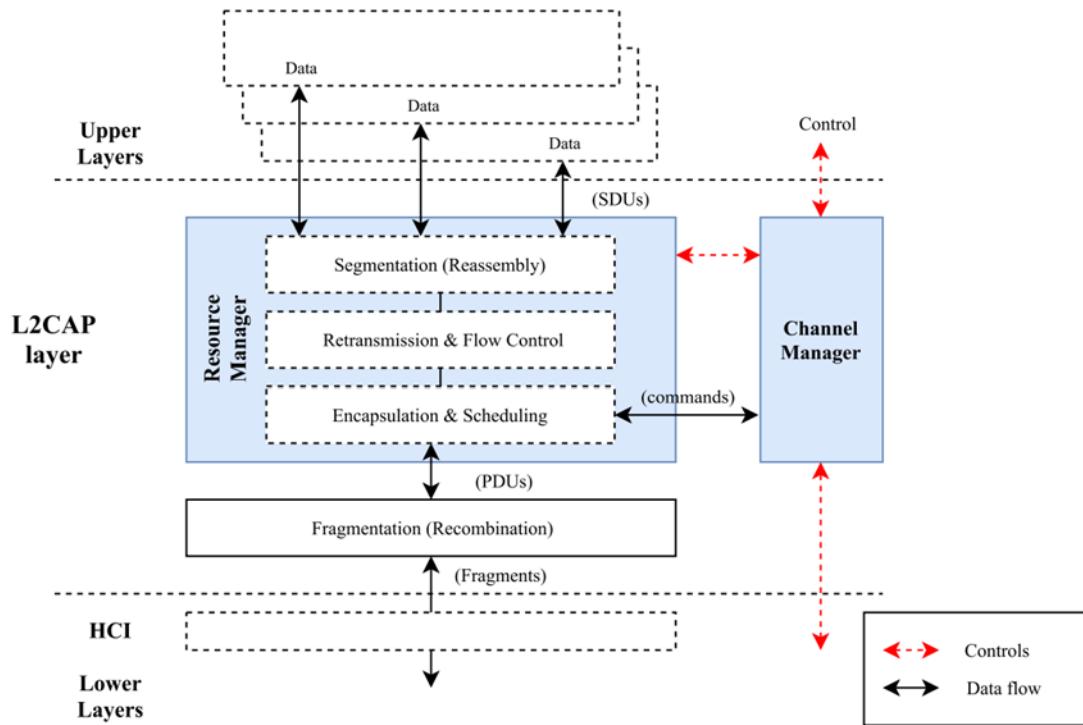
	Mô tả
Trạng thái chờ (Standby state)	Nó là trạng thái mặc định của lớp liên kết. Khi lớp liên kết không nhận được bất kỳ gói tin nào, nó sẽ chuyển sang trạng thái này, trong đó không có gói nào được gửi hoặc nhận.
Trạng thái Advertising (Advertiser)	Lớp liên kết không chỉ truyền các gói advertising ở trạng thái này, mà nó còn có thể lắng nghe phản hồi đối với các gói advertising. Một thiết bị muốn tham gia mạng BLE sẽ vào trạng thái này
Trạng thái tìm kiếm (Scanner)	Thiết bị ở trạng thái này khi nó lắng nghe các gói advertising từ thiết bị khác và phản hồi bằng cách yêu cầu thông tin bổ sung.
Trạng thái khởi tạo (Initiating)	Thiết bị ở trạng thái này khi nó muốn lắng nghe các gói tin advertising từ một thiết bị khác yêu cầu thiết lập kết nối. Có hai vai trò trong trạng thái này: Master và Slave. Master đi vào từ trạng thái khởi tạo, trong khi Slave đi vào từ trạng thái advertising

#### 2.6.4.4 HCI

Host-Controller Interface (HCI) nhằm quản lý giao tiếp giữa khối Controller và khối Host. Nó thực hiện một cơ chế tiêu chuẩn hóa giữa lớp trên và lớp dưới của kiến trúc BLE. Có bốn giao thức giữa khối Host và khối Controller: USB, Secure Digital (SD), UART và UART three-wire.

#### 2.6.4.5 L2CAP

Logical Link Control and Adaptation Protocol (L2CAP) truyền dữ liệu giữa các cấu hình (profiles) nằm ở các lớp trên của kiến trúc. L2CAP nhằm mục đích đảm bảo Chất lượng Dịch vụ (QoS), định tuyến, phân mảnh và phân đoạn cũng như tập hợp lại các gói từ các lớp cao hơn. Bên cạnh đó, nó cũng phân đoạn và tập hợp lại các gói lớn hơn như truyền các gói IPv6 qua BLE. Hình bên dưới minh họa cấu trúc bên trong **L2CAP**.



Hình 41. Mô hình kết nối giữa các thành phần trong lớp L2CAP

Trình quản lý kênh (Channel Manager) quản lý tất cả các tín hiệu nội bộ và cung cấp chức năng điều khiển. Trình quản lý tài nguyên (Resource Manager) bao gồm ba chức năng chính: phân đoạn và lắp ráp lại, truyền lại và kiểm soát luồng, đóng gói và sắp xếp truyền tin. Khôi điều khiển luồng (flow control) và truyền lại (retransmission) cung cấp khả năng điều khiển luồng trên mỗi kênh và khôi phục lỗi bằng cách sử dụng truyền lại gói.

#### 2.6.4.6 Security Manager (Trình bảo mật)

Trình quản lý bảo mật (SM) mã hóa và giải mã các gói dữ liệu, xác định các lược đồ xác thực phân tích cú pháp (quá trình có găng tin tưởng một thiết bị khác thông qua xác thực). Bên cạnh đó, SM cũng cung cấp một hộp công cụ bảo mật để tạo hash, giá trị xác nhận và tạo khóa ngắn hạn trong quá trình ghép nối. Quá trình ghép nối tạo thành một kết nối đáng tin cậy với mã hóa. Có ba bước chính:

1. Trao đổi tính năng Ghép nối. (bắt buộc)
2. Xác thực và Mã hóa. (bắt buộc)
3. Phân phối khóa cụ thể (tùy chọn)

### 2.6.4.7 ATT

Giao thức Thuộc tính (ATT) thiết lập giao tiếp giữa các thiết bị bằng cách sử dụng cấu trúc client - server. Một thuộc tính có ba yếu tố được liên kết với nó:

- Phân loại: hiển thị những gì thuộc tính mô tả.
- Cách xử lý: chức năng như một mã định danh cho client khi giao tiếp với server
- Quyền: thiết lập mức khả năng truy cập như quyền đọc / ghi.

ATT mô tả sáu loại thông điệp, bao gồm:

1. Yêu cầu từ client đến server.
2. Phản hồi từ server đến client để trả lời một yêu cầu.
3. Các lệnh từ client đến server không nhận được bất kỳ phản hồi nào.
4. Thông báo từ server đến client mà không nhận được bất kỳ xác nhận nào.
5. Các chỉ định từ server đến client.
6. Xác nhận từ client đến server để trả lời một chỉ báo.

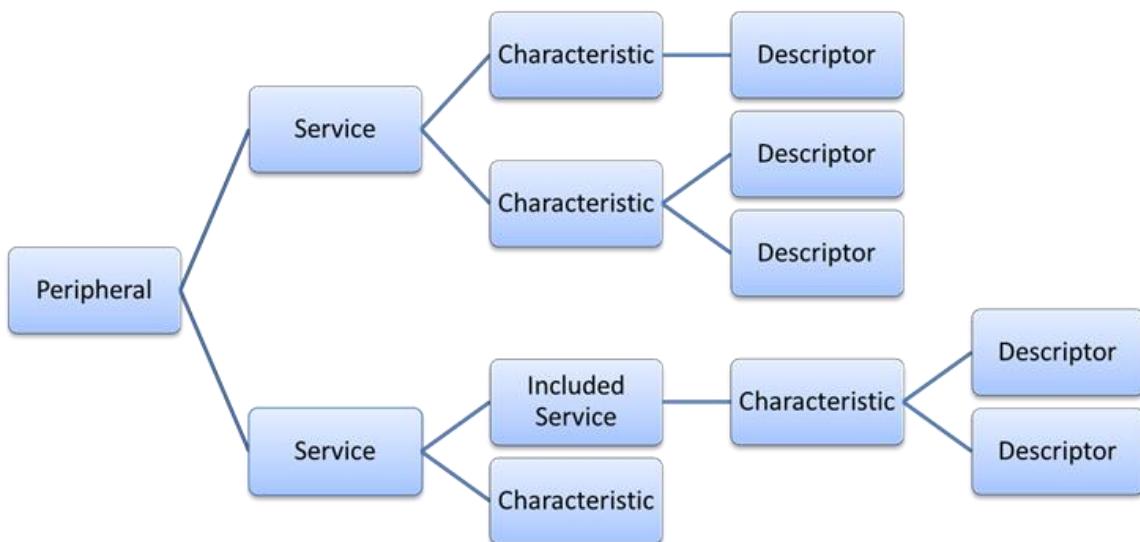
GATT sử dụng ATT để mô tả hệ thống phân cấp của các dịch vụ và đặc điểm, sẽ được trình bày chi tiết trong phần tiếp theo.

### 2.6.4.8 GATT

GATT thu thập các thuộc tính thành cấu trúc để quản lý thay vì danh sách các thuộc tính. GATT là một thông số kỹ thuật chung để gửi và nhận các phần dữ liệu ngắn (thuộc tính) qua một liên kết năng lượng thấp. GATT có thuật ngữ sau.

- Client: client là một thiết bị khởi tạo các lệnh và yêu cầu GATT. Nó cũng chấp nhận phản hồi. Ví dụ, một máy tính hoặc một điện thoại thông minh.
- Server: server là thiết bị nhận cả lệnh và yêu cầu GATT, sau đó trả về phản hồi. Ví dụ, một cảm biến độ ẩm.
- Đặc trưng: Giá trị được truyền giữa client và server là đặc trưng. Ví dụ, thông tin cảm biến hiện tại.

- Dịch vụ: Một nhóm các đặc tính liên quan hoạt động cùng nhau để thực hiện một chức năng cụ thể được gọi là dịch vụ. Ví dụ: Thông tin độ âm bao gồm các đặc điểm cho giá trị đo độ âm và khoảng thời gian giữa các lần đo.
- Bộ mô tả: Bộ mô tả cung cấp thêm thông tin về một đặc tính. Một ví dụ rõ ràng là đặc tính độ âm có thể có thuộc tính của các đơn vị của nó và phạm vi giá trị mà cảm biến có thể đo được. Bộ mô tả là tùy chọn trong mô hình Bluetooth.
- UUID: UUID là số nhận dạng của các thuộc tính GATT bao gồm các dịch vụ, đặc điểm và bộ mô tả. Có hai loại UUID: mã ngắn (16-bit hoặc 32-bit) và mã hoàn chỉnh (128-bit).



Hình 42. Cấu trúc GATT

#### 2.6.4.9 GAP

Generic Access Profile (GAP) kiểm soát các kết nối và quảng cáo trong BLE. GAP sẽ quyết định rằng các thiết bị có thể tương tác với nhau.

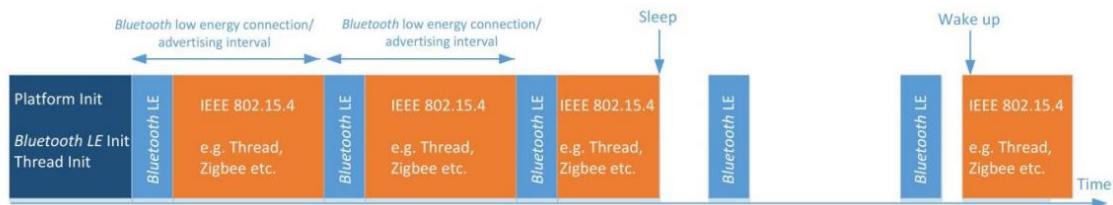
Có bốn vai trò mà GAP đã xác định cho các thiết bị sử dụng BLE. Hơn nữa, một thiết bị có thể hoạt động với nhiều vai trò cùng một lúc nếu lớp kết hỗ trợ. Các vai trò này được mô tả dưới đây.

- Broadcaster: gửi các sự kiện quảng cáo không kết nối được.

- Observer: quét các sự kiện quảng cáo từ một đài truyền hình; tuy nhiên, không thể bắt đầu kết nối.
- Peripheral: truyền các gói quảng cáo có thể kết nối. Các thiết bị ngoại vi cũng có thể chấp nhận thiết lập liên kết BLE. Sau khi kết nối thành công, các thiết bị hoạt động ở vai trò slave.
- Central: quét và khởi tạo kết nối BLE. Sau khi kết nối thành công, các thiết bị sẽ hoạt động ở các vai trò chính.

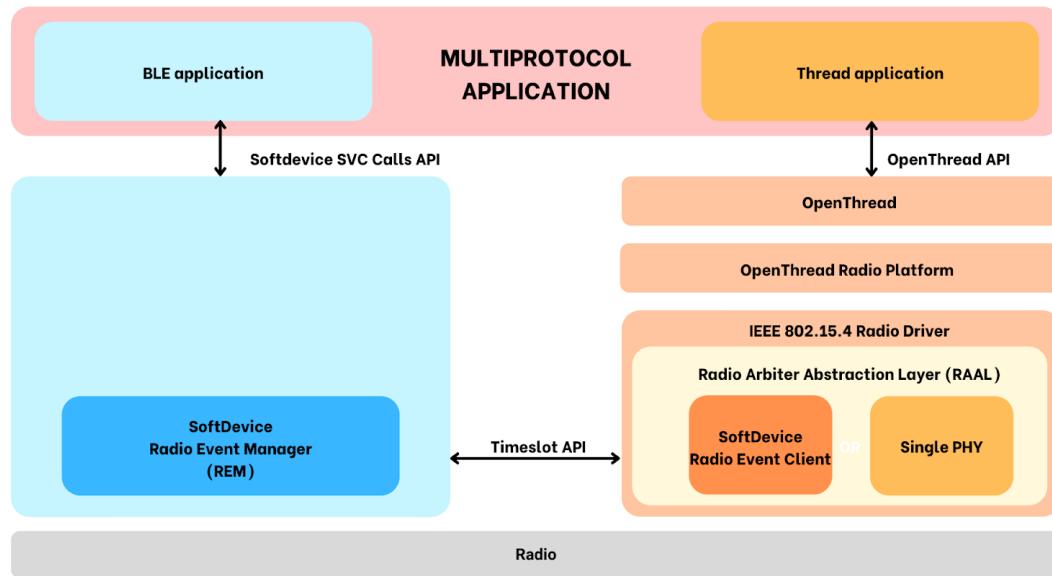
## 2.7 Kỹ thuật đa giao thức động

Kỹ thuật dynamic multiprotocol cho phép vài giao thức truyền thông có thể chạy đồng thời trên cùng một SoC, cùng một anten, cùng bộ Radio Peripheral và cũng không cần phải hủy bỏ (uninitialization) và khởi tạo (initialization) giữa những lần chuyển đổi qua lại giữa các giao thức [25].



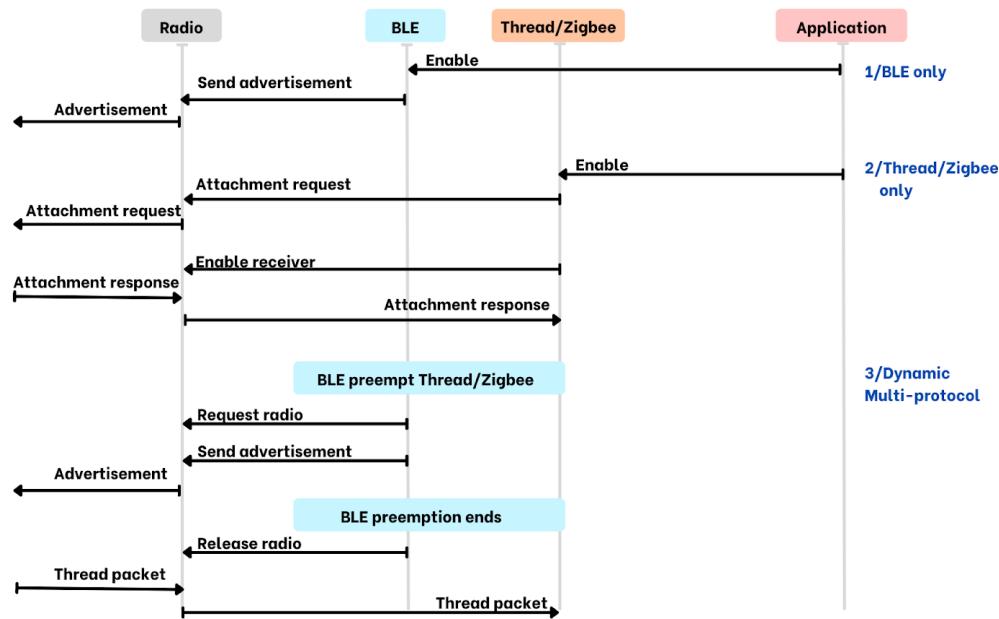
Hình 43. Lược đồ thời gian của bộ vô tuyến chạy Dynamic Multiprotocol BLE và Thread

Như trong lược đồ thời gian trên ta thấy BLE và Thread/Zigbee được khởi tạo một lần và chạy xuyên suốt ứng dụng. Radio Peripheral luôn chạy những giao thức (BLE và Thread/Zigbee) trong những khe thời gian của chúng. Do đó kỹ thuật này có thể duy trì các kết nối của các giao thức đang chạy tại.



Hình 44. Kiến trúc đa giao thức động của Nordic

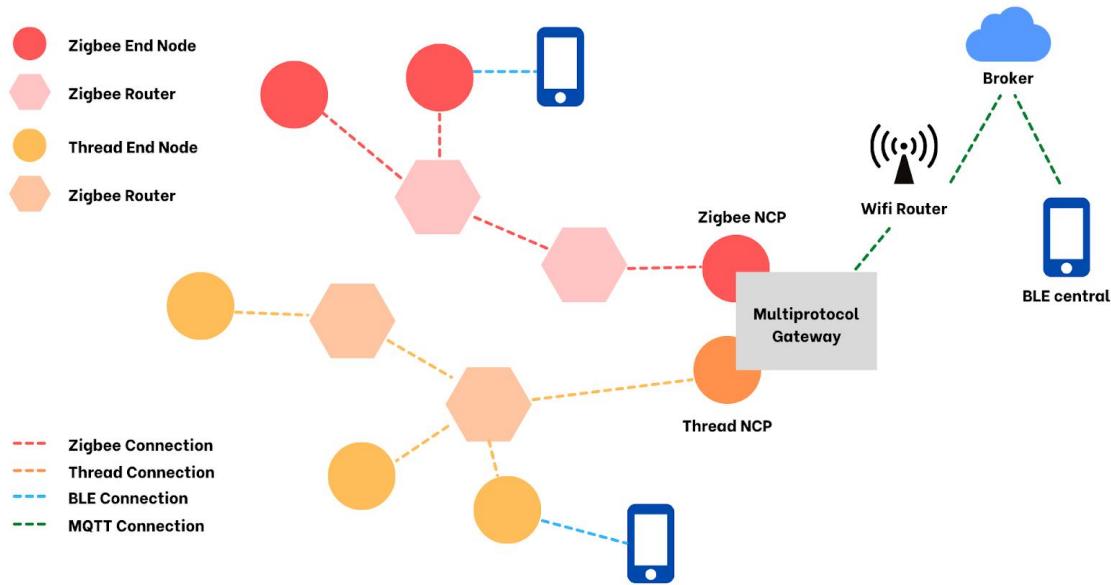
Việc truyền-nhận dữ liệu của giao thức này cũng không làm mất kết nối của các giao thức khác. Để đạt được kỹ thuật này cần có kỹ thuật cho phép các bộ Radio Driver truy cập bộ Radio Peripheral đồng thời, kỹ thuật này được gọi là Time-multiplexed Radio Access. Kỹ thuật Time-multiplexed Radio Access được thực hiện nhờ một bộ Radio Arbiter [7]. Softdevice của Nordic có bộ Radio Event Manager (REM), quản lý sự phân chia vô tuyến (radio arbitration) và cung cấp các Timeslot API cho các bộ Radio Driver khác. Bộ 802.15.4 Radio Driver cho nền tảng nRF – sử dụng trong OpenThread – có bộ Radio Arbiter Client, cái này sử dụng các Timeslot API để quản lý các quá trình truyền-nhận của nó và yêu cầu Timeslot trước khi thực hiện bất kỳ hoạt động vô tuyến nào.



Hình 45. Độ ưu tiên gói tin khi chạy đa giao thức động

Việc chuyển đổi qua lại giữa các giao thức đang chạy được thực hiện tự động ở lớp nền (background) của ứng dụng đang chạy. Do đó, kỹ thuật dynamic multiprotocol này là trong suốt (transparent) với người phát triển ứng dụng. Viết phần BLE cho ứng dụng multiprotocol cũng giống như viết ứng dụng chỉ BLE, tương tự với Thread/Zigbee.

## CHƯƠNG 3. ĐẶC TẢ HỆ THỐNG



Hình 46. Topology hệ thống

### 3.1 Khối Router

Hỗ trợ giao thức MQTT-SN để kết nối với MQTT Broker và gửi/nhận dữ liệu qua mạng Internet. Từ đó, người dùng có thể điều khiển hoặc xem trạng thái của các thiết bị trong nhà qua ứng dụng trên Smartphone khi không có mặt ở nhà. Node Router đóng vai trò MQTT-SN Client kết nối với MQTT-SN Gateway (Border Router) trỏ đến MQTT Broker. Do đó, Border Router cần hỗ trợ thư viện MQTT-SN Gateway để điều hướng gói tin đến broker.

Hỗ trợ giao thức BLE để điều khiển node Router trực tiếp khi người dùng đang ở gần trong khi vẫn duy trì chế độ chờ trong nội mạng Thread/Zigbee. Điều này giúp tiết kiệm năng lượng hơn so với giao thức MQTT-SN thông thường. Khi thực hiện phần mềm cần một BLE GATT Service để điều khiển trực tiếp từ Smartphone.

Hỗ trợ tính năng chuyển tiếp gói tin từ BLE sang mạng Thread, nghĩa là khi node Router thiết lập kết nối với Smartphone, Smartphone có thể điều khiển các node mạng lân cận đang kết nối với Router. Trong trường hợp Router bị sập (cúp điện, hư nguồn) nên mạng Thread không thể kết nối Internet, đây là giải pháp giúp người dùng vẫn điều khiển

## ĐẶC TÍNH HỆ THỐNG

---

được những thiết bị trong nhà. Để chuyển tiếp gói tin đến node mạng khác, thiết bị BLE cần đính kèm địa chỉ IPv6 của node mạng đích và chuyển đi bằng giao thức CoAP.

Node Router sẽ được bổ sung nguồn trong quá trình hoạt động do tần suất chuyển phát gói tin liên tục sẽ tồn nồng lượng hơn so với end node.

### 3.2 Khối End node

Đối với End node mạng Thread, do Thread chạy trên nền IPv6, ta có thể tích hợp tính năng MQTT-SN client để nhận tín hiệu điều khiển hoặc thu thập dữ liệu qua Broker. Đối với End node mạng Zigbee, do Zigbee không chạy trên nền IPv6, nên không thể tích hợp giống Thread. Việc thu thập/điều khiển qua Broker sẽ do Gateway và Zigbee NCP đảm nhận.

Giao thức BLE được lược bỏ trên firmware của End node và giảm duty cycle để tiết kiệm pin. Những nodes này có tên gọi khác là Sleepy end device. Một chu kỳ thông thường của sleepy end device là:

- Thức dậy từ trạng thái ngủ
- Khởi tạo radio
- Vào chế độ nhận gói tin/truyền gói tin
- Gửi/nhận gói tin ACK xác nhận việc truyền tin thành công
- Chuyển sang trạng thái ngủ

### 3.3 Khối Gateway

Trong mạng Thread, Gateway đóng vai trò là Border Router. Đây là bộ định tuyến nằm ở rìa của mạng Thread và định tuyến giữa mạng Thread và mạng bên ngoài. Border Router cung cấp kết nối của các node trên mạng Thread với các thiết bị khác trong mạng bên ngoài (Wi-Fi). Do đó, các yêu cầu cần có của Border Router:

- Hỗ trợ NAT64, DNS64 bởi vì Thread dựa trên IPv6, Thread có thể mang một hoặc nhiều protocol đồng thời ở lớp ứng dụng. Border Router có thể giao tiếp với mạng ngoài qua IPv6 (không cần translator) hoặc IPv4 (sử dụng NAT64 để dịch từ IPv6 sang IPv4, DNS64 để node chạy Thread giao tiếp được với các server IPv4)

## ĐẶC TẢ HỆ THỐNG

---

- Hỗ trợ external commissioning (Một thiết bị ngoài mạng Thread (smartphone) cấp quyền cho các node cảm biến gia nhập mạng Thread hiện tại và truyền các chứng chỉ mạng, khóa bí mật với Border Router để xác thực)
- Trong mạng Zigbee, Gateway đóng vai trò bộ đệm thông tin, dữ liệu mạng Zigbee đến các chủ đề trên Broker qua giao thức MQTT
- Gateway cần kết nối với NCP (Network Co-processor) để quản lý, tương tác với mạng Thread qua thư viện Pyspinne và tương tác với mạng Zigbee qua giao thức ZRSP.

### 3.4 Khối Broker

Broker cần định tuyến được những gói tin gửi đến theo chủ đề cho các clients đã đăng ký khi kết nối vào mạng. Có thể chọn Broker là một server được host bởi các bên thứ ba (HiveMQ, Eclipse Project, ... ) hoặc chúng ta host server cho Broker trên Border Router.

### 3.5 Khối Điều khiển

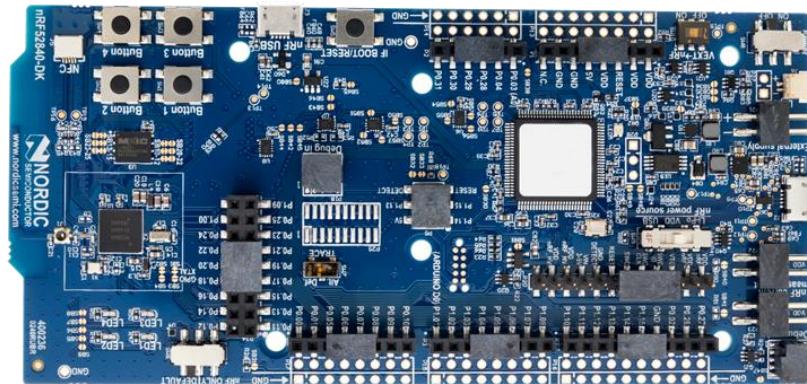
Smartphone/PC cần hỗ trợ Bluetooth phiên bản 4.0 trở lên (Từ phiên bản 4.0, Bluetooth hỗ trợ chuẩn Bluetooth Low Energy, tiết kiệm năng lượng hơn so với Bluetooth classic).

Smartphone/PC cần cài phần mềm hỗ trợ chức năng commissioning (mô tả ở khối Border Router), điều các nodes trong mạng qua BLE hoặc MQTT.

## CHƯƠNG 4. THIẾT KẾ VÀ THỰC HIỆN

### 4.1 Mô tả phần cứng

#### 4.1.1 nRF52840 Development Kit



Hình 47. nRF52840 Development Kit

Vi điều khiển

- ARM Cortex M4 32 bit
- Tốc độ clock 64 MHz
- Hỗ trợ debug qua serial (SWD)
- 1 MB flash và 256 kB RAM

Kết nối RF:

- Bluetooth 5, IEEE 802.15.4, bộ thu phát 2.4 GHz
- Độ nhạy - 95 dBm khi ở chế độ BLE 1 Mbps
- Độ nhạy - 103 dBm khi ở chế độ BLE 125 kbps (truyền tầm xa)
- Công suất TX từ - 20 đến + 8 dBm

Công suất thấp:

- Điện áp nguồn hỗ trợ từ 1.7 V to 5.5 V
- Điện áp cho ngoại vi hỗ trợ từ 1.8 V to 3.3 V
- Active-Mode RX (CPU Idle): 4.6 mA
- Active-Mode TX ở 0 dBm (CPU Idle): 4.8 mA

Ngoại vi

## THIẾT KẾ VÀ THỰC HIỆN

---

- Hỗ trợ easyDMA
- 5 x 32-bit timer
- 12-bit, 200 ksps ADC
- Hỗ trợ Softdevice - Nordic cho ứng dụng Multiprotocol
- Cảm biến nhiệt độ
- 4 x SPI
- 2 x UART

nRF52840 Dongle và nRF52840 Customized Kit đều sử dụng lõi nRF52840 SoC nhưng đã được tối giản để phù hợp với ứng dụng. Cụ thể:

- Dongle được sử dụng làm Network Co-Processor, kết nối với Border Router (Raspberry Pi) để thành lập, quản lý mạng Thread.
- Customized Kit là phiên bản rút gọn của Development kit được sử dụng làm các node thu thập dữ liệu, yêu cầu ít sức mạnh xử lý và công suất thấp.

### 4.1.2 CC2538 Customized Kit



Hình 48. CC2538 Customized Kit

Vi điều khiển:

- ARM Cortex M3
- Tốc độ clock 32 MHz
- Hỗ trợ On-Chip Over-the-Air Upgrade (OTA)
- Hỗ trợ Dual ZigBee Application ProNes
- 32kB RAM
- cJTAG và JTAG Debugging

Kết nối RF:

## THIẾT KẾ VÀ THỰC HIỆN

- Hỗ trợ thu phát băng tần 2.4 GHz IEEE 802.15.4
- Độ nhạy tín hiệu -97 dBm - Robustness to Interference With ACR of 44 dB -
- Công suất phát tín hiệu lên đến 7 dBm

Công suất thấp:

- Active-Mode RX (CPU Idle): 20 mA
- Active-Mode TX ở 0 dBm (CPU Idle): 24 mA
- Điện áp hỗ trợ từ 2 V to 3.6 V

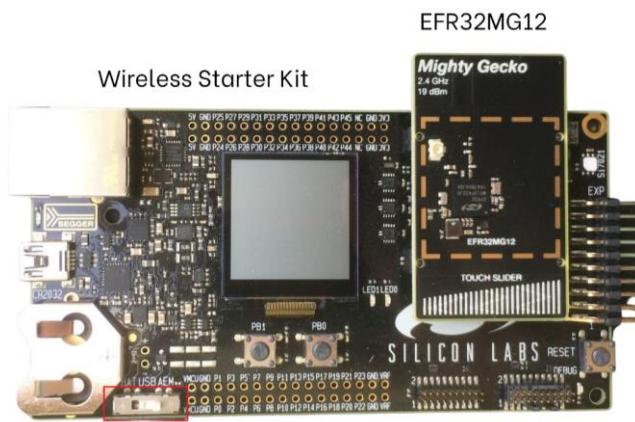
Ngoại vi:

- uDMA
- 4 x General-Purpose Timers
- 32-Bit 32-kHz Sleep Timer
- 12-Bit ADC với 8 kênh và độ phân giải thay đổi được

Quản lý dung lượng pin

- 2 x SPI
- 2 x UART
- I2C
- 32 GPIO Pins (28 x 4 mA, 4 x 20 mA)

### 4.1.3 Silabs EFR32MG12 + Wireless Starter Kit



Hình 49. EFR32MG12

Vị điều khiển:

- ARM Cortex M4

## THIẾT KẾ VÀ THỰC HIỆN

---

- Tốc độ clock 40 MHz
- Hỗ trợ On-chip Over-the-Air (OTA)
- Hỗ trợ Dynamic Multiprotocol
- J-Link debug
- 256 kB RAM

Kết nối RF:

- Hỗ trợ thu phát băng tần 2.4 GHz IEEE 802.15.4
- Công suất phát tín hiệu lên đến 19 dB

Công suất thấp:

- Active-Mode RX (CPU Idle): 11 mA
- Active-Mode TX ở 0 dBm (CPU Idle): 8 mA
- 70 µA/MHz ở Active Mode (EM0)

Ngoại vi:

- 65 GPIOs
- 8 kênh DMA
- $2 \times$  16-bit Timer/Counter
- $2 \times$  32-bit Timer/Counter
- Low Energy UART
- Low Energy Sensor Interface

### 4.1.4 Raspberry PI 3

## THIẾT KẾ VÀ THỰC HIỆN



Hình 50. Raspberry PI

Vi xử lý

- Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz
- Bộ nhớ RAM 1GB LPDDR2 SDRAM

Kết nối RF

- GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- Gigabit Ethernet over USB 2.0 ( throughput tối đa 300 Mbps)

Cổng kết nối:

- 4 x USB 2.0
- 40-pin GPIO header
- Hỗ trợ thẻ nhớ SD cho OS và lưu dữ liệu
- 1 cổng HDMI

Nguồn hoạt động:

- 5V/2.5A DC qua cổng nguồn microUSB
- 5V DC qua GPIO header
- Power over Ethernet (PoE) (yêu cầu PoE HAT)

## 4.2 Thực hiện Firmware

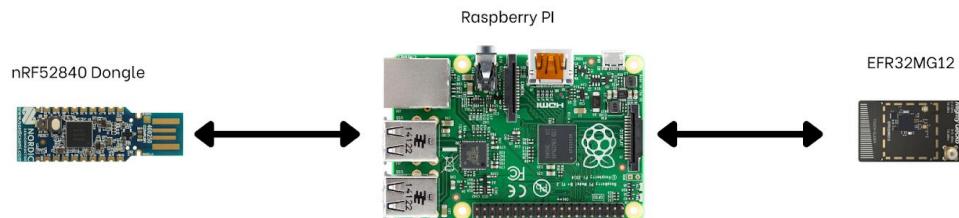
### 4.2.1 Khởi gateway

Về phần cứng, Multiprotocol Gateway gồm 3 phần chính: Border Router (hoặc Linux-based Host) và hai thiết bị NCP (Network Co-Processor) cho mạng OpenThread

## THIẾT KẾ VÀ THỰC HIỆN

và mạng Zigbee . Ở trong luận văn này, Border Router là Raspberry Pi 3B và NCP là nRF52840 Dongle (OpenThread NCP), EFR32MG12 (Zigbee NCP). Các thành phần này kết nối với nhau thông qua chuẩn USB, tốc độ baud 100000 baud/s.

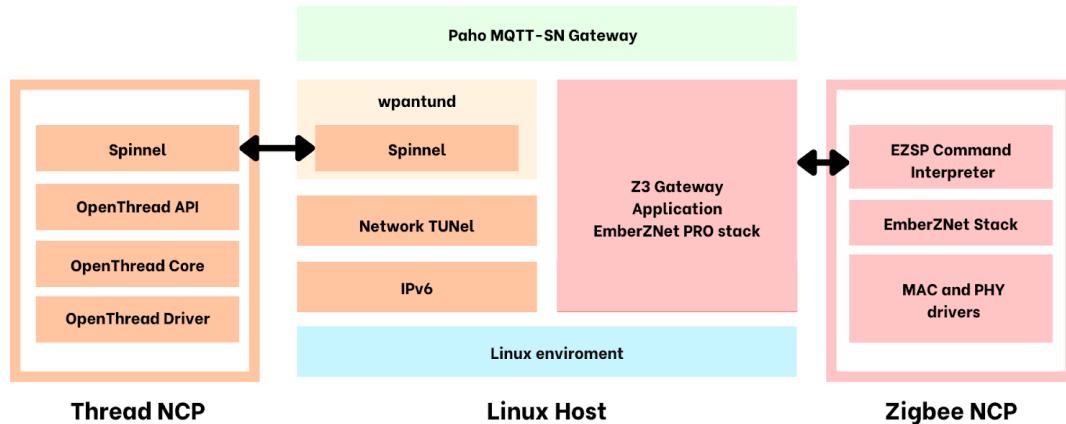
### Phần cứng



Hình 51. Phần cứng Gateway

Về phần mềm để khởi tạo mạng OpenThread, nhóm đã xây dựng thành công một Thread Border Router, cái này được gọi là OpenThread Border Router (OTBR). OTBR hiện tại hỗ trợ 2 nền tảng là Raspberry Pi 3B và BeagleBone Black. OTBR trên Raspberry Pi 3B đã được cấp chứng chỉ Thread (Thread Certified Component) khi hoạt động với thiết bị NCP dựa trên nền tảng Nordic nRF52840. Mã nguồn OTBR có sẵn tại: <https://github.com/openthread/ot-br-posix>. Firmware NCP cho nền tảng nRF52840 có sẵn trong SDK for Thread and Zigbee v4.0.1 của Nordic.

### Phần mềm



Hình 52. Mô hình phần mềm của Gateway

Tương tự với mạng Zigbee, nhóm cũng tích hợp tùy chọn sau vào firmware Zigbee Gateway: cJSON (Hỗ trợ file JSON cho gói tin MQTT), device-table (Lưu danh sách các thiết bị trong mạng Zigbee). Để tích hợp chức năng MQTT-SN gateway vào Border Router, chúng ta cần cài thêm service paho-mqtt-sn. Mã nguồn MQTT-SN gateway có sẵn tại: <https://github.com/eclipse/paho.mqtt-sn.embedded-c/tree/master/MQTTSGateway>.

Kết quả thực hiện Gateway:

## THIẾT KẾ VÀ THỰC HIỆN

The screenshot shows a Linux terminal window with three tabs open:

- Thread Network Log:** Displays network traffic on port 25.305. It includes entries for MeshForwarder and MeshManager, showing messages like "Send Advertisement" and "Set tx power mode".
- Zigbee Network Log:** Displays configuration commands for Zigbee interfaces. It includes "Esp Config" and "Esp Policy" entries, such as "set address table size to 0x0002:Success" and "allow for valid endpoints & clusters".
- MQTT Gateway Log:** Displays MQTT client activity. It shows connections to "gw/00057FFFFE648D00", publishing messages to topics like "gw/00057FFFFE648D00/commands" and "gw/00057FFFFE648D00/publishstate", and subscribing to "gw/00057FFFFE648D00/updatesettings".

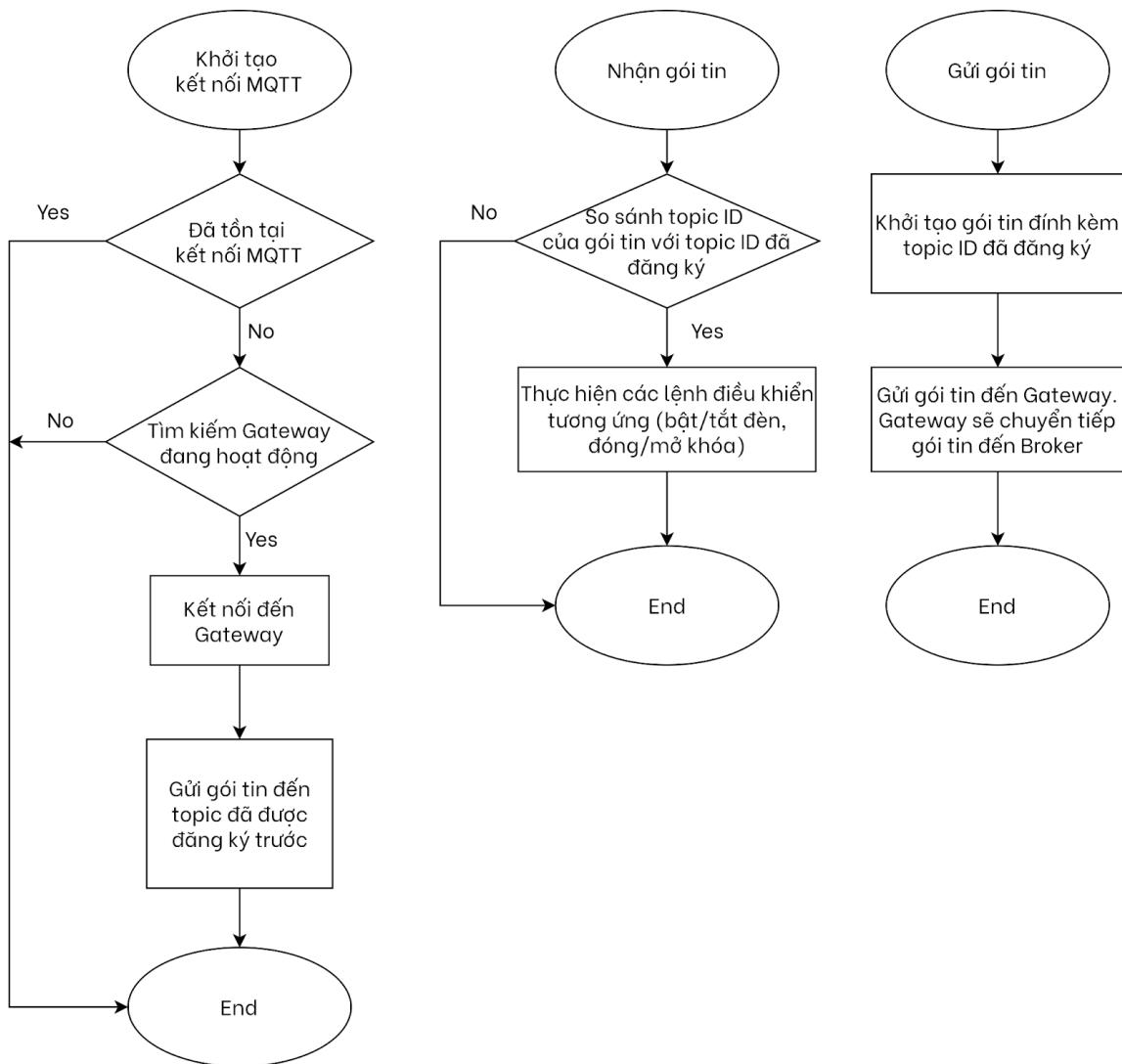
Hình 53. Thông tin mạng hiển thị trên Linux-host

### 4.2.2 Khởi Router

#### 4.2.2.1 MQTT-SN client trên nền mạng Thread

MQTT-SN là một giao thức ở lớp ứng dụng được xây dựng trên nền của mạng Thread. Để xây dựng firmware cho tính năng MQTT-SN, cần khởi tạo mạng Thread, sau đó tiến hành các kết nối MQTT đến Broker thông qua Gateway. Ứng dụng này của nhóm có thể tích hợp tính năng publish và subscribe vào một node mạng, tùy vào tình huống sử dụng để tối ưu năng lượng. Sơ đồ mô tả hoạt động:

## THIẾT KẾ VÀ THỰC HIỆN



Hình 54. Giải thuật nhận/gửi gói tin MQTT trên Gateway

(a) Quá trình khởi tạo kết nối và tìm kiếm Gateway/Broker để gửi gói tin đăng ký

Quá trình kết nối đến MQTT Broker bắt đầu bằng bước kiểm tra xem đã tồn tại kết nối chưa. Nếu chưa, node mạng sẽ bắt đầu tìm kiếm gateway đang hoạt động trong thời gian nhất định. Kết quả được trả về qua hàm gateway\_info\_callback hoặc searchgw\_timeout\_callback. Node mạng sẽ gửi gói tin CONNECT đến thông tin gateway từ hàm gateway\_info\_callback. Sau khi kết nối thành công với gateway, node mạng tiến hành đăng ký chủ đề bằng gói tin SUBSCRIBE (mang theo topic ID).

(b) Quá trình xử lý khi nhận được gói tin từ chủ đề đã đăng ký

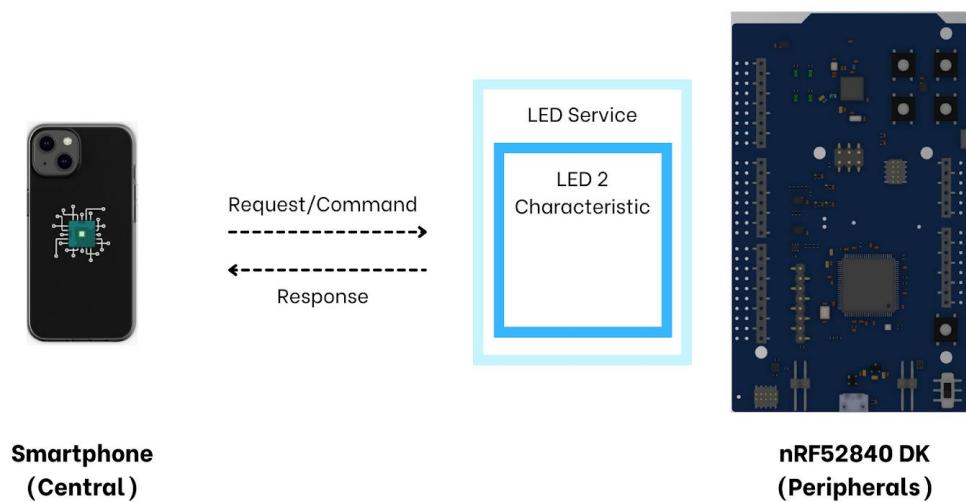
## THIẾT KẾ VÀ THỰC HIỆN

Khi nhận được gói tin từ gateway, node sẽ xử lý thông qua hàm received\_callback. Node sẽ tiến hành so sánh trường topic\_id để xác định đây là gói tin từ chủ đề đã đăng ký. Nếu đúng, node sẽ đọc payload của gói tin và thực hiện tác vụ tương ứng (bật/tắt đèn LED, đóng mở chốt cửa, ...). Ngược lại, nếu không phải thì node sẽ bỏ gói tin.

### (c) Quá trình gửi gói tin

Để gửi gói tin đến Broker cần xác định các thông số: p\_client, topic\_id, p\_payload, payload\_len, p\_msg\_id. Sau đó sử dụng hàm mqtsn\_client\_publish có sẵn từ Nordic.

#### 4.2.2.2 BLE service điều khiển LED



Hình 55. BLE service điều khiển LED

Ở phần này, nhóm sẽ tiến hành xây dựng một BLE service để điều khiển đèn LED 2 trên các board nRF52840 gọi là led\_service. Mô tả của led\_service như sau:

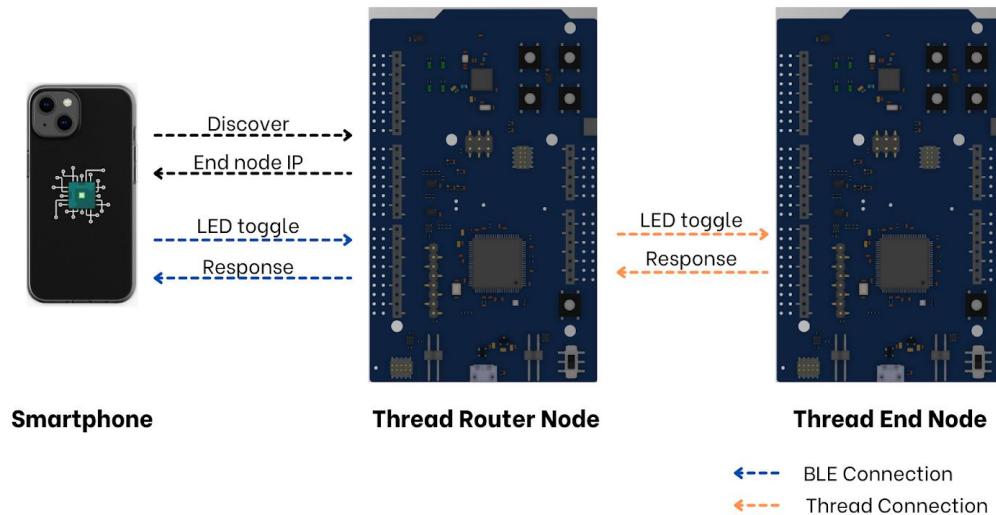
- LED service UUID: E54B0000-67F5-479E-8711-B3B99198CE6C
- LED 2 characteristic UUID: E54B0002-67F5-479E-8711-B3B99198CE6C
- Permission của service:
  - Đọc
  - Ghi
  - Không thông báo

Các hàm quan trọng trong service:

## THIẾT KẾ VÀ THỰC HIỆN

- Khởi tạo LED service: `uint32_t ble_led_service_init(ble_led_service_t * p_led_service, const ble_led_service_init_t * p_led_service_init);`
- Xử lý sự kiện BLE: `void ble_led_service_on_ble_evt(ble_evt_t const * p_ble_evt, void * p_context);`

### 4.2.2.3 BLE/Thread Forwarding service



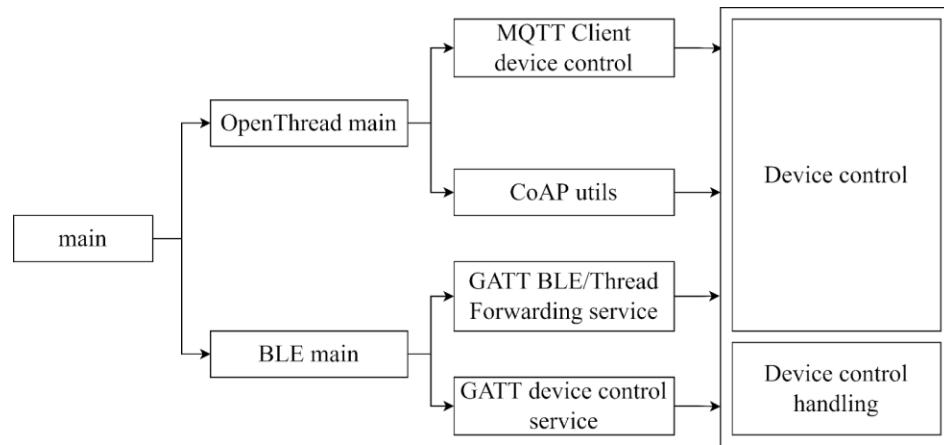
Hình 56. Nguyên lý hoạt động của BLE/Thread Forwarding Service

BLE/Thread Forwarding service được viết trên lớp GATT (Generic Attribute Profile) của BLE. Mục đích của service này là để chuyển tiếp lệnh điều khiển từ điện thoại thông minh đến các thiết bị chỉ chạy Thread. Đa số các thiết bị điện thoại thông minh trên thị trường hiện nay không hỗ trợ Thread hay các giao thức IEEE 802.15 nói chung. Do đó, trong trường hợp Thread Border Router gặp sự cố ngưng hoạt động, BLE/Thread Forwarding Service hỗ trợ người dùng truy cập điều khiển các thiết bị có trong mạng Thread mà không bị gián đoạn.

Về firmware, BLE/Thread Forwarding Service được xây dựng trên một node chạy song song hai giao thức động BLE và Thread. Hình bên dưới mô tả cấu trúc firmware của một BLE/Thread Forwarder node. Firmware này bao gồm hai thành phần: OpenThread main và BLE main. Phía BLE của node chịu trách nhiệm kết nối và nhận lệnh từ điện thoại

## THIẾT KẾ VÀ THỰC HIỆN

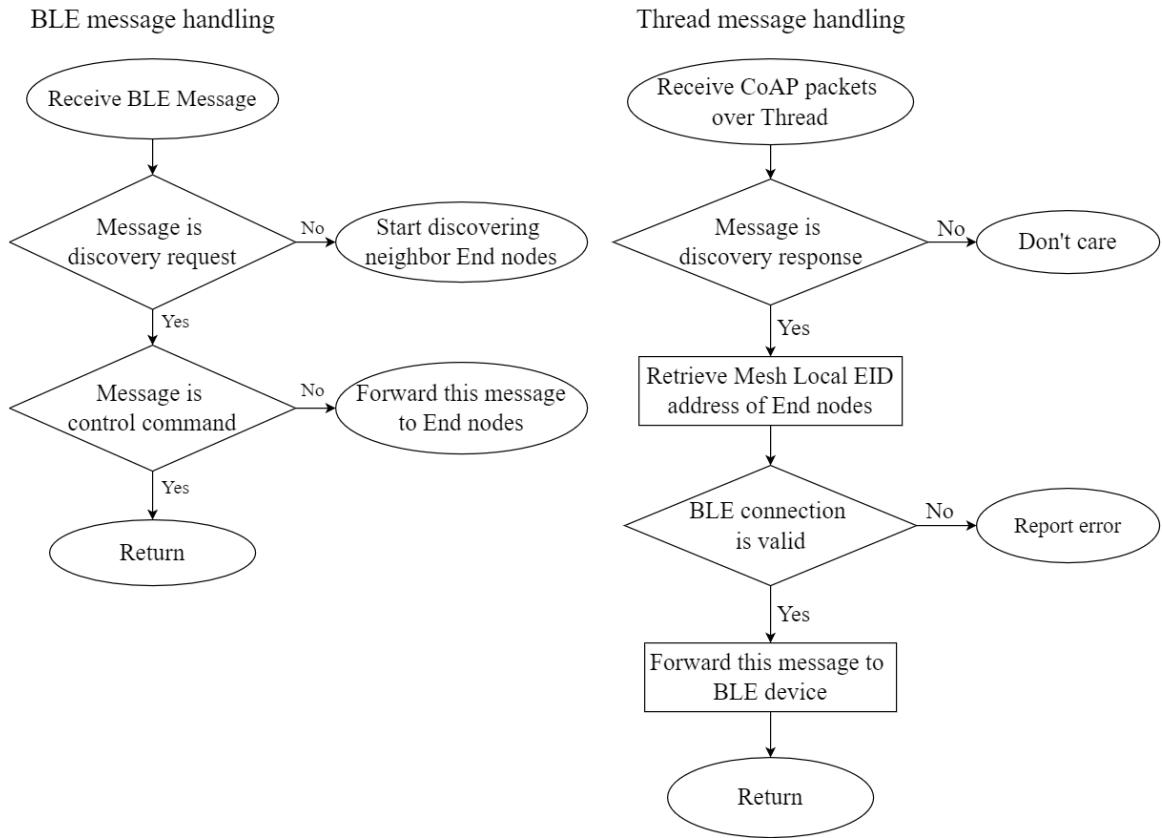
thông minh qua BLE. Phía OpenThread xử lý và chuyển tiếp các gói tin này đến các Thread node.



Hình 57. Cấu trúc BLE/Thread Forwarding

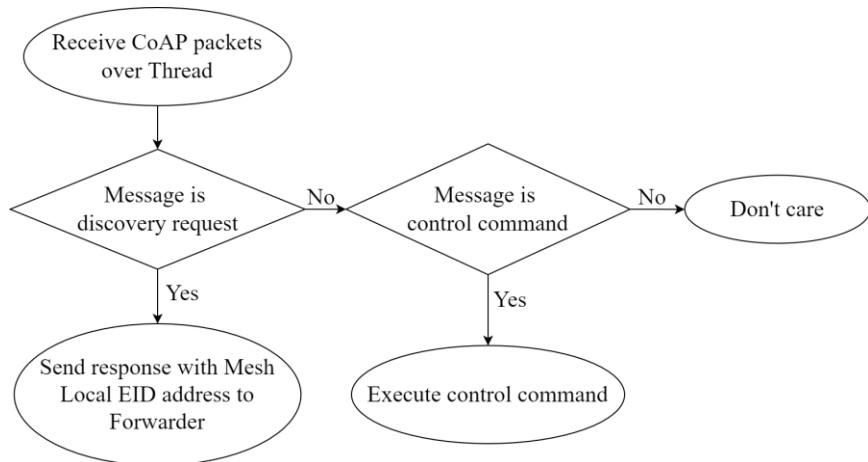
Giải thuật của quá trình chuyển tiếp gói tin trên một node Forwarder được diễn ra như hình bên dưới. Trong đó, người dùng có thể gửi lệnh “discover” để tìm kiếm các node Thread lân cận node Forwarder, sau đó gửi lệnh điều khiển dựa trên danh sách thiết bị lân cận phát hiện được (BLE message handling). Trong quá trình “discover”, các node Thread lân cận node Forwarder sau khi nhận được discover request sẽ gửi gói tin trả lời trong đó có chứa địa chỉ IPv6 của node. node Forwarder khi nhận được gói tin này sẽ gửi đến điện thoại thông minh qua BLE để thông báo cho người dùng (Thread message handling).

## THIẾT KẾ VÀ THỰC HIỆN



Hình 58. Giải thuật gửi gói tin BLE/Thread Forwarding Service

Hình bên dưới mô tả giải thuật xử lý các gói tin từ node Forwarder trên phía Thread End node.



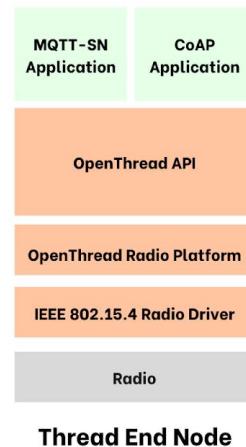
Hình 59. Giải thuật xử lý gói tin BLE/Thread Forwarding Service

### 4.2.3 Khối End Node

## THIẾT KẾ VÀ THỰC HIỆN

End node là các node lá, mục đích thu thập dữ liệu từ cảm biến hoặc nhận tín hiệu điều khiển. Do đó, End node sẽ chỉ thức dậy khi cần thiết để tiết kiệm năng lượng. Để thực hiện phần mềm nhúng cho khôi End node, mô hình firmware của End node do nhóm đề xuất là:

Đối với mạng Thread:



Hình 60. Stack của Thread End Node

- Nhóm sẽ lược bỏ giao thức BLE, chỉ sử dụng giao thức Thread, MQTT và CoAP để giao tiếp với mạng.
- Nhóm liên tục đưa End node vào trạng thái ngủ sau khi xử lý event. Lúc này, giải thuật xử lý gói tin MQTT-SN được bổ sung hàm đưa thiết bị vào trạng thái ngủ ở cuối mỗi chu trình xử lý event.

Đối với mạng Zigbee:



Hình 61. Stack của Zigbee End Node

- Tương tự, nhóm sẽ lược bỏ giao thức BLE, chỉ sử dụng giao thức Zigbee.
- Do phần xử lý mạng MQTT được thiết kế ở NCP và Gateway nên sẽ không có lớp MQTT-SN client ở Zigbee End Node giống mạng Thread.

### 4.2.4 Khối Broker

Khối Broker có nhiệm vụ nhận và chuyển tiếp dữ liệu từ khôi Gateway đến khôi điều khiển và ngược lại. Nhóm sử dụng HiveMQ (Public Broker miễn phí) làm Broker cho hệ thống. Bên cạnh đó, nhóm phải chỉnh sửa file config trên Gateway để mạng Thread có thể giao tiếp được với Broker.

```
BrokerName = broker.hivemq.com # Địa chỉ của Broker  
BrokerPortNo = 1883 # Port giao tiếp  
GatewayUDP6Port = 47193  
GatewayUDP6Broadcast = ff03::1  
GatewayUDP6If = wpan0  
GatewayUDP6Hops = 64
```

Kết quả kết nối Broker thành công: Node mạng nRF52840\_publisher có thể gửi gói tin CONNECT và REGISTER đến Broker.

## THIẾT KẾ VÀ THỰC HIỆN

```
* Author : Tomoaki YAMAGUCHI
* Version: 1.5.1
*****
20220423 102648.556 PahoGateway-01 has been started.

ConfigFile: ./gateway.conf
SensorN/W:   Gateway Port: 47193 Broadcast Address: ff03::1 Interface: wpan0 Hops: 64
Broker:      137.135.83.217 : 1883, 8883
Max number of Clients: 30
RootApath:  (null)
RootCafile: (null)
CertKey:    (null)
PrivateKey: (null)

20220423 102654.239  SEARCHGW      <--- Client          03 01 01
20220423 102654.239  GWTINFO       ---> Clients        03 02 01
20220423 102657.597  CONNECT        <--- nRF52840_publisher 18 04 04 01 00 3E 5E 52 46 35 32 38 34 30 5F 70 75 62 6C 69 73 68
65 72
20220423 102658.004  CONNECT        ==> nRF52840_publisher 10 1E 00 04 4D 51 54 54 04 02 00 3C 00 12 6E 52 46 35 32 38 34 30
5F 76 75 62 6C 68 73 68 65 72
20220423 102658.415  CONNACK       <== nRF52840_publisher 20 02 00 00
20220423 102658.415  CONNACK       ---> nRF52840_publisher 03 05 06
20220423 102658.426  REGISTER       0001 <--- nRF52840_publisher 1D 0A 00 00 00 01 6E 52 46 35 32 38 34 30 5F 72 65 73 6F 75 72 63
65 73 2F 6C 65 64 33
20220423 102658.427  REGACK        0001 ---> nRF52840_publisher 07 0B 00 01 00 01 00
20220423 102702.034  PUBLISH        0002 <--- nRF52840_publisher 08 0C 20 00 01 00 02 01
20220423 102702.035  PUBLISH        0002 ==> nRF52840_publisher 32 1C 00 17 6E 52 46 35 32 38 34 30 5F 72 65 73 6F 75 72 63 65 73
2F 6C 65 64 33 00 02 01
20220423 102703.072  PUBACK        0002 <== nRF52840_publisher 40 02 00 02
20220423 102703.072  PUBACK        0002 ---> nRF52840_publisher 07 0D 00 01 00 02 00
```

Hình 62. Kết quả thiết kế Gateway thành công

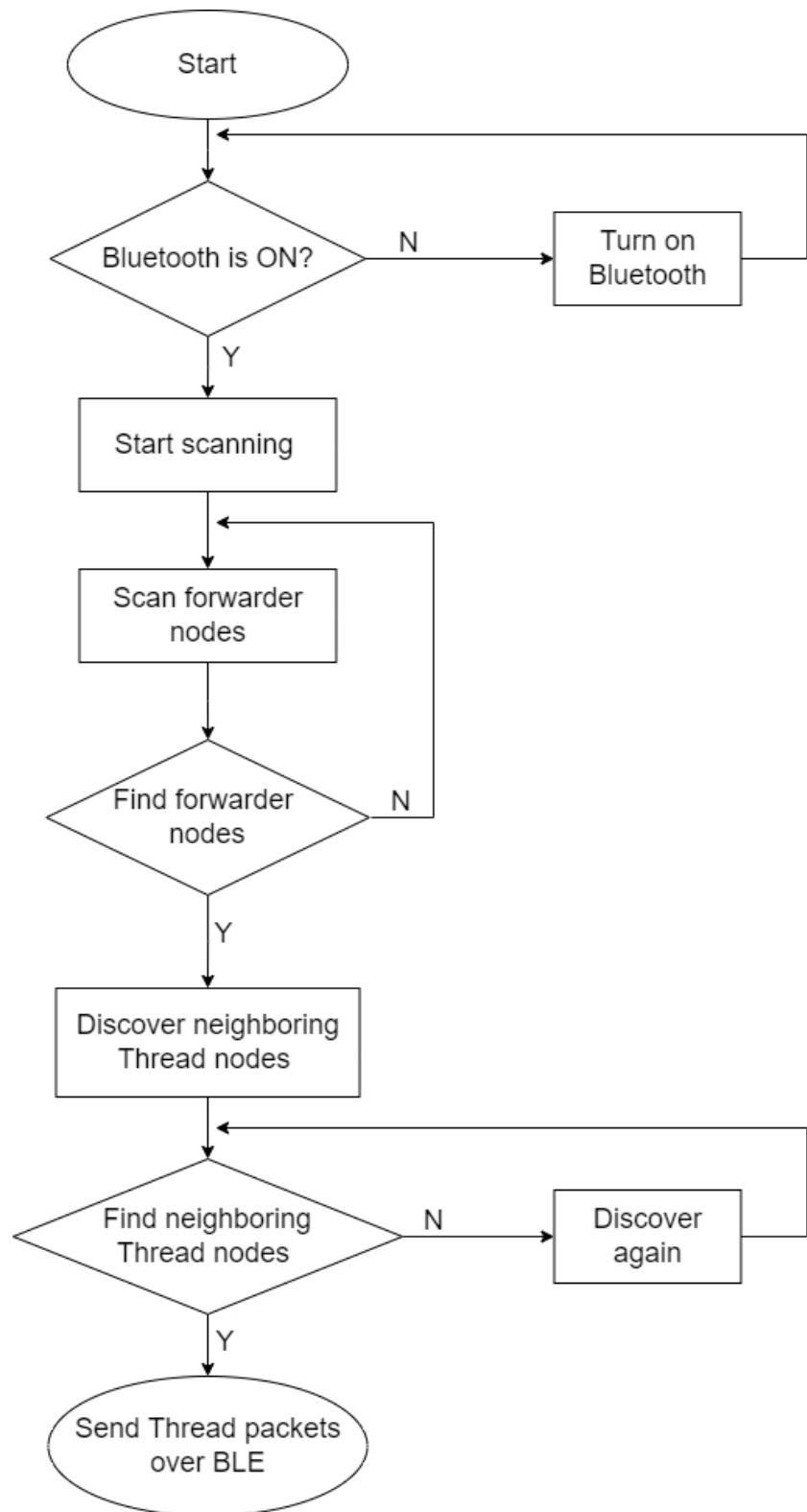
### 4.2.5 Khối điều khiển

Để thuận tiện hơn cho việc đánh giá độ hiệu quả của BLE/Thread Forwarding Service. Nhóm đã xây dựng một app trên hệ điều hành Android. App này có tính năng:

- Quét (Scan) các nút Forwarder dựa trên UUID thông qua BLE
- Cho phép người dùng kết nối với nút Forwarder
- Cho phép người dùng thực hiện tính năng Forwarding đến nút đích Thread

## THIẾT KẾ VÀ THỰC HIỆN

---

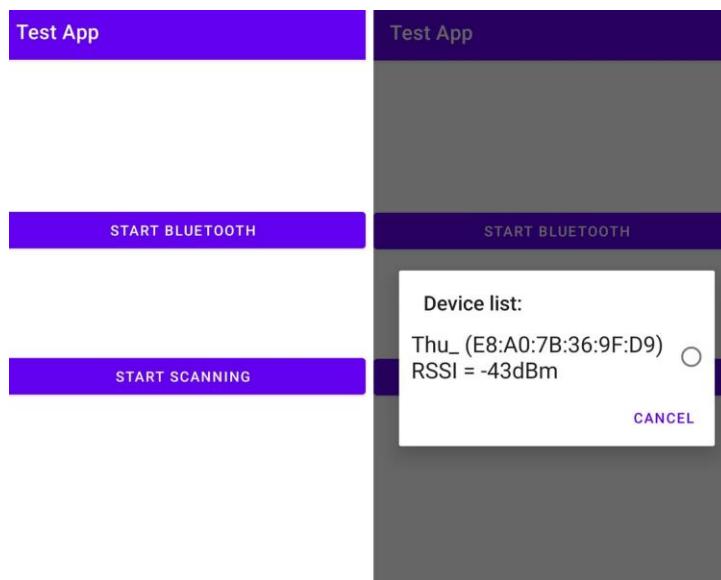


Hình 63. Giải thuật cho app điện thoại

## THIẾT KẾ VÀ THỰC HIỆN

Với phần mềm điện thoại này, người dùng có thể quét các nút BLE Forwarder xung quanh. Ở đây, trong thuật toán app, có sử dụng tính năng ScanFilter để lọc các thiết bị BLE theo UUID. Như vậy, khi app quét các thiết bị BLE, chỉ những thiết bị BLE cung cấp tính năng Forwarding mới hiển thị trên màn hình.

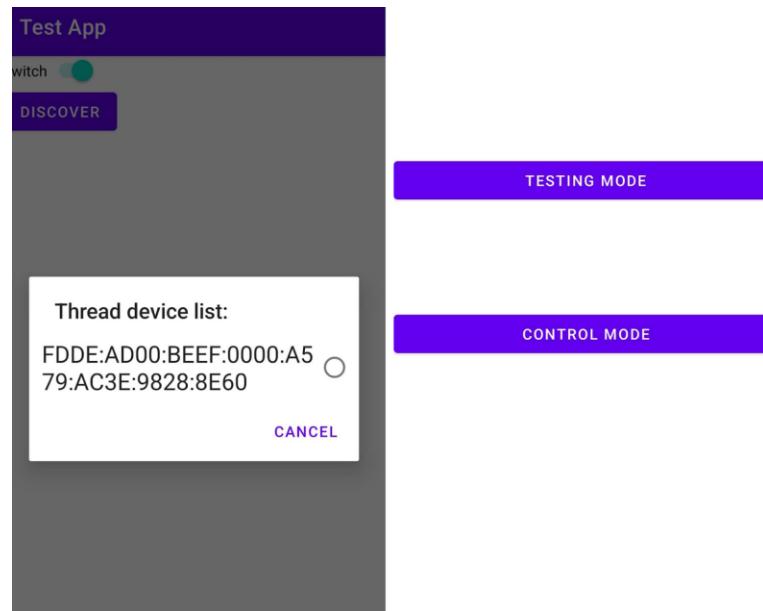
Khi kết nối với thiết bị Forwarder, giao diện sẽ hiện ra hai lựa chọn cho người dùng: BLE Control và BLE Forwarding. Với BLE Control, người dùng điều khiển trực tiếp LED trên Forwarder qua BLE. Với BLE Forwarding, người dùng có thể quét các nút Thread xung quanh và gửi các gói tin Thread thông qua BLE.



Hình 64. Giao diện app cho phép người dùng quét và kết nối với nút Forwarder thông qua BLE

Khi người dùng nhấn chọn “Discover” trong cửa sổ BLE Forwarding, các gói tin BLE mang nội dung “discover” sẽ được gửi đến Forwarder. Nút Forwarder sau đó sẽ xử lý các gói tin này như giải thuật đã mô tả trong khái Forwarder. Nội dung Forwarder trả về sẽ là địa chỉ của các nút Thread xung quanh (Hình 65). Khi người dùng chọn một trong các nút Thread trên, người dùng sẽ có hai lựa chọn là điều khiển LED hoặc gửi 1000 packets để test kết nối.

## THIẾT KẾ VÀ THỰC HIỆN



Hình 65. Giao diện app hiển thị kết quả cho quá trình discover và gửi các gói tin cho nút đích Thread

## CHƯƠNG 5. ĐÁNH GIÁ KẾT QUẢ

### 5.1 Đánh giá độ xa của đường truyền

#### 5.1.1 Kịch bản thử nghiệm

Kịch bản thực hiện với mục đích đánh giá khoảng cách tối đa mà node Forwarder chạy giao thức Thread có thể phát hiện được những node lân cận và khoảng cách tối đa điện thoại có thể phát hiện được nút Forwarder. Khoảng cách này được đo với các công suất truyền đầu vào khác nhau (0 - 4 - 8 dBm).

#### 5.1.2 Môi trường thử nghiệm

- Thực hiện ngoài trời (đường truyền Line-of-sight)
- Thực hiện trong nhà - nhà thi đấu (đường truyền Line-of-sight)
- Sử dụng Anten đơn cực có sẵn trên phần cứng

#### 5.1.3 Cách thức thực hiện

Đối với mỗi giao thức, thiết lập kết nối giữa 2 nút mạng với một nút cố định và một nút di động. Nút di động gửi tín hiệu điều khiển đến nút cố định. Cả hai nút đều được đặt cách mặt đất 1m. Khoảng cách sau đó sẽ được đo bằng GPS trên điện thoại smartphone (Samsung S21 FE) (đối với ngoài trời) và thước (đối với trong nhà).

#### 5.1.4 Kết quả

Bảng 15. Kết quả đánh giá độ xa của đường truyền

Phần cứng	Giao thức	Công suất TX (dBm)					
		0		4		8	
		Ngoài trời	Trong nhà	Ngoài trời	Trong nhà	Ngoài trời	Trong nhà

## ĐÁNH GIÁ KẾT QUẢ

nRF52840 Customized Kit	Thread	37m	42m	48m	52m	60m	62m
nRF52840 Development Kit	Thread	40m	43m	52m	56m	63m	65m
nRF52840 Customized Kit	BLE 1Mbps	51m	57m	62m	70m	74m	78m
nRF52840 Development Kit	BLE 1Mbps	53m	59m	65m	72m	79m	80m

### 5.1.5 Nhận xét

- Từ kết quả thí nghiệm cho thấy, ở môi trường ngoài trời, với công suất phát 0dBm, trong khoảng cách xấp xỉ 50m, điện thoại của người dùng vẫn có thể phát hiện được nút Forwarder thông qua BLE. Đồng thời, nút Forwarder cũng phát hiện được các nút Thread xung quanh trong bán kính xấp xỉ 37m.
- Trong khi đó, ở môi trường trong nhà, với công suất phát 0dBm, trong khoảng cách xấp xỉ 57m, điện thoại của người dùng vẫn có thể phát hiện được nút Forwarder thông qua BLE. Đồng thời, nút Forwarder cũng phát hiện được các nút Thread xung quanh trong bán kính xấp xỉ 42m.
- Khi đặt các nút trong môi trường nhiều vật cản (cây, tòa nhà,...), các thông số này giảm đáng kể.
- Thông số ghi nhận được ở môi trường trong nhà tốt hơn ngoài trời một phần là vì các hiện tượng phản xạ của sóng BLE, Thread.

## 5.2 Xác định khoảng cách truyền tối ưu (PRR > 95%)

### 5.2.1 Kích bản thử nghiệm

Kích bản thực hiện với mục đích đánh giá khoảng cách tối ưu của các giao thức Thread và Zigbee trên các phần cứng Nhóm đang sử dụng. Khoảng cách tối ưu được định nghĩa trong thí nghiệm này là khoảng cách mà kết nối giữa hai nút mạng cho kết quả PRR (Packet Receipt Ratio) lớn hơn 95%. Công suất phát của các thiết bị trong thí nghiệm này là 0dBm.

## ĐÁNH GIÁ KẾT QUẢ



Hình 66. Mô tả môi trường thử nghiệm cho đánh giá đường truyền tối ưu

### 5.2.2 Môi trường thử nghiệm

- Thực hiện trên đường trong khu dân cư, ngoài trời (đường truyền Line-of-sight)
- Thực hiện trong nhà - nhà thi đấu (đường truyền Line-of-sight)
- Sử dụng Anten đơn cực có sẵn trên phần cứng

### 5.2.3 Cách thức thực hiện

Được xây dựng tương tự với thí nghiệm đánh giá độ xa của đường truyền, đối với mỗi giao thức, thiết lập kết nối giữa 2 nút mạng với một nút cố định và một nút di động. Khoảng cách sau đó sẽ được đo bằng GPS trên điện thoại smartphone (Samsung S21 FE). Nút di động gửi 1000 gói tin chứa lệnh điều khiển (64 bytes data) đến nút cố định, mỗi gói tin cách nhau 200ms, công suất phát TX = 0 dBm. PRR = (số gói tin nhận được/1000)\*100%

## ĐÁNH GIÁ KẾT QUẢ

Thực hiện lệnh ping trên Gateway

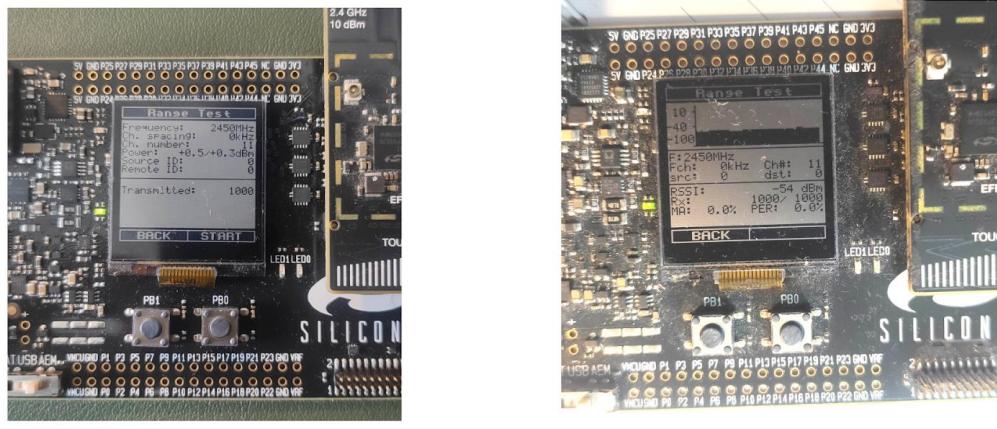
```

root@1357015887c7:/app
File Edit View Search Terminal Tabs Help
      thhp0 - x
root@1357015887c7:/app
File Edit View Search Terminal Help
      help
        lconfig
        logdr
        ipnaddr
        joiner
        joinerport
        keyspace
        leaderdata
        leaderpartitionid
        leaderweight
        mac
        netfilter
        nascryptkey
        mode
        neighbor
        netdataregiter
        netdevshow
        networkdiagnostic
        networkdimeout
        networkkname
        parent
        parentprior
        ping
        pollperfd
        previous
        prefix
        preferrouterld
        pskc
        releaserouterid
        res
        rloc16
        route
        router
        routelognogradethreshold
        routervrable
        routerselectionjitter
        routerupgradethreshol
        scan
        service
        singleton
        snntp
        state
        thread
        typepower
        udp
        version
        Done
> laddr
ff02:4476:4201:94a:4:0:ffff:fe00:4800
fd85:cl3e:738a:1:dbc2:1e3f:3db2:8f6c
ffdf:4476:4201:94a:4:916b:abcd:04a:41b9
fe80:0:0:0:8d7:6545:faf:b931
Done
> []

```

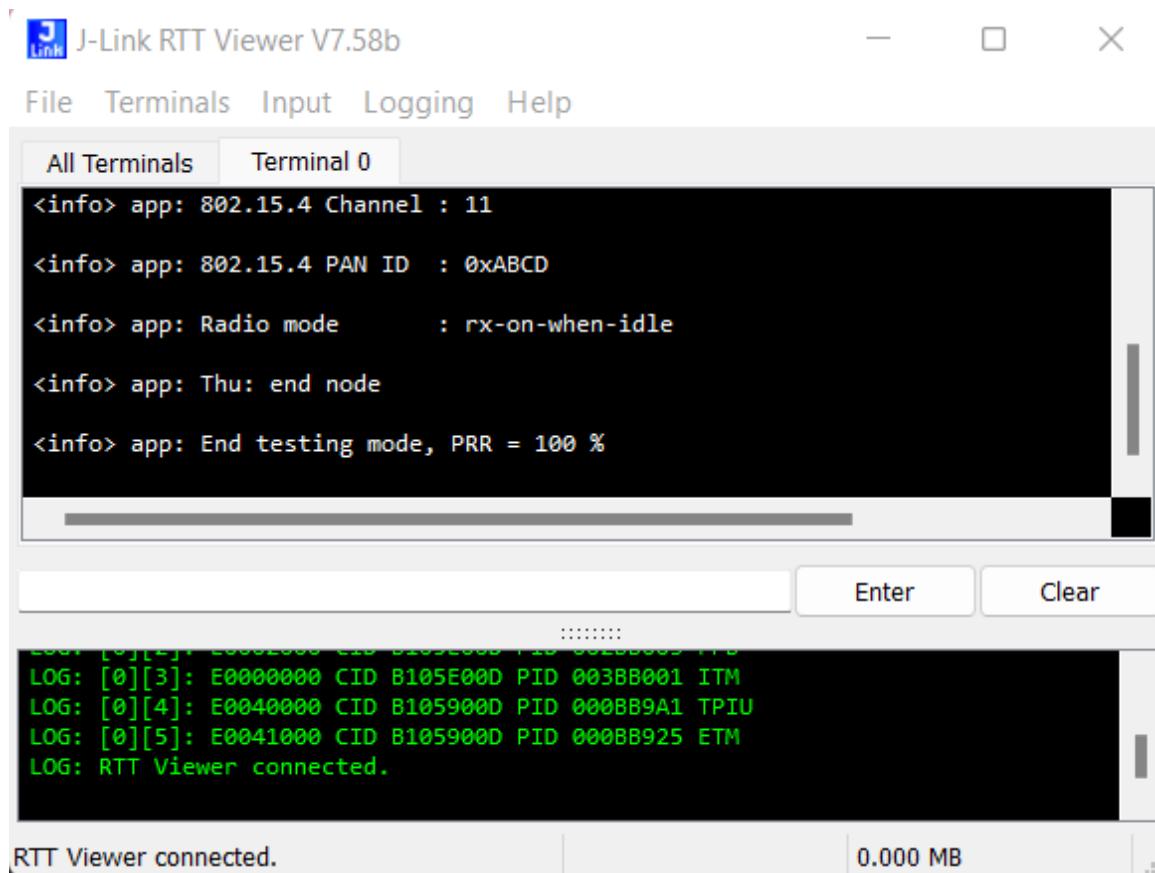
Node Thread đã kết nối vào mạng

Hình 67. Kết quả thực hiện với giao thức OpenThread 1



Hình 68. Kết quả thực hiện với giao thức Zigbee

## ĐÁNH GIÁ KẾT QUẢ



Hình 69. Kết quả thực hiện với giao thức OpenThread 2

### 5.2.4 Kết quả

Bảng 16. Kết quả xác định khoảng cách truyền tối ưu

Phần cứng	Giao thức	Packet Receipt Ratio - PRR (%)									
		100		99		98		97		96	
		Ngoài trời	Trong nhà	Ngoài trời	Trong nhà	Ngoài trời	Trong nhà	Ngoài trời	Trong nhà	Ngoài trời	Trong nhà
CC2538 Customized Kit	Thread	9m	11m	12m	13m	13m	15m	15m	17m	16m	18m
nRF52840 Customized Kit	Thread	10m	11m	11m	13m	13m	15m	15m	18m	17m	20m

## ĐÁNH GIÁ KẾT QUẢ

nRF52840 Development Kit	Thread	11m	13m	12m	15m	15m	17m	17m	19m	19m	22m
EFR32MG12	Zigbee	10m	12m	14m	15m	16m	18m	19m	21m	20m	22m

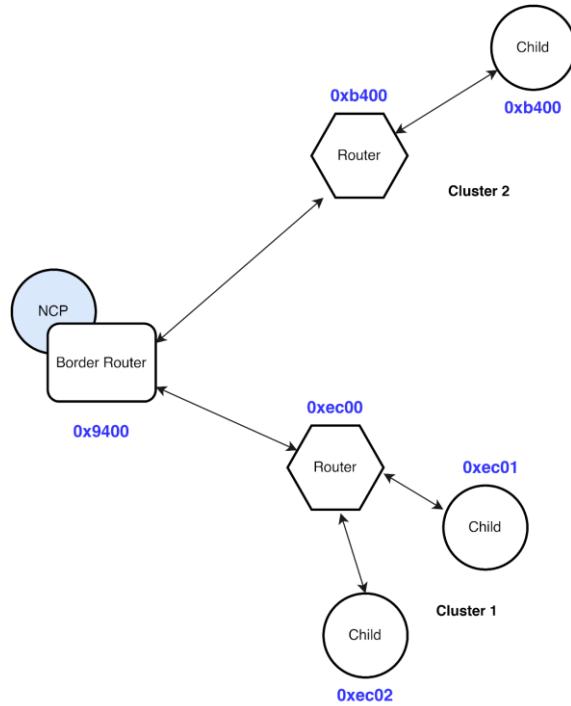
### 5.2.5 Nhận xét

- Do thực hiện trên đường truyền Line-of-sight nên PRR của các node mạng rất tốt trong khoảng cách từ 10 - 22m
- Dựa trên kết quả của thử nghiệm này, nhóm quyết định nhóm khoảng cách tối ưu, đảm bảo các ứng dụng sẽ hoạt động tốt là 10m (PRR ~ 100%) để tiến hành các kịch bản thử nghiệm tiếp theo.

## 5.3 Đánh giá truyền tin đa chặng

### 5.3.1 Kịch bản thử nghiệm 1

Trong môi trường hoạt động thực tiễn, các node cảm biến được bố trí trong một phòng và truyền tin đến Border Router. Các node trong mỗi phòng tạo thành một cụm (cluster). Kịch bản thử nghiệm này chia các node trong mạng thành các cụm nhằm minh họa trường hợp nêu trên. Khoảng cách giữa mỗi cụm và Border Router là khoảng 10 m (Khoảng cách chi tiết của từng node được mô tả ở bảng XX trong phần kết quả). Tất cả các phần cứng đều sử dụng antenna có sẵn trên SoC với công suất TX là 0 dBm.



Hình 70. Mô hình kết nối mạng đánh giá truyền tin đa chặng 1

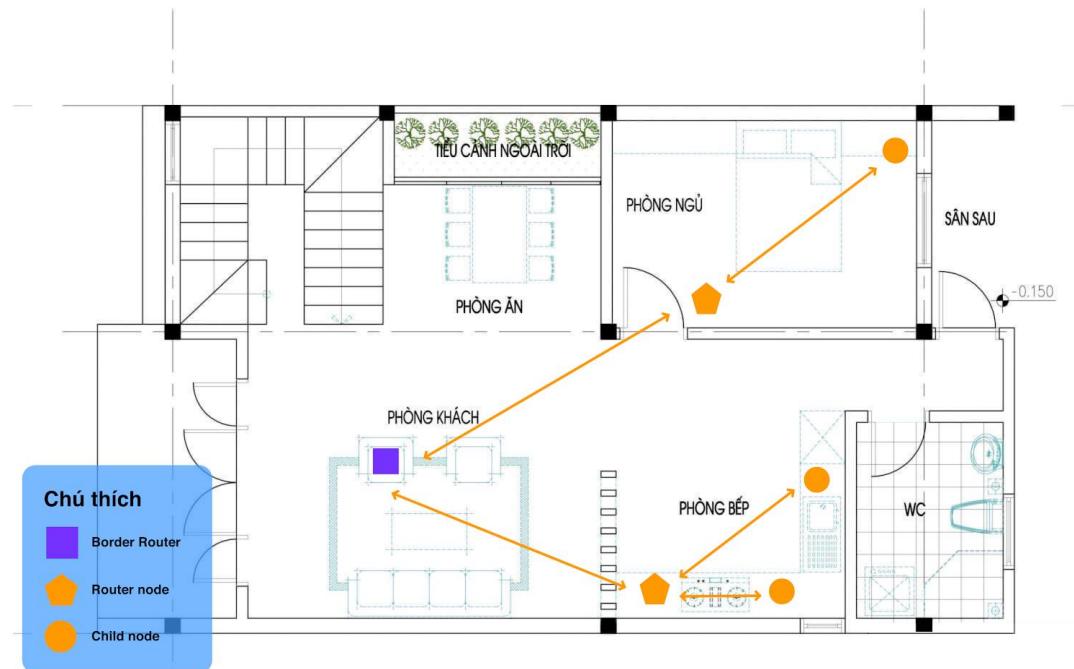
### 5.3.1.1 Môi trường thử nghiệm

- Thực hiện ngoài trời. Vật cản: Hầu như rất ít, chỉ có vài cây.
- Thực hiện trong nhà. Vật cản: tường gạch dày khoảng 10 cm và nhiều vật dụng như bàn, ghế và các đồ dùng trong nhà.

## ĐÁNH GIÁ KẾT QUẢ



Hình 71. Ảnh chụp thực tế quá trình thực hiện đo ngoài trời



Hình 72. Topology khi thực hiện đánh giá môi trường trong nhà

## ĐÁNH GIÁ KẾT QUẢ

### 5.3.1.2 Cách thức thực hiện

Thực hiện lệnh ping 1000 gói tin 64 bytes từ Border Router đến các node Router và node Child trong mỗi cụm.

```
> state
router
Done
> ipaddr
fd11:1111:1122:8:0:ff:fe00:ec00
fd11:1111:1122:8:7631:4357:84b6:8744
fe80:0:0:0:d049:1391:8fa2:5de7
Done
> rloc16
ec00
```

```
64 bytes from fd11:1111:1122::ff:fe00:ec00: icmp_seq=997 ttl=64 time=13.9 ms
64 bytes from fd11:1111:1122::ff:fe00:ec00: icmp_seq=998 ttl=64 time=16.8 ms
64 bytes from fd11:1111:1122::ff:fe00:ec00: icmp_seq=999 ttl=64 time=30.1 ms
64 bytes from fd11:1111:1122::ff:fe00:ec00: icmp_seq=1000 ttl=64 time=15.1 ms
^C
--- fd11:1111:1122:8:0:ff:fe00:ec00 ping statistics ---
1000 packets transmitted, 957 received, 4% packet loss, time 1002181ms
rtt min/avg/max/mdev = 13.595/18.729/43.796/4.213 ms
pi@raspberrypi:~ $ ^C
pi@raspberrypi:~ $ -
```

Hình 73. Kết quả thực hiện lệnh ping

### 5.3.1.3 Kết quả thu được

Bảng 17. Kết quả đánh giá truyền tin đa chặng ở môi trường không có vật cản

Cụm	Node	Firmware	Khoảng cách từ node đến BR (m)	Số hop về BR (hop)	RTT trung bình (ms)	PRR (%)
1	Router A	Thread-only	10	1	14.21	100
1	Child B	Thread-only	4	2	15.48	99
1	Child C	Thread-only	3	2	15.23	100
2	Router D	Dynamic multiprotocol	10	1	13.42	99
2	Child E	Dynamic multiprotocol	8	2	15.62	99

## ĐÁNH GIÁ KẾT QUẢ

Bảng 18. Kết quả đánh giá truyền tin đa chặng ở môi trường có vật cản

Cụm	Node	Firmware	Khoảng cách từ node đến BR (m)	Số hop về BR (hop)	RTT trung bình (ms)	PRR (%)
1	Router A	Thread-only	10	1	18.73	96
1	Child B	Thread-only	4	2	34.13	90
1	Child C	Thread-only	3	2	32.14	92
2	Router D	Dynamic multiprotocol	10	1	15.12	98
2	Child E	Dynamic multiprotocol	8	2	30.63	94

### 5.3.1.4 Nhân xét

- PRR của môi trường ngoài trời và môi trường trong nhà không có sự khác biệt đáng kể. RTT của môi trường trong nhà lớn hơn do có sự xuất hiện của nhiều vật cản, tuy nhiên, hệ thống vẫn hoạt động tốt ở môi trường trong nhà (PRR > 90% với tất cả các node).
- Tốc độ ổn định liên kết của SoC từ Nordic nhanh hơn so với Texas Instruments.
- Khoảng cách truyền không chỉ phụ thuộc vào công suất thu phát, độ lợi antenna, mà còn phụ thuộc rất nhiều vào yếu tố môi trường và độ cao bộ truyền nhận. Mặc dù cùng công suất TX (0 dBm) nhưng antenna của nRF52840 nhỉnh hơn so với các phần cứng còn lại trong mạch.

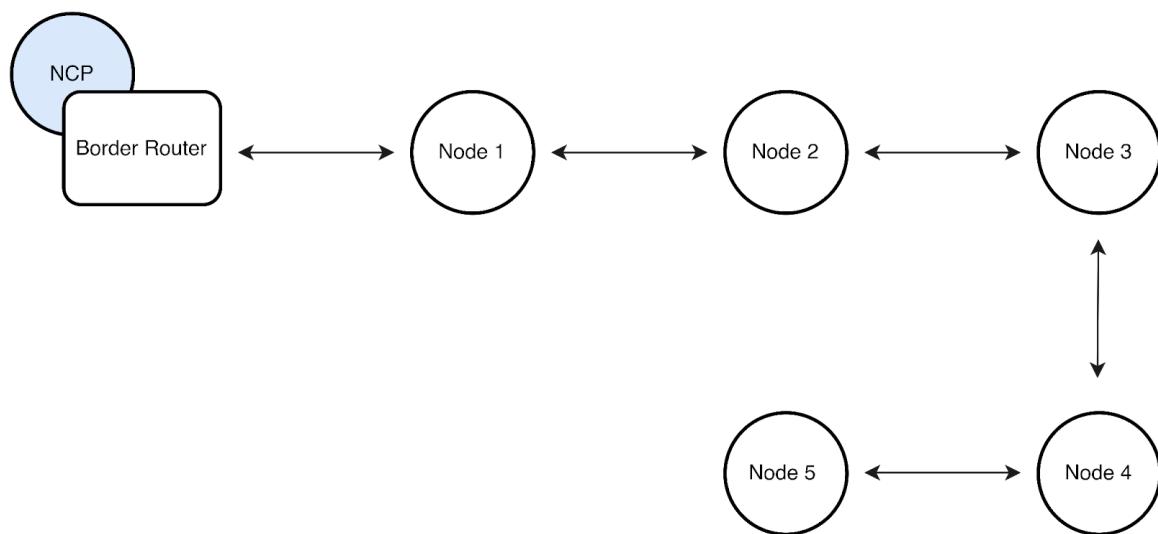
## 5.3.2 *Kịch bản thử nghiệm 2*

Thử nghiệm này nhằm đánh giá hiệu suất truyền tin đa chặng của giao thức mạng trong trường hợp ít nhiễu. Thông số kết quả quan tâm trong bài thử nghiệm này là PRR và RTT. Tất cả các phần cứng đều sử dụng antenna có sẵn trên SoC với công suất TX là 0 dBm.

### 5.3.2.1 Môi trường thử nghiệm:

## ĐÁNH GIÁ KẾT QUẢ

Thực hiện ngoài trời, hầu như không có vật cản (đường truyền Line-of-sight).



Hình 74. Mô hình kết nối mạng đánh giá truyền tin đa chặng 2

### 5.3.2.2 Cách thức thực hiện

Thực hiện lệnh ping 1000 gói tin 64 bytes từ Border Router đến tất cả các node trong mạng.

## ĐÁNH GIÁ KẾT QUẢ

### 5.3.2.3 Kết quả thu được

Bảng 19. Kết quả đánh giá truyền tin đa chặng 2

Node	Phần cứng	Firmware	Khoảng cách từ node đến BR (m)	Số hop về BR (hop)	RTT trung bình (ms)	PRR (%)
1	CC2538	Thread-only	10	1	14.23	100
2	nRF52840	Dynamic multiprotocol	20	2	19.43	100
3	CC2538	Thread-only	30	3	25.12	99
4	nRF52840	Dynamic multiprotocol	40	4	28.38	98
5	CC2538	Thread-only	50	5	35.94	96

### 5.3.2.4 Nhân xét

- Hệ thống đáp ứng ổn khi truyền qua 3 chặng với khoảng cách tối đa là 30m. Điều này chứng tỏ hệ thống hoạt động có tiềm năng hoạt động ổn định trong các ứng dụng nhà thông minh.
- Dynamic multiprotocol có thể một phần ảnh hưởng đến kết quả thu được của thử nghiệm này do có sự phân chia khe thời gian cho giao thức BLE (Giao thức BLE được ưu tiên hơn Thread). Để giảm thiểu tối đa ảnh hưởng, ta có thể tăng chu kỳ BLE advertising để tránh ảnh hưởng đến kết nối Thread.
- Khi truyền các gói tin qua nhiều chặng, multihop PRR giảm đáng kể. Tuy nhiên, ta có thể tăng công suất phát và sử dụng antenna ngoài trên các node Router. Từ đó tăng PRR từng chặng và tăng multihop PRR và vẫn đảm bảo các End node hoạt động ở công suất thấp.

## 5.4 Điều khiển qua MQTT

### 5.4.1 *Kịch bản thử nghiệm:*

Khi người dùng không ở trong vùng kết nối BLE, người dùng có thể cập nhật trạng thái của thiết bị trong mạng qua ứng dụng trên smartphone qua giao thức MQTT. Bài thử

## ĐÁNH GIÁ KẾT QUẢ

nghiêm mục đích đánh giá hoạt động của giao thức MQTT qua độ trễ (delay time) và Round-trip time (RTT) của gói tin.

### 5.4.2 Cách thực hiện

Thực hiện một script gửi 1000 gói tin (tần suất: 1 gói tin/giây, QoS = 1) từ Broker đến End node và ngược lại.

```
1 function random(min, max) {
2 | return Math.round(Math.random() * (max - min)) + min
3 }
4
5 function handlePayload(value) {
6
7 | let _value = value
8 | if (typeof value == 'string') {
9 | | _value = JSON.parse(value)
10 | }
11
12 | _value.led = random(0, 1)
13
14 | return JSON.stringify(_value, null, 1)
15 }
16
17 execute(handlePayload)
```

```
1 [2022-04-07 11:32:59] [INFO] APP init
2 [2022-04-07 11:33:13] [INFO] Testing was cleaned history connection messages
3 [2022-04-07 11:33:14] [INFO] MQTTX client with ID 7376bc08-1b6e-4e98-bb5f-e7181f761485 assigned
4 [2022-04-07 11:33:14] [INFO] Connect client Testing, MQTT/TCP connection: mqtt://mqtt.eclipseprojects.io:1883
5 [2022-04-07 11:33:15] [INFO] Testing connect success, MQTT.js onConnect trigger
6 [2022-04-07 11:47:54] [INFO] Testing set script successsed
7 [2022-04-07 11:48:06] [INFO] Testing sucessfully published message "{\n  \"led\": 0\n}" to topic "test"
8 [2022-04-07 11:48:06] [INFO] Testing opened timed message successfully, frequency(s): 1s
9 [2022-04-07 11:48:07] [INFO] Testing sucessfully published message "{\n  \"led\": 1\n}" to topic "test"
10 [2022-04-07 11:48:08] [INFO] Testing sucessfully published message "{\n  \"led\": 1\n}" to topic "test"
11 [2022-04-07 11:48:09] [INFO] Testing sucessfully published message "{\n  \"led\": 1\n}" to topic "test"
12 [2022-04-07 11:48:10] [INFO] Testing sucessfully published message "{\n  \"led\": 0\n}" to topic "test"
```

Hình 75. MQTT script trên phần mềm MQTTX

### 5.4.3 Kết quả thu được

Bảng 20. Kết quả thử nghiệm điều khiển qua MQTT

Giao thức	PRR	Delay time	RTT
Thread	94%	175 ms	350 ms
Zigbee	92%	180 ms	360 ms

### 5.4.4 Nhận xét

## ĐÁNH GIÁ KẾT QUẢ

Kết quả thu được phụ thuộc phần lớn vào tốc độ Internet. Nhìn chung, hệ thống vẫn đảm bảo được tính soft real-time của hệ thống

### 5.5 Điều khiển nội bộ

#### 5.5.1 Kịch bản thử nghiệm

Kịch bản thử nghiệm này giả lập trường hợp người dùng không có kết nối internet để điều khiển các node qua MQTT, họ có thể điều khiển bằng BLE/Thread forwarding service trên điện thoại thông minh. Qua thử nghiệm này, ta có thể đánh giá được hiệu suất xử lý của node Forwarder. Thông số quan tâm trong thử nghiệm này là PRR và RTT

#### 5.5.2 Môi trường thực hiện

Thực hiện trong nhà. Vật cản: tường gạch dày khoảng 10 cm và nhiều vật dụng như bàn, ghế và các đồ dùng trong nhà.

#### 5.5.3 Cách thức thực hiện

Thử nghiệm được thực hiện bằng cách gửi 1000 gói chuyển tiếp từ thiết bị BLE đến Forwarder (một gói mỗi giây). Sau đó Forwarder sẽ chuyển tiếp những thông điệp đó đến một node End node lân cận (1 hop). RTT sẽ là RTT trung bình của 1000 gói tin.

#### 5.5.4 Kết quả thu được

Bảng 21. Kết quả thử nghiệm nội bộ

Kết nối	PRR	RTT
Smartphone -> Forwarder: 5m	100%	6.2 ms
Forwarder -> Thread: 10m	96%	15.3ms

#### 5.5.5 Nhận xét

Ở khoảng cách ngắn, giao thức BLE hoạt động rất tốt, PRR hầu như là 100% nên tính BLE/Thread forwarding service phụ thuộc phần lớn vào liên kết mạng Thread. Vì có

## ĐÁNH GIÁ KẾT QUẢ

vật cản là tường gạch nên dù ở khoảng cách 10m, PRR của kết nối giữa forwarder và thread vẫn giảm đáng kể.

### 5.6 Đo công suất tiêu thụ và dự đoán thời gian hoạt động

#### 5.6.1 Kích bản thử nghiệm

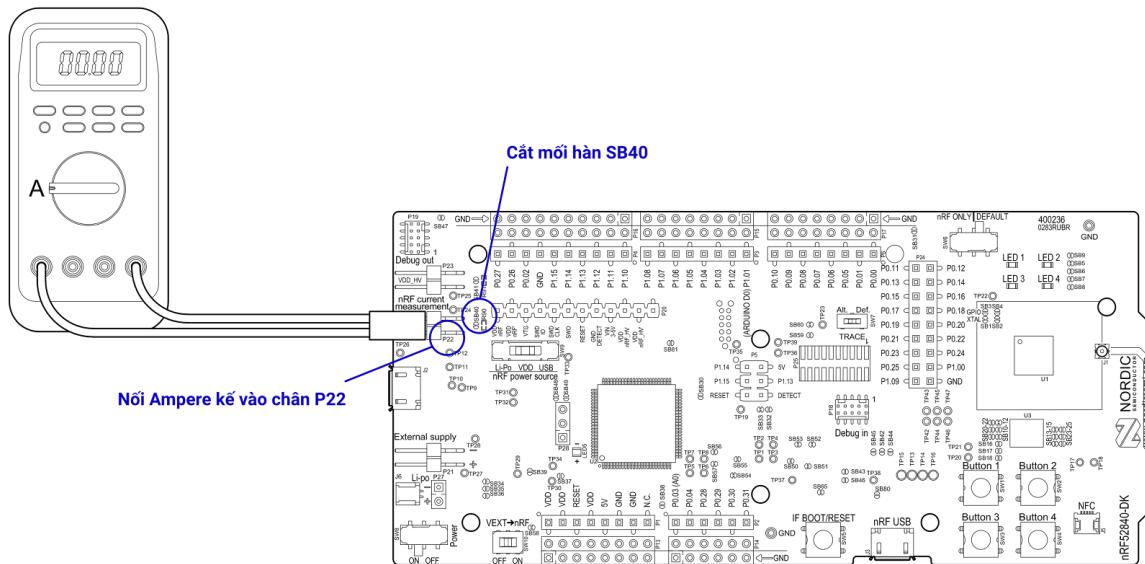
Bài thử nghiệm mục đích đo công suất của node mạng khi ở trạng thái hoạt động bình thường, từ đó dự báo thời gian hoạt động của từng node mạng. Thông số quan tâm trong bài thử nghiệm: Dòng hoạt động trung bình (mA)

#### 5.6.2 Cách thức thực hiện

- Nạp chương trình cho kit nRF52840 (Firmware node Forwarder hoặc firmware End node)
- Cắt mối hàn SB40 như hình vẽ. Lưu ý, lúc này không thể nạp chương trình cho kit được nữa.
- Nối ampere kế vào chân P22 để tiến hành đo dòng
- Uớc tính thời gian hoạt động với thông số được cung cấp từ Nordic cho nRF52840 DK và cấu hình mặc định của firmware.

Bảng 22. Thông số cơ bản cung cấp bởi Nordic

Điện trở (VDD)	3V
Trạng thái ngoại vi	Idle
Chu kỳ BLE advertising	100 ms
Dung lượng pin (CR2032)	200 mA



Hình 76. Mô tả quá trình đo dòng của thử nghiệm

### 5.6.3 Kết quả thu được

Bảng 23. Kết quả thử nghiệm dự đoán thời gian hoạt động

Node	Trạng thái Idle	Tổng dòng trung bình	Ước tính thời gian hoạt động
End node	2.7 uA	42 uA	4600 giờ ~ 190 ngày
Forwarder	3.1 uA	215 uA	900 giờ ~ 36 ngày

### 5.6.4 Nhận xét

Từ bảng kết quả nhận thấy, do phải xử lý nhiều tác vụ hơn các end node, forwarder tiêu thụ nhiều năng lượng hơn. Một số giải pháp đề xuất:

- Node forwarder vai trò chính là điều hướng các gói tin và mạng Thread có thể thăng cấp end node thành forwarder trong trường hợp forwarder hết năng lượng.
- Kết nối forwarder với các nguồn ngoài (Border Router hoặc nguồn pin) để tăng thời gian hoạt động, như vậy sẽ loại bỏ sự phụ thuộc của forwarder lên nguồn pin cell.

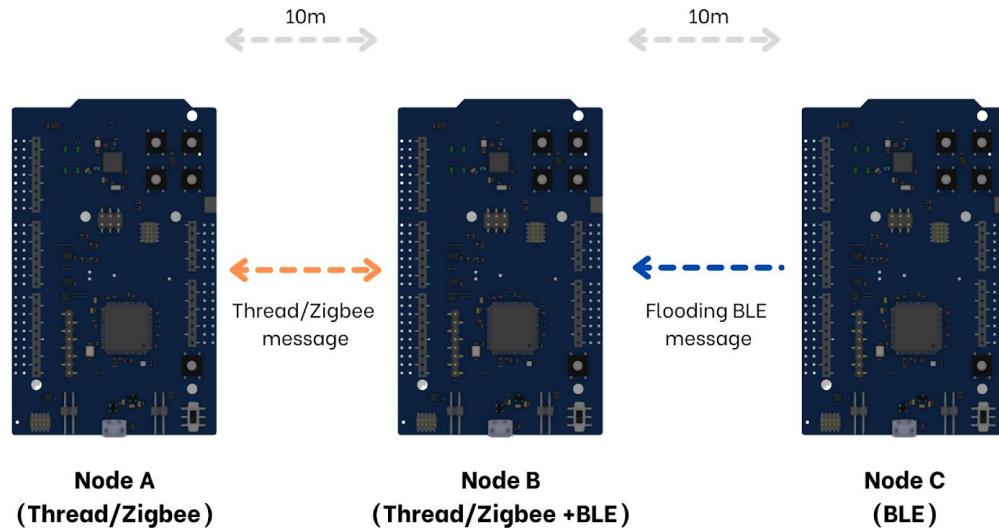
## 5.7 Đánh giá độ hiệu quả của ứng dụng đa giao thức

### 5.7.1 Kịch bản thử nghiệm

## ĐÁNH GIÁ KẾT QUẢ

Kịch bản thử nghiệm này nhằm đánh giá độ hiệu quả của ứng dụng đa giao thức. Vấn đề dễ nhận thấy khi chạy đa giao thức là hiện tượng mất gói tin của giao thức còn lại. Kịch bản này sử dụng ba node mạng:

- Node A chỉ chạy giao thức Thread/Zigbee
- Node B chạy đa giao thức Thread và BLE hoặc Zigbee và BLE.
- Node C chỉ chạy giao thức BLE



### 5.7.2 Môi trường thử nghiệm

- Khoảng cách giữa các node với nhau là 10m (đường truyền Line-of-sight)
- Sử dụng antenna có sẵn trên SoC với công suất TX là 0 dBm.

### 5.7.3 Cách bước thực hiện

## ĐÁNH GIÁ KẾT QUẢ

**Node A:** Tạo mạng Thread/Zigbee trên node A.

**Node B:** Thực hiện kết nối node B vào mạng. Sau đó, tìm kiếm kết nối BLE.

- Với kịch bản 1, kết nối với Node C
- Với kịch bản 2, để node B ở chế độ tìm kiếm kết nối

**Node C:** Cấu hình liên tục gửi gói tin BLE connection/advertising với tần suất 100ms/gói đến node B.

```
COM5 (Silicon Labs CP210x USB to UART Bridge (COM5))
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages
Quick connect...
>
>
>
>
==== Test Finished ====
Test duration: 85822ms

Average CPU utilization:
  Local: 100.00%
  Remote: 100.00%

Without retransmissions:
  PER: 34.30%
  Throughput: 5 kbps

With retransmissions:
  PER: 0.00%
  Throughput: 3 kbps
```

Hình 78. Kết quả thực hiện của thử nghiệm đánh giá ứng dụng đa giao thức

### 5.7.4 Kết quả thực hiện

Bảng 24. Kết quả thu được của thử nghiệm đánh giá ứng dụng đa giao thức

	RTT	PRR (without retransmissions)	PRR (with retransmissions)
Kịch bản 1	85.8 ms	65.7%	100%
Kịch bản 2	82.3 ms	80.2%	100%

### 5.7.5 Nhận xét

- Trường hợp truyền gói tin không sử dụng ACK, số lượng gói tin bị mất là rất lớn. Do đó việc sử dụng ACK xác nhận sẽ giải quyết được điều này.
- Nhược điểm của kỹ thuật sử dụng ACK là tăng năng lượng tiêu hao và giảm throughput của liên kết. Tuy nhiên, đối với hệ thống nhóm đang thực hiện, giao thức BLE chỉ được sử dụng trong trường hợp Gateway gấp sự cố.

## CHƯƠNG 6. KẾT LUẬN VÀ PHƯƠNG HƯỚNG

### 6.1 Kết luận

Kết quả đạt được:

- Nghiên cứu đã thu thập các thông số của mạng Thread, Zigbee và BLE dựa trên System-on-Chip Nordic Semiconductor nRF52840. Ngoài ra, nghiên cứu đã mở rộng trên phần cứng của CC2538 (Texas Instrument) và EFR32 (Silabs), thực hiện các kịch bản đánh giá trong môi trường hoạt động lý tưởng, thực tế.
- Nghiên cứu đã thực hiện phần mềm nhúng điều khiển đèn LED dựa trên nền tảng System-on-Chip Nordic Semiconductor nRF52840 hỗ trợ đa giao thức động (multi-protocol). Đây là nền tảng để tiếp tục phát triển các ứng dụng phức tạp như báo cháy, chống trộm, ...
- Nghiên cứu đã xây dựng thành công Forwarding service, nghĩa là gửi gói tin BLE qua mạng Thread (Thread packet over Bluetooth Low Energy).
- Xây dựng ứng dụng Android để smartphone tương tác với các node trong mạng Thread thông qua giao thức Bluetooth Low Energy

### 6.2 Phương hướng phát triển

- Tích hợp thêm giao thức Bluetooth Mesh để thành lập mạng hỗ trợ được ba giao thức mạng Mesh đang phổ biến trên thị trường hiện nay (Thread, Bluetooth mesh và Zigbee)
- Thực hiện test trên nhiều phần cứng hơn để kiểm tra sự tương thích và khả năng ứng dụng thực tế của giao thức IoT.

## TÀI LIỆU THAM KHẢO

- [1] Jin-Shyan Lee, Yuan-Ming Wang, “Experimental Evaluation of ZigBee-Based Wireless Networks in Indoor Environments,” *Journal of Engineering*, January 2013.
- [2] I. SA, “IEEE Standard for Low-Rate Wireless Networks,” 2020. [Trực tuyến]. Available: <https://standards.ieee.org/ieee/802.15.4/7029/>. [Đã truy cập 31 May 2022].
- [3] Kibum Choi, Yunmok Son, Juhwan Noh, Hocheol Shin, “Dissecting Customized Protocols: Automatic Analysis for Customized Protocols based on IEEE 802.15.4,” trong *The 9th ACM Conference*, 2016.
- [4] M. Ladan, “The Next Generation Internet Protocol, IPV6: An Overview,” 2009.
- [5] Google, “IPv6 Addressing,” Google, 2021. [Trực tuyến]. Available: <https://openthread.io/guides/thread-primer/ipv6-addressing>. [Đã truy cập 31 May 2022].
- [6] J. Olsson, “6LoWPAN demystified,” Texas Instruments, [Trực tuyến]. Available: <https://www.ti.com/lit/wp/swry013/swry013.pdf>. [Đã truy cập 31 May 2022].
- [7] IETF, “User Datagram Protocol, RFC 768,” Internet Standard, 28 August 1980. [Trực tuyến]. Available: <https://datatracker.ietf.org/doc/html/rfc768>. [Đã truy cập 31 May 2022].
- [8] T. Group, “Thread Technical White Paper,” 13 July 2015. [Trực tuyến]. Available: [https://www.threadgroup.org/Portals/0/documents/support/CommissioningWhitePaper\\_658\\_2.pdf](https://www.threadgroup.org/Portals/0/documents/support/CommissioningWhitePaper_658_2.pdf). [Đã truy cập 31 May 2022].
- [9] I. Unwala, “Thread: An IoT Protocol,” trong *IEEE Green Technologies Conference (IEEE-Green)*, 2018.

## TÀI LIỆU THAM KHẢO

---

- [10] Nisha Ashok Somani, Yask Patel, “Zigbee: A Low Power Wireless Technology for Industrial Applications,” trong *International Journal of Control Theory and Computer Modeling* 2, 2012.
- [11] S. Labs, “Zigbee Fundamentals,” 2017. [Trực tuyến]. Available: <https://www.silabs.com/documents/public/user-guides/ug103-02-fundamentals-zigbee.pdf>. [Đã truy cập 31 May 2022].
- [12] Z. Alliance, “Zigbee Specification,” 5 August 2015. [Trực tuyến]. Available: <https://zigbeealliance.org/wp-content/uploads/2019/11/docs-05-3474-21-0csg-zigbee-specification.pdf>. [Đã truy cập 31 May 2022].
- [13] Ahmed Fathy, Ibrahim F. Tarrad, Hesham F. A. Hamed & Ali Ismail Awad, “Advanced Encryption Standard Algorithm: Issues and Implementation Aspects,” *Advanced Machine Learning Technologies and Applications*, số AMLTA 2012, pp. 516 - 523, 2012.
- [14] Lindsey N. Whitehurrst, Todd R. Andel, J. Todd McDonald, “Exploring security in ZigBee networks,” trong *CISR* 14, 2014.
- [15] W. Telecom, “ZigBee,” trong *The Internet of Things: Key Applications and Protocols*, Wiley, 2012, pp. 93 - 137.
- [16] C. Gezer, “A ZigBee Smart Energy Implementation for Energy Efficient Buildings,” trong *IEEE Conference on Vehicular Technology (VTC)*, 2011.
- [17] D. Gislason, *Zigbee Wireless Networking*, 2008.
- [18] Meena Singh, M.A. Rajan, V.L. Shivraj, P. Balamuralidhar, “Secure MQTT for Internet of Things (IoT),” trong *International Conference on Communication Systems and Network Technologies (CSNT)*, 2015.
- [19] Urs Hunkeler, Hong Linh Truong, Andy Stanford-Clark, “MQTT-S — A publish/subscribe protocol for Wireless Sensor Networks,” trong *International Conference on Communication Systems Software and Middleware (COMSWARE)*, 2008.

## TÀI LIỆU THAM KHẢO

---

- [20] M. U. H. A. Rasyid, “Implementation MQTT-SN Protocol on Smart City Application based Wireless Sensor Network,” trong *International Conference on Science in Information Technology (ICSI Tech)*, 2019.
- [21] C. Bormann, “CoAP: An Application Protocol for Billions of Tiny Internet Nodes,” *IEEE Internet Computing*, tập 16, số 2, pp. 62 - 67, 2012.
- [22] D. Thangavel, “Performance evaluation of MQTT and CoAP via a common middleware,” trong *Intelligent Sensors, Sensor Networks and Information Processing Conference (ISSNIP)*, 2014.
- [23] B. S. I. Group, “Bluetooth Core Specification 5.1,” US, 2019.
- [24] M. Afaneh, “Intro to Bluetooth Low Energy,” Novel Bits, 2019.
- [25] N. Semiconductor, “Multiprotocol support with BLE/Bluetooth,” Nordic Semiconductor, 25 June 2019. [Trực tuyến]. Available: [https://infocenter.nordicsemi.com/topic/sdk\\_tz\\_v3.1.0/ble\\_154\\_multiprotocol.html](https://infocenter.nordicsemi.com/topic/sdk_tz_v3.1.0/ble_154_multiprotocol.html). [Đã truy cập 20 December 2021].